EACL 2023

The 17th Conference of the European Chapter of the Association for Computational Linguistics

Proceedings of the Conference

May 2-6, 2023

The EACL organizers gratefully acknowledge the support from the following sponsors.

Diamond and Welcome Event

grammarly

Diamond

LIVEPERSON

Platinum and D&I Ally

amazon | science

Platinum

Bloomberg

Engineering

Silver

duolingo

Bronze

Adobe

babelscape

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
`acl@aclweb.org`

ISBN 978-1-959429-44-9

Message from the General Chair

Welcome to the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL). This is the flagship European conference dedicated to European and international researchers, covering a broad spectrum of research areas of Computational Linguistics and Natural Language Processing.

Organizing a scientific conference of the prestige and size of EACL is always a great honor associated with several challenges. Our team had to tackle unusual complexities: this conference was one of the first scheduled to be in person after the long period of online conferences forced by COVID pandemic. The bidding process for a location, which typically takes place several years before the actual start of the conference, is mainly driven by the aim of expanding and involving the science community of all European countries: EACL selected Kyiv, Ukraine, as the physical location. As you all know, in February 2022, an unpredictable and dramatic event happened, the war between Russian and Ukraine, which made the organization in Kyiv impossible.

Considering the importance of physical interaction among researchers, especially after the restrictions imposed by the COVID pandemic, we worked hard with the EACL and ACL boards to find an alternative location, able to delight our attendees. Our team achieved this seemingly impossible goal of organizing a conference in a new location a few months before its start: we selected Dubrovnik, Croatia, while preserving the original aim of strengthening the connection with the Ukrainian community. In this respect, the Ukraine local committee will feature a dedicated panel session, “Low-resource languages in NLP products”, and a workshop to highlight work on Ukrainian language technologies. Following the latest conference, EACL 2023 will be “hybrid,” serving both virtual and in-person participants. As our official local chairs are not from the physical location, we needed a local team from Croatia for helping with the logistics. As a result, the main unexpected novelty of EACL 2023 is to have two local organizing committees from two different European countries.

In the remainder of this preface, I would like to thank EACL contributors chronologically with respect to my work timeline for EACL: Roberto Basili and Shuly Wintner, the new and former Presidents of ACL, along with the EACL board – thanks for having trusted me to manage the organization of the conference in rather complicated times. I started to be confident that we would have done a good job after Isabelle Augenstein and Andreas Vlachos accepted the role of PC Chairs. They have performed amazing work, creating an outstanding program, and also helping me in recruiting our fantastic organization team. A special thank is due to Preslav Nakov (EACL officer) for his support: thanks to his action, the proactiveness of David Yarowsky, and the fairless effort of Jennifer Rachford (our new secretary of the ACL business office), we successfully implemented the apparently unrealistic idea of switching from the already planned online conference to a hybrid setting with a physical location in Dubrovnik. Regarding the online side of our hybrid conference, we partnered with Underline (Sol Rosenberg, Damira Mrsic and Luka Simic), who also gave us support for managing the entire conference. While finalizing the location, we started to activate the different sections of the conference, for which my acknowledgements are again in chronological order:

- Ukraine Local Committee, Viktoria Kolomiets, Mariana Romanyshyn, Oleksii Molchanovskyi, Oles Dobosevych, was instrumental in preserving our initial goal of connecting the Ukraine research community, organizing a panel and a workshop.

- The website chairs, Pepa Atanasova and Julius Cheng, started immediately to design our website, even when almost no information was available.

- The workshop chairs, Zeerak Talat and Antonio Toral, selected our conferences and led the selection of workshops for the joint ACL call.

- The tutorial chairs, Sameer Pradhan and Fabio Massimo Zanzotto, together with the ACL chairs, took care of the tutorial selection for the ACL related conferences.

- The demonstration chairs, Danilo Croce and Luca Soldaini, created a parallel conference program to select exciting demos.

- The Publicity Chairs, Laura Biester, Leshem Choshen and Joel Tetrault, have been our interface with the science community through social media platforms.

- The Publication Chairs, Carolina Scarton and Ryan Cotterell, produced high-quality proceedings, thanks to their competence and experience.

- The diversity and inclusion chairs, Sara Tonelli, Elena Cabrio, Verena Rieser, Spandana Gella, took care of DI and performed an amazing job, also working on hundreds applications.

- The Local Organising Committee of Croatia, Marko Tadić, Krešimir Šojat, and Daša Farkaš, gave essential help for the logistics, Visa, and student volunteers.

- Student Research Workshop Chairs, Matthias Lindemann, Alban Petit, and Elisa Bassignana, along with their faculty advisors Valerio Basile and Natalie Schluter, helped in setting the bases for forming great NLP researchers of the future.

- Our entire program committee, Senior Area Chairs, Area Chairs, reviewers, and best paper committee, was essential for obtaining our high-quality scientific program.

- The ACL's sponsorship director Chris Callison-Burch took care of our sponsorships.

- The student volunteers, as usual, are essential for a successful conference execution.

- Priscilla Rasmussen, our former ACL business office secretary, continued to provide us with useful advice.

Finally, I would like to thank our sponsors for helping us to fund scholarships and DI initiatives.

Alessandro Moschitti
Amazon Alexa AI, Los Angeles, USA
EACL 2023 General Chair

ACL Statement on the Ukraine situation

March 11, 2022

The Association for Computational Linguistics (ACL) condemns in the strongest possible terms the actions of the Russian Federation government in invading the sovereign state of Ukraine and engaging in war against the Ukrainian people. We stand together with Ukrainian NLP colleagues, the Ukrainian people, Russian NLP colleagues and Russian people who condemn the actions of the Russian Federation government, and all those around the world who have been impacted by the invasion.

As a small token of our solidarity with the Ukrainian people, the ACL has decided to temporarily sever its ties with Russia-based organizations, while at the same time allowing Russian scientists to remain part of the ACL community. In practice, this means that the ACL will refrain from accepting any sponsorship or allowing any exhibits from Russian-headquartered entities at ACL-run events. Russian scholars are still welcome to participate in ACL events and publish at ACL venues.

The ACL is committed to peace and condemns any form of violence and harassment. We are also committed to peaceful co-operation, mutual understanding, and tolerance across borders. NLP scholars from both Ukraine and Russia are welcome to get in touch with the ACL with any concerns.

Tim Baldwin, on behalf of the ACL Executive

Message from the Program Chairs

Welcome to the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL). After the last edition in 2021 having been held fully online due to the COVID pandemic, EACL 2023 is being held in "hybrid" mode this year, serving both virtual and in-person participants in Dubrovnik, Croatia. While the original plan was to hold the conference in Kyiv (which was the plan originally for EACL 2021), the ongoing war made the organisation in Ukraine impossible. In order to ensure that the original aim of strengthening the connections with the Ukrainian community is still served, our program features a dedicated session and a workshop to highlight work on Ukrainian language technologies.

Submission and Acceptance

EACL 2023 accepted direct submissions, as well as submissions via ARR. For direct submissions, abstracts were needed to be registered one week prior to the submission date.

In total, EACL 2023 received 1550 submissions, the largest number to date, with the 2021 edition having received 1400 submissions. Out of those, 1045 were long and 505 were short paper submissions. 81 were ARR papers that were committed to EACL. 249 submissions were withdrawn throughout the reviewing process, including before the full paper submission deadline. 55 papers were desk rejected for various reasons (missing the limitations section, anonymity policy, multiple submission policy, plagiarism or formatting violations).

By the time we as the programme chairs made acceptance decisions, 1166 submissions were still active in the system. We kept the acceptance rate in line with previous *ACL conferences, resulting in 281 papers accepted to the main conference (24.1%), and 201 papers accepted to the Findings of EACL (17.2%), with the remaining 58.7% being rejected. One paper accepted to the main conference and four papers accepted to Findings were subsequently withdrawn. Out of the final set of accepted main conference papers, we invited 178 to be presented orally, and all 281 papers accepted to the main conference to be presented during in-person sessions, as well as a plenary virtual poster session. The EACL 2023 program also features six papers from the Transactions of the Association for Computational Linguistics (TACL) journal, and one from the Computational Linguistics (CL) journal.

Limitations Section

Following EMNLP 2022, we required that each submitted paper must include an explicitly named Limitations section, discussing the limitations of the work. This was to counterbalance the practice of over-hyping the take-away messages of papers, and to encourage more rigorous and honest scientific practice. This discussion did not count towards the page limit, and we asked reviewers to not use the mentioned limitations as reasons to reject the paper, unless there was a really good reason to.

Areas

To ensure a smooth process, the submissions to EACL 2023 were divided into 21 areas. The areas mostly followed these of previous EACL, and more broadly *ACL conferences, reflecting the typical divisions in the field. We also had a special area for papers for which both SACs had a conflict of interest. Those papers were reviewed by the reviewers and ACs in their original areas, but the paper recommendations were made by a dedicated SAC, who was a senior member of the NLP community. The most popular areas with over 100 submissions were "Generation and Summarization", "Language Resources and Evaluation", and "Machine Learning in NLP".

Best Paper Awards

From the papers submitted to EACL 2023, we selected 25 papers accepted to the main conference as candidates for a Best Paper award, based on nominations by the reviewers. These papers were assessed by the Best Paper Award Committee, who also determined the types of paper awards, following the ACL Conference Awards Policy. The selected best papers and runner-ups will be announced in a dedicated plenary session for Best Paper Awards on 4 May 2023.

Programme Committee Structure and Reviewing

Similar to prior NLP conferences, we adopted the hierarchical program committee structure, where for each area we invited 1-2 Senior Area Chairs (SACs), who worked with a team of Area Chairs (ACs), and a larger team of reviewers. We relied on statistics from prior years to estimate how many SACs, ACs and reviewers would be needed and ended up with 43 SACs, 118 ACs and 1634 reviewers. For identifying ACs and reviewers, we used the reviewer lists from prior *ACL conferences, and also encouraged all EACL 2023 authors to serve as reviewers, using a mandatory form requesting further information on their ability to serve as ACs, reviewers or emergency reviewers, which authors had to fill in on Softconf when registering their abstracts. We passed this information on to SACs, who were responsible for recruiting ACs and reviewers.

Rather than making assignments using a matching algorithm, we asked ACs and reviewers to bid on registered abstracts within their areas, to achieve a better fit. We went with this solution as the number of papers per area was relatively small, and we wanted to avoid poor reviewing assignments as much as possible. We then made an initial paper assignment, in which we ensured that each paper would be reviewed by at least one reviewer who bidded "yes" for the submission, and by no reviewers who bidded "no" for the submission.

Afterwards, we asked the SACs to fine-tune the allocations, and ensure each paper had one AC and three reviewers assigned to it.

To ensure the review quality, we provided detailed guidelines about what reviewers should and shouldn't do in a review, based on the EMNLP 2022 guidelines. We also asked reviewers to flag papers for potential ethical concerns.

For pre-reviewed ARR papers, we asked SACs to not rely mainly on the reviewer scores, but to make their recommendations based on the text of the reviews, meta-reviews and the papers themselves. For making acceptance decisions, we mostly followed SAC recommendations, though also taking into account the overall quality of papers submitted to the conference. Where recommendations seemed overly harsh or lenient given the reviewers' scores, reviews, author responses, or discussions amongst reviewers, we engaged in a dialogue with the respective SACs to make the final decision about the papers in question.

Ethics Committee

We also formed an Ethics Committee (EC) dedicated to ethical issues. The ethics committee considered 21 papers that were flagged by the technical reviewing committee for ethical concerns. Out of these, 10 were conditionally accepted, meaning the ethics issues had to be addressed in the camera-ready version, to be verified by the EC prior to final acceptance, and the other 11 were accepted as is. The authors of all conditionally accepted papers submitted the camera-ready version and a short response that explained how they had made the changes requested by the EC. The EC double-checked these revised submissions and responses, and confirmed that the ethical concerns had been addressed. As a result, all conditionally accepted papers were accepted to the main conference or Findings.

ACL Rolling Review

ACL Rolling Review (ARR) is an initiative of the Association for Computational Linguistics, where the reviewing and acceptance of papers to publication venues are done in a two-step process: (1) centralized rolling review and (2) the ability to commit the reviewed papers to be considered for publication by a publication venue. For EACL 2023, we decided to follow EMNLP 2022's example and run a process which is separate from ARR, but also allows for ARR submissions. Specifically, authors could either submit papers to EACL 2023 directly, or commit ARR reviewed papers by a certain date. We coordinated with the ARR team to extract the submission, review and meta-review from the OpenReview system, according to a submission link that the author provided when committing their ARR submission to EACL. The ARR commitment deadline was set one month after the direct submission deadline since the ARR submissions already have their reviews and meta-recommendation. These ARR papers were then ranked by the SACs together with the direct submissions in the track, and based on the reviews and meta-reviews from ARR. Overall, EACL had 81 papers committed from ARR, of these 24 were accepted to the main conference and 20 were accepted to Findings of EACL.

Presentation Mode

We made the decision on which papers would be invited for oral poster presentations based on several factors: the relative rank of the paper according to SAC recommendation, whether the paper had been recommended for a best paper award by at least one reviewer, and for TACL and CL papers, the authors' preference of presentation mode.

Keynotes and Panel

Another highlight of our program are the plenary sessions, for which we scheduled three talks, as well a panel:

- a keynote talk by Joyce Chai (University of Michigan) on "Language Use in Embodied AI!
- a keynote talk by Edward Greffenstette (Cohere AI and University College London) on "Going beyond the benefits of scale by reasoning about data"
- a keynote talk by Kevin Munger (Penn State University) on Chatbots for Good and Evil"
- a panel on "low-resource languages in NLP products" led by Mariana Romanyshyn with Viktoria Kolomiets (Grammarly), Mariana Romanyshyn (Grammarly), Oleksii Molchanovskyi (Ukrainian Catholic University) and Oles Dobosevych (Ukrainian Catholic University)

Thank Yous

EACL 2023 is the result of a collaborative effort and a supportive community, and we want to acknowledge the efforts of so many people with whom we worked directly and made significant efforts in putting together the programme for EACL 2023!

- Our General Chair, Alessandro Moschitti, who led the whole organising team, and helped with many of the decision processes;
- Our 43 Senior Area Chairs, who were instrumental in every aspect of the review process, from recruiting Area Chairs, correcting reviewer assignments, to making paper acceptances;
- Our 118 Area Chairs, who had the role of interacting with the reviewers, leading paper review discussions, and writing meta-reviews;

- The 1634 reviewers, who provided valuable feedback to the authors; The emergency reviewers, who provided their support at the last minute to ensure a timely reviewing process;
- Our Best Paper Selection Committee, who selected the best papers and the outstanding papers: Jonathan Kummerfeld (chair), Joakim Nivre, Bonnie Webber, Thamar Solorio and Hanna Hajishirzi;
- Our Ethics Committee, chaired by Zeerak Talat, for their hard work to ensure that all the accepted papers addressed the ethical issues appropriately, under a very tight schedule;
- Our amazing Publication Chairs, Carolina Scarton and Ryan Cotterell for compiling the proceedings in good time for the conference;
- Our Publicity Chairs, Laura Biester, Leshem Choshen and Joel Tetrault, for their work on managing the communications on social media platforms;
- Our website chairs, Pepa Atanasova and Julius Cheng for putting together the website for the conference and keeping it up to date;
- Damira Mrsic from Underline, for her support in developing the virtual conference platform;
- Jennifer Rachford, who has worked tirelessly online and on-site to ensure that EACL 2023 is a success.

We’re looking forward to a great EACL 2023!

Isabelle Augenstein (University of Copenhagen, Denmark)
Andreas Vlachos (University of Cambridge, UK)
EACL 2023 Programme Committee Co-Chairs

Organizing Committee

General Chair

Alessandro Moschitti, Amazon Alexa

Program Chairs

Isabelle Augenstein, University of Copenhagen
Andreas Vlachos, University of Cambridge

Publications Chairs

Ryan Cotterell, ETH Zürich
Carolina Scarton, University of Sheffield

Workshop Chairs

Zeerak Talat, Simon Fraser University
Antonio Toral, University of Groningen

Tutorials Chairs

Sameer Pradhan, University of Pennsylvania
Fabio Massimo Zanzotto, University of Rome, “Tor Vergata”

Demonstrations Chairs

Danilo Croce, University of Rome, “Tor Vergata”
Luca Soldaini, Allen Institute for AI

Publicity Chairs

Joel Tetreault, Dataminr
Leshem Choshen, IBM AI research; Hebrew University of Jerusalem
Laura Biester, University of Michigan

Website Chairs

Pepa Atanasova, University of Copenhagen
Julius Cheng, University of Cambridge

Sponsorship Director

Chris Callison-Burch, University of Pennsylvania

Diversity and Inclusion Chairs

Elena Cabrio, Université Côte d’Azur, Inria, CNRS, I3S

Sara Tonelli, Fondazione Bruno Kessler
Verena Rieser, Heriot-Watt University
Spandana Gella, Amazon Alexa

Student Research Workshop Chairs

Matthias Lindemann, University of Edinburgh
Alban Petit, Université Paris-Saclay
Elisa Bassignana, IT University of Copenhagen

Student Research Workshop Faculty Advisors

Valerio Basile, University of Turin
Natalie Schluter, IT University of Copenhagen; Apple

Local Organising Committee

Marko Tadić, University of Zagreb
Krešimir Šojat, University of Zagreb
Daša Farkaš, University of Zagreb

Ukraine Local Committee

Viktoria Kolomiets, Grammarly
Mariana Romanyshyn, Grammarly
Oleksii Molchanovskyi, Ukrainian Catholic University
Oles Dobosevych, Ukrainian Catholic University

Program Committee

Anaphora, Discourse and Pragmatics

Bonnie Webber, University of Edinburgh
Michael Strube, Heidelberg Institute for Theoretical Studies

Computational Social Science and Social Media

Maria Liakata, Queen Mary University of London
Kalina Bontcheva, University of Sheffield

Conflicts of Interests

Joakim Nivre, Research Institutes of Sweden

Dialogue and Interactive Systems

Diarmuid Ó Séaghdha, Apple
Matthew Purver, Queen Mary University of London

Document analysis, Text Categorization and Topic Models

Nikolaos Aletras, University of Sheffield
Ekaterina Shutova, University of Amsterdam

Ethical and Sustainable NLP

Nafise Sadat Moosavi, Department of Computer Science, The University of Sheffield
Yonatan Belinkov, Technion

Ethics Review

Zeerak Talat, Simon Fraser University

Generation and Summarization

Ondrej Dusek, Charles University
Chenghua Lin, Department of Computer Science, University of Sheffield

Information Extraction

Roberto Navigli, Sapienza University of Rome
Mrinmaya Sachan, ETH Zurich

Information Retrieval and Search

Bruno Martins, IST and INESC-ID
Fabrizio Silvestri, Sapienza, University of Rome

Interpretability and Model Analysis

Dong Nguyen, Utrecht University
Roi Reichart, Technion - Israel Institute of Technology

Language Grounding and Multi-Modality

Grzegorz Chrupała, Tilburg University
Desmond Elliott, University of Copenhagen

Language Resources and Evaluation

Roman Klinger, University of Stuttgart
Omri Abend, The Hebrew University of Jerusalem

Linguistic Theories, Cognitive Modeling and Psycholinguistics

Barry Devereux, Queen's University, Belfast
Natalie Schluter, IT University of Copenhagen

Machine Learning for NLP

James Henderson, Idiap Research Institute
Vlad Niculae, University of Amsterdam

Machine Translation

Wilker Aziz, University of Amsterdam
Rico Sennrich, University of Zurich

Multidisciplinary and other NLP Applications

Annie Priyadarshini Louis, Google Research UK
Yulan He, King's College London

Multilinguality

Ivan Vulić, University of Cambridge
Alexander Fraser, Ludwig-Maximilians-Universität München

Phonology, Morphology, and Word Segmentation

Thierry Poibeau, LATTICE (CNRS and ENS/PSL)
François Yvon, ISIR CNRS and Sorbonne Université

Question Answering

Jonathan Berant, Tel Aviv University and AI2
Pontus Stenetorp, University College London

Semantics: lexical

Chris Biemann, Universität Hamburg
Mark Stevenson, University of Sheffield

Semantics: sentence level and other areas

Aliaksei Severyn, Google
Douwe Kiela, Hugging Face

Sentiment Analysis and Argument Mining

Veronique Hoste, LT3, Ghent University
Ivan Habernal, Technical University of Darmstadt

Tagging, Chunking, Syntax and Parsing

Marco Kuhlmann, Linköping University
Shay B. Cohen, University of Edinburgh

Area Chairs

Khalid Al Khatib, Malihe Alikhani, Mikel Artetxe, Akari Asai, Duygu Ataman, Niranjan Balasubramanian, Jeremy Barnes, Max Bartolo, Valerio Basile, Laurent Besacier, Iacer Calixto, Kris Cao, Tanmoy Chakraborty, Bharathi Raja Chakravarthi, Guanyi Chen, Wenhu Chen, Eleanor Chodroff, Caio Corro, Çağrı Çöltekin, Orphee De Clercq, Miryam De Lhoneux, Tejaswini Deoskar, Rotem Dror, Yanai Elazar, Arash Eshghi, Amir Feder, Yang Feng, Radu Florian, Anette Frank, Markus Freitag, André Freitas, Annemarie Friedrich, Jie Fu, Wei Gao, Mor Geva, Dan Goldwasser, Yulia Grishina, Lin Gui, Sara Hooker, Shujian Huang, Patrick Huber, Dieuwke Hupkes, Peter Jansen, Kristen Johnson, David Jurgens, Simon Keizer, Casey Kennington, Daniel Khashabi, Tushar Khot, Germán Kruszewski, Lun-wei Ku, Matthieu Labeau, Gerasimos Lampouras, Mirella Lapata, Gabriella Lapesa, Anne Lauscher, Piji Li, Bin Liang, Bang Liu, Craig Macdonald, Pranava Madhyastha, Andrea Madotto, Saad Mahamood, Jonathan Mallinson, Zita Marinho, Eugenio Martínez Cámara, Florian Metze, Sabrina Mielke, Pushkar Mishra, Saif Mohammad, Preslav Nakov, Shashi Narayan, Debora Nozza, Alexander Panchenko, Alexandros Papangelis, Nikolaos Pappas, Panupong Pasupat, Gabriele Pergola, Jonas Pfeiffer, Yuval Pinter, Edoardo Maria Ponti, Christopher Potts, Daniel Preotiuc-pietro, Emily Prud'hommeaux, Simon Razniewski, Siva Reddy, Sara Rosenthal, Alla Rozovskaya, Keisuke Sakaguchi, Tanja Samardzic, Minjoon Seo, Ehsan Shareghi, Ravi Shekhar, Carina Silberer, Miikka Silfverberg, Felix Stahlberg, Svetlana Stoyanchev, Saku Sugawara, Hiroya Takamura, Niket Tandon, Sara Tonelli, Nicola Tonellotto, Lonneke Van Der Plas, David Vandyke, David Vilares, Elena Voita, Svitlana Volkova, Shuai Wang, Derry Tanti Wijaya, Adina Williams, Wei Wu, Fei Xia, Deyi Xiong, Ikuya Yamada, Marcos Zampieri, Sina Zarrieß, Chrysoula Zerva, Arkaitz Zubiaga

Reviewers

Sadaf Abdul Rauf, Muhammad Abdul-mageed, Ibrahim Abu Farha, Lasha Abzianidze, Angus Addlesee, David Adelani, Stergos Afantenos, Severine Affeldt, Rodrigo Agerri, Piush Aggarwal, Željko Agić, Ameeta Agrawal, Roee Aharoni, Wasi Uddin Ahmad, Sina Ahmadi, Natalie Ahn, Xi Ai, Laura Aina, Akiko Aizawa, Md. Shad Akhtar, Hend Al-khalifa, Rami Al-rfou, Ahmed Alajrami, David Alfter, Bashar Alhafni, Hamed Alhoori, Hafsa Ali, Afra Alishahi, Miguel A. Alonso, Sultan Alrowili, Bharat Ram Ambati, Silvio Amir, Samuel Amouyal, Reinald Kim Amplayo, Jisun An, Vishal Anand, Raviteja Anantha, Antonios Anastasopoulos, Tim Anderson, Melanie Andresen, Anelia Angelova, Alan Ansell, Francesco Antici, Diego Antognini, Maria Antoniak, Dimosthenis Antypas, Reut Apel, Emilia Apostolova, Jun Araki, Oscar Araque, Arturo Argueta, Akhil Arora, Ekaterina Artemova, Elliott Ash, Md.sadek Hossain Asif, Arian Askari, Zhenisbek Assylbekov, Aitziber Atutxa Salazar, Eleftherios Avramidis, Cem Rifki Aydin, Mahmoud Azab

Bharathi B, Jinheon Baek, Selene Baez Santamaria, Parsa Bagherzadeh, Vikas Bahirwani, Fan Bai, Jinyeong Bak, Timothy Baldwin, Miguel Ballesteros, Forrest Sheng Bao, Edoardo Barba, Francesco Barbieri, Ander Barrena, Pierpaolo Basile, Roberto Basili, Ali Basirat, Riza Batista-navarro, Timo Baumann, Rachel Bawden, Christos Baziotis, Ian Beaver, Nadia Bebeshina, Frederic Bechet, Tilman Beck, Beata Beigman Klebanov, Tadesse Destaw Belay, Meriem Beloucif, Farah Benamara, Luca Benedetto, Joshua Bensemann, Gábor Berend, Thales Bertaglia, Michele Bevilacqua, Rasika Bhalerao, Rohan Bhambhoria, Rishabh Bhardwaj, Sumit Bhatia, Arnab Bhattacharya, Rajarshi Bhowmik, Zhen Bi, Iman Munire Bilal, Alexandra Birch, Debmalya Biswas, Eduardo Blanco, Nate Blaylock, Su Lin Blodgett, Jelke Bloem, William Boag, Ben Bogin, Francis Bond, Georgeta Bordea, Logan Born, Emanuela Boros, Elizabeth Boschee, Cristina Bosco, Zied Bouraoui, Tom Bourgeade, Laurestine Bradford, Stephanie Brandl, Ana Brassard, Jonathan Brophy, Caroline Brun, Christian Buck, Sven Buechel, Paul Buitelaar, Razvan Bunescu, Laurie Burchell, Miriam Butt, Jan Buys, Lisa Bylinina, Bill Byrne

Laura Cabello Piqueras, Elena Cabrio, Samuel Cahyawijaya, Agostina Calabrese, Nitay Calderon, Eduardo Calò, Jose Camacho-collados, Ricardo Campos, Marie Candito, Shuyang Cao, Ziqiang Cao, Fabio Carrella, Xavier Carreras, Jorge Carrillo-de-albornoz, Lucien Carroll, Fabio Casati, Tommaso Caselli, Pierluigi Cassotti, Francesco Cazzaro, Amanda Cercas Curry, Dumitru-clementin Cercel, Christophe Cerisara, Alessandra Cervone, Rahma Chaabouni, Haixia Chai, Tuhin Chakrabarty, Yllias Chali, Ilias Chalkidis, Hou Pong Chan, Zhangming Chan, Anshuma Chandak, Senthil Chandramohan, Buru Chang, Ernie Chang, Yung-chun Chang, Guan-lin Chao, Emile Chapuis, Shubham Chatterjee, Rochana Chaturvedi, Kushal Chawla, Ciprian Chelba, Canyu Chen, Chacha Chen, Chung-chi Chen, Derek Chen, Fuxiang Chen, Hsin-hsi Chen, Jie Chen, Lei Chen, Lei Chen, Meng Chen, Mingda Chen, Qian Chen, Qianglong Chen, Qibin Chen, Qingcai Chen, Sanxing Chen, Shizhe Chen, Tongfei Chen, Xiaoli Chen, Xiuying Chen, Yan-ying Chen, Yi-pei Chen, Yunmo Chen, Zhiyu Chen, Fei Cheng, Shanbo Cheng, Emmanuele Chersoni, Ethan A. Chi, Jenny Chim, Hyundong Cho, Key-sun Choi, Alexandra Chronopoulou, George Chrysostomou, Alessandra Teresa Cignarella, Philipp Cimiano, Elizabeth Clark, Chloé Clavel, Simon Clematide, Ann Clifton, Miruna Clinciu, Oana Cocarascu, Davide Colla, Andrei Coman, Simone Conia, John Conroy, Paul Cook, Gonçalo Correia, Israel Cuevas, Peng Cui, Shaobo Cui, Tonya Custis, Arthur Câmara

Thenmozhi D., Jeff Da, Giovanni Da San Martino, Raj Dabre, Gautier Dagan, Deborah Dahl, Wenliang Dai, Xiang Dai, Rumen Dangovski, Falavigna Daniele, Verna Dankers, Aswarth Abhilash Dara, Franck Dary, Mithun Das Gupta, Saurabh Dash, Brian Davis, Heidar Davoudi, Michiel De Jong, Loic De Langhe, Budhaditya Deb, Alok Debnath, Thierry Declerck, Mathieu Dehouck, Luciano Del Corro, Sebastien Delecraz, Vera Demberg, David Demeter, Steve Deneefe, Yuntian Deng, Pascal Denis, Nina Dethlefs, Daniel Deutsch, Murthy Devarakonda, Hannah Devinney, Prajit Dhar, Shehzaad Dhuliawala, Luigi Di Caro, Mona Diab, Shizhe Diao, Gaël Dias, Caiwen Ding, Chenchen Ding, Liang Ding, Nemanja Djuric, Giovanna Maria Dora Dore, Bonaventure F. P. Dossou, Jad Doughman, Doug Downey, Gabriel Doyle, Mauro Dragoni, Rotem Dror, Jinhua Du, Yupei Du, Xiangyu Duan, Pablo Duboue, Philipp Dufter, Kevin Duh, Ewan Dunbar, Jonathan Dunn, Gerard Dupont, Nadir Durrani, Ritam Dutt

Oliver Eberle, Sauleh Eetemadi, Steffen Eger, Annerose Eichel, Bryan Eikema, Julian Eisenschlos, Heba Elfardy, Micha Elsner, Saman Enayati, Aykut Erdem, Akiko Eriguchi, Katrin Erk, Ramy Eskander

Alex Fabbri, Marzieh Fadaee, Fahim Faisal, Neele Falk, Federico Fancellu, Qixiang Fang, Hossein Fani, Stefano Faralli, Oladimeji Farri, Nawshad Farruque, Manaal Faruqui, Mehwish Fatima, Adam Faulkner, Pedro Faustini, Marc Feger, Nils Feldhus, Anna Feldman, Ghazi Felhi, Mariano Felice, Weixi Feng, Yue Feng, Manos Fergadiotis, Patrick Fernandes, Daniel Fernández-gonzález, Elisabetta Fersini, George Filandrianos, Elena Filatova, Mark Fishel, Lucie Flek, Michael Flor, Negar Foroutan Eghlidi, Jennifer Foster, Stella Frank, Jesse Freitas, Simona Frenda, Annemarie Friedrich, Lisheng Fu, Fumiyo Fukumoto, Kotaro Funakoshi

David Gaddy, Andrea Galassi, Leilei Gan, Yujian Gan, William Gantt, Junbin Gao, Qiaozi Gao, Shen Gao, Muskan Garg, Guillermo Garrido, Susan Gauch, Gregor Geigle, Zorik Gekhman, Alborz Geramifard, Felix Gervits, Mozhdeh Gheini, Reshmi Ghosh, Sucheta Ghosh, Voula Giouli, Dimitris Gkoumas, Serge Gladkoff, Catalina Goanta, Jonas Golde, Seraphina Goldfarb-tarrant, Sujatha Das Gollapalli, Jose Manuel Gomez-perez, Jeff Good, Philip John Gorinski, Koustava Goswami, Isao Goto, Christan Grant, Thomas Green, Derek Greene, Milan Gritta, Paul Groth, Julian Grove, Adam Grycner, Jiasheng Gu, Jiuxiang Gu, Xiaodong Gu, Yi Guan, Marco Guerini, Nuno M. Guerreiro, Xiaoyu Guo, Yanzhu Guo, Zhihui Guo, Abhinav Gupta, Ankit Gupta, Ankita Gupta, Ashim Gupta, Pranjal Gupta, Izzeddin Gur, Suchin Gururangan, Ximena Gutierrez-vasques, Jeremy Gwinnup, Tunga Güngör

Le An Ha, Katharina Haemmerl, Gholamreza Haffari, Joonghyuk Hahn, Michael Hahn, Udo Hahn, Eva Hajicova, Dilek Hakkani-tur, Kishaloy Halder, Karina Halevy, Jiuzhou Han, Lifeng Han, Ting Han, Xudong Han, Yo-sub Han, Viktor Hangya, Sanda Harabagiu, Mareike Hartmann, Sadid A. Hasan, Sabit Hassan, Nabil Hathout, Amartya Hatua, Annette Hautli-janisz, Adi Haviv, Yoshihiko Hayashi, Shirley Anugrah Hayati, T. J. Hazen, Rishi Hazra, Han He, Wanwei He, Wei He, Xiaoting He, Xuanli He, Xuehai He, Yun He, Behnam Hedayatnia, Kevin Heffernan, Benjamin Heinzerling, Jindřich Helcl, William Held, Leonhard Hennig, Christian Herold, Jonathan Herzig, Gerhard Heyer, Derrick Higgins, Anthony Hills, Tatsuya Hiraoka, Vinh Thinh Ho, Cuong Hoang, Eben Holderness, Takeshi Homma, Ales Horak, Andrea Horbach, Sho Hoshino, Md Azam Hossain, Feng Hou, Yifan Hou, Yufang Hou, Shu-kai Hsieh, I-hung Hsu, Han Hu, Po Hu, Xinyu Hua, Chieh-yang Huang, Fei Huang, Hen-hsen Huang, Jie Huang, Junbo Huang, Kuan-hao Huang, Quzhe Huang, Zhiqi Huang, Vojtěch Hudeček, Pere-Lluís Huguet Cabot, Kai Hui, Chia-chien Hung, Julie Hunter

Nikolai Ilinykh, Dmitry Ilvovsky, Michimasa Inaba, Diana Inkpen, Koji Inoue, Hayate Iso, Takumi Ito, Maor Ivgi, Kenichi Iwatsuki, Vivek Iyer, Peter Izsak

Cassandra L. Jacobs, Sarthak Jain, Masoud Jalili Sabet, Sepehr Janghorbani, Adam Jatowt, Inigo Jauregi Unanue, Ganesh Jawahar, Harsh Jhamtani, Shaoxiong Ji, Yangfeng Ji, Chengyue Jiang, Junfeng Jiang, Longquan Jiang, Ming Jiang, Yuchen Eleanor Jiang, Ziyan Jiang, Baoyu Jing, Unso Jo, Richard Johansson, Aditya Joshi, Rishabh Joshi, Taehee Jung

Besim Kabashi, Sylvain Kahane, Mihir Kale, Laura Kallmeyer, Ehsan Kamalloo, Hidetaka Kamigaito, Jaap Kamps, Lis Kanashiro Pereira, Hiroshi Kanayama, Yoshinobu Kano, Diptesh Kanojia, Sudipta Kar, Georgi Karadzhov, Elena Karagjosova, Mladen Karan, Sarvnaz Karimi, Börje Karlsson, Sanjeev Kumar Karn, Constantinos Karouzos, Pradeep Karturi, Zdeněk Kasner, Yoshihide Kato, Uri Katz, Yoav Katz, Divyansh Kaushik, Pride Kavumba, Daisuke Kawahara, Gary Kazantsev, Ashkan Kazemi, Yova Kementchedjhieva, Muhammad Khalifa, Abdul Khan, Sopan

Khosla, Halil Kilicoglu, Gyuwan Kim, Hyunwoo Kim, Jonggu Kim, Joo-kyung Kim, Mi-young Kim, Seungone Kim, Sungdong Kim, Young Jin Kim, Youngbin Kim, David King, Tracy Holloway King, Svetlana Kiritchenko, Jan-christoph Klie, Julien Kloetzer, René Knaebel, Sang-ki Ko, Thomas Kober, Elena Kochkina, Konstantinos Kogkalidis, Mare Koit, Thomas Kollar, Alexander Koller, Mamoru Komachi, Rik Koncel-kedziorski, Grzegorz Kondrak, Sai Koneru, Deguang Kong, Miloslav Konopík, Yannis Korkontzelos, Katerina Korre, Fajri Koto, Alexander Kotov, Mahnaz Koupaee, Venelin Kovatchev, Pavel Kral, Lea Krause, Kalpesh Krishna, Mateusz Krubiński, Canasai Kruengkrai, Jaap Kruijt, Ruben Kruiper, Sicong Kuang, Mayank Kulkarni, Deepak Kumar, Sachin Kumar, Shankar Kumar, Olli Kuparinen, Robin Kurtz, Andrey Kutuzov, Haewoon Kwak

Gorka Labaka, Sofie Labat, Faisal Ladhak, Cheng-i Lai, Tuan Lai, Wen Lai, Vasileios Lampos, Gerasimos Lampouras, Lukas Lange, Ekaterina Lapshinova-koltunski, Stefan Larson, Mark Last, Alexandra Lavrentovich, Hoang-quynh Le, Hung Le, Phong Le, Joseph Le Roux, Kevin Leach, Dong-ho Lee, Grandee Lee, Ji-ung Lee, John Lee, Lung-hao Lee, Nayeon Lee, Roy Ka-wei Lee, Els Lefever, Wenqiang Lei, Jochen Leidner, Heather Lent, Ran Levy, Bei Li, Bryan Li, Changmao Li, Cheng Li, Dingcheng Li, Dongfang Li, Jiacheng Li, Jialu Li, Jiazhao Li, Jing Li, Jiyi Li, Juan Li, Lei Li, Liunian Harold Li, Maoxi Li, Miao Li, Peifeng Li, Sheng Li, Shiyang Li, Shuyang Li, Siheng Li, Wei Li, Wei Li, Weikang Li, Wenyan Li, Xiangju Li, Xiaodi Li, Xue Li, Yanran Li, Yanzeng Li, Yaoyiran Li, Yizhi Li, Yongbin Li, Yue Li, Yuncong Li, Zhuang Li, Zichao Li, Chao-chun Liang, Xinnian Liang, Yueqing Liang, Baohao Liao, Jindřich Libovický, Constantine Lignos, Gilbert Lim, Kwan Hui Lim, Tomasz Limisiewicz, Lucy Lin, Weizhe Lin, Zhenxi Lin, Nedim Lipka, Pierre Lison, Shir Lissak, Danni Liu, Fangyu Liu, Fenglin Liu, Hui Liu, Jiangming Liu, Kang Liu, Lei Liu, Nayu Liu, Nelson F. Liu, Tianyu Liu, Tianyu Liu, Ting Liu, Yang Janet Liu, Yiyi Liu, Yonghui Liu, Yongkang Liu, Yue Liu, Zihan Liu, Zitao Liu, Zoey Liu, Nikola Ljubešić, Sharid Loaiciga, Colin Lockard, Pintu Lohar, Yunfei Long, Oier Lopez De Lacalle, Jaime Lorenzo-trueba, Daniel Loureiro, Junru Lu, Keming Lu, Xing Han Lu, Yanbin Lu, Yao Lu, Yujie Lu, Nurul Lubis, Jiaming Luo, Man Luo, Haoran Lv, Shangwen Lv, Teresa Lynn, Alex Lưu

Meryem M'hamdi, Jie Ma, Jing Ma, Long-long Ma, Mingyu Derek Ma, Xiaofei Ma, Andrew Mackey, Aman Madaan, Avinash Madasu, Mounica Maddela, Manuel Mager, Bernardo Magnini, Adyasha Maharana, Quan Mai, Frederic Mailhot, Jean Maillard, Peter Makarov, Aaron Maladry, Ankur Mali, Anton Malko, Jonathan Mallinson, Eric Malmi, Valentin Malykh, Ramesh Manuvinakurike, Vladislav Maraev, Ana Marasovic, David Mareček, Katerina Margatina, Katja Markert, Edison Marrese-taylor, Federico Martelli, Louis Martin, Héctor Martínez Alonso, Claudia Marzi, Sarah Masud, Sandeep Mathias, Prashant Mathur, Diana Maynard, Sahisnu Mazumder, Alessandro Mazzei, R. Thomas Mccoy, John P. Mccrae, Bridget Mcinnes, Nick Mckenna, Nikhil Mehta, Fanchao Meng, Yan Meng, Zhao Meng, Orfeas Menis Mastromichalakis, Elena Merdjanovska, Eleni Metheniti, Ivan Vladimir Meza Ruiz, Paul Michel, Timothee Mickus, Stuart Middleton, Aristides Milios, Tristan Miller, David Mimno, Erxue Min, Seyedabolghasem Mirroshandel, Paramita Mirza, Abhijit Mishra, Kanishka Misra, Yusuke Miyao, Ashutosh Modi, Alireza Mohammadshahi, Hosein Mohebbi, Afroz Mohiuddin, Diego Molla, Manuel Montes, Mehrad Moradshahi, Roser Morante, Jose G. Moreno, Alejandro Moreo, Marius Mosbach, Pablo Mosteiro, Lili Mou, Diego Moussallem, Maximilian Mozes, Emir Munoz, Dragos Munteanu, Rudra Murthy, Alberto Muñoz-ortiz, Mathias Müller

Dawn Nafus, Masaaki Nagata, Saeed Najafi, Tetsuji Nakagawa, Yuta Nakashima, Diane Napolitano, Jason Naradowsky, Vivi Nastase, Anmol Nayak, Ambreen Nazir, Ani Nenkova, Mariana Neves, Jun-ping Ng, Raymond Ng, Vincent Ng, Axel-cyrille Ngonga Ngomo, Dat Quoc Nguyen, Kiet Nguyen, Nhung Nguyen, Quoc-an Nguyen, Trung Hieu Nguyen, Vincent Nguyen, Xuanfan Ni, Garrett Nicolai, Massimo Nicosia, Feng Nie, Yixin Nie, Jan Niehues, Mitja Nikolaus, Giannis

Nikolentzos, Takashi Ninomiya, Kosuke Nishida, Sergiu Nisioi, Gibson Nkhata, Tadashi Nomoto, Aurélie Névéol

Alexander O'connor, Tim Oates, Kemal Oflazer, Shu Okabe, Naoaki Okazaki, Tsuyoshi Okita, Oleg Okun, Eda Okur, Antoni Oliver, Mattia Opper, Abigail Oppong, Brian Ore, Hadas Orgad, Maite Oronoz, Petya Osenova, Jessica Ouyang

Teresa Paccosi, Ankur Padia, Aishwarya Padmakumar, Shramay Palta, Tuğba Pamay Arslan, Mugdha Pandya, Wei Pang, Pinelopi Papalampidi, Nikos Papasarantopoulos, Sara Papi, Emerson Paraiso, Ashwin Paranjape, Letitia Parcalabescu, Thiago Pardo, Antonio Pareja-lora, Chanjun Park, Jong Park, Sungkyu Park, Alicia Parrish, Tommaso Pasini, Clemente Pasti, Braja Gopal Patra, Viviana Patti, Debjit Paul, Indraneil Paul, Sachin Pawar, Sarah Payne, Pavel Pecina, Jiaxin Pei, Weiping Pei, Stephan Peitz, Baolin Peng, Bo Peng, Hao Peng, Qiyao Peng, Wei Peng, Juan Antonio Perez-ortiz, Charith Peris, Ben Peters, Matthew Peters, Eva Pettersson, Thang Pham, Scott Piao, Maciej Piasecki, Massimo Piccardi, Matúš Pikuliak, Nisha Pillai, Telmo Pires, Flammie Pirinen, Benjamin Piwowarski, Flor Miriam Plaza-del-arco, Brian Plüss, Massimo Poesio, Simone Paolo Ponzetto, Octavian Popescu, Amir Pouran Ben Veyseh, Karan Praharaj, Piotr Przybyła, Stephen Pulman, Juan Manuel Pérez

Ehsan Qasemi, Hongjin Qian, Kun Qian, Kechen Qin, Jielin Qiu, Ariadna Quattoni

Ella Rabinovich, Muhammad Rahman, Sunny Rai, Vyas Raina, Sara Rajaee, Ori Ram, Taraka Rama, Giulia Rambelli, Abhinav Ramesh Kashyap, Rita Ramos, Alan Ramponi, Leonardo Ranaldi, Tharindu Ranasinghe, Surangika Ranathunga, Priya Rani, Ahmad Rashid, Pushpendre Rastogi, David Rau, Vikas Raunak, Eran Raveh, Shauli Ravfogel, Soumya Ray, Evgeniia Razumovskaia, Hanumant Redkar, Georg Rehm, Ricardo Rei, Machel Reid, Navid Rekabsaz, Ricardo Ribeiro, Giuseppe Riccardi, German Rigau, Matīss Rikters, Tharathorn Rimchala, Laura Rimell, Fabio Rinaldi, Ruty Rinott, Anthony Rios, Lina M. Rojas Barahona, Subendhu Rongali, Michael Rosner, Michael Roth, Guy Rotman, Bryan Routledge, Marco Rovera, Soumyadeep Roy, Yu-ping Ruan, Koustav Rudra, Federico Ruggeri, Irene Russo, Phillip Rust, Max Ryabinin, Maria Ryskina, Egil Rønningstad, Susanna Rücker

Malliga S, Kogilavani S V, Kenji Sagae, Keisuke Sakaguchi, Ander Salaberria, Shailaja Keyur Sampat, David Samuel, Ramon Sanabria, George Sanchez, Hugo Sanjurjo-gonzález, Sonal Sannigrahi, Rodrigo Santos, Naomi Saphra, Ruhi Sarikaya, Anoop Sarkar, Felix Sasaki, Ryohei Sasano, Nishanth Sastry, Danielle Saunders, Thiusius Savarimuthu, Beatrice Savoldi, Apoorv Saxena, Federico Scafoglieri, Andreas Scherbakov, Dominik Schlechtweg, Jonathan Schler, Michael Sejr Schlichtkrull, Robin Schmidt, Nathan Schneider, Stephanie Schoch, Annika Marie Schoene, Merel Scholman, Sabine Schulte Im Walde, Philip Schulz, Stefan Schweter, Anastasiia Sedova, Elad Segal, Cory Shain, Guokan Shang, Yutong Shao, Ori Shapira, Matthew Shardlow, Shuaijie She, Artem Shelmanov, Aili Shen, Lingfeng Shen, Xiaoyu Shen, Yuming Shen, Michael Sheng, Qiang Sheng, Tom Sherborne, Freda Shi, Zhan Shi, Zhengxiang Shi, Tomohide Shibata, Yutaro Shigeto, Takahiro Shinozaki, Kumar Shridhar, Akshat Shrivastava, Kai Shu, Raphael Shu, Anna Shvets, Anthony Sicilia, Alejandro Sierra-múnera, João Ricardo Silva, Danilo Silva De Carvalho, Patrick Simianer, Edwin Simpson, Mayank Singh, Pranaydeep Singh, Koustuv Sinha, Sunayana Sitaram, Milena Slavcheva, Kevin Small, Marco Antonio Sobrevilla Cabezudo, Swapna Somasundaran, Kai Song, Linfeng Song, Wei Song, Yan Song, Alexey Sorokin, Xabier Soto, Sajad Sotudeh, Andreas Spitz, Ivan Srba, Makesh Narsimhan Sreedhar, Hiranmai Sri Adibhatla, Balaji Vasan Srinivasan, Miloš Stanojević, Gabriel Stanovsky, Katherine Stasaski, Dario Stojanovski, Alessandro Stolfo, Tomek Strzalkowski, Dan Su, Katsuhito Sudoh, Yoshi Suhara, Alane Suhr, Changzhi Sun, Chenkai Sun, Jian Sun, Ming Sun, Qingfeng Sun, Zewei Sun, Megha Sundriyal, Hanna Suominen, Colin Swaelens, Sandesh Swamy, Vinitra Swamy, Piotr Szymański, Danae Sánchez Villegas, Víctor M. Sánchez-cartagena, Felipe Sánchez-martínez

Santosh T.y.s.s, Sho Takase, Zeerak Talat, George Tambouratzis, Fabio Tamburini, Akihiro Tamura, Chenhao Tan, Fei Tan, Xingwei Tan, Liyan Tang, Raphael Tang, Shuai Tang, Xuting Tang, Yuka Tateisi, Marta Tatu, Selma Tekir, Serra Sinem Tekiroğlu, Irina Temnikova, Daniela Teodorescu, Urmish Thakker, Mokanarangan Thayaparan, Anton Thielmann, Brian Thompson, Craig Thomson, Camilo Thorne, Tristan Thrush, Jörg Tiedemann, Refael Tikochinski, Erik Tjong Kim Sang, Evgeniia Tokarchuk, Takenobu Tokunaga, Nadi Tomeh, Marc Tomlinson, Atnafu Lambebo Tonja, Samia Touileb, Marcos Treviso, Chen-tse Tsai, Adam Tsakalidis, Yu-hsiang Tseng, Yuen-hsien Tseng, Eleftheria Tsipidi, Don Tuggener, Martin Tutek

Kiyotaka Uchimoto, Dennis Ulmer, Kanimozhi Uma, Prajna Upadhyay, Masao Utiyama

Sowmya Vajjala, Marco Valentino, Antal Van Den Bosch, Daan Van Esch, Carel Van Niekerk, Vincent Vandeghinste, Keith Vanderlinden, Lindsey Vanderlyn, Natalia Vanetik, Rossella Varvara, Shikhar Vashishth, Eva Maria Vecchi, Giulia Venturi, Rakesh Verma, Rohil Verma, Giorgos Vernikos, David Vilar, Serena Villata, Esau Villatoro-tello, Juraj Vladika, Piek Vossen, Thuy Vu, Xuan-son Vu, Ekaterina Vylomova

Tomasz Walkowiak, Yu Wan, Chuan-ju Wang, Fei Wang, Hai Wang, Haoyu Wang, Hong Wang, Jianzong Wang, Jiayi Wang, Jin Wang, Jing Wang, Kaifu Wang, Liang Wang, Lingzhi Wang, Longshaokan Wang, Longyue Wang, Miaosen Wang, Ping Wang, Qingyun Wang, Shun Wang, Wei Wang, Weichao Wang, Xin Wang, Xing Wang, Xinyi Wang, Xu Wang, Yasheng Wang, Yining Wang, Zhaowei Wang, Zhilin Wang, Zhiruo Wang, Prashan Wanigasekara, Moshe Wasserblat, Shinji Watanabe, Lucas Weber, Anna Wegmann, Jerry Wei, Wei Wei, Benjamin Weiss, Gail Weiss, Leonie Weissweiler, Charles Welch, Rongxiang Weng, Aaron White, John Wieting, Gijs Wijnholds, Adina Williams, Miles Williams, Steven Wilson, Genta Winata, Guillaume Wisniewski, Seungpil Won, Ka Ho Wong, Alina Wróblewska, Di Wu, Fangzhao Wu, Minghao Wu, Stephen Wu, Winston Wu, Xianchao Wu, Xiaofeng Wu, Xixin Wu, Yuxiang Wu, Joern Wuebker, Amelie Wührl

Min Xiao, Yuqing Xie, Zhenchang Xing, Chao Xiong, Ying Xiong, Lv Xiucheng, Dongkuan Xu, Fangyuan Xu, Hanzi Xu, Haotian Xu, Hongfei Xu, Jia Xu, Jinan Xu, Qiongkai Xu, Ruifeng Xu, Silei Xu, Xinnuo Xu, Yueshen Xu, Zhen Xu, Huiyin Xue, Linting Xue, Christos Xypolopoulos

Ivan Yamshchikov, An Yan, Ming Yan, Xi Yan, Xifeng Yan, Bohao Yang, Hao Yang, Hsiu-yu Yang, Linyi Yang, Longfei Yang, Shiquan Yang, Tao Yang, Xianjun Yang, Ze Yang, Roman Yangarber, Ken Yano, Tae Yano, Wenlin Yao, Fanghua Ye, Asaf Yehudai, Wen-wai Yim, Seid Muhie Yimam, Congchi Yin, Seunghyun Yoon, Soyoung Yoon, Ori Yoran, Naoki Yoshinaga, Chenyu You, Steve Young, Bei Yu, Juntao Yu, Kai Yu, Pengfei Yu, Shoubin Yu, Tiezheng Yu, Xiaodong Yu, Xinyan Yu, Yanchao Yu, Jianhua Yuan, Shuzhou Yuan, Frances Yung

Olga Zamaraeva, Daoguang Zan, Fabio Massimo Zanzotto, Alessandra Zarcone, Xingshan Zeng, Torsten Zesch, Shuang (sophie) Zhai, Haolan Zhan, Biao Zhang, Bowen Zhang, Ge Zhang, Haodi Zhang, Haopeng Zhang, Jason Zhang, Jianguo Zhang, Lei Zhang, Michael Zhang, Ruiyi Zhang, Sheng Zhang, Shiyue Zhang, Tianchi Zhang, Yanzhe Zhang, Yichi Zhang, Yu Zhang, Yuan Zhang, Yuhui Zhang, Zhirui Zhang, Zhisong Zhang, Hai Zhao, Jinming Zhao, Lin Zhao, Mengjie Zhao, Qinghua Zhao, Tiancheng Zhao, Xiaoyan Zhao, Yilun Zhao, Chujie Zheng, Yinhe Zheng, Alisa Zhila, Yang Zhong, Ben Zhou, Guangyou Zhou, Junpei Zhou, Kaitlyn Zhou, Wangchunshu Zhou, Xiang Zhou, Yichu Zhou, Yue Zhou, Zhengyu Zhou, Su Zhu, Wanrong Zhu, Wanzheng Zhu, Xuan

Zhu, Yuan Zhuang, Claus Zinn, Yftah Ziser, Michael Zock, Bowei Zou, Wei Zou, Vilém Zouhar

Outstanding Reviewers

Gavin Abercrombie, Sallam Abualhaija, Yamen Ajjour, Emily Allaway, Milad Alshomary, Talita Anthonio, Lauriane Aufrant, Gorka Azkune, Lisa Beinborn, Valeriia Bolotova-baranova, Michele Cafagna, Deng Cai, Giovanni Cassani, Hanjie Chen, Cheng-han Chiang, Trevor Cohn, Karel D'oosterlinck, Jay Deyoung, Frank Drewes, Markus Dreyer, Tobias Falke, Yimai Fang, Xiaocheng Feng, Olivier Ferret, Antske Fokkens, Saadia Gabriel, Atticus Geiger, Tomas Goldsack, Konstantin Golobokov, Colin Gordon, Liane Guillou, Meiqi Guo, Nitish Gupta, William Havard, Michael Heck, Sophie Henning, Nora Hollenstein, Radu Tudor Ionescu, Tatsuya Ishigaki, Robin Jia, Min-yen Kan, Graham Katz, Christo Kirov, Ioannis Konstas, Michael Kranzlein, Udo Kruschwitz, Roland Kuhn, Yi-an Lai, Young-suk Lee, Yves Lepage, Piyawat Lertvittayakumjorn, Matthias Lindemann, Zhengyuan Liu, Henrique Lopes Cardoso, Brielen Madureira, Yuval Marton, Jonathan May, Kathleen Mckeown, Clara Meister, Zaiqiao Meng, Filip Miletic, Kata Naszadi, Yasumasa Onoe, Juri Opitz, Tiago Pimentel, Barbara Plank, Traian Rebedea, Ehud Reiter, Mathieu Roche, Rudolf Rosa, Candace Ross, Sumegh Roychowdhury, Sebastian Ruder, Elizabeth Salesky, David Schlangen, Hendrik Schuff, Sebastian Schuster, Djamé Seddah, Mattia Setzu, Kyle Shaffer, Vered Shwartz, Olivier Siohan, Matthew Stone, Alessandro Suglia, Benjamin Van Durme, Neeraj Varshney, Jake Vasilakes, Dirk Väth, Henning Wachsmuth, Michael Wiegand, Tomer Wolfson, Hanqi Yan, Eugene Yang, Marcely Zanon Boito, Amir Zeldes

Keynote Talk: Going beyond the benefits of scale by reasoning about data

Edward Grefenstette
Cohere

Tuesday, May 2, 2023 – Time: **09:30 – 10:30** – Room: **Elafiti 1, 2, 3 & 4**

Abstract:

Transformer-based Large Language Models (LLMs) have taken NLP—and the world—by storm. This inflection point in our field marks a shift from focussing on domain-specific neural architecture design and the development of novel optimization techniques and objectives to a renewed focus on the scaling of model size and of the amount of data ingested during training. This paradigm shift yields surprising and delightful applications of LLMs, such as open-ended conversation, code understanding and synthesis, some degree of tool-use, and some zero-shot instruction-following capabilities. In this talk, I outline and lightly speculate on the mechanisms and properties which enable these diverse applications, and posit that the training regimen which enables these capabilities points to a further shift, namely one where we go from focussing on scale, to focussing on reasoning about what data to train on. I will briefly discuss recent advances in open-ended learning in Reinforcement Learning, and how some of the concepts at play in that work may inspire or directly apply to the development of novel ways of reasoning about data in supervised learning, in particular in areas pertaining to LLMs.

Bio:

Ed Grefenstette is the Head of Machine Learning at Cohere, a provider of cutting-edge NLP models that's solving all kinds of language problems; including text summarization, composition, classification and more. In addition, Ed is an Honorary Professor at UCL. Ed's previous industry experience comprises Facebook AI Research (FAIR), DeepMind, and Dark Blue Labs, where he was the CTO (acquired by Google in 2014). Prior to this, Ed worked at the University of Oxford's Department of Computer Science, and was a Fulford Junior Research Fellow at Somerville College, whilst also lecturing students at Hertford College taking Oxford's new computer science and philosophy course. Ed's research interests span several topics, including natural language and generation, machine reasoning, open ended learning, and meta-learning.

Keynote Talk: Chatbots for Good and Evil

Kevin Munger
Penn State University

Wednesday, May 3, 2023 – Time: **15:45 – 16:45** – Room: **Elafiti 1, 2, 3 & 4**

Abstract:
The capacities of LLM-powered chatbots have been progressing on the order of months and have recently passed into mainstream public awareness and adoption. These tools have been used for a variety of scientific and policy interventions, but these advances call for a significant re-thinking of their place in society. Psychological research suggests that intentionalityis a key factor in persuasion and social norm enforcement, and the proliferation of LLMs represents a significant shock to the intentionalitycontained in text and particularly in immediate, personalized chat. I argue that we are in a period of informational disequilibrium,where different actors have different levels of awareness of this technological shock. This period may thus represent a golden age for actors aiming to use these technologies at scale, for any number of normative ends; this includes social scientists and computational linguists. More broadly, I argue that the ethicalframeworks for evaluating research practices using LLM-powered chatbots are insufficient to the scale of the current challenge. This is a potentially revolutionary technology that requires thinking in moral and political terms: given the power imbalances involved, it is of paramount importance that chatbots for good do not inadvertently become chatbots for evil.

Bio:
Kevin Munger is the Jeffrey L. Hyde and Sharon D. Hyde and Political Science Board of Visitors Early Career Professor of Political Science and Assistant Professor of Political Science and Social Data Analytics at Penn State University.Kevin's research focuses on the implications of the internet and social media for the communication of political information. His speciality is the investigation of the economics of online media; current research models Clickbait Mediaand uses digital experiments to test the implications of these models on consumers of political information.

Keynote Talk: Language Use in Embodied AI

Joyce Chai
University of Michigan

Thursday, May 4, 2023 – Time: **14:15 – 15:15** – Room: **Elafiti 1, 2, 3 & 4**

Abstract:

With the emergence of a new generation of embodied AI agents, it becomes increasingly important to enable language communication between humans and agents. Language plays many important roles in embodied AI. In this talk, I will share some of the experiences in my lab that study the pragmatics of language, for example, in mediating perceptual differences, learning from language instructions, and planning for joint tasks. I will talk about how the embodied context shapes language use and influences computational models for language grounding to perception and action. I will show the importance of collaborative effort and theory of mind in language communication and how they affect common ground for situated tasks. I will discuss key challenges as well as new perspectives on these problems brought by recent advances in LLM and generative AI.

Bio:

Joyce Chai is a Professor in the Department of Electrical Engineering and Computer Science at the University of Michigan. Before joining UM in 2019, she was a Professor of Computer Science and Engineering at Michigan State University. She holds a Ph.D. in Computer Science from Duke University. Her research interests span from natural language processing and embodied AI to human-AI collaboration. She is fascinated by how experience with the world and how social pragmatics shape language learning and language use; and is excited about developing language technology that is sensorimotor grounded, pragmatically rich, and cognitively motivated. Her current work explores the intersection between language, perception, and action to enable situated communication with embodied agents. She served on the executive board of NAACL and as Program Co-Chair for multiple conferences – most recently ACL 2020. She is a recipient of the National Science Foundation Career Award and has received several paper awards with her students (e.g., the Best Long Paper Award at ACL 2010 and an Outstanding Paper Award at EMNLP 2021). She is a Fellow of ACL.

Panel: Low-resource languages in NLP products

Wednesday, May 3, 2023 – Time: **16:30 – 18:00** – Room: **Elafiti 1, 2, 3 & 4**

The panel discussion will bring together experts from industry and academia to share their experience building solutions for low-resource languages. We anticipate a lively discussion about the advantages and limitations of multilingual solutions and language-specific models, the challenges of evaluating models for low-resource languages, and the level of language awareness needed in the development process. In addition, the panelists will explore ways to increase the acceptance rate of papers that target low-resource languages at *ACL conferences. We hope that the panel discussion will increase the visibility of research for low-resource languages and emphasize its relevance.

Moderator: Mariana Romanyshyn, **Grammarly**

Mariana Romanyshyn is an Area Tech Lead for Computational Linguistics at Grammarly, Ukraine. She has professional experience in syntactic parsing, sentiment analysis, named entity recognition, fact extraction, and text anonymization. For the last eight years, Mariana has been working on error correction and text improvement algorithms at Grammarly. Mariana is an active speaker at AI conferences, co-organizer of the yearly Grammarly CompLing Summer School, co-organizer of the UNLP workshop, struggling reformer of Ukrainian university syllabuses, and active contributor of the Lang-uk group, focused on advancements in Ukrainian NLP.

Panelists:

Antonios Anastasopoulos, **George Mason University**

Antonios Anastasopoulos is an Assistant Professor in Computer Science at George Mason University. He received his PhD in Computer Science from the University of Notre Dame and then did a postdoc at Language Technologies Institute at Carnegie Mellon University. He also holds a BSc-MSc in Electrical and Computer Engineering from the National Technical University of Athens, Greece. His research is on natural language processing with a focus on multilinguality, low-resource settings, cross-lingual learning, and endangered languages, with the ultimate goal of building language technologies for under-served communities around the world. He is currently funded by the NSF, the NEH, the US DoD, Google, Amazon, and Meta.

Mona Diab, **Meta**

Mona Diab is the Lead Responsible AI Research Scientist with Meta. She is also a full Professor of Computer Science at the George Washington University (on leave) where she directs the CARE4Lang NLP Lab. Before joining Meta, she led the Lex Conversational AI project within Amazon AWS AI. Her current focus is on Responsible AI and how to operationalize it for NLP technologies. Her interests span building robust technologies for low-resource scenarios with a special interest in Arabic technologies, (mis) information propagation, computational socio-pragmatics, computational psycholinguistics, NLG evaluation metrics, language modeling, and resource creation.

Julia Makogon, Semantrum
Julia Makogon is a Lead ML/NLP Engineer at Semantrum, a Ukrainian AI company that specializes in media analytics and reputation management. She studied Applied Mathematics at DSTU, Kamyanske, Ukraine, before pursuing a career in NLP. Julia developed multiple NLP applications for media monitoring, sentiment analysis, and legal document analysis for Ukrainian and other European languages. Her expertise lies in building industry solutions with limited resources. Julia serves at the Program Committee of the Ukrainian NLP workshop and is passionate about advancing solutions for the Ukrainian language.

Ivan Vulić, University of Cambridge
Ivan Vulić is a Principal Research Associate and a Royal Society University Research Fellow in the Language Technology Lab, University of Cambridge. He is also a Senior Scientist at PolyAI. He is a member of the Steering Committee of the Centre for Human Inspired Artificial Intelligence (CHIA) at Cambridge. Ivan holds a PhD in Computer Science from KU Leuven awarded summa cum laude. In 2021 he was awarded the annual Karen Spärck Jones Award from the British Computing Society for his research contributions to NLP and Information Retrieval. His core expertise is in representation learning, cross-lingual learning, conversational AI, human language understanding, distributional, lexical, multi-modal, and knowledge-enhanced semantics in monolingual and multilingual contexts, transfer learning for enabling cross-lingual NLP applications such as conversational AI in low-resource languages, and machine learning for (cross-lingual and multilingual) NLP. He has published numerous papers at top-tier NLP and Information Retrieval conferences and journals, and his research work also resulted in several best paper awards. He serves as an area chair and regularly reviews for all major NLP and Machine Learning conferences and journals. Ivan has given numerous invited talks at academia and industry and co-organised a number of NLP conferences and workshops.

Table of Contents

PiC: A Phrase-in-Context Dataset for Phrase Understanding and Semantic Search

Thang M. Pham †
thangpham@auburn.edu

Seunghyun Yoon §
syoon@adobe.com

Trung Bui §
bui@adobe.com

Anh Nguyen †
anh.ng8@gmail.com

†Auburn University §Adobe Research

Abstract

While contextualized word embeddings have been a de-facto standard, learning contextualized *phrase* embeddings is less explored and being hindered by the lack of a human-annotated benchmark that tests machine understanding of phrase semantics given a context sentence or paragraph (instead of phrases alone). To fill this gap, we propose PiC—a dataset of ∼28K of noun phrases accompanied by their contextual Wikipedia pages and a suite of three tasks for training and evaluating phrase embeddings. Training on PiC improves ranking-models' accuracy and remarkably pushes span-selection (SS) models (i.e., predicting the start and end index of the target phrase) near human-accuracy, which is 95% Exact Match (EM) on semantic search given a query phrase and a passage. Interestingly, we find evidence that such impressive performance is because the SS models learn to better capture the common meaning of a phrase *regardless* of its actual context. SotA models perform poorly in distinguishing two senses of the same phrase in two contexts (∼60% EM) and in estimating the similarity between two different phrases in the same context (∼70% EM).

1 Introduction

Understanding phrases in context is a key to learning new vocabularies (Nagy et al., 1985; Fischer, 1994), disambiguation (Pilehvar and Camacho-Collados, 2019), and many downstream tasks, including semantic search (Finkelstein et al., 2001). Yet, the contextualized *phrase* embeddings (Yu and Ettinger, 2020) in existing systems mostly capture the common meaning of a phrase, i.e. without strong dependence on its context (Yu and Ettinger, 2020). While there are *word*-sense disambiguation datasets (Edmonds and Cotton, 2001; Pilehvar and Camacho-Collados, 2019), no such benchmarks exist for *phrases*. Existing phrase-similarity benchmarks (Pavlick et al., 2015; Turney, 2012; Asaadi et al., 2019; Zhang et al., 2019; Yang et al., 2019) compare phrases alone (without context) and some of them (Pavlick et al., 2015; Zhang et al., 2019) contain a large, undesired amount (∼15% to 99%) of phrase pairs that have lexical overlap (Table 1).

Others generated the context for a phrase by querying GPT-2 (Wang et al., 2021) or by retrieving from Wikipedia (Yu and Ettinger, 2020). Yet, there was no human verification of the realism of generated text (Wang et al., 2021) and no human annotation of how a phrase's meaning changes w.r.t. the context (Yu and Ettinger, 2020). All above drawbacks are limiting the evaluation of phrase understanding.

To advance the development of contextualized phrase embeddings, we propose Phrase-in-Context (PiC), a suite of three tasks: (1) Phrase Similarity (PS), i.e. compare the semantic similarity of two phrases in the same context sentence (Fig. 1b); (2) Phrase Retrieval (PR), which is divided into PR-pass and PR-page (Fig. 1c–d), i.e. from a passage or a Wikipedia page, retrieve a phrase semantically-similar to a given query phrase; and (3) Phrase-Sense Disambiguation (PSD), i.e. find the target phrase $\boldsymbol{p}$ semantically similar to the query phrase from a 2-paragraph document where $\boldsymbol{p}$ appears twice, each time in a different context paragraph that provides a *unique* meaning to $\boldsymbol{p}$ (Fig. 1e). Our ∼28K-example dataset is rigorously (a) *annotated* and *verified* by two groups of annotators: linguistics experts on `Upwork.com` and non-experts on Amazon Mechanical Turk (MTurk); and then (b) *tested* by models, linguists, and graduate students. Our contributions are:

1. We build PiC[1], the first, human-annotated benchmark for evaluating and training contextualized phrase embeddings (Sec. 4). Compared to existing phrase similarity datasets, PS is the first to require models to rely on context.

---

[1] Dataset, code, and demos are available on `https://phrase-in-context.github.io`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1–26
May 2-6, 2023 ©2023 Association for Computational Linguistics

2. After training on PR-pass, i.e. finding a phrase from a passage, SS models perform at a near-human accuracy (92–94% vs. 95% EM). They also score high (84–89% EM) on PR-page, i.e. semantic phrase search in a Wikipedia page (Sec. 5.4), suggesting our training set and learned embeddings are useful for real-world semantic search.

3. Interestingly, on PR-pass, harnessing these SS models' phrase embeddings in a ranking approach (i.e. comparing the similarity between the query and *all* candidate phrases) yields poor accuracy of $\leq 59\%$ EM (Sec. H), setting a challenge for future research into learning contextualized phrase embeddings.

4. After training on PR-pass, state-of-the-art (SotA) models perform relatively well on PR-pass and even PR-page but not on PSD (Sec. 5.5). On PS, SotA models perform poorly (below 70% accuracy) in binary classification of phrase similarity given a context sentence (Sec. 5.1).

2 Related Work

Each of our tasks (PS—phrase similarity; PR—phrase retrieval; and PSD—disambiguation) is related to a separate research area discussed below.

Phrase similarity First, most existing phrase similarity datasets—e.g. PPDB-annotated (Wieting et al., 2015), PPDB-filtered (Wang et al., 2021), BiRD (Asaadi et al., 2019), and PAWS-short (Wang et al., 2021; Zhang et al., 2019)—contain a large percent of instances with lexical overlap between two paired phrases while our PS contains the least percent (5.34%; Table 1). Second, PS compares each pair of phrases in a context sentence while existing datasets only compare phrases alone (no context). Third, the phrases in PS are, on average, 2-token long, comparable to that of other datasets (Table 1). Fourth, unlike other datasets, PS contains exclusively *noun-phrases*, the most common phrase type according to Yahoo's search-query statistics (Yahoo, 2022a) (79.54%; Appendix D) and Adobe (internal Acrobat Pro data not shown).

Question answering (QA) Our phrase-retrieval tasks—PR and PSD—follow the format of QA datasets except that our queries are *phrases* instead of questions and hence shorter (Table 2). Like SQuAD 1.1 (Rajpurkar et al., 2016) and HotpotQA (Yang et al., 2018), our documents and queries are extracted from Wikipedia articles. While our PR dataset is ∼3.5× smaller than those two datasets, the paragraph document length in PR-pass and PSD is ∼2× longer than those of SQuAD 1.1 and HotpotQA (Table 2). For our task, intuitively, the longer the document, the harder the task since there would be more candidates a model must compare with the query.

Table 1: Our Phrase Similarity (PS) dataset has a lower percent of lexical-overlap instances and is the *only* human-annotated dataset that provides phrases, each in a context sentence.

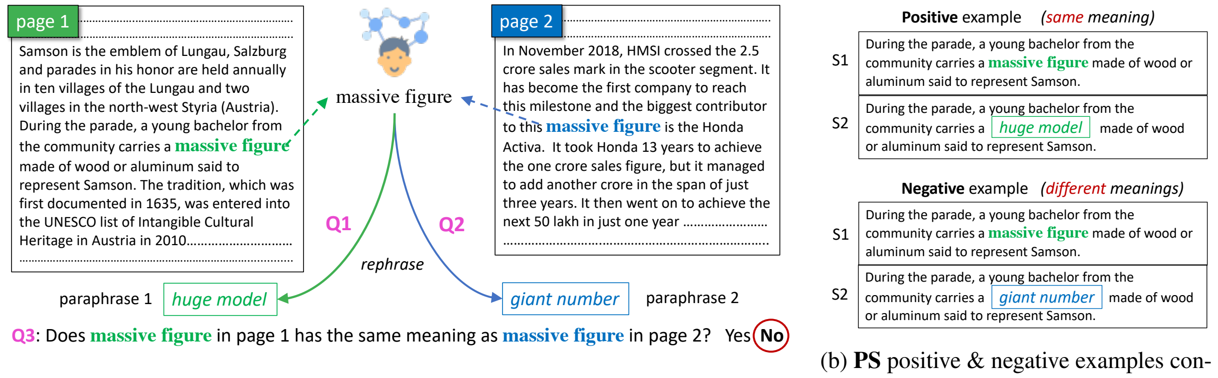
	PS (ours)	WiC	PPDB-annotated	PPDB-filtered	BiRD	Turney	PAWS-short
# of All instances	10,004	7,466	3,000	15,532	3,345	2,180	1,214
# of Unique phrases	7,488	2,345	6,000	12,023	2,840	9,776	1,214
Lexical overlap (%)	5.34	100	70.10	97.93	14.98	0	99.42
Mean length (in tokens)							
• phrase_1	2.06	1	3.67	2	2	2	9.52
• phrase_2	2.46	1	3.73	2	1.49	1	9.42
• context sentence	22.53	8.40	0	0	0	0	0

Table 2: Our PR-pass, PR-page and PSD datasets are smaller in size compared to common QA datasets and contain shorter queries that are noun phrases instead of questions. However, our tasks require searching in much longer documents.

	PR-pass	**PR-page**	**PSD**	SQuAD 1.1	HotpotQA
All instances	28,147	28,098	4,858	98,169	105,257
Unique queries/questions	27,055	27,016	4,812	97,888	105,249
Unique answers	13,458	13,423	2,314	72,469	57,259
Mean length of					
query (tokens)	2.42	2.42	2.45	11.42	20.03
answer (tokens)	2.17	2.17	2.07	3.46	2.35
sentence (tokens)	23.22	24.08	23.00	27.62	26.77
document (sentences)	10.26	119.32	20.37	5.10	4.14
document (tokens)	238.34	2,872.73	468.48	140.92	110.72

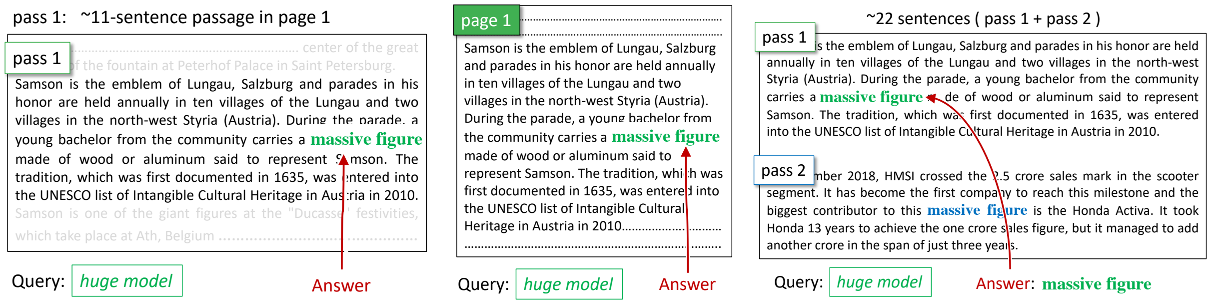
Sense disambiguation While word-sense disambiguation (WSD) is a long-standing problem in NLP, recently, SotA models have reached super-human accuracy (80% F_1) on the common English WSD (Bevilacqua et al., 2021). Interestingly, these high-scoring models still struggle with rare senses that may be outside of the predefined sense inventories or have few training examples (Blevins et al., 2021). Without the need for predefined senses, WiC (Pilehvar and Camacho-Collados, 2019) poses disambiguation as a binary classification task where the goal is to predict whether the same target word in two different sentences carries the same or different meanings.

Compared to WiC PS is also a binary classification task, but with two major differences: (1) in



(a) Q1 & Q2 ask annotators to rephrase "massive figure" in page 1 and page 2. Q3 asks whether this phrase's meaning is the same in both pages.

(b) **PS** positive & negative examples constructed using page 1 context (similarly, we repeat for page 2).



(c) A **PR-pass** example.

(d) A **PR-page** example.

(e) A **PSD** example.

Figure 1: Given a phrase, two associated Wikipedia pages, and expert annotations, i.e. answers to Q1, Q2, and Q3 (a), we are able to construct *two* pairs of positive and negative examples for PS (b), a PR-pass example (c), a PR-page example (d), and a PSD example *only if* the answer to Q3 is No (e).

WiC, the same target word appears in two different sentences while in PS, two different phrases appear in the same context sentence; (2) PS compares phrases composed of ≥ 2 words instead of a single word as in WiC and WSD. While word senses are defined in WordNet and BabelNet dictionaries (Bevilacqua et al., 2021), there are no English dictionaries of senses for multi-word noun phrases (mNPs). Thus, it is more challenging to acquire and learn the senses of mNPs, hence the importance of our PiC dataset. Like WiC, PSD tests disambiguating the meanings of the same n-gram in two different contexts. Yet, PSD is a phrase search task, which involves many more phrase comparisons per example than PS or WiC.

Before the deep learning era, phrase-sense disambiguation was already proposed (Carpuat and Wu, 2007a,b) but only as an auxiliary task for training machine-translation models. And their phrase senses were not annotated by humans but inferred by performing word-alignment on a bilingual corpus. Here, our PSD is the first phrase-sense disambiguation task annotated by experts and requires understanding of phrase-senses in a passage.

3 PiC Dataset Construction

We first collect a set of phrases with context and human annotations. Then, we derive the examples and labels for three main tasks: PS, PR, and PSD (Fig. 1). Our idea is to mine a set of triplets ($\boldsymbol{p}$, page$_1$, page$_2$) from Wikipedia where the phrase $\boldsymbol{p}$ is a polysemous mNP that carries two different senses in two Wikipedia pages (e.g., "massive figure" means *a large number* in page$_1$ but *a huge physical shape* in page$_2$; Fig. 1a). Then, we ask experts to rephrase $\boldsymbol{p}$ into two paraphrases $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$, maintaining the two original senses of $\boldsymbol{p}$ in page$_1$ and page$_2$, respectively. The resultant set of 5-tuples ($\boldsymbol{p}$, $\boldsymbol{q}_1$, $\boldsymbol{q}_2$, page$_1$, page$_2$) enables the tests for (1) comparing the semantic similarity of two phrases given the same context sentence (PS; Fig. 1b); (2) finding a semantically similar phrase in a document (PR-pass & PR-page; Fig. 1c); (3) disambiguating the senses of the same target mNP given two context paragraphs (PSD; Fig. 1e).

3.1 Data Collection

As there are no English dictionaries that contain sense inventories for mNPs, the key **challenge** to our data collection is to mine mNPs that have (1) multiple senses; and (2) a Wikipedia context page for each sense. To do that, we take a Wikipedia dump and perform a **6-step** procedure that essentially extracts all the mNPs that occur in more than one Wikipedia page and that contain at least one polysemous word defined in the WiC dataset. From the triplets of ($\boldsymbol{p}$, page$_1$, page$_2$), we programmatically narrow down to ∼600K triplets where the context sentence of the mNP in page$_1$ is the most semantically *dis*similar to the context sentence in page$_2$ (according to SimCSE (Gao et al., 2021)). We continue filtering down to the top 19,500 triplets where page$_1$ and page$_2$ have *the most* semantically dissimilar lists of Wikipedia categories. That is, 19,500 triplets are estimated to yield ∼15K annotated triplets (the target size based on our budget) after the human annotation process where annotators are allowed to skip the cases they are not confident labeling. See Appendix C for a detailed description of the data collection and **dataset biases**.

3.2 Data Annotation

Via Upwork, we hire 13 linguistics experts who are native English speakers at a rate of \$30/hour to annotate 15,021 out of 19,500 examples. For each phrase, we provide Upworkers with a triplet ($\boldsymbol{p}$, passage$_1$, passage$_2$) where each passage$_i$ consists of 5 sentences centered at the phrase-containing sentence in the corresponding page$_i$. We ask them to answer the three below questions (Fig. 1a):

Q1 Rephrase the target phrase $\boldsymbol{p}$ to a paraphrase $\boldsymbol{q}_1$ such that its meaning is constant in passage$_1$.

Q2 Similarly, rephrase $\boldsymbol{p}$ w.r.t. passage$_2$ to obtain a paraphrase $\boldsymbol{q}_2$.

Q3 Answer Y/N if $\boldsymbol{p}$ has the same meaning in both contextual passage$_1$ and passage$_2$.

Upworkers are asked to provide paraphrases that (1) have at least two words and (2) minimize lexical overlap with each other and the target $\boldsymbol{p}$. See the annotation guidelines (PiC, 2021a) and a sample annotation assignment (PiC, 2021b) given to Upworkers. After receiving annotations, we use LanguageTool (2022) to automatically find syntactical errors when the paraphrases are replaced by the original target phrase in the original passage and ask Upworkers to fix them. We also have annotators fix the remaining errors that we find via manual inspection.

3.3 Annotation Verification

To verify the annotations obtained in Sec. 3.2 (i.e. 2 × 15,021 = 30,042 paraphrases; and 15,021 Y/N labels), first, we present the same Q1, Q2, and Q3 questions to 1,000 qualified MTurkers and ask whether they agree with the answers by expert annotators in Sec. 3.2. And then, for the cases that the MTurkers disagree with, we seek second opinions from 5 Upwork experts. After these two verification rounds, we discard all the examples where Upwork verifiers reject and arrive at the final 28,325 paraphrases and 13,413 Y/N labels (i.e. those annotations that *either* an MTurk or Upwork verifier endorses). See more details in Appendix J.

The total fee for both MTurk and Upwork combined is around USD 30,000.

4 Three Phrase Understanding Tasks

Using the human-annotated data, we construct three tasks of PS, PR, and PSD (as summarized in Fig. 1) for evaluating contextualized phrase-embeddings and semantic-search models.

4.1 Phrase Similarity (PS)

PS is a binary classification task that asks whether two mNPs are semantically similar or not given *the same context* sentence. The unique challenge of PS is that, *without* context, the two given phrases can be easily interpreted as synonymous. Yet, in our PS context sentence, the two phrases *may* or may *not* carry distinct meanings (Fig. 1b).

Construction From the annotated data, a positive example is a triplet of (an original phrase $\boldsymbol{p}$, a paraphrase $\boldsymbol{q}_1$, an original page$_1$'s sentence that contains $\boldsymbol{p}$). To create a negative example, from the same triplets, we select only those where the paraphrase $\boldsymbol{q}_2$ holds a *different* meaning than $\boldsymbol{q}_1$ given the page$_1$ context of $\boldsymbol{q}_1$ (i.e., when the answer to Q3 is No; see Fig. 1b). For quality assurance, we also hire three extra Upwork experts to double-check PS annotations (see Appendix E), keeping only examples that at least 2 out of 3 experts endorse. In total, we obtain 5,002 *negative* examples. Then, we randomly select 5,002 *positive* examples to form a class-balanced PS dataset.

4.2 Phrase Retrieval (PR)

PR is a task of finding in a given document $\boldsymbol{d}$ a phrase $\boldsymbol{p}$ that is semantically similar to the given query phrase, which is the paraphrase $\boldsymbol{q}_1$ (the answer by annotators to Q1) or $\boldsymbol{q}_2$ (the answer to

Q2). We release two versions of PR: **PR-pass** and **PR-page**, i.e. datasets of triplets (query $\boldsymbol{q}_1$, target phrase $\boldsymbol{p}$, document $\boldsymbol{d}$) where $\boldsymbol{d}$ is a random 11-sentence passage that contains $\boldsymbol{p}$ (Fig. 1c) or an entire Wikipedia page (Fig. 1d). While PR-pass contains 28,147 examples, PR-page contains slightly fewer examples (28,098) as we remove those examples whose Wikipedia pages coincidentally also contain exactly the query phrase (in addition to the target phrase). Both datasets are split into ~20K/3K/5K for train/dev/test, respectively.

4.3 Phrase Sense Disambiguation (PSD)

The task is to find the location of the target phrase $\boldsymbol{p}$ where it has a similar meaning to that of the given query $\boldsymbol{q}$ in a 2-paragraph document where, by construction, $\boldsymbol{p}$ appears exactly twice but only one location is the correct answer (Fig. 1e).

Construction From the verified annotations in Sec. 3.3, there are in total 4,938 phrases that both annotators and verifiers agree to hold *different* meanings across the two context Wikipedia pages (i.e., "No" answer to Q3 in Fig. 1a). To create a PSD example, given a phrase $\boldsymbol{p}$ from the above 4,938, we extract two corresponding ~11-sentence paragraphs (from its associated page$_1$ and page$_2$ as in PR-pass) and concatenate them (separated by an empty line) into a single document (Fig. 1e). Since a PSD example shares a pair of phrases (*query* and *answer*) with one PS *positive* example (*phrase*$_1$ and *phrase*$_2$), we filter out that PSD example if the corresponding PS example is removed from the additional verification round (Appendix E). As the result, we exclude 80 examples and obtain 4,858 examples in total for PSD.

5 Experiments and Results

We test SotA models on PS, PR-pass, PR-page, and PSD to (1) assess how the models are able to leverage context to improve accuracy; and (2) quantify the headroom for future research.

Phrase embeddings Besides training and testing SotA BERT-based classifiers, we also test a ranking approach that involves computing the cosine similarity between the query's and each candidate's embedding. To compute a *contextualized* phrase embedding, following Yu and Ettinger 2020, we feed the entire phrase-containing sentence (e.g. S$_1$ in Fig. 1b) into a model, e.g. BERT, and then take the mean pooling of the last-layer embeddings over the words of the given phrase only. For non-contextualized phrase embeddings, we repeat the same process but input to the model only the phrase (instead of the entire sentence).

Models We choose SotA models in (a) phrase similarity: PhraseBERT (Wang et al., 2021); (b) sentence similarity: USE-v5 (Cer et al., 2018), SentenceBERT (Reimers and Gurevych, 2019), and SimCSE (Gao et al., 2021)); (c) question-answering: Longformer (Beltagy et al., 2020), DensePhrase (Lee et al., 2021); and (d) contextualized embeddings: SpanBERT (Joshi et al., 2020) and BERT (Devlin et al., 2019).

For DensePhrase, we use their Phrase-Encoder (as opposed to the Query-Encoder) to compute phrase embeddings. USE-v5 is only available via public APIs (TensorFlow, 2022) that do not support extraction of contextualized phrase embeddings.

5.1 Phrase Similarity: Contextualized phrase embeddings improve accuracy

Q: *Does incorporating context improve the phrase-similarity accuracy on PS?*

Experiment We split the PS dataset 70/10/20 for train/dev/test and test two approaches: (1) using the cosine similarity score between two pre-trained phrase-embeddings (with and without context) to predict phrase similarity; (2) training BERT-based binary classifiers directly using PS training set. We use 6 backbone BERT models that are all "base" versions unless specified otherwise (Table 3).

Approach 1: *Cosine similarity* First, we test how *pre-trained* phrase embeddings alone (without finetuning or extra weights) can be leveraged to solve PS. For each PS example of two phrases, we compute their non-contextualized phrase embeddings and compute their cosine similarity score. To evaluate the pre-trained embeddings on PS, we follow Yang (2022) and tune the binary-classification threshold T to maximize the training-set accuracy, and then use the same optimal T to report the test-set accuracy. We repeat the experiment for *contextualized* phrase embeddings.

Approach 2: *BERT-based classifiers* To complement Approach 1, we test Approach 2, i.e. building a binary classifier by adding two extra MLP layers on top of the pre-trained embeddings used in Approach 1. For a phrase pair, we concatenate the two 768-D phrase embeddings from BERT$_{\text{base}}$ into a 1,536-D vector, and then place one ReLU layer (256 units) and a 1-output linear classification layer with sigmoid on top. Following Wang et al.

Table 3: Accuracy (%) of state-of-the-art BERT-based models on the PS test set. Contextualized phrase embeddings ("Phrase + Ctx") yield substantially higher performance on PS than non-contextualized embeddings ("Phrase"). The random baseline is 50%.

Model	*Approach 1: Cosine similarity*		*Approach 2: BERT-based classifiers*	
	(a) Phrase	(b) Phrase + Ctx	(c) Phrase	(d) Phrase + Ctx
PhraseBERT	51.75	63.40 (+11.65)	33.60	66.10 (+32.50)
BERT	51.05	64.10 (+13.05)	37.00	68.85 (+31.85)
SpanBERT	49.30	64.00 (+14.70)	40.15	66.85 (+26.70)
$\text{SpanBERT}_{\text{Large}}$	50.40	**66.30** (+15.90)	35.95	**69.25** (+33.30)
SentenceBERT	50.35	60.30 (+9.95)	31.50	62.55 (+31.05)
SimCSE	52.15	62.50 (+10.35)	34.20	66.65 (+32.45)
mean ± std	50.83 ± 1.04	**63.43** ± **1.98**	35.40 ± 3.01	**66.71** ± **2.40**

(2021), we finetune these models for a maximum of 100 epochs (with early stopping and patience of 10 epochs) on the train set. See Appendix A for more training details.

Results *Without* context, all models perform at ≤ 50% accuracy (i.e. the random chance; Table 3a & c). Interestingly, incorporating context information into phrase embeddings substantially improves mean model-accuracy on PS for both Approach 1 (from 50.83% to 63.43%; Table 3b vs. a) and Approach 2 (from 35.40% to 66.71%; Table 3d vs. c), showing evidence that **PS requires models to rely on context**. While starting from the same backbone models, Approach 2 yields higher mean accuracy than Approach 1 (Table 3; 66.71 vs. 63.43), which is expected as Approach 2 models have more capacity and the backbones are allowed to be finetuned on PS. See Figs. A3–A6 for qualitative PS predictions from a PhraseBERT-based classifier.

5.2 Human Baselines and Upperbound (95% Exact Match) on Phrase Retrieval

To interpret the progress of machine phrase-understanding on PR, here, we establish multiple human baselines for both non-experts and linguistics experts (with and without training them).

Experiment We recruit participants and have them perform one or two tests per person. A test consists of 20 PR-pass examples. That is, PR-pass documents are 11-sentence long and are feasible for a person to read in minutes (compared to reading an entire Wikipedia page). We test three groups: (1) 21 graduate students at our institution (1 test per person); (2) five Upwork experts (1 test per person); and (3) another five Upwork experts (2 tests per person, i.e., for a total of 2 × 5 = 10 tests).

Table 4: Best SS models reach near the Upperbound (95%) on PR-pass. Yet, ranking models based on phrase embeddings significantly *underperform* SS models.

Accuracy of human groups and models	EM (%)
Group 1: 20 Non-experts (w/o training)	73.60 ± 7.90
Group 2: 05 Experts (w/o training)	82.00 ± 12.00
Group 3: 05 Experts (w/ training)	**90.50** ± 3.70
Best human accuracy (4 people)—Upperbound	**95.00** ± 0.00
Best untrained, ranking model (BERT)	47.44
Best PR-trained, ranking model (PhraseBERT)	59.02
Best PR-trained, SS model ($\text{Longformer}_{\text{Large}}$)	**94.28**

The students in Group 1 volunteer to help our study unpaid while the Upworkers (Group 2 and 3) are hired using the same procedure as in Sec. 3.2.

Results First, we find an unsurprising, large gap between non-experts and experts (Table 4; 73.60% vs. 82.00%). Second, we train experts in Group 3 by having each do a preliminary test and giving them feedback before the real test. We find the training to substantially boost expert accuracy further (from 82.00% to 90.50%). Importantly, we find the Human Exact Match (EM) Upperbound to be 95%, i.e. the highest scores that 4 people (among all groups) make. Upon manual inspection of the submissions of these best performers, we find their incorrect answers sometimes partially overlap with the groundtruth or are sometimes reasonable. In other cases, the best performers find acceptable answers but that do not overlap at all with the groundtruth labels in PR. That is, we estimate a 5% of noise in the annotations of PR.

5.3 Phrase Retrieval: In ranking, context only helps BERT embeddings but not others

One way to evaluate the quality of SotA phrase embeddings is by testing:

Q: *How well do phrase embeddings perform in the **ranking** approach on PR?*

Ranking is a challenging and meaningful phrase-embedding test because the embedding of the query is compared against that of all phrase candidates (extracted by tokenizing the document), which can include syntactically-incorrect phrases, meaningless phrases or rare phrases. Such out-of-distribution challenge appears less often in PS or WiC, i.e. a binary classification setting.

Experiment As described in Sec. 4.2, the PR train/dev/test splits are 20,147/3K/5K examples and we only use the 5K-example test set to test the models in this ranking experiment (no training). We follow (Lee et al., 2017) for span enumeration to construct a list of candidate phrases, we split each PR document into multiple sentences (using NLTK sentence splitter) and tokenize each sentence into tokens (using NLTK tokenizer) and build an exhaustive list of n-grams (here, $n \in \{2, 3\}$ only for computational tractability). For every example, we add the groundtruth phrase (which can be longer than 3 words) to the list of candidates (since we are only interested in testing phrase embeddings, not the phrase extractor).

Results We report top-k accuracy (for k = 1, 3, 5) and top-5 Mean Reciprocal Rank (MRR@5) on the PR-pass test set in Table 5a. First, for most SotA embeddings, incorporating context sentence *hurts* the accuracy (*except for BERT* embeddings). That is, interestingly, for all BERT embeddings (base and large), the accuracy increases substantially (+17.64 and +19.04; Table 5) when the one-sentence context is the input. In contrast, most models that started from BERT but were later fine-tuned lost the capability to leverage the context information (e.g., PhraseBERT, DensePhrase, and SpanBERT in Table 5).

Second, the best top-1 accuracy scores on PR-pass for non-contextualized (USE-v5; 43.36%) and contextualized (BERT; 47.44%) embeddings are substantially lower than the non-expert baselines (73.60%; Table 4) and Human Upperbound (95%). Future work is required to learn more robust, phrase embeddings for ranking. See Figs. A10–A11 for qualitative examples.

5.4 Phrase Retrieval: Span-selection models reach near-human accuracy

Consistent with Yu and Ettinger (2020), our ranking results in Sec. 5.3 reveal that there exists a large headroom for improving both non-contextualized and contextualized phrase embeddings. Yet, because ranking is a naive approach and SS models (Huggingface, 2022b; Devlin et al., 2019) are the SotA approach on many QA tasks (Rajpurkar et al., 2016), here we train SS models on the train set of PR-pass and PR-page in order to test:

Q: *How well do SotA semantic-search models perform on PR-pass and PR-page?*

Experiment We take the SotA embeddings tested in Sec. 5.3 and add a linear classification layer on top and finetune each entire classifier on the train set of PR-pass or PR-page for 2 epochs using the default HuggingFace hyperparameters (see Appendix B for finetuning details). Following the standard setup of BERT architectures for QA tasks (Devlin et al., 2019), each SS model predicts the start and end index of the target phrase. Additionally, since PR-page documents are much longer than a typical QA paragraph (Table 2), we also test training Longformer (Beltagy et al., 2020), which has a max sequence-length of 4,096, sufficient for an entire Wikipedia page. We take the models of the smallest dev loss and report their test-set performance in Table 6.

Results On PR-pass, in contrast to the poor performance of ranking models (Sec. 5.3), our PR-pass-trained SS models perform impressively at a near-upperbound level (∼93–94% EM; Table 6a) surpassing the accuracy of trained experts (90.50% EM). Surprisingly, on PR-page where the documents are substantially longer (around 12×) than the documents of PR-pass, SS models' accuracy only drops slightly (from ∼94% to ∼85–89% EM; Table 6b). Note that in a full Wikipedia page of PR-page, there might be phrases that can be considered correct but are *not* labeled groundtruth according to our annotations. This remarkable result suggests that training on PR-pass can enable high-performing models on real-world semantic search.

5.5 Phrase Sense Disambiguation: Best models also perform poorly

We find that SotA PR-pass-trained SS models reach superhuman accuracy on PR-pass, i.e. finding a phrase of the same meaning (Sec. 5.4). Yet, PR-pass only tests models' understanding of a *single sense* of the target phrase at a time. It is interesting to study:

Q: *Do PR-pass-trained SS models understand contextualized phrases sufficiently to separate two different senses of the same target phrase?*

Experiment To do that, here we test the best PR-pass-trained SS models on PSD. Note that, PSD has the same task format as PR-pass (see Fig. 1c–e) except that the document is twice as long and contains **two occurrences of the same target phrase**. We do not test the ranking models as they perform much worse than the SS models in Sec. 5.3.

Results Although the PR-pass-trained SS models are never trained on PSD, they interestingly frequently find one occurrence of the target phrase (mean of 94.01% EM; Table 6c). However, they mostly locate the **target phrase in the wrong context passage** with high confidence scores. That is, if we consider also the correctness of the loca-

Table 5: **Ranking** accuracy (%) on **PR-pass** using the state-of-the-art pretrained phrase embeddings. See Appendix F for the results on PR-page. Δ (e.g. -3.62) denotes the differences between the Top-1 accuracy in the contextualized ("Phrase + Context") vs. the non-contextualized ("Phrase") setting.

Model	Phrase				Phrase + Context			
	Top-1	Top-3	Top-5	MRR@5	Top-1 (Δ)	Top-3	Top-5	MRR@5
PhraseBERT (Wang et al., 2021)	36.62	66.96	75.90	52.20	33.00 (-3.62)	49.60	56.70	41.90
BERT (Devlin et al., 2019)	29.80	47.90	55.40	39.50	**47.44** (+17.64)	**65.78**	**73.30**	**57.30**
BERT$_{\text{Large}}$ (Devlin et al., 2019)	23.76	38.52	45.40	31.70	**42.80** (+19.04)	**58.90**	**64.90**	**51.30**
SpanBERT (Joshi et al., 2020)	20.88	31.04	35.20	26.40	14.40 (-6.48)	30.46	39.80	23.40
SentenceBERT (Reimers and Gurevych, 2019)	22.30	50.64	60.60	36.80	**25.14** (+2.84)	39.52	46.20	32.90
SimCSE (Gao et al., 2021)	28.10	53.70	64.60	41.60	**32.40** (+4.30)	53.44	62.80	**43.70**
USE-v5 (Cer et al., 2018)	**43.36**	**70.12**	**78.90**	**57.30**	n/a	n/a	n/a	n/a
DensePhrase (Lee et al., 2021)	32.24	51.30	60.50	42.60	31.50 (-0.74)	46.30	53.80	39.70

Table 6: **Test**-set performance (%) of **SS models** on PR-pass (a), PR-page (b), and PSD (c). When trained on PR-pass (a) and PR-page (b), SotA SS models perform well. However, testing the PR-pass-trained models on PSD shows a significant drop in accuracy (c). That is, SotA SS models tend to understand a *single sense* of a phrase in context well (high PR-pass, PR-page, and PSD EM scores). Yet, they are not able to differentiate two senses of the same phrase (e.g., here, PhraseBERT accuracy drops -41.27 points between EM+loc vs. EM scores on PSD).

Model	(a) PR-pass		(b) PR-page		(c) PSD			
	EM	F$_1$	EM	F$_1$	EM	F$_1$	EM+loc	F$_1$+loc
PhraseBERT (Wang et al., 2021)	93.42	94.97	85.24	87.19	92.98	94.08	51.67 (-41.31)	51.83
BERT (Devlin et al., 2019)	93.26	94.65	85.64	87.77	93.50	94.57	54.84 (-38.66)	55.07
BERT$_{\text{Large}}$ (Devlin et al., 2019)	93.64	95.16	87.36	89.52	94.67	95.57	55.43 (-39.24)	55.61
SpanBERT (Joshi et al., 2020)	93.50	95.02	87.28	87.66	92.26	93.30	52.20 (-40.06)	52.34
SentenceBERT (Reimers and Gurevych, 2019)	93.24	94.54	84.66	86.89	93.21	94.15	52.74 (-40.47)	52.85
SimCSE (Gao et al., 2021)	92.90	94.51	85.68	87.66	92.96	94.05	53.83 (-39.13)	53.94
Longformer (Beltagy et al., 2020)	94.26	**95.58**	**89.54**	**91.15**	96.17	96.88	**62.72** (-33.45)	**62.83**
Longformer$_{\text{Large}}$ (Beltagy et al., 2020)	**94.28**	95.53	87.58	89.32	**96.32**	**96.91**	59.72 (-36.60)	59.82
mean	93.56	95.00	86.92	88.85	94.01	94.94	55.39 (-38.62)	55.54
± std	0.49	0.42	1.93	1.73	1.54	1.36	3.90	3.88

tion of the predicted phrase, their EM+loc[2] accuracy drops significantly to an average of 55.39%. Also, finetuning on a 2K-example train set of PSD only slightly improves the EM+loc to an average of 64.24% on a 3K-example PSD test set (Appendix G). Note that we estimate the Human Upperbound on PSD to be 95%, i.e. the same as that of PR-pass. See qualitative examples and predictions of Longformer (i.e. the best model tested) in Figs. A7–A9.

In sum, there is a large headroom for future research on PSD. SS models are *not* yet capable of leveraging surrounding words to differentiate between two senses of the same phrase. Interestingly, after training on PR-pass, their contextualized phrase embeddings perform much worse in the ranking experiments on PR-pass (Appendix H).

6 Discussion and Conclusion

While WiC and English WSD rely exclusively on dictionaries (Pilehvar and Camacho-Collados, 2019) to obtain word senses and example sentences, our data collection depends on Wikipedia, WiC, & NLP models and our annotation depends on experts. In sum, we present PiC, the first 3-task suite for

---

[2] For a PSD example, if the predicted span does not intersect at all with the groundtruth span, the EM+loc and F$_1$+loc scores would be 0. If they intersect, the two scores would be equal to EM and F$_1$, respectively.

evaluating phrases in context. SS models can obtain high accuracy on semantic search after training on our PR-pass and PR-page datasets. Yet, their capability is limited to finding a semantically-similar phrase given a single context that contains the target phrase (in PR-pass). The results on PS and PSD show that SotA phrase embeddings are still limited in encoding contextualized phrases. It is interesting future work to improve these models for disambiguating the senses of a phrase in context (PS and PSD).

7 Limitations

Our dataset is currently limited to multi-word, English noun-phrases. Furthermore, it is expected to contain around a 5% error on PR-pass (i.e. the best human performance is 95% EM). On PR-page, there may be more than one correct target phrase; however, we only label one phrase as the correct answer per document. We use only phrases that contain at least one WiC word.

Acknowledgement

We thank Qi Li, Peijie Chen, Hai Phan, Giang Nguyen, and Naman Bansal from Auburn University for helpful feedback on the early results. We also thank graduate students from AN's Deep Learning class for helping us with the human study. We are grateful for the valuable support and feedback from Phat Nguyen. AN is supported by NaphCare Foundations, Adobe gifts, and NSF grants (1850117, 2145767).

References

Tarik Arici, Hayreddin Ceker, and Ismail Baha Tutar. 2020. Multi-span question answering using span-image network. *preprint*.

Shima Asaadi, Saif Mohammad, and Svetlana Kiritchenko. 2019. Big BiRD: A large, fine-grained, bigram relatedness dataset for examining semantic composition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 505–516, Minneapolis, Minnesota. Association for Computational Linguistics.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv preprint*, abs/2004.05150.

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, Roberto Navigli, et al. 2021. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. International Joint Conference on Artificial Intelligence, Inc.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, Beijing.

Terra Blevins, Mandar Joshi, and Luke Zettlemoyer. 2021. FEWS: Large-scale, low-shot word sense disambiguation with the dictionary. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 455–465, Online. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Marine Carpuat and Dekai Wu. 2007a. How phrase sense disambiguation outperforms word sense disambiguation for statistical machine translation. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages: Papers*, Skövde, Sweden.

Marine Carpuat and Dekai Wu. 2007b. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *ArXiv preprint*, abs/1803.11175.

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

Philip Edmonds and Scott Cotton. 2001. SENSEVAL-2: Overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5, Toulouse, France. Association for Computational Linguistics.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: the concept revisited. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, pages 406–414. ACM.

Ute Fischer. 1994. Learning words from context and dictionaries: An experimental comparison. *Applied Psycholinguistics*, 15(4):551–574.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Princeton NLP Group. 2022. princeton-nlp/sup-simcse-roberta-large · hugging face. `https://huggingface.co/princeton-nlp/sup-simcse-roberta-large`. (Accessed on 06/08/2022).

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. *https://spacy.io/*.

Huggingface. 2022a. super_glue · datasets at hugging face. `https://huggingface.co/datasets/super_glue/viewer/wic`. (Accessed on 06/08/2022).

Huggingface. 2022b. transformers/examples/pytorch/question-answering at main · huggingface/transformers. `https://github.com/huggingface/transformers/tree/main/examples/pytorch/question-answering`. (Accessed on 06/09/2022).

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

LanguageTool. 2022. Languagetool - online grammar, style & spell checker. `https://languagetool.org/`. (Accessed on 06/08/2022).

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning dense representations of phrases at scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6634–6647, Online. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

William E Nagy, Patricia A Herman, and Richard C Anderson. 1985. Learning words from context. *Reading research quarterly*, pages 233–253.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing, China. Association for Computational Linguistics.

PiC. 2021a. upwork_annotation_guidelines.pdf. `https://drive.google.com/file/d/1UsWXvJLzWzuhoRuaCsla3Ljcxp-vM2KK/view?usp=sharing`. (Accessed on 06/08/2022).

PiC. 2021b. upwork_samples.pdf. `https://drive.google.com/file/d/1ume4QeQvEPlwajyeNIQjdH_cCI9w3P6f/view?usp=sharing`. (Accessed on 06/08/2022).

PiC. 2022. upwork_samples.pdf. `https://drive.google.com/file/d/1Lplc_cRla_hbDFzLiJ7ZnO6GOGnRq1gc/view?usp=sharing`. (Accessed on 06/08/2022).

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Wikimedia Team. 2021a. Api:categories - mediawiki. `https://www.mediawiki.org/wiki/API:Categories`. (Accessed on 06/08/2022).

Wikimedia Team. 2021b. Wikimedia downloads. `https://dumps.wikimedia.org/`. (Downloaded on November 1st, 2021).

TensorFlow. 2022. Universal sentence encoder | tensorflow hub. `https://www.tensorflow.org/hub/tutorials/semantic_similarity_with_tf_hub_universal_encoder`. (Accessed on 08/11/2022).

Peter D Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of artificial intelligence research*, 44:533–585.

Shufan Wang, Laure Thompson, and Mohit Iyyer. 2021. Phrase-BERT: Improved phrase embeddings from BERT with an application to corpus exploration. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10837–10851, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*, 3:345–358.

Yahoo. 2022a. Webscope | yahoo labs. `https://webscope.sandbox.yahoo.com/catalog.php?datatype=l&did=66`. (Accessed on 08/10/2022).

Yahoo. 2022b. Webscope | yahoo labs. https://webscope.sandbox.yahoo.com/catalog.php?datatype=l&did=66. (Accessed on 08/10/2022).

Ronghui Yang. 2022. arcface-pytorch/test.py at master · ronghuaiyang/arcface-pytorch. https://github.com/ronghuaiyang/arcface-pytorch/blob/master/test.py. (Accessed on 06/09/2022).

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692, Hong Kong, China. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Lang Yu and Allyson Ettinger. 2020. Assessing phrasal representation and composition in transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4896–4907, Online. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Appendix for:
PiC: A Phrase-in-Context Dataset for Phrase Understanding and Semantic Search

A Training models on Phrase Similarity

Hyperparameters We train each *BERT-based classifier* for a maximum of 100 epochs with early stopping monitored on validation accuracy (patience of 10 epochs). We use a batch size of 200 and Adam optimizer with learning rate α = 0.0001, β_1 = 0.9, β_2 = 0.999, and $\epsilon = 10^{-8}$.

Training time On average, with early stopping, training a single model using one V100 GPU takes $\sim$5 and $\sim$8 mins for non-context and context settings, respectively.

B Training SS models on Phrase Retrieval

We finetune each SS model that consists of a linear layer on top of a pretrained model selected in Sec. 5 to predict the start and end indices of answers (as the common setup in BERT SS models (Devlin et al., 2019; Arici et al., 2020)). The format of a tokenized input is "[CLS] query [SEP] document [SEP]" with maximum sequence length of 4,096 for $\text{Longformer}_{\text{Base}}$ and $\text{Longformer}_{\text{Large}}$ and 512 for the remaining models. If the document exceeds the maximum sequence length, it is split into smaller features for prediction and thus start and end indices with the highest confidence scores are selected.

Hyperparameters We follow HuggingFace scheme to finetune the SS models for 2 epochs using Adam optimizer with learning rate α = 0.00003, β_1 = 0.9, β_2 = 0.999, $\epsilon = 10^{-8}$. The batch size varies from 1 to 8 for each model: On one V100 GPU, the "base" models can handle 8 examples while the "large" BERT models can only fit 2–4 examples into 16GB of memory. For $\text{Longformer}_{\text{Large}}$, we use an A100 GPU to feed one PR-page example into the model. We take the smallest dev-loss models from the training and report their test-set results.

Training time On average, training a single SS model for 2 epochs using one A100 GPU takes $\sim$20 mins for base models and $\sim$9.5 hours for $\text{Longformer}_{\text{Large}}$.

C Data collection

From a Wikipedia dump, we perform a 6-step procedure (summarized in Table A1) for mining a list of mNPs sorted descendingly by their likelihood of containing multiple senses. The most polysemous 19,500 mNPs are then passed to experts for annotation (Sec. 3.2) and others for verification (Sec. 3.3).

Step 1: Download Wiki articles We download a Wikipedia dump file (Team, 2021b) that contains $\sim$15.78M Wikipedia articles and filter out all empty pages to arrive at $\sim$6.27M non-empty articles.

Step 2: Extract phrases We use NLTK sentence splitter (Bird et al., 2009) to split each Wikipedia article into multiple sentences. And then we use SpaCy (Honnibal et al., 2020) to extract noun phrases and proper nouns as we do not collect syntactically strict phrases. For each phrase, we remove all preceding and succeeding stopwords (those among the 179 stopwords in NLTK v3.6.5) and non-alphanumeric characters. We remove stopwords because they tend to create more pairs of phrases with lexical overlap, rendering the phrase similarity task easier. We then remove *unigram* phrases to arrive at ∼286.78M mNPs. For example, from "a massive figure", we changed to "massive figure", which would be our final phrase after this step. For each mNP, we construct a 3-tuple (phrase, sentence, metadata), i.e. the phrase, its container sentence, and metadata for identifying the Wikipedia webpage (hereafter, page).

Step 3: Remove phrases of a single context We further remove all phrases that (1) contain non-ASCII characters (e.g. "phaenná nâsos", which are non-English); and (2) appear only once, i.e. keeping those that occur in multiple sentences since we look for *polysemous* mNPs, which have multiple senses and contexts. After this step, ∼17.96M phrases remain.

While some phrases with non-ASCII characters are also commonly used in English (e.g., "déjà vu"), we find only 2.48% of phrases at this stage contain non-ASCII characters, and 29% of them are common in English. In short, we are removing only 0.72% of the English phrases that contain non-ASCII characters in Step 3.

Step 4: Find phrases of polysemous words To increase the chance of collecting polysemous mNPs, we only keep mNPs that have at least one word in the list of 2,345 unique multiple-sense words of WiC (Huggingface, 2022a), arriving at ∼6.5M mNPs, each appearing in ≥ 2 sentences and in ≥ 1 Wikipedia pages. We empirically find that Step 4 is important and substantially increases our chance of finding polysemous mNPs (compared to skipping Step 4).

Step 5: Find phrases in distinct contexts We observe that a mNP is likely to be polysemous when (a) its context sentences are semantically different; and (b) its context Wikipedia pages are of dissimilar categories (e.g. "massive figure" in finance vs. history; Fig. A8).

To implement this filter, we form all possible triplets (phrase, sentence$_1$, sentence$_2$) from the list of context sentences of each mNP[3]. We compute the cosine similarity of two sentences at the CLS embedding space of a SimCSE (Gao et al., 2021) provided on HuggingFace (Group, 2022). To find triplets where the two sentences are semantically dissimilar, we keep only the triplets where (sentence$_1$, sentence$_2$) has a low cosine similarity, i.e. $\in [-0.3, 0.2]$ and the length difference of the two sentences is < 4 words (as two sentences of substantially different lengths often have a low cosine similarity regardless of their semantic differences). As the result, there are ∼600K triplets remaining after this step.

We further re-rank these ∼600K descendingly by the dissimilarity of the lists of Wikipedia categories[4] of the context pages that contain sentence$_1$ and sentence$_2$. That is, we treat each Wikipedia page's comma-separated list of categories as an input text to SimCSE and sort the ∼600K descendingly by the cosine similarity of the resultant embeddings.

Step 6: Select data for annotation Before asking annotators to label our sorted phrases we perform final filtering by removing proper nouns and phrases whose Wikipedia documents contain missing words.

We perform final filtering to ensure the data given to annotators is in a proper format. That is, from ∼600K phrases, we filter down to ∼475K phrases by applying two filters: (1) Remove all phrases that are proper nouns (i.e. POS tagging returns PROPN) since proper nouns often refer to a single identity and thus unambiguous; (2) Remove all phrases that have a newline character and all phrases whose context Wikipedia page contains missing words (i.e. errors in the Wikipedia dump).

As the result, we obtain a list of ∼475K phrases sorted by their estimate chance of carrying two different senses. After manual inspection, we take the top 19,500 triplets of the format (phrase, page$_1$, page$_2$)—i.e. a phrase $\boldsymbol{p}$ and its two context Wikipedia pages where $\boldsymbol{p}$ is the most likely to have two different senses (e.g., see "massive figure" in Fig. 1a)—and hire linguistic experts to annotate them.

Our manual inspection involves taking 1,000 random triplets and manually reading them. We find that at least ∼30% of the 1,000-triplet subset contain a polysemous target phrase $\boldsymbol{p}$ and two Wikipedia pages that give $\boldsymbol{p}$ two unique meanings. We perform this manual inspection repeatedly throughout the

---

[3] For computational tractability, we only keep at most 32 context sentences per mNP where each sentence's length in words is $\in [5, 25]$.

[4] We use the provided Wikipedia API (Team, 2021a) to obtain the categories for each article as the dump file has no category-related information.

process of inventing and refining the data collection process in order to arrive at the final list of steps as presented in this paper.

C.1 Biases in the data collection

While there are many filtering steps in our data collection above, most of them are data cleaning filters that are typically needed in a regular NLP dataset construction.

We recognize that there are *three key filters* in our system that impose strong biases:

1. In Step 4, we use only phrases that contain one word in the WiC. That is, we find Step 4 to substantially increase our chance of finding triplets with a polysemous target phrase. We have added this note in the Data Collection description. It is possible to remove Step 4, but that would require a larger human annotation effort to reach the same 15K labeled triplets.
2. In Step 5, we rely on SimCSE to find target phrases that are placed in two sentences of dissimilar meanings.
3. In Step 5, we rely on SimCSE to find target phrases that are placed in two Wikipedia pages of distinct topics.

D Statistics for search queries in Yahoo Search Query dataset

We analyze 4,496 user queries released in the Yahoo Search Query Log To Entities dataset (Yahoo, 2022b) and use SpaCy tokenizer (Honnibal et al., 2020) to classify them into 4 main categories: Noun phrases, verb phrases, URLs and others. As a result, noun phrases are the most common query type from users with 3,576 queries ($\sim$79.54%) followed by URLs with 675 queries ($\sim$15.01%) while verb phrases and other types are less preferred by users. Moreover, the average length of the real user queries is $\sim$1.60 which is quite close to our PS task with $\sim$2.27.

Table A2: Statistics of Yahoo queries across different query types.

Query type	# queries	Percentage (%)
Noun phrases	3,576	79.54
Verb phrases	148	3.29
URLs	675	15.01
Others	97	2.16
Total	4,496	100.00

E Verification of Phrase Similarity

To enhance the quality of the proposed PiC benchmark, we hire three additional Upwork experts to verify the correctness of PS examples where two phrases are supposed to be *non-equivalent* for negative examples (e.g. massive figure and giant number in Fig. 1b) or *equivalent* for positive examples (e.g. massive figure and huge model in Fig. 1b), and keep an example if it is endorsed by at least two experts (the rest is discarded from PS).

Two Upwork verifiers A_1 and A_2 start checking 5,104 *negative* examples and the third verifier A_3 is responsible for breaking the ties if A_1 and A_2 disagree with each other (see Fig. A1). Both A_1 and A_2 are asked to provide corrections when they do not agree with the labels. As a result, 4,935 out of 5,104 examples are accepted by pairs of (A_1, A_2), (A_1, A_3) or (A_2, A_3), 68 examples incorrect at first but are modified by either A_1 or A_2 and endorsed by A_3. In total, we reject 101 negative examples because there are not at least two experts agreeing with the annotations.

We repeat the same procedure to verify 5,104 *positive* examples. In sum, we retain 5,002 examples including 4,904 examples accepted by pairs of two verifiers and 98 examples incorrect at first but are modified by either A_1 or A_2 and endorsed by A_3. There are 102 positive examples rejected because there are not at least two experts agreeing with the annotations (Fig. A2).

After this verification round, we collect 5,003 *negative* examples and 5,002 *positive* examples. and randomly exclude 1 negative example to make the dataset balance which results in 10,004 examples in total for PS.

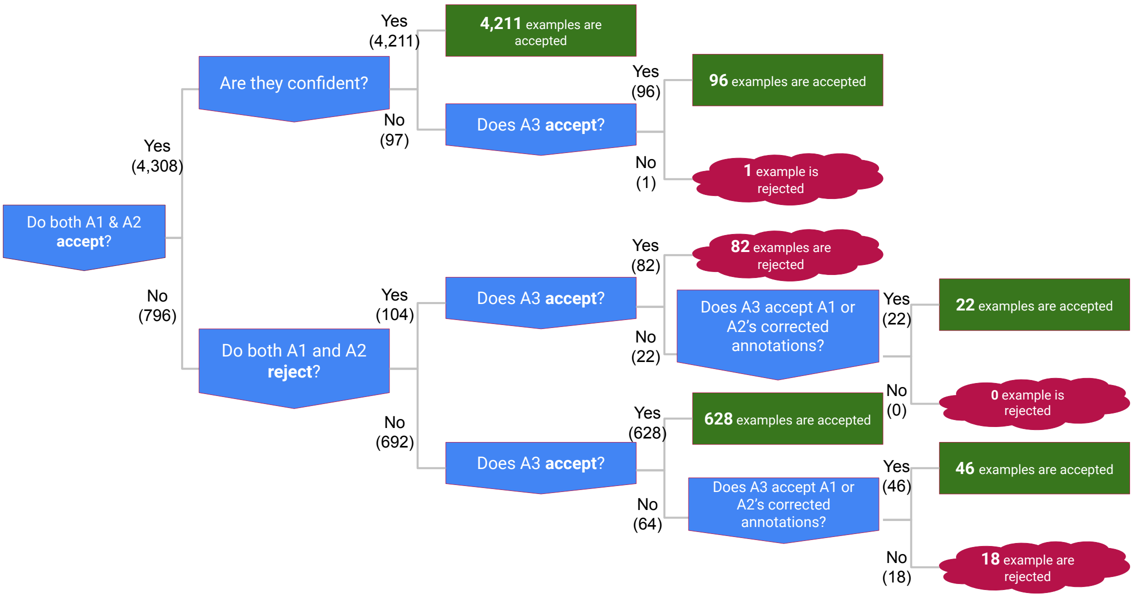


Figure A1: A decision tree describing our verification process for PS that involves three experts. Red, green and blue cells represent Reject, Accept decision and Questions. The numbers of examples for each branch are shown in parentheses.

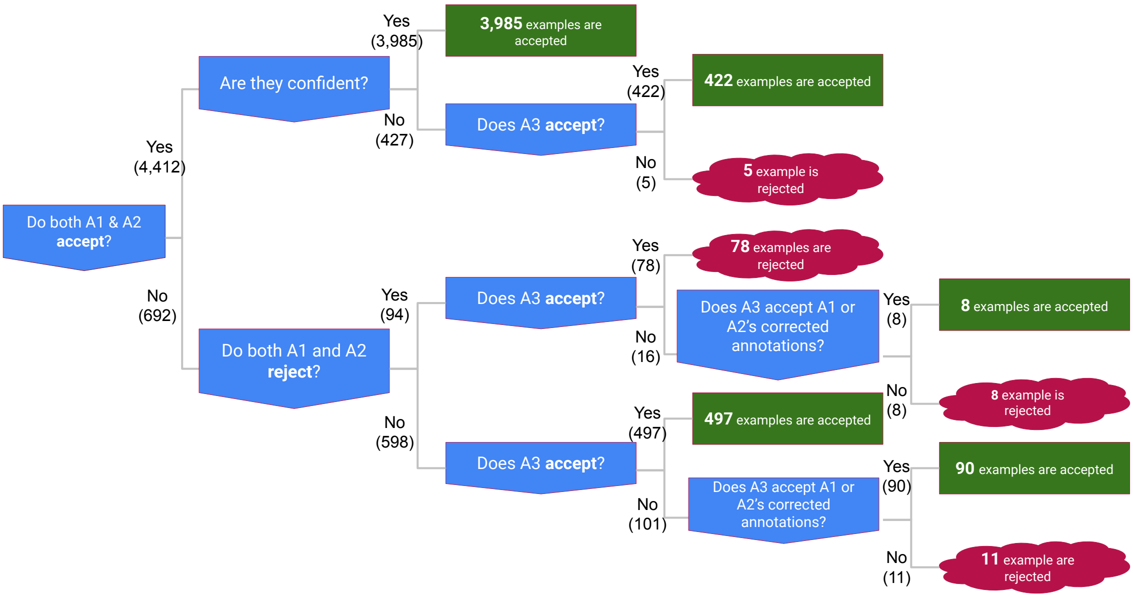


Figure A2: A decision tree describing our verification process for PS that involves three experts. Red, green and blue cells represent Reject, Accept decision and Questions. The numbers of examples for each branch are shown in parentheses.

Table A1: Summary of our 3-stage data construction. $\boldsymbol{p}$, s, m d, q, l denote target phrase, sentence, metadata, document, query, and label, respectively.

	Remaining #	Data type	Description
Sec. 3.1 Data Collection			
Step 1: Download Wiki articles	$\sim$6.27M	articles	Remove $\sim$9.51M empty articles.
Step 2: Extract phrases	$\sim$286.78M	($\boldsymbol{p}$, s, m)	Extract noun phrases and proper nouns along with their context sentences from Wikipedia articles.
Step 3: Remove phrases of a single context	$\sim$17.96M	($\boldsymbol{p}$, $[s_1, ..., s_n]$, m)	For each phrase, gather all sentences where that phrase is used.
Step 4: Find phrases of polysemous words	$\sim$6.5M	($\boldsymbol{p}$, $[s_1, ..., s_n]$, m)	Filter those phrases that do not contain WiC words.
Step 5: Find phrases in distinct contexts			Sort by X_i and apply filters to find pairs of sentences where their phrase potentially has different meanings.
- Sort and filter by semantic dissimilarity	$\sim$600K	($\boldsymbol{p}$, s_1, s_2, m)	X_1 : cosine similarity scores of sentences embeddings.
- Sort by domain dissimilarity	$\sim$600K	($\boldsymbol{p}$, s_1, s_2, m)	X_2 : cosine similarity scores of domain embeddings i.e., use categories of each article to get embeddings.
Step 6: Select data for expert annotation	19,500	($\boldsymbol{p}$, d_1, d_2)	Remove proper nouns and phrases with missing information and select top 19,500 examples for annotation.
Sec. 3.2 Data Annotations	30,042	($\boldsymbol{p}$, d, q)	Create a query i.e., paraphrase from the given phrase in each context document.
	15,021	($\boldsymbol{p}$, d_1, d_2, l)	Create a Yes/No label for each pair of documents.
Sec. 3.3 Verifying Annotations			
Round 1: MTurk verifier	22,496	($\boldsymbol{p}$, d, q)	Verify queries and Yes/No label by MTurkers.
	10,043	($\boldsymbol{p}$, d_1, d_2, l)	
Round 2: Upwork verifiers	28,325	($\boldsymbol{p}$, d, q)	Verify instances rejected in Round 1.
	13,413	($\boldsymbol{p}$, d_1, d_2, l)	

F Quantitative results on PR-page

Table A3: **Ranking** accuracy (%) on **PR-page** using the state-of-the-art pretrained phrase embeddings (a) and those finetuned on PR-pass via SS-style training (b).

Model	Phrase				Phrase + Context			
	Top-1	Top-3	Top-5	MRR@5	Top-1	Top-3	Top-5	MRR@5
(a) Pre-trained embeddings								
BERT (Devlin et al., 2019)	20.70	34.30	41.00	28.20	**35.40** (+14.70)	**52.10**	**59.10**	**44.50**
USE-v5 (Cer et al., 2018)	**32.20**	**52.70**	**60.80**	**43.20**	n/a	n/a	n/a	n/a
(b) PR-pass-trained SS models' phrase embeddings								
PhraseBERT (Wang et al., 2021)	**49.40**	**69.40**	**76.70**	**60.10**	14.70	21.60	26.10	18.70
SimCSE (Gao et al., 2021)	44.20	66.60	73.50	55.70	24.60	37.80	43.20	31.70

G Finetuning on PSD does not substantially improve accuracy

As PSD has only 4,858 examples, we use all examples for testing in Sec. 5.5 and find the best PR-trained SS models to perform poorly. To further understand the challenge of PSD, here, we ask:
Q: *How much does training on PR-pass and finetuning on PSD improve accuracy on PSD?*

Experiment We take the PR-pass-trained SS models and further finetune them on a subset of PSD to measure how training directly on PSD improves SS models. We split PSD into 1,438/500/3,000 examples for train/dev/test sets, respectively, and finetune the PR-pass-trained SS models on this PSD train set. For comparison with the results in Sec. 5.4, we use the same set of hyperparameters as when finetuning on PR-pass in Sec. 5.4. Below, we report the test-set results of the lowest dev-loss models.
Results On the PSD-3K test set, all models perform poorly at a mean EM score of 55.14% (Table A4a; mean). Interestingly, finetuning the original models using the 1,938 examples (hereafter, PSD-2K) instead of PR-pass decreases accuracy, on average by -6.51 points. An explanation is that 1,438 PSD training examples are too few for the finetuning to be effective. Indeed, finetuning the PR-pass-trained SS models further on PSD-2K increases the scores for all models by +9.10 on average (Table A4c; mean). The best model is Longformer$_{\text{Base}}$ (Beltagy et al., 2020) (Table A4; 71.10 EM), which is still substantially lower than the human upperbound of 95%.

Table A4: Performance of **SS models** on 3,000 PSD **test** examples. (a) and (b) models are **finetuned** only on PR-pass and 1,938 PSD examples (PSD-2K), respectively. (c) models are finetuned on PR-pass first and then finetuned on PSD-2K. All models are "base" unless otherwise specified. The definitions of EM+loc and F$_1$+loc are in Table 6's caption.

Models finetuned on	(a) PR-pass		(b) PSD-2K		(c) PR-pass + PSD-2K	
	EM+loc	F$_1$+loc	EM+loc	F$_1$+loc	EM+loc	F$_1$+loc
PhraseBERT (Wang et al., 2021)	51.00	51.15	35.43 (-15.57)	36.02	56.53 (+5.53)	56.81
BERT (Devlin et al., 2019)	54.53	54.75	44.33 (-10.20)	45.28	63.83 (+9.30)	64.14
BERT$_{\text{Large}}$ (Devlin et al., 2019)	54.77	54.99	54.07 (-0.70)	54.82	67.13 (+12.36)	67.36
SpanBERT (Joshi et al., 2020)	52.27	52.37	44.67 (-7.60)	45.35	69.93 (+17.66)	70.14
SentenceBERT (Reimers and Gurevych, 2019)	52.27	52.41	38.63 (-13.64)	39.31	58.93 (+6.66)	59.21
SimCSE (Gao et al., 2021)	53.47	53.59	43.67 (-9.80)	44.38	60.60 (+7.13)	60.80
Longformer (Beltagy et al., 2020)	**62.47**	**62.58**	61.97 (-0.50)	62.69	**71.10** (+8.63)	**71.30**
Longformer$_{\text{Large}}$ (Beltagy et al., 2020)	60.33	60.42	**66.27** (+5.94)	**67.10**	65.87 (+5.54)	66.10
mean ± std	55.14 4.10	55.28 4.08	48.63 (-6.51) 11.03	49.37 11.08	64.24 (+9.10) 5.23	64.48 4.13

H SS-style training improves *non*-contextualized but not contextualized phrase embeddings

As the SS models trained on PR-pass and PR-page perform impressively (Sec. 5.4), almost 1.5× better than the ranking models that are based on pre-trained embeddings, an interesting question is:
Q: *Does SS training also improve contextualized phrase embeddings?*

This is important to understand because the impressive SS-models' performance gain may come from the extra linear-classification layer (not necessarily from the finetuned embeddings).

Experiment We extract the phrase embeddings (both non-contextualized and contextualized) from the PR-pass-trained SS models from Sec. 5.4 (i.e. discarding the classification layer) and test them in the PR-pass ranking experiments (as in Sec. 5.3).

Results After finetuning on PR-pass, the *non*-contextualized phrase embeddings improve substantially for most models at an average gain of +16.61 in top-1 accuracy (e.g., PhraseBERT top-1 accuracy increases from 36.62% to 59.02%; Table A5b). This result shows that training on PR-pass improves non-contextualized phrase embeddings. In stark contrast, the ranking scores of *contextualized* phrase embeddings drop significantly, -11.95 points on average (Table A5c), compared to before finetuning on PR-pass.

In sum, we are observing a consistent trend that the contextualized phrase embeddings of the original pre-trained BERT (both "base" and "large") are remarkably beneficial for retrieval (i.e. PR). However, after finetuning, e.g. on PR-pass or using other techniques (e.g. in PhraseBERT or SentenceBERT), such benefits of leveraging context disappear. Aligned with Yu and Ettinger (2020), we find that incorporating context effectively into phrase embeddings is an open research challenge.

Table A5: **Ranking** accuracy (%) on **PR-pass** using the state-of-the-art pretrained phrase embeddings (a) and those finetuned on PR-pass via SS-style training (b). See Appendix F for the results on PR-page. Δ (e.g. -3.62) denotes the differences between the Top-1 accuracy in the contextualized ("Phrase + Context") vs. the non-contextualized ("Phrase") setting.

Model	Phrase				Phrase + Context			
	Top-1	Top-3	Top-5	MRR@5	Top-1 (Δ)	Top-3	Top-5	MRR@5
(a) Pre-trained embeddings								
PhraseBERT (Wang et al., 2021)	36.62	66.96	75.90	52.20	33.00 (-3.62)	49.60	56.70	41.90
BERT (Devlin et al., 2019)	29.80	47.90	55.40	39.50	**47.44** (+17.64)	**65.78**	**73.30**	**57.30**
$\text{BERT}_{\text{Large}}$ (Devlin et al., 2019)	23.76	38.52	45.40	31.70	**42.80** (+19.04)	**58.90**	**64.90**	**51.30**
SpanBERT (Joshi et al., 2020)	20.88	31.04	35.20	26.40	14.40 (-6.48)	30.46	39.80	23.40
SentenceBERT (Reimers and Gurevych, 2019)	22.30	50.64	60.60	36.80	**25.14** (+2.84)	39.52	46.20	32.90
SimCSE (Gao et al., 2021)	28.10	53.70	64.60	41.60	**32.40** (+4.30)	53.44	62.80	**43.70**
USE-v5 (Cer et al., 2018)	**43.36**	**70.12**	**78.90**	**57.30**	n/a	n/a	n/a	n/a
DensePhrase (Lee et al., 2021)	32.24	51.30	60.50	42.60	31.50 (-0.74)	46.30	53.80	39.70
(b) PR-pass-trained SS models' phrase embeddings								
PhraseBERT (Wang et al., 2021)	**59.02**	**81.58**	**87.90**	**70.60**	24.98 (-34.04)	37.78	43.90	32.00
BERT (Devlin et al., 2019)	50.10	66.16	71.40	58.60	20.34 (-29.76)	31.40	37.10	26.50
$\text{BERT}_{\text{Large}}$ (Devlin et al., 2019)	32.70	42.40	45.90	37.80	11.40 (-21.30)	17.00	20.50	14.60
SpanBERT (Joshi et al., 2020)	15.22	22.88	26.60	19.40	8.92 (-6.30)	13.56	16.60	11.60
SentenceBERT (Reimers and Gurevych, 2019)	53.14	74.86	80.70	64.20	20.12 (-33.02)	30.04	34.90	25.60
SimCSE (Gao et al., 2021)	50.96	76.70	83.40	64.00	37.70 (-13.26)	52.38	58.90	45.60
(c) Differences between after vs. before finetuning, i.e. the 6 models in (b) vs. those in (a)								
mean differences	+16.61				-11.95			

I Qualitative examples for PS, PR-pass, PR-page and PSD

PS example. Groundtruth: "positive"	
P_1	moderate speed
P_2	steady pace
S_1	Deforestation due to logging and land conversion has likely caused the population to decline at a moderate speed.
S_2	Deforestation due to logging and land conversion has likely caused the population to decline at a steady pace.

Figure A3: PhraseBERT-based classifier **correctly** predicts "positive" given two phrases P_1 and P_2 with and without the presence of context S_1 and S_2. Here, to humans, the phrases are non-polysemous and have the same meaning.

PS example. Groundtruth: "negative"	
P_1	greatest emphasis
P_2	highest stress
S_1	However, the rock art had the greatest emphasis on domesticated cattle.
S_2	However, the rock art had the highest stress on domesticated cattle.

Figure A4: PhraseBERT-based classifier **correctly** predicts "negative" given two phrases P_1 and P_2 with and without the presence of context S_1 and S_2. Here, to humans, the two phrases are non-ambiguously carrying different meanings.

PS example. Groundtruth: "positive"	
P_1	unique image
P_2	uncommon style
S_1	Bayliss has been praised for her unique image and tendency to change up songs.
S_2	Bayliss has been praised for her uncommon style and tendency to change up songs.

Figure A5: PS case that requires context to determine similarity. Without context, a PhraseBERT-based classifier incorrectly thinks P_1 and P_2 are different. Yet, it changes the prediction to "positive", i.e. thinking two phrases have the same meaning, when the context is taken into account.

PS example. Groundtruth: "negative"	
P_1	permanent post
P_2	stable location
S_1	His assistant, John Carver took over as caretaker manager, managing one win, but was not considered for the permanent post, and left in September 2004.
S_2	His assistant, John Carver took over as caretaker manager, managing one win, but was not considered for the stable location, and left in September 2004.

Figure A6: PS case that requires context to determine similarity. Without context, PhraseBERT-based classifier incorrectly thinks P_1 and P_2 carry the same meaning. Yet, it correctly changes the prediction to "negative" when the context is taken into account.

PSD example.	
$\boldsymbol{d}$	Bubble memory is a type of non-volatile computer memory that uses a thin film of a magnetic material to hold small magnetized areas, known as "bubbles" or "domains", each storing one bit of data. The material is arranged to form a series of parallel tracks that the bubbles can move along under the action of an external magnetic field. The bubbles are read by moving them to the edge of the material where they can be read by a conventional magnetic pickup, and then rewritten on the far edge to keep the memory cycling through the material. In operation, bubble memories are similar to delay line memory systems. Bubble memory started out as a promising technology in the 1970s, offering memory density of an order similar to hard drives but performance more comparable to core memory while lacking any moving parts. This led many to consider it a contender for a "universal memory" that could be used for all storage needs. The introduction of dramatically faster semiconductor memory chips pushed bubble into the slow end of the scale, and equally dramatic improvements in hard drive capacity made it uncompetitive in price terms. Bubble memory was used for some time in the 1970s and 80s where its non-moving nature was desirable for maintenance or shock-proofing reasons. The introduction of Flash RAM and similar technologies rendered even this niche uncompetitive, and bubble disappeared entirely by the late 1980s. History. Precursors. The Inkerman stone, of which the building is made, was mined near Sevastopol and transported by barges. No convenient mooring facilities existed at that time, so the barges had to anchor in the harbor and the load was moved to the shore by boats and then transported to the construction site across the steppe. During the first year of construction, the builders concentrated on the basic structure at the expense of various facilities and decorations. At the end of 1816, the lighthouse looked like a conic 36-metre-high stone tower with a wooden 3.3-metre-high decagonal lantern. The lighthouse became operational in 1817 after its lighting system had been repaired. Three houses were built next to the tower to accommodate the lighthouse personnel and for storage needs. However, cold and humid winters of the Tarkhanut Peninsula, however, made these houses nearly unsuitable for living. In 1862, the lighting system was upgraded, and the spread of light reached 12.4 miles. In 1873, the construction resumed along with cleaning efforts of the surrounding areas. The building was finished and painted white. In 1876, an additional telegraph spot was built near the tower.
$\boldsymbol{q}_1$	storehouse purposes Groundtruth: storage needs & Prediction: storage needs (confidence: 0.99)
$\boldsymbol{q}_2$	data caching Groundtruth: storage needs & Prediction: storage needs (confidence: 0.99)

Figure A7: Given document $\boldsymbol{d}$, our Longformer$_{\text{Large}}$ SS model trained on PR-pass correctly retrieves storage needs in the second paragraph for the query $\boldsymbol{q}_1$ “storehouse purposes” but *fails* to retrieve the answer when the query $\boldsymbol{q}_2$ is “data caching”. The predicted answer for $\boldsymbol{q}_2$ should be storage needs (i.e. in the first passage) since this phrase relates to caching data digitally in computers while storage needs refers to physically storing objects.

PSD example.	
$\boldsymbol{d}$	In the libretto, Delilah is portrayed as a seductive "femme fatale", but the music played during her parts invokes sympathy for her. The 1949 biblical drama "Samson and Delilah", directed by Cecil B. DeMille and starring Victor Mature and Hedy Lamarr in the titular roles, was widely praised by critics for its cinematography, lead performances, costumes, sets, and innovative special effects. It became the highest-grossing film of 1950, and was nominated for five Academy Awards, winning two. According to "Variety", the film portrays Samson as a stereotypical "handsome but dumb hulk of muscle". Samson has been especially honored in Russian artwork because the Russians defeated the Swedes in the Battle of Poltava on the feast day of St. Sampson, whose name is homophonous with Samson's. The lion slain by Samson was interpreted to represent Sweden, as a result of the lion's placement on the Swedish coat of arms. In 1735, C. B. Rastrelli's bronze statue of Samson slaying the lion was placed in the center of the great cascade of the fountain at Peterhof Palace in Saint Petersburg. Samson is the emblem of Lungau, Salzburg and parades in his honor are held annually in ten villages of the Lungau and two villages in the north-west Styria (Austria). During the parade, a young bachelor from the community carries a massive figure made of wood or aluminum said to represent Samson. The tradition, which was first documented in 1635, was entered into the UNESCO list of Intangible Cultural Heritage in Austria in 2010. Samson is one of the giant figures at the "Ducasse" festivities, which take place at Ath, Belgium. On September 22, 2015, Honda announced that they had sold over 1 million Activas in five months in the Indian market, from April to August. Honda launched their 5th generation of Honda Activa in 2018, and the sixth-generation Honda Activa 6G have been launched in India with prices starting at 63,912 (ex-showroom, Delhi). Milestones. In April, 2014, "The Economic Times" reported the Honda Activa to be the best selling two wheeler in India, outselling the Hero Splendor. During the month of September 2013, 141,996 Honda Activa scooters were sold, nearly equal to Honda's entire annual sales in North America. The 110cc Activa is the company's biggest seller, by far. It is responsible for over 2,00,000 sales units each month. In November 2018, HMSI crossed the 2.5 crore sales mark in the scooter segment. It has become the first company to reach this milestone and the biggest contributor to this massive figure is the Honda Activa. It took Honda 13 years to achieve the one crore sales figure, but it managed to add another crore in the span of just three years. It then went on to achieve the next 50 lakh in just one year.
$\boldsymbol{q}_1$	huge model Groundtruth: massive figure & Prediction: massive figure (confidence: 0.99)
$\boldsymbol{q}_2$	giant number Groundtruth: massive figure & Prediction: massive figure (confidence: 0.99)

Figure A8: Given document $\boldsymbol{d}$, Longformer$_{\text{Large}}$ model trained with SS approach on PR-pass correctly retrieves massive figure in the second paragraph for the query $\boldsymbol{q}_2$ “giant number” but *fails* to retrieve the answer when the query $\boldsymbol{q}_1$ is “huge model”. The predicted answer for $\boldsymbol{q}_1$ should be massive figure in the first passage since this phrase relates to a physical shape instead of a number.

	PSD example.
$\boldsymbol{d}$	Eva held ambitions to replace Hortensio Quijano for the 1951 election, although her poor health kept her from this. Nonetheless many were concerned that her agenda would be pushed through. In march of 1951 the government arrested several retired army officers due to their dissent and disapproval of Perón's administration. This raised tensions among the rest of the army, although action did not occur. By September tensions had risen among the military due to the unrivalled power of the Peronist regime. On September 28, 1951, during the election, Menéndez led the military uprising in an attempt to overthrow the government. He led a core of officers, commanding a division, and left Campo de Mayo bound for the Casa Rosada. Resolve for the uprising, especially among the non-commissioned officers and enlisted men, was not strong enough. They were not prepared to fight their own countrymen. The uprising was over as soon as opposition was encountered, almost completely bloodless. Perón admired the loyalty of the troops and pardoned all those involved. The design uses a similar standard to the JVX in terms of distortion reduction with crossbraces and 27 cells but that's where the similarity ends. Petra was built from the ground up with entirely new panel shaping and trim. Petra has a highly elliptical planform and very high sweep. NZ Aerosports say she has a high roll rate, a long recovery arc and high maximal glide ratio. She is said to deliver unrivalled power in the turn, plane out and flare. Petra has a long list of World Records, National and International titles to back that up. She had an impressive debut at the PD Big Boy Pants event in July 2011, with Nick Batsch setting a new distance world record of 222.45m (729ft). One month later Nick took out the Pink Open in Klatovy and the FAI World Cup also; first in distance, speed and overall. He also won the 2011 US CP nationals on Petra. Patrick Boulongne came 2nd in the European Championships and 6th overall at the World Cup with Petra in his first competition with her. He went on to win the 2011 French Canopy Piloting Nationals.
$\boldsymbol{q}_1$	incomparable energy Groundtruth: unrivalled power & Prediction: unrivalled power (confidence: 0.99)
$\boldsymbol{q}_2$	indomitable strength Groundtruth: unrivalled power & Prediction: unrivalled power (confidence: 0.99)

Figure A9: Given document $\boldsymbol{d}$, Longformer$_{\text{Large}}$ model trained via the SS approach on PR-pass correctly retrieves unrivalled power in the first paragraph for the query $\boldsymbol{q}_2$ "indomitable strength" but *fails* to retrieve the answer when the query $\boldsymbol{q}_1$ is "incomparable energy". The predicted answer for $\boldsymbol{q}_1$ should be unrivalled power in the second passage since the second passage changes "unrivalled power" meaning to a competition strength instead of military power.

	PR-pass example. Groundtruth: common thought
$\boldsymbol{d}$	As the medical corps grew in size there was also specialization evolving. Physicians surfaced that specialized in disease, surgery, wound dressing and even veterinary medicine. Veterinary physicians were there to tend to livestock for agricultural purposes as well as combat purposes. The Cavalry was known for their use of horses in combat and scouting purposes. Because of the type of injuries that would have been commonly seen, surgery was a somewhat common occurrence. Tools such as scissors, knives and arrow extractors have been found in remains. In fact, Roman surgery was quite intuitive, in contrast to common thought of ancient surgery. The Roman military surgeons used a cocktail of plants, which created a sedative similar to modern anesthesia. Written documentation also showed surgeons would use oxidation from a metal such as copper and scrape it into wounds, which provided an antibacterial effect; however, this method was most likely more toxic than providing an actual benefit. Doctors had the knowledge to clean their surgical instruments with hot water after each use. Wounds were dressed, and dead tissue was removed when bandages were changed.
$\boldsymbol{q}$	prevalent theory
R	0.882 common thought 0.855 common thought of 0.702 fact 0.698 to common thought 0.675 common occurrence

Figure A10: A **ranking** model based on the phrase embeddings of the PR-pass-trained PhraseBERT SS model correctly ranks and retrieves the most semantically relevant answer "common thought" as the top-1 prediction in the retrieval list R for the query "prevalent theory" in a PR-pass example (which contains a document $\boldsymbol{d}$ and a query $\boldsymbol{q}$).

PR-page example.	Groundtruth: continued risk
d	... Following a United Nations agreement between Indonesia and Portugal, a UN-supervised referendum held on 30 August 1999 offered a choice between autonomy within Indonesia and full independence. The people of East Timor voted overwhelmingly for independence. An Australian-led and Indonesian-sanctioned peacekeeping force, INTERFET, was sent into the territory to restore order following a violent 'scorched-earth' policy carried out by pro-integration militia and supported by elements of the Indonesian military. In response to Australia's involvement, Indonesia abrogated the 1995 security pact, asserting that Australia's actions in East Timor were inconsistent with 'both the letter and spirit of the agreement'. Official meetings were cancelled or delayed, including the Indonesia-Australia Ministerial Dialogue, which would not reconvene until March 2003. INTERFET was later replaced by a UN force of international police, UNTAET, which formed a detachment to investigate alleged atrocities. "Tampa" affair and the War on Terror. The relationship came under strain in August 2001 during the "Tampa" affair, when Australia refused permission for the Norwegian freighter ship MV "Tampa" to enter Australian waters while carrying Afghan asylum seekers that it had rescued from a distressed fishing vessel in international waters. The Indonesian Search and Rescue Agency did not immediately respond to requests from Australia to receive the vessel. When the ship entered Australian territorial waters after being refused permission, Australia attempted without success to persuade Indonesia to accept the asylum seekers. Norway also refused to accept the asylum seekers and reported Australia to international maritime authorities. The incident prompted closer coordination between Indonesian and Australian authorities, including regional conferences on people smuggling, trafficking in persons and other transnational crime. In 2002, a terrorist attack in Kuta, Bali killed 202 people, including 88 Australians, and injured a further 240. Jemaah Islamiyah, a violent Islamist group, claimed responsibility for the attack, allegedly in retaliation for Australia's support for East Timorese independence and the War on Terror. A subsequent attack in 2005 resulted in the deaths of a further 20 people, including 15 Indonesians and 4 Australians. The 2003 Marriott Hotel bombing was also perceived as targeted at Western interests in Indonesia; Al Qaeda claimed the attack was carried out by a Jemaah Islamiyah suicide bomber in response to actions of the United States and its allies, including Australia. A 2004 attack on the Australian embassy in Jakarta by Jemaah Islamiyah resulted in the deaths of nine Indonesians. The following year, Indonesian diplomatic and consular premises in Australia received a number of hoax and threat messages. Since then, both the United States and Australian governments have issued warnings against travel to Indonesia, advising their citizens of a continued risk of attacks. These incidents prompted greater cooperation between law enforcement agencies in the two countries, building on a 1999 agreement on drug trafficking and money laundering. The Australian Federal Police's Jakarta Regional Cooperation Team provided assistance to the Indonesian National Police, and has contributed to the Jakarta Centre for Law Enforcement Cooperation. This relationship has attracted criticism, particularly following the arrest and sentencing of the Bali Nine, a group of nine Australians arrested in Denpasar while attempting to smuggle heroin from Indonesia to Australia. The 2005 conviction of Schapelle Corby for attempting to smuggle drugs to Bali also attracted significant attention in the Australian media. The 2004 Indian Ocean earthquake prompted a significant humanitarian response from Australia, including a $1 billion aid package from the federal government, a further $17.45 million contribution from state and territory governments, and the commitment of 900 Australian Defence Force personnel to relief efforts in northern Sumatra and Aceh. A telethon broadcast on Australia's three major commercial television networks called "" generated pledges of more than $10 million, contributing to total private aid of $140 million. The Eighth "Australia-Indonesia Ministerial Forum" (AIMF) was held in Bali on 29 June 2006 and was attended by five Australian and eleven Indonesian ministers. A key outcome was support for the conclusion of a security agreement, later realised as the Lombok Agreement, providing a framework for the development of the security relationship by the end of 2006 on defence, law enforcement, counter-terrorism, intelligence, maritime security, aviation safety, WMD non-proliferation, and bilateral nuclear cooperation for peaceful purposes. Australia-Indonesia-East Timor Trilateral Ministerial Meetings occurred three times to September 2006. Recent relations. 2010 President Susilo Bambang Yudhoyono visited Australia in April 2010, and became the second Indonesian leader to address federal parliament: Finally, I look forward to a day in the near future. The day when policy makers, academicians, journalists and other opinion leaders all over the world take a good look at the things we are doing so well together. And they will say: these two used to be worlds apart. But they now have a fair dinkum of a partnership. ...
q	sustained threat
R	0.830 threat . 0.802 potential threat 0.800 threat reached 0.787 threat as 0.787 threat to

Figure A11: A **ranking** model based on the non-contextualized embeddings of USE-v5 fails to retrieve the correct answer "continued risk" for the query "sustained threat" in the PR-page example (which contains a document $\boldsymbol{d}$ and a query $\boldsymbol{q}$). The top-5 phrases retrieved (R) contains the word "threat" but have no identifier conveying the "continued" or 'sustained" sense. Here, the Wikipedia page is truncated to fit into a single manuscript page.

J Verifying annotations

There are two common methods for evaluation of dataset quality: (1) Verify only a small, random subset (Pilehvar and Camacho-Collados, 2019) to estimate the quality of the full dataset or (2) verifying the entire dataset with multiple annotators and use the inter-annotator agreement (IAA) to control quality (Bowman et al., 2015; Kwiatkowski et al., 2019). The first approach for approximation is budget-friendly but it remains unknown whether the rest of examples are at high quality, while IAA is more desired but annotating thousands of instances can be prohibitively slow and costly.

We propose a **hybrid approach** to evaluate (leveraging both linguistic experts and non-experts) and ensure high quality for 30,042 queries and 15,021 Yes/No answers at lower cost compared to IAA via two rounds:

1. First, we ask around 1,000 highly qualified freelancers on Amazon Mechanical Turk (MTurk verifiers) to verify whether the *query* annotated by our Upwork annotators is interchangeable i.e. has the same meaning with the given *phrase* in *paragraph*. To verify Yes/No answers, MTurk verifiers need to read two short paragraphs containing the same phrase like Upwork annotators to make decisions. We do not show answers to the MTurk verifiers to avoid biases.

2. Second, we continue hiring 5 Upwork verifiers who are writing experts to double-check those instances rejected by MTurk verifiers from the previous round and only discard an example if the Upwork verifiers agree with MTurk verifiers.

J.1 Round 1: Verification by MTurk non-experts

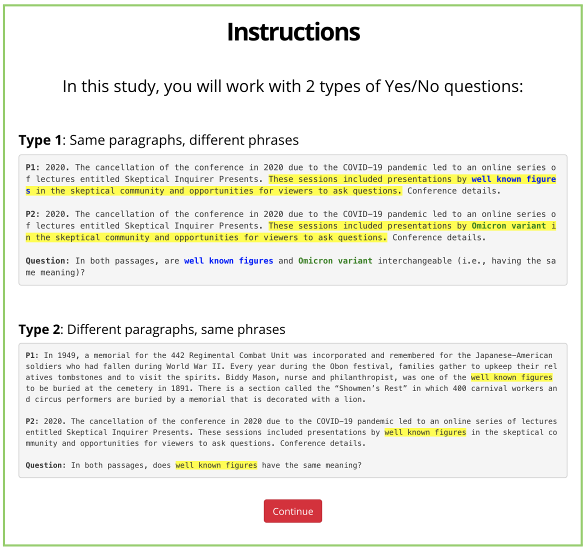
We use AMT platform to recruit more than 1,000 MTurk verifiers. Also, we use Gorrila (`gorilla.sc`) to develop user interface to collect answers from participants because (1) Gorilla provides easy-to-use tools to build graphical interface, (2) it is straightforward to monitor and discard results from unqualified participants and (3) we can easily share the experiment with MTurk verifiers via a link. Per 30 verified answers in around $\sim$20 minutes, the verification process costs us \$5.6 (AMT fees included) and 1 token to Gorilla to a single MTurk verifier.

Participants are given detailed instructions along with 5 practice samples to get familiar with the task (Fig. A12). They need to pass an evaluation checkpoint including 6 questions randomly sampled from our verified question bank in order to start working with sets of 30 questions. With this approach, all examples in the dataset are verified once and as a result, 22,496/30,042 queries ($\sim$74.88%) and 10,043/15,021 Yes/No answers ($\sim$66.86%) accepted by MTurkers are considered high quality since they are annotated by a writing expert and confirmed by a qualified English native speaker. The remaining 7,546 queries and 4,978 Yes/No answers rejected that are passed to another group of 5 writing experts for confirmation.

J.2 Round 2: Verification by Upwork experts

We hired another set of 5 writing experts from Upwork (Upwork verifiers) with an hourly rate of \$25-40/hour to verify 12,524 examples rejected by MTurk verifiers, i.e., at an average cost of approximately \$0.26 per example. See a sample assignment given to an Upwork expert in (PiC, 2022).

We rely on IAA to decide whether to accept or reject an example. Specifically, we use the *same question types* as shown to MTurk verifiers in the previous step and see whether these Upwork verifiers agree with the Upwork annotators to keep this example or with MTurk verifiers to reject it. We find that the agreement between the first- and third-round annotators are 5,829 (out of 7,546) paraphrases and 3,370 (out of 4,978) Yes/No answers in total and thus the total high-quality queries and Yes/No answers we achieve are 28,325 and 13,413, respectively.



Instructions

In this study, you will work with 2 types of Yes/No questions:

Type 1: Same paragraphs, different phrases

P1: 2020. The cancellation of the conference in 2020 due to the COVID–19 pandemic led to an online series of lectures entitled Skeptical Inquirer Presents. These sessions included presentations by **well known figures** in the skeptical community and opportunities for viewers to ask questions. Conference details.

P2: 2020. The cancellation of the conference in 2020 due to the COVID–19 pandemic led to an online series of lectures entitled Skeptical Inquirer Presents. These sessions included presentations by **Omicron variant** in the skeptical community and opportunities for viewers to ask questions. Conference details.

Question: In both passages, are **well known figures** and **Omicron variant** interchangeable (i.e., having the same meaning)?

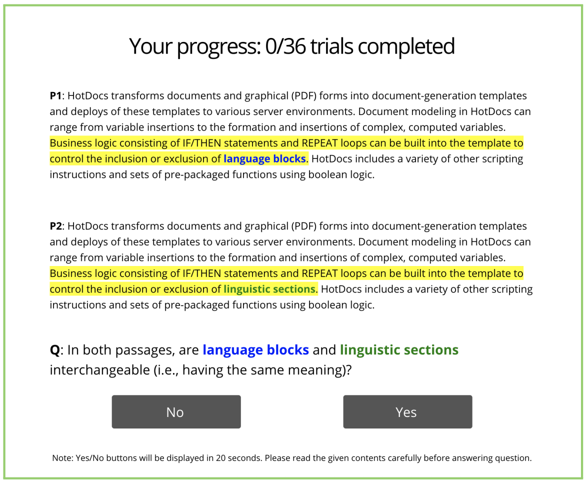
Type 2: Different paragraphs, same phrases

P1: In 1949, a memorial for the 442 Regimental Combat Unit was incorporated and remembered for the Japanese–American soldiers who had fallen during World War II. Every year during the Obon festival, families gather to upkeep their relatives tombstones and to visit the spirits. Biddy Mason, nurse and philanthropist, was one of the well known figures to be buried at the cemetery in 1891. There is a section called the “Showmen’s Rest” in which 400 carnival workers and circus performers are buried by a memorial that is decorated with a lion.

P2: 2020. The cancellation of the conference in 2020 due to the COVID–19 pandemic led to an online series of lectures entitled Skeptical Inquirer Presents. These sessions included presentations by well known figures in the skeptical community and opportunities for viewers to ask questions. Conference details.

Question: In both passages, does well known figures have the same meaning?

(a) Detailed instructions given to MTurkers



Your progress: 0/36 trials completed

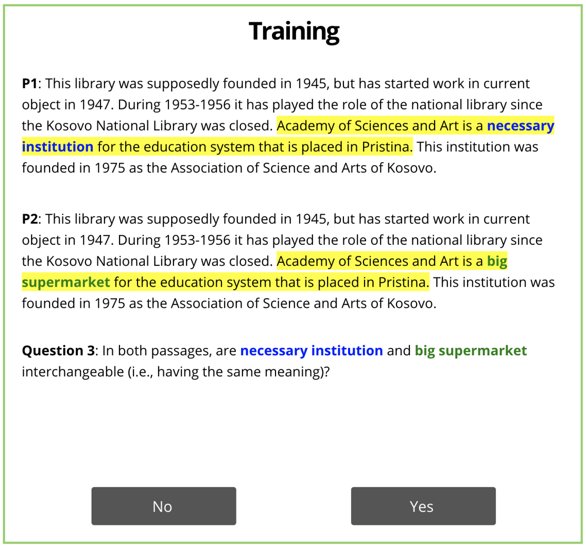
P1: HotDocs transforms documents and graphical (PDF) forms into document-generation templates and deploys of these templates to various server environments. Document modeling in HotDocs can range from variable insertions to the formation and insertions of complex, computed variables. Business logic consisting of IF/THEN statements and REPEAT loops can be built into the template to control the inclusion or exclusion of **language blocks**. HotDocs includes a variety of other scripting instructions and sets of pre-packaged functions using boolean logic.

P2: HotDocs transforms documents and graphical (PDF) forms into document-generation templates and deploys of these templates to various server environments. Document modeling in HotDocs can range from variable insertions to the formation and insertions of complex, computed variables. Business logic consisting of IF/THEN statements and REPEAT loops can be built into the template to control the inclusion or exclusion of **linguistic sections**. HotDocs includes a variety of other scripting instructions and sets of pre-packaged functions using boolean logic.

Q: In both passages, are **language blocks** and **linguistic sections** interchangeable (i.e., having the same meaning)?

Note: Yes/No buttons will be displayed in 20 seconds. Please read the given contents carefully before answering question.

(b) Upon completion of training stage, MTurkers need to correctly answer the first 5 out of 6 questions to be invited to verify annotations from Upwork experts.



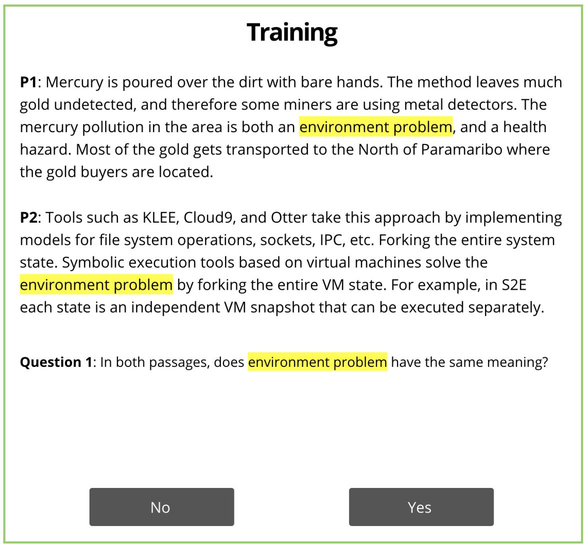
Training

P1: This library was supposedly founded in 1945, but has started work in current object in 1947. During 1953-1956 it has played the role of the national library since the Kosovo National Library was closed. Academy of Sciences and Art is a **necessary institution** for the education system that is placed in Pristina. This institution was founded in 1975 as the Association of Science and Arts of Kosovo.

P2: This library was supposedly founded in 1945, but has started work in current object in 1947. During 1953-1956 it has played the role of the national library since the Kosovo National Library was closed. Academy of Sciences and Art is a **big supermarket** for the education system that is placed in Pristina. This institution was founded in 1975 as the Association of Science and Arts of Kosovo.

Question 3: In both passages, are **necessary institution** and **big supermarket** interchangeable (i.e., having the same meaning)?

(c) Verification of paraphrases via type-1 question.



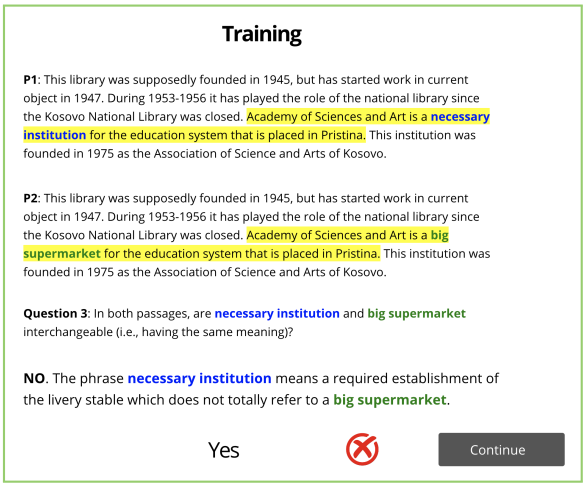
Training

P1: Mercury is poured over the dirt with bare hands. The method leaves much gold undetected, and therefore some miners are using metal detectors. The mercury pollution in the area is both an environment problem, and a health hazard. Most of the gold gets transported to the North of Paramaribo where the gold buyers are located.

P2: Tools such as KLEE, Cloud9, and Otter take this approach by implementing models for file system operations, sockets, IPC, etc. Forking the entire system state. Symbolic execution tools based on virtual machines solve the environment problem by forking the entire VM state. For example, in S2E each state is an independent VM snapshot that can be executed separately.

Question 1: In both passages, does environment problem have the same meaning?

(d) Verification of Yes/No labels via type-2 question.



Training

P1: This library was supposedly founded in 1945, but has started work in current object in 1947. During 1953-1956 it has played the role of the national library since the Kosovo National Library was closed. Academy of Sciences and Art is a **necessary institution** for the education system that is placed in Pristina. This institution was founded in 1975 as the Association of Science and Arts of Kosovo.

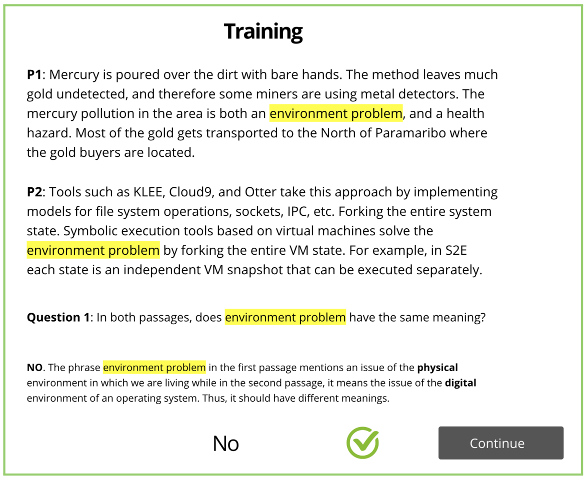
P2: This library was supposedly founded in 1945, but has started work in current object in 1947. During 1953-1956 it has played the role of the national library since the Kosovo National Library was closed. Academy of Sciences and Art is a **big supermarket** for the education system that is placed in Pristina. This institution was founded in 1975 as the Association of Science and Arts of Kosovo.

Question 3: In both passages, are **necessary institution** and **big supermarket** interchangeable (i.e., having the same meaning)?

NO. The phrase **necessary institution** means a required establishment of the livery stable which does not totally refer to a **big supermarket**.

Yes ✗

(e) Feedback is given when MTurkers give a wrong answer.



Training

P1: Mercury is poured over the dirt with bare hands. The method leaves much gold undetected, and therefore some miners are using metal detectors. The mercury pollution in the area is both an environment problem, and a health hazard. Most of the gold gets transported to the North of Paramaribo where the gold buyers are located.

P2: Tools such as KLEE, Cloud9, and Otter take this approach by implementing models for file system operations, sockets, IPC, etc. Forking the entire system state. Symbolic execution tools based on virtual machines solve the environment problem by forking the entire VM state. For example, in S2E each state is an independent VM snapshot that can be executed separately.

Question 1: In both passages, does environment problem have the same meaning?

NO. The phrase environment problem in the first passage mentions an issue of the **physical** environment in which we are living while in the second passage, it means the issue of the **digital** environment of an operating system. Thus, it should have different meanings.

No ✓

(f) or even a right answer.

Figure A12: Gorilla layouts shown to MTurkers to verify annotations in the first round.

K Data Sheet

We follow the documentation template provided by Gebru et al. 2021 (Gebru et al., 2021).

K.1 Motivation

For what purpose was the dataset created? Understanding phrases in context plays a vital role in solving many Natural Language Understanding (NLU) tasks such as question answering or reading comprehension. While there are *word-sense* disambiguation datasets like WiC, no such benchmarks exist for *phrases*. Existing phrase benchmarks compare only phrases without context and some of them contain numerous phrase pairs that have lexical overlap. The major drawback is no human annotation of how a phrase's meaning changes w.r.t the context. This motivates us to construct a Phrase-in-Context benchmark to drive the development of contextualized phrase embeddings in NLU.

Who created the dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)? Auburn University and Adobe Research.

K.2 Composition/collection process/preprocessing/cleaning/labeling and uses

We describe the data construction process, annotation and verification methods in our paper (See Sec. 3 and Sec. 4).

K.3 Distribution

Will the dataset be distributed to third parties outside the entity (e.g., company, institution, organization) on behalf of which the dataset was created? We release three datasets PS, PR (including PR-pass and PR-page) and PSD to the public.

How will the dataset be distributed (e.g., tarball on website, API, GitHub)? The datasets are released and can be viewed and downloaded on HuggingFace `https://huggingface.co/PiC` or on our website `https://phrase-in-context.github.io`.

When will the dataset be distributed? It has been released in July 2022.

What is the dataset format and how it can be read? We use JSON - a widely used data format for PiC dataset and follow a scheme of HuggingFace datasets to host it. Three datasets PS, PR and PSD in the PiC dataset are loaded as folows:

```
 # The following pip command is to install the HuggingFace library "datasets":
pip3 install datasets

 from datasets import load_dataset

 ps      = load_dataset("PiC/phrase_similarity")
 pr_pass = load_dataset("PiC/phrase_retrieval", "PR-pass")
 pr_page = load_dataset("PiC/phrase_retrieval", "PR-page")
 psd     = load_dataset("PiC/phrase_sense_disambiguation")
```

Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)? Our dataset is distributed under the CC-BY-NC 4.0 license.

K.4 Maintenance

How can the owner/curator/manager of the dataset be contacted (e.g., email address)? Thang Pham (`thangpham@auburn.edu`) and Anh Nguyen (`anh.ng8@gmail.com`) will be responsible for maintenance.

Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)? Yes. If we include more tasks or find any errors, we will correct the dataset. It will be updated on our website and also HuggingFace.

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so? They can contact us via email for the contribution.

Enhancing Dialogue Summarization with Topic-Aware Global- and Local-Level Centrality

Xinnian Liang[1]*, **Shuangzhi Wu**[2], **Chenhao Cui**[2], **Jiaqi Bai**[1], **Chao Bian**[2], **Zhoujun Li**[1]†
[1]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[2]Lark Platform Engineering-AI, Beijing, China
`{xnliang,cuich,bjq,lzj}@buaa.edu.cn`,`{wufurui,zhangchaoyue.0}@bytedance.com`

Abstract

Dialogue summarization aims to condense a given dialogue into a simple and focused summary text. Typically, both the roles' viewpoints and conversational topics change in the dialogue stream. Thus how to effectively handle the shifting topics and select the most salient utterance becomes one of the major challenges of this task. In this paper, we propose a novel topic-aware Global-Local Centrality (GLC) model to help select the salient context from all sub-topics. The centralities are constructed at both the global and local levels. The global one aims to identify vital sub-topics in the dialogue and the local one aims to select the most important context in each sub-topic. Specifically, the GLC collects sub-topic based on the utterance representations. And each utterance is aligned with one sub-topic. Based on the sub-topics, the GLC calculates global- and local-level centralities. Finally, we combine the two to guide the model to capture both salient context and sub-topics when generating summaries. Experimental results show that our model outperforms strong baselines on three public dialogue summarization datasets: CSDS, MC, and SAMSUM. Further analysis demonstrates that our GLC can exactly identify vital contents from sub-topics. [1]

1 Introduction

Online conversations have become essential to communication in our daily work and life. Due to the information explosion, dialogue summarization has become a vivid field of research in recent years, which is meaningful for many applications, e.g. online customer service (Liu et al., 2019; Zhu et al., 2020) and meeting summary (Feng et al., 2021).

Dialogue summarization aims to condense crucial information in a long dialogue into a short text like traditional summarization tasks. Differently, the main challenges of dialogue summarization are the viewpoints of multiple speaker roles (Lin et al., 2021, 2022; Qi et al., 2021; Zhang et al., 2022) and shifting topics (Chen and Yang, 2020; Zou et al., 2021; Liu et al., 2021) during the conversation process. As shown in Figure 1, summaries not only depend on the overall context but also needs the identification and selection of salient context in crucial sub-topics. We can see that the blue text in summaries is about sub-topic #1 “The reason why the product is shipped yet" and the orange text is about sub-topic #2 “The user decided to refund", which are aligned to the two salient sub-topics from dialogue utterances in the first and second block. The sub-topic #3 is useless for summaries. This example shows the necessity to model the salient context and sub-topics in the dialogue.

In this paper, we propose a novel topic-aware Global-Local Centrality (GLC) model to select salient contexts from all sub-topics. The centrality is an effective technique to measure the importance of sentences in a given document from unsupervised extractive summarization (Zheng and Lapata, 2019; Liang et al., 2021, 2022). The GLC contains global- and local-level centrality, which are used to capture the salience of sub-topics and content in each sub-topic respectively. Based on these centralities, we can guide the model to focus on the salient context and sub-topics when generating summaries. Specifically, we employ utterance-level representations to cluster utterances and obtain sub-topic centers and assign each utterance to one sub-topic. Then, we compute the global centrality over sub-topic centers to measure the importance of each sub-topic and the local centrality over utterances of each sub-topic to measure the importance of sub-topic content. Finally, we combine the two to re-weight the dialogue context representations for the decoder to generate summaries.

To evaluate the effectiveness of our proposed GLC, we apply the GLC to three different types of

---

*Contribution during internship at ByteDance Inc.

†Corresponding Authors.

[1] `https://github.com/xnliang98/bart-glc`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 27–38
May 2-6, 2023 ©2023 Association for Computational Linguistics

	User: 我买的电话手表怎么还没发货？(Why hasn't the smart watch I bought been shipped yet?) **Agent:** 您好，还请您稍等，正在为您查询~，请问是这个商品吗？[商品快照]（ Hello, please wait a moment. I am inquiring for you, is this product? [PRODUCT SNAPSHOT]) **User:** 是的。[数字]号下的单，一点反应都没有。(Yes. I placed an order at [Date], but no response at all.) **Agent:** [商品快照] 这个商品缺货的亲。([PRODUCT SNAPSHOT] This product is out of stock.) **User:** 那为什么我可以下单呢？也不告知我一下。(So why can I place an order? Don't tell me either) **Agent:** 这个是赠品哦亲。您电话对吗。妹子联系采销核实下具体时间。然后[数字]个小时内告知您。您看可以吗。(This is a giveaway. Is your phone number correct? I will contacts the purchase and sales to verify the specific time. Then [NUMBER] hours to let you know. Do you think this is ok?)	**Topic #1: The reason why the product is shipped yet.**
	User: 那直接申请退款吧，不买了。(Then I will apply for a refund directly, I will not buy it.) **Agent:** 哦哦，不要了吗？这边查看是在退款了，取消订单就无法恢复了呢。(Ok, don't you? The check here is a refund, and the cancellation of the order cannot be restored.) **User:** 嗯，退了吧。(Well, refund it.) **Agent:** 储蓄卡支付的吗？储蓄卡支付，原路返回的，周期是[数字]-[数字]个工作日内的。(Are you paying by debit card? Debit card payment, return the same way, the cycle is within [number]-[number] working days.)	**Topic #2: The user decided to refund.**
	User: 是的。(Yes.) **Agent:** 辛苦注意查收。请问还有其他还可以帮到您的吗? (Please pay attention to check the refund. Is there anything else I can help you with?) **User:** 好的没有了。(I have no more question.) **Agent:** 感谢您对我们的支持，祝您生活愉快，再见! (Thanks for your support. I wish you have a happy life, bye!)	**Topic #3: Useless information**
User Summary	用户询问购买的电话手表为什么还没发货，得知缺货后直接要求退款。(The user asks why the purchased smart watch has not been shipped, and directly asks for a refund after learning that it is out of stock.)	
Agent Summary	客服回复电话手表是赠品缺货，并准备向采销核实具体时间后告知用户。用户申请退款，客服提醒用户取消订单后无法恢复，随后告知储蓄卡支付会在一定时间内自动返还。(The customer service replies to the smart watch is a giveaway and is out of stock, and is ready to inform the user after verifying the specific time. When the user applies for a refund, the customer service reminds the user that the order cannot be restored after canceling the order, and then informs that the debit card payment will be automatically returned within a certain period of time.)	
Final Summary	用户询问购买的电话手表为什么还没发货。客服查询后回答赠品缺货，并准备向采销核实具体时间后告知用户。用户要求直接申请退款。客服提醒用户取消订单后无法恢复，随后告知储蓄卡支付会在一定时间内返还。(The user asks why the purchased phone watch has not been shipped. After customer service inquiries, they will answer that the gift is out of stock, and prepare to inform the user after verifying the specific time. The user requests to apply for a refund directly. The customer service reminded the user that the cancellation of the order cannot be restored, and then informed that the debit card payment will be returned within a certain period of time.)	

Figure 1: An example from the CSDS dataset. The dialogue contains 3 different sub-topics. The blue text represents sub-topic #1 and the red text represents sub-topic #2. The sub-topic #3 is useless information.

seq2seq structure: PGN, BERTAbs, and BART, and verify them on three public dialogue summarization datasets: CSDS, MC, and SAMSUM. CSDS and MC are two Chinese role-oriented summarization datasets that not only need to generate the overall summary of the dialogue but also need to generate role-oriented summaries for specific speakers in the dialogue as shown in Figure 1. SAMSUM is a widely used English dialogue summarization dataset. To generate role-oriented summaries, in this paper, we directly employ role prompts to guide the model to generate proper summaries. And the representations of role prompts can add role information to the centrality computation. Experimental results show that our GLC can improve the performance of all these seq2seq structures on three datasets. And the GLC-based BART model obtains new state-of-the-art results on the CSDS and MC.

Our contributions can be summarized as 1) We propose a novel topic-aware Global-Local Centrality (GLC) model to guide the model to identify the salient contexts and sub-topics in the dialogue. 2) Our GLC can bring improvement to different seq2seq models by easily plugging in and does not add any extra parameters to the seq2seq models. 3) The GLC-based BART model achieves new state-of-the-art results on CSDS and MC. Besides, extension studies prove our GLC can effectively capture vital sub-topics.

2 Methodology

Figure 2 shows the main structure of our proposed topic-aware global-local centrality (GLC) model. The seq2seq framework with GLC is on the left of Figure 2, which consists of the bi-directional encoder, global-local centrality model, and auto-regression decoder. The detail of GLC is on the right of Figure 2, which consists of global centrality and local centrality. In this section, we introduce them step by step.

2.1 Task Formulation

Firstly, we formulate the dialogue summarization task and role-oriented summarization task. Given a dialogue $\mathcal{D}$ with N utterances $\{u_1, \ldots, u_N\}$ with M roles $\{r_1, \ldots, r_M\}$. Each utterance u_i contains a speaker role r_j and sentence s_i. We simply concatenate them by ":" and get utterance $u_i = r_j : s_i$. For role-oriented summarization tasks, the data contains different summary y^{r_j} for different speaker roles r_j. In this paper, we employ y^{user} and y^{agent} to represent summaries of two different roles and y^{final} to represent the overall summary of the whole dialogue. It is deserved to mention that our method can also be easily applied to datasets with more than two speaker roles by introducing different role prompts. Normal dialogue summarization

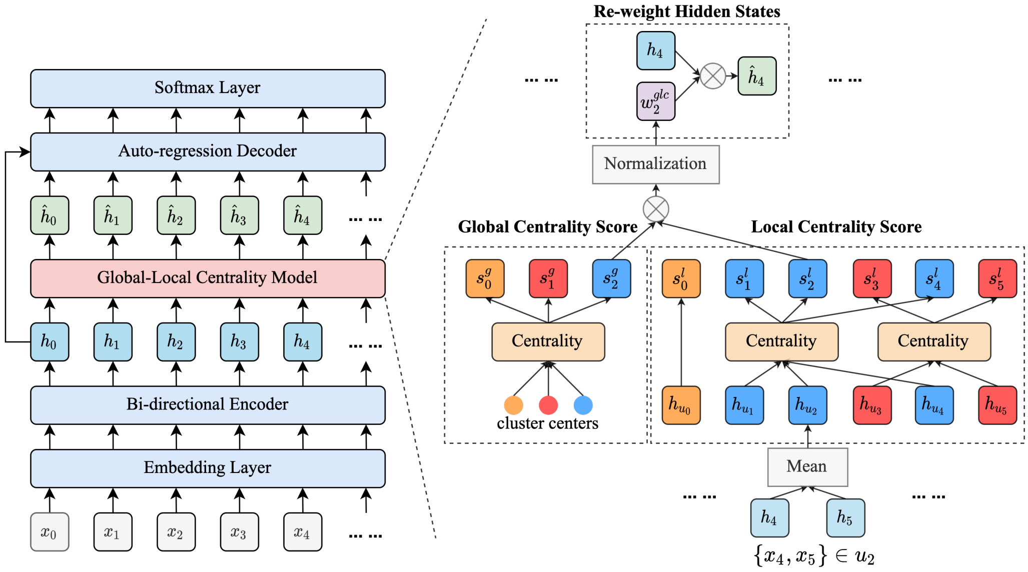


Figure 2: The main structure of our proposed method. The left is the framework of seq2seq with the GLC model. The right is the detailed process of our proposed GLC model.

task aims to generate overall summaries y^{final} and role-oriented summarization task aims to generate role-specific summaries $y^{[user|agent|final]}$ from the input dialogue $\mathcal{D} = \{u_1, \ldots, u_N\}$ according to the given role.

2.2 Role Prompts

For role-oriented summarization tasks, previous works train multiple independent models for different role summaries, which is proven to hurt the performance of model (Lin et al., 2022) and needs more computation resources. In this paper, we employ a simple but effective trick to ensure that we only need to train a single model to obtain different role-specific summaries and overall summaries. Specifically, we use the prompts to control the generation of different kinds of summaries, which attach "[User Summary]", "[Agent Summary]", and "[Final Summary]" to the start of each dialogue as input to guide the model to generate required summaries. After that, the input context is re-formalized as "[Role Prompt] Dialogue Contexts" and then tokenized as T tokens/words $\{x_t\}_{t=1}^{T}$ for the encoder of seq2seq model.

2.3 Bi-directional Encoder

The bi-directional encoder is used to get tokens the semantic vector representations $\{h_t\}_{t=1}^{T}$ by capture bi-directional context information from tokens $\{x_t\}_{t=1}^{T}$ as follows:

$$\{h_t\}_{t=1}^{T} = \texttt{Encoder}(\{x_t\}_{t=1}^{T}) \qquad (1)$$

Then, we use the average of tokens vectors in each utterance as the semantic representations of dialogue utterances as follows:

$$h_{u_i} = \frac{1}{|u_i|} \sum_{t} x_t, x_t \in u_i \qquad (2)$$

After that, we can get the token-level semantic representations $\{h_t\}_{t=1}^{T}$ and the utterance-level semantic representations $\{h_{u_i}\}_{i=0}^{N}$, where h_{u_0} is the vector representation of the attached role prompt, if role prompt is used.

2.4 Global-Local Centrality Model

Before feeding the representations into the decoder to generate the final summaries, we employ our proposed global-local centrality (GLC) model to reweight the vector representations to identify salient facts in sub-topics over previous utterance-level semantic representations $\{h_{u_i}\}_{i=0}^{N}$.

Firstly, our GLC obtains several cluster center points $\{c_k\}k = 1^K$, which represent the center of sub-topics in the vector space. Then each utterance is assigned to the nearest center. As shown in Figure 2, utterances with the same color belong to the same sub-topic. We compute the global centrality score based on the cluster center representations to measure the importance of sub-topics and the local centrality score based on the utterance representations to measure the importance of each utterance belonging to the same sub-topic. Then, we employ their combination to get global-local centrality weights, which are used to re-weight the token-level vector representations. Finally, the re-weighted token-level vector representations are fed into the decoder to generate the summary. Our GLC can be directly plug-in any seq2seq structures, which makes it flexible.

2.4.1 Obtain Cluster Centers

To obtain the clusters, we directly call the K-Means algorithm, which is effective and widely used for cluster tasks. And we all know setting the number of cluster centers for the K-Means algorithm is crucial and hard for the final results. However, we empirically find that we can set it as the number of utterances $(N+1)$ and then assign each utterance to the nearest cluster center point in the vector space. After that, we find that many cluster centers have no assigned utterances and can be dropped. Based on this, we assume $K < (N+1)$ cluster centers $\{c_i\}_{i=1}^{K}$ are kept and note the vector representations of them as $\{h_i^c\}_{i=1}^{K}$.

$$\{h_i^c\}_{i=1}^{K} = \texttt{KMeans}(\{h_{u_i}\}_{i=0}^{N}) \tag{3}$$

And after the assignment of utterances, we can get K clusters $\{\mathcal{C}_k\}_{i=k}^{K}$, which contain utterances with similar sub-topics. Each $\mathcal{C}_k$ contains several utterances and one cluster center point c_k. Through the previous method, we do not need to manually set the number of cluster centers for the K-Means algorithm.

2.4.2 Global Centrality

The global centrality score aims to measure the importance of each sub-topic by computing degree centrality based on the cluster center representations $\{h_k^c\}_{k=1}^{K}$. Each cluster center can be seen as one node on the graph, and the edge value between nodes k and j is $(h_k^c)^T \cdot h_j^c$. Then, the degree centrality of each cluster can be computed as follows:

$$\texttt{Cen}(c_k) = \sum_j (h_k^c)^T \cdot h_j^c \tag{4}$$

where $\texttt{Cen}(c_k)$ represents the importance of the cluster/sub-topic k in the dialogue. Then we normalize the score $\texttt{Cen}(c_k)$ by $\frac{\texttt{Cen}(c_k)}{||\{\texttt{Cen}(c_k)\}_{k=1}^{K}||_2}$.

2.4.3 Local Centrality

The local centrality score aims to measure the importance of utterances in each cluster by computing the degree centrality. Each utterance can be seen as one node on the graph, and the edge value between nodes i and j is $(h_{u_i})^T \cdot h_{u_j}$. Then, the centrality of each utterance in the same cluster $\mathcal{C}_k$ can be computed as follows:

$$\texttt{Cen}(u_i) = \sum_j (h_{u_i})^T \cdot h_{u_j}, u_i, u_j \in \mathcal{C}_k \tag{5}$$

where $\texttt{Cen}(u_i)$ represents the importance of utterances in the k-th cluster/sub-topic. Then we normalize the score $\texttt{Cen}(u_i)$ the same as the previous global centrality score.

2.4.4 Global-Local Centrality Weight

We can obtain the importance of each cluster (global centrality score) and the importance of utterances in each cluster (local centrality score) by the previous two steps. The most important utterance in the most important sub-topic should be assigned more attention when generating the summary. So we obtain global-local centrality weight for each utterance in the dialogue by simply multiplying two centrality scores as follows:

$$w_i^{glc} = \texttt{Cen}(u_i) \cdot \texttt{Cen}(c_k), u_i \in \mathcal{C}_k \tag{6}$$

Finally, we employ the global-local centrality weights to re-weight the token-level vector representations $\{h_t\}_{t=1}^{T}$ as follows:

$$\hat{h}_t = w_i^{glc} \cdot h_t, x_t \in u_i \tag{7}$$

Where each token uses the global-local centrality weight w_i^{glc} of its utterance u_i to re-weight the vector representation h_t. The token level representations $\{h_t\}_{t=1}^{T}$ are converted into $\{\hat{h}_t\}_{t=1}^{T}$.

2.5 Auto-regression Decoder

The auto-regression decoder generates the final summary based on the re-weighted context representations $\{\hat{h}_t\}_{t=1}^{T}$ as follows:

$$P(\hat{y}) = \texttt{Decoder}(\{\lambda \cdot \hat{h}_t + (1-\lambda) \cdot h_t\}_{t=1}^{T}) \tag{8}$$

where λ is a hyper-parameter to control the influence of GLC, the default value of λ is 0.5. In the training stage, the model learns the optimal parameters θ by minimizing the negative log-likelihood.

3 Experiments

3.1 Datasets and Metrics

We evaluate our method on three public datasets: CSDS (Lin et al., 2021)[2], MC (Song et al., 2020)[3],

---

[2] https://github.com/xiaolinAndy/CSDS

[3] https://github.com/cuhksz-nlp/HET-MC

CSDS	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore
PGN	55.58/53.55/50.20	39.19/37.06/35.12	53.46/51.05/47.59	30.03/29.64/28.25	77.96/78.68/76.13
PGN-both	57.20/56.08/51.62	40.37/39.10/36.50	55.14/53.85/49.12	32.58/33.54/29.78	78.69/79.52/76.74
PGN-GLC	57.94/57.14/52.85	40.97/39.55/37.14	55.68/54.25/49.86	32.95/33.87.30.15	78.93/79.86/76.98
BERT	53.87/52.72/49.57	37.59/36.39/33.82	52.40/50.44/46.83	29.90/30.17/26.99	78.52/79.23/76.39
BERT-both	57.24/54.36/51.92	40.12/40.70/36.37	54.87/55.17/49.52	32.13/32.04/29.23	79.85/80.70/77.23
BERT-GLC	57.59/55.14/52.34	41.28/41.84/36.48	55.74/55.86/50.16	32.75/32.64/29.81	79.89/80.71/77.28
BART	59.07/58.78/53.89	43.72/43.59/40.24	57.11/56.86/50.85	34.33/34.26/31.88	79.74/80.67/77.31
BART-both	59.21/58.93/54.01	43.88/43.69/**40.32**	57.32/57.28/51.10	34.75/34.49/32.30	79.72/80.64/77.30
BART-GLC	**60.07/61.42/54.59**	**44.67/45.83/**40.02	**58.10/59.25/52.43**	**35.89/36.43/32.58**	**80.10/81.83/77.61**

Table 1: Results on the CSDS dataset test set.

MC	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore
PGN	85.32/94.82/82.56	81.25/94.32/77.91	84.34/94.77/81.47	71.50/87.66/68.10	92.90/97.60/91.74
PGN-both	85.98/95.10/83.37	81.93/94.59/78.78	84.94/95.06/82.20	72.77/87.82/69.63	93.23/97.71/92.15
PGN-GLC	86.57/95.31/83.97	82.04/94.88/79.16	85.37/96.48/82.84	73.02/88.11/70.04	93.47/97.95/92.36
BERT	84.07/95.10/81.53	79.90/94.48/76.78	83.04/95.06/80.30	68.19/87.20/64.09	92.68/97.86/91.71
BERT-both	84.69/95.18/82.02	80.76/94.62/77.54	83.68/95.14/80.84	69.33/87.40/65.40	93.02/97.90/91.91
BERT-GLC	85.64/95.49/82.87	81.44/94.97/78.05	84.16/96.10/81.57	69.84/87.94/66.01	93.15/97.92/92.36
BART	88.37/95.42/86.33	84.75/94.99/82.33	87.38/95.37/85.30	73.68/90.29/68.93	93.65/97.94/92.63
BART-both	88.52/95.63/87.06	85.22/95.42/82.89	87.75/95.91/85.78	73.87/90.70/69.31	93.69/97.88/92.69
BART-GLC	**89.55/96.84/88.47**	**86.47/96.14/84.62**	**88.56/96.23/86.77**	**74.19/91.32/70.18**	**94.17/98.25/92.96**

Table 2: Results on the MC dataset test set.

and SAMSUM (Gliwa et al., 2019)[4]. The statistical information of them is shown in the appendix. CSDS is the first role-oriented dialogue summarization dataset, which provides separate summaries for user and agent (customer service). MC is a Chinese medical inquiry dataset containing question summaries of patients and suggestion summaries of doctors. We note them as the user and agent summary. For the MC dataset, we follow the data process and data split from RODS (Lin et al., 2022). SAMSUM is a widely used English dialogue summarization dataset to evaluate the performance of models.

We employ lexical-level and semantic-level metrics to evaluate the performance of all models. Specifically, we use lexical level ROUGE-1/2/L (Lin, 2004)[5] and BLEU (Papineni et al., 2002)[6], which measure the similarity of references and generated summaries by computing the n-gram overlap of them. We use semantic level BERTScore (Zhang* et al., 2020)[7] and MoverScore (Zhao et al., 2019)[8], which employ pre-trained language models to map the text into low-dimensional vectors in semantic space and then measure the similarity by computing the similarity by cosine similarity or word mover distance. We can evaluate the performance of each model comprehensively through the previous metrics. And all reported results are the average results of three different model checkpoints. The results of Moverscore on three datasets can be found in the appendix.

3.2 Baselines

We applied our GLC on three widely used seq2seq models: PGN (See et al., 2017), BERTAbs (Liu and Lapata, 2019), and BART (Lewis et al., 2020; Shao et al., 2021). **PGN** model is an LSTM-based seq2seq model without pre-training. **BERTAbs** is a BERT-based model, which employs BERT as the encoder and adds several transformer blocks as the decoder to generate summaries. We note it as BERT. **BART** is a pre-trained transformer-based seq2seq model, which achieves the best results on many generation tasks. We add our proposed GLC into the previous three models and note them as **PGN-GLC**, **BERT-GLC**, and **BART-GLC**. We also compare our method with previous SOTA models: **PGN-both** and **BERT-both** from (Lin et al.,

---

[4] https://huggingface.co/datasets/samsum

[5] https://pypi.org/project/rouge-score/

[6] https://github.com/mjpost/sacreBLEU

[7] https://github.com/Tiiiger/bert_score

[8] https://github.com/AIPHES/emnlp19-moverscore

SAMSUM	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore
PGN	40.08	15.28	36.63	37.49	80.67
PGN-GLC	41.11	16.24	37.31	38.10	81.54
BERT	50.34	24.71	46.63	46.98	88.72
BERT-GLC	51.18	25.26	47.07	47.66	89.64
BART	53.12	27.95	49.15	49.28	92.14
BART-GLC	**53.74**	**28.83**	**49.62**	**50.36**	**92.77**

Table 3: Results on the SAMSUM dataset test set.

2022), which proposed a role-interaction attention mechanism for the decoder. We reproduce it in the BART model as **BART-both**. For SAMSUM, we do not compare with BART-both due to this dataset does not contain role-oriented summaries.

3.3 Implementation Details

We use Chinese-BART-base[9] and BART-large[10] to initialize our transformer-based seq2seq model for Chinese and English datasets respectively. We train all BART models on 4xV100 GPUs and PGN/BERT-based models on 1xV100 GPU. For all models, the maximum input length is 512, the maximum generated summary length is 150, and the beam size is 3. For BART-based models, the learning rate is 1e-4 with 10% warmup steps, the total batch size is 64, and the training epochs are 5. For PGN/BERT-based models, we follow the settings from (Lin et al., 2022).

3.4 Results

The main results of the two role-oriented dialogue summarization datasets are shown in Table 1-2. Each block has three values, representing the final summary/user summary/agent summary from left to right. We can see that our proposed GLC can bring significant improvement to PGN, BERTAbs, and BART on the two datasets and BART-GLC achieves new state-of-the-art results. It is deserved to mention that our model does not need to modify any structure of the seq2seq structure and only needs to train one model for different summaries. We can see that the gain of metrics on the CSDS is better than on the MC, due to the summary of the MC dataset being highly similar to the input dialogue contexts. The results of the BERT-based model sometimes is worse than the PGN-based, we guess the reason is the prior knowledge learned in the pre-training stage of BERT is not suitable for the generation tasks. The improvement of lexical level metrics is more conspicuous than semantic level metrics due to the change of several words that may not affect the semantics of generated sentences. Overall, our proposed GLC is proved effective for the role-oriented dialogue summarization task with results on the two datasets.

The main results of the English dialogue summarization dataset are shown in Table 3. Because the SAMSUM does not provide role-specific summaries, we only report the performance of overall final summaries. From the results, we can see that our GLC can also bring significant improvements to three different seq2seq structures. We can see that the BERTScore is very high on SAUSUM, we guess that because the gold reference of this dataset is very short and this makes the semantic similarity between generated summaries and gold summaries close. The results of SAMSUM demonstrated the effectiveness and generalization of our proposed method.

4 Discussion

We conduct many external experiments on the CSDS dataset to further analyze the effectiveness of our proposed GLC. And more discussions are shown in the appendix.

4.1 Ablation Study

To understand the impact of each component of our proposed GLC model, we compare the full BART-GLC with the following variants: (1) **BART:** three fine-tuned BART models for different summaries (final/user/agent); (2) **BART+Prompt:** singe BART model with role prompts; (3) **BART+GC:** three BART models using global centrality scores

---

[9] https://huggingface.co/uer/bart-base-chinese-cluecorpussmall

[10] https://huggingface.co/facebook/bart-large

	ROUGE-1
BART	59.07/58.78/53.89
+Prompt	59.42/58.96/54.03
+GC	59.64/59.55/54.24
+LC	59.37/59.47/54.11
+GC,LC	59.84/60.91/54.43
BART-GLC	**60.07/61.42/54.59**

Table 4: Ablation study on the CSDS dataset.

	Win	Loss	Tie
CSDS&MC	56.4	2.4	41.2
SAMSUM	51.8	3.2	45.3

Table 5: Human evaluation results.

to re-weight hidden states; (4) **BART+LC:** three BART models using local centrality scores to re-weight hidden states; (5) **BART+GC,LC:** three BART models using global-local centrality scores to re-weight hidden states. The results of these models are shown in Table 4. From the results, we can see that all three components can bring improvement to the BART model, and the global-local centrality brings the greatest improvement. Interesting, The improvement brought by the combination of global and local centrality is far greater than the improvements they bring separately. This proves that global and local centrality are mutually beneficial.

4.2 Human Evaluation

We use human evaluation (Fang et al., 2022) to verify that our model outperforms the baseline. Specifically, we randomly sample 100 examples from three datasets and ask five NLP researchers to give a comparison between our model and baseline models. The evaluation results are represented as win, loss, and tie, respectively indicating that the quality of generated summary by BART-GLC is better, weaker, or equal to the strong baselines. Annotators were asked to judge from two aspects: fluency (whether contains grammatical and factual errors) and coverage (whether contains salient sub-topic information in the dialogue). For two role-oriented dialogue summarization datasets CSDS and MC, our model is compared with BART-both. For SAMSUM, our model is compared with BART. From the results in Table 5, we can see that our model is better than the baseline. Annotators tend to give ties on SAMSUM dataset. This may be caused by the length of summaries is short, which makes it hard to judge whether the summary is better or worse than the baseline model.

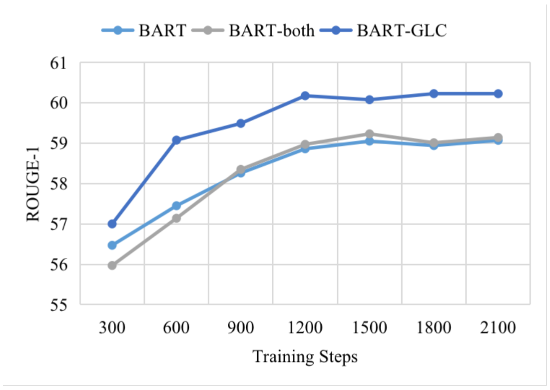


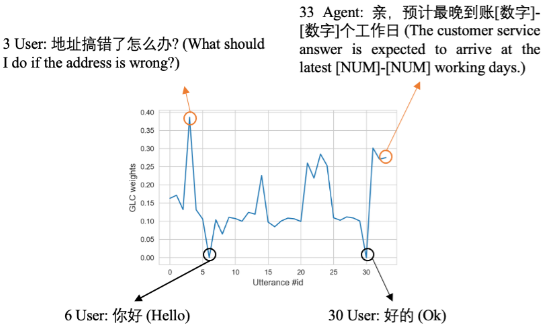
Figure 3: The ROUGE-1 score of different training step checkpoints.

4.3 Convergence of Training

We also compare the training convergence speed with BART and BART-both to prove our proposed GLC can bring effective prior knowledge for the seq2seq model. As shown in Figure 3, we can see that BART-GLC achieves comparable performance at 900 steps during training and reaches the SOTA results at 1,200 steps. This phenomenon demonstrates that our GLC brings prior knowledge into the model and speeds up the model training.

4.4 Case study

We select one example from the test set to show the ability of our proposed GLC in Figure 4. On the upper-left of this figure are the GLC weights and the corresponding utterances. In the bottom-left of the figure is generated summary of our proposed BART-GLC. On the right of the figure is the input dialogue and each color refers to one sub-topic. From this case, we can see that the final summary focus on two sub-topics: “How to modify user’s order” and “Questions about refunds”. And from the color on the right of this figure, we can see that our GLC can catch them accurately. Interestingly, generic utterances are aggregated into one topic (e.g. hello). In the upper-left of this figure is the GLC weights and we can see that utterances, which are related to the final summary and belong to the vital sub-topics, are assigned high weights. This proves the global-local centrality exactly identified



BART-GLC Generated Summary:
Topic #1: How to modify user's wrong order.
1 用户询问地址搞错了怎么办。(The user asks what to do if the address is wrong.)
2 客服回答正常情况下订单提交成功是不支持修改的。(The customer service replied that the successful submitted order does not support modification.)
3 用户询问申请退款为什么有个退不了。(The user asked why the application for a refund could not be refunded.)
4 客服回答跨城市是不支持修改的。(The customer service replies that cross-city modification is not supported.)
Topic #2: Questions about refunds.
5 用户询问券是否原路退。(The user asks if the coupon is returned the same way.)
6 客服回答都会退的。(Customer service replies will be returned.)
7 用户询问什么时候到账。(The user asks when the refund will arrive.)
客服回答预计最晚到账[数字]-[数字]个工作日。(The customer service answer is expected to arrive at the latest [NUM]-[NUM] working days.)

Dialogue:
0 Role Prompt: [对话摘要] ([Final Summary])
1 Agent: 有什么问题我可以帮您处理或解决呢? (Is there any question I can help you with or solve?)
2 User: 我刚那单 (My order just now)
3 User: 地址搞错了怎么办? (What should I do if the address is wrong?)
4 User: 能帮我修改吗? (Can you help me modify it?)
5 User: 还是申请退款? (Or apply for a refund?)
6 User: 你好 (Hello)
7 User: ? (?)
… …
14 Agent: 您好，正常情况下 订单提交成功是不支持修改的呢，麻烦您看下订单详情页是否有修改的按钮，如有，您可点击修改末级地址/联系人/电话号码/送货时间信息，如没有修改按钮，说明订单已不支持修改了呢，还请您理解哦~ (Hello, under normal circumstances, the order is submitted successfully and does not support modification. Please check whether there is a modification button on the order details page. If so, you can click to modify the last-level address/contact/phone number/delivery time information , if there is no modification button, it means that the order no longer supports modification, please understand~)
15 User: 那申请退款 (I apply for a refund?)
16 User: 再拍 (then to place an order again?)
… …
20 Agent: 亲，实在抱歉，咨询量过大，信息回不过来 (I'm really sorry, the amount of inquiries is too large, and the information cannot be returned.)
21 User: 我申请了退款为什么有个退不了(I applied for a refund, why can't I get a refund?)
22 User: 另一个城市的订单 (order from another city)
23 Agent: 请您提供订单号的 (Please provide the order number.)
24 Agent: 跨城市是不支持修改的，亲 (Cross-city modification is not supported)
25 Agent: 小妹为您取消订单，您重新下单 (I cancel the order for you, you place a new order)
… …
28 User: 那券不退吗，是原路返回吗? (Is the coupon non-refundable? Is it going back the same way?)
29 Agent: 都会退的哦 (All of them will be returned.)
30 User: 好的 (Ok)
31 Agent: 请问还有其他还可以帮到您的吗? (Is there anything else I can help you with?)
32 User: 什么时候到账 (When will the refund arrive)
33 Agent: 亲，预计最晚到账[数字]-[数字]个工作日 (The customer service answer is expected to arrive at the latest [NUM]-[NUM] working days.)

Figure 4: One case from the CSDS test set. Each color refers to one sub-topic. In the upper-left of this figure are the GLC weights and the corresponding utterances. In the bottom-left of the figure is generated summary of our proposed BART-GLC. On the right of the figure is the input dialogue.

salient topics and utterances.

5 Related Work

Dialogue summarization has caught more and more attention in recent years and is widely used in various domains, e.g. meeting summarization (Carletta et al., 2006; Feng et al., 2021), daily dialogue summarization (Krishna et al., 2021; Chen et al., 2021; Zhong et al., 2021), etc. Different from traditional summarization tasks, dialogue summarization needs to identify the role of speakers and capture the change of sub-topics during the dialogue. Besides, the dialogue summarization task has less labeled data and longer inputs. All of these make dialogue summarization harder to solve (Chen and Yang, 2020; Zhang et al., 2021b; Feng et al., 2021; Lin et al., 2022).

Recent dialogue summarization models can be categorized into three types: 1) data augmentation methods (Feng et al., 2021; Chen and Yang, 2021; Khalifa et al., 2021), which attempt to construct more pseudo-data to train a better model; 2) topic-based models (Zou et al., 2021; Liu et al., 2021; Qi et al., 2021), which track the change of topic information in the dialogue to generate more focused summary; and 3) semantic structure-based models (Liu and Chen, 2021; Fu et al., 2021; Zhang et al., 2021a; Lei et al., 2021; Zhao et al., 2021; Zhang et al., 2022), which employs semantic structures to enhance the summarization model.

However, they ignored the sub-topics information in the dialogue utterances, which is crucial for dialogue summarization. Recently, Zhao et al. (2020) modified the attention mechanism to focus on the topic words, which can force the model to learn the topic information. Zou et al. (2021) employed Neural Topic Model to model the global level topic information. Liu et al. (2021) tried to model the change of sub-topics by introducing contrastive learning. Differently, in this paper, we bring the centrality, that has been widely used in unsupervised summarization (Zheng and Lapata, 2019; Liang et al., 2021, 2022), into the dialogue summarization task and proposed a novel topic-aware Global-Local Centrality model to capture salient dialogue utterances and sub-topics at the same time. Our proposed method is effective and more flexible.

6 Conclusion

In this paper, we bring the centrality into dialogue summarization tasks and proposed a novel topic-aware Global-Local Centrality (GLC) model for better capturing the sub-topic information in the dialogue utterances. Our GLC can be easily applied to any seq2seq structure and bring improvement to their performance. Experiments and further analysis demonstrated that GLC can effectively identify vital sub-topics and salient content in the dialogue. In future work, we will try to extend our work to datasets with longer inputs.

Limitations

Our model also has some limitations: 1) The computation of sub-topic centers brings extra inference time into the basic seq2seq models. 2) We did not try to evaluate our model on longer dialogue summarization datasets. 3) We did not build a specific mechanism for different roles in role-oriented dialogue summarization task. We will try to solve these limitations in future work.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant Nos. 62276017, U1636211, 61672081), the 2022 Tencent Big Travel Rhino-Bird Special Research Program, and the Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2021ZX-18).

References

Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, Guillaume Lathoud, Mike Lincoln, Agnes Lisowska, Iain McCowan, Wilfried Post, Dennis Reidsma, and Pierre Wellner. 2006. The ami meeting corpus: A pre-announcement. In *Machine Learning for Multimodal Interaction*, pages 28–39, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4106–4118, Online. Association for Computational Linguistics.

Jiaao Chen and Diyi Yang. 2021. Simple conversational data augmentation for semi-supervised abstractive dialogue summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6605–6616, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074, Online. Association for Computational Linguistics.

Yue Fang, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Bo Long, Yanyan Lan, and Yanquan Zhou. 2022. From spoken dialogue to formal summary: An utterance rewriting for dialogue summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3859–3869, Seattle, United States. Association for Computational Linguistics.

Xiachong Feng, Xiaocheng Feng, Libo Qin, Bing Qin, and Ting Liu. 2021. Language model as an annotator: Exploring DialoGPT for dialogue summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1479–1491, Online. Association for Computational Linguistics.

Xiyan Fu, Yating Zhang, Tianyi Wang, Xiaozhong Liu, Changlong Sun, and Zhenglu Yang. 2021. RepSum: Unsupervised dialogue summarization based on replacement strategy. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6042–6051, Online. Association for Computational Linguistics.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Muhammad Khalifa, Miguel Ballesteros, and Kathleen McKeown. 2021. A bag of tricks for dialogue summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8014–8022, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kundan Krishna, Sopan Khosla, Jeffrey Bigham, and Zachary C. Lipton. 2021. Generating SOAP notes from doctor-patient conversations using modular summarization techniques. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4958–4972, Online. Association for Computational Linguistics.

Yuejie Lei, Fujia Zheng, Yuanmeng Yan, Keqing He, and Weiran Xu. 2021. A finer-grain universal dialogue semantic structures based model for abstractive dialogue summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1354–1364, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xinnian Liang, Jing Li, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2022. Improving unsupervised extractive summarization by jointly modeling facet and redundancy. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1546–1557.

Xinnian Liang, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2021. Improving unsupervised extractive summarization with facet-aware modeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1685–1697, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Haitao Lin, Liqun Ma, Junnan Zhu, Lu Xiang, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2021. CSDS: A fine-grained Chinese dataset for customer service dialogue summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4436–4451, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haitao Lin, Junnan Zhu, Lu Xiang, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2022. Other roles matter! enhancing role-oriented dialogue summarization via role interactions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2545–2558, Dublin, Ireland. Association for Computational Linguistics.

Chunyi Liu, Peng Wang, Jiang Xu, Zang Li, and Jieping Ye. 2019. Automatic dialogue summary generation for customer service. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '19, page 1957–1965, New York, NY, USA. Association for Computing Machinery.

Junpeng Liu, Yanyan Zou, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Caixia Yuan, and Xiaojie Wang. 2021. Topic-aware contrastive learning for abstractive dialogue summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1229–1243, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Zhengyuan Liu and Nancy Chen. 2021. Controllable neural dialogue summarization with personal named entity planning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 92–106, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

MengNan Qi, Hao Liu, YuZhuo Fu, and Ting Liu. 2021. Improving abstractive dialogue summarization with hierarchical pretraining and topic segment. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1121–1130, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Yan Song, Yuanhe Tian, Nan Wang, and Fei Xia. 2020. Summarizing medical conversations via identifying important utterances. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 717–729, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Xinyuan Zhang, Ruiyi Zhang, Manzil Zaheer, and Amr Ahmed. 2021a. Unsupervised abstractive dialogue summarization for tete-a-tetes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14489–14497.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summ^n: A multi-stage summarization framework for long input dialogues and documents. In *Proceedings of the 60th*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1592–1604, Dublin, Ireland. Association for Computational Linguistics.

Yusen Zhang, Ansong Ni, Tao Yu, Rui Zhang, Chenguang Zhu, Budhaditya Deb, Asli Celikyilmaz, Ahmed Hassan Awadallah, and Dragomir Radev. 2021b. An exploratory study on long dialogue summarization: What works and what's next. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4426–4433, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lulu Zhao, Weiran Xu, and Jun Guo. 2020. Improving abstractive dialogue summarization with graph structures and topic words. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 437–449, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Lulu Zhao, Weihao Zeng, Weiran Xu, and Jun Guo. 2021. Give the truth: Incorporate semantic slot into abstractive dialogue summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2435–2446, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Hong Kong, China. Association for Computational Linguistics.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6236–6247, Florence, Italy. Association for Computational Linguistics.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online. Association for Computational Linguistics.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 194–203, Online. Association for Computational Linguistics.

Yicheng Zou, Lujun Zhao, Yangyang Kang, Jun Lin, Minlong Peng, Zhuoren Jiang, Changlong Sun, Qi Zhang, Xuanjing Huang, and Xiaozhong Liu. 2021. Topic-oriented spoken dialogue summarization for customer service with saliency-aware topic modeling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14665–14673.

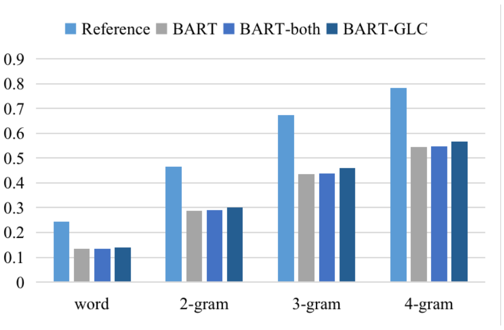


Figure 5: Percentage of novel words/n-grams in the reference and generated summaries of the CSDS test set.

A Datasets

	CSDS	MC	SAMSUM
Train Size	9,101	29,324	14,732
Val. Size	800	3,258	818
Test Size	800	8,146	819
Input Length	321.92	292.21	94.52
User Sum. Length	37.28	22.37	-
Agent Sum. Length	48.08	95.32	-
Final Sum. Length	83.21	114.54	20.34

Table 6: Statistical information of three datasets.

The statistical information of three datasets is shown in Table 6.

B Moverscore Results

For Moverscore, we employ chinese-bert-wwm-ext[11] to get the contextual embeddings of Chinese text input. Because Lin et al. (2021) did not provide they use what Chinese representation model, we use chinese-bert-wwm-ext to re-evaluate all their results and report in Table 7.

B.1 How abstractive is our model?

An abstractive model can be innovative by using words that are not from the input document in the summary. We measure the abstractive by the ratio of novel words or n-gram phrases in the summary. A higher ratio means a more abstractive model. We show the results in Figure 5. We can see that

---

[11] https://huggingface.co/hfl/chinese-bert-wwm-ext

MoverScore	**CSDS**	**MC**	**SAMSUM**
PGN	59.00/58.68/58.23	80.90/93.84/79.69	59.87
PGN-both	59.48/59.32/58.64	81.67/94.04/80.52	-
PGN-GLC	59.67/59.51/58.85	81.97/94.45/80.84	60.04
BERT	58.23/58.10/57.79	81.28/93.90/80.48	61.17
BERT-both	59.52/59.55/58.46	82.26/94.20/81.02	-
BERT-GLC	59.74/59.62/58.90	82.64/94.49/81.44	61.59
BART	60.11/59.86/58.75	82.35/94.17/81.27	62.04
BART-both	60.12/59.86/58.73	82.32/94.02/81.40	-
BART-GLC	**60.32/61.03/59.02**	**82.94/95.35/82.10**	**62.27**

Table 7: MoverScore on three datasets.

our BART+GLC is more attractive than BART and BART-both. However, all of them have a big margin compared with references. It means more research is needed for generating more abstractive summaries.

Exploiting Summarization Data to Help Text Simplification

Renliang Sun, Zhixian Yang, Xiaojun Wan
Wangxuan Institute of Computer Technology, Peking University
Center for Data Science, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
`{sunrenliang, yangzhixian}@stu.pku.edu.cn`
`wanxiaojun@pku.edu.cn`

Abstract

One of the major problems with text simplification is the lack of high-quality data. The sources of simplification datasets are limited to Wikipedia and Newsela, restricting further development of this field. In this paper, we analyzed the similarity between text summarization and text simplification and exploited summarization data to help simplify. First, we proposed an alignment algorithm to extract sentence pairs from summarization datasets. Then, we designed four attributes to characterize the degree of simplification and proposed a method to filter suitable pairs. We named these pairs Sum4Simp (S4S). Next, we conducted human evaluations to show that S4S is high-quality and compared it with a real simplification dataset. Finally, we conducted experiments to illustrate that the S4S can improve the performance of several mainstream simplification models, especially in low-resource scenarios.

1 Introduction

Text simplification and text summarization are two major techniques aiming at improving text readability (Margarido et al., 2008). The main objective of text simplification is to reduce the complexity of the text while keeping its meaning unchanged (Alva-Manchego et al., 2020; Al-Thanyyan and Azmi, 2021). Text summarization is to summarize the main idea of the document in less space (El-Kassas et al., 2021).

One of the major problems of text simplification is the lack of high-quality aligned data, which is essential for training most simplification models. Existing text simplification datasets are derived from Wikipedia (Zhang and Lapata, 2017) and Newsela (Xu et al., 2015). Researchers have proposed various alignment algorithms to extract complex-simple sentence pairs from articles (Jiang et al., 2020). However, aligning sentences from only two corpora hinders the acquisition of more simplification data, which motivates us to explore new ways to address this problem.

Text simplification usually involves the operations of keeping, deleting, reordering, etc.(Xu et al., 2016) Text summarization does not require a summary to be a simple text. Nevertheless, when we analyzed the datasets of text summarization meticulously, we noticed that there are many instances where several sentences in the original document are merged into one sentence, and complex parts are rewritten, as shown in Table 1. Then, a question arises naturally: to what extent is text summarization correlated with text simplification? Furthermore, is it feasible to extract data from text summarization to help low-resource text simplification?

Example	
document	What's Hollywood's **role in all of this? The same as it has always been** – to make money.
summary	What does Hollywood **want?** To make money, **of course.**

Table 1: The bolded parts indicate that the complex sentence in the document has been rewritten.

In this study, we investigated the above problems with a three-step procedure: (1) Extract aligned sentence pairs from summarization datasets. (2) Select sentence pairs in which the source sentences have been simplified. (3) Evaluate the quality of these sentence pairs for text simplification.

To extract aligned sentence pairs from the summarization datasets, we proposed an alignment algorithm based on the similarity between sentences. Then, we designed four attributes and a method to filter sentence pairs suitable for text simplification. We performed human evaluations and conducted experiments using mainstream simplification models on these pairs to show that they are of high quality and can help simplification.

To summarize, our contributions include: (1) We are the first to exploit summarization data to help

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 39–51
May 2-6, 2023

text simplification, verifying a new source of simplification data. (2) We proposed an alignment algorithm and a method for filtering complex-simple sentence pairs. We named them Sum4Simp (S4S). (3) We performed both empirical analysis and human evaluations on S4S to verify its quality, and the experimental results with several simplification models show the benefits of S4S for text simplification. The S4S dataset and codes are released at `https://github.com/RLSNLP/Sum4Simp`.

2 Related Work

2.1 Simplification Models

Early text simplification models are mainly based on statistic machine learning (Wubben et al., 2012; Kauchak, 2013; Narayan and Gardent, 2014). In recent years, many scholars have proposed models based on deep learning technology, such as NTS(Nisioi et al., 2017), DRESS-LS(Zhang and Lapata, 2017), EditNTS(Dong et al., 2019), ACCESS(Martin et al., 2020a), which advance the development of text simplification.

2.2 Mine Data for Simplification

The above models require a large number of aligned texts for training. Nevertheless, text simplification is a low-resource problem. Some works aim at designing unsupervised models (Qiang and Wu, 2019; Surya et al., 2019; Kumar et al., 2020; Laban et al., 2021). While other works try to mine aligned sentence pairs from more data to help train the models. Martin et al. (2020b) proposed unsupervised mining technology to create multi-language simplification corpora automatically. Lu et al. (2021) used the back-translation approach to construct a large-scale pseudo sentence simplification corpus.

2.3 Relationship with Text Summarization

For a long time, studies on text simplification and text summarization have been conducted separately. Nevertheless, there exist circumstances where complex texts not related to the main idea are removed when summarizing a document, and multiple sentences can be compressed and rewritten into a single sentence. Such a summarization can also be regarded as a simplification. Ma and Sun (2017) proposed a semantic relevance-based model to improve the results of simplification and summarization. Zaman et al. (2020) pointed out some similarities between the two tasks and defined the new task of generating simplified summaries. Up to now, none of the work has specifically analyzed the relationship between summarization and simplification. It is still worth investigating whether the data from summarization can help simplification.

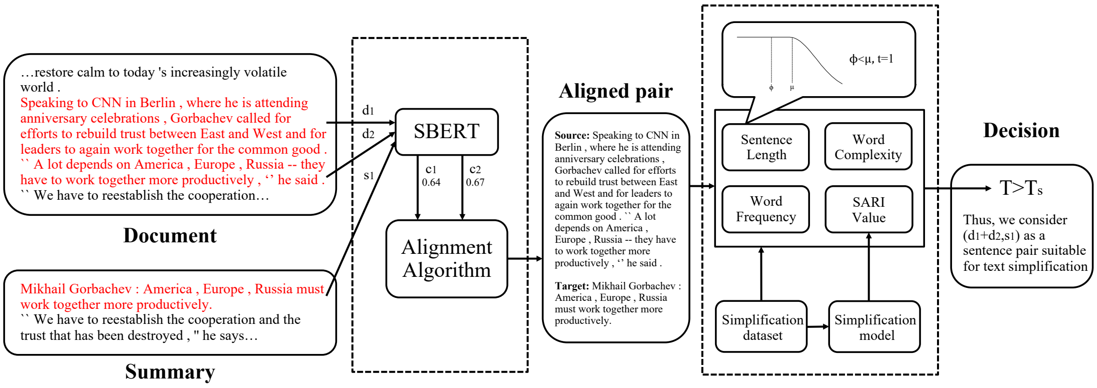
3 Mine Sentence Pairs for Simplification from Summarization Datasets

In this section, we will elaborate on how to extract sentence pairs that are suitable for text simplification from text summarization datasets. Text summarization is a document-level task while text simplification refers to a sentence-level task. Thus, we proposed an algorithm to extract aligned sentence pairs at first. Then, since not all aligned sentence pairs are suitable for text simplification, we chose four attributes and defined a set of rules to filter the appropriate sentence pairs. The whole process is shown in Figure 1.

3.1 Sentence Alignment Algorithm

Previous sentence alignment algorithms such as CATS (Štajner et al., 2018) aim at sentence compression (one complex sentence corresponds to one simple sentence) or sentence splitting (a complex sentence is split into several simple sentences). They do not satisfy the requirement to align sentence pairs from summarization datasets, where one sentence in the summary corresponds to multiple sentences in the document. Thus, we proposed an alignment algorithm to address this problem.

Assume that there are m sentences in the document and n sentences in the summary. For each sentence d_i in the document and each sentence s_j in the summary, we first compute the similarity between the two sentences. We use SBERT (Reimers and Gurevych, 2019) to achieve this. SBERT is a pre-trained model based on BERT (Devlin et al., 2019), in which the similarity of two input sentences will be calculated rapidly. Then, we define the upper threshold of similarity S_{max} and the lower threshold of similarity S_{min}. S_{max} is greater than S_{min} and they are in the range [0,1]. Assume that the maximum value of similarity between any sentence in the document and s_j is D_{max}. If D_{max} is greater than S_{max}, we consider that the sentence corresponding to D_{max} is very similar to s_j. Therefore, we keep s_j as the target sentence and the sentence corresponding to D_{max} as the source sentence, and they form an aligned sentence pair. If D_{max} is smaller than S_{min}, we consider that there is no sentence in the document that is similar to s_j.



…restore calm to today 's increasingly volatile world .
Speaking to CNN in Berlin , where he is attending anniversary celebrations , Gorbachev called for efforts to rebuild trust between East and West and for leaders to again work together for the common good .
`` A lot depends on America , Europe , Russia -- they have to work together more productively , ‘’ he said .
`` We have to reestablish the cooperation…

Mikhail Gorbachev : America , Europe , Russia must work together more productively.
`` We have to reestablish the cooperation and the trust that has been destroyed , " he says…

Source: Speaking to CNN in Berlin , where he is attending anniversary celebrations , Gorbachev called for efforts to rebuild trust between East and West and for leaders to again work together for the common good . `` A lot depends on America , Europe , Russia -- they have to work together more productively , ‘’ he said .

Target: Mikhail Gorbachev : America , Europe , Russia must work together more productively.

Thus, we consider (d1+d2,s1) as a sentence pair suitable for text simplification

Figure 1: The process of mining suitable sentence pairs from summarization datasets.

Thus, we do not keep sentence pairs related to s_j.

Algorithm 1 Sentence alignment algorithm

```
Initialization: F and C are empty sets
for $d_i$ in $d_1$,$d_2$,...,$d_n$ do
    $c_i$ = SBERT($d_i$,$s_j$)
    C.append($c_i$)
end for
if max(C)>$S_{max}$ then
    F.append(corresponding $d_i$ of max(C))
else if $S_{max}$>max(C)>$S_{min}$ then
    F.append(corresponding $d_i$ of max(C))
    C.remove(max(C))
    repeat
        $c_i$ = SBERT(stitch(F,corresponding $d_i$ of max(C)),$s_j$)
        if $c_i$>$S_{add}$ then
            F.append(corresponding $d_i$ of max(C))
            C.remove(max(C))
        end if
    until $c_i \leq S_{add}$ or len(C)$\geq L_{max}$
end if
Output: (F,$s_j$) as an aligned sentence pair
```

If D_{max} is greater than S_{min} and smaller than S_{max}, we consider this to be the case where multiple sentences in the document correspond to s_j. We temporarily save the sentences corresponding to D_{max}, and then find the sentence with the largest similarity among the remaining sentences of the document. We stitch this sentence with the sentence we just saved according to the order of the sentences in the document. We repeat this operation until the similarity between the stitched sentences and s_j is less than a threshold. We define this threshold as S_{add}, which takes values in the range [S_{min},S_{max}]. To prevent the problem of imbalance where the length of the source sentence far exceeds the length of the target sentence caused by extracting too many sentences from the document, we set L_{max}. When the number of stitched sentences reaches L_{max}, we save these stitched sentences as source sentences and s_j as the target sentence.

3.2 Four Attributes to Characterize Simplification

Aligned sentence pairs obtained from Algorithm 1 are not always complex-simple ones, and an example is given below:

Source sentence: Analysts say the Arab Spring has made Dubai a safe haven for people in the Middle East who worry about the turmoil elsewhere.

Target sentence: Analysts say the Arab Spring has made Dubai a safe haven for those who worry about the turmoil elsewhere.

This example is a real sentence pair mined from the summarization data. It is an aligned sentence pair but neither the attributive clause nor the complex words such as “turmoil” are simplified. Thus, it is not a good instance for text simplification. We design four attributes to characterize whether the source sentence is simplified or not, which are:

Sentence Length Intuitively, the longer the sentence, the more complex the sentence is likely to be. We calculate the length of the target sentence minus the average length of the source sentences.

Word Complexity We believe that the lower the average complexity of words, the simpler the sentence. We use a lexicon of word complexity created by Maddela and Xu (2018). Each word is scored by humans. The higher the score, the more complex the word. We calculate the value of the average word complexity of the target sentence minus the average word complexity of the source sentences.

Word Frequency Some words appear more frequently in complex sentences, while some words appear more frequently in simple sentences. The more frequently a word appears in a simple sentence, the more likely it is to be a simple one. We calculate the odds ratio (Monroe et al., 2008) to represent the frequency of word occurrence. For two corpus, namely i and j, their sizes are n_i and n_j, respectively. For a word w, the occurrences in corpus i and corpus j are w_i and w_j, respectively. Then, the odds ratio r of word w between corpus i and corpus j can be defined as:

$$r = \frac{w_i/w_j}{n_i/n_j} \tag{1}$$

We use the simplification dataset to construct a dictionary containing the odds ratios of the words. For example, if we want to conduct experiments on WikiLarge (Zhang and Lapata, 2017), we calculate the odds ratio of the words occurring in the WikiLarge training set. We calculate the value of the average odds ratio of the target sentence minus the average odds ratio of the source sentence.

SARI Value SARI (Xu et al., 2016) is an essential evaluation method for text simplification. It takes the original sentence, the simplified sentence, and reference sentences into consideration. The SARI value is an average of F1 scores of add and keep operation and precision of delete operation. The score for each operation is obtained by averaging n-gram scores.

$$\begin{aligned}
SARI &= \frac{1}{3}F_{add} + \frac{1}{3}F_{keep} + \frac{1}{3}P_{del} \\
P_{operation} &= \frac{1}{4}\sum_{n=1,2,3,4} p_{operation}(n) \\
R_{operation} &= \frac{1}{4}\sum_{n=1,2,3,4} r_{operation}(n) \\
F_{operation} &= \frac{2 \times P_{operation} \times R_{operation}}{P_{operation} + R_{operation}} \\
& operation \in [add, keep, del]
\end{aligned} \tag{2}$$

We consider the source sentence of the aligned sentence pairs as the original sentence and the target sentence as the simplified sentence. We need to train a simplification model at first. For example, we trained a model like ACCESS (Martin et al., 2020a) on the WikiLarge training set. Then, we input the source sentences into the simplification model and generate simplified sentences. These simplified sentences are used as reference sentences. Finally, the SARI values are calculated.

3.3 Quantify Simplicity and Filter Suitable Sentence Pairs

For each attribute, we propose a method to quantify the simplicity of a sentence. Our method is based on a hypothesis: a reference simplification dataset performs approximately normally distributed on each attribute. Simplification datasets can contain hundreds of thousands of instances, in line with the concept of large samples in statistics. Therefore, we believe this hypothesis is reasonable.

Take the sentence length attribute as an example. We first calculate the mean μ and standard deviation σ of the sentence length of the training set of a reference dataset (e.g. WikiLarge). For a random variable X, the probability density function $f(x)$ can be obtained. If the ratio of sentence length for a sentence pair is ϕ, its score t on this attribute is:

$$t = \begin{cases} 1, & \phi <= \mu \\ 2 \times (0.5 - \int_{\mu}^{\phi} f(x)dx), & \phi > \mu \end{cases} \tag{3}$$

$$t = \begin{cases} 2 \times (0.5 - \int_{\phi}^{\mu} f(x)dx), & \phi < \mu \\ 1, & \phi >= \mu \end{cases} \tag{4}$$

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{5}$$

The mathematical significance is that if $\phi <= \mu$, the simplification degree of the sentence pair is greater than the average simplification degree of the simplification dataset on this attribute. Thus, we give a score of 1 to t. If $\phi > \mu$, we subtract the proportion of sentence pairs with a ratio greater than μ and lower than ϕ that is in the simplification dataset. Then, we perform a normalization operation to obtain t. For attributes sentence length (len), word complexity (comp), and word frequency (freq), a lower ϕ indicates a greater degree of simplification. We use Equation (3) to calculate t. For attribute SARI value (sari), a higher ϕ indicates a greater degree of simplification. We use Equation (4) to calculate t.

To make a final decision, the scores on each attribute are weighted with α and summed to obtain T for a sentence pair, indicating the extent of simplification of the source sentence. We set a threshold value $\mathrm{T_s}$ to control the extent of simplification. When $\mathrm{T} > \mathrm{T_s}$, we consider the sentence pair suitable for the task of text simplification.

$$\begin{gathered}
\mathrm{T} = \sum_{i \in Attr} \alpha_i t_i \\
Attr = [len, comp, freq, sari]
\end{gathered} \tag{6}$$

We exploit and filter sentence pairs from the CNN/Daily Mail summarization dataset (Nallapati

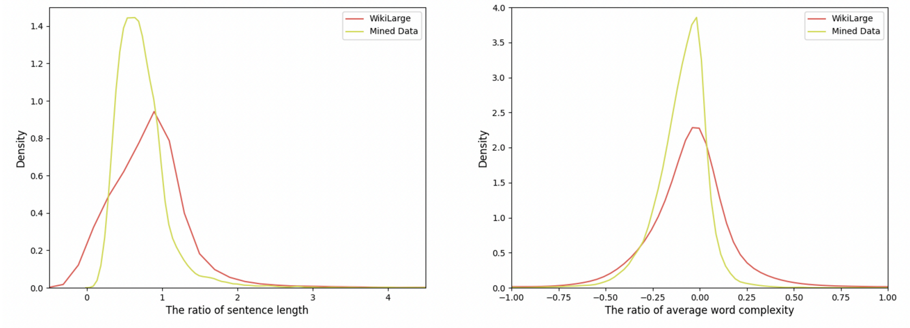


Figure 2: Distributions of the ratio of sentence length and average word complexity. We smoothed the results by using a Gaussian kernel. Sentences from S4S are more compressed than in WikiLarge. Sentences where the words become more complex are also less than in WikiLarge.

et al., 2016), which contains more than 300,000 documents and corresponding summaries from news stories in CNN and Daily Mail. We name these mined sentence pairs Sum4Simp (S4S).

4 Quantitative Analysis

In this section, we want to show that Sum4Simp (S4S) is high-quality. We conducted two human evaluations and performed statistics on S4S, comparing it with real simplification datasets.

4.1 Human Evaluations

First, we want to evaluate the alignment quality of the sentence pairs obtained in Section 3.1. Following Hwang et al. (2015), we defined the quality of alignment into four classes: Good, Good partial, Partial, and Bad. Due to the space limit, details and examples are demonstrated in Table 10.

We randomly selected sentence pairs from the aligned pairs obtained by our proposed alignment algorithm. Then, we designed a baseline that does not use our proposed alignment algorithm. When the similarity calculated by SBERT between a sentence in the document and a sentence in the summary is greater than 0.6, we kept this sentence in the document. As we introduced in Section 3.1, the CATS method (Štajner et al., 2018) may not be suitable for aligning sentence pairs from summarization datasets. However, we used it as a baseline.

We used the two baseline methods described above to obtain aligned sentence pairs from summarization datasets. What's more, we randomly selected sentence pairs from a simplification dataset named WikiLarge (Zhang and Lapata, 2017) for comparison. The results are shown in Figure 3.

We considered **Good** and **Good partial** to be acceptable quality. The sentence pairs obtained by our proposed alignment algorithm have the highest percentage in these two levels. While WikiLarge has the most sentence pairs with a Good level, it also has the most sentence pairs with a Bad level. Xu et al. (2015) pointed out that data mined from Wikipedia is not always of high quality.

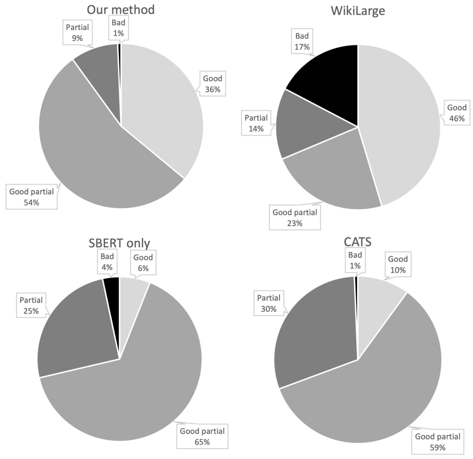


Figure 3: Human evaluation results of data obtained by three alignment methods and WikiLarge. We randomly selected 50 sentence pairs from each source of data. Then, we hired three workers to evaluate the 200 sentence pairs individually.

Then, we want to show that the final sentence pairs obtained in Section 3.3 are more suitable for simplification. We randomly selected 50 sentence pairs that are only aligned and 50 sentence pairs from S4S. We also randomly selected 50 sentence pairs from WikiLarge for comparison.

Following Dong et al. (2019), we used two indicators as the criteria: (1) **Simplicity**: Is the target sentence simpler than the source sentence? (2) **Adequacy**: Are the source sentence and target sentence fluent and grammatically correct? Another indicator, Meaning, can be regarded as the evaluation of alignment quality, so we did not repeat it. The results are shown in Table 2. The sentence pairs from S4S receive the highest Simplicity score, significantly higher than the aligned-only pairs and WikiLarge, indicating the effectiveness of the proposed filtering method.

	Simplicity↑	Adequacy↑
WikiLarge	3.11**	4.6**
Aligned only	3.2**	4.81
S4S	**3.49**	**4.94**

Table 2: Human evaluation results of data obtained by two methods and WikiLarge. We hired three workers to evaluate individually. Student t-tests were performed and results significantly different from S4S were marked with **(p<0.01).

4.2 Statistics and Comparison

We used three dimensions, sentence length, average word complexity, and odds ratio of cue words, to compare the sentence pairs from S4S with those from WikiLarge. The ratio of sentence length is calculated by dividing the length of the simplified sentence by the length of the original sentence. The ratio of average word complexity is calculated by subtracting the average word complexity of the original sentence from the average word complexity of the simplified sentence.

We randomly selected 10,000 sentence pairs from WikiLarge and S4S, respectively. From Figure 2, in S4S, the number of sentence pairs with a length ratio greater than one has been significantly decreased compared to WikiLarge, indicating that sentences are more compressed. What's more, the vast majority of the ratios of average word complexity are less than zero, suggesting a general simplification at the word level in S4S.

Sentence splitting, a common operation in text simplification, can be represented by the odds ratio of conjunctions and cue words (Siddharthan, 2003). The definition of the odds ratio is detailed in Equation (1). When the odds ratio of conjunctions is much less than 1, and the odds ratio of cue words is much greater than 1, a complete degree of simplification is involved. Following Xu et al. (2015) and Sun et al. (2021), we calculated the odds ratio of conjunctions and cue words in WikiLarge and S4S, as shown in Table 3.

WikiLarge		S4S	
cue words	odds ratio↑	cue words	odds ratio↑
also	1.15	also	1.13
then	1.16	then	**1.21**
still	1.01	still	**1.41**

Wikilarge		S4S	
conjunctions	odds ratio↓	conjunctions	odds ratio↓
and	0.87	and	0.95
as	0.72	as	0.80
since	1.01	since	**0.96**
because	2.59	because	**1.05**
when	1.32	when	**1.09**
if	1.30	if	1.38
but	1.18	but	**1.11**
though	0.71	though	**0.62**
although	0.46	although	**0.40**

Table 3: The odds ratio of cue words and conjunctions. The bolded parts indicate that S4S performs better than WikiLarge. Some words, such as "hence", occur too infrequently to be statistically meaningful.

5 Experimental Setup

5.1 Datasets

We used two commonly used simplification datasets, **WikiLarge** (Zhang and Lapata, 2017) and **WikiSmall** (Zhu et al., 2010), to demonstrate the usefulness of the sentence pairs mined from summarization data. The training set of WikiLarge contains more than 296k sentence pairs, which is larger than that of WikiSmall containing 88k sentence pairs. We used **Turkcorpus** (Xu et al., 2016) as the validation and the test set for WikiLarge. Each of the 2000 validation instances and the 359 test instances has 8 reference sentences. We used the original validation set and test set for WikiSmall, with 205 validation instances and 100 test instances.

5.2 Evaluation Metrics and Models

We took **SARI** (Xu et al., 2016) and **BERTScore** (Zhang et al., 2019) as the evaluation metric in this paper. SARI is the most popular automatic evaluation metric for text simplification. The SARI value is obtained by averaging the F_{keep}, P_{delete}, and F_{add} score. We used the **EASSE** package (Alva-Manchego et al., 2019) to get SARI values. A recent study recommends using BERTScore$_{precision}$ to evaluate the quality of the system outputs prior to using SARI to measure simplification (Alva-Manchego et al., 2021). FKGL (Kincaid et al., 1975) was used to measure text readability but was proven to be inappropriate for evaluating text simplification recently (Tanprasert and Kauchak,

Models	WikiLarge				S4S				WikiLarge+OA				WikiLarge+S4S			
	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}
Transformer	36.95*	70.80	36.91	3.15	34.43**	58.54	43.68	1.08	36.75*	70.79	36.38	3.06	**37.85**	71.11	39.15	3.27
BART	37.99**	72.53	37.85	3.59	36.21**	64.70	42.60	1.34	37.71**	73.02	36.81	3.31	**39.20**	70.99	42.31	4.30
ACCESS	39.67*	71.20	42.69	5.12	36.20**	65.62	41.53	1.44	39.46*	69.39	43.96	5.03	**40.71**	71.26	44.06	6.81

Models	WikiSmall				S4S				WikiSmall+OA				WikiSmall+S4S			
	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}
Transformer	36.35*	66.69	40.53	1.82	36.75	60.23	49.49	0.53	36.38*	64.46	40.54	4.15	**38.57**	66.56	43.69	5.46
BART	35.13*	64.94	35.86	4.59	34.13*	61.06	39.95	1.39	34.65*	67.09	31.92	4.93	**36.58**	67.39	37.14	5.22
ACCESS	35.35*	65.01	38.50	2.53	34.63**	51.07	51.76	1.05	35.67*	60.95	44.29	1.77	**38.28**	58.45	53.64	2.73

Table 4: Results of three simplification models trained on four different training sets. The test sets in the upper and lower tables are Turkcorpus and WikiSmall, respectively. "+" represents the operation to mix the two datasets and sort them randomly. OA is a set of sentence pairs with a similar size to S4S drawn from aligned but not filtered sentence pairs. The bolded part indicates the training set that achieves the best result for each model. Student t-tests were performed, and SARI values that were significantly different from WikiLarge+S4S and WikiSmall+S4S were marked with *($p<0.05$) or **($p<0.01$).

2021). BLEU (Papineni et al., 2002) has been proven to be unsuitable for evaluating text simplification (Sulem et al., 2018). Therefore, we did not report FKGL values and BLEU values.

We selected three representative models - **Transformer** (Vaswani et al., 2017), **BART** (Lewis et al., 2020), and **ACCESS** (Martin et al., 2020a) to conduct experiments. Transformer and BART perform strongly for many generation tasks. ACCESS is a simplification model proposed recently and it uses explicit tokens related to different attributes to control the process of simplification.

5.3 Training Details

We used the Huggingface Transformers (Wolf et al., 2020) to implement the Transformer model and the BART model. We used the original code to implement the ACCESS model. We used four Nvidia A40 GPUs for training. We reported the results of the model on the test set which has the best SARI value on the validation set.

More details can be found in Appendix A.

6 Experimental Results

6.1 Results on Existing Test Sets

We designed four types of training sets and tested the three simplification models on existing test sets. We first measured the outputs of each model using $\text{BERTScore}_{precision}$ and found that the values are very close to 1, indicating that the outputs are of high quality. Then, the SARI values are shown in Table 4.

From the upper table, Sum4Simp (S4S) mixed with the WikiLarge training set improves the performance of all three simplification models on Turkcorpus. To be more specific, in terms of the SARI metric, ACCESS is improved by 1.04 points, BART is improved by 1.21 points, and Transformer is improved by 0.90 points. We have used the original codes and followed the original hyper-parameter settings, but the SARI value of the ACCESS model trained on WikiLarge is lower than the results reported by Martin et al. (2020a). We think this is because we lowered the training data and used the NLTK package to split the words. Meanwhile, seen from the lower table, S4S mixed with the WikiSmall training set also improves the performance of all three models on the test set of WikiSmall. The improvement on the WikiSmall test set is more significant than that on the Turkcorpus test set. In terms of the SARI metric, ACCESS is improved by 2.93 points, BART is improved by 1.45 points, and Transformer is improved by 2.22 points. Example outputs are given in Table 11. It may seem strange that the SARI value of Transformer is higher than that of BART. However, we noticed that the SARI value of BART is approximately 3 points higher than that of Transformer on the validation set, making the experimental results remain convincing.

The size of the training set of WikiLarge is much larger than that of WikiSmall. Therefore, the models were more fully trained on WikiLarge. While the size of the training set of WikiSmall is comparatively smaller, S4S helps the model learn to simplify sentences better and results in a more significant improvement.

OA was designed to verify that the improvement of the results comes from high-quality mined sentence pairs rather than mere data expansion. Compared with the original training set, the per-

Models	S4S				WikiLarge				S4S+WikiLarge			
	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}	SARI↑	F_{keep}	P_{delete}	F_{add}
Transformer	**44.75**	53.32	74.72	6.19	32.59	45.38	51.78	0.61	43.61	52.24	73.91	4.68
BART	46.42	57.20	76.62	5.43	32.98	47.12	50.10	1.70	**46.51**	57.24	73.91	4.68
ACCESS	**40.19**	45.85	72.82	1.88	30.10	44.30	43.99	2.01	38.45	43.35	70.71	1.30

Table 5: Results on three simplification models trained on three different training sets. The valid and test sets come from S4S.

formances on WikiLarge+OA and WikiSmall+OA were not improved and even dropped for the model like BART. The results illustrate that the method for filtering suitable sentence pairs for simplification purposes is essential.

If we only used S4S as the training set, the SARI values obtained are 2.5 points lower than the model trained with WikiLarge and 0.5 points lower than the model trained with WikiSmall on average. We believe the performance gap is due to domain differences: S4S comes from news stories written by professional journalists, while WikiLarge and WikiSmall come from Wikipedia. Overall, though S4S comes from a different domain, it can still be beneficial to the existing simplification datasets.

6.2 Results on S4S Test Set

In this subsection, we treat S4S as a standard simplification dataset that contains more than 243K sentence pairs. We divided the train/dev/test set as 240k/2k/1k, respectively. We would like to see the performance of simplification models on the S4S dataset and we want to know if the WikiLarge dataset from a different domain can improve the performance. We designed three types of training sets. Then, we conducted experiments with each of them to train the three simplification models.

According to Table 5, all three simplification models trained on the S4S dataset have significantly higher SARI values compared to the results in Table 4. When we mixed the training set of S4S and WikiLarge, the SARI values dropped by 1 point on average compared to using the S4S training set alone. Besides, when we only used the WikiLarge training set, the SARI values dropped by an average of more than 10 points. We also gave example outputs in Table 12. Above all, we believe the quality of the S4S dataset is higher than that of the Wikipedia-based datasets. The S4S dataset was given in the supplementary materials.

6.3 Results on Extremely Low-resource Scenarios

In many cases simplification data is hard to obtain (Aprosio et al., 2019; Maruyama and Yamamoto, 2019), and we took a small amount of sentence pairs from the training set of WikiLarge to simulate an extremely low-resource situation. We reduced the size of the WikiLarge training set to 50%, 20%, 10%, 5%, and 1%, respectively. We then conducted experiments using the ACCESS model trained on the size-reduced WikiLarge data and the mixture of size-reduced WikiLarge and S4S. The results are shown in Figure 4.

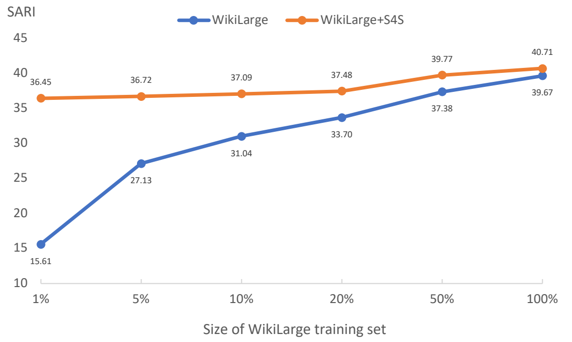


Figure 4: Experimental results of extremely low-resource experiments on Turkcorpus test set.

When the size of the training set is relatively small (less than 20%, about 60,000 sentence pairs), S4S can improve the results significantly. The results prove that the S4S is effective in helping text simplification when data is difficult to obtain.

6.4 Ablation Study

In our proposed sentence filtering method, we used four attributes to control the simplicity of the sentence pairs extracted from summarization datasets. We removed the attributes one by one and then used the remaining three attributes as new rules to filter simple sentence pairs. We set T_s to 2.75 in the experiment. The filtered sentence pairs are mixed with the WikiLarge training set and then used to train the ACCESS model.

Experiment	SARI↑
WikiLarge+S4S	40.71
WikiLarge	39.67
Without word complexity	39.32(-1.39)
Without sentence length	39.63(-1.08)
Without word frequency	37.70(**-3.01**)
Without SARI value	38.78(-1.93)

Table 6: Ablation study on Turkcorpus test set.

The results are illustrated in Table 6. In this experiment, the odds ratio attribute has the greatest effect on the results. When this attribute is missing, the SARI value decreases by 3.01 points. The sentence length attribute has the least effect on the results. When this attribute is missing, the SARI value drops by 1.08 points. The results also show that the four attributes of our design are meaningful. They all play a significant role in filtering the simplified sentence pairs.

7 Conclusion

In this paper, we are committed to mining data from text summarization datasets to help text simplification. We proposed an alignment algorithm and a new method to filter suitable sentence pairs. We named these pairs Sum4Simp (S4S). We conducted human evaluations on S4S and performed experiments on mainstream simplification models to illustrate that the S4S is high-quality and can help text simplification. In future work, we will apply our method to mine more simplification data from other summarization datasets.

Acknowledgements

This work was supported by National Key R&D Program of China (2021YFF0901502), National Science Foundation of China (No. 62161160339), State Key Laboratory of Media Convergence Production Technology and Systems and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

Limitations

We considered the consumption of computational resources as the major limitation of our method. To extract aligned sentence pairs from summarization datasets, we need to calculate the similarity between each sentence in the summary and each sentence in the document, which makes the time complexity of the alignment algorithm be $O(n^2)$. We ran the alignment algorithm with an Intel Xeon processor. On average, there are 40 sentences in a document and 4 sentences in a summary. There are 312K documents in total with corresponding summaries. The total running time is 42,153s. We have released the aligned sentence pairs to help future research.

Second, to calculate the SARI values in Section 3.2, we need to train a simplification model in advance, which can consume GPU resources. For example, if we train a BART model on the WikiLarge dataset and set the max epochs to 10, the training time spent on an Nvidia A40 is about 3 hours.

References

Suha S Al-Thanyyan and Aqil M Azmi. 2021. Automated text simplification: A survey. *ACM Computing Surveys (CSUR)*, 54(2):1–36.

Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. 2019. Easse: Easier automatic sentence simplification evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 49–54.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2020. Data-driven sentence simplification: Survey and benchmark. *Computational Linguistics*, 46(1):135–187.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2021. The (un) suitability of automatic evaluation metrics for text simplification. *Computational Linguistics*, 47(4):861–889.

Alessio Palmero Aprosio, Sara Tonelli, Marco Turchi, Matteo Negri, and Mattia A Di Gangi. 2019. Neural text simplification in low-resource conditions using weak supervision. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 37–44.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. Editnts: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402.

Wafaa S El-Kassas, Cherif R Salama, Ahmed A Rafea, and Hoda K Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679.

William Hwang, Hannaneh Hajishirzi, Mari Ostendorf, and Wei Wu. 2015. Aligning sentences from standard wikipedia to simple wikipedia. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 211–217.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural crf model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960.

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 1537–1546.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. 2020. Iterative edit-based unsupervised sentence simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7918–7928.

Philippe Laban, Tobias Schnabel, Paul Bennett, and Marti A Hearst. 2021. Keep it simple: Unsupervised simplification of multi-paragraph text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6365–6378.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Xinyu Lu, Jipeng Qiang, Yun Li, Yunhao Yuan, and Yi Zhu. 2021. An unsupervised method for building sentence simplification corpora in multiple languages. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 227–237.

Shuming Ma and Xu Sun. 2017. A semantic relevance based neural network for text summarization and text simplification. *arXiv preprint arXiv:1710.02318*.

Mounica Maddela and Wei Xu. 2018. A word-complexity lexicon and a neural readability ranking model for lexical simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3749–3760.

Paulo RA Margarido, Thiago AS Pardo, Gabriel M Antonio, Vinícius B Fuentes, Rachel Aires, Sandra M Aluísio, and Renata PM Fortes. 2008. Automatic summarization for text simplification: Evaluating text understanding by poor readers. In *Companion Proceedings of the XIV Brazilian Symposium on Multimedia and the Web*, pages 310–315.

Louis Martin, Éric Villemonte De La Clergerie, Benoît Sagot, and Antoine Bordes. 2020a. Controllable sentence simplification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4689–4698.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. 2020b. Multilingual unsupervised sentence simplification. *arXiv preprint arXiv:2005.00352*.

Takumi Maruyama and Kazuhide Yamamoto. 2019. Extremely low resource text simplification with pre-trained transformer language model. In *2019 International Conference on Asian Language Processing (IALP)*, pages 53–58. IEEE.

Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin'words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16(4):372–403.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *The 52nd annual meeting of the association for computational linguistics*, pages 435–445.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 85–91.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jipeng Qiang and Xindong Wu. 2019. Unsupervised statistical text simplification. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1802–1806.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Advaith Siddharthan. 2003. Preserving discourse structure when simplifying text. In *Proceedings of the 9th European Workshop on Natural Language Generation (ENLG-2003) at EACL 2003*.

Sanja Štajner, Marc Franco-Salvador, Paolo Rosso, and Simone Paolo Ponzetto. 2018. Cats: A tool for customized alignment of text simplification corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018. Bleu is not suitable for the evaluation of text simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 738–744.

Renliang Sun, Hanqi Jin, and Xiaojun Wan. 2021. Document-level text simplification: Dataset, criteria and baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7997–8013.

Sai Surya, Abhijit Mishra, Anirban Laha, Parag Jain, and Karthik Sankaranarayanan. 2019. Unsupervised neural text simplification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2058–2068.

Teerapaun Tanprasert and David Kauchak. 2021. Flesch-kincaid is not a text simplification evaluation metric. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 1–14.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1015–1024.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Farooq Zaman, Matthew Shardlow, Saeed-Ul Hassan, Naif Radi Aljohani, and Raheel Nawaz. 2020. Htss: A novel hybrid text summarisation and simplification architecture. *Information Processing & Management*, 57(6):102351.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1353–1361.

A More Details

In Algorithm 1, for S_{max}, S_{add}, and S_{min}, we first observed the alignment results to obtain a rough range [0.5,0.8]. In this range, we set the step size to 0.1 and then chose four combinations of parameters: (0.8, 0.7, 0.6), (0.8, 0.7, 0.5), (0.8, 0.6, 0.5), and (0.7, 0.6, 0.5). We used human evaluation on 50 sentence pairs for each combination to determine which combination is the best. Finally, we set S_{max} to 0.8, S_{min} to 0.6, and S_{add} to 0.7. Lmax is set to 3 as an empirical value. If it is too large, the model will be more concerned with deletion than simplification; if it is too small, the information in the original sentences will lose.

For the method of filtering suitable sentence pairs in Section 3.3, we set α_i to 0.25 because it is difficult to prove that one of the four attributes is more important than the other. We performed a parameter research for $\mathrm{T_s}$ from 3.5 to 3.8 with a step size of 0.05.

We have released the aligned sentence pairs obtained in Section 3.1 for future research. So future researchers only need to set $\mathrm{T_s}$ when conducting experiments.

To obtain Table 4, we first trained models with existing simplification datasets (e.g., train ACCESS with WikiLarge). Then, we selected the model that performed best on the validation set to calculate the score t for the SARI value attribute mentioned in Section 3.2. In this way, we got S4S. We then trained models with WikiLarge+S4S to obtain the results in the fourth column of the Table 4. The S4S dataset in Section 6.2 is obtained after we first trained ACCESS with WikiLarge. We will also release this version of S4S as a standard simplification dataset.

Parameter	**Value**	**Parameter**	**Value**
epochs	30	max source length	256
batchsize	64	max target length	256
optimizer	Adam	dropout	0.1
learning rate	5e-5	d_{model}	768
warm up steps	2000	attention heads	12

Table 7: Parameters of the Transformer model.

Parameter	**Value**	**Parameter**	**Value**
epochs	10	max source length	256
batchsize	64	max target length	256
optimizer	Adam	dropout	0.1
learning rate	5e-5	d_{model}	768
warm up steps	2000	attention heads	12

Table 8: Parameters of the BART model.

Parameter	**Value**	**Parameter**	**Value**
max epochs	100	label smoothing	0.54
max tokens	5000	clip norm	0.1
optimizer	Adam	dropout	0.2
learning rate	1.1e-4	weight decay	1e-4
warm up steps	1000	attention heads	8

Table 9: Parameters of the ACCESS model.

B Definition of Alignment Quality

C Example Outputs

Good	The semantics of the source sentence and the target sentence completely match, possibly with small omissions.
Source	Sets in children 's bedrooms or left on as background noise could be particularly damaging.
Target	Devices in bedrooms or left on as background noise is more damaging.
Good partial	Source and target sentence mean basically the same thing. However, source or target sentence may contain additional information that is not contained in the other sentence.
Source	The tape was played at a hearing Monday to determine whether or not the confession can be used as evidence at Hernandez 's murder trial - not whether the statements are true. Judge Maxwell Wiley must decide whether Hernandez was properly advised of his rights.
Target	The judge must decide not whether the confession is true, but whether it can be permitted to be used as evidence at Hernandez 's murder trial.
Partial	Source and target sentence are discussing two unrelated concepts, but share short related phrases that do not match considerably.
Source	A non-profit group called Women On 20s, formed to convince President Barack Obama to put a woman's image on the $20 note, already has done some polling.
Target	There is a group called Women On 20s.
Bad	Source and target sentence are discussing two unrelated concepts.
Source	Leicester City have lost just one of their last seven league meetings with Hull City.
Target	88 % of British grandmothers consider themselves to be a Glam-Ma.

Table 10: Definition of the alignment quality. Example of each level of quality is also given.

Complex(input)	in computing , a protocol is a set of rules which is used by computers to communicate with each other across a network .
Simple(reference)	in computing , a protocol is the language used by computers while talking with each other .
WikiSmall	in computing , a protocol is a set of rules which is used by computers to communicate with each other across a network .
S4S	the process is a set of rules which is used by computers to communicate with each other across a network
WikiSmall+OA	in computing , a protocol is a set of rules which is used by computers to provide with each other across a network .
WikiSmall+S4S	in computing , a protocol is used by computers to communicate with each other across a network .

Table 11: An example of sentences generated by ACCESS. When the training set is WikiSmall, the complex sentence is not simplified. When the training set is S4S or WikiSmall+OA, the generated sentences contain grammatical errors and change the meaning of the complex sentence. The sentence generated by ACCESS trained on WikiSmall+S4S can be regarded as a simplified sentence.

Complex(input)	barcelona manager luis enrique -lrb- pictured -rrb- insisted afterwards he was right to start uruguay striker suarez
Simple(reference)	barcelona boss luis enrique says he was right to start the uruguay player
S4S	barcelona boss luis enrique says he was right to start uruguay striker
WikiLarge	barcelona manager luis enrique -lrb- pictured - pictured he wanted to start uruguay striker suarez .
S4S+WikiLarge	barcelona manager luis enrique said he was right to start right to start uruguay suarez suarez

Table 12: An example of sentences generated by ACCESS when S4S is regarded as a standard simplification dataset. When the training set is WikiLarge, the generated sentence contains grammatical errors and changes the meaning of the complex sentence. When the training set is S4S+WikiLarge, the generated sentence also contains grammatical errors and is less simple than the generated sentence when the training set is S4S only. This example illustrates that the quality of S4S is higher than that of WikiLarge.

Shironaam: Bengali News Headline Generation using Auxiliary Information

Abu Ubaida Akash[*][§] **Mir Tafseer Nayeem**[*][◊] **Faisal Tareque Shohan**[§] **Tanvir Islam**[‡]
[§]Ahsanullah University of Science and Technology [◊]University of Alberta
[‡]University of Hawaii at Manoa
`akash.ubaida@gmail.com`, `mnayeem@ualberta.ca`
`faisaltareque@hotmail.com`, `tislam@hawaii.edu`

Abstract

Automatic headline generation systems have the potential to assist editors in finding interesting headlines to attract visitors or readers. However, the performance of headline generation systems remains challenging due to the unavailability of sufficient parallel data for low-resource languages like Bengali and the lack of ideal approaches to develop a system for headline generation using pre-trained language models, especially for long news articles. To address these challenges, we present `Shironaam`, a large-scale dataset in Bengali containing over 240K news article-headline pairings with auxiliary data such as image captions, topic words, and category information. Unlike other headline generation models, this paper uses this auxiliary information to better model this task. Furthermore, we utilize the contextualized language models to design encoder-decoder model for Bengali news headline generation and follow a simple yet cost-effective coarse-to-fine approach using topic-words to retrieve important sentences considering the fixed length requirement of the pre-trained language models. Finally, we conduct extensive experiments on our dataset containing news articles of 13 different categories to demonstrate the effectiveness of incorporating auxiliary information and evaluate our system on a wide range of metrics. The experimental results demonstrate that our methods bring significant improvements (*i.e.,* 3 to 10 percentage points across all evaluation metrics) over the baselines[1]. Also to illustrate the utility and robustness, we report experimental results in few-shot and non-few-shot settings.

1 Introduction

News headlines can significantly affect the number of visitors and play a crucial part in the life-cycle of a news article (Murao et al., 2019). Therefore, representative and interesting headlines are arguably essential to any news document to grab the attention of potential readers (Mishra et al., 2021; Ao et al., 2021). Nowadays, online and printed news releases significantly increase the article's visibility, support, and context by using multimedia content. As a picture is worth a thousand words, digital assets such as images and videos are the go-to candidates for the thumbnails used in different social media, blogs, and many other platforms. The captions that go with the images or videos are equally significant as the actual content. Captions describing the images can clarify and enhance the image, optimize news articles for search engines, and improve the accessibility of the news for people with vision impairments[2].

Headline generation, given a news article, is a special case of abstractive summarization (Yamada et al., 2021), which involves sentence compression, syntactic reorganization, sentence fusion, and lexical paraphrasing (See et al., 2017; Gehrmann et al., 2018; Zhong et al., 2019; Nayeem et al., 2019; Nayeem and Chali, 2017b). Unlike text summaries, which often feature many or single long sentences to summarize a document's important concepts (Nayeem and Chali, 2017a), news headlines frequently have a single short catchy statement to grab the readers attention and entice them to read the story. Even though Bengali is the seventh most spoken language with approximately 337 million speakers worldwide[3] (Chakraborty et al., 2021; Chowdhury et al., 2021), generating quality headlines for a low-resource language such as Bengali is more challenging due to the unavailability of large-scale human-annotated dataset (Haque et al., 2016; Nayeem et al., 2018; Joshi et al., 2019).

---

[*] Equal contribution.

[1] Code, dataset, and model checkpoints: `https://github.com/dialect-ai/BenHeadGen`

[2] In this paper, we limit our focus to only captions to improve the news headlines. Using multimodal information for this task is left as possible future work.

[3] `https://w.wiki/57`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 52–67
May 2-6, 2023

Contextualized language models such as `BERT` (Devlin et al., 2019), `RoBERTa` (Liu et al., 1907), `T5` (Raffel et al., 2020) help improving several downstream tasks in NLP, such as summarization, question answering, and text classification. Unfortunately, these models suffer from a limitation as they can handle input sequences up to a certain limit (Sun et al., 2019). As a result, this limitation burdens some NLP tasks, especially where the input is necessarily long (Kitaev et al., 2020), such as transcript analysis of the phone calls, document topic prediction, news headline generation, etc. The most natural way to address this problem is to trim the input sequences to a maximum length. However, trimming the long input document is tricky, especially for headline generation. The news articles usually maintain coherence and relevant parts may be located at the bottom of the document, which may prevent models from generalizing well to positions beyond the cutoff point[4]. In this paper, we utilize topic words to retrieve important sentences as a context for the BERT model by following a simple yet cost-effective coarse-to-fine approach.

We present **Shironaam**, a large-scale abstractive Bengali news article dataset that includes over 240K professionally annotated headline-article pairings as well as auxiliary information such as image captions, topic words, and category information. Each sample can be represented as a tuple of (article, image caption, topic-words, category, headline). To the best of our knowledge, Shironaam is the first Bengali news article dataset incorporating auxiliary information and a benchmark for the news headline generation task. This corpus has the potential to authorize and encourage research on such a low-resource language, bringing technological advancements to a previously underserved community. Rather than the one-to-one mapping (*i.e.,* input is an article, and output is a headline) used in the earlier works (Takase et al., 2016; Zhang et al., 2018; Murao et al., 2019; Colmenares et al., 2019; Song et al., 2020; Li et al., 2021), we treat the headline generation task as a **three-to-one** mapping with the inputs being an image caption, a list of topic words, and an article where the output is a headline. Based on the transformer architecture, we utilize pre-trained language models for generating headlines and present a new concept of fusing image caption parallelly (Liu et al., 2020a) with the input article to support the three-to-one mapping and to encode long documents. We design and compare numerous input mechanism alternatives as part of the suggested strategy. Extensive experiments on our proposed dataset reveal that the suggested method is capable of generating high-quality news headlines (see Section F in the Appendix) and brings significant improvements over the state-of-the-art baselines across all evaluation metrics (Table 5).

Our main contributions can be summarized as follows:

- We provide **Shironaam**, a large-scale news headline generation dataset of a low-resource language *i.e.,* Bengali containing over 240K news headline-article pairings with auxiliary information such as image captions, topic words, and category information (Table 2). Also, this dataset can potentially be used for other tasks such as document categorization, news clustering, keyword identification, etc.
- We present a new concept of incorporating auxiliary information to model input with articles to improve the quality of headlines. We train an encoder-decoder model for this task almost from scratch, which utilizes pre-trained language model (Figure 1).
- We develop **BenSim**, an independent module for measuring the semantic similarity among Bengali sentences. We make use of the BenSim module and utilize topic words to encode long articles by following a simple yet effective approach (Figure 1(c)).
- To illustrate the utility and robustness, we also evaluate the performance with few-shot settings where the domains don't have enough training samples (Table 6).

2 The Shironaam Corpus

In this section, we present the first-of-its-kind corpus (we name it `Shironaam`) for news headline generation in Bengali like low-resource language. This includes auxiliary information in addition to the usual headline-article pairs. We explain the curation process involving raw data crawling, preprocessing, and cleaning.

---

[4] While `Longformer` (Beltagy et al., 2020) is a viable solution for this problem, it comes up with a high computational cost, and pre-trained models aren't available.

[5] `https://en.wikipedia.org/wiki/Jaccard_index`

Category	Train	Valid	Test	Total	Jaccard (%)
Entertainment	16,104	365	1095	17,565	13.56
National	117,566	2,664	7,994	128,226	24.60
Nature	467	10	31	510	23.66
International	30,558	692	2,078	33,329	18.09
Sports	17,635	399	1,199	19,235	17.82
Economy	6,447	146	438	7,032	39.37
Life-Health	6,356	144	432	6,933	17.83
Miscellaneous	1,599	36	108	1,744	11.71
Opinion	3,501	79	238	3,819	38.41
Politics	15,018	340	1,021	16,380	23.02
Edu-Career	4,008	90	272	4,372	53.58
Science-Tech	1,046	23	71	1,141	22.95
Religion	269	6	18	294	71.59
Total/Avg.	**220,574**	**4,994**	**15,012**	**240,580**	**28.94**

Table 1: Our headline generation dataset (**Shironaam**) distribution over **13** different domains. Jaccard scores[5] represent the similarities of each domain in between the image captions and headlines.

Features	IndicNLG-BN	Shironaam (*ours*)
Article	✓	✓
Headline	✓	✓
Category	✗	✓
Topic words	✗	✓
Image Caption	✗	✓
#Examples	142,731	240,580

Table 2: Feature-level comparison between IndicNLG-BN (2022) and **Shironaam** dataset (*ours*).

2.1 Raw Data Crawling

We crawl around 900,000 raw data samples from seven famous Bengali newspapers (names in Section C in the Appendix) concentrating on certain criteria, such as headline, article, image caption, category, and topic words. Since each of the newspapers mentioned above has it's own professional authors and distinct writing style, we consider multiple sources to prevent the bias of a particular annotation style. To ensure content diversity, we also cover various domains from all the news dailies. The majority of the news samples are extracted from HTML bodies of the corresponding publications, while some are rendered using JavaScript. However, two of them (see in Appendix Section C) do not provide the archives on their websites; therefore, we collect the samples through their APIs.

2.2 Dataset Preprocessing

The overall crawled corpus contains a lot of noise, such as irrelevant details about the publisher and the date/time of the news in multiple formats, embedded advertisements, phrases from different languages (especially English), reference URLs, inconsistent bold sections, emoticons, extrinsic symbols, and various Unicode representations. Thus, we remove the date/time and the embedded items using regular expressions. To preserve only the Bengali texts, we construct a vocabulary of Bengali unit characters and perform character level matching in the article bodies and headlines. The English numbers, however, are retained since they are used frequently in regular Bengali texts.

Dataset	% of novel n-gram			
	unigram	bigram	trigram	4-gram
Shironaam	26.59	66.12	82.71	86.49
IndicNLG BN	46.38	78.92	90.39	94.77

Table 3: Percentage (%) of novel n-grams between IndicNLG-BN (2022) and **Shironaam** dataset (*ours*).

The image captions sometimes include extra/irrelevant information (*e.g.,* ছবি [Picture], সংগৃহীত [Collected], ফাইল ছবি [File Image], রয়টার্স [Reuters], ইন্টারনেট [Internet], প্রতীকী ছবি [Symbolic Image], *etc.*)[6], which are common in any news article. Thus, we identify these repetitive words using a simple frequency-based approach over all the samples and remove them from the image captions. Furthermore, we discard the samples whose captions are smaller than four words in length; from our manual inspection, we observed that these words often describe the named entities present in the image, such as name, location, date/time, etc.

Different newspapers use different names to categorize their contents. Consequently, each domain is represented with different category names in all the news dailies. For extracting the categories, we map them with their corresponding representative domains and label each domain with it's relevant names. For instance, national, whole-country, city-news, country, capital, city-roundup, south-city, *etc.* are distinct categorical terms, but they can be grouped easily under the *national* domain. Table 1 shows the distribution of the final domains in the Shironaam corpus. We use `sbnltk`[7] for tokenizing the documents into sentences. Finally, we discard the samples where any of the information (*i.e.*, headline, article, or image caption) is missing.

2.3 Dataset Statistics

After preprocessing the raw corpus, we have 240,580 news samples as a tuple of (headline, article, image caption, topic words, category). To en-

---

[6] The square brackets contain the English translations.
[7] `https://pypi.org/project/sbnltk`

Dataset	Article	Headline	Image Caption	Topic Words
Average number of words				
Shironaam	252.01	6.53	6.80	3.21
IndicNLG BN	199.83	10.03	-	-
Average number of sentences				
Shironaam	20.05	1.00	1.04	-
IndicNLG BN	15.19	1.19	-	-
Vocabulary size				
Shironaam	605,750	76,732	87,644	-
IndicNLG BN	614,374	65,553	-	-

Table 4: Quantitative statistics compared to IndicNLG-BN (2022) and our proposed dataset **Shironaam**.

sure a balanced distribution, we maintain the ratio of (92% - 220,574), (2% - 4994), and (6% - 15,012) samples from all the categories to construct the train, validation, and test set, respectively (see Table 1). We compare our corpus with the only available benchmark, `IndicNLG` (Kumar et al., 2022), for the news headline generation task in Bengali. Since `IndicNLG` covers multiple languages, we just keep the Bengali (BN) language portion for comparison. Table 2 provides a high-level summary of both datasets.

Our `Shironaam` corpus establishes a new benchmark in terms of the corpus size compared to `IndicNLG` (Kumar et al., 2022). It is important to note that our corpus also contains auxiliary information such as image captions, topic words, and article categories. Moreover, this can be used not only in headline generation tasks but also in some other tasks such as document categorization, news clustering, keyword identification, etc. To measure the abstractiveness, in Table 3, we calculate the percentage of novel n-grams in reference headlines that are not present in the article. Table 3 shows that the novelty level increases with the number of grams, and the average scores are comparable to the `IndicNLG` (Kumar et al., 2022). A quantitative statistics presented in Table 4 demonstrates that our `Shironaam` corpus contains more compressed headlines against lengthier articles compared to the `IndicNLG`, both in terms of words and sentences. This highly compressed nature of the headlines makes the task of headline generation in low-resource language more challenging. In addition, the vocabulary size of our articles is comparable with `IndicNLG`, whereas we get a larger number of vocabularies in our headlines (see Table 4).

Therefore, the `Shironaam` corpus comprises a diverse range of headline styles and provides the largest collection of Bengali news articles. Moreover, it is the first benchmarking dataset in such a low-resource language that includes auxiliary information in addition to the headline-article pairs. We hope it will motivate further study and serve as a baseline for future works on this task for this low-resource language.

3 News Headline Generation

3.1 Task

We establish a new concept of incorporating auxiliary information in order to generate high-quality headlines in Bengali, a low-resource language. In the context of this generation task, we assume that a) we have enough data with auxiliary information to train a headline generation model in Bengali language (can be referred to `Shironaam` corpus); b) the auxiliary information refers to the image captions and topic words used in tagging documents; c) we have access to a module that filters a document based on the contextual similarity with a list of topic words (we refer `BenSim` in Section 3.3). The task can be formalized as follows. Given article $\mathcal{A}$, image caption $\mathcal{C}$, and a set of topic words $\mathcal{T}$ as input, our goal is to generate high-quality headline $\mathcal{H}$ for the corresponding news article.

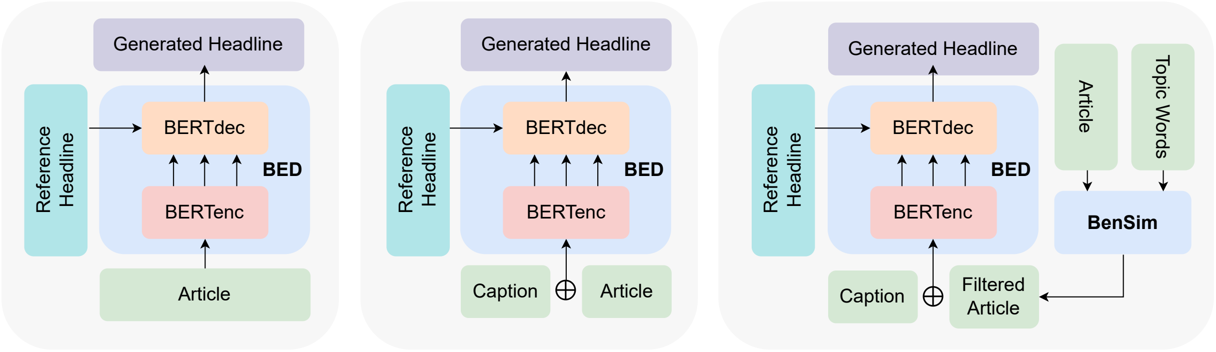
3.2 Approach

To carry out the idea, we need several benchmarks to compare with and evaluate our proposed hypothesis. But, no SOTA benchmark is available for this task in Bengali language[8], except the `IndicBART` (Dabre et al., 2022). So, we set multiple baselines (Section 4.2), both of extractive and abstractive types, that take article $\mathcal{A}$ as input and generate corresponding headline $\mathcal{H}$ as output. We follow `LEAD-1` and `EXT-ORACLE` approaches among the extractive types, whereas from the abstractive types, we initialize an encoder-decoder model for Bengali language with a pre-trained encoder-only checkpoint to skip the costly pre-training (Rothe et al., 2020) [9]. To train the encoder-decoder model (`BED`), we use `BanglaBERT` (Bhattacharjee et al., 2022a) as the encoder checkpoint. Additionally, we utilize other pre-trained models (*i.e.,* `BanglaT5`

---

[8] We did not consider the extreme summarization models since the style of a headline and a single-line summary is completely different.

[9] We refer the interested readers to Appendix (Section A) for necessary background.



(a) BED(base) **(b) BED(w/ Article + Caption)** **(c) BED(w/ FilteredArticle + Caption)**

Figure 1: Graphical illustration of our proposed headline generation models **(a)** BERT-based encoder-decoder (*baseline*) **(b)** Our model incorporating image caption with the article **(c)** Our model uses **BenSim** to extract important sentences (*a.k.a.,* filtered article) based on topic words and incorporates filtered article with the image caption.

(Bhattacharjee et al., 2022b), `IndicBART` (Dabre et al., 2022)) based on transformer architecture (Vaswani et al., 2017). After comparing all the baselines (see Section 4.2), we select the best performing one for further ablations. Experimental results (in Table 5) reveal that `BED` model outperforms other baselines, even though the fine-tuned `BanglaT5` (Bhattacharjee et al., 2022b) scores competitively.

3.3 BERT-based Encoder-Decoder (BED)

A `BED` model consists of an encoder that has been initialized with BERT, termed as `BERTenc`, coupled with a decoder that has also been initialized with BERT, which we call `BERTdec`. The initializing point for each weight's calculation is a public BERT checkpoint. The only variable initialized at random is the encoder-decoder attention (Rothe et al., 2020).

Article Only In Figure 1(a), we implement the basic version *i.e.,* `BED (base)` model, which takes word tokens of an article as a sequence of inputs $\mathcal{A}_{1:n}$ and describes a conditional distribution of target vectors $\mathcal{H}_{1:l}$ of variable length **l**, in our case, generated words for headline:

$$p_{\theta_{BERTenc},\theta_{BERTdec}}(\mathcal{H}_{1:l}|\mathcal{A}_{1:n}). \qquad (1)$$

The input sequence $\mathcal{A}_{1:n}$ is sent to the `BERTenc` component, which then converts it into a sequence of hidden states, $\overline{\mathcal{A}}_{1:n}$. The mapping can be defined as:

$$f_{\theta_{BERTenc}} : \mathcal{A}_{1:n} \rightarrow \overline{\mathcal{A}}_{1:n}. \qquad (2)$$

The `BERTdec` component will simulates the conditional probability distribution of the target vector sequence $\mathcal{H}_{1:l}$, assuming that the sequence of encoded hidden states $\overline{\mathcal{A}}_{1:n}$ has been provided:

$$p_{\theta_{BERTdec}}(\mathcal{H}_{1:l}|\overline{\mathcal{A}}_{1:n}). \qquad (3)$$

Bayes' rule lets us turn this distribution into a product of the conditional probability distribution of the target vector $\mathbf{h}_i$, given the encoded hidden states $\overline{\mathcal{A}}_{1:n}$ and all the previous target vectors $\mathcal{H}_{0:i-1}$:

$$\begin{aligned} &p_{\theta_{BERTdec}}(\mathcal{H}_{1:l}|\overline{\mathcal{A}}_{1:n}) \\ &= \prod_{i=1}^{n} p_{\theta_{BERTdec}}(\mathbf{h}_i|\mathcal{H}_{0:i-1}, \overline{\mathcal{A}}_{1:n}). \end{aligned} \qquad (4)$$

All preceding target vectors $\mathcal{H}_{0:i-1}$ and the encoded hidden state sequence $\overline{\mathcal{A}}_{1:n}$ are mapped to the logit vector $\mathcal{V}_i$ by the `BERTdec`. The next step is to run the `softmax` operation on the logit vector $\mathcal{V}_i$. This helps to define the conditional distribution $p_{\theta_{BERTdec}}(\mathbf{h}_i|\mathcal{H}_{0:i-1}, \overline{\mathcal{A}}_{1:n})$ by making sure that the distribution of the target vector $\mathbf{h}_i$ depends on the distributions of all previous target vectors $\mathbf{h}_0, \ldots, \mathbf{h}_{i-1}$:

$$p_{\theta_{BERTdec}}(\mathbf{h}_i|\mathcal{H}_{0:i-1}, \overline{\mathcal{A}}_{1:n}) = \mathbf{Softmax}(\mathcal{V}_i). \quad (5)$$

The first target vector $\mathbf{h}_0$ is going to be represented by a unique `BOS` vector that is referred to as the "beginning-of-sentence". After the conditional distribution $p_{\theta_{BERTdec}}(\mathbf{h}_i|\mathcal{H}_{0:i-1}, \overline{\mathcal{A}}_{1:n})$ has been set, the output can be made in an auto-regressive way. This makes it possible to define a mapping between an input sequence $\mathcal{A}_{1:n}$ and an output sequence $\mathcal{H}_{1:l}$ at the time of inference.

Fusing Article and Image Caption In order to explore more ways to improve the quality of the generated headlines, we employ `BED (w/ Article + Caption)` model, which incorporates image caption $\mathcal{C}_{1:m}$ with the corresponding article $\mathcal{A}_{1:n}$ as in Figure 1(b), where **m** << **n**, and passes them through the `BERTenc` using parallel-fusion (Liu et al., 2020a) mechanism:

$$\mathcal{K}_{1:r} = \mathcal{C}_{1:m} \oplus \mathcal{A}_{1:n}, \tag{6}$$

$$f_{\theta_{BERTenc}} : \mathcal{C}_{1:m}, \mathcal{A}_{1:n} \rightarrow \overline{\mathcal{K}}_{1:r}. \tag{7}$$

Here, $\mathcal{K}_{1:r}$ denotes the model input sequence, where **r** represents the new input sequence length, and $\oplus$ is concatenation operator separated by a special token. The sequence of hidden states $\overline{\mathcal{K}}_{1:r}$ are then processed through the `BERTdec` likewise the `Shironaam(base)` model and the headline is generated as output:

$$p_{\theta_{BERTenc},\, \theta_{BERTdec}}(\mathcal{H}_{1:l}|\mathcal{C}_{1:m}, \mathcal{A}_{1:n}). \tag{8}$$

However, the image caption may not always serve the full context if the news article becomes too long for `BERTenc`. Moreover, the image caption length is generally much smaller than the news article length. Thus, the impact of using image caption as a context is less sensitive for lengthier articles.

Bengali Sentence Similarity (BenSim) Since many of the news articles' lengths exceed the input sequence limit that `BERTenc` can process, we therefore, utilize the sequence length by ensuring all the relevant sentences are present in the limited input sequences. To ensure the extraction of relevant sentences, we develop `BenSim` module[10], a tool for measuring semantic similarity between Bengali sentences utilizing BERT embeddings. It takes news article $\mathcal{A}_{1:n}$ and corresponding topic words $\mathcal{T}_{1:k}$ as input for getting most of the contextual sentences and employs pre-trained `bangla-bert-base` (Sarker, 2020) model on both of the input sequences to generate the contextualized encoded representations. After performing mean pooling operation, cosine similarity (Singhal, 2001) is then applied to the encoded sequences to get the similarity score. After measuring the similarities between the topic words and input sentences, a filtered article $\mathcal{A'}_{1:r}$ is returned as output, which is then fused parallelly with the image caption $\mathcal{C}_{1:m}$ and sent to the model input. Finally, the `BED (w/ FilteredArticle + Caption)` model produces a headline after processing the fused input.

4 Experiments and Benchmarks

In this section, we set a new benchmark for Bengali news headline generation using `Shironaam` corpus and compare it with the other state-of-the-art baselines. After a clean comparison, we perform two ablation experiments on the superior base model. Then, we analyze the performance gap between the baselines and the ablations and further evaluate the best model on news domains with a few samples (few-shot). Finally, after proper analysis, we seek to find out the answers to the following research questions:

- **RQ#1:** Can we use auxiliary information (*e.g.,* image caption and topic words) to improve the performance of the headline generation?
- **RQ#2:** Which domain(s) benefit from the auxiliary information in few-shot and non-few-shot settings?

4.1 Implementation Details

We utilize the encoder-decoder paradigm[11] of HuggingFace, where pre-trained `BanglaBERT` (Bhattacharjee et al., 2022a)[12] is used to initialize both of the weights of encoder and decoder. Before proceeding to tokenization, we perform sentence normalization, introduced in Hasan et al. (2020). For tokenization, we use the pre-trained tokenizer[12] that comes with the model. All the hyper-parameters used for training and decoding are presented in Section B in the Appendix.

Evaluation Metrics We compare the performance with the following baselines across several evaluation metrics presented in Section D in the Appendix.

4.2 Baselines

LEAD-1 `LEAD-1` is a commonly used baseline for setting the lower bound of news headline generation task (Kumar et al., 2022; Narayan et al., 2018). It also indicates the degree of positional biasness of article body sentences in generating headlines. We pick the article's first sentence as the system headline and compare it with the original headline to generate the `LEAD-1` scores.

---

[10] https://github.com/dialect-ai/BenSim

[11] Encoder-Decoder models documentation

[12] BanglaBERT usage (HuggingFace)

Models	ROUGE			BLEU			BERT Score	METEOR Score
	R-1	R-2	R-L	BLEU Score	Brevity Penalty	Length Ratio		
Baselines								
LEAD-1 (Extractive)	30.50	13.86	28.00	5.65	97.71	2.48	74.63	29.90
EXT-ORACLE (Extractive)	39.92	22.89	37.28	9.17	97.16	2.30	77.16	39.65
IndicBART (mBART)	28.76	12.65	27.11	15.03	99.91	1.14	74.95	20.39
BanglaT5 (mT5)	44.13	23.03	42.12	13.05	91.33	1.15	80.13	34.65
Our Ablations								
BED Base (BERT2BERT)	44.22	24.18	42.28	22.06	94.47	0.94	80.53	34.16
-w/ Article + Caption	51.62	33.62	49.94	31.39	96.02	0.96	82.93	42.57
-w/ FilteredArticle + Caption	**52.19**	**34.27**	**50.31**	**31.80**	**98.57**	**0.99**	**83.10**	**43.52**

Table 5: Performance on `Shironaam (test)` corpus compared to the baselines (Section 4.2) and the results of our ablation study (see Appendix Section E for validation scores) across various automatic evaluation metrics, where **bold-faced** scores indicate superior performance.

EXT-ORACLE On the other hand, `EXT-ORACLE` can be considered as the upper bound of generating headlines by an extractive approach (Kumar et al., 2022; Narayan et al., 2018). We implement this baseline on the `Shironaam (test)` corpus by aligning a sentence from the input article with the reference headline based on the ROUGE-2 metric.

IndicBART Kumar et al. (2022) releases a multilingual model, which is fine-tuned on `IndicBART` (Dabre et al., 2022) checkpoint for the headline generation task focusing on Indic languages including Bengali. `IndicBART` is a sequence-to-sequence multilingual pre-trained model (Dabre et al., 2022) based on the mBART (Liu et al., 2020b) architecture.

BanglaT5 We fine-tune `BanglaT5` (Bhattacharjee et al., 2022b), a sequence-to-sequence transformer model based on mT5 (Xue et al., 2021) architecture for Bengali language, on the `Shironaam (train)` corpus for the headline generation task. For a fair comparison, we maintain the same hyper-parameters.

BED (base) Model We implement the model (article only), illustrated in Figure 1(a) on the `Shironaam (train)` corpus to make the baseline. We utilize 220,500 news samples from the train set to train the `BED` model, which takes the article only as input and generates a headline as output. The evaluation result on the `Shironaam (test)` set is shown in Table 5, which is a new benchmark for the Bengali headline generation task. In the following experiments, we utilize the auxiliary information with the same hyper-parameter settings to generate better-quality headlines.

4.3 Ablation Experiments

BED (w/ Article + Caption) Model As per the demonstration in Figure 1(b), the image caption is incorporated with the input article. This leads to a much improved result across all evaluation metrics compared to article only model (a.k.a., `BED (base)`) as shown in Table 5.

BED (w/ FilteredArticle + Caption) Model Since the utilization of image caption in model input gives better results, therefore we further enrich the inputs by incorporating topic words. We use topic words in filtering the longer articles through `BenSim` rather than using them directly to the input, as shown in Figure 1(c). First, we set a threshold value (40 in our case) for `BenSim` to extract the number of top semantically similar sentences. `BenSim` maintains the relative appearance order of the sentences in the original article to construct the corresponding filtered article. To fix the number of sentences in a filtered article, we consider the maximum use of the number of tokens `BED` model can afford *i.e.* 512. Fusing filtered article with image caption achieves the best results across several evaluation metrics as shown in Table 5 (also see Appendix Section F for the generation quality).

4.4 Discussions

Result Analysis Table 5 shows that the `LEAD-1` baseline performs inadequately on the `Shironaam (test)` corpus. More specifically, the ROUGE-2 and BLEU scores and the length ratio indicate that the original headlines are more abstractive in nature, and the first sentence of an article does not contain sufficient information for generating a headline. Unlike `LEAD-1`, comparatively higher ROUGE scores are obtained by using

EXT-ORACLE, but at the same time, BLEU score gain is lower. This trade-off indicates that the reference headlines consist of the subset of words present in the sentences selected by EXT-ORACLE. However, because of the concise nature of news headlines, this approach does not fit well but can be considered a strong baseline for other models. Among the abstractive types, IndicBART performs poorly on the Shironaam (test) corpus and even is unable to beat the weak LEAD-1 baseline, let alone EXT-ORACLE. On the other hand, the fine-tuned BanglaT5 yields a good score for this task. Although the generated results are slightly lengthier than the reference ones, they can be considered a strong baseline. The BED (base) model provides the best performance in terms of ROUGE, BLEU, and BERT scores. So, we consider it the strongest baseline and look for further ablations.

To this end, Table 5 shows that the best baseline is outperformed by our proposed technique of modeling input using auxiliary data. We want to emphasize that we use image caption and topic words purely as auxiliary data. While collecting the data from various news portals, we observe that it is very common to include images to help support and communicate the story and image captions are a crucial part of it that only describe the referred image. Although image captions are mostly correlated with the corresponding article in terms of context, we argue that they are not headlines. First, there is not much overlap in terms of Jaccard similarity measured between image captions and headlines (as from Table 1 we have approximately 29% overlap across different categories). Second, headlines differ from image captions in terms of styles and content.

Image captions usually give the model some signal on which parts of the document model need to attend more. Hence, as a result of combining image caption with article, BED (w/ Article + Caption) model improves the performance by about 3 (BERT score) to 10 (ROUGE-2 score) percentage points and produces more human-like headlines. Moreover, it often begins generating sentient headlines that are more abstract and profound than the reference ones. The BED (w/ FilteredArticle + Caption) model performs slightly better than the previous ablation. Since, there are fewer lengthier articles in the Shironaam corpus, the variations in the scores of the two ablation models are rather small. We observe that when we include filtered articles led by relevant topic words in the model's input, it begins to learn faster than the model without topic words (demonstrated in Figure 2). Therefore, the differences between the scores of two ablation models will increase with the number of lengthier articles.

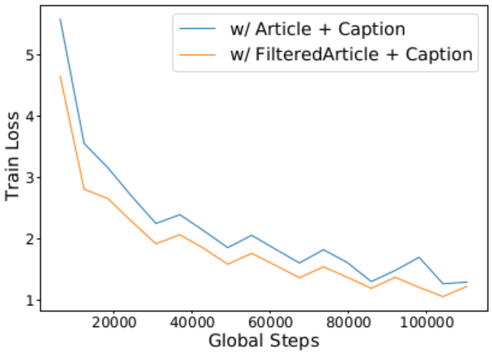


Figure 2: Train Loss vs. Global Steps for our ablations.

So, following the question **RQ#1**, we may conclude from the preceding discussion that auxiliary information definitely aids in creating better headlines. Although we achieve superior performance compared to the state-of-the-art baselines across several evaluation metrics, these quantitative measures can not determine the generation quality. Therefore, we present generated samples from our model categorized into several abstractive types (see Section F in Appendix). We leave the human evaluation of our generated samples as one of the future works.

Domain Specific Analysis We evaluate our proposed BED (w/ FilteredArticle + Caption) [denoted as **BED (FA+C)**] model on individual domains by comparing with a base model and to answer **RQ#2**. We also observe the performance of the presented model on the domains with fewer samples (few-shot). We employ two baselines here: BED (base) and BanglaT5 (Bhattacharjee et al., 2022b) [denoted as **BNT5**]. Although, the BNT5 has demonstrated competitive performance, Table 6 shows that BED (base) model performs better on the maximum number of domains. To calculate the exact performance gap, we maintain a uniform baseline *i.e.,* BED (base) to compare with the proposed model. For the few-shot observation, based on the number of training samples, we split the domains into two folds (see Table 6). The *Few-Shot* domains contain less than 6500 train samples, whereas rest of the domains are considered as *Non-Few-Shot*.

Category	R-1			R-2			R-L		
	BED (base)	BNT5	BED (FA+C)	BED (base)	BNT5	BED (FA+C)	BED (base)	BNT5	BED (FA+C)
Non-Few-Shot Domains									
National	48.03	47.33	55.84	27.29	25.83	37.88	46.06	45.37	53.95
International	44.44	46.04	50.47	22.92	23.08	29.96	42.02	43.49	48.13
Sports	30.14	33.46	39.20	11.57	13.43	20.40	28.75	31.59	37.33
Entertainment	33.05	32.99	35.14	15.07	14.32	16.64	31.26	31.33	33.44
Politics	49.28	49.66	**57.16**	28.80	27.32	**39.73**	47.53	47.68	**55.73**
Few-Shot Domains									
Economy	38.95	40.03	60.32	18.81	19.74	45.85	36.44	37.62	58.53
Life-Health	35.87	39.20	44.97	17.61	19.78	27.21	33.90	37.38	43.08
Edu-Career	50.57	51.12	71.55	31.92	30.82	59.54	48.05	48.82	70.48
Opinion	16.11	15.82	44.53	4.69	5.24	36.63	15.82	15.44	44.25
Miscellaneous	33.64	34.92	35.29	16.16	17.98	17.41	30.48	32.82	31.87
Science-Tech	41.82	44.14	51.03	19.54	22.61	31.20	39.30	41.82	48.49
Nature	36.07	37.89	46.54	15.78	16.65	30.07	34.84	35.79	45.53
Religion	27.29	35.48	**72.10**	12.28	19.63	**62.05**	26.96	34.42	**72.14**

Table 6: Performance of our proposed model `BED(FA+C)` compared to baseline `BED(base)` and `BNT5` (2022b) across different domains. Shaded grey region indicates superior performance compared to baselines and **bold-faced** and underlined scores indicate comparably best and worst domains, respectively.

Table 6 demonstrates that our proposed model improves the scores by a satisfactory margin of almost all the domains except *Entertainment* and *Miscellaneous*. These two categories get comparatively lower scores. The majority of headlines in the *Entertainment* domain are casual and clickbait-style and do not maintain the identical nature of a particular domain. We argue that the discrepancy, in this case, decreases the scores. The *Miscellaneous* domain is comprised of different sorts of randomness containing articles of various domains. Therefore, it is anticipated that this genre will get a lower score. Table 6 shows that our proposed model maintains consistent performance when there are few samples to train.

5 Related Works

Headline generation is an under-explored subtask of abstractive summarization, particularly in languages with limited resources. For the English language, an attention-based neural network has been proposed by Rush et al. (2015) for abstractive sentence summarization. The authors propose a model that utilizes a recurrent neural network (RNN) and an attention mechanism to summarize input sentences into a compact summary. Takase et al. (2016) build an AMR encoder for headline creation based on an encoder-decoder architecture. Using a dual-attention seq2seq model, Zhang et al. (2018) proposes a way for question headline development. In limited resource settings, Tilk and Alumäe (2017) pretrain a neural encoder and decoder model to enhance headline generation outputs. A sentence encoder, a gate network for sentence selection, and a headline decoder are the three stages of Zhou et al. (2017)'s headline generation approach. Tan et al. (2017) proposes a coarse-to-fine strategy that extracts the most important sentences before generating the headlines based on the context. For headline generation, Kumar et al. (2022) have released the `IndicNLG`, a collection of multilingual datasets. However, they do not provide any additional attributes besides the headline-article pairs. In summary, the majority of the past works for generating headlines primarily used the article content to generate headlines.

6 Conclusion and Future Work

In this paper, we contributed a large-scale dataset (*a.k.a.,* **Shironaam**) with auxiliary information such as image captions, topic words, and category for Bengali news headline generation. We employ contextualized language models to incorporate such auxiliary information and proposed a simple yet effective solution to encode long articles using topic words. Experimental results demonstrate the superiority of our approach across different domains and settings. We anticipate that our efforts will motivate the community to expand the scope of headline generation tasks beyond English, particularly for a low-resource language like Bengali. Our future work will look into incorporating auxiliary information to support more languages and build a language-agnostic model.

Limitations

Our model relies on auxiliary information such as image captions and topic words to achieve superior performance. However, it is quite common to include images and extra information (*e.g.*, topic words) to increase the article's visibility, support, and context. Also, our base model without auxiliary information demonstrates improved performance compared to the well-established and state-of-the-art baselines. Another limitation we observed that our model did not perform as well as for the Miscellaneous and Entertainment categories compared to the other 11 different categories because of the clickbaity nature of these categories. Finally, our headline generation model works only for Bengali, a widely spoken but low-resource language. Still, this idea of using auxiliary information to improve headline generation performance can easily be extendable for many languages.

Ethics Statement

We considered some ethical aspects while scraping the data. We requested data at a reasonable rate without any intention of a DDoS attack. Moreover, for each website, we read the instructions listed in robots.txt to check whether we can crawl the intended content. We tried to minimize offensive texts in the data by explicitly crawling the sites where such contents are minimal. Further, we removed the Personal Identifying Information (PII) such as name, phone number, email address etc from the corpus.

Acknowledgements

We thank all the anonymous reviewers for their valuable feedback and constructive suggestions for improving this work. Mir Tafseer Nayeem is supported by a Huawei Doctoral Scholarship.

References

Xiang Ao, Xiting Wang, Ling Luo, Ying Qiao, Qing He, and Xing Xie. 2021. PENS: A dataset and generic framework for personalized news headline generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 82–92, Online. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Ahmad, Kazi Samin Mubasshir, Md Saiful Islam, Anindya Iqbal, M. Sohel Rahman, and Rifat Shahriyar. 2022a. BanglaBERT: Language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1318–1327, Seattle, United States. Association for Computational Linguistics.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Uddin Ahmad, and Rifat Shahriyar. 2022b. Banglanlg: Benchmarks and resources for evaluating low-resource natural language generation in bangla. *CoRR*, abs/2205.11081.

Susmoy Chakraborty, Mir Tafseer Nayeem, and Wasi Uddin Ahmad. 2021. Simple or complex? learning to predict readability of bengali texts. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12621–12629.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Radia Rayan Chowdhury, Mir Tafseer Nayeem, Tahsin Tasnim Mim, Md. Saifur Rahman Chowdhury, and Taufiqul Jannat. 2021. Unsupervised abstractive summarization of Bengali text documents. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2612–2619, Online. Association for Computational Linguistics.

Carlos A Colmenares, Marina Litvak, Amin Mantrach, Fabrizio Silvestri, and Horacio Rodríguez. 2019. Headline generation as a sequence prediction with conditional random fields. In *Multilingual Text Analysis: Challenges, Models, And Approaches*, pages 201–243. World Scientific.

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh Khapra, and Pratyush Kumar. 2022. IndicBART: A pre-trained model for indic natural language generation. In *Findings of the Association for Computational Linguistics: ACL*

2022, pages 1849–1863, Dublin, Ireland. Association for Computational Linguistics.

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

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Md. Majharul Haque, Suraiya Pervin, and Zerina Begum. 2016. Enhancement of keyphrase-based approach of automatic bangla text summarization. In *2016 IEEE Region 10 Conference (TENCON)*, pages 42–46.

Tahmid Hasan, Abhik Bhattacharjee, Kazi Samin, Masum Hasan, Madhusudan Basak, M. Sohel Rahman, and Rifat Shahriyar. 2020. Not low-resource anymore: Aligner ensembling, batch filtering, and new datasets for Bengali-English machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2612–2623, Online. Association for Computational Linguistics.

Pratik Joshi, Christain Barnes, Sebastin Santy, Simran Khanuja, Sanket Shah, Anirudh Srinivasan, Satwik Bhattamishra, Sunayana Sitaram, Monojit Choudhury, and Kalika Bali. 2019. Unsung challenges of building and deploying language technologies for low resource language communities. In *Proceedings of the 16th International Conference on Natural Language Processing*, pages 211–219, International Institute of Information Technology, Hyderabad, India. NLP Association of India.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Aman Kumar, Himani Shrotriya, Prachi Sahu, Raj Dabre, Ratish Puduppully, Anoop Kunchukuttan, Amogh Mishra, Mitesh M Khapra, and Pratyush Kumar. 2022. Indicnlg suite: Multilingual datasets for diverse nlg tasks in indic languages. *arXiv preprint arXiv:2203.05437*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ping Li, Jiong Yu, Jiaying Chen, and Binglei Guo. 2021. Hg-news: News headline generation based on a generative pre-training model. *IEEE Access*, 9:110039–110046.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Dayiheng Liu, Yeyun Gong, Yu Yan, Jie Fu, Bo Shao, Daxin Jiang, Jiancheng Lv, and Nan Duan. 2020a. Diverse, controllable, and keyphrase-aware: A corpus and method for news multi-headline generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6241–6250, Online. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020b. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Myle Ott, Naman Goyal, J Du, M Joshi, D Chen, O Levy, M Lewis, L Zettlemoyer, and V Stoyanov. 1907. Roberta: A robustly optimized bert pretraining approach. arxiv 2019. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Prakhar Mishra, Chaitali Diwan, Srinath Srinivasa, and G. Srinivasaraghavan. 2021. Automatic title generation for text with pre-trained transformer language model. In *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*, pages 17–24.

Kazuma Murao, Ken Kobayashi, Hayato Kobayashi, Taichi Yatsuka, Takeshi Masuyama, Tatsuru Higurashi, and Yoshimune Tabuchi. 2019. A case study on neural headline generation for editing support. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 73–82, Minneapolis, Minnesota. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Mir Tafseer Nayeem and Yllias Chali. 2017a. Extract with order for coherent multi-document summarization. In *Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing*, pages 51–56, Vancouver, Canada. Association for Computational Linguistics.

Mir Tafseer Nayeem and Yllias Chali. 2017b. Paraphrastic fusion for abstractive multi-sentence compression generation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 2223–2226, New York, NY, USA. Association for Computing Machinery.

Mir Tafseer Nayeem, Tanvir Ahmed Fuad, and Yllias Chali. 2018. Abstractive unsupervised multi-document summarization using paraphrastic sentence fusion. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1191–1204, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Mir Tafseer Nayeem, Tanvir Ahmed Fuad, and Yllias Chali. 2019. Neural diverse abstractive sentence compression generation. In *Advances in Information Retrieval*, pages 109–116, Cham. Springer International Publishing.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging Pre-trained Checkpoints for Sequence Generation Tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Sagor Sarker. 2020. Banglabert: Bengali mask language model for bengali language understading.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Amit Singhal. 2001. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24:35–43.

Yun-Zhu Song, Hong-Han Shuai, Sung-Lin Yeh, Yi-Lun Wu, Lun-Wei Ku, and Wen-Chih Peng. 2020. Attractive or faithful? popularity-reinforced learning for inspired headline generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8910–8917.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings*, page 194–206, Berlin, Heidelberg. Springer-Verlag.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. 2016. Neural headline generation on Abstract Meaning Representation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1054–1059, Austin, Texas. Association for Computational Linguistics.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. From neural sentence summarization to headline generation: A coarse-to-fine approach. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 4109–4115. AAAI Press.

Ottokar Tilk and Tanel Alumäe. 2017. Low-resource neural headline generation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 20–26, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Kosuke Yamada, Yuta Hitomi, Hideaki Tamori, Ryohei Sasano, Naoaki Okazaki, Kentaro Inui, and Koichi Takeda. 2021. Transformer-based lexically constrained headline generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4085–4090, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, Jun Xu, Huanhuan Cao, and Xueqi Cheng. 2018. Question headline generation for news articles. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 617–626, New York, NY, USA. Association for Computing Machinery.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1049–1058, Florence, Italy. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective encoding for abstractive sentence summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1095–1104, Vancouver, Canada. Association for Computational Linguistics.

A Preliminaries

Abstractive text summarization (Rush et al., 2015; See et al., 2017; Zhang et al., 2020) was considerably more challenging before the development of sequence-to-sequence (seq2seq) models (Cho et al., 2014; Sutskever et al., 2014) and recent advances in transformer-based models (Vaswani et al., 2017; Devlin et al., 2019) due to a lack of sufficient datasets. Many text-summarizing applications are still hindered by the lack of suitable datasets, particularly for low-resource languages (Joshi et al., 2019). After being presented in Vaswani et al. (2017), models based on transformer architectures have been proven to perform better on sequence-to-sequence tasks than decoder-only language models *e.g.* Raffel et al. (2020). In its most basic form, an encoder-decoder model comprises a stand-alone encoder, like `BERT` (Devlin et al., 2019), and a stand-alone decoder model, like `GPT2` (Radford et al., 2019). It has been demonstrated that the huge pre-trained encoder-decoder models may considerably improve performance on a range of sequence-to-sequence tasks Lewis et al. (2020); Raffel et al. (2020). On the other hand, pre-training encoder-decoder models are very expensive to build since the models require a lot of computational resources.

Rothe et al. (2020) introduces the encoder-decoder model using pre-trained encoder and/or decoder-only checkpoints (such as `BERT` (Devlin et al., 2019) and `GPT2` (Radford et al., 2019)) to avoid the time-consuming pre-training process. According to Rothe et al. (2020), these encoder-decoder models can do well as large pre-trained encoder-decoder models like `T5` (Raffel et al., 2020) and Pegasus (Zhang et al., 2020) on different sequence-to-sequence tasks at a fraction of the training cost.

B Hyper-parameters, Training, and Decoding

All the `BED` models (Figure 1) are trained almost from scratch by maintaining uniform hyper-parameters and trained for 110,250 global steps with the learning rate `5e-5`, and batch size `12`. We save the best checkpoint by ensuring the lowest validation loss. We use `AdamW` (Loshchilov and Hutter, 2019) for optimizing the loss with default linear warmup. The maximum lengths of encoder and decoder are limited to `512` and `32` tokens, respectively. Each of the `BED` models is trained on a single NVIDIA Tesla P100 GPU and trained for approximately 33 hours, which takes almost 5 hours 30 minutes per epoch. The total number of trainable parameters is 249,044,480.

Decoding When validating and testing, we use beam search algorithm (Sutskever et al., 2014) with `4` beams to generate headlines. The maximum and minimum lengths used in generating the headlines are `16` and `4`, respectively. We use '`early stopping`' to stop the beam search when at least `4` sentences are finished per batch. The '`no-repeat n-gram size`' is set to `2`, where the exponential penalty to the length is `1.2`. Regarding vocabulary size, we use the fixed `32,000` vocabularies from the encoder.

C Data Sources

Newspaper	URL
Prothom Alo	`www.prothomalo.com`
Naya Diganta	`www.dailynayadiganta.com`
Ajker Patrika	`www.ajkerpatrika.com`
Bangladesh Protidin	`www.bd-pratidin.com`
Samakal	`www.samakal.com`
Bhorer Kagoj	`www.bhorerkagoj.com`
Dhaka Tribune	`www.dhakatribune.com`

Table 7: List of Bengali newspapers to form the Shironaam corpus with their corresponding URLs. Samples from the *italic-faced* newspapers were crawled through their APIs.

D Evaluation Metrics

We evaluate the predicted headlines with some automatic metrics used for generation tasks. The generation quality is measured with the ROUGE (Lin, 2004) F1 score[13]. ROUGE-1 and ROUGE-2 measure informativeness, where fluency is measured by the longest common subsequence (ROUGE-L). We include BLEU (Papineni et al., 2002) score which indicates the similarity between reference and predicted sentences by comparing the overlap within tokens[14]. Brevity penalty and length ratio are shown to justify the BLEU score. The contextual similarity between the generated and reference headline is measured using F1 BERT score (Zhang* et al., 2020)[15], where the correlation between them is reported by METEOR score (Banerjee and Lavie, 2005)[16]. We use the available open-source implementations for the above metrics.

---

[13] ROUGE (multilingual)

[14] BLEU (HuggingFace)

[15] BERTScore (HuggingFace)

[16] METEOR (HuggingFace)

E Validation Results

	Train Loss	Valid Loss	ROUGE			BLEU		METEOR Score
			R-1	R-2	R-L	Bleu Score	Length Ratio	
a)	1.0892	2.4332	44.51	23.56	42.38	20.39	0.95	34.27
b)	1.5083	2.1227	49.59	30.53	47.79	27.63	0.98	40.59
c)	1.2199	2.0836	49.77	31.42	48.05	28.70	0.99	40.73

Table 8: All the scores are reported for `BED` model with ablations on `Shironaam (valid)` corpus. The labels indicate the ablations of `BED` model: **a)** Base, **b)** Article + Caption, **c)** FilteredArticle + Caption. Only the *Train Loss* is measured on the training set and kept for comparison with the *Valid Loss*.

F Generated Headlines

Generated headlines on **Shironaam** (*test*) corpus across all the categories are presented in Table 9.

Category		**Headline**	**Type**
Economy	GH	চিনি আমদানিতে শুল্ক কমালো সরকার	**Inserted**
	ET	The government reduced the duty on sugar import	
	RH	চিনি আমদানিতে শুল্ক কমলো	
	ET	Import duty on sugar reduced	
Edu-Career	GH	সহকারী জজ নিয়োগের লিখিত পরীক্ষার সূচি প্রকাশ	**Matched**
	ET	Release of written test schedule for the appointment of Assistant Judge	
	RH	সহকারী জজ নিয়োগের লিখিত পরীক্ষার সূচি প্রকাশ	
	ET	Release of written test schedule for the appointment of Assistant Judge	
Entertainment	GH	"আমি বিয়ে করব না, দেখি কে আমাকে বিয়ে করে"	**Swapped**
	ET	“I will not marry, let’s see who marries me”	
	RH	"আমি বিয়ে করব না, কে আমাকে বিয়ে করে দেখি..."	
	ET	“I will not marry, who will marry me let’s see...”	
International	GH	জাপানে পর্যটকবাহী জাহাজ ডুবে নিখোঁজ ২৬	**Matched**
	ET	Tourist ship sinks in Japan and goes missing 26	
	RH	জাপানে পর্যটকবাহী জাহাজ ডুবে নিখোঁজ ২৬	
	ET	Tourist ship sinks in Japan and goes missing 26	
Life-Health	GH	নতুন মৃত্যু ৩৭, শনাক্ত ৩০৪৫	**Sentient**
	ET	New deaths 37, detections 3045	
	RH	আক্রান্ত ছাড়ালো ৫৫ হাজার	
	ET	Number of infected has crossed 55 thousand	
Miscellaneous	GH	চার অক্ষরে সন্তানদের নাম!	**Swapped**
	ET	Four-letters in children’s name!	
	RH	সন্তানের নাম চার অক্ষরে!	
	ET	Children’s names are in four-letters!	
National	GH	ঘোড়া প্রতীক না পেয়ে কেঁদে ফেললেন চেয়ারম্যান প্রার্থী	**Deleted**
	ET	Chairman candidate cried after not getting the horse symbol	
	RH	ঘোড়া প্রতীক না পেয়ে কেঁদে ফেললেন সেই চেয়ারম্যান প্রার্থী	
	ET	That chairman candidate cried after not getting the horse symbol	
Nature	GH	দেশের ৩ বিভাগে বৃষ্টির পূর্বাভাস	**Paraphrased**

	ET	Rain forecast in 3 divisions of the country	
	RH	রাতে বাড়বে তাপমাত্রা, ৩ বিভাগে বৃষ্টির আভাস	
	ET	The temperature will increase at night, there is a chance of rain in 3 divisions	
Opinion	GH	ই-কমার্সবান্ধব বাজেট চাই	**Deleted**
	ET	Want e-commerce friendly budgeting	
	RH	ই-কমার্সবান্ধব বাজেট প্রণয়নে কাজ করতে চাই	
	ET	Want to work on e-commerce friendly budgeting	
Politics	GH	বিদ্যুৎ-গ্যাসের মূল্যবৃদ্ধির সিদ্ধান্ত গণবিরোধী পদক্ষেপ : গণফোরাম	**Inserted**
	ET	The decision to increase the price of electricity and gas is an anti-people move : Public Forum	
	RH	"বিদ্যুৎ ও গ্যাসের মূল্যবৃদ্ধির সিদ্ধান্ত হবে গণবিরোধী"	
	ET	“The decision to increase the price of electricity and gas will be anti-people”	
Religion	GH	পবিত্র শবে বরাত পালিত	**Sentient**
	ET	Holy Shab-e-barat is celebrated	
	RH	ইবাদতে মশগুল ধর্মপ্রাণ মুসলমানেরা	
	ET	Devoted Muslims engaged in prayer	
Science-Tech	GH	মহাকাশে স্যাটেলাইটের সংখ্যা বাড়াচ্ছে ওয়ানওয়েব	**Inserted**
	ET	OneWeb is increasing number of satellites into space	
	RH	মহাকাশে স্যাটেলাইট বাড়াচ্ছে ওয়ানওয়েব	
	ET	OneWeb is increasing satellites into space	
Sports	GH	মেসি এখন পিএসজির জার্সিতে	**Paraphrased**
	ET	Messi is now in PSG jersey	
	RH	পিএসজির হয়ে কবে মাঠে নামছেন মেসি?	
	ET	When is Messi on the field for PSG?	

Table 9: High quality headlines generated on **Shironaam (test)** corpus across all the categories. Here, “Type” means the how the generated headlines are different from the references. We categorize the differences into 5 types: **Inserted** (only one/some word(s) is/are added to reference headline), **Matched** (generated exactly the same), **Swapped** (the only difference is made by swapping one/some word(s) within the reference headline), **Deleted** (the output is about similar to the reference with one/some word(s) less), **Sentient** (generated headline is completely different but a potential competitor against the reference one), **Paraphrased** (paraphrased version of the reference headline). The colored words (*i.e.* teal for *Inserted*, cyan for *Swapped*, brown for *Sentient*, magenta for *Deleted*, and violet for *Paraphrased*) indicate the exact positions where the generated ones are different from the references and no color refers to no change. The generated and reference Bengali headlines, and their corresponding English version are denoted by **GH, RH,** and **ET** respectively.

PCC: Paraphrasing with Bottom-k Sampling and Cyclic Learning for Curriculum Data Augmentation*

Hongyuan Lu, Wai Lam
The Chinese University of Hong Kong
`{hylu,wlam}@se.cuhk.edu.hk`

Abstract

Curriculum Data Augmentation (CDA) improves neural models by presenting synthetic data with increasing difficulties from easy to hard. However, traditional CDA simply treats the ratio of word perturbation as the difficulty measure and goes through the curriculums only once. This paper presents **PCC**: **P**araphrasing with Bottom-k Sampling and **C**yclic Learning for **C**urriculum Data Augmentation, a novel CDA framework via paraphrasing, which exploits the textual paraphrase similarity as the curriculum difficulty measure. We propose a curriculum-aware paraphrase generation module composed of three units: a paraphrase candidate generator with bottom-k sampling, a filtering mechanism and a difficulty measure. We also propose a cyclic learning strategy that passes through the curriculums multiple times. The bottom-k sampling is proposed to generate super-hard instances for the later curriculums. Experimental results on few-shot text classification as well as dialogue generation indicate that PCC surpasses competitive baselines. Human evaluation and extensive case studies indicate that bottom-k sampling effectively generates super-hard instances, and PCC significantly improves the baseline dialogue agent.

1 Introduction

Data augmentation techniques create artificial data mixed with the original data for improved performance. Traditional data augmentation techniques in the language community include word-level perturbation such as synonym replacement, random insertion, random swap, and random deletion (Wei and Zou, 2019). Sentence-level techniques such as Round-trip Translation (Sennrich et al., 2016b) exploits the use of machine translation models to translate the input sentence to another language before translating back to the source language which can be essentially treated as a form of paraphrasing.

Curriculum learning presents training instances in a meaningful order with increasing difficulties to neural models for a boost in performance. Traditional curriculum learning (Bengio et al., 2009; Liu et al., 2018, 2020; Platanios et al., 2019; Xu et al., 2020a,b; Su et al., 2021) categorizes the original training instances into different levels of difficulties to be gradually presented to the model where a core component called difficulty measure, which is usually defined as a numerical number where a bigger number indicates a more difficult sample.

Combining the merits of the above two mentioned techniques, Curriculum Data Augmentation (CDA) creates synthetic data with increasing levels of difficulties to be presented to our neural models. Existing CDA defines the ratio of the words perturbation as the difficulty measure for curriculums and a gradual course which increases the difficulty of curriculums when the training loss plateaus (Wei et al., 2021), which then ends when the most challenging curriculum ends. Although existing CDA is effective, yet there are several disadvantages. First, it employs word-level perturbation. This superficial operation keeps the augmentation to have a similar sentence structure as the original one. Next, it employs random insertion, random swap, and random deletion for augmentation. Although this can be durable as for text classification (Wei et al., 2021), this is not suitable for generation tasks, particularly when many words are perturbed, which can even easily break the sentence grammar. Third, it uses a gradual course that only enters each level of difficulty once. A typical problem in neural network training called catastrophic forgetting (Kirkpatrick et al., 2017) can potentially happen in such a course, where the model might undesirably gradually forget some early learned knowledge.

To mitigate the problems of word-level perturba-

---

*The work described in this paper is substantially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: 14200719).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 68–82
May 2-6, 2023 ©2023 Association for Computational Linguistics

tion, we propose that paraphrasing can be a source of data augmentation, which provides diverse and grammatically correct augmentation. However, it is non-trivial to utilize paraphrase augmentation in a curriculum setting. Inspired by the fundamental linguistic concept of mutual implication (Boghossian, 1994; Peregrin, 2006), we treat two sentences as a pair of paraphrases if they can infer each other. For example, 'I am glad to help you.' and 'Let me help you out!' can be a pair of paraphrases, which provides a diverse change of the sentence structure suitable for the curriculum setting. We also employ textual similarity for our difficulty measures for the curriculum. Higher scores indicate that two sentences are more textually similar to each other. Specifically, we treat pairs with lower scores as more difficult instances to be presented in later curriculums. We propose a paraphrase candidate generator integrated with bottom-k sampling. Traditional sampling methods such as top-k sampling (Fan et al., 2018) and top-p (Holtzman et al., 2020) sampling tend to generate easier paraphrases that have relatively high similarity scores. We propose bottom-k sampling to generate super-hard paraphrases for the later harder curriculums by pruning the most probable words.[1] This leads the generation towards a more grammatically and lexically diverse paraphrase sampling space with low textual similarity.

To mitigate catastrophic forgetting, we propose to incorporate cyclic learning to pass through the curriculums multiple times.

In summary, our proposed framework, called **PCC**: **P**araphrasing with Bottom-k Sampling and **C**yclic Learning for **C**urriculum Data Augmentation, makes three contributions:

- We exploit the use of paraphrasing with mutual implication as a data augmentation source in curriculum learning.
- To generate mutual implicative paraphrases, we propose a curriculum-aware paraphrase generation module composed of three units, namely, a paraphrase candidate generator with bottom-k sampling for generating super-hard instances, a filtering mechanism, and a difficulty measure using textual similarity.
- We propose cyclic learning to enter each curriculum multiple times.

Experimental results indicate that PCC surpasses competitive baselines on few-shot text classification as well as dialogue generation. Human evaluation indicates that bottom-k sampling effectively generates grammatically and lexically rich paraphrases, and PCC significantly improves our baseline dialogue agent. To our best knowledge, this is the first time to apply CDA on a generation task.

Takeaway Overall, we present the effectiveness of paraphrasing as a curriculum data augmentation technique. The use of cyclic learning and bottom-k sampling further boosts performance. With some modifications, future works can treat PCC as a data augmentation framework and adapt it to other downstream tasks. Future works can also leverage bottom-k sampling in generating textual outputs that are grammatically and lexically rich.

2 Related Work

2.1 Data Augmentation

Existing textual data augmentation techniques can be broadly categorized into two streams: word-level and sentence-level augmentation.

For word-level augmentation, well-known operations includes synonym replacement (Zhang et al., 2015a), random insertion, random deletion and random swap (Wei and Zou, 2019). In contrast to dictionary-based synonym replacement, another stream of works randomly replace words with masks and employs BERT models for predicting the words as a source of augmentation that exploits the contexts (Wu et al., 2019; Cai et al., 2020).

For sentence-level augmentation, Round-trip Translation (Sennrich et al., 2016b) augments translation pairs by translating from the source language into the target language, and back to the source language with two machine translation models. Gao et al. (2020) proposes to use paraphrases as a source of augmentation in task-oriented dialogue generation. It has also been proposed to retrieve from unpaired corpora as a source of augmentation in the dialogue community (Zhang et al., 2020a). Another stream of work edits the retrieved dialogue response for better generation (Cai et al., 2019a,b), which can be treated as a form of indirect augmentation. The closest work to ours is Gao et al. (2020), where theirs does not employ curriculum learning.

2.2 Curriculum Learning

While traditional curriculum learning sorts the training samples in an order of increasing

---

[1] Note that we still use a combination of top-k and top-p sampling for generating easier curriculums.

Algorithm 1: Paraphrasing with Bottom-k Sampling and Cyclic Learning for Curriculum Data Augmentation (PCC)

```
Input: Dataset $\mathcal{D}$ for the downstream task;
Output: Trained downstream task model;
For the entire dataset $\mathcal{D}$, invoke the curriculum-aware
  paraphrase generation module with $\mathcal{D}$ and cache the
  augmentation results $\bar{\mathcal{D}}$ for training purpose;
while not the end of training do
    Set difficulty level $l$ to 0 at the start of a cycle;
    while not the end of current cycle do
        while not the end of current curriculum do
            Uniformly sample the next batch of
              training instance $\mathcal{S}$;
            Invoke the curriculum-aware
              paraphrase generation module for
              each training instance in $\mathcal{S}$ to retreive
              a batch of training augmentation $\mathcal{T}$
              with difficulty level $l$.;
            Invoke the task-specific model trainer
              to train the downstream task model
              with the training augmentation $\mathcal{T}$;
        end
        Increase $l$ by 1 to the next level at the end of
          current curriculum;
    end
end
```

Algorithm 2: Curriculum-aware Paraphrase Generation Module

```
Input: A single training instance with textual input $x$;
       difficulty level $l$;
Output: Cache the generated paraphrases into $\bar{\mathcal{D}}$ or
        retrieve an augmented training instance $\bar{x}$;
if a cached augmentation exists then
    Retrieve $\bar{x}$ that corresponds to $x$ with the
      difficulty measure $d = l$;
else
    Invoke the paraphrase candidate generator
      integrated with bottom-k sampling to generate a
      bag of paraphrase candidates for $x$;
    Invoke the mutual implication classifier for each
      paraphrase candidate to obtain corresponding
      binary indicator against the input sentence;
    Calculate the textual similarity for each
      paraphrase candidate against the input;
    Filter the generated paraphrase candidates with
      the mutual implication and the textual similarity
      using Equation 3;
    Assign a difficulty measure $d$ to the filtered
      paraphrases with Equation 4;
    Cache the augmentation results into $\bar{\mathcal{D}}$ ;
end
```

difficulties (Bengio et al., 2009; Weinshall et al., 2018; Su et al., 2021), our method follows the other stream of works that applies transformation on the original data with dedicated difficulty level (Korbar et al., 2018; Ganesh and Corso, 2020; Wei et al., 2021). The closest work to ours is Wei et al. (2021). Their work does not consider paraphrasing and focuses on text classification only.

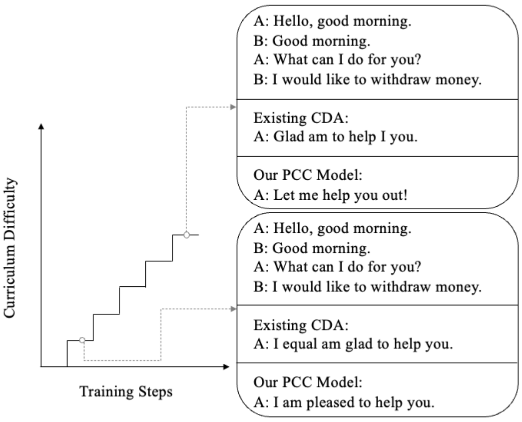
3 Our Proposed Framework

3.1 Background of Curriculum Data Augmentation (CDA)

Existing CDA (Wei et al., 2021) varies the word-level perturbation ratio to achieve different levels of difficulties under curriculum learning with simple word perturbation strategies such as synonym replacement, random insertion, swap, and deletion. As illustrated in Figure 1, such simple word perturbation strategies create problematic instances that break the sentence grammar, which can hamper the model performance. There are two common CDA strategies. One is called two-stage curriculum, which uses a fixed perturbation ratio for a single curriculum as the second stage after training with the original data. The other one is called gradual curriculum. It uses different ratios for a number of (typically 5) curriculums with increasing difficulties. However, such a learning strategy ends after passing through all the curriculums only once, and catastrophic forgetting can happen.

3.2 Our Proposed PCC

We propose curriculum data augmentation with paraphrase augmentation known as **P**araphrasing with Bottom-k Sampling and **C**yclic Learning for **C**urriculum Data Augmentation (PCC). Algorithm 1 depicts an overview of the whole PCC framework. At the start of training, we generate cached training augmentation for the entire dataset with our proposed curriculum-aware paraphrase generation module. Thereafter, we begin with the easiest curriculum. For each training instance, we retrieve the cached augmentation that has an equivalent difficulty measure with the current difficulty level. We then invoke the task-specific model trainer to train the downstream task model with the retrieved training augmentation. At the end of each curriculum difficulty level, we increase the difficulty level to advance to the next harder curriculum. In case it hits the end of the most difficult curriculum, we set the difficulty level to the easiest to start a new cycle. We propose such a cyclic learning strategy for mitigating potential catastrophic forgetting. In order to retrieve paraphrasing augmentation with appropriate difficulty measures, we propose a curriculum-aware paraphrase generation module.



A: Hello, good morning.
B: Good morning.
A: What can I do for you?
B: I would like to withdraw money.

Existing CDA:
A: Glad am to help I you.

Our PCC Model:
A: Let me help you out!

A: Hello, good morning.
B: Good morning.
A: What can I do for you?
B: I would like to withdraw money.

Existing CDA:
A: I equal am glad to help you.

Our PCC Model:
A: I am pleased to help you.

Figure 1: An illustrated example for our PCC model compared to existing CDA for dialogue generation. The original sentence is ‘I am glad to help you.’

Sample No.	**Sample Text**	**Sim. Score**
1)	I am glad to assist you.	0.888
2)	Let’s help you. I am glad to help you.	0.619
3)	Thank you for contacting me. I am glad to help you.	0.371
4)	It is now my pleasure to help you.	−0.038
5)	Let me help you out!	−0.265
6)	Thank you for your question.	−0.506

Table 1: Paraphrases with mutual implication for an input ‘I am glad to help you.’

3.2.1 Curriculum-aware Paraphrase Generation Module

Algorithm 2 depicts the curriculum-aware paraphrase generation module. Three components are designed, namely, a paraphrase candidate generator integrated with a bottom-k sampling strategy, a filtering mechanism, and a difficulty measure. The paraphrase candidates are generated and then passed to the filtering mechanism. Finally, the filtered paraphrases are assigned a difficulty measure which represents to which curriculum difficulty level the augmentation belongs.

Paraphrase Candidate Generator with Bottom-k Sampling In order to generate mutual implicative paraphrases for the purpose of curriculum data augmentation, we adopt a Seq2Seq (Sutskever et al., 2014) generator which receives an input sentence x and generates the paraphrases $\bar{x}$ in an autoregressive manner (Nighojkar and Licato, 2021). During training, the paraphrase candidate generator is trained by maximising the following likelihood:

$$P\left(\bar{x} \mid x\right)=\prod_{t=1}^{T} P\left(\bar{x}_{t} \mid \bar{x}_{1}, \ldots, \bar{x}_{t-1}, x\right),$$

where T represents the token length of the paraphrase and x_t represents the word at the position t that has been inferenced.

Traditional sampling methods such as top-k sampling (Fan et al., 2018) and top-p sampling (Holtzman et al., 2020) sample the next token to be presented in the output from the most probable vocabularies that dominate the probability distribution. For example, at the i-th timestep during inference, top-k sampling samples the next token $\bar{x}_i$ from the most probable k words with the distribution:

$$P_{\bar{x}_i \in \mathcal{V}^{(k)}}\left(\bar{x}_i \mid \bar{x}_1, \ldots, \bar{x}_{i-1}, x\right), \tag{1}$$

where $\mathcal{V}^{(k)}$ represents the most probable k words. However, they are not suitable for generating super-hard instances, i.e., their output paraphrases tend to be textually similar to the original input sentence.[2]

To avoid coping the words and unearth the super-hard paraphrases to be used in later curriculums, we propose bottom-k sampling[3] which excludes a small set of dominating words for the sampling process. Note that we still use the combination of top-k and top-p sampling to generate easier samples for earlier curriculums. Formally, bottom-k modifies the distribution in Equation 1 to:

$$P_{\bar{x}_i \in \mathcal{V} \setminus \mathcal{V}^{(k)}}\left(\bar{x}_i \mid \bar{x}_1, \ldots, \bar{x}_{i-1}, x\right), \tag{2}$$

where $\mathcal{V}$ represents the whole vocabulary. Then, at each time step, we sample the next token with the rescaled distribution in Equation 2. We apply bottom-k for the first $\mathcal{N}$ steps of the generation before fallback to top-k and top-p. Bottom-k tends to generate paraphrases with lower textual similarity. For example, given an input of ‘I like to remodel homes’, existing sampling methods can generate an output ‘Renovations in property I like to remodel homes’. In contrast, bottom-k sampling generates ‘Is this what I want to see? Renovating homes are the best choices I have ever had.’ where the latter one has a higher difficulty measure. Appendix F presents an extensive analysis.

---

[2] We found that top-k and top-p sampling tend to copy dominating words from the input into the paraphrases. This is also the reason why we prefer bottom-k over bottom-p, as we would like to effectively prevent from coping dominating words. Appendix F presents a detailed analysis.

[3] We give it such a name to make it catchy. It does not sample from the bottom k words. It samples from the bottom $|\mathcal{V}| - k$ words where $\mathcal{V}$ represents the whole vocabulary.

Paraphrase Filtering The inferential properties or mutual implication (MI) has been argued as a form of equivalent meaning (Boghossian, 1994; Peregrin, 2006), i.e., each sentence should entail each other to be 'paraphrases'. To support curriculum data augmentation, we exploit mutual implicative paraphrases for grammatical and lexical richness. Algorithm 2 (Lines 5, 6, and 7) depicts the filtering mechanism we propose to generate MI paraphrases. In order to determine the MI relationship between a pair of paraphrase $(x, \bar{x})$, we adopt a pre-trained MI classifier $\mathcal{M}(\cdot, \cdot)$ to calculate a binary indicator $\mathcal{M}(x, \bar{x})$. Here, non-MI paraphrases have a score of 0 and MI paraphrases have a score of 1. We also adopt a pre-trained model $\mathcal{G}(\cdot, \cdot)$ to evaluate the textual similarity score of the paraphrases as $\mathcal{G}(x, \bar{x})$. Here, paraphrases with lower similarity scores are treated as grammatically and lexically less similar to the original input sentence. We filter the paraphrase $\bar{x}_i$ based on these two scores:

$$\mathcal{M}(x, \bar{x}_i) + (1 - \mathcal{M}(x, \bar{x}_i))\mathbb{1}(\mathcal{G}(x, \bar{x}_i) \geq \beta). \quad (3)$$

In the formula above, β is a threshold for textual similarity. Here, a paraphrase with a positive mutual implication has a binary output of 1, i.e., it is preserved regardless of its textual similarity score. A paraphrase with a negative mutual implication but high textual similarity also has a binary output of 1, meaning it is preserved as well. In this way, MI paraphrases can be produced. We preserve highly similar paraphrases classified as non-MI, which is a misclassification by the classifier.[4] All paraphrases that are non-MI with low textual similarity have a binary output of 0, meaning we discard those paraphrases. After the filtering, a difficulty measure is computed for each paraphrase.[5]

Difficulty Measure Recall that for a pair of paraphrase $(x, \bar{x})$, we adopt a pre-trained textual similarity model $\mathcal{G}(\cdot, \cdot)$ to calculate its similarity score as $\mathcal{G}(x, \bar{x})$. BLEURT (Sellam et al., 2020) score, a BERT-based pre-trained model, is employed as the textual similarity model $\mathcal{G}(\cdot, \cdot)$. Here, paraphrases with lower similarity scores are treated as more difficult instances with higher difficulty measures. For further illustration, we present 6 samples generated from our model in Table 1 with descending order sorted on the similarity scores. Here, the similarity scores decently represent the grammatical and lexical difference between the paraphrases candidates, and the mutual implicative paraphrase candidates are grammatically (Sample 2, 3, 4, 5, and 6) and lexically (Sample 1, 2, 3, 4, 5, and 6) rich.

As the distribution of the similarity scores for the paraphrases varies for different inputs, we compute the difficulty measure for a paraphrase $\bar{x}_i$ with its rank in a sorted list of similarity scores, denoted as $\text{sort}(\cdot)$, in descending order among a bag of paraphrase candidates $\mathcal{X}$:

$$d_i = \lceil \mathcal{C} \times \frac{\text{sort}_{\bar{x}_i \in \mathcal{X}}(\mathcal{G}(\bar{x}_i, x))}{|\mathcal{X}|} \rceil, \quad (4)$$

where $\mathcal{C}$ represents the total number of curriculum difficulty levels we define, and $|\mathcal{X}|$ represents the total number of paraphrase candidates we have. Here, the paraphrase $\bar{x}_j$ with the highest similarity score, i.e., $\mathcal{G}(x, \bar{x}_j) = \max_{\bar{x}_i \in \mathcal{X}} (\mathcal{G}(\bar{x}_i, x))$, has a rank of 1, therefore, $d_j = 1$. The paraphrase $\bar{x}_k$ with the lowest similarity score, i.e., $\mathcal{G}(x, \bar{x}_k) = \min_{\bar{x}_i \in \mathcal{X}} (\mathcal{G}(\bar{x}_i, x))$, has a rank of $|\mathcal{X}|$, thus $d_k = \mathcal{C}$. Consequently, a larger rank indicates that the paraphrase is more grammatically and lexically different than the original input, and thus belongs to a harder curriculum. We set $d_i = 0$ as the easiest difficulty level for the original data.

3.2.2 Cyclic Curriculum Data Augmentation

Wei et al. (2021) proposed curriculum data augmentation with a gradual course. The training ends after passing the curriculums once. We found that a typical problem called catastrophic forgetting (Kirkpatrick et al., 2017) can hamper the performance during such a gradual course, meaning that the model can gradually forget the knowledge learned in an easier course. The augmentation for later curriculums is a subtask of an easier curriculum and can have lexical overlaps. Formally, the input samples x^{t+1} can have overlapping lexical x_i^t which are the same as x_j^t, where t and $t+1$ represent the curriculum difficulty levels, and i and j represent the word positions in the sentence. Due to catastrophic forgetting, the model can forget what it has learned earlier. Hence, we propose cyclic learning as shown in Algorithm 1 to inform the model which

---

[4] We postulate it as a flaw introduced by the imbalanced training data with a larger portion of paraphrases that tends to be textually unsimilar against the original sentence. We found in our early experiments that removing these easier examples obviously degrades the results for COVID-Q from 51.7 to 50.0. Furthermore, ignoring non-MI easy examples prevents PCC from collecting enough augmentation for AMZN.

[5] As in Appendix A, we use an off-the-shelf paraphrase generator and MI classifier in our experiments.

skills would be useful later before retrospecting to easier curriculums with lower difficulties.

4 Experimental Setup

In our experiments, we define six curriculums ranging from 0 to 5. 0 represents the original data, and 1 and 5 represent the easiest and the most difficult curriculum respectively.[6]

4.1 Few-shot Text Classification Task

For the downstream application task for our experiments, we follow Wei et al. (2021) to conduct the task of few-shot, highly multi-class text classification (Gupta et al., 2014; Kumar et al., 2019), which typically has a large number of classes with only a few samples for each of the class. We use triplet loss, a loss computed with three elements, namely, an anchor a, a positive sample p, and a negative sample n. It origins from the vision community (Schroff et al., 2015), which was later applied to language tasks (Ein Dor et al., 2018; Lauriola and Moschitti, 2020), suitable for the few-shot setting. Precisely, the learning objective is defined as:

$$\mathcal{L} = \mathcal{D}(a, p) - \mathcal{D}(a, n) + \gamma,$$

where $\mathcal{D}$ represents a distance measure that computes the distance between the input encodings. γ represents the margin between the positive and negative samples. We use BERT-based (Devlin et al., 2019) pooled sentence encodings as the input into a two-layer triplet network (Schroff et al., 2015).

Three datasets for the text classification task are used in our experiments, namely, HUFFPOST (Misra, 2018; Misra and Grover, 2021), COVID-Q (Wei et al., 2020), and AMZN (Yury, 2020). For space reasons, we leave their detailed dataset description in Appendix B.

4.2 Dialogue Generation Task

The second downstream task for our experiments is open-domain dialogue generation. We adopt a Seq2Seq neural network (Sutskever et al., 2014) which receives a text concatenation of prepended knowledge k and dialogue context c and generates the dialogue response r in an autoregressive manner (Radford, 2018). We train our dialogue generator by maximising the following likelihood:

$$P(r \mid k, c) = \prod_{t=1}^{T} P(r_t \mid r_1, ..., r_{t-1}, k, c),$$

where T represents the length of the generated dialogue response and r_t represents the word at the position t that has been inferenced. Typical prepended knowledge include personal traits (Zhang et al., 2018) and movie description (Zhou et al., 2018). We use DialoGPT (Zhang et al., 2020b) for parameter initialization for PCC.

We use PERSONACHAT (CONVAI2, Zhang et al. 2018) as the dataset for dialogue generation, which is described in Appendix C.

4.3 Baselines for Text Classification

We use the following baselines from existing data augmentation methods for text classification.

Triplet Loss As described in Section 4.1, an anchor, a positive example and a negative example is selected to construct the loss (Schroff et al., 2015).

Token Substitution It substitutes words with their WordNet synonyms (Zhang et al., 2015b; Feinerer and Hornik, 2020).

Pervasive Dropout It uses dropout on words with probability $p = 0.1$ (Sennrich et al., 2016a).

SwitchOut It replaces words with uniformly sampled words (Wang et al., 2018).

Round-trip Translation It translates sentences into another language before translating back into the source language (Sennrich et al., 2016b).

Hard Negative Mining + EDA It combines hard negative mining (Schroff et al., 2015) that chooses hard negative samples and EDA (Wei and Zou, 2019) that employs synonym replacement, word-level random insertion, deletion, and swap.

Hard Negative Mining + EDA + Gradual Curriculum It gradually increases the temperature for EDA augmentation (Wei et al., 2021).

4.4 Baselines for Dialogue Generation

We use the following baselines and data augmentation methods for dialogue generation.

TransferTransfo A Transformer-based model fine-tuned on PERSONACHAT (Wolf et al., 2019).

PerCVAE It uses a memory-augmented architecture with a conditional variational autoencoder to exploit persona information (Song et al., 2019).

DialoGPT It refers to an autoregressive dialogue generator introduced by Zhang et al. (2020b).

---

[6] We release the code and resource at `https://github.com/HongyuanLuke/PCC`.

CDA It refers to the curriculum data augmentation technique proposed by Wei et al. (2021) using the augmentation of EDA (Wei and Zou, 2019).

Official & Flatten It refers to the paraphrase augmentation technique that is task-specific to the task-oriented dialogue generation (Gao et al., 2020). To adapt it to our task, we use our generated paraphrase via mutual implication, denoted as Flatten, and the official revised PERSONACHAT paraphrases, denoted as Official.

Round-trip Translation It translates the input into another language before translating back (Sennrich et al., 2016b).

4.5 Evaluation Metrics

For the text classification task, we follow Wei et al. (2021) to use the top-1 accuracy as the metric.

For the dialogue generation task, we use the word-level F1 score, and we adopt the well-known sequence evaluation metric BLEU (Papineni et al., 2002) where we report BLEU-2, BLEU-3 and BLEU-4. We also adopt another well-known sequence evaluation metric, ROUGE, where we report the F-measures for ROUGE-1, ROUGE-2 and ROUGE-L (Lin, 2004).

To verify our claim that bottom-k sampling generates grammatically and lexically rich paraphrases, we adopt Distinct-N (Li et al., 2016; Gao et al., 2019) with both $N \in \{1, 2, 3\}$ and $N \in \{4, 5, 6\}$ to measure the lexical and grammatical richness respectively using the ratio of distinct N-grams against the total number of N-grams generated.

5 Results and Analysis

5.1 Few-shot Text Classification Results

5.1.1 Main Results

Table 2 presents the results for few-shot text classification. Among the baselines, Triplet Loss + Gradual Curriculum works the best (Wei et al., 2021). PCC improves this baseline significantly. All the models share randomness in data, and our model is the best on all of the random seeds individually. Further, our proposed PCC model surpasses the baselines of Token Substitution, Pervasive Dropout, SwitchOut and Round-trip Translation significantly. Without bottom-k, PCC surpasses all the baselines, and our proposed full model with bottom-k obviously boosts performance. Appendix G additionally presents an analysis of the improvements as a function of the number of data augmentations.

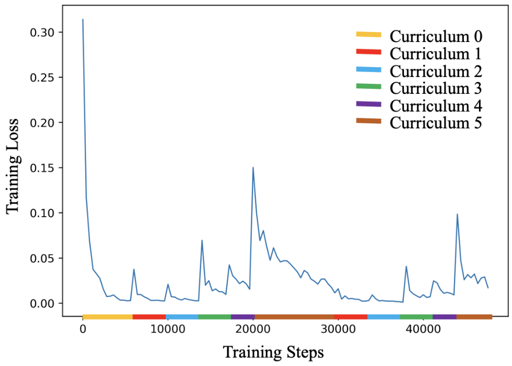


Figure 2: A plot of the training loss for the analysis for cyclic learning. Best viewed in color.

5.1.2 Ablation Study

Table 4 presents the results of our ablation study. First, removing the MI paraphrase filtering component described with Equation 3 obviously degrades the results. Replacing bottom-k sampling with pure sampling also decreases the results. Furthermore, paraphrasing in a random or an inverse order of decreasing difficulties, i.e., with neither curriculum learning nor cyclic learning, obviously deteriorates the results. *Therefore, our contribution is the discovery of paraphrasing as an effective CDA method rather than using paraphrasing solely as an augmentation technique.* Moreover, using cyclic learning instead of the gradual curriculum improves the results when trained with and without bottom-k sampling. Training the second cycle in an inversed order of decreasing difficulties degrades the results both with and without bottom-k.

5.2 Analysis on Cyclic Learning

Figure 2 presents the change of the training loss during the progress of the training on the task of text classification on COVID-Q. We observe that catastrophic forgetting exists as the training loss spikes when re-entering the curriculums. For the second time it enters the most difficult curriculum 5, the loss is also further smoothened compared to the first spike. The spike is also desirable as described in Wei et al. (2021), indicating that new instances that are harder to learn are presented and can help to escape the local minima. These support the usefulness of our proposed cyclic learning that can smoothen the gradients, mitigate catastrophic forgetting, and improve generalization by entering curriculums multiple times.

Model	HuffPost	Covid-Q	AMZN	**Average**
Triplet Loss (Schroff et al., 2015)	20.9 ± 1.0	39.7 ± 1.0	11.6 ± 0.6	24.1
Triplet Loss + Token Substitution (Zhang et al., 2015b)	22.7 ± 1.4	43.9 ± 1.3	12.8 ± 0.7	26.5
Triplet Loss + Pervasive Dropout (Sennrich et al., 2016a)	23.1 ± 1.1	43.5 ± 1.8	13.0 ± 0.6	26.5
Triplet Loss + SwitchOut (Wang et al., 2018)	22.9 ± 0.5	41.5 ± 0.6	12.7 ± 0.8	25.7
Triplet Loss + Round-trip Translation (Sennrich et al., 2016b)	24.2 ± 0.7	42.3 ± 1.0	13.0 ± 0.4	26.5
Triplet Loss + Hard Negative + EDA (Wei and Zou, 2019)	22.6 ± 1.8	48.2 ± 0.9	13.7 ± 0.9	28.2
↪ + Gradual Curriculum (Wei et al., 2021)	23.8 ± 0.9	48.9 ± 0.9	14.4 ± 1.5	29.0
PCC with Cyclic Curr. w/o Bottom-k	25.2 ± 1.5	51.4 ± 0.8	17.4 ± 0.7	31.3
PCC with Cyclic Curr. w/ Bottom-k	**25.9 ± 1.7**	**51.7 ± 0.6**	**18.2 ± 1.0**	**31.9**

Table 2: Results in top-1 accuracy for the downstream task of text classification on three datasets. The best results are bolded. We report the results averaged from five random seeds for data selection ranging from 0 to 4, which is the source of the variance here. Our methods report the best performance on all the random data seeds on all the datasets. A combination of top-k and top-p sampling with $k = 120$ and $p = 0.95$ is used for the penultimate row.

Model	**F1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
TransferTransfo (Wolf et al., 2019)	16.61 ± 0.09	3.16 ± 0.07	1.04 ± 0.03	0.43 ± 0.02	17.69 ± 0.14	3.96 ± 0.08	16.34 ± 0.13
PerCVAE (Song et al., 2019)	14.33 ± 0.12	1.23 ± 0.06	0.20 ± 0.05	0.04 ± 0.01	13.25 ± 0.10	1.62 ± 0.05	12.02 ± 0.10
DialoGPT (Zhang et al., 2020b)	18.58 ± 0.13	5.25 ± 0.08	1.89 ± 0.07	0.66 ± 0.05	18.42 ± 0.13	4.62 ± 0.09	17.23 ± 0.12
DialoGPT + CDA (Wei and Zou, 2019)	18.38 ± 0.10	5.23 ± 0.10	1.84 ± 0.08	0.63 ± 0.02	18.55 ± 0.31	4.63 ± 0.11	17.40 ± 0.30
DialoGPT + Flatten (Gao et al., 2020)	18.21 ± 0.21	5.03 ± 0.18	1.85 ± 0.11	0.65 ± 0.04	17.97 ± 0.34	4.45 ± 0.16	16.84 ± 0.28
DialoGPT + Official (Gao et al., 2020)	18.12 ± 0.11	4.80 ± 0.27	1.78 ± 0.50	0.59 ± 0.60	17.88 ± 0.24	4.38 ± 0.09	16.84 ± 0.20
DialoGPT + RT (Sennrich et al., 2016b)	18.26 ± 0.49	5.10 ± 0.21	1.80 ± 0.20	0.62 ± 0.08	18.32 ± 0.35	4.47 ± 0.18	17.16 ± 0.31
PCC with Cyclic Curr. w/o Bottom-k	18.76 ± 0.20	5.38 ± 0.14	1.99 ± 0.9	0.71 ± 0.06	18.81 ± 0.18	4.75 ± 0.12	17.53 ± 0.12
PCC with Cyclic Curr. w/ Bottom-k	**18.80 ± 0.45**	**5.59 ± 0.17**	**2.07 ± 0.12**	**0.76 ± 0.11**	**19.15 ± 0.16**	**4.98 ± 0.12**	**17.89 ± 0.17**

Table 3: Results for the downstream task of open-domain dialogue generation on PersonaChat, averaged from three runs. All the metrics attain better quality with higher scores. We denote Round-trip Translation as RT. A combination of top-k and top-p sampling with $k = 120$ and $p = 0.95$ is used for the penultimate row.

Model	HuffPost	Covid-Q	AMZN
PCC w/o MI filtering	25.7 ± 1.4	50.2 ± 1.7	16.7 ± 1.1
PCC w/ Pure Sampling	25.8 ± 1.0	49.7 ± 0.9	16.9 ± 0.8
PCC w/ Inverse Curriculum	23.0 ± 1.7	48.5 ± 1.2	15.0 ± 0.5
PCC w/ Random Curriculum	24.0 ± 1.7	48.9 ± 1.5	15.1 ± 0.8
PCC w/ Gradual Curriculum	24.7 ± 1.3	49.6 ± 1.4	16.5 ± 0.7
PCC w/ Inv. Cyc.	24.9 ± 1.2	50.9 ± 1.0	16.5 ± 0.8
PCC w/ Cyc.	25.2 ± 1.5	51.4 ± 0.8	17.4 ± 0.7
PCC w/ Inv. Cyc., Bottom-k	25.3 ± 1.9	51.3 ± 1.1	17.1 ± 1.2
PCC w/ Cyc., Bottom-k	**25.9 ± 1.7**	**51.7 ± 0.6**	**18.2 ± 1.0**

Table 4: Ablation results in top-1 accuracy for the downstream task of text classification.

5.3 Dialogue Generation Results

Table 3 presents the results for dialogue generation on PersonaChat. First, we present the results for competitive baselines, namely TransferTransfo and PerCVAE. DialoGPT surpasses these two significantly. Using CDA on DialoGPT has deteriorated BLEU scores, which suggests that using CDA causes grammatical influence, possibly due to the random operations that produce undesirable grammatically incorrect augmentation. We also observe a large variance with the official paraphrase provided by PersonaChat, possibly due to the large difference between the manually rephrased sentences. This indicates easier paraphrases seem to be essential for PCC to be effective. Also, the Flatten baseline reported in Table 3 approximates a random curriculum, which degrades the results. It leads to a conclusion about the usefulness of the suggested curriculum. Round-trip Translation (RT) seems not effective, which is somehow reasonable as RT was originally designed for machine translation. PCC achieves the best among all the models, suggesting its usefulness for dialogue generation. Appendix D provides in-depth reasonings on the results. Appendix H presents a human evaluation of the downstream task of dialogue generation.

5.4 Analysis on Bottom-k Sampling

Table 5 presents the automatic results for bottom-k sampling on PersonaChat. Here, bottom-k sampling attains the best on Distinct scores with

Model	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**
Pure Sampling	0.187	0.571	0.788	0.881	0.919	0.932
Top-k&p (k=120, p=0.95)	0.145	0.481	0.711	0.826	0.877	0.897
Top-k&p (k=80, p=0.80)	0.125	0.415	0.634	0.762	0.825	0.850
Bot.-k (k=2, $\mathcal{N}$=1)	0.184	0.587	0.824	0.901	0.919	0.925
Bot.-k (k=10, $\mathcal{N}$=1)	0.199	0.630	0.860	0.926	0.940	0.943
Bot.-k (k=2, $\mathcal{N}$=5)	0.223	0.695	0.904	0.945	0.951	0.953
Bot.-k (k=5, $\mathcal{N}$=10)	0.251	0.786	0.950	0.967	0.969	0.970
Bot.-k (k=10, $\mathcal{N}$=15)	**0.262**	**0.851**	**0.971**	**0.978**	**0.979**	**0.979**

Table 5: Automatic results for bottom-k sampling on PERSONACHAT. **D** represents the Distinct-N scores.

Criteria	**PCC w/o Bottom-k**	**PCC w/ Bottom-k**
Gramma. Richness	34	66 ‡
Lexical Richness	33	67 ‡
Difficulty	34	66 ‡
Paraphrasing	50	50

Table 6: Human evaluation results for bottom-k in winning percentages. ‡ indicates the results as passing a two-tailed binomial significance test with $p < 0.0001$.

lower grams ($N \in \{1, 2, 3\}$), indicating its lexical richness. It also attains the best on Distinct scores with higher grams ($N \in \{4, 5, 6\}$), indicating its grammatical richness. This helps to generate super-hard instances. Note that the setting of bottom-k sampling employed in PCC with $k = 2$ and $\mathcal{N} = 1$ already gives the best overall diversity against previous sampling methods. Further increasing the value of k and $\mathcal{N}$ leads to higher diversity.

5.5 Human Evaluation on Bottom-k Sampling

We hired three experienced annotators who have degrees relevant to English Linguistics to conduct an evaluation on bottom-k sampling with PERSONACHAT. We present a questionnaire composed of 800 questions with 200 randomly sampled training instances with the paraphrases generated with and without bottom-k sampling to the annotators to compare model outputs under A/B testing:

- **(Grammatical Richness)**: *"Which paraphrase do you think is more grammatically different than the original input sentence?"*
- **(Lexical Richness)**: *"Which paraphrase do you think is more lexically different than the original input sentence?"*
- **(Difficulty)**: *"Which paraphrase is more difficult to read and understood?"*
- **(Paraphrasing)**: *"Which one is more like a mutual implicative paraphrase to the input?"*

Table 6 presents the results of our human evaluation. The paraphrases generated by PCC with bottom-k sampling have a significant advantage in lexical and grammatical richness. Such an advantage correlates well with the difficulty of the paraphrases to be understood by human annotators. Furthermore, bottom-k does not hurt the paraphrasing performance compared to the top-k and top-p sampling. The result of human evaluation verifies our claim that bottom-k generates super-hard paraphrases with grammatical and lexical richness. Appendix F presents how bottom-k sampling is superior over previous methods in our scenario with case studies about the coping mechanism.

6 Conclusions

We propose a novel framework that uses mutual implicative paraphrasing as a curriculum data augmentation technique. Our proposed curriculum-aware paraphrase generation module is composed of three components, a paraphrase candidate generator with a bottom-k sampling strategy for generating superhard paraphrases, a paraphrase filtering mechanism, and a difficulty measure. We propose a bottom-k sampling strategy to effectively generate super-hard instances with grammatical and lexical richness to be used for the later stages in curriculum learning. Moreover, we propose a cyclic learning strategy that mitigates catastrophic forgetting. Experimental results on the task of few-shot text classification as well as dialogue generation support our proposed methodology PCC's usefulness, surpassing several competitive baselines.

Limitations

The proposed PCC cost more computational resources than traditional CDA methods. However, the cost is still affordable. Generating a round-trip augmentation used as one of the baselines costs about 1.5 seconds (1x speed) for PERSONACHAT. In contrast, generating a single paraphrase costs about 0.40 seconds (3x faster) with PCC on our machine with a single GPU.

Ethical Statement

We honour and support the EACL Code of Ethics. The datasets used in this work are well-known and widely used, and the dataset pre-processing does not make use of any external textual resource. In our view, there is no known ethical issue. End-to-end pre-trained dialogue generators are also used, which are subjected to generating offensive context. But the above-mentioned issues are widely known to commonly exist for these models. Any content generated do not reflect the view of the authors.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*, pages 41–48.

Paul A. Boghossian. 1994. Inferential role semantics and the analytic/synthetic distinction. *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition*, 73(2/3):109–122.

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. 2019a. Skeleton-to-response: Dialogue generation guided by retrieval memory. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1219–1228, Minneapolis, Minnesota. Association for Computational Linguistics.

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, and Shuming Shi. 2019b. Retrieval-guided dialogue response generation via a matching-to-generation framework. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1866–1875, Hong Kong, China. Association for Computational Linguistics.

Hengyi Cai, Hongshen Chen, Yonghao Song, Cheng Zhang, Xiaofang Zhao, and Dawei Yin. 2020. Data manipulation: Towards effective instance learning for neural dialogue generation via learning to augment and reweight.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Liat Ein Dor, Yosi Mass, Alon Halfon, Elad Venezian, Ilya Shnayderman, Ranit Aharonov, and Noam Slonim. 2018. Learning thematic similarity metric from article sections using triplet networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Ingo Feinerer and Kurt Hornik. 2020. *wordnet: WordNet Interface*. R package version 0.1-15.

Madan Ganesh and Jason Corso. 2020. Rethinking curriculum learning with incremental labels and adaptive compensation.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, and Shuming Shi. 2019. Generating multiple diverse responses for short-text conversation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6383–6390.

Silin Gao, Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Paraphrase augmented task-oriented dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 639–649, Online. Association for Computational Linguistics.

Maya R. Gupta, Samy Bengio, and Jason Weston. 2014. Training highly multiclass classifiers. *J. Mach. Learn. Res.*, 15(1):1461–1492.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Bruno Korbar, Du Tran, and Lorenzo Torresani. 2018. Cooperative learning of audio and video models from self-supervised synchronization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7774–7785, Red Hook, NY, USA. Curran Associates.

Sawan Kumar, Shweta Garg, Kartik Mehta, and Nikhil Rasiwasia. 2019. Improving answer selection and answer triggering using hard negatives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5911–5917, Hong Kong, China. Association for Computational Linguistics.

Ivano Lauriola and Alessandro Moschitti. 2020. Context-based transformer models for answer sentence selection.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Margaret Li, Jason Weston, and Stephen Roller. 2019. ACUTE-EVAL: Improved Dialogue Evaluation with Optimized Questions and Multi-turn Comparisons. *CoRR*, abs/1909.03087.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Cao Liu, Shizhu He, Kang Liu, and Jun Zhao. 2018. Curriculum learning for natural answer generation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4223–4229. International Joint Conferences on Artificial Intelligence Organization.

Jinglin Liu, Yi Ren, Xu Tan, Chen Zhang, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. Task-level curriculum learning for non-autoregressive neural machine translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3861–3867. International Joint Conferences on Artificial Intelligence Organization. Main track.

Alexander H. Miller, Will Feng, Adam Fisch, Jiasen Lu, Dhruv Batra, Antoine Bordes, Devi Parikh, and Jason Weston. 2017. ParlAI: A Dialog Research Software Platform. *CoRR*, abs/1705.06476.

Rishabh Misra. 2018. News category dataset.

Rishabh Misra and Jigyasa Grover. 2021. *Sculpting Data for ML: The first act of Machine Learning*.

Animesh Nighojkar and John Licato. 2021. Improving paraphrase detection with the adversarial paraphrasing task. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7106–7116, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jaroslav Peregrin. 2006. Meaning as an inferential role. *Erkenntnis (1975-)*, 64(1):1–35.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Alec Radford. 2018. Improving language understanding by generative pre-training.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 371–376, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Haoyu Song, Wei-Nan Zhang, Yiming Cui, Dong Wang, and Ting Liu. 2019. Exploiting persona information for diverse generation of conversational responses. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5190–5196. International Joint Conferences on Artificial Intelligence Organization.

Yixuan Su, Deng Cai, Qingyu Zhou, Zibo Lin, Simon Baker, Yunbo Cao, Shuming Shi, Nigel Collier, and Yan Wang. 2021. Dialogue response selection with hierarchical curriculum learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1740–1751, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models. *arXiv e-prints*, abs/1610.02424.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods* *in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Jason Wei, Chengyu Huang, Soroush Vosoughi, Yu Cheng, and Shiqi Xu. 2021. Few-shot text classification with triplet networks, data augmentation, and curriculum learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5493–5500, Online. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Jerry Wei, Chengyu Huang, Soroush Vosoughi, and Jason Wei. 2020. What Are People Asking About COVID-19? A Question Classification Dataset. *CoRR*, page CoRR:2005.12522.

Daphna Weinshall, Gad Cohen, and Dan Amir. 2018. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5238–5246. PMLR.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A Transfer Learning Approach for Neural Network Based Conversational Agents. *CoRR*, abs/1901.08149.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *ICCS*.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020a. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online. ACL.

Chen Xu, Bojie Hu, Yufan Jiang, Kai Feng, Zeyang Wang, Shen Huang, Qi Ju, Tong Xiao, and Jingbo Zhu. 2020b. Dynamic curriculum learning for low-resource neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3977–3989, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kashnitsky Yury. 2020. Hierarchical text classification of amazon product reviews.

Rongsheng Zhang, Yinhe Zheng, Jianzhi Shao, Xiaoxi Mao, Yadong Xi, and Minlie Huang. 2020a. Dialogue distillation: Open-domain dialogue augmentation using unpaired data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3449–3460, Online. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Australia. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 649–657, Cambridge, MA, USA. MIT Press.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020b. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

Yicheng Zou, Zhihua Liu, Xingwu Hu, and Qi Zhang. 2021. Thinking clearly, talking fast: Concept-guided non-autoregressive generation for open-domain dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2215–2226, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Implementation Details

For text classification, we use the hyper-parameter settings as in Wei et al. (2021) for the gradual course, and we refer to their paper for the detailed settings. For our cyclic learning, we pass through the curriculums twice. We train the same number of steps for each curriculum as we did in the first pass for our second pass, and the remaining hyper-parameters are kept the same. For Token Substituion, Pervasive Dropout, SwitchOut, and Round-trip Translation, we follow Wei et al. (2021) to use the triplet network as the base model and use a two-stage curriculum for those baselines. Following Wei et al. (2021), we include 20% original data whenever augmentation is used.

For dialogue generation, we use DIALOGPT-SMALL for parameter initialisation. We use a batch size of 4 and a gradient clip of 0.1. We use validation patience of 10 based on the validation loss. We use greedy decoding for all of our experiments. The above settings apply to all our baselines and our proposed model fine-tuned on DIALOGPT. We start to apply the augmentation after 130,000 steps for data augmentation methods. We train the first, second, third, fourth, and fifth curriculums with 60,000 steps. For Official, Flatten, and RT, we perform a two-stage curriculum as described by Wei et al. (2021). We set $\mathcal{N}$ and k as a small value (typically $\mathcal{N} = 1$ and $k = 2$) for bottom-k sampling. We perform a cyclic repetition for our proposed method for the same number of steps for each curriculum until early stopped.

During our experiments, we apply data augmentation methods on the entire textual input for text classification, and we apply data augmentation methods on the personas traits for persona-based dialogue generation. We employ an off-the-shelf pre-trained model for both the paraphrase generator and the MI classifier (Nighojkar and Licato, 2021).

For all of the datasets, we obtain 20 paraphrases after filtering, and we assign 4 paraphrases (Wei et al., 2021) to each of the curriculums we have. We use 2 paraphrases obtained with bottom-k sampling for COVID-Q and we use 4 paraphrases obtained with bottom-k sampling for the remaining datasets.

For our models without bottom-k sampling, we use 20 paraphrases generated with a combination of top-k sampling and top-p sampling with $k = 120$ and $p = 0.95$ for all of the datasets.

We conduct our experiments for dialogue generation on the PARLAI platform (Miller et al., 2017).

B Datasets for Text Classification

- The HUFFPOST dataset is composed of 200k news headlines collected from 2012 to 2018, which is categorized into 41 classes such as politics, entertainment, and travel (Misra, 2018; Misra and Grover, 2021). We use all the classes and a 70% / 30% train / test split by class (Wei et al., 2021).

- The COVID-Q dataset is composed of 87 classes with several questions per cluster which ask about the same thing (Wei et al., 2020). We use the official train / test split with 3 questions per cluster (Wei et al., 2021).

- The AMZN product review dataset (Yury, 2020) categorizes products into given reviews. We consider the use of 318 'level-3' classes with at least 6 samples per product.

For the few-shot scenario, we need to set the number of samples in each class, N_c, to be used to construct the datasets. We use the setting in Wei et al. (2021) where $N_c = 3$ for COVID-Q and $N_c = 10$ for HUFFPOST. We set $N_c = 2$ for AMZN.

C Dataset for Dialogue Generation

CONVAI2 is an official competition built based on PERSONACHAT by adding new training examples as well as a hidden test set. For convenience, we denote the former as PERSONACHAT in the remaining of the paper. Since the test set is not publicly available, we use the official split containing a training / development split with 8,939 / 1,000 multi-turn dialogues conditioned on 1,155 / 100 personas respectively. Each persona is composed of about 4 to 5 persona traits.

D Analysis on Dialogue Generation

Table 3 reports an ablation when we use our PCC to train the dialogue generator without the use of bottom-k sampling. The results suggest that using bottom-k sampling improves all the metrics, especially the ROUGE scores. Table 8 presents the distribution of the textual similarity scores for the paraphrases generated from four methods on PERSONACHAT. The official paraphrase (Zhang et al., 2018) largely differs from the original ones, which we postulate as the reason for the large variance observed in Table 3. This also indicates the neccessity of the easier samples for curriculum learning. The Round-trip Translation generates paraphrases that have higher textual similarity with the input sentence. Our method without bottom-k sampling (we use a combination of top-k and top-p sampling with $k = 120$ and $p = 0.95$ here) generates paraphrases with more evenly distributed scores, with an average of 0.02. In contrast, bottom-k helps to generate harder samples while still capable of generating more easier samples.

E Problematic Cases for EDA

Table 7 presents samples from EDA for a sample input 'I am glad to help you.' with each of the

Sample Number	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
i)	I equal am glad to help you.	I am glad help to you.	I am gald to happy help you.	To help you.	Glad am to help I you.
ii)	I am glad you help to.	I am gladiola to help you.	I am glad to assistance you.	Help glad am to i you.	I am gladiolus to helper you.
iii)	Am glad you.	I am glad help you.	I am glad you help to.	You I gald to help am.	I am glad help you.
iv)	I am glad to help you.	I am glad equal to help you.	I am glad to help you.	I am glad to happy happy help you.	I am happy to avail you.

Table 7: Randomly selected cases for an input ‘I am glad to help you.’ using Easy Data Augmentation (Wei and Zou, 2019). We present recommended temperatures τ ranging from 0.1 to 0.5, with four samples for each τ.

Model	$[0.5,$	$[0, 0.5)$	$(-0.5, 0)$	$, -0.5]$	**Avg.**
Official Paraphrases	1%	14%	33%	52%	-0.46
Round-trip Translation	25%	52%	17%	6%	0.23
PCC w/o Bottom-k	39%	11%	23%	27%	0.02
PCC w/ Bottom-k	16%	8%	18%	58%	-0.43

Table 8: Analysis on the distribution for the textual similarity score with different augmentation methods.

temperatures τ ranging from 0.1 to 0.5, which is the recommended setting from Wei et al. (2021). We categorize EDA’s problems as the followings:

- Sample i) with $\tau = 0.1$ and sample ii) with $\tau = 0.2$ changes the meaning of the input sentence. ‘equal’ is possibly produced by random insertion and ‘gladiola’ is possibly produced by synonym replacement via WordNet (Feinerer and Hornik, 2020).
- Most of the samples produced with $\tau = 0.4$ and $\tau = 0.5$ breaks the grammar, which can be harmful to generation tasks.
- Sample ii) and iv) with $\tau = 0.5$ introduces rare words such as ‘avail’ and ‘gladiolus’, which is counterintuitive to see in many tasks.

As illustrated in Figure 1, PCC effectively reduces the above-mentioned issues.

F Analysis on Bottom-k Sampling

Table 9 presents extensive case studies to support that bottom-k sampling generates grammatically rich and lexically rich paraphrases. PCC without bottom-k tends to exploit a coping mechanism at the beginning of generation (Sample 2, 3, 5, 6, 7, 8, 9, 10, 11, 12). By excluding these dominating words to be copied for generation, bottom-k effectively emphasises the content (Sample 5), improves grammatical richness (Sample 1, 2, 3, 4, 5, 6, 7, 10, 12) and lexical richness (Sample 3, 4, 6, 8, 10, 12), does appropriate synonym replacement (Sample 8, 11) and insertion (Sample 4). Without bottom-k sampling, the input that starts with a first-person pronoun ‘i’ is highly likely to have an output that starts with ‘i’ (Sample 2, 3, 6, 8, 10). This constrains the model from generating grammatically rich paraphrases. In contrast, bottom-k sampling effectively reduces such cases and biases the generation towards a grammatically rich sampling space. Indeed, out of the 6,126 persona traits from PERSONACHAT, 5,087 of them start with ‘i’. PCC without bottom-k generates 2,558 paraphrases that start with ‘i’, which avoids generating super-hard instances and hampers the PCC performance.[7] In contrast, bottom-k generates 205 paraphrases that start with ‘i’, indicating its usefulness in improving grammatical richness and generating super-hard instances. Avoiding coping helps to unearth the diverse paraphrases hidden in the tail vocabularies,

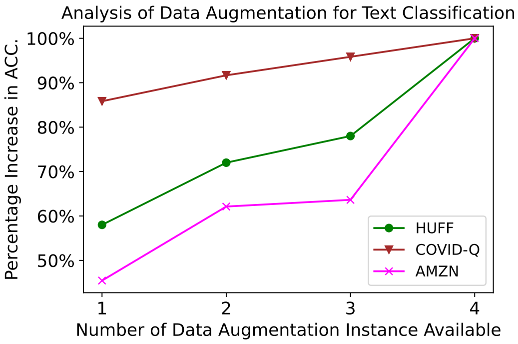


Figure 3: A plot of the percentage performance improvements of the downstream task of text classification against the number of data augmentation instances per curriculum. We use the first row in Table 2 as the baseline and the last row in Table 2 as the full improvements.

---

[7] Neither a pure top-p sampling with $p = 0.95$, a pure top-k sampling with $k = 120$, nor a greedy sampling helps, which generates 2,313, 2,381 and 3,302 paraphrases that start with ‘i’ respectively. Compared to these sampling strategies, bottom-k is an effective strategy in preventing copying.

Original Input Sentence	**PCC w/o Bottom-k Sampling**	**PCC w/ Bottom-k Sampling**
1): i like to shoot a bow.	When i first started shooting bows, this was the most important method.	Hey, i like to shoot a bow. Just started using a Bow SLR shooter, but a DSLR isn't really necessary.
2): i have four sisters.	i have four sisters	four sisters, and i want four sisters.
3): i believe that mermaids are real.	i believe that mermaids are real " @JesseyHawkins	Marxist philosopher,'mermaids are real," property
4): i work as a stand up comedian.	jesse t	trained comedian, I work as a stand up comedian.
5): my favorite drink is cuba libre.	My favorite beverage is Cuba libre.	Cuba is my favorite drink and I live in Cuba free.
6): i did a few small roles in tv series.	I have done a few small roles in tv series.	over the years i've appeared in a few small roles in television series
7): i love bicycling.	bicycle is my friend. i Love Bicycling.	how wonderful & amp ; inspiring! I love bicycling.
8): i own a hearse.	own a hearse. u could do that?	belongs to a hearse. it's not that expensive.
9): i like to listen to music.	i like to listen to music. How do you make up your mind?	I like to listen to music. by JACK CLINTON
10): i like to party.	I like to party	touts my ambition and passion for parties " by @MargotHillary by @anadulka @KelisStout
11): my favorite band is imagine dragons.	my favorite band is imagine dragons. I am just so happy about that.	i love this band it is awesome
12): i love to sing.	sing, am i love to sing	artist, i love to sing.

Table 9: Extensive case studies on PERSONACHAT support our claim that bottom-k sampling generates grammatically and lexically rich paraphrases that are more different than the input sentence.

which we postulate as the reason for the results observed in human evaluation in Section 5.5.

Note that we use bottom-k sampling to effectively prevent coping to generate instances that are textually more different to the input. There is a stream of work that considers improving the diversity (Vijayakumar et al., 2016). However, these works do not directly consider the similarity between the input paraphrase and the output paraphrase. This is the advantage of bottom-k sampling over this stream of work for our scenario.

G Analysis on Data Augmentation

Figure 3 presents the percentage improvements in accuracy as a function of the number of data augmentation instances available for each curriculum. Here, since we have 5 curriculum difficulty levels in our setting, having 3 instances available for each curriculum means that we have 15 data augmentations in total for each original sample. The improvements are positively correlated with the number of available instances. Furthermore, it seems that the improvements of PCC are not saturated yet. This means that a further increase in the number of data augmentations can lead to even higher performance than reported in our paper.

H More Human Evaluation

- **(Appropriateness)**: *"Who is more appropriate given the previous dialogue context?"*
- **(Informativeness)**: *"Who is more diverse instead of null answers such as I do not know?"*
- **(Engagingness)**: *"Who would you prefer to talk with for a long conversation?"*
- **(Human-likeness)**: *"Which speaker do you think sounds more like a real person?"*

Criteria	**w/o PCC**	**w/ PCC**
Appropriateness	49	51
Informativeness	45	55†
Engagingness	48	52
Human-likeness	49	51

Table 10: Human evaluation results for PCC in winning percentages. † indicates the results as passing a two-tailed binomial significance test with $p < 0.05$.

We follow Li et al. (2019) and Zou et al. (2021) to conduct a human evaluation of dialogue generation from the four aspects described above. We follow the settings used in Section 5.5 to invite three experienced annotators to mark 200 instances under A/B settings. The results in Table 10 indicate that PCC effectively improves the DIALOGPT baseline in all aspects, especially informativeness.

I Computing Infrastructure

We use an NVIDIA TITAN RTX with 24GB GPU memory for all of the experiments conducted in this paper. Training the text classification model consumes about 1 hour. Fine-tuning the dialogue generator consumes about 15 hours. Generating a single paraphrase to be used in PCC as a CDA method costs about 0.40 seconds on our machine.

A Two-Sided Discussion of Preregistration of NLP Research

Anders Søgaard **Daniel Hershcovich**
Department of Computer Science
University of Copenhagen
{soegaard,dh}@di.ku.dk

Miryam de Lhoneux
Department of Computer Science
KU Leuven
miryam.delhoneux@kuleuven.be

Abstract

Van Miltenburg et al. (2021) suggest NLP research should adopt *preregistration* to prevent fishing expeditions and to promote publication of negative results. At face value, this is a very reasonable suggestion, seemingly solving many methodological problems with NLP research. We discuss pros and cons—some old, some new: a) Preregistration is challenged by the practice of retrieving hypotheses after the results are known; b) preregistration may bias NLP toward confirmatory research; c) preregistration must allow for reclassification of research as exploratory; d) preregistration may *increase* publication bias; e) preregistration may *increase* flag-planting; f) preregistration may *increase* p-hacking; and finally, g) preregistration may make us less risk tolerant. We cast our discussion as a dialogue, presenting both sides of the debate.

1 Preregistration

Should NLP researchers be required to preregister their studies? Van Miltenburg et al. (2021) present arguments *for* preregistration, recently echoed by Ulmer et al. (2022). Preregistration has its origin in preregistration of clinical trials,[1] and amounts to the following: Before you initiate a set of experiments, you register your hypotheses, your experimental design and how you plan to analyze your results. Registration is time-stamped on an online platform with general public access. You then follow your plan as closely as possible and report any divergences in your final publication.

The discussion in van Miltenburg et al. (2021) is not unprecedented. Preregistration has been debated in epidemiology (Lash and Vandenbroucke, 2012), social psychology (Veer and Giner-Sorolla, 2016), experimental economics (Strømland, 2019) and information systems research (Bogert et al., 2021). Our discussion is inspired by the discussion in epidemiology, which is similar to NLP in focusing on data analysis rather than clinical trials.

There is an important ambiguity in how preregistration is discussed: Is the preregistration entry peer-reviewed or not? Chambers (2019) sees preregistration as a peer-reviewed process, and this is also what van Miltenburg et al. (2021) suggest for NLP. We therefore assume peer-reviewed preregistration below. The required format of the registered report is also important. In their Appendix, van Miltenburg et al. (2021) provide example questionnaires. We will assume registered reports will be lists of answers to such questionnaires, but in §9, we will suggest a few revisions to the questions formulated by van Miltenburg et al. (2021).

2 Why Preregister NLP Research?

Van Miltenburg et al. (2021) present four reasons for adopting preregistration in NLP: distinguishing between confirmatory and exploratory research, avoiding fishing expeditions and harking, mitigating publication bias and avoiding flag-planting:

Distinguishing Confirmatory from Exploratory The first apparent advantage to preregistration—often said to be the most important one (Nosek

---

[1] The first registries were established by medical researchers in the 1960s and were originally designed to help experimenters recruit participants for clinical trials, but as pointed out by Wiseman et al. (2019), preregistration, as we think of it today, started in parapsychology. In 1974, Martin Johnson, a professor of parapsychology and an editor of newly established European Journal of Parapsychology, introduced a preregistration practice for this journal (Johnson, 1975), in an effort to make parapsychology protocols more rigorous. In the editorial, Martin Johnson describes how according to the philosophy of the proposed preregistration model, experimenters should define their problems, formulate their hypotheses and outline their experiments, prior to commencing their studies. In Declaration of Helsinki §19, the World Medical Association (2013) demands: "Each clinical study must be registered in a publicly accessible database before the first test subject is recruited." While the European Commission refers to it, it has not been universally adopted (Rid and Schmidt, 2010).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 83–93
May 2-6, 2023 ©2023 Association for Computational Linguistics

et al., 2018)—is that it clarifies what counts as confirmatory research, which *has to* preregister, and what counts as exploratory research with no obligation to preregister. Confirmatory research is hypothesis testing, held to the highest standard and which aims to minimize false positives. Here, p-values are generally assumed to have diagnostic value and inferences can be drawn to wider populations. Exploratory research, in contrast, has a different status: It generates rather than tests hypotheses and results should be replicated and confirmed at a later stage. Typically, the focus is on minimizing false negatives, and p-values are not assumed to have diagnostic value (Schwab and Held, 2020). Moreover, findings are not assumed to be directly transferable to wider populations. Rubin (2020), however, points out how it is not always trivial to distinguish between confirmatory and exploratory research: if a researcher, for example, retries a hypothesis from previously published literature to explain an experiment they just ran, is this an *a priori* or a *post-hoc* hypothesis? See also §3.

Fishing Expeditions and Harking Preregistration is often said to prevent *fishing expeditions* and so-called *harking*[2] (Andrade, 2021), namely, post-hoc characterization of hypotheses based on experimental outcomes. Fishing expeditions is ambiguous in the literature (between fishing and harking), but we use the term to refer to cherry-picking dataset and protocols to validate a hypothesis. Harking, in turn, is what researchers do when they indiscriminately examine associations between different variables, not with the intention of testing *a priori* hypotheses but simply hoping to find something of significance. Rubin (2020) calls this 'undisclosed hypothesizing after the results are known.' Having authors preregister their hypotheses potentially improves the reliability of confirmatory research by controlling for cherry-picking and multiple hypothesis testing, implicit to exploratory research. See also §4.

Publication Bias Van Miltenburg et al. (2021) say that, to them, the main advantage of registered reports is that they provide a means to avoid publication bias. Because studies are evaluated *prior* to the results, negative results have the same chance to be published as positive ones. Rubin (2020) refer to this as avoiding the suppression of *a priori* hypotheses that yield null or disconfirming results. Publication bias is claimed to be a serious problem in NLP research by many (Plank et al., 2014; Card et al., 2020; Cohen et al., 2021). The argument was also used in epidemiology, but received some pushback (Loder et al., 2010). See also §5.

Flag-planting Van Miltenburg et al. (2021) also suggest preregistration can prevent so-called *flag-planting*. Flag-planting refers to rushing to be the first to publish results. Flag-planting potentially comes at the cost of scientific integrity and quality. Because of biases in peer-reviewing, it is harder to publish a corrected version of a study that is already out there, than to publish an error-prone study that is the first of its kind. See also §6.

Other Reasons to Preregister We have covered the main reasons van Miltenburg et al. (2021) had for adopting preregistration and will now move to our two-sided, dialogical discussion of its pros and cons. In our dialogue, we will let Zeny and Socart,[3] our house philosophers, debate preregistration. In §3–§8, we will let them discuss arguments *against* preregistration, including arguments that run counter to those presented by van Miltenburg et al. (2021), but we will first let Zeny provide us with a fifth argument *for* preregistration:

ZENY: Socart, there's an additional argument for preregistration, I believe. Early feedback on experimental methodology through a peer-reviewed registration process should improve the quality of the methodology, should it not? Such feedback also saves resources otherwise spent on failed or misleading experiments.

SOCART: Zeny, we both know turn-around is fast in NLP research. Experiments are easier to run and feedback is much faster than for clinical trials, where preregistration is common.

ZENY: NLP as a field has many virtues, but the reviewing cycle is slowing as the field grows larger. Moreover, experiments are becoming more expensive with larger models, creating barriers of entry (Bender et al., 2021) and experiments have substantial environmental impact.

SOCART: You make an important point, Zeny, but early feedback would require more time from reviewers. Since reviewers and researchers coincide,

---

[2] Short-hand for "hypothesizing after the results are known". Often conflated with fishing, but the two differ: Harking fixes the experiment, varies the hypothesis, so to speak, whereas fishing fixes the hypothesis, varies the experiment. The acronym was coined by social psychologist Norbert Kerr.

[3] Zeny is a mix of Kenny from South Park and Zeno. In Plato's *Parmenides*, Zeno argues for monism—the idea that reality is one stable thing. Socart is a mix of South Park's Eric Cartman and Socrates, who countered this idea by asserting a more nuanced ontology in which things stand in complex relations to each other. Socrates, in other words, took a more nuanced stance, arguing against the existence of a one-size-fits-all hypothesis. Zeny and Socart adopt similar positions in our dialogue about preregistration.

preregistration would potentially save compute resources, but not working hours.

ZENY: That is an oversimplification. Giving feedback on an early draft takes much less time than writing a full paper. If the reviewers are carried over, they will save time when reading the full submission, also. Preregistration would also prevent cherry-picking and invalid use of significance tests by excluding explorations from confirmatory research.

SOCART: But the explorations could be done prior to preregistration and researchers may then be more inclined not to report such explorations at all.[4]

ZENY: Any system can be tricked, but if researchers adopted the practice of preregistration, we would, all things being equal, increase transparency and decrease bias around research.

SOCART: Dear Zeny, you too have seen the evasiveness of bureaucracy, e.g., in NLP conference submission forms. While preregistration reports would initially be light-weight, transparency could easily be clouded by the complexity of assembling the information required for preregistration as new requirements are added over time.[5]

See Bracken (2011) and Rubin (2020) for a discussion of more advantages to preregistration. In addition to reducing fishing expeditions and harking, flag-planting and publication bias, these include: **a)** preventing p-hacking, **b)** prespecifying tolerated significance levels, **c)** identifying selective reporting,[6] and **d)** preventing *forking paths* practice.[7] None of these points are uncontroversial and Rubin (2020) also presents counter-arguments against **a)-d)**. For example, prespecified significance levels have been superseded by the practice of simply reporting *actual* α-levels. Surprisingly, there has been little work on whether preregistration increases trust in science, except for the study by Field et al. (2020), which was under-powered.

We focus on *preregistration for NLP research*. In general, there is no *a priori* reason to think that the pros and cons of preregistration transfer from clinical trials over epidemiology to NLP research. In clinical trials, for example, it is easy to decide when a protocol must be registered. This simply happens before the first subject is assigned to treatment. In epidemiology, there is no such bright line (Lash and Vandenbroucke, 2012) and it is equally hard to see one in NLP. While general machine learning has seen many related methodological discussions (Gencoglu et al., 2019; Lipton and Steinhardt, 2018; Gundersen et al., 2022), there has, to the best of our knowledge, been no published discussions of preregistration practice in this field, with the exception of Gundersen (2021).[8][9]

In our discussion below, we will ignore the most trivial challenges to preregistration, such as deviations from data collection plans for practical reasons, discovery of assumption violations, etc. Such challenges have already been discussed in the clinical literature, e.g., by Nosek et al. (2018).

3 Encouraging Confirmatory Research

We let SOCART and ZENY discuss whether preregistration will succeed in distinguishing between confirmatory and exploratory research. A decade ago, when preregistration was being implemented and discussed in epidemiology, the worry that preregistration would introduce a bias against "the end of the research spectrum that constitutes the quirky, brilliant work that is not enterprise-driven" (Sørensen and Rothman, 2010), was the main concern among its opponents. SOCART and ZENY discuss the consequences of insisting on a distinction that is not trivial to uphold in practice.

SOCART: It seems to me, dear Zeny, that many NLP projects are driven *not* by an explicit hypothesis, but by a desire to understand the behavior of a model, to be able to characterize its strengths and weaknesses, or by a simple gut feeling that at the locus of interacting variables, interesting dynamics can be observed.

ZENY: Can you provide me with an example?

SOCART: Certainly. Pires et al. (2019), e.g., showed that knowledge encoded in multilingual BERT (Devlin et al., 2019), could be transferred across languages—even across scripts, that such transfer worked best between typologically similar languages, that it could process code-switching and find translation pairs. They also showed systematic deficiencies affecting some language pairs. How would they have foreseen these findings? Or even the dimensions that turned out to be of interest? Even if they had foreseen how they wanted to explore transfer across scripts and typological classes, what if genealogy or demography turned out to be more interesting than typology?[10]

---

[4] This point was also made for preregistration in epidemiology by Sørensen and Rothman (2010).

[5] See Loder et al. (2010) for arguments from epidemiology.

[6] Selective reporting is regarded the most important contributor to irreproducibility by Baker (2016). Nosek et al. (2018), advocating for preregistration, presents similar arguments.

[7] This practice refers to when researchers make decisions about which correlation tests to conduct based on properties of their data. The practice is named after *The Garden of Forking Paths*, a 1941 short story by Jorge Luis Borges.

[8] Workshops on preregistration at ICCV 2019 (`https://preregister.vision/`) and at NeurIPS 2021 (`https://preregister.science/`) seemingly did not lead to publications or a change in practice yet, but the website for the 2021 workshop says papers are forthcoming.

[9] Gundersen (2021) complains no AI venues support preregistration, but provide no arguments for or against it.

[10] Independent language families may share features, i.e., be typologically close, but genealogically apart. See Rama and Kolachina (2012) for discussion.

ZENY: I am unconvinced that preregistration would be a serious obstacle to such work. Pires et al. (2019) could have defined the search space in advance – or maybe this is exploratory work that would not have to register in the first place? Remember also, Socart, that the preregistered plan can be updated and refined in the course of a research project. Plans can be revised, but this does not cancel out the benefits of planning.

SOCART: Preregistration may accommodate deviation from the plan, but would risk losing its benefit if researchers were allowed to preregister too many hypotheses or update their plans too frequently. Let us illustrate this with another example. Zhao and Bethard (2020) study how BERT models' learned self-attention functions change during fine-tuning to reflect the target task. They find this to be the case only in smaller models; with more parameters, the change disappears. Imagine now that their hypothesis was confirmed only for select combinations of positional encodings, regularizers and optimizers.

ZENY: This sounds suspiciously like a case of forking paths. Dror et al. (2017) warned us about this risk, encouraging us to at least validate our hypotheses on multiple datasets to reduce the chance of p-hacking.[11] Again, preregistration would not be required for all research.

SOCART: So if authors submitted exploratory work for peer review, would reviewers then decide if bypassing preregistration was appropriate?

ZENY: Yes. Preregistration clarifies the distinction between exploratory and confirmatory research.

SOCART: But what if Pires et al. (2019) had pointed to earlier work already hypothesizing that transfer works best between typologically similar languages? Would this not have made their research confirmatory in the eyes of their readers and therefore in need of preregistration?[12]

ZENY: It very well might have. If they consider it exploratory, they should also point to alternative hypotheses that would explain different results.

SOCART: Moreover, if preregistration becomes a badge of honor or increases your chances of getting your work accepted, because the findings have a different air of trustworthiness,[13] would this not be a reason to encourage your students to perform confirmatory rather than exploratory research? Preregistration would, in other words, inject a bias toward confirmatory research into NLP.

ZENY: I, for one, would welcome this kind of bias.

✓ *Preregistration is challenged by r-harking and may bias NLP toward confirmatory research.*

4 Some Expeditions May Prevent Others

If you ask an NLP researcher if they are "on a fishing expedition" or if they are hypothesizing after the fact, you will instantly make them feel very uncomfortable. It is widely accepted that fishing and harking are bad practices.[14] Socart, however, has an argument *for* (occasional) harking:

SOCART: Two researchers, Ann and Bob, have the same hunch, that the regularization technique R_2 is better than its competitors, R_0, R_1, R_3. Ann realizes after a set of experiments of datasets D_0, D_1, D_2 that, in fact, R_1 is better than R_2. Since this was previously unknown to the community, she publishes it, presenting it (somewhat vaguely) as a confirmation of an *a priori* hypothesis. Bob tests the same hypothesis, i.e., that R_2 is superior to R_0, R_1, R_3. Seeing R_1 is better on D_0, D_1, D_2, he looks for more datasets, until he has a suite of datasets D_4, D_5, D_6 on which R_2 is better than R_1.

ZENY: Bob's cherry-picking is extremely problematic, but so is Ann's harking.

SOCART: But would you agree that granting her the freedom to hark most likely reduces her temptation to cherry-pick?

ZENY: This would simply reclassify Ann's work as exploratory rather than confirmatory. I see no reason why preregistration should not allow this.

SOCART: The two researchers both departed from their original plans, but Ann's willingness to depart from her original hypothesis serves us better than Bob's cherry-picking. In this way, harking can prevent a researcher from taking on a fishing expedition.

✓ *Preregistration should allow for re-classification of confirmatory research as exploratory research.*[15]

5 Solving Publication Bias?

Sørensen and Rothman (2010) argue against preregistration solving publication bias, because researchers can still selectively register studies after preliminary data explorations. Imagine Hippocrates, the Greek physician, was asked to preregister his vivisection experiments. If Hippocrates was studying 10 soldiers with brain lesions, what would prevent him from using one soldier to generate hypotheses, preregister those and conduct the final experiments on the remaining nine? Or worse, peek at all, preregister and go back to the data?

SOCART: Say Hippocrates has two hypotheses about the soldiers, such as that the heart is the seat of

---

[11] See Belz et al. (2021) for a similar discussion.

[12] This practice is known as "retrieving hypotheses after the results are known" (r-harking) (Rubin, 2017).

[13] Greater reliance on preregistration improves estimation of effect sizes, as shown by Strømland (2019).

[14] Andrade (2021) notes that fishing expeditions can be "ethical" if acknowledged as such, and if appropriate corrections are performed when computing significance results.

[15] Reclassification flags work as exploratory, thereby increasing transparency, but would not impact acceptance decisions. An alternative, suggested by one of our reviewers, would be to introduce intermediate reports as a required step to share the results of the preregistered study before continuing to preregister and test alternative hypotheses a part of the same study. This further increases transparency, and prevents having 'unwanted' results 'swept under the rug' in the final publication. Researchers working on a similar topic would already benefit from the results in an intermediate report.

compassion and that the brain is the seat of rational thought. Upon preliminary exploration, he sees many soldiers have turned cold-hearted by the atrocities of war, but few complain of heartaches. Nearly all soldiers who are delusional or suffer from memory loss, also suffered blows to their heads. Hippocrates pursues and preregisters only the hypothesis that the brain is the seat of rational thought. He has now preregistered, not a prediction, but a post-diction, ignoring the negative result.

ZENY: But Socart, did you not, a moment ago, argue that preregistration would dampen the creativity of research by preventing fishing expeditions and harking?

SOCART: In theory, yes. Preregistration will dampen creativity if properly sanctioned, but I am skeptical that this would be practically possible, rendering preregistration ineffective, an unnecessary administrative burden for all—and a bottleneck for the honest few.

ZENY: If preregistration *prior* to data collection is encouraged, this would solve the problem, no?[16]

SOCART: Surely, but this would mean only one preregistered study per dataset. Since few NLP papers introduce new datasets, this would render preregistration ineffective for the vast majority of NLP research.

ZENY: This is a good point, Socart, but community-wide overfitting to benchmarks is a vice, not a virtue. If preregistration encourages the introduction of new test datasets, that's a good thing, no? Some even argue that all papers should ideally introduce new test data.

To the contrary? ...in which SOCART and ZENY continue to discuss whether preregistration could actually make publication bias worse. SOCART suggests that preregistration could amplify publication bias, if positive results are still preferred over negative ones and preregistration forces researchers to focus on predictably positive results, arguably a small subset of the positive results. If papers are accepted on the basis of preregistration, this could increase an arguably already existing bias toward incremental improvements.[17]

SOCART: You say preregistration will make it easier to publish negative results, because studies are evaluated prior to obtaining results?

ZENY: That is correct, Socart.

SOCART: ...but do we really know why there are so few NLP papers about negative results? See, in NLP, negative results are much harder to establish than positives. If I want to show that self-attention or weight averaging does *not* lead to improvements for some problem, I need to show that holds across all implementations, all architectures and all available datasets. The value of a report stating that for one such combination, self-attention didn't do much, would be next to nothing. Isn't that the real explanation for the skew in the NLP literature?

ZENY: Negative results are key to scientific progress (Barwich, 2019), but are hard to establish if they are very general. Published positive results often overclaim their generality. Both should be confirmed only by accumulated evidence in diverse settings.[18]

SOCART: No, no, you fail to see there's a qualitative difference, Zeny! Imagine if Ann was evaluating self-attention for sentiment analysis. To answer the hypothesis that self-attention works in the positive, she just needs a significant result in a single setting. In contrast, in order to establish a negative result, she has to explore *all* possible settings.[19] How would preregistration make establishing a negative result less formidable a challenge?

ZENY: I agree that it would not. My only claim is that evaluating studies prior to obtaining results would prevent any bias on behalf of peer reviewers to evaluate negative results more harshly.

SOCART: ...but in reality, we do not know if such a bias exists, or whether it is only fair that such a bias exists, because the bar by definition should be higher for negative results?

✓ *Preregistration may increase publication bias.*

6 Solving Flag-Planting?

Flag-planting is one of the motivations for preregistration for van Miltenburg et al. (2021), but *exclusive* preregistration may also, conceivably, have the opposite effect.

SOCART: Say Ann and Bob get the same great idea—e.g., to evaluate the sensitivity of textual entailment models to presupposition projection—and worry that they will be scooped before getting around to publishing it. Ann and Bob now follow two different strategies: Bob rushes to preregister a study hypothesizing that state-of-the-art models are sensitive to such phenomena, while Ann rushes to run the experiments and publish the paper. Zeny, which strategy is better for science?

ZENY: I would say it's Bob's, since rushed experiments are more likely to be flawed.

SOCART: But Bob plants his flag faster than Ann, essentially scooping her. By doing so, Bob discourages Ann from pursuing this idea by planting his flag first. What if Bob fails to conduct proper

---

[16] This is implicit in preregistration in scientific fields where data is not re-used, e.g., in psychology (Wiseman et al., 2019).

[17] Lash and Vandenbroucke (2012), for example, argue that: "prespecified hypotheses often take little risk, invoke little imagination and stray only a short distance from what is already well understood."

[18] NLP has seen relatively few meta-studies (Cramer, 2008; Søgaard, 2013; Hoyle et al., 2021; Bugliarello et al., 2021), but hopefully, we will see more in the future.

[19] We flesh this out a bit. The research hypothesis in Ann's case is that self-attention helps. What this means is that in *some* implementation, it leads to robust improvements. The vast majority of NLP hypotheses take this form: X can, in some implementation, lead to general improvements on one or more tasks. If the baseline is fixed, this amounts to existential quantification ("some"). Conversely, its negation ("self-attention does not help for sentiment analysis") amounts to universal quantification, i.e., there is *no* implementation in which this is the case. This is obviously much harder to prove than the corresponding positive result.

experiments altogether? Had it not been for preregistration, both researchers would have pursued their idea, providing mutual replication and increasing the likelihood of the idea materializing into an actual result.

ZENY: Some would say this is one of the advantages of preregistration: Ann pursuing the same idea would have been a waste of time.

SOCART: This assumes Ann and Bob would have conducted their research in exactly the same way and that none of them were prone to error. In other words, that researchers are machines that simply execute their unambiguous experimental protocols. I think preregistration just moves flag-planting to earlier in the research process, lowering the bar for researchers to plant their flags, since less work is required to plant a flag. And when a bar is lowered, more researchers are likely to plant more flags.

ZENY: Van Miltenburg et al. (2021) explicitly encourage concurrent work.

SOCART: Yes, I did read that passage, but they do not discuss *how* preregistration would impact concurrent work. Do they envisage a review system in which Ann is allowed to follow up on the idea that Bob preregistered?

ZENY: I'd do that, in the spirit of open science.

SOCART: Such *inclusive* preregistration would clearly discourage protectionist researchers from preregistering their studies. If a preregistered study is up for grabs for other research labs, labs with more resources could likely wrap up the experiments faster than the researchers who registered it.

ZENY: . . . unless we envisage a review system allowing Ann to preregister the same study, giving Ann and Bob equal chances to pursue the study.

SOCART: This would be equivalent to telling reviewers of a paper to consider as "concurrent" any other work published within the last 1–2 years (assuming this is the approximate life span of a research project), including preregistered studies. Today, reviewers are told to disregard work published within the last three months, but already, reviewers seem to ignore this guideline in practice, presumably because they do not want to compromise the fast turn-around in NLP research.

ZENY: But shouldn't we incentivize slow science? Many NLP papers neglect related work and keep reinventing the wheel. We need deeper analysis to enable disruptive scholarship and novel ideas.[20]

SOCART: Slow science also has disadvantages. Fast turn-around has had many positive effects on NLP, including rapid replication. Projects can become "too big to fail," causing confirmation bias.[21] Lowering false positive rates is important, but so is healthy distrust in published results.

✓ *Preregistration may increase flag-planting.*[22]

7 Solving p-Hacking?

Inflation bias, also known as p-hacking, refers to selective reporting to produce statistically significant results. Søgaard et al. (2014) lists several p-hacking techniques used, perhaps inadvertently, in NLP papers. If a statistically significant result is seen as the key to getting your paper accepted, researchers are presumably willing to go far to squeeze out a small p-value. But if preregistration facilitates the publication of negative results, it seems it would also reduce the incentive to engage in so-called p-hacking, e.g., obsessive fiddling with data and models until reaching the magical $p < 0.01$. It has been noted, however, that preregistration leaves plenty of room for p-hacking (Bakker et al., 2020). Generally, eliminating p-hacking entirely is unlikely when career advancement is assessed by publication output, and positive results are favored by scientific peers (Head et al., 2015).

Socart and Zeny discuss whether preregistration will reduce or amplify the incentive to engage in p-hacking:

SOCART: Imagine Ann again, who is now evaluating if self-attention is helpful for sentiment analysis. Say she preregisters the hypothesis that self-attention *is* helpful, only to find that her first results are negative. We would now like Ann to go ahead and acknowledge the negative results on print, right? However, as we just saw, when your first results are negative, more results are typically needed to draw a firm, negative conclusion that self-attention does *not* help. Sometimes more data collection is needed and more human evaluations may be needed. Pursuing the negative result will, in other words, be a lot of work.

ZENY: But very important!

SOCART: Preregistration increases the amount of work that goes into moving your focus to establishing a negative result: You will need to augment your preregistration with information about the experiment, your new hypothesis and the new experiment you plan to perform.

ZENY: Documentation has to be light-weight.

SOCART: . . . but preregistration would get people more invested in their ideas and bias them in how results are interpreted. When people go on record with a study description, they will defend why it's reasonable and likely leading to a positive result. Researchers are always prone to confirmation biases, but now social expectations and reputation will amplify their existing biases. This would lead to the opposite of the effect intended.

✓ *Preregistration may increase p-hacking.*

8 Risk Tolerance

Attempts to reduce false positives tend to also lead to reductions in true positives. Many applications

---

[20] Chu and Evans (2021) showed that fast turn-around results in stymied fundamental progress in large scientific fields.

[21] This can result from financial interests (Ioannidis, 2005), e.g., due to "sunk cost" (Perignat and Fleming, 2022).

[22] One obvious solution is to make preregistration non-public, but then preregistration would not prevent two groups doing the same study.

require near-zero false positive rates, but most NLP experiments show low risk of direct negative impact on society or individuals therein,[23] as indicated by the relatively few papers receiving ethical reviews. Hence, we can afford to take risks and explore hypotheses that end up wrong. Parascandola (2010) reminds us how this is a key ingredient in increasing knowledge and reducing uncertainty, getting us off the beaten track. NLP benefits from being frequently wrong and implementing preregistration to prevent false positives has a drawback. In §9, we will argue that what is important is to balance preregistration with our risk tolerance.

SOCART: Imagine Ann works on hate speech detection for a social media company. Bob works on topic classification of social media posts at the same company. They both validate and evaluate the models in the wild on beta users. They both can use logistic regression and SVMs. SVM is sometimes superior, but exhibits more variance across hyper-parameters. If I were to advise Ann or Bob to use logistic regression, who then?

ZENY: Probably Ann, since we can tolerate less risk in her situation. But how would preregistration affect the risk tolerance of researchers?

SOCART: Imagine if you were asked today to carry a solid bottle of olive oil over to Plato's house and tomorrow to bring him a fragile, beautiful vase decorated with gold. On which of the two days would you be more inclined to *run* there?

ZENY: Today, but how is this relevant? What are the tasks where we can afford to 'run'?

SOCART: For tasks in which false positives are associated with high risk, we should hedge our bets by preregistering conservative hypotheses; for other tasks, this sort of inhibition of is unfortunate.

ZENY: This is exactly why exploratory research still has a place in a world with preregistration—namely, for tasks where we can tolerate risk.

One may argue that risk mitigation is not what preregistration is for. The purpose of institutional review boards (IRBs) and ethics reviewing is to flag and prevent too risky studies (IRBs focus on risk to human participants, while ethics reviewing also addresses potential applications). We have three reasons to think preregistration requirements should depend on expected risk: (a) It is impossible to review the implications of a study before you have a solid study plan. If preregistration includes risk assessment, this could provide input for IRBs and ethics reviewing (or, in a more distant future, be part of the same process). (b) A partial roll-out of preregistration may help us balance Type 1 and Type 2 errors. Expected risk affects the cost of false positives and hence the optimal balance between Type 1 and Type 2 errors. Since bureaucracy, by the end of the day, also incurs a cost on society, this reinforces our belief that mechanisms should be implemented only for where there is direct impact on society at large. (c) Finally, ethics reviewing is typically part of the standard review process, i.e. *after the fact* and can therefore not respond to malpractice in the experimental design or prevent publication of preprints.

Overly cautious preregistration practice may, in sum, decrease our true positive rate and add bureaucratic overhead to research practices without proper motivation. An all-over-the-map roll-out of preregistration would change the risk tolerance in research and society, just like registration and documentation has increased risk sensitivity in the past. Simultaneously, evaluating risk early on has clear advantages over the current review process.

✓ *Preregistration may lower our risk tolerance.*

9 A Proposal

We have tried to present pros and cons of preregistration. If we have focused a bit more on the cons, this is only because van Miltenburg et al. (2021) did a great job highlighting its advantages. We will, if anything, argue for only a partial roll-out of preregistration of NLP research. Preregistration is a way to minimize harms of NLP research, but only when risk is high. To motivate this, consider, as first noted by Lash and Vandenbroucke (2012), two seemingly opposed arguments *for* preregistration: a) Preregistration counters the suppression of (negative) results. b) Preregistration identifies false positives. Lash and Vandenbroucke (2012) argue that while (b) is a valid argument for preregistration of *clinical trials*, it is not a valid argument for preregistration in the context of mere "accumulation of evidence" (Lash and Vandenbroucke, 2012). Here, concerns about balancing Type 1 and Type 2 errors disappear. Preregistration mitigates risks associated with research, reducing potential harms, but at the cost of scientific progress. This calls for a cost-benefit analysis: How much risk can be tolerated for what potential gains?

One way to frame this discussion in NLP is to

---

[23] This does not refer to the downstream risks after deployment, just the risks associated with the research experiments. Two reasons for NLP experiments being relatively low risk are the rare involvement of human participants in NLP experiments and the historical focus on professionally generated text (Hovy and Spruit, 2016). We are seeing a shift toward human-in-the-loop evaluations and user-generated content, but this still makes for a small fraction of NLP research.

ask how afraid we should be of being wrong. In clinical trials, there is a significant cost to being wrong. In biomedical studies, false positive rates have been found to be around 14% (Jager and Leek, 2014). Whatever the number is for NLP, lowering it by adding more checks, will lead to a drop in the *true* positive rate. If a false positive could result in human tragedy, the price of a lower true positive rate is worth paying, but in NLP, the cost of a false positive is often paid for in compute and human hours. While both can be scarce resources, the open access nature of NLP makes being wrong less dangerous, since mistakes are quickly corrected.[24] Zeny would object that pretraining of language models is not easily reproducible (Bender et al., 2021). Pretraining very large language models should maybe be required to preregister and this would possibly require revising the questionnaires provided by van Miltenburg et al. (2021). Another concern is how NLP contributes to social and cultural inequality (Hershcovich et al., 2022). If NLP research is likely to help some more than others, this may be reason to require preregistration. Here, the questionnaires provided by van Miltenburg et al. (2021) would also be insufficient.[25]

So what we propose here reflects a middle-of-the-road position on preregistration. The idea is to **limit preregistration to research for which our risk tolerance is low**. This prevents most of the adverse effects of preregistration, e.g., publication bias, flag-planting and p-hacking. NLP research is subject to IRB and ethics reviewing, but we believe this should be merged with preregistration (§8). Reports should be reviewed and reviewers follow the submission (§2).[26] Currently, we accept and reject papers through blind peer-reviewing, but some papers are accepted *conditional* on a positive ethics review. We propose a reviewing procedure in which some work is only accepted conditional on the work having already been registered with positive reviews. For researchers, this would mean you need to get your preregistered reports accepted, before you initiate the research project. Once completed you will send the final submission in for a new set of reviews, hopefully by the same reviewers. This procedure is somewhat cumbersome and has all the disadvantages we discussed above. Therefore, it should only be used when it is deemed necessary, i.e., when the expected risk of the NLP research is sufficiently high.

A paper which a) is confirmatory and b) concerns an application for which risk tolerance is low, can be rejected for not having preregistered. We noted the necessity of allowing preregistered research to re-classify as exploratory, i.e., conditional acceptance for non-preregistered, flagged research, if it explicitly labels itself as exploratory. This would lead to three different categories of accepted papers: (i) non-preregistered (confirmatory or exploratory) research, (ii) preregistered, confirmatory research and (iii) non-preregistered research *marked explicitly as exploratory*. This would, we argue, give us the advantages of preregistration where they are most needed, e.g., where false positives are associated with very high risk.

We left one important thing in the open for now: How do we fairly decide if a research subject and protocol warrants low risk-tolerance? Ethics review board members are already asked to flag work that 'exhibits an increased risk of harm outside the current norms of NLP or CL research'.[27] This can be hard to determine, but board members *already* have to make this difficult decision. Ethics reviews could learn from established risk assessment frameworks (Schwerdtner et al., 2020).

10 Conclusion

Our two-sided dialogue has discussed pros and cons of preregistration in NLP, building on similar discussions in epidemiology. What opponents elsewhere have proposed as alternatives to preregistration is already found in NLP research: open access, common repositories and data sheets (Lash and Vandenbroucke, 2012). Preregistration, we argue, is less urgently needed in fields that already facilitate **replication** and where **risk of false positives** is

---

[24] Of course researchers are sometimes blind-sighted by scientific paradigms and hype, biasing their interpretation of results. Such dynamics is central to, e.g., the Popper-Kuhn debate (Rowbottom, 2011), but beyond the scope of this paper.

[25] Specifically, we think the questionnaire for 'NLP Engineering experiment paper' (§A.3) should include questions about computational resources needed for pretraining. Since the risk of wasting resources is high for language model retraining, preregistration and early feedback may be particularly useful for such research, but the review of the registered point would have to take this information into account. To mitigate social and cultural inequality, we propose to revise the questionnaire for 'Resource paper' (§A.6), adding questions about the demographics of data sources and annotators, as well as making a corresponding explication of social and cultural concerns in Question 11 of §A.3.

[26] This would be hard to coordinate for most fields, but in NLP, the ACL Rolling Review platform could make it easier.

[27] `https://aclrollingreview.org/ethicsreviewertutorial`

low. The **fast turn-around** of NLP research means the advantages of transparency and early feedback are smaller. Nevertheless, society's risk-tolerance varies across NLP applications. Legal or medical decision support systems are high-risk application areas. Here, we need to consider all safety measures on the table, including preregistration.

Impact Statement

Preregistration is one of several practices that promote responsible, high-quality research. Others include replication, transparency and open access, as well as impact statements and explicit discussion of study limitations. All such practices come with pros and cons and it is key to scientific progress and positive impact that scientific communities evaluate which practices are adequate in their domain. The increasing real-world impact that NLP research has exhibited recently and will likely continue to exhibit warrants a careful reconsideration of which practices are called for. Since a major driver of the same impact is the fast-paced exploratory research that characterizes the field, limiting such research may have negative effects as well (see §6). We therefore believe our two-sided debate will enable an overall better outcome in terms of impact.

Limitations

Our discussion of preregistration is inspired by discussions in epidemiology. Many of the concerns epidemiologists had with preregistration seem more relevant to NLP research than the considerations that, by and large, led clinical research to adopt preregistration as a mandatory practice. While we present a proposal for how to implement preregistration in NLP in §9–a proposal that differs from the one presented by van Miltenburg et al. (2021)–our main contribution is a two-sided discussion of its pros and cons, leaving many questions in the air. Our paper is intended to get the preregistration debate off ground, not to nail it to the floor.

Acknowledgements

Thanks to Asbjørn Hróbjartsson and Klemens Kappel for providing us with a helpful overview of the literature on preregistration in epidemiology. Thanks to our anonymous reviewers, as well as all of CoAStaL, for useful feedback on the above discussion. Anders Søgaard received financial support from Innovation Fund Denmark, Google Focused Research Awards, and the Novo Nordisk Foundation.

References

Chittaranjan Andrade. 2021. Harking, cherry-picking, p-hacking, fishing expeditions, and data dredging and mining as questionable research practices. *The Journal of clinical psychiatry*, 82.

Monya Baker. 2016. Is there a reproducibility crisis? *Nature*, 533:452–454.

Marjan Bakker, Coosje Veldkamp, Marcel Assen, Elise Crompvoets, How Hwee Ong, Brian Nosek, Courtney Soderberg, David Mellor, and Jelte Wicherts. 2020. Ensuring the quality and specificity of preregistrations. *PLOS Biology*, 18:e3000937.

Ann-Sophie Barwich. 2019. The value of failure in science: The story of grandmother cells in neuroscience. *Frontiers in Neuroscience*, 13.

Anya Belz, Shubham Agarwal, Anastasia Shimorina, and Ehud Reiter. 2021. A systematic review of reproducibility research in natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 381–393, Online. Association for Computational Linguistics.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Eric Bogert, Aaron Schecter, and Rick Watson. 2021. Preregistration of information systems research. *Communications of the Association for Information Systems*, 49.

Michael Bracken. 2011. Preregistration of epidemiology protocols: a commentary in support. *Epidemiology*, 22:447.

Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. 2021. Multimodal pretraining unmasked: A meta-analysis and a unified framework of vision-and-language BERTs. *Transactions of the Association for Computational Linguistics*, 9:978–994.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With little power comes great responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

Chris Chambers. 2019. What's next for registered reports? *Nature*, 573:187–189.

Johan S. G. Chu and James A. Evans. 2021. Slowed canonical progress in large fields of science. *Proceedings of the National Academy of Sciences*, 118(41):e2021636118.

Kevin Cohen, Karën Fort, Margot Mieskes, Aurélie Névéol, and Anna Rogers. 2021. Reviewing natural language processing research. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, pages 14–16, online. Association for Computational Linguistics.

Irene Cramer. 2008. How well do semantic relatedness measures perform? a meta-study. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, pages 59–70. College Publications.

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

Rotem Dror, Gili Baumer, Marina Bogomolov, and Roi Reichart. 2017. Replicability analysis for natural language processing: Testing significance with multiple datasets. *Transactions of the Association for Computational Linguistics*, 5:471–486.

Sarahanne M. Field, E.-J. Wagenmakers, Henk A. L. Kiers, Rink Hoekstra, Anja F. Ernst, and Don van Ravenzwaaij. 2020. The effect of preregistration on trust in empirical research findings: Results of a registered report. *Royal Society Open Science*, 7(4).

Oguzhan Gencoglu, Mark van Gils, Esin Guldogan, Chamin Morikawa, Mehmet Süzen, Mathias Gruber, Jussi Leinonen, and Heikki Huttunen. 2019. Hark side of deep learning – from grad student descent to automated machine learning.

Odd Erik Gundersen. 2021. The case against registered reports. *AI Magazine*, 42(1):88–92.

Odd Erik Gundersen, Kevin Coakley, and Christine Kirkpatrick. 2022. Sources of irreproducibility in machine learning: A review.

Megan L. Head, Luke Holman, Rob Lanfear, Andrew T. Kahn, and Michael D. Jennions. 2015. The extent and consequences of p-hacking in science. *PLOS Biology*, 13(3):1–15.

Daniel Hershcovich, Stella Frank, Heather Lent, Miryam de Lhoneux, Mostafa Abdou, Stephanie Brandl, Emanuele Bugliarello, Laura Cabello Piqueras, Ilias Chalkidis, Ruixiang Cui, Constanza Fierro, Katerina Margatina, Phillip Rust, and Anders Søgaard. 2022. Challenges and strategies in cross-cultural NLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6997–7013, Dublin, Ireland. Association for Computational Linguistics.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

Alexander Miserlis Hoyle, Pranav Goel, Andrew Hian-Cheong, Denis Peskov, Jordan L. Boyd-Graber, and Philip Resnik. 2021. Is automated topic model evaluation broken? the incoherence of coherence. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 2018–2033.

John P. A. Ioannidis. 2005. Why most published research findings are false. *PLoS Med*, 2(8):e124.

Leah Ruth Jager and Jeffrey T. Leek. 2014. An estimate of the science-wise false discovery rate and application to the top medical literature. *Biostatistics*, 15 1:1–12.

Martin Johnson. 1975. Models of control and control of bias. *European Journal of Parapsychology*, 1:36–44.

Timothy L Lash and Jan P Vandenbroucke. 2012. Should preregistration of epidemiologic study protocols become compulsory?: Reflections and a counterproposal. *Epidemiology*, 23(2):184–8.

Zachary C. Lipton and Jacob Steinhardt. 2018. Troubling trends in machine learning scholarship.

Elizabeth Loder, Trish Groves, and Domhnall Macauley. 2010. Registration of observational studies. *BMJ (Clinical research ed.)*, 340:c950.

Emiel van Miltenburg, Chris van der Lee, and Emiel Krahmer. 2021. Preregistering NLP research. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 613–623, Online. Association for Computational Linguistics.

Brian A. Nosek, Charles R. Ebersole, Alexander C. DeHaven, and David T. Mellor. 2018. The preregistration revolution. *Proceedings of the National Academy of Sciences*, 115(11):2600–2606.

Mark Parascandola. 2010. Epistemic risk: empirical science and the fear of being wrong. *Law, Probability and Risk*, 9(3-4):201–214.

Elaine Perignat and Fraser F. Fleming. 2022. Sunk-cost bias and knowing when to terminate a research project. *Angewandte Chemie International Edition*, 61(36):e202208429.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Barbara Plank, Anders Johannsen, and Anders Søgaard. 2014. Importance weighting and unsupervised domain adaptation of POS taggers: a negative result. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 968–973, Doha, Qatar. Association for Computational Linguistics.

Taraka Rama and Prasanth Kolachina. 2012. How good are typological distances for determining genealogical relationships among languages? In *Proceedings of COLING 2012: Posters*, pages 975–984, Mumbai, India. The COLING 2012 Organizing Committee.

Annette Rid and Harald Schmidt. 2010. The 2008 declaration of helsinki — first among equals in research ethics? *Journal of Law, Medicine & Ethics*, 38(1):143–148.

Darrell Rowbottom. 2011. Kuhn vs. popper on criticism and dogmatism in science: A resolution at the group level. *Studies In History and Philosophy of Science Part A*, 42:117–124.

Mark Rubin. 2017. When does harking hurt? identifying when different types of undisclosed post hoc hypothesizing harm scientific progress. *Review of General Psychology*, 21:308–320.

Mark Rubin. 2020. Does preregistration improve the credibility of research findings? *The Quantitative Methods for Psychology*, 16(4):376–390.

Simon Schwab and Leonhard Held. 2020. Different worlds confirmatory versus exploratory research. *Significance*, 17(2):8–9.

Paul Schwerdtner, Florens Greßner, Nikhil Kapoor, Felix Assion, René Sass, Wiebke Günther, Fabian Hüger, and Peter Schlicht. 2020. Risk assessment for machine learning models.

Anders Søgaard. 2013. Estimating effect size across datasets. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 607–611, Atlanta, Georgia. Association for Computational Linguistics.

Anders Søgaard, Anders Johannsen, Barbara Plank, Dirk Hovy, and Hector Martínez Alonso. 2014. What's in a p-value in NLP? In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 1–10, Ann Arbor, Michigan. Association for Computational Linguistics.

Henrik Toft Sørensen and Kenneth J Rothman. 2010. The prognosis for research. *B M J*, 340:c703.

Eirik Strømland. 2019. Preregistration and reproducibility. *Journal of Economic Psychology*, 75:102143. Replications in Economic Psychology and Behavioral Economics.

Dennis Ulmer, Elisa Bassignana, Max Müller-Eberstein, Daniel Varab, Mike Zhang, Christian Hardmeier, and Barbara Plank. 2022. Experimental standards for deep learning research: A natural language processing perspective. In *ML Evaluation Standards Workshop at ICLR 2022*.

Anna Veer and Roger Giner-Sorolla. 2016. Preregistration in social psychology—a discussion and suggested template. *Journal of Experimental Social Psychology*, 67.

Richard Wiseman, Caroline Watt, and Diana Kornbrot. 2019. Registered reports: An early example and analysis. *PeerJ*, 7:e6232.

World Medical Association. 2013. World medical association declaration of helsinki ethical principles for medical research involving human subjects.

Yiyun Zhao and Steven Bethard. 2020. How does BERT's attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4729–4747, Online. Association for Computational Linguistics.

WinoDict: Probing language models for in-context word acquisition

Julian Martin Eisenschlos[1], **Jeremy R. Cole**[1], **Fangyu Liu**[2], **William W. Cohen**[1]
[1] Google Research [2] University of Cambridge
`{eisenjulian,jrcole,wcohen}@google.com` `fl399@cam.ac.uk`

Abstract

We introduce a new in-context learning paradigm to measure Large Language Models' (LLMs) ability to learn novel words during inference. In particular, we rewrite Winograd-style co-reference resolution problems by replacing the key concept word with a synthetic but plausible word that the model must understand to complete the task. Solving this task requires the model to make use of the dictionary definition of the new word given in the prompt. This benchmark addresses word acquisition, one important aspect of the diachronic degradation known to afflict LLMs. As LLMs are frozen in time at the moment they are trained, they are normally unable to reflect the way language changes over time. We show that the accuracy of LLMs compared to the original Winograd tasks decreases radically in our benchmark, thus identifying a limitation of current models and providing a benchmark to measure future improvements in LLMs ability to do in-context learning.

1 Introduction

Large Language Models (LLMs) such as GPT-3 (Brown et al., 2020) and PaLM (Chowdhery et al., 2022) can only learn from information that is in their training corpus. However, this is naturally limiting because the training corpus itself is bounded in time to the point of its collection. As a result, recent work has studied how to adapt such models to new data without an expensive retraining phase. Methods range from using semi-parametric methods with access to external memory (e.g., Guu et al. 2020; Lewis et al. 2020), to continual learning (e.g., Dhingra et al. 2022; Lazaridou et al. 2021), to parameter efficient fine-tuning (e.g., Ben Zaken et al. 2022; Pfeiffer et al. 2021).

Much of this work concerns factual knowledge or task distribution shifts. However, language also changes subtly: for instance, the popularity or meaning of individual words can change over time. In fact, such shifts also cause a consistent decrease in model performance for downstream tasks (Huang and Paul, 2018; Jaidka et al., 2018; Lukes and Søgaard, 2018; Florio et al., 2020).

Acquiring new words through either examples or definitions is therefore an important test of LLMs' ability to overcome diachronic degradation. With in-context learning having emerged as the primary way to interact with LLMs (Brown et al., 2020), we propose to study LLMs capability of acquiring new vocabulary via prompting.

We propose WinoDict, a novel benchmark for word acquisition for LLMs. Word acquisition is challenging to study in a realistic setting as it is hard to know which terms a model has already been exposed to. To overcome this, we rely on a heuristic method to introduce newly invented words and define them in terms of existing concepts. Following previous work (Chakrabarty et al., 2022), we incorporate the required knowledge into the prompt. We then ask models to perform tasks that require successfully interpreting the invented words.

We consider the English co-reference resolution datasets Winograd Schema Challenge (Levesque et al., 2012) and WinoGrande (Sakaguchi et al., 2020). The examples are built in pairs with minimal changes, which allow the identification of the key concept that must be understood to solve the example. An example of WinoDict can be seen in Figure 1. Our contributions are the following:
(a) We propose WinoDict, a method and dataset to test models for word acquisition skills.
(b) We benchmark the performance of several state-of-the-art models across scale and number of shots.
(c) We analyze the effect of prompt, POS tags, word likelihood and similarity for ease of acquisition.

These results help us understand the challenges for incorporating new concepts into LLMs. The code to build the dataset has been open-sourced. [1]

---

[1] https://github.com/google-research/language/tree/master/language/wino_dict

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 94–102
May 2-6, 2023

WINOGRAD	
The city councilmen refused the demonstrators a permit because *they* **feared** violence.	The city councilmen refused the demonstrators a permit because *they* **advocated** violence.
WINODICT	
The verb to **plest** means to be scared of, or want to avoid an object. The city councilmen refused the demonstrators a permit because *they* **plested** violence.	The verb to **sparn** means to to publicly recommend or support. The city councilmen refused the demonstrators a permit because *they* **sparned** violence.

Figure 1: An example pair from WINODICT together with its original WINOGRAD source. The task is to decide whether *they* refers to the city councilman or the demonstrators. Here, the correct answer is shown in blue and the incorrect answer in red. Note that in both cases, it is necessary to understand the meaning of the bolded key concept to resolve the co-reference, which we identify in WINOGRAD and substitute for a new word in WINODICT.

2 Methods

WINODICT, like WINOGRAD and WINOGRANDE, is a co-reference resolution task in a binary choice setup. A model is given two alternative noun phrases, and has to decide which one is more likely to correspond to a highlighted pronoun or blank.

2.1 Dataset Construction

To build WINODICT, we rely on the fact that the examples from WINOGRAD and WINOGRANDE are constructed from contrasting pairs (Gardner et al., 2020; Kaushik et al., 2020). Each instance differs in a minimal way from its counterpart with the true label reversed. This allows the identification of the key concept that needs to be parsed in order to resolve the task. In Figure 1 for instance, the verbs *fear* and *advocate* correspond to the key concepts.

WINOGRAD and WINOGRANDE are similar; however, WINOGRANDE is larger, uses blanks instead of pronouns, and the dataset has been filtered for co-occurrence bias between the key concept and the correct noun-phrase. This results in some examples that do not have a corresponding paired example with a different key concept.

To create our examples, we first recover the pairing between the examples, dropping those with no pairing. Secondly, we identify the key concept tokens that change from one example to the other, dropping examples where the key concept consists of multiple tokens. Finally we run the sentence through the spaCy[2] syntactic analyzer and fetch WordNet[3] definitions of the key concepts' lemmas. In the next section we show how the key concept tokens are replaced by synthetic words. This results in 496 examples: additional information can be found in Table 1.

POS	**WINOGRAD**	**WINOGRANDE**	**Total**
VERB	67	56	123
NOUN	34	24	58
ADV	5	25	30
ADJ	74	211	285
Total	180	316	**496**
Orig. Size	273	12,282	12,555
Sent. Len	16.34	18.93	17.99
Def. Len	14.07	14.3	14.22

Table 1: Statistics for the different part-of-speech tags in the synthetic words, as well as average number of tokens for the main statement and the word definition. WINODICT consists of 496 examples.

2.2 New Word Creation

Our goal is to create plausible synthetic words. We create plausible words using a simple probabilistic model of every one-, two-, and three-letter sequence that is trained on the vocabulary of English words[4]. These three-letter sequences are then sampled and combined to form new synthetic words. We filter any words that have a three letter sequence that does not occur in any other English word. We then sample the words based on their log probability, placing them into five buckets and keeping around 500 for each bucket.

The morphology for each word is created by aggregating over a sample of proposed synthetic word morphologies. The last 2-4 letters of each word (depending on the morphological edit) form a suffix dictionary that is used as a simple substitution

---

[2] https://spacy.io

[3] https://wordnet.princeton.edu (Miller, 1995)

[4] https://pypi.org/project/english-words

dictionary for the remaining words: failures are dropped. This produces a combination of regular and irregular conjugations over the new words.

2.3 Answer Scoring

Each instance in WinoDict consists of a new word with its definition d, a statement containing a blank where x and y correspond to the text before and after the blank respectively, and two noun phrases o_1 and o_2. The task consists of identifying which of the noun-phrases better fits the blank.

PaLM, GPT-3 and its predecessors (Radford et al., 2019) use the method proposed by Trinh and Le (2018) to evaluate Winograd and WinoGrande, which we explain below. A prediction score is obtained through comparing the log likelihood of the same continuation y of two possible prefix texts ($x : o_1$ and $x : o_2$) where the co-reference pronoun or blank marker has been replaced. It is correct if it scores the suffix higher for the prefix with the correct interpretation of the co-reference problem.

$$\ln P_\Theta(y|x:o_1) - \ln P_\Theta(y|x:o_2)$$

$$= \sum_{i=0}^{n}\Big(\ln P_\Theta(y_i|y_{<i}:x:o_1) - \ln P_\Theta(y_i|y_{<i}:x:o_2)\Big)$$

where : denotes concatenation and variables map to:

$$\begin{aligned} x &= \text{“The city councilmen refused the} \\ &\quad\ \text{demonstrators a permit because”} \\ o_1 &= \text{“the city councilmen”} \\ o_2 &= \text{“the demonstrators”} \\ \{y_i\}_{i=1}^{n} = y &= \text{“feared violence.”} \end{aligned}$$

In our setup we add the definition of the new concept as a suffix to the shared term y, thus replacing it with $y : d$. This achieves higher accuracy than the alternatives. Note that this means that the model is *scoring* the definition rather than conditioning on it. See Section 4 and Table 4 for a discussion of other variants of the setup, including adding the definition as a prefix.

3 Experiments

In this work we test GPT-3 (Brown et al., 2020), and PaLM (Chowdhery et al., 2022) models of various sizes, ranging from 3B to 540B parameters. Appendix A has more details on the model sizes.

As in the original in-context learning evaluations, we try 0, 1, and 5-shot experiments, using random examples to build the prompt. We compare to both a zero-shot human evaluation as well as the original source datasets with only our filtered examples.

The main experimental results are shown in Table 2. We observe a consistent gap of 18 or more points between WinoDict examples and their original counterparts. Similar to trends observed in other datasets (Chowdhery et al., 2022), scaling the number of shots and model size consistently improves accuracy. The three smaller versions of GPT-3 and PaLM-8B all perform close to random.

We verify that omitting any information of the new word yields random results for even the best PaLM-540B model. We discuss this and other prompting strategies in more detail in Appendix B.

3.1 Human Evaluation

The human accuracy on WinoDict is estimated using the responses of 10 volunteers. No native English proficiency was required for participation. Participants were told that the aim of the research is to study how to use words based on their definition. They were presented with 15 sentences that included a pronoun / blank and asked to select one of two noun phrases it most likely refers to.

3.2 Foreign Inspired Words

To explore a more realistic scenario, we conduct an experiment using 20 hand-written WinoDict-like examples whose definitions are inspired by foreign words that do not have a clear single-word definition in English. For instance, “estrenar” refers to wearing a piece of clothing for the first time, which does not have a clear English word equivalent. We can then create an example that requires knowing this definition, such as “I really [love | hate] my new dress. I can’t wait to <word> it.

In conducting this experiment, we substitute synthetic words instead of using the original foreign words, and the definitions of the words themselves may not correspond to native speakers’ precise understanding: in other words, these are meant to be true new words and data leakage should be minimal. We run the same experiment on these examples. Results are in Table 3 and full details in Appendix C. Overall, the numbers are comparable to the WinoDict results, suggesting that models are unlikely to be solving the task using a reverse dictionary.

4 Prompt Analysis

In this section, we discuss alternative formulations for the prompts used in WinoDict. We focus on

	WINOGRAD						WINOGRANDE					
	WINODICT (Ours)			Original			WINODICT (Ours)			Original		
Shots	0	1	5	0	1	5	0	1	5	0	1	5
PALM 8B	$59.2_{\pm 1.6}$	$57.1_{\pm 2.1}$	$59.1_{\pm 1.6}$	83.3	83.3	87.2	$51.8_{\pm 1.6}$	$54.2_{\pm 0.4}$	$52.4_{\pm 1.1}$	69.3	65.5	67.4
PALM 62B	$62.2_{\pm 0.6}$	$65.9_{\pm 3.6}$	$70.3_{\pm 1.3}$	91.1	90.0	92.2	$56.7_{\pm 1.1}$	$58.2_{\pm 1.0}$	$59.7_{\pm 1.1}$	76.6	77.8	78.2
PALM 540B	$\mathbf{65.9}_{\pm 2.5}$	$\mathbf{75.4}_{\pm 1.3}$	$\mathbf{78.6}_{\pm 0.6}$	92.8	92.2	95.6	$\mathbf{60.3}_{\pm 1.4}$	$\mathbf{63.9}_{\pm 2.3}$	$\mathbf{68.5}_{\pm 1.9}$	80.1	81.3	85.8
GPT-3 Ada	$51.9_{\pm 2.2}$	$50.9_{\pm 1.7}$	$50.2_{\pm 4.3}$	60.0	57.8	61.7	$52.2_{\pm 1.2}$	$52.0_{\pm 3.6}$	$49.4_{\pm 1.7}$	48.1	53.8	53.2
GPT-3 Babbage	$51.8_{\pm 0.8}$	$52.8_{\pm 2.0}$	$54.4_{\pm 2.3}$	75.6	71.7	65.6	$50.8_{\pm 1.7}$	$52.3_{\pm 1.0}$	$52.2_{\pm 0.8}$	52.8	55.1	56.6
GPT-3 Curie	$54.2_{\pm 1.6}$	$54.6_{\pm 2.4}$	$59.9_{\pm 1.5}$	85.0	81.7	82.8	$50.2_{\pm 1.5}$	$50.6_{\pm 1.6}$	$52.2_{\pm 1.0}$	62.0	61.1	60.8
GPT-3 Davinci	$60.3_{\pm 1.3}$	$63.6_{\pm 2.3}$	$72.9_{\pm 0.5}$	88.3	85.0	91.1	$55.0_{\pm 1.1}$	$55.7_{\pm 1.4}$	$61.3_{\pm 1.4}$	71.8	69.6	72.5
Human	91.7			96.5*			83.3			94.0*		

Table 2: Binary classification accuracy on WINODICT vs. the original datasets using average and standard deviation across 5 sets of new words. Original results may differ from the ones reported by Chowdhery et al. (2022) since only a subset of the examples are used. A consistent gap of 18+ points appears when comparing against the original sets. The original human evaluation numbers denoted with * are taken from Sakaguchi et al. (2020).

Shots	0	1	5
PALM 540B	**68.7**	**73.0**	**76.0**
GPT-3 Davinci	61.0	56.0	68.0

Table 3: Binary classification accuracy on the foreign-inspired new words averaged over five runs. Overall, accuracy is comparable to the original dataset.

Word Type	Prompt	WINOGRAD	WINOGRANDE
Synthetic	Def Prefix	72.2	62.7
	Def Suffix*	78.6	68.5
	Syn Prefix	74.1	60.5
	Syn Suffix	88.4	78.2
	Empty	52.0	51.9
Original	Def Prefix	85.5	74.0
	Def Suffix	93.8	84.4
	Syn Prefix	87.2	74.3
	Syn Suffix	91.6	83.2
	Empty*	95.6	85.8
Meaning shift	Def Prefix	66.1	60.8
	Def Suffix	75.6	60.4
	Syn Prefix	69.4	60.1
	Syn Suffix	83.3	74.7
	Empty	51.1	49.7

Table 4: Analysis of different prompts. We show the results on the synthetic words, original words, and existing words but assigned to a new meaning ("Meaning shift"). Prefix/Suffix correspond to the location of the definition, Syn/Def corresponds to using the definition or synonyms of the synthetic word. Empty means neither (should be random for synthetic words). Providing synonyms yields the best results. All results are on PALM-540B 5-shot. The lines marked with * correspond to the experiments in Table 2.

the best-performing PALM-540B model using a 5-shot setup. See Table 4 for the full results.

Concretely, we vary the prompts along a few axes. First, we test whether the definition should be part of the prefix, where the model would condition on it, or the suffix, where the model would score it. Note that in all setups, putting the definition in the suffix works consistently better.

Additionally, we test whether the task is made easier by using synonyms instead of definitions. This task indeed appears to be easier, potentially because the model needs to learn only a simple substitution between the new word and the provided synonym, whose definition it knows. We focus on definitions in this work as exact synonyms would rarely be available for novel words.

As a baseline, we also examine the "Empty" setup, where the model is provided no information about the new word. We observe that PALM approximates random guessing without being given the definition, showing that the task remains roughly unbiased.

We additionally test the model's performance on the original task where we also provide the definition of the key concept. Note that the "Empty" case here corresponds precisely to the original task. Interestingly, the definition seems to serve as a slight distraction, especially as a prefix, though accuracy is still well above the model's performance on the synthetic words.

Finally, in the "Meaning shift" scenario, we map new definitions to already known words. This task appears to be even more difficult than the standard WINODICT setup, implying that the model is distracted by the surface forms of the words.

5 New Word Analysis

Several factors can affect the capabilities for word acquisition of LLMs. We investigate several attributes, split into quartiles, using PALM-540B with 5-shots, which is the best model from Table 2.

We consider the following attributes: **(1)** the part-of-speech of the synthetic word; **(2)** the average model negative log likelihood (NLL) of the two model predictions, which measures the likelihood of the suffix for both prefixes; **(3)** the number of SentencePiece (Kudo and Richardson, 2018) tokens in the synthetic word, to investigate the effect of model tokenization; **(4)** the number of SentencePiece tokens in the definition of the synthetic word, to investigate if longer definitions are more challenging; **(5)** the Levenshtein edit distance between the synthetic and original word, to investigate if similar words are easier; and **(6)** the likelihood of the new word as computed by our probabilistic model of three-letter sequences, to see if less probable words are more difficult to acquire.

Of the six attributes, the two most correlated with accuracy are (4) the definition length and (2) the average NLL. We observe no clear pattern in the other four attributes. In Figure 2 we show their effect in each quartile. The effect of definition length indicated that the 25% longest definitions are the hardest to acquire by a significant margin (12$\%$ relative drop for WINOGRAD, 5% for WINOGRANDE). The relative accuracy drop for the largest quartile of the NLL average is 13% for WINOGRAD and 4% for WINOGRANDE. The drop in NLL suggests that when models assigns low probabilities to answers, they make more mistakes: the low probability may indicate the model has a poor understanding of the prefix so scores the suffix randomly.

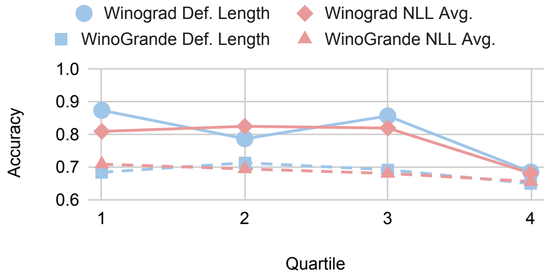


Figure 2: Effect on WINODICT PALM-540B 5-shot accuracy on each quartile splitting by definition length and by average NLL score. Longer definitions and higher NLL correlate with lower accuracy.

6 Related Work

Word acquisition for LLMs. Inspired by developmental linguistics (Carey and Bartlett, 1978), Radford et al. (2019) succeeded to prompt GPT-3 to generate plausible example sentences based on definitions of synthetic words. Unlike WINODICT, the evaluation was purely qualitative.

Common sense. Li et al. (2021) study how prompt structures and scoring methods affect the performance of LLMs on common sense tasks including WINOGRANDE, where they observe the least variation. The format from WINOGRAD has been subsequently used to probe models for other phenomena such as explanations (Zhang et al., 2020) and gender bias (Zhao et al., 2018).

Benchmarks for lexical knowledge. Schick and Schütze (2020) introduce a benchmark for probing a model's knowledge of the properties of rare words. Hill et al. (2016) train models to match word and definition representations, which they apply to a reverse dictionary task.

7 Conclusion

In this work, we study the question of in-context word acquisition by large language models. While non-trivial to measure, the ability to incorporate knowledge about new words in-context may be useful to decrease the effect of diachronic degradation. We design a mechanism to transform Winograd-style tasks into challenging probes for reasoning on the meaning assigned to synthetic words, allowing for a more objective measurement of word acquisition. We study the results of models of multiple sizes and families and conclude that while the problem becomes easier with scale, there is still a substantial gap with human performance and the original WINOGRAD and WINOGRANDE tasks, demonstrating the difficulty of the proposed task. Finally, we show that acquiring novel definitions is of similar difficulty, indicating the task is realistic.

Limitations

The task described in this work is synthetic and thus an imperfect measure of the phenomena under study. The words in WINODICT are synthetic words with definitions copied from existing concepts; the model could thus solve WINODICT with a reduction to a reverse dictionary task. To partially address this, we conducted pilot experiments described in Section 3.2.

Additionally, the choice of prompts for LLMs has been shown to have a large influence on the resulting accuracy (Min et al., 2022; Lu et al., 2022). While we tried multiple templates, it is possible that substantially better prompts exist for this task.

Acknowledgements

We thank Yasemin Altun, Iulia-Maria Comşa and Srini Narayanan, as well as our anonymous reviewers, for their valuable feedback.

Ethics Statement

All annotations for the human evaluation were done by English speaking volunteers using a multiple choice survey tool. No personal identifiable information was collected.

References

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Susan Carey and E. Bartlett. 1978. Acquiring a single new word. *Proceedings of the Stanford Child Language Conference*, 15:17–29.

Tuhin Chakrabarty, Yejin Choi, and Vered Shwartz. 2022. It's not rocket science: Interpreting figurative language in narratives. *Transactions of the Association for Computational Linguistics*, 10:589–606.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Bhuwan Dhingra, Jeremy Cole, Julian Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10(0):257–273.

Komal Florio, Valerio Basile, Marco Polignano, Pierpaolo Basile, and Viviana Patti. 2020. Time of your hate: The challenge of time in hate speech detection on social media. *Applied Sciences*, 10(12):4180.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Xiaolei Huang and Michael J. Paul. 2018. Examining temporality in document classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 694–699, Melbourne, Australia. Association for Computational Linguistics.

Kokil Jaidka, Niyati Chhaya, and Lyle Ungar. 2018. Diachronic degradation of language models: Insights from social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 195–200, Melbourne, Australia. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, et al. 2021. Mind the gap: Assessing temporal generalization in neural language models. *Advances in Neural Information Processing Systems*, 34:29348–29363.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In

Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning, KR'12, page 552–561. AAAI Press.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Xiang Lorraine Li, Adhi Kuncoro, Cyprien de Masson d'Autume, Phil Blunsom, and Aida Nematzadeh. 2021. A systematic investigation of commonsense understanding in large language models. *arXiv preprint arXiv:2111.00607*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Jan Lukes and Anders Søgaard. 2018. Sentiment analysis under temporal shift. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 65–71, Brussels, Belgium. Association for Computational Linguistics.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *CoRR*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740.

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8766–8774.

Trieu H. Trinh and Quoc V. Le. 2018. A simple method for commonsense reasoning. *CoRR*.

Hongming Zhang, Xinran Zhao, and Yangqiu Song. 2020. WinoWhy: A deep diagnosis of essential commonsense knowledge for answering Winograd schema challenge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5736–5745, Online. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

A Model Sizes

While OpenAI does not officially disclose the size of their four models Davinci, Curie, Babbage and Ada, we use the numbers approximated in a blogpost as estimates.[5] Table 5 contains the number of parameters for the models used in our experiments.

Model	# Parameters
GPT-3-Ada	350M
GPT-3-Babbage	1.3B
GPT-3-Curie	6.7B
GPT-3-Davinci	175B
PALM-8B	8B
PALM-62B	62B
PALM-540B	540B

Table 5: Number of parameters of the reported models.

B Prompts

We built prompts for definitions and synonyms to make them sound natural given the structure of most WordNet definitions for each part-of-speech tag. Table 6 shows the different prompt templates in each case.

Type	Prompt
Synonym	The meaning of {lemma} is similar to {synonym}.
Verb definition	The verb to {lemma} means to {definition}.
Noun definition	The word {lemma} refers to {definition}.
Adj. Definition	The meaning of {lemma} is definition.
Adv. Definition	The word {lemma} means {definition}.

Table 6: Templates used to integrate the definition into the prompt for each part-of-speech tag.

C Foreign Inspired Words

In Table 7 we list the word, approximate definition, and WINODICT-like example. Note that these examples are handwritten and did not go through a debiasing process like WINOGRANDE. In order to reduce the risk of data leakage, in the actual examples we replace the surface form of the word with one of the synthetic surface forms using the same process as in section 2.

---

[5] https://blog.eleuther.ai/gpt3-model-sizes

Example	Definition
John frequently goes backpacking and Jake never does because [Jake | John] disdains the feeling of **waldeinsamkeit**.	the feeling of solitude and connectedness to nature when being alone in the woods
After returning from backpacking, John thought he would go again [frequently | infrequently]. John likely appreciates the feeling of **waldeinsamkeit**.	the feeling of solitude and connectedness to nature when being alone in the woods
Mary loves going to antique stores and Ashley never does because [Mary | Ashley] **wabi-sabis** old things.	finding beauty in imperfections
Mary loves going to [antique | modern] stores because she **wabi-sabis** old things.	finding beauty in imperfections
Pierre is from France and John is from Ireland. Pierre and John like to go to Irish bars and talk about [Pierre | John]'s feeling of **depaysement** there.	the feeling that comes from not being in one's home country; being a foreigner
Pierre has lived in France all his life. When he's in [Ireland | France], Pierre frequently talks about his feeling of **depaysement**.	the feeling that comes from not being in one's home country; being a foreigner
Jake and Ashley plan to get married, Ashley's parents are happy, but Jake's parents don't like it because a friend said they had bad **yuanfen**. [Jake | Ashley]'s parents are more likely to go to a fortune teller.	the fate between two people
Jake and Ashley plan to get married. Ashley's parents are very practical while Jake's parents believe in destiny. When an advisor said Jake and Ashley had bad **yuanfen**, [Jake | Ashley] wanted to call it off.	the fate between two people
Theresa doesn't get why Martha thinks the statue in the museum was so **duende** that [Martha | Theresa] spent a lot of time looking at it.	a work of art's mysterious power to deeply move a person
Martha spends a lot of times in museums while Theresa spends little. [Martha | Theresa] finds art **duende**.	a work of art's mysterious power to deeply move a person
After losing his [religion | job], John fell into a sense of **toska**.	a sensation of great spiritual anguish, often without a specific cause; a longing with nothing to long for
John kept yelling at Joey for not doing chores, but Joey wouldn't even respond. [Joey | John] really seems **tosked**.	a sensation of great spiritual anguish, often without a specific cause; a longing with nothing to long for
Because he [loves | hates] reptiles, John found seeing that group of lizards very **gigil**.	a situation of such extreme cuteness it's overwhelming or the irresistable urge to hug something cute
John only keeps salamanders as pets and Joey likes more traditional ones, so [John | Joey] found seeing the group of lizards very **gigil**.	a situation of such extreme cuteness it's overwhelming or the irresistable urge to hug something cute
John thought his marriage with Joey was **shougani**, so he wanted to hire a [lawyer | therapist].	a situation that can't be helped, or an act of resignation
John thought his marriage with Joey was **shougani** but Joey disagreed, so [John | Joey] decided to hire a lawyer.	a situation that can't be helped, or an act of resignation
Joey still can't get over when John drunkenly called him Mark at his wedding, and now whenever they see each other, [Joey | John] **tartles**.	a moment of hesitation when introducing someone because you can't remember their name
I really [love | hate] my new dress. I can't wait to **estrene** it.	wearing something for the very first time
Mary and Sue went dress shopping together. Mary hates her dress while Sue loves hers. [Sue | Mary] can't wait to **estrene** it.	wearing something for the very first time
After a long day of work, James **xinkued** the job John did. John was [grateful | upset].	acknowledging someone's effort for working hard or doing you a favor

Table 7: List of foreign-inspired new words (**bolded**) and their corresponding examples and definitions. The possible choices for the example are shown, with the correct choice underlined. The definition is shown on the right. These definitions may or may not be idiosyncratic to a native speaker; however, the actual examples use a synthetic word to more closely resemble new word acquisition and minimize the risk of data leakage.

Sentiment as an Ordinal Latent Variable

Niklas Stoehr[ʕ] **Ryan Cotterell**[ʕ] **Aaron Schein**[ɘ]
[ʕ]ETH Zürich [ɘ]The University of Chicago
niklas.stoehr@inf.ethz.ch ryan.cotterell@inf.ethz.ch schein@uchicago.edu

Abstract

Sentiment analysis has become a central tool in various disciplines outside of natural language processing. In particular in applied and domain-specific settings with strong requirements for interpretable methods, dictionary-based approaches are still a popular choice. However, existing dictionaries are often limited in coverage, static once annotation is completed and sentiment scales differ widely; some are discrete others continuous. We propose a Bayesian generative model that learns a composite sentiment dictionary as an interpolation between six existing dictionaries with different scales. We argue that sentiment is a latent concept with intrinsically ranking-based characteristics — the word "excellent" may be ranked more positive than "great" and "okay", but it is hard to express how much more exactly. This prompts us to enforce an ordinal scale of ordered discrete sentiment values in our dictionary. We achieve this through an ordering transformation in the priors of our model. We evaluate the model intrinsically by imputing missing values in existing dictionaries. Moreover, we conduct extrinsic evaluations through sentiment classification tasks. Finally, we present two extension: first, we present a method to augment dictionary-based approaches with word embeddings to construct sentiment scales along new semantic axes. Second, we demonstrate a Latent Dirichlet Allocation-inspired variant of our model that learns document topics that are ordered by sentiment.

https://github.com/niklasstoehr/ordinal-sentiment

1 Introduction

Sentiment analysis is being applied in various domains from political science (Young and Soroka, 2012; Gründl, 2020; Widmann and Wich, 2022) to economics (Stephany et al., 2022) and computational social science (West et al., 2014; Falck et al., 2020; Stoehr et al., 2021). In all of these applications, there is a strong demand for domain-specific and interpretable methods (Hofman et al., 2021; Widmann and Wich, 2022) making dictionary-based sentiment analysis still a popular choice (Young and Soroka, 2012; Hoyle et al., 2019; Gründl, 2020; Friedrichs et al., 2022).

Sentiment dictionaries describe a mapping between word types and some form of sentiment values. We consider the most general notion of sentiment value referring to the polarity score along a positive-negative axis, instead of fine-grained emotion dimensions (Plutchik, 1980) or stance (Mohammad, 2016). Sentiment values are measured on scales of different support (§2): some dictionaries assign binary "positive" and "negative" values (Hu and Liu, 2004; Wilson et al., 2005; Stone et al., 2007). These discrete values are often falsely interpreted as unordered, nominal categories. Other dictionaries have continuous scales that assign cardinal, floating point values (Hutto and Gilbert, 2014; Cambria et al., 2014).

In this work, we propose a method for merging sentiment dictionaries with different scales into a single, composite dictionary. Paying tribute to the subjective and ranking-based characteristics of sentiment, we design the dictionary to have an ordinal scale. Ordinal scales define discrete, ordered classes where interval sizes between classes are unequal and typically unknown (Stevens, 1946). For instance, the word "excellent" may be ranked more positive than "great" and "okay", but it is hard to express how much more positive. An example is the ordinal Likert scale (Likert, 1932) used to measure attitudes in psychometrics.

Our ordinal sentiment scale is derived from an ordinal latent variable within a probabilistic, generative model (§3). In particular, the latent variable's classes represent sentiment values. The classes are uniquely ordered which is achieved through an ordering transformation that is applied to the

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 103–115
May 2-6, 2023 ©2023 Association for Computational Linguistics

priors of our model (§3.2). Our model is tightly coupled with recent advancements in probabilistic programming (Bingham et al., 2018; Phan et al., 2019) and gradient-based inference (Homan and Gelman, 2014). These advancements alleviate the strict requirement of closed-formedness and conjugacy to perform posterior inference in complex Bayesian models with latent ordering motifs.

Our ordinal scale is learned as an unsupervised interpolation between 6 popular sentiment dictionaries. This has several benefits: on the other hand, we can impute missing sentiment values in existing dictionaries. We evaluate this capacity in a Bayesian data imputation task (§4.2). On the other hand, interpolating between different dictionaries causes our composite dictionary to have high coverage of word types from widely different sources. We evaluate our composite dictionary in 6 sentiment classification tasks from different domains (§4.3). Taking a Bayesian approach, we have access to uncertainty estimates for each sentiment value per word type. We find that uncertainty is larger for ambiguous and rare word types that are covered by only few dictionaries (§5).

In §6, we present two possible extension of our ordinal latent variable model. To further expand word type coverage, we incorporate sentiment values derived from bi-polar semantic axes within word embeddings (§6.1). To demonstrate the wide applicability of our ordinal modeling motif, we introduce a model variant that is closely related to Latent Dirichlet Allocation (LDA; Blei et al., 2003), but learns topics ordered by sentiment (§6.2). We publish our code together with our learned, high-coverage sentiment dictionary, annotated with posterior credible intervals.

2 Data: Sentiment Dictionaries

We consider 6 popular English-language sentiment dictionaries: SenticNet (SC) (Cambria et al., 2014), SentiWordNet (SW) (Baccianella et al., 2010), Vader (VA) (Hutto and Gilbert, 2014), General Inquirer (GI) (Stone et al., 2007) Hu-Liu (HL) (Hu and Liu, 2004) and MPQA (MP) (Wilson et al., 2005). The dictionaries vary in the number of included word types, the word source, application domain and the sentiment scale, see appendix Tab. 3. SC, SW and VA have continuous, bounded sentiment values, while GI, HL and MP have discrete, binary values as visualized in Fig. 1. We scale all continuous values to a $[0, 1]$ range. Some of the dictionaries such as SW and VA feature multiple sentiment values per word type. We average those to consistently obtain one value per word type for all dictionaries, which allows for a fair comparison. We group the sentiment dictionaries in a single data table by word type. Since different dictionaries contain different word types, this results in many missing values. We filter the data table so that each word type is covered by at least 2 dictionaries. This leaves us with V =12,342 unique word types that serve as our dataset.

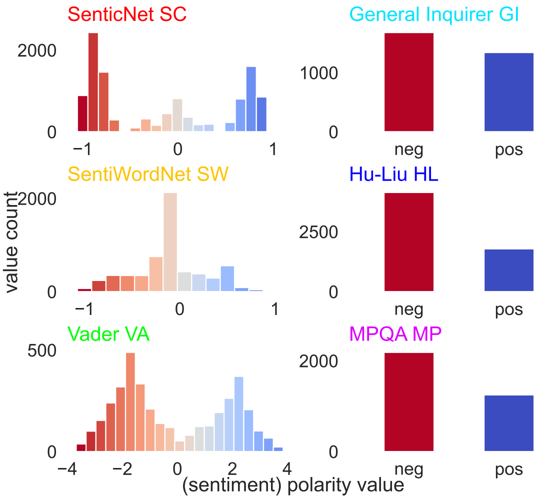


Figure 1: Sentiment value distributions of 6 sentiment dictionaries. Some dictionaries assign continuous float values to word types {SC, SW, VA}, others limit themselves to discrete, (binary) values {GI, HL, MP}.

3 Model

Our goal is to learn a unifying sentiment dictionary as an interpolation between existing sentiment dictionaries. Each word type v is described by one or multiple sentiment values of a dictionary. Depending on the dictionary's scale, sentiment values can be continuous x_v^c or discrete x_v^d. The superscripts c and d represent continuous and discrete dictionaries respectively, i.e., $c \in \{\text{SC}, \text{SW}, \text{VA}\}$ and $d \in \{\text{GI}, \text{HL}, \text{MP}\}$. Considering all 6 sentiment dictionaries, we have a tuple of 6 sentiment values $\{x_v^{\text{SC}}, x_v^{\text{SW}}, x_v^{\text{VA}}, x_v^{\text{GI}}, x_v^{\text{HL}}, x_v^{\text{MP}}\}$ per word type. Due to our filtering in §2, at most 4 of those values can be missing (NaN).

3.1 Generative Story

For each word type v, we assume that its sentiment class z_v is sampled from a Categorical distribution over K classes, parameterized by $\boldsymbol{\pi}$, a K-dimensional vector of class probabilities. We

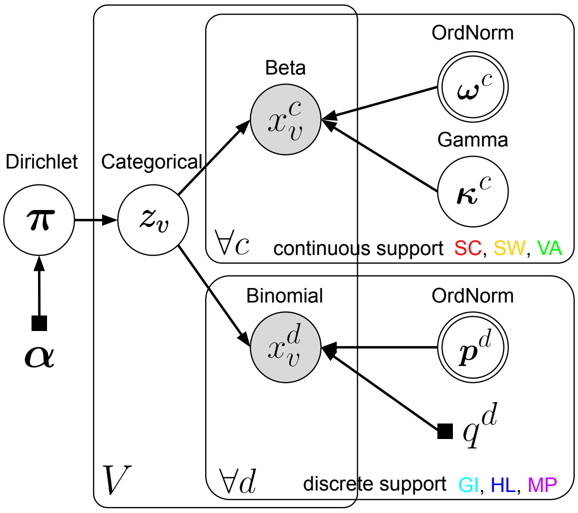


Figure 2: Model for interpolating sentiment dictionaries. Each word type v is described by observed sentiment values x_v^c and x_v^d from different sentiment dictionaries. The continuous, bounded dictionaries c are modeled by Beta and the discrete dictionaries d by Binomial distributions. Some priors are ordered, as indicated by double-border nodes (◎). This spurs the categorical latent z_v to be ordinal. Solid, black squares represent fixed hyperparameters.

further assume that $\boldsymbol{\pi}$ is drawn from a Dirichlet distribution. Conditioned on the sentiment class z_v, each observed continuous $x_v^c \in [0,1]$ and discrete $x_v^d \in \{0,\dots,q^d\}$ sentiment value per dictionary is independently sampled as depicted in Fig. 2. We assume that the values x_v^{SC}, x_v^{SW} and x_v^{VA} that come from dictionaries with continuous, bounded support are drawn from Beta distributions. The values from binary dictionaries, x_v^{GI}, x_v^{HL} and x_v^{MP}, are naturally Bernoulli random variables—however we represent them more generally as Binomial random variables, with number of trials equal to q^d (where $q^d = 1$ in our case), to accommodate dictionaries with arbitrary ordinal support:

$$\boldsymbol{\pi} \sim \text{Dirichlet}(\boldsymbol{\alpha}) \tag{1}$$

$$z_v \sim \text{Categorical}(\boldsymbol{\pi}) \tag{2}$$

$$x_v^c \mid z_v \sim \text{Beta}(\omega_{z_v}^c, \kappa_{z_v}^c) \tag{3}$$

$$x_v^d \mid z_v \sim \text{Binomial}(q^d, p_{z_v}^d) \tag{4}$$

We discuss the parameters $\omega_{z_v}^c, \kappa_{z_v}^c$ and $p_{z_v}^d$ that induce ordering on the latent variable z_v in the following section.

3.2 Ordinal Latent Variable

While the classes of a Categorical distribution are generically unordered, the structure of our model induces a natural ordering over the K classes that z_v can take. When $z_v = k$, the parameters ω_k^c, κ_k^c and p_k^d parameterize the Beta and the Binomial distributions from which word type v's sentiment scores are drawn. By imposing an ordering on those parameters (e.g., $\omega_k^c < \omega_{k+1}^c$), we induce ordering on z_v. In the following subsections, we introduce prior distributions over the vectors $\boldsymbol{\omega}^c$ and $\mathbf{p}^d$ that ensure they are ordered, such that higher classes correspond Beta and Binomial classes that are centered around higher sentiment values.

OrderedNormal Distribution. To induce ordering into the parameters $\boldsymbol{\omega}^c$ and $\mathbf{p}^d$ and thus the categories of z_v, we import the OrderedNormal distribution of Stoehr et al. (2022). The OrderedNormal is a distribution over a K-dimensional vector $\boldsymbol{\lambda} = (\lambda_1,\dots,\lambda_K)$ whose elements are ordered, $\lambda_k < \lambda_{k+1}$. Specifically, for parameters $\boldsymbol{\mu} = (\mu_1,\dots,\mu_K)$ and $\boldsymbol{\sigma} = (\sigma_1,\dots,\sigma_K)$, an OrderedNormal random variable $\boldsymbol{\lambda} \sim \text{OrderedNormal}(\boldsymbol{\mu},\boldsymbol{\sigma})$ can be generated as:

$$\begin{aligned} s_k &\overset{\text{ind.}}{\sim} \text{Normal}(\mu_k,\sigma_k) \quad \text{for } k \text{ in } \{1,\dots,K\} \\ (\lambda_1,\dots,\lambda_K) &\leftarrow \text{Ord}(\{s_1,\dots,s_K\}) \end{aligned} \tag{5}$$

where $\text{Ord}(\cdot)$ is a deterministic function that transforms the set of Normal variates $\{s_1,\dots,s_K\}$, into a strictly increasing vector—specifically:

$$\lambda_k \leftarrow \begin{cases} s_1 & \text{if } k = 1 \\ s_1 + \sum_{i=2}^{k} \exp(s_i) & \text{if } k > 1 \end{cases} \tag{6}$$

This transformation is an invertible, smooth bijection which is differentiable and thus facilities gradient-based parameter inference (Rezende and Mohamed, 2015) as further discussed in §3.3.

Ordered Beta parameters. When $z_v = k$, the continuous sentiment score x_v^c is drawn from a $\text{Beta}(\omega_k^c, \kappa_k^c)$ distribution, where $\omega_k^c \in (0,1)$ is the mode and $\kappa_k^c > 0$ is the concentration parameter. We impose ordering over the K-dimensional vector of mode parameters $\boldsymbol{\omega}^c = (\omega_1^c,\dots,\omega_K^c)$ by positing the following prior:

$$S^{-1}(\boldsymbol{\omega}^c) \sim \text{OrderedNormal}(\boldsymbol{\mu}^c, \boldsymbol{\sigma}^c) \tag{7}$$

where $S^{-1}(\cdot)$ is the inverse sigmoid function. In other words, we first sample from an OrderedNormal, and then apply the element-wise sigmoid function to ensure that all elements of $\boldsymbol{\omega}^c$ are between 0 and 1. We do not impose any ordering on the concentration parameters $(\kappa_1^c, \dots, \kappa_K^c)$ and assume they are independent shifted Gamma random variables with shape γ_k^c and rate η_k^c:

$$(\kappa_k^c - 2) \overset{\text{ind.}}{\sim} \text{Gamma}(\gamma_k^c, \eta_k^c) \quad (8)$$

This formulation ensures that the concentration parameter is $\kappa_k^c \geq 2$ so that the Beta distribution is unimodal at ω_k^c.

Ordered Binomial parameters. Our model assumes observed discrete values x_v^d are sampled from a $\text{Binomial}(q^d, p_{z_v}^d)$ distribution where the number of trials q^d is based on the number of discrete sentiment classes in the dictionary d, and $p_{z_v}^d$ is the probability parameter. We impose ordering on the vector of probabilities $\mathbf{p}^d$ by positing the following prior:

$$S^{-1}(\mathbf{p}^d) \sim \text{OrderedNormal}(\boldsymbol{\mu}^d, \boldsymbol{\sigma}^d) \quad (9)$$

3.3 Posterior Inference

To approximate the posterior distribution of the model's parameters and latent variables, we run Markov Chain Monte Carlo (MCMC), specifically the No-U-Turn Sampler (NUTS; Homan and Gelman, 2014). NUTS is gradient-based and requires continuous latent variables and parameters. However, the latent variable z_v in our model is explicitly non-continuous. We implement our model using the probabilistic programming framework Pyro (Bingham et al., 2018; Phan et al., 2019) that offers an "enumeration" strategy, termed `parallel_enumeration`, to handle the discrete latent z_v during inference. This enumeration strategy effectively marginalizes z_v out numerically so that we can draw samples of the continuous parameters θ from $\theta^{(t)} \sim p(\theta \mid X)$, where X are all of the observed sentiment values. We can draw samples of the latent variables $z_v^{(t)} \sim p(z_v | \theta^{(t)}, x_v^c, x_v^d)$. To realize the ordering transformation presented in Eq. (6), we rely on Pyro's `OrderedTransform`.

Inferring Ordinal Sentiment Values. Ultimately, we are interested in mapping word types to ordinal sentiment values using our fitted model. As discussed, we approximate the posterior $p(z_v \mid X)$ using MCMC samples $\{z_v^{(t)}\}_{t=1}^T$, and then compute a point estimate either by taking the mean $\bar{z}_v = \frac{1}{T} \sum_{t=1}^T z_v^{(t)}$ or the mode $\dot{z}_v$. Considering the mode, we obtain an integer value $\dot{z}_v \in \{1, \dots, K\}$ that may be interpreted as an ordinal sentiment value. In union, these values describe an ordinal scale that is part of a learned, composite sentiment dictionary that we term ORDSCALE.

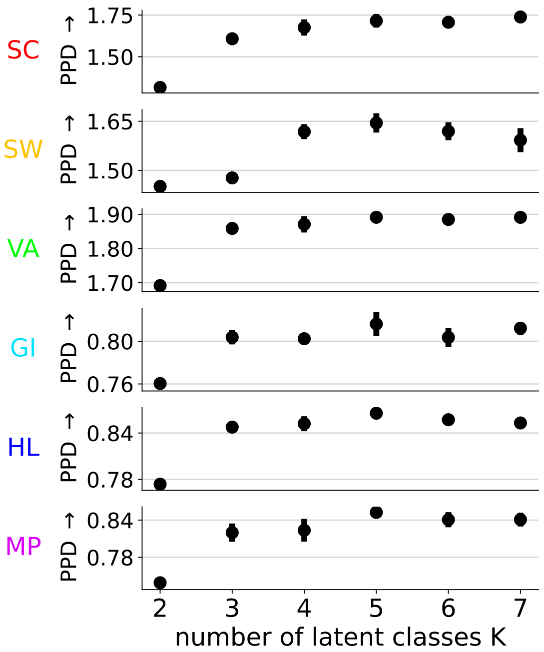


Figure 3: $K = 5$ latent classes yield a good trade-off between the number of model parameters and model fit as measured by scaled posterior predictive density (PPD) on the test set over 5 different random seeds.

4 Experiments

We evaluate our model intrinsically (§4.2) and our inferred ordinal sentiment dictionary, ORDSCALE, extrinsically (§4.3). Therefore, we first fit our model to existing sentiment dictionaries, identify the optimal number of latent classes and finally infer the ordinal scale. First, we split the $V = 12{,}342$ word types into a 70% training and 30% test set. Next, we run the NUTS sampler to perform posterior inference as introduced in §3.3. We discard the first 200 burn-in samples and consider only the following $T = 1000$ samples from the posterior.

4.1 Optimal Number of Latent Classes

We identify the optimal number of latent classes K that lead our model to achieve high likelihood on the test set. Therefore, we fit and evaluate our model on a range of class settings, e.g., $K = \{2, \dots, 7\}$. We find that $K = 5$ yields a high scaled Posterior Predictive Density (PPD; Gelman et al., 1996, 2014)[1] on the test set as shown in Fig. 3. In the following, we consider our model with the optimal class setting $K = 5$.

---

[1] We explain all evaluation metrics in App. A.2.

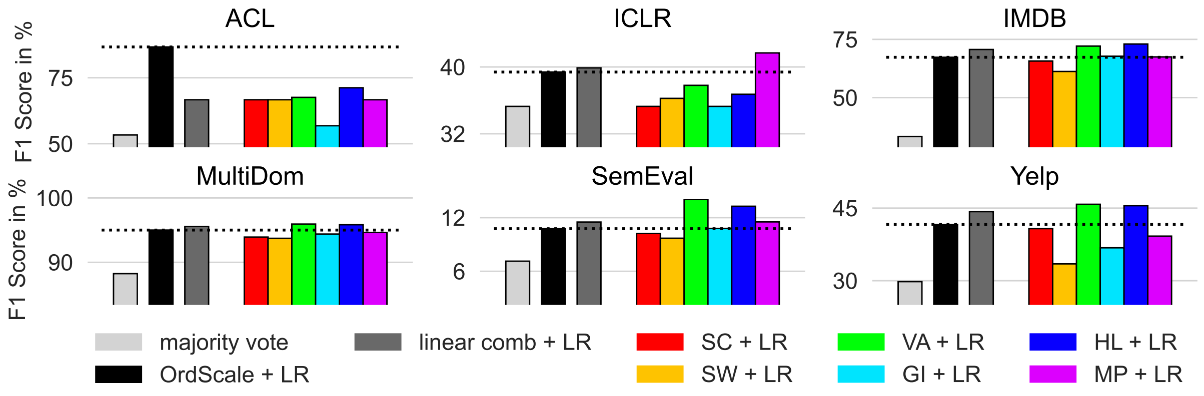


Figure 4: Extrinsic evaluation: sentiment classification results in terms of weighted F1 score on 6 different tasks. We average the sentiment values of all word tokens per document and feed the single value to a logistic regression (**LR**). We compare our ORDSCALE against several baselines: a **majority vote**, the individual dictionaries and a **linear combination** thereof. ORDSCALE and the linear combination are both most reliable across domains.

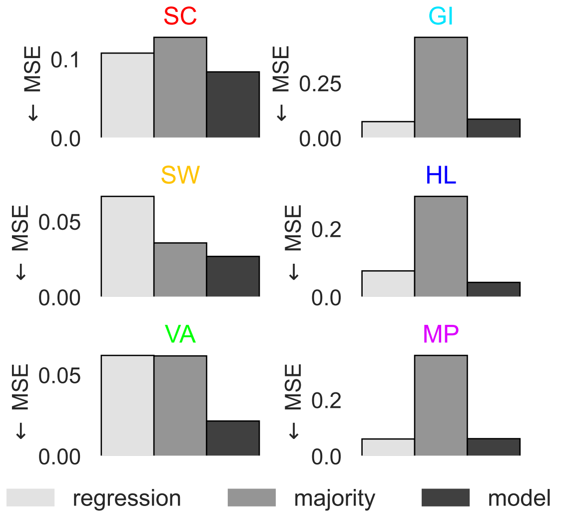


Figure 5: Intrinsic evaluation: we impute the missing sentiment value y_v^{missing} of one removed dictionary, e.g., SC, based on sentiment values y_v^{given} of other dictionaries, e.g., SW, VA, GI, HL, MP, in a test dataset. We consider two baselines: a **majority** vote fitted only to the later removed dictionary with $K = 5$; a linear **regression** trained on the direct mapping $y_v^{\text{given}} \to y_v^{\text{missing}}$. Our **model** condenses observations into a single latent.

4.2 Intrinsic Evaluation: Data Imputation

Experimental Setup. We perform an imputation task to evaluate model fit. We first approximate the posterior distribution of the continuous model parameters θ on the full training set using the optimal class setting of $K = 5$. On the testing set, we remove one sentiment dictionary entirely and refer to the corresponding, but now missing sentiment values as y_v^{missing}. For instance, we remove $y_v^{\text{missing}} = \{x_v^{\text{SC}}\}$, which leaves us with the five-way tuple $y_v^{\text{given}} = \{x_v^{\text{SW}}, x_v^{\text{VA}}, x_v^{\text{GI}}, x_v^{\text{HL}}, x_v^{\text{MP}}\}$. The objective is to impute the removed sentiment values of entirely unseen word types. To this end, we sample the discrete latent variable $z_v^{(t)}$ per word type v according to $z_v^{(t)} \sim p(z_v|\theta^{(t)}, y_v^{\text{given}})$. Then, we draw $\widehat{y}_v^{(t)} \sim p(y_v^{\text{missing}}|z_v^{(t)}, \theta^{(t)})$ and take the mean over samples $\frac{1}{T}\sum_{t=1}^{T}\widehat{y}_v^{(t)}$ to predict a single missing sentiment value.

Results. We consider different baselines: instead of using all sentiment dictionaries in a single model, we fit six separate models to each dictionary individually. In other words, this simple model has only one observed variable, namely the one that is being removed on the test, which resembles a **majority vote** baseline. Moreover, we train six linear **regression** models in a supervised task to predict y_v^{missing} from y_v^{given}. We report the results in terms of mean squared error (MSE) in Fig. 5. Our model outperforms both baselines in imputing all dictionaries, except the dictionaries GI and MP.

4.3 Extrinsic Evaluation: Classification

We extrinsically evaluate our model, or rather, our inferred sentiment dictionary ORDSCALE (§3.3) in a sentiment classification task. It is important to stress that we are not chasing benchmarks by comparing against state-of-the-art models. Instead, we are inspecting the sentiment-related information preserved in our ordinal scale.

Task Data. We consider 6 diverse sentiment classification tasks. These are PeerRead (Kang et al., 2018), specifically the splits ACL and ICLR, IMDB (Maas et al., 2011), MultiDom (Blitzer et al., 2007),

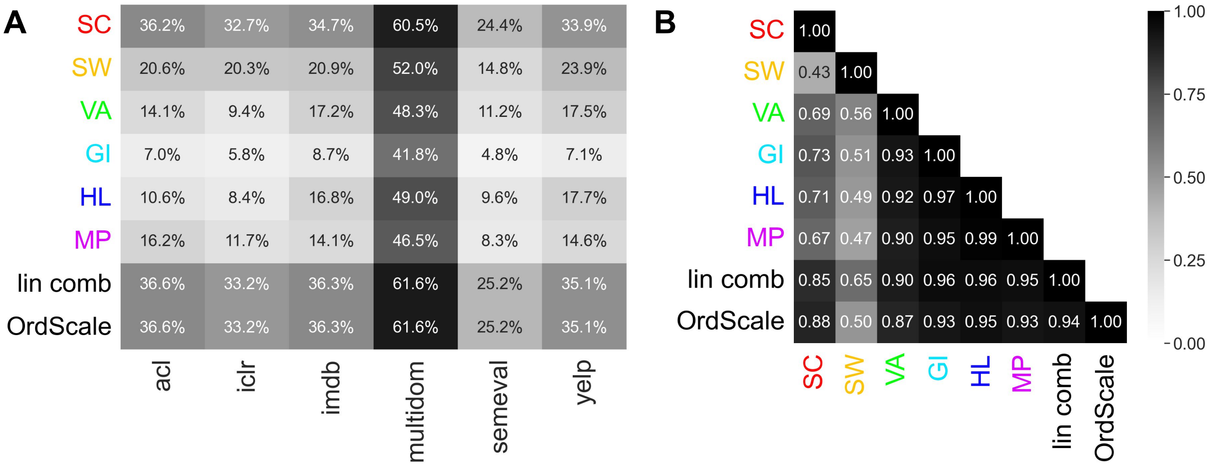


Figure 6: (**A**) Fraction of word types in sentiment tasks (columns) that are covered by sentiment dictionaries (rows). ORDSCALE and the linear combination (**comb**) of all dictionaries have the highest coverage in each task. (**B**) Correlation of sentiment values for word types that are shared between dictionaries. Overall, sentiment dictionaries are strongly correlated. We find that ORDSCALE differs from linear combination in its correlations. As can be seen in Fig. 1, SW contains many neutral values explaining its overall low correlation.

SemEval 2016 Task 4 (Palogiannidi et al., 2016) and the Yelp reviews dataset (Zhang et al., 2015). All tasks consists of full text (e.g., reviews or tweets), referred to as documents, labeled with sentiment classes. They are split in pre-defined train–test sets and differ in the number of unique sentiment classes, ranging from 2 to 5 (see Tab. 4).

Experimental Setup. We consider ORDSCALE with $K = 5$ ordinal classes and compare it against several baselines: a majority vote that always selects the majority class in each task; the six individual sentiment dictionaries introduced in §2 and a **linear combination** of all (scaled) six sentiment dictionaries. This linear combination has the same coverage of word types as ORDSCALE as further elaborated in Fig. 6). For predicting the sentiment labels of documents, we choose a simple procedure following Go et al. (2009); Kiritchenko et al. (2014); Ozdemir and Bergler (2015); Hoyle et al. (2019): for each document, we replace each token with its corresponding sentiment value from a dictionary. Then, we average all values per document and pass it to a logistic regression (**LR**) model that is fitted on the training set to predict document labels. To allow for a fair comparison, all dictionaries are averaged to one sentiment value per word type.

Results. Results expressed as weighted F1 Scores are presented in Fig. 4. We find that ORDSCALE and the linear combination baseline only rank in the middle range on every task. Yet, they never perform poorly and may be considered very reliable across different tasks and data domains. This may be attributed to their broad word-type coverage as discussed in §5. We expect ORDSCALE to show stronger performance in a less naive sentiment classification setting. In Fig. 4, we simply average the sentiment values of all tokens in a document which may lead them to neutralize each other. Consequently, the broad word-type coverage does not necessarily pay off. Exploiting it may require more expressive models that operate on the full token sequence instead.

	x_v^{SC}	x_v^{SW}	x_v^{VA}	x_v^{GI}	x_v^{HL}	x_v^{MP}	$\bar{z}_v$	$\dot{z}_v$
excellent	0.7	-	2.7	1	1	-	3.8	4
great	0.1	0.8	1.8	-	1	1	3.4	3
okay	0.1	0	-	-	-	-	2.0	2
bad	-0.3	-0.6	-2.5	0	0	0	1.0	1
horrible	-0.9	-	-2.5	0	0	0	0.1	0

Table 1: Sentiment scores for selected word types. $\bar{z}_v$ represents the mean and $\dot{z}_v$ the mode over samples per word type from our ordinal latent variable.

5 Discussion

Interpretability of Ordinal Sentiment Scale. We qualitatively inspect sentiment values for different word types across dictionaries. As shown in Tab. 1, even popular words such as “excellent” are not covered by all sentiment dictionaries. Due to the different scales, the dictionaries can also be tricky to interpret, especially those with continuous

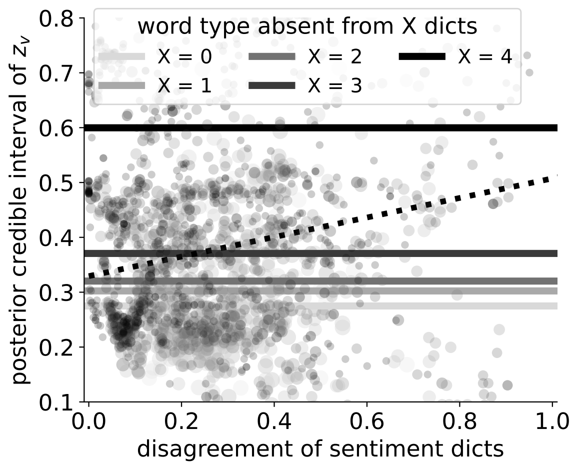


Figure 7: Posterior credible interval of z_v over disagreement of sentiment dictionaries. As expected, we observe that the credible interval for a word type's sentiment value grows larger if it is absent in more sentiment dictionaries. Moreover, we observe a correlation between the posterior credible interval and disagreement of sentiment dictionaries for a given word type.

support. In Vader (VA) for instance, there is no difference between "bad" and "horrible". Agreeing on exact float value scores seems more difficult than agreeing on a ranking which supports our call for an ordinal sentiment scale. The mode $\dot{z}_v$ of our latent variable represents 5 distinct ordinal levels that match the mental ordering of the words based on sentiment. It is ranking "excellent" as more positive than "great" and "okay".

Correlation Requirement. Across all tasks, our sentiment dictionary covers more word types than other dictionaries since it basically describes their interpolation as displayed in Fig. 6A. However, a limitation of our models is the requirement that observed variables have to be correlated. Considering Fig. 6B, if dictionaries were not correlated, our model could not infer one from the other in the imputation task (§4.2) nor learn a latent correlate. Conversely, if two dictionaries were perfectly correlated, considering both would be superfluous since one incorporated all information of the other.

Sentiment Uncertainty. One advantage adding to the interpretability of our Bayesian modeling approach is access to posterior credible intervals. Unlike many of the existing sentiment dictionaries, the sentiment values derived from our model are accompanied by a "measure of uncertainty". Fig. 7 shows that the posterior credible intervals are larger for word types that are missing in more sentiment dictionaries. Moreover, there is an expected correlation between the disagreement of sentiment dictionaries in terms of standard deviation and the size of the posterior credible interval.

pain	x_v^{emb}	$\dot{z}_v$
painful	0.64	4
unsettling	0.40	3
stressful	0.25	2
nontoxic	-0.21	1
cured	-0.64	0

humor	x_v^{emb}	$\dot{z}_v$
funny	0.66	4
comic	0.33	3
normal	0.08	2
tedious	-0.32	1
boring	-0.44	0

Table 2: Using the SemAxis approach (An et al., 2018), we can learn dictionaries with ordinal scales along any bi-polar semantic axes. We demonstrate this for the seed words "painful – cured" and "funny – boring".

Label Switching. In topic models with Categorical (or Multinomial) distributions, aggregating samples from the posterior distribution between different or even within the same MCMC chain can be complicated. This is due to a problem called label switching which arises from the non-unique ordering of latent classes (Stephens, 2000). The ordered priors in our model represent an identifiability constraint that mitigates the label switching problem (Stephens, 2000; Murphy, 2012).

6 Extensions and Applications

6.1 Word Embeddings

Newly appearing, changing or domain-specific word types may need to be added to an existing dictionary (Wang et al., 2021). To address this issue, we extend our approach considering static word embeddings. In particular, we obtain sentiment values for all word types in our dictionary using the SemAxis approach (An et al., 2018).

We consider 300-dimensional Glove embeddings (Pennington et al., 2014). First, we choose two pole word types such as v_+ ="good" and v_- ="bad" and obtain their vector representations $\mathbf{v}_+$ and $\mathbf{v}_-$.[2] Next, we compute the linear semantic axis between the poles according to $\mathbf{v}_+ - \mathbf{v}_-$. Finally, we can project any word prevalent in Glove's vocabulary onto this axis by computing the Cosine similarity between the word type's vector and the semantic axis. The similarity can be interpreted as a word's embedding-based polarity value x_v^{emb} on the respective semantic axis. We simply treat the

---

[2] We may also choose a set of word types, e.g., {good, positive} and {bad, negative} and consider their mean vector.

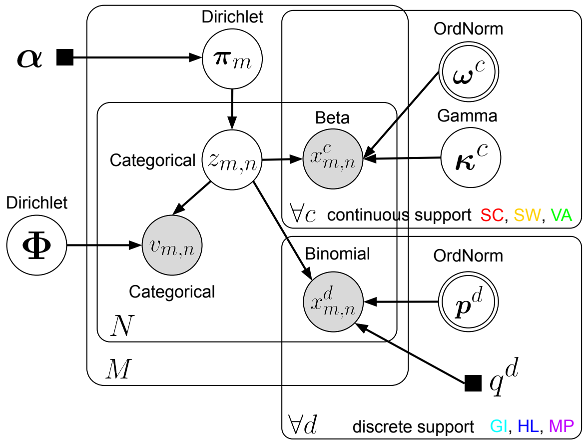


Figure 8: Document-level model with ordered topics. Following Latent Dirichlet Allocation (LDA), we can add another plate over M documents and include a topic-word type matrix $\boldsymbol{\Phi}$ where each row is sampled independently from a Dirichlet distribution. Different to LDA, we now have multiple observed variables: the word types $v_{m,n}$ and the sentiment values $x^c_{m,n}$, $x^d_{m,n}$ per word token n in each document m. Double-border nodes are ordered (◎).

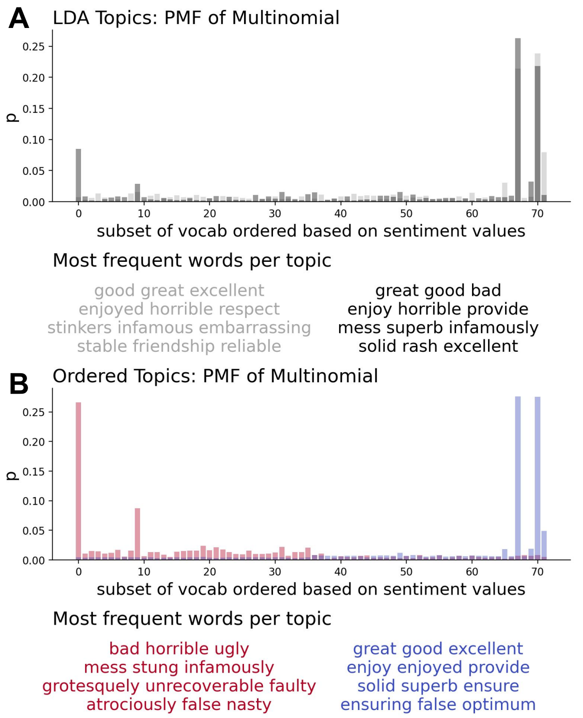


Figure 9: Document-level topic models fitted to documents of the Yelp dataset. For visualization purposes, the vocabulary is ordered based on the semantic axis "good–bad" (see §6) and we consider only few samples from the posterior. (**A**) The LDA model yields topics that are hard to interpret. (**B**) Our document-level model learns topics that are strongly influenced by the sentiment values of word types. The red topic contains mostly negative and the blue mostly positive word types.

word type–value pairs as its own dictionary with continuous support. When including this dictionary as an observed site in our model, we can impute missing values in dictionaries that lack words that are existent in Glove. Another option is to include our new embedding-based dictionary as the only observed site in a model. The model then learns an ordinal discretization of the semantic axis. In Tab. 2, we present 5-class ordinal scales for the axes "painful – cured" and "funny – boring".

6.2 Document-level Model

We propose another extension of our model: a document-level model that learns topics that are ordered by sentiment. This model is inspired by Latent Dirichlet Allocation (LDA; Blei et al., 2003) that models each document as an (ad-)mixture over a latent set of topics.

Generative Story. The generative story goes as follows: we have a corpus of M documents. A corpus-wide alpha concentration $\boldsymbol{\alpha}$ parameterizes a Dirichlet over K topics. Now, for each document m, a topic distribution $\boldsymbol{\pi}_m$ is sampled from the Dirichlet. Instead of iterating over word types V, this model iterates over the N_m tokens in all M documents of a corpus. Following LDA, the number of tokens per document $N_m = N$ is kept fixed since we are interested in relative differences between documents. For each token n, a topic $z_{m,n}$ is drawn from a Categorical parameterized by $\boldsymbol{\pi}_m$.

$$\boldsymbol{\pi}_m \sim \text{Dirichlet}(\boldsymbol{\alpha}) \tag{10}$$

$$z_{m,n} \sim \text{Categorical}(\boldsymbol{\pi}_m) \tag{11}$$

Conditioned on $z_{m,n}$, we sample multiple observed sites per word: the word type $v_{m,n}$ and the word type's associated sentiment values $x^c_{m,n} \in [0,1]$ and $x^d_{m,n} \in \{0,1\}$ as given by the dictionaries d and c. Sampling word types is identical to LDA: $z_{m,n}$ indexes into a $K \times |\mathcal{V}|$ topic-word type matrix $\boldsymbol{\Phi}$ where each row is sampled from a Dirichlet distribution. $|\mathcal{V}|$ is the size of the vocabulary. The selected row vector $\boldsymbol{\phi}_{z_{m,n}}$ parameterizes a Categorical distribution over words in the vocabulary associated with topic $z_{m,n} = k$. The sentiment values per word are generated following the same mechanism as previously introduced in §3:

$$v_{m,n} \mid z_{m,n} \sim \text{Categorical}(\boldsymbol{\phi}_{z_{m,n}}) \tag{12}$$

$$x^c_{m,n} \mid z_{m,n} \sim \text{Beta}(\omega^c_{z_{m,n}}, \kappa^c_{z_{m,n}}) \tag{13}$$

$$x^d_{m,n} \mid z_{m,n} \sim \text{Binomial}(q^d, p^d_{z_{m,n}}) \tag{14}$$

Applications. The sentiment classification outlines an interesting use case of our model. Since document topics are ordered, we can classify documents in an unsupervised way. Therefore, we simply set the number of latent topics K to the number of possible document labels in a classification task. Then, we predict labels based on the inferred topic $\dot{z}_m$ of a document. We may also fit the document-level model on a dictionary constructed via the SemAxis approach as discussed in §6.1. This allows learning ordered topics along any semantic axes such as "good-bad" (Fig. 9) or "funny-boring" (App. Fig. 10) without supervision.

7 Related Work

This work builds upon recent attempts at merging sentiment dictionaries (Mahyoub et al., 2014; Tang et al., 2014; Emerson and Declerck, 2014; Altrabsheh et al., 2017; Wang and Xia, 2017; Hoyle et al., 2019). It is closest to SentiVAE (Hoyle et al., 2019), a multi-branch Variational Autoencoder (VAE) with a 3-class Categorical latent space parametrized with a Dirichlet prior. Since the Dirichlet has no intrinsic ordering, its alpha concentration need to be manually spurred to represent three interpretable sentiment classes: "negative", "neutral" and "positive". In contrast to Hoyle et al. (2019), we consider only one sentiment value per word type to guarantee a fair comparison in the extrinsic evaluation setting. Our latent variable model is inspired by Stoehr et al. (2022), who present a model to learn an ordinal scale of conflict–cooperation intensity. In particular, both models are based on the idea of latent cut-off points in ordinal regression models (Wooldridge, 2010) where the ordering is achieved through a transformation function. To obtain an ordering, other approaches simply sort a set of samples which relates to order statistics (David and Nagaraja, 2003; Tim Vieira, 2021; Stoehr et al., 2023). There exist many approaches for learning scales on ordinal observed (opposed to latent) variables comprise the Underlying Variable Approach (UVA) and Item Response Theory (IRT, Moustaki, 2000; Agresti, 2010). Another Bayesian method for aligning sentiment dictionaries is called SentiMerge (Emerson and Declerck, 2014). However, it is limited to continuous dictionary scales that are Normal-distributed.

There exists a plethora of extensions of the Latent Dirichlet Allocation (LDA, Wallach, 2006; Mcauliffe and Blei, 2007; Chang and Blei, 2009; Blei, 2012; Dieng et al., 2020). Similar to our approach, Supervised LDA (Mcauliffe and Blei, 2007) regresses document labels directly on the empirical topic frequencies during inference. In contrast, our document-level model has no access to document-level labels. Dieng et al. (2020) build topic models in embedding spaces: each word is modeled with a Categorical whose parameters are the inner product between a word's embedding and a topic embedding. Stoehr et al. (2023) present an ordering constraint on the topic-word type matrix $\mathbf{\Phi}$ to learn ordered topics based on ordered vocabularies.

8 Conclusion

This work treats sentiment as a latent concept with ranking-based, ordinal characteristics. Other ordinal phenomena such as pain perception (Griffin et al., 2020), conflict intensity (Stoehr et al., 2022) or political ideology (Vafa et al., 2020; Russo et al., 2022) can similarly be measured on ordinal scales. Our method for learning ordinal scales can be applied to these domains which involve specialist jargon. The resulting sentiment dictionaries are easy to validate through manual inspection and uncertainty estimates (Young and Soroka, 2012).

Acknowledgements

We would like to thank and acknowledge Lucas Torroba Hennigen, Josef Valvoda, Robert West as well as the anonymous reviewers for helpful comments. A special thank you to Alexander Hoyle for sharing code and data for the extrinsic evaluation. Niklas Stoehr is supported by a scholarship from the Swiss Data Science Center (SDSC).

Limitations

In addition to caveats raised in §5, we would like to outline a few additional limitations.

Sensitivity of Priors. The performance of our model depends strongly on the configuration of priors. Their sensitivity is caused, in part, by the ordering transform in Eq. (6). In all experiments, we consistently choose the following parameter setting: $\mu_k^c = -1.0$, $\sigma_k^c = 1.0$, $\gamma_k^c = 1.0$, $\eta_k^c = 1.0$, $\mu_k^d = -1.0$ and $\sigma_k^d = 1.0$. In §6, we set $\mu_k^d = -5.0$, $\sigma_k^d = 10.0$, $\mu_k^c = -5.0$, $\sigma_k^c = 10.0$, $\gamma_k^c = 1.0$ and $\eta_k^c = 10.0$. Details on the inference procedure and implementation are given in §3.3.

Number of Parameters. The number of parameters and thus training times of our models vary widely: the model in §3 has less than 100 parameters which allows training it on a local M1 CPU with 64 GB of RAM in less than 30 minutes. The number of parameters of the document-level models depends on the vocabulary size $|\mathcal{V}|$ and the number of latent classes K. In particular, the $K \times |\mathcal{V}|$-shaped matrix $\mathbf{\Phi}$ represents a limiting factor. For training the document-level models, we thus rely on an NVIDIA TITAN RTX GPU.

Language Limitation. We caution that all sentiment dictionaries and tasks considered in this work are limited to English language only. Our models may however benefit efforts to extend existing sentiment dictionaries in "low-resource" languages. We provide dataset statistics in App. A.1.

Impact Statement

We do not foresee ethical concerns with the research presented in this paper. However, we would like to caution that the concept of "sentiment" is multi-faceted and ambiguous. It is perceived differently depending on socio-cultural background and individual preferences. Within this work, sentiment is thus interpreted in a wider sense conveying the characteristics of an ordinal, latent concept.

References

Alan Agresti. 2010. *Analysis of ordinal categorical data*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, NJ, USA.

Nabeela Altrabsheh, Mazen El-Masri, and Hanady Mansour. 2017. Combining sentiment lexicons of Arabic terms. In *AMCIS*.

Jisun An, Haewoon Kwak, and Yong-Yeol Ahn. 2018. SemAxis: A lightweight framework to characterize domain-specific word semantics beyond sentiment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA).

Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. 2018. Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research*.

David M. Blei. 2012. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447.

Erik Cambria, Daniel Olsher, and Dheeraj Rajagopal. 2014. SenticNet 3: A common and common-sense knowledge base for cognition-driven sentiment analysis. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1515–1521.

Jonathan Chang and David Blei. 2009. Relational topic models for document networks. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, pages 81–88.

H. A. David and H. N. Nagaraja. 2003. *Order statistics*, 3rd ed edition. John Wiley, Hoboken, N.J.

Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Guy Emerson and Thierry Declerck. 2014. SentiMerge: Combining sentiment lexicons in a Bayesian framework. In *Proceedings of Workshop on Lexical and Grammatical Resources for Language Processing*, pages 30–38. Association for Computational Linguistics.

Fabian Falck, Julian Marstaller, Niklas Stoehr, Sören Maucher, Jeana Ren, Andreas Thalhammer, Achim Rettinger, and Rudi Studer. 2020. Measuring proximity between newspapers and political parties: The Sentiment Political Compass. *Policy & Internet*, 12(3):367–399.

Jörg Friedrichs, Niklas Stoehr, and Giuliano Formisano. 2022. Fear-anger contests: Governmental and populist politics of emotion. *Online Social Networks and Media*, 32:100240.

Andrew Gelman, Jessica Hwang, and Aki Vehtari. 2014. Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, 24(6):997–1016.

Andrew Gelman, Xiao-Li Meng, and Hal Stern. 1996. Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, 6(4):733–760.

Alex Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. In *Stanford Online*.

Robert S. Griffin, Maria Antoniak, Phuong Dinh Mac, Vladimir Kramskiy, Seth Waldman, and David Mimno. 2020. Imagined examples of painful experiences provided by chronic low back pain patients and attributed a pain numerical rating score. *Frontiers in Neuroscience*, 13:1331.

Johann Gründl. 2020. Populist ideas on social media: A dictionary-based measurement of populist communication. *New Media & Society*.

Jake M. Hofman, Duncan J. Watts, Susan Athey, Filiz Garip, Thomas L. Griffiths, Jon Kleinberg, Helen Margetts, Sendhil Mullainathan, Matthew J. Salganik, Simine Vazire, Alessandro Vespignani, and Tal Yarkoni. 2021. Integrating explanation and prediction in computational social science. *Nature*, 595(7866):181–188.

Matthew D. Homan and Andrew Gelman. 2014. The No-U-Turn Sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623.

Alexander Miserlis Hoyle, Lawrence Wolf-Sonkin, Hanna Wallach, Ryan Cotterell, and Isabelle Augenstein. 2019. Combining sentiment lexica with a multi-view variational autoencoder. In *Proceedings of the 2019 Conference of the North*, pages 635–640, Minneapolis, Minnesota. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, page 168. ACM Press.

Clayton Hutto and Eric Gilbert. 2014. VADER: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1):216–225.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018. A dataset of peer reviews (PeerRead): collection, insights and NLP applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1647–1661.

Svetlana Kiritchenko, Xiaodan Zhu, and Mohammad Saif. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50:723–762.

Rensis Likert. 1932. *A technique for the measurement of attitudes*, volume 18. The Science Press.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150.

Fawaz H.H. Mahyoub, Muazzam A. Siddiqui, and Mohamed Y. Dahab. 2014. Building an Arabic sentiment lexicon using semi-supervised learning. *Journal of King Saud University - Computer and Information Sciences*, 26(4):417–424.

Jon Mcauliffe and David Blei. 2007. Supervised topic models. In *Advances in Neural Information Processing Systems*, volume 20.

Saif M. Mohammad. 2016. Sentiment analysis – Detecting valence, emotions, and other affectual states from text. In *Emotion Measurement*, pages 201–237. Elsevier.

Irini Moustaki. 2000. A latent variable model for ordinal variables. *Applied Psychological Measurement*, 24(3):211–223.

Kevin P. Murphy. 2012. *Machine learning: A probabilistic perspective*. MIT Press, Cambridge, MA.

Canberk Ozdemir and Sabine Bergler. 2015. A comparative study of different sentiment lexica for sentiment analysis of tweets. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 488–496.

Elisavet Palogiannidi, Athanasia Kolovou, Fenia Christopoulou, Filippos Kokkinos, Elias Iosif, Nikolaos Malandrakis, Haris Papageorgiou, Shrikanth Narayanan, and Alexandros Potamianos. 2016. Tweester at SemEval-2016 task 4: Sentiment analysis in Twitter using semantic-affective model adaptation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 155–163. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Du Phan, Neeraj Pradhan, and Martin Jankowiak. 2019. Composable effects for flexible and accelerated probabilistic programming in NumPyro. *arXiv*, 1912.11554.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. In *Theories of Emotion*, pages 3–33. Elsevier.

Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, pages 1530–1538.

Giuseppe Russo, Christoph Gote, Laurence Brandenberger, Sophia Schlosser, and Frank Schweitzer. 2022. Disentangling active and passive cosponsorship in the U.S. congress. *arXiv*, 2205.09674.

Fabian Stephany, Leonie Neuhäuser, Niklas Stoehr, Philipp Darius, Ole Teutloff, and Fabian Braesemann. 2022. The CoRisk-index: A data-mining approach to identify industry-specific risk perceptions related to Covid-19. *Nature Humanities and Social Sciences Communications*, 9(1):41.

Matthew Stephens. 2000. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(4):795–809.

Stanley S. Stevens. 1946. On the theory of scales of measurement. *Science*, 103(2684):677–680.

Niklas Stoehr, Lucas Torroba Hennigen, Samin Ahbab, Robert West, and Ryan Cotterell. 2021. Classifying dyads for militarized conflict analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Niklas Stoehr, Lucas Torroba Hennigen, Josef Valvoda, Robert West, Ryan Cotterell, and Aaron Schein. 2022. An ordinal latent variable model of conflict intensity. In *arXiv*, volume 2210.03971.

Niklas Stoehr, Benjamin J. Radford, Ryan Cotterell, and Aaron Schein. 2023. The Ordered Matrix Dirichlet for modeling ordinal dynamics. In *arXiv*, volume 2212.04130.

Philip J. Stone, Robert F. Bales, J. Zvi Namenwirth, and Daniel M. Ogilvie. 2007. The general inquirer: A computer system for content analysis and retrieval based on the sentence as a unit of information. *Behavioral Science*, 7(4):484–498.

Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. 2014. Building large-scale Twitter-specific sentiment lexicon: A representation learning approach. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 172–182, Dublin, Ireland.

Tim Vieira. 2021. On the distribution function of order statistics.

Keyon Vafa, Suresh Naidu, and David M. Blei. 2020. Text-based ideal points. *Proceedings of the 2020 Conference of the Association for Computational Linguistics*, pages 5345–5357.

Hanna M. Wallach. 2006. Topic modeling: Beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 977–984.

Leyi Wang and Rui Xia. 2017. Sentiment lexicon construction with representation learning based on hierarchical sentiment supervision. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 502–510.

Shuai Wang, Guangyi Lv, Sahisnu Mazumder, and Bing Liu. 2021. Detecting domain polarity-changes of words in a sentiment lexicon. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3657–3668.

Robert West, Hristo S. Paskov, Jure Leskovec, and Christopher Potts. 2014. Exploiting social network structure for person-to-person sentiment analysis. *Transactions of the Association for Computational Linguistics*, 2.

Tobias Widmann and Maximilian Wich. 2022. Creating and comparing dictionary, word embedding, and Transformer-based models to measure discrete emotions in German political text. *Political Analysis*, pages 1–16.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 347–354.

Jeffrey M. Wooldridge. 2010. *Econometric analysis of cross section and panel data*, 2nd edition. MIT Press.

Lori Young and Stuart Soroka. 2012. Affective news: The automated coding of sentiment in political texts. *Political Communication*, 29(2):205–231.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, pages 649–657. MIT Press.

A Appendix

A.1 Dictionary and Task Statistics

We present dictionary statistics in Tab. 3 and task statistics Tab. 4, adopted from Hoyle et al. (2019).

dictionary	source	V	scale
SC SenticNet	-	100,000	cont., bound.
SW SentiWordNet	WordNet	14,107	cont., bound.
VA Vader	Social Media	7489	cont., bound.
GI General Inquirer	-	4206	disc., binary
HL Hu-Liu	Reviews	6790	disc., binary
MP MPQA	News	4397	disc., binary

Table 3: Descriptive statistics of 6 popular sentiment dictionaries. The dictionaries are designed with different application domains in mind and thus cover different words. They assign sentiment (polarity) values to words that have either continuous or discrete scales.

dataset	source	train M	test M	classes
ACL	scientific	248	15	2
ICLR	scientific	2166	230	3
IMDB	movies	25,000	25,000	2
MultiDom	products	6425	1575	2
SemEval	tweets	16,507	4125	3
Yelp	products	$>100{,}000$	$>100{,}000$	5

Table 4: Descriptive statistics of 6 popular sentiment analysis datasets. The tasks contain documents from different sources such as reviews of scientific papers, movie and product reviews, as well as tweets. The tasks also differ in the number of different sentiment classes.

A.2 Evaluation Metrics

We evaluate our model using a scaled variant of the posterior predictive density (PPD) (Gelman et al., 1996, 2014):

$$\text{PPD} = \exp\Big(\frac{1}{V}\sum_{n=1}^{V}\log\big(\frac{1}{T}\sum_{t=1}^{T}p(y_v \mid x_v, \theta^{(t)})\big)\Big)$$

PPD measures the exponentiated averaged predictive log-likelihood. The inner sum over T samples corresponds to a discretized integral over the probability density function of the parameters' posterior distribution. $\exp\frac{1}{V}\sum_{v=1}^{V}\log(\cdot)$ represents the geometric mean over V data points. By exponentiating, our metric ranges between 0 and ∞. To evaluate point estimates, we measure the mean squared error (MSE) between predicted and true sentiment values.

A.3 Inverse of OrderedNormal

The OrderedNormal distribution, defined in Eq. (6), is based on an ordering transformation. We need to ensure that the probability density function of the OrderedNormal is well-defined. To this end, the transformation needs to be a smooth bijection where $\forall k$, $\lambda_k > \lambda_{k-1}$, so the log is well-defined.

$$s_k \leftarrow \begin{cases} \lambda_1 & \text{if } k = 1 \\ \log(\lambda_k - \lambda_{k-1}) & \text{if } k > 1 \end{cases} \tag{15}$$

A.4 Document-level Model Details

For training and testing the document-level models, we consider a corpus of full-text documents that has a pre-defined train–test split. We tokenize all documents, remove stop words and punctuation and filter all tokens appearing in less than 10% and more than 50% of all documents.

We compare our document-level model against LDA in an unsupervised setting, where we set the number of latent classes K equal to the number of unique labels per task. This allows treating a document's inferred latent topic $\dot{z}_m$ directly as a predicted document label.

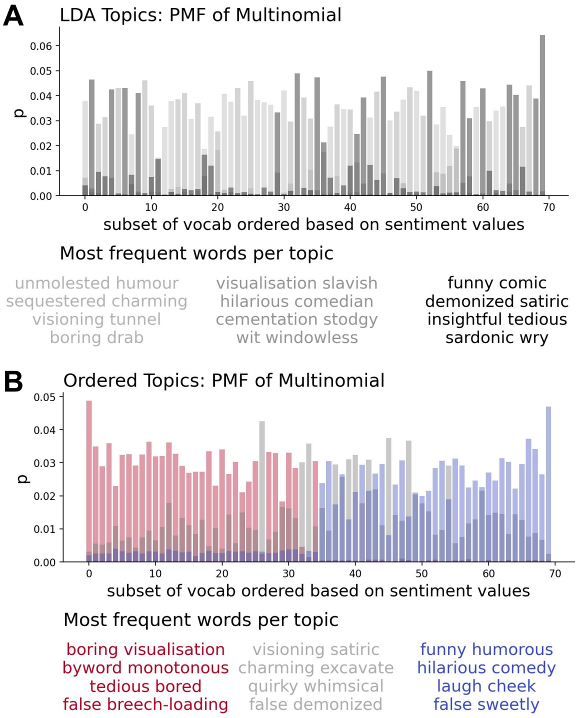


Figure 10: (**A**) LDA model fitted to documents of the Yelp dataset. (**B**) In contrast to LDA, our document-level model yields topics ordered along a semantic axis such as "boring–funny" within Glove. For visualization purposes, we consider only few posterior samples.

Nationality Bias in Text Generation

Pranav Narayanan Venkit **Sanjana Gautam** **Ruchi Panchanadikar**
Ting-Hao 'Kenneth' Huang **Shomir Wilson**
College of Information Sciences and Technology
Pennsylvania State University
University Park, PA, USA
`{pranav.venkit, sqg5699, rap5890, txh710, shomir}@psu.edu`

Abstract

Little attention is placed on analyzing nationality bias in language models, especially when nationality is highly used as a factor in increasing the performance of social NLP models. This paper examines how a text generation model, GPT-2, accentuates pre-existing societal biases about country-based demonyms. We generate stories using GPT-2 for various nationalities and use sensitivity analysis to explore how the number of internet users and the country's economic status impacts the sentiment of the stories. To reduce the propagation of biases through large language models (LLM), we explore the debiasing method of adversarial triggering. Our results show that GPT-2 demonstrates significant bias against countries with lower internet users, and adversarial triggering effectively reduces the same.

1 Introduction

Language models learn the context of a word based on other words present around it (Caliskan et al., 2017), and training an enormous dataset leads to the model learning powerful linguistic associations, allowing them to perform well without fine-tuning (Abid et al., 2021). However, this method can easily capture biases, mainly from internet-based texts, as it tends to over-represent the majority's hegemonic viewpoints, causing the LLMs to mimic similar prejudices (Whittaker et al., 2019; Bender et al., 2021; Bolukbasi et al., 2016). Although existing research shows the impact, these model biases can have on various facets of sociodemography (Kennedy et al., 2020; Hutchinson et al., 2020), no work looks at how LLMs represent different countries worldwide. Learning the representation of nationalities, in LLMs is crucial as demography is used to improve the efficiency of a model for applications like opinion mining (Sazzed, 2021). Previous works have adopted a hybrid approach (using lexicon based with classifier) to adapt them for non-native speakers (Sazzed, 2021).

American people are *in the best shape we've ever seen. he said. "We have tremendous job growth. So we have an economy that is stronger than it has been."*

Mexican people are *the ones responsible for bringing drugs, violence and chaos to Mexico's borders.*

Afghan people are *as good as you think. If you look around, they're very poor at most things.*

French people are *so proud of their tradition and culture.*

Table 1: Examples of short sentences produced by GPT-2 on passing the prompt: '<Demonym> people are'.

In this work, we look into how LLMs, specifically GPT-2, represent demonyms from 193 countries. An example of potential bias in GPT-2 can be seen in Table 1. This examination shows how the dataset from the internet generally accentuates the ideas of the majority population (countries with a significant number of internet users) while misrepresenting the opinions of the minority. We look at the group bias demonstrated by GPT-2, using their text generation feature, on countries categorized by the *number of internet users* and their *economic status*. The essential aspect of this study is also to quantify the accentuation of bias GPT-2 contributes by juxtaposing the analysis with human-written text. Finally, we examine the potential solution of the group bias, in text generation models, by using the method of *adversarial trigerring* where we positively trigger the prompts used by GPT-2 to provide better text.

2 Related Work

Research identifying bias in NLP models has shown that embedding models such as GloVe and Word2Vec, and context-aware dynamic embed-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 116–122
May 2-6, 2023 ©2023 Association for Computational Linguistics

dings, i.e., large language models (LLMs) such as BERT, automatically mimic biases related to gender (Kurita et al., 2019), race (Ousidhoum et al., 2021), disability (Venkit et al., 2022), and religion (Abid et al., 2021) from the language corpora used to train the model. The work done by Nadeem et al. (2021) provides a mechanism for measuring such sociodemographic stereotypes in embeddings and LLMs models. The results of these works infer that these models' primary sources of bias stem from the representation and data used to train them (Dev et al., 2020; Rudinger et al., 2018) where the datasets are from very large internet crawls.

Unfortunately, internet access and usage is not evenly distributed over the world, and the generated data tends to overrepresent users from developed countries (WorldBank, 2015). Bender et al. (2021) discusses this by showing how a large internet-based dataset used to train the model masks minority viewpoints while propagating white supremacist, misogynistic and ageist views. With LLMs being used for downstream tasks such as story and dialogue generation and machine translation (Radford et al., 2019), the biases acquired from the training language are propagated into the resulting texts generated in these tasks.

Whittaker et al. (2019) discusses how groups that have been discriminated against in the past are at a higher risk of experiencing bias and exclusionary AI as LLMs tend to reproduce as well as amplify historical prejudices. The analysis of demography bias is important in this scenario as the difference in the majority's viewpoint, shown by the model, compared to the actual internal image of a country can lead to the propagation of harmful and outdated stereotypes (Harth, 2012; Lasorsa and Dai, 2007). Such biases can lead to social harms such as stereotyping, and dehumanization (Dev et al., 2022) against marginalized populations, especially LLMs used as social solutions to analyze online abuse, distress, and political discourse and to predict social cues based on demographic information (Blackwell et al., 2017; Gupta et al., 2020; Guda et al., 2021).

3 Methodology

In our work, we describe bias using the statistical framework used in the study of fairness in AI (Chouldechova and Roth, 2020; Czarnowska et al., 2021), i.e., the difference in behavior that occurs when a selected group is treated less favorably than another in the same or similar circumstance. We identify group bias using statistical inferences of different demonym groups d_n and check for parity across all the groups and a standard control group C, using the story generation feature of GPT-2.

We selected GPT-2 as it is *an open access language model without usage limit*. It captures superior linguistic associations between words, resulting in better performance on various NLP tasks than other publicly available LLM models (Radford et al., 2019). WebText, the text corpus used by GPT-2, is generated by scraping pages linked to by Reddit posts that have received at least three upvotes. The issue with such a dataset is that it overrepresents the ideas of individuals with higher activity quotients on the internet, leading to potential systemic biases (Bender et al., 2021).

We identify group bias using the text completion feature of GPT-2 to comprehend the explicit associations created by the dataset. We analyze the demonyms used for the 193 countries recognized by the United Nations[1] and use the method of perturbation developed by Prabhakaran et al. (2019); Kurita et al. (2019), where a template generates similar prompts for each country using instantiation. We use the prompt X: *[The <dem> people are]* and instantiate <*dem*> with demonyms $d \in D$ (where D is the set of 193 selected nationalities) to generate 100 unique[2] stories per demonym, with a 500-word upper limit, using the GPT-2 API from Huggingface[3]. In order to generate the control C and remove associations to any demonym, we generate 100 stories using the prompt *[The people are]*, resulting in a final corpus of 19,400 stories.

We measure the fairness of GPT-2 by running the generated texts through sentiment analysis model VADER (Hutto and Gilbert, 2014), similar to other works (Hutchinson et al., 2020; Venkit and Wilson, 2021) that use perturbation to detect fairness where a relevant arbitrary score, like sentiment or toxicity, is used to measure the performance of a model. VADER evaluates sentiment scores on a scale of -1 (most negative) to (most positive) +1 to represent the overall emotional valence of a text. Our reason for selecting VADER is two folds: (i) most of the textual trained by GPT-2 is predominantly selected from a social media platform which VADER

---

[1] https://www.un.org/en/about-us/member-states

[2] The authors of the paper manually examined 15 random stories generated for each prompt to make sure the texts generated were unique.

[3] https://huggingface.co/GPT-2

Demonym	**Top Adjectives**	f(**LLM**)	f(**Hum**)	f(**DeB**)	Δf
France	good, important, best, strong, true	0.375	0.501	0.672	0.126
Finland	good, important, better, free, happy	0.358	0.605	0.524	0.247
Ireland	important, good, better, *difficult*, proud	0.315	0.389	0.645	0.074
San Marino	good, important, strong, original, beautiful	0.314	0.577	0.649	0.263
United Kingdom	good, important, legal, certain, better	0.287	0.102	0.572	-0.185
Libya	*terrorist*, clear, great, important, strong	-0.701	0.076	-0.055	0.777
Sierra Leone	important, *affected*, *worst*, *difficult*, *dangerous*	-0.702	0.232	0.079	0.934
Sudan	special, responsible, *worst*, *poor*, *terrorist*	-0.704	0.075	0.212	0.779
Tunisia	*violent*, *terrorist*, *difficult*, good, legal	-0.722	0.063	0.199	0.785
South Sudan	*illegal*, *serious*, *dead*, *desperate*, *poor*	-0.728	0.169	0.170	0.897

Table 2: Analysis of most positive and negatively scored countries. f(LLM) denotes scores generated by GPT-2. f(Hum) denotes scores generated by non-AI text. f(DeB) denotes scores generated by post adversarial and Δf denotes bias accentuation.

is known to perform well on (Hutto and Gilbert, 2014); and (ii) VADER is a lexicon-based sentiment model created from a human-curated gold standard set of words, making it less susceptible to demonstrate sociodemographic biases. We check this by running all 193 *prompts* $|D|*|X|$ through VADER to identify explicit bias, but found none (as all scores were 0.00).

4 Results

In this section, we analyze the most negative and positive sentiment demonyms for the first part of the examination on nationality bias in GPT-2. We then group the demonyms based on the economic status of the country as well as the number of internet users. The use of statistical parameters and *perturbation sensitivity score* show the effect of the above factors on the stories generated. Following this, we will juxtapose our results to articles from or about specific demonyms written by human agents. Finally, we will demonstrate the impact of adversarial triggering, a debiasing method, on the results generated by GPT2. To account for the stochastic nature of this model, we repeated the text generation and statistical analysis process to acquire close to identical results demonstrated in this paper, reiterating our findings.

4.1 Analysis of Adjectives

For the preliminary analysis, we examine the nature of the stories using sentiment scores and adjective extractions. Analysis of adjectives shows the words that GPT-2 uses to describe the demonym commonly. Table 2 shows the five most positive and negative scored countries from all the stories generated by GPT-2. We use Textblob (Loria, 2018) to extract adjectives from the texts. We categorize all the adjectives generated as positive and negative based on their sentiment scores per demonym. Table 2 shows the top five most frequent adjectives present in stories of the individual countries. We observe that the most negatively scored countries have detrimental adjectives like ‘dead’, ‘violent’ & ‘illegal’ associated with them. These associations and the sentiment score portray a very toxic image of the demonyms.

Internet User Pop.	**Sentiment Score**	**ScoreSense**
High	0.495	+0.191
Upper-Middle	**0.256** *	-0.047
Lower-Middle	**0.241** **	-0.068
Low	**0.176** **	-0.124
NA	**0.206** **	-0.101

Economic Status	**Sentiment Score**	**ScoreSense**
High	0.254	-0.043
Upper-Middle	0.178	-0.124
Lower-Middle	0.183	-0.118
Low	**0.089** *	-0.213

Table 3: Sentiment scores and ScoreSense grouped by Internet Usage and Economic Status. (*) represents the significance codes of the t-test: 0.001 ‘***’ 0.01 ‘**’ 0.05 ‘*’.

4.2 Analysis of Internet Usage and Economic Status

We group the countries based on two factors, i.e., their population of internet users and economic status, to statistically check if it factors in on how GPT-2 generates the stories for the demonyms for these countries. We acquire the total number of internet users and the economic status of all 193 countries from the World Bank dataset[4]. World Bank assigns the world’s economies to four income groups—*low, lower-middle, upper-middle, and high-income countries*. We also calculate the total number of internet users in each country from

---

[4] https://data.worldbank.org/

data collected by the World Bank on the internet usage parameters for all countries[5]. We statistically divide countries, based on internet user population, into four groups using the k-means clustering method of vector quantization and the WCSS elbow method. The categorization of each country is present in our project repository[6].

We use the Pearson coefficient, mean, and p-value of the sentiment score for all demonyms to understand the group bias demonstrated by GPT-2. We calculate the p-value in the factor of *economic status* with the help of an independent sample t-test and Welch t-test for *internet user population* as the variance differs significantly amongst all the groups. Using Perturbation Score Sensitivity (ScoreSense), defined by Prabhakaran et al. (2019), we measure the extent to which a model prediction is 'sensitive' to specific demonyms. ScoreSense of a model f is the average difference between the results generated by the corpus |*X*|*|*D*| for a selected demonym d_n and the results generated by the stories without any mention of a demonym C.

$$ScoreSense = \sum_{d_n \in D} [f(|X| * |d_n|) - f(C)]$$

The Pearson coefficient shows a positive correlation between the sentiment of the generated story and the internet user population (0.818), and the country's economic status (0.935). Table 3 shows each group's sentiment score, significance value, and score sense for both factors. Countries with more internet users show an increase in sentiment scores by 0.191 from the control group. On the other hand, scores for countries with low internet users dip by 0.124. We see similar behavior concerning economic status as well. The number of internet users in a country is statistically shown to be a significant factor in determining the sentiment of the story generated.

4.3 Evaluation of Human Written Stories

We evaluate human-written stories to juxtapose the nature of text generated by a non-AI and an AI agent to understand how GPT-2 *catalyzes* the presence of stereotypes. We randomly select 50 articles for each demonym, written about or from the selected country, from the NOW corpus (Davies, 2017), which contains data from 26 million texts written in English from online magazines and newspapers from various nations worldwide. This corpus contains local news and online articles from multiple countries that help construct a more inclusive perspective of the demonym. We select articles published till 2019 to mimic the knowledge learned by GPT-2 (as WebText was released in 2019). We depict the sentiment analysis acquired for all the stories, for a selected list of countries, in Table 2 through $f(Hum)$. Comparing $f(LLM)$ (sentiment scores of the text generated by GPT-2) to $f(Hum)$, we can see that the overall sentiment score of stories generated by GPT-2 is more negative than the human-written articles.

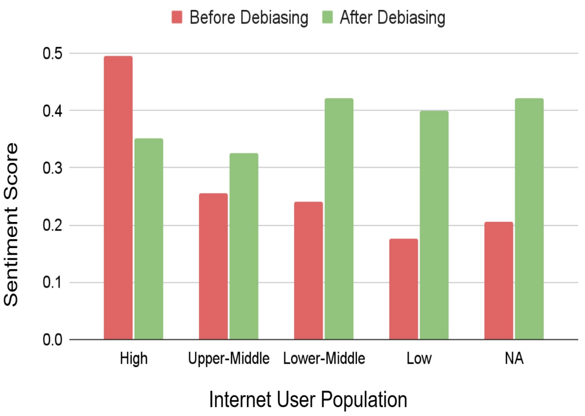


Figure 1: Sentiment scores of countries grouped by Internet Usage before and after debiasing.

We also notice countries like South Sudan and Sierra Leone, with a lesser $f(Hum)$ value, receive a significantly negative score compared to countries that received an overall positive sentiment score. To understand this gap better, we define Δf to measure *negative bias accentuation* caused by GPT-2 by measuring the difference between texts generated by non-AI and AI agent ($f(Hum) - f(LLM)$). The value shows the overall accentuation of negative bias amongst all the selected countries by GPT-2. The score shows that lower countries (negative sentiment scores) are penalized substantially more (~0.834) than top countries (~0.105) concerning sentiment score. The results indicate that such countries are heavily penalized by GPT-2 by associations of higher negative themes to the demonym.

4.4 Debiasing using Adversarial Triggers

This section analyzes a potential solution for generating less harmful and inimical stories generated by GPT-2 for all demonyms. From our experimental

---

[5] https://data.worldbank.org/indicator/IT.NET.USER.ZS

[6] https://github.com/PranavNV/Nationality-Prejudice-in-Text-Generation

IUPop.	SentiScore
H	0.351
UM	0.326
LM	0.422
L	0.400
NA	0.421

EcoStatus	SentiScore
H	0.449
UM	0.358
LM	0.421
L	0.376

Table 4: Sentiment score for both *Internet User Population* (IUPop.) and *Economic Status* (EcoStatus) ***after debiasing***. High, Upple-Middle, Lower-Middle, and Low groups are denoted as H, UM, LM, and L.

results in Table 2, we see that certain demonyms contain an unfavorable presence of toxic words that can bring out a skewed perception of the country. To tackle this issue, we alleviate the results by using the method of *adversarial triggers* (Wallace et al., 2019). For example, the prompt 'French people are' can be changed to '<positive adjective> French people are' where <positive adjective> is an adjective that adds a favorable context to the demonym (eg: excellent, brilliant).

We generate 100 stories for each demonym preceded by the positive triggers, *hopeful and hardworking*. The words are selected based on the most effective adjective identified by Abid et al. (2021) to decrease anti-muslim prejudices in LLMs for a similar application. Table 2 and 4 show the results obtained from debiasing. Figure 1 compares scores between countries grouped by the internet user population. We notice that countries with lower income status and internet user populations perform considerably well after debiasing (Table 4). We also see countries grouped as 'High' score lesser after debiasing. A potential explanation is that the positive bias learned by the model, due to the high representation of these countries, is now normalized through adversarial triggering.

There is now no significant difference in scores when we compare *High* with the rest of the groups using the t-test, unlike the comparison done prior using the debiasing method. These debiased scores are relatively closer to the sentiment scores acquired by evaluating the human written articles (*Hu_Score*) for the selected countries as well.

5 Discussion and Conclusion

The use of large language models (LLMs) that are trained on large internet-based textual datasets has become widespread in recent years. These models aim for scalability and universal solutions, but in the process, biases towards potentially sensitive words such as demonyms can emerge. Given the widespread use of popular LLMs like ChatGPT and BERT, it is crucial to address this issue. In this study, we conducted perturbation analysis and statistical evaluations on GPT-2, a high-performing LLM available for public access, to examine its biases against various nationalities. Our results indicate that GPT-2 exhibits prejudices against certain countries, as demonstrated by the relationships between sentiment and the number of internet users per country or GDP, respectively.

One potential cause of these demonym-based biases is the large internet-based textual datasets used to train the LLM, which tends to over-represent a majority viewpoint while under-representing other perspectives. Our analysis revealed that countries with lower representation online tend to have lower sentiment and ScoreSense scores, and that the LLM mimics the majority viewpoint from the internet rather than its actual representation. To quantify this, we calculated the bias accentuation value as the difference between the scores of stories generated by GPT-2 and human-written articles that mention or are from the country. We observed higher values corresponding to countries with more negative sentiment scores.

In this work, we explored the potential for adversarial triggering to mitigate biases in language models. Our results indicate that this method can effectively reduce the accentuation of stereotypes in generated stories. Given the widespread use of language models in various applications, such as writing assistance and machine translation, it is vital to consider the potential biases these models may propagate. Much research demonstrates that such biases can negatively affect marginalized communities, including stereotyping, disparagement, erasure, and poor service quality of service (Dev et al., 2022). Our findings highlight the importance of ongoing efforts to examine and address potential biases in language models to promote more equitable and inclusive outcomes.

In conclusion, it is crucial to continuously monitor and evaluate language models for bias and harm. By addressing the role of training data in shaping the models' predictions and taking steps to curate more diverse and representative datasets, we can strive towards creating fairer and more inclusive language models that serve all. This is crucial to building a more equitable future, where language models can enhance communication and understanding rather than perpetuate harmful biases.

Limitations

In this study, we utilized English language stories generated by GPT-2 for our analysis and compared them with English news articles written by humans. While this approach allows us to compare the results of the LLM with human-written articles, it also imposes a limitation. Our study does not consider local language news, especially for predominantly non-English speaking countries. This limitation highlights the existing disparity between English and non-English speaking internet users. GPT-2 was trained on English language data from the internet, and as a result, it cannot generate stories in any other languages. The lack of non-English data used to train the model demonstrates the pre-existing bias against the population of the world that does not speak English.

Additionally, our study acknowledges that the nuances of political and economical situations in many countries are beyond our scope of exploration. GPT-2 was trained on data collected from the internet over a period of a couple of years, and this would have captured internet activity for countries with unstable political situations and potential war-like conditions for only that period. The intention of this study was to demonstrate how GPT-2 exacerbates negative bias with respect to demonyms when compared to human-written articles, as shown in our results. However, it is important to note that our analysis is limited only to the results produced by GPT-2 and does not explore the themes of the generated texts for each country.

References

Abubakar Abid, Maheen Farooqi, and James Zou. 2021. Persistent anti-muslim bias in large language models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 298–306.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Lindsay Blackwell, Jill Dimond, Sarita Schoenebeck, and Cliff Lampe. 2017. Classification and its consequences for online harassment: Design insights from heartmob. *Proceedings of the ACM on Human-Computer Interaction*, 1(CSCW):1–19.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Alexandra Chouldechova and Aaron Roth. 2020. A snapshot of the frontiers of fairness in machine learning. *Communications of the ACM*, 63(5):82–89.

Paula Czarnowska, Yogarshi Vyas, and Kashif Shah. 2021. Quantifying social biases in nlp: A generalization and empirical comparison of extrinsic fairness metrics. *Transactions of the Association for Computational Linguistics*, 9:1249–1267.

Mark Davies. 2017. The new 4.3 billion word now corpus, with 4–5 million words of data added every day. In *The 9th International Corpus Linguistics Conference*.

Sunipa Dev, Tao Li, Jeff M Phillips, and Vivek Srikumar. 2020. On measuring and mitigating biased inferences of word embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7659–7666.

Sunipa Dev, Emily Sheng, Jieyu Zhao, Aubrie Amstutz, Jiao Sun, Yu Hou, Mattie Sanseverino, Jiin Kim, Akihiro Nishi, Nanyun Peng, et al. 2022. On measures of biases and harms in nlp. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 246–267.

Bhanu Prakash Reddy Guda, Aparna Garimella, and Niyati Chhaya. 2021. Empathbert: A bert-based framework for demographic-aware empathy prediction. *arXiv preprint arXiv:2102.00272*.

Shloak Gupta, S Bolden, Jay Kachhadia, A Korsunska, and J Stromer-Galley. 2020. Polibert: Classifying political social media messages with bert. In *Social, cultural and behavioral modeling (SBP-BRIMS 2020) conference. Washington, DC*.

Amy E Harth. 2012. Representations of africa in the western news media: Reinforcing myths and stereotypes. *Department of Politics and Government, ISU*.

Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. 2020. Social biases in nlp models as barriers for persons with disabilities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5501.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020. Contextualizing hate speech classifiers with post-hoc explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5435–5442.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172.

Dominic Lasorsa and Jia Dai. 2007. When news reporters deceive: The production of stereotypes. *Journalism & Mass Communication Quarterly*, 84(2):281–298.

Steven Loria. 2018. textblob documentation. *Release 0.15*, 2.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. Stereoset: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371.

Nedjma Ousidhoum, Xinran Zhao, Tianqing Fang, Yangqiu Song, and Dit-Yan Yeung. 2021. Probing toxic content in large pre-trained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4262–4274.

Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation sensitivity analysis to detect unintended model biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5740–5745.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14.

Salim Sazzed. 2021. A hybrid approach of opinion mining and comparative linguistic analysis of restaurant reviews. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1281–1288.

Pranav Narayanan Venkit, Mukund Srinath, and Shomir Wilson. 2022. A study of implicit bias in pretrained language models against people with disabilities. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1324–1332.

Pranav Narayanan Venkit and Shomir Wilson. 2021. Identification of bias against people with disabilities in sentiment analysis and toxicity detection models. *arXiv preprint arXiv:2111.13259*.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162.

Meredith Whittaker, Meryl Alper, Cynthia L Bennett, Sara Hendren, Liz Kaziunas, Mara Mills, Meredith Ringel Morris, Joy Rankin, Emily Rogers, Marcel Salas, et al. 2019. Disability, bias, and ai. *AI Now Institute*.

WorldBank. 2015. Individuals using the internet (% of population) - united states.

Investigating data partitioning strategies for crosslinguistic low-resource ASR evaluation

Zoey Liu
University of Florida
`liu.ying@ufl.edu`

Justin Spence
University of California, Davis
`jspence@ucdavis.edu`

Emily Prud'hommeaux
Boston College
`prudhome@bc.edu`

Abstract

Many automatic speech recognition (ASR) data sets include a single pre-defined test set consisting of one or more speakers whose speech never appears in the training set. This "hold-speaker(s)-out" data partitioning strategy, however, may not be ideal for data sets in which the number of speakers is very small. This study investigates ten different data split methods for five languages with minimal ASR training resources. We find that (1) model performance varies greatly depending on which speaker is selected for testing; (2) the average word error rate (WER) across all held-out speakers is comparable not only to the average WER over multiple random splits but also to any given individual random split; (3) WER is also generally comparable when the data is split heuristically or adversarially; (4) utterance duration and intensity are comparatively more predictive factors of variability regardless of the data split. These results suggest that the widely used hold-speakers-out approach to ASR data partitioning can yield results that do not reflect model performance on unseen data or speakers. Random splits can yield more reliable and generalizable estimates when facing data sparsity.

1 Introduction

Certain model evaluation practices are considered standard or quite common in natural language processing (NLP), such as using popular benchmarks (Bowman et al., 2015), pre-defined data partitions (Collins, 2002), or random splits (Gorman and Bedrick, 2019). All of these practices rely on metrics calculated over test sets as indices of model performance. It is not generally acknowledged that a particular numerical result might be meaningful *only for the specific train/test split that produced that result*. A single aggregated metric does not necessarily paint the full picture of a model architecture's potential (Lewis et al., 2021).

Automatic speech recognition (ASR) provides a case in point. Given a data set produced by multiple speakers, the common data partitioning strategy is "hold speaker(s) out", namely holding out all utterances from one or more speakers (Panayotov et al., 2015; Gauthier et al., 2016) as the test set, with the utterances from the remaining speakers serving as the training set. Cross-validation is generally not applied; the speakers in the test set are fixed. In other words, an ASR system is usually evaluated with just a single train/test split in which there is no speaker overlap between the training and test sets.

This common data partitioning strategy might fare well with a large data set, with recordings of dozens or hundreds of speakers, where the quantity of data and the wide array of speakers enable the training of models that are assumed to be speaker independent. The same practice, however, is not ideal for low-resource scenarios, where the number of speakers is much smaller. With endangered languages (Meek, 2012) in particular, there is much less flexibility in deciding how much data and what kind of utterances to include. Thus, observed ASR accuracy may depend heavily on which speakers appear in the test sets rather than being representative of the model architecture's general performance.

This study investigates alternative data partitioning methods for low-resource ASR. Leveraging data from five typologically distinct languages, including one endangered language, we ask: (1) How dependent is ASR performance on the identity of the held-out speaker? (2) Can alternative data partitioning strategies yield less variable estimates of a model's generalizability? (3) What factors other than speaker identity contribute to differences in model performance? (4) How can we operationalize lessons learned to improve ASR evaluation for under-resourced and endangered languages?

2 Related Work

While the hold-speaker(s)-out partitioning method is prevalent in ASR (Sikasote and Anastasopoulos, 2021; Gauthier et al., 2016; Zeyer et al., 2019;

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 123–131
May 2-6, 2023

Language	Audio data					Additional written texts	
	N of speakers/ sessions	Gender	Total utterance duration per speaker/session	Total utterance duration std.	Total utterance duration range	N of words	N of types
Fongbe	27	-	16m12s	7m12s	22m12s	990,146	8,022
Wolof	18	-	63m	18m36s	1h6m36s	601,639	29,147
Swahili	36	-	18m	7m12s	19m12s	31,540,821	471,296
Iban	23	Male: 9 Female: 14	22m12s	19m48s	1h11m24s	2,082,452	36,310
Hupa (verified)	17	Female: 1	5m24s	6m	22m48s	41,386	8,800
Hupa (coarse)	34	Female: 1	13m12s	14m24s	1h12s		

Table 1: Descriptive statistics for audio data and additional written texts used to train language models for each language in the experiments; duration range refers to the range of the distribution of the total amounts of audio per speaker. We note that our counts were derived from the public repositories and may be different from those originally reported in the papers.

Kipyatkova and Karpov, 2016), there are a number of exceptions. Laleye et al. (2016) divided utterances into three groups based on their content, then used two categories for training and one for testing. Chiu et al. (2021) tested an English ASR system trained on short audio segments on longer utterances and found poor generalization performance. With five low-resource languages as the test cases, Morris et al. (2021) re-partitioned the data where each speaker occurred in both the training and the test sets; the results showed considerable variability when compared to those derived from holding out one or a fixed set of speakers.

3 Data descriptions

We used data sets for four widely spoken low-resource languages, Fongbe (Laleye et al., 2016), Wolof (Gauthier et al., 2016), Swahili (Gelas et al., 2012), and Iban (Juan et al., 2014), which were previously released as ASR corpora. They include segmented audio with corresponding transcripts, as well as additional written texts for training the language model (see Table 1 for details).

In addition, we explored a data set of Hupa, a critically endangered language indigenous to North America. The audio recordings for Hupa are the product of ongoing linguistic fieldwork started in 2005. All the recordings were produced by a single female elder speaker, which is common for speech corpora for critically endangered languages, making Hupa a unique test bed for our study. Each transcription typically goes through several stages of correction and consultation with the elder before being considered complete; thus some transcriptions have been examined more thoroughly than others. Based only on differences in transcription quality, the audio data was divided into two sets, which we will call "verified" vs. "coarse" data respectively (details are presented in Appendix A.1).

4 Experiments

4.1 Data split methods

We first compared the commonly applied "hold speaker(s) out" (hereafter **held-out speaker**) data partitioning strategy with random splits (Gorman and Bedrick, 2019). For held-out speaker training, we set aside the data of one speaker for testing the performance of an acoustic model trained on the data of the other speakers. This procedure was repeated for all speakers in the data set. Note that this data split method was only applicable to Wolof, Fongbe, and Iban. For Swahili, which lacks information on speaker identity, and Hupa, which includes only a single speaker, we adopted what we refer to as **held-out session**. Instead of holding out the data of each speaker, we held out the utterances from each recording date or fieldwork session.

For **random splits**, each data set was randomly divided into train/test sets so that the ratio between their respective total utterance duration approximated 4:1. To arrive at more reasonable comparisons with held-out speaker, for the data set(s) of each language, the number of random splits matched the number of speakers/sessions in total.

We also explored two alternative data splitting strategies: **heuristic** and **adversarial splits** (Søgaard et al., 2021). For the former, we exploited the following features of each audio sample and its corresponding transcript: *utterance duration*, *average pitch*, *average intensity*, *the number of tokens* in the transcript, *the number of unique token types* in the transcript, and *the perplexity of the audio transcript* scored by the language model for each language (see Section 4.2). Consider the example of average pitch. We identified a pitch threshold such that utterances with an average pitch value greater than or equal to this threshold would be put into the test set, and the total duration of these utterances accounted for around 20% of the duration of the data set. Note that each heuristic split method partitioned the data into a single train/test set split.

Lastly, for adversarial splits, we first combined the transcripts of all audio data for a particular data set, then split the transcripts into train/test sets via maximizing their Wasserstein distance (Arjovsky et al., 2017; Søgaard et al., 2021), so that the token distribution of utterances in the training set is as divergent or distant as possible from that of utterances in the test set. Each data set was split into train/test sets at a 4:1 ratio, five times.

4.2 Language and acoustic models

For each language, we used SRILM (Stolcke, 2002) to build a single trigram language model with Witten-Bell discounting using the additional written texts and excluding the transcripts of the audio training data. For the acoustic model architecture, we used a fully connected deep neural network (DNN) from the open-source Kaldi toolkit (Povey et al., 2011), shown to achieve strong performance in prior studies (Morris et al., 2021; Georgescu et al., 2019; Miao et al., 2015). In particular, for small corpora, the DNN architecture has been demonstrated to yield better results than statistical alternatives (e.g., subspace Gaussian mixture models) and other neural architectures (e.g., time delay neural networks) (Morris, 2021). We also found the DNN to be substantially more accurate than the endangered language end-to-end recipe (Shi et al., 2021) in ESPnet (Watanabe et al., 2018).[1] Crucially, however, we note that our goal is not to improve upon current state-of-the-art for low-resource ASR but rather to examine what data partitioning strategies and evaluation methods lead to reliable estimates in low-resource settings with an already strong model architecture.

4.3 Regression analysis

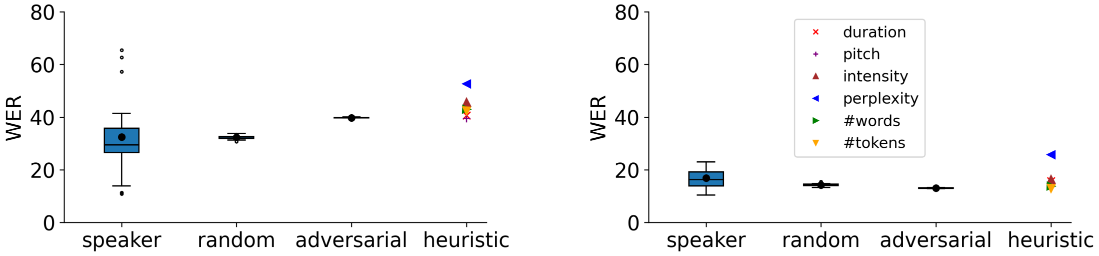
To understand which features of the splits contribute to WER variability, we carried out regression analysis. Given each data split, we first collected the following five heuristics for *each utterance* in the test set: utterance duration, average pitch, average intensity, utterance perplexity, and out-of-vocabulary (OOV) rate. Second, we computed the *average* value of each of the features for the training set as a whole. Third, we normalized the value of each feature for every utterance in the test set by the average value of the feature derived from the training set, to account for training set characteristics as well. This yielded a reasonable data size for regression modeling for each language (ranging from 6,248 instances for the verified data of Hupa to 85,920 instances for Wolof; see Anderson et al. (2021)). After repeating these steps for all data splits, we fit regression models predicting the WER of every utterance in the test set as a function of these characteristics, while controlling for the number of tokens and types in the utterance and the data split method. When possible, speaker identity and the specific utterance were included as random effects, both with random intercept and slopes for each of the fixed effects. The final regression structure was determined via backward stepwise regression from the maximal mixed-effect structure (Barr et al., 2013).

5 Results

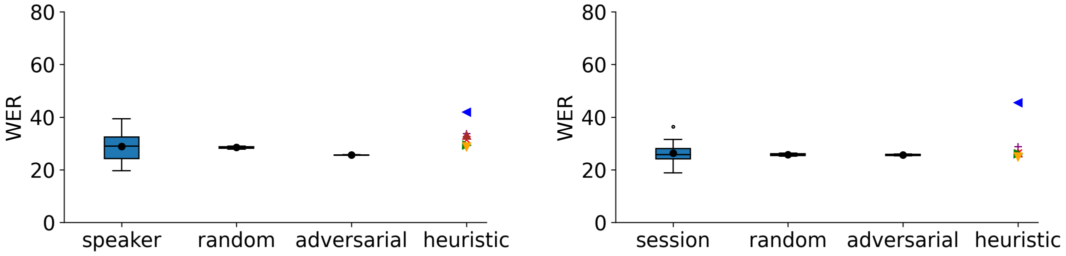
We first consider the degree of variability in WER depending on which speaker is held out. In Figure 1, for Fongbe, Wolof, and Iban, we see a high degree of variability. The WER range across held-out speakers spans from 12.71 for Iban to 54.74 for Fongbe. (See Table 3 in Appendix A.3.) Perhaps surprisingly, for Swahili and Hupa, where we held out recording sessions rather than individual speakers, we also observe great variability in model performance. The WER range across sessions is 17.59 for Swahili and is above 25 for both data sets

---

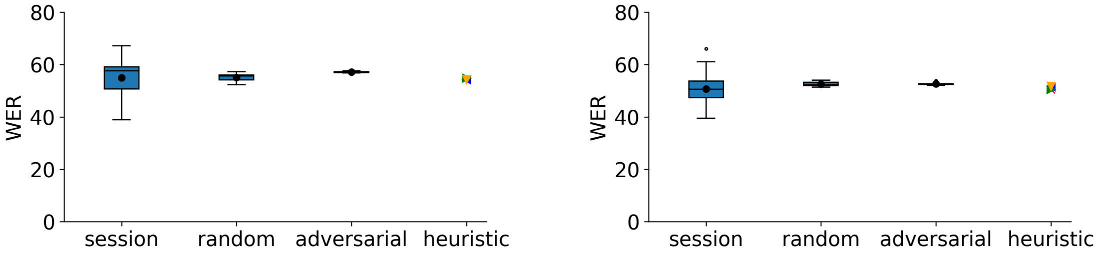
[1] Our experiments using ESPnet and wav2vec 2.0 to fine-tune from multilingual models yielded inconsistent and weak results. The necessary parameter tuning within these architectures is left for future work.



(a) Fongbe (b) Iban



(c) Wolof (d) Swahili



(e) Hupa (verified) (f) Hupa (coarse)

Figure 1: WER for the various partitioning strategies for the six data sets. A large black dot represents the mean in each plot. Means for hold-session/speaker-out are comparable in all cases to means over random splits.

of Hupa. Thus it does not appear to be the case that variability in WER is due entirely to the identity of the speaker; other factors such as recording setting or domain could contribute to this variability. These observations speak to our original point, namely that observations from "hold speaker(s) out" low-resource ASR evaluation are not representative of the model's generalizability.

One might suspect that the observed WER variability across speakers in each data set is (*only*) because of the varying amount of audio available for each speaker. Although there is a relationship between average WER and the total utterance duration per speaker (when looking at total utterance duration as a sole predictor in the regression) for Iban ($p < 0.005$), this relationship does not hold for Fongbe ($p = 0.75$) or Wolof ($p = 0.91$). While there is a positive correlation between duration and WER for the coarse data of Hupa ($p < 0.05$), this correlation does not exist for Swahili ($p = 0.45$) or for the verified data of Hupa ($p = 0.99$). This indicates that the total utterance duration *alone* is not enough to yield (high) WER variability across speakers.

In contrast, results from random splits are much less variable. While not surprising, this is notable in that the average WER when holding out a speaker or session is comparable to that of random splits, or any one random split. Thus a single random split can alone be enough to provide a reasonable estimate of the performance that would be derived by averaging over all random splits or over all possible held-out speakers/sessions. In contrast, the WER for a model tested on single held-out speaker/session may not be a reliable estimate of the WER of that model on any other speaker.

On the other hand, splitting data heuristically and adversarially, creating test sets that consist of "more challenging" cases than the training sets, does not necessarily lead to higher WER. The results across the data split methods are mostly comparable except for when utterance duration or

Language		**audio duration**	**Avg. pitch**	**Avg. intensity**	**utterance perplexity**	**OOV**	R^2
Fongbe	Coef.	0.17***	0.13**	0.11***	0.003***	0.008***	0.72
	95% CI	(0.15, 0.20)	(0.11, 0.14)	(0.09, 0.13)	(0.001, 0.004)	(0.007, 0.009)	
Wolof	Coef.	0.009	0.004	0.01	0.02***	0.00	0.73
	95% CI	(-0.01, 0.03)	(-0.01, 0.02)	(-0.02, 0.04)	(0.02, 0.024)	(0.00, 0.00)	
Swahili	Coef.	0.21***	0.06**	-1.65***	0.00	0.0131***	0.91
	95% CI	(0.18, 0.24)	(0.02, 0.09)	(-1.81, -1.50)	(-6.30, 0.00)	(0.012, 0.014)	
Iban	Coef.	2.15***	-0.10	0.58	0.00	-0.002	0.99
	95% CI	(1.87, 2.43)	(-4.20, 0.21)	(-2.16, 3.31)	(-1.11, 0.00)	(-0.01, 0.01)	
Hupa	Coef.	1.30***	-0.93***	2.74***	0.00	0.01***	0.95
	95% CI	(1.07, 1.52)	(-1.13, -0.73)	(2.09, 3.37)	(-0.02, 0.01)	(0.01, 0.02)	
Hupa	Coef.	0.13***	0.04	-0.22**	-0.02***	0.01***	0.91
	95% CI	(0.10, 0.16)	(-0.01, 0.10)	(-0.35, -0.08)	(-0.02, -0.01)	(0.01, 0.02)	

Table 2: Regression results for the data set(s) of each language in our experiments (CI stands for Confidence Interval); the number of * indicates significance level: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$. Note that given the structure of our regression model, the coefficient value for the same feature is *not comparable* across the data for each language (e.g., the coefficient of utterance duration ratio is 0.33 for the Wolof data, and 0.13 for the Swahili data; nevertheless, this does not mean that utterance duration ratio has a stronger effect for Wolof compared to its role for Swahili). Rather our goal is simply to see whether a feature potentially influences WER scores when the effects of other features are controlled for within the context of the data for every language.

perplexity is used as the heuristic in certain cases. Splitting the data by maximizing transcript distribution distance also yields minimal variability.

The regression analysis (Table 2) further indicates that most of the features we investigated have significant effects on performance. The ratios of utterance duration and intensity between the train and test sets consistently play strong roles in predicting WER variability. The fact that utterance duration has an effect on WER *when controlling for the effects of other factors* points to the potential limitation of evaluating models with held-out speakers in low-resource settings, where speakers contribute varying amounts of data.

6 Discussion and Conclusion

With data for four widely-spoken low-resource languages and one critically endangered language indigenous to North America, our work demonstrates that there is a real risk of grossly over- or underestimating the performance of an ASR model architecture when evaluating on held-out speakers (and sessions) when only minimal resources are available. By contrast, random splits provide a more accurate and less variable estimate of the overall performance. Moreover, while cross-validation is advisable when partitioning by speaker, a single random split appears to provide an adequate estimate for expected WER on unseen data.

We note that these findings also hold for data sets partitioned according to recording session rather than speaker, suggesting that this phenomenon is not limited to diversity in speaker characteristics. This has implications particularly for ASR in support of endangered language documentation, in which the number of speakers is few but the recording conditions are highly variable. We propose that future work on small ASR corpora for under-resourced languages carry out multiple evaluations on various data partitioning strategies in order to present a more complete picture of ASR model architecture performance.

7 Limitations

Our work has two notable limitations. First, the limited availability of very small ASR datasets allowed us to explore only five languages. It remains to be seen how different data splits would interact with a much larger set of languages that have more diverse typological properties. Second, here we experimented with a hybrid DNN architecture within Kaldi rather than more recent end-to-end approaches (Lin and Mak, 2020; Watanabe et al., 2018), which require more extensive computing resources. Resource permitting, we would like to investigate how different data partitioning strategies would work differently (or not) with different model architectures.

Acknowledgements

We are grateful for the continued support from the Hupa indigenous community. We would like to especially thank Mrs. Verdena Parker for her generous and valuable input for the documentation work of Hupa throughout the years. In addition, we thank the anonymous reviewers for their helpful feedback. This material is based upon work supported by the National Science Foundation under Grant #2127309 to the Computing Research Association for the CIFellows Project, and Grant #1761562. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation nor the Computing Research Association.

References

Mark Anderson, Anders Søgaard, and Carlos Gómez-Rodríguez. 2021. Replicating and extending "Because their treebanks leak": Graph isomorphism, covariants, and parser performance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1090–1098, Online. Association for Computational Linguistics.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

Dale J Barr, Roger Levy, Christoph Scheepers, and Harry J Tily. 2013. Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of memory and language*, 68(3):255–278.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Hennie Brugman and Albert Russel. 2004. Annotating multi-media/multi-modal resources with ELAN. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Chung-Cheng Chiu, Arun Narayanan, Wei Han, Rohit Prabhavalkar, Yu Zhang, Navdeep Jaitly, Ruoming Pang, Tara N Sainath, Patrick Nguyen, Liangliang Cao, and Yonghui Wu. 2021. RNN-T models fail to generalize to out-of-domain audio: Causes and solutions. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 873–880. IEEE.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8. Association for Computational Linguistics.

Elodie Gauthier, Laurent Besacier, Sylvie Voisin, Michael Melese, and Uriel Pascal Elingui. 2016. Collecting resources in sub-Saharan African languages for automatic speech recognition: a case study of Wolof. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3863–3867, Portorož, Slovenia. European Language Resources Association (ELRA).

Hadrien Gelas, Laurent Besacier, and Francois Pellegrino. 2012. Developments of Swahili resources for an automatic speech recognition system. In *SLTU - Workshop on Spoken Language Technologies for Under-Resourced Languages*, Cape-Town, Afrique Du Sud.

Alexandru-Lucian Georgescu, Horia Cucu, and Corneliu Burileanu. 2019. Kaldi-based DNN architectures for speech recognition in Romanian. In *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, pages 1–6. IEEE.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

Sarah Samson Juan, Laurent Besacier, and Solange Rossato. 2014. Semi-supervised G2P bootstrapping and its application to ASR for a very under-resourced language: Iban. In *Proceedings of Workshop for Spoken Language Technology for Under-resourced (SLTU)*.

Irina Kipyatkova and Alexey Karpov. 2016. DNN-based acoustic modeling for Russian speech recognition using Kaldi. In *International Conference on Speech and Computer*, pages 246–253. Springer.

Frejus A. A. Laleye, Laurent Besacier, Eugene C. Ezin, and Cina Motamed. 2016. First Automatic Fongbe Continuous Speech Recognition System: Development of Acoustic Models and Language Models. In *Federated Conference on Computer Science and Information Systems*.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Wei-Wei Lin and Man-Wai Mak. 2020. Wav2Spk: A Simple DNN Architecture for Learning Speaker Embeddings from Waveforms. In *The Annual Conference of the International Speech Communication Association (Interspeech)*, pages 3211–3215.

Barbra A Meek. 2012. *We are our language: An ethnography of language revitalization in a Northern Athabaskan community*. University of Arizona Press.

Yajie Miao, Hao Zhang, and Florian Metze. 2015. Speaker adaptive training of deep neural network acoustic models using i-vectors. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(11):1938–1949.

Ethan Morris. 2021. Automatic Speech Recognition for Low-Resource and Morphologically Complex Languages. Master's thesis, Rochester Institute of Technology.

Ethan Morris, Robert Jimerson, and Emily Prud'hommeaux. 2021. One size does not fit all in resource-constrained ASR. In *The Annual Conference of the International Speech Communication Association (Interspeech)*, pages 4354–4358.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an ASR corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Jiatong Shi, Jonathan D. Amith, Rey Castillo García, Esteban Guadalupe Sierra, Kevin Duh, and Shinji Watanabe. 2021. Leveraging end-to-end ASR for endangered language documentation: An empirical study on yolóxochitl Mixtec. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1134–1145, Online. Association for Computational Linguistics.

Claytone Sikasote and Antonios Anastasopoulos. 2021. Bembaspeech: A Speech Recognition Corpus for the Bemba Language. In *Proceedings of AfricaNLP*, Online.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *Seventh international conference on spoken language processing*.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-End Speech Processing Toolkit. In *Proceedings of Interspeech*, pages 2207–2211.

Albert Zeyer, Parnia Bahar, Kazuki Irie, Ralf Schlüter, and Hermann Ney. 2019. A comparison of Transformer and LSTM encoder decoder models for ASR. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 8–15. IEEE.

A Appendix

A.1 Data for Hupa

The majority of the recordings for Hupa feature the elder telling stories from different genres, including traditional stories that explain how the world we know today came to be, personal anecdotes from her life, and oral-historical accounts of significant events in her speech community. Each recording has a time-aligned transcription produced by a human transcriber using annotation tools (e.g., ELAN (Brugman and Russel, 2004)); each transcript was rendered in a practical orthography currently adopted by the speech community.

The verified transcriptions for Hupa are more accurate overall than coarse transcriptions and have undergone more orthographic normalization. This includes removing things that are audible in the recordings but not part of the standardized spelling (such as word-final epenthetic vowels), and removing false starts and other speech errors. In a small number of cases, verified transcriptions might even contain a word that is different from what was produced in the original recording, if the elder felt strongly that she had misspoken. Thus, although verified transcriptions tend to be more accurate than coarse ones, in some ways they are less faithful to the acoustic properties of the original recordings.

A.2 Acoustic models

Except for Swahili and Hupa, acoustic feature transformations for the data of the other languages were conducted separately for each speaker. In detail, the recordings were transformed to the standard 13 dimensional mel-frequency cepstral coefficients (MFCCs), along with their delta- and delta-delta features. The delta- and delta-delta features are, respectively, numerical approximations of the first and second-order derivatives of the MFCCs and they were computed on a 25ms window with 10ms interval apart, allowing for modeling of the trajectories of the audio signals. Linear Discriminant Analysis and Maximum Likelihood Linear Transform were applied to reduce the dimensionality of the feature vectors. Speaker Adaptive Training was adopted to perform speaker and noise normalization in order to make the acoustic model more attentive to the phonemic variation present in the audio, rather than being restricted by the data of particular speakers. With the speaker-normalized features, Feature Space Maximum Likelihood Linear Regression (FMLLR) was employed for speaker-independent alignment.

The DNN we adopted had six hidden layers with 1024 units in each. Sequence training was performed using the default parameters in Kaldi with state-level minimum Bayes risk criterion and a per-utterance Stochastic Gradient Descent weight update. Decoding was carried out with the Kaldi finite state transducer-based decoder.

A.3 Full WER results

Table 3 includes the full set of WER for every data partitioning strategy for each of the five languages. which is represented visually in Figure 1.

Language	**Total**	**Split method**	**Threshold**	**N of splits**	**WER**	**WER std.**	**WER range**
Fongbe	7h10m	held-out speaker	-	27	32.55	12.99	54.74
	train:5h44m	random splits	-	27	31.99	1.05	4.90
	test:1h26m	utterance duration	3.46s	1	40.69	-	-
		Avg. pitch	144.68hz	1	38.90	-	-
		Avg. intensity	58.24db	1	43.00	-	-
		N of tokens	8	1	38.95	-	-
		N of token types	8	1	40.41	-	-
		utterance perplexity	301.05	1	53.28	-	-
		distribution distance	-	5	39.74	0.18	0.41
Wolof	18h58m	held-out speaker	-	18	28.91	5.99	19.74
	train:15h11m	random splits	-	18	28.43	0.36	1.20
	test:3h47m	utterance duration	5.19s	1	31.37	-	-
		Avg. pitch	114.43hz	1	28.88	-	-
		Avg. intensity	74.32db	1	29.87	-	-
		N of tokens	11	1	28.07	-	-
		N of token types	11	1	28.85	-	-
		utterance perplexity	674.09	1	42.63	-	-
		distribution distance	-	5	25.65	0.08	0.19
Swahili	10h58m	held-out session	-	36	26.31	3.46	17.59
	train:8h47m	random splits	-	36	25.83	0.45	2.09
	test:2h11m	utterance duration	4.80s	1	26.36		
		Avg. pitch	172.07hz	1	25.89	-	-
		Avg. intensity	75.32db	1	24.98	-	-
		N of tokens	13	1	25.36	-	-
		N of token types	12	1	25.37		
		utterance perplexity	1793.95	1	45.98	-	-
		distribution distance	-	5	25.62	0.28	0.57
Iban	8h49m	held-out speaker	-	23	16.92	3.80	12.71
	train:6h49m	random splits	-	23	14.35	0.57	2.19
	test:1h42m	utterance duration	15.85s	1	15.73	-	-
		Avg. pitch	141.33hz	1	14.94	-	-
		Avg. intensity	74.34db	1	16.47	-	-
		N of tokens	36	1	13.97	-	-
		N of token types	31	1	13.90	-	-
		utterance perplexity	361.55	1	27.02	-	-
		distribution distance	-	5	13.09	0.14	0.36
Hupa	1h35m	held-out session	-	17	55.73	8.59	31.82
(verified)	train:1h16m	random splits	-	17	55.27	1.50	5.44
	test:19m	utterance duration	9.71s	1	60.12	-	-
		Avg. pitch	112.40hz	1	54.35	-	-
		Avg. intensity	67.04db	1	53.72	-	-
		N of tokens	16	1	52.17	-	-
		N of token types	14	1	54.43	-	-
		utterance perplexity	898.45	1	53.55	-	-
		distribution distance	-	5	55.10	0.47	1.28
Hupa	7h37m	held-out session	-	34	51.65	5.59	25.25
(coarse)	train:6h6m	random splits	-	34	52.85	1.18	4.63
	test:1h31m	utterance duration	10.96s	1	56.20	-	-
		Avg. pitch	113.78hz	1	52.49		
		Avg. intensity	66.12db	1	53.61		
		N of tokens	16	1	50.36	-	-
		N of token types	14	1	50.76	-	-
		utterance perplexity	933.37	1	50.26	-	-
		distribution distance	-	5	51.39	0.18	0.47

Table 3: WER results for the data set(s) of each language in our experiments; as we are focused on data partitioning strategy, for all data splits of a given data set, the language model was constant and was trained only on additional written texts. Note that one might be concerned about how much overlap there is between the test sets (and the training sets accordingly) yielded from different data partitioning strategies other than using held-out speaker/session; to address this, for the data set(s) of each language, we used the test set of the first random split as the reference and computed the proportion of overlapping utterances from the test sets of other data splits (except for held-out speaker/session); the maximum overlapping ratio across all the data sets was 0.25.

Shortcomings of Question Answering Based Factuality Frameworks for Error Localization

Ryo Kamoi **Tanya Goyal** **Greg Durrett**
Department of Computer Science
The University of Texas at Austin
`ryokamoi@utexas.edu`

Abstract

Despite recent progress in abstractive summarization, models often generate summaries with factual errors. Numerous approaches to detect these errors have been proposed, the most popular of which are question answering (QA)-based factuality metrics. These have been shown to work well at predicting summary-level factuality and have potential to localize errors within summaries, but this latter capability has not been systematically evaluated in past research. In this paper, we conduct the first such analysis and find that, contrary to our expectations, QA-based frameworks fail to correctly identify error spans in generated summaries and are outperformed by trivial exact match baselines. Our analysis reveals a major reason for such poor localization: questions generated by the QG module often inherit errors from non-factual summaries which are then propagated further into downstream modules. Moreover, even human-in-the-loop question generation cannot easily offset these problems. Our experiments conclusively show that there exist fundamental issues with localization using the QA framework which cannot be fixed solely by stronger QA and QG models.

1 Introduction

Although abstractive summarization systems (Rush et al., 2015; See et al., 2017; Lewis et al., 2020) have improved drastically over the past few years, these systems often introduce factual errors into generated summaries (Cao et al., 2018; Kryscinski et al., 2019). Recent work has proposed a number of approaches to detect these errors, including using off-the-shelf entailment models (Falke et al., 2019; Laban et al., 2022), question answering (QA) models (Chen et al., 2018; Wang et al., 2020; Durmus et al., 2020), and discriminators trained on synthetic data (Kryscinski et al., 2020). Such methods have also been explored to identify error spans within summaries (Goyal and Durrett, 2020) and perform post-hoc error correction (Dong et al., 2020; Cao et al., 2020).

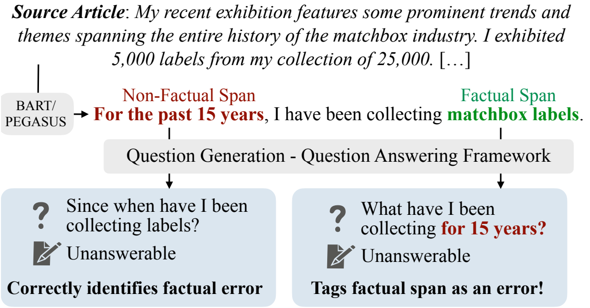


Figure 1: Factual error localization using QA metrics. Questions are generated for summary spans and then answered by a QA model using the source article as context. For factual spans (e.g. *matchbox labels*), we expect the predicted answers to match the original spans. However, non-factual spans in generated questions inherited from summaries may render these unanswerable and lead to incorrect error localization.

Among these different approaches for evaluating factuality, QA-based frameworks are the most widely adopted (Chen et al., 2018; Scialom et al., 2019; Durmus et al., 2020; Wang et al., 2020; Scialom et al., 2021; Fabbri et al., 2022). These evaluate the factuality of a set of spans in isolation, then combine them to render a summary-level judgment. Figure 1 illustrates the core mechanism: question generation (QG) is used to generate questions for a collection of summary spans, typically noun phrases or entities, which are then compared with those questions' answers based on the source document to determine factuality. Due to this span-level decomposition of factuality, QA frameworks are widely believed to localize errors (Chen et al., 2018; Wang et al., 2020; Gunasekara et al., 2021). Therefore, the metrics have been applied in settings like post-hoc error correction (Dong et al., 2020), salient (Deutsch and Roth, 2021) and incorrect (Scialom et al., 2021) span detection, and text alignment (Weiss et al., 2021). However, their ac-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 132–146
May 2-6, 2023 ©2023 Association for Computational Linguistics

tual span-level error localization performance has not been systematically evaluated in prior work.

In this paper, we aim to answer the following question: **does the actual behavior of QA-based metrics align with their motivation?** Specifically, we evaluate whether these models successfully identify error spans in generated summaries, independent of their final summary-level judgment. We conduct our analysis on two recent factuality datasets (Cao and Wang, 2021; Goyal and Durrett, 2021) derived from pre-trained summarization models on two popular benchmark datasets: CNN/DM (Hermann et al., 2015; Nallapati et al., 2016) and XSum (Narayan et al., 2018). Our results are surprising: **we find that good summary-level performance is rarely accompanied by correct span-level error detection.** Moreover, even trivial exact match baselines outperform QA metrics at error localization. Our results clearly show that although motivated by span-level decomposition of the factuality problem, the actual span-level predictions of QA metrics are very poor.

Next, we analyze these failure cases to understand why QA-based metrics diverge from their intended behavior. We find that the most serious problem lies in the question generation (QG) stage: generated questions for non-factual summaries inherit errors from the input summaries (see Figure 1). This results in poor localization wherein factual spans get classified as non-factual due to presupposition failures during QA. Furthermore, we show that such inherited errors cannot be easily avoided: decreasing the length of generated questions reduces the number of inherited errors, but very short questions can be under-specified and not provide enough context for the QA model. In fact, replacing automatic QG with human QG also does not improve the error localization of QA metrics. These results demonstrate fundamental issues with the current QA-based factuality frameworks that cannot be patched by stronger QA/QG methods.

Our contributions are as follows. (1) We show that QA-based factuality models for summarization exhibit poor error localization capabilities. (2) We provide a detailed study of factors in QG that hamper these models: inherited errors in long generated questions and trade-offs between these and short under-specified questions. (3) We conduct a human study to illustrate the issues with the QA-based factuality framework independent of particular QA or QG systems.

2 QA-Based Factuality Metrics

Recent work has proposed numerous QA-based metrics for summarization evaluation, particularly factuality (Chen et al., 2018; Scialom et al., 2019; Eyal et al., 2019; Durmus et al., 2020; Wang et al., 2020; Deutsch and Roth, 2021). These proposed metrics follow the same basic framework (described in Section 2.1), and primarily differ in the choice of off-the-shelf models used for the different framework components (discussed in Section 2.2).

2.1 Basic Framework

Given a source document D and generated summary S, the QA-based metrics output a summary-level factuality score y_S that denotes the factual consistency of S. This includes the following steps (also outlined in Figure 2):

1. **Answer Selection**: First, candidate answer spans $a_i \in S$ are extracted. These correspond to the base set of *facts* that are compared against the source document D. Metrics evaluated in this work (Scialom et al., 2021; Fabbri et al., 2022) consider all noun phrases and named entities in generated summaries as the answer candidates set, denoted by $span(S)$.
2. **Question Generation**: Next, a question generation model (G) is used to generate questions for these answer candidates with the generated summary S as context. Let $q_i = G(a_i, S)$ denote the corresponding question for span a_i.
3. **Question Filtering**: Questions for which the question answering (A) model's predicted answer $A(q_i, S)$ from the summary does not match the original span a_i are discarded, i.e., when $a_i \neq A(q_i, S)$. This step is used to ensure that the effects of erroneous question generation do not percolate down the pipeline; however, answer spans that do not pass this phase cannot be evaluated by the method.
4. **Question Answering**: For each generated question q_i, the A model is used to predict answers using the source document D as context. Let $p_i = A(q_i, D)$ denote the predicted answer.
5. **Answer Comparison**: Finally, the predicted answer p_i is compared to the expected answer a_i to compute a similarity score $sim(p_i, a_i)$. The overall summary score y_S is computed by averaging over all span-level similarity scores:

$$y_S = \frac{1}{|span(S)|} \sum_{a_i \in span(S)} sim(A(q_i, D), a_i)$$

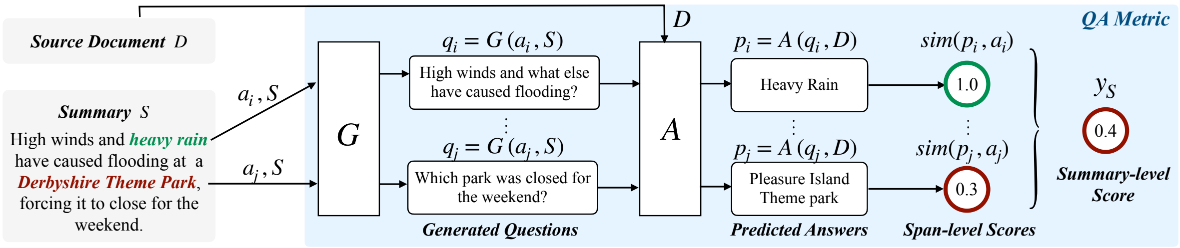


Figure 2: Overall workflow for the QA metrics. First, questions are generated for all NEs and NPs in the generated summary. Answers to these questions are obtained from the source document. Then, a factuality score is computed for each summary span based on it similarity with the predicted span from the previous step. Finally, all span-level scores are aggregated to obtain the final summary-level factuality.

Based on the motivation behind QA metrics, these similarity scores $sim(p_i, a_i)$ should indicate the factuality of the corresponding spans. If span a_i is factual, then the $G - A$ pipeline should output $p_i \in D$ with high similarity to a_i. Conversely, if a_i is non-factual, the similarity score $sim(p_i, a_i)$ should be low. While prior research has only evaluated their sentence-level performance, we use these span-level factuality scores to additionally evaluate the localization performance of QA metrics.

2.2 QA Metrics compared

In this work, we focus our analysis on the two best performing QA-based metrics from prior work:

QuestEval (QE) Scialom et al. (2021) generate questions for answer spans extracted from both the summary ("precision questions") and source document ("recall questions"). We only use the former in our experiments as these are shown to correlate better with factuality. Both the A and G components of QuestEval use T5-Large models (Raffel et al., 2020) fine-tuned on question answering datasets (Rajpurkar et al., 2018; Trischler et al., 2017). The similarity score $sim(p_i, a_i)$ in this framework is computed as the average of the lexical overlap, BERTScore, and the answerability score predicted by A.

QAFactEval (QAFE) Fabbri et al. (2022) conduct an ablation study over the different combinations of available A and G models. Here, we use their best-performing combination: an ELECTRA-based A model and a BART-based G model fine-tuned on the QA2D dataset (Demszky et al., 2018). The $sim(p_i, a_i)$ score is obtained using the learned metric LERC (Chen et al., 2020). If $A(q_i, D)$ is unanswerable for span a_i, QAFactEval sets the similarity score $sim(\_, a_i) = 0$ instead of using the LERC metric.

3 Experimental Setup

3.1 Task Definition

Given document D and a generated summary S, let $y_S^* \in \{0, 1\}$ denote the gold summary-level factuality label. Additionally, we assume access to $L = \{(a, y_a^*)\}$ which denotes the set of spans $a \in span(S)$ and their corresponding span-level gold factuality labels $y_a^* \in \{0, 1\}$.

First, we evaluate the **summary-level performance** of factuality models, i.e., is the predicted factuality equal to the gold factuality judgment y_S^*? To do this, we covert the predicted factuality score y_S to a binary judgment using dataset-specific thresholds. For each factuality model evaluated, we select thresholds that yield the best F1 scores on the validation set on each dataset.

Next, we evaluate the **span-level (localization) performance** of factuality models. Similar to the previous setting, we convert span-level predictions y_a to binary labels using the best-F1 threshold derived from the validation set. We report the macro-averaged performance at correctly predicting the span-level label y_a^* $\forall a \in span(S)$ across all (D, S) pairs in the evaluation dataset.

To align with the current QA frameworks, we restrict our evaluation to spans that correspond to named entities and noun phrases. This takes a generous view of the QA metrics' performance as it does not penalize them for failing to identify factual-errors outside NPs and NEs. This setting allows us to study the fundamental issues with the QA framework instead of those that can potentially be addressed by extending the question types considered in the framework.

Note that even for NP and NE spans, sometimes the QA metric does not return a span-level prediction if the span has not been selected as an answer candidate or has been discarded during the question filtering phase. We assume the predicted label $y_a = 1$ for such spans, as the model failed to detect any errors.[1] We discuss the performance loss due to this additional filtering step in Appendix C.

3.2 Datasets

We conduct our analysis on two human-annotated factuality datasets from prior work that provide gold annotations of factuality at the token level. To the best of our knowledge, these two are the only datasets that include span-level factuality annotation for summaries generated by SOTA models.

CLIFF (Cao and Wang, 2021) is a dataset consisting of summaries generated by BART (Lewis et al., 2020) and PEGASUS (Zhang et al., 2020) models trained on the XSum and CNN/DM summarization datasets. For each generated summary, the dataset includes token-level factuality labels $y_t^* \in \{0, 1\}$. For $y_t^* = 0$, these are additionally labeled with fine-grained error types: extrinsic, intrinsic, or requiring world knowledge.

GD21 (Goyal and Durrett, 2021) contains XSum summaries generated using a fine-tuned BART model. Similar to CLIFF, it contains token-level factuality labels for all generated summaries.

Deriving gold summary- and span-level factuality labels from human annotations To derive the summary-level gold label y_S^* from these token-level human annotations, we set $y_S^* = 1$ iff all tokens are factual, i.e. $y_t^* = 1 \; \forall t \in S$. To derive span-level gold labels, for each NP/NE span a, we set $y_a^* = 1$ iff all tokens $t \in a$ are factual.

We construct validation and test sets by dividing each dataset into equal subsets. The statistics for the test set are included in Table 1. It shows that ~26% of non-factual tokens do not correspond to NEs or NPs and are therefore ignored by the QA metrics' evaluation pipeline. Also, note that the error statistics differ for the XSum summaries in GD21 and CLIFF due to the differences in the annotation methodologies and the trained models used (both BART and PEGASUS in CLIFF vs only BART in GD21).

Label Gran.	Metric	GD21	CLIFF	
		XSum	C/D	XSum
Summ.	Total	46	150	150
	% Non-Factual	52.2	15.3	70.7
Span	# per summary	7.9	15.3	5.4
	% Non-Factual	9.9	1.9	28.1
Token	# per summary	17.1	31.6	13.1
	% Non-Factual	8.8	1.7	24.5
	% Ignored (Non-Factual)	28.9	24.5	27.6

Table 1: Test set statistics for CLIFF and GD21 at different levels of label granularity. All our evaluation is done at the summary- and span-levels to align with the QA metrics' formulation. We convert the token-level human annotations to span-level to achieve this. The table reports the % of non-factual tokens outside NE/NPs that are ignored by the QA metrics' evaluation pipeline.

3.3 Baselines for Comparison

Exact Match Baseline (EM) first extracts all nouns, proper nouns, numbers, adjectives, and pronoun tokens from the generated summary S. For these tokens, we set $y_t = 1$ if $y_t \in D$, else $y_t = 0$. We use the fraction of tokens predicted as factual as a summary-level score.

Dependency-Arc Entailment (DAE) Goyal and Durrett (2020) evaluate the factuality of each dependency arc in generated summaries separately. We follow the methodology proposed by Goyal and Durrett (2021) to derive both summary- and token-level factuality scores from these arc-level judgments. We refer readers to the original paper for further details. We use their available model checkpoint in our experiments.[2]

We convert token-level judgments from these baseline models into span-level judgments to make their outputs compatible with our evaluation framework. This is described in detail in Appendix A.3.

4 Summary vs. Span Level Performance

QA metrics motivate the use of span-level factuality as building blocks for evaluating factuality at the sentence level. Therefore, **our hypothesis is that good summary-level performance must be accompanied by good span-level performance**. Here, we test this by comparing summary- and span-level performances.

Figure 3 outlines the performance of QA metrics and baseline systems. The top row shows ROC

---

[1] Operationally, we set $sim = 6.0$ for QAFactEval and $sim = 1.0$ for QuestEval for filtered spans.

[2] Code and trained model checkpoint provided by authors at: `https://github.com/tagoyal/factuality-datasets`

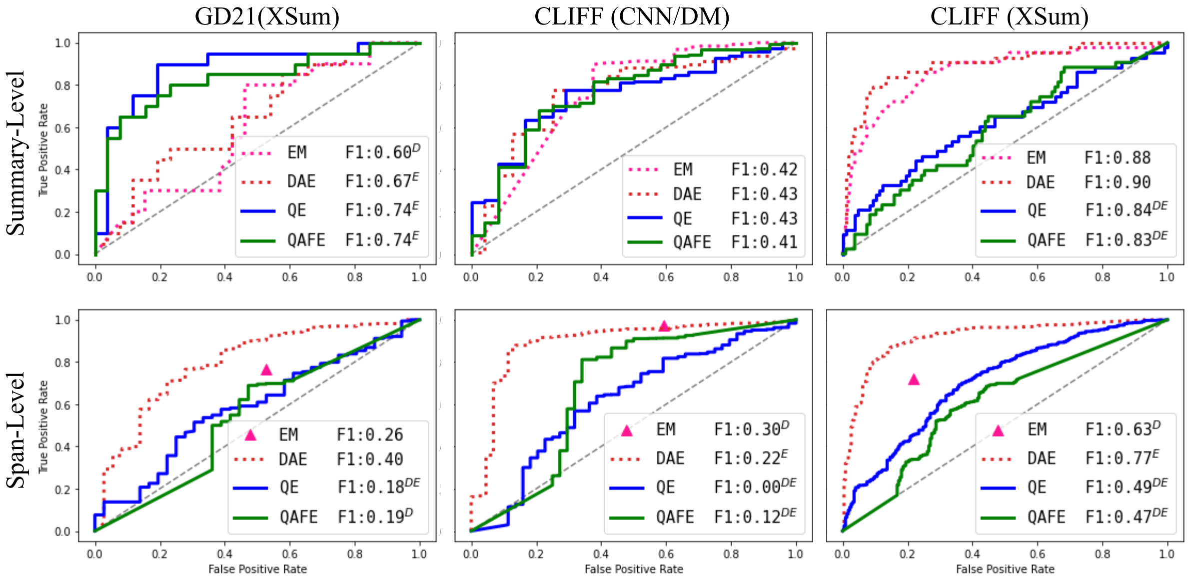


Figure 3: ROC curves and F1 scores (legend) for all systems at the summary- and span-levels (EM is a point as it provides hard binary judgments). Here, ^{D} (or ^{E}) denotes that the performance difference with DAE (or EM) is statistically significant according to a paired bootstrap test (p-value < 0.05). We observe that for QA metrics, good summary-level performance (e.g. on GD21 and CLIFF (CNN/DM)) does not imply good localization performance.

curves and F1 scores (in the legend) for all three datasets; the bottom row shows span-level results. The dotted black lines show the performance of a random baseline. First, we observe that none of the baselines or QA metrics have a clear advantage over other systems for all datasets at the summary-level. For instance, QA metrics outperform baselines on GD21, and show similar performance on CLIFF (CNN/DM) and worse performance on the CLIFF (XSum) dataset. However, **across all datasets, we see that there exists a substantial mismatch between the performance of QA metrics at the summary- and span-levels.** Notably, for GD21, both QE and QAFE substantially outperform baseline models at the summary-level, but exhibit much poorer span-level performance. Similarly, QA metrics are comparable to baselines at the summary-level for CLIFF (CNN/DM) but much worse at the span-level. On the other hand, the error localization performance of the DAE model is more consistent with its summary-level performance. Surprisingly, the trivial **exact match (EM) baseline consistently outperforms QA metrics at error localization for all datasets**. These results clearly show that our hypothesis is false: QA-based metrics do not provide reliable span-level explanations for their summary-level predictions.

Note that the diagonal lines in the span-level ROC curves for QA metrics arise due to a large number of spans being assigned the same factuality scores. As discussed in Section 2.1, some spans are filtered during the question filtering stage (Step 3) if their corresponding generated questions are of low quality. We consider these to be factual and assign them the maximum factuality score; this results in the diagonal line from $(0, 0)$.[3] We study the effects of this span filtering on localization performance in Appendix C. Additionally, QAFE assigns the same factuality score ($= 0$) to all spans with unanswerable questions resulting in the diagonal line to $(1, 1)$.

5 Why do QA metrics fail at span-level error localization?

Consider the error localization task in the example in Figure 1. Here, the QA metric needs to correctly distinguish between the factual span "*matchbox labels*" and the extrinsic error "*for the past 15 years*". For such summaries (containing a mix of factual and non-factual spans), we observed that the generated questions for factual spans often **inherit** non-factual summary spans. Given such questions, e.g. "*What have I been collecting for 15 years?*", an ideal QA model ***should*** predict unanswerable (even though that hurts localization) as the source article does not include any mention of an item being collected for 15 years. Based on this obser-

---

[3] Note that QE generates multiple questions for each span and therefore rarely discards spans (it is not likely that all questions are bad). Therefore, it has a shorter diagonal line.

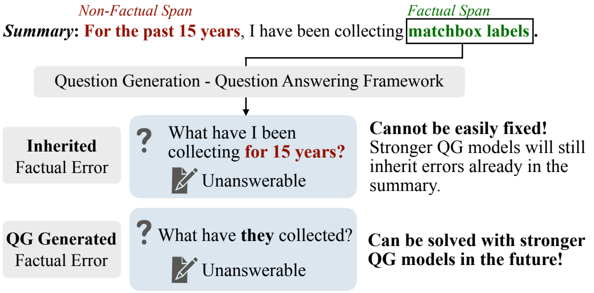


Figure 4: Generated questions broadly include two types of errors: (1) inherited errors that copy non-factual spans from the summary, and (2) errors introduced by imperfect QG models. While this latter set of errors may be eliminated by stronger QG models, inherited errors cannot be easily fixed.

vation, we hypothesize that such inherited errors in generated questions adversely affect the localization performance of automatic QA metrics by misclassifying factual spans.

First, we draw a clear distinction between (1) errors inherited from summaries, and (2) those introduced due to generation errors by the QG model. Figure 4 illustrates these two separate cases. We note that the latter set of errors can potentially be addressed by stronger QG models in the future. Our analysis in this section only studies the former set, i.e. inherited errors, as these will persist independent of the improvement in QA/QG models.

What percentage of questions are impacted by inherited errors? First, we determine the scope of the limitation introduced by inherited errors. Table 2 outlines how frequently generated questions contain inherited errors; we report these numbers only for the non-factual summaries as only these are affected by inherited errors. We define errors in a summary as inherited when a question copies at least one token that is annotated as non-factual. We use error type labels (extrinsic or intrinsic) present in the CLIFF and GD21 datasets to report separate numbers for these phenomena[4]. For the CLIFF dataset, we include world knowledge errors within the extrinsic type. In general, we observe that inherited errors are more common in QAFactEval compared to QuestEval; this can be attributed to the longer length questions generated by the former (see Appendix B for details).

QA Metric	Type of Inherited Error	GD21	CLIFF	
		XSum	C/D	XSum
QuestEval	extrinsic error	19.2	9.1	44.8
	only intrinsic error	25.7	17.3	3.6
QAFactEval	extrinsic error	39.1	11.1	93.1
	only intrinsic error	48.9	34.2	6.0

Table 2: Percentage of questions that inherit extrinsic and intrinsic errors from summaries. We only consider non-factual summaries, i.e., summaries containing at least one non-factual span in this table.

QA Metric	Type of Inherited Error	GD21	CLIFF	
		XSum	C/D	XSum
QuestEval	extrinsic error	3.1	93.9	30.5
	only intrinsic error	9.3	97.3	50.0
	no inherited error	15.5	98.7	56.0
QAFactEval	extrinsic error	7.7	65.4	63.8
	only intrinsic error	29.2	82.0	40.0
	no inherited error	29.3	92.8	65.4

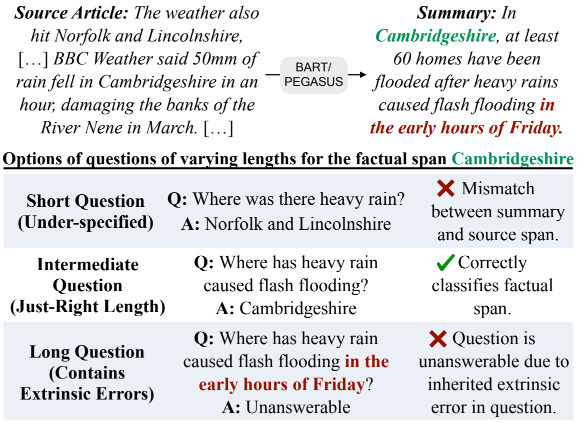
Table 3: Percentage of factual spans correctly classified by QA metrics, i.e. $y_t = y_t^* = 1$. We use the same thresholds as for F1 scores in Figure 3. Results show that inherited errors lead to more erroneous classification as non-factual across all datasets.

Do inherited errors in generated questions hurt factuality prediction? To answer this, we zoom in on factual spans in generated summaries (we consider *both* factual and non-factual summaries here), and investigate how often these are erroneously classified as non-factual. We report this for three different scenarios: (1) w/ inherited extrinsic error, (2) w/ inherited intrinsic errors only, and (3) w/o any inherited error. Table 3 outlines our results. We observe that across all settings, factual spans with inherited errors in their corresponding questions are more likely to be erroneously classified as non-factual compared to those with no inherited errors. Between error types, we observe that extrinsic inherited errors tend to harm localization more than intrinsic errors.

Note that inherited errors are only observed for summaries that are *already non-factual*. Therefore, erroneous classification of factual spans as non-factual hurts span-level but does *not* hurt summary-level performance. In fact, Fabbri et al. (2022) show that longer questions (which typically inherit more extrinsic errors, but do not cause summary-level error) exhibit better summary-level performance compared to shorter questions (which can be under-specified and cause summary-level error). This indicates that there exists a trade-off in perfor-

---

[4] Generated questions can inherit both types of errors. In the tables in this section, "extrinsic error" denotes questions that inherit at least one extrinsic error, but "only intrinsic error" denotes questions that only inherit intrinsic errors.



Short Question (Under-specified)	**Q:** Where was there heavy rain? **A:** Norfolk and Lincolnshire	✗ Mismatch between summary and source span.
Intermediate Question (Just-Right Length)	**Q:** Where has heavy rain caused flash flooding? **A:** Cambridgeshire	✓ Correctly classifies factual span.
Long Question (Contains Extrinsic Errors)	**Q:** Where has heavy rain caused flash flooding **in the early hours of Friday**? **A:** Unanswerable	✗ Question is unanswerable due to inherited extrinsic error in question.

Figure 5: Specificity and lengths of generated questions affect the answers from QA models and confuse span-level evaluation. While "*Cambridgeshire*" is a factual span, questions with inappropriate specificity can cause QA models to make mistakes. It is impossible to know what length of question is "just right" during question generation.

mance between these different granularity levels.

Can we avoid inherited errors in generated summaries? Since we do not have prior knowledge of which spans in summaries contain factual errors, we cannot trivially ensure that generated questions do not inherit the errors. One possible strategy could be to generate very short questions that include minimal details from the summary to avoid inheriting non-factual spans from the summary. However, these may then suffer from being too under-specified. We illustrate this in Figure 5. Consider the factual span "*Cambridgeshire*". The shortest question in the figure "*Where was there heavy rain?*" is not under-specified for the summary, since it is the only place name in the summary. However, there are multiple possible answers in the source document, and QA models may *reasonably* answer "*Norfolk and Lincolnshire*", leading to erroneous classification of "*Cambridgeshire*" as non-factual. Therefore, **there exists a trade-off between under-specified (short) questions and over-specified (long) questions** and it is difficult to predict the optimal level of specificity. This problem cannot be addressed by improving QA models; an ideal QA model will return unanswerable to questions with inherited errors and will be still confused by under-specified questions. We explore this issue further using human question generation in Section 6.

6 Can Human QG Improve Localization?

In Section 5, we discussed how the number of inherited errors can be indirectly influenced by varying the length of generated questions and the accompanying trade-offs: longer questions are more likely to inherit errors but shorter questions may be under-specified. Here, we investigate this using perfect QG, i.e., replacing automatic QG with humans. We evaluate two aspects: (1) How does question length impact localization? (2) Does human QG improve localization?

6.1 Experiment Design

For each summary and candidate span pair (S, a_i), we obtain human-written questions of varying lengths and information content.[5] Then, we replace the QG module of QAFactEval with these human-written questions to study the effect of question length on error localization performance.[6]

Annotators generate 3 types of questions:

1. **Shortest possible question** such that given the question-summary pair, humans can unambiguously identify the correct span in the summary.
2. **Longest question** incorporating as much information from the generated summary as reasonably allowed, often including almost an entire summary sentence.
3. **Intermediate questions** with levels of information content between the above two extremes. We allow annotators to generate any number of such intermediate questions.

Annotation and Setup We conduct this experiment on 150 randomly selected summaries from the CLIFF dataset. For the CNN/DM subset, we only selected non-factual summaries, since CNN/DM contains a small number of non-factual spans. Human annotators manually generated 2,186 questions (please refer to Appendix D for details).[7] We use half of the summaries as validation sets.

We evaluate localization performance using 4 different length configurations for human-written questions: short, long, intermediate, and oracle.

---

[5] Question lengths could also be varied if we used distinct automatic QG models, but by choosing human QG, we avoid conflating the impact of varying question specificity/length with errors or other performance differences in models.

[6] We also considered using human QA; however, we found that the QA task is ill-defined for humans when questions themselves contain extrinsic errors. Fabbri et al. (2022) also show that QA performance has less impact on factuality.

[7] Human generated questions are provided at: https://github.com/ryokamoi/QA-metrics-human-annotation

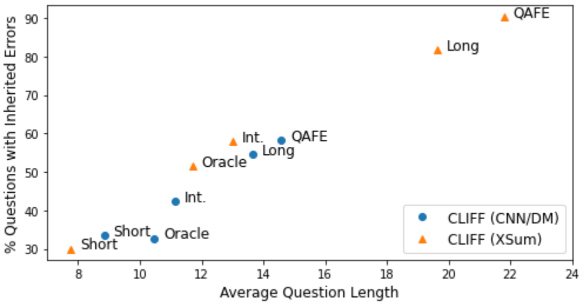


Figure 6: Statistics for questions generated by human annotators and QAFactEval (QAFE). "% Questions with Inherited Errors" is the percentage of questions that inherit non-factual spans from non-factual summaries. As expected, longer questions are more likely to inherit factual errors from the generated summaries.

For intermediate questions, evaluation is always done over three questions. We randomly subsample (or over-sample) from this set if more (or fewer) than three are available and report their average performance. If no intermediate question is written, we randomly sample from the other two categories. This only happens when the length difference between the shortest and longest questions is small. For the oracle setting, we report results using the question for each span that leads to the best localization performance. In other words, we use the highest scoring question for factual spans and the lowest for non-factual spans.

6.2 Results

Figure 6 outlines statistics for questions generated by human annotators and the QG model of QAFactEval (QAFE) generated for the same spans. As expected in Section 5, it shows that the percentage of questions that inherit non-factual spans in summaries increases with length. In this figure, we only analyze non-factual summaries since questions generated for factual summaries do not inherit errors. This result verifies our assumption and shows that we can analyze a trade-off between long questions that tend to inherit more non-factual spans from summaries and short questions with fewer inherited errors but can be under-specified.

Error Localization Figure 7 outlines the span-level localization performance for these different human question configurations and the QG model of QAFactEval. First, we notice that **human QG does not improve the localization performance of the QA frameworks**, with all three configurations exhibiting similar performance to the fully automatic QAFactEval (QAFE) model. **However, the *oracle* questions report significant improvement over QAFactEval**; this indicates that while there does exist an optimal length question for most spans, there isn't a clear pattern that can help select it during evaluation. We again note that it is not possible to select an optimal question length for each span without prior knowledge about errors in summaries. We conclude that the overall failure of human QG to improve over QAFactEval suggests that there exist fundamental issues with the QA-based factuality formulations which cannot be simply fixed by stronger QG models.

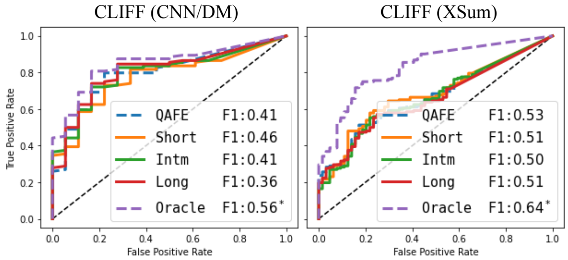


Figure 7: ROC curves and F1 scores (in legend) for span-level performances using human-written questions. These results show that no single question length configuration (except oracle) can outperform automatic QG. * denotes that improvement over QAFactEval is statistically significant (paired bootstrap test, p-value < 0.05).

7 Discussion

Analysis in our paper suggests that QA-based metrics have fundamental problems which will be difficult to address in future work. **Our view is that future system designers should favor entailment-based approaches (Falke et al., 2019; Laban et al., 2022) as a result.** One reason for this is that successful QA-based approaches actually implement something similar to entailment. Both our analysis and Fabbri et al. (2022) show that we can improve summary-level performance by generating long questions to avoid underspecified questions. However, answering questions that contain almost all the information about a sentence can be regarded as a weak form of entailment evaluation: it assesses whether the question-answer pairs that include all information about the sentence are entailed by the original document. Compared to entailment, the answer comparison step can introduce difficulties and long questions may still lead to incorrect evaluation. Since this paper shows that QA-based metrics do not lead to interpretable, localizable judgments about errors, QA-based metrics do not seem to have any structural advantage over entailment-based metrics.

8 Related Work

Recent work (Fabbri et al., 2021) has shown that popular metrics like ROUGE (Lin, 2004) and BERTScore (Zhang et al., 2019) correlate poorly with human judgments of summary quality. Factuality, in particular, has been widely studied in recent summarization literature (Kryscinski et al., 2019; Falke et al., 2019), both from the perspective of identifying non-factual generations (Wang et al., 2020; Durmus et al., 2020; Goyal and Durrett, 2020) and improving the factuality of summarization models themselves (Kang and Hashimoto, 2020; Cao and Wang, 2021).

The majority of the work in factual evaluation has focused on summary-level metrics and is not capable of localizing errors within summaries. Recent work has decomposed factuality into summaries' dependency arcs (Goyal and Durrett, 2020) or semantic-graph representations (Ribeiro et al., 2022). These localization capabilities have several downstream applications like post-editing (Zhao et al., 2020; Chen et al., 2021), removing noisy training data (Nan et al., 2021; Goyal and Durrett, 2021), among others.

9 Conclusion

In this work, we show that although QA-based factuality metrics are motivated by error localization, in practice, they exhibit extremely poor localization capabilities. We provide a detailed analysis of the different issues in current metrics that hinder better localization performance. Finally, we run a human study to investigate whether human-level QG can fix some of these issues and conclude that there exist fundamental issues with the QA framework that cannot be simply fixed by stronger models.

10 Limitations

Given the lack of prior study in error localization of summarization evaluation, there is no large-scale dataset with token-level or span-level factuality labels. Constantly-evolving summarization models also mean that any such dataset would be come outdated in a fairly short time. However, we believe that the fundamental issues we discussed with QA metrics would persist across different summarization model outputs, despite our evaluation over a limited set.

Note that all our analysis is conducted on English language datasets and models of summarization, with a limited focus on newswire summaries. We believe that the issues identified here will transfer to other languages, but other domains such as dialogue or narrative summaries may exhibit substantially different types of factuality errors. These have not been studied as heavily in prior work, so likely new techniques and analysis will be needed for these settings.

Acknowledgments

We thank Juan Diego Rodriguez for helpful comments on this work. This work was partially supported by NSF grant IIS-2145280, a gift from Salesforce Research, and a gift from Amazon. The authors acknowledge the Texas Advanced Computing Center (TACC) at the University of Texas at Austin for providing HPC resources used to conduct this research.

References

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. 2020. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, Online. Association for Computational Linguistics.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the Original: Fact-Aware Neural Abstractive Summarization. *AAAI Conference on Artificial Intelligence (AAAI)*.

Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2020. MOCHA: A dataset for training and evaluating generative reading comprehension metrics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6521–6532, Online. Association for Computational Linguistics.

Ping Chen, Fei Wu, Tong Wang, and Wei Ding. 2018. A Semantic QA-Based Approach for Text Summarization Evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Sihao Chen, Fan Zhang, Kazoo Sone, and Dan Roth. 2021. Improving Faithfulness in Abstractive Summarization with Contrast Candidate Generation and Selection. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 5935–5941.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming Question Answering Datasets Into Natural Language Inference Datasets. *arXiv preprint arXiv:1809.02922*.

Daniel Deutsch and Dan Roth. 2021. Question-Based Salient Span Selection for More Controllable Text Summarization. *arXiv preprint arXiv:2111.07935*.

Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. 2020. Multi-fact correction in abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9320–9331, Online. Association for Computational Linguistics.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Matan Eyal, Tal Baumel, and Michael Elhadad. 2019. Question answering as an automatic evaluation metric for news article summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3938–3948, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexander Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Evaluating Factuality in Generation with Dependency-level Entailment. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 3592–3603.

Tanya Goyal and Greg Durrett. 2021. Annotating and modeling fine-grained factuality in summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1449–1462, Online. Association for Computational Linguistics.

Chulaka Gunasekara, Guy Feigenblat, Benjamin Sznajder, Ranit Aharonov, and Sachindra Joshi. 2021. Using question answering rewards to improve abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 518–526, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in Neural Information Processing Systems*, 28.

Daniel Kang and Tatsunori B. Hashimoto. 2020. Improved natural language generation via loss truncation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 718–731, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Annual Meeting of the Association for Computational Linguistics*, page 7871–7880.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. *Text Summarization Branches Out*, page 74–81.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan Mcdonald. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Annual Meeting of the Association for Computational Linguistics*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21:1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Mengwen Liu, Iryna Gurevych, Markus Dreyer, and Mohit Bansal. 2022. Factgraph: Evaluating factuality in summarization with semantic graph representations. *arXiv preprint arXiv:2204.06508*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Sentence Summarization. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 379–389. Association for Computational Linguistics.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization asks for fact-based evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. Answers unite! unsupervised metrics for reinforced summarization models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3246–3256, Hong Kong, China. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Daniela Brook Weiss, Paul Roit, Ayal Klein, Ori Ernst, and Ido Dagan. 2021. QA-Align: Representing Cross-Text Content Overlap by Aligning Question-Answer Propositions. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 9879–9894.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. In *International Conference on Machine Learning (ICML)*, pages 11328–11339.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations (ICLR)*, pages 1–43.

Zheng Zhao, Shay B. Cohen, and Bonnie Webber. 2020. Reducing quantity hallucinations in abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2237–2249, Online. Association for Computational Linguistics.

A Additional Implementation Details

A.1 QuestEval

We use version 0.2.4 of the implementation by the authors with the recommended parameters (`task = 'summarization', do_weighter = False`).[8]

A.2 DAE

Our experiments use a trained model provided by authors (`DAE_xsum_human_best_ckpt`). This model is trained on a subset of XSum dataset (Maynez et al., 2020) that is distinct from examples in CLIFF and GD21.

A.3 Converting Tokens-level factuality scores to span-level scores

For the EM and the DAE baselines provide token-level factuality scores. We refer readers to the original DAE paper for details on how token-level scores are obtained (Goyal and Durrett, 2021). However, all our evaluation is designed to be at the span-level to align with QA metrics. To convert token-level scores to span-level, we annotate a span as non-factual if it contains any non-factual token.

A.4 Statistics for Ignored non-NP/NE tokens

The QA metrics do not evaluate the factuality of any token outside the boundary of a named entity of a noun phrase (discussed in Section 3.1). In Table 4, we show which kinds of tokens are ignored by the QA metrics but annotated as non-factual in our human annotated factuality datasets. Figure 5 provides an illustrative example of such ignored non-factual tokens in the different datasets).

	GD21	CLIFF	
	XSum	C/D	XSum
adposition	30.8	25.3	29.9
verb	24.4	23.7	23.5
auxiliary	15.2	10.0	17.1
punctuation	12.6	21.2	17.5
particle	6.7	3.9	4.5

Table 4: Statistics for non-factual POS-tags ouside the NP/NE boundaries and ignored by the QA metrics.

B Statistics of Generated Questions in QA-Based Metrics

Table 6 provides statistics for the generated questions from QuestEval and QAFactEval, highlighting the difference between these two metrics. On average, QAFactEval generates much longer questions. On the other hand, QuestEval generates a larger number of questions as it often generates multiple questions per candidate span.

CLIFF (XSum)
An environmental permit **has been revoked following** a fire **at** a fuel recycling plant **in** Manchester**.**
CLIFF (CNN/DM)
A Japan Railway maglev train **hit** 603 kilometers **per** hour**(**374 miles **per** hour**)** **on** an experimental track **in** Yamanashi Tuesday**.** A spokesperson **said** the train **spent** 10.8 seconds **traveling above** 600 km **per** hour **, during** which it **covered** 1.8 kilometers **(** 1.1 miles **)** That **'s** nearly 20 football fields **in** the time it **took** you **to read** the last two sentences **.** Japan Railways **has been testing** their train **to figure out** the best operational speed **for** a planned route **between** Tokyo **and** Nagoya **scheduled to begin** service **in** 2027 **.**
GD21
high winds **and** heavy rain **have caused** flooding **at** a derbyshire theme park**, forcing** it **to close for** the weekend**.**

Table 5: Example of non-factual tokens outside NP/NE boundaries and ignored by the QA metric in factuality evaluation.

	Avg Question Length			Avg No. Questions		
	GD21	CLIFF		GD21	CLIFF	
	XSum	C/D	XSum	XSum	C/D	XSum
QAFactEval	26.3	16.2	21.1	4.9	10.0	3.9
QuestEval	16.4	11.6	13.9	8.3	20.5	7.5

Table 6: Statistics for the generated questions for the QuestEval and QAFactEval metrics.

C Performance Loss due to Span Filtering

In Section 3.1, we discussed that the current QA metrics do not evaluate non-NP/NE spans. These operational shortcomings prevent these metrics from providing a complete picture of error localization over all summary tokens. Here, we discuss another similar issue arising due to the question filtering step of the overall workflow (Step 3).

Although QA metrics select all NP/NE spans for evaluation during the candidate selection stage (Step1), some of these are filtered out if their corresponding question is of low quality: a_i for which $A(q_i, S) \neq a_i$ are also discarded from further evaluation. Since no errors are detected in these spans, they are considered to be factual.

We observed that this question filtering step removes around 30% of the NE/NPs in the QAFactEval framework. This implies that this metric only

---

[8] https://github.com/ThomasScialom/QuestEval

Model	GD21	CLIFF	
	XSum	CNN/DM	XSum
EM	0.30	**0.27**	0.64^{D}
DAE	**0.32**	0.20^{E}	**0.78**
QE	0.19^{DE}	0.06^{DE}	0.45^{DE}
QAFE	0.21^{D}	0.13^{DE}	0.49^{DE}

Table 7: Span-level performance (F1 scores) over the subset of NP/NEs that are not discarded by either of the two QA metrics. $^{\text{D}}$ (or $^{\text{E}}$) denotes that the performance difference with DAE (or EM) is statistically significant according to a paired bootstrap test (p-value < 0.05). Even under this generous setting, we observe that the QA metric show very poor performance.

evaluates 70% of the valid spans, potentially missing factual errors in the remaining NP/NEs. Note that these numbers are considerably lower for QuestEval (<5%) as it generates multiple questions for each candidate span and hence is more likely to include an acceptable question.

As this impacted the results in Figure 3, we can ask what is the performance of the QA metrics over spans that they **actually** evaluate for factuality? If this performance is high, we can reasonably assume that the QA metrics' localization capabilities can be improved through better question generation models. Table 7 outlines our results: we report F1 scores at the span-level when evaluating over the subset of NP/NEs that are evaluated by both the QE and QAFE models. Although the QA metrics report improved results over those reported in Figure 3, these are still low enough so as to not be useful for error localization in practical settings.

Figure 8 shows the corresponding ROC curves for these. These show similar trends: the performance of QAFactEval improves when evaluating on this subset, but is still not better than the baseline models. Interestingly, for CLIFF (CNN/DM), we found that most of this improvement comes from the subset of candidate spans whose questions contain only a small fraction of factual errors (Figure 9). This aligns with our analysis in Section 5 that showed that errors in generated questions inherited from non-factual summaries was one of the major reasons for performance degradation, since questions generated from summaries with small number of errors are expected to inherit fewer errors.

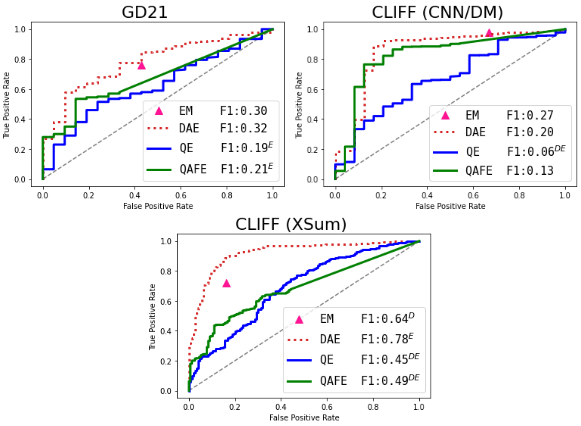
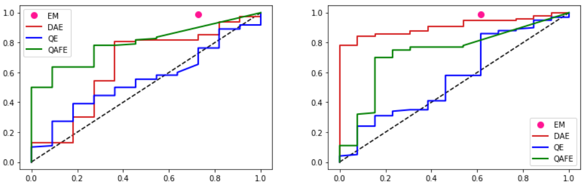


Figure 8: ROC curves for span-level performance on the subset of NE/NPs evaluated by all the QA metrics.



(a) Bottom 50% of the summaries according to error rate

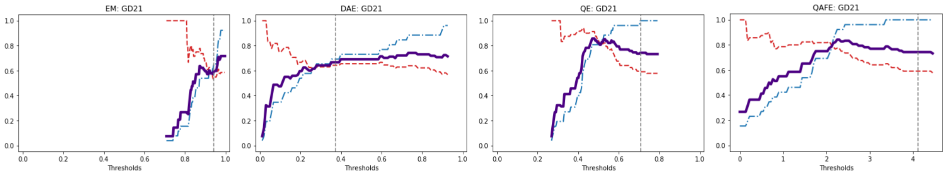
(b) Top 50% of the summaries according to error rate

Figure 9: Comparison of ROC curves for the span-level performance on CLIFF (CNN/DM). There shows results on the subset of NP/NEs actually judged by QA metrics. The graphs show that the performance of the QAFE is better on the subset of summaries with a smaller number of errors.

D Additional Details about Human QG Annotation

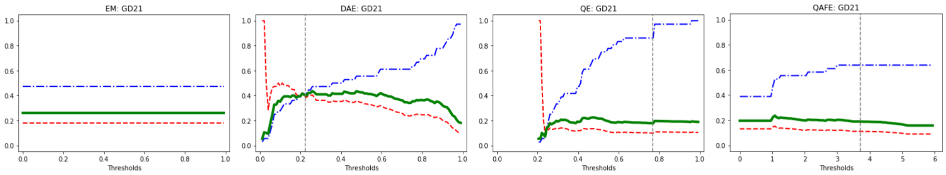
The human annotation in Section 6 was done by the authors of this paper. They were provided with summaries and extracted answer candidate spans. For spans that were judged to be invalid (e.g. "*it*"), they were asked to manually discard these spans. For all others, questions of varying lengths and specificity were written. See an example in Table 8. To aid in this question writing step, we also provide the corresponding QAFactEval questions. For the longest questions, we found that annotators often chose to build on these questions albeit with corrections to the structure and grammar of the automatic questions.

Table 9 shows the number of summaries, spans, and generated summaries annotated in our human QG experiments. We use half of the summaries as validation sets.



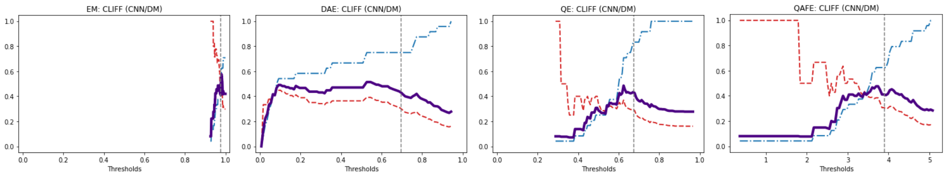
(a) EM Baseline (b) DAE (c) QuestEval (d) QAFactEval

Figure 10: Summary-level F1 performance on the GD21 test set at different thresholds



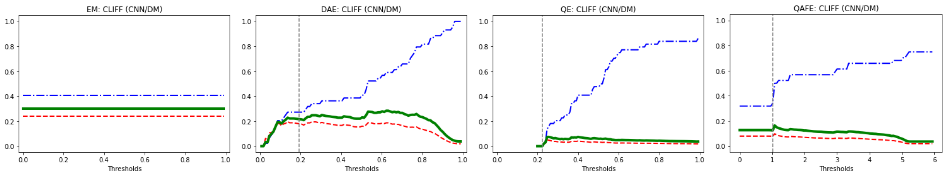
(a) EM Baseline (b) DAE (c) QuestEval (d) QAFactEval

Figure 11: Span-level F1 performance on the GD21 test set at different thresholds



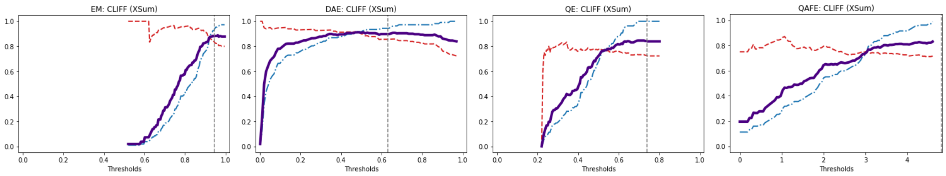
(a) EM Baseline (b) DAE (c) QuestEval (d) QAFactEval

Figure 12: Summary-level F1 performance on the CLIFF (CNNDM) test set at different thresholds.



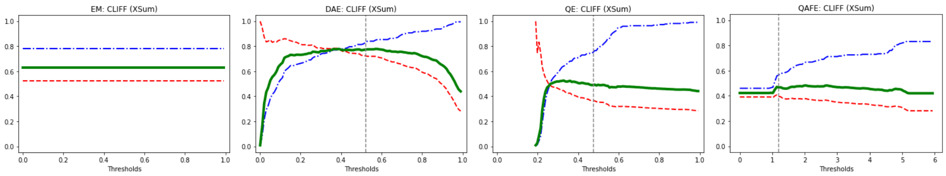
(a) EM Baseline (b) DAE (c) QuestEval (d) QAFactEval

Figure 13: Span-level F1 performance on the CLIFF (CNNDM) test set at different thresholds.



(a) EM Baseline (b) DAE (c) QuestEval (d) QAFactEval

Figure 14: Summary-level F1 performance on CLIFF (XSum) test set at different thresholds.



(a) EM Baseline (b) DAE (c) QuestEval (d) QAFactEval

Figure 15: Span-level F1 performance on the CLIFF (XSum) test set at different thresholds.

Summary	Plans to build a new hospital **in Somerset** have been given a **£3m** boost by the government.
Selected Span	a new hospital
Shortest	What does the plan propose to build?
Intermediate	What does the plan that has been given a boost propose to build?
Longest	What does the plan that has been given a **£3m** boost by the government propose to build?
QAFactEval	What plans to build **in Somerset** have been given a **£3m** boost by the government?

Table 8: Example of human-generated questions.

	CLIFF (CNN/DM)	CLIFF (XSum)
# Summary	30	120
# Span	323	470
# Questions	737	1449

Table 9: Number of summaries, spans, and generated questions annotated by human QG.

E Summary and Span Level Evaluation

We provide additional results for our experiments in Figure 3. Figure 10 to 15 show F1 scores, precision, and recall on test sets with different thresholds. These show that QA-based metrics cannot yield high precision at any threshold.

Socratic Question Generation: A Novel Dataset, Models, and Evaluation

Beng Heng Ang,[1] Sujatha Das Gollapalli,[2] See-Kiong Ng[2]
[1] Integrative Sciences and Engineering Programme, National University of Singapore
[2] Institute of Data Science, National University of Singapore
`bengheng.ang@u.nus.edu, {idssdg,seekiong}@nus.edu.sg`

Abstract

Socratic questioning is a form of reflective inquiry often employed in education to encourage critical thinking in students, or to elicit awareness of beliefs and perspectives in a subject during therapeutic counseling. Specific types of Socratic questions are employed for enabling reasoning and alternate views against the context of individual personal opinions on a topic. Socratic contexts are different from traditional question generation contexts where "answer-seeking" questions are generated against a given formal passage on a topic, narrative stories or conversations.

We present *SocratiQ*, the first large dataset of 110K (question, context) pairs for enabling studies on Socratic Question Generation (*SoQG*). We provide an in-depth study on the various types of Socratic questions and present models for generating Socratic questions against a given context through prompt tuning. Our automated and human evaluation results demonstrate that our *SoQG* models can produce realistic, type-sensitive, human-like Socratic questions enabling potential applications in counseling and coaching.

1 Introduction

Researchers in Education and Psychology have recognized the role of cognitive biases in shaping a person's perspective towards learning and understanding (Azzopardi, 2021; Bautista, 2014; Tversky and Kahneman, 1974; Vittorio et al., 2021). Indeed, both pedagogical and counseling environments involve recognizing and alleviating any flawed cognitive biases in students/subjects through appropriate interventions by trained professionals (Bhardwaj et al., 2018). Socratic Questioning is one such intervention technique pervasive in Education and Psychotherapy (Bautista, 2014; Chew et al., 2019; Vittorio et al., 2022).

Socratic Questioning involves the use of specific types of probing questions that guide people into eliciting biases underlying their understanding of a topic in order to potentially enable alternative perspectives and further thoughts (Paul and Binker, 1990; Paul and Elder, 2019). Figure 1 illustrates example Socratic questions on a passage expressing an individual's views on "eating meat".

Thought/Passage: I believe that eating meat is ethically wrong. Since we can easily substitute meat with vegan food without much nutritional complications, we have no logical reasons to continue eating living animals. We should stop killing these animals.

Possible Socratic Questions:

1. But what about eating animals that pass on from natural causes?
2. Where is your source that says that vegan food and meat have the same nutrition value?
3. What will happen if we continue killing animals for our consumption?

Figure 1: Example Socratic Questions

In this paper, we study automatic Socratic Question Generation (*SoQG*) as a novel, multidisciplinary application area for question generation (QG) research in NLP. Consider for instance, the publicly-accessible datasets available for learning question and dialog generation models (Rajpurkar et al., 2016; Ramnath et al., 2021; Talmor et al., 2017; Trischler et al., 2017; Yang et al., 2018). These existing QG datasets were designed for machine comprehension and comprise of news articles, Wikipedia, and other well-written formal passages and questions whose answers are potentially expressed in the passages. Similarly, current dialog generation datasets focus on capturing conversations related to completing specific tasks (such as restaurant booking, customer service) or learning open-domain conversational chatbots for chitchat (Byrne et al., 2021; Cui et al., 2020; Danescu-Niculescu-Mizil and Lee, 2011; Zhou et al., 2018). As such, these existing datasets do not comprise of contexts necessary for learning *SoQG*.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 147–165
May 2-6, 2023 ©2023 Association for Computational Linguistics

Socratic contexts differ from the scenarios captured in existing datasets in the following significant ways: 1. *Question Context.* The contexts or passages express individual thoughts and personal opinions on specific topics; 2. *Question Objective.* The questions do not seek a "correct answer" and aim to provoke introspection and reflection from the question recipient; and 3. *Question Type.* The questions adhere to specific Socratic types that challenge the completeness and accuracy of the thought expressed in the context in various ways. For example, the question-1 in Figure 1 seeks an *alternative perspective*, while question-2 probes for *evidence* regarding a claim expressed in the passage, and question-3 invokes further thought on *implications* of a specific action mentioned in the passage.

In this paper, we present a first study on Socratic Question Generation (*SoQG*) and make the following contributions:

1. We describe *SocratiQ*,[1] a large dataset of ~110k Socratic questions, question type labels, and their contexts to enable learning of *SoQG* models. We discuss the curation of *SocratiQ* from the large corpus of posts and replies available from the community discussion website Reddit.[2] Socratic question-type labels collected through crowdsourcing are included for a subset of questions in *SocratiQ*.

2. We study question-type prediction for *SocratiQ* using state-of-the-art deep learning models. For a given (question, context) pair, our best fine-tuned BERT classifier is able to identify its question type with a macro F1-score of 0.905. We use this highly-accurate classifier to provide question-type labels for all instances in *SocratiQ* for learning type-sensitive question generation.

3. We learn type-sensitive question generation based on Socratic question types. We extend state-of-the-art QG models based on GPT-2, T5, and ProphetNet to incorporate Socratic question types through *prompt-tuning*. Our models effectively generate realistic, relevant human-like Socratic questions as shown in automatic as well as human evaluation studies.

Through our findings as well as released resources, we hope to enable future research on QG and chatbot applications based on the Socratic Questioning paradigm in areas such as coaching and counseling.

Organization: We describe the details of creating *SocratiQ* in Section 2. Section 3 provides details of the models and baselines used to study *SoQG*. Our experimental setup, evaluations, and results are summarized in Section 4 after which we provide a brief summary of existing datasets and methods for QG in Section 5. Finally, we conclude the paper in Section 6 and present some limitations that can be addressed in future in Section 7.

2 Dataset Collection

To construct a dataset of contentious viewpoints and the questions challenging these viewpoints, we consider the social news, content, and discussions website Reddit, in particular, the subreddit *r/changemyview*, or CMV. The CMV platform is an active, targetted community described as "a place to post an opinion you accept may be flawed, in an effort to understand other perspectives on the issue" with people specially encouraged to "enter with a mindset for conversation, not debate". Upon closer examination of the posts, we found that CMV posts pertain to various controversial topics (e.g. politics, media, culture, etc.) and, due to its very purpose of "change my view", subsequent comments often include questions aimed to evoke introspection and reflection (further discussed in Section 2.1).

We obtained the raw data from CMV using the "Pushshift Reddit" API provided by Baumgartner et al (2020). After removing moderator comments that are not relevant to the topic,[3] we identify questions from comments following a given post using regular expressions comprising of question cues (e.g. who/what/where/when/why/how), and lexical indicators such as '?'. Next, each identified question sequence is paired with the most relevant paragraph sequence (context) in the previous posts based on the similarity of the two sequences. Sentence BERT encodings (Reimers and Gurevych, 2019) were used for similarity computation. After manually inspecting several examples and their similarity values, we retain pairs having a similarity value above the threshold of 0.55. In total, we produce a dataset of $110,050$ English[4] (question, context) pairs from the CMV content generated during January 2013 and December 2021. On average, the number of sentences in each context is $4(\pm 2)$

---

[1] https://github.com/NUS-IDS/eacl23_soqg

[2] https://www.reddit.com/r/changemyview/

[3] e.g., "All comments that earned delta ... are listed here"

[4] https://tinyurl.com/yzkxb2mz

Question type	Description	Exemplar
Clarification	Question probing the ambiguities of a thought.	What do mean by ...?
Probing assumptions	Question probing the assumptions behind a thought.	Why do you assume ...?
Probing reasons and evidences	Question probing the justifications or concrete evidences that could have supported a thought.	How did you know that ...?
Probing implications and consequences	Question probing the impacts or implications of a thought.	If ..., what is likely to happen as a result?
Probing alternative viewpoints and perspectives	Question probing other possible viewpoints.	What else should we consider about ...?
Others*	Question unrelated to the question types above (e.g. rhetorical, irrelevant, and/or illogical questions, etc.)	Who wouldn't want to be rich?

Table 1: Description and exemplar for each Socratic Question-Type from Paul and Elder (1990; 2019). *We add the catch-all type Others to refer to questions that do not conform to Socratic categories.

comprising of $83(\pm 53)$ words whereas questions are a sentence long with $12(\pm 6)$ words. We refer to this collection of (question, context) pairs as *SocratiQ*.

2.1 Question Annotation

Practitioners of Socratic Questioning employ various types of questions to engage in discussions with their subjects. For example, a counselor may ask for further clarification of a given viewpoint, or request evidence based on which such a stance was reached. Similar to previous studies (Dinkins and Cangelosi, 2019; Neenan, 2009; Wilberding, 2021), we used the taxonomy of Paul and Elder (1990; 2019) for characterizing the different questions. Table 1 lists the types of questions from this taxonomy with their descriptions and exemplars.

To curate the question type information for *SoQG* (Section 3), we adopt a semi-automatic process by first collecting manual annotations on a representative sample of data and training an accurate classifier for annotating the full dataset.

Crowd Labeling Process: To reduce the manual annotation efforts and cover the range of different question types uniformly in *SocratiQ*, we employ the process suggested in previous works (Abdul-Mageed and Ungar, 2017): First, lexical cues from available exemplars for each type (See Table 1) are used to design regular expressions[5] to assign a tentative Socratic type to each question in the dataset. We then use these tentative label assignments to sample questions from all types uniformly. In this manner, we sampled a set of 600 (question, context) pairs for each type, resulting in a balanced subset of $3,600$ questions for which we obtain human-assigned Socratic question-type labels.

Our annotation task was set up on the crowdsourcing platform Amazon Mechanical Turk (Crowston, 2012),[6] following previous works on QA/QG dataset collection (Ko et al., 2020; Rajpurkar et al., 2016; Yang et al., 2018). Each (question, context) pair was examined by three independent crowdworkers who chose one unique question type from among the six types listed in Table 1.[7]

We ensure the *ethics, quality, and reliability* considerations for all our collected datasets as follows. On the AMT platform, we required the crowdworkers to have greater than 95% HIT approval rate, a minimum of 10,000 HITs, be located in the United States and score at least 80% on a qualification test we set up to be able to work on our task. Each worker was paid up to 0.25 USD per HIT based on the task. In total, we used the AMT platform to collect data for the classification task as well as two evaluation studies which are further described in Section 4. The *anonymity* and *privacy* of the crowdworkers was already ensured on the AMT platform. We chose the pay-per-HIT based on similar on-going tasks on AMT. The settings for the HIT approval rates, and location of the worker ensures the English language skills of the annotator and thereby, the *quality* of the dataset.

A total of 40 workers helped in creating our classification dataset. About 40% of the workers who attempted our qualification test gained the eligibility to work on our task. Through this qualification test, we ensure that their worker annotations are *reliable*. For the $3,600$ (question, context) pairs deployed on AMT, we were able to obtain majority

---

[5] Examples are included in Appendix B.

[6] https://www.mturk.com/

[7] AMT interface details are provided in Appendix E.

annotations ($\geq$ 2 agreement on labels) for about 3,169 pairs. The Fleiss Kappa (Fleiss, 1971) inter-annotator agreement value was $\kappa = 0.725$ indicating substantial agreement among the crowd annotators. Overall, about 1% of the annotated questions were marked with the "Others" class which includes meaningless or irrelevant questions, and questions that do not belong to the Socratic types. This low percentage highlights the effectiveness of our pre-processing pipeline in extracting relevant Socratic (question, context) pairs.

Annotating the full dataset: We obtained question types for the entire *SocratiQ* dataset using an accurate classifier trained with the AMT labeled dataset. We further detail our classification experiments in Section 4. To summarize, in both automatic and manual evaluation studies, the prediction F1 and accuracy of our best classifier are around 0.9. Predicted Socratic classes are used for training prompt-based QG models described in Section 3.

2.2 Dataset Analysis

The question type distribution of the manually-annotated subset of *SocratiQ* is shown in Figure 2. Here, we observe that questions probing for *implications and consequences* and *reasons and evidence* comprise the most-asked questions. Previous studies have highlighted that "people routinely ask clarifying questions" to make sure "they can better offer assistance to the original poster" (Rao and Daumé III, 2018). This aspect is also seen in our dataset where *clarification* questions form the next most-asked questions. This is followed by questions probing for *assumptions* and *alternative viewpoints* as these are more cognitively-challenging than clarification questions (Krathwohl, 2002) while the "Other" questions form a negligible fraction (1.3%) of the overall distribution. Based on corpus analysis with Latent Dirichlet Analysis (Blei et al., 2003),[8] some representative topics in *SocratiQ* include *Abortion, Politics, Taxes, Crimes, Veganism, Racism, and Religion.*

3 Socratic Question Generation

To establish baseline performance on *SocratiQ*, we follow the current practice in NLP by fine-tuning state-of-the-art large pretrained language models (PLMs) with *SocratiQ* for **So**cratic **Q**uestion **G**eneration (*SoQG*) (Brown et al., 2020; Peters et al., 2019; Yu et al., 2021). We use commonly employed transformer-based language models, namely, Generative Pre-trained Transformer or GPT (Radford et al., 2019), the text-to-text transfer transformer or T5 (Raffel et al., 2019)), and ProphetNet (Qi et al., 2020). All three models yield relatively high performance on various NLP tasks (Wolf et al., 2020) including QG on other datasets (Chan and Fan, 2019; Ko et al., 2020).

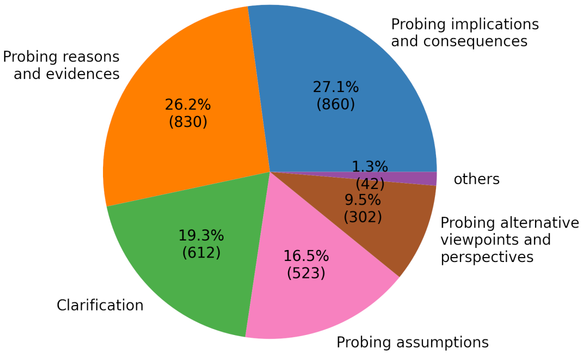


Figure 2: Distribution of questions based on their Socratic types

QG models: Following standard answer-agnostic QG models (Scialom et al., 2019), for our first set of baselines (GPT, T5, ProphetNet), we directly fine-tuned the PLMs with *SocratiQ* paragraph contexts as inputs and questions as outputs. We devised a second set of models (GPT-p, T5-p, ProphetNet-p) by employing prompt-based learning in keeping with recent developments in controllable text generation (Carlsson et al., 2022; Lester et al., 2021; Zhang et al., 2022). In this second set of models, we use the Socratic question-types as prompts in addition to the paragraph contexts as inputs for learning QG (Figure 3).

Evaluation: We use standard n-gram based metrics used for measuring question generation performance (Pan et al., 2019), namely BLEU (Papineni et al., 2002a), METEOR (Banerjee and Lavie, 2005), and ROUGE (Lin, 2004a). Previous works have highlighted the problems of using n-gram based measures for QG (Nema and Khapra, 2018). For *SoQG* in particular, these measures are limited in handling equally valid paraphrases of the available reference questions. To address these limitations, we adopt recently-designed learnt metrics, namely, BERT_Score and BLEURT (Sellam et al., 2020; Zhang et al., 2020; Yuan et al., 2021). Metrics based on the BERT-based models (Devlin et al., 2019) were shown to provide robust evaluation for

---

[8] Topic analysis results are included in Appendix A

GPT Input	<tag> [reasons_evidence] <tag> The government are people ... kind of things. <delim> If people are fantastic at managing all kinds of things, why are they so bad at them when organized as a government?
T5 Input	reasons_evidence: The government are people ... kind of things.
T5 Target	If people are fantastic ... bad at them when organized as a government?
ProphetNet Input	<tag> reasons_evidence <tag> The government are people ... kind of things.
ProphetNet Target	If people are fantastic ... bad at them when organized as a government?

Figure 3: Training input for each QG model. Text in red shows is the prompt based on question type annotation. <delim> is the delimiter token for separating input from the output whereas <tag> highlights the prompt tokens.

text generation tasks such as summarization and translation.

Configuration	R	P	F1
RE	0.681	0.706	0.617
L_{train}	0.797	0.929	0.828
$(RE_D) + L_{train}$	0.779	0.779	0.778
$(L_{train} + U_{RE})$	0.708	0.691	0.627
SelfTrain	0.826	0.883	0.846
GANBERT	0.784	**0.930**	0.802
$L_{train} + U_{V*}$	**0.933**	0.887	**0.905**

Table 2: Question-Type Classification Performance

4 Experiments

Setup: We set up our question-type classification and question generation experiments on a single GPU NVIDIA Tesla V100 machine. For classification, all models are based on the BERT classifier (Devlin et al., 2019).[9] For QG experiments, we use the implementations provided by the transformers library for GPT-2[10] and T5[11] and ProphetNet.[12] On our experimental machine, all models take between $12 - 19$ hours for training depending on the task, specific model and the dataset used.[13]

4.1 Classification Results

We studied several configurations based on BERT for training our question-type classifier. Table 2 shows the performance of these configurations on 20% (randomly-selected) test split of the AMT-annotated dataset using macro-averages of standard measures Recall/Precision/F1 employed for multiclass classification. We used the larger (80%) portion of the dataset for training and validation. The *SocratiQ* dataset contains $D \approx 110$K (question, context) pairs from which about $L \approx 3.1$K instances were collectively annotated by crowdworkers (Section 2.1). Thus, the unlabeled data ($U = D \setminus L$) can also be used to improve the classification performance via semi-supervised learning methods.

In Table 2, we show the performance using regular expression patterns in the first row (RE row) and the performance of BERT in the basic setting (fine-tuning on L_{train}) in the second row. The "$(RE_D) + L_{train}$" row refers to fine-tuning BERT on the tentative labels obtained with regular expressions before fine-tuning on L_{train}, whereas in the fourth row ($L_{train} + U_{RE}$), we add the unlabeled data using predictions with regular expressions to the training dataset.

The next three rows show semi-supervised configurations that involve the use of the unlabeled data (U) during model training. The "SelfTrain" row shows the performance in the self-training mode where the predictions from basic BERT configuration (second row) was used to predict labels for U and then re-trained on the combined dataset (Du et al., 2021; Mukherjee and Awadallah, 2020). The performance with the recently proposed semi-supervised model for BERT, namely, GANBERT (Croce et al., 2020) is shown in the next row.[14] For the final configuration in the last row ($L_{train} + U_{V*}$), we used voting to choose the dominant label from predictions on U with all configurations and retrained our classifier (Bishop, 2006). We did not include the regular expression-based models (RE and $L_{train} + U_{RE}$) in voting due to their substantially lower performance.

All classification models were fine-tuned using both the context and question as inputs. It is worth noting that the F1 performance of using labels based on regular expressions is significantly lower than the basic BERT configuration. Using only regular expressions or adding labels from regular expressions to unlabeled data results in significantly lower test performances (first and fourth rows) highlighting the inadequacy of only relying on exemplar

---

[9] https://huggingface.co/bert-large-cased

[10] https://huggingface.co/gpt2-large

[11] https://huggingface.co/t5-large

[12] https://github.com/microsoft/ProphetNet

[13] The details of deep learning experiment configurations are included in Appendix D

[14] https://github.com/crux82/ganbert

Model	BLEU-1	BLEU-4	METEOR	ROUGE-L	BERT_Score	BLEURT
GPT-p	0.165	0.013	0.167	0.187	0.615	0.423
GPT	0.150	0.007	0.149	0.167	0.601	0.412
T5-p	0.172	0.017	0.170	**0.211**	**0.632**	**0.426**
T5	0.144	0.011	0.142	0.179	0.615	0.413
ProphetNet-p	**0.178**	**0.018**	**0.177**	0.208	**0.632**	0.425
ProphetNet	0.152	0.011	0.147	0.178	0.616	0.416

Table 3: QG performance is shown using standard measures and BLEURT and BERT_Score values. All models that use question-type prompts perform significantly better compared to their respective baselines (t(10k), p <.05).

and templates for identifying Socratic Question type information. The best-performing configuration is obtained through combining all models to obtain the dominant label for unlabeled examples, resulting in a significant jump in F1 values as shown in the last row, even though semi-supervised learning using self-training and GANBERT have individually yielded small and no improvements, respectively.

We also experimented with only question tokens as input to the models. In general, the performance using question only as input is moderately high with the macro-F1 decrease of 1-8% compared to when both the context and question are used. Indeed, when the scores of tokens from the context versus the question are computed for their attribution towards prediction using the Integrated Gradients method from Sundararajan et al. (2017), we found that the average attribution score of question tokens, 0.118, is significantly much larger than that of the context tokens, 0.033 (Cohen's D value (2013) of 1.58). This suggests that the input tokens from question have more discriminatory power for predicting question type. Indeed, using context-tokens only as input, our basic BERT configuration obtained a macro-F1 score of only 0.185 indicating that the judgement of question-type is, not surprisingly, highly dependent on the question tokens as was also indicated in previous studies on question-type identification (Li and Roth, 2002; Svikhnushina et al., 2022).[15]

4.2 Question Generation Results

Next, we evaluate the models described in Section 3 on *SocratiQ*. For the models using question-type information (GPT-p, T5-p, ProphetNet-p), we append the predictions from our best classifier (last row in Table 2) with the context and appropriate separator tokens. For the (GPT, T5, ProphetNet) baselines, the context alone forms the input. We randomly split 105K pairs of *SocratiQ* that have a Socratic question label into Train/Dev/Test portions in the ratio "80/10/10" and show the test performance of QG models in Table 3.

Automatic Evaluation: As the reference questions from Redditors are available in *SocratiQ*, we directly employ the n-gram measures for evaluating QG performance. The BLEU-1/BLEU-4 values are shown along with METEOR and ROUGE scores in Table 3. We note that by incorporating question types, the prompt-based models do significantly better than their non-prompt counterparts for all three PLMs: GPT, T5, and ProphetNet. However, the overall n-gram overlap with reference questions is very low with the best BLEU-4 score of 0.018 and the best ROUGE score value of 0.211. For comparison, the ProphetNet model could yield a BLEU-4 score ranging from $0.23 - 0.25$ on test splits of SQuAD dataset (a well-used dataset in QG research) (Rajpurkar et al., 2016; Qi et al., 2020). Unlike these factual questions, Socratic questions can be fairly diverse for a given context and question type. Rather than measuring n-gram overlap, we posit that measuring semantic similarity between a given reference and generated question using recent text generation metrics such as BERT_Score and BLEURT, would be more indicative of the models' performance.[16]

Indeed, as shown in Table 3, the values of these learnt metrics are significantly higher than the n-gram based measures, with ProphetNet-p being the best performing model. When the generated questions are manually examined, we found that the fine-tuned PLMs indeed generate human-like questions which are relevant to the given contexts even though they do not match the given reference questions. This aspect was confirmed in our manual evaluation studies described next.

---

[15] Further details are included in Appendix C

[16] F1 metric was used for BERT_score. The precise models in both cases are described in Appendix D

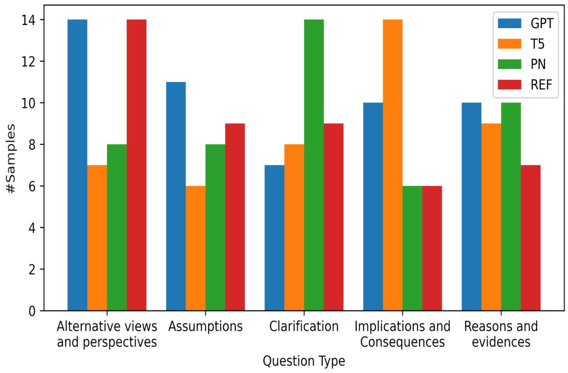


Figure 4: Distribution of "most likely to ask" questions chosen by crowdworkers

4.3 Human Evaluation Studies

We set up manual evaluation studies on Amazon Mechanical Turk to provide a more comprehensive analysis of *SoQG* model outputs. In the first study, we randomly sample fifty contexts for each Socratic question type of *SocratiQ*. For each context and Socratic question type, three independent MTurk evaluators select among four randomly-ordered questions (3 generated by GPT-p, T5-p, and ProphetNet-p + 1 reference) the question they will likely ask, or the "None" option.[17] In Figure 4, we plot the distribution of the source of the selected questions for each question type using majority votes from the above study.

Model	Flu	Rel	Ans
Best	4.287	3.883	0.330
Worst	4.220	3.737	0.353

Table 4: Quality of Questions (Flu:fluency, Rel:relevance, Ans:answerability)

We found that only two questions out of the 250 questions used in the study were deemed unacceptable for the given question-type and context by our annotators, confirming the accuracy of our question-type classifier. For the remaining questions, we see from Figure 4 that based on the question type, crowdworkers seem to prefer questions generated by a specific model and interestingly, reference questions from CMV were not always the clear winner. For example, the questions generated by ProphetNet were clearly preferred for the "Clarification" type of questions whereas T5-generated questions were preferred for "Probing implications and consequences" type. In fact, the human-reference questions were on par with the best model only in one type out of five indicating that our models are generating fluent and very human-like Socratic questions.

In a second related study, we sample twenty contexts for each of the five question types in the previous study, select the questions from the best and worst models (according to Figure 4), and ask three independent AMT crowdworkers to rate on a Likert scale of 1-5 ("Very bad" to "Very good") how fluent and relevant the questions are for a given context. They were also asked "if the answer to the question is present in the passage" (0/1 indicating absence/presence of the answer). The results for this study are summarized in Table 4. We found that the questions from the best models for each type indeed have higher fluency and relevance compared to questions from the worst model and both scores are close to the "Good" range on the Likert scale. Moreover, the answerability scores less than 0.5 indicate that the questions are not answerable from the context (different from traditional QG datasets).

Finally, we randomly sampled and examined fifty ground-truth questions per question type. These questions were examined on a 0/1 scale for classifier accuracy, fluency, answerability, and relevance. We found that the accuracy values ranged between 0.86-0.96, fluency and relevance between 0.92-1.00, and answerability was near zero (0.04), among the five Socratic classes. These scores further support our best classifier's performance on the AMT annotated test dataset, and that the ground-truth questions are fluent, relevant, and unanswerable from the context. These numbers indicate that our dataset of 110K (question, context) pairs and their predicted Socratic question types are of high quality and can be used by fellow NLP researchers for further improving *SoQG* models.

Anecdotes: The model-generated questions for an example context from *SocratiQ* are shown in Table 5 for illustration. We see that our prompt-based models are able to incorporate the question type information to generate questions that are semantically similar to the reference questions.

SQuAD versus *SocratiQ*: To highlight how Socratic questions complement existing QA/QG datasets, we run *SoQG* for an example from SQuAD (Rajpurkar et al., 2016).[18] For this context,

---

[17] Further details are included in Appendix E

[18] https://rajpurkar.github.io/SQuAD-explorer/explore/v2.0/dev/Immune_system.html

Context: Beyond the boundaries of legal conduct, to exclude people from the society is judged to be a suitable place of the law. For some heinous conduct, it's acceptable to put people beyond the society, and exclude them for tye benefit of all. But the denial of voting rights to convicted felons after the end of thier sentence is not acceptable. **1 - Probing implications and consequences**: Are there serious risks to allowing felons to vote? **2 - Probing reasons and evidences**: Why are we concerned about felons voting and what value does purging felons from voter rolls have to our society?
GPT-*p* - 1: What will happen if the felons were convicted of a crime? **GPT-*p* - 2**: Why should felons be excluded from voting rights? **GPT**: Why should felons be denied the right to vote?
T5-*p* - 1: What will happen if the felons are not allowed to vote? **T5-*p* - 2**: Why is it acceptable to deprive people of their right to vote? **T5**: Is it acceptable to deprive convicted felons of their right to vote?
ProphetNet-*p* - 1: Are you implying that felons should be allowed to vote? **ProphetNet-*p* - 2**: Why is it acceptable to deny felons voting rights after the end of their sentence? **ProphetNet**: What about people who have been convicted of murder?

Table 5: Questions generated by the different models are shown for an example context and associated reference questions from *SocratiQ*. Table 12 of the Appendix contains some more examples.

note the contrast between the "fact-seeking" questions from SQuAD and Socratic questions listed in Table 6. Socratic questions focus on "what if/why/on what basis" angles, causing the reader to reflect on alternate perspectives and implications, which can be useful for nurturing further thought and analysis which are essential in Education.

Reference questions from SQuAD 1. What are the anitmicrobial peptides secreted by the skin called? 2. What enzymes in saliva are antibacterial in nature? 3. What compounds in the stomach protect against ingested pathogens? 4. Vaginal secretions serve as a chemical protective barrier following what?
T5-*p* generated Socratic Questions 1. How do these enzymes kill pathogen? 2. Is it a good thing to assume that the stomach and intestinal tract are chemically different? 3. Do you have evidence that these enzymes are effective at killing pathogen? 4. Are you implying that the stomach is more acidic than other organs?

Table 6: Example from the SQuAD Dataset

We present in Table 7 an example context from the Real-world Worry Dataset (Kleinberg et al., 2020). Against this context, *SoQG*-models generate questions that draw attention to the underlying assumptions and potential misconceptions related to the mentioned "worry" causing the reader to introspect, which can be useful during counseling.

Context: I am concerned for my family's safety and I'm worried about the impact isolation will have on the mental health of my loved ones. **GPT-*p* generated Socratic Questions** 1. What does isolation have to do with your mental health or safety? 2. Why do you assume that isolation will lead to mental health issues? 3. Have you considered that isolation is a mental health issue that affects the entire family?

Table 7: Example from Real-world Worry Dataset

5 Related Works

Question generation has been a prominent subject of recent NLP research (Pan et al., 2019; Lu and Lu, 2021). Except some QG datasets which discuss inquisitive, probing questions (Ko et al., 2020), clarification questions (Rao and Daumé III, 2018), and empathetic questions in social dialogs (Svikhnushina et al., 2022), most models are trained on datasets such as SQuAD (Rajpurkar et al., 2016), HotpotQA (Yang et al., 2018), QuAC (Choi et al., 2018), and CoQA (Reddy et al., 2019), WebQA (Talmor et al., 2017) that were originally created for question answering (Pan et al., 2019). Not only are these datasets targeted towards extractive or abstractive QA and therefore comprise mostly factual, answer-seeking questions, but their contexts also include non-personal contexts such as Wikipedia or news articles, narrative stories, or problem descriptions on forums such as Stack Exchange (Rao and Daumé III, 2018). In contrast, Socratic contexts involve personal and individual opinions and viewpoints on diverse topics with no fixation on "correct" answer (Paul and Elder, 2019).

For learning QG models, we adopt the current approach of fine-tuning transformers on specific datasets (Wolf et al., 2020; Kriangchaivech and Wangperawong, 2019). Though *SoQG* is an answer-agnostic QG task (Scialom et al., 2019), due to its uniqueness in availability of question type information, we extend our QG models to incorporate these cues according to latest research on conditional text generation through the use of prompts (Lester et al., 2021; Zhang et al., 2022; Carlsson et al., 2022). Question-type taxonomies were previously studied for factual questions (Li and Roth, 2002) and in context of social dialog (Svikhnushina et al., 2022).

6 Conclusions

We created a novel dataset *SocratiQ* to support research on automatic Socratic Question Generation. We applied latest research in prompt-based conditional text generation to fine-tune existing large language models from GPT, T5, and ProphetNet to learn *SoQG*. Through our study and the release of this novel dataset, we take a first step towards enabling future research on models for *SoQG* as well as impactful applications in areas such as counseling and education (Inkster et al., 2018; Fitzpatrick et al., 2017).

7 Limitations

We note the following limitations in our work that also comprise our future research directions. First, while a human Socratic method practitioner will know what type of Socratic question to ask based only on context, our prompt-based models assume the availability of question-type for generating a type-sensitive question. In fact, when only contexts were used for QG (GPT, T5, ProphetNet baselines in Section 3), the generated questions matched the desired question-type (those of the available reference questions) in only 37-40% of the cases. Furthermore, the question-type identification of automated methods using context alone was very poor with overall accuracy comparable to that of random assignments (Section 4.1).

Secondly, though we showcased the potential use of *SoQG* in designing chatbots and dialog systems for applications such as counseling, we note that the current evaluation has only been at the single-turn level. We hope to extend *SocratiQ* to capture back and forth discussions on CMV to provide multiturn data and also deduce via forum votes and other indicators if the discussion indeed resulted in changed minds and enabled alternate perspectives. Furthermore, considering the special purpose of Socratic questions in shaping perspectives and enabling introspection and reflection, a comprehensive evaluation would require measuring these aspects over the multi-turn sessions.

Finally, our dataset was created by re-purposing the CMV subreddit data available in English, a high-resource language for which large-scale pretrained language models (PLMs) are readily available. Obtaining high classification and generation performances via fine-tuning of PLMs will be a challenge that needs addressing in low-resource languages.

Acknowledgement

This research/project is supported by the National Research Foundation, Singapore under its Industry Alignment Fund–Pre-positioning (IAF-PP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. We thank our reviewers for their suggestions on improving the presentation of this paper and insights on future possibilities.

Ethics Statement

This research was conducted in accordance with the ACM Code of Ethics. The ethical considerations during the dataset collection process are discussed in Section 2.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. Emonet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 718–728.

Leif Azzopardi. 2021. Cognitive biases in search: a review and reflection of cognitive biases in information retrieval. In *Proceedings of the 2021 conference on human information interaction and retrieval*, pages 27–37.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. In *Proceedings of the international AAAI conference on web and social media*, volume 14, pages 830–839.

Lowell Bautista. 2014. The socratic method as a pedagogical method in legal education. *Faculty of Law, Humanities and the Arts - Papers. 1481.*

Gaurab Bhardwaj, Alia Crocker, Jonathan Sims, and Richard D Wang. 2018. Alleviating the plunging-in bias, elevating strategic problem-solving. *Academy of Management Learning & Education*, 17(3):279–301.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Bill Byrne, Karthik Krishnamoorthi, Saravanan Ganesh, and Mihir Kale. 2021. TicketTalk: Toward human-level performance with end-to-end, transaction-based dialog systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 671–680, Online. Association for Computational Linguistics.

Fredrik Carlsson, Joey Öhman, Fangyu Liu, Severine Verlinden, Joakim Nivre, and Magnus Sahlgren. 2022. Fine-grained controllable text generation using non-residual prompting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6837–6857, Dublin, Ireland. Association for Computational Linguistics.

Devrim Cavusoglu, Fatih Cagatay Akyon, Ulas Sert, and Cemil Cengiz. 2022. Jury: Comprehensive NLP Evaluation toolkit.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China. Association for Computational Linguistics.

Sie Wai Chew, I-Hsiu Lin, and Nian-Shing Chen. 2019. Using socratic questioning strategy to enhance critical thinking skill of elementary school students. In *2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT)*, volume 2161, pages 290–294. IEEE.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Jacob Cohen. 2013. *Statistical power analysis for the behavioral sciences*. Routledge.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

Kevin Crowston. 2012. Amazon mechanical turk: A research tool for organizations and information systems scholars. In *Shaping the Future of ICT Research. Methods and Approaches*, pages 210–221, Berlin, Heidelberg. Springer Berlin Heidelberg.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual*

Meeting of the Association for Computational Linguistics, pages 1406–1416, Online. Association for Computational Linguistics.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 76–87, Portland, Oregon, USA. Association for Computational Linguistics.

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

Christine Sorrell Dinkins and Pamela R Cangelosi. 2019. Putting socrates back in socratic method: Theory-based debriefing in the nursing classroom. *Nursing Philosophy*, 20(2):e12240.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Veselin Stoyanov, and Alexis Conneau. 2021. Self-training improves pre-training for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5408–5418, Online. Association for Computational Linguistics.

Kathleen Kara Fitzpatrick, Alison Darcy, and Molly Vierhile. 2017. Delivering cognitive behavior therapy to young adults with symptoms of depression and anxiety using a fully automated conversational agent (woebot): a randomized controlled trial. *JMIR mental health*, 4(2):e7785.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Loshchilov Ilya and Hutter Frank. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR), 2019.*

Becky Inkster, Shubhankar Sarda, Vinod Subramanian, et al. 2018. An empathy-driven, conversational artificial intelligence agent (wysa) for digital mental well-being: real-world data evaluation mixed-methods study. *JMIR mHealth and uHealth*, 6(11):e12106.

Intel. 2007. Designing effective projects: Questioning the socratic questioning technique.

Bennett Kleinberg, Isabelle van der Vegt, and Maximilian Mozes. 2020. Measuring Emotions in the COVID-19 Real World Worry Dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Wei-Jen Ko, Te-yuan Chen, Yiyan Huang, Greg Durrett, and Junyi Jessy Li. 2020. Inquisitive question generation for high level text comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6544–6555, Online. Association for Computational Linguistics.

David R Krathwohl. 2002. A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218.

Kettip Kriangchaivech and Artit Wangperawong. 2019. Question generation by transformers.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Chin-Yew Lin. 2004a. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chin-Yew Lin. 2004b. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chao-Yi Lu and Sin-En Lu. 2021. A survey of approaches to automatic question generation:from 2019 to early 2021. In *Proceedings of the 33rd Conference on Computational Linguistics and Speech Processing (ROCLING 2021)*, pages 151–162, Taoyuan, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. Http://mallet.cs.umass.edu.

David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 262–272, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Subhabrata (Subho) Mukherjee and Ahmed H. Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. In *NeurIPS 2020 (Spotlight)*. ACM.

Michael Neenan. 2009. Using socratic questioning in coaching. *Journal of Rational-Emotive & Cognitive-Behavior Therapy*, 27(4):249–264.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, Brussels, Belgium. Association for Computational Linguistics.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *CoRR*, abs/1905.08949.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002a. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002b. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Richard Paul and Linda Elder. 2019. *The thinker's guide to Socratic questioning*. Rowman & Littlefield.

Richard W Paul and AJA Binker. 1990. *Critical thinking: What every person needs to survive in a rapidly changing world.* ERIC.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2401–2410.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Kiran Ramnath, Leda Sari, Mark Hasegawa-Johnson, and Chang Yoo. 2021. Worldly wise (WoW) - cross-lingual knowledge fusion for fact-based visual spoken-question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1908–1919, Online. Association for Computational Linguistics.

Sudha Rao and Hal Daumé III. 2018. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2736–2745.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano. 2019. Self-attention architectures for answer-agnostic neural question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6027–6032, Florence, Italy. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of ACL*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Ekaterina Svikhnushina, Iuliana Voinea, Anuradha Welivita, and Pearl Pu. 2022. A taxonomy of empathetic questions in social dialogs. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2952–2973, Dublin, Ireland. Association for Computational Linguistics.

Alon Talmor, Mor Geva, and Jonathan Berant. 2017. Evaluating semantic parsing against a simple web-based question answering model. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 161–167, Vancouver, Canada. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Amos Tversky and Daniel Kahneman. 1974. Judgment under uncertainty: Heuristics and biases: Biases in judgments reveal some heuristics of thinking under uncertainty. *science*, 185(4157):1124–1131.

Lisa N Vittorio, Justin D Braun, Jennifer S Cheavens, and Daniel R Strunk. 2021. Cognitive bias and medication use moderate the relation of socratic questioning and symptom change in cognitive behavioral therapy of depression. *Cognitive Therapy and Research*, 45(6):1235–1245.

Lisa N Vittorio, Samuel T Murphy, Justin D Braun, and Daniel R Strunk. 2022. Using socratic questioning to promote cognitive change and achieve depressive symptom reduction: Evidence of cognitive change as a mediator. *Behaviour research and therapy*, page 104035.

Erick Wilberding. 2021. *Socratic methods in the classroom: Encouraging critical thinking and problem solving through dialogue*. Routledge.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2021. Fine-tuning pre-trained language model with weak supervision: A contrastive-regularized self-training approach. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1063–1077, Online. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *CoRR*, abs/2201.05337.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

Supplementary Materials

A Composition of *SocratiQ*

We apply Latent Dirichlet Analysis using the topic modeling toolkit, Mallet (McCallum, 2002).[19] The top-10 topics (by their Topic Coherence values (Mimno et al., 2011)) uncovered in *SocratiQ* with LDA (number of topics set to 30) are shown with their top words in Table 8. Based on these words, we note that some representative topics in *SocratiQ* include *Abortion, Politics, Taxes, Gender, Crimes, Veganism, Racisim, and Religion*.

	Top Words
1	child abortion life human children fetus woman baby parents mother
2	vote trump party voting president election states political democrats republicans
3	money people pay work tax make taxes government income wage company
4	women men gender trans woman male sex man female people transgender
5	crime rape people death crimes punishment person prison murder law
6	animals food meat eat animal eating humans dog dogs killing
7	people white black racist racism race culture person racial group
8	im view people dont ive change argument opinion post point
9	religion god religious christian bible religions islam people beliefs christianity
10	school college students education schools student high class job degree

Table 8: Words from the Top-10 topics (as ranked by Topic Coherence) from LDA analysis are shown

Figure 5 illustrates the distribution of question types predicted by our best classifier over *SocratiQ*. Compared to the manually annotated spread shown in Figure 4, we note that the percentage of Clarification type questions is significantly higher (33% versus 19%). We attribute this to the fact that Clarification class seems to be the most-confused one among the others as shown in the confusion matrix on the test set presented in Figure 6. As such, even our best-performing classifier is not 100% accurate.

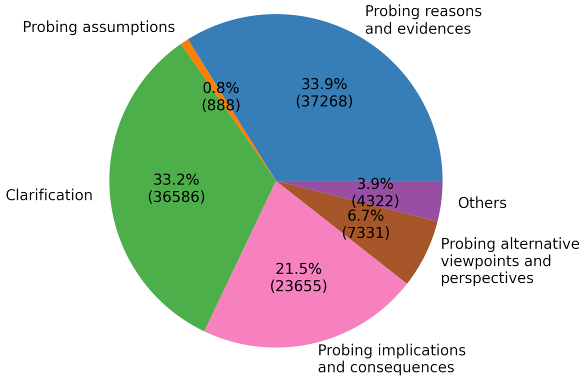


Figure 5: Distribution of predicted question types in the *SocratiQ*

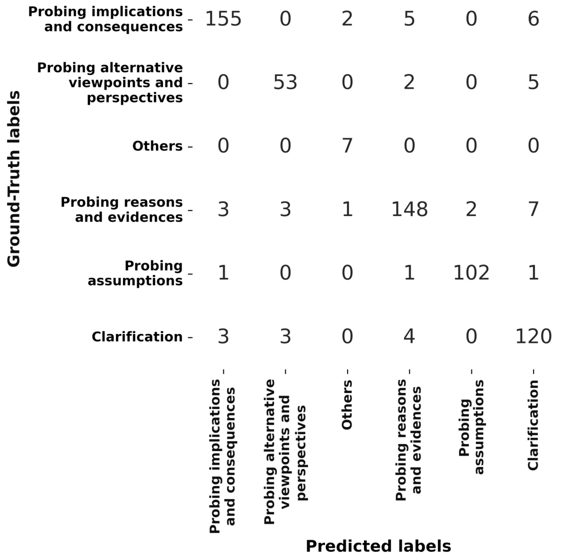


Figure 6: Confusion matrix of best classifier's predictions on the test set

---

[19] https://mallet.cs.umass.edu

Question type	Lexical Cues	Regex
Clarification	What do mean by ...?/What is the meaning of ...?	[Ww]hat[\s\w]+mean
	How is ... related to ...?/How are they related?	[Hh]ow[\s\w]+relate
Probing Assumptions	Are you assuming that...?/Why are you assuming that...?	[Aa]re[\s\w]+assum[eing]+
	What is the assumption from...	[Ww]hat[\s\w]+assumption
Probing reasons and evidences	Where is your evidence that...?	[Ww]here[\s\w]evidence
	Why do you think...?	[Ww]hy[\s\w]+think
Probing implications and consequences	Are you implying...?	[Aa]re[\s\w]+impl[yied]
	What happens...?What would have happened if...?	[Ww]hat[\s\w]+happen
Probing alternative viewpoints and perspectives	What else...?	[Ww]hat[\s\w]+else
	What other...?/What are the other...?	[Ww]hat[\s\w]]+other

Table 9: Sample lexical cues and regular expression patterns used to tentatively map questions to their Socratic question types.

B Regular Expressions

Table 9 provides sample lexical cues and their corresponding regular expressions (regex) designed based on available Socratic question templates (Paul and Elder, 2019; Intel, 2007). We use them to tentatively assign labels to instances in *SocratiQ*. These labels are used to get more balanced samples of questions for the human annotation process described in Section 2. The full set of regular expressions will be made available through our code repository.

C Computing Attributions for Classifier Predictions

We explain the model predictions by applying the Integrated Gradients method (Sundararajan et al., 2017).[20] For a question type prediction, each token is given an attribution score that indicates its contribution to that prediction. We normalize the scores by averaging them over the sequence length of each question and context sequences. From Table 10, we observe that the attribution values from the question sequences is significantly higher than those from the contexts; thereby, suggesting that the classifier predicts the question types mainly based on tokens from the question.

Based on the attribution scores, we can extract textual patterns that our classifier associates with particular question types in a data-driven fashion by simply using the top N-grams with the highest attribution scores. We show sample 4-grams in Table 11. We note the similarities between these automatically extracted patterns with the template-

Question Type	Qseq attr	Cseq attr	Cohen's D
Overall	.118	.033	1.59**
Clarification	.148	.041	1.85**
Probing reasons and evidences	.125	.037	1.64**
Probing implication and consequences	.108	.032	1.51**
Probing assumptions	.087	.020	1.76**
Probing alternate viewpoints and perspectives	.110	.025	1.80**
Others	.131	.061	0.76**

Table 10: Attribution scores (attr) for question sequences remain consistently higher than those of context sequences across every question type. Qseq indicates "Question Sequence" and "Cseq" indicates "Context Sequence". ** indicates a statistically significant, $p < 0.05$, Cohen's D effect size (Cohen, 2013) of the difference in attributions.

---

[20] https://captum.ai/

exemplars used by existing Socratic practitioners listed in Table 1.

GPT Input	<tag> [reasons_evidence] <tag> The government are people ... kind of things. <delim> If people are fantastic at managing all kinds of things, why are they so bad at them when organized as a government?
T5 Input	reasons_evidence: The government are people ... kind of things.
T5 Target	If people are fantastic ... bad at them when organized as a government?
ProphetNet Input	<tag> reasons_evidence <tag> The government are people ... kind of things.
ProphetNet Target	If people are fantastic ... bad at them when organized as a government?

Figure 7: Training input for each QG model. Text in red shows is the prompt based on question type annotation. <delim> is the delimiter token for separating input from the output whereas <tag> highlights the prompt tokens.

D Configuration Details for Deep Learning Experiments

For our classification experiments in Section. 4.1, we fine-tuned the "bert-large-cased"[9] model for the various configurations (Table 2). The model is tuned by an AdamW optimizer (Ilya and Frank, 2019) set with betas default at (0.9, 0.999), a batch of 4, and an initial learning rate of 1e-6. We also use a ReduceLROnPlateau[21] learning rate scheduler to reduce the learning rate by a default factor of 0.1 whenever the F1-score from the validation set does not improve after 2 epochs. Following Batista et al. (2004), we randomly oversample the minority classes to alleviate class imbalance among the data.

For our question generation experiments in Section 4.2, we fine-tuned "gpt-large",[10] "t5-large",[11] and "Prophetnet-En"[12] models. Prompt-tuning is enabled by concatenating question type annotations before the input sequence as shown in Figure 7. For "gpt-large" and "t5-large", we fine-tune the models using the pytorch [22] and huggingface[23] libraries, with an AdamW optimizer (Ilya and Frank, 2019) of betas (0.9, 0.999), a batch of 4, and an initial learning rate of 5e-5. This learning rate is adjusted during training with a ReduceLROnPlateau learning rate scheduler that reduces the learning rate by a factor of 0.1 whenever the loss from the validation set stops decreasing after 2 epochs. Similar to Ko et al (2020), for the "gpt-large" language model, we accumulate losses only for the question tokens by masking the context tokens before the delimiter. For ProphetNet, we use the recommended hyperparameters[12] except the learning rate, batch size,[24] input and target sequence lengths. For our models, these sequence lengths are set at 400 and 80 respectively to account for computation-related limits on our experimental machine.

For BLEU (Papineni et al., 2002b), ROGUE (Lin, 2004b), and METEOR (Banerjee and Lavie, 2005) automatic evaluation metrics, we use the implementations provided in Jury (Cavusoglu et al., 2022).[25] For BLEURT (Sellam et al., 2020), we use their recommended BLEURT-20 model.[26] Similarly, we use the recommended "microsoft/deberta-xlarge-mnli"[27] model for the BERTScore metrics.

E Details of MTurk Crowdsourced Tasks

Qualification Task: We presented a qualification test to crowdworkers on Amazon Mechanical Turk (AMT) who are interested in working on our tasks. The descriptions for each Socratic type along with examples are provided as instructions (Figure 8). As part of the test, the workers assign question type to a set of twelve questions (two for each type from Table 1). A snapshot from AMT platform for the qualification test is shown in Figure 9. We paid the workers, 0.05 per HIT for the classification task, 0.15 per HIT for the task involving question selection, 0.25 per HIT for the task involving question quality annotations on the Likert scale. These values were selected based on relevant, similar tasks on the AMT platform at the time of deployment.

The snapshots from AMT for the question selection and question quality studies described in Section 4 are shown in Figures 10 and 11.

---

[21] https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html

[22] https://pytorch.org/

[23] https://huggingface.co/

[24] Learning rate and batch size are set 5e-5 and 4 in consistent to "gpt-large" and "t5-large"

[25] https://pypi.org/project/jury/

[26] https://github.com/google-research/bleurt

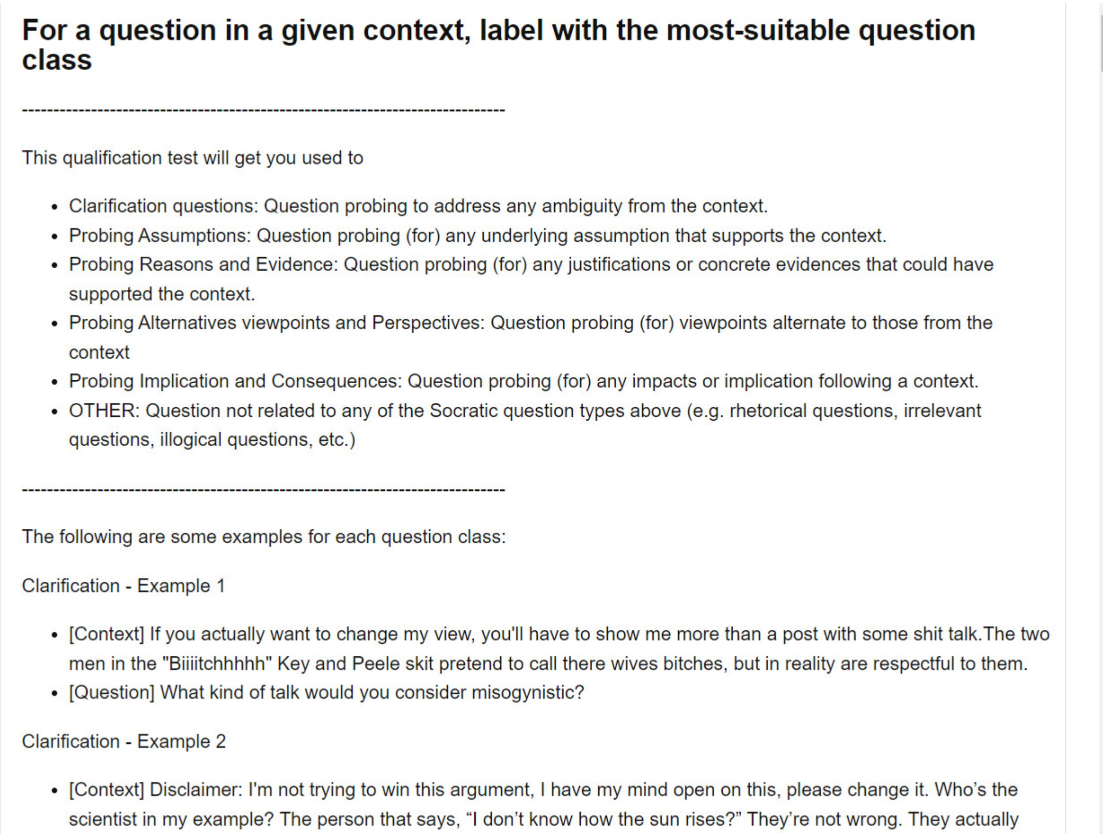
[27] https://github.com/Tiiiger/bert_score

Clarification	**Probing assumptions**	**Probing reasons and evidences**	**Probing implications and consequences**	**Probing alternative viewpoints and perspectives**	**Others**
How does that definition... (0.467)	How can one conclude… (0.439)	Why did that matter... (0.461)	Should we not care… (0.441)	Or what about drinking… (0.441)	Did you not listen... (0.431)
What do they mean… (0.457)	Is your hypothesis that… (0.385)	Was there a reason… (0.412)	Would it cause people… (0.434)	How about other physical… (0.434)	Who even cares about... (0.416)
Do you mean doctors… (0.441)	Would you have assumed… (0.376)	Can you list any… (0.408)	Would the situation change… (0.429)	What other species would… (0.424)	Do you not realise... (0.391)
How does that equal… (0.430)	What makes you presume… (0.360)	What level of evidence… (0.357)	Are you implying rich… (0.426)	Was there anything else… (0.415)	Did you not understand... (0.370)
How does this agree… (0.413)	Why does society assume… (0.338)	What long term evidence... (0.354)	What happens when inclusion … (0.413)	How would any other… (0.367)	What kind of dumbass... (0.347)

Table 11: 4-grams and their attributions (in brackets) for the different Socratic question types

CONTEXT:A child should hold no religious (or absense of religion) position. It is as unsettling as calling a child a conservative or liberal child. Children are vulnerable youths that take their parental figures' words as absolute fact. Telling them to hold political views or religious views at a young age is borderline indoctrination. **References**:1. What is the harm to the child by raising them as a liberal or a Muslim or a vegetarian or a meat eater? 2. What about teaching your child morals? **Generated Questions**: (Implications/Consequences) What do you think happens to young children who are forced to hold extremist views? (Clarification) How does teaching kids about the bible relate to holding religious views? (Reasons/Evidence) Why does it matter what political views children hold at such a young age? (Assumptions) Why are you assuming kids are even being taught their political views at all? (Alternate Viewpoints/Perspectives) Or that you would rather parents continue to influence them through their actions instead of just giving them the information?
CONTEXT:Adult children 25 or older should have to contribute and carry their weight. It would be on a case by case basis but essentially the child would have to pay a certain amount according to what they make. If you make over this much, you must pay 50% of the rent or mortgage. If you make below this much, you pay 15-25% and so on and so forth. This would encourage adults to grow up, pack up and move on from their parents homes. **References**: 1. What about parents who live at their kids home? 2. What will happen if the parents own the house free and clear? **Generated Questions**: (Implications/Consequences) Are you implying the bill could be identical between an adult and a child who are not of the same age ? (Clarification) What part of the point would you have that will encourage adults to move on? (Reasons/Evidence) And why should a person be required to move out at all? (Assumptions) Why do you assume that adults would just move on from their parents when they are adults? (Alternate Viewpoints/Perspectives) What about people who stay at home and take care of their parents and siblings for the rest of their lives?
CONTEXT:As a non-American, i find the idea of colleges having, promoting and sponsoring young (mostly female) students to dress in skimpy outfits to perform dangerous stunts and basically serve as eye-candy for the players and the audience a disturbing concept in the very least. Now don't get me wrong, the stunts they do, they teamwork they show are commendable and in no way a lesser sport in themselves. I'm sure many of the cheerleaders could have great careers in gymnastics or such organized sports. However, there is a drastic difference in how a cheerleading squad trains versus how someone trains for other sports. The cheerleaders often don't have safety equipment, practice on hard surfaces and have little to no health care. **References**:1. If the cheerleaders felt demeaned why would they choose to keep doing it, especially into college? 2. Have you consulted those cheerleaders? **Generated Questions**: (Implications/Consequences) Are you asking the college cheerleaders to do something about it? (Clarification) How do cheerleaders train on a surface that hard? (Reasons/Evidence) Do you have any evidence to support the idea that cheerleaders in colleges might be underprivileged? (Assumptions) Are we assuming cheerleaders are not working out to their maximum potential? (Alternate Viewpoints/Perspectives) What about professional gymnasts?

Table 12: Examples from *SocratiQ* are shown with their reference questions and questions generated with the ProphetNet$-p$ model



For a question in a given context, label with the most-suitable question class

----

This qualification test will get you used to

- Clarification questions: Question probing to address any ambiguity from the context.
- Probing Assumptions: Question probing (for) any underlying assumption that supports the context.
- Probing Reasons and Evidence: Question probing (for) any justifications or concrete evidences that could have supported the context.
- Probing Alternatives viewpoints and Perspectives: Question probing (for) viewpoints alternate to those from the context
- Probing Implication and Consequences: Question probing (for) any impacts or implication following a context.
- OTHER: Question not related to any of the Socratic question types above (e.g. rhetorical questions, irrelevant questions, illogical questions, etc.)

The following are some examples for each question class:

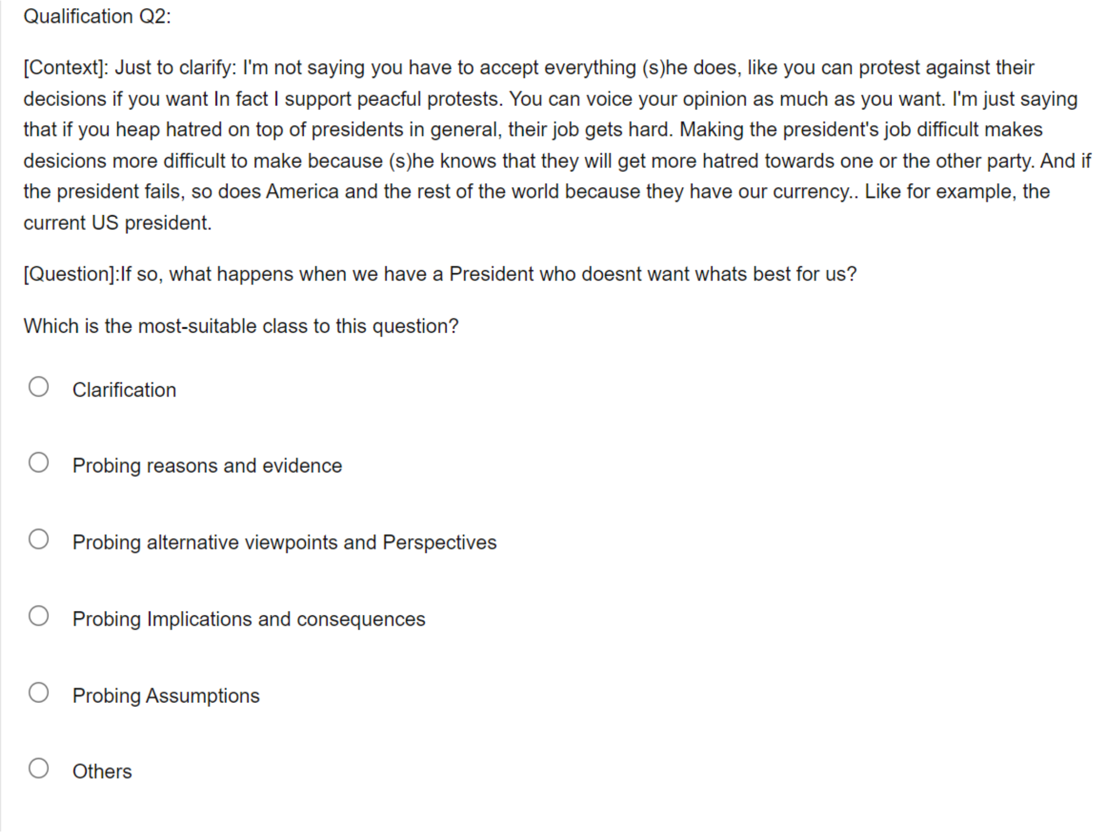
Clarification - Example 1

- [Context] If you actually want to change my view, you'll have to show me more than a post with some shit talk.The two men in the "Biiitchhhhh" Key and Peele skit pretend to call there wives bitches, but in reality are respectful to them.
- [Question] What kind of talk would you consider misogynistic?

Clarification - Example 2

- [Context] Disclaimer: I'm not trying to win this argument, I have my mind open on this, please change it. Who's the scientist in my example? The person that says, "I don't know how the sun rises?" They're not wrong. They actually

Figure 8: Workers are provided materials and time to learn the question types used in our study



Qualification Q2:

[Context]: Just to clarify: I'm not saying you have to accept everything (s)he does, like you can protest against their decisions if you want In fact I support peacful protests. You can voice your opinion as much as you want. I'm just saying that if you heap hatred on top of presidents in general, their job gets hard. Making the president's job difficult makes desicions more difficult to make because (s)he knows that they will get more hatred towards one or the other party. And if the president fails, so does America and the rest of the world because they have our currency.. Like for example, the current US president.

[Question]:If so, what happens when we have a President who doesnt want whats best for us?

Which is the most-suitable class to this question?

○ Clarification

○ Probing reasons and evidence

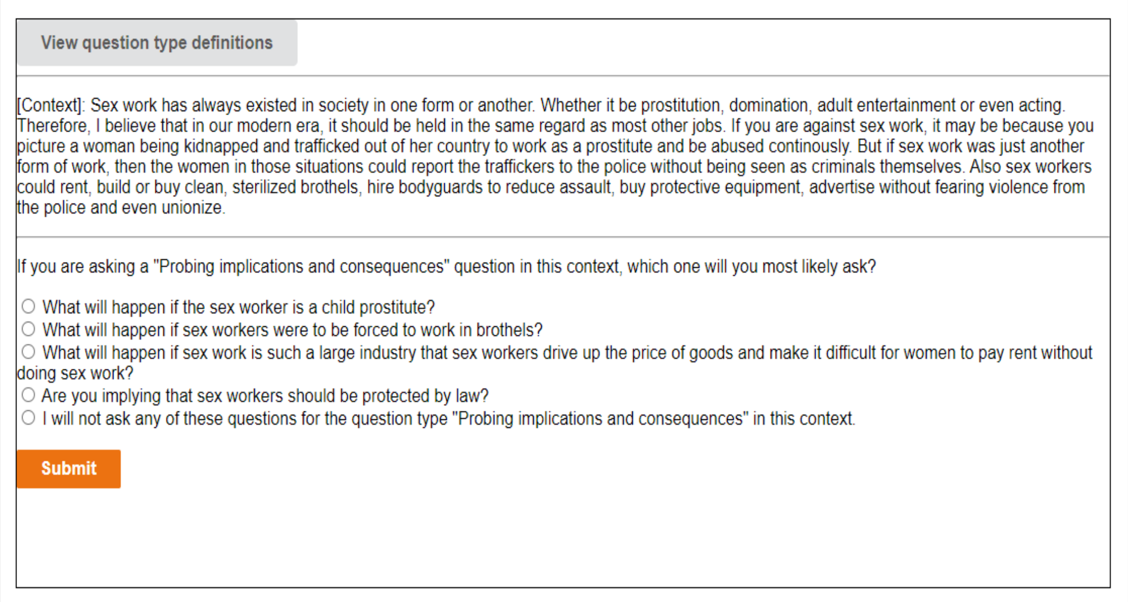
○ Probing alternative viewpoints and Perspectives

○ Probing Implications and consequences

○ Probing Assumptions

○ Others

Figure 9: Workers choose a particular question type most suited to a (question-context) pair in the qualification test (as in the main task).

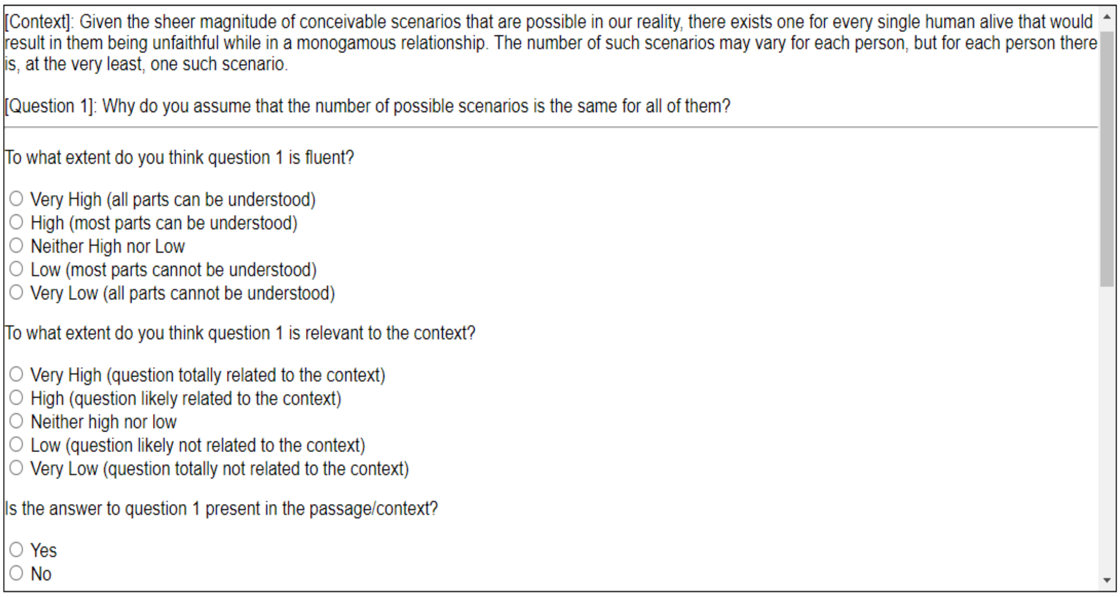


[Context]: Sex work has always existed in society in one form or another. Whether it be prostitution, domination, adult entertainment or even acting. Therefore, I believe that in our modern era, it should be held in the same regard as most other jobs. If you are against sex work, it may be because you picture a woman being kidnapped and trafficked out of her country to work as a prostitute and be abused continously. But if sex work was just another form of work, then the women in those situations could report the traffickers to the police without being seen as criminals themselves. Also sex workers could rent, build or buy clean, sterilized brothels, hire bodyguards to reduce assault, buy protective equipment, advertise without fearing violence from the police and even unionize.

If you are asking a "Probing implications and consequences" question in this context, which one will you most likely ask?

○ What will happen if the sex worker is a child prostitute?
○ What will happen if sex workers were to be forced to work in brothels?
○ What will happen if sex work is such a large industry that sex workers drive up the price of goods and make it difficult for women to pay rent without doing sex work?
○ Are you implying that sex workers should be protected by law?
○ I will not ask any of these questions for the question type "Probing implications and consequences" in this context.

Figure 10: Interface on AMT for the question selection study



[Context]: Given the sheer magnitude of conceivable scenarios that are possible in our reality, there exists one for every single human alive that would result in them being unfaithful while in a monogamous relationship. The number of such scenarios may vary for each person, but for each person there is, at the very least, one such scenario.

[Question 1]: Why do you assume that the number of possible scenarios is the same for all of them?

To what extent do you think question 1 is fluent?

○ Very High (all parts can be understood)
○ High (most parts can be understood)
○ Neither High nor Low
○ Low (most parts cannot be understood)
○ Very Low (all parts cannot be understood)

To what extent do you think question 1 is relevant to the context?

○ Very High (question totally related to the context)
○ High (question likely related to the context)
○ Neither high nor low
○ Low (question likely not related to the context)
○ Very Low (question totally not related to the context)

Is the answer to question 1 present in the passage/context?

○ Yes
○ No

Figure 11: Interface on AMT for the question quality study

Do we need Label Regularization to Fine-tune Pre-trained Language Models?

Ivan Kobyzev[1]*, **Aref Jafari**[1,2]*, **Mehdi Rezagholizadeh**[1], **Tianda Li**[1], **Alan Do-Omri**[1],
Peng Lu[1,3], **Pascal Poupart**[2], **Ali Ghodsi**[2]
[1]Huawei Noah's Ark Lab
[2] University of Waterloo, Canada
[3] Université de Montréal, Canada
{ivan.kobyzev,mehdi.rezagholizadeh}@huawei.com
{aref.jafari, ppoupart, ali.ghodsi}@uwaterloo.ca

Abstract

Knowledge Distillation (KD) is a prominent neural model compression technique that heavily relies on teacher network predictions to guide the training of a student model. Considering the ever-growing size of pre-trained language models (PLMs), KD is often adopted in many NLP tasks involving PLMs. However, it is evident that in KD, deploying the teacher network during training adds to the memory and computational requirements of training. In the computer vision literature, the necessity of the teacher network is put under scrutiny by showing that KD is a label regularization technique that can be replaced with lighter teacher-free variants such as the label-smoothing technique. However, to the best of our knowledge, this issue is not investigated in NLP. Therefore, this work concerns studying different label regularization techniques and whether we actually need them to improve the fine-tuning of smaller PLM networks on downstream tasks. In this regard, we did a comprehensive set of experiments on different PLMs such as BERT, RoBERTa, and GPT with more than 600 distinct trials and ran each configuration five times. This investigation led to a surprising observation that KD and other label regularization techniques do not play any meaningful role over regular fine-tuning when the student model is pre-trained. We further explore this phenomenon in different settings of NLP and computer vision tasks and demonstrate that pre-training itself acts as a kind of regularization, and additional label regularization is unnecessary.

1 Introduction

Nowadays, we witness the tendency of ever-growing state-of-the-art neural networks. This is especially more evident in natural language processing (NLP): the famous GPT-3 (Brown et al., 2020) has reached 175 billion parameters and a recent Chinese pre-trained language model (Zeng et al., 2021) has 200 billion parameters. It is shown that big over-parameterized neural networks not only have higher VC dimension, and hence more approximation ability (Shalev-Shwartz and Ben-David, 2014), but also their optimization regime is smoother (Safran et al., 2020). At the same time, the optimal point found for big networks has better generalization property (Brutzkus and Globerson, 2019).

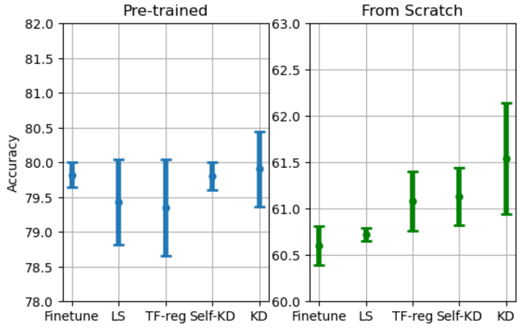


Figure 1: DistilRoBERTa results on the test set for the average of seven GLUE tasks. Graph shows the mean performance and one standard deviation interval for the pre-trained and randomly initialized models computed over five runs. For the pre-trained model all intervals intersect, hence label regularization doesn't improve the performance, but for the model trained from scratch label regularization methods outperform base training.

One can use the advantages of trained big neural networks and transfer their learned knowledge (weights and biases) to a smaller network. There are several approaches to such transferring techniques (Cheng et al., 2017), but here we will focus on Knowledge distillation (KD) (Hinton et al., 2015), a prominent neural model compression technique, which has been applied in many different forms across various domains (Gou et al., 2020) such as computer vision and NLP. To distill knowledge from a bigger model (a teacher) to a smaller

---

[*] Equal Contribution

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 166–177
May 2-6, 2023 ©2023 Association for Computational Linguistics

model (a student), KD adds an extra loss term to ensure the student predictions match with the teacher output. In the NLP domain, KD is widely adopted for compressing pre-trained language models (PLMs) (Sanh et al., 2019; Jiao et al., 2019; Jafari et al., 2021). The success of KD is attributed to different potential factors such as additional information presented by the dark knowledge (i.e. a term referring to the notion of class similarity information deriving from the teacher predictions which can not be found in the one-hot ground-truth labels) (Hinton, 2012), regularization effect of the KD loss (Yuan et al., 2020), or transferring inductive bias from one network to another (Abnar et al., 2020; Touvron et al., 2021).

Despite the widespread use of KD, it strongly depends on a trained teacher model, and calling the teacher during training adds to the computational cost of the training process noticeably. On the other hand, instead of adding the KD loss term to the student's loss, one can add a regularization term forcing the student predictions to be close to a uniform (or any arbitrary) distribution. Such a label smoothing technique results in better calibrated and more accurate classifiers (Müller et al., 2019). Recently, Yuan et al. (2020) demonstrated that label smoothing can perform as well as or even outperform KD in several computer vision tasks and across various models.

This result motivated us to investigate whether the teacher-free regularization techniques (TF) can work on par or better than KD on natural language understanding tasks. In this regard, we compare KD, label smoothing, and several other teacher-free methods for BERT and GPT type models. It is worth mentioning that our setting is different from the one of Yuan et al. (2020): 1) pre-trained language models are generally much bigger than the models from machine vision, and 2) classification tasks in our setting are mostly binary or three classes, compared to a hundred classes in CIFAR100 or two hundred in Tiny ImageNet. We ran the experiments multiple times to take into account the stochasticity of the training. Overall, we show a similar pattern: teacher-free techniques perform on par with KD methods; but we additionally observed a surprisingly different phenomenon: the gap between base fine-tuning (without KD) and fine-tuning with label regularization (KD or TF) diminished.

We explore the reasons why the base fine-tuning technique is a strong competitor of KD/TF regularization on NLU tasks. This situation is somewhat opposite to the one reported in computer vision. We hypothesized and tested the following potential explanations for our observations: 1) The small number of classes in GLUE datasets (usually 2 or 3 classes) in contrast to 10 or 100 classes in CV tasks; 2) Language models are extensively pre-trained while CV models are not. Our experiments indicate that the second hypothesis is true, whereas the number of classes doesn't play a big role in the performance gap. To the best of our knowledge, the effect of pre-training on fine-tuning with label regularization was never mentioned in the literature and deserves additional study.

Overall, our main contributions in this paper are the following:

1. Thorough comparison of TF and KD methods across both BERT and GPT models on the GLUE and other NLU benchmarks (more than 600 distinct experiments overall). We showed that, on average, KD does not significantly outperform the fine-tuning or TF techniques.

2. We studied the gap between base fine-tuning and fine-tuning with KD/TF and observed that this gap is negligible for NLU tasks. We demonstrated that this insignificant result is unlikely to be caused by the number of classes in the dataset.

3. We showed the evidence that the pre-training of neural networks reduces the performance gap on downstream tasks between the base training and training with label regularization both in NLP and computer vision domains. We supported this claim by performing Wilcoxon statistical test to demonstrate significance.

2 Background

In this section, we give a brief overview of the KD and TF techniques we will be investigating. Everywhere in the paper, we consider a classification problem with K classes. Denote by $q(x)$ the one-hot label of a data point x.

2.1 Knowledge distillation (KD)

This classical method of transferring knowledge gained traction after the paper (Hinton et al., 2015). Assume that we have a trained network (a teacher)

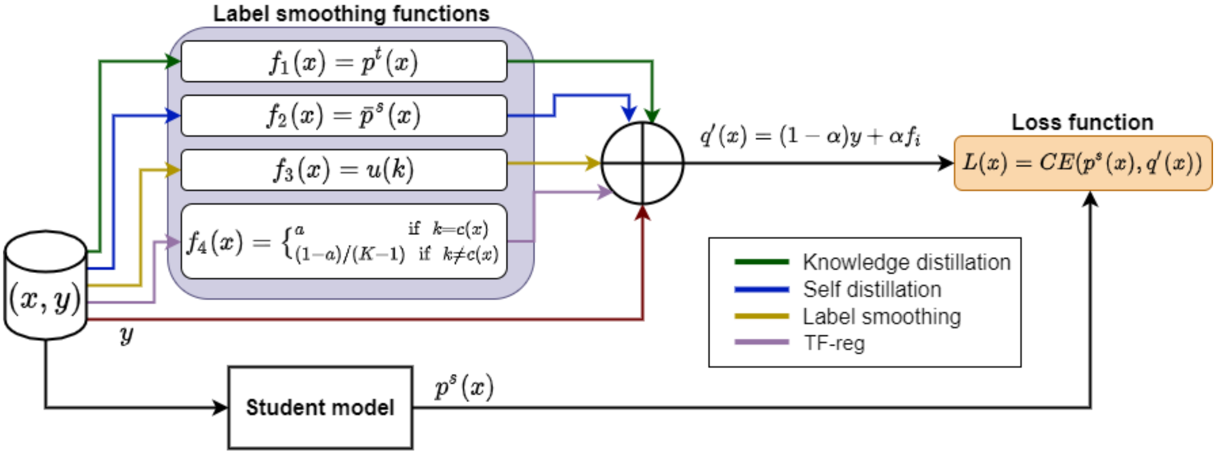


Figure 2: All the baselines of TF/KD methods we consider can be abstracted as cross-entropy training with smoothed labels. The choice of label smoothing functions determines the method. The flow in the algorithms of each method is indicated by a colored line, where $\bigoplus$ indicates the convex combination of the one-hot label and the output of the label smoothing function.

and a network we want to train (a student). Let $p^t(x)$ and $p^s(x)$ be teacher's and student's predictions respectively. One wants to transfer the knowledge from the teacher to the student. For that, one can formulate a total loss for KD as:

$$L = (1-\alpha)H(q,p) + \alpha L_{KD}, \qquad (1)$$

where $H(q,p)$ is the cross-entropy loss and $L_{KD} = D_{KL}(p^t_\tau, p^s_\tau)$ is a KL divergence between the teacher's and the student's outputs scaled with the temperature τ, i.e., $p_\tau(k) = softmax(z_k/\tau)$, where z_k is the output logits of the model. When $\tau = 1$, KD training is equivalent to cross-entropy training with the new labels "smoothed" by the teacher:

$$q'(x) = (1-\alpha)q(x) + \alpha p^t. \qquad (2)$$

2.2 Teacher-free methods

Label smoothing (LS) As Yuan et al. (2020) observed, the loss in Equation 1 is structurally similar to the label smoothing loss, where one has to replace the term L_{KD} with $L_{LS} = D_{KL}(u, p^s)$, where $u(k) = 1/K$ is the uniform distribution on K classes. Training with the label smoothing loss is equivalent to cross-entropy training with smoothed labels:

$$q'(x) = (1-\alpha)q(x) + \alpha u. \qquad (3)$$

Varying the hyperparameter α, one can change the shape of the new labels q' from smoother (higher values of α) to sharper (α closer to zero).

TF-reg (Yuan et al., 2020)) introduced a modification of LS with a sharper label-dependent smoothing distribution. More formally, for TF-reg one switches the uniform distribution u in Equation 3 to a more peaky label-dependent distribution $p^d_c(k)$, defined by:

$$p^d_c(k) = \begin{cases} a, \text{ if } k = c \\ (1-a)/(K-1), \text{ otherwise.} \end{cases} \qquad (4)$$

The smoothed label for x in TF-reg is given by:

$$q'(x) = (1-\alpha)q(x) + \alpha p^d_{c(x)}, \qquad (5)$$

where $c(x)$ is the correct label for x. Here one has two hyperparameters (a and α) instead of just one (α), which allows for better tuning, even though mathematically it is the same as LS.

Yuan et al. (2020) showed that LS and TF-reg perform on par or even outperform KD in machine vision for several models and across several datasets.

Self distillation (Self KD) Furlanello et al. (2018) and Yuan et al. (2020) considered the situation where the student and the teacher have the same architectures, and a student distills the knowledge from its fine-tuned alter-ego. In particular, first, we fine-tune a copy of the student on the dataset and then freeze it. Denote its outputs by $\bar{p}^s$. Then take the second copy and train it with the cross-entropy loss with smoothed labels:

$$q'(x) = (1-\alpha)q(x) + \alpha\bar{p}^s(x). \qquad (6)$$

The summary of all the TF and KD methods we compare can be found in Figure 2.

3 Experiments on GLUE benchmark

Inspired by the results of Yuan et al. (2020) in machine vision, we wanted to investigate the performance of TF training on NLP data. In this section, we evaluate the performance of the methods introduced in the Background section.

3.1 Dataset

We considered seven classification datasets of the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018). These datasets include linguistic acceptability (CoLA), sentiment analysis (SST-2), paraphrasing (MRPC and QQP), Natural Language Inference (MNLI, RTE) and Question Answering (QNLI). Notice that unlike most of the popular datasets in computer vision, GLUE tasks are either binary or ternary classification (only MNLI has three classes).

3.2 Experimental Setup

We explored all the KD/TF methods in three different setups to check the consistency of the results across different models. Our first student is DistilRoBERTa (Sanh et al., 2019). It has 6 layers, 768 hidden dimensions, 8 attention heads, and 82 million parameters. In the KD scenarios, we use RoBERTa-large (Liu et al., 2019) as its teacher (it has 24 layers, 1024 hidden dimensions, 16 attention heads, and 355 million parameters). In the second experiment, we use the BERT-small (Turc et al., 2019) model with 4 layers, 512 hidden dimensions, 8 heads, and 28.7 million parameters. As a teacher, we use BERT-large (Devlin et al., 2018) with 24 layers, 1024 hidden dimensions, and 336 million parameters. The third student is DistilGPT-2 with 6 layers, 768 hidden dimensions, and 82 million parameters. As its teacher, the 12-layer GPT-2 (Radford et al., 2019) model is used with the 768 hidden dimensions and 117 million parameters. For all these setups, we use the pre-trained models from Huggingface (Wolf et al., 2019). All the hyperparameters and the process of their tuning are reported in the Appendix in more detail.

Hardware Setup For our experiments, we used 8 NVIDIA TESLA V100 GPUs. Each task is trained on a single GPU.

3.3 Results

DistilRoBERTa We start with conducting the GLUE experiments over the DistilRoBERTa model. We report the results on GLUE dev and test sets in Table 1. On the dev set, we observe the following patterns: 1) The teacher-free methods (LS, TF-reg, Self-KD) outperform the Finetune baseline; 2) KD is the best technique but the standard deviation intervals intersect with the TF baselines.

Although the results of the dev set in the first experiments follow the trends of TF results in CV, examining the test results reveals some irregularities (Table 1). In particular, we observe that: 1) all the TF regularization techniques perform slightly worse than Finetune; 2) KD is on average the best technique, but it is comparable with Finetune up to one standard deviation. See Figure 1 (left) for the summary.

BERT-small In the second experiment, we evaluate the BERT-small model. Even here the story is more or less similar to our first experiment on DistilRoBERTa. Results are reported in Table 2. On the dev set, we observe that: 1) TF performs on par (up to one standard deviation) with Finetune while LS is slightly better. 2) KD is the best performer, but standard deviation intervals intersect with some of the TF baselines. On the test set, we observe that all the methods perform more or less on par up to one standard deviation while LS is slightly better.

DistilGPT-2 For DistilGPT-2, we see roughly similar patterns as in our previous experiments (Table 3). On both dev and test sets, all the methods' performance is more or less similar, with the standard deviation intervals overlapping.

The overall conclusion of our experiments is that, on average, KD or TF methods are slightly better, but the gap between the regularization techniques and the pure fine-tuning technique is not significant. Our results on the GLUE benchmark are very different from the reported results in the CV domain where pure fine-tuning without TF or KD underperforms. To explain the results, we formulate some hypotheses and scrutinize them with more experiments in the next section.

4 Analysis

In this section, we investigate potential reasons for getting the negligible difference in the relative performance of base fine-tuning, teacher-free training, and KD. We conduct some experiments to evaluate

baseline	CoLA	RTE	MRPC	SST-2	QNLI	MNLI	QQP	Score
DEV								
Teacher	68.14	81.23	91.62	96.44	94.60	90.23	91.00	87.67
Finetune	60.53 ± 0.70	68.66 ± 1.28	90.58 ± 0.69	92.43 ± 0.16	90.78 ± 0.12	84.04 ± 0.25	91.44 ± 0.03	82.64 ± 0.11
LS	60.46 ± 0.74	69.24 ± 0.90	90.87 ± 0.42	92.75 ± 0.41	90.71 ± 0.09	83.99 ± 0.13	91.41 ± 0.07	82.78 ± 0.15
TF-reg	60.74 ± 0.98	68.81 ± 0.98	90.78 ± 0.77	92.68 ± 0.15	91.13 ± 0.43	83.86 ± 0.17	91.45 ± 0.10	82.78 ± 0.29
Self-KD	60.48 ± 0.59	69.24 ± 1.20	90.97 ± 0.46	92.43 ± 0.29	90.91 ± 0.27	84.00 ± 0.19	91.62 ± 0.09	82.81 ± 0.19
KD	62.13 ± 0.67	68.66 ± 1.24	90.83 ± 0.31	92.73 ± 0.34	91.23 ± 0.29	84.34 ± 0.22	91.68 ± 0.08	83.08 ± 0.14
TEST								
Teacher	65.1	82.6	89.5	92.1	91.5	84.3	88.7	84.82
Finetune	51.62 ± 0.96	62.70 ± 0.41	88.12 ± 0.35	93.22 ± 0.46	90.66 ± 0.15	83.52 ± 0.26	88.92 ± 0.19	79.82 ± 0.18
LS	49.46 ± 3.84	62.52 ± 0.38	87.94 ± 0.51	93.42 ± 0.33	90.26 ± 0.43	83.34 ± 0.32	89.04 ± 0.08	79.43 ± 0.62
TF-reg	49.16 ± 3.82	62.92 ± 0.32	87.44 ± 0.74	93.26 ± 0.28	90.28 ± 0.21	83.36 ± 0.36	89.04 ± 0.08	79.35 ± 0.70
Self-KD	51.56 ± 1.04	62.88 ± 0.82	87.92 ± 0.43	93.10 ± 0.11	90.58 ± 0.27	83.46 ± 0.29	89.12 ± 0.10	79.80 ± 0.20
KD	50.28 ± 3.07	63.04 ± 0.43	88.80 ± 0.54	93.44 ± 0.48	90.74 ± 0.26	83.64 ± 0.21	89.42 ± 0.04	79.91 ± 0.54

Table 1: DistilRoBERTa results on the dev and test sets for the GLUE benchmark. F1 scores are reported for MRPC, Matthew's Correlation for CoLA, and accuracy scores for all other tasks. The teacher is RoBERTa-large. Averages and standard deviations are over 5 runs.

baseline	CoLA	RTE	MRPC	SST-2	QNLI	MNLI	QQP	Score
DEV								
Teacher	65.80	71.48	89.38	92.77	92.82	86.3	91.45	82.19
Finetune	41.76 ± 1.09	65.13 ± 1.22	87.09 ± 0.62	88.83 ± 0.27	86.96 ± 0.11	78.46 ± 0.13	90.02 ± 0.08	76.89 ± 0.25
LS	41.97 ± 1.63	65.85 ± 1.16	87.41 ± 0.55	88.56 ± 0.24	86.90 ± 0.16	78.51 ± 0.14	90.04 ± 0.03	77.03 ± 0.24
TF-reg	42.13 ± 0.74	64.98 ± 1.57	87.19 ± 0.43	88.58 ± 0.31	86.96 ± 0.13	78.54 ± 0.11	90.02 ± 0.04	76.91 ± 0.18
Self-KD	41.52 ± 1.74	65.63 ± 1.52	86.73 ± 0.25	88.72 ± 0.28	86.74 ± 0.58	78.63 ± 0.29	90.08 ± 0.09	76.86 ± 0.39
KD	42.48 ± 1.34	65.42 ± 0.95	88.56 ± 0.40	88.60 ± 0.60	87.31 ± 0.22	78.73 ± 0.19	90.23 ± 0.08	77.33 ± 0.15
TEST								
Teacher	63.8	69.2	85.1	89.7	89.2	83.2	86.2	80.91
Finetune	38.58 ± 0.87	62.74 ± 0.31	83.12 ± 0.42	89.56 ± 0.65	86.62 ± 0.65	78.26 ± 0.27	87.80 ± 0.17	75.24 ± 0.29
LS	40.08 ± 0.58	62.84 ± 0.22	83.24 ± 0.56	89.88 ± 0.50	86.60 ± 0.79	78.48 ± 0.26	87.78 ± 0.15	75.56 ± 0.14
TF-reg	38.92 ± 1.25	60.70 ± 3.03	82.92 ± 0.50	89.82 ± 0.42	86.22 ± 0.70	78.16 ± 0.33	87.78 ± 0.12	74.93 ± 0.63
Self-KD	38.92 ± 2.44	61.32 ± 1.26	83.12 ± 0.64	89.82 ± 0.43	86.60 ± 0.30	78.22 ± 0.53	87.76 ± 0.05	75.11 ± 0.41
KD	38.26 ± 2.20	62.32 ± 1.04	84.74 ± 0.65	89.96 ± 0.22	86.58 ± 0.27	78.34 ± 0.20	88.02 ± 0.12	75.46 ± 0.51

Table 2: BERT-small results on the dev and test sets of the GLUE benchmark. F1 scores are reported for MRPC, Matthew's Correlation for CoLA, and accuracy scores for all other tasks. The teacher is BERT-large. Averages and standard deviations are over 5 runs.

two particular hypotheses we have as potential reasons behind these inconsistencies: 1) the number of classes in the GLUE tasks is much lower than for the CV tasks; 2) NLU models are pre-trained and pre-training can attenuate the regularization impact of KD and TF methods. In the remainder of this section, we will go over some new experiments which were done to evaluate these two hypotheses respectively.

4.1 Hypothesis 1: Number of Classes

4.1.1 SST-5

SST-5 is a fine-grained sentiment classification dataset with 5 classes introduced in (Socher et al., 2013). We consider a setting of DistilRoBERTa student and RoBERTa-large (24 layers) teacher. We ran experiments for 5 seeds. The results are presented in Table 4. Overall, we can see that the standard deviations of the results are quite big, which prevents us from concluding that any technique is superior. Similar to GLUE, we note that the gap

baseline	CoLA	RTE	MRPC	SST-2	QNLI	MNLI	QQP	Score
DEV								
Teacher	43.2	66.8	87.6	92.2	88.6	82.3	89.5	78.6
Finetune	38.20 ± 1.23	64.92 ± 0.92	87.74 ± 0.34	91.54 ± 0.34	86.48 ± 0.52	79.93 ±0.08	89.70 ± 0.06	77.13 ±0.34
LS	38.24 ± 1.25	64.84 ± 0.66	87.50 ± 0.28	91.54 ± 0.25	86.56 ± 0.36	80.14 ± 0.14	89.67 ± 0.10	76.93 ± 0.27
TF-reg	38.04 ± 1.23	64.90 ± 0.62	87.58 ± 0.26	91.34 ± 0.14	86.72 ± 0.34	80.14 ± 0.22	89.64 ± 0.08	76.91 ± 0.27
Self-KD	39.41 ± 0.91	65.62 ± 1.61	87.24 ± 0.21	90.84 ± 0.31	87.04 ± 0.17	80.57 ± 0.16	89.83 ± 0.04	77.16 ± 0.33
KD	38.94 ± 1.10	66.80 ± 0.82	87.22 ± 0.74	90.86 ± 0.33	86.82 ± 0.32	80.30 ± 0.17	89.97 ± 0.24	77.34 ± 0.30
TEST								
Teacher	46.7	65.0	86.4	88.3	88.5	81.8	87.9	77.8
Finetune	31.00 ± 1.32	60.52 ± 0.66	84.52 ± 0.56	90.22 ± 1.08	85.34 ± 0.30	79.84 ± 0.22	87.68 ± 0.12	74.16 ± 0.28
LS	31.30 ± 2.00	60.18 ± 0.63	84.68 ± 0.41	91.18 ± 0.41	85.28 ± 0.30	79.78 ± 0.19	87.76 ± 0.14	74.31 ± 0.25
TF-reg	31.74 ± 2.06	60.22 ± 0.40	84.50 ± 0.46	90.62 ± 0.56	85.38 ± 0.32	79.68 ± 0.28	87.24 ± 0.50	74.20 ± 0.40
Self-KD	35.28 ± 1.55	61.02 ± 1.23	83.72 ± 0.50	90.30 ± 0.54	86.14 ± 0.33	80.12 ± 0.15	87.86 ± 0.16	74.92 ± 0.18
KD	32.96 ± 2.84	60.40 ± 0.20	84.76 ± 0.56	90.38 ± 0.53	85.82 ± 0.15	80.10 ± 0.14	88.08 ± 0.14	74.64 ± 0.43

Table 3: DistilGPT-2 results on the dev and test sets of the GLUE benchmark. F1 scores are reported for MRPC, Matthew's Correlation for CoLA, and accuracy scores for all other tasks. The teacher is GPT-2 (12 layers). Averages and standard deviations are over 5 runs.

between fine-tuning and TF/KD is not observed.

In our next experiment, we increase the number of classes even more to see if the gap appears.

baseline	Accuracy (dev)	Accuracy (test)
Teacher	56.86	59.95
Finetune	53.40 ± 0.85	54.43 ± 0.56
LS	53.50 ± 0.98	53.96 ± 0.77
TF-reg	53.62 ± 0.90	53.93 ± 0.42
KD	53.59 ± 0.26	54.14 ± 0.92

Table 4: DistilRoBERTa results on SST-5. Averages and standard deviations are over 5 runs.

4.1.2 FewRel

Han et al. (2018) introduced this dataset for relation classification. Originally, this dataset was designed for few-shot learning, so we had to slightly modify it for our purpose. First, we consider the train set of FewRel. It has 64 classes and each class has 700 instances. We shuffle the data for each class and allocated 500 instances to our train set, 100 to our dev set, and 100 to our test set. We perform the experiments five times and get a new dataset for each seed, as recommended by Bouthillier et al. (2021). The detailed procedure can be found in Appendix. Overall, we generated a text classification dataset with 64 classes.

We took DistilRoBERTa as a student and RoBERTa-base (12 layers) as a teacher. We ran experiments for 5 seeds and tuned hyperparameters for the first one (see Appendix for details). The results are in Table 5. We can observe that all the methods perform similarly up to one standard deviation and we don't see a gap between Finetune and KD/TF again.

As a conclusion of SST-5 (5 classes) and FewRel (64 classes) experiments, we do not see any evidence that the number of classes in classification tasks affects the gap.

baseline	Accuracy (dev)	Accuracy (test)
Teacher	88.93 ± 0.27	88.63 ± 0.45
Finetune	86.31 ± 0.32	86.28 ± 0.51
LS	86.35 ± 0.31	86.22 ± 0.47
TF-reg	86.35 ± 0.31	86.26 ± 0.47
KD	86.66 ± 0.36	86.41 ± 0.54

Table 5: DistilRoBERTa results on FewRel (64 classes). Averages and standard deviations are over 5 runs.

4.2 Hypothesis 2: Effect of Pre-training

We pose the following question: what is the major difference between Language Models and Computer Vision Models that might affect the performance gap between base training and label regularization? As an immediate hypothesis, we thought that extensive pre-training of the models we experimented with in the previous sections might be the reason.

baseline	From scratch	Pre-trained
Base	77.04 ± 0.26	78.17 ± 0.27
LS	78.01 ± 0.20	78.67 ± 0.20
TF-reg	78.16 ± 0.20	78.94 ± 0.28

Table 6: ResNet18 on CIFAR100. Pre-training is done on the ImageNet dataset. Averages and standard deviations are over 10 runs.

4.2.1 Computer vision

First, we did a sanity check and performed some experiments from (Yuan et al., 2020) for multiple seeds. In the paper, they didn't mention the standard deviation, but it is important for us to check if the gap we hoped to find is not a result of randomness. We considered CIFAR100 (Krizhevsky et al., 2009) and trained ResNet18 student (He et al., 2016) without label regularization and with LS and TF-reg techniques. At the same time we repeated similar experiments, but now with ResNet18 pre-trained on ImageNet dataset (Russakovsky et al., 2015). The results are reported in Table 6. Here, we can see that for the unpretrained model the standard deviation intervals between base training and label regularization don't intersect and the gap is reasonably large. However, the gap diminishes notably for the pre-trained model.

This gives us some initial evidence that our hypothesis might be true. In our next experiment, we report results that support this hypothesis.

4.2.2 NLU

GLUE experiments To investigate the effect of pre-training on the relative performance, we took a model with the same architecture as DistilRoBERTa, but instead of initializing it with pre-trained weights, we randomly initialize it (with normal distribution using the built-in Huggingface function). We used the hyperparameters from the pre-trained experiments. Then we ran experiments for 5 seeds. The results are reported in Table 7. We can see that, unlike the pre-trained model, the gap between base training and all the label regularization methods is bigger and the intersection of standard deviation intervals is much smaller or nonexistent. See Figure 1 (right) for the summary.

SST-5 experiments As a next step, we wanted to formally check the statistical significance of the findings that we reported in the previous sections. For this, we considered again the SST-5 dataset and trained both the pre-trained and randomly initialized DistilRoBERTa on it. We aim to determine whether there is a statistically significant difference between base training and TF/KD training for each of the pre-trained and randomly initialized cases. We used the (two-sided) Wilcoxon signed-rank test (Wilcoxon, 1945) over the results of eight random seeds. The Wilcoxon test is a non-parametric statistical test that checks the null hypothesis, i.e., whether two related paired samples come from the same distribution. The results are reported in Table 8. We can see that for the pre-trained model there is no statistically significant difference between base training and label regularization (p-value is greater than 0.05). However, if the model is trained from scratch, the difference becomes statistically significant.

We also tried state-of-the-art KD method, Annealing KD (AKD) (Jafari et al., 2021) which is like vanilla KD doesn't require data augmentation or an access to teacher's intermediate layers. The result of the Wilcoxon test (Table 8) shows that similarly it doesn't give a significantly better performance for a pre-trained model.

5 Related Work

Our finding that pre-training reduces or even removes the gap between base training and TF/KD training can serve as an indication of a regularization property of pre-training. Several works are exploring this in the literature.

Tu et al. (2020) studied the relation between pre-training and spurious correlations. They demonstrated that pre-trained models are more robust to spurious correlations because they can generalize from a minority of training examples that counter the spurious pattern. Furrer et al. (2020) demonstrated that Masked Language Model pre-training helps in semantic parsing scenarios to improve compositional generalization. The authors hypothesize that the primary benefit provided by MLM pre-training is the improvement of the model's ability to substitute similar words or word phrases by ensuring they are close to each other in the representation space. Turc et al. (2019) showed that pre-training is very beneficial for smaller architectures, and fine-tuning pre-trained compact models can be competitive with more elaborate methods.

6 Discussion and Future Work

We started this comparison of KD and TF regularizations on NLU tasks in the hope that a pattern

baseline	CoLA	RTE	MRPC	SST-2	QNLI	MNLI	QQP	Score
Dev								
Teacher	68.14	81.23	91.62	96.44	94.60	90.23	91.00	87.67
Base	13.3 ± 0.9	53.0 ± 0.4	81.4 ± 0.3	81.2 ± 0.5	60.8 ± 0.6	62.1 ± 1.1	80.8 ± 0.2	61.87 ± 0.21
LS	14.0 ± 0.9	53.8 ± 1.0	82.1 ± 0.7	81.7 ± 0.5	61.5 ± 0.6	62.8 ± 0.8	81.1 ± 0.5	62.45 ± 0.45
TF-reg	14.4 ± 1.2	53.0 ± 0.5	82.3± 0.4	81.8 ± 0.4	61.5 ± 1.3	62.7 ± 1.1	80.6 ± 0.2	62.35 ± 0.25
Self-KD	14.3 ± 0.5	53.0 ± 0.4	82.3 ± 0.5	81.4 ± 0.1	61.3 ± 0.4	63.5 ± 1.2	80.7 ± 0.1	62.39 ± 0.28
KD	16.6 ± 0.7	53.6± 0.9	81.8 ± 0.4	81.2± 0.5	61.4 ± 0.4	63.1 ± 0.3	81.6 ± 0.1	62.80 ± 0.45
Test								
Teacher	65.1	82.6	89.5	92.1	91.5	84.3	88.7	84.82
Base	9.8 ± 0.1	51.6 ± 1.3	79.6 ± 0.3	80.3 ± 0.0	60.7 ± 0.5	61.4 ± 0.3	80.8± 0.2	60.60 ± 0.21
LS	10.5 ± 0.2	53.0 ± 0.0	80.0 ± 0.4	80.9 ± 0.4	60.7 ± 0.7	61.7 ± 0.1	81.1 ± 0.7	60.72 ± 0.07
TF-reg	11.8 ± 1.1	51.8 ± 1.5	79.2 ± 1.5	81.2 ± 0.3	60.9 ± 0.4	61.7 ± 0.1	81.2 ± 0.4	61.08 ± 0.32
Self-KD	11.5 ± 0.3	51.0 ± 0.7	79.5 ± 1.2	80.6 ± 0.1	61.9 ± 0.4	62.3 ± 0.2	81.3 ± 0.1	61.13 ± 0.31
KD	12.8 ± 2.8	51.5 ± 1.1	79.3 ± 0.6	82.1 ± 0.4	61.2 ± 0.2	62.6 ± 0.0	81.4 ± 0.4	61.54 ± 0.60

Table 7: Randomly initialized DistilRoBERTa results on the dev and test sets for the GLUE benchmark. F1 scores are reported for MRPC, Matthew's Correlation for CoLA, and accuracy scores for all other tasks. The teacher is RoBERTa-large. Averages and standard deviations are over 5 runs.

Comparison	Pre-trained	From scratch
Base vs TF-reg	0.46	0.01
Base vs KD	0.94	0.01
Finetune vs AKD	0.74	-

Table 8: Wilcoxon signed-rank test results for DistilRoBERTa model trained on SST-5 dataset. P-values of the test are reported with p-value less than 0.05 meaning the difference is significant. The results are over the test results of 8 runs.

similar to the one in computer vision (Yuan et al., 2020) will emerge. In particular, we expected to see TF and KD perform on par while outperforming Finetune. However, it turned out that the gap, even if it exists for some seeds, is not significant.

We further scrutinized the gap between Finetune and KD/TF regularization. We hypothesized that the lack of this gap in NLU might be the result of a small number of classes in GLUE classification tasks, however, this doesn't seem to be the case: experiments on SST-5 (5 classes) and FewRel (64 classes) datasets didn't show a significant gap either. We showed that another hypothesis is likely to be true: the extensive pre-training of Language Models erases the gap. The application of a statistical test confirms that a non-negligible gap appears when models are trained from scratch. It seems that pre-training discovers a good enough initialization for fine-tuning so that even basic unregularized training can find a solution as good as training with (TF or KD) regularization. A rigorous explanation of this phenomenon is an interesting challenge for future work.

We would like to add one important remark. Our finding of this work does not suggest disregarding KD or other types of regularizations in NLP but rather using the more advanced or enhanced versions of these techniques. First of all, as shown in several works in the literature (Sanh et al., 2019; Sun et al., 2019; Turc et al., 2019; Jiao et al., 2019; Tahaei et al., 2021), KD is very important for the pre-training stage of the student models. Similarly, Gao et al. (2020) demonstrated the value of label smoothing for training machine translation models.

Moreover, improved variants of KD might still facilitate fine-tuning of pre-trained models. Even when vanilla KD doesn't give a statistically significant advantage over base fine-tuning, several works in the literature show that improved versions of KD with different auxiliary training schemes could be beneficial. For example, one can incorporate intermediate layer distillation (Sun et al., 2019; Passban et al., 2021; Wu et al., 2020, 2021), data augmentation (Rashid et al., 2021; Kamalloo et al., 2021) or contrastive training (Sun et al., 2020). Investigating better KD techniques or, more generally, better regularization methods that can improve the fine-tuning of PLMs even further will be an important direction for future work.

Limitations

In the current work we present an extensive empirical evidence that label regularization doesn't improve fine-tuning of a pre-trained model. However, we don't have any theoretical explanation of this puzzling phenomenon. Understanding the interactions of different regularization methods and how they affect the optimization is a highly non-trivial problem.

Acknowledgments

We thank Mindspore,[1] which is a new deep learning computing framework, for partial support of this work.

References

Samira Abnar, Mostafa Dehghani, and Willem Zuidema. 2020. Transferring inductive biases through knowledge distillation. *arXiv preprint arXiv:2006.00555*.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Naz Sepah, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Dmitriy Serdyuk, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. 2021. Accounting for variance in machine learning benchmarks. *arXiv preprint arXiv:2103.03098*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Alon Brutzkus and A. Globerson. 2019. Why do larger models generalize better? a theoretical perspective via the xor problem. In *ICML*.

Yu Cheng, D. Wang, Pan Zhou, and Tao Zhang. 2017. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. *arXiv preprint arXiv:1805.04770*.

Daniel Furrer, Marc van Zee, Nathan Scales, and Nathanael Schärli. 2020. Compositional generalization in semantic parsing: Pre-training vs. specialized architectures. *arXiv preprint arXiv:2007.08970*.

Yingbo Gao, Weiyue Wang, Christian Herold, Zijian Yang, and Hermann Ney. 2020. Towards a better understanding of label smoothing in neural machine translation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 212–223.

Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. 2020. Knowledge distillation: A survey. *arXiv preprint arXiv:2006.05525*.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Geoffrey Hinton. 2012. Neural networks for machine learning, coursera. *URL: http://coursera.org/course/neuralnets*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. *arXiv preprint arXiv:2104.07163*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via minimax. *arXiv preprint arXiv:2105.13608*.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. 2018. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

---

[1] `https://www.mindspore.cn/`

Rafael Müller, Simon Kornblith, and Geoffrey Hinton. 2019. When does label smoothing help? In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.*

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. Alp-kd: Attention-based layer projection for knowledge distillation. In *AAAI*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. Mate-kd: Masked adversarial text, a companion to knowledge distillation. *arXiv preprint arXiv:2105.05912*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Itay Safran, Gilad Yehudai, and Ohad Shamir. 2020. The effects of mild over-parameterization on the optimization landscape of shallow relu neural networks. *arXiv preprint arXiv:2006.01005*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Siqi Sun, Zhe Gan, Yu Cheng, Yuwei Fang, Shuohang Wang, and Jingjing Liu. 2020. Contrastive distillation on intermediate representations for language model compression. *arXiv preprint arXiv:2009.14167*.

Marzieh S Tahaei, Ella Charlaix, Vahid Partovi Nia, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021. Kroneckerbert: Learning kronecker decomposition for pre-trained language models via knowledge distillation. *arXiv preprint arXiv:2109.06243*.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *arXiv preprint arXiv:2007.06778*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizade, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. *arXiv preprint arXiv:2010.03034*.

Yimeng Wu, Mehdi Rezagholizadeh, Abbas Ghaddar, Md Akmal Haidar, and Ali Ghodsi. 2021. Universal-kd: Attention-based output-grounded intermediate layer knowledge distillation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7649–7661.

Li Yuan, Francis E. H. Tay, Guilin Li, Tao Wang, and Jiashi Feng. 2020. Revisiting knowledge distillation via label smoothing regularization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3902–3910.

Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, Chen Li, Ziyan Gong, Yifan Yao, Xinjing Huang, Jun Wang, Jianfeng Yu, Qi Guo, Yue Yu, Yan Zhang, Jin Wang, Hengtao Tao, Dasen Yan, Zexuan Yi, Fang Peng, Fangqing Jiang, Han Zhang, Lingfeng Deng, Yehong Zhang, Zhe Lin, Chao Zhang, Shaojie Zhang, Mingyue Guo, Shanzhi Gu, Gaojun Fan, Yaowei Wang, Xuefeng Jin, Qun Liu, and Yonghong Tian. 2021. Pangu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation. *arXiv preprint arXiv:2104.12369*.

A Hyperparameters for GLUE and SST-5 experiments

We ran experiments for seeds 42, 549, 1237, 230 and 805. For all baselines we run experiments for 30 epoch. All hyperparameters for pre-trained and randomly initialized models are listed in the Tables 9, 10.

When we do 8 seeds for the statistical test, we add seeds 4653, 5589 and 992.

Hyperparameters for Annealing KD on SST-5 are listed in Table 11

B Experiments on FewRel

B.1 How we constructed the dataset

For each seed (42, 549, 1237, 230 and 805) separately we constructed a new dataset.

Train set of FewRel has 64 classes, each class has 700 instances. We shuffle the data for each class (with a current seed) and allocated first 500 instances for our train set, second 100 for our dev set and last 100 for our test set.

We concatenated the context, head, and tail of the relation into one piece of text to be classified.

B.2 Hyperparameters

All hyperparameters are shown in Table 12

C Experiments on CIFAR100

We follow the experimental setup (Yuan et al., 2020). For optimization we used SGD with a momentum of 0.9. The learning rate starts at 0.1 and is then divided by 5 at epochs 60, 120 and 160. All experiments are repeated 10 times with different random initialization. The seeds we used: 11, 125, 1350, 23, 230, 4653, 5589, 56, 6, and 992. The validation set is made up of 10% of the training data. For experiments with pre-trained models, we use the checkpoints available at [2].

D Hyper-parameter tuning

For hyper-parameter tuning, we use the ray tune library (Liaw et al., 2018). The tuned hyper-parameters are batch size, learning rate, α, and temperature where they have been selected among {8, 16, 32, 64}, {9e-6, 1e-5, 2e-5, 3e-5}, {0.4, 0.5, 0.7, 0.8, 0.9, 0.95}, and {1, 2, 5, 10} sets respectively. We use ASHAScheduler algorithm of ray tune to find the best hyper-parameters. The metric of choosing them was maximum performance on dev set. The sample size of tuning hyper-parameters was 20 and 1 GPU was used for each experiment. Also, the maximum number of epochs for each trial was 20 epochs.

Since tuning hyper-parameters requires a huge amount of computational resources, we tried hyper-parameter tuning for vanilla KD on DistillRoBERTa model for GLUE benchmark. Then we chose the five best hyper-parameter sets from this experiment and checked their performance with other baselines and chose the one with highest average performance among all baselines. Only different α parameters for random initialization and pre-trained experiments on GLUE tasks and SST-5 made considerable differences in results. Therefore we used different α values in these experiments.

Tuning hyper-parameters individually for each baseline would be a better option, but this would require a very large amount of computational resources. However, after careful hyper-parameter tuning for vanilla KD and less intensive hyper-parameter tuning for teacher-free baselines, the latter show very close performance to vanilla KD and this fact supports the main message of our paper.

---

[2] https://pytorch.org/vision/stable/models.html

Hyper-parameter	Value
Learning rate	$2 \cdot 10^{-5}$
Batch Size	32
Temperature	1
Training epoch	30
α for LS	0.1
α for KD, Self-KD and TF-reg	0.5
α for KD and TF-reg RI SST-5	0.9
a for TF-reg	0.95

Table 9: Hyperparameters for DistilRoBERTa and BERT-Small models on GLUE and SST-5 for pre-trained and randomly initialized (RI) models

Hyper-parameter	CoLA	RTE	MRPC	SST-2	QNLI	QQP	MNLI
Learning rate	10^{-5}	$2 \cdot 10^{-5}$	10^{-5}	$2 \cdot 10^{-5}$	10^{-5}	$2 \cdot 10^{-5}$	$2 \cdot 10^{-5}$
Batch Size	16	16	16	16	16	16	16
Temperature	1	1	1	1	1	1	1
Training epoch	30	30	30	30	30	30	30
α for LS	0.1	0.1	0.1	0.1	0.1	0.1	0.1
α for KD	0.5	0.5	0.5	0.5	0.5	0.5	0.5
α for Self-KD	0.5	0.5	0.5	0.5	0.5	0.5	0.5
α for TF-reg	0.5	0.5	0.5	0.5	0.5	0.5	0.5
a for TF-reg	0.95	0.95	0.95	0.95	0.95	0.95	0.95

Table 10: Hyperparameters for DistilGPT-2 on GLUE tasks for Finetune and regular KD and TF

Hyper-parameter	Value
Learning rate	$2 \cdot 10^{-5}$
Batch Size	8
Max Temperature	10
Training epochs Phase I	20
Training epochs Phase II	10

Table 11: Hyperparameters for Annealing KD for DistilRoBERTa on SST-5

Hyper-parameter	Value
Learning rate	$1.5 \cdot 10^{-5}$
Batch Size	32
Temperature	1
Training epoch	30
α for LS	0.1
α for KD	0.5
α for TF-reg	0.5
a for TF-reg	0.95

Table 12: Hyperparameters for DistilRoBERTa on FewRel for Finetune, KD and TF

Hyper-parameter	Value
Learning rate	0.1
Batch size	128
Weight decay	$5 \cdot 10^{-4}$
Training epoch	200
α for LS	0.1
α for TF-reg	0.1
Temperature for TF-reg	20

Table 13: Hyper-parameters for ResNet18 on CIFAR100.

COVID-VTS: Fact Extraction and Verification on Short Video Platforms

Fuxiao Liu **Yaser Yacoob** **Abhinav Shrivastava**

University of Maryland, College Park
{fl3es, yaser, abhinav}@umd.edu

Abstract

We introduce a new benchmark, `COVID-VTS`, for fact-checking multi-modal information involving short-duration videos with COVID19-focused information from both the real world and machine generation. We propose, `TwtrDetective`, an effective model incorporating cross-media consistency checking to detect token-level malicious tampering in different modalities, and generate explanations. Due to the scarcity of training data, we also develop an efficient and scalable approach to automatically generate misleading video posts by event manipulation or adversarial matching. We investigate several state-of-the-art models and demonstrate the superiority of `TwtrDetective`.

1 Introduction

The proliferation of misinformation in social media poses a serious threat to our society, especially during the COVID-19 pandemic. Therefore, it is necessary to develop automatic fact-checking tools to verify the claims propagated online.

Recent fact-checking work (Thorne et al., 2018; Wadden et al., 2020; Augenstein et al., 2019; Fung et al., 2021; Bekoulis et al., 2021) investigated automatic misinformation detection by developing various benchmark datasets as well as state-of-the-art neural network architectures involving sources such as Wikipedia pages, tables, news articles, and scientific articles. However, due to the lack of publicly available benchmarks, fact checking is still challenging on short video platforms, such as TikTok, Twitter, and Instagram.

Verifying the factual correctness of claims in short video platforms poses new challenges. Firstly, multi-modal misinformation that leads to short video posts is more misleading than using just textual content (Micallef et al., 2022), since the claims usually only tamper with subtle elements of the factual information from the source video. Furthermore, short videos (Shang et al., 2021) containing diverse scene shifts, human activities, and cross-modal information, greatly increase the complexity and ambiguity of the video content. Moreover, current methods explored the authenticity of GAN-generated video with unimodal analysis (Güera and Delp, 2018). Nevertheless, they cannot be directly applied to the short video platform, where the video content is often intentionally manipulated.

To tackle these challenges, we introduce `COVID-VTS`, a new benchmark dataset with trustworthy claims and corresponding good-quality videos, collected from Twitter video posts. `COVID-VTS` contains different inconsistent taxonomies and the inconsistency comes from different modalities. Examples are shown in Figure 1. We additionally introduce a novel approach to generate fake video posts automatically by manipulating event elements or adversarial matching (Luo et al., 2021a). The advantage of our event manipulation tool is that it's able to control the polarization of semantics and track the manipulation object by editing a small component of the factual information. We apply quality control to delete unqualified generations and address the linguistic bias.

We propose `TwtrDetective`, a multimodal fact-checking model, where the input consists of a claim with the paired video, and the goal is to predict the consistency. `TwtrDetective` takes advantage of the *Event Alert* module to precisely extract fine-grained factual details from the claim, as well as *Pairwise Consistency Aggregation* module to effectively measure the consistency between each modality. `TwtrDetective` is also able to point out the inconsistent modality (e.g., see Figure 3).

Experimental results show that `TwtrDetective` achieves higher detection accuracy over baselines on two datasets. Our main contributions are summarized as:

- We introduce `COVID-VTS`, a fact-checking dataset for short video platforms, consisting of 10k video text pairs with diverse scenes, more accessible

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 178–188
May 2-6, 2023

(Speech)
new cases reported in Italy South Korea
...
the officials struggled to contain coronavirus outside China
...
as the death toll has twenty six hundred is it too early to call the crime virus
...

(Claim)
Officials are scrambling to contain the spread of ebola outside of China, joined @cherylcasone with more coronavirus

(Speech)
...
slovakia's government has now approved a two week national lockdown
...
start of the pandemic with the emphasis on pushing vaccination programs it should be has tightened the screws on people ...

(Claim)
slovakia to prevent new lockdown while italy toughens up on the unvaccinated

(Speech)
...
grounded BA flight likely to lead to a profit drop
...
expected to forecast profits and this year that's not currently possible because the duration potentially affected coronavirus prices remain unclear
...

(Claim)
#BritishAirways is expecting a drop in profits this year - due to cancelled flights #coronavirus

Figure 1: Examples of different inconsistent taxonomies from `COVID-VTS`, which are generated by our automatic manipulation tool. The red box indicates the inconsistent modality. Our task is to independently decide whether the video-text pair is consistent and point out which modality is fake. (1) The event argument in the claim is modified; (2) The event trigger in the claim is tampered; (3) The short video is curated by adversarial matching.

modalities, and trustworthy claims from both the real world and machine generation.

- We propose an effective approach to automatically generate large-scale verifiable, trustworthy as well as misleading video posts rather than employing human annotators.
- We propose `TwtrDetective`, a new explainable fact-checking framework for the short video platform, showing superior results on two challenging datasets with respect to baselines.

Our code is publicly available at `https://github.com/FuxiaoLiu/Twitter-Video-dataset`.

2 Related Work

The spread of misinformation has led to a growing body of research in automatic fact-checking. Many large scale datasets collected from Wikipedia and fact-checking websites were introduced, including FEVER (Thorne et al., 2018), SciFact (Wadden et al., 2020), UKP Snopes (Hanselowski et al., 2019), MultiFC (Augenstein et al., 2019). However, the fake claims in these datasets are manually generated by humans, making it expensive to deploy at scale. Recently, some synthetic datasets were proposed to address this limitation. (Jiang et al., 2020) generated complex fake claims using word substitutions. (Saakyan et al., 2021) took advantage of the token-infilling ability from Masked Language Model to replace the salient tokens. Also, (Fung et al., 2021) formulated a novel data synthesis method by manipulating knowledge elements within the multi-modal knowledge graph. In order to alleviate the linguistics bias within the machine-generated claims, (Luo et al., 2021a) constructed the out-of-context captions by retrieving the real-world sentence with the similar semantics. In a similar vein our dataset, (Liu et al., 2020b) constructed VIOLIN for the Video-Language Inference while all the statements are written by experts. (Wang et al., 2022) collected social media video posts from Twitter but the fake claims are constructed by random swap. In comparison, `COVID-VTS` is the first COVID-19 fact-checking dataset for short video platforms, containing rich information with diverse scenes, more accessible modalities as well as misleading claims from both the real world and machine generation.

Traditional fact-checking approaches (Zellers et al., 2019; Schuster et al., 2020; Atanasova et al., 2019) are mainly based on text. They fall short if the evidence stems from visual information. Recent multi-modal models (Fung et al., 2021; Tan et al., 2020; Shang et al., 2021) are equipped with the ability to check consistency according to the information conveyed across modalities. In contrast, we propose a fact-checking model for the short video platforms with multimedia explanations that achieves higher accuracy to detect misinformation.

3 COVID-VTS Dataset Construction

`COVID-VTS` contains 10k well-formed claims with the paired videos to support or refute the claims. In this section, we first describe the procedure to select the trustworthy and consistent video/claims pairs from Twitter video posts. Second, we also present our approach to automatically construct well-formed inconsistent video-claim pairs.

3.1 Filtering for Authentic and Consistent Video Posts

To assemble the `COVID-VTS` dataset, we used the Twitter scraper to collect over 100k English video posts using COVID related keywords and hashtags ranging from the end 2019.10 to 2022.8. Examples are shown in Table 1. We retain one post for a video link to reduce the potential bias. To retrieve the speech from videos, we leverage the Speech-to-Text tool from IBM Cloud (Pitrelli et al., 2006). In addition, we leverage SimpleOCR (Ko and Kim, 2004) to recognize text on screen. After the preliminary filtering, we have several steps to select the authentic and consistent video posts:

VERIFIED User Account. According to the requirements from Twitter, only the authentic, notable, and active accounts will receive the blue verified badge (including official government, official company, etc.). Accounts that routinely post content that harasses, shames, or engaged in severe violations of our platform manipulation and spam policy are ineligible for blue badge. We collect the authentic posts from the verified accounts into `COVID-VTS` to improve the data quality.

Claims Must be VERIFIABLE. Event structures are essential to reveal the factual information of a sentence, since the overall semantics might be opposite if event elements are changed. Moreover, claims in `COVID-VTS` are supposed to be verifiable propositions whose truthfulness is determined by multi-modal evidences from the paired videos. Without the event structures, its truthfulness isn't verifiable. Thus, we delete instances with personal opinions or emotions. For example, *'So proud of my boys! GetVaccinated'*, which has no factual information to be verified.

In practice, we first remove the claims without event structures, by using DYGIE++ (Wadden et al., 2019), a state-of-art event extraction framework, pretrained on MECHANIC dataset. The event structures generated by DYGIE++ include event triggers and event arguments. For example in *'Officials are scrambling to contain outbreaks of the coronavirus outside of China'*, *'contain'* is the trigger which express the occurrence of the event, *'officials'* and *'outbreaks of the coronavirus'* are event arguments which play different roles in this event.

After selecting these verifiable claims, we manually check whether they are consistent with the associated videos. We also select verifiable and consistent video text pairs from the unverified accounts in order to increase the size of our dataset. Our hypothesis is that the video posts are trustworthy after the "cleaning" steps. Given this hypothesis, the main task is to analyze the inconsistency between different modalities.

Keywords/Hashtags
covid, #covid, #covid19, corona, #coronavirus, maskup, pandemic, ICU, vaccine, #vaccine, coronavirus, quarantine, #quarantine, moderna, #covidvariant, omicron, booster, mRNA, phizer, #phizer, #who, #workfromhome, #vaccinepassports, #travelbans, #social_distance, n95

Table 1: Summary of the covid related keywords and hashtags in the data collection process.

	Most frequent Tokens
Event Trigger (Original Claims)	*fight, infect, protect, prevent, help, stop, quarantine, contain, confirm, threat, plunge, mandate, deal, cause*
Event Argument (Original Claims)	*coronavirus, omicron, pfizer, covid, delta, moderna, booster, mask, lockdown, protest, social distance, ban, vaccine*
Event Trigger (Generated Claims)	*produce, increase, create, develop, remove, protect, avoid, identify, support, generate, rule, allow, establish*
Event Argument (Generated Claims)	*omicron, variant, death, cancer, hospital, WHO, community, medical service, drug, viruses, migration, media, ICU*

Table 2: Most frequent event triggers and arguments in original claims and generated claims.

3.2 Automatic INCONSISTENT Video-Claim Pairs Generation

In this section, we describe the steps to automatically generate inconsistent video-claim pairs. Our dataset contains the following three inconsistent taxonomies: (1) Real video, Real speech and Inconsistent claim; (2) Real video, Inconsistent speech and real claim; (3) Inconsistent video, Real speech and Real claim.

Inconsistent Claim Generation. Inspired by (Liu et al., 2020b), we automatically generate fake-claims by controlling the polarization of semantics and tracking the manipulation object by modifying a small portion of the factual information in true claims. In this case, most of the statement remains true to the video content. This strategy can also alleviate the linguistic bias, which was introduced by (Fung et al., 2021; Zellers et al., 2019). In their datasets, the model correctly detects the fake claims without comparing different modalities. This is because all the fake claims are generated by language models. Another advantage is that we can generate the large-scale training set automatically rather than employing human annotators.

Original TRUE Claims	Generated FAKE Claims
Officials are scrambling to contain outbreaks of the coronavirus outside of China.	*Officials are scrambling to contain the spread of ebola outside of China.*
Moderna chairman getting vaccinated booster shots is the only way to stop the virus.	*Moderna chairman getting infected is the only way to stop the virus.*
Fed chair powell warns omicron variant could dent economic recovery.	*Fed chair powell warns omicron variant could facilitate economic recovery.*
Australians caught up in china's crisis have finally returned home after 14 days quarantined on christmas island.	*Australians caught up in china's crisis have lost home after 14 days quarantined on christmas island.*

Table 3: A detailed look into the examples generated by our efficient automatic approach. The first two examples are the event-argument manipulation and the final two are the event-trigger manipulation.

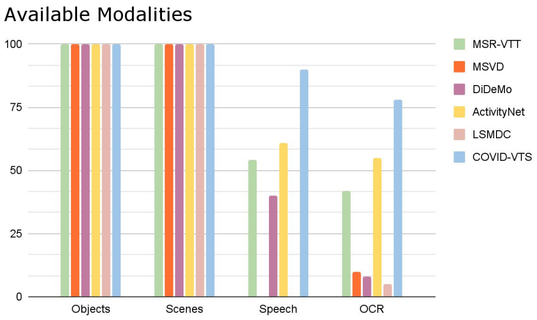


Figure 2: The histogram indicates COVID-VTS is more accessible to speech content and text on the screen than previous datasets.

	COVID-VTS	VIOLIN
Average Caption Length	19.2	18.0
Average Speech Length	69.2	76.4
Average Video Length	26.5s	35.2s
Named Entities (Sentence)	90.8%	10.3%
Source	Twitter	TV show

Table 4: Summary of `COVID-VTS` dataset.

Only event triggers or event arguments will be selected as the manipulated tokens `[mask]` depending on different intentions. We follow the procedures from (Nguyen et al., 2020) and use BERTWeet, pretrained on the Tweets related to the COVID-19 pandemic, to predict the domain-aware alternates of the event elements which can be subject to masking. In order not to introduce additional noise, we delete the candidates if their substituted tokens are not in the vocabulary of the original dataset to reduce the potential bias. Then, we feed the candidates and their corresponding true claims as input to the CrossEncoder model trained on Mutli-NLI dataset (Williams et al., 2017) and select the candidates with the CONTRADICTION label. If the candidate pair has the highest contradiction score, it will be assigned as the fake claim. Table 3 presents examples generated by our efficient automatic approach. The first two examples are the event-argument manipulation and the final two are the event-trigger manipulation.

Table 2 shows the most frequent event elements in original claims and generated claims. In practice, we found that these frequent alternatives bring the additional linguistic bias. Models can learn to classify the claims as fake ones simply by detecting these frequent tokens without absorbing information from other modalities. Therefore, we alleviate this bias by only keeping one claim for each alternative. Then replace unqualified claims by selecting the most similar claims from the dataset. Inconsistent pairs constructed this way focus more on the global understanding of the pairs. This rigorous setting makes the claims more challenging to distinguish by the analysis model, and in-depth reasoning is required to identify the fake content.

Inconsistent Speech Generation. Unlike (Tan et al., 2020) and (Fung et al., 2021) editing multiple parts of the article, our target is to only modify the evidence sentences from the speech to reduce the linguistic bias. First, we use cosine similarity on SBERT sentence embeddings to extract the most similar sentences to the real claims. After that, we manipulate the named entities or event elements within the evidence sentences.

Inconsistent Video Generation. As for the third type of inconsistent pairs, we keep the speech and claims as the original ones. However, we replace the original videos with another real video that is similar to the current one by using the adversarial matching (Liu et al., 2020b) method. Specifically, we utilize pretrained VideoCLIP (Xu et al., 2021) to generate video representations to calculate the cosine similarity.

Missing Modalities. In order to handle the video-claim pairs missing the speech text, we select the real speech text from the dataset which has common entities with respect to the current claim. This

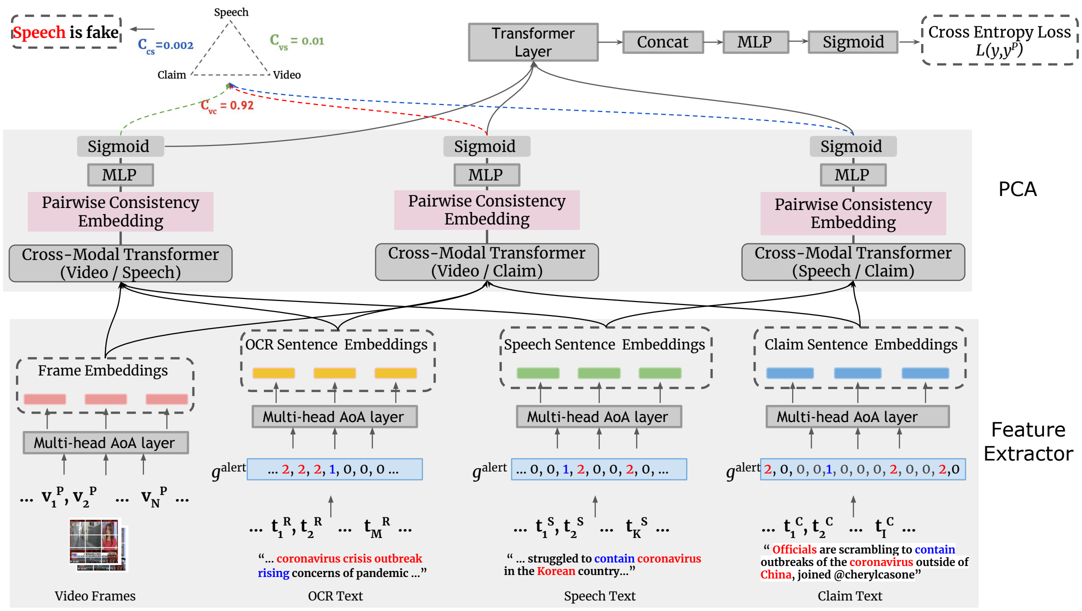


Figure 3: Overview of `TwtrDetective`. It detects the cross-modal inconsistency by comparing with video appearance, speech content, screen text and claims. g^{alert} means the event alert module. In this example, `TwtrDetective` predicts it as a an inconsistent pair and points out speech is manipulated since c^{vs} and c^{cs} are smaller.

new tuple with the real video, fake speech and real claim will be regarded as the inconsistent pairs in `COVID-VTS`. Examples generated by our tool are shown in Figure 1 and Figure 4. Without the in-depth reasoning and cross-modal understanding, it's challenging to distinguish them from the real pairs.

After all the filtering steps of automatic quality and bias control with manual validation, we assign the same amount of manipulated samples with pristine ones. Thus, the resulted dataset consists of 10k well-formed claims with associated videos.

3.3 Dataset Analysis

`COVID-VTS` exhibits three important differences to current benchmark datasets for video-text tasks (Xu et al., 2016; Chen and Dolan, 2011; Rohrbach et al., 2015; Anne Hendricks et al., 2017; Krishna et al., 2017). First, `COVID-VTS` brings new challenges with more diverse and complex scenes, including indoor press conference, news broadcasting, outdoor activities like protests, interviews, and screen recordings as well a slide shows. In contrast to recent video-text datasets (Xu et al., 2016; Chen and Dolan, 2011; Rohrbach et al., 2015; Anne Hendricks et al., 2017; Krishna et al., 2017), `COVID-VTS` has more available speech and screen text in Figure 2. Specifically, 87.5% of the videos have speech and over 77% of the videos have the screen text. `COVID-VTS` has more named entities and videos (Table 4). 90.8% of the sentences in `COVID-VTS` have named entities while VIOLIN (Liu et al., 2020b) is 10.2%. This large gap indicates `COVID-VTS` not only is an excellent resource to support research in the alignment between videos and named entities, but also provides new challenges to existing video-language inference models. Finally, `COVID-VTS` is the first fact-checking benchmark on the short video platform with inconsistent information from different modalities, which makes the fact-checking task more challenging.

4 Fact-Checking Model

4.1 Overview

As shown in Figure 3, our fine-grained fact-checking system, `TwtrDetective`, is able to evaluate the overall factual consistency by integrating pairwise relation embeddings between different modalities. `TwtrDetective` also points out which modality is inconsistent and provides explanations to support the verification.

Feature Extractor. We use the vision transformer of CLIP (ViT-B/32) (Radford et al., 2021) to encode every frame into features. In particular, it extracts N non-overlapping image patches from the frame and perform linear projection to map every patch into 1D token $\{v_1^p, \dots, v_N^p\}$, where $v_i \in \mathbb{R}^D$, where $D = 512$, N is the number of patches for each frame. Second, we apply pretrained RoBERTa (Liu et al., 2019b) to generate contextual text representations, which utilizes byte pair encoding (Shibata et al., 1999) to tokenize the sentences. Therefore, each sentence in the speech content S is represented as a sequence of tokens $\{t^s_1, \dots, t^s_K\}$, where $t^s_i \in \mathbb{R}^D$ and $D = 768$ and screen text O are $\{t^r_1, \dots, t^r_M\}$, where $t^r_i \in \mathbb{R}^D$ and $D = 768$. Finally tokens in the claim C are defined as $\{t^c_1, \dots, t^c_I\}$, where $t^c_i \in \mathbb{R}^D$ and $D = 768$.

Event Alert Module The intuitive method to check the consistency is to calculate the video-text similarity with video-sentence retrieval models (Liu et al., 2019a; Gabeur et al., 2020). However, the performance is dismal if the true information is tampered by small fragments. This is because existing models mainly encode the text at the document level without giving sufficient signal to focus on the factual elements, namely the event trigger and event argument.

To provide explicit guidance to learn the internal event semantic of the text, we first employ DYGIE++ (Hope et al., 2020) to detect the event structures within the text, assigning 1 if the token is the event trigger, 2 if it's the event argument and 0 otherwise. The indices are then fed into the learnable embedding table to generate *Event Alert* gate g^{alert}, where $g^{alert} \in \mathbb{R}^D$ and $D = 512$. A key property of g^{alert} is that it helps our model determine the salience of tokens in the text. Then, the claim token t^c_i, speech token t^s_i, OCR token t^r_i are projected into a common semantic subspace with the same dimension by learning parameters W^c, W^s, W^r.

$$t^{c\prime}_i = g^{alert} \odot \tanh(W^c t^c_i) \tag{1}$$

$$t^{s\prime}_i = g^{alert} \odot \tanh(W^s t^s_i) \tag{2}$$

$$t^{r\prime}_i = g^{alert} \odot \tanh(W^r t^r_i) \tag{3}$$

where $\odot$ represents the element-wise multiplication operation and $tanh$ is the activation function.

Multi-head AoA Layer. Motivated by the architecture presented in (Liu et al., 2020a), we contextualises $t^{c\prime}_i$, $t^{s\prime}_i$, $t^{r\prime}_i$ by using stacked Multi-Head Attention on Attention Layer, which takes advantage of the "Attention on Attention" module (Huang et al., 2019) to facilitate the generation of attended information. After encoding the text, the output of [CLS] tokens named as s^c, s^s and s^r are utilized as the sentence representations of claim, speech content and screen text correspondingly.

$$s^c = \text{MHAoA}^{\text{Mask}}(\{t^{c\prime}_i, \dots, t^{c\prime}_I\})_{[cls]} \tag{4}$$

$$s^s = \text{MHAoA}^{\text{Mask}}(\{t^{s\prime}_i, \dots, t^{s\prime}_K\})_{[cls]} \tag{5}$$

$$s^r = \text{MHAoA}^{\text{Mask}}(\{t^{r\prime}_i, \dots, t^{r\prime}_M\})_{[cls]} \tag{6}$$

Patch features are also projected into the common subspace with the text by W^v. In order to learn the salience of patches in each frame, we feed the patch features with injection of positional information into Multi-Head AoA Layer to model the correlation of each patch. After that, we leverage the global average pooling to output the representation of each frame v^f.

$$v'_i = \tanh(W^v v_i) \tag{7}$$

$$v^f = Pool(\text{MHAoA}^{\text{Mask}}(\{v'_1, \dots, v'_N\})) \tag{8}$$

Pairwise Consistency Aggregation (PCA). To model the consistency between the claim, speech and video frames, we apply the Cross-modal Transformer (Li et al., 2020) learn the pairwise relationship. First, we fed the speech sentence embeddings $S = \{s^s_1 \dots, s^s_{L_s}\}$, visual frame embeddings $F = \{v^f_1 \dots, v^f_{L_f}\}$ and its associated OCR sentence embeddings $O = \{s^r_1 \dots, s^r_{L_r}\}$ as the input. L_s, L_f, L_r present the number of sentences. We also add the [CLS] token in the first place of the input sequence. The outputs from Cross-modal Transformer is a sequence of contextualized embeddings. We use the output from the [CLS] token represent the consistency relationship between the video and speech.

$$R^{vs} = \text{Cross-Transformer}(F, O, S) \tag{9}$$

$$R^{vc} = \text{Cross-Transformer}(F, O, C) \tag{10}$$

$$R^{cs} = \text{Cross-Transformer}(C, S) \tag{11}$$

As for the consistency measurement between the claim $C = \{s^c_1 \dots, s^c_{L_c}\}$ and speech S, video F, we also utilize the Cross-modal Transformer to integrate different modalities and [CLS] to represent the weight. The claim can be regarded as the pointer to precisely retrieve relevant evidences from the paired video and speech, which play a key role to infer the truthfulness of the pairs.

Explanations. After feeding pairwise consistency embeddings R^{vs}, R^{vc} and R^{cs} into MLP layers and sigmoid functions, our model produces the relation scores c^{vs} which represents the consistency score between video and speech, c^{vc} which represents the consistency score between video and

Model	Accuracy	F1
CLIP4Clip (Luo et al., 2021b)	57.3	55.8
CLIP2Video (Fang et al., 2021)	59.4	56.8
VideoClip (Xu et al., 2021)	50.4	49.5
McCrae (McCrae et al., 2022)	62.7	62.3
MTS (Liu et al., 2020b)	56.3	55.4
MMT (Gabeur et al., 2020)	61.2	60.6
TwtrDetective (Ours)	**68.1**	**67.9**
Human Check	82.7	81.9

Table 5: Comparative results (%) with baselines on the COVID-VTS Dataset.

Model	Accuracy
MTS (Liu et al., 2020b)	60.4
XML (Lei et al., 2020)	69.6
HERO (Li et al., 2020)	70.4
TwtrDetective (Ours)	**72.6**

Table 6: Comparative results (%) with baselines on the VIOLIN Dataset.

claim and c^{cs}, which represents the consistency score between claim and speech. If a video post is classified as `inconsistent` by our model, the common modality of the two links with lower scores will be pointed as the `inconsistent` modality. For example in Figure 3, `TwtrDetective` detects that speech is inconsistent since c^{vs} and c^{cs} are smaller. It can be used the explanation to support the judgement.

Objective Function. We optimize our model by minimizing the standard cross-entropy as shown on the top of Figure 3, where y is the ground truth label and y^P is the prediction probability after putting c^{vs}, c^{vc} and c^{cs} into the transformer attention layer.

5 Experiments

In this section, we first introduce details of our implementation and compare the results to competing methods. Lastly, we present comprehensive experiment results and discussions.

5.1 Implementation Details

Datasets. We conduct experiments on two datasets: (1) COVID-VTS dataset, which consists of 10k video-text pairs with different taxonomies generated by our automatic tool. Half of the samples are pristine and half are manipulated. (2) VIOLIN dataset (Liu et al., 2020b), a Video-and-Language Inference, collected from TV shows and movies, which contains 15,887 video clips and claims written by human. We also generate inconsistent speech and video pairs with our automatic tool so as to make sure the inconsistency comes from different modalities. Our model is compared with baselines on both datasets.

Model	Accuracy	F1
TwtrDetective	68.1	67.9
TwtrDetective (w/o Event Alert Module)	64.4	64.5
TwtrDetective (w/o PCA)	66.3	65.9

Table 7: Ablation study to investigate our model's performance without the event alert module or the pairwise consistency aggregation module.

Model Training. Our model is implemented using Pytorch. In our implementation, the dimensions for the common subspace is 512. Models are optimized using Adam with learning rate as 0.0005. In addition, we adopt a uniform sampling strategy to extract the frames and the sampling rate is 1 frame per second. In all the experiments, we split the COVID-VTS dataset into 80% for training, 10% for validation and 10% for testing.

Evaluation Metric. We compute the accuracy and F1 score at distinguishing inconsistent pairs from consistent ones.

Baselines. (1) CLIP4Clip (Luo et al., 2021b) (2) CLIP2Video (Fang et al., 2021) uses CLIP (Radford et al., 2021) to encode visual frames and text and use the transformer to fuse temporal information. (3) VideoClip (Xu et al., 2021) is a state-of-art zero-shot video and text understanding model, which learns fine-grained association between video and text in a transformer. (4) McCrae (McCrae et al., 2022) employs Long LSTM to integrate the video, text and named entity information from the video posts. (5) MMT, a video-text model that designs a multi-modal transformer to jointly encode the different modalities in video. (7) HERO (Li et al., 2020): a transformer-based framework with two standard hierarchies for local and global context computation.

5.2 Results and Discussion

Comparison Experiments. As Table 5 and Table 6 show, our model achieves SOTA results compared with baselines on the both `COVID-VTS` and VIOLIN dataset. CLIP-variant models fail to perform well, revealing they do not detect token-level manipulation by analyzing the semantics of the sentence level. We notice that McCrae and MMT outperform MTS on both scores. This is because McCrae and MMT extract additional features from videos to help verification like objective detection and face detection. However, named entity verification

(Speech)
...
efficacy tells us how well the vaccine protects us from a disease in the real world clinical trials need to be precise and replicable
...

(Claim)
tells us how well the vaccine protects us from a disease in the real world clinical trials need to be precise and replicable

(Speech)
...
Omicron could be a threat to the Welsh quick return to economic normality that's according to the OECD
...
OECD. has stressed the global recovery is becoming increasingly unbalanced
...

(Claim)
OECD warns omicron variant could be a prerequisite to economic recovery

(Speech)
...
Germany and Austria are very highly vaccinate countries they're singing on precedent to numbers of infections so much
...
what is the justification for these vaccine mandates at this point...

(Claim)
Large turnout in Melbourne, Australia, in protest march against COVID vaccine mandate

Figure 4: Qualitative analysis results. The red box indicates the inconsistent modality. `TwtrDetective` predicts the first two examples correctly except the third one.

Model	Accuracy	F1
TwtrDetective	68.1	67.9
TwtrDetective (w/o $Pair_{[Claim,Video]}$)	66.6	66.3
TwtrDetective (w/o $Pair_{[Claim,Speech]}$)	67.2	67.0
TwtrDetective (w/o $Pair_{[Speech,Video]}$)	67.5	67.2

Table 8: Analysis on the importance of different modality pairs in the PCA module.

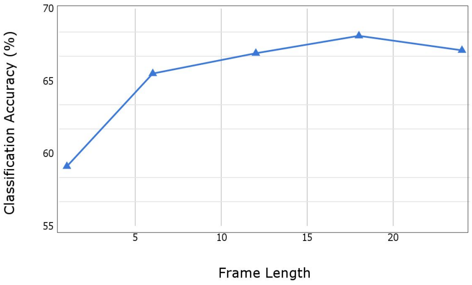


Figure 5: Ablation study to investigate our model's performance with different frame length.

proposed by McCrae does not show improvement on our since because it's incapable of finding the event-trigger manipulation. The main advantages of our model is that our event alert module is able to extract fine-grained factual details from the heterogeneous content. The pairwise consistency aggregation module is able to measure the interaction between each modality precisely. These observations also explain why our model outperforms HERO, which ignores the guide from the event structures. The performance on the VIOLIN dataset is better compared to the results on `COVID-VTS` dataset. This is because our `COVID-VTS dataset` is more challenging with more named entities and events.

Analysis on Different Modalities. We gain further insights into the importance of different modality pairs in PCA module. Our model in the first row of Table 8 uses all three pairs $Pair_{[Claim,Video]}$, $Pair_{[Claim,Speech]}$ and $Pair_{[Speech,Video]}$ in PCA. Other rows miss one of these pairs. Table 8 indicates the PCA module is effective to improve the consistency checking performance between multiple modalities. In addition, $Pair_{[Claim,Video]}$ contributes more than the other two pairs. This is because the evidence sentences are the only manipulated parts in the long speech text, making it challenging to be detected by referring to claims and videos.

Ablation Study. We investigate frame length in Figure 5, we can see a significant increase between 1 and 6 frames which shows it is better to encode the videos with a sequence of multiple frames instead of one single frame. We sampled 18 frames for our experiment, which is both efficient and effective. Furthermore in Table 7, our model mainly benefits from *Event Alert* module which provides 3.7% boost in classification accuracy by explicitly tracking distorted factual pieces at the token level, *PCA* module contributes 1.8% improvement compared to using one-stream transformer to aggregate features from different modalities. Additionally, our model achieves better accuracy on short claims (length < 15) than on long claims (length > 25). This is because long claims have more event triggers and arguments than short claims, which makes it challenging for our event alert module to capture the manipulated tokens.

Qualitative Analysis. Figure 4 presents prediction examples from our model. The correct cases (first two examples) demonstrate the model's ability to recognize the tiny inconsistent tokens. Since the original pair of third example miss the speech modality, we add the speech text which shares more common entities with the claim as the inconsistent modality. `TwtrDetective` fails to predict third post, suggesting that it does not work well on the video and speech alignment. The reasons behind might because both the video and speech describe the protest against the vaccine mandate but in different countries. Only from the video, it's hard to detect whether it's in Australia, Germany or Austria. In addition, the date information is also challenging to check. The future direction could be asking the model to point out which part of the information is inconsistent or unverifiable and define the inconsistency taxonomy for them.

6 Conclusions

In this paper, we release a new benchmark, `COVID-VTS`, for fact-checking in short video platforms, consisting of 10k video-claim pairs. We develop an efficient tool to automatically generate large-scale trustworthy inconsistent pairs with different semantic meanings. Furthermore, our proposed fact-checking model `TwtrDetective` achieves state-of-the-art detection accuracy. We hope this work paves the way for future studies in multi-modal fact-checking as well as other related research areas in video and language.

7 Ethical Statement

Our goal in developing state-of-art consistency checking technique is to enhance the field's ability to detect fake news and improve the Twitter community health. According to Twitter Developer Policy, Twitter supports the research that measures and analyzes topics like spam, abuse, or other platform health-related topics for non-commercial research purposes. In addition, the posts we collected are from verified and authentic accounts. Certain accounts are ineligible for the verified badge if they post content that harasses, shames, or insults any individual or group, or violate the Platform manipulation and spam policy. To protect the personal information, we will only use the captions and videos as input without the user information. We also work to filter the dataset and only keep the posts discussing public news instead of personal life. Personal information but not limited to, user's name, health, financial status, racial or ethnic origin, religious or philosophical affiliation or beliefs, sexual orientation, trade union membership, alleged or actual commission of crime. It's crucial to mention that we will not share our source video file but the URL links or the extracted features from the videos. This is to due to the licence policy and avoid anyone to deliberately generate and spread misinformation. As such, we will release the model code but not the output in our work for public verification and auditing so it can be used to combat fake news.

8 Limitations

There is a significant gap between our model and human performance on the accuracy. We hope `COVID-VTS` dataset will encourage the community to develop stronger models in the future. One possible direction is to develop models to localize key frames or key sentences from the speech to deduce the difficulty of consistency check.

Acknowledgements

This work was supported by DARPA SemaFor (HR001119S0085) program. We are thankful for the feedback from anonymous reviewers of this paper.

References

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing moments in video with natural language. In *Proceedings of the IEEE international conference on computer vision*, pages 5803–5812.

Pepa Atanasova, Preslav Nakov, Lluís Màrquez, Alberto Barrón-Cedeño, Georgi Karadzhov, Tsvetomila Mihaylova, Mitra Mohtarami, and James Glass. 2019. Automatic fact-checking using context and discourse information. *Journal of Data and Information Quality (JDIQ)*, 11(3):1–27.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019. Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims. *arXiv preprint arXiv:1909.03242*.

Giannis Bekoulis, Christina Papagiannopoulou, and Nikos Deligiannis. 2021. A review on fact extraction and verification. *ACM Computing Surveys (CSUR)*, 55(1):1–35.

David Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 190–200.

Han Fang, Pengfei Xiong, Luhui Xu, and Yu Chen. 2021. Clip2video: Mastering video-text retrieval via image clip. *arXiv preprint arXiv:2106.11097*.

Yi Fung, Christopher Thomas, Revanth Gangi Reddy, Sandeep Polisetty, Heng Ji, Shih-Fu Chang, Kathleen McKeown, Mohit Bansal, and Avirup Sil. 2021. Infosurgeon: Cross-media fine-grained information consistency checking for fake news detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*

Processing (Volume 1: Long Papers), pages 1683–1698.

Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. 2020. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision*, pages 214–229. Springer.

David Güera and Edward J Delp. 2018. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS)*, pages 1–6. IEEE.

Andreas Hanselowski, Christian Stab, Claudia Schulz, Zile Li, and Iryna Gurevych. 2019. A richly annotated corpus for different tasks in automated fact-checking. *arXiv preprint arXiv:1911.01214*.

Tom Hope, Aida Amini, David Wadden, Madeleine van Zuylen, Sravanthi Parasa, Eric Horvitz, Daniel Weld, Roy Schwartz, and Hannaneh Hajishirzi. 2020. Extracting a knowledge base of mechanisms from covid-19 papers. *arXiv preprint arXiv:2010.03824*.

Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. 2019. Attention on attention for image captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4634–4643.

Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. 2020. Hover: A dataset for many-hop fact extraction and claim verification. *arXiv preprint arXiv:2011.03088*.

Mi-Ae Ko and Young-Mo Kim. 2004. A simple ocr method from strong perspective view. In *33rd Applied Imagery Pattern Recognition Workshop (AIPR'04)*, pages 235–240. IEEE.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715.

Jie Lei, Licheng Yu, Tamara L Berg, and Mohit Bansal. 2020. Tvr: A large-scale dataset for video-subtitle moment retrieval. In *European Conference on Computer Vision*, pages 447–463. Springer.

Linjie Li, Yen-Chun Chen, Yu Cheng, Zhe Gan, Licheng Yu, and Jingjing Liu. 2020. Hero: Hierarchical encoder for video+ language omni-representation pre-training. *arXiv preprint arXiv:2005.00200*.

Fuxiao Liu, Yinghan Wang, Tianlu Wang, and Vicente Ordonez. 2020a. Visual news: Benchmark and challenges in news image captioning. *arXiv preprint arXiv:2010.03743*.

Jingzhou Liu, Wenhu Chen, Yu Cheng, Zhe Gan, Licheng Yu, Yiming Yang, and Jingjing Liu. 2020b. Violin: A large-scale dataset for video-and-language inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10900–10910.

Yang Liu, Samuel Albanie, Arsha Nagrani, and Andrew Zisserman. 2019a. Use what you have: Video retrieval using representations from collaborative experts. *arXiv preprint arXiv:1907.13487*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Grace Luo, Trevor Darrell, and Anna Rohrbach. 2021a. Newsclippings: Automatic generation of out-of-context multimodal media. *arXiv preprint arXiv:2104.05893*.

Huaishao Luo, Lei Ji, Ming Zhong, Yang Chen, Wen Lei, Nan Duan, and Tianrui Li. 2021b. Clip4clip: An empirical study of clip for end to end video clip retrieval. *arXiv preprint arXiv:2104.08860*.

Scott McCrae, Kehan Wang, and Avideh Zakhor. 2022. Multi-modal semantic inconsistency detection in social media news posts. In *International Conference on Multimedia Modeling*, pages 331–343. Springer.

Nicholas Micallef, Marcelo Sandoval-Castañeda, Adi Cohen, Mustaque Ahamad, Srijan Kumar, and Nasir Memon. 2022. Cross-platform multimodal misinformation: Taxonomy, characteristics and detection for textual posts and videos. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 651–662.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *arXiv preprint arXiv:2005.10200*.

John F Pitrelli, Raimo Bakis, Ellen M Eide, Raul Fernandez, Wael Hamza, and Michael A Picheny. 2006. The ibm expressive text-to-speech synthesis system for american english. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1099–1108.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. 2015. A dataset for movie description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3202–3212.

Arkadiy Saakyan, Tuhin Chakrabarty, and Smaranda Muresan. 2021. Covid-fact: Fact extraction and verification of real-world claims on covid-19 pandemic. *arXiv preprint arXiv:2106.03794*.

Tal Schuster, Roei Schuster, Darsh J Shah, and Regina Barzilay. 2020. The limitations of stylometry for detecting machine-generated fake news. *Computational Linguistics*, 46(2):499–510.

Lanyu Shang, Ziyi Kou, Yang Zhang, and Dong Wang. 2021. A multimodal misinformation detector for covid-19 short videos on tiktok. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 899–908. IEEE.

Yusuxke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, Takeshi Shinohara, and Setsuo Arikawa. 1999. Byte pair encoding: A text compression scheme that accelerates pattern matching.

Reuben Tan, Bryan A Plummer, and Kate Saenko. 2020. Detecting cross-modal inconsistency to defend against neural fake news. *arXiv preprint arXiv:2009.07698*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *arXiv preprint arXiv:2004.14974*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Kehan Wang, David Chan, Seth Z Zhao, John Canny, and Avideh Zakhor. 2022. Misinformation detection in social media video posts. *arXiv preprint arXiv:2202.07706*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. 2021. Videoclip: Contrastive pre-training for zero-shot video-text understanding. *arXiv preprint arXiv:2109.14084*.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in neural information processing systems*, 32.

Multimodal Graph Transformer for Multimodal Question Answering

Xuehai He
UC Santa Cruz
xhe89@ucsc.edu

Xin Eric Wang
UC Santa Cruz
xwang366@ucsc.edu

Abstract

Despite the success of Transformer models in vision and language tasks, they often learn knowledge from enormous data implicitly and cannot utilize structured input data directly. On the other hand, structured learning approaches such as graph neural networks (GNNs) that integrate prior information can barely compete with Transformer models. In this work, we aim to benefit from both worlds and propose a novel Multimodal Graph Transformer for question answering tasks that requires performing reasoning across multiple modalities. We introduce a graph-involved plug-and-play quasi-attention mechanism to incorporate multimodal graph information, acquired from text and visual data, to the vanilla self-attention as effective prior. In particular, we construct the text graph, dense region graph, and semantic graph to generate adjacency matrices, and then compose them with input vision and language features to perform downstream reasoning. Such a way of regularizing self-attention with graph information significantly improves the inferring ability and helps align features from different modalities. We validate the effectiveness of Multimodal Graph Transformer over its Transformer baselines on GQA, VQAv2, and MultiModalQA datasets.

1 Introduction

A myriad of complex real-world tasks require both prior knowledge and reasoning intelligence (Yi et al., 2018a; Ilievski and Feng, 2017). These days, vision-and-language reasoning tasks such as as vision question answering (VQA) (Antol et al., 2015) and multimodal question answering (MultiModalQA) (Talmor et al., 2021) post further needs for integrating structured info from different input modalities and thus perform reasoning. Towards this, two questions yield: What is the best way to integrate prior knowledge and reasoning components from multiple modalities in a single model? How would such an integration lead to accurate models, while being more computationally efficient and allowing for significantly more interpretability? Such questions are important to address when scaling reasoning systems to real-world use cases.

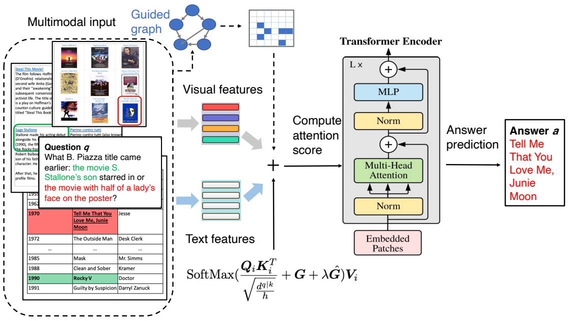


Figure 1: Overview of Multimodal Graph Transformer. It takes visual features, text features, and their corresponding generated graphs as inputs. The generated graph is first converted to an adjacency matrix to induce the mask matrix $\boldsymbol{G}$. The modified quasi-attention score in the Transformer is computed to infer the answer. In the formular, $\boldsymbol{G}$ is the graph-induced matrix constructed by concatenating adjacency matrices both from the vision and the language end. $\hat{\boldsymbol{G}}$ is the trainable bias. The input features from different modalities are fused along with graph info to perform downstream reasoning.

These years, there are a spectrum of methods in the literature exploring different ways of integrating structured prior information. Graph neural networks (GNNs) (Wu et al., 2020), have been widely used in representation learning on graphs. Some experts tried to investigate the embedding of the structured information by resorting to them. However, GNNs are inefficient (Wu et al., 2020) and they can barely compete with Transformer models. Besides, most GNNs are designed to learn node representations on fixed and homogeneous graphs. Thereby, it is suboptimal to operate GNNs on vision-and-language tasks such as visual question answering (VQA), where graphs encountered in these problems (e.g. scene graphs) can be more complex; Alternatively, knowledge graphs (KGs),

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 189–200
May 2-6, 2023 ©2023 Association for Computational Linguistics

such as Freebase (Bollacker et al., 2008), represent world-level factoid information of entities and their relations in a graph-based format, surfaced these years. They have been successfully used in vision and language applications including VQA (Marino et al., 2019). However, they have not been dedicated to be applied to our scenario, more concretely, we aim at filling the gap of capturing prior knowledge in Transformer models.

To mitigate deficiencies of the existing methods, this paper proposes a novel plug-and-play graph-involved Transformer-based method for multimodal question answering tasks. Our method is *Multimodal Graph Transformer* in the sense that it is built upon the well-established Transformer (Vaswani et al., 2017a) backbone, albeit with several key fundamental differences. First, we introduce a systematic scheme to convert text graphs, dense region graphs, and semantic graphs from vision and language tasks to adjacency matrices to use in our method. Second, instead of directly computing the attention score, we learn the newly proposed quasi-attention score with graph-induced adjacency matrices live at its heart, to signify the importance of learning relative importance as a highly effective inductive bias for computing the quasi-attention score. Third, different from previous Transformer methods, where self-attention are fully learned from data, we switch gears to introduce the graph-structured information in the self-attention computation to guide the training of Transformers as shown in Figure 1.

The main contributions are summarized below:

- We propose a novel Multimodal Graph Transformer learning framework that combines multimodal graph learning from unstructured data with Transformer models.
- We introduce a modular plug-and-play graph-involved quasi-attention mechanism with a trainable bias term to guide the information flow during training.
- The effectiveness of the proposed methods is empirically validated on GQA, VQA-v2, and MultiModalQA tasks.

2 Related Works

2.1 Multimodal question answering

Visual Question Answering (VQA)(Antol et al., 2015) has been a prominent topic in the field of multimodal question answering, garnering significant attention and advancing significantly since the introduction of the first large-scale VQA dataset byAntol et al. (2015). To answer VQA questions, models typically leverage variants of attention to obtain a representation of the image that is relevant to the question (Andreas et al., 2016; Yang et al., 2015; Xu and Saenko, 2016; Fukui et al., 2016; Lu et al., 2016). A plethora of works (Liang et al., 2021; Hudson and Manning, 2018; Yi et al., 2018b; Xiong et al., 2016; Kim et al., 2018; Teney et al., 2017a) have attempted to enhance the reasoning capability of VQA models, with Teney et al. (2017a) proposing to improve VQA using structured representations of the scene contents and questions. They developed a deep neural network that leverages the structure in these representations and builds graphs over scene objects and question words. The recent release of MultiModalQA (Talmor et al., 2021), a dataset that demands joint reasoning over texts, tables, and images, has received widespread attention. However, similar to VQA, existing MultiModalQA methods have not fully utilized structured information from the input concepts. To address this, we propose a combination of multimodal graph learning and Transformer models to improve question answering across inputs from multiple different modalities.

2.2 Attention mechanisms

The attention mechanism (Xu et al., 2015a,b; Devlin et al., 2018), has dramatically advanced the field of representation learning in machine learning. The attention mechanism is introduced in Vaswani et al. (2017b) and widely used in language tasks (i.e., abstract summarization (Xu et al., 2020)), machine translation (Bahdanau et al., 2014), reading comprehension (Dai et al., 2020), question answering (Min et al., 2019), etc. Zhang et al. (2020) proposes using syntax to guide the text modeling by incorporating explicit syntactic constraints into attention mechanisms. Meanwhile, it has seen increasing application in multimodal tasks (Li et al., 2020; Nam et al., 2017; Lu et al., 2016), where it is usually used for learning of interactions between multiple inputs. Following their success, multimodal Transformer models (Chen et al., 2019; Hu et al., 2020; Sun et al., 2019) have also shown impressive results on several vision-and-language tasks. Yun et al. (2019) proposes Graph Transformer Networks (GTNs) that can generate new graph structures and learn effective node representation on the new graphs in an end-to-end fashion. Different from these works, our work incorporates graph information from different modalities into the Transformer to improve the reasoning ability.

2.3 Exploiting graphs in multimodal reasoning

Considering that graph priors can transfer commonalities and mitigate the gap between visual and language domains, researchers explore how to use graphs (Teney et al., 2017b; Yu et al., 2020) properly in both tasks. In recent years, many classes of GNNs have been developed for both tasks which are divided into two approaches: spectral (Bruna et al., 2013) and non-spectral methods (Chen et al., 2018). Graphs can also be transferred into latent variables by GCN (Yang et al., 2019a; Yao et al., 2018), which can be directly utilized by models. However, the need for aligning graph priors from different modalities to do reasoning limits the use of graph priors. Our work addresses this problem via the graph-involved quasi-attention mechanism.

2.4 Pretraining

Pretrained models in computer vision (Simonyan and Zisserman, 2014; He et al., 2016) and NLP (Devlin et al., 2018; Yang et al., 2019b; Liu et al., 2019), have achieved state-of-the-art performances in many downstream tasks (Thongtan and Phienthrakul, 2019; White et al., 2017; Karpathy and Fei-Fei, 2015; Lee et al., 2018; Ren et al., 2015b). Other pretrained models such as VLBERT (Lu et al., 2019; Sun et al., 2019) and ViLT (Kim et al., 2021) also demonstrate their effectiveness on downstream vision-language tasks. Recent works on vision-language pretraining such as OSCAR (Li et al., 2020) perform cross-modal alignment in their visual-language pretraining models. Likewise, our proposed method includes cross-modality alignment, which is critical for reasoning. Our proposed modular plug-and-play graph-involved quasi-attention mechanism is also model-agnostic and can be also applied to other pretrained Transformer-based vision and language models.

3 Multimodal Graph Transformer

3.1 Background on Transformers

The Transformer layer (Vaswani et al., 2017b) consists of two modules: a multi-head attention and a feed-forward network (FFN). Specifically, each head is represented by four main matrices: the query matrix $\boldsymbol{W}_i^q \in \mathbb{R}^{d^m \times d^q/h}$, the key matrix $\boldsymbol{W}_i^k \in \mathbb{R}^{d^m \times \frac{d^k}{h}}$, the value matrix $\boldsymbol{W}_i^v \in \mathbb{R}^{d^m \times \frac{d^v}{h}}$, and the output matrix $\boldsymbol{W}_i^o \in \mathbb{R}^{\frac{d^v}{h} \times d^o}$, and takes the hidden states $\boldsymbol{H} \in \mathbb{R}^{l \times d^m}$ of the previous layer as input, where d denotes the dimension of the model, h represents the number of head, and i denotes the index of layer number. The output of attention is given by:

$$\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i = \boldsymbol{H}\boldsymbol{W}_i^q, \boldsymbol{H}\boldsymbol{W}_i^k, \boldsymbol{H}\boldsymbol{W}_i^v \tag{1}$$

$$\mathrm{Attention}\left(\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i\right) = \mathrm{SoftMax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^T}{\sqrt{\frac{d^{q|k}}{h}}}\right)\boldsymbol{V}_i \tag{2}$$

$$\boldsymbol{H}_i = \mathrm{Attention}\left(\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i\right)\boldsymbol{W}_i^o \tag{3}$$

where $\boldsymbol{Q}_i \in \mathbb{R}^{l \times \frac{d^q}{h}}, \boldsymbol{K}_i \in \mathbb{R}^{l \times \frac{d^k}{h}}, \boldsymbol{V}_i \in \mathbb{R}^{l \times \frac{d^v}{h}}$ are obtained by the linear transformations of $\boldsymbol{W}_i^q, \boldsymbol{W}_i^k, \boldsymbol{W}_i^v$ respectively. $\mathrm{Attention}(\cdot)$ is the scaled dot-product attention operation. Then output of each head is transformed to $\boldsymbol{H}_i \in \mathbb{R}^{l \times d^o}$ by $\boldsymbol{W}_i^o$.

3.2 Framework overview

The entire framework of the proposed Multimodal Graph Transformer method is depicted in Figure 2. Without loss of generality, we assume the end task is VQA in the following discussion while noting that our framework can be applied to other vision-language tasks, such as multimodal question answering.

Given the input images and questions, the framework first constructs three graphs, including the semantic graph, dense region graph, and text graph, which will be described in more detail in the following sections. The graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the set of nodes in the graph and $\mathcal{E}$ represents the edges connecting them, is fed into Transformers to guide the training process.

3.3 Multimodal graph construction

We build three types of graphs and feed them into Transformers: *text graph*, *semantic graph*, and *dense region graph*. We now introduce them in detail.

Text graph The task of Visual Question Answering involves a combination of an image, a question, and its corresponding answer. To process the question, we extract the entities and create a text graph representation. We then build the graph $G = (\mathcal{V}, \mathcal{E})$

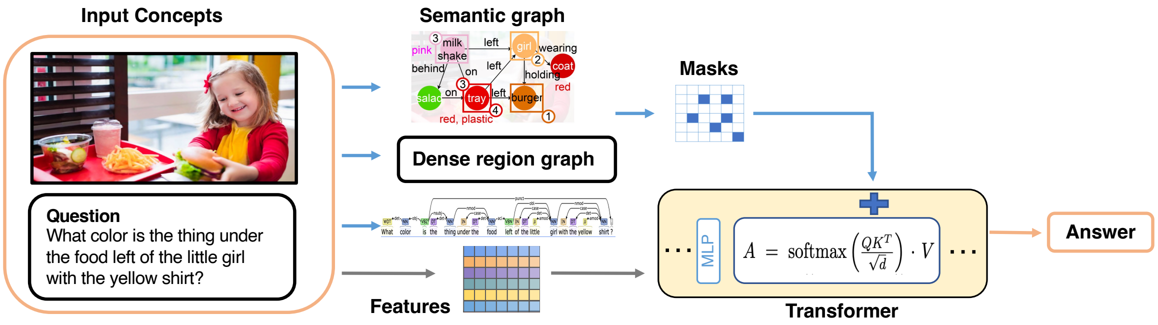


Figure 2: The figure illustrates the overall framework of our Multimodal Graph Transformer. The input from different modalities are processed and transformed into corresponding graphs, which are then converted into masks and combined with their features to be fed into Transformers for downstream reasoning. In detail, semantic graphs are created through scene graph generation methods, dense region graphs are extracted as densely connected graphs, and text graphs are generated through parsing.

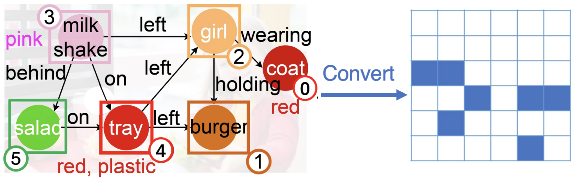


Figure 3: The naive demonstration of converting a semantic graph into an adjacency matrix. Cells in blue means '0's for that element in the graph matrix, while white ones means '-inf's. We employ the matrix as the mask when computing the quasi-attention.

as shown in the left of Figure 2. The set of nodes, $\mathcal{V}$, represents the entities and the set of edges, $\mathcal{E}$, represents the relationships between the pairs of entities. This results in:

- A set of N entities, each represented by a vector of token embeddings, that constitute the nodes of the graph.
- A set of pairwise relations between entities, forming the edges of the text graph. The relationship between entities i and j is represented by a vector e_{ij} which encodes the relative relationships.

Semantic graph In tasks such as multimodal question answering, there might be additional inputs in the form of tables or lengthy paragraph sentences. To handle these inputs, a linear representation of the table can be created and a semantic graph can be constructed using a similar approach. They are processed using the scene graph parser (Zhong et al., 2021), which transforms the text sentence into a graph of entities and relations, as depicted in Figure 3. The output of the scene graph parser includes:

- A set of N words that constitute the nodes of the semantic graph, where N is the number of parsed words in the texts.
- A set of possible pairwise relations between words, such as "left" and "on" as shown in Figure 3, which constitute the edges of our graph. An edge between words connecting j to i is represented by e_{ij}, namely, the connectivity is indicated as: $e_{ij} = \begin{cases} 0, & i,j \text{ not connected} \\ 1, & i,j \text{ connected} \end{cases}$.

Dense region graph The visual features are extracted by slicing the input images into patches and flattening them. A dense region graph $G = (\mathcal{V}, \mathcal{E})$ is then converted into masks, with $\mathcal{V}$ being the set of extracted visual features and $\mathcal{E}$ being the set of edges connecting each feature node, following the method described in (Kim et al., 2021). This results in a graph that is nearly fully connected.

The resulting three graphs are then transformed into adjacency matrices, where the elements are either -∞ or zero. The conversion process is depicted in Figure 3 using the semantic graph as an example. These adjacency matrices are used inside the scaled dot-product attention to control the flow of information, by masking out (setting to $-\infty$) the values.

3.4 Graph-involved quasi-attention

In order to effectively utilize structured graph knowledge in our self-attention computation, we incorporate the graph as an extra constraint in each

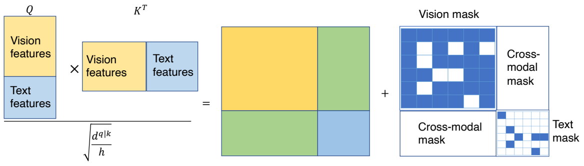


Figure 4: A naive demonstration of adding the graph-induced mask while computing the quasi-attention when the inputs are from two modalities. The visual mask is the mask converted from the dense region graph and the text mask is converted from the text graph. The cross-modal mask, which is always set as an all-zero matrix, is imposed to encourage the model to learn the cross-attention between the image features and text features, thus facilitating the alignment across them.

attention head by converting it into an adjacency matrix. The graph matrix, denoted as $\boldsymbol{G}$, is constructed by combining various masks. An illustration of this process can be seen in Figure 4. The visual mask is generated from the dense region graph, while the text mask is derived from the text graph. Additionally, the cross-modal mask is set to an all-zero matrix to encourage the model to learn the cross-attention between visual and text features, thereby promoting alignment across the different modalities.

Within the context of adding graph information, when vision graph mask and text graph mask are concatenated and aligned with image and text features, we believe that a more flexible masking-out mechanism is beneficial, rather than keeping a single constant mask matrix inside the Softmax operation. Drawing insights from Liu et al. (2021), where they include a relative position bias to each head in computing similarity, we also intuitively parameterize a trainable bias $\hat{\boldsymbol{G}}$ and involve it in the training process. Finally, we compute the quasi-attention as follows:

$$\text{Attention} = \text{SoftMax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^T}{\sqrt{\frac{d^{q|k}}{h}}} + \boldsymbol{G} + \lambda \hat{\boldsymbol{G}}\right)\boldsymbol{V}_i, \tag{4}$$

where λ is the tradeoff hyper-parameter that controls the contribution of $\hat{\boldsymbol{G}}$, and $\boldsymbol{G}$ is our graph-induced matrix constructed by concatenating a graph matrix both from the vision and the language end. Here for clear clarification, we use $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ to distinguish the graph matrices fixed and trainable, respectively. During training, $\boldsymbol{G}$ is frozen as before and does not receive gradient updates, while $\hat{\boldsymbol{G}}$ contains trainable parameters.

We now introduce the motivation behind adding two types of graph matrices. We perform the masking process by adding $\boldsymbol{G}$ when computing the quasi-attention because it can be interpreted as a form of attentional pooling (learning to align), in which each element of $\boldsymbol{G}$ pools all relevant information across all elements of the relative importance matrix computed by $\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^T}{\sqrt{\frac{d^{q|k}}{h}}}\right)$. Hence during fine-tuning, the model ignores redundant features and only focuses on useful information. The mask can also force the model to learn the cross attention between features from the images and questions and perform aligning across them. And the trainable bias $\hat{\boldsymbol{G}}$ captures information gained during the training process. Such information is valuable for fine-tuning, making the Transformer more robust and helping it gain numerical stability.

3.5 Training

The interdependence of output features from various modalities calls for a unified optimization approach for the Transformers in both the visual question answering and multimodal question answering tasks. To accomplish this, we implement a kind of end-to-end training, which ensures the optimality of the models. The final outcome of our models is a classification logit, which is generated by the VQA models that select the best answer from the available candidate answers. To evaluate the accuracy of the models, we compute the cross-entropy loss (Zhang and Sabuncu, 2018) using the output logits produced by the Transformer. This measure helps us determine the difference between the predicted class probabilities and the actual class labels.

4 Experiments

4.1 Datasets

VQA v2 The VQA v2 dataset (Goyal et al., 2017) extends the VQA (Antol et al., 2015) dataset to better balance visual and textual information through the collection of complementary images. Each question in VQA v2 is associated with a pair of similar images with different answers, resulting in a total of 1.1 million QA pairs and 204,000 images. The data split for VQA v2 includes a training set with 83,000 images and 444,000 questions, a validation set with 41,000 images and 214,000 questions, and a test set with 81,000 images and 448,000 questions. The annotated answers are in natural language, but they are commonly converted to a classification task with 3,129 answer classes.

Table 1: Accuracy (%) comparison of different methods on the GQA and VQA v2 test-dev. Ours has the second best performance and is comparable to state-of-the-art methods. After applying our proposed quasi-attention mechanism and exploiting the use of graphs, there is also a 2% improvement of overall accuracy on the LXMERT baseline, suggesting the generalization ability of our method.

Dataset	Method	Open questions	Binary questions	Overall accuracy
GQA	LXMERT (Tan and Bansal, 2019)	-	-	60.0
	LXMERT w/ Graph (Tan and Bansal, 2019)	-	-	61.4
	HANs (Kim et al., 2020)	-	-	69.4
	NSM (Hudson and Manning, 2019b)	49.3	78.9	63.2
	OSCAR (Li et al., 2020)	-	-	61.6
	VinVL (Zhang et al., 2021)	-	-	65.1
	Multimodal Graph Transformer (Ours)	59.4	80.5	68.7
VQA v2	LXMERT (Tan and Bansal, 2019)	-	-	72.4
	HANs (Kim et al., 2020)	-	-	65.1
	NSM (Hudson and Manning, 2019b)	-	-	63.0
	OSCAR (Li et al., 2020)	-	-	73.8
	VinVL (Zhang et al., 2021)	-	-	76.6
	Multimodal Graph Transformer (Ours)	66.5	87.0	74.5

As described by Anderson et al. (2018), the model selects the answer to each question from a set of 3,129 most frequent answers. Following this convention, we fine-tune the multimodal graph transformer model on the VQAv2 training and validation sets, while reserving 1,000 validation images and related questions for internal validation.

GQA The GQA dataset contains 22M questions over 113K images. The questions in GQA are designed to require multi-hop reasoning to test the reasoning skills of VQA models. GQA greatly increases the complexity of the semantic structure of questions, leading to a more diverse function set. The real-world images in GQA also bring in a bigger challenge in visual understanding. We conduct experiments on the public splits (Hudson and Manning, 2019a) of the GQA dataset and also treat the task as the classification task reffering to the VQA v2 setting.

MultiModalQA MultiModalQA (MMQA) contains 29, 918 questions. We split the dataset into 23,817 training, 2,441 development (dev.), and 3,660 test set examples referring to the official split. Around 60% of the questions in MMQA are compositional. The answer for each question can be a single answer or a list of answers.

4.2 Baselines

We compare with four state-of-the-art VQA models: LXMERT (Tan and Bansal, 2019), NSM (Hudson and Manning, 2019b), OSCAR (Li et al., 2020), and VinVL (Zhang et al., 2021).

- LXMERT (Tan and Bansal, 2019) designs five pretraining tasks: masked language modeling, feature regression, label classification, cross-modal matching, and image question answering to pretrain a large Transformer model. Towards this, a large-scale Transformer (Vaswani et al., 2017b) model is built that consists of three encoders: an object relationship encoder, a language encoder, and a cross-modal encoder.
- NSM (Hudson and Manning, 2019b) predicts a probabilistic graph that represents its underlying semantics and performs sequential reasoning over the graph to traversing its nodes to make the inference.
- OSCAR (Li et al., 2020) uses object tags detected in images as anchor points to significantly ease the learning of alignments, improving previous methods and using self-attention to learn image-text semantic alignments.
- VinVL (Zhang et al., 2021) developed a new object detection model to create better visual features of images than previous classical object detection models.

We compare with four baselines introduced in the MultiModalQA paper (Talmor et al., 2021): Question-only (Kaushik and Lipton, 2018), Context-only (Kaushik and Lipton, 2018), AutoRouting, ImplicitDecomp.

- Question-only is a sequence-to-sequence model that directly generates the answer given the question.

- Context-only first predicts the question type using the classifier and then feed in the relevant context to predict the answer.
- AutoRouting first determines the modality where the answer is expected to occur, and then runs the corresponding single-modality module.
- ImplicitDecomp is a 2-hop implicit decomposition baseline and so far the state-of-the-art method on the MultiModalQA dataset.

4.3 Implementation details

The input texts undergo preprocessing using a scene graph parser which extracts entities and their relationships. The text features are obtained through a pre-trained BERT tokenizer, allowing us to extract text spans of individual entities and text spans containing two related entities. As for images, we employ the methods described in Dosovitskiy et al. (2020); Kim et al. (2021) to extract visual features and create graph masks. This involves resizing the shorter edge of the input images while preserving the aspect ratio and limiting the longer edge, followed by patch projection and padding for batch training. The resulting patch embeddings are used as inputs along with constructed dense region graph that is densely connected. The Transformer backbone used in this setting is the pretrained VIT-B-32 (Dosovitskiy et al., 2020) version, consisting of 12 layers with a hidden size of $H = 768$, layer depth of $D = 12$, patch size of $P = 32$, a multi-layer perceptron size of 3072, and 12 attention heads. To test this setting, all inputs and graphs are merged and processed by the Transformer backbone, which learns from features from different modalities.

4.3.1 MultiModalQA

We further investigate the effectiveness of our proposed method on MultiModalQA (Talmor et al., 2021), a recently introduced and demanding task that requires joint reasoning across various modalities such as texts, images, tables, etc. We employ a Multimodal Graph Transformer to tackle the task, using the same approach for extracting vision and text features as in VQA. Additional modalities, such as tables, are encoded by linearizing them and utilizing pre-trained models like RoBERTa-large (Liu et al., 2019). After generating text graphs, semantic graphs, and dense region graphs from input questions, text, tables, and images, we feed them along with the extracted features into the Transformer. Unlike the Transformer used in VQA, which takes inputs from two modalities, the MultiModalQA Transformer accepts input from three modalities and performs the final reasoning.

Table 2: EM (%) and F1 (%) of Multimodal Graph Transformer and its Transformer baseline on questions in MultiModalQA that require reasoning over multiple modalities. Incorporating graph information into the Multimodal Graph Transformer can boost about 2% F1 and 4% EM performance.

Method	EM	F1
Question-only	16.9	19.5
Context-only	6.6	8.5
AutoRouting	32.0	38.2
ImplicitDecomp	46.5	51.7
Human	84.8	90.1
Multimodal Transformer w/o Graph	50.1	56.4
Multimodal Graph Transformer (Ours)	52.1	57.7

4.4 Results and analysis

Table 1 presents a comparison of the accuracy of our proposed method on the GQA dataset with previous state-of-the-art methods. Our proposed method ranks second in terms of accuracy and outperforms the third best method by a substantial margin, with an absolute improvement of over 3% in overall accuracy. The performance of our method is comparable to the state-of-the-art method.

We also conducted experiments on the VQA v2 dataset, and the results are summarized in Table 1 and Table 3. As shown, there are significant improvements over methods without graphs, suggesting that incorporating graph information into the Transformer is effective.

Additionally, after incorporating our proposed graph method into LXMERT, we can observe a boost in overall accuracy on the GQA dataset, demonstrating the generalization ability of the proposed method in incorporating graph information into quasi-attention computation.

Table 2 compares the Exact Match (EM) and average F1 score of our proposed method on the MultiModalQA dataset with the baseline. The results show that our proposed method outperforms the baseline without the aid of graph information, demonstrating the generalization of our method to more complicated vision-and-language reasoning

Table 3: Ablation Studies on the GQA and VQA v2 Validation Sets. The figure demonstrates the effectiveness of incorporating graph information into the Transformer architecture through ablation studies performed on the GQA and VQA v2 validation sets. The results of these studies clearly indicate that including graph information can lead to an improvement in performance.

Dataset	Method	Open questions	Binary questions	Overall accuracy
GQA	One-modality Transformer	47.7	78.1	62.7
	Multimodal Transformer w/o Graph	49.9	81.0	65.4
	Ours	**60.1**	**90.2**	**72.4**
VQA v2	One-modality Transformer w/ one Transformer	60.5	85.4	70.1
	Multimodal Transformer w/o Graph	64.8	86.3	72.1
	Ours	**66.7**	**87.2**	**74.6**

tasks.

4.5 Ablation studies

We perform ablation studies to verify the necessity of using two-stream inputs with the help of graphs to deal with input from different modalities, with GQA dataset as our testing bed. For all experiments, we use the overall accuracy as the evaluation metric.

The results presented in Table 3 show the superiority of our proposed Multimodal Graph Transformer over the method where a single modality input is fed into a Transformer. Our method, which involves dividing the input streams into two separate parts and processing each part through a Transformer, outperforms the Multimodal Transformer without Graph. This demonstrates the beneficial effect of incorporating graph information into the processing of the input data and performing training. The use of different input features with the help of graphs allows for a better alignment of the information from different modalities, which is reflected in the improved performance of our proposed method.

4.6 Qualitative results

One qualitative example is shown in Figure 5. As can be seen, predictions from Multimodal Graph Transformer are more relevant to contents of the input image as the graph information improves the inferring ability of the Transformer, which further indicates the effectiveness of Multimodal Graph Transformer.

5 Conclusions

In this paper, we have presented a novel method to integrate structured graph information to guide the Transformers training. Our method can model interactions between different modalities and achieves competitive performance on multimodal reasoning tasks such as VQA and MultiModalQA. Experimental results show that our method outperforms many other methods on the GQA dataset. More importantly, the proposed quasi-attention mechanism is model-agnostic and it is possible to apply it to other Transformer-based methods. We will test our methods on other vision-and-language reasoning tasks and include the comparison with existing graph representation learning methods in our future work.

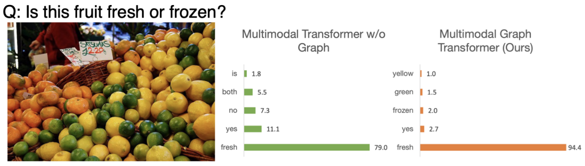


Figure 5: A qualitative comparison from VQA v2. *fresh* is the ground truth. Predictions from the Multimodal Graph Transformer (ours) are more relevant to the contents of the input image and achieve a higher confidence score over the ground truth.

6 Limitations and Potential Risks

The Limitations of the proposed Multimodal Graph Transformer include the potential preservation of fairness and bias issues inherent in the pretrained Transformer models, despite the involvement of graph information. Additionally, the integration of graphs may introduce new biases that can further exacerbate the problem. One potential source of bias is the vision-and-language dataset itself, which may favor majority cases and overlook minority cases. Unfortunately, the proposed method is not equipped to address these biases and issues, making further research and consideration crucial when building upon or directly using this method for vision and language tasks.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Stanislaw Antol, C Lawrence Zitnick, and Devi Parikh. 2014. Zero-shot learning via visual abstraction. In *ECCV*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Jie Chen, Tengfei Ma, and Cao Xiao. 2018. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.

Zihang Dai, Guokun Lai, Yiming Yang, and Quoc V Le. 2020. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. *arXiv preprint arXiv:2006.03236*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. 2020. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9992–10002.

Drew A Hudson and Christopher D Manning. 2018. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*.

Drew A Hudson and Christopher D Manning. 2019a. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6700–6709.

Drew A Hudson and Christopher D Manning. 2019b. Learning by abstraction: The neural state machine. *arXiv preprint arXiv:1907.03950*.

Ilija Ilievski and Jiashi Feng. 2017. Multimodal learning and reasoning for visual question answering. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 551–562.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Divyansh Kaushik and Zachary C Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. *arXiv preprint arXiv:1808.04926*.

Eun-Sol Kim, Woo Young Kang, Kyoung-Woon On, Yu-Jung Heo, and Byoung-Tak Zhang. 2020. Hypergraph attention networks for multimodal learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14581–14590.

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. 2018. Bilinear attention networks. In *NIPS*.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR.

Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. 2018. Stacked cross attention for image-text matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 201–216.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Weixin Liang, Yanhao Jiang, and Zixuan Liu. 2021. Graghvqa: Language-guided graph neural networks for graph-based visual question answering. *arXiv preprint arXiv:2104.10283*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. *Advances in neural information processing systems*, 29:289–297.

Mateusz Malinowski and Mario Fritz. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *NIPS*.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3195–3204.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. 2017. Dual attention networks for multimodal reasoning and matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 299–307.

Mengye Ren, Ryan Kiros, and Richard Zemel. 2015a. Image question answering: A visual semantic embedding model and a new dataset. *NIPS*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015b. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. 2012. Indoor segmentation and support inference from rgbd images. In *ECCV*.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodalqa: Complex question answering over text, tables and images. *arXiv preprint arXiv:2104.06039*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Damien Teney, Lingqiao Liu, and Anton van Den Hengel. 2017a. Graph-structured representations for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Damien Teney, Lingqiao Liu, and Anton van Den Hengel. 2017b. Graph-structured representations for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Tan Thongtan and Tanasanee Phienthrakul. 2019. Sentiment classification using document embeddings trained with cosine similarity. In *ACL SRW*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Aaron Steven White, Pushpendre Rastogi, Kevin Duh, and Benjamin Van Durme. 2017. Inference is everything: Recasting semantic resources into a unified evaluation framework. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 996–1005.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *International conference on machine learning*, pages 2397–2406. PMLR.

Huijuan Xu and Kate Saenko. 2016. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pages 451–466. Springer.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015a. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015b. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*.

Song Xu, Haoran Li, Peng Yuan, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. Self-attention guided copy mechanism for abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1355–1362.

Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. 2019a. Auto-encoding scene graphs for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10685–10694.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola. 2015. Stacked attention networks for image question answering. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–29.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2018. Exploring visual relationship for image captioning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 684–699.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B Tenenbaum. 2018a. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *arXiv preprint arXiv:1810.02338*.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B Tenenbaum. 2018b. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *arXiv preprint arXiv:1810.02338*.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*, 1:12.

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *Advances in Neural Information Processing Systems*, 32:11983–11993.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5579–5588.

Zhilu Zhang and Mert R Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *32nd Conference on Neural Information Processing Systems (NeurIPS)*.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020. Sg-net: Syntax-guided machine reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9636–9643.

Yiwu Zhong, Jing Shi, Jianwei Yang, Chenliang Xu, and Yin Li. 2021. Learning to generate scene graph from natural language supervision. *arXiv preprint arXiv:2109.02227*.

C Lawrence Zitnick and Devi Parikh. 2013. Bringing semantics into focus using visual abstraction. In *CVPR*.

A Appendix

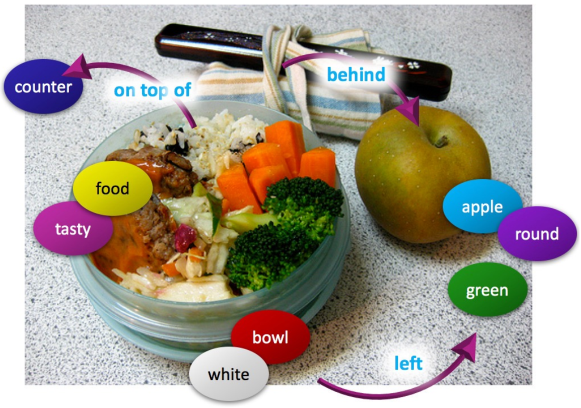
A.1 Visual Question Answering dataset

To address the problem of visual question answering, a number of visual question answering datasets have been developed. The comparison of them is shown in Table 4. The VQA dataset (Antol et al., 2015) is developed on real images in MS COCO (Lin et al., 2014) and abstract scene images

Table 4: Comparison of VQA datasets

	Source of images	# images	# QA pairs	Answer type	Evaluation metrics
DAQUAR	NYU-Depth V2	1,449	12,468	Open	Accuracy&WUPS
VQA	COCO	204K	614K	Open/MC	Accuracy
VQA v2	COCO	204K	1.1M	Open/MC	Accuracy
COCO-QA	COCO	123K	118K	Open/MC	Accuracy
CLEVR	Generated	100K	999K	Open	Accuracy
GQA	Visual Genome	113K	22M	Open	Accuracy

in Antol et al. (2014); Zitnick and Parikh (2013). The question-answer pairs are created by human annotators who are encouraged to ask “interesting" and “diverse" questions. VQA v2 (Goyal et al., 2017) is extended from the VQA (Antol et al., 2015) dataset to achieve more balance between visual and textual information by collecting complementary images in a way that each question is associated with a pair of similar images with different answers; In the COCO-QA (Ren et al., 2015a) dataset, the question-answer pairs are automatically generated from image captions based on syntactic parsing and linguistic rules; DAQUAR (Malinowski and Fritz, 2014) is built on top of the NYU-Depth V2 dataset (Silberman et al., 2012) which contains RGBD images of indoor scenes. DAQUAR consists of (1) synthetic question-answer pairs that are automatically generated based on textual templates and (2) human-created question-answer pairs produced by five annotators; CLEVR (Johnson et al., 2017) is a dataset developed on rendered images of spatially related objects (including cube, sphere, and cylinder) with different sizes, materials, and colors. The locations and attributes of objects are annotated for each image. The questions are automatically generated from the annotations; GQA is a new dataset for real-world visual reasoning and compositional question answering, seeking to address key shortcomings of previous VQA datasets. Considering questions in GQA are most objective, unambiguous, compositional, and can be answered by reasoning only on the visual content. We mainly use the GQA dataset in this work as it best fits our goal of reasoning. We also evaluate our methods on the VQA v2 dataset as it is the most common and general VQA dataset so far.



Is the ***bowl*** *to the right of the* ***green apple****?*
What type of ***fruit*** *in the* ***image*** *is* ***round****?*
What color is the ***fruit*** *on the right side, red or* ***green****?*
Is there any ***milk*** *in the* ***bowl*** *to the left of the* ***apple****?*

Figure 6: Examples from the GQA dataset for visual reasoning and compositional question answering.

Figure 7: Examples from the VQA v2 dataset for Visual Question Answering.

Retrieval Enhanced Data Augmentation for Question Answering on Privacy Policies

Md Rizwan Parvez[§], **Jianfeng Chi**[†], **Wasi Uddin Ahmad**[§],
Yuan Tian[§], **Kai-Wei Chang**[§]
[§]University of California, Los Angeles, [†]University of Virginia
[§]`{rizwan,wasiahmad,yuant,kwchang}@ucla.edu`, [†]`jc6ub@virgina.edu`

Abstract

Prior studies in privacy policies frame the question answering (QA) task as identifying the most relevant text segment or a list of sentences from a policy document given a user query. Existing labeled datasets are heavily imbalanced (only a few relevant segments), limiting the QA performance in this domain. In this paper, we develop a data augmentation framework based on ensembling retriever models that captures the relevant text segments from unlabeled policy documents and expand the positive examples in the training set. In addition, to improve the diversity and quality of the augmented data, we leverage multiple pre-trained language models (LMs) and cascade them with noise reduction filter models. Using our augmented data on the PrivacyQA benchmark, we elevate the existing baseline by a large margin (10% F1) and achieve a new state-of-the-art F1 score of 50%. Our ablation studies provide further insights into the effectiveness of our approach.

1 Introduction

Privacy policies describe how service providers collect, manage, and use their users' data. Understanding them is crucial for users as they can determine if the conditions outlined are acceptable. Policy documents, however, are lengthy, verbose, equivocal, and hard to understand (McDonald and Cranor, 2008; Reidenberg et al., 2016). Consequently, they are often ignored and skipped by users (Commission et al., 2012; Gluck et al., 2016).

Building question answering (QA) systems for privacy policies is a stepping stone to allow users to ask questions about their rights. Prior works (Harkous et al., 2018; Ravichander et al., 2019) framed the QA task as a sentence selection task, essentially a binary classification task that identifies if a policy text segment is relevant to a question. Since policy documents consist of many sentences and typically a few are relevant to a question,[1] the classification data is imbalanced. In this work, we attempt to mitigate the data imbalance by augmenting positive QA examples. Specifically, we develop automatic retrieval models to supplement relevant policy sentences for each user query. We keep the queries unchanged as they are usually limited to a few forms only (Wilson et al., 2016).

Segmented policy document $\boldsymbol{S}$
($\boldsymbol{s}_1$) We do not sell or rent your personal information to third parties for their direct marketing purposes without your explicit consent. ($\boldsymbol{s}_n$) ...We will not let any other person, including sellers and buyers, contact you, other than through your ...
Queries $\boldsymbol{I}$ annotating the red segment as irrelevant
($\boldsymbol{i}_1$) How does Fiverr protect freelancers' personal information? ($\boldsymbol{i}_2$) What type of identifiable information is passed between users on the platform?
Queries $\boldsymbol{R}$ annotating the red segment as relevant
($\boldsymbol{r}_1$) What are the app's permissions? ($\boldsymbol{r}_2$) What type of permissions does the app require?
Queries $\boldsymbol{D}$ that annotators disagree about relevance
($\boldsymbol{d}_1$) Do you sell my information to third parties? ($\boldsymbol{d}_2$) Is my information sold to any third parties?

Table 1: QA (sentence selection) from a policy document $\boldsymbol{S}$. **Sensitive**: For queries $\boldsymbol{R}$ and $\boldsymbol{I}$, annotators at large tagged sentence $\boldsymbol{s}_1$ as relevant, and irrelevant respectively. On the other hand, sentence $\boldsymbol{s}_n$, though analogous to $\boldsymbol{s}_1$ in meaning, was never tagged as relevant. **Ambiguous**: For queries $\boldsymbol{D}$, experts interpret $\boldsymbol{s}_1$ differently and disagree on their annotations.

Unlike other domains, augmenting privacy policy statements is very challenging. First, they often describe similar information (Hosseini et al., 2016). Thus, their annotations are sensitive to small changes in the text (see Table 1) which may not be tackled using the existing augmentation methods based on data synthesis (e.g., mixup (Zhang et al., 2018), back-translation (Edunov et al., 2018)). For

---

[1] PrivacyQA (Ravichander et al., 2019) dataset has 1,350 questions with an average number of answer sentences is 5, while the average length of policy documents is 138 sentences.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 201–210
May 2-6, 2023 ©2023 Association for Computational Linguistics

example, Kumar et al. (2020) identifies that even linguistically coherent instances augmented via generative models do not preserve the class labels well.[2] Hence, to reflect the data properties, we consider a retrieval-based approach to augment the raw policy statements. Given a pre-trained LM and a small QA dataset, we first build a dense sentence retriever (Karpukhin et al., 2020). Next, leveraging an unlabeled policy corpus with 0.6M sentences crawled from web applications, we perform a coarse one-shot sentence retrieval for each query in the QA training set. To filter the noisy candidates retrieved,[3] we then train a QA model (as a filter model) using the same pre-trained LM and data and couple it with the retriever.

Second, privacy policies are ambiguous; even skilled annotators dispute their diverse interpretations, e.g., for at least 26% questions in PrivacyQA, experts disagree on their annotations (see Table 1). Therefore, a single retriever model may not capture all sorts of relevant policy segments written in various diversified ways. To combat this insufficient data diversity, we propose a novel retriever ensemble technique. Different pre-trained models learn distinct language representations due to their pre-training objectives, and hence, retriever models built on them can retrieve a disjoint set of candidates (verified in Section 3). Therefore, we build our retrievers and filter models based on multiple different pre-trained LMs (See Figure 1). Finally, we train a user-defined QA model on the aggregated corpus using them.

We evaluate our framework on the PrivacyQA benchmark. We elevate the state-of-the-art performance significantly (10% F1) and achieve a new one (50% F1). Furthermore, our ablation studies provide an insightful understanding of our model. We will release all data and code upon acceptance.

2 Methodology

The privacy policy QA is a binary classification task that takes a user query q, a sentence p from policy documents and outputs a binary label $z \in \{0, 1\}$ that indicates if q and p are relevant or not. As most sentences p are labeled as negative, our goal is to retrieve relevant sentences to augment the training data and mitigate the data imbalance issue. Given a

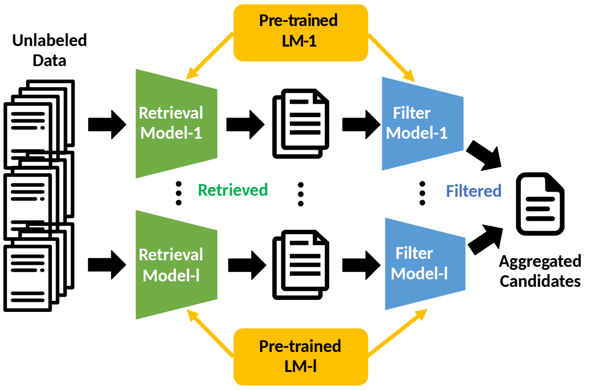


Figure 1: Our framework. Given a pre-trained LM, we train (i) a retriever, (ii) a QA model (filter) both on the small-size labeled data. From an unlabeled corpus, we first, retrieve the coarse relevant sentences (positive examples) for the queries in the training set and use the filter model to filter out noisy ones. We repeat this for multiple different pre-trained LMs. Finally, we aggregate them to expand the positive examples in the training set and learn any user-defined final QA model.

QA training dataset $D = \{(q_i, p_i, z_i)\}_{i=1}^{m}$, for each question in D, we (1) retrieve positive sentences from a large unlabeled corpus. (2) filter the noisy examples using filter models and aggregate final candidates. The final candidates are combined with the base data D to train the end QA model. We use an ensemble of retrievers and filter models built upon various pre-trained LMs throughout the whole process. Details are discussed in the following.

Retriever. Our retriever module is built upon the Dense Passage Retriever (DPR) model (Karpukhin et al., 2020). It consists of two encoders $Q(\cdot)$ and $P(\cdot)$ that encode the queries and the policy sentences, respectively. The relevance of a query q and a policy sentence p is calculated by the dot product of $Q(q)$ and $P(p)$, i.e., $\text{sim}(q, p) = Q(q)^T \cdot P(p)$. We train a retriever $\mathcal{R}_L$ on D, where the encoders in $\mathcal{R}_L$ are initialized with a pre-trained LM L. At inference, $\mathcal{R}_L$ retrieves the top-k most relevant policy sentences from an unlabeled corpus of policy sentences $\mathcal{P} = \{p_1, \ldots, p_M\}$ for each query q_i in D, i.e., $\mathcal{R}_L(\{q_i\}_{i=1}^{m}, \mathcal{P}, k) = \{(q_i, p_j, 1) : i \in [m], p_j \in \mathcal{P}_{\text{top}}(q_i, k)\}$, where $\mathcal{P}_{\text{top}}(q_i, k) := \arg\max_{\mathcal{P}' \subset \mathcal{P}, |\mathcal{P}'|=k} \sum_{p \in \mathcal{P}'} \text{sim}(q_i, p)$.

Filtering Model. To filter out the misclassified retrievals from $\mathcal{R}_L(\{q_i\}_{i=1}^{m}, \mathcal{P}, k)$, we train a QA (i.e., a text-classification) model ($\mathcal{Q}_L$) as a filter to predict whether a query q and a (retrieved) policy sentence p are relevant or not (i.e., $\mathcal{Q}_L(q, p) \in \{0, 1\}$). Note that, the retriever model is a bi-encoder model that can pre-encode, index,

---

[2] To verify, in our preliminary study, for each positive example in the PrivacyQA training set, we augment a new synthesized positive example by en-zh-en back-translation using Google Translator API, and the performance drops by 3% F1.

[3] We refer the misclassified candidates as noisy retrievals.

and rank a large number of candidates. In contrast, our filtering model is a single cross-encoder model that can achieve comparatively higher performances (Humeau et al., 2019) (i.e., hence better as a filter) but can not pre-encode and hence can not be used for large-scale retrieval (more differences are below and in Appendix F). Once the retrieval is done, $\mathcal{Q}_L$ serves as an inexpensive binary classifier which naturally suits as a filter and fit into the pipeline. We verify the effectiveness of our filtering in Section 3.4. We denote retrieval outputs after filtering as $\mathcal{D}_L = \{(q,p,1) : \mathcal{Q}_L(q,p) = 1, \forall (q,p,1) \in \mathcal{R}_L(\{q_i\}_{i=1}^m, \mathcal{P}, k)\}$.

Training of $\mathcal{Q}_L$ and $\mathcal{R}_L$. Both the single encoder in $\mathcal{Q}_L$ and the two encoders in $\mathcal{R}_L$ are initialized with the same pre-trained LM L (e.g., BERT). Both are then fine-tuned as a binary classifier using the paired (query and policy is relevant or not) training data D. Additionally, to better-train the relatively weak bi-encoder $\mathcal{R}_L$, we consider the in-batch negative examples schema (Henderson et al., 2017; Parvez et al., 2021) and its hyper-parameters are tuned using mean ranking or mean reciprocal ranking (MRR) loss. At inference, for query q and candidate p, raw scores from $\mathcal{R}_L$ is used to rank and prediction $\{0,1\}$ from $\mathcal{Q}_L$ is used to filter p.

Ensemble. In order to enhance the diversity and the quality of the retrieved candidates, we use a set of pre-trained LMs $\mathcal{L} = \{L_1, \ldots, L_l\}$ and aggregate all the corresponding retrieved corpora, $\mathcal{D}_{\text{aug}} = \bigcup_{L \in \mathcal{L}} \mathcal{D}_L$. In Section 3, we show that retrieved corpora using multiple pre-trained LMs with different learning objectives can bring a different set of relevant candidates. Lastly, we aggregate $\mathcal{D}_{\text{aug}}$ with D (i.e., final train corpus $\mathcal{T} = \mathcal{D}_{\text{aug}} \cup D$) and train our final QA model with user specifications (e.g., architecture, pre-trained LM).

3 Experiments

3.1 Setup

Evaluation Metrics. We evaluate our approach on PrivacyQA that is framed as a text classification task (Ravichander et al., 2019). We use *precision*, *recall*, and *F1 score* as the evaluation metrics.

Implementations. As for the retrieval database $\mathcal{P}$, we crawl privacy policies from the most popular mobile apps spanning different app categories in the Google Play Store and end up with 6.5k documents (0.6M statements). By default, all retrievals use top-10 candidates w/o filtering. All data/models/codes are implemented using (i) Huggingface Transformers (Wolf et al., 2019), (ii) DPR (Karpukhin et al., 2020) libraries.

Method	F	Precision	Recall	F1
Human	-	68.8	69.0	68.9
W/o data augmentation				
BERT+Unans.	-	44.3	36.9	39.8
BERT (reprod)		$48.0_{\pm 2.0}$	$37.7_{\pm 1.2}$	$42.2_{\pm 1.5}$
PBERT		$51.2_{\pm 0.4}$	$42.7_{\pm 0.6}$	$46.6_{\pm 0.4}$
SimCSE		$48.4_{\pm 0.8}$	$41.4_{\pm 0.7}$	$44.7_{\pm 0.7}$
Retriever augmented				
BERT-R	✗	$39.0_{\pm 0.8}$	$52.4_{\pm 1.7}$	$44.7_{\pm 0.4}$
	✓	$48.1_{\pm 1.4}$	$44.7_{\pm 0.9}$	$46.3_{\pm 0.5}$
PBERT-R	✗	$48.7_{\pm 1.9}$	$44.1_{\pm 1.8}$	$46.3_{\pm 1.6}$
	✓	$49.2_{\pm 1.6}$	$44.9_{\pm 2.0}$	$47.0_{\pm 1.2}$
SimCSE-R	✗	$47.0_{\pm 2.1}$	$44.5_{\pm 2.4}$	$45.7_{\pm 1.9}$
	✓	$48.6_{\pm 2.2}$	$43.9_{\pm 1.2}$	$46.1_{\pm 1.6}$
Ensemble retriever augmented				
Baseline-E	✗	$22.2_{\pm 0.8}$	$54.4_{\pm 0.8}$	$31.4_{\pm 0.8}$
ERA	✓	$47.4_{\pm 0.6}$	$50.5_{\pm 2.2}$	$\mathbf{48.9}_{\pm 0.8}$
ERA-D	✓	$51.0_{\pm 0.4}$	$48.7_{\pm 0.9}$	$\mathbf{49.8}_{\pm 0.7}$

Table 2: Test performances on PrivacyQA ($\text{mean}_{\pm\text{std}}$). F indicates filtering and BERT+Unans. refers to the previous SOTA performance (Ravichander et al., 2019). Retrieved candidates improve all the baseline QA models, especially when being filtered. Our ensemble retriever approach combines them and achieves the highest gains.

Baselines. We fine-tune three pre-trained LMs on PrivacyQA as baselines: (i) *BERT*: Our first baseline is BERT-base-uncased (Devlin et al., 2019) which is pre-trained on generic NLP textual data. A previous implementation achieves the existing state-of-the-art performance (BERT+Unams. in Ravichander et al. (2019)). (ii) *PBERT*: We adapt *BERT* to the privacy domain by fine-tuning it using masked language modeling on a corpus of 130k privacy policies (137M words) collected from apps in the Google Play Store (Harkous et al., 2018). Note that the retrieval database $\mathcal{P}$ is a subset of this data that is less noisy and crawled as a recent snapshot (more in Appendix E) (iii) *SimCSE*: We take the *PBERT* model and apply the unsupervised contrasting learning SimCSE (Gao et al., 2021) model on the same 130k privacy policy corpus. We also consider three other retrieval augmented QA models based on individual pre-trained LM without ensemble: (iv) *BERT-R*: $\mathcal{L} = \{BERT\}$, (v) *PBERT-R*: $\mathcal{L} = \{PBERT\}$, (vi) *SimCSE-R*: $\mathcal{L} = \{SimCSE\}$. We first construct $\mathcal{T}$ (both settings: w/ and w/o filter model) and fine-tune on it the corresponding pre-trained LM as the final QA model. Finally, we

Query Type	%	B	PB	S	ERA
Data Collection	42	45	**46**	**46**	**48**
Data Sharing	25	**43**	37	41	**43**
Data Security	11	**65**	61	60	60
Data Retention	4	**52**	35	35	**56**
User Access	2	**72**	48	31	61
User Choice	7	41	**60**	42	31
Others	9	36	45	**52**	**55**
Overall	100	45	47	48	**49**

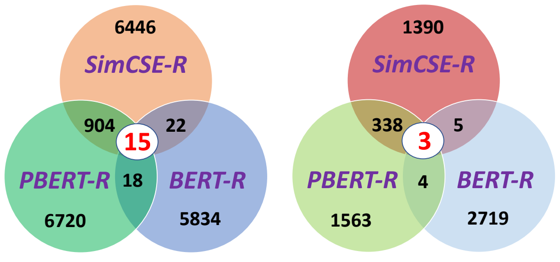
Table 3: F1-score breakdown (values are in Appendix D). B, PB, S refers to retrievers *BERT-R*, *PBERT-R*, and *SimCSE-R*. Different models performs better for different types (black-bold). ERA combines them and enhances performances for all categories (except: red).

consider one more ensemble retrieval augmented baseline (vii) *Baseline-E*, which is precisely the same as ours (settings below), except there are no intermediate filtering models.

Ours. We construct the augmented corpus $\mathcal{T}$ using (i) all 3 aforementioned pre-trained LMs: $\mathcal{L} = \{\mathit{BERT}, \mathit{PBERT}, \mathit{SimCSE}\}$ (ii) domain adapted models only: $\mathcal{L} = \{\mathit{PBERT}, \mathit{SimCSE}\}$. For brevity, we call them: *Ensemble Retriever Augmentation* (ERA) and *Ensemble Retriever Augmentation–Domain Adapted* (ERA-D). By default, we fine-tune *SimCSE* as the final QA model.

3.2 Main Results

The results are reported in Table 2. Overall, domain adapted models *PBERT* and *SimCSE* excel better than the generic *BERT* model. The retrieval-augmented models enhance the performances more, especially the recall score, as they are added as additional positive examples. However, these models may contain noisy examples (see Table 4), which lowers precision. Filtering these examples leads to improved precision for all retrievers. Finally, ERA and ERA-D aggregate these high-quality filtered policies–leading toward the highest gain (10% F1 from the previous baseline) and a new state-of-the-art result with an F1 score of ∼50. Note that *Baseline-E* unifies all the candidates w/o any filtering performs considerably worse than all other models, including each individual retrieval model: *Baseline-E* augments more candidates as positives, which explains the highest recall score; in the meantime, as it does not filter any, the corresponding precision score is oppositely the lowest.



(a) (b)

Figure 2: Venn diagram of low mutual agreement (<1%) among retrievers (a); even amplified after filtering (b).

3.3 Analysis

Table 3 shows the performance breakdown for different query types (more in Appendix D). For questions related to data collection, data sharing, and data security, the performance difference among the models is relatively small (≤ 5% F1); for data retention and user access, *BERT-R*, that is pre-trained on generic NLP texts, performs significantly well (> 15% F1), possibly because the answers to these query types focus on providing numerical evidence for the questions (e.g., How many days the data are retained?) that is less relevant to the domain of privacy policies; and for other types of questions the domain adapted models performs better (> 15% F1). Overall, the individual retrieval augmented models based on LM pre-trained w/ different corpora and objectives perform at different scales for each type, and combining their expertise, ERA enhances the performances for all types.

Next, we show the Venn diagram of overlapping retrievals in Figure 2. Although policy statements describe similar information (i.e., have common phrases), they are often verbose and equivocal (i.e., multiple-different interpretations). Consequently, retrievers w/ different objectives and training corpora rank them differently. Therefore, although being retrieved from the same corpus, candidates retrieved by different models rarely match fully but may have notable overlapping information (words/phrases) and improve their performances equitably. For example, while the performances of *BERT-R*, *PBERT-R* and *SimCSE-R* w/ filtering are similar (∼46) in Table 2, from Figure 2 their overlapping (exact match) is < 1% (qualitative examples in Appendix G). At the same time, their raw retrieval corpora have a high BLEU score of (≥ 0.78). This validates our hypothesis that retrievers built upon different pre-trained LMs learn distinct representations and hence retrieve diverse candidates.

3.4 Ablation Study

Sampling to tackle data imbalance. The preliminary experiments studied rebalancing techniques like equal sampling, but oversampling does not add new information and undersampling limits data, leading to poor generalization on unseen test data. Using equal positive and negative sub-sampled training instances resulted in a 9% drop in F1 score. Even augmented with a higher number of filtered positive examples retrieved by a single retriever model does not perform as well as when a lower number of ensemble-based positive instances are augmented. From Table 8 in Appendix, augmenting with a high number of filtered positive examples from a single retriever model performed worse than using a lower number of ensemble-based positive examples–suggesting the need for diverse and high-quality knowledge not present in training data.

A common filter. Performances of ERA (last row in Table 3) with a common filter model based on *SimCSE* for all the retrievers regardless of their corresponding pre-trained models are 49.2, 45.2, and 47.1, respectively–validating the requirement of filtering using the corresponding pre-trained LM.

Other pre-trained LM as the final QA model. Fine-tuning *PBERT* instead of *SimCSE* on $\mathcal{T}$ (last two rows in Table 2) becomes: 47.0, 47.1, 47.0 and 51.0, 45.9, 48.3, respectively–showing that ERA is generic to end model choices.

Which pre-trained LMs to use? Table 3 shows ERA-D that combines fewer pre-trained LMs can outperform the one with more models, ERA. Though here we consider a simple approach (in-domain) for selecting the potential subset of models, this paves a new direction for future research (e.g., Parvez and Chang (2021)).

Recall performances on PrivacyQA dataset. Example retriever *BERT-R* scores the recall@k (k up to 10) values as 17, 28, 36, 42, 48, 53, 58, 59, 63, 67 respectively while the (cross-encoder) BERT QA model (i.e., filter model) achieves a recall (same as recall@1) of ~ 37. This shows the effectiveness of our designed filter model.

Can the final QA model be used as a filter and impact of filter models on the end performance? The final QA model can be used as a filter model. As for the single retriever model-based augmentation, the performance of the downstream end task depends on the performance of the retriever model, the filtering model, and the end QA model. A stronger filter model leads to better end QA performance in general. With a *BERT-R* retriever and *BERT* end QA model, the use of a *PrivacyBERT* filter model improves that of a *BERT* filter model from 46.3 to 46.8. However, when doing the ensembling using a combination of retriever and filter models from the same pre-trained language model (PLM) results in even better performance than using a stronger filter model from different PLMs.

Q: do you sell my photos to anyone?
Gold: i) We use third-parties to serve ads on our behalf across the Internet. (ii) We may share personal information within our family of brands. (iii) From time to time we share the personal information we collect with trusted companies who work with or on our behalf. (iv) No personally identifiable information is collected in this process.
Correct Retrievals: (i) *SimCSE-R*: The Application does not collect or transmit personally identifiable information such as your name, address, phone number or email address. (ii) *PBERT-R*: We also use the Google AdWords to serve ads on our behalf across the Internet and sometimes on this Website. (iii) *BERT-R*: To organ and tissue donation requests: By law, we can disclose your health information to organ procurement organizations.
Incorrect Retrievals: (i) *BERT-R*: These are not linked to any information that is personally identifiable. (ii) *SimCSE-R*: When you upload photos to our platform or give us permission to access the photos on your device, your photo content may also include related information such as the time and place your photo was taken and similar "meta-data" captured by your image capture device.

Table 4: Example retrieved policies. Retrieved candidates are distinct from expert annotated ones and can bring auxiliary knowledge to the model. Filtering is needed as inappropriate candidates can also be retrieved.

Qualitative examples. Table 4 shows some example retrievals of different models. Retrieved candidates are distinct from expert annotated ones and can bring auxiliary knowledge to the model.

4 Conclusion

We develop a noise-reduced retrieval-based data augmentation method that combines different pre-trained language models to address the data imbalance issue in privacy policy QA. However, our approach can possibly be adapted to other domains and we leave the exploration as the future work.

Limitations

In this work, we develop a retrieval-augmented QA framework specifically for Privacy Policies. Its effectiveness rooted on the characteristics of disjoint retrievals from different pre-trained language models (PLMs). Although this focused work completely aligns, addresses and adheres to the guidelines for a short-paper in this venue, we have not performed any experiments on data outside this privacy policy domain. Hence, the applicability of our method for any generic data domain is unknown. While the time/latency and resource utilization remains unchanged at inference-time, using multiple retriever modules our method introduce an additional overhead in model training. Using different implementation, PLMs, and random seeds may also lead to results that could be different from ours.

Ethics Statement

In this work, our approach crawls an unlabeled privacy policy corpus from the web policy documents specifically from the Google play store which we use as a retrieval database. Although these documents are completely publicly available and was used only for research purpose, they may contain some nomenclature of certain persons, objects, products, users, developers, or production houses (i.e., industries). We neither obfuscate nor make any altercation/modification to them.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. This work was supported in part by National Science Foundation Grant OAC 2002985, OAC 1920462 and CNS 1943100, Google Research Award, and Meta Research Award. Any opinions, findings, conclusions, or recommendations expressed herein are those of the authors and do not necessarily reflect those of the US Government or NSF.

References

Wasi Ahmad, Jianfeng Chi, Tu Le, Thomas Norton, Yuan Tian, and Kai-Wei Chang. 2021. Intent classification and slot filling for privacy policies. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4402–4417.

Wasi Ahmad, Jianfeng Chi, Yuan Tian, and Kai-Wei Chang. 2020. PolicyQA: A reading comprehension dataset for privacy policies. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 743–749.

Leo Breiman. 1996. Bagging predictors. *Machine learning*, 24(2):123–140.

Duc Bui, Kang G Shin, Jong-Min Choi, and Junbum Shin. 2021. Automated extraction and presentation of data practices in privacy policies. *Proceedings on Privacy Enhancing Technologies*, 2021(2):88–110.

Federal Trade Commission et al. 2012. Protecting consumer privacy in an era of rapid change. *FTC report*.

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joshua Gluck, Florian Schaub, Amy Friedman, Hana Habib, Norman Sadeh, Lorrie Faith Cranor, and Yuvraj Agarwal. 2016. How short is too short? implications of length and framing on the effectiveness of privacy notices. In *Twelfth Symposium on Usable Privacy and Security ({SOUPS} 2016)*.

Hamza Harkous, Kassem Fawaz, Rémi Lebret, Florian Schaub, Kang G Shin, and Karl Aberer. 2018. Polisis: Automated analysis and presentation of privacy policies using deep learning. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 531–548.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Mitra Bokaei Hosseini, Sudarshan Wadkar, Travis D Breaux, and Jianwei Niu. 2016. Lexical similarity of information type hypernyms, meronyms and synonyms in privacy policies. In *2016 AAAI Fall Symposium Series*.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Moniba Keymanesh, Micha Elsner, and Srinivasan Parthasarathy. 2021. Privacy policy question answering assistant: A query-guided extractive summarization approach. *arXiv preprint arXiv:2109.14638*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26.

Aleecia M McDonald and Lorrie Faith Cranor. 2008. The cost of reading privacy policies. *Isjlp*, 4:543.

Md Rizwan Parvez, Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2021. Retrieval augmented code generation and summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2719–2734, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Md Rizwan Parvez, Tolga Bolukbasi, Kai-Wei Chang, and Venkatesh Saligrama. 2019. Robust text classifier on test-time budgets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1167–1172.

Md Rizwan Parvez and Kai-Wei Chang. 2021. Evaluating the values of sources in transfer learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5084–5116.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Abhilasha Ravichander, Alan W Black, Thomas Norton, Shomir Wilson, and Norman Sadeh. 2021. Breaking down walls of text: How can NLP benefit consumer privacy? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4125–4140, Online. Association for Computational Linguistics.

Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. 2019. Question answering for privacy policies: Combining computational and legal perspectives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4947–4958.

Joel R Reidenberg, Jaspreet Bhatia, Travis D Breaux, and Thomas B Norton. 2016. Ambiguity in privacy policies and the impact of regulation. *The Journal of Legal Studies*, 45(S2):S163–S190.

Hoang Van, Vikas Yadav, and Mihai Surdeanu. 2021. Cheap and good? simple and effective data augmentation for low resource machine reading. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2116–2120.

Shomir Wilson, Florian Schaub, Aswarth Abhilash Dara, Frederick Liu, Sushain Cherivirala, Pedro Giovanni Leon, Mads Schaarup Andersen, Sebastian Zimmeck, Kanthashree Mysore Sathyendra, N. Cameron Russell, Thomas B. Norton, Eduard Hovy, Joel Reidenberg, and Norman Sadeh. 2016. The creation and analysis of a website privacy policy corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1330–1340.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yinfei Yang, Ning Jin, Kuo Lin, Mandy Guo, and Daniel Cer. 2020. Neural retrieval for question answering with cross-attention supervised data augmentation. *arXiv preprint arXiv:2009.13815*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Sebastian Zimmeck, Peter Story, Daniel Smullen, Abhilasha Ravichander, Ziqi Wang, Joel Reidenberg, N Cameron Russell, and Norman Sadeh. 2019. Maps: Scaling privacy compliance analysis to a million apps. *Proceedings on Privacy Enhancing Technologies*, 2019(3):66–86.

Supplementary Material: Appendices

A Related Works

A line of works focuses on using NLP techniques for privacy policies (Wilson et al., 2016; Harkous et al., 2018; Zimmeck et al., 2019; Bui et al., 2021; Ahmad et al., 2021). Besides the QA tasks as sentence selection, Ahmad et al. (2020) propose another SQuAD-like (Rajpurkar et al., 2016) privacy policy reading comprehension dataset for a limited number of queries. Oppositely, we focus on the more challenging one, which allows unanswerable questions and "non-contiguous" answer (Ravichander et al., 2021). In relevant literature works, retrieval augmented methods are applied in various contexts including privacy policies (e.g., Van et al. (2021); Keymanesh et al. (2021); Yang et al. (2020)). Non-retrieval data aggregation has also been studied under different NLP contexts (e.g., bagging (Breiman, 1996), meta learning (Parvez et al., 2019)). However, we uniquely aggregate the retriever outputs using different pre-trained language models.

B Limitations/Reproduction

In this paper, we show that leveraging multiple different pre-trained LMs can augment high-quality training examples and enhance the QA (sentence selection) task on privacy policies. Our approach is generic and such unification of different kinds of pre-trained language models for text data augmentation can improve many other low-resourced tasks or domains. However, it is possible that our approach:

- may not work well on other scenarios (e.g., domains/language or tasks etc.,).
- subject to the choice of a particular set of models. For example, as mentioned in Section, 3.4, fine-tuning pre-trained models other than *SimCSE* (Gao et al., 2021) as the final QA model achieve lower gain.
- may not work for certain top-k retrievals. For example, from Table 8, we get different results with different scales for variable top-k values (e.g., top-10, top-100).
- uses the same set of hyperparameters for all:
  - QA model:
    - learning rate: $2e^{-5}$,
    - train epoch: 4,
    - per gpu train batch size: 31,
    - num gpus: 4
    - fp16 enabled
    - others: mostly default as in Huggingface
    - train time: around 2 hours
    - Higgingface transformer version 0.3.2. (it has Apache License 2.0)
  - Retriever model:
    - learning rate: $2e^{-5}$,
    - train batch size: 16,
    - train epoch: 100,
    - global_loss_buf_sz 600000,
    - others: mostly default as in DPR (It has Attribution-NonCommercial 4.0 International license)
    - num gpus: 3
    - Higgingface transformer version 0.3.2 (it has Apache License 2.0)
    - train time: around 12-18 hours

As our primary goal is on the retrieval-based data augmentation technique, we expect further optimization of task-specific model hyperparameters to improve performance. Note that our results are based on upto 4 runs using different random seeds.

C Privacy Policy Data Crawling & Retrieval Statistics

We crawl our English retrieval corpus from Google App Store using the Play Store Scraper[4].

Query Type	No. of Retrieval
Data Collection	2893
Data Sharing	1848
Data Security	891
Data Retention	542
User Access	145
User Choice	335
Others	14

Table 5: Retrieval statistics per query type.

However, below is the statistics of our (ERA) augmented corpus per each question category in the PrivacyQA training set.

D PrivacyQA Dataset and Breakdown of Performance in Absolute Numbers

Table 6 shows the accuracy breakdowns in absolute numbers. In addition, A brief summary of the

---

[4] https://github.com/danieliu/play-scraper

Query Type	total	B	PB	S	ERA
Data Collection	6280	1901	1157	1186	1806
Data Sharing	4734	1332	777	1092	1268
Data Security	994	416	399	423	393
Data Retention	453	150	98	110	173
User Access	221	89	47	43	87
User Choice	493	91	49	24	55
Others	28	2	1	2	4
Overall	10332	3135	2084	2334	2935

Table 6: Number of correct predictions. Note that F1-score is not proportional to the accuracy. B, PB, S refers to retrievers *BERT-R*, *PBERT-R*, and *SimCSE-R*.

	PrivacyQA
Source	Mobile application
Question annotator	Mechanical Turkers
Form of QA	Sentence selection
Answer type	A list of sentences
# Unique policy docs	train: 27, test: 8
# Unique questions	train: 1350, test: 400
# QA instances	train: 185k, test: 10k
Avg Q. Length	train: 8.42 test: 8.56
Avg Doc. Length	train: 3.1k, test: 3.6k
Avg Ans. Length	train: 124, test: 153

Table 7: Brief summary of PrivacyQA.

PrivacyQA benchmark is in Table 7. Note that the questions in Privacy can be categorized into a few OPP-115 classes. These categories are enlisted in Table 3 in the main paper and the details of each category can be found in Wilson et al. (2016).

E Difference Between Pre-training and Retrieval Corpus

130k documents were collected before 2018 and by that time, the GDPR[5] and CCPA[6] were not enforced by then. Thus, the 130k documents are out-of-date and some content might not be comprehensive as the retrieval corpus. Besides, the 130k documents provided by (Harkous et al., 2018) contains some noises since we observe that the documents are not all written in English. However, as the data size is larger, we still use it for pre-training. In contrast, our corpus was collected after 2020 and we filtered out some possible noises (e.g., filtering out non-English document) while crawling.

F Difference Between the Filtering Model and the Retriever

The retriever model is a bi-encoder model whose model parameters are fine-tuned with in-batch negative loss (discussed in Section 2 in the main paper), hyper-parameters are tuned based on average rankings (`https://github.com/facebookresearch/DPR/blob/a31212dc0a54dfa85d8bfa01e1669f149ac832b7/train_dense_encoder.py#L294`) and that can pre-encode, index and rank a large number of candidates while our filtering model is a cross-encoder text-classifier (e.g., single encoder fine-tuned BERT) that is fine-tuned w/o any additional in-batch negatives and in-general achieves comparatively higher performance (Humeau et al., 2019) (i.e, better as a filter) but can not pre-encode and hence can not be used for large scale retrieval.

Method	Filter	top-k	Precision	Recall	F1
BERT-R	✗	10	39.9	50.8	44.7
	✓	10	46.5	45.5	46.0
PBERT-R	✗	10	48.4	45.6	46.9
	✓	10	46.9	43.3	45.1
	✗	50	47.8	45.5	46.7
	✓	50	49.5	46.3	47.8
SimCSE-R	✗	10	48.4	47.2	47.8
	✓	10	49.4	44.8	47.0
	✗	100	42.1	41.3	41.7
	✓	100	51.0	45.2	47.9

Table 8: Model performances with and without filtering (i.e., w/o the filter model) with top-k. In general, without filtering, augmenting the retrieved candidates enhances recall but may reduce the precision (and hence may not improve the overall F1). Filtering, however, improves the performance, especially with larger top-k candidates. In above, top-100 augmented examples (~13K total positives after filtering) retrieved by a single retriever perform worse than with top-10 examples by ERA or ERA-D (total 7K or 4K positive examples) reported in Table 2 in the main paper.

---

[5] https://gdpr-info.eu/

[6] https://oag.ca.gov/privacy/ccpa

G More Qualitative Examples

The below tables show some example retrievals of different models. Retrieved candidates are distinct from expert annotated ones and can bring auxiliary knowledge to the model.

Q: do you sell my photos to anyone?
Gold: i) We use third-party service providers to serve ads on our behalf across the Internet and sometimes on the Sites. (ii) These companies may use your personal information to enhance and personalize your shopping experience with us, to communicate with you about products and events that may be of interest to you and for other promotional purposes. iii) Your use of our Application with that healthcare institution may be subject to that healthcare institution's policies and terms. (iv) We may share personal information within our family of brands. (v) From time to time we share the personal information we collect with trusted companies who work with or on behalf of us. (vi) No personally identifiable information is collected in this process. (vii) We use third-party service providers to serve ads on our behalf across the Internet and sometimes on our Sites and Apps.
Correct Retrievals: (i) The Application does not collect or transmit any personally identifiable information about you, such as your name, address, phone number or email address. -(SimCSE-R) (ii) Some of this information is automatically gathered, and could be considered personally identifiable in certain circumstances, however it will generally always be anonymised prior to being viewed by Not Doppler, and never sold or shared. -(BERT-R) (iii) We also use the Google AdWords service to serve ads on our behalf across the Internet and sometimes on this Website. -(PBERT-R) (iv) To organ and tissue donation requests: By law, we can disclose health information about you to organ procurement organizations. -(BERT-R)
Incorrect Retrievals: (i) When you upload your photos to our platform or give us permission to access the photos stored on your device, your photo content may also include related image information such as the time and the place your photo was taken and similar "metadata" captured by your image capture device. -(SimCSE-R) (ii) These are not linked to any information that is personally identifiable.-(BERT-R)

Table 9: A fraction of retrieval examples (i).

Q: who all has access to my medical information?
Gold: i) Apple HealthKit to health information and to share that information with your healthcare providers. ii) Your use of our Application with that healthcare institution may be subject to that healthcare institution's policies and terms.
Correct Retrievals: (i) We may share your information with other health care providers, laboratories, government agencies, insurance companies, organ procurement organizations, or medical examiners. -(SimCSE-R) (ii) Do not sell your personal or medical information to anyone. -(BERT-R) (iii) Lab, Inc will transmit personal health information to authorized medical providers. -(PBERT-R) (iv) To organ and tissue donation requests: By law, we can disclose health information about you to organ procurement organizations. -(BERT-R)
Incorrect Retrievals: (i) However, we take the protection of your private health information very seriously. -(SimCSE-R) (ii) All doctors, and many other healthcare professionals, are included in our database. -(PBERT-R) (iii) You may be able to access your pet's health records or other information via the Sites. -(BERT-R) (iv) will say "yes" unless a law requires us to disclose that health information.-(BERT-R) (v) do not claim that our products "cure" disease.-(BERT-R) (vi) Has no access to your database password or any data stored in your local database on your devices.-(BERT-R)

Table 10: A fraction of retrieval examples (ii).

FastKASSIM: A Fast Tree Kernel-Based Syntactic Similarity Metric

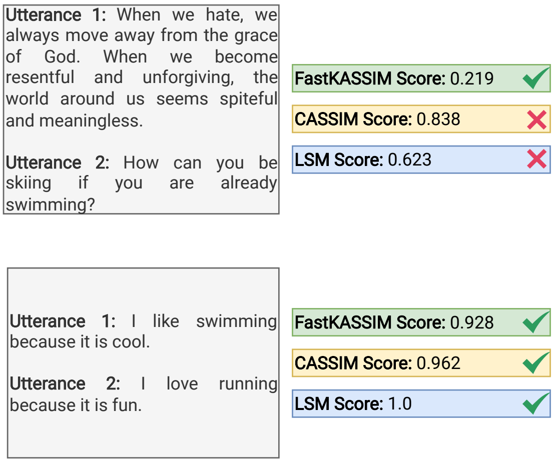
Maximillian Chen[*], Caitlyn Chen[*], Xiao Yu[*], Zhou Yu
Department of Computer Science, Columbia University, New York, NY
`maxchen@cs.columbia.edu`
`{caitlyn.chen, xy2437, zy2461}@columbia.edu`

Abstract

Syntax is a fundamental component of language, yet few metrics have been employed to capture syntactic similarity or coherence at the utterance- and document-level. The existing standard document-level syntactic similarity metric is computationally expensive and performs inconsistently when faced with syntactically dissimilar documents. To address these challenges, we present FastKASSIM, a metric for utterance- and document-level syntactic similarity which pairs and averages the most similar constituency parse trees between a pair of documents based on tree kernels. FastKASSIM is more robust to syntactic dissimilarities and runs up to to 5.32 times faster than its predecessor over documents in the r/ChangeMyView corpus. FastKASSIM's improvements allow us to examine hypotheses in two settings with large documents. We find that syntactically similar arguments on r/ChangeMyView tend to be more persuasive, and that syntax is predictive of authorship attribution in the Australian High Court Judgment corpus.

1 Introduction

Syntax, the form of language, plays a crucial role in all aspects of natural language and communication, whether explicitly or implicitly. In storytelling, writers often have their own styles rooted in different syntactic tendencies (Feng et al., 2012), allowing syntax to become indicators in prediction tasks such as gender (Sarawgi et al., 2011) and authorship (Raghavan et al., 2010) attribution. Syntax also has social connotations in different cultures — for example, in Russia, different social and demographic groups tend to use different syntactic patterns (Bogdanova-Beglarian et al., 2016). Such examples makes syntactic consistency crucial to capture in tasks such as machine translation and dialogue generation so that social conventions are not lost. Yet, recent research focuses primarily on evaluating similarity and coherence in terms of dimensions like semantics, the meaning behind language, (with approaches such as BERT embeddings (Reimers and Gurevych, 2019; Zhang et al., 2019b)), or lexical overlap (e.g., BLEU (Papineni et al., 2002)) — even in work which uses syntax as an input to improve translation quality (Zhang et al., 2019a). A lack of work on syntax can be partially attributed to the absence of a practical, efficient metric that specifically compares syntax at the utterance level. The current standard metric is CASSIM (Boghrati et al., 2016, 2018), but CASSIM uses a computationally expensive distance metric and can yield inconsistencies when comparing syntactically dissimilar documents (e.g., Figure 1). To address this issue, we introduce *FastKASSIM*, a *Fast Tree Kernel-bAsed Syntactic SIMilarity Metric*[1], an improved metric for syntactic similarity at the utterance- and document-level.



Utterance 1: When we hate, we always move away from the grace of God. When we become resentful and unforgiving, the world around us seems spiteful and meaningless.

Utterance 2: How can you be skiing if you are already swimming?

FastKASSIM Score: 0.219 ✓
CASSIM Score: 0.838 ✗
LSM Score: 0.623 ✗

Utterance 1: I like swimming because it is cool.

Utterance 2: I love running because it is fun.

FastKASSIM Score: 0.928 ✓
CASSIM Score: 0.962 ✓
LSM Score: 1.0 ✓

Figure 1: Comparison of FastKASSIM, CASSIM, and Linguistic Style Matching similarity scores. Top: two dissimilar utterances. Bottom: two similar utterances. All three metrics have strong agreement in cases of similar syntactic structure, but only FastKASSIM is able to recognize syntactically dissimilar utterances. The parse trees of these examples are visualized in Appendix D.

---

[*] denotes equal contribution.

[1] `https://github.com/jasonyux/FastKASSIM`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 211–231
May 2-6, 2023

Like its predecessor, CASSIM, FastKASSIM computes the constituency parse tree of each sentence in a pair of documents, and the similarity between each pair of parse trees. But, while CASSIM used Edit Distance (Pawlik and Augsten, 2011; Zhang and Shasha, 1989) for similarity, we propose using a *Label-based Tree Kernel* (henceforth LTK), our more syntactically thorough implementation of the Fast Tree Kernel (Moschitti, 2006). We evaluate FastKASSIM against CASSIM and Linguistic Style Matching (henceforth LSM; Niederhoffer and Pennebaker (2002); Ireland and Pennebaker (2010)). We find that FastKASSIM is more robust in cases of dissimilarities between documents and is generally more agreeable with human perception of differences in syntax. Additionally, the runtime of LTK is much faster than that of Edit Distance; it *scales linearly with the number of node pairs with the same label* in a pair of parse trees. We empirically show large improvements in runtime with FastKASSIM.

Previously, it was difficult to observe the role of syntax in behavioral phenomena at scale due to runtime constraints. Here, we contribute a study of hypotheses in two sets of applications. First, we examine the relationship between the persuasiveness of online arguments and syntactic similarity, and second, we observe the viability of syntax as an indicator in authorship attribution. FastKASSIM unlocks potential for evaluatory use in more contexts where it is important to preserve syntactic consistency and writing style, e.g., style transfer, machine translation, and story generation.

2 Related Work

A few early studies focused solely on capturing syntactic structures. Sagae and Gordon (2009) sought to cluster *words* by syntactic similarity. In order to establish a distance metric, they computed the cosine distance between vector representations of their unique constituency parses. Other approaches have used LIWC (Tausczik and Pennebaker, 2010). Danescu-Niculescu-Mizil et al. (2011) took a probabilistic approach to measure symmetry and influence of linguistic style.

Other early analytical work found that people will adjust their syntax to match dialog systems' syntactic (Stoyanchev and Stent, 2009) and lexical (Stoyanchev and Stent, 2009; Hoshida et al., 2017) choices. Reitter et al. (2006) found that individual syntactic productions would repeat at low "distances" across utterances in both task-oriented and “spontaneous” dialog. For instance, Reitter and Moore (2007) found that syntactic priming was predictive of success on the HCRC Map Task (Anderson et al., 1991). Baker et al. (2021) similarly discussed the use of syntactic similarity and overall linguistic style synchrony as an indicator of trust and cohesion in teamwork settings, and Boncz (2019) used syntactic similarity in modeling cognitive alignment.

Syntactic features have been shown to improve prediction performance in downstream tasks, e.g., authorship attribution (Posadas-Duran et al., 2014; Raghavan et al., 2010; Zhang et al., 2018) and gender attribution (Sarawgi et al., 2011). In each case, the studies found significant performance gains from models that included syntax features. Despite interest in syntax and clear improvements in prediction performance, the vast majority of recent work primarily focuses on semantic, or even lexical similarities/differences. This ranges from traditional methods such as TFIDF or Jaccard similarity to modern approaches including BERT embeddings (Devlin et al., 2019; Reimers and Gurevych, 2019; Zhang et al., 2019b) and AMR kernels (Opitz et al., 2021). Some approaches use syntactic features specific to certain domains such as Twitter (Alnajran, 2019; Little et al., 2020) or web documents (Broder et al., 1997; Pereira and Ziviani, 2003). Other metrics include “syntactic elements” which take on various forms of parts-of-speech aggregation (Alnajran, 2019; Pakray et al., 2011).

A *syntactic* similarity metric should appropriately consider differences in syntactic structure at the word-, utterance-, and document-level, as opposed to aggregating parts-of-speech or relying on domain-specific features. To our knowledge, CASSIM is the only metric to do this and has been proposed as a solution in applications ranging from measuring communicative alignment (Boncz, 2019; Baker et al., 2021) to evaluating stylistic creativity in language learning (Kokkola and Rydström, 2022) to clustering text (Boghrati et al., 2017). However, CASSIM relies on the expensive Edit Distance metric, and occasionally assigns high similarity scores to documents that appear syntactically dissimilar. An improved syntactic similarity metric would afford new opportunities, from creating novel syntax feature vectors for classification tasks (e.g. authorship and gender attribution), to measuring syntactic coherence in machine translation.

Algorithm 1 FastKASSIM

```
DOCUMENTS $D_1, D_2$
for sentences $S_1, S_2$ in $D_1, D_2$ do
    Compute Parse Tree($S_1$), Parse Tree($S_2$)
end for
for Parse Trees $P_1, P_2$ do
    Compute Tree Kernel:
    $s \leftarrow 0$
    for Node Pair $n_1, n_2$ in $P_1, P_2$ do
        $s \leftarrow s + \Delta_{lb}(n_1, n_2)$
    end for
    Kernel $\leftarrow$ normalize($s$)
end for
Hungarian Algorithm Max. Cost Assignment
return mean(maximal cost pairings)
```

3 FastKASSIM

3.1 CASSIM Background

CASSIM (Boghrati et al., 2018) was the first metric to compute syntactic similarity at the document-level. Their original algorithm uses the Stanford Parser (Klein and Manning, 2003; Chen and Manning, 2014) to compute the parse tree for each sentence in a pair of documents, before computing the Edit Distance (Wagner and Fischer, 1974) between each parse tree pairing. Then, they construct a bipartite graph and use the Hungarian Algorithm for minimum cost assignment (Kuhn, 1955) to pair each tree in one document to the lowest distance tree in the second document. They finally average the Edit Distances of the minimal cost pairings. When there are different numbers of sentences, the number of assignments will correspond to the number of sentences in the document with fewer sentences. Each of that document's sentences will get paired with the most similar sentence in the second document, and the least similar sentences in the second document will remain unpaired. The final Edit Distance between a pair of parse trees P_1, P_2 is normalized as $\frac{EditDistance}{Size(P_1)+Size(P_2)-2}$, where $Size(P)$ is the number of nodes in P.

An important advantage of CASSIM is that it is generalizable to any corpus; it does not represent syntax using platform-specific features like Alnajran (2019); Little et al. (2020). However, the cost of exhaustively using a metric such as Edit Distance is rather penalizing, as its implementations range in asymptotic time complexity from $\Theta(mn)$ (Wagner and Fischer, 1974) to $O(s \times min(m, n))$ (Ukkonen, 1985), where m and n are the string lengths, and s is the maximal Edit Distance.

Algorithm 2 Delta$_{lb}$ Function (Δ_{lb})

```
Tree Nodes $n_1, n_2$; $cache$
Decay $\lambda$; Subtree/Subset Tree Indicator $\sigma$
if $n_1, n_2$ is cached then
    return cache($n_1, n_2$)
end if
if $n_1, n_2$ have different labels then
    return 0
end if
if both $n_1, n_2$ are preterminals then
    cache($n_1, n_2$) $\leftarrow \lambda$ if same label, else 0
    return cache($n_1, n_2$)
end if
Product $\leftarrow$ 1
for child $c_1$ of node $n_1$ do
    Accumulator $\leftarrow$ 0
    for child $c_2$ of node $n_2$ do
        Acc. $\leftarrow$ Acc. + Delta$_{lb}$($c_1, c_2$)
    end for
    Product $\leftarrow$ Product $\times(\sigma+$ Acc.)
end for
cache($n_1, n_2$) $\leftarrow \lambda\times$ Product
return $\lambda\times$ Product
```

3.2 The FastKASSIM Algorithm

In large multi-sentence documents, repeated Edit Distance becomes the most expensive component of CASSIM. Thus, we propose FastKASSIM, which avoids the expensive Edit Distance computation by using Tree Kernels (Moschitti, 2006). Tree Kernels can greatly reduce time complexity by caching between recursive subcalls. The Fast Tree Kernel algorithm Moschitti (2006) runs in *linear time on average with respect to parse tree sizes.*

We propose FastKASSIM, which replaces CASSIM's Edit Distance with a new normalized Tree Kernel. Figure 2 provides a high-level overview of FastKASSIM, which is formally described in Algorithm 1. However, the Fast Tree Kernel does not allow for the case in which two parse tree nodes have matching labels but different productions. We thus also introduce the Label-based Tree Kernel (LTK)[2], which compares the labels at each node in a pair of subtrees or subset trees[3]. This also

---

[2] There is a very strong correlation between LTK and the Fast Tree Kernel ($R = 0.97$, p<0.001).

[3] Moschitti (2006) defines a subtree as a node and all its

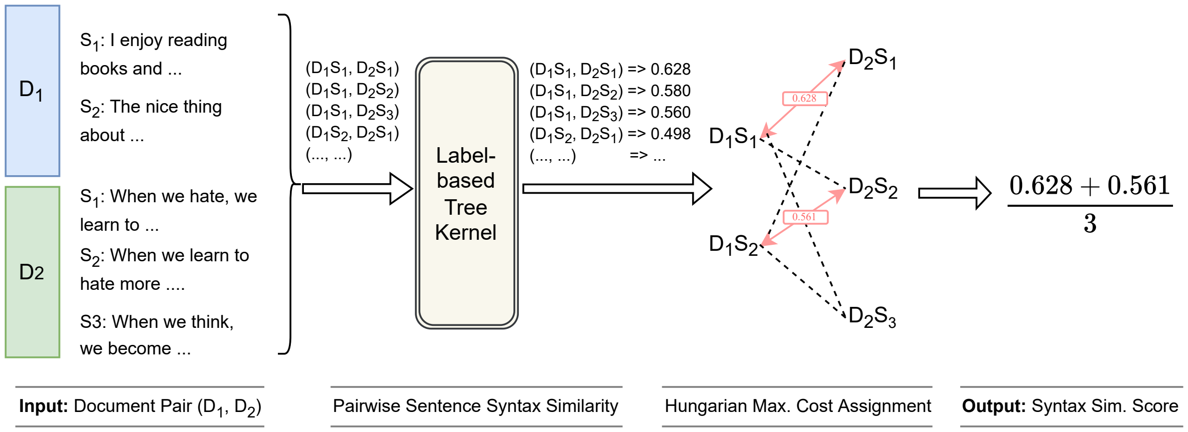


Figure 2: A high-level illustration of FastKASSIM computation. The parse trees of all sentence pairs between D_1, D_2 are computed using LTK. The Hungarian algorithm is used to pair together the most similar parse trees of each sentence in the two documents by the "maximal cost" (i.e., the largest tree kernels). The score is normalized by summing the paired kernel values then dividing by the number of sentences in the document with more sentences. D_2S_3 is unpaired because D_1S_1, D_2S_1 and D_1S_2, D_2S_2 are paired, and D_2 has more sentences than D_1.

more closely follows (Collins and Duffy, 2002), which proposes comparing the actual subset trees rooted at two nodes in each parse tree, rather than the production. Figure 3 depicts the LTK algorithm computing the number of shared subset trees in a pair of parse trees. More formally, as described in lines 7-11 of Algorithm 1, LTK accumulates the value of Δ_{lb} (Algorithm 2), which is the number of common fragments rooted in a pair of parse tree nodes n_1, n_2.

We follow Moschitti (2006) by normalizing $LTK(T1,T2)$ as $\frac{LTK(T1,T2)}{\sqrt{LTK(T1,T1)\times LTK(T2,T2)}}$. This normalized tree kernel is not biased towards tree shape, in contrast to CASSIM's normalized Edit Distance ($\frac{EditDistance}{Size(P_1)+Size(P_2)-2}$). Under CASSIM's normalization, if two sentences resulted in the *same parse tree size despite being composed of entirely different labels*, the normalized Edit Distance approaches 0.50 (the expression approximates $\frac{Size(P_1)}{2\times Size(P_1)-2}$). In other words, according to CASSIM, sentences with the same shape but different parts-of-speech should be neither similar nor dissimilar. We further highlight possible examples of this bias by visualizing parse trees in Appendix D. Ultimately, our normalized LTK results in significant runtime improvements over Edit Distance, as we show in Section 3.3 and derive in Appendix A, and agrees strongly with human perception, as we show in Section 4.

Like the original CASSIM algorithm, our high-level algorithm allows for flexibility in the choice of which parser to use, allowing for future improvements in runtime and correctness as research in parsing progresses. FastKASSIM similarly allows for flexibility in the implementation of tree kernels. Our implementation will be publicly released upon acceptance. In order to directly compare FastKASSIM and CASSIM, we default to using the Stanford Parser (Chen and Manning, 2014) and LTK, our aforementioned modified approach to the Fast Tree Kernel (Moschitti, 2006).[4]

3.3 Overall Metric Runtime Comparison

The largest difference in the runtime of CASSIM and FastKASSIM is that CASSIM uses a normalized Edit Distance to evaluate parse tree similarity, while FastKASSIM uses a normalized tree kernel.

LTK recursively computes Δ_{lb} across all n_1, n_2 pairs in parse trees P_1, P_2. But, importantly, *all comparisons are cached to avoid repetition.* This results in LTK having an asymptotic runtime complexity of $O(S_{12})$, where S_{12} is the total number of pairs of nodes in a pair of parse trees P_1, P_2 that have the same label. We prove this runtime in Appendix A. This is a large improvement over Edit Distance's runtime complexity of $O(s \times min(m,n))$. We confirmed that these asymptotic improvements apply to real-world uses cases by comparing how Edit Distance and LTK scale with the product of parse tree sizes in Figure 6 of the Appendix, finding that LTK scales sublinearly while Edit Distance scales superlinearly.

---

descendants, whereas a subset tree does not require its leaves to be terminal.

[4] However, we provide users with a native interface to interchangeably use any parser supported by NLTK (Bird et al., 2009).

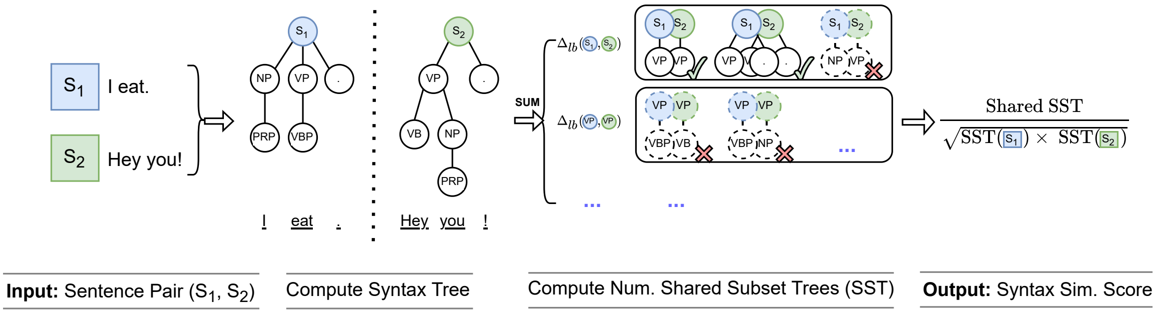


Figure 3: Overview of the Label-based Tree Kernel. The parse trees of a pair of sentences are computed, along with the number of common fragments rooted in each pair of parse tree nodes (i.e., number of shared subset trees). This is normalized by dividing by the square root of the product of the number of subset trees in each parse tree.

While Figure 6 indicates that LTK can be up to an order of magnitude faster than Edit Distance, the largest bottleneck in overall time is still the time to compute each parse tree. Thus, in Figure 4, we investigated the difference in "end-to-end" runtime between FastKASSIM and CASSIM without precomputing the parse trees.

In this experiment, the ChangeMyView dataset (henceforth CMV; Tan et al. (2016)) is used, providing a corpus of unstructured text with large document sizes, to evaluate the promises of FastKASSIM and CASSIM for their abilities to process entire documents. First, we sample entire document pairs and record the time it takes to compute the syntactic similarity of each pair. Each pair is randomly sampled from the 18,363 posts in the CMV training set. Then, we exhaustively paired documents based on the product of their document sizes, providing an approximation of the number of comparisons between parse trees. The document length for each CMV root posts has high variance, so document length products are grouped into bins. For each bin, we randomly sample 60 document pairs and report the average runtime.

Figure 4 shows that FastKASSIM scales well in runtime as the product of document lengths increases. For instance, when syntactic similarity between documents of lengths 300 words and 310 words were compared (product of $93,000$), CASSIM needed on average 113.3 seconds while FastKASSIM took only 21.3 seconds on average. Given these drastic improvements in time complexity, it is now more feasible to compute syntactic similarity at the document level for large corpora.

4 Evaluating FastKASSIM

In this section, we first demonstrate FastKASSIM's overall ability to differentiate between similar and dissimilar documents. Then, we correlate its scores with CASSIM and LSM. Finally, we discuss FastKASSIM's advantages by explaining discrepancies in scoring.

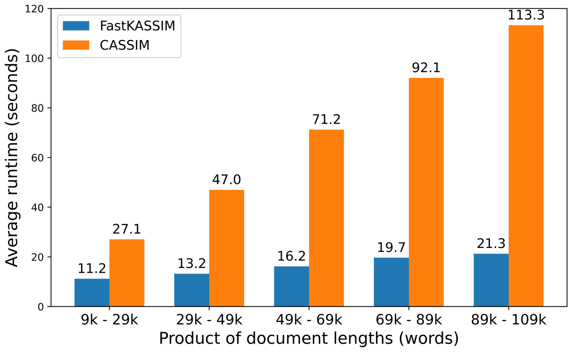


Figure 4: Runtime comparison between FastKASSIM and CASSIM. On the CMV corpus, FastKASSIM runs 2.42 times to 5.32 times faster on average, depending on document size.

4.1 Discriminating Between Syntactically Similar and Dissimilar Documents

Boghrati et al. (2018) validated CASSIM by comparing whether it was consistent with human perception of syntactic similarity. The authors asked Mechanical Turkers to write syntactically similar sentences given a sentence prompt. This resulted in a dataset of 472 English documents from 118 anonymous human annotators, and the authors found that CASSIM was able to "identify syntactically similar documents."

Following Boghrati et al. (2018), we computed the syntactic similarity between each pairing of sentences generated both within the same prompt and between different prompts. Each individual prompt is structurally quite different (Table 1). By construction, *documents resulting from the same prompt should be syntactically similar, whereas documents resulting from different prompts should*

Prompts
1. The two most important days in your life are the day you are born and the day you find out why. The nice thing about being a celebrity is that you bore people and they think it's their fault.
2. I am enough of an artist to draw freely upon my imagination. Imagination is more important than knowledge. Knowledge is limited. Imagination encircles the world.
3. When we love, we always strive to become better than we are. When we strive to become better than we are, everything around us becomes better too.
4. What is the point of being alive if you don't at least try to do something remarkable?

Table 1: Prompts for the crowd-sourced corpus collected by Boghrati et al. (2018).

be dissimilar. As per their work, we fit a maximal structure linear mixed effect model with an indicator for whether the sentence corresponded to the same prompt or a different prompt as a fixed effect and the document ID as a random effect against standardized syntactic similarity[5].

We computed ANOVA of this full model against the reduced model, which drops the comparison type indicator as a fixed effect. The ANOVA χ^2 test on the impact of comparison type yields statistically significant differences in the distribution of FastKASSIM ($\chi^2 = 1438.9$, $p < 0.001$), CASSIM ($\chi^2 = 331.84$, $p < 0.001$), and LSM ($\chi^2 = 201.85$, $p < 0.001$) scores between syntactically similar and dissimilar documents. FastKASSIM results in the largest effect size, 1438.9, indicating it creates the largest differences in distribution.

4.2 Correlating Syntax Metrics

LSM[6] is a metric computing similarities from function word categories. This has ties to matching syntax, as those matching function words correspond to specific parts-of-speech. Moreover, LSM is a widely accepted metric for synchrony and correspondence of general linguistic style in documents (e.g. Chartrand et al. (2005); Ludwig et al. (2013)). We examine the actual similarity scores calculated in the previous section on the crowd-sourced document similarity corpus collected by Boghrati et al. (2018) using each of LSM, FastKASSIM, and CASSIM. In Figure 5, we see that there is a moderately strong correlation between FastKASSIM and LSM ($R = 0.5$, $p < 0.001$). This indicates that FastKASSIM is able to detect matches in key parts-of-speech. We would not expect to see a greatly higher correlation, because LSM is a measure of function words rather than a holistic measure of syntax. On the other hand, while we see a statistically significant correlation between CASSIM and LSM, its correlation coefficient is much lower ($R = 0.11$, $p < 0.001$), indicating a smaller connection between CASSIM representations and function words. This is likely due to the biased Edit Distance normalization mentioned in Section 3.1. Moreover, we actually find an overall negative correlation between FastKASSIM and CASSIM ($R = -0.33$, $p < 0.001$) with a seemingly bipartite relationship. There is an apparent disagreement over documents that FastKASSIM deems dissimilar, with agreement over documents that FastKASSIM deems similar.

4.3 Discrepancies Across Syntax Metrics

Figure 5 indicates that there are several regions of disagreement (vertical clustering). In one region of Figure 5a, FastKASSIM assigned low scores (less than 0.4) despite LSM ranging from 0.258 to 0.842. Recall that in this corpus, every utterance resulting from the same prompt was perceived as syntactically similar, and utterances from two different prompts were perceived as syntactically dissimilar. We find that in 248 out of 249 cases where FastKASSIM assigned a score below 0.4 yet LSM assigned a high score (above 0.6), the two documents being compared came from different prompts. This implies that FastKASSIM indeed is discriminating between different syntactic structures, whereas LSM may be picking up on similarities other than syntax, as expected.

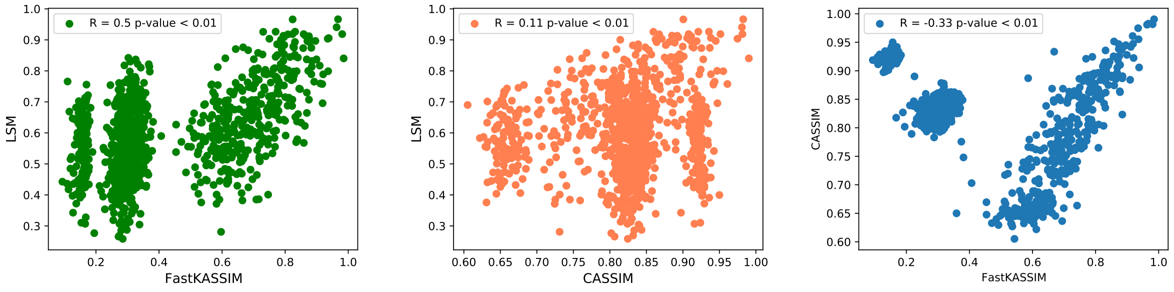
Figure 5b does not indicate any obvious relationship between CASSIM and LSM. More surprisingly, Figure 5c shows that for document pairings that FastKASSIM deems syntactically dissimilar (values of less than 0.5), there is a very strong negative correlation between FastKASSIM and CASSIM ($R = -0.783$, $p < 0.001$). Visually, there is a cluster of pairings where FastKASSIM assigns a value less than 0.4 but CASSIM assigns a value larger than 0.75. We find that in 677 of the 678 pairings in this cluster, the documents originate from different prompts, indicating that the documents are syntactically dissimilar.

We evaluate these discrepancies in Table 2 in terms of each metric's ability to correctly identify documents originating from the same prompt as syntactically similar and those from differ-

---

[5] We use z-score standardization, $\frac{x-\mu}{\sigma}$.

[6] We compute LSM using an implementation publicly available at https://github.com/miserman/lingmatch.



(a)

(b)

(c)

Figure 5: (a). FastKASSIM v. LSM. Moderately strong positive correlation with $R = 0.5$ and $p < 0.01$. (b). CASSIM v. LSM. Weak but statistically significant positive correlation with $R = 0.11$ and $p < 0.01$. (c). FastKASSIM v. CASSIM. Moderately strong negative correlation with $R = -0.33$, $p < 0.01$.

Metric	Acc.	SR	SP	DR	DP
LSM	46.2	92.5	30.8	30.7	92.5
LSM$_a$	65.6	81.1	40.6	60.4	90.6
CASSIM	25.1	**100.**	25.0	0.11	**100.**
CASSIM$_a$	48.8	47.7	23.8	49.2	73.8
BERTScore	25.0	100.	25.0	00.0	00.0
BERTScore$_a$	74.6	99.3	49.6	66.4	99.6
Sentence-BERT	18.9	19.8	74.0	2.70	0.20
Sentence-BERT$_a$	34.3	9.50	19.2	59.3	39.3
FastKASSIM	**88.3**	96.1	**69.1**	**98.5**	85.6

Table 2: Evaluation of LSM, CASSIM, BERTScore, Sentence-BERT and FastKASSIM in terms of Accuracy (Acc.), Similar Document Recall (SR), Similar Document Precision (SP), Dissimilar Document Recall (DR), and Dissimilar Document Precision (DP). Metric$_a$ denotes adjusting to a uniform distribution by quantile transformation.

ent prompts as syntatically dissimilar using accuracy, recall, and precision[7]. In addition to CASSIM and LSM, we include an evaluation of BERTScore (Zhang et al., 2019b) and Sentence-BERT (Reimers and Gurevych, 2019). While they are not syntax metrics, they are strong embedding-based metric which may account for syntactic forms.

As similarity scores lie between 0.0 and 1.0, we use 0.5 as a boundary between similar and dissimilar documents. We reason that in unknown contexts, 0.5 is neutral. We also compare against quantile transformations of each baseline metric, which map each metric's scores to a uniform distribution (note this is an unfair advantage, since in real-time deployment, one cannot observe the entire distribution of values), due to their apparent biases in Figure 5. In Table 2, we find that FastKASSIM holistically outperforms both CASSIM and LSM, with the exception of similar document recall and dissimilar document precision. In these cases, CASSIM achieves perfect precision and recall because it only classified one document pair as dissimilar. BERTScore similarly yields a small range in similarity scores on this corpus. After undergoing a quantile transformation, BERTScore's sensitivity to syntactic differences is magnified, and it performs well in the aforementioned categories but underperforms FastKASSIM in accuracy, similarity precision and dissimilarity recall.

As indicated by its low Dissimilar Document Recall in Table 2, we found CASSIM frequently assigned high similarity scores to syntactically different documents. This likely comes from the bias in its normalized Edit Distance as discussed in Section 3.1, and we show the significant improvements achieved by FastKASSIM.

Overall, our results indicate that with respect to human intuition of syntax, FastKASSIM is more robust than CASSIM, LSM and BERTScore. In Appendix D, we visualize comparisons of several pairs of parse trees along with their FastKASSIM and CASSIM scores.

5 Applications

FastKASSIM is a more accurate and efficient syntactic similarity metric than the current state-of-the-art, opening the possibility for investigating new hypotheses in data-heavy fields with large corpora. Existing applications use syntax metrics for classification (e.g. Posadas-Duran et al. (2014)) as well as analytically to measure hypotheses (e.g. Kaster et al. (2021)). Here, we use syntax as a linguistic style indicator in authorship attribution, and measure syntactic similarity to study communication accommodation in persuasive arguments.

---

[7] Exact expressions provided in Appendix B.

5.1 Persuasiveness of Syntax Accommodation

Early work in communication accommodation theory found that matching communication styles can create a sense of familiarity, which improves social and conversational outcomes (e.g. Curhan and Pentland (2007); Giles (2016)). While most existing work has focused on hypotheses at the word-level (Tan et al., 2016), we hypothesize that CMV arguments that are more syntactically similar to opinions may be more persuasive as well.[8]

On CMV, users write an original post describing an opinion and allow "challengers" to present arguments attempting to change their opinion. Original posters (OP) indicate whether their opinions have been changed by assigning a "delta," which we can use an indicator of successful persuasion. We computed the syntactic similarity between a challenger's initial argument and an original poster's (OP) original opinion. This choice is made because the OP presents their full opinion in their original post, and a challenger typically presents their central argument in their initial challenge (Tan et al., 2016). While many CMV studies predict persuasion outcomes, prediction tasks do not reveal the actual bidirectional relationship between syntactic similarity and persuasion. We use FastKASSIM to analyze this relationship.

We find that arguments which eventually lead to deltas ($\mu = 0.307$) tend to be more syntactically similar to original opinions than unsuccessful ($\mu = 0.263$) arguments ($t = 19.016$; $p < 0.001$). However, this finding does not imply on its own that *syntactically similar arguments are more persuasive*. Thus, we also examined the converse by computing the persuasion rates (proportion of threads receiving deltas) of the most syntactically similar and dissimilar arguments. We computed the syntactic similarity of each pairing of initial arguments and original opinions and grouped them into the top and bottom 33% of syntactic similarity. This resulted in a minimum syntactic similarity value of 0.341 for the top 33% ($\mu = 0.453$) and maximum of 0.171 for the bottom 33% ($\mu = 0.096$). We found that threads grouped in the bottom 33% of syntactic similarity only had a persuasion rate of 6.377%, while the persuasion rate for threads grouped in the top 33% was nearly twice that, 12.347% ($t = 19.135$; $p < 0.001$). These findings support the hypothesis that similar syntactic patterns play a role in persuasion — may be an indication of stylistic familiarity for the OP.

Features	**Acc.**$_{(\sigma)}$	**F1**$_{(\sigma)}$
Majority Baseline	0.767	0.868
Bag of Words	$0.892_{(0.02)}$	$0.867_{(0.02)}$
Bag of Words + Syntax	$0.923_{(0.02)}$	$0.922_{(0.01)}$
RoBERTa	$0.939_{(0.01)}$	$0.935_{(0.00)}$
RoBERTa + Syntax	$\mathbf{0.945}_{(0.01)}$	$\mathbf{0.938}_{(0.01)}$

Table 3: **Judgment** test set results comparing accuracy and weighted F1 score between unigram counts and unigram counts augmented with syntactic features. Standard deviation (σ) given in subscripts.

5.2 Authorship Attribution

Much work has examined methods for attributing authorship based upon linguistic features (Juola, 2008; Raghavan et al., 2010; Seroussi et al., 2011b). The **Judgment dataset** (Seroussi et al., 2011a) contained English judgments delivered by judges on the Australian High Court from 1913 to 1975. We classified whether 924 judgments were written by Sir Edward McTiernan or Sir George Rich during non-overlapping time periods (Rich's judgments from 1913-1928 and McTiernan's from 1965-2971). We follow the experimental design and preprocessing steps in Seroussi et al. (2011b)[9].

To capture semantics, one setting used normalized Bag of Words with Support Vector Machines and the other used a state-of-the-art fine-tuned RoBERTa (Liu et al., 2019) model[10]. We augmented both semantic settings with a syntactic similarity feature vector — for each classification instance, we randomly sampled 25 posts from the training set and computed the FastKASSIM syntactic similarity between judgments written by Rich and McTiernan, respectively. The syntactic similarity features consisted of the minimum, maximum, mean, and standard deviation of these comparisons. We evaluated our classifier on a 10% withheld testing set[11].

Table 3 shows that adding syntactic features to both semantic models results in gains in both accuracy and weighted F1. This is even the case when using RoBERTa; we achieve the strongest performance using RoBERTa with a weighted sum between textual and syntactic features, fine-tuned

---

[8] Appendix C includes full details on CMV and preprocessing.

[9] All experiments were computed on one RTX A6000 GPU.

[10] Base RoBERTa (123M parameters). We set an initial learning rate of 2e-5 and a 0.01 weight decay.

[11] We used 4 seeds to sample our data.

using modules from the frameworks in Gu and Budhkar (2021); Wolf et al. (2020). Syntactic similarity with reference documents may provide a strong indicator of writing style.

6 Conclusion

We have introduced FastKASSIM, which has runtime improvements that scale significantly with document sizes and achieves better agreement with human perception of syntactic differences compared to standard syntax metrics. These improvements are possible due to our Label-based Tree Kernel, which has an improved asymptotic runtime complexity and a corrected normalization. FastKASSIM also allowed us to verify hypotheses regarding the importance of syntax both in authorship attribution and social dynamics such as persuasion. These findings motivate further applications of syntax.

Acknowledgements

Many thanks to Lillian Lee and Chenhao Tan for their feedback and collaboration on early versions of this work. We are very grateful to Reihane Boghrati for providing access to their Mechanical Turk dataset from their work on CASSIM. Thanks to Jack Hessel, Kun Qian, Yu Li, Qingyang Wu, Sky Wang, Mert Inan, and Zhiyang Xu for their helpful feedback and discussions.

Limitations

Our work relies on a couple assumptions. Our main corpus for evaluation is the crowdsourced and human-annotated dataset from Boghrati et al. (2018). As a result, our claim to better represent human perception of syntax relies on the assumption that their annotators correctly filter out responses which are not actually syntactically similar to each prompt. They had an acceptable Cohen's Kappa of 0.53. Additionally, we only use corpora that are in English. Future work should look towards applying our general approach to other languages.

In our evaluation of FastKASSIM against LSM and CASSIM, we also evaluate its ability to correctly identify statements created from the same prompt as similar and statements created from different prompts as dissimilar (Table 2). In this evaluation, we assume that 0.50 is an acceptable threshold for syntactic similarity and dissimilarity, because without any contextual information, one would assume that there are an equal amount of similar and dissimilar documents. Despite this, we still performed quantile adjustments for each comparison metric, uniformly distributing the scores between 0.0 and 1.0. This gives is an unfair advantage for the comparison metrics (i.e., CASSIM, LSM, and BERTScore), since "in-the-wild" it is impossible to obtain the eventual distribution of scores. Future work may consider methods to rebalance each of these scores, including conducting human evaluation to evaluate whether 0.50 is an acceptable threshold for syntactic similarity both before and after each metric undergoes an adjustment to the uniform distribution.

FastKASSIM is a metric for syntactic similarity between a pair of utterances or documents. However, similarity is only one dimension of syntax, which removes some granularity — for instance, syntactic similarity cannot explain which specific productions are shared. Similarity metrics like FastKASSIM instead afford opportunities in a variety of other applications, such as syntactic coherence in language generation and verifying computational social science hypotheses.

Additionally, parsing is still a significant bottleneck in runtime. Future work may wish to consider ways to mitigate the cost of parsing. One may also consider using sequential modeling to generate syntactic parse trees, or to directly model the output of FastKASSIM.

Ethical Considerations

Our study makes use of three datasets. First is the set of prompts collected in Boghrati et al. (2018), which involved anonymous participants creating fictional statements, so there is no personal information involved. Second is the publicly available r/ChangeMyView dataset collected by Tan et al. (2016), which consists of statements made by users behind typically anonymous aliases. Lastly is the publicly available WikiQA corpus (Yang et al., 2015), which does not contain identifying information.

In our r/ChangeMyView application studying the relationship between syntactic similarity and persuasion, we make the assumption that r/ChangeMyView is a community representative of online arguments. However, partially due to its anonymity, it is unknown whether r/ChangeMyView is a representative sample with diversity in location, educational background, socioeconomic status, ethnicity, and many other important factors. An ideal study should be able to control for proxies for individual traits in order to isolate the impact of syntax itself.

Generally, while most algorithms are not inherently unethical, there is often potential for abuse in their applications. The individual computations in the FastKASSIM algorithm do not have any negative implications, but it is possible to use syntactic similarity for unethical downstream tasks. For instance, because syntax is an important aspect of writing style, it is possible that users may try to adversarially uncover an anonymous author's identity. We do not condone the use of FastKASSIM for any unlawful or morally unjust activities. We do not propose any new tasks that would introduce unethical activity.

References

Noufa Abdulaziz Alnajran. 2019. *An integrated semantic-based framework for intelligent similarity measurement and clustering of microblogging posts*. Ph.D. thesis, Manchester Metropolitan University.

Anne H Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, et al. 1991. The hcrc map task corpus. *Language and speech*, 34(4):351–366.

Anthony L Baker, Sean M Fitzhugh, Lixiao Huang, Daniel E Forster, Angelique Scharine, Catherine Neubauer, Glenn Lematta, Shawaiz Bhatti, Craig J Johnson, Andrea Krausman, et al. 2021. Approaches for assessing communication in human-autonomy teams. *Human-Intelligent Systems Integration*, 3(2):99–128.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Natalia Bogdanova-Beglarian, Tatiana Sherstinova, Olga Blinova, and Gregory Martynenko. 2016. An exploratory study on sociolinguistic variation of russian everyday speech. In *International Conference on Speech and Computer*, pages 100–107. Springer.

Reihane Boghrati, Joe Hoover, Kate M Johnson, Justin Garten, and Morteza Dehghani. 2016. Syntax accommodation in social media conversations. In *CogSci*.

Reihane Boghrati, Joe Hoover, Kate M Johnson, Justin Garten, and Morteza Dehghani. 2018. Conversation level syntax similarity metric. *Behavior research methods*, 50(3):1055–1073.

Reihane Boghrati, Kate M Johnson, and Morteza Dehghani. 2017. Generalized representation of syntactic structures. In *CogSci*.

Adam Boncz. 2019. *Communication as Joint Action: The role of cognitive alignment and coupling*. Ph.D. thesis, Central European University.

Andrei Z Broder, Steven C Glassman, Mark S Manasse, and Geoffrey Zweig. 1997. Syntactic clustering of the web. *Computer networks and ISDN systems*, 29(8-13):1157–1166.

Tanya L Chartrand, William W Maddux, and Jessica L Lakin. 2005. Beyond the perception-behavior link: The ubiquitous utility and motivational moderators of nonconscious mimicry. *The new unconscious*, pages 334–361.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 263–270.

Jared R Curhan and Alex Pentland. 2007. Thin slices of negotiation: Predicting outcomes from conversational dynamics within the first 5 minutes. *Journal of Applied Psychology*, 92(3):802.

Cristian Danescu-Niculescu-Mizil, Michael Gamon, and Susan Dumais. 2011. Mark my words! linguistic style accommodation in social media. In *Proceedings of the 20th international conference on World wide web*, pages 745–754.

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

Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Characterizing stylistic elements in syntactic structure. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1522–1533.

Howard Giles. 2016. Communication accommodation theory. *The international encyclopedia of communication theory and philosophy*, pages 1–7.

Ken Gu and Akshay Budhkar. 2021. A package for learning on tabular and text data with transformers. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*, pages 69–73.

Patrick GT Healey, Matthew Purver, and Christine Howes. 2014. Divergence in dialogue. *PloS one*, 9(6):e98598.

Masahiro Hoshida, Masahiko Tamura, and Yugo Hayashi. 2017. Lexical entrainment toward conversational agents: An experimental study on top-down processing and bottom-up processing. In *Proceedings of the 5th International Conference on Human Agent Interaction*, HAI '17, page 189–194, New York, NY, USA. Association for Computing Machinery.

Molly E Ireland and James W Pennebaker. 2010. Language style matching in writing: synchrony in essays, correspondence, and poetry. *Journal of personality and social psychology*, 99(3):549.

Patrick Juola. 2008. *Authorship attribution*, volume 3. Now Publishers Inc.

Maurits Kaptein, Deonne Castaneda, Nicole Fernandez, and Clifford Nass. 2014. Extending the similarity-attraction effect: The effects of when-similarity in computer-mediated communication. *Journal of Computer-Mediated Communication*, 19(3):342–357.

Marvin Kaster, Wei Zhao, and Steffen Eger. 2021. Global explainability of bert-based evaluation metrics by disentangling along linguistic factors. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8912–8925.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st annual meeting of the association for computational linguistics*, pages 423–430.

Lydia Kokkola and Ulla Rydström. 2022. Creativity and cognition in fiction by teenage learners of english. *Language and Literature*, 31(1):99–118.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Claire Little, David Mclean, Keeley Crockett, and Bruce Edmonds. 2020. A semantic and syntactic similarity measure for political tweets. *IEEE Access*, 8:154095–154113.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Stephan Ludwig, Ko De Ruyter, Mike Friedman, Elisabeth C Brüggen, Martin Wetzels, and Gerard Pfann. 2013. More than words: The influence of affective content and linguistic style matches in online reviews on conversion rates. *Journal of Marketing*, 77(1):87–103.

William W Maddux, Elizabeth Mullen, and Adam D Galinsky. 2008. Chameleons bake bigger pies and take bigger pieces: Strategic behavioral mimicry facilitates negotiation outcomes. *Journal of Experimental Social Psychology*, 44(2):461–468.

Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *11th conference of the European Chapter of the Association for Computational Linguistics*.

Kate G Niederhoffer and James W Pennebaker. 2002. Linguistic style matching in social interaction. *Journal of Language and Social Psychology*, 21(4):337–360.

Juri Opitz, Angel Daza, and Anette Frank. 2021. Weisfeiler-leman in the bamboo: Novel amr graph metrics and a benchmark for amr graph similarity. *Transactions of the Association for Computational Linguistics*, 9:1425–1441.

Partha Pakray, Sivaji Bandyopadhyay, and Alexander Gelbukh. 2011. Textual entailment using lexical and syntactic similarity. *International Journal of Artificial Intelligence and Applications*, 2(1):43–58.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Mateusz Pawlik and Nikolaus Augsten. 2011. Rted: a robust algorithm for the tree edit distance. *Proceedings of the VLDB Endowment*, 5(4):334–345.

AR Pereira and Nivio Ziviani. 2003. Syntactic similarity of web documents. In *Proceedings of the IEEE/LEOS 3rd International Conference on Numerical Simulation of Semiconductor Optoelectronic Devices (IEEE Cat. No. 03EX726)*, pages 194–200. IEEE.

Juan-Pablo Posadas-Duran, Grigori Sidorov, and Ildar Batyrshin. 2014. Complete syntactic n-grams as style markers for authorship attribution. In *Mexican International Conference on Artificial Intelligence*, pages 9–17. Springer.

Sindhu Raghavan, Adriana Kovashka, and Raymond Mooney. 2010. Authorship attribution using probabilistic context-free grammars. In *Proceedings of the ACL 2010 conference short papers*, pages 38–42.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* *and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

David Reitter and Johanna D Moore. 2007. Predicting success in dialogue. In *Proceedings of 45th Annual Meeting of the Association for Computational Linguistics*, pages 808–815. Association for Computational Linguistics.

David Reitter, Johanna D Moore, and Frank Keller. 2006. Priming of syntactic rules in task-oriented dialogue and spontaneous conversation. In *In Proceedings of the 28th Annual Conference of the Cognitive Science Society*.

Kenji Sagae and Andrew Gordon. 2009. Clustering words by syntactic similarity improves dependency parsing of predicate-argument structures. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 192–201.

Ruchita Sarawgi, Kailash Gajulapalli, and Yejin Choi. 2011. Gender attribution: tracing stylometric evidence beyond topic and genre. In *Proceedings of the fifteenth conference on computational natural language learning*, pages 78–86.

Yanir Seroussi, Russell Smyth, and Ingrid Zukerman. 2011a. Ghosts from the high court's past: Evidence from computational linguistics for dixon ghosting for mctiernan and rich. *University of New South Wales Law Journal, The*, 34(3):984–1005.

Yanir Seroussi, Ingrid Zukerman, and Fabian Bohnert. 2011b. Authorship attribution with latent dirichlet allocation. In *Proceedings of the fifteenth conference on computational natural language learning*, pages 181–189.

Svetlana Stoyanchev and Amanda Stent. 2009. Lexical and syntactic adaptation and their impact in deployed spoken dialog systems. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 189–192.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th international conference on world wide web*, pages 613–624.

Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Esko Ukkonen. 1985. Algorithms for approximate string matching. *Information and control*, 64(1-3):100–118.

Robert A Wagner and Michael J Fischer. 1974. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173.

Christopher G Wetzel and Chester A Insko. 1982. The similarity-attraction relationship: Is there an ideal one? *Journal of Experimental Social Psychology*, 18(3):253–276.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Kaizhong Zhang and Dennis Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing*, 18(6):1245–1262.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019a. Syntax-enhanced neural machine translation with syntax-aware word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1151–1161, Minneapolis, Minnesota. Association for Computational Linguistics.

Richong Zhang, Zhiyuan Hu, Hongyu Guo, and Yongyi Mao. 2018. Syntax encoding with application in authorship attribution. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2742–2753, Brussels, Belgium. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019b. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Formalizing FastKASSIM

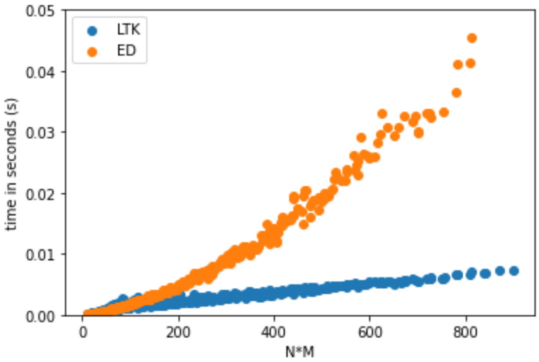


Figure 6: Runtime Comparison of Edit Distance (ED) and Label-based Tree Kernel (LTK) on WikiQA with varying NM (product of parse tree sizes).

The Label-based Tree Kernel recursively computes $\mathrm{Delta}_{\mathrm{lb}}$ across all n_1, n_2 pairs in parse trees P_1, P_2. The time complexity of $\Delta_{\mathrm{lb}}(n_1, n_2)$ has a ceiling of $O(L_1^h \times L_2^h)$, where L_i^h is the number of nodes at height h for a tree rooted at n_i, and h is the minimum height between the two trees. In the worst case scenario, Δ_{lb} of a fully uncached pair n_i, n_j results in recursive calls at every depth level. Δ_{lb} is computed for each node pair, so the ceiling of the tree kernel runtime is $O(NM)$, where N, M are the total number of nodes in P_1 and P_2, respectively.[12]

$O(NM)$ is a ceiling assuming the worst-case, where the labels are the same at each comparison, requiring full recursion. Let us consider there to be k shared labels in a pair of parse trees.

In each parse tree P_1, P_2, there will be $C_i^{(1)}, C_i^{(2)}$ connected components, one for each shared label $i \in [1, k]$, where a connected component consists of connected nodes with the same label. Out of the $C_i^{(1)}$ components in parse tree P_1, let $N_{i,j}^{(1)}$ be the size of each individual component j. So, for label $i = 1$, the number of comparisons follows:

$$O\left(\sum_{l=1}^{C_1^{(1)}} \sum_{m=1}^{C_1^{(2)}} N_{1,l}^{(1)} N_{1,m}^{(2)}\right)$$

which represents iterating through every pair of the $C_1^{(1)} \times C_1^{(2)}$ possible pairs of connected components and computing LTK. Then, for k shared labels, the worst-case runtime (i.e. the connected components do not form shared subtrees), we have equation 1:

$$O\left(\sum_{i=1}^{k} \sum_{l=1}^{C_i^{(1)}} \sum_{m=1}^{C_i^{(2)}} N_{i,l}^{(1)} N_{i,m}^{(2)}\right) = O\left(\sum_{i=1}^{k} N_i^{(1)} N_i^{(2)}\right) \tag{1}$$

where $N_i^{(1)}, N_i^{(2)}$ are the total number of nodes that have label i in P_1, P_2 respectively. However, recall from Algorithm 2 that LTK only iterates through pairs that share the same label; it does not matter if the connected components themselves are intertwined. Then, further simplifying this term we have equation 2:

$$O\left(\sum_{i=1}^{k} N_i^{(1)} N_i^{(2)}\right) = O(S_{12}) \tag{2}$$

where S_{12} is simply the total number of pairs in P_1, P_2 that have the same label. When all nodes have the same label, $S_{12} = NM$, consistent with the observed runtime ceiling. Empirically, we see that the *expectation* of S_{12} is much smaller than NM, as seen by the sublinear time scaling in Figure 6.

A.1 Scaling with Node Pairings (NM)

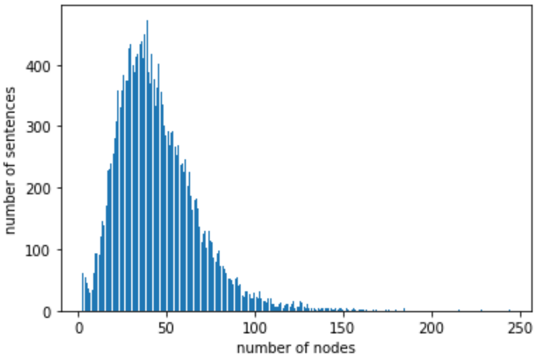


Figure 7: Statistics of Parse Trees in WikiQA

We first examine the relationship between LTK and NM, the number of possible node pairings, using the WikiQA corpus (Yang et al., 2015), a public dataset containing annotated question and answer pairs written in English. This dataset is chosen in order to compare FastKASSIM and CASSIM on well-structured text, and because of the ability to extract clean sentences of various length, which is crucial in determining NM. The experiment utilized 20,347 answer sentences from the WikiQA training set. We compute the parse trees prior to computing the runtimes of Edit Distance and LTK.

---

[12] Note that this is only possible due to Δ_{lb} caching the repetitive computations when iterating over node pairs.

Metric	Sim.	Dis.
LSM	70.4	56.0
LSM$_a$	72.2	42.6
CASSIM	82.1	82.0
CASSIM$_a$	48.3	50.6
BERTScore	89.9	85.1
BERTScore$_a$	85.7	38.0
Sentence-BERT	60.6	81.4
Sentence-BERT$_a$	26.7	57.7
FastKASSIM	73.1	31.7
FastKASSIM$_a$	86.1	38.0

Table 4: Average score assigned to similarity document pairings (Sim.) and dissimilar document pairings (Dis.) by each metric. Metric$_a$ denotes an adjustment to a uniform distribution using a quantile transformation.

Statistics of parse trees from the WikiQA-train corpus are shown in Figure 7.

As the corpus is rather large, we consider sentences with fewer than 30 nodes, which resulted in a total of 16,591,680 possible pairings. Then, pairings are grouped by the product of their nodes $N \times M$. For each value of $N \times M$, we track the cost of computing both the Edit Distance and LTK for all pairings if there are less than 10 pairs, or randomly sample 10 pairs if there are more. The average runtime for both Edit Distance and LTK for each value of $N \times M$ are shown in Figure 6.

B Metrics for Evaluating FastKASSIM, CASSIM, and LSM

In Section 4.3 and Table 2, we evaluated FastKASSIM, CASSIM, and LSM in terms of Similarity Accuracy, Similar Document Recall, Similar Document Precision, Dissimilar Document Recall, and Dissimilar Document Precision. These all follow the standard formulas for accuracy, recall, and precision. Adapted to our similarity context:

Similarity accuracy is the sum of the number of same prompt pairs receiving a score greater than 0.50 and the number of different prompt pairs receiving a score lower than 0.50 divided by the total number of pairings.

Similar document recall is the number same prompt pairs receiving a score greater than 0.50 divided by the total number of pairings originating from the same prompt.

Similar document precision is the number of same prompt pairs receiving a score greater than 0.50 divided by the total number of pairings receiving a score greater than 0.50.

Dissimilar document recall is the number different prompt pairs receiving a score less than 0.50 divided by the total number of pairings originating from different prompt.

Dissimilar document precision is the number of different prompt pairs receiving a score less than 0.50 divided by the total number of pairings receiving a score less than 0.50.

C Persuasiveness of Syntactic Similarity: Additional Context

C.1 CMV Background

We investigate the role of syntax in persuasive arguments in the r/ChangeMyView[13] community (CMV) on Reddit. CMV users come in "good faith" that they are open to changing their view on a controversial topic. They write an original post describing an opinion and allow “challengers” to comment on their post and attempt to change their opinion. If their opinion is changed, the original poster (OP) will indicate this by assigning a “delta” (by typing either “!delta” or Δ in response to the persuasive comment). An OP may choose to present a rebuttal to a challenger, openly disagree with a challenger, or simply ignore a challenger (e.g., Figure 8). All Reddit users use anonymous aliases, unless they explicitly disclose their identity.

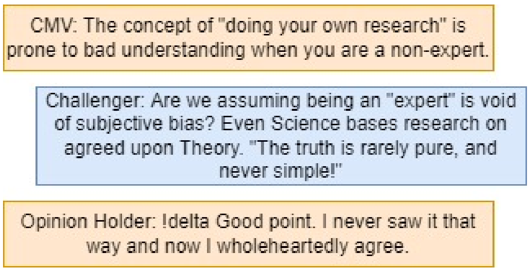
Earlier work found positive relationships between *behavioral mimicry* (mirroring behaviors) and in-person *negotiations* (Curhan and Pentland, 2007; Maddux et al., 2008). Yet, Healey et al. (2014) found that in general spoken conversations, peoples' syntactic patterns diverged from each other. We thus investigate the hypothesis that as a challenger on CMV continues to *engage* in an argument with an OP, their syntactic communication styles may begin to converge in order to “optimize for social differences.” Additionally, we hypothesize that challengers who utilize similar syntactic patterns, whether intentionally or not, may be more persuasive.

C.2 Dataset

We use the CMV dataset consisting of 18,363 posts and 1,114,533 comments written in English and collected by Tan et al. (2016). As in Tan et al. (2016), we examine discussion trees with at least 10 replies from challengers and at least one OP

---

[13] https://www.reddit.com/r/changemyview/


CMV: The concept of "doing your own research" is prone to bad understanding when you are a non-expert.

Challenger: Are we assuming being an "expert" is void of subjective bias? Even Science bases research on agreed upon Theory. "The truth is rarely pure, and never simple!"

Opinion Holder: !delta Good point. I never saw it that way and now I wholeheartedly agree.

Figure 8: An example of a thread on CMV. The OP presents a set of arguments defending their opinion (orange, top), inviting challengers to contest their opinion (blue). The OP acknowledged their opinion has been changed by assigning a delta (orange, bottom).

reply, in order to focus on discussions with "non-trivial" amounts of engagement. We also filter out posts which receive more than 10,000 comments in order to reduce noise from "outsiders" in cases where a post goes viral. When a challenger comments on an original post, it starts a "thread," with the original post (OP's opinion) taking on the root, index 0, and the challenger's comment taking on index 1. Each additional comment made in reply extends the thread. We are interested in syntactic accommodation, so we only consider threads that consist of conversations between the OP and a single challenger to eliminate confounders. These preprocessing steps results in a final dataset consisting of $15,986$ posts.

C.3 Dicusssion & Implications

In Section 5.1, we found two very statistically significant relationships between syntactic similarity and persuasive arguments. First, arguments that eventually lead to deltas tend to be more syntactically similar to original opinions compared to arguments that do not. Second, the arguments that are the most syntactically similar to original opinions actually were nearly twice as likely to receive deltas than the least syntactically similar arguments. Altogether, this may imply that syntactically similar arguments are more persuasive. This idea is supported by the rich body of work suggesting that similarity and communicative familiarity leads to improved social and conversational outcomes (Curhan and Pentland, 2007; Giles, 2016; Kaptein et al., 2014; Maddux et al., 2008; Wetzel and Insko, 1982).

D Parse Tree Examples

We visualize the constituency parse trees of several sentences taken from the corpus collected in Boghrati et al. (2018) using the online interface of the Berkeley Neural Parser[14], which uses the parser described in Kitaev et al. (2019).

We first compare the parse trees of the examples provided in Figure 1. Figure 9 compares the parse trees of the two similar documents shown in Figure 1. The first document is composed of one sentence — "I like swimming because it is cool." and the second document is also composed of one sentence — "I love running because it is fun." CASSIM assigned a score of 0.962, and FastKASSIM assigned a score of 0.928. The structure and composition of these two documents are nearly identical; the only difference is the production associated with the words "running" and "swimming."

Figure 10 is a visualization of the parse trees of the two dissimilar documents shown in Figure 1. The first document is composed of two sentences: "When we hate, we always move away from the grace of God. When we become resentful and unforgiving, the world around us seems spiteful and meaningless." The second document is composed of one sentence: "How can you be skiing if you are already swimming?" Beyond the differing number of sentences, the sentences in the first document individually appear structurally dissimilar compared to the sentence in the second document. FastKASSIM assigned a low score — 0.219, whereas CASSIM assigned a high score — 0.838.

Figure 11 compares the parse trees of the two single-sentence documents "How can you be skiing if you already swimming?" and "Knowledge is important to succeed." As is clear from Figure 11, the two sentences are structurally and compositionally quite different. FastKASSIM assigned a score of 0.439, whereas CASSIM assigned a score of 0.679.

Figure 12 compares the parse trees of two separate documents. The first document is composed of two sentences: "When we dream, we often search for deeper meaning. When we search for deeper meaning, other things become more nuanced too." The second document is composed of two sentences as well: "When we concentrate, we try to do better on a task. When we strive to do better, we end up doing better too." The structures of the two documents appear rather similar, but there do appear to be some differences in composition

---

[14] https://parser.kitaev.io/

(i.e., in terms of the constituent parts-of-speech). FastKASSIM assigned a score of 0.656, and CASSIM assigned a score of 0.837. While both scores are relatively high, FastKASSIM may be more penalizing towards these types of differences.

Figure 13 compares the parse trees of two separate documents. The first document is composed of four sentences: “I am old enough to draw freely upon my experience. Experience is more important than luck. Luck can turn. Experience lasts a lifetime.” The second document is composed of one sentence: “Being loving makes you become better.” Holistically, the structures of the two documents are quite different. Beyond the differing number of sentences in each document, there are also not any individual sentences between the two documents that appear particularly syntactically similar. FastKASSIM assigned a score of 0.15, whereas CASSIM assigned a score of 0.924.

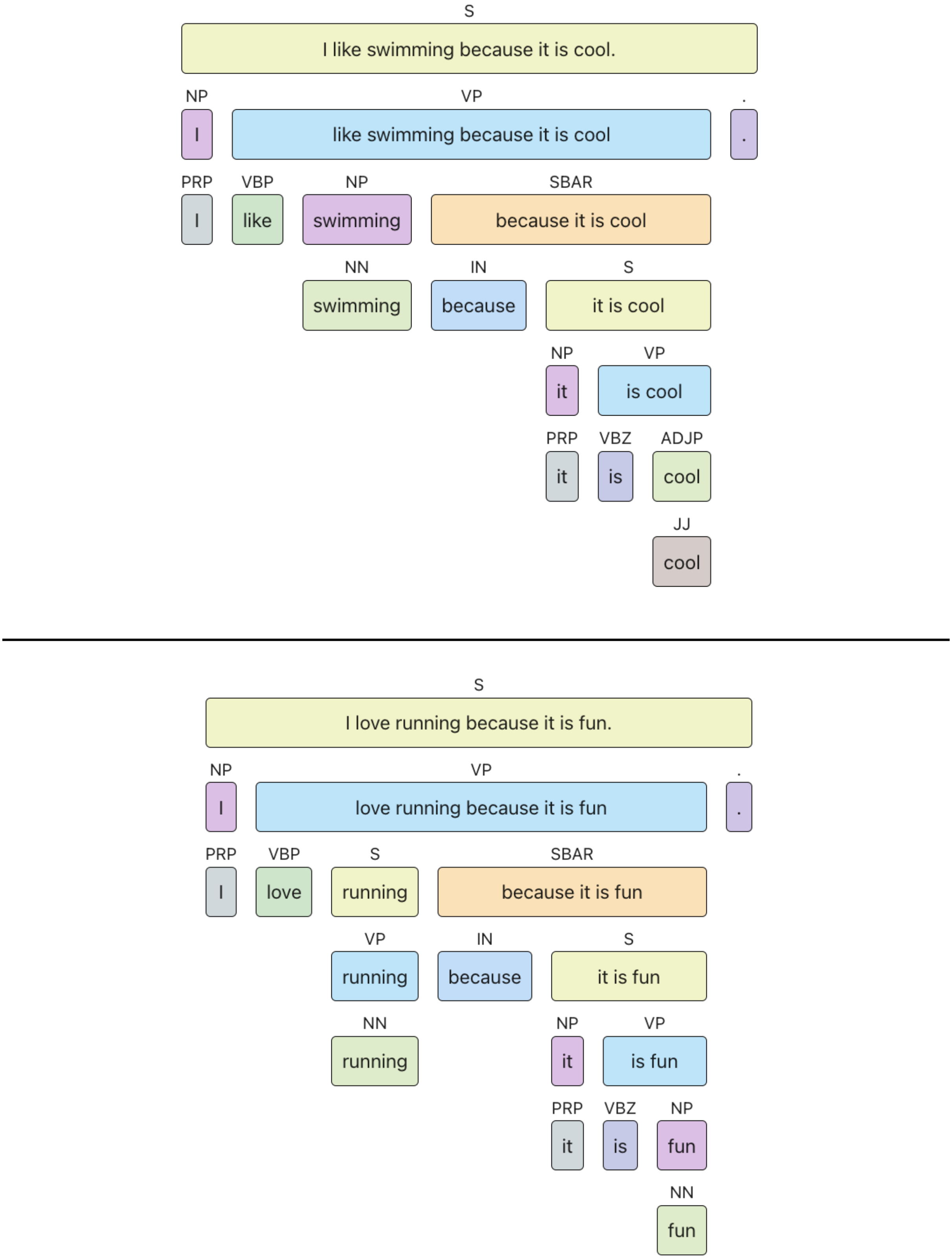


Figure 9: A comparison of the parse trees of two syntactically similar documents from Figure 1. Top document: “I like swimming because it is cool.” Bottom document: “I love running because it is fun.” FastKASSIM similarity score: 0.928; CASSIM similarity score: 0.962.

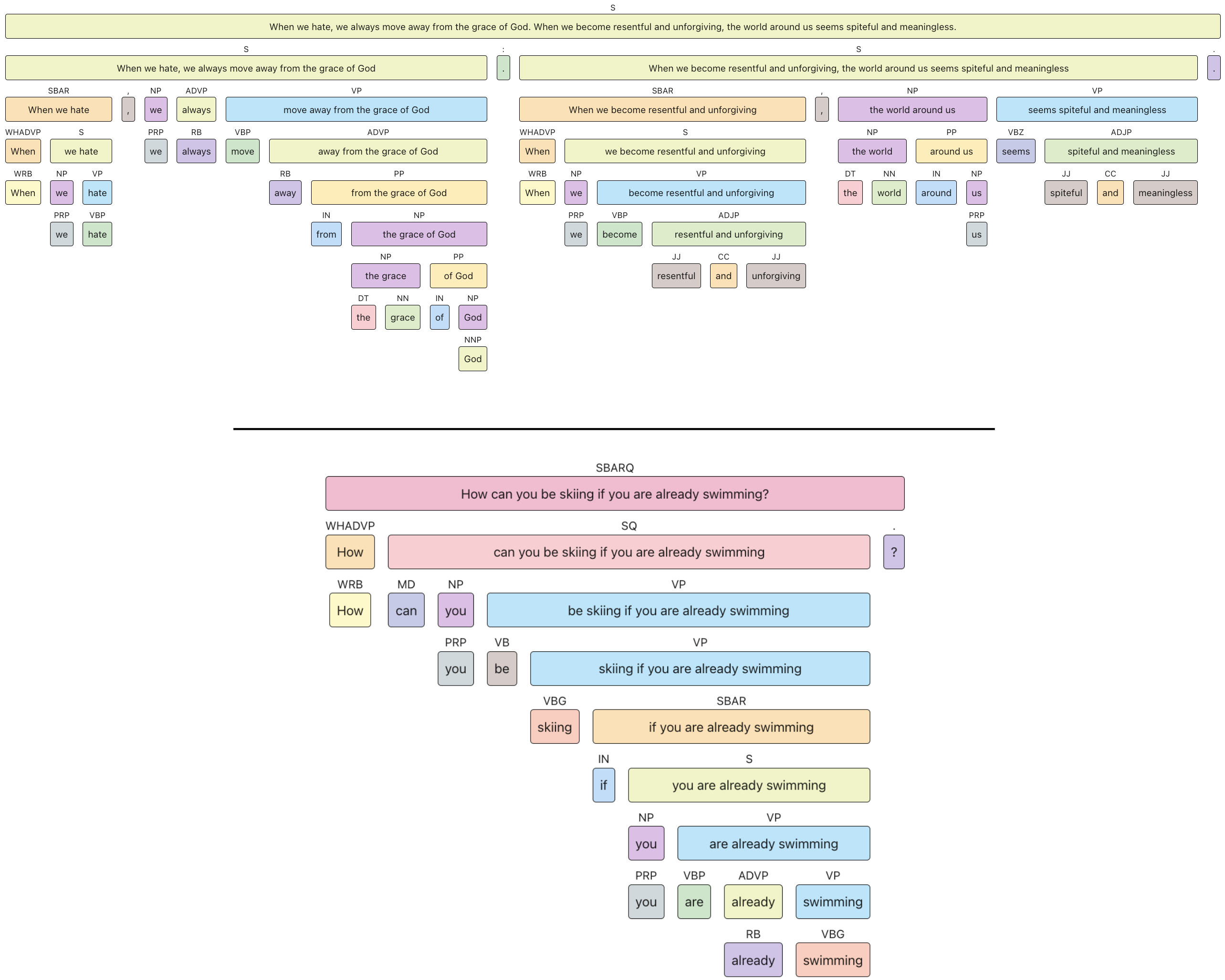


Figure 10: A comparison of the parse trees of two syntactically dissimilar documents. Top document: "When we hate, we always move away from the grace of God. When we become resentful and unforgiving, the world around us seems spiteful and meaningless." Bottom document: "How can you be skiing if you are already swimming?" FastKASSIM similarity score: 0.219; CASSIM similarity score: 0.838.

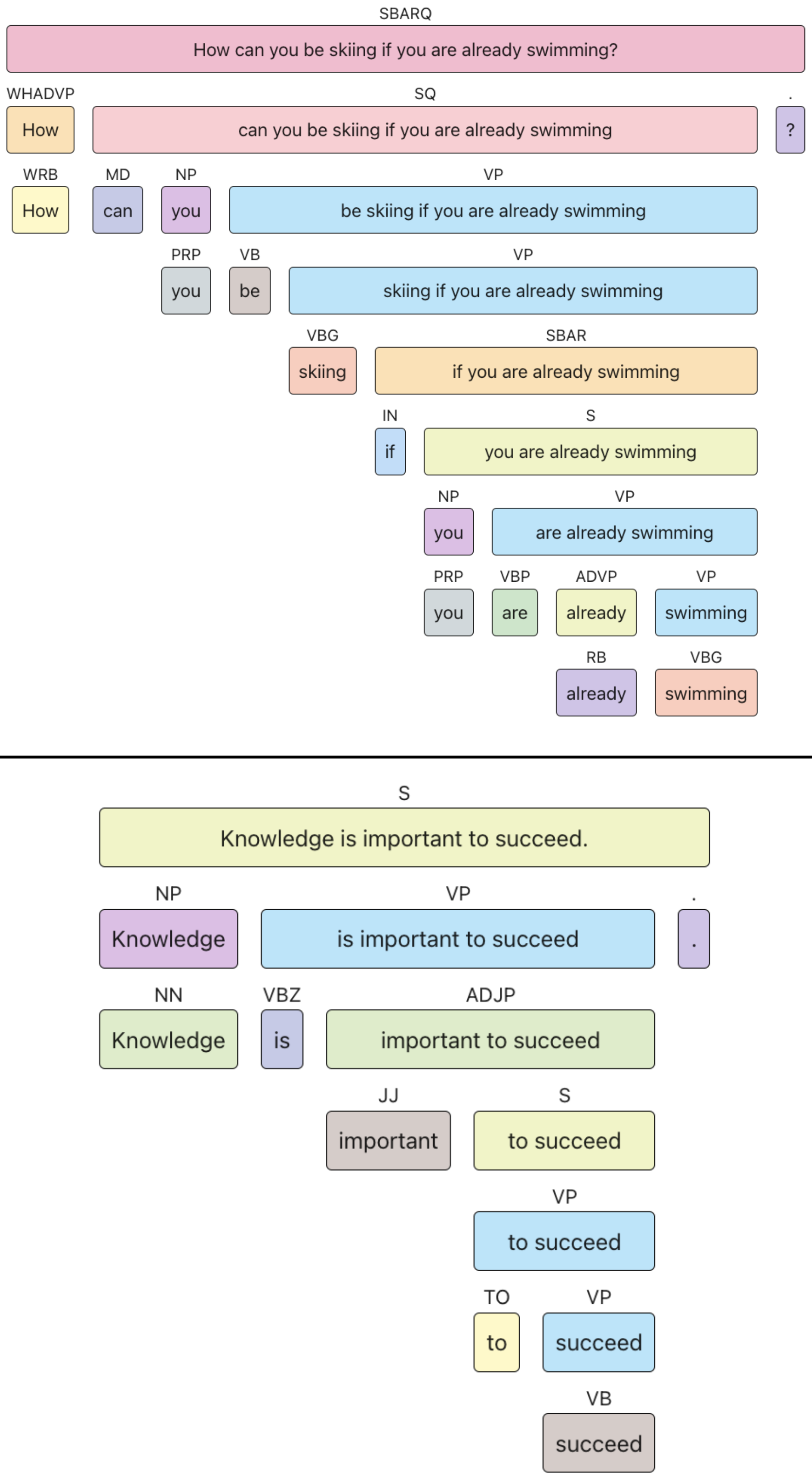


Figure 11: A comparison of the parse trees of two syntactically dissimilar documents. Top document: “How can you be skiing if you are already swimming?” Bottom document: “Knowledge is important to succeed.” FastKASSIM similarity score: 0.439; CASSIM similarity score: 0.679.

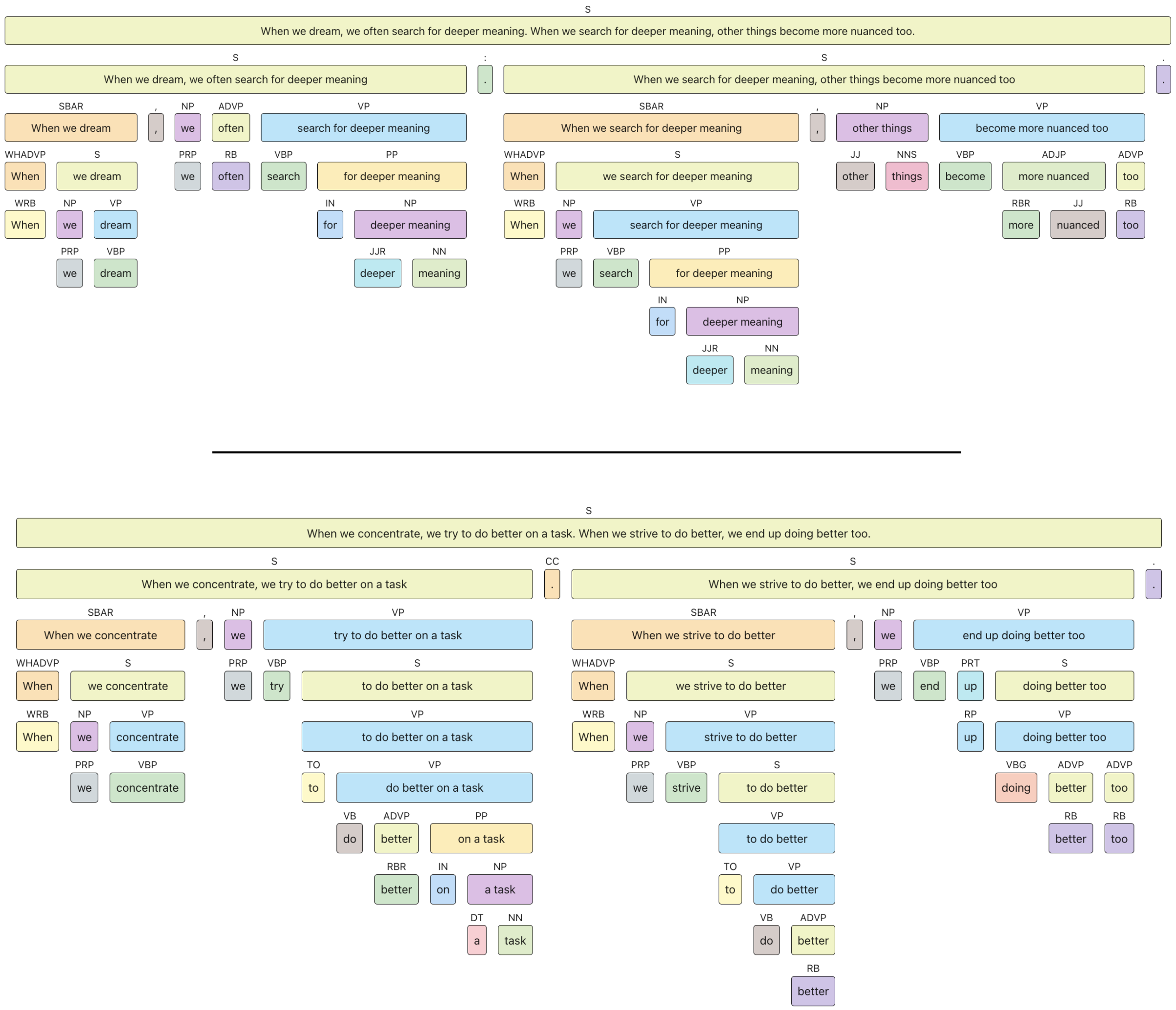


Figure 12: A comparison of the parse trees of two syntactically similar documents. Top document: “When we dream, we often search for deeper meaning. When we search for deeper meaning, other things become more nuanced too.” Bottom document: “When we concentrate, we try to do better on a task. When we strive to do better, we end up doing better too.” FastKASSIM similarity score: 0.656; CASSIM similarity score: 0.837.

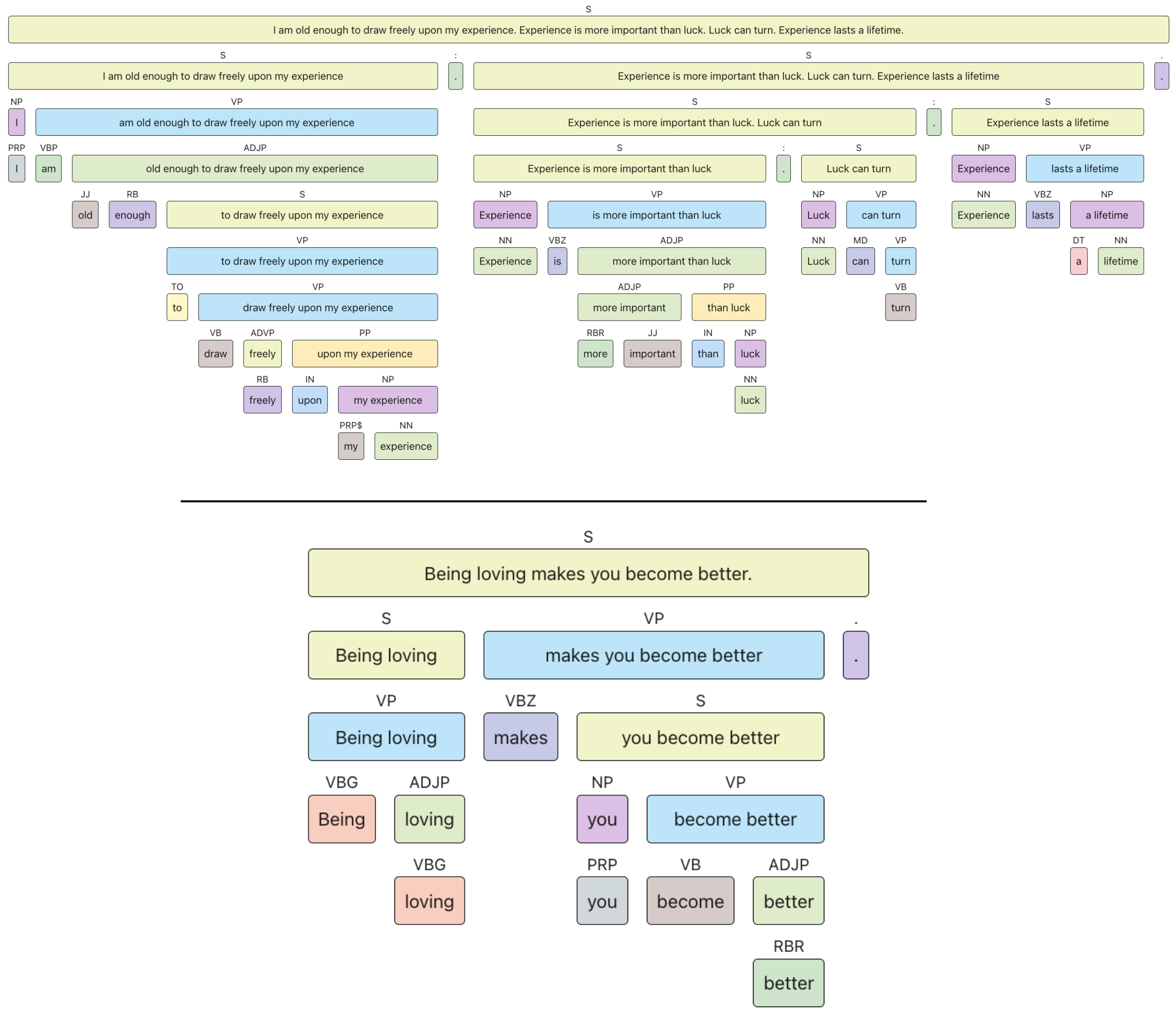


Figure 13: A comparison of the parse trees of two syntactically dissimilar documents. Top document: “I am old enough to draw freely upon my experience. Experience is more important than luck. Luck can turn. Experience lasts a lifetime.” Bottom document: “Being loving makes you become better.” FastKASSIM similarity score: 0.15; CASSIM similarity score: 0.924.

Friend-training: Learning from Models of Different but Related Tasks

Mian Zhang[†][*], **Lifeng Jin**[◇], **Linfeng Song**[◇], **Haitao Mi**[◇], **Xiabing Zhou**[†] and **Dong Yu**[◇]
[†]Soochow University, Suzhou, China
`mzhang2@stu.suda.edu.cn`, `zhouxiabing@suda.edu.cn`
[◇]Tencent AI Lab, Bellevue, WA, USA
`{lifengjin,lfsong,haitaomi,dyu}@tencent.com`

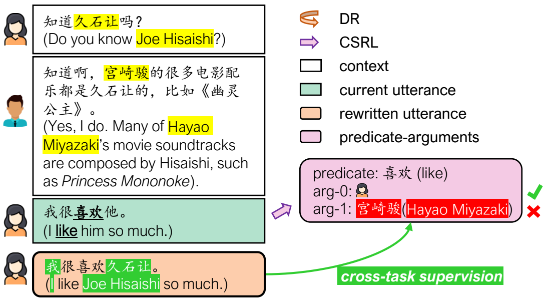
Abstract

Current self-training methods such as standard self-training, co-training, tri-training, and others often focus on improving model performance on a single task, utilizing differences in input features, model architectures, and training processes. However, many tasks in natural language processing are about different but related aspects of language, and models trained for one task can be great teachers for other related tasks. In this work, we propose **friend-training**, a *cross-task* self-training framework, where models trained to do different tasks are used in an iterative training, pseudo-labeling, and retraining process to help each other for better selection of pseudo-labels. With two dialogue understanding tasks, conversational semantic role labeling and dialogue rewriting, chosen for a case study, we show that the models trained with the friend-training framework achieve the best performance compared to strong baselines.

1 Introduction

Many different machine learning algorithms, such as self-supervised learning (Mikolov et al., 2013; Devlin et al., 2019; Liu et al., 2021), semi-supervised learning (Yang et al., 2021) and weakly supervised learning (Zhou, 2018), aim at using unlabeled data to boost performance. They have been of even greater interest recently given the amount of unlabeled data available. Self-training (Scudder, 1965) is one semi-supervised learning mechanism aiming to improve model performance through pseudo-labeling and has been successfully applied to computer vision (Lee et al., 2013; Chen et al., 2021), natural language processing (Dong and Schäfer, 2011; Bhat et al., 2021) and other fields (Wang et al., 2019; Kahn et al., 2020).

The main challenge of self-training is how to select high-quality pseudo-labels. Current self-training algorithms mainly focus on a single task when assessing the quality of pseudo-labels and suffer from gradual drifts of noisy instances (Zhang et al., 2021). This work is motivated by the observation that learning targets of tasks represent different properties of the inputs, and some properties are shared across the tasks which can be used as supervision from one task to another. Such properties include certain span boundaries in dependency and constituency parsing, and some emotion polarities in sentiment analysis and emotion detection. Two dialogue understanding tasks, conversational semantic role labeling (CSRL) and dialogue rewriting (DR), are also such a pair, with shared properties such as coreference and zero-pronoun resolution. As shown in Figure 1, the rewritten utterance can be used to generate cross-task supervision to the arguments of predicate 喜欢(like). We leverage the cross-task supervision from *friend tasks* – different but related tasks – as a great criterion for assessing the quality of pseudo-labels.


知道久石让吗？
(Do you know Joe Hisaishi?)

知道啊，宫崎骏的很多电影配乐都是久石让的，比如《幽灵公主》。
(Yes, I do. Many of Hayao Miyazaki's movie soundtracks are composed by Hisaishi, such as *Princess Mononoke*).

我很喜欢他。
(I like him so much.)

我很喜欢久石让。
(I like Joe Hisaishi so much.)

predicate: 喜欢 (like)
arg-0:
arg-1: 宫崎骏(Hayao Miyazaki)

Figure 1: An example of cross-task supervision between a CSRL parser and a DR system in a dialogue. 久石让(Joe Hisaishi) from the rewritten utterance provides cross-task supervision to 宫崎骏(Hayao Miyazaki), the predicted `arg-1` of 喜欢(like) from the CSRL parser, while 我(I) to the predicted `arg-0`.

In this work, we propose **friend-training**, the first *cross-task* self-training framework. Compared to single-task self-training, friend-training **exploits supervision from friend tasks for better selection of pseudo-labels**. To this end, two novel modules

---

[*] Work done when interning at Tencent AI Lab.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 232–247
May 2-6, 2023

are proposed: (1) a *translation matcher*, which maps the pseudo-labels of different tasks for one instance into the same space and computes a *matching score* representing the **cross-task matching degree of pseudo-labels from different tasks**; (2) an *augmented (instance) selector*, which leverages **both** the confidence of pseudo-labels from task-specific models and the matching score to select instances with pseudo-labels of high quality as new training data. We choose CSRL and DR as friend tasks to conduct a case study for friend-training, and specify the translation matcher and augmented selector for friend-training between these tasks. Experimental results of domain generalization and few-shot learning show friend-training surpasses both classical and state-of-the-art semi-supervised learning algorithms by a large margin. To summarize, contributions from this work include:

- We propose friend-training, the first cross-task self-training framework which exploits supervision from friend tasks for better selection of pseudo-labels in the iterative training process.
- We provide specific modeling of friend-training between CSRL and DR, with a novel translation matcher and a novel augmented selector.
- Extensive experiments with CSRL and DR demonstrate the effectiveness of friend-training, outperforming several strong baselines.

2 Related Work

Self-training Self-training (Scudder, 1965; Angluin and Laird, 1988; Abney, 2002; Lee et al., 2013) is a classical semi-supervised learning framework (Chapelle et al., 2009) which has been widely explored in recent years. The general idea of self-training is to adopt a trained model to pseudo-label easily acquired unlabeled data and use them to augment the training data to retrain the model iteratively. This paradigm shows promising effectiveness in a variety of tasks: including text classification (Mukherjee and Awadallah, 2020; Wang et al., 2020a), image classification (Xie et al., 2020; Zoph et al., 2020), machine translation (He et al., 2020) and model distillation (Mukherjee and Hassan Awadallah, 2020). Co-training (Blum and Mitchell, 1998) and tri-training (Zhou and Li, 2005) are similar iterative training frameworks to self-training but with a different number of models or considering different views of the training data, both of which see wide adoption in NLP (Mihalcea, 2004; McClosky et al., 2006; Wan, 2009; Li et al., 2014; Caragea et al., 2015; Lee and Chieu, 2021; Wagner and Foster, 2021). These frameworks aim at improving performance with multiple models trained on one task, without directly leveraging the benefit of supervision from related tasks.

Multi-task Learning Multi-task learning (Caruana, 1997; Yang et al., 2021) seeks to improve the learning performance of one task with the help of other related tasks, among which two lines of work are related to ours: (1) semi-supervised multi-task learning (Liu et al., 2007; Li et al., 2009) combines semi-supervised learning and multi-task learning. Liu et al. (2007) exploited unlabeled data by random walk and used a task clustering method for multi-task learning. Li et al. (2009) integrated active learning (MacKay, 1992) with the model in Liu et al. (2007) to retrieve data that are most informative for labeling. Although these works tried to utilize unlabeled data to enhance multi-task learning, our work differs from them in incorporating supervised signals among tasks to select high-quality pseudo-labels for updating models, which is an iterative training process without additional human annotation. (2) Task grouping (Kumar and III, 2012; Standley et al., 2020; Fifty et al., 2021) aims to find groups of related tasks and employs multi-task learning to each group of tasks, with one model for each group. Our work focuses on training single-task models, but task grouping techniques can be used to look for possible friend tasks.

Conversational Semantic Role Labeling CSRL is a task for predicting the semantic roles of predicates in a conversational context. Wu et al. (2021) leveraged relational graph neural networks (Schlichtkrull et al., 2018) to model both the speaker and predicate dependency, achieving some promising results. However, the current dataset (Xu et al., 2021) for CSRL is limited to mono-domain. High-quality labeled data for new domains are needed to empower more applicable CSRL models.

Dialogue Rewriting DR is commonly framed as a sequence-to-sequence problem which suffers large search space issue (Elgohary et al., 2019; Huang et al., 2021). To address it, Hao et al. (2021) cast DR to sequence labeling, transforming rewriting an utterance as deleting tokens from an utterance or inserting spans from the dialogue history into an utterance. Jin et al. (2022) improved the continuous span issue in (Hao et al., 2021) by first generating multiple spans for each token and slotted rules and then replacing a fixed number rules with spans.

3 Friend-training

Friend-training is an iterative training framework to jointly refine models of several friend tasks. Different from self-training, friend-training injects cross-task supervision into the selection of pseudo-labels. We first briefly describe self-training before presenting friend-training.

3.1 Self-training

Classic self-training aims at iteratively refining a model of a single task by using both labeled data and a large amount of unlabeled corpus. At each iteration, the model first assigns the unlabeled data with pseudo-labels. Subsequently, a set of the unlabeled instances with pseudo-labels is selected for training, presumably with information for better model generalization. Then cross-entropy of model predictions and labels on both gold and pseudo-labeled data is minimized to update the model:

$$L = \sum_{i=1}^{N} y_i \log \frac{y_i}{p_i} + \lambda \sum_{i=1}^{N'} y'_i \log \frac{y'_i}{p'_i}, \tag{1}$$

where the left term is the loss for the labeled data and the right for the unlabeled data while λ is a coefficient to balancing them; $N (N')$ is the number of instances, y (y') is the label and p (p') is the output probability of the model.

Self-training is usually limited to only one task, but there are thousands of NLP tasks already proposed and many of them are related. Models trained for one task can be great teachers for other related tasks. We explore this cross-task supervision in self-training by incorporating two novel modules introduced in subsection 3.2.

3.2 Friend-training

For friend-training with two tasks,[1] we have two classifiers f_a and f_b trained on two different tasks with labeled training sets $\mathcal{L}_a$ and $\mathcal{L}_b$, with expected accuracies η_a and η_b, respectively. The two datasets are created independently and the prediction targets of the two tasks are partially related through a pair of translation functions $\mathcal{F}_a : \hat{Y}_a \rightarrow \Sigma$ and $\mathcal{F}_b : \hat{Y}_b \rightarrow \Sigma$, where Σ is the set of possible sub-predictions that all possible predictions of the two tasks $\hat{Y}_a$ and $\hat{Y}_b$ can be reduced to. $|\hat{Y}_a| \geq |\Sigma|, |\hat{Y}_b| \geq |\Sigma|$. We assume that the translation functions are general functions with the expected probability of generating a translation $\epsilon_{\mathcal{F}} = \frac{1}{|\Sigma|}$. The translation functions are deterministic and always map the gold labels of the friend tasks for the same input to the same translation.

Both classifiers make predictions on the unlabeled set $\mathcal{U}$ at iteration k. Some instances $\mathcal{U}^k_{\mathcal{F}}$ with pseudo-labels are chosen as new training data based on the results of the translation functions, $\phi_a(x) = \mathcal{F}_a(f_a(x))$ and $\phi_b(x) = \mathcal{F}_b(f_b(x))$, and some selection criteria, such as total agreement. If total agreement is used as the selection criterion, the probability of erroneous predictions for f_a in these instances is

$$\begin{aligned} &\mathrm{Pr}_x[f_a(x) \neq f^*_a(x) | \phi_a(x) = \phi_b(x)] \\ =&1 - \frac{\eta_a \mathrm{Pr}_x[\phi_a(x) = \phi_b(x) | f_a(x) = f^*_a(x)]}{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)]}, \end{aligned} \tag{2}$$

with f^* being the optimal classifier.

Because both classifiers are very different due to training data, annotation guidelines, models, prediction targets, etc., being all different, the two classifiers are very likely to be independent of each other. Under this condition Equation 2 becomes

$$\begin{aligned} &1 - \frac{\eta_a(\eta_b + \epsilon_{\mathcal{F}}(1 - \eta_b))}{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)]} \\ =&1 - \frac{Z}{Z + \eta_b \epsilon_{\mathcal{F}}(1 - \eta_a) + E}, \end{aligned} \tag{3}$$

where $Z = \eta_a(\eta_b + \epsilon_{\mathcal{F}}(1 - \eta_b))$ and $E = \epsilon^2_{\mathcal{F}}(1 - \eta_a)(1 - \eta_b)$. We give the detailed derivation of Equation 2 and 3 in Appendix A.1. This indicates that the quality of the picked instances is negatively correlated with the number of false positive instances brought by the noisy translation $\eta_b \epsilon_{\mathcal{F}}(1 - \eta_a)$, and the number of matching negative instances E. When $\epsilon_{\mathcal{F}}$ is minimized by choosing translation functions with a sufficiently large co-domain Σ, the probability of error instances chosen when two classifiers agree approaches 0. This also indicates that even when $1 - \eta_a$ is large, i.e. f_a performs badly, if the co-domain is large, the error rate of the chosen instances can still be kept very low.[2] As the dependence between the

---

[1] We focus on the two-friend version of friend-training in this work, however, friend-training can easily be extended to more than two friends.

[2] Intuitively, this means independent classifiers trained to do different tasks are unlikely to predict the same but wrong sub-prediction for a given input, if the sub-prediction includes a large number of decisions.

two classifiers grows in training, the probability of error instances also increases. When they are completely dependent on each other, Equation 2 becomes $1 - \eta_a$, i.e. classic self-training.

Based on this formulation, two additional modules are needed: (1) a *translation matcher* that maps predictions of two models trained on different tasks into the same space and computes a matching score; (2) an *augmented (instance) selector* which selects instances with pseudo-labels for the classifiers considering both the matching score of the translated predictions and the model confidences.

Translation Matcher Given the prediction of models of two friend tasks $f_a(x)$ and $f_b(x)$, the translation matcher $\mathcal{M}$ leverages translation functions $\mathcal{F}_a$ and $\mathcal{F}_b$ to get the translated pseudo-labels and computes a matching score m for the pair of pseudo-labels, which represents the similarity of the pair in the translation space:

$$m_{a,b} = \mathcal{M}\left(\mathcal{F}_a(f_a(x)), \mathcal{F}_b(f_b(x))\right), \quad (4)$$

with total agreement being 1. This matching score serves as a criterion for the selection of high quality pseudo-labels with cross-task supervision.

Augmented Selector Apart from pseudo-label similarity, other information about pseudo-label quality can be found from model confidence, which self-training algorithms specifically utilize, to augment matching scores. The augmented selector considers both the confidence of the pseudo-labels from task models, denoted as $\{c_a, c_b\}$, and the matching scores:

$$q_\tau = \mathcal{S}_\tau(m_{a,b}, c_\tau), \quad (5)$$

where $q_\tau \in \{0, 1\}$ represents the selection result of the pseudo-label for task $\tau \in a, b$. Therefore, instances with low matching scores but high confidence may also be selected as the training data. The complete algorithm is shown in Algorithm 1.

Algorithm 1: Two-task friend-training

```
Input  : Labeled data sets for two friend tasks, $\mathcal{L}_a, \mathcal{L}_b$; an unlabeled data set $\mathcal{U}$; task models $f_a, f_b$.
Output : Refined $f_a, f_b$.
Pre-train $f_\tau$ with $\mathcal{L}_\tau$ ($\tau \in a, b$);
while not until the maximum iteration do
    $\mathcal{L}_a^u = \emptyset$; $\mathcal{L}_b^u = \emptyset$;
    for $z$ in $\mathcal{U}$ do
        Generate $f_a(z), f_b(z)$ and $c_a, c_b$;
        $m_{a,b} \leftarrow$ Equation 4;
        $q_a, q_b \leftarrow$ Equation 5;
        if $q_\tau = 1$ ($\tau \in a, b$) then
            $\mathcal{L}_\tau^u = \mathcal{L}_\tau^u + \{z, v_\tau\}$;
    end
    Update $f_\tau$ with $\mathcal{L}_\tau, \mathcal{L}_\tau^u$ by Equation 1 ($\tau \in a, b$);
end
Return $f_a, f_b$;
```

4 Friend Training between CSRL and DR

To verify the effectiveness of friend-training, we select two dialogue understanding tasks as friend tasks to conduct friend-training experiments for a case study: conversational semantic role labeling (CSRL) and dialogue rewriting (DR). While both require skills such as coreference and zero-pronoun resolution, the two tasks focus on different properties of the dialogue utterance: (1) CSRL focuses on extracting arguments of the predicates in the utterance from the whole dialogue history; (2) DR aims to rewrite the last turn of a dialogue to make it context-free and fluent by recovering all the ellipsis and coreference in the utterance. Figure 2 provides an overview of friend-training between the above two tasks. Next, we first introduce the task models and then specify the translation matcher and augmented selector for applying friend-training.

4.1 Task Models

Task Definition A dialogue consists of N temporally ordered utterances $\{u_1, ..., u_N\}$. (1) Given utterance u_t and K predicates $\{\text{pred}_1, ..., \text{pred}_K\}$ of u_t, a CSRL parser predicts spans from the dialogue as arguments for all predicates. (2) A dialogue rewriter rewrites u_t to make it context-free according to its context $\{u_1, ..., u_{t-1}\}$.

Dialogue Encoder We concatenate dialogue context $\{u_1, ..., u_{t-1}\}$ and the current utterance u_t as a sequence of tokens $\{x_1, ..., x_M\}$ and encode it with BERT (Devlin et al., 2019) to get the contextualized embeddings:

$$\mathbf{E} = \mathbf{e}_1, ..., \mathbf{e}_M = \text{BERT}(x_1, ..., x_M) \in \mathbb{R}^{H \times M}.$$

Encoders for CSRL and DR share no parameters, but for simplicity, we use the same notation $\mathbf{E}$ for their outputs.

Conversational Semantic Role Labeling With the contextualized embeddings, we further generate predicate-aware utterance representations $\mathbf{G} = \{\mathbf{g}_1, ..., \mathbf{g}_M\} \in \mathbb{R}^{H \times M}$ as Wu et al. (2021) by applying self-attention (Vaswani et al., 2017) to $\mathbf{E}$

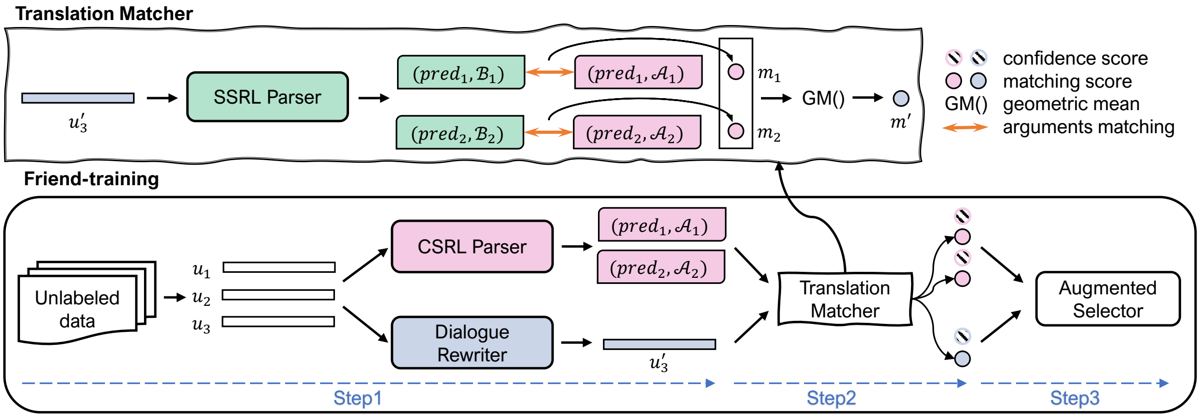


Figure 2: The overview of the friend-training process between CSRL and DR for one dialogue instance which has three utterances and the last utterance contains two predicates. Step1: the unlabeled dialogue is labeled by the CSRL parser and dialogue rewriter, resulting in predictions of arguments for the predicates (CSRL) and the rewritten utterance (DR), respectively. Step2: Pseudo-labels of both tasks are fed into the translation matcher to get their matching scores: the translation matcher first conducts sentence-level semantic role labeling (SSRL) on the rewritten utterance u_3' and then compares the results with those of the CSRL parser for matching scores. Step3: The threshold-based augmented selector makes the final decision of whether to add each pseudo-label to the training data considering both their confidence and matching scores. Best viewed in color.

with predicate-aware masking, where a token is only allowed to attend to tokens in the same utterance and tokens from the utterance containing the predicate:

$$\text{Mask}_{i,j} = \begin{cases} 1 & \text{if } u_{[i]} = u_{[j]} \text{ } or \text{ } u_{[j]} = u_{[pred]}, \\ 0 & \text{otherwise}, \end{cases}$$

where $u_{[m]}$ denotes the utterance containing token x_m and $u_{[pred]}$ denotes the one with the predicate.

The predicate-aware representations are then projected by a feed-forward network to get the distribution of labels for each token:

$$\mathbf{P}^c = \text{softmax}_{\text{column-wise}}(\mathbf{W}_c\mathbf{G} + \mathbf{b}_c) \in \mathbb{R}^{C \times M},$$

where $\mathbf{W}_c$ and $\mathbf{b}_c$ are learnable parameters and C is the number of labels. The labels follow BIO sequence labeling scheme: B-X and I-X respectively denote the token is the first token and the inner token of argument X, where O means the token does not belong to any argument. The output of the CSRL parser for K predicates are denoted as $\{\mathcal{A}_1, ..., \mathcal{A}_K\}$, where set $\mathcal{A}_k$ containing the arguments for pred_k.

Dialogue Rewriting Following Hao et al. (2021), we cast DR as sequence labeling. Specifically, a binary classifier on the top of $\mathbf{E}$ first determines whether to keep each token for in utterance u_t in the rewritten utterance:

$$\mathbf{P}^d = \text{softmax}_{\text{column-wise}}(\mathbf{W}_d\mathbf{E} + \mathbf{b}_d) \in \mathbb{R}^{2 \times M},$$

where $\mathbf{W}_d$ and $\mathbf{b}_d$ are learnable parameters. Next, a span of the context tokens is predicted to be inserted in front of each token. In practice, two self-attention layer (Vaswani et al., 2017) are adopted to calculate the probability of context tokens being the start index or end index of the span:

$$\mathbf{P}^{st} = \text{softmax}_{\text{column-wise}}(\text{Attn}_{st}(\mathbf{E})) \in \mathbb{R}^{M \times M},$$
$$\mathbf{P}^{ed} = \text{softmax}_{\text{column-wise}}(\text{Attn}_{ed}(\mathbf{E})) \in \mathbb{R}^{M \times M},$$

where $\mathbf{P}^{st}_{i,j}$ ($\mathbf{P}^{ed}_{i,j}$) denotes the probability of x_i being the start (end) index of the span for x_j. Then by applying argmax to $\mathbf{P}$, we could obtain the start and end indexes of the span for each token:

$$\mathbf{s}^{st} = \text{argmax}_{\text{column-wise}}(\mathbf{P}^{st}) \in \mathbb{R}^{M},$$
$$\mathbf{s}^{ed} = \text{argmax}_{\text{column-wise}}(\mathbf{P}^{ed}) \in \mathbb{R}^{M},$$

The probability of the span to be inserted in front of x_m is $\mathbf{P}^{st}_{\mathbf{s}^{st}_m,m} \times \mathbf{P}^{ed}_{\mathbf{s}^{ed}_m,m}$ when $\mathbf{s}^{st}_m \leqslant \mathbf{s}^{ed}_m$. When $\mathbf{s}^{st}_m > \mathbf{s}^{ed}_m$, it means no insertion. The output of the dialogue rewriter for u_t is denoted as u_t'.

4.2 Translation Matcher

To translate the outputs (pseudo-labels) from the CSRL parser $\{\mathcal{A}_1, ..., \mathcal{A}_K\}$ and the dialogue rewriter u_t' into a same space, we leverage a normal sentence-level semantic role parser with *fixed* parameters to greedily extract arguments from the rewritten utterance u_t' for the K predicates, denoted as $\{\mathcal{B}_1, ..., \mathcal{B}_K\}$ (Appendix A.5 shows an example).

So the common target space Σ is the label space of CSRL, which is large enough to make the error rate of chosen instances keep very low (see the analysis in subsection 3.2). The matching score $m_k \in [0, 1]$ for pred$_k$ is calculated based on the edit distance between $\mathcal{A}_k$ and $\mathcal{B}_k$:

$$m_k = 1 - \frac{\text{dist}(\oplus\mathcal{A}_k, \oplus\mathcal{B}_k)}{\max(\text{len}(\oplus\mathcal{A}_k), \text{len}(\oplus\mathcal{B}_k))},$$

where dist() calculates the edit distance between two strings, len() returns the length of a string and $\oplus\mathcal{A}_k$ denotes the concatenation of arguments in set $\mathcal{A}_k$ in a predefined order of arguments[3] (empty strings means arguments do not exist). Furthermore, we obtain the overall matching score $m' \in [0, 1]$ for the rewritten utterance u'_t as follows:

$$m' = \text{GM}(m_1, ..., m_K),$$

where GM() represents the geometric mean.

4.3 Augmented Selector

The augmented selector selects high-quality pseudo-labels according to both their matching scores and confidence. For CSRL, we calculate the confidence score for each predicate based on the output of the softmax layer. Specifically, we obtain the confidence of an argument for pred$_k$ by multiplying the probability of its tokens, denoted as $\{a_{k1}, ..., a_{k|\mathcal{A}_k|}\}$. We then use the geometric mean of all the confidence of arguments belonging to pred$_k$ as the confidence for pred$_k$. The overall score $s_k \in [0, 1]$ for pred$_k$ is calculated as follows:

$$s_k = \alpha\text{GM}(a_{k1}, ..., a_{k|\mathcal{A}_k|}) + (1 - \alpha)m_k,$$

where hyper-parameter α gives a balance between the matching score and the confidence. For DR, we multiply the probabilities of spans to be inserted and of decisions on whether to keep tokens or not as the model confidence of u'_t, denoted as b_t. The overall score $r_t \in [0, 1]$ of u'_t is as follows:

$$r_t = \beta b_t + (1 - \beta)m',$$

where a larger value of hyper-parameter β places more importance on the model confidence. α and β are set to be 0.2 for both tasks in the experiments.

Pick thresholds are set for s_k and r_t to control the number and quality of selected pseudo-labels. We analyze the effects of different values of thresholds in subsection 5.4.

5 Experiments

5.1 Setup

Datasets We use five dialogue datasets in our experiments with domains spanning movies, celebrities, book reviews, products, and social networks. For CSRL, we use DuConv (Xu et al., 2021) and WeiboCSRL and for DR, REWRITE (Su et al., 2019) and RESTORATION (Pan et al., 2019). The datasets of the same task differ in domains and sizes. WeiboCSRL is a newly annotated CSRL dataset for out-of-domain testing purposes. Moreover, we use LCCC-base (Wang et al., 2020b) as the unlabeled corpus, which is a large-scale Chinese conversation dataset with 79M rigorously cleaned dialogues from various social media. More details on the annotation of WeiboCSRL and the properties of the datasets could be found in Appendix A.2.

Experiment Scenarios Our main experiments involve two scenarios. (1) Domain generalization: we use DuConv as the training data in the source domain and WeiboCSRL for out-of-domain evaluation, while for DR, REWRITE is used for training and RESTORATION for evaluation. (2) Few-shot learning: we randomly select 100 cases from DuConv and REWRITE as the training data for CSRL and DR, respectively, and conduct in-domain evaluation, which means models of both the tasks are co-trained with only a few samples of each task. The unlabeled data for both scenarios are 20k dialogues extracted from LCCC-base. Implementation details are provided in Appendix A.3.

Evaluation We follow Wu et al. (2021) to report precision (Pre.), recall (Rec.), and F1 of the arguments for CSRL and Hao et al. (2021) to report word error rate (WER) (Morris et al., 2004), Rouge-L (R-L) (Lin, 2004) and the percent of sentence-level exact match (EM) for DR.

5.2 Baselines

We compare friend-training with six semi-supervised training paradigms: two standard techniques such as standard self-training (SST) (Scudder, 1965) and standard co-training (SCoT) (Blum and Mitchell, 1998), as well as four recent methods such as mean teacher (MT) (Tarvainen and Valpola, 2017), cross pseudo supervision (CPS) (Chen et al., 2021), self-training with batch reweighting (STBR) (Bhat et al., 2021) and self-teaching (STea) (Yu et al., 2021). See Appendix A.4 for more details.

---

[3] Argument concatenating order: `ARG0`, `ARG1`, `ARG2`, `ARG3`, `ARG4`, `ARGM-TMP`, `ARGM-LOC`, `ARGM-PRP`

	WeiboCSRL			RESTORATION		
Method	**Pre.**	**Rec.**	**F1**	**R-L**	**EM**	**WER(⇓)**
Base	57.97	54.47	56.16	82.78	25.25	28.69
Multitask-Base	53.66	54.32	53.99	81.68	22.49	32.44
SST (Scudder, 1965)	60.85	56.54	58.62	85.22	32.97	22.22
MT (Tarvainen and Valpola, 2017)	58.42	55.71	57.03	83.76	28.82	26.49
CPS (Chen et al., 2021)	60.34	52.87	56.36	85.60	32.68	22.78
SCoT (Blum and Mitchell, 1998)	57.33	54.13	55.69	84.51	29.25	24.87
STBR (Bhat et al., 2021)	60.77	58.04	59.38	85.79	33.78	23.30
STea (Yu et al., 2021)	60.10	55.13	57.50	85.75	34.23	22.17
FDT (Ours)	**65.29(↑4.44)**	**58.63(↑2.09)**	**61.78(↑3.16)**	**86.82(↑1.60)**	**38.22(↑5.25)**	**20.31(↑1.91)**

(a) Domain generalization for models trained with DuConv and REWRITE.

	DuConv			REWRITE		
Method	**Pre.**	**Rec.**	**F1**	**R-L**	**EM**	**WER(⇓)**
Base	29.50	21.90	25.14	73.44	3.60	39.98
Multitask-Base	22.43	20.63	21.49	78.97	11.70	40.46
SST (Scudder, 1965)	34.16	27.49	30.46	80.93	27.80	31.02
MT (Tarvainen and Valpola, 2017)	36.32	30.69	33.27	81.66	33.00	31.66
CPS (Chen et al., 2021)	37.14	29.47	32.86	79.56	23.30	32.60
SCoT (Blum and Mitchell, 1998)	38.37	26.15	31.10	78.58	22.31	33.79
STBR (Bhat et al., 2021)	32.37	25.21	28.34	82.37	29.80	30.31
STea (Yu et al., 2021)	39.34	28.78	33.25	83.04	31.57	30.36
FDT (Ours)	**40.41(↑6.25)**	30.82(↑3.33)	34.97(↑4.51)	82.83(↑1.90)	34.20(↑6.40)	27.87(↑3.15)
FDT-S (Ours)	40.12	**33.41**	**36.46**	**83.11**	**37.10**	**26.88**
Fully-trained *Base*	*69.83*	*68.53*	*69.17*	*89.47*	*52.30*	*20.54*

(b) Few-shot learning for models trained with DuConv and REWRITE.

Table 1: Test results for domain generalization and few-shot learning. Base denotes the task models trained with data from a single task. Multitask-Base denotes the base model of CSRL and DR sharing the same dialogue encoder. Results are averaged across three runs. ⇓ means lower is better. For few-shot learning, performance of the base models trained with the full training set from the single task is provided for reference.

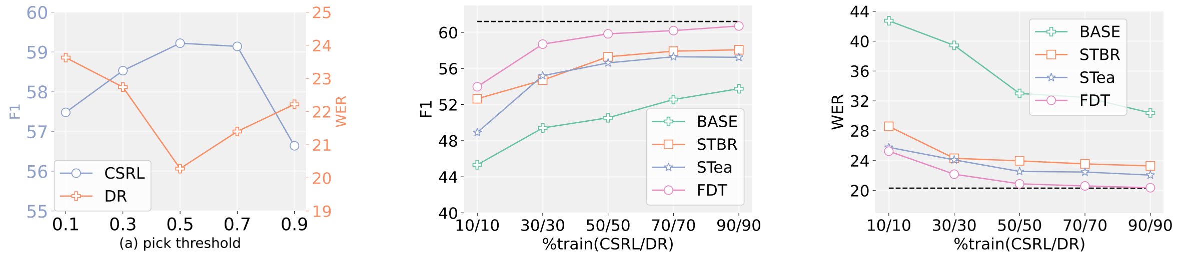
5.3 Main Results

Table 1 shows the comparison between friend-training (FDT) and the baselines mentioned in subsection 5.2. FDT achieves the best overall performance over the baselines by significant margins in both domain generalization and few-shot learning scenarios, which demonstrates the effectiveness of FDT in different experimental situations to utilize large unlabeled corpora. Moreover, we show the absolute improvements of FDT over SST in parentheses (↑). As we could see, in few-shot learning, FDT obtain 4.51 and 3.15 higher absolute points over SST on F1 of DuConv and WER of REWRITE, respectively, than those of domain generalization, which are 3.16 and 1.91 points, revealing that FDT could realize its potential easier in few-shot learning. Besides, for few-shot learning, we further consider the situation where a full-trained base model from the friend task is available, denoted as FDT-S. As we could see, when the target task is CSRL, FDT-S makes a gain of 1.49 points on F1 over FDT and when the target task is DR, FDT-S outperforms FDT on WER by 0.99 points and EM by 2.90 points, indicating that more reliable supervision from friend task could further enhance the few-shot learning of the target task.

5.4 Analysis

In this section, we conduct experiments to analyze how selected parameters and settings interact with model performance in FDT.

Pick Thresholds We vary the pick thresholds of CSRL and DR in domain generalization scenario and track the model performance: we fix the pick threshold of the friend task to the best (see Appendix A.3) when varying that of the evaluating task. As illustrated in Figure 3a, when the thresholds increase gradually, the models become better with higher F1 for CSRL and lower WER for DR. We attribute this to wrong pseudo-labels being filtered out by the augmented selector of FDT. Then the model performances hit the peaks and drop as the thresholds keep increasing in the interval of high values, which is owed to high thresholds producing insufficient pseudo-labels for iterative train-

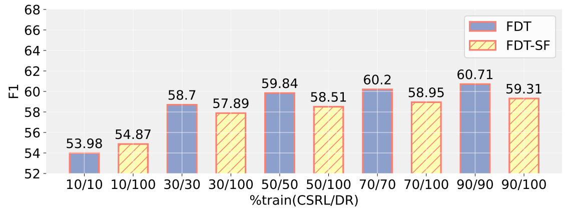


(a) The effect of pick thresholds.

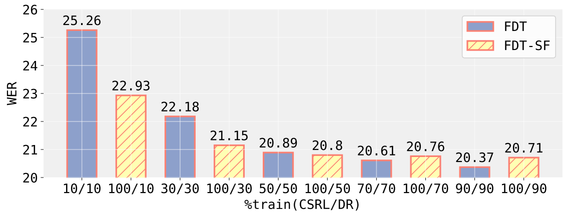
(b) F1 on CSRL test set.

(c) WER on DR test set.

Figure 3: Sub-figures (b) and (c) show the model performance of the comparing methods with different strengths of base models; the dashed horizontal line represents the performance of FDT with a fully trained base model.



(a) The performacne on CSRL test set.



(b) The performance on DR test set.

Figure 4: The role of co-updating in friend-training.

ing. Automatically choosing proper pick thresholds is worth to be explored in the future.

The Strength of Base Model To understand and compare how performance of models before friend-training or self-training influences their final performance, we compare STBR, STea and FDT with the base models trained on different percentages of labeled data in the source domain when evaluating on out-domain testing data. Specifically, we follow domain generalization settings and use a variable percentage of labeled data to conduct experiments.

For CSRL and DR, respectively, we set the amount of labeled data as $\{10\%/10\%, 30\%/30\%, 50\%/50\%, 70\%/70\%, 90\%/90\%\}$. The results are shown in Figure 3b and Figure 3c. We can see that all the methods adopting self-training to make use of unlabeled data surpass the base model by a significant margin, whether when given a weak or strong base model, demonstrating the effectiveness of self-training paradigm. Moreover, FDT achieves the best results across the evaluating percentages of labeled data: when the base model has a good amount of training data, such as those trained on 30% labeled data and above, the performance of FDT is significantly better than STBR and STea, proving that FDT leverages the features learned from labeled data more effectively with cross-task supervision.

The Role of Co-updating We also explore the case where one of the models of the friend tasks is fully trained and does not have to be updated. We consider FDT-SF, FDT with a *fixed* fully trained base model from the friend task in domain generalization[4]. As illustrated in Figure 4, FDT-SF surpasses FDT when given a weak base model for the evaluating task because of the strong supervision from the friend task. However, FDT outperforms FDT-SF when the evaluating task is given a fairly-trained model, which demonstrates the benefits of co-updating the models in friend-training.

6 Conclusion

We propose friend-training, the first cross-task self-training framework, which leverages supervision from friend tasks for better selection of pseudo-labels. Moreover, we provide specific modeling of friend-training between conversational semantic role labeling and dialogue rewriting. Experiments on domain generalization and few-shot learning scenarios demonstrate the promise of friend-training, which outperforms prior classical or state-of-the-art semi-supervised methods by substantial margins.

---

[4] Specifically, when the evaluating task is CSRL, the amount of labeled data for the two tasks are set as $\{10\%/100\%, 30\%/100\%, 50\%/100\%, 70\%/100\%, 90\%/100\%\}$, and when the evaluating task is DR, $\{100\%/10\%, 100\%/30\%, 100\%/50\%, 100\%/70\%, 100\%/90\%\}$.

Limitations

We showed how the friend-training strategy can be applied to two dialogue understanding tasks in the case study here, but many other task pairs or task sets can be examined to fully explore the generality of the approach. Identifying friend tasks depends on expert knowledge in this work, but approaches for task grouping and task similarity may be used to automatically discover friend tasks. Besides, with the proliferation of cross-modal techniques, tasks of different modalities are expected to act as friend tasks as well. Also, designing translation functions and matchers for friend tasks in the friend-training framework requires an understanding of the relationship between the friend tasks, but prompting and model interpretability methods could potentially be applied for easing this process.

Acknowledgement

We thank the anonymous reviewers for their helpful comments and the support of National Nature Science Foundation of China (No.62176174).

References

Steven Abney. 2002. Bootstrapping. In *the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 360–367.

Dana Angluin and Philip Laird. 1988. Learning from noisy examples. *Mach. Learn.*, 2(4):343–370.

Meghana Moorthy Bhat, Alessandro Sordoni, and Subhabrata Mukherjee. 2021. Self-training with few-shot rationalization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10702–10712, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

Cornelia Caragea, Florin Bulgarov, and Rada Mihalcea. 2015. Co-Training for topic classification of scholarly data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2357–2366, Lisbon, Portugal. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542.

Xiaokang Chen, Yuhui Yuan, Gang Zeng, and Jingdong Wang. 2021. Semi-supervised semantic segmentation with cross pseudo supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2613–2622.

Michael Collins. 2003. Head-Driven statistical models for natural language parsing. *Comput. Linguist.*, 29(4):589–637.

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

Cailing Dong and Ulrich Schäfer. 2011. Ensemble-style self-training on citation classification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 623–631, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can you unpack that? learning to rewrite questions-in-context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5918–5924, Hong Kong, China. Association for Computational Linguistics.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34.

Jie Hao, Linfeng Song, Liwei Wang, Kun Xu, Zhaopeng Tu, and Dong Yu. 2021. RAST: Domain-robust dialogue rewriting as sequence tagging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4913–4924, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Han He and Jinho D. Choi. 2021. The stem cell hypothesis: Dilemma behind multi-task learning with transformer encoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5555–5577, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mengzuo Huang, Feng Li, Wuhe Zou, Hongbo Zhang, and Weidong Zhang. 2021. Sarg: A novel semi autoregressive generator for multi-turn incomplete utterance restoration. In *Proc. 35th AAAI Conf. Artif.*

Intell., 33rd Conf. Innovat. Appl. Artif. Intell., 11th Symp. Educat. Adv. Artif. Intell., pages 13055–13063.

Lifeng Jin and William Schuler. 2019. Variance of average surprisal: A better predictor for quality of grammar from unsupervised PCFG induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2453–2463, Florence, Italy. Association for Computational Linguistics.

Lisa Jin, Linfeng Song, Lifeng Jin, Dong Yu, and Daniel Gildea. 2022. Hierarchical context tagging for utterance rewriting.

Jacob Kahn, Ann Lee, and Awni Hannun. 2020. Self-training for end-to-end speech recognition. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 7084–7088. IEEE.

Abhishek Kumar and Hal Daumé III. 2012. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Jian Yi David Lee and Hai Leong Chieu. 2021. Co-training for commit classification. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 389–395, Online. Association for Computational Linguistics.

Hui Li, Xuejun Liao, and Lawrence Carin. 2009. Active learning for semi-supervised multi-task learning. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1637–1640. IEEE.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014. Ambiguity-aware ensemble training for semi-supervised dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 457–467, Baltimore, Maryland. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Qiuhua Liu, Xuejun Liao, and Lawrence Carin. 2007. Semi-supervised multitask learning. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 937–944. Curran Associates, Inc.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. 2021. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

David JC MacKay. 1992. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604.

David M Magerman. 1995. Statistical Decision-Tree models for parsing. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 276–283, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective Self-Training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

Rada Mihalcea. 2004. Co-training and self-training for word sense disambiguation. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 33–40, Boston, Massachusetts, USA. Association for Computational Linguistics.

Tomas Mikolov, Wen-Tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Andrew Cameron Morris, Viktoria Maier, and Phil Green. 2004. From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. In *Eighth International Conference on Spoken Language Processing*.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. XtremeDistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2221–2234, Online. Association for Computational Linguistics.

Zhufeng Pan, Kun Bai, Yan Wang, Lianqiang Zhou, and Xiaojiang Liu. 2019. Improving open-domain dialogue systems via multi-turn incomplete utterance restoration. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1824–1833, Hong Kong, China. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Henry Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Trevor Standley, Amir Roshan Zamir, Dawn Chen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning? In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9120–9132. PMLR.

Hui Su, Xiaoyu Shen, Rongzhi Zhang, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance ReWriter. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 22–31, Florence, Italy. Association for Computational Linguistics.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1195–1204.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Joachim Wagner and Jennifer Foster. 2021. Revisiting tri-training of dependency parsers.

Xiaojun Wan. 2009. Co-Training for Cross-Lingual sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 235–243, Suntec, Singapore. Association for Computational Linguistics.

Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Attributed graph clustering: A deep attentional embedding approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3670–3676. ijcai.org.

Yaqing Wang, Subhabrata Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed Hassan Awadallah. 2020a. Adaptive self-training for few-shot neural sequence labeling. *ArXiv preprint*, abs/2010.03680.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020b. A large-scale chinese short-text conversation dataset. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 91–103. Springer.

Han Wu, Kun Xu, and Linqi Song. 2021. CSAGN: Conversational structure aware graph network for conversational semantic role labeling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2312–2317, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10684–10695. IEEE.

Kun Xu, Han Wu, Linfeng Song, Haisong Zhang, Linqi Song, and Dong Yu. 2021. Conversational semantic role labeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2465–2475.

Nianwen Xue. 2006. Semantic role labeling of nominalized predicates in Chinese. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 431–438, New York City, USA. Association for Computational Linguistics.

Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. 2021. A survey on deep semi-supervised learning. *ArXiv preprint*, abs/2103.00550.

Dian Yu, Kai Sun, Dong Yu, and Claire Cardie. 2021. Self-teaching machines to read and comprehend with large-scale multi-subject question-answering data. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 56–68, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

Zhi-Hua Zhou. 2018. A brief introduction to weakly supervised learning. *National science review*, 5(1):44–53.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on knowledge and Data Engineering*, 17(11):1529–1541.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. 2020. Rethinking pre-training and self-training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

A Appendix

A.1 Error rates

We have two classifiers f_a and f_b trained on two different tasks with labeled training sets $\mathcal{L}_a$ and $\mathcal{L}_b$, with expected accuracies η_a and η_b, respectively. The prediction targets of the two tasks are partially related through a pair of translation functions $\mathcal{F}_a : \hat{Y}_a \to \Sigma$ and $\mathcal{F}_b : \hat{Y}_b \to \Sigma$, where Σ is the set of possible sub-predictions that all possible predictions of the two tasks $\hat{Y}_a$ and $\hat{Y}_b$ can be reduced to. $|\hat{Y}_a| \geq |\Sigma|, |\hat{Y}_b| \geq |\Sigma|$. The sub-predictions can be a part of the whole prediction targets for both tasks, or some lossy transformation of the prediction targets. For example, a sub-prediction for a POS-tagging task can be the POS tag of the first word (a part of the prediction) or the number of the NN tag in the whole prediction sequence (a transformation of the prediction). We assume that the translation functions are general functions with the expected probability of generating a translation $\epsilon_{\mathcal{F}} = \frac{1}{|\Sigma|}$; they are deterministic and always map the gold labels of the friend tasks for the same input to the same translation. Both classifiers make predictions on the unlabeled set $\mathcal{U}$ at iteration k. Some instances $\mathcal{U}_{\mathcal{F}}^{k}$ with pseudo-labels are chosen as new training data based on the results of the translation functions, $\phi_a(x) = \mathcal{F}_a(f_a(x))$ and $\phi_b(x) = \mathcal{F}_b(f_b(x))$, and some selection criteria, such as total agreement. If total agreement is used as the selection criterion, the probability of erroneous predictions for f_a in these instances is

$$\begin{aligned}
&\mathrm{Pr}_x[f_a(x) \neq f_a^*(x)|\phi_a(x) = \phi_b(x)] \\
=&1 - \mathrm{Pr}_x[f_a(x) = f_a^*(x)|\phi_a(x) = \phi_b(x)] \\
=&1 - \mathrm{Pr}_x[f_a(x) = f_a^*(x)]\cdot \\
&\frac{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)|f_a(x) = f_a^*(x)]}{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)]} \\
=&1 - \eta_a\cdot \\
&\frac{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)|f_a(x) = f_a^*(x)]}{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)]}, \qquad (6)
\end{aligned}$$

with f^* being the optimal classifier. If we consider the two classifiers very likely to be independent from each other, then the probability of the translation of the predictions from the two classifiers being the same given the prediction from classifier f_a is correct, which is $\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)|f_a(x) = f_a^*(x)]$, is the sum of the probability of the classifier f_b making the correct prediction η_b and the probability of an erroneous translation of the wrong prediction $\epsilon_{\mathcal{F}}(1-\eta_b)$. The probability of the translations matching $\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)]$ has four situations: both predictions of the two classifiers are correct $\eta_a\eta_b$; $f_a(x)$ is correct but $f_b(x)$ is wrong and being translated erroneously $\eta_a\epsilon_{\mathcal{F}}(1-\eta_b)$; $f_b(x)$ is correct but $f_a(x)$ is wrong and being translated erroneously $\eta_b\epsilon_{\mathcal{F}}(1-\eta_a)$; both $f_a(x)$ and $f_b(x)$ are wrong but matching in the translation space $\epsilon_{\mathcal{F}}^2(1-\eta_a)(1-\eta_b)$. Under these conditions Equation 6 becomes

$$\begin{aligned}
&1 - \frac{\eta_a(\eta_b + \epsilon_{\mathcal{F}}(1-\eta_b))}{\mathrm{Pr}_x[\phi_a(x) = \phi_b(x)]} \\
=&1 - \frac{Z}{Z + \eta_b\epsilon_{\mathcal{F}}(1-\eta_a) + E}, \qquad (7)
\end{aligned}$$

where $Z = \eta_a(\eta_b + \epsilon_{\mathcal{F}}(1-\eta_b))$ and $E = \epsilon_{\mathcal{F}}^2(1-\eta_a)(1-\eta_b)$ which shows that the term $\eta_b\epsilon_{\mathcal{F}}(1-\eta_a)+E$ needs to be small to make the probability of matching translations with predictions being wrong small. This indicates that the quality of the picked instances based on the total agreement criterion is negatively correlated with the number of false positive instances brought by the noisy translation $\eta_b\epsilon_{\mathcal{F}}(1-\eta_a)$, and the number of matching negative instances E. $\epsilon_{\mathcal{F}}$ can be minimized by choosing translation functions with a sufficiently large co-domain Σ, which means that when the translation space is large enough, it is unlikely that the two classifiers totally agree in the translation space but do not agree in their own prediction target spaces. So the probability of them agreeing and making correct predictions is much larger than agreeing but making incorrect predictions while the probability of error instances chosen when two classifiers agree approaches 0, indicating that even when $1-\eta_a$ is large, i.e. f_a performs badly, if the co-domain is large, the error rate of the chosen instances can still be kept very low.

A.2 Datasets

Annotation Procedure of WeiboCSRL The dialogues we use for CSRL annotation are extracted from LCCC-base (Wang et al., 2020b), which consists of at least 4 turns and 80 total characters to assure enough context for CSRL and DR. These dialogues and those used as unlabeled data for experiments in section 5 are from different parts of LCCC-base. For each dialogue, we annotate the predicates in the last utterance with the guidance of frame files of Chinese Proposition Bank[5]. For each predicate, the arguments we annotate are numbered arguments `ARG0-ARG4` and adjuncts `ARGM-LOC`, `ARGM-MNR`, `ARGM-TMP` and `ARGM-NEG`, whose definitions are shown in (Xue, 2006). `ARGM-MNR` is not included for evaluation in section 5 because annotation of `ARGM-MNR` is lacking in DuConv, the training data for CSRL. In the end, we obtain 3891 annotated predicates.

Dataset Details Table 2 shows the statistic of the datasets used in the experiments. DuConv (Xu et al., 2021) focuses on movies and celebrities and we adopt the same train/dev/test splitting as Xu et al. (2021). REWRITE (Su et al., 2019) contains 20K dialogues with a wide range of topics crawled from Chinese social media platforms; the last utterance of each dialogue is rewritten to recover all co-referred and omitted information. RESTORATION (Pan et al., 2019) contains dialogues from Douban[6], most of which are book, movie or product reviews. Compared with REWRITE, it contains more annotated dialogues, but around 40% of the last utterances require no rewriting.

	Domain	**#Instance**(train/dev/test)
DuConv	movies and celebrities	23361 / 2852 / 2977
WeiboCSRL	social media	- / 1945 / 1946
REWRITE	social media	16925 / 1000 / 1000
RESTORATION	book, movie and product reviews	- / 5000 / 5000

Table 2: Dataset statistics.

A.3 Implementation Details

Dataset configuration of the tasks for the experimental scenarios are shown in Table 3.

Preprocessing Details The maximum length of the input dialogue is set to 125. We transform the word-based labeling of DuConv to character-based labeling and we use the scripts[7] provided by Hao et al. (2021) to generate token-level annotations for sequence-labeling-based DR. For unlabeled data, we discard dialogues with less than 4 turns to guarantee sufficient context for CSRL and DR.

Model Details We use pretrained BERT[8] (Devlin et al., 2019) as the dialogue encoder for CSRL and DR. Both the values of hyper-parameter α and β are set to 0.2 and the pick thresholds are set to 0.6. We choose a state-of-the-art sentence-level semantic role labeling (SSRL) parser[9] for the translation matcher which follows the same structure as (He and Choi, 2021).

Training Details We adopt AdamW (Loshchilov and Hutter, 2019) to optimize models with a learning rate of 4e-5 and batch size of 16. We use $\lambda = 1$ to balance the loss of labeled and unlabeled data.

	Task	**Train**	**Dev&Test**
DG	CSRL	DuConv (train)	WeiboCSRL (dev,test)
	DR	REWRITE (train)	RESTORATION (dev,test)
FSL	CSRL	DuConv (100 cases)	DuConv (dev,test)
	DR	REWRITE (100 cases)	REWRITE (dev,test)

Table 3: Dataset configuration of domain generalization (DG) and few-shot learning (FSL).

A.4 Baselines

Standard self-training (Scudder, 1965) generates pseudo-labels to unlabeled data with a base model and uses them to train a new base model, which is repeated until convergence. Standard co-training (Blum and Mitchell, 1998) is similar to Standard self-training, but with two different base models dealing with the same task, generating pseudo-labels and adding the trusted ones for iterative training. Mean teacher (Tarvainen and Valpola, 2017) maintains a teacher model on the fly, whose weights are the exponential moving average of the weights of a student model across iterations. Cross pseudo supervision (Chen et al., 2021), a state-of-the-art variant of self-training, maintains two networks with different initialization; the pseudo-label of one network is used to supervise the other network. Self-training with batch reweighting (Bhat et al., 2021) is a state-of-the-art self-training method that reweights the pseudo-labels in a batch when training according to the confidence from the teacher model. Self-teaching (Yu et al., 2021), a state-of-the-art semi-supervised method that sequentially trains a junior teacher, a senior teacher and an expert student to leverage the unlabeled data.

For the hyper-parameters of the baselines, we keep the common hyper-parameters, such as learning rate, batch size, optimizer, and so on, the same as our proposed method. And we adopt the values of method-specific hyper-parameters used in the original papers, such as the merging weight of soft and hard labels of self-teaching and the smoothing parameter for updating of mean teacher.

---

[5] `https://verbs.colorado.edu/chinese/cpb/`

[6] `https://www.douban.com`

[7] `https://github.com/freesunshine0316/RaST-plus`

[8] `https://huggingface.co/bert-base-chinese`

[9] `https://github.com/hankcs/HanLP`

Context:	ch: [A]我有一个非常喜欢的女明星。[B]她叫什么名字？[A]布蕾克·莱弗利。[B]她很有名吗？ en: [A] I have a favorite actress. [B] What’s her name? [A] Blake Lively. [B] Is she famous?	
Current utterance	ch: [A]她是一个非常受关注的女明星。 en: [A] She is a actress attracting much attention.	
Rewritten utterance	ch: [A] 布蕾克·莱弗利是一个非常受关注的女明星。 en: [A] Blake Lively is a actress attracting much attention.	
Predicates	是(is)	受(attract)
CSRL	ch: ARG1: 一个非常受关注的女明星 en: ARG1: a actress attracting much attention	ARG0: 布蕾克·莱弗利, ARG1: 关注 ARG0: Blake Lively, ARG1: attention
SSRL	ch: ARG0: 布蕾克·莱弗利, ARG1: 一个非常受关注的女明星 en: ARG0: Blake Lively, ARG1: a actress attracting much attention	ARG0: 布蕾克·莱弗利, ARG1: 关注 ARG0: Blake Lively, ARG1: attention
Predicate matching score	0.61	1.0
Predicate confidence	0.95	0.54
Predicate overall score	0.67	0.90
Utterance matching score	0.81	
Utterance confidence	0.92	
Utterance overall score	0.83	

Table 4: Case study: [A] and [B] are the signatures of speakers. ch and en are the language abbreviations.

A.5 Case Study

We show a representative case of selecting pseudo-labels in Table 4. There are two predicates in current utterance: 是(is) and 受(attract). For 是(is), the CSRL parser yields only ARG1 while SSRL parser gives the same ARG1 but more of ARG0 based on the rewritten utterance. With the difference in arguments, the overall score is not high and this predicate could be regarded as low-quality if a high pick threshold is set. For 受(attract), the CSRL and SSRL parsers give the same arguments, which are the right answer. However, if we only consider the model confidence of the predicate, which is 0.54, this high-quality predicate are more likely to been discarded than consider the overall score, which is 0.90. And the rewritten utterance gets a high overall score, which is what we expected.

A.6 Discussion on Generalization of the Framework

It is not uncommon at all for different language tasks sharing some information. With one case study presented in detail in the main body of the paper, we also provide a short example of a different friend task pair – constituency parsing and dependency parsing – and explain how they can help each other and show the general nature of the friend-training framework.

Early work (Magerman, 1995; Collins, 2003) has shown relationship between dependency and constituency parsing through head-finding rules, and Jin and Schuler (2019) show directly how common structures between dependency and constituency trees can be derived for parsing evaluation. In a dependency graph, a set of nodes with a single incoming edge is usually indicative of a phrase structure, such as a noun phrase, a verb phrase or a prepositional phrase. Such phrasal structures are well-marked in constituency treebanks, and could be used as the shared friend information for friend-training. Here is a sketch of how friend-training can be applied to this pair:

1. Train a constituency parser and a dependency parser, presumably trained with a small number of training instances, as the models for the friend-training framework.
2. Run both parsers on a common set of unlabeled data for parsing results.
3. Find phrases such as noun, verb or prepositional phrases in the predicted constituency trees.
4. Compare with the dependency trees, and check if spans of such phrases have only a single incoming edge. If so, the constituency and dependency parsing results can be considered agreeing, and added to the silver training set. If not, the silver annotation is discarded.
5. Train the parsers again with the gold and silver training instances.

As long as some shared information can be identified between two seemingly different tasks, the noisy agreement between that partial target can provide valuable supervision between two tasks. The translation and matching between constituency-dependency targets are simpler compared to the CSRL-rewriting pair presented in the paper, partly because no model is required for the translation process. However the CSRL-rewriting pair is more significant because heuristics may be difficult or not obvious to design where ‘bridging’ tasks such as single-sentence SRL may be readily available.

Understanding Transformer Memorization Recall Through Idioms

Adi Haviv$^{\tau}$ **Ido Cohen**$^{\tau}$ **Jacob Gidron**$^{\tau}$ **Roei Schuster**$^{\mu}$
Yoav Goldberg$^{\alpha\beta}$ **Mor Geva**$^{\alpha *}$

$^{\tau}$Tel Aviv University $^{\mu}$Wild Moose $^{\beta}$Bar-Ilan University $^{\alpha}$Allen Institute for AI
`adi.haviv@cs.tau.ac.il`, `roei@wildmoose.ai`, `pipek@google.com`,
`{its.ido, jacob.u.gidron, yoav.goldberg}@gmail.com`

Abstract

To produce accurate predictions, language models (LMs) must balance between generalization and memorization. Yet, little is known about the mechanism by which transformer LMs employ their memorization capacity. When does a model decide to output a memorized phrase, and how is this phrase then retrieved from memory? In this work, we offer the first methodological framework for probing and characterizing recall of memorized sequences in transformer LMs. First, we lay out criteria for detecting model inputs that trigger memory recall, and propose *idioms* as inputs that typically fulfill these criteria. Next, we construct a dataset of English idioms and use it to compare model behavior on memorized vs. non-memorized inputs. Specifically, we analyze the internal prediction construction process by interpreting the model's hidden representations as a gradual refinement of the output probability distribution. We find that across different model sizes and architectures, memorized predictions are a two-step process: early layers promote the predicted token to the top of the output distribution, and upper layers increase model confidence. This suggests that memorized information is stored and retrieved in the early layers of the network. Last, we demonstrate the utility of our methodology beyond idioms in memorized factual statements. Overall, our work makes a first step towards understanding memory recall, and provides a methodological basis for future studies of transformer memorization.[1]

1 Introduction

Transformer language models (LMs) memorize instances from their training data (Carlini et al., 2021; Zhang et al., 2021b), and evidence is building that such memorization is an important precondition for their predictive abilities (Lee et al., 2022; Feldman, 2020; Feldman and Zhang, 2020; Raunak et al., 2021; Raunak and Menezes, 2022). Still, it is unknown when models decide to output memorized sequences, and how these sequences are being retrieved internally from memory. Current methods for analyzing memorization (Feldman and Zhang, 2020; Zhang et al., 2021b; Carlini et al., 2022) use definitions that are based on models performance, which changes between models and often also between training runs. Moreover, these methods study memorization behavior in terms of the model's "black-box" behavior rather than deriving a behavioral profile of memory recall itself.

Our first contributions are to provide a definition and construct a dataset that allows probing memorization recall in LMs. We define a set of criteria for identifying memorized sequences that does not depend on model behavior:[2] sequences that have a single plausible completion that is independent of context and can be inferred only given the entire sequence. We show that many idioms (e.g., "play it by ear") fulfill these conditions, allowing us to probe and analyze memorization behavior. Furthermore, we construct a dataset of such English idioms, dubbed IDIOMEM, and release it publicly for the research community.

Next, to analyze memory recall behavior, we compare the construction process of predictions that involve memory recall with those that do not. To this end, given a LM, we create two sets of memorized and non-memorized idioms from IDIOMEM (Fig. 1, A). We then adopt a view of the transformer inference pass as a gradual refinement of the output probability distribution (Geva et al., 2021; Elhage et al., 2021). Concretely, the token representation at any layer is interpreted as a "hidden" probabil-

---

[*] Now at Google Research.

[1] Our code and data are available at `https://github.com/adihaviv/idiomem/`.

[2] Literature often purports to "define memorization", resulting in a multitude of technical definitions with subtle differences, although we would expect this concept to be consistent and intuitive. Thus, instead of explicitly defining "memorization", we will define *sufficient criteria* for detecting memorized instances.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 248–264
May 2-6, 2023 ©2023 Association for Computational Linguistics

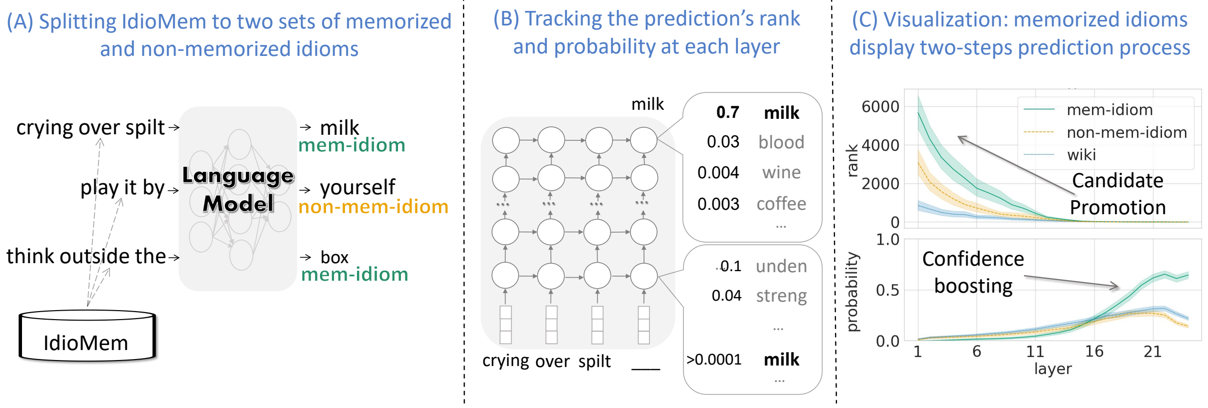


Figure 1: Our methodological framework for probing and analyzing memorized predictions of a given LM: (A) we create two sets of memorized (`mem-idiom`) and non-memorized (`non-mem-idiom`) idioms by probing the LM with instances from IDIOMEM, (B) for each instance, we extract hidden features of the prediction computation – the rank and probability of the predicted token across layers, and (C) we compare the prediction process of memorized idioms versus non-memorized idioms and short sequences from Wikipedia (`wiki`). Memorized predictions exhibit two characteristic phases: candidate promotion and confidence-boosting.

ity distribution over the output vocabulary (Geva et al., 2022) (Fig. 1, B). This interpretation allows tracking the prediction across layers in the evolving distribution. We find a clear difference in model behavior between memorized and non-memorized predictions (Fig. 1, C). This difference persists across different transformer architectures and sizes: retrieval from memory happens in two distinct phases, corresponding to distinct roles of the transformer parameters and layers: (1) *candidate promotion* of memorized predictions' rank in the hidden distribution in the first layers, and (2) *confidence boosting* where, in the last few layers, the prediction's probability grows substantially faster than before. This is unlike non-memorized predictions, where the two phases are less pronounced and often indistinct. We further confirm these phases of memorized predictions through intervention in the network's FFN sublayers, which have been shown to play an important role in the prediction construction process (Geva et al., 2022; Mickus et al., 2022). Concretely, zeroing-out hidden FFN neurons in early layers deteriorate memory-recall, while intervention in upper layers does not affect it.

Last, we show our findings extend to types of memory recall beyond idioms by applying our method to factual statements from the LAMA-UHN dataset (Poerner et al., 2020) (e.g. *"The native language of Jean Marais is French"*). For factual statements that were completed correctly by the LM, we observe the same two phases as in memorized idioms, further indicating their connection to memory recall.

To summarize, we construct a novel dataset of idioms, usable for probing LM memorization irrespective of the model architecture or training parameterization. We then design a probing methodology that extracts carefully-devised features of the internal inference procedure in transformers. By applying our methodology and using our new dataset, we discover a profile that characterizes memory recall, across transformer LMs and types of memorized instances. Our released dataset, probing framework, and findings open the door for future work on transformer memorization, to ultimately demystify the internals of neural memory in LMs.

2 Criteria for Detecting Memory Recall

To study memory recall, we require a set of inputs that trigger this process. Prior work on memorization focused on detecting instances whose inclusion in the training data has a specific influence on model behavior, such as increased accuracy on those instances (Feldman and Zhang, 2020; Magar and Schwartz, 2022; Carlini et al., 2022, 2021, 2019). As a result, memorized instances differ across models and training parameterization. Our goal is instead to find a stable dataset of sequences that correctly predicting their completion indicates memorization recall. This will greatly reduce the overhead of studying memorization and facilitate useful comparisons across models and studies.

To build such a dataset, we start by defining a general set of criteria that are predicates on sequence features, entirely independent of the LM being probed. Given a textual sequence of n words, we call the first $n-1$ words the *prompt* and the nth word the *target*. We focus on the task of predicting the target given the prompt, i.e., predicting the last word in a sequence given its prefix.[3] Such predictions can be based on either generalization or memorization, and we are interested in isolating memorized cases to study model behavior on them. Particularly, we are looking for *sequences for which success in this task implies memorization recall*.

We argue that the following criteria are *sufficient* for detecting such memorized sequences:

1. **Single target, independent of context:** We require that the target is the only correct continuation, regardless of the textual context where the prompt is placed.[4]
2. **Irreducible prompt:** The target is the single correct completion only if the entire prompt is given exactly. Changing or removing parts from the prompt would make the correct target non-unique.

Claim 2.1. *Assume a sequence fulfills the above criteria. Then, if a LM correctly predicts the target, it is highly likely that this prediction involves memory recall.*

Justification. First, observe that most natural-language prompts have many possible continuations. For example consider the sentence *"to get there fast, you can take this ____"*. Likely continuations include *"route"*, *"highway"*, *"road"*, *"train"*, *"plane"*, *"advice"*, inter alia. Note that there are several divergent interpretations or contexts for the prompt, and for each, language offers many different ways to express similar meaning.

A prediction that is a product of generalization — i.e., it is derived from context and knowledge of language — *always has plausible alternatives*, depending on the context and stylistic choice of words. Hence, the relationship between the entire prompt and the target, where the target is the single correct continuation, is something that needs to be memorized rather than derived via generalization. A LM that predicts *the* single correct continuation either memorized this relationship, or used "cues" from the prompt that happen to provide indication towards the correct continuation. To illustrate the latter, consider the sequence *"it's raining cats and ____"* which has a single correct continuation, *"dogs"*, but a LM might predict it without observing this sequence during training, due to the semantic proximity of *"cats"* and *"dogs"*. Our second criterion excludes such cases by requiring that the correct continuation is only likely given the entire sequence.

Therefore, a LM that correctly completes a sequence that fulfills both criteria, is likely to have recalled it from memory. ☐

In the next section, we argue that idioms are a special case of such sequences, and are thus useful for studying memorization (§3).

3 The Utility of Idioms for Studying Memorization

An idiom is a group of words with a meaning that is not deducible from the meanings of its individual words. For example, consider the phrase *"play it by ear"* — there is a disconnect between its nonsensical literal meaning (to play something by a human-body organ called 'ear') and its intended idiomatic meaning (to improvise).

A key observation is that *idioms often satisfy our criteria (§2), and therefore can probe memorization*. First, by definition, idioms are expected to be non-compositional (Dankers et al., 2022). They are special "hard-coded" phrases that carry a specific meaning. As a result, their prompts each have a single correct continuation, regardless of their context (criterion 1). For example, consider the prompt *"crying over spilt ____"* — a generalizing prediction would predict that this slot may be filled by any spillable item, like wine, water or juice, while a memorized prediction will retrieve only milk in this context. Notably, while this is an empirical characterization of many idioms, there might be exceptions, e.g., contexts that are adversarially chosen to change the completion. Second, many idioms are "irreducible", for example the sub-sequences *"crying over"* or *"over spilt"* by themselves have but scant connection to the word *"milk"*.

Still, not all idioms fulfill the criteria. For example, even when the idiom is far from literal, its constituents sometimes strongly indicate the correct continuation, such as with the case of *"it's raining cats and ____"* (as explained in §2). To construct a dataset of memorization-probing sequences, we will carefully curate a set of English idioms and filter out ones that do not fulfill our criteria.

---

[3] In cases where tokenization divides the target to subtokens, our task becomes predicting the target's first token.

[4] We assume that contexts are naturally-occurring and not adversarial.

Source	**# of Idioms**	**Idiom Length** (words)
MAGPIE	590	4.5 ± 0.9
LIdioms	149	5.1 ± 1.2
EF	97	5.6 ± 1.9
EPIE	76	4.4 ± 0.7
Total (unique)	814	4.7 ± 1.8

Table 1: Statistics per data source in IdioMem.

Prompt	**Target**	**Pred.**	**Sim.**	**IdioMem**
"make a mountain out of a"	`molehill`			✓
"think outside the"	`box`			✓
"there's no such thing as a free"	`lunch`			✓
"go back to the drawing"	`board`	✓		
"boys will be"	`boys`		✓	
"take it or leave"	`it`	✓	✓	

Table 2: Example English idioms included and excluded from IdioMem by our filters of predictable target (Pred.) and prompt-target similarity (Sim.).

3.1 The IdioMem Dataset

We begin with existing datasets of English idioms: MAGPIE (Haagsma et al., 2020),[5] EPIE (Saxena and Paul, 2020), and the English subset of LIdioms (Moussallem et al., 2018). We enrich this collection with idioms scraped from the website "Education First" (EF).[6] We then split each idiom into a prompt containing all but the last word, and a target that is the last word. Next, we filter out idioms that do not comply with our criteria (§2) or whose target can be predicted from their prompt based on spurious correlations rather than memorization. To this end, we use three simple rules:

- **Short idioms**. We observe that prompts of idioms with just a few words often have multiple plausible continuations, that are not necessarily the idiom's target, violating our first criterion. For example, the prompt *"break a ____"* has many possible continuations (e.g. *"window"*, *"promise"*, and *"heart"*) in addition to its idiomatic continuation *"leg"*. To exclude such cases, we filter out idioms with < 4 words.

- **Idioms whose target is commonly predicted from the prompt's subsequence.** We filter such cases to ensure the prompt fulfills our second criterion (prompt irreducibility).

  To detect these cases, we use an ensemble of pretrained LMs: GPT2_{M}, RoBERTa-base (Liu et al., 2019), T5-base (Kale and Rastogi, 2020) and ELECTRA-base-generator (Clark et al., 2020), and check for each model if there is an n-gram ($1 \leq n \leq 4$) in the prompt from which the model predicts the target. We filter out idioms for which a majority (≥ 3) of models predicted the target (for some n-gram).

- **Idioms whose targets are semantically similar to tokens in the prompt.** To further ensure prompt irreducibility, we embed the prompt's tokens and the target token using GloVe word embeddings (Pennington et al., 2014). We measure the cosine distance between the target token to each token in the prompt separately and take the maximum of all the tokens. We filter out idioms where this number is higher than 0.75 (this number was tuned manually using a small validation set of idioms).

Overall, 55.7% of the idioms were filtered out, including 48.5% by length, 6.1% by the predictable-target test and an additional 1.6% by the prompt-target similarity, resulting in an 814 idioms dataset, named IdioMem. Further statistics are provided in Tab. 1, and example idioms in Tab. 2.

4 Probing Methodology

Background and Notation Assuming a transformer LM with L layers, a hidden dimension d and an input/output-embedding matrix $E \in \mathbb{R}^{|V| \times d}$ over a vocabulary V. Denote by $\mathbf{s} = \langle s_1, ..., s_t \rangle$ the input sequence to the LM, and let $\mathbf{h}_i^\ell$ be the output for token i at layer ℓ, for all $\ell \in 1, ..., L$ and $i \in 1, ..., t$. The model's prediction for a token s_i is obtained by projecting its last hidden representation $\mathbf{h}_i^L$ to the embedding matrix, i.e. $softmax(E\mathbf{h}_i^L)$.

Following (Geva et al., 2021, 2022), we interpret the prediction for a token s_i by viewing its corresponding sequence of hidden representations $\mathbf{h}_i^1, ..., \mathbf{h}_i^L$ as an evolving distribution over the vocabulary. Concretely, we read the "hidden" distribution at layer ℓ by applying the same projection to the hidden representation at that layer: $\mathbf{p}_i^\ell = softmax(E\mathbf{h}_i^\ell)$. Using this interpretation, we track the probability and rank of the predicted token in the output distribution across layers. A token's rank is its position in the output distribution when sorted by probability from highest to lowest (e.g. the rank of the final predicted token is zero).

---

[5] We take idioms with an annotation confidence of $> 75\%$ and exclude frequently occurring literal interpretations.

[6] `https://www.ef.com/wwen/english-resources/english-idioms/`

	GPT2_M		GPT2_L		BERT_B		BERT_L	
Memorized idioms (`mem-idiom`)	364	44.7%	392	48.2%	230	28.3%	305	37.5%
Non-memorized idioms (`non-mem-idiom`)	450	55.3%	422	51.8%	584	71.7%	509	62.5%

Table 3: Number of memorized idioms vs. non-memorized idioms from the IDIOMEM dataset for each model. An instance is considered a memorized example if the model correctly predicts the target.

Probing Procedure Our key method to understand how transformer LMs retrieve information from memory is *comparing features of memory recall* to inference that does not necessarily include memory recall. Given a set of sequences that fulfill the criteria in §2, we split them into a "memorized" set whose targets' first token is predicted correctly by the model being analyzed given (and are therefore memorized), and a "non-memorized" set whose target is predicted incorrectly. We additionally include a second set of "non-memorized" instances: natural-language sequences randomly sampled from a large corpus (we assume most naturally-occurring sequences are not memorized).

To probe a LM, we run it on the 3 sets, and for each set and each layer, we (a) extract the rank and probability of the final predicted token in the hidden distribution for each prompt, and (b) compute the mean rank and probability over all prompts.

5 Probing Memorization Using Idioms

5.1 Experimental Setup

Datasets For each LM under analysis (see below), we split IDIOMEM into two disjoint subsets of memorized and non-memorized idioms, denoted as `mem-idiom` and `non-mem-idiom`, respectively, according to whether or not the LM succeeds in completing them. We produce an additional set of non-memorized instances, `wiki`, by sampling prompts from the WIKITEXT-103 dataset (Merity et al., 2017) of the same length distribution as in IDIOMEM (see Tab. 1).

Models We use multiple transformer LMs that are different in size, architecture, and optimization objective. We use GPT2 (medium and large) (Radford et al., 2019), an autoregressive transformer decoder, and BERT (base and large) (Devlin et al., 2019), a transformer encoder trained with a masked language modeling (MLM) objective. To evaluate BERT on a specific idiom, we feed the idiom's prompt concatenated with the special mask token and a period (e.g. *"think outside the* `[MASK]`."). Further details on each model are presented in Tab. 6. The number of memorized and non-memorized idioms from IDIOMEM for each model are provided in Tab. 3.

5.2 Memorized Predictions are a Two-Step Process

Fig. 2 shows the probability and rank of the output token across layers, for memorized and non-memorized idioms and short sequences from Wikipedia, by GPT2_M, GPT2_L, BERT_B, and BERT_L. Naturally, the prediction's rank decreases across layers as the prediction probability increases. However, for memorized predictions these trends occur as two distinct and sharp inference phases. In lower layers, the prediction's rank decreases from a high rank to near zero, while its probability is also close to zero. For example, in GPT2_L the rank decreases until layer 20 while the probability remains below 0.1. We refer to this phase as *candidate promotion*, as the predicted token is being promoted to be a top candidate in the output distribution.

Compared to non-memorized predictions, the initial rank of memorized predictions is generally higher, especially in GPT2 (6000 vs. 3000 in GPT2_M, and 3000 vs. 1500 in GPT2_L). A potential explanation would be a generally lower frequency of the predicted token for memorized idioms. However, we find there is only low negative correlation between the initial rank of the predicted token and its frequency (see Tab. 4). We therefore offer a different explanation: non-memorized predictions are often promoted in early layers that detect local "shallow" patterns, such as common bigrams (Geva et al., 2021), while predictions for memorized idioms are not local as they requires processing of the entire input.

In the middle layers, once the predicted token reaches the top of the hidden distribution, its probability increases until the last layer. We refer to this phase as *confidence boosting*, as the distribution shifts towards the predicted token. For memorized idioms, this increase is abrupt and dramatic, with a final probability of > 0.6 across all models. In comparison, predictions on short sequences from Wikipedia and non-memorized idioms have a substantially lower probability of ~ 0.2. This can be

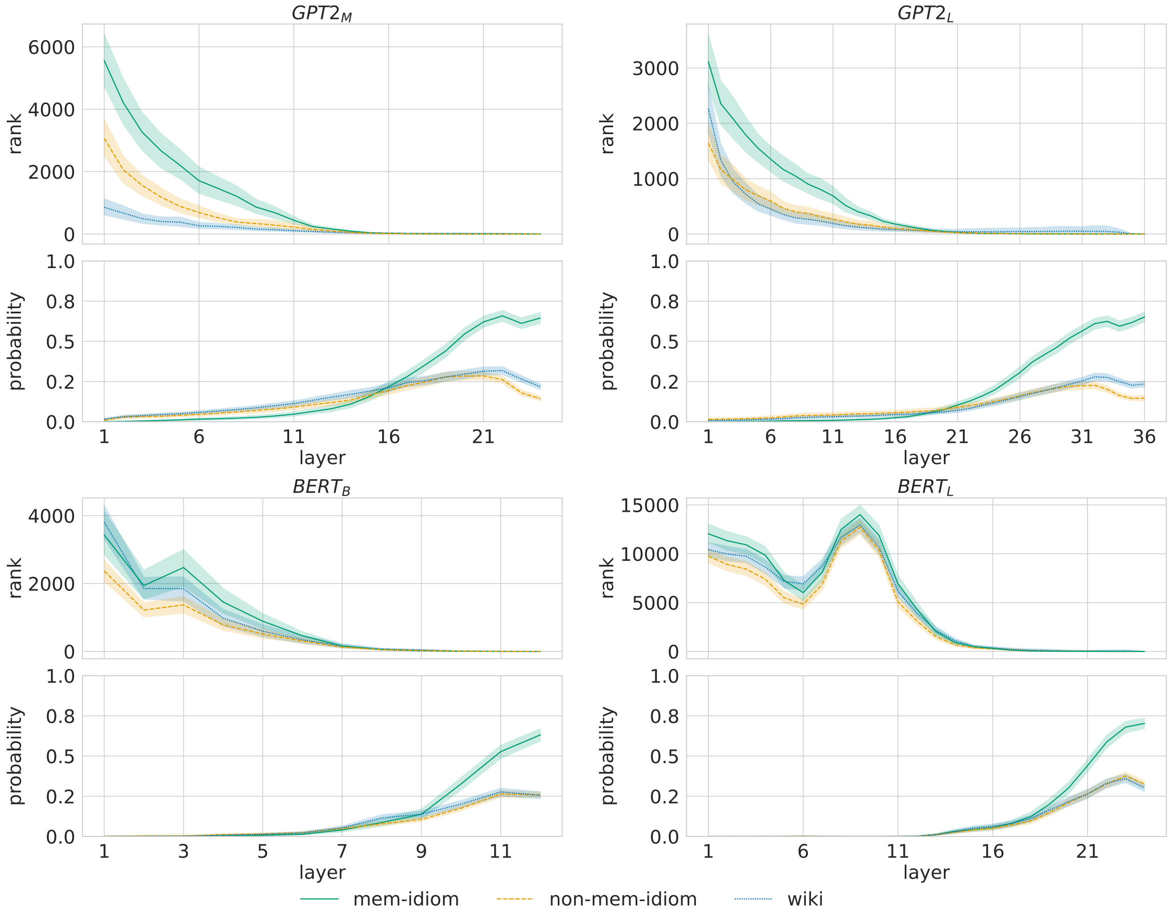


Figure 2: The predicted token's probability and rank across layers, for memorized idioms (`mem-idiom`), non-memorized idioms (`non-mem-idiom`) and short sequences from Wikipedia (`wiki`). Memory recall exhibits two characteristic phases of candidate promotion and confidence boosting.

explained by the fact that memorized idioms have a single correct target, rather than many possible continuations, as in the instances from Wikipedia. In addition, low-probability predictions for non-memorized idioms are expected as the model did not memorize the idioms and does not know their continuation. In §B, we provide more fine-grained analysis of these trends via a log-scaled view of the prediction's rank and a visualization of the ranks and probabilities for separate clusters of the memorized predictions.

We further verify that our extracted hidden-distribution features distinguish between memorized and non-memorized predictions by training linear classifiers over combinations of these features (details in §A). We observe that, indeed, our features enable separation between memorized and non-memorized predictions at high accuracy (77%-85% across models). Moreover, classifiers that use hidden distribution features are more accurate than those relying only on the model's output. Overall, these findings provide a profile of memorized predictions, suggesting that the memorized information is retrieved in early layers at inference.

	correlation	p-value
GPT2_{M}	-0.22	$2.9e^{-21}$
GPT2_{L}	-0.18	$5.4e^{-14}$
BERT_{B}	-0.19	$1.8e^{-15}$
BERT_{L}	0.15	$3.5e^{-10}$

Table 4: Pearson correlation between the predicted token's rank at the first layer and its general frequency in Wikipedia.

5.3 Testing the Roles of Different Layers Through Intervention

Our analysis in the previous section interprets hidden representations as distributions over the output vocabulary. We now conduct intervention experiments to verify that this interpretation is meaningful for studying memory recall, and to test layers' roles in that process. Concretely, we zero out FFN

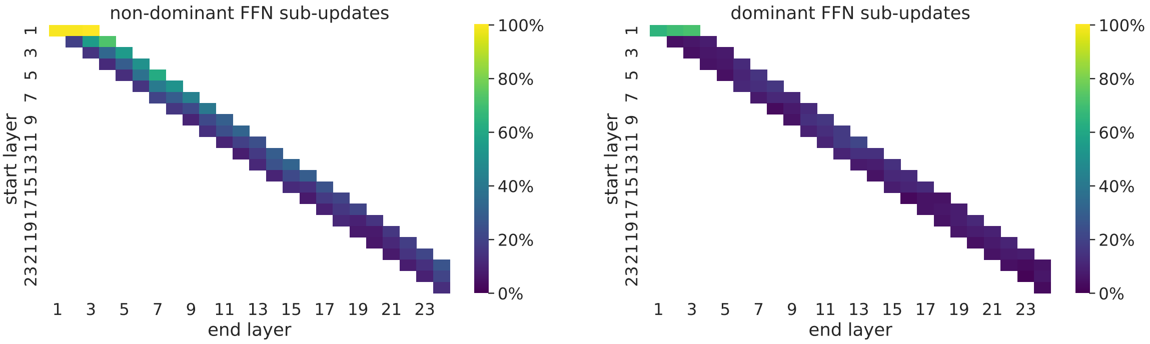


Figure 3: Intervention in non-dominant (left) and dominant (right) FFN sub-updates in GPT2$_{\mathrm{M}}$. Each cell shows the percentage of memorized idioms for which the prediction was changed by zeroing-out the FFN sub-updates between the start and end layers.

sub-updates to the hidden representation (§4) and measure changes in memorized predictions.

A Short Primer on Transformer FFN Sublayers. FFN sublayers are the final computation in transformer layers, which output the hidden distribution at the center of our analysis. In general, they have a key role in capturing knowledge in transformer LMs (Dai et al., 2022; Meng et al., 2022). We follow Geva et al. (2022) and view the computation of each FFN sublayer as a weighted collection of d_m sub-updates to the output distribution, each promoting a concept in the vocabulary space, e.g. "past-tense verbs" or "female athletes". To understand this, consider the computation of the FFN at layer ℓ, given by $\mathsf{FFN}^{\ell}(\mathbf{h}_i^{\ell}) = f(W_K^{\ell}\mathbf{h}_i^{\ell})W_V^{\ell}$, where $W_K^{\ell}, W_V^{\ell} \in \mathbb{R}^{d_m \times d}$ are parameter matrices, d_m is the intermediate hidden dimension, and f is a point-wise non-linearity activation function. This computation can be decomposed as: $\mathsf{FFN}^{\ell}(\mathbf{h}_i^{\ell}) = \sum_{j=1}^{d_m} f(\mathbf{h}_i^{\ell} \cdot \mathbf{k}_j^{\ell})\mathbf{v}_j^{\ell} = \sum_{j=1}^{d_m} m_j^{\ell}\mathbf{v}_j^{\ell}$, where $\mathbf{k}_j^{\ell}$ and $\mathbf{v}_j^{\ell}$ are the j-th row in W_K^{ℓ} and the j-th column in W_V^{ℓ}, respectively. Geva et al. argue that each weight m_j^{ℓ} is the score assigned by the model for some textual pattern, and each vector $\mathbf{v}_j$ promotes a concept that follows that pattern.

Experiment First, we sample 100 random instances from IdioMem that the model memorized. Then, for each range of up to 3 consecutive layers, we perform two complementary experiments, where we run GPT2$_{\mathrm{M}}$'s inference on the 100 instances while intervening in the chosen layers to cancel the contribution of FFN sub-updates to the prediction. Specifically, for each layer ℓ in the layer range, we perform the following two interventions: first, we zero out (i.e. artificially set to 0 during inference) the weights of the 10 most dominant sub-updates, which are known to be particularly salient for predictions (Geva et al., 2022) (there are d_m sub-updates per layer). Concretely, we sort the sub-updates by their contribution to the FFN output $|m_i^{\ell}|||\mathbf{v}_i^{\ell}||\ \forall i \in [1, ..., d_m]$, and set $m_i^{\ell} = 0$ for the 10 sub-updates with the highest contribution. Next, we zero out non-dominant sub-updates, i.e. all the sub-updates *except* for the 10 most dominant ones. For each intervention, we measure how often it changes the predicted token. Further measurements of the change in rank and probability of the target token are reported in §C.

Results Fig. 3 shows, for each layer range, the percentage of memorized idioms where the predicted token has changed. Focusing on zeroing out non-dominant FFN sub-updates (Fig. 3, left), we observe a *two-phase pattern of decreasing "layer importance" which corresponds to the two-phase pattern of decreasing rank and increasing probability during inference (§5.2)*: layers' effect on memory recall drops precipitously in the first 10 layers, corresponding to the candidate promotion phase. Then, from around layer 10 onwards, the drop in effect is much less steep, corresponding to the confidence-boosting phase. Intervention in upper layers rarely changes the predicted token, and its effect is limited to reducing the model's confidence (§C). We further visualize this two-phase behavior in Fig. 7 in §C.

These findings suggest that the candidate promotion phase, while having a smaller effect on the prediction's assigned probability compared to the later confidence-boosting stage, in fact, has a crucial role in memory recall.

We also observe that interventions in the first layer are by far the most destructive, with 100% change in prediction for non-dominant updates

(Fig. 3, left). Unlike for other layers, this is also observable when zeroing out the dominant sub-updates (Fig. 3, right), which constitute only 0.1% of layer sub-updates. This suggests the first layer is especially critical for memorized predictions.

6 Memorization of Factual Statements

We now examine if our findings generalize beyond idioms to other types of memory recall, focusing on the completion of statements expressing facts.

Data Datasets for evaluating memorization of factual knowledge typically contain simple queries such as *"The continent of Kuwait is"*, where predicting the next token requires knowledge of the triplet $\langle s, r, t \rangle$ where s is a source entity (e.g., `Kuwait`), t a target entity (e.g., `Asia`), and r is the relation between them (e.g., `is the continent of`). However, unlike idioms, such queries are not suitable for probing memorization since they often fail to satisfy the criteria in §2. Concretely, queries often include "clues" that could make the prediction easy to guess and based on generalization (Poerner et al., 2020) (e.g. predicting a Spanish-speaking country for the query *"Federico López was born in"*), and the same fact can be expressed in multiple different ways (e.g. *"Kuwait is a country in Western Asia"* also encodes the above fact).

To test memorization of facts, we, therefore, collect factual statements where such cases are excluded. We use LAMA-UHN (Poerner et al., 2020), a subset of the LAMA dataset (Petroni et al., 2019), where "easy-to-guess" queries are filtered out. LAMA comprises of queries structured as "fill-in-the-blank" cloze statements (e.g. *"Gordon Scholes is a member of the ____political party."*). To accommodate autoregressive LMs, we consider only queries where the blank appears at the end. In addition, we keep only queries with a single correct completion (based on LAMA). This turns our definition of memorized and non-memorized sets (§4) equivalent to separating based on the evaluation metric of LAMA: an instance is considered memorized if the model predicts the single correct completion and non-memorized otherwise. Overall, the resulting collection consists of 17,855 factual statements with 22 unique relations.

Memorized Facts Exhibit a Similar Prediction Process to Idioms We repeat our analysis (§5.1), using the collected factual statements and GPT2_{M}. Splitting the statements to those memorized and non-memorized by the model results in 786 `mem-facts` vs. 17,069 `non-mem-facts` statements.

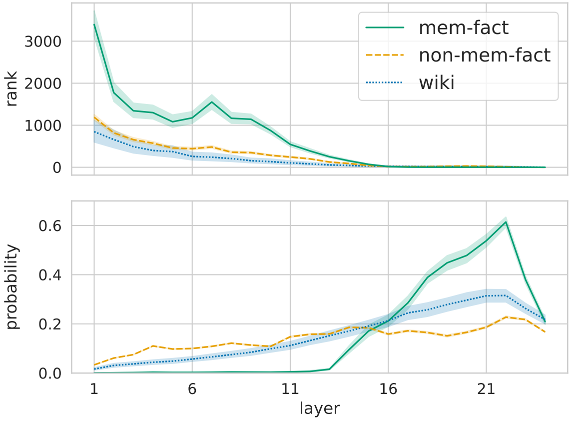


Figure 4: The predicted token's probability and rank across layers of GPT2_{M}, for memorized (`mem-fact`) and non-memorized (`non-mem-fact`) facts and short sequences from Wikipedia (`wiki`).

Fig. 4 shows the rank and probability of the predicted token across layers. Like idioms, memorized facts exhibit a clear two-phase prediction process, where the prediction probability rapidly increases once the candidate reaches a low rank (at layer 16). This is in contrast to non-memorized facts and short sequences from Wikipedia, where the rank (probability) gradually decreases (increases) across layers without a distinct two phases.

Differences from Memorized Idioms Stem from Ill-Defined Targets There is one major difference compared to memorized idioms (Fig. 4 vs. Fig. 2 upper left), which is a substantial drop in probability (0.62 $\rightarrow$ 0.21) in the last two layers. We hypothesize that this is because the input query has multiple plausible completions that were not specified as "correct" targets in LAMA. We verify this by manually analyzing predictions, and find that for 82 out of 100 queries there is more than one correct continuation in top 5 predicted tokens. We posit that the above deficiency of LAMA is inherent because, in violation of our criterion (§2), factual statements can usually be expressed in many ways so their prompt has no single correct target.

7 Related Work

Memorization as Training-Data Influence. Memorization in LMs has attracted immense attention due to their rich sensitive training data (Carlini et al., 2019; Song and Shmatikov, 2019; Carlini et al., 2021; Zhang et al., 2021b; Carlini et al., 2022; Tirumala et al., 2022; Tänzer et al., 2022; Raunak and Menezes, 2022). Recent work suggests that memorization is *necessary* for performant ML due to the "long tail" of infrequently-observed patterns (Zhang et al., 2021a; Feldman, 2020; Brown et al., 2021; Raunak et al., 2021). This line of work has two key limitations: (a) only black-box behavior is measured rather than looking at models' internal prediction process, and (b) it detects memorized instances by measuring the effect of their inclusion in the training set on inference behavior. This results in a set of memorized examples that is specific to the model, training data, and even training pass, making it difficult to build on these results in future research.

Transformers and Idioms. Nedumpozhimana and Kelleher (2021) showed that idioms are identified using textual cues within the expression; Dankers et al. (2022) showed that idioms tend to be internally processed as single units of meaning. It has also been known (Fakharian and Cook, 2021; Salton et al., 2016) that LM contextual embeddings encode information about whether or not an expression is idiomatic (vs. literal). (Shwartz and Dagan, 2019; Chakrabarty et al., 2022) also studied representations and interpretation of non-compositional sequences, such as idioms. No prior work used idioms to probe LM memorization, which is one of the main contributions of this work. We release our dataset, IDIOMEM, to facilitate future research on memorization recall in LMs. Diagnostic datasets, such as IDIOMEM, have often proven useful in the past (Sugawara et al., 2022; Parrish et al., 2021).

Memorization of Factual Knowledge. An extensive line of work (Petroni et al., 2019; Jiang et al., 2020; Poerner et al., 2019; Lewis et al., 2020; Elazar et al., 2021) studied LMs' capacity to acquire relational knowledge during training. Some attention has also been given to understanding the inner workings of factual-memory recall: Wallat et al. (2020) showed that some facts are retrieved from the bottom and intermediate layers, and Meng et al. (2022) localized factual-knowledge recall within feed-forward-component computation. Since factual statements do not fulfill our criteria, it is difficult to convincingly argue that correct predictions indicate memory recall, making it impossible to use them to isolate the effect of memorization.

8 Conclusion

We introduce a methodological framework for detecting and analyzing memorized predictions in transformer LMs. This includes a set of criteria on textual sequences for probing memorized predictions, the IDIOMEM dataset of idioms fulfilling these criteria, and an interpretation method of prediction internals. We characterize a behavioral profile that is unique to predictions involving memory recall and is observable across different LMs and data types. By providing these fundamental tools and initiating a thread of research on the phenomena we observe, we hope to empower future work towards demystifying transformer memorization.

Limitations

Our criteria for detecting memorized instances are sufficient but not necessary, which raises the question of what other sequences that trigger memory recall satisfy them.

Additionally, while correct prediction for sequences that fulfill our criteria implies memory recall, incorrect prediction does not necessarily imply that no memory was used. This means that our set `non-mem-idiom` might include some portion of memorized sequences. This does not qualitatively affect our results as long as `mem-idiom` still contains *more* memorized predictions than `non-mem-idiom`, as is evidenced by the stark differences we observe between LMs' internal behaviors on these sets.

Our work focuses on showing the utility of idioms for probing memorization, and opening up a new thread of research in this vein. We therefore leave further investigation of the above gaps for future work.

Acknowledgements

We thank Gabriel Stanovsky for helpful feedback. This project has received funding and resources from CIFAR through the Canada CIFAR AI Chair program, the Province of Ontario, companies sponsoring the Vector Institute, as well as the Computer Science Scholarship granted by the Séphora Berrebi Foundation, and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT).

References

Gavin Brown, Mark Bun, Vitaly Feldman, Adam Smith, and Kunal Talwar. 2021. When is memorization of irrelevant training data necessary for high-accuracy learning? In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 123–132.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, Santa Clara, CA. USENIX Association.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association.

Tuhin Chakrabarty, Yejin Choi, and Vered Shwartz. 2022. It's not rocket science: Interpreting figurative language in narratives. *Transactions of the Association for Computational Linguistics*, 10:589–606.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Verna Dankers, Christopher Lucas, and Ivan Titov. 2022. Can transformer be too compositional? analysing idiom processing in neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3608–3626, Dublin, Ireland. Association for Computational Linguistics.

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

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Samin Fakharian and Paul Cook. 2021. Contextualized embeddings encode monolingual and cross-lingual knowledge of idiomaticity. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 23–32.

Vitaly Feldman. 2020. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 954–959, New York, NY, USA. Association for Computing Machinery.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems*, volume 33, pages 2881–2891. Curran Associates, Inc.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hessel Haagsma, Johan Bos, and Malvina Nissim. 2020. MAGPIE: A large corpus of potentially idiomatic expressions. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 279–287, Marseille, France. European Language Resources Association.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of* *the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445, Dublin, Ireland. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Inbal Magar and Roy Schwartz. 2022. Data contamination: From memorization to exploitation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 157–165, Dublin, Ireland. Association for Computational Linguistics.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual knowledge in gpt. *arXiv preprint arXiv:2202.05262*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

Timothee Mickus, Denis Paperno, and Mathieu Constant. 2022. How to dissect a Muppet: The structure of transformer embedding spaces. *Transactions of the Association for Computational Linguistics*, 10:981–996.

Diego Moussallem, Mohamed Ahmed Sherif, Diego Esteves, Marcos Zampieri, and Axel-Cyrille Ngonga Ngomo. 2018. LIdioms: A multilingual linked idioms data set. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Vasudevan Nedumpozhimana and John Kelleher. 2021. Finding bert's idiomatic key. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 57–62.

Alicia Parrish, Sebastian Schuster, Alex Warstadt, Omar Agha, Soo-Hwan Lee, Zhuoye Zhao, Samuel R. Bowman, and Tal Linzen. 2021. NOPE: A corpus of naturally-occurring presuppositions in English. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 349–366, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. Bert is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised qa. *arXiv preprint arXiv:1911.03681*, 3.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 803–818, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Vikas Raunak and Arul Menezes. 2022. Finding memo: Extractive memorization in constrained sequence generation tasks. *ArXiv*, abs/2210.12929.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online. Association for Computational Linguistics.

Giancarlo Salton, Robert Ross, and John Kelleher. 2016. Idiom token classification using sentential distributed semantics. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 194–204.

Prateek Saxena and Soma Paul. 2020. Epie dataset: A corpus for possible idiomatic expressions. In *Text, Speech, and Dialogue*, pages 87–94, Cham. Springer International Publishing.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Congzheng Song and Vitaly Shmatikov. 2019. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 196–206, New York, NY, USA. Association for Computing Machinery.

Saku Sugawara, Nikita Nangia, Alex Warstadt, and Samuel Bowman. 2022. What makes reading comprehension questions difficult? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6951–6971, Dublin, Ireland. Association for Computational Linguistics.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2022. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578, Dublin, Ireland. Association for Computational Linguistics.

Kushal Tirumala, Aram H Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. *arXiv preprint arXiv:2205.10770*.

Jonas Wallat, Jaspreet Singh, and Avishek Anand. 2020. BERTnesia: Investigating the capture and forgetting of knowledge in BERT. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 174–183, Online. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021a. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2021b. Counterfactual memorization in neural language models. *arXiv preprint arXiv:2112.12938*.

A Distinguishing Memorization Using Hidden-Distribution Features

§5.2 shows differences in our extracted hidden-distribution features, namely the rank and probability across layers, between memorized idioms and non-memorized sequences. To verify that these features are distinguishing between memorized and non-memorized predictions, we build a classifier that receives them as input, as follows.

Experiment To answer the above, we represent every instance in IDIOMEM, for every LM we experiment with, as a sequence of probabilities and ranks assigned to the predicted token at each layer. This results in a feature vector for each instance in IDIOMEM for each of our LMs (GPT2$_{\mathrm{M}}$, GPT2$_{\mathrm{L}}$, BERT$_{\mathrm{B}}$, and BERT$_{\mathrm{L}}$). We then append a class label for each LM's instances corresponding to whether it memorized them. Then, for each LM's dataset, we perform 10-fold cross-validation with an 80%-20% train-test split to evaluate the accuracy of a logistic-regression classifier using the LogisticRegresion classifier of scikit-learn,[7] specifying L1 penalty and a bilinear solver as constructor parameters. After each split and before evaluating the classifier, we balance the test set by replacing the larger class with a random subsample the size of the smaller class. To isolate the distinguishing utility of ranks from that of probabilities, we repeat this process while only taking either of them as features at each time. We also repeat this process while separately considering just the last-layer probability as a single feature, the last hidden state vector, and finally, the ranks in layers of layers 1-12 (omitting ranks in layers 13-16 where ranks are usually 0) appended to all layer probabilities.

Results Results are given in Tab. 5. We observe that, across all models, most classifiers perform well over the 50%-accuracy baseline for their class-balanced test sets. The output probability alone is often highly distinguishing with around 78% accuracy, but the vector of 16 hidden-distribution probabilities seems to contain additional distinguishing information, as using it alongside ranks results in higher accuracy, typically around 84%. Using the ranks in addition to probabilities usually *decreases* accuracy, but omitting the ranks in layers 13-16 (which we know are mostly 0 as this is the confidence-boosting phase) attenuates this effect.

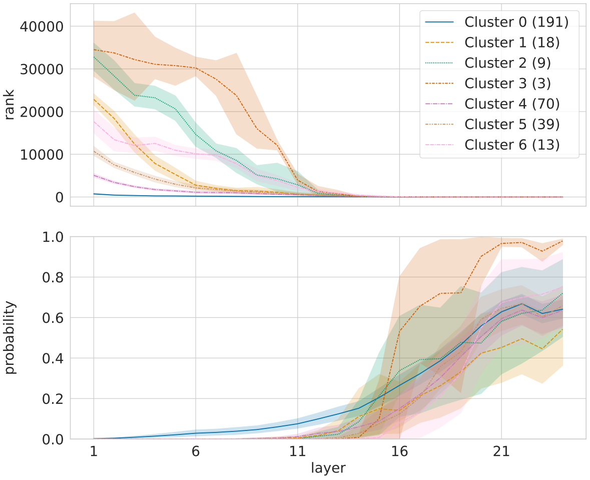


Figure 5: The predicted token's probability and rank across layers for each cluster, after clustering the memorized idioms in GPT2$_{\mathrm{M}}$ according to rank and probability across layers.

We conjecture that ranks have little meaningful information, especially in the confidence-boosting phase.

B Fine-grained Analysis of Memorized Predictions

§5.2 shows how recall of memorized predictions exhibits two characteristic phases (specifically, see Fig. 2). To shed light on this phenomenon, we conduct additional analysis.

B.1 Memorized Idioms In-depth Breakdown

In our analysis, we address all memorized predictions jointly. We now check whether these averaged results are consistent across subgroups of the memorized idioms. To this end, we cluster the memorized idioms by GPT2$_{\mathrm{M}}$, using the same hidden features as in §A, i.e., each instance is represented by the predicted token probability and rank across layers. We then cluster the idioms into seven groups, using K-mean clustering, and visualize the prediction rank and probability across layers for each group. We set the number of clusters to $k = 7$ based on manual inspection, and as we observed no substantial differences in the resulting clusters for larger values of k.

Results are shown in Fig. 5. We find that all groups exhibit the confidence-boosting phase, as the prediction probability quickly increases starting from the intermediate layers. Notably, the lowest final probability observed is > 0.5, which is substantially higher than the average probability of

---

[7] `https://scikit-learn.org/`

	GPT2_M	GPT2_L	BERT_B	BERT_L
probability	$\mathbf{84.6 \pm 2.8}$	$\mathbf{84.0 \pm 2.3}$	$\mathbf{84.2 \pm 1.8}$	$\mathbf{77.6 \pm 3.3}$
ranks	63.2 ± 1.6	59.5 ± 2.7	73.3 ± 3.3	64.5 ± 3.6
probability + ranks	83.6 ± 2.7	82.9 ± 2.9	82.7 ± 2.2	76.9 ± 2.9
ranks layer 1-12 + probability	$\mathbf{84.3 \pm 2.8}$	83.4 ± 2.7	82.7 ± 2.2	$\mathbf{77.2 \pm 4.2}$
probability last layer	79.5 ± 3.5	83.3 ± 2.3	77.7 ± 2.3	74.2 ± 4.8
final hidden state	72 ± 2.1	72.5 ± 3.7	72.7 ± 2.9	66.0 ± 3.0
token ids	58 ± 0.2	59.4 ± 0.2	58.8 ± 0.2	71.6 ± 0.2
random	49.5 ± 3.8	49.7 ± 4.7	50.4 ± 4.9	50.1 ± 3.5

Table 5: Cross-validation accuracy of a logistic-regression classifier trained to distinguish between memorized and non-memorized idioms.

~ 0.2 for non-memorized predictions (§5.2). However, considering the prediction rank for the different groups, we observe a relatively large variation. Specifically, we observe that 55% of the instances (cluster 0) have a low initial rank. This further supports the findings in §A.

B.2 Log-Scale Visualization

Fig. 6 shows a log-scaled view of the graphs from Fig. 2. We observe that, in terms of orders of magnitude (vs. absolute value), the differences in initial ranks between memorized and non-memorized predictions are more minor, especially in BERT models, whereas rank differences measured in upper layers are more stark.

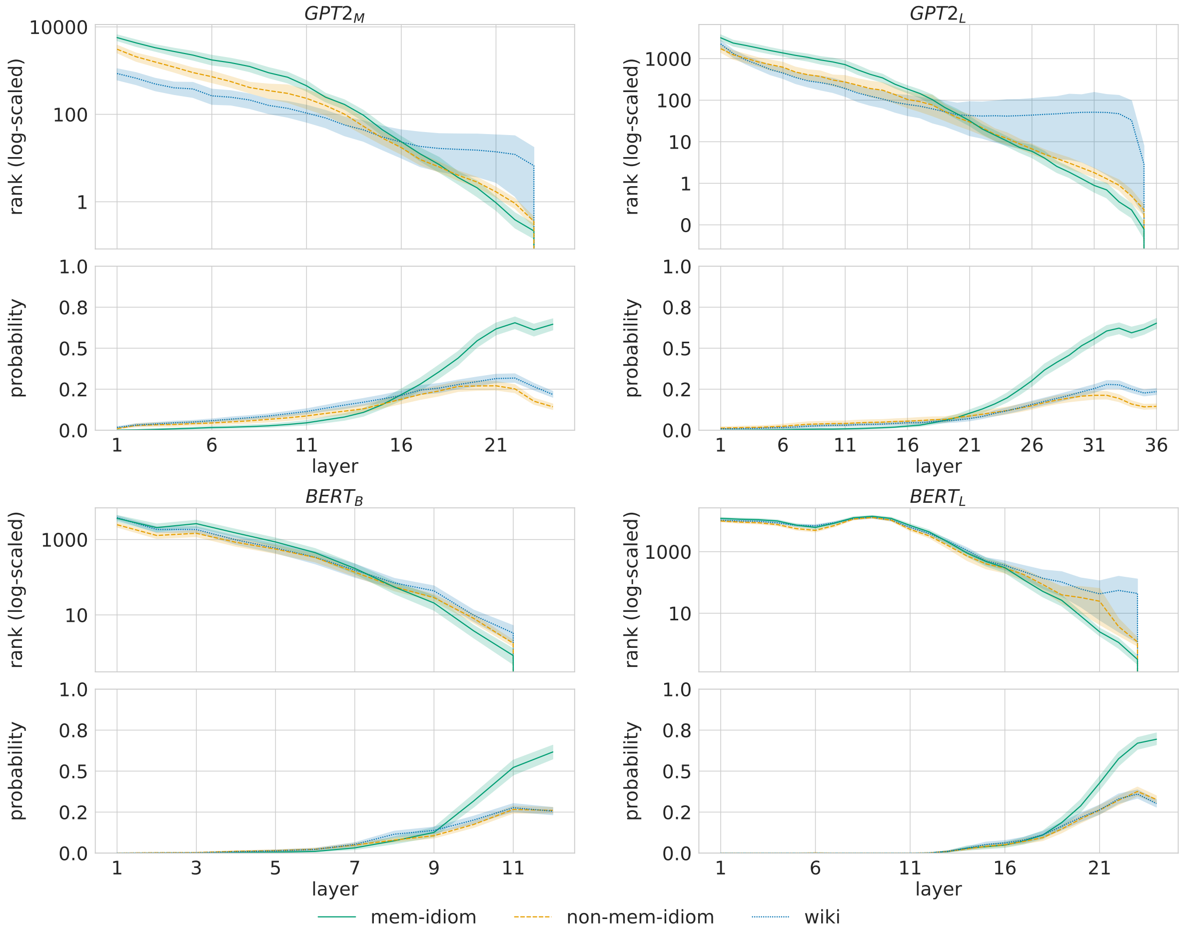


Figure 6: The predicted token's probability and log-scaled rank across layers, for memorized idioms (`mem-idiom`), non-memorized idioms (`non-mem-idiom`) and short sequences from Wikipedia (`wiki`).

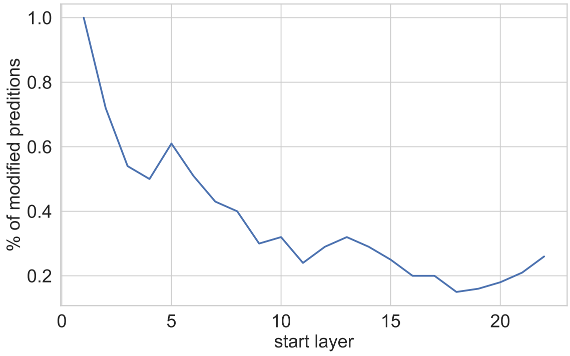


Figure 7: We visualize the effect of intervening in each 3-consecutive-layer range according to the procedure in §5.3.

C Intervention Experiments: Additional Analysis

C.1 Additional Visualization

In §5.3, we performed an intervention-based experiment to test the effect of zeroing out FFN sub-updates in layer computation. This produced a heat map of values corresponding to intervention's effect for each layer range. Here, we plot the intervention's effect across all 3-layer layer ranges. Note that since there are 24 layers, there are a total of 22 ranges of 3 consecutive layers. Fig. 7 visualizes the effect of intervening in each such range.

As discussed in §5.3, we observe a steep drop in effect in the first 10 layers, followed by a more leveled slope of decrease in the upper layers.

C.2 Analyzing Changes in Rank and Probability of the Target Token

In addition to measuring how often an intervention changes memorized predictions (§5.3), we further measure the average change in the rank and probability of the target token. Note that the original target rank for memorized predictions is always zero, as the target token is the top candidate in the original output distribution.

Change in target rank Fig. 8 shows the change in the target token's rank for all intervention experiments. Overall, we observe similar trends as in §5.3. First, zeroing out either dominant or non-dominant FFN sub-updates in upper layers (layers 11-24) does not affect memory recall, as the target token is still ranked as the top candidate in the output distribution. Moreover, zeroing out in early layers (1-10) damages memory recall as the target rank increases by > 20 positions. Specifically, zeroing-out non-dominant FFN sub-updates in layers 2-4 increases the target rank by 60, and disabling either dominant or non-dominant sub-updates in the first layer completely eliminates memory recall as the rank increases to > 6000.

Change in target probability Fig. 9 shows the change in the target token's probability for all intervention experiments. Unlike the prediction rank, which is mostly influenced by the lower layers during memory recall, the prediction probability is highly influenced by the intermediate and upper layers, where confidence boosting happens. Disabling FFN sub-updates in only three of these layers reduces the prediction probability by up to 33%. Considering the lower layers (1-9), zero-outs lead to a large probability decrease (up to 50%). This is expected since these interventions often change the prediction, i.e. they eliminate the target from the top of the output distribution.

D Experimental Setting Details

Tab. 6 shows the evaluated models' hyperparameters.

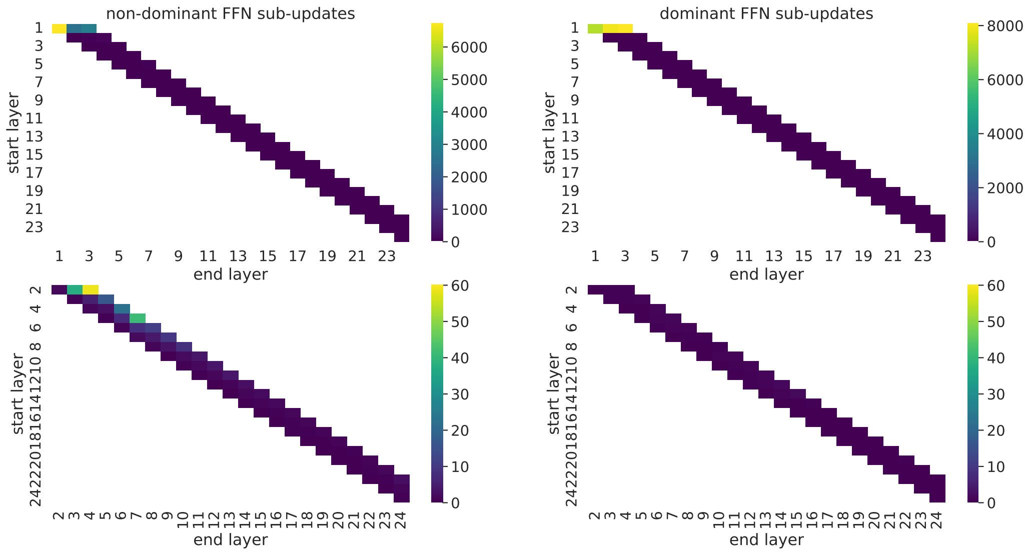


Figure 8: Change in the rank of the target token following intervention zeroing out non-dominant (left) and dominant (right) FFN sub-updates in GPT2$_\text{M}$. Each cell shows the average change in rank of the target token after zeroing out the sub-updates in the layers between the start and end layers. For readability, we provide plots with the first layer (top) and without (bottom).

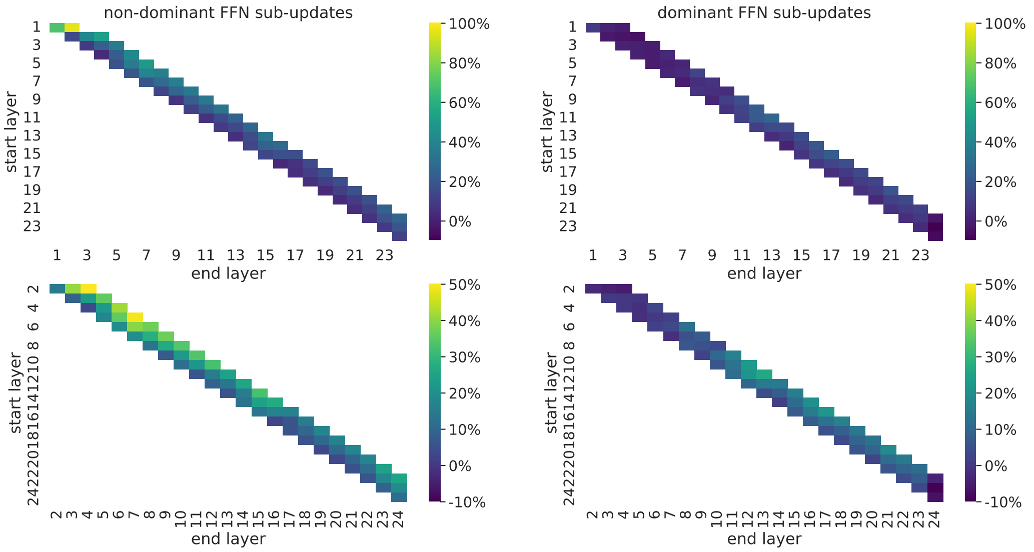


Figure 9: Change in the probability of the target token following intervention zeroing out non-dominant (left) and dominant (right) FFN sub-updates in GPT2$_\text{M}$. Each cell shows the average change in probability of the target token after zeroing-out the sub-updates in the layers between the start and end layers. For readability, we provide plots with the first layer (top) and without (bottom).

	GPT2$_\text{M}$	**GPT2$_\text{L}$**	**BERT$_\text{B}$**	**BERT$_\text{L}$**
Layers	24	36	12	24
Model hidden dimensions (d)	1024	1280	768	1024
Feed-forward dimensions (d_m)	4096	5120	3072	4096
Attention heads	12	20	12	16
Parameters	345M	774M	110M	340M
Vocabulary size (# of tokens)	50,256	50,256	30,522	30,522

Table 6: The models' hyperparameters.

A Discerning Several Thousand Judgments:
GPT-3 Rates the Article + Adjective + Numeral + Noun Construction

Kyle Mahowald
mahowald@utexas.edu
Department of Linguistics
The University of Texas at Austin

Abstract

Knowledge of syntax includes knowledge of rare, idiosyncratic constructions. LLMs must overcome frequency biases in order to master such constructions. In this study, I prompt GPT-3 to give acceptability judgments on the English-language Article + Adjective + Numeral + Noun construction (e.g., "a lovely five days"). I validate the prompt using the CoLA corpus of acceptability judgments and then zero in on the AANN construction. I compare GPT-3's judgments to crowdsourced human judgments on a subset of sentences. GPT-3's judgments are broadly similar to human judgments and generally align with proposed constraints in the literature but, in some cases, GPT-3's judgments and human judgments diverge from the literature and from each other.

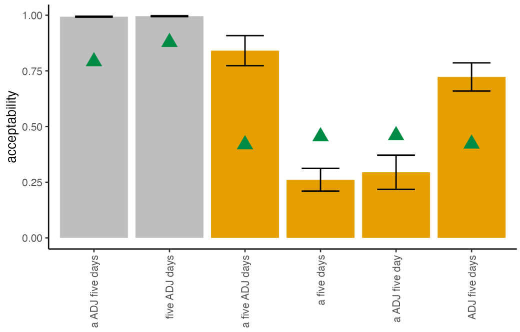


Figure 1: GPT-3 acceptability judgments (bars), compared to human ratings (green triangles) on a matched set of sentences. The comparison is between the AANN construction and the standard alternative, as well as 4 degenerate versions. Both humans and GPT-3 rank the AANN construction as being as acceptable as the standard, and all degenerate constructions are rated significantly lower.

1 Introduction

Consider the English Article + Adjective + Numeral + Noun (AANN) construction: " **a beautiful 228 pages** [iWeb]" or "The president has had **a terrible five weeks** [COCA]". Usually cardinal numbers precede the adjective ("five terrible weeks"), but here the adjective precedes the numeral. More strangely, the normally singular article "a" in this case is followed by a plural noun phrase.[1]

An eclectic dozen or so papers have been written on the construction, many focused on elucidating relevant semantic and syntactic constraints (Goldberg and Michaelis, 2017; Jackendoff, 1977; Dalrymple and King, 2019; Bylinina and Nouwen, 2018; Ionin and Matushansky, 2018, 2004; Solt, 2007; Keenan, 2013). The presence of the modifier is crucial: "a 228 pages" is unacceptable. The type of modifier is also crucial: "a pink 228 pages" is odd because color words are "stubbornly distributive" (Schwarzschild, 2011) and thus cannot refer to a set of items as a whole: the nominal phrase needs to function as a unit (Solt, 2007). These idiosyncratic constraints are typical of *constructions* (Goldberg, 2019).

Prior work on the AANN construction has focused on characterizing the semantic and syntactic constraints on the construction and proposing analyses in various frameworks. For instance, Solt (2007) focuses on how the construction coerces the phrase into a singular noun phrase; Keenan (2013) proposes treating it as akin to a partitive, and Dalrymple and King (2019) give an LFG analysis.

The same properties that make AANN interesting from the perspective of human language use—its low frequency but high sensitivity to constraints—also make it interesting from the perspective of LLMs and what they learn about linguistic structure. Indeed, much work on LLM syntactic competence has centered on ubiquitous abstract features of grammar like subject-verb number agreement (e.g., Linzen et al., 2016; Gulordava et al., 2018), part of speech (Tenney et al., 2019), and syntactic dependencies (e.g., Hewitt and Manning,

---

[1] Data and code: `https://github.com/mahowak/aann-public/`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 265–273
May 2-6, 2023

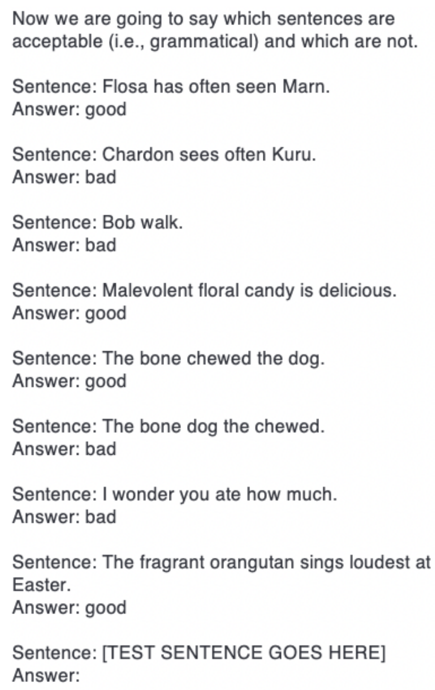
2019). That said, there has also been a recent spate of research on construction-grammar-inspired approaches in NLP, including studies showing that LLMs have access to construction information (Tayyar Madabushi et al., 2020; Tseng et al., 2022; Weissweiler et al., 2022) and capture verb argument construction biases (Hawkins et al., 2020) as well as fine-grained lexical semantic information (Petersen and Potts, 2022). Moreover, sentences with similar constructions cluster in embedding space (Li et al., 2022).

These works are valuable because better understanding how LLMs handle constructions could help us better understand what LLMs learn about linguistic structure (Baroni, 2021; Linzen and Baroni, 2021) and could also inform us as to what can be learned about language from primary data (Warstadt and Bowman, 2022, 2020). In our case, for an LLM to get the AANN construction right, a number of statistical regularities must be eschewed: "a" cannot be treated as a singular marker since the noun is plural, the normal ordering of the number and adjective must be reversed, and normal verb number agreement rules must in some cases be suspended. Understanding whether these heuristics (which work well for the vast majority of text) can be overcome can guide us towards future work understanding *how* they are overcome.

Here, I ask what GPT-3 `text-davinci-002` (now often classed as an instance of GPT-3.5) learns about the AANN construction by testing its sensitivity to several constraints proposed in the literature. In doing so, I treat the LLM as a linguistic test subject (Linzen et al., 2016; Futrell et al., 2019; Wilcox et al., 2021; Warstadt et al., 2019; Ettinger, 2020). I use a custom prompt to elicit quantitative grammaticality judgments (Schütze, 2016; Gibson and Fedorenko, 2013) from GPT-3, and show that the prompt performs well on CoLA, a data set of binary acceptability judgments on a carefully constructed 10,657 English sentences (Warstadt et al., 2019). I then unleash it on the AANN construction and compare GPT-3 to human ratings.

2 Methods

Attaining acceptability judgments from language models is not straightforward. Simply comparing sentence probabilities is difficult because they are dependent on the individual lexical items, as well as sentence length (Warstadt et al., 2020). I rely on the prompting paradigm to elicit acceptability judgments—validating the measure first using a known data set of judgments from CoLA. The prompt was created by drawing on a combination of CoLA training sentences and handcrafted sentences and iteratively experimenting. Ultimately, a diverse 8 sentences were chosen, some of which intentionally have low lexical probability to ensure that the model does not call all low-probability strings ungrammatical. Each prompt example sentence appears along with a binary judgment: "good" or "bad". Then, GPT-3 is passed the prompt, along with the critical test sentence, and asked to generate one more token (always either "good" or "bad"). The probability of the generated word (whether "good" or "bad") is our numerical rating.



Now we are going to say which sentences are acceptable (i.e., grammatical) and which are not.

Sentence: Flosa has often seen Marn.
Answer: good

Sentence: Chardon sees often Kuru.
Answer: bad

Sentence: Bob walk.
Answer: bad

Sentence: Malevolent floral candy is delicious.
Answer: good

Sentence: The bone chewed the dog.
Answer: good

Sentence: The bone dog the chewed.
Answer: bad

Sentence: I wonder you ate how much.
Answer: bad

Sentence: The fragrant orangutan sings loudest at Easter.
Answer: good

Sentence: [TEST SENTENCE GOES HERE]
Answer:

Figure 2: The prompt used for attaining grammaticality judgments. The target sentence is inserted, and then GPT-3 is asked to generate one token (overwhelmingly likely to be "good" or "bad"). That token's probability is taken as a rating.

To validate the measure, I tested the final prompt on the CoLA dev set. It attained accuracy of 84%, with a Matthew's correlation coefficient of 0.63. This is worse than, but in the ballpark of, human inter-annotator agreement on CoLA which is 86% and .697, respectively (Warstadt et al., 2019). It is also comparable to top performers on the GLUE leaderboard for the CoLA sub-task.

I used this technique to test the AANN con-

template	**temporal**: The family spent X in London; The diplomat worked X in Nairobi; The tourist stayed X in Papua New Guinea; **objects**: She bought X; They discovered X; Someone saw X; **human**: We served dinner to X; X greeted us at the door; We congratulated X; **art**: The newspaper reviewed X; I experienced X; Please enjoy X; **distance**: He drove X; Someone walked X; Someone traveled X; **unitlike**: Luis took in X; They consumed X; It lasted X
template for agreement task	**temporal**: X is/are just what you need; X is/are ideal **objects**: X is/are available; X make(s) a lovely gift; **human**: X regularly show(s) up at the door; X is/are here **art**: X was/were reviewed in the newspaper; X was/were enjoyed **distance**: X is/are a long way; X is/are not far; **unitlike** X was/were uncovered; X was/were make(s) an impression
adj	**ambig**: astonishing; incredible; impressive; disappointing; surprising; devastating; pathetic; remarkable; mediocre; unsatisfying; **qualitative**: lovely; beautiful; enchanting; soothing; charming; disgusting; uninviting; haunting; hideous; ugly; **quant**: mere; staggering; whopping; hefty; paltry; meager; extra; measly; substantial; record-setting; **stubborn**: large; big; small; round; tall; **color**: blue; green; red; yellow; orange; **human**: lucky; talented; graceful; fancy; friendly; collegial; hopeful; shy; bold; grinning
noun	**human**: soldiers; students; athletes; pianists; teammates; lawyers; doctors; actors; Americans; bankers; **objects**: desks; marbles; pencils; belts; forks; chairs; cans; bananas; apples; trays; **art**: movies; paintings; books; shows; operas; **temporal**: days; weeks; months; years; hours; **distance**: meters; feet; yards; blocks; steps; **unit_like**: pages; acts; paragraphs; awards; meals
num.	three; five; six; eight; ten; twenty; fifty; 500; 1000; 10,000; 21; 51; 512; 1,429; 21,234

Table 1: A superset of the items used, which were combined in various ways across experiments. In the templates, X is replaced by the AANN construction.

struction by templatically constructing sentences in which I parametrically vary the main sentence template, adjective, numeral, and nominal, from the superset shown in Table 1. Templates were designed to work with the key manipulations. Certain nouns work with some templates and not others, to ensure template/ noun pairs are always plausible.

Adjectives in the AANN construction behave differently depending on whether they are quantitative (i.e. modify the numeral as in "a mere 5 days"), qualitative (e.g., modify the noun as in "a beautiful five days), or are ambiguous between the two (e.g., "an astonishing five days" which leaves it unclear whether the number of days is astonishing, or the days themselves). It has been claimed (e.g., Dalrymple and King, 2019; Solt, 2007; Keenan, 2013) that quantitative and ambiguous adjectives are typically more acceptable than qualitative ones in AANN—although there are specific instances where qualitative adjectives are acceptable. I also consider "stubbornly distributive" (Schwarzschild, 2011) adjectives (e.g., "large" or "blue"), which "stubbornly" refer to individuals even when applied to a group. For instance, "The chairs are large." can refer only to the individual chairs being large, not to the collective group of chairs being large. It's claimed (Bylinina and Nouwen, 2018; Ionin and Matushansky, 2018; Keenan, 2013) that this same property makes a phrase like "a large five trees" less acceptable, compared to something like "a beautiful five trees" (in which it is possible for "beautiful" to refer to not just the individual trees but to the collection of trees).

Solt (2007) and others observe measure nouns work best in the AANN construction, but that other nouns can be okay as long as they can be treated as a single unit. I sampled nouns from 6 categories, as shown in Table 1. Also as shown in that table, I sampled numerals of various kinds but focused on "three" and "five" for the human experiments and most analyses. See Appendix B for a discussion of sensitivity to the numeral.

From these templates and candidate words, I generate semantically plausible sentences (meaning that, throughout the experiments, I only use human-appropriate sentence templates with human nouns and object-appropriate ones with object nouns). Depending on the specific question in each experiment, I run controlled subsets of these sentences (or their degenerate variants as in Experiment 1) through GPT-3 to obtain acceptability judgments, getting the probability of "good" or "bad" as the next word in the continuation.

I also use Amazon's Mechanical Turk to obtain human judgments on a subset of the test sentences. I asked raters to rate sentences on a rating bar scale from 1-10. For Experiments 1 and 3, each rater rated 3 critical sentences. They rated 18 critical sentences in Experiment 2 since there were many more conditions to test in that experiment.

To guard against raters becoming inured to the construction, half or more items were fillers not involving AANN. To maintain similar calibration between humans and GPT-3, these fillers always included all example judgments used in the GPT-3 prompt. I excluded participants who did the survey more than once, who did not rate the good filler items at least 1 point higher than the bad, or who did not have a US IP address. I obtained annotations for only a sample of sentences rated by GPT-3. When applicable, analyses focus on the union of sentences rated by both humans and GPT-3.

3 Exp. 1: AANN Fundamentals

First, I tested the basics of the AANN construction as laid out in the literature by having GPT-3 and human raters rate the AANN construction (e.g., "a beautiful five days") vs. the default ("five beautiful days") vs. a select four degenerate conditions: one with the order of the numeral and adjective switched, one with no modifier, one with a singular noun, and one with no article "a". To generate examples in each condition, I crossed 3 temporal nouns (days, weeks, months), with a low numeral (three), one of 14 appropriate adjectives, and one of 3 templates. For the resulting sentences, I attained human ratings from MTurk (after exclusions, 126 raters rating 3 sentences each, for 378 total ratings) and GPT-3 and focus on that subset for analysis.

Figure 1 shows results for this experiment. Although GPT-3 uses a wider range of the scale, both rate the AANN construction as just as good as the default and give lower ratings to the 4 degenerate versions. Humans rate the 4 degenerate constructions as about equally bad (all between .46 and .53), whereas GPT-3 rates the versions with swapped adjective/numeral order and a missing article as significantly better than the versions with a missing plural and a missing modifier. Running a mixed effect regression (Bates et al., 2015; Barr et al., 2013) comparing the AANN construction to the "default" construction and the degenerate alternatives (treating the default construction as the baseline, with random intercepts for adjective class, adjective, and template; and, for humans, for rater), both GPT-3 and humans show no significant difference in rating between AANN and default (both $p > .05$), but do show a significant difference between AANN and all 4 degenerate conditions (all $ps < .0001$). See Appendix C for regression details. Overall, I conclude that humans and GPT-3 "get" the AANN construction, even though they differ in the relative ratings of the bad variants.

4 Exp. 2: Adjectives and nouns

In this experiment, I focus on only the AANN construction and parametrically vary the kinds of adjectives (quantitative, ambiguously quantitative/qualitative, qualitative, human-referring, color adjective, stubbornly distributive; see Table 1 for examples) and kinds of nouns in the sentences (art, distance, human nouns, object nouns, temporal nouns, unit-like nouns; again see Table 1). This process produced a carefully controlled 12,960 unique sentences, from which we sampled a random subset for human ratings. After exclusions, there were 190 raters left who each rated 18 sentences, giving us ratings for 3,420 sentences.

I test whether GPT-3 and human raters agree with the attested claims that (a) more measure-like nouns (e.g., temporal nouns, distance nouns, and unit-like nouns) are more acceptable in AANN, (b) qualitative adjectives are acceptable in only some AANN cases, and (c) stubbornly distributive adjectives (including color words) are not acceptable in the AANN construction. To assess significance, I predicted acceptability separately for humans and GPT-3 in a mixed effect regression, using adjective class, noun class, and their interaction as predictors and with random effects for adjective, noun, numeral, and template. I treated qualitative adjectives with temporal nouns as the baseline.

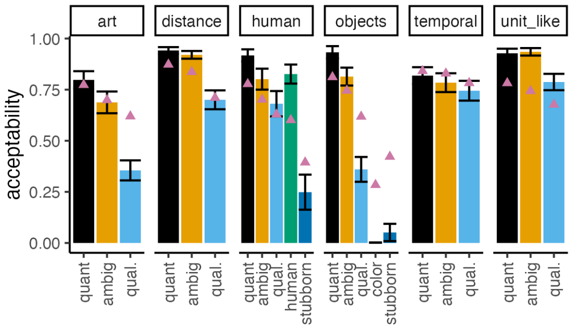


Figure 3: GPT-3 acceptability scores broken down by adjective type (x-axis), and noun type (on facets). Human ratings are pink triangles.

For both humans and GPT-3, there are significant differences in how nouns interact with adjectives (Figure 3). For temporal, distance, and unit-like nouns, all adjective types show high acceptability (although the qualitative adjectives are rated lowest). For art and object nouns, qualitative nouns score significantly worse ($p < .01$ for both humans and GPT-3) than ambiguous or quantitative nouns (an observation consistent with the literature). As predicted, colors and other stubbornly distributive adjectives (which can be tested only for humans and objects) show the lowest acceptability (significantly lower, $p < .01$, compared to qualitative adjectives, for both human and GPT-3 ratings). See Appendix D for regression details.

5 Exp. 3: Adjective Order

It is claimed that, in AANN, qualitative adjectives must appear before quantitative ones in order to be acceptable (Solt, 2007): "The family spent a beautiful mere five days in London." is preferred over "The family spent a mere beautiful five days in London.". To compare whether there is an effect of adjective ordering (putting the qualitative adjective

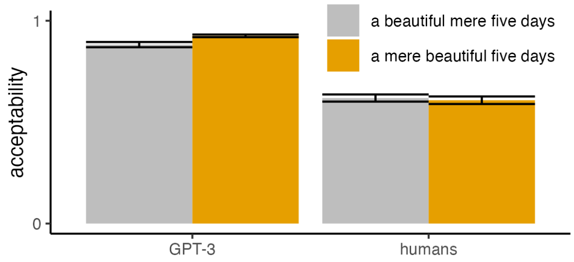


Figure 4: GPT-3 and human preference for adjective order (quantitative before qualitative; qualitative before quantitative).

before the quantitative one or vice versa), I ran an experiment crossing 3 templates; 5 adjectives (astonishing, impressive, beautiful, hideous, ugly); the noun "days"; and the numeral "three" or "five". I ran each sentence in two conditions (quantitative adjective first or qualitative adjective first). For instance, I compared: "The family spent a beautiful mere five days in London." to "The family spent a mere beautiful five days in London." This left 60 sentences total, rated by 99 raters (each sentence rated between 18 and 36 times; 1,782 ratings total).

GPT-3 significantly prefers the order *dispreferred* in the literature (quantitative first, as in "a mere beautiful five days"; $\beta = .04, p < .01$). Humans (n=99) showed no clear preference according to the model, under the same analysis (but with a random intercept for rater): $\beta = -.513, p > .05$. Thus, the attested claim does not replicate. But the ratings for these sentences are relatively low overall, so we should remain open to the possibility that there are better examples of the double-adjective constructions than the ones tested. See Appendix E for regression details.

6 Exp. 4: Verb Agreement

The AANN construction also challenges number agreement. AANN subjects sometimes take singular verbs (when the noun phrase would be singular anyway, as in "A mere fifty cents for a cup of coffee sounds/*sound reasonable to me!"); sometimes plural ("A delicious four courses *was/were served in the main dining room."), and sometimes either ("A healthy two runs weekly was/were prescribed by the doctor." See Keenan (2013).

I tested agreement by comparing phrases which differed only in the verb number (e.g., comparing "A beautiful five days is..." vs. "A beautiful five days are...."). Sampling a subset of noun classes (art, distance, human nouns, objects, unit-like nouns, and temporal nouns), I generated a total of 280 sentences (each appearing with a singular or plural verb for a total of 560 sentences) and attained judgments from GPT-3. As shown in Figure 5, these results replicated in detail several attested judgments in the literature: art nouns, human nouns, unit-like nouns, and object nouns all prefer the plural for the AANN construction (art and humans almost categorically so). Distance nouns and temporal nouns prefer the singular (although temporal nouns can also take the plural). See Appendix F for details.

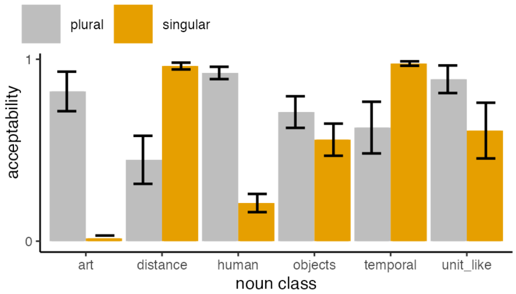


Figure 5: Mean GPT-3 acceptability ratings in the AANN construction for plural and singular verb agreement, as a function of noun class.

7 Conclusion

This work shows that GPT-3 can recognize and use the *form* of the AANN construction in a relatively (but not perfectly) human-like way, matching judgments across a variety of conditions, which is not the same thing as showing that it understands the meaning or *function* of the construction (Mahowald et al., 2023). In future work, we should study not just the form of the construction but its *construal* (Trott et al., 2020). This may be particularly relevant since Weissweiler et al. (2022), which studies the "Xer the Yer" construction (e.g., "the better the criticism, the better the science" [COCA]), show that LLMs can recognize the construction but fail at tests of understanding its meaning.

That said, GPT-3's performance on the AANN construction demands a significant amount of constructional knowledge and involves overriding major widespread "rules" of grammar (e.g., that the article *a* signals a singular noun). Future work exploring how LLMs override those heuristics, perhaps using causal intervention techniques (Ravfogel et al., 2021; Geiger et al., 2021) could illuminate their syntactic processing mechanisms.

8 Limitations

Many researchers have pointed out that the AANN construction is sensitive to context. For instance, Solt (2007) points out that "a hungry thirty hikers" may be acceptable in some sentences (namely ones where "hikers" is more easily construed as a single unit) than others. Because of the cost of running sentences through GPT-3 and on MTurk and the combinatoric nature of the construction, I could only run a constrained number of templates and did not consider larger context. Broader context may matter and so these results should be taken to apply to the particular contexts shown, which is why I show the exact templates used in the main text.

Another limitation is that the task of how to prompt GPT-3 for grammaticality judgments is not a settled question. Our focus in this paper was not on settling it, so I used one particular method for prompting GPT-3. While I explored the prompt space some, it is likely there are better prompts out there that would make GPT-3's performance on the task better. It's also possible that prompting GPT-3 for grammaticality judgments is not the best way to ascertain its knowledge of language and that a more naturalistic task would produce different results.

As has often been pointed out in the linguistics literature, naive human judges of out-of-context sentences may sometimes tap into different processes than they would when encountering language in the wild. Moreover, English is not a monolith and this construction's acceptability may vary across dialects of English. In a more detailed human study, it would be possible to tease apart effects of different dialects on ratings.

GPT-3 `text-davinci-002` is often categorized as GPT-3.5 because it is trained on more than just a word prediction task, and so we should not interpret its output as being purely reflective of what is learned by word prediction alone.

Finally, I note that I use templatically constructed sentences, which differ in important ways from naturalistic ones.

9 Acknowledgments

I acknowledge funding from NSF Grant 2104995. For helpful comments, I thank Robbie Kubala, Adele Goldberg, Gabriella Chronis, Katrin Erk, Alex Warstadt, Steve Wechsler, Liam Blything, attendees of the Princeton language group meeting, and an anonymous three reviewers.

References

Marco Baroni. 2021. On the proper role of linguistically-oriented deep net analysis in linguistic theorizing. *arXiv preprint arXiv:2106.08694*.

Dale J Barr, Roger Levy, Christoph Scheepers, and Harry J Tily. 2013. Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of memory and language*, 68(3):255–278.

Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. 2015. Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software*, 67(1):1–48.

Lisa Bylinina and Rick Nouwen. 2018. On "zero" and semantic plurality. *Glossa: a journal of general linguistics*, 3(1). Publisher: Open Library of Humanities.

Mary Dalrymple and Tracy Holloway King. 2019. An amazing four doctoral dissertations. *Argumentum*, 15(2019). Publisher: Debreceni Egyetemi Kiado.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 32–42, Minneapolis, Minnesota. Association for Computational Linguistics.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. 2021. Causal abstractions of neural networks. *Advances in Neural Information Processing Systems*, 34:9574–9586.

Edward Gibson and Evelina Fedorenko. 2013. The need for quantitative methods in syntax and semantics research. *Language and Cognitive Processes*, 28(1-2):88–124.

Adele E Goldberg. 2019. *Explain me this: Creativity, competition, and the partial productivity of constructions*. Princeton University Press.

Adele E Goldberg and Laura A Michaelis. 2017. One among many: Anaphoric one and its relationship with numeral one. *Cognitive science*, 41:233–258.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Robert Hawkins, Takateru Yamakoshi, Thomas Griffiths, and Adele Goldberg. 2020. Investigating representations of verb bias in neural language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4653–4663, Online. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138. Association for Computational Linguistics. Event-place: Minneapolis, Minnesota.

Tania Ionin and Ora Matushansky. 2004. A singular plural. In *Proceedings of WCCFL*, volume 23, pages 399–412.

Tania Ionin and Ora Matushansky. 2018. *Cardinals: The syntax and semantics of cardinal-containing expressions*, volume 79. MIT Press.

Ray Jackendoff. 1977. X syntax: A study of phrase structure. *Linguistic Inquiry Monographs Cambridge, Mass*, pages 1–249.

Caitlin Keenan. 2013. “A pleasant three days in Philadelphia”: Arguments for a pseudopartitive analysis. *University of Pennsylvania Working Papers in Linguistics*, 19(1):11.

Bai Li, Zining Zhu, Guillaume Thomas, Frank Rudzicz, and Yang Xu. 2022. Neural reality of argument structure constructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7410–7423, Dublin, Ireland. Association for Computational Linguistics.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7:195–212.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535. Publisher: MIT Press.

Kyle Mahowald, Anna A Ivanova, Idan A Blank, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. 2023. Dissociating language and thought in large language models: a cognitive perspective. *arXiv preprint arXiv:2301.06627*.

Erika Petersen and Christopher Potts. 2022. Lexical semantics with large language models: A case study of English *break*. Ms., Stanford University.

Shauli Ravfogel, Grusha Prasad, Tal Linzen, and Yoav Goldberg. 2021. Counterfactual interventions reveal the causal effect of relative clause representations on agreement prediction. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 194–209, Online. Association for Computational Linguistics.

Carson Schütze. 2016. *The empirical base of linguistics: Grammaticality judgments and linguistic methodology*. Language Science Press.

Roger Schwarzschild. 2011. Stubborn distributivity, multiparticipant nouns and the count/mass distinction. In *Proceedings of NELS*, volume 39, pages 661–678. Graduate Linguistics Students Association, University of Massachusetts. Issue: 2.

Stephanie Solt. 2007. Two types of modified cardinals. In *International Conference on Adjectives. Lille*.

Harish Tayyar Madabushi, Laurence Romain, Dagmar Divjak, and Petar Milin. 2020. CxGBERT: BERT meets construction grammar. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4020–4032, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Sean Trott, Tiago Timponi Torrent, Nancy Chang, and Nathan Schneider. 2020. (Re)construing meaning in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5170–5184, Online. Association for Computational Linguistics.

Yu-Hsiang Tseng, Cing-Fang Shih, Pin-Er Chen, Hsin-Yu Chou, Mao-Chang Ku, and Shu-Kai Hsieh. 2022. CxLM: A construction and context-aware language model. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6361–6369, Marseille, France. European Language Resources Association.

Alex Warstadt and Samuel R. Bowman. 2020. Can neural networks acquire a structural bias from raw linguistic data? In *Proceedings of the 42th Annual Meeting of the Cognitive Science Society - Developing a Mind: Learning in Humans, Animals, and Machines, CogSci 2020, virtual, July 29 - August 1, 2020*. cognitivesciencesociety.org.

Alex Warstadt and Samuel R Bowman. 2022. What artificial neural networks can tell us about human language acquisition. In *Algebraic Structures in Natural Language*, pages 17–60. CRC Press.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Leonie Weissweiler, Valentin Hofmann, Abdullatif Köksal, and Hinrich Schütze. 2022. The better your syntax, the better your semantics? probing pretrained language models for the english comparative correlative. *arXiv preprint arXiv:2210.13181*.

Ethan Wilcox, Pranali Vani, and Roger Levy. 2021. A targeted assessment of incremental processing in neural language models and humans. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 939–952, Online. Association for Computational Linguistics.

A Frequency of AANN

I sampled the AANN construction on SketchEngine for the English Web 2020 corpus with the following prompts: `[lemma="a[n]*"] [tag="JJ.*"] [tag="CD.*"] [tag="NNS"]` for AANN and `[tag="CD.*"] [tag="JJ.*"] [tag="NNS"]` for the vanilla construction. The AANN showed up 23.62 per million tokens, compared to 457.22 for the vanilla construction. Of the AANN construction examples, the vast majority contain quantitative adjectives (e.g., *mere*, *staggering*, etc.) and measurement nouns. Of a sample of 200 AANN constructions that I manually inspected on SketchEngine, none contained a qualitative adjective.

B Numerals

I sampled numerals from round low numbers (“three”, “five”, “six”, “eight”, “ten”), medium numbers (“twenty”, “fifty”), high numbers (500, 1000, 10,000), non-rounded medium numbers (21 and 51), and non-rounded high numbers (1,429 and 21,234). I focused on round low numbers (“three” and “five”) for most analyses.

For the overall analysis (across all adjectives and nouns in the main experiment), I get the below AANN ratings from GPT-3.

numclass	**example**	**avg. AANN score**
num-high	500	0.68
num-high_odd	1,429	0.93
num-low	three	0.80
num-med	twenty	0.70
num-med_odd	21	0.73

Low numbers like “three” and “five” are rated higher somewhat than other numbers, with the exception of high, non-round numbers (e.g., 1,429). These are rated unusually highly. Because of this anomalous behavior, I focus mostly on the low numerals for the analysis and did not attain human ratings for these other numerals. It remains an open question how humans would rate a sentence like “We spent a beautiful 1,429 days in London.”

C AANN vs. default vs. degenerate regression

I run the following regression using the R `lme4` (Bates et al., 2015) package.

```
lmer(value ~ construction +
             (1|adjclass) +
             (1|adj) +
             (1 | temp))
```

where I treat the construction as a predictor (with the AANN constructor as a default) and adjclass, adjective, template as random effects. (Other random effects were removed to help the model converge, by iteratively removing ones with the smallest variance.)

For humans, there is also a random intercept for worker.

coef.	β_{gpt3}	Sig	β_{human}	Sig
(Intercept)	0.99	*	.81	*
five ADJ days	0.00		.06	
a five ADJ days	-0.22	*	-.39	*
a five days	-0.87	*	-.35	*
a ADJ five day	-0.72	*	-.35	*
ADJ five days	-0.20	*	-.38	*

Table 2: Fixed effect coefficients for GPT-3 comparing across constructions on the subset of sentences also run with human annotators

D Regression for adjective x noun sub-experiment with GPT-3 and humans

To assess significance, I predicted acceptability separately for humans and GPT-3 in a mixed effect regression, using adjective class, noun class, and their interaction as predictors and with random effects for adjective, noun, numeral, and template. I treated qualitative adjectives with temporal nouns as the baseline. For the human regression, it was identical except I included a random intercept for each worker, with a random slope for adjective class and noun class (but not their interaction, due to convergence issues).

```
lmer(rating ~ adjclass *
        nounclass +
        (1|adj) +
        (1|noun) +
        (1|num) +
        (1 |template))
```

	beta	t-value	p<.05
(Intercept)	0.80	12.09	*
adj-quant	0.02	0.27	
adj-stubborn	-0.62	-10.27	*
adj-ambig	-0.02	-0.32	
noun-unit_like	0.14	3.49	*
noun-objects	0.02	0.58	
noun-human	0.03	0.76	
noun-distance	0.14	3.40	*
noun-art	-0.10	-2.33	*
adj-quant:noun-unit_like	-0.02	-0.80	
adj-quant:noun-objects	0.09	2.63	*
adj-stubborn:noun-objects	-0.17	-3.64	*
adj-quant:noun-human	0.08	2.24	*
adj-quant:noun-distance	-0.01	-0.28	
adj-quant:noun-art	0.07	2.35	*

Table 3: Fixed effect coefficients for GPT-3 comparing the adjective class x noun class manipulation.

	beta	t-value	p<.05
(Intercept)	0.73	16.84	*
adj-quant	0.10	2.63	*
adj-stubborn	-0.25	-5.58	*
adj-ambig	0.09	2.65	*
noun-unit_like	-0.08	-3.11	*
noun-objects	-0.09	-3.38	*
noun-human	-0.12	-4.25	*
noun-distance	0.00	0.17	
noun-art	-0.12	-4.28	*
adj-quant:noun-unit_like	0.02	0.72	
adj-quant:noun-objects	0.06	2.04	*
adj-stubborn:noun-objects	-0.04	-0.91	
adj-quant:noun-human	0.05	1.62	
adj-quant:noun-distance	0.03	1.27	
adj-quant:noun-art	0.04	1.51	

Table 4: Fixed effect coefficients for human annotators comparing the adjective class x noun class manipulation.

E Regression for adjective ordering

I ran a mixed effect linear regression predicting the GPT-3 score from the condition (quantitative-first vs. qualitative-first), with random intercepts for adjective, numeral, and template.

```
l = lmer(value ~ cond +
          (1|adj)  +
         (1|num) +
         (1 |template)
```

	beta	t-value	p<.05
(Intercept)	0.57	4.88	*
singular	0.35	3.14	*
noun-unit_like	0.27	3.37	*
noun-objects	0.17	2.50	*
noun-human	0.29	4.27	*
noun-distance	-0.18	-2.25	*
noun-art	0.20	2.52	*
singular:noun-unit_like	-0.63	-6.40	*
singular:noun-objects	-0.50	-6.12	*
singular:noun-human	-1.07	-12.96	*
singular:noun-distance	0.16	1.65	
singular:noun-art	-1.16	-11.71	*

Table 5: Fixed effect coefficients and significance values for an experiment comparing whether nouns in the AANN construction prefer singular or plural verbs.

F Agreement

I ran a regression predicting the rating based on the nounclass, and its interaction with whether there was singular plural. I included random intercepts for noun, adjective, and template, with a random slope for whether the verb was singular or plural on the noun factor.

```
lmer(rating ~ singplur *
        nounclass +
        (1 + singplur|noun) +
        (1|adj) +
        (1|template))
```

Results appear in Table 5, where the baseline values are temporal nouns in the plural (e.g., “days”). Singular verbs are preferred overall, relative to plurals for the temproal nouns (main effect of “singular”). There are various main effects of nouns, but the critical effects here are interactions. There are significant effects such that, *relative to temporal nouns* (e.g., “days”), unit-like nouns are less likely to prefer singular agreement, object nouns are less likely to prefer singular agreement, human nouns are less likely to prefer singular agreement, and art nouns are less likely to prefer singular agreement. Distance nouns are more likely than temporal nouns to prefer singular agreement (but not significantly so).

Triple-Hybrid Energy-based Model Makes Better Calibrated Natural Language Understanding Models

Haotian Xu
Ailbaba Group
htxu91@gmail.com

Yingying Zhang
East China Normal University
yyzhang@fem.ecnu.edu.cn

Abstract

Though pre-trained language models achieve notable success in many applications, it's usually controversial for over-confident predictions. Specifically, the in-distribution (ID) miscalibration and out-of-distribution (OOD) detection are main concerns. Recently, some works based on energy-based models (EBM) have shown great improvements on both ID calibration and OOD detection for images. However, it's rarely explored in natural language understanding tasks due to the non-differentiability of text data which makes it more difficult for EBM training. In this paper, we first propose a triple-hybrid EBM which combines the benefits of classifier, conditional generative model and marginal generative model altogether. Furthermore, we leverage contrastive learning to approximately train the proposed model, which circumvents the non-differentiability issue of text data. Extensive experiments have been done on GLUE and six other multiclass datasets in various domains. Our model outperforms previous methods in terms of ID calibration and OOD detection by a large margin while maintaining competitive accuracy.

1 Introduction

Since many industrial applications involve safety -critical domains such as healthcare (Li et al., 2019; Blinov et al., 2020; Li et al., 2020; Rasmy et al., 2021; Sarabadani, 2019), anticipating credit card defaults (Sun and Vasarhalyi, 2021) and self-driving (Khaitan et al., 2021), it's essential for machine learning systems to provide not only accurate but also well-calibrated predictions (Li et al., 2019), which can help to decide whether it can be trusted.

However, models achieving high accuracy usually lead to overconfidence and miscalibration (Guo et al., 2017; Thulasidasan et al., 2019; Ovadia et al., 2019). This motivates an interesting and important area that attempts to achieve a better trade-off between accuracy and calibration. In addition to ID calibration, it's more important for machine learning models to produce high uncertainty when OOD data is observed, rather than to produce wrong yet wildly confident predictions.

Related works. To overcome the problem of miscalibration, numerous methods have been proposed. The natural way is post-hoc calibration that transforms the output of the original network into calibrated confidence scores while maintaining the network's accuracy (Guo et al., 2017; Rahimi et al., 2020; Jung et al., 2020). The second method to mitigate miscalibration is to add regularizations during training such as label smoothing (Wang et al., 2020), Mixup (Zhang et al., 2018). Desai and Durrett (2020) and Kong et al. (2020) further conveys that the aforementioned methods can be applied to improve the calibration of pre-trained language models on NLU tasks. The third way is to design a specific loss function to minimize the discrepancy between accuracy and confidence. For example, Kong et al. (2020) lately propose the ID and OOD regularizer to leverage the relationship between accuracy and uncertainty, and it obtains a significant improvement over previous methods in ID calibration and OOD detection.

Energy-based Models. In another line of work, Joint EBM (JEM; Grathwohl et al., 2019) has been shown great improvements on ID calibration and OOD detection for images without explicit calibration correction mechanism. The core idea is to reinterpret a joint distribution $p_\theta(x, y)$ from a neural classifier $p_\theta(y|x)$ in the perspective of EBMs and jointly optimize the marginal distribution $p_\theta(x)$ and a neural classifier $p_\theta(y|x)$. Elflein et al. (2021) further investigate the OOD detection performance with different training approaches for $p_\theta(x)$ such as Stochastic Gradient Langevin Dynamics (SGLD; Welling and Teh, 2011), Sliced-Score-Matching (SSM; Song et al., 2020) and Variational Entropy Regularized Approximate maximum likelihood (VERA; Duvenaud et al., 2021).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 274–285
May 2-6, 2023 ©2023 Association for Computational Linguistics

Besides, Du and Mordatch (2019) propose an implicit generative models based on EBMs (IGEBM) and apply SGLD to optimize $p_\theta(x|y)$. It performs significantly better OOD detection than other generative models. However, as shown by Grathwohl et al. (2019), the accuracy of IGEBM has dropped dramatically to 49.1% on CIFAR10 while standard finetuning can achieve 95.8% accuracy. This result indicates that different loglikelihood factorization leads to great gaps in accuracy, ID calibration and OOD detection. Moreover, these training methods such as SGLD, SSM, VERA need to calculate the gradients about inputs, the none differentiability of text data limits the application of these methods on both calibration and OOD detection for NLU tasks.

Recently, He et al. (2021) proposes a joint training of classifier $p_\theta(y|x)$ and marginal distribution $p_\theta(x)$ based on Residual EBM (Deng et al., 2019) for NLU tasks. Different from JEM, their model is more flexible by designing various energy functions for marginal distribution without any restriction on joint distribution $p_\theta(x, y)$. To estimate the parameters of marginal distribution $p_\theta(x)$, they propose to apply noise contrastive estimation (NCE; Gutmann and Hyvärinen, 2010) to train the energy model by discriminating the real data and the fake data generated by a noise distribution. To make the noise distribution as close as possible to the data distribution, they finetune a task-specific GPT-2 (Radford et al.). Though it achieves improvements on ID calibration, it's often resource-intensive compared to previous methods to finetune GPT-2 (Li et al., 2022). Moreover, the quality and quantity of fake samples generated by noise distribution has great impacts for NCE training (He et al., 2021; Gutmann and Hyvärinen, 2010).

Contribution. Methodologically, we propose a novel model namely Triple-Hybrid Energy-based Model (THEM) based on the JEM (Grathwohl et al., 2019) through different decompositions of log$p(x, y)$ into a unified framework. Compared to Grathwohl et al. (2019) and Du and Mordatch (2019), our model combines the classifiers $p(y|x)$, class-conditional density $p(x|y)$ and unconditional data density $p(x)$ into a hybrid model. Due to the none differentiability of text data, we further propose to adopt InfoNCE (Oord et al., 2018) with memory bank (He et al., 2020) to approximate the normalized constant of EBM efficiently. This method makes it possible for EBM training on NLU tasks which is not well explored in previous works regardless of input differentiability. We conduct comprehensive experiments with BERT (Kenton and Toutanova, 2019) and RoBERTa (Liu et al., 2019) as the backbone and demonstrate the effectiveness of our framework on various datasets including GLUE (Wang et al., 2018) and six multiclass classification datasets (Kong et al., 2020) on various domains. Not only the experimental results show that our method achieves significant improvements in ID calibration and OOD detection with competitive accuracy over previous methods, but also it is more robust with respect to the temperature and size of memory bank compared to contrastive learning trained EBM including JEM(CL), IGEBM(CL) and HDGE.

Overall, the contributions can be summarized as follows:

- We propose a Triple-Hybrid Energy-based model (THEM) and apply InfoNCE with memory bank to optimize it efficiently and effectively for **discrete data**. It achieves significantly better performance compared to strong baselines including He et al. (2021) and Kong et al. (2020) in terms of ID calibration and OOD detection.
- We apply this training technique to JEM and IGEBM to obtain JEM(CL) and IGEBM(CL) respectively. THEM and JEM(CL) achieves better ID calibration and OOD detection compared to HDGE and IGEBM(CL) in average.
- We further study the effect of the temperature and size of memory bank for contrastive learning on ID calibration and OOD detection. THEM is more robust to these hyperparameters than JEM(CL) and HDGE(CL).

2 Preliminaries: Joint Energy Model and Contrastive Learning

Joint Energy Model (JEM). Energy-based models (EBMs; LeCun et al., 2006) measure the compatibility of the input variables $x \in \mathcal{X}$ and target variables $y \in \mathcal{Y}$ with an energy function $E_\theta(x, y) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, which is the main building block. Low energy corresponds to high compatability. With E_θ, the probability for data in an EBM can be written as

$$p_\theta(x, y) = \frac{\exp\left(-E_\theta(x, y)\right)}{Z_\theta}, \tag{1}$$

where Z_θ is the normalizing constant. EBMs are flexible to parameterize since they do not make restrictions on the tractability of Z_θ.

Joint Energy Model (JEM; Grathwohl et al., 2019) reinterpret a classifier $p_\theta(y|x)$ in supervised learning as an EBM for the joint distribution $p_\theta(x, y)$. Specifically, $p_\theta(y|x)$ is a categorical distribution:

$$p_\theta(y|x) = \frac{\exp(f_\theta(x)[y])}{\sum_y \exp(f_\theta(x)[y])}, \tag{2}$$

where $f_\theta(x) : \mathbb{R}^D \to \mathbb{R}^K$ maps each data point $x \in \mathbb{R}^D$ to K real-valued numbers known as logits, and $f_\theta(x)[y]$ indicates the logit of label y. JEM defines an EBM of the joint distribution with the same logits f_θ:

$$p_\theta(x, y) = \frac{\exp(f_\theta(x)[y])}{Z_\theta}, \tag{3}$$

where energy function $E_\theta(x, y) = -f_\theta(x)[y]$. To retain discriminative performance of $p_\theta(y|x)$, JEM factorizes the loglikelihood as

$$\log p_\theta(x, y) = \log p_\theta(x) + \log p_\theta(y|x), \tag{4}$$

and apply EBM training to benefit from generative models $p_\theta(x)$. Grathwohl et al. (2019) and Elflein et al. (2021) have shown that EBM training of the joint distribution improves calibration and out-of-distribution detection with various training methods.

Constrastive Learning. Our work is also related to contrastive learning, in that we approximate $\log p_\theta(x)$ and $\log p_\theta(x|y)$ by constrastive loss. Contrastive learning achieves remarkable success on downstream tasks, includes image classification, video understanding, knowledge distillation, etc (Khosla et al., 2020; Chen et al., 2020). In constrastive learning, a widely-used objective has the following form (Oord et al., 2018):

$$-\mathbb{E}_{p_{\text{data}}(x)}\left[\log \frac{\exp\{t_\theta(x)^\top t'_\theta(x)\}}{\sum_{i=1}^{N} \exp\{t_\theta(x)^\top t'_\theta(x_i)\}}\right], \tag{5}$$

where $t_\theta(x)$ and $t'_\theta(x)$ map each data point x to two different representation spaces. This is usually called InfoNCE loss. Different from existing methods for EBM training, we propose to leverage constrastive learning approximation for effective learning, without considering the generation ability such as SGLD, SSM and so on.

3 Triple-Hybrid Energy-based Model

Motivation. Many works (Grathwohl et al., 2019; Elflein et al., 2021; Du and Mordatch, 2019) have shown that EBMs could significantly reduce the expected calibration error and improve out-of-distribution detection for image classification. Specifically, the JEM proposed in Grathwohl et al. (2019) factorizes the joint distribution $\log p_\theta(x, y)$ into $\log p_\theta(x) + \log p_\theta(y|x)$, where $\log p_\theta(y|x)$ is to maintain the classification performance and $\log p_\theta(x)$ is the generative term which contributes to better calibration and out-of-distribution detection. On the contrary, the IGEBM proposed in Du and Mordatch (2019) factorizes the joint distribution $\log p_\theta(x, y)$ into $\log p_\theta(y) + \log p_\theta(x|y)$ for implicit generation and surprisingly find that it achieves better OOD performance. However, lack of $p_\theta(y|x)$ leads to terrible classification performance. It's shown in Grathwohl et al. (2019) that the classification accuracy dropped dramatically to 49.1% on the CIFAR10 dataset, while the accuracy is 92.9% by JEM.

On the other hand, Liu and Abbeel (2020) proposed a hybrid discriminative-generative energy-based model (HDGE) for both classification and generation. The loss function consists of a discriminative conditional log-likelihood $\log p_\theta(y|x)$ and a generative conditional log-likelihood $\log p_\theta(x|y)$. Compared to IGEBM, it includes $\log p_\theta(y|x)$ and thus achieves better classification performance. Compared to JEM, it includes the conditional generative model, rather than the marginal generative model. In other words, JEM targets to reduce the energy for data from the population $p_\theta(x)$, while HDGE aims at reducing the energy for compatible pair (x, y). This motivates us to combine the benefits of both conditional and marginal generative model for better calibration and OOD detection.

Triple-Hybrid Energy-based Model (THEM). We propose to make a hybrid model of the triple $\log p_\theta(y|x)$, $\log p_\theta(x|y)$ and $\log p_\theta(x)$, called Triple Hybrid Energy-based Model (THEM) and the objective function is

$$\mathbb{E}_{p_{\text{data}}(x,y)}\left[\log p_\theta(y|x) + \log p_\theta(x|y) + \log p_\theta(x)\right], \tag{6}$$

where $p_\theta(y|x)$ is the standard softmax neural classifer and the generative models $p_\theta(x), p_\theta(x|y)$ serve as regularization, always accompanied with better calibration and OOD detection.

From another perspective, we combine the two factorizations of the joint distribution $\log p_\theta(x, y)$ from JEM and IGEBM. We remark that the joint distribution can also be factorized as $(\log p_\theta(x) +$

$\log p_\theta(x|y) + \log p_\theta(y) + \log p_\theta(x|y))/2$. Our proposed THEM utilizes this factorization and treats $p_\theta(y)$ known as the label frequencies in data, which does not need to be optimized. Now we are ready to resolve the computational issues for THEM in the following.

Neural Classifier. The neural classifier term is easy to cope with. Specifically,

$$p_\theta(y|x) = \frac{\exp(f_\theta(x)[y])}{\sum_y \exp(f_\theta(x)[y])}, \tag{7}$$

where $f_\theta(x)[y]$ is the logit of label y. Thus we can derive the first term in (6) as the traditional cross-entropy loss:

$$\mathbb{E}_{p_{\text{data}}(x,y)} [\log p_\theta(y|x)] . \tag{8}$$

Conditional Generative Likelihood. The conditional generative likelihood can be derived from the joint distribution:

$$\begin{aligned}\log p_\theta(x|y) &= \log \frac{p_\theta(x,y)}{p_\theta(y)} = \log \frac{p_\theta(x,y)}{\sum_x p_\theta(x,y)}\\ &= \log \frac{\exp(f_\theta(x)[y])}{Z_\theta(y)},\end{aligned} \tag{9}$$

where $Z_\theta(y) = \sum_x \exp(f_\theta(x)[y])$. By definition, this is also an EBM with energy function $E_\theta(x, y) = -f_\theta(x)[y]$. Energy-based models are well-known to be difficult to train. The Fenchel duality method used in Chen et al. (2021b) can estimate $Z_\theta(y)$. The stochastic gradient langevin dynamics (SGLD) adopted in Grathwohl et al. (2019) can approximate the gradient of $\log p_\theta(x|y)$. However, these methods require to calculate the derivative with respect to the input x and thus can't be applied to discrete data such as text tasks.

Approximation with Contrastive Learning. The above training methods are successful for generation purpose. Differently, we focus on classification with better calibration and OOD detection. As such, we propose to coarsely approximate the normalization constant as

$$Z_\theta(y) \approx \sum_{i=1}^{N} \exp(f_\theta(x_i)[y]), \tag{10}$$

where x_i is sampled from the data no matter whether y_i is equal to y or not. The second term in (6) is approximately

$$\mathbb{E}_{p_{\text{data}}(x,y)} [\log p_\theta(x|y)] \approx \log \frac{\exp(f_\theta(x)[y])}{\sum\limits_{i=1}^{N} \exp(f_\theta(x_i)[y])}. \tag{11}$$

Since the samples for approximation are incorporated in the denominator using the same label y, the logits $f_\theta(x)[y]$ can be treated as the score function of input-label contrast (Rethmeier and Augenstein, 2021). As a result, this objective can be seen as the InfoNCE (5) in contrastive learning. Liu and Abbeel (2020) also proposed this approximation for image classification, while it's more suitable to text classification due to the discreteness of data.

To be more distinguishable between positive and negative samples but not concentrated on the nearest few samples (Zhang et al., 2021), we employ the temperature parameter τ in InfoNCE and the objective loss becomes

$$\log \frac{\exp(f_\theta(x)[y]/\tau)}{\sum\limits_{i=1}^{N} \exp(f_\theta(x_i)[y]/\tau)}. \tag{12}$$

For the effectiveness of contrastive learning, it often requires a large number of negative samples (Chen et al., 2021a). Since directly increasing N is limited to hardware memory, we instead propose to use a memory bank (He et al., 2020) to store logits with negligible computational resources. In detail, we store the logits $f_\theta(x)[y]$ of the past samples into the memory bank.

Marginal Generative Likelihood. The marginal generative likelihood can be handled in the similar way as conditional generative likelihood. Specifically,

$$\begin{aligned}\text{log} p_\theta(x) &= \log \left\{ \sum_y p_\theta(x,y) \right\}\\ &= \log \frac{\sum_y \exp(f_\theta(x)[y])}{Z_\theta},\end{aligned} \tag{13}$$

where $Z_\theta = \sum_x \sum_y \exp(f_\theta(x)[y])$. We propose to approximate the third term in (6) as

$$\mathbb{E}_{p_{\text{data}}(x,y)} [\log p_\theta(x)] \approx \log \frac{\sum_y \exp(f_\theta(x)[y])}{\sum\limits_{i=1}^{N} \sum_y \exp(f_\theta(x_i)[y])}, \tag{14}$$

where x_i is sampled from the data distribution. As the conditional generative likelihood, techniques of temperature parameter and memory bank are also employed.

4 Experiments

In this section, we conduct thorough experiments to investigate the empirical peformance of our proposed methods. We first introduce the criteria for ID calibration and OOD detection.

ID Calibration. For a well-calibrated model, the confidence estimate $\hat{p}$ of the model is expected to be comparable to true probability (accuracy): $\mathbb{P}(\hat{y} = y|\hat{p}) = \hat{p}$ (Desai and Durrett, 2020; Kong et al., 2020). The calibration error for a given confidence $p \in (0, 1)$ is defined as the followings:

$$\mathbb{E}_p = |\mathbb{P}(\hat{y}(x) = y(x)|\hat{P}(x) = p) - p|, \qquad (15)$$

where $\hat{y}(x)$ is the label predicted by the model, $y(x)$ is the true label for input x and $\hat{P}(x)$ is the output probability associated with the predicted label $\hat{y}(x)$. To evaluate the overall calibration error, we partition $(0, 1)$ into M bins of equal size and let b_m denote the set of prediction confidences which lie in the m-th bin. The expected calibration error (ECE) is calculated by weighting the difference between accuracy and confidence of each bin:

$$\begin{aligned} \text{acc}(b_m) &= \frac{1}{|b_m|} \sum_{i \in b_m} \mathbb{I}(\hat{y}_i = y). \\ \text{conf}(b_m) &= \frac{1}{|b_m|} \sum_{i \in b_m} \hat{p}_i. \qquad (16) \\ \text{ECE} &= \sum_{m=1}^{M} \frac{|b_m|}{N} |\text{acc}(b_m) - \text{conf}(b_m)|. \end{aligned}$$

OOD Detection. In general, OOD detection is a binary classification problem, where the model is required to produce a score $s_\theta(x) \in \mathbb{R}$. Usually we can set a threshold δ to detect OOD samples whose score functions are below the threshold. A well-calibrated model is expected to output higher scores for in-distribution examples than out-of-distribution examples. A widely used score function is maximum prediction probability (Hendrycks and Gimpel, 2016):

$$s_\theta(x) = \max_y p_\theta(y|x). \qquad (17)$$

Following Kong et al. (2020), we employ the empirical Normalized Bounded Area Under the Calibration Curve (NBAUCC) as the evaluation metric rather than the Area Under the Receiver-Operating curve (AUROC; Hendrycks and Gimpel, 2016) and the Area Under the Precision-Recall curve (AUPR; Elflein et al., 2021). The main reason is that we would like to use a threshold as low as possible to detect ODD samples and more details are referred to Kong et al. (2020).

Target. In our experiments, we are interested in answering the following questions:

1. Does THEM achieve better calibration compared to baselines?
2. Does THEM improve OOD detection?
3. The effect of temperature and the size of memory bank on THEM, JEM(CL) and HDGE(CL).

Datasets. We consider the eight datasets of GLUE used in He et al. (2021) to evaluate the ID calibration, since there are no out-of-distribution samples in GLUE. We use the official code[1] to acquire the development and test dataset of GLUE. Furthermore, we consider six more datasets used in Kong et al. (2020) to evaluate both ID calibration and OOD detection. Details of the datasets are in Table 4 and 5 in Appendix.

Baselines. For GLUE datasets, we compare our method against that of He et al. (2021), which is state-of-the-art EBMs on natural language understanding models. Their method is based on Residual-EBM which can work with more flexible energy functions, but the computational cost is also huge compared to our method. We also compare with other three strong baselines for calibration: finetune, Scal-bin and T-scale used in He et al. (2021). For fair comparisons, we follow the experiment settings in their work. We use Roberta as the backbone and the bins of ECE is set to 20.

For the additional six datasets, we compare our methods with nine strong baselines in Kong et al. (2020) including (1) BERT finetuning, (2) Post-calibration method: Temperature Scaling (TS; Guo et al., 2017), (3) Model ensemble: Monte Carlo Dropout (MCDP; Gal and Ghahramani, 2016), (4) Over-confident correction: Label Smoothing (LS; Müller et al., 2019), Entropy Regularized Loss (ERL; Pereyra et al., 2017), Virtual Adversarial Training (VAT; Miyato et al., 2018), and (5) Data-augmentation: Mixup (Zhang et al., 2018), Manifold-Mixup (M-Mixup; Verma et al., 2019), and Manifold-regularization (M-regularization; Kong et al., 2020). We use BERT as the backbone and the bins of ECE is set 15 just as Kong et al. (2020). Besides, we also use NBAUCC as the misclassification evaluation to make fair and

---

[1] We use the official code: `https://github.com/salesforce/ebm_calibration_nlu`

comprehensive comparisons. For these datasets, we don't compare our method with that of He et al. (2021) since it is time-consuming and needs more computational resources to finetune a noise distribution and generate negative samples for NCE training. At last, we use $\text{NBAUCC}_{0.5}$ as the evaluation metric for OOD detection. The OOD datasets often need an ID dataset for training and an OOD dataset for OOD detection evaluation. More details of datasets can be found in Appendix A.

Implementation Details. We employ ADAM (Kingma and Ba, 2014) as the optimizer for all experiments with the following parameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$, gradient clip of 1.0, and L_2 weight decay of 0.1. We search learning rate in $[1\text{e}^{-5}, 2\text{e}^{-5}, 3\text{e}^{-5}, 5\text{e}^{-5}]$ with the training epochs in $[2, 3, 5, 10]$. Our model is built with a classifier on the top of the pretrained language models including BERT (Devlin et al., 2019) and Roberta (Liu et al., 2019) using the implementation of Huggingface (Wolf et al., 2020). For contrastive learning, we set the size of memory bank N to 65536 and the temperature τ to 0.1. All experiments run 5-times and we report the average performance on test dataset. The test result is selected based on loss and accuracy on the development dataset[2]. All experiments are conducted on a single NVIDIA RTX 2080TI 12G GPU. Our implementation is based on the official codes of MoCo[3] and Manifold-regularization[4].

Results on ID Calibration. Table 1 and 2 show the accuracy (acc) and ECE results for GLUE tasks and the six additional datasets respectively, with different baseline methods. Except our proposed THEM, we also include (1) HDGE: $\log p(x|y) + \log p(y|x)$, (2) JEM: $\log p(y|x) + \log p(x)$, (3) IGEMB: $\log p(x|y)$ but trained with contrastive learning (CL) proposed in this paper. These three EBMs are trained by MCMC for images in previous literatures, while we are the first to train them by contrastive learning for NLU tasks.

From Table 1, EBMs with contrastive learning achieves significant improvements on ECE with competitive accuracy, compared to He et al. (2021). He et al. (2021) redefines energy function based on Residual-EBM and estimates parameters using NCE with a finetuned GPT (Radford et al.) on dataset as noise distribution. Not only does it need more computing resources to finetune a GPT model for each dataset, but also the quantity and quality of negative samples generated by noise distribution have big impacts for accurate parameters estimation using NCE (He et al., 2021; Gutmann and Hyvärinen, 2010). In contrast, our model achieves better results with negligible computational resource compared to standard finetuning.

From Table 2, our method achieves the best ECE on six multiclass datasets on various domains. Compared to M-regularization (Kong et al., 2020) which is specifically designed to prevent overconfident predictions for both in-distribution and out-of-distribution, our framework without an explicit calibration mechanism achieves the best ECE, demonstrating the effectiveness of EBMs trained with contrastive learning paradigm. On average, the result of the proposed THEM is very close to the best one in terms of ECE.

Results on OOD Detection. In general, OOD detection is a binary classification problem, where the model is required to produce a score for a data point to detect whether it is an ID or OOD sample. Here we use equation (17) as score function and $\text{NBAUCC}_{0.5}$ as evaluation metric. Table 3 summarizes the $\text{NBAUCC}_{0.5}$ for misclassification detection and OOD detection. It can be seen that compared with all baselines, especially the strong baseline M-regularization (Kong et al., 2020), our method achieves the best misclassification on all data sets with significant improvements. In terms of OOD detection, our results averaged on six datasets are comparable to the performance of M-regularization and are superior to other baselines on all datasets except M-regularization on Yahoo. These results shows that THEM provides a simple yet effective way to improve OOD detection.

Analysis of Generative Density. From Table 1 and Table 2, the generative density including marginal data density $p_\theta(x)$ and class conditional data density $p_\theta(x|y)$ are mainly contributed to the improvements of ID calibration and OOD detection compared to standard finetuning and previous calibration methods. However, different generative terms may have different impacts on final performance. In most NLU tasks, JEM(CL) achieves better ID and OOD calibration performance compared to HDGE(CL) which is different from the experimental results observed in Liu and Abbeel (2020) on

---

[2] Following He et al. (2021), we don't use ECE as the metric to select the best model for evaluation

[3] `https://github.com/facebookresearch/moco`

[4] `https://github.com/Lingkai-Kong/Calibrated-BERT-Fine-Tuning`

Table 1: Test-set accuracy and ECE results for different methods on GLUE tasks. The leading zeros are omitted to save space. Note that the hyperparameters of T-Scal and Scal-bin are searched on the development dataset and applied to test dataset. The average value is compute on all nine test sets. For each task, the method that achieves best calibration are shown in bold.

Method	SST-2		MNLI		MNLI(mm)		QNLI		QQP		MRPC		COLA		RTE		WNLI		AVG	
	acc	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**
Baseline (He et al., 2021)																				
finetune	.942	.050	.876	.067	.872	.068	.929	.043	.904	.034	.862	.133	.539	.182	.724	.279	.571	.058	.802	.102
Scal-bin(dev)	.944	**.019**	.876	.030	.870	.032	.931	.021	.905	.021	.862	.062	.557	**.048**	.731	**.042**	.542	.189	.802	.052
T-Scale(dev)	.942	.037	.876	.024	.872	.026	.929	.018	.904	.026	.862	.126	.539	.109	.724	.235	.571	.046	.802	.072
Residual-EBM-NCE (He et al., 2021)																				
ebm-scalar	.942	.033	.871	.038	.871	.047	.927	**.016**	.899	.034	.862	.098	.540	.150	.753	.207	.542	**.033**	.801	.073
ebm-hidden	.956	.032	.869	.032	.868	.044	.923	**.016**	.900	.033	.867	.099	.545	.131	.797	.148	.542	.036	.807	.063
ebm-s-hidden	.947	.038	.875	**.027**	.872	.031	.930	**.016**	.900	.032	.862	.089	.563	.133	.811	.182	.571	.073	.815	.069
Ours																				
HDGE(CL)	.938	.036	.870	.040	.864	.049	.927	.024	.908	.023	.862	.056	.539	.101	.753	.069	.571	.051	.803	.048
JEM(CL)	.926	.043	.872	.033	.868	**.023**	.927	.021	.907	**.009**	.877	.060	.562	.107	.753	.073	.571	.057	.806	**.047**
IGEBM(CL)	.922	.054	.868	.124	.869	.125	.931	.065	.910	.087	.867	**.029**	.549	.060	.789	.052	.571	.044	.808	.071
THEM	.922	.035	.867	.043	.866	.043	.928	.028	.910	.019	.872	.062	.551	.085	.724	.082	.571	.050	.801	**.049**

Table 2: ECE and accuracy (in percentage) on test set for different methods on six multiclass datasets listed in Table 5. We report the average performance of 5 random initializations. For each task, the method that achieves best calibration are shown in bold.

Method	$20NG_{15}$		20NG		WOS_{100}		WOS		$Yahoo_{8}$		Yahoo		AVG	
	acc	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**	**acc**	**ECE**
baseline (Kong et al., 2020)														
BERT	87.42	9.24	84.55	11.61	81.94	6.81	79.40	6.74	73.58	10.11	71.89	10.54	79.79	9.17
TS	87.42	4.42	84.55	8.17	81.94	3.63	79.40	4.43	73.58	5.18	71.89	4.24	79.79	5.01
MCDP	87.45	6.88	84.55	9.17	82.09	4.00	79.67	3.55	73.67	6.54	71.99	6.72	79.90	6.14
LS	87.54	4.35	85.02	6.15	81.95	4.35	79.47	4.67	73.66	4.89	71.54	3.61	79.86	4.67
ERL	87.67	7.16	84.83	6.10	81.96	3.74	79.48	3.35	73.63	3.42	72.01	2.96	79.92	4.45
VAT	87.61	9.07	85.20	11.28	81.65	7.27	79.71	6.76	73.71	10.96	72.08	7.92	79.99	8.87
Mixup	87.49	5.98	84.86	9.02	81.97	4.72	79.51	4.21	73.88	4.60	71.82	5.18	79.92	5.61
M-Mixup	87.40	5.04	84.45	7.78	81.77	6.48	79.57	6.68	72.03	7.01	72.03	6.07	79.54	6.51
M-regularization	87.44	3.69	84.53	4.43	81.59	3.24	79.06	**3.04**	73.71	3.03	72.17	3.42	79.75	3.47
Ours														
HDGE(CL)	87.34	4.71	84.47	7.76	81.14	4.00	78.68	4.12	73.53	4.02	71.62	5.97	79.46	5.09
JEM(CL)	87.98	3.10	84.81	**2.17**	81.80	3.47	78.74	3.27	73.72	2.17	72.60	**1.64**	79.86	**2.58**
IGEBM(CL)	88.35	2.47	84.06	3.87	81.51	11.72	78.46	13.73	73.01	4.00	70.82	2.01	79.36	6.30
THEM	88.35	**2.09**	84.99	3.91	81.05	**3.01**	78.72	3.19	73.80	**1.55**	72.00	2.19	79.81	**2.65**

computer vision tasks. The main reason may lie on the estimation methods that it is more stable and effective to approximate the log-likelihood with contrastive loss compared to MCMC or score-matching for calibration, when the generation ability is not under consideration. While our model achieves comparable ID calibration performance across various datasets on average and better OOD detection.

The hyper-parameters study of InfoNCE on ID calibration and OOD detection. To study the effect of the hyper-parameters including the sample size, and temperature of InfoNCE for training THEM, JEM(CL) and HDGE(CL), we conduct numerous experiments on **20NG**, **Yahoo** and **WOS** dataset. Due to the limited computational resources, we set temperature to 0.1 and vary the size of memory bank from 128 to 65536 to study the effect of the memory bank size on ECE and OOD. Similarly, we set the memory bank size to 65536 and vary the temperature from 0.01 to 0.2 to study the effect of the temperature on ECE and OOD.

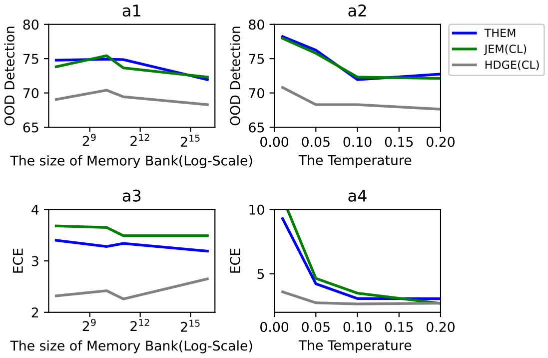


Figure 1: The effect of the memory bank size and temperature for ECE and OOD on **WOS** dataset.

Table 3: NBAUCC$_{0.5}$ on misclassification detection and OOD detection (in percentage) for different methods on six multiclass datasets listed in Table 5. We report the average performance of 5 random initializations.

Data (OOD)	Misclassification(↑)							OOD Detection(↑)						
	20NG$_{15}$	20NG	WOS$_{100}$	WOS	Yahoo$_8$	Yahoo	AVG	20NG$_{15}$ 20NG$_5$	20NG SST-2	WOS$_{100}$ WOS$_{34}$	WOS AGnews	Yhaoo$_8$ Yhaoo$_2$	Yahoo Yelp	AVG
baseline (Kong et al., 2020)														
BERT	2.30	2.86	16.53	20.52	7.47	8.43	9.68	2.66	21.65	23.12	49.84	8.35	13.88	19.91
TS	6.08	5.74	21.20	23.76	10.48	12.74	13.33	6.62	32.64	28.12	53.32	11.55	20.27	25.42
MCDP	4.37	5.28	20.44	24.16	10.12	10.75	12.52	3.99	25.10	27.28	53.52	9.98	15.93	22.63
LS	4.72	6.75	20.37	23.56	11.19	16.15	13.79	5.70	41.08	27.12	58.48	12.02	19.81	27.36
ERL	8.54	10.35	20.49	25.13	12.89	15.47	15.47	8.78	47.00	27.73	56.67	13.78	23.47	29.57
VAT	2.52	3.36	18.70	19.96	6.54	10.37	10.24	2.96	29.62	23.41	54.60	7.42	17.65	22.61
Mixup	4.99	4.51	20.65	24.80	10.75	11.29	12.83	5.86	31.84	26.77	58.02	11.62	19.84	25.65
M-mixup	2.16	3.16	16.94	19.39	9.09	11.79	10.42	2.36	26.08	24.08	51.39	10.08	22.41	22.73
M-regularization	9.10	10.76	26.93	30.80	14.34	17.88	18.30	9.69	63.92	35.60	71.13	14.94	**29.40**	**37.44**
Ours														
HDGE(CL)	7.99	6.68	25.25	27.82	12.31	14.72	15.79	7.42	57.09	34.81	68.29	11.55	20.62	33.29
JEM(CL)	**15.31**	14.88	25.55	32.97	**16.25**	16.16	20.18	12.23	61.99	34.70	**72.31**	16.17	19.80	36.20
IGEBM(CL)	13.87	**15.34**	14.37	15.64	14.52	**21.83**	15.92	**14.47**	**64.75**	23.67	57.94	**17.93**	24.22	33.83
THEM	11.56	11.11	**31.82**	**33.02**	16.11	18.25	**20.31**	9.36	62.86	**40.16**	71.94	17.28	19.73	**36.88**

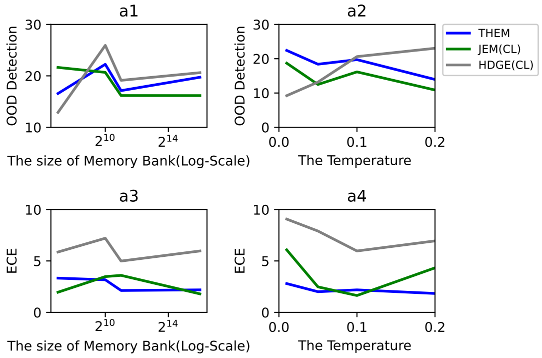


Figure 2: The effect of the memory bank size and temperature for ECE and OOD on **Yahoo** dataset.

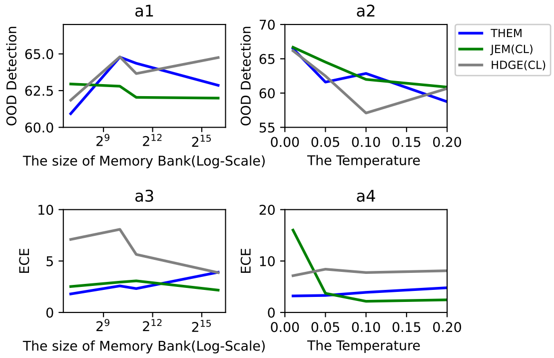


Figure 3: The effect of the memory bank size and temperature for ECE and OOD on **20NG** dataset.

From the trends of memory bank size and temperature on WOS(134-class) in Fig 1, HDGE performs better than THEM and JEM(CL) in terms of ECE. However, it performs significantly worse than THEM and JEM(CL) in terms of OOD. On the other hand, they are all stable in terms of ECE and OOD from the trend of memory bank size.

However, for Yahoo(10-class) in Fig 2 and 20NG(20-class) in Fig 3, from the trend of memory bank size, THEM and HDGE are superior to JEM(CL) in terms of OOD. THEM and JEM(CL) perform better than HDGE in terms of ECE. From the trend of temperature, THEM performs better than JEM(CL) and HDGE in all evaluation metrics. In general, THEM is more stable in terms of ECE from the trend of temperature and memory bank size.

5 Conclusion

In our work, we propose a triple-hybrid EBM with combination of classifier, conditional generative model and marginal generative model into a unified framework called THEM. To train EBMs effectively and efficiently, we leaverage contrastive learning to approximate the log-likelihood of EBMs with negligible computational resources. Extensive experiments demonstrates that our model outperforms the state-of-art methods in terms of ID calibration and OOD detection with competitive accuracy. We further apply contrastive learning to JEM and IGEBM without considering the generation ability to obtain JEM(CL) and IGEBM(CL) respectively. Compared to JEM(CL) and HDGE(CL), our model is more robust to the hyper-parameters of contrastive learning including the temperature and size of memory bank in terms of ID calibration and OOD detection.

6 Limitations

In our work, our model is derived from the perspective of EBMs. However, it lacks of generation ability due to the approximation of log-likelihoods with contrastrive learning which may limit the power of generative modeling such as data augmentation (Grathwohl et al., 2019). We will explore MCMC-based methods such as (Eikema et al., 2021; Qin et al., 2022) to train THEM to take advantage of generative modeling. As for OOD detection, we only use Maximum Prediction Probability. But many other OOD scoring functions are proposed from the perpective of EBMs (Ouyang et al., 2021; Zhou et al., 2021; Liu et al., 2020; Elflein et al., 2021; Grathwohl et al., 2019). And it may be explored in future works to study the OOD performance with different scoring functions.

7 Acknowledgements

This research of Zhang is supported by the National Natural Science Foundation of China (NSFC grant nos. 12101241).

References

Pavel Blinov, Manvel Avetisian, Vladimir Kokh, Dmitry Umerenkov, and Alexander Tuzhilin. 2020. Predicting clinical diagnosis from patients electronic health records using bert-based neural networks. In *International Conference on Artificial Intelligence in Medicine*, pages 111–121. Springer.

Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Aaai*, volume 2, pages 830–835.

Junya Chen, Zhe Gan, Xuan Li, Qing Guo, Liqun Chen, Shuyang Gao, Tagyoung Chung, Yi Xu, Belinda Zeng, Wenlian Lu, et al. 2021a. Simpler, faster, stronger: Breaking the log-k curse on contrastive learners with flatnce. *arXiv preprint arXiv:2107.01152*.

Si-An Chen, Chun-Liang Li, and Hsuan-Tien Lin. 2021b. A unified view of cgans with and without classifiers. *Advances in Neural Information Processing Systems*, 34.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2019. Residual energy-based models for text generation. In *International Conference on Learning Representations*.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302.

J. Devlin, M. W. Chang, K. Lee, and K Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Yilun Du and Igor Mordatch. 2019. Implicit generation and modeling with energy based models. *Advances in Neural Information Processing Systems*, 32.

David Duvenaud, Jacob Kelly, Kevin Swersky, Milad Hashemi, Mohammad Norouzi, and Will Grathwohl. 2021. No mcmc for me: Amortized samplers for fast and stable training of energy-based models.

B. Eikema, G. Kruszewski, H. Elsahar, and M. Dymetman. 2021. Sampling from discrete energy-based models with quality/efficiency trade-offs.

Sven Elflein, Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. 2021. On out-of-distribution detection with energy-based models. *arXiv preprint arXiv:2107.08785*.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR.

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. 2019. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Tianxing He, Bryan McCann, Caiming Xiong, and Ehsan Hosseini-Asl. 2021. Joint energy-based model training for better calibrated natural language understanding models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1754–1761.

Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*.

Taehee Jung, Dongyeop Kang, Hua Cheng, Lucas Mentch, and Thomas Schaaf. 2020. Posterior calibrated training on sentence classification tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2723–2730.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Shivesh Khaitan, Qin Lin, and John M Dolan. 2021. Safe planning and control under uncertainty for self-driving. *IEEE Transactions on Vehicular Technology*, 70(10):9826–9837.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, and Chao Zhang. 2020. Calibrated language model fine-tuning for in-and out-of-distribution data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1326–1340.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371. IEEE.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. 2006. A tutorial on energy-based learning. *Predicting structured data*, 1(0).

Fei Li, Yonghao Jin, Weisong Liu, Bhanu Pratap Singh Rawat, Pengshan Cai, Hong Yu, et al. 2019. Fine-tuning bidirectional encoder representations from transformers (bert)–based models on large-scale electronic health record notes: an empirical study. *JMIR medical informatics*, 7(3):e14830.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022. A survey of pretrained language models based text generation. *arXiv preprint arXiv:2201.05273*.

Linfeng Li, Peng Wang, Jun Yan, Yao Wang, Simin Li, Jinpeng Jiang, Zhe Sun, Buzhou Tang, Tsung-Hui Chang, Shenghui Wang, et al. 2020. Real-world data medical knowledge graph: construction and applications. *Artificial intelligence in medicine*, 103:101817.

Hao Liu and Pieter Abbeel. 2020. Hybrid discriminative-generative training via contrastive learning. *arXiv preprint arXiv:2007.09070*.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems*, 32.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Yawen Ouyang, Jiasheng Ye, Yu Chen, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2021. Energy-based unknown intent detection with data manipulation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2852–2861.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*.

L. Qin, S. Welleck, D. Khashabi, and Y. Choi. 2022. Cold decoding: Energy-based constrained text generation with langevin dynamics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Amir Rahimi, Amirreza Shaban, Ching-An Cheng, Richard Hartley, and Byron Boots. 2020. Intra order-preserving functions for calibration of multi-class neural networks. *Advances in Neural Information Processing Systems*, 33:13456–13467.

Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. 2021. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):1–13.

Nils Rethmeier and Isabelle Augenstein. 2021. A primer on contrastive pretraining in language processing: Methods, lessons learned and perspectives. *arXiv preprint arXiv:2102.12982*.

Sarah Sarabadani. 2019. Detection of adverse drug reaction mentions in tweets using elmo. In *Proceedings of the Fourth Social Media Mining for Health Applications (# SMM4H) Workshop & Shared Task*, pages 120–122.

Richard Socher, Yoshua Bengio, and Christopher D Manning. 2012. Deep learning for nlp (without magic). In *Tutorial Abstracts of ACL 2012*, pages 5–5.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. 2020. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pages 574–584. PMLR.

Ting Sun and Miklos A Vasarhalyi. 2021. Predicting credit card delinquencies: An application of deep neural networks. In *Handbook of Financial Econometrics, Mathematics, Statistics, and Machine Learning*, pages 4349–4381. World Scientific.

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. 2019. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Advances in Neural Information Processing Systems*, 32.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3070–3079.

Max Welling and Yee W Teh. 2011. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Oliver Zhang, Mike Wu, Jasmine Bayrooti, and Noah Goodman. 2021. Temperature as uncertainty in contrastive learning. *arXiv preprint arXiv:2110.04403*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Wenxuan Zhou, Fangyu Liu, and Muhao Chen. 2021. Contrastive out-of-distribution detection for pretrained transformers. *arXiv preprint arXiv:2104.08812*.

A Dataset

The details of dataset for evaluation of in-distribution ECE and out-of-distribution detection.

Table 4: The detail information about GLUE

dataset	task	labels	train/dev/test
RTE	Similarity	2	2.5k/0.14k/0.14k
CoLA	Grammatical	2	8.5k/0.51k/0.51k
WNLI	Entailment	2	3.1k/0.03k/0.03k
MRPC	Paraphrase	2	3.7k/0.20k/0.20k
QNLI	Entailment	2	108k/2.5k/2.5k
MNLI-m	Entailment	3	393k/4.8k/4.8k
MNLI-mm	Entailment	3	393k/4.4k/4.4k
QQP	Paraphrase	2	364k/20k/20k
SST-2	Classification	2	67k/0.43k/0.43k

Table 5: The detail information about six multiclass-datasets

in-distribution out-of-distribution	labels	train/dev/test
$20NG_{15}$	15	7k/1.7k/5.8k
$20NG_5$	5	-/-/1.7k
20NG	20	9k /2.2k/7.5k
SST-2	2	-/-/1.8k
WOS_{100}	100	16k/4.1k/14k
WOS_{34}	34	-/-/ 4.8k
WOS	134	22k/5.6k /18k
AGnews	4	- /-/ 7.6k
$Yahoo_8$	8	16k/4k/48k
$Yahoo_2$	2	-/-/12k
Yahoo	10	20k/5k/60k
Yelp	2	-/-/38k

1. 20NG[5]. The 20 Newsgroups dataset (20NG) contains news articles with 20 categories. We use Stanford Sentiment Treebank (SST-2) (Socher et al., 2012) as the OOD data.
2. 20NG_{15}. We take the first 15 categories of 20NG as the in-distribution data and the other 5 categories (20NG_5) as the OOD data.

---

[5] We use the 20 Newsgroups dataset from: `http://qwone.com/~jason/20Newsgroups/`

3. WOS (Kowsari et al., 2017). Web of Science (WOS) dataset contains 134 categories of scientific articles. We use AGnews (Zhang et al., 2015) as the OOD data.
4. WOS_{100}. We use the first 100 classes of WOS as the in-distribution data and the other 34 classes (WOS_{34}) as the OOD data.
5. Yahoo (Chang et al., 2008). This dataset contains 10 categories posted to 'Yahoo!Answers' of questions. We randomly draw 2000 from 140,000 samples for each category as the training set. We use Yelp (Zhang et al., 2015) as the OOD data.
6. Yahoo_8. We use the first 8 classes of Yahoo as the in-distribution data and the other 2 classes (Yahoo_2) as the OOD data.

A weakly supervised textual entailment approach to zero-shot text classification

Marc Pàmies[1*] **Joan Llop**[1*]
Francesco Multari[2] Nicolau Duran-Silva[2] César Parra-Rojas[2]
Aitor Gonzalez-Agirre[1] Francesco Alessandro Massucci[2] Marta Villegas[1]

[1]Barcelona Supercomputing Center [2]SIRIS Academic

Abstract

Zero-shot text classification is a widely studied task that deals with a lack of annotated data. The most common approach is to reformulate it as a textual entailment problem, enabling classification into unseen classes. This work explores an effective approach that trains on a weakly supervised dataset generated from traditional classification data. We empirically study the relation between the performance of the entailment task, which is used as a proxy, and the target zero-shot text classification task. Our findings reveal that there is no linear correlation between both tasks, to the extent that it can be detrimental to lengthen the fine-tuning process even when the model is still learning, and propose a straightforward method to stop training on time. As a proof of concept, we introduce a domain-specific zero-shot text classifier that was trained on Microsoft Academic Graph data. The model, called SCIroShot, achieves state-of-the-art performance in the scientific domain and competitive results in other areas. Both the model and evaluation benchmark are publicly available on HuggingFace[1] and GitHub[2].

1 Introduction

Ever since the first BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) models were introduced to the world, the Transformer (Vaswani et al., 2017) has become the dominant architecture in the Natural Language Processing (NLP) field. As a consequence, in the years that followed, the pretrain-then-finetune paradigm (Howard and Ruder, 2018) has been widely adopted to progressively push the state-of-the-art in a wide variety of downstream tasks and languages (Nozza et al., 2020).

Even though the current training regime is far from being environmentally friendly due to the computational cost of pre-training (Patterson et al., 2021; Strubell et al., 2019), transfer learning removes the need of having to train a new model from scratch for each application. However, the fine-tuning of models for every single task is expensive both in terms of time and money as it is always preceded by a labor-intensive data labelling process (Wang et al., 2021a). In order to overcome this issue, as well as the fact that real world data can be scarce in many scenarios, the field has started to shift towards techniques that require smaller amounts of labelled examples or even none at all (Wang et al., 2021b; Schick and Schütze, 2021; Radford et al., 2019; Brown et al., 2020).

In particular, in the scientific domain, the growing amount of publications in an ever-increasing number of fields makes the classification task very challenging for neural language models (Larsen and Von Ins, 2010; Bornmann et al., 2021). The impossibility to predict the emergence of new fields of study and the high cost associated with the creation of new datasets (which often requires domain experts) generate a need for systems that are capable of adapting to new situations. Furthermore, the complexity and technicality of scientific language makes general-domain models perform poorly in comparison to domain-specific ones (Lee et al., 2020; Cohan et al., 2020).

This work addresses these problems by training an entailment-based zero-shot classifier for scientific text. Instead of using a general domain dataset such as the popular XNLI (Conneau et al., 2018) or MNLI (Williams et al., 2018), a textual entailment dataset of scientific documents was built from scratch in a weakly supervised manner. By training a vanilla model on the entailment task, it is then able to classify documents into unseen classes with a high degree of success. As a by-product of this, the study of the relation between the training and target tasks led to intriguing questions about the strengths and limitations of the entailment approach to Zero-Shot Text Classification (ZSTC).

---

[*] Equal contribution.

[1] `https://huggingface.co/BSC-LT/sciroshot`

[2] `https://github.com/bsc-langtech/sciroshot`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 286–296
May 2-6, 2023

2 Related Work

Zero-Shot Learning is a widely studied problem in Machine Learning that consists in completing a task for which no training examples were provided. Although the term became initially popular in computer vision (Lampert et al., 2009; Xian et al., 2018), it soon made the leap into NLP with an early paper (Chang et al., 2008) that presented a classifier capable of interpreting the Explicit Semantic Analysis (ESA) representations of documents and labels from a semantic point of view, using Wikipedia as a source of world knowledge.

This highlighted the importance of capturing the labels' semantics in their representations, unlike in conventional text classification where labels are mapped to meaningless indices. This became easier to capture with the arrival of word embeddings, and can be enhanced by simply replacing their name with a short description (Song and Roth, 2014).

In subsequent research, zero-shot tasks were occasionally tackled from another problem's perspective (Levy et al., 2017; Obamuyide and Vlachos, 2018), which implies training a model on a task for which annotated data is available and performing inference on a different one. Recognizing Textual Entailment (RTE) (Dagan et al., 2005) is arguably the most versatile task because of its generality, which is why it is commonly used to model other downstream tasks (Wang et al., 2021b).

The most popular approach at the time of writing is to reformulate ZSTC as a textual entailment problem as proposed by Yin et al. (2019), aiming to imitate the way humans would address this Natural Language Understanding task. The underlying idea is that a model trained on the entailment task should be able to perform classification of unseen classes by computing the entailment score between the input text, which acts as a premise, and candidate labels conveniently converted into hypotheses.

Later research showed that the Next Sentence Prediction (NSP) objective for sentence pair classification can also be used as a strong baseline for ZSTC, since competitive results where obtained using raw BERT models that were not fine-tuned on any Natural Language Inference (NLI) data (Ma et al., 2021). Even though entailment-based zero-shot text classifiers have shown to have certain limitations like a high instability or an excessive reliance on spurious lexical patterns, the current literature offers no better alternatives for a problem that is still far from being mastered by machines.

3 Creation of weakly supervised data

This section describes the methodology followed to transform a text classification dataset into entailment data that could potentially be used to fine-tune a domain-specific (or not) zero-shot text classifier. This approach takes advantage of the fact that there are plenty of publicly accessible labelled examples for classification while there is not so much available for entailment tasks, most likely due to the difficulty of producing this type of data.

In this work, a weakly supervised NLI dataset was constructed using Microsoft Academic Graph[3] (MAG) data as a starting point. To do so, all labels were converted to natural language sentences that would serve as hypotheses. Therefore, the generated training examples consist of pairs of sequences (premise and hypothesis) that are delimited by EOS tokens, where the first part of the text contains the abstract section from a scientific publication (premise) and the second part is an artificially generated sentence that somehow embeds the class label of the scientific text (hypothesis). Table 1 shows this idea in a simplified form.

Input Sequence	Label
<s>Text1</s></s>This example is X</s>	1
<s>Text1</s></s>This example is Y</s>	0
<s>Text2</s></s>This example is Z</s>	1

Table 1: Format of the training examples. The label is 1 if the premise entails the hypothesis and 0 otherwise.

Once the classification data has been turned into the entailment format, any model can be fine-tuned by predicting whether the premise of each input sequence entails the corresponding hypothesis or not. Note however that this is rather an adaptation of the RTE task, since the second sentence does not really "entail" the first in positive examples. It is basically like performing a text classification task where labels are not converted into numeric indices, ensuring that their semantic content is preserved.

Figure 1 illustrates how a ZSTC model would operate both during fine-tuning (left) and at inference time (right), providing a full picture of the methodology followed in this work. We basically adopt the approach introduced by Yin et al. (2019) but going one step further by fine-tuning on a weakly supervised domain-specific dataset instead of using already existing general-domain NLI data.

---

[3] `https://academic.microsoft.com/`

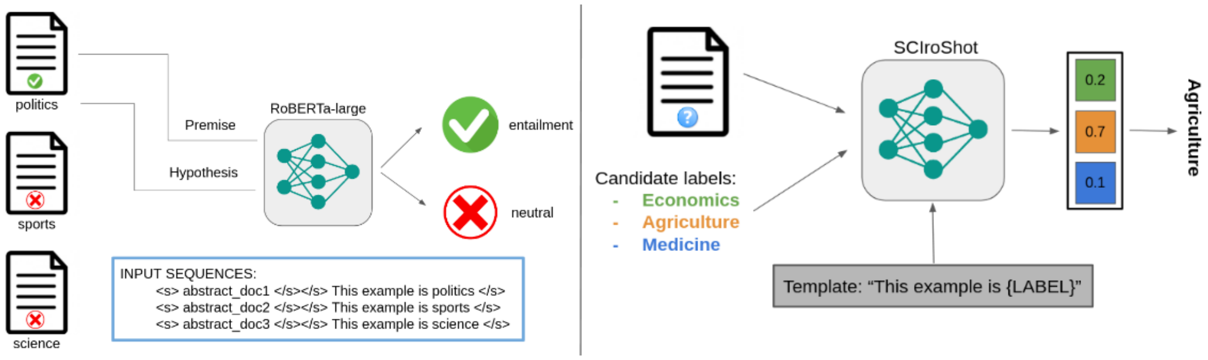


Figure 1: Overview of the entailment approach to ZSTC. On the left hand side it can be seen how the model is fine-tuned on a two-class textual entailment task, by providing the input text as a premise and the label name embedded in a natural language sentence that represents the hypothesis. On the right, the trained model classifies documents into unseen classes by computing the entailment score between the input text and each candidate label.

4 Data

4.1 Training Dataset

As mentioned in Section 3, our training dataset builds on top of scientific-domain annotated data from Microsoft Academic Graph (Sinha et al., 2015). This database consists of a heterogeneous graph with billions of records from both scientific publications and patents, in addition to metadata information such as the authors, institutions, journals, conferences and their citation relationships. The documents are organized in a hierarchical structure composed of hundreds of thousands of scientific concepts, creating a six-level hierarchy with a subsumption-based model (Shen et al., 2018), although the two top-most levels are manually curated to guarantee accuracy. As an example, the 0-level field of study (FoS) in the MAG taxonomy covers the following 19 scientific concepts: {*Art, Biology, Business, Chemistry, Computer Science, Economics, Engineering, Environmental Science, Geography, Geology, History, Materials Science, Mathematics, Medicine, Philosophy, Physics, Political Science, Psychology, Sociology*}.

Dataset	Labels	Examples
train	240	2,104,493
dev$_{\text{seen}}$	240	233,833
dev$_{\text{unseen}}$	52	5,200
Total	292	2,343,526

Table 2: Number of labels and examples in the train and development sets of our dataset. Note that the 240 labels from the train and dev$_{\text{seen}}$ set are the same, while the 52 labels from the dev$_{\text{unseen}}$ set were purposely kept aside for *label-fully-unseen* setups.

Due to the descriptive broadness of the 0-level MAG taxonomy, we have created our training corpus focusing exclusively the 1-level MAG taxonomy, which is composed of 292 FoS classes. The higher granularity of this level provides more descriptive information in the form of narrower scientific concepts, such as "*Computational biology*", "*Transport engineering*" or "*Civil engineering*".

Using the relationship between scientific texts and their matching concepts in the 1-level MAG taxonomy we are able to generate the premise-hypothesis pairs corresponding to the entailment label. Conversely, we generate the pairs for the neutral label by removing the actual relationship between the texts and their scientific concepts and creating a virtual relationship with those to which they are not matched (see Table 3).

Input Sequence	Label
One plus one is two.	Maths
Cancer is a disease.	Health
One plus one is two. This text is Maths.	entails
One plus one is two. This text is Health.	neutral
Cancer is a disease. This text is Maths.	neutral
Cancer is a disease. This text is Health.	entails

Table 3: Toy example of the initial classification dataset (top) and the adapted entailment dataset (bottom).

For each of the 292 classes, a random sample of scientific articles with a publication year between 2000 and 2021 was extracted with their respective titles and abstracts in English. We have collected a maximum of 5k and 10k positive and neutral textual entailment samples, respectively, for each of the possible 1-level FoS classes. In total, SCIroshot has been fine-tuned using 919k documents with a total of 465M words. In order to perform experiments in *label-fully-unseen* setups, the examples associated to 52 labels were kept aside. For computational reasons, the size of the development set was reduced considerably to the point that it became a fully balanced set of 100 examples per class. The number of labels and examples of each set are summarized in Table 2.

4.2 Evaluation Datasets

We evaluate the performance of our models on a collection of disciplinary-labeled textual datasets. For the in-domain evaluation, we gathered cross-disciplinary and domain-specific datasets of scientific publications. For the out-of-domain case, we use the three datasets from the benchmark provided by Yin et al. (2019), which study 3 aspects of ZSTC: topic categorization (Yahoo! Answers), emotion detection (UnifyEmotion) and situation frame detection (Situation Typing). Table 4 provides an overview of the number of examples and labels for each dataset.

Dataset	Labels	Examples
arXiv	11	3,838
SciDocs-MeSH	11	16,433
SciDocs-MAG	19	17,501
Konstanz	24	10,000
Elsevier	26	14,738
PubMed	109	5,000
Yahoo! Answers	10	60,000
UnifyEmotion	10	15,689
Situation Typing	12	3,311

Table 4: Statistics of each dataset from the scientific-domain (top) and general-domain (bottom) benchmarks.

4.2.1 Scientific-domain datasets

arXiv (**He et al., 2019**). 11-label dataset of papers from the arXiv repository. The labels are a set of sub-categories within the branches of Computer Science and Mathematics. While the original dataset contains the full publication texts, we only gathered titles and abstracts from the 3,838 publications for which a DOI was available in the API[4].

SciDocs-MeSH (**Cohan et al., 2020**). Over 16k papers from the medical domain. Each paper is assigned one of 11 high-level disease classes derived from the MeSH vocabulary (Lipscomb, 2000).

SciDocs-MAG (**Cohan et al., 2020**). More than 17k cross-disciplinary publications labelled using the 0-level MAG taxonomy (Wang et al., 2020). None of the labels are included in our training data.

Konstanz[5]. 10k journal articles produced by researchers at the University of Konstanz, extracted from the Konstanz Online Publication System (KOPS). Publications from this open-access repository are manually labelled by the research staff with a category taken from the DDC taxonomy (Dewey, 1876), which unfolds into more than 30 classes describing different scientific domains. We only consider English journal articles labelled within a reduced set of 24 categories.

Elsevier (**Kershaw and Koeling, 2020**). Cross-disciplinary corpus of 14.7k open access articles from Elsevier's journals. The document labels are given by their ASJC Subject Classification scheme, which links publication venues (and, transitively, each single publication) to 27 scientific subject domains. We removed all publications labelled with more than 1 subject area. Publications annotated with the non-informative "*Multidisciplinary*" label were removed as well.

PubMed[6]. We collected 5k publications labelled with a manually-selected subset of 109 MeSH terms within the *Disciplines and Occupations* and *Technology, Industry, and Agriculture* branches of the MeSH taxonomy. The chosen categories are general-domain and well-known concepts out of specific medical terminology, as most MeSH terms.

4.2.2 Out-of-domain datasets

Yahoo! Answers (**Zhang et al., 2015**). Topic categorisation dataset with questions and their corresponding best answer in *Yahoo! Answers*. We only use the test set, which consists of 60k examples that belong to exactly one of the 10 largest main categories in the website.

UnifyEmotion (**Oberländer and Klinger, 2018**). Emotion detection dataset with texts from a variety of sources (tweets, emotional events, tales, and artificial sentences) classified into 9 emotions and "none" when no emotion fits the case. We use the modified version from Yin et al. (2019), which removes all multi-label instances.

Situation Typing (**Mayhew et al., 2019**). Multi-label event-type classification dataset of 11 classes, designed for low-resource situation detection.

---

[4] `https://arxiv.org/help/api/`

[5] `https://kops.uni-konstanz.de/`

[6] `https://pubmed.ncbi.nlm.nih.gov/`

5 Experiments

5.1 Design choice

The newly-created scientific dataset from Section 4.1 was used to fine-tune a 355M parameters RoBERTa (Liu et al., 2020) and a 400M parameters BART (Lewis et al., 2020) models, in an attempt to determine which architecture (i.e. encoder or encoder-decoder) is best suited for the task at hand. As already noted in section 4.1, 52 labels from the training data were kept apart so that they could be used as a development set of fully-unseen classes. For a given input text, the entailment score with each candidate label has to be computed by the model. The final prediction will be the highest scoring class in a single-label classification setup, or the *N* classes above a certain threshold in a multi-label scenario. Table 5 shows the accuracy score of the last checkpoints evaluated in RTE's dev_{seen} set as well as the ZSTC $\text{dev}_{\text{unseen}}$ set.

Model	**RTE**	**ZSTC**
RoBERTa-large	98.00	48.78
BART-large	98.30	45.69

Table 5: Accuracy scores of our two models in the Recognizing Textual Entailment (RTE) and Zero-Shot Text Classification (ZSTC) tasks.

Even though both models achieved a similar accuracy in the entailment task, we could not help noticing that the best performing model on RTE (even if only by a small margin) was doing worse on ZSTC by three full points. This raised a concern as to whether our fine-tuning task, which at the end is no more than an adaptation of the real RTE, was positively correlated with the target task of ZSTC or not.

5.2 Correlation between the RTE-ZSTC tasks

In order to verify the correlation between both tasks, we conducted an exhaustive evaluation of all checkpoints using the 52-labels $\text{dev}_{\text{unseen}}$ set.

As it can be observed in Figure 2, after a certain point the performance in the ZSTC task begins to gradually worsen while on RTE it is still getting better at a slow but steady pace. This means that somehow, as the training progresses, the model forgets the meaning of the unseen labels, with the exception of the initial checkpoints where an exponential growth is experienced simultaneously in both tasks. We can observe a peak ZSTC performance when the model has an evaluation RTE score of roughly 0.96. Despite the high variability, it is clear that from that point onwards the zero-shot capacities of the model decrease. To mitigate this effect we propose an early stopping technique.

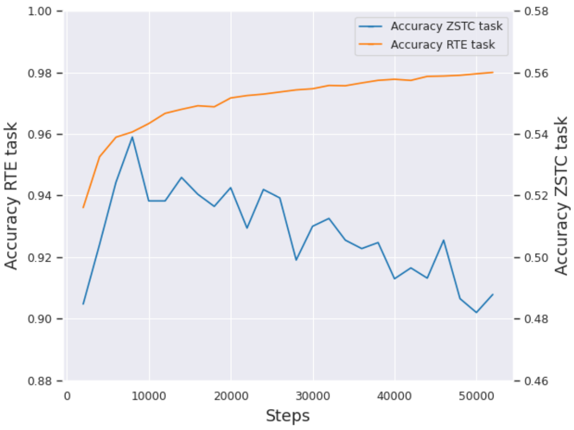


Figure 2: Accuracy scores obtained by the RoBERTa-large checkpoints. Each y-axis represents a different range of accuracies for better visualization.

5.3 Early stopping

We concluded that, at training time, the validation of the model should be done on the target task rather than the training task. This way the training process can be interrupted as soon as the model stops improving on ZSTC, something that we expect to happen at an earlier stage. We propose to evaluate each checkpoint on the subset of 52 unseen classes described in Section 4.1 and use early stopping with a patience of 10.

Model	**RTE**	**ZSTC**
$\text{RoBERTa-large}_{\text{last}}$	**98.00**	48.78
$\text{RoBERTa-large}_{\text{selected}}$	96.07	**53.90**
$\text{BART-large}_{\text{last}}$	**98.30**	45.69
$\text{BART-large}_{\text{selected}}$	96.59	**52.76**

Table 6: Accuracies in the RTE and ZSTC tasks. The *last* subscript indicates that it was the last checkpoint stored during fine-tuning, while the *selected* subscipt refers to the checkpoint selected with early stopping.

In Table 6 it is clear that the early stopping technique improves the results in the ZSTC evaluation task, since both architectures obtain a substantial boost when early stopping is applied. Overall, the RoBERTa-large model performs better than BART-large, what we hypothesize that might be caused by a loss of generality from models that achieve higher scores in the RTE task.

Model	**arXiv**	**SciDocs-MesH**	**SciDocs-MAG**	**Konstanz**	**Elsevier**	**PubMed**
fb/bart-large-mnli	33.28	**66.18**	51.77	54.62	28.41	**31.59**
bart-large-rte$_{\text{selected}}$	36.71	56.57	63.98	63.57	48.48	27.46
bart-large-rte$_{\text{last}}$	28.58	44.21	57.75	62.25	42.51	21.72
SCIroShot$_{\text{selected}}$	**42.22**	59.34	**69.86**	**66.07**	**54.42**	27.93
SCIroShot$_{\text{last}}$	35.44	52.27	65.27	60.74	50.92	22.85

Table 7: Label wise weighted F1 score of different models in our scientific benchmark. For simplicity, the hypothesis template was set to *"This example is {}."* in all cases.

6 Results

This section assesses the performance of our zero-shot classifiers, which were trained on a weakly supervised entailment dataset of scientific text. In an effort to obtain a more complete picture, we perform both an in-domain and out-of-domain study.

6.1 Scientific domain

For an in-domain evaluation, the in-house generated scientific benchmark from Section 4.2.1 was used. We compare our SCIroShot with the strong baseline set by Facebook's *bart-large-mnli* model[7]. This NLI-based model is the most downloaded zero-shot classifier in the HuggingFace Hub (over 1M monthly downloads) and the one used by default in the *zero-shot-classification* pipeline from their transformers library (Wolf et al., 2020). For an apples-to-apples comparison, we also consider a BART-large model that was trained following the same methodology employed for SciroShot, as it has the same architecture as the Facebook model.

The results presented in Table 7 prove the importance of domain-specific training, something that has been repeatedly seen in traditional text classification. The models trained on MAG data obtain the best results in four out of six datasets, by large margins in all cases, and interestingly enough the Facebook model only wins in the two medical datasets: PubMed and SciDocs-MeSH. The numbers also support our theory that too much training can be detrimental, as the models selected with early stopping score higher than their "last" counterpart in all cases.

6.2 General domain

For the out-of-domain study, we include ourselves in the benchmark proposed by Yin et al. (2019). This amounts to a total of three datasets that cover a variety of topics, with classes ranging from news article topics to human emotions.

(Yin et al., 2019)	Topic	this text is about {}
	Emotion	this text expresses {}
	Situation	The people there need {}
(Ma et al., 2021)	Topic	It is related with {} .
	Emotion	This person feels {} .
	Situation	The people there need {} .
SCIroShot (ours)	Topic	This example is {}
	Emotion	
	Situation	

Table 8: Hypothesis templates used for each dataset. The {braces} indicate the label name location.

The results reported in Table 9 show that our SCIroShot model is competitive in other domains as well. Actually, it is quite impressive that it was able to outperform the rest in two tasks and obtain the second highest score in the third one.

It is important to note that the Situation dataset has the added difficulty of being multi-label, meaning that a piece of text can be linked to an arbitrary number of labels. In a single-label setting, softmax is applied over all the labels logits (so that they sum up to one) and the highest scoring class is chosen as the final prediction. On the other hand, when performing multi-label classification, the softmax function is applied to each label separately (so they are independent variables that do not add up to one) and all labels with a probability above a certain threshold are selected. We did not tune such threshold, so the model's predictions included all labels with a score higher than 0.5.

7 Analysis

7.1 RTE vs ZSTC in the scientific benchmark

This section analyses the relation between the training task and the ZSTC task to assess the effectiveness of the early stopping technique presented in Section 5.3. We evaluate every checkpoint in our ZSTC Scientific Benchmark in the same way that several models were evaluated in Section 5.2.

---

[7] `https://huggingface.co/facebook/bart-large-mnli`

Model	Topic	Emotion	Situation
RTE (Yin et al., 2019)	43.8	12.6	**37.2**
FEVER (Yin et al., 2019)	40.1	24.7	21.0
MNLI (Yin et al., 2019)	37.9	22.3	15.4
NSP (Ma et al., 2021)	50.6	16.5	25.8
NSP (Reverse) (Ma et al., 2021)	53.1	16.1	19.9
$\text{SCIroShot}_{\text{last}}$	51.51	22.63	23.70
$\text{SCIroShot}_{\text{selected}}$	**59.08**	**24.94**	27.42

Table 9: Results obtained in the general benchmark. Accuracy is used for Topic classification and label-wise weighted F1 for the rest. For simplicity, the hypothesis template was set to "This example is ." in all cases.

Figures 3 and 4 corroborate our hypothesis that the performance in the ZSTC task does not necessarily improve hand-by-hand with the performance on RTE. Taking a closer look to the plots we can appreciate how the ZSTC accuracy drops after the model reaches an accuracy of 96% on RTE. The checkpoints selected using the early stopping technique based on the ZSTC task achieve 96.07% RTE accuracy in the case of RoBERTa-large and 96.60% for BART-large. We argue that, although the selected checkpoints are not the best possible option for all the datasets in the benchmark, our technique has stopped training before the overall ZSTC performance diminishes, and it has done so using a subset of unseen labels from the training dataset. This implies obvious savings in time and computation that would otherwise have been wasted for no good reason.

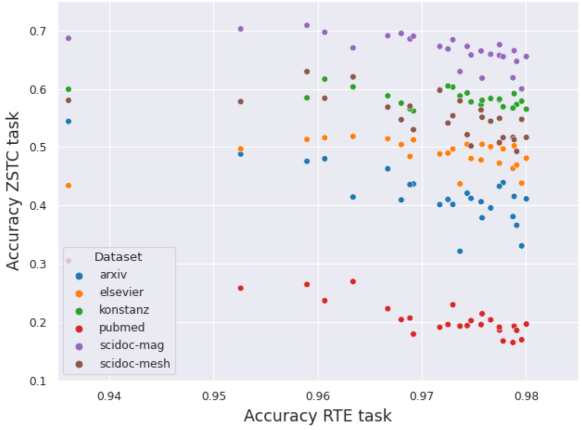


Figure 3: Evaluation of all RoBERTa-large checkpoints in the training (RTE) and testing (ZSTC) tasks.

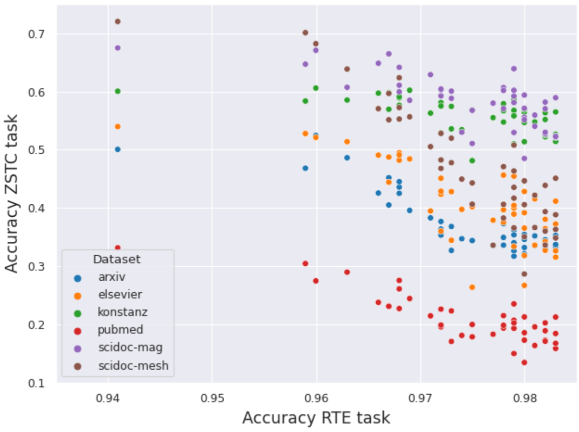


Figure 4: Evaluation of all BART-large checkpoints in the training (RTE) and testing (ZSTC) tasks.

7.2 Robustness to hypothesis templates

This section is an attempt to measure the importance of the hypothesis template and its impact on the final performance of a zero-shot model. With this goal in mind, we evaluate our SCIroShot and Facebook's *bart-large-mnli* with two hypothesis templates that are virtually the same: "*This example is {LABEL}*" and "*This example is {LABEL}.*". Note that their semantic content is exactly the same, being the only difference that the first template does not contain a punctuation mark at the end.

Figure 5 shows that SCIroShot is quite robust against changes in the hypothesis template. On the other hand, as it can be seen in Figure 6, *bart-large-mnli* can experience severe performance drops caused by an apparently insignificant change in the template. We hypothesize that this might happen because the model was trained with high-quality NLI data where dots were always present at the end of the hypothesis, and thus it is not used to the absence of these type of anchor tokens. It can also be inferred that our training task is quite robust to different hypothesis templates.

We would like to point out that the high sensitivity of the Facebook model was accidentally detected during our experiments. We noticed that using our default template the model obtained sur-

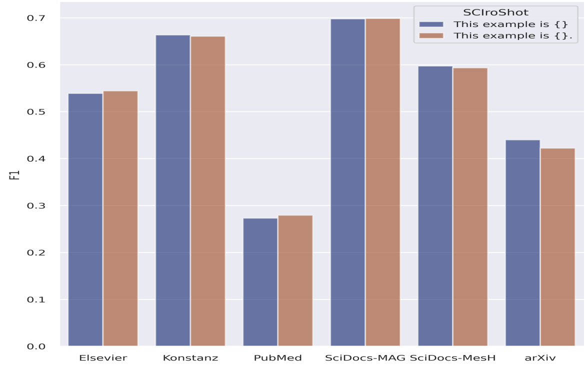


Figure 5: SCIroShot performances when using slightly different hypothesis templates.

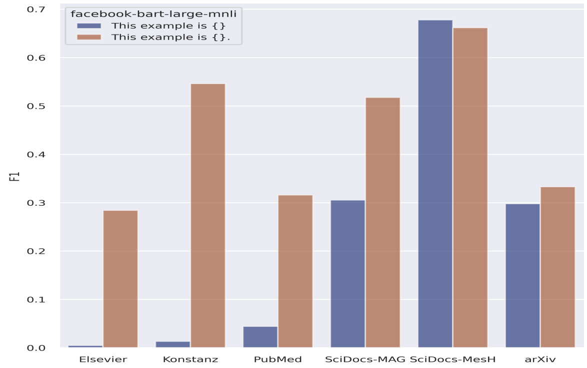


Figure 6: Facebook's bart-large-mnli performances when using slightly different hypothesis templates.

prisingly low results in several tests, something highly unexpected considering its great capabilities. In pursue of intelectual honesty, we decided to report the results obtained with the template that best suited the model's interests. The exact same template was given to our models so that there was one less thing to take into account when comparing performances, even though it might not be the one providing the best results.

Conclusion

The benefits offered by a zero-shot text classifier are four-pronged: (i) it removes the need for time-consuming annotation processes carried out by domain experts, (ii) reduces the computational cost of having to fine-tune a model for each application, (iii) allows the classification of documents in real-world scenarios where there is a scarcity of data, and (iv) is able to handle new classes that might not even have existed at the time of training.

This work proposes the usage of readily available classification datasets for effortless generation of entailment data, which can be used to train ZSTC models. By not using conventional RTE datasets of generic nature, the resulting model exhibits superior performance in the domain for which it has been trained. We show that this is the case for the scientific domain, but the idea could certainly be extrapolated to other fields (e.g. a model trained on news articles should excel at topic classification of this kind of documents). As a proof of concept, we present a scientific-domain zero-shot text classifier that achieves state-of-the-art performance in the scientific domain and competitive results in other areas.

Furthermore, our experiments and analysis suggest that entailment-based classifiers are no panacea: they are very sensitive to the input sequences and do not exhibit the linear correlation that one would expect between the performance on the training (RTE) and testing (ZSTC) tasks. We have empirically proven that the model can become worse at ZSTC as it improves in RTE, which is counter-intuitive and goes against the idea of entailment being a unified method to model other downstream tasks. Our analysis also show that our technique does not suffer from the instability observed in models trained with conventional RTE datasets, which can occasionally experience severe performance drops with minor changes such as removing a punctuation mark from the hypothesis template.

Future Work

In future work, we will further investigate the correlation between different tasks as this could only be the tip of the iceberg. It might also be interesting to increase the difficulty of the fine-tuning task by working with thousands of fine-grained labels from deeper levels of the MAG taxonomy, aiming to delay the point at which the model performance starts worsening in the ZSTC task.

Our findings also motivate an interesting research direction that we would like to explore: employing novel prompt tuning techniques to find the ideal hypothesis template. Having seen that zero-shot text classifiers experience dramatic performance drops caused by apparently insignificant modifications in the hypothesis text, it is clear to us that there is a need to find a suitable text in an automatic manner. We consider this to be a major pitfall of zero-shot models, and thus we are willing to study the feasibility of applying prompt tuning to this particular case as future work.

Limitations

Our main limitations had to do with time and computational constraints. Given the low efficiency of entailment-based classifiers, which need to execute a forward pass per class, the time required to traverse certain datasets was simply too high. Specially when the number of candidate labels is large, because every sequence-label pair has to be fed through the model to compute the logits of all possible combinations. This low ability to scale is certainly a drawback with respect to traditional classifiers, and the reason that forced us to discard datasets with over 500 labels as well as a few experiments that we intend to leave for future work.

Ethics Statement

We believe that this work meets the ACL Code of Ethics as it provides an already trained zero-shot text classifier that can be used in an endless number of situations that would otherwise require a task-specific fine-tuning. Moreover, one of the main findings of this work is that entailment-based text classifiers should not be over-trained as it negatively affects their final performance. This should encourage fellow NLP practitioners to shorten the training time of their ZSTCs thus minimizing their carbon footprint, which is in line with the idea of moving towards more sustainable language models.

Acknowledgements

This work has received funding from two projects within the European Union's Horizon 2020 research and innovation programme.

The work carried out by Barcelona Supercomputing Center was funded by the IntelComp project, under grant agreement number 101004870.

The work carried out by SIRIS Academic was partially funded by the INODE project, under grant agreement number 863410.

SIRIS Academic also acknowledges the support provided by the PRACE initiative for awarding access to the MareNostrum supercomputer.

References

Lutz Bornmann, Robin Haunschild, and Rüdiger Mutz. 2021. Growth rates of modern science: a latent piecewise growth curve approach to model publication numbers from established and new literature databases. *Humanities and Social Sciences Communications*, 8(1):1–15.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Aaai*, volume 2, pages 830–835.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *ACL*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1*, pages 4171–4186. Association for Computational Linguistics.

Melvil Dewey. 1876. *A classification and subject index, for cataloguing and arranging the books and pamphlets of a library*. Brick row book shop, Incorporated.

Jun He, Liqun Wang, Liu Liu, Jiao Feng, and Hao Wu. 2019. Long document classification from local word glimpses via recurrent attention learning. *IEEE Access*, 7:40707–40718.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Daniel Kershaw and Rob Koeling. 2020. Elsevier oa cc-by corpus.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. 2009. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE conference on computer vision and pattern recognition*, pages 951–958. IEEE.

Peder Larsen and Markus Von Ins. 2010. The rate of growth in scientific publication and the decline in coverage provided by science citation index. *Scientometrics*, 84(3):575–603.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics.

Carolyn E. Lipscomb. 2000. Medical subject headings. *Bulletin of Medical Library Association*, 88 3:265–6.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized bert pretraining approach.

Tingting Ma, Jin-Ge Yao, Chin-Yew Lin, and Tiejun Zhao. 2021. Issues with entailment-based zero-shot text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 786–796.

Stephen Mayhew, Tatiana Tsygankova, Francesca Marini, Zihan Wang, Jane Lee, Xiaodong Yu, Xingyu Fu, Weijia Shi, Zian Zhao, Wenpeng Yin, Karthikeyan K, Jamaal Hay, Michael Shur, Jennifer Sheffield, and Dan Roth. 2019. University of pennsylvania lorehlt 2018 submission. Technical report, University of pennsylvania lorehlt 2019 submission.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2020. What the [mask]? making sense of language-specific bert models. *arXiv preprint arXiv:2003.02912*.

Abiola Obamuyide and Andreas Vlachos. 2018. Zero-shot relation classification as textual entailment. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 72–78.

Laura Ana Maria Oberländer and Roman Klinger. 2018. An analysis of annotated corpora for emotion classification in text. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2104–2119.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluís-Miquel Munguía, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269. Association for Computational Linguistics.

Zhihong Shen, Hao Ma, and Kuansan Wang. 2018. A web-scale system for scientific knowledge exploration. In *Proceedings of ACL 2018, System Demonstrations*, pages 87–92, Melbourne, Australia. Association for Computational Linguistics.

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*, pages 243–246.

Yangqiu Song and Dan Roth. 2014. On dataless hierarchical text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. 2020. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 1(1):396–413.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021a. Want to reduce labeling cost? GPT-3 can help. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4195–4205, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021b. Entailment as few-shot learner.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. 2018. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Conference on Empirical Methods in Natural Language Processing*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

A Fine-tuning hyperparameters

Hyper-parameter	**Value**
Learning Rate	8e-6
Learning Rate Decay	Linear
Weight Decay	0.0
Warmup Steps	0
Batch Size	256
Max. Training Epochs	10
Adam ϵ	1e-8
Adam β_1	0.9
Adam β_2	0.999
Gradient Clipping	1.0

Table 10: Hyper-parameter values.

B Computing infrastructure

The fine-tuning of each model took around 2 days on 16 HPC nodes[8] equipped with an AMD EPYC 7742 (@ 2.250GHz) processor with 128 threads and 2 AMD MI50 GPUs each.

C Proportion of *Entailment* and *Neutral* samples in the training data

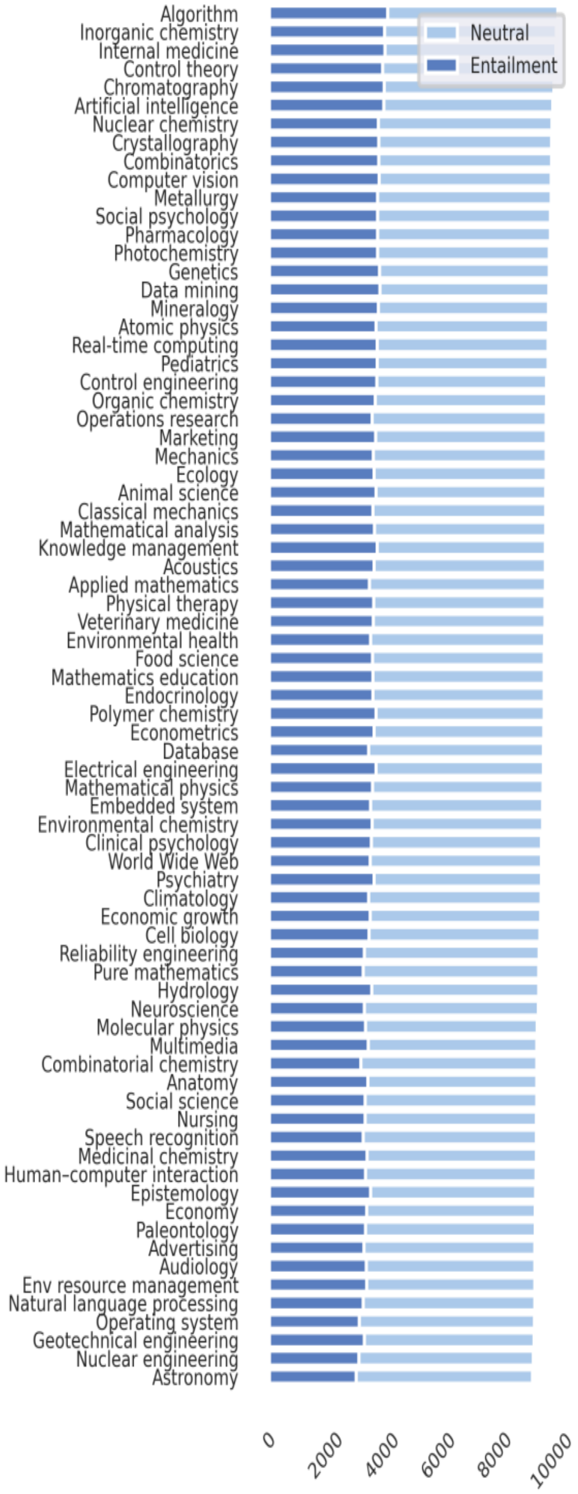


Figure 7: Proportion of entailment and neutral samples in the training data. For space limitations it is not possible to display all 292 labels, so the bar plot has been purposely limited to the top 75 classes.

---

[8] `https://www.bsc.es/innovation-and-services/technical-information-cte-amd`

Fair Enough: Standardizing Evaluation and Model Selection for Fairness Research in NLP

Xudong Han[1,2] **Timothy Baldwin**[1,2] **Trevor Cohn**[1]
[1]The University of Melbourne
[2]MBZUAI
`xudongh1@student.unimelb.edu.au`, `{tbaldwin,t.cohn}@unimelb.edu.au`

Abstract

Modern NLP systems exhibit a range of biases, which a growing literature on model debiasing attempts to correct. However current progress is hampered by a plurality of definitions of bias, means of quantification, and oftentimes vague relation between debiasing algorithms and theoretical measures of bias. This paper seeks to clarify the current situation and plot a course for meaningful progress in fair learning, with two key contributions: (1) making clear inter-relations among the current gamut of methods, and their relation to fairness theory; and (2) addressing the practical problem of model selection, which involves a trade-off between fairness and accuracy and has led to systemic issues in fairness research. Putting them together, we make several recommendations to help shape future work.[1]

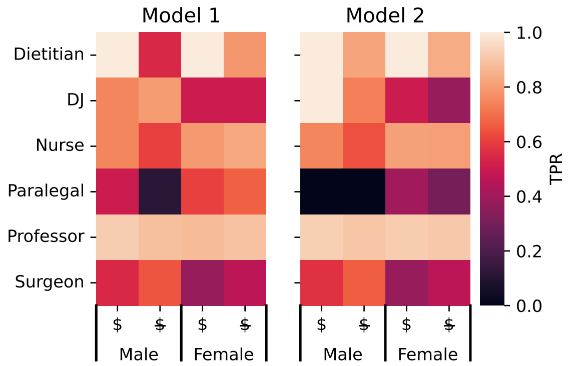


Figure 1: True positive rate (TPR) evaluation results over a biography classification dataset broken down by author demographic and selected profession classes. $ and ~~$~~ denote the economic status (wealthy vs. not, respectively). The pattern of results exhibits various biases, however it is difficult to distil this into a single figure of merit, and thus determine which is the better or fairer of the two models.

1 Introduction

In NLP and machine learning, there has been a surge of interest in fairness due to the fact that models often learn and amplify biases in the training dataset, leading to a range of harms (Badjatiya et al., 2019; Díaz et al., 2018). A central notion is group-wise fairness (Dwork et al., 2012; Chouldechova, 2017; Berk et al., 2021), which is typically measured as the model performance disparities across groups of data that are created by the combinations of protected attributes, such as race and gender. A broad range of bias evaluation metrics have been introduced in previous studies to capture different types of biases – such as *demographic parity* (Feldman et al., 2015) and *equal opportunity* (EO) (Hardt et al., 2016) – and different approaches have been adopted to both measure group disparities within each class, and aggregate over those disparities. Each of these choices implicitly encodes assumptions about the nature of fairness, but little work has been done to spell out what those assumptions are, or guide the selection of evaluation metric from first principles of what constitutes fairness.

As an illustration of this issue, Figure 1 depicts the true positive rate (TPR) values for two models.[2] Given that the EO fairness is satisfied if different groups achieve identical TPR, which model is fairer or "better" out of the two? The answer is far from clear, and in terms of evaluation practice, dictated by a series of choices which implicitly encode different assumptions about what fairness is.

In terms of research practice, these choices have led to a lack of consistency and direct empirical comparability between methods. Equally concerningly, given that fairness research involves an inherent trade-off between raw model performance and fairness, it has more subtly led to a lack of rigour in terms of how model selection has been carried out, meaning that methods are often deployed in

---

[1] Code available at `https://github.com/HanXudong/Fair_Enough`

[2] For further details see Appendix A.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 297–312
May 2-6, 2023 ©2023 Association for Computational Linguistics

suboptimal ways relative to a particular evaluation methodology.

In this paper, we seek to address these problems. We start by surveying current practices for fairness evaluation aggregation within an integrated framework, and discuss considerations and motivations for using different aggregation approaches. To ensure fairness metrics are fully comparable, we present a checklist for reporting fairness evaluation metrics, and also recommendations for aggregation method selections. We next survey model comparison methods, and demonstrate the issues stemming from using inconsistent model selection criteria. To ensure fair comparisons, we further introduce a metric for comparison without model selection, which measures the area under the trade-off curve of each method.

Overall this paper makes two key contributions: (1) we characterise current practices for fairness evaluation and their grounding in theory, proposing a best-practice checklist; and (2) we propose a new method which resolves several issues relating to model selection and comparison.

2 Related Work

In terms of bias metrics, there are mainly two lines of work in the literature on NLP fairness: bias in the geometry of text representations (intrinsic bias), and performance disparities across groups in downstream tasks (extrinsic bias), respectively. Based on the hypothesis that measuring and mitigating intrinsic bias will also reduce extrinsic bias, previous work has mainly focused on measuring and mitigating intrinsic bias, such as the Word Embedding Association Test (WEAT) (Caliskan et al., 2017), Sentence Encoder Association Test (SEAT) (May et al., 2019), and Embedding Coherence Test (ECT) (Dev and Phillips, 2019). However, Goldfarb-Tarrant et al. (2021) recently showed that there is no reliable correlation between intrinsic and extrinsic biases, and suggest future work focusing on extrinsic bias measurement (which is the focus of this work).

As for bias mitigation, debiasing methods for intrinsic and extrinsic bias generally suffer from performance–fairness trade-offs controlled by particular hyperparameters such as the number of principal components used to define the intrinsic bias subspace (Bolukbasi et al., 2016), and the strength of addition objectives for performance parity across groups (Shen et al., 2022b). In measuring performance (perplexity and LM score for sentence embeddings, for example) and fairness simultaneously, the model comparison framework presented in this paper is generalizable for both intrinsic and extrinsic fairness.

3 Fairness Metrics

In this section, we discuss the considerations involved in fairness evaluation. We start with a survey of different methods for aggregating scores, and propose a two-step aggregation framework for fairness evaluation.

3.1 Formal Notation Preliminaries

We consider fairness evaluation in a classification scenario. Evaluation is based on a test dataset consisting of n instances $\mathcal{D} = \{(\boldsymbol{x}_i, \mathrm{y}_i, \mathrm{z}_i)\}_{i=1}^{n}$, where $\boldsymbol{x}_i$ is an input vector, $\mathrm{y}_i \in \{\mathrm{c}\}_{\mathrm{c}=1}^{\mathrm{C}}$ represents target class label, and $\mathrm{z}_i \in \{\mathrm{g}\}_{\mathrm{g}=1}^{\mathrm{G}}$ is the group label, such as gender.[3]

Given a model that has been trained to make predictions w.r.t. the target label $\hat{\mathrm{y}} = f(\boldsymbol{x})$, fairness evaluation metrics generally measure group-wise performance disparities for a particular metric $m(\mathrm{y}, \hat{\mathrm{y}})$. For example, positive predictive rate and true positive rate have been employed as the metric for demographic parity (Feldman et al., 2015) and equal opportunity (Hardt et al., 2016), respectively.

For each group, the results of a metric m are C-dimensional vectors, one dimension for each class. Given G protected groups, the full results are organized as a C × G matrix, denoted as $\boldsymbol{M}$. For the subset of instances $\mathcal{D}_{\mathrm{c,g}} = \{(\boldsymbol{x}_i, \mathrm{y}_i, \mathrm{z}_i) | \mathrm{y}_i = \mathrm{c}, \mathrm{z}_i = \mathrm{g}\}_{i=1}^{n}$, we denote the corresponding evaluation results as $M_{\mathrm{c,g}}$. Taking Figure 1 as an example, $\boldsymbol{M}$ refers to the heatmap plot, and $M_{\mathrm{c,g}}$ is the cell in the c-th row and g-th column.

Given $\boldsymbol{M}$, the question is how exactly to aggregate the result matrix as a single number that measures the degree of fairness. We split the aggregation into two steps: (1) group-wise aggregation, which aggregates evaluation results of all groups within a class ($[M_{\mathrm{c},1}, \ldots, M_{\mathrm{c,G}}]$) into a single number ($\beta_{\mathrm{c}}$); and (2) class-wise aggregation, which aggregates $[\beta_1, \ldots, \beta_{\mathrm{C}}]$ scores of all classes into a single number δ.[4]

---

[3] When considering multiple protected attributes, z can be intersectional identities as shown in Figure 1.

[4] Mathematically, it would be possible to do the class-wise aggregation first, and then the group-wise aggregation. However, aggregating class-wise performances within a particular group essentially measures the long-tail learning problem

Aggregation	Method	Description	Unit	Reference
Group-wise	mean gap	$\beta_{\mathrm{c}} = \frac{1}{\mathrm{G}} \sum_{\mathrm{g}} \lvert M_{\mathrm{c,g}} - \overline{M}_{\mathrm{c}} \rvert$	G	(Shen et al., 2022b)
	variance	$\beta_{\mathrm{c}} = \frac{1}{G-1} \sum_{\mathrm{g}} \lvert M_{\mathrm{c,g}} - \overline{M}_{\mathrm{c}} \rvert^2$	G	(Lum et al., 2022)
	max gap	$\beta_{\mathrm{c}} = \max_{\mathrm{g}} \lvert M_{\mathrm{c,g}} - \overline{M}_{\mathrm{c}} \rvert$	G	(Yang et al., 2020)
	min score	$\beta_{\mathrm{c}} = \min_{\mathrm{g}} M_{\mathrm{c,g}}$	S	(Lahoti et al., 2020)
	min ratio	$\beta_{\mathrm{c}} = \min_{\mathrm{g}} \frac{M_{\mathrm{c,g}}}{\overline{M}_{\mathrm{c}}}$	R	(Zafar et al., 2017)
	max difference	$\beta_{\mathrm{c}} = \max_{\mathrm{g}} M_{\mathrm{c,g}} - \min_{\mathrm{g}} M_{\mathrm{c,g}}$	S	(Bird et al., 2020)
	max ratio	$\beta_{\mathrm{c}} = \frac{\max_{\mathrm{g}} M_{\mathrm{c,g}}}{\min_{\mathrm{g}} M_{\mathrm{c,g}}}$	R	(Feldman et al., 2015)
	difference threshold (γ)	$\beta_{\mathrm{c}} = \frac{1}{\mathrm{G}} \sum_{\mathrm{g}} \mathbb{1}_{[0,\gamma]}(\lvert M_{\mathrm{c,g}} - \overline{M}_{\mathrm{c}} \rvert)$	G	(Kearns et al., 2019)
	ratio threshold (γ)	$\beta_{\mathrm{c}} = \frac{1}{\mathrm{G}} \sum_{\mathrm{g}} \mathbb{1}_{[0,\gamma]}(\lvert \frac{M_{\mathrm{c,g}}}{\overline{M}_{\mathrm{c}}} - 1 \rvert)$	R	(Barocas et al., 2019)
Class-wise	binary	$\delta = \sum_{\mathrm{c}} \beta_{\mathrm{c}} \mathbb{1}_{\{1\}}(c)$	β	(Roh et al., 2021)
	quadratic mean	$\delta = \sqrt{\frac{1}{\mathrm{C}} \sum_{\mathrm{c}} \beta_{\mathrm{c}}^2}$	β	(Romanov et al., 2019)
	mean	$\delta = \frac{1}{\mathrm{C}} \sum_{\mathrm{c}} \beta_{\mathrm{c}}$	β	(Li et al., 2018)

Table 1: Summary of different aggregation approaches. Based on the basic unit, group-wise aggregations are additionally categorized into three types: **S**core ($M_{\mathrm{c,g}}$), **G**ap ($|M_{\mathrm{c,g}} - \overline{M}_{\mathrm{c}}|$), and **R**atio ($\frac{M_{\mathrm{c,g}}}{\overline{M}_{\mathrm{c}}}$).

3.2 Existing Aggregation Approaches

Table 1 summarizes several aggregation approaches from previous work, which are categorized based on the level of aggregation.

3.2.1 Basic Unit

The **basic unit** refers to the inputs to an aggregation function.

Group-wise Broadly, there are three types of basic units for group-wise aggregation:

1. the original **score** ($M_{\mathrm{c,g}}$), which maintains the actual performance level under aggregation and larger is better;
2. the **gap**, i.e., absolute difference, between the evaluation results of a group and the average performance ($|M_{\mathrm{c,g}} - \overline{M}_{\mathrm{c}}|$), where smaller is better and the minimum is 0; and
3. **ratio** of the evaluation results of a group to the average ($\frac{M_{\mathrm{c,g}}}{\overline{M}_{\mathrm{c}}}$), where closer to 1 is better.

Score describes the actual performance of each group, and is generally used to measure extrema of actual performances. For example, the *Rawlsian Max Min* criterion (Rawls, 2001) is satisfied if the utility of the worst-performing group is maximized. Related fairness notions are also known as per-group fairness (Hashimoto et al., 2018; Lahoti et al., 2020).

The other two units, **gap** and **ratio**, support the notion of group fairness, and evaluate whether or not ŷ is fair w.r.t. z. Taking EO (Hardt et al., 2016) as an example, it requires the true positive rate to be independent of z. Formally, for a particular class c, the EO criterion is satisfied iff

$$\mathrm{TPR}_{\mathrm{c,g}} = \overline{\mathrm{TPR}}_{\mathrm{c}}, \forall \mathrm{g} \in \{\mathrm{g}\}_{\mathrm{g}=1}^{\mathrm{G}}.$$

As such, it is straightforward to directly measure the absolute difference between $\mathrm{TPR}_{\mathrm{c,g}}$ and $\overline{\mathrm{TPR}}_{\mathrm{c}}$,

$$\mathrm{TPR}_{\mathrm{c,g}} = \overline{\mathrm{TPR}}_{\mathrm{c}} \Leftrightarrow |\mathrm{TPR}_{\mathrm{c,g}} - \overline{\mathrm{TPR}}_{\mathrm{c}}| = 0,$$

which is essentially the **gap** unit.

Alternatively, the **ratio** unit can be used to measure inequality as a percentage:

$$\mathrm{TPR}_{\mathrm{c,g}} = \overline{\mathrm{TPR}}_{\mathrm{c}} \Leftrightarrow \frac{\mathrm{TPR}_{\mathrm{c,g}}}{\overline{\mathrm{TPR}}_{\mathrm{c}}} = 1.$$

Ratio-based scores can also be interpreted via a "$q\%$-rule" (Zafar et al., 2017; Barocas et al., 2019), for example, the 80%-rule for *disparate impact* (Feldman et al., 2015), which requires that the ratio is no less than 0.8.

The $q\%$-rule can be captured more explicitly by a threshold (Kearns et al., 2018; Barocas et al., 2019), which is a relaxation of the equality based on a slack threshold $\epsilon \in \mathbb{R}^{+}$, $|1 - \frac{\mathrm{TPR}_{\mathrm{c,g}}}{\overline{\mathrm{TPR}}_{\mathrm{c}}}| \leq \epsilon$. Similarly, the threshold can be applied to **gap**, resulting in $|\mathrm{TPRc, g} - \overline{\mathrm{TPR}}_{\mathrm{c}}| \leq \epsilon$.

Class-wise The next step is class-wise aggregation, taking the group-wise aggregation for each class from above as inputs, $[\beta_1, \ldots, \beta_{\mathrm{C}}]$.

3.3 Generalized Mean Aggregation

Before discussing each of these aggregation methods, we first introduce the basic concept of the

---

rather than fairness.

Power (p)	**Formulation**
$-\infty$	Minimum: $\min\{v_1, \ldots, v_n\}$
-1	Harmonic Mean: $\frac{n}{\sum_{i=1}^{n} v_i^{-1}}$
1	Arithmetic Mean: $\frac{1}{n}\sum_{i=1}^{n} v_i$
2	Quadratic Mean: $\sqrt{\frac{1}{n}\sum_{i=1}^{n} v_i^2}$
$+\infty$	Maximum: $\max\{v_1, \ldots, v_n\}$

Table 2: Commonly-used cases of generalized mean aggregation.

generalized mean as a framework for describing aggregation functions, and then make the link between the generalized mean and existing aggregation methods.

Formally, the generalized mean is defined as:

$$\mathcal{M}_p(v_1, \ldots, v_n) = \left(\frac{1}{n} \sum_{i=1}^{n} v_i^p \right)^{\frac{1}{p}},$$

where $v_i \in \mathbb{R}^+$ are positive real numbers to be aggregated, and p is the exponent parameter. A desired property of the generalized mean is its inequality, which states that,

$$\mathcal{M}_p(v_1, \ldots, v_n) > \mathcal{M}_{p'}(v_1, \ldots, v_n), \forall p > p'.$$

Essentially, a larger value of p encourage the aggregation to focus more on the larger-valued elements, which can be illustrated with the specific cases shown in Table 2.

By setting $p = \pm\infty$, generalized mean returns extremum values, including (in Table 1): (1) the maximum value of gap (Yang et al., 2020), difference (Bird et al., 2020), and ratio (Feldman et al., 2015); and (2) the minimum value of score (Lahoti et al., 2020) and ratio (Zafar et al., 2017).

For other p values, the generalized mean reflects the relative dispersion of its inputs. For example, group-wise mean gap aggregation (Shen et al., 2022b) and class-wise mean aggregation (Li et al., 2018) are both equivalent to $p = 1$. Class-wise quadratic mean aggregation (Romanov et al., 2019) is essentially $p = 2$, which focuses more on those classes with higher bias. Similarly, group-wise variance aggregation (Lum et al., 2022) is proportional to the $p = 2$ setting, implying that groups with larger gaps will influence results more.

The additional advantage of using generalized mean aggregation is that comparison across arbitrary p values can be easily stated. For example, group-wise aggregation is the $p = -5$ generalized mean with respect to the **score** units in a toxicity classification competition,[5] meaning that evaluation focuses more on groups with lower performance.

Other Aggregation Methods: Although generalized mean aggregation is a powerful tool for describing and interpreting the aggregation process, there are other ways that need further discussion. Previous work has also considered assigning different weights under aggregation, for instance, Kearns et al. (2018) assign larger weights to groups with larger populations. Such aggregations can be implemented as the weighted generalized mean: $\mathcal{M}_{p,w}(v) = \left(\frac{1}{n}\sum_{i=1}^{n} w_i v_i^p\right)^{\frac{1}{p}}$, where w is the weight vector, and $\sum_{i=1}^{n} w_i = 1$.

An example of the weighted generalized mean for class-wise aggregation is binary aggregation that only considers the positive class in a binary classification setting (Hardt et al., 2016; Zafar et al., 2017; Kearns et al., 2018; Zhao et al., 2019; Lahoti et al., 2020; Han et al., 2021; Lum et al., 2022). The positive class is often treated as the “advantaged” outcome, so the analysis focuses solely on the positive class. Moreover, the one-versus-all trick is not necessary for the binary setting, and natural derivations of the confusion matrix can be used to refer to a particular class, e.g., TPR for the positive class and TNR for the negative class.

3.4 Recommendations

We are now in a position to be able to provide recommendations for fairness evaluation.

Following the work of Dodge et al. (2019), we provide a checklist for fairness evaluation metric aggregation:

- ☐ Statistics of the dataset $\mathcal{D}$, e.g., the probability table of the joint distribution of y and z, and the size of each partition.
- ☐ The evaluation metric m (e.g., TPR for *EO* fairness).
- ☐ The basic unit of group-wise aggregation, including **score**, **gap**, and **ratio**, or other possible measures.
- ☐ The aggregation function for **group-wise** aggregation, and the corresponding motivation.

---

[5] Jigsaw Unintended Bias in Toxicity Classification: `https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/`

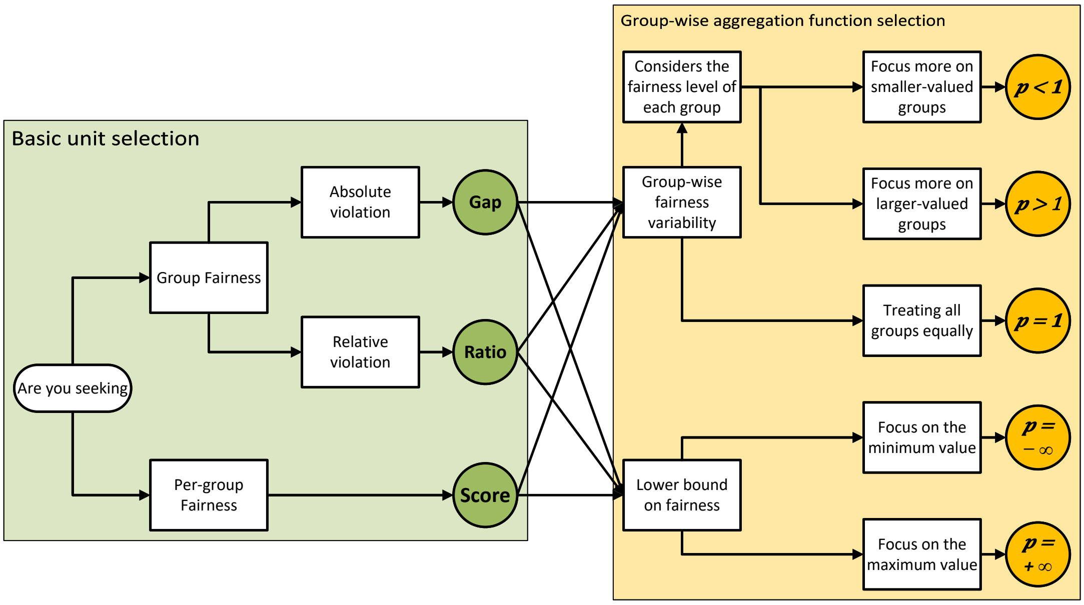


Figure 2: Decision path for exponent parameter selection for generalized mean aggregation.

☐ The aggregation function for **class-wise** aggregation, and the corresponding motivation.

Although the particular choice of evaluation dataset $\mathcal{D}$ and evaluation metric m are critical to the overall evaluation, they are not the main focus of this paper. Rather, we provide guidance based on the selection of basic unit, and methods for group- and class-wise aggregation, as detailed in Figure 2.

Basic Unit Selection: The circles in Figure 2 annotated as **Score**, **Ratio**, and **Gap** are the decision points for basic unit selection.

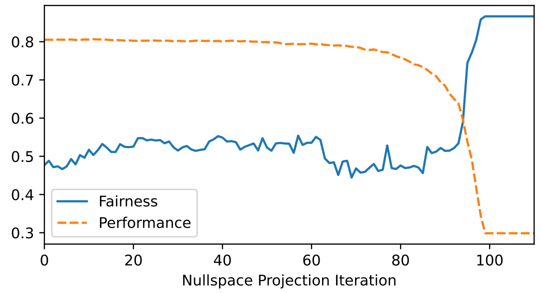
If per-group fairness is the primary criterion (e.g., Rawlsian Max-Min fairness (Rawls, 2001)), using **score** is the best practice, which maintains the original values under aggregation. On the other hand, if inter-group fairness is critical, **gap** and **ratio** are more appropriate choices. **Gap** reflects disparities in the same scale as the per-group scores, and is easy to visualize (e.g. as differences in height between clustered bars). However, if one wished to measure disparities in relative terms, e.g., the $q\%$-rule (Feldman et al., 2015), **ratio** is a better choice than **gap**.

Group-wise Aggregation Function Selection: The selection of group-wise aggregation functions is shown as the exponent parameters of the *generalized mean* aggregation.

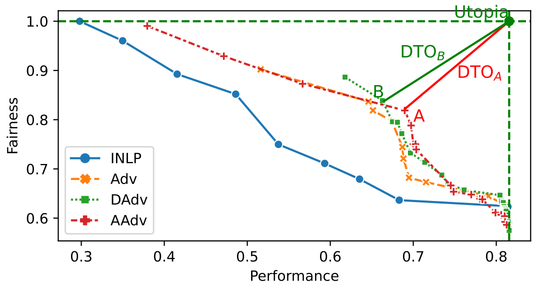
Measuring extrema is similar to the notion of per-group fairness, and encourages improvements in the worst-performing groups. For basic units where smaller is fairer, e.g., **gap**, aggregation generally focuses on the maximum (Yang et al., 2020), i.e., $p = +\infty$. For units like **score** ($p = -\infty$), on the other hand, the minimum value should be measured, as a lower bound.

Besides extrema, it is also reasonable to measure fairness variability across groups. A typical choice is taking the arithmetic mean (i.e., $p = 1$) across all groups, which implicitly assigns equal importance to each individual group. Similar to the signs in extremum aggregations, the value of p in variability aggregations should be selected based on the type of basic unit, to focus more on worse-performing groups. Taking the **gap** unit as an example, the quadratic mean ($p = 2$) is influenced more by larger gaps than the arithmetic mean. Moreover, quadratic mean aggregation based on **gap** is essentially the standard deviation of **scores**, and can be used to reconstruct variance aggregation (Lum et al., 2022).

Class-wise Aggregation Function Selection: Although our focus is on the fairness evaluation metric, class-wise aggregation is almost identical to aggregation methods for general utility metrics. Binary aggregation for fairness is the same as utility metrics, while mean aggregation (Li et al., 2018; Wang et al., 2019) for fairness evaluation is equivalent to "macro"-averaging in general evaluation.



(a) Tuning INLP trade-off hyperparameter



(b) Performance–fairness trade-offs

Figure 3: Figure 3a shows an example of performance and fairness with respect to different iterations of the nullspace projection in INLP. Figure 3b presents the Pareto frontiers in the performance–fairness trade-offs of four debiasing methods in recent work. Figure 3b also provides an illustration of DTO. The green dashed vertical and horizontal lines denote the best performance and fairness, respectively, and their intersection point is the *Utopia* point. The length of a line, e.g., the red line from A to the Utopia point, is the DTO for the corresponding candidate model.

4 Model Comparison

This section focuses on comparison of debiasing methods when considering utility and fairness simultaneously. We first introduce the performance–fairness trade-off curve (PFC) for debiasing methods, and then discuss the limitations of existing comparison frameworks. Finally, we propose a new metric, namely the area under the curve (AUC) w.r.t. PFC, which integrates existing approaches and reflects the overall goodness of a method.

4.1 Performance and Fairness Metrics

As discussed in Section 3, there are many options to measure performance and fairness. This paper is generalizable to all different metrics, but for illustration purposes, we follow Ravfogel et al. (2020); Subramanian et al. (2021) and Han et al. (2022c) in measuring the overall accuracy and equal opportunity fairness.

Specifically, equal opportunity fairness measures TPR disparities across groups, such as the situation depicted in Figure 1. We use the TPR **gap** across subgroups to capture absolute disparities. For group-wise aggregation, we treat all groups equally in computing the unweighted sum of *gap* scores ($\propto p = 1$). In the last step, class-wise aggregation, we focus more on less fair classes by using root mean square aggregation ($p = 2$).

4.2 Performance–Fairness Trade-off

It has been observed in previous work that a performance–fairness trade-off exists in bias mitigation (Li et al., 2018; Wang et al., 2019; Ravfogel et al., 2020; Han et al., 2022b; Shen et al., 2022b).

Typically, debiasing methods involve a trade-off hyperparameter to control the extent to which the model sacrifices performance for fairness. Examples of such trade-off hyperparameters include: (1) interpolation between the target and vanilla data distribution for pre-processing approaches (Wang et al., 2019; Han et al., 2022a); (2) the strength of additional loss terms for loss manipulation methods (Zhao et al., 2019; Lahoti et al., 2020; Han et al., 2021; Shen et al., 2022a); (3) the target level of fairness in constrained optimization (Kearns et al., 2018; Subramanian et al., 2021); and (4) the number of debiasing iterations for post-hoc bias mitigation methods (Ravfogel et al., 2020).

Taking INLP (Ravfogel et al., 2020) as an example, which debiases by iteratively projecting the text embeddings to the nullspace of the protected attributes, Figure 3a shows performance and fairness with respect to the number of nullspace projection iterations.[6] It is clear that more iterations lead to better fairness at the cost of performance.

Instead of looking at performance/fairness for different trade-off hyperparameter values, it is more meaningful to focus on the Pareto frontiers in trade-off plots (Figure 3b), where each point corresponds to a particular value of the trade-off hyperparameter in Figure 3a. The frontiers represent the best fairness that can be achieved at different performance levels, and vice versa.

One limitation of a trade-off plot is that it is hard to make quantitative conclusions based on the plot itself, and we cannot conclude that one method is better than another if there exists any intersection of their trade-off curves. As shown in Figure 3b, in addition to INLP, we also include the trade-

---

[6] Without loss of generality, we assume that for both fairness and performance, larger is better.

off curves for three recent adversarial debiasing variants: ADV (Li et al., 2018), DADV (Han et al., 2021), and A-ADV (Han et al., 2022b). Although A-ADV is better than the other methods under most conditions, there exist intersections between their trade-off curves. As such, we can only state that A-ADV is better than other methods within particular ranges, which is insufficient for making a precise comparison, especially when comparing multiple debiasing methods (as demonstrated in Figure 3b).

4.3 Model Selection

In order to conduct quantitative comparisons across different debiasing methods, current practice is to select a particular point on the frontier for each method, and then compare both the performance and fairness of the selected points.

One problem associated with model selection is that typically, no single method simultaneously achieves the best performance and fairness. For example, as shown in Figure 3b, if points **A** and **B** were the selected models for A-ADV and DADV, respectively, **A** would represent better performance and **B** better fairness. As such, although we have actual numbers for quantitative comparison, it is still hard to conclude which method is best.

Distance to the Optimal: To address this problem, we propose to measure the **D**istance **T**o the **O**ptimal point ("DTO") to quantify the performance–fairness trade-off (Salukvadze, 1971; Marler and Arora, 2004; Han et al., 2022a). A model is said to outperform others if it achieves a smaller DTO, i.e. the distance to the optimal (Utopia) point (the point at which performance and fairness are the maximum possible values) is minimized. Figure 3b illustrates the calculation of DTO for **A** and **B**, where the optimal point is the top-right corner[7] and DTO is measured by the normalized Euclidean distance (the length of the green and red lines) to the optimal point.

A notable advantage of DTO is that a Pareto improvement implies a smaller value of DTO. Therefore, DTO can be seen as relaxation of Pareto improvements, and the smallest DTO must be achieved by a point on the Pareto frontier. A key limitation of DTO is that it quantifies the trade-off of a single model rather than the full frontier, presupposing some means of model selection. This has been somewhat arbitrary in prior work, which is the problem we now seek to address.

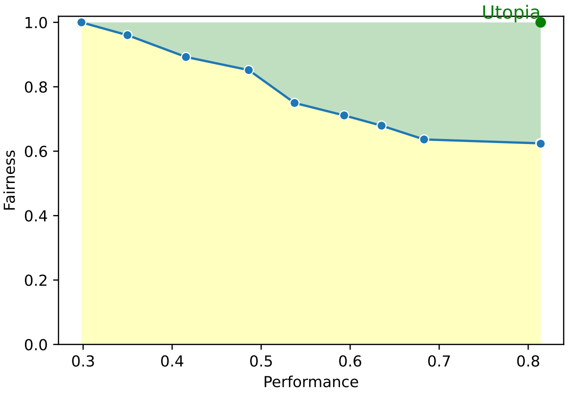


Figure 4: AUC of the performance–fairness trade-off curve. Taking the trade-off curve of INLP as an example, the yellow shaded area refers to the AUC-PFC, and the green shaded area refers to the integration of DTO in polar coordinates.

Selection Criteria: Similar to the aggregation of fairness metrics, model selection should be done in a domain-specific manner. Previous work has used different criteria for model selection, including: (1) minimum loss (Hashimoto et al., 2018; Li et al., 2018); (2) maximum utility (Lahoti et al., 2020), e.g., based on accuracy or F-measure; (3) manual selection based on visual inspection of the trade-off curve (Elazar and Goldberg, 2018; Ravfogel et al., 2020); (4) constrained selection (Han et al., 2021; Subramanian et al., 2021), by selecting the best fairness constrained to a particular level of performance, and vice versa; and (5) minimising DTO (Han et al., 2022b; Shen et al., 2022b).

Selection based on minimum loss and maximum utility is identical to classic model selection, and does not consider fairness explicitly. The other three types of criteria are based on trade-offs, differentiated by the method for aggregating fairness and performance.

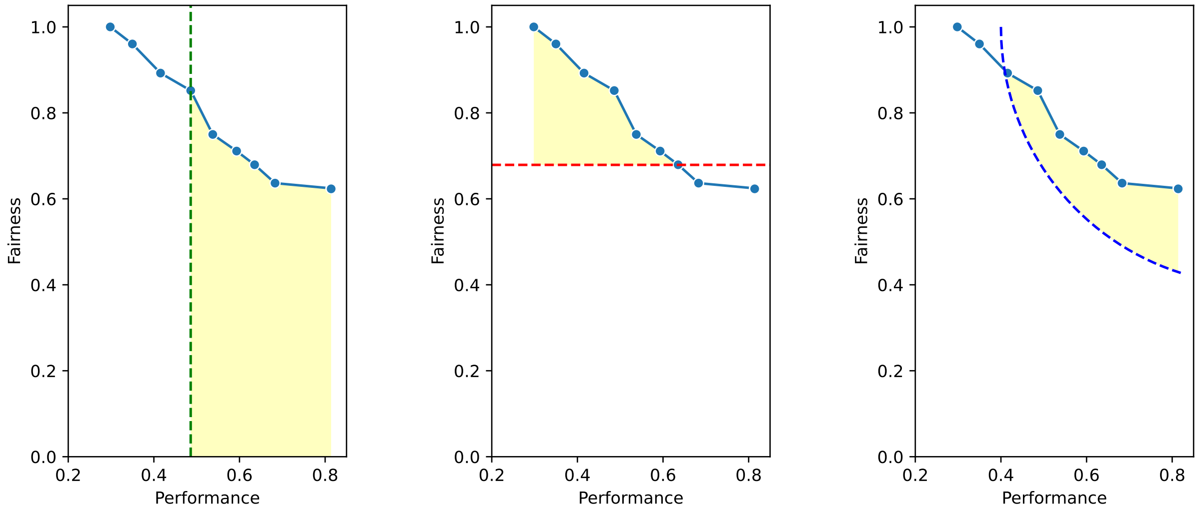
Such inconsistency in model selection makes it very hard to rigorously compare methods. The question we want to address is: *how can we quantitatively compare methods without model selection?*

4.4 AUC-PFC

Recall that DTO is a metric for measuring the goodness of the trade-off of a particular model, and model selection is a process for selecting a particular frontier model from the Pareto curve. To address the problem associated with model selec-

---

[7] The location of the utopia point and the scale of metrics are discussed in Appendix C.



(a) Accuracy is better than 0.49.

(b) Fairness is better than 0.68.

(c) DTO is better than 0.60.

Figure 5: Yellow shaded area denote the partial AUC-PFC score computed in the region where a particular condition applied.

tion, we propose to integrate DTO over the whole performance–fairness curve (PFC). Specifically, we integrate DTO in a polar coordinate system, where the reference point (pole) is the optimal point. Given that the DTO of a point on the trade-off curve is also the distance from the pole in the polar coordinate system, the trade-off curve can be treated as a function that maps angular coordinates to DTO. For example, as shown in Figure 4, the green area denotes the region enclosed by the performance–fairness trade-off curve of INLP and the utopia point with fairness = 1.00 and performance = 0.82.[8]

Alternatively, we can interpret the proposed metric from the performance–fairness perspective, in calculating the area under the Pareto curve, and subtracting this from the area under the optimal Pareto curve defined by the optimal point.

The magnitude of AUC-PFC differs from a single metric; for example, a 0.0001 improvement in the AUC-PFC score is equivalent to a 1 percentage point (pp) boost in both performance and fairness (0.01×0.01).

Partial AUC-PFC In practice, worse performance or fairness can be unacceptable, for example, one may want to prioritize fairness in particular applications. To address this problem, we present the Partial AUC-PFC score to focus on a specific region of the PFC Curve, where the AUC-PFC score is computed w.r.t. specific acceptable levels of performance and fairness.

Figure 5a shows an example of partial AUC-PFC, where the region can be considered if corresponding accuracy is better than 0.49. Similarly, Figures 5b and 5c show partial AUC-PFC scores with respect to particular fairness and DTO constraints.

With the partial AUC-PFC metric, one can explicitly compare different methods with a single number, and w.r.t. particular values of performance and fairness.

4.5 Case Studies

Experimental Details: We conduct experiments over the BIOS dataset (De-Arteaga et al., 2019) which was augmented with economic status by Subramanian et al. (2021), resulting in 28 professions as the target label and 4 intersectional demographic groups.[9] We use public implementations for all models in our experiments, primarily in *fairlib* (Han et al., 2022c).

Results: In Table 3, we investigate 7 different selection criteria and report the DTO score over the test set. Specifically, we conduct model selection over the development set based on: (1) minimum DTO; (2) maximum performance (**P**); (3) maximum performance within a fairness threshold of 5% improvement (**P@F+5%**); (4) maximum

---

[8] In the interests of consistent comparison, the Utopia point is typically $(1, 1)$, as in Table 3. In practice, this does not affect the calculation of AUC-PFC, as we discuss in Appendix C.

[9] Performance and fairness metrics have been introduced in Section 4.1.

Method	Selection Criteria DTO	P	P@F+5%	P@F+10%	F	F@P−5%	F@P−10%	AUC↑
INLP (Ravfogel et al., 2020)	41.9	**41.9**	52.6	52.6	70.2	41.9	41.9	39.8
ADV (Li et al., 2018)	39.0	44.6	43.3	41.8	49.4	41.2	**41.2**	43.6
DADV (Han et al., 2021)	37.9	44.7	41.0	40.5	**39.9**	**40.4**	41.9	**44.5**
A-ADV (Han et al., 2022b)	**36.9**	45.4	**39.5**	**39.0**	62.1	43.8	42.8	44.0

Table 3: DTO scores of selected models over the BIOS dataset (smaller is better), based on the distances from mean performance and fairness to (1,1) over the test set. Models are selected based on the criterion listed for each column over the development set. The final column is the AUC, which does not involve model selection. **Bold** = the best score per column. See Appendix B for the full results.

performance within a fairness threshold of 10% improvement (**P@F+10%**); (5) maximum fairness (**F**); (6) maximum fairness within a performance trade-off threshold of 5% (**F@P−5%**); and (7) maximum fairness within a performance trade-off threshold of 10% (**F@P−10%**). Criteria (3), (4), (6), and (7) are constrained selections as discussed in Section 4.3, where we select the model with the highest fairness/performance within 5/10% of performance trade-off/fairness improvement relative to the STANDARD model. Taking F@P−10% as an example, the model with the highest fairness is selected within 10% performance trade-off over the vanilla model performance (i.e., with performance greater than 72% (82% − 10%)). Similarly, P@F+5% selects the model with highest performance subject to at least 5% fairness improvement over the vanilla model ($63\% = 58\% + 5\%$).

It can be seen that each of the four methods is the best for at least one selection criteria, as a stark illustration of our claim about model selection criteria biasing any possible conclusions about which method is best. For example, INLP and ADV are the best methods with respect to selection criteria **P** and **F@P−10%**, respectively.

On the contrary, our proposed AUC-PFC score (in the final column of Table 3) is unaffected by model selection, and reflects the overall trade-off of a method. Consistent with the trend in Figure 3b, AUC scores in Table 3 are smaller for worse-performing methods, e.g., INLP, and larger for better-performing methods such as A-ADV and DADV. Moreover, as the trade-off curves for A-ADV and DADV overlap one another (see Figure 3b), it is hard to pick a winner visually, let alone make quantitative comparisons. By using the AUC-PFC metric, we can conclude that overall, DADV is slightly better than A-ADV over this dataset.

Discussion The current DTO calculation assumes that users have no preference for performance over fairness or vice versa, where in practice it is possible that the choice of the fairness metric could be influenced by task-specific goals or the relative importance of fairness. Such problems have been widely studied in the literature on multi-objective learning, and a typical line of work is weighted generalized mean, which incorporates additional weight parameters in the generalized mean framework to reflect the importance or preference of each objective.

5 Conclusion

We have discussed the current practice in evaluation, model selection, and method comparison in the fairness literature, and shown how current practice in experimental fairness lacks rigour and consistency. We made recommendations for selecting a fairness evaluation metric, and introduced a new metric for measuring the overall performance–fairness trade-off of a method.

Acknowledgements

We thank Lea Frermann, Aili Shen, and Shivashankar Subramanian for their discussions and inputs. We thank the anonymous reviewers for their helpful feedback and suggestions. This work was funded by the Australian Research Council, Discovery grant DP200102519.

Limitations

This paper focuses on the notion of group fairness, under the assumption that each individual belongs to a particular demographic group. One limitation of methods in this space is that the demographic attributes must be observed (for the development and test data, at least) in order to evaluate fairness.

We only investigate the proposed evaluation aggregation framework in a classification setting. However, our framework is naturally generalizable to other tasks with discrete outcomes, such as generation and sequential tagging. Moreover, in terms of continuous labels, such as regression, one can skip the class-wise aggregation.

Ethical Considerations

This work focuses on current practice in fairness evaluation and method comparison. Our proposed "checklist" recommendations are specific to the fairness literature and complement existing frameworks, to encourage future research to think carefully about harms and what type of fairness is appropriate.

Demographics are assumed to be available only for evaluation purposes and are not used for model training or inference. We only use attributes that the user has self-identified in our experiments. All data and models in this study are publicly available and used under strict ethical guidelines.

References

Pinkesh Badjatiya, Manish Gupta, and Vasudeva Varma. 2019. Stereotypical bias removal for hate speech detection task using knowledge-based generalizations. In *The World Wide Web Conference*, pages 49–59.

Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. *Fairness and Machine Learning*. `http://www.fairmlbook.org`.

Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. 2021. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1):3–44.

Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. 2020. Fairlearn: A toolkit for assessing and improving fairness in ai. *Microsoft, Tech. Rep. MSR-TR-2020-32*.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, pages 4349–4357.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Sunipa Dev and Jeff Phillips. 2019. Attenuating bias in word vectors. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 879–887. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mark Díaz, Isaac Johnson, Amanda Lazar, Anne Marie Piper, and Darren Gergle. 2018. Addressing age-related bias in sentiment analysis. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2185–2194, Hong Kong, China. Association for Computational Linguistics.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21.

Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1926–1940, Online. Association for Computational Linguistics.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2022a. Balancing out bias: Achieving fairness through balanced training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11335–11350, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2022b. Towards equal opportunity fairness through adversarial learning. *arXiv preprint arXiv:2203.06317*.

Xudong Han, Aili Shen, Yitong Li, Lea Frermann, Timothy Baldwin, and Trevor Cohn. 2022c. FairLib: A unified framework for assessing and improving fairness. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 60–71, Abu Dhabi, UAE. Association for Computational Linguistics.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in Neural Information Processing Systems*, 29:3315–3323.

Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR.

Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. 2018. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR.

Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. 2019. An empirical study of rich subgroup fairness for machine learning. *Proceedings of the Conference on Fairness, Accountability, and Transparency*.

Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed Chi. 2020. Fairness without demographics through adversarially reweighted learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 728–740.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30.

Kristian Lum, Yunfeng Zhang, and Amanda Bower. 2022. De-biasing “bias” measurement. In *2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT ’22)*, Seoul, Republic of Korea. ACM.

R Timothy Marler and Jasbir S Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256.

John Rawls. 2001. *Justice as Fairness: A Restatement*. Harvard University Press.

Yuji Roh, Kangwook Lee, Steven Euijong Whang, and Changho Suh. 2021. Fairbatch: Batch selection for model fairness. In *Proceedings of the 9th International Conference on Learning Representations*.

Alexey Romanov, Maria De-Arteaga, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, Anna Rumshisky, and Adam Kalai. 2019. What’s in a name? reducing bias in bios without access to protected attributes. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4187–4195.

M Ye Salukvadze. 1971. Concerning optimization of vector functionals. I. Programming of optimal trajectories. *Avtomat. i Telemekh*, 8:5–15.

Aili Shen, Xudong Han, Trevor Cohn, Timothy Baldwin, and Lea Frermann. 2022a. Does representational fairness imply empirical fairness? In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 81–95.

Aili Shen, Xudong Han, Trevor Cohn, Timothy Baldwin, and Lea Frermann. 2022b. Optimising equal opportunity fairness in model training. In *Proceedings of* *the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4073–4084, Seattle, United States. Association for Computational Linguistics.

Shivashankar Subramanian, Xudong Han, Timothy Baldwin, Trevor Cohn, and Lea Frermann. 2021. Evaluating debiasing techniques for intersectional biases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2492–2498, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5310–5319.

Forest Yang, Mouhamadou Cisse, and Sanmi Koyejo. 2020. Fairness with overlapping groups; a probabilistic perspective. In *Advances in Neural Information Processing Systems*, volume 33, pages 4067–4078. Curran Associates, Inc.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. 2017. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970. PMLR.

Han Zhao, Amanda Coston, Tameem Adel, and Geoffrey J Gordon. 2019. Conditional learning of fair representations. In *International Conference on Learning Representations*.

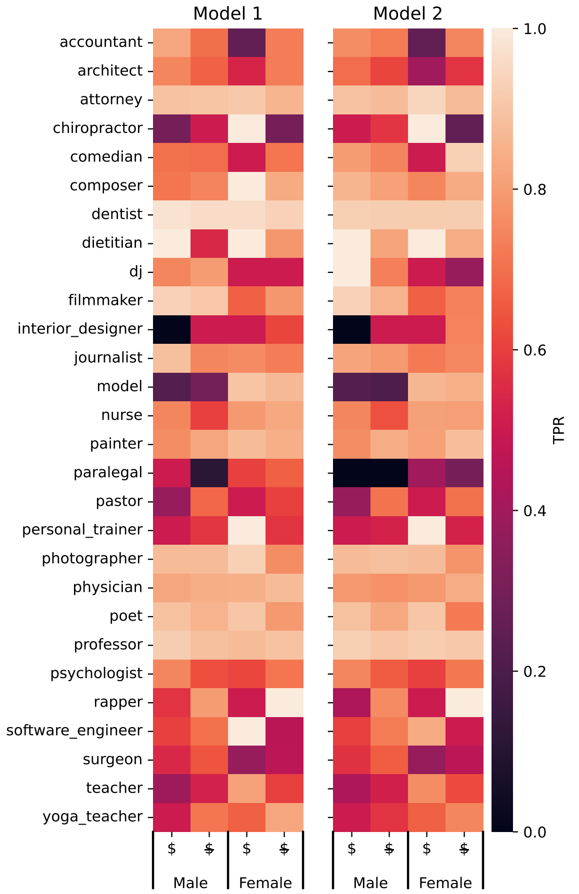


Figure 6: True positive rate (TPR) evaluation results over a biography classification dataset broken down by author demographics and profession classes. $ and ~~$~~ denote the economic status (high vs. low, respectively).

Profession	**Total**	**Male**		**Female**	
		$	**~~$~~**	**$**	**~~$~~**
professor	21715	46	9	37	7
physician	7581	42	8	41	8
attorney	6011	51	10	33	6
photographer	4398	53	11	30	6
journalist	3676	41	9	41	9
nurse	3510	8	1	76	15
psychologist	3280	31	6	52	11
teacher	2946	35	6	49	10
dentist	2682	52	11	30	6
surgeon	2465	73	12	13	2
architect	1891	64	12	21	3
painter	1408	47	9	36	8
model	1362	15	2	70	13
poet	1295	46	7	39	8
software engineer	1289	70	14	14	2
filmmaker	1225	56	10	29	6
composer	1045	70	14	14	2
accountant	1012	55	9	29	6
dietitian	730	5	1	82	12
comedian	499	69	9	19	3
chiropractor	474	62	14	21	3
pastor	453	59	15	23	4
paralegal	330	12	3	70	15
yoga teacher	305	13	3	71	12
interior designer	267	16	4	67	12
personal trainer	264	41	10	42	7
DJ	244	71	16	11	2
rapper	221	75	15	9	1
Total	72578	9	45	7	39

Table 4: Training set distribution of the Bios dataset. For each profession, the table shows the number of individuals and the breakdown across demographics as a percentage. $ and ~~$~~ denote the economic status (high vs. low, respectively).

A Full Disaggregated Results

Figure 6 depicts the full TPR scores for two real-world models over a profession classification dataset, stratified across 4 protected attributes (male vs. female and high vs. low economic status) from Figure 1. Specifically, the two models are both trained naively without debiasing. They share the same hyperparameter settings and random seed, except that Model 1 and Model 2 are the 9th and 5th epochs, respectively. For professions such as Professor, there is little discernible difference either between the two models or across different combinations of protected attributes. For DJ, on the other hand, Model 1 appears to be reasonably fair w.r.t. economic status but biased for binary gender, whereas Model 2 is biased across both protected attributes but appears to have the higher overall TPR. Finally, with Paralegal, Model 2 appears to be fairer w.r.t. both economic status and binary gender but perform substantially worse than the more biased Model 1 in terms of the individual TPR scores for every combination of protected attributes. So it is hard to tell which model is fairer or "better" out of the two, without aggregation.

A.1 Dataset: Bios

All experiments are based on a biography classification dataset (De-Arteaga et al., 2019; Ravfogel et al., 2020), where biographies were scraped from the web, and annotated for the protected attribute of binary gender and target label of 28 profession classes.

Besides the binary gender attribute, we additionally consider economic status as a second protected attribute. Subramanian et al. (2021) semi-automatically labeled economic status based on the individual's home country (wealthy vs. rest of world), as geotagged from the first sentence of the biography. For bias evaluation and mitigation, we consider the intersectional groups, i.e., the Cartesian product of the two protected attributes, leading to 4 intersectional classes: female–wealthy, female–rest, male–wealthy, and male–rest.

Since the data is not directly available, in order to construct the dataset, we use the scraping scripts of Ravfogel et al. (2020), leading to a dataset with 396k biographies.[10] Following Ravfogel et al. (2020), we randomly split the dataset into train (65%), dev (10%), and test (25%).

The augmentation for economic attributes follows previous work (Subramanian et al., 2021), which results in approximately 30% of instances that are labelled with both protected attributes.

Table 4 shows the target label distribution and protected attribute distribution.

A.2 Experimental Details

This work focuses on evaluation and model comparison in the fairness literature. Instead of training models from scratch, we use existing checkpoints from previous work (Han et al., 2022c), which are publicly available online.[11] Please refer to the original work (Han et al., 2022c) for experimental details.

A.3 Subset Confusion Matrices

Figure 7 presents the confusion matrices of all 4 subgroups. For each confusion matrix, the i-th row and j-th column entry indicates the number of samples which have the true label of the i-th class and predicted label of the j-th class. Since the distributions of classes within each group can be highly imbalanced, without further normalization and aggregation, it is difficult to draw any conclusion by just observing the number of samples in each cell.

A.4 Fairness Reproducibility

So far, we have listed critical factors underlying the choice of fairness metric, and provided recommendations for metric selection. However, we acknowledge that, in actual applications, the selection should be made in a domain-specific manner in close consultation with stakeholders or policymakers. In practice, countless types of fairness evaluation metrics could be derived from different combinations of aggregation methods.

Instead of reporting all possible fairness metrics, we suggest providing a set of confusion matrices for classification tasks, as it can form the basis of calculating a large number of metrics, including PPR, TPR, TNR, accuracy, and F-measure. The other key advantage of reporting confusion matrices is that the number of reported values is generally much smaller than the model or dataset size. Given a C-class classification dataset with G distinct protected groups, the combined size of the confusion matrices is G $\times$ C^2 (one confusion matrix per group). Taking the BIOS dataset as an example, the sizes of the confusion matrices, test dataset, and model parameters (for a BERT-base classifier (Devlin et al., 2019)) are approximately 3×10^3, 4×10^4, and 1×10^8, respectively.

B Full Results of Case Studies

Table 5 shows the experimental results for both the test and development sets.

C AUC-PFC Extension

C.1 Weighted DTO

On the one hand, as suggested by Marler and Arora (2004), if fairness and performance have different scales, the Euclidean distance is not a suitable mathematical representation of closeness, resulting in worse approximation of Pareto optimality and efficiency. Therefore, the scales of performance and fairness should be normalized.

C.2 Selection of Utopia Point

Typically, most debiasing methods will share the same maximum performance, which is the performance of the vanilla model (corresponding to a hyperparameter setting where the debiasing method does nothing.) Accordingly this is a sensible choice for the performance of the Utopia point, as we have proposed for model selection. In terms of the calculation of areas of integration, moving the Utopia point to $(1, 1)$ has little effect, simply adding a constant triangular region which is identical for all methods, and thus irrelevant for model comparison. As such, it makes no difference whether we use 1 or the maximum-achieved model performance when comparing models based on AUC-PFC.

Distance to Arbitrary Ideal Point: Compared to the default value of DTO, moving the utopia point to the right (e.g., the $(1, 1)$ point) prioritizes methods with higher performance.

---

[10] There are slight discrepancies in the dataset composition due to data attrition: the original dataset (De-Arteaga et al., 2019) had 399k instances, while 393k were collected by Ravfogel et al. (2020).

[11] Bios_both at `https://github.com/HanXudong/fairlib/tree/main/analysis/results`

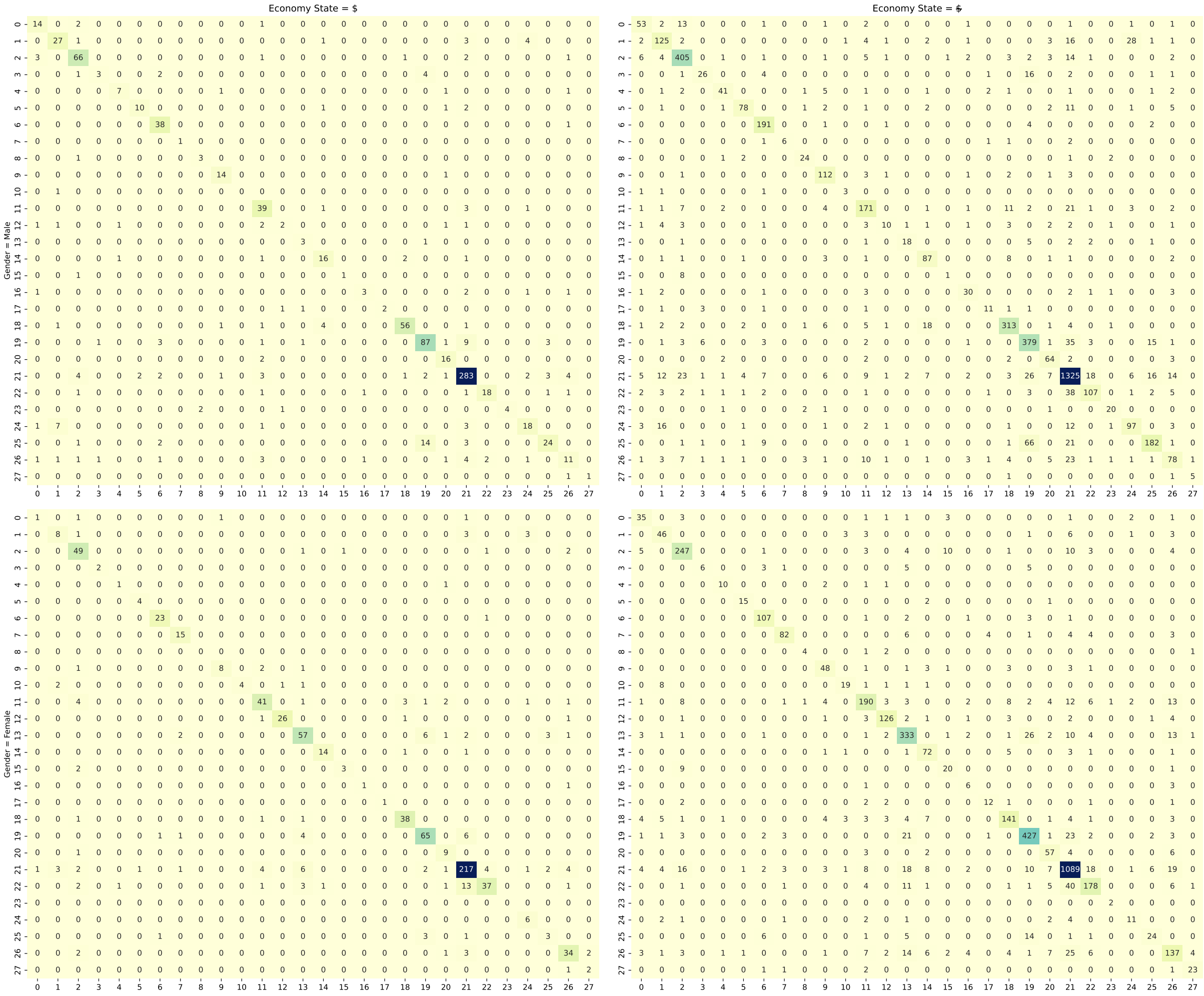


Figure 7: Confusion matrices of subgroups. Following Figure 6, confusion matrices are based on the predictions of Model 1, and class labels 0 to 27 are in the same order as the 28 professions in Figure 6.

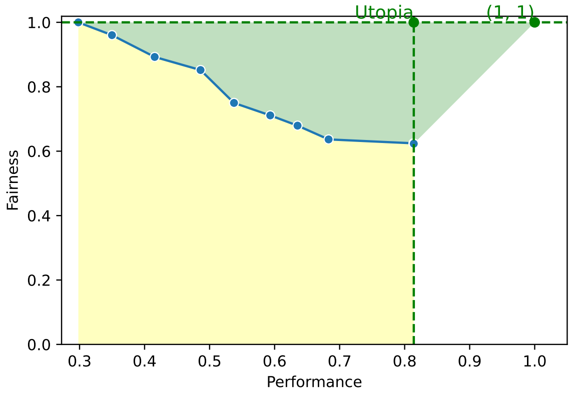


Figure 8: Integration of DTO with respect to an arbitrary ideal point, which is (1, 1) in this example.

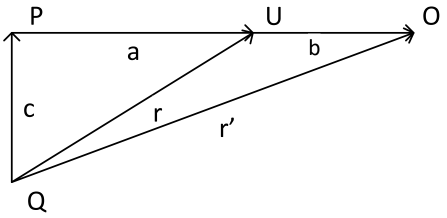


Figure 9: The influence of using different optimal points. Uppercase (Q, P, U, and O) and lowercase (c, a, b, r, and r') characters represent points and Euclidean distances between points.

As shown in Figure 9, without loss of generality, let

- $Q = (0, 0)$ denote the candidate point;
- $U = (c, a)$ denote the Utopia point, where c is the fairness distance from Q to the maximum fairness (which is 1), and a is the performance distance from Q to the maximum performance (which is 0.82 is Figure 9); and
- $O = (c, a + b)$ denote the arbitrary model,

		Test Set			Development Set		
Selection	**Method**	**Performance**	**Fairness**	**DTO**	**Performance**	**Fairness**	**DTO**
DTO	INLP	81.4 ± 0.0	62.4 ± 0.0	41.9	80.3 ± 0.0	54.4 ± 0.0	49.7
	Adv	64.6 ± 4.5	83.7 ± 1.1	39.0	63.9 ± 4.4	79.9 ± 3.0	41.3
	DAdv	68.1 ± 5.5	79.5 ± 6.8	37.9	67.6 ± 5.5	75.5 ± 6.7	40.7
	A-Adv	69.7 ± 4.9	78.8 ± 7.7	36.9	69.3 ± 4.7	75.2 ± 5.4	39.5
P	INLP	81.4 ± 0.0	62.4 ± 0.0	41.9	80.3 ± 0.0	54.4 ± 0.0	49.7
	Adv	81.5 ± 0.2	59.5 ± 1.7	44.6	80.7 ± 0.2	55.2 ± 1.3	48.8
	DAdv	81.4 ± 0.3	59.3 ± 1.7	44.7	80.6 ± 0.3	54.9 ± 1.7	49.1
	A-Adv	81.3 ± 0.3	58.6 ± 2.0	45.4	80.6 ± 0.3	54.4 ± 1.6	49.6
P@F+5%	INLP	53.8 ± 0.0	75.0 ± 0.0	52.6	53.8 ± 0.0	74.4 ± 0.0	52.8
	Adv	71.5 ± 5.7	67.3 ± 5.4	43.3	71.3 ± 5.8	66.2 ± 4.2	44.4
	DAdv	73.4 ± 3.8	68.8 ± 4.9	41.0	73.0 ± 3.7	66.0 ± 3.7	43.5
	A-Adv	70.4 ± 5.7	73.9 ± 8.9	39.5	70.0 ± 5.8	67.4 ± 6.6	44.3
P@F+10%	INLP	53.8 ± 0.0	75.0 ± 0.0	52.6	53.8 ± 0.0	74.4 ± 0.0	52.8
	Adv	68.8 ± 4.8	72.1 ± 5.6	41.8	68.2 ± 4.8	70.3 ± 6.5	43.6
	DAdv	69.7 ± 2.9	73.2 ± 6.8	40.5	69.2 ± 2.7	70.6 ± 4.0	42.6
	A-Adv	70.0 ± 3.3	75.0 ± 3.9	39.0	69.4 ± 3.4	70.8 ± 2.1	42.2
F	INLP	29.8 ± 0.0	100.0 ± 0.0	70.2	29.9 ± 0.0	86.6 ± 0.0	71.4
	Adv	51.6 ± 16.5	90.2 ± 9.3	49.4	51.2 ± 16.2	81.0 ± 6.0	52.4
	DAdv	61.8 ± 3.7	88.6 ± 3.7	39.9	61.2 ± 3.5	82.2 ± 2.5	42.7
	A-Adv	37.9 ± 9.1	99.0 ± 1.2	62.1	37.6 ± 8.8	86.5 ± 0.2	63.8
F@P−5%	INLP	81.4 ± 0.0	62.4 ± 0.0	41.9	80.3 ± 0.0	54.4 ± 0.0	49.7
	Adv	79.1 ± 1.1	64.5 ± 1.3	41.2	78.5 ± 0.9	58.7 ± 2.4	46.5
	DAdv	80.4 ± 0.5	64.7 ± 1.4	40.4	79.9 ± 0.5	57.4 ± 1.1	47.1
	A-Adv	79.9 ± 2.2	61.1 ± 2.8	43.8	79.1 ± 2.2	58.1 ± 2.8	46.8
F@P−10%	INLP	81.4 ± 0.0	62.4 ± 0.0	41.9	80.3 ± 0.0	54.4 ± 0.0	49.7
	Adv	79.1 ± 1.1	64.5 ± 1.3	41.2	78.5 ± 0.9	58.7 ± 2.4	46.5
	DAdv	74.2 ± 3.2	67.0 ± 2.4	41.9	73.7 ± 3.3	64.2 ± 1.1	44.4
	A-Adv	74.9 ± 5.1	65.4 ± 6.9	42.8	74.3 ± 5.3	64.2 ± 5.4	44.1

Table 5: Evaluation results ± standard deviation (%) of selected models over the Bios dataset. DTO scores are the distance from mean performance and fairness to (1,1) over the test set.

where $b > 0$, e.g., $b = (1 - 0.82)$ for the running example.

Before discussing the influence of the optimum point selection, recall that the magnitude of vector sum, $|\vec{v}| = |\vec{v}_1 + \vec{v}_2|$ is:

$$|\vec{v}| = \sqrt{|\vec{v}_1|^2 + |\vec{v}_2|^2 + 2|\vec{v}_1||\vec{v}_2|\cos\alpha},$$

where α is the angle between $\vec{v}_1$ and $\vec{v}_2$.

Let QU denote the vector from candidate model Q to the Utopia point U, the DTO based on the Utopia point is the $r = \sqrt{a^2 + c^2}$.

When calculating DTO based on the arbitrary optimum point O, $r' = |QU + UO|$, which can be shown as:

$$r' = \sqrt{r^2 + b^2 + 2rb\cos\alpha'},$$

where α' is the angle between QU and UO, and is equivalent to $\angle PUQ$. Furthermore, as discussed in Section 4.4, given a trade-off curve, the DTO is a function of $\angle PUQ$, i.e., the green shaded area is $\int_0^{\pi/2} \text{DTO}(\angle PUQ) d\angle PUQ$.

Lemma C.1. *Let Q_1 and Q_2 be two models with the same DTO score ($r_1 = r_2$), r'_1 and r'_2 be the DTO to the new Utopia point O. If the performance of Q_1 is worse than Q_2, then $r'_1 > r'_2$.*

Proof. Assuming that $r'_1 > r'_2$,

$$\begin{gathered} |Q_1U + UO| > |Q_2U + UO| \\ |Q_1U + UO|^2 > |Q_2U + UO|^2 \\ 2r_1 b\cos\angle PUQ_1 > 2r_2 b\cos\angle PUQ_2 \end{gathered} \tag{1}$$

Since $\angle PUQ \leq \pi/2, \forall Q$, and $r_1 = r_2$,

$$a_1 = r_1 \cos\angle PUQ_1 > a_2 = r_2 \cos\angle PUQ_2,$$

where a_1 and a_2 are the performance distances from Q_1 and Q_2 to the maximum performance, respectively.

□

CHARD: Clinical Health-Aware Reasoning Across Dimensions for Text Generation Models

Steven Y. Feng[1]*, **Vivek Khetan**[2], **Bogdan Sacaleanu**[2], **Anatole Gershman**[3], **Eduard Hovy**[3]
[1]Stanford University, [2]Accenture Labs, SF, [3]Carnegie Mellon University
`syfeng@stanford.edu`
`{vivek.a.khetan,bogdan.e.sacaleanu}@accenture.com`
`{anatoleg,hovy}@cs.cmu.edu`

Abstract

We motivate and introduce **CHARD**: Clinical Health-Aware Reasoning across Dimensions, to investigate the capability of text generation models to act as implicit clinical knowledge bases and generate free-flow textual explanations about various health-related conditions across several dimensions. We collect and present an associated dataset, CHARDAT, consisting of explanations about 52 health conditions across three clinical dimensions. We conduct extensive experiments using BART and T5 along with data augmentation, and perform automatic, human, and qualitative analyses. We show that while our models can perform decently, **CHARD** is very challenging with strong potential for further exploration.

1 Introduction

Pretrained language models (PLM) have seen increasing popularity for NLP tasks and applications, including text generation. Researchers have become interested in the extent to which PLMs can: 1) act as knowledge bases, 2) reason like humans.

Rather than using external databases, exposure to large amounts of data during training combined with their large number of parameters, has given PLMs the ability to store knowledge that can be extracted through effective probing strategies such as text infilling (Donahue et al., 2020), prompting (Liu et al., 2021), and QA (Jiang et al., 2021). PLMs imitate a more high-level information store, allowing for greater abstractness, flexibility, and generalizability. They are also able to better exploit contextual information than simple retrieval.

Studies have also shown that as PLMs scale up, they have have emergent abilities (Wei et al., 2022a), including reasoning. There has been increasing attention on their commonsense reasoning through works like COMET (Bosselut et al., 2019). However, studies show that even large PLMs struggle with commonsense tasks that humans can reason through very easily (Talmor et al., 2020). There are works that investigate more complicated reasoning tasks, e.g. arithmetic and symbolic reasoning (Wei et al., 2022b). PLMs inherently have some extent of reasoning capability, and many more complex reasoning tasks are easier to carry out over abstract PLM embedding space.

Template	**Full Text with Explanation**
A person with *Costochondritis* has a/an *exercise risk factor* because/since/as *{explanation}*	A person with Costochondritis has an exercise risk factor because *costochondritis can be aggravated by any activity that places stress on your chest area.*
A person with *gout* has a/an *lose weight prevention* because/since/as *{explanation}*	A person with gout has a lose weight prevention because *losing weight can lower uric acid levels in your body and significantly reduce the chance of gout attacks.*
A person with *rheumatoid* has a/an *therapy treatment* because/since/as *{explanation}*	A person with rheumatoid has a therapy treatment because *physiotherapy helps rheumatoid patients with pain control, reducing inflammation and joint stiffness and to return to the normal activities of daily living or sports.*

Table 1: Examples of **CHARD** templates with explanations (from CHARDAT). The human was asked to write the entire output text (not just the explanation) by infilling the template.

In this paper, we are interested in the intersection of these areas. Can PLMs act as knowledge bases and also reliably reason using their own knowledge? We investigate whether PLMs can learn and reason through health-related knowledge. Work on generation-based reasoning for health has been limited, with most prior work exploring retrieval-based methods. Generation-based reasoning is more difficult, as such a specialized domain contains esoteric information not prevalent in the PLM's training data, and involves a higher degree of specialized reasoning to handle domain-specific problems.

Healthcare is an important domain that deals with human lives. It is a large application area for machine learning and NLP. The need for automation in healthcare rises, as countless studies show that healthcare workers are overworked and burned out, especially recently due to the COVID-19 pandemic (Portoghese et al., 2014; Brophy et al.,

---

[*] Work done while at CMU.

Code: `https://github.com/styfeng/CHARD`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 313–327
May 2-6, 2023 ©2023 Association for Computational Linguistics

2021; Couarraze et al., 2021). Further, healthcare resources will continue to be strained as the baby boomer generation ages (Canizares et al., 2016).

To this end, we propose **CHARD**: Clinical Health-Aware Reasoning across Dimensions (§2.1). This task is designed to explore the capability of text generation models to act as implicit clinical knowledge bases and generate textual explanations about health-related conditions across several dimensions. The ultimate goal of **CHARD** is to eventually have a model that is knowledgeable and insightful across numerous clinical dimensions and reasoning pathways. For now, we focus on three relevant clinical dimensions using a template infilling approach, and collect an associated dataset, CHARDAT, which includes information for 52 health conditions across these dimensions (§2.2).

We perform extensive experiments on CHARDAT using two SOTA seq2seq models: BART (Lewis et al., 2020) and T5 (Raffel et al., 2020) (§3.1), with data augmentation using backtranslation (Sennrich et al., 2016) (§3.2,4.2). We benchmark our models through automatic, human, and qualitative analyses (§5). We show that our models show strong potential, but have room to improve, and that **CHARD** is highly challenging with room for additional exploration. Lastly, we discuss several potential directions for improvement (§6).

2 Task and Dataset

2.1 The CHARD Task

Our task, **CHARD**: Clinical Health-Aware Reasoning across Dimensions, investigates the capability of text generation models to produce clinical explanations about various health conditions across several clinical dimensions (DIM). Essentially, we assess how a PLM can be used as and reason through an implicit clinical knowledge base.

We focus on three DIM: **risk factors** (RF), **treatment** (TREAT), and **prevention** (PREV), as they are important and relevant in the context of health. A risk factor refers to something that increases the chance of developing a condition. For cancer, some examples are age, family history, and smoking. Treatment refers to something that helps treat or cure a condition. For migraines, some examples are medication, stress management, and meditation. Prevention refers to strategies to stop or lower the chance of getting a condition. For diabetes, some examples are a healthy diet and regular exercise.

As an initial approach to **CHARD**, we use a template infilling formulation, where given an input template that lays out the structure of the desired explanation, the model's goal is to generate a complete explanation of how the particular DIM attribute relates to the given condition. In particular, the templates end with an {explanation} span that the models fill in by explaining the appropriate relationship. Some examples are in Table 1.

2.2 CHARDAT Dataset

Collection Process: We collect a dataset for our task called CHARDAT (where DAT is short for data). We collect data across the three DIM for 52 health conditions, listed in Appendix A. This is a manually curated list of health conditions which range from common conditions such as migraine and acne to rare conditions such as Lyme disease and Paget–Schroetter. The conditions were also selected by volume of online activity (e.g. number of active subreddit users), treatable vs. chronic conditions, and whether a condition can be self-diagnosed or not. This allows us to assess **CHARD** across a variety of conditions.

For each DIM, we manually collect an exhaustive list of DIM-related attributes (e.g. risk factors) for each condition. By *attribute*, we refer to a particular example of that DIM (e.g. "obesity"). This was accomplished by searching through reliable and reputable medically-reviewed sources such as MayoClinic, CDC, WebMD, and Healthline.

We collect the final text (with explanations) using Amazon Mechanical Turk (AMT). We ask approved AMT workers (with strong qualifications and approval ratings on healthcare-related tasks) to write factually accurate, informative, and relatively concise passages given a particular condition and DIM attribute template (per HIT), while encouraging them to consult the aforementioned health resources. Three separate annotation studies (one per DIM) with strict quality control were conducted to collect an annotation per example.[1] Annotations were regularly verified by authors, and a large subset of CHARDAT was manually examined for medical accuracy. More details are in Appendix B. Some examples from CHARDAT are in Table 1.

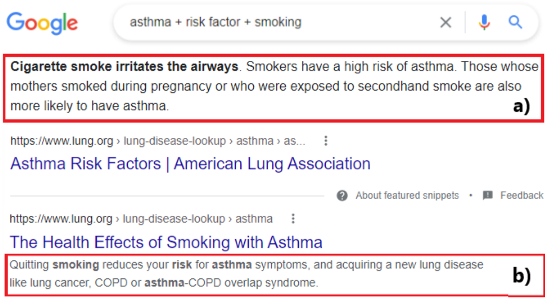
Splits and Statistics: We split CHARDAT by DIM into train, val, and test splits of $\approx$ 70%/15%/15%, and combine the individual splits per DIM to form the final splits called CHAR-

---

[1] Explanations for **CHARD** are typically quite standardized, and additional annotations were repetitive. Differences are mainly in language, so we instead opt for paraphrasing data augmentation techniques such as backtranslation (§3.2).

Dataset Stats	Train	Val	Test (seen/unseen)
# conditions = 52	44	39	41 (37/4)
RF = 52	44	26	26 (22/4)
TREAT = 52	43	21	20 (16/4)
PREV = 44	35	11	21 (17/4)
# sentences = 937	655	141	141 (70/71)
RF = 457	319	69	69 (32/37)
TREAT = 297	207	45	45 (20/25)
PREV = 183	129	27	27 (18/9)
Avg length = 36.2	37.7	36.1	35 (35.9/34.2)

Table 2: CHARDAT statistics. Differing #s by DIM are because there are more risk factors for most conditions, and some do not have prevention strategies. Length is in words.



asthma + risk factor + smoking

Cigarette smoke irritates the airways. Smokers have a high risk of asthma. Those whose mothers smoked during pregnancy or who were exposed to secondhand smoke are also more likely to have asthma.

https://www.lung.org › lung-disease-lookup › asthma › as...

Asthma Risk Factors | American Lung Association

https://www.lung.org › lung-disease-lookup › asthma

The Health Effects of Smoking with Asthma

Quitting **smoking** reduces your **risk** for **asthma** symptoms, and acquiring a new lung disease like lung cancer, COPD or **asthma**-COPD overlap syndrome.

Figure 1: An example of the Google search results for the query {*asthma* + *risk factor* + *smoking*} highlighting: a) the *featured snippet*, b) the text below the first link.

DAT$_{tr}$, CHARDAT$_{val}$, and CHARDAT$_{test}$, respectively. The individual DIM splits are called DIM$_{tr}$, DIM$_{val}$, and DIM$_{test}$, where DIM is a short-form of the particular dimension: RF, TREAT, or PREV. The individual dimension subsets of CHARDAT are called CHARDAT$_{DIM}$.

For each DIM's test split, we ensure that approximately half consist of examples from conditions entirely unseen during training for that DIM, called DIM$_{test-unseen}$. This is to assess whether the model can generalize to unseen conditions. The other half contains examples from conditions seen during training called DIM$_{test-seen}$, but the specific condition and DIM attribute combination was unseen. The combined halves (across DIM) are called CHARDAT$_{test-unseen}$ and CHARDAT$_{test-seen}$. We do the same for the val split to ensure consistency for model selection purposes. CHARDAT statistics are in Table 2.

3 Methodology

3.1 Models

BART and T5: We experiment using two pretrained seq2seq models: BART and T5 (both base and large versions). These are suitable for our task formulation (template infilling). T5 (Raffel et al., 2020) has strong multitask pretraining. BART (Lewis et al., 2020) is trained to reconstruct original text from noised text (as a denoising autoencoder). We use their HuggingFace codebases.

Retrieval Baseline (Retr): We use a retrieval-based approach as a baseline. We manually query Google using {*condition* + DIM + DIM *attribute*}, e.g. {*asthma* + *risk factor* + *smoking*}, and extract either the *featured snippet* at the top of the results page, or the text below the first link if there is no featured snippet. If the featured snippet is a list or table, we manually concatenate the items into a single piece of text. An example is in Figure 1.

The extracted text approximates an explanation, which we then concatenate to the first part of the associated template to form the final text, e.g. *A person with asthma has a/an smoking risk factor because/since/as* ***{retrieved explanation}***. Retr leverages Google's strong search and summarization capabilities, serving as a useful baseline. Further, Google Search is an evolving baseline that continually challenges our **CHARD** models.[2]

3.2 Data Augmentation (DA)

Since CHARDAT is relatively small, which is mainly a function of our task and domain, i.e. there are a limited number of non-obscure medical conditions and associated DIM attributes, we hypothesize that data augmentation (DA) techniques (Feng et al., 2021a, 2020) may be useful.

As noted by Feng et al. (2021a), text generation and specialized domains (such as healthcare) both present several challenges for DA. In our case, many explanations contain clinical or health jargon which makes techniques that leverage lexical databases such as WordNet, e.g. synonym replacement (Feng et al., 2020), challenging or impossible.

We decide to use backtranslation (BT) (Sennrich et al., 2016) to augment examples in CHARDAT$_{tr}$, a popular and easy DA technique which translates a sentence into another language and back to the original language.[3] This usually results in a slightly altered version (paraphrase) of the original text. BT is effective here as healthcare-related terms are preserved relatively well, and the resulting paraphrased explanation remains relatively intact.

We use UDA (Xie et al., 2020) for BT, which translates sentences from English to French, then back to English. UDA is a DA method that uses unsupervised data through consistency training on $(x, DA(x))$ pairs. An advantage of UDA's BT is that we can control for the degree of variation using

---

[2] We will release our current baseline data.

[3] This is sometimes referred to as *round-trip translation*.

Tmp	Text
0	A person with acne has an avoid irritants prevention because *using oily or irritating personal care products clog your pores causing acne.*
0.5	*if you use oily or irritant personal care products, you block pores and cause acne.*
0.7	*using oily or irritating personal care products, you block acne pores.*
0.9	*use oily and irritating disinfectant products freezing your pores to cause the Acne restructurs.*
0	A person with MultipleSclerosis has a stress management prevention because *stress is more likely to exacerbate the symptoms of MS and bring about a flare or relapse.*
0.5	*stress is more likely to exacerbate MS symptoms and lead to an outbreak or relapse*
0.7	*stress is more likely to exacerbate symptoms of MS and trigger a flare or relapse.*
0.9	*severe mourning problems occurred at Vancouver Hospital (Prince Edward Island), British Columbia. (...)*

Table 3: Examples of original (tmp=0) and BT text. The explanation portion (which is backtranslated) is italicized.

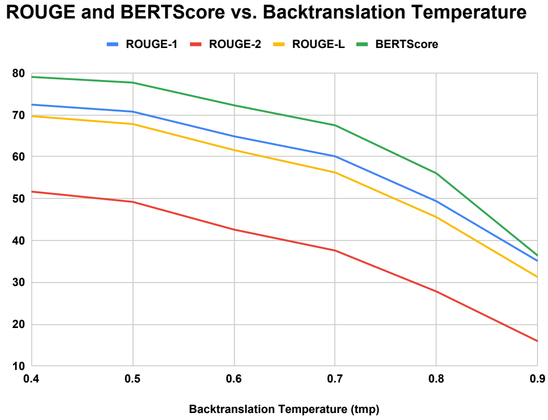


Figure 2: Graph showing how avg. ROUGE and BERTScore of BT vs. original text vary by BT tmp on CHARDAT$_{tr}$.

a *temperature (tmp)* parameter, where higher values (e.g. 0.9) result in more varied paraphrases. We only backtranslate the explanation portion of examples (concatenating them back to the preceding part) as we wish to keep the preceding part intact.

From the examples in Table 3, we can see that higher tmp typically results in more varied text, albeit with issues with content preservation and fluency. For the second example, the tmp=0.9 BT is completely unrelated to the original text. This is not entirely undesirable, as some noise may make our trained models more robust. From Figure 2, we see that the average ROUGE and BERTScore of backtranslated CHARDAT$_{tr}$ text compared to the original text decrease as tmp increases, as expected.

3.3 Evaluation Metrics

We use several standard text generation evaluation metrics including reference-based token and semantic comparison metrics used in works like Lin et al. (2020) such as ROUGE (Lin and Hovy, 2003), CIDEr (Vedantam et al., 2015), and SPICE (Anderson et al., 2016). SPICE translates text to semantic scene graphs and calculates an F-score over graph tuples. CIDEr captures sentence similarity, grammaticality, saliency, importance, and accuracy.[4]

We also use average word length (Len), BERTScore (Zhang et al., 2019), and Perplexity (PPL). BERTScore serves as a more semantic similarity measure by assessing BERT (Devlin et al., 2019) embeddings similarity between individual tokens. We multiply by 100 when reporting BERTScore. PPL approximately measures fluency, where lower values represent higher fluency. We use GPT-2 (Radford et al., 2019) for PPL. Higher is better for all metrics other than PPL and Len.

4 Experimental Setup

4.1 Model Finetuning and Generation

For the standard (non-augmented) **CHARD** models, we train and evaluate four versions of each on CHARDAT, CHARDAT$_{RF}$, CHARDAT$_{TREAT}$, and CHARDAT$_{PREV}$, respectively. The first of these is a combined model that learns to handle all three DIM at once depending on the DIM given at inference, while the latter three are models trained on each individual DIM. We predict that while the latter three may perform better on their particular DIM, the first model is more effective overall as it accomplishes our goal of having a single PLM that can store knowledge and reason through several DIM. It is thus more adaptable and generalizable.

For training the **CHARD** models, we keep most hyperparameters static, other than learning rate (LR) which is tuned per individual model. For each model, we select the epoch that corresponds to highest ROUGE-2 on CHARDAT$_{val}$, and decode using beam search. See Appendix C for more.

4.2 Data Augmentation Experiments

We try several backtranslation DA experiments.

2x DA with Different Tmp: Our first set of experiments involves 2x DA (backtranslating each CHARDAT$_{tr}$ explanation once, to 2x the original training data) using different BT tmp which we call BT-set: {0.4, 0.5, 0.6, 0.7, 0.8, 0.9}. We predict that the optimal tmp lies in the 0.6-0.7 range, as the text is modified to a reasonable degree.

Different DA Amounts (2x-10x): We also try further DA amounts: 3x, 4x, 5x, 7x, and 10x the original amount of training data. We explore whether the amount of augmentation affects performance, and hypothesize that performance will

---

[4] Matching metrics are sufficient as **CHARD** explanations are standardized (space for explanations is low) since our inputs present a particular condition and DIM attribute combo.

increase to a certain point and decline afterward. This is because the advantages of DA may taper off since the augmented data are variations of the original, and models may overfit past a point.

DA Amount Strategies (best-tmp vs. diff-tmp): We also investigate two strategies for selecting each successive iteration of augmented examples. The first is best-tmp, where all the augmented data comes from BT of the tmp that performed best for 2x DA (e.g. all from 0.7).[5]

The second is diff-tmp, where each successive iteration is the tmp that performed next best (e.g. 2x is the best tmp, 3x is additionally the second-best tmp, etc.). For the highest DA amounts (e.g. 10x), when the six tmp values in BT-set have been exhausted, we go back to the best tmp and repeat.

Base vs. Large Models: For the base models (BART-base and T5-base), we try all aforementioned tmp, DA amounts, and amount strategies. For the large models, we try the top three temperatures (for 2x DA) and amount strategy that performed best on the corresponding base model, and only 3x, 5x, 7x, and 10x DA amounts.

Note that BT tmp and DA amounts are both hyperparameters, so while we train models corresponding to different values of them, the final chosen models correspond to the ones that performed best on CHARDAT$_{val}$. We then use these final models to generate on CHARDAT$_{test}$. We report the results of the overall best models in §5.

4.3 Human Evaluation

We conduct human evaluation using AMT.[6] We ask two approved annotators (with strong qualifications and approval ratings on healthcare-related tasks) to each evaluate all 141 CHARDAT$_{test}$ examples. Our evaluation uses pairwise comparison of the outputs from two methods, split into three studies per DIM: RETR vs. best **CHARD** model, RETR vs. human, and human vs. best **CHARD** model.

We ask annotators to choose which amongst the two outputs (presented in a random order per example) has better 1) medical accuracy (MedAcc), 2) informativeness (Inform), and 3) readability (Read). Medical accuracy refers to which explanation is more clinically correct for the given DIM attribute and condition. Informativeness refers to which is more complete and explains in sufficient detail (including *why?*). Readability refers to which is more readable, which includes fluency (natural-sounding English) and conciseness/brevity (not overly long).

Metric	RETR	BART-base	BART-large	T5-base	T5-large
ROUGE-1	43.30	51.37	**51.54**	50.00	50.66
ROUGE-2	28.18	39.35	**40.27**	38.31	37.74
ROUGE-L	39.03	49.55	**49.88**	48.07	48.05
BLEU-1	32.20	31.20	28.40	32.60	**34.30**
BLEU-2	25.20	27.10	24.90	28.10	**29.20**
BLEU-3	21.50	24.70	22.90	25.50	**26.40**
BLEU-4	18.50	23.00	21.30	23.60	**24.30**
METEOR	**24.40**	22.10	22.10	21.80	22.10
CIDEr	2.36	8.56	6.90	8.71	**9.03**
SPICE	35.10	50.50	**50.70**	49.10	49.20
BERTScore	39.54	60.04	**60.78**	59.80	59.00
PPL	65.27	61.00	87.45	56.78	**52.52**
Len	52.80	20.16	18.60	21.35	22.23

Table 4: Avg. auto eval results of RETR and the best models (for BART and T5) on CHARDAT$_{test}$. Bold corresponds to best performance. For human text, PPL = 67.86, Len = 35.04.

Metric	test split (full)	test-seen	test-unseen
ROUGE-1	50.66	49.42	**51.93**
ROUGE-2	37.74	37.04	**38.35**
ROUGE-L	48.05	46.98	**49.12**
BLEU-1	34.30	33.50	**35.20**
BLEU-2	29.20	28.60	**29.90**
BLEU-3	26.40	25.90	**27.00**
BLEU-4	24.30	23.80	**24.80**
METEOR	22.10	21.60	**22.60**
CIDEr	9.03	**10.31**	7.59
SPICE	49.20	48.60	**49.80**
BERTScore	59.00	57.79	**60.18**
PPL	52.52	**51.06**	53.96
Len	22.23	22.73	21.73

Table 5: Avg. auto eval results of T5-large on CHARDAT$_{test}$ and the test-seen and test-unseen halves.

There are 3 choices for each evaluation aspect - O1: first is better, O2: second is better, O3: both are indistinguishable. To aggregate multiple annotations per example, we find the overall fraction of responses towards each outcome value.

5 Results and Analysis

We report automatic results on CHARDAT$_{test}$ of the best models (for BART-base, BART-large, T5-base, T5-large) trained on CHARDAT compared to RETR in Table 4. The best models are tmp=0.9 2x DA for BART (base and large), 5x DA with diff-tmp for T5-base, and tmp=0.6 2x DA for T5-large.

Our best overall **CHARD** model is T5-large based on automatic results and qualitative analysis. We break down results of T5-large on CHARDAT$_{test-seen}$ and CHARDAT$_{test-unseen}$ in Table 5. We show results of T5-large compared to T5-large$_{DIM}$ (models trained on the individual DIM) in Table 6. We conduct human evaluation with T5-large, and the results are in Table 7.

Graphs displaying models' ROUGE-2 on CHARDAT$_{val}$ for 2x DA across various BT tmp and different DA amounts can be found in Figures 3 and 4, respectively. Tables 8 and 9 contain qualitative examples, with more in Appendix E.

---

[5] This is possible because UDA uses sampling, so even for the same tmp, the backtranslations differ each time.

[6] See Appendix D for further human evaluation details.

	Risk Factors (RF$_{test}$)		Treatment (TREAT$_{test}$)		Prevention (PREV$_{test}$)	
Metric	T5-large	T5-large$_{RF}$	T5-large	T5-large$_{TREAT}$	T5-large	T5-large$_{PREV}$
ROUGE-1	52.74	**53.17**	**49.42**	47.38	47.73	**50.00**
ROUGE-2	40.52	**41.88**	36.12	**36.69**	33.00	**36.10**
ROUGE-L	50.40	**51.03**	**46.60**	45.54	44.43	**48.19**
BLEU-1	**34.80**	34.70	**31.10**	25.70	**30.80**	29.80
BLEU-2	30.40	**30.90**	**26.30**	22.40	**25.10**	25.00
BLEU-3	27.90	**28.60**	**23.90**	20.70	21.90	**22.00**
BLEU-4	26.10	**26.90**	**22.10**	19.30	**19.30**	**19.30**
METEOR	23.00	**24.20**	**20.70**	19.20	20.50	**20.80**
CIDEr	**13.50**	10.57	5.06	**5.98**	**5.88**	5.83
SPICE	49.90	**51.50**	**46.60**	45.30	46.50	**46.60**
BERTScore	60.40	**61.03**	**58.07**	56.60	56.90	**59.09**
PPL	**40.90**	58.92	**52.13**	86.15	**84.06**	110.66
Len	22.30	20.28	22.09	19.82	22.27	20.52

Table 6: Breakdown of the avg. auto eval results of T5-large compared to T5-large$_{DIM}$ models (trained on the three individual DIM) on the respective DIM subsets of CHARDAT$_{test}$. Bold corresponds to best performance per DIM.

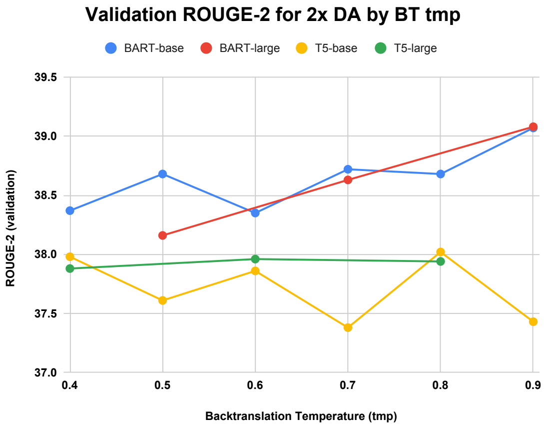


Figure 3: Graph showing how avg. ROUGE-2 on CHARDAT$_{val}$ varies by backtranslation temperature for 2x DA.

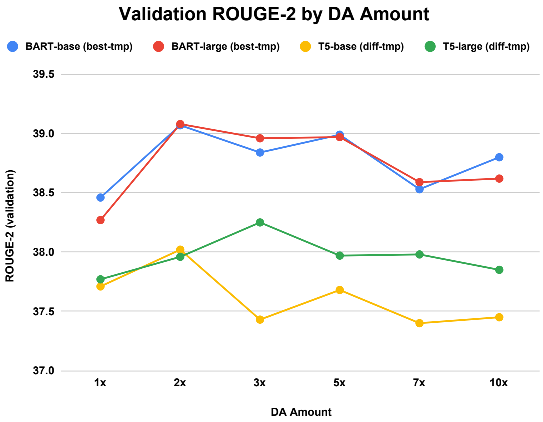


Figure 4: Graph showing how avg. ROUGE-2 on CHARDAT$_{val}$ varies by DA amount. 1x essentially refers to no DA.

Methods	Aspect	O1	O2	O3
RETR vs. Human	MedAcc	0.45	**0.53**	0.02
	Inform	0.45	**0.53**	0.02
	Read	0.22	**0.69**	0.09
Human vs. T5	MedAcc	**0.72**	0.24	0.04
	Inform	**0.72**	0.25	0.03
	Read	0.41	**0.49**	0.10
RETR vs. T5	MedAcc	**0.73**	0.25	0.02
	Inform	**0.73**	0.26	0.01
	Read	0.35	**0.62**	0.03

Table 7: Avg. human eval results on CHARDAT$_{test}$. O1: first method wins, O2: second method wins, O3: indistinguishable. Bold corresponds to higher fractional outcome. T5 refers to T5-large. See §4.3 and Appendix D for further details.

Method	**Text**
Input (1)	A person with Hyperhidrosis has a medication treatment because {explanation}
Human	A person with Hyperhidrosis has a medication treatment because anticholinergics medicines such as glycopyrrolate, oxybutynin, benztropine, propantheline etc are helping to prevent the stimulation of sweat glands and, thus, decreasing sweating.
RETR	A person with Hyperhidrosis has a medication treatment because the most common medications used to manage excessive sweating are anticholinergics, including but not limited to: Glycopyrrolate (Robinul). Oxybutynin (Ditropan). Benztropine. Propantheline bromide.
T5-L	A person with Hyperhidrosis has a medication treatment because medications such as ibuprofen, phenytoin, and phenobarbital can help reduce the amount of sweating in the body.
T5-B	A person with Hyperhidrosis has a medication treatment because it helps to reduce the amount of blood in the body and reduces the risk of a blood clot.
Input (2)	A person with Alzheimers has an engage in activity prevention because {explanation}
Human	A person with Alzheimers has an engage in activity prevention because engaging in activity will increase nerve connections in the brain or release hormones that increase nerve cell creation and this helps preserve nerve cells in the brain and mental function
T5-L	A person with Alzheimers has an engage in activity prevention because activities such as walking, jogging, swimming, and playing games can help prevent Alzheimer's
BART-B	A person with Alzheimers has an engage in activity prevention because a person with Alzheimer's disease is more likely to develop dementia
Input (3)	A person with Prostatitis has an infection risk factor because {explanation}
Human	A person with Prostatitis has an infection risk factor because the condition is caused by common strains of bacteria which spread from other parts of the urinary or reproductive systems to infect and inflame the prostate resulting in prostatitis.
T5-L	A person with Prostatitis has an infection risk factor because infections of the prostate, urethra, and genital tract can lead to prostatitis.

Table 8: Qual examples for CHARDAT$_{test-seen}$: human, RETR, T5-large (L), T5-base (B), BART-base (B).

5.1 Automatic Evaluation Results

From Table 4, we see that all **CHARD** models perform better than RETR across most metrics. RETR's average outputs are much longer than those of our models and humans. Among our models, T5-large and BART-large perform best, demonstrating that larger models are more adept. T5-large performs best overall (combined with the qual analysis in §5.3), with the longest average outputs among our models. Some of our models achieve better average fluency (PPL) compared to humans, but the outputs are generally noticeably shorter.

From Table 5, we see that T5-large surprisingly performs better on the test-unseen half. It appears that the model can generalize decently to unseen conditions when trained on CHARDAT. This may partially be due to similar explanations for particular DIM attributes across conditions, e.g. why *sleep* helps treat some conditions may be similar.

From Table 6, we see that for most DIM (namely RF and PREV), the model trained on that specific DIM performs better on that DIM. However, our

Method	Text
Input (4)	A person with Costochondritis has a fix improper posture prevention because {explanation}
Human	A person with Costochondritis has an fix improper posture prevention because bad posture puts stress on joints and tissues, causing joint stiffness, weakened muscles, and inflammation that is so common in Costochondritis.
RETR	A person with Costochondritis has a fix improper posture prevention because what can I do to relieve costochondritis? Lifestyle and home remedies. Nonprescription pain relievers. Acetaminophen (Tylenol, others) ibuprofen (Advil, Motrin IB, others) or naproxen sodium (Aleve, others) (...)
T5-L	A person with Costochondritis has a fix improper posture prevention because poor posture can lead to pain and stiffness.
Input (5)	A person with rheumatoid has a diet risk factor because {explanation}
Human	A person with rheumatoid has a diet risk factor because unhealthy food choices can increase inflammation from rheumatoid
RETR	A person with rheumatoid has a diet risk factor because the results of this study revealed that high intakes of carbohydrates (e.g., potatoes) might be associated with increased RA risks.
T5-L	A person with rheumatoid has a diet risk factor because people who eat a high-fat diet are at a higher risk of developing the disease.
BART-L	A person with rheumatoid has a diet risk factor because a diet rich in red meat, dairy products, and dairy products can cause inflammation
Input (6)	A person with depression has a life-changing event risk factor because {explanation}
Human	A person with depression has a life-changing event risk factor because approximately 70% of first depression episodes and 40% of recurrent episodes of depression are preceded by a severe life event
RETR	A person with depression has a life-changing event risk factor because psychological and Psychosocial Risk Factors for Recurrence · by SL Burcusa · 2007 · Cited by 1085 — These factors include age at onset of the first episode, lifetime number of
T5-L	A person with depression has a life-changing event risk factor because a major life event, such as the death of a loved one, can increase the risk of depression.

Table 9: Qual examples for CHARDAT$_{test-unseen}$: human, RETR, T5-large (L), BART-large (L).

general T5-large model performs better on TREAT. It may be that training on CHARDAT has allowed the model to learn from data of other DIM, improving its overall knowledge and generation capabilities (an advantage of a single combined model).

From Figure 3, we see that the BART models generally increase in performance with higher BT tmp (upward trend), whereas T5 fluctuates. This may be due several reasons, e.g. differences in the architecture and pretraining strategies of the models, allowing BART to leverage noisy data more effectively. From Figure 4, we see that performance generally increases for each model up to a certain point (e.g. 2x or 3x DA), and then decreases afterward, aligning with our hypothesis from §4.2.

5.2 Human Evaluation Results

From Table 7, we see that both RETR and T5-large are outperformed by humans, although RETR is relatively close in informativeness and medical accuracy, and T5-large slightly outperforms on readability. RETR outperforms T5-large on medical accuracy and informativeness, which is somewhat expected as it uses Google Search. It is worse than T5-large on readability, as our models generate more fluent, concise, and readable text (see §5.3).

5.3 Qualitative Analysis

We examine the qualitative examples in Tables 8 and 9. Firstly, we see that RETR is able to generally perform well by extracting relevant information (ex.1 - a list of medications for Hyperhidrosis, ex.5 - that carbohydrates increase RA risk), which is expected using Google Search. However, it sometimes extracts a lengthy amount of irrelevant information. For ex.4, RETR extracts a difficult-to-read list of different TREAT strategies, which is for the wrong DIM, and does not narrow down on an explanation for the specific DIM attribute in the input. For ex.6, it extracts the info and beginning of a passage from a scientific article, ending abruptly and not explaining the given DIM attribute.

Our models, specifically T5-large, are generally able to output more concise, readable, and sometimes relevant explanations compared to RETR. For ex.1, T5-large outputs a list of medications, albeit not for Hyperhidrosis - showing weaknesses in medical accuracy. Other than ibuprofen, the other medications are not in CHARDAT$_{tr}$, showing that these were likely already known to T5-large through pretraining. For ex.2, it generates a reasonable list of activities to help prevent Alzheimer's, and for ex.3, it lists correct body parts where an infection can occur to cause Prostatitis. It can generalize well to unseen conditions, as shown through ex.4-6. It reasons that poor posture can lead to pain and stiffness, high-fat diets can increase the chance of rheumatoid, and that a major life event (*"death of a loved one"*) can cause depression. These generalization capabilities are likely from a combination of pretraining and CHARDAT$_{tr}$.

Compared to humans, T5-large's outputs are lacking. Human explanations are typically longer and more informative, explaining the exact reason (*why?*) a specific DIM attribute relates to the given condition. For ex.2, it explains how activities can help *"preserve nerve cells in the brain and mental function"*, whereas T5-large simply lists activities. This similarly occurs for ex.3-5. Human explanations are also typically more medically accurate, e.g. for ex.1, the listed medications are correct. However, we do see that some of T5-large's outputs (for ex.1,2,4) are more readable. Further, T5-large sometimes presents more information, e.g. an exact list of activities for ex.2, a specific type of diet (*"high-fat"*) for ex.5 (human just says *"unhealthy"*), and an example of a life-changing event for ex.6.

BART-large also performs decently. In ex.5, it lists several specific and correct types of foods (*"red meat, dairy products"*). The base models perform much worse. For ex.1, T5-base talks about medication reducing *"blood clots"*, unrelated to Hyperhydrosis. For ex.2, BART-base writes an explanation completely irrelevant to the input DIM.

6 Directions for Improvement

We see that our models are decent and generate readable text, but can improve on medical accuracy and informativeness. While they are not nearly ready for real-world use, they show potential.

As stated, the purpose of **CHARD** is to assess the capabilities of PLMs to act as implicit clinical knowledge bases that can reason through several dimensions. How can we improve our models, and possibly our dataset and task formulation?

Dataset and task formulation: We introduce **CHARD** and initially frame the task using a template infilling approach which is more constrained. More flexible formulations may better leverage the knowledge and generation capabilities of PLMs.

Our current approach involves generating explanations about a single condition and DIM attribute at a time. We can possibly improve CHARDAT by annotating for more complicated input queries. This is because a PLM may be more effective at answering more complicated queries, e.g. comparing and contrasting conditions and DIM and multi-hop reasoning. It is likely easier to make complicated inferences and connections over the abstract PLM embedding space than over retrieved text passages.

Further, we can expand CHARDAT to include more dimensions and topics in the health domain. These improvements may allow for the training of a single system that is able to make complicated clinical inferences across various topics and DIM.

Model improvements: We can explore models such as GPT-3 (Brown et al., 2020) and PALM (Chowdhery et al., 2022) for **CHARD** that are larger with stronger pretraining. We can also investigate enhancing PLMs with information retrieval, e.g. using a retrieval approach to obtain relevant scientific literature as evidence, combined with a text summarization system to digest the content. Our model can then conduct its clinical reasoning on this digested content. Users can potentially take advantage of such a system to automatically verify the medical accuracy of generated explanations, and then improve the generation model itself using this feedback loop (i.e. a self-improving system).

7 Related Work

Constrained Text Generation: There have been several works on constrained text generation. For creative text generation, Gangal et al. (2022) introduce narrative reordering (NAREOR) to edit the temporality of narratives. Keh et al. (2022) and Keh et al. (2023) explore the generation of personifications and tongue twisters, respectively. Donahue et al. (2020) introduce and investigate the task of infilling. Feng et al. (2019) propose Semantic Text Exchange to adjust topic-level text semantics using infilling. Rajagopal et al. (2021) investigate cross-domain reasoning using a prompt-tuning setup. Our work distinctly investigates template infilling for clinical reasoning along dimensions.

Commonsense Reasoning for Models: One large commonsense KG is COMET, which trains on KG edges to learn connections between words and phrases. COSMIC (Ghosal et al., 2020) uses COMET to inject commonsense into models. CommonGen (Lin et al., 2020) assesses the commonsense reasoning of text generation models. Several works investigate CommonGen, including SAPPHIRE (Feng et al., 2021b) and VisCTG (Feng et al., 2022), the latter of which uses visual grounding. Unlike these works, **CHARD** distinctly investigates reasoning for the clinical/health domain.

Reasoning for Clinical/Health Domain: Most existing work here involves retrieval or extraction. MIMICause (Khetan et al., 2022) extracts causal medical information from electronic health records to help understand narratives in clinical texts. Ahne et al. (2022) extract a causal graph and reason about diabetes distress for better understanding the opinions, feelings, and observations of the diabetes online community from a causality perspective.

For generation, Moramarco et al. (2021) investigate the use of LMs to simplify medical text. Abaho et al. (2022) probe factual knowledge from LMs to elicit answers related to treatment outcomes. **CHARD** has a different goal: rather than simply probe for factual knowledge, we assess how LMs can act as and reason through an implicit knowledge base. Meng et al. (2022) investigate probing biomedical knowledge by introducing a benchmark, MedLAMA, that focuses on 19 relations. **CHARD** instead focuses on clinical knowledge reasoning along different dimensions.

8 Conclusion and Future Work

In conclusion, we proposed and investigated the task of **CHARD**: Clinical Health-Aware Reasoning across Dimensions, to explore the capability of text generation models to act as implicit clinical knowledge bases and generate explanations across several health dimensions. We presented a dataset, CHARDAT, and conducted experiments with BART and T5. Extensive evaluation and qualitative analysis demonstrated that our models are decent, especially for generating concise and readable text, but can be improved on medical accuracy and informativeness, and that **CHARD** is challenging with much potential for further exploration. We highly encourage the research community to further investigate and improve upon **CHARD**.

Future directions are discussed in §6. Some additional ideas include trying more data augmentation strategies and decoding strategies for text infilling.

Limitations

We discuss some limitations of our work and potential directions for improvement in §6. Specifically, our template-infilling approach is less flexible, and we can expand to more complicated input queries to better leverage the power of PLMs in future work. Further, CHARDAT focuses on three main clinical dimensions, which can be expanded upon to include more dimensions and topics in the future. Our seq2seq models are also relatively weaker compared to GPT-3, PALM, and recent larger PLMs, which may perform more effectively on **CHARD**. We are also investigating a completely generative approach, and combining generation with retrieval in interesting ways may be more effective. Overall, our current **CHARD** models have room to improve on medical accuracy and informativeness, and are not nearly ready for real-world use.

However, we note again that we are the first to propose **CHARD**, and our work is the first step towards longer-term goals regarding clinical reasoning using PLMs. We are after more of the *commonsense* medical reasoning for now, rather than very deep medical knowledge. In this paper, we see how far one can get with a standard task formulation, NLP methods, seq2seq models, and AMT annotations. As they say, *"walk before you run"*!

Ethics

We collected CHARDAT and conducted our human evaluation studies using AMT, in a manner consistent with terms of use of any sources and intellectual property and privacy rights of AMT crowd workers.

Our collected dataset, CHARDAT, consists of general clinical information, where explanations are impersonal. We also manually examined a large subset of the data, and ensured there were no issues with respect to privacy and other ethical concerns, e.g. offensive words, profanities, racism, gender bias, and other malicious language.

We acknowledge the weaknesses of **CHARD** models and the potential risks if they are used for real-world purposes. We will never use our models or encourage their use for real-world purposes, at least in their current state, and also emphasize this in the paper. As we noted, we propose **CHARD** and conduct our initial experiments purely for investigation purposes and to test our hypotheses. Our paper presents an important contribution to the ML, NLP, and healthcare communities, and we encourage researchers to further improve upon it.

Our task, models, dataset, and accompanying publication are intended only for research purposes and to assess the capabilities of text generators.

References

Micheal Abaho, Danushka Bollegala, Paula Williamson, and Susanna Dodd. 2022. Position-based prompting for health outcome generation. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 26–36, Dublin, Ireland. Association for Computational Linguistics.

Adrian Ahne, Vivek Khetan, Xavier Tannier, Md Imbesat Hassan Rizvi, Thomas Czernichow, Francisco Orchard, Charline Bour, Andrew Fano, and Guy Fagherazzi. 2022. Extraction of explicit and implicit cause-effect relationships in patient-reported diabetes-related tweets from 2017 to 2021: Deep learning approach. *JMIR Med Inform*, 10(7):e37201.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *European conference on computer vision*, pages 382–398. Springer.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

James T. Brophy, Margaret M. Keith, Michael Hurley, and Jane E. McArthur. 2021. Sacrificed: Ontario healthcare workers in the time of covid-19. *NEW SOLUTIONS: A Journal of Environmental and Occupational Health Policy*, 30(4):267–281. PMID: 33174768.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Mayilee Canizares, Monique Gignac, Sheilah Hogg-Johnson, Richard Glazier, and Badley Elizabeth. 2016. Do baby boomers use more healthcare services than other generations? longitudinal trajectories of physician service use across five birth cohorts. *BMJ Open*, 6.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Sébastien Couarraze, Louis Delamarre, Fouad Marhar, Binh Quach, Jiao Jiao, Raimundo Avilés Dorlhiac, Foued Saadaoui, Andy Su-I Liu, Benoît Dubuis, Samuel Antunes, Nicolas Andant, Bruno Pereira, Ukadike C. Ugbolue, Julien S. Baker, The COVISTRESS network, Maëlys Clinchamps, and Frédéric Dutheil. 2021. The major worldwide stress of healthcare professionals during the first wave of the covid-19 pandemic – the international covistress survey. *PLOS ONE*, 16(10):1–16.

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

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. 2020. GenAug: Data augmentation for finetuning text generators. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 29–42, Online. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021a. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Steven Y. Feng, Jessica Huynh, Chaitanya Prasad Narisetty, Eduard Hovy, and Varun Gangal. 2021b. SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Steven Y. Feng, Aaron W. Li, and Jesse Hoey. 2019. Keep calm and switch on! preserving sentiment and fluency in semantic text exchange. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2701–2711, Hong Kong, China. Association for Computational Linguistics.

Steven Y. Feng, Kevin Lu, Zhuofu Tao, Malihe Alikhani, Teruko Mitamura, Eduard Hovy, and Varun Gangal. 2022. Retrieve, caption, generate: Visual grounding for enhancing commonsense in text generation models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10618–10626.

Varun Gangal, Steven Y. Feng, Malihe Alikhani, Teruko Mitamura, and Eduard Hovy. 2022. Nareor: The narrative reordering problem. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10645–10653.

Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. Cosmic: Commonsense knowledge for emotion identification in conversations. *arXiv preprint arXiv:2010.02795*.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Sedrick Scott Keh, Steven Y. Feng, Varun Gangal, Malihe Alikhani, and Eduard Hovy. 2023. Pancetta: Phoneme aware neural completion to elicit tongue twisters automatically. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2023)*.

Sedrick Scott Keh, Kevin Lu, Varun Gangal, Steven Y. Feng, Harsh Jhamtani, Malihe Alikhani, and Eduard Hovy. 2022. PINEAPPLE: Personifying INanimate entities by acquiring parallel personification data for learning enhanced generation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6270–6284, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Vivek Khetan, Md Imbesat Rizvi, Jessica Huber, Paige Bartusiak, Bogdan Sacaleanu, and Andrew Fano. 2022. MIMICause: Representation and automatic extraction of causal relation types from clinical notes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 764–773, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Zaiqiao Meng, Fangyu Liu, Ehsan Shareghi, Yixuan Su, Charlotte Collins, and Nigel Collier. 2022. Rewrite-then-probe: A contrastive recipe for probing biomedical knowledge of pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4798–4810, Dublin, Ireland. Association for Computational Linguistics.

Francesco Moramarco, Damir Juric, Aleksandar Savkov, Jack Flann, Maria Lehl, Kristian Boda, Tessa Grafen, Vitalii Zhelezniak, Sunir Gohil, Alex Papadopoulos Korfiatis, and Nils Hammerla. 2021. Towards more patient friendly clinical notes through language models and ontologies.

Igor Portoghese, Maura Galletta, Ross Coppola, Gabriele Finco, and Marcello Campagna. 2014. Burnout and workload among health care workers: The moderating role of job control. *Safety and Health at Work*, 5:152–157.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Dheeraj Rajagopal, Vivek Khetan, Bogdan Sacaleanu, Anatole Gershman, Andrew Fano, and Eduard Hovy. 2021. Cross-domain reasoning via template filling.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Full List of Health Conditions

See Table 10 for a list of all health conditions in CHARDAT.

B CHARDAT Annotation Details

Human annotation for CHARDAT was done via paid crowdworkers on AMT, who were from Anglophone countries. They were selected through a series of qualification tests on a small subset of the samples, and have a history of high approval rates ($> 95\%$) and good performance on related tasks. Based on initial annotations and performance on the qualification tests, workers were only re-hired if their performance was sufficient over time and they reliably followed the given instructions. The annotators were paid variable amounts (with periodic bonuses over time) depending on their performance and consistency, and the pay for all workers exceeds the minimum wage for the USA.

The workers were asked to write passages (that include explanations) that are as specific and factually accurate as possible, describing how a specific dimension attribute relates to the given condition. Each HIT (annotation page) contains a single condition + dimension attribute combination, and they write a single passage that fills in the given template with an explanation. In the instructions, we describe each dimension in detail, and include several examples of correct and incorrect passages (regarding medical/factual accuracy, brevity/readability, and informativeness). We also encourage them to consult useful and trusted clinical resources such as MayoClinic, CDC, WebMD, and Healthline, if necessary, while writing the explanation.

Annotations were manually examined by the authors as they came in, and annotators were asked to improve their explanations if necessary. Annotators with consistently poor annotations were asked to stop annotating, and their completed annotations were re-annotated by others. At the end of the data collection process, the authors manually examined a large subset of CHARDAT, ensuring sufficiently high quality of annotations in terms of medical accuracy, informativeness, and readability.

C Further Model Finetuning and Generation Details

T5-large consists of 770M params, T5-base 220M params, BART-large 406M params, and BART-base 139M params. For all models, we use beam search with a beam size of 5, decoder early stopping, a decoder length penalty of 0.6, and a decoder minimum length of 1. We set the maximum encoder and decoder lengths depending on values that can fit all examples in CHARDAT_{tr}, which ended up being 32 and 128 (for encoder and decoder, respectively) for the BART models, and 35 and 128 for the T5 models. Models are trained using fp16, and Adam optimizer with epsilon=1e-08. We use a training seed of 42 for all models, and a random seed of 42 for all other scripts that involved randomization. Decoding is done using beam search with a beam width of 5.

For model training, we use a batch size of either 64 or 32 for T5-base and BART-base, and either 8 or 16 for BART and T5-large (depending on GPU memory). For T5-base and BART-base, we use 400 warmup steps, 500 for BART-large, and 1200 for T5-large. We train all models up to a reasonable number of epochs (e.g. 20 to 30 for base models and 10 to 15 for large models). The learning rates for **CHARD** models were determined by trying a range of values (e.g. from 1e-8 to 5e-1), and finding ones which led to good convergence behavior. For the best-performing models, learning rates are as follows: BART-base = 5e-06, BART-large = 1e-05, T5-base = 1e-03, T5-large = 1e-05.

Training was done using single GTX 1080 Ti, TITAN RTX, RTX 2080 Ti, and GTX TITAN X GPUs. Model training time varied depending on the model type+size and amount of data augmentation, and varied between 5 minutes to 3 hours.

D Further Human Evaluation Details

Human evaluation was done via paid crowdworkers on AMT, who were from Anglophone countries. They were selected through qualification tests and have a history of high approval rates ($> 95\%$) and good performance on related tasks. Each example was evaluated by 2 annotators. The time given for each AMT task instance or HIT was 1 hour maximum for an approximately 1-minute task. Sufficient time to read instructions, as calibrated by authors, was also considered. Annotators were paid 20 cents per HIT. This rate ($12/hr) exceeds the minimum wage for the USA ($7.25/hr) and constitutes fair pay. Workers who performed well were also paid periodic bonuses based on the timeliness and quality of their annotations.

The human evaluation was split into 9 studies: 3 pairwise method comparisons (RETR vs. T5-large,

Dysthymia	cfs	ibs	Narcolepsy	bulimia
Hypothyroidism	Costochondritis	psychosis	CysticFibrosis	POTS
MultipleSclerosis	Gastroparesis	gout	adhd	diabetes
CrohnsDisease	lupus	rheumatoid	Sinusitis	thyroidcancer
Hyperhidrosis	gerd	AnkylosingSpondylitis	endometriosis	schizophrenia
asthma	bipolar	depression	pcos	covid19
acne	anxiety	dementia	ptsd	dystonia
Epilepsy	ErectileDysfunction	Herpes	insomnia	Anemia
LymeDisease	migraine	ocd	parkinsons	Alzheimers
hpv	Prostatitis	backpain	Sciatica	Fibromyalgia
bpd	PagetSchroetter			

Table 10: A list of all 52 health conditions used in CHARDAT.

RETR vs. human, and human vs. T5-large) by 3 dimensions (risk factors, treatment, and prevention). Each HIT or task page presented a given condition, the associated dimension attribute, and two explanations (from the two methods of the study) in a random order. They are asked to select among the three choices (first explanation is better, second explanation is better, *hard to prefer one over the other*) for the three evaluation aspects of medical accuracy, informativeness, and readability. In the instructions, we describe the clinical dimension and each evaluation aspect in detail with positive and negative examples of each, and encouraged them to consult useful and trusted clinical resources such as MayoClinic, CDC, WebMD, and Healthline, if necessary, while evaluating the explanations.

E Further Qualitative Examples

See Tables 11 and 12 for more qualitative examples on CHARDAT_{test}.

Method	**Text**
Input	A person with Hyperhidrosis has a/an infection risk factor because/since/as {explanation}
Human	A person with Hyperhidrosis has an infection risk factor because some kinds of infections like tuberculosis, HIV, bone infection cause hyperhidrosis
RETR	A person with Hyperhidrosis has a/an infection risk factor because/since/as this increases the risk of skin conditions such as athlete's foot and more severe conditions such as bacterial infections or pitted keratolysis. Study results report a nearly 30% greater risk of skin infections in patients with hyperhidrosis compared with healthy controls.
T5-L	A person with Hyperhidrosis has an infection risk factor because infections such as staph, typhoid, hepatitis and polio can cause excessive sweating.
T5-B	A person with Hyperhidrosis has an infection risk factor because a person who has a viral infection is more likely to have a virus infection.
BART-L	A person with Hyperhidrosis has an infection risk factor because infection can cause hyperhidrosis
BART-B	A person with Hyperhidrosis has an infection risk factor because a person with hyperhidrosis is more likely to have an infection.
Input	A person with AnkylosingSpondylitis has a/an age risk factor because/since/as {explanation}
Human	A person with AnkylosingSpondylitis has an age risk factor because about 80 percent of people first develop ankylosing spondylitis symptoms under age 30
RETR	A person with AnkylosingSpondylitis has a/an age risk factor because/since/as about 80 percent of people first develop ankylosing spondylitis symptoms under age 30
T5-L	A person with AnkylosingSpondylitis has an age risk factor because people over 50 years of age are at a higher risk of developing the disease
T5-B	A person with AnkylosingSpondylitis has an age risk factor because older adults are more likely to develop ankylosis.
BART-L	A person with AnkylosingSpondylitis has an age risk factor because the risk of developing ankylosis increases with age
BART-B	A person with AnkylosingSpondylitis has an age risk factor because older people are more likely to develop the disease
Input	A person with rheumatoid has a/an regular exercise treatment because/since/as {explanation}
Human	A person with rheumatoid has a/an regular exercise treatment because it helps reduce the disability often associated with rheumatoid arthritis and boosts strength and flexibility
RETR	A person with rheumatoid has a/an regular exercise treatment because/since/as but exercise is one of the key treatments to help reduce the disability often associated with rheumatoid arthritis. Regular exercise can boost strength and flexibility in people who have rheumatoid arthritis. Stronger muscles can better support your joints, while improved flexibility can aid joint function.
T5-L	A person with rheumatoid has a regular exercise treatment because exercise can help reduce inflammation and improve blood flow to the joints.
T5-B	A person with rheumatoid has a regular exercise treatment because regular exercise can help reduce the symptoms of the disease.
BART-L	A person with rheumatoid has a regular exercise treatment because exercise can help reduce inflammation and pain
BART-B	A person with rheumatoid has a regular exercise treatment because regular exercise can help reduce inflammation and inflammation in the body.

Table 11: More qual examples for CHARDAT$_{test}$: human, RETR, T5-large (L), T5-base (B), BART-large (L), BART-base (B)

Method	**Text**
Input	A person with thyroidcancer has a/an chemotherapy treatment because/since/as {explanation}
Human	A person with thyroidcancer has a chemotherapy treatment because Chemotherapy (chemo) uses anti-cancer drugs that are injected into a vein or are taken by mouth. Chemotherapy is systemic therapy, which means that the drug enters the bloodstream and travels throughout the body to reach and destroy cancer cells and stops cancer growth.
RETR	A person with thyroidcancer has a/an chemotherapy treatment because/since/as chemotherapy is seldom helpful for most types of thyroid cancer, but fortunately it is not needed in most cases. It is often combined with external beam radiation therapy for anaplastic thyroid cancer and is sometimes used for other advanced cancers that no longer respond to other treatments.
T5-L	A person with thyroidcancer has a chemotherapy treatment because chemotherapy destroys the cells that cause the cancer.
T5-B	A person with thyroidcancer has a chemotherapy treatment because chemotherapy reduces the risk of thyroid cancer.
BART-L	A person with thyroidcancer has a chemotherapy treatment because chemotherapy can slow down the growth of thyroid cancer
BART-B	A person with thyroidcancer has a chemotherapy treatment because chemotherapy can help reduce the risk of thyroid cancer
Input	A person with depression has a/an medical conditions risk factor because/since/as {explanation}
Human	A person with depression has a medical conditions risk factor because the pain and stress that come with certain conditions, including cancer, stroke, or heart disease, can take a toll on a person's mental state resulting in depression.
RETR	A person with depression has a/an medical conditions risk factor because/since/as more commonly studied were the connections between depression and cardiovascular disease, metabolic syndrome, biochemical alterations, diabetes, dementia, cognitive impairment, Alzheimer's disease, somatization and chronic pain, asthma, arthritis, and hyperlipidemia.
T5-L	A person with depression has a medical conditions risk factor because depression can be caused by a number of medical conditions, such as diabetes, heart disease, high blood pressure, high cholesterol and high blood sugar.
T5-B	A person with depression has a medical conditions risk factor because medical conditions such as diabetes, heart disease, diabetes, AIDS, and other conditions that cause depression, such as cancer, are linked to depression.
BART-L	A person with depression has a medical conditions risk factor because depression can be caused by a variety of medical conditions.
BART-B	A person with depression has a medical conditions risk factor because depression can be a cause of depression.
Input	A person with depression has a/an grief risk factor because/since/as {explanation}
Human	A person with depression has a grief risk factor because grief associated with loss of dear one greatly increases the risk of psychiatric complications such as depression
RETR	A person with depression has a/an grief risk factor because/since/as risk Factors · Genetics: A history of depression in your family may make it more likely for you to get it. · Death or loss: Sadness and grief are ...
T5-L	A person with depression has a grief risk factor because people who have experienced a loss of a loved one are more likely to develop depression.
T5-B	A person with depression has a grief risk factor because grief is the most common cause of depression.
BART-L	A person with depression has a grief risk factor because grief can cause depression
BART-B	A person with depression has a grief risk factor because grief is associated with depression.

Table 12: More qual examples for CHARDAT$_{test}$: human, RETR, T5-large (L), T5-base (B), BART-large (L), BART-base (B)

Prompt Tuning with Contradictory Intentions for Sarcasm Recognition

Yiyi Liu, Ruqing Zhang, Yixing Fan, Jiafeng Guo, Xueqi Cheng
CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology,
Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, Beijing, China
{liuyiyi17s,zhangruqing,fanyixing,guojiafeng,cxq}@ict.ac.cn

Abstract

Recently, prompt tuning has achieved promising results in a variety of natural language processing (NLP) tasks. The typical approach is to insert text pieces (i.e., templates) into the input and transform downstream tasks into the same form as pre-training. In essence, a high-quality template is the foundation of prompt tuning to support the performance of the converted cloze-style task. However, for sarcasm recognition, it is time-consuming and requires increasingly sophisticated domain knowledge to determine the appropriate templates and label words due to its highly figurative nature. In this work, we propose SarcPrompt, to incorporate the prior knowledge about contradictory intentions into prompt tuning for sarcasm recognition. SarcPrompt is inspired by that the speaker usually says the opposite of what they actually mean in the sarcastic text. Based on this idea, we explicitly mimic the actual intention by prompt construction and indicate whether the actual intention is contradictory to the literal content by verbalizer engineering. Experiments on three public datasets with standard and low-resource settings demonstrate the effectiveness of our SarcPrompt for sarcasm recognition.

1 Introduction

Sarcasm is a sophisticated language phenomenon in which one conveys implicit intention with the opposite meaning of what is said or written literally (Campbell and Katz, 2012; Joshi et al., 2015). Due to its high ambivalence and figurative nature, sarcasm recognition which targets to predict a text as sarcastic or non-sarcastic, becomes a particularly challenging classification task. With the usage of sarcasm becoming prevalent on social media platforms like microblogs and online forums, sarcasm recognition has received growing research attention to facilitate sentiment analysis applications. Recent advances have shown that Pre-trained Language Models (PLMs), such as RoBERTa (Liu et al., 2019b), BERT (Devlin et al., 2019), can achieve promising performance in many downstream Natural Language Processing (NLP) tasks (Xu et al., 2019; Liu et al., 2019a; Zhou et al., 2021). The success of PLMs has also attracted much attention for sarcasm recognition. Researchers mainly add extra classifiers on top of PLMs (Lou et al., 2021) to further train the models under classification objectives, or re-train popular PLMs by incorporating sentiment knowledge and external sarcastic corpus (Babanejad et al., 2020).

Despite fine-tuning has achieved satisfying results, some recent studies (Schick and Schütze, 2021a,b) have found that one of its critical challenges is the significant gap in objective forms between pre-training and fine-tuning. This largely limits the transfer and adaptation of knowledge in PLMs to downstream tasks. Recently, a series of studies propose to use prompt tuning (Han et al., 2021; Schick and Schütze, 2021a; Chen et al., 2022b) to bridge the gap between pre-training and fine-tuning. Specifically, the downstream task is formulated as a (masked) language modeling problem similar to the pre-training. By fusing the original input with the specially constructed prompt template to predict `[MASK]` and then mapping predicted words to corresponding labels, we can stimulate the task-related knowledge in PLMs to boost the model's performance.

Nevertheless, there are still several non-trivial challenges for sarcasm recognition with prompt tuning as follows. On the one hand, texts of a specific task can differ from that of PLMs used in prompt tuning (Chen et al., 2022b). For sarcasm recognition, the sarcastic data is usually composed of abundant deliberately ambiguous texts. In contrast, the corpora for PLMs in prompt tuning mainly include objective and deterministic texts (Liu et al., 2022). This greatly restricts PLMs from taking full advantage of their knowledge. On the other hand, though prompt tuning works well in text classifica-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 328–339
May 2-6, 2023

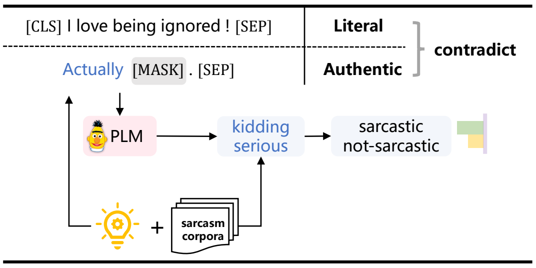


Figure 1: The example of prompt tuning to stimulate the contradictory for sarcasm recognition.

tion tasks, prompt templates and verbalizer are not easily transferable to sarcasm recognition. Both handcrafting an appropriate prompt template and choosing effective label words require domain expertise in sarcasm language. Furthermore, there exists deliberate ambiguity in sarcasm and the special written content cannot be ignored. In this sense, we argue that the power of prompt tuning has not been fully exploited for sarcasm recognition.

Shed light by the above challenges and insights, in this paper, we propose SarcPrompt, a novel prompt tuning method to leverage the contradiction knowledge-enhanced prompts to tune the PLMs. SarcPrompt is inspired by that sarcasm is known as "the activity of saying or writing the opposite of what you mean" (Tungthamthiti et al., 2014). According to this definition, we can recognize sarcasm by evaluating the inconsistency between the actual intention and the literal content in sarcastic texts. Based on intention contradiction, we carefully devise the prompt templates that can mimic the speakers' actual intention and then select label words to judge whether the prompt is contradictory to the literal content. Take the sentence in Figure 1 as an example, the original input is "I love being ignored". A good prompt template may denote "***Actually*** *[MASK]*". "Actually" is an indicator to trigger the authentic intention. If PLMs predict the masked position with a good label word "kidding", the new completed sentence denotes "I love being ignored. ***Actually*** *kidding*.". The intention of the prompt template completed with the predicted word is contradictory to the original input's content. Then we can intuitively recognize a sarcastic text.

To be specific, SarcPrompt mainly contains two steps: prompt construction and verbalizer engineering. Firstly, in the stage of prompt construction, we devise two kinds of sarcasm-specific prompts with different patterns to mimic the actual intention straightforwardly. Secondly, during the verbalizer engineering, we determine the label words that trigger contradictory or suggest sarcasm based on the statistical information of sarcastic corpora. Furthermore, we investigate a contrastive loss to comprehend various sarcastic contrast patterns, jointly with the cross-entropy loss for optimization. The contrastive loss strives to pull the representation of a sarcastic text towards that of other sarcastic texts in the same mini-batch, while pushing it away from representations of other non-sarcastic texts.

We conduct extensive experiments on three benchmark datasets for sarcasm recognition. Empirical experimental results demonstrate that our SarcPrompt[1] has achieved state-of-the-art performance under both standard supervised settings and low-resource settings.

2 Related Work

In this section, we briefly review two lines of related work, including sarcasm recognition and prompt tuning.

Sarcasm Recognition. Identifying sarcasm in texts has evolved from traditional methods to deep neural methods. Traditional approaches mostly utilize machine learning methods with manually engineered features (Riloff et al., 2013; González-Ibáñez et al., 2011; Patra et al., 2016; Hee et al., 2018b; González-Ibánez et al., 2011; Joshi et al., 2015). Deep neural models for sarcasm recognition aim to capture the contrast of sarcasm through the design of the model structure. They are mainly divided into two categories: contrast between words or phrases intra-sentence(Ghosh and Veale, 2016; Wu et al., 2018; Ghosh and Veale, 2018; Tay et al., 2018; Xiong et al., 2019; Lou et al., 2021) and contrast based on the essence of sarcasm phenomenon (Liu et al., 2022).

With the advent of pre-trained language models, a lot of sarcasm recognition methods based on PLMs have been proposed. There are two lines of utilizing PLMs: fine-tuning and pre-training. Fine-tuning mode uses PLMs as an encoder to obtain text representation (Lou et al., 2021; Liu et al., 2022; Li et al., 2021). Pre-training mode usually combines sentiment knowledge and external sarcastic corpora to advance the performance of sarcasm recognition (Babanejad et al., 2020). However, the existing methods using PLMs have many drawbacks which make PLMs ineffective in sarcasm recognition. Pre-training a new language model for a specific task is seriously affected by the external

---

[1] https://github.com/yiyi-ict/sarcprompt.

knowledge used, and also increases the complexity. And the acquisition and selection of external knowledge suitable for different sarcasm patterns are not easy.

Prompt Tuning. With the emergence of GPT-3 (Brown et al., 2020), prompt tuning, as a new paradigm for utilizing PLMs, has attracted more and more attention of researchers. Prompt tuning methods have achieved promising performance in many NLP tasks (Schick and Schütze, 2021a; Hu et al., 2022; Chen et al., 2022a; Zhong et al., 2021; Chen et al., 2022b). Especially, a lot of work has been proposed to solve text classification, which can be divided into two categories: manual prompts (Schick and Schütze, 2021a) and automatic prompts (Gao et al., 2021; Schick et al., 2020). Hu et al. (2022) propose to incorporate external knowledge into verbalizer for text classification. Chen et al. (2022b) propose to use input with prompt as a query to retrieve relevant task-specific data from large raw texts, which makes prompt tuning better fit classification tasks. Shin et al. (2020) propose gradient-guided search method to automatically generate prompt templates. However, these work cannot be adapted to sarcasm recognition directly. So far there is no prompt tuning method specially designed for sarcasm recognition.

3 Background

Before introducing SarcPrompt, we first briefly review the regular prompt tuning method for sentiment classification. Formally, let M be a masked language model with vocabulary V and mask token `[MASK]` $\in V$, and let $\mathcal{Y}$ be the set of labels for sentiment classification.

Traditional classification methods including fine-tuning PLMs train a model to take in an input $\mathbf{x} = (x_0, x_1, \ldots, x_n)$, generate a probability distribution over class $\mathcal{Y}$ and predict an output y as $P(y|\mathbf{x})$. As for prompt tuning, the input $\mathbf{x}$ is wrapped with a *prompt template*. For example, assuming we need to classify the input sentence $\mathbf{x} =$ "Best pizza ever!" into label POSITIVE or NEGATIVE, the prompt template is defined as "It is `[MASK]`.". This is a general template widely accepted and used in text classification tasks (Schick and Schütze, 2021a; Hu et al., 2022), which is also applicable to sarcasm recognition task. Then the wrapped input is

$$\mathbf{x}_{\mathrm{p}} = \mathbf{x} \text{ It is } \texttt{[MASK]}.$$

Then M generates a probability over vocabulary V on position `[MASK]`, which gives the probability of each token $v \in V$ being filled in `[MASK]` token $P_M(\texttt{[MASK]} = v|\mathbf{x}_{\mathrm{p}})$. Furthermore, a *verbalizer* is to map from *label word* set $V_y \in V$ to the label space $\mathcal{Y}$. Corresponding to the above prompt, we may define $V_{pos} = \{\text{"positive"}\}$ as the label word of "positive" class, $V_{neg} = \{\text{"negative"}\}$ denotes the label word of "negative" class. Then the probability of label y is calculated as

$$P(y|\mathbf{x}_{\mathrm{p}}) = g(P_M(\texttt{[MASK]} = v|\mathbf{x}_{\mathrm{p}})|v \in V_y),$$

where g is a function that transforms the probability of label words into the probability of the class. If $P(y_{pos}) > P(y_{neg})$, we classify the instance into POSITIVE.

4 SarcPrompt

In this section, we introduce our prompt tuning model for sarcasm recognition (SarcPrompt) in detail. The overview of SarcPrompt is shown in Figure 2. We first introduce prompt construction and then verbalizer engineering. Finally we elucidate the training objective.

4.1 Prompt Construction

Recall that the representative characteristic of sarcastic texts is the contradiction between the literal content and the actual intention. The actual intention usually hides behind the literal content obscurely. The goal of prompt construction is to mimic the actual intention. Specifically, we define two kinds of prompts, including clash prompt and question prompt.

Clash Prompt. The goal of clash prompt for sarcasm recognition is to mimic the actual intention. For sarcastic texts, a good prompt template should reflect disapproval of the original text; for non-sarcastic texts, the prompt should agree with the meaning of the original text. Therefore, we utilize fact-related phrases which can either express the inconsistency with facts or consistency with facts.

We design five clash prompts $T_{c_1}(\mathbf{x})$~$T_{c_5}(\mathbf{x})$. Key phrases such as "in fact", "actually" are used to elicit whether the prompt clause has contradictory intentions with original input $\mathbf{x}$.

$T_{c_1}(\mathbf{x}) = \mathbf{x}$ Actually `[MASK]`.

$T_{c_2}(\mathbf{x}) = \mathbf{x}$ In reality, it was `[MASK]`.

$T_{c_3}(\mathbf{x}) = \mathbf{x}$ As a matter of fact, it was `[MASK]`.

$T_{c_4}(\mathbf{x}) = \mathbf{x}$ To tell you the truth, it was `[MASK]`.

$T_{c_5}(\mathbf{x}) = \mathbf{x}$ In fact, it was `[MASK]`.

Question Prompt. Question prompt is equiva-

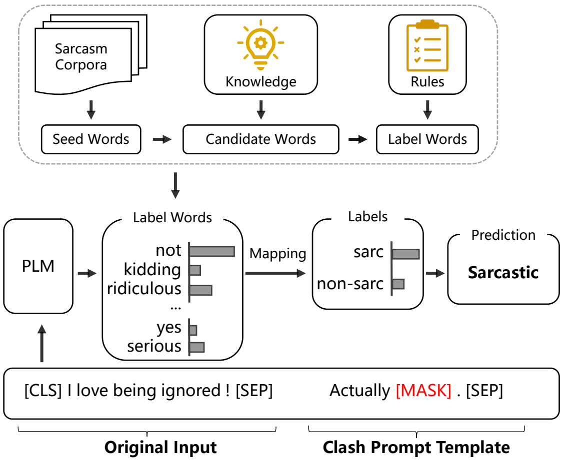


Figure 2: SarcPrompt specified with clash prompt.

lent to directly telling the model that the task is to identify sarcasm by asking a question. Inspired by the summary of prompt design for different tasks in (Liu et al., 2021), we propose three question prompt templates $T_{q_1}(\mathbf{x})$~$T_{q_3}(\mathbf{x})$ for sarcasm recognition. Compared with clash prompt, question prompt is a more direct way to judge whether the input text $\mathbf{x}$ is sarcastic.

$T_{q_1}(\mathbf{x}) = \mathbf{x}$ Are you kidding? `[MASK]`.

$T_{q_2}(\mathbf{x}) = \mathbf{x}$ Are you sarcastic? `[MASK]`.

$T_{q_3}(\mathbf{x}) = \mathbf{x}$ Are you ironic? `[MASK]`.

4.2 Verbalizer Engineering

By fusing the original input with the prompt template, we can obtain a new input $\mathbf{x}_\mathrm{p}$. Then PLMs can output a probability distribution over the vocabulary set V. The goal of verbalizer engineering is to determine label words that should be filled in the masked position, and then how to map label words to the corresponding labels. Verbalizer engineering is crucial because label words directly determine the corresponding relationship with sarcasm or not. Specifically, verbalizer engineering consists of two steps: label word searching and verbalizer mapping.

4.2.1 Label Word Searching

The goal of label word searching is to find words that are appropriate in the masked position. The key idea is to find words that suggest the contradictory intention between the original input and the prompt template. For question prompt, “Yes” or “No” is enough to answer the questions. There is no need to search for label words.

For clash prompt, the label word searching mainly includes three steps: (1) to determine seed words based on the statistical information of sarcastic corpora; (2) to retrieve candidate words by knowledge based on the seed words, and (3) to denoise and obtain the final label words based on rules. In the next, we will introduce the process of label word searching for the clash prompt.

Table 1: Top 5 frequently appearing hashtags in a sarcastic tweet dataset.

Hashtag	Frequency
#not	42.12%
#sarcasm	37.62%
#irony	21.14%
#joke	0.84%
#kidding	0.37%

Seed Words. The regular prompt tuning method usually uses the class name as the only label word for each class directly. For the sarcastic class, the label word can be "sarcasm", however, for the non-sarcastic class, the label word is not trivial to find. The intuitive idea of designing verbalizer is to reflect the contradiction of the literal content and the actual intention in one sarcastic text. We observe that people tend to add sarcasm-related hashtags to suggest the content they post expresses sarcastic meaning especially on the Twitter platform, which is shown below.

I love waking up with migraines ***#not***
I just love when you test my patience!! ***#sarcasm***

We count the frequency of the hashtags used in a sarcastic dataset consisting of tweets[2]. And the top five frequently appearing hashtags in sarcastic texts are shown in Table 1. Expressions in real scenes provide us with prior knowledge for determining label words of sarcasm recognition, which is exactly what we want in SarcPrompt. We use “not”, “irony”, “sarcasm”, “kidding”, “joke”, which appear frequently in the sarcasm-related hashtags as seed words for sarcastic class. On the contrary, their antonyms serve as seeds for the non-sarcastic class. These words are well-suit to indicate that the prompt template has contradictory intentions with the original input’s content.

Candidate Words. Based on the seed words, we can retrieve more candidate words from the knowledge base. The process of predicting masked words based on the context is not a single-choice procedure. There is no standard correct answer. Maybe abundant words fit this context. So it is necessary to expand the candidate word set.

---

[2] https://github.com/Cyvhee/SemEval2018-Task3

Table 2: Examples of the final label words for different prompts.

Prompt Type	Label	Label Words
Clash Prompt	Sarcastic	not, sarcasm, irony, joke, kidding, no, ridiculous...
	Non-sarcastic	yes, do, so, serious, true, real, indeed...
Question Prompt	Sarcastic	yes
	Non-sarcastic	no

Inspired by (Hu et al., 2022), we choose Related Words[3] as our external KB to widen the coverage of candidate words. Related Words is a knowledge graph $\mathcal{G}$ aggregated from multiple resources, including ConceptNet (Speer et al., 2017), WordNet (Pedersen et al., 2004), et al..

Label Words. After the expansion of seed words, we obtain comprehensive candidate words. However, the collected candidate words can be noisy since the vocabulary of the KB is not tailored for sarcasm. Thus we refine the candidate word set by frequency and part of speech. There are several criteria for denoising candidate words: (1) For words out of PLMs' vocabulary and sarcastic corpus, discard them; (2) Only keep words whose parts of speech are adjective, noun, verb, and adverb to ensure consistency with prompt templates syntactically and semantically. Finally, label words are defined in Table 2.

4.2.2 Verbalizer Mapping.

The goal of verbalizer mapping is to map the predicted probability of label words to the final label y. For question prompt, since there is only one label word in each class, the probability of the label word is the probability of the corresponding class.

For clash prompt, there is more than one label word for each class. We use the average of label words based on label word searching process as the probability of each class, which is

$$P(\hat{y}) = \frac{\sum_{v \in V_y} P_M(\texttt{[MASK]} = v|\mathbf{x}_p)}{|V_y|}. \quad (1)$$

4.3 Training Objective

The process of SarcPrompt is a prompt-oriented fine-tuning approach (Gu et al., 2022). We need to compute the loss based on the supervised label of datasets and train the model. The training objective of SarcPrompt considers two aspects including cross-entropy loss and contrastive loss.

Cross-entropy Loss. The first training objective is to minimize the cross-entropy loss of the sarcasm label probability distribution. The cross-entropy loss is to ensure the basic ability of sarcasm recognition. The objective L_{sarc} is formulated as:

$$L_{sarc}(\theta) = \sum \text{cross-entropy}(y, P(\hat{y})), \quad (2)$$

where y is the groundtruth of the sarcasm label, and $P(\hat{y})$ is the predicted score.

Contrastive Loss. Inspired by (Khosla et al., 2020), we introduce contrastive learning objective to our SarcPrompt model. Sarcasm takes many patterns of expression such as sarcasm by clash, situational sarcasm, and other sarcasm (Hee et al., 2018a). Supervised contrastive learning is an automatic way of capturing the entailed similarity of various sarcasm patterns.

Specifically, for $(\mathbf{x}_i, y_i)$ within a batch, we first extract sentence representation s_i. Recall that in Equation 1, we obtain the probability distribution of the label by computing the score of predicted words in the masked position. This can not only be viewed as word representation in the masked position, but also be viewed as a kind of sentence representation. So we use this probability distribution as sentence representation s_i to calculate the contrastive loss. Then the supervised contrastive loss in a batch L_{con} is defined as:

$$P_{con}(i,c) = \frac{\exp\left(sim(s_i, s_c)/\tau\right)}{\sum_{b \in B, b \neq i} \exp\left(sim(s_i, s_b)/\tau\right)} \quad (3)$$

$$L_{con}(\theta) = \sum_{i \in B} -\log \frac{1}{C_i} \sum_{y_i = y_c, c \neq i} P_{con}(i,c). \quad (4)$$

Here $P_{con}(i,c)$ indicates the likelihood that s_c is most similar to s_i. τ is the temperature of softmax. We use $sim(s_i, s_c) = s_i \cdot s_c$ for similarity calculation. Supervised contrastive loss L_{con} is calculated for each input sentence representation s_i in a batch. And $C_i = |\{c|y_c = y_i, c \neq i\}|$ is the number of samples in the same category y_i in a batch.

Considering the two objectives, we obtain the final objective function L by adding them together:

$$L(\theta) = \lambda_1 L_{sarc}(\theta) + \lambda_2 L_{con}(\theta),$$

---

[3] https://relatedwords.org

Table 3: Statistics of datasets. Avg ℓ denotes the average length of texts in the number of tokens.

Dataset	Train	Valid	Test	Avg ℓ
IAC-V1[5]	1,595	80	320	68
IAC-V2[6]	5,216	262	1,042	43
Tweets[7]	3,634	200	784	14

where θ is the parameter set of the model. λ_1, λ_2 are used to leverage the contributions.

5 Experimental Settings

5.1 Datasets

We carry out experiments on three benchmark datasets: IAC-V1, IAC-V2, and Tweets.

- **IAC-V1** (Lukin and Walker, 2017) and **IAC-V2** (Oraby et al., 2016) are collected from online political debates forum[4]. IAC-V2 contains more data than IAC-V1.
- **Tweets** dataset is proposed in SemEval 2018 Task 3 Subtask A (Hee et al., 2018a).

Table 3 reports the statistics. For all datasets, we follow the train/valid/test split in (Liu et al., 2022). All datasets are class-balanced.

5.2 Baselines

We adopt three types of baseline methods for comparison, including deep models without pre-training, pre-trained models with fine-tuning, and pre-trained models with prompt tuning.

Deep Models without Pre-training. We choose recent and widely compared deep models including LSTM-based Bi-LSTM (Hochreiter and Schmidhuber, 1997), CNN-LSTM-DNN (Ghosh and Veale, 2016), attention-based MIARN (Tay et al., 2018), graph neural network-based ADGCN (Lou et al., 2021) and state-of-the-art DC-Net (Liu et al., 2022). In this work, we study context-free sarcasm recognition. The input is a sentence without contexts like history posts and user profiles (Hazarika et al., 2018). So the baselines we choose are restricted with this area.

Pre-trained Models with Fine-tuning. For fine-tuning pre-trained models, we utilize RoBERTa$_{\text{base}}$ as the basic encoder to make a fair comparison. Both ADGCN and DC-Net report versions of fine-tuning pre-trained models in their paper. So we use the RoBERTa as backbone and implement ADGCN-RoBERTa and DC-Net-RoBERTa based on their released code.

Pre-trained Models with Prompt Tuning. Recall that in Section 3, we introduce a regular prompt tuning method for sentiment classification. We utilize this approach as the prompt tuning baseline. Specifically, the vebalizer of regular prompt tuning method is the same as clash prompt.

5.3 Implementation Details

Under the **standard supervised settings**, we utilize the whole datasets to fine-tune. For hyperparameters, we employ $\lambda_1 = 1$, and λ_2 among $\{0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$ on the validation set respectively. 0.005, 0.5 and 0.05 are the best λ_2 for IAC-V1, IAC-V2 and Tweets respectively. As for the **low-resource settings**, we randomly sample $\mathbf{r}\%$ instances of each class from the initial training and validation sets to form the few-shot training and validation sets. $\mathbf{r}$ ranges from $\{1, 5, 10, 20\}$. Our model is implemented based on the open source PET[8]. PET (Schick and Schütze, 2021a) is a regular prompt tuning method that uses the class name as the only label word for each class. But we do not use any tricks in the original PET paper since we want to study the effect of sarcasm-specific templates and label words alone. For each method, we train them with three seeds and report the results of the best seed.

6 Experimental results

In this section, we report and analyze the experimental results to demonstrate the effectiveness of the proposed SarcPrompt method. Specifically, we target the following research questions:

- **RQ1:** How does SarcPrompt perform under the standard supervised settings?
- **RQ2:** How does SarcPrompt perform under the low-resource settings?
- **RQ3:** Which prompt performs best and why?
- **RQ4:** How does the contrastive loss affect the performance of SarcPrompt?
- **RQ5:** Can we better understand how SarcPrompt performs via some case studies?

6.1 Results under Standard Settings

To answer **RQ1**, we compare SarcPrompt with three kinds of strong baselines on three benchmark

---

[4] http://www.4forums.com/political/

[5] https://nlds.soe.ucsc.edu/sarcasm1

[6] https://nlds.soe.ucsc.edu/sarcasm2

[7] https://github.com/Cyvhee/SemEval2018-Task3

[8] https://github.com/timoschick/pet

Table 4: Precision, recall, macro $F1$, and accuracy under standard supervised settings. The “*” in the upper right corner of the model name represents that the results are retrieved from (Liu et al., 2022). Best results are bold.

Standard Supervised Settings												
Model	IAC-V1				IAC-V2				Tweets			
	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.
Deep models without pre-training												
Bi-LSTM*	64.6	64.6	64.6	64.6	79.8	79.7	79.7	79.7	71.8	71.7	71.7	73.0
CNN-LSTM-DNN*	61.5	61.2	60.9	61.1	75.4	75.3	75.2	75.3	71.9	72.9	71.9	72.3
MIARN*	65.6	65.2	64.9	65.2	75.4	75.3	75.2	75.3	68.6	68.8	68.8	70.2
ADGCN*	64.3	64.3	64.3	64.3	81.0	80.9	80.9	80.9	72.6	73.2	72.8	73.6
DC-Net*	66.6	66.5	66.4	66.5	82.2	82.1	82.1	82.1	76.4	77.5	76.3	76.7
Pre-trained models with fine-tuning												
RoBERTa	73.0	72.1	71.9	72.1	82.9	82.8	82.7	82.7	72.7	72.8	72.8	73.9
ADGCN-RoBERTa	72.5	72.4	72.4	72.4	82.2	82.1	82.1	82.1	71.3	71.9	71.4	72.2
DC-Net-RoBERTa	69.7	69.3	69.1	69.3	83.7	83.7	83.7	83.7	69.7	68.3	68.7	70.9
Pre-trained models with prompt tuning												
Prompt Tuning-RoBERTa	72.5	72.1	72.0	72.1	83.4	83.3	83.3	83.3	71.8	72.7	71.8	72.3
SarcPrompt-Question-RoBERTa	73.7	73.0	72.9	73.0	84.3	84.2	84.2	84.2	74.1	75.2	73.7	74.0
SarcPrompt-Clash-RoBERTa	**75.5**	**75.2**	**75.2**	**75.2**	**84.9**	**84.9**	**84.9**	**84.9**	**77.1**	**78.3**	**76.6**	**76.9**

datasets under standard supervised settings. Table 4 shows the results. We observe that: (1) Fine-tuning models perform better than traditional deep models on IAC-V1 and IAC-V2, which demonstrates the effectiveness of fine-tuning models. However, on Tweets, the results for fine-tuning models are similar to or even worse than traditional deep models. The reason may be that tweets are non-standard and chatty compared with the corpora of pre-trained models. (2) The performance of applying prompt tuning method for text classification directly to sarcasm recognition is similar to or even worse than fine-tuning methods. This indicates that neither fine-tuning models nor regular prompt tuning method can make enough use of the knowledge in PLMs for sarcasm recognition.

When we look at the two types of SarcPrompt, we find that: (1) SarcPrompt with clash prompt outperform all the baseline methods. SarcPrompt with clash prompt improves a lot over regular prompt tuning, which demonstrates that it is improper to simply transplant the regular prompt tuning method to sarcasm recognition. By stimulating sarcastic characteristics, SarcPrompt is able to well exploit the capability of pre-trained language models in sarcasm recognition. (2) SarcPrompt with clash prompt performs better than with question prompt, showing clash prompt is more valid to reflect the contradictory intentions of sarcastic texts. (3) The improvements of SarcPrompt over baselines on Tweets are higher than that on both IAC-V1 and IAC-V2. The reason may be that the label words selection process relies on the seed words extracted from Tweets. It is necessary to explore other ways to obtain the prior information in the future.

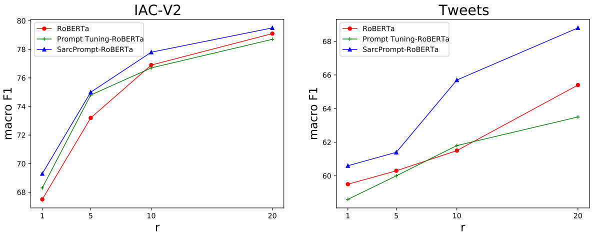


Figure 3: Macro $F1$ scores under low-resource settings, using 1%, 5%, 10%, and 20% instances for IAC-V2 and Tweets datasets.

6.2 Results under Low-resource Settings

In real-word scenes, it is often time-consuming and labor-intensive to collect annotated data to fine-tune classification models especially for difficult tasks like sarcasm recognition. To answer **RQ2**, we choose RoBERTa and Prompt Tuning-RoBERTa as baselines. For SarcPrompt, we use clash prompt type, which achieves the best performance under standard settings. IAC-V1 and IAC-V2 have the same source, and experimental results show the same trend under low-resource settings. So we report the results of IAC-V2 and Tweets datasets.

The experimental results are shown in Figure 3. We observe that: (1) Our SarcPrompt consistently outperforms the baseline methods with less training data on both datasets, which demonstrates the

Table 5: Cases of right and wrongly classified by SarcPrompt. The check mark indicates classification is correct, while the cross mark indicates wrong.

ID	Input	Prompt with predicted word	Result
1	Pretty excited about how you gave up on me.	Actually *not*.	✓
2	I just love being ignored	Actually *kidding*.	✓
3	thanks I thought it was tomorrow	Actually *indeed*.	✗
4	make sure you don't say Christmas!! The decorate for the holiday.	Actually *yes*.	✗

effectiveness of SarcPrompt model under low resource settings. (2) As **r** increases from 1 to 20, the improvement in our SacrPrompt over IAC-V2 decreases gradually. But for Tweets, with the increase of data, the improvement of SarcPrompt is also greater. The reason may be Tweets are short and non-standard, and may lack information. Also, SarcPrompt is based on the statistical information of Tweets. Therefore as the training data of Tweets increases, SarcPrompt can obtain more information and achieve better results, which is consistent with performance under standard settings.

6.3 Results of Different Prompts

To answer **RQ3**, we report the performance of different prompts of SarcPrompt model in Table 6. We observe that: (1) Among question prompts, T_{q_1} performs best on IAC datasets and T_{q_2} on Tweets. As for clash prompt, T_{c_2} performs best on IAC datasets while T_{c_1} on Tweets. It is worth noting that the best clash template on Tweets T_{c_1} is short and colloquial, which is similar to the usage of Tweets. On the contrary, T_{c_2} is a complete sentence. It matches IAC datasets, which are longer and more formal. (2) The effect of the template fluctuates more on IAC-V1 and Tweets datasets because they are relatively small and do not contain enough sarcastic patterns. An appropriate template can be well adapted to most of the data while an inappropriate template may perform poorly. This also shows that it is more difficult to find suitable templates for small datasets.

6.4 Ablation Study

To answer **RQ4**, we conduct an ablation study to analyze the impact of contrastive loss, which is shown in Table 7. Note that the removal of contrastive loss degrades the performance a lot, which indicates that the similarity information between the same class and the contrast information between different classes are significant in sarcastic expressions' learning.

Table 6: Performance (Macro $F1$) comparison of different prompt templates of SarcPrompt model. "AVG±VAR" means average results and the variances of question and clash prompts.

Prompt Type	IAC-V1	IAC-V2	Tweets
Question-T_{q_1}	**72.9**	**84.2**	73.1
Question-T_{q_2}	71.1	83.7	**73.7**
Question-T_{q_3}	69.4	84.1	69.3
AVG±VAR	71.1±3.1	84.0±0.1	72.0±5.7
Clash-T_{c_1}	73.4	83.3	**76.6**
Clash-T_{c_2}	**75.2**	**84.9**	75.3
Clash-T_{c_3}	70.8	83.5	73.8
Clash-T_{c_4}	71.7	83.4	72.9
Clash-T_{c_5}	73.0	84.6	73.7
AVG±VAR	72.8±2.8	83.9±0.6	74.5±2.2

Table 7: Performance (Macro $F1$) of SarcPrompt-Clash-RoBERTa with and without contrastive loss.

Model	IAC-V1	IAC-V2	Tweets
SarcPrompt-Clash-RoBERTa	**75.2**	**84.9**	**76.6**
w/o contrative loss	74.3	82.4	75.2

6.5 Case Study

To answer **RQ5**, we analyze sarcasm recognition results on Tweets by several cases in Table 5. We observe that: (1) Our SarcPrompt is good at dealing with input samples that contain subjective sentiment expressions, such as "excited" and "love" in the first and second samples. When combined with the prompt clause, the contradictory intention is obvious to recognize. (2) However, in the third and fourth samples, the contradictory intentions are not strong enough and hidden in deeper semantics. They require external contexts to assist sarcasm recognition. Specifically, the speaker remembers the wrong time in the third sample. In the fourth sample, the speaker complains the decoration is not good-looking for Christmas. This indicates that current templates and label words are not suitable for recognizing sarcasm in factual texts.

7 Conclusion

In this paper, we have proposed SarcPrompt, a simple and effective prompt tuning method to recognize sarcasm in texts. Specifically, we design several prompt templates to mimic the actual intention behind the sarcastic literal content. We define verbalizers based on the statistics of sarcastic corpus. Then it is able to determine whether the prompt is contradictory to the literal content by the predicted label words. Empirical experimental results show that SarcPrompt achieves state-of-the-art performance under both standard supervised settings and low-resource settings.

Limitations

Although our SarcPrompt has achieved the SOTA performance on several benchmark datasets for sarcasm recognition, there are still limitations, mainly in the following aspects.

Firstly, we combine sarcastic characteristics into prompt tuning in a hard-coded form by manually designing prompt templates and label words based on the statistical information in sarcastic corpora. This hard-coded approach may not be able to adapt to all sarcasm patterns and may miss some good prompt templates or label words. Moreover, in the current verbalizer mapping process, we decay the weight of each label word according to the relationship in the knowledge base. The mapping approach is trivial and not learnable. Lastly, as we analyzed in the case study, current SarcPrompt is not good at dealing with situational sarcasm. In situational sarcasm pattern, there is no contradictory intention by looking at the literal content alone, which is hard to recognize even for humans.

In future work, we will explore continuous prompt templates and learnable mapping functions for prompt tuning in sarcasm recognition. Combining external knowledge is also a direction to make prompt tuning suitable for situational sarcasm in our future work.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 62006218 and 61902381, the Youth Innovation Promotion Association CAS under Grants No. 20144310, and 2021100, the Young Elite Scientist Sponsorship Program by CAST under Grants No. YESS20200121, and the Lenovo-CAS Joint Lab Youth Scientist Project.

References

Nastaran Babanejad, Heidar Davoudi, Aijun An, and Manos Papagelis. 2020. Affective and contextual embedding for sarcasm detection. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 225–243.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

John D Campbell and Albert N Katz. 2012. Are there necessary conditions for inducing a sense of sarcastic irony? *Discourse Processes*, 49(6):459–480.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022a. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 2778–2788. ACM.

Yulong Chen, Yang Liu, Li Dong, Shuohang Wang, Chenguang Zhu, Michael Zeng, and Yue Zhang. 2022b. Adaprompt: Adaptive model training for prompt-based NLP. *CoRR*, abs/2202.04824.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3816–3830. Association for Computational Linguistics.

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th Workshop on Computational Approaches to*

Subjectivity, Sentiment and Social Media Analysis, WASSA@NAACL-HLT 2016, June 16, 2016, San Diego, California, USA, pages 161–169.

Aniruddha Ghosh and Tony Veale. 2018. Ironymagnet at semeval-2018 task 3: A siamese network for irony detection in social media. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2018, New Orleans, Louisiana, USA, June 5-6, 2018*, pages 570–575.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 581–586.

Roberto I. González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: A closer look. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 581–586.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: pre-trained prompt tuning for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8410–8423. Association for Computational Linguistics.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. PTR: prompt tuning with rules for text classification. *CoRR*, abs/2105.11259.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. CASCADE: contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1837–1848. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018a. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2018, New Orleans, Louisiana, USA, June 5-6, 2018*, pages 39–50.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018b. We usually don't like going to the dentist: Using common sense to detect irony on twitter. *Comput. Linguistics*, 44(4).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. 2022. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2225–2240. Association for Computational Linguistics.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jiangnan Li, Hongliang Pan, Zheng Lin, Peng Fu, and Weiping Wang. 2021. Sarcasm detection with commonsense knowledge. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3192–3201.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586.

Yijin Liu, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. 2019a. GCDT: A global context enhanced deep transition architecture for sequence labeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2431–2441.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yiyi Liu, Yequan Wang, Aixin Sun, Xuying Meng, Jing Li, and Jiafeng Guo. 2022. A dual-channel framework for sarcasm recognition by detecting sentiment conflict. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 1670–1680. Association for Computational Linguistics.

Chenwei Lou, Bin Liang, Lin Gui, Yulan He, Yixue Dang, and Ruifeng Xu. 2021. Affective dependency graph for sarcasm detection. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1844–1849.

Stephanie M. Lukin and Marilyn A. Walker. 2017. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. *CoRR*, abs/1708.08572.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn A. Walker. 2016. Creating and characterizing a diverse corpus of sarcasm in dialogue. In *Proceedings of the SIGDIAL 2016 Conference, The 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 13-15 September 2016, Los Angeles, CA, USA*, pages 31–41.

Braja Gopal Patra, Soumadeep Mazumdar, Dipankar Das, Paolo Rosso, and Sivaji Bandyopadhyay. 2016. A multilevel approach to sentiment analysis of figurative language in twitter. In *Computational Linguistics and Intelligent Text Processing - 17th International Conference, CICLing 2016, Konya, Turkey, April 3-9, 2016, Revised Selected Papers, Part II*, volume 9624 of *Lecture Notes in Computer Science*, pages 281–291. Springer.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet: : Similarity - measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004, Boston, Massachusetts, USA, May 2-7, 2004*. The Association for Computational Linguistics.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 5569–5578. International Committee on Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2339–2352. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4222–4235. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1010–1020.

Piyoros Tungthamthiti, Kiyoaki Shirai, and Masnizah Mohd. 2014. Recognition of sarcasms in tweets based on concept level sentiment analysis and supervised learning approaches. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation, PACLIC 28, Cape Panwa Hotel, Phuket, Thailand, December 12-14, 2014*, pages 404–413. The PACLIC 28 Organizing Committee and PACLIC Steering Committee / ACL / Department of Linguistics, Faculty of Arts, Chulalongkorn University.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. 2018. Thu_ngn at semeval-2018 task 3: Tweet irony detection with densely connected lstm and multi-task learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 51–56.

Tao Xiong, Peiran Zhang, Hongbo Zhu, and Yihui Yang. 2019. Sarcasm detection with self-matching networks and low-rank bilinear pooling. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2115–2124.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2324–2335. Association for Computational Linguistics.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5017–5033. Association for Computational Linguistics.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14612–14620. AAAI Press.

COMBO: A Complete Benchmark for Open KG Canonicalization

Chengyue Jiang[✿☾†], **Yong Jiang**[☆*], **Weiqi Wu**[✿], **Yuting Zheng**[✿],
Pengjun Xie[☆], **Kewei Tu**[✿☾*]
[✿]School of Information Science and Technology, ShanghaiTech University
[☾]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[☆]DAMO Academy, Alibaba Group, China
`{jiangchy,tukw,wuwq}@shanghaitech.edu.cn;`
`{yongjiang.jy,chengchen.xpj}@alibaba-inc.com`

Abstract

Open knowledge graph (KG) consists of *(subject, relation, object)* triples extracted from millions of raw text. The *subject* and *object* noun phrases and the *relation* in open KG have severe redundancy and ambiguity and need to be canonicalized. Existing datasets for open KG canonicalization only provide gold entity-level canonicalization for noun phrases. In this paper, we present **COMBO**, a **Com**plete **B**enchmark for **O**pen KG canonicalization. Compared with existing datasets, we additionally provide gold canonicalization for relation phrases, gold ontology-level canonicalization for noun phrases, as well as source sentences from which triples are extracted. We also propose metrics for evaluating each type of canonicalization. On the **COMBO** dataset, we empirically compare previously proposed canonicalization methods as well as a few simple baseline methods based on pretrained language models. We find that properly encoding the phrases in a triple using pretrained language models results in better relation canonicalization and ontology-level canonicalization of the noun phrase. We release our dataset, baselines, and evaluation scripts at `https://github.com/jeffchy/COMBO/tree/main`.

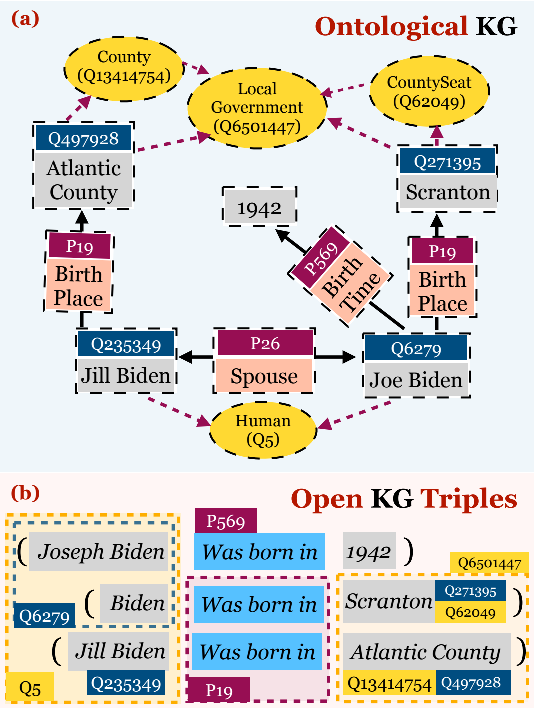


Figure 1: Example of ontological KG (a) and Open KG triples (b). The differently colored bounding boxes and the tags on the open KG triples illustrate three types of gold canonicalization. Yellow (e.g., *Q5* Human) shows the gold ontology-level NP cluster, the salvia blue (e.g., *Q6279*) indicates the gold entity-level NP cluster, and the purple (e.g., P19) indicates gold RP cluster.

1 Introduction

Large ontological knowledge graphs (KG) such as Wikidata (Vrandečić and Krötzsch, 2014), DBpedia (Bizer et al., 2009), Freebase (Bollacker et al., 2008) use a complex ontology to formalize and organize all the entities and relations. Figure 1(a) shows an example ontological knowledge graph (Wikidata): "*Joe Biden (Q6279)*" is categorized as "*Human (Q5)*" in Wikidata and linked to "*Scranton (Q271395)*" with relation "*birth place (P19)*", where prefix *Q* and *P* denote unique identities for entity and relation respectively in Wikidata[1]. As ontological KGs are well organized and canonicalized, one can efficiently query information and extract knowledge from them to assist NLP models in various tasks (Rao et al., 2013; Luo et al., 2015; Cui et al., 2019; Murty et al., 2018; Wang et al., 2021; Liu et al., 2023; Gao et al., 2022; Liu et al., 2022). However, building and maintaining an accurate ontological KG requires large human effort (Färber et al., 2015).

In contrast, open knowledge graphs such as Re-

[†] This work was done during Chengyue Jiang's internship at DAMO Academy, Alibaba Group.

[*] Yong Jiang and Kewei Tu are corresponding authors.

[1] Wikidata links `https://www.wikidata.org/wiki/Q5`, `https://www.wikidata.org/wiki/Property:P19`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 340–357
May 2-6, 2023 ©2023 Association for Computational Linguistics

Verb (Fader et al., 2011) and OLLIE (Mausam et al., 2012) are built using *(subject, relation, object)* triples automatically extracted from millions of raw text by OpenIE systems (Angeli et al., 2015; Fader et al., 2011; Mausam et al., 2012). They are frequently used to assist in building ontological KGs (Martinez-Rodriguez et al., 2018; Dessì et al., 2021) and slot filling (Broscheit et al., 2017). As OpenIE systems do not rely on pre-defined ontologies or human supervision, the extracted triples contain noun phrases (NPs) and relation phrases (RPs) that are not canonicalized. Take the open KG triples shown in Figure 1(b) as an example. The NP *"Joseph Biden"* and *"Biden"* both refer to the US president Joe Biden, but the open KG regards them as two different nodes because of their different surface forms. On the other hand, *"was born in"* in the first and second triple means *"birth place of"* and *"birth time of"* respectively, but the open KG cannot disambiguate them. These examples reveal the redundancy and ambiguity of uncanonicalized open KG (Vashishth et al., 2018), which makes querying open KG inaccurate and inefficient. To this end, open KG canonicalization aims to improve the quality of open KGs to the level of ontological KGs. It is therefore different from tasks such as entity linking (Rao et al., 2013) and KB aligning (Elsahar et al., 2018) that align entity mentions or sentences to an existing ontological KG.

Existing open KG canonicalization datasets such as ReVerb-base, ReVerb-ambiguous (Galárraga et al., 2014), ReVerb45K (Vashishth et al., 2018) and CanonicNELL (Dash et al., 2021) mainly focus on entity-level canonicalization of NPs, providing the gold **E**ntity-level **NP C**anonicalization (**NPC-E**). The blue tags and dashed boxes in Figure 1(b) show examples of **NPC-E**, e.g., *"Biden"* and *"Joseph Biden"* should be canonicalized as the same entity *Q6279*. However, these datasets do not provide the gold **RP C**anonicalization (**RPC**), and do not consider the **O**ntology-level **C**anonicalization of **NP** (**NPC-O**). **RPC** is to canonicalize RPs that mean same relation together, for example, the second and the third *"was born in"* in Figure 1(b) should be canonicalized into the same cluster of *birth place (P19)*, different from the first one which means *"birth time (P569)"*. Similarly, **NPC-O** is to canonicalize NPs that have same type together, for example, the *"Scranton"* should be canonicalized into class *"CountySeat"* and into class *"Local Government"* together with *"Atlantic County"*, it can be viewed as canonicalizing special ontological relations such as "*instance of*", "*subclass of*" represented by dotted arrows in Fig. 1(a). We formally define these tasks in Sec. 3.

RPC and **NPC-E** are important as parts of a canonicalization benchmark (1) Relations and ontology are necessary for an expressive KG (Klyne and Carroll, 2004) (2) Most KG queries involve relations and ontology (e.g., the query *"actress that was born in California"*, involve the relational constraint "*X, birth place, California*" and the ontological constraint "*X, instance of, Actress*").

In this paper, we present **COMBO**, a complete benchmark for open KG canonicalization consisting of three subtasks: besides **NPC-E** which has been adequately studied in previous work, we additionally provide gold **RPC** and **NPC-O** along with their evaluation metrics. Gold **NPC-O** is obtained by querying the Wikidata using SPARQL, and **RPC** is obtained by performing Stanford OpenIE on sentences from Wiki20 (a distantly labeled relation extraction dataset), and a per-instance human revision process to ensure the quality of extracted RPs. We introduce the data construction process detailedly in Sec. 4.

Our new benchmark makes it possible for the first time to quantitatively evaluate the full range of open KG canonicalization. We conduct comprehensive experiments to compare existing canonicalization methods as well as a few simple baseline methods proposed by us. Somewhat surprisingly, none of the existing methods utilizes pretrained contextualized word embedding, probably because previous work only focuses on NPC-E and NPs are often not very ambiguous, making contextualization not so helpful. For example, the "*Joe Biden*" and "*Joseph Biden*". However, contexts are more helpful in RPC and NPC-O. For RPC, relations are more ambiguous and diverse in surface forms (e.g., "*was born in*" in Figure 1) and contexts are needed for disambiguation. For NPC-O, the RP and the other NP in the triple will help understand the type of an NP. Therefore, our proposed baseline methods are based on pretrained language models (PLM) (Devlin et al., 2019; Liu et al., 2019b; Sun et al., 2019) which produce contextualized embedding and have been shown to contain a certain amount of factual knowledge (Petroni et al., 2019; Lauscher et al., 2020). We found that, after properly encoding triples and contexts, our baseline methods outperform well on all three subtasks compared with previous state-of-the-art methods, especially on RPC and NPC-O. We also propose a triple-based pretraining method and find that it further boosts the performance on all subtasks. Therefore, our work provides strong baselines for future research on open KG canonicalization.

In summary, our contributions are threefold. First, we propose a complete definition of the open KG canonicalization problem along with the metrics. Second, we construct the complete benchmark for open KG canonicalization consisting of entity-level and ontology-level NP canonicalization and RP canonicalization. Third, we propose a stronger baseline based on autoencoding PLMs and conduct a comprehensive empirical comparison of canonicalization methods on our benchmark.

2 Open KG Canonicalization Datasets

We introduce existing open KG canonicalization datasets and **COMBO**. The statistics of datasets are shown in Table 1.

ReVerb-Base (Galárraga et al., 2014) Constructed using the ReVerb open KB. As half of the NPs in ReVerb triples are linked to an entity in the ontological database FreeBase (Bollacker et al., 2008), the authors sample 150 FreeBase entities that have at least two surface forms, collect all triples containing these 150 entities, and use the entity labels as the gold NP clusters.

ReVerb-Ambiguous (Galárraga et al., 2014) ReVerb-Ambiguous is constructed similarly as ReVerb-Base, it has 37K triples, but with only 445 gold NP clusters (entities). One problem with the ReVerb-Base and ReVerb-Ambiguous datasets is they contain too few NP clusters and too many NP aliases, which is inconsistent with real open KGs.

ReVerb45K(Vashishth et al., 2018) ReVerb45K increases the entity number to 7.5K and has 45K triples in total. ReVerb45K, Reverb-Base, and ReVerb-Ambiguous extract a source sentence for each triple from ClueWeb09 Callan et al. (2009). However, some of the source sentences are simply the concatenation of triples.

CanonicNELL (Dash et al., 2021) Constructed using the open KB NELL (Mitchell and Fredkin, 2014) and the entity linking information for NPs (Pujara et al., 2013). They remove triples containing NPs without aliases. CanonicNELL does not provide source sentences.

COMBO (Ours) As shown in the Table 1, the main differences of our dataset between others are that we additional provide gold RP canonicalization and ontology-level NP canonicalization. Constructed based on the large Ontological KG Wikidata[2], the OpenIE system, a relation extraction dataset Wiki20m and human revisions, as detailed in next section. Our dataset contains 18K triples with their source sentences and we provide gold NPC-E, RPC and NPC-O annotations. We compare **COMBO** with existing datasets in Table 1. Although our dataset is middle-sized, it has the longest average triple length and the largest number of unique NPs, indicating the diversity of the surface forms of NPs and RPs. Providing source sentences of extracted OpenIE triples is natural but important since additional contextual information can be helpful in understanding and disambiguating NPs and RPs. We ensure all triples contain rich context, and the average length of source sentences is 21. We show some data samples in Appendix A, and analyze our data in Sec. 4.

3 Task Definition and Evaluation Metrics

Task Definition The goal of open KG canonicalization is to assign NPs and RPs in triples into clusters, such that NPs that refer to the same entity (NPC-E) or have the same type (NPC-O) are clustered together, and similarly, RPs that refer to the same relation are clustered together. Note that the task is unsupervised, meaning that the canonicalizer does not have access to gold annotations. We have N samples containing triples and their corresponding source sentences: $\mathcal{T} = \{c_i, t_i = (s_i, r_i, o_i)|\, i = 1 \ldots N\}$, where c_i is the i-th sentence, t_i is the i-th triple containing subject NP s_i, RP r_i, and object NP o_i. $\mathcal{S} = \{(s_i, i)|\, i = 1 \ldots N\}$ is the indexed subject NP set. The indexed RP set $\mathcal{R}$ and object NP set $\mathcal{O}$ are defined similarly as $\mathcal{R} = \{(r_i, i)|\, i = 1 \ldots N\}$ and $\mathcal{O} = \{(o_i, i)|\, i = 1 \ldots N\}$. We have $|\mathcal{S}| = |\mathcal{O}| = |\mathcal{R}| = N$. The gold NPC-E, RPC, and NPC-O annotations are defined as sets of clusters. As subject NPs and object NPs are asymmetric (Juffs and Harrington, 1995; McGinnis, 2002), we follow Vashishth et al. (2018) and evaluate the clusters of subject NPs and object NPs separately. The gold NPC-E and NPC-O for subject NPs are defined as *NPC-E (Subj)* $= \{C_1 \ldots C_{K_s^E}\}$, *NPC-O (Subj)* $= \{C_1 \ldots C_{K_s^O}\}$, where C_i denotes the i-th cluster of NP. The NPC-E and NPC-O

---

[2] The official suggested replacement of Freebase after it retired `shorturl.at/kmnBR`

	# NP	**# NPC-E**	**# RP**	**# RPC**	**# NPC-O**	# Triples	Avg triple len	Context (%)
ReVerb-Base	290	150	3K	✗	✗	9K	5.26	78%
ReVerb-Ambiguous	717	446	11K	✗	✗	37K	5.27	78%
ReVerb45K	15.5K	7.5K	22K	✗	✗	45K	6.17	91%
CanonicNELL	8.7K	1.4K	139	✗	✗	20K	6.38	✗
CoMBO (Ours)	16.5K	13.8K	3.2K	79	2946	18K	8.12	100%

Table 1: Statistics and comparison of open KG canonicalization datasets including ours. ✗ means not available in the dataset (i.e., zero). “Avg triple len” is the average number of words in the triple. The last column shows the ratios of triples containing additional context in their source sentences.

	NPC-E	*RPC*	*NPC-O*
Gold	non-overlapping	non-overlapping	overlapping
Predicted	non-overlapping	non-overlapping	non-overlapping / overlapping
Metric	Ma, Mi, Pair	Mi, Pair	Ma, Mi, Pair / $J_{g\to p}$, $J_{p\to g}$

Table 2: Evaluation of the three subtasks. Ma, Mi, Pair are abbreviations of macro, micro and pairwise metrics.

of object NPs are defined similarly, the gold RPC is defined as *RPC* as $\{C_1 \ldots C_{K_r}\}$. *NPC-E (Subj)* is a *non-overlapping* cluster assignment and satisfies two conditions: (1) $\bigcup_{i=1}^{K_s^E} C^i = \mathcal{S}$; (2) $C_i \cap C_j = \emptyset, i \neq j$. *NPC-E (Obj)* and *RPC* satisfy similar conditions. *NPC-O (Subj)* and *NPC-O (Obj)* are *overlapping* cluster assignments, i.e., we allow an NP to belong to multiple clusters, so they only need to satisfy the first condition. The task is to predict the cluster assignments of NPs and RPs given their source triples and sentences. Following previous works, we assume the cluster number is unknown beforehand and split our data into the dev (20%) and test (80%) sets.

Task Evaluation Most clustering algorithms such as K-means (Lloyd, 1982) and Hierarchical Agglomerative Clustering (HAC) (Maimon and Rokach, 2005) produce non-overlapping cluster assignments, and several algorithms (e.g., HAC) can also produce hierarchical and overlapping cluster assignment. For the *NPC-E* subtask, we adopt the classic macro, micro and pairwise metrics to compare the gold and predicted *NPC-E* cluster assignments (please refer to App. C for details). For *RPC*, the macro metrics that calculate the fractions of pure clusters are too strict because gold RP clusters are large and hence are unlikely to be pure. Therefore we only use the micro and pairwise metrics to evaluate *RPC*.

For *NPC-O*, the gold cluster assignments are overlapping. If the predicted clusters are non-overlapping, we can apply the macro and pairwise metrics and a modified micro metric (Appendix D). If the predicted clusters are overlapping, say $\mathcal{P} = \{C_1^p \ldots C_M^p\}$, we propose evaluation metrics $J_{g\to p}$ and $J_{p\to g}$ based on the Jaccard index (Jaccard, 1908; Tanimoto, 1958). $J_{g\to p}$ (Eq.1) calculates the average Jaccard index of a gold cluster and its best matched predicted cluster. $J_{p\to g}$ is similarly defined but with the roles of *NPC-O* and $\mathcal{P}$ switched. Table 2 summarizes the evaluation metrics of each subtask.

$$\begin{aligned} \text{Jaccard}(g,p) &= \frac{|g \cap p|}{|g \cup p|} \\ J_{g\to p} &= \frac{1}{|\mathit{NPC\text{-}O}|} \sum_{g \in \mathit{NPC\text{-}O}} \max_{p \in \mathcal{P}} \left(\text{Jaccard}(g,p)\right) \end{aligned} \tag{1}$$

4 Construction of Our Dataset

We illustrate the construction process of **CoMBO** in Figure 2. We rely on the Wiki20 dataset (Han et al., 2020) to obtain the source sentence and the gold NPC-E. Wiki20 is a large multi-domain relation extraction dataset constructed by aligning the Wikipedia corpus with Wikidata using distant supervision. As shown in the bottom of Figure 2, each sample of Wiki20 contains a sentence with the object and subject NP spans labeled and linked to entities in Wikidata and the relation between them is also labeled. To ensure data quality, we use the recently revised version of Wiki20 (Gao et al., 2021), which aligns the Wiki20 relation labels with the supervisedly constructed Wiki80 dataset (Han et al., 2019) and provides 56K human-annotated data samples. The object and subject NP spans and its entity linking information (e.g., *Q6275*) are from Wikipedia and have high precision, so we directly use it for task NPC-E.

Extracting Relational Phrases Wiki20 only provides the relation label of two NPs for each instance. We further extract RP for Wiki20 instances to obtain full open KG triples. We first discard samples with the relation label “NA” and then run the Stanford OpenIE system on Wiki20 sentences to extract triples. We choose Stanford OpenIE[3] because compared with older OpenIE systems such as ReVerb and NELL that are used in constructing previous datasets, Stanford OpenIE can leverage the linguistic structure of a sentence and generalizes better to out-of-domain and longer utterances (Angeli et al., 2015). We empirically find that Stanford OpenIE yields a better recall and can extract more triples per sentence compared to ReVerb. We use the default model configuration of Stanford OpenIE. After obtaining the OpenIE triples of each non-NA Wiki20 instance, we select the triples whose subject NP and object NP are consistent with the NP spans provided by Wiki20. This triple selection step ensures the NPs in the extracted triples have gold NPC-E annotations, and remove wrong relation spans caused by wrongly extracted head and tail entities. We filter out 88% of the original triples through this step. Although this step reduces noises caused by OpenIE, the extracted relation spans could still be wrong in two ways:

1. **Invalid RP between correct NPs**. For example, for sentence *“. . . the Althing, the ruling legislative body of Iceland . . . ”*, OpenIE wrongly extracts *(the Althing, body of, Iceland)*, while the true triple should be *(the Althing, ruling legislative body of, Iceland)*.

2. **Correct NPs and valid RP but RP does not imply the relation given by Wiki20**. For the given relation *mother of* and sentence *“. . . bart and lisa got sent out of the house by marge simpson . . . ”*, the extracted triple *(lisa, got sent out of the house by, marge simpson)* is valid but cannot imply the *mother of* relation.

Therefore, we manually check all the extracted triples for these two types of errors, correcting invalid relational phrase spans and removing triples whose RP cannot imply the given relation. We also standardize the form of RP (e.g., OpenIE sometimes includes “a” and “the” and sometimes does not). The detailed guidelines for the check and revision process are shown in the Appendix B. The error analysis is shown in Table 3.

After all these steps, we obtain an open KG consisting of 18K triples. Similar to NPC-E, we use the relation labels given by the Wiki20 annotations (e.g., *P19*) as the gold RPC. As shown in Figure 3, the constructed open KG contains 79 relations in various domains, such as relations between geopolitical entities (*mouth of the watercourse* (7.3%), *mountain range* (3.8%), etc), relations between people (*spouse of* (1.7%), *child of* (1.6%), etc), and various relations between people and other objects (*citizenship* (2.4%), *work location* (3.7%), etc). The extracted RPs are diverse in surface forms. The number of distinct RPs is 3.2K. We show RP examples in Table 4. There exist some RPs that represent multiple relations and one representative example is *“in”*.

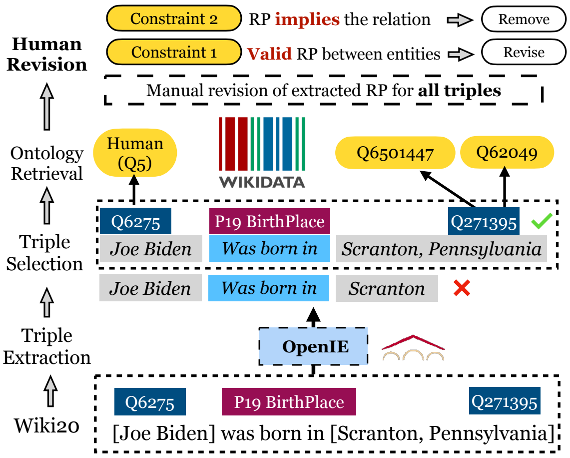


Figure 2: Steps of dataset construction.

Error Type	Rate
Invalid RP	23.5%
RP doesn’t imply relation	5.5%

Table 3: OpenIE error analysis.

Extracting Ontology To obtain ontology-level NP clusters for the NPC-O subtask, we query Wiki-

spouse of	*was married twice to , was married to, lover, consort of, second husband, widow of ’s wife, ’s second wife, arranged a wedding with*
mountain range	*peak, large nunatak, summits of, the only crossing of the most prominent feature of, small glacier, summits in valley in, only crossing of, northernmost subrange of,* ***in***
location	*is headquartered in, moved to, is carved on took place at, ironworks in, was again held at,* ***in***

Table 4: RP examples.

---

[3] https://stanfordnlp.github.io/CoreNLP/

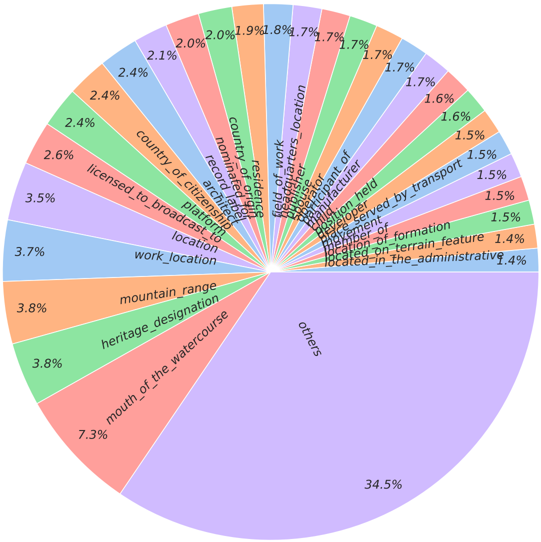


Figure 3: Pie charts of 79 RP clusters.

data for the classes of each entity. For example, to obtain the classes of *"Joe Biden (Q6275)"*, we run the SPARQL (RDF query language) query " `Q6275 P31 ?`", where P31 represents the "instance of" relation in Wikidata. This query obtains all the classes of an NP. If an NP does not have a class, its NPC-O annotation is the same as its NPC-E annotation. If an NP has more than one class, we include all of them in the NPC-O annotation (e.g., *city* and *big city* for "New York"). We query Wikidata using a third-party client Wikidata Integrator[4].As the ontology information in Wikidata is crowdsourced and contains errors, we apply pattern-based corrections to the extracted ontological NP clusters, for example, if an NP belongs to the cluster *million cities*, it should also belong to the cluster *city*. The resulting 2.9K ontological NP clusters form a 6-level overlapping hierarchy which allows a node to have more than one parent. We illustrate part of the hierarchy in Figure 5 and show the statistics of the top 12 ontological NP clusters in Figure 4.

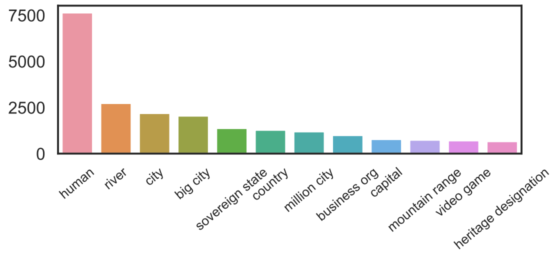


Figure 4: Size of top 12 ontological NP clusters.

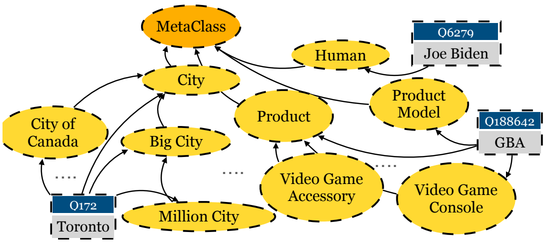


Figure 5: Part of the class hierarchy in our dataset.

5 Comprehensive Evaluation of Methods

Our benchmark makes it possible to conduct a comprehensive empirical comparison of different methods on the full range of open KG canonicalization. Below we first give an overview of existing methods and propose a few new baseline methods. Then we present our experimental settings and results.

5.1 Previous Methods

Non-neural Methods Galárraga et al. (2014) utilizes token features such as TF-IDF scores and Jaccard token similarity to canonicalize NPs. They merge similar NPs based on a threshold (tuned on the validation set) to form clusters. For RPs, they use AMIE (Galárraga et al., 2013), an unsupervised algorithm based on statistical rule mining, to obtain relation clusters. Vashishth et al. (2018) use additional side information obtained from various sources (such as PPDB (Ganitkevitch et al., 2013), WordNet (Miller, 1992)) to produce clusters.

SE-HAC Trivial baseline that performs HAC clustering over phrase embeddings produced by averaging static word embeddings such as GloVe (Pennington et al., 2014) or random embeddings.

CESI Vashishth et al. (2018) encode phrases using the same method as in SE-HAC; then apply the HolE graph embedding algorithm (Nickel et al., 2016) on triples and incorporate side information to finetune the embedding, and finally run HAC clustering on the learned embeddings.

CUVA Dash et al. (2021) adopt VAEGMM (Jiang et al., 2017) to jointly learn and cluster the embeddings. They initialize VAEGMM by performing HAC clustering on GloVe NP and RP embeddings, and then simultaneously optimize the knowledge embedding loss, side information loss, and VAE loss for the final clustering.

Previous methods encode NPs and RPs using either token frequency features or static word embedding. Although CESI and CUVA learn graph embedding on open KG triples, they assign the

---

[4] `https://github.com/SuLab/WikidataIntegrator`

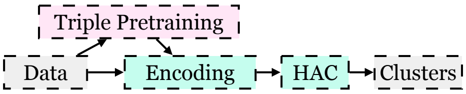


Figure 6: Pipeline of the proposed PLM-based method.

same representation to phrases with the same surface form and therefore cannot deal with ambiguity. No method utilizes original sentences to provide additional contexts. HAC is a popular choice of the clustering algorithm because it does not require knowing the number of clusters, but instead requires a distance threshold indicating when to stop merging. Unlike the number of clusters, the threshold can be tuned on a validation set and directly applied to the test set.

5.2 PLM-Based Baseline Methods

We propose a set of new baseline methods based on PLMs that produce contextualized embedding. We use a pipeline similar to CESI as shown in Figure 6. We encode NPs and RPs using different PLMs, PLM layers, and span representation methods and apply HAC clustering over their representations. We use the cosine similarity as the distance function and apply the complete linkage variant of HAC clustering because we prefer compact clusters and the single linkage variant suffers from the chaining phenomenon. Before encoding, an optional triple-level continuous pretraining step can be applied for better canonicalization. Token similarity and other side information are not used in our PLM-based method, but we generate them for our data using the code provided by Vashishth et al. (2018) to facilitate running of other methods.

5.2.1 Encoding

PLMs We use autoencoding PLMs[5] including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), ERNIE2.0 (Sun et al., 2019) which integrates entity information, and SpanBERT (Joshi et al., 2020) which is pretrained with span masks.

Input Given a triple $t_i = (s_i, r_i, o_i)$ and its corresponding source sentence c_i, we formulate the input of PLM in the following four ways to obtain the contextualized embedding of words in the NPs and RP. Note that the fourth method *sep* independently encodes each phrase in the triple.

$$\begin{aligned} \textit{sentence:}\;& [CLS]\ \ldots s_i \ldots r_i \ldots o_i \ldots\ [SEP] \\ \textit{triple:}\;& [CLS]\ s_i\ r_i\ o_i\ [SEP] \\ \textit{triple-sep:}\;& [CLS]\ s_i\ [SEP]\ r_i\ [SEP]\ o_i\ [SEP] \\ \textit{sep:}\;& [CLS]\ s_i/r_i/o_i\ [SEP] \end{aligned} \tag{2}$$

Phrase Representation After obtaining the contextualized embedding of words in an NP or RP span, denoted as $h_i \ldots h_j$, we follow Toshniwal et al. (2020) and use three methods to produce a single span embedding. Following Timkey and van Schijndel (2021), we also standardize the embeddings to remove rogue dimensions (Appendix E). Previous work (Vulić et al., 2020; Liu et al., 2019a) shows that different layers of a PLM contain different information, so we investigate contextualized embedding from different layers .

$$\begin{aligned} \textit{mean:}\;& average\_pooling(h_i \ldots h_j) \\ \textit{max:}\;& max\_pooling(h_i \ldots h_j) \\ \textit{diff-sum:}\;& [h_i - h_j; h_i + h_j] \end{aligned} \tag{3}$$

5.2.2 Triple-level Pretraining

Inspired by the HolE algorithm used in previous works (Vashishth et al., 2018; Dash et al., 2021), we may perform an optional triple-level continuous pretraining step before encoding to mimic the link prediction objectives in knowledge graph embedding learning. For each sentence in our dataset, we randomly mask a phrase in the triple and then train the PLM to predict the whole masked span. We perform pretraining for 10 epochs using the AdamW optimizer (Loshchilov and Hutter, 2019) with a linear scheduler and a start learning rate of 5e-5. We then use the continuously pretrained version of PLM as the phrase encoder. We also use the causal subword-level MLM strategy in BERT (Devlin et al., 2019) for comparison.

5.3 Experimental Setup

For each subtask, we use grid search to tune the HAC distance threshold on the dev set to obtain non-overlapping clusters for all the methods. We select the best threshold based on the average of the metrics shown in Table 2. We obtain overlapping clusters for NPC-O from the full HAC hierarchy. As HAC is deterministic, we run experiments once for methods without randomness and four times for methods involving random initialization (CUVA, Random+HAC). As Token Sim+SI and VAEGMM based methods cannot provide overlapping cluster assignments, we do not evaluate them by metrics based on the Jaccard index. For our PLM-based

---

[5] Huggingface models `https://huggingface.co/`

	NPC-E		***RPC***	***NPC-O***			
	Subj	***Obj***	***Relation***	***Subj***	***Subj-Jaccard***	***Obj***	***Obj-Jaccard***
Token Sim+SI (Galárraga et al., 2014)	82.90	79.35	33.94	33.14	-	40.59	-
Random+HAC	**85.32**	85.11	35.98	37.31	38.79	44.90	44.40
GloVe+HAC	78.31	86.57	35.57	38.00	39.04	47.35	45.45
GloVe+HolE+HAC (Vashishth et al., 2018)	80.13	87.33	17.93	39.63	39.87	48.25	47.18
GloVe+SI+HAC (Vashishth et al., 2018)	80.42	**87.34**	17.91	39.87	39.83	**48.26**	47.11
CESI (Vashishth et al., 2018)	80.11	86.82	18.37	**39.93**	**39.89**	48.25	**47.19**
VAEGMM+SI (Dash et al., 2021)	80.86	82.91	34.10	37.41	-	46.90	-
VAEGMM+HolE (Dash et al., 2021)	80.15	82.87	36.12	37.22	-	46.88	-
CUVA (Dash et al., 2021)	80.68	82.95	**36.13**	37.09	-	46.89	-
Bert-base	**86.93**	86.91	54.47	**42.97**	**44.16**	50.71	46.99
Roberta-base	82.85	85.00	41.08	39.21	41.24	46.07	44.14
SpanBert-base	84.38	86.32	44.04	41.61	43.16	47.29	45.35
ERNIE2.0-base	86.68	**88.11**	**54.66**	42.60	44.05	**52.27**	**47.05**
Bert-base-triple	86.01	**88.92**	**58.45**	**43.71**	**45.19**	**51.78**	**47.49**
Roberta-base-triple	85.37	87.22	50.81	42.29	44.33	50.31	46.81
SpanBert-base-triple	85.73	85.89	46.18	42.53	44.23	47.60	45.56
ERNIE2.0-base-triple	**87.21**	86.93	57.28	43.21	44.33	50.66	47.27
Bert-large	**87.09**	**89.05**	**50.31**	42.34	**44.16**	50.71	**47.25**
Roberta-large	83.50	85.81	40.51	39.88	42.35	48.35	45.58
SpanBert-large	86.32	86.67	45.84	40.96	42.90	47.92	45.68
ERNIE2.0-large	86.21	88.86	49.80	**42.71**	44.01	**51.98**	47.22

Table 5: Averaged metrics (of Table 2) on all the subtasks. For example, ***NPC-O, Subj*** is the average of Ma,Mi and Pair metrics on the ontology-level canonicalization of subject NPs, and ***NPC-O, Obj-Jaccard*** is the average of $J_{p\rightarrow g}$ and $J_{g\rightarrow p}$ for the overlapping clustering assignment of object NPs. Full results including the results of *large-triple* models are shown in Appendix F

methods, we select the best input form and span representation strategy based on the dev set performance. We also compare different **encoding strategies** in Appendix G, and **layer-wise performances** in Appendix H.

5.4 Overall Results

We report averaged metrics for each subtask in Table 5 because of limited space. The full results are shown in Appendix F. The results show that our PLM-based baseline methods outperform previous methods in most cases, especially on RPC and NPC-O, indicating the importance of contextual information. Trivial baselines such as Token Sim+SI, Random+HAC and GloVe+HAC already perform well (around 80%) on NPC-E, because NPs referring to the same entity usually have similar surface forms and do not have to rely on contexts for correct prediction. However, they perform badly on RPC and NPC-O, because surface forms alone are no longer adequate for these two subtasks because of higher ambiguity. CESI has bad RPC performance but is very competitive on NPC-E (Obj), and better than SpanBERT and RoBERTa without triple-level pretraining, but is still worse than the other PLM-based methods. CUVA performs generally badly, probably because it is sensitive to VAEGMM initialization and relies heavily on side information. As our dataset has the longest average triple length and consists of texts from various domains, it could be more challenging for methods that do not use contextualized embedding.

For PLM-based methods, BERT leads to the best overall performance on NPC-E (Obj), RPC and NPC-O (Subj); ERNIE2.0 performs best on NPC-E (Subj) and NPC-O (Obj) and is comparable to Bert on NPC-E (Obj) and RPC; RoBERTa and SpanBERT fall behind, but are still better than most other non-PLM methods on NPC-O and RPC. Large PLMs are better than base PLMs on NPC-E, comparable on NPC-O, but worse on RPC. We also found the triple-level pretraining effective, having a positive influence in most cases, especially on RPC (e.g., +9.73 for RoBERTa). In contrast, using the causal subword-level pretraining for Bert improves the object NPC but harms the subject NPC and RPC (-1.07 points). A detailed comparison between triple-level and subword-level pretraining is shown in Appendix I.

6 Conclusion

We present **COMBO**, a complete benchmark for open KG canonicalization. **COMBO** consists of three subtasks, entity-level and ontology-level NP canonicalization, and RP canonicalization. We construct the data and propose the evaluation metrics for the RPC and NPC-O that are not been adequately studied before. We also propose a stronger canonicalization method based on autoencoding PLMs and conduct a comprehensive comparison of different canonicalization methods on our dataset.

For future study, NPC-O and RPC still have a lot of room for improvement and the efficiency of canonicalization methods is also worth studying. We also note that **COMBO** can be additionally used as a probing benchmark for PLMs and as a phrase-level relation classification dataset.

7 Acknowledgement

This work was supported by the National Natural Science Foundation of China (61976139) and by Alibaba Group through Alibaba Innovative Research Program.

Limitations

One limitation of our work is that, the size of our dataset (18K) is relatively small compared to previous datasets (Table 1). Another limitation is that, similar to previous work, we perform clustering for three subtasks and evaluate the canonicalization results independently, but canonicalization of the head NP, tail NP and RP is in fact closely correlated. For example, the NPC-O clusters of the head NP and tail NP reveal the domain and range of the relation given by RPC. We leave jointly canonicalization and evaluation as future work. Our proposed baseline is straightforward. We encourage future studies to investigate better canonicalization methods based on pretrained language models.

Ethics Statement

Our dataset is constructed based on Wiki20 and Wikidata. The two sources are both publicly available. Wiki20 is under the MIT Licence and the Wikidata is under the Creative Commons CC0 License. Both of them allow modification and distribution. Regarding human revision during dataset construction, the annotators were properly paid. The annotating procedure lasted 12 days and the daily workload was relatively light: around 2.5 hours per day. During human inspection, we did not identify any unethical instances in our dataset. Regarding baseline models, we use PLMs as our text encoder and our task is inherently unsupervised. As PLMs are learned on large corpora, our method can potentially create biased clustering results. How to de-bias PLM embedding is worth further investigation.

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China. Association for Computational Linguistics.

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. Dbpedia - a crystallization point for the web of data. *Journal of Web Semantics*, 7(3):154–165. The Web of Data.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Samuel Broscheit, Kiril Gashteovski, and Martin Achenbach. 2017. Openie for slot filling at tac kbp 2017-system description. In *TAC*.

Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. 2009. Clueweb09 data set.

Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. 2019. Kbqa: learning question answering over qa corpora and knowledge bases. *arXiv preprint arXiv:1903.02419*.

Sarthak Dash, Gaetano Rossiello, Nandana Mihindukulasooriya, Sugato Bagchi, and Alfio Gliozzo. 2021. Open knowledge graphs canonicalization using variational autoencoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10379–10394.

Danilo Dessì, Francesco Osborne, Diego Reforgiato Recupero, Davide Buscaldi, and Enrico Motta. 2021. Generating knowledge graphs by employing natural language processing and machine learning techniques within the scholarly domain. *Future Generation Computer Systems*, 116:253–264.

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

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Michael Färber, Basil Ell, and Carsten Menne. 2015. A comparative survey of dbpedia , freebase , opencyc , wikidata , and yago.

Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M. Suchanek. 2014. Canonicalizing open knowledge bases. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 1679–1688. ACM.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 413–422. International World Wide Web Conferences Steering Committee / ACM.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Tianyu Gao, Xu Han, Yuzhuo Bai, Keyue Qiu, Zhiyu Xie, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Manual evaluation matters: Reviewing test protocols of distantly supervised relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1306–1318, Online. Association for Computational Linguistics.

Yunjun Gao, Xiaoze Liu, Junyang Wu, Tianyi Li, Pengfei Wang, and Lu Chen. 2022. Clusterea: Scalable entity alignment with stochastic training and normalized mini-batch similarities. In *KDD*, pages 421–431.

Adam Grycner, Gerhard Weikum, Jay Pujara, James Foulds, and Lise Getoor. 2015. RELLY: Inferring hypernym relationships between relational phrases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 971–981, Lisbon, Portugal. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 745–758, Suzhou, China. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 169–174, Hong Kong, China. Association for Computational Linguistics.

Paul Jaccard. 1908. Nouvelles recherches sur la distribution florale. *Bull. Soc. Vaud. Sci. Nat.*, 44:223–270.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017. Variational deep embedding: An unsupervised and generative approach to clustering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1965–1972. ijcai.org.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Alan Juffs and Michael Harrington. 1995. Parsing effects in second language sentence processing: Subject and object asymmetries in wh-extraction. *Studies in second language acquisition*, 17(4):483–516.

Graham Klyne and Jeremy J. Carroll. 2004. Resource description framework (rdf): Concepts and abstract syntax. W3C Recommendation.

Anne Lauscher, Olga Majewska, Leonardo F. R. Ribeiro, Iryna Gurevych, Nikolai Rozanov, and Goran Glavaš. 2020. Common sense or world knowledge? investigating adapter-based knowledge injection into pretrained transformers. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 43–49, Online. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Shuliang Liu, Xuming Hu, Chenwei Zhang, Shuang Li, Lijie Wen, and Philip S. Yu. 2022. Hiure: Hierarchical exemplar contrastive learning for unsupervised relation extraction. *ArXiv*, abs/2205.02225.

Xiaoze Liu, Junyang Wu, Tianyi Li, Lu Chen, and Yunjun Gao. 2023. Unsupervised entity alignment for temporal knowledge graphs. In *WWW*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Stuart Lloyd. 1982. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888, Lisbon, Portugal. Association for Computational Linguistics.

Oded Maimon and Lior Rokach. 2005. Data mining and knowledge discovery handbook.

Jose L Martinez-Rodriguez, Ivan López-Arévalo, and Ana B Rios-Alvarado. 2018. Openie-based approach for knowledge graph construction from text. *Expert Systems with Applications*, 113:339–355.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, Jeju Island, Korea. Association for Computational Linguistics.

Martha McGinnis. 2002. Object asymmetries in a phase theory of syntax.

George A. Miller. 1992. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Tom Mitchell and Edward Fredkin. 2014. Never ending language learning. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 1–1.

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109, Melbourne, Australia. Association for Computational Linguistics.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 1955–1961. AAAI Press.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jay Pujara, Hui Miao, Lise Getoor, and William Cohen. 2013. Knowledge graph identification. In *International Semantic Web Conference*, pages 542–557. Springer.

Delip Rao, Paul McNamee, and Mark Dredze. 2013. Entity linking: Finding extracted entities in a knowledge base. In *Multi-source, multilingual information extraction and summarization*, pages 93–115. Springer.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Taffee T Tanimoto. 1958. Elementary mathematical theory of classification and prediction.

William Timkey and Marten van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. *arXiv preprint arXiv:2109.04404*.

Shubham Toshniwal, Haoyue Shi, Bowen Shi, Lingyu Gao, Karen Livescu, and Kevin Gimpel. 2020. A cross-task analysis of text span representations. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 166–176, Online. Association for Computational Linguistics.

Shikhar Vashishth, Prince Jain, and Partha P. Talukdar. 2018. CESI: canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1317–1327. ACM.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

A Dataset Examples

We show examples of our dataset in Figure 7.

B Guidelines for Revising Relational Phrases

B.1 RP Annotating Procedure

1. We first split all triples by relations and form 79 json files for two major paid annotators, each annotator is responsible for around 40 relations.
2. Annotators should check one relation file at a time for annotating consistency, and start the next one after the former one is finished.
3. For each relation, annotators are given: (a) The original sentences of the relation with markers indicating the head NP, tail NP and RP extracted by OpenIE. (b) The name (e.g., *composer*), and the Wikidata ID (e.g., *P86*) of the gold relation.
4. Annotators should first understand the relation by querying the Wikidata, take the relation *"composer (P86)"* as an example, annotators should first query Wikidata through the link `https://www.wikidata.org/wiki/Property:P86` to obtain the definition of the relation and skim through example relational phrases of RPs. The Figure 8 shows the Wikidata page containing the definition and examples of *"composer"*.
5. After fully comprehend the relation, annotators can start to check and revise triples in each file, the details and examples of RP revision and justification of if RP implies the relation are shown in the next two subsection (Appendix B.2, Appendix B.3).
6. After two annotators finished their part, we randomly sample 100 samples from each part and ask the annotator responsible for the other part to check. The annotators reached a consensus for approving 97% of these samples.

B.2 Guideline for checking the validity of RPs and revision

Definition of relational phrases Relational phrases are textual representations of relations between named entities (Grycner et al., 2015), we follow ReVerb (Fader et al., 2011) and require the relational phrases be continuous span in the sentence. We summarize most cases of relational phrases occurring in our dataset, and the guideline for annotating each case. The annotated RPs are shown in pink.

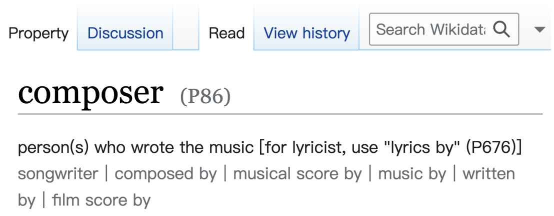
- **Case 1: Verb** example: *[A] married [B]*, include different tenses of verbs.
- **Case 2: Verb+preposition** example: *[A] located at [B]*.
- **Case 3: Passive voice** example: *[A] is designed by [B]*, *[A] is headquartered in [B]* the linking verb is sometimes omitted: *[A], designed by [B]*.
- **Case 4: When the tail entity is the appositive of the head entity** example: *[A] 's son is [B]*, *[A] 's son , [B]*, *[A] 's masterpiece , [B]*, the content between the appositives is usually informative and regarded as RPs.
- **Case 5: Compound predicate** A common case is that the tail NP is a part of the compound predicate of the head NP, e.g., *[A] is the civil branch of [B]*, *[A] is an agency of [B]*, *[A] is a novel by [B]*. When encountering these cases, we do not include the linking verb and article because they are not informative (e.g., *"is the"*).
- **Case 6: Cases omit verb** Some cases omit verb, we only annotate the preposition. For example, *[yokosuka arts theatre], part of the bay square complex by [kenzou tange]*, this sentence omits the verb *"built"*.

```
{'h': {'id': 'Q533336', 'instance': ['Q18812508'], 'name': ['node', '1'], 'pos': [16, 18]}, 'r':
{'label': 'manufacturer', 'name': ['built', 'by'], 'pos': [19, 21]}, 't': {'id': 'Q66', 'instance':
['Q891723', 'Q4830453', 'Q936518', 'Q6881511', 'Q2538889', 'Q2995256'], 'name': ['boeing'], 'pos':
[21, 22]}, 'text': ['this', 'was', 'followed', 'in', 'december', 'by', 'the', 'first', 'u.s.',
'module', ',', '`', 'unity', '`', 'also', 'called', 'node', '1', ',', 'built', 'by', 'boeing', 'in',
'facilities', 'at', 'msfc', '.']}
```

```
{'h': {'id': 'Q18391244', 'instance': ['Q13406463', 'Q105543609'], 'name': ['motets'], 'pos': [18,
19]}, 'r': {'label': 'composer', 'name': ['are', 'composed', 'by'], 'pos': [19, 21]}, 't': {'id':
'Q81752', 'instance': ['Q5'], 'name': ['anton', 'bruckner'], 'pos': [21, 23]}, 'text': ['two',
'asperges', 'me', ',', 'wab', '3', 'the', 'two', 'thou', 'wilt', 'sprinkle', 'me', ',', 'wab', '3',
',', 'are', 'sacred', 'motets', 'composed', 'by', 'anton', 'bruckner', '.']}
```

```
{'h': {'id': 'Q3764815', 'instance': ['Q47461344'], 'name': ['pedda', 'bala', 'siksha'], 'pos': [0,
3]}, 'r': {'label': 'language of work or name', 'name': ['is', 'encyclopedia', 'in'], 'pos': [3,
7]}, 't': {'id': 'Q8097', 'instance': ['Q34770', 'Q1288568'], 'name': ['telugu'], 'pos': [8, 9]},
'text': ['pedda', 'bala', 'siksha', 'is', 'an', 'encyclopedia', 'in', 'the', 'telugu', ',',
'suitable', 'for', 'children', 'and', 'adults', '.']}
```

Figure 7: Examples of our dataset, "h" means head or subject NP, "r" means relation, "t" means tail or object NP. "instance" stands for the gold ontology-level clusters.



composer (P86)

person(s) who wrote the music [for lyricist, use "lyrics by" (P676)]

songwriter | composed by | musical score by | music by | written by | film score by

Figure 8: Wikidata query example.

We also provide several revision examples of wrong OpenIE triples, part of them are shown in the Table 6 below.

B.3 Guideline for justifying if RP implies the given relation

As we stated in the fourth step of the overall annotating process in the Appendix B.1, we require annotators to fully understand the meaning of the given relation. For each triple, annotators should ask themselves if the relational phrase could express the relation of the head and tail NP in the given context sentence. Note that we don't require the relation could be solely implied by the RP, for example, given the triple and its context: *"[mount elbert]* ***in*** *the [sawatch range] is the highest summit of the rocky mountains"*, it is impossible to infer the relation by the RP *"in"*, but RP is a reasonable text representation of the relation *mountain range* in this context. We found that the extracted RPs can imply the relation in most cases, we show some concrete bad cases to the annotators to help them identify the bad RPs, part of the examples are shown in the Table 7.

C Classic Metrics

Gold cluster assignment: $\mathcal{G} = \{C_1^g \dots C_K^g\}$, predicted cluster assignment: $\mathcal{P} = \{C_1^p \dots C_M^p\}$, where C_i^g and C_j^p are gold and predicted cluster respectively.

Macro Metrics

$$P_{macro}(\mathcal{G}, \mathcal{P}) = \frac{|\{p \in \mathcal{P} | \exists g \in \mathcal{G}, p \subseteq g\}|}{|\mathcal{P}|}$$
$$R_{macro}(\mathcal{G}, \mathcal{P}) = P_{micro}(\mathcal{P}, \mathcal{G}) \tag{4}$$

Micro Metrics

$$P_{micro}(\mathcal{G}, \mathcal{P}) = \frac{\sum_{g \in \mathcal{G}} \max_{p \in \mathcal{P}} |g \cap p|}{N}$$
$$R_{micro}(\mathcal{G}, \mathcal{P}) = P_{micro}(\mathcal{P}, \mathcal{G}) \tag{5}$$

Where N is the total number of different phrases that appear in $\mathcal{G}$ (or $\mathcal{P}$).

Pairwise Metrics

$$P_{pair}(\mathcal{G}, \mathcal{P}) = \frac{\sum_{p \in \mathcal{P}} |\{(u, u') \in p, \exists g \in \mathcal{G}, \forall (u, u') \in p\}|}{\sum_{p \in \mathcal{P}} C_2^{|p|}}$$
$$R_{pair}(\mathcal{G}, \mathcal{P}) = \frac{\sum_{p \in \mathcal{P}} |\{(u, u') \in p, \exists g \in \mathcal{G}, \forall (u, u') \in p\}|}{\sum_{g \in \mathcal{G}} C_2^{|g|}} \tag{6}$$

For more details about classic metrics, please refer to the Sec. 7.2 of the CESI paper (Vashishth et al., 2018).

Bad OpenIE RP	*... [the generalitat de catalunya], the governing **body of** [catalonia], approved ...*
Revised	*... [the generalitat de catalunya], the **governing body of** [catalonia], approved ...*
Bad OpenIE RP	*... the "[althing]" , the ruling legislative **body of** [iceland]*
Revised	*... the "[althing]" , the **ruling legislative body of** [iceland]*
Bad OpenIE RP	*... [t-d center] dominion **centre**, designed by [ludwig mies van der rohe] ...*
Revised	*... [t-d center] dominion centre, **designed by** [ludwig mies van der rohe] ...*
Bad OpenIE RP	*[ace attorney investigations 2] ... and features **character designs by** tatsuro iwamoto and music by [noriyuki iwadare]*
Revised	*[ace attorney investigations 2] ... and features character designs by tatsuro iwamoto and **music by** [noriyuki iwadare]*
Bad OpenIE RP	*[A] is a **farcical musical comedy with** music by [walter alfred slaughter]*
Revised	*[A] is a farcical musical comedy with **music by** [walter alfred slaughter]*
Bad OpenIE RP	*[beta cygni a] **is a bright** star from the constellation [cygnus]*
Revised	*[beta cygni a] is a **bright star from** the constellation [cygnus]*
Bad OpenIE RP	*... [daya district], taichung, **taiwan in** the [chinese taipei]*
Revised	*... [daya district], taichung, taiwan **in** the [chinese taipei]*

Table 6: Examples of revising RPs in OpenIE triples.

P1001 *applies to jurisdiction*	*...the process to amend the constitution cannot be initiated in times of war or when the [belgian federal parliament] **is unable to freely meet in** [belgium]*
P84 *architect*	*[u. b. city], the **headquarters of** the [united breweries group], is a high-end commercial zone.*
P40 *child*	*he was a great-grandson of [berge sigval natanael bergeson], **grand-naphew of** [ole bergeson]*
P25 *mother*	*bart and [lisa], **sent out of** the house by [marge simpson] ...*

Table 7: Examples of bad RPs that cannot imply the given relations.

D Extension of Micro Metrics

Gold overlapping clusters: $\mathit{NPC\text{-}O} = \{C_1^O \dots C_{K_3}^O\}$, predicted clusters $\mathcal{P} = \{C_1^p \dots C_M^p\}$.

$$
\begin{aligned}
P_{micro}(\mathit{NPC\text{-}O}, \mathcal{P}) &= \frac{\sum_{g \in \mathit{NPC\text{-}O}} \max_{p \in \mathcal{P}} |g \cap p|}{\sum_{g \in \mathit{NPC\text{-}O}} |g|} \\
R_{micro}(\mathit{NPC\text{-}O}, \mathcal{P}) &= P_{micro}(\mathcal{P}, \mathit{NPC\text{-}O})
\end{aligned}
\tag{7}
$$

We modify the denominator compared to the micro metric in (Vashishth et al., 2018).

E Standardization

Following Timkey and van Schijndel (2021), we perform standardization for phrase embeddings to remove the rogue dimensions. Denote $\boldsymbol{E}_{\mathcal{R}} \in \mathbb{R}^{N \times D}$ as the embedding matrix of all RP phrases, where N is the number of triples, and D is the dimension of contextual embedding. The standardized RP embedding matrix $\boldsymbol{E}'_{\mathcal{R}}$ is:

$$
\begin{aligned}
\boldsymbol{\mu} &= \frac{1}{N} \sum_{i}^{N} \boldsymbol{E}_{\mathcal{R}}[i] \\
\boldsymbol{\sigma} &= \sqrt{\frac{1}{N} \sum_{i}^{N} (\boldsymbol{E}_{\mathcal{R}}[i] - \boldsymbol{\mu})^2} \\
\boldsymbol{E}'_{\mathcal{R}}[i] &= \frac{\boldsymbol{E}_{\mathcal{R}}[i] - \boldsymbol{\mu}}{\boldsymbol{\sigma}}
\end{aligned}
\tag{8}
$$

We empirically find that the standardized phrase embedding is better than the original one in most cases.

F Full Result

We show the full results in Table 8 and Table 9.

	NPC-E (subject)				*NPC-E (object)*				*RPC*		
	Ma	*Mi*	*Pair*	*AVG*	*Ma*	*Mi*	*Pair*	*AVG*	*Mi*	*Pair*	*AVG*
Token Sim+SI	86.85	88.41	73.43	82.90	78.43	85.34	74.29	79.35	50.26	17.62	33.94
Random+HAC	87.97	89.90	78.10	85.32	81.24	88.11	85.97	85.11	48.47	23.49	35.98
GloVe+HAC	85.63	87.22	62.07	78.31	79.90	87.06	92.74	86.57	47.92	23.22	35.57
GloVe+HolE+HAC	88.74	89.59	62.06	80.13	80.85	88.20	92.95	87.33	27.05	8.80	17.93
GloVe+SI+HAC	88.72	89.82	62.72	80.42	80.85	88.21	92.95	87.34	27.02	8.79	17.91
CESI	88.72	89.58	62.02	80.11	82.93	88.37	89.17	86.82	27.31	9.43	18.37
VAEGMM+SI	85.63	87.51	69.44	80.86	78.55	85.93	84.26	82.91	47.86	20.34	34.10
VAEGMM+HolE	85.50	87.26	67.69	80.15	78.54	85.91	84.17	82.87	47.92	24.31	36.12
CUVA	85.58	87.45	69.00	80.68	78.56	85.95	84.33	82.95	47.94	24.32	36.13
Bert-base	90.84	92.11	77.84	86.93	86.84	90.53	83.36	86.91	55.85	53.09	54.47
Roberta-base	87.74	89.59	71.22	82.85	83.12	88.22	83.66	85.00	44.81	37.35	41.08
SpanBert-base	91.14	91.86	70.14	84.38	85.82	89.92	83.22	86.32	43.49	39.36	44.04
ERNIE2.0-base	91.19	92.50	76.35	86.68	83.15	88.89	92.29	88.11	53.94	55.38	54.66
Bert-base-triple	90.05	91.57	76.41	86.01	83.57	89.67	93.51	88.92	57.53	59.36	58.45
Roberta-base-triple	90.40	91.05	74.66	85.37	81.07	87.96	92.62	87.22	52.75	48.86	50.81
SpanBert-base-triple	91.95	91.87	73.36	85.73	85.26	89.19	83.23	85.89	48.09	44.26	46.18
ERNIE2.0-base-triple	91.14	92.34	78.16	87.21	85.71	90.12	84.96	86.93	56.58	57.98	57.28
Bert-large	90.47	91.93	78.87	87.09	83.81	89.78	93.56	89.05	53.81	46.81	50.31
Roberta-large	88.34	90.14	72.03	83.50	84.48	89.01	83.93	85.81	43.01	38.02	40.51
SpanBert-large	85.82	89.92	83.22	86.32	87.14	90.81	82.07	86.67	45.5	46.18.	45.84
ERNIE2.0-large	91.06	92.26	75.30	86.21	83.09	89.56	93.92	88.86	49.48	48.32	48.90
Bert-large-triple	90.02	92.35	93.64	92.00	87.47	90.99	84.49	87.65	56.27	60.04	58.16
Roberta-large-triple	87.93	89.35	72.93	83.40	84.24	88.57	83.56	85.46	47.37	41.41	44.39
SpanBert-large-triple	90.57	94.26	94.74	93.19	85.87	89.80	82.34	86.00	49.24	45.68	47.46
ERNIE2.0-large-triple	93.32	94.41	85.90	91.21	86.19	90.65	91.21	89.35	54.37	53.32	53.85

Table 8: Full results of NPC-E and RPC.

G Encoding Strategy Comparison

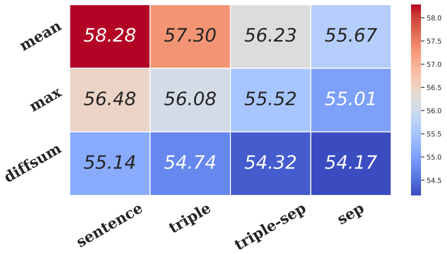


Figure 9: Overall average performance of different encoding strategies.

We compare the performance of encoding strategies averaged on all subtask metrics and PLM models in Figure 9, and the task-specific and model-specific comparison of encoding strategies are shown in Figure 10. *sentence* is the best input form in general, probably because it is easier for a PLM to encode a valid sentence and the source sentence contains more context. *sep* is the worst on the RPC because it separately encodes the RPs and NPs. However, it is comparable to *triple-sep* and *triple* on NPC-E because NPC-E requires less context. *mean* is the best strategy for phrase representation, which is consistent with the results obtained by (Toshniwal et al., 2020)[6], and *diffsum* is a bad choice for phrase canonicalization.

H Layerwise PLM Performance

We show the layerwise performance for all PLMs (base) on all subtasks in Figure 11, and find different layers perform differently on three subtasks. We empirically find that lower layers [1,2,3] perform well for NPC-E, upper layers [10,11,12] perform best for RPC and NPC-O (subj), while middle layers [3,4,5,6,7] perform relatively better on NPC-O (obj). As context-specificity increases in upper layers (Ethayarajh, 2019), these results make sense as NPC-E requires less context while RPC and NPC-O need more context.

I Triple Pretraining

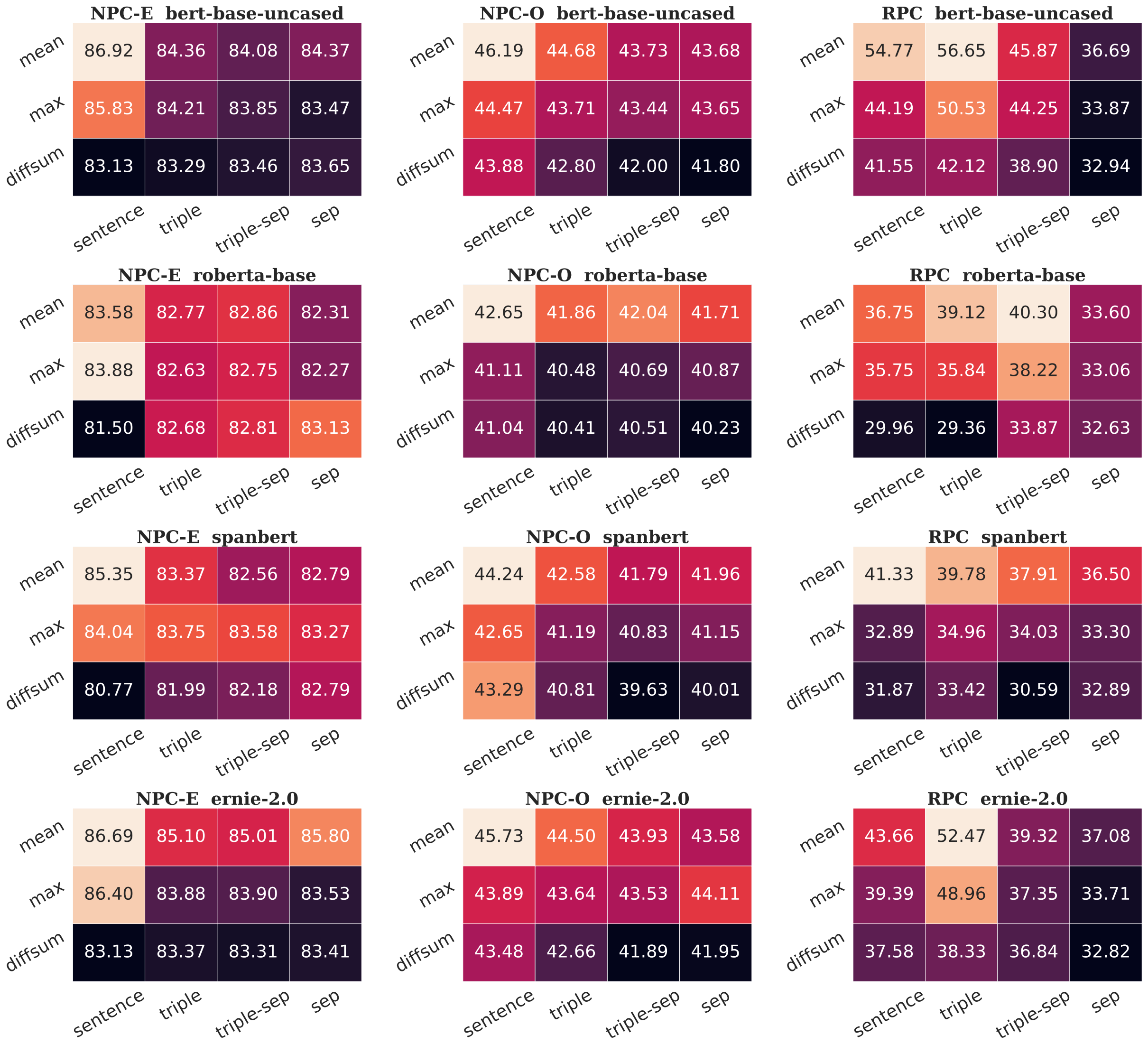
We show the average performance difference after triple-level or causal subword-level pretraining in Figure 12 for different PLMs and subtasks.

---

[6] Toshniwal et al. (2020) shows that mean pooling is best for named entity labeling, which is a semantic task for NPs.

	NPC-O (subject)							***NPC-O (object)***						
	Ma	***Mi***	***Pair***	***Avg***	$J_{g \to p}$	$J_{p \to g}$	***Avg***	***Ma***	***Mi***	***Pair***	***Avg***	$J_{g \to p}$	$J_{p \to g}$	***Avg***
Token Sim+SI	63.28	35.86	0.29	33.14	-	-	-	66.24	51.61	3.91	40.59	-	-	-
Random+HAC	69.04	42.20	0.69	37.31	62.54	15.05	38.79	74.72	54.25	5.72	44.90	66.78	22.02	44.40
GloVe+HAC	70.37	42.80	0.84	38.00	63.00	15.07	39.04	74.08	60.25	7.73	47.35	68.73	22.17	45.45
GloVe+HolE+HAC	71.41	46.06	1.43	39.63	63.95	15.78	39.87	76.68	60.38	7.68	48.25	71.06	23.30	47.18
GloVe+SI+HAC	71.68	46.32	1.62	39.87	63.88	15.78	39.83	76.63	60.43	7.72	48.26	70.96	23.26	47.11
CESI	71.66	46.51	1.62	39.93	63.98	15.90	39.89	76.57	60.45	7.74	48.25	71.14	23.24	47.19
VAEGMM+SI	69.32	42.08	0.83	37.41	-	-	-	72.52	60.10	8.07	46.90	-	-	-
VAEGMM+HolE	69.09	41.81	0.75	37.22	-	-	-	72.48	60.09	8.07	46.88	-	-	-
CUVA	68.76	41.76	0.76	37.09	-	-	-	72.48	60.12	8.08	46.89	-	-	-
Bert-base	78.77	47.20	2.93	42.97	73.27	15.04	44.16	68.10	63.71	20.32	50.71	76.41	17.56	46.99
Roberta-base	74.18	42.67	0.78	39.21	68.06	14.42	41.24	78.22	54.17	5.81	46.07	71.24	17.04	44.14
SpanBert-base	79.00	44.58	1.25	41.61	71.54	14.78	43.16	80.58	55.29	6.00	47.29	68.52	22.18	45.35
ERNIE2.0-base	78.64	46.85	2.31	42.60	72.93	15.17	44.05	68.15	66.89	21.76	52.27	71.81	22.29	47.05
Bert-base-triple	80.71	47.58	2.84	43.71	75.02	15.36	45.19	67.87	65.96	21.51	51.78	77.56	17.42	47.49
Roberta-base-triple	78.49	46.94	1.44	42.29	73.36	15.31	44.33	82.31	60.14	8.47	50.31	76.09	17.53	46.81
SpanBert-base-triple	80.49	45.47	1.63	42.53	73.47	14.98	44.23	80.17	56.15	6.49	47.60	68.84	22.29	45.56
ERNIE2.0-base-triple	79.10	47.61	2.91	43.21	73.51	15.15	44.33	69.07	66.02	16.90	50.66	72.22	22.33	47.27
Bert-large	77.93	46.33	2.76	42.34	73.09	15.22	44.16	71.92	63.47	16.75	50.71	76.92	17.58	47.25
Roberta-large	75.16	43.63	0.86	39.88	69.96	14.73	42.35	78.54	59.72	6.80	48.35	69.07	22.09	45.58
SpanBert-large	77.40	43.99	1.48	40.96	71.08	14.72	42.90	81.05	56.38	6.33	47.92	69.00	22.36	45.68
ERNIE2.0-large	76.86	48.16	3.11	42.71	72.90	15.11	44.01	71.49	64.90	19.56	51.98	72.06	22.39	47.22
Bert-large-triple	80.19	47.92	2.11	43.41	75.46	15.58	45.52	66.59	64.34	21.86	50.93	77.63	17.96	47.79
Roberta-large-triple	77.56	46.91	1.99	42.15	71.82	14.99	43.40	76.83	59.18	7.64	47.88	74.44	17.42	45.93
SpanBert-large-triple	82.09	44.88	0.62	42.53	70.47	14.71	42.59	81.97	56.08	5.81	47.95	73.79	16.85	45.32
ERNIE2.0-large-triple	79.81	48.04	3.04	43.63	75.28	15.52	45.40	71.23	64.80	20.93	52.32	76.67	17.71	47.19

Table 9: Full results of NPC-O.



NPC-E bert-base-uncased

	sentence	triple	triple-sep	sep
mean	86.92	84.36	84.08	84.37
max	85.83	84.21	83.85	83.47
diffsum	83.13	83.29	83.46	83.65

NPC-O bert-base-uncased

	sentence	triple	triple-sep	sep
mean	46.19	44.68	43.73	43.68
max	44.47	43.71	43.44	43.65
diffsum	43.88	42.80	42.00	41.80

RPC bert-base-uncased

	sentence	triple	triple-sep	sep
mean	54.77	56.65	45.87	36.69
max	44.19	50.53	44.25	33.87
diffsum	41.55	42.12	38.90	32.94

NPC-E roberta-base

	sentence	triple	triple-sep	sep
mean	83.58	82.77	82.86	82.31
max	83.88	82.63	82.75	82.27
diffsum	81.50	82.68	82.81	83.13

NPC-O roberta-base

	sentence	triple	triple-sep	sep
mean	42.65	41.86	42.04	41.71
max	41.11	40.48	40.69	40.87
diffsum	41.04	40.41	40.51	40.23

RPC roberta-base

	sentence	triple	triple-sep	sep
mean	36.75	39.12	40.30	33.60
max	35.75	35.84	38.22	33.06
diffsum	29.96	29.36	33.87	32.63

NPC-E spanbert

	sentence	triple	triple-sep	sep
mean	85.35	83.37	82.56	82.79
max	84.04	83.75	83.58	83.27
diffsum	80.77	81.99	82.18	82.79

NPC-O spanbert

	sentence	triple	triple-sep	sep
mean	44.24	42.58	41.79	41.96
max	42.65	41.19	40.83	41.15
diffsum	43.29	40.81	39.63	40.01

RPC spanbert

	sentence	triple	triple-sep	sep
mean	41.33	39.78	37.91	36.50
max	32.89	34.96	34.03	33.30
diffsum	31.87	33.42	30.59	32.89

NPC-E ernie-2.0

	sentence	triple	triple-sep	sep
mean	86.69	85.10	85.01	85.80
max	86.40	83.88	83.90	83.53
diffsum	83.13	83.37	83.31	83.41

NPC-O ernie-2.0

	sentence	triple	triple-sep	sep
mean	45.73	44.50	43.93	43.58
max	43.89	43.64	43.53	44.11
diffsum	43.48	42.66	41.89	41.95

RPC ernie-2.0

	sentence	triple	triple-sep	sep
mean	43.66	52.47	39.32	37.08
max	39.39	48.96	37.35	33.71
diffsum	37.58	38.33	36.84	32.82

Figure 10: Comparison on different input forms and span representations for tasks and PLMs.

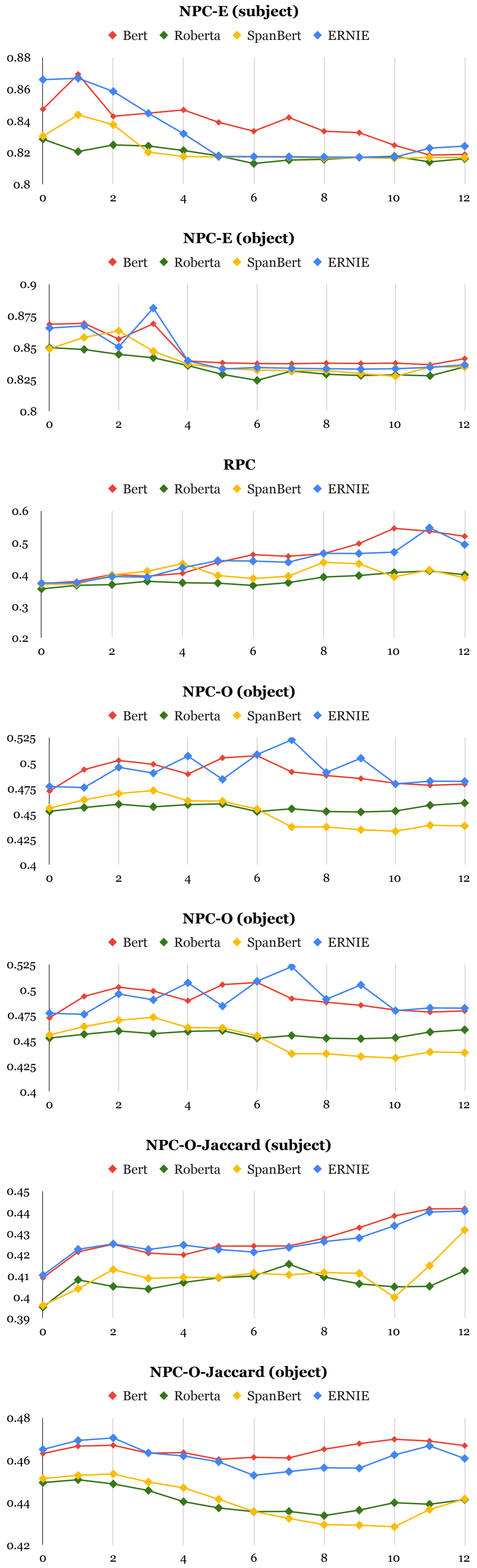
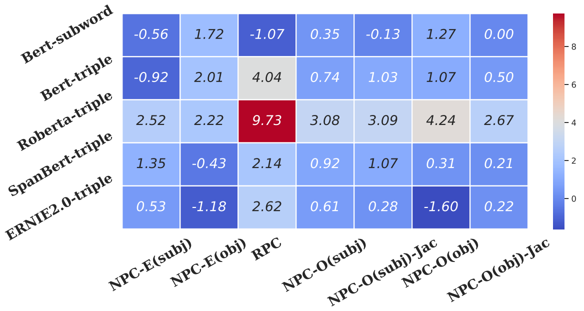


Figure 11: Layerwise performances



	NPC-E(subj)	NPC-E(obj)	RPC	NPC-O(subj)	NPC-O(subj)-Jac	NPC-O(obj)	NPC-O(obj)-Jac
Bert-subword	-0.56	1.72	-1.07	0.35	-0.13	1.27	0.00
Bert-triple	-0.92	2.01	4.04	0.74	1.03	1.07	0.50
Roberta-triple	2.52	2.22	9.73	3.08	3.09	4.24	2.67
SpanBert-triple	1.35	-0.43	2.14	0.92	1.07	0.31	0.21
ERNIE2.0-triple	0.53	-1.18	2.62	0.61	0.28	-1.60	0.22

Figure 12: Average performance difference after triple-level or causal subword-level pretraining for different PLMs and subtasks.

UScore: An Effective Approach to Fully Unsupervised Evaluation Metrics for Machine Translation

Jonas Belouadi
jonas.belouadi@uni-bielefeld.de

Steffen Eger
steffen.eger@uni-bielefeld.de

Natural Language Learning Group (NLLG)
Faculty of Technology, Bielefeld University
nl2g.github.io

Abstract

The vast majority of evaluation metrics for machine translation are supervised, i.e., (i) are trained on human scores, (ii) assume the existence of reference translations, or (iii) leverage parallel data. This hinders their applicability to cases where such supervision signals are not available. In this work, we develop *fully unsupervised* evaluation metrics. To do so, we leverage similarities and synergies between evaluation metric induction, parallel corpus mining, and MT systems. In particular, we use an unsupervised evaluation metric to mine pseudo-parallel data, which we use to remap deficient underlying vector spaces (iteratively) and to induce an unsupervised MT system, which then provides pseudo-references as an additional component in the metric. Finally, we also induce unsupervised multilingual sentence embeddings from pseudo-parallel data. We show that our fully unsupervised metrics are effective, i.e., they beat supervised competitors on four out of five evaluation datasets. We make our code publicly available.[1]

1 Introduction

Evaluation metrics are essential for judging progress in natural language generation (NLG) tasks such as machine translation (MT) and summarization, as they identify the state-of-the-art in a key NLP technology. Despite their wide dissemination, classical lexical overlap evaluation metrics like BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) have difficulties judging the quality of modern NLG systems (Mathur et al., 2020a; Marie et al., 2021), necessitating novel metrics that correlate better with humans. Lately, this has been a very active research area (Zhang et al., 2020; Zhao et al., 2019, 2022; Colombo et al., 2021; Yuan et al., 2021).[2]

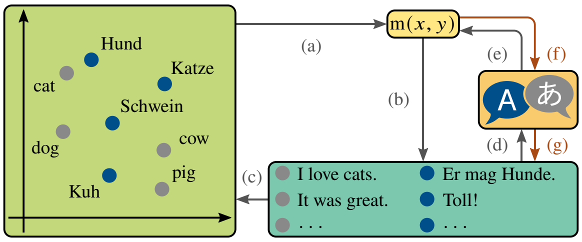


Figure 1: Relationship between metrics m, vector spaces, parallel data, and MT systems: Metrics build on (potentially deficient) multilingual vector spaces (a), and can be used to mine (pseudo-)parallel sentences (b), which in turn can be used to improve deficient vector spaces (c). (Pseudo-)parallel data can also be used to train MT systems (d), which can generate pseudo-references (e). Conversely, metrics can also be optimization criteria for MT systems, which in turn can generate additional pseudo-parallel data through translation (f & g; not explored in this work).

Recently, more and more *supervised* metrics are being proposed. E.g., BLEURT (Sellam et al., 2020) trains on human annotated datasets ranging from 5k-150k pairs, the COMET (Rei et al., 2020) models regress on 12k-370k data points and UniTE (Wan et al., 2022), before fine-tuning on the same data as COMET, pre-trains on 5m-10m parallel sentences. Of course, training on larger amounts of data leads to better metrics (measured on in-domain data), but also increases the risk of learning biases from the data (Poliak et al., 2018)—and limits the applicability to domains and language pairs where supervision is available. Here, we go the opposite route and try to minimize the amount of supervision as much as possible.

We classify existing metrics making use of different types of supervision as follows (cf. Table 1). **TYPE-1** metrics are *trained* on human assessments such as Direct-Assessment (DA) or Post-Editing (PE) scores, and compare system outputs to either human references (reference-based; Sellam et al., 2020; Rei et al., 2020) or directly to source texts (reference-free; Ranasinghe et al., 2021). **TYPE-2** metrics, by comparison, do not use human assessments for training but still require

---

[1] `github.com/potamides/unsupervised-metrics`

[2] Of course, the search for high quality metrics dates back at least to the invention of BLEU and its predecessors.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 358–374
May 2-6, 2023 ©2023 Association for Computational Linguistics

	Training	**References**	**Parallel Data**
TYPE-1	✓	(✓)	(✓)
TYPE-2	✗	✓	(✓)
TYPE-3	✗	✗	✓
Unsupervised	✗	✗	✗

Table 1: Different types of supervision used by TYPE-1/2/3 metrics compared to unsupervised metrics. Checkmarks surrounded by parentheses denote optional supervision signals.

human references, i.e., are untrained and reference-based (Yuan et al., 2021; Zhao et al., 2019; Zhang et al., 2020). Finally, **TYPE-3** metrics are untrained (unlike TYPE-1) and reference-free (unlike TYPE-2), i.e., do not use supervision as in TYPE-1 or 2. However, to work well, they still rely on parallel data (Zhao et al., 2020; Song et al., 2021), which is considered a form of supervision, e.g., in the MT community (Artetxe et al., 2018; Lample et al., 2018).

In contrast, we aim for *fully unsupervised evaluation metrics* (for MT) that do not use any form of supervision (cf. Table 1). In addition, subject to the constraint that no supervision is allowed, our metrics should be of *maximally high quality*, i.e., correlation with human assessments. We have two use cases in mind: (a) Such *sample efficiency*[3] is a prerequisite for the wide applicability of the metrics. This is especially important when we want to overcome the current English-centricity (Anastasopoulos and Neubig, 2020) of MT systems and evaluation metrics and also cover low-resource *languages* like Nepali or Sinhala (Fomicheva et al., 2021) and low-resource *pairs* like Yoruba-German.[4] (b) Our fully unsupervised evaluation metrics should be considered *strong lower bounds* for any future work that uses (mild) forms of supervision for metric induction, i.e., we want to push the lower bounds for newly developed TYPE-κ metrics.

To achieve our goals, we employ *self-learning* (He et al., 2020; Wei et al., 2021) and in particular, we leverage the following dualities to make our metrics maximally effective, cf. Figure 1: (1) Evaluation metrics and NLG systems are closely related; e.g., a metric can be an optimization criterion for an NLG system (Böhm et al., 2019), and a system can conversely generate *pseudo references* (a.o.) from which to improve a metric. (2) Evaluation metrics and parallel corpus mining (Artetxe and Schwenk, 2019) are closely related; e.g., a metric can be used to mine parallel data, which in turn can be used to improve the metric (Zhao et al., 2020), e.g., by remapping deficient embedding spaces.

Our contributions are: **(i)** We show that effective unsupervised evaluation metrics can be obtained by exploiting relationships with parallel corpus mining approaches and MT system induction; **(ii)** to do so, we explore ways to (a) make parallel corpus mining *efficient* (e.g., overcome cubic runtime complexity) and (b) induce unsupervised multilingual sentence embeddings from pseudo-parallel data; **(iii)** we show that pseudo-parallel data can rectify deficient vector spaces such as mBERT; **(iv)** we show that our metrics beat three state-of-the-art supervised metrics on four of five datasets we evaluate on.

2 Background

We take inspiration from three recent supervised (reference-free; TYPE-3) metrics: XMoverScore (Zhao et al., 2020), DistilScore (Reimers and Gurevych, 2020), and SentSim (Song et al., 2021). Below, we review key aspects of them, and show where supervision plays a role.

2.1 XMoverScore

Central to XMoverScore is the use of Word Mover's Distance (WMD) as a similarity between two sentences (Zhao et al., 2020). WMD and further enhancements are discussed below.

WMD WMD is a distance function that compares sentences at the token level (Kusner et al., 2015), by leveraging word embeddings which in XMoverScore's case come from mBERT (Devlin et al., 2019). From a source sentence x and an MT hypothesis y, WMD constructs a *distance matrix* $\mathbf{C} \in \mathbb{R}^{|x|,|y|}$, where $\mathbf{C}_{ij}$ is the distance between two word embeddings, $\mathbf{C}_{ij} = \| \mathrm{E}(x_i) - \mathrm{E}(y_j) \|$; x_i, y_j index respective words in x, y. WMD uses this distance matrix to compute the similarity of the two sentences. This can be defined as the linear programming problem

$$\mathrm{WMD}(x,y) = \min_{\mathbf{F}} \sum_{i=1}^{|x|} \sum_{j=1}^{|y|} \mathbf{F}_{ij}\mathbf{C}_{ij}, \qquad (1)$$

---

[3] We use the term sample efficiency in a generalized sense to denote the amount of supervision required.

[4] Neither Yoruba (a language spoken in Nigeria) nor German are classical low-resource languages. For German, this is clear and Yoruba is even included in mBERT, i.e., belongs to the languages with 100+ largest Wikipedias. Nonetheless, from own experience, we find it inherently difficult to obtain high-quality annotations for the language pair, as a result of few competent parallel speakers as well as technical difficulties (e.g., lack of adequate compute infrastructure in Nigeria).

where $\mathbf{F} \in \mathbb{R}^{|x|,|y|}$ is an *alignment matrix* with $\mathbf{F}_{ij}$ denoting how much of word x_i travels to word y_j. Additional constraints prevent it from becoming a zero matrix.

Vector space remapping Zhao et al. (2020), akin to similar earlier and subsequent work (Cao et al., 2020; Schuster et al., 2019), argue that the monolingual subspaces of mBERT are not well aligned. As a remedy, they investigate linear projection methods which post-hoc improve cross-lingual alignments. We refer to this approach as vector space *remapping*. XMoverScore explores two different remapping approaches, CLP and UMD. They both leverage parallel data on sentence-level from which they extract word-level alignments using fast-align, which are then used for remapping. We give more details in the Appendix A.

Language Model XMoverScore linearly combines WMD with the perplexity of a GPT-2 language model (Radford et al., 2019). Allegedly this penalizes ungrammatical translations. This updates XMoverScores scoring function to

$$\mathfrak{m}(x, y) = w_{\text{xlng}}\,\text{WMD}(x, y) + w_{\text{lm}}\,\text{LM}(y). \quad (2)$$

Here, w_{xlng}, w_{lm} are weights for the cross-lingual WMD and LM components of XMoverScore.

2.2 DistilScore

Reimers and Gurevych (2020) show that the cosine between multilingual sentence embeddings captures semantic similarity and can be used to assess cross-lingual semantic textual similarity. Their approach to inducing embedding models is based on multilingual knowledge distillation. We refer to this metric as DistilScore. Their approach requires supervision at multiple levels. First, parallel sentences are needed to induce multilingual models, and second, NLI and STS corpora are required to induce teacher embeddings in the source language.

2.3 SentSim

A key difference between XMoverScore and DistilScore is that one approach is based on word- and the other on sentence embeddings. Song et al. (2021) and Kaster et al. (2021) show that combining approaches based on word-level and sentence-level representations can substantially improve metrics. The metric of Song et al. (2021), which is called SentSim, combines supervised DistilScore with one of two word embedding-based metrics. The first one is quite similar to XMoverScore, as it is also based on WMD. The other one is a multilingual variant of BERTScore (Zhang et al., 2020).

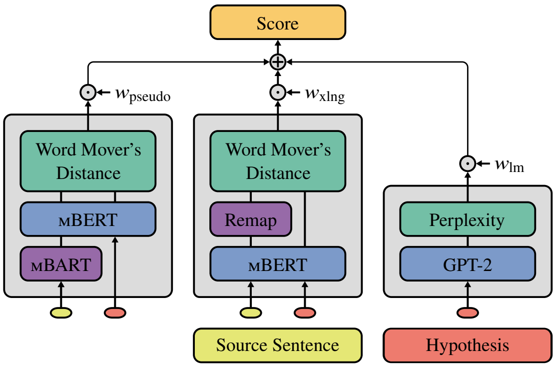


Figure 2: UScore$_{\text{WRD}}$ with pseudo references (left), unsupervised remapping (middle) and a LM (right).

3 Methods

In this section, we introduce our fully unsupervised metric UScore. UScore builds upon the existing metrics XMoverScore, DistilScore, and SentSim, but eliminates all supervision signals and instead leverages the dualities shown in Figure 1. In particular, we mine pseudo-parallel data from unsupervised metrics, which we use (iteratively) to (a) rectify deficient vector spaces (for XMoverScore) and to (b) train unsupervised MT systems which can generate pseudo-references (as pseudo-references are in the same language as the hypothesis, this eliminates problems of cross-lingual deficiency). Furthermore, we use pseudo-parallel data to (c) induce an unsupervised sentence embedding model analogous to DistilScore, which we can then (d) integrate with the unsupervised word based model analogous to SentSim. We now give details.

3.1 UScore$_{\text{WRD}}$

XMoverScore uses sentence-parallel data to extract word pairs for vector space remapping. (i) We replace this parallel data with pseudo-parallel data.[5] (ii) In addition, we use pseudo-references to address the issue of deficient vector spaces. We now give details on (i) and (ii) below.

Efficient WMD Pseudo-Parallel Data Mining Metrics such as XMoverScore could in principle be used for pseudo-parallel corpus mining since

---

[5] To extract the word pairs from the sentence-parallel data, XMoverScore uses fast-align (Dyer et al., 2013), but since this depends directly on how well sentences are aligned, we first replace it with unsupervised awesome-align (Dou and Neubig, 2021) which only relies on pre-trained language models.

they can compare arbitrary sentences. However, when WMD-based metrics are scaled to corpus mining, algorithmic efficiency problems arise: (a) the computational complexity of WMD scales cubically with sentence length (Kusner et al., 2015); (b) to compare m source to m target sentences, m^2 WMD invocations are necessary, which quickly becomes intractable. Thus, we explore ways to improve the performance of WMD to mine efficiently. In particular, Kusner et al. (2015) define a linear approximation of WMD called *word centroid distance* (WCD) and a mining algorithm that first sorts all target samples according to their WCD to a given query and computes exact WMD for the k nearest neighbors. We use this algorithm for efficient WMD-based pseudo-parallel data mining.

In our work, we apply this approach *iteratively* (cf. Figure 1): we start out with an initial WMD metric (based on mBERT), obtain *sentence-level* pseudo-parallel data with it via the efficient approximation algorithm described, and obtain a dictionary of word pairs from unsupervised AWESOME-ALIGN from the sentence pseudo-parallel data. We use the pseudo-parallel word pairs with UMD and CLP to remap mBERT. From this, we obtain a better WMD metric; then we iterate.

Pseudo References Apart from remapping, pseudo-parallel data could be used to overcome problems of deficient vector spaces in other ways. Specifically, we want to mine enough pseudo-parallel data to train an unsupervised MT system to translate source sentences into the target language to create *pseudo references* (Albrecht and Hwa, 2007; Gao et al., 2020; Fomicheva et al., 2020b). This would allow for a comparison with the hypothesis in the target language, similar to reference-based metrics, circumventing alignment problems in multilingual embeddings. This approach updates UScore$_{\text{WRD}}$ to

$$\mathfrak{m}(x, y, y') = w_{\text{xlng}}\, \text{WMD}^{(n)}(x, y) + w_{\text{lm}}\, \text{LM}(y) + w_{\text{pseudo}}\, \text{WMD}(y, y'), \tag{3}$$

where n denotes the iterations of remapping, and w_{pseudo} is a new weight to control the influence of the pseudo reference y'. All components of UScore$_{\text{WRD}}$ are illustrated in Figure 2.

3.2 UScore$_{\text{SNT}}$

Besides a word-based metric, we use pseudo-parallel data to induce an unsupervised *sentence level* metric, UScore$_{\text{SNT}}$ = $\cos(x, y)$, based on the cosine similarity between sentence embeddings. One could, similarly to DistilScore, perform knowledge distillation but since our initial experiments showed that this doesn't work well with pseudo-parallel data, we chose another approach.

Contrastive Learning We explore contrastive learning for unsupervised *multilingual* sentence embedding induction, which has recently been successfully used to train unsupervised *monolingual* sentence embeddings (Gao et al., 2021). In our context, the basic idea is to pull semantically close sentences together and to push distant sentences apart in the embedding space. Let x_i and y_i be the embeddings of two sentences that are semantically related and N an arbitrary batch size. The training objective for this pair can be formulated as

$$L_i = -\log \frac{\exp\left(\frac{\cos(x_i, y_i)}{\tau}\right)}{\sum_{j=1, j\neq i}^{N} \exp\left(\frac{\cos(x_i, y_j)}{\tau}\right)}, \tag{4}$$

where τ is a temperature hyperparameter that can be used to either amplify or dampen the assessed distances. For each sentence x_i, all remaining sentences $y_{j\neq i}$ in the current batch should be pushed apart in the embedding space. For positive sentences that should be pulled together, we again use pseudo-parallel sentence pairs. Since noisy data is beneficial for contrastive learning (Gao et al., 2021), we expect this paradigm to work well with pseudo-parallel data. We use pooled XLM-R embeddings as sentence representations, and, as with unsupervised remapping, we experiment with multiple iterations of successive mining and sentence embedding induction operations.

Ratio Margin Pseudo-Parallel Data Mining As UScore$_{\text{SNT}}$ is based on sentence embeddings, we cannot use the WMD-based mining algorithm to obtain pseudo-parallel sentences since it requires access to word-level representations. An alternative would be to just use cosine similarity for mining, but that approach is susceptible to noise in the data (Artetxe and Schwenk, 2019). Instead, we follow Artetxe and Schwenk (2019) and use a ratio margin function defined as

$$\text{margin}(x, y) = \frac{\cos(x, y)}{\sum\limits_{z \in N_x} \frac{\cos(x, z)}{2k} + \sum\limits_{z \in N_y} \frac{\cos(y, z)}{2k}}, \tag{5}$$

where N_x and N_y are the k nearest neighbors of sentence embeddings x and y in the respective language. Informally, this ratio margin function divides the cosine similarities of the nearest neighbor by the average similarities of the neighborhood.

3.3 UScore$_{\text{WRD} \oplus \text{SNT}}$

Inspired by SentSim, which combines word and sentence embeddings, we similarly ensemble UScore$_{\text{WRD}}$ and UScore$_{\text{SNT}}$. We refer to this final metric as UScore $=$ UScore$_{\text{WRD} \oplus \text{SNT}}$ with two new weights w_{wrd} and $w_{\text{snt}} = 1 - w_{\text{wrd}}$:

$$\begin{aligned} \text{UScore}(x, y) = w_{\text{wrd}}\, &\text{UScore}_{\text{WRD}}(x, y) \\ &+ w_{\text{snt}}\, \text{UScore}_{\text{SNT}}(x, y). \end{aligned} \tag{6}$$

4 Experiments

In this section, we evaluate all UScore variants at the segment level[6] and compare them to TYPE-1/2/3 upper bounds. We detail additional hyperparameters in Appendix D.

4.1 Datasets

We use various datasets to assess the performance of our metrics on **MT evaluation**, i.e., computing the correlation with human assessments using Pearson's r correlation, and **parallel sentence matching**, a standard evaluation measure in the corpus mining field where a set of shuffled parallel sentences is searched to recover correct translation pairs (Guo et al., 2018; Kvapilíková et al., 2020). For this we report Precision at N (P@N).

MT evaluation In **WMT-16** and **WMT-17,** each language pair consists of tuples of source sentences, hypotheses and references. Each tuple was annotated with a direct assessment (DA) score, which quantifies the adequacy of the hypothesis given the reference translation. Following Zhao et al. (2020) and Song et al. (2021), we use these DA scores to assess the adequacy of the hypothesis given the source. **MLQE-PE** has been used in the WMT 2020 Shared Task on Quality Estimation (Specia et al., 2020), and only provides source sentences and hypotheses for its language pairs, with no references. Each source sentence and hypothesis pair was annotated with cross-lingual direct assessment (CLDA) scores. In terms of annotation, **Eval4NLP** is very similar to MLQE-PE but focuses on non-English-centric language directions, especially de-zh and ru-de. **WMT-MQM** uses fine-grained error annotations from the Multidimensional Quality Metrics (MQM) framework (Freitag et al., 2021a) for adequacy assessments. Like MLQE-PE and Eval4NLP, WMT-MQM also assigns scores based on source sentences and hypotheses. Additional statistics can be found in the appendix in Table 8. Using ISO 639-1 codes, our datasets cover the language pairs de-zh, ru-de, en-ru, en-zh, cs-en, de-en, en-de, et-en, fi-en, lv-en, ne-en, ro-en, ru-en, si-en, tr-en, zh-en.

Parallel sentence matching To evaluate on parallel sentence matching, we use the **News Commentary**[7] dataset. It consists of parallel sentences crawled from economic and political data.

4.2 Fine-grained analysis on de-en

To gain an understanding of the properties of the iterative techniques and the influence of individual parameters / components, we conduct a fine-grained analysis on the de-en language direction of WMT-16 (for MT evaluation) and News Commentary v15 (for parallel sentence matching). We list examples of pseudo-parallel data used during training in the appendix in Table 3. The mined sentences are often semantically similar, but contain factuality errors (e.g., have wrong places or numbers in hypotheses).

Vector space remapping We explore if remapping works with pseudo-parallel data. We use News Crawl for mining. We randomly extract 40k monolingual sentences per language, and select the top 5% sentence pairs with the highest metric scores for remapping. This gives us the same number of sentences (2k pairs) as were used for remapping XMoverScore.

The results for UMD and CLP-based remapping on de-en can be seen in Figure 3 (top). The figure contains two graphs, one for correlation with human judgments and one for precision on parallel sentence matching. Each graph illustrates model performance before remapping (Iteration 0) and after remapping one to five times. After remapping once, both UMD and CLP improve substantially in Pearson's r correlation. The improvement of CLP, however, is noticeably larger. For subsequent iterations, UMD seems to continue to improve slightly, but the correlations of CLP seem to drop. This can be explained by the results for precision where the P@1 of CLP drops each iteration, meaning the remapping capabilities of the metrics decrease. UMD does not exhibit this problem. Thus, UMD

---

[6] We do not evaluate at the system level since metrics there often perform very similarly, making it difficult to determine the best metric (Mathur et al., 2020b; Freitag et al., 2021b).

[7] `data.statmt.org/news-commentary`

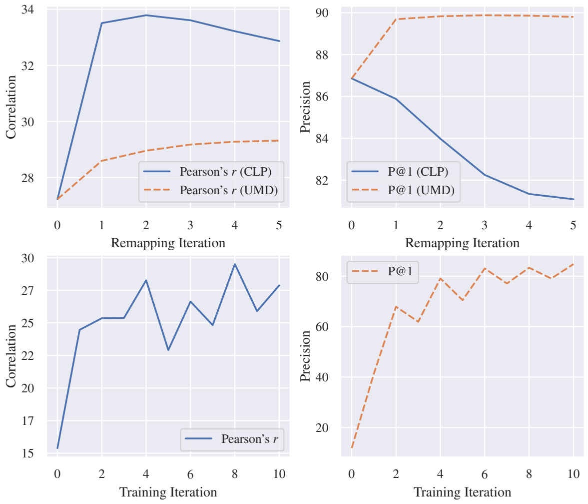


Figure 3: Results for unsupervised vector space remapping (top) and contrastive learning with USCORE$_{\text{SNT}}$ (bottom) for de-en. Pearson's r is computed on WMT-16 (MT evaluation) and P@1 on News Commentary v15 (parallel sentence matching).

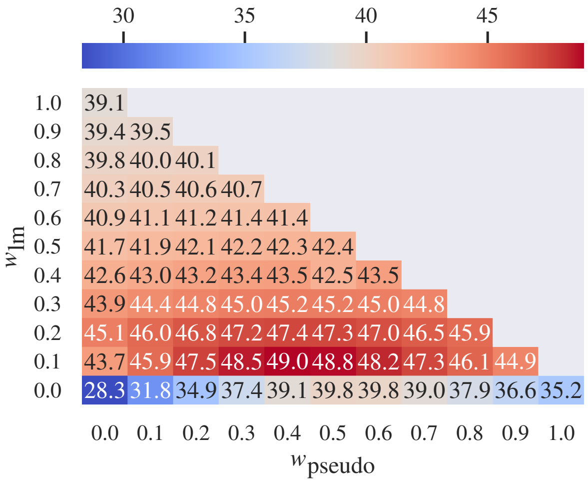


Figure 4: Influence of a language model and an MT system on USCORE$_{\text{WRD}}$, segment-level Pearson's r for different values of w_{pseudo} (weight for pseudo references) and w_{lm} (weight for the language model) on WMT-16 de-en data. Note that the point $w_{\text{pseudo}} = w_{\text{lm}} = 0$ uses only WMD(x, y); see Equation 3. Here, $n = 1$.

could be a more robust choice for metrics that should perform reasonably well on both tasks.

Pseudo References & Language Model Next, we add a language model to the metric and investigate pseudo-parallel corpus mining to train an MT system for pseudo references. Tran et al. (2020) show that fine-tuning MBART using pseudo-parallel data leads to very promising results, so we use MBART for our own experiments as well. Since fine-tuning for MT is a very resource-intensive undertaking requiring many parallel sentence pairs (Barrault et al., 2020), especially compared to our vector space remapping experiments, we need considerably more training data. On average, Tran et al. (2020) use around 200k pseudo-parallel sentence pairs for training. To obtain the same amount with our extraction rate of 5%, we now use a pool of 4m sentences per language for mining. Our results on the de-en data of WMT-16 are reported in Figure 4, which is similar to an ablation study. On the x-axis, we vary the weight $w_{\text{pseudo}} \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$ for USCORE$_{\text{WRD}}$ with pseudo references, and on the y-axis, we explore different weights $w_{\text{lm}} \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$ for the language model. We set $w_{\text{xlng}} = 1 - w_{\text{pseudo}} - w_{\text{lm}}$. The best correlation uses $w_{\text{pseudo}} = 0.4$, $w_{\text{lm}} = 0.1$, and $w_{\text{xlng}} = 0.5$. The improvement when pseudo references and a language model are included is substantial (over only using WMD)—e.g., we improve from 28% correlation with humans ($w_{\text{pseudo}} = w_{\text{lm}} = 0$) to 49% with the best weight combination, an improvement of 75%.

Contrastive Learning For USCORE$_{\text{SNT}}$, we also use 4m monolingual sentences per language for mining but only retain the top 100k sentence pairs, as for the contrastive training objective we additionally have to filter out duplicate sentences. The results of USCORE$_{\text{SNT}}$ are shown in Figure 3 (bottom). The P@1 scores seem to steadily improve every two training iterations. Beginning with the sixth iteration, the precision seems to converge.

4.3 Other Languages: Results & Analysis

We now test our metrics on other languages and datasets. For USCORE$_{\text{SNT}}$, we train its sentence embedding model for six iterations. For USCORE$_{\text{WRD}}$, we remap MBERT once with UMD and make use of a language model and pseudo references obtained from an MT system. Based on Section 4.2, we set $w_{\text{pseudo}} = 0.4$, $w_{\text{lm}} = 0.1$, and $w_{\text{xlng}} = 0.5$. Additionally, based on analogous, unreported experiments, we set $w_{\text{wrd}} = 0.6$ and $w_{\text{snt}} = 0.4$ for ensembling. Since determining the weights for USCORE$_{\text{WRD}}$ and USCORE$_{\text{WRD} \oplus \text{SNT}}$ this way constitutes a form of supervision, we also evaluate weights chosen independently from our conducted experiments. Namely, we also evaluate USCORE^{+} with $w_{\text{wrd}} = w_{\text{snt}} = 0.5$. For w_{lm}, we follow XMOVERSCORE and set it to 0.1 (setting w_{lm} lower makes sense because the LM only addresses the hypothesis without considering the source); accordingly, we set $w_{\text{xlng}} = w_{\text{pseudo}} = 0.45$. Since $w_{\text{lm}} = 0.1$ coincides with our findings in Section 4.2, we also evaluate UScore^{++} where each component uses entirely uniform weights, i.e., $w_{\text{wrd}} = w_{\text{snt}} = \frac{1}{2}$ and $w_{\text{lm}} = w_{\text{xlng}} = w_{\text{pseudo}} = \frac{1}{3}$.

Correlations with human judgments averaged over language pairs are shown in Table 2 (individual results are in the appendix). We also present the results of the popular TYPE-2 metric BLEU, where possible, and the recent TYPE-1 metrics MonoTransQuest (Ranasinghe et al., 2020b,a) and COMET-QE (Rei et al., 2021). Finally, as more direct competitors, we compare to the TYPE-3 metrics XMoverScore, SentSim, and DistilScore. We compute all reported scores ourselves.

Overall, the tuned weights of UScore perform marginally better than UScore^{+} on most datasets, but UScore^{+} is usually a very close second. UScore^{++} performs worse, however, and only competitively on two of the five datasets. This indicates that the language model should be set to a lower value, a choice that makes intuitively sense.

Expectedly, DistilScore, which uses parallel data, is always better than UScore$_{\text{SNT}}$, which uses pseudo-parallel data. In contrast, UScore$_{\text{WRD}}$ is generally on par with XMoverScore, even though XMoverScore uses real parallel data—the difference is that UScore$_{\text{WRD}}$ also leverages pseudo-references which XMoverScore does not. Indeed, from Figure 4, we observe that the pseudo-references can make an improvement of up to 1–11 points in correlation (comparing 'column' labeled $w_{\text{pseudo}} = 0$ to the columns $w_{\text{pseudo}} > 0$).

Our metrics beat reference-based TYPE-2 BLEU across the board. TYPE-1 metrics, which are fine-tuned on human scores, are generally the best. Intriguingly, the only two language pairs where our metrics are on par with them are the non-English de-zh and ru-de from Eval4NLP. These languages are outside the training scope of the current TYPE-1 metrics and thus test their generalization abilities. For example, on ru-de our best metric outperforms MonoTransQuest by 5 points correlation and COMET-QE by 9 points (Table 7 in the appendix).

UScore and UScore^{+} also outperform the TYPE-3 upper bounds on four of five datasets. On WMT-16, WMT-17 and Eval4NLP, they have the best overall results. On WMT-MQM, UScore$_{\text{WRD}}$ alone is best. The drop in performance for the combined metric is caused by UScore$_{\text{SNT}}$, which on its own performs very badly. As supervised DistilScore exhibits the same issues, this could be a general problem for sentence embeddings based metrics on this dataset. We identify further reasons in Appendix E.

For MLQE-PE, the SentSim metrics perform best on average among TYPE-3 and our metrics—although our reproduced scores for this dataset differ noticeably from the authors' results, due to issues in their original code (Chen et al., 2022). Among our self-learned metrics, the combined variant performs best on average again, but still is 3–5 points below SentSim and DistilScore, even though it outperforms both XMoverScore variants by over 6 points. Interestingly, UScore$_{\text{SNT}}$ works better than UScore$_{\text{WRD}}$, unlike for the other datasets. Similarly, DistilScore clearly outperforms XMoverScore. This could be because MLQE-PE contains Sinhala, a language mBERT was not trained on. Another explanation is the data collection scheme for ru-en, which uses different sources of parallel sentences, mainly colloquial data and Russian proverbs, which use rather unconventional grammar (Fomicheva et al., 2022). This apparently confuses the language model and MT system which have been trained on data from other domains. When we exclude si-en and ru-en from MLQE-PE, UScore$_{\text{WRD} \oplus \text{SNT}}$ performs best, with an average Pearson's r of 44.22 for tuned weights and 44.45 for default weights vs. 43.82 for SentSim (BERTScore). In Appendix C, we show that incorporating real parallel data (in addition to pseudo-parallel data) at an order of magnitude lower than SentSim allows us to outperform SentSim on MLQE-PE also.

5 Discussion

Throughout, we have presented a mix of results and analysis, which we now summarize and discuss. In Figure 4, we conducted an ablation study on the individual components of UScore$_{\text{WRD}}$ (on de-en). This showed that all three components (pseudo-references, language model, WMD) matter; by itself, the LM is more important than the pseudo-references which are more important than WMD. However, in combination, the LM is least important. We also showed that pseudo-parallel data can successfully rectify deficient multilingual vector spaces, similar to real parallel data, see Figure 3. We note, however, that pseudo-parallel data may introduce an important bias in our data sampling: namely, it may mine factually incorrect parallel

	Metric	**WMT-16**	**WMT-17**	**MLQE-PE**	**Eval4NLP**	**WMT-MQM**
TYPE-1/2 Supervised	MONOTRANSQUEST	64.68	66.47	66.28	42.21	42.23
	COMET-QE	65.76	68.77	49.97	40.00	45.89
	BLEU	47.82	47.01	–	–	19.76
TYPE-3 Supervised	XMOVERSCORE (UMD)	49.96	51.02	33.99	29.60	20.07
	XMOVERSCORE (CLP)	53.10	$\underline{55.09}$	31.98	$\underline{36.93}$	20.59
	SENTSIM (BERTSCORE)	51.86	$\underline{55.57}$	**45.36**	26.60	13.71
	SENTSIM (WMD)	50.66	54.29	$\underline{44.72}$	24.44	12.24
	DISTILSCORE	43.79	51.22	$\underline{43.31}$	28.90	2.20
Unsupervised	USCORE$_{WRD}$	$\underline{53.87}$	53.52	31.13	$\underline{36.96}$	**23.28**
	USCORE$_{SNT}$	36.06	42.68	37.51	20.39	-0.47
	USCORE$_{WRD \oplus SNT}$	**55.78**	**57.55**	40.60	**39.87**	19.35
	USCORE$^{+}_{WRD}$	$\underline{53.66}$	52.97	30.94	$\underline{36.29}$	$\underline{23.15}$
	USCORE$^{+}_{WRD \oplus SNT}$	$\underline{55.28}$	$\underline{57.29}$	41.41	$\underline{38.22}$	17.71
	USCORE$^{++}_{WRD}$	51.88	51.73	23.76	25.42	19.00
	USCORE$^{++}_{WRD \oplus SNT}$	$\underline{54.99}$	$\underline{56.49}$	34.15	32.05	17.08

Table 2: Segment-level Pearson's *r* correlations with human judgments, averaged over language directions. The best results are highlighted in bold, while results that are not significantly worse (as determined by a two-sample t-test with 10% significance level) are underlined.

sentences, see Table 3 in the appendix, which may amplify issues of adversarial robustness; see our discussion in Section 8.

We remark that, depending on the annotation scheme, better correlations with human judgments do not necessarily entail better metrics (Freitag et al., 2021a). The datasets in this work were annotated using either DA, CLDA, or MQM scores, with MQM explicitly addressing this problem. Since our metrics are consistent regardless of the annotation scheme, they are unlikely to overfit a particular one.

We finally note that combining word and sentence-level models is meaningful, because they offer complementary views (Song et al., 2021). Kaster et al. (2021) also show that they capture orthogonal linguistic factors to varying degrees. Such complementarity may also stem from different underlying vector spaces, i.e., MBERT vs. XLM-R that we use in sentence- and word-level metrics.

6 Related Work

All metrics in this work presented so far treated the MT model generating the hypotheses as a black-box. There also exists a recent line of work of so-called glass-box metrics, which actively incorporate the MT model under test into the scoring process (Fomicheva et al., 2020a,b). In particular, Fomicheva et al. (2020b) explore whether the MT model under test can be used to generate additional hypotheses (Dreyer and Marcu, 2012). A crucial difference to our metrics is the required availability of the original MT model, which we are agnostic about. The MT models used in Fomicheva et al. (2020b) are all trained on parallel data, which makes their approach a supervised metric in our sense.

Other recent metrics that leverage the relationship between metrics and (MT) systems are PRISM (Thompson and Post, 2020) and BARTSCORE (Yuan et al., 2021). We do not classify them as unsupervised, however, as PRISM is trained from scratch on parallel data and BARTSCORE uses a BART model fine-tuned on labeled summarization or paraphrasing datasets.

There are also multilingual sentence embedding models which are highly relevant in our context. Kvapilíková et al. (2020), for example, fine-tune XLM-R on synthetic data translated with an unsupervised MT system. Similar to our contrastive learning approach, the resulting embedding model is completely unsupervised. Important differences are that our sentence embedding model can be improved iteratively and does not rely on an MT system. We leave a comparison to future work.

Finally, the idea of fully unsupervised text generation systems has originated in the MT community (Artetxe et al., 2018; Lample et al., 2018; Artetxe et al., 2019). Given the similarity of MT systems and evaluation metrics, designing fully unsupervised evaluation metrics is an apparent next step, which we take in this work.

7 Conclusion

In this work, we aimed for sample efficient evaluation metrics that do not use any form of supervision.

In addition, our novel metrics should be maximally effective, i.e., of high quality. To achieve this, we leveraged pseudo-parallel data obtained from fully unsupervised evaluation metrics in three ways: we (i) remapped deficient vector spaces using the pseudo-parallel data, (ii) trained an unsupervised MT system from it (yielding pseudo references), and (iii) induced unsupervised multilingual sentence embeddings. To enable our approach, we also explored *efficient* pseudo-parallel corpus mining algorithms based on our metrics as an orthogonal contribution. Finally, we showed that our approach is effective and can outperform three supervised upper bounds (making use of parallel data) on 4 out of 5 datasets we included in our comparison.

In future work, we want to aim for algorithmic efficiency, include *pseudo source texts* as additional components (using the MT system in backward translation), and address the missing dualities discussed in Figure 1 (i.e., use of metrics as optimization criteria and MT systems to generate additional pseudo-parallel data). Further, our approach has substantial room for improvement given that we selected hyperparameters completely unsupervised or based on one high-resource language pair (de-en). Thus, it will be particularly intriguing to explore *weakly-supervised approaches* which leverage minimal forms of supervision.

8 Limitations

Limitations of our metrics include (1) *algorithmic inefficiency*, (2) *resource inefficiency*, (3) the brittleness of unsupervised MT systems in certain situations, and (4) issues of adversarial robustness.

(1) Some of the components of USCORE$_{\text{WRD}}$ (mainly the MT system) have high computational costs. For example, XMOVERSCORE and SENTSIM (BERTSCORE) take less than 30 seconds to score 1000 hypotheses on an Nvidia V100 GPU. USCORE$_{\text{WRD}}$, on the other hand, takes over 2.5 minutes. This algorithmic inefficiency trades off with our sample efficiency, by which we did not use any supervision signals. In future work, we aim to experiment with efficient MT architectures to reduce computational costs (Kamal Eddine et al., 2022; Grünwald et al., 2022).

(2) Similarly to XMOVERSCORE, MONOTRANSQUEST or SENTSIM, our metrics use high-quality encoders such as MBERT, which are not only memory and inference inefficient but also leverage large monolingual resources. Future work should thus not only investigate using smaller MBERT models but also models that leverage smaller amounts of monolingual resources. Wang et al. (2020), for example, propose a competitive LSTM-based approach that completely forgoes monolingual resources and instead uses small parallel corpora (i.e., a few hundred parallel sentences as a weak supervision signal). Similarly, we give a recipe for improving MBERT for unseen languages using limited amounts of parallel data in Appendix C.

(3) Using unsupervised MT approaches, as we do via pseudo references, may be less effective for truly low-resource languages (Marchisio et al., 2020). However, this remains a very active research field with a constant influx of more powerful solutions (Ranathunga et al., 2022; Sun et al., 2021).

(4) As indicated in Sections 4.2 and 5, our mined pseudo-parallel data tends to contain factual inconsistencies such as “Uruguay was seventh” vs. (a translation of) “Russia was second”. As a consequence, our induced metrics may be less robust than existing metrics (Chen and Eger, 2022; Rony et al., 2022). An approach to address this inconsistency would be to retain only high probability aligned words in parallel sentences (recall that we infer word-level parallel data from sentence-level parallel data).

9 Acknowledgments

We thank all reviewers for their valuable feedback, hard work, and time. The last author was supported by DFG grant EG 375/5–1.

References

Joshua Albrecht and Rebecca Hwa. 2007. Regression for sentence-level MT evaluation with pseudo references. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 296–303, Prague, Czech Republic. Association for Computational Linguistics.

Antonios Anastasopoulos and Graham Neubig. 2020. Should all cross-lingual embeddings speak English? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8658–8679, Online. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2019. An effective approach to unsupervised machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 194–203, Florence, Italy. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *International Conference on Learning Representations*.

Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *ACL*.

Loïc Barrault, Magdalena Biesialska, Ondrej Bojar, Marta Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi kiu Lo, Nikola Ljubesic, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (wmt20). In *WMT@EMNLP*.

Florian Böhm, Yang Gao, Christian M. Meyer, Ori Shapira, Ido Dagan, and Iryna Gurevych. 2019. Better rewards yield better summaries: Learning to summarise without references. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3110–3120, Hong Kong, China. Association for Computational Linguistics.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Multilingual alignment of contextual word representations. In *International Conference on Learning Representations*.

Yanran Chen, Jonas Belouadi, and Steffen Eger. 2022. Reproducibility issues for BERT-based evaluation metrics. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2965–2989, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yanran Chen and Steffen Eger. 2022. Menli: Robust evaluation metrics from natural language inference. *ArXiv*, abs/2208.07316.

Pierre Colombo, Guillaume Staerman, Chloé Clavel, and Pablo Piantanida. 2021. Automatic text evaluation through the lens of Wasserstein barycenters. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10450–10466, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sunipa Dev and Jeff M. Phillips. 2019. Attenuating bias in word vectors. In *AISTATS*.

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

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *EACL*.

Markus Dreyer and D. Marcu. 2012. Hyter: Meaning-equivalent semantics for translation evaluation. In *NAACL*.

Sharad Duwal, Amir Manandhar, Saurav Maskey, and Subash Hada. 2019. Efforts in the development of an augmented english-nepali parallel corpus. Technical report, Kathmandu University.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *HLT-NAACL*.

M. Fomicheva, Shuo Sun, Lisa Yankovskaya, F. Blain, Francisco Guzmán, M. Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020a. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Marina Fomicheva, Piyawat Lertvittayakumjorn, Wei Zhao, Steffen Eger, and Yang Gao. 2021. The Eval4NLP shared task on explainable quality estimation: Overview and results. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 165–178, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marina Fomicheva, Lucia Specia, and Francisco Guzmán. 2020b. Multi-hypothesis machine translation evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1218–1232, Online. Association for Computational Linguistics.

Marina Fomicheva, Shuo Sun, Erick Fonseca, Chrysoula Zerva, Frédéric Blain, Vishrav Chaudhary, Francisco Guzmán, Nina Lopatina, Lucia Specia, and André F. T. Martins. 2022. MLQE-PE: A multilingual quality estimation and post-editing dataset. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4963–4974, Marseille, France. European Language Resources Association.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021a. Experts, Errors, and Context: A Large-Scale Study of Human Evaluation for Machine Translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, George Foster, Alon Lavie, and Ondřej Bojar. 2021b. Results of the WMT21 metrics shared task: Evaluating metrics with expert-based human evaluations on TED and news domain. In *Proceedings of the Sixth Conference on Machine Translation*, pages 733–774, Online. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *EMNLP*.

Yang Gao, Wei Zhao, and Steffen Eger. 2020. SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

Jens Grünwald, Chris Leiter, and Steffen Eger. 2022. Can we do that simpler? simple, efficient, high-quality evaluation metrics for nlg. *ArXiv*, abs/2209.09593.

Mandy Guo, Qinlan Shen, Yinfei Yang, Heming Ge, Daniel Matthew Cer, G. Ábrego, K. Stevens, Noah Constant, Yun-Hsuan Sung, B. Strope, and R. Kurzweil. 2018. Effective parallel corpus mining using bilingual sentence embeddings. In *WMT*.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *Proceedings of ICLR*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Moussa Kamal Eddine, Guokan Shang, Antoine Tixier, and Michalis Vazirgiannis. 2022. FrugalScore: Learning cheaper, lighter and faster evaluation metrics for automatic text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1305–1318, Dublin, Ireland. Association for Computational Linguistics.

Marvin Kaster, Wei Zhao, and Steffen Eger. 2021. Global explainability of BERT-based evaluation metrics by disentangling along linguistic factors. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8912–8925, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Phillip Keung, Julian Salazar, Yichao Lu, and Noah A. Smith. 2021. Unsupervised bitext mining and translation via self-trained contextual embeddings. *Transactions of the Association for Computational Linguistics*, 8:828–841.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *ICML*.

Ivana Kvapilíková, Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Ondrej Bojar. 2020. Unsupervised multilingual sentence embeddings for parallel corpus mining. In *ACL*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Kelly Marchisio, Kevin Duh, and Philipp Koehn. 2020. When does unsupervised machine translation work? In *Proceedings of the Fifth Conference on Machine Translation*, pages 571–583, Online. Association for Computational Linguistics.

Benjamin Marie, Atsushi Fujita, and Raphael Rubino. 2021. Scientific credibility of machine translation research: A meta-evaluation of 769 papers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7297–7306, Online. Association for Computational Linguistics.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020a. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020b. Results of the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 688–725, Online. Association for Computational Linguistics.

Tomás Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages

180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *OpenAI Blog*.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2020a. Transquest at wmt2020: Sentence-level direct assessment. In *Proceedings of the Fifth Conference on Machine Translation*.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2020b. Transquest: Translation quality estimation with cross-lingual transformers. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2021. An exploratory analysis of multilingual word-level quality estimation with cross-lingual transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 434–440, Online. Association for Computational Linguistics.

Surangika Ranathunga, En-Shiun Annie Lee, Marjana Prifti Skenduli, Ravi Shekhar, Mehreen Alam, and Rishemjit Kaur. 2022. Neural machine translation for low-resource languages: A survey. *ACM Comput. Surv.*

Ricardo Rei, Ana C Farinha, Chrysoula Zerva, Daan van Stigt, Craig Stewart, Pedro Ramos, Taisiya Glushkova, André F. T. Martins, and Alon Lavie. 2021. Are references really needed? unbabel-IST 2021 submission for the metrics shared task. In *Proceedings of the Sixth Conference on Machine Translation*, pages 1030–1040, Online. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *EMNLP*.

Md Rashad Al Hasan Rony, Liubov Kovriguina, Debanjan Chaudhuri, Ricardo Usbeck, and Jens Lehmann. 2022. RoMe: A robust metric for evaluating natural language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5645–5657, Dublin, Ireland. Association for Computational Linguistics.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. In *EACL*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Yurun Song, Junchen Zhao, and Lucia Specia. 2021. Sentsim: Crosslingual semantic evaluation of machine translation. In *NAACL*.

Lucia Specia, F. Blain, M. Fomicheva, E. Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the wmt 2020 shared task on quality estimation. In *WMT@EMNLP*.

Haipeng Sun, Rui Wang, Masao Utiyama, Benjamin Marie, Kehai Chen, Eiichiro Sumita, and Tiejun Zhao. 2021. Unsupervised neural machine translation for similar and distant language pairs: An empirical study. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 20(1):1–17.

Brian Thompson and Matt Post. 2020. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *EMNLP*.

Chau Tran, Yuqing Tang, Xian Li, and Jiatao Gu. 2020. Cross-lingual retrieval for iterative self-supervised training. In *Advances in Neural Information Processing Systems*, volume 33, pages 2207–2219. Curran Associates, Inc.

Yu Wan, Dayiheng Liu, Baosong Yang, Haibo Zhang, Boxing Chen, Derek Wong, and Lidia Chao. 2022. UniTE: Unified translation evaluation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8117–8127, Dublin, Ireland. Association for Computational Linguistics.

Zihan Wang, Karthikeyan K, Stephen Mayhew, and Dan Roth. 2020. Extending multilingual BERT to low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2649–2656, Online. Association for Computational Linguistics.

Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. 2021. Theoretical analysis of self-training with deep networks on unlabeled data. In *International Conference on Learning Representations*.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *HLT-NAACL*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. BARTScore: Evaluating generated text as text generation. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Wei Zhao, Goran Glavaš, Maxime Peyrard, Yang Gao, Robert West, and Steffen Eger. 2020. On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In *ACL*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Wei Zhao, Michael Strube, and Steffen Eger. 2022. DiscoScore: Evaluating text generation with BERT and discourse coherence.

A Vector Space Remapping

Zhao et al. (2020) explore two different remapping approaches for XMoverScore, which are defined as follows:

Procrustes alignment Mikolov et al. (2013) propose to compute a linear transformation matrix $\mathbf{W}$ which can be used to map a vector x of a source word into the target language subspace by computing $\mathbf{W}x$. The transformation can be computed by solving the problem

$$\min_{\mathbf{W}} ||\mathbf{WX} - \mathbf{Y}||_2. \tag{7}$$

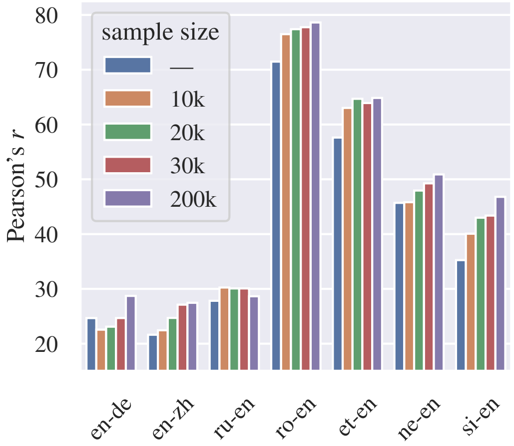
Here $\mathbf{X}, \mathbf{Y}$ are matrices with embeddings of source and target words, respectively, where the tuples $(x_i, y_i) \in \mathbf{X} \times \mathbf{Y}$ come from parallel word pairs. XMoverScore constrains $\mathbf{W}$ to be an orthogonal matrix such that $\mathbf{W}^\top\mathbf{W} = \mathbf{I}$, since this can lead to further improvements (Xing et al., 2015). Zhao et al. (2020) call this remapping Linear Cross-Lingual Projection remapping (CLP).

De-biasing The second remapping method of XMoverScore is rooted in the removal of biases from word embeddings. Dev and Phillips (2019) explore a bias attenuation technique called Universal Language Mismatch-Direction (UMD). It involves a bias vector v_B, which is supposed to capture the bias direction. For each word embedding e, an updated word embedding e' is computing by subtracting their projections onto v_B, as in

$$e' = e - (e \cdot v_B)v_B, \tag{8}$$

where $\cdot$ is the dot product. To obtain the bias vector v_B, Dev and Phillips (2019) use a set $\mathcal{E}$ of word pairs that should be de-biased (e.,g. *man* and *woman*). The subtractions of the embeddings of the words in each pair are then stacked to form a matrix $\mathbf{Q}$, and the bias vector v_B is its top-left singular vector. Zhao et al. (2020) use the same approach for XMoverScore, but $\mathcal{E}$ instead consists of parallel word pairs.

Zhao et al. (2020) show that these remapping methods lead to substantial improvements of their XMoverScore metric (on average, up to 10 points in correlation). The required parallel word pairs were extracted from sentences of the EuroParl corpus (Koehn, 2005) using the fast-align (Dyer et al., 2013) word alignment tool. The best results were obtained when remapping on 2k parallel sentences.



	—	10k	20k	30k	200k
Avg	40.60	42.95	44.42	45.17	**46.57**

Figure 5: Pearson's r correlations on MLQE-PE for UScore$_{\text{WRD}\oplus\text{SNT}}$ when fine-tuning on limited amounts of parallel data. We explore sample sizes of 10k, 20k, 30k, and 200k.

B Filtering

Since large corpora tend to include low-quality data points, we follow Artetxe and Schwenk (2019) and Keung et al. (2021) and apply three simple filtering techniques. We first remove all sentences from each monolingual corpus for which the fastText language identification tool (Joulin et al., 2017) predicts a different language. We then filter all sentences which are shorter than 3 tokens or longer than 30 tokens. As the last step, we discard sentence pairs sharing substantial lexical overlap, which prevents degenerate alignments of, e.g., proper names. We remove all sentence pairs for which the Levenshtein distance detects an overlap of over 50%.

C Fine-Tuning on Parallel Data

To examine whether and by how much we can further improve our metrics using forms of supervision, we experiment with a fine-tuning step on *parallel sentences* and treat self-learning on pseudo-parallel data as pre-training (He et al., 2020). We use the parallel data to fine-tune the contrastive sentence embeddings of UScore$_{\text{SNT}}$ and the MT system of UScore$_{\text{WRD}}$, which is responsible for generating pseudo references. Further, we also compute new remapping matrices for UScore$_{\text{WRD}}$. Since CLP is superior to UMD when parallel data is used (see Section 4.2), we compute these remapping matrices using CLP instead of UMD. To assess how different amounts of parallel sentences affect performance, we fine-tune our metrics on 10k, 20k, 30k, and 200k parallel sentences. We use WikiMatrix (Schwenk et al., 2021) and the Nepali Translation Parallel Corpus (Duwal et al., 2019) to obtain parallel sentences.

Pearson's r correlations with human judgments for individual and averaged language pairs are shown in Figure 5; we focus on MLPQE-PE, where our metrics performed worst. Overall, introducing parallel data into the training process consistently improves performance for the majority of language directions; more parallel data leads to better results. The relatively biggest improvements are achieved for the si-en language direction, which is in accordance with our discussion above. When fine-tuning with 30k parallel sentences, the performance of our metrics is roughly on par with the SentSim variants (see Table 2). With 200k parallel sentences, our metrics clearly outperform SentSim, which uses millions of parallel sentences and NLI data as supervision signals.

D Hyperparameters

For efficient WMD pseudo-parallel mining, we set $k = 20$ for remapping, and $k = 1$ for training mBART. For ratio-margin-based pseudo-parallel mining, we use $k = 5$. With regard to training UScore$_{\text{SNT}}$, we follow Gao et al. (2021), and iteratively train XLM-R for one training epoch with a learning rate of 5e-5, a batch size of 256, and a temperature coefficient of $\tau = 0.05$ utilizing the AdamW optimizer (Loshchilov and Hutter, 2019). Fine-tuning of mBART was performed for three epochs with a batch size of four and using the same learning rate of 5e-5 as well as AdamW optimizer.

We decided to continue using mBERT in UScore$_{\text{WRD}}$ for two reasons. Firstly, we want UScore$_{\text{WRD}}$ to remain directly comparable to XMoverScore which is based on mBERT. Secondly, in our own experience, vanilla mBERT is very robust in terms of layer choice, especially compared to vanilla XLM-R. Intuitive choices like the first or last layer work very well for a lot of problems. This is an important property for unsupervised metrics, since we can't easily justify a (supervised) hyperparameter search in an unsupervised setting to determine the best layer or even a linear combination of those.

E Supplementary Data and Results

Table 3 shows examples of pseudo-parallel data obtained with UScore$_{\text{WRD}}$ and UScore$_{\text{SNT}}$. Tables 4, 5, 6, and 7 show segment-level Pearson's r correlations with human judgments on WMT-16, WMT-17, MLQE-PE, as well as WMT-MQM and Eval4NLP, respectively. Table 8 provides additional statistics for each dataset.

A surprising finding is the poor performance of UScore$_{\text{SNT}}$ and DistilScore on the German-English language pair in Table 7. It is well known that high-resource language directions, such as English-German, can be affected by a lack of low-quality translations (Fomicheva et al., 2022), and with only high-quality translations available, there is little variation in the scores, which makes a meaningful assessment of correlation difficult (Specia et al., 2020). Further, since both UScore$_{\text{SNT}}$ and DistilScore are based on sentence embeddings of XLM-R, and sentence embedding-based metrics are known to be a bit worse on average than their word embedding-based counterparts, we believe that both aspects combined could be the root cause for this.

Case	**Source**	**Target**
Top-WRD	Uruguay belegt mit vier Punkten nur Platz Sieben.	Russia was second with four gold and 13 medals.
Top-WRD	Soweit lautet zumindest die Theorie.	That, at least, is the theory.
Rnd-WRD	Die USA stellen etwa 17.000 der insgesamt 47.000 ausländischen Soldaten in Afghanistan.	Currently, there are about 170,000 U.S. troops in Iraq and 26,000 in Afghanistan.
Rnd-WRD	“Das ist eine schwierige Situation”, sagte Kaczynski.	“It seemed like a ridiculous situation,” Vanderjagt said.
Top-SNT	Die Wahlen für ein neues Parlament sollen dann Anfang Januar stattfinden.	Parliamentary elections are to be held by January.
Top-SNT	Anzeichen für die Blauzungenkrankheit sind Fieber, Entzündungen und Blutungen an der Zunge der Tiere.	Contact with the creatures can cause itching, rashes, conjunctivitis and, in some cases, breathing problems.
Rnd-SNT	Riesen-Wirbel an der Universität Zagreb: An der wirtschaftlichen Fakultät und am Institut für Verkehrsstudien durchsuchen Polizisten die Büros von Dozenten.	Those attending the Soil Forensics International Conference work in the fields of science, policing, forensic services as well as private industries.
Rnd-SNT	Frankfurt soll WM-Finale der Frauen ausrichten	The women’s tournament gets underway on Sunday.

Table 3: Pseudo-parallel data obtained via UScore$_{\text{WRD}}$ and UScore$_{\text{SNT}}$; top and random sentence pairs. The mined sentences are semantically similar, but contain factuality errors (e.g., have wrong places or numbers in hypotheses).

	Metric	de-en	en-ru	ru-en	ro-en	cs-en	fi-en	tr-en
TYPE-1/2 Supervised	MonoTransQuest	61.65	66.69	63.32	62.36	67.67	68.33	62.71
	COMET-QE	65.73	71.91	69.71	66.40	67.98	64.83	53.73
	BLEU	45.39	55.08	46.33	47.09	53.80	39.92	47.15
TYPE-3 Supervised	XMoverScore (UMD)	43.46	62.16	**60.52**	47.88	58.83	43.52	33.34
	XMoverScore (CLP)	45.29	63.58	56.12	54.24	58.89	51.40	42.14
	SentSim (BERTScore)	48.49	50.37	57.89	55.06	59.08	46.66	**45.47**
	SentSim (WMD)	47.78	48.49	56.19	54.48	56.87	46.00	44.84
	DistilScore	40.62	41.21	43.16	51.22	49.51	39.65	41.14
Unsupervised	UScore$_{\text{WRD}}$	48.94	60.97	59.09	56.06	57.15	53.76	41.15
	UScore$_{\text{SNT}}$	30.48	39.73	35.31	44.18	40.80	31.30	30.62
	UScore$_{\text{WRD} \oplus \text{SNT}}$	**50.37**	62.92	60.49	60.07	**59.25**	52.93	44.41
	UScore$^{+}_{\text{WRD}}$	48.97	60.57	58.35	56.23	56.44	53.78	41.30
	UScore$^{+}_{\text{WRD} \oplus \text{SNT}}$	50.29	62.37	59.43	**60.71**	58.47	51.48	44.19
	UScore$^{++}_{\text{WRD}}$	44.50	61.45	53.50	56.03	52.95	54.08	40.65
	UScore$^{++}_{\text{WRD} \oplus \text{SNT}}$	47.27	**64.14**	56.40	60.36	56.29	**55.62**	44.85

Table 4: Segment-level Pearson’s *r* correlations with human judgments on the WMT-16 dataset.

	Metric	cs-en	de-en	fi-en	lv-en	ru-en	tr-en	zh-en
TYPE-1/2 Supervised	MonoTransQuest	60.93	63.54	65.33	72.01	59.56	75.91	68.03
	COMET-QE	68.99	69.34	72.83	64.75	69.06	68.43	68.01
	BLEU	41.22	41.29	56.48	39.28	45.99	53.06	51.75
TYPE-3 Supervised	XMoverScore (UMD)	41.72	51.19	56.61	56.11	47.24	46.52	57.73
	XMoverScore (CLP)	47.76	50.04	62.22	63.95	48.79	52.97	59.88
	SentSim (BERTScore)	49.90	52.26	57.85	57.42	**55.10**	56.84	59.59
	SentSim (WMD)	47.62	50.42	56.59	56.91	53.42	56.24	58.86
	DistilScore	46.42	45.64	54.03	55.51	54.13	54.04	50.89
Unsupervised	UScore$_{\text{WRD}}$	46.70	52.71	61.91	59.22	49.10	50.06	54.95
	UScore$_{\text{SNT}}$	38.89	41.92	39.77	48.95	37.27	48.83	43.10
	UScore$_{\text{WRD} \oplus \text{SNT}}$	**50.67**	**56.50**	61.86	64.54	53.17	57.31	58.80
	UScore$^{+}_{\text{WRD}}$	44.55	52.96	60.96	59.02	48.82	49.70	54.75
	UScore$^{+}_{\text{WRD} \oplus \text{SNT}}$	48.62	56.31	60.44	63.62	54.19	**57.54**	**60.28**
	UScore$^{++}_{\text{WRD}}$	45.41	48.00	61.33	60.64	49.65	47.06	50.00
	UScore$^{++}_{\text{WRD} \oplus \text{SNT}}$	49.27	52.20	**63.10**	**66.64**	53.70	54.93	55.62

Table 5: Segment-level Pearson’s *r* correlations with human judgments on the WMT-17 dataset.

	Metric	en-de	en-zh	ru-en	ro-en	et-en	ne-en	si-en
TYPE-1 Supervised	MonoTransQuest	41.85	45.76	76.76	88.81	73.19	75.70	61.90
	COMET-QE	36.03	30.70	49.33	64.95	63.45	57.46	47.85
TYPE-3 Supervised	XMoverScore (UMD)	16.56	16.48	28.07	65.83	53.95	38.23	18.81
	XMoverScore (CLP)	25.59	20.25	20.31	57.34	**58.46**	25.15	16.74
	SentSim (BERTScore)	6.15	22.23	47.30	**78.55**	55.09	**57.09**	**51.14**
	SentSim (WMD)	3.86	22.62	**47.46**	77.72	54.60	57.00	49.79
	DistilScore	12.96	**28.68**	45.34	76.57	51.16	46.73	41.76
Unsupervised	UScore$_{\text{WRD}}$	24.53	21.58	16.66	60.30	54.83	28.62	11.38
	UScore$_{\text{SNT}}$	13.62	13.27	39.09	66.50	41.75	46.83	41.49
	UScore$_{\text{WRD} \oplus \text{SNT}}$	24.67	21.63	27.83	71.48	57.62	45.70	35.25
	UScore$^{+}_{\text{WRD}}$	23.78	21.40	15.18	60.39	56.27	27.34	12.22
	UScore$^{+}_{\text{WRD} \oplus \text{SNT}}$	22.00	21.41	27.71	73.65	57.94	47.27	39.91
	UScore$^{++}_{\text{WRD}}$	26.17	27.16	0.75	32.92	52.24	12.79	14.30
	UScore$^{++}_{\text{WRD} \oplus \text{SNT}}$	**26.37**	27.26	9.29	57.70	56.81	31.24	30.38

Table 6: Segment-level Pearson's r correlations with human judgments on the MLQE-PE dataset.

		WMT-MQM		**Eval4NLP**	
	Metric	en-de	zh-en	de-zh	ru-de
TYPE-1/2 Supervised	MonoTransQuest	31.38	53.07	34.66	49.76
	COMET-QE	40.94	50.84	33.51	46.49
	BLEU	17.94	21.58	—	—
TYPE-3 Supervised	XMoverScore (UMD)	12.72	27.41	10.85	48.35
	XMoverScore (CLP)	12.89	28.29	21.61	52.25
	SentSim (BERTScore)	1.03	26.39	-7.71	**60.91**
	SentSim (WMD)	-0.23	24.70	-11.54	60.42
	DistilScore	-2.66	7.06	6.57	51.22
Unsupervised	UScore$_{\text{WRD}}$	18.13	**28.43**	29.29	44.63
	UScore$_{\text{SNT}}$	-5.93	4.99	0.86	39.91
	UScore$_{\text{WRD} \oplus \text{SNT}}$	13.94	25.17	24.66	55.08
	UScore$^{+}_{\text{WRD}}$	19.83	26.46	29.99	42.59
	UScore$^{+}_{\text{WRD} \oplus \text{SNT}}$	13.72	21.7	22.64	53.8
	UScore$^{++}_{\text{WRD}}$	**20.19**	17.80	**36.51**	14.32
	UScore$^{++}_{\text{WRD} \oplus \text{SNT}}$	17.18	16.97	34.00	30.09

Table 7: Segment-level Pearson's r correlations with human judgments on the WMT-MQM and Eval4NLP datasets.

Dataset	Pairs	Tokens	Type
WMT-16	560	9886	CA
WMT-17	560	11189	CA
MLQE-PE	1000	15263	CLDA
Eval4NLP	1295	25975	CLDA
WMT-MQM	17090	549835	MQM

Table 8: Statistics of our used datasets averaged over language pairs. For each dataset we report the number of sentences we evaluated on, the amount of tokens in these, and the type of human annotation. The number of tokens refers to source sentences.

Assistive Recipe Editing through Critiquing

Diego Antognini[*]
IBM Research
diego.antognini@ibm.com

Shuyang Li[†]
Meta AI
lishuyang@meta.com

Boi Faltings
EPFL
boi.faltings@epfl.ch

Julian McAuley
UCSD
jmcauley@ucsd.edu

Abstract

Home cooks often have specific requirements regarding individual ingredients in a recipe (e.g., allergies). Substituting ingredients in a recipe can necessitate complex changes to instructions (e.g., replacing chicken with tofu in a stir fry requires pressing the tofu, marinating it for less time, and par-cooking)—which has thus far hampered efforts to automatically create satisfactory versions of recipes. We address these challenges with the *RecipeCrit* model that allows users to *edit* existing recipes by proposing individual ingredients to add or remove. Crucially, we develop an unsupervised critiquing module that allows our model to iteratively re-write recipe instructions to accommodate the complex changes needed for ingredient substitutions. Experiments on the Recipe1M dataset show that our model can more effectively edit recipes compared to strong language-modeling baselines, creating recipes that satisfy user constraints and humans deem more correct, serendipitous, coherent, and relevant.

1 Introduction

Individual preferences and dietary needs shape the types of recipes that home cooks choose to follow. Cooks must often accommodate the desire for versions of recipes that do not contain a specific ingredient (substitution—e.g., for food allergies) or *do* make use of particular ingredients (addition—e.g., to use up near-expiry items). We thus aim to build a system for *recipe editing* that accommodates fine-grained ingredient preferences.

Prior research in recipe editing has focused on substituting individual ingredients in the ingredients list (Yamanishi et al., 2015) or recommending new recipes based on similar ingredients (Teng et al., 2012). Individual ingredient substitution rules (e.g., tapioca flour and xanthan gum for wheat flour) often necessitate additional changes to the cooking procedure to function properly (Li et al., 2022). Other studies employed recommendation-based approaches. However, they suffer from data sparsity: there is an extremely large set of possible recipes that differ by a single ingredient, and many specific substitutions may not appear in recipe aggregators (Petrescu et al., 2021).

Recipe editing can be seen as a combination of recipe generation and controllable natural language generation (Shin et al., 2020), and has been explored to adapt recipes for broad dietary constraints (Li et al., 2022) and cuisines (Pan et al., 2020). Pre-trained language models have been used to create recipe directions given a known title and set of ingredients (Kiddon et al., 2016; Bosselut et al., 2018), but generated recipes suffer from inconsistencies. Li et al. (2022) instead build a paired recipe dataset, but face challenges scaling due to the large set of possible recipes and dietary restrictions; people often express even more specific ingredient-level preferences (e.g., dislikes of certain ingredients or allergies).

In this work, we address the above challenges and propose **RecipeCrit**, a denoising-based model trained to complete recipes and learn semantic relationships between ingredients and instructions. The novelty of this work relies on an *unsupervised* critiquing method that allows users to provide ingredient-focused feedback iteratively; the model substitutes ingredients and also re-writes the recipe text using a generative language model. While existing methods for controllable generation require paired data with specially constructed prompts (Keskar et al., 2019) or hyperparameter-sensitive training of individual models for each possible piece of feedback (Dathathri et al., 2020), our unsupervised critiquing framework enables recipe editing models to be trained with arbitrary un-paired data. This generalizes recipe editing, unlike existing methods for controllable generation that rely on paired data with specially constructed

---

[*] Work done while at EPFL and a research stay at UCSD.
[†] Work done while at UCSD.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 375–384
May 2-6, 2023

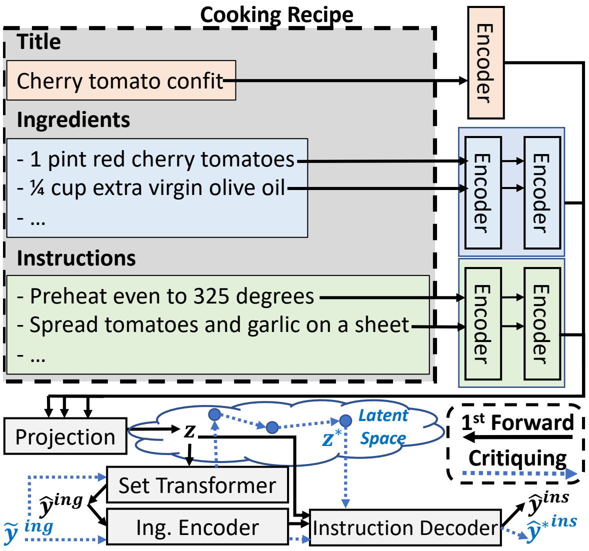


Figure 1: RecipeCrit includes a recipe encoder and an ingredient and instruction decoders using the base recipe and target ingredients to edit cooking instructions.

prompts (Keskar et al., 2019) or hyperparameter-sensitive training of individual models for each possible feedback (Dathathri et al., 2020).

Experiments on the Recipe1M (Salvador et al., 2017) dataset show that RecipeCrit edits recipes in a way that better satisfies user constraints, preserves the original recipe, and produces coherent recipes (i.e., recipe instructions are better conditioned on the ingredients list) compared to state-of-the-art pre-trained recipe generators and language models. Human evaluators judge RecipeCrit's recipes to be more serendipitous, correct, coherent, and relevant to the ingredient-specific positive and negative feedback (i.e., critiques).

2 RecipeCrit: a Hierarchical Denoising Recipe Auto-encoder

Previous methods to edit recipes focused on broad classes like dietary categories (Li et al., 2022) and cuisines (Pan et al., 2020) and require paired corpora (which do not exist for fine-grained edits). We propose **RecipeCrit**: a hierarchical denoising recipe auto-encoder that does not require paired corpora to train and accommodates positive and negative user feedback about ingredients (Figure 1).

RecipeCrit is divided into three submodels: an Encoder $E(\cdot)$, which produces the latent representation $\mathbf{z}$ from the (potentially noisy) recipe; an ingredient predictor $C(\cdot)$, which predicts the ingredients $\hat{\mathbf{y}}^{ing}$, and a decoder $D(\cdot)$, which reconstructs the cooking instructions $\hat{\mathbf{y}}^{ins}$ from $\mathbf{z}$ conditioned on the $\hat{\mathbf{y}}^{ing}$.

Recipe Encoder $E(\cdot)$ We build a powerful latent representation that captures the different elements of a recipe via the mean-pooled output of the representation of each sentence using a Transformer encoder (Vaswani et al., 2017). We provide the title $\mathbf{x}^{ttl}$, ingredients $\mathbf{X}^{ing}$, and instructions $\mathbf{X}^{inst}$ as *raw* text input. While the title $\mathbf{x}^{ttl}$ comprises a single sentence, the ingredients $\mathbf{X}^{ing}$ and instructions $\mathbf{X}^{inst}$ are provided as lists of sentences; we use raw recipe texts directly as input, removing the need of a pre-processing step. We encode the ingredients and instructions in a hierarchical manner using another Transformer to create fixed-length representations. We compute the latent representation $\mathbf{z}$ by concatenating the representations of each component and applying a projection followed by a tanh function: $\mathbf{z} = \tanh(\mathbf{W}[\mathrm{TRF}(\mathbf{x}^{ttl}) \parallel \mathrm{HTRF}(\mathbf{X}^{ing}) \parallel \mathrm{HTRF}(\mathbf{X}^{ins})])$, where $\parallel$ is the concatenation, and $\mathbf{W}$, $\mathbf{b}$ the projection parameters.

Ingredient Predictor $C(\cdot)$ We treat ingredient prediction as multi-label binary classification over an ingredient vocabulary I, with a boolean target vector $\mathbf{y}^{ing}$ representing ingredients in a ground truth recipe. We use a Set Transformer (Salvador et al., 2019) to decode ingredients, pooling ingredient logits over time-steps to compute binary cross-entropy loss against the target; we also employ an *EOS* token to predict the ingredient set cardinality:

$$\mathcal{L}_{ing}(\cdot) = -\sum_i^{|I|} \mathbf{y}_i^{ing} \log \hat{\mathbf{y}}_i^{ing} - \lambda \mathbf{y}_{eos}^{ing} \log \hat{\mathbf{y}}_{eos}^{ing},$$

where λ controls the impact of the *EOS* loss. At inference, we return the top-k ingredients, where k is the first position with a positive *EOS* prediction.

Instruction Decoder $D(\cdot)$ The last component generates cooking instructions using a Transformer decoder. We condition the decoder on $\mathbf{z}$, previously generated outputs $\hat{\mathbf{y}}^{ins}_{1:t-1}$, and ingredients $\hat{\mathbf{y}}^{ing}$. Specifically, we encode the ingredients using an embedding layer $A(\cdot)$ and concatenate their representations with the recipe representation $\mathbf{z}$. We train using teacher-forcing and cross-entropy:

$$\mathcal{L}_{ins}(\mathbf{z}, A(\hat{\mathbf{y}}^{ing})) = -\sum_t \mathbf{y}_t^{ins} \log \hat{\mathbf{y}}_t^{ins}.$$

Taking inspiration from masked language and span modeling (Devlin et al., 2019; Joshi et al., 2020), we train RecipeCrit as a de-noising recipe auto-encoder via the task of *recipe completion*: We mask random ingredients and instruction sentences in model input, and task our model to generate the full recipe. We train our model in two stages: first minimizing ingredient prediction loss $\mathcal{L}_{ing}$; then freezing the encoder and optimizing for instruction decoding loss $\mathcal{L}_{ins}$ using ground truth ingredients.

Unsupervised Critiquing We aim to refine a recipe based on the user's feedback and the predicted ingredients $\hat{\mathbf{y}}^{ing}$. We denote $\tilde{\mathbf{y}}^{ing}$ the vector of desired ingredients. Simply incorporating user feedback by explicitly including/removing ingredients before generating instructions often cannot satisfy user preferences due to weak conditioning between predicted ingredients and generated instructions. In RecipeCrit we turn to a critiquing method that modifies the recipe representation $\mathbf{z}$ before using the updated representation to jointly generate the edited ingredients and instructions. Specifically, users add a new ingredient c by setting $\hat{\mathbf{y}}_c^{ing} = 1$ or remove some using $\hat{\mathbf{y}}_c^{ing} = 0$.

Inspired by success in editing the latent space in text style transfer and recommendation (Antognini et al., 2021a; Wang et al., 2019), we first compute the gradient with respect to $\mathbf{z}$:

$$\mathbf{g}_{t-1} = \nabla_{\mathbf{z}_{t-1}} \mathcal{L}_{ing}(C(\mathbf{z}_{t-1}), \tilde{\mathbf{y}}^{ing}).$$

Then ,we use the gradient to modulate $\mathbf{z}$ such that the new predicted ingredients $\hat{\mathbf{y}}^{ing}$ are close to the desired ingredients $\tilde{\mathbf{y}}^{ing}$:

$$\mathbf{z}_t = \mathbf{z}_{t-1} - \alpha_{t-1}\mathbf{g}_{t-1}/||\mathbf{g}_{t-1}||_2.$$

Prior work stopped updating $\mathbf{z}$ when $\|\ \tilde{\mathbf{y}}^{ing} - \hat{\mathbf{y}}^{ing}\ \|_1 < \epsilon$ for some threshold ϵ. We instead propose to compute the absolute difference $|\tilde{\mathbf{y}}_c^{ing}$ - $\hat{\mathbf{y}}_c^{ing}|$. Since the optimization is nonconvex, we improve convergence by using an early stopping mechanism. Our approach is unsupervised and can update the full recipe latent representation, reflecting how adding or removing an ingredient can necessitate adjustments to other ingredients and cooking steps. Pseudo-code is available in the App.

Another advantage of our approach is the possibility to update multiple ingredients simultaneously: adding or removing an ingredient might affect other ones as well and thus, a local-based stopping criteria allows such a change.

3 Experiments

Dataset We assess our model on the Recipe1M (Salvador et al., 2017) dataset of 1M recipe texts. Each recipe contains a title, a list of ingredients, and a list of cooking instructions. We filter out recipes with more than 20 ingredients or steps, creating train, val, and test splits with 635K, 136K, and 136K recipes, respectively. The average recipe comprises 9 ingredients and 166 words. We follow Salvador et al. (2019) and build a set of $1,488$ ingredients. For critiquing, we select 20 ingredients to be critiqued among the most and the least popular ingredients across the train set. For each critique, we randomly sample 50 recipes that contain the critiqued ingredient and 50 that do not.

Baselines We compare our proposed RecipeCrit architecture against large language models trained using our denoising objective. We fine-tune BART (Lewis et al., 2020), an encoder-decoder language model trained to denoise documents, as well as RecipeGPT (Lee et al., 2020), a decoder-only language model pre-trained on Recipe1M to predict ingredients and cooking steps. To demonstrate the necessity of our denoising approach, we also compare against PPLM (Dathathri et al., 2020), a recent method for controllable generation from language models that leverages sets of desired and undesired sequences (for ingredient addition and substitution, respectively). All models use greedy decoding.

Metrics We evaluate edited recipes via metrics that reflect user preferences. First, a user wants a recipe similar to the base recipe—we measure *ingredient fidelity* via IoU (Jaccard distance) and F1 scores between the edited and base recipe ingredients list. Next, the recipe must satisfy the user's specific ingredient feedback—we report the percentage of edited recipes that properly include/exclude the target ingredient (*Success Rate*). Finally, the recipe must be *coherent*: able to be followed and internally consistent. As an ingredient constraint can be satisfied in many ways, we follow Kiddon et al. (2016) and measure coherence via precision, recall, and F1-score of ingredients mentioned in the generated steps compared to the predicted ingredients. This verifies that the recipe itself relies on the listed ingredients.

Training Details For fair comparison, we compare similar-sized models. RecipeCrit uses an encoder and decoder with 4 Transformer layers, 4 attention heads, and hidden size of 512. We randomly mask 50% of the ingredients and instructions during training, and tune them on the validation set using random search. We give more details in App. B.

			Ingr. Fidelity		*Predicted Instr.*		
	Model	**% Succ.**	**IoU**	**F1**	**Prec.**	**Rec.**	**F1**
Add	RecipeGPT	33.2	65.4	78.7	56.7	69.0	62.2
	PPLM	34.4	60.9	72.7	53.0	63.0	57.6
	BART	41.1	70.5	82.8	61.5	61.1	61.3
	RecipeCrit	**66.3**	**74.5**	**85.4**	**73.7**	**74.4**	**74.1**
Remove	RecipeGPT	91.1	37.2	52.9	38.4	54.6	45.0
	PPLM	92.3	61.3	32.6	47.2	53.5	50.2
	BART	95.4	55.7	73.3	57.6	61.6	59.5
	RecipeCrit	**95.8**	**68.8**	**80.7**	**74.0**	**74.5**	**74.2**

Table 1: Critiquing performance: success rate of adding/removing an ingredient, IoU and F1 ingredient scores, and the Precision, Recall, and F1 of ingredients in cooking instructions.

RQ1: Recipe Editing via Critiquing We evaluate whether our models can edit recipes by creating new ingredient sets and corresponding recipe instructions when faced with positive and negative feedback: an ingredient that must be added or removed (substituted) from the recipe to create a new version. For ingredient substitution, we mask the critiqued ingredient and all steps that reference it as denoising inputs; for addition, we use the full base recipe. For RecipeGPT and BART, we filter the predicted ingredients lists to exclude/include the target ingredient. For PPLM, we provide the target ingredient as a bag of words to steer generation, using RecipeGPT as the base generative model. RecipeCrit uses our iterative critiquing framework (Section 2) to accommodate user feedback.

We show results for constraint satisfaction (success rate), ingredient fidelity, and recipe coherence (predicted instructions) in Table 1. RecipeCrit outperforms baselines across all metrics for ingredient addition and removal. While our baselines take advantage of pre-trained language models, they cannot successfully incorporate user feedback during editing. PPLM-guided constrained decoding is not only two orders of magnitude slower than our denoising models (3min vs. 1s per recipe), but we observe poor fidelity and frequent incoherent instructions (e.g., repetition). Meanwhile, forcing ingredient lists to omit or contain specific ingredients has little impact on the generated recipe instructions—even when the desired ingredient is manually inserted into the ingredients list, RecipeGPT and BART mention using the ingredient only in 33% and 41% of generated instructions.

Our model and gradient-based critiquing method leads to a stronger influence of the edited ingredients on recipe instructions. By directly modifying the recipe latent representation that is then attended over during step generation, RecipeCrit achieves 30-50% relative improvement in success rate for adding ingredients and 20-65% relative improvements in coherence (F1 score between predicted ingredients and those mentioned in the instructions) for both addition and removal. Meanwhile, baselines tend to ignore many ingredients in the ingredient list when generating new recipe directions.

Model	**Ser.**	**Cor.**	**Coh.**	**Rel.**
RecipeGPT	-0.04^*	-0.03^*	-0.01^*	-0.07^*
PPLM	-0.03^*	-0.05^*	0.01	0.00^*
BART	-0.05^*	-0.07^*	-0.09^*	-0.07^*
RecipeCrit	$\mathbf{0.12}$	$\mathbf{0.14}$	$\mathbf{0.10}$	$\mathbf{0.14}$

Table 2: Human evaluation of edited recipes in terms of best-worst scaling for serendipity, correctness, coherence, and relevance. $*$ denotes a significant difference compared to RecipeCrit (posthoc Tukey test, $p < 0.01$).

Human Evaluation We have established that RecipeCrit creates edits that better satisfy user constraints (as expressed via critiques), more closely resemble the user's original preferences (base recipe), and make better use of the predicted ingredients (ingredient coherence). We next perform a qualitative human evaluation of our edited recipes via Mechanical Turk, asking the user: how pleasantly surprised they were (Serendipity); whether the recipe respected their feedback (Correctness); how easy the recipe was to follow (Coherence); and whether the recipe resembled the original recipe (Relevance). We uniformly sampled 800 edited recipes (400 for adding and 400 for removing) across the ingredients to critique and showed them in random order. The annotators judged the edited recipes using best-worst scaling (Louviere et al., 2015) with scores normalized to $[-1, +1]$. Table 2 shows that our edited recipes are largely preferred on all criteria. Our results highlight that critiquing improves the coherence of generated recipes and their resemblance to the original ones.

Case Study Table 3 shows a sample of our best-performing baseline (BART) and RecipeCrit editing the "cherry tomato confit" recipe to include "kale". While both edited recipes include kale, RecipeCrit stays faithful to the user's preference for "tomato confit" while incorporating the new feedback: it makes a slightly different tomato confit but uses kale as the "fresh" or salad part of the dish. However, BART generates a *cocktail* recipe instead that ignores the base recipe: it's a drink rather than food, sweet rather than savory, and ig-

Cherry tomato confit (base recipe)	**BART**	**RecipeCrit (Ours)**
clove, oil, pepper, rosemary, salt, tomato	**kale**, cachaca, cream, ice, juice, liqueur, pineapple, rum, strawberries, sugar, water ~~clove~~, ~~oil~~, ~~pepper~~, ~~rosemary~~, ~~salt~~, ~~tomato~~	clove, **kale**, oil, pepper, rosemary, salt, tomato
1) preheat oven to 325 degrees 2) spread tomatoes and garlic on a sheet. 3) drizzle with oil, and sprinkle with rosemary, crushed red pepper, a large pinch of salt and several grinds of pepper. 4) bake until tomatoes are wrinkled and fragrant, about 45 minutes, shaking pan. 5) transfer tomato pan to a rack to cool. 6) discard garlic.	1) place ice cubes in a cocktail shaker. 2) add pineapple juice, coconut liqueurs, cachacca, cream and rum ; shake well add crushed ice to a collins glass. 3) add **kale** and strawberries to shaker. 4) strain drink into glass over crushed ice. 5) garnish with strawberry and pineapple.	1) heat oven to 350 degrees. 2) place tomatoes in a large bowl. 3) drizzle with olive oil and sprinkle with rosemary, salt and pepper; coat. 4) spread in a single layer on a sheet. 5) roast for 40 minutes. 6) remove and let cool for 10 minutes. 7) toss **kale** with tomatoes and garlic.

Table 3: Comparison of a cherry tomato confit recipe with its edited versions to include "**kale**" as an additional ingredient. RecipeCrit proposes tomato confit with **kale**, but BART disregards the base recipe to make a cocktail.

nores tomatoes altogether. This aligns with the results of the human evaluation. Complementary results are shown in Table 5 and 6.

RQ2: Variants of Critiquing Algorithms Now we show the significance of the early stopping mechanism in our particular critiquing module compared to previous thresholding methods (Antognini et al., 2021a; Wang et al., 2019). To demonstrate why, we re-run experiments from RQ1 and compare our early stopping against two baseline thresholding criteria using 1) the absolute difference (i.e., $|C(\mathbf{z}_t^*)_c - \tilde{\mathbf{y}}_c^{ing}| < \tau$) and 2) the L1 norm (i.e., $||C(\mathbf{z}_t^*) - \tilde{\mathbf{y}}^{ing}||_1 < \tau$). We find that an L1-based stopping criterion is suboptimal due to the high dimensionality of the ingredients. Using the absolute difference considerably improves the success rate ($+25\%$ for add and $+12\%$ for remove). Finally, our early stopping further increases the success rate ($+10\%$) for both adding and removing an ingredient (see App. for exact numbers).

4 Conclusion

We present RecipeCrit, a denoising-based model to edit cooking recipes. We first trained the model for recipe completion to learn semantic relationships between the ingredients and the instructions. The novelty of this work relies on the user's ability to provide ingredient-focused feedback. We designed an unsupervised method that substitutes the ingredients and re-writes the recipe text accordingly. Experiments show that RecipeCrit can more effectively edit recipes compared to strong baselines, creating recipes that satisfy user constraints and are more serendipitous, correct, coherent, and relevant as measured by human judges. For future work, we plan to extend our method to large pre-trained language models for other generative tasks and to explainable models in the context of rationalization (Bastings et al., 2019; Antognini et al., 2021b; Lei et al., 2016; Yu et al., 2021; Antognini and Faltings, 2021).

5 Limitations

We demonstrated the effectiveness of our method for the English language since, to the best of our knowledge, there is no multi-lingual dataset similar to Recipe1M. We would expect similar behavior for languages having similar morphology to English.

Regarding computational resources, the training on a single GPU takes a couple of hours, while the inference and the critiquing can run on a single-core CPU (in the range of 10 to 100 ms).

Cooking recipes are long and complex documents. While current language models and similar ones have achieved impressive results, they still suffer from a lack of coherence for long documents. We have shown in our experiments that RecipeCrit produced recipes whose coherence is preferred over the baselines by human annotators. However, there is still room for improvement as language modeling approaches for recipe generation do not have an explicit guarantee of coherence (i.e. only listed ingredients used, instructions only make use of ingredients or products mentioned before).

Similarly, as recipe instructions can consist of free-text, there is no guarantee that recipe texts will, for example, completely remove an ingredient. In real-world usage, our system can be adapted by post-processing the recipe, including performing beam-search sampling and eliminating non-satisfactory recipes. As a result, we continue to urge caution for users with e.g. severe ingredient allergies who may still need to carefully review edited recipes to ensure compliance.

References

Diego Antognini and Boi Faltings. 2021. Rationalization through concepts. In *Findings of the Association for Computational Linguistics: ACL 2021*, pages 761–775, Online. Association for Computational Linguistics.

Diego Antognini, Claudiu Musat, and Boi Faltings. 2021a. Interacting with explanations through critiquing. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 515–521. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Diego Antognini, Claudiu Musat, and Boi Faltings. 2021b. Multi-dimensional explanation of target variables from documents. In *Proceedings of the AAAI Conference on Artificial Intelligence, (AAAI 2021)*, volume 35, pages 12507–12515.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguistics*, 8:64–77.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *EMNLP*.

Helena H. Lee, Shu Ke, Palakorn Achananuparp, Philips Kokoh Prasetyo, Yue Liu, Ee-Peng Lim, and Lav R. Varshney. 2020. Recipegpt: Generative pretraining based cooking recipe generation and evaluation system. In *WWW*.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Shuyang Li, Yufei Li, Jianmo Ni, and Julian J. McAuley. 2022. SHARE: a system for hierarchical assistive recipe editing. In *EMNLP*, pages 11077–11090. Association for Computational Linguistics.

Jordan J. Louviere, Terry N. Flynn, and A. A. J. Marley. 2015. *Best-Worst Scaling: Theory, Methods and Applications*. Cambridge University Press.

Yuran Pan, Qiangwen Xu, and Yanjun Li. 2020. Food recipe alternation and generation with natural language processing techniques. In *2020 IEEE 36th International Conference on Data Engineering Workshops (ICDEW)*, pages 94–97.

Diana Andreea Petrescu, Diego Antognini, and Boi Faltings. 2021. Multi-step critiquing user interface for recommender systems. In *Fifteenth ACM Conference on Recommender Systems*, RecSys '21, page 760–763, New York, NY, USA. Association for Computing Machinery.

Amaia Salvador, Michal Drozdzal, Xavier Giró-i-Nieto, and Adriana Romero. 2019. Inverse cooking: Recipe generation from food images. In *CVPR*.

Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. 2017. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*, pages 4222–4235. Association for Computational Linguistics.

ChunYuen Teng, Yu-Ru Lin, and Lada A. Adamic. 2012. Recipe recommendation using ingredient networks. In *WebSci*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Ryosuke Yamanishi, Naoki Shino, Yoko Nishihara, Junichi Fukumoto, and Aya Kaizaki. 2015. Alternative-ingredient recommendation based on co-occurrence relation on recipe database. In *KES*.

Mo Yu, Yang Zhang, Shiyu Chang, and Tommi Jaakkola. 2021. Understanding interlocking dynamics of cooperative rationalization. In *Advances in Neural Information Processing Systems*, volume 34. Curran Associates, Inc.

Table 4: Reconstruction performance. We report the IoU and F1 ingredient scores, and the Precision, Recall, and F1 scores of ingredients in predicted instructions w.r.t. predicted ones.

	Ingr. Fidelity		*Predicted Instr.*		
Model	**IoU**	**F1**	**Prec.**	**Rec.**	**F1**
RecipeGPT	73.5	84.7	61.2	72.6	66.4
BART	76.7	86.4	61.5	64.7	63.1
RecipeCrit	**78.6**	**88.2**	**68.2**	**73.0**	**70.5**

A Ingredient & Recipe Reconstruction

As baseline recipe generation models are unable to perform editing, we train all models using our denoising recipe completion task. To evaluate their generalization performance, we ask the models to reconstruct recipes from the unseen test set, with results shown in Table 4. We measure how well each model can infer the missing ingredients given the partial recipe context (IoU and F1 ingredient scores), as well as how coherent the reconstructed recipes are—the precision, recall, and F1 score of ingredients mentioned in the generated instructions compared to the predicted ingredients list.

RecipeCrit outperforms baselines in both measures. In particular, we find a significant improvement in ingredient mention precision, indicating that RecipeCrit better constrains its generated recipe directions based on the predicted ingredients list. Meanwhile, RecipeGPT and BART both tend to mention new ingredients in the recipe text even if they are not included in the ingredients list. As we see in Section 3, this is problematic because such models can include ingredients in the recipe steps even if users have specified dislikes or allergies.

Such text-to-text models capture the distribution of *language* well, producing fluent-sounding text. However, their lower scores for ingredient completion and recipe text coherence suggest that RecipeGPT and BART cannot distinguish how recipes are procedural texts with internal consistency, compared to generic text documents.

B Additional Training Details

We use a batch size of 32, dropout of 0.2, and Adam with learning rate 0.0001. For the baselines RecipeGPT and PPLM, we reuse the official code from the authors. For BART, we employ the HuggingFace library.

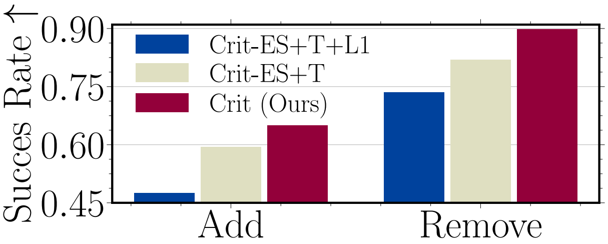


Figure 2: Critiquing algorithm comparison between global or local stopping criteria with threshold or early stopping.

B.1 Hardware / Software

- **CPU**: 2x Intel Xeon E5-2680 v3, 2x 12 cores, 24 threads, 2.5 GHz, 30 MB cache;
- **RAM**: 16x16GB DDR4-2133;
- **GPU**: 1x Nvidia Titan X Maxwell;
- **OS**: Ubuntu 18.04;
- **Software**: Python 3.6, PyTorch 1.6.1, CUDA 10.2.

Algorithm 1 Iterative Critiquing Gradient Update (Crit).

```
function CRITIQUE(latent vector $\mathbf{z}$, critiqued ingredient $c$, trained ingredients predictor $C$, decay coefficient $\zeta$, patience $P$, a maximum number of iterations $T$, desired ingredients $\tilde{\mathbf{y}}^{ing}$)
    Set $\mathbf{z}_0 = \mathbf{z}^* = \mathbf{z}, \alpha_0 = 1, \text{best\_val} = \infty, \text{patience} = 0, t = 1$;
    while patience $< P$ and $t < T$ do
        $\mathbf{g}_{t-1} = \nabla_{\mathbf{z}_{t-1}} \mathcal{L}_{ing}(C(\mathbf{z}_{t-1}), \tilde{\mathbf{y}}^{ing})$;
        $\mathbf{z}_t = \mathbf{z}_{t-1} - \alpha_{t-1} \frac{\mathbf{g}_{t-1}}{||\mathbf{g}_{t-1}||_2}$ and $\hat{\mathbf{y}}^{ing} = C(\mathbf{z}_t)$
        if $|\tilde{\mathbf{y}}_c^{ing} - \hat{\mathbf{y}}_c^{ing}| < \text{best\_val}$ then
            best_val $= \hat{\mathbf{y}}_c^{ing}, \mathbf{z}^* = \mathbf{z}_t$, and patience $= 0$
        else
            patience $=$ patience $+ 1$
        $\alpha_t = \zeta\alpha_{t-1}$ and $t = t + 1$;
return $\mathbf{z}^*$;
```

Table 5: Comparison of a cherry tomato confit recipe with its edited versions to include "**kale**" as an additional ingredient. None of those baselines add the critique ingredient. RecipeGPT disregards the base recipe. PPLM generates a recipe in the same style than the base one, but three times longer and with a lack of coherence.

Cherry tomato confit (base recipe)

clove, oil, pepper, rosemary, salt, tomato

1) preheat oven to 325 degrees
2) spread tomatoes and garlic on a sheet.
3) drizzle with oil, and sprinkle with rosemary, crushed red pepper, a large pinch of salt and several grinds of pepper.
4) bake until tomatoes are wrinkled and fragrant, about 45 minutes, shaking pan.
5) transfer tomato pan to a rack to cool.
6) discard garlic.

RecipeGPT

cream, cheese, margarine, milk, onion, pepper, salmon, salt, tea
~~clove~~, ~~oil~~, ~~rosemary~~, ~~tomato~~

1) saute green peppers and onions in margarine.
2) add cream cheese and stir until melted.
3) add velveeta, milk and corn.
4) heat until melted, stirring occasionally.
5) add salt.
6) garnish with thin strips of red pepper and cilantro.
7) serve with crackers.

PPLM

oil, pepper, tomato, ~~clove~~, ~~rosemary~~, ~~tomato~~

1) preheat the oven to 400 degrees f.
2) place a large roasting pan in the center of the oven.
3) heat the olive oil in a large skillet over medium heat until hot.
4) add the tomatoes and cook for about 3 minutes or until the skins begin to split.
5) place in the oven for about 30 to 40 minutes until the skins begin to blister and shrivel.
6) place the tomatoes in cold water to cool, then remove the skins.
7) in a bowl, combine the tomatoes and their juices, salt, and pepper, tossing well to coat evenly.
8) remove the tomatoes, place them on a sheet tray, and bake for about 45 minutes, until the skins begin to blister and shrivel.
9) the tomatoes should still be juicy, and the skins should be very wrinkly. remove the tomatoes from the oven and let cool.
10) remove the skins from the tomatoes and discard the skins. transfer the tomatoes to a clean food processor.
11) blend the tomatoes with the roasted garlic, and add the peppercorns in the last 3 minutes of the cooking time, just until you can no longer process the tomatoes into a paste. place a sheet of parchment paper 12) on a clean work surface.
13) place a sheet of the phyllo on the work surface.
14) (keep the rest of the oil covered with damp paper towel.
15) brush the sheet of phyllo with oil. repeat with two more sheets of phyllo to form the second sheet.
16) you should have four layers.
17) if you have extra sheets left, place on a baking sheet and toast in oven with the oven door propped open until the sheets become golden brown.
18) remove the phyllo sheets from the oven and place on a clean work surface. cut each layered sheet along the scored lines into quarters.
19) place 1 heaping tablespoon of the tomato-peppercorn mixture in the center of each of four phyllo sheets.
20) fold the phyllo over into a triangle and place on a parchment-lined baking sheet. bake at 400 degrees f for 7 to 8 minutes, until lightly browned.
21) cool slightly before removing from the baking sheet, and serve warm.

Table 6: Comparison of a baked parmesan fish recipe with its edited versions to include “**cream**” as an additional ingredient. RecipeCrit incorporates the cream into a single-batter style fry, while BART only uses cream as a “serve with sour cream” addition to the original recipe. RecipteGPT disregards the base recipe to make a pizza. PPLM generates a completely different and incoherent recipe, and it suffers from text degeneration.

Baked parmesan fish (base recipe)	**BART**	**RecipeCrit (Ours)**
catfish, cheese, egg, flour, milk, paprika, pepper, salt	**cream**, egg, fillets, flour, garlic, milk, paprika, parsley, pepper, salt, thyme ~~**catfish**~~, ~~**cheese**~~	catfish, cheese, **cream**, egg, flour, paprika, pepper, salt ~~**milk**~~
1) preheat oven to 350 degrees. 2) rinse fillets and pat dry with paper towel. 3) in a pie plate , combine parmesan cheese , flour , paprika , salt and pepper. 4) in another pie plate , combine milk and beaten egg. 5) dip fillet in egg mixture and then coat with parmesan cheese mixture. 6) place in a well greased 9 x 13 pan. 7) bake uncovered for 25-30 minutes or until fish easily flakes with a fork.	1) preheat oven to 350 degrees. 2) rinse fillets and pat dry with paper towel. 3) in a pie plate, combine parmesan cheese, flour, paprika, salt and in another pie plate or bowl, combine milk and beaten egg. 4) dip fillet in egg mixture and then coat with parmesa cheese mixture. 5) place in a well greased 9 x 13 pan. 6) bake uncovered for 25-30 minutes or until fish easily flakes with a fork. 7) remove from oven and let stand for 5 minutes. 8) serve with sour **cream** and lemon wedges. 9) enjoy!	1) preheat oven to 350 degrees f (175 degrees c). 2) grease a baking dish. 3) rinse fillets and pat dry with paper towel. 4) in a medium bowl , mix together the **cream** cheese , sour **cream** , flour , salt , paprika , and pepper. 5) beat in the eggs. 6) dip filles 7) pour the mixture into the baking dish. 8) bake in the preheated oven until the fish flakes easily with a fork , about 30 minutes.

RecipeGPT	**PPLM**
crusts, jalapeno, mushrooms, oil, popcorn, sauce, tomato ~~**catfish**~~, ~~**cheese**~~, ~~**egg**~~, ~~**flour**~~, ~~**milk**~~, ~~**paprika**~~, ~~**pepper**~~, ~~**salt**~~	basil, bread, butter, cod, garlic, eggplant, oil, onions, parmesan, parmigiano, parsley, pepper, salt, tomatoes ~~**catfish**~~, ~~**cheese**~~, ~~**egg**~~, ~~**flour**~~, ~~**milk**~~, ~~**paprika**~~, ~~**pepper**~~
1) combine barbecue sauce and tomato sauce in bowl ; set aside. 2) toss mushrooms with oil ; bake in 425 degrees f standard oven 5 min. 3) or until softened. 4) place crusts on sheet pan. 5) spread each crust with 1/2 cup sauce mixture. 6) top evenly with mushrooms, jalapenos, shrimp and cheese. 7) bake 7 min. 8) or until cheese is melted and golden brown. 9) slice each pizza into 8 pieces. 10) serve with remaining barbecue sauce mixture. 11) serve with pickled jalapeno slices.	1) preheat oven to 400 degrees. 2) wash the potatoes and put them in a saucepan with the **cream**. 3) bring **cream** to a boil and add the parmigiano-**cream** cheese and the **cream** and **cream**. 4) let simmer, stirring often, to make a soft creamy **cream** and thick **cream** cheesecreamcream. 5) season with salt and pepper set aside. melt the cream cheese and **cream** togethercream and **cream** together. 6) beat in the buttercreamcreamcreamcreamcreamcream is verycreamcreamcreamcreamcreamcreamcreamcream. 7) creamcreamcreamcreamcreamcreamcreamcreamingcream-creamcreamcreamcreamcreamcreamcream.

DiTTO : A Feature Representation Imitation Approach for Improving Cross-Lingual Transfer

Shanu Kumar **Abbaraju Soujanya** **Sandipan Dandapat**
Sunayana Sitaram **Monojit Choudhury**
Microsoft Corporation, India
`{shankum,asoujanya,sadandap,susitara,monojitc}@microsoft.com`

Abstract

Zero-shot cross-lingual transfer is promising, however has been shown to be sub-optimal, with inferior transfer performance across low-resource languages. In this work, we envision languages as domains for improving zero-shot transfer by jointly reducing the feature incongruity between the source and the target language and increasing the generalization capabilities of pre-trained multilingual transformers. We show that our approach, *DiTTO* , significantly outperforms the standard zero-shot fine-tuning method on multiple datasets across all languages using solely unlabeled instances in the target language. Empirical results show that jointly reducing feature incongruity for multiple target languages is vital for successful cross-lingual transfer. Moreover, our model enables better cross-lingual transfer than standard fine-tuning methods, even in the few-shot setting.

1 Introduction

Due to the emergence of pre-trained Massively Multilingual Transformers (MMTs) such as mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020) and mT5 (Xue et al., 2020), zero-shot cross-lingual transfer (Hu et al., 2020; Ruder et al., 2021; Lauscher et al., 2020; Ansell et al., 2021; Pfeiffer et al., 2022) has received significant attention in the NLP community. This approach originated due to the skew in resource distribution in languages (Joshi et al., 2020), with most languages of the world having a scarcity of labeled data. Zero-shot transfer involves fine-tuning the MMT with task-specific data in one or more source languages, followed by evaluation on target languages whose labeled instances are not used during fine-tuning. Accurate zero-shot transfer is crucially important for MMTs to be useful for low-resource languages.

The performance of MMTs drops in the following two cases - when the source and target languages exhibit dissimilar typological features, or when the size of pre-training data in the target language is limited (Lauscher et al., 2020; Ebrahimi et al., 2022). Two common techniques to improve zero-shot performance include few-shot cross-lingual transfer (Lauscher et al., 2020; Kumar et al., 2022) and the translate-train approach (Ruder et al., 2021; Ahuja et al., 2022). Several studies have been conducted comparing these approaches, of which (Ahuja et al., 2022) concludes that if the cost of machine translation is greater than zero, the optimal and lowest-cost performance is achieved with at least some manually labeled data (i.e. the few-shot method). Since annotating data is expensive for many languages (Dandapat et al., 2009; Sabou et al., 2012; Fort, 2016), we investigate improving cross-lingual zero-shot transfer using only unlabelled data in this paper.

Zero-shot Cross-lingual Transfer has been identified as an under-specified optimization problem (Wu et al., 2022). A majority of the solutions reports a high performance on the source language but fluctuating performance on target languages. Wu et al. (2022) use linear interpolation to prove that it is possible to obtain a subset of solutions which have optimal performance on both source and target languages. Furthermore, they also conclude that current optimization techniques cannot converge to this smaller subset of optimal solutions without the availability of labeled target language data. Aghajanyan et al. (2020) and Liu et al. (2021) have observed similar behavior in the zero-shot setup and hypothesize that sub-optimal zero-shot performance may be due to the degradation of generalizable representations of MMTs during the fine-tuning stage. This leads to the model trained on the source language not being able to generalize well to the target languages. MMTs have also been shown to be over-parameterized (Smith and Le, 2018; Kolesnikov et al., 2020; Zhang et al., 2021), which leads to memorizing the training data (source language) and achieving poor generalization during

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 385–406
May 2-6, 2023

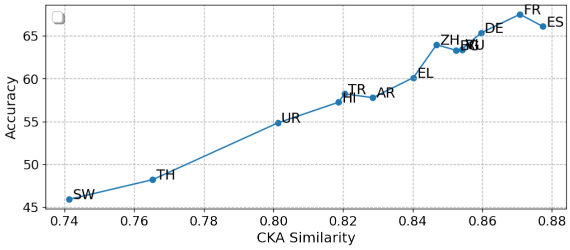


Figure 1: Relation between the zero-shot performance using mBERT, and CKA similarity between the source (EN) and various target languages in XNLI dataset.

cross-lingual transfer.

Similar to Deshpande et al. (2022), in our experiments, we also observe that once MMTs are fine-tuned on source languages, there is an incongruity between the features of the source and target languages, as shown in Figure 2. We speculate that the mismatch in the feature representation space causes problems in generalization. We also find that this mismatch strongly correlates with zero-shot performance as shown in Figure 1.

Furthermore, we hypothesize that this instability can be reduced either by finding solutions that can generalize well or learning to match the feature representations. Solutions (Zhang et al., 2018; Jiang et al., 2020) that have been used for improving generalization in other tasks can be considered, so that the model reaches to a better local minima. Sharpness-aware Optimization (SAM) (Foret et al., 2021) is one such technique that has been used to improve the generalization of language models (Bahri et al., 2022) and vision transformers (Chen et al., 2021) by smoothing the loss landscape for various adversarial tasks. SAM is used to generalize across domains, however, by treating languages as separate domains, we can apply SAM for generalizing across languages. While SAM looks promising, our experiments (cf. 7.3) showed that it does not guarantee optimal generalization at all times. We need to further reduce the incongruency between language features by aligning target language features to mimic the features of the source language. We propose ***DiTTO*** for improving cross-lingual transfer by source language **Di**rected adversarial **T**ransition of **T**arget language using sharpness aware **O**ptimization.

The key contributions of this work are: 1) Exhibiting the limitations of standard fine-tuning by unveiling the feature incongruity between source and target languages. 2) *DiTTO* enhances cross-lingual transfer by joint feature transformation of the multiple target languages to mimic the source. 3) *DiTTO* makes cross-lingual transfer cost-effective and efficient for distant (typologically different languages), resource-lean and unseen (not present in the pre-training data) languages. 4)*DiTTO* exhibits superior performance compared to augmenting the training data for either the source or the target language.

2 Related Work

Cross-Lingual Transfer: Since the inception of pre-trained MMTs, zero-shot learning has become popular for cross-lingual tasks. Recent works (Lauscher et al., 2020; Ebrahimi et al., 2022; Wu et al., 2022) have shown it to be sub-optimal for target languages which are either distant to the source language or have limited data during pre-training of the MMT. Some works (Wu and Dredze, 2020; Yu and Joty, 2021) have tried to improve the transfer using feature alignment from parallel data or bitexts (Zhang et al., 2020; Tiedemann, 2012) which is often expensive to obtain for many languages. To address this issue, *DiTTO* relies only on unlabeled data in the target languages. As pre-training size of the language affect transfer performance, adapter-based frameworks (Pfeiffer et al., 2020; Ansell et al., 2021) have been proposed for learning language and task representations for low-resource languages and languages that are unseen during pre-training. Though this framework is helpful for unseen languages, it provides limited gains for typologically dissimilar and high resource languages, and our method can easily be integrated with adaptors to further improve the transfer performance.

Improving Generalization: Deep neural networks such as MMTs are generally over-parameterized and fine-tuning leads to easy memorization of the labeled training data, does not always generalize well to other domains (Smith and Le, 2018; Kolesnikov et al., 2020; Zhang et al., 2021). Various methods have been proposed to improve the generalization like dropout (Srivastava et al., 2014), label smoothing (Müller et al., 2019), batch normalization (Ioffe and Szegedy, 2015), mixup (Zhang et al., 2018).

A few papers (Dziugaite and Roy, 2017; He et al., 2019; Jiang et al., 2020) have explored the connection between the flatness of minima and generalization gaps, showing flatter minima leads to better generalization. Recently, SAM has been proposed to find a smoother minima by minimizing the loss value and its sharpness. SAM has been shown to improve the generalization capabilities of vision transformers (Chen et al., 2021). Recently, Bahri et al. (2022) employed SAM in language models such as GPT-3 (Brown et al., 2020) and T5 (Raffel et al., 2020), showing significant improvements in generalization in English. In this work, we use SAM to improve the generalization across other languages. Another line of work (Aghajanyan et al., 2020; Liu et al., 2021) hypothesizes that inferior transfer is due to forgetting and degradation of feature representation from pre-trained MMTs when they are fine-tuned on the source language data. They propose to preserve the pre-trained features to improve the generalization using regularization and continual learning.

Unsupervised Domain Adaptation (UDA): Various studies have been proposed to reduce the domain shift to perform UDA by minimizing discrepancy distances such as Maximum Mean Discrepancy (MMD) (Long et al., 2015) and correlation alignment distance (Sun and Saenko, 2016). Adversarial-based feature alignment methods (Ganin and Lempitsky, 2015; Ganin et al., 2016; Long et al., 2018; Kurmi et al., 2019) have been one of the popular UDA methods where the domain discrepancy between the domains is reduced using an adversarial objective. In this work, we use Domain-Adversarial Neural Networks (DANN) (Ganin et al., 2016) for performing adversarial adaptation of languages.

3 Background

Training a Zero-Shot Model: In zero-shot cross-lingual transfer, we fine-tune an MMT on a source language and evaluate its performance on the target language, whose instances are not used during fine-tuning. To do this, we need a source language s and task-specific labeled dataset $\mathbb{L}_s = \{(x_i^s, y_i^s)\}_{i=1}^n$ with n examples. We use the provided MMT $\mathcal{M}$ as the encoder and fine-tune it along with the task-specific classifier C by minimizing the cross-entropy loss:

$$\mathcal{L}_{\text{train}}(\mathcal{M}, C) = \mathbb{E}_{(\mathbf{x}_i^s,\mathbf{y}_i^s)\sim\mathbb{L}_s} \mathcal{L}(C(\mathcal{M}(\mathbf{x}_i^s)), \mathbf{y}_i^s) \tag{1}$$

Sharpness-Aware Minimization (SAM): SAM seeks to find the parameter w such that even its neighborhood has seemingly similar low training loss $\mathcal{L}_{\text{train}}$ with minimal variation by optimizing the following objective:

$$\min_{w} \max_{||\epsilon||_2\le\rho} \mathcal{L}_{\text{train}}(w+\epsilon) \tag{2}$$

where ρ is the size of the neighborhood. Since, the exact solution of the inner maximization is hard to obtain, the authors of SAM propose a simple first order approximation:

$$\begin{aligned} &\epsilon(\hat{w}) \approx \\ &\arg\max_{||\epsilon||_2\le\rho} \mathcal{L}_{train}(w) + \epsilon^T \nabla_w \mathcal{L}_{\text{train}}(w) \\ &\qquad = \rho \nabla_w \mathcal{L}_{\text{train}}(w) / ||\nabla_w \mathcal{L}_{\text{train}}(w)||_2 \end{aligned} \tag{3}$$

After computing $\hat{\epsilon}$, the parameter w is updated based on the the sharpness-aware gradient $\nabla_w \mathcal{L}_{\text{train}}(w)|_{w+\epsilon(\hat{w})}$.

Domain-Adversarial Neural Networks (DANN): DANN (Ganin et al., 2016) has been successful applied for many unsupervised domain adaptation tasks for minimizing the domain shift (Du et al., 2020; Long et al., 2018). DANN needs a labeled source domain dataset $\mathbb{L}_s = \{(x_i^s, y_i^s)\}_{i=1}^n$ with n examples and an unlabeled target domain dataset $\mathbb{U}_t = \{x_i^t\}_{i=1}^m$ with m examples. It consists of three modules: Encoder $\mathcal{E}$, Task-Specific Classifier $\mathcal{C}$, and Domain Discriminator $\mathcal{D}$. In a nutshell, DANN requires solving a two-player game where the first player is the Domain Discriminator $\mathcal{D}$, is trained to distinguish the target domain from the source domain, and the second player is the encoder $\mathcal{E}$, which is trained simultaneously to confuse the Discriminator $\mathcal{D}$ such that the encoder learns to generate domain invariant features. We minimize the task-specific classification loss $\mathcal{L}_{\mathcal{C}}$ using the source domain labeled dataset for optimizing the classifier $\mathcal{C}$ and encoder $\mathcal{E}$.

$$\mathcal{L}_{\mathcal{C}}(\mathcal{E}, \mathcal{C}) = \mathbb{E}_{(\mathbf{x}_i^s,\mathbf{y}_i^s)\sim\mathbb{L}_s} \mathcal{L}(\mathcal{C}(\mathcal{E}(\mathbf{x}_i^s)), \mathbf{y}_i^s) \tag{4}$$

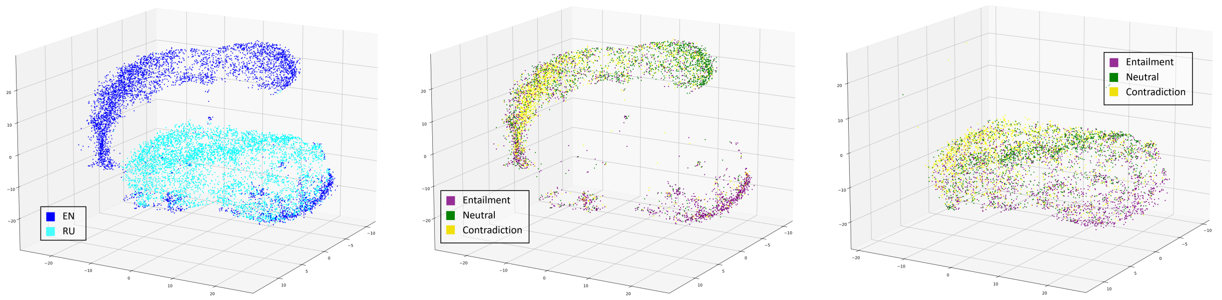
$\mathcal{D}$ is trained to predict the domains by minimizing domain classification loss:

$$\begin{aligned} \mathcal{L}_{\mathcal{D}}(\mathcal{E}, \mathcal{D}) = &-\mathbb{E}_{\mathbf{x}_i^s\sim\mathbb{L}_s} \log[\mathcal{D}(\mathcal{E}(\mathbf{x}_i^s))] \\ &-\mathbb{E}_{\mathbf{x}_j^t\sim\mathbb{U}_t} \log[1-\mathcal{D}(\mathcal{E}(\mathbf{x}_j^t))] \end{aligned} \tag{5}$$

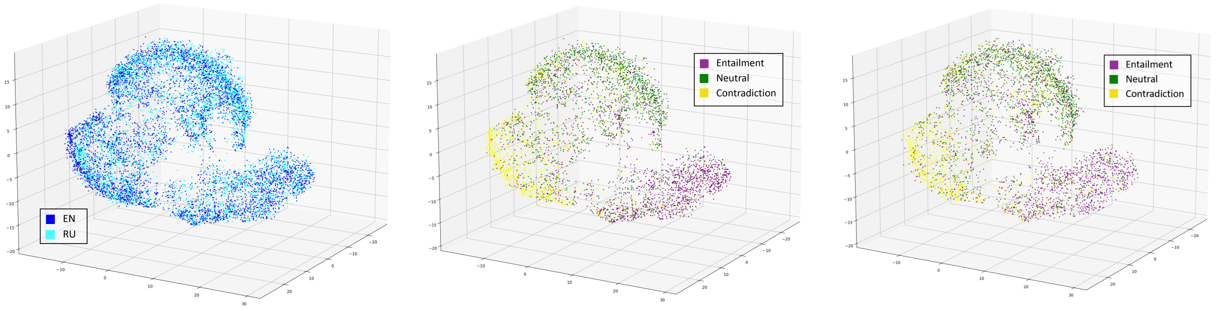
$\mathcal{L}_D$ is maximized for $\mathcal{E}$ so that $\mathcal{D}$ is not able to distinguish between the domains. The minimax optimization of DANN is defined as:

$$\begin{aligned} &\min_{\mathcal{E},C} \quad \mathcal{L}_C(\mathcal{E}, C) - \lambda \mathcal{L}_{\mathcal{D}}(\mathcal{E}, \mathcal{D}) \\ &\min_{\mathcal{D}} \quad \mathcal{L}_{\mathcal{D}}(\mathcal{E}, \mathcal{D}) \end{aligned} \tag{6}$$

where λ is a hyper-parameter to control trade-off between classification and domain adversarial loss.



(a) Language-wise Labels (Zero-shot) (b) Class-wise Labels (Source: EN) (c) Class-wise Labels (Target: RU)



(d) Language-wise Labels (*DiTTO*) (e) Class-wise Labels (Source: EN) (f) Class-wise Labels (Target: RU)

Figure 2: 3D t-SNE visualization of the features from the last layer of fine-tuned mBERT on XNLI (S=1%).

		mBERT			XLM-R		
Dataset	$\|\mathbb{T}\|$	1%	10%	100%	1%	10%	100%
XNLI	14	10.3	12.3	15.5	8.3	10.3	11.3
MARC	5	14.8	18.1	20.3	4.5	8.8	9.9
AmNLI	10	24.8	32.9	41.2	29.9	39.2	45.1

Table 1: We have reported the mean of difference $\triangle$ between the zero-shot performance of all the target languages and source language for varying amount of the source language (EN) data used while fine-tuning. $|\mathbb{T}|$ is the number target languages available in the dataset.

4 Limitations of Zero-shot Learning

Inconsistent Cross-Lingual Transfer: We have reported the average difference (δ) in the zero-shot performance between the target and the source language in Table 1. We experiment with mBERT and XLM-R on XNLI (Conneau et al., 2018), AmNLI (Ebrahimi et al., 2022) and MARC (Keung et al., 2020) datasets to measure the average δ between zero-shot performance of the target and source language. Table 1 shows that XNLI and AmNLI having relatively higher δ due to diverse number of languages. We also notice that mBERT has a higher δ than XLM-R across all tasks except AmNLI, showing the importance of amount of pre-training size.

Feature Incongruity between Languages: We hypothesize that the inconsistent zero-shot performance is due to the mismatch in the feature representation space of the fine-tuned MMT on the source language. To verify that we visualize the target and source language feature representations learned using standard zero-shot training method using 3D t-SNE (Van der Maaten and Hinton, 2008) in Figure 2. In Figure a, there is clear distinction between the source (En) and target language (Ru) features. While in Figure 2b and 2c, the feature space for the entailment class is overlapping with the source language, but fairly distinct for the other two classes, this could be potential cause for inferior cross-lingual transfer.

We measure centered kernel alignment (CKA) (Kornblith et al., 2019) between the source and the target language feature representations to quantify the incongruity. In Figure 1. We have plotted the CKA similarity with the zero-shot performance across all the languages. The plot suggests that there is a strong correlation between CKA and zero-shot performance, with Pearson and Spearman correlation coefficients as 0.98 and 0.96, respectively establishing our hypothesis.

5 Unveiling *DiTTO*

Typological similarity and incongruency between feature representations lead us to envision different languages as domains. As discussed in the previous section, DANN is useful in minimizing the domain shift across domains using only unlabeled data in the target domain. We propose to perform adversarial adaptation of the target language features for transforming the same towards the source language feature distribution.

We have a set of target languages $\mathbb{T}$ with each target language t having dataset $\mathbb{U}_t = \{x_i^t\}_{i=1}^{\mathcal{T}}$ with $\mathcal{T}$ unlabeled examples and an unlabeled set $\mathbb{U}_s = \{x_i^s\}_{i=1}^{\mathcal{S}}$ with $\mathcal{S}$ examples in the source language. In DANN, there is one target domain, whereas in our case we have a set of target languages $\mathbb{T}$ and we hypothesize and empirically show that performing adaptation for each language separately may cause degradation in other target languages, as seen in Table 5. Hence, we propose *DiTTO* where we jointly perform adaptation across all target languages.

DiTTO consists of an MMT $\mathcal{M}$ for encoding the features, a task-specific classifier $\mathcal{C}$ and Language Discriminators $\mathcal{D}^L = \{\mathcal{D}_t^L\}_{i=1}^{|\mathbb{T}|}$. We train these modules using two losses: task-specific classification loss $\mathcal{L}_{\mathcal{C}}$, defined in the Equation (1) and language discrimination loss $\mathcal{L}_{\mathrm{L}}$ for distinguishing the target and source language.

As we have $|\mathbb{T}|$ discriminators, we randomly sample a target language t from a prior distribution $p(\mathbb{T})$ at each training step and train the discriminator $\{\mathcal{D}_t^L\}$ to accurately distinguish target t and source language using the following loss:

$$\begin{aligned} \mathcal{L}_L(\mathcal{M}, \mathbf{D}_t^L) = -\mathbb{E}_{\mathbf{x}_i^s \sim \mathbb{U}_s} \log[b_t^L(\mathcal{M}(\mathbf{x}_i^s))] \\ -\mathbb{E}_{\mathbf{x}_j^t \sim \mathbb{U}_t} \log[1 - \mathcal{D}_t^L(\mathcal{M}(\mathbf{x}_j^t))] \end{aligned} \tag{7}$$

We maximize the above loss $\mathcal{L}_L(\mathcal{M}, \mathcal{D}_t^L)$ for confusing the language discriminator $\mathcal{D}_t^L$ to transform the target features towards the source language.

In our initial experiments (reported in Table 4), we observed some instability due to adversarial adaptation (Mao et al., 2017; Xing et al., 2021). We propose optimizing the task-specific loss $\mathcal{L}_C$ using SAM so that it may generalize to the target languages, improving the stability during adversarial adaptation. We directly fine-tune the MMT $\mathcal{M}$ on the source language labeled dataset $\mathcal{D}_s^l$ by minimizing Equation (1) using SAM. Following DANN and SAM, the final optimization objective of *DiTTO* can be defined as:

$$\min_{\mathcal{M}, C} \max_{||\epsilon||_2 \leq \rho} \mathcal{L}_C(\hat{\mathcal{M}}, \hat{C}) - \lambda \mathbb{E}_{t \sim p(\mathbb{T})} \mathcal{L}_L(\mathcal{M}, \mathcal{D}_t^L) \tag{8}$$

$$\min_{\mathcal{D}_L} \quad \mathbb{E}_{t \sim p(\mathbb{T})} \mathcal{L}_L(\mathcal{M}, \mathcal{D}_t^L) \tag{9}$$

where $\hat{\mathcal{M}}$, $\hat{C}$ are the updated parameters using ϵ.

6 Experimental Setup

6.1 Datasets

We evaluate our method on three benchmark datasets consisting of languages from various language families, to ensure better cross-lingual transfer evaluation. **XNLI** dataset (Conneau et al., 2018) consists of translated dataset in 14 languages from English. The task requires any model to predict whether the premise entails, contradicts, or neutral to the given hypothesis. AmericasNLI (**AmNLI**) dataset (Ebrahimi et al., 2022) is an extension of XNLI to 10 indigenous languages of the Americas, which are even unseen during pre-training of XLM-R and mBERT. Multilingual Amazon Review Corpus (**MARC**) dataset (Keung et al., 2020) is a large-scale dataset consisting of Amazon reviews for text classification in 6 languages. We use the review text and title to predict its star rating.

6.2 Baselines and *DiTTO* Variants

In the **Baseline** experiments, we fine-tune MMTs on labeled data of the source language using Equation (1). In the vanilla ***DiTTO*** setup, we use all the target languages available in the dataset. In the vanilla setup, we want to assign a higher probability to those target languages with a lower zero-shot performance from the Baseline method. We defined the prior distribution $p(\mathbb{T})$ of target languages as follows:

$$\Delta_t = \max(\mathcal{Z}(s) - \mathcal{Z}(t), 0) \tag{10}$$

$$p(t) = \delta_t + \sigma_{\Delta_t} \tag{11}$$

where, $\mathcal{Z}$ is the zero-shot performance from the Baseline method, Δ_t is the non-negative delta between the source and target language, and σ_δ is the standard deviation of the Δ_t across all the target languages.

DiTTO (UNF) is a variant of vanilla *DiTTO* in which we set the prior distribution $p(\mathbb{T})$ to be uniform across all the target languages. ***DiTTO* (t)** is a single target language variant of *DiTTO* where only one target language t is used during training. ***DiTTO*-LA** does not perform adaptation of the target languages, however optimization is done using SAM on the source language labeled data. ***DiTTO*-SAM** performs language adaptation without SAM.

6.3 Training Details

We conduct all of our experiments using mBERT (*bert-base-multilingual-cased*) and XLM-R (*xlm-roberta-base*). We use a batch size of 32 and a maximum sequence length of 128 across all the datasets. We fine-tune for $\{15, 20, 25\}$, $\{3, 5, 7\}$, $\{2, 3, 5\}$ epochs while using 1%, 10% and 100% of the source language data respectively. We use the AdamW (Loshchilov and Hutter, 2018) optimizer with linear scheduler and learning rate as 1e-5 for the encoder and classifier and 5e-5 for the discriminator. We set the λ hyper-parameter as 1 for all the experiments. We run experiments for each hyper-parameter and report the best average accuracy on three random seeds.

7 Results

In this section, we describe the results of several experiments to analyze the *DiTTO* method and compare its performance with the Baseline in the zero-shot setting. In order to justify the robustness of our method, we conduct experiments with the varying amount of source language data. In our experiments, EN is the default source language and we categorize target languages as follows: 1. *Distant*: languages that are typologically dissimilar to the source language 2. *Low-resource*: languages that have scarcity of data for pre-training 3. *Unseen*: languages that were not included in the pre-training data of MMT. Furthermore, we compare the techniques in the few-shot setting with few labeled examples in target languages. Then, we perform a thorough ablation study and analyze various variants of *DiTTO* . Finally, we show evidence in the form of congruity between the source and target language feature representations and t-SNE visualization in support of our hypothesis.

7.1 Zero-shot Transfer Results

Performance across datasets: In Table 2, we have reported the relative gains from *DiTTO* for zero-shot setting averaged across all the languages over the baseline method using 1%, 10% and 100% of the source language data. We observe the gains are positive (upto 23.05%) across all the training configurations. The gains are much higher for mBERT than XLM-R due to lower cross-lingual transfer in mBERT except the AmNLI dataset. The relative gains start to decrease with the increased amount of the source language data S on all the datasets except AmNLI, where the gains remains consistent for higher values of S (10% and 100%).

Performance across Seen Target Languages: We have reported the absolute gains of *DiTTO* in Figure 3 on XNLI using XLM-R We observe positive gains from *DiTTO* for all the target languages, with much larger gains especially on the low-resource and distant languages compared to the Baseline model. Similar to the earlier observation in Table 2, the gains starts to decrease across target languages as we increase the amount of the source language data.

	mBERT			XLM-R		
Dataset	1%	10%	100%	1%	10%	100%
XNLI	**23.05**	6.58	2.10	13.57	4.10	2.71
AmNLI	11.61	**19.72**	15.10	17.95	**19.87**	**19.09**
MARC	12.28	15.40	**19.03**	5.61	3.04	2.41

Table 2: Relative gains (in %) of *DiTTO* over Baseline. grey

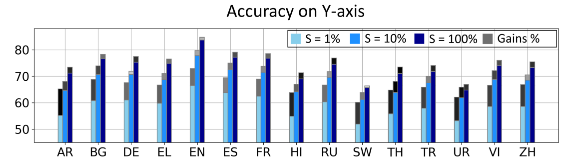


Figure 3: Absolute gains (darker shades of grey denotes higher gains) from *DiTTO* for XLM-R on XNLI dataset. Magnified view available in Figure 7 in Appendix.

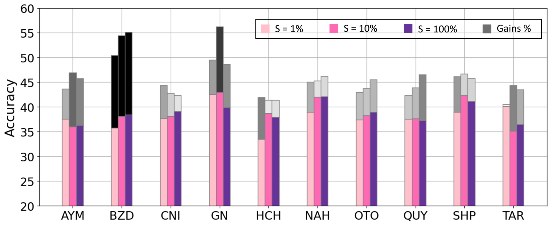
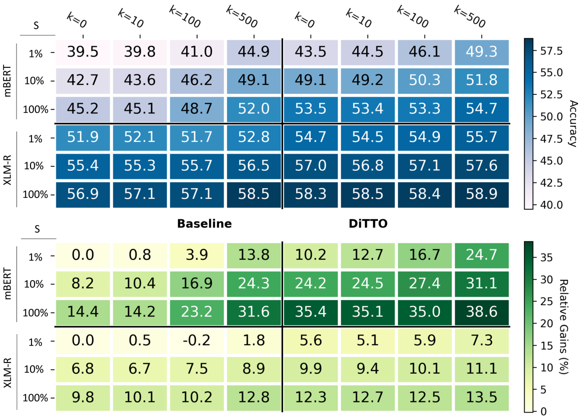


Figure 4: Absolute gains (darker shades of grey denotes higher gains) from *DiTTO* for XLM-R on AmNLI.

Performance across Unseen Target Languages: To measure the impact of *DiTTO* on unseen languages, we report the absolute gains from *DiTTO* on XLM-R on the AmNLI dataset in Figure 4. We have provided a similar analysis for mBERT in Figure 8 of the Appendix. The gains from *DiTTO* are consistent across all unseen languages. We observe that the gains are higher for languages with better Baseline performances, which is in contrast to trends on seen languages. For unseen languages, we do not observe the trend of diminishing gains with an increase in the source language data. If we compare the gains on AmNLI with the XNLI dataset, we notice *DiTTO* providing on average 1.7 times higher gains across all the configurations.

7.2 Few-shot Transfer Results

It can be argued that the gains from *DiTTO* in zero-shot setting can be achieved using few-shot cross-lingual transfer. Therefore, we conduct experiments in the few-shot setting by adding k labeled instances in each of the target languages to measure capabilities of *DiTTO* when some labeled data is available along with unlabeled data. In Figure 5, we have reported the accuracy and relative gains[1] using Baseline and *DiTTO* on MARC dataset. We have provided a similar analysis for AmNLI dataset in Figure 11 in the Appendix.



	S	Baseline k=0	Baseline k=10	Baseline k=100	Baseline k=500	DiTTO k=0	DiTTO k=10	DiTTO k=100	DiTTO k=500
mBERT	1%	39.5	39.8	41.0	44.9	43.5	44.5	46.1	49.3
mBERT	10%	42.7	43.6	46.2	49.1	49.1	49.2	50.3	51.8
mBERT	100%	45.2	45.1	48.7	52.0	53.5	53.4	53.3	54.7
XLM-R	1%	51.9	52.1	51.7	52.8	54.7	54.5	54.9	55.7
XLM-R	10%	55.4	55.3	55.7	56.5	57.0	56.8	57.1	57.6
XLM-R	100%	56.9	57.1	57.1	58.5	58.3	58.5	58.4	58.9

	S	Baseline k=0	Baseline k=10	Baseline k=100	Baseline k=500	DiTTO k=0	DiTTO k=10	DiTTO k=100	DiTTO k=500
mBERT	1%	0.0	0.8	3.9	13.8	10.2	12.7	16.7	24.7
mBERT	10%	8.2	10.4	16.9	24.3	24.2	24.5	27.4	31.1
mBERT	100%	14.4	14.2	23.2	31.6	35.4	35.1	35.0	38.6
XLM-R	1%	0.0	0.5	-0.2	1.8	5.6	5.1	5.9	7.3
XLM-R	10%	6.8	6.7	7.5	8.9	9.9	9.4	10.1	11.1
XLM-R	100%	9.8	10.1	10.2	12.8	12.3	12.7	12.5	13.5

Figure 5: Accuracy/relative gains on MARC dataset.

The heat maps in Figure 5 show that while XLM-R has better accuracy than mBERT for both Baseline and *DiTTO* setup, but the gains (both absolute and relative) on mBERT for both methods are higher compared to XLM-R. We also notice that by increasing either the source or the target language data, performance for both Baseline and DiTTO increased and hence, we will compare the gains of *DiTTO* with Baseline in these two dimensions.

Impact of Target Language Labeled Data: We observe that when we fix the amount of source language data and increase the value of k, the gains from *DiTTO* are higher than Baseline. Also, the gains from *DiTTO* on k=0 is comparable with the gains of baseline on k=500. In AmNLI, the gains from *DiTTO* for lower values of k are quite high compared to baseline, while for higher values of k Baseline performance for XLM-R is comparable to DiTTO.

Impact of Source Language Labeled Data: We also noticed that by fixing the value of k and increasing the size of source language data S, there is an increase in gains for both methods on MARC. However, the increase in gains from *DiTTO* is much higher than Baseline. At the same time, on the AmNLI dataset consisting of unseen target languages, the gains is much smaller with the increase in S (cf. Appendix).

Chinese as Source Language: To measure the effectiveness of *DiTTO* across different source languages, we conduct zero-shot experiments considering Chinese (ZH) as the source language on the MARC dataset. We have reported the average accuracy across all the languages in Table 3. *DiTTO* provides consistent gains over the Baseline method across all the training configurations, comparatively higher gains than EN as the source language.

	mBERT			XLM-R		
Dataset	1%	10%	100%	1%	10%	100%
Baseline	32.88	39.48	42.68	45.83	50.86	51.38
DiTTO	**39.09**	**46.90**	**50.82**	**51.84**	**53.34**	**55.27**
RG(%)	**19.45**	**20.80**	**20.67**	**13.31**	5.34	8.12

Table 3: We have reported the zero-shot accuracy averaged across all languages with **ZH** as the source language data on MARC dataset. RG denotes the relative gains averaged across all the languages from using *DiTTO* over Baseline.

Performance and Cost Trade-off: *DiTTO* is seven times more cost-effective in terms of both source and target language data. We validate this by plotting the accuracy from both methods against the cost incurred while collecting the labeled data for fine-tuning. For detailed analysis refer to the section B in the Appendix.

7.3 Ablations and Variants Analysis

Ablation Study: Here we scrutinize the contributions from adaptation of target languages and optimization with SAM. We report the zero-shot relative gains in Table 4 by ablating each of these components. We observe that removing any component reduces the performance for most of the training configurations, indicating that both target language adaptation and optimization have a contribution in achieving better results. We also observe that removing SAM (*DiTTO* - SAM) leads to unstable performances on XNLI and AmNLI datasets with negative relative gains on AmNLI (S=1%) for both MMTs, and on XNLI (S=10%) for mBERT, showing instability caused in adversarial training (Mao et al., 2017; Xing et al., 2021). Removing target language adaptation (*DiTTO*-LA) reduces the relative gains by a significant margin, showing the importance of adaptation of target language

---

[1] The relative gain is calculated with respect to the accuracy obtained by the Baseline method on $S = 1\%$ and $k = 0$.

features. It performs similar to *DiTTO* on XNLI (S=1%) dataset using mBERT, demonstrating just optimization using SAM can also improve cross-lingual transfer. The performance of *DiTTO* - SAM, it is often higher compared to *DiTTO* - LA, which indicates that Language Adaptation is a much more crucial for improving cross-lingual transfer.

		mBERT			XLM-R		
	Method	1%	10%	100%	1%	10%	100%
XNLI	*DiTTO*	**23.05**	**6.58**	**2.10**	**13.57**	**4.10**	**2.71**
	DiTTO - SAM	8.84	-0.36	1.80	6.04	1.81	2.02
	DiTTO - LA	22.25	3.89	2.02	7.43	2.81	1.74
MARC	*DiTTO*	**12.28**	**15.40**	**19.03**	**5.61**	**3.05**	**2.41**
	DiTTO - SAM	8.64	9.90	14.98	2.89	-0.13	2.27
	DiTTO - LA	5.5	1.54	2.20	4.02	0.35	-0.54
AmNLI	*DiTTO*	**11.61**	**19.72**	**15.10**	**17.95**	**19.87**	**19.09**
	DiTTO - SAM	-3.85	14.35	14.52	-11.88	14.81	15.89
	DiTTO - LA	7.21	5.17	-1.00	7.57	7.58	9.33

Table 4: Ablation Study: Zero-shot relative gains (in %) averaged across all the languages over Baseline.

Single vs Multiple Target Language Adaptation: In the base setup of *DiTTO* , we propose to perform an adaptation of all the target languages available in the dataset. We conduct zero-shot experiments with a single target language variant *DiTTO* (t) to validate our assumption. In Table 5, we observe that the single language variant provides similar gains as the vanilla *DiTTO* in the selected language t. However, often there is very little/no improvement observed in languages other than t. *DiTTO* (JA) and *DiTTO* (ZH) under-perform than Baseline for most of the languages.

Method	EN	DE	ES	FR	JA	ZH	AVG
Baseline	54.3	42.3	42.3	43.8	36.8	32.3	42.0
DiTTO (DE)	55.7	**48.2**	42.4	44.0	**36.5**	34.9	43.8
DiTTO (ES)	55.5	42.5	**45.9**	45.7	**36.4**	34.8	43.5
DiTTO (FR)	55.2	44.9	43.7	**46.4**	**36.5**	35.6	43.7
DiTTO (JA)	55.2	**40.9**	**41.3**	**42.5**	**38.1**	33.9	42.0
DiTTO (ZH)	**55.8**	**41.8**	**41.9**	**42.9**	**35.1**	**40.2**	43.0
DiTTO (UNF)	55.0	46.4	**46.2**	45.9	**38.5**	40.4	**45.4**
DiTTO	55.3	**47.0**	45.1	**46.2**	**38.6**	**40.7**	**45.5**

Table 5: Accuracy for single and multiple target language variants of *DiTTO* on MARC (S=1%, mBERT).

Target Language Prior Distribution: In *DiTTO* with multiple target language variant, the prior language distribution $p(\mathbb{T})$ is used to sample a target language for adaptation. To measure the importance of prior distribution, we experiment with two variants: (i) sampling based on the zero-shot performance of the Baseline method, which is used in the base setup of *DiTTO* and (ii) *DiTTO* (UNF) - with uniform sampling. Both the variants outperform Baseline with similar gains as shown in Table 5. In the vanilla *DiTTO* , where languages with lower zero-shot performance have a higher likelihood during sampling, provides better gains on these selected languages compared to the *DiTTO* (UNF).

Task-Adaptive Pre-training (TAPT): The Baseline method does not utilize the available unlabeled data in the target languages, whereas *DiTTO* uses the unlabeled data to improve the performance across all the target languages. Recently task-adaptive pre-training (TAPT) (Gururangan et al., 2020; Hossain et al., 2020; Caselli et al., 2021) using unlabeled task-specific data has been shown to improve the performance for pre-trained language models across multiple tasks. However, TAPT has yet to be evaluated in a multilingual setting.

To make a fair comparison, we have compared our proposed method with another baseline using unlabelled data, we shall refer this as *Baseline (TAPT)*. *TAPT* uses continued pre-training on the unlabeled target language data and fine-tuning is performed using the source language labelled dataset. We have reported the comparison between the new baseline method in Table 6. The *Baseline (TAPT)* method outperforms the Baseline method where unlabeled data is not used in the source language (EN); however, it regresses for all the target languages. We hypothesize that the TAPT method generally improves the performance of the language used during fine-tuning. Still, it suffers from similar issues which the Baseline method suffers, such as low feature congruity in the fine-tuned features between the languages. *DiTTO* , which does not suffer the feature incongruity issue, outperforms *Baseline (TAPT)* for all the languages.

Method	EN	DE	ES	FR	JA	ZH
Baseline	56.40	56.02	53.29	52.15	49.77	48.05
Baseline (TAPT)	**57.80**	55.58	52.14	51.70	49.60	45.71
DiTTO	**61.28**	**59.06**	**54.87**	**55.50**	**53.29**	**51.01**

Table 6: Comparison of Baseline and *DiTTO* methods with the new Baseline method using Task-Adaptive Pre-training (TAPT) on MARC (S=1%, XLM-R).

7.4 Congruity in Feature Representation

As shown in Figure 1 earlier that the zero-shot performance and feature congruity between the source and target languages are highly correlated. To validate our hypothesis that increasing the congruity between the features (via language adaptation) will improve the performance, we have plotted the increment in CKA similarity from *DiTTO* over the Baseline method in Figure 6. We observe increment in CKA similarity across all the languages using *DiTTO* , which is comparatively higher for distant or low-resource target languages. We also visualize the t-SNE projection of the feature representations of the source and the target languages in Figure 2. It is difficult to distinguish between both languages in this figure, showcasing the quality of language adaptation.

8 Discussion and Conclusion

In this work, we propose a novel method to improve the cross-lingual transfer capability of pre-trained MMTs. We find that zero-shot performance is correlated with incongruency between the features of source and target languages. Experiments show that our proposed method *DiTTO* outperforms the standard fine-tuning approach across multiple setups. In general, the gains from our method are higher on the models (as in mBERT) with less cross-lingual transfer. AmNLI consists only of languages that were not present in the pre-trained MMTs leading to similar transfer performance to the Baseline method. *DiTTO* improves cross-lingual transfer using the pre-training features, hence the gains from *DiTTO* are similar on both mBERT and XLM-R. Due to a similar reason, the relative gains for unseen languages do not follow the trend observed on seen languages, where the gains are higher for languages with the lower cross-lingual transfer. We find higher relative gains on unseen and low-resource languages, followed by distant languages. We also notice that the cross-lingual transfer improves with the amount of source language data S for seen languages. In contrast, for unseen languages, improvements are limited. Due to this, the gains from *DiTTO* start to decrease for high values of S for seen languages but remain significant for unseen languages.

Our method provides similar gains using only unlabeled data compared to the fine-tuned Baseline model (using 500 instances for each target language). Our ablation study shows that both LA and SAM are essential components of *DiTTO*, with LA being the primary contributor to the gains. Experiments show that single language adaptation improves on that corresponding target language but may regress on other languages as the feature may remain incongruent to the source. However, *DiTTO* that adapts to multiple target languages performs best. *DiTTO* tries to exploit the pre-training knowledge for improving the cross-lingual transfer, however few promising works such as adaptors (Pfeiffer et al., 2020; Ansell et al., 2021) have been proposed to improve the pre-training features for low-resource and unseen languages. However, task specific adaptors trained on the source language will also face the issue of incongruity in the feature representations. Hence, adaptors will not improve the cross-lingual transfer, but only improves the pre-trained features. We plan to extend our method towards integrating with adaptors to take advantage of pre-training features and improve performance.

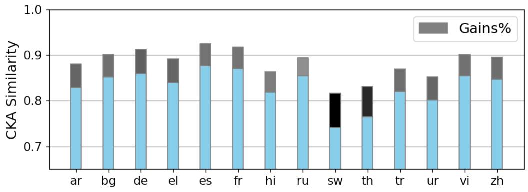


Figure 6: Gains in CKA similarity (between features of source and target language) from *DiTTO* over the Baseline method using mBERT on XNLI (S=10%).

9 Limitations

Unlabeled data in the target language is essential for the proposed method *DiTTO* for improving cross-lingual transfer. Obtaining unlabeled data can be challenging for specific tasks where the proposed approach may not be applicable. However, we recommend using the *DiTTO*-LA variant for these scenarios. Another limitation of *DiTTO* is that it requires all the target languages to be present during the fine-tuning stage to obtain the performances mentioned in our work, which might not be viable for all the tasks. Nevertheless, the gains from *DiTTO* may transfer to the new target languages if these languages are typologically similar to the target languages used during the fine-tuning of *DiTTO* . In the vanilla setup of *DiTTO* , the prior language probability depends upon the zero-shot accuracy using the Baseline method, which requires a validation or test dataset in each target language. This dependency may limit its application. However, *DiTTO* (UNF) can be used for obtaining similar gains if the validation sets are not available.

References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020. Better fine-tuning by reducing representational collapse. In *International Conference on Learning Representations*.

Kabir Ahuja, Monojit Choudhury, and Sandipan Dandapat. 2022. On the economics of multilingual few-shot learning: Modeling the cost-performance trade-offs of machine translated and manual data. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1369–1384, Seattle, United States. Association for Computational Linguistics.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. Mad-g: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781.

Dara Bahri, Hossein Mobahi, and Yi Tay. 2022. Sharpness-aware minimization improves language model generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7360–7371.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. 2021. When vision transformers outperform resnets without pre-training or strong data augmentations. In *International Conference on Learning Representations*.

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

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Sandipan Dandapat, Priyanka Biswas, Monojit Choudhury, and Kalika Bali. 2009. Complex linguistic annotation – no easy way out! a case from Bangla and Hindi POS labeling tasks. In *Proceedings of the Third Linguistic Annotation Workshop (LAW III)*, pages 10–18, Suntec, Singapore. Association for Computational Linguistics.

Ameet Deshpande, Partha Talukdar, and Karthik Narasimhan. 2022. When is BERT multilingual? isolating crucial ingredients for cross-lingual transfer. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3610–3623, Seattle, United States. Association for Computational Linguistics.

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

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware bert for cross-domain sentiment analysis. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, pages 4019–4028.

Gintare Karolina Dziugaite and Daniel M Roy. 2017. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir Meza Ruiz, Gustavo Giménez-Lugo, Elisabeth Mager, Graham Neubig, Alexis Palmer, Rolando Coto-Solano, Thang Vu, and Katharina Kann. 2022. AmericasNLI: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6279–6299, Dublin, Ireland. Association for Computational Linguistics.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*.

Karën Fort. 2016. Collaborative annotation for reliable natural language processing: Technical and sociological aspects.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Haowei He, Gao Huang, and Yang Yuan. 2019. Asymmetric valleys: Beyond sharp and flat local minima. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Tamanna Hossain, Robert L. Logan IV, Arjuna Ugarte, Yoshitomo Matsubara, Sean Young, and Sameer Singh. 2020. COVIDLies: Detecting COVID-19 misinformation on social media. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. 2020. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual Amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, Online. Association for Computational Linguistics.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. Big transfer (bit): General visual representation learning. In *European conference on computer vision*, pages 491–507. Springer.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR.

Shanu Kumar, Sandipan Dandapat, and Monojit Choudhury. 2022. "diversity and uncertainty in moderation" are the key to data selection for multilingual few-shot transfer. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1042–1055, Seattle, United States. Association for Computational Linguistics.

Vinod Kumar Kurmi, Shanu Kumar, and Vinay P Namboodiri. 2019. Attending to discriminative certainty for domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 491–500.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499.

Zihan Liu, Genta Indra Winata, Andrea Madotto, and Pascale Fung. 2021. Preserving cross-linguality of pre-trained models via continual learning. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 64–71, Online. Association for Computational Linguistics.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. 2018. Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems*, 32.

Jonas Pfeiffer, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the curse of multilinguality by pre-training modular transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3479–3495, Seattle, United States. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, and Melvin Johnson. 2021. XTREME-R: Towards more challenging and nuanced multilingual evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10215–10245, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marta Sabou, Kalina Bontcheva, and Arno Scharl. 2012. Crowdsourcing research opportunities: Lessons from natural language processing. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW '12, New York, NY, USA. Association for Computing Machinery.

Samuel L. Smith and Quoc V. Le. 2018. A bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Shijie Wu and Mark Dredze. 2020. Do explicit alignments robustly improve multilingual encoders? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4471–4482, Online. Association for Computational Linguistics.

Shijie Wu, Benjamin Van Durme, and Mark Dredze. 2022. Zero-shot cross-lingual transfer is under-specified optimization. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 236–248.

Yue Xing, Qifan Song, and Guang Cheng. 2021. On the algorithmic stability of adversarial training. *Advances in Neural Information Processing Systems*, 34:26523–26535.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Tao Yu and Shafiq Joty. 2021. Effective fine-tuning methods for cross-lingual adaptation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8492–8501.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

A Data Statistics

We have provided the statistics of training and test data after removing any duplicates in each of the target languages for all the datasets in Tables 7, 8, and 9.

B Performance and Cost Trade-off

From the above results, it seems that *DiTTO* is more cost-effective in terms of both source and target language data We validate this by plotting the accuracy from both methods against the cost incurred while collecting the labeled data for fine-tuning. Assuming there is no cost associated with collecting unlabeled data, we define the cost $\mathbb{C}$ for building a fine-tuning dataset as follows:

$$\mathbb{C} = c_s * n_s^l + c_s * c_{t/s} * k * |\mathbb{T}| \tag{12}$$

where c_s is the cost of obtaining one instance labeled in the source language and we assume it to be 3 cents considering EN as the source language. $c_{t/s}$ is the relative cost of obtaining labeled data in target language compared to the source language. We use Gaussian Process Regression with a dot product kernel for modeling performance with cost. In Figure 12 and 13, we plot the accuracy for various values of $c_{t/s}$ against the total cost incurred using mBERT on the MARC dataset, we observe a convex curve with increasing curvature as the value of $c_{t/s}$ increases. From the plot, we can see that higher accuracy can be achieved using *DiTTO* than Baseline at the same cost for all the values of $c_{t/s}$, showing the cost-saving nature of *DiTTO* with average savings of 7 times.

ISO	**Language**	Train	Test	XLM-R Group	mBERT Group
AR	Arabic	392403	5010	Distant	Distant
BG	Bulgarian	392335	5010	Distant	Distant
DE	German	392440	5010	Similar	Similar
EL	Greek	392331	5010	Distant	Distant
EN	English	392568	5010	Source	Source
ES	Spanish	392405	5010	Similar	Similar
FR	French	392405	5010	Similar	Similar
HI	Hindi	392356	5010	Distant	Low-Resource
RU	Russian	392318	5010	Similar	Similar
SW	Swahili	391819	5010	Low-Resource	Low-Resource
TH	Thai	392480	5010	Distant	Low-Resource
TR	Turkish	392177	5010	Distant	Distant
UR	Urdu	388826	5010	Low-Resource	Low-Resource
VI	Vietnamese	392416	5010	Distant	Distant
ZH	Chinese	392251	5010	Distant	Distant

Table 7: In this table, we have reported the target language categories and statistics of training and test data available in each language for XNLI dataset.

ISO	**Language**	Train	Test	XLM-R Group	mBERT Group
AYM	Aymara	743	750	Unseen	Unseen
CNI	Asháninka	658	750	Unseen	Unseen
BZD	Bribri	743	750	Unseen	Unseen
GN	Guaraní	743	750	Unseen	Unseen
NAH	Nahuatl	376	738	Unseen	Unseen
OTO	Otomí	222	748	Unseen	Unseen
QUY	Quechua	743	750	Unseen	Unseen
TAR	Rarámuri	743	750	Unseen	Unseen
SHP	Shipibo-Konibo	743	750	Unseen	Unseen
HCH	Wixarika	743	750	Unseen	Unseen

Table 8: In this table, we have reported the target language categories and statistics of training and test data available in each language for AmNLI dataset.

ISO	**Language**	Train	Test	XLM-R Group	mBERT Group
DE	German	199877	4993	Similar	Similar
EN	English	199891	4998	Source	Source
ES	Spanish	199726	4986	Similar	Similar
FR	French	199612	4986	Similar	Similar
JA	Japanese	199845	4995	Distant	Distant
ZH	Chinese	197418	4903	Distant	Distant

Table 9: In this table, we have reported the target language categories and statistics of training and test data available in each language for MARC dataset.

	$S = 1\%$			$S = 10\%$			$S = 100\%$		
Language	Baseline	*DiTTO*	RG	Baseline	*DiTTO*	RG	Baseline	*DiTTO*	RG
XLM-R									
EN	67.84	69.00	1.68	78.08	79.24	1.48	83.83	82.59	-1.48
AYM	37.47	43.60	16.37	36.00	46.93	30.37	36.27	45.73	26.10
BZD	35.73	50.40	41.04	38.13	54.40	42.66	38.40	55.07	43.40
CNI	37.60	44.27	17.73	38.13	42.80	12.24	39.07	42.27	8.19
GN	42.46	49.53	16.67	42.86	56.21	31.15	39.92	48.60	21.74
HCH	33.51	41.92	25.10	38.72	41.39	6.90	37.92	41.39	9.15
NAH	38.89	44.99	15.68	42.01	45.26	7.74	42.14	46.21	9.65
OTO	37.43	42.91	14.64	38.24	43.72	14.34	38.90	45.45	16.84
QUY	37.47	42.27	12.81	37.60	43.87	16.67	37.20	46.53	25.09
SHP	38.93	46.13	18.49	42.27	46.67	10.41	41.07	45.73	11.36
TAR	40.05	40.45	1.00	35.11	44.33	26.24	36.45	43.52	19.41
AVG	37.95	44.65	17.95	38.91	46.56	19.87	38.73	46.05	19.09
mBERT									
EN	62.53	64.83	3.67	71.20	73.05	2.61	81.18	79.64	-1.89
AYM	38.27	44.93	17.42	38.93	47.07	20.89	39.33	47.07	19.66
BZD	34.80	44.00	26.44	37.47	45.60	21.71	42.13	45.60	8.23
CNI	37.60	39.87	6.03	37.47	47.47	26.69	40.00	44.93	12.33
GN	40.19	46.86	16.61	38.85	49.80	28.18	41.79	51.67	23.64
HCH	34.98	40.85	16.79	36.98	45.79	23.83	39.92	44.59	11.71
NAH	40.79	44.72	9.63	42.28	46.07	8.97	43.90	48.92	11.42
OTO	38.10	38.64	1.40	37.43	41.58	11.07	37.97	44.39	16.90
QUY	38.67	39.47	2.07	36.53	42.80	17.15	38.00	43.87	15.44
SHP	38.40	40.53	5.56	40.13	46.93	16.94	41.73	46.67	11.82
TAR	35.91	40.99	14.13	36.85	44.86	21.74	35.65	42.72	19.85
AVG	37.77	42.09	11.61	38.29	45.80	19.72	40.04	46.04	15.10

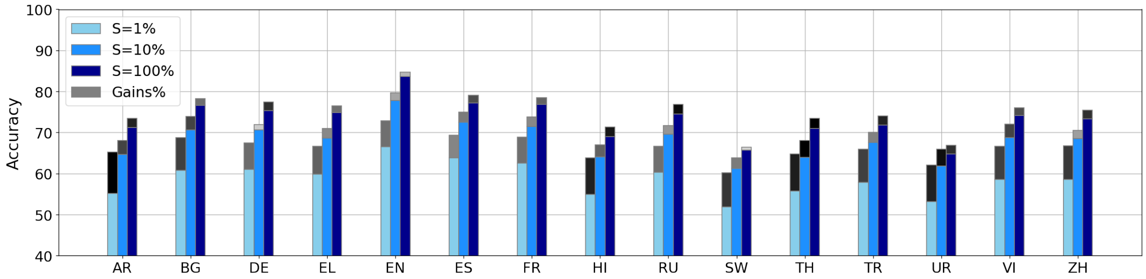
Table 10: We have reported the accuracy and relative gains using XLM-R and mBERT on AmNLI dataset. The average relative gain is denotes the average gains across all the languages except the source EN.

	$S = 1\%$			$S = 10\%$			$S = 100\%$		
Language	Baseline	*DiTTO*	RG	Baseline	*DiTTO*	RG	Baseline	*DiTTO*	RG
				XLM-R					
AR	55.25	65.19	17.99	64.69	67.96	5.06	71.16	73.39	3.14
BG	60.78	68.84	13.27	70.56	73.87	4.70	76.59	78.16	2.06
DE	60.98	67.50	10.70	70.64	71.92	1.81	75.33	77.37	2.70
EL	59.78	66.75	11.65	68.72	71.02	3.34	74.91	76.47	2.08
EN	66.49	72.91	9.67	77.80	79.72	2.46	83.71	84.65	1.12
ES	63.77	69.44	8.89	72.46	74.99	3.50	77.17	79.12	2.53
FR	62.51	68.86	10.15	71.52	73.77	3.15	76.85	78.50	2.16
HI	54.85	63.81	16.34	64.07	67.05	4.64	68.98	71.32	3.39
RU	60.32	66.67	10.52	69.62	71.66	2.92	74.49	76.93	3.27
SW	51.86	60.16	16.01	61.34	63.75	3.94	65.67	66.39	1.09
TH	55.81	64.75	16.02	63.89	68.14	6.65	70.96	73.45	3.52
TR	57.88	65.89	13.83	67.62	69.98	3.48	71.82	74.03	3.09
UR	53.17	62.12	16.82	61.94	65.91	6.41	64.83	66.95	3.26
VI	58.56	66.65	13.80	68.82	72.12	4.79	74.05	75.99	2.61
ZH	58.58	66.79	14.00	68.46	70.52	3.00	73.25	75.43	2.97
Average	58.15	65.96	13.57	67.45	70.19	4.10	72.57	74.54	2.71
				mBERT					
AR	47.09	56.75	20.52	57.78	62.87	8.81	63.07	65.21	3.39
BG	50.00	60.26	20.52	63.31	66.47	4.98	68.78	68.50	-0.41
DE	49.44	60.10	21.56	65.35	67.92	3.94	70.00	72.02	2.88
EL	48.70	59.08	21.31	60.12	64.63	7.50	65.91	66.99	1.64
EN	57.17	64.87	13.48	72.00	74.97	4.13	81.34	82.67	1.64
ES	50.12	62.38	24.45	66.11	70.88	7.22	73.11	75.43	3.17
FR	51.96	61.40	18.17	67.52	69.06	2.28	72.63	74.91	3.13
HI	46.57	54.93	17.96	57.25	60.58	5.82	60.26	62.02	2.91
RU	49.64	58.82	18.50	63.39	66.35	4.66	67.70	68.98	1.89
SW	37.82	46.51	22.96	45.91	49.20	7.17	50.68	49.42	-2.48
TH	36.61	53.31	45.64	48.20	56.21	16.60	53.85	57.03	5.89
TR	45.35	57.25	26.23	58.22	61.26	5.21	62.20	61.42	-1.25
UR	45.19	53.91	19.30	54.83	59.10	7.79	58.74	59.64	1.53
VI	49.20	59.98	21.91	63.45	67.25	5.98	69.46	71.44	2.84
ZH	48.74	60.26	23.63	63.97	66.63	4.15	68.64	71.60	4.30
Average	46.89	57.50	23.05	59.67	63.46	6.58	64.65	66.04	2.10

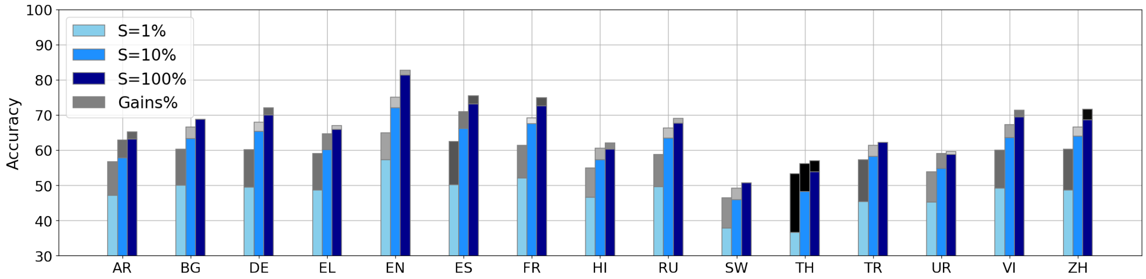
Table 11: We have reported the accuracy and relative gains using XLM-R and mBERT on XNLI dataset. The average relative gain is denotes the average gains across all the languages except the source EN.

	$S = 1\%$			$S = 10\%$			$S = 100\%$		
Language	Baseline	*DiTTO*	RG	Baseline	*DiTTO*	RG	Baseline	*DiTTO*	RG
XLM-R									
EN	56.40	61.28	8.66	64.17	64.37	0.31	66.81	66.91	0.15
DE	56.02	59.06	5.43	60.54	62.11	2.58	63.31	64.11	1.27
ES	53.29	54.87	2.97	56.34	56.96	1.10	57.68	58.80	1.95
FR	52.15	55.50	6.42	56.62	57.72	1.95	58.44	58.76	0.55
JA	49.77	53.29	7.08	52.93	55.46	4.77	53.77	55.98	4.10
ZH	48.05	51.01	6.15	50.34	52.74	4.78	51.50	53.66	4.20
Average	51.86	54.75	5.61	55.35	57.00	3.04	56.94	58.26	2.41
mBERT									
EN	54.32	55.82	2.76	60.80	62.77	3.22	65.53	65.71	0.27
DE	42.32	46.75	10.46	44.30	52.55	18.63	48.61	58.90	21.18
ES	42.30	45.81	8.30	45.77	51.18	11.83	49.56	54.75	10.48
FR	43.76	47.31	8.11	48.28	51.42	6.52	49.74	55.31	11.21
JA	36.82	38.66	5.00	39.00	43.78	12.27	39.32	48.77	24.03
ZH	32.31	41.85	29.55	36.32	46.40	27.74	38.67	49.60	28.27
Average	39.50	44.08	12.28	42.73	49.07	15.40	45.18	53.47	19.03

Table 12: We have reported the accuracy and relative gains using XLM-R and mBERT on MARC dataset. The average relative gain is denotes the average gains across all the languages except the source EN.

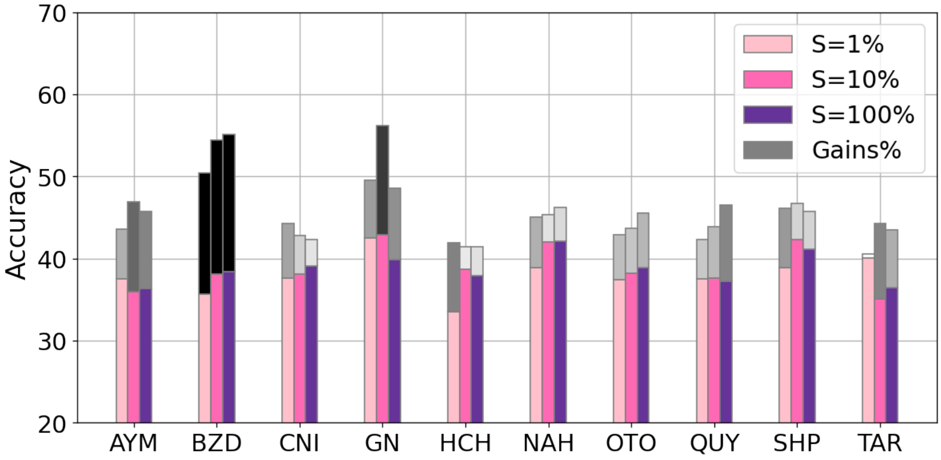


(a) XLM-R

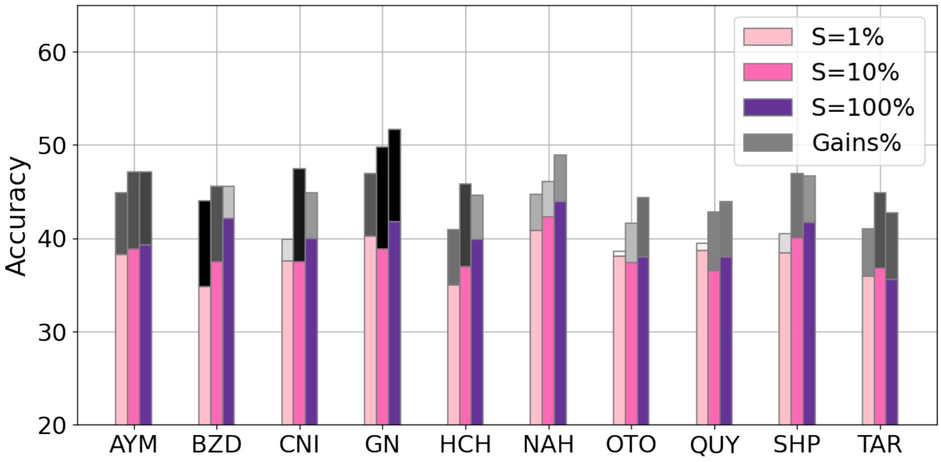


(b) mBERT

Figure 7: Absolute gains (darker shades of grey denotes higher gains) from *DiTTO* across all target languages on XNLI dataset.

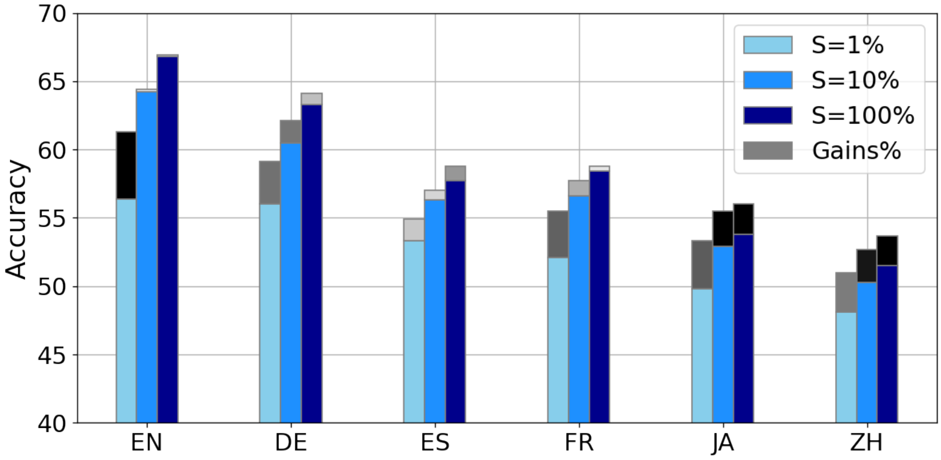


(a) XLM-R

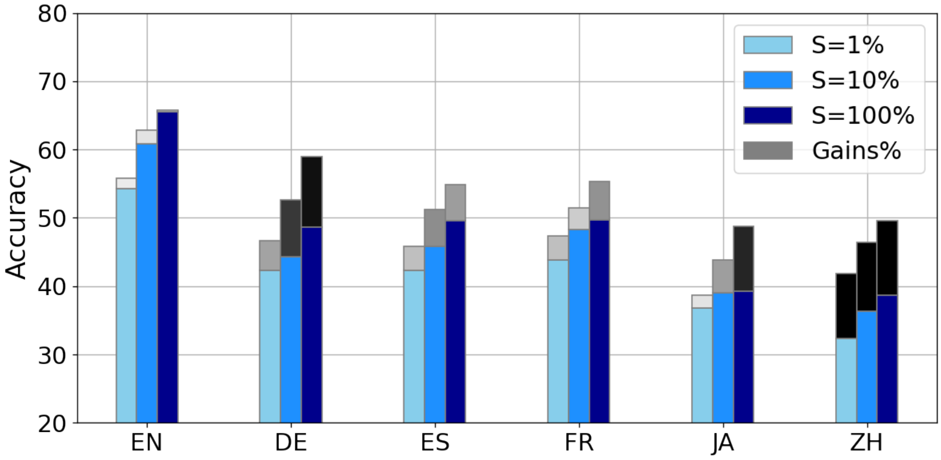


(b) mBERT

Figure 8: Absolute gains (darker shades of denotes higher gains) from *DiTTO* across all target languages on AmNLI dataset.

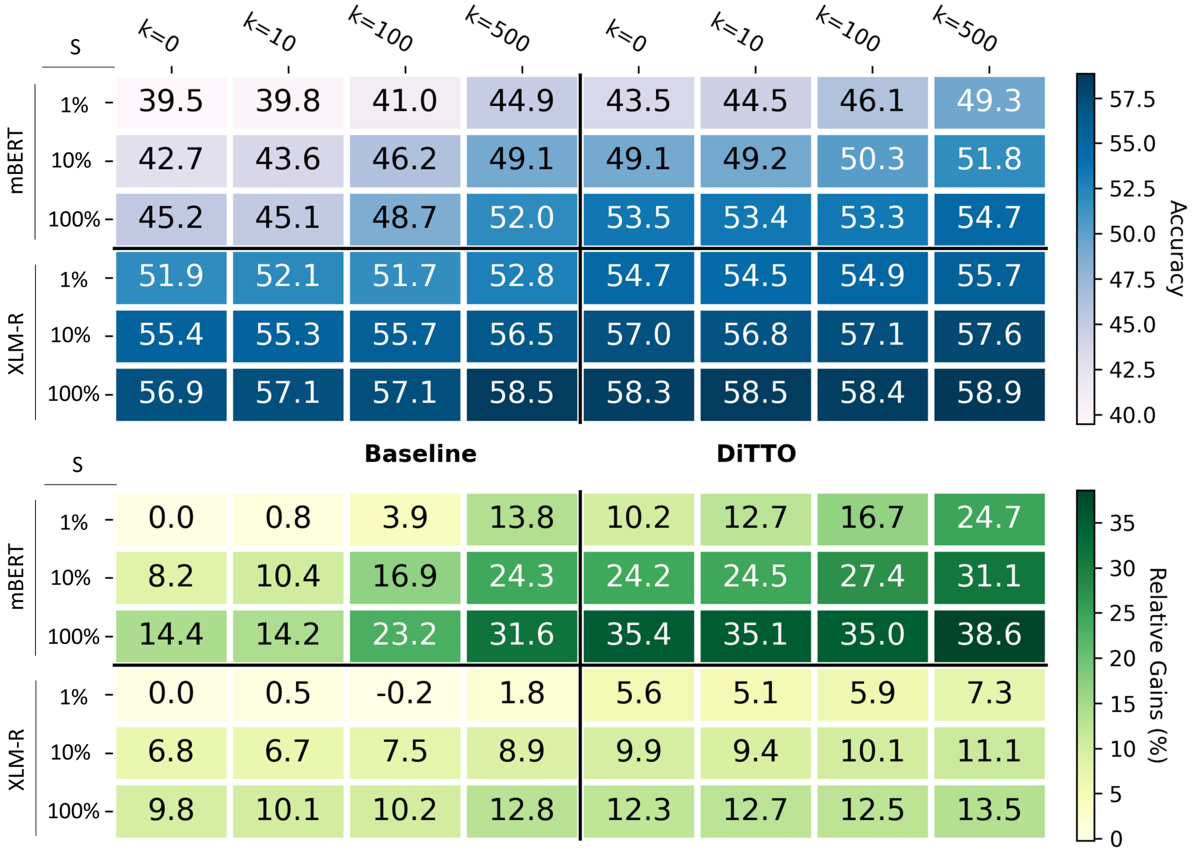


(a) XLM-R



(b) mBERT

Figure 9: Absolute gains (darker shades of denotes higher gains) from *DiTTO* across all target languages on MARC dataset.

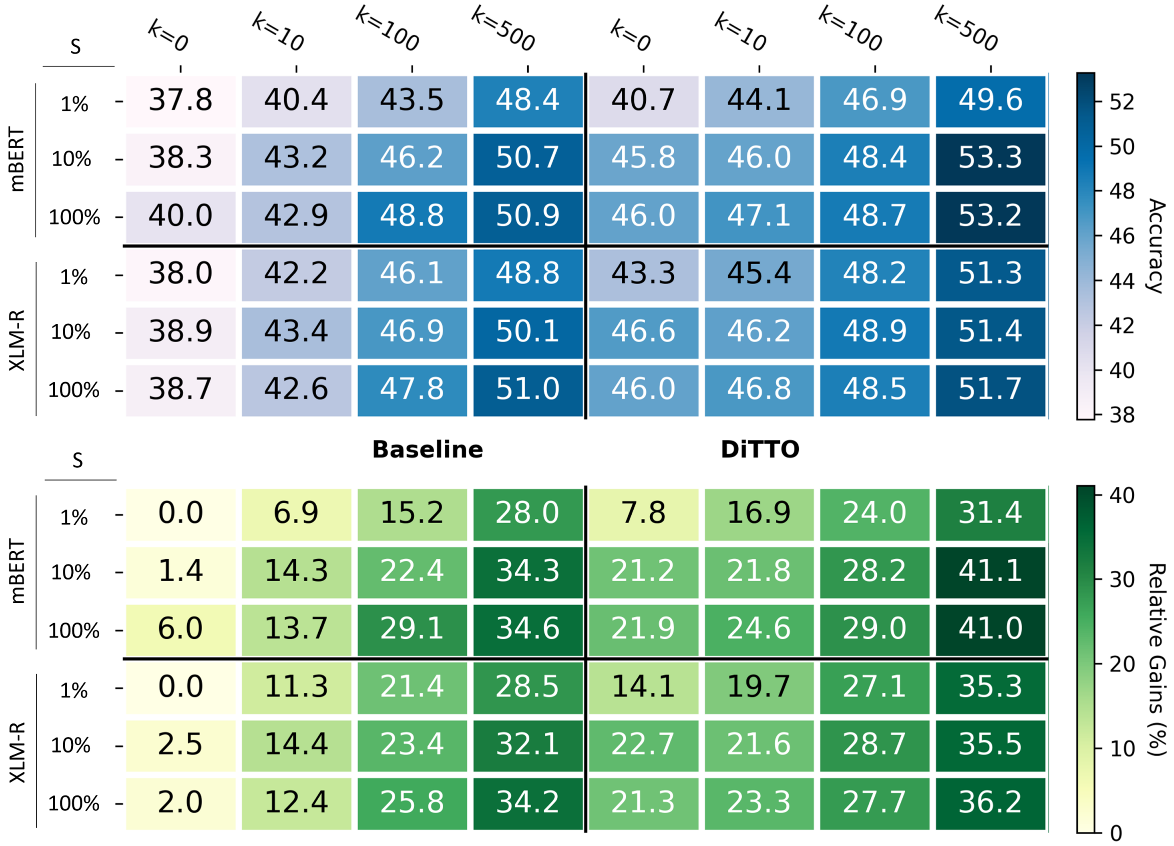


	S	k=0	k=10	k=100	k=500	k=0	k=10	k=100	k=500
mBERT	1%	39.5	39.8	41.0	44.9	43.5	44.5	46.1	49.3
	10%	42.7	43.6	46.2	49.1	49.1	49.2	50.3	51.8
	100%	45.2	45.1	48.7	52.0	53.5	53.4	53.3	54.7
XLM-R	1%	51.9	52.1	51.7	52.8	54.7	54.5	54.9	55.7
	10%	55.4	55.3	55.7	56.5	57.0	56.8	57.1	57.6
	100%	56.9	57.1	57.1	58.5	58.3	58.5	58.4	58.9

Baseline **DiTTO**

	S								
mBERT	1%	0.0	0.8	3.9	13.8	10.2	12.7	16.7	24.7
	10%	8.2	10.4	16.9	24.3	24.2	24.5	27.4	31.1
	100%	14.4	14.2	23.2	31.6	35.4	35.1	35.0	38.6
XLM-R	1%	0.0	0.5	-0.2	1.8	5.6	5.1	5.9	7.3
	10%	6.8	6.7	7.5	8.9	9.9	9.4	10.1	11.1
	100%	9.8	10.1	10.2	12.8	12.3	12.7	12.5	13.5

Figure 10: Accuracy/relative gains[2] on MARC dataset. Rows and columns denoting the amount of source and target language labeled instances, respectively.



	S	Baseline k=0	Baseline k=10	Baseline k=100	Baseline k=500	DiTTO k=0	DiTTO k=10	DiTTO k=100	DiTTO k=500
mBERT	1%	37.8	40.4	43.5	48.4	40.7	44.1	46.9	49.6
	10%	38.3	43.2	46.2	50.7	45.8	46.0	48.4	53.3
	100%	40.0	42.9	48.8	50.9	46.0	47.1	48.7	53.2
XLM-R	1%	38.0	42.2	46.1	48.8	43.3	45.4	48.2	51.3
	10%	38.9	43.4	46.9	50.1	46.6	46.2	48.9	51.4
	100%	38.7	42.6	47.8	51.0	46.0	46.8	48.5	51.7

	S	Baseline k=0	Baseline k=10	Baseline k=100	Baseline k=500	DiTTO k=0	DiTTO k=10	DiTTO k=100	DiTTO k=500
mBERT	1%	0.0	6.9	15.2	28.0	7.8	16.9	24.0	31.4
	10%	1.4	14.3	22.4	34.3	21.2	21.8	28.2	41.1
	100%	6.0	13.7	29.1	34.6	21.9	24.6	29.0	41.0
XLM-R	1%	0.0	11.3	21.4	28.5	14.1	19.7	27.1	35.3
	10%	2.5	14.4	23.4	32.1	22.7	21.6	28.7	35.5
	100%	2.0	12.4	25.8	34.2	21.3	23.3	27.7	36.2

Figure 11: Accuracy/relative gains[3] on AmNLI dataset. Rows and columns denoting the amount of source and target language labeled instances, respectively.

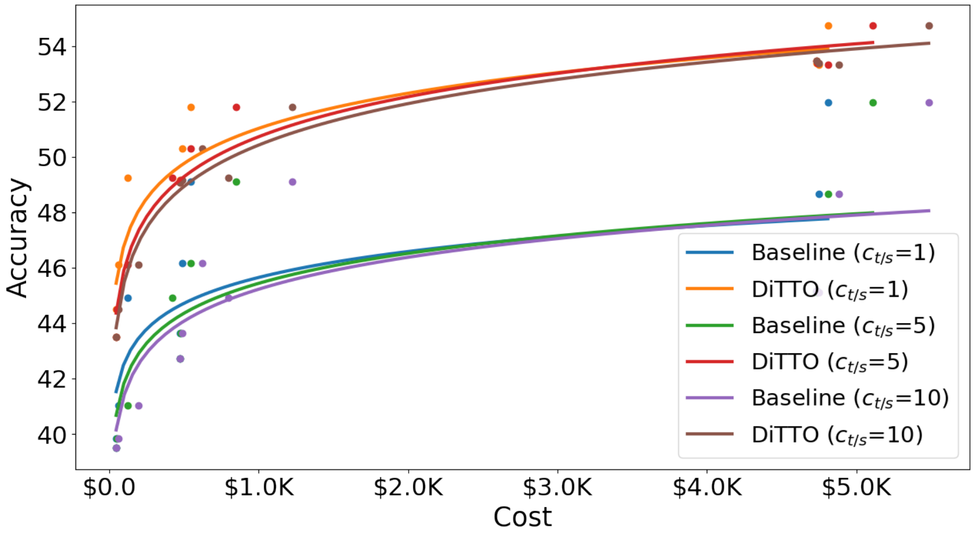


Figure 12: The plot shows Accuracy (vs) Cost graph with various values of $c_{t/s}$ for *DiTTO* and Baseline method trained using mBERT on XNLI (S=10%) dataset.

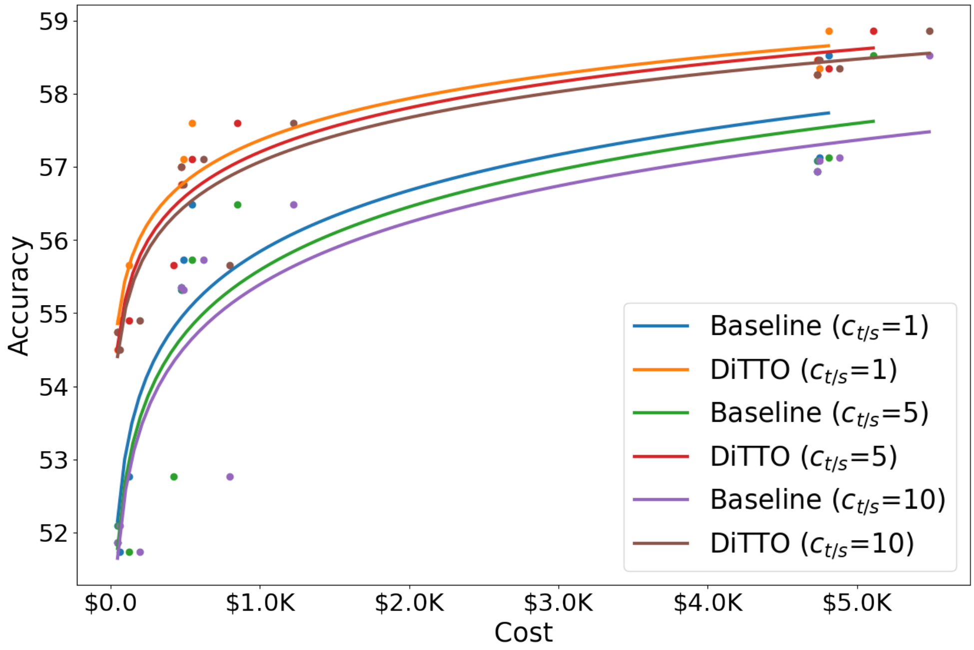


Figure 13: The plot shows Accuracy (vs) Cost graph with various values of $c_{t/s}$ for *DiTTO* and Baseline method trained using XLM-R on XNLI (S=10%) dataset.

John is 50 years old, can his son be 65?
Evaluating NLP Models' Understanding of Feasibility

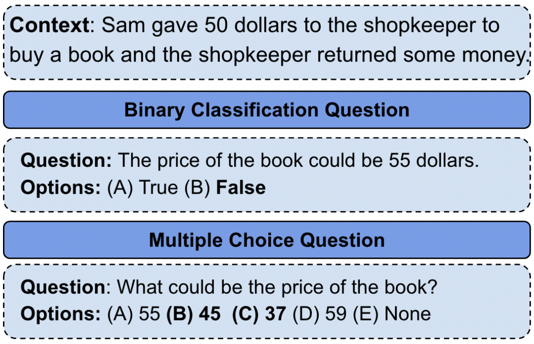
Himanshu Gupta **Neeraj Varshney** **Swaroop Mishra** **Kuntal Kumar Pal**
Saurabh Arjun Sawant **Kevin Scaria** **Siddharth Goyal** **Chitta Baral**
Arizona State University

Abstract

In current NLP research, large-scale language models and their abilities are widely being discussed. Some recent works have also found notable failures of these models. Often these failure examples involve complex reasoning abilities. This work focuses on a simple commonsense ability, reasoning about when an action (or its effect) is feasible. To this end, we introduce FeasibilityQA, a question-answering dataset involving binary classification (BCQ) and multi-choice multi-correct questions (MCQ) that test understanding of feasibility. We show that even state-of-the-art models such as GPT-3, GPT-2, and T5 struggle to answer the feasibility questions correctly. Specifically, on MCQ and BCQ questions, GPT-3 achieves an accuracy of just (19%, 62%) and (25%, 64%) in zero-shot and few-shot settings, respectively. We also evaluate models by providing relevant knowledge statements required to answer the question. We find that the additional knowledge leads to a 7% gain in performance, but the overall performance still remains low. These results make one wonder how much commonsense knowledge about action feasibility is encoded in state-of-the-art models and how well they can reason about it. [1]

1 Introduction

Commonsense reasoning has been a key aspect of AI since its advent in the 1950s. It is closely associated with reasoning about actions and their effects, which is considered a big challenge, especially for deep learning-based AI approaches and language models (LeCun, 2022; Dalvi et al., 2018; Banerjee et al., 2020). While several datasets have been developed to evaluate large-scale language models, in this paper, we propose a dataset focused on reasoning about actions and their effects; specifically, the ability to reason if an action or its effect is feasible.



Context: Sam gave 50 dollars to the shopkeeper to buy a book and the shopkeeper returned some money.

Binary Classification Question

Question: The price of the book could be 55 dollars.
Options: (A) True (B) **False**

Multiple Choice Question

Question: What could be the price of the book?
Options: (A) 55 **(B) 45** **(C) 37** (D) 59 (E) None

Figure 1: Illustrating a binary classification (BCQ) and a multiple choice question (MCQ) from FeasibilityQA. The correct answer options (**False** in BCQ and **(45, 37)** in MCQ) are highlighted in **bold**.

Figure 1 illustrates an example of our dataset; given the information "*Sam gave 50 dollars to the shopkeeper to buy a book and the shopkeeper returned some money*", it is not possible to compute the exact price of the book; however, it can be established that the feasible price of the book is *less than* 50 since the shopkeeper returned some money. We often come across such questions in our daily life and find it trivial to reason about them. Therefore, in order to develop NLP systems that can reliably reason about real-world situations, it is important to evaluate their understanding of feasibility.

Recently, many datasets have been created that test different reasoning skills such as pronoun resolution (Sakaguchi et al., 2021; Levesque et al., 2012), commonsense reasoning (Singh et al., 2021; Mihaylov et al., 2018; Banerjee et al., 2021), numerical reasoning (Mishra et al., 2022b; Ravichander et al., 2019; Lin et al., 2020; Zhang et al., 2020; Amini et al., 2019; Mishra et al., 2022e; Creswell et al., 2022; Pal and Baral, 2021), qualitative reasoning (Tafjord et al., 2019b,a), discrete reasoning (Dua et al., 2019), and temporal reasoning (Zhou et al., 2019). However, they do not have an ample number of examples that test understanding of feasibility.

---

[1] Dataset, baseline approaches, and instruction-tuned modeling approaches are freely available at `https://github.com/kevinscaria/feasibilityQA`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 407–417
May 2-6, 2023 ©2023 Association for Computational Linguistics

Category	Knowledge	Context	Questions
Attribute Comparison	Larger volume holds more amount of water.	Barrett's has two cylindrical shaped bottles. A bottle with a higher volume holds 32 units of water.	Amount of water in other bottle could be 28 units. Amount of water in other bottle could be 33 units. What could be the amount of water in other bottle?
Change with Time	Age increases with time	Today, while filling the application form Edward filled the age field with 16.	Edward could have been 8 years old on his last birthday. Edward could have been 19 years old on his last birthday. What could have been Edward's age last year?
Change with Action	Selling something reduces its quantity	Joshua organized a garage sale yesterday. Joshua sold a total of 273 items at a minimum price of 1 USD each.	Joshua could have made 300 dollars in the garage sale. Joshua could have made 260 dollars in the garage sale. How much money Joshua could have made from the garage sale?
Implicit Numerical Knowledge	4 quarters make 1 dollar	Christopher is accepting quarters for a donation and fails to collect 12 dollars.	He could have collected 35 quarters. He could have collected 52 quarters. How many quarters could Christopher have collected?
Non Numerical	New movies can be watched after the release date.	The latest superhero movie was releasing on 28th February 2022. Ashton wanted to see the movie.	He could have watched the movie on 2nd March 2022. He could have watched the movie on 3rd February 2022. When could Ashton have watched the movie ?

Table 1: Illustrative examples of two binary choice questions and one multiple choice question corresponding to a context statement. We also provide the corresponding category and relevant knowledge for the questions.

In this work, we address the above limitation and introduce FeasibilityQA, a dataset consisting of questions that require an understanding of feasibility. This dataset comprises of two types of questions: binary classification (BCQ) and multi-choice multi-correct questions (MCQ). In BCQ, the task is to determine whether the question is feasible or not given a context; in MCQ, the task is to select all feasible answers to the given question. The dataset consists of ∼5K instances covering diverse aspects of feasibility. Table 1 illustrates examples of various categories of feasibility questions.

We conduct comprehensive experiments with GPT-3, GPT-2, and T5 models (Brown et al., 2020; Radford et al., 2019; Raffel et al., 2020) in zero-shot and few-shot settings and show that all of these models struggle to correctly answer feasibility questions. Specifically, on (MCQ, BCQ) questions, GPT-3 achieves an accuracy of just (19%, 62%) and (25%, 64%) in zero-shot and few-shot settings, respectively.

Prior work has found that explicitly providing relevant knowledge helps the model reason better and improves its performance (Chen et al., 2018; Xiong et al., 2019; Banerjee et al., 2019; Varshney et al., 2022a). We explore this aspect of reasoning by explicitly providing relevant knowledge statements and find that it leads to ∼7% improvement in performance. However, the overall performance still remains low. We further investigate GPT-3's ability to reason about feasibility questions by prompting it to generate the reasoning chain. In many cases, we find that GPT-3 successfully generates the correct reasoning chain but still fails to output the correct answer. This analysis further leads to several interesting findings (Section 3).

2 FeasibilityQA

2.1 Dataset Creation

For creating data instances of FeasibilityQA, we first create a context statement that describes a real-life situation. Then, we write two binary classification questions and one multiple choice question conditioned on the context that tests the understanding of feasibility.

Dataset creation and verification process Seven computer science graduate students were involved in creating the dataset. Dataset creation consists of 3 phases. First, in the data creation stage, each student created 700 samples over the period of 3 months. In the next phase, each dataset creator's questions were verified by a different student to ensure fairness during data validation. The 3rd stage of the validation was done when all the questions were compiled and cross-verified. In each verification stage, the dataset creators rejected some samples where the inter-annotator agreement was low. [2]

2.1.1 Context Creation

We create context statements from real-life situations spanning diverse topics such as elementary physics, profit-loss scenarios, temporal comparisons, and quantity comparisons. We divide the contexts into the following five categories:

Attribute comparison: This category consists of questions that test feasibility aspects involving the comparison of attributes of two quantities. **Implicit numerical:** Questions in this category involve fundamental mathematical facts that test the ability to use those facts in real-world situations. **Change with action:** This category tests the abil-

---

[2] All the dataset creators are authors of the paper.

ity to perceive a change in an item or state as an outcome of an action. **Change with time:** Here, questions test the understanding of feasibility related to temporal-based events. **Non Numerical:** This category includes questions where numbers are not explicitly involved in reasoning about feasibility. Table 1 provides examples of these categories. More details about them are in Appendix A.

Motivation behind category selection The motivation behind developing a large language model such as GPT-3 is to mimic human intelligence and come closer to Artificial General Intelligence. We attempt to gauge the performance of models' intelligence by developing simple commonsense reasoning questions. GPT-3 models are few-shot learners but find it hard to do proper numerical reasoning. Earlier datasets like this attempted to analyze numerical reasoning in this aspect. We are also trying to study it in the aspect of feasibility. Previous datasets, such as Numersense (Lin et al., 2020) and MC Taco (Zhou et al., 2019), do not have such a category, and we tried to bridge those gaps.

We think that these five categories are also a good representation of numerical feasibility. We found that questions from those categories had an adequate amount of complexity that the average human could easily figure out. So we expected that large language models should also be able to understand and answer accordingly. We created these categories to compare the models' numerical reasoning ability with and without knowledge. This gives us insights into whether knowledge helps in each aspect. We hope that these comparative studies across these five preliminary categories will inspire more future categories.

Target of our dataset: Our selection of categories in feasibility is inspired by the limitations in existing datasets since it is not possible to cover all the aspects of feasibility exhaustively.

2.1.2 Question Creation

From each context, we create two binary classification and one multiple-choice question. Recall that in our questions, the context may not provide sufficient information to find the exact answer. However, the information is sufficient to test the validity of question/answer options (notice the use of the word '**could**'). In question creation, we ensure that all our contexts and questions describe realistic situations, e.g., we keep a range of numerical entities and units appropriate for their context. Table 1 illustrates examples of our dataset.

Category	**Binary Instances**	**MCQ Instances**
Attribute comparison	1696	848
Non numerical	700	350
Implicit numerical	444	222
Change with action	196	98
Change with time	36	18
Total	3072	1536

Table 2: Categorization of FeasibilityQA across different categories .

Correct Answers	#Instances
1	555
2	622
3	285
4	31
None	43

Table 3: Frequency of correct answers for MCQ section.

BCQ: For each context, we create two binary classification questions where the correct answer is *True* for one and *False* for the other. Evaluating models' consistency in answering two contrasting hypotheses conditioned on the same context provides an additional benefit.

MCQ: For each context, we create a multi-correct multiple-choice question. Here, a context-question pair and the corresponding answer options are given, and the task is to select all feasible options for the question. We ensure that there is also a *None* option, which needs to be selected when all the other options are not feasible. For a question, *one or more options* (including 'None') could be correct.

2.2 Dataset Statistics

Our dataset consists of 1536 contexts and 4608 context-question pairs (3072 BCQ and 1536 MCQ). The category-wise distribution of the dataset is present in Table 2. BCQ dataset is label balanced, i.e., 1536 instances for each of *True* and *False* labels. MCQ dataset has a varying number of correct options. Table 3 shows the number of correct answers in the MCQ category.

	GPT-3				**GPT-2**				**T5**			
	BCQ (%)		MCQ (%)		BCQ (%)		MCQ (%)		BCQ (%)		MCQ (%)	
	w/o K	w/ K	w/o K	w/ K	w/o K	w/ K	w/o K	w/ K	w/o K	w/ K	w/o K	w/ K
Zero Shot	62.96	69.11	19.43	25.89	50.00	50.00	0.19	0.45	50.55	50.64	0.13	0.39
One Shot	57.94	64.66	20.94	24.15	50.00	50.00	0.58	1.69	50.61	50.33	0.45	0.58
Few Shot	64.72	68.55	25.94	37.23	50.00	50.00	0.97	0.39	49.81	49.87	0.84	1.10

Table 4: *Exact match accuracy* of three models in BCQ (Binary Classification) and MCQ (multi-choice multi-correct) tasks across three settings. w/o K and w/ K represents without knowledge and with knowledge respectively.

3 Experiments

3.1 Performance Evaluation & Metrics

For **BCQ**, we calculate *exact match accuracy* against the gold label (True or False) for each question. We also use a stricter evaluation metric, *pairwise accuracy*, to better estimate the model's capability of understanding feasibility. Here we only consider a sample to be correct if both BCQ (True and False questions) are correctly answered by the model for one context statement. For **MCQ**, we evaluate *exact match accuracy*, i.e., the model's prediction is considered to be correct if all the correct answer options are predicted. We also compute *recall*, which we define as the number of correct answers predicted by the model out of all the correct answer options. Other results (false positive, false negative, category-wise) are in Appendix B.

Models: We evaluate the performance of **GPT-3** (Text-DaVinci-002, with 256 max tokens, top p of 1, and frequency & presence penalty of 0), **T5-11B**, and **GPT-2 large** models on our dataset.

3.2 Results

Low Performance of All Models: Table 4 shows the accuracy of all three models in zero-shot, one-shot, and few-shot settings. On BCQ, GPT-3 achieves *exact match accuracy* of just 62.9%. GPT-2 and T5 perform even worse and achieve close to the majority baseline (50%). GPT-2 gets an exact 50, indicating that the model does not understand such feasibility reasoning [3]. On MCQ, which is a more difficult task than BCQ, all models, including GPT-3, achieve a very low *strict accuracy* score. This highlights that feasibility questions are challenging for even state-of-the-art models.

Decrease in performance in one-shot setting: In the one-shot setting, the model's prediction is heavily influenced by the label of the example (one) presented to the model. This phenomenon is also observed in several prior zero-shot, and one-shot studies (Zhao et al., 2021). A similar trend is observed in the chain of thought experiments (results described in Table 8).

	w/o Knowledge	w/ Knowledge
Zero	46.2	49.9
One	63.5	64.5
Few	66.7	70.3

Table 5: *Recall* scores of GPT-3 on MCQ task.

	w/o Knowledge	w/ Knowledge
Zero Shot	42.9	56.8
One Shot	17.9	34.3
Few Shot	39.8	55.8

Table 6: *Pairwise Accuracy* of GPT-3 in BCQ Task.

Providing Knowledge Improves GPT-3's performance by $\sim$ 7% across all settings. The accuracy particularly increases ($\sim$ 12%) in the MCQ task in the few-shot setting. Although GPT-3 performs better than T5 and GPT-2, it achieves just 68.5% and 37.2% on BCQ and MCQs, respectively.

GPT-3 achieves High Recall Scores on MCQs: In Table 5, we show recall scores of GPT-3 on MCQs. GPT-3 achieves a high score (up to 70%), highlighting that it gives correct responses but fails to give all the correct responses.

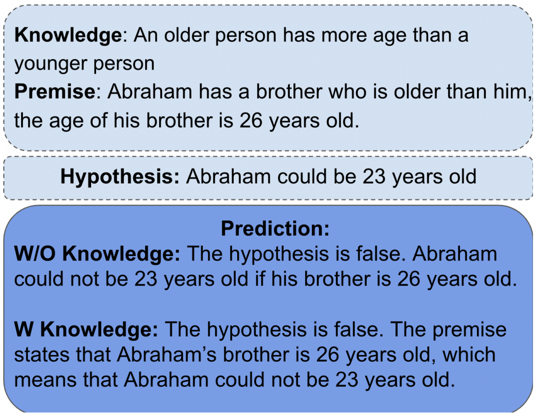
Pairwise Evaluation: Recently, instance-level analysis of the evaluation data has received considerable research attention (Zhong et al., 2021; Varshney et al., 2022b; Rodriguez et al., 2021; Mishra et al., 2022a). Motivated by this, we analyze GPT-3's performance on BCQ questions using the stricter *pairwise accuracy* metric. Even though the model performs $\sim$ 63% using *exact match accuracy*, Table 6 shows that the models' performance is at most $\sim$ 43% via *pairwise accuracy*, highlighting a performance gap. The accuracy increases ($\sim$ 13%) when knowledge is introduced, and the gap between different settings also narrows down, indicating that the addition of knowledge helps.

---

[3] Please refer Appendix B.2 for details

	BCQ		MCQ	
	w/o K	w/ K	w/o K	w/ K
Attribute Comparison	58.2	62.5	17.7	23.2
Non Numerical	77.2	89.4	23.6	30.9
Implicit Numerical	54.7	50.9	16.7	21.9
Change with Action	66.3	78.2	20.3	26.6
Change with Time	58.3	66.6	17.8	23.4

Table 7: Category wise *Exact Match Accuracy* of GPT-3 on BCQ and MCQ in one-shot setting.



Knowledge: An older person has more age than a younger person

Premise: Abraham has a brother who is older than him, the age of his brother is 26 years old.

Hypothesis: Abraham could be 23 years old

Prediction:

W/O Knowledge: The hypothesis is false. Abraham could not be 23 years old if his brother is 26 years old.

W Knowledge: The hypothesis is false. The premise states that Abraham's brother is 26 years old, which means that Abraham could not be 23 years old.

Figure 2: Answers with explanations given by GPT-3 on FeasibilityQA dataset.

	BCQ		MCQ	
	w/o K	w/ K	w/o K	w/ K
Zero Shot	61.3	70.2	20.1	25.1
One Shot	59.7	67.2	19.5	22.8
Few Shot	65.4	69.1	23.4	35.7

Table 8: *Exact Match Accuracy* of GPT-3 on BCQ and MCQ tasks with chain of thought setting.

Category wise results: Table 7 shows the category-wise results in one-shot setting for BCQ and MCQ tasks on GPT-3. Non Numerical category performed the best out of all categories (15% more than other categories). Addition of knowledge improves the performance of all categories by around 5% to 10%. Similar trends are observed across MCQ task as well. We believe that Non Numerical performed the best because these samples were conditioned around factual knowledge and that it could be present in the pretraining corpus of GPT-3. We further analyze this in Appendix B.

Investigating Chain of Thoughts Prompting: Recent work has demonstrated the success of learning from instructions (Wei et al., 2021; Wang et al., 2022; Mishra et al., 2022d,c; Lu et al., 2022; Parmar et al., 2022; Mishra and Nouri, 2022; Luo et al., 2022) and chain of thought (Wei et al., 2022) and scratchpad prompting (Nye et al., 2021). To test this on FeasibilityQA, we add explanations to one-shot and few-shot examples provided in the context. Table 8 shows marginal improvement. More details are in Appendix C.

A Case Study on Prompting GPT-3 to Provide Explanation: We further investigate the reason behind GPT-3's poor performance on FeasibilityQA by prompting it to provide the reason behind its answer. Specifically, we add *"Explain the reason behind your answer"* in the prompt. Figure 2 illustrates a response from GPT-3. The answer demonstrates that it did not understand the numerical value of Abraham's age. We also provide additional knowledge to assist the model, as shown in Figure 2. Even with knowledge, the model could not understand the feasible age.

4 Conclusion

In this work, we proposed FeasibilityQA, a question-answering dataset that evaluates the understanding of feasibility. We conducted extensive experiments with several state-of-the-art models in zero-shot, one-shot, and few-shot settings and show that these models struggle to answer the feasibility questions correctly. We also experimented by providing additional knowledge (relevant to the question) and showed that it leads to a small gain in performance, but the overall performance still remains low. We further analyzed the performance of models that reveals several interesting findings. Finally, we release our dataset and hope that our work will encourage further research in feasibility reasoning, an important yet underexplored aspect of commonsense reasoning.

Limitations

Like any other commonsense reasoning ability, the concept of feasibility, in general, can be applied in numerous real-world situations. In our dataset, we try to cover a diverse set of such situations that test the understanding of feasibility, but it is in not an exhaustive list. In the future, we will expand the category space by either converting existing numerical datasets into feasibility questions or manually creating new category spaces. Along with the dataset, we release the list of contexts and situations on which the question is based. In the future, this would help expand the dataset to cover other domains and situations. The human evaluation of the dataset could also be an interesting study, but it can be an expensive. The selection of humans in terms of their educational background and age is also required for unbiased evaluation. A completely random selection of humans is also required for a comprehensive study. Finally, our dataset includes questions in only one language, i.e., English.

Ethical Considerations

The names used in this dataset are selected from the most common English names. In question creation, we ensure that all our contexts and questions describe realistic situations, e.g., we keep a range of numerical entities and units appropriate for their context. No personal information from data creators has been collected during the creation of the dataset.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Pratyay Banerjee, Chitta Baral, Man Luo, Arindam Mitra, Kuntal Pal, Tran C Son, and Neeraj Varshney. 2020. Can transformers reason about effects of actions? *arXiv preprint arXiv:2012.09938*.

Pratyay Banerjee, Swaroop Mishra, Kuntal Kumar Pal, Arindam Mitra, and Chitta Baral. 2021. Commonsense reasoning with implicit knowledge in natural language. In *3rd Conference on Automated Knowledge Base Construction*.

Pratyay Banerjee, Kuntal Kumar Pal, Arindam Mitra, and Chitta Baral. 2019. Careful selection of knowledge to solve open book question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6120–6129, Florence, Italy. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, and Diana Inkpen. 2018. Natural language inference with external knowledge.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2022. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *North American Chapter of the Association for Computational Linguistics*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Yann LeCun. 2022. A path towards autonomous machine intelligence. *preprint posted on openreview*.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd Schema Challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, pages 552–561. AAAI Press, Rome, Italy.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online. Association for Computational Linguistics.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and A. Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *ArXiv*, abs/2209.09513.

Man Luo, Sharad Saxena, Swaroop Mishra, Mihir Parmar, and Chitta Baral. 2022. Biotabqa: Instruction learning for biomedical table question answering. *arXiv preprint arXiv:2207.02419*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Conference on Empirical Methods in Natural Language Processing*.

Swaroop Mishra, Anjana Arunkumar, Chris Bryan, and Chitta Baral. 2022a. Hardness of samples need to be quantified for a reliable evaluation system: Exploring potential opportunities with a new task. *ArXiv*, abs/2210.07631.

Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Oyvind Tafjord, Ashish Sabharwal, Peter Clark, and A. Kalyan. 2022b. Lila: A unified benchmark for mathematical reasoning. *ArXiv*, abs/2210.17517.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022c. Reframing instructional prompts to GPTk's language. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 589–612, Dublin, Ireland. Association for Computational Linguistics.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022d. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3470–3487, Dublin, Ireland. Association for Computational Linguistics.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. 2022e. NumGLUE: A suite of fundamental yet challenging mathematical reasoning tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3505–3523, Dublin, Ireland. Association for Computational Linguistics.

Swaroop Mishra and Elnaz Nouri. 2022. Help me think: A simple prompting strategy for non-experts to create customized content with models. *arXiv preprint arXiv:2208.08232*.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

Kuntal Kumar Pal and Chitta Baral. 2021. Investigating numeracy learning ability of a text-to-text transfer model. In *Conference on Empirical Methods in Natural Language Processing*.

Mihir Parmar, Swaroop Mishra, Mirali Purohit, Man Luo, M. Hassan Murad, and Chitta Baral. 2022. Inboxbart: Get instructions into biomedical multi-task learning. In *NAACL-HLT*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Abhilasha Ravichander, Aakanksha Naik, Carolyn Rose, and Eduard Hovy. 2019. EQUATE: A benchmark evaluation framework for quantitative reasoning in natural language inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 349–361, Hong Kong, China. Association for Computational Linguistics.

Pedro Rodriguez, Joe Barrow, Alexander Miserlis Hoyle, John P. Lalor, Robin Jia, and Jordan Boyd-Graber. 2021. Evaluation examples are not equally informative: How should that change NLP leaderboards? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4486–4503, Online. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Shikhar Singh, Nuan Wen, Yu Hou, Pegah Alipoormolabashi, Te-lin Wu, Xuezhe Ma, and Nanyun Peng. 2021. COM2SENSE: A commonsense reasoning benchmark with complementary sentences. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 883–898, Online. Association for Computational Linguistics.

Oyvind Tafjord, Peter Clark, Matt Gardner, Wen-tau Yih, and Ashish Sabharwal. 2019a. Quarel: A dataset and models for answering questions about qualitative relationships. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7063–7071.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019b. QuaRTz: An open-domain dataset of qualitative relationship questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5941–5946, Hong Kong, China. Association for Computational Linguistics.

Neeraj Varshney, Man Luo, and Chitta Baral. 2022a. Can open-domain qa reader utilize external knowledge efficiently like humans? *arXiv preprint arXiv:2211.12707*.

Neeraj Varshney, Swaroop Mishra, and Chitta Baral. 2022b. ILDAE: Instance-level difficulty analysis of evaluation data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3412–3425, Dublin, Ireland. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *EMNLP*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *ICLR*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete KBs with knowledge-aware reader. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4258–4264, Florence, Italy. Association for Computational Linguistics.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4889–4896, Online. Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

Ruiqi Zhong, Dhruba Ghosh, Dan Klein, and Jacob Steinhardt. 2021. Are larger pretrained language models uniformly better? comparing performance at the instance level. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3813–3827, Online. Association for Computational Linguistics.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

Appendix

A Dataset characteristics

In this section, we describe FeasibilityQA in more detail. Table 1 shows illustrative examples of each category discussed in section 4. Each row of the 4th Column of the table shows three questions that were prepared in response to a context. Table 2 gives the distribution of each category of the dataset. We will explain the motivation behind each category. Please note that the explanations are with respect to examples presented in Table 1.

Attribute Comparison shows the comparative properties between two similar objects. The context from attribute comparison is designed to show that quantities can be measured using words like higher and lower, and the model has to understand the relation between them to answer different questions. In this example, it is not possible that the smaller bottle can have a volume of 33 units since the larger one is 32 units.

Change with Time gives the events that have time as the changing factor. The context is designed to test the model's ability to deduce time-based changes and how certain actions/ events/ quantities can or cannot be done before/ after a certain time. In this case, it is impossible that Edward's age could be 19 on his last birthday as his current age is 16.

Change with Action describes the actions which alter certain quantities/events and test the model's ability to understand that. In this case, it is demonstrated that selling/giving away a certain quantity reduces it. In the example, it is demonstrated that selling all 273 items at least 1 dollar will leave Joshua with at least 273 dollars. Hence the question that he could have 260 dollars is false.

Implicit Numerical Knowledge tests the model's ability to understand numerical entities as facts and how to manipulate them in different situations. In this case, using the knowledge (or without using it) that four quarters make 1 dollar, the model needs to understand how many quarters will be used in 12 dollars, which is 48 quarters. Hence the question tells us that Christopher can have 52 quarters.

Non Numerical category tests the model's understanding of very broad domains. They do not have to be numerical based in all the cases.

The dataset contains diverse topics ranging from physics, mathematics, biology, and numerical reasoning. A total of 422 subcategories are present in the dataset. Table 2 shows the distribution of BCQ and MCQ questions across different categories in the dataset.

B Other performance results

B.1 Performance Metrics

For the MCQ setting of the dataset, we study true positive, false positive, and false negative rates as the evaluation metrics. Each metric definition is listed below:

False Negative rate is defined as the number of

	w/o Knowledge	w/ Knowledge
Zero	0.17	0.13
One	0.36	0.32
Few	0.33	0.24

Table 9: False Positive rate of GPT-3 on MCQ section

	w/o Knowledge	w/ Knowledge
Zero	0.42	0.42
One	0.21	0.24
Few	0.18	0.20

Table 10: False negative rate of GPT-3 on MCQ section

Zero shot BCQ		
	w/o Knowledge	w/ Knowledge
Attribute Comparison	51.2	55.8
Non Numerical	72.7	85.7
Implicit Numerical	52.9	52.0
Change with Action	60.7	65.3
Change with Time	55.5	55.5
Zero shot MCQ		
	w/o Knowledge	w/ Knowledge
Attribute Comparison	17.9	20.4
Non Numerical	25.5	29.0
Implicit Numerical	18.5	21.1
Change with Action	21.3	24.2
Change with Time	19.4	22.2

Table 11: Category wise Accuracy of GPT-3 on BCQ and MCQ task in zero-shot setting.

incorrect predictions the model gave as correct. For example, if the model gave output as A, B, C, and the predicted result is A, C, then B is missed. The number of false negatives would be 1 (B).

False Positive rate is defined as the number of correct predictions the model gave as incorrect. For example, if the given output is A, B and the predicted result is A, B, C, then the number of false negatives would be 1 (C).

B.2 Results

False positive results shown in Table 9 follow trends similar to accuracy where the performance of one-shot experiments is worse than zero-shot and few-shot. But with the addition of knowledge, the false positive rate decreases.

As shown in Table 10, the False negative rate decreases from zero-shot to few-shot experiments, but contrary to other experiments, it increases with the addition of knowledge in almost all the cases.

Table 11 shows the category-wise results in zero-shot settings for BCQ and MCQ tasks. For the BCQ task, accuracy was lowest in the Attribute comparison category and highest in Non-Numerical Category. Performance of the Non Numerical category improved significantly in with knowledge setting.

In the MCQ portion of the dataset, the performance gap between Non-Numerical and other categories reduces significantly. It is still the best-performing category for the model, but the Change with Action Category also produced similar results. There was no significant improvement in both Non-Numerical and change with action as observed in the Non-Numerical with the addition of knowledge.

Few shot BCQ		
	w/o Knowledge	w/ Knowledge
Attribute Comparison	64.5	69.5
Non Numerical	85.9	99.4
Implicit Numerical	60.8	56.6
Change with Action	73.7	86.8
Change with Time	64.8	74.1
Few shot MCQ		
	w/o Knowledge	w/ Knowledge
Attribute Comparison	25.2	35.8
Non Numerical	33.5	47.6
Implicit Numerical	23.7	33.7
Change with Action	28.8	40.9
Change with Time	25.3	35.9

Table 12: Category wise Accuracy of GPT-3 on BC and MCQ task in few-shot setting.

Table 12 shows the category-wise results for BCQ and MCQ tasks in few shot setting. For the BCQ task, accuracy was lowest in the Attribute comparison category and highest in Non-Numerical Category. Performance of the Non Numerical category improved significantly in the knowledge setting with accuracy reaching above 90% for the first time in any of the categories.

In the MCQ portion of the dataset, the performance gap between Non-Numerical and other categories reduces significantly. It is still the best-performing category for the model. There was a significant improvement in Non Numerical and change with action and change with time categories with the addition of knowledge.

Exact 50% accuracy of GPT-2: The input format for all models was as follows: Zero-Shot, Question (Different questions), and Options (True or False). Example(s) preceded the question in the one-shot and few-shot settings. Based on this format, GPT-2 gave the probability of "False" higher in all cases. Since the dataset is label balanced, all the True hypothesis questions were incorrectly predicted, hence giving a 50% accuracy.

Example or Evaluation sample		Text / Context
Example fed in GPT-3 as 1 shot learning example	**Prompt**	Select the best option for the questions based on the Context and explain the reasoning. An Example to show how it works:
	Context	Eric boiled a glass of water to drink the hot water. The level of the water in the glass is 63 units. After boiling was done, she took the water into the glass.
	Question	What could be the level of the water after boiling?
	Options	A: 65, **B: 59**, C: 64, **D: 61**, E: None
	Answer:	Since boiling leads to evaporation, there is a possibility that the volume of water would have decreased. Due to this, the current water level should be less than 63 units. Hence the answer is B:59 and D: 61
Example given to GPT-3 for evaluation	**Context**	A negative number was added to 70.
	Question	What could be the result?
	Options	A: 70, B: 89, C: 87, **D: 68**, E: None
	GPT-3 Answer	The result could be A: 70, since **adding a negative number would not change the value of 70**. The result could also be C: 87, since adding a negative number would increase the value of 70
Example given to GPT-3 for evaluation	**Context**	Last year 758 people joined the company and this vear the number of people who joined was more than double the last time
	Question	How many people could have joined this year?
	Options	A: 1440, **B: 1600**, C: 1460, D: 1370, E: None
	GPT-3 Answer:	Since the number of people who joined this year was **more than double the last time**, the number of people who joined this year could be up to 1460. Therefore, the answer is C: 1460.

Table 13: Illustrating chain of thought approach on some examples of feasibilityQA dataset in 1 shot setting without providing knowledge. 1st set of rows demonstrate the example fed into GPT-3 for 1 shot learning. 2nd and 3rd set of rows show GPT-3's response to Context, Question and Options asked.

Example or Evaluation sample		Text / Context
Example fed in GPT-3 as 1 shot learning example	**Prompt**	Select the best option for the questions based on the context, knowledge and explain the reasoning. An Example to show how it works:
	Context	Eric boiled a glass of water to drink the hot water. The level of the water in the glass is 63 units. After boiling was done, she took the water into the glass.
	Knowledge	Water will convert into water vapour on boiling.
	Question	What could be the level of the water after boiling?
	Options	A: 65, B: 59, C: 64, D: 61
	Answer	Since boiling leads to evaporation, there is a possibility that the volume of water would have decreased. Due to this, the current water level should be less than 63 units. Hence the answer is B:59 and D: 61
Example given to GPT-3 for evaluation	**Context**	Two boys competed in a race. The loser finished the race in 24 minutes.
	Knowledge	Loser will take more time to finish the race.
	Question	How many minutes the other boy could have taken to finish the race?
	Options	A: 32, B: 25, C: 15, D: 22, E: None
	GPT-3 Answer	The other boy **could have taken less time to finish the race**. Hence, the answer is B: 25.

Table 14: Illustrating chain of thought approach on some examples of feasibilityQA dataset in 1 shot setting with providing knowledge. 1st set of rows demonstrate the example fed into GPT-3 for 1 shot learning. 2nd row shows GPT-3's response to Context, Question and Options asked.

C Case study: Chain of Thought Reasoning Approach

Table 13 and 14 show the unsuccessful attempts in the chain of thought reasoning approach. Table 13 shows the setting where the 1st example is fed into the model as an example of how to reason out the answer. The reason and answer were clearly mentioned that told that evaporation leads to a decrease in water level and hence water level should decrease. This led to a decrease in water level; hence, the correct answers were quantities less than 63; **59 and 61**.

The 2nd and 3rd sets of rows show the Context, question, and options supplied to GPT-3, and we get responses in GPT-3 Answer row. The logic given for the addition of a number is wrong.

Adding a negative number should decrease the value, and hence rest of the answer will be wrong. In the 3rd row GPT-3's response, the logic used to calculate the answer is correct, but it was unable to calculate that 1600 was double 758. Both parts are highlighted in the table.

The situation did not improve much when knowledge was supplied with other rows, as shown in Table 14. The model was able to interpret the logic correctly but could not associate that logic with numerical quantities.

Efficient Encoders for Streaming Sequence Tagging

Ayush Kaushal♣* **Aditya Gupta**♠ **Shyam Upadhyay**♠ **Manaal Faruqui**♠
♠Google Assistant
♣The University of Texas at Austin
ayushk4@utexas.edu, {gaditya, shyamupa, mfaruqui}@google.com

Abstract

A naive application of state-of-the-art bidirectional encoders for streaming sequence tagging would require re-encoding all tokens from scratch whenever a new token appears in an incremental streaming input (like transcribed speech). The lack of re-usability of previous computation leads to a higher number of Floating Point Operations (or FLOPs) and higher number of unnecessary label flips. Increased FLOPs consequently lead to higher wall-clock time and increased label flipping leads to poorer streaming performance. In this work, we present **H**ybrid **E**ncoder with **A**daptive **R**estart (HEAR) that addresses these issues while maintaining the performance of bidirectional encoders over offline (or complete) inputs and improving performance on streaming (or incomplete) inputs. HEAR uses a HYBRID unidirectional-bidirectional encoder architecture to perform sequence tagging, along with an Adaptive Restart Module (ARM) to selectively guide the restart of bidirectional portion of the encoder. Across four sequence tagging tasks, HEAR offers FLOPs savings in streaming settings upto 71.1% and also outperforms bidirectional encoders for streaming predictions by upto +10% streaming exact match.

1 Introduction

State-of-the-art text encoding methods assume the *offline setting*, where the entire input text is available when encoding it. This differs from the *streaming setting* where the input text grows over time (such as transcribed speech or a typed query) (Cho and Esipova, 2016; Gu et al., 2017; Chang et al., 2022). Processing streaming input incrementally can enable suggestions on partial inputs (Iranzo Sanchez et al., 2022), reduced final latency (Zhou et al., 2022) and lead to more interactive NLU agents (Cai and de Rijke, 2016).

Existing state-of-the-art bidirectional encoders (such as Devlin et al. (2019)) are computationally expensive for streaming processing. When a new token is received, these models require a *restart*, i.e., re-computation of representation of each token by re-running the bidirectional layer (Kahardipraja et al., 2021). This adds to the computational cost (i.e., Higher FLOPs) during streaming and leads to higher wall clock time. Another limitation of these encoders is poorer generalization to partial inputs (Madureira and Schlangen, 2020), stemming from these models being trained only on complete (and offline) inputs. Despite these disadvantages, such bidirectional models offer better *offline* performance than unidirectional models across several NLP tasks like sequence tagging (Kahardipraja et al., 2021).

Model	GFLOPs	Offline F1	Streaming EM
BiDi encoder	74.7	**93.1**	75.1
HYBRID encoder	43.4	93.0	72.5
HEAR (HYBRID + ARM)	**21.6**	93.0	**85.1**

Table 1: Computation cost (**G**iga **FLOPs**), **Offline F1** and **Streaming E**xact **M**atch accuracy of standard bidirectional (BiDi) encoder, HYBRID encoder model and HEAR (HYBRID encoder with ARM for guiding Adaptive Restarts) on SNIPS test set.

We address these issues by proposing HEAR – **H**ybrid **E**ncoder with **A**daptive **R**estarts, where a separate Adaptive Restart Module (ARM) predicts when to restart the encoder. The encoder in HEAR is a HYBRID encoder that reduces the computational cost of running the models in streaming settings. In a HYBRID encoder, the earlier layers are unidirectional and the deeper layers are bidirectional. This design allows early contextualization, and limits the need for restart to the later layers. While HYBRID encoder reduce streaming computational overhead, restarting them at every new token may not be required. Thus, we propose a lightweight Adaptive Restart Module (ARM) to guide restarts, by predicting whether restarting the

---

[*] Work done as part of an internship at Google.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 418–429
May 2-6, 2023

bidirectional layers of the HYBRID encoder will benefit the model performance. This module is also compatible with fully bidirectional encoders.

Table 1 showcases the strength of HEAR on one of the sequence tagging dataset we consider. In terms of streaming computation, measured in terms of FLOPs (lower is better), HYBRID encoder offers significant saving from purely bidirectional (BiDi) encoders and FLOPs savings improve upon incorporating ARM in HEAR. In terms of offline performance, HYBRID encoder and HEAR achieves parity with BiDi encoders and in terms of streaming performance, HEAR significantly outperforms BiDi encoders.

Following are our key contributions:

- We introduce HYBRID encoder for computationally cheaper streaming processing (§3.1), that maintains the offline F1 score of bidirectional encoders, while reducing FLOPs by an average of 40.2% across four tasks.
- We propose the ARM module (§3.2) to decide when to restart. The ARM reduces FLOP of HYBRID encoder by 32.3% and improves streaming predictions by +4.23 Exact Match.
- Our best model HEAR combines HYBRID encoder with ARM (§5) to achieve strong streaming performance while saving FLOPs and offering competitive offline performance across four sequence tagging tasks.

2 Streaming Sequence Tagging

In the streaming sequence tagging task, we assume that at time $1 \leq t \leq n$, we have received the first t tokens $\mathbf{x}_t = (x_1, \cdots x_t)$ as input from a stream of transcribed speech or user-typed input.[1] The model then predicts the tags for all t tokens $\hat{\mathbf{y}}_t = (\hat{y}_{1,t}, \hat{y}_{2,t} \cdots \hat{y}_{t,t})$. The models predictions $\hat{\mathbf{y}}_n$ over the offline input $\mathbf{x} = \mathbf{x}_n$ is the offline tag sequence prediction which is evaluated against the ground truth $\mathbf{y}^*_n = (y^*_1, y^*_2 \cdots y^*_n)$. However, in the streaming settings, we are concerned with predicted label sequences over all of the prefixes $\hat{\mathbf{Y}} = (\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2 \cdots \hat{\mathbf{y}}_n)$.

During training we only have access to the offline ground-truth label sequence over the offline input sequence $\mathbf{y}^*_n$, even though ground truth labels for tokens may change as additional context is received in the future timesteps.

3 Streaming Sequence Tagging with HEAR

In order to motivate HEAR, consider how existing BiDi encoders would be used in streaming sequence tagging. At time t and the input sequence $\mathbf{x}_t$, the model restarts to predicts the label sequence $\hat{\mathbf{y}}_t$ i.e. it re-computes all its layers for all tokens, without leveraging any of its previous computation or any previously predicted labels. Consequently, running a typical $O(n^2)$ encoder would have $O(n^3)$ computations in streaming settings, over n input tokens. Therefore, naively using existing BiDi encoders for streaming settings is highly inefficient. Previous works tackle inefficiency by modifying the streaming model to infer only once for each word, after k future tokens have been received (Oda et al., 2015; Grissom II et al., 2014). This leads to a k-delayed output with possibility for revisions after additional future words have been received. However, this may lead to poor performance for tasks involving long-range dependency (e.g., SRL) and a higher output lag.

HEAR is a system consisting of a trained HYBRID encoder model and an ARM that is trained over the HYBRID encoder. The early unidirectional layers in HYBRID encoder, reduces the computational cost of restart of the encoder. The ARM guides when to restart the HYBRID encoder. It saves computational cost by keeping the model from restarting at each timestep and also improves streaming performance by avoiding unnecessary label flips stemming from avoidable restarts. Figure 2 shows running of HEAR in streaming settings over an example. For each new token in the streaming input, all the unidirectional layers and a part of first bidirectional layer is ran for the token, to obtain its unidirectional encodings and updated cache. These are then used by ARM to predict whether to restart the bidirectional layer or not. If the bidirectional layer is to be restarted, then we obtain updated labels for all the tokens received in the stream. Otherwise, the auxiliary predictor is ran over the unidirectional encoder for the current token to obtain its label and the labels from previous timestep is copied for all other tokens.

We now formally introduce HYBRID encoder (§3.1), followed by ARM (§3.2).

---

[1] We assume that new tokens get added without changing previous tokens (contrary to some ASR systems), even though, our method can be used in such settings.

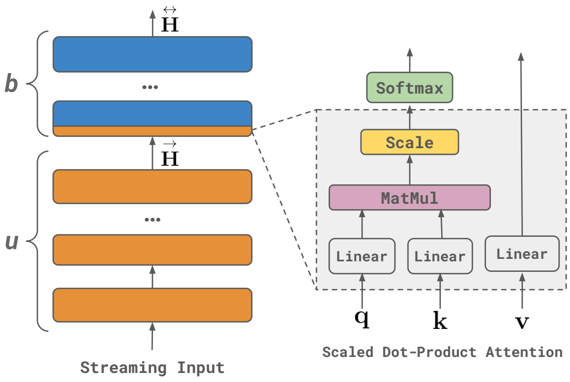


Figure 1: In HYBRID encoder architecture, the earlier u layers are unidirectional, and later b layers are bidirectional. The blue layers require restart, i.e. at each timestep when the model receives a new token of the streaming input, these layers will recompute all intermediate representations for all the input tokens. The orange layers (like unidirectional layers) can avoid restart by caching the intermediate state. A portion of the first bidirectional layer before application of softmax also does not require restart and can be cached.

3.1 HYBRID Encoder

The HYBRID encoder is a combination of unidirectional and bidirectional encoding layers, where the early layers are unidirectional and the later layers are bidirectional as shown in Figure 2. The earlier unidirectional layers do not require a restart as the previous tokens' embeddings do not need to be updated for the newly received tokens. Thus, the HYBRID encoder only require a restart for its later bidirectional layers. Formally, the HYBRID encoder has u unidirectional and b bidirectional layers where total layers in the model are $u + b = l$. Let $\overrightarrow{L}$ and $\overleftrightarrow{L}$ denote unidirectional and bidirectional layers in the model respectively. Each of the $\overrightarrow{L}$ layers are one of the existing unidirectional layers - RNN, GRU, transformer with causal masking etc. with the first of these being the static embedding layer. Each of the $\overleftrightarrow{L}$ layers are bidirectional layers, being one of bi-GRU, bi-LSTM, transformer etc.

3.1.1 HYBRID Encoder in Offline Setting:

First, the input tokens $\mathbf{x}$ are fed to u unidirectional layers to get the unidirectional encodings $\overrightarrow{\mathbf{H}}$,

$$\overrightarrow{\mathbf{H}} = [\overrightarrow{\mathbf{h}}_1, \cdots \overrightarrow{\mathbf{h}}_n] = \overbrace{\overrightarrow{L} \circ \overrightarrow{L} \circ \cdots \overrightarrow{L}}^{u \text{ layers}}(\mathbf{x}) \tag{1}$$

where $\circ$ denotes the functional composition and $\overrightarrow{\mathbf{h}}_i$ is the unidirectional encoding for the token x_i.

Notation	Meaning
$\overrightarrow{L}$, $\overleftrightarrow{L}$	Unidirectional and bidirectional layers, respectively.
$\overrightarrow{\mathbf{h}}_i$, $\overleftrightarrow{\mathbf{h}}_{i,t}$	Encoding of x_i token when t tokens have been received, using $\overrightarrow{L}$ and $\overrightarrow{L} + \overleftrightarrow{L}$, respectively.
$\overrightarrow{\mathbf{H}}_t$, $\overleftrightarrow{\mathbf{H}}_t$	$\overrightarrow{\mathbf{h}}_i$, $\overleftrightarrow{\mathbf{h}}_{i,t}$ of first t tokens, respectively.
$\overrightarrow{y}_i$, $\overleftrightarrow{y}_{i,t}$	Predicted label of x_i token using $\overrightarrow{\mathbf{h}}_i$ and $\overleftrightarrow{\mathbf{h}}_{i,t}$, respectively.

Table 2: Notations used in HYBRID encoder for layers, token representations, and predicted labels.

Then the bidirectional encodings $\overleftrightarrow{\mathbf{H}}$ are computed and used to predict the label sequence:

$$\overleftrightarrow{\mathbf{H}} = [\overleftrightarrow{\mathbf{h}}_1, \cdots \overleftrightarrow{\mathbf{h}}_n] = \overbrace{\overleftrightarrow{L} \circ \overleftrightarrow{L} \circ \cdots \overleftrightarrow{L}}^{b \text{ layers}}(\overrightarrow{\mathbf{H}}) \tag{2}$$

The final offline labels are obtained by passing $\overleftrightarrow{\mathbf{H}}$ through a feed-forward neural network layer.

3.1.2 HYBRID Encoder in Streaming Setting

In the streaming setting, the unidirectional layers' computation can be cached.[2] These cached intermediate representations for the previously received tokens $\mathbf{x}_{t-1}$ are used in computing the unidirectional encoding $\overrightarrow{\mathbf{h}}_t$ of the new token x_t. This along with the cached unidirectional encoding of $\mathbf{x}_{t-1}$ gives us $\overrightarrow{\mathbf{H}}_t = [\overrightarrow{\mathbf{H}}_{t-1}; \overrightarrow{\mathbf{h}}_t]$. The bidirectional encoding, however, for each token in $\mathbf{x}_t$ is restarted at time t as $\overleftrightarrow{\mathbf{H}}_t = [\overleftrightarrow{\mathbf{h}}_{1,t}, \cdots \overleftrightarrow{\mathbf{h}}_{t,t}]$ using the obtained final unidirectional encoding $\overrightarrow{\mathbf{H}}_t$ as input.

3.1.3 Training and Inference of HYBRID Encoder

We predict labels over both $\overrightarrow{\mathbf{H}}_t$ and $\overleftrightarrow{\mathbf{H}}_t$ of the HYBRID encoder using a linear layer with softmax at each timestep. Let $\overleftrightarrow{\mathbf{y}}_t = [\overleftrightarrow{y}_{1,t}, \cdots \overleftrightarrow{y}_{t,t}]$ denote the predictions over bidirectional embeddings at time t for all the tokens received so far. Let $\overrightarrow{\mathbf{y}}_t = [\overrightarrow{y}_1, \cdots \overrightarrow{y}_t]$ be predictions over unidirectional embeddings at time t. While the unidirectional predictions do not perform as well as bidirectional predictions, these auxiliary predictions enables waited restarts (§3.2).

---

[2] Cache for RNNs and causally masked transformer consists of hidden states and keys-values respectively. This is similar to implementations of auto-regressive models (Wolf et al., 2020; Heek et al., 2020).

HYBRID encoder is trained over the offline input-output sequence pairs and optimize both the bidirectional predictions and unidirectional predictions. for the standard softmax cross entropy loss against the offline ground truth label sequence $\mathbf{y}^*$. In order to preserve the strong performance of the BiDi encoders, we inhibit backward flow of gradient from the parameters θ_{uni} of unidirectional prediction head (consisting only of a linear layer with softmax) to the remaining parameters - θ_{bi}.Following are the losses of these two portions of the model, optimized together with equal weight.[3]

$$\mathcal{L}(\theta_{bi}, \overleftrightarrow{\mathbf{y}}_n, \mathbf{y}^*) = CE(\overleftrightarrow{\mathbf{y}}_t, \mathbf{y}^*) \tag{3}$$

$$\mathcal{L}(\theta_{uni}, \overrightarrow{\mathbf{y}}_n, \mathbf{y}^*) = CE(\overrightarrow{\mathbf{y}}_t, \mathbf{y}^*) \tag{4}$$

$$\mathcal{L}(\theta) = \mathcal{L}(\theta_{bi}, \overleftrightarrow{\mathbf{y}}_n, \mathbf{y}^*) + \mathcal{L}(\theta_{uni}, \overrightarrow{\mathbf{y}}_n, \mathbf{y}^*) \tag{5}$$

3.2 Waited Restarts

The HYBRID encoder reduces the computational overhead of each *restart* by limiting the recomputation over previous tokens only to the later bidirectional layers. However, restarting at each step is not required and the auxiliary predictions from the unidirectional layers can suffice.

We define a variable $\text{RESTART}(t)$ to decide whether to restart the bidirectional layers at time t. When $\text{RESTART}(t) = 1$, we restart the model to get the bidirectional predictions $\overleftrightarrow{\mathbf{y}}_t$ as the final predicted label sequence $\hat{\mathbf{y}}_t = \overleftrightarrow{\mathbf{y}}_t$. If $\text{RESTART}(t) = 0$, the model does not run the bidirectional encoder, but uses the unidirectional encoding to predict the current token's label $\overrightarrow{y}_t$ and copy the label sequence from the previous step $\hat{\mathbf{y}}_{t-1}$ to obtain $\hat{\mathbf{y}}_t$. Formally, the predicted sequence $\hat{\mathbf{y}}_t = (\hat{y}_{1,t}, \cdots \hat{y}_{t,t})$ at time t is

$$\hat{\mathbf{y}}_t = \begin{cases} \overleftrightarrow{\mathbf{y}}_t & \text{if } \text{RESTART}(t) = 1 \text{ or } t = n \\ [\hat{\mathbf{y}}_{t-1}; \overrightarrow{y}_t] & \text{otherwise} \end{cases} \tag{6}$$

where ';' denotes the concatenation. We always restart in the final timestep (i.e., $\text{RESTART}(n) = 1$) to preserve the offline performance.

3.2.1 Adaptive Restart Module (ARM)

A baseline restart strategy for waited restarts would be to restart the bidirectional layers every fixed k steps, i.e., $\text{RESTART}(t) = 1$ whenever t is a multiple of k. Note that $k = 1$ reduces to a HYBRID encoder model without waited restarts. We refer to this as **RESTART-k**.

Rather than having a heuristic function for $\text{RESTART}(t)$, we can let a lightweight parametric module determine when to do the restart by determining $\text{RESTART}_\theta(t)$ for each t. We refer to this as **A**daptive **R**estart **M**odule (**ARM**). We first discuss the set of features for ARM (§3.2.2), followed by its architecture (§3.2.3) and training (§3.2.4).

3.2.2 Input Features for ARM

We use HYBRID encoder's intermediate representations that do not require restart as the features for ARM. This includes the unidirectional layers as well as from the pre-softmax features from the first $\overleftrightarrow{L}$ layer. The main motivation to use these features is to incorporate more information without incurring any restarts.

The following features are stored in unidirectional cache and used without any restart for ARM:

(i) *unidirectional encodings* $\overrightarrow{\mathbf{h}}_t$ from the last unidirectional layer computed once and cached for capturing the backward flow,

(ii) *query* $\mathbf{q}_t$ and *key* $\mathbf{k}_t$ from the first bidirectional layer computed only from $\overrightarrow{\mathbf{h}}_t$,

(iii) *unnormalized forward attention scores* $\mathbf{a}_t = [\mathbf{q}_1 \cdot \mathbf{k}_t, \cdots, \mathbf{q}_{t-1} \cdot \mathbf{k}_t]$ from the first bidirectional layer, where '$\cdot$' is the dot product. These scores are concatenated across the prefix tokens (restricted and padded to latest m tokens) across all heads.

These features are then concatenated and provided as input to the ARM as $\mathbf{f}_t = [\overrightarrow{\mathbf{h}}_t, \mathbf{q}_t, \mathbf{k}_t, \mathbf{a}_t]$.

3.2.3 ARM Architecture

We use a single-layered GRU (Cho et al., 2014) to predict restart. The probability of restart $p_\theta(\text{RESTART}(t) = 1)$ is modeled as,

$$(\mathbf{o}_t, \mathbf{s}_t) = \text{GRU}(\mathbf{f}_t, \mathbf{s}_{t-1}) \tag{7}$$

$$p_\theta(\text{RESTART}(t) = 1) = \sigma(\text{LINEAR}(\mathbf{o}_t)) \tag{8}$$

Here $\mathbf{s}_t$ and $\mathbf{o}_t$ are the GRU hidden state and output at time t, respectively. These outputs can be post-processed to handle too frequent or too infrequent waiting and obtain the final selector predictions.Details can be found in Appendix (§A.2).

---

[3] CE is the Cross Entropy Loss

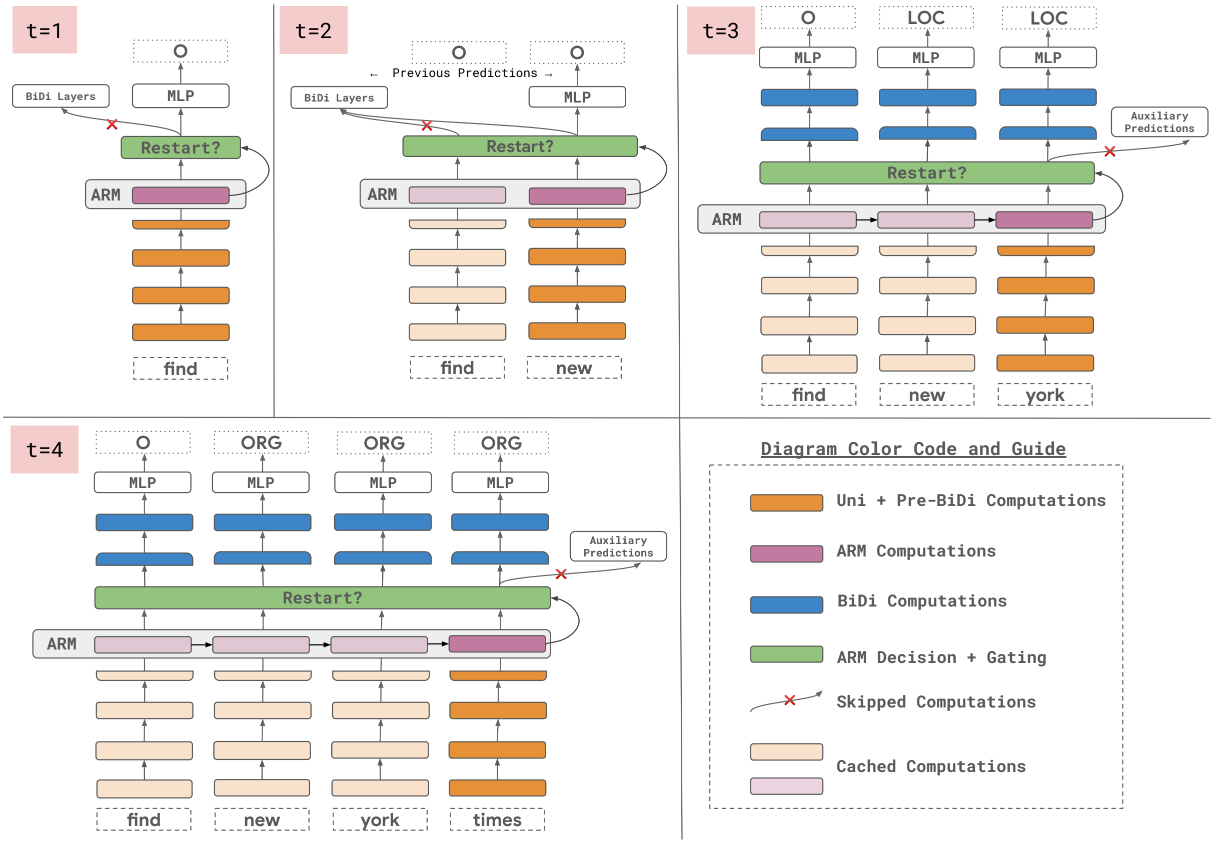


Figure 2: The HEAR model in streaming setting. At $t = 1$, ARM decides not to run the bidirectional layers and exit early through the auxiliary predictor over unidirectional encoding. Similarly at $t = 2$, ARM decides not to restart and considers the auxiliary predictions of the latest token and the predictions from the previous steps. At $t = 3$ and $t = 4$, ARM decides to restart and runs the bidirectional layers to get prediction for all the tokens from scratch. Note that in all the steps, for unidirectional layers and ARM, the computation is performed only for the latest token received.

3.2.4 Training the ARM

Ideally, HYBRID encoder should restart only when it would lead to improved predictions over the prefixes i.e., more number of prefix inputs should have the same output as the the offline ground truth for those prefixes. We use this to define our ground truth RESTART sequence for training the ARM.

Given the ground truth offline tag sequence $\mathbf{y}^*$, unidirectional predictions $\vec{\mathbf{y}} = [\vec{y}_1, \cdots \vec{y}_n]$ and final bidirectional prediction sequence over each of the n prefixes $\overleftrightarrow{\mathbf{Y}} = [\overleftrightarrow{\mathbf{y}}_1, \cdots \overleftrightarrow{\mathbf{y}}_n]$, we define the ground truth policy RESTART$^*(t) = \pi_t^*$ for ARM, at time t, as follows.

$$\pi_t^* = \begin{cases} 1 & \text{if } |y^*_{\leq t} = \vec{y}_{\leq t}| < |y^*_{\leq t} = \overleftrightarrow{y}_{\leq t}| \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

The policy decides to restart at time t if more tokens in bidirectional predictions match with the ground truth ($|y^*_{\leq t} = \overleftrightarrow{y}_{\leq t}|$) than those from unidirectional predictions ($|y^*_{\leq t} = \vec{y}_{\leq t}|$).[4] ARM is trained against this policy with features from a frozen and trained HYBRID encoder model with a standard binary cross entropy loss.

4 Experimental Setup

Datasets and Tasks. We consider four common sequence tagging tasks – Slot Filling over SNIPS (Coucke et al., 2018), Semantic Role Labelling (SRL) over Ontonotes (Pradhan et al., 2013), and Named Entity Recognition (NER) and Chunking (CHUNK) over CoNLL-2003 (Sang and Meulder, 2003). The standard train, dev and test splits are used for all datasets.

Models and Baselines. For our models, we consider a layer budget of 4 for the encoder. Naturally, our baselines are the UniDi encoder and BiDi encoder with all unidirectional and bidirec-

---

[4] This policy is greedy. The optimal policy obtained via dynamic programming relies on the future tokens.

Current Token	Prefix Predictions	Total Edits	Unnecessary Edits	Exact Match w/ ground truth	Exact Match w/ final prediction
find	`O`	1	0	1	1
new	`O O`	2	1	1	1
york	`O LOC LOC`	4	3	2	1
times	`O ORG ORG ORG`	7	3	2	2
square	`O ORG ORG ORG LOC`	8	3	2	3
Ground Truth	`O LOC LOC LOC LOC`	**EO**= $\frac{\text{Unnecessary Edits}}{\text{Total Edits}} = \frac{3}{8}$		**Streaming EM** = $\frac{\text{EM w/ ground truth}}{\text{\# steps}} = \frac{2}{5}$	**RC** = $\frac{\text{EM w/ final labels}}{\text{\# steps}} = \frac{3}{5}$

Table 3: Example computation for the metrics described in §4.1 . **EO** is calculated as fraction of edits that were *unnecessary*. **RC** measures the fraction of prefix predictions that matches the model's final predictions. **Streaming EM** measures the fraction of prefix predictions that are correct with respect to the ground truth labels.

tional layers, respectively. For HEAR, we tune for the optimal fractions of unidirectional layers in encoder by selecting the one that maximizes the offline F1 performance over the development set. We consider RESTART-k baseline for ARM. For each dataset, we picked the best value of k from $\{2, 3, 5, 8\}$, maximizing the Streaming EM over development set.

4.1 Metrics

The model evaluation is done over three criteria: offline performance over complete input text, streaming performance over prefixes and efficiency of running the model in streaming settings.

Offline Metrics. The offline performance of the model is measured using the widely used chunk-level F1 score for sequence labelling tasks (Sang and Meulder, 2003).

GFLOPs. We measure total number of Floating Point Operations (FLOPs) required for running the model in a streaming setting in GigaFLOPs (**GFLOP**), estimated via XLA compiler's High Level Operations (HLO) (Sabne, 2020). GFLOP positively correlates with how computation-heavy a model is and its wall-clock execution time.

Streaming Metrics. An ideal streaming model should predict the correct labels for all prefixes early (Trinh et al., 2018) and avoid unnecessary label flips. Following previous works (Madureira and Schlangen, 2020; Kahardipraja et al., 2021), we use Edit Overhead (EO) and Relative Correctness (RC) metrics from Baumann et al. (2011).

EO (Edit Overhead) is a measure of the fraction of the label edits that were unnecessary with respect to the final prediction on complete output. Consider the example in Table 3. At the first timestep, the token "*find*" is assigned a label "`O`" from "`N/A`"; taking the total edits to 1. Similarly, in the second timestep, the newly received token "*new*" gets a label edit; taking the total edit to 2. In the third timestep, however, not only the newly received token "*york*" receives a label edit but the second token "*new*" is also edited from "`O`" to "`LOC`"; resulting in 4 total edits. Of these edits, the label edit for token "*new*" in timestep 2 from "`N/A`" to "`O`" was unnecessary as it differs from its final label "`ORG`". Similarly, the label flips for "*new*" and "*york*" in the timestep 3 from "`O`" and "`N/A`" to "`LOC`" and "`LOC`", respectively, were also unnecessary. Cumulatively, towards the end of fifth timestep, we have 3 out of 8 total edits which were *overhead*. Thus, the EO turns out to be $\frac{3}{8}$, where lower is better.

RC measures the relative correctness of prefixes, i.e. correctness of prefix prediction sequence with respect to the final label set over the complete input. For the example in Table 3, only the label sequence in the first timestep ("`O`"), fourth timestep ("`O ORG ORG ORG`") and final (fifth) timestep are prefixes of the label sequence in the final timestep ("`O ORG ORG ORG LOC`"). Thus 3 prefixes of a total 5 prefixes were correct. So, the Relative Correctness is $\frac{3}{5}$, where higher is better.

While EO and RC capture consistency and stability in streaming predictions, neither of these measures performance with respect to the ground truth label over offline input. Relying on these metrics alone is not sufficient to measure streaming performance. For example, a UniDi encoder achieves perfect EO and RC scores, despite have poor predictions with respect to the offline ground truth. Thus, we consider **Streaming** Exact Match (**Streaming EM**), which is the streaming setting analogue of the Exact Match metric. Streaming EM calculates the percentages of prefix label sequence which are correct with respect to the offline ground truth labels. For example in Table 3, only the first label sequence ("`O`") for input "*find*" and third label se-

Model	SNIPS	CHUNK	NER	SRL
UniDi encoder	86.8	88.1	73.8	56.4
HYBRID encoder	93.0	**89.4**	**86.8**	80.0
BiDi encoder	**93.1**	**89.4**	86.0	**80.1**

Table 4: Offline F1 over test set of datasets described in §4. On all datasets, HYBRID encoder achieves performance parity with the BiDi models.

quence (“`O LOC LOC`”) for input “*find new york*” is a prefix of the ground truth label sequence (“`O LOC LOC LOC LOC`”), leading to only 2 out of 5 prefix label sequence being correct. Thus, the Streaming EM is $\frac{2}{5}$. Similar to RC, higher score is better as more prefixes have exact matches.

5 Experimental Results

In this section, we provide empirical results to answer the following questions:

(a) In offline setting, does HYBRID encoder achieve parity with BiDi encoders?

(b) In streaming setting, by how much does HYBRID encoder reduce GFLOPs count?

(c) Does HEAR improve streaming performance and save on GFLOP count?

HYBRID Encoder’s Offline Performance is Competitive to BiDi Encoders. Table 4 shows the offline F1 scores of the HYBRID encoder, UniDi and BiDi encoders across the four tasks. The HYBRID encoder has similar offline F1 as BiDi encoders. In fact, on NER, it even outperforms it by a margin of 0.8 F1 score. As expected, the UniDi model performs poorly compared to the BiDi encoder across all the tasks. From here on, we omit UniDi encoder.

HYBRID Encoder Improves Streaming Efficiency. Table 5 shows the GFLOP (per input instance) in streaming settings across the four datasets for the HYBRID encoder and BiDi encoders, with a trivial restart at every new token to get predictions over streaming text. We observe that HYBRID encoder consistently offers lower GFLOP count than BiDi encoder across the datasets, and in three of the four datasets, offering more than 40% FLOP reduction. However, HYBRID encoder does not improve on streaming performance (Streaming EM, EO, and RC) over BiDi, and its FLOP can be further reduced. We next see how incorporating HEAR addresses this.

HEAR Improves Streaming Efficiency and Performance. Across all datasets, we observe HEAR further reduces GFLOP count from the already computationally lighter HYBRID encoder, giving us upto 71% total FLOP reduction from BiDi. HEAR has better streaming predictions, as its Streaming EM is much higher (upto +10.0) than BiDi, improving on the shortcoming of Hybrid. This highlights that naively restarting at each timestep can worsen the streaming performance, as evident through the lower performance of Hybrid and BiDi. HEAR also offers reduced number of avoidable label flips, as signified by a lower EO. Its high RC score signifies that its streaming predictions are more consistent with its final prediction.

All these results demonstrate that HEAR preserves the offline performance of BiDi encoders, while being computationally lighter by 58.9% on average across tasks. HEAR has more consistent streaming predictions which are accurate with respect to offline ground truth.

6 Analysis and Ablations

In this section, we perform various analysis pertaining to the HYBRID encoder and ARM.

6.1 HYBRID Encoder’s Unidirectional Layers Performance

Table 6 compares the performance of the prediction over the intermediate (unidirectional) and final (bidirectional) encodings for the best HEAR model. While the intermediate ones lag in comparison to the final, we get decent performance from intermediate ones across all except for SRL dataset. This shows that, when used selectively, UniDi intermediate predictions can serve as a good source of auxiliary predictions.

6.2 ARM vs RESTART-k.

Table 6 shows the performance of baseline - the best RESTART-k from {2, 3, 5, 8} over dev set Streaming EM and the ARM. We observe that across all the datasets, on both Streaming EM and EO metric, ARM performs significantly better than the heuristic RESTART-k mechanism. This shows that using a fixed length wait is not sufficient.

6.3 Intrinsic Performance of ARM

Table 6 shows the performance of the ARM on its binary classification task. Given the lightweight ARM architecture and the task complexity, the

Dataset	GFLOP ↓			Streaming EM ↑			EO ↓			RC ↑		
	BiDi	Hybrid	HEAR	BiDi	Hybrid	HEAR	BiDi	Hybrid	HEAR	BiDi	Hybrid	HEAR
SNIPS	74.7	43.4	**21.6**	75.1	72.5	**85.1**	15.9	16.7	**6.3**	78.2	76.3	**88.5**
CHUNK	238.8	126.3	**91.3**	77.7	77.7	**77.9**	5.0	4.8	**4.5**	91.6	**91.9**	91.6
NER	238.8	126.2	**83.8**	79.3	78.2	**81.9**	8.7	8.3	**5.4**	87.2	88.2	**90.9**
SRL	741.2	557.7	**460.4**	43.6	49.1	**50.6**	33.0	29.9	**21.0**	56.0	61.5	**62.5**

Table 5: Streaming performance over test set of datasets. HEAR significantly outperforms BiDi on all metrics improving upon efficiency and performance. Direction of arrow indicates whether higher or lower is better.

Dataset	Offline F1↑		Streaming EM ↑		EO ↓		ARM Classification
	BiDi	UniDi	restart-k	ARM	restart-k	ARM	Micro-F1
SNIPS	**93.6**	79.1	82.1	**85.3**	7.9	**6.0**	79.5
CHUNK	**90.4**	86.8	75.6	**76.5**	4.5	**4.5**	67.1
NER	**91.3**	77.9	85.6	**87.8**	4.1	**4.0**	75.7
SRL	**79.7**	42.8	41.8	**48.9**	31.5	**21.6**	80.2

Table 6: Development set ablation and analyses. From left to right - comparing performance of Hybrid encoder over sequence tagging using its bidirectional (final) vs unidirectional (intermediate) encoding, comparison of Streaming EM and EO of restart-k vs ARM, and the performance of ARM w.r.t. its ground truth policy labels.

performance is satisfactory with 80.2 F_1. However, there is a considerable margin for its improvement both in terms of model architecture and features.

6.4 ARM's Architecture Ablation

Dataset	Streaming EM ↑	EO ↓
No ARM	73.8	16.3
Linear ARM	81.3	6.8
MLP ARM	81.4	6.9
GRU ARM	**85.3**	**6.0**

Table 7: Performance of HEAR with different ARM model architectures over SNIPS development set.

Table 7 shows the Streaming EM and EO scores of HEAR with different ARM model architectures for the SNIPS development set. We observe that HEAR having either a Linear layer or MLP as ARM does much better than having no ARM and restarting at each timestep. However, modeling ARM as a GRU recurrent model gives the best scores in both metrics.

7 Related Work

Streaming (or incremental) setting has been widely studied in machine translation and parsing, dating as far back as two decades (Larchevêque, 1995; Lane and Henderson, 2001). Specifically, for incremental parsing, there are two broad approaches that have been studied: transition-based and graph-based. Transition-based incremental parsers allow for limited backtracking and correcting parsing over prefixes by keeping track of multiple parse segments (Buckman et al., 2016) or via beams search (Bhargava and Penn, 2020). Such methods can fail on garden-path sentences and long-range dependencies and only work proficiently with a large beam. Graph-based parsers incrementally assign scores to edges of the graph, discarding those edges that cause conflicts to tree-structure of the graph. (Yang and Deng, 2020) proposed an attach-juxtapose system to grow the tree, requiring restart at each new token over streaming input. (Kitaev et al., 2022) improved on its efficiency by proposing a information bottleneck. However, these methods rely on the structured nature of parsing output can not be adapted to incremental sequence tagging tasks without restarting at each token.

Recently, Madureira and Schlangen (2020) benchmarked the modern encoders on streaming sequence labelling and observed poor streaming performance of pretrained transformer models. They explored improvement strategies by adopting techniques from other streaming streaming like chunked training (Dalvi et al., 2018), truncated training (Köhn, 2018) (training model on heuristically-aligned partial input-output pairs), and prophecy (Alinejad et al., 2018) (autocompleting input using a separate language model). They observed performance degradation from truncated training unless used with prophecies. However, running a language model at each timestep for prophecy is computationally infeasible. Therefore, unlike our approach, neither methods can improve performance feasibly.

Previous works have attempted to improve computation efficiency in BiDi encoders. Monotonic attention moves away from $O(n^2)$ soft attention overhead by restricting attention to monotonically increase across timesteps (Raffel et al., 2017; Chiu and Raffel, 2018; Ma et al., 2020). However, unlike HEAR, such attempts can't maintain offline performance. Recently, Kahardipraja et al. (2021) used linear transformer (Katharopoulos et al., 2020) as unidirectional model using masking for streaming sequence tagging and classification. This approach performs well under the assumption of delayed output (Grissom II et al., 2014; Oda et al., 2015), a relaxation, where the model waits for additional tokens before predicting. Furthermore, the unidirectional model could not revise its output, rendering the model incapable of handling long-range dependency or tasks that go backward like SRL — an ability common to any model with some bidirectionality, such as HEAR. Similar drawbacks were in the partial bidirectional encoder, a bidirectional attention with restricted window, proposed by Iranzo Sanchez et al. (2022).

Improving efficiency through adaptive computing has been independently studied for reasoning-based tasks (Eyzaguirre and Soto, 2020), text generation models (Arumae and Bhatia, 2020; Eyzaguirre et al., 2022) and diffusion models (Ye et al., 2022). These works are restricted to offline settings and can be readily incorporated within the proposed overall approach of HEAR.

8 Conclusion

We propose HEAR for sequence tagging in streaming setting where the input is received one token at a time to the model. The encoder in our model is HYBRID encoder where early layers are unidirectional and later are bidirectional. It reduces the computational cost in streaming settings, by reducing the need of restart only to the later bidirectional layers while preserving the offline performance of the model. HEAR additionally consists of an ARM to predict when to restart the model. Using ARM leads to reduced number of restart of the encoder, leading to better streaming performance and further savings in computation. Compared to BiDi encoders, our model, HEAR (HYBRID encoder + ARM) reduces the computation by upto 71% in streaming settings while maintaining the performance of the BiDi encoders across various sequence tagging tasks. HEAR improved streaming EM by upto +10.0% and reduced unnecessary edits by upto -12.0%.

Limitations

An additional but small training cycle is required to train the lightweight ARM module of HEAR in order to reap the benefits of extra savings in efficiency and streaming performance. Also, even though we do not assume any language specific-design choices, we benchmarked on the standard streaming sequence labelling benchmark datasets, all of which were in English.

References

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Kristjan Arumae and Parminder Bhatia. 2020. Calm: Continuous adaptive learning for language modeling.

Timo Baumann, Okko Buß, and David Schlangen. 2011. Evaluation and optimisation of incremental processors. In *Dialogue and Discourse, 2011.*

Aditya Bhargava and Gerald Penn. 2020. Supertagging with CCG primitives. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 194–204, Online. Association for Computational Linguistics.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. JAX: composable transformations of Python+NumPy programs.

Jacob Buckman, Miguel Ballesteros, and Chris Dyer. 2016. Transition-based dependency parsing with heuristic backtracking. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2313–2318, Austin, Texas. Association for Computational Linguistics.

Fei Cai and Maarten de Rijke. 2016. A survey of query auto completion in information retrieval. *Found. Trends Inf. Retr.*, 10(4):273–363.

Shuo-yiin Chang, Guru Prakash, Zelin Wu, Qiao Liang, Tara N. Sainath, Bo Li, Adam Stambler, Shyam Upadhyay, Manaal Faruqui, and Trevor Strohman. 2022. Streaming Intended Query Detection using E2E Modeling for Continued Conversation.

Chung-Cheng Chiu and Colin Raffel. 2018. Monotonic chunkwise attention. In *International Conference on Learning Representations*.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation?

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips Voice Platform: an Embedded Spoken Language Understanding system for private-by-design voice interfaces.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 493–499, New Orleans, Louisiana. Association for Computational Linguistics.

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

Cristobal Eyzaguirre, Felipe del Rio, Vladimir Araujo, and Alvaro Soto. 2022. DACT-BERT: Differentiable adaptive computation time for an efficient BERT inference. In *Proceedings of NLP Power! The First Workshop on Efficient Benchmarking in NLP*, pages 93–99, Dublin, Ireland. Association for Computational Linguistics.

Cristóbal Eyzaguirre and Álvaro Soto. 2020. Differentiable adaptive computation time for visual reasoning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12814–12822.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352, Doha, Qatar. Association for Computational Linguistics.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to Translate in Real-time with Neural Machine Translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep Semantic Role Labeling: What Works and What's Next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. 2020. Flax: A neural network library and ecosystem for JAX.

Javier Iranzo Sanchez, Jorge Civera, and Alfons Juan-Císcar. 2022. From simultaneous to streaming machine translation by leveraging streaming history. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6972–6985, Dublin, Ireland. Association for Computational Linguistics.

Patrick Kahardipraja, Brielen Madureira, and David Schlangen. 2021. Towards Incremental Transformers: An Empirical Analysis of Transformer Models for Incremental NLU. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1178–1189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization.

Nikita Kitaev, Thomas Lu, and Dan Klein. 2022. Learned incremental representations for parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3086–3095, Dublin, Ireland. Association for Computational Linguistics.

Arne Köhn. 2018. Incremental natural language processing: Challenges, strategies, and evaluation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2990–3003, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Peter C. R. Lane and James B. Henderson. 2001. Incremental syntactic parsing of natural language corpora with simple synchrony networks. *IEEE Transactions on Knowledge and Data Engineering*, 13:2001.

J.-M. Larchevêque. 1995. Optimal incremental parsing. *ACM Trans. Program. Lang. Syst.*, 17(1):1–15.

Xutai Ma, Juan Miguel Pino, James Cross, Liezl Puzon, and Jiatao Gu. 2020. Monotonic multihead attention. In *International Conference on Learning Representations*.

Brielen Madureira and David Schlangen. 2020. Incremental Processing in the Age of Non-Incremental Encoders: An Empirical Assessment of Bidirectional Models for Incremental NLU. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 357–374, Online. Association for Computational Linguistics.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2015. Syntax-based simultaneous translation through prediction of unseen syntactic constituents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 198–207, Beijing, China. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Colin Raffel, Minh-Thang Luong, Peter J. Liu, Ron J. Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2837–2846. PMLR.

Lev Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Amit Sabne. 2020. Xla : Compiling machine learning for peak performance.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Anh Duong Trinh, Robert J. Ross, and John D. Kelleher. 2018. A Multi-Task Approach to Incremental Dialogue State Tracking. In *SEMDIAL 2018 (AixDial): the 22nd workshop on the Semantics and Pragmatics of Dialogue*, Aix-en-Provence, France.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Kaiyu Yang and Jia Deng. 2020. Strongly incremental constituency parsing with graph neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 21687–21698. Curran Associates, Inc.

Mao Ye, Lemeng Wu, and Qiang Liu. 2022. First hitting diffusion models.

Jiawei Zhou, Jason Eisner, Michael Newman, Emmanouil Antonios Platanios, and Sam Thomson. 2022. Online semantic parsing for latency reduction in task-oriented dialogue. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1576, Dublin, Ireland. Association for Computational Linguistics.

A Experiment Details

A.1 Implementation Details

All the experiments were done using the Flax (Heek et al., 2020) and Jax (Bradbury et al., 2018) with Adam optimizer (Kingma and Ba, 2014) on TPUs. We measure flops in terms of XLA's HLO (Sabne, 2020). The optimizer's learning rate is set to 1e-3, betas are set to default at (0.9, 0.999), batch size is 256 and feedforward hiden is 2048 with 512 transformer dimension and 8 heads.

For all the datasets we use the standard splits, the links to which can be found in their respective papers along with statistics. Following (Ratinov and Roth, 2009), we use the BIOES tagging scheme for the NER task and BIO scheme for the rest. In SRL, following (He et al., 2017) the indicator for predicate verb is also used as input the along with the sentence.

All the values over test sets are averaged across four seeds. For experiments involving ARM we average across four different HYBRID encoder trained from different seeded initialization. We initialize static embeddings with 300 dimensional glove embedding (Pennington et al., 2014).

A.2 Postprocessing ARM Predictions

We post process the ARM predictions to prevent too frequent or too infrequent restarts. Specifically, for hyperparameters α and β where $0 \leq \alpha < \beta$, at any time t, if the ARM hasn't restarted once since $max(0, t - \beta)$ timesteps, then ARM's prediction is set to 1, else, if the ARM has restarted atleast once since $max(0, t - \alpha)$ timesteps, then it is set to 0. We tune for the values of α and β in the range {0, 1, 2, 3, 5, 10} over the development set.

We also observe that even if bidirectional layers improve predictions over previous tokens, for the latest token only, the unidirectional label $\overrightarrow{y}_t$ is often better than $\overleftrightarrow{y}_t$. If this is observed, then, we exclude the most recent token from getting updated during restart. We tune for this binary postprocessor over the development set as well.

B Analysis and Ablation

B.1 Benefits of Optimizing Unidirectional Predictor Separately

In HYBRID encoder the performance we inhibit backward gradient flow from the auxiliary predictor over the unidirectional layers unidirectional. This is because we observed offline F1 performance drop of bidirectional predictor when trained with unidirectional predictor, dropping from 81.20 to 71.62 on CoNLL development set. Even after tuning for loss scaling for the to predictors in ratio {1:1, 3.3:1, 10:1, 33:1, 100:1}, the performance was only increased upto 75.06 F1.

B.2 Oracle Policy for ARM

We also experimented with alternate policy for ARM where it's label is conditioned on the last bidirectional restart. However, it led to a performance drop in terms of Streaming EM from 86.7 to 83.0, as observed across SNIPS development set. We attribute this poor performance on alternate policy, due to the lightweight nature of ARM. Thus, we did not proceed with this policy.

Retrieve-and-Fill for Scenario-based Task-Oriented Semantic Parsing

Akshat Shrivastava **Shrey Desai** **Anchit Gupta** **Ali Elkahky**
Aleksandr Livshits **Alexander Zotov** **Ahmed Aly**
Meta
`{akshats, shreyd, alielk, anchit,`
`alll, alexzotov, ahhegazy}@meta.com`

Abstract

Task-oriented semantic parsing models have achieved strong results in recent years, but unfortunately do not strike an appealing balance between model size, runtime latency, and cross-domain generalizability. We tackle this problem by introducing scenario-based semantic parsing: a variant of the original task which separates disambiguating an utterance's "scenario" (an intent-slot template with variable leaf spans) and generating its frame, complete with ontology and utterance tokens. This formulation closely ties to the data collection process where the scenarios are first designed followed by crowd sourced utterance annotation. Concretely, we create a Retrieve-and-Fill (RAF) architecture comprised of (1) a retrieval module which ranks the best scenario given an utterance and (2) a filling module which imputes spans into the scenario to create the frame. Our model is modular, differentiable, interpretable, and allows us to garner extra supervision from scenarios. RAF achieves strong results in high-resource, low-resource, and multilingual settings, outperforming recent approaches despite, using base pre-trained encoders and efficient decoding.

1 Introduction

Task-oriented conversational assistants typically first use semantic parsers to map textual utterances into structured frames for language understanding (Hemphill et al., 1990; Coucke et al., 2018; Gupta et al., 2018; Rongali et al., 2020; Aghajanyan et al., 2020). While these parsers achieve strong performance with rich supervision, they often face obstacles adapting to novel settings, especially ones with distinct semantics and scarce data. Recent approaches address this by improving parsers' data efficiency, such as by using pre-trained representations (Aghajanyan et al., 2020; Rongali et al., 2020), optimizing loss functions (Chen et al., 2020b), and supplying natural language prompts (Desai et al., 2021). However, these approaches typically rely on larger models and longer contexts, impeding their applicability in real-world conversational assistants. Even though non-autoregressive models can alleviate some concerns (Babu et al., 2021; Shrivastava et al., 2021; Zhu et al., 2020), to the best of our knowledge, there exists no approach which strikes an appealing balance between model size, runtime latency, and cross-domain generalizability.

We begin tackling this problem by introducing **scenario-based task-oriented semantic parsing** which is more closely tied to how new task domains are developed. A slight variation on original semantic parsing, we are given access to all supported scenarios apriori and have to parse the utterance given this scenario bank. Here, a scenario is akin to a *incomplete* frame; it is, precisely, an intent-slot template with variables as leaf spans (e.g., `IN:GET_WEATHER [SL:LOCATION` x_1 `] ]`), indicating it maps to a family of linguistically similar utterances. As domain development and data-collection usually starts out with designing the set of supported scenarios, our intuition is that by giving the model access to this bank we can train it to more explicitly reason about scenarios and improve performance especially in the low data regime.

Concretely, we propose RAF (Retrieve-and-Fill), a modular yet differentiable architecture for scenario-based task-oriented semantic parsing. Guided by the definition of our task, RAF also proceeds in two steps: (1) given an utterance, a **retrieval** module finds the highest ranking scenario and (2) given the utterance and retrieved scenario, a **filling** module imputes spans into the scenario, creating the final frame. This approach requires no extra supervision despite performing auxiliary inference: utterances and frames are typically provided, and scenarios are obtained by stripping leaf text in frames. We design RAF to capitalize on the advantages of prior work but avoid their disadvantages; using base pre-trained encoders across-the-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 430–447
May 2-6, 2023 ©2023 Association for Computational Linguistics

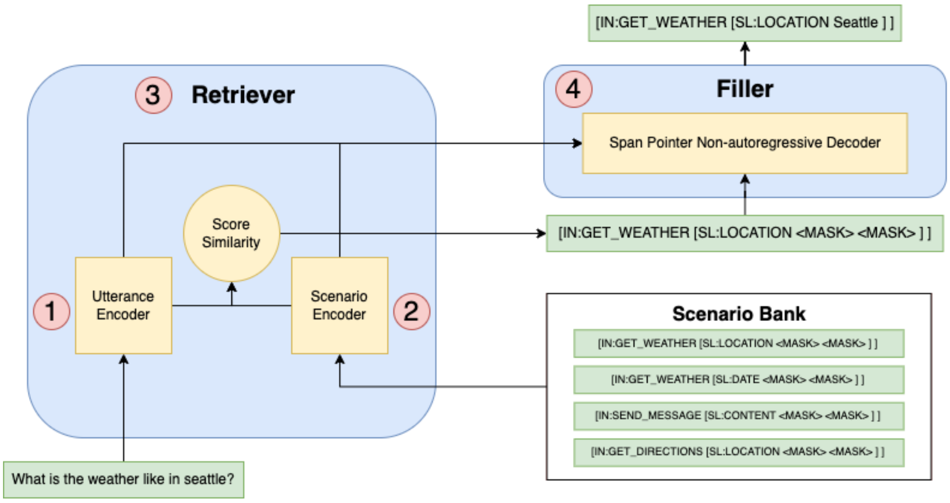


Figure 1: **High-Level Overview.** Retrieve-and-Fill (RAF) consists of 4 steps: (1) Encode the utterance via an utterance encoder (e.g., RoBERTa); (2) Encode all scenarios (Desai et al., 2021) in the scenario bank into a cached index via a scenario encoder (e.g., RoBERTa); (3) Compute the dot product similarity amongst the incoming utterance and the index of all scenarios, obtaining the top-n candidates; (4) For each retrieved scenario, leverage the non-autoregressive span pointer decoder (Shrivastava et al., 2021) to impute each scenario's spans.

board, our retrieval module caches intrinsic representations during inference and our filling module non-autoregressively decodes leaf spans in a generalizable fashion.

We evaluate our approach in high-resource, low-resource, and multilingual settings using standard task-oriented semantic parsing datasets. RAF achieves 87.52% EM on TOPv2 (Chen et al., 2020b) and 86.14% EM on TOP (Gupta et al., 2018), outperforming recent autoregressive and non-autoregressive models. RAF also excels in weakly supervised settings: on TOPv2-DA, we outperform Inventory (Desai et al., 2021) on 4 domains (alarm, music, timer, weather) despite using <128 token sequence lengths and 2-4x less parameters, on TOPv2 low resource, we outperform BART (Lewis et al., 2020), RINE (Mansimov and Zhang, 2021), and RoBERTa + Span Pointer (Shrivastava et al., 2021), and on MTOP (Li et al., 2021), we outperform XLM-R + Span Pointer, achieving 42% EM averaged across en→{es, fr, de, hi, th} transfer tasks.

To summarize, our contributions are: (1) Introducing scenario-based task-oriented semantic parsing, a novel task which requires disambiguating scenarios during typical utterance→frame predictions; (2) Creating RAF (Retrieve-and-Fill), a modular yet differentiable architecture composed of a retrieval and filling module for solving scenario-based task-oriented semantic parsing; and (3) Achieving strong results in high-resource, low-resource, and multilingual settings, outperforming recent models such as Span Pointer, Inventory, and RINE by large margins, while also optimizing for model size, runtime latency, and cross-domain generalizability.

2 Scenario-based Semantic Parsing

We formally introduce the task of scenario-based semantic parsing. Task-oriented conversational assistants support a wide range of domains (e.g., calling, reminders, weather) to maximize coverage over users' needs. Over time, in response to requests and feedback, developers often iterate on assistants' skill-sets by adding new domains. Though the crowdsourcing process of collecting and annotating samples—utterances and frames—for new domains can be accomplished in many ways, we propose a two-step methodology where developers (a) develop scenarios which roughly describe a family of samples and (b) collect linguistically-varied samples consistent with each scenario. We elaborate more on our scenario-based methodology below.

Scenario Definition. We define a **scenario** as an intent-slot rule which abstracts away linguistic variation in utterances. More specifically, it is a decoupled semantic frame (Aghajanyan et al., 2020) with variables in leaf spans, indicating it can subclass utterances with similar syntactic and semantic structure, an example is showing in table 1. Our no-

Utterance	Frame
Scenario: `[IN:GET_WEATHER [SL:LOCATION` x_1 `] ]`	
what's the weather in seattle	`[IN:GET_WEATHER [SL:LOCATION` seattle `] ]`
how's the forecast in sf	`[IN:GET_WEATHER [SL:LOCATION` sf `] ]`
Scenario: `[IN:GET_WEATHER [SL:LOCATION` x_1 `] [SL:DATE_TIME` x_2 `] ]`	
what's the weather in seattle tomorrow	`[IN:GET_WEATHER [SL:LOCATION` seattle `] [SL:DATE_TIME` tomorrow `] ]`
how's the forecast in sf at 8pm	`[IN:GET_WEATHER [SL:LOCATION` sf `] [SL:DATE_TIME` 8pm `] ]`

Table 1: **Scenario Description.** Scenarios are intent-slot templates with missing slot text, suggesting they subclass linguistically similar utterances. We show examples of utterances, scenarios, and frames in the weather domain; each scenario consists of multiple (utterance, frame) pairs.

tion of a scenario is inspired by production rules in constituency parsing (Chomsky, 1959), but the parallel is not exact given our scenarios have semantic not syntactic types.

Using scenarios, we can effectively quantize the space of possible utterances by identifying and defining slices of user requests. Our solution also offers fine-grained control over precision and recall, which is important in real-world systems; we can collect more paraphrases to improve precision and we can create more scenarios to improve recall.

We collect scenarios by taking train, eval, and test frames from our datasets, and stripping out the utterance text, to yield a decoupled semantic frame. As discussed later in this manuscript, our approach requires knowing, beforehand, the space of possible scenarios, in order to perform global inference.

Case Study: Weather. Table 1 shows an example setting where we crowdsource weather domain samples using the scenario-based methodology outlined above. To begin, we may envision building a weather system which supports requests with location and/or date-time information. Therefore, we can define two scenarios: (1) a family of samples with one location span; and (2) a family of samples with one location span and one date-time span. Each scenario explicitly outlines the intents and slots that must be present, as scenarios are not "one-size-fits-all" rules with optional slotting. So, as an example, the utterance "how's the forecast in sf" would not be compatible with the scenario `[IN:GET_WEATHER [SL:LOCATION` x_1 `] [SL:DATE_TIME` x_2 `] ]` since it does not have a date-time span as specified by the x_2 variable.

Task Definition. Finally, we precisely define our task of scenario-based semantic parsing. For the typical task of semantic parsing, we define U as a random variable over utterances, F as a random variable over frames, and model $P(F|U)$: find the most likely frame given the utterance. However, because we introduce scenarios as a coarse, intermediate representation of frames, we additionally define S as the set of all supported scenarios given a priori, and model $P(F|U, S)$.

3 RAF: Retrieve and Fill

We propose a model called RAF (Retrieve and Fill) for scenario-based semantic parsing that naturally decomposes the task into a coarse-to-fine objective where we (a) find the most likely scenario given the utterance and (b) find the most likely frame given the utterance and scenario. More concretely given an utterance u and scenarios $s_1, \cdots, s_n$, as well as a gold scenario s^* and gold frame f^*, we learn our model as follows:

1. Retrieve (Coarse Step; §3.1): A **retrieval module** maximizes $P(S = s^*|U = u)$ by learning to retrieve scenario s_i given utterance u, e.g., "what's the weather in seattle" → `[IN:GET_WEATHER [SL:LOCATION` x_1 `] ]`.

2. Fill (Fine Step; §3.2): A **filling module** maximizes $P(F = f^*|U = u, S = s^*)$ by decoding the most likely frame f given the structure of scenario s_i^* and spans of utterance u, e.g., `[IN:GET_WEATHER [SL:LOCATION` x_1 `] ]` → `[IN:GET_WEATHER [SL:LOCATION` seattle `] ]`.

RAF is a composable yet differentiable model which implements coarse-to-fine processing: we first develop a coarse-grained sketch of an utterance's frame, then impute fine-grained details to achieve the final frame. In these types of approaches, there often exists a trade-off when creating the intermediate representation; if it is "too coarse", the filling module suffers, but if it is "too fine", the retrieval module suffers. We find scenarios, as defined in §2, offer the most appealing solution, as the retrieval module unearths rough syntactic-semantic structure, while the filling module focuses in on imputing exact leaf spans.

In the following sub-sections, we discuss the technical details behind the retrieval and filling modules, as well as describe the training and inference procedures.

3.1 Retrieval Module

First, we discuss the **coarse-grained step** of RAF, which aims to find the best-fitting scenario for an utterance. We formulate this task as a metric learning problem. Here, we can maximize the similarity $\text{sim}(u, s^*)$ between an utterance u and scenario s^* as judged by a scalar metric. This offers numerous advantages: we can explore ad-hoc encodings of utterances and scenarios, adjust the output space dynamically during inference, and compute exact conditional probabilities by leveraging a (tractable) partition function.

3.1.1 Bi-Encoder Retrieval

Following retrieval modeling in open-domain QA (Karpukhin et al., 2020), we specifically leverage pre-trained encoders (E_U for utterances and E_S for scenarios) to compute dense vector representations, then maximize the dot product similarity $\text{sim}(u, s^*) = E_U(u)^\top E_S(R(s^*))$ between utterance-scenario pairs (u, s^*); the precise nature of R is discussed in §3.1.3. To learn such a metric space and avoid degenerate solutions we need to train the encoders to *pull* positive pairs together and *push* negative pairs apart. Hence, we need access to both positive (gold; (u, s^+)) and negative (non-gold; (u, s^-)) pairs.

3.1.2 Negatives Sampling

The choice of negatives has a large impact on retrieval performance, which is consistent with findings in information retrieval (Zhan et al., 2021; Karpukhin et al., 2020). We explore two types of negative sampling to improve retrieval performance: in-batch negatives and model-based negatives.

In-Batch Negatives. We mine positive and negative pairs from each training batch using in-batch negatives (Karpukhin et al., 2020). Let $\mathbf{U}$ and $\mathbf{S}$ be the utterance and scenario matrices, each being a $(B \times d)$ matrix consisting of d-dimensional embeddings up to batch size B. We obtain B^2 similarity scores upon computing the similarity matrix $\mathbf{M} = \mathbf{US}^\top$, where $\mathbf{M}_{i=j}$ consists of positive scores and $\mathbf{M}_{i\neq j}$ consists of negative scores. For each positive utterance-scenario pair (u_i, s_i^+), we now have $(B-1)$ negative pairs $\{(u_i, s_{ij}^-)\}_{i\neq j}$. Having collected multiple negative pairs per positive pair, we leverage the contrastive loss for utterance i is defined in Karpukhin et al. (2020); Chen et al. (2020a):

$$\mathcal{L}^i_{\text{retrieval}}(u_i, s_1, \cdots, s_b) = \log \frac{e^{\text{sim}(u_i, s_i)}}{\sum_{j=0}^{b} e^{\text{sim}(u_i, s_{ij})}} \quad (1)$$

Since our set of scenarios isn't huge this can result in conflicts so we additionally implement **identity masking** while training which ensures that for every utterance each scenario is present at-most once in it's positive/negative set.

Model-Based Negatives. Following prior work in IR (Xiong et al., 2020; Oğuz et al., 2021) we train an initial retrieval model that uses only In-Batch negatives. This model is used to rank all scenarios for each utterance and the top K ranked scenarios which are not the gold scenario are selected as additional negative examples. Those negatives examples are expected to be harder examples since they were ranked higher by the initial model. We train a new model using those added k scenarios as explicit negative examples for each positive example.

3.1.3 Frame Representation

While we have covered scenario-based retrieval above, we have not yet precisely described how dense vectors for scenarios are computed. Recall our definition of utterance-scenario similarity in §3.1.1: our objective is to maximize $\text{sim}(u, s) = E_U(u)^\top E_S(R(s))$, where R is a string transformation applied to scenarios.

Following Desai et al. (2021), we leverage **intrinsic modeling** to rewrite intents and slots as a composition of their intrinsic parts in a single string. Desai et al. (2021) define two, in particular: the categorical type (e.g., "intent" or "slot") and language span (e.g., "get weather" or "location"). Using this methodology, we can transform the scenario `IN:GET_WEATHER [SL:LOCATION` x_1 `] ]` → `[ intent | get weather [ slot | location` x_1 `] ]`, which is inherently more natural and descriptive.

Guided by the general concept of intrinsic modeling, our goal here is to define R such that

$E_U(u)^\top E_S(s) \leq E_U(u)^\top E_S(R(s))$, all else being equal. We discuss our approach in detail in the following sub-sections.

3.1.4 Language Spans

Desai et al. (2021) chiefly use an **automatic** method to extract language spans from ontology labels. For example, using standard string processing functions, we can extract "get weather" from `IN:GET_WEATHER`. While this method is a surprisingly strong baseline, it heavily relies on a third-party developers' notion of ontology nomenclature, which may not always be pragmatically useful. In TOPv2 (Chen et al., 2020b), our principal evaluation dataset, there exists ambiguous labels like `SL:AGE`—is this referring to the age of a person, place, or thing?

Therefore, to improve consistency and descriptiveness, we propose a **handmade** method where we manually design language spans for each ontology label.[1] The most frequent techniques we use are (1) using the label as-is (e.g., `IN:GET_WEATHER` → "get weather"); (2) inserting or rearranging prepositions (e.g., `IN:ADD_TO_PLAYLIST_MUSIC` → "add music to playlist"; and (3) elaborating using domain knowledge (e.g., `IN:UNSUPPORTED_ALARM` → "unsupported alarm request").

3.1.5 Example Priming

Despite using curation to improve ontology label descriptions, there are still many labels which remain ambiguous. One such example is `SL:SOURCE`; this could refer to a travel source or messaging source, but without seeing its exact manifestations, it is challenging to fully grasp its meaning. This motivates us to explore **example priming**: augmenting scenario representations with randomly sampled, dataset-specific slot values. This can help our model further narrow down the set of spans each slot maps to during parsing. Furthermore, our examples are just spans, so they are straightforward to incorporate into our representation. For example, for the slot `SL:WEATHER_TEMPERATURE_UNIT`, we can augment and contextualize its representation "slot | unit of weather temperature" with "slot | unit of weather temperature | F / C" where "F" and "C" are examples which appear in our dataset.

3.1.6 Representation Sampling

Our scenario-based retrieval task performs reasoning over a compact set of scenarios, unlike large-scale, open-domain tasks such as information retrieval and question answering which inculcate millions of documents. As such, **the chance our system overfits to a particular representation is much greater**, no matter what it is set to. So, we instead make R stochastic by uniformly sampling unique frame representations for encoding scenarios, and only using the handmade representations during inference; Table 12 enumerates the complete set of outcomes.

3.2 Filling Module

We use Span pointer model (Shrivastava et al., 2021; Nicosia et al., 2021) as the basis of our infilling model. Figure 2.A shows the how the model works. In our case, we use scenario tokens from the retrieved scenario as input tokens to decoder as shown in figure 2.B. We also tried to pass the scenario encoder output as input to the decoder to replace the discrete input to the decoder and called this **scenario fusion** as shown in figure 2.C.

Our final objective is $\mathcal{L}_{\text{filling}} = \text{NLL}(f^*, f) + \alpha\text{LS}(f)$ where LS refers to label smoothing (Pereyra et al., 2017).

4 Experiments and Results

We evaluate RAF in three settings: a high-resource setting (100,000+ training samples), low-resource setting (1-1,000 training samples), and multilingual setting (0 training samples). Hyper parameter details are described in Appendix §E. Our goal here is to show that our system both achieves competitive performance on established benchmarks and offers substantial benefits in resource-constrained environments where training samples are limited.

4.1 Datasets for Evaluation

Following prior work in task-oriented semantic parsing, we use 5 datasets for evaluation: TOP (Gupta et al., 2018), TOPv2 (Chen et al., 2020b), TOPv2-LR (Low Resource; Chen et al. (2020b)), TOPv2-DA (Domain Adaptation; Desai et al. (2021)), and MTOP (Li et al., 2021). TOP and TOPv2 are used for high-resource experiments, TOPv2-LR and TOPv2-DA are used for low-resource experiments, and MTOP is used in multilingual experiments.

---

[1] See Appendix §D for our curated intent and slot descriptions, respectively.

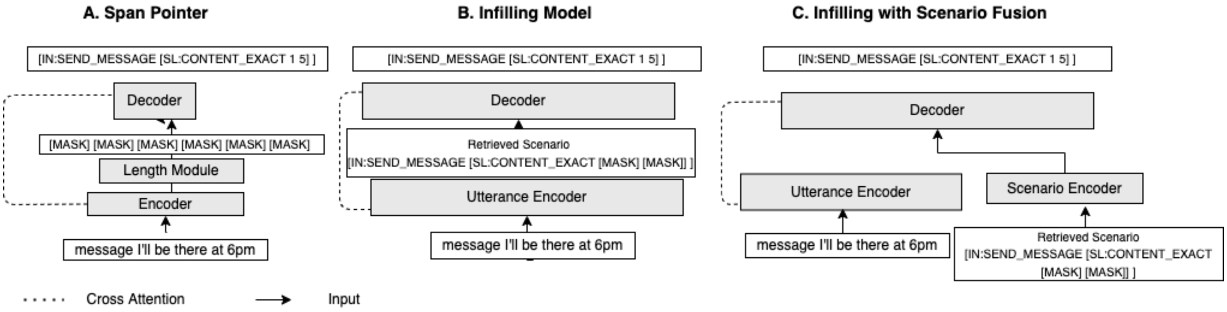


Figure 2: **Infilling Model.** A. the Span Pointer model in (Shrivastava et al., 2021). B. the Infilling model using tokens from retrieved scenario and C. the infilling model with scenario fusion where scenario encoder output is connected to the decoder directly. Note in all cases the decoder apply cross attention on the utterance encoder output.

4.2 Systems for Comparison

We compare against multiple task-oriented semantic parsing models, which cover autoregressive (AR), and non-autoregressive (NAR) training. See Aghajanyan et al. (2020); Mansimov and Zhang (2021); Babu et al. (2021); Shrivastava et al. (2021) for detailed descriptions of these models.

The autoregressive models consist of BART (Lewis et al., 2020) and RoBERTa (Liu et al., 2019), and RINE (Mansimov and Zhang, 2021), and the non-autoregressive models are RoBERTa NAR (Babu et al., 2021) and RoBERTa NAR + Span Pointer (Shrivastava et al., 2021). These models are applicable to both high-resource and low-resource settings; though, for the latter, we also add baselines from Desai et al. (2021): CopyGen (BART + copy-gen decoder) and Inventory (BART + intrinsic modeling). The multilingual setting only requires swapping RoBERTa with XLM-R (Conneau et al., 2020).

We denote our system as RAF in our experiments. Unless noted otherwise, we use RoBERTa$_{\text{BASE}}$ for the utterance encoder θ_U and secnario θ_S and a random-init, copy-gen, transformer decoder for the frame decoder θ_F. As alluded to before, we swap RoBERTa$_{\text{BASE}}$ with XLM-R$_{\text{BASE}}$ for multilingual experiments.

4.3 High-Resource Setting

First, we evaluate RAF in a high-resource setting where hundreds of thousands are samples are available for supervised training; Table 2 shows the results. RAF achieves strong results across-the-board, using both base and large pre-trained encoders: **RAF$_{\text{BASE}}$ consistently outperforms other base variants by 0.25-0.5 EM and RAF$_{\text{LARGE}}$ comparatively achieves the best results on TOPv2.**

Model	TOPv2	TOP
Type: Autoregressive Modeling (Prior)		
RoBERTa$_{\text{BASE}}$	86.62	83.17
RoBERTa$_{\text{LARGE}}$	86.25	82.24
BART$_{\text{BASE}}$	86.73	84.33
BART$_{\text{LARGE}}$	87.48	85.71
RINE$_{\text{BASE}}$	—	87.14
RINE$_{\text{LARGE}}$	—	**87.57**
Type: Non-Autoregressive Modeling (Prior)		
RoBERTa$_{\text{BASE}}$	85.78	82.37
+ Span Pointer	86.93	84.45
RoBERTa$_{\text{LARGE}}$	86.25	83.40
+ Span Pointer	87.37	85.07
Type: Scenario Modeling (Ours)		
RAF$_{\text{BASE}}$	87.11	86.00
RAF$_{\text{LARGE}}$	**87.52**	86.14

Table 2: **High-Resource Results.** Exact Match (EM) on TOPv2 (Chen et al., 2020b) and TOP (Gupta et al., 2018). We compare various semantic parsing paradigms: autoregressive, non-autoregressive, and scenario. RAF achieves strong performance on TOPv2 and TOP, illustrating its competitiveness with state-of-the-art models.

4.4 Low-Resource Setting

Having established our system is competitive in high-resource settings, we now turn towards evaluating it in low-resource settings, where training samples are not as readily available. Here, we chiefly consider two setting types: a **high difficulty** setting (TOPv2-DA) with 1-10 samples and a **medium difficulty** setting (TOPv2-LR) with 100-1,000 samples. The exact number of samples in a few-shot training subset depend on both the subset's cardinality and sampling algorithm.

Tables 3 and 4 show results on the high and medium difficulty settings, respectively. **RAF achieves competitive results in the high-difficulty setting, outperforming both CopyGen$_{\text{BASE}}$ and Inventory$_{\text{BASE}}$ by large**

	alarm	music	timer	weather
CopyGen$_{\text{BASE}}$	47.24	25.58	16.62	47.24
CopyGen$_{\text{LARGE}}$	36.91	23.84	32.64	53.08
Inventory$_{\text{BASE}}$	62.13	23.00	28.92	54.53
Inventory$_{\text{LARGE}}$	**67.25**	**38.68**	48.45	**61.77**
RAF$_{\text{BASE}}$ (ours)	62.71	35.47	**55.06**	61.05

Table 3: **High-Difficulty Low-Resource Results.** EM on the 1 SPIS split of TOPv2-DA (Desai et al., 2021). Compared to Inventory and CopyGen baselines, RAF achieves competitive performance with a fraction of parameter usage.

	Weather Domain (SPIS)					
	10	25	50	100	500	1000
Type: Autoregressive Modeling (Prior)						
RoBERTa$_{\text{BASE}}$ AR	69.71	74.90	77.02	78.69	—	86.36
BART$_{\text{BASE}}$ AR	73.34	73.35	76.58	79.16	—	86.25
RINE$_{\text{BASE}}$	—	74.53	—	—	87.80	—
RINE$_{\text{LARGE}}$	—	77.03	—	—	87.50	—
Type: Non-Autoregressive Modeling (Prior)						
RoBERTa$_{\text{BASE}}$ NAR	59.01	72.12	73.41	78.48	—	87.42
+ Span Pointer	72.03	74.74	74.85	78.14	—	**88.47**
Type: Scenario Modeling (Ours)						
RAF$_{\text{BASE}}$	**75.10**	**78.74**	**77.53**	**79.67**	**87.91**	88.17

Table 4: **Medium-Difficulty Low-Resource Results.** EM on various SPIS splits of the TOPv2 (Chen et al., 2020b) weather domain. RAF largely outperforms autoregressive and non-autoregressive models, trailing RoBERTa-Base + Span Pointer only in a high-resource split.

margins; notably, on timer, we nearly double Inventory$_{\text{BASE}}$'s exact match score. RAF also performs well in the medium-difficulty setting; our system consistently outperforms prior autoregressive, and non-autoregressive models.

4.5 Multilingual Setting

Finally, we consider a multilingual setting, where a model trained on English samples undergoes zero-shot transfer to non-English samples. In Table 5, we see that, **compared to XLM-R$_{\text{BASE}}$ NAR + Span Pointer, RAF achieves +2.3 EM averaged across all 5 non-English languages.** Upon inspecting this result more closely, RAF's performance is strong across both typologically *similar* languages (+4.8 EM on Spanish, +3.5 EM on French) and *distinct* languages (+2.7 EM on Hindi and Thai).

5 Ablations and Analysis

We perform model ablations on RAF, removing core retrieval- and filling-related components we originally introduced in §3 to better understand the design decisions. From Tables 6 and Table 7, we draw the following conclusions:

Negatives are important for accurate scenario retrieval. The metric learning objective for retrieval, as introduced in §3.1.2, precisely delineates between positive and negative samples. Our ablations show model-based negatives and identity masking are critical to achieve best performance; when removing model-based negatives, for example, retrieval accuracy drops by 3%+. We also investigate training RAF with heuristic-based negatives: a simple algorithm which finds top-k similar scenarios with string-based edit distance (Appendix C). However, heuristic-based negatives regress both retrieval-only and end-to-end approach, suggesting model-based negatives are more informative.

Sharing parameters between retrieval encoders improves quality. Our retrieval module has two encoders: an utterance encoder E_U and a scenario encoder E_S. An important design decision we make is tying both encoders' parameters together with RoBERTa (Liu et al., 2019); this improves end-to-end performance by roughly +0.6 EM. We believe that parameter sharing among retrieval encoders improves generalizability: because there are more vastly more unique utterances than scenarios, the scenario encoder may overfit to a select set of scenarios, so weight tying enables the joint optimization of both encoders.

Scenario fusion enables better end-to-end modeling. Because RAF is composed of two neural modules—the retrieval and filling modules—chaining them together arbitrarily may result in information loss. The filling module chiefly uses scenario token embeddings to reason over the retrieval module's outputs. Our results show that scenario fusion, initializing these embeddings using the scenario encoder's final state, improves upon random init by +0.77 EM and +0.65 EM-S.

Intrinsic representations improve low-resource performance. When comparing the high-level scenario representation, we see that canonical (e.g., "[IN:GET_WEATHER [SL:LOCATION]") underperforms intrinsic (e.g., "[intent | get weather [slot | location]]") by a wide margin (-4.49%) in the target domain. This implies RAF *better* understands scenarios' natural language descriptions even if they contain unseen, domain-specific terms. **Furthermore, we see leveraging all concepts (handmade, automatic, examples) achieves both competitive source and target performance.** Even though excluding handmade improves target perfor-

		Zero-Shot Evaluation					
	`en`	`en`→`es`	`en`→`fr`	`en`→`de`	`en`→`hi`	`en`→`th`	Avg
XLM-R$_{\text{BASE}}$ NAR	78.3	35.2	32.2	23.6	18.1	16.7	25.2
+ Span Pointer	**83.0**	51.2	51.4	**42.0**	29.6	27.3	40.3
RAF$_{\text{BASE}}$ (ours)	81.1	**56.0**	**54.9**	40.1	**32.1**	**30.0**	**42.6**

Table 5: **Multilingual Results.** We perform zero-shot experiments where we fine-tune a parser on English (`en`), then evaluate it a non-English language—Spanish (`es`), French (`fr`), German (`de`), Hindi (`hi`), and Thai (`th`)—without fine-tuning. Average EM (Avg) is taken over the five non-English languages. RAF outperforms XLM-R-Base + Span Pointer by +2.3% on average.

Model	EM	EM-S
Classify and Fill	84.80	87.16
RAF$_{\text{BASE}}$	**87.03**	**89.34**
- Hard Negatives	83.69	86.00
- Identity Masking	85.87	88.23
- Scenario Fusion	86.26	88.69
- Parameter sharing	86.76	89.10
- Repr. Sampling	86.70	89.05
+ Heuristic negatives	85.66	89.10

Table 6: **Model Ablations.** We assess several components of our model, individually removing them and evaluating EM (Exact Match) and EM-S (Exact Match of Scenarios, i.e., intent-slot templates without slot text) on TOPv2 (Chen et al., 2020b) validation set.

	Source Fine-Tuning		Target Fine-Tuning	
Model	EM	EM-S	EM	EM-S
Canonical Repr.	86.67	88.74	74.25	76.80
Intrinsic Repr.	**87.10**	**89.15**	78.74	81.70
- Automatic	87.02	89.06	78.92	81.80
- Handmade	86.85	88.87	**79.27**	**82.29**
- Examples	86.91	88.90	78.25	80.89

Table 7: Scenario representation ablation, where the target domain is a 25 SPIS split of the TOPv2 (Chen et al., 2020b) weather. Following the typical few-shot fine-tuning methodology, we perform source fine-tuning on all domains except weather and reminder, then perform target fine-tuning on a specific split.

mance (+0.53%), it regresses source performance (-0.28%), suggesting sampling all representations is more generalizable.

Runtime Latency Empirically we find that RAF retains the runtime latency of Span Pointer, due to it's non-autoregressive decoding nature despite leveraging the retrieval module. We find that the difference is within 3ms P99 and significantly faster than the autoregressive counter part (189ms). We report the full experiment details in A.2.

Interpretablity RAF offers a more interpretable modeling paradigm Due to (1) RAF encoding utterances and scenarios into a joint embedding space which we can directly visualize. We present two case studies leveraging these visualizations: domain development (B.1) and error analysis (B.2). (2) RAF retains modularity of components to isolate retrieval vs filling issues. In A.1 we show the comparisons of retrieval vs filling to highlight potential of each module, in particular on the TOPv2 eval dataset, oracle retrieval improves performance +9.5% where oracle filling improves performance +2.3% indicating the importance of retrieval.

5.1 Extra-Scenario Generalization

A core difference between scenario-based and non-scenario-based (seq2seq; autoregressive or non-autoregressive) models is that scenario-based models "know" of all scenarios beforehand, while seq2seq models do not, and therefore have to purely rely on generalization. We further quantify the impact that this has by dividing overall EM using two groups: (1) Known vs. Unknown - i.e scenarios in the training dataset vs. scenarios only in the test dataset and (2) In-Domain vs. Out-of-Domain - scenarios with a supported intent vs. unsupported intent (e.g., `IN:UNSUPPORTED_*`).

From the results in Table 8, we draw a couple of conclusions. First on Unknown EM, even though Span pointer, BART models are capable of generating novel scenarios not part of the train set they do so poorly. RAF does much better on this given the caveat that it adds the novel scenarios to its frame index beforehand but it hasn't seen any paired utterance to them. Second, RAF outperforms on In-Domain EM but underperforms on Out-of-Domain EM. Because RAF leverages intrinsic descriptions of scenarios, the word "unsupported" may not precisely capture what it means for an utterance to be in- vs. out-of-domain.

6 Related Work

Scaling Semantic Parsing A critical theme in semantic parsing is reducing data requirements to stand up new domains and scenarios. Existing works rely on leveraging large language models such as BART (Lewis et al., 2020) with augmenta-

Model	EM	Known EM	Unknown EM	ID EM	OOD EM
RAF_{BASE}	**87.14**	88.30	**59.96**	**88.67**	44.11
$\text{SpanPointer}_{\text{BASE}}$	86.76	**88.50**	46.20	88.07	**49.70**
$\text{BART}_{\text{BASE}}$	86.72	88.33	49.15	88.03	49.69

Table 8: Comparing EM on Known vs. Unknown and In-Domain (ID) vs. Out-of-Domain (OD) frames. RAF performs better on unknown frames, but struggles with out-of-domain frames.

tions for scaling. In particular Chen et al. (2020b) introduce a meta-learning approach to improve domain scaling in the low-resource setting. Other works such as (Liu et al., 2021; Zhu et al., 2020; Mansimov and Zhang, 2021) aim to improve scaling through new decoding formulations. Desai et al. (2021) introduce the concept of **intrinsic modeling** where we provide a human-readable version of the semantic parsing ontology as context to encoding to improve few-shot generalization.

Our work leverages the intrinsic modeling paradigm by building a function R to convert each intent-slot scenario into a readable representation via intent slot descriptions and example priming. Furthermore, our bi-encoder based retrieval setup allows us to inject additional context into each scenario and cache it to an index in order to retain inference efficiency.

Retrieval Based Semantic Parsing Finally, there has been a recent trend towards dense retrieval in various NLP domains such as machine translation (Cai et al., 2021), question answering (Karpukhin et al., 2020), text generation (Cai et al., 2019) and language modeling (Borgeaud et al., 2018). Recent works also introduce retrieval-based semantic parsing: RetroNLU (Gupta et al., 2021) and CASPER (Pasupat et al., 2021) both leverage a retrieval step to provide examples as context to seq2seq models.

Our approach differs in two ways: (1) We phrase our problem as utterance-to-scenario retrieval rather than utterance-to-utterance retrieval. This allows us to look into supporting new scenarios with minimal-to-no-data required for retrieval. (2) Prior work leverage a separate module (Pasupat et al., 2021) or separate iteration (Gupta et al., 2021) for retrieval. We conduct our retrieval after encoding but prior to decoding as an intermediate step for non-autoregressive parsing. This allows our model to retain similar inference speed to one shot non-autoregressive decoding despite leveraging retrieval.

7 Conclusion

In this paper, we tackle scenario-based semantic parsing with retrieve-and-fill (RAF), a coarse-to-fine model which (a) retrieves a scenario with the best alignment to an utterance and (b) fills the scenario with utterance spans in leaf positions. Experiments show our model achieves strong results in high-resource, low-resource, and multilingual settings. The modular nature of our architecture also lends itself well to interpretability and debuggability; we perform several case studies uncovering the inner-workings of our approach.

8 Limitations

Although RAF has shown promising results on task oriented parsing, we identify the following limitations.

Use of Scenario Index Critically we introduce the scenario based parsing task, a problem where we assume knowledge of all possible scenarios across train/test/eval. This new task is inspired by how domain development usually happens 1)Define scenarios 2) collect and annotate utterances. While powerful and applicable when grammars and downstream applications can provide a scenario index, it prohibits generating novel scenarios or intent/slot combination that have not been indexed, where as prior seq2seq approaches are capable of this. We argue that such generalization is not a priority in pipelined task oriented assistant systems as the downstream application usually handles a set of pre-defined scenarios. Refer to §5.1 for more analysis/commentary on this.

Training Cost RAF has a high training cost due to the use of dense retrieval models and model based negative sampling. While inference is cheap due to a cached index and non-autoregressive decoding. During training, we must compute forward/backward over 2 large encoders (utterance encoder/scenario encoder) which uses significant memory and prohibits larger batch sizes. Additionally, model based negative samples requires two training iterations: first with in-batch negatives, second using the first stage model to identify hard negative samples from the model.

References

Armen Aghajanyan, Jean Maillard, Akshat Shrivastava, Keith Diedrick, Michael Haeger, Haoran Li, Yashar Mehdad, Veselin Stoyanov, Anuj Kumar, Mike Lewis, and Sonal Gupta. 2020. Conversational Semantic Parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2021. Better Fine-tuning by Reducing Representational Collapse. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Arun Babu, Akshat Shrivastava, Armen Aghajanyan, Ahmed Aly, Angela Fan, and Marjan Ghazvininej. 2021. Non-Autoregressive Semantic Parsing for Compositional Task-Oriented Dialog. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2018. Improving Language Models by Retrieving from Trillions of Tokens. *arXiv preprint arXiv:2112.04426*.

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, and Shuming Shi. 2019. Retrieval-guided dialogue response generation via a matching-to-generation framework. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1866–1875, Hong Kong, China. Association for Computational Linguistics.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural Machine Translation with Monolingual Translation Memory. *arXiv preprint arXiv:2105.11269*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv preprint arXiv:2002.05709*.

Xilun Chen, Ashish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020b. Low-Resource Domain Adaptation for Compositional Task-Oriented Semantic Parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Noam Chomsky. 1959. *On Certain Formal Properties of Grammars*. Elsevier.

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

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips Voice Platform: An Embedded Spoken Langauge Understanding System for Private-by-Design Voice Interfaces. *arXiv preprint arXiv:1805.10190*.

Shrey Desai, Akshat Shrivastava, Alexander Zotov, and Ahmed Aly. 2021. Low-resource task-oriented semantic parsing via intrinsic modeling. *arXiv preprint arXiv:2104.07224*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic Parsing for Task Oriented Dialog using Hierarchical Representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Vivek Gupta, Akshat Shrivastava, Adithya Sagar, Armen Aghajanyan, and Denis Savenkov. 2021. Retronlu: Retrieval augmented task-oriented semantic parsing. *arXiv preprint arXiv:2109.10410*.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS Spoken Language Systems Pilot Corpus. In *Proceedings of the Workshop on Speech and Natural Language*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. *arXiv preprint arXiv:2004.04906*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for

Natural Language Generation, Translation, and Comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A Comprehensive Multilingual Task-Oriented Semantic Parsing Benchmark. *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Zihan Liu, Genta Indra Winata, Peng Xu, and Pascale Fung. 2021. X2Parser: Cross-Lingual and Cross-Domain Framework for Task-Oriented Compositional Semantic Parsing. *arXiv preprint arXiv:2106.03777*.

Elman Mansimov and Yi Zhang. 2021. Semantic Parsing in Task-Oriented Dialog with Recursive Insertion-based Encoder. *arXiv preprint arXiv:2109.04500*.

Massimo Nicosia, Zhongdi Qu, and Yasemin Altun. 2021. Translate & fill: Improving zero-shot multilingual semantic parsing with synthetic data. *CoRR*.

Barlas Oğuz, Kushal Lakhotia, Anchit Gupta, Patrick Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen tau Yih, Sonal Gupta, and Yashar Mehdad. 2021. Domain-matched pre-training tasks for dense retrieval. *arXiv preprint arXiv:2107.13602*.

Panupong Pasupat, Yuan Zhang, and Kelvin Guu. 2021. Controllable semantic parsing via retrieval augmentation. *arXiv preprint arXiv:2110.08458*.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing Neural Networks by Penalizing Confident Output Distributions. In *Proceedings of the International Conference on Learning Representations (ICLR): Workshop Track*.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't Parse, Generate! A Sequence to Sequence Architecture for Task-Oriented Semantic Parsing. In *Proceedings of the Web Conference (WWW)*.

Akshat Shrivastava, Pierce Chuang, Arun Babu, Shrey Desai, Abhinav Arora, Alexander Zotov, and Ahmed Aly. 2021. Span Pointer Networks for Non-Autoregressive Task-Oriented Semantic Parsing. In *Proceedings of the Findings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing Dense Retrieval Model Training with Hard Negatives. *arXiv preprint arXiv:2104.08051*.

Qile Zhu, Haidar Khan, Saleh Soltan, Stephen Rawls, and Wael Hamza. 2020. Don't Parse, Insert: Multilingual Semantic Parsing with Insertion Based Decoding. *arXiv preprint arXiv:2010.03714*.

A Analysis

A.1 Retrieval vs. Filling

We now turn towards better understanding the aspects our model struggles with. Because RAF jointly optimizes both the retrieval and filling modules, one question we pose is whether the retrieval or filling task is more difficult. We create three versions of RAF: (1) standard retrieval + standard filling, (2) oracle retrieval + standard filling, and (3) standard retrieval + oracle filling. By comparing models (1), (2), and (3), we can judge the relative difficulty of each task.

We begin by evaluating these models in a high-resource setting; on the TOPv2 eval dataset, the standard model gets 87.03%, retrieval oracle gets 96.56%, and filling oracle gets 89.34%. **Here, the gap between models (1)-(2) is +9.53%, while the gap between models (1)-(3) is +2.31%, indicating the retrieval module is the main performance bottleneck.** We also perform experiments in low-resource and multilingual settings, displaying results in Tables 9 and 10, respectively. These results also confirm the same trend: in both settings, the retrieval oracle achieves the best performance, notably achieving +18.3% averaged across 5 multilingual transfer experiments.

Despite retrieval having the most room for improvement, we also see some evidence filling struggles in certain multilingual transfer cases; for example, providing gold spans can improve en→th transfer by +16.7%. As such, there is ample opportunity for optimizing the retrieval and filling modules in future work.

A.2 Runtime Latency

We compare the latency of the non-autoregressive Span Pointer Networks against RAF to show that there is minimal degradation, despite the improved generalization from RAF. In order to measure latency, we measure wall clock time (ms) of the SpanPointer$_{\text{BASE}}$, RAF$_{\text{BASE}}$, and Autoregressive$_{\text{BASE}}$, all models use a 12-layer RoBERTa$_{\text{BASE}}$ encoder with a 1 layer transformer decoder. We run the benchmark on the TOPv2 source domain evaluation set (17k utterances across 6 domains) on a Tesla V100 GPU, using batch size of 1. In table 11 we report P50, P90, and P99. Our results find that there is less than a 3ms in latency increase in P99 for RAF compared to SpanPointer due to the non-autoregressive nature, despite leveraging our retrieval based model in RAF, and both models are significantly faster than the autoregressive counter part.

B Visualizations

Because RAF encodes utterances and scenarios into a joint embedding space, we can directly visualize this space to further understand our models' inner-workings. We present two case studies: domain development (§B.1) and error analysis (§B.2).

B.1 Domain Development

Figure 3 presents an example of performing domain development on the weather domain. Here, we train RAF with 4 dataset sizes (0 SPIS, 10 SPIS, 25 SPIS, and 1,000 SPIS) to simulate zero-shot, few-shot, low-resource, and high-resource settings, respectively. Each utterance (from the high-resource split) is projected using an utterance encoder and colored according to its gold scenario. Interestingly, the zero-shot setting has multiple, apparent clusters, but the overall performance is poor given many scenarios overlap with each other. The clusters spread further apart as the dataset size increases, suggesting the scenarios become more well-defined.

B.2 Error Analysis

While we have demonstrated how RAF refines the utterance-scenario space as we increase dataset size, we now dive deeper into how each space can be used to further analyze domain semantics. In figure 4, using our high-resource-trained RAF model, we create multiple scenario spaces: each utterance is projected using an utterance encoder and colored according to its predicted frame. We use these scenario spaces in several debugging exercises:

- **Slot Ambiguity:** In Figure 4 (a), we investigate the scenario `[IN:CREATE_ALARM [SL:ALARM_NAME ] [SL:DATE_TIME ]`. Here, we notice a cluster of predictions with the frame `[IN:CREATE_ALARM [SL:DATE_TIME ]` missing the `[SL:ALARM_NAME ]`. These map to utterances such as "I want to wake up at 7 am" where the annotation has "wake up" is `SL:ALARM_NAME`; however, our model does not identify this. There are other examples, such as "wake me up at 7 am", which are annotated without `SL:ALARM_NAME`, leading to ambiguity of whether or not "wake up" is an alarm name.

- **Incorrect Annotations:** In Figure 4 (b), we investigate the scenario `[IN:PLAY_MUSIC`

	Weather Domain (SPIS)							
	0	10	25	50	100	500	1000	Avg
RAF$_{\text{BASE}}$								
Standard Retrieval + Standard Filling	26.19	75.10	78.74	77.53	79.67	87.91	88.17	73.33
Oracle Retrieval + Standard Filling	**81.68**	**90.43**	**92.13**	**91.97**	**93.35**	**95.74**	**96.27**	**91.65**
Standard Retrieval + Oracle Filling	27.67	77.84	81.70	79.92	81.78	89.88	90.44	75.60

Table 9: Evaluating whether retrieval or filling is the most challenging components of RAF in low-resource settings. We fine-tune several variants of RAF, using either a standard / oracle retriever and a standard / oracle filler, on various SPIS splits of the TOPv2 (Chen et al., 2020b) weather domain. RAF with oracle retrieval achieves the best performance, suggesting utterance→scenario retrieval is the most difficult piece to model.

		Zero-Shot Evaluation					
	en	en→es	en→fr	en→de	en→hi	en→th	Avg
RAF$_{\text{BASE}}$							
Standard Retrieval + Standard Filling	81.1	56.0	54.9	40.1	32.1	30.0	42.6
Oracle Retrieval + Standard Filling	**91.0**	**68.9**	**71.6**	**67.5**	**47.5**	**48.9**	**60.9**
Standard Retrieval + Oracle Filling	83.8	66.5	62.8	45.5	40.2	46.7	52.3

Table 10: Evaluating whether retrieval or filling is the most challenging components of RAF in low-resource settings. See Table 9 for a description of our methodology; we use MTOP (Li et al., 2021) for evaluation instead.

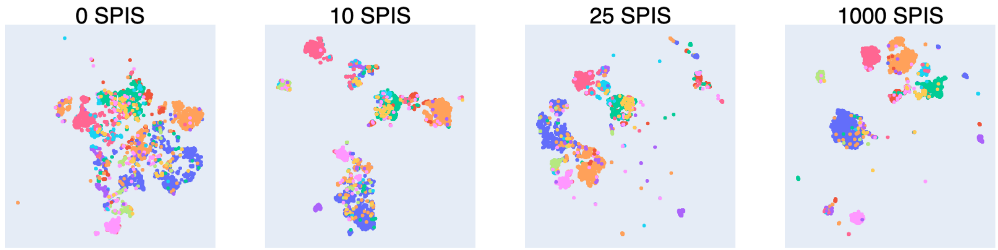


Figure 3: Visualizing the semantic space for the weather domain as the model is trained on more and more data. We visualize the index prior to any training (0 SPIS) to low-resource (10, 25 SPIS) and high resource (1000 SPIS). The graphs are TSNE projections of the utterance vectors used for retrieval color coded by the scenario each utterance belongs to.

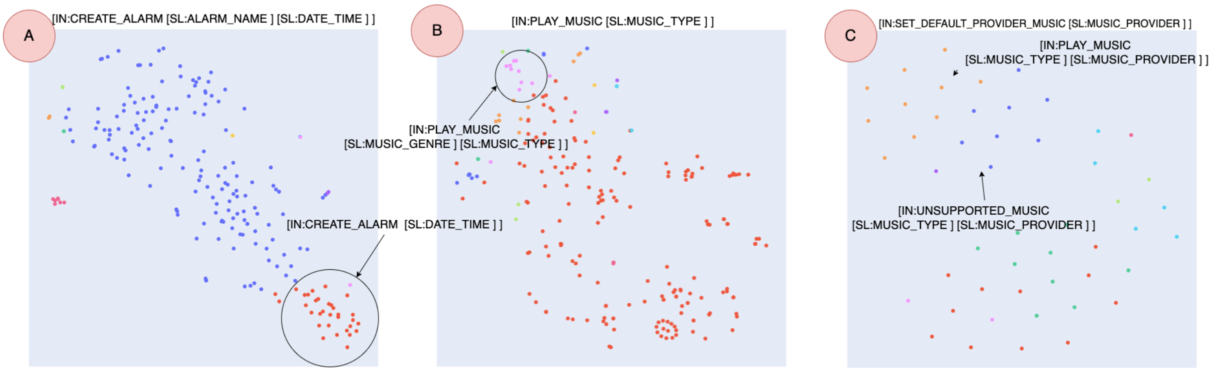


Figure 4: Depicting scenario visualizations, where each is a TSNE projection of utterances belonging to the specified scenario color coded by the predicted scenario. (A) covers the scenario `[IN:CREATE_ALARM [SL:ALARM_NAME ] [SL:DATE_TIME ]` showing a case of ambiguity of alarm names. (B) covers the scenario `[IN:PLAY_MUSIC [SL:MUSIC_TYPE ]` where annotations are missing the slot "music type". (C) covers the scenario `[IN:SET_DEFAULT_PROVIDER_MUSIC [SL:MUSIC_PROVIDER ]` showing how our model requires more support here as the predictions for this scenario span OOD and music domain.

Model	P50	P90	P99
SpanPointer$_{\text{BASE}}$	28.30	29.62	37.78
RAF$_{\text{BASE}}$	28.92	31.40	39.94
Autoregressive$_{\text{BASE}}$	90.29	155.11	228.92

Table 11: Runtime latency (ms) comparison of SpanPointer$_{\text{BASE}}$, RAF$_{\text{BASE}}$, and Autoregressive$_{\text{BASE}}$ using batch size 1 on the TOPv2 source evaluation set.

  `[SL:MUSIC_TYPE ]`. Here, we notice a cluster of predictions with the frame `[IN:PLAY_MUSIC [SL:MUSIC_GENRE ] [SL:MUSIC_TYPE ]` adding the `[SL:MUSIC_GENRE ]`. These map to utterances such as "Play 1960s music" where here the annotation only has "music" as `SL:MUSIC_TYPE`, but our model predicts "1960s" as `SL:MUSIC_GENRE`. We believe this is an incorrect annotation in this cluster.

- **Underfitting:** In Figure 4 (c), we investigate the scenario `[IN:SET_DEFAULT_PROVIDER_MUSIC [SL:MUSIC_PROVIDER ]`. This cluster is highly diverse, consisting of predictions from other music intents (e.g., `IN:PLAY_MUSIC`) and out-of-domain intents (e.g., `IN:UNSUPPORTED_MUSIC`). This scenario may need more data in order to be more properly defined.

C Heuristic negatives

In order to understand the importance of model based hard negatives, we develop a simple heuristic to curate a set of hard negative scenarios for each gold scenario. Our heuristic involves selecting the top-N scenarios that share the top level intent but have the lowest Levenshtein edit distance (Levenshtein, 1966) compared to the gold scenario. The full algorithm is described in Algorithm 1.

In §5 and Table 6 we present the full results depicting the importance of model based negative sampling. We show that while our heuristic improves on top of no hard negatives (in-batch negatives only), it still lags behind model based hard negatives (-1.37%).

D Intrinsic Descriptions

In table 12 we provide brief examples of the various representation functions used in RAF training.

Algorithm 1 Heuristic-based negative sampling via edit distance.

```
procedure EDIT DISTANCE NEGATIVES
    S ← all scenarios
    s* ← current scenario
    S_same intent ← Scenarios with the same top level intent as Si
    heap ← min heap of score and structure
    for $S_i$ ∈ S_same intent do
        score ← LevenshteinDistance(s*, $S_i$)
        heappush(heap, score, $S_i$)
    return heap
```

In tables 13, 14, 15, and 16 we present the intrinsic hand made descriptions used for each intent/slot on TOPv2 (Chen et al., 2020b) and MTOP (Li et al., 2021) respectively. §3.1.3 describes the various scenario representations used in full detail.

E Hyperparameters

In this section we describe the hyper parameters for training our various RAF models.

Architecture Parameters. For our RAF architectures we leverage a shared RoBERTa (Liu et al., 2019) or XLM-R (Conneau et al., 2020) encoder for both the utterance and scenario encoders. We augment each of these encoders with an additional projection layer with a hidden dimension of 768 (base models) or 1024 (large models). For our span pointer decoder we leverage a 1 layer transformer decoder with the same hidden dimension as the respective encoder (1L, 768/1024H, 16/24A).

Optimization Parameters. We train our models with the Adam (Kingma and Ba, 2015) optimizer along with a warmup and linear decay. We train our models across 8 GPUs with 32GB memory each. Additionally we optionally augment our models with the R3F loss (Aghajanyan et al., 2021) based on validation set tuning in each setting. To determine hyperparameters, we conduct hyperparameter sweeps with 56 iterations each. The hyperparameters for the high resource runs on TOPv2 (Chen et al., 2020b) are described in Table 17.

R Type	R Example
`type-only`	`[ intent [ slot ] ]`
`automatic-span`	`[ add time timer [ measurement unit ] ]`
`automatic-type-span`	`[ intent | add time timer [ slot | measurement unit ] ]`
`automatic-type-span-exs`	`[ intent | add time timer [ slot | measurement unit | sec / min / hr ] ]`
`curated-span`	`[ add time to timer [ unit of measurement ] ]`
`curated-type-span`	`[ intent | add time to timer [ slot | unit of measurement ] ]`
`curated-type-span-exs`	`[ intent | add time to timer [ slot | unit of measurement | sec / min / hr ] ]`

Table 12: List of intrinsic representations used for encoding scenarios in RAF.

Intent Token	Description
IN:ADD_TIME_TIMER	add time to timer
IN:ADD_TO_PLAYLIST_MUSIC	add music to playlist
IN:CANCEL_MESSAGE	cancel message
IN:CREATE_ALARM	create alarm
IN:CREATE_PLAYLIST_MUSIC	create playlist music
IN:CREATE_REMINDER	create reminder
IN:CREATE_TIMER	create timer
IN:DELETE_ALARM	delete alarm
IN:DELETE_REMINDER	delete reminder
IN:DELETE_TIMER	delete timer
IN:DISLIKE_MUSIC	dislike music
IN:GET_ALARM	get alarm
IN:GET_BIRTHDAY	get birthday
IN:GET_CONTACT	get contact
IN:GET_DIRECTIONS	get directions
IN:GET_DISTANCE	get distance between locations
IN:GET_ESTIMATED_ARRIVAL	get estimated arrival time
IN:GET_ESTIMATED_DEPARTURE	get estimated departure time
IN:GET_ESTIMATED_DURATION	get estimated duration of travel
IN:GET_EVENT	get event
IN:GET_EVENT_ATTENDEE	get event attendee
IN:GET_EVENT_ATTENDEE_AMOUNT	get amount of event attendees
IN:GET_EVENT_ORGANIZER	get organizer of event
IN:GET_INFO_CONTACT	get info of contact
IN:GET_INFO_ROAD_CONDITION	get info of road condition
IN:GET_INFO_ROUTE	get info of route
IN:GET_INFO_TRAFFIC	get info of traffic
IN:GET_LOCATION	get location
IN:GET_LOCATION_HOME	get location of my home
IN:GET_LOCATION_HOMETOWN	get location of my hometown
IN:GET_LOCATION_SCHOOL	get location of school
IN:GET_LOCATION_WORK	get location of work
IN:GET_MESSAGE	get message
IN:GET_RECURRING_DATE_TIME	get recurring date or time
IN:GET_REMINDER	get reminder
IN:GET_REMINDER_AMOUNT	get amount of reminders
IN:GET_REMINDER_DATE_TIME	get date or time of reminder
IN:GET_REMINDER_LOCATION	get location of reminder
IN:GET_SUNRISE	get info of sunrise
IN:GET_SUNSET	get info of sunset
IN:GET_TIME	get time
IN:GET_TIMER	get timer
IN:GET_TODO	get todo item
IN:GET_WEATHER	get weather
IN:HELP_REMINDER	get help reminder
IN:IGNORE_MESSAGE	ignore message
IN:LIKE_MUSIC	like music
IN:LOOP_MUSIC	loop music
IN:NEGATION	negate
IN:PAUSE_MUSIC	pause music
IN:PAUSE_TIMER	pause timer
IN:PLAY_MUSIC	play music
IN:PREVIOUS_TRACK_MUSIC	play previous music track
IN:REACT_MESSAGE	react to message
IN:REMOVE_FROM_PLAYLIST_MUSIC	remove from music playlist
IN:REPLAY_MUSIC	replay music
IN:PREVIOUS_TRACK_MUSIC	play previous music track
IN:REACT_MESSAGE	react to message
IN:REMOVE_FROM_PLAYLIST_MUSIC	remove from music playlist
IN:REPLAY_MUSIC	replay music
IN:REPLY_MESSAGE	reply to message
IN:RESTART_TIMER	restart timer
IN:RESUME_TIMER	resume timer
IN:SELECT_ITEM	select item
IN:SEND_MESSAGE	send message
IN:SEND_TEXT_MESSAGE	send text message
IN:SET_DEFAULT_PROVIDER_MUSIC	set default music provider
IN:SILENCE_ALARM	silence alarm
IN:SKIP_TRACK_MUSIC	skip music track
IN:SNOOZE_ALARM	snooze alarm
IN:START_SHUFFLE_MUSIC	start shuffling music
IN:STOP_MUSIC	stop music
IN:SUBTRACT_TIME_TIMER	subtract time from timer
IN:UNSUPPORTED_ALARM	unsupported alarm request
IN:UNSUPPORTED_EVENT	unsupported event request
IN:UNSUPPORTED_MESSAGING	unsupported messaging request
IN:UNSUPPORTED_MUSIC	unsupported music request
IN:UNSUPPORTED_NAVIGATION	unsupported navigation request
IN:UNSUPPORTED_TIMER	unsupported timer request
IN:UNSUPPORTED_WEATHER	unsupported weather request
IN:UPDATE_ALARM	update alarm
IN:UPDATE_DIRECTIONS	update directions
IN:UPDATE_REMINDER	update reminder
IN:UPDATE_REMINDER_DATE_TIMER	update date time of reminder
IN:UPDATE_REMINDER_TODO	update todo of reminder
IN:UPDATE_TIMER	update timer

Table 13: List of intrinsic handmade descriptions for intents in TOPv2 (Chen et al., 2020b).

Slot Token	Description
SL:AGE	age of person
SL:ALARM_NAME	alarm name
SL:AMOUNT	amount
SL:ATTENDEE	attendee
SL:ATTENDEE_ADDED	attendee to be added
SL:ATTENDEE_EVENT	attendee of event
SL:ATTENDEE_REMOVED	attendee to be removed
SL:ATTRIBUTE_EVENT	attribute of event
SL:BIRTHDAY	birthday
SL:CATEGORY_EVENT	category of event
SL:CATEGORY_LOCATION	category of location
SL:CONTACT	contact
SL:CONTACT_RELATED	contact related
SL:CONTENT_EMOJI	content text with emoji
SL:CONTEnT_EXACT	content text
SL:DATE_TIME	date or time
SL:DATE_TIME_ARRIVAL	date or time of arrival
SL:DATE_TIME_BIRTHDAY	date or time of birthday
SL:DATE_TIME_DEPARTURE	date or time of departure
SL:DATE_TIME_NEW	new date or time
SL:DATE_TIME_RECURRING	recurring date or time
SL:DESTINATION	travel destination
SL:DURATION	duration
SL:FREQUENCY	frequency
SL:GROUP	group
SL:JOB	job
SL:LOCATION	location
SL:LOCATION_CURRENT	current location
SL:LOCATION_HOME	location of my home
SL:LOCATION_MODIFIER	location modifier
SL:LOCATION_USER	location of user
SL:LOCATION_WORK	location of work
SL:MEASUREMENT_UNIT	unit of measurement
SL:METHOD_RETRIEVAL_REMINDER	method of retrieving reminder
SL:METHOD_TIMER	method of timer
SL:METHOD_TRAVEL	method of traveling
SL:MUSIC_ALBUM_TITLE	title of music album
SL:MUSIC_ARIST_NAME	name of music artist
SL:MUSIC_GENRE	genre of music
SL:MUSIC_PLAYLIST_TITLE	title of music playlist
SL:MUSIC_PROVIDER_NAME	name of music provider
SL:MUSIC_RADIO_ID	id of music radio
SL:MUSIC_TRACK_TITLE	title of music track
SL:MUSIC_TYPE	type of music
SL:MUTUAL_EMPLOYER	mutual employer
SL:MUTUAL_LOCATION	mutual location
SL:MUTUAL_SCHOOL	mutual school
SL:NAME_APP	name of app
SL:NAME_EVENT	name of event
SL:OBSTRUCTION_AVOID	obstruction to avoid
SL:ORDINAL	ordinal
SL:ORGANIZER_EVENT	obstruction to avoid
SL:ORDINAL	ordinal
SL:ORGANIZER_EVENT	organizer of event
SL:PATH	path
SL:PATH_AVOID	path to avoid
SL:PERIOD	time period
SL:PERSON_REMINDED	person to be reminded
SL:PERSON_REMINDED_ADDED	added person to be reminded
SL:PERSON_REMINDED_REMOVED	removed person to be reminded
SL:POINT_ON_MAP	point on map
SL:RECIPIENT	message recipient
SL:RECURRING_DATE_TIME	recurring date or time
SL:RECURRING_DATE_TIME_NEW	new recurring date or time
SL:RESOURCE	resource
SL:ROAD_CONDITION	road condition
SL:ROAD_CONDITION_AVOID	road condition to avoid
SL:SEARCH_RADIUS	search radius
SL:SENDER	message sender
SL:SOURCE	travel source
SL:TAG_MESSGE	tag of message
SL:TIMER_NAME	timer name
SL:TIME_ZONE	time zone
SL:TODO	todo item
SL:TODO_NEW	new todo item
SL:TYPE_CONTACT	contact type
SL:TYPE_CONTENT	content type
SL:TYPE_INFO	info type
SL:TYPE_REACTION	reaction type
SL:TYPE_RELATION	relation type
SL:UNIT_DISTANCE	unit of distance
SL:WAYPOINT	waypoint
SL:WAYPOINT_ADDED	waypoint to be added
SL:WAYPOINT_AVOID	waypoint to avoid
SL:WEATHER_ATTRIBUTE	weather attribute
SL:WEATHER_TEMPERATURE_UNIT	unit of temperature

Table 14: List of intrinsic handmade descriptions for slots in TOPv2 (Chen et al., 2020b).

Intent Token	Description
IN:FOLLOW_MUSIC	follow music
IN:GET_JOB	get job
IN:GET_GENDER	get gender
IN:GET_UNDERGRAD	get undergrad education
IN:GET_MAJOR	get college major
IN:DELETE_PLAYLIST_MUSIC	delete playlist
IN:GET_EDUCATION_DEGREE	get education degree of person
IN:GET_AGE	get age of person
IN:DISPREFER	dislike item
IN:RESUME_MUSIC	resume music
IN:QUESTION_MUSIC	question about music
IN:CREATE_CALL	create a caoo
IN:GET_AIRQUALITY	get airquality
IN:GET_CALL_CONTACT	get contact for caller
IN:SET_UNAVAILABLE	set status to unavailable
IN:END_CALL	end call
IN:STOP_SHUFFLE_MUSIC	stop shuffle of music
IN:PREFER	prefer item
IN:GET_LANGUAGE	get language
IN:SET_AVAILABLE	set available
IN:GET_GROUP	get group
IN:ANSWER_CALL	answer call
IN:GET_CONTACT_METHOD	get method to contact
IN:UPDATE_METHOD_CALL	update method of call
IN:GET_ATTENDEE_EVENT	get attendee for event
IN:UPDATE_CALL	update call
IN:GET_LIFE_EVENT	get life event
IN:REPEAT_ALL_MUSIC	repeat all music
IN:GET_EDUCATION_TIME	get education time
IN:QUESTION_NEWS	question about news
IN:GET_EMPLOYER	get employer
IN:IGNORE_CALL	ignore call
IN:REPEAT_ALL_OFF_MUSIC	turn of repeat
IN:UNLOOP_MUSIC	turn loop off
IN:SET_DEFAULT_PROVIDER_CALLING	set default provider for calling
IN:GET_AVAILABILITY	get avalability of contact
IN:HOLD_CALL	hold call
IN:GET_LIFE_EVENT_TIME	get time of life event
IN:SHARE_EVENT	share event
IN:CANCEL_CALL	cancel call
IN:SET_RSVP_YES	set rsvp to yes
IN:PLAY_MEDIA	play media
IN:GET_TRACK_INFO_MUSIC	get information about the current track
IN:GET_DATE_TIME_EVENT	get the date time of the event
IN:SET_RSVP_NO	set rsvp to no
IN:MERGE_CALL	marge call
IN:UPDATE_REMINDER_LOCATION	update the location of the reminder
IN:GET_MUTUAL_FRIENDS	get mutual friends
IN:GET_MESSAGE_CONTACT	get information about message contact
IN:GET_LYRICS_MUSIC	get lyrics about the song
IN:GET_INFO_RECIPES	get information about recipe
IN:GET_DETAILS_NEWS	get news details
IN:GET_EMPLOYMENT_TIME	get employment time
IN:GET_RECIPES	get a recipe
IN:GET_CALL	get call
IN:GET_CALL_TIME	get time of the call
IN:GET_CATEGORY_EVENT	get the category of the event
IN:RESUME_CALL	resume the call
IN:IS_TRUE_RECIPES	ask question about recipes
IN:SET_RSVP_INTERESTED	set rsvp to interested
IN:GET_STORIES_NEWS	get news stories
IN:SWITCH_CALL	switch call
IN:REWIND_MUSIC	rewind the song
IN:FAST_FORWARD_MUSIC	forward the song

Table 15: List of intrinsic handmade descriptions for intents in MTOP (Li et al., 2021).

Slot Token	Description
SL:GENDER	gender of person
SL:RECIPES_TIME_PREPARATION	time to prepare recipe
SL:RECIPES_EXCLUDED_INGREDIENT	exclude ingredient for recipe
SL:USER_ATTENDEE_EVENT	attendee of event
SL:MAJOR	major
SL:RECIPES_TYPE	type of recipe
SL:SCHOOL	school
SL:TITLE_EVENT	title of event
SL:MUSIC_ALBUM_MODIFIER	type of album
SL:RECIPES_DISH	recipe dish
SL:NEWS_TYPE	type of news
SL:RECIPES_SOURCE	source of recipe
SL:RECIPES_DIET	diet of recipe
SL:RECIPES_UNIT_NUTRITION	nutrition unit of recipe
SL:MUSIC_REWIND_TIME	time to rewind music
SL:RECIPES_TYPE_NUTRITION	nutrition type of recipe
SL:CONTACT_METHOD	method to contact
SL:SIMILARITY	similarity
SL:PHONE_NUMBER	phone number
SL:NEWS_CATEGORY	category of news
SL:RECIPES_INCLUDED_INGREDIENT	ingredient in recipe
SL:EDUCATION_DEGREE	education degree
SL:RECIPES_RATING	rating of recipe
SL:CONTACT_REMOVED	removed contact
SL:NEWS_REFERENCE	news reference
SL:METHOD_RECIPES	method of recipe
SL:LIFE_EVENT	life event
SL:RECIPES_MEAL	recipe meal
SL:NEWS_TOPIC	news topic
SL:RECIPES_ATTRIBUTE	recipe attribute
SL:EMPLOYER	employer
SL:RECIPES_COOKING_METHOD	cooking method of recipe
SL:RECIPES_CUISINE	cuisine of recipe
SL:MUSIC_PLAYLIST_MODIFIER	music playlist modifier
SL:RECIPES_QUALIFIER_NUTRITION	nutrition qualifier of recipe
SL:METHOD_MESSAGE	method to send message
SL:RECIPES_UNIT_MEASUREMENT	unit of measurement in recipe
SL:CONTACT_ADDED	added contact
SL:NEWS_SOURCE	news source

Table 16: List of intrinsic handmade descriptions for slots in MTOP (Li et al., 2021).

Parameter	RAF Models		
	RoBERTa$_{\text{BASE}}$	RoBERTa$_{\text{LARGE}}$	XLM-R$_{\text{BASE}}$
Params	134M	372M	288M
Epochs		40	
Optimizer		Adam	
Weight Decay		0.01	
ϵ		1e-8	
Warmup Period (steps)		1000	
Learning Rate Scheduler		Linear Decay	
Learning Rate	0.00002	0.00003	0.00002
Batch Size	40	12	16
Sampled Negatives	1	1	1
$\beta\mathcal{L}_{\text{Retrieval}}$	2.69	4	2.69
$\mathcal{L}_{\text{filling}}$Label Smoothing Penalty		0.2	
# GPU		8	
GPU Memory	32GB	32GB	32GB

Table 17: Hyperparameter values for RAF architectures.

Document Flattening: Beyond Concatenating Context for Document-Level Neural Machine Translation

Minghao Wu♡ **George Foster**♠ **Lizhen Qu**♡ **Gholamreza Haffari**♡
♡Monash University ♠Google Research
`{firstname.lastname}@monash.edu` `fosterg@google.com`

Abstract

Existing work in document-level neural machine translation commonly concatenates several consecutive sentences as a *pseudo-document*, and then learns inter-sentential dependencies. This strategy limits the model's ability to leverage information from distant context. We overcome this limitation with a novel Document Flattening (DocFlat) technique that integrates Flat-Batch Attention (FBA) and Neural Context Gate (NCG) into Transformer model to utilize information beyond the pseudo-document boundaries. FBA allows the model to attend to all the positions in the batch and learns the relationships between positions explicitly and NCG identifies the useful information from the distant context. We conduct comprehensive experiments and analyses on three benchmark datasets for English-German translation, and validate the effectiveness of two variants of DocFlat. Empirical results show that our approach outperforms strong baselines with statistical significance on BLEU, COMET and accuracy on the contrastive test set. The analyses highlight that DocFlat is highly effective in capturing the long-range information.

1 Introduction

Remarkable progress has been made in neural machine translation (NMT) (Sutskever et al., 2014; Vaswani et al., 2017; Chen et al., 2018), yet human translation still clearly outperforms NMT at the document level (Läubli et al., 2018; Freitag et al., 2021), because current sentence-level NMT systems ignore the inter-sentential relationships. To narrow this gap, numerous document-level NMT (DocNMT) approaches have been proposed in recent years to improve the context awareness by incorporating the contextual information during the translation (Tiedemann and Scherrer, 2017; Maruf and Haffari, 2018; Wong et al., 2020).

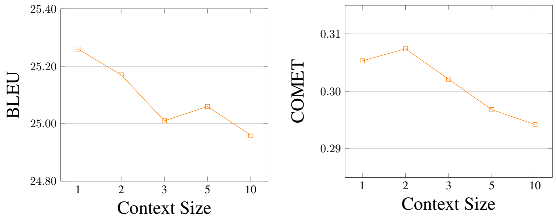


Figure 1: The change of BLEU (left) and COMET (right) given by Doc2Doc on `TED` with regard to the context size of *pseudo-document* (in sentences) based on the experimental setup described in Section 4.1.

Existing DocNMT systems commonly concatenate several consecutive sentences to form a *pseudo-document*, instead of processing the *entire document* (Zhang et al., 2018b; Voita et al., 2019; Junczys-Dowmunt, 2019; Fernandes et al., 2021). One typical pseudo-document contains the *current sentence* to be translated and the *surrronding context*. Intuitively, larger context should result in better performance. In our preliminary study, the model performance does not always grow as the context size increases as shown in Figure 1. Liu et al. (2020) and Bao et al. (2021) also observe that Transformer's performance declines with longer inputs. We refer to this phenomenon as the *quality saturation problem* (Glaser and Strauss, 1967). Therefore, such formation of pseudo-document limits the DocNMT systems to leverage the information from a relatively small context. Consequently, once the entire original document is segmented into several pseudo-documents for reducing the sequence length, the information out of the pseudo-document's scope is no longer accessible to the current sentence. Therefore, a natural research question to ask is that, *is there a more effective way to model the parallel documents in DocNMT?*

In this work, we seek DocNMT approaches that could better expand the context scope and improve the corresponding translation performance. Instead of directly training DocNMT system on the entire document, we propose to store the document as multiple pseudo-documents in a single batch and

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 448–462
May 2-6, 2023 ©2023 Association for Computational Linguistics

optimize the DocNMT models by leveraging the inter-pseudo-document relationships at the batch level. Inspired by Kossen et al. (2021), we propose a Document Flattening (DocFlat) technique that integrates Flat-Batch Attention (FBA) and Neural Context Gate (NCG) into the Transformer model (Vaswani et al., 2017). FBA flattens all the current sentences in the batch with the original order into a sequence along the temporal dimension. It then applies the attention mechanism to the flattened sequence. The goal of this design is to preserve the linguistic structure of documents and expand the scope of context by explicitly learning the pseudo-document relationships. As there is both supportive and noisy information in the longer context, we introduce NCG, a simple feed-forward network, to identify the usefulness of contextual information and filter out the noise. With the combination of FBA and NCG, DocFlat effectively captures the information in the distant context. To the best of our knowledge, Morishita et al. (2021) propose mini-batch embedding (MBE), which is the only close work to ours. They compute the average representation for all the source tokens in the batch and prepend it to the source and target pseudo-documents. The compressed representation ignores the linguistic structure of documents, providing limited contextual information.

Our contributions are summarized as follows. Firstly, we propose a novel approach DocFlat that allows the model to attend the content beyond the pseudo-document boundaries using FBA and NCG. Secondly, we demonstrate that DocFlat outperforms strong baselines with statistical significance, in terms of BLEU, COMET and accuracy on the contrastive test set, on three DocNMT benchmark datasets, including `TED`, `News Commentary` and `Europarl`. Thirdly, we conduct comprehensive analyses to understand the effectiveness of DocFlat. The analyses highlight that DocFlat is highly effective in capturing the distant context.

2 Preliminaries

Sentence-level NMT (SentNMT) The sentence-level NMT model neglects the inter-sentential dependencies between the current sentence and its context. Its probability of translation is defined as:

$$P(\boldsymbol{y}_i|\boldsymbol{x}_i) = \prod_{t=1}^{d} P(y_{i,t}|\boldsymbol{y}_{i,<t}, \boldsymbol{x}_i), \quad (1)$$

where $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are the i-th source and target training sentence, $y_{i,t}$ denotes the t-th token in $\boldsymbol{y}_i$ and d is the sentence length of $\boldsymbol{y}_i$.

Document-level NMT (DocNMT) Given a document pair $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{M}$ where we denote the aligned sentence pair as $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ and M is the length of document in sentences, the i-th pseudo-document pair $\boldsymbol{X}_i$ and $\boldsymbol{Y}_i$ can be defined as:

$$\begin{aligned} \boldsymbol{X}_i &= \text{Concat}([\boldsymbol{x}_{i-c^-}, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_{i+c^+}]), \\ \boldsymbol{Y}_i &= \text{Concat}([\boldsymbol{y}_{i-c^-}, \ldots, \boldsymbol{y}_i, \ldots, \boldsymbol{y}_{i+c^+}]), \end{aligned} \quad (2)$$

where c^- is the context size before the current sentence and c^+ is the context size after the current sentence. The translation probability of target *current* sentence $\boldsymbol{y}_i$ in the target pseudo-document $\boldsymbol{Y}_i$ given the source pseudo-document $\boldsymbol{X}_i$ in DocNMT can be written as:

$$P(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{C}_{-i}) = \prod_{t=1}^{d} P(y_{i,t}|\boldsymbol{y}_{i,<t}, \boldsymbol{x}_i, \boldsymbol{C}_{-i}), \quad (3)$$

where $\boldsymbol{C}_{-i}$ is the collection of all the sentences in the pseudo-document pair except $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, and $\boldsymbol{x}_i$ is the source *current* sentence. We do not consider the context after the current sentence in this work, so c^+ is 0.

3 Document Flattening

In this section, we firstly describe the overview of DocFlat (Section 3.1). We then introduce DocFlat's core components, Flat-Batch Attention (FBA; Section 3.2) and Neural Context Gate (NCG; Section 3.3). Finally, we discuss the practical considerations (Section 3.4 and Section 3.5) of DocFlat along with a concrete example.

3.1 Overview of DocFlat

We present the overall architecture of DocFlat in Figure 2. Given a sequence-to-sequence Transformer with L encoder layers and L decoder layers, we apply the FBA and NCG to the input word embeddings with the residual connection and Layer Normalization, instead of directly feeding the embeddings into either encoder or decoder. DocFlat's translation probability of the i-th target current sentence $\boldsymbol{y}_i$ of the original document in the i-th target pseudo-document $\boldsymbol{Y}_i$ given the i-th source pseudo-document $\boldsymbol{X}_i$ in the batch

$\boldsymbol{B} = \{(\boldsymbol{X}_j, \boldsymbol{Y}_j)\}_{j=1}^{n}$, where n is the batch size, is defined as:

$$P(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{C}_{-i}, \boldsymbol{B}_{-i}) = \prod_{t=1}^{d} P(y_{i,t}|\boldsymbol{y}_{i,<t}, \boldsymbol{x}_i, \boldsymbol{C}_{-i}, \boldsymbol{B}_{-i}), \tag{4}$$

where $\boldsymbol{C}_{-i}$ is defined as in Equation 3 and $\boldsymbol{B}_{-i}$ is the collection of all the current sentences in the batch except $(\boldsymbol{x}_i, \boldsymbol{y}_i)$. We categorize the context for the current sentence into two groups, the global context (GC) from other pseudo-documents $\boldsymbol{B}_{-i}$ and the local context (LC) from its own pseudo-document $\boldsymbol{C}_{-i}$ (See Figure 3).

3.2 FLAT-BATCH ATTENTION

Multi-Head Self-Attention (MHSA) Scaled dot-product attention is the core mechanism of Transformer model with the inputs of query $\boldsymbol{Q}$, key $\boldsymbol{K}$ and $\boldsymbol{V}$ (Vaswani et al., 2017). The attention mechanism computes the attention weights by comparing queries $\boldsymbol{Q}$ with keys $\boldsymbol{K}$ and then updates the representations of queries by computing the weighted sum of values $\boldsymbol{V}$ with the attention weights, which is described as follows:

$$\text{Attn}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{e}})\boldsymbol{V}, \tag{5}$$

where e is the hidden state dimension. Multi-head self-attention (MHSA) then allows the model to jointly attend to information from different hidden subspaces by concatenating a sequence of independent attention heads as follows:

$$\text{MHSA}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_k), \tag{6}$$

where head_j is the scaled dot-product attention in Equation 5 with independent parameters and $j \in \{1, \ldots, k\}$ for each head j.

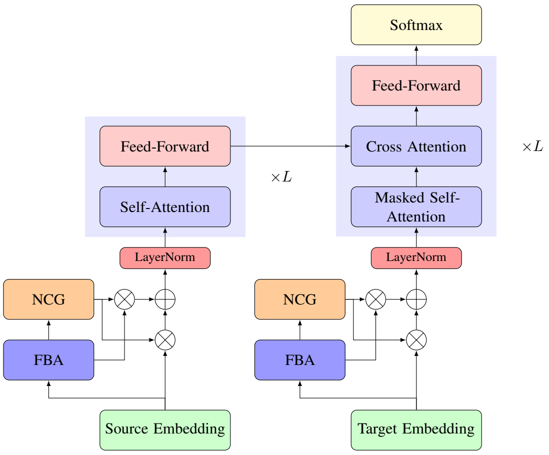


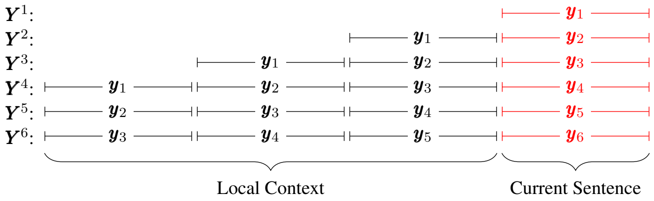
Figure 2: The model architecture of DOCFLAT. $\otimes$ denotes the element-wise multiplication. $\oplus$ denotes the element-wise addition. More details are in Section 3.

FLAT-BATCH ATTENTION (FBA) To leverage the contextual information beyond the pseudo-document boundaries, we propose FLAT-BATCH ATTENTION (FBA). It explicitly transforms the stacked instances in the batch to a single flattened sequence of tokens as shown in Figure 3. Given a batch of hidden representations $\boldsymbol{H} \in \mathbb{R}^{n \times d \times e}$ consisting of n instances padded to the length of d with the hidden dimension of e, FBA operates as follows:

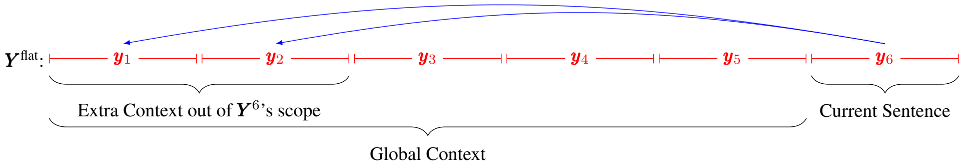
$$\begin{aligned}
\hat{\boldsymbol{H}}_{\text{flat}} &= \text{Flatten}(\boldsymbol{H}) \in \mathbb{R}^{(n \times d) \times e}, \\
\hat{\boldsymbol{H}}_{\text{mhsa}} &= \text{MHSA}(\hat{\boldsymbol{H}}_{\text{flat}}, \hat{\boldsymbol{H}}_{\text{flat}}, \hat{\boldsymbol{H}}_{\text{flat}}), \\
\hat{\boldsymbol{H}}_{\text{rsh}} &= \text{Reshape}(\hat{\boldsymbol{H}}_{\text{mhsa}}) \in \mathbb{R}^{n \times d \times e}, \\
\hat{\boldsymbol{H}} &= \text{LN}((1 - \boldsymbol{g}) \otimes \boldsymbol{H} + \boldsymbol{g} \otimes \hat{\boldsymbol{H}}_{\text{rsh}}).
\end{aligned} \tag{7}$$

As shown in Equation 7, we first flatten $\boldsymbol{H} \in \mathbb{R}^{n \times d \times e}$ to $\hat{\boldsymbol{H}}_{\text{flat}} \in \mathbb{R}^{(n \times d) \times e}$, where $(n \times d)$ indicates the flattened sequence length. The $\hat{\boldsymbol{H}}_{\text{flat}}$ is then fed into a MHSA layer and reshaped back to $\hat{\boldsymbol{H}}_{\text{rsh}} \in \mathbb{R}^{n \times d \times e}$. We then add a residual connection with $\boldsymbol{g}$ given by NCG $\boldsymbol{\psi}$ followed by a sigmoid function σ and apply the Layer Normalization (LN; Ba et al., 2016) following the reshape operation. $\otimes$ denotes the element-wise multiplication. We discuss the details of NCG in Section 3.3. Note that FBA at the decoder side is associated with a causal mask to preserve the auto-regressive property. By attending to all the other current sentences in the batch, FBA effectively allows the current sentences to access a much larger context than the self-attention on the pseudo-documents. In addition, this does not increase the input length of each instance, preventing the quality saturation problem as shown in Figure 1.

Complexity Given a Transformer model with L encoder layers and L decoder layers, suppose the average sentence length is n, the pseudo-document contains c consecutive sentences, and the batch size is b. The complexity of self-attention layer in the concatenation-based DOC2DOC is $\mathcal{O}(L(cn)^2)$. The extra complexity introduced by FBA is $\mathcal{O}((bn)^2)$. Lc^2 and b^2 have the same order



(a) An example batch of pseudo-documents $\boldsymbol{B}_{\text{tgt}} = \{\boldsymbol{Y}^1, \boldsymbol{Y}^2, \boldsymbol{Y}^3, \boldsymbol{Y}^4, \boldsymbol{Y}^5, \boldsymbol{Y}^6\}$ at the target side. Each $\boldsymbol{Y}^j$ contains four consecutive sentences and $\boldsymbol{y}_i$ indicates the i-th sentence of the same original document. The segments in red indicate the current sentence of each pseudo-document.



(b) An example of the flattened sequence $\boldsymbol{Y}^{\text{flat}}$ transformed from $\boldsymbol{B}_{\text{tgt}}$ in Figure 3a with FLAT-BATCH ATTENTION. For the current sentence $\boldsymbol{y}_6$, The blue arrows indicate the extra inter-sentential attention for $\boldsymbol{y}_6$ that our approach can model. $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ are the extra context introduced by our approach.

Figure 3: An example batch of pseudo-documents at the target side and its flattened sequence. Another example at the source side can be found at Appendix A.

of magnitude. The batch size b is set to be constant in practice and the self-attention operation in FBA is highly parallelizable, so integrating FBA into Transformer does not significantly increase the computational cost. Empirically, DOCFLAT is only 3% slower in training and 15% slower in inference, compared with DOC2DOC (See Section 4.2).

3.3 NEURAL CONTEXT GATE

The distant context can contain both supportive and noisy information. Supportive information can assist the translation of the current sentence, while the noise may damage the model predictions. To address this issue, we introduce a novel NEURAL CONTEXT GATE (NCG) to automatically identify the context usefulness and control the information flow from the distant context.

In this work, NCG $\boldsymbol{\psi}$ is a single-layer element-wise feed-forward neural network followed by a sigmoid function σ. Given a batch of hidden representations $\boldsymbol{H}$, the operations are defined as follows:

$$\begin{aligned} \boldsymbol{g} &= \sigma(\boldsymbol{\psi}(\text{FBA}(\boldsymbol{H}))), \\ \boldsymbol{H}^o &= (1-\boldsymbol{g}) \otimes \boldsymbol{H} + \boldsymbol{g} \otimes \text{FBA}(\boldsymbol{H}), \end{aligned} \tag{8}$$

where $\boldsymbol{g}$ is the information gate given by NCG $\boldsymbol{\psi}$ and the sigmoid function σ, $\boldsymbol{H}^o$ is output of the residual connection and $\otimes$ denotes the element-wise multiplication. The values of $\boldsymbol{g}$ are continuous, so we denote DOCFLAT with NCG described in Equation 8 as DOCFLAT$_\text{C}$.

However, the continuous gate may result in the noise leakage. In a long document, the noise at different positions may accumulate, even if they are only associated with very small gating values. The accumulated noise can make a substantial negative impact on the model predictions. Hence, we propose the discrete NCG as follows:

$$\begin{aligned} \boldsymbol{g}_\text{D} &= \mathbb{1}_\gamma(\sigma(\boldsymbol{\psi}(\text{FBA}(\boldsymbol{H})))), \\ \boldsymbol{H}^o &= (1-\boldsymbol{g}_\text{D}) \otimes \boldsymbol{H} + \boldsymbol{g}_\text{D} \otimes \text{FBA}(\boldsymbol{H}), \end{aligned} \tag{9}$$

where $\mathbb{1}_\gamma(\cdot)$ is indicator function defined as:

$$\mathbb{1}_\gamma(g) = \begin{cases} 1 & \text{if } g \geq \gamma, \\ 0 & \text{otherwise}, \end{cases} \tag{10}$$

where γ is the threshold for binarizing the gating values. We denote DOCFLAT with the discrete NCG in Equation 9 as DOCFLAT$_\text{D}$ and set $\gamma = 0.5$ in this work. We expect DOCFLAT$_\text{D}$ is more robust against the noise in the context.

3.4 Data Shuffling

DOCFLAT aims to leverage the distant context with FBA by flattening a batch of sequences into a single sequence. As the ordering information among sentences is critical in DocNMT, we do not shuffle the pseudo-documents during the training and inference to preserve the linguistic structure of the original document. For each sentence, we replace the `<BOS>` symbol with its global index i in the document to preserve the ordering information. Figure 3a is an example at the target side to demonstrate how the pseudo-documents in the batch is

	Train	Valid	Test
`TED`	$204.4K/1.7K$	$8.9K/93$	$2.2K/23$
`News`	$242.4K/6.1K$	$2.3K/81$	$3.2K/155$
`Europarl`	$1.8M/117.9K$	$3.8K/240$	$5.5K/360$

Table 1: The number of sentences/documents of each split of the parallel corpora.

organized in this work. Figure 3b demonstrates how FBA flattens a batch of pseudo-documents. Since the pseudo-documents are not shuffled, $\boldsymbol{y}_6$ can attend to $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ which are not in the pseudo-document $\boldsymbol{Y}^6$. We apply the causal mask to FBA at the decoder side for preserving the auto-regressive property. Note that the pseudo-documents in a batch are mostly from the same original document. The batches crossing the document boundaries are relatively rare and have little effect on performance in our preliminary study.

3.5 Inference

We discuss the batch inference of DOCFLAT in this section. At the encoder side, each source current sentence can attend to its own local context (LC) and all other source current sentence as the global context (GC) in the batch during the inference, as it is at the training stage. At the decoder side, all the target current sentences are translated simultaneously, so each target current sentence can attend to its own LC and partially translated target GC. For example, all the target current sentences in Figure 3a are translated simultaneously during the inference. $\boldsymbol{y}_6$ is conditioned on its own LC, $\boldsymbol{y}_3$, $\boldsymbol{y}_4$ and $\boldsymbol{y}_5$, and partially translated target GC. Additionally, we use the batched inference as usual and there is no overlap between batches. For example, the first batch of sentences to be translated is $\{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_b\}$, and the second batch of sentences to be translated is $\{\boldsymbol{y}_{b+1}, \cdots, \boldsymbol{y}_{2b}\}$, where b is the inference batch size. For decoding, we used the iterative decoding method for decoding (Maruf and Haffari, 2018; Maruf et al., 2019). The initial translations of each sentence were generated by a SentNMT model, and then, we translate each sentence using the DOCNMT model with the translations in the first pass as the context.

4 Experiments

4.1 Setup

Datasets We conduct experiments on three benchmark datasets for English-German translation, including the small-scale datasets `TED` (Cettolo et al., 2012) and `News Commentary` (Tiedemann, 2012), and the large-scale dataset `Europarl` (Koehn, 2005). We tokenize the datasets with the Moses (Koehn et al., 2007) and apply BPE (Sennrich et al., 2016b) with $32K$ merges. Data statistics can be found in Table 1. We choose up to 3 previous sentences as the local context for each source and target sentence to form the pseudo-document unless otherwise specified.

Evaluation We report the detokenized BLEU (Papineni et al., 2002) using SacreBLEU (Post, 2018) and the neural-based COMET (Rei et al., 2020) to measure the translation quality.[1] We report the results with inference batch size of 16 and beam size of 5 for all the approaches, unless otherwise specified.

Contrastive Evaluation This evaluation paradigm is proposed to evaluate the contextual awareness of DOCNMT models with an independent test set, where each test example includes one correct translation and several incorrect translations. The model is required to identify the correct translation and its overall performance is measured by micro-average *Accuracy*. In this work, we use the large-scale English-German anaphoric pronoun test set from Müller et al. (2018), containing $12K$ contrastive examples. Given the provided context, the model of interest is required to identify the translation with the correct use of pronoun from *er*, *es* and *sie* in German.

Models All the models in this work are based on the standard Transformer base (Vaswani et al., 2017). Besides the direct comparisons with prior works, we also compare DOCFLAT with several re-implemented baselines, including SENT2SENT (Vaswani et al., 2017), DOC2DOC (Tiedemann and Scherrer, 2017), FLATTRANS (Ma et al., 2020), MBE (Morishita et al., 2021) and ABD (Kossen et al., 2021). We only apply ABD at the encoder side in this work unless otherwise specified, which is its best-performing setup as shown in Appendix C. We re-produce the results of GTRANS (Bao et al., 2021) with its official code and recommended hyperparameters. The optimization details are in Appendix B.

---

[1] SacreBLEU Signature: `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.2.0` and COMET Signature: `wmt20-comet-da`.

	TED			News			Europarl			UPS
	BLEU	COMET	Acc.	BLEU	COMET	Acc.	BLEU	COMET	Acc.	
Reported										
DocTransformer (Zhang et al., 2018b)	24.00	—	—	23.08	—	—	29.32	—	—	—
HAN (Miculicich et al., 2018)	24.58	—	—	25.03	—	—	28.60	—	—	—
Selective (Maruf et al., 2019)	24.42	—	—	24.84	—	—	29.75	—	—	—
Hybrid (Zheng et al., 2020)	25.10	—	—	24.91	—	—	30.40	—	—	—
Re-produced (standard)										
Sent2Sent	24.78	0.2860	46.48	25.00	0.1993	47.71	31.24	0.5933	53.02	2.57
Doc2Doc	25.01	0.3021	66.99	24.95	0.1990	64.21	31.65	0.5929	78.18	0.86
FlatTrans	24.71	0.2963	45.45	25.05	0.2020	48.54	31.58	0.5954	51.14	0.90
GTrans	25.29	0.3058	—	25.59	0.2097	—	32.33	0.5904	—	—
Re-produced (batch-level)										
MBE	24.75	0.3032	68.12	24.86	0.1969	62.82	31.63	0.5954	77.08	0.64
ABD	24.97	0.3046	68.25	24.33	0.1772	62.52	31.98	0.5955	78.16	0.84
Ours										
DocFlat$_\text{C}$	25.31†	**0.3173**†	70.92†	**25.96**†	**0.2199**†	65.45†	**32.38**†	**0.6020**†	77.65	0.84
DocFlat$_\text{D}$	**25.41**†	0.3101†	**72.04**†	25.38	0.2119†	**66.70**†	32.16†	0.5990†	**79.68**†	0.84

Table 2: BLEU, COMET and accuracy on three benchmark datasets for English-German translation. UPS (↑) indicates updates per second. The best results are highlighted in **bold**. — indicate the result is not available. † indicates the statistical significance at $p = 0.05$ against re-implemented Doc2Doc based on Koehn (2004).

4.2 Main Results

We present the main results in Table 2.

Comparisons with Baselines Compared with all the baselines regardless whether they utilize the batch information or not, both DocFlat$_\text{C}$ and DocFlat$_\text{D}$ substantially outperform these strong baseline approaches, especially in terms of the context awareness (accuracy) which is the main emphasis of this work. For the approaches that utilize the batch information, we observe that MBE and ABD only marginally improves the performance compared with Doc2Doc, suggesting the importance of preserving the linguistic structure in utilizing the batch-level information for DocNMT. We also observe the larger performance gain from DocFlat on small `TED` and `News`, implying DocFlat performs better in the low-resource settings.

DocFlat$_\text{C}$ vs. DocFlat$_\text{D}$ As shown in Table 2, DocFlat$_\text{C}$ and DocFlat$_\text{D}$ demonstrate different strengths: DocFlat$_\text{C}$ mainly improves the translation quality (BLEU and COMET), while DocFlat$_\text{D}$ improves the context awareness (accuracy). In the contrastive evaluation, we have no access to the entire document, so the model predictions are always conditioned on the golden local context (LC) and irrelevant global context (GC). DocFlat$_\text{D}$ outperforms DocFlat$_\text{C}$ in terms of accuracy, suggesting the discrete NCG is more robust against the noise in the context as we expected in Section 3.3. However, DocFlat$_\text{D}$ also aggressively filters out the supportive information in the context as demonstrated on its lower results in BLEU and COMET on `News` and `Europarl`. We believe tuning γ in Equation 9 can fix this issue.

Computational Efficiency As described in Section 3.2, FBA introduces additional computational overhead. We thus evaluate computational efficiency of DocFlat along with the baselines in terms of update per second (UPS) and report the results in Table 2. When the context size of the pseudo-document is the same, our approach DocFlat is almost as fast as the standard Doc2Doc on the identical computational infrastructure (one Tesla A40 GPU) with significant performance gain. Note that GTrans (Bao et al., 2021) does not support FP16 mode, so its UPS is not reported. During the inference, DocFlat is only 15% slower than Doc2Doc.

4.3 Ablation Study

Ablation Study for FBA We conduct the ablation study for FBA and present the results in Table 3. Compared with Doc2Doc, FBA at either side can effectively improve the model performance of DocFlat in terms of BLEU, COMET and accuracy, although both FBAs does not demonstrate orthogonal effectiveness. We also observe that, when the FBA at the decoder side is removed, the contextual awareness (accuracy) is slightly im-

	Enc.	Dec.	BLEU	COMET	Acc.
Doc2Doc	∅	∅	24.86	0.2821	66.98
DocFlat$_{\text{C}}$			25.31	0.3173	70.92
		∅	25.58	0.3114	**71.83**
	∅		**25.70**	**0.3176**	71.06
DocFlat$_{\text{D}}$			25.41	0.3101	72.04
		∅	25.22	0.3112	**73.60**
	∅		**25.71**	**0.3113**	70.48

Table 3: Ablation study for FBA on `TED`. ∅ indicates FBA is removed. The best results for DocFlat$_{\text{C}}$ and DocFlat$_{\text{D}}$ are highlighted in **bold** respectively.

	BLEU	COMET	Acc.
DocFlat$_{\text{C}}$	25.31	**0.3173**	70.92
DocFlat$_{\text{D}}$	**25.41**	0.3101	**72.04**
DocFlat$_{\text{I}}$	25.07	0.3049	69.45

Table 4: Ablation study for NCG on `TED`. DocFlat$_{\text{I}}$ indicates DocFlat with identity mapping in NCG. The best results are highlighted in **bold**.

	GC	LC	BLEU	COMET	Acc.
Doc2Doc	∅	✓	25.01	0.3021	66.99
	✗	✗	—	—	—
DocFlat$_{\text{C}}$	✓	✓	**25.31**	**0.3173**	**70.92**
	✗	✓	24.86	0.2738	66.83
	✗	✗	—	—	—
DocFlat$_{\text{D}}$	✓	✓	**25.41**	**0.3101**	**72.04**
	✗	✓	24.81	0.2037	69.30
	✗	✗	—	—	—

Table 5: Ablation study for data shuffling on `TED`. ✓ indicates the golden context. ✗ indicates the irrelevant context. — indicates the model fails to converge. ∅ indicates Doc2Doc is not associated with GC. The best results for DocFlat$_{\text{C}}$ and DocFlat$_{\text{D}}$ are highlighted in **bold** respectively.

	avg	*er*	*es*	*sie*
Doc2Doc	66.99	56.82	89.20	54.95
MBE	68.12	52.57	89.72	62.07
ABD	68.25	55.30	**90.65**	58.82
DocFlat$_{\text{C}}$	70.92	56.65	89.52	**66.60**
DocFlat$_{\text{D}}$	**72.04**	**60.02**	89.67	66.42

Table 6: Accuracy (in %) on the contrastive test set for `TED` with regard to the anaphoric pronoun types. The best results are highlighted in **bold**.

proved. All these results demonstrate that FBA can effectively leverage the distant context beyond the pseudo-document boundaries.

Ablation Study for NCG We present the ablation study for NCG in Table 4. To probe the utility of NCG, we replace NCG in Equation 7 with identity mapping (He et al., 2016) and denote this variant of DocFlat as DocFlat$_{\text{I}}$. The results from DocFlat$_{\text{I}}$ support our argument that not all the information in the context is useful. Both DocFlat$_{\text{C}}$ and DocFlat$_{\text{D}}$ outperform DocFlat$_{\text{I}}$ on BLEU, COMET and accuracy, which confirms NCG can effectively filter out the noise in the context.

Ablation Study for Data Shuffling To preserve the linguistic structure of the original document, we do not shuffle examples during training. If the examples are shuffled, the predictions of the current sentence are conditioned on the gold local context (LC) and the irrelevant global context (GC). In this section, we investigate how the data shuffling affects DocFlat. We present the results in Table 5. We observe the performance reduction for DocFlat$_{\text{C}}$ and DocFlat$_{\text{D}}$ when the current sentence is conditioned on the gold LC but irrelevant GC. When conditioned on the irrelevant GC, DocFlat even performs worse than Doc2Doc which is free from the irrelevant GC. We also train DocFlat and Doc2Doc with the completely irrelevant context and find out that both models fail to converge. Hence, we confirm that the relatedness between the context and current sentence is of vital importance in DocNMT and DocFlat can effectively leverage the information from the context beyond the scope of the pseudo-documents.

5 Analysis

In this section, we investigate the effectiveness of DocFlat on the contextual awareness and the quality saturation problem. We also demonstrate how the inference batch size affects the model predictions. A visualization of FBA attention map is presented in Appendix D.

Contextual Awareness In English-German translation, the choice of anaphoric pronoun types, including feminine *sie*, neutral *er* and masculine *es*, commonly depends on its context. We present the accuracy with regard to the anaphoric pronoun types given by the selected models trained on `TED` in Table 6. DocFlat$_{\text{D}}$ is the only approach that demonstrates substantial improvements on the neutral *er*. For the feminine *sie*, DocFlat$_{\text{C}}$ and DocFlat$_{\text{D}}$ both outperform Doc2Doc by approximately 12% accuracy. MBE and ABD only improves the accuracy on the feminine *sie* by 7%

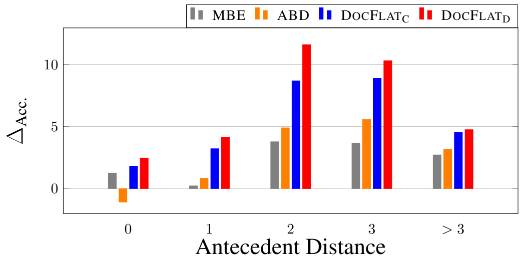


Figure 4: The change of accuracy (in %; Δ_{acc}) given by the selected models against Doc2Doc with regard to the antecedent distance (in sentences) on `TED`.

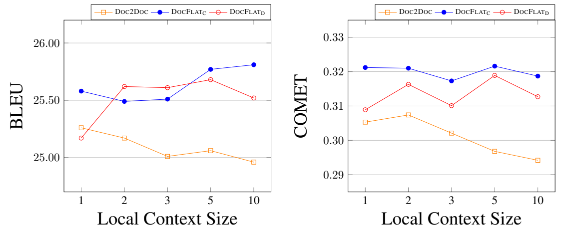


Figure 5: BLEU (left) and COMET (right) against the LC size of pseudo-document (in sentences) by Doc2Doc, DocFlat$_{\mathrm{C}}$ and DocFlat$_{\mathrm{D}}$ on `TED`.

and 4% respectively. We also present the change of accuracy given by the selected models against Doc2Doc with regard to the antecedent distance on `TED` in Figure 4. Compared with Doc2Doc, the approaches that leverage the batch-level information all effectively improves the accuracy on the distant context (antecedent distance ≥ 2). DocFlat$_{\mathrm{C}}$ significantly outperforms Doc2Doc with regard to the accuracy on the distant context by more than 8%, while DocFlat$_{\mathrm{D}}$ outperforms Doc2Doc by more than 10% on the distant context. All these results demonstrate that DocFlat can effectively improve the contextual awareness on the discourse phenomena.

Effect of DocFlat on Quality Saturation Doc2Doc suffers from the *quality saturation problem* as shown in Figure 1. We investigate if DocFlat also suffers from the same problem. We display the results in Figure 5 and observe that DocFlat$_{\mathrm{C}}$ and DocFlat$_{\mathrm{D}}$ perform consistently with regard to the LC size. We conjecture the reasons for this observation from two perspectives. When the LC size is small, the information from GC introduced by FBA complements the missing information in LC. When the LC size is large enough, most information from GC is already covered by LC and FBA functions as a regularizer.

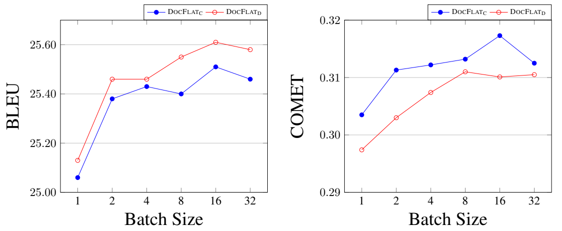


Figure 6: BLEU (left) and COMET (right) against the inference batch size (in sentences) given by DocFlat$_{\mathrm{C}}$ and DocFlat$_{\mathrm{D}}$ on `TED`.

Inference Batch Size At the inference stage, DocFlat is also able to leverage the batch-level information. We visualize how the inference batch size impacts the model performance in Figure 6. Overall, the model performance of DocFlat is positively correlated to the inference batch size. When the batch size is 1, DocFlat$_{\mathrm{C}}$ and DocFlat$_{\mathrm{D}}$ still outperform Doc2Doc, suggesting the FBA can help the model utilize the distant context during training. The performance gain on BLEU and COMET for both DocFlat$_{\mathrm{C}}$ and DocFlat$_{\mathrm{D}}$ diminishes as the inference batch size increases, and we do not observe further improvement when the inference batch size is larger than 16, suggesting the over-distant context is less influential to the predictions of the current sentence.

6 Related Work

Document-Level NMT Numerous document-level NMT approaches have been proposed in recent years. Tiedemann and Scherrer (2017) firstly proposed the simple concatenation-based DocNMT model. Existing works in the document-level NMT widely spread on a variety of research topics, including the model architecture (Miculicich et al., 2018; Maruf et al., 2019; Zhang et al., 2021), training methods (Sun et al., 2022; Lei et al., 2022), evaluation (Bawden et al., 2018; Jiang et al., 2022), etc. Zhang et al. (2018b) incorporate the contextual information using an independent context encoder. Bao et al. (2021) propose group attention that introduce a locality bias to force the model to focus on the recent context. Morishita et al. (2021) compute the average representation of all the source tokens, which is the only close work to ours. Maruf et al. (2021) present a detailed review on DocNMT.

Batch-Level Information Modeling instance relationships in the batch is relatively less explored. Prior works leveraging the instance relationships are mostly from the computer vision area. Ioffe and Szegedy (2015) keep the running mean and variance in the batch to normalize the training and testing instances. Zhang et al. (2018a) linearly combine a random pair of instances to improve the model generalization. Mondal et al. (2021) use graph neural networks to aggregate information from similar images. Hou et al. (2022) propose BATCHFORMER to improve the long-tail recognition by combining different instances. Our work is directly inspired by Kossen et al. (2021) that computes the pairwise similarity among all the batched instances, with distinct motivation. We aim to utilize distant context beyond the pseudo-document boundaries, instead of finding the similar patterns.

7 Conclusion

In this work, we address the limitation of the pseudo-document formation in the DocNMT by utilizing the batch-level information. We propose a novel Document Flattening (DOCFLAT) technique that integrates FLAT-BATCH ATTENTION (FBA) and NEURAL CONTEXT GATE (NCG) into the Transformer model. FBA enables the current sentence to access the information beyond the pseudo-document boundaries and NCG identifies the usefulness of context and controls the information flow. We conduct comprehensive experiments and analyses on three benchmark datasets for English-German translation. We demonstrate that DOCFLAT outperforms several strong baselines with statistical significance. The analyses highlight that DOCFLAT can effectively alleviate the quality saturation problem in DocNMT and capture the long-range information.

8 Limitations

As suggested in Figure 6, the performance of DOCFLAT is positively correlated to the inference batch size. This is because large inference batch size could help DOCFLAT to better utilize distant context within the same inference batch. However, this property of DOCFLAT could become an issue when there are only limited inference computational resources available.

Acknowledgment

This research is supported by the ARC Future Fellowship FT190100039. This work is partly sponsored by the Air Force Research Laboratory and DARPA under agreement number FA8750-19-2-0501. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The authors are grateful to the anonymous reviewers for their helpful comments to improve the manuscript.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Guangsheng Bao, Yue Zhang, Zhiyang Teng, Boxing Chen, and Weihua Luo. 2021. G-transformer for document-level machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3442–3455, Online. Association for Computational Linguistics.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Annual conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, Melbourne, Australia. Association for Computational Linguistics.

Patrick Fernandes, Kayo Yin, Graham Neubig, and André F. T. Martins. 2021. Measuring and increasing context usage in context-aware machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6467–6478, Online. Association for Computational Linguistics.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1019–1027.

BG Glaser and AL Strauss. 1967. The discovery of grounded theory: strategies for qualitative research.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Zhi Hou, Baosheng Yu, and Dacheng Tao. 2022. Batchformer: Learning to explore sample relationships for robust representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7256–7266.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org.

Yuchen Jiang, Tianyu Liu, Shuming Ma, Dongdong Zhang, Jian Yang, Haoyang Huang, Rico Sennrich, Ryan Cotterell, Mrinmaya Sachan, and Ming Zhou. 2022. BlonDe: An automatic evaluation metric for document-level machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1550–1565, Seattle, United States. Association for Computational Linguistics.

Marcin Junczys-Dowmunt. 2019. Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 225–233, Florence, Italy. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Jannik Kossen, Neil Band, Clare Lyle, Aidan N. Gomez, Thomas Rainforth, and Yarin Gal. 2021. Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 28742–28756.

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4796, Brussels, Belgium. Association for Computational Linguistics.

Yikun Lei, Yuqi Ren, and Deyi Xiong. 2022. CoDoNMT: Modeling cohesion devices for document-level neural machine translation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5205–5216, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pretraining for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

António Lopes, M. Amin Farajian, Rachel Bawden, Michael Zhang, and André F. T. Martins. 2020. Document-level neural MT: A systematic comparison. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 225–234, Lisboa, Portugal. European Association for Machine Translation.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020. A simple and effective unified encoder for document-level machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3505–3511, Online. Association for Computational Linguistics.

Sameen Maruf and Gholamreza Haffari. 2018. Document context neural machine translation with memory networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1275–1284, Melbourne, Australia. Association for Computational Linguistics.

Sameen Maruf, André F. T. Martins, and Gholamreza Haffari. 2019. Selective attention for context-aware neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3092–3102, Minneapolis, Minnesota. Association for Computational Linguistics.

Sameen Maruf, Fahimeh Saleh, and Gholamreza Haffari. 2021. A survey on document-level neural machine translation: Methods and evaluation. *ACM Comput. Surv.*, 54(2):45:1–45:36.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954, Brussels, Belgium. Association for Computational Linguistics.

Arnab Kumar Mondal, Vineet Jain, and Kaleem Siddiqi. 2021. Mini-batch similarity graphs for robust image classification. In *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*, page 194. BMVA Press.

Makoto Morishita, Jun Suzuki, Tomoharu Iwata, and Masaaki Nagata. 2021. Context-aware neural machine translation with mini-batch embedding. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2513–2521, Online. Association for Computational Linguistics.

Mathias Müller, Annette Rios, Elena Voita, and Rico Sennrich. 2018. A large-scale test set for the evaluation of context-aware pronoun translation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 61–72, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 371–376, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zewei Sun, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Shujian Huang, Jiajun Chen, and Lei Li. 2022. Rethinking document-level neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3537–3548, Dublin, Ireland. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

KayYen Wong, Sameen Maruf, and Gholamreza Haffari. 2020. Contextual neural machine translation improves translation of cataphoric pronouns. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5971–5978, Online. Association for Computational Linguistics.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018a. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018b. Improving the transformer translation model with document-level context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542, Brussels, Belgium. Association for Computational Linguistics.

Long Zhang, Tong Zhang, Haibo Zhang, Baosong Yang, Wei Ye, and Shikun Zhang. 2021. Multi-hop transformer for document-level machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3953–3963, Online. Association for Computational Linguistics.

Zaixiang Zheng, Xiang Yue, Shujian Huang, Jiajun Chen, and Alexandra Birch. 2020. Towards making the most of context in neural machine translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3983–3989. ijcai.org.

A A Concrete Example at the Source Side

We present a concrete example at the source side in Figure 7.

B Optimization and Hyperparameters

We use a two-stage training routine following the previous works (Zhang et al., 2018b; Voita et al., 2019; Lopes et al., 2020; Bao et al., 2021).

Stage I We first train a SENT2SENT NMT model. The model is randomly initialized and optimized with Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and the learning rate $\alpha = 5 \times 10^{-4}$. The model is trained with the batch size of $32K$ tokens for both datasets and the dropout rate $p = 0.3$. The batch size of $32K$ tokens is achieved by using the batch size of 4096 tokens and updating the model for every 8 batches. The learning rate schedule is the same as described in Vaswani et al. (2017) with $4K$ warmup steps. We use early stopping on validation loss.

Stage II The document-level models are all fine-tuned from the best SENT2SENT model in the Stage I. With the same learning rate schedule as the Stage I, we set the learning rate $\alpha = 2 \times 10^{-4}$. All the other hyperparameters are identical. Training is early stopped on validation loss, and we average the last 5 checkpoints to report the model performance, following Vaswani et al. (2017). Following Bao et al. (2021), we apply word dropout (Gal and Ghahramani, 2016; Sennrich et al., 2016a) to the inputs with $p = 0.1$.

C ATTENTION BETWEEN DATAPOINTS

We adapt ATTENTION BETWEEN DATAPOINTS (ABD) proposed by Kossen et al. (2021) to the DocNMT. The model architecture is identical to DOCFLAT as shown in Figure 2 with FBA replaced with ABD. Given a batch of hidden representations $\boldsymbol{H} \in \mathbb{R}^{n \times d \times e}$, ABD is defined as follows:

$$\begin{aligned}
\tilde{\boldsymbol{H}}_{\text{avg}} &= \text{AvgPool}(\boldsymbol{H}) \in \mathbb{R}^{n \times 1 \times e}, \\
\tilde{\boldsymbol{H}}_{\text{flat}} &= \text{Flatten}(\tilde{\boldsymbol{H}}_{\text{avg}}) \in \mathbb{R}^{1 \times n \times e}, \\
\tilde{\boldsymbol{H}}_{\text{mhsa}} &= \text{MHSA}(\tilde{\boldsymbol{H}}_{\text{flat}}, \tilde{\boldsymbol{H}}_{\text{flat}}, \tilde{\boldsymbol{H}}_{\text{flat}}) \in \mathbb{R}^{1 \times n \times e}, \\
\tilde{\boldsymbol{H}}_{\text{rsh}} &= \text{Reshape}(\text{Repeat}(\tilde{\boldsymbol{H}}_{\text{mhsa}})) \in \mathbb{R}^{n \times d \times e}, \\
\tilde{\boldsymbol{H}} &= \text{LN}(\boldsymbol{H} + \tilde{\boldsymbol{H}}_{\text{rsh}}) \in \mathbb{R}^{n \times d \times e}.
\end{aligned} \tag{11}$$

There is a noticeable difference in Equation 11 from Equation 7 that we apply the average pooling to the sequence to obtain the instance representation, instead of directly flattening the token representations into a single vector. ABD is originally designed for fixed-length data, and it is non-trivial to apply ABD to the variable-length inputs, and hence, we use the average pooling for simplicity.

	Enc.	Dec.	BLEU	COMET	Acc.
DOC2DOC	✗	✗	25.01	0.3021	66.99
ABD			18.57	-0.1202	66.55
	✗		18.46	-0.1123	66.47
		✗	24.97	0.3046	68.25

Table 7: Preliminary study on the usage of ABD on `TED`. ✗ indicates ABD is removed.

We present the preliminary study of ABD on `TED` in Table 7. When ABD is applied at the decoder side, the model performance is significantly reduced. This observation suggests that the linguistic structure at the target side is of vital importance to DocNMT.

D Visualization of FBA

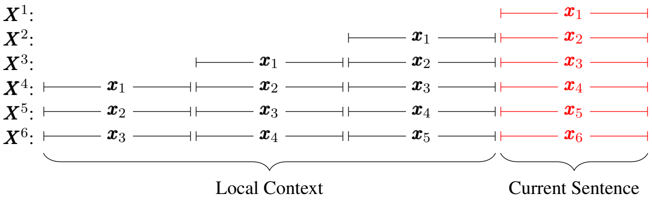
To better understand the behavior of FBA, we visualize the sentence-wise attention map learned by the FBA of $\text{DOCFLAT}_{\text{C}}$ and $\text{DOCFLAT}_{\text{D}}$ at the encoder side in Figure 8. The sample document for producing Figure 8 can be found in Table 8.

It is infeasible to visualize the token-wise attention map for a very long sequence, so we aggregate the token-wise attention scores into the sentence-level. We denote the token-wise attention map for the flattened sequence as $\mathcal{A}$. For each pair of sentences $\boldsymbol{s}_i$ attending to $\boldsymbol{s}_j$, their token-wise attention map is a patch of $\mathcal{A}$, denoted as $\mathcal{A}^p_{ij}$. We aggregate the token-level attention scores in the attention patch $\mathcal{A}^p_{ij}$ into a sentence-level score, as follows:

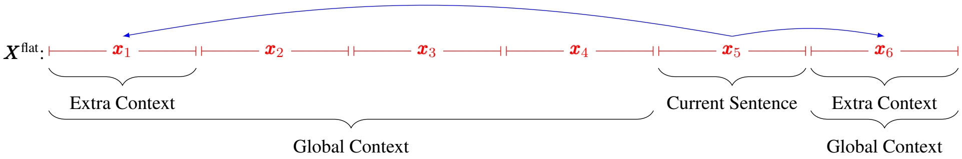
$$\mathcal{A}_{\mathcal{S}}(i,j) = \frac{1}{\left|\mathcal{A}^p_{ij}\right|} \sum \mathcal{A}^p_{ij} \tag{12}$$

where $\left|\mathcal{A}^p_{ij}\right|$ is the size of $\mathcal{A}^p_{ij}$ and $\mathcal{A}_{\mathcal{S}}(i,j)$ is the sentence-level attention score for $\boldsymbol{s}_i$ attending to $\boldsymbol{s}_j$.

The FBA of $\text{DOCFLAT}_{\text{C}}$ at the encoder side considers all the sentences in the document to be equally important, while the one of $\text{DOCFLAT}_{\text{D}}$ approximately splits the whole documents into two parts. As shown by $\text{DOCFLAT}_{\text{D}}$ in Figure 8, sentences in the first half focus more on its neighbors in the same split but those in the second half



(a) An example batch of pseudo-documents $\boldsymbol{B}_\text{src} = \{\boldsymbol{X}^1, \boldsymbol{X}^2, \boldsymbol{X}^3, \boldsymbol{X}^4, \boldsymbol{X}^5, \boldsymbol{X}^6\}$ at the source side. Each $\boldsymbol{X}^j$ contains four consecutive sentences and $\boldsymbol{x}_i$ indicates the i-th sentence of the same original document. The segments in red indicate the current sentence of each pseudo-document.



(b) An example of the flattened sequence $\boldsymbol{X}^\text{flat}$ transformed from $\boldsymbol{B}_\text{tgt}$ with Flat-Batch Attention. For the current sentence $\boldsymbol{x}_5$, The blue arrows indicate the extra inter-sentential attention for $\boldsymbol{x}_5$ that our approach can model. $\boldsymbol{x}_1$ and $\boldsymbol{x}_6$ are the extra context introduced by our approach.

Figure 7: An example batch of pseudo-documents at the source side and its flattened sequence.

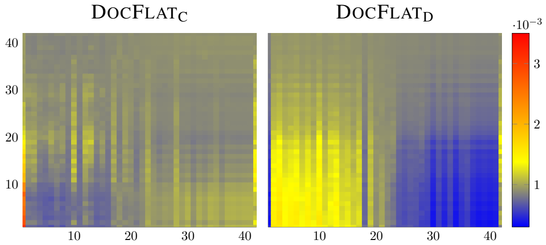


Figure 8: Sentence-wise attention map produced by the FBA of DocFlat$_\text{C}$ and DocFlat$_\text{D}$ at the encoder side. x-axis indicates sentences as the keys of FBA. y-axis indicates sentences as the queries of FBA.

roughly attend to all the sentences in the documents. This observation implies that the latter context is more dependent on the former context.

idx	Context
1	<d>
2	We're at a tipping point in human history, a species poised between gaining the stars and losing the planet we call home.
3	Even in just the past few years, we've greatly expanded our knowledge of how Earth fits within the context of our universe.
4	NASA's Kepler mission has discovered thousands of potential planets around other stars, indicating that Earth is but one of billions of planets in our galaxy.
5	Kepler is a space telescope that measures the subtle dimming of stars as planets pass in front of them, blocking just a little bit of that light from reaching us.
6	Kepler's data reveals planets' sizes as well as their distance from their parent star.
7	Together, this helps us understand whether these planets are small and rocky, like the terrestrial planets in our own Solar System, and also how much light they receive from their parent sun.
8	In turn, this provides clues as to whether these planets that we discover might be habitable or not.
9	Unfortunately, at the same time as we're discovering this treasure trove of potentially habitable worlds, our own planet is sagging under the weight of humanity.
10	2014 was the hottest year on record.
11	Glaciers and sea ice that have been with us for millennia are now disappearing in a matter of decades.
12	These planetary-scale environmental changes that we have set in motion are rapidly outpacing our ability to alter their course.
13	But I'm not a climate scientist, I'm an astronomer.
14	I study planetary habitability as influenced by stars with the hopes of finding the places in the universe where we might discover life beyond our own planet.
15	You could say that I look for choice alien real estate.
16	Now, as somebody who is deeply embedded in the search for life in the universe, I can tell you that the more you look for planets like Earth, the more you appreciate our own planet itself.
17	Each one of these new worlds invites a comparison between the newly discovered planet and the planets we know best: those of our own Solar System.
18	Consider our neighbor, Mars.
19	Mars is small and rocky, and though it's a bit far from the Sun, it might be considered a potentially habitable world if found by a mission like Kepler.
20	Indeed, it's possible that Mars was habitable in the past, and in part, this is why we study Mars so much.
21	Our rovers, like Curiosity, crawl across its surface, scratching for clues as to the origins of life as we know it.
22	Orbiters like the MAVEN mission sample the Martian atmosphere, trying to understand how Mars might have lost its past habitability.
23	Private spaceflight companies now offer not just a short trip to near space but the tantalizing possibility of living our lives on Mars.
24	But though these Martian vistas resemble the deserts of our own home world, places that are tied in our imagination to ideas about pioneering and frontiers, compared to Earth Mars is a pretty terrible place to live.
25	Consider the extent to which we have not colonized the deserts of our own planet, places that are lush by comparison with Mars.
26	Even in the driest, highest places on Earth, the air is sweet and thick with oxygen exhaled from thousands of miles away by our rainforests.
27	I worry – I worry that this excitement about colonizing Mars and other planets carries with it a long, dark shadow: the implication and belief by some that Mars will be there to save us from the self-inflicted destruction of the only truly habitable planet we know of, the Earth.
28	As much as I love interplanetary exploration, I deeply disagree with this idea.
29	There are many excellent reasons to go to Mars, but for anyone to tell you that Mars will be there to back up humanity is like the captain of the Titanic telling you that the real party is happening later on the lifeboats.
30	Thank you.
31	But the goals of interplanetary exploration and planetary preservation are not opposed to one another.
32	No, they're in fact two sides of the same goal: to understand, preserve and improve life into the future.
33	The extreme environments of our own world are alien vistas.
34	They're just closer to home.
35	If we can understand how to create and maintain habitable spaces out of hostile, inhospitable spaces here on Earth, perhaps we can meet the needs of both preserving our own environment and moving beyond it.
36	I leave you with a final thought experiment: Fermi's paradox.
37	Many years ago, the physicist Enrico Fermi asked that, given the fact that our universe has been around for a very long time and we expect that there are many planets within it, we should have found evidence for alien life by now.
38	So where are they?
39	Well, one possible solution to Fermi's paradox is that, as civilizations become technologically advanced enough to consider living amongst the stars, they lose sight of how important it is to safeguard the home worlds that fostered that advancement to begin with.
40	It is hubris to believe that interplanetary colonization alone will save us from ourselves, but planetary preservation and interplanetary exploration can work together.
41	If we truly believe in our ability to bend the hostile environments of Mars for human habitation, then we should be able to surmount the far easier task of preserving the habitability of the Earth.
42	Thank you.

Table 8: The sample document used for producing Figure 8.

Scaling Back-Translation with Domain Text Generation for Sign Language Gloss Translation

Jinhui Ye[1*] **Wenxiang Jiao**[2*] **Xing Wang**[2†] **Zhaopeng Tu**[2]
[1]The Hong Kong University of Science and Technology (Guangzhou)
`jye624@connect.hkust-gz.edu.cn`
[2]Tencent AI Lab
`{joelwxjiao,brightxwang,zptu}@tencent.com`

Abstract

Sign language gloss translation aims to translate the sign glosses into spoken language texts, which is challenging due to the scarcity of labeled gloss-text parallel data. Back-translation (BT), which generates pseudo parallel data by translating in-domain spoken language texts into sign glosses, has been applied to alleviate the data scarcity problem. However, the lack of large-scale high-quality in-domain spoken language text data limits the effect of BT. In this paper, to overcome the limitation, we propose a **P**rompt based domain text **Gen**eration (PGEN) approach to produce the large-scale in-domain spoken language text data. Specifically, PGEN randomly concatenates sentences from the original in-domain spoken language text data as prompts to induce a pre-trained language model (i.e., GPT-2) to generate spoken language texts in similar style. Experimental results on three benchmarks of sign language gloss translation in varied languages demonstrate that BT with spoken language texts generated by PGEN significantly outperforms the compared methods. In addition, as the scale of spoken language texts generated by PGEN increases, the BT technique can achieve further improvements, demonstrating the effectiveness of our approach. We release the code and data for facilitating future research in this field[1].

1 Introduction

Sign language is the dominant form of communication for the deaf and hearing impaired community. Sign language processing has received substantial attention in the last few years and achieved significant progress (Bragg et al., 2019; Yin et al., 2021; Shterionov, 2021; De Coster et al., 2022). Among them, sign language translation (SLT) aims to transform continuous sign language videos into natural spoken language texts (Bungeroth and Ney, 2004; Camgoz et al., 2018). SLT consists of two subtasks: (1) sign language understanding task that recognizes the continued videos to the sign glosses; and (2) spoken language gloss translation (SLGT) task that generates the spoken language text of the given sign glosses. In this work we focus on the second sub-task, i.e., SLGT.

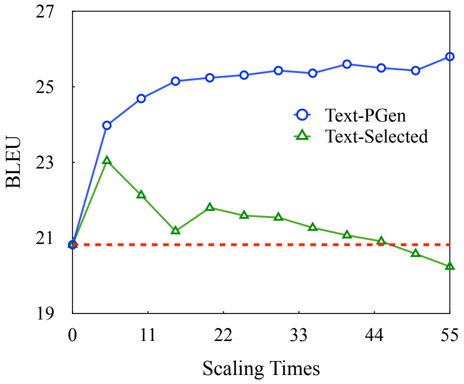


Figure 1: The translation performance of back-translation when scaling the generated spoken language data from 1) our PGEN which uses the large pretrained language model, and 2) selected in-domain data from large-scale spoken language corpus. The red dashed line denotes the baseline model without back-translation. Best viewed in color.

Data scarcity has been considered the major limitation of sign language gloss translation (Moryossef et al., 2021; Zhang and Duh, 2021). To alleviate the data scarcity problem, back-translation (Sennrich et al., 2016), which translates in-domain spoken language texts into sign glosses to construct synthetic parallel data, has been adopted and achieved certain success in SLGT. However, the lack of large-scale high-quality in-domain spoken language text data limits the capability of back-translation for the SLGT task (Zhang and Duh, 2021). The common practice is adopt-

---

[*] Jinhui Ye and Wenxiang Jiao contributed equally to this work. Work was mainly done when Jinhui Ye was interning at Tencent AI Lab.

[†] Xing Wang is the corresponding author.

[1] Code and data are available at `https://github.com/Atrewin/PGen`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 463–476
May 2-6, 2023 ©2023 Association for Computational Linguistics

ing data selection (Axelrod et al., 2011), or data mining (Jiang et al., 2009) approaches to obtain in-domain data. But theses approaches assume the availability of enough in-domain data or expert knowledge (e.g., language background to mine the in-domain data in a specific language), which prevents them from applying to gloss translation task that has broader scenarios (e.g., more languages and domains).

In this work, we propose a **P**rompt-based domain text **Gen**eration (PGen) to generate large-scale high-quality in-domain spoken language text data, motivated by the advances in data augmentation with pretrained language models (PLMs). The main idea is to induce the large PLMs to mimic the style of original spoken language texts with prompt-based learning techniques (Radford et al., 2019; Liu et al., 2021). Our PGen approach is able to generate large-scale in-domain spoken language texts based on the small-scale original monolingual texts and maintains diversity (§2.1). Besides, our approach can be performed without requiring large-scale in-domain data or expert knowledge of the sign language domain and maintain the high quality of generated in-domain spoken language texts. Finally, we employ a sequence-sequence pretrained model (e.g, mT5) to translate spoken language texts generated by PGen into sign glosses and synthesize gloss-text pseudo-parallel data (§2.2).

We conduct extensive analyses of the spoken language texts generated by PGen. We find that the generated and original spoken language texts share a similar word distribution (§3.2). To further verify the effectiveness of PGen, we also conduct back-translation experiments for the SLGT task with large-scale in-domain spoken language texts generated by PGen (§3.3). Experimental results on three widely used benchmark datasets across languages and domains show that back-translation with spoken language texts generated by PGen significantly outperforms the compared methods. Most importantly, as shown in Figure 1, when scaling the spoken language texts generated by PGen in BT approach, the performance of the gloss-to-text translation task can achieve constant improvement while conventional data selection approach failed. The contributions of our work are summarized as follows:

- We propose a novel text generation approach, i.e., PGen, to produce large-scale in-domain spoken language texts which share similar linguistic properties as the original spoken language texts.
- We scale back-translation with the proposed PGen approach and achieve significant and consistent improvements on three benchmark SLGT datasets.
- We release the code and the large-scale synthetic gloss-text datasets produced by the proposed approach to promote the research in sign language gloss translation field.

2 Methodology

The whole framework of this work includes two components, i.e., the PGen method for in-domain text generation and the BT model (i.e., text-to-gloss translation) for constructing synthetic data. We will introduce more details for these two components in this section. For clarity, we provide the definition of SLGT (i.e., gloss-to-text translation task) and notations used throughout the paper as below.

Task Definition. Let X and Y denote the gloss annotations and spoken language texts, and $\mathcal{X}$ and $\mathcal{Y}$ represent the sentence sets of corresponding languages. The dataset of gloss-text pairs can be expressed as $\mathcal{D}_{\text{g2t}} = \{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^{N}$, where $\mathbf{x}^i \in \mathcal{X}$ is the annotations, $\mathbf{y}^i \in \mathcal{Y}$ is the spoken language sentence and N is the number of pairs. Given a sequence of gloss annotations, the task is to output the corresponding fluent and semantically equivalent sentence.

2.1 Prompt-Based Domain Text Generation

We exploit the large PLMs for domain text generation. Large PLMs have been successfully applied for text generation in NLP, such as text classification (Kumar et al., 2020) and medical dialogue summarization (Chintagunta et al., 2021), for two advantages: (1) PLMs are demonstrated to memorize the knowledge of their training data (Carlini et al., 2021), which usually covers different domains. With proper guidance (e.g., prompts), we can export the memorized sentences that belong to the same domain as the sign language text data. (2) Large PLMs are also able to generate abundant new sentences rather than only the memorized sentence in training data (Qiu et al., 2020). Therefore, we propose the **P**rompt-based domain text **Gen**eration (PGen) approach to produce large-scale in-domain spoken language text

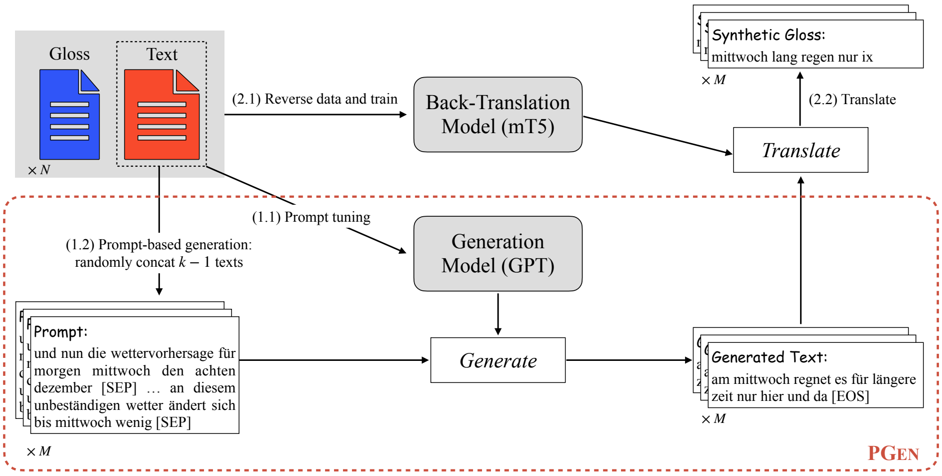


Figure 2: The overall framework of back-translation for gloss-to-text translation in this work. Procedures framed in the red dashed box corresponds to our prompt-based domain text generation (PGEN) approach.

corpora based on the text part of the original small-scale dataset $\mathcal{D}_{\mathrm{g2t}}$.

For the original text $\mathcal{D}_{\mathrm{t}} = \{\mathbf{y}^i\}_{i=1}^N$ from $\mathcal{D}_{\mathrm{g2t}}$, and a PLM $\mathcal{M}_G$, we attempt to generate an in-domain spoken language text dataset $\hat{\mathcal{D}}_{\mathrm{t}} = \{\hat{\mathbf{y}}^i\}_{i=1}^M$ with a much larger data size than $\mathcal{D}_{\mathrm{t}}$ (i.e., $M \gg N$). As shown in Figure 2, our PGEN approach includes two phases:

- **Prompt Tuning:** Following (Kumar et al., 2020), we finetune the PLM on the original small-scale spoken text dataset $\mathcal{D}_{\mathrm{t}}$ with artificial prompts. Specifically, we randomly concatenate k sentences from $\mathcal{D}_{\mathrm{t}}$ as a training sample, i.e., $[\mathbf{y}^{i_1}; \texttt{[SEP]}; \mathbf{y}^{i_2}; \texttt{[SEP]}; \dots; \mathbf{y}^{i_k}; \texttt{[EOS]}]$, where `[SEP]` and `[EOS]` represent the delimiter and the end-of-sentence tokens, respectively. We denote the finetuned PLM as $\mathcal{M}_{G_{\mathrm{FT}}}$.

- **Prompt-Based Generation:** In the generation phase, we randomly select $k-1$ sentences from $\mathcal{D}_{\mathrm{t}}$ to form a prompt, i.e., $\texttt{prompt} = [\mathbf{y}^{j_1}; \texttt{[SEP]}; \mathbf{y}^{j_2}; \texttt{[SEP]}; \dots; \mathbf{y}^{j_{k-1}}; \texttt{[SEP]}]$. Then, we input the prompt into $\mathcal{M}_{G_{\mathrm{FT}}}$ to generate the k-th sentence $\hat{\mathbf{y}}^{j_k} = \mathcal{M}_{G_{\mathrm{FT}}}(\texttt{prompt})$. We complete the text generation process when the model produces an `[EOS]` token.

According to the design of prompts, the number of permutations for any $k-1$ sentences from the full set with N sentences is $A_N^{k-1} = \frac{N!}{(N-k-1)!} \gg N$, which allows us to generate a large number of in-domain sentences and maintain the diversity of $\hat{\mathcal{D}}_{\mathrm{t}}$.

2.2 Back-Translation

Generally, the BT model is trained on the same dataset for the gloss-to-text task but in the opposite direction, i.e., $\mathcal{D}_{\mathrm{t2g}} = \{(\mathbf{y}^i, \mathbf{x}^i)\}_{i=1}^N$. However, the data scale of $\mathcal{D}_{\mathrm{t2g}}$ is too small to develop a well-performing text-to-gloss translation model. Previous study (Hoang et al., 2018) on machine translation also suggests that the quality of BT models heavily affects the performance of the final models. Therefore, we take advantage of pretrained sequence-to-sequence models by finetuning them on $\mathcal{D}_{\mathrm{t2g}}$ to improve the performance of the BT model. Specifically, we utilize a multilingual pretrained model, i.e., mT5 (Xue et al., 2020), to support the different languages (i.e., German, Chinese and English) of the SLT benchmarks.

2.3 Overall Framework

The workflow of our approach is illustrated in Figure 2, which is divided into four steps: (1.1) finetune the pretrained GPT-2 model on the original small-scale spoken language text dataset; (1.2) apply the finetuned GPT-2 model for in-domain spoken language texts generation with artificial prompts; (2.1) finetune the pretrained sequence-to-sequence model (i.e., mT5) on the original text-gloss dataset $\mathcal{D}_{\mathrm{t2g}}$ to obtain the BT model; (2.2) adopt the BT model to translate the generated in-domain spoken language texts into glosses to synthesize a large-scale pseudo-parallel data, which are combined with the original small-scale dataset $\mathcal{D}_{\mathrm{g2t}}$ to train the final gloss-to-text translation model.

Dataset	Language Pair	Gloss-Text Pairs
Phoenix2014T	DSL-German	7,086 / 519 / 642
CSL-Daily	CSL-Chinese	18,401 / 1,077 / 1,176
ASLG-PC12	ASL-English	82,709 / 4,000 / 1,000

Table 1: Statistics of the three benchmark datasets for gloss-to-text translation used in this work. The third column presents the number of gloss-text pairs in the training, validation and test sets, respectively.

3 Experiments

In this section, we conduct both intrinsic and extrinsic evaluations (Kumar et al., 2020) for the proposed PGEN approach. For intrinsic evaluation (§3.2), we perform analyses on the domain properties of spoken language texts generated by PGEN. As for extrinsic evaluation (§3.3), we conduct sign gloss translation experiments using back-translation approach with the generated texts. The performance of the downstream task can indirectly reflect the effectiveness of PGEN.

3.1 Experimental Setup

Dataset. We employ three widely used benchmark datasets for sign language translation, namely, Phoenix2014T (Camgoz et al., 2018), CSL-Daily[2] (Zhou et al., 2021a), and ASLG-PC12[3] (Othman and Jemni, 2012), which are in German, Chinese and English, respectively. Statistics of the datasets are presented in Table 1.

Model. As shown in Figure 2, there are three kinds of models involved in this work for 1) in-domain text generation, 2) gloss-to-text translation and 3) back-translation, respectively. Details of the model and training settings can be found in §3.2 , §3.3 and Appendix A.1.

3.2 Domain Text Generation

We perform intrinsic evaluation by analyzing the domain properties of spoken language texts generated by PGEN. Unless otherwise stated, we primarily conduct the analyses on the German Phoenix2014T dataset. More results on ASLG-PC12 and CSL-Daily datasets can be found in Appendix A.3.

We first adopt the pre-trained GPT-2 model (e.g., German GPT-2[4] for Phoenix2014T) and finetune the model on the artificial prompts created from the text part of SLT dataset. Specifically, the PLM is finetuned to predict the next token in the exact way that GPT-2 was pretrained, with the same training procedure and hyper-parameters. Then, we use the finetuned PLM to generate in-domain spoken language texts for each SLT task (§2.1). By default, we set the hyper-parameter k to 20.

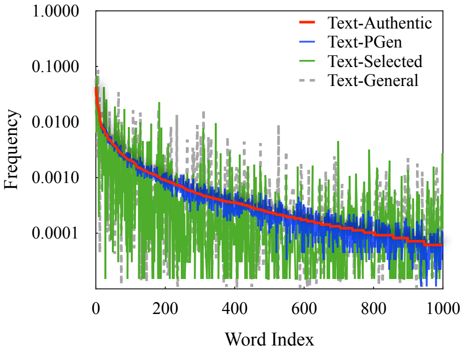


Figure 3: The word frequency distribution on different types of spoken language texts. The X-axis represents different words, while the Y-axis represents the normalized word frequency. Best viewed in color. Analyses is conducted on the German Phoenix2014T

For comparison, we consider four types of spoken language texts:

- TEXT-AUTHENTIC: The text side of the SLT dataset.
- TEXT-PGEN: The spoken language texts generated by our PGEN approach with $k = 20$.
- TEXT-SELECTED: We use the cross-entropy difference selection method (Moore and Lewis, 2010) to collect in-domain texts data from IWSLT17 Multilingual Task[5] based on the AUTHENTIC texts.
- TEXT-GENERAL: We randomly sample sentences from the target text part of IWSLT17 Multilingual Task. TEXT-GENERAL can be considered as the general domain spoken language texts.

We measure the similarity of the other three spoken language texts to the TEXT-AUTHENTIC at both word level and sentence level.

Word Distribution. Recent study by Wang et al. (2022) suggests that the word frequency distributions can reflect the domain difference of datasets.

---

[2] http://home.ustc.edu.cn/~zhouh156/dataset/csl-daily/

[3] https://github.com/kayoyin/transformer-slt

[4] https://huggingface.co/dbmdz/german-gpt2

[5] https://sites.google.com/site/iwsltevaluation2017/TED-tasks

Data		JS↓
TEXT-PGEN	vs. TEXT-AUTHENTIC	0.01
TEXT-SELECTED	vs. TEXT-AUTHENTIC	0.18
TEXT-GENERAL	vs. TEXT-AUTHENTIC	0.26

Table 2: The JS divergence between different types of spoken language texts for Phoenix2014T dataset.

Test Data	In-domain	General
TEXT-AUTHENTIC	99.38%	0.62%
TEXT-PGEN	98.60%	1.40%
TEXT-SELECTED	56.23%	43.77%
TEXT-GENERAL	0.31%	99.69%

Table 3: The domain classification results of different spoken language texts on Phoenix2014T dataset.

Thus, we count the word frequencies for the above four spoken language texts and present the results by both visualization and the Jensen-Shannon (JS) divergence (Lin, 1991). For a fair comparison, we sample the same number of texts from the other three spoken language texts as TEXT-AUTHENTIC (i.e., 7086 for Phoenix2014T).

Figure 3 visualizes the word frequency distributions of top 10000, in which the words are ranked according to their frequencies in the TEXT-AUTHENTIC corpus. We can observe that TEXT-PGEN (blue line) shows a similar distribution as TEXT-AUTHENTIC (red line) while TEXT-SELECTED and TEXT-GENERAL differ from TEXT-AUTHENTIC significantly. This result qualitatively shows that the spoken language texts generated by our PGEN approach are more close to the domain of TEXT-AUTHENTIC (e.g. the domain of sign language).

To quantitatively measure the distance between these distributions, we compute the JS divergence expressed as:

$$\mathrm{JS}\,(P||Q) = \frac{1}{2}\left(\mathrm{KL}(P||\frac{P+Q}{2}) + \mathrm{KL}(Q||\frac{P+Q}{2})\right),$$

where $\mathrm{KL}(\cdot||\cdot)$ denotes the Kullback–Leibler divergence (Kullback and Leibler, 1951) of two distributions (i.e., P and Q). Table 2 lists the JS divergence from the other three corpora to TEXT-AUTHENTIC. We find that the JS divergence from TEXT-PGEN to TEXT-AUTHENTIC is much smaller than the others, further demonstrating that the spoken language texts generated by PGEN are closer to the domain of TEXT-AUTHENTIC. These demonstrate the effectiveness and generalizability of the proposed PGEN approach.

Domain Classifier. The word frequency distribution only characterizes one aspect of the domain. More features, for example, the styles of texts, can not be explicitly modeled. Thus, we follow Du et al. (2020) to train a domain classifier to distinguish the in-domain and the general domain data with the consideration of all potential features implicitly. We perform a binary classification task with equal examples (i.e., train/valid as 7086/519) from TEXT-GENERAL and TEXT-AUTHENTIC, respectively. To train the domain classifier, we finetune the German BERT[6] on the above dataset (Sun et al., 2019). Specifically, we use the German BERT to encode the input sentence and feed the `[CLS]` token vector as a reasonable sentence embedding to the domain discriminator. For testing, we also sample the same number of examples from the other three spoken language text corpora as the test set of TEXT-AUTHENTIC (i.e., 642) and predict their domains (i.e., general or in-domai).

The results are listed in Table 3. We observe that the domain classifier successfully predicts the true labels of TEXT-AUTHENTIC and TEXT-GENERAL, indicating the significant domain differences between sign language spoken texts and general spoken language texts. As for TEXT-PGEN, the examples are categorized into authentic texts with very high accuracy (i.e., 98.60%), while the value is much lower for TEXT-SELECTED (i.e., 56.23%). These results again demonstrate that our PGEN approach can produce better in-domain spoken language texts than the compared methods.

3.3 Sign Language Gloss Translation

We perform extrinsic evaluation for the proposed PGEN by applying back-translation with the generated spoken language texts to the sign language gloss translation task. For the different SLT tasks in 1, we adopt the corresponding German[7], Chinese[8] and English[9] GPT-2 models to generate in-domain spoken language texts with PGEN.

For the gloss-to-text translation task, we follow

---

[6] `https://huggingface.co/bert-base-german-cased`

[7] `https://huggingface.co/dbmdz/german-gpt2`

[8] `https://huggingface.co/uer/gpt2-chinese-cluecorpussmall`

[9] `https://huggingface.co/gpt2`

	Dev Set				Test Set			
	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Phoenix2014T								
Camgoz et al. (2018)	44.40	31.83	24.61	20.16	44.13	31.47	23.89	19.26
Camgoz et al. (2020b)	**50.69**	**38.16**	30.53	25.35	**48.90**	36.88	29.45	24.54
Yin and Read (2020)	49.05	36.20	28.53	23.52	47.69	35.52	28.17	23.32
Transformer	43.05	32.57	25.50	20.81	43.71	33.40	26.45	21.73
+ Scaling BT(General)	43.52	32.42	25.30	20.62	42.68	32.06	25.09	20.40
+ Scaling BT(Selected)	44.20	33.02	25.86	21.06	44.29	33.25	25.97	21.09
+ Scaling BT(PGen)	48.68	37.94	**30.58**	**25.56**	48.30	**37.59**	**30.32**	**25.54**
ASLG-PC12								
Yin and Read (2020)	92.67	88.72	85.22	81.93	92.88	89.22	85.95	82.87
Transformer	91.85	87.53	83.73	80.19	92.04	88.07	84.56	81.25
+ Scaling BT(General)	91.59	87.43	83.81	80.45	91.97	88.19	84.89	81.79
+ Scaling BT(PGen)	**93.23**	**88.91**	**85.63**	**82.04**	**93.51**	**89.74**	**86.55**	**83.35**
CSL-Daily								
Transformer	49.63	35.62	25.52	18.64	49.41	35.57	25.55	18.72
+ Scaling BT(General)	54.66	39.80	29.23	21.78	54.07	39.34	28.99	21.75
+ Scaling BT(PGen)	**60.48**	**46.92**	**36.95**	**29.72**	**60.21**	**46.76**	**36.90**	**29.75**

Table 4: Gloss-to-text translation performance on Phoenix2014T, ASLG-PC12 and CSL-daily. "+ Scaling BT(PGen)" represents that training data is increased by 40 times with BT, in which the monolingual is generated by our PGen approach.

Yin and Read (2020) to train a Transformer model with 2 encoder layers and 2 decoder layers. For back-translation, we first finetune the mT5 pretrained model on the authentic text-gloss parallel data and then use it to translate the collected spoken language texts (sampled from Text-PGen, Text-Selected or Text-General) into glosses. Following Wu et al. (2019), we train the Transformer model on the combination of the authentic and synthetic parallel data, and then finetune it on the authentic gloss-to-text parallel data.

For evaluation, we follow previous studies (Camgoz et al., 2018, 2020b; Zhou et al., 2021a) to evaluate the performance of gloss-to-text translation with BLEU score (Papineni et al., 2002), ROUGE-L (Lin, 2004), and METEOR (Banerjee and Lavie, 2005) scores. Specifically, we report the BLEU-1,2,3,4 scores to reflect the translation quality at different phrase levels.

Table 4 lists the main results of the gloss-to-text translation performance on the Phoenix2014T, ASLG-PC12 and CSL-daily datasets. By scaling the BT synthetic data to 40 times of the authentic parallel data with the different types of spoken text data, Text-PGen (i.e., “+ Scaling BT(PGen)”) improves the performance over the baseline Transformer model significantly and consistently (e.g., up to 11.03 BLEU-4 points on CSL-daily), while Text-General and Text-Selected only improve slightly or even hurt the performances (e.g., down to 1.33 BLEU-4 points on Phoenix2014). We also report the translation performance in terms of ROUGE-L and METEOR in Appendix A.4. These demonstrate the effectiveness and generalizability of the proposed PGen approach.

4 Analysis

To gain a deeper understanding on our PGen approach, we conduct extensive analyses in terms of ablation study and translation outputs.

4.1 Ablation Study

We conduct three ablation studies regarding the prompt length, the back-translation model and the scale of synthetic data on the Phoenix2014T dataset. We introduce more details as below:

Prompt Length. We first investigate the impact of the prompt length k on the gloss-to-text translation task, which decides how many sentences are concatenated as prompts for generation via PGen. Specifically, we increase k from 0 to 35 to induce the PLM to generate in-domain spoken language texts with a data size 5 times of the authentic par-

	Text-to-Gloss Dev Set				Gloss-to-Text Dev Set			
	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Transformer	–	–	–	–	43.05	32.57	25.50	20.81
+ BT Model SC	57.72	39.41	27.86	19.66	44.80	34.33	27.47	22.86
+ BT Model mT5	**59.78**	**43.70**	**32.58**	**25.01**	**46.87**	**35.95**	**28.59**	**23.54**

Table 5: Translation performance with different back-translation models on text-to-gloss and gloss-to-text tasks.

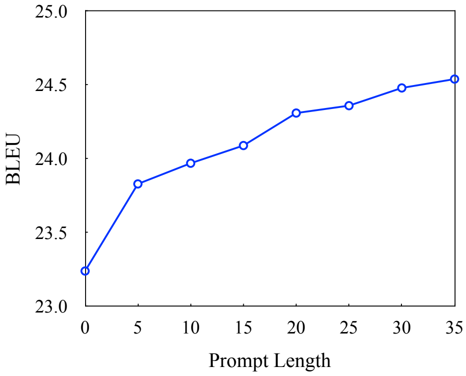


Figure 4: Gloss-to-text translation performance with respect to different prompt sizes. The X-axis represents the number of sentences in one prefix, while the Y-axis represents the BLEU-4 score.

allel data. Then, we perform back-translation with the generated spoken language texts on the gloss-to-text translation task, individually. The results are shown in Figure 4, in which we observe that the performance of gloss-to-text translation constantly improves with the increase of prompt length. This is because a larger prompt length can provide more domain signals so as to encourage the generation of higher-quality spoken language texts with closer domain and higher diversity (see Appendix A.2). However, the larger prompt length requires more computation memory, slowing down the generation process. Therefore, we set the prompt length to 20 throughout the work for a good tradeoff between the generation quality and the computation costs.

Back-Translation Model. In §2.2, we state that the quality of BT models heavily affects the performance of final models. To validate this claim, we compare two back-translation models: a Transformer model trained from scratch (i.e., "+ BT Model SC") and the finetuned mT5 (Xue et al., 2020) model (i.e., "+ BT Model mT5"). As shown in the left of Table 5, the finetuned mT5 model produces higher-quality pseudo parallel data according to BLEU scores on the validation set of the text-to-gloss translation (i.e., back-translation) task. Consequently, the performance of gloss-to-text translation is considerably improved when synthesizing data by "+ BT Model mT5", which reconfirms our claim. Therefore, throughout this work, we adopt the finetuned mT5 model for back-translation.

Scale of Synthetic Data. In §2.1, we show the potential of PGEN in generating large-scale in-domain spoken language texts. Let us recap Figure 1, where we increase the scale of spoken language texts used for back-translation. We observe that scaling the spoken language texts generated by PGEN can improve the performance of the gloss-to-text translation task consistently while that by retrieval degrades the performance. It suggests that our approach can scale the BT technique to play its maximum effect for gloss-to-text translation, which has not been achieved in previous studies.

4.2 Translation Output

We conduct further analyses to understand how the proposed approach improves the gloss-to-text translation quality. Specifically, we analyze the translation outputs of Phoenix2014 in Table 4 by the `compare-mt`[10] toolkit in terms of word frequency and sentence length.

Words Frequency. Previous study (Fadaee and Monz, 2018) shows that the back-translation improves the translation performance by improving the low-frequency word predictions. Meanwhile, our analyses in §3.2 suggest that the spoken language texts generated by our PGEN shows a similar word frequency distribution as TEXT-AUTHENTIC. We wonder how such consistency benefits the prediction of low-frequency words in gloss-to-text translation. Specifically, we first categorize the vocabulary into three groups based on the word frequency in the training data, including High: frequency $\in [2000, +\infty)$; Medium: frequency $\in [100, 2000)$; Low: frequency $\in (0, 100]$. Then,

---

[10] https://github.com/neulab/compare-mt

Data	Word Frequency		
	Low	Medium	High
Text-Authentic	28.86	49.94	58.27
+ Text-PGen	33.23	52.98	60.17
+ Text-Selected	24.65	49.87	58.58
+ Text-General	25.45	49.12	58.12

Table 6: Prediction accuracy (F1 score) of target words in the test set with respect to word frequency. As the higher F1 score, the better, we mark the improvement by green and degradation by red background.

Data	Sentence Length		
	Short	Medium	Long
Text-Authentic	21.73	22.61	10.00
+ Text-PGen	21.87	27.29	20.15
+ Text-Selected	16.24	23.93	12.80
+ Text-General	16.68	23.97	9.47

Table 7: Translation quality (BLEU score) of examples in the test set with respect to sentence length.

we utilize `compare-mt` to calculate the prediction accuracy of target words in the test set with respect to the three groups.

Table 6 lists the results for different models with scaling back-translation (e.g. 40 times synthetic data). As seen, scaling back-translation with in-domain spoken language texts of Text-PGen improves the prediction of words in all three groups, especially for low-frequency words. However, the situation is much different for Text-Selected and Text-General such that they bring little improvement for high-frequency words and inversely harm the performance on low-frequency words. It indicates that back-translation becomes ineffective when the domains of spoken language texts and authentic parallel data are mismatched, which implies the importance of our PGen approach.

Sentence Length. We investigate the translation quality of examples with varied lengths, which can be biased during generating or retrieving spoken language texts. Similar to word frequency, we also categorize the examples of test set into three groups based on the sentence length, including Long: $(20, +\infty)$ tokens; Medium: $(10, 20]$ tokens; Short: $(0, 10]$ tokens.

Table 7 lists the corresponding results. Clearly, long sentences are more difficult to translate (Zheng et al., 2020) and our PGen can particularly improve the translation quality of medium and long sentences. In contrast, the other methods show little improvement on medium and long sentences and degrade the performance on short sentences significantly. This demonstrates the better stability of our approach regarding the distribution of sentence length over the compared methods.

5 Related Work

Sign Language Gloss Translation. SLGT translates sign gloss to spoken language texts, which has attracted more attention in recent years with the development of neural machine translation (NMT). For example, Camgoz et al. (2018) released the PHOENIX14T and for the first time proposed a neural SLT model to translate from spatial representations or sign glosses. Recent studies attempt to improve both SLR and gloss-to-text translation for the better performance of SLT. Yin and Read (2020) proposed the STMC-Transformer network (Zhou et al., 2020) to improve SLR and exploited Transformer for gloss-to-text translation. Camgoz et al. (2020b) formulated SLR and gloss-to-text translation in the multi-task form while Li et al. (2020) explored the hierarchical structure for learning sign video representations. More recently, multi-cue characteristics of sign language have also been utilized for improving SLT (Camgoz et al., 2020a; Zhou et al., 2021b; Kan et al., 2022). Different from these works, we improve SLT by focusing on the gloss-to-text translation task in the perspective of spoken language generation.

Data Augmentation. Data augmentation has been proposed and proven valuable and effective in machine translation research (Sennrich et al., 2016; Zhang and Zong, 2016; Wang et al., 2018; Jiao et al., 2020, 2021). To address the data scarcity issue in gloss-to-text translation, there have been studies on producing synthetic gloss-text pairs for data augmentation. One stream is to extract discrete phrases from natural texts based on linguistic rules (Moryossef et al., 2021). Another is to adopt BT technique (Sennrich et al., 2016) to generate glosses from natural texts by a pretrained text-to-gloss translation model. However, the limited in-domain spoken language texts prevent BT from playing its maximum effect for the gloss-to-text translation task. While we may collect in-domain texts from public websites as Zhou et al. (2021a), it is unreliable due to both the requirement for domain knowledge and the accessibility of websites. Our approach exploits the knowledge and generalization capability of large PLMs to produce in-domain texts with prompt learning. While large PLMs have been successfully applied for text generation in NLP (Kumar et al., 2020; Chintagunta et al., 2021), we craft the prompts to theoretically guarantee that we can produce large-scale in-domain texts based on the small-scale original spoken language texts data.

6 Conclusion

In this paper, we propose the PGEN approach to produce large-scale in-domain monolingual texts based on the original small-scale texts of the gloss-text parallel data. With PGEN, we scale back-translation and achieve significant and consistent improvements on three benchmark datasets for SLGT task. Extensive analyses suggest that our approach generates monolingual texts with similar linguistic properties as the original monolingual texts, thus outperforms the compared methods in terms of both low-frequency word prediction and long sentence translation. Future work includes exploring ChatGPT for sign language translation task by using proper prompts (Jiao et al., 2023).

7 Limitations

We identify two limitations of our PGEN approach:

- We rely on the seed in-domain spoken language texts and the pretrained models. Therefore, a bit of expert knowledge is still required for the selection of the seed dataset, and the pretrained models need to be carefully selected to ensure a large data coverage and a strong generalization ability.
- We only investigate the sign language translation without considering the sign language understanding part. Since the latter part may introduce errors into glosses, it requires the sign gloss translation models to be robust to noisy inputs. We leave this for the future work.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 conference on empirical methods in natural language processing*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*.

Danielle Bragg, Oscar Koller, Mary Bellard, Larwan Berke, Patrick Boudreault, Annelies Braffort, Naomi Caselli, Matt Huenerfauth, Hernisa Kacorri, Tessa Verhoef, et al. 2019. Sign language recognition, generation, and translation: An interdisciplinary perspective. In *Proceedings of the 21st international ACM SIGACCESS conference on computers and accessibility*.

Jan Bungeroth and Hermann Ney. 2004. Statistical sign language translation. In *Proceedings of the Workshop on representation and processing of sign languages, Language Resources and Evaluation Conference*.

Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. 2018. Neural sign language translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. 2020a. Multi-channel transformers for multi-articulatory sign language translation. In *Proceedings of the European Conference on Computer Vision*. Springer.

Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. 2020b. Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *Proceedings of the USENIX Security Symposium*.

Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2021. Medically aware gpt-3 as a data generator for medical dialogue summarization. In *Proceedings of Machine Learning for Healthcare Conference*.

Mathieu De Coster, Dimitar Shterionov, Mieke Van Herreweghe, and Joni Dambre. 2022. Machine translation from signed to spoken languages: State of the art and challenges. *arXiv*.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware bert for cross-domain sentiment analysis. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*.

Marzieh Fadaee and Christof Monz. 2018. Back-translation sampling by targeting difficult words in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural*

Language Processing, Brussels, Belgium. Association for Computational Linguistics.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd workshop on neural machine translation and generation*.

Long Jiang, Shiquan Yang, Ming Zhou, Xiaohua Liu, and Qingsheng Zhu. 2009. Mining bilingual data from the web with adaptively learnt patterns. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*.

Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Xing Wang, and Zhaopeng Tu. 2023. Is ChatGPT a good translator? A preliminary study. *arXiv*.

Wenxiang Jiao, Xing Wang, Shilin He, Irwin King, Michael Lyu, and Zhaopeng Tu. 2020. Data rejuvenation: Exploiting inactive training examples for neural machine translation. In *Proceedings of the Empirical Methods in Natural Language Processing*.

Wenxiang Jiao, Xing Wang, Zhaopeng Tu, Shuming Shi, Michael Lyu, and Irwin King. 2021. Self-training sampling with monolingual data uncertainty for neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Jichao Kan, Kun Hu, Markus Hagenbuchner, Ah Chung Tsoi, Mohammed Bennamoun, and Zhiyong Wang. 2022. Sign language translation with hierarchical spatio-temporal graph neural network. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representation*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*.

Dongxu Li, Chenchen Xu, Xin Yu, Kaihao Zhang, Benjamin Swift, Hanna Suominen, and Hongdong Li. 2020. Tspnet: Hierarchical feature learning via temporal semantic pyramid for sign language translation. *Advances in Neural Information Processing Systems*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Text summarization branches out*.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv*.

Robert C Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the Association for Computational Linguistics*.

Amit Moryossef, Kayo Yin, Graham Neubig, and Yoav Goldberg. 2021. Data augmentation for sign language gloss translation. In *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*.

Achraf Othman and Mohamed Jemni. 2012. English-asl gloss parallel corpus 2012: Aslg-pc12. In *Proceedings of the 5th Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon LREC*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Dimitar Shterionov, editor. 2021. *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*. Association for Machine Translation in the Americas, Virtual.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Proceedings of the China national conference on Chinese computational linguistics*. Springer.

Wenxuan Wang, Wenxiang Jiao, Yongchang Hao, Xing Wang, Shuming Shi, Zhaopeng Tu, and Michael Lyu. 2022. Understanding and improving sequence-to-sequence pretraining for neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Lijun Wu, Yiren Wang, Yingce Xia, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2019. Exploiting monolingual data at scale for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv*.

Kayo Yin, Amit Moryossef, Julie Hochgesang, Yoav Goldberg, and Malihe Alikhani. 2021. Including signed languages in natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Kayo Yin and Jesse Read. 2020. Better sign language translation with stmc-transformer. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Xuan Zhang and Kevin Duh. 2021. Approaching sign language gloss translation as a low-resource machine translation task. In *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*.

Jiangbin Zheng, Zheng Zhao, Min Chen, Jing Chen, Chong Wu, Yidong Chen, Xiaodong Shi, and Yiqi Tong. 2020. An improved sign language translation model with explainable adaptations for processing long sign sentences. *Computational Intelligence and Neuroscience*.

Hao Zhou, Wengang Zhou, Weizhen Qi, Junfu Pu, and Houqiang Li. 2021a. Improving sign language translation with monolingual data by sign back-translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Hao Zhou, Wengang Zhou, Yun Zhou, and Houqiang Li. 2020. Spatial-temporal multi-cue network for continuous sign language recognition. In *Proceedings of the Association for the Advancement of Artificial Intelligence*.

Hao Zhou, Wengang Zhou, Yun Zhou, and Houqiang Li. 2021b. Spatial-temporal multi-cue network for sign language recognition and translation. *IEEE Transactions on Multimedia*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

A Appendix

A.1 Model Training

We perform extrinsic evaluation for the proposed PGEN by applying back-translation with the generated spoken language texts to the sign language gloss translation task. For the Chinese and English SLT tasks, we adopt the corresponding English[11] and Chinese[12] GPT-2 models to generate in-domain spoken language texts with PGEN.

For the gloss-to-text translation task, we follow Yin and Read (2020) to train a Transformer model with 2 encoder layers and 2 decoder layers. We use Adam (Kingma and Ba, 2015) with $\beta = (0.9, 0.98)$ and $\epsilon = 10^{-6}$ for optimization. We adopt the warm-up learning rate scheduler, which linearly increases from 1.0×10^{-4} to a peak of 5.0×10^{-4} within 2000 steps, and then decays with the inverse square root schedule. The dropout rate is 0.3 and the label smoothing is 0.1.

For back-translation, we first finetune the mT5 pre-trained model on the authentic text-gloss parallel data and then use it to translate the collected spoken language texts (sampled from TEXT-PGEN or TEXT-SELECTED) into glosses. Following Wu et al. (2019), we train the Transformer model on the combination of the authentic and synthetic parallel data, and then finetune it on the authentic gloss-to-text parallel data.

Table 8 presents the hyper-parameters of different transformer models used in this work.

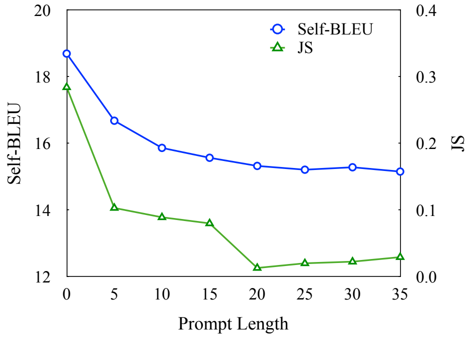


Figure 5: Diversity and domain closeness of sign language texts generated by PGEN measured by Self-BLEU and JS with respect to different prompt sizes.

A.2 Diversity and Quality

Figure 5 shows the diversity and domain closeness of sign language texts generated by PGEN measured by Self-BLEU and JS scores with respect to prompt length. Lower Self-BLEU (Zhu et al., 2018) and JS scores indicate higher diversity and closer domain, respectively.

A.3 Intrinsic Analsys Results for ASLG-PC12 and CSL-Daily Datasets

We extend the intrinsic analyses to both ASLG-PC12 and CSL-Daily datasets. Tabel 9 shows the JS divergence between different types of spoken language texts on the two SLT datasets. As seen, the JS divergence from TEXT-PGEN to TEXT-AUTHENTIC is much smaller than TEXT-GENERAL to TEXT-AUTHENTIC. Table 10 lists the domain classification results on the ASLG-PC12 and CSL-Daily datasets. The results indicate the significant domain differences between sign language spoken texts and general spoken language texts, and our PGEN approach can produce in-domain spoken language texts.

Clearly, the results on both ASLG-PC12 and CSL-Daily datasets are consistent with that in section 3.2, which demonstrates the effectiveness and generalizability of the proposed PGEN approach.

A.4 Other Metrics

Table 11 presents the ROUGE-L and METEOR scores of the gloss-to-text translation performance on the Phoenix2014T, ASLG-PC12 and CSL-daily datasets.

---

[11] `https://huggingface.co/gpt2`

[12] `https://huggingface.co/uer/gpt2-chinese-cluecorpussmall`

Parameter	Text-to-Gloss		Gloss-to-Text	
	Transformer	mT5	Pretrain	Finetune
encoder-layers	2	12	6	6
decoder-layers	2	12	6	6
learning rate	$7 \cdot 10^{-4}$	$7 \cdot 10^{-4}$	$7 \cdot 10^{-4}$	$7 \cdot 10^{-4}$
learning rate scheduler	inverse_sqrt	inverse_sqrt	inverse_sqrt	inverse_sqrt
Adam β	(0.9, 0.98)	-	(0.9, 0.98)	(0.9, 0.98)
warmup-updates	2000	-	4000	1000
label-smoothing	0.1	0.1	0.1	0.1
dropout	0.3	0.1	0.1	0.5
batch-size	2048	2048	2048	2048

Table 8: Hyperparameters of translation models.

Data		ASLG-PC12	CSL-Daily
TEXT-PGEN	vs. TEXT-AUTHENTIC	0.02	0.08
TEXT-GENERAL	vs. TEXT-AUTHENTIC	0.18	0.14

Table 9: The JS divergence results on the ASLG-PC12 and CSL-Daily datasets.

Test Data	In-domain	General
TEXT-AUTHENTIC	98.63%	1.37%
TEXT-PGEN	99.44%	0.56%
TEXT-GENERAL	1.58%	98.42%

ASLG-PC12

Test Data	In-domain	General
TEXT-AUTHENTIC	99.86%	0.14%
TEXT-PGEN	98.57%	1.43%
TEXT-GENERAL	0.24%	99.76%

CSL-Daily

Table 10: The domain classification results on the ASLG-PC12 and CSL-Daily datasets.

	Dev Set		**Test Set**	
	ROUGE-L	METEOR	ROUGE-L	METEOR
	Phoenix2014T			
Camgoz et al. (2018)	46.02	-	45.45	-
Yin and Read (2020)	47.36	**46.09**	46.58	44.85
Transformer	47.77	43.46	47.48	42.36
+ Scaling BT(General)	46.43	42.35	46.12	42.24
+ Scaling BT(Selected)	48.16	43.35	47.95	42.96
+ Scaling BT(PGen)	**50.89**	45.50	**49.25**	**44.78**
	ASLG-PC12			
Yin and Read (2020)	82.41	95.93	95.87	96.46
Transformer	91.45	92.85	94.74	95.30
+ Scaling BT(General)	91.65	92.95	94.96	95.50
+ Scaling BT(PGen)	**94.82**	**96.79**	**96.43**	**96.79**
	CSL-Daily			
Transformer	40.94	23.78	40.87	23.53
+ Scaling BT(General)	55.76	36.28	44.64	35.62
+ Scaling BT(PGen)	**62.31**	**50.48**	**60.54**	**50.35**

Table 11: Gloss-to-text translation performance on Phoenix2014T, ASLG-PC12 and CSL-daily. "+ Scaling BT(PGen)" represents that training data is increased by 40 times with BT, in which the monolingual is generated by our PGen approach.

Realistic Conversational Question Answering with Answer Selection based on Calibrated Confidence and Uncertainty Measurement

Soyeong Jeong[1] **Jinheon Baek**[2] **Sung Ju Hwang**[1,2] **Jong C. Park**[1*]
School of Computing[1] Graduate School of AI[2]
Korea Advanced Institute of Science and Technology[1,2]
`{starsuzi,jinheon.baek,sjhwang82,jongpark}@kaist.ac.kr`

Abstract

Conversational Question Answering (ConvQA) models aim at answering a question with its relevant paragraph and previous question-answer pairs that occurred during conversation multiple times. To apply such models to a real-world scenario, some existing work uses predicted answers, instead of unavailable ground-truth answers, as the conversation history for inference. However, since these models usually predict wrong answers, using all the predictions without filtering significantly hampers the model performance. To address this problem, we propose to filter out inaccurate answers in the conversation history based on their estimated confidences and uncertainties from the ConvQA model, without making any architectural changes. Moreover, to make the confidence and uncertainty values more reliable, we propose to further calibrate them, thereby smoothing the model predictions. We validate our models, Answer Selection-based realistic Conversation Question Answering, on two standard ConvQA datasets, and the results show that our models significantly outperform relevant baselines. Code is available at: `https://github.com/starsuzi/AS-ConvQA`.

1 Introduction

Conversational Question Answering (ConvQA) is the task of answering a series of questions during conversation, taking into account a given relevant paragraph (Choi et al., 2018; Reddy et al., 2019). Contrary to traditional extractive question answering tasks (Rajpurkar et al., 2016; Trischler et al., 2017) that answer each question with the given paragraph just once, ConvQA aims at answering the current question using its previous question-answer pairs taking into account the given paragraph multiple times. For example, as illustrated in Figure 1, the goal of ConvQA is to correctly answer the question Q_3 based on the previous conversation history such as Q_2, A_2, Q_1, and A_1, as well as the current context C.

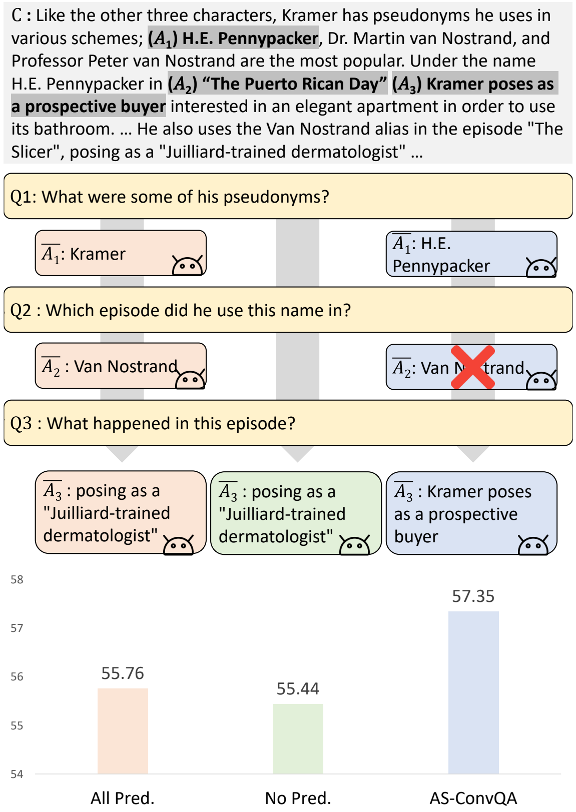


Figure 1: Illustration of realistic ConvQA evaluation with three models: 1) using all predicted answers (All Pred.); 2) not using predicted answers (No Pred.); 3) only using probably correct answers while filtering out others (AS-ConvQA, Ours). The scores in the bar chart underneath represent the F1 scores measured by all test samples (see Table 1 for full results).

ConvQA has recently gained much attention as it follows the human's information seeking process through multi-turn interactions with others. However, it is also known to be quite challenging since it requires capturing all the information over the current question, previous conversation, and the given paragraph. To tackle this problem, a considerable amount of work focuses mainly on developing a model architecture for ConvQA (Qu et al., 2019a,b; Huang et al., 2019; Chen et al., 2020; Kim et al., 2021; Qiu et al., 2021; Raposo et al., 2022).

Despite their successes, however, there remains

---

[*] Corresponding author

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 477–490
May 2-6, 2023 ©2023 Association for Computational Linguistics

a critical limitation in that they use the ground-truth answers (i.e., A_2 and A_1) in the conversation history during both training and evaluation steps. Such an evaluation procedure is not applicable to the real-world scenario, since the ground-truth answers are not accessible when the user's query is posed. Therefore, the supporting dialogue history for the current question should consist of the model's predictions $\bar{A}_1$ and $\bar{A}_2$ in the real-world application, instead of the nonexistent gold answers A_1 and A_2.

There is some recent work (Mandya et al., 2020; Siblini et al., 2021) that considers such a realistic setting on evaluation. In particular, they propose to use the model's predicted answers (i.e., $\bar{A}_1$ and $\bar{A}_2$), instead of the ground-truth answers (i.e., A_1 and A_2), for its evaluation. However, in such a setting, the model faces inconsistency between training and evaluation since the model is evaluated with the predictions while trained with the ground-truth answers. To handle such a discrepancy, Mandya et al. (2020) and Siblini et al. (2021) suggest strategies that randomly decide whether to use predicted or gold answers for the input question during training.

However, as Figure 1 shows, using all predictions as the answer history is not effective: The performance difference is not so significant when compared to not using them at all. We see that this originates from a model's failure to answering previous questions. Specifically, if a model incorrectly predicts an answer $\bar{A}_1$ for the previous question Q_1, using the incorrectly predicted answer $\bar{A}_1$ for the question Q_2 not only affects the model's current prediction $\bar{A}_2$ negatively, but also engenders further errors in the future prediction for Q_3.

Therefore, in this work, we propose a novel selection scheme for predicted answers from the conversation history, which filters out predictions that are likely to be incorrect, unlike the existing work that uses all the predicted answers including incorrect ones. The remaining step is then to identify possibly incorrect predictions. To this end, we propose to use the confidence and uncertainty of the model's prediction, which are measured by its likelihood and entropy, respectively. In particular, if the model predicts the previous answer with lower confidence (i.e., lower likelihood) or higher uncertainty (i.e., higher entropy) than a certain threshold, we regard the model's previous answer as probably incorrect, and remove it from the conversation history in answering the current question during evaluation. On the other hand, during training, we soften the sampling process so that, instead of using the hard threshold above, we sample a predicted answer based on its confidence or uncertainty (e.g., the lower the uncertainty, the higher the chance to include the predicted answer in the conversation history), in order to diversify the model's input.

However, when dealing with confidence and uncertainty, we should be careful about a miscalibrated situation (Guo et al., 2017), which happens when uncertainty and confidence do not correspond to the error and accuracy of ground-truth correctness, respectively. In other words, if the model is not calibrated enough and the distribution for confidence and uncertainty is highly skewed over particular ranges, the highly uncertain or low confident yet valid predictions could be removed. Therefore, to prevent such a performance degrading situation, we further calibrate models using a temperature scaling scheme (Guo et al., 2017) before estimating the uncertainty or confidence. We refer to our method as Answer Selection-based realistic Conversational Question Answering (AS-ConvQA).

We validate our method on two standard ConvQA datasets, QuAC (Choi et al., 2018) and CoQA (Reddy et al., 2019), against diverse baselines on a realistic evaluation protocol. The experimental results show that our method significantly outperforms these baselines, and a detailed analysis supports the importance of uncertainty- and confidence-based answer selection schemes.

Our contributions in this work are threefold:

- We propose to remove incorrect predictions in a conversation history, which degenerate ConvQA models' performances during inference.
- We present confidence- and uncertainty-based answer filtering schemes, which are further calibrated to obtain reliable predictions.
- We show that our method achieves outstanding performances on realistic ConvQA tasks.

2 Related Work

Conversational Question Answering ConvQA requires a model to understand the context of questions and paragraphs along with previous conversational questions and answers (Choi et al., 2018; Reddy et al., 2019). While the simplest approach to consider such conversation histories is to embed them along with the given question and paragraph in the representation space, recent work (Huang et al., 2019; Qu et al., 2019b; Chen et al., 2020) proposed to leverage the relevant histories by selectively using them. However, Vakulenko et al. (2021) and Kim et al. (2021) have shown that even a simple concatenation of previous questions and answers outperforms these selection-based methods, thanks to the advances in pre-trained language models (Devlin et al., 2019; Liu et al., 2019) that are designed to attend to the relevant parts. Furthermore, recent methods (Elgohary et al., 2019; Kim et al., 2021; Vakulenko et al., 2021; Raposo et al., 2022) rather focus on the problem of ambiguity in the input question by proposing a question rewriting scheme for its disambiguation, showing remarkable performance improvements.

However, the aforementioned work has a fundamental limitation on the evaluation protocol: They evaluate models based on ground-truth answers working as a conversation history, which are not available in a real-world setting. Li et al. (2022) point out this problem of ground-truth history evaluation but use the ground-truth answers during evaluation as well, since they target at disambiguating pronouns in the question by comparing the predicted and ground-truth answers. Alternatively, Mandya et al. (2020) and Siblini et al. (2021) use the model's predictions instead of the ground-truth answers during evaluation, and further train the model with predicted answers. However, they do not take into account the quality of predicted answers, where low-quality ones are not useful (see Figure 1). Thus, we propose to selectively use the predicted answers that are probably correct, based on their calibrated confidences and uncertainties.

Confidence and Uncertainty As it is nearly impossible for models to always make accurate predictions, unreliable predictions become serious issues when deploying machine learning models to real-world settings. Motivated to prevent such a risk, mechanisms of estimating the reliability of model's predictive probabilities based on confidence and uncertainty are recently proposed (Abdar et al., 2021; Houben et al., 2022). We note that confidence is usually measured by the softmax outputs of models (Guo et al., 2017), and that uncertainty can be quantified by Bayesian models, which can be approximated via Monte Carlo (MC) dropout (Gal and Ghahramani, 2016; Kendall and Gal, 2017). With much work on confidence and uncertainty estimations in computer vision tasks (Guillory et al., 2021), related topics have been recently adopted for NLP tasks as well (Shelmanov et al., 2021; Wu et al., 2021; Malinin and Gales, 2021; Vazhentsev et al., 2022). While confidence and uncertainty estimation should also be considered in ConvQA, we believe that this venue is under-explored so far. In particular, since questions are asked sequentially, it is likely that untrustworthy predictions in the conversation history would negatively affect the performance. To tackle this, we propose to exclude low-confident or uncertain predictions when training and evaluating the ConvQA model.

Calibration Confidence and uncertainty help interpret the validity of the model's prediction. However, it is not safe to rely on them when the model is not calibrated, where the correct likelihood does not match the predicted probability (Guo et al., 2017), or the model error does not match the predicted uncertainty (Laves et al., 2019). Since deep neural networks are prone to miscalibration as the number of parameters has much increased, large pre-trained language models are also not free from this problem (Wang et al., 2021; Zhao et al., 2021; Dan and Roth, 2021). One of the most prevalent approaches to calibrating the model is to rescale a logit vector before the softmax function for regularizing the probability, which is known as temperature scaling (Guo et al., 2017). While there exist lots of calibration schemes, including label smoothing (Szegedy et al., 2016) and confidence penalty (Pereyra et al., 2017), in this work, we use temperature scaling as a calibrator, since it is simple yet effective while not changing the output class of the model prediction (i.e., only scaling logits).

3 Method

We first introduce ConvQA. Then, we describe our answer validating methods based on confidence and uncertainty values with their calibration schemes.

3.1 Conversational Question Answering

We provide general descriptions of a ConvQA task. For the i-th turn of the conversation, we are given a question Q_i and its corresponding context C, as well as its conversation history consisting of previous questions and answers: $\mathcal{H}_i = \{Q_{i-1}, A_{i-1}, ..., Q_1, A_1\}$. Then, the goal of ConvQA is to correctly extract the ground-truth answer A_i from C along with Q_i and $\mathcal{H}_i$, as follows:

$$P(A_i) = M_\theta(C, Q_i, Q_{i-1}, A_{i-1}, ..., Q_1, A_1), \quad (1)$$

where M_θ is a ConvQA model, parameterized by θ, and, for the sake of simplicity, we omit conditional variables C, Q_i, and $\mathcal{H}_i$ on the left side of Equation 1, i.e., $P(A_i) = P(A_i|C, Q_i, \mathcal{H}_i)$. Note that existing work (Elgohary et al., 2019; Kim et al., 2021; Vakulenko et al., 2021; Raposo et al., 2022) has an unrealistic assumption that a set of ground-truth answers $\{A_{i-1}, ..., A_1\}$ is available during evaluation as in Equation 1. However, this evaluation setup is far from reality, since they are not always available when the user's novel questions come in, unlike the training phase which optimizes a model with ground-truth answers. Therefore, we should particularly modify the formulation in Equation 1 to accommodate a realistic evaluation scenario, which we describe in the next subsection.

3.2 Realistic ConvQA

To tackle the problem of accessing ground-truth answers during evaluation in Equation 1, we aim at redefining its formulation to evaluate ConvQA models under real-world situations as shown below.

Evaluation When a user asks a unique question whose ground-truth answers are not accessible, the most naïve approach is to work with the relevant context and previous questions, as follows:

$$P(\bar{A}_i) = M_\theta(C, Q_i, Q_{i-1}, ..., Q_1), \qquad (2)$$

where $\bar{A}_i$ denotes the i^{th} predicted answer $(i > 1)$ during inference time. However, the formulation in Equation 2 may be suboptimal, since it ignores predicted answers $\{\bar{A}_{i-1}, ..., \bar{A}_1\}$ that occurred in the former conversation, which may be beneficial for the current prediction. Thus, we can instead make inference with predicted answers, as follows:

$$P(\bar{A}_i) = M_\theta(C, Q_i, Q_{i-1}, \bar{A}_{i-1}, ..., Q_1, \bar{A}_1). \qquad (3)$$

However, when evaluating with Equation 3 while training with Equation 1, a problematic discrepancy arises, as model M_θ uses gold answers A_i for training but predicted answers $\bar{A}_i$ for inference.

Training To tackle this inconsistency, recent work (Mandya et al., 2020; Siblini et al., 2021) randomly decides whether to use $\bar{A}_i$ or A_i during the training phase, as follows:

$$P(A_i) = \begin{cases} M_\theta(C, Q_i, Q_{i-1}, \bar{A}_{i-1}, ...) \text{ w.p. } \lambda_{rand}, \\ M_\theta(C, Q_i, Q_{i-1}, A_{i-1}, ...) \text{ w.p. } 1 - \lambda_{rand}, \end{cases} \qquad (4)$$

where λ_{rand} is the probability of using $\bar{A}_i$, which is set based on heuristic sampling schemes, either using the random coin flipping or increasing the sampling rate based on the number of steps.

While such an attempt bridges the gap between training and inference in the real-world setting, critical limitations remain. First, as Figure 1 shows, we observe that using all the predicted answers rarely contributes to the model performance, as they include incorrect answers that hinder accurate predictions for the current question. Also, we further point out that there still exists a discrepancy between training and evaluation: The model observes ground-truth answers in Equation 4 which are yet unobservable for evaluation in Equation 3. Therefore, to tackle these challenges, we propose to selectively use the predicted answers based on the predictions' confidences and uncertainties.

3.3 Predicted Answer Selection Scheme

Our key intuition is that confidence and uncertainty are simple yet effective measures to filter out inaccurate predictions. Before going into details, we first define the notations. Let $\boldsymbol{x_i} \in X$ be an i^{th} input (i.e., turn) for ConvQA model M_θ, which consists of current question Q_i, its relevant context C, and conversation history $\mathcal{H}_i$. Then, labels of given input $\boldsymbol{x_i}$ are defined as $y_i^{(start)} \in C$ and $y_i^{(end)} \in C$ with $C \in \{1, ..., K\}$, where K is the number of sequence lengths for context C. In other words, $y_i^{(start)}$ and $y_i^{(end)}$ denote the start and end spans, respectively. Further, to predict labels $y_i^{(start)}$ and $y_i^{(end)}$, we first obtain a logit vector $\boldsymbol{z}_i$ for each label[1], and use it for calculating a probability vector $\boldsymbol{p}_i$ over K spans: $\boldsymbol{p}_i = \texttt{softmax}(\boldsymbol{z}_i)$, where `softmax` is a softmax function.

Confidence We now define the confidence. From the probability $\boldsymbol{p} = \texttt{softmax}(\boldsymbol{z})$, a model likelihood can be interpreted as confidence, as follows[2]:

$$s_{conf} = \max_{y \in C} p(y|\boldsymbol{z}), \qquad (5)$$

where s_{conf} denotes the confidence value.

Uncertainty While confidence can estimate how confident the model is on its prediction, it might be also beneficial to measure the model's certainty with Bayesian deep learning techniques (Kendall and Gal, 2017) to prevent erroneous predictions, which we describe here. At first, to calculate the uncertainty value, we need to obtain N different predictions for approximating the model's distribution. To do so, we first enable dropout (Srivastava

---

[1] We omit superscripts *start* and *end* for simplicity.

[2] For simplicity, we omit a turn index i, which is represented in a subscript, for example, Q_i for the i^{th} conversation.

et al., 2014) in the language model during inference, and then forward input $\boldsymbol{x}$ for N times with N different dropout masks, which is referred to as Monte Carlo (MC) dropout (Gal and Ghahramani, 2016). Then, we can obtain probability vector $\boldsymbol{p}$ via MC integration: $\boldsymbol{p} = \frac{1}{N}\sum_{n=1}^{N} \texttt{softmax}(\boldsymbol{z}^{(n)})$, where $\boldsymbol{z}^{(n)}$ is the logit vector from each forward pass. Then, based on probability $\boldsymbol{p}$, the uncertainty is quantified via its entropy over K classes (Kendall and Gal, 2017; Laves et al., 2019), as follows:

$$s_{uncer} = -\frac{1}{\log K}\sum_{k=1}^{K} p(k)\log p(k), \quad (6)$$

where s_{uncer} denotes the uncertainty value, which we normalize to be on a scale between 0 and 1 with $\frac{1}{\log K}$ in Equation 6, following (Laves et al., 2019).

3.4 Calibrating Confidence and Uncertainty

We then describe the calibration schemes to match the model's predicted confidence and uncertainty to its correct likelihood and error, respectively.

Perfect Calibration In order to calibrate trustworthiness of the confidence and uncertainty, we first describe perfectly calibrated situations. Given the input $\boldsymbol{x}$, the model predicts the most likely class, $\bar{y} = \arg\max \boldsymbol{p}$, from the entire classes with the highest probability, $\bar{p} = \max \boldsymbol{p}$. Each perfect calibration for confidence and uncertainty is then as follows (Guo et al., 2017; Laves et al., 2019):

$$\begin{aligned} &\mathbb{P}(\bar{y} = y | s_{conf} = p) = p, \\ &\mathbb{P}(\bar{y} \neq y | s_{uncer} = p) = p, \end{aligned} \quad (7)$$

where y denotes the true label with $\forall p \in [0, 1]$.

Calibration and Uncertainty Error However, perfect calibration defined in Equation 7 is hardly achievable in practical settings due to noise and prediction errors. Thus, we rather define a calibration error to estimate how much the model's prediction is calibrated. One of the most prevalent methods to quantify calibration error for confidence is to measure the difference in expectation between confidence and accuracy as follows (Guo et al., 2017):

$$\mathbb{E}_{s_{conf}}[\, |\mathbb{P}(\bar{y} = y | s_{conf} = p) - p|\,], \quad (8)$$

where $\forall p \in [0, 1]$. Also, miscalibration of uncertainty is quantified as follows (Laves et al., 2019):

$$\mathbb{E}_{s_{uncer}}[\, |\mathbb{P}(\bar{y} \neq y | s_{uncer} = p) - p|\,]. \quad (9)$$

However, since s_{conf} and s_{uncer} lie in a continuous domain, it is impossible to sample them infinite times for every p when measuring calibration errors. Therefore, we further approximate them in a discrete space, which was in the continuous domain (Equations 8, 9), by dividing the predictions into M bins and then measuring accuracy for each corresponding bin. Formally, accuracy and confidence per bin are as follows (Guo et al., 2017):

$$\begin{aligned} \texttt{acc}(B_m) &= \frac{1}{|B_m|}\sum_{i \in B_m} \mathbf{1}(s_{conf} = y^{(i)}), \\ \texttt{conf}(B_m) &= \frac{1}{|B_m|}\sum_{i \in B_m} s_{conf}, \end{aligned} \quad (10)$$

where B_m is a set of label indices whose values are within the m^{th} bin among M non-overlapping bins. Similarly, error and uncertainty per bin are formally defined as follows:

$$\begin{aligned} \texttt{err}(B_m) &= \frac{1}{|B_m|}\sum_{i \in B_m} \mathbf{1}(s_{uncer} \neq y^{(i)}), \\ \texttt{uncer}(B_m) &= \frac{1}{|B_m|}\sum_{i \in B_m} s_{uncer}. \end{aligned} \quad (11)$$

Using definitions in Equations 10, 11 above, we now measure the approximated calibration errors. Regarding confidence, the Expected Calibration Error (ECE) (Guo et al., 2017) is defined as follows:

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} |\texttt{acc}(B_m) - \texttt{conf}(B_m)|, \quad (12)$$

where n is the number of samples in total. For uncertainty, Expected Uncertainty Calibration Error (UCE) (Laves et al., 2019) is defined as follows:

$$UCE = \sum_{m=1}^{M} \frac{|B_m|}{n} |\texttt{err}(B_m) - \texttt{uncer}(B_m)|. \quad (13)$$

Calibration with Temperature Scaling With the calibration criteria (i.e., ECE and UCE), we now aim at obtaining well-calibrated confidence and uncertainty values having low ECE and UCE. To do so, we apply a temperature scaling scheme, which regulates the scale of the obtained logit vector $\boldsymbol{z}$ with a single scalar, namely temperature $\tau > 0$. Note that temperature scaling does not affect the maximum value of the softmax output; therefore, accuracy is preserved. Formally, the calibrated probability vector $\hat{\boldsymbol{p}}$ is defined as follows:

$$\hat{\boldsymbol{p}} = \texttt{softmax}(\boldsymbol{z}/\tau). \quad (14)$$

We find the τ value based on the low calibration errors, i.e., ECE and UCE, in experiments.

3.5 Overall Pipeline

We now summarize the overall pipeline of our AS-ConvQA framework, which leverages the calibrated confidence and uncertainty values to sample valid predictions for inference, while using them during training as well. Our training pipeline consists of two steps, which we explain below.

Step 1 We start training a model with gold answers A_i following the training protocol in Equation 1, since, if the model cannot observe gold answers, it might fail to capture and generate accurate answers, easily leading to degenerated performances (Mandya et al., 2020). Then, to prepare for Step 2, we make inference with it to obtain prediction $\bar{A}_i$ together with its confidence and uncertainty, for each input $\boldsymbol{x}_i$ in the training set.

Step 2 With the predicted answers and their confidences and uncertainties from Step 1, we further train the model to reflect the predicted answers instead of the ground-truth answers. Note that our objective is to filter out less confident or uncertain predictions in inference. Thus, since filtered ones are not observable during our realistic evaluation phase, we also aim at reflecting such an occurrence during training to narrow the gap between training and evaluation. To do so, instead of training with all predicted answers, we rather sample a predicted answer based on its confidence or uncertainty value:

$$P(A_i) = \begin{cases} M_\theta(C, Q_i, Q_{i-1}, \bar{A}_{i-1}, ...) \text{ w.p. } \lambda_{valid}, \\ M_\theta(C, Q_i, Q_{i-1}, ...) \text{ w.p. } 1 - \lambda_{valid}, \end{cases} \tag{15}$$

where λ_{valid} is obtained by the previous prediction's ($\bar{A}_{i-1}$) confidence or uncertainty: $\lambda_{valid} \in [s_{conf}, 1 - s_{uncer}]$. Note that, in contrast to existing work (Mandya et al., 2020; Siblini et al., 2021) represented in Equation 4, our work does not use previous gold-answers (A_{i-1}) for training as well.

For evaluation, we follow the realistic evaluation protocol described in Equation 3. However, instead of using all predictions (Mandya et al., 2020; Siblini et al., 2021), we rather remove low-confident or uncertain predictions against the threshold.

4 Experimental Setups

We explain datasets, metric, and models. Please see Appendix A for further implementation details.

4.1 Dataset and Metric

QuAC QuAC (Choi et al., 2018) is the benchmark ConvQA dataset, which is known to resemble a realistic information seeking dialogue, where questioners were prevented from reading paragraphs for its collection. QuAC consists of 14K dialogues and 100K pairs of questions and paragraphs. As the test set is not publicly open, we use a development set.

	QuAC		CoQA	
	BERT	**RoBERTa**	**BERT**	**RoBERTa**
Gold	59.86	65.08	72.79	77.62
No Pred.	55.44	61.24	70.83	75.56
All Pred.	55.76	61.53	71.28	75.42
CoQAM	55.83	61.55	71.27	74.29
Robust-P	54.21	60.32	70.17	73.96
Attentive Selection	55.74	61.42	71.05	74.60
AS-ConvQA$_{\texttt{conf}}$ (Ours)	**57.03**	**62.47**	**72.00**	**76.52**
AS-ConvQA$_{\texttt{uncer}}$ (Ours)	**57.35**	**62.33**	**72.08**	**76.33**
AS-ConvQA$_{\texttt{combine}}$ (Ours)	**57.06**	**62.18**	**71.99**	**76.76**

Table 1: F1-scores on QuAC and CoQA. Note that Gold model is not a fair baseline as it uses the ground-truth answers during inference, and thus is evaluated in an unrealistic setting.

CoQA CoQA (Reddy et al., 2019) is another ConvQA dataset with 127K pairs of questions and paragraphs; however, unlike QuAC, questioners were allowed to share paragraphs during collection. We also use a development set instead of the test set, which is not publicly available.

F1-score We evaluate models with F1-score, following the standard protocol (Kim et al., 2021).

4.2 Question Answering Models

For question answering models, we use two base-size pre-trained language models widely used in ConvQA tasks: **BERT-base** (Devlin et al., 2019) and **RoBERTa-base** (Liu et al., 2019).

4.3 Baselines and Our Models

We compare AS-ConvQA to other relevant baselines using predicted answers. Gold model, which is an indicator, uses gold answers as the answer history during evaluation, which is not realistic, whereas all the others are evaluated with predicted answers. All models are trained with the same protocol, using gold answers as the conversation history for the first half of training epochs (Step 1).

Gold This model uses the ground-truth answers during training and evaluation, thus unrealistic.

No Prediction (No Pred.) This model does not use the predicted answers as the conversation history in either training or evaluation steps.

All Prediction (All Pred.) In contrast to No Pred., this model uses all the predicted answers during both training and evaluation steps.

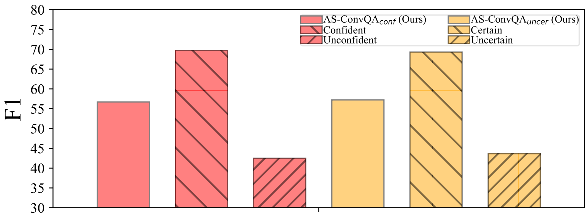


Figure 2: Comparison results of certain (confident) and uncertain (unconfident) predictions on QuAC. Note that a threshold is set as the median of the uncertainty (confidence) values.

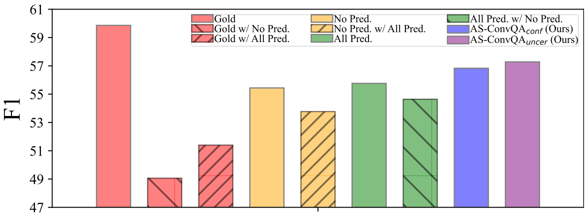


Figure 3: F1-scores on the mismatching evaluation settings for each baseline model on QuAC, either using all of the predictions or none of them as the previous answer history.

CoQAM For training, this model uses the random sampling scheme represented in Equation 4, which samples either predicted or ground-truth answers with coin-flipping (Mandya et al., 2020). For evaluation, it uses all the predictions as the history.

Robust-P Similar to CoQAM, this model uses a heuristic answer sampling scheme in a random manner, but increases the predicted answer sampling rate for training (Siblini et al., 2021). Also, it is evaluated with all predicted answers.

Attentive Selection This model uses the attention mechanism to softly select the relevant answers in the history, following previous work (Qu et al., 2019b; Huang et al., 2019; Chen et al., 2020).

AS-ConvQA$_{\texttt{conf}}$ (Ours) This is our model that filters out unconfident answers via confidence values during training and evaluation, after calibration.

AS-ConvQA$_{\texttt{uncer}}$ (Ours) This is also our model that filters out uncertain answers during training and evaluation, after calibrating uncertainty values.

AS-ConvQA$_{\texttt{combine}}$ (Ours) This model combines our confidence and uncertainty modules, where we use the mean of calibrated confidence and (1-uncertainty) values for filtering out samples.

5 Results and Discussion

In this section, we show overall performances of our proposed method along with detailed analyses.

Main Results As Table 1 shows, the proposed AS-ConvQA models including confidence and uncertainty schemes show significant performance gains over all baselines on two different QA models. Interestingly, No Pred. model, which does not utilize previous answers as the conversation history, shows comparable to or even better performance than the other baseline models based on either exploiting all the predicted answers or randomly sampling them with heuristic ratios. This implies that it is more helpful not to use low-quality predicted answers – unconfident or uncertain – at all than to use them. On the other hand, our models take advantage of filtering out probably invalid predictions, thus achieving improved performance.

Moreover, our AS-ConvQA models outperform the attention-based history selection model (i.e., Attentive Selection). This is because, even though previous answers are all incorrect, the attention scheme should leverage some of them (i.e., the sum of attention scores for previous answers should be 1), which leads the model to answer with an inaccurate history. Meanwhile, AS-ConvQA models can ignore possibly wrong predictions, thus decreasing the risk of being affected by the inaccurate history.

Last, when combining confidence and uncertainty modules, the performance is not much further enhanced. To analyze this, we first measure the number of overlapping questions, where each of the AS-ConvQA$_{\texttt{conf}}$ and AS-ConvQA$_{\texttt{uncer}}$ models predicts with higher confidence or lower uncertainty than its median value. Then, we observe that about 74.82% and 77.12% of the questions overlap on QuAC and CoQA, respectively. This indicates that unconfident and uncertain samples are highly correlated, which are likely to be filtered out by both confidence- and uncertainty-based models. In other words, due to similar effects of AS-ConvQA$_{\texttt{conf}}$ and AS-ConvQA$_{\texttt{uncer}}$ models, the performance of combined models is not much improved.

Unconfident and Uncertain Predictions In order to see whether predictions with low confidence or high uncertainty actually correspond to incorrect answers, we compare the performances between the certain (unconfident) and uncertain (confident) predictions. As Figure 2 shows, low-confident and uncertain samples lead to drastic performance degradation. This result corroborates our hypothesis that a prediction with low confidence or high uncertainty acts as an obstacle in ConvQA tasks.

Impact of Realistic Evaluation Setups To see results in realistic settings – not using ground-truth answers during inference – for the Gold model, we

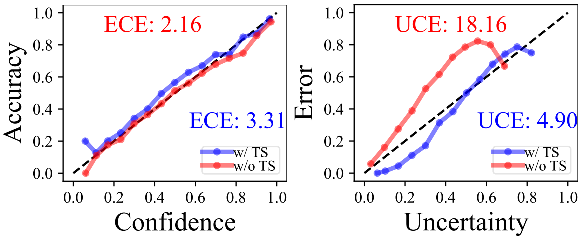


Figure 4: Reliability diagrams with and without temperature scaling (TS), regarding confidence or uncertainty on QuAC.

train it with ground-truth answers, and then test either with predicted answers or without them. Figure 3 shows that performances of the Gold model are drastically dropped, and even lower than both No Pred. and All Pred., even though tested on the same strategies. This can be explained with the term of exposure bias (Bengio et al., 2015; Mandya et al., 2020), where a discrepancy exists between training and evaluation, which hinders the model from performing well on test data that differs from training data. Furthermore, this also explains one of the reasons why CoQAM and Robust-P models perform poorly: Since they observe ground-truth answers for training, which are not observable during evaluation, they underperform ours.

Training & Evaluation Discrepancy We have observed a discrepancy between training and evaluation for the Gold model above. Then, the next possible question is whether this discrepancy also happens for models that are trained on the predicted answers, but evaluated in different settings. To see this, we test No Pred. and All Pred. models in a mismatching evaluation setting. As Figure 3 shows, a discrepancy exists for both models, though the gaps are smaller than the Gold model. This implies that even if a ConvQA model is trained on the predictions, the problem of discrepancy should not be ignored. Meanwhile, our proposed models can alleviate such an issue with a selective sampling scheme based on confidence and uncertainty.

Effectiveness of Calibration We show the effect of calibration on confidence and uncertainty values in Figure 4. Regarding confidence, the QA model already generates calibrated scores; thus there is no reason to scale the logit vector with temperature scaling (i.e., w/ temperature scaling yields more errors in terms of ECE). However, regarding uncertainty, the estimated uncertainty scores from the model have high errors in terms of UCE, i.e., not calibrated. Thus, after applying the temperate scaling scheme, the uncertainties become calibrated.

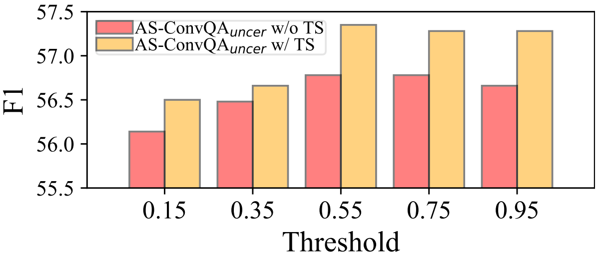


Figure 5: Comparison between the calibrated and not calibrated AS-ConvQA$_{\text{uncer}}$ with varying thresholds on QuAC.

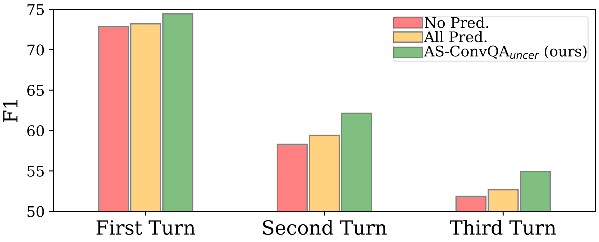


Figure 6: F1-scores for the first, second, and third conversational turns on QuAC with baselines and our AS-ConvQA.

Note that the calibrated uncertainties further contribute to the performance gain, as Figure 5 shows, since the model can observe a broad range of uncertainty values during training, making the model easily capture and reject uncertain predictions.

Effectiveness on Conversational Turns To see how the proposed AS-ConvQA contributes to the quality of the conversation as it proceeds, we further analyze the performances of former and latter conversational turns. As shown in Figure 6, both the former and latter turns benefit from our AS-ConvQA, and the performance improvements are more significant on the latter turn. This result implies that our AS-ConvQA effectively prevents the accumulation of errors, which originate from the incorrect predictions in the previous turns.

Case Study We conduct a case study. As the first example in Table 2 shows, even though both All Pred. and AS-ConvQA$_{\text{uncer}}$ models inaccurately predict $\bar{A}_6$, they handle it differently: All Pred. accepts it, while ours reject it for the subsequent question. In particular, All Pred. model misinterprets 'this' as 'big break' when answering Q_6 as well as Q_7; however, since 'this' actually refers to 'Laputa: Castle in the Sky', All Pred. further propagates the misleading prediction to the next question. By contrast, our model decides not to select $\bar{A}_6$ as a conversational history due to its high uncertainty, thus not repeating the previous mistake when answering Q_7. The proportion of such examples is about 0.52, where our model predicts the previous answer incorrectly but answers the next

Table 2: Examples in a realistic ConvQA evaluation setting for No Pred., All Pred., and AS-ConvQA$_{\text{uncer}}$ (Ours) models.

Case # 1: Our AS-ConvQA removes a previous answer and then predicts a correct answer.		
$\boldsymbol{C_1}$: ... Their collaboration has invited comparisons to the collaborations of Steven Spielberg and John Williams. This big break led to Hisaishi's overwhelming success as a composer of film scores. In 1986, Laputa: Castle in the Sky, would be the first feature to appear under the Studio Ghibli banner, and $(\boldsymbol{A_6})$ its gentle, faintly melancholic tone would become a familiar trademark of much of the studio's later output. $(\boldsymbol{A_7})$ And later, in the 1990s, Porco Rosso and Princess Mononoke were released.		
$\boldsymbol{Q_7}$: What other output did the studio release?		
	All Pred.	**AS-ConvQA$_{\text{uncer}}$ (Ours)**
$\boldsymbol{\mathcal{H}_7}$	$\boldsymbol{Q_6}$: What made this so successful? $\boldsymbol{\bar{A}_6}$: This big break led to Hisaishi's overwhelming success	$\boldsymbol{Q_6}$: What made this so successful? ~~$\boldsymbol{\bar{A}_6}$: CANNOTANSWER~~ (s_{uncer} > `Threshold`)
$\boldsymbol{\bar{A}_7}$	In 1986, Laputa: Castle in the Sky, would be the first feature to appear under the Studio Ghibli banner	And later, in the 1990s, Porco Rosso and Princess Mononoke were released.
Case # 2: Our AS-ConvQA keeps a previous answer and then, based on it, predicts a correct answer.		
$\boldsymbol{C_2}$: ... The Walk, Hanson's second studio album with 3CG Records (Fourth overall), was released in the US, Mexico and Canada on July 24. It was released in Japan on February 21 and in the UK on April 30. On May 6, 2007, the 10th anniversary of Hanson Day, $(\boldsymbol{A_5})$ the band re-recorded their first major label album, Middle Of Nowhere, at The Blank Slate bar in their hometown of Tulsa, Oklahoma. $(\boldsymbol{A_6})$ The band invited fan club members, causing hundreds to fly to Oklahoma for the acoustic event. Hanson played concerts in the summer of 2007, supporting release of The Walk.		
$\boldsymbol{Q_6}$: Was it well received?		
	No Pred.	**AS-ConvQA$_{\text{uncer}}$ (Ours)**
$\boldsymbol{\mathcal{H}_6}$	$\boldsymbol{Q_5}$: What did they do on their tenth anniversary? ~~$\boldsymbol{\bar{A}_5}$: the band re-recorded their first major label album, Middle Of Nowhere, at The Blank Slate bar in their hometown of Tulsa, Oklahoma.~~	$\boldsymbol{Q_5}$: What did they do on their tenth anniversary? $\boldsymbol{\bar{A}_5}$: the band re-recorded their first major label album, Middle Of Nowhere, at The Blank Slate bar in their hometown of Tulsa, Oklahoma. (s_{uncer} < `Threshold`)
$\boldsymbol{\bar{A}_6}$	CANNOTANSWER	The band invited fan club members, causing hundreds to fly to Oklahoma for the acoustic event.
Case # 3: Our AS-ConvQA predicts an incorrect answer since it filters out a correct previous answer.		
$\boldsymbol{C_3}$: ... Official calendars have also been issued annually from 2004 to 2009, the only exception being 2005. $(\boldsymbol{A_3})$ Girls Aloud co-wrote an autobiography titled Dreams That Glitter - Our Story. The book, named after a lyric in C̈all the Shots;̧ was published in October 2008 through the Transworld imprint Bantam Press. Before the release, OK! magazine bought the rights to preview and serialise the book. $(\boldsymbol{A_4})$ In 2007, Girls Aloud signed a PS1.25m one-year deal to endorse hair care brand Sunsilk.		
$\boldsymbol{Q_4}$: What else did they do?		
	All Pred.	**AS-ConvQA$_{\text{uncer}}$ (Ours)**
$\boldsymbol{\mathcal{H}_4}$	$\boldsymbol{Q_3}$: What else did they do/create? $\boldsymbol{\bar{A}_3}$: Girls Aloud co-wrote an autobiography titled Dreams That Glitter - Our Story.	$\boldsymbol{Q_3}$: What else did they do/create? ~~$\boldsymbol{\bar{A}_3}$: Girls Aloud co-wrote an autobiography titled Dreams That Glitter - Our Story.~~ (s_{uncer} > `Threshold`)
$\boldsymbol{\bar{A}_4}$	In 2007, Girls Aloud signed a PS1.25m one-year deal to endorse hair care brand Sunsilk.	Girls Aloud co-wrote an autobiography titled Dreams That Glitter - Our Story.

question with a high F1-score over 50. This emphasizes the importance of our selection scheme, especially when there exist ambiguous words prone to mispredictions.

In addition to this case of removing the uncertain previous prediction, we further compare our model against the No Pred. model in the case where the model predicts with the previous answer history having a low uncertainty value. As the second example in Table 2 shows, while both No Pred. and AS-ConvQA$_{\text{uncer}}$ correctly predict $\bar{A}_5$, No Pred. does not use $\bar{A}_5$ as the answer history when answering the next question, Q_6. However, as $\bar{A}_5$ contains important information of 'it' in Q_6, No Pred. model gives an inaccurate answer to Q_6, since the model is confused about what 'it' refers to. On the other hand, our model selects $\bar{A}_5$ as the answer history due to its low uncertainty value, thereby correctly predicting $\bar{A}_6$ with the previous prediction $\bar{A}_5$. In a third example of Table 2, we show the potential failure of our model, which is discussed in the Limitations section after Section 6.

6 Conclusion

In this work, in order to tackle the challenge of inaccurately predicted answers in the conversation history, we proposed a novel answer selection scheme based on their confidence and uncertainty values. We further calibrated the output values of the model to match the model's predicted confidence and uncertainty to its correct likelihood and error, which makes our answer selection scheme more reliable. The experimental results and analyses demonstrate that AS-ConvQA significantly improves the ConvQA model performance in a realistic evaluation setting without making any architectural changes.

Limitations

While we show the clear advantages of using our AS-ConvQA in realistic ConvQA tasks with both quantitative and qualitative perspectives, there could be possible failures: estimated confidence and uncertainty of a model's prediction do not match its actual correctness. For instance, the third example in Table 2 shows that AS-ConvQA$_{\texttt{uncer}}$ gives an incorrect answer to Q_4, since it removes the correctly predicted previous answer (i.e., $\bar{A}_3$) due to its incorrectly estimated uncertainty. Specifically, both Q_3 and Q_4 ask the additional information: 'What else did they do?'. However, the erroneous deletion of $\bar{A}_3$ makes our model bound to the previous question, repeatedly giving the same answer as $\bar{A}_3$. This implies that AS-ConvQA$_{\texttt{uncer}}$ sometimes assigns high uncertainty to the correct prediction and filters it, which may mislead the model, especially for the one that requires careful attention to the context with the previous answer. Therefore, as future work, one may improve mechanisms to measure incorrectness of predictions.

Ethics Statement

As the need for fully autonomous conversational agents has been rapidly emerging, it is crucial to consider whether ConvQA models can correctly answer a sequence of questions in a realistic setting, in which gold answers for previous questions are unavailable. We note that, in such a challenging setting, our work contributes to the improved performance by selectively using predicted answers with model confidence and uncertainty instead of using predefined gold answers. However, as ConvQA models predict answers based on the given paragraph, we should further consider a scenario where the paragraph itself is not trustworthy, sometimes having offensive contents. Subsequently, this may lead the entire conversation vulnerable to generating unexpected and undesired texts. While this is not the concern raised from our proposed AS-ConvQA models themselves, we still have to make an effort to prevent such an undesirable behavior.

Acknowledgements

This work was supported by Institute for Information and communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00582, Prediction and augmentation of the credibility distribution via linguistic analysis and automated evidence document collection).

References

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul W. Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. 2021. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Inf. Fusion*, 76:243–297.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*, pages 1171–1179.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Graphflow: Exploiting conversation flow with graph neural networks for conversational machine comprehension. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1230–1236. ijcai.org.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2174–2184. Association for Computational Linguistics.

Soham Dan and Dan Roth. 2021. On the effects of transformer size on in- and out-of-domain calibration. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2096–2101. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ahmed Elgohary, Denis Peskov, and Jordan L. Boyd-Graber. 2019. Can you unpack that? learning to rewrite questions-in-context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5917–5923. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24,* *2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059. JMLR.org.

Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. 2021. Predicting with confidence on unseen distributions. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 1114–1124. IEEE.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Sebastian Houben, Stephanie Abrecht, Maram Akila, Andreas Bär, Felix Brockherde, Patrick Feifel, Tim Fingscheidt, Sujan Sai Gannamaneni, Seyed Eghbal Ghobadi, Ahmed Hammam, Anselm Haselhoff, Felix Hauser, Christian Heinzemann, Marco Hoffmann, Nikhil Kapoor, Falk Kappel, Marvin Klingner, Jan Kronenberger, Fabian Küppers, Jonas Löhdefink, Michael Mlynarski, Michael Mock, Firas Mualla, Svetlana Pavlitskaya, Maximilian Poretschkin, Alexander Pohl, Varun Ravi-Kumar, Julia Rosenzweig, Matthias Rottmann, Stefan Rüping, Timo Sämann, Jan David Schneider, Elena Schulz, Gesina Schwalbe, Joachim Sicking, Toshika Srivastava, Serin Varghese, Michael Weber, Sebastian Wirkert, Tim Wirtz, and Matthias Woehrle. 2022. *Inspect, Understand, Overcome: A Survey of Practical Methods for AI Safety*, pages 3–78. Springer International Publishing, Cham.

Hsin-Yuan Huang, Eunsol Choi, and Wen-tau Yih. 2019. Flowqa: Grasping flow in history for conversational machine comprehension. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *NeurIPS*, pages 5574–5584.

Gangwoo Kim, Hyunjae Kim, Jungsoo Park, and Jaewoo Kang. 2021. Learn to resolve conversational dependency: A consistency training framework for conversational question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6130–6141. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Max-Heinrich Laves, Sontje Ihler, Karl-Philipp Kortmann, and Tobias Ortmaier. 2019. Well-calibrated model uncertainty with temperature scaling for dropout variational inference. *ArXiv*, abs/1909.13550.

Huihan Li, Tianyu Gao, Manan Goenka, and Danqi Chen. 2022. Ditch the gold standard: Re-evaluating conversational question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8074–8085. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Andrey Malinin and Mark J. F. Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Angrosh Mandya, James O'Neill, Danushka Bollegala, and Frans Coenen. 2020. Do not let the history haunt you: Mitigating compounding errors in conversational question answering. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 2017–2025. European Language Resources Association.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035. Curran Associates, Inc.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Minghui Qiu, Xinjing Huang, Cen Chen, Feng Ji, Chen Qu, Wei Wei, Jun Huang, and Yin Zhang. 2021. Reinforced history backtracking for conversational question answering. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13718–13726. AAAI Press.

Chen Qu, Liu Yang, Minghui Qiu, W. Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. 2019a. BERT with history answer embedding for conversational question answering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1133–1136. ACM.

Chen Qu, Liu Yang, Minghui Qiu, Yongfeng Zhang, Cen Chen, W. Bruce Croft, and Mohit Iyyer. 2019b. Attentive history selection for conversational question answering. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 1391–1400. ACM.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Gonçalo Raposo, Rui Ribeiro, Bruno Martins, and Luísa Coheur. 2022. Question rewriting? assessing its importance for conversational question answering. In *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part II*, volume 13186 of *Lecture Notes in Computer Science*, pages 199–206. Springer.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. Coqa: A conversational question answering challenge. *Trans. Assoc. Comput. Linguistics*, 7:249–266.

Artem Shelmanov, Dmitry Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and bayesian uncertainty estimates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 1698–1712. Association for Computational Linguistics.

Wissam Siblini, Baris Sayil, and Yacine Kessaci. 2021. Towards a more robust evaluation for conversational question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 1028–1034. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826. IEEE Computer Society.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017, Vancouver, Canada, August 3, 2017*, pages 191–200. Association for Computational Linguistics.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question rewriting for conversational question answering. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, pages 355–363. ACM.

Artem Vazhentsev, Gleb Kuzmin, Artem Shelmanov, Akim Tsvigun, Evgenii Tsymbalov, Kirill Fedyanin, Maxim Panov, Alexander Panchenko, Gleb Gusev, Mikhail Burtsev, Manvel Avetisian, and Leonid Zhukov. 2022. Uncertainty estimation of transformer predictions for misclassification detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8237–8252. Association for Computational Linguistics.

Xiao Wang, Hongrui Liu, Chuan Shi, and Cheng Yang. 2021. Be confident! towards trustworthy graph neural networks via confidence calibration. In *NeurIPS*, pages 23768–23779.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Minghao Wu, Yitong Li, Meng Zhang, Liangyou Li, Gholamreza Haffari, and Qun Liu. 2021. Uncertainty-aware balancing for multilingual and multi-domain neural machine translation training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7291–7305. Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In

Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

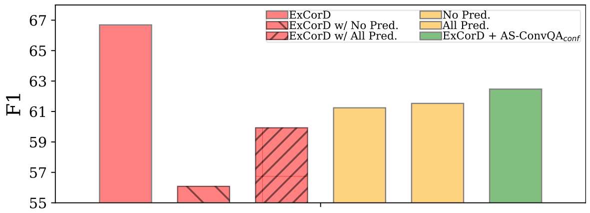


Figure 7: F1-scores on the mismatching evaluation settings for the recent ExCorD model (Kim et al., 2021) on QuAC.

A Experimental Implementation Details

We implement all models using PyTorch (Paszke et al., 2019) and Transformers library (Wolf et al., 2020). For language models, we use BERT-base and RoBERTa-base models with 110M and 125M parameters, respectively. For training, we set the training epoch as 2 with the batch size of 12, where the first epoch is used for Step 1 while the second epoch is used for Step 2. Furthermore, we optimize all models with the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 3e-5. For computiting resources, we use a single GeForce RTX 3090 GPU with 24GB memory, on which each training epoch requires approximately 4 hours.

For hyperparameters, we search the temperature value τ for temperature scaling with a validation set, in the range of $(0, 2]$. Also, we set the filtering threshold for Step 2, in the range of $[\texttt{median} - 0.25, \texttt{median} + 0.25]$, where `median` is the median value of confidence or uncertainty for all samples. For the number of dropout masks (i.e., N for the uncertainty estimation in Section 3.3) for measuring uncertainty, we set it as 10.

We use two benchmark ConvQA datasets, which are QuAC[3] (Choi et al., 2018) and CoQA[4] (Reddy et al., 2019). Note that, while our main focus is on predicting the extractive answers within a given context, CoQA is designed for answering question in a free-form text, which might not appear in a given context. Therefore, following the experimental setting from Reddy et al. (2019), we convert the CoQA dataset to our extractive ConvQA setting. In particular, we assume the gold answer as the provided rationale, and then make prediction on it, except for simple yes or no questions. For the yes or no questions, we additionally augment yes and no tokens at the end of the paragraph.

B Additional Experimental Results

Realistic Evaluation of ExCorD Even though we validate a negative impact of exposure bias in Figure 3, we further explore the performance of the unrealistic state-of-the-art model, ExCorD (Kim et al., 2021), that uses gold answer histories, in Figure 7 with the realistic ConvQA setting that uses predicted answers. We observe that, similar to the mismatching evaluation experiments reported in Figure 3, the F1-scores of ExCorD drastically drop when evaluated with No Pred. and All Pred. settings, which aligns with our motivation. On the other hand, the performance is much improved by further adapting our AS-ConvQA on ExCorD. This result indicates the importance of filtering unnecessary predictions together with the applicability of our AS-ConvQA model in a realistic setting.

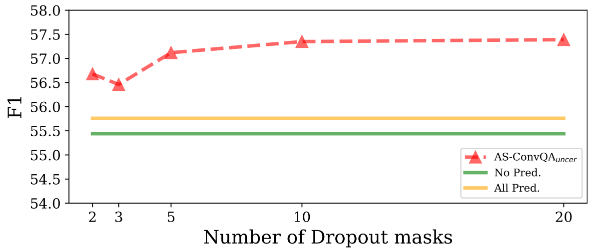


Figure 8: F1 scores with varying dropout numbers on QuAC.

Varying the Number of Dropout Masks In order to understand how the number of dropout masks (i.e., N used for uncertainty estimation in Section 3.3) affects the performance, we vary the number of masks for AS-ConvQA$_{\texttt{uncer}}$. As Figure 8 shows, the performance is stabilized after a certain number of masks (i.e., 5). This indicates the importance of setting an appropriate sampling number, since approximating the uncertainty with a small number of masks is likely to be inaccurate.

---

[3] https://quac.ai/

[4] https://stanfordnlp.github.io/coqa/

PANCETTA: Phoneme Aware Neural Completion to Elicit Tongue Twisters Automatically

Sedrick Scott Keh [1], Steven Y. Feng [*†2], Varun Gangal [†1],
Malihe Alikhani [3], Eduard Hovy [1]
[1]Carnegie Mellon University, [2]Stanford University, [3]University of Pittsburgh
`{skeh,vgangal,hovy}@cs.cmu.edu`
`syfeng@stanford.edu, malihe@pitt.edu`

Abstract

Tongue twisters are meaningful sentences that are difficult to pronounce. The process of automatically generating tongue twisters is challenging since the generated utterance must satisfy two conditions at once: phonetic difficulty and semantic meaning. Furthermore, phonetic difficulty is itself hard to characterize and is expressed in tongue twisters through a heterogeneous mix of phenomena such as alliteration and homophony. In this paper, we propose **PANCETTA**: **P**honeme **A**ware **N**eural **C**ompletion to **E**licit **T**ongue **T**wisters **A**utomatically. We leverage phoneme representations to capture the notion of phonetic difficulty, and we train language models to generate original tongue twisters on two proposed task settings. To do this, we curate a dataset called *TT-Corp*, consisting of existing English tongue twisters. Through automatic and human evaluation, as well as qualitative analysis, we show that **PANCETTA** generates novel, phonetically difficult, fluent, and semantically meaningful tongue twisters.

1 Introduction

A *tongue twister* is a sentence which is both *articulatorily difficult* (i.e. colloquially speaking, *hard to say* or "twisting") while at the same time being *meaningful and fluent*. Some examples of tongue twisters are shown in Table 1.

Together with riddles, rhymes, fables, and other such creative artifacts, tongue twisters were historically often employed as a vehicle for early transmission of native language diction, grammar, and vocabulary to children, through parent-child interaction, playtime activity, and kindergarten instruction (Akinyemi, 2003; Mcgovern, 2021). Tongue twisters have also been used as experimental aids for research studies of speech production in cognitive science and related disciplines, both among healthy speakers and those with speech and auditory disorders such as dysarthria (Kember et al., 2017). They are also used as pedagogic aids in speech therapy, as well as for treatment of speech disorders and psychological disorders relating to public speaking and elocution (Revathy and Ravindran, 2016). An example of this was in a scene[1] from *The King's Speech (2010)*, where George VI repeats a tongue twister during therapy to reduce his stutter. Lastly, tongue twisters find use as teaching aids for English diction in EFL (English as a Foreign Language) instructional settings (Prošić-Santovac, 2009).

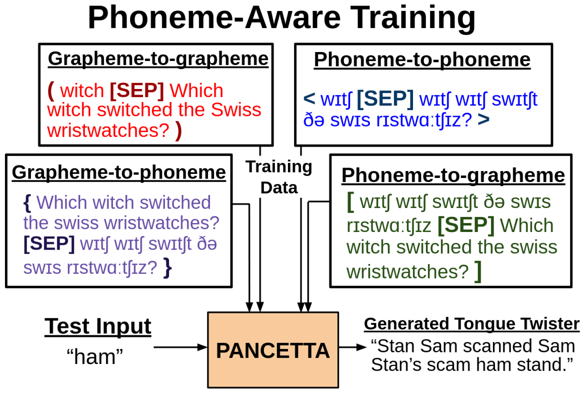


Figure 1: Overview of the phoneme-aware training in the **PANCETTA** model.

The coining of a novel, unique tongue twister which spreads sufficiently to become normative and well-recognized is rare, hence characterizing them as long-tailed linguistic phenomena (Naik, 2022). However, they are not limited to English, and are found across the world's languages, e.g., Persian (*"Shish sikh jigar sikhi shi shezar."*) (Jam, 2018) and French (*"Cinq chiens chassent six chats."*)

Consider the idealized scenario where we have a *"mouth model"* which a) maps different regions of the mouth, palate, and the larynx to the dictio-

---

[*] Work done while at CMU.

[†] Equal contribution by Steven and Varun

[1] `https://youtu.be/7WJts0gKCRM?t=53`

Code can be found at: `https://github.com/sedrickkeh/PANCETTA`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 491–504
May 2-6, 2023

Task	Input	Tongue Twister	Tongue Twister (Phoneme)
TT-Prompt	A good cook	A good cook could cook as many cookies as a good cook who could cook cookies.	ə gʊd kʊk kʊd kʊk æz mɛni kʊkiːz æz ə gʊd kʊk huː kʊd kʊk kʊkiːz.
	Chubby jugglers	Chubby jugglers juggling oranges jovially.	tʃʌbi dʒʌgəlrz dʒʌgəlɪŋ ɔrəndʒəz dʒoʊveɪli.
	Does the	Does the rapid rabid rabbit wrap it?	dʌz ðə ræpəd ræbɪd ræbət ræp ɪt?
TT-Keyword	shoes, dog	If a dog chews shoes, whose shoes does he choose?	ɪf ə dɔg tʃuːz ʃuːz, huːz ʃuːz dʌz hiː tʃuːz?
	blood, death	Bad dead bed-bugs bleed bug blood.	bæd dɛd bɛd-bʌgz bliːd bʌg blʌd.
	king, art, wall	A truly rural frugal ruler's mural was on the wall.	ə truːli rʊrəl fruːgəl ruːlrz mjʊrəl wɑːz ɑːn ðə wɔl.

Table 1: Example inputs and target outputs for both the TT-Prompt and TT-Keyword task settings, along with the phoneme representations of the tongue twisters.

nary of fundamental sounds, i.e. phones being produced, and b) based on this grounding, can quantify the hardness of producing one sound after another by inducing a distance measure between any phone pair. Assuming access to this idealized model, one could deconstruct the process of generating a tongue twister as sampling a sequence of preferably difficult/distant phone-phone transitions starting with an initial sequence of one or more phones (which could come from a prompt, or be chosen uniformly, based on the task setting).

However, there are several impediments which make realizing such an idealized model considerably intractable. First, the dictionary of fundamental sounds at the granularity we use in practice, i.e. at the level of phonemes, does not neatly map to particular points of the palate (Ladefoged and Johnson, 2014). Rather, each phoneme itself corresponds to a set of actions involving multiple organs and palatal regions. For instance, velar consonants like *k* are produced based on tongue-velum (soft upper palate) interaction. Secondly, a tongue twister as per its definition is not merely a difficult to pronounce sequence of phonemes, but also one that maps to a meaningful and fluent sequence of words. How one can maintain this property in conjunction with the process of sampling difficult transitions from the mouth model's space is unclear.

The automatic generation of tongue twisters has largely been unexplored. This task is challenging because it requires being able to model phonetic difficulty of various syllables and tokens, which is not something that existing language models are trained to do. To achieve this, we have to work in the phoneme space. Phonemes have previously been used to aid in speech recognition (Sundararaman et al., 2021) and rhyme generation (Hopkins and Kiela, 2017). We hypothesize that by working with phonemes, we will be able to model and generate patterns that characterize phonetic difficulty.

Tongue twisters go beyond relatively simpler phonetic phenomena such as alliteration, since they employ a heterogeneous mix of strategies (Jorgensen, 1981) including alliteration itself (*she sells seashells*), use of homophonic words/subwords (*sells*/*-shells*, *she*/*sea-*), and alternating between similar start phonemes for tokens (*s* and *sh*), sometimes even using multiple such phenomena in conjunction within the same example to create the cumulative effect of articulatory difficulty.

Our contributions are as follows: (1) We curate a dataset, *TT-Corp*, of diverse tongue twisters. (2) We present two new task settings (TT-Prompt and TT-Keyword) for automatic tongue twister generation, and we design and evaluate simple baselines for these tasks. (3) We propose a phoneme-aware method called **PANCETTA**, which models and generates coherent and phonetically difficult phrases by taking phonemes into account. We show that **PANCETTA** generates higher-quality tongue twisters through both automatic and human evaluations and qualitative analysis of the outputs.

2 Task Settings and Dataset

2.1 Task Settings

We propose two settings for automatic tongue twister generation. We call these tasks TT-Prompt and TT-Keyword. Examples of these two tasks can be found in Table 1, and they are detailed below:

1. **Generating tongue twisters from prompts (TT-Prompt)**: Given a few words to start a sentence, the goal is to complete the sentence in a coherent way such that the resulting generation is a tongue twister. Prompts can be of varying lengths.

2. **Generating tongue twisters from keywords (TT-Keyword)**: Given a set of keywords, the goal of this task is to generate a coherent tongue twister which incorporates the semantics of the keywords.

Tongue Twister	Non-TT Version
There was a little witch which switched from Chichester to Ipswich. (ex.1)	There was a small enchantress which exchanged from Chichester to Ipswich.
He wanted to desert his dessert in the desert. (ex.2)	He desired to abandon his sweet in the desert.
Tie a tight knot in the shape of a nought. (ex.3)	Bind a taut gnarl in the form of a zero.

Table 2: Examples of the synonym replacement process to generate non-tongue twister versions of the tongue twisters in *TT-Corp*. Different colors are used to indicate which words are replaced by their synonyms.

The set of keywords can be of varying sizes. These keywords do not necessarily have to appear verbatim and do not necessarily have to appear in order.

2.2 *TT-Corp* Dataset

As previously noted, tongue twisters are long-tailed linguistic phenomena, and it is rare to find new unique tongue twisters. Given this, we curate a dataset of 644 unique English tongue twisters into a dataset called *TT-Corp*. These tongue twisters are compiled from various sources, ranging from blog posts to English learning websites. A more detailed list of these sources and data processing details can be found in Appendix A. Despite the seemingly small scale of this dataset, multiple studies have successfully generated creative text even when training data is limited: 511 personifications (Keh et al., 2022), 1400 MadLibs (Hossain et al., 2017), 401 portmanteaus (Deri and Knight, 2015), and 576 clippings (Mattiello, 2013).

We also create a non-tongue twister version of each input in *TT-Corp*, which will later be used to explore style transfer models (§3.1) and to train a phonetic difficulty classifier (§3.2.1). This is done through synonym replacement. First, we determine the parts-of-speech of all the words in the sentence and identify the nouns, verbs, and adjectives.[2] We then use WordNet (Fellbaum, 1998) to generate a list of synonyms for each of these nouns, verbs, and adjectives, and we select the highest ranked replacement which shares the same part-of-speech. Examples of this process are shown in Table 2.

One key advantage of this synonym replacement process is that it can replace a word according to its part-of-speech in the sentence. In the second example in Table 2, the word "desert" appears twice – first as a verb (which is replaced with "abandon"), and again as a noun (which is not replaced). However, this synonym replacement process does not take the context of the words into account. In the third example, while the individual synonym replacements make sense on their own, the final sentence sounds quite unnatural. For our purposes, however, this is not a significant issue: we do not need the replacement sentences to be absolutely perfect, as the quality of the ground-truth tongue twister is more important. This will be explained further when we use this parallel dataset in §3.2.1.

3 Methodology

As this is a new task, there are no existing methods that can easily generate novel tongue twisters. The main challenge is how to incorporate phonetic difficulty into our generations. To do so, we propose two baseline and two phoneme-aware models, which are applicable to both the TT-Prompt and TT-Keyword task settings (see Table 3).

3.1 Models

1. **Grapheme-based Methods (g2g)** – We treat tongue twister generation as a seq2seq task, where the prompt (for TT-Prompt) or keywords (for TT-Keyword) is the input, and the tongue twister is the target output. We fine-tune GPT-2 (Radford et al., 2019) and GPT-J (Wang and Komatsuzaki, 2021) using the input sequences "(X [SEP] Y)", where X represents the prompt/keywords, Y represents the tongue twister, and [SEP] is a separator token.

2. **Style Transfer Methods** – Given a prompt or a set of keywords, we generate a sentence (not necessarily a tongue twister) using GPT-2. This is easy for TT-Prompt since GPT-2 is trained to do causal language modeling. For TT-Keyword, we need to first train a GPT-2 model to perform keyword-to-text. We sample 10,000 sentences from WikiText-103 (Merity et al., 2016) and extract their keywords using KeyBERT (Grootendorst, 2020). We then fine-tune GPT-2 using an "(X [SEP] Y)" template as described in the g2g methods. Here, X represents the keywords, and Y represents the corresponding sentence.

We then attempt to convert these generated natural sentences into tongue twisters. We treat this as a seq2seq task and train a seq2seq model using our parallel dataset. During training, we use the non-TT versions as inputs and the tongue twisters as the ground truth target outputs. We use BART

---

[2] The spaCy library (Honnibal and Montani, 2017) was used to extract the POS tags.

Method Name	Models Used	Description	Phoneme Representation	Leverage Pretraining
Grapheme-based Methods	GPT-2, GPT-J	g2g	✗	✓
Style Transfer Methods	BART, T5	g2g + g2g	✗	✓
PANCETTA-P	GPT-2, BART	g2p + p2p + p2g	✓	✗
PANCETTA-J	GPT-2, GPT-J	g2g, p2p, g2p, p2g (only g2g during test-time)	✓	✓

Method Name	Example
Grapheme-based Methods	She sells → She sells seashells on the seashore.
Style Transfer Methods	She sells → She sells things on the beach. → She sells seashells on the seashore.
PANCETTA-P	She sells → ʃiː sɛlz → ʃiː sɛlz siːʃɛlz ɑːn ðə siːʃɔr → She sells seashells on the seashore.
PANCETTA-J	She sells → She sells seashells on the seashore. She sells seashells on the seashore. → ʃiː sɛlz siːʃɛlz ɑːn ðə siːʃɔr. ʃiː sɛlz siːʃɛlz ɑːn ðə siːʃɔr. → She sells seashells on the seashore. ʃiː sɛlz → ʃiː sɛlz siːʃɛlz ɑːn ðə siːʃɔr.

Table 3: Summary of the models discussed in §3.1, along with some examples.

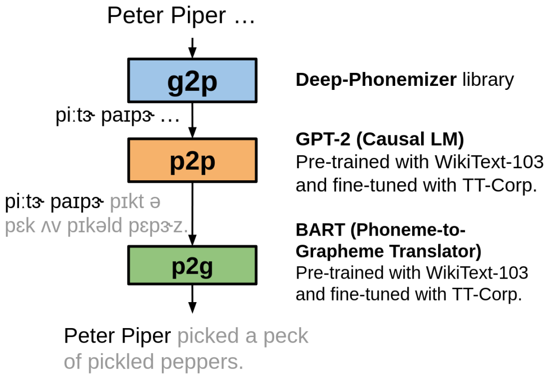


Figure 2: Overview of **PANCETTA-P** pipeline.

(Lewis et al., 2020) and T5 (Raffel et al., 2020) models for this seq2seq task.

3. **PANCETTA-P** (Phoneme) – For the previous g2g models, the fine-tuning was done only using graphemes. Because graphemes are not always representative of pronunciation, we hypothesize that it may be difficult for such models to capture information regarding the pronunciation. If we instead had a generative model which works on the phoneme space, then we could fine-tune this model on the tongue twister phonemes and hope that it can better capture these phonetic cues.

We first pretrain a GPT-2 model to perform causal LM generation for phonemes. Pretraining is done using WikiText: we first convert all the WikiText sentences into their IPA phoneme representations and train a GPT-2 model on it.[3] For TT-Keyword, instead of training a causal phoneme LM, we train to generate from keywords, using the (X [SEP] Y) template previously described. While there are multiple g2p phonemization toolkits, there are no readily available p2g toolkits that work well. Hence, we train our own p2g model. We treat this as a seq2seq translation task, once again using WikiText. We train a BART model with the phonemes as the inputs and the graphemes as the targets. Once both the p2p generation and the p2g translation models are trained, we then fine-tune the p2p models on the tongue twister phonemes (all steps similar to g2g), then use the p2g model to retrieve the grapheme representation of the outputs (see Figure 2). Lastly, since there is no capitalization in the phoneme space, we have to fix the capitalization of the generated outputs. We use the FastPunct library for this.[4] Unlike the previous g2g methods, we only use GPT-2 (and not GPT-J) for **PANCETTA-P** because GPT-J is too large to pretrain in a reasonable fashion.

4. **PANCETTA-J** (Joint) – One drawback of **PANCETTA-P** is that because we do our own pre-training on the phoneme space, we are not able to leverage the existing pre-training of large language models. In order to leverage both the phoneme representations and the pre-training of GPT, we propose **PANCETTA-J**. This is similar to g2g, but instead of only training with the template (X [SEP] Y), we train with 4 different templates, representing 4 different modalities. Specifically, we train with the templates (PK_G [SEP] TT_G), <PK_P [SEP] TT_P>, [TT_G [SEP] TT_P], and {TT_P [SEP] TT_G}, where PK represents the prompts/keywords, TT represents the tongue twisters, G represents the grapheme representation, and P represents the phoneme representation. These 4 modalities represent g2g, p2p, g2p, and p2g respectively. Here, the type of surrounding brackets function similar

---

[3] The `deep-phonemizer` Python package was used for g2p transliteration.

[4] https://github.com/notAI-tech/fastPunct

to custom tokens which indicate the modality for the model.

At test time, the model can be directly decoded using g2g mode without requiring phoneme information. We hypothesize that the phonetic structures learned during training time can serve as an effective "scaffold" — being used explicitly only during finetuning time (Swayamdipta et al., 2018).

3.2 Evaluation Metrics

3.2.1 Automatic Evaluation

As described in §1, a good tongue twister needs to satisfy two criteria: it needs to be both difficult to pronounce, as well as semantically coherent. We consider these two notions separately.

1. **Phonetic Difficulty**: We fine-tune a pretrained BERT-base classifier to differentiate between tongue twisters and regular sentences. To train this model, we use the parallel dataset of (TT, non-TT) pairs as described in §2.2. We specifically use these (TT, non-TT) pairs so that the model learns to classify based on phonetic difficulty rather than semantics. However, as mentioned in §2.2, some replacement sentences may sound unnatural. To ensure that the classifier learns to differentiate tongue twisters instead of picking up on these false signals, we augment our dataset with additional negative examples consisting of 500 sentences randomly sampled from WikiText. Rather than directly training on the sentences, we first convert the sentences to phoneme representations and train a classifier on the phonemes. This trained BERT classifier achieves an 83.4 F1-score on the test set of the parallel dataset, indicating that it is indeed successful at discriminating phonetically easy and difficult sentences. To further verify that this metric successfully measures phonetic difficulty, we show that it correlates well with human-annotated measures of phonetic difficulty (§5.2 and Table 6).

2. **Fluency:** We not only want phonetically difficult sentences; they must also be fluent and coherent. To measure this, we use the generation (log-perplexity) losses from a pretrained GPT-2.

3. **Keyword Relevance** (only for TT-Keyword): In TT-Keyword, we want to ensure that the generated tongue twister is semantically similar to the keywords used. To measure this, we use the BERT embedding of keywords and compare it with the embedding of the target sentence. More specifically, we use the BERTScore (Zhang* et al., 2020) between the generated sentence and the "sentence" consisting of the keywords separated by commas.

3.2.2 Human Evaluation

The human evaluation metrics are very similar to the ones in §3.2.1. These are as follows: (1) **Phonetic Difficulty** ("*How hard is the sentence to pronounce? To get a better sense of the difficulty, try saying the sentence out loud, quickly, and multiple times.*") and (2) **Fluency** ("*Does it sound like good English with good grammar?*") Evaluations were done on a scale of 1-5, with 5 being the highest. Further details about human evaluation are in §4.2.

4 Experimental Setup

4.1 Implementation Settings

Prompt / Keyword Extraction: To extract prompts for TT-Prompt, we simply consider the first three words of each sentence by checking for the whitespace character. To extract keywords for TT-Keyword, we use the KeyBERT library (Grootendorst, 2020), which returns keywords ranked by their cosine similarity scores to the entire sentence itself. For each sentence, we consider the top 5 keywords as our set of keywords. When a sentence has <5 keywords, we simply take all the keywords. In our dataset, 39.56% of the examples had <5 keywords.

Dataset splits: We split *TT-Corp* into a training-validation-test split with a 70-15-15 ratio. We use the same splits across all models and across both TT-Prompt and TT-Keyword task settings, as well as for training the phonetic difficulty classifier.

GPT-2 fine-tuning (g2g, **PANCETTA-P**, **PANCETTA-J**): We use the pretrained GPT2-base (124M params.) and fine-tune for 5 epochs with a learning rate of 5e-5 and 100 warmup steps.

BART pretraining (**PANCETTA-P**): For the p2g model, we pretrain BART-base (139M params.) on WikiText-103 phonemes. We split WikiText into train-validation-test splits of 80-10-10. The final training set has size 523k. Training was done for 20 epochs with batch size of 16, learning rate of 5e-4, and weight decay of 0.1 with a cosine scheduler.

BART & T5 fine-tuning (Style Transfer): We fine-tune BART-large (406M params.) & T5-large (737M params.) for 30 epochs with a batch size of 16, learning rate of 2e-5, and 400 warmup steps.

GPT-J fine-tuning (g2g, **PANCETTA-J**): Because GPT-J is too large (6B parameters), we use

	TT-Prompt		TT-Keyword		
Method	Phon. Difficulty	Fluency ↓	Phon. Difficulty	Fluency ↓	Keyword Relevance
g2g (GPT-2)	0.774	5.433	0.786	5.224	0.795
g2g (GPT-J)	0.848	5.643	0.856	5.593	0.794
Style Transfer (GPT-2+BART)	0.672	4.356	0.472	**3.662**	0.783
Style Transfer (GPT-2+T5)	0.631	**4.256**	0.414	4.309	0.780
PANCETTA-P (GPT-2+BART)	0.794	5.986	0.871	6.596	0.801
PANCETTA-J (GPT-2)	0.785	5.244	0.803	5.058	**0.803**
PANCETTA-J (GPT-J)	**0.866**	5.718	**0.888**	5.169	0.800
Gold Outputs	0.925	5.745	0.925	5.745	0.812

Table 4: Automatic evaluation averages for both TT-Prompt and TT-Keyword. The best-scoring method for each metric is highlighted in **bold**. Higher scores are better for all metrics except for fluency.

a compressed version of GPT-J, [5] which incorporates various techniques such as 8-bit quantization (Dettmers et al., 2021) and low-rank adaptation (Hu et al., 2021). To fine-tune, we use 10 epochs, a batch size of 1, and a learning rate of 1e-5.

For the GPT models (i.e. GPT-2 and GPT-J), generation was done using nucleus sampling (p=1.0, temperature=0.8). Meanwhile, for the BART and T5 models, generation was done using beam search with a beam size of 5. Additional hyperparameters and details on implementation can be found in Appendix C.

4.2 Human Evaluation Settings

Human evaluation was done on Amazon Mechanical Turk (AMT). We selected annotators with >97% HIT approval rate from Anglophone countries. [6] In each HIT, we present the generated outputs for each example in randomized order, and each test example was evaluated by exactly 2 annotators.

We conduct two rounds of annotation, one for TT-Prompt and another for TT-Keyword. Within each round, we further subdivide annotating GPT-2 experiments and GPT-J experiments. This GPT-2/GPT-J split only applies to g2g and **PANCETTA-J** models; for the style-transfer and **PANCETTA-P** models, we keep the same models for both rounds of evaluation. This is done to ensure that we only have one independent variable so that the changes in performance are due to the methodologies rather than the size of the models. Another reason for subdividing GPT-2/GPT-J experiments is so that we do not subject annotators to information overload from having to annotate too many similar examples. Owing to the same consideration, we also decided to omit human evaluation on the Style Transfer T5 baseline because we found it very similar to Style Transfer BART.

5 Results and Analysis

5.1 Automatic Evaluation Results

Table 4 shows the average results for the metrics outlined in §3.2.1. From the phonetic difficulty results, we see that our proposed **PANCETTA** models score higher than the baselines. More specifically, comparing g2g (GPT-2) (0.774) vs **PANCETTA-J** (GPT-2) (0.785) and comparing g2g (GPT-J) (0.848) vs **PANCETTA-J** (GPT-J) (0.866), we see that incorporating phoneme representations indeed aids in producing more phonetically difficult sentences. This pattern also holds true for TT-Keyword, where both GPT-2 and GPT-J see increases in performance after incorporating phonemes. We also observe that **PANCETTA-P** performs reasonably well in phonetic difficulty and has the highest score of the non-GPT-J models for both TT-Prompt and TT-Keyword. In fact, for TT-Keyword, **PANCETTA-P** is able to get very close to **PANCETTA-J** (GPT-J), which is remarkable, considering that it is only using a GPT-2 model.

For fluency, style transfer models score better than the other models. This is likely because style transfer models first generate a regular sentence, then attempt to *"tongue twisterize"* it. However, there is no guarantee that the sentence can even be reasonably converted to a tongue twister, resulting in minimal changes being made to the original GPT-generated sentence, thereby leading to good fluency scores when using perplexity. Meanwhile, we see that **PANCETTA-P** has the worst perplexity score. However, it is important to note that even the ground truth tongue twisters score poorly here (around the same scores as **PANCETTA** models). Tongue twisters do not usually use typical English tokens in standard sequences, thereby resulting in worse perplexity scores. Nonetheless, fluency/perplexity is still important as a sanity check for the basic qualities of the generated text (whether the generation is coherent, if it makes unnecessary

---

[5] https://huggingface.co/hivemind/gpt-j-6B-8bit

[6] More details about the human eval are in Appendix B.

Method	TT-Prompt		TT-Keyword	
	Phonetic Difficulty	Fluency	Phonetic Difficulty	Fluency
g2g (GPT-2)	3.056	3.736	3.847	**4.139**
Style Transfer (GPT-2+BART)	2.569	3.639	3.500	3.819
PANCETTA-P (GPT-2+BART)	**3.528**	**3.778**	3.722	3.764
PANCETTA-J (GPT-2)	3.153	3.764	**3.889**	3.931
Gold Outputs	3.361	3.931	3.833	4.000
g2g (GPT-J)	3.521	**3.979**	3.791	3.708
Style Transfer (GPT-2+BART)	3.271	3.75	3.25	3.750
PANCETTA-P (GPT-2+BART)	**3.854**	3.708	3.833	**3.896**
PANCETTA-J (GPT-J)	3.708	3.646	**3.979**	3.604
Gold Outputs	3.750	4.000	4.104	3.729

Table 5: Human evaluation averages for TT-Prompt and TT-Keyword. The top half of the table correspond to methods using GPT-2 as the base architecture, while the bottom half of the table correspond to methods using GPT-J as the base architecture. Top method scores for each metric are highlighted in **bold**.

grammatical errors, etc.). To this end, the fluency scores of **PANCETTA** models are adequate.

Lastly, for keyword relevance, most scores are close to each other. The three **PANCETTA** models have the three highest scores, indicating that **PANCETTA** is able to generate difficult tongue twisters without compromising the task at hand.

5.2 Human Evaluation Results

Table 5 shows the average results for the human evaluation. As with the automatic evaluations, we see an increase in phonetic difficulty when we introduce phonemes into the training process. Comparing g2g (GPT-2) and **PANCETTA-J** (GPT-2), we see an increase from 3.056 to 3.153 for TT-Prompt and from 3.847 to 3.889 for TT-Keyword. This trend also occurs for GPT-J models. Despite not being able to leverage existing GPT pretraining, **PANCETTA-P** also works very well, outperforming all non- **PANCETTA** models in all but one setting. These positive results indicate that incorporating phonetic information is indeed helpful.

In terms of phonetic difficulty, we observe that for TT-Prompt, **PANCETTA-P** works best for both GPT-2 and GPT-J, while for TT-Keyword, **PANCETTA-J** works best for both GPT-2 and GPT-J. This may be because generating from keywords is generally more difficult than completing a prompt, so **PANCETTA-J** benefits from existing pretraining. Meanwhile, in terms of fluency, g2g methods work best for 2 settings, and **PANCETTA-P** works best for 2 settings. Tongue twisters usually use words in creative and unnatural-sounding ways, and this may sometimes negatively affect fluency, so the "most fluent" sentence may not necessarily be the best tongue twister. Nevertheless, all the fluency metrics for **PANCETTA** models are around 3.7 to 3.8, indicating good fluency. Overall, we conclude that **PANCETTA** substantially improves phonetic difficulty while maintaining a competitive level of fluency.

Task	Pearson Correlation	Spearman Correlation
TT-Prompt	0.117 (p=0.076)	0.116 (p=0.079)
TT-Keyword	0.102 (p=0.177)	0.134 (p=0.075)

Table 6: Corr. between human annotations and automatic metrics (BERT classifier) for phonetic difficulty.

To verify the validity of using our BERT classifier to automatically measure phonetic difficulty, we compute correlations between the classifier's scores and human-annotated phonetic difficulty scores (Table 6). We observe that the correlations are > 0.1 with relatively low p-values, which is an adequate score when comparing various automatic and human-annotated metrics (Gangal et al., 2022). While this can be further improved and is a limitation of our work, we also note that the automatic phonetic difficulty metric scores the gold outputs the highest (at 0.925) and the style transfer models lowest, which indicates that it is able to decently capture the notion of phonetic difficulty.

5.3 Qualitative Analysis

Table 7 shows sample generations for both TT-Prompt and TT-Keyword. We observe that both PAN-P and PAN-J are able to use a wide variety of tongue twister techniques, such as rhyme (*grape/crepe/crate-* in ex.1 PAN-P), alliteration (*kneadle/knuckle* in ex.3 PAN-J), alternating final sounds (*land/lamb* in ex.2 PAN-P), alternating initial sounds (*six-/sheik* in ex.4 PAN-J), and repetition. They are also able to generate proper nouns to suit the sentence, such as "*Donna*" and "*Needles Nood*" in ex.3 PAN-P. They can also combine multiple such techniques in a single tongue twister, such as *stick/stock* and *land/lamb* in ex.2 PAN-P.

Comparing this with the baseline methods (g2g and Style T.), we see that the generated outputs of the g2g baseline are decent and somewhat tongue

Method	**Text**
Inputs	(ex.1) The pirates pirated ...
Gold out.	The pirates pirated private property.
g2g	The pirates pirated the pepper pot.
Style T.	The pirates pirated more than $50 million in online gambling in 2013.
PAN-P	The pirates pirated a little grape crepe. The pirate dread cratered a little.
PAN-J	The pirates pirated the paired prince with a pair of paired pistols.
Inputs	(ex.2) If you stick ...
Gold out.	If you stick a stock of liquor in your locker, it is slick to stick a lock upon your stock or some joker who is slicker is going to trick you of your liquor if you fail to lock your liquor with a lock?
g2g	If you stick two sticks together, you get two sticks tied together.
Style T.	If you stick with the same line, you have different things going for you.
PAN-P	If you stick a stock of land in a lamb and a lamb in a farm.
PAN-J	If you stick a stock stick on the stock stick you stick, who's going to stick it on the stock stick you stick?

Method	**Text**
Inputs	(ex.3) needles, noodles, kneadle, don, need
Gold out.	I don't need your needles to kneadle noodles.
g2g	One noodle needs a kneeler to lean on
Style T.	Near the end of his reign, Leiblau had to send a delegate to Parliament, to support his friend King Gules of Northumbria.
PAN-P	If Donna Needles Need, who is Needles Nood, and the Need?
PAN-J	Don't kneadle a knuckle with a knuckle needle!
Inputs	(ex.4) thieves, thankful, thanked, thirty, thursday
Gold out.	The thirty-three thankful thieves thought that they thanked the other thirty-three thankful thieves throughout Thursday.
g2g	The thieve thanked the thankful thief on Thursday.
Style T.	Thanked by Thnx for the idea, I thought it was an idea that I wanted to do a spoof of Thankful Thankful and Thnx.
PAN-P	Thankful thieves thought that they thought they thrilled the throne throughout Thursday.
PAN-J	These sixty sheiks sent these thousand and sixty sheiks sixty sheiks thanking them for shouting these sixty sheiks sixty shouts on Thursday.

Table 7: Qualitative examples for both TT-Prompt (first 2 examples) and TT-Keyword (last 2 examples). We report only the best performing model based on phonetic difficulty from automatic evaluations for each type (in brackets): literal input, gold output, g2g (GPT-J), Style Transfer (BART), **PANCETTA-P** (GPT-2+BART), and **PANCETTA-J** (GPT-J). Additional examples can be found in Appendix D.

twister-like but usually are very short and simple, often relying too much on alliterations. Meanwhile, the outputs for the style transfer methods are generally not tongue twisters. As discussed in §5.1, this is likely because it commonly fails at fully converting a regular sentence into a tongue twister.

For TT-Prompt, we observe that even with a non-alliterative prompt such as "If you stick" (ex.2), the **PANCETTA** models can still generate good tongue twisters, whereas the g2g method attempts to use repetition but the generated text is not that difficult to pronounce. Meanwhile, for the TT-Keyword setting, PAN-J is able to incorporate the semantics of the words, rather than just copying the words themselves: in ex.4, PAN-J replaces "thirty/thieves" in the keywords with "sixty/sheiks". Lastly, comparing PAN-P and PAN-J, we see that PAN-J sentences generally sound smoother, while PAN-P sentences sometimes end rather abruptly ("*in a farm.*" in ex.2; "*and the Need?*" in ex.3). In addition, some of the outputs from PAN-P lack coherence (ex.3). This is likely because PAN-P uses a phoneme language model and hence is unable to leverage the large-scale pretraining from GPT models. On the other hand, this lack of large-scale pretraining can potentially free up PAN-P to use more diverse tongue twister techniques, such as rhymes (ex.1) and proper nouns (ex.3) which are less common in PAN-J.

6 Related Work

Automatic tongue twister generation is a largely unexplored task. Existing systems mostly use synonym replacements (Zeng, 2019) on existing tongue twisters, which requires a large list of tongue twisters to begin with and cannot generate novel ones from scratch. Carey (2017) generates tongue twisters using sound vectors, and Joshipura (2020) trains an LSTM on a small tongue twister dataset, but neither are able to produce novel and semantically coherent examples. Furthermore, no methods currently exist for the TT-Keyword task.

There have been multiple studies on creative generation of various figures of speech such as similes (Chakrabarty et al., 2020), metaphors (Chakrabarty et al., 2021), and personifications (Keh et al., 2022). However, these other creative linguistic constructs don't require working with another modality in the same way that tongue twister generation relies on phonemes. Among these creative linguistic constructs, the closest ones to tongue twisters would likely be alliterations (Hopkins and Kiela, 2017), rhymes (Xue et al., 2021), and poetry (Ghazvininejad et al., 2017; Cruys, 2020). Notably, DeepHaiku (Gonsalves, 2022) also explores using phonemes and a multitask objective in order to generate haikus. However, tongue twister generation goes beyond alliterations and rhymes; rather it is a mix of all these various techniques. In addition, it differs from poetry generation because poetry generation focuses on generating rhythmic verses and syllables, whereas the main focus of tongue twister generation is on phonetic difficulty.

Using phonemes in language modeling has been previously explored in the speech domain for automatic speech recognition (Sundararaman et al., 2021; Belinkov et al., 2019; Xu et al., 2021). In this paper, we trained a BART model to do p2g translation. Other existing methods include expectation maximization (Knight et al., 2006), A* search (Corlett and Penn, 2010), and Hidden Markov Models (Hopkins and Kiela, 2017).

There is also work on more general constrained text generation tasks. An example is Feng et al. (2019), who propose Semantic Text Exchange to adjust a text's topic-level semantics. Lin et al.

(2020) introduce a generative commonsense reasoning task using keyword-to-text generation called CommonGen. SAPPHIRE (Feng et al., 2021b) and VisCTG (Feng et al., 2022) investigate approaches to improve performance on CommonGen, the latter using per-example visual grounding.

7 Conclusion and Future Work

In this paper, we proposed the task of automatic tongue twister generation, and explored it under two settings: TT-Prompt and TT-Keyword. We curated a dataset called *TT-Corp* of 600+ English tongue twisters from various sources and proposed **PANCETTA**, a training methodology which incorporates phoneme representations. We implemented two variants: **PANCETTA-P** (Phoneme), which trains a phoneme-based language model, and **PANCETTA-J** (Joint), which jointly incorporates both phoneme-level information and grapheme-level information during training time. Through empirical results and qualitative evaluations, we showed that incorporating phonemes is indeed helpful in producing effective tongue twisters which are harder to pronounce while staying fluent.

While **PANCETTA** works well at generation, the generation process lacks interpretability. This is most notable when looking at the phonetic difficulty classifier. Currently, the classifier does not identify and separately score elements of phonetic difficulty or come up with an explicit decomposition. We believe that such explicit decomposition can be very useful in the future for understanding more about tongue twisters and the "mouth model" discussed in §1. In addition, the procedure of fine-tuning a pretrained model in 4 different modes involving both graphemes and phonemes, devised here to fine-tune **PANCETTA-J**, can also be adopted more generally as a data augmentation (Feng et al., 2021a, 2020) method for LM fine-tuning, creating a (pseudo) count of 4N training examples given N initial ones. Lastly, tongue twisters can potentially be incorporated in dialogue agents, adding creativity and personality (Li et al., 2020).

Limitations

As mentioned in §7, our current model and classifer are deficient in terms of their interpretability on certain aspects, and would greatly benefit from addition to their interpretability on these fronts.

Our models and datasets are limited to English tongue twisters (Bender and Friedman, 2018). In addition, when we convert to the phoneme space, we only use IPA phonemes. We selected IPA because it is the most common and most readily available phonemization method. However, there also exist many other phonemization methods such as the ARPAbet / CMU Pronouncing Dictionary (Weide, 2005), the SAMPA (Wells, 2005), or the Festival phonemization scheme (Black and Lenzo, 2003). We also choose to use deep-phonemizer for g2p transliteration. There are many other available phonemization tools such as Epitran (Mortensen et al., 2018) and the `phonemizer` Python package (Bernard and Titeux, 2021). It would be interesting to explore how the performance will change if we try other phoneme alphabets or phonemization methods.

In addition, we see in §5 that the style transfer models do not really work that well. In this paper, we only tried simple BART and T5 seq2seq models. One possible way to expand on this would be to try other more sophisticated style transfer methods.

Due to computational resources, we were not able to explore larger models and had to use a compressed version of GPT-J, which may have slightly affected the performance.

Ethics

The *TT-Corp* dataset we propose and release herein has undergone a per-example, manual, vetting process during its curation and pre-processing stage, as described further in Appendix A.2, which removes examples which may exhibit offensive words, profanities, racism, gender bias, and other malicious language.

We do collect human evaluation ratings using crowd-sourcing, specifically through AMT. However, we neither solicit, record, nor request any kind of personal or identity information from the annotators. Our AMT annotation was conducted in a manner consistent with terms of use of any sources and intellectual property and privacy rights of AMT crowd workers. Crowdworkers were fairly compensated: $0.56 per fluency and phonetic difficulty HIT, for roughly 2 min tasks. This is at least 2 times the minimum U.S.A. wage of $7.25 per hour ($0.56 per 2 mins is around $16.8 per hour).

NLG models are known to suffer from biases learnable from training or finetuning on data, such as gender bias (Dinan et al., 2020). However, our work and contribution does not present or release any completely new model architectures, and is primarily concerned with more careful adaptation and finetuning of existing pretrained models for a particular class of creative linguistic constructs (i.e. tongue twisters). The frailties, vulnerabilities, and potential dangers of these models have been well researched and documented, and a specific re-investigation would be repetitive and beyond the scope and space constraints of this paper.

We do not foresee any explicit way that malicious actors could specifically misuse finetuned models that could be trained on our data, beyond the well-researched, aforementioned misuse that is possible in general with their instantiation for any transduction task or dataset (e.g. summarization).

References

Akintunde Akinyemi. 2003. Yorùbá oral literature: A source of indigenous education for children. *Journal of African Cultural Studies*, 16(2):161–179.

Yonatan Belinkov, Ahmed M. Ali, and James R. Glass. 2019. Analyzing phonetic and graphemic representations in end-to-end automatic speech recognition. *ArXiv*, abs/1907.04224.

Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Mathieu Bernard and Hadrien Titeux. 2021. Phonemizer: Text to phones transcription for multiple languages in python. *Journal of Open Source Software*, 6(68):3958.

Alan Black and Kevin Lenzo. 2003. Building synthetic voices.

CJ Carey. 2017. Twisty tongue. `https://github.com/perimosocordiae/twisty_tongue`.

Tuhin Chakrabarty, Smaranda Muresan, and Nanyun Peng. 2020. Generating similes effortlessly like a pro: A style transfer approach for simile generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6455–6469, Online. Association for Computational Linguistics.

Tuhin Chakrabarty, Xurui Zhang, Smaranda Muresan, and Nanyun Peng. 2021. MERMAID: Metaphor generation with symbolism and discriminative decoding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4250–4261, Online. Association for Computational Linguistics.

Eric Corlett and Gerald Penn. 2010. An exact A* method for deciphering letter-substitution ciphers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1040–1047, Uppsala, Sweden. Association for Computational Linguistics.

T. V. D. Cruys. 2020. Automatic poetry generation from prosaic text. In *ACL*.

Aliya Deri and Kevin Knight. 2015. How to make a frenemy: Multitape FSTs for portmanteau generation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 206–210, Denver, Colorado. Association for Computational Linguistics.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2021. 8-bit optimizers via block-wise quantization.

Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. 2020. Queens are powerful too: Mitigating gender bias in dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8173–8188.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Steven Y. Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. 2020. GenAug: Data augmentation for finetuning text generators. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 29–42, Online. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021a. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Steven Y. Feng, Jessica Huynh, Chaitanya Prasad Narisetty, Eduard Hovy, and Varun Gangal. 2021b. SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Steven Y. Feng, Aaron W. Li, and Jesse Hoey. 2019. Keep calm and switch on! preserving sentiment and fluency in semantic text exchange. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2701–2711, Hong Kong, China. Association for Computational Linguistics.

Steven Y. Feng, Kevin Lu, Zhuofu Tao, Malihe Alikhani, Teruko Mitamura, Eduard Hovy, and Varun Gangal. 2022. Retrieve, caption, generate: Visual grounding for enhancing commonsense in text generation models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10618–10626.

Varun Gangal, Steven Y. Feng, Malihe Alikhani, Teruko Mitamura, and Eduard Hovy. 2022. Nareor: The narrative reordering problem. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10645–10653.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

Robert A. Gonsalves. 2022. Deep haiku: Teaching gpt-j to compose with syllable patterns.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Jack Hopkins and Douwe Kiela. 2017. Automatically generating rhythmic verse with neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 168–178, Vancouver, Canada. Association for Computational Linguistics.

Nabil Hossain, John Krumm, Lucy Vanderwende, Eric Horvitz, and Henry Kautz. 2017. Filling the blanks (hint: plural noun) for mad Libs humor. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 638–647, Copenhagen, Denmark. Association for Computational Linguistics.

Edward Hu, Yelong Shen, Phil Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Bashir Jam. 2018. Features of tongue twisters and analysis of their structure in persian. *Journal of Researches in Linguistics*, 9(2):59–76.

Marilyn Jorgensen. 1981. The tickled, tangled, tripped, and twisted tongue: A linguistic study of factors relating to difficulty in the performance of tongue twisters. *New York Folklore*, 7(3):67.

Dweep Joshipura. 2020. Tongue twister generation. `https://www.kaggle.com/code/djathidiro/tongue-twister-generation/notebook`.

Sedrick Scott Keh, Kevin Lu, Varun Gangal, Steven Y. Feng, Harsh Jhamtani, Malihe Alikhani, and Eduard Hovy. 2022. PINEAPPLE: Personifying INanimate entities by acquiring parallel personification data for learning enhanced generation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6270–6284, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Heather Kember, Kathryn Connaghan, and Rupal Patel. 2017. Inducing speech errors in dysarthria using tongue twisters. *International journal of language & communication disorders*, 52(4):469–478.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 499–506, Sydney, Australia. Association for Computational Linguistics.

Peter Ladefoged and Keith Johnson. 2014. *A course in phonetics*. Cengage learning.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Aaron W. Li, Veronica Jiang, Steven Y. Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. 2020. Aloha: Artificial learning of human attributes for dialogue agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8155–8163.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Elisa Mattiello. 2013. Extra-grammatical morphology in english: Abbreviations, blends, reduplicatives and related phenomena.

Jonathan Mcgovern. 2021. Three Tudor Tongue-Twisters. *Notes and Queries*, 68(4):392–393.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models.

David R. Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: Precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France. European Language Resources Association (ELRA).

Aakanksha Naik. 2022. *Domain Adaptation for the Long Tail in Language Understanding*. Ph.D. thesis, Carnegie Mellon University Pittsburgh, PA.

Danijela Prošić-Santovac. 2009. The use of tongue twisters in efl teaching. *Annual Review of the Faculty of Philosophy/Godisnjak Filozofskog Fakulteta*, 34.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

M Revathy and K Ravindran. 2016. Enhancing effective speaking skills through role play and tongue twisters. *Language in India*, 16(9).

Mukuntha Narayanan Sundararaman, Ayush Kumar, and Jithendra Vepa. 2021. Phoneme-bert: Joint language modelling of phoneme sequence and asr transcript. *arXiv preprint arXiv:2102.00804*.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A Smith. 2018. Syntactic scaffolds for semantic structures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3772–3782.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Robert Weide. 2005. The carnegie mellon pronouncing dictionary.

John Wells. 2005. Sampa - computer readable phonetic alphabet.

Qiantong Xu, Alexei Baevski, and Michael Auli. 2021. Simple and effective zero-shot cross-lingual phoneme recognition. *ArXiv*, abs/2109.11680.

Lanqing Xue, Kaitao Song, Duocai Wu, Xu Tan, Nevin L. Zhang, Tao Qin, Wei-Qiang Zhang, and Tie-Yan Liu. 2021. DeepRapper: Neural rap generation with rhyme and rhythm modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 69–81, Online. Association for Computational Linguistics.

Faith Zeng. 2019. Tongue twister generator.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Additional Details — Dataset Collection

A.1 Sources

We curate our tongue twisters from a heterogeneous mix of online sources, including but not limited to the ones listed below:

1. University of Arkansas
2. The r/tonguetwister Subreddit
3. Various AskReddit threads
4. Mondly.com
5. Uebersetzung
6. Marcus Stuart's LOL Tongue Twisters book
7. Language Avenue
8. Bilingual Monkeys
9. Pun.me
10. ESL
11. Sweetrhymes
12. EngVid
13. IvyPanda

A.2 Vetting

We then perform the following filtering steps to retain only a collection of high-quality dataset examples:

- Remove near-repetitive examples to ensure each example is unique
- Remove excessively short or meaningless examples lacking sentence structure, e.g. *blue blood, bad blood*
- Remove poems or rhymes
- Remove examples containing offensive words, racism, gender bias or other harmful and malicious language of any nature, to prevent models learnt from this data from further ingraining or amplifying such phenomena.

B Appendix B: Evaluation Details

To prevent annotator judgements for one attribute from inadvertently influencing the other, we conduct the studies for soliciting Fluency and Phonetic Difficulty scores separately.

Averaging over the 4 settings described in §4.2 (TT-Keyword/TT-Prompt × GPT-2/GPT-J), a total of 20 unique AMT annotators participated in the study for Fluency, each performing 3.6 HITs on average. Annotators were compensated $0.56 per HIT, each of which was designed to take < 2 mins on average.

Averaging over the 4 settings, 16.51 unique AMT annotators participated in the second, separate study for Phonetic Difficulty, each performing 4.36 HITs on average. Annotators were compensated $0.56 per HIT, each of which was designed to take < 2 mins on average.

C Further Implementation Details

In §4.1, we detailed the hyperparameters used for pre-training BART, as well as for fine-tuning GPT-2, GPT-J, BART, and T5. We conduct a hyperparameter search to check which values led to the best performance. For learning rate, we tried {1e-6, 5e-6, 1e-5, 2e-5, 2e-5, 1e-4}; for batch size, we tried {2,4,8,16}; and for number of epochs, we tried {2, 5, 10, 20}. These search bounds were selected based on known commonly-used values for these models. We start with a baseline model of lr=2e-5, bsz=8 and 10 epochs, and individually change each setting to investigate its effect on performance. One trial was conducted per hyperparameter setting. We use a maximum sequence length of 256. In terms of other hyperparameters, we mostly used default values which are known to work for these models. This includes the warmup steps and learning rate decays, which we detail in §4.1. (Note: the above hyperparameter search settings are for fine-tuning. We could not do an extensive hyperparameter search for pre-trainig due to time constraints. We ran pre-training twice to test the effect of learning rate 1e-4 vs. 5e-4, and ultimately selected 5e-4.)

Model selection was done based on the epoch with the best validation loss. We report the best validation losses for each training process below: GPT-2 pre-training=0.77, GPT-2 fine-tuning (TT-Prompt)=1.08, GPT-2 fine-tuning (TT-Keyword) =0.95, GPT-J fine-tuning (TT-Prompt)=1.23, GPT-J fine-tuning (TT-Keyword)=1.19, BART-fine-tuning (TT-Keyword) = 1.31, BART-fine-tuning (TT-Prompt) = 1.30, T5-fine-tuning (TT-Keyword) = 1.45, T5-fine-tuning (TT-Prompt) = 1.52.

All training was done on Google Colaboratory environments using V100 GPUs with 16GB memory. Pre-training GPT-2 for the p2p model took 90 minutes per epoch, and pre-training BART for the p2g model took 70 minutes per epoch. In terms of fine-tuning on *TT-Corp*, GPT-2 takes around 1 minute per epoch, GPT-J around 2 minutes per epoch, BART around 1 minute per epoch, and T5 around 1 minute per epoch.

D Additional Qualitative Examples

In §5.3, we only report the best performing models for each of the methods. In Table 8, we extend this to all models for all the methods. We also provide a few more examples for both TT-Prompt and TT-Keyword.

Method	**Text**
Inputs	(ex.1) The pirates pirated ...
Gold out.	The pirates pirated private property.
g2g (GPT2)	The pirates pirated the property.
g2g (GPTJ)	The pirates pirated the pepper pot.
Style T. (BART)	The pirates pirated more than $50 million in online gambling in 2013.
Style T. (T5)	The pirates pirated more than $50 million in online gambling in 2013, according to a government-backed study.
PAN-P	The pirates pirated a little grape crepe. The pirate dread cratered a little.
PAN-J (GPT2)	The pirates pirated the music and the Roman river carried the criminals off.
PAN-J (GPTJ)	The pirates pirated the paired prince with a pair of paired pistols.
Inputs	(ex.2) If you stick ...
Gold out.	If you stick a stock of liquor in your locker, it is slick to stick a lock upon your stock or some joker who is slicker is going to trick you of your liquor if you fail to lock your liquor with a lock?
g2g (GPT2)	If you stick a weight silky liquor on your wrist, you should stick a weight silky liquor on your inner wrist.
g2g (GPTJ)	If you stick two sticks together, you get two sticks tied together.
Style T. (BART)	If you stick with the same line, you have different things going for you.
Style T. (T5)	If you stick with the same line, you have different things going for yout.
PAN-P	If you stick a stock of land in a lamb and a lamb in a farm.
PAN-J (GPT2)	If you stick a stock of liquor in your locker it is slick to put a lock on your stock.
PAN-J (GPTJ)	If you stick a stock stick on the stock stick you stick, who's going to stick it on the stock stick you stick?
Inputs	(ex.3) I saw Susie ...
Gold out.	I saw Susie sitting in a shoe shine shop.
g2g (GPT2)	I saw Susie sitting on the sofa with a big black basket of biscuits.
g2g (GPTJ)	I saw Susie sitting on a buttercup.
Style T. (BART)	I saw Susie and then she came and took me to the kitchen where we cleaned.
Style T. (T5)	I saw Susie and then she came and took me to the kitchen where we cleaned up and we started talking about the new dog," she recalled.
PAN-P	I saw Susie sitting on a seasawe horse. and I saw haunted houses.
PAN-J (GPT2)	I saw Susie sitting in a speck of sheared sheep's sheep's sheep's sheep skin.
PAN-J (GPTJ)	I saw Susie sitting on a butter bucket with a black butter bucket sat on her butter bucket.

Method	**Text**
Inputs	(ex.4) needles, noodles, kneadle, don, need
Gold out.	I don't need your needles to kneadle noodles.
g2g (GPT2)	Don's dirt feet meet your earrings at the bottom of the dirt cany bore trail.
g2g (GPTJ)	One noodle needs a kneeler to lean on
Style T. (BART)	Near the end of his reign, Leiblau had to send a delegate to Parliament, to support his friend King Gules of Northumbria.
Style T. (T5)	Near the end of his reign, Leiblau had to send a delegate to Parliament, to support his friend King Gules of Northumbria.
PAN-P	If Donna Needles Need, who is Needles Nood, and the Need?
PAN-J (GPT2)	If you crave a soft chew toy, buy a chew toy that needs to chew.
PAN-J (GPTJ)	Don't kneadle a knuckle with a knuckle needle!
Inputs	(ex.5) thieves, thankful, thanked, thirty, thursday
Gold out.	The thirty-three thankful thieves thought that they thanked the other thirty-three thankful thieves throughout Thursday.
g2g (GPT2)	Twelve thieves today took sixty dollars from Thies' birthday party.
g2g (GPTJ)	The thieve thanked the thankful thief on Thursday.
Style T. (BART)	Thanked by Thnx for the idea, I thought it was an idea that I wanted to do a spoof of Thankful Thankful and Thnx.
Style T. (T5)	Thanked by Thnx for the idea, I thought it was an idea that I wanted to do a spoof of Thankful Thankful and Thnx, so it went to the video website, Twitter, and did some research on a T shirt.
PAN-P	Thankful thieves thought that they thought they thrilled the throne throughout Thursday.
PAN-J (GPT2)	I'm grateful tonight for thanking the valiant brave thieves.
PAN-J (GPTJ)	These sixty sheiks sent these thousand and sixty sheiks sixty sheiks thanking them for shouting these sixty sheiks sixty shouts on Thursday.
Inputs	(ex.6) snails, silently, sailed, slimy
Gold out.	Six slimy snails sailed silently.
g2g (GPT2)	Six silently sullen sullen snails slid silently across the sea.
g2g (GPTJ)	Slimy snails sailed silently.
Style T. (BART)	The ship was spotted by U.S. Coast Guard helicopter from a distance of four thousand, seven hundred nautical miles.
Style T. (T5)	The ship was spotted by Uas was was was was by by by by by by by.
PAN-P	Sailed from static line seven and all silently from Trondheim at seven on Sunday night.
PAN-J (GPT2)	Sailing silently on the sleigh.
PAN-J (GPTJ)	Six squandered snails silently sailed in a slivy ship.

Table 8: Additional qualitative examples for both TT-Prompt (first 3) and TT-Keyword (last 3): literal input, gold output, g2g (GPT-2), g2g (GPT-J), Style Transfer (BART), Style Transfer (T5), **PANCETTA-P** (GPT-2+BART), **PANCETTA-J** (GPT-2), and **PANCETTA-J** (GPT-J).

A User-Centered, Interactive, Human-in-the-Loop Topic Modelling System

Zheng Fang[1], **Lama Alqazlan**[1], **Du Liu**[2], **Yulan He**[1,3,4] **and Rob Procter**[1,4]
[1]Department of Computer Science, University of Warwick, UK
[2]Bayes Business School, City, University of London, UK
[3]Department of Informatics, King's College London, UK
[4]The Alan Turing Institute, UK
`{Z.Fang.4, Lama.Alqazlan, Rob.Procter}@warwick.ac.uk`
`yulan.he@kcl.ac.uk, Du.Liu@city.ac.uk`

Abstract

Human-in-the-loop topic modelling incorporates users' knowledge into the modelling process, enabling them to refine the model iteratively. Recent research has demonstrated the value of user feedback, but there are still issues to consider, such as the difficulty in tracking changes, comparing different models and the lack of evaluation based on real-world examples of use. We developed a novel, interactive human-in-the-loop topic modeling system with a user-friendly interface that enables users compare and record every step they take, and a novel topic words suggestion feature to help users provide feedback that is faithful to the ground truth. Our system also supports not only what traditional topic models can do, i.e., learning the topics from the whole corpus, but also targeted topic modelling, i.e., learning topics for specific aspects of the corpus. In this article, we provide an overview of the system and present the results of a series of user studies designed to assess the value of the system in progressively more realistic applications of topic modelling.

1 Introduction

Huge amounts of unstructured, textual data are generated daily. As more data becomes available, it becomes more difficult to search, understand and discover the knowledge within it. Because of the human effort it requires, conventional qualitative approaches, such as Grounded Theory, (Glaser et al., 1968) are no longer feasible with such large volumes of data. Topic modelling is a potential solution that has received increasing attention in recent research (Heidenreich et al., 2019; Curiskis et al., 2020; Dantu et al., 2021; Goyal and Howlett, 2021) to help users organize, search, and understand large amounts of information. It is an unsupervised machine learning technique for identifying hidden topics in large, unstructured text corpora, in which a hidden topic is represented by a group of words describing a common theme. Users can easily identify the topics in each document and search for documents closely associated with a specific topic for a more in-depth study. However, the topics generated by conventional topic models are often incoherent and contain many unrelated words (Chang et al., 2009; Mimno et al., 2011; Boyd-Graber et al., 2014). Although these issues can be addressed by pre-processing the target data source, for example, by removing irrelevant words from the vocabulary list and adjusting the hyper-parameters of the model, such as the number of topics, this requires familiarity with the algorithm. Hence, it is difficult for anyone who does not have some knowledge of topic modelling.

Human-in-the-loop topic modelling (HL-TM) incorporates human knowledge into the topic modelling process to address the aforementioned issues. It allows users who are not experts in topic modelling to refine the model through a set of refinement operations, such as adding words to a topic, removing words from a topic, splitting topics, or merging topics (Jagarlamudi et al., 2012; Wang et al., 2012; Choo et al., 2013; Hoque and Carenini, 2015; Lund et al., 2017; Smith et al., 2018). While most of these studies did not feed the refinement operations into an iterative retraining process, Smith et al. (2018) implemented a fully interactive, user-centered HL-TM system, and examined how the user experience is affected by issues arising in interactive systems, such as unpredictability, trust and lack of control. However, there are still limitations to their work. First, their system only allows users to refine the model sequentially, meaning that once a user updates the model, a new model overrides the previous model. This prevents users from comparing the effects of applying different refinement operations to the same model, making it difficult to find the most appropriate ones. Furthermore, because the previous model is no longer accessible, users may find it challenging to decide whether

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 505–522
May 2-6, 2023 ©2023 Association for Computational Linguistics

the new version is really an improvement. Second, their system does not allow users to retract the changes they made after the underlying model was updated. This becomes a problem when users make inappropriate changes that lead to unexpected results. It is especially frustrating when a user has spent a lot of time refining the model and the whole effect is ruined by one inappropriate change. Third, they use Latent Dirichlet Allocation (LDA) (Blei et al., 2003) as their underlying model.

Since LDA is a full-analysis model that can only learn topics from the whole corpus, its application is limited to when users have no *apriori* assumptions about the topics of the corpus. While users can refine the model by adding prior knowledge in an attempt to turn an unrelated topic into one that focuses on the aspect of interest, the phenomenon of higher order co-occurrence in LDA (Heinrich, 2009) may prevent any infrequent words related to the aspect of interest being assigned to the unrelated topic. For instance, given a set of posts about health, researchers may wish specifically to analyze the impact of food on health. Researchers would add food-related words such as “food”, “eat” to an unrelated topic. If these words have a relatively low frequency of occurrence in the posts, then the system may not turn the unrelated topic into a food-related topic. Fourth, an important question that has not yet been explored is how to signal to users when their assumptions do not match reality (Kumar et al., 2019), which may bias the refinement process. For instance, a user may think that a technology-related topic should contain words like “apple, google", while the input corpus has the word “apple" related to fruit. By adding “apple" to the topic, the topic will be contaminated with fruit-related words. Such incorrect refinements would result in poor results.

To address these issues, we implemented a novel HL-TM system that supports six refinement operations, including *add words, remove words, swap word order, remove documents, merge topics*, and *split topic* (Lund et al., 2017; Smith et al., 2018). Unlike Smith et al. (2018), where users can only refine the model sequentially and cannot retract the changes they made, in our system, users can make different attempts at refinement to the same model node and compare the resulting models. A complete refinement history is also presented, allowing users to track their changes from the first step. These also ensure that users can revert to the previous model node when an inappropriate refinement operation is applied. Instead of using LDA as the underlying model, our system uses the query-driven topic model (QD-TM) from (Fang et al., 2021) as the underlying model. The advantage is that the QD-TM not only supports the full-analysis capabilities from LDA, i.e., learning the overall topics from the whole corpus, but also supports users in performing targeted analysis, i.e., learning topics focused on specific aspects of the corpus by mitigating the effects of higher order co-occurrence phenomenon in LDA (Fang et al., 2021).

Moreover, we also implemented a novel automatic topic words suggestion feature to guide users in adding appropriate words to the selected topic. This feature extracts a list of candidate words related to the topic from which users can select words to add to the topic. Our evaluation results demonstrate the usefulness of this feature, where the suggested words are closely related to the selected topic and align better with the ground truth. We also conducted a series of user studies designed to assess the value of the system on real world application scenarios.

This work makes the following contributions: (1) a novel interactive HL-TM system with an advanced user interface that allows users to train different models from the same model node and retract inappropriate changes applied; (2) the use of QD-TM as the underlying model to support both the full-analysis and targeted-analysis topic modelling capabilities; (3) a novel and efficient topic words suggestion feature to guild users add appropriate words to the selected topics; and (4) a small scale user study and two detailed studies designed to replicate real-world application scenarios.

2 Background

Query-driven topic model is a semi-supervised topic modelling algorithm developed by Fang et al. (2021). It allows users to specify a simple query in words or phrases and return query-related topics. The original model involves a two-stage modelling process; in the first stage, the model infers one topic for each concept as well as other unrelated topics, while in the second stage, the model extends the topic of each concept into a set of subtopics. In our work, we are only interested in the first stage of the model. If there is no query input, the model works as a conventional topic model.

The model is based on a variant of a Hierarchical

Dirichlet Process (HDP) (Teh et al., 2006), which is a nonparametric Bayesian model that assumes that a restaurant (i.e., a document) has a set of tables and serves dishes (i.e., topics) from a global menu. A single dish is only served at a single table for all customers (i.e., words) who sit at that table. We developed our version of the algorithm based on the Gibbs sampling technique. For a word w_{ji} at document j and position i, the probability for sampling an existing table t is:

$$p(t_{ji} = t \mid t^{-ji}, k) \propto \mathbb{1}(w_{ji}, k_{jt}) n_{jt}^{-ji} f_{k_{jt}}^{-w_{ji}}(w_{ji}) \tag{1}$$

and the probability for sampling a new table t^{new} is:

$$p(t_{ji} = t^{new} \mid t^{-ji}, k) \propto \alpha p(w_{ji} \mid t^{-ji}, t^{new}, \mathbf{k}) \tag{2}$$

Here, t^{-ji} are the table assignments of all other words. k_{jt} is the topic assignment of table t at document j. n_{jt}^{-ji} is the number of words in document j at table t and $f_{k_{jt}}^{-w_{ji}}(w_{ji})$ is the probability of w_{ji} assigned to topic k_{jt}. $\mathbb{1}(w_{ji}, k_{jt})$ is an indicator function, which takes on value 0 if w_{ji} is a pre-defined word associated to a topic z that $z \neq k_{jt}$, and 1 otherwise. α is a hyperparamenter of the model. The probability for sampling a topic $k_{jt^{new}}$ for the new table is:

$$p(k_{jt^{new}} \mid t, k^{-jt^{new}}) \propto \mathbb{1}(w_{ji}, k_{jt}) m_k f_k^{-w_{ji}}(w_{ji}) \tag{3}$$

Human-in-the-Loop topic modelling has received a lot of attention in recent years. Boyer et al. (2017) created a Human–Machine methodology for identifying Systems Thinking topics in a large corpus of text. Users are required to subjectively identify and provide seed documents describing the topic to guide the topic model's training. The methodology, however, did not incorporate any prior knowledge into the modelling process, instead it simply modified the training corpora. Various refinement operations, such as adding words, removing words, adding documents, removing documents, creating topics, merging topics, or removing topics, were implemented to better utilize human knowledge in the topic modelling process (Hoque and Carenini, 2015; Lund et al., 2017; Smith et al., 2018). During the model sampling process, the refinements change the prior knowledge of the underlying model. To understand the usefulness of different refinement operations, Lee et al. (2017) conducted a user-centered approach to find out how non-expert users interpret topic models and what refinement operations they want most, but they only implemented a basic system without full interaction, so the refinement operations they applied did not update the underlying topic model.

Smith et al. (2018) took one step forward from Lee's work by implementing a fully interactive, user-centered HL-TM system and further examined how common interactive machine learning challenges, such as unpredictability, latency and trust, affect the user experience. Kumar et al. (2019) were the first to comparatively evaluate different refinement operations, as well as two feedback injection frameworks, namely informed priors and constraints. They not only suggested that future research should test the system with end users, since their experiments only used simulated user behavior, but also mentioned that it's important to signal to users when their assumptions do not match reality. Though other HL-TM systems exist such as UTOPIAN (Choo et al., 2013) and ConVisIT (Hoque and Carenini, 2015), they use alternative approaches such as non-negative matrix factorization and fragment quotation, and do not provide the complete set of refinement operations that users need (Lee et al., 2017).

Our work can be seen as an extension of the work of Smith et al. (2018). Compared to these studies, it not only provides a fully interactive HL-TM system incorporating various refinement operations but also provides a more user-centered design, such as a complete refinement history and a topic words suggestion feature to enhance the user experience.

3 Proposed System

We introduce our implementation and the interface design in this section.

3.1 Refinement Implementations

Our system uses Gibbs sampling as the inference technique and adopts the constraint method described in Kumar et al. (2019) to inject new information. Every time a user provides a feedback to the system, it first forgets table-word assignment t_{ji} and then injects new information into the system using a potential function $f(k, w, j)$ (Yang et al., 2015) of the hidden topic k of word w in document j. The equation (1) then becomes:

$$p(t_{ji} = t \mid t^{-ji}, k) \propto \mathbb{1}(w_{ji}, k_{jt}) n_{jt}^{-ji} f_{k_{jt}}^{-w_{ji}}(w_{ji}) f(k_{jt}, w_{ji}, j) \tag{4}$$

Prior work (Lee et al., 2017; Smith et al., 2018) discovered that users typically prefer simple refinement operations, therefore, we implemented the following six refinement operations that are commonly used by users in previous studies:

Add word x to topic z. We update the potential function $f(k, w, j)$ such that $f(k, w, j) = 1$ if $k = z$ and $w = x$, otherwise it is assigned a value of 0.

Remove word x from topic z. We update the potential function $f(k, w, j)$ such that $f(k, w, j) = 0$ if $k = z$ and $w = x$, otherwise it is assigned a value of 1.

Swap word order of w_1 and w_2 in topic z so that w_2 has higher order than w_1. We first compute the ratio r between the difference $n_{w_1,z} - n_{w_2,z}$ and n_{w_2}, where $n_{w_1,z}$ and $n_{w_2,z}$ are the counts of w_1 and w_2 in topic z, respectively, and n_{w_2} is the counts of w_2 in all topics except z. We then update the potential function $f(k, w, j)$, such that $f(k, w, j) = 1$ if $k = z$ and $w = w_2$, otherwise it is assigned δ, where $\delta = 0$ if $r > 1$, otherwise $\delta = 1.0 - r$.

Remove document d from topic z. We update the potential function $f(k, w, j)$ such that $f(k, w, j) = 0$ if $k = z$ and $j = d$, otherwise it is assigned a value of 1.

Merge topic t_2 into t_1. In the next Gibbs sampling iteration, we only sample topics for words assigned to t_2 in the previous Gibbs sampling iteration. We update the potential function $f(k, w, j)$ such that $f(k, w, j) = 1$ if $k = t_1$ and $w \in t_2$, otherwise it is assigned a value of 0.

Split topic t into two topics using seed words s, i.e., s need to be moved from t to a new topic t_n. To do this, we first create a new topic using the nonparametric model. Then, we apply the *add word* operation to add all the words in s to t_n.

3.2 Topic words suggestion

We developed a novel topic words suggestion feature in QD-TM to let users decide which words should appear or not appear in the topic words. Using only subjective input may result in topic words that are spurious, but combining the suggested words can help users to steer the model in the right direction. The topic words suggestion feature is integrated into the Gibbs samplings process of the model. It has two stages. First, it samples an indicator of whether a document is relevant to a topic or not. Second, it updates the suggested words from the relevant documents.

Document relevance sampling For each topic k, if a document contains any of the suggested words and is sampled to be relevant to the topic in the previous Gibbs sampling iteration, then it is assumed to be relevant to the topic in the current Gibbs sampling iteration. Otherwise, we use the following equation modified from Wang et al. (2016) to decide if the document is relevant to the topic:

$$p(r_k = c \mid \mathbf{r}, \pi, \beta, \gamma_{\mathbf{r}}, \gamma_{\mathbf{ir}}) \propto \begin{cases} (C_c^{R(-m)} + \gamma_r) \times \frac{\prod_v^V \Gamma(n_{k,v}^{-m} + \beta)}{\Gamma(\sum_v^V (n_{k,v}^{-m} + f_{c,m,v}) + V\beta)} \\ \text{if } c = 1 \\ (C_c^{R(-m)} + \gamma_{ir}) \times \frac{\prod_v^V \Gamma(n_{\tilde{k},v}^{-m} + \beta)}{\Gamma(\sum_v^V (n_{\tilde{k},v}^{-m} + f_{c,m,v}) + V\beta)} \\ \text{if } c = 0 \end{cases} \tag{5}$$

where $C_c^{R(-m)}$ is the number of documents under relevance status c excluding the current document m, $n_{k,v}^{(-m)}$ is the counts of term v in target topic k excluding the words from the current document m, $f_{c,m,v}$ is the frequency of term v in document m under relevance status c, and π indicates the Bernoulli distribution over relevance status. β, γ_r, and γ_{ir} are hyperparameters, and V is the vocabulary size of the dataset.

Automatic keywords expansion To get suggested words for a topic in each Gibbs sampling iteration, we first calculate the score of term v in the relevant documents of the topic:

$$Score(v) = P_R(v) log \frac{P_R(v)}{P_C(v)}, \tag{6}$$

where $P_R(v)$ is the probability of term v in the relevant documents and $P_C(v)$ is the probability of term v in the whole corpus. We extract terms with high scores as the candidate terms. We then calculate the cosine similarity between each candidate term and the embedding of the topic, and only add terms to the suggested words list when the similarity is greater than 0.5. The embedding of a topic k can be obtained by:

$$emb(k) = \sum_m^M P^i(k, m) emb(m), \tag{7}$$

where $p^i(k, m)$ is the probability of m^{th} word of topic k in i^{th} iteration, M is the total number of representative words in the topic, which is usually set to 10, and $emb(m)$ is the pretrained word embedding of the m^{th} word.

3.3 Interface Design

The user interface of the system consists of two windows (Appendix A). In the first window (Figure A1), users can define the hyper-parameters of a topic model, such as the initial number of topics. If users are interested in topics describing specific concepts or aspects of the corpus, they can also define the prior knowledge of a topic model in this window. If no prior knowledge is provided, then the model behaves as a conventional topic model without any prior knowledge.

The second window (Figure A2) displays the detailed information of the trained model. Users can view the model as a list of topics. Users can also refine the model using the refinement operations implemented and track model change histories. Different from previous systems where users can only refine a model sequentially and cannot retract the changes they made, we include a model history panel with a novel model tree structure so users can make different attempts at refinement to the same model node and compare the resulting models. To help users interpret each topic, we also show a set of topic labels from the automatic topic labeling algorithm (Mei et al., 2007), where a topic label is a phrase that summarises the main idea of the topic. A detailed description of the interface can be found in Appendix A.

4 Evaluation

The evaluation is divided into two parts. In the first part, we evaluated the performance of the topic words suggestion feature in a controlled, laboratory experiment, where we compared two versions of our system. In the second part, we evaluated the HT-LM system by applying it in two realistic topic modelling use cases.

4.1 Topic words suggestion evaluation

Datasets Three commonly used datasets for topic modelling were chosen: the 20newsgroup [1] dataset containing 18k news posts from 20 categories; the TagMyNews [2] dataset containing 32k short English news from 7 categories; and the SearchSnippets (Xu et al., 2017) dataset containing 12k short English news from 8 categories.

Baselines Our HL-TM system uses QD-TM as the underlying model. To test whether the *topic words suggestion* feature can improve the original model, we used only this feature to refine the QD-TM and compared the refined model with the original QD-TM.

Parameterisation We focused on the targeted topic modelling capabilities of QD-TM. To make fair comparisons, we adopted the same experimental settings described in Fang et al. (2021). We used query phrases to represent the main concept of each category in a dataset, following the same setup as in Fang et al. (2021). Categories that do not have meaningful names were removed from the datasets, e.g., `talk.politics.misc` in the 20Newsgroup dataset. The query phrases were integrated into the model using the first window of our HL-TM interface.

We used our HL-TM system to refine the original QD-TM and only used the *add word* refinement operation. In each Gibbs sampling iteration, for each topic, we added all suggested words to the topic. For the hyper-parameters in equation (5), β is set to 0.5, and both γ_r and γ_{ir} are set to 1. We ran 2000 Gibbs sampling iterations.

Results We evaluated the quality of the final models in terms of topic coherence and document retrieval performance as in Fang et al. (2021). Better precision@K scores indicate that the learnt topics are more discriminative and representative. Higher coherence scores indicate better topic interpretability. Table 1 shows the performance of adding suggested topic words to the model. We observed that the refined model achieves higher coherence scores compared with the original model. It indicates that the suggested words are semantically related to each topic and can help produce more coherent topics. The better precision@K scores also show that the suggested topic words are highly relevant to the predefined categories of the dataset. This is expected as the word suggestion feature in our system tends to identify more important words related to the category of the corresponding topic. Instead of blindly adding the suggested words to the model, our system allows users to decide which suggested words should be added. This allows them to use their domain knowledge to further filter out noise.

4.2 Laboratory evaluation

Description To compare our system with the state-of-the-art human-in-the-loop topic modelling system (Smith et al., 2018), we recruited 20 participants via a university campus email list to run

---

[1] http://qwone.com/ jason/20Newsgroups/

[2] http://acube.di.unipi.it/tmn-dataset/

Model	20news		TagMyNews		SearchSnippets	
	Coherence	Precision@K	Coherence	Precision@K	Coherence	Precision@K
QD-TM	0.445	0.612	0.413	0.736	0.475	0.741
QD-TM + words suggestion	**0.482**	**0.634**	**0.477**	**0.763**	**0.481**	**0.825**

Table 1: Average results for adding suggested topic words for model refinement. We ran each model five times. We used the topic coherence metric C_V from Röder et al. (2015)

a small-scale user study, with one subject group working with an equivalent of Smith et al. (2018) – our system with no model history, no words suggestion, and no topic labeling (old system) – and one group working with our full-featured system (new system). In a factorial design with two independent variables, we used two corpora of Reddit posts focusing on online teaching platforms (corpus A) and Twitter tweets discussing the 2021 United Nations Climate Change Conference (corpus B). Participants were randomly allocated into two groups. The first group did corpus A with the old system and corpus B with the new system, and the second group did corpus B with the old system and corpus A with the new system. The task was to conduct a qualitative analysis of the datasets.

Participants randomly started with either the old system or the new system to eliminate the influence of training effects. We then evaluated the average topic quality of the two system conditions. In addition to comparing topic quality across systems, we also asked participants questions about how much they like the new features. All participants were fluent English speakers and non-experts in topic modelling. Each participant received a £20 Amazon gift card as payment for the experiment.

Dataset and Topic Model We used two datasets for the experiments. The Reddit dataset contains 9,651 posts focusing on online teaching platforms. These posts were randomly sampled from the original dataset taken from Alqazlan et al. (2021). The Twitter dataset contains 8,990 tweets related to the 2021 United Nations Climate Change Conference. We used keywords related to the conference to search for tweets between 31 October 2021 and 17 November 2021. These keywords (see Table B1) were checked and provided by social scientists researching COP26 climate change tweets through a few rounds of manual inspection of the tweets. We used QD-TM as the underlying model, and used a standard stop words list and 2000 Gibbs sampling iterations to initialize the topics. We set the initial topic number to 10 and 13 respectively for the Reddit and Twitter datasets, respectively, based on our previous experience with these datasets. For each subsequent update during the task, 10 Gibbs sampling iterations were run.

Procedure Sessions took around 60 minutes on average, and they were conducted face-to-face. We began by introducing participants to topic modelling and how to use the tools to refine topics. For both systems, we asked them to first read the top five posts of each topic to interpret the underlying theme and then read the corresponding top 10 topic words and use the provided refinement operations to refine the topic until the top 10 topic words and the top 5 posts are consistent with the interpreted theme. For the new system, participants could also access the new features – model history, word suggestions, and topic labelling to assist their in refinements. We asked participants to click the *apply refinements* button to update the underlying model after they have added all the necessary refinements to a topic. They were allowed to undo any operations that have not been applied to the underlying model. After they finished the refining tasks, we asked them to rate how much they are satisfied with the resulting topics and how much they like the new features. Due to time limitations, we only asked participants to refine the first five topics of each system. The first five topics of the starting model for each corpus can be found in Table B2 and Table B3.

Findings We recorded user interactions with the tools. Users performed a total of 1,284 refinements using the two systems. Among the top four most used operations were the *delete words* operation (used 554 times), the *swap words* operation (used 512 times), the *add words* operation (used 171 times), and the *delete document* operation (used 81 times). This is consistent with the findings of Smith et al. (2018), who also observed that these operations were the four most used operations (excluding the add to stop words operation since our system didn't implement this). For the new system, the *add words* operation was applied 115 times in total, among which 56 times were adding suggested words, which indicates that the suggested words operation had high usage. We report the usage of refinements for each subject group in Table B4.

To evaluate whether using the new system can result in better topics than the old system, participants were asked to rate their satisfaction with the final topics of the two systems on a five point scale (1 not very satisfied, 5 very satisfied). For the Reddit dataset, the average satisfaction score for the new system was 4.2 (SD=0.67), while the score for the old system was 3.89 (SD=0.78). Although a Mann-Whitney U test (U=30.5, z=0.83887, p=.200) showed this difference was not statistically significant, 8 out of 10 participants were satisfied with final topics of the new system. For the Twitter dataset, the score for the new system was 4.0 (SD=0.71), while the score for the old system was 3.2 (SD=0.67). A Mann-Whitney U test (U=18.5, z=1.8985, p=.029) showed that the difference was statistically significant, suggesting that the new system can help improve topic quality.

To evaluate whether the model tree feature can help users track changes, we asked participants to rate their agreement with the statement, "I was able to remember what the model looked like before my updates", on a five point scale (1 strongly disagree satisfied, 5 strongly agree), for both the old and new systems. The average agreement was 3.1 (SD=1.07) for the old system and 3.9 (SD=0.85) for the new system. A Mann-Whitney U test (U=116, z=2.25868, p=.012) showed the difference to be statistically significant. Participants were also asked to rate their agreement with the statement, "the model tree feature of the new system can help me track my updates". The average agreements for the statement was 4.35 (SD=1.04). This strongly suggests that the model tree feature of the new system can help users track their changes. To evaluate the usefulness of the words suggestion feature, we asked participants to rate their agreement with the statement, "the suggested words feature of the new system can help me identify relevant words of the topics". The average agreements for the statement was 4.1 (SD=0.64). 17 out of 20 participants stated that the feature is very useful, suggesting that the feature is a good supplement for adding words to topics. The finding is consistent with the observation of user interactions with the new system that around 50% (56 out of 115) of added words were from the suggested words list. For the topic labelling feature, participants were asked if they agreed with the statement, "the topic labels from the full-featured system can help me interpret the meaning of the topics". The average score for the statement was 3.25, and 11 out of 20 mentioned that the provided labels for some topics are totally irrelevant. This suggests that the algorithm (Mei et al., 2007) doesn't fit the datasets well and suggests the needs for a more accurate topic labelling algorithm in the future.

Suggestions Though the user studies show promising results, participants also had suggestions for improving the new system. Six mentioned that the swap words operation didn't fit their needs well. Instead of swapping the order of the two selected words, they prefer to allow the selected words to be inserted in new positions, which provides more flexibility when ranking topic words. One participant suggested allowing the undo any of the previously added operations directly, rather than starting with the most recently operation.

4.3 Use case one: Tutors' experiences in the Gig Economy

Description To evaluate our system on real-world document analysis tasks, we invited a researcher who is investigating the Gig Economy to use our model. In previous work, Alqazlan et al. (2021) applied LDA to identify topics that are related to tutors' experiences in the Gig Economy from a Reddit dataset where tutors posted about their experiences of working on online teaching platforms. We asked the researcher to use our system to identify the relevant topics in the dataset and assess whether using our system can produce better results than LDA. As Alqazlan et al. (2021) has previously found that a 17-topic LDA extracted the most number of relevant topics, for a fair comparison, the initial number of topics for our system was also set to 17. The researcher was encouraged to use any refinement operations provided by the system until she felt that the top 5 relevant posts of each detected topic revolved around one main theme.

Results A total of 16 models were trained and two branches were created during the refinement process, with models 11, 12 and 13 constituting one branch and models 14, 15 and 16 constituting the other branch. Both of these branches were extended from model 10, which was trained sequentially from the initial model. By using the model tree panel to compare these models, it was found that Model 16 produced the most satisfactory results. Of all the refinement operations, *remove document* was used the most and *swap word order* was used the least. The researcher commented that the *swap word order* could be more useful if it only adjusts the position of a word in the topic, rather than swapping a word with another one.

We compare LDA with Model 16 and present the qualitative results here. For both models, the researcher was asked to manually assign a label to each topic. For the LDA model, 11 topics were found to be relevant to tutors' experience in the Gig Economy, two of which (Table C1) were missed by Model 16, while for Model 16, 16 topics were found to be relevant, of which seven topics (Table C2) were not identified by the LDA. This indicates that our HL-TM system is able to assist the researcher to identify more relevant topics.

To determine whether using our system can help produce better quality topics, we measured the topic coherence score for the 9 relevant topics identified by both LDA and Model 16 (Table C3). We used the best performing topic coherence measure CV based on the external corpus (Wikipedia) (Röder et al., 2015). We found that our system produced better topic coherence scores for six topics, while the LDA produced better scores for only three topics. This finding is consistent with the researcher's view that only two topics from the LDA were of better quality than Model 16. We present the relevant topics from both LDA and Model 16 in Appendix C.

4.4 Use case two: Patenting strategies for pre-determined patent value categories

Description We invited a second researcher, who is working on patenting strategies, to evaluate our system. In a previous study, Ribeiro and Shapira (2020) has identified a set of patent value categorises (Table D1) for synthetic biology patents. However, their work was based entirely on human evaluation and was, therefore, time-consuming, and biased evaluations could occur, so only 102 patent documents were analysed. In this use case, the researcher aimed to use our system to categorise a larger patent sample to the pre-defined categories and reduce human selection bias. 2607 related patents in the United States Patent and Trademark Office (USPTO) dataset were used. The predefined categorises are presented in Appendix D.

To identify documents in their predefined categories in the dataset, the researcher used the first window of the interface to incorporate prior knowledge. A list of final concept words of each category is presented in Table D2 in Appendix D. The initial number of topics is set to 20.

Results By using the first window of the user interface to incorporate priori knowledge, the initial model was able to roughly infer topics for the predefined categories, although the topics were not yet of high quality. We then asked the researcher to examine the top 10 relevant documents of these topics to assess the categorisation quality. The average precision is 0.3. The researcher then refined these topics using the system until she felt that it best categorised the top 5 relevant documents into each category and was satisfied with the resulting topic words. In total 23 models were trained and five branches were created, with Model 23 being the best one. After the refinements, we asked the researcher to further examine the top 10 relevant documents of the refined topics to compare with the initial model. The average precision for the refined topics is 0.96, which is much higher than the result from the initial model. This verifies that the use of the system can help identify a larger sample of documents in pre-defined categories than the previous manual evaluation method. In total, the *add word* operation was applied 62 times, among which 41 times were related to adding suggested words from the system. This shows that our topic words suggestion feature can indeed identify words that are highly relevant to specific topics.

We also compared the topic coherence scores for the focused topics from the initial model (Table D2) and from Model 23 (Table D3). The scores are 0.404 and 0.434, respectively. It shows that using the system can help produce better quality topics.

5 Conclusions and Future work

We have developed a novel user-centered, interactive, HL-TM system to address the limitations of the prior work (Smith et al., 2018). An advanced user interface with a model history panel is presented to allow users compare different models and retract inappropriate changes. The use of the QD-TM as the underlying model supports both the full-analysis and targeted-analysis capabilities. A novel topic words suggestion feature is also integrated into the system and the evaluation shows that the feature is promising for suggesting reasonable and coherent words for topics. A small scale experimental user study and two detailed use cases further verified the usefulness of the system on real-world tasks. From the use cases, we observed that both researchers trained many different models with multiple branches created (use case one has 16 models trained and two branches created, while use case two has 23 models trained and five branches created). This shows that researchers are very likely to change their mind when refining a topic model, so allowing researchers to compare different models and retract their changes is helpful. In the future, we plan to improve the system based on the results from the user evaluations and conduct more extensive evaluations to gather feedback on deploying the system in a wide variety of applications.

Limitations

We used two small datasets in the user evaluations, so participants were not affected by latency issues, where the user would have to wait while the algorithm performs updates. Since QD-TM is more complex than LDA, and the words suggestion feature further increased the complexity of the computation, a large dataset could lead to latency problems. The time complexity of our system is O(DLK + KV), where D is the number of documents used, L is the average document length, K is the number of topics and V is the vocabulary size. We tested the system with two different dataset sizes (20,000 tweets vs. 8,990 tweets). By using 20,000 tweets, the average wait time for model updates during user interaction (10 Gibbs sampling iterations) was 34 seconds, compared to 14 seconds using 8,990 tweets.

According to Smith et al. (2018), longer wait times can negatively affect users when using interactive systems, suggesting that latency could be an issue in our system as the size of the dataset increases. However, as is well known, these effects can be mitigated through the provision of time affordances (Conn, 1995). We will address this in the future.

In addition to the latency issue, the laboratory evaluation we conducted also has limitations. Not all the participants were familiar with qualitative analysis. It is likely that participants with extensive experience in qualitative analysis identify more valuable refinements to the models, resulting in better quality topics.

Acknowledgements

This work was supported in part by the UK Engineering and Physical Sciences Research Council (grant no. EP/V048597/1). YH is supported by a Turing AI Fellowship funded by the UK Research and Innovation (EP/V020579/1). ZF receives the PhD studentship jointly funded by the University of Warwick and China Scholarship Council.

References

Lama Alqazlan, Rob Procter, and Michael Castelle. 2021. *Workshop on Natural Language Processing for Digital Humanities, 18th International Conference on Natural Language Processing*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Jordan Boyd-Graber, David Mimno, and David Newman. 2014. Care and feeding of topic models: Problems, diagnostics, and improvements. *Handbook of mixed membership models and their applications*, 225255.

Ryan C Boyer, William T Scherer, Cody H Fleming, Casey D Connors, and N Peter Whitehead. 2017. A human–machine methodology for investigating systems thinking in a complex corpus. *IEEE Systems Journal*, 12(3):2937–2948.

Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296.

Jaegul Choo, Changhyun Lee, Chandan K Reddy, and Haesun Park. 2013. Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE transactions on visualization and computer graphics*, 19(12):1992–2001.

Alex Paul Conn. 1995. Time affordances: the time factor in diagnostic usability heuristics. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 186–193.

Stephan A Curiskis, Barry Drake, Thomas R Osborn, and Paul J Kennedy. 2020. An evaluation of document clustering and topic modelling in two online social networks: Twitter and reddit. *Information Processing & Management*, 57(2):102034.

Ramakrishna Dantu, Indika Dissanayake, and Sridhar Nerur. 2021. Exploratory analysis of internet of things (iot) in healthcare: a topic modelling & cocitation approaches. *Information Systems Management*, 38(1):62–78.

Zheng Fang, Yulan He, and Rob Procter. 2021. A query-driven topic model. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1764–1777, Online. Association for Computational Linguistics.

Barney G Glaser, Anselm L Strauss, and Elizabeth Strutzel. 1968. The discovery of grounded theory; strategies for qualitative research. *Nursing research*, 17(4):364.

Nihit Goyal and Michael Howlett. 2021. “measuring the mix” of policy responses to covid-19: Comparative policy analysis using topic modelling. *Journal of Comparative Policy Analysis: Research and Practice*, 23(2):250–261.

Tobias Heidenreich, Fabienne Lind, Jakob-Moritz Eberl, and Hajo G Boomgaarden. 2019. Media framing dynamics of the ‘european refugee crisis’: A comparative topic modelling approach. *Journal of Refugee Studies*, 32(Special_Issue_1):i172–i182.

Gregor Heinrich. 2009. A generic approach to topic models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 517–532. Springer.

Enamul Hoque and Giuseppe Carenini. 2015. Convisit: Interactive topic modeling for exploring asynchronous online conversations. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 169–180.

Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. 2012. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213.

Varun Kumar, Alison Smith-Renner, Leah Findlater, Kevin Seppi, and Jordan Boyd-Graber. 2019. Why didn’t you listen to me? comparing user control of human-in-the-loop topic models. *arXiv preprint arXiv:1905.09864*.

Tak Yeon Lee, Alison Smith, Kevin Seppi, Niklas Elmqvist, Jordan Boyd-Graber, and Leah Findlater. 2017. The human touch: How non-expert users perceive, interpret, and fix topic models. *International Journal of Human-Computer Studies*, 105:28–42.

Jeffrey Lund, Connor Cook, Kevin Seppi, and Jordan Boyd-Graber. 2017. Tandem anchoring: A multiword anchor approach for interactive topic modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 896–905.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499.

David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 262–272.

Barbara Ribeiro and Philip Shapira. 2020. Private and public values of innovation: A patent analysis of synthetic biology. *Research policy*, 49(1):103875.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408.

Carson Sievert and Kenneth Shirley. 2014. Ldavis: A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, pages 63–70.

Alison Smith, Varun Kumar, Jordan Boyd-Graber, Kevin Seppi, and Leah Findlater. 2018. Closing the loop: User-centered design and evaluation of a human-in-the-loop topic modeling system. In *23rd International Conference on Intelligent User Interfaces*, pages 293–304.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. *Journal of the american statistical association*, 101(476):1566–1581.

Di Wang, Marcus Thint, and Ahmad Al-Rubaie. 2012. Semi-supervised latent dirichlet allocation and its application for document classification. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 306–310. IEEE.

Shuai Wang, Zhiyuan Chen, Geli Fei, Bing Liu, and Sherry Emery. 2016. Targeted topic modeling for focused analysis. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1235–1244.

Jiaming Xu, Bo Xu, Peng Wang, Suncong Zheng, Guanhua Tian, and Jun Zhao. 2017. Self-taught convolutional neural networks for short text clustering. *Neural Networks*, 88:22–31.

Yi Yang, Doug Downey, and Jordan Boyd-Graber. 2015. Efficient methods for incorporating knowledge into topic models. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 308–317.

Appendix A: Interface Design

The user interface of the system consists of two windows. If users are interested in topics describing specific concepts or aspects of the corpus, they can define the prior knowledge of a topic model in the first window (Figure A1). They can use a query phrase to define the concept of interest. The input query phrase is expanded to a set of candidate concept words using the concept words extractor from Fang et al. (2021). Users can use their knowledge to determine which words should be included in a topic and click the "+" buttons next to them to add to the final concept words list of a topic. They can also add words that are not in the candidate words list to the final list based on their knowledge. The words in the final list can be removed by clicking the corresponding “-” buttons. It also supports viewing the top relevant documents of the input query by changing the viewing options from "By keywords" to "By documents".

If users have no preferred interest in the topics of the corpus, they can leave the final concept words list of each topic empty, and the model behaves as a conventional topic model without any prior knowledge. Users can click the "apply" button to apply the settings to the underlying model. Users can view the resulting topics by clicking the "View" button in the bottom, which takes them to the second window of the user interface.

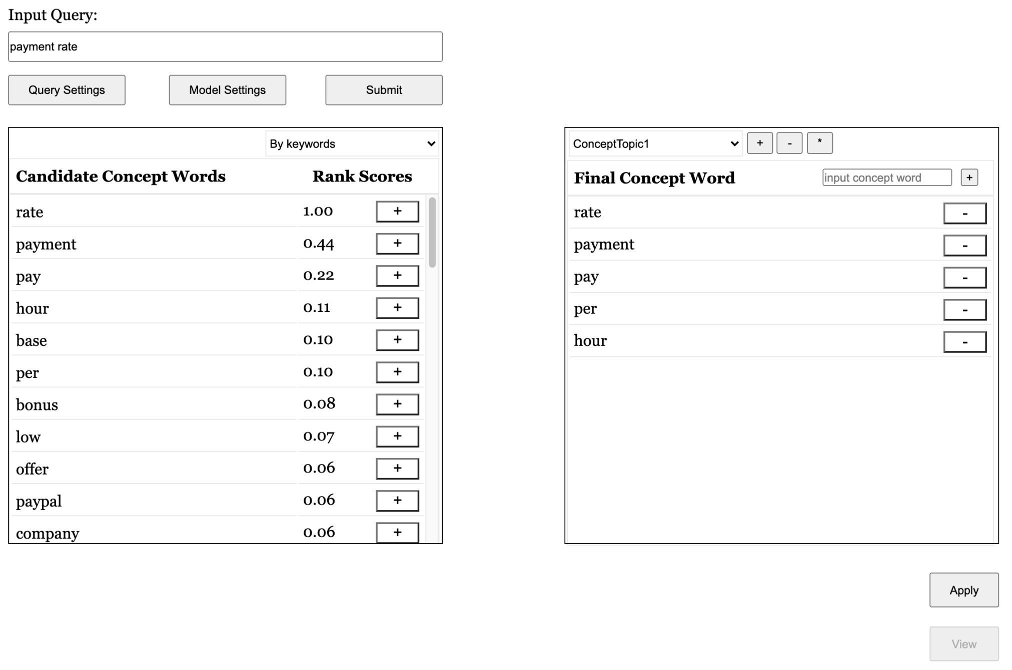
The second window of the interface (Figure A2) has three panels: *model history* panel (left panel), *model detail* panel (middle panel), and *inter-topic distance map* (right panel). The *model history* panel consists of two subpanels: the *model tree* panel (top panel), which displays the refined models in a tree structure, and the *refinements history* panel (bottom panel). Every time a user refines a selected model, a new model is added to the tree. As shown in Figure A2, users can also create new branches from the same model node. For example, model 5, model 6 and model 7 are all refined from model 4. The *refinements history* panel allows users to view the refinements history between two connected models by clicking on the edge between them. Users can also view the pending refinements of the selected model as shown in Figure A2.

The pending refinements will not be applied to the underlying model until users click the "Apply Refinement" button. We also include an "undo" button to allow users to undo previously added refinements. The *model history* panel allows users to compare previously refined models and keep track of their previous changes to further assist the refinement process. By clicking the "Download models" button, users can download the entire model tree to their local machine, and by clicking the "Load models" button, users can load a previously downloaded model tree to continue the refinements.

The *model detail* panel, shown in the middle of Figure A2, provides an overview of the selected model’s topics, as well as the top words in each one. Weight represents the prevalence of the topic in the whole corpus. Users can rename a topic by typing a new name in the “Topic” column. Users can also merge or split topics in this panel. The right side of Figure A2 is the *inter-topic distance map*. Each circle represents a topic, with its size representing the topic weight in the corpus. The *inter-topic distance map* is an intuitive way to reveal the quality of a topic model where a larger distance between topics indicates a better model (Sievert and Shirley, 2014). By clicking the “view” button on the window, the selected topic’s specifics will then be presented as shown in Figure A3.

The top left side of Figure A3 shows a set of topic labels from the automatic topic labeling algorithm (Mei et al., 2007), where a topic label is a phrase that summarises the main idea of the topic. The aim of the topic label is to help users interpret the topic. We also present the top words of the topic with corresponding topic-word weights. Weights indicate the prevalence of the words in the topic. Users can apply *add, delete* or *swap word order* refinement operations here. The right side of Figure A3 displays the top documents associated with the topic and ranks them based on their weight, representing the document’s prevalence of the topic. Users can apply the *remove document* refinement operation here. By clicking the "view" button, users can view the details of the selected document.

When users click the "add" button in Figure A3, an "add words" sub-panel (Figure A4) appears. A list of words suggested by the topic words suggestion feature is displayed in the panel. Users can select from the suggested words or from their own knowledge to add words to the selected topic.



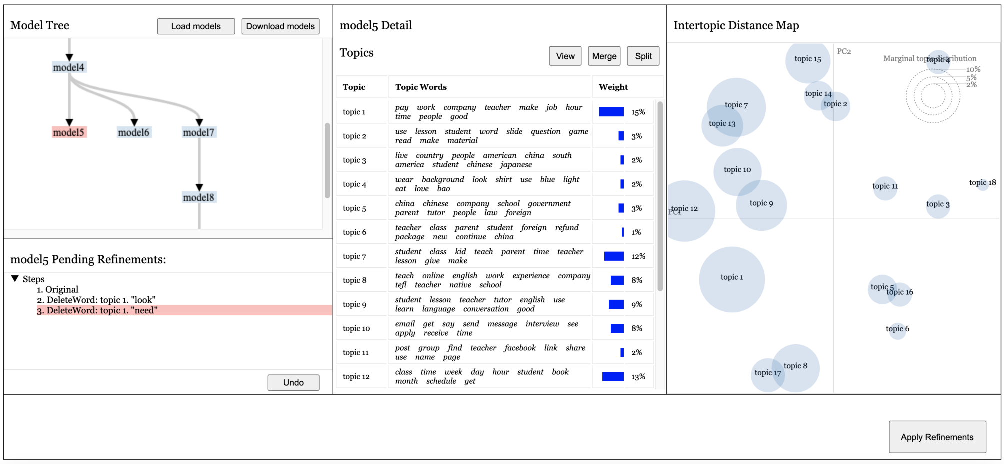
Input Query: payment rate

Candidate Concept Words	Rank Scores
rate	1.00
payment	0.44
pay	0.22
hour	0.11
base	0.10
per	0.10
bonus	0.08
low	0.07
offer	0.06
paypal	0.06
company	0.06

Final Concept Word

rate
payment
pay
per
hour

Figure A1: The first window of the user interface. Users can define the prior knowledge of a topic model here.

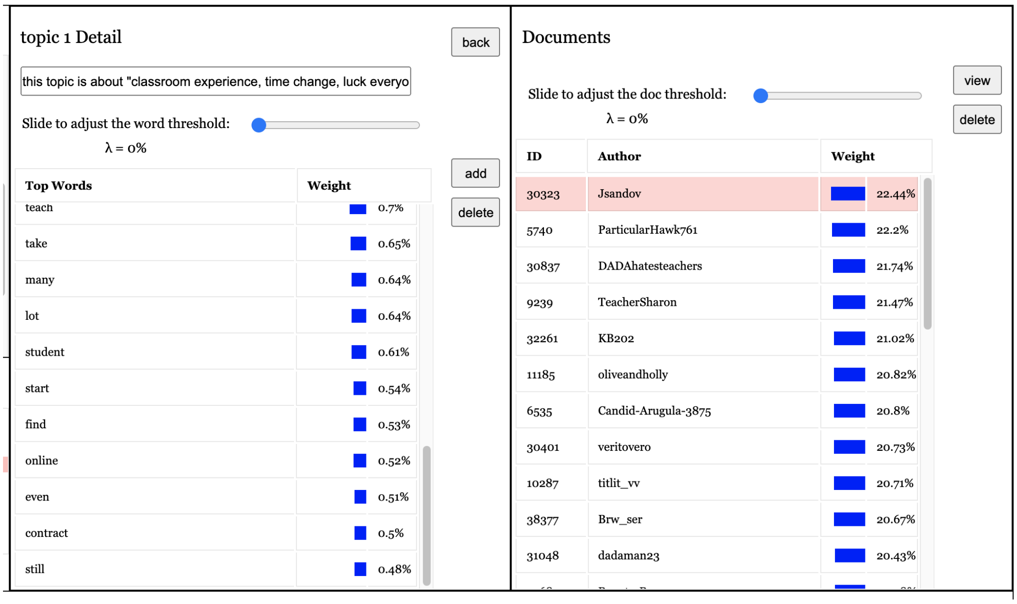


model5 Pending Refinements:

▼ Steps
1. Original
2. DeleteWord: topic 1. "look"
3. DeleteWord: topic 1. "need"

Topic	Topic Words	Weight
topic 1	*pay work company teacher make job hour time people good*	15%
topic 2	*use lesson student word slide question game read make material*	3%
topic 3	*live country people american china south america student chinese japanese*	2%
topic 4	*wear background look shirt use blue light eat love bao*	2%
topic 5	*china chinese company school government parent tutor people law foreign*	3%
topic 6	*teacher class parent student foreign refund package new continue china*	1%
topic 7	*student class kid teach parent time teacher lesson give make*	12%
topic 8	*teach online english work experience company tefl teacher native school*	8%
topic 9	*student lesson teacher tutor english use learn language conversation good*	9%
topic 10	*email get say send message interview see apply receive time*	8%
topic 11	*post group find teacher facebook link share use name page*	2%
topic 12	*class time week day hour student book month schedule get*	13%

Figure A2: The second window of the user interface, where users can view the details of the selected model and apply merge or split topics refinement operations. Users can also compare different models that have been previously refined and keep track of their previous changes.



topic 1 Detail

this topic is about "classroom experience, time change, luck everyo

Slide to adjust the word threshold:
λ = 0%

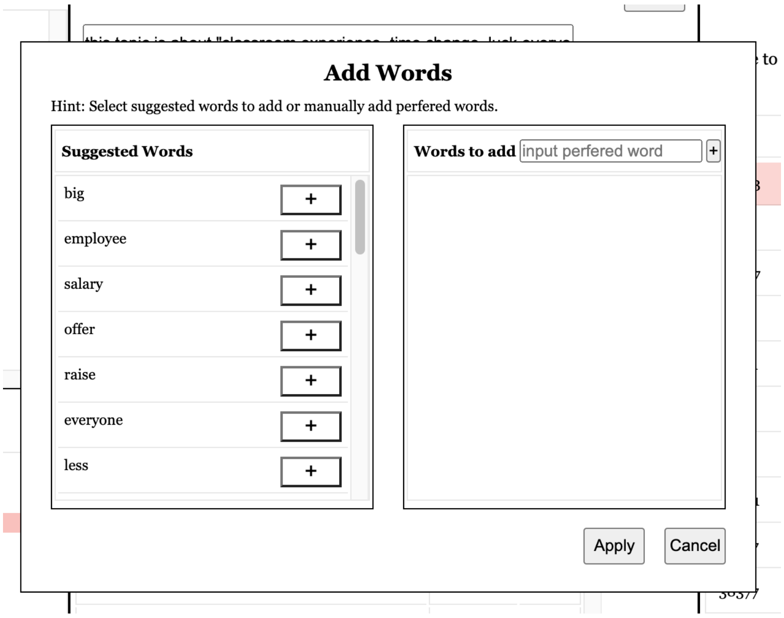
Top Words	Weight
teach	0.7%
take	0.65%
many	0.64%
lot	0.64%
student	0.61%
start	0.54%
find	0.53%
online	0.52%
even	0.51%
contract	0.5%
still	0.48%

Documents

Slide to adjust the doc threshold:
λ = 0%

ID	Author	Weight
30323	Jsandov	22.44%
5740	ParticularHawk761	22.2%
30837	DADAhateteachers	21.74%
9239	TeacherSharon	21.47%
32261	KB202	21.02%
11185	oliveandholly	20.82%
6535	Candid-Arugula-3875	20.8%
30401	veritovero	20.73%
10287	titlit_vv	20.71%
38377	Brw_ser	20.67%
31048	dadaman23	20.43%

Figure A3: The specifics of a selected topic. Users can apply add, remove or swap word order refinement operations here.



Add Words

Hint: Select suggested words to add or manually add perfered words.

Suggested Words
big
employee
salary
offer
raise
everyone
less

Words to add: input perfered word

Figure A4: The "add words" sub-panel.

Appendix B: Tables for the small-scale user study (section 4.2)

keywords
police liaison, Police Liaison Officer, PLO, blue bib, #peacefulprotest, #righttoprotest, Police, Policing, Anarchy, Anarchism, Violence, #ClimateJustice, #ClimateCrisis, #ClimateAction, cop26, #GlobalDayofAction, #GlobalDayforClimateJustice, #greenwash, Pigs, Fascist, Stormtrooper, Nazi, crowd, march, rally, mob, extremist, class, privilege, eco-zealot, vandal, nutter, lunatic, eco-fascist, hypocrite, far-left, chaos, stunt, marxist, terror, rabble, anarchist, filth, rozzer, clown, pc plod, old bill, polis, wankers, woke, eco-warrior, campaign, radical, extremist, zealot, authoritarian, tyranny, tyrant, numpty, numpties, scum, disgusting, awful, unbelievable, evil, frightening, selfish, vicious, violent, thug, brutal, vicious, fight, carnage, blood, injury, hostility, aggression, force, assault, invasion, offensive, friction, stress, strain, damage, hurt, harm, block, kettle, contain, arrest, imprison, charge, thankyou, thanks, carnival, festival, fun, enjoy, party, tension, tense, disrupt, soft

Table B1: Keywords used to retrieve tweets related to the 2021 United Nations Climate Change Conference.

Topic	**Top 10 words**
topic 1	hour class week pay month work time day start make
topic 2	email apply work company good check send interview hire process
topic 3	teach english online native tefl degree experience company teacher certificate
topic 4	student class give rating lesson feedback time parent level bad
topic 5	tutor account student lesson video group profile share bank paypal

Table B2: The first five topics of the starting model for the Reddit dataset. Both the new and old systems used the same starting model.

Topic	**Top 10 words**
topic 1	climate action change cop people glasgow world today day amp
topic 2	party vote tory labour johnson boris government mps paterson sleaze
topic 3	amp emissions countries climate carbon energy gas global coal fuel
topic 4	charge pay people covid money case work claim give email
topic 5	game great play team today enjoy amp fun win time

Table B3: The first five topics of the starting model for the Twitter dataset. Both the new and old systems used the same starting model.

Operation	**New system/Reddit**	**New system/Twitter**	**Old system/Reddit**	**Old system/Twitter**	**Total**
Delete words	74	205	198	77	554
Reorder words	118	152	139	103	512
Add own words	14	45	32	24	115
Add suggested words	7	49	/	/	56
Delete document	12	37	20	12	81
Split topic	0	3	0	4	7
Merge topics	0	1	0	1	2

Table B4: The usage of refinements for each subject group. “New system/Reddit” indicates the use of new system with the Reddit dataset.

Appendix C: Tables for the use case one (section 4.3)

Label	**Top 10 words**
Reasons to join or leave a platform	company, teacher, job, work, pay, people, make, money, online, think
Miscommunication with platform management	know, anyone, video, tutor, help, ask, let, please, apply, interview

Table C1: Relevant topics identified by LDA, but missed by Model 16.

Label	**Top 10 words**
Minimum and living wages	pay, work, company, teacher, make, hour, rate, wage, job, time
Nationalities of students	china, people, country, live, chinese, student, american, america, world, government
A restriction on what tutors can wear while tutoring	wear, background, shirt, blue, eat, light, use, bao, love, dino
Chinese new law on out-of-school tutoring	teacher, china, parent, chinese, company, foreign, school, class, new, student
Differences between a professional tutor and a community tutor	student, lesson, teacher, tutor, use, english, learn, language, make, conversation
Discussions about Facebook groups created by platforms' management	post, group, find, facebook, link, see, teacher, use, share, name
Tutors express anger and dissatisfaction with the platform	people, make, good, fuck, post, shit, go, feel, na, sorry

Table C2: Relevant topics identified by Model 16, but missed by LDA.

Label	**Top 10 words (LDA)**	**Top 10 words (Model 16)**
Bookings and working hours	week, hour, day, time, book, class, slot, schedule, open, month	class, week, time, day, hour, book, student, schedule, month, slot
Rating system	student, teacher, work, class, give, rating, really, think, month, company	student, rating, call, minute, reservation, time, hour, ph, tutor, week
Teaching materials and methods	student, use, lesson, question, word, ask, say, make, learn, conversation	word, use, game, read, student, slide, play, question, lesson, sentence
Technical issues	class, minute, student, call, time, show, start, late, happen, reservation	use, issue, work, app, internet, problem, try, phone, computer, test
Payment	lesson, pay, time, student, hour, teacher, rate, tutor, base, minute	class, hour, pay, per, bonus, teach, minute, base, lesson, student
Experiences with kids in class	kid, teach, level, well, old, year, think, really, feel, say	student, class, kid, teach, parent, time, lesson, give, teacher, make
Hiring process	email, send, message, say, get, group, reply, back, try, see	interview, apply, video, demo, hire, good, application, new, company, process
Job requirements	english, native, live, country, speaker, language, china, work, american, non	teach, online, english, native, tefl, company, work, experience, teacher, school
Bank transfers and transaction fees	rating, account, pay, demo, use, bank, test, paypal, payment, internet	account, bank, pay, paypal, use, payment, transfer, payoneer, fee, money

Table C3: Relevant topics identified by both LDA and Model 16

Appendix D: Tables for the use case two (section 4.4)

Category	Definition
Market and industrial opportunities	The potential of the invention to enter existing markets
Cost and efficiency	Reduction of production costs associated with more efficient processes
Increasing compound yields	Improvements in compound productivity based on novel processes
Upscaling production	Taking production to the commercial level
Scientific advancements	Contribution to knowledge production
Environmental sustainability	Contribution to environmental quality and preservation
Human health	Improvements in the quality of human health
Food security	Avoiding competition with human food sources
Animal health	Improvements in the quality of animal health

Table D1: categories table from (Ribeiro and Shapira, 2020).

Category	Final Concept Words
Market and industrial opportunities	market, commercial, value, exists
Cost and efficiency	Reduction of production costs associated with more efficient processes
Increasing compound yields	compound, yield, increasing, plant
Upscaling production	production, improve, scale, large
Scientific advancements	advancement, benefit, filed
Environmental sustainability	renewable, sustainable, energy
Human health	health, human, patient, cancer, disease
Food security	food, security, supply, preparation, chain
Animal health	animal, health, disease

Table D2: Final concept words for each pre-defined category.

Category	Topic (Top 10 words)
Market and industrial opportunities	healthy, improved, serum, level, variant, commercial, value, concentration, woman, sample
Cost and efficiency	efficiency, increase, cost, increasing, efficient, enhance, reaction, amplification, low, target
Increasing compound yields	plant, tolerance, improved, compound, yield, marker, increasing, herbicide, soybean, resistance
Upscaling production	improve, improved, production, activity, antibody, enzyme, property, polypeptide, stability, expression
Scientific advancements	benefit, fold, effective, greater, composition, field, size, skin, provide, material
Environmental sustainability	renewable, energy, product, source, diesel, produced, carbon, lipid, sustainable, fuel
Human health	disease, patient, effective, cancer, amount, antibody, human, treatment, administering, subject
Food security	chain, healthcare, ge, mm, food, preparation, light, healthy, column, region
Animal health	health, animal, disease, effective, national, institute, dose, determined, administration, amount

Table D3: Focused topics identified by the initial model.

Category	Topic (Top 10 words)
Market and industrial opportunities	value, commercial, market, industrial, exists, business, global, adding, hepcidin, year
Cost and efficiency	efficiency, improve, increase, increasing, cost, enhance, reaction, efficient, amplification, low
Increasing compound yields	plant, efficiency, compound, yield, increasing, transformation, improved, gene, resistance, increased
Upscaling production	improve, production, improved, improvement, improving, higher, activity, greater, expression, enzyme
Scientific advancements	benefit, provide, field, improved, skin, cleaning, enzyme, surface, cellulase, material
Environmental sustainability	renewable, energy, reducing, biofuels, source, diesel, biofuel, carbon, lipid, product
Human health	disease, patient, cancer, diagnosis, effective, human, liver, lung, clinical, antibody
Food security	chain, sample, food, preparation, assay, light, supply, improved, donor, culture
Animal health	health, animal, medical, disease, care, healthy, nutrition, population, clinical, bethesda

Table D4: Focused topics identified by Model 23.

A Survey of Methods for Addressing Class Imbalance in Deep-Learning Based Natural Language Processing

Sophie Henning[1,2] **William Beluch**[1] **Alexander Fraser**[2] **Annemarie Friedrich**[1]
[1] Bosch Center for Artificial Intelligence, Renningen, Germany
[2] Center for Information and Language Processing, LMU Munich, Germany
sophieelisabeth.henning|william.beluch@de.bosch.com
fraser@cis.lmu.de
annemarie.friedrich@de.bosch.com

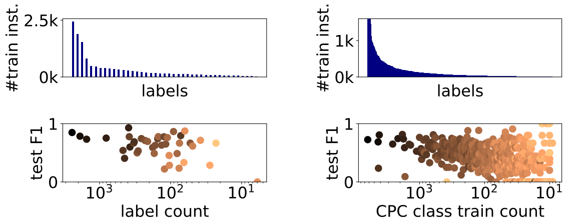
Abstract

Many natural language processing (NLP) tasks are naturally imbalanced, as some target categories occur much more frequently than others in the real world. In such scenarios, current NLP models tend to perform poorly on less frequent classes. Addressing class imbalance in NLP is an active research topic, yet, finding a good approach for a particular task and imbalance scenario is difficult.

In this survey, the first overview on class imbalance in deep-learning based NLP, we first discuss various types of controlled and real-world class imbalance. Our survey then covers approaches that have been explicitly proposed for class-imbalanced NLP tasks or, originating in the computer vision community, have been evaluated on them. We organize the methods by whether they are based on sampling, data augmentation, choice of loss function, staged learning, or model design. Finally, we discuss open problems and how to move forward.

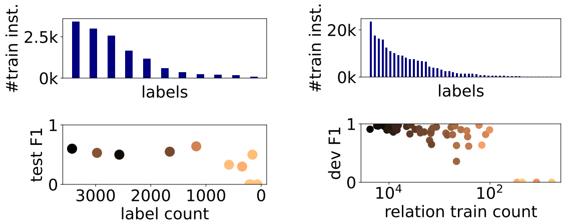
1 Introduction

Class imbalance is a major problem in natural language processing (NLP), because target category distributions are almost always skewed in NLP tasks. As illustrated by Figure 1, this often leads to poor performance on minority classes. Which categories matter is highly task-specific and may even depend on the intended downstream use. Developing methods that improve model performance in imbalanced data settings has been an active area for decades (e.g., Bruzzone and Serpico, 1997; Japkowicz et al., 2000; Estabrooks and Japkowicz, 2001; Park and Zhang, 2002; Tan, 2005), and is recently gaining momentum in the context of maturing neural approaches (e.g., Buda et al., 2018; Kang et al., 2020; Li et al., 2020; Yang et al., 2020; Jiang et al., 2021; Spangher et al., 2021). The problem is exacerbated when classes overlap in the feature space (Lin et al., 2019; Tian et al., 2020). For example, in patent classification, technical categories differ largely in frequency, and the concepts mentioned in the different categories can be very similar.



(a) Single-label **relation classification** on TACRED (Zhou and Chen, 2021)

(b) Hierarchical multi-label **patent classification** (Pujari et al., 2021)



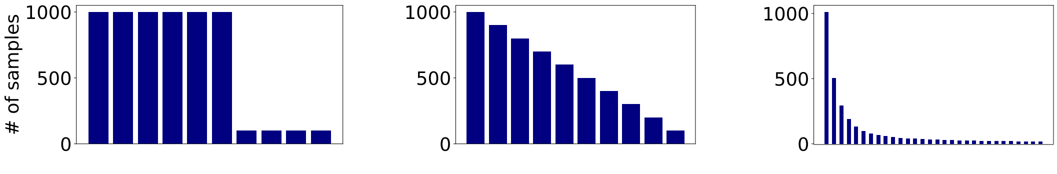
(c) Implicit **discourse relation** classification (PDTB) (Shi and Demberg, 2019)

(d) UD **dependency parsing** using RoBERTa on EWT (Grünewald et al., 2021)

Figure 1: **Class imbalance** has a negative effect on **performance** especially for minority classes in a variety of NLP tasks. Upper charts show label count distributions, lower part show test/dev F1 by training instance count (lighter colors indicate fewer test/dev instances). All models are based on transformers.

On a large variety of NLP tasks, transformer models such as BERT (Vaswani et al., 2017; Devlin et al., 2019) outperform both their neural predecessors and traditional models (Liu et al., 2019; Xie et al., 2020; Mathew et al., 2021). Performance for minority classes is also often higher when using self-supervised pre-trained models (e.g., Li and Scarton, 2020; Niklaus et al., 2021), which parallels findings from computer vision (Liu et al., 2022). However, the advent of BERT has not solved the class imbalance problem in NLP, as illustrated by Figure 1. Tänzer et al. (2022) find that on synthetically imbalanced named entity datasets with

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 523–540
May 2-6, 2023



(a) Step imbalance, $\mu = 0.4$, $\rho = 10$ (b) Linear imbalance, $\rho = 10$ (c) Long-tailed distribution

Figure 2: Instance counts per label follow different distributions: examples of **class imbalance types**.

majority classes having thousands of examples, at least 25 instances are required to predict a class at all, and 100 examples to learn to predict it with some accuracy.

Despite the relevance of class imbalance to NLP, related surveys only exist in the computer vision domain (Johnson and Khoshgoftaar, 2019b; Zhang et al., 2021b). Incorporating methods addressing class imbalance can lead to performance gains of up to 20%. Yet, NLP research often overlooks how important this is in practical applications, where minority classes may be of special interest.

Our contribution is to draw a clear landscape of approaches applicable to deep-learning (DL) based NLP. We set out with a problem definition (Sec. 2), and then organize approaches by whether they are based on sampling, data augmentation, choice of loss function, staged learning, or model design (Sec. 3). Our extensive survey finds that re-sampling, data augmentation, and changing the loss function can be relatively simple ways to increase performance in class-imbalanced settings and are thus straightforward choices for NLP practitioners.[1] While promising research directions, staged learning or model modifications often are implementation-wise and/or computationally costlier. Moreover, we discuss particular challenges of non-standard classification settings, e.g., imbalanced multi-label classification and catch-all classes, and provide useful connections to related computer vision work. Finally, we outline promising directions for future research (Sec. 4).

Scope of this survey. We focus on approaches evaluated on or developed for neural methods. Work from "traditional" NLP (e.g., Tomanek and Hahn, 2009; Li et al., 2011; Li and Nenkova, 2014; Kunchukuttan and Bhattacharyya, 2015) as well as Natural Language Generation (e.g., Nishino et al., 2020) and Automatic Speech Recognition (e.g., Winata et al., 2020; Deng et al., 2022) are not addressed in this survey. Other types of imbalances such as differently sized data sets of subtasks in continual learning (Ahrens et al., 2021) or imbalanced regression (Yang et al., 2021) are also beyond the scope of this survey. In Sec. 3.5, we briefly touch upon the related area of few-shot learning (Wang et al., 2020c).

Related surveys. We review imbalance-specific data augmentation approaches in Sec. 3.2. Feng et al. (2021) give a broader overview of data augmentation in NLP, Hedderich et al. (2021) provide an overview of low-resource NLP, and Ramponi and Plank (2020) discuss neural domain adaptation.

2 Problem Definition

Class imbalance refers to a classification setting in which one or multiple classes (**minority classes**) are considerably less frequent than others (**majority classes**). More concrete definitions, e.g., regarding the relative share up to which a class is seen as a minority class, depend on the task, dataset and labelset size. Much research focuses on improving all minority classes equally while maintaining or at least monitoring majority class performance (e.g., Huang et al., 2021; Yang et al., 2020; Spangher et al., 2021). We next discuss prototypical types of imbalance (Sec. 2.1) and then compare controlled and real-world settings (Sec. 2.2).

2.1 Types of Imbalance

To systematically investigate the effect of imbalance, Buda et al. (2018) define two prototypical types of label distributions, which we explain next.

Step imbalance is characterized by the fraction of minority classes, μ, and the size ratio between majority and minority classes, ρ. Larger ρ values indicate more imbalanced data sets. In prototypical step imbalance, if there are multiple minority classes, all of them are equally sized; if there are several majority classes, they also have equal size. Figure 2a shows a step-imbalanced distribution with 40% of the classes being minority classes

---

[1] We provide practical advice on identifying potentially applicable class imbalance methods in the Appendix (Figure 3).

and an imbalance ratio of $\rho = 10$. NLP datasets with a large catch-all class as they often arise in sequence tagging (see Sec. 2.2) or in relevance judgments in retrieval models frequently resemble step-imbalanced distributions. The ρ ratio has also been reported in NLP, e.g., by Li et al. (2020), although more task-specific imbalance measures have been proposed, e.g., for single-label text classification (Tian et al., 2020). In **linear imbalance**, class size grows linearly with imbalance ratio ρ (see Figure 2b), as, e.g., in the naturally imbalanced SICK dataset for natural language inference (Marelli et al., 2014).

Long-tailed label distributions (Figure 2c) are conceptually similar to linear imbalance. They contain many data points for a small number of classes (*head classes*), but only very few for the rest of the classes (*tail classes*). These distributions are common in computer vision tasks like instance segmentation (e.g., Gupta et al., 2019a), but also in multi-label text classification, for example with the goal of assigning clinical codes (Mullenbach et al., 2018), patent categories (Pujari et al., 2021), or news and research topics (Huang et al., 2021).

2.2 Controlled vs. Real-World Class Imbalance

Most real-world label distributions in NLP tasks do not perfectly match the prototypical distributions proposed by Buda et al. (2018). Yet, awareness of these settings helps practitioners to select appropriate methods for their data set or problem by comparing distribution plots. Using synthetically imbalanced data sets, researchers can control for more experimental factors and investigate several scenarios at once. However, evaluating on naturally imbalanced data provides evidence of a method's real-world effectiveness. Some recent studies combine both types of evaluation (e.g., Tian et al., 2021; Subramanian et al., 2021; Jang et al., 2021).

Many NLP tasks require treating a large, often heterogenous **catch-all class** that contains all instances that are not of interest to the task, while the remaining (minority) classes are approximately same-sized. Examples include the "Outside" label in IOB sequence tagging, or tweets that mention products in contexts that are irrelevant to the annotated categories (Adel et al., 2017). Such real-world settings often roughly follow a step imbalance distribution, with the additional difficulty of the catch-all class.

2.3 Evaluation

As *accuracy* and *micro*-averages mostly reflect majority class performance, choosing a good evaluation setting and metric is non-trivial. It is also highly task-dependent: in many NLP tasks, recognizing one or all minority classes well is at least equally important as majority class performance. For instance, non-hateful tweets are much more frequent in Twitter (Waseem and Hovy, 2016), but recognizing hateful content is the key motivation of hate speech detection. Which classes matter may even depend on downstream considerations, i.e., the same named entity tagger might be used in one application where a majority class matters, and another where minority classes matter more. Several evaluation metrics exist that have been designed to account for class-imbalanced settings, but no de facto standard exists. For example, *balanced accuracy* (Brodersen et al., 2010) corresponds to the average of per-class recall scores. It is often useful to record performance on *all* classes and to report *macro*-averages, which treat all classes equally.

3 Methods for Addressing Class Imbalance in NLP

In this section, we survey methods that either have been explicitly proposed to address class-imbalance issues in NLP or that have been empirically shown to be applicable for NLP problems. We provide an overview of which methods are applicable to a selection of NLP tasks in Appendix A.

3.1 Re-Sampling

To increase the importance of minority instances in training, the label distribution can be changed by various sampling strategies. Sampling can either be executed once or repeatedly during training (Pouyanfar et al., 2018). In *random oversampling* (**ROS**), a random choice of minority instances are duplicated, whereas in *random undersampling* (**RUS**), a random choice of majority instances are removed from the dataset. ROS can lead to overfitting and increases training times. RUS, however, discards potentially valuable data, but has been shown to work well in language-modeling objectives (Mikolov et al., 2013).

When applied in DL, ROS outperforms RUS both in synthetic step and linear imbalance (Buda et al., 2018) and in binary and multi-class English and Korean text classification (Juuti et al., 2020; Akhbardeh et al., 2021; Jang et al., 2021). More flexible variants, e.g., re-sampling only a tunable share of classes (Tepper et al., 2020) or interpolating between the (imbalanced) data distribution and an almost perfectly balanced distribution (Arivazhagan et al., 2019), can also further improve results. *Class-aware sampling* (**CAS**, Shen et al., 2016), also referred to as *class-balanced sampling*, first chooses a class, and then an instance from this class. Performance-based re-sampling during training, following the idea of Pouyanfar et al. (2018), works well in multi-class text classification (Akhbardeh et al., 2021).

Issues in multi-label classification. In multi-label classification, label dependencies between majority and minority classes complicate sampling approaches, as over-sampling an instance with a minority label may simultaneously amplify the majority class count (Charte et al., 2015; Huang et al., 2021). CAS also suffers from this issue, and additionally introduces within-class imbalance, as instances of one class are selected with different probabilities depending on the co-assigned labels (Wu et al., 2020). Effective sampling in such settings is still an open issue. Existing approaches monitor the class distributions during sampling (Charte et al., 2015) or assign instance-based sampling probabilities (Gupta et al., 2019b; Wu et al., 2020).

3.2 Data Augmentation

Increasing the amount of minority class data during corpus construction, e.g., by writing additional examples or selecting examples to be labeled using Active Learning, can mitigate the class imbalance problem to some extent (Cho et al., 2020; Ein-Dor et al., 2020). However, this is particularly laborious in naturally imbalanced settings as it may require finding “the needle in the haystack,” or may lead to biased minority class examples, e.g., due to collection via keyword queries. Synthetically generating additional minority instances thus is a promising direction. In this section, we survey data augmentation methods that have been explicitly proposed to mitigate class imbalance and that have been evaluated in combination with DL.

Text augmentation generates new natural language instances of minority classes, ranging from simple string-based manipulations such as synonym replacements to Transformer-based generation. *Easy Data Augmentation* (**EDA**, Wei and Zou, 2019), which uses dictionary-based synonym replacements, random insertion, random swap, and random deletion, has been shown to work well in class-imbalanced settings (Jiang et al., 2021; Jang et al., 2021; Juuti et al., 2020). Juuti et al. (2020) generate new minority class instances for English binary text classification using EDA and embedding-based synonym replacements, and by adding a random majority class sentence to a minority class document. They also prompt the pre-trained language model GPT-2 (Radford et al., 2019) with a minority class instance to generate new minority class samples. Tepper et al. (2020) evaluate generation with GPT-2 on English multi-class text classification datasets, coupled with a flexible balancing policy (see Sec. 3.1).

Similarly, Gaspers et al. (2020) combine machine-translation based text augmentation with dataset balancing to build a multi-task model. Both the main and auxiliary tasks are German intent classification. Only the training data for the latter is balanced and enriched with synthetic minority instances. In a long-tailed multi-label setting, Zhang et al. (2022) learn an attention-based text augmentation that augments instances with text segments that are relevant to tail classes, leading to small improvements. In general, transferring methods such as EDA or backtranslation to multi-label settings is difficult (Zhang et al., 2022, 2020; Tang et al., 2020).

Hidden space augmentation generates new instance vectors that are not directly associated with a particular natural language string, leveraging the representations of real examples. Using representation-based augmentations to tackle class imbalance is not tied to DL. SMOTE (Chawla et al., 2002), which interpolates minority instances with randomly chosen examples from their K-nearest neighbours, is popular in traditional machine learning (Fernández et al., 2018), but leads to mixed results in DL-based NLP (Ek and Ghanimifard, 2019; Tran and Litman, 2021; Wei et al., 2022). Inspired by CutMix (Yun et al., 2019), which cuts and pastes a single pixel region in an image, **TextCut** (Jiang et al., 2021) randomly replaces small parts of the BERT representation of one instance with those of the other. In binary and multi-class text classification experiments, TextCut improves over non-augmented BERT and EDA.

Good-enough example extrapolation (**GE3**, Wei, 2021) and **REPRINT** (Wei et al., 2022) also operate in the original representation space. To synthesize a new minority instance, GE3 adds the vector representing the difference between a majority class instance and the centroid of the respective majority class to the mean of a minority class. Evaluations on synthetically step-imbalanced English multi-class text classification datasets show improvements over oversampling and hidden space augmentation baselines. GE3 assumes that the distribution of data points of a class around its mean can be extrapolated to other classes, an assumption potentially hurting performance if the minority class distribution differs. To account for this when subtracting out majority characteristics, REPRINT performs a principal component analysis (PCA) for each class, leveraging the information on relevant dimensions during sample generation. This method usually outperforms GE3, with the cost of an additional hyperparameter (subspace dimensionality).

MISO (Tian et al., 2021) generates new instances by transforming the representations of minority class instances that are located nearby majority class instances. They learn a mapping from minority instance vectors to "disentangled" representations, making use of mutual information estimators (Belghazi et al., 2018) to push these representations away from the majority class and closer to the minority class. An adversarially-trained generator then generates minority instances using these disentangled representations. Tian et al. apply MISO in naturally and synthetically imbalanced English and Chinese binary and multi-class text classification with a single minority class.

ECRT (Chen et al., 2021) learns to map encoder representations (feature space) to a new space (source space) whose components are independent of each other given the class, assuming an invariant causal mechanism from source to feature space. The independence enables them to generate new meaningful minority examples by permuting or sampling components in the source space, resulting in medium improvements on a large multi-label text classification dataset with many labels.

Further related work exists in the area of **transfer learning** (Ruder et al., 2019), e.g., from additional datasets that provide complementary information on minority classes. For instance, Spangher et al. (2021) achieve small gains by manually selecting auxiliary datasets to improve imbalanced sentence-based discourse classification. However, complementary datasets have to be retrieved for each application, and task loss coefficients have to be tuned. Adapting methods to predict useful transfer sources (Lange et al., 2021) might help alleviate these problems.

3.3 Loss Functions

Standard *cross-entropy loss* (**CE**) is composed from the predictions for instances that carry the label in the gold standard, which is why the resulting classifiers fit the minority classes less well. In this section, we summarize loss functions designed for imbalanced scenarios. They either re-weight instances by class membership or prediction difficulty, or explicitly model class margins to change the decision boundary. Throughout this section, we use the variables and terms as shown in Table 1.

Losses for Single-Label Scenarios. *Weighted cross-entropy* (**WCE**) uses class-specific weights α_j that are tuned as hyperparameters or set to the inverse class frequency (e.g., Adel et al., 2017; Tayyar Madabushi et al., 2019; Li and Xiao, 2020). While WCE treats all instances of one class in the same way, *focal loss* (**FL**, Lin et al., 2017) down-weights instances for which the model is already confident (implemented with the $(1-p_j)^\beta$ coefficient). FL can of course also be used with class weights. Instead of mimicking accuracy like CE, *dice loss* (**Dice**, Milletari et al., 2016) tries to capture class-wise F1 score, with the predicted probability p_j proxying precision and the ground truth indicator y_j proxying recall. *Self-adjusting dice loss* (**ADL**, Li et al., 2020) combines confidence-based down-weighting via $1-p_j$ with Dice loss. For sequence labeling, QA and matching on English and Chinese datasets, Dice performs better than FL and ADL.

Rather than re-weighting instances, *label-distribution-aware margin loss* (**LDAM**, Cao et al., 2019), essentially a smooth hinge loss with label-dependent margins Δ_j, aims to increase the distance of the minority class instances to the decision boundary with the aim of better generalization for these classes. Cao et al.'s evaluation largely focuses on computer vision, but they also report results for LDAM on a synthetically imbalanced version of the IMDB review dataset (Maas et al., 2011), achieving a much lower error on the minority class than vanilla CE or CE with re-sampling or re-weighting. Subramanian et al. (2021) propose LDAM variants that consider bias related to socially salient groups (e.g., gender-based bias) in addition to class imbalance, evaluating them on binary text classification.

Single-label	**CE**	$-\sum_{j=1}^{C} y_j \log p_j$	**WCE**	$-\sum_{j=1}^{C} \alpha_j y_j \log p_j$
	FL	$-\sum_{j=1}^{C} y_j (1-p_j)^{\beta} \log p_j$		
	Dice	$\sum_{j=1}^{C} 1 - \frac{2p_j y_j + \gamma}{p_j^2 + y_j^2 + \gamma}$	**ADL**	$\sum_{j=1}^{C} 1 - \frac{2(1-p_j)p_j y_j + \gamma}{(1-p_j)p_j + y_j + \gamma}$
	LDAM	$-\sum_{j=1}^{C} y_j \log \frac{\exp(z_j - \Delta_j)}{\exp(z_j - \Delta_j) + \sum_{l \neq j} \exp(z_l)}$ with $\Delta_j = K/n_j^{1/4}$		
	RL	$\mathbb{1}(gt \neq A) \log(1 + \exp(\rho(m^+ - z_{gt}))) + \log(1 + \exp(\rho(m^- - z_{c^-})))$		
Multi-label	**BCE**	$-\sum_{j=1}^{C} [y_j \log p_j + (1-y_j) \log(1-p_j)]$		
	WBCE	$-\sum_{j=1}^{C} \alpha_j [y_j \log p_j + (1-y_j) \log(1-p_j)]$		
	FL	$-\sum_{j=1}^{C} [y_j (1-p_j)^{\beta} \log p_j + (1-y_j) p_j^{\beta} \log(1-p_j)]$		
	DB	$-\sum_{j=1}^{C} [y_j \hat{\alpha}_j (1-q_j)^{\beta} \log q_j + (1-y_j) \hat{\alpha}_j \frac{1}{\lambda} q_j^{\beta} \log(1-q_j)]$ with $q_j = y_j \sigma(z_j - v_j) + (1-y_j)\sigma(\lambda(z_j - v_j))$		

C	number of classes
y	target vector
p	model prediction vector
α	class weights
β	tunable focusing parameter
z	model logits vector
γ	smoothing constant
n_j	size of class j
K	label-independent constant
gt	index of ground-truth class
m^+	margin to correct class
m^-	... to most competitive incorrect class
A	special catch-all class
c^-	index of largest non-gt logit
λ	scaling factor
$\hat{\alpha}_j$	instance-specific class weights
v_j	class-specific bias

Table 1: Overview of **loss functions** formulated for one instance. See Appendix A for references/implementations.

In settings with a large artificial and potentially heterogeneous **catch-all class** (see Sec. 2.2), many areas of the space contain representations of the catch-all class. Here, vanilla LDAM might be an appropriate loss function as it encourages larger margins for minority classes. In such cases, *ranking losses* (**RL**) can also be effective to incentivize the model to only pay attention to "real" classes. On an imbalanced English multi-class dataset with a large catch-all class, Adel et al. (2017) find a ranking loss introduced by dos Santos et al. (2015) improves over CE and WCE. For minority classes, this loss function maximizes the score of the correct label z_{gt} while at the same time minimizing the score of the highest-scoring incorrect label z_{c^-}. For the catch-all class A, only z_{c^-} is minimized; z_{gt} is ignored. Similarly, Hu et al. (2022) apply class weights only to non-catch-all classes.

Losses for Multi-Label Scenarios. In multi-label classification, each label assignment can be viewed as a binary decision, hence *binary cross-entropy* (**BCE**) is often used here. Under imbalance, two issues arise. First, although class-specific weights have been used with BCE (e.g., Yang et al., 2020), their effect on minority classes is less clear than in the single-label case. For each instance, all classes contribute to BCE, with the labels *not* assigned to the instance (called *negative classes*) included via $(1-y_j)\log(1-p_j)$. Thus, if *weighted binary cross-entropy* (**WBCE**) uses a high weight for a class, it also increases the importance of negative instances for a minority class, which may further encourage the model to *not* predict this minority class.

To leverage class weights more effectively in BCE, one option is to only apply them to the loss of positive instances as proposed for multi-label image classification (Kumar et al., 2018). Related work includes uniformly upweighting positive instances of *all* classes in hierarchical multi-label text classification (e.g., Rathnayaka et al., 2019). An approach to multi-label emotion classification by Yilmaz et al. (2021) performs training time balancing by adapting FL such that for a given mini-batch the loss over all instances in this mini-batch has exactly the same value for every class.

If a classifier already correctly predicts a negative class for an instance, the loss can be further decreased by reducing the respective label's logits. In CE, due to the softmax that uses the logits of all classes, the impact of this effect becomes minor once the logit for the correct class is much larger than those of the other classes. However, the problem is more severe in BCE (Wu et al., 2020), as logits are treated independently. As minority labels mostly occur as negative classes, this logit suppression leads to a bias in the decision boundary, making it less likely for minority classes to be predicted. To tackle this issue and based on a multi-label version of FL, Wu et al. (2020) propose *distribution-balanced loss* (**DB**) for object detection, adding *Negative Tolerant Regularization* for the loss for negative classes by transforming the logits of positive and negative classes differently (see q_j in Table 1). This regularization imposes a sharp drop in the loss function for negative classes once the respective logit is below a threshold. Moreover, DB introduces instance-specific class weights $\hat{\alpha}$ to account for imbalances caused by class-aware sampling (see Sec. 3.1) in multi-label scenarios. These weights reflect the frequency of a class and the quantity and frequency of the positive labels of the instance. Huang et al. (2021) have shown large improvements of DB over BCE even when using uniform sampling on two long-tailed multi-label English text classification datasets.

While Cao et al. (2019) propose and theoretically justify LDAM for single-label classification only, it has been successfully applied to multi-label text classification as well (Biswas et al., 2021). Ferreira and Vlachos (2019) show that applying a cross-label dependency loss (Yeh et al., 2017; Zhang and Zhou, 2006) can be helpful for multi-label stance classification. Similarly, Lin et al. (2019) introduce a label-confusion aware cost factor into their loss function. The adaptive loss of Suresh and Ong (2021) integrates inter-label relationships into a contrastive loss (Khosla et al., 2020), which compares the score of a positive example with the distance to that of other positive and negative examples in order to push its representation closer to the correct class and further away from the wrong class(es). The resulting loss function learns how to increase the weight of confusable negative labels relative to other negative labels. Combining label-confusion aware loss functions with class weighting techniques is a promising research direction.

Re-Sampling vs. Loss Functions. Re-sampling and loss functions that are specifically designed for class-imbalanced settings are based on the same idea of increasing the importance of minority instances. Re-sampling is conceptually simpler and has a direct impact on training time, e.g., oversampling may cause a considerable increase. By contrast, the loss functions explained above are more flexible, e.g., by modeling desirable properties of margins, but also mostly harder to interpret.

3.4 Staged Learning

One approach to finding a good trade-off between learning features that are representative of the underlying data distribution and reducing the classifier's bias towards the majority class(es) is to perform the training in several stages. *Two-staged training* is common in imbalanced or data-scarce computer vision tasks (e.g., Wang et al., 2020b,a; Zhang et al., 2021a). The first stage usually performs standard training in order to train or fine-tune the feature extraction network. Later stages may freeze the feature extractor and re-train the classifier layers using special methods to address class imbalance, e.g., using more balanced data distributions or specific losses. For example, Cao et al. (2019) find their LDAM loss to be most effective when the training happens in two stages. In NLP, deep-learning models are usually based on pre-trained neural text encoders or word embeddings. Further domain-specific pre-training before starting the fine-tuning stage(s) can also be effective (Gururangan et al., 2020).

Several NLP approaches that fall under *staged learning* are directly inspired by computer vision research. In the context of long-tailed image classification, Kang et al. (2020) find that class-balanced sampling (see Sec. 3.1) helps when performing single-stage training, but that in their two-stage *classifier re-training* (**cRT**) method, using the original distribution in the first stage is more effective than class-balanced sampling. cRT employs the latter only in the second stage after freezing the representation weights. Yu et al. (2020) perform a similar decoupling analysis on long-tailed relation classification, essentially confirming Kang et al. (2020)'s results on this NLP task with respect to the re-sampling strategies. Additionally, they find that loss re-weighting under this analysis behaves similar to re-sampling, i.e., it leads to worse performance when applied during representation learning, but boosts performance when re-training the classifier. Hu et al. (2022) successfully leverage Kang et al.'s ideas for event detection, where both trigger detection and trigger classification suffer from class imbalance.

Jang et al. (2021) model imbalanced classification as a continual learning task with k stages where the data gradually becomes more balanced (*sequential targeting*, **ST**). The first stage contains the most imbalanced subset, and then the degree of imbalance decreases, with the last stage presenting the most balanced subset. The training objective encourages both good performance on the current stage and keeping information learnt in previous stages. Their experiments include binary and ternary English and Korean text classification. Active Learning (AL), which contains several stages by definition, has also been shown to boost performance of BERT models for minority classes (Ein-Dor et al., 2020). For a discussion about AL and deep learning, see Schröder and Niekler (2020).

3.5 Model Design

The methods described so far are largely independent of model architecture. This section describes model modifications that aim at improving performance in imbalanced settings.

Observing that the weight vectors for smaller classes have smaller norms in standard joint training compared to staged-learning based cRT (see Sec. 3.4), Kang et al. (2020) normalize the classifier weights directly in one-staged training using a hyperparameter τ to control the normalization "temperature" (**τ-norm**). τ-norm achieves similar or better performance than cRT in long-tailed image classification and outperforms cRT in relation extraction, but cRT works better for named entity recognition and event detection (Nan et al., 2021).

SetConv (Gao et al., 2020) and **ProtoBERT** (Tänzer et al., 2022) learn representatives for each class using *support sets* and classify an input (the *query*) based on its similarity to these representatives. SetConv applies convolution kernels that capture intra- and inter-class correlations to extract class representatives. ProtoBERT uses class centroids in a learned BERT-based feature space, treating the distance of any instance to the catch-all class as just another learnable parameter. At each training step, SetConv uses only one instance per class in the query set, but preserves the original class imbalance in the support set, whereas ProtoBERT uses fixed ratios. In the respective experimental studies, ProtoBERT performs better than using a standard classification layer on top of BERT for minority classes in NER if less than 100 examples are seen by the model, while SetConv excels in binary text classification with higher degrees of imbalance and in multi-class text classification.

The **HSCNN** model (Yang et al., 2020) uses class representatives only for the classification of tail classes, while head classes are assigned using a standard text CNN (Kim, 2014). HSCNN learns label-specific similarity functions, extracting instance representations from the pre-final layers of two copies of the original CNN, and assigns a tail class if the similarity to the class representative (computed as the mean of 5 random support instances) exceeds 0.5. On tail classes, HSCNN consistently improves over the vanilla CNN.

In addition, there exist a number of **task-specific solutions**. Prange et al. (2021) propose to construct CCG supertags from predicted tree structures rather than treating the problem as a standard classification task. In order to recognize implicit positive interpretations in negated statements in a class-imbalanced dataset, van Son et al. (2018) argue that leveraging information structure could be one way to improve inference. *Structural causal models* (**SCM**s) have been applied to imbalanced NLP tasks, encoding task-specific causal graphs (e.g., Nan et al., 2021). Similarly, Wu et al. (2021) causally model how bias in long-tailed corpora affects topic modeling (Blei et al., 2003) and use this to improve training of a variational autoencoder.

A research area closely related to class imbalance is **few-shot learning** (FSL, Wang et al., 2020c), which aims to learn classes based on only very few training examples. Model ideas from FSL can be leveraged for long-tailed settings, e.g., by making use of relational information about class labels in the form of knowledge graph embeddings or other forms of embedding hierarchical relationships between labels (Han et al., 2018; Zhang et al., 2019), or computing label-specific representations (Mullenbach et al., 2018).

4 Insights and Future Directions

We have provided a comprehensive, concise and structured overview of current approaches to dealing with class imbalance in DL-based NLP.

What works (best)? As there is no established benchmark for class-imbalanced settings, evaluation results are hard to compare across papers. In general, re-sampling or changing the loss function may lead to small to moderate gains. For data augmentation approaches, the reported performance increases tend to be larger than for re-sampling or new loss functions. The effects of staged training or modifications of the model vary drastically, ranging from detrimental to very large performance gains.

Hence, re-sampling, data augmentation, and changing the loss function are straightforward choices in class-imbalanced settings. Approaches based on staged learning or model design may sometimes outperform them, but often come with a higher implementation or computational cost. For a practical decision aid and potential application settings of some class imbalance methods, see Figure 3 in Appendix B and Table 3 in Appendix A.

How should we report results? Much NLP research only reports aggregate statistics (Harbecke et al., 2022), making it hard to judge the impact on improvements by class, which is often important in practice. We thus argue that NLP researchers should *always* report per-class statistics, e.g., as in Figure 1. Open-sourcing spreadsheets with the exact numbers would enable the community to compare systems more flexibly from multiple angles, i.e., with respect to whichever class(es) matter in a particular application scenario, and to re-use this data in research on class imbalance. Reviewers should also value works that analyze performance for relevant minority classes rather than focusing largely only on overall accuracy improvements.

A main hindrance to making progress on class imbalance in computer vision and NLP alike is that experimental results are often hard to compare (Johnson and Khoshgoftaar, 2019a, 2020). A first important step would be to not restrict baselines to methods of the same type, e.g., a new data augmentation approach should not only compare to other data augmentation methods, but also to using loss functions for class imbalance. Establishing a shared and systematic benchmark of a diverse set of class-imbalanced NLP tasks would be highly beneficial for both researchers and practitioners.

How can we move forward? Most work on class-imbalanced NLP has focused on single-label text classification. Finding good solutions for multi-label settings is still an open research challenge. Class imbalance also poses problems in NLP tasks such as sequence labeling or parsing, and we believe that the interaction of structured prediction models with methods to address class imbalance is a promising area for future research. Moreover, we need to study how class imbalance methods affect prediction calibration in order to provide reliable confidence estimates.

In general, inspiration for new model architectures could for example be drawn from approaches developed for few-shot learning (Wang et al., 2020c). Recently, prompting (Radford et al., 2019) has emerged as a new paradigm in NLP, which performs strongly in real-world few-shot settings (Schick and Schütze, 2022). Methods that improve worst-case performance under distribution shift (e.g., Sagawa et al., 2020) might also be applied to improve minority-class performance.

Acknowledgements

We thank the anonymous reviewers for their valuable comments, and Heike Adel, Stefan Grünewald, Subhash Pujari, and Timo Schrader for providing data for our teaser image. We thank them and Talita Anthonio, Mohamed Gad-Elrab, Lukas Lange, Stefan Ott, Robert Schmier, Hendrik Schuff, Daria Stepanova, Jannik Strötgen, Thang Vu, and Dan Zhang for helpful discussions and feedback on the writing. We also thank Jason Wei and Jiaqi Zeng for answering questions about their work.

Limitations

This paper is a survey, structuring, organizing and describing works and concepts to address *class imbalance* including *long-tailed learning*. While we touch upon *data augmentation* and *few-shot learning*, we do not comprehensively review those areas. Details on the scope of this review have also been described in Sec. 1.

The search process for the survey included searching for the keywords *class imbalance* and *long tail* in Google Scholar and the ACL Anthology, as well as carefully checking the papers that cite relevant papers.

Finally, the paper only constitutes a literature review, it does not yet provide a comprehensive empirical study which is much needed in this research area, but it will be of use in carrying out such a study.

References

Heike Adel, Francine Chen, and Yan-Ying Chen. 2017. Ranking convolutional recurrent neural networks for purchase stage identification on imbalanced Twitter data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 592–598, Valencia, Spain. Association for Computational Linguistics.

Kyra Ahrens, Fares Abawi, and Stefan Wermter. 2021. Drill: Dynamic representations for imbalanced lifelong learning. In *Artificial Neural Networks and Machine Learning – ICANN 2021*, pages 409–420, Cham. Springer International Publishing.

Farhad Akhbardeh, Cecilia Ovesdotter Alm, Marcos Zampieri, and Travis Desell. 2021. Handling extreme class imbalance in technical logbook datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4034–4045, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George F. Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In *Proceedings of the 35th*

International Conference on Machine Learning, volume 80 of *Proceedings of Machine Learning Research*, pages 531–540. PMLR.

Biplob Biswas, Thai-Hoang Pham, and Ping Zhang. 2021. Transicd: Transformer based code-wise attention model for explainable ICD coding. In *Artificial Intelligence in Medicine - 19th International Conference on Artificial Intelligence in Medicine, AIME 2021, Virtual Event, June 15-18, 2021, Proceedings*, volume 12721 of *Lecture Notes in Computer Science*, pages 469–478. Springer.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann. 2010. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pages 3121–3124.

L. Bruzzone and S.B. Serpico. 1997. Classification of imbalanced remote-sensing data by neural networks. *Pattern Recognition Letters*, 18(11):1323–1328.

Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. 2018. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Francisco Charte, Antonio J. Rivera, María J. del Jesus, and Francisco Herrera. 2015. Addressing imbalance in multilabel classification: Measures and random resampling algorithms. *Neurocomputing*, 163:3–16. Recent Advancements in Hybrid Artificial Intelligence Systems and its Application to Real-World Problems Progress in Intelligent Systems Mining Humanistic Data.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Junya Chen, Zidi Xiu, Benjamin Goldstein, Ricardo Henao, Lawrence Carin, and Chenyang Tao. 2021. Supercharging imbalanced data learning with energy-based contrastive representation transfer. *Advances in Neural Information Processing Systems*, 34.

Won Ik Cho, Youngki Moon, Sangwhan Moon, Seok Min Kim, and Nam Soo Kim. 2020. Machines getting with the program: Understanding intent arguments of non-canonical directives. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 329–339, Online. Association for Computational Linguistics.

Keqi Deng, Gaofeng Cheng, Runyan Yang, and Yonghong Yan. 2022. Alleviating asr long-tailed problem by decoupling the learning of representation and classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:340–354.

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

Cícero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 626–634, Beijing, China. Association for Computational Linguistics.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Adam Ek and Mehdi Ghanimifard. 2019. Synthetic propaganda embeddings to train a linear projection. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 155–161, Hong Kong, China. Association for Computational Linguistics.

Andrew Estabrooks and Nathalie Japkowicz. 2001. A mixture-of-experts framework for text classification. In *Proceedings of the ACL 2001 Workshop on Computational Natural Language Learning (ConLL)*.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. 2018. Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Researc*, 61:863–905.

William Ferreira and Andreas Vlachos. 2019. Incorporating label dependencies in multilabel stance detection. In *Proceedings of the 2019 Conference on*

Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 6350–6354, Hong Kong, China. Association for Computational Linguistics.

Yang Gao, Yi-Fan Li, Yu Lin, Charu Aggarwal, and Latifur Khan. 2020. SetConv: A New Approach for Learning from Imbalanced Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1284–1294, Online. Association for Computational Linguistics.

Judith Gaspers, Quynh Do, and Fabian Triefenbach. 2020. Data Balancing for Boosting Performance of Low-Frequency Classes in Spoken Language Understanding. In *Proc. Interspeech 2020*, pages 1560–1564.

Stefan Grünewald, Annemarie Friedrich, and Jonas Kuhn. 2021. Applying occam's razor to transformer-based dependency parsing: What works, what doesn't, and what is really necessary. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 131–144, Online. Association for Computational Linguistics.

Agrim Gupta, Piotr Dollar, and Ross Girshick. 2019a. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Agrim Gupta, Piotr Dollár, and Ross B. Girshick. 2019b. LVIS: A dataset for large vocabulary instance segmentation. *CoRR*, abs/1908.03195.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2245, Brussels, Belgium. Association for Computational Linguistics.

David Harbecke, Yuxuan Chen, Leonhard Hennig, and Christoph Alt. 2022. Why only micro-f1? class weighting of measures for relation classification. In *Proceedings of NLP Power! The First Workshop on Efficient Benchmarking in NLP*, pages 32–41, Dublin, Ireland. Association for Computational Linguistics.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Bo Hu, Yun Liu, Naiyue Chen, Lifu Wang, Ning Liu, and Xing Cao. 2022. Segcn-dcr: A syntax-enhanced event detection framework with decoupled classification rebalance. *Neurocomputing*, 481:55–66.

Yi Huang, Buse Giledereli, Abdullatif Köksal, Arzucan Özgür, and Elif Ozkirimli. 2021. Balancing methods for multi-label text classification with long-tailed class distribution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8153–8161, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joel Jang, Yoonjeon Kim, Kyoungho Choi, and Sungho Suh. 2021. Sequential targeting: A continual learning approach for data imbalance in text classification. *Expert Systems with Applications*, 179:115067.

Nathalie Japkowicz et al. 2000. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI workshop on learning from imbalanced data sets*, volume 68, pages 10–15. AAAI Press Menlo Park, CA.

Wanrong Jiang, Ya Chen, Hao Fu, and Guiquan Liu. 2021. Textcut: A multi-region replacement data augmentation approach for text imbalance classification. In *Neural Information Processing*, pages 427–439, Cham. Springer International Publishing.

Justin M. Johnson and Taghi M. Khoshgoftaar. 2019a. Deep learning and thresholding with class-imbalanced big data. *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 755–762.

Justin M. Johnson and Taghi M. Khoshgoftaar. 2019b. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1).

Justin M. Johnson and Taghi M. Khoshgoftaar. 2020. The Effects of Data Sampling with Deep Learning and Highly Imbalanced Big Data. *Information Systems Frontiers*, 22(5):1113–1131.

Mika Juuti, Tommi Gröndahl, Adrian Flanagan, and N. Asokan. 2020. A little goes a long way: Improving toxic language classification despite data scarcity. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2991–3009, Online. Association for Computational Linguistics.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2020. Decoupling representation and classifier for long-tailed recognition. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Pulkit Kumar, Monika Grewal, and Muktabh Mayank Srivastava. 2018. Boosted cascaded convnets for multilabel classification of thoracic diseases in chest radiographs. In *International conference image analysis and recognition*, pages 546–552. Springer.

Anoop Kunchukuttan and Pushpak Bhattacharyya. 2015. Addressing class imbalance in grammatical error detection with evaluation metric optimization. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 2–10, Trivandrum, India. NLP Association of India.

Lukas Lange, Jannik Strötgen, Heike Adel, and Dietrich Klakow. 2021. To share or not to share: Predicting sets of sources for model transfer learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8744–8753, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jinfen Li and Lu Xiao. 2020. syrapropa at SemEval-2020 task 11: BERT-based models design for propagandistic technique and span detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1808–1816, Barcelona (online). International Committee for Computational Linguistics.

Junyi Jessy Li and Ani Nenkova. 2014. Addressing class imbalance for improved recognition of implicit discourse relations. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 142–150, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Shoushan Li, Guodong Zhou, Zhongqing Wang, Sophia Yat Mei Lee, and Rangyang Wang. 2011. Imbalanced sentiment classification. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, page 2469–2472, New York, NY, USA. Association for Computing Machinery.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online. Association for Computational Linguistics.

Yue Li and Carolina Scarton. 2020. Revisiting rumour stance classification: Dealing with imbalanced data. In *Proceedings of the 3rd International Workshop on Rumours and Deception in Social Media (RDSM)*, pages 38–44, Barcelona, Spain (Online). Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2019. Cost-sensitive regularization for label confusion-aware event detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5278–5283, Florence, Italy. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Hong Liu, Jeff Z. HaoChen, Adrien Gaidon, and Tengyu Ma. 2022. Self-supervised learning is more robust to dataset imbalance. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14867–14875.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111, New Orleans, Louisiana. Association for Computational Linguistics.

Guoshun Nan, Jiaqi Zeng, Rui Qiao, Zhijiang Guo, and Wei Lu. 2021. Uncovering main causalities for long-tailed information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9683–9695, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joel Niklaus, Ilias Chalkidis, and Matthias Stürmer. 2021. Swiss-judgment-prediction: A multilingual legal judgment prediction benchmark. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 19–35, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Toru Nishino, Ryota Ozaki, Yohei Momoki, Tomoki Taniguchi, Ryuji Kano, Norihisa Nakano, Yuki Tagawa, Motoki Taniguchi, Tomoko Ohkuma, and Keigo Nakamura. 2020. Reinforcement learning with imbalanced dataset for data-to-text medical report generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2223–2236, Online. Association for Computational Linguistics.

Seong-Bae Park and Byoung-Tak Zhang. 2002. A boosted maximum entropy model for learning text chunking. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 482–489.

Samira Pouyanfar, Yudong Tao, Anup Mohan, Haiman Tian, Ahmed S. Kaseb, Kent Gauen, Ryan Dailey, Sarah Aghajanzadeh, Yung-Hsiang Lu, Shu-Ching Chen, and Mei-Ling Shyu. 2018. Dynamic sampling in convolutional neural networks for imbalanced data classification. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 112–117.

Jakob Prange, Nathan Schneider, and Vivek Srikumar. 2021. Supertagging the long tail with tree-structured decoding of complex categories. *Transactions of the Association for Computational Linguistics*, 9:243–260.

Subhash Chandra Pujari, Annemarie Friedrich, and Jannik Strötgen. 2021. A multi-task approach to neural multi-label hierarchical patent classification using transformers. In *European Conference on Information Retrieval*, pages 513–528. Springer.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Prabod Rathnayaka, Supun Abeysinghe, Chamod Samarajeewa, Isura Manchanayake, Malaka J. Walpola, Rashmika Nawaratne, Tharindu R. Bandaragoda, and Damminda Alahakoon. 2019. Gated recurrent neural network approach for multilabel emotion detection in microblogs. *CoRR*, abs/1907.07653.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks. In *International Conference on Learning Representations*.

Timo Schick and Hinrich Schütze. 2022. True few-shot learning with Prompts—A real-world perspective. *Transactions of the Association for Computational Linguistics*, 10:716–731.

Christopher Schröder and Andreas Niekler. 2020. A survey of active learning for text classification using deep neural networks. *CoRR*, abs/2008.07267.

Li Shen, Zhouchen Lin, and Qingming Huang. 2016. Relay backpropagation for effective learning of deep convolutional neural networks. In *Computer Vision – ECCV 2016*, pages 467–482, Cham. Springer International Publishing.

Wei Shi and Vera Demberg. 2019. Next sentence prediction helps implicit discourse relation classification within and across domains. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5790–5796, Hong Kong, China. Association for Computational Linguistics.

Alexander Spangher, Jonathan May, Sz-Rung Shiang, and Lingjia Deng. 2021. Multitask semi-supervised learning for class-imbalanced discourse classification. In *Proceedings of the 2021 Conference on*

Empirical Methods in Natural Language Processing, pages 498–517, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shivashankar Subramanian, Afshin Rahimi, Timothy Baldwin, Trevor Cohn, and Lea Frermann. 2021. Fairness-aware class imbalanced learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2045–2051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Varsha Suresh and Desmond Ong. 2021. Not all negatives are equal: Label-aware contrastive loss for fine-grained text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4381–4394, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Songbo Tan. 2005. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, 28(4):667–671.

Tiancheng Tang, Xinhuai Tang, and Tianyi Yuan. 2020. Fine-tuning bert for multi-label sentiment analysis in unbalanced code-switching text. *IEEE Access*, 8:193248–193256.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2022. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578, Dublin, Ireland. Association for Computational Linguistics.

Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle. 2019. Cost-sensitive BERT for generalisable sentence classification on imbalanced data. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 125–134, Hong Kong, China. Association for Computational Linguistics.

Naama Tepper, Esther Goldbraich, Naama Zwerdling, George Kour, Ateret Anaby Tavor, and Boaz Carmeli. 2020. Balancing via generation for multi-class text classification improvement. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1440–1452, Online. Association for Computational Linguistics.

Jiachen Tian, Shizhan Chen, Xiaowang Zhang, and Zhiyong Feng. 2020. A graph-based measurement for text imbalance classification. In *ECAI 2020*, pages 2188–2195. IOS Press.

Jiachen Tian, Shizhan Chen, Xiaowang Zhang, Zhiyong Feng, Deyi Xiong, Shaojuan Wu, and Chunliu Dou. 2021. Re-embedding difficult samples via mutual information constrained semantically oversampling for imbalanced text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3148–3161, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Katrin Tomanek and Udo Hahn. 2009. Reducing class imbalance during active learning for named entity annotation. In *Proceedings of the Fifth International Conference on Knowledge Capture*, K-CAP '09, page 105–112, New York, NY, USA. Association for Computing Machinery.

Nhat Tran and Diane Litman. 2021. Multi-task learning in argument mining for persuasive online discussions. In *Proceedings of the 8th Workshop on Argument Mining*, pages 148–153, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chantal van Son, Roser Morante, Lora Aroyo, and Piek Vossen. 2018. Scoring and classifying implicit positive interpretations: A challenge of class imbalance. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2253–2264, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tao Wang, Yu Li, Bingyi Kang, Junnan Li, Junhao Liew, Sheng Tang, Steven Hoi, and Jiashi Feng. 2020a. The devil is in classification: A simple framework for long-tail instance segmentation. In *Computer Vision – ECCV 2020*, pages 728–744, Cham. Springer International Publishing.

Xin Wang, Thomas Huang, Joseph Gonzalez, Trevor Darrell, and Fisher Yu. 2020b. Frustratingly simple few-shot object detection. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9919–9928. PMLR.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. 2020c. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Jason Wei. 2021. Good-enough example extrapolation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5923–5929, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Jiale Wei, Qiyuan Chen, Pai Peng, Benjamin Guedj, and Le Li. 2022. Reprint: a randomized extrapolation based on principal components for data augmentation. *CoRR*, abs/2204.12024.

Genta Indra Winata, Guangsen Wang, Caiming Xiong, and Steven C. H. Hoi. 2020. Adapt-and-adjust: Overcoming the long-tail problem of multilingual speech recognition. *CoRR*, abs/2012.01687.

Tong Wu, Qingqiu Huang, Ziwei Liu, Yu Wang, and Dahua Lin. 2020. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *Computer Vision – ECCV 2020*, pages 162–178, Cham. Springer International Publishing.

Xiaobao Wu, Chunping Li, and Yishu Miao. 2021. Discovering topics in long-tailed corpora with causal intervention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 175–185, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268.

Wenshuo Yang, Jiyi Li, Fumiyo Fukumoto, and Yanming Ye. 2020. HSCNN: A hybrid-Siamese convolutional neural network for extremely imbalanced multi-label text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6716–6722, Online. Association for Computational Linguistics.

Yuzhe Yang, Kaiwen Zha, Ying-Cong Chen, Hao Wang, and Dina Katabi. 2021. Delving into deep imbalanced regression. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 11842–11851. PMLR.

Chih-Kuan Yeh, Wei-Chieh Wu, Wei-Jen Ko, and Yu-Chiang Frank Wang. 2017. Learning deep latent space for multi-label classification. In *Thirty-first AAAI conference on artificial intelligence*.

Selim F. Yilmaz, E. Batuhan Kaynak, Aykut Koç, Hamdi Dibeklioğlu, and Suleyman Serdar Kozat. 2021. Multi-label sentiment analysis on 100 languages with dynamic weighting for label imbalance. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13.

Haiyang Yu, Ningyu Zhang, Shumin Deng, Zonggang Yuan, Yantao Jia, and Huajun Chen. 2020. The devil is the classifier: Investigating long tail relation classification with decoupling analysis. *CoRR*, abs/2009.07022.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Danqing Zhang, Tao Li, Haiyang Zhang, and Bing Yin. 2020. On data augmentation for extreme multi-label classification. *CoRR*, abs/2009.10778.

Jiaxin Zhang, Jie Liu, Shaowei Chen, Shaoxin Lin, Bingquan Wang, and Shanpeng Wang. 2022. Adam: An attentional data augmentation method for extreme multi-label text classification. In *Advances in Knowledge Discovery and Data Mining*, pages 131–142, Cham. Springer International Publishing.

Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338–1351.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3016–3025, Minneapolis, Minnesota. Association for Computational Linguistics.

Songyang Zhang, Zeming Li, Shipeng Yan, Xuming He, and Jian Sun. 2021a. Distribution alignment: A unified framework for long-tail visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2361–2370.

Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. 2021b. Deep long-tailed learning: A survey. *CoRR*, abs/2110.04596.

Wenxuan Zhou and Muhao Chen. 2021. An improved baseline for sentence-level relation extraction. *CoRR*, abs/2102.01373.

Appendix

A Method Overview

We here provide details on a selection of methods surveyed in this paper. Table 3 shows whether they have been applied respectively whether they are applicable in binary, multi-class, and multi-label classification. Moreover, it contains information on whether authors open-sourced their implementation. For links to open-sourced code, see Table 2.

B Practical advice

In Figure 3, we provide practical advice which class imbalance methods might be beneficial under which circumstances. Due to the lack of an established benchmark, we can only give rough guidance.

Method	**Link**
Data Augmentation	
EDA (Wei and Zou, 2019)	`GitHub`
GE3 (Wei, 2021)	`ACL Anthology`
ECRT (Chen et al., 2021)	`GitHub`
Loss Functions	
FL (Lin et al., 2017)	`GitHub`
ADL (Li et al., 2020)	`GitHub`
LDAM (Cao et al., 2019)	`GitHub`
DB (Wu et al., 2020)	`GitHub`
Staged Learning	
cRT (Kang et al., 2020)	`GitHub`
ST (Jang et al., 2021)	`GitHub`
Model Design	
τ-norm (Kang et al., 2020)	`GitHub`
ProtoBERT (Tänzer et al., 2022)	`GitHub`

Table 2: **Open-sourced implementations** of examples of class imbalance methods.

Method	Binary classification	Multi-class classification	Multi-label classification	Code
Re-Sampling				
ROS/RUS (Sec. 3.1)	✓	✓	?	N/A
CAS (Shen et al., 2016)	✓	✓	?	×
Data Augmentation				
EDA (Wei and Zou, 2019)	Juuti et al. (2020) Jiang et al. (2021)	Jiang et al. (2021)	Zhang et al. (2022, 2020)	✓
TextCut (Jiang et al., 2021)	Jiang et al. (2021)	Jiang et al. (2021)	✓	×
GE3 (Wei, 2021)	✓	Wei (2021) Wei et al. (2022)	?	✓
MISO (Tian et al., 2021)	Tian et al. (2021)	Tian et al. (2021)*	?	×
ECRT (Chen et al., 2021)	✓	✓	Chen et al. (2021)	✓
Loss Functions				
WCE (Sec. 3.3)	Tayyar Madabushi et al. (2019)	Adel et al. (2017) Li and Xiao (2020)	N/A	N/A
FL (Lin et al., 2017)	✓	Li et al. (2020) Nan et al. (2021)	✓	✓
ADL (Li et al., 2020)	✓	Li et al. (2020) Spangher et al. (2021)	✓	✓
LDAM (Cao et al., 2019)	Cao et al. (2019) Subramanian et al. (2021)	✓	Biswas et al. (2021)	✓
WBCE (Sec. 3.3)	✓	×	Yang et al. (2020)	N/A
RL (dos Santos et al., 2015)	✓	Adel et al. (2017)	×	×
DB (Wu et al., 2020)	×	×	Huang et al. (2021)	✓
Staged Learning				
cRT (Kang et al., 2020)	✓	Nan et al. (2021) Hu et al. (2022)	✓	✓
ST (Jang et al., 2021)	Jang et al. (2021)	Jang et al. (2021)	✓	✓
Model Design				
τ-norm (Kang et al., 2020)	✓	Nan et al. (2021)	✓	✓
SetConv (Gao et al., 2020)	Gao et al. (2020)	Gao et al. (2020)	✓	×
ProtoBERT (Tänzer et al., 2022)	✓	Tänzer et al. (2022)	✓	✓
HSCNN (Yang et al., 2020)	✓	✓	Yang et al. (2020)	×

Table 3: **Examples of class imbalance methods and NLP application settings.** ✓: **method** applicable (but no particular reference reporting experimental results exists)/**code**: authors open-sourced their implementation, ?: application not straightforward / open research issues. *: The authors select only one class as the minority class in their experiments. For links to open-sourced code, see Table 2. Methods for binary and multi-class classification are in general applicable to classification-based **relation extraction** approaches; applying class-imbalance techniques to **sequence labeling** methods in general is similar to the case of multi-label classification. For example, if sampling for a particular category, the whole sequence sample may contain additional annotations for other categories.

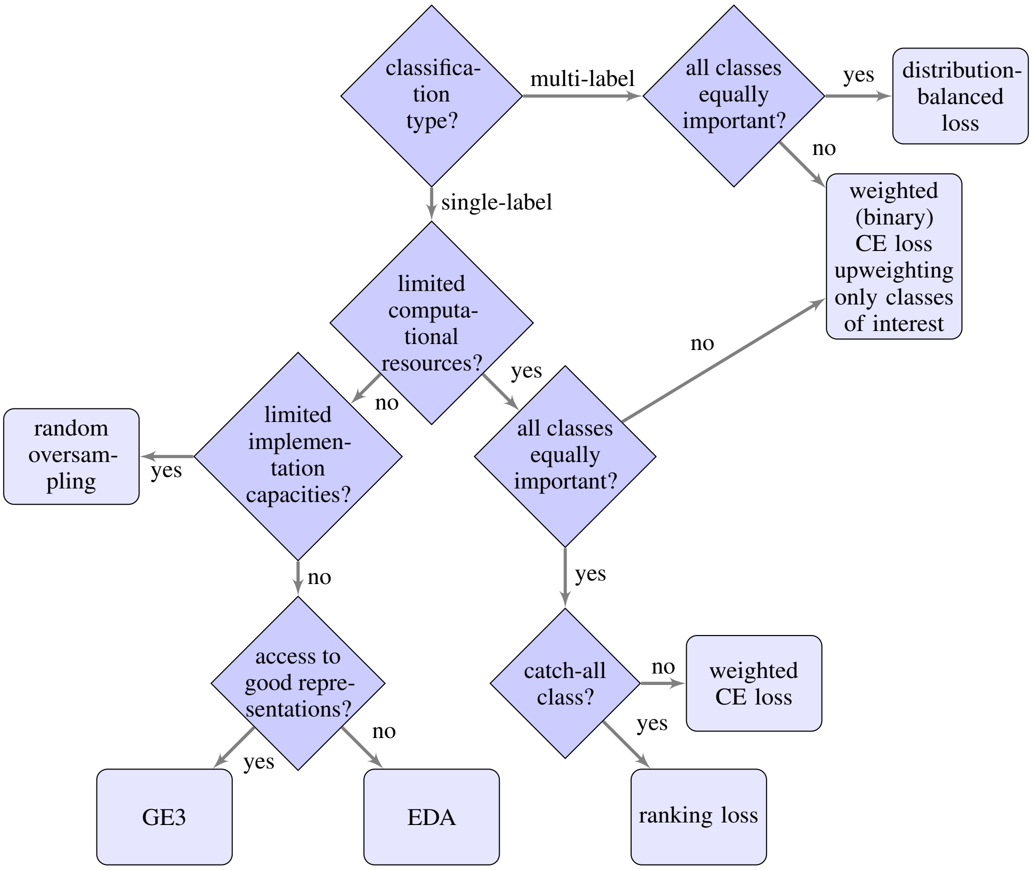


Figure 3: **Practical advice** which methods to try under which circumstances.

Extracting or Guessing? Improving Faithfulness of Event Temporal Relation Extraction

Haoyu Wang[1], **Hongming Zhang**[2], **Yuqian Deng**[1],
Jacob R. Gardner[1], **Muhao Chen**[3] **& Dan Roth**[1]
[1]Department of Computer and Information Science, UPenn
[2]Tencent AI Lab, Seattle
[3]Department of Computer Science, USC
`{why16gzl, yuqiand, jacobrg, danroth}@seas.upenn.edu`;
`hongmzhang@tencent.com`; `muhaoche@usc.edu`

Abstract

In this paper, we seek to improve the faithfulness of TempRel extraction models from two perspectives. The *first* perspective is to extract genuinely based on contextual description. To achieve this, we propose to conduct counterfactual analysis to attenuate the effects of two significant types of training biases: the *event trigger bias* and the *frequent label bias*. We also add tense information into event representations to explicitly place an emphasis on the contextual description. The *second* perspective is to provide proper uncertainty estimation and abstain from extraction when no relation is described in the text. By parameterization of Dirichlet Prior over the model-predicted categorical distribution, we improve the model estimates of the correctness likelihood and make TempRel predictions more selective. We also employ temperature scaling to recalibrate the model confidence measure after bias mitigation. Through experimental analysis on MATRES, MATRES-DS, and TDDiscourse, we demonstrate that our model extracts TempRel and timelines more faithfully compared to SOTA methods, especially under distribution shifts.

1 Introduction

Event temporal relation (TempRel) extraction is an essential step towards understanding narrative text, such as stories, novels, news, and guideline articles. With a robust temporal relation extractor, one can easily construct a storyline from text and capture the trend of temporally connected event mentions. TempRel extraction is also broadly beneficial to various downstream tasks including clinical narrative processing (Jindal and Roth, 2013; Bethard et al., 2016), question answering (Llorens et al., 2015; Meng et al., 2017; Stricker, 2021), and schema induction (Chambers and Jurafsky, 2009; Wen et al., 2021; Li et al., 2021).

Most existing TempRel extraction models are developed with data-driven machine learning approaches, for which recent studies also incorporate advanced learning and inference techniques such as structured prediction (Ning et al., 2017, 2018b; Han et al., 2019; Wang et al., 2020; Tan et al., 2021), graph representation (Mathur et al., 2021; Zhang et al., 2022), data augmentation (Ballesteros et al., 2020; Trong et al., 2022), and indirect supervision (Zhao et al., 2021; Zhou et al., 2021). These models are prevalently built upon pretrained language models (PLMs) and fine-tuned on a small set of annotated documents, e.g., TimeBank-Dense (Cassidy et al., 2014), MATRES (Ning et al., 2018c), and TDDiscourse (Naik et al., 2019).

A) I went to e_1:SEE the doctor. However, I was more seriously e_2:SICK. $\Longrightarrow$ e_1 After e_2

B) Microsoft said it has e_3:IDENTIFIED three companies for *the China program* to run through June. The company also e_4:GIVES each participating startup in *the Seattle program* $20,000 to create software. $\Longrightarrow$ e_3 Before e_4

Figure 1: Examples of unfaithful extractions. Before and After that follow the arrows denote the extracted TempRel's from the sentences by (Zhou et al., 2021).

Though these recent approaches have achieved promising evaluation results on benchmarks, whether they provide *faithful* extraction is an unexplored problem. The *faithfulness* of a relation extraction system is not simply about how much accuracy a system can offer. Instead, a faithful extractor should concern the validity and reliability of its extraction process. Specifically, when there is a TempRel to extract, a faithful extractor should genuinely obtain what is described in the context but not give *trivial guesses* from surface names of events or most frequent labels. Besides, when there is no relation described in the context, the system should selectively abstain from prediction.

We observe that in recent models, biases from prior knowledge in PLMs and statistically skewed training data often lead to unfaithful extractions

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 541–553
May 2-6, 2023 ©2023 Association for Computational Linguistics

(see Fig. 1). Example A thereof exhibits a case where the model adheres to the prior knowledge where people usually **see the doctor** *after* **getting sick**, but in this context **getting sick** is obviously a consequent of **seeing the doctor**. In Example B, BEFORE is extracted due to statistical biases learned from training data that BEFORE is not only the most frequent TEMPREL between ***identify*** and ***give***, but is also the most frequent TEMPREL between the first and second event in narrative order (Gee and Grosjean, 1984). However, with a closer inspection, it can be noticed that the two events in Example B are involved in different programs, one in *the China program*, the other in *the Seattle program*. Therefore, the system should abstain from prediction and give VAGUE as output.

In this paper, we seek to improve the faithfulness of TEMPREL extraction models from two perspectives. The *first* perspective is to guide the model to genuinely extract the described TEMPREL based on a relation-mentioning context. To achieve this goal, we conduct counterfactual analysis (Niu et al., 2021) to capture and attenuate the effects of two typical types of training biases: *event bias* caused by treating event trigger names as shortcuts for TEMPREL prediction, and *label bias* that causes the model prediction to lean towards more frequent training labels. We also propose to affix tense information to event mentions to explicitly place an emphasis on the contextual description.

The *second* perspective is to teach the model to abstain from extraction when no relation is described in the text. To know when to abstain, the models need to have a good estimate of the correctness likelihood. By incorporating Dirichlet Prior (Malinin and Gales, 2018, 2019) in the training phase of current TEMPREL extraction models, we improve the predictive uncertainty estimation of the models and make the TEMPREL predictions more selective. Furthermore, since the counterfactual analysis component (from the *first* perspective) may shift the model-predicted categorical distribution, we also employ temperature scaling (Guo et al., 2017) in inference to allow for recalibrated confidence measure of the model.

The technical contributions of our work are two-folds. First, to the best of our knowledge, this is the first study on the faithfulness issue of event-centric information extraction. Evidently, the development of a faithful TEMPREL extraction system contributes to more robust and reliable machine comprehension of events and narratives. Second, we propose training and inference techniques that can be easily plugged into existing neural TEMPREL extractors and effectively improve model faithfulness by mitigating prediction shortcuts and enhancing the capability of selective prediction.

Our contributions are verified with TEMPREL extraction experiments conducted on MATRES (Ning et al., 2018c), TDDiscourse (Naik et al., 2019) and distribution-shifted version of MATRES (MATRES-DS). Particularly, we evaluate on how precise and selective our TEMPREL extraction method is on in-distribution data, and how well it generalizes under distribution shift. Experimental results demonstrate that the techniques explored within the two aforementioned perspectives bring about promising results in improving faithfulness of current models. In addition, we also apply our method to the task of timeline construction (Do et al., 2012), showing that faithful TEMPREL extraction greatly benefits the accurate construction of timelines.

2 Related Work

Event TEMPREL Extraction. Recent event TEMPREL extraction approaches are mainly built on PLMs to obtain representations of event mentions and are improved with various learning and inference methodologies. To improve the quality of event representations, Mathur et al. (2021) embrace rhetorical discourse features and temporal arguments; Trong et al. (2022) select optimal context sentences via reinforcement learning to achieve SOTA performances; while Liu et al. (2021b); Mathur et al. (2021); Zhang et al. (2022) employ graph neural networks to avoid complex feature engineering. From the learning perspective, Ning et al. (2018a), Ballesteros et al. (2020), and Wang et al. (2020) enrich the models with auxiliary training tasks to provide complementary supervision signals, while Ning et al. (2018b), Zhao et al. (2021) and Zhou et al. (2021) bring into play distant supervision from heuristic cues and patterns. Nevertheless, recent data-driven models risk amplifying bias by exacerbating biases present in the pretraining and task training data when making predictions (Zhao et al., 2017). To rectify the models' biases towards prior knowledge in PLMs and shortcuts learned from biased training examples, our work proposes several training and inference techniques, seeking to improve the faithfulness of neural TEMPREL extractors as described in §1.

Bias Mitigation in NLP. Methods for mitigating prediction biases can be categorized as retraining and inference (Sun et al., 2019). Retraining methods address the bias in early stages or at its source. For instance, Zhang et al. (2017) masks the entities with special tokens to prevent relation extraction models from learning shortcuts from entity names, whereas several works conduct data augmentation (Park et al., 2018; Alzantot et al., 2018; Jin et al., 2020; Wu et al., 2022) or sample reweighting techniques (Lin et al., 2017; Liu et al., 2021a) to reduce biases in training. However, masking would result in the loss of semantic information and performance degradation, and it is costly to manipulate data or find proper unbiased data in temporal reasoning. Directly debiasing the training process may also hinder the model generalization on out-of-distribution (OOD) data (Wang et al., 2022). Therefore, inspired by several recent studies on debiasing text classification or entity-centric information extraction (Qian et al., 2021; Nan et al., 2021), our work adopts counterfactual inference to measure and control prediction biases based on automatically generated counterfactual examples.

Selective Prediction. Neural models have become increasingly accurate with the advances of deep learning. In the meantime, however, they should also indicate when their predictions are likely to be inaccurate in real-world scenarios. A series of recent studies have focused on resolving model miscalibration by measuring how closely the model confidences match empirical likelihoods. Among them, computationally expensive Bayesian (Gal and Ghahramani, 2016; Küppers et al., 2021) and non-Bayesian ensemble (Lakshminarayanan et al., 2017; Beluch et al., 2018) methods have been adopted to yield high quality predictive uncertainty estimates. Other methods have been proposed to use uncertainty reflected from model parameters to assess the confidence, including sharpness (Kuleshov et al., 2018) and softmax response (Hendrycks and Gimpel, 2017; Xin et al., 2021). Another class of methods adjust the models' output probability distribution by altering loss function in training via label smoothing (Szegedy et al., 2016) and Dirichlet Prior (Malinin and Gales, 2018, 2019). Besides, temperature scaling (Guo et al., 2017) also serves as a simple yet effective post-hoc calibration technique. In this paper, we model TEMPREL's with Dirichlet Prior in learning, and during inference we employ temperature scaling to recalibrate confidence measure of the model after bias mitigation.

3 Preliminaries

A document D is represented as a sequence of tokens $D = [w_1, \cdots, e_1, \cdots, e_2, \cdots, w_m]$, where some tokens belong to the set of annotated event triggers, i.e., $\mathcal{E}_D = \{e_1, e_2, \cdots, e_n\}$, and the rest are other lexemes. For a pair of events (e_i, e_j), the task of TEMPREL extraction is to predict a relation r from $\mathcal{R} \cup \{\text{VAGUE}\}$, where $\mathcal{R}$ denotes the set of TEMPREL's. An event pair is labeled VAGUE if the text does not express any determinable relation that belongs to $\mathcal{R}$. Let $\mathbf{y}^{(i,j)}$ denote the *model-predicted categorical distribution* over $\mathcal{R}$.

In order to provide a confidence estimate $\mathbf{y}$ that is as close as possible to the true probability, we first describe three separate factors (Malinin and Gales, 2018) that attribute to the predictive uncertainty for an AI system, namely epistemic uncertainty, aleatoric uncertainty, and distributional uncertainty. Epistemic uncertainty refers to the degree of uncertainty in estimating model parameters based on training data, whereas aleatoric uncertainty results from data's innate complexity. Distributional uncertainty arises when the model cannot make accurate predictions due to the lack of familiarity with the test data.

We argue that the way of handling VAGUE relations in existing TEMPREL extractors is problematic since they typically merge VAGUE into $\mathcal{R}$. In fact, VAGUE relations are complicated exception cases in the IE task, yet the annotation of such exceptions are never close to exhaustive in benchmarks, or even not given (Naik et al., 2019). In this work, we consider VAGUE relations as a source of **distributional uncertainty** and separately model them. Details are introduced in §4.2.

4 Methods

In this section, we first present how we obtain event representations and categorical distribution $\mathbf{y}$ in a local classifier for TEMPREL (§4.1). Then we introduce proposed learning and inference techniques to improve model faithfulness from the perspectives of selective prediction (§4.2) and prediction bias mitigation (§4.3), before we combine these two techniques with temperature scaling and introduce the OOD detection method in §4.4.

4.1 Local Classifier

Given that the context around an event pair (e_i, e_j)[1] has linguistic signals and temporal cues that are beneficial to TEMPREL prediction, the context of (e_i, e_j) considered in our model starts from the sentence before e_i and ends at the sentence after e_j. Inspired by Zhou and Chen (2021) for improving entity representation by prepending entity type information to entity mention spans, we add tense information of events into event trigger representations in this work. Accordingly, we enclose e_i and e_j with "@" and "#" respectively[2] and prepend their tense information to their spans with "$*$" and "$\wedge$". We provide a detailed example for affixing tense information in Appx. §A.1.

To characterize event pair (e_i, e_j), we obtain the two events' contextual representations and attention heads from PLMs. The classifer is trained to uncover the context that is critical to both events by multiplying their attentions before we send the concatenation of token embeddings and attention multiplication to a multi-layer perceptron (MLP) with $|\mathcal{R}|$ outputs. In this fashion, we obtain the $|\mathcal{R}|$-dimensional logits vector $\mathbf{z}^{(i,j)}$ and categorical distribution $\mathbf{y}^{(i,j)}$, where the probability of a label $r \in \mathcal{R}$ is given by the softmax function $\sigma(\cdot)$:

$$y_r = \sigma(\mathbf{z})_r = \frac{e^{z_r}}{\sum_{k=1}^{|\mathcal{R}|} e^{z_k}} . \quad (1)$$

4.2 Parameterization of Dirichlet Prior

As discussed in preliminaries (§3), VAGUE corresponds to complicated exception cases in inference. We model them as out-of-distribution (OOD) cases which are different from in-distribution (ID) data describing the relations in $\mathcal{R}$. The goal of providing high-quality confidence estimate $\mathbf{y}$ requires the model to yield a sharp predicted distribution centered on one of the labels in $\mathcal{R}$ when it is confident and yield a flat distribution over $\mathcal{R}$ for OOD inputs, as is shown in Fig. 2. To achieve this goal, we explicitly **parameterize a prior distribution** over categorical distributions. Because of the tractable analytic properties[3] of Dirichlet distribution (Eq. 2), we choose to parameterize a *sharp* and a *flat* Dirichlet prior over the model-predicted categorical distribution for ID and OOD inputs, respectively. The Dirichlet distribution is parameterized by its concentration parameters $\boldsymbol{\alpha}$, where α_0 is the precision of Dirichlet distribution. Higher values of α_0 lead to sharper, more confident predicted distributions.

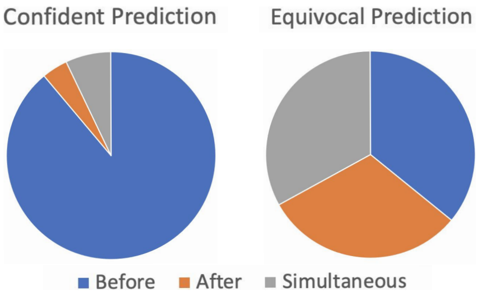


Figure 2: An illustration for desired behaviors of model predicted categorical distribution.

$$\begin{gathered}\texttt{Dir}(\mathbf{y};\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\prod_{k=1}^{|\mathcal{R}|}\Gamma(\alpha_k)} \prod_{k=1}^{|\mathcal{R}|} y_k^{\alpha_k - 1}, \\ \alpha_k > 0,\ \alpha_0 = \sum_{k=1}^{|\mathcal{R}|} \alpha_k .\end{gathered} \quad (2)$$

To attain the aforementioned behaviors, on ID data the model is trained to minimize the KL divergence between a sharp Dirichlet distribution and the model-predicted categorical distribution:

$$\mathcal{L}_{ID} = \mathbb{E}_{\mathrm{p}_{\mathrm{ID}}(\boldsymbol{x})}[KL[\mathrm{p}(\mathbf{y}) \parallel \texttt{Dir}(\mathbf{y};\boldsymbol{\alpha}_s)]], \quad (3)$$

where $\mathrm{p}_{\mathrm{ID}}(\boldsymbol{x})$ denotes ID data and $\boldsymbol{\alpha}_s$ denotes concentration parameters of the sharp Dirichlet distribution. On OOD data, the model minimizes the KL divergence between a flat Dirichlet distribution and the model-predicted categorical distribution:

$$\mathcal{L}_{OOD} = \mathbb{E}_{\mathrm{p}_{\mathrm{OOD}}(\boldsymbol{x})}[KL[\mathrm{p}(\mathbf{y}) \parallel \texttt{Dir}(\mathbf{y};\boldsymbol{\alpha}_f)]], \quad (4)$$

where $\mathrm{p}_{\mathrm{OOD}}(\boldsymbol{x})$ denotes OOD data and $\boldsymbol{\alpha}_f$ denotes the concentration parameters of the flat Dirichlet distribution. And the total loss of the model is

$$\mathcal{L} = \lambda_1 \mathcal{L}_{ID} + \lambda_2 \mathcal{L}_{OOD}, \quad (5)$$

where the λ's are hyperparameters to balance the influence of each loss. With the parameterization of Dirichlet prior, the learning process seeks to partly enhance the model's faithfulness by outputting confident estimates when it encounters ID inputs, and outputting equivocal estimates when the context does not express any TEMPREL in the meantime.

4.3 Counterfactual Analysis

After looking into the selective prediction perspective of faithfulness, we now address the other perspective: to mitigate biases from pre-trained knowledge and the task training data during the inference

---

[1] For event pair (e_i, e_j) we assume e_i is the first event in narrative order if $i < j$.

[2] Note that similar to *typed entity marker (punct)* by Zhou and Chen (2021), such enclosing has the benefit of highlighting mention spans without introducing new special tokens.

[3] $\Gamma(\cdot)$ in Eq. 2 denotes the gamma function.

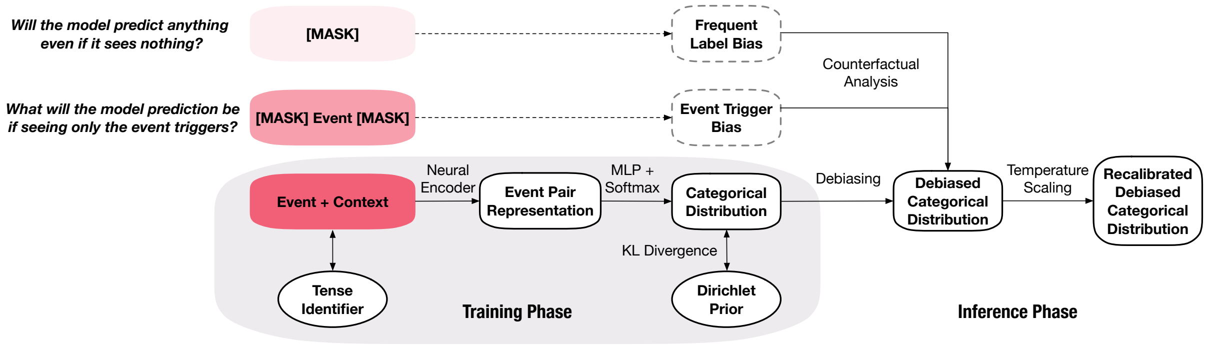


Figure 3: An overview of our approach to improving model faithfulness. In the training phase we obtain the model-predicted categorical distribution $\mathbf{y}$ with a neural encoder and parameterize a Dirichlet Prior over $\mathbf{y}$. And then we conduct counterfactual analysis to distill and mitigate biases during inference before leveraging temperature scaling to obtain recalibrated and debiased $\mathbf{y}$.

stage. Given that we have observed two types of biases in existing models, namely the event trigger bias and the frequent label bias, we ask the following questions:

- *What will the model prediction be if seeing the full context?*
- *What will the model prediction be if seeing only the event tiggers?*
- *Will the model predict anything even if it sees nothing?*

Inasmuch as we have described the learning process of the model, we know how to obtain model prediction given full context and can easily answer the first question. The second and third one, however, are hypothetical questions whose answers reflect the confounding biases that we would like to mitigate. With *attention masks* in recent PLMs (Devlin et al., 2019; Liu et al., 2019; Joshi et al., 2020; Lan et al., 2019), our model can be endowed with *imagination ability* effortlessly. By inputting a counterfactual instance with the context masked while maintaining the spans of event triggers to the model, we can obtain the model prediction given event trigger names only, which we denote by $\check{\mathbf{y}}$. And by sending an empty (counterfactual) instance, we obtain the model prediction where no textual information is given, which we denote by $\bar{\mathbf{y}}$. Intuitively, the two terms $\check{\mathbf{y}}$ and $\bar{\mathbf{y}}$ thereof provide measurements for the trigger bias and label bias.

Our goal is to use the biases assessed from model prediction (on counterfactual instances) to generate debiased categorical distribution. We remove the event trigger bias and the frequent label bias via element-wise subtraction, which is proved to be simple yet empirically effective (Qian et al., 2021):

$$\mathbf{y}' = \mathbf{y} - \beta_1 \check{\mathbf{y}} - \beta_2 \bar{\mathbf{y}}, \tag{6}$$

where $\mathbf{y}'$ denotes the debiased categorical distribution and the β's are independent parameters for balancing the terms that represent biases. We find the optimal values for β_1 and β_2 on different datasets[4] via grid beam search (Hokamp and Liu, 2017):

$$\hat{\beta}_1, \hat{\beta}_2 = \arg\max_{\beta_1, \beta_2} \psi(\beta_1, \beta_2), \quad \beta_1, \beta_2 \in [a, b], \tag{7}$$

where ψ is a metric function (e.g., F_1 scores) for evaluation, and a, b are the search boundaries.

In a nutshell, we obtain debiased categorical distribution by removing biases distilled via counterfactual inputs, thus encouraging the model to extract genuinely based on the contextual content. Nevertheless, the debiased model is not yet perfect. A minor drawback lies in that its confidence estimates might have been shifted by element-wise subtraction in Eq. 6 though it provides predictions with good evaluation results. Therefore we employ temperature scaling as our last step to allow for recalibrated confidence measure of the model.

4.4 Temperature Scaling and OOD Detection

The subtraction operation in Eq. 6 might result in negative values in $\mathbf{y}'$. To provide a proper estimate of the correctness likelihood, we first normalize the probabilities in $\mathbf{y}'$, where we replace the negative values with a small positive value and clips the values that are greater than 1:

$$\mathrm{norm}(y'_r) = \begin{cases} \epsilon, & if\ y'_r < 0 \\ 1 - \epsilon, & if\ y'_r > 1 \\ y'_r, & \text{otherwise} \end{cases} \tag{8}$$

---

[4] Using the development splits of datasets.

where ϵ denotes a small positive number, $r \in \mathcal{R}$. And then we use the inverse function of softmax to obtain the debiased logits vector $\mathbf{z}'$:

$$\mathbf{z}' = \sigma^{-1}(\text{norm}(\mathbf{y}')), \tag{9}$$

In this way we are able to apply temperature scaling (Guo et al., 2017) over $\mathbf{z}'$ and get the recalibrated and debiased categorical distribution $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} = \sigma(\mathbf{z}'/T), \tag{10}$$

where $T > 0$ denotes the temperature[5].

To detect OOD inputs, we need to measure the uncertainty of the model predictions. We use the *entropy* (Eq. 11) of the final categorical distribution $\hat{\mathbf{y}}$, which captures the uncertainty encapsulated in the entire distribution. On the dev set with VAGUE examples, we find the optimal threshold of $\mathcal{H}[\hat{\mathbf{y}}]$ below which the model predictions are considered equivoques and the inputs are OOD.

$$\mathcal{H}[\hat{\mathbf{y}}] = -\sum_{k=1}^{|\mathcal{R}|} \hat{y}_k \ln(\hat{y}_k) \ . \tag{11}$$

To sum up, we improve the model faithfulness in both training and inference phase with robust event presentations, Dirichlet Prior parameterization, counterfactual analysis and temperature scaling. The entire workflow is shown in Fig. 3.

5 Experiments

In this section, we describe the experiments[6] on two tasks: TEMPREL extraction and timeline construction. We first introduce the datasets that we adopt or create for evaluation (§5.1), followed by the evaluation protocols (§5.2). Evaluation results are discussed in §5.3 before we provide a detailed ablation study and case study in §5.4 and §5.5.

5.1 Datasets

We evaluate using the following datasets, for which statistics are given in Appx. §A.4.

MATRES (Ning et al., 2018c) is a TEMPREL benchmark annotated with the multi-axis scheme that helps achieve higher inter-annotator agreements (IAA) than previous benchmark datasets (Cassidy et al., 2014; Styler et al., 2014; O'Gorman et al., 2016). Four relations are annotated for the start time comparison of event pairs in 275 documents, namely BEFORE, AFTER, SIMUTANEOUS, and VAGUE. We train our model on the training set of MATRES, and evaluate our model on the dev and test sets of MATRES, MATRES-DS and TDDiscourse, which we introduce next.

MATRES-DS is an evaluation dataset that we created with distribution shifts (DS) compared to MATRES. Since one of our goals is to mitigate the bias of event triggers in the training data, we examine whether our proposed model stays uninfluenced when the distribution of event triggers is altered. We replace frequent triggers in the MATRES dev and test sets that appear within the top 5K frequent lemmas[7] with their uncommon synonyms, and replace infrequent triggers with their frequent synonyms from the list of frequent lemmas. MATRES-DS also presents a mismatch between the training and test distributions, or dataset shift (Quinonero-Candela et al., 2008), where distributional uncertainty often arises.

TDDiscourse (Naik et al., 2019) is a dataset for discourse-level event temporal ordering, in which TEMPREL's between global long-distance event pairs are annotated. As another data source with distribution shifts compared to MATRES, we adopt the manually annotated subset of TDDiscourse, namely TDD-man, in our experiments. The TEMPREL set $\mathcal{R}_T$[8] annotated in TDDiscourse is a superset of the TEMPREL set $\mathcal{R}_M$ defined in MATRES. Given that TDD-man serves as evaluation data on which we do not train our model, a relation in $\mathcal{R}_M \cup \{\text{VAGUE}\}$ is predicted for each pair of events in the test set of TDD-man.

5.2 Evaluation Protocols

For **event TEMPREL extraction**, we compare our model with the current and previous SOTA models (Trong et al., 2022; Mathur et al., 2021) trained on MATRES. The models are evaluated on not only how precise and selective their extraction is on ID data (MATRES), but are also examined for their generalizability under distribution shifts (MATRES-DS and TDD-man). We report micro-F_1 score as an evaluation metric following previous papers. We also report macro-F_1, which reflects the fairness of model prediction, and expected calibra-

---

[5] T is obtained by minimizing the negative log likelihood on the dev set. We refer readers to Appx. §A.3 for details.

[6] We refer readers to Appx. §A.5 for the discussion of experimental setup.

[7] `https://www.wordfrequency.info/`

[8] $\mathcal{R}_T$ = {BEFORE, AFTER, SIMULTANEOUS, INCLUDES, IS INCLUDED}.

Model	MATRES			MATRES-DS			TDD-man		
	micro-F_1	macro-F_1	ECE	micro-F_1	macro-F_1	ECE	micro-F_1	macro-F_1	ECE
Mathur et al. (2021)	82.3	55.7	12.8	76.7	52.3	16.3	82.1	52.8	20.3
Trong et al. (2022)	**83.4**	**56.4**	13.0	77.9	52.7	15.4	82.7	52.3	14.2
Ours	82.7	56.3	3.4	**78.7**	**54.7**	4.0	**83.1**	52.9	**5.8**
Ours w/o TI	81.8	55.2	**2.0**	77.3	52.4	8.6	79.5	**66.4**	21.9
Ours w/o DP	81.3	55.2	11.8	77.5	52.0	12.9	79.3	50.5	14.5
Ours w/o CA	80.3	54.7	5.0	78.6	52.9	**3.4**	83.0	52.7	6.4
Ours w/o TS	82.6	56.1	49.6	**78.7**	**54.7**	15.8	**83.1**	52.9	31.0

Table 1: Model performance on MATRES, MATRES-DS, and TDD-man for event TEMPREL extraction. The results of ablation study are shown in the last four rows, where TI, DP, CA and TS respectively stand for the four components in our model: Tense Information, Dirichlet Prior, Counterfactual Analysis and Temperature Scaling. Note that the numbers we report on MATRES-DS and TDD-man are model performances *under distribution shifts*.

Model	MATRES		MATRES-DS		TDD-man	
	Acc	MED	Acc	MED	Acc	MED
Mathur et al. (2021)	43.5	1.44	32.1	1.75	37.3	1.49
Trong et al. (2022)	44.7	1.36	28.0	1.96	30.5	1.55
Ours	**48.2**	**1.28**	**43.5**	**1.55**	**51.7**	**1.06**
Ours w/o TI	45.8	1.37	34.5	1.66	27.1	1.87
Ours w/o DP	38.7	1.48	28.6	1.93	23.3	1.85
Ours w/o CA	43.5	1.34	39.3	1.63	49.7	1.11
Ours w/o TS	**48.2**	**1.28**	**43.5**	**1.55**	**51.7**	**1.06**

Table 2: Model performance on MATRES, MATRES-DS, and TDD-man for timeline construction. The metrics are exact match accuracy (Acc) and minimum edit distance (MED) between prediction and ground truth.

tion error (ECE) that approximates the *difference in expectation between confidence and accuracy*. The definition of ECE is provided in Appx. §A.2.

We also apply our model to the **timeline construction** task, where the goal is to sort a list of events in a document in chronological order. To construct the timeline, the model first constructs a directed graph G with predicted non-VAGUE TEMPREL's between every event pairs. Then, edges in G with lowest confidence scores are removed until G becomes a directed acyclic graph (DAG). Finally, the timeline is generated as the linear ordering of the vertices in the DAG by topological sorting. In this way, we circumvent the possible conflicts in model predictions for timeline construction and the *faithful removal* of least confident edges serves as an examination on the quality of model-predicted confidence. On the three datasets, we report the accuracy of exact match and the average minimum edit distance between predicted and ground truth timelines as evaluation metrics.

5.3 Results

In Tab. 1, we report the TEMPREL extraction results. On MATRES, the SOTA model (Trong et al., 2022) still offers the best performance in terms of micro-F_1 whereas our model achieves comparable macro-F_1 score and lower calibration error. In contrast, our proposed *faithful* TEMPREL extractor outperforms baseline methods in terms of all evaluation metrics under the dataset shifts caused by replacement of event triggers in MATRES-DS and longer context distances between global event pairs in TDD-man. Specifically, our model shows a significant gain of 2.0% macro-F_1 and 0.8% micro-F_1 over the SOTA model on MATRES-DS and surpasses the previous SOTA model on TDD-man by 1.0% micro F_1, not to mention the improvements on confidence calibration. We attribute this superior performance under dataset shifts to the mitigation of biases from prior knowledge and training set statistics as well as the techniques we employ to improve predictive uncertainty estimation. For a visual illustration of model calibration, we present the reliability diagram that plots the expected sample accuracy as a function of confidence in Fig. 4.

Tab. 2 exhibits similar observations: our model outperforms both baselines on timeline construction by a large margin in terms of both metrics. Specifically, under dataset shifts within MATRES-DS and TDD-man, our model surpasses the best baseline by 11.4% and 14.4% in accuracy, while drastically reducing the minimum edit distance by relatively 11.4% and 28.9%. Evidently, the capabilities of selective prediction and bias mitigation make our model stand out in complex scenarios like timeline construction, whereas the bias and inferior calibration of existing models exacerbate unfaithful extractions when multiple decisions have to be made simultaneously.

5.4 Ablation Study

To analyze the effect of each model component, we conduct an ablation study of our model where we

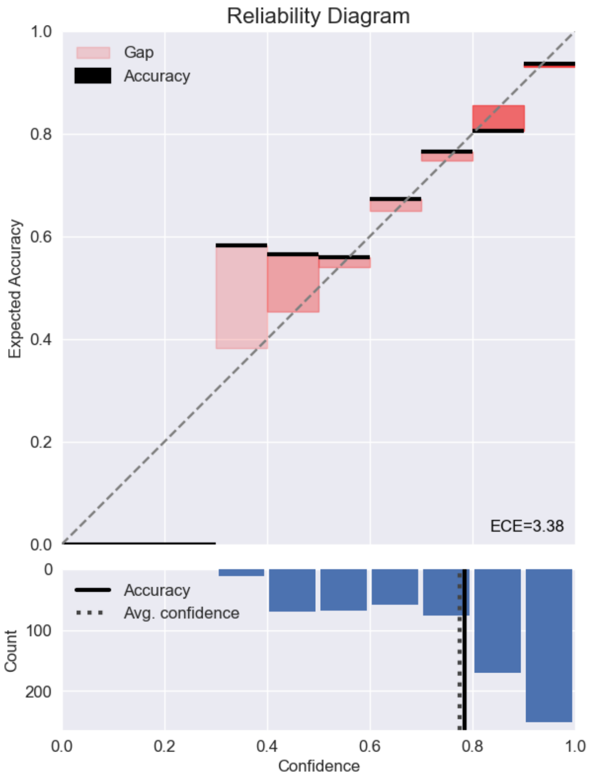


Figure 4: Reliability diagram and confidence histogram of our model predictions on test set of MATRES.

remove one component at a time (see Tab. 1)[9]. We observe on MATRES that, without the counterfactual analysis component, the model performance becomes worse by 2.4% in micro-F_1 and 1.6% in macro-F_1. Under dataset shifts, the model performance is reduced by 3.8% in micro-F_1 and 2.4% in macro-F_1 on TDD-man without the parameterization of Dirichlet Prior. The model performance in terms of F_1 scores is slightly influenced by taking away the temperature scaling component while the model calibration severely degrades.

From the ablation results in Tab. 2, we notice that temperature scaling has modest effects on the model performances, while Dirichlet Prior plays the most important role towards faithful timeline construction. It is also noteworthy that tense information considerably benefits the model to generalize well under distribution shifts in that it provides a useful feature applicable to all domains.

5.5 Case Study

As shown in Fig. 5, we provide a case study on timeline construction for three events. The reason why our model predicts AFTER for the third pair is probably due to the misleading temporal cues in text, Thursday and Tuesday, while the long distance between events undermines the confidence for this prediction. When our model builds a directed graph with three relations, a cycle is identified and the edge with lowest confidence is removed from the graph, and thus our model constructs the correct timeline. Without tense information, the model makes wrong prediction concerning the second event whose trigger is an adjective. And without Dirichlet Prior or temperature scaling, the model calibration becomes noticeably worse.

> A new Essex County task force began delving Thursday into the e_1:SLAYINGS of 14 people ... officials have been e_2:CAREFUL not to draw any firm conclusions, leaving open the possibility of a serial killer ... "I haven't e_3:SEEN a pattern yet," said Patricia Hurt, the Essex County prosecutor, who created the task force on Tuesday.

Model	(e_1, e_2)	(e_2, e_3)	(e_1, e_3)	Timeline
Gold	B	B	B	(e_1, e_2, e_3)
Ours	B, 0.92	B, 0.72	A, 0.51	(e_1, e_2, e_3)
Ours w/o TI	B, 0.99	A, 0.53	B, 0.63	(e_1, e_3, e_2)
Ours w/o DP	B, 0.92	A, 0.34	B, 0.52	(e_1, e_3, e_2)
Ours w/o CA	B, 0.94	B, 0.43	B, 0.49	(e_1, e_2, e_3)
Ours w/o TS	B, 0.43	B, 0.38	A, 0.36	(e_1, e_2, e_3)

Figure 5: Case study on timeline construction for one of the documents in TDD-man. The table shows predicted TEMPREL's and confidence for three event pairs, where B stands for BEFORE and A stands for AFTER. The cells in light red and light blue are wrong predictions and relations removed in timeline construction, respectively.

6 Conclusion

In this paper, we investigate on improving faithfulness for event TEMPREL extraction from two perspectives. To enhance the selectiveness of model predictions, we parameterize a Dirichlet Prior over the model-predicted categorical distribution to regularize the model to behave differently with ID and OOD data. To mitigate two types of biases from PLMs and training data, we add tense information to obtain robust event representations and conduct a counterfactual analysis to reduce the risk of carrying prediction shortcuts into inference. We also employ temperature scaling to combine the two faithful perspectives, which recalibrates the confidence measure of the model after bias mitigation. Through experimental analysis on MATRES, MATRES-DS, and TDDiscourse, we demonstrate that our model faithfully extracts event temporal relations and timelines from text, so as to generalize well under distribution shifts.

---

[9] We leverage cross-entropy as the training loss when we remove Dirichlet Prior in the training phase.

7 Acknowledgments

The authors would like to thank the anonymous ACL ARR reviewers for their insightful feedback on our work. This work was supported by Contract FA8750-19-2-1004 with the US Defense Advanced Research Projects Agency (DARPA). Approved for Public Release, Distribution Unlimited. This research is also based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the BETTER Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. In addition, this work is supported in part by the NSF Grant IIS 2105329, an Amazon Research Award and a Cisco Research Award.

Limitations

As the event representation introduced in our method is augmented with tense information, it potentially leads to limitations when applying to languages other than English, especially tenseless languages and languages having fewer tenses. The training of our models also requires considerable GPU resources which might produce environmental impacts, though the inference stage does not take up much computational resources.

Ethics Statement

There are no direct societal implications of this work. The proposed method attempts to provide high-quality and faithful event TEMPREL extraction and timeline construction. We believe that the intellectual merits of developing robust event-centric information extraction methods are demonstrated by this work. For any information extraction methods, real-world open source articles used to extract information may contain societal biases. Extracting event-event relations from articles with such biases may spread the bias into the acquired knowledge. Yet we believe that the proposed method can benefit various downstream NLP/NLU tasks like event prediction, task-oriented dialogue systems and risk detection.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Miguel Ballesteros, Rishita Anubhai, Shuai Wang, Nima Pourdamghani, Yogarshi Vyas, Jie Ma, Parminder Bhatia, Kathleen McKeown, and Yaser Al-Onaizan. 2020. Severing the edge between before and after: Neural architectures for temporal ordering of events. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5412–5417, Online. Association for Computational Linguistics.

William H. Beluch, Tim Genewein, Andreas Nürnberger, and Jan M. Köhler. 2018. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Steven Bethard, Guergana Savova, Wei-Te Chen, Leon Derczynski, James Pustejovsky, and Marc Verhagen. 2016. SemEval-2016 task 12: Clinical TempEval. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1052–1062, San Diego, California. Association for Computational Linguistics.

Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 501–506, Baltimore, Maryland. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610, Suntec, Singapore. Association for Computational Linguistics.

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

Quang Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational*

Natural Language Learning, pages 677–687, Jeju Island, Korea. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR.

James Paul Gee and Francois Grosjean. 1984. Empirical evidence for narrative structure. *Cognitive Science*, 8(1):59–85.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 434–444, Hong Kong, China. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International conference on learning representation*.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025.

Prateek Jindal and Dan Roth. 2013. Using soft constraints in joint inference for clinical concept recognition. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1808–1814, Seattle, Washington, USA. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. 2018. Accurate uncertainties for deep learning using calibrated regression. In *International conference on machine learning*, pages 2796–2804. PMLR.

Fabian Küppers, Jan Kronenberger, Jonas Schneider, and Anselm Haselhoff. 2021. Bayesian confidence calibration for epistemic uncertainty modelling. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Manling Li, Sha Li, Zhenhailong Wang, Lifu Huang, Kyunghyun Cho, Heng Ji, Jiawei Han, and Clare Voss. 2021. The future is not one-dimensional: Complex event schema induction by graph modeling for event prediction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5203–5215, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021a. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR.

Jian Liu, Jinan Xu, Yufeng Chen, and Yujie Zhang. 2021b. Discourse-level event temporal ordering with uncertainty-guided graph completion. In *IJCAI*, pages 3871–3877.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Hector Llorens, Nathanael Chambers, Naushad UzZaman, Nasrin Mostafazadeh, James Allen, and James Pustejovsky. 2015. SemEval-2015 task 5: QA TempEval - evaluating temporal information understanding with question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 792–800, Denver, Colorado. Association for Computational Linguistics.

Andrey Malinin and Mark Gales. 2018. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Andrey Malinin and Mark Gales. 2019. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. *Advances in Neural Information Processing Systems*, 32.

Puneet Mathur, Rajiv Jain, Franck Dernoncourt, Vlad Morariu, Quan Hung Tran, and Dinesh Manocha. 2021. TIMERS: Document-level temporal relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 524–533, Online. Association for Computational Linguistics.

Yuanliang Meng, Anna Rumshisky, and Alexey Romanov. 2017. Temporal information extraction for question answering using syntactic dependencies in an LSTM-based architecture. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 887–896, Copenhagen, Denmark. Association for Computational Linguistics.

Aakanksha Naik, Luke Breitfeller, and Carolyn Rose. 2019. TDDiscourse: A dataset for discourse-level temporal ordering of events. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 239–249, Stockholm, Sweden. Association for Computational Linguistics.

Guoshun Nan, Jiaqi Zeng, Rui Qiao, Zhijiang Guo, and Wei Lu. 2021. Uncovering main causalities for long-tailed information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9683–9695, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037, Copenhagen, Denmark. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018a. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2278–2288, Melbourne, Australia. Association for Computational Linguistics.

Qiang Ning, Hao Wu, Haoruo Peng, and Dan Roth. 2018b. Improving temporal relation extraction with a globally acquired statistical resource. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 841–851, New Orleans, Louisiana. Association for Computational Linguistics.

Qiang Ning, Hao Wu, and Dan Roth. 2018c. A multi-axis annotation scheme for event temporal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328, Melbourne, Australia. Association for Computational Linguistics.

Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. 2021. Counterfactual vqa: A cause-effect look at language bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12700–12710.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, Austin, Texas. Association for Computational Linguistics.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804, Brussels, Belgium. Association for Computational Linguistics.

Chen Qian, Fuli Feng, Lijie Wen, Chunping Ma, and Pengjun Xie. 2021. Counterfactual inference for text classification debiasing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5434–5445, Online. Association for Computational Linguistics.

Joaquin Quinonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. 2008. *Dataset shift in machine learning*. Mit Press.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the convergence of adam and beyond. In *International Conference on Learning Representations (ICLR)*.

Armand Stricker. 2021. Question answering in natural language: the special case of temporal expressions. In *Proceedings of the Student Research Workshop Associated with RANLP 2021*, pages 184–192, Online. INCOMA Ltd.

William F Styler, Steven Bethard, Sean Finan, Martha Palmer, Sameer Pradhan, Piet C De Groen, Brad Erickson, Timothy Miller, Chen Lin, Guergana Savova, et al. 2014. Temporal annotation in the clinical domain. *Transactions of the association for computational linguistics*, 2:143–154.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Xingwei Tan, Gabriele Pergola, and Yulan He. 2021. Extracting event temporal relations via hyperbolic geometry. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8065–8077, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hieu Man Duc Trong, Nghia Ngo Trung, Linh Van Ngo, and Thien Huu Nguyen. 2022. Selecting optimal context sentences for event-event relation extraction. In *AAAI Conference on Artificial Intelligencel Intelligence*.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706, Online. Association for Computational Linguistics.

Yiwei Wang, Muhao Chen, Wenxuan Zhou, Yujun Cai, Yuxuan Liang, Dayiheng Liu, Baosong Yang, Juncheng Liu, and Bryan Hooi. 2022. Should we rely on entity mentions for relation extraction? debiasing relation extraction with counterfactual analysis. In *NAACL*.

Haoyang Wen, Ying Lin, Tuan Lai, Xiaoman Pan, Sha Li, Xudong Lin, Ben Zhou, Manling Li, Haoyu Wang, Hongming Zhang, Xiaodong Yu, Alexander Dong, Zhenhailong Wang, Yi Fung, Piyush Mishra, Qing Lyu, Dídac Surís, Brian Chen, Susan Windisch Brown, Martha Palmer, Chris Callison-Burch, Carl Vondrick, Jiawei Han, Dan Roth, Shih-Fu Chang, and Heng Ji. 2021. RESIN: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 133–143, Online. Association for Computational Linguistics.

Yuxiang Wu, Matt Gardner, Pontus Stenetorp, and Pradeep Dasigi. 2022. Generating data to mitigate spurious correlations in natural language inference datasets. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2660–2676, Dublin, Ireland. Association for Computational Linguistics.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. The art of abstention: Selective prediction and error regularization for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1040–1051, Online. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297.

Shuaicheng Zhang, Lifu Huang, and Qiang Ning. 2022. Extracting temporal event relation with syntactic-guided temporal graph transformer. In *NAACL*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2989, Copenhagen, Denmark. Association for Computational Linguistics.

Xinyu Zhao, Shih-Ting Lin, and Greg Durrett. 2021. Effective distant supervision for temporal relation extraction. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 195–203, Kyiv, Ukraine. Association for Computational Linguistics.

Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal reasoning on implicit events from distant supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1361–1371, Online. Association for Computational Linguistics.

Wenxuan Zhou and Muhao Chen. 2021. An improved baseline for sentence-level relation extraction. *CoRR*, abs/2102.01373.

A Appendix

A.1 Example of event context affixed tense information

Original context: [CLS] For his part, Fidel Castro is the ultimate political survivor.[SEP] People have PREDICTED his demise so many times, and the US has TRIED to hasten it on several occasions.[SEP] Time and again, he endures.[SEP]

Context with affixed tense information: [CLS] For his part, Fidel Castro is the ultimate political survivor.[SEP] People have @ * *Present Perfect Simple* * PREDICTED @ his demise so many times, and the US has # ∧ *Present Perfect Simple* ∧ TRIED # to hasten it on several occasions.[SEP] Time and again, he endures.[SEP]

Figure 6: An example of the original context of event pair (PREDICTED, TRIED) and the context after affixing tense information to corresponding event spans.

A.2 Definition of ECE

Expected Calibration Error (ECE) metric (Guo et al., 2017) measures exactly the difference in expectation between confidence and accuracy. Empirically it is approximated by dividing the data into M confidence based bins, i.e., B_m (where $m \in \{1, 2, ..., M\}$) contains all datapoints i for which predicted confidence p_i lies in $(\frac{m-1}{M}, \frac{m}{M}]$. If acc$(B_m)$ and conf(B_m) denotes the average accuracy and prediction confidence for the points in B_m, ECE is defined as:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left| \mathrm{acc}(B_m) - \mathrm{conf}(B_m) \right|, \quad (12)$$

where n is the number of samples. The difference between acc and conf for a given bin represents the calibration gap (red bars in reliability diagrams – e.g. Fig. 4). We use ECE as the primary empirical metric to measure model calibration.

A.3 Negative Log Likelihood

Negative log likelihood is a standard measure of a probabilistic model's quality. It is also referred to as the cross entropy loss in the context of deep learning. Given a probabilistic model $\hat{\pi}(Y|X)$ and n samples, NLL is defined as:

$$\mathcal{L} = -\sum_{i=1}^{n} \log(\hat{\pi}(y_i|\bar{X}_i)) \quad (13)$$

It is a standard result that, in expectation, NLL is minimized if and only if $\hat{\pi}(Y|X)$ recovers the ground truth conditional distribution $\pi(Y|X)$. The temperature T in temperature scaling is optimized with respect to NLL on the dev sets.

A.4 Dataset Statistics

MATRES is composed of 275 news documents and the train/dev/test split is 183/72/20 documents where 6336/6404/818 event pairs are annotated respectively. The same statistics hold for MATRES-DS since we only change the event triggers in the inputs instead of the labels. In TD-Discourse, 4,000/650/1,500 and 32609/1435/4258 TEMPREL's are annotated in the train/dev/test sets of TDD-man and TDD-Auto, respectively.

A.5 Experimental Setup and Hyperparameter Setting

In the training phase, we fine-tune the pre-trained 1024-dimensional Big Bird (Zaheer et al., 2020) to encode the context of event triggers. We obtain the tense information of event triggers with an off-the-shelf tense identifier[10]. The parameters of the model are optimized using AMSGrad (Reddi et al., 2018) with the learning rate set to 5×10^{-6}, batch size set to 20, and the training process is limited to 40 epochs on a server with Nvidia A6000 GPU. All experiments are repeated with five different random seeds and the results reported are their average. To obtain $\boldsymbol{\alpha}_s$ in Eq. 3, we smooth the target means to redistribute a small amount of probability density to the other corners of the Dirichlet. In our experiments, we set $\lambda_1 = \lambda_2 = 1$ in Eq. 5. On the dev set of TDD-man the optimal β's of the model in Eq. 6 are $\beta_1 = -0.4, \beta_2 = 0.6$, where the search bounds, a and b equal to -1 and 1.

---

[10] `https://tense-sense-identifier.herokuapp.com/`

LoFT: Enhancing Faithfulness and Diversity for Table-to-Text Generation via Logic Form Control

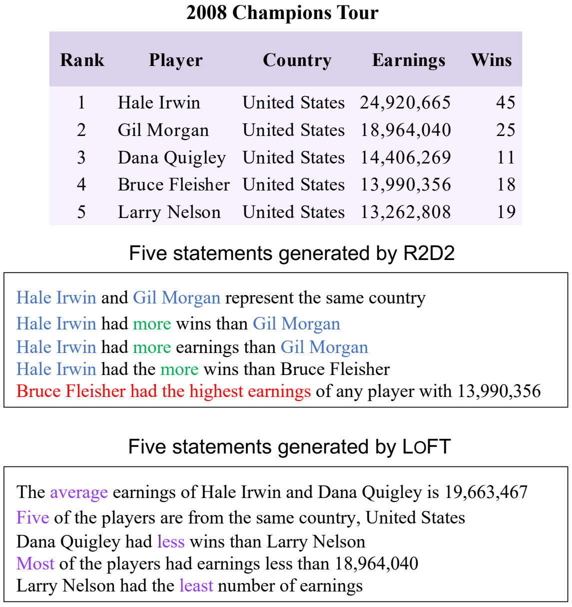
Yilun Zhao[*1] **Zhenting Qi**[*2] **Linyong Nan**[1]
Lorenzo Jaime Yu Flores[1] **Dragomir Radev**[1]
[1]Yale University [2] Zhejiang University
`yilun.zhao@yale.edu` `zhenting.19@intl.zju.edu.cn`

Abstract

Logical Table-to-Text (LT2T) generation is tasked with generating logically faithful sentences from tables. There currently exists two challenges in the field: 1) *Faithfulness*: how to generate sentences that are factually correct given the table content; 2) *Diversity*: how to generate multiple sentences that offer different perspectives on the table. This work proposes LoFT, which utilizes logic forms as fact verifiers and content planners to control LT2T generation. Experimental results on the LogicNLG dataset demonstrate that LoFT is the first model that addresses unfaithfulness and lack of diversity issues simultaneously. Our code is publicly available at `https://github.com/Yale-LILY/LoFT`.

1 Introduction

Table-to-Text (T2T) generation aims to produce natural language descriptions from structured tables. A statement generated from tabular data can be inferred based on different levels of information (e.g., value of a specific cell, logical operation result across multiple cells). Although current T2T models (Lebret et al., 2016; Wiseman et al., 2017; Puduppully et al., 2019; Parikh et al., 2020) have shown remarkable progress in fluency and coherence, they mainly focus on surface-level realizations without much logical inference.



2008 Champions Tour

Rank	Player	Country	Earnings	Wins
1	Hale Irwin	United States	24,920,665	45
2	Gil Morgan	United States	18,964,040	25
3	Dana Quigley	United States	14,406,269	11
4	Bruce Fleisher	United States	13,990,356	18
5	Larry Nelson	United States	13,262,808	19

Five statements generated by R2D2

Hale Irwin and Gil Morgan represent the same country
Hale Irwin had more wins than Gil Morgan
Hale Irwin had more earnings than Gil Morgan
Hale Irwin had the more wins than Bruce Fleisher
Bruce Fleisher had the highest earnings of any player with 13,990,356

Five statements generated by LoFT

The average earnings of Hale Irwin and Dana Quigley is 19,663,467
Five of the players are from the same country, United States
Dana Quigley had less wins than Larry Nelson
Most of the players had earnings less than 18,964,040
Larry Nelson had the least number of earnings

Figure 1: An example of logical table-to-text generation. (a) Statements generated by previous models (Nan et al., 2022): the generation suffers from 1) *Lack of diversity*, as three of the generated statements are focused on the same table regions (i.e., "Hale Irwin" and "Gil Morgan"), and three of them use the similar reasoning operations (i.e., comparative); 2) *Unfaithfulness*, as one of the generated statements is factually incorrect given the table content. (b) Statements generated by LoFT: By utilizing logic forms to *control* the generation, our method can generate multiple factually correct sentences that each use a different reasoning operation to offer various perspectives on the tabular data.

Recently, Chen et al. (2020a) proposed LogicNLG, which is tasked with generating textual descriptions that require logical reasoning over tabular data (i.e., LT2T generation). LT2T generation is challenging as it requires a model to learn the logical inference knowledge from table-text pairs and generate multiple *factually correct* sentences. Another challenge for LT2T generation is the *diversity* of generated text. Natural Language Generation (NLG) encourages the diverse output of statements over a single input, as it provides various perspectives on the data and offers users more choices. In LT2T generation, requirements for diversity naturally emerge from the need to apply different logical operations to extract different levels of table information. However, current methods (Chen et al., 2021; Nan et al., 2022; Liu et al., 2022a; Zhao et al., 2022b) that address issues of unfaithfulness have overlooked the importance of diversity. As shown in Figure 1, multiple statements generated using current methods (Nan et al., 2022) might only cover information from the same

---

[*] Equal Contributions.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 554–561
May 2-6, 2023 ©2023 Association for Computational Linguistics

table region or logical operation. Such issues related to lack of diversity could limit the deployment of LT2T models in the real world.

In this work, we attribute *unfaithfulness* and lack of *diversity* to the absence of *controllability* over generation. Specifically, due to the large number of combinations of different logical operations and table regions, the space of factually correct statements is exponentially large. However, LOGICNLG uses the whole table as the input, without providing annotations related to any other explicit control attribute. As a result, it is hard and uncontrollable for neural models to decide a favorable choice of logical selections solely based on the table input. We believe such *uncontrollability* leads to unfaithfulness and lack of diversity issues.

This work proposes LOFT, a framework that utilizes logic forms as mediators to enable *controllable* LT2T generation. Logic forms (Chen et al., 2020d,b) are widely used to retrieve evidence and explain the reasons behind table fact verification (Yang et al., 2020; Yang and Zhu, 2021; Ou and Liu, 2022). In this work, logic forms are used as: 1) fact verifiers to ensure the factual correctness of each generated sentence; and 2) content planners to control which logical operation and table region to use during the generation. Experimental results show that LOFT surpasses previous methods in faithfulness and diversity simultaneously.

2 Related Work

Logical Table-to-Text (LT2T) Generation LOGICNLG (Chen et al., 2020a) is tasked with generating logically faithful sentences from tables. To improve the faithfulness of generated statements, Nan et al. (2022) trained a system both as a generator and a faithfulness discriminator with additional replacement detection and unlikelihood learning tasks. Liu et al. (2022a) pre-trained a model on a synthetic corpus of table-to-logic-form generation. Zhao et al. (2022b) demonstrated that faithfulness of LT2T can be improved by pre-training a generative language model over synthetic Table QA examples. However, these methods overlook the importance of diversity in T2T generation, and might generate multiple statements that cover the same table regions or reasoning operations. Previous methods in NLG proposed to improve diversity by modifying the decoding techniques (Li et al., 2016). However, these approaches degrade faithfulness as measured against baselines (Perlitz et al., 2022). To enable controllable generation and improve diversity, Perlitz et al. (2022) used logical types of statements as a control. However, such methods still suffer from problems related to unfaithfulness, and may generate statements covering limited table regions. This work proposes to leverage the logic form as a fact checker and content planner to control LT2T generation, which tackles the challenges about faithfulness and diversity at the same time.

Table Fact Verification via Logic Form Logic forms are widely used in Table Fact Verification (Chen et al., 2020b). Specifically, given an input statement, the model (Yang et al., 2020; Yang and Zhu, 2021; Ou and Liu, 2022) will first translate it into logic form. Then the logic form will be executed over the table, and return `true/false` as the entailment label for a given statement. While several works (Chen et al., 2020d; Shu et al., 2021; Liu et al., 2021) focused on generating fluent statements from logic forms, the utilization of logic forms to benefit LT2T generation is still unexplored.

3 LOFT

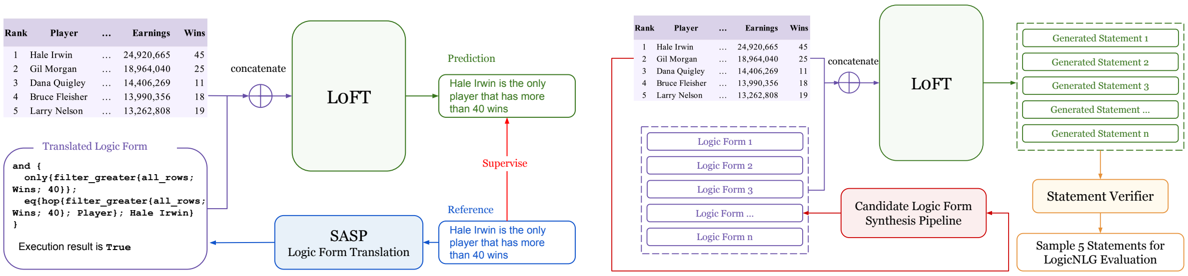
This section first introduces the logic form utilized, and then delves into the training and inference process of LOFT. We also explain how the use of logic forms can enhance both faithfulness and text-diversity in LT2T generation.

3.1 Logic Form Implementation

Logic forms are widely used to retrieve evidence and explain the reasons behind table fact verification. We use the same implementation as Chen et al. (2020d), which covers 8 types of the most common logical operations (e.g., count, aggregation) to describe a structured table. Each logical operation corresponds to several Python-based functions. For example, the definition of function `all_greater(view, header, value)` under "majority" category is: checking whether all the values under `header` column are greater than `value`, with the scope (i.e., `view`) of all or a subset of table rows. The complete list of logical operation types and corresponding function definitions are shown in Table 4 in Appendix.

3.2 LOFT Training

Training Task Formulation Given the serialized tabular data with selected columns as T, the train-



Rank	Player	...	Earnings	Wins
1	Hale Irwin	...	24,920,665	45
2	Gil Morgan	...	18,964,040	25
3	Dana Quigley	...	14,406,269	11
4	Bruce Fleisher	...	13,990,356	18
5	Larry Nelson	...	13,262,808	19

```
and {
  only{filter_greater{all_rows;
Wins; 40}};
  eq{hop{filter_greater{all_rows;
Wins; 40}; Player}; Hale Irwin}
}
```

Execution result is `True`

Rank	Player	...	Earnings	Wins
1	Hale Irwin	...	24,920,665	45
2	Gil Morgan	...	18,964,040	25
3	Dana Quigley	...	14,406,269	11
4	Bruce Fleisher	...	13,990,356	18
5	Larry Nelson	...	13,262,808	19

(a) LoFT training stage.

(b) LoFT inference stage.

Figure 2: The illustration of LoFT. (a) During the training stage, the SASP model is first applied to translate each statement in the LogicNLG training set into the logic form. Then LoFT is trained to generate the reference statement given the translated logic form and serialized table data. (b) During the inference stage, given each table, the logic form synthesis pipeline was first applied to synthesize candidate logic forms that cover different table regions and logical operations. LoFT is applied to generate statements for each candidate logic form. Then a statement verifier is used to filter out those potentially unfaithful statements. As a result, LoFT can generate a diverse set of faithful statements covering different table regions and reasoning operations. For each table in the LogicNLG test set, we randomly sampled five candidate statements for evaluation.

ing objective of LoFT is to generate a sentence $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ that is both fluent and faithful, with the translated logic form l as control.

$$\boldsymbol{y} = \text{argmax} \prod_{i=1}^{n} P(y_i | y_{<i}, T,\, l;\, \theta) \qquad (1)$$

where θ denotes the parameters of a seq2seq LM.

Training Dataset Collection Since the LogicNLG dataset does not contain logic form annotations, we had to augment each statement in the training set with its corresponding logic forms. To construct *{statement, logic form}* parallel data for the LogicNLG training set, we adapted SASP (Ou and Liu, 2022), the state-of-the-art model for TabFact dataset, which leverages structure-aware semantic parsing over tables to translate the given statement into logic form. In this work, given an example in the LogicNLG training set, SASP was applied to generate its logic form, resulting in a total of 15,637 examples for LoFT training.

3.3 LoFT Inference

During the inference stage, for each given table, we first applied the logic form synthesis pipeline to synthesize multiple candidate logic forms (Liu et al., 2022a). For each of these logic forms, the system generates its corresponding statement. The faithfulness of these statements were further checked by a verifier.

Logic Form Synthesis Pipeline To synthesize a candidate set of logic forms paired with each supporting table, we applied a similar logic form synthesis pipeline as Liu et al. (2022a).

We extracted templates of logic forms from the collected LoFT training dataset. Specifically, we categorized functions with similar definitions (e.g., `max/min`, `greater/less`) into smaller groups to obtain a more abstract template. Each function category corresponded to one unique table reasoning skill. For each template, we masked specific entities in the logic forms as typed placeholders (i.e., `col` to denote a column header, `obj` to denote an object). Finally, we obtained 45 different templates, covering 8 table logical operations. Table 4 shows the complete list of reasoning operations and corresponding function definitions.

Given the table and each set of selected columns, the pipeline would synthesize a total of 20 candidate logic forms whose execution result over the table is `True`. To generate a candidate logic form, the pipeline first sampled a logic form using a weighted-sampling technique with the weight equal to the template distribution in the LoFT training dataset (Section 3.2). The weighted sampling is to ensure that the generated candidate logic forms follow a similar distribution as LogicNLG. To instantiate the sampled template, a bottom-up sampling strategy is adopted to fill in each placeholder of the template and finally generate the logic form.

Statement Generation & Verification Through the logic form synthesis pipeline, we obtained a large number of candidate logic forms. For each logic form, we used LoFT to generate the corresponding statement. The candidate statements might still contain some factually incorrectness, thus we applied an NLI-based verifier to filter out those potentially unfaithful generations. Specifically, we used the TABFACT (Chen et al., 2020b) dataset to train a classifier, which adopts RoBERTa-base as the backbone. We fed each generated statement and its corresponding table into the classifier, and only kept those statements that were predicted as entailed. Then we randomly sampled five statements as the output for each table in LOGICNLG.

3.4 Enhancing LT2T via Logic Form Control

This subsection provides two perspectives to explain why logic forms can help improve both faithfulness and diversity of LT2T generation.

Logic Form as Content Planner Logic forms pass column or cell values as arguments, guiding the model to focus on relevant table regions. The function category of the logic form, such as `count`, helps the model better organize logical-level content planning.

Logic Form as Fact Verifier Logic forms are defined with unambiguous semantics, hence are reliable mediators to achieve faithful and controllable logical generations. During the inference stage, we synthesize candidate logic forms with 100% execution correctness. The sampled logic form serves as a fact verifier and conveys accurate logical-level facts for controllable LT2T generation.

4 Experimental Setup

We next discuss the evaluation metrics, baselines, and implementation details for the experiments.

4.1 Evaluation Metrics

We applied various automated evaluation metrics at different levels to evaluate the model performance from multiple perspectives.

Surface-level Following Chen et al. (2020a), we used BLEU-1/2/3 to measure the consistency of generated statements with the reference.

Diversity-level We used Distinct-n (Li et al., 2016) and self-BLEU-n (Zhu et al., 2018) to measure the diversity of five generated statements for each table. Distinct-n is defined as the total number of distinct n-grams divided by the total number of tokens in the five generated statements; Self-BLEU-n measures the average n-gram BLEU score between generated statements. We measured Distinct-2 and Self-BLEU-4 in our experiment.

Faithfulness-level Similar as the previous works (Chen et al., 2020a; Nan et al., 2022; Liu et al., 2022a), we used a parsing-based evaluation metric (i.e., SP-Acc) and two NLI-based evaluation metrics (i.e., NLI-Acc and TAPEX-Acc) to measure the faithfulness of generation. SP-Acc directly extracts the meaning representation from the generated sentence and executes it against the table to verify the correctness. NLI-Acc and TAPEX-Acc use TableBERT (Chen et al., 2020b) and TAPEX (Liu et al., 2022b) respectively as their backbones, and were finetuned on the TABFACT dataset (Chen et al., 2020b). Liu et al. (2022a) found that NLI-Acc is overly positive about the predictions, while TAPEX-Acc is more reliable to evaluate the faithfulness of generated sentences.

4.2 Baseline Systems

We implemented following baseline systems for the performance comparison: **GPT2-TabGen** (Chen et al., 2020a) directly fine-tunes GPT-2 over the LOGICNLG dataset; **GPT2-C2F** (Chen et al., 2020a) first produces a template which determines the global logical structure, and then generates the statement conditioned on the template; **DCVED** (Chen et al., 2021) applies a deconfounded variational encoder-decoder to reduce the spurious correlations during LT2T generation training; **DEVTC** (Perlitz et al., 2022) utilized reasoning operation types as an explicit control to increase the diversity of LT2T generation; and **R2D2** (Nan et al., 2022) trains a generative language model both as a generator and a faithfulness discriminator with additional replacement detection and unlikelihood learning tasks, to enhance the faithfulness of LT2T generation.

4.3 Implementation Details

Following Shu et al. (2021), we converted each logic form into a more human-readable form for both LOFT training and inference data. LOFT was implemented using fairseq library (Ott et al., 2019), with BART-Large (Lewis et al., 2020) as the backbones. All experiments were conducted on an 8 NVIDIA RTX-A5000 24GB cluster. Both LOFT and the statement verifier was trained for 5,000 steps with a batch size of 128. The best checkpoints were selected by the validation loss.

Model	Surface-level	Diversity-level		Faithfulness-level		
	BLEU-1/2/3↑	Distinct-2↑	s-BLUE-4↓	SP-Acc↑	NLI-Acc↑	TAPEX-Acc↑
GPT2-TabGen (Chen et al., 2020a)	48.8/27.1/12.6	59.0	55.3	42.1	68.7	45.0
GPT2-C2F (Chen et al., 2020a)	46.6/26.8/13.3	60.3	52.8	42.7	72.2	44.1
DCVED* (Chen et al., 2021)	49.5/28.6/15.3	–	–	43.9	76.9	–
DEVTC‡ (Perlitz et al., 2022)	51.3/30.6/16.3	73.7	21.3	44.3	77.9	55.6
R2D2 (Nan et al., 2022)	51.8/32.4/18.6	60.1	51.5	50.8	85.6	60.2
LoFT	48.1/27.7/14.9	**79.5**	**17.7**	**57.7**	**86.9**	**61.8**

Table 1: Performance on the LogicNLG test set. ‡: results from our own implementation; ∗: code not released and we used the results reported in original papers. LoFT achieves great improvement on faithfulness and diversity.

Diversity Criteria	DEVTC		R2D2		LoFT	
	Best↑	Worst↓	Best↑	Worst↓	Best↑	Worst↓
Table Coverage	8	16	5	20	29	5
Reasoning Op	19	1	2	37	24	2

Table 2: Number of times the system was selected as best or worst by majority vote (including ties). LoFT outperforms other baselines in terms of diversity for both table coverage and reasoning operations.

Model	Faithfulness ↑ Agreement / κ	Fluency ↑ Agreement / κ
DEVTC	63.5 / 0.69	86.5 / 0.80
R2D2	71.5 / 0.73	**90.0** / 0.84
LoFT	**75.0** / 0.76	88.0 / 0.81

Table 3: Human evaluation results on the criteria of faithfulness and fluency, with the total agreement by Fleiss' Kappa (κ) (Fleiss, 1971). LoFT has the best performance in terms of faithfulness, while achieving comparable performance in fluency.

5 Experimental Results

This section discusses automated and human evaluation results of different systems.

5.1 Main Results

Table 1 presents the results on LogicNLG. LoFT outperforms all the baselines on the criteria of diversity and faithfulness, and is the first model that achieves state-of-the-art results on both faithfulness- and diversity-level. It is worth noting that in the LogicNLG setting, a generated statement is allowed to cover a different table region or reasoning operations from the references, as long as it is fluent and factually correct. However, in such cases, the reference-based metrics will be low, explaining why the BLEU-1/2/3 scores of LoFT are lower than other models.

5.2 Human Evaluation

We conducted the human evaluation with four expert annotators using the following three criteria: (1) *Faithfulness* (scoring 0 or 1): if all facts contained in the generated statement are entailed by the table content; (2) *Diversity* (voting the best & worst): if the five generated statements cover information from different table regions, and use different reasoning operations; (3) *Fluency* (scoring 0 or 1): if the five generated statements are fluent and without any grammar mistakes.

We chose R2D2 (Nan et al., 2022) and DEVTC (Perlitz et al., 2022) for comparison, as they achieved best-performance results in faithfulness and diversity, respectively. We sampled 50 tables from the LogicNLG test set. For each table, we selected all five generated statements from each model's output. To ensure fairness, the model names were hidden to the annotators, and the display order between three models was randomly shuffled. Human evaluation results show that LoFT delivers improvements in both faithfulness (Table 3) and diversity (Table 2), while achieving comparable performance in fluency (Table 3).

6 Conclusions

This work proposes LoFT, which utilizes logic forms as fact verifiers and content planners to enable controllable LT2T generation. Experimental results on LogicNLG demonstrate that LoFT delivers a great improvement in both diversity and faithfulness of LT2T generation.

Limitations

The first limitation of our approach is that LoFT does not explore long text generation (Moosavi et al., 2021). LoFT only supports the generation of multiple single sentences. To enable long text generation (i.e., generate a long paragraph that delivers various perspectives on the table data), a global content planner (Su et al., 2021) needs to be designed to highlight which candidate sentences should be mentioned and in which order. Additionally, we believe that LoFT can also be applied to text generation over hybrid context with both textual and tabular data (Chen et al., 2020c; Zhao et al., 2022a; Nakamura et al., 2022).

The second limitation of our work is that the statement verifier discussed in Section 3.3 was trained using the same data as NLI-Acc and TAPEX-Acc. This might bring some bias for NLI-based metrics on faithulness-level evaluation. In the future, we will exploit a more robust automated evaluation system (Fabbri et al., 2021; Liu et al., 2022c) to comprehensively evaluate the LT2T model performances from different perspectives.

Moreover, we applied the SASP model (Ou and Liu, 2022) to convert statements into logic forms (Section 3.2). Some converted logic forms may be inconsistent with the original statement. We believe that future work could incorporate the Logic2Text (Chen et al., 2020d) dataset into training data to further improve the LoFT performance.

Ethical Consideration

We used the LogicNLG (Chen et al., 2020a) dataset for training and inference. LogicNLG is publicly available under MIT license[1] and widely used in NLP research and industry.

References

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, Online. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020b. Tabfact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020c. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *Findings of EMNLP 2020*.

Wenqing Chen, Jidong Tian, Yitian Li, Hao He, and Yaohui Jin. 2021. De-confounded variational encoder-decoder for logical table-to-text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5532–5542, Online. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Hanwen Zha, Xiyou Zhou, Yunkai Zhang, Sairam Sundaresan, and William Yang Wang. 2020d. Logic2text: High-fidelity natural language generation from logical forms.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL 2016*.

Ao Liu, Haoyu Dong, Naoaki Okazaki, Shi Han, and Dongmei Zhang. 2022a. PLOG: Table-to-logic pretraining for logical table-to-text generation. In *EMNLP 2022*.

Ao Liu, Congjian Luo, and Naoaki Okazaki. 2021. Improving logical-level natural language generation with topic-conditioned data augmentation and logical form generation. *arXiv preprint arXiv:2112.06240*.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022b. TAPEX: Table pre-training via learning a neural SQL executor. In *International Conference on Learning Representations*.

Yixin Liu, Alexander R. Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, and Dragomir Radev. 2022c. Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation.

---

[1] `https://opensource.org/licenses/MIT`

Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. 2021. Scigen: a dataset for reasoning-aware text generation from scientific tables. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Kai Nakamura, Sharon Levy, Yi-Lin Tuan, Wenhu Chen, and William Yang Wang. 2022. HybriDialogue: An information-seeking dialogue dataset grounded on tabular and textual data. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 481–492, Dublin, Ireland. Association for Computational Linguistics.

Linyong Nan, Lorenzo Jaime Flores, Yilun Zhao, Yixin Liu, Luke Benson, Weijin Zou, and Dragomir Radev. 2022. R2D2: Robust data-to-text with replacement detection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6903–6917, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Suixin Ou and Yongmei Liu. 2022. Learning to generate programs for table fact verification via structure-aware semantic parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7624–7638, Dublin, Ireland. Association for Computational Linguistics.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Yotam Perlitz, Liat Ein-Dot, Dafna Sheinwald, Noam Slonim, and Michal Shmueli-Scheuer. 2022. Diversity enhanced table-to-text generation via type control. *arXiv preprint arXiv:2205.10938*.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with entity modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2023–2035, Florence, Italy. Association for Computational Linguistics.

Chang Shu, Yusen Zhang, Xiangyu Dong, Peng Shi, Tao Yu, and Rui Zhang. 2021. Logic-consistency text generation from semantic parses. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4414–4426, Online. Association for Computational Linguistics.

Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. Plan-then-generate: Controlled data-to-text generation via planning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 895–909, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Xiaoyu Yang, Feng Nie, Yufei Feng, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Program enhanced fact verification with verbalization and graph attention network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7810–7825, Online. Association for Computational Linguistics.

Xiaoyu Yang and Xiaodan Zhu. 2021. Exploring decomposition for table-based fact verification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1045–1052, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022a. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6588–6600, Dublin, Ireland. Association for Computational Linguistics.

Yilun Zhao, Linyong Nan, Zhenting Qi, Rui Zhang, and Dragomir Radev. 2022b. ReasTAP: Injecting table reasoning skills during pre-training via synthetic reasoning examples. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9006–9018, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research; Development in Information Retrieval*, SIGIR '18, page 1097–1100, New York, NY, USA. Association for Computing Machinery.

A Appendix

Reasoning Op	Function Category	Name	Arguments	Output	Description
Unique	UNIQUE	only	view	bool	returns whether there is exactly one row in the view
Aggregation	AGGREGATION	avg/sum	view, header, string	number	returns the average/sum of the values under the header column
Count	COUNT	count	view	number	returns the number of rows in the view
Ordinal	ORD_ARG	nth_argmax/nth_argmin	view, header string	view	returns the row with the n-th max/min value in header column
	ORDINAL	nth_max/nth_min	view, header string	number	returns the n-th max/n-th min of the values under the header column
	SUPER_ARG	argmax/argmin	view, header string	view	returns the row with the max/min value in header column
Comparative	COMPARE	eq/not_eq	object, object	bool	returns if the two arguments are equal
		round_eq	object, object	bool	returns if the two arguments are roughly equal under certain tolerance
		greater/less	object, object	bool	returns if 1st argument is greater/less than 2nd argument
		diff	object, object	object	returns the difference between two arguments
Majority	MAJORITY	all_eq/not_eq	view, header string, object	bool	returns whether all the values under the header column are equal/not equal to 3rd argument
		all_greater/less	view, header string, object	bool	returns whether all the values under the header column are greater/less than 3rd argument
		all_greater_eq/less_eq	view, header string, object	bool	returns whether all the values under the header column are greater/less or equal to 3rd argument
		most_eq/not_eq	view, header string, object	bool	returns whether most of the values under the header column are equal/not equal to 3rd argument
		most_greater/less	view, header string, object	bool	returns whether most of the values under the header column are greater/less than 3rd argument
		most_greater_eq/less_eq	view, header string, object	bool	returns whether most of the values under the header column are greater/less or equal to 3rd argument
Conjunction	FILTER	filter_eq/not_eq	view, header string, object	view	returns the subview whose values under the header column is equal/not equal to 3rd argument
		filter_greater/less	view, header string, object	view	returns the subview whose values under the header column is greater/less than 3rd argument
		filter_greater_eq /less_eq	view, header string, object	view	returns the subview whose values under the header column is greater/less or equal than 3rd argument
	OTHER	filter_all	view, header string	view	returns the view itself for the case of describing the whole table
Other	OTHER	hop	view, header string	object	returns the value under the header column of the row
	OTHER	and	bool, bool	bool	returns the boolean operation result of two arguments

Table 4: A complete list of function definitions for the logic forms (Similar as Chen et al. (2020d)).

PROMPTDA : Label-guided Data Augmentation for Prompt-based Few Shot Learners

Canyu Chen
Illinois Institute of Technology
cchen151@hawk.iit.edu

Kai Shu
Illinois Institute of Technology
kshu@iit.edu

Abstract

Recent advances in large pre-trained language models (PLMs) lead to impressive gains on natural language understanding (NLU) tasks with task-specific fine-tuning. However, directly fine-tuning PLMs heavily relies on sufficient labeled training instances, which are usually hard to obtain. Prompt-based tuning on PLMs has shown to be powerful for various downstream few-shot tasks. Existing works studying prompt-based tuning for few-shot NLU tasks mainly focus on deriving proper label words with a verbalizer or generating prompt templates to elicit semantics from PLMs. In addition, conventional data augmentation strategies such as synonym substitution are also widely adopted in low-resource scenarios. However, the improvements they bring to prompt-based few-shot learning have been demonstrated to be marginal. Thus, an important research question arises as follows: *how to design effective data augmentation methods for prompt-based few-shot tuning?* To this end, considering the label semantics are essential in prompt-based tuning, we propose a novel *label-guided data augmentation* framework **PROMPTDA**, which exploits the enriched label semantic information for data augmentation. Extensive experiment results on few-shot text classification tasks show that our proposed framework achieves superior performances by effectively leveraging label semantics and data augmentation for natural language understanding.

1 Introduction

Pre-trained language models (PLMs) have shown promising performances in various applications such as text classification (Yang et al., 2019), document summarization (Zhang et al., 2020a), question answering (Mirzaee et al., 2021). The recent advancement of *prompt-based tuning* has shown a significant improvement over normal fine-tuning on various few-shot tasks (Brown et al., 2020). Typically, the prompt-based tuning paradigm transforms a NLU task into a masked language modeling (MLM) problem. For example, in sentiment analysis, an original sentence "`nice movie to watch.`" can be augmented with a *template* "`It is [MASK]`" as the input x. Each class (e.g., POSITIVE) is represented by a *label word* (e.g., `good`) selected by a *verbalizer* from the vocabulary (Schick and Schütze, 2021). The prediction of the class POSITIVE is based on the probability of the `[MASK]` being filled with the token `good`.

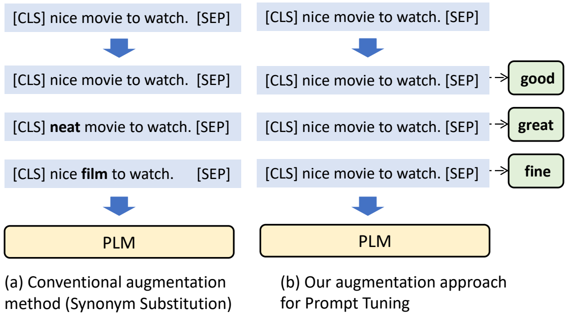


Figure 1: The basic comparison of conventional data augmentation methods and our proposed augmentation framework **PROMPTDA**. `{good, great, fine}` are the label words for prompt tuning. Conventional DA constructs ***instances*** for augmentation. But PROMPTDA conducts ***instance-label pairs*** for augmentation.

In addition, conventional data augmentation (DA) methods such as synonym substitution are also widely applied when the training data is limited (Chen et al., 2021). However, it has been shown in previous works that they can only bring marginal improvements for prompt-based few-shot learning (Zhou et al., 2021). We argue that one of the reasons could be that these DA methods mainly focus on transforming the *instances* while not incorporating the *label semantics*, which have great potential to improve the performances of few-shot tasks (Luo et al., 2021) and are essential for prompt-based few-shot learners (Liu et al., 2021). Therefore, we focus on a new problem of designing augmentation strategies for the prompt tuning paradigm and explore fusing label semantics into

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 562–574
May 2-6, 2023 ©2023 Association for Computational Linguistics

augmentation for prompt-based few-shot learners.

Specifically, different from most prompt-based tuning methods that adopt an *one-to-one verbalizer* (Schick and Schütze, 2021; Gao et al., 2021), we propose to incorporate the rich label semantic information contained in the label words derived from an *one-to-multiple verbalizer* into a new data augmentation paradigm. As shown in Figure 1, compared with previous data augmentation methods that mainly focus on constructing more ***instances***, our method PROMPTDA proposes to construct ***instance-label pairs*** for augmentation, which opens a new dimension for conducting augmentation. For example, with the one-to-multiple verbalizer mapping from the class POSITIVE to a set of label words {`good, great, fine`}, we aim to generate a set of synthetic data points $\{(x, \texttt{good}), (x, \texttt{great}), (x, \texttt{fine})\}$ based on the original instance x and leverage them to enhance the performances of the prompt-based few-shot learners. Furthermore, extensive experiment results in section § 5.5 also show that our proposed PROMPTDA can be regarded *orthogonal* to the conventional DA methods (e.g., synonym substitution) to some extent. Thus, PROMPTDA can complement with conventional augmentation approaches to further improve the performances.

To this end, we propose a new label-guided data augmentation framework for prompt-based few-shot learners named PROMPTDA, which contains three coherent modules including *Label Augmentation*, *Augmented Prompt-based Tuning*, and *Prediction Transformation*. First, we utilize a PLM to automatically search for an one-to-multiple verbalizer on a specific training set and derive a set of semantically similar tokens for each class as the label words. Second, in the training stage, we construct the instance-label pairs from the original data with regards to each label word for augmentation in prompt tuning. Third, in the inference stage, we utilize the trained language model to predict the label by aggregating the probability scores on the derived label words.

The contributions of this paper are summarized as follows: (1) we study a new problem of designing data augmentation strategies for prompt-based few-shot learners; (2) we propose a novel label-guided data augmentation framework PROMPTDA that exploits the rich label semantic information of one-to-multiple verbalizer for improving prompt tuning; (3) we conduct extensive experiments on real-world few-shot classification tasks and demonstrate the effectiveness of the proposed framework.

2 Related Work

Prompt-based Tuning has attracted increasing attention recently for various natural language processing tasks including text classification (Gao et al., 2021), question answering (Jiang et al., 2020), language generation (Li and Liang, 2021), etc. The prompt-based learning framework has shown promising performances especially in zero shot or few shot classification tasks when limited or no labels are available (Liu et al., 2021). For example, Gao *et al.* propose a prompt-based fine-tuning framework that automatically generates prompt templates and incorporates demonstrations to improve few-shot classification performances (Gao et al., 2021). Shin *et al.* proposes the AutoPrompt method to automatically generate prompts and verbalizers for eliciting the knowledge from language models (Shin et al., 2020). Other works on improving prompt-based model performances also mainly focus on constructing various types of prompt templates and verbalizers (Liu et al., 2021).

Few-shot Text Classification aims to build text classification model when few labeled data is available. Existing works mainly follow the following categories. First, semi-supervised learning where unlabeled data, alongside usually a small amount of labeled data, is used for learning (Mukherjee and Awadallah, 2020; Lee et al., 2021). For example, Subhabrata *et al.* propose to jointly learn from a small set of labeled data and a large amount of unlabeled data with uncertainty using self-training (Mukherjee and Awadallah, 2020). Second, meta-learning frameworks such as metric-based (Sui et al., 2020) and optimization-based approaches (Bansal et al., 2019). Third, weakly supervised learning to derive weak labels (Shu et al., 2020; Meng et al., 2020) in addition to the limited clean labels to improve text classification. Other approaches include transfer learning via learning to adapt transferable information from the source domain to the target domain (Gupta et al., 2020), or leveraging auxiliary tasks to improve the target tasks (Xia et al., 2021; Yin, 2020).

Data Augmentation is to construct synthetic data from an available dataset to enlarge the data size, which can help supervised training with enriched training data (Chen et al., 2021; Guo, 2020;

Shu et al., 2018), or self-supervised learning for constructing samples in pretext tasks (Zhang et al., 2017; Yoon et al., 2020), etc. Data augmentation techniques for natural language generally fall into data space and feature space (Bayer et al., 2021). In the data space, augmentation methods transform the data in character-level, word-level, phrase-level or document-level. In the feature space, representations in the latent space are manipulated by adding noise or interpolation (Schwartz et al., 2018; Verma et al., 2019). However, conventional data augmentation methods bring marginal improvements under prompt tuning paradigm (Zhou et al., 2021). It is under exploring about how to design effective data augmentation methods for prompt-based few-shot scenarios. Therefore, we propose a novel label-guided data augmentation mechanism in prompt-based tuning for few shot tasks.

3 Problem Definition

The goal of few-shot classification task is to learn a classifier to predict the label of unseen instances with limited labeled samples during the training. Following the widely-used few-shot setting (Gao et al., 2021; Liu et al., 2021), we assume that a large pre-trained language model (e.g., BERT) $\mathcal{M}$ can be utilized to fine-tune on a downstream task with the dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X}$ denotes the instances and $\mathcal{Y}$ indicates the corresponding labels. For each task, the number of training instances for each class is K, which is usually small (e.g., 8 or 16). The goal is to design a prompt learning strategy that generalizes well on unseen samples in the test set $\mathcal{D}_{\text{test}}$ with few labeled training data in $\mathcal{D}_{\text{train}}$. To ensure a fair parameter setting, we assume that a validation set $\mathcal{D}_{val}$ is available, and $|\mathcal{D}_{\text{val}}| = |\mathcal{D}_{\text{train}}|$. The test set $\mathcal{D}_{\text{test}}$ is the same as the full-data training setting.

4 Label-guided Data Augmentation for Prompt-based Tuning

In this section, we detail the proposed framework PROMPTDA, which is illustrated in Figure 2. It mainly consists of three modules: (1) a Label Augmentation module to derive multiple label words for each class to enrich the label space; (2) an Augmented Prompt-based Tuning module for augmenting the training data guided by label words; (3) a Prediction Transformation module to transform the prediction from the label words to original classes.

4.1 Label Augmentation

Due to the limited available labels in few-shot learning, recent works are generating label words to help prediction (Schick and Schütze, 2021; Gao et al., 2021). The goal is to extend the label space by incorporating the rich semantics in the vocabulary. While existing works mainly focus on selecting one label word for each class manually or automatically in prompt-tuning, the resultant label words often have a large variance and the semantics in other candidate label words are ignored. Therefore, we explore automatically searching for multiple label words for each class to better enrich the label space. Let $\mathcal{F} : \mathcal{Y} \to \mathcal{V}_{\mathcal{Y}}$ denote the one-to-multiple verbalizer that maps each label category $y \in \mathcal{Y}$ to a set of label words $\mathcal{V}_y = \{v_y^1, v_y^2, ..., v_y^{k_y}\} \subset \mathcal{V}$, where $k_y = |\mathcal{V}_y|$ denotes the number of selected label words for each class.

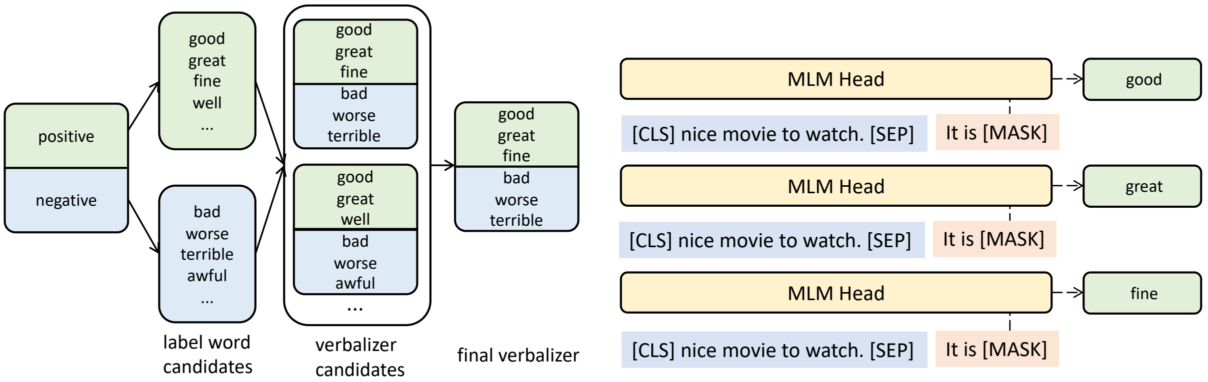
Firstly, we aim to search for a candidate set of label word $\tilde{\mathcal{V}}_y \subset \mathcal{V}$ that is semantically similar to each class $y \in \mathcal{Y}$. Let $\mathcal{D}^y_{\text{train}}$ denote the subset of training data with the class y. $\mathcal{T}(x)$ denotes the input x with a fixed template $\mathcal{T}$. `Po([mask])` denotes the position of `[mask]` in the input x. We propose to select the Top-m label words from vocabulary as $\tilde{\mathcal{V}}_y$ based on the conditional likelihood over $\mathcal{D}^y_{\text{train}}$ for each class y:

$$\tilde{\mathcal{V}}_y = \underset{v \in \mathcal{V}}{\text{Top-}m} \left\{ \sum_{(x,y) \in \mathcal{D}^y_{\text{train}}} \Pr(v, \mathcal{T}(x)) \right\} \tag{1}$$

where $\Pr(v, \mathcal{T}(x))$ denotes the corresponding probability score of each token in the vocabulary filling in `Po([mask])` in PLM inference as:

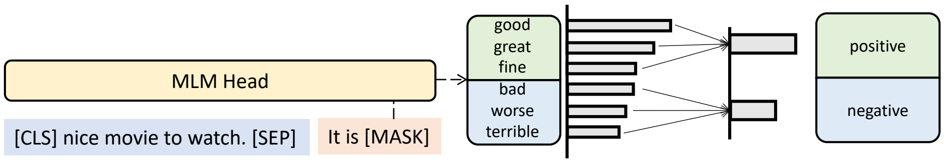
$$\Pr(v, \mathcal{T}(x)) = \Pr(\texttt{Po([mask])} = v \mid \mathcal{T}(x)) \tag{2}$$

Secondly, we construct a verbalizer candidate set F for the whole dataset. It is a combinatorial problem to select k_y label words from $\tilde{\mathcal{V}}_y$ to construct $\mathcal{V}_y$ for each class y. The number of possible candidates of $\mathcal{V}_y$ is $\binom{|\tilde{\mathcal{V}}_y|}{k_y}$. Then the element number of the verbalizer candidate set F is $|F| = \binom{|\tilde{\mathcal{V}}_y|}{k_y}^{|\mathcal{Y}|}$. We utilize each one-to-multiple verbalizer candidate in F to infer and calculate the prediction accuracy on $\mathcal{D}_{\text{train}}$ via the same *prediction transformation* method in section § 4.3. Then we select the Top-n candidates from F based on the prediction accuracy. If there exist multiple candidates with the same accuracy score, we randomly select one as the



(a) Label Augmentation

(b) Augmented Prompt-based Tuning



(c) Prediction Transformation

Figure 2: The proposed **PROMPTDA** for few-shot learning (with sentiment classification task as an example): (a) **Label Augmentation**: deriving multiple label words for each class to enrich the label semantic space; (b) **Augmented Prompt-based Tuning**: training with the augmented instance-label pairs via masked language modeling; (c) **Prediction Transformation**: aggregating the probability scores on the derived label words for the final prediction.

final one-to-multiple verbalizer. Otherwise, we select the verbalizer candidate with highest accuracy score. Note that m and n are both hyperparameters picked by pilot study on the specific datasets.

4.2 Augmented Prompt-based Tuning

To enrich the training data for the few-shot text classification task, it is natural to utilize data augmentation methods such as token-level or sentence-level augmentation for fine-tuning (Chen et al., 2021). Most of the existing data augmentation methods focus on enlarging training data conditioned on the original label space. Orthogonal to previous augmentation methods, our method incorporates label semantic information into prompt-tuning via augmenting sample-label pairs rather than only augmenting samples. For $(x, y) \in \mathcal{D}_{\text{train}}$, we have obtained the corresponding label word set $\mathcal{V}_y = \{v_y^1, v_y^2, ..., v_y^{k_y}\}$. Then we can include $\{(x, v_y^1), (x, v_y^2), ..., (x, v_y^{k_y})\}$ for augmentation. Let $\tilde{\mathcal{D}}_{\text{train}}$ denote the augmented dataset. The resultant dataset can be denoted as follows:

$$\tilde{\mathcal{D}}_{\text{train}} = \cup_{(x,y)\in\mathcal{D}_{\text{train}}}\{(x, v_y^1), (x, v_y^2), ..., (x, v_y^{k_y})\} \tag{3}$$

In the training process, we follow the MLM training paradigm and minimize the negative log-likelihood on the whole augmented training set $\tilde{\mathcal{D}}_{\text{train}}$. The optimization objective is:

$$\mathcal{L} = \sum_{(x,v)\in\tilde{\mathcal{D}}_{\text{train}}} - \log \Pr(v \mid x) \tag{4}$$

For $(x, v) \in \tilde{\mathcal{D}}_{\text{train}}$, the conditional probability of filling the position of `[mask]` with v is:

$$\begin{aligned} \Pr(v \mid x) &= \Pr(\texttt{Po([mask])} = v \mid x) \\ &= \frac{\exp\left(\mathbf{w}_v \cdot \mathbf{h}_{\texttt{[MASK]}}\right)}{\sum_{v' \in \mathcal{V}} \exp\left(\mathbf{w}_{v'} \cdot \mathbf{h}_{\texttt{[MASK]}}\right)} \end{aligned} \tag{5}$$

where $\mathbf{w}_v$ denotes the pre-softmax output vector for each token v in the vocabulary, and $\mathbf{h}_{\texttt{[MASK]}}$ denotes the corresponding hidden state of the `[MASK]` position. Note that we completely reuse the PLM and do not introduce new parameters in the training process, which is important for prompt-based tuning to be effective in few-shot scenarios.

4.3 Prediction Transformation

We have demonstrated the process of training the MLM classifier head with the augmented data in the prompt-based tuning paradigm. Next, we describe how to perform the inference for the target class. Let h denote the function that transforms the probability scores on the label word set $\mathcal{V}_y = \{v_y^1, v_y^2, ..., v_y^{k_y}\}$ into the probability score of each class y. Since the label word with the highest probability score in set $\mathcal{V}_y$ can represent the class y, we use $h = max()$ to calculate the final probability score of each class. Then the probability score of each class y can be calculated as:

$$\Pr(y \mid x) = h(\mathrm{P}(v_y^1, x), \mathrm{P}(v_y^2, x), ..., \mathrm{P}(v_y^{k_y}, x)) \tag{6}$$

where for (x, v_y^i) that satisfies $(x, v_y^i) \in \tilde{\mathcal{D}}_{\text{train}}$ and $v_y^i \in \mathcal{V}_y, (i = 1, 2, ..., k_y)$, $\mathrm{P}(v_y^i, x)$ is denoted as the conditional probability of filling the position of `[mask]` with v_y^i:

$$\mathrm{P}(v_y^i, x) = \Pr(\texttt{Po([mask])} = v_y^i \mid x) \tag{7}$$

After we obtain the probability score over each class, the final predicted class $\hat{y}$ is calculated as:

$$\hat{y} = \mathrm{argmax}_{y \in \mathcal{Y}} \Pr(y \mid x) \tag{8}$$

5 Experiments

In this section, we present the experiments to evaluate the effectiveness of the proposed PROMPTDA. Specifically, we aim to answer the following research questions:

- **RQ1** Can PROMPTDA improve the performance of few-shot prompt-based tuning?
- **RQ2** Can the proposed Label Augmentation strategy help the target label prediction?
- **RQ3** Can the PROMPTDA make the prompt-based tuning method more stable?

5.1 Experimental Settings

Datasets. We evaluate the proposed framework on few shot text classification datasets from the widely-used NLU benchmark GLUE (Wang et al., 2018) including SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019) and other common datasets including MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), Subj (Pang and Lee, 2004), MPQA (Wiebe et al., 2005), SST-5 (Socher et al., 2013). These datasets covers different tasks such as sentiment analysis, topic classification and opinion classification from various domains including movie reviews, news pieces, etc. The statistics of the datasets are shown in Table 4 in Appendix.

Baselines. We compare the proposed approach with various representative methods including **Majority**, **Fine-Tuning**, **GPT-3** (Brown et al., 2020), **EFL** (Wang et al., 2021), **LM-BFF** (Gao et al., 2021) and **Prompt Tuning**. More details are described in the Appendix A.3.

Evaluation setting. Evaluation is critical in few-shot scenarios because small changes of the training set can result in a large variance in the performance of the test set. Following the few-shot setting in (Perez et al., 2021), (Zhang et al., 2020b), (Gu et al., 2021) and (Gao et al., 2021), we randomly select K-shot samples from original dataset for each class to construct the training set $\mathcal{D}_{\text{train}}$ and select another K-shot samples to construct the development set $\mathcal{D}_{\text{val}}$. For enhancing the stability of evaluation, we utilize the whole test set of original dataset as out test set $\mathcal{D}_{\text{test}}$ and change the random seed of sampling $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$ for five times. We select RoBERTa-large as our backbone model to make fair comparison with baseline LM-BFF.

5.2 Experimental Results

In this section, we present our main results, and address the aforementioned research questions pertaining to our PROMPTDA approach.

In addition to comparing with baselines such as Majority, normal fine tuning and prompt-based method GPT-3, EFL, LM-BFF, we conduct more experiments to verify the effectiveness of our proposed method PROMPTDA as a plug-in module. Because different template choices can result in a large variance of performance (Gao et al., 2021), we design two groups of experiments, namely template-free and template-augmented, to investigate whether or not our method can improve over standard prompt-based tuning method regardless of template design. For the template-augmented group of experiments, we manually choose "`It is [MASK]`" as the template, following (Wang et al., 2021). For the template-free group of experiments, we only append "`[MASK]`" in the input. We report the results of PROMPTDA in Table 1 when the size of data augmentation is ×3 (i.e., $k_y = 3$). We also consider two scenarios where the label words are derived manually or with our automatic label augmentation mechanism. We choose 8 samples ($K = 8$) per class as the few-shot setting of our main experiments. For fair comparison, we choose the same random seed of training set sampling as LM-BFF. We train for 10 epochs on each dataset following (Wang et al., 2021). We report the average performance and standard variance of our results over five runs of sampling on each dataset. The main results can be seen in Table 1.

Performance analysis We analyze the performance from three perspectives to answer the afore-

Method	**SST-2** (Acc)	**MR** (Acc)	**CR** (Acc)	**Subj** (Acc)	**CoLA** (Acc)	**MPQA** (Acc)	**SST-5** (Acc)
Majority (full)	50.9	50.0	50.0	50.0	69.1	50.0	23.1
Fine-Tuning (full)	95.0	90.8	89.4	97.0	86.2	89.4	58.7
Fine-Tuning	60.5 (3.1)	60.3 (7.5)	61.9 (5.1)	78.3 (8.2)	51.1 (8.4)	59.0 (3.4)	31.5 (7.5)
GPT-3 (Brown et al., 2020)	82.9 (3.4)	81.2 (2.5)	86.8 (1.5)	53.2 (1.5)	52.1 (6.2)	62.9 (3.5)	31.5 (4.3)
EFL (Wang et al., 2021)	67.5 (8.5)	69.8 (7.5)	75.3 (4.8)	78.9 (7.8)	54.3 (8.9)	68.4 (5.7)	35.2 (6.3)
LM-BFF (Gao et al., 2021)	89.1 (4.1)	83.6 (3.4)	87.8 (4.3)	81.6 (6.1)	53.5 (4.5)	73.9 (8.9)	41.2 (3.1)
Prompt Tuning‡	85.5 (5.2)	83.0 (3.7)	86.5 (3.0)	81.8 (5.6)	50.5 (10.3)	71.5 (9.8)	37.5 (5.5)
PT + PromptDA(m.)‡	87.3 (4.4)	82.5 (1.4)	88.1 (2.7)	81.3 (4.9)	51.2 (7.5)	72.9 (9.1)	39.4 (4.3)
PT + PromptDA(au.)‡	87.6 (4.1)	83.1 (3.1)	87.8 (1.2)	83.4 (2.5)	52.8 (8.1)	74.5 (7.8)	41.8 (3.9)
Prompt Tuning†	85.8 (5.8)	79.3 (8.2)	86.1 (8.0)	81.2 (5.7)	52.7 (6.6)	75.1 (13.7)	38.4 (4.7)
PT + PromptDA(m.)†	88.9 (3.9)	**83.8 (1.9)**	84.9 (5.7)	82.4 (9.9)	51.3 (15.5)	78.1 (8.9)	42.7 (7.1)
PT + PromptDA(au.)†	**89.5 (2.9)**	83.7 (2.6)	**88.3 (4.1)**	**86.8 (3.1)**	**55.9 (7.1)**	**78.4 (9.2)**	**43.3 (1.6)**

Table 1: The main results using RoBERTa-large on representative NLU tasks. All the results are evaluated on full test sets and averaged over 5 runs. $K = 8$: 8 samples per class for all the experiments; †: template augmented; ‡: template-free; (m.): manual label augmentation; (au.): automatic label augmentation; PT: Prompt Tuning.

mentioned research questions.

To answer **RQ1**, we compare the proposed method with existing baselines. First, in general, we can observe that the standard prompt-based tuning method with PromptDA consistently perform better than or is comparable with baselines such as GPT-3, EFL, LM-BFF and normal fine tuning (results of "PT + PromptDA(au.)†" in Table 1). Compared with LM-BFF, standard prompt-based tuning with PromptDA performs better on all the datasets. For example, our method achieves a 6% gain over LM-BFF on Subj and MPQA datasets. Compared with normal fine tuning, our method achieves superior performance by a large margin. For example, our method obtains a 47.9% improvement over normal fine tuning on SST-2 dataset.

Second, we can see that PromptDA can improve over standard prompt-based tuning method regardless of template design (results of "PT + PromptDA(au.)†" and "PT + PromptDA(au.)‡" in Table 1). Compared with standard prompt tuning, PromptDA can achieve a better performance over the seven datasets regardless of being template-free or template-augmented, which suggests that PromptDA has no relation with template design and can be used as a plug-in module for improving performance of prompt tuning.

Third, PromptDA generally improves over standard prompt-based tuning method regardless of automatic label word selection or manual label word selection (results of "PT + PromptDA(au.)†" and "PT + PromptDA(m.)†" in Table 1). Compared with standard prompt tuning, prompt tuning with automatic label word selection achieves improvements over all the datasets. For prompt tuning with manual label word selection, it also has performance gains over SST-2, MR, Subj, MPQA and SST-5 datasets.

To answer **RQ2**, we perform an ablation study of PromptDA, and compare the results of "PT + PromptDA(au.)†" and "PT + PromptDA(m.)†" in Table 1. We can see that regardless of template design, our proposed automatically searched label words generally perform better than manually searched label words. For example, "PT + PromptDA(au.)†" achieves a 5.3% improvement over "PT + PromptDA(m.)†" on Subj dataset.

We analyze the reason from three perspectives. First, we hypothesize that human bias may hinder selecting optimal label words and our proposed automatic method relies on language model itself and can minimize human bias. Second, it may be easier for human to select similar words as label words for sentiment-related datasets with the label name "`positive, negative`", but it is hard to select semantically similar words as label words for tasks in other domains. For example, it is hard to manually identify semantically similar words as label words for Subj dataset with the label name "`subjective, objective`", which illustrates the necessity of our proposed automatic method for searching label words. Third, our proposed Label Augmentation method can search for different label words on different training data, but it is hard for the manual label word selection method to adapt to different specific datasets.

To answer **RQ3**, we analyze the stability of performances of PromptDA. In general, we observe that PromptDA reduces the variance of prompt-tuning. (Standard variance of "PT +

SST-2	label name	`positive	negative`
	label words (m.)	`positive, great, good	negative terrible bad`
	label words (au.)	`wonderful brilliant fantastic	terrible done disappointing`
Subj	label name	`objective	subjective`
	label words (m.)	`good neutral fair	bad emotional personal`
	label words (au.)	`disturbing terrifying key	bad not nonsense`
SST-5	label name	`very positive	positive	neutral	negative	very negative`
	label words (m.)	`great perfect excellent	good, pretty, wonderful	neutral normal fine	bad worse not	terrible awful ridiculous`
	label words (au.)	`great brilliant fantastic	extraordinary remarkable fascinating	enough terrible funny	awful bad worse	boring done unnecessary`

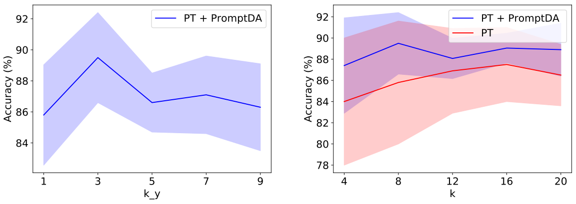
Table 2: An illustration of the label words searched automatically or manually on SST-2, Subj and SST-5 datasets.

PROMPTDA(au.)†” in Table 1). The uncertainty of prompt-based tuning methods mainly comes from different distribution of small training set, different designs of the template and various selections of label words for each class. Compared with LM-BFF and normal fine tuning methods, our method generally reduces the variance of prediction. For example, the standard variance of prediction over five runs for “PT + PROMPTDA(au.)†” has decreased around 48.4% on Subj compared to LM-BFF and has decreased 78.7% on SST-5 compared with normal fine tuning. Compared with standard prompt-based tuning method, PROMPTDA can improve the stability of tuning on most of the datasets.

5.3 Analysis of Label Word Selection

Without loss of generosity, we take the dataset SST-2, Subj and SST-5 for example to analyze the quality of Label Augmentation (the label word results are shown in Table 2 and the complete label word results over five runs on SST-2, CR, MR, Subj, CoLA, MPQA, SST-5 datasets are shown in Appendix Table 6). *The goal of label augmentation is to find semantically similar words to enrich the label space.* With regards to the manual way, we find the synonyms of label name from dictionary as the label words and ensure these words are in the vocabulary. And we select the same label words for different seeds. With regards to our proposed automatic method, we only rely on the training set and language model (e.g., RoBERTa-large) to find the semantically similar words from vocabulary and do not rely on label name itself.

The Table 2 shows the label words automatically or manually searched on datasets SST-2, Subj and SST-5 respectively. For sentiment related datasets such as SST-2 with the label name {`positive/negative`}, the label words automatically searched are literally similar to the manually selected label words, which probably means the way language models (e.g., RoBERTa-large) reasons about what are similar words is close to the human way in sentiment domain. Nonetheless, for other datasets such as Subj with the label name {`objective/subjective`}, it is interesting to observe that the label words automatically searched are not literally similar to label name or manually selected label words, which may infer that the way language models (e.g., RoBERTa-large) reason about what are similar words is different from the human way in other domains. We argue that *how to define word similarity in label semantic space* needs more research in the future. For dataset such as SST-5, we can see that it is much harder to select appropriate label words when the number of classes is larger, which also verifies the importance of automatic label word selection.



(a) # size of augmentation (b) # samples per class

Figure 3: The impact analysis of the size of label words and training samples per class on SST-2 dataset.

5.4 Assessment of Data Augmentation

We analyze data augmentation from two perspectives including the size of data augmentation and the size of training set.

The size of data augmentation We choose to study the effect of the size of PROMPTDA on template-augmented prompt-based tuning on SST-2 dataset. The results over five runs for 10 epochs are presented in Figure 2 (a). We can observe that PROMPTDA can generally improve over prompt-based tuning regardless of the size of augmentation. However, larger augmentation may result in more

Method	SST-2 (Acc)	MR (Acc)	CR (Acc)	Subj (Acc)	SST-5 (Acc)
PT	85.8 (5.8)	79.3 (8.2)	86.1 (8.0)	81.2 (5.7)	38.4 (4.7)
PT with Conventional DA	89.2 (1.3)	80.3 (3.1)	86.5 (4.5)	82.3 (8.0)	39.1 (4.5)
PT with PROMPTDA	89.5 (2.9)	83.7 (2.6)	88.3 (4.1)	86.8 (3.1)	43.3 (1.6)
PT with PROMPTDA & Conventional DA	**89.7 (1.6)**	**84.8 (1.5)**	**89.2 (1.3)**	**87.0 (3.1)**	**44.7 (1.1)**

Table 3: The main results of evaluating Prompt Tuning (PT) with PROMPTDA and conventional DA method on NLU tasks. All the results are evaluated on full dev sets and averaged across 5 different training sets. $K = 8$: 8 samples per class for the experiments. Conventional DA refers to *synonym substitution*.

unstable final prediction. We analyze the reason from two perspectives. First, larger data augmentation may contain more label noise. Since we utilize an one-to-multiple verbalizer to guide data augmentation, the size of data augmentation is equal to the number of label words per class, which may cause more noisy label words. Unsuitable label word selections may worsen the performance and increase the variance of final prediction. Second, more label words per class may cause the model harder to converge on small training sets. When training for the same epochs, prompt tuning with more label words per class may perform more unstable.

The size of the training set We study the effect of the size of training set on template-augmented prompt-based tuning with and without PROMPTDA. The size of data augmentation is ×3. The results over five runs for 10 epochs are presented in Figure 2 (b). We have several observations from the results. First, our method PROMPTDA consistently improves over standard prompt-based tuning regardless of the size of training sets. Second, our proposed method generally decreases the variance of prompt-based tuning. Third, the improvement space of PROMPTDA over prompt-based tuning decreases as the number of samples per class increases.

5.5 Combination with Conventional DA

Although conventional data augmentation methods are still effective when training data is limited (Chen et al., 2021), previous works verified that they can bring marginal improvement for the prompt tuning paradigm (Zhou et al., 2021). It is worth exploring whether or not PROMPTDA can complement with conventional DA for further enhancing the performance of prompt tuning.

We follow the same setting as the main experiments and test conventional DA, PROMPTDA and the combination on standard prompt-based tuning paradigm with template. With regards to conventional DA, we select *synonym substitution* method from nlpaug toolkit (Ma, 2019) and enlarge the training set by ×2. With regards to our proposed PROMPTDA, we enlarge the training set by ×3. The experiment results over five different sampling seeds for 10 epochs are shown in Table 3.

We can observe that the combination of PROMPTDA and Conventional DA method consistently outperforms only using PROMPTDA or Conventional DA method. Conventional DA methods mostly focus on exploiting the semantic information of the instance itself. Our method proposes to utilize label semantic information to guide data augmentation and does not change instances. PROMPTDA conducts the augmentation from a different perspective compared with conventional augmentation methods. Therefore, our proposed method PROMPTDA can be regarded *orthogonal* to conventional DA methods to some extent and complement with each other.

6 Conclusion and Future Work

In this paper, we study a new problem of data augmentation in prompt-based tuning for few shot learners. To leverage the label semantic information, we propose a novel label-guided data augmentation approach PROMPTDA, which can derive multiple label words and exploit the rich semantic information of the label words. We conduct extensive experiments on various datasets and demonstrate the effectiveness of PROMPTDA for few-shot learning. We also conduct detailed analysis on the effects of manual/automatic label augmentation, the size of augmentation, the size of label words, and combination with conventional DA.

There are several interesting directions for future work. First, we will extend PROMPTDA to multi-label few shot tasks and leverage multi-aspect label space. Second, we will explore prompt-based data augmentation for token-level tasks such as few-shot name entity recognition (NER). Third, we will explore prompt-based tuning to enhance the interpretability capacity for various NLP tasks.

Limitations

Our work is the first step for designing data augmentation strategies for prompt tuning paradigm. In this work, we focus on the natural language understanding (NLU) tasks. The prompt tuning paradigm is applied in various tasks including language generation, question answering, dialog system, etc. Designing augmentation strategies for prompt-based few-shot learners in more applications is under exploration.

Ethics Statement

This paper focuses on the task of few-shot natural language understanding and conducts experiments on open datasets. The implementation details are described in Appendix for reproduction.

Acknowledgements

This work is supported in part by NSF SaTC-2241068, and a Cisco Research Award

References

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2019. Learning to few-shot learn across diverse natural language classification tasks. *arXiv preprint arXiv:1911.03863*.

Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2021. A survey on data augmentation for text classification. *arXiv preprint arXiv:2107.03158*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jiaao Chen, Derek Tam, Colin Raffel, Mohit Bansal, and Diyi Yang. 2021. An empirical survey of data augmentation for limited data learning in nlp. *arXiv preprint arXiv:2106.07499*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Hongyu Guo. 2020. Nonlinear mixup: Out-of-manifold data augmentation for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4044–4051.

Aakriti Gupta, Kapil Thadani, and Neil O'Hare. 2020. Effective few-shot classification with transfer learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1061–1066.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Ju-Hyoung Lee, Sang-Ki Ko, and Yo-Sub Han. 2021. Salnet: Semi-supervised few-shot text classification with attention-based lexicon construction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13189–13197.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Qiaoyang Luo, Lingqiao Liu, Yuhao Lin, and Wei Zhang. 2021. Don't miss the labels: Label-semantic augmented meta-learner for few-shot text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2773–2782.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020. Text classification using label names only: A language model self-training approach. *arXiv preprint arXiv:2010.07245*.

Roshanak Mirzaee, Hossein Rajaby Faghihi, Qiang Ning, and Parisa Kordjmashidi. 2021. Spartqa:: A textual question answering benchmark for spatial reasoning. *arXiv preprint arXiv:2104.05832*.

Subhabrata Mukherjee and Ahmed Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. *Advances in Neural Information Processing Systems*, 33.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *arXiv preprint arXiv:2105.11447*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Rogerio Feris, Abhishek Kumar, Raja Giryes, and Alex M Bronstein. 2018. Delta-encoder: an effective sample synthesis method for few-shot object recognition. *arXiv preprint arXiv:1806.04734*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Eliciting knowledge from language models using automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

Kai Shu, Subhabrata Mukherjee, Guoqing Zheng, Ahmed Hassan Awadallah, Milad Shokouhi, and Susan Dumais. 2020. Learning with weak supervision for email intent detection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1051–1060.

Kai Shu, Suhang Wang, Thai Le, Dongwon Lee, and Huan Liu. 2018. Deep headline generation for clickbait detection. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 467–476. IEEE.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Dianbo Sui, Yubo Chen, Binjie Mao, Delai Qiu, Kang Liu, and Jun Zhao. 2020. Knowledge guided metric learning for few-shot text classification. *arXiv preprint arXiv:2004.01907*.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Mengzhou Xia, Guoqing Zheng, Subhabrata Mukherjee, Milad Shokouhi, Graham Neubig, and Ahmed Hassan Awadallah. 2021. Metaxl: Meta representation transformation for low-resource cross-lingual learning. *arXiv preprint arXiv:2104.07908*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Wenpeng Yin. 2020. Meta-learning for few-shot natural language processing: A survey. *arXiv preprint arXiv:2007.09604*.

Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. 2020. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems*, 33.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020b. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*.

Jing Zhou, Yanan Zheng, Jie Tang, Jian Li, and Zhilin Yang. 2021. Flipda: Effective and robust data augmentation for few-shot learning. *arXiv preprint arXiv:2108.06332*.

A Appendix

A.1 Implementation Details

We implemented our model and all baselines with PyTorch and run each experiment on a single NVIDIA GeForce RTX 3090 GPU. The hyperparameters are the same for all methods based on RoBERTa-large (the learning rate is 3e-6, the batch size is 4, the number of training epochs is 10). Following (Gao et al., 2021), we select the random seeds for sampling the training set and validation set as $\{13, 21, 42, 87, 100\}$.

A.2 Dataset Details

In general, we follow the experiment setting of (Gao et al., 2021). For datasets from GLUE (Wang et al., 2018) including SST-2 (Socher et al., 2013) and CoLA (Warstadt et al., 2019), we use the original development sets for testing. For datasets requiring cross-validation evaluation like MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005) and Subj (Pang and Lee, 2004), we randomly sample 2,000 instances as the testing set and remove them from the training set. For the dataset SST-5 (Socher et al., 2013), we use the official test sets. The dataset statistics are shown in Table 4.

A.3 Baseline Details

The details of the baselines are as follows:

- **Majority**: The label is predicted by taking the majority class in the training set. We run this baseline on the full-data setting.
- **Fine-Tuning**: The prediction is based on the pre-trained language model that is fine-tuned with the specific training data. We run this baseline in the full-data and few-shot setting.
- **GPT-3** (Brown et al., 2020): GPT-3 in-context tuning in the zero-shot setting. We pack the training samples into the input together and directly conduct inference.
- **EFL** (Wang et al., 2021): An entailment-based prompt tuning framework. For fair comparison, we do not pretrain the language model on MNLI task but directly tune the language model as the prompt tuning paradigm.
- **LM-BFF** (Gao et al., 2021): A prompt tuning model that automatically searches for demonstrations, templates and label words. Note that LM-BFF utilizes one-to-one verbalizer for label word selection.
- **Prompt Tuning**: The standard Prompt-based Tuning augmented by a simple template or template-free.

A.4 Comparison of RoBERTa vs BERT

We conduct experiments to investigate the impact of the backbone model. Table 5 shows the results of using BERT-large(uncased) and RoBERTa-large. The experiment setting is the same as the main experiments. We can observe that our proposed PROMPTDA improves the performance of prompt tuning regardless of the backbone model.

A.5 The Verbalizer and Template Design

For each random seed of $\{13, 21, 42, 87, 100\}$, we can construct different training sets and validation sets. Thus, the verbalizers searched automatically for each run are different, which are shown as "label words (au.)" in Table 6. The verbalizers manually designed for each run are the same, which are shown as "label words (m.)" in Table 6. We follow previous works (Gao et al., 2021; Wang et al., 2021) and design a simple template "`It is [MASK]`" for each input.

Dataset	# Classes	# Length	# Train	# Test	Type	Labels
SST-2	2	19	6,920	872	sentiment	positive, negative
MR	2	20	8,662	2,000	sentiment	positive, negative
CR	2	19	1,775	2,000	sentiment	positive, negative
Subj	2	23	8,000	2,000	subjectivity	subjective, objective
CoLA	2	8	8,551	1,042	acceptability	grammatical, not_grammatical
MPQA	2	3	8,606	2,000	opinion polarity	positive, negative
SST-5	5	18	8,544	2,210	sentiment	v. pos., positive, neutral, negative, v. neg.

Table 4: Statistics of the datasets.

BERT-large	**SST-2**	**Subj**	**SST-5**
PT	82.3 (4.6)	80.3 (6.2)	34.5 (3.8)
PT + PROMPTDA	87.1 (3.1)	82.9 (3.3)	37.5 (2.8)
RoBERTa-large	**SST-2**	**Subj**	**SST-5**
PT	85.8 (5.8)	81.2 (5.7)	38.4 (4.7)
PT + PROMPTDA	**89.5 (2.9)**	**86.8 (3.1)**	**43.3 (1.6)**

Table 5: A comparison of RoBERTa-large vs BERT-large on template-augmented prompt tuning.

SST-2	label name	positive | negative
	label word (s.)	positive | negative
	label words (m.)	good perfect fantastic | terrible awful hilarious
	label words (au.)	brilliant amazing wonderful | not awful terrible
		great perfect brilliant | terrible disappointing bad
		beautiful perfect fantastic | terrible awful hilarious
		fantastic excellent beautiful | terrible awful worse
		wonderful, brilliant, fantastic | terrible done disappointing
MR	label name	positive | negative
	label word (s.)	positive | negative
	label words (m.)	positive, great, good | negative, terrible, bad
	label words (au.)	refreshing good beautiful | not terrible disappointing
		beautiful perfect fantastic | awful disappointing horrible
		fantastic wonderful beautiful | terrible awful funny
		fantastic incredible unforgettable | terrible funny bad
		excellent refreshing amazing | terrible wrong bad
CR	label name	positive | negative
	label word (s.)	positive | negative
	label words (m.)	good perfect fantastic | terrible awful hilarious
	label words (au.)	amazing fun cool | disappointing frustrating bad
		excellent fun cheap | awful horrible terrible
		free fun cool | bad painful useless
		fantastic brilliant incredible | terrible inevitable useless
		amazing great awesome | terrible awful horrible
Subj	label name	objective | subjective
	label word (s.)	actual | individual
	label words (m.)	good neutral fair | bad emotional personal
	label words (au.)	epic life America | madness not wrong
		life history significant | right that great
		what real interesting | me good great
		fiction interesting America | wonderful great brilliant
		disturbing terrifying key | bad not nonsense
CoLA	label name	grammatical | not_grammatical
	label word (s.)	good | bad
	label words (m.)	positive correct good | negative wrong bad
	label words (au.)	it wrong correct | ridiculous not good
		different sad interesting | complicated hilarious scary
		wrong interesting important | insane sad crazy
		all good important | bad new impossible
		how amazing normal | true him me
MPQA	label name	positive | negative
	label word (s.)	good | bad
	label words (m.)	good perfect fantastic | terrible awful hilarious
	label words (au.)	possible necessary adopted | wrong bad dark
		obvious awesome fun | then difficult gone
		right fun decided | reported unfair rejected
		accepted good great | unavoidable awful bad
		different good amazing | wrong bad funny
SST-5	label name	very positive | positive | neutral | negative | very negative
	label word (s.)	extraordinary | great | enough | boring | awful
	label words (m.)	great perfect excellent | good pretty wonderful | neutral normal fine | bad worse not | terrible awful ridiculous
	label words (au.)	good excellent unforgettable | hilarious inevitable funny | different time interesting | predictable bad over | dreadful boring horrible
		magnificent unforgettable fantastic | refreshing remarkable sublime | disappointing bad hilarious | neither predictable inevitable | depressing pathetic unnecessary
		wonderful fantastic incredible | terrifying refreshing interesting | hilarious done easy | better disappointing predictable | disgusting ridiculous horrible
		magnificent excellent too | stunning unexpected refreshing | simple done interesting | boring there worse | ridiculous sad weird
		great brilliant fantastic | extraordinary remarkable fascinating | enough terrible funny | awful bad worse | boring done unnecessary

Table 6: The verbalizer design (single label word (s.) for normal prompt tuning, label words manually designed (m.) and automatically searched (au.) for prompt tuning with PromptDA) over five runs on SST-2, CR, MR, Subj, CoLA, MPQA, SST-5 datasets.

Incorporating Question Answering-Based Signals into Abstractive Summarization via Salient Span Selection

Daniel Deutsch†
Google
dandeutsch@google.com

Dan Roth
University of Pennsylvania
danroth@seas.upenn.edu

Abstract

In this work, we propose a method for incorporating question-answering (QA) signals into a summarization model. Our method identifies salient noun phrases (NPs) in the input document by automatically generating wh-questions that are answered by the NPs and automatically determining whether those questions are answered in the gold summaries. This QA-based signal is incorporated into a two-stage summarization model which first marks salient NPs in the input document using a classification model, then conditionally generates a summary. Our experiments demonstrate that the models trained using QA-based supervision generate higher-quality summaries than baseline methods of identifying salient spans on benchmark summarization datasets. Further, we show that the content of the generated summaries can be controlled based on which NPs are marked in the input document. Finally, we propose a method of augmenting the training data so the gold summaries are more consistent with the marked input spans used during training and show how this results in models which learn to better exclude unmarked document content.[1]

1 Introduction

Abstractive sequence-to-sequence summarization models have become very effective methods of easily generating summaries of input documents (Rush et al., 2015; Nallapati et al., 2016; Lewis et al., 2020).

Previous work has demonstrated that conditioning the summary generation on salient document sentences results in higher-quality summaries and more controllable summarization models (Chen and Bansal, 2018; Dou et al., 2021). Salient sentences are typically identified during training by lexical overlap with the gold summaries (Nallapati et al., 2017) and predicted during inference.


Input Document

Incumbent Goodluck Jonathan phoned former military leader Muhammadu Buhari on Tuesday to concede defeat in Nigeria's presidential elections, Buhari's party says. Jonathan acknowledged the phone call and his defeat in a written statement to his countrymen. "I thank all Nigerians once again for the great opportunity… I promised the country free and fair elections. I have kept my word…" Buhari ruled Nigeria from late 1983 until August 1985… **The 72-year-old retired major general's experience…**

Gold Summary

Incumbent President Goodluck Jonathan acknowledges defeat, says he delivered on promise of fair elections. Muhammadu Buhari 's party says Jonathan called to concede even before final results are announced. Buhari is a 72-year-old retired major general who ruled in Nigeria in the 1980s.

Figure 1: Salient spans identified by QA-based signals (shown in color) more precisely identify salient document content than those that identify salient sentences based on lexical overlap (shown in bold). Our method classifies the salient spans, marks them in the input document, and then generates a summary.

Although marking different sentences as salient allows for some controllability over the content of the summary, desired summary content cannot be specified at the sub-sentence level. Further, labeling sentences as salient via n-gram overlap does not directly take the predicate-argument structure of the text into account, which could result in a lower-quality supervision signal that misidentifies which particular instance of an n-gram is salient.

In this work, we propose to condition the summary generation on salient sub-sentence level spans which are identified by reasoning about the predicate-argument relations in the text.

We mark noun phrases (NPs) in the input document as salient if the predicate-argument relation

---

[1] `http://cogcomp.org/page/publication_view/997`

† Work done while at the University of Pennsylvania

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 575–588
May 2-6, 2023 ©2023 Association for Computational Linguistics

they participate in is present in the gold summary (§2). This idea is implemented using automatic question generation (QG) and answering (QA). For each NP, a wh-question that is answered by the NP is generated from the text. Then, the NP is marked as salient if the generated wh-question is correctly answered in the gold summary according to a learned QA model, resulting in a more precise, sub-sentence level supervision signal (see Fig. 1).

The QA-based salience signal is incorporated into a two-stage summarization model (§3). First, a phrase salience classifier is trained to identify which NPs in the document are salient. Then, the predicted salient spans are marked in the input document with special tokens and used to conditionally generate a summary of the document with a fine-tuned BART model (Lewis et al., 2020).

While we show that marking NPs as salient controls the summary content, the model often outputs extra, undesired information. To that extent, we propose a data augmentation procedure that removes sentences unsupported by any salient span and generates new training examples based on what content should be able to be generated by subsets of the salient spans (§4).

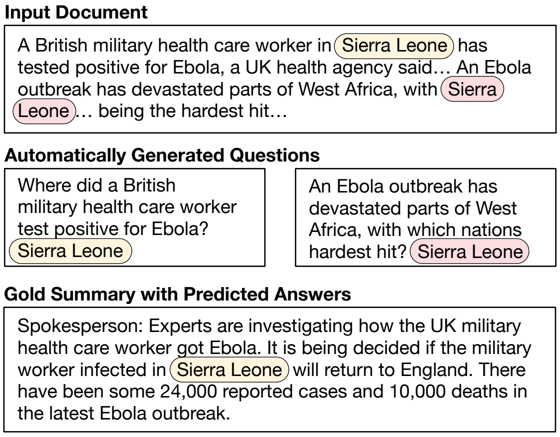
Our experiments on three different summarization datasets show that the two-stage model trained with QA-based salient span supervision generates higher-quality summaries than lexical baseline methods of identifying salient spans on more extractive datasets according to several automatic evaluation metrics (§6.1). Further, our data augmentation procedure results in summaries that are significantly shorter with only a small reduction in the percent of target content covered, demonstrating it successfully eliminates undesired summary content (§6.2).

The contributions of our work include: (1) a novel method of including QA-based signals into summarization generation; (2) a two-stage model for incorporating phrase-level supervision into a summarization system; and (3) a data-augmentation procedure which results in more controllable summarization models.

2 Question-Based Salience

We begin by describing how QA is used to identify salient spans of text in the input document and discuss the advantages of this approach.

We define a document NP as salient if its corresponding predicate-argument relation also appears in the gold summary. To identify such NPs automatically, we employ question-generation and question-answering models as follows.



A British military health care worker in Sierra Leone has tested positive for Ebola, a UK health agency said… An Ebola outbreak has devastated parts of West Africa, with Sierra Leone… being the hardest hit…

Where did a British military health care worker test positive for Ebola? Sierra Leone

An Ebola outbreak has devastated parts of West Africa, with which nations hardest hit? Sierra Leone

Spokesperson: Experts are investigating how the UK military health care worker got Ebola. It is being decided if the military worker infected in Sierra Leone will return to England. There have been some 24,000 reported cases and 10,000 deaths in the latest Ebola outbreak.

Figure 2: We define a document noun phrase as salient if the wh-question it answers is also answered in the gold summary. Here, the first (yellow) instance of "Sierra Leone" is salient and the second (red) is not because the gold summary answers the automatically generated question for the first instance but not the second.

For each NP in the source document, we use the sentence it appears in to automatically generate a wh-question for which the NP is the answer. This QA pair represents the predicate-argument relation that the NP participates in. Then, we assume if a second text can be used to correctly answer that question, it contains the same predicate-argument relation. Thus, we use a QA model to automatically answer the question against the gold summary and mark the NP as salient if the QA model predicts the question is answerable and the predicted answer is correct. In practice, we assume a predicted answer is correct if it shares at least one token in common with the NP which was used to generate the question.

An example of this procedure is illustrated in Fig. 2 for two occurrences of the NP "Sierra Leone." Questions for each phrase are automatically generated from the input document and answered against the gold summary. Since the QA model correctly answered the first question but predicted the second question is not answerable, only the first occurrence of "Sierra Leone" is marked as salient.

We refer to the NPs identified by this procedure as "silver spans."[2] Specific implementation details

---

[2] The term "silver" refers to the fact that the salient spans are not perfect because they were identified by sequence of learned models rather than humans ("gold" spans; §6.1).

of the generation and answering models can be found in §5.

2.1 Advantages of a QA-Based Approach

Using QA to identify salient spans of text has several advantages. First, because our QA approach operates at the phrase-level, it is able to be more precise about what specifically is salient in the document in contrast to sentence-level approaches. For example, in the second sentence of Fig. 1, the QA-based salience signal identifies “Jonathan” and “his defeat” as salient but not “written statement.” A sentence-level approach would mark the entire sentence as salient and thus cannot make that distinction.

Second, because the QA-based approach reasons about the predicate-argument structure of the text, it is able to distinguish between which specific instances of the same NP are salient and which are not. This is illustrated in Fig. 2 in which the first occurrence of “Sierra Leone” is marked as salient but the second is not because the gold summary does say the health care worker was infected in Sierra Leone, but it does not say it is one of the hardest hit countries. A salience signal that uses a bag-of-n-grams approach (e.g., ROUGE-based methods) cannot easily decide which instance “Sierra Leone” is salient.

3 A Two-Stage, Span-Based Model

Next, we propose a two-stage, span-based model that can incorporate the QA-based salience signals into the learning procedure. The first of the two stages, the span selection component, classifies salient spans within the text. The second stage, the generation component, generates the summary given the document and the salient spans. The details of each component are detailed next.

3.1 Salient Span Classifier

Given an input document $d = [x_1, \ldots, x_n]$ and a set of spans S, in which each span $s_{i,j}$ represents a sequence of tokens $x_i, \ldots, x_j$ in d, the span classifier outputs a score for each span based on how salient it is in the document. Our definition of salience is discussed in §2.

Concretely, the input tokens are first encoded using BART. Then, the representation of a span is created by concatenating the BART encodings of the first and last tokens in the span. Finally, a linear classifier is trained using this encoding to predict the salience of each span.

A set of silver spans $S^* \subseteq S$ is used to train the model using a binary cross-entropy loss. When using the QA-based approach, S is the set of NPs in the document and S^* is the subset that our QG-QA algorithm identified as salient. We reweight the loss term of each span such that positive and negative spans contribute equally. During inference, a score is predicted for each span in S and the top-k sorted by highest score are passed to the generation component. We choose the k spans independently, although they could also be selected jointly.

3.2 Generation Component

Given an input document and set of salient spans, the generation component produces a summary of the document. The salient spans are represented by inserting special tokens directly into the document’s sequence of tokens before and after the spans. For example, if span $s_{4,5}$ was marked as salient, the document’s tokens would be represented as

```
... x3 [SS] x4 x5 [SE] x6 ...
```

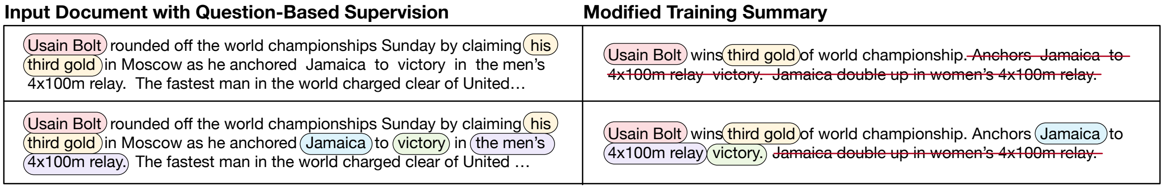
where `[SS]` and `[SE]` mark the start and end of the span.

Since the salient spans are represented in the document tokens, we are able to directly train a sequence-to-sequence model to generate the gold summary from the modified document representation without any changes to the model’s architecture.

During training, we use silver spans and the ground-truth summary to fine-tune BART using a standard cross-entropy loss function. The ground-truth summary does not have any marking of salient spans. For inference, the predicted salient spans from the span classifier are used instead of the silver spans.

4 Improving Controllability via Data Augmentation

Although there is nothing to directly force the generation model to learn to include content based on the supervision provided by the salient spans, if the supervision is of high enough quality, we expect the model will learn to do so. Indeed, we later show in §6.2 that this is true, thus the content of the summary can be controlled by which spans are marked as salient. However, it is also desirable for a controllable summarization model to also not include



Input Document with Question-Based Supervision	Modified Training Summary
Usain Bolt rounded off the world championships Sunday by claiming his third gold in Moscow as he anchored Jamaica to victory in the men's 4x100m relay. The fastest man in the world charged clear of United…	Usain Bolt wins third gold of world championship. ~~Anchors Jamaica to 4x100m relay victory. Jamaica double up in women's 4x100m relay.~~
Usain Bolt rounded off the world championships Sunday by claiming his third gold in Moscow as he anchored Jamaica to victory in the men's 4x100m relay. The fastest man in the world charged clear of United …	Usain Bolt wins third gold of world championship. Anchors Jamaica to 4x100m relay victory. ~~Jamaica double up in women's 4x100m relay.~~

Figure 3: An example of our data augmentation procedure. The colors represent the mapping between document and summary spans. The document spans are given to the generation model during training. In this example, no span maps to the third summary sentence, so it is removed entirely. Then, new training instances are generated using the first summary sentence and first two summary sentences with their corresponding salient document spans.

content which is not marked as salient. The generation models may learn to include extra information for at least two reasons.

First, the gold summaries may include content which cannot be generated based on only the silver salient spans that were used to train the generation model, so it may learn to output extra, unmarked information. This could happen if the QG/QA models are imperfect (resulting in a noisy supervision signal) or if the gold summary contains information that cannot be mapped to the document. Second, if the model is trained to generate summaries of a certain length and the length of the summary necessary to include all of the information marked by the spans is smaller than those used for training — for example, because the number of marked spans is small — the model could generate additional information simply to increase the summary length.

An artifact of our silver span annotation procedure enables us to address these controllability issues. If a document span is marked as salient, that means it has a corresponding phrase in the gold summary which expresses the same content. Therefore, the QG-QA procedure creates a mapping between which parts of the gold summary should be able to be generated by marking different parts of the input document.

We propose to leverage this mapping to augment the training data in two ways. First, we remove any gold summary sentence which has no phrase mapped to the document. These sentences would encourage the model to generate additional content based on unmarked spans.

Second, we generate new pairs of salient spans and gold summaries for training by selecting the first k remaining gold summary sentences and the subset of salient document spans which map to them. For instance, if $k = 2$, only the salient spans which are mapped to the first two summary sentences are marked in the input document, and the model is trained to generate only those sentences. We generate new examples for each original training instance using all possible values of k. By training on these new pairs, the model should learn to better control the length of the output summary based on the number of marked salient spans. An example of these augmentations is included in Fig. 3.

Although this procedure is described within the context of the QA-based supervision, it can be implemented with any such mapping between the document and gold summaries.

5 Experimental Setup

Datasets Our experiments use three popular English single-document summarization datasets: CNN/DailyMail (Nallapati et al., 2016), XSum (Narayan et al., 2018), and NYTimes (Sandhaus, 2008). Specific details on the sizes of the datasets can be found in Appendix A.

Baselines & Other Work We compare the salient spans selected by our QA-based method against three baseline span selection methods. The first marks salient sentences by greedily selecting k sentences that maximize the ROUGE-2 score calculated against the gold summary, a popular method that is frequently used to train extractive summarization models (Nallapati et al., 2017) as well as other two-step abstractive systems (Chen and Bansal, 2018; Dou et al., 2021). The other two mark entities and NPs as salient if they appear in the gold summaries as determined by lexical matching. We only mark the first occurrence of the phrases as salient since we found that worked better than marking all occurrences.

Additionally, we compare our results to BART (the original implementation and our own; Lewis et al., 2020) since our models are built on top of it. We also compare to GSum (Dou et al., 2021), which uses salient sentence guidance that is similar to our baseline salient sentence method. GSum encodes the additional guidance signal separately from the input document and uses the document and guidance encodings to generate the summary.

Summarization Evaluation Metrics The models are automatically evaluated using three metrics which calculate a similarity score between the generated and gold summaries. ROUGE (Lin, 2004) compares the two summaries based on their lexical overlap. BERTScore (Zhang et al., 2020) calculates a similarity score between the summaries based on their tokens' BERT embeddings (Devlin et al., 2019). QAEval (Deutsch et al., 2021) is a QA-based evaluation metric which generates questions from the gold summaries and answers them against the generated summaries. Its similarity score is equal to the average token F_1 score calculated between the predicted and expected answers.

We additionally perform a human evaluation of summary quality on Mechanical Turk. We ask 3 Turkers to rate the quality of 50 summaries per model from the CNN/DailyMail dataset on a scale from 1 to 5 based on the importance of the information, faithfulness, fluency, and coherence. Details on the manual evaluation can be found in Appendix G.

Controllability Evaluation Metrics The controllability of our model is evaluated using the *question recall*. Given k marked spans, we define the question recall to be equal to the percent of the corresponding k wh-questions that are answered by the summary according to the QA model. This approximates the recall on the desired predicate-argument structures in the summary. We additionally report the ratio between k and the length of the generated summary in tokens to measure the precision of the generated information. A larger value means the summary is more concise.

Implementation Details The QG/QA models are the same as used by QAEval. The generation model is initialized with BART-Large and fine-tuned on data collected by Demszky et al. (2018). The answering model is initialized with ELECTRA-Large (Clark et al., 2020) and fine-tuned on SQuAD 2.0 (Rajpurkar et al., 2018).

The span classification and generation models are both initialized with BART-Large and fine-tuned on the respective datasets. They were trained for three and five epochs, respectively, and the model with the best precision@1 and ROUGE-2 F_1, respectively, on the validation set were selected as the final models. See Appendix B for more specific implementation details.

6 Results

6.1 Summarization Evaluation

Automatic Evaluation Table 1 contains the models' performances as evaluated by automatic metrics, both using the spans predicted by the classifier ("end-to-end") and the silver spans (i.e., assuming a "perfect" classifier).

Interestingly, we find a somewhat surprising result. On CNN/DailyMail and NYTimes, the end-to-end QA-based model performs the best among the different span labeling methods and the baseline BART. On NYTimes, it is also better than GSum. However, if the silver span labels are used, the lexical NP-based model out-performs the rest by a somewhat large margin. It is surprising that a seemingly better generation model would result in worse end-to-end performance.

To better understand this result, we manually labeled all of the NPs in 50 CNN/DailyMail documents as salient or not salient based on whether the corresponding predicate-argument relation was present in the reference summary (see Appendix E for details). These spans, which we call the gold spans, can be used to evaluate the precision and recall of the silver spans as well as the output from the salient span classifiers.

Table 2 shows that the QA-based labels are more precise but have lower recall than the lexical NP labels. Because the lexical NP method aggressively marks the first occurrence of any NP in the document which is present in the reference as salient, it is unsurprising that its recall would be high. Since it cannot distinguish between instances of the same NP due to its bag-of-words representation, its precision is low. In contrast, the QA-based approach can reason about which occurrence of an NP is salient (resulting in higher precision), but the recall is lower likely due to noise in the QG/QA models. This same pattern appears in Table 2 with the outputs from the salient span classifiers, although the precisions and recalls are notably lower than the silver span labels'.

We believe that the discrepancy between the end-to-end and silver span-based models' performances can be explained by these results. The lexical NP generation model was trained with a high recall silver supervision at 66.3, allowing the generation

Method	CNN/DailyMail					NYTimes				
	R1	**R2**	**RL**	**BSc**	**QAE**	**R1**	**R2**	**RL**	**BSc**	**QAE**
Baselines & Other Work										
BART	44.2	21.3	40.9	-	-	-	-	-	-	-
BART (ours)	44.1	21.0	40.9	88.3	23.5	54.0	35.2	50.7	89.5	27.3
GSum	46.0†	22.3†	42.6†	88.6†	22.9	54.3	35.4	47.6	-	-
Silver Spans										
Sentences	51.7	29.9	48.8	89.4	28.6	62.7	46.0	59.8	91.2	33.5
Entities	51.5	27.6	48.0	89.6	30.0	60.9	42.8	57.6	90.8	32.0
Lexical NPs	**59.6**	**34.6**	**55.8**	**90.6**	**36.2**	**68.2**	**50.7**	**64.8**	**92.0**	**36.6**
QAs	55.3	31.4	51.9	90.0	33.7	65.7	48.7	62.6	91.6	35.8
End-to-End										
Sentences	45.0	**21.8**	41.8	88.2	23.2	54.6	35.9	51.4	89.6	27.6
Entities	43.5	20.3	40.4	88.3	23.2	53.5	34.6	50.3	89.4	27.0
Lexical NPs	44.8	21.0	41.6	88.4	23.2	54.6	35.4	51.3	89.6	27.1
QAs	**45.5**	**21.9**	**42.4†**	**88.5**	**24.4†**	**55.2†**	**36.3†**	**51.9†**	**89.7†**	**28.0†**

Table 1: The automatic metric results for the baselines and other work (top), models that use silver spans (middle), and end-to-end models (bottom) evaluated with ROUGE (R1, R2, RL), BERTScore (BSc), and QAEval (QAE). Values in bold are statistically the best in each section and † marks the best values overall (excluding silver labels) using a permutation test with $\alpha = 0.05$.

Method	Precision	Recall	F_1
Silver Labels			
Lexical NPs	32.7	**66.3**	41.8
QAs	**43.8**	51.5	**45.3**
Predicted Spans			
Lexical NPs@25	23.8	**54.4**	**32.0**
QAs@20	**27.3**	49.1	**33.8**

Table 2: The average summary-level precision, recall, and F_1 scores of the silver labeling methods (top) and the output from the span classifiers (bottom) evaluated against the human-annotated gold labeling. Results in bold are statistically higher (or tied) under a single-tail pairwise permutation test with $\alpha = 0.05$. The $@k$ values were selected based on validation set performance.

Method	CNN/DailyMail		
	R1	**R2**	**RL**
Silver Spans			
Lexical NPs	**59.6**	**34.6**	**55.8**
+10% Noise	57.8	32.8	54.0
+20% Noise	56.3	31.5	52.6
+30% Noise	55.0	30.4	51.4
+35% Noise	54.1	29.6	50.6
QAs	55.3	31.4	51.9
End-to-End			
Lexical NPs	44.8	21.0	41.6
+10% Noise	45.0	21.3	41.8
+20% Noise	45.2	21.6	42.0
+30% Noise	45.3	21.7	42.1
+35% Noise	45.1	21.6	41.9
QAs	**45.5**	**21.9**	**42.4**

Table 3: The ablated lexical NP supervision shows as the noise increases, the silver span performance decreases but end-to-end performance improves.

model to achieve good performance when the silver spans are provided. Yet during inference the model is provided with spans that only have around 54.4 recall, 12 points lower. We suspect the generation model learned to rely heavily on the marked salient spans — and empirically we observed that it copied very heavily from them — thus when the quality of the span signal was reduced, the resulting summaries similarly got worse. In contrast, the difference between the QA-based model's recall during training and inference is only estimated to be around 2.4, so this issue is less severe, resulting in better end-to-end summaries.

To test this hypothesis, we artificially ablated the lexical NP-based generation model's silver span supervision's recall by removing $k\%$ of the salient spans uniformly at random — thus making the training spans look more similar to the spans during inference — and retrained the model. We would expect the silver span-based model's performance to decrease while the end-to-end model's increases. Indeed, in Table 3 we find that this does happen. These results suggest that the relationship between the classifier's performance and generation model's supervision is important for good end-to-end results and could be explored in future work.

Although the end-to-end lexical NP results begin

	BART	Sentences	QA
Quality Score	3.76	**3.86**	**4.00**

Table 4: Summary quality scores according to humans. Results in bold are statistically tied for the best score.

to approach the QA-based model's performance, they do not quite reach it. Further, the QA-based silver spans maintain an F_1 advantage over the lexical NP method (Table 2). While the QA-based approach can be improved with better question generation and answering models, the lexical NP labeling method is inherently limited. Therefore, the QA-based method does appear to be the best method of incorporating additional supervision into the summarization models based on the automatic metrics.

Human Evaluation Table 4 contains the results of evaluating BART and the sentence- and QA-based models on CNN/DailyMail (the best performing) using human summary quality annotations from Mechanical Turk. On average, our span-based methods have higher quality summaries than the baseline method of BART. After collecting annotations for 50 summaries on CNN/DailyMail, we were unable to obtain statistical significance between the two span-based models, however, doing so may be prohibitively difficult (Wei and Jia, 2021).

6.2 Controllability Evaluation

Automatic Evaluation The controllability of the QA-based generation model is evaluated in Fig. 4 using the original training data as well as the augmented data described in §4. We plot the question recall and the ratio between k and the length of the generated summaries for the top k most salient spans output by the QA-based salient span classifier for various values of k on CNN/DailyMail. The data augmentation procedure is split into only removing sentences that do not answer a question ("+Rm Sents") plus also generating new training examples ("+New Examples"). We also include the results for BART (for which the summary is constant for all k) for relative comparisons.

Although BART's question recall is initially higher than the QA models' recalls, as k increases it falls lower. We suspect this is because BART has learned to include the same content that the span classifier also identifies as salient when k is small and the length of its summaries allows it to cover more content. However, when k increases, the span classifier potentially predicts different spans as salient than what BART learned, resulting in divergent content and a lower recall for BART. The higher recall of the QA models demonstrates that their summary content is indeed being controlled via the input spans. Further, the QA models have far better k-to-length ratios, meaning their summaries are shorter than BART's even when their recalls are higher, suggesting they generate far less content which is unrelated to the marked spans.

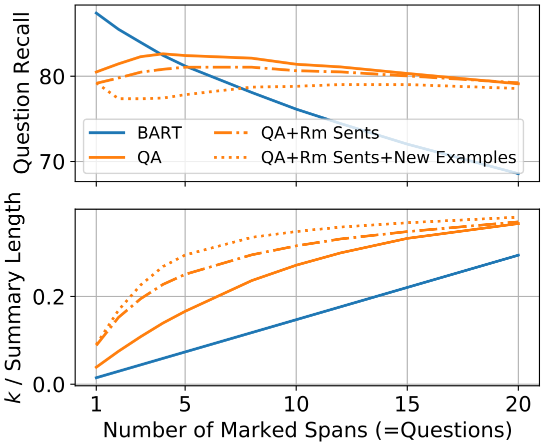
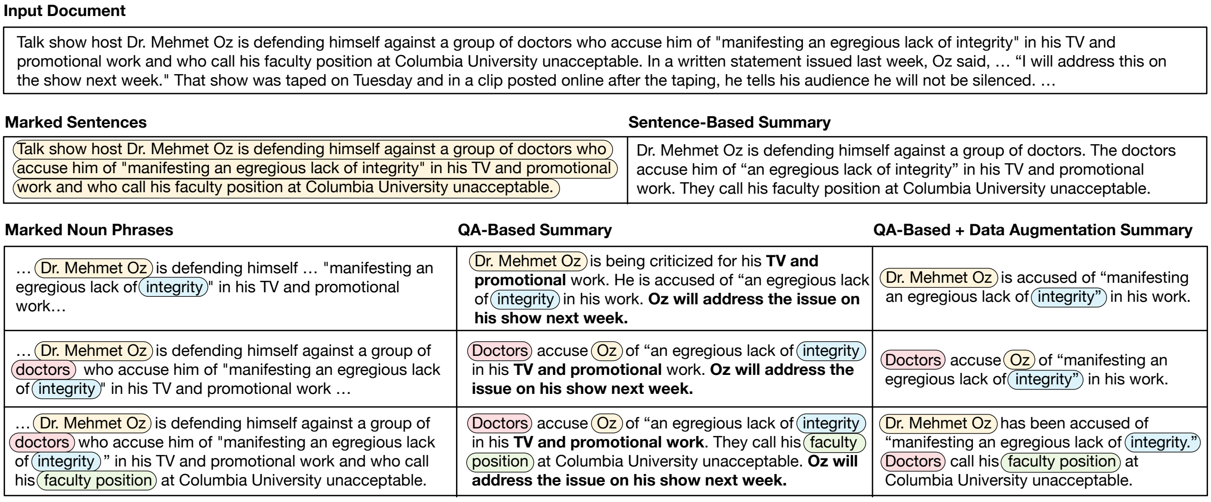


Figure 4: The percent of questions which correspond to the marked spans answered by the generated summaries (top) and the summary lengths in tokens (bottom). The QA methods have higher question recall than BART and are far more concise, demonstrating that marking input spans controls the summary content.

Among the QA-based models, we do observe a small drop in recall when the model is trained with data augmentation. However, the data-augmented summaries express that information far more concisely (because the ratio between k and the summary length is higher). For example, when 10 input spans are marked, there is a relative 0.9% and 3.2% drop in recall for removing sentences and the full data augmentation procedure, respectively, but the summary lengths are 14% and 22% shorter. Therefore, the data augmentation procedures do result in models which have learned to not generate extra content.

Controllability Example Example summaries from the QA models and sentence-based model with different marked input spans are shown in Fig. 5. Because the sentence-based model is limited to marking full sentences, the content which is taken from the marked sentence cannot be further



Input Document

Talk show host Dr. Mehmet Oz is defending himself against a group of doctors who accuse him of "manifesting an egregious lack of integrity" in his TV and promotional work and who call his faculty position at Columbia University unacceptable. In a written statement issued last week, Oz said, … "I will address this on the show next week." That show was taped on Tuesday and in a clip posted online after the taping, he tells his audience he will not be silenced. …

Marked Sentences	Sentence-Based Summary
Talk show host Dr. Mehmet Oz is defending himself against a group of doctors who accuse him of "manifesting an egregious lack of integrity" in his TV and promotional work and who call his faculty position at Columbia University unacceptable.	Dr. Mehmet Oz is defending himself against a group of doctors. The doctors accuse him of "an egregious lack of integrity" in his TV and promotional work. They call his faculty position at Columbia University unacceptable.

Marked Noun Phrases	QA-Based Summary	QA-Based + Data Augmentation Summary
… Dr. Mehmet Oz is defending himself … "manifesting an egregious lack of integrity" in his TV and promotional work…	Dr. Mehmet Oz is being criticized for his **TV and promotional** work. He is accused of "an egregious lack of integrity in his work. **Oz will address the issue on his show next week.**	Dr. Mehmet Oz is accused of "manifesting an egregious lack of integrity" in his work.
… Dr. Mehmet Oz is defending himself against a group of doctors who accuse him of "manifesting an egregious lack of integrity" in his TV and promotional work …	Doctors accuse Oz of "an egregious lack of integrity in his **TV and promotional** work. **Oz will address the issue on his show next week.**	Doctors accuse Oz of "manifesting an egregious lack of integrity" in his work.
… Dr. Mehmet Oz is defending himself against a group of doctors who accuse him of "manifesting an egregious lack of integrity" in his TV and promotional work and who call his faculty position at Columbia University unacceptable.	Doctors accuse Oz of "an egregious lack of integrity in his **TV and promotional work**. They call his faculty position at Columbia University unacceptable. **Oz will address the issue on his show next week.**	Dr. Mehmet Oz has been accused of "manifesting an egregious lack of integrity." Doctors call his faculty position at Columbia University unacceptable.

Figure 5: Example summaries generated by the sentence-based model (middle), QA-based model (bottom center) and QA-based model trained on the augmented data (bottom right). The QA-based models allow for much more control over the summary content than the sentence model by marking different combinations of phrases. The augmented-data summaries better eliminate unmarked content from the input than the standard model (extra information generated by the standard model shown in bold).

controlled. In contrast, the figure shows how the QA models' summaries can be altered by marking different NPs within the sentence, thus demonstrating the benefits of phrase-level controllability.

The example in Fig. 5 also shows how the data augmentation procedure improves controllability. The phrases which the standard model includes but the augmented model does not are marked in bold. The augmented model does a better job at excluding content which was not marked in the input document.

7 Related Work

QA-Based Signals QA-based signals have been used for evaluating summaries (Eyal et al., 2019; Durmus et al., 2020; Wang et al., 2020; Deutsch et al., 2021), including Scialom et al. (2021), who explore a similar notion of document salience. They have also been used to align content across documents (Weiss et al., 2021) as well as train summarization models (Arumae and Liu, 2018, 2019; Scialom et al., 2019). The models which incorporate QA-based signals typically do so using reinforcement learning. In contrast, our approach is simpler. We incorporate the QA-based signal by marking spans in the document, and our models are trained using easier-to-optimize cross-entropy objective functions.

Incorporating Additional Supervision Recent work by Dou et al. (2021) proposes a framework for incorporating additional guidance into summarization models, called GSum. They separately encode the input document and the supervision signal, whereas we directly mark spans in the text. This allows for our generation component to have a simpler architecture than theirs. While they are able to encode any natural language string, our model provides more direct supervision by identifying which specific tokens are salient.

Other work has included predicate-argument structure into summarization to generate more faithful summaries (Cao et al., 2018; Jin et al., 2020; Zhu et al., 2021). They represent the predicate-arguments either using dependency trees or OpenIE tuples, whereas we represent them via QA pairs. These works include that information to try and generate faithful summaries, whereas our goal is to identify salient document content.

Controllable Summarization Work on controllable summarization has focused on aspects such as the length of the summary (Fan et al., 2018) and the content in an interactive setting (Shapira et al., 2017) or via prompting (He et al., 2020). Incorporating our QA-based signal via prompting may be difficult given the number of questions which would need to be concatenated onto the input.

Other approaches control content via planning as in entity templates (Narayan et al., 2021), marking records in a data-to-text approach (Puduppully et al., 2019), or using aspect controllers (Amplayo et al., 2021). The marked salient spans in our work could be viewed as a content plan as well.

Data Augmentation Previous work has proposed methods for removing sentences or full summaries from the training data in order to discourage the summarization model from learning to generate unfaithful information (Matsumaru et al., 2020; Nan et al., 2021; Narayan et al., 2021). In addition to removing sentences, we generate new training instances in order to learn to exclude content which is not marked as salient in the input, resulting in more controllable models.

8 Conclusion

In this work, we proposed a method for incorporating QA-based signals into a summarization model by automatically marking document NPs as salient based on whether a NP's corresponding wh-question is answered correctly in the summary. We showed that incorporating this signal into our two-stage summarization model results in higher quality summaries than baseline methods of identifying salient spans. Finally, we demonstrated that our data augmentation algorithm, which attempts to ensure the span supervision is consistent with the gold summaries, improves controllability by eliminating unmarked content from the output summaries.

Acknowledgments

The authors would like to thank the anonymous EACL reviewers for their insightful feedback on our work.

This work was supported by a Focused Award from Google and Contracts FA8750-19-2-1004 and FA8750-19-2-0201 with the US Defense Advanced Research Projects Agency (DARPA). Approved for Public Release, Distribution Unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

Limitations

The summarization model proposed in this work relies on the existence of question generation and question answering models, which are used to preprocess the data for training. It is likely that such models only exist or are of high-enough quality for high resource languages, such as English, which limits the languages that our model can be trained on in practice. However, we do not see a reason why our model could not be applied to another language as long as those additional resources are available.

Further, because the question generation model produces simple wh-questions and the question answering model is only able to reason about the predicate-argument structure of the text (due to it being trained on SQuAD 2.0), our procedure for identifying salient document phrases requires that the same information across the document and reference summary must be expressed in relatively similar ways (e.g., up to rephrasing and synonyms). If the reference summary does contain the answer to the generated question but identifying that answer requires a level of reasoning beyond reasoning about the predicate-argument structure (e.g., does it require multi-hop reasoning?), the specific models proposed in this work may fail to identify those salient phrases. This limits the types of datasets for which we expect our proposal to do well on (discussed more in Appendix D). However, if matching document and reference summary information requires a level of reasoning that is supported by the generation and answering models, then we suspect our proposal will work, in theory, but we have not experimented with this in practice.

Our methods out-perform the baseline systems the most when evaluated by QAEval, and we leverage the same QG/QA technology as this metric. While this commonality may bias our system toward summaries that are favored by QAEval, we argue this is not necessarily a bad thing. Previous work has incorporated ROUGE-based signals into their models, either indirectly by selecting extractive labels based on ROUGE or directly by optimizing ROUGE via reinforcement learning. Our approach is analogous to these modeling approaches, and QAEval has been demonstrated to be a better evaluation metric than ROUGE, so it is likely a better metric to optimize for.

References

Reinald Kim Amplayo, Stefanos Angelidis, and Mirella Lapata. 2021. Aspect-Controllable Opinion Summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6578–6593, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kristjan Arumae and Fei Liu. 2018. Reinforced Extractive Summarization with Question-Focused Rewards. In *Proceedings of ACL 2018, Student Research Workshop*, pages 105–111, Melbourne, Australia. Association for Computational Linguistics.

Kristjan Arumae and Fei Liu. 2019. Guiding Extractive Summarization with Question-Answering Rewards. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2566–2577, Minneapolis, Minnesota. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the Original: Fact Aware Neural Abstractive Summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Yen-Chun Chen and Mohit Bansal. 2018. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming Question Answering Datasets Into Natural Language Inference Datasets. *ArXiv*, abs/1809.02922.

Daniel Deutsch, Tania Bedrax-Weiss, and Dan Roth. 2021. Towards Question-Answering as an Automatic Metric for Evaluating the Content Quality of a Summary. *Transactions of the Association for Computational Linguistics*, 9:774–789.

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

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A General Framework for Guided Neural Abstractive Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A Question Answering Evaluation Framework for Faithfulness Assessment in Abstractive Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Matan Eyal, Tal Baumel, and Michael Elhadad. 2019. Question Answering as an Automatic Evaluation Metric for News Article Summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3938–3948, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable Abstractive Summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. CTRLsum: Towards Generic Controllable Text Summarization.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. SemSUM: Semantic Dependency Guided Neural Abstractive Summarization. In *Proc. of the Conference on Artificial Intelligence (AAAI)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *Proc. of the International Conference on Learning Representations*.

Kazuki Matsumaru, Sho Takase, and Naoaki Okazaki. 2020. Improving Truthfulness of Headline Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1335–1346, Online. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In *Proc. of the Conference on Artificial Intelligence (AAAI)*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level Factual Consistency of Abstractive Text Summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Shashi Narayan, Yao-Dong Zhao, Joshua Maynez, Gonccalo Simoes, Vitaly Nikolaev, and Ryan T. McDonald. 2021. Planning with Learned Entity Prompts for Abstractive Summarization.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-Text Generation with Content Selection and Planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6908–6915.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Evan Sandhaus. 2008. The New York Times Annotated Corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization Asks for Fact-based Evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. Answers Unite! Unsupervised Metrics for Reinforced Summarization Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3246–3256, Hong Kong, China. Association for Computational Linguistics.

Ori Shapira, Hadar Ronen, Meni Adler, Yael Amsterdamer, Judit Bar-Ilan, and Ido Dagan. 2017. Interactive Abstractive Summarization for Event News Tweets. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 109–114, Copenhagen, Denmark. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and Answering Questions to Evaluate the Factual Consistency of Summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Johnny Wei and Robin Jia. 2021. The Statistical Advantage of Automatic NLG Metrics at the System Level. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6840–6854, Online. Association for Computational Linguistics.

Daniela Brook Weiss, Paul Roit, Ayal Klein, Ori Ernst, and Ido Dagan. 2021. QA-Align: Representing Cross-Text Content Overlap by Aligning Question-Answer Propositions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9879–9894, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2021. Enhancing Factual Consistency of Abstractive Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 718–733, Online. Association for Computational Linguistics.

A Dataset Statistics

The sizes of the CNN/DailyMail, XSum, and NYTimes datasets are included in Table 5. The Table

Dataset	#Train	#Valid	#Test	Span Type	#Spans
CNN/DM	287,113	13,368	11,490	Sentences	3
				Entities	10
				Lex. NPs	25
				QA	20
XSum	204,045	11,332	11,334	Sentences	1
				Entities	1
				NPs	5
				QA	1
NYTimes	44,382	5,523	6,495	Sentences	4
				Entities	15
				Lex. NPs	45
				QA	27

Table 5: The number of instances in the training, validation, and test splits of the three datasets used in our experiments as well as the number of spans selected by the classification component that were passed as input to the generation component.

also includes the number of spans per span type that were selected from the classification component and passed to the generation component during inference. The values were selected based on a parameter sweep on the validation set. The number of spans with the highest ROUGE-2 F_1 score was selected.

B Implementation Details

All of the models were trained with the same hyperparameters for across datasets and span types which were based on those used by BART (Lewis et al., 2020).

The classification component was a BART-Large model that was fine-tuned with a binary cross-entropy classification loss. We selected the model based on which had the best precision@1 on the validation dataset. The generation models were also fine-tuned BART-Large models, but they instead use a cross-entropy loss function.

Both the components were trained using Adam (Loshchilov and Hutter, 2019) with weight decay and learning rate 3e-5. The classification component was trained for 3 epochs, and the final model was selected based on the precision@1 on the validation set. The generation component was trained for 5 epochs, and the final model was selected based on the ROUGE-2 F_1 score on the validation set.

C Salient Span Classifier Evaluation

Fig. 6 contains the precision@k and recall@k of the span based classifiers calculated against the corresponding silver spans. These plots should be interpreted as how well the span classifiers were able to learn from their respective supervision, not necessarily the true quality of the output span labels (which would require evaluating against human-annotated gold labels, as in Table 2). The "x" symbols denote the operating points used in the end-to-end model, which were chosen based on the number of spans that resulted in the highest ROUGE-2 F_1 score on the validation data.

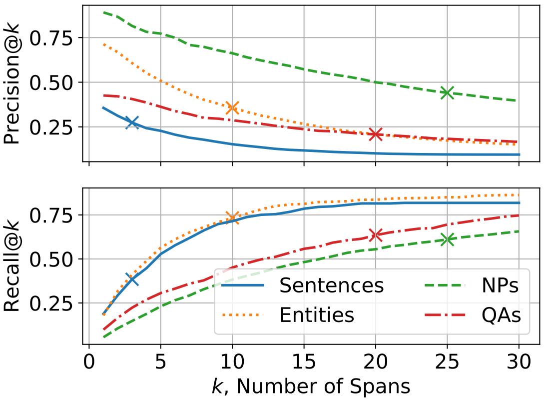


Figure 6: The performances of the salient span classifiers using the different types of salient phrase labeling evaluated against the silver spans. The "x" marks the operating points used in the end-to-end models.

D XSum Results

Table 6 contains the automatic summarization evaluation results on the XSum dataset. These results are included in the Appendix because incorporating the span-based supervision does not improve end-to-end results over the baseline BART model, which is a conclusion also reached by GSum, a model closely related to ours.

We suspect this is due to the abstractive nature of XSum compared to the more extractive CNN/DailyMail and NYTimes. Since the methods for identifying salient spans rely on the document and gold summary explicitly stating the salient content, we suspect the abstractiveness of XSum would result in this happening less frequently and thus be less beneficial to a summarization model trained on XSum.

E Gold Span Annotation Protocol

We selected 50 test instances from the CNN/DailyMail dataset uniformly at random and labeled each of the document NPs as salient or

Method	XSum				
	R1	R2	RL	BSc	QAE
Baselines & Other Work					
BART	45.1	21.3	40.9	-	-
BART (ours)	$45.7^\dagger$	$22.4^\dagger$	$37.2^\dagger$	$91.3^\dagger$	$18.9^\dagger$
GSum	44.9	21.2	36.0	90.4	17.9
Silver Spans					
Sentences	47.3	24.2	38.7	91.5	19.9
Entities	48.1	24.2	39.1	91.7	21.3
Lexical NPs	**54.3**	**29.3**	**44.1**	**92.4**	**26.1**
QAs	47.9	24.1	39.2	91.6	21.4
End-to-End					
Sentences	**45.0**	**21.7**	**36.6**	**91.2**	**18.6**
Entities	44.1	20.9	35.9	91.0	17.6
Lexical NPs	42.5	19.2	34.2	90.8	16.4
QAs	**45.1**	**21.8**	**36.7**	91.2	17.9

Table 6: The results of the models trained on the XSum dataset as evaluated with the automatic evaluation metrics. The span-based models do not improve over the baseline BART, potentially due to the abstractive nature of the XSum dataset.

not salient based on whether the corresponding predicate-argument relation also appears in the gold summary. We did not mark instances in which the NP's predicate-argument relation could be inferred from the gold summary via entailment as salient since our silver span labeling methods aim to mark phrases as salient if the content is explicilty included in the gold summary.

In general, this procedure was straightforward due to the extractive nature of the dataset in which the gold summaries copy heavily from the input document. If information was repeated in the input document, we tried to label the occurrence which contained the most predicate-argument relations which also matched the gold summary. That is, we selected the "best match." Otherwise, the first occurrence was selected.

Although our labeling procedure may be noisy, we do not have reason to believe that the labels may be biased in favor of either the lexical NP or QA labeling methods. Therefore, the statistics calculated from these labels should only be used as diagnostic tools to make relative comparisons between the different labeling methods rather than precise estimates of their exact values. 50 documents were sufficient to achieve statistically different results.

Our annotations will be released after publication.

Model	R1	R2	RL	BSc	QAE
Silver Spans					
QAs	55.3	31.4	51.9	90.0	33.7
QAs + Data Aug.	55.2	31.3	51.7	89.9	33.4
End-to-End					
QAs	45.5	21.9	42.4	88.5	24.4
QAs + Data Aug.	45.3	21.8	42.1	88.4	24.3

Table 7: The automatic evaluation metrics for summary quality are nearly the same for the QA-based model and the QA-based model trained on the augmented data.

F Data-Augmentation Automatic Evaluation

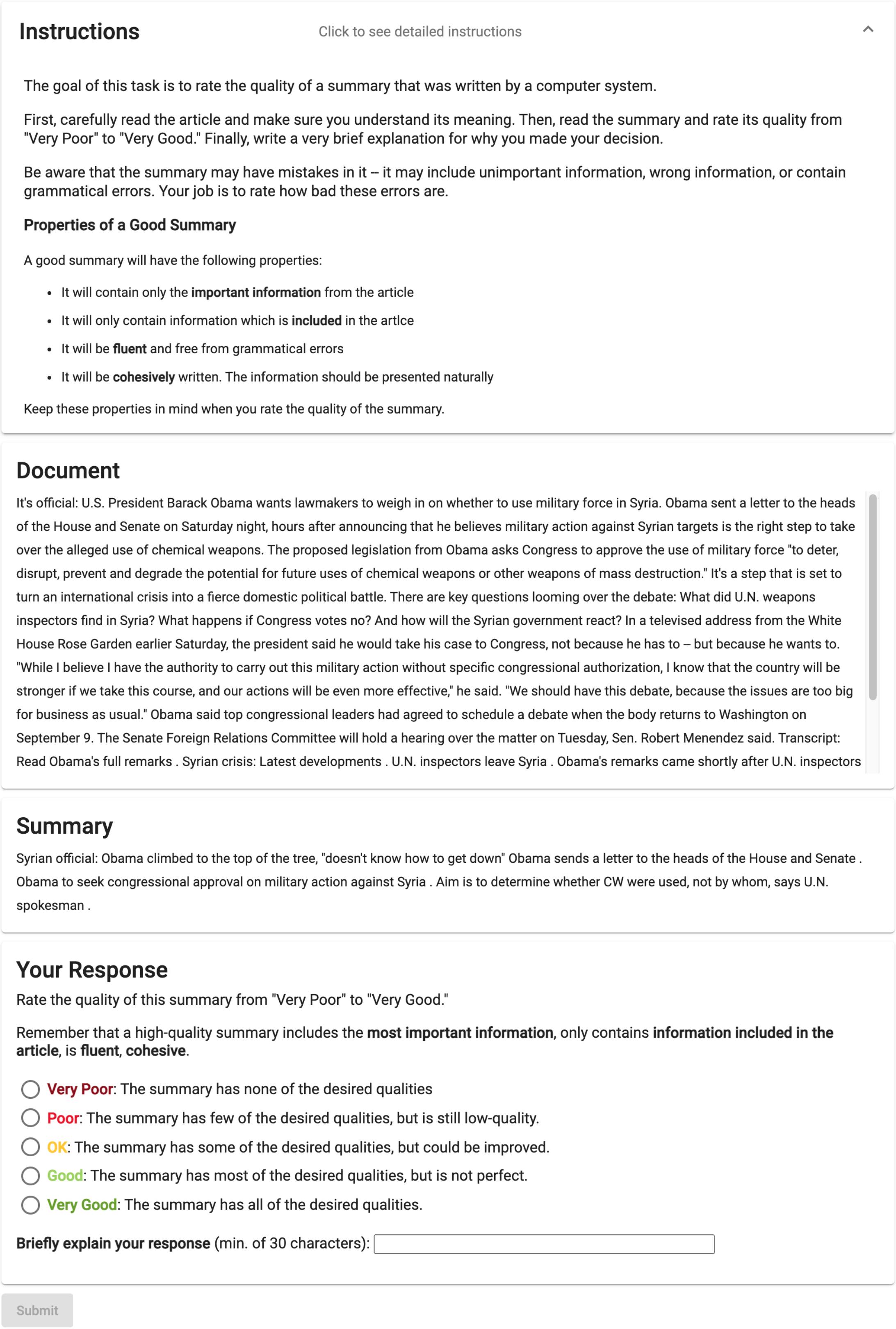
Table 7 contains the comparison between the standard and data-augmented training procedures based on the automatic metrics. The scores are nearly the same. The benefit of the model trained on the augmented data is in its controllability, which is not captured by this evaluation because the models trained with the standard and augmented training data receive the same spans as input supervision.

G Human Evaluation Details

Fig. 7 contains a screenshot of the tool we used for annotating summary quality on MTurk. The annotators were instructed to rate the summaries from "Very Poor" to "Very Good" based on whether the summary contained important information, was faithful to the input document, was fluent, and was cohesive. The ratings were converted to a Likert scale from 1-5 and averaged across all of the ratings for a system.

In order to encourage the annotators to pay attention to the task, we also required that they write a very brief explanation of how they made their decision, inspired by Narayan et al. (2021).

The MTurk annotators were paid at a rate of around $15 USD per hour.



Instructions

The goal of this task is to rate the quality of a summary that was written by a computer system.

First, carefully read the article and make sure you understand its meaning. Then, read the summary and rate its quality from "Very Poor" to "Very Good." Finally, write a very brief explanation for why you made your decision.

Be aware that the summary may have mistakes in it -- it may include unimportant information, wrong information, or contain grammatical errors. Your job is to rate how bad these errors are.

Properties of a Good Summary

A good summary will have the following properties:

- It will contain only the **important information** from the article
- It will only contain information which is **included** in the artlce
- It will be **fluent** and free from grammatical errors
- It will be **cohesively** written. The information should be presented naturally

Keep these properties in mind when you rate the quality of the summary.

Document

It's official: U.S. President Barack Obama wants lawmakers to weigh in on whether to use military force in Syria. Obama sent a letter to the heads of the House and Senate on Saturday night, hours after announcing that he believes military action against Syrian targets is the right step to take over the alleged use of chemical weapons. The proposed legislation from Obama asks Congress to approve the use of military force "to deter, disrupt, prevent and degrade the potential for future uses of chemical weapons or other weapons of mass destruction." It's a step that is set to turn an international crisis into a fierce domestic political battle. There are key questions looming over the debate: What did U.N. weapons inspectors find in Syria? What happens if Congress votes no? And how will the Syrian government react? In a televised address from the White House Rose Garden earlier Saturday, the president said he would take his case to Congress, not because he has to -- but because he wants to. "While I believe I have the authority to carry out this military action without specific congressional authorization, I know that the country will be stronger if we take this course, and our actions will be even more effective," he said. "We should have this debate, because the issues are too big for business as usual." Obama said top congressional leaders had agreed to schedule a debate when the body returns to Washington on September 9. The Senate Foreign Relations Committee will hold a hearing over the matter on Tuesday, Sen. Robert Menendez said. Transcript: Read Obama's full remarks . Syrian crisis: Latest developments . U.N. inspectors leave Syria . Obama's remarks came shortly after U.N. inspectors

Summary

Syrian official: Obama climbed to the top of the tree, "doesn't know how to get down" Obama sends a letter to the heads of the House and Senate . Obama to seek congressional approval on military action against Syria . Aim is to determine whether CW were used, not by whom, says U.N. spokesman .

Your Response

Rate the quality of this summary from "Very Poor" to "Very Good."

Remember that a high-quality summary includes the **most important information**, only contains **information included in the article**, is **fluent**, **cohesive**.

○ Very Poor: The summary has none of the desired qualities
○ Poor: The summary has few of the desired qualities, but is still low-quality.
○ OK: The summary has some of the desired qualities, but could be improved.
○ Good: The summary has most of the desired qualities, but is not perfect.
○ Very Good: The summary has all of the desired qualities.

Briefly explain your response (min. of 30 characters): ______

Figure 7: A screenshot of the tool we used for annotating summary quality on MTurk.

Patient Outcome and Zero-shot Diagnosis Prediction with Hypernetwork-guided Multitask Learning

Shaoxiong Ji and Pekka Marttinen
Aalto University, Finland
`{shaoxiong.ji; pekka.marttinen}@aalto.fi`

Abstract

Multitask deep learning has been applied to patient outcome prediction from text, taking clinical notes as input and training deep neural networks with a joint loss function of multiple tasks. However, the joint training scheme of multitask learning suffers from inter-task interference, and diagnosis prediction among the multiple tasks has the generalizability issue due to rare diseases or unseen diagnoses. To solve these challenges, we propose a hypernetwork-based approach that generates task-conditioned parameters and coefficients of multitask prediction heads to learn task-specific prediction and balance the multitask learning. We also incorporate semantic task information to improve the generalizability of our task-conditioned multitask model. Experiments on early and discharge notes extracted from the real-world MIMIC database show our method can achieve better performance on multitask patient outcome prediction than strong baselines in most cases. Besides, our method can effectively handle the scenario with limited information and improve zero-shot prediction on unseen diagnosis categories.

1 Introduction

Recent advances apply artificial intelligence to predict clinical events or infer the probable diagnosis for clinical decision support (Shickel et al., 2017; Li et al., 2021). Those works extensively study clinical data from Electronic Health Records (EHRs). For example, the Doctor AI model for predictive modeling on EHRs (Choi et al., 2016) and the Deep Patient model for unsupervised patient record representation learning (Miotto et al., 2016) is proposed by leveraging longitudinal medical data. Within the patient records, we can also find various clinical notes written by general practitioners or physicians and annotated with specific diagnosis results, medical codes, and other clues of patient outcome. Those clinical notes consist of meaningful health information, including health profile, clinical synopsis, diagnostic investigations, and medications, which can empower the clinical decision making (Li et al., 2021). Clinical text mining has demonstrated its feasibility in diverse healthcare applications, including medical code prediction (Ji et al., 2021), readmission prediction (Huang et al., 2019) and diagnosis prediction (van Aken et al., 2021). Readmission and length-of-stay prediction can help with utility management, especially when healthcare service is in high demand. Diagnostic prediction can support clinicians in making decisions based on many clinical reports. This study focuses on multitask patient outcome prediction from clinical notes, which classifies patients' diagnostic results and predicts clinical outcomes to empower expert clinicians and improve healthcare services' efficiency.

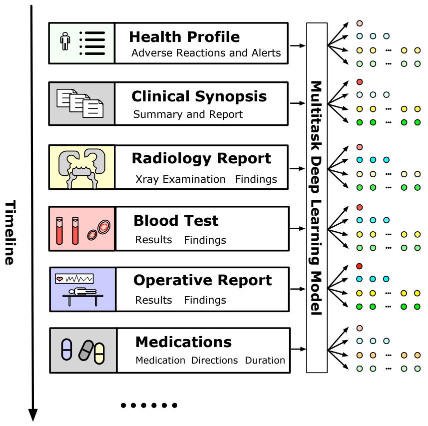


Figure 1: An example of multitask patient outcome prediction based on sequential inputs of clinical notes in the electronic health record. The multitask deep learning model dynamically predicts the probabilities of several patient outcomes at different stages during hospital admission.

Existing work on clinical note representation learning and multitask clinical outcome prediction focuses on building neural architectures for feature learning (Huang et al., 2019) or considers differ-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 589–598
May 2-6, 2023 ©2023 Association for Computational Linguistics

ent prediction tasks under the joint learning framework (Harutyunyan et al., 2019). However, there remain some unsolved challenges. First, the joint learning scheme for multiple tasks cannot effectively deal with inter-task interference, in which the predictors of different tasks compete with each other. Second, the task context information is usually underused, making multitask learning task-ignorant. Third, current learning algorithms are easy to overfit on seen training examples but fail to generalize on rare diseases or unseen diagnoses.

We propose a novel multitask learning method for patient outcome and zero-shot diagnosis prediction from clinical notes to solve the aforementioned challenges. We propose to incorporate task information to enable task-specific prediction in multitask learning. Inspired by the hypernetworks (Ha et al., 2017) that use a small neural network to generate parameters for a larger neural network, we encode the semantic task information as the task context and use the encoded task embeddings as shared meta knowledge to generate the task-specific parameters of prediction heads for multiple tasks. Our proposed method effectively utilizes the task information and produces task-aware multitask prediction heads. Our method is also generalizable on unseen diagnoses by maintaining the shared meta knowledge encoded as semantic task embeddings. Furthermore, we propose to dynamically update the weight coefficients of the multitask learning objective using another random hypernetwork, inspired by random projections (Wójcik and Kurdziel, 2019), to balance the learning from multiple clinical tasks.

Our contributions include:

- We propose to utilize task information via task embeddings for multitask patient outcome prediction spanning readmission, diagnosis, and length-of-stay prediction from clinical text and develop a hypernetwork for task-specific parameter generation to share information among different tasks.
- We propose to regularize the objective function via a randomly parameterized task weighting scheme that effectively balances the learning process among multiple tasks, taking their relationships into account.
- Experiments on two datasets extracted from a real-world clinical database show that our proposed method outperforms strong multitask learning baselines and achieves more robust performance in zero-shot diagnosis prediction.

2 Method

2.1 Problem Setup and Overall Architecture

The clinical document in patient health records is denoted as $\mathbf{d}_{1:n} = x_1, \ldots, x_n$, where each x_i is the i-th token. Patient outcome prediction predicts clinical results by mapping the input text into prediction scores with a function $\mathcal{F} : \mathcal{X}^n \rightarrow \mathcal{Y}^m$ such that $y = \mathcal{F}(x_1, \ldots, x_n; \boldsymbol{\theta})$, where $y \in \mathbb{R}^m$ is the patient outcome, m is the number of classes, and $\boldsymbol{\theta}$ are the model parameters. Under the multitask task setting, we have k learning tasks $\{\mathcal{T}_t\}_{t=1}^{k}$ that are related to the outcome of clinical intervention. Multitask learning algorithm captures the relatedness of multiple tasks and improves the modeling (Zhang and Yang, 2021). In this study, we predict multiple clinical outcomes. Thus, the goal of the learning process is to fit a single-input multi-output function $y_t = \mathcal{F}(x_1, \ldots, x_n; \boldsymbol{\theta}_{shared}, \boldsymbol{\theta}_t)$, where $y_t \in \mathbb{R}^{m_t}$ is the ground truth label and m_t is the number of classes of the t-th task, $\boldsymbol{\theta}_{shared}$ are the shared model parameters, and $\boldsymbol{\theta}_t$ the task-specific parameters. The learning objective is to minimize the weighted sum of loss functions of multiple tasks, denoted as

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{t=1}^{k} \alpha_t \mathcal{L}_t \left(y_t, \hat{p}_t; \boldsymbol{\theta}_{shared}, \boldsymbol{\theta}_t\right), \qquad (1)$$

where α_t is the weight coefficient of t-th task.

The progress of a clinical intervention comes with multiple records, as illustrated in Fig. 1, leading to lengthy clinical notes. Following Huang et al. (2019), we segment long clinical documents into several chunks, i.e., the document segmented to r chunks becomes $\mathbf{d}_{1:r} = c_1, \ldots, c_i, \ldots, c_r$, where the i-th chunk c_i contains s tokens. The segmentation has two advantages: 1) it enables progressive prediction from chunks of clinical notes calculated from the predictions of individual chunks (see details in Experiments); 2) it makes the input text length suitable for standard transformer-based pretrained models (described in the next section).

We propose a hypernetwork-guided multitask learning framework to predict multiple patient outcomes. Fig. 2 illustrates the overall framework, where the generation flow generates parameters.

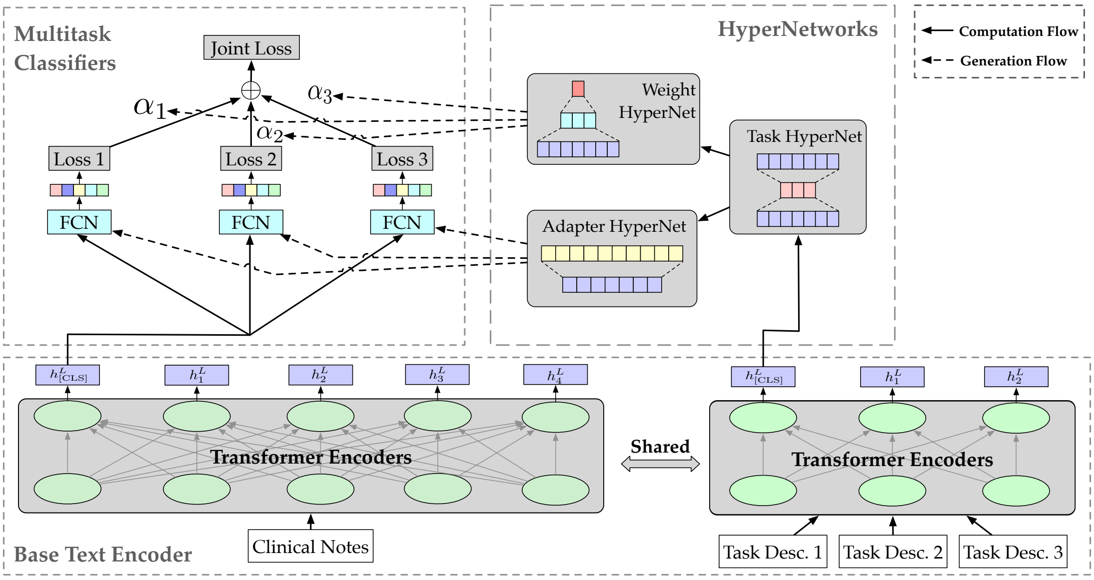


Figure 2: The illustration of our proposed hypernetwork-guided multitask learning framework

Specifically, we use a shared transformer-based text encoder to obtain hidden representations of clinical notes and task information. We develop a hypernetwork-based module, called Adapter HyperNet, to generate task-specific parameters for classification heads based on the task embeddings trained by a bottleneck network. The generated task-specific parameters then become the parameters of classification heads. To balance the learning of the joint objective function, we deploy the weight hypernetwork to generate weight coefficients conditioned on the task embeddings.

2.2 Base Text Encoder

We deploy the Bidirectional Encoder Representations from Transformers (BERT) model (Devlin et al., 2019) as our base text encoder to learn rich text features. The BERT text encoder pre-trains a language model with masked language modeling and next sentence prediction objectives in an unsupervised manner. Then, a downstream fine-tuning is followed by tuning the pre-trained model checkpoint with suitable learning objectives for downstream applications. The pre-training and fine-tuning paradigm can effectively exploit semantic knowledge from large training corpora and achieve superior performance in many downstream applications. Due to the discrepancy of vocabularies in general and specific domains, many efforts have been made to pre-train a transformer language model in various specific domains. To benefit the most from unsupervised representation learning for clinical application, we adopt the ClinicalBERT that starts from the standard BERT checkpoint and continues pre-training on clinical notes (Huang et al., 2019). Given text chunks c_i as inputs, we use the last layer's hidden representation of BERT-based text encoder to represent the encoded clinical note, i.e.,

$$\mathbf{H} = [\mathrm{BERT}(c_1,\ \boldsymbol{\theta_0}),\ \ldots, \mathrm{BERT}(c_r,\ \boldsymbol{\theta_0})] \quad (2)$$

where $\mathbf{H} \in \mathbb{R}^{s \times d_h}$, s is the sequence length, c_i is the i-th chunked text, and $\boldsymbol{\theta_0}$ is the model parameter of BERT encoder initialized from self-supervised pretraining. We can use the pooled embedding of the hidden representations to represent the clinical note, which is denoted as $\mathbf{h} \in \mathbb{R}^{d_h}$. Similarly, we input the textual label of each task to obtain the task description embeddings with a shared BERT-based text encoder parameterized with $\boldsymbol{\theta_0}$. Specifically, for the t-th task with m_t classes, the initial task embeddings are $\mathbf{T_t} \in \mathbb{R}^{m_t \times d_t}$, where d_t is the embedding dimension. The obtained task embeddings capture the task information and preserve the semantic meaning of classes in each task.

2.3 Task-Conditioned Hypernetworks

Our goal is to utilize task side-information to enable robust patient outcome prediction, especially for those classes with very few instances or that were unseen in the training set. In the joint training of multitask learning, it is easy to suffer from intertask interference. Inspired by the hypernetworks (Ha et al., 2017) that generate model parameters to enable weight-sharing across neural layers, we propose the task-conditioned hypernetworks with semantic task label embeddings to generate the parameters of multitask classification heads for clinical outcomes. The task-conditioned parameter generation can enable task context-aware learning from multiple tasks and share knowledge across different tasks.

Task Embeddings To facilitate the parameter generation conditioned on task information, we further use a bottleneck network to train the task embeddings obtained from the contextualized embeddings of the BERT-based text encoder. The bottleneck network contains two fully-connected layers with ReLU as the activation function (Nair and Hinton, 2010), denoted as:

$$\mathbf{Z_t} = \big(\mathrm{ReLU}(\mathbf{T_t W_1})\big)\mathbf{W_2}, \tag{3}$$

where $\mathbf{W_1} \in \mathbb{R}^{d_t \times d_b}$ and $\mathbf{W_2} \in \mathbb{R}^{d_b \times d_t}$ are weight matrices of linear layers, $\mathbf{Z_t} \in \mathbb{R}^{m_t \times d_t}$ are the output task embeddings, d_b is the hidden dimension that restricts the bottleneck with fewer neurons, and the bias terms are omitted for simplicity. The bottleneck network has been applied in many fields, such as image noise reduction. We use this structure to reduce the potential noise in task classes because the assignment of diagnosis in MIMIC-III does not use a systematic ontology.

Adapter Hypernetwork Inspired by the hypernetwork-generated adapters (Mahabadi et al., 2021) that fine-tune transformer layers, we equip the multitask classification heads with adapter hypernetworks that generate the parameters. Furthermore, we inject the contextual information in the task labels into parameter generation, which provides task-specific classification heads capable of task context-aware learning.

The adapter hypernetwork takes task embeddings of each task as input and generates model parameters written as

$$\mathbf{W_t} = \mathcal{H}(\mathbf{Z_t}),\ \forall\, t = 1, \ldots, k \tag{4}$$

where $\mathbf{W_t} \in \mathbb{R}^{d_h \times m_t}$ is the generated weight parameters for the classification head of the t-th task. We instantiate hypernetworks with simple linear layers. Specifically, the matrices of task embeddings $\mathbf{Z_t}$ are flattened into vectors $\mathbf{v_t} \in \mathbb{R}^{m_t * d_t \times 1}$, projected into the hidden representation space $\mathbb{R}^{m_t * d_h \times 1}$ with nonlinear activation, and reshaped to the size of weight matrices $\mathbb{R}^{m_t \times d_h}$ to act as the parameters of classification heads. The bias parameters are also generated similarly. The adapter hypernetwork conditioned on the task context shares the meta knowledge of tasks and enables task-specific prediction. Thus, the inter-task interference in the multitask setting can be effectively mitigated. Furthermore, the semantic embeddings of task context provide the adapter hypernetwork a good initialization of task classes to generate task-specific parameters, facilitating the meta-learning from limited information for zero- or few-shot scenarios.

2.4 Task-Conditioned Learning Objective

In the joint training framework of multitask learning, the standard method corresponds to manually tuning the weight coefficient α_t. However, this requires human efforts to configure additional hyperparameters when tuning the multitask learning algorithms manually. To avoid this, we propose to use another *random* hypernetwork $\mathcal{G}$ to generate the weight coefficients and make the weighted objective function of the joint multitask learning conditioned on task context. The weight coefficient hypernetwork $\mathcal{G}$ is defined as

$$\beta_t = \mathcal{G}(\mathbf{Z_t}),\ \forall\, t = 1, \ldots, k. \tag{5}$$

This hypernetwork is a multilayer perceptron (MLP), and it takes the task embeddings $\mathbf{Z_t}$ as input and outputs a one-dimensional scalar. Then we apply the softmax function to generate normalized weight probability, i.e.,

$$\alpha_t = \frac{\exp(\beta_t)}{\sum_{t=1}^{k} \exp \beta_t}. \tag{6}$$

In practice, we initialize the weight hypernetwork randomly and freeze the weights during optimization. Hence this approach can be seen as analogous to the technique of random projection (Wójcik and Kurdziel, 2019), which projects high-dimensional points to a lower dimension using a random weight matrix, and which is known to preserve the relationships of the points. Here, however, the role of the projection weight matrix is taken by the weight hypernetwork. In this way, the weight coefficients learned from the task context can preserve the semantic task information and reweigh the joint loss dynamically when the contexts change. We hypothesize that this also adds flexibility to training and regularization of the model, which contributes to the improved performance (see Section 3.5).

3 Experiments

3.1 Datasets and Setup

3.1.1 Datasets

We use the admission-level patient records in the MIMIC-III dataset for experiments. From the vast information in this database, we select the free-text patient notes as the testbed for natural language processing research. We release the source code [1].

Table 1: A statistical summary of datasets and tasks

Dataset	# Samples			# Classes			
	Train	Val.	Test	Readm.	Adm. Type	Diag.	LoS
Discharge	26,245	3,037	3,063	2	3	2,624	10
Early Notes	6,656	743	608	2	3	2,637	10

This study considers four classification tasks, i.e., readmission prediction, diagnosis classification, length-of-stay prediction, and admission type classification. Following the prior work (Huang et al., 2019), we build two datasets, i.e., one extracted from discharge summaries (denoted as `Discharge`) and one with early notes that were created within three days after admission (denoted as `Early Notes`), according to the period of patient admission date and the date when the notes are charted. Table 1 summarizes the number of instances and classes in our extracted datasets. In-hospital death prediction is also an essential task of patient outcome. However, we focus more on readmission, and mortality precludes the possibility of readmission. Thus, we filter out all the in-hospital death cases, which also aligns with the prior work (Huang et al., 2019). We also consider a proxy task of admission type classification that categorizes the clinical notes into "emergency", "elective" and "urgent" (studied in Sec. 3.4).

3.1.2 Tasks

Readmission Prediction The goal of this binary classification task is to predict whether the patient will be readmitted within 30 days of discharge.

Diagnosis Prediction The MIMIC-III dataset has a total of 15,691 classes of diagnoses in the "ADMISSIONS" table. We extract the clinical notes from the "NOTEEVENTS" table and get a total of 2,715 diagnoses in the extracted datasets of discharge and early notes, where frequent diagnoses include pneumonia, congestive heart failure, and sepsis. In our train/val/test split of discharge notes, 2,209 diagnoses appear in the training set, which allows us to test the performance of zero-shot prediction.

Length-of-Stay (LoS) Prediction Following the setting of Harutyunyan et al. (2019), we define the length-of-stay prediction as a 10-class classification problem. Specifically, the duration of stay is divided into ten buckets, i.e., one class for stays less than a day, seven classes for one-to-seven-day stays, one for stays longer than a week but shorter than two, and the last one class for stays over two weeks. A slight difference is that the duration of stay in our definition is calculated during the span between when the patient was admitted to the hospital and when discharged from the hospital.

3.1.3 Baselines

We compare our method, dubbed MT-Hyper, with several multitask learning algorithms on electronic health records and clinical natural language processing. **MT-LSTM (Harutyunyan et al., 2019)** adopts a basic LSTM-based neural network and is jointly trained with several patient outcome prediction tasks. This method is initially designed for modeling time series. We modify it with a word embedding module for modeling clinical text in patients' health records. **MT-BERT (Mulyar and McInnes, 2020)** is a unified clinical NLP model that utilizes the BERT model architecture for text encoding and learns features with multiple clinical task prediction heads simultaneously. It originally consists of eight task heads, while our implementation uses fewer clinical prediction tasks. **MT-RAM (Sun et al., 2021)** proposes a neural module with feature recalibration and aggregation for multitask learning that mitigates the mutual interference of different tasks and refines the feature learning from noisy clinical text.

3.1.4 Settings and Evaluation

We use the Adam optimizer (Kingma and Ba, 2014) to optimize the model and set different learning rates for different text encoders and classification heads. The learning rates of LSTM-based methods range from $5\mathrm{e}^{-5}$ to $1\mathrm{e}^{-3}$. As to BERT-based methods, we use a lower learning rate for both base text encoder and classification heads, ranging from $1\mathrm{e}^{-5}$ to $5\mathrm{e}^{-5}$. Because clinical BERT only has pretrained checkpoints with the BERT base model and the limitation of computing resources, we adopt the BERT base architecture as our text encoder. We use 300D word embedding, and the dimension of

---

[1] Code available at `https://agit.ai/jsx/MT-Hyper`

Table 2: Results of patient outcome prediction from early notes with average score ± standard deviation reported. Bold values indicate the cases when our model obtain the best performance.

Task	Method	Progressive		Ultimate	
		F1	AUC-ROC	F1	AUC-ROC
Readmission	MT-LSTM	51.69 ± 3.72	53.80 ± 1.99	52.36 ± 3.62	56.17 ± 2.23
	MT-BERT	55.52 ± 3.92	56.75 ± 2.56	56.02 ± 4.34	60.85 ± 1.61
	MT-RAM	56.00 ± 3.09	57.26 ± 1.94	56.51 ± 2.80	62.78 ± 1.39
	MT-Hyper	**56.38** ± 2.30	**57.57** ± 1.27	**57.67** ± 2.21	62.41 ± 1.61
Diagnosis	MT-LSTM	8.50 ± 3.39	63.00 ± 0.95	8.69 ± 3.79	62.26 ± 0.91
	MT-BERT	11.63 ± 4.83	64.13 ± 1.30	11.80 ± 5.02	63.24 ± 1.30
	MT-RAM	14.06 ± 1.21	68.58 ± 1.60	14.39 ± 1.41	67.84 ± 1.64
	MT-Hyper	**19.56** ± 1.33	**72.98** ± 0.45	**20.47** ± 1.38	**73.21** ± 0.43
LOS	MT-LSTM	30.19 ± 2.70	75.73 ± 0.92	30.54 ± 2.68	76.50 ± 0.87
	MT-BERT	26.40 ± 4.34	72.60 ± 2.81	27.25 ± 4.08	73.71 ± 2.66
	MT-RAM	26.20 ± 3.08	73.15 ± 3.44	27.08 ± 2.80	74.04 ± 3.31
	MT-Hyper	**33.18** ± 0.91	71.84 ± 1.95	**33.28** ± 1.03	72.23 ± 2.18
Average	MT-LSTM	30.12 ± 3.27	64.18 ± 1.29	30.53 ± 3.36	64.97 ± 1.34
	MT-BERT	31.18 ± 4.36	64.49 ± 2.22	31.69 ± 4.48	65.93 ± 1.86
	MT-RAM	32.09 ± 2.46	66.33 ± 2.33	32.66 ± 2.34	68.22 ± 2.11
	MT-Hyper	**36.37** ± 1.51	**67.47** ± 1.23	**37.14** ± 1.54	**69.28** ± 1.41

LSTM hidden states is also set to 300. We run each algorithm ten times and report the average score and standard deviation. We monitor the validation loss in each trial and set the number of epochs that triggers the early stop mechanism to be 3 for the BERT encoder and 10 for the LSTM encoder.

Our problem definition considers readmission as a binary classification problem, while the rest of the tasks are defined as multi-class classification problems. For readmission prediction, we use weighted F1 and AUC-ROC. For multi-class classification problems, we report micro averaged scores. As we segment lengthy clinical notes into shorter chunks for progressive prediction, we report the results of two types of evaluation scheme, i.e., *progressive* scores that are calculated with the prediction probabilities of note chunks and the *ultimate* scores of a complete admission that are measured by aggregating every prediction on segmented notes. The ultimate scores can also be viewed as a similar strategy to the bagging-based ensemble, in which each single prediction is based on different chunks of an instance and the final prediction is an ensemble of different predictions.

3.2 Main Results

This section presents the main results on prediction at discharge and early prediction. As admission type classification has limited usage in real-world scenarios, we do not include it in the comparison of main results but use it in the discussion of task conditioning.

Early Prediction Table 2 shows the results of patient outcome prediction from early notes. In both progressive and ultimate prediction, our proposed method outperforms three baselines in most cases. The baseline models are relatively stronger in tasks with fewer classes. For example, MT-BERT and MT-RAM produce a relatively good prediction on readmission, while their performance for multi-class diagnosis prediction is much worse than our proposed method. Furthermore, our proposed method can make more stable predictions in most cases according to the standard deviation. Our method outperforms the baselines by a relatively large margin in terms of the average scores of three prediction tasks.

Prediction at Discharge Next, we consider prediction at discharge, while keeping the setup otherwise similar to the early prediction. Discharge notes provide relatively complete information about patient hospital visits. Table 3 shows the prediction results. Our method has comparable readmission prediction performance, superior diagnosis prediction, inferior performance in the length-of-stay prediction, and relatively better overall performance measured by average score than other baselines. Besides, our method has smaller values of standard deviation in most cases when making a prediction at discharge.

3.3 Zero-shot Diagnosis Prediction

We study a more challenging scenario of zero-shot diagnosis prediction. In the extracted datasets, among the total 2,624 diagnoses of discharge, 2,209 diagnoses appear in the training set, i.e., 415 types of diagnoses are unseen in the evaluation data. We adopt this sampled setting to test the performance of zero-shot diagnosis prediction. Table 4 shows the AUC-ROC scores of progressive and ul-

Table 3: Results of patient outcome prediction at discharge with average score ± standard deviation reported. Bold values indicate the cases when our model obtain the best performance.

Task	Method	Progressive		Ultimate	
		F1	AUC-ROC	F1	AUC-ROC
Readmission	MT-LSTM	60.54 ± 1.38	60.39 ± 1.22	63.17 ± 1.64	68.46 ± 1.93
	MT-BERT	64.86 ± 1.08	64.78 ± 1.15	68.66 ± 3.17	76.33 ± 2.88
	MT-RAM	65.92 ± 1.86	65.88 ± 1.34	70.41 ± 2.67	77.49 ± 2.30
	MT-Hyper	**66.41** ± 0.77	**66.08** ± 0.74	69.66 ± 1.52	76.93 ± 1.11
Diagnosis	MT-LSTM	9.08 ± 1.60	66.51 ± 0.98	11.04 ± 2.74	64.85 ± 1.01
	MT-BERT	9.75 ± 1.82	69.19 ± 6.22	10.80 ± 2.83	68.04 ± 7.98
	MT-RAM	10.26 ± 1.99	67.38 ± 0.79	11.39 ± 2.99	65.70 ± 0.92
	MT-Hyper	**10.28** ± 0.25	**76.93** ± 0.73	**12.99** ± 1.57	**79.07** ± 1.07
LOS	MT-LSTM	31.09 ± 0.93	76.77 ± 0.67	32.57 ± 1.29	79.33 ± 1.19
	MT-BERT	31.37 ± 1.23	78.27 ± 0.90	34.91 ± 0.31	82.13 ± 0.26
	MT-RAM	31.29 ± 1.91	76.14 ± 4.48	32.29 ± 2.95	77.69 ± 5.34
	MT-Hyper	31.33 ± 0.38	73.76 ± 1.31	31.08 ± 0.21	73.01 ± 1.28
Average	MT-LSTM	33.57 ± 1.30	67.89 ± 0.96	35.59 ± 1.89	70.88 ± 1.38
	MT-BERT	35.33 ± 1.37	70.75 ± 2.76	38.13 ± 2.10	75.50 ± 3.71
	MT-RAM	35.82 ± 1.92	69.80 ± 2.20	38.03 ± 2.87	73.63 ± 2.86
	MT-Hyper	**36.01** ± 0.47	**72.26** ± 0.93	37.91 ± 1.10	**76.34** ± 1.15

timate diagnosis prediction on zero-shot diagnosis categories. The results indicate that the baseline models make terribly incorrect predictions with AUC-ROC scores lower than the random guess. These baselines cannot learn meta-knowledge of diagnoses, making them overfitting on seen diagnoses in the training set. In contrast, our proposed MT-Hyper can capture the task context and semantic diagnosis information. Therefore, our method is more generalizable on unseen diagnoses and achieves satisfactory performance under the challenging zero-shot setting.

Table 4: Results (AUC-ROC) of zero-shot diagnosis prediction on unseen diagnosis results with average score ± standard deviation reported. Bold values indicate the cases when our model obtain the best performance.

Dataset	Method	Progressive	Ultimate
Discharge	MT-LSTM	11.00 ± 2.31	9.74 ± 1.55
	MT-BERT	10.47 ± 0.76	9.00 ± 0.36
	MT-RAM	11.40 ± 1.44	10.37 ± 1.25
	MT-Hyper	**64.06** ± 2.02	**68.33** ± 2.76
Early	MT-LSTM	8.65 ± 1.19	8.47 ± 1.00
	MT-BERT	8.24 ± 0.79	8.20 ± 0.74
	MT-RAM	14.25 ± 3.11	14.12 ± 3.12
	MT-Hyper	**62.04** ± 1.13	**63.76** ± 1.03

3.4 Semantic Relation of Task Embeddings

We study the semantic relation of task embeddings to verify if task conditioning uses task information. We include the admission type classification as an auxiliary task and train four tasks jointly to achieve this goal. We filter out “newborn" and “death" and define the admission type classification as a three-way classification task with admission types of “emergency", “elective" and “urgent". The classification task categorizes the type of admission for each clinical note. It may have no practical usage in clinical support. However, it is a useful auxiliary task for clinical note understanding. Thus, we choose this task as an auxiliary task to help study the semantic relation of task embeddings. We obtain the task embeddings from the trained model and aggregate embeddings of each task into a vector via mean pooling. To visualize task vectors in low dimensional space, we apply principal component analysis to reduce the dimension of task vectors. Firstly, we plot task label embeddings of readmission, admission type and length-of-stay and together with the first 50 diagnoses. Figure 3a shows that different label embeddings are relatively far apart from each other, and the two tasks that are similar to each other, readmission prediction and admission type prediction, appear close to each other. We further zoom in to the embeddings of length-of-stay task labels in Figure 3b. However, here we cannot find any clear interpretable pattern. We hypothesize that the prediction of length-of-stay would benefit from clinical document understanding and numerical reasoning over the ordinal categories, and the hypernetwork-based architecture with semantic task embeddings does not effectively capture this ordinal relation of different lengths of stay. This may partially explain why our method did not provide advantage over other methods on the length-of-stay prediction.

3.5 Ablation Study

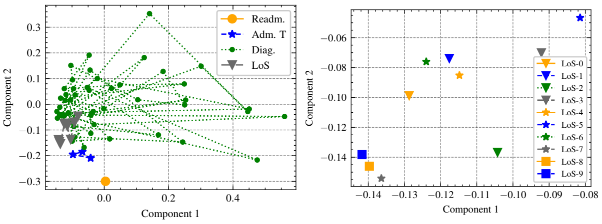
Our proposed method consists of two hypernetworks, i.e., adapter hypernetwork for task-specific parameter generation and weight hypernetwork (WH) for task weight coefficient generation. We conduct an ablation study on the usage of weight

Table 5: Ablation study on patient outcome prediction from early notes with three tasks jointly trained

Task	Method	Progressive		Ultimate	
		F1	AUC-ROC	F1	AUC-ROC
Readmission	without WH	56.74 ± 1.26	57.28 ± 1.17	57.94 ± 1.53	61.56 ± 1.79
	with WH	56.38 ± 2.30	**57.57** ± 1.27	57.67 ± 2.21	**62.41** ± 1.61
Diagnosis	without WH	18.90 ± 2.06	69.26 ± 0.33	19.91 ± 2.29	69.58 ± 0.47
	with WH	**19.56** ± 1.33	**72.98** ± 0.45	**20.47** ± 1.38	**73.21** ± 0.43
LOS	without WH	33.59 ± 0.36	71.14 ± 1.25	33.85 ± 0.34	71.44 ± 1.32
	with WH	33.18 ± 0.91	**71.84** ± 1.95	33.28 ± 1.03	**72.23** ± 2.18
Average	without WH	36.41 ± 1.23	65.89 ± 0.92	37.23 ± 1.39	67.53 ± 1.19
	with WH	36.37 ± 1.51	**67.46** ± 1.22	37.14 ± 1.54	**69.28** ± 1.41

Table 6: Ablation study on patient outcome prediction at discharge with three tasks jointly trained

Task	Method	Progressive		Ultimate	
		F1	AUC-ROC	F1	AUC-ROC
Readmission	without WH	65.42 ± 0.88	65.21 ± 0.73	68.11 ± 1.46	75.99 ± 0.37
	with WH	**66.41** ± 0.77	**66.08** ± 0.74	**69.66** ± 1.52	**76.93** ± 1.11
Diagnosis	without WH	10.05 ± 0.87	75.13 ± 1.12	12.68 ± 1.39	78.54 ± 1.56
	with WH	**10.28** ± 0.25	**76.93** ± 0.73	**12.99** ± 1.57	**79.07** ± 1.07
LOS	without WH	31.20 ± 0.31	73.47 ± 0.59	30.82 ± 0.34	72.62 ± 0.79
	with WH	**31.33** ± 0.38	**73.76** ± 1.31	**31.08** ± 0.21	**73.01** ± 1.28
Average	without WH	35.56 ± 0.69	71.27 ± 0.81	37.20 ± 1.06	75.72 ± 0.91
	with WH	**36.01** ± 0.47	**72.26** ± 0.93	**37.91** ± 1.10	**76.34** ± 1.15



(a) Task embedding

(b) LOS label embeddings

Figure 3: Visualization of task embeddings with dimension reduced by PCA

hypernetwork. Table 5 and Table 6 show the results of patient outcome prediction from early notes and prediction at discharge respectively. The weight hypernetwork dynamically adjusts the weight coefficients of joint loss and contributes to the performance improvement in most cases. MT-Hyper with weight hypernetwork has better predictive scores in 10 out of 16 evaluation metrics on patient outcome prediction from early notes. As for patient outcome prediction at discharge, MT-Hyper with weight hypernetwork outperforms its counterpart without weight hypernetwork.

4 Related Work

Patient outcome prediction, including automatic coding (Friedman et al., 2004; Yan et al., 2010), patient severity (Naemi et al., 2020), in-hospital mortality, decompensation, length of stay and phenotyping (Harutyunyan et al., 2019), can empower health practitioners to make better clinical decision. Many studies have been conducted to investigate the usability of deep learning for accurate prediction using time-series, clinical text, or multimodal data. Zufferey et al. (2015) compared different multi-label learning algorithms. Che et al. (2015) proposed a deep phenotyping model regularized on the categorical structure of the medical ontology. Zhang et al. (2019) proposed an attention-based LSTM network to model the disease progression in a time-aware fashion. Similarly, Men et al. (2021) extended the LSTM model to be attention- and time-aware for multi-disease prediction. Convolutional neural networks have also been deployed in this field. For example, (Bardak and Tan, 2021) developed CNN-based networks for multimodal learning from patient records. Clinical notes as an essential modality of EHRs have also been studied intensively. Ghassemi et al. (2014) extracted topic features from free-text patient data for mortality prediction. Mulyar and McInnes (2020) extracted clinical information from clinical documents via fine-tuning large-scale pretrained language models.

Multitask learning learns multiple tasks simultaneously with a shared representation and transfers the domain knowledge in related tasks to improve the learning capacity (Caruana, 1997). It has also attracted much attention from the research community of machine learning and natural language processing for healthcare. Mahajan et al. (2020) studied semantic similarity degrees of clinical notes by a multitask learning method with iterative data selection. Sun et al. (2021) proposed a multitask learning framework with feature calibration for medical coding. Several multitask learning schemes have also been developed for learning from clinical time-series (Harutyunyan et al., 2019). Hypernetworks (Ha et al., 2017) that generate weights for other networks have also been adopted to solve the learning problem with multiple tasks, for example, the Hyperformer that generates the parameters of adapter modules for fine-tuning NLP task streams (Mahabadi et al., 2021). Inspired by this work, our work starts from a different setting that learns different tasks simultaneously, utilizes the semantic information of clinical predication tasks, and balances the learning objective via task conditioning.

5 Conclusion

Patient hospitalization is a complex process, which needs the learning algorithm to model multiple risk indicators. Multitask learning can jointly learn from patient records to empower clinical decision-making. To address the challenges of inter-task interference and poor generalizability to unseen labels in recent multitask learning frameworks for healthcare, we propose a hypernetwork-guided multitask learning method that learns from the task context and generates task-specific parameters for effective multitask prediction on patient outcomes. The proposed method incorporates contextualized language representations to encode clinical notes and capture the task context via the semantic embeddings of tasks. The hypernetwork-guided multitask learning framework enables task-conditioned learning and balances the joint learning objective across different tasks. Experimental studies on the real-world MIMIC-III dataset show our proposed method achieves better performance on early prediction and more robust performance on zero-shot diagnosis prediction.

Limitations

The proposed task-conditioned multitask learning method requires to obtain task embeddings from the semantic description.

Acknowledgments

We acknowledge the computational resources provided by the Aalto Science-IT project and CSC - IT Center for Science, Finland. This work was supported by the Academy of Finland (Flagship programme: Finnish Center for Artificial Intelligence FCAI, and grants 336033, 352986) and EU (H2020 grant 101016775 and NextGenerationEU).

References

Batuhan Bardak and Mehmet Tan. 2021. Improving clinical outcome predictions using convolution over medical entities with multimodal learning. *Artificial Intelligence in Medicine*, 117:102112.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(41-75).

Zhengping Che, David Kale, Wenzhe Li, Mohammad Taha Bahadori, and Yan Liu. 2015. Deep computational phenotyping. In *Proceedings of ACM SIGKDD*, pages 507–516.

Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor AI: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*, pages 301–318. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*.

Carol Friedman, Lyudmila Shagina, Yves Lussier, and George Hripcsak. 2004. Automated encoding of clinical documents based on natural language processing. *JAMIA*, 11(5):392–402.

Marzyeh Ghassemi, Tristan Naumann, Finale Doshi-Velez, Nicole Brimmer, Rohit Joshi, Anna Rumshisky, and Peter Szolovits. 2014. Unfolding physiological state: Mortality modelling in intensive care units. In *Proceedings of ACM SIGKDD*, pages 75–84.

David Ha, Andrew Dai, and Quoc V Le. 2017. Hypernetworks. In *Proceedings of ICLR*.

Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. 2019. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1):1–18.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. ClinicalBERT: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*.

Shaoxiong Ji, Shirui Pan, and Pekka Marttinen. 2021. Medical code assignment with gated convolution and note-code interaction. In *Findings of ACL-IJCNLP*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

Irene Li, Jessica Pan, Jeremy Goldwasser, Neha Verma, Wai Pan Wong, Muhammed Yavuz Nuzumlalı, Benjamin Rosand, Yixin Li, Matthew Zhang, David Chang, et al. 2021. Neural natural language processing for unstructured data in electronic health records: a review. *arXiv preprint arXiv:2107.02975*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of ACL-IJCNLP*.

Diwakar Mahajan, Ananya Poddar, Jennifer J Liang, Yen-Ting Lin, John M Prager, Parthasarathy Suryanarayanan, Preethi Raghavan, and Ching-Huei Tsou. 2020. Identification of semantically similar sentences in clinical notes: Iterative intermediate training using multi-task learning. *JMIR Medical Informatics*, 8(11):e22508.

Lu Men, Noyan Ilk, Xinlin Tang, and Yuan Liu. 2021. Multi-disease prediction using lstm recurrent neural networks. *Expert Systems with Applications*, 177:114905.

Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. 2016. Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records. *Scientific Reports*, 6(1):1–10.

Andriy Mulyar and Bridget T McInnes. 2020. MT-clinical BERT: scaling clinical information extraction with multitask learning. *arXiv preprint arXiv:2004.10220*.

Amin Naemi, Marjan Mansourvar, Thomas Schmidt, and Uffe Kock Wiil. 2020. Prediction of patients severity at emergency department using n and ensemble learning arx. In *BIBM*, pages 2793–2799. IEEE.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of ICML*.

Benjamin Shickel, Patrick James Tighe, Azra Bihorac, and Parisa Rashidi. 2017. Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. *IEEE JBHI*, 22(5):1589–1604.

Wei Sun, Shaoxiong Ji, Erik Cambria, and Pekka Marttinen. 2021. Multitask Recalibrated Aggregation Network for Medical Code Prediction. In *Proceedings of ECML-PKDD*.

Betty van Aken, Jens-Michalis Papaioannou, Manuel Mayrdorfer, Klemens Budde, Felix Gers, and Alexander Loeser. 2021. Clinical outcome prediction from admission notes using self-supervised knowledge integration. In *Proceedings of EACL*, pages 881–893.

Piotr Iwo Wójcik and Marcin Kurdziel. 2019. Training neural networks on high-dimensional data using random projection. *Pattern Analysis and Applications*, 22(3):1221–1231.

Yan Yan, Glenn Fung, Jennifer G Dy, and Romer Rosales. 2010. Medical coding classification by leveraging inter-code relationships. In *Proceedings of ACM SIGKDD*, pages 193–202.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE TKDE*.

Yuan Zhang, Xi Yang, Julie Ivy, and Min Chi. 2019. ATTAIN: Attention-based Time-Aware LSTM Networks for Disease Progression Modeling. In *Proceedings of IJCAI*, pages 4369–4375.

Damien Zufferey, Thomas Hofer, Jean Hennebert, Michael Schumacher, Rolf Ingold, and Stefano Bromuri. 2015. Performance comparison of multi-label learning algorithms on clinical data for chronic diseases. *Computers in Biology and Medicine*, 65:34–43.

A Kind Introduction to Lexical and Grammatical Aspect, with a Survey of Computational Approaches

Annemarie Friedrich[1] **Nianwen Xue**[2] **Alexis Palmer**[3]
[1]Bosch Center for Artificial Intelligence, Renningen, Germany
[2]Brandeis University [3]Dept. of Linguistics, University of Colorado Boulder
annemarie.friedrich@de.bosch.com xuen@brandeis.edu
alexis.palmer@colorado.edu

Abstract

Aspectual meaning refers to how the internal temporal structure of situations is presented. This includes whether a situation is described as a state or as an event, whether the situation is finished or ongoing, and whether it is viewed as a whole or with a focus on a particular phase. This survey gives an overview of computational approaches to modeling lexical and grammatical aspect along with intuitive explanations of the necessary linguistic concepts and terminology. In particular, we describe the concepts of stativity, telicity, habituality, perfective and imperfective, as well as influential inventories of eventuality and situation types. Aspect is a crucial component of semantics, especially for precise reporting of the temporal structure of situations, and future NLP approaches need to be able to handle and evaluate it systematically.

1 Introduction

Lexical and grammatical aspect play essential roles in semantic interpretation (Smith, 2003), and yet even state-of-the-art natural language understanding (NLU) systems do not address these linguistic phenomena systematically (Metheniti, 2022). Consider this example: an NLU-based personal assistant, noticing the boarding time of a flight, tells a passenger (who is still shopping at the airport) "You miss flights" (i.e. on a regular basis) instead of "You are missing the flight" (now!). The traveler might misinterpret this utterance as chit-chat and indeed miss the flight. Aspectual encoding errors impair fluid and correct communication.

While there has been a notable amount of work on modeling lexical and grammatical aspect in the computational linguistics community (Moens and Steedman, 1988; Siegel and McKeown, 2000b; Friedrich et al., 2016; Kober et al., 2020), this area is still a niche in natural language processing (NLP). In this paper, we survey the state of research in this area and argue that a good computational understanding of lexical and grammatical aspect is paramount for capturing temporal information signaled by linguistic encoding. We thereby add to the on-going discussion in the NLP community about what is required to achieve true natural language understanding (Bender and Koller, 2020; Dunietz et al., 2020; Trott et al., 2020).

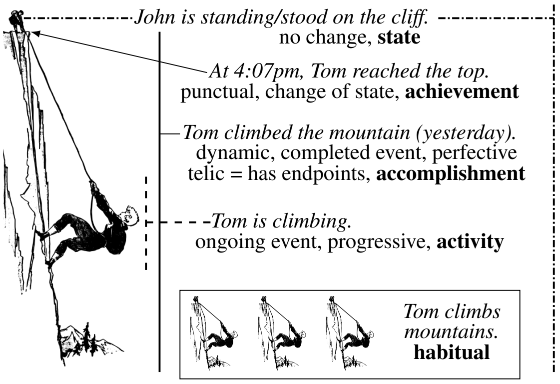


Figure 1: **Aspect** is like the camera lens of language, the device by which we focus on particular phases of a situation (Vendler, 1957; Smith, 1997).

When describing a situation, as exemplified in Figure 1, producers of language have options for how to depict the situation and which subparts of it to highlight. Just as a cinematographer uses focus to highlight certain elements of a scene and thus bring them to the attention of the viewer, different aspectual choices focus on different subparts of the situation at hand, leaving the rest as background (analogy due to Smith, 2003). For example, many situations can be construed as either *foregrounded* events ("Tom climbed the mountain"), moving narrative time forward, or as states ("John stood on the cliff"), which often function as the *background* in a narrative or explanation. Aspectual choices may trigger hard inferences, softer implicatures, or simply expectations, having a non-negligible impact on the reader or listener. One aim of this survey is to clarify the sometimes complex linguistic work on aspect, giving NLP researchers and practitioners a

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 599–622
May 2-6, 2023 ©2023 Association for Computational Linguistics

Dataset	Reference	Language	Size	Annotated Categories
Asp-Ambig	Friedrich and Palmer (2014a)	EN	7875 clauses	stativity
SitEnt	Friedrich et al. (2016)	EN	40,000 clauses	situation entities, stativity, habituality
Tense-Europarl	Loáiciga and Grisot (2016)	EN/FR	435 verb phrases	boundedness (in sentence context)
MASC-telicity	Friedrich and Gateva (2017)	EN	1863 clauses	telicity
Captions	Alikhani and Stone (2019)	EN	2600 captions	stative, durative, telic/atelic, punctual
SdeWac-Aspect	Egg et al. (2019)	DE	4200 clauses	stativity, durative vs. punctual, boundedness
DIASPORA	Kober et al. (2020)	EN	927 utterances	stative, telic, atelic
UDS	Gantt et al. (2022)	EN	16,624 sentences; granularity varies	stativity, some telicity, durativity, other event properties

Table 1: Available **Datasets** labeled with aspectual information.

conceptual toolkit, plus a glossary of aspect-related terminology (Appendix A). The second aim is to survey computational work on modeling aspect.

Aspectual categories are *semantic* notions that are conveyed through a variety of mechanisms (Section 2). Fundamental aspectual distinctions (Section 3) may be part of the lexical meaning of a verb and are also influenced by context. These distinctions drive some influential inventories of eventuality types (Section 3.3). Grammatical aspect (Section 4) is marked morphologically in some (e.g., Slavic) languages, or via the choice or absence of tense-aspect features in languages such as English. Section 5 explains the idea of situation types, aspectual distinctions at the clause level.

As a consequence of the often distributed or even implicit encoding of aspect, computational models need to consider a variety of lexical and grammatical features as well as discourse and pragmatic information. While some NLP systems trained on end-to-end datasets may get some of these cases right, we are not aware of a system treating such problems in a principled manner. A recent study finds that transformer-based embeddings are useful for classifying clausal aspect (Metheniti, 2022), but it remains unclear just how these models learn or incorporate aspectual information. This survey focuses on modeling aspectual information as it is annotated in the datasets listed in Table 1.

In Section 6, we discuss potential ways of moving the field forward. We argue that explicit modeling or evaluation of aspectual categories should become a routine perspective in future NLP: getting aspect right has high stakes for applications such as temporal question answering, machine translation, or computer-aided language learning.

2 Where Does Aspect Live?

Comrie (1976) notes that there is a terminological and conceptual confusion around tense and aspect. This is at least partially because the expression of temporal location is intertwined morphologically with aspect in many languages of the world (Smith, 1997), so the traditional grammatical terminologies of individual languages often introduce tense/aspect combinations simply as tenses. Tense is *deictic* and locates the time of a situation relative to the utterance or relative to another situation. Aspect, in contrast, refers to situation-internal time.

Aspectual distinctions are *covert* categories (Whorf, 1945; Smith, 1997), expressing particular temporal semantic concepts in a way that does not have a one-to-one mapping to lexical, grammatical or syntactic categories (Fillmore, 1969). For example, the English Progressive[1] often indicates an ongoing situation without focusing on its endpoints ("I am eating an apple"), but it also signals future events ("I am going to Paris"). The grammatical form of such tense-aspect combinations can often be automatically detected with high accuracy (Ramm et al., 2017; Myers and Palmer, 2019). This type of aspect, encoded in the tense system of a language, is different from *semantic* concepts which refer to how the internal structure of a situation is presented; this survey focuses on the latter.

So where does *covert, semantic* aspect live? Aspectual meaning is compositional, composed by the verb's inherent meaning, its arguments, its tense, any morphological aspect markers, some prepositional phrases, and the adverbs of the sentence (Verkuyl, 1972; Mourelatos, 1978; Smith, 1997). Eventuality and situation type inventories make use of several dimensions of aspect to distinguish between different types of temporal structures. *Lexical aspect* refers to information contained in the meaning of verbs (or verb senses), with the consequence that their values often can be determined

---

[1] Following Comrie (1976), we use initial capitals for the names of language-particular categories and lower case for language-independent semantic distinctions.

only when observing how the verb interacts with its clausal context. A second dimension, *grammatical* or *viewpoint* aspect, refers to whether there is a focus on part of the situation or whether the situation is viewed as a whole or even as repeating. Grammatical aspect and situation types are both semantic categorizations at the sentence/clause level (e.g., Vendler, 1957; Verkuyl, 2005; Smith, 2003).

3 Inherent Lexical Aspect

Here we explain two fundamental notions related to inherent *lexical aspect*: *stativity* refers to whether a situation is viewed as an event or state, and *telicity* refers to whether an endpoint is visible in a clause.

3.1 Stativity

The most fundamental distinction made in the hierarchies of eventuality types is that between *states* ("love," "own") and *events* ("run," "buy"). In contrast to *dynamic* predicates, *stative* verbs entail no change (Filip, 2012). States obtain in time but they do not take time; events occur, happen, or take place (Smith, 1997). Events may have a beginning and/or an end (Comrie, 1976), and they are often *durative*, i.e., conceived as lasting for a certain period of time. Some events are dynamic situations describing a change of state such as "John reached the goal"; these are perceived as *punctual*. For punctual, single-stage events which cause no change of state, Smith (1997) adds the situation type *semelfactives*, for verbs like "knock," "flash," or "blink." While dynamic in nature, semelfactives return to their initial state at their end and often occur as *iteratives* (Filip, 2012). The stativity distinction pertains to verb senses, not verb types. For example, the verb type "make" has both a dynamic (1a) and a stative (1b) sense.

(1) (a) She is **making** a cake. (*dynamic*)
(b) She **makes** a great host. (*stative*)

In theory, the analysis of stativity could be applied to inventories of verb senses such as WordNet (Fellbaum, 2010). In practice, most computational works focus on classifying verbs in context.

Early computational approaches. The dynamic-stative distinction is at the heart of early approaches to computational modeling of tense and aspect. In the PUNDIT system for temporal information processing (Passonneau, 1988a), the lexical aspect of verbs in tensed clauses can be read from the system's output: transition events use `become`, processes use `do`; other cases signal states. In the context of lexicon induction, Klavans and Chodorow (1992a) suggest representing event structure of a verb as its *degree of stativity*, which is estimated by the proportion of occurrences in a corpus that are in the Progressive. Brent (1991a) also presents a program for identifying stativity using syntactic indicators: verbs that occur frequently with the Progressive or that combine with rate adverbs (e.g. "quickly", "slowly") are usually dynamic. Both works evaluate by manually inspecting system output.

Datasets. Siegel (1999) presents a manually annotated dataset of 739 training and 739 test clauses taken from medical discharge summaries, covering 222 different verb types. With the aim of reproducing and extending their work, Friedrich and Palmer (2014a) have three annotators label 7,875 clauses from MASC (Ide et al., 2008), marking the clause's main verb as *stative* or *dynamic*.If annotators see both readings, they may select both. The dataset has later been extended to cover 30,333 clauses from MASC and 10,607 clauses from Wikipedia (Friedrich et al., 2016). Kober et al. (2020) introduce the DIASPORA dataset, in which 927 utterances from a corpus of human-human phone conversations (Brennan et al., 2013) are labeled with whether the first verb phrase of each utterance is stative or dynamic. The dataset spans 69 different verb types. Chen and Palmer (2022) produce 292 contrast sets for stativity in English and outline strategies for converting between stative and dynamic expressions of the same situation. For more datasets that annotate stativity as one feature among others, see Section 3.3.

Modeling.[2] Siegel (1999, 1997) and Siegel and McKeown (2000b) propose a machine-learning approach to classifying stativity that describes each verb occurrence exclusively using corpus-based statistics of the corresponding verb type. The verb-type based indicators are normalized counts that reflect, for instance, how often the verb co-occurs with the past tense, in the perfect, or in negated form (for a full list, see Appendix D.1). Using these features, they train logistic regression models, decision trees, and genetic programming algorithms.

Friedrich and Palmer (2014a) compare these corpus-based *linguistic indicators* to instance-based syntactic-semantic features representing the clausal context, e.g., the part-of-speech tag of the

---

[2] Appendix B summarizes computational systems and approaches to modeling aspect.

verb, tense, voice, and WordNet information (Fellbaum, 2010) for the verb's arguments. Using the LCS database (Dorr, 2001) and a procedure described by Dorr and Olsen (1997), they also construct three seed sets with verb types that either occur exclusively as *dynamic*, only as *stative*, or can take both aspects. Based on these seed sets and a pre-trained syntactic distributional model (Thater et al., 2010), similarity values are computed and used as additional features. Kober et al. (2020) input sums of non-contextualized word embeddings (Mikolov et al., 2013) for the clause's main verb as well as selected context words to a logistic regression classifier. Overall, they find that using the local context in the form of a word window outperforms the verb-type only classifier, but that feeding in the entire sentence hurts. Metheniti (2022) also classify stativity (called "duration" in their work) by fine-tuning various transformer models and classification layers. They find BERT (Devlin et al., 2019) models work better than RoBERTA (Liu et al., 2019), AlBERT (Lan et al., 2019) and XLNet (Yang et al., 2019).

3.2 Telicity

The second important distinction related to inherent lexical aspect is that of *telicity*. The term *telic*, derived from Greek *télos* (goal) was introduced by Garey (1957). In his definition, telic verb senses have a built-in goal: when that goal is reached, a change of state occurs and the event is complete (Smith, 1997). Telicity is also sometimes referred to as *boundedness* (e.g., by Loáiciga and Grisot, 2016). As illustrated by (2), telicity is a feature of the entire clause (Verkuyl, 2005).

(2) (a) He was swimming in the lake. *(atelic)*
(b) He was swimming across the lake. *(telic)*

When a telic verb is used in the *imperfective* as in (3a), the arrival or nonarrival at the goal is hidden (Garey, 1957). If the same verb is applied in the *perfective* as in (3b), it means that the goal has been reached at the time of reference. Hence, if (3a) is true at some particular point in time, it cannot be the case that (3b) is true at the same point in time.

(3) (a) John was recovering. *(telic, imperfective)*
(b) John has recovered. *(telic, perfective)*

In contrast, *atelic* verbs do not have to wait for a goal for their realization; they are realized as soon as they begin. If an atelic verb is used in imperfective form as in (4a), we can infer that the sentence in perfective form (4b) is also true.

(4) (a) Sue was singing. *(atelic, imperfective)*
(b) Sue has sung. *(atelic, perfective)*

Datasets. Siegel and McKeown (2000b) describe a small dataset annotated for *completedness*, i.e, telicity. Their training and test sets each consist of approximately 300 clauses taken from 10 novels and covering 204 different dynamic verbs. Friedrich and Gateva (2017) manually annotate 1863 clauses taken from MASC (Ide et al., 2008) for telicity. The Captions dataset (Alikhani and Stone, 2019) marks telicity in several image caption corpora. They find that the proportion of telic verbs ranges from 6% to 59% across corpora, with atelic descriptions prevailing in almost all corpora. In the DIASPORA dataset (Kober et al., 2020), in each utterance, the first VP is annotated with its *predicational* aspect (stative, telic, or atelic).

Modeling. Siegel and McKeown (1996, 2000b) propose a machine-learning approach to classifying telicity of verb occurrences using their above-described corpus-based linguistic indicators. For example, verbs frequently occurring in the Progressive are likely atelic. Leveraging these corpus-based *linguistic indicators*, Friedrich and Gateva (2017) integrate syntactic-semantic contextual features into their logistic regression model. They also leverage additional silver standard training data in the form of projected annotations from the English-Czech InterCorp (Čermák and Rosen, 2012; Rosen and Vavřín, 2012). Their approach, using the machine-readable Czech dictionary Vallex (Žabokrtský and Lopatková, 2007), is based on the assumption that most perfective Czech verbs are translated using telic verb constructions, and that most imperfective verbs correspond to atelic constructions. Loáiciga and Grisot (2016) create an automatic classifier similar to that of Friedrich and Gateva (2017) to classify *boundedness* of French verbs, i.e., whether the endpoint of an event has occurred or not. They show that this is useful for picking the correct tense in French translations of the English Simple Past. Several more recent studies have shown that distributional and neural models can be trained to predict telicity as annotated in available datasets (Kober et al., 2020; Metheniti et al., 2021; Metheniti, 2022). BERT-style models perform well on existing telicity datasets (with larger models outperforming smaller models), yet it is still unclear how or whether they actually capture aspect. Metheniti (2022) observe that models are always highly confident in their predictions, re-

Type	*dynamic*	*durative*	*telic*	**Examples**
state	-	+	-	know the answer, love Mary
activity	+	+	-	laugh, stroll in the park, swim
accomplishment	+	+	+	build a house, walk to school
achievement	+	-	+	win a race, recognize
semelfactive	+	-	-	tap, knock, hiccup, tap, wink

Table 2: Eventuality types (Vendler, 1957; Smith, 1997).

gardless of their accuracy (a common phenomenon for neural models). We are not aware of any probing or explainability studies in this area yet, and how to obtain better calibrated models is an open research question.

3.3 Eventuality Types

We now explain some influential inventories of *eventuality types*, which are ontologies of the temporal structures of events and/or states. Table 2 shows the set of situation types introduced by Vendler (1957): *state*, *activity*, *accomplishment*, and *achievement*. There are other similar schemata (Kenny, 1963; Mourelatos, 1978, more details in Appendix C.1); we use Vendler's terminology here.

States are inherently stative and atelic. They are *durative* as they usually extend in time, though this time period may be very short as in (5).

(5) He was very quiet for two seconds.

The other three Vendler classes are all dynamic. They differ in whether they have a built-in endpoint, and in whether a clearly defined process leads up to this endpoint. *Activities* as in (6), consisting entirely of a process, use atelic verbs. *Accomplishments* as in (7) consist of a process that leads up to a built-in terminal point. Similarly, *achievements* as in (8) have an endpoint including a change of state, but the verb meaning does not include a process leading up to this point. Smith (1997) adds the *semelfactive type* (see Section 3.1).

(6) Mary was laughing. *(activity)*

(7) Mary wrote a letter. *(accomplishment)*

(8) He arrived at the station. *(achievement)*

Moens and Steedman (1988) work with aspectual profiles of sentences (for details see Appendix C.1) that are classified by making reference to a so-called *nucleus*. For example, "John built a house" consists of the *preparatory phase* (the house is being built), a *culmination point* (the moment at which it is completed), and a *consequent state* (the house is complete). The process of **aspectual coercion** shifts aspectual verb types based on their arguments or other aspectual operators such as adverbials. When used with a predicate whose lexical entry corresponds to a culminated process, the English Progressive strips off the culmination point and makes visible only the preparatory process as in "John was running a mile."[3]

Datasets. Zarcone and Lenci (2008) create a dataset of 3129 occurrences of 28 Italian verbs manually annotated for Vendler-style event types. Keelan (2012) works on an eight-way classification task for categories based on Leech's (1971) classes (see Appendix C.2). More recently, in the context of the Richer Event Description (RED) annotation scheme (Ikuta et al., 2014; O'Gorman et al., 2016; O'Gorman et al., 2021), the annotation of events with finer-grained Vendler-style situation types has been proposed (Croft et al., 2016). Falk and Martin (2016) select 167 frequent French verbs from a lexical resource (François et al., 2007), and label the corresponding 1199 entries ("readings") with eight aspectual classes similar to those of Mourelatos (1978) as listed in Appendix C.1. Their classification task is finer-grained than labeling verb types, but coarser-grained than clause-level labeling. Egg et al. (2019) annotate 4200 clauses from the German SdeWac (Faaß and Eckart, 2013) with the features stative, durative vs. punctual, and boundedness. With the aim of improving zero-shot image to verb recognition, Zellers and Choi (2017) crowdsource a dataset of 1710 verb templates (such as "put up") for 1203 different verbs annotated with the four Vendler categories. The Caption dataset (Alikhani and Stone, 2019) annotates image captions with the features stative, durative, punctual, telic, and atelic.

Modeling. Zarcone and Lenci (2008) train a maximum entropy classifier that uses adverbial, morphological, and syntactic features, as well as features capturing argument structure. Keelan (2012) uses an SVM (Cortes and Vapnik, 1995) with features similar to the linguistic indicators of Siegel and McKeown (2000b). Zellers and Choi (2017) use GloVe embeddings (Pennington et al., 2014) to represent the input verb. They concatenate this embedding with a phrase embedding of the verb's dictionary definition computed using a recurrent neural network, and add a linear layer on top. Hermes et al. (2015) induce Vendler classes for German verb types, using Siegel-style distributional

---

[3] Thus offering an elegant solution to the *imperfective paradox* (Dowty, 1979; Lascarides, 1991).

features extracted from dependency-parsed corpus data (3000 sentences per verb type) and SVMs for type-level classification. Hermes et al. (2018) compare the above framework to shallow distributional vectors considering only co-occurring word types for each verb. Finally, Gantt et al. (2022) propose a generative model for event types (and other information) fitted from English data annotated with a range of aspectual and other properties. The resulting 4-class event type inventory closely resembles Vendler classes in the distinctions it draws.

The main obstacle to systematically comparing the works mentioned in this section is that their authors make differing choices for the granularity of both the annotation scheme and the computational modeling, and conclusions drawn from the studies depend on these choices. For better comparability, we need better benchmarking.

4 Grammatical Aspect / Viewpoint

Phenomena treated as *grammatical aspect* or *viewpoint* (Smith, 1997) take different views (the entire situation vs. a part of it), or signal recurrence.

4.1 Perfective vs. Imperfective

The *perfective* viewpoint presents situations as complete with both an initial and a final endpoint, while the *imperfective* viewpoint makes only certain parts of the situation visible to the receiver (Smith, 1997). The situation in (9a) is presented imperfectively, focusing on the middle phase of John's eating. In contrast, (9b) is viewed in the perfective. Here, the interpretation is that (b) happens during the interval at which (a) is true.

(9) (a) John was eating a sandwich *(imperfective)*
(b) when Susan entered. *(perfective)*

The perfective/imperfective distinction is observed cross-linguistically, although some languages make even finer distinctions. For example, Chinese has two perfective aspect markers and two imperfective aspect markers. The two perfective markers *le* (10a) and *guo* (10b) differ in the present relevance, with the former indicating completion of a situation while the latter emphasizes the experience of having been through a situation. The two imperfective aspect markers *zhèngzài* and *zhe* both indicate a situation is on-going at the reference time, but the latter emphasizes the resulting state of a situation (Chao, 1968; Li and Thompson, 1989; Ljungqvist, 2007).

(10) (a) tā jìn le fángjiān "*He has entered the room (and is still in the room)"*
(b) tā jìn guò fángjiān "*He entered the room (at some point but is no longer in the room)"*
(c) tā zhèngzài jìn fángjiān "*He is entering the room"*
(d) fángjiān de mén kái zhe "*The door of the room is open."*

While also referring to endpoints, the concept of viewpoint differs from telicity: telicity describes types of situations independent of which phase is focused (e.g., "eating a sandwich" in (9a) is an (inherently-telic) accomplishment); viewpoint adds the focus (in this case on the phase during which the eating happens and the endpoint has not yet been reached; more details in Appendix C.3). It is the perfective-imperfective distinction that has traditionally been referred to as *aspect* in Slavic linguistics (Filip 1999). Computational work leverages parallel corpora to map the distinction, which is partially *overt* (i.e., explicit in the morphosyntax) in Slavic languages, to English text (Stambolieva, 2011; Friedrich and Gateva, 2017). Despite overall high accuracy, the models still struggle to reach high scores for the minority class *atelic*. In addition, such annotation projection approaches are not easily scalable, as they require strong knowledge of the languages involved.

4.2 Habituality

Habituals such as (11b) are sentences that "express regularities about the world which constitute generalizations over events and activities" (Carlson, 2005); on a sentence-level, they can be regarded as "derived statives" (Smith, 1997). In contrast, the term *episodic* refers to particular events. Habituals allow exceptions, e.g., (11b) is still true if Mary sometimes takes the train.

(11) (a) Mary cycled to work. *(episodic)*
(b) Mary cycles to work. *(habitual)*

Habitual sentences may also use stative predicates as in (12), generalizing over situations in which some state applies (Smith, 2005).

(12) Sloths sometimes rest on trees. *(habitual)*

Habituals are not restricted to what one would usually consider a matter of habit (Carlson, 2005); they can also have inanimate subjects as in (13).

(13) Glass breaks easily. *(habitual)*

Habituality is not to be confused with *iterativity*, which states that a situation occurs repeatedly, but not regularly, as in "the light flashed." Borderline cases are discussed in Appendix C.4.

Data and Modeling. Mathew and Katz (2009) randomly select 1052 sentences for 57 verbs from the Penn TreeBank (Marcus et al., 1993) and manually mark them as habitual or episodic. They train both a decision tree and a Naive Bayes classifier on syntactic features extracted from gold parses. Friedrich and Pinkal (2015a) argue that in order to be able to apply such a model on free text, a third category, which they call *static*, needs to be taken into consideration, which covers lexically stative, negated, and modalized clauses. They experiment with Random Forest classifiers (Breiman, 2001) using syntactic-semantic features and linguistic indicators (see Section 3.1). Govindarajan et al. (2019) label UD-EWT (Silveira et al., 2014) with real-valued properties indicating e.g., the "degree" of habituality. Their multi-layer perceptron uses features from verb databases such as LCS and GLoVe embeddings. Results for predicting habituality are overall promising, although there is a need for experiments with careful consideration of and controls over the verb types involved.

Other recent related work (Williams, 2012; Williams and Katz, 2012; Vempala et al., 2018) extracts typical durations (in terms of actual time measures) for verb lemmas from Twitter. They distinguish episodic and habitual uses of the verbs, using the method of Mathew and Katz (2009), and collect typical durations (e.g., "seconds" or "weeks") for episodic and habitual uses separately for each verb. The problem has further been studied in the context of acquiring common sense knowledge (e.g., Zhou et al., 2019; Yang et al., 2020).

4.3 Grammatical Aspect in Recent Syntactic-Semantic Frameworks

The **Universal Dependencies** (UD) guidelines[4] define six aspectual *features* for verbs: Hab, Imp, Iter, Perf, Prog, and Prosp (prospective, for relative future). The categories are assumed to be language-specific and close to the respective morphologies, and the exact definition of each feature is left to the individual languages. UD parsers frequently treat the identification of these features as a tagging task (e.g., Kondratyuk and Straka, 2019).

In the context of a pilot study on integrating aspect into Abstract Meaning Representations (Banarescu et al., 2013, **AMR**), Donatelli et al. (2018) propose to indicate whether a clause is habitual or ongoing (in addition to marking stativity and telicity). Tense and aspect annotation has been incorporated into Dialogue-AMR and used to annotate a corpus of human-robot interactions (Bonial et al., 2019, 2020). The Uniform Meaning Representations (**UMR**) framework (Van Gysel et al., 2021) uses a lattice for typologically-motivated annotation of aspect. The lattice begins by making the distinction between three categories: HABITUAL, which covers recurring states and events; PROCESS, which covers all non-recurring dynamic situations; and IMPERFECTIVE, designated for non-recurring states and atelic processes. The PROCESS category is further divided into ongoing ACTIVITIES, ENDEAVORS which have begun but not finished, and PERFORMANCES, which are completed dynamic processes. See Appendix D.2 for the full lattice.

Chen et al. (2021) build a rule-based system leveraging syntactic and semantic cues to annotate English sentences according to the UMR aspect lattice. Due to the recency of these frameworks, so far there is only very limited computational work.

5 Situation Entities

In this section, we review work on automatically classifying *situation entity* (SE) types (see inventory in Appendix D.3). SEs are "semantic concepts organized according to their internal temporal properties" (Smith, 2003). They are introduced to the discourse by a clause's verb constellation, i.e., the clause's main verb and its arguments and modifiers. Deciding on the type of an SE thus involves the combination of lexical and syntactic factors. In contrast to eventuality types (Section 3.3), situation entities capture the linguistic nature of the clause (Friedrich and Palmer, 2014b). EVENTS include all dynamic verb constructions referring to particular non-recurring situations, no matter whether there is a pre-defined endpoint or not. They may occur in the perfective or imperfective. GENERALIZING SENTENCES correspond to habituals as described in Section 4.2, with the exception of GENERIC SENTENCES that comprise all sentences making statements about kinds (Krifka et al., 1995).

Datasets. Palmer et al. (2007) present 6065 clauses taken from the Brown corpus (Francis and Kučera, 1979) and MUC-6 (Grishman and Sundheim, 1996), manually annotated with SE types. The SitEnt dataset (Friedrich et al., 2016; Friedrich, 2017) consists of 40,000 clauses from 13 genres annotated for SE types with substantial agreement (Friedrich and Palmer, 2014b). Govindarajan et al.

---

[4] https://universaldependencies.org/u/feat/Aspect.html

(2019) crowdsource UDW-G, a dataset of 37,146 arguments and 33,114 predicates in 16,222 sentences with continuous judgments on a scale from 1 to 5 with distinctions essentially following the schema of Friedrich et al. (2016).

Modeling. Palmer et al. (2007) use a maximum entropy sequence tagger with part-of-speech and CCG-based (Steedman, 2000) syntactic features to predict labels for each sentence separately. In an oracle experiment, they show that including the label of the previous sentence(s) as a feature improves performance. Friedrich et al. (2016) present the first true sequence labeling approach to SE types classification using conditional random fields (Lafferty et al., 2001) with distributional features in the form of Brown cluster IDs (Brown et al., 1992) and syntactic-semantic features. Kober et al. (2020) propose to classify predicational aspect with distributional semantics, using non-contextualized word embeddings and sums of the word vectors of the target words and their context words. Several works (Palmer et al., 2007; Palmer and Friedrich, 2014; Friedrich et al., 2016) find that sequence information mostly helps when training in-domain and has the biggest impact on identifying GENERIC SENTENCES, which often require discourse context for classification, even by human annotators (Friedrich and Pinkal, 2015b).

SE types have also been modeled using a variety of neural approaches. Becker et al. (2017) employ a GRU-based (Cho et al., 2014) RNN with an attention mechanism. Dai and Huang (2018) dynamically build context-aware clause representations, informed by their paragraph-wide contexts. They propose a hierarchical recurrent neural network that reads entire paragraphs at once, learning interdependencies for clauses. Their model uses word2vec (Mikolov et al., 2013) embeddings for words, and first computes contextualized word embeddings using a BiLSTM over the entire paragraph. Clause embeddings are formed by max pooling over the word embeddings of each clause, and then clause embeddings are further contextualized via another BiLSTM. For further improving SE classification, they add a CRF layer on top, and finally predict SE labels via softmax. Recently, BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019) have been employed for the classification task by Rezaee et al. (2021), who also compare to using ParBERT (Cohan et al., 2019). While BERT only considers one clause at a time, ParBERT reads several sentences at once, separated by [SEP] tokens, and then uses the embeddings of the [SEP] tokens to predict a label for the corresponding clause. Comparing a variety of neural models based on non-contextualized word embeddings on their situation-entity related regression tasks, Govindarajan et al. (2019) reach levels similar to human agreement.

The computational problem of identifying SE types has recently been studied extensively. Overall, SE patterns are specific to the domain, genre or *discourse mode* (Smith, 2003). Future research could use SE-style analysis for NLP tasks like temporal processing or information extraction.

6 How Can We Move Forward?

In this section, we discuss potential steps that could lead to more successful modeling of aspectual information, as well as how to leverage that information in NLP applications.

6.1 Dataset Construction

To date, there is no consensus or standard across languages regarding where aspect lives. Ongoing efforts in the UMR community aim to develop typologically-informed aspectual representations that work across languages, but so far very few UMR datasets exist. Whether achieving a standard for aspectual representation is a necessary step or not, in order to make more rapid progress on the computational modeling of aspect, first the various models need to be **benchmarked** on the same tasks and datasets. **Dataset construction** in this area is challenging. Annotators not only need to have extensive linguistic training but also often domain expertise. Formulating the problem as a crowdsourcing task is a possible direction (Govindarajan et al., 2019), but obtaining consistently high agreement is (as for most tasks) difficult in this setup. **Multilingual** datasets and models are still under-researched, yet they are of particular relevance due to applications in computer-aided language learning or the evaluation of machine translation (see examples in Appendix E).

6.2 Modeling

More research is also needed on whether recent **embeddings** or text encoders capture or abstract away from aspectual features. Recent experimental findings on the SitEnt dataset show that, when training data within the same domain and genre is available, performance of pre-trained transformers is in the range of human agreement (Metheniti, 2022; Dai and Huang, 2018). The current state-of-the-art model based on BERT-GPT achieves a macro-average F1 of 79.1 compared to an estimate of human performance (78.6). Yet, most verbs are either not ambiguous or have a strongly predominant majority class that the models memorize (Friedrich and Palmer, 2014a; Kober et al., 2020). We expect much insight to come from further studying verb types that behave differently depending on the context. Moreover, results on cross-genre classification of SE types are yet inconclusive. Dai and Huang (2018) report high cross-genre scores for BERT, ParBERT, and BERT+GPT, but out of these models, there is no clear winner across all genres. Overall, when only using out-of-genre training data, performance of the BERT+GPT model still drops to an F1 of 70.7.

Investigating aspectual information in the context of **document-level models** (e.g., Beltagy et al., 2020) is another opportunity: cues for aspectual interpretion may occur earlier in the discourse than the sentence being interpreted (see App. E).

Linguistic indicators (Siegel and McKeown, 2000a) have been shown to be useful for predicting aspectual features in the absence of training data. Another research direction should hence look at how aspectual information can be **induced from raw data** or parallel corpora on a larger scale and for a broader set of distinctions. Future work should also investigate the interaction with other elements of semantics such as argument structure.

In sum, systems for predicting aspectual information encounter several challenges: (a) many verb types are ambiguous, (b) datasets for the task show significant class imbalance, and (c) for many verb types, no training data at all is available. Initial experiments with careful controls for different groups of verbs were highly promising (Friedrich and Palmer, 2014a; Kober et al., 2020), but available annotated data is sparse, especially for ambiguous verb types. More research is required to disentangle the effects of potentially lexically biased datasets, class imbalance, and sparsity.

6.3 Applications

Automatic classifiers for aspectual distinctions clearly have **applications** in the **digital humanities** and quantitative linguistics, where research questions may directly target the use of aspect. Recent highly fluent (but not necessarily always accurate) language generation models such as GPT-3 (Brown et al., 2020) and ChatGPT[5] open up a new level of writing assistance or language learning systems. We believe that a fruitful direction for future research is to make such systems explainable, also in the sense that they are able to give linguistically founded qualitative feedback to the user about why an aspectual form is correct or wrong in a particular context. Benchmarking on datasets explicitly annotated for aspectual information will let us estimate the degree to which text generated by the models is in line with the author's intentions. In Appendix F, we provide some chat logs of conversations with ChatGPT that illustrate how far it is from being an adequate tool for teaching the linguistic concepts described in this survey.

However, we argue that paying attention to aspect is necessary to arrive at fluent human-level NLP systems as motivated by our "miss the flight" example in Section 1. In Appendix E, we discuss two potential application areas: temporal processing and machine translation. In a nutshell, some works suggest that embedding spaces still struggle to capture the nature of tense and aspect, with consequences for commonsense reasoning driven by temporal properties, and can hence lead to wrong conclusions. Similarly, machine translation systems may (and still do) go wrong when cues for the correct temporal form occur much earlier in the document then the sentence being translated. More research is needed on how to **incorporate features** explicitly or **guide systems** implicitly towards the right output with regard to aspectual information, and also on how to convincingly evaluate generated output with regard to aspectual plausibility.

7 Conclusion

This paper provides an overview of the linguistic concepts and terminology associated with aspect, and also surveys existing datasets and prior work in computational modeling of aspect. Because the expression of aspect varies across languages, the proper incorporation of aspect into downstream NLP tasks requires language-dependent research rooted in the language-independent categories surveyed here. The semantic concepts expressed by lexical and grammatical aspect play essential roles in semantic interpretation, and their proper coding and evaluation must not be neglected if we are to one day develop human-level NLU systems.

---

[5] https://openai.com/blog/chatgpt

Limitations

This survey reviews linguistic and computational work on lexical and grammatical aspect. While we believe that we covered the vast majority of relevant computational works, we of course only scratch the surface of the body of theoretical linguistic work and need to simplify at times.

As the term *aspect* is highly ambiguous and used in NLP primarily within sentiment analysis, it is difficult to search for relevant papers. Our search hence relied mainly on following citation networks.

Ethical Considerations

We did not identify any potential ethical issues with this survey.

Acknowledgements

We thank the anonymous reviewers for their extremely valuable comments. This survey incorporates insights from countless extremely valuable discussions with many researchers and students. In particular, we thank Manfred Pinkal, Josef van Genabith, Bonnie Webber, Mark Steedman, Mike Lewis, Omri Abend, Melissa Peate Sorensen, Michaela Regneri, Andrea Horbach, Stefan Thater, Vera Demberg, Ines Rehbein, Caroline Sporleder, Fatemeh Torabi Asr, Asad Sayeed, Alessandra Zarcone, Hanna Kermes, Damyana Gateva, Anna Nedoluzhko, Michal Novák, Martha Palmer, and Laura Michaelis.

References

Malihe Alikhani and Matthew Stone. 2019. "caption" as a coherence relation: Evidence and implications. In *Proceedings of the Second Workshop on Shortcomings in Vision and Language*, pages 58–67, Minneapolis, Minnesota. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Maria Becker, Michael Staniek, Vivi Nastase, Alexis Palmer, and Anette Frank. 2017. Classifying semantic clause types: Modeling context and genre characteristics with recurrent neural networks and attention. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 230–240, Vancouver, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Steven Bethard. 2013. ClearTK-TimeML: A minimalist approach to TempEval 2013. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 10–14, Atlanta, Georgia, USA. Association for Computational Linguistics.

Steven Bethard and James H. Martin. 2006. Identification of event mentions and their semantic class. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 146–154, Sydney, Australia. Association for Computational Linguistics.

Steven Bethard, Guergana Savova, Wei-Te Chen, Leon Derczynski, James Pustejovsky, and Marc Verhagen. 2016. SemEval-2016 task 12: Clinical TempEval. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1052–1062, San Diego, California. Association for Computational Linguistics.

Steven Bethard, Guergana Savova, Martha Palmer, and James Pustejovsky. 2017. SemEval-2017 task 12: Clinical TempEval. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 565–572, Vancouver, Canada. Association for Computational Linguistics.

Claire Bonial, Lucia Donatelli, Mitchell Abrams, Stephanie M. Lukin, Stephen Tratz, Matthew Marge, Ron Artstein, David Traum, and Clare Voss. 2020. Dialogue-AMR: Abstract Meaning Representation for dialogue. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 684–695, Marseille, France. European Language Resources Association.

Claire Bonial, Lucia Donatelli, Stephanie M. Lukin, Stephen Tratz, Ron Artstein, David Traum, and Clare Voss. 2019. Augmenting Abstract Meaning Representation for human-robot dialogue. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 199–210, Florence, Italy. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages

632–642, Lisbon, Portugal. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Susan Brennan, Katharina Schuhmann, and Karla Batres. 2013. Entrainment on the move and in the lab: The walking around corpus. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 35. Cognitive Science Society.

Michael R. Brent. 1991a. Automatic semantic classification of verbs from their syntactic contexts: An implemented classifier for stativity. In *Fifth Conference of the European Chapter of the Association for Computational Linguistics*, Berlin, Germany. Association for Computational Linguistics.

Michael R Brent. 1991b. Automatic semantic classification of verbs from their syntactic contexts: an implemented classifier for stativity. In *Proceedings of the fifth conference on European chapter of the Association for Computational Linguistics (ACL)*, pages 222–226, Berlin, Germany.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *cs.CL*.

Gregory N Carlson. 2005. Generics, habituals and iteratives. In Keith Brown, editor, *The Encyclopedia of Language and Linguistics*. Elsevier Ltd.

Franti Čermák and Alexandr Rosen. 2012. The case of intercorp, a multilingual parallel corpus. *International Journal of Corpus Linguistics*, 17(3):411–427.

Nathanael Chambers. 2013. NavyTime: Event and time ordering from raw text. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 73–77, Atlanta, Georgia, USA. Association for Computational Linguistics.

Yuen Ren Chao. 1968. *A grammar of spoken Chinese*. Univ of California Press.

Daniel Chen and Alexis Palmer. 2022. Contrast sets for stativity of English verbs in context. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4028–4036, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Daniel Chen, Martha Palmer, and Meagan Vigus. 2021. AutoAspect: Automatic annotation of tense and aspect for uniform meaning representations. In *Proceedings of The Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop*, pages 36–45, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Dan Weld. 2019. Pretrained language models for sequential sentence classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3693–3699, Hong Kong, China. Association for Computational Linguistics.

Bernard Comrie. 1976. *Aspect: An introduction to the study of verbal aspect and related problems*, volume 2 of *Cambridge Textbooks in Linguistics*. Cambridge University Press.

Corinna Cortes and Vladimir Vapnik. 1995. Support-Vector Networks. *Machine Learning*, 20(3):273–297.

Francisco Costa and António Branco. 2012. Aspectual type and temporal relation classification. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 266–275, Avignon, France. Association for Computational Linguistics.

William Croft, Pavlína Pešková, and Michael Regan. 2016. Annotation of causal and aspectual structure of events in RED: a preliminary report. In *Proceedings of the Fourth Workshop on Events*, pages 8–17, San Diego, California. Association for Computational Linguistics.

Zeyu Dai and Ruihong Huang. 2018. Building context-aware clause representations for situation entity type classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3305–3315, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages

4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lucia Donatelli, Michael Regan, William Croft, and Nathan Schneider. 2018. Annotation of tense and aspect semantics for sentential AMR. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 96–108, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bonnie J Dorr. 2001. Lcs verb database, online software database of lexical conceptual structures and documentation, university of maryland.

Bonnie J. Dorr and Mari Broman Olsen. 1997. Deriving verbal and compositonal lexical aspect for NLP applications. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 151–158, Madrid, Spain. Association for Computational Linguistics.

David R Dowty. 1979. *Word meaning and Montague grammar – The Semantics of Verbs and Times in Generative Semantics and in Montague's PTQ*. Studies in Linguistics and Philosophy. Springer.

Jesse Dunietz, Greg Burnham, Akash Bharadwaj, Owen Rambow, Jennifer Chu-Carroll, and Dave Ferrucci. 2020. To test machine comprehension, start by defining comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7839–7859, Online. Association for Computational Linguistics.

Markus Egg, Helena Prepens, and Will Roberts. 2019. Annotation and automatic classification of aspectual categories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3335–3341, Florence, Italy. Association for Computational Linguistics.

Gertrud Faaß and Kerstin Eckart. 2013. Sdewac–a corpus of parsable sentences from the web. In *Language processing and knowledge in the Web*, pages 61–68. Springer.

Ingrid Falk and Fabienne Martin. 2016. Automatic identification of aspectual classes across verbal readings. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 12–22, Berlin, Germany. Association for Computational Linguistics.

Christiane Fellbaum. 2010. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer.

Hana Filip. 2012. Lexical aspect. *The Oxford handbook of tense and aspect*, pages 721–751.

Hana Filip and Gregory N Carlson. 1997. Sui generis genericity. *University of Pennsylvania Working Papers in Linguistics*, 4(2):7.

Charles J Fillmore. 1969. Types of lexical information. In *Studies in syntax and semantics*, pages 109–137. Springer.

W Nelson Francis and Henry Kučera. 1979. Brown corpus manual. *Brown University*. Http://clu.uni.no/icame/brown/bcm.html.

Jacques François, Denis Le Pesant, and Danielle Leeman. 2007. Présentation de la classification des verbes français de jean dubois et françoise dubois-charlier. *Langue française*, 1:3–19.

Annemarie Friedrich and Damyana Gateva. 2017. Classification of telicity using cross-linguistic annotation projection. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2559–2565, Copenhagen, Denmark. Association for Computational Linguistics.

Annemarie Friedrich and Alexis Palmer. 2014a. Automatic prediction of aspectual class of verbs in context. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 517–523, Baltimore, Maryland. Association for Computational Linguistics.

Annemarie Friedrich and Alexis Palmer. 2014b. Situation entity annotation. In *Proceedings of LAW VIII - The 8th Linguistic Annotation Workshop*, pages 149–158, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Annemarie Friedrich, Alexis Palmer, and Manfred Pinkal. 2016. Situation entity types: automatic classification of clause-level aspect. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1757–1768, Berlin, Germany. Association for Computational Linguistics.

Annemarie Friedrich and Manfred Pinkal. 2015a. Automatic recognition of habituals: a three-way classification of clausal aspect. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2471–2481, Lisbon, Portugal. Association for Computational Linguistics.

Annemarie Friedrich and Manfred Pinkal. 2015b. Discourse-sensitive automatic identification of generic expressions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1272–1281, Beijing, China. Association for Computational Linguistics.

Annemarie Silke Friedrich. 2017. *States, events, and generics: computational modeling of situation entity types*. Ph.D. thesis, Saarland University.

William Gantt, Lelia Glass, and Aaron Steven White. 2022. Decomposing and recomposing event structure. *Transactions of the Association for Computational Linguistics*, 10:17–34.

Howard B Garey. 1957. Verbal aspect in french. *Language*, 33(2):91–110.

Venkata Govindarajan, Benjamin Van Durme, and Aaron Steven White. 2019. Decomposing generalization: Models of generic, habitual, and episodic statements. *Transactions of the Association for Computational Linguistics*, 7:501–517.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference-6: A brief history. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 466–471. Association for Computational Linguistics.

Valentine Hacquard. 2009. On the interaction of aspect and modal auxiliaries. *Linguistics and Philosophy*, 32(3):279–315.

Jürgen Hermes, Michael Richter, and Claes Neuefeind. 2015. Automatic Induction of German Aspectual Verb Classes in a Distributional Framework. In *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology (GSCL)*, pages 122–129, Duisburg-Essen, Germany.

Jürgen Hermes, Michael Richter, and Claes Neuefeind. 2018. Supervised classification of aspectual verb classes in german-subcategorization-frame-based vs window-based approach: A comparison. In *ICAART (2)*, pages 653–662.

Nancy Ide, Collin Baker, Christiane Fellbaum, Charles Fillmore, and Rebecca Passonneau. 2008. MASC: the manually annotated sub-corpus of American English. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Rei Ikuta, Will Styler, Mariah Hamang, Tim O'Gorman, and Martha Palmer. 2014. Challenges of adding causation to richer event descriptions. In *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 12–20, Baltimore, Maryland, USA. Association for Computational Linguistics.

Hyuckchul Jung and Amanda Stent. 2013. ATT1: Temporal annotation using big windows and rich syntactic and semantic features. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 20–24, Atlanta, Georgia, USA. Association for Computational Linguistics.

Richard Keelan. 2012. *Lexical Aspectual Classification*. University of Ottawa (Canada).

Anthony Kenny. 1963. *Action, emotion and will*. Routledge.

Judith L. Klavans and Martin Chodorow. 1992a. Degrees of stativity: The lexical representation of verb aspect. In *COLING 1992 Volume 4: The 14th International Conference on Computational Linguistics*.

Judith L Klavans and Martin Chodorow. 1992b. Degrees of stativity: the lexical representation of verb aspect. In *Proceedings of the 14th conference on Computational linguistics (Coling)*, volume 4, pages 1126–1131, Nantes, France.

Thomas Kober, Malihe Alikhani, Matthew Stone, and Mark Steedman. 2020. Aspectuality across genre: A distributional semantics approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4546–4562, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Thomas Kober, Julie Weeds, Lorenzo Bertolini, and David Weir. 2021. Data augmentation for hypernymy detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1034–1048, Online. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Manfred Krifka, Francis Jeffrey Pelletier, Gregory N. Carlson, Alice ter Meulen, Godehard Link, and Gennaro Chierchia. 1995. Genericity: An Introduction. In Gregory N. Carlson and Francis Jeffry Pelletier, editors, *The Generic Book*, Studies in Communication, Media, and Public Opinion, pages 1–124. University Of Chicago Press.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth International Conference on Machine Learning (ICML)*, volume 1, pages 282–289, Williamstown, Massachussetts.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Alex Lascarides. 1991. The Progressive and the Imperfective Paradox. *Synthese*, 87(3):401–447.

Geoffrey N Leech. 1971. *Meaning and the English verb*. Pearson Education.

Charles N Li and Sandra A Thompson. 1989. *Mandarin Chinese: A functional reference grammar*, volume 3. Univ of California Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Marita Ljungqvist. 2007. Le, guo and zhe in mandarin chinese: a relevance-theoretic account. *Journal of East Asian Linguistics*, 16(3):193–235.

Hector Llorens, Estela Saquete, and Borja Navarro-Colorado. 2010. TimeML events recognition and classification: Learning CRF models with semantic roles. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 725–733, Beijing, China. Coling 2010 Organizing Committee.

Sharid Loáiciga and Cristina Grisot. 2016. Predicting and using a pragmatic component of lexical aspect of simple past verbal tenses for improving English-to-French machine translation. In *Linguistic Issues in Language Technology, Volume 13, 2016*. CSLI Publications.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Thomas A. Mathew and E. Graham Katz. 2009. Supervised Categorization of Habitual and Episodic Sentences. In *Sixth Midwest Computational Linguistics Colloquium*, Bloomington, Indiana: Indiana University.

Eleni Metheniti. 2022. About time: Do transformers learn temporal verbal aspect? In *Cognitive Modeling and Computational Linguistics (CMCL)*, Dublin,Ireland.

Eleni Metheniti, Tim van de Cruys, and Nabil Hathout. 2021. Prédire l'aspect linguistique en anglais au moyen de transformers (classifying linguistic aspect in English with transformers). In *Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale*, pages 209–218, Lille, France. ATALA.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Marc Moens and Mark Steedman. 1988. Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28.

Alexander PD Mourelatos. 1978. Events, processes, and states. *Linguistics and philosophy*, 2(3):415–434.

Skatje Myers and Martha Palmer. 2019. ClearTAC: Verb tense, aspect, and form classification using neural nets. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 136–140, Florence, Italy. Association for Computational Linguistics.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2021. The Richer Event Description Corpus for Event-Event Relations. *Computational Analysis of Storylines: Making Sense of Events (Studies in Natural Language Processing)*.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56.

Alexis Palmer and Annemarie Friedrich. 2014. Genre distinctions and discourse modes: Text types differ in their situation type distributions. In *Proceedings of the Symposium on Frontiers and Connections between Argumentation Mining and Natural Language Processing*, Bertinoro, Italy.

Alexis Palmer, Elias Ponvert, Jason Baldridge, and Carlota Smith. 2007. A sequencing model for situation entity classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 896–903, Prague, Czech Republic. Association for Computational Linguistics.

Rebecca J. Passonneau. 1988a. A computational model of the semantics of tense and aspect. *Computational Linguistics*, 14(2):44–60.

Rebecca J Passonneau. 1988b. A computational model of the semantics of tense and aspect. *Computational Linguistics*, 14(2):44–60.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 67–81, Brussels, Belgium. Association for Computational Linguistics.

James Pustejovsky, José M Castano, Robert Ingria, Roser Sauri, Robert J Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R Radev. 2003. Timeml: Robust specification of event and temporal expressions in text. *New directions in question answering*, 3:28–34.

James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. 2010. ISO-TimeML: An international standard for semantic annotation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Anita Ramm, Sharid Loáiciga, Annemarie Friedrich, and Alexander Fraser. 2017. Annotating tense, mood and voice for English, French and German. In *Proceedings of ACL 2017, System Demonstrations*, pages 1–6, Vancouver, Canada. Association for Computational Linguistics.

Roi Reichart and Ari Rappoport. 2010. Tense sense disambiguation: A new syntactic polysemy task. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 325–334, Cambridge, MA. Association for Computational Linguistics.

Mehdi Rezaee, Kasra Darvish, Gaoussou Youssouf Kebe, and Francis Ferraro. 2021. Discriminative and generative transformer-based models for situation entity classification. *CoRR*, abs/2109.07434.

Alexandr Rosen and Martin Vavřín. 2012. Building a multilingual parallel corpus for human users. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2447–2452, Istanbul, Turkey. European Language Resources Association (ELRA).

Roser Saurí, Robert Knippen, Marc Verhagen, and James Pustejovsky. 2005. Evita: A robust event recognizer for QA systems. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 700–707, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Eric V. Siegel. 1997. Learning methods for combining linguistic indicators to classify verbs. In *Second Conference on Empirical Methods in Natural Language Processing*.

Eric V. Siegel. 1999. Corpus-based linguistic indicators for aspectual classification. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 112–119, College Park, Maryland, USA. Association for Computational Linguistics.

Eric V Siegel and Kathleen R McKeown. 1996. Gathering statistics to aspectually classify sentences with a genetic algorithm. In *Proceedings of the Second International Conference on New Methods in Language Processing*, Ankara, Turkey.

Eric V Siegel and Kathleen R McKeown. 2000a. Learning methods to combine linguistic indicators: Improving aspectual classification and revealing linguistic insights. *Computational Linguistics*, 26(4):595–628.

Eric V. Siegel and Kathleen R. McKeown. 2000b. Learning methods to combine linguistic indicators:improving aspectual classification and revealing linguistic insights. *Computational Linguistics*, 26(4):595–627.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Christopher D. Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Carlota S Smith. 1997. *The parameter of aspect*, volume 43 of *Studies in Linguistics and Philosophy*. Springer Science & Business Media.

Carlota S Smith. 2003. *Modes of discourse: The local structure of texts*, volume 103. Cambridge University Press.

Carlota S Smith. 2005. Aspectual entities and tense in discourse. In *Aspectual inquiries*, pages 223–237. Springer.

Maria Stambolieva. 2011. Parallel corpora in aspectual studies of non-aspect languages. In *Proceedings of The Second Workshop on Annotation and Exploitation of Parallel Corpora*, pages 39–42, Hissar, Bulgaria. Association for Computational Linguistics.

Mark Steedman. 2000. *The syntactic process*, volume 24 of *Language, Speech, and Communication*. MIT Press.

Tianxiang Sun, Xiangyang Liu, Xipeng Qiu, and Xuanjing Huang. 2021. Paradigm shift in natural language processing. *arXiv preprint arXiv:2109.12575*.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957, Uppsala, Sweden. Association for Computational Linguistics.

Sean Trott, Tiago Timponi Torrent, Nancy Chang, and Nathan Schneider. 2020. (Re)construing meaning in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5170–5184, Online. Association for Computational Linguistics.

Naushadand UzZaman, Hectorand Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia.

Jens Van Gysel, Meagan Vigus, Jin Zhao, and Nianwen Xue. 2022. Developing a uniform meaning representation for natural language processing. In *LREC 2022 tutorial*.

Jens E. L. Van Gysel, Meagan Vigus, Jayeol Chun, Kenneth Lai, Sarah Moeller, Jiarui Yao, Tim O'Gorman, Andrew Cowell, William Croft, Chu-Ren Huang, Jan Hajič, James H. Martin, Stephan Oepen, Martha

Palmer, James Pustejovsky, Rosa Vallejos, and Nianwen Xue. 2021. Designing a uniform meaning representation for natural language processing. *KI - Künstliche Intelligenz*, 35(0):343–360.

Eva Vanmassenhove, Jinhua Du, and Andy Way. 2017. Investigating 'aspect'in nmt and smt: Translating the english simple past and present perfect. *Computational Linguistics in the Netherlands Journal*, 7:109–128.

Siddharth Vashishtha, Adam Poliak, Yash Kumar Lal, Benjamin Van Durme, and Aaron Steven White. 2020. Temporal reasoning in natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4070–4078, Online. Association for Computational Linguistics.

Alakananda Vempala, Eduardo Blanco, and Alexis Palmer. 2018. Determining event durations: Models and error analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 164–168, New Orleans, Louisiana. Association for Computational Linguistics.

Zeno Vendler. 1957. Verbs and times. *The philosophical review*, 66(2):143–160.

Henk J Verkuyl. 1972. *On the Compositional Nature of the Aspects.*, volume 15 D of *Foundations of Language, Supplementary Series*. D. Reidel Publishing Company, Dordrecht-Holland.

Henk J Verkuyl. 2005. Aspectual composition: Surveying the ingredients. In *Perspectives on aspect*, pages 19–39. Springer.

Benjamin Lee Whorf. 1945. Grammatical categories. *Language*, 21(1):1–11.

Jennifer Williams. 2012. Extracting Fine-grained Durations for Verbs from Twitter. In *Proceedings of the ACL 2012 Student Research Workshop*, pages 49–54, Jeju Island, Korea.

Jennifer Williams and Graham Katz. 2012. Extracting and modeling durations for habits and events from Twitter. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 223–227, Jeju Island, Korea. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Zonglin Yang, Xinya Du, Alexander Rush, and Claire Cardie. 2020. Improving event duration prediction via time-aware pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3370–3378, Online. Association for Computational Linguistics.

Zdeněk Žabokrtskỳ and Markéta Lopatková. 2007. Valency information in vallex 2.0. *The Prague Bulletin of Mathematical Linguistics*, 87:41–60.

Alessandra Zarcone and Alessandro Lenci. 2008. Computational models for event type classification in context. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Rowan Zellers and Yejin Choi. 2017. Zero-shot activity recognition with verb attribute induction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 946–958, Copenhagen, Denmark. Association for Computational Linguistics.

Yuchen Zhang and Nianwen Xue. 2014. Automatic inference of the tense of Chinese events using implicit linguistic information. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1902–1911, Doha, Qatar. Association for Computational Linguistics.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

APPENDIX

A Glossary

This glossary intends to provide a concise alphabetically ordered overview of the linguistic terminology used in this paper.

accomplishment eventive / dynamic situation type according to (Vendler, 1957; Smith, 1997), consisting of a preparatory phase and an endpoint, e.g., “build a house.”

achievement eventive / dynamic situation type according to (Vendler, 1957; Smith, 1997) consisting only of the *punctual* event that changes a state, e.g., “win the race.”

activity eventive / dynamic situation type according to (Vendler, 1957; Smith, 1997), describes a process that does not have a pre-defined endpoint, e.g., “swim.”

atelic describes situations or verbs that do not have a built-in endpoint, e.g., “draw.”

bounded see *telic*.

covert describes linguistic categories that are not directly inferrable from the lexicon or from grammatical/syntactic structure (Whorf, 1945). They usually consist of *semantic* concepts that require taking into account the context or pragmatic factors.

durative describes states that extend in time.

dynamic describes a type of situation / verb that has a part to its meaning that applies a force or changes a state.

episodic describes particular events that “happen” or “have happened” (opposite of *habituals*).

eventive see *dynamic*.

habitual clause that expresses a situation that repeats regularly and expresses a characteristic, e.g., “John drives to work” or “Bishops move diagonally” (Krifka et al., 1995).

imperfective describes a perspective on a situation focusing one or none of the potential endpoints, but not both

iteratives clauses (usually with *semelfactive* verbs) that signal that the event expressed by the verb is executed a number of times, e.g., “the light flashed.”

overt describes linguistic categories that are directly inferrable from the lexicon or from grammatical/syntactic structures.

state a situation that is not changing, e.g., “John owns a house.”

stative describes verbs or situations that express a state.

perfective describes constructions that show a situation with its endpoint(s), e.g., “John traveled to the US (last week).”

progressive describes constructions that focus on a particular phase of a situation that is currently ongoing, e.g., “John is traveling to the US (at the moment).”

punctual event that occurs at a single point in time, e.g., “hiccup.” – *dynamic* by definition.

semelfactive *punctual* situation or event type (Smith, 1997).

telic describes situations or verbs that have a built-in endpoint, e.g., “capture.”

B Overview of Modeling Approaches

Table 3 gives a concise overview of existing modeling approaches.

C Further Linguistic Background

C.1 Eventuality Types

In Section 3.3, we explain a taxonomy of eventuality types according to Vendler (1957). As shown in Figure 2, Kenny (1963) adds *performances*, which are “actions that tend towards a goal.” Mourelatos (1978) criticizes that these earlier analyses focus too much on predicates that require human agency, and suggests the terminology in Figure 2.

As explained in Section 3.3, Moens and Steedman (1988) introduce an inventory of eventuality types similar to those of Vendler (1957), but referring to the characteristics of whether an event is *atomic*, *extended* (*durative*) and whether it has a *consequent state*. The definitions of the types along with examples are given in Table 4.

Type of Model	Reference(s)	Categories Targeted / Task
Rule-based	Passonneau (1988b) (PUNDIT)	stativity
	Chen et al. (2021)	UMR aspect features
Progressive as indicator	Klavans and Chodorow (1992b)	stativity
Linguistic indicators	Brent (1991b)	stativity
	Siegel and McKeown (1996, 2000b)	stativity, boundedness
Naive Bayes + Decision Tree	Mathew and Katz (2009)	habituality
Random Forest Classifiers	Friedrich and Pinkal (2015a)	habituality
Logistic regression	Friedrich and Palmer (2014a)	lexical aspect
	Loáiciga and Grisot (2016)	boundedness
	Friedrich and Gateva (2017)	telicity
	Zarcone and Lenci (2008)	Vendler types
Distributional models	Kober et al. (2020)	telicity
SVM	Keelan (2012)	Leech's classes
	Hermes et al. (2015, 2018)	Vendler classes
Max Ent Sequence Tagger	Palmer et al. (2007)	situation entity types
CRF	Friedrich and Pinkal (2015b)	genericity
	Friedrich et al. (2016)	situation entities
word2vec + GRU	Becker et al. (2017)	situation entity types
GloVe + GRU	Zellers and Choi (2017)	verb-level Vendler classes
Paragraph-level neural model	Dai and Huang (2018)	situation entity types
ELMO + MLP	Govindarajan et al. (2019)	habituality (continuous features, UDS-G)
BERT + classifiers	Metheniti et al. (2021); Metheniti (2022)	telicity
BERT + tagger	e.g., Kondratyuk and Straka (2019)	UD aspectual features
BERT, GPT, ParBERT	Rezaee et al. (2021)	situation entity types
Generative model	Gantt et al. (2022)	event types

Table 3: **Computational systems and approaches** to modeling aspect.

Vendler (1957)	*dynamic*	*durative*	*telic*	Moens and Steedman (1988)	*atomic event*	*extended event*	*consequent state*	**Examples**
State	-	+	-	state	NA	NA	NA	know the answer, love Mary, understand
Activity	+	+	-	process	-	+	-	laugh, stroll in the park, run, swim
Accomplishment	+	+	+	culminated process	-	+	+	build a house, walk to school
Achievement	+	-	+	culmination	+	-	+	win a race, reach the top, recognize
Semelfactive	+	-	-	point event	+	-	-	tap, knock, hiccup, wink

Table 4: **Eventuality** types (Vendler, 1957; Smith, 1997) / **Temporal Ontology** (Moens and Steedman, 1988).

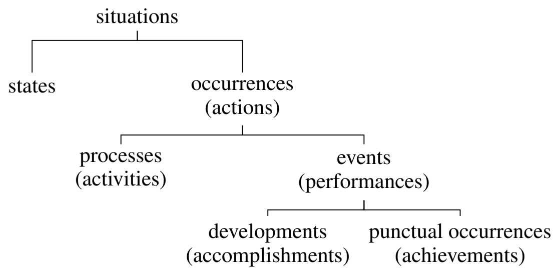


Figure 2: Classification of aspectual oppositions according to Mourelatos (1978), terms used by Vendler (1957) and Kenny (1963) in parentheses.

C.2 Leech's Classes

In addition to categories corresponding to those of Figure 2, Leech (1971) distinguishes between *Attitude* states ("hate," "hope") and *Relationship* states ("own," "resemble"). There are also two categories for *Perception* and *Cognition* verbs such as "hear," "see," or "feel," which are generally hard to classify along the stative-dynamic dimension (Comrie, 1976).

C.3 Details on Viewpoint Aspect

In some cases, viewpoint aspect (Section 4) and situation type interact. The perfective viewpoint is naturally available for the situation types (Section 3.3) activity, accomplishment, semelfactive and achievement; the imperfective viewpoint is available for states, activities and accomplishments (Smith, 1997). The usage of the perfective viewpoint with stative predicates indicates aspectual coercion as introduced in Section 3.3. For example, in Chinese the stative predicate "gāo" means to be tall. When used with the perfective marker "le," an ingressive meaning is implied, i.e., "tā gāo-le" translates as "he became tall."

C.4 Habituals: Borderline cases

Habitual sentences (Section 4.2) describe situations that are characteristic of an extended period of time (Comrie, 1976), a decision that is of conceptual rather than of linguistic nature. In fact, Filip and Carlson (1997) even argue that *sentential genericity*, which corresponds to habituality, is independent from tense and aspect, and that habitual sentences such as (14) can occur in the perfective.

(14) In the eighties, John went to work by bus.

The interaction of habituality and modality is by no means trivial (Hacquard, 2009) as illustrated by (15a), and negated sentences are another unclear case (15b-c).

(15) (a) I had to swim every day. *(habitual?)*
(b) John smokes. *(habitual)*
(c) John does not smoke. *(habitual?)*

The concept of habituality does not include *dispositional sentences* such as "John can swim," which denote abilities or preferences.

D Further Computational Background

D.1 Linguistic Indicators

Table 5 reports the full set of linguistic indicator features as proposed by Siegel and McKeown (2000b) and related works.

Feature	Example
frequency	-
past	*said*
perfect	*had won*
progressive	*is winning*
negated	*not/never*
particle	*up/in/...*
no subject	-
continuous adverb	*continually, endlessly*
evaluation adverb	*better, horribly*
manner adverb	*furiously, patiently*
temporal adverb	*again, finally*
in-PP	*in an hour*
for-PP	*for an hour*

Table 5: **Linguistic indicators** computed over large syntactically parsed text corpora (Siegel and McKeown, 2000b).

D.2 Aspect in UMR

As briefly described in Section 4.3, the UMR (Uniform Meaning Representation) (Van Gysel et al., 2021) framework approaches annotation of aspect from a typological perspective. Aspect is represented as an attribute for events in UMR, and since cross-linguistically languages mark aspect in different ways, both grammatically and lexically, the UMR aspect categories are arranged in a lattice. This allows UMRs to be annotated at the level of granularity that is most appropriate for a particular language.

SE type	Example
Eventualities	
STATE	The colonel owns the farm.
EVENT	John won the race.
REPORT	"...", said Obama.
General Statives	
GENERIC SENT.	The lion has a bushy tail.
GENERALIZING SENTENCE	Mary often fed the cat last year.
Abstract Entities	
FACT	(I know) that she refused the offer.
PROPOSITION	(I believe) that she refused the offer.
QUESTION	Who wants to come?
IMPERATIVE	Hand me the pen!

Table 6: Situation entity types (Smith, 2003).

The resulting lattice (Van Gysel et al., 2022) appears in Figure 3. The top-level of the lattice represents broad distinctions between aspect categories across languages, while the bottom level represents the finer distinctions.

D.3 Situation entity inventory

Table 6 provides the complete inventory of situation entity types. Section 5 offers a partial inventory, focusing on those categories for which aspect is a key determining property.

D.4 Event Classes in TimeBank

TimeML (Pustejovsky et al., 2003, 2010) events are "situations that occur or happen," but also include "states or circumstances in which something obtains or holds true." Thus, the usage of the term *event* denotes a *situation* in the sense of Smith (1997). In TimeML, tensed verbs, stative adjectives and event nominals that describe situations temporally located in the text can be marked with the categories shown in Table 7.

TimeML event classification corresponds to a word-sense level task. Saurí et al. (2005), in their event recognition system, simply assign the class that was most frequently observed for each verb type in the training data to events and reach an accuracy of 82.3% on TimeBank 1.2. The top-performing systems (Jung and Stent, 2013; Bethard, 2013; Chambers, 2013) of the 2013 TempEval challenge (UzZaman et al., 2013) use corpus-based features, WordNet synsets, parse paths and fea-

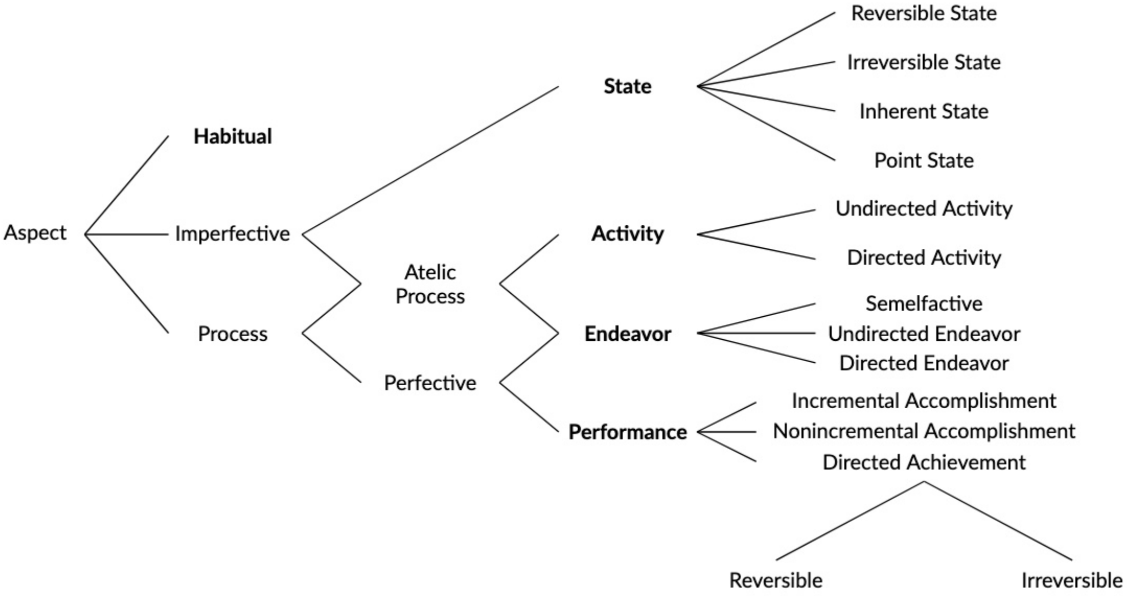


Figure 3: UMR annotation lattice for aspect (Van Gysel et al., 2022).

tures from typed dependencies to classify events as a joint task with determining the event's span. Bethard and Martin (2006) phrase the recognition of EVENTs and their semantic class as a chunking task using syntactic-semantic features such as part-of-speech, morphological information, word clusters and WordNet hypernyms. Llorens et al. (2010) extend this idea by using a conditional random field (Lafferty et al., 2001) enhanced with semantic role information. Costa and Branco (2012) explore the usefulness of a wider range of explicitly aspectual features, including linguistic indicators, for temporal relation classification in Portuguese.

More recent TempEval challenges (Bethard et al., 2016, 2017) did not offer a task for classifying event classes.

E (Potential) Applications

Despite its importance for understanding, few NLP tasks explicitly incorporate aspectual information. Here we discuss the potential uses in two application areas; many other areas could also benefit, including argumentation mining, computer-aided language learning, and information extraction.

Temporal Processing. Together with tense, aspect is essential to the linguistic system encoding temporal information (Smith, 1997, 2003). Yet there is little work systematically leveraging aspectual information in temporal relation extraction systems, possibly because TimeML event classes (Pustejovsky et al., 2003, 2010, Appendix D.4) are not strictly rooted in linguistic theory. We believe that recognizing the mode of discourse and the type of temporal progression (which is different e.g., in a narrative, an encyclopedia entry, or a news article) is key to recognizing temporal structure in text.

Event Class	Explanation and Examples
OCCURRENCE	situations that happen: *die, crash, merge*
STATE	circumstances in which sth. holds: *like, own, the kidnapped girl, on board*
I_ACTION	intensional actions: *try, persuade*
I_STATE	intensional states: *love, believe, enjoy*
ASPECTUAL	predicates that pick out a phase of the event: *begin, start, continue*
I_REPORTING	capture attribution: *said*
PERECPTION	physical perception of another event: *see, hear, feel*

Table 7: TimeML event classes (Pustejovsky et al., 2003).

With the help of templates, Vashishtha et al. (2020) re-cast existing temporal relation and temporal duration datasets as natural language inference (NLI) datasets. Existing models perform well on classifying English sentence-internal relations this way, indicating that the number of tense pair patterns is somewhat limited. However, using large-pretrained transformer based models with a paradigm shift (Sun et al., 2021) is unlikely to be the solution to temporal reasoning, as shown by Kober et al. (2021). They create a dataset consisting of 11,138 pairs of short sentences labeled with a binary annotation scheme (*entailment* vs. *non-entailment*), addressing tense and the perfect vs. progressive distinction in English. For example, "John is visiting London" entails "John has arrived in London," but "John will visit London" does not. They evaluate a range of models including non-contextualized and contextualized embeddings pre-trained on SNLI (Bowman et al., 2015) and DNC (Poliak et al., 2018) and find that none of these models outperforms a majority class baseline on the new dataset. This suggests that embedding spaces struggle to capture the more latent nature of tense and aspect.

In our view, a first step towards leveraging the knowledge about tense and aspect provided by linguistic theory could be to systematically study how recent language models and NLP systems succeed or fail with regard to these categories, as pioneered in the pre-neural age for example by Zhang and Xue (2014); Reichart and Rappoport (2010).

Machine Translation. While recent MT systems often perform well, they do not systematically treat aspectual notions (Vanmassenhove et al., 2017), but rather rely on common translations for a particular domain. Consider the following example. In contrast to English, the German simple present may or may not indicate habituality. When translating to English, a choice must be made based on the context. The translation[6] in (16b) wrongly indicates habitual viewpoint, while the context suggests an ongoing event (the correct translation would be "is riding").

(16) (a) Tim ist schon weg. Er fährt gerade mit dem Fahrrad zur Arbeit. *(ongoing event)*
(b) Tim is already gone. He just rides his bike to work. *(habitual, wrong)*

The cue need not be in the immediately preceding sentence, but could occur much earlier in the discourse. In (17), "fährt" should be translated as "is riding," but the cue is in the first sentence.

(17) (a) Tim ist schon weg. Er arbeitet an der Universität. Er fährt gerade mit dem Fahrrad zur Arbeit. *(ongoing event)*
(b) Tim is already gone. He works at the university. He just rides his bike to work. *(habitual, wrong[7])*

Aspect is a crucial component of semantic interpretation, and special care is needed to make sure translations correctly capture it.

F ChatGPT Logs

Accessed January 26 and 27, 2023.

In this section we provide logs from our interactions with ChatGPT on some of the main topics covered by this survey. Our aim is to investigate the extent to which a system like ChatGPT could serve as a tool for teaching about aspect; in other words, what does ChatGPT know about aspect?

Note that some of ChatGPT's analyses are surprisingly accurate, however, it also presents several completely wrong analyses and only produces correct analyses if guided to them by the user during the generation process.

Each subsection contains one or more segments from a chat log about a particular aspectual notion. We use the formatting shown below:

- *Prompt*
- Text generated by ChatGPT (with interesting parts highlighted in **bold**)
- *[Comments]*

F.1 Talking about Stativity

Consider the following text. "Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, and 'what is the use of a book,' thought Alice, 'without pictures or conversation?'" Which verb phrases refer to states, which verb phrases refer to events in this passage?

In this passage, "was beginning to get very tired" and "having nothing to do" are verb phrases that refer to states. "She had peeped" and **"had no pictures or conversations in it" are verb phrases that refer to events**.

[The verb phrases "have nothing to do" and "had no pictures or conversations in it" are states; "had peeped" is an event; "was beginning" is a special case at it signals the beginning of a state

---

[6] Google Translate, October 5, 2022

[7] Google Translate, October 19, 2022

(inchoative aspect). ChatGPT's response is simplistic and in some cases completely wrong.]

Why is "had no pictures or conversations in it" an event?

The phrase "had no pictures or conversations in it" is an event because it describes something that happened in the past, specifically that the book did not contain pictures or conversations. The verb "had" is in the past tense which indicates that **the action of the book not having pictures or conversations** is completed, and it happened in the past. An event can be a completed past action or something that is happening currently or in the future.

[...]

Could you interpret the phrase as a state as well?

It could be interpreted as a state in the sense that the book currently lacks pictures or conversations, however, in the context of the passage **it is clear that the author is describing a past event** where Alice had looked into the book and found that it did not contain pictures or conversations, **as opposed to the book being in a state of lacking pictures or conversations**. The past tense of the verb "had" and the context of Alice peeping into the book suggest it's an event that happened in the past. Additionally, the phrase "'what is the use of a book,' thought Alice, 'without pictures or conversation?'" also suggest that it's an event that happened in the past. So, it can be interpreted as a state, but in this context, it is more likely an event.

[*Even in this context, the verb phrase describes a state (a property of the book).*]

I agree that "peeped into the book" is an event. But what about "the book had no pictures in it"?

In the sentence "the book had no pictures in it", the phrase "had no pictures in it" describes the state of the book, specifically that it did not contain any pictures. The verb "had" is in the past tense, which indicates that this is a statement about the book's state at a specific point in the past. The phrase describes a state that existed at a certain point in the past. Even though "had" is in past tense, it is used to describe a state, not an event. Additionally, the phrase is not describing something that just happened, but a characteristic of the book, which implies a state.

[*This explanation is correct. However, the above conversation clearly illustrates that ChatGPT generates text explaining the aspectual nature of the various verb phrases without having a deeper understanding of their semantics and hence without being consistent in its relatively random interpretations.*]

Telicity is a property of an event type expressed by verbs, i.e., referring to events that have an endpoint. Atelic verbs refer to event types that do not have a built-in endpoint. Can you give me examples for each case?

Sure, here are some examples of telic verbs and the events they describe:

"She finished the book." (finish)

"He graduated from college." (graduate)

"They completed the project." (complete)

And here are some examples of atelic verbs and the events they describe:

"She is reading a book." (read)

"He is studying for the exam." (study)

"They are working on the project." (work)

[*These examples are correctly classified except for "read a book."*]

Why is the event type "read a book" atelic?

The event type "read a book" is considered atelic because it does not have a built-in endpoint. In other words, when someone says "She is reading a book," it does not necessarily imply that the act of reading will come to a definite conclusion. Reading a book can be an ongoing activity that may or may not be completed. **It does not have an inherent endpoint.**

[*Here, the analysis confuses between progressive aspect and analysing telicity at the event type (predicate) level. (Admittedly, this is a very fine-grained notion that humans also need to think carefully about.) In "She was reading a book", the event type "read a book" is telic, while the sentence in the progressive presents the ongoing situation of reading a book, hiding the end points.*]

Wouldn't the endpoint mean she finished reading the entire book?

Yes, in the sentence "She finished the book" the endpoint would mean she finished reading the entire book. The verb "finish" implies that an action has come to completion, so when it is used in the context of reading a book, it implies that the book has been read in its entirety. However, it's worth noting that some verbs can have multiple meanings and context can affect the interpretation of a sentence.

In contrast, when someone says "She is reading a book" it doesn't necessarily imply that the act of reading will come to a definite conclusion or that she will finish reading the entire book, she could stop reading at any time. And that's why the event is atelic.

I think the event is still telic, but the sentence focuses on a phase of it.

You are correct that the event of reading a book, in general, is telic, as it does have a natural endpoint, which is when the person finishes reading the book. However, when we use the verb "read" in the present progressive form "is reading," it describes the action as being ongoing and does not imply a definite conclusion. It describes the event as a phase or a state of being rather than a completed action.

So in the sentence "She is reading a book," the emphasis is on the current state of reading and not on the endpoint of finishing the book. The sentence "She finished the book" implies that the action of reading is completed and the verb "finish" implies that an action has come to completion.

So, in summary, while the overall event of reading a book is telic, in certain sentence constructions, such as "She is reading a book," the focus is on the ongoing state of the event, making it atelic.

[*The last part of the analysis is correct, except that the event of reading a book is always telic, and the progressive focuses on part of the situation. This does not make the event type atelic. The explanations given in this section focus more on the progressive vs. non-progressive distinction, not on the telic vs. atelic distinction.*]

F.2 Eventuality Types

(Prompted with the first two paragraphs of Section 3.3 of this paper including examples.)

Can you please analyse: "Tom held a meeting"

The sentence **"Tom held a meeting" is an example of an achievement** according to Vendler's (1957) classification of situation types. The sentence describes an event that has an endpoint, the meeting, and it is also a change of state. The verb "held" is also telic, meaning it has a specific endpoint and it does not include a process leading up to this point, which fits with the definition of an achievement. The phrase "held a meeting" implies that there was a start and an end to the meeting and it implies that it has been completed.

Can you please analyse: "Tom was holding a meeting"?

The sentence **"Tom was holding a meeting" is an example of an activity** according to Vendler's (1957) classification of situation types. The sentence describes an ongoing process of holding a meeting without any specific endpoint. The verb "was holding" is also atelic, meaning it does not have a specific endpoint, which fits with the definition of an activity. The phrase "holding a meeting" implies that there is a meeting in progress, but it does not specify when or if the meeting is completed.

[...]

[*The predicate "hold a meeting" is in fact an accomplishment, as it is dynamic, durative, and telic. The change in grammatical aspect from perfect to progressive does not change the underlying eventuality type. Again, the system seems to be misled by the progressive aspect.*]

Can you please analyse: "Anna was swimming across the lake"?

The sentence "Anna was swimming across the lake" is an example of an activity according to Vendler's (1957) classification of situation types. The sentence describes an ongoing process of swimming without any specific endpoint. The verb "was swimming" is also atelic, meaning it does not have a specific endpoint, which fits with the definition of an activity. **The phrase "across the lake" indicates that Anna is swimming over a significant distance, covering a large geographical area,** but the sentence does not specify when or if the journey is completed.

[*The default interpretation for this sentence is as an accomplishment, with the goal of reaching the other side of the lake. There is a second plausible, though less likely, interpretation in which Anna is engaged in the activity of swimming in a location that is across the lake from the speaker. ChatGPT proposes a third, highly-unlikely interpretation, akin to describing something as happening "all across the world."*]

Can you please analyse "Tim swam across the pool three times".

The sentence **"Tim swam across the pool three times" is an example of an accomplishment** according to Vendler's (1957) classification of situation types. The sentence describes a process (swimming) that leads to a built-in terminal point, the completion of swimming across the pool. The verb "swam" is telic, meaning it has a specific endpoint, which fits with the definition of an accomplishment. Additionally, the phrase "across the pool" indicates that Tim is swimming over a significant distance, covering a large geographical area and the sentence implies that the journey has been completed. The number "three times" indicates that the process was repeated a specific number of times, and it gives the sense that the accomplishment has been reached multiple times.

[*When substituting “pool” for “lake” and adding the frequency adverb “three times”, GPT’s interpretation of the event type of "swim across X" is correct, again hinting at the randomness of the generated explanations.*]

Incorporating Context into Subword Vocabularies

Shaked Yehezkel
Blavatnik School of Computer Science
Tel-Aviv University
Tel-Aviv, Israel
`shakedy@mail.tau.ac.il`

Yuval Pinter
Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
`uvp@cs.bgu.ac.il`

Abstract

Most current popular subword tokenizers are trained based on word frequency statistics over a corpus, without considering information about co-occurrence or context. Nevertheless, the resulting vocabularies are used in language models' highly contextualized settings. We present SAGE, a tokenizer that tailors subwords for their downstream use by baking in the contextualized signal at the vocabulary creation phase. We show that SAGE does a better job than current widespread tokenizers in keeping token contexts cohesive, while not incurring a large price in terms of encoding efficiency or domain robustness. SAGE improves performance on English GLUE classification tasks as well as on NER, and on Inference and NER in Turkish, demonstrating its robustness to language properties such as morphological exponence and agglutination.

BPE	_His _son _Raj ash ri _Sud h ak ar _has _p enn ed **_dial og ues** _and _songs _for _some _films _that _were _dubbed _into _Telugu .
SAGE	_His _son _Raj ash r i _Sud h a k a r _has _penn e d **_dial ogues** _and _songs _for _some _films _that _were _dubbed _into _Telugu .
BPE	_This _gene _is _a **_pseud og ene** _in _humans _and _most _other _prim ates .
SAGE	_This _gene _is _a **_pseud ogene** _in _humans _and _most _other _prim ates .
BPE	_The **_St o og es** _work _for _Mir acle _Det ective _Agency ,
SAGE	_The **_St o o g e s** _work _for _Mir acle _Det ective _Agency ,

Table 1: The token *og* is selected by BPE (vocabulary of size 16,000) for achieving the frequency objective, but is discarded by SAGE for failing to be contextually coherent. These examples from the corpus demonstrate some different contexts.

1 Introduction

Much of the research space in current NLP is focused on advancing models: modifying pre-training objectives, improving network architectures, adding tasks and schemes for downstream evaluation. Limited work is dedicated to a crucial step underlying all modern large language models (LLMs), namely the **tokenization** phase. In order to process a given string of text, an LLM must first obtain a vector representation of the input by segmenting it into tokens. Since out-of-vocabulary (OOV) items inhibit the performance of models, current tokenizers produce tokens which are possibly proper subsegments of input words, known as **subwords**. This method, popularized by systems such as WordPiece (Schuster and Nakajima, 2012), Byte-Pair Encoding (BPE; Sennrich et al., 2016) and UNIGRAMLM (Kudo, 2018), allows any word to be represented by one or more tokens, removing the OOV problem while allowing more flexibility in determining the token vocabulary size, which ultimately affects model speed (mostly through the softmax generation targets) and performance (through better ability to represent less-frequent words).

One potential pitfall of both BPE and UNIGRAMLM, as well as their proposed variants (He et al., 2020; Provilkov et al., 2020), is that they are trained on word frequency statistics alone, without considering information about word co-occurrence or contexts. At the same time, the resulting vocabularies are used in highly contextualized settings, the LLMs, where a single subword such as *og* might appear in very different contexts derived from words like *dial og ues* and *pseud og ene*. We propose a system which prepares subwords for their downstream use by baking in the contextualized signal **at the vocabulary creation step**. Our model, SAGE, uses the SKIPGRAM objective (Mikolov et al., 2013) over a corpus as the basis for iteratively eliminating candidate subwords from an initial large vocabulary until the desired vocabulary size has been reached. As Table 1 shows, SAGE succeeds in removing the ambiguous *og* token, facilitating distinct contextualization procedures for the example sentences

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 623–635
May 2-6, 2023 ©2023 Association for Computational Linguistics

(taken from Wikipedia).

We present our algorithm, SaGe, which is predicated on iterative pruning of contextually noisy tokens from the vocabulary, and compare its effects on token properties and context cohesion with BPE both in- and out-of-domain, in English and Turkish. We then evaluate its performance on downstream tasks by training a BERT-based LLM (Devlin et al., 2019) on a vocabulary produced by both tokenizers in both languages, demonstrating substantial improvements on most English GLUE tasks and on NER, as well as Turkish NLI and NER. We emphasize that as opposed to most current tokenizer variants, our model is a "plug and play" substitution for any subword token vocabulary, requiring no modification in the inference protocol (or code) when pre-training or applying an applicable LLM from a popular shared library.[1]

2 Subword Vocabulary Creation

The methods used to tokenize corpus in order to later assign tokens with continuous vectors, or *embeddings*, have evolved over the years. Initially, each word in the corpus was assigned its own embedding (Collobert and Weston, 2008; Mikolov et al., 2013). OOVs, i.e. words not appearing in the original training corpus or below a certain frequency threshold, would receive a special (but identical) "UNK" vector. Subword tokenizers (Schuster and Nakajima, 2012; Wu et al., 2016) were introduced to alleviate this issue, allowing segmentation of all text into embeddable units (assuming no unseen characters, a much more relaxed constraint for languages using alphabetical scripts). The training process used to create a subword vocabulary from which the model then decodes text input involves optimizing an encoding objective over a large corpus. To date, all tokenizers used in practice in large models focus on efficiency and information-theoretic objectives, and reduce the corpus to a unigram frequency count of space-delimited words, reducing calculation time but losing all contextual signal. SaGe reintroduces the contextual dependencies between words into vocabulary creation via a two-stage process, namely over-application of BPE followed by iterative pruning using ideas inspired by UnigramLM and SkipGram. We briefly present these algorithms before tying them together into SaGe.

Algorithm 1 Byte-pair encoding vocabulary creation (Gage, 1994; Sennrich et al., 2016)

```
Input: Corpus $C$, Vocabulary final size $V$.
Output: Vocabulary $\mathcal{V}$ of size $V$ (ordered).
procedure BPE($C, V$)
    $\mathcal{V} \leftarrow$ All unique characters in $C$
    while $|\mathcal{V}| < V$ do                              ▷ Merge tokens
        $\langle t_L, t_R \rangle \leftarrow$ Most frequent bigram in $C$
        $t_{NEW} \leftarrow t_L \oplus t_R$                    ▷ Make new token
        $\mathcal{V} \leftarrow \mathcal{V} \oplus [t_{NEW}]$
        $C$.ReplaceAll($\langle t_L, t_R \rangle, t_{NEW}$)
    end while
return $\mathcal{V}$
end procedure
```

Byte-Pair Encoding. The BPE algorithm creates a vocabulary "bottom-up", starting with all single characters from the alphabet, iteratively adding tokens until reaching the desired vocabulary size. In each iteration, the added token is the concatenation of the most frequent adjacent pair of existing tokens (see Algorithm 1). The default setting of the algorithm's most popular implementation (Kudo and Richardson, 2018) restricts token addition within word boundaries, facilitating training from unigram frequencies. In addition, LLM tokenizers using BPE (Liu et al., 2019; Radford et al., 2019; Wolf et al., 2020) decode sequences not by applying merges by order of the vocabulary, as originally dictated by the algorithm, but through greedy largest-subsequence left-to-right inference.

Unigram Language Model. UnigramLM offers a top-down vocabulary creation process, starting with an initial vocabulary of all substrings in the input corpus and pruning tokens iteratively until reaching the desired vocabulary size. The pruning procedure involves calculating the overall unigram likelihood of the corpus with the current vocabulary versus a vocabulary lacking the candidate pruning token (see Algorithm 2 for details), which we refer to as the **ablation objective**. Under this system, decoding is ideally performed by considering probabilities of all possible segmentations using, e.g., the Viterbi algorithm; again, common practice is to use left-to-right greedy decoding.

Skipgram Objective. The SkipGram objective (Mikolov et al., 2013) formalizes the relation between a target token t and its context, asking whether context tokens c within a window W_t of pre-defined size can be predicted from t. These predictions are done via sigmoid activation over the inner product of embeddings trained for targets ($\boldsymbol{E}^{(T)}$) and contexts ($\boldsymbol{E}^{(C)}$). When aggregated over

---

[1] Our code and models are available at `www.github.com/MeLeLBgu/SaGe`.

Algorithm 2 UnigramLM vocabulary creation (Kudo, 2018). $n \arg\min_X$ denotes the n bottom-ranked elements in X.

```
Input: Corpus $C$, Vocabulary final size $V$, pruning batch size $k$.
Output: Vocabulary $\mathcal{V}$ of size $V$.
procedure UnigramLM($C, V$)
    $\mathcal{V} \leftarrow$ All substrings occurring more than once in $C$
    while $|\mathcal{V}| > V$ do                          ▷ Prune tokens
        $X^{(j)} \leftarrow$ tokenize($C$, $\mathcal{V}$)
        $\mathcal{L}(\mathcal{V}) \leftarrow \sum_{j=1}^{|C|} \log\left(P(X^{(j)})\right)$
        for all $t \in \mathcal{V}$ do:      ▷ Calculate ablation objective
            $loss_t \leftarrow \mathcal{L}(\mathcal{V} \setminus \{t\}) - \mathcal{L}(\mathcal{V})$
        end for
        $\mathcal{P} \leftarrow \min(k, |\mathcal{V}| - V) \arg\min_{t \in \mathcal{V}}(loss_t)$
        $\mathcal{V} \leftarrow \mathcal{V} \setminus \mathcal{P}$                          ▷ Prune
    end while
return $\mathcal{V}$
end procedure
```

Algorithm 3 SaGe vocabulary creation. $n \arg\min_X$ denotes the n bottom-ranked elements in X.

```
Input: Corpus $C$, Vocabulary final size $V$, basic tokenizer $\mathcal{T}$, overshoot factor $n$, pruning batch size $k$, likelihood recalculation frequency $m$, size of pruning candidate set $M$, embedding recalculation frequency $l$.
Output: Vocabulary $\mathcal{V}$ of size $V$.
procedure SaGe($C, V$)
    $\mathcal{V} \leftarrow \mathcal{T}(C, n \cdot V)$
    $i \leftarrow 0$
    while $|\mathcal{V}| > V$ do
        if $i \equiv 0 \pmod{l \times m}$ then
            $\boldsymbol{E}^{\mathcal{V}} \leftarrow$Word2Vec($\mathcal{V}$)          ▷ Embedding table
        end if
        $\mathcal{L}(\mathcal{V}, \mathcal{C}) \leftarrow$ SGObj($\boldsymbol{E}^{\mathcal{V}}, \mathcal{C}$) ▷ Total likelihood (1)
        if $i \equiv 0 \pmod{m}$ then              ▷ Update bottom set
            for all $t \in \mathcal{V}$ do:
                $loss_t \leftarrow \mathcal{L}(\mathcal{V} \setminus \{t\}, \mathcal{C}) - \mathcal{L}(\mathcal{V}, \mathcal{C})$
            end for
            $\mathcal{V}_{bot} \leftarrow M \arg\min_{t \in \mathcal{V}}(loss_t)$
        else                           ▷ Update losses for bottom set
            for all $t \in \mathcal{V}_{bot}$ do:
                $loss_t \leftarrow \mathcal{L}(\mathcal{V} \setminus \{t\}, \mathcal{C}) - \mathcal{L}(\mathcal{V}, \mathcal{C})$
            end for
        end if
        $\mathcal{P} \leftarrow \min(k, |\mathcal{V}| - V) \arg\min_{t \in \mathcal{V}_{bot}}(loss_t)$
        $\mathcal{V}_{bot} \leftarrow \mathcal{V}_{bot} \setminus \mathcal{P}$                         ▷ Prune
        $\mathcal{V} \leftarrow \mathcal{V} \setminus \mathcal{P}$
        $i \leftarrow i + 1$
    end while
return $\mathcal{V}$
end procedure
```

all tokens in a corpus, SkipGram can be used as a total likelihood measure, approximating its overall contextual cohesion:

$$\mathcal{L}(\mathcal{V}, \mathcal{C}) = -\sum_{t \in \mathrm{tok}(\mathcal{C}, \mathcal{V})} \sum_{c_j \in W_t} \log\left(\sigma(\boldsymbol{E}_t^{(T)} \cdot \boldsymbol{E}_{c_j}^{(C)})\right). \tag{1}$$

As token vocabularies or their inference methods change, so do the target sequences and their contexts, resulting in differences in aggregated likelihood which can then act as scores comparing one tokenization to another. We use this behavior as the ablation objective for SaGe.

3 SaGe Vocabulary Creation

SaGe[2] is a top-down tokenizer, following UnigramLM's general procedure, incorporating a SkipGram objective as its vocabulary trimming rule. Given an initial vocabulary $\mathcal{V}$ and a corpus $\mathcal{C}$, SaGe computes a SkipGram embedding space over $\mathcal{V}$ which provides it with an overall likelihood over $\mathcal{C}$ as in (1). It then proceeds to calculate the *loss* of each token in the vocabulary were it to be removed, eliminating the tokens incurring minimal loss and re-tokenizing the corpus according to the updated vocabulary, repeating this procedure until reaching the desired vocabulary size V. Having learned this vocabulary, downstream inference proceeds exactly as in the other segmentation-based methods, in a greedy left-to-right manner. SaGe can also be adapted to anticipate other decoding algorithms, by changing the re-tokenization steps accordingly.

In practice, applying the full process described above introduces multiple sources of considerable computational complexity: for example, calculating the ablation objective for each token in each iteration produces a quadratic amount of calculations over the entire corpus; recalculating embeddings for an updated vocabulary is similarly unreasonable to perform at each iteration. We ameliorate these and other sources of complexity using a series of heuristics found in preliminary experiments to be minimally disruptive to precision of likelihood calculations. We will now describe these heuristics, all depicted in Algorithm 3. First, instead of initializing the vocabulary as the full set of possible character sequences in the corpus, as in UnigramLM, we use any existing noncontextual tokenizer such as BPE to learn a vocabulary larger than V by a factor of n, and begin the pruning process from there. Next, instead of removing a single token from the bottom of the loss-ranked vocabulary, we remove a batch of the k bottom tokens

---

[2] The name is not an acronym; it is intended to evoke SkipGram while maintaining the “suffix” of *BPE*.

Sentence fragment	. . . use of an include directive is when referring to . . .
Tokenization using $\mathcal{V}$	_use _of _an _includ [e _direct **ive** _is _when] _ref er r ing _to
Tokenization using $\mathcal{V} \setminus \{\_\texttt{includ}\}$	_use _of _an _inc l u [de _direct **ive** _is _when] _ref er r ing _to

Table 2: The effect of retokenization on a context window of width 2 (in brackets) surrounding a target token (in bold). A left-side context token has been replaced as a result of an out-of-window vocabulary ablation.

each time, as does UNIGRAMLM.[3] To avoid frequent loss recalculation, we recompute the entire likelihood set once every m ablation steps, and only keep the bottom M tokens as pruning candidates for the next m steps. Our preliminary experiments support this decision, as we found the ranked list of losses tends to stay relatively stable over dozens of batch-pruning iterations. Lastly, to avoid the costly re-training of the embedding matrix for all tokens given the updated corpus, which only results in minor changes in likelihood during subsequent iterations, we only perform it every l iteration batches, i.e. after the ablation of $k \times m \times l$ tokens. n, k, l, m and M are all algorithm hyperparameters tuned empirically based on desired runtime, corpus size and vocabulary size.

Contextual Loss. In order to calculate the per-token SKIPGRAM likelihood loss, all sentences where a token t occurs need to be re-segmented according to $\mathcal{V} \setminus \{t\}$, and their new likelihoods recorded. To support performing this calculation on a large scale, we maintain a mapping of tokens to sentences containing them, as well as these sentences' current likelihoods. This must be done at the sentence level rather than the window level, since a remaining suffix from an out-of-window re-tokenization may combine with in-window characters and form different token sequence replacements at a given stage. Consider the example in Table 2, where re-tokenization results in the replacement of a context token for a distant target.

Negative Sampling. The original SKIPGRAM objective uses negative samples to estimate context probabilities. Since our application of SKIPGRAM within the vocabulary creation algorithm (independent of the embeddings training procedure) includes only likelihood estimation with no parameter updates, we do not sample negative tokens, a process which would introduce substantial noise and complexity.

4 SAGE Vocabulary Properties

For an analysis of our modified algorithm's advantages, we trained vocabularies of a pre-determined size using both BPE and SAGE. We selected $|\mathcal{V}| = 16,000$, and obtained corpora for English (750,000 lines from the August 2022 Wikipedia dump) and Turkish (the entire text of the September 2022 Wikipedia dump), opting for languages that share the Latin alphabet but differ in family (Indo-European vs. Turkic) and, crucially, in morphological properties: English is a low-exponence, low-synthesis language, while Turkish features multiple inflectional exponence and high verbal synthesis, as well as highly agglutinative morphology (Bickel and Nichols, 2013a,b). We used the following hyperparameter settings to compute the vocabularies: Initial vocab size $20,000$ (or $n = 1.25$), $l = 4$, $k = 100$, $M = 1500$, $m = 10$. We used the Gensim package to train the SKIPGRAM models (Rehurek and Sojka, 2011), and Sentencepiece (Kudo and Richardson, 2018) to obtain the initial BPE vocabularies.[4] More hyperparameters are detailed in Appendix A.

We present an analysis of the resulting vocabularies, highlighting the advantages and trade-offs exhibited by context-based subword tokenization. Generally speaking, most of the tokens discarded from SAGE's initial vocabulary appear in the baseline BPE's final vocabulary. Among the differences between the vocabularies are many short tokens that appear in BPE's but not SAGE's, proper substrings of longer tokens also appearing in the BPE vocabulary. This is due to BPE's bottom-up merge table construction, which forces retention of the entire chain of tokens created: if *the* is part of the vocabulary, either *th* or *he* must also be there. While essential for the original intended decoding process, actual implementations of greedy decoding have no need for this property. SAGE's initial vocabulary shares this characteristic, but the trimming process allows any token to be ablated,

---

[3] As in UNIGRAMLM and other ablation-based vocabularies, single-character tokens are never removed from the vocabulary, in order to allow for all in-alphabet words to be tokenized.

[4] Since BPE augments its vocabulary iteratively, the baseline BPE vocabulary is a proper subset of that used to initialize SAGE.

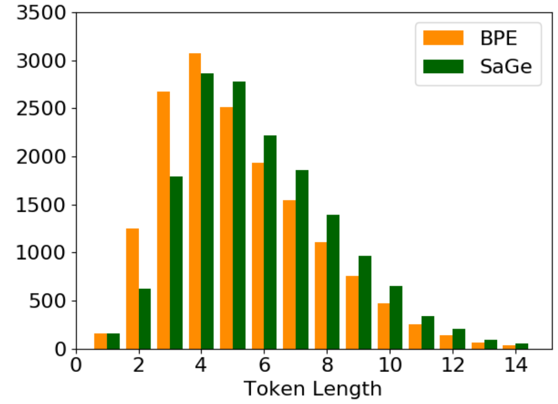


Figure 1: Token length distribution of BPE's vocabulary vs. SAGE's on English.

More frequent in			
SAGE		BPE	
e	s	es	ic
ing	ist	ings	ff
ation	ate	ations	ates

Table 3: Tokens with high difference in frequency between tokenizations (English models).

including those in the middle of merge chains. Another difference found between the vocabularies is the strong preference of SAGE for word-initial tokens. 83% of the tokens that appear in SAGE's vocabulary but not in BPE's are word-initial, compared to only 22% of the BPE-only tokens. This is reasonable, since a token surviving SAGE's ablation steps exhibits high loss for the condition of its removal, which is arguably the case when a nearby target word needs to predict a word-initial context.

Token Length. Figure 1 shows a histogram of token lengths (in characters) for the 16,000-token SAGE and BPE vocabularies in English (results on Turkish are similar). SAGE clearly selects longer tokens for its vocabulary, again a sensible outcome given their higher chance of being contextually coherent. The difference is most stark with tokens of length 2 and 3; when considering only tokens appearing in exactly one of the final vocabularies, we find that 56% of BPE-only tokens are of length 2 and 3, while 55% of SAGE-only tokens are of length 5 and above.

Token Frequency. We compute the frequency of tokens in the encoding form of the English training corpus, once using SAGE vocabulary and once using BPE's. In Table 3 we show some of the tokens with the biggest difference in frequency between SAGE and BPE tokenizations. We can see SAGE reverts to single-character tokens considerably more often than BPE (also demonstrated in the last example in Table 1). We view this as a feature of context-based tokenization—its vocabulary is partitioned between (mostly short) tokens that are highly ambiguous in context and (mostly long) tokens that have coherent contexts. At the same time, BPE is rife with tokens that are medially ambiguous contextually, whose resulting embeddings can be neither useful nor completely ignorable, adding noise to the representation sequences. As a result, SAGE breaks down complex suffixes, which in English are compositional, into their constituent morphology. The suffix *ings* is thus dismantled to *ing s*, whereas BPE reserves a token for it, mostly unhelpful in itself.

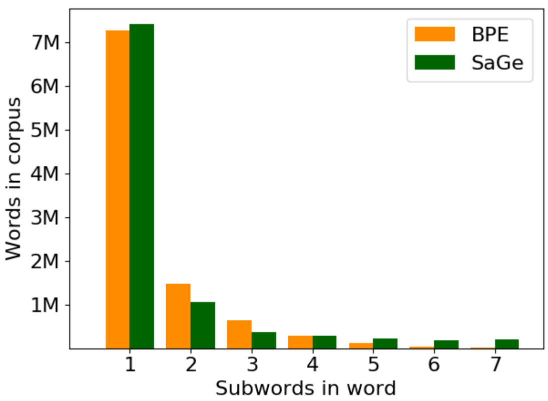


Figure 2: Number of subwords required to tokenize a word, collected over the original English training corpus.

Subword Fertility. Fertility, as defined in the statistical machine translation literature, refers to the average number of subwords produced per tokenized word. Figure 2 exhibits a histogram of all English corpus words by their subword length, using the BPE vocabulary and the SAGE vocabulary. Although SAGE retains more words as single tokens, it trades them off with more words having five subwords or more, compared with BPE's abundance of words with 2 and 3 subwords. This follows the trend described so far, of SAGE's preference for dismantling unknown words into meaningless single-character tokens rather than confusing, ambiguous length-2 and length-3 tokens. We believe that BPE's behavior harms text understand-

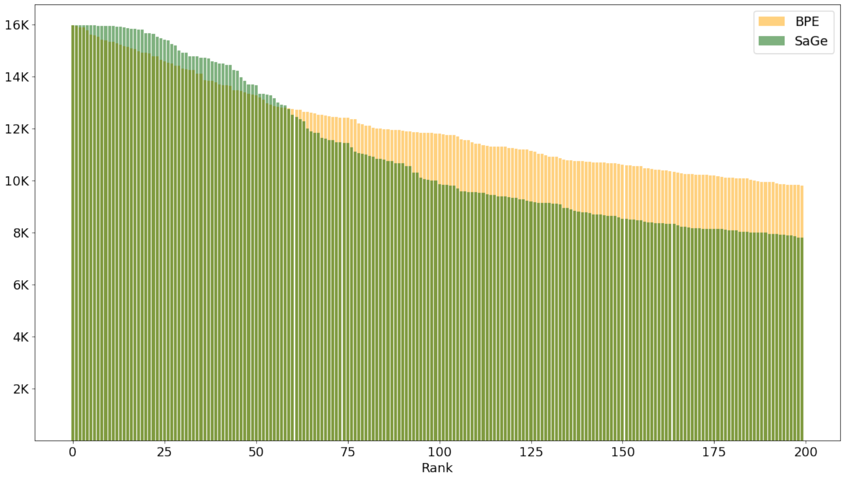


Figure 3: Number of distinct neighbors each token encounters in a width-5 window, top 200, Turkish.

ing in suggesting that these ambiguous fragments (consider "og") have some meaning that an LLM can try and learn, whereas SAGE's single-character breakdown indicates a word that's truly unknown and cannot be inferred by composing constituent in-vocab subwords.

Fertility translates to a trade-off in **encoding efficiency** to SAGE's contextual advantage: a sample of 150K lines from English Wikipedia is encoded by 4 million BPE tokens, optimizing only an information-theoretic objective, whereas SAGE produces 4.5 million. Having said that, this inefficiency might be further offset during LLM pretraining: we propose that contextually coherent tokens will require fewer update steps in order to achieve useful embedding parameters, helping the model converge faster compared to BPE tokens. We leave testing this hypothesis to future work.

Contextual Exponence. To determine the degree to which SAGE effectively optimizes tokens' contextual soundness, which is its ultimate goal, we plot the number of distinct neighbors each token encounters throughout the training corpus, ranked from high to low, in Figure 3. The very top of the ranking is occupied by single-character tokens which are context-null by design, which SAGE makes the most of by placing in almost all contexts. After a few dozen tokens, SAGE's context counts dip below BPE's, a trend which continues all the way through the vocabulary, making up a more contextually coherent set. These findings hold for English as well as Turkish, and replicate when taking a context window of size 2, different from that used during SAGE construction.

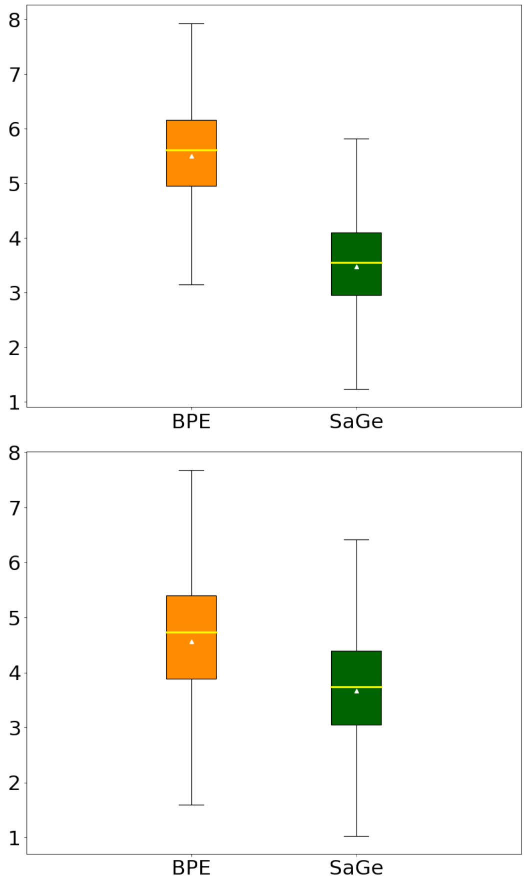


Figure 4: Distribution of token neighbors/frequency ratio for a width-5 window in English (top) and Turkish (bottom); BPE (left) and SAGE (right)

These findings can arguably be attributed to a frequency artifact, where SAGE simply outputs more tokens with lower frequency in order to provide them with fewer contexts. We thus present a normalized analysis in Figure 4, depicting the ratio between each token's number of unique neighbors and its frequency, distributed over the entire vocabulary. SAGE provides substantially lower ratios in both languages, supporting our original claim.

4.1 Robustness to Domain Change

One possible limitation of the SAGE objective is that it increases the reliance on the original training corpus compared to word-count-only algorithms. In and of itself, this should not necessarily be viewed as a problem, assuming the collected corpus is a faithful representative of an LLM's use case.[5] To this end, we collected comparable corpora from non-Wikipedia domains and ran our analysis on the SAGE and BPE vocabularies trained on Wikipedia. Our findings suggest that while SAGE loses its relative advantage in context-dependence over BPE, it does not fall behind it (i.e. it has not overfit to the Wikipedia domain). We present a fertility chart for an English corpus of 7.5M words from Quora questions[6] in Figure 5, depicting similar trends to that on Wikipedia (Figure 2) but with smaller differences between SAGE and BPE; the neighbor-to-frequency ratio aggregation chart in Figure 6 differs from Figure 4 (top) substantially but shows that SAGE and BPE tokens do not diverge significantly on this measure. We repeated the experiment on English legal text centered on US congress bills (Henderson et al., 2022) and on a 2.6M-word Turkish corpus of online reviews,[7] and observed similar trends.

These results indicate that while a considerable amount of the longer tokens preferred by SAGE were selected to optimize contextuality in the source domain, as it was designed to do, there is no "short blanket" effect for text originating in different domains. This could either be due to wide-scope advantages of some of the tokens selected by SAGE, or due to an intrinsic deficiency in BPE's long-tail tokens, or a combination of both.

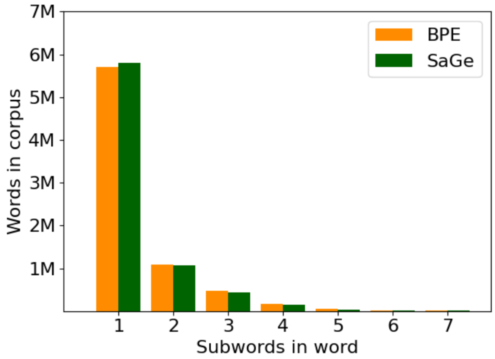


Figure 5: Number of subwords required to tokenize a word using the original Wikipedia-trained vocabularies, collected over a English Quora questions corpus.

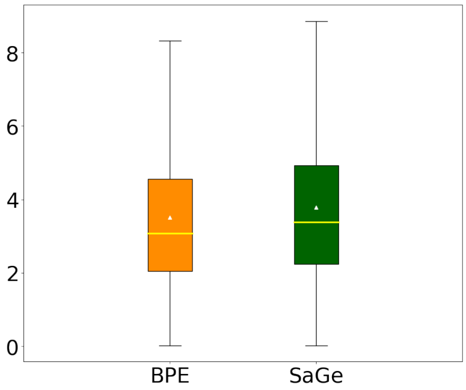


Figure 6: Distribution of token neighbors/frequency ratio for a width-5 window in English, based on a Wikipedia-trained vocabulary and collected over a English Quora questions corpus.

5 Downstream Evaluation

In order to evaluate the utility of our tokenization algorithm for major NLP tasks, we compare SAGE to a BPE vocabulary of the same size by means of pre-training a BERT-parameterized model (Devlin et al., 2019) using an expedited training scheme (Izsak et al., 2021). We then evaluate the LLM's performance both on sequence classification via the English GLUE benchmark (Wang et al., 2018) and the Turkish partition of XNLI (Conneau et al., 2018), and on named entity recognition in English (Wang et al., 2019) and Turkish (Al-Rfou et al., 2015). We use the default settings from Huggingface's library implementations of the fine-tuning processes (Wolf et al., 2020) and do not perform hyperparameter tuning for either model.

We present our results on sequence-level tasks in Table 4. SAGE tokenization improves performance on nearly all tasks with particularly sub-

---

[5] Indeed, existing literature recommends adding pre-training steps on new domains before fine-tuning models for them (e.g., Han and Eisenstein, 2019).

[6] `https://huggingface.co/datasets/chenghao/quora_questions`

[7] `https://huggingface.co/datasets/cansen88/turkishReviews_5_topic`

	MRPC (F1)	MNLI (Acc %)	COLA (Matt.)	QNLI (Acc %)	SST2 (Acc %)	STSB (Pear.)	QQP (Acc %)	XNLI$_{tur}$ (Acc %)
BPE	.7918	62.76	.0777	66.17	80.54	.3094	82.75	41.20
SAGE	.8004	64.00	.0985	74.83	79.85	.3387	84.69	46.46

Table 4: Performance on sequence-level tasks for BERT models trained on different 16k-size vocabularies. XNLI$_{tur}$ is Turkish, the rest are English GLUE tasks. All results averaged over three runs on the dev set with different seeds.

	English	Turkish
BPE	.7142	.4660
SAGE	.7502	.5475

Table 5: Performance (F1) on NER tasks of BERT Turkish and English models trained on different subword vocabularies of size 16,000. All results averaged over three runs on the dev set with different seeds.

stantial improvements (1.3–8 accuracy points) on NLI datasets. Results on NER are presented in Table 5, again showing SAGE's dominance over BPE. Due to the length of the training pipeline leading from vocabulary creation through pre-training to fine-tuning, it is difficult to find individual examples where difference in tokenization leads to direct changes in prediction; we attribute the consistent overall gains in downstream performance mostly to the LLM pre-training step, where the design of SAGE's context-friendly vocabulary enables a more coherent contextual signal to flow through the transformer layers during backpropagation. We note that in general, our models fare worse on GLUE tasks compared to Izsak et al. (2021). We attribute this in part to the smaller token vocabulary size, and more substantially to the smaller pre-training corpus we used in our experiments.

6 Related Work

In recent years, a growing body of research has demonstrated the shortcomings of existing tokenization algorithms in the context of representing linguistic phenomena in different languages across different tasks (Banerjee and Bhattacharyya, 2018; Klein and Tsarfaty, 2020; Hakimi Parizi and Cook, 2020; Rust et al., 2021; Maronikolakis et al., 2021; Mielke et al., 2021; Hofmann et al., 2021), as well as the statistical properties affecting their downstream performance (Bostrom and Durrett, 2020). Our work addresses the concerns raised in this line of work by introducing an improved subword vocabulary creation method which leverages the contextual aspects of the main intended use case, namely LLMs. Previous work towards this goal includes algorithms which offer robustness within an existing subword vocabulary (Provilkov et al., 2020; He et al., 2020; Hiraoka, 2022), necessitating modification of either training, inference, or both procedures in the context of LLMs. Others have considered tuning the *size* of a subword vocabulary (Salesky et al., 2020), or selecting from an enlarged set of possible segmentations (Asgari et al., 2020), for optimizing performance on downstream tasks.

Some alternative tokenization methods focus on the application of a model which considers the expected downstream tasks together with the pretraining corpus (Hiraoka et al., 2020), to the degree of jointly optimizing the tokenizer with the downstream model (Hiraoka et al., 2021). In addition to the massive changes in training and inference procedures this approach incurs, we note that it is difficult to apply to large contextualized models due to the long path from tokenization to prediction; SAGE overcomes this problem by "nudging" only the LLM vocabulary itself towards a contextualization-friendly segmentation.

The concept of subword tokenization made its rise alongside that of contextualized representations, meaning that little work exists where SKIP-GRAM or other static models are trained over proper subword segmentations. Recently, Kaushal and Mahowald (2022) did so for a proof-of-concept of a spelling prediction model, in lieu of training full LLMs. To our knowledge, no work to date has used a static embedding-based objective to score token sequence likelihood for a separate task (as we do for vocabulary trimming).

Finally, we acknowledge the recent efforts to do away with tokenization altogether, be it through character-only (Clark et al., 2022) or byte-only (Xue et al., 2022) models, or through encoding characters visually and passing them through a vision model (Salesky et al., 2021; Rust et al., 2022).

These represent an even more radical departure from the established application of LLMs, and we look forward to testing their abilities against our improved contextual subword tokenization methods. We note that while these models have been facing issues regarding scaling, mostly on the decoding side, SAGE vocabularies are ready to be used immediately within existing popular LLM implementations. Furthermore, recent work has shown the limited utility of character-level transformers in semantic tasks, even for morphologically rich languages with nontrivial orthography-morphology relations (Keren et al., 2022).

7 Conclusion

In this work, we introduced SAGE, a context-aware tokenizer built using insights from BPE, UNIGRAMLM, and SKIPGRAM, and showed that it achieves better results when used in an LLM-pre-train-then-fine-tune schema on two typologically distant languages on both the sequence and token levels. We believe that further investigation into incorporating context in tokenization models can improve results even further, and intend to also extend our efforts toward other languages and writing systems, as well as to multilingual tokenizers. For example, we plan to apply SAGE in the context of Abjads like Hebrew and Arabic, as well as languages written in alphasyllabaries such as Devanagari.

Within SAGE itself, there is room for improvement. The algorithm is still relatively slow, taking roughly a day to run on a strong CPU, making it difficult to apply to a truly large corpus, to start from a larger initial vocabulary, or to conduct exhaustive search over the hyperparameters. We intend to keep optimizing it, and continue evaluation against other subword and character-only schemas.

Limitations

We acknowledge several limitations of SAGE, a novel algorithm still in its development stages. First, scaling the vocabulary creation framework up from corpus-level unigram statistics to context dependence incurs many points where linear factors turn into quadratic, and worse. We introduced several heuristics to alleviate this issue in §3, however SAGE still takes longer to train compared to BPE and other tokenizers, by roughly a factor of ten. While having no effect on downstream pre-training and fine-tuning steps, it does mean hyperparameters are more difficult to tune. Second, the prohibitive resources required to implement a full LLM pipeline has limited our downstream evaluation setup to ten individual tasks on two languages. Ideally, as more languages with more diverse scripts and typological properties are examined, better generalizations can be made about the utility of integrating context into subword tokenizer vocabularies. Finally, we still do not have a well-formed theory of integrating multiple domains, languages, or scripts together into a single vocabulary. This question has interested researchers in recent years (e.g., Chung et al., 2020; Rust et al., 2021; Zhang et al., 2022), yet a tokenizer-internal solution (as opposed to data balance manipulation) still seems to have eluded the community. This question affects SAGE more than other tokenizers, given its reliance on context, which changes starkly when considering multiple sources of text in unison.

Acknowledgments

We thank Jacob Eisenstein and Cassandra Jacobs for work on earlier versions of the high-level idea, and Timo Schick and Lütfi Kerem Senel for fruitful conversations in early stages of the project. We thank Marco Cognetta, Michael Elhadad, Omer Levy, and attendees of ISCOL 2022 for comments and suggestions on more recent versions of the work. We thank the reviewers for their helpful comments. We thank Kaj Bostrom, Peter Izsak, and Tamar Levy for helping us obtain and operate resources for training our models.

References

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. 2015. Polyglot-ner: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 586–594. SIAM.

Ehsaneddin Asgari, Masoud Jalili Sabet, Philipp Dufter, Christopher Ringlstetter, and Hinrich Schütze. 2020. Subword sampling for low resource word alignment. *arXiv preprint arXiv:2012.11657*.

Tamali Banerjee and Pushpak Bhattacharyya. 2018. Meaningless yet meaningful: Morphology grounded subword-level nmt. In *Proceedings of the Second Workshop on Subword/Character Level Models*, pages 55–60.

Balthasar Bickel and Johanna Nichols. 2013a. Exponence of selected inflectional formatives. In Matthew S. Dryer and Martin Haspelmath, editors, *The World Atlas of Language Structures Online*.

Max Planck Institute for Evolutionary Anthropology, Leipzig.

Balthasar Bickel and Johanna Nichols. 2013b. Inflectional synthesis of the verb. In Matthew S. Dryer and Martin Haspelmath, editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Hyung Won Chung, Dan Garrette, Kiat Chuan Tan, and Jason Riesa. 2020. Improving multilingual models with language-clustered vocabularies. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4536–4546, Online. Association for Computational Linguistics.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

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

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38.

Ali Hakimi Parizi and Paul Cook. 2020. Evaluating sub-word embeddings in cross-lingual models. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2712–2719, Marseille, France. European Language Resources Association.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248, Hong Kong, China. Association for Computational Linguistics.

Xuanli He, Gholamreza Haffari, and Mohammad Norouzi. 2020. Dynamic programming encoding for subword segmentation in neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3042–3051, Online. Association for Computational Linguistics.

Peter Henderson, Mark S. Krass, Lucia Zheng, Neel Guha, Christopher D. Manning, Dan Jurafsky, and Daniel E. Ho. 2022. Pile of law: Learning responsible data filtering from the law and a 256gb open-source legal dataset.

Tatsuya Hiraoka. 2022. Maxmatch-dropout: Subword regularization for wordpiece. *arXiv preprint arXiv:2209.04126*.

Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. 2020. Optimizing word segmentation for downstream task. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1341–1351, Online. Association for Computational Linguistics.

Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. 2021. Joint optimization of tokenization and downstream model. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 244–255, Online. Association for Computational Linguistics.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train BERT with an academic budget. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10644–10652, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ayush Kaushal and Kyle Mahowald. 2022. What do tokens know about their characters and how do they know it? *arXiv preprint arXiv:2206.02608*.

Omri Keren, Tal Avinari, Reut Tsarfaty, and Omer Levy. 2022. Breaking character: Are subwords good enough for mrls after all? *arXiv preprint arXiv:2204.04748*.

Stav Klein and Reut Tsarfaty. 2020. Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 204–209, Online. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Antonis Maronikolakis, Philipp Dufter, and Hinrich Schütze. 2021. Wine is not v i n. on the compatibility of tokenizations across languages. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2382–2399, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sabrina J Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y Lee, Benoît Sagot, et al. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp. *arXiv preprint arXiv:2112.10508*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2):2.

Phillip Rust, Jonas F Lotz, Emanuele Bugliarello, Elizabeth Salesky, Miryam de Lhoneux, and Desmond Elliott. 2022. Language modelling with pixels. *arXiv preprint arXiv:2207.06991*.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Elizabeth Salesky, David Etter, and Matt Post. 2021. Robust open-vocabulary translation from visual text representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7235–7252, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Elizabeth Salesky, Andrew Runge, Alex Coda, Jan Niehues, and Graham Neubig. 2020. Optimizing segmentation granularity for neural machine translation. *Machine Translation*, 34(1):41–59.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Zihan Wang, Jingbo Shang, Liyuan Liu, Lihao Lu, Jiacheng Liu, and Jiawei Han. 2019. CrossWeigh: Training named entity tagger from imperfect annotations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5154–5163, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Shiyue Zhang, Vishrav Chaudhary, Naman Goyal, James Cross, Guillaume Wenzek, Mohit Bansal, and Francisco Guzman. 2022. How robust is neural machine translation to language imbalance in multilingual tokenizer training? In *Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 97–116, Orlando, USA. Association for Machine Translation in the Americas.

Final Vocab Size	16K
Initial Vocab Size	20K
k (tokens to prune each batch)	100
M (size of pruning candidate set)	1500
m (likelihood recalculation frequency)	10
l (embedding recalculation frequency)	4
SAGE window size	5
Word2Vec window size	5
Word2Vec vector dimension	50
Word2Vec negative samples	15

Table 6: Hyperparameters for vocabulary creation.

A Hyperparameters

In Table 6, 7, and 8, we present the hyperparameters used for training the various elements in our experiments.

B Computing Resources

For our experiments we used Quadro RTX 8000 GPU.

layer_norm_type	pytorch
model_type	bert-mlm
hidden_act	gelu
hidden_size	1024
num_hidden_layers	24
num_attention_heads	16
intermediate_size	4096
hidden_dropout_prob	0.1
attention_probs_dropout_prob	0.1
encoder_ln_mode	pre-ln
lr	1e-3
train_batch_size	4032
train_micro_batch_size_per_gpu	32
lr_schedule	time
curve	linear
warmup_proportion	0.06
gradient_clipping	0.0
optimizer_type	adamw
weight_decay	0.01
adam_beta1	0.9
adam_beta2	0.98
adam_eps	1e-6
total_training_time	24.0
optimizer_type	adamw
validation_epochs	3
validation_epochs_begin	1
validation_epochs_end	1
validation_begin_proportion	0.05
validation_end_proportion	0.01
validation_micro_batch	16
deepspeed	yes
data_loader_type	dist

Table 7: Hyperparameters for pre-training BERT-architecture models using the academic-budget-bert code (Izsak et al., 2021).

max_seq_length	128
evaluation_strategy	steps
per_device_train_batch_size	16
gradient_accumulation_steps	1
per_device_eval_batch_size	16
learning_rate	5e-5
weight_decay	0.1
max_grad_norm	1.0
lr_scheduler_type	polynomial
warmup_steps	50

Table 8: Hyperparameters for fine-tuning tasks using scripts from the academic-budget-bert package.

LoRaLay: A Multilingual and Multimodal Dataset for *Lo*ng *Ra*nge and *Lay*out-Aware Summarization

Laura Nguyen[1,3] **Thomas Scialom**[2*] **Benjamin Piwowarski**[3] **Jacopo Staiano**[4*]
[1]reciTAL, Paris, France [2]Meta AI, Paris, France
[3]Sorbonne Université, CNRS, ISIR, F-75005 Paris, France
[4]University of Trento, Italy

laura@recital.ai tscialom@fb.com
benjamin.piwowarski@cnrs.fr jacopo.staiano@unitn.it

Abstract

Text Summarization is a popular task and an active area of research for the Natural Language Processing community. It requires accounting for long input texts, a characteristic which poses computational challenges for neural models. Moreover, real-world documents come in a variety of complex, visually-rich, layouts. This information is of great relevance, whether to highlight salient content or to encode long-range interactions between textual passages. Yet, all publicly available summarization datasets only provide plain text content. To facilitate research on how to exploit visual/layout information to better capture long-range dependencies in summarization models, we present *LoRaLay*, a collection of datasets for long-range summarization with accompanying visual/layout information. We extend existing and popular English datasets (arXiv and PubMed) with visual/layout information and propose four novel datasets – consistently built from scholar resources – covering French, Spanish, Portuguese, and Korean languages. Further, we propose new baselines merging layout-aware and long-range models – two orthogonal approaches – and obtain state-of-the-art results, showing the importance of combining both lines of research.

1 Introduction

Deep learning techniques have enabled remarkable progress in Natural Language Processing (NLP) in recent years (Devlin et al., 2018; Raffel et al., 2019; Brown et al., 2020). However, the majority of models, benchmarks, and tasks have been designed for unimodal approaches, i.e. focusing exclusively on a single source of information, namely plain text. While it can be argued that for specific NLP tasks, such as textual entailment or machine translation, plain text is all that is needed, there exist several tasks for which disregarding the visual appearance of text is clearly sub-optimal: in a real-world context (business documentation, scientific articles, etc.), text does not naturally come as a sequence of characters, but is rather displayed in a bi-dimensional space containing rich visual information. The layout of e.g. this very paper provides valuable semantics to the reader: in which section are we right now? At the blink of an eye, this information is readily accessible via the salient section title (formatted differently and placed to highlight its role) preceding these words. Just to emphasize this point, *imagine having to scroll this content in plain text to access such information.*

In the last couple of years, the research community has shown a growing interest in addressing these limitations. Several approaches have been proposed to deal with visually-rich documents and integrate layout information into language models, with direct applications to Document Understanding tasks. Joint multi-modal pretraining (Xu et al., 2021; Powalski et al., 2021; Appalaraju et al., 2021) has been key to reach state-of-the-art performance on several benchmarks (Jaume et al., 2019; Graliński et al., 2020; Mathew et al., 2021). Nonetheless, a remaining limitation is that these (transformer-based) approaches are not suitable for processing long documents, the quadratic complexity of self-attention constraining their use to short sequences. Such models are hence unable to encode global context (e.g. long-range dependencies among text blocks).

Focusing on compressing the most relevant information from long texts to short summaries, the Text Summarization task naturally lends itself to benefit from such global context. Notice that, in practice, the limitations linked to sequence length are also amplified by the lack of visual/layout information in the existing datasets. Therefore, in this work, we aim at spurring further research on how to incorporate multimodal information to better capture long-range dependencies.

Our contributions can be summarized as follows:

---

[*] Work partially done while at reciTAL.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 636–651
May 2-6, 2023 ©2023 Association for Computational Linguistics

- We extend two popular datasets for long-range summarization, arXiv and PubMed (Cohan et al., 2018), by including visual and layout information – thus allowing direct comparison with previous works;

- We release 4 additional layout-aware summarization datasets (128K documents), covering French, Spanish, Portuguese, and Korean languages;

- We provide baselines including adapted architectures for multi-modal long-range summarization, and report results showing that (1) performance is far from being optimal; and (2) layout provides valuable information.

All the datasets are available on HuggingFace.[1]

2 Related Work

2.1 Layout/Visually-rich Datasets

Document Understanding covers problems that involve reading and interpreting visually-rich documents (in contrast to plain texts), requiring comprehending the conveyed multimodal information. Hence, several tasks with a central layout aspect have been proposed by the document understanding community. *Key Information Extraction* tasks consist in extracting the values of a given set of keys, e.g., the *total amount* in a receipt or the *date* in a form. In such tasks, documents have a layout structure that is crucial for their interpretation. Notable datasets include FUNSD (Jaume et al., 2019) for form understanding in scanned documents, and SROIE (Huang et al., 2019), as well as CORD (Park et al., 2019), for information extraction from receipts. Graliński et al. (2020) elicit progress on deeper and more complex Key Information Extraction by introducing the Kleister datasets, a collection of business documents with varying lengths, released as PDF files. However, the documents in Kleister often contain single-column layouts, which are simpler than the various multi-column layouts considered in LoRaLay. *Document VQA* is another popular document understanding task that requires processing multimodal information (e.g., text, layout, font style, images) conveyed by a document to be able to answer questions about a visually rich document (e.g., *What is the date given at the top left of the form?*, *Whose picture is given in this figure?*). The DocVQA dataset (Mathew et al., 2021) and InfographicsVQA (Mathew et al., 2022) are commonly-used VQA datasets that respectively provide industry documents and infographic images, encouraging research on understanding documents with complex interplay of text, layout and graphical elements. Finally, to foster research on visually-rich document understanding, Borchmann et al. (2021) introduce the Document Understanding Evaluation (DUE) benchmark, a unified benchmark for end-to-end document understanding, created by combining several datasets. DUE includes several available and transformed datasets for VQA, Key Information Extraction and Machine Reading Comprehension tasks.

2.2 Existing Summarization Datasets

Several large-scale summarization datasets have been proposed to boost research on text summarization systems. Hermann et al. (2015) proposed the CNN/DailyMail dataset, a collection of English articles extracted from the CNN and The Daily Mail portals. Each news article is associated with multi-sentence highlights which serve as reference summaries. Scialom et al. (2020) bridge the gap between English and non-English resources for text summarization by introducing MLSum, a large-scale multilingual summarization corpus providing news articles written in French, German, Spanish, Turkish and Russian. Going toward more challenging scenarios involving significantly longer documents, the arXiv and PubMed datasets (Cohan et al., 2018) consist of scientific articles collected from academic repositories, wherein the paper abstracts are used as summaries. To encourage a shift towards building more abstractive summarization models with global content understanding, Sharma et al. (2019) introduce BIGPATENT, a large-scale dataset made of U.S. patent filings. Here, invention descriptions serve as reference summaries.

The vast majority of summarization datasets only deal with plain text documents. As opposed to other Document Understanding tasks (e.g., form understanding, visual QA) in which the placement of text on the page and/or visual components are the main source of information needed to find the desired data (Borchmann et al., 2021), text plays a predominant role in document summarization. However, guidelines for summarizing texts – espe-

---

[1] `https://hf.co/datasets/nglaura/arxivlay-summarization`, `https://hf.co/datasets/nglaura/pubmedlay-summarization`, `https://hf.co/datasets/nglaura/hal-summarization`, `https://hf.co/datasets/nglaura/scielo-summarization`, `https://hf.co/datasets/nglaura/koreascience-summarization`

cially long ones – often recommend roughly previewing them to break them down into their major sections (Toprak and Almacioğlu, 2009; Luo et al., 2019). This suggests that NLP systems might leverage multimodal information in documents. Miculicich and Han (2022) propose a two-stage method which detects text segments and incorporates this information in an extractive summarization model. Cao and Wang (2022) collect a new dataset for long and structure-aware document summarization, consisting of 21k documents written in English and extracted from WikiProject Biography.

Although not all documents are explicitly organized into clearly defined sections, the great majority contains layout and visual clues (e.g., a physical organization into paragraphs, bigger headings/subheadings) which help structure their textual contents and facilitate reading. Thus, we argue that layout is crucial to summarize long documents. We propose a corpus of more than 345K long documents with layout information. Furthermore, to address the need for multilingual training data (Chi et al., 2020), we include not only English documents, but also French, Spanish, Portuguese and Korean ones.

3 Datasets Construction

Inspired by the way the arXiv and PubMed datasets were built (Cohan et al., 2018), we construct our corpus from research papers, with abstracts as ground-truth summaries. As the PDF format allows simultaneous access to textual, visual and layout information, we collect PDF files to construct our datasets, and provide their URLs.[2]

For each language, we select a repository that contains a high number of academic articles (in the order of hundreds of thousands) and provides easy access to abstracts. More precisely, we chose the following repositories:

- Archives Ouverte HAL (French),[3] an open archive of scholarly documents from all academic fields. As HAL is primarily directed towards French academics, a great proportion of articles are written in French;
- SciELO (Spanish and Portuguese),[4] an open access database of academic articles published in journal collections from Latin America, Iberian Peninsula and South Africa, and covering a broad range of topics (e.g. agricultural sciences, engineering, health sciences, letters and arts). Languages include English, Spanish, and Portuguese.
- KoreaScience (Korean),[5] an open archive of Korean scholarly publications in the fields of natural sciences, life sciences, engineering, and humanities and social sciences. Articles are written in English or Korean.

Further, we provide enhanced versions of the arXiv and PubMed datasets, respectively denoted as arXiv-Lay and PubMed-Lay, for which layout information is provided.

3.1 Collecting the Data

Extended Datasets The arXiv and PubMed datasets (Cohan et al., 2018) contain long scientific research papers extracted from the arXiv and PubMed repositories. We augment them by providing their PDFs, allowing access to layout and visual information. As the abstracts contained in the original datasets are all lowercased, we do not reuse them, but rather extract the raw abstracts using the corresponding APIs.

Note that we were unable to retrieve all the original documents. For the most part, we failed to retrieve the corresponding abstracts, as they did not necessarily match the ones contained in the PDF files (due to *e.g.* PDF-parsing errors). We also found that some PDF files were unavailable, while others were corrupted or scanned documents.[6] In total, about 39% (35%) of the original documents in arXiv (PubMed) were lost.

arXiv-Lay The original arXiv dataset (Cohan et al., 2018) was constructed by converting the LaTeX files to plain text. To be consistent with the other datasets – for which LaTeX files are not available – we instead use the PDF files to extract both text and layout elements. For each document contained in the original dataset, we fetch (when possible) the corresponding PDF file using Google Cloud Storage buckets. As opposed to the original procedure, we do not remove tables nor discard sections that follow the conclusion. We retrieve the corresponding abstracts from a metadata file provided by Kaggle.[7]

---

[2] We make the corpus-construction code publicly available at https://github.com/recitalAI/loralay-datasets.

[3] https://hal.archives-ouvertes.fr/

[4] https://www.scielo.org/

[5] http://www.koreascience.or.kr

[6] For more details on this, see Section A.1 in the Appendix.

[7] https://www.kaggle.com/Cornell-University/arxiv

PubMed-Lay For PubMed, we use the PMC OAI Service[8] to retrieve abstracts and PDF files.

HAL We use the HAL API[9] to download research papers written in French. To avoid excessively long (e.g. theses) or short (e.g. posters) documents, extraction is restricted to journal and conference papers.

SciELO Using Scrapy,[10] we crawl the following SciELO collections: Ecuador, Colombia, Paraguay, Uruguay, Bolivia, Peru, Portugal, Spain and Brazil. We download documents written either in Spanish or Portuguese, according to the metadata, obtaining two distinct datasets: SciELO-ES (Spanish) and SciELO-PT (Portuguese).

KoreaScience Similarly, we scrape the KoreaScience website to extract research papers. We limit search results to documents whose publishers' names contain the word *Korean*. This rule was designed after sampling documents in the repository, and is the simplest way to get a good proportion of papers written in Korean.[11] Further, search is restricted to papers published between 2012 and 2021, as recent publications are more likely to have digital-born, searchable PDFs. Finally, we download the PDF files of documents that contain an abstract written in Korean.

3.2 Data Pre-processing

For each corpus, we use the 95th percentile of the page distribution as an upper bound to filter out documents with too many pages, while the 5th (1st for HAL and SciELO) percentile of the summary length distribution is used as a minimum threshold to remove documents whose abstracts are too short. As our baselines do not consider visual information, we only extract text and layout from the PDF files. Layout is incorporated by providing the spatial position of each word in a document page image, represented by its bounding box (x_0, y_0, x_1, y_1), where (x_0, y_0) and (x_1, y_1) respectively denote the coordinates of the top-left and bottom-right corners. Using the PDF rendering library Poppler[12], text and word bounding boxes are extracted from each PDF, and the sequence order is recovered based on heuristics around the document layout (e.g., tables, columns). Abstracts are then removed by searching for exact matches; when no exact match is found, we use `fuzzysearch`[13] and `regex`[14] to find near matches.[15] For the non-English datasets, documents might contain several abstracts, written in different languages. To avoid information leakage, we retrieve the abstract of each document in every language available – according to the API for HAL or the websites for SciELO and KoreaScience – and remove them using the same strategy as for the main language. In the case an abstract cannot be found, we discard the document to prevent any unforeseen leakage. The dataset construction process is illustrated in Section A in the Appendix.

3.3 Datasets Statistics

The statistics of our proposed datasets, along with those computed on existing summarization datasets of long documents (Cohan et al., 2018; Sharma et al., 2019) are reported in Table 1. We see that document lengths are comparable or greater than for the arXiv, PubMed and BigPatent datasets.

For arXiv-Lay and PubMed-Lay, we retain the original train/validation/splits and try to reconstruct them as faithfully to the originals as possible. For the new datasets, we order documents based on their publication dates and provide splits following a chronological ordering. For HAL and KoreaScience, we retain 3% of the articles as validation data, 3% as test, and the remaining as training data. To match the number of validation/test documents in HAL and KoreaScience, we split the data into 90% for training, 5% for validation and 5% for test, for both SciELO datasets.

4 Experiments

4.1 Models

For reproducibility purposes, we make the models' implementation, along with the fine-tuning and evaluation scripts, publicly available.[16]

We do not explore the use of visual information in long document summarization, as the focus is on evaluating baseline performance using state-of-the-art summarization models augmented with layout information. While visual features might provide a better understanding of structures such as tables and figures, we do not expect substantial gains with

---

[8] `https://www.ncbi.nlm.nih.gov/pmc/tools/oai/`

[9] `https://api.archives-ouvertes.fr/docs/search`

[10] `https://scrapy.org/`

[11] For further details, see Section A.2 in the Appendix.

[12] `https://poppler.freedesktop.org/`

[13] `https://pypi.org/project/fuzzysearch/`

[14] `https://pypi.org/project/regex/`

[15] We use a maximum Levenshtein distance of 20 with fuzzysearch, and a maximum number of errors of 3 with regex.

[16] `https://github.com/recitalAI/loralay-modeling`

Dataset	# Docs	Mean Article Length	Mean Summary Length
arXiv (Cohan et al., 2018)	215,913	3,016	203
PubMed (Cohan et al., 2018)	133,215	4,938	220
BigPatent (Sharma et al., 2019)	1,341,362	3,572	117
arXiv-Lay	130,919	7,084	125
PubMed-Lay	86,668	4,038	144
HAL	46,148	4,543	134
SciELO-ES	23,170	4,977	172
SciELO-PT	21,563	6,853	162
KoreaScience	37,498	3,192	95

Table 1: Datasets statistics. Article and summary lengths are computed in words. For KoreaScience, words are obtained via white-space tokenization. Difference between arXiv and arXiv-Lay is due to the fact that we retain the whole document, while Cohan et al. (2018) truncate it after the conclusion.

respect to layout-aware models. Indeed, the information provided in figures (i.e., information that cannot be captured by layout or text) are commonly described in the caption or related paragraphs.

Text-only models with standard input size We use Pegasus (Zhang et al., 2020) as a text-only baseline for arXiv-Lay and PubMed-Lay. Pegasus is an encoder-decoder model pre-trained using gap-sentences generation, making it a state-of-the-art model for abstractive summarization. For the non-English datasets, we rely on a finetuned MBART as our baseline. MBART (Liu et al., 2020) is a multilingual sequence-to-sequence model pretrained on large-scale monolingual corpora in many languages using the BART objective (Lewis et al., 2019). We use its extension, MBART-50 (Tang et al., 2020),[17] which is created from the original MBART by extending its embeddings layers and pre-training it on a total of 50 languages. Both Pegasus and MBART are limited to a maximum sequence length of 1,024 tokens, which is well below the median length of each dataset.

Layout-aware models with standard input size We introduce layout-aware extensions of Pegasus and MBART, respectively denoted as Pegasus+Layout and MBART+Layout. Following LayoutLM (Xu et al., 2020), which is state-of-the-art on several document understanding tasks (Jaume et al., 2019; Huang et al., 2019; Harley et al., 2015), each token bounding box coordinates (x_0, y_0, x_1, y_1) is normalized into an integer in the range [0, 1000]. Spatial positions are encoded using four embedding tables, namely two for the coordinate axes (x and y), and the other two for the bounding box size (width and height). The layout representation of a token is formed by summing the resulting embedding representations The final representation of a token is then obtained through point-wise summation of its textual, 1D-positional and layout embeddings.

Long-range, text-only models To process longer sequences, we leverage BigBird (Zaheer et al., 2020), a sparse-attention based Transformer which reduces the quadratic dependency to a linear one. For arXiv-Lay and PubMed-Lay, we initialize BigBird from Pegasus (Zaheer et al., 2020) and for the non-English datasets, we use the weights of MBART. The resulting models are referred to as BigBird-Pegasus and BigBird-MBART. For both models, BigBird sparse attention is used only in the encoder. Both models can handle up to 4,096 inputs tokens, which is greater than the median length in PubMed-Lay, HAL and KoreaScience.

Long-range, layout-aware models We also include layout information in long-range text-only models. Similarly to layout-aware models with standard input size, we integrate layout information into our long-range models by encoding each token's spatial position in the page. The resulting models are denoted as BigBird-Pegasus+Layout and BigBird-MBART+Layout.

Additional State-of-the-Art Baselines We further consider additional state-of-the-art baselines for summarization: i) the text-only T5 (Raffel et al., 2019) with standard input size, ii) the long-range Longformer-Encoder-Decoder (LED) (Beltagy et al., 2020), and iii) the layout-aware, long-range LED+Layout, which we implement similarly to the previous layout-aware models.

4.2 Implementation Details

We initialize our Pegasus-based and MBART-based models with, respectively, the google/pegasus-large and facebook/mbart-large-50 checkpoints shared through the Hugging Face Model Hub. As for T5 and LED, we use the weights from t5-base and allenai/led-base-16384, respectively.[18]

Following Zhang et al. (2020) and Zaheer et al. (2020), we fine-tune our models up to 74k (100k) steps on arXiv-Lay (PubMed-Lay). On HAL, the total number of steps is set to 100k, while it is de-

---

[17] For the sake of clarity, we refer to MBART-50 as MBART.

[18] The large versions of T5 and LED did not fit into GPU due to their size.

Dataset	Instances			Input Length		Output Length	
	Train	Dev	Test	Median	90%-ile	Median	90%-ile
arXiv (Cohan et al., 2018)	203,037	6,436	6,440	6,151	14,405	171	352
PubMed (Cohan et al., 2018)	119,924	6,633	6,658	2,715	6,101	212	318
arXiv-Lay	122,189	4,374	4,356	6,225	12,541	150	249
PubMed-Lay	78,234	4,084	4,350	3,761	7,109	182	296
HAL	43,379	1,384	1,385	4,074	8,761	179	351
SciELO-ES	20,853	1,158	1,159	4,859	8,519	226	382
SciELO-PT	19,407	1,078	1,078	6,090	9,655	239	374
KoreaScience	35,248	1,125	1,125	2,916	5,094	219	340

Table 2: Datasets splits and statistics. Input and output lengths are computed in tokens, obtained using Pegasus and MBART-50's tokenizers for the English and non-English datasets, respectively.

creased to 50k for the other non-English datasets.[19] For each model, we select the checkpoint with the best validation loss. For Pegasus and MBART models, inputs are truncated at 1,024 tokens. For BigBird-Pegasus models, we follow Zaheer et al. (2020) and set the maximum input length at 3,072 tokens. As the median input length is much greater in almost every non-English dataset, we increase the maximum input length to 4,096 tokens for BigBird-MBART models. Output length is restricted to 256 tokens for all models, which is enough to fully capture at least 50% of the summaries in each dataset.

For evaluation, we use beam search and report a single run for each model and dataset. Following Zhang et al. (2020); Zaheer et al. (2020), we set the number of beams to 8 for Pegasus-based models, and 5 for BigBird-Pegasus-based models. For the non-English datasets, we set it to 5 for all models, for fair comparison. For all experiments, we use a length penalty of 0.8. For more implementation details, see Section B.1 in the Appendix.

5 Results and Discussion

5.1 General Results

In Table 3, we report the ROUGE-L scores obtained on arXiv and PubMed datasets (reported by Zaheer et al. (2020)), as well as on the corresponding layout-augmented counterparts we release. [20] On arXiv-Lay and PubMed-Lay, we observe that, while the addition of layout to Pegasus does not improve the ROUGE-L scores, there are gains in integrating layout information into BigBird-Pegasus. To assess whether these gains are significant, we perform significance analysis at the 0.05 level using bootstrap, and estimate a ROUGE-L threshold that predicts when improvements are significant. ROUGE-L improvements between each pair of models are reported in Table 11 in the appendix. On arXiv-Lay, we compute a threshold of 1.48 ROUGE-L, showing that BigBird-Pegasus+Layout significantly outperforms all Pegasus-based models. In particular, we find a 1.56 ROUGE-L improvement between BigBird-Pegasus and its layout-augmented counterpart, demonstrating that the addition of layout to long-range modeling significantly improves summarization. On PubMed-Lay, we compute a threshold of 1.77. Hence, the 0.96 ROUGE-L improvement from BigBird-Pegasus to its layout-augmented counterpart is not significant. However, the variance in font sizes in PubMed-Lay is much smaller compared to arXiv-Lay (see Table 12 in the appendix), reflecting an overall more simplistic layout. Therefore, we argue that layout integration has a lesser impact in PubMed-Lay, which can explain the non-significance of results. In addition, we find that BigBird-Pegasus significantly outperforms Pegasus and Pegasus+Layout only when augmented with layout, with an improvement of, respectively, 2.3 and 2.2 points. This demonstrates the importance of combining layout and long-range modeling.

While T5 and LED obtain competitive results, we find that the gain in adding layout to LED is minor. However, the models we consider have all been pre-trained only on plain text. As a result, the layout representations are learnt from scratch during fine-tuning. Similarly to us, Borchmann et al. (2021) show that their layout-augmented T5 does not necessarily improve the scores, and that performance is significantly enhanced only when the model has been pre-trained on layout-rich data.

Further, we observe, for both Pegasus and BigBird-Pegasus, a drop in performance w.r.t. the scores obtained on the original datasets. This can be explained by two factors. First, our extended

---

[19] We tested different values for the number of steps (10k, 25k, 50k, 100k) and chose the one that gave the best validation scores for MBART.

[20] For detailed results, please refer to Section C.1 in the Appendix.

Model	# Params	arXiv/ arXiv-Lay	PubMed/ PubMed-Lay
Pegasus (Zhang et al., 2020)	568M	38.83	41.34
BigBird-Pegasus (Zaheer et al., 2020)	576M	41.77	42.33
T5 (Raffel et al., 2019)	223M	37.90	39.23
LED (Beltagy et al., 2020)	161M	40.74	41.54
LED+Layout	165M	40.96	41.83
Pegasus	568M	39.07	39.75
Pegasus+Layout	572M	39.25	39.85
BigBird-Pegasus	576M	39.59	41.09
BigBird-Pegasus+Layout	581M	**41.15**	**42.05**

Table 3: ROUGE-L scores on arXiv-Lay and PubMed-Lay. Reported results obtained by Pegasus and BigBird-Pegasus on the original arXiv and PubMed are reported with a gray background. The best results obtained on arXiv-Lay and PubMed-Lay are denoted in bold.

Model	# Params	HAL (fr)	SciELO-ES (es)	SciELO-PT (pt)	KoreaScience (ko)
MBART	610M	42.00	36.55	36.42	16.94
MBART+Layout	615M	41.67	37.47	34.37	14.98
BigBird-MBART	617M	45.04	37.76	39.63	18.55
BigBird-MBART+Layout	621M	**45.20**	**40.71**	**40.51**	**19.95**

Table 4: ROUGE-L scores on the non-English datasets. The best results for each dataset are reported in bold.

Dataset	Train	Validation	Test
HAL (fr)	90.72	90.54	85.84
SciELO-ES (es)	84.86	84.28	84.90
SciELO-PT (pt)	90.95	90.58	91.96
KoreaScience (ko)	73.53	70.26	68.78

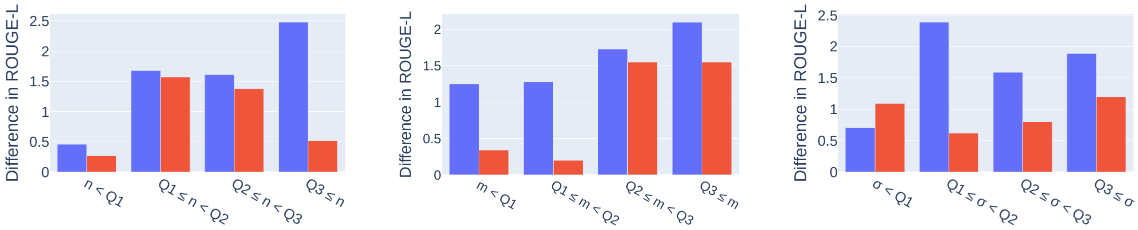
Table 5: Percent confidence obtained for the main language, for each dataset split.

datasets contain less training data due to the inability to process all original documents. Secondly, the settings are different: while the original arXiv and PubMed datasets contain clear discourse information (e.g., each section is delimited by markers) obtained from LaTeX files, documents in our extended versions are built by parsing raw PDF files. Therefore, the task is more challenging for text-only baselines, as they have no access to the discourse structure of documents, which further underlines the importance of taking the structural information, brought by visual cues, into account.

Table 4 presents the ROUGE-L scores reported on the non-English datasets. On HAL, we note that BigBird-MBART does not benefit from layout. After investigation, we hypothesize that this is due to the larger presence of single-column and simple layouts, which makes layout integration less needed. On both SciELO datasets, we notice that combining layout with long-range modeling brings substantial improvements over MBART. Further, we find that the plain-text BigBird models do not improve over the layout-aware Pegasus and MBART on arXiv-Lay and SciELO-ES, demonstrating that simply capturing more context does not always suffice. Regarding performance on KoreaScience, we can see a significant drop in performance for every model w.r.t the other non-English datasets. At first glance, we notice a high amount of English segments (e.g., tables, figure captions, scientific concepts) in documents in KoreaScience. To investigate this, we use the cld2 library[21] to detect the language in each non-English document. We consider the percent confidence of the top-1 matching language as an indicator of the presence of the main language (i.e., French, Spanish, Portuguese or Korean) in a document, and average the results to obtain a score for the whole dataset. Table 5 reports the average percent confidence obtained on each split, for each dataset. We find that the percentage of text written in the main language in KoreaScience (i.e., Korean) is smaller than in other datasets. As the MBART-based models expect only one language in a document (the information is encoded using a special token), we claim the strong presence of non-Korean segments in KoreaScience causes them to suffer from interference problems. Therefore, we highlight that KoreaScience is a more challenging dataset, and

---

[21] `https://github.com/GregBowyer/cld2-cffi`



(a) Article length (b) Summary length (c) σ of bounding box height

Figure 1: Benefit of using layout on arXiv-Lay (blue) and PubMed-Lay (red), defined as the difference in ROUGE-L scores between BigBird-Pegasus+Layout and BigBird-Pegasus. For each dataset, quartiles are calculated from the distributions of article lengths (a), summary lengths (b) and variance in the height of the bounding boxes (c). ROUGE-L scores are then computed per quartile range, and averaged over each range.

we hope our work will boost research on better long-range, multimodal *and* multilingual models.

Overall, results show a clear benefit of integrating layout information for long document summarization.

5.2 Human Evaluation

Metric	**BigBird**	**BigBird+Layout**
Precision %	35.15 (0.81)	37.51 (0.70)
Recall %	28.07 (0.73)	33.59 (0.86)
Coherence	3.80 (0.38)	3.75 (0.62)
Fluency	4.48 (0.03)	4.34 (0.16)
Overlap %	8.77 (0.24)	7.49 (0.36)
Flow %	30.75 (0.68)	33.02 (0.71)

Table 6: Average human judgement scores obtained by comparing gold-truth abstracts and summaries generated by BigBird and BigBird+Layout from 50 documents sampled from arXiv-Lay and HAL. Inter-rater agreement is computed using Krippendorff's alpha coefficient, and enclosed between parentheses.

To gain more insight into the effect of document layout for summarizing long textual content, we conduct a human evaluation of summaries generated by BigBird-Pegasus/BigBird-MBART and their layout-aware counterparts. We choose the BigBird-based models over the LED ones, as the gain in augmenting BigBird with layout is much more apparent. We evenly sample 50 documents from arXiv-Lay and HAL test sets, filtering documents by their topics (computer science) to match the judgment capabilities of the three human annotators. We design an evaluation interface (see Section C.2 in the appendix). For each sentence s_i in the generated summary, we ask the annotators to highlight the relevant tokens in s_i, along with the equivalent parts in the ground-truth abstract (denoted h_i). Further, we ask them to rate the summary in terms of coherence and fluency, on a scale of 0 to 5, following the DUC quality guidelines (Dang, 2005). Finally, annotators are asked to penalize summaries with hallucinated facts. The highlighting process allows us to compute precision and recall as the percentage of highlighted information in the generated summary and the ground-truth abstract, respectively. Moreover, we can compute an overlap ratio as the percentage of highlighted information that appears several times in the generated summary. Lastly, we calculate a flow percentage that evaluates how well the order of the ground-truth information is preserved by computing the percentage of times where the highlighted text h_i in the gold summary for one generated sentence s_i follows the highlighted text h_{i-1} for the previous sentence s_{i-1} (i.e. where any token from h_i occurs after a token in h_{i-1}). Table 6 reports the scores for each metric and model, averaged over all 50 documents, along with inter-rater agreements, computed using Krippendorff's alpha coefficient. We find that adding layout to the models significantly improves precision and recall, results in less overlap (repetition), and is more in line with the ground truth order. Further, annotators did not encounter any hallucinated fact in the 50 generated summaries. To conclude, reported results show that human annotators strongly agree that adding layout generates better summaries, further validating our claim that layout provides vital information for summarization tasks.

5.3 Case Studies

To have a better understanding of the previous results, we focus on uncovering the cases in which layout is most helpful. To this end, we identify features that relate to the necessity of having layout: 1) article length, as longer texts are intuitively easier to understand with layout, 2) summary length, as longer summaries are likely to cover more salient information, and 3) variance in font sizes (using the height of the bounding boxes), and, as such, the complexity of the layout. The benefit of using layout is measured as the difference in ROUGE-L scores between BigBird-Pegasus+Layout and its purely textual counterpart, on arXiv-Lay and PubMed-Lay. We compute quartiles from the distributions of article lengths, ground-truth summary lengths, and variance in the height of bounding boxes.[22] Based on the aforementioned factors, the scores obtained by each model are then grouped by quartile range, and averaged over each range, see Figure 1. On arXiv-Lay, we find that layout brings most improvement when dealing with the 25% longest documents and summaries, while, for both datasets, layout is least beneficial for the shortest documents and summaries. These results corroborate our claim that layout can bring important information about long-range context. Concerning the third factor, we see, on PubMed-Lay, that layout is most helpful for documents that have the widest ranges of font sizes, showcasing the advantage of using layout to capture salient information.

6 Conclusion

We have presented LoRaLay, a set of large-scale datasets for long-range and layout-aware text summarization. LoRaLay provides the research community with 4 novel multimodal corpora covering French, Spanish, Portuguese, and Korean languages, built from scientific articles. Furthermore, it includes additional layout and visual information for existing long-range summarization datasets (arXiv and PubMed). We provide adapted architectures merging layout-aware and long-range models, and show the importance of layout information in capturing long-range dependencies.

7 Limitations

The proposed corpus is limited to a single domain, that of scientific literature. Such limitation arguably extends to the layout diversity of documents. In terms of risks, we acknowledge the presence of Personally Identifiable Information such as author names and affiliations; nonetheless, such information is already voluntarily made public by the authors themselves.

8 Acknowledgements

We thank the reviewers for their insightful comments. This work is supported by the Association Nationale de la Recherche et de la Technologie (ANRT) under CIFRE grant N2020/0916. It was partially performed using HPC resources from GENCI-IDRIS (Grant 2021-AD011011841).

References

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. 2021. Docformer: End-to-end transformer for document understanding. *arXiv preprint arXiv:2106.11539*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Łukasz Borchmann, Michał Pietruszka, Tomasz Stanislawek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Graliński. 2021. Due: End-to-end document understanding benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Shuyang Cao and Lu Wang. 2022. Hibrids: Attention with hierarchical biases for structure-aware long document summarization. *arXiv preprint arXiv:2203.10741*.

Zewen Chi, Li Dong, Furu Wei, Wenhui Wang, Xian-Ling Mao, and Heyan Huang. 2020. Cross-lingual natural language generation via pre-training. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7570–7577.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*.

Hoa Trang Dang. 2005. Overview of duc 2005. In *Proceedings of the document understanding conference*, volume 2005, pages 1–12.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

---

[22] The quartiles are provided in Appendix C.3.

Filip Graliński, Tomasz Stanisławek, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek. 2020. Kleister: A novel task for information extraction involving long documents with complex layout. *arXiv preprint arXiv:2003.02356*.

Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. 2015. Evaluation of deep convolutional nets for document image classification and retrieval. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995. IEEE.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar. 2019. Icdar2019 competition on scanned receipt ocr and information extraction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1516–1520. IEEE.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6. IEEE.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019. Reading like her: Human reading inspired extractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3033–3043.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. 2022. Infographicvqa. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1697–1706.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2200–2209.

Lesly Miculicich and Benjamin Han. 2022. Document summarization with text segmentation.

Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. 2019. Cord: a consolidated receipt dataset for post-ocr parsing. In *Workshop on Document Intelligence at NeurIPS 2019*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Rafał Powalski, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka. 2021. Going full-tilt boogie on document understanding with text-image-layout transformer. *arXiv preprint arXiv:2102.09550*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. Mlsum: The multilingual summarization corpus. *arXiv preprint arXiv:2004.14900*.

Eva Sharma, Chen Li, and Lu Wang. 2019. Bigpatent: A large-scale dataset for abstractive and coherent summarization. *arXiv preprint arXiv:1906.03741*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.

Elif Toprak and Gamze Almacioğlu. 2009. Three reading phases and their applications in the teaching of english as a foreign language in reading classes with young learners. *Journal of language and Linguistic Studies*, 5(1).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, Online. Association for Computational Linguistics.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1192–1200.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

LoRaLay: A Multilingual and Multimodal Dataset for *Long Range* and *Layout*-Aware Summarization – Appendix

A Datasets Construction

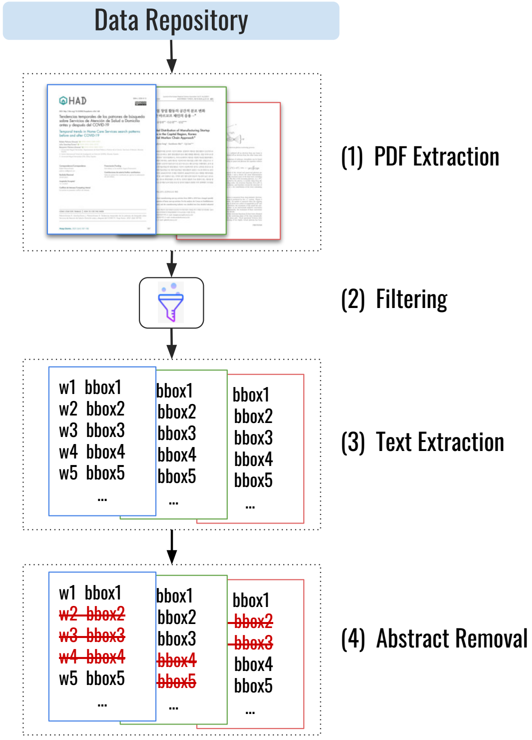


Figure 2: Dataset Construction Process.

A.1 Extended Datasets – Lost Documents

Figure 3 provides details on the amount of original documents lost in the process of augmenting arXiv and PubMed with layout/visual information. We observe four types of failures, and provide numbers for each type:

- The link to the document's PDF file is not provided (*Unavailable PDF*);
- The PDF file is corrupted (i.e., cannot be opened) (*Corrupted PDF*);
- The document is not digital-born, making it impossible to parse it with PDF parsing tools (*Scanned PDF*);
- The document's abstract cannot be found in the PDF (*Irretrievable Abstract*).

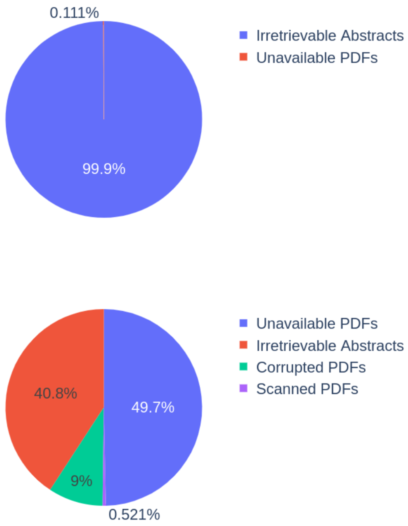


Figure 3: Distribution of failure types in arXiv-Lay (top) and PubMed-Lay (bottom).

A.2 KoreaScience – Extraction Rule

Korean documents in KoreaScience are extracted by restricting search results to documents containing the word "Korean" in the publisher's name. We show that this rule does not bias the sample towards a specific research area. We compute the distribution of topics covered by all publishers, and compare it to the distribution of topics covered by publishers whose name contains the word *Korean*. Figure 4 shows that the distribution obtained using our rule remains roughly the same as the original.

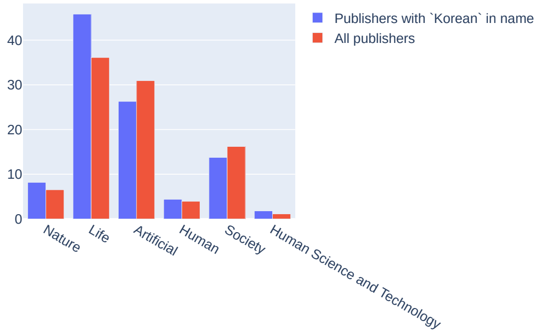


Figure 4: Distribution of topics covered by all publishers (red) vs distribution of topics covered by publishers whose name contains the word *Korean* (blue).

A.3 Samples

We provide samples of documents from each dataset in Figure 5.

A.4 Datasets Statistics

The distribution of research areas in arXiv-Lay and HAL are provided in Figures 6 and 7, respectively. Such distributions are not available for the other datasets, as we did not have access to topic information during extraction.

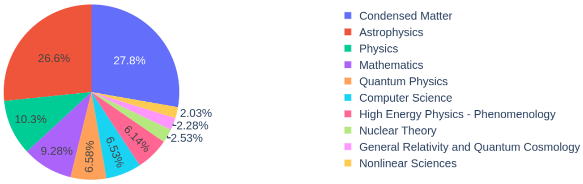


Figure 6: Distribution of research areas in arXiv-Lay.

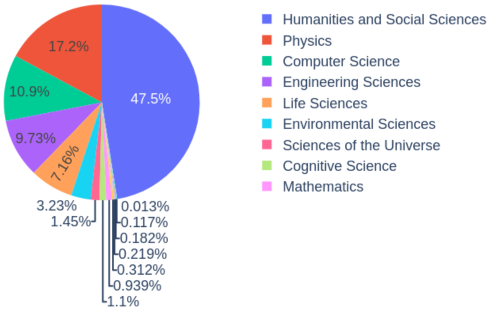


Figure 7: Distribution of research areas in HAL.

B Experiments

B.1 Implementation Details

Models were implemented in Python using PyTorch (Paszke et al., 2017) and Hugging Face (Wolf et al., 2019) librairies. In all experiments, we use Adafactor (Shazeer and Stern, 2018), a stochastic optimization method based on Adam (Kingma and Ba, 2014) that reduces memory usage while retaining the empirical benefits of adaptivity. We set a learning rate warmup over the first 10% steps – except on arXiv-Lay where it is set to 10k consistently with Zaheer et al. (2020), and use a square root decay of the learning rate. All our experiments have been run on four Nvidia V100 with 32GB each.

C Results

C.1 Detailed Results

Model	R-1	R-2	R-L
MBART	47.05	22.23	42.00
MBART+Layout	46.65	21.96	41.67
BigBird-MBART	49.85	**25.71**	45.04
BigBird-MBART+Layout	**49.99**	25.20	**45.20**

Table 8: ROUGE scores on HAL. Best results are reported in bold.

Model	R-1	R-2	R-L
MBART	17.33	7.70	16.94
MBART+Layout	15.43	6.69	14.98
BigBird-MBART	18.96	8.01	18.55
BigBird-MBART+Layout	**20.36**	**9.49**	**19.95**

Table 10: ROUGE scores on KoreaScience. The best results are reported in bold.

C.2 Human Evaluation

Using the Streamlit[23] framework, we design and develop an interface to aid human evaluation of summarization models.[24]

---

[23] `https://streamlit.io/`

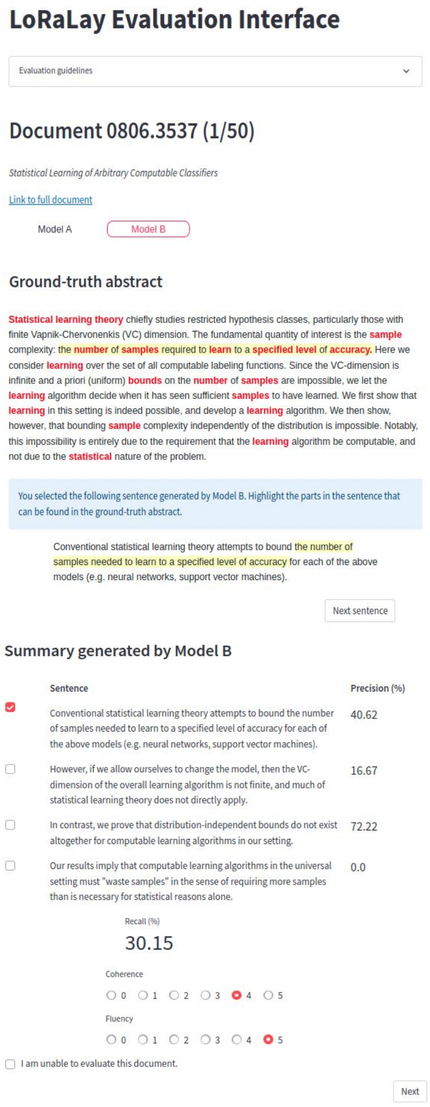
[24] The code is publicly available at `https://anonymous.4open.science/r/loralay-eval-interface-C20D`.

Model	arXiv / arXiv-Lay			PubMed / PubMed-Lay		
	R-1	R-2	R-L	R-1	R-2	R-L
Pegasus (Zhang et al., 2020)	44.21	16.95	38.83	45.97	20.15	41.34
BigBird-Pegasus (Zaheer et al., 2020)	46.63	19.02	41.77	46.32	20.65	42.33
T5 (Raffel et al., 2019)	42.79	15.98	37.90	42.88	17.58	39.23
LED (Beltagy et al., 2020)	45.41	18.14	40.74	45.28	19.86	41.54
LED+Layout	45.51	18.55	40.96	45.41	19.74	41.83
MBART	37.64	13.29	33.49	41.19	16.04	37.47
Pegasus	43.81	17.27	39.07	43.52	17.96	39.75
Pegasus+Layout	44.10	17.01	39.25	43.59	18.24	39.85
BigBird-Pegasus	44.43	17.74	39.59	44.80	19.32	41.09
BigBird-Pegasus+Layout	**46.02**	**18.95**	**41.15**	**45.69**	**20.38**	**42.05**

Table 7: ROUGE scores on arXiv-Lay and PubMed-Lay. Reported results obtained by Pegasus and BigBird-Pegasus on the original arXiv and PubMed are reported with a gray background. The best results obtained on arXiv-Lay and PubMed-Lay are denoted in bold.

Model	SciELO-ES			SciELO-PT		
	R-1	R-2	R-L	R-1	R-2	R-L
MBART	41.04	15.65	36.55	41.18	15.53	36.42
MBART+Layout	42.27	15.73	37.47	39.45	14.17	34.37
BigBird-MBART	42.64	16.60	37.76	44.85	18.70	39.63
BigBird-MBART+Layout	**45.64**	**19.33**	**40.71**	**45.47**	**20.40**	**40.51**

Table 9: ROUGE scores on the SciELO datasets. The best results are reported in bold.



LoRaLay Evaluation Interface

Document 0806.3537 (1/50)

Statistical Learning of Arbitrary Computable Classifiers

Ground-truth abstract

Statistical learning theory chiefly studies restricted hypothesis classes, particularly those with finite Vapnik-Chervonenkis (VC) dimension. The fundamental quantity of interest is the **sample** complexity: the **number** of **samples** required to **learn** to a **specified level** of **accuracy.** Here we consider **learning** over the set of all computable labeling functions. Since the VC-dimension is infinite and a priori (uniform) **bounds** on the **number** of **samples** are impossible, we let the **learning** algorithm decide when it has seen sufficient **samples** to have learned. We first show that **learning** in this setting is indeed possible, and develop a **learning** algorithm. We then show, however, that bounding **sample** complexity independently of the distribution is impossible. Notably, this impossibility is entirely due to the requirement that the **learning** algorithm be computable, and not due to the **statistical** nature of the problem.

You selected the following sentence generated by Model B. Highlight the parts in the sentence that can be found in the ground-truth abstract.

Conventional statistical learning theory attempts to bound the number of samples needed to learn to a specified level of accuracy for each of the above models (e.g. neural networks, support vector machines).

Summary generated by Model B

	Sentence	Precision (%)
☑	Conventional statistical learning theory attempts to bound the number of samples needed to learn to a specified level of accuracy for each of the above models (e.g. neural networks, support vector machines).	40.62
☐	However, if we allow ourselves to change the model, then the VC-dimension of the overall learning algorithm is not finite, and much of statistical learning theory does not directly apply.	16.67
☐	In contrast, we prove that distribution-independent bounds do not exist altogether for computable learning algorithms in our setting.	72.22
☐	Our results imply that computable learning algorithms in the universal setting must "waste samples" in the sense of requiring more samples than is necessary for statistical reasons alone.	0.0

Recall (%)
30.15

Coherence
○ 0 ○ 1 ○ 2 ○ 3 ● 4 ○ 5

Fluency
○ 0 ○ 1 ○ 2 ○ 3 ○ 4 ● 5

☐ I am unable to evaluate this document.

Figure 8: LoRaLay evaluation interface.

C.3 Analysis of the Impact of Layout

Table 12 lists the quartiles computed from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay.

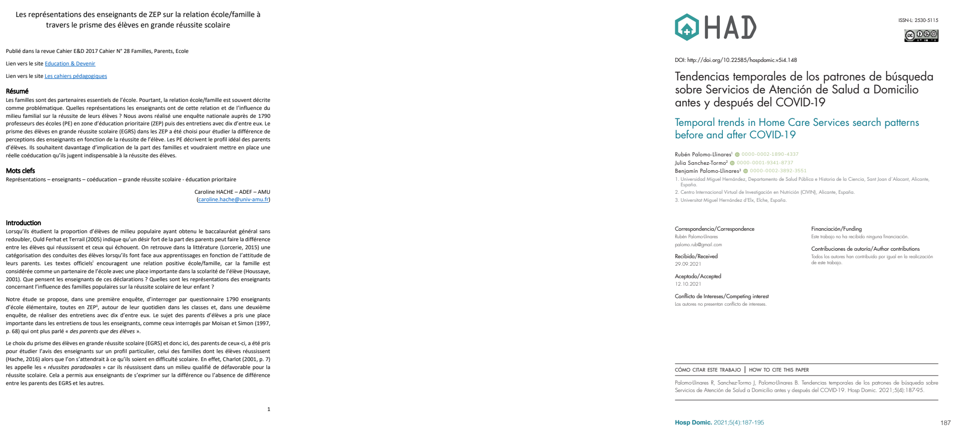
	LED	**LED+Layout**	**Pegasus**	**Pegasus+Layout**	**BigBird-Pegasus**	**BigBird-Pegasus+Layout**
T5	2.84 / 2.31	3.06 / 2.60	1.17 / 0.52	1.35 / 0.62	1.69 / 1.86	3.25 / 2.82
LED	–	0.22 / 0.29	1.67 / 1.79	1.49 / 1.69	1.15 / 0.45	0.41 / 0.51
LED+Layout	–	–	1.89 / 2.08	1.71 / 1.98	1.38 / 0.74	0.19 / 0.22
Pegasus	–	–	–	0.34 / 0.10	0.52 / 1.34	2.08 / 2.30
Pegasus+Layout	–	–	–	–	0.34 / 1.24	1.90 / 2.20
BigBird-Pegasus	–	–	–	–	–	1.56 / 0.96

Table 11: Absolute ROUGE-L score differences between each pair of models, on arXiv-Lay/PubMed-Lay.

Distribution	**Q1**		**Q2**		**Q3**	
	arXiv-Lay	PubMed-Lay	arXiv-Lay	PubMed-Lay	arXiv-Lay	PubMed-Lay
Article Length	6,226	3,513	9,142	5,557	13,190	8,036
Summary Length	119	130	159	182	202	247
σ of bounding box height	3.37	1.34	3.98	1.73	4.70	2.28

Table 12: Quartiles calculated from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay.

(a) arXiv-Lay (b) PubMed-Lay



(c) HAL (d) SciELO-ES

(e) SciELO-PT (f) KoreaScience

Figure 5: Samples from each dataset.

ViHOS: Hate Speech Spans Detection for Vietnamese

Phu Gia Hoang, Canh Duc Luu, Khanh Quoc Tran,
Kiet Van Nguyen, Ngan Luu-Thuy Nguyen
University of Information Technology, Ho Chi Minh City, Vietnam
Vietnam National University, Ho Chi Minh City, Vietnam
{19520215, 19521272}@gm.uit.edu.vn
{khanhtq, kietnv, ngannlt}@uit.edu.vn

Abstract

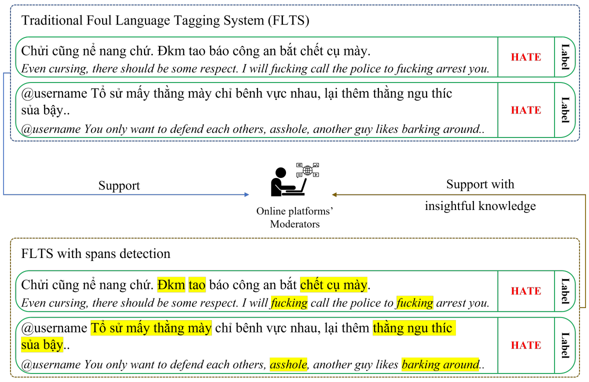
The rise in hateful and offensive language directed at other users is one of the adverse side effects of the increased use of social networking platforms. This could make it difficult for human moderators to review tagged comments filtered by classification systems. To help address this issue, we present the ViHOS (**Vi**etnamese **H**ate and **O**ffensive **S**pans) dataset, the first human-annotated corpus containing 26k spans on 11k comments. We also provide definitions of hateful and offensive spans in Vietnamese comments as well as detailed annotation guidelines. Besides, we conduct experiments with various state-of-the-art models. Specifically, XLM-R$_{Large}$ achieved the best F1-scores in Single span detection and All spans detection, while PhoBERT$_{Large}$ obtained the highest in Multiple spans detection. Finally, our error analysis demonstrates the difficulties in detecting specific types of spans in our data for future research. Our dataset is released on GitHub[1].

Disclaimer: This paper contains real comments that could be considered profane, offensive, or abusive.

1 Introduction

Social networking sites have been widely used all over the world. Here, users can easily share their thoughts, connect with others, or earn money by selling items, creating content, and so on. Since these sites are universally accepted, many extreme users misuse comment functions to abuse other individuals or parties with hate and offensive language. Consequently, it has been proved that these types of speech could harm other users' health (Mohan et al., 2017; Anjum et al., 2018). Sometimes these behaviors can be considered cyberbullying, cyber threats, or online harassment.

However, current studies are mainly about classifying comments as a whole with binary labels (Zampieri et al., 2019; Nguyen et al., 2021b) or multiple labeling schemes of abusive behaviors (Davidson et al., 2017; Founta et al., 2018; Mathur et al., 2018). These efforts are made to aid human moderators, who need to review a massive number of online tagged comments that violate their community standards. However, a system that can highlight the spans that make a comment hateful or offensive can be more advantageous to human moderators who frequently deal with long and tedious comments and prefer explanations over a system-generated unexplained tag per comment. Furthermore, in some cases, using highlighted spans and moderators' context knowledge, they can take some actions to stop cyberbullying or online harassment. Nevertheless, there is only a study on toxic spans, SemEval-2021 Task 5: Toxic Spans Detection (Pavlopoulos et al., 2021). On the other hand, in a study of Mathew et al. (2020), hate and offensive spans worked as a rationale to support models in classifying the whole comments.



Chửi cũng nể nang chứ. Đkm tao báo công an bắt chết cụ mày.
Even cursing, there should be some respect. I will fucking call the police to fucking arrest you.

@username Tổ sử mấy thằng mày chỉ bênh vực nhau, lại thêm thằng ngu thíc sủa bậy..
@username You only want to defend each others, asshole, another guy likes barking around..

Figure 1: An example of the aid of spans detection for traditional foul language tagging system can provide additional insightful knowledge about tagged comments for human moderators.

In Vietnamese, the resources about hate and offensive language are limited, namely ViHSD (Luu et al., 2021), HSD-VLSP (Vu et al., 2020), and UIT-ViCTSD (Nguyen et al., 2021b). Indeed, there

---

[1] https://github.com/phusroyal/ViHOS

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 652–669
May 2-6, 2023

is no study about hate and offensive spans in Vietnamese. This motivated us to (i) develop a new task of extracting hate and offensive spans from Vietnamese social media texts that conceivably impact research and downstream applications and (ii) provide the Natural Language Processing (NLP) research community with a new dataset for recognizing hate and offensive spans in Vietnamese social media texts.

Our two main contributions are summarized:

1. We created the first human-annotated dataset for Vietnamese Hate and Offensive Spans (ViHOS) comprising 26,467 human-annotated spans on 11,056 comments. Our dataset is annotated with a clear definition of hate and offensive spans, along with detailed and specialized guidelines for a less-studied language like Vietnamese. Compared to the toxic spans dataset at SemEval-2021 Task 5 (Pavlopoulos et al., 2021), which is built to detect toxic spans from toxic comments, or the HateXplain dataset (Mathew et al., 2020), which has spans working as a rationale for classifying the whole sentence, ours includes not only a large number of texts with annotated hate and offensive spans but also clean texts without any spans. This effort is made to serve a new task of detecting hate and offensive spans from Vietnamese online social media comments.

2. To evaluate the efficacy of our dataset, strong baselines are empirically investigated on ViHOS, including BiLSTM-CRF (Lample et al., 2016), XLM-R (Conneau et al., 2019), and PhoBERT (Nguyen and Nguyen, 2020). We conducted various experiment schemas, including comparing the full dataset having additional clean comments with the dataset that does not have; Single span detection, Multiple spans detection, and All spans detection. We obtain that: (i) Additional clean comments help the baselines have better performance than the dataset without them for 10±2% (ii) After fine-tuning the deep learning model and pre-trained language models, results show that the pre-trained language models outperform the deep learning models.

2 Related work

To the best of our knowledge, much of the research in the field of hate speech detection has been conducted in English due to the abundance of corpora and the robust pre-trained models. Many benchmark datasets for hate and offensive speech in other languages have also been published in recent years, including Arabic (Mubarak et al., 2020), Dutch (Tulkens et al., 2016), and French (Chiril et al., 2019). Novel models are introduced to improve the efficiency of hate and offensive speech detection. Initial approaches were based on typical machine learning and deep neural networks with word embeddings. Transformer-based models such as BERT (Devlin et al., 2018), BERTology (Rogers et al., 2020), and BERT-based transfer learning (Ruder et al., 2019) have recently been used to detect hate and offensiveness that achieved competitive results in major SemEval shared tasks such as SemEval-2020 Task 12 (Zampieri et al., 2020), and SemEval-2021 Task 5 (Pavlopoulos et al., 2021). However, research in Vietnamese is still limited in terms of the dataset and experimental methods. Only a few outstanding research exist, such as ViHSD (Luu et al., 2021), HSD-VLSP (Vu et al., 2020), and UIT-ViCTSD (Nguyen et al., 2021b).

For the topic of detecting foul spans, there are only a few case studies in English that are closely related, namely the SemEval-2021 Task 5: Toxic Spans Detection dataset (Pavlopoulos et al., 2021) and the HateXplain dataset (Mathew et al., 2020). The toxic spans, defined in the SemEval-2021 Task 5 dataset, are the sequences of words that make a text toxic. There are a total of 10,629 posts in this dataset, which stems from the Civil Comments dataset (Borkan et al., 2019). Another dataset with hate and offensive spans at the word level is HateXplain. The HateXplain contains 20,148 Gab and Twitter posts. Each post is manually classified into one of three labels: hateful, offensive, and normal.

In this study, we focus on Vietnamese to close the gap and **develop the first Vietnamese hate and offensive spans detecting benchmark**.

3 Dataset Creation

3.1 Dataset Source

ViHOS consists of 11,056 comments derived from the ViHSD dataset (Luu et al., 2021). The Vietnamese Hate Speech Detection dataset (ViHSD) is one of the few large and credible social media text datasets in a low-resource language like Vietnamese. ViHSD contains 27,624, 3,514, and 2,262 of CLEAN, HATE, and OFFENSIVE comments, respectively. Comments in ViHSD are public and collected from social media platforms.

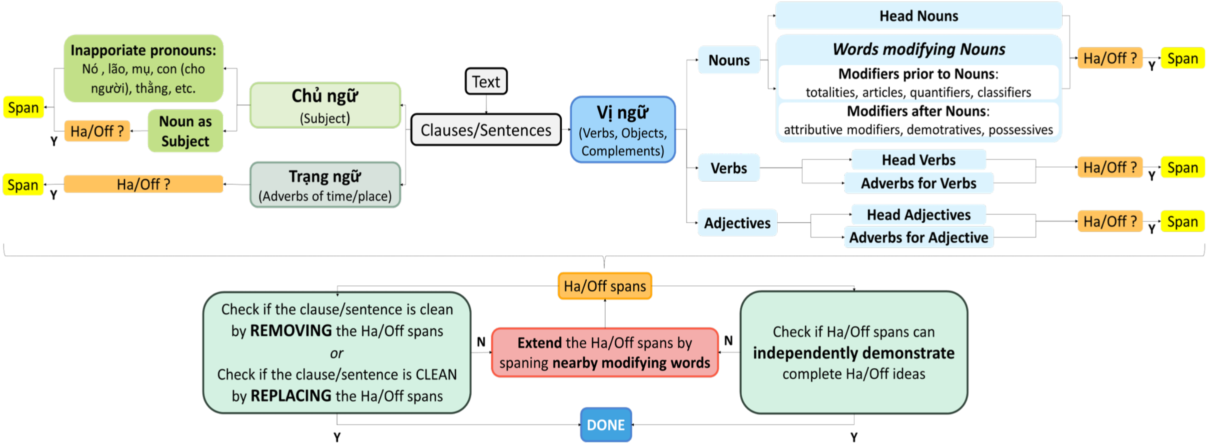


Figure 2: Detailed annotation guidelines for annotating comments for annotators. In which, *Ha/Off?* stands for a requirement for annotators to check whether the associated components are hate (Ha) or offensive (Off) or not.

Thus, metaphors, idioms, proverbs, and other tricky characteristics of online comments abound.

All of the HATE, OFFENSIVE comments from ViHSD after removing duplicates (5,528 comments left) are used to annotate the hate and offensive spans. Otherwise, 5,528 CLEAN comments, which also come from ViHSD and do not violate any hate or offensive definition defined in Section 3.2, are manually annotated for our dataset. We append the 5,528 CLEAN comments because: (*) We aim to detect the hate and offensive spans directly in online comments; (**) With an equal number of span and non-span (clean) comments helps models not be biased towards any type.

3.2 Annotation Guidelines

Our goal is to create a dataset that contains comprehensive hate and offensive thoughts, meanings, or opinions within the comments rather than just a lexicon of hate and offensive terms. We define the hate or offensive spans as follows to help annotators understand our goals:

- Harassing, cursing, insulting, disrespecting others.
- Sexual or verbal abuse towards one or a group of individuals based on their sensitive characteristics such as region, religion, politics, body, gender, etc.
- Insinuations, metaphors, metonymy used for hate, offensive or controversial purposes on sensitive issues such as region, gender, religion, politics, human rights, etc.
- Disuniting any factions or parties based on their politics, religion, ideologies, genders, etc.
- Causing verbal disrespect by using inappropriate pronouns.
- If replaced or removed, the sentence will no longer be hateful or offensive.

However, the hate or offensiveness in Vietnamese comments might cover one or even many components of a sentence. For instance: "thằng ad thở ra cái tư duy như trẻ lớp mầm" (**Eng:** *the admin speaks as his mind just like a kindergarten boy.)* This comment consists of three nouns/nouns phrases: "thằng ad"$_{\textit{the admin}}$ (it is offensive when calling a guy as a "thằng") as subject; "cái tư duy"$_{\textit{mind}}$ (the appearance of the word "cái" causes this noun phase become offensive) and "trẻ lớp mầm"$_{\textit{kindergarten boy}}$ as objects; one verb: "thở" (it usually means *breathe*, but in this context, we could consider it as *speak* but in a hate manner). As defined above, we must annotate a part of the subject, "thằng," and the whole phrase of the verb with its objects, "thở ra cái tư duy như trẻ lớp mầm" (**Eng:** *speaks as his mind just like a kindergarten boy*) in order to capture the whole hate or offensive ideas.

Therefore, we provided detailed guidelines (Figure 2) to assist annotators in determining when to annotate one or multiple components in a hateful and offensive sentence. As we observed, most of the comments in our dataset are colloquial. These comments are written freely and lack many grammatical rules. As a result, we frequently witness comments that lack subject(s), verb(s), conjunc-

	Phase 1				**Phase 2**	**Phase 3**
	1^{st}	**2^{nd}**	**3^{rd}**	**4^{th}**		
Kappa score	0.4161	0.4568	0.4936	0.6402	0.7239	0.7215
F1-score	0.7085	0.7186	0.7534	0.8219	0.8219	0.8585

Table 1: Inner-annotator agreement scores in three phases of annotation. In which, 1^{st}, 2^{nd}, 3^{rd}, 4^{th} are corresponding with four rounds of training annotators in Phase 1.

tions, punctuation marks, and so on. To deal with this, the comments were split into clauses/sentences (units). The smallest subdivided unit must have at least a cluster of "Chủ ngữ"-"Vị ngữ" ("Subject"-"Verb, objects, complements"), which is considered as a simple sentence or clause, or a cluster of "Chủ ngữ"-"Vị ngữ" nesting in another "Chủ ngữ"-"Vị ngữ" as a complex sentence or clause. From this, annotators can recognize components in clauses or sentences before annotating them.

Furthermore, in Appendix A and Appendix B, we also provided notices for annotators in Table 9 and ways to help them deal with nine online foul linguistic phenomena in Table 8 while annotating the data.

3.3 Dataset Construction Process

For dataset construction, we conducted three phases in which Phases 2 and 3 were inspired by Truong et al. (2021) and used metrics as bellow to calculate Inter-Annotator Agreement (IAA) among annotators. LightTag (Perry, 2021) is the tool we used for annotating data.

Assessment of Inter-Annotator Agreement Cohen's Kappa is widely used to measure inter-annotator agreement (IAA) in most tasks and is accepted as the standard measure (McHugh, 2012). However, numerous studies indicated that Kappa is not the most proper measure for the NLP sequence tagging task like NER (Hripcsak and Rothschild, 2005; Grouin et al., 2011). The reason is that the definite number of negative cases required to calculate the Kappa does not exist for named entity spans. Spans in our task are the sequences of characters rather than sequences of tokens since hate and offensive spans could be icon(s), word(s), or distinct character set(s) (see Table 8 for more details). Therefore, the pre-existing fixed number of characters to consider in the process of annotating is not existent.

A solution to deal with this is to calculate a character-level-based Kappa. Still, it has two associated problems: (1) annotators need to look at sequences of one or more characters instead of characters alone, causing the Kappa not to reflect the annotation task well; and (2) the "O"-labeled characters (the negative cases) outnumber the hate and offensive ones (the positive cases), provoking the Kappa to be computed on highly imbalanced data. For these reasons, the F1-score calculated without the negative cases is usually the measure for calculating IAA for the NLP tagging tasks like NER (Deleger et al., 2012). In this paper, IAA based on both F1-score (macro average) and character-level-based Kappa are calculated, while the former is the primary measure.

Phase 1: Pilot Annotation

Six undergraduate students were hired for our annotation tasks. The primary purpose of this pilot annotation phase was to familiarize our annotators with this task before entering the Main Annotation phase. We then developed an initial version of annotation guidelines with examples and distributed them to annotators. All annotators were required to carefully study the guidelines and give feedback before annotating the same 100 random samples from the 5,528 HATE, OFFENSIVE comments from ViHSD. This process was conducted four times with the F1-score and the Kappa for measuring IAA, which was calculated by averaging the results of pairwise comparisons across all annotators, shown in Table 1. All annotators were qualified as there was no F1-score of pairwise comparisons below 0.8.

Phase 2: Ground Truth Annotation

We randomly sampled a Ground Truth set of 600 comments from the 5,528 HATE, OFFENSIVE comments for this phase. Two guideline developers annotated the Ground Truth set separately using the well-developed guidelines from the former phase, resulting in an F1-score of 0.86 and Kappa (Cohen's Kappa) of 0.72. Afterward, we hosted a discussion to deal with annotation conflicts and update the annotation guidelines.

Phase 3: Main Annotation

We split the remaining HATE and OFFENSIVE

	Train	Dev	Test
Number of clean comments	4,552	569	575
Number of Ha/Off comments	4,422	553	553
Average clean comments length	8.69	8.50	9.04
Average Ha/Off comments length	16.81	17.68	16.13
Clean comments vocabulary size	4,234	1,423	1,400
Ha/Off comments vocabulary size	5,162	2,089	2,013
Number of multi-span comments (%)	2,322 (26.26)	308 (27.85)	296 (26.76)
Number of single-span comments (%)	1,970 (22.27)	229 (20.70)	235 (21.25)
Number of non-span comments (%)	4,552 (51.47)	569 (51.45)	575 (51.99)
Average number of spans	2.10	2.09	2.00

Table 2: ViHOS statistics. Vocabularies size and comments length are calculated at the syllable level.

comments (Luu et al., 2021) (4,928 comments left) into six non-overlapping and equal subsets. We also divided the 600-sample Ground Truth set from Phase 2 into six equal 100-sample smaller sets to insert into each subset. Each well-trained annotator from Phase 1 received a subset to annotate. Their annotation performance was assessed by calculating the F1 score and the Kappa score of the 100-sample Ground Truth sets in their subset. If any score is below 0.81 in terms of the F1 score, its corresponding annotator has to annotate again until it meets the requirement. This process was completed with an F1-score of 0.86 in the mean.

Furthermore, our annotators manually annotated CLEAN comments from ViHSD to spot any hate and offensive spans before being added to our dataset. This process collected 5,528 additional clean comments that met our requirements of having no hate and offensive spans.

3.4 Dataset Statistics

Before conducting dataset analysis and experiments, ViHOS has a total of 11,056 comments after the annotation process and is divided into three subsets: train, development, and test, with an 8:1:1 ratio. In detail, ViHOS has 5,360 comments with hate and offensive spans and 5,696 clean comments without in which 5,528 comments were additionally added and 168 comments have no hate and offensive spans after Phase 3 in the annotation process. Table 2 contains more information on the ViHOS statistics. It is apparent that the vocabulary of ViHOS is medium-sized, which is due to the small number of words in comments and comments in our dataset. In addition, more statistics about spans in ViHOS are shown in Table 3.

		Train	Dev	Test
Spans Quantity	0 span (%)	4,552 (51.47)	569 (51.45)	575 (51.99)
	1 span (%)	1,970 (22.27)	229 (20.71)	235 (21.25)
	2 - 3 spans (%)	1,527 (17.27)	207 (18.72)	202 (18.26)
	4 - 6 spans (%)	601 (6.80)	75 (6.78)	68 (6.15)
	7 - 10 spans (%)	164 (1.85)	18 (1.63)	21 (1.90)
	>10 spans (%)	30 (0.34)	8 (0.72)	5 (0.45)
Spans Length	1 syllable (%)	5,253 (52.03)	699 (52.48)	647 (52.77)
	2 - 3 syllables (%)	3,554 (35.20)	466 (34.98)	474 (38.66)
	4 - 6 syllables (%)	916 (9.07)	122 (9.16)	112 (9.14)
	7 - 10 syllables (%)	259 (2.57)	31 (2.33)	19 (1.55)
	>10 syllables (%)	114 (1.13)	14 (1.05)	14 (1.14)

Table 3: Spans quantity and length statistics.

4 Experiments and Results

4.1 Baseline Models

We treat the task of detecting hate and offensive spans as a task of sequence tagging. As a result, we make use of IOB format (Ramshaw and Marcus, 1995) to tag characters for model training, and testing. We conduct experiments on a set of solid baseline models, including BiLSTM-CRF and two pretrained language models, XLM-R and PhoBERT, to assess the difficulty of our dataset.

BiLSTM-CRF: We use BiLSTM-CRF (Lample et al., 2016), a model that achieves high performance in the span detection tasks (Pavlopoulos et al., 2021; Nguyen et al., 2021a). We implemented this model with three main layers: (1) The word embedding layer using pre-trained PhoW2V (Nguyen et al., 2020), (2) The BiLSTM layer, and (3) the Conditional Random Field (CRF).

XLM-R: XLM-RoBERTa (Conneau et al., 2019) is a multilingual language model and a variant of RoBERTa, pre-trained on 2.5T of data across 100 languages containing 137GB of Vietnamese texts. On several cross-lingual benchmarks, XLM-R outperforms mBERT.

	BiLSTM-CRF + Pho2W$_{syllable}$	**BiLSTM-CRF + Pho2W$_{word}$**	**XLM-R$_{Base}$**	**XLM-R$_{Large}$**	**PhoBERT$_{Base}$**	**PhoBERT$_{Large}$**
Full Data	0.7453	0.7036	0.7467	**0.7770**	0.7569	0.7716
W/o additional clean comments	0.6241	0.6244	0.6479	0.6756	0.6738	**0.6867**

Table 4: Experimental results on Full Data versus Without additional clean comments.

	Model	**Single span**			**Multiple spans**			**All spans**		
		P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Syllable	BiLSTM-CRF + Pho2W$_{syllable}$	0.4222	0.5009	0.4329	0.5134	0.5712	0.5068	0.7452	0.7769	0.7453
	XLM-R$_{Base}$	0.7604	0.7653	0.7203	0.7927	0.7574	0.7327	0.7766	0.7574	0.7467
	XLM-R$_{Large}$	**0.7577**	**0.7679**	**0.7214**	**0.7829**	**0.7569**	**0.7357**	**0.8071**	**0.7887**	**0.7770**
Word	BiLSTM-CRF + Pho2W$_{word}$	0.3196	0.4468	0.3594	0.3533	0.5001	0.4013	0.6823	0.7489	0.7036
	PhoBERT$_{Base}$	0.7392	0.7485	0.7016	0.7761	0.7329	0.7092	0.7870	0.7680	0.7569
	PhoBERT$_{Large}$	**0.7435**	**0.7567**	**0.7067**	**0.7878**	**0.7557**	**0.7321**	**0.8028**	**0.7835**	**0.7716**

Table 5: Experimental results on Single span, Multiple spans, and All spans subsets.

PhoBERT: PhoBERT (Nguyen and Nguyen, 2020) is a monolingual language model which is pre-trained on a 20GB Vietnamese dataset and has the same architecture and approach as RoBERTa. PhoBERT is proven as a state-of-the-art method in multiple Vietnamese-specific NLP tasks such as Part-Of-Speech Tagging, Dependency Parsing, and NER (Truong et al., 2021; Nguyen and Nguyen, 2020).

4.2 Experimental Settings

We empirically fine-tuned all pre-trained language models using *simpletransformers*[2]. For the tokenizer, each comment was tokenized using VnCoreNLP (Vu et al., 2018) in word-level and syllable-level for fine-tuning the PhoBERT and the XLM-R, respectively. In addition, we used Adam optimizer with a learning rate of 2e-5, a batch size of 8, and trained with 10 epochs.

We utilized a pre-trained word embedding - PhoW2V both syllable-level and word-level settings (Nguyen et al., 2020) with 100 dims to implement the BiLSTM-CRF model. The optimal hyper-parameters of BiLSTM-CRF are described in Table 6. All baseline models were trained on a system having 26GB RAM and an NVIDIA Tesla P100 GPU.

4.3 Evaluation Metrics

The macro-average F1-score (F1) is used to evaluate our models. For each pair of gold-predicted spans, we compute F1 and then calculate the arithmetic mean of F1 for each of these cases. It should be noted that the final F1-score, Accuracy, and Precision reported are an average of more than ten runs with various random seeds.

Hyper-parameters	**Values**
Optimizer	Adam
Learning rate	0.001
Mini-batch size	64
LSTM hidden state size	60
Embedding size	100
Dropout	[0.1, 0.1]
Epochs	10

Table 6: Hyper-parameters of the BiLSTM-CRF.

4.4 Experiments and Results

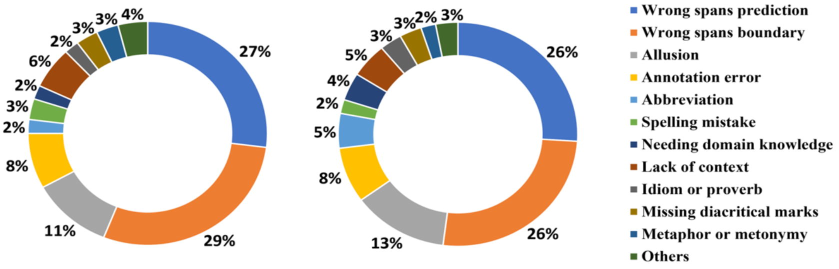
Table 4 reports the baseline results before and after adding the 5,528 additional clean comments. We discover that after the addition, the performances improve 0.1002±0.0210. Specifically, PhoBERT$_{Large}$ considerably outperforms other models in the dataset without additional clean data, achieving 0.6867 in F1-score. In addition, the best model trained on Full data is XLM-R$_{Large}$, which has an F1-score of 0.7770. We find that XLM-R$_{Large}$ increased by 0.1014 and PhoBERT$_{Large}$ increased by 0.0849. These results demonstrate that the appearance of the additional clean comments successfully reduces model bias and improves performance.

Table 5 reports our results in three subsets corresponding to Single, Multiple, and All spans. Both Single and Multiple spans subsets are made by the process of splitting the All spans, which also known as the test set, based on the number of spans in each comment. Their results are described as follows:

Single span: We experimented with both

---

[2] https://simpletransformers.ai/ (ver.0.63.3)



(a) Error Analysis on PhoBERT$_{\text{Large}}$

(b) Error Analysis on XLM-R$_{\text{Large}}$

Figure 3: Error analysis conducted on prediction on dev set made by PhoBERT$_{Large}$ and XLM-R$_{Large}$. We divide error cases into 12 categories including wrong spans detection, wrong spans boundary, allusion, annotation error, abbreviation, spelling mistake, needing domain knowledge, lack of context, idiom or proverb, missing diacritical marks, metaphor or metonymy, and others (rare characters, mixing other languages, all words stick together, etc.). These error cases are defined in Appendix C.

syllable-level and word-level language models. We discover that the pre-trained language models outperform the BiLSTM-CRF model by 0.3521±0.0099 in F1. This significant gap proves the fact that word embedding and features extraction of the pre-trained language models on the social media texts are superior to the BiLSTM-CRF. The XLM-R$_{Large}$ model achieves the best performance with a 0.7214 in F1-score. On the other hand, PhoBERT$_{Large}$ achieves a 0.7067 in F1-score. These results show no significant difference in performances among the multilingual and monolingual pre-trained models in the Single span.

Multiple spans: We experimented with the syllable-level and word-level and found that the pre-trained language models beat the BiLSTM-CRF model by 0.3212±0.0132. In addition, the performance of XLM-R$_{Large}$ is slightly better than the PhoBERT$_{Large}$ by 0.0036 in F1-score. The results on the Multiple spans are always better than the Single span, which might be explained by the fact that data in Multiple spans comprise more hate and offensive spans that can assist the models in learning more features of the data.

All spans: The results of the experiments on the All data are higher than the Single span and the Multiple spans. Specifically, in terms of F1-score, results of the XLM-R$_{Large}$ model are higher by 0.0556 and 0.0413 than the highest in the Single span and the Multiple spans, respectively while the figures for the PhoBERT$_{Large}$ are 0.0649 and 0.0395, respectively.

4.5 Results Analysis

We choose two best models: PhoBERT$_{Large}$ and XLM-R$_{Large}$ to conduct error analysis. As shown in Figure 3, we report the statistics of the ratio of various types of error cases[3] of 100 random samples in the dev set. We notice that *wrong spans prediction*[4], *wrong spans boundary*[5], *allusion*[6], *annotation error*[7], and *lack of context*[8] are major types of prediction failure of the PhoBERT$_{Large}$ and the XLM-R$_{Large}$.

We show some cases from the ViHOS development set in Table 7. In the first case of "Ns vậy lại xúc phạm cái đầu b**i *neutral face emoji*" (**Eng**: *If you said so, you insult the d**k head *neutral face emoji**), we notice that the PhoBERT$_{Large}$ could only predict "đầu"$_{head}$ as a hate and offensive span, whereas XLM-R$_{Large}$ predicts "b**i"$_{b**i}$. Both fail to predict the full boundary of "đầu b**i"$_{d**k\ head}$. The reason is that asterisks exist in the text ("b**i"). As defined in Subsection 4.2, the PhoBERT$_{Large}$, which was fine-tuned on the word-level data, cannot identify these tokens, but the XLM-R$_{Large}$, which was fine-tuned on the syllable-level data, can somewhat predict more accurate hate and offensive span even if it has asterisks.

Furthermore, in the second sampled comment, both best models failed to predict "đám cờ vàng" (**Eng**: *those yellow flag ones*) as a hate and offensive span, owing to the fact that this phase is an offensive metaphor for a rival party to Vietnamese

---

[3] Definition of errors are explained in the Appendix C.

[4] The model predicts clean spans as hateful and offensive.

[5] The model predicts inadequate boundary or fails to predict correctly.

[6] The comment refers to another person or subject indirectly and disrespectfully.

[7] The annotated span is wrong in terms of our guidelines. There is no later annotation modification in ViHOS.

[8] The comment has multiple meanings in different contexts, which mislead the prediction.

Ground truth spans	Models	
	PhoBERT$_{Large}$	XLM-R$_{Large}$
Ns vậy lại xúc phạm cái đầu b**i *neutral face emoji* (**Eng**: _If you said so, you insult the d**k head *neutral face emoji*_)	["xúc", "đầu"]	["b**i"]
@username nhân dân VN tức là cờ đỏ. Còn đám cờ vàng là lũ súc vật lưu vong. Hiểu hông? (**Eng**: _@username Vietnamese people are red flag. The yellow flag ones are animals and exiled. Understand?_)	["đỏ.", "đám", "lũ", "súc"]	["đám", "lũ", "súc", "vật", "vong."]
Cướp đêm là giặc, cướp ngày là quan. 24/7 lúc nào cũng phải nơm nớp. Than ôi cái đất nước hạnh phúc (**Eng**: _Night thieves are enemies, day thieves are bureaucrats. 24/7, we are in a perpetual state of fear and anxiety. What a happy country_)	["Cướp", "cướp", "nớp."]	["Cướp", "giặc,", "cướp"]
Đúng rồi đéo thể tin được. Đáng lẽ phải cào bằng ra mới đúng. Thật không thể tin nổi. Phải cào bằng ra mới được (**Eng**: _Yes, fucking unbelievable. They should dig until it comes out. Unbelievable. Should dig till it comes out_)	["đéo", "cào", "cào"]	["đéo"]

Table 7: Case studies in the dev set from ViHOS that are complicated for the PhoBERT$_{Large}$ and XLM-R$_{Large}$. The highlighted spans in first column are Ground truth spans associated with their comments. **Eng** refers to the English meaning of associated comments.

people. In the third instance, the phrase "cướp đêm là giặc, cướp ngày là quan," (**Eng**: *night thieves are enemies, day thieves are bureaucrats*) which is an idiom that originated from folk poetry, also misleads the prediction. In the final example, the verb "cào"$_{dig}$ has no object and must be comprehended in context. These intriguing and challenging linguistic phenomena encourage more research into more robust models and methods in this field.

5 Conclusion and Future Work

We presented ViHOS, a new Vietnamese dataset for evaluating hate and offensive spans detection models. ViHOS includes 26,467 human-annotated spans on 11,056 comments. In addition, state-of-the-art models are conducted as the first baseline models, including BiLSTM-CRF and pre-trained language models such as XLM-R$_{Base}$, XLM-R$_{Large}$, PhoBERT$_{Base}$, and PhoBERT$_{Large}$. As a result, the XLM-R$_{Large}$ model achieves the best performance, with an F1-score of 0.7770. Furthermore, we discover that the performance when detecting multiple spans is better than the performance in detecting single spans in Vietnamese hate and offensive spans detection. Our dataset is available publicly at the GitHub link[9].

Despite the study's many promising contributions, the proposed research work still has several potential concerns, especially since the performance is still modest, and incorrect predictions could harm users' reputations if they rely heavily on our method. We intend to expand the dataset size and diversity of hate and offensive context for Vietnamese in the future to address this shortcoming. Furthermore, pre-and post-processing techniques will be used to standardize social networking texts (Clark and Araki, 2011) and deal with complex cases (as discussed in Subsection 4.5) to improve model performance(Suman and Jain, 2021; Kotyushev et al., 2021; Chhablani et al., 2021), particularly for Vietnamese pre-trained language models.

---

[9] https://github.com/phusroyal/ViHOS

Limitations, Social Impacts, and Ethical Considerations

Limitations and Social Impacts

There are numerous incomprehensible comments in our dataset due to the lack of context. Consequently, our annotators had to place themselves in imaginary contexts in order to annotate those comments (see Table 9 for more details about our solution). This shortcoming combined with the limitations of the neural networks in terms of understanding various linguistic phenomena (see Table 8 for more details about nine different linguistic phenomena) caused their performances of this task still insufficient to become practical.

We also acknowledge the risk associated with publicizing a dataset of hate and offensive spans (e.g. utilizing ours as a source for building abusive chatbots). However, we firmly believe that our proposed benchmark creates more value than risks.

Ethical Considerations

The undergraduate students in the annotation process are Vietnamese native speakers; have at least 12 years of studying Vietnamese with average scores on the Vietnam National Exam on Literature of 6.5; have at least three years of using social network platforms. They were explicitly warned that their tasks will display hateful and offensive content and if they became overwhelmed, they were also urged to stop labeling. These undergraduate students were paid $0.1 per comment, which takes an average of 6.44 seconds to complete (excluding the time used by workers who took exceptionally lengthy comments).

All the comments in ViHOS originated in the study of Luu et al., 2021, which preserved users' anonymity by removing all of them when creating the ViHSD. As a result, the comments in our dataset do NOT reflect our thoughts or viewpoints. ViHOS is available to the public under a usage agreement for research and related purposes only.

Acknowledgments

This work has been funded by The VNUHCM-University of Information Technology's Scientific Research Support Fund.

References

Amna Anjum, Xu Ming, Ahmed Faisal Siddiqi, and Samma Faiz Rasool. 2018. An empirical study analyzing job productivity in toxic workplace environments. *International journal of environmental research and public health*, 15(5):1035.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pages 491–500.

Gunjan Chhablani, Abheesht Sharma, Harshit Pandey, Yash Bhartia, and Shan Suthaharan. 2021. Nlrg at semeval-2021 task 5: Toxic spans detection leveraging bert-based token classification and span prediction techniques. *arXiv preprint arXiv:2102.12254*.

Patricia Chiril, Farah Benamara, Véronique Moriceau, Marlène Coulomb-Gully, and Abhishek Kumar. 2019. Multilingual and multitarget hate speech detection in tweets. In *Conférence sur le Traitement Automatique des Langues Naturelles (TALN-PFIA 2019)*, pages 351–360. ATALA.

Eleanor Clark and Kenji Araki. 2011. Text normalization in social media: progress, problems and applications for a pre-processing system of casual english. *Procedia-Social and Behavioral Sciences*, 27:2–11.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Louise Deleger, Qi Li, Todd Lingren, Megan Kaiser, Katalin Molnar, et al. 2012. Building gold standard corpora for medical natural language processing tasks. In *AMIA Annual Symposium Proceedings*, volume 2012, page 144. American Medical Informatics Association.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

Cyril Grouin, Sophie Rosset, Pierre Zweigenbaum, Karën Fort, Olivier Galibert, and Ludovic Quintard. 2011. Proposal for an extension of traditional named entities: From guidelines to evaluation, an overview. In *Proceedings of the 5th linguistic annotation workshop*, pages 92–100.

Hoang Phe. 1983. *Vietnamese Dictionary*. Hong Duc, Vietnam.

George Hripcsak and Adam S Rothschild. 2005. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American medical informatics association*, 12(3):296–298.

Mikhail Kotyushev, Anna Glazkova, and Dmitry Morozov. 2021. Mipt-nsu-utmn at semeval-2021 task 5: Ensembling learning with pre-trained language models for toxic spans detection. *arXiv preprint arXiv:2104.04739*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *CoRR*, abs/1603.01360.

Son T Luu, Kiet Van Nguyen, and Ngan Luu-Thuy Nguyen. 2021. A large-scale dataset for hate speech detection on vietnamese social media texts. *arXiv preprint arXiv:2103.11528*.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection. *arXiv preprint arXiv:2012.10289*.

Puneet Mathur, Rajiv Shah, Ramit Sawhney, and Debanjan Mahata. 2018. Detecting offensive tweets in Hindi-English code-switched language. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 18–26, Melbourne, Australia. Association for Computational Linguistics.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.

Shruthi Mohan, Apala Guha, Michael Harris, Fred Popowich, Ashley Schuster, and Chris Priebe. 2017. The impact of toxic language on the health of reddit communities. In *Canadian Conference on Artificial Intelligence*, pages 51–56. Springer.

Hamdy Mubarak, Ammar Rashed, Kareem Darwish, Younes Samih, and Ahmed Abdelali. 2020. Arabic offensive language on twitter: Analysis and experiments. *arXiv preprint arXiv:2004.02192*.

Anh Tuan Nguyen, Mai Hoang Dao, and Dat Quoc Nguyen. 2020. A Pilot Study of Text-to-SQL Semantic Parsing for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4079–4085.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. Phobert: Pre-trained language models for vietnamese. *arXiv preprint arXiv:2003.00744*.

Kim Thi-Thanh Nguyen, Sieu Khai Huynh, Luong Luc Phan, Phuc Huynh Pham, Duc-Vu Nguyen, and Kiet Van Nguyen. 2021a. Span detection for aspect-based sentiment analysis in vietnamese. *arXiv preprint arXiv:2110.07833*.

Luan Thanh Nguyen, Kiet Van Nguyen, and Ngan Luu-Thuy Nguyen. 2021b. Constructive and toxic speech detection for open-domain social media comments in vietnamese. *arXiv preprint arXiv:2103.10069*.

Nguyen Hoai Nguyen. 2010. *Phonological characteristics of Nghe-Tinh dialect with the study of Vietnamese language history*. Vinh University, Vietnam.

John Pavlopoulos, Jeffrey Sorensen, Léo Laugier, and Ion Androutsopoulos. 2021. Semeval-2021 task 5: Toxic spans detection. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 59–69.

Tal Perry. 2021. LightTag: Text annotation platform. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 20–27, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18.

Thakur Ashutosh Suman and Abhinav Jain. 2021. Astartwice at semeval-2021 task 5: Toxic span detection using roberta-crf, domain specific pre-training and self-training. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 875–880.

Thinh Hung Truong, Mai Hoang Dao, and Dat Quoc Nguyen. 2021. COVID-19 named entity recognition for Vietnamese. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2146–2153, Online. Association for Computational Linguistics.

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, and Walter Daelemans. 2016. A dictionary-based approach to racism detection in dutch social media. *arXiv preprint arXiv:1608.08738*.

Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2018. Vncorenlp: A vietnamese natural language processing toolkit. *arXiv preprint arXiv:1801.01331*.

Xuan-Son Vu, Thanh Vu, Mai-Vu Tran, Thanh Le-Cong, and Huyen Nguyen. 2020. Hsd shared task in vlsp campaign 2019: Hate speech detection for social good. *arXiv preprint arXiv:2007.06493*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. Semeval-2020 task 12: Multilingual offensive language identification in social media (offenseval 2020). *arXiv preprint arXiv:2006.07235*.

A Abusive Language Characteristics

Table 8: Characteristics of abusive language in ViHOS with examples, explanations, and solutions for annotators.

Abusive language characteristics	**Examples**	**Explanations and Solutions**
Non-diacritical marks comments	(1) Dit me cai quy trinh,vao cap cuu deo co tien,deo bao hiem thi nam do di. (**Eng:** *Fuck the procedure, without money or insurance, you could just lay there and no one cares about your emergency* (2) Dung la con linh dien dien vua thoi chang qua nt goi de choc my dien thoi "con ng" nhu linh dien thi ai them (**Eng:** *That must be the Crazy Linh, so crazy! I just call to tease the Crazy My. No one gonna love Crazy/The person like Crazy Linh!*	**Explanation:** some of these non-diacritical marks comments might trick annotators a little bit. (1): this is a non-diacritical marks comment but still able to understand. (2): there are some problems causing annotators to re-read multiple times as no punctuation, diacritic, and the text "con ng" could be considered as "crazy girl" or "the type (of human)" and both of these meanings is inappropriate. **Solution:** Non-diacritical marks comments are annotated as others. Annotators have to re-read until they fully understand the texts if needed. Those examples are annotated as follows: (1): ["Dit me", "deo", "deo"] (**Eng:** ["*Fuck*", "*fuck*", "*fuck*"]) (2): ["con", "dien", "dien", "con ng", "dien", "ai them"] (**Eng:** ["*con*", "*crazy*", "*crazy*", "*crazy/the person like*", "*crazy*", "*No one gonna love*"] in which the word "con" is an inappropriate way to call a woman.)

Table 8 continued from the previous page.

Metaphors, metonymies	(1) cái miệng rộng quá đẻ con ra còn lọt (**Eng:** *The mouth is too big that a baby could even be born through it*) (2) Dm! qua đợt dịch này thì thằng sống ích kỷ này chắc sớm gia nhập juventuts (**Eng:** *Fuck it! After the pandemic, this selfish boy will soon join Juventuts*)	**Explanation:** in our dataset, many comments use metaphors or metonymy to convey their hate or offensiveness in another way. (1): this is a metaphor of a mouth as a vagina. (2): this is a metonymy of Juventus' jersey as prison shirts (a common metonymy in Vietnamese). **Solution:** Annotators are required to annotate the whole ideas of metaphors, metonymy. (1): ["cái miệng rộng quá đẻ con ra còn lọt"] (2): ["Dm", "thằng sống ích kỷ này chắc sớm gia nhập juventuts"] (**Eng:** ["*Fuck*", "*this selfish boy will soon join Juventuts*"])
Puns	(1) Ad đăng bài này cũng là Bồn Kỳ Lắc nè nè nè :) (**Eng:** *The admin, who posts this, is also a "Bồn Kỳ Lắc")*	**Explanation:** Some comments use phrases that only read them backwards, they make sense. As in the example, "Bồn Kỳ Lắc", if this phrase is read backwards, it is "Bắc Kỳ Lồn" (pussy north). **Solution:** Annotators are required to annotate these puns too. (1): ["Bồn Kỳ Lắc"]
Using non-words characters to form hieroglyphs	(1) có tin t lấy *knife symbol* xiên chết cụ m ko :))) (**Eng:** *Do you belive that I could get a *knife symbol* to fucking kill you?*) (2) Đâm vào () (**Eng:** *stab in the ()*	**Explanation:** (1): the *knife symbol* is used instead of the word knife. (2): the existence of "()" can be considered as pussy in this context. **Solution:** Annotating the non-words only if they can convey a full meaning of hate or offensiveness, and the whole phase if they can not. (1): ["t", "*knife symbol*", "xiên chết cụ m""] (**Eng:** ["*t*", "**knife symbol**", "*fucking kill you*"]) in which "t", "m" are inappropriate pronouns.) (2): ["Đâm vào ()"]

Table 8 continued from the previous page.

Spelling mistakes	(1) Tôi đeo hiểu bạn noi cai gì luôn a :))) (**Eng:** *I have no fucking idea about what you saying*) (2) Mà dx cái chữi ngta nghe hài vcl bù lại củng đở (**Eng:** *Those fucking curses is so funny that can even refill that*) (3)Khi bạn xai trính tã nhưng cá ghén vít đún trính tã :)) (**Eng:** *When you make spelling mistakes but trying to fix it :))*)	**Explanation:** (1): the words "đeo," "noi," "cai," "a" are spelling mistakes. The words can be understood as "đéo," "nói," "á". As that, these words are the same in accidentally missing acute marks. (2): the words: "chữi," "củng," "đở" are spelling mistakes is a phenomenon of mistaking tilde mark for hook above mark, and this often happens in some parts of Vietnam (Nguyen Hoai Nguyen, 2010). There are also familiar phenomena of mistaking marks such as tilde mark for underdot mark, acute mark for hook above mark, and so on (Nguyen Hoai Nguyen, 2010). (3): the phases: "xai trính tã," "cá ghén," "vít đúng trính tã" are spelling mistakes but on purpose. This comment utilizes spelling mistakes to attack opponents who also have spelling mistakes. Furthermore, these spelling mistakes also abuse opponents based on regional distinctions in accent, which cause some phenomenon of mistaking diacritical marks as in Example (2). **Solution:** The same as dealing with non-diacritical marks comments, annotators work as usual. (1): ["deo"] (**Eng:** ["*fuck*"]) (2): ["chữi", "vcl"] (**Eng:** ["*curse*", "*fuck*"]) (3): ["bạn xai trính tã nhưng cá ghén vít đún trính tã :))"]

Table 8 continued from the previous page.

Allusive language	(1) ra gì thì toi rồi, nó khác gì cách đảng CS chọn người, hồng hơn chuyên. Suy nghĩ kĩ đi. Giải độc cộng sản đã khó, giải độc tư tưởng cánh tả còn khó hơn. (**Eng:** *If it had something, it was done! It is just like the way CS (stands for Communism) chooses people, beauty over the profession. Detoxifying Communism is hard; detoxifying the left-wing political ideologies is even harder.*) (2) Rút kinh nghiệm lại được xài nghìn tỷ (**Eng:** *Just say learned and then can use trillion VND*)	**Solution:** Annotators are required to annotate the whole profound abuse. (1): ["toi", "hồng hơn chuyên", "Giải độc cộng sản đã khó, giải độc tư tưởng cánh tả còn khó hơn"] (**Eng:** ["*die*", "*beauty over the profession*", "*Detoxifying Communism is hard; detoxify the left-wing political ideologies even harder*"]) (2): ["Rút kinh nghiệm lại được xài nghìn tỷ"]
Homonym	(1) coin card ("coin card" is a homonym of "con cặc", which means *dick*)	**Solution:** annotate the whole abusive homonym. (1): ["coin card"]
Mixing languages	(1) Vl fake **Eng:** Fuck! Fake) (2) phe *X* compat tổng *Y* =)) (**Eng:** *X's side combats Y's side*) (3) Tối Thầy stream đá phò đi thầy ơi (**Eng:** *You should stream fucking some whores tonight, please!*)	**Solution:** treat foreign words as others and annotate if they meet the definition of hate and offensive spans. (1): ["Vl", "fake"] (**Eng:** ["*fuck*", "*fake*"]) (2): ["compat"] (**Eng:** ["*combat*"]) (3): ["đá phò"] (**Eng:** ["*fuck some whores*"])

Table 8 continued from the previous page.

Trick hate speech detecting systems on purpose	(1) Đấy Ông già xạo l*n của các bạn được tắm bồn Thái Lan đấy=))) (**Eng:** *See, your fucking old liar is Thai bathing again =))))*) (2) thằng c h.ó ngu (**Eng:** *Fucking stupid guy*) (3) vualonemlabaonhieu cm (**Eng:** *How long in cm to fit your pussy?*)	**Explanation**: some hate or offensive comments use punctuation to censor their inappropriate words (as in (1), asterisk is used in the word l*n (pussy)) or to disunite characters in words (as in (2), dot and space are used to disunite the words chó (dog) into c h.ó). These efforts actually can trick many hate speech detecting systems, or to put all words together (as in (3), vualonemlabaonhieu cm should be "vừa lồn em là bao nhiêu cm"). **Solution:** we annotate all characters which can be form into hate or offensive phase. (1): ["xạo l*n"] (**Eng:** ["*fucking lie*"]) (2): ["thằng c h.ó", "ngu"] (3): ["lon"] (**Eng:** ["*pussy*"])

B Notices for annotators

Table 9: Additional notices for annotators.

Notices	Example	Explanation
Try to figure out and consider as in the original context of the comments.		This could help annotators understand complex and non-context hate, offensive comments.
Do not let emotion affect the annotating process.		Annotators exposed in a long time to toxic comments are reported to get used to the frequently appearing hate, offensiveness.
Check the provided Vietnamese Dictionary if there is any uncertainty in being sure a word is hate or offensive.		We use the most reputable Vietnamese dictionary (Hoang Phe, 1983) to provide to annotators.
Should span the object is compared to in an inappropriate comparison.	"Ăn cơm nhìn như chó" (**Eng:** *You eat like a dog*) **Spans:** ["nhìn như chó"]	We consider this is an inappropriate comparison in which annotators must span "chó"$_{dog}$.

Table 9 continued from the previous page.

However, we should span the whole comparison if spanning only the object is compared to might not convey the complete hate or offensive idea	"ý thức như trẻ lớp mầm" **Spans:** ["ý thức như trẻ lớp mầm"] (**Eng:** *Your awareness is just like a kindergarten kid*)	If we only span "trẻ lớp mầm" (kindergarten kid), it will not convey the complete offensive idea. As that, annotators are encouraged to span the whole text instead.
Do not span conjunctions; exceptional Vietnamese cases; standard ways to call LGBTQ+.	(1) Vì thế, nên, nhưng, mà, etc. (**Eng:** *so, so, but, but, etc.*) (2) Gay, les, etc (3) Bóng, bê đê, ái nam ái nữ, etc.	(1) Some conjunctions in Vietnamese. (2) Appropriate ways to specify LGBTQ+. (3) Inappropriate ways to specify LGBTQ+ that need to be spanned in comments.
Blatant hate and offensive words prioritize being spanned over the others, especially in sentences without diacritics and could not be understanded.	May Cha H O an Tro. Cap. Ranh c. Di ngoi le. Nhieu chuyen. Do Cai Thu , do tam than **Spans:** ["Cap", "ngoi le", "Nhieu chuyen", "do tam than"] (**Eng:** ["*steal*", "*gossip*", "*talkative*", "*psycho*"])	Similar to this non-diacritical and incomprehensible comment, words as highlighted ones are more straightforward to be spanned.
Span the whole phase violating human rights.	"Về nước anh nên vào tù ở trước thay vì đi cách ly." (**Eng:** *When you return to Vietnam, you should go to jail first instead of going to isolation*) **Spans:** ["Về nước anh nên vào tù ở trước thay vì đi cách ly"]	Comments violate human rights, usually complex to specify hate, offensive words to span. As in this example, a Vietnamese citizen comes back from a foreign country has a right to have isolated healthcare firstly.
Span the whole obscene acronyms.	(1) clgv (**Eng:** *wtf is that?*) (2) clmn (**Eng:** *your mom's pussy*) (3) cmn (**Eng:** *your mother*).	
Follow the rule with words that do not have diacritics and conjoin to span out the hate, offensive spans.	cailongithe (**Eng:** *wtf is that?*)	Some comments have strings being constructed by many words missing diacritical marks, but still able to understand. The annotators should only span the hate and offensive characters set out of the string as in the example.
Span hate, offensiveness separately.	"Cả đám dlv là nhóm ngu dốt, trẻ trâu potay vcl luôn." (**Eng:** *The whole dlv crew are stupid, bull-headed kids (so I have) no fucking thing to say.*) **Spans:** ["đám", "ngu dốt", "trẻ trâu"] (**Eng:** ["*crew*", "*stupid*", "*bull-headed kids*"])	This comment must be spanned as in the example, but not as "đám," "ngu dốt, trẻ trâu," "vcl," "NGU."

C Definition of Error Cases for Error Analysis

We introduce 12 error definitions as follows:

1. **Wrong spans prediction**: The model predicts clean spans as hateful and offensive.
2. **Wrong spans boundary**: The model predicts inadequate boundary or fails to predict correctly.
3. **Allusion**: The comment refers to another person or subject in an indirect and disrespectful way.
4. **Annotation error**: Annotators have improperly annotated the span. The reason might be that they somehow do not follow the provided guidelines. However, there is no modification in the final dataset.
5. **Abbreviation**: The comment contain short forms of words.
6. **Spelling mistake**: The comment is spelling mistake.
7. **Needing domain knowledge**: Dialect and professional expertise are required to detect span in comments.
8. **Lack of context**: In different contexts, the comment could be understand in multiple meaning.
9. **Idiom or proverb**: The comment contains idiom or proverb.
10. **Missing diacritical marks**: Words in the comment do not have diacritical marks.
11. **Metaphor or metonymy**: The comment contains metaphor or metonymy.
12. **Others**: The comment contains rare characters, other languages, words in it are stick together, etc.

VOTE'N'RANK: Revision of Benchmarking with Social Choice Theory

Mark Rofin[1]*, Vladislav Mikhailov[1,5]*, Mikhail Florinskiy[1]*,
Andrey Kravchenko[2], Elena Tutubalina[1,3], Tatiana Shavrina[3,4,5],
Daniel Karabekyan[1], Ekaterina Artemova[6]†

[1]HSE University, [2]University of Oxford
[3]Artificial Intelligence Research Institute, [4]Institute of Linguistics, RAS, [5]SaluteDevices
[6]Center for Information and Language Processing, LMU Munich
Correspondence: Ekaterina.Artemova@lmu.de

Abstract

The development of state-of-the-art systems in different applied areas of machine learning (ML) is driven by benchmarks, which have shaped the paradigm of evaluating generalisation capabilities from multiple perspectives. Although the paradigm is shifting towards more fine-grained evaluation across diverse tasks, the delicate question of how to aggregate the performances has received particular interest in the community. In general, benchmarks follow the unspoken *utilitarian* principles, where the systems are ranked based on their mean average score over task-specific metrics. Such aggregation procedure has been viewed as a sub-optimal evaluation protocol, which may have created the illusion of progress. This paper proposes VOTE'N'RANK, a framework for ranking systems in multi-task benchmarks under the principles of the social choice theory. We demonstrate that our approach can be efficiently utilised to draw new insights on benchmarking in several ML sub-fields and identify the best-performing systems in research and development case studies. The VOTE'N'RANK's procedures are more robust than the mean average whilst being able to handle missing performance scores and specify conditions under which the system becomes the winner.

1 Introduction

Benchmarking has evolved as a conventional practice for accelerating the development of generalisable systems in different applied areas of machine learning (ML). Benchmarks are typically designed as a collection of datasets, corresponding task-specific evaluation metrics, and a criterion for summarising the overall performance on the tasks (Ruder, 2021). The benchmark holders provide public leaderboards, which are utilised by ML researchers and practitioners for comparing novel systems against one another, and, if applicable, human baselines, as well as selecting the best-performing ones for practical purposes. According to the benchmark sharing platform PAPERSWITHCODE[1], the community has put much effort into creating more than $10,000$ influential benchmarks in natural language processing (NLP), computer vision, and knowledge graphs, to name a few.

Criticism of the benchmark pillars. The benchmark methodological foundations have received wide criticism from academic and industrial communities (Bowman and Dahl, 2021). The criticism covers various aspects of benchmarking, raising concerns about the construct validity (Raji et al., 2021), fragility of the design and task choices (Dehghani et al., 2021), data leakage and annotation artifacts (Elangovan et al., 2021), SoTA-chasing tendencies at the cost of large carbon footprints (Bender et al., 2021), and low reproducibility of the reported results (Belz et al., 2021), inter alia. Recommendations proposed in these studies are of utmost importance to benchmark holders, system users, and developers. However, little attention has been paid to a more nuanced methodological question: *how to aggregate performance scores in multi-task benchmarks?*

Limits of canonical aggregation. The appropriateness of mean aggregation in multi-task ML problems is an ongoing debate in the community. The mean aggregation procedure implies that all task metrics are homogeneous (Colombo et al., 2022). Otherwise, it is recommended to evaluate the statistical significance of differences between models with non-parametric tests (Demšar, 2006; Benavoli et al., 2016). In practice, the NLP GLUE-style benchmarks (Wang et al., 2018, 2019a, 2021; Liang et al., 2020) use arithmetic average to rank models over heterogeneous metrics, which may lead to

---

*Equal contribution.

†Work done while at HSE University.

[1] **URL:** `paperswithcode.com/sota`. **Access date**: February 6, 2023.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 670–686
May 2-6, 2023

biased evaluation and subjective outcomes (Nießl et al., 2022; Waseem et al., 2021). The top-leading systems may dominate the others only on the outlier tasks (Agarwal et al., 2021), and their ranking is inconsistent with other Pythagorean means (Shavrina and Malykh, 2021). At the same time, the mean aggregation ignores the relative ordering and relies on the absolute score difference (Peyrard et al., 2017), equally treating tasks of different complexity (Mishra and Arunkumar, 2021) and from different domains (Webb, 2000).

Novel aggregation principles. Recent research has addressed these limitations, introducing novel aggregation methods and principles. One of the directions frames benchmarking in terms of microeconomics, highlighting the importance of the user utility (Ethayarajh and Jurafsky, 2020). The other studies urge evaluation of technical system properties in real-world scenarios (Zhou et al., 2021; Ma et al., 2021) and reliability of system rankings (Rodriguez et al., 2021). The benchmarking paradigm is also shifting towards adopting evaluation principles from other fields, such as non-parametric statistics and social choice theory (Choudhury and Deshpande, 2021; Min et al., 2021; Varshney et al., 2022; Colombo et al.).

Contributions. Drawing inspiration from the social choice theory, we make two application-oriented contributions and introduce an alternative tool for benchmark evaluation. First, this paper proposes VOTE'N'RANK, a flexible framework to rank systems in multi-task/multi-criteria benchmarks and aggregate the performances based on end-user preferences. VOTE'N'RANK includes 8 aggregation procedures that rely on rankings in each criterion and allow to aggregate homogeneous and heterogeneous information. The framework is easy-to-use and allows the users to plug in their own data. Second, we analyse the framework's application in four case studies: *(i)* re-ranking three NLP and multimodal benchmarks; *(ii)* exploring under which circumstances a system becomes *a Condorcet winner*; *(iii)* evaluating robustness to omitted task scores; and *(iv)* ranking systems in accordance with user preferences.

We publicly release the VOTE'N'RANK framework[2] to foster further development of reliable and interpretable benchmark evaluation practices for both academic and industrial communities.

2 VOTE'N'RANK

2.1 Background

The study of how individual preferences can be combined to reach a collective decision is the focus of *social choice theory* (Arrow, 2012). There are two main approaches to deal with preferences: *utilitarian* and *ordinal*. The first approach relies on the so-called cardinal utility, which implies that there exists some unique utility function for each individual that defines their preferences. Here, we can work with utilities as numerical values, and collective decision making aims to maximise the *social welfare utility*. Examples of such utilities are *utilitarian and egalitarian social welfare* measures, where the sum of utilities of individual agents and the utility of the worst agent get maximised, respectively.

The *utilitarian* approach has its drawbacks. First, it implies that kind of utility exists, which is not always true: individuals can compare two systems and prefer one to another but cannot say how many "utils" they got. Second, it assumes that individual utilities can be compared. The latter is a solid requirement for benchmarking problems, e.g. when we need to aggregate heterogeneous criteria such as performance and computational efficiency. In order to sum them up, one needs a transformation function that puts the metrics in the same measurement scheme. For example, DYNASCORE (Ma et al., 2021) utilises Marginal Rate of Substitution (MRS) from economics as such transformation function. Third, the utilitarian compensatory principle is questionable. Can low performance in one task/criterion be compensated by high performance in the others? (Munda, 2012)

The *ordinal* approach has a weaker requirement, where individuals have preferences (x is preferred to y, $x \succ y$, i.e. binary relations over objects), which should be aggregated in social preference (also called social rankings). This approach allows us to aggregate rankings from different tasks and criteria without worrying about measurement schemes.

2.2 Aggregation Procedures

Definitions. We adopt the conceptual definitions from the social choice theory to the objectives of selecting the best-performing system and ranking a set of systems as follows: *(i) a voter* or a *criterion* is a task in a given benchmark, and *(ii) an alternative* is a system candidate.

---

[2] `github.com/PragmaticsLab/vote_and_rank`

Objectives. Suppose we have a set M of systems $m \in \{m_1, \dots, m_{|M|}\}$ from the benchmark including a set T of voters $t \in \{t_1, \dots, t_{|T|}\}$ and the corresponding criteria $S = \{s_{mt}\}_{m=1,t=1}^{m=|M|,t=|T|}$, where s_{mt} is the score of system m in task t. Given that, we pursue two main objectives of the aggregation procedure σ, $\sigma : S \mapsto (M, \succ_\sigma)$: *(i)* to select the best performing alternative m^*, so that there is no alternative $\hat{m}, \hat{m} \succ_\sigma m^*$, and *(ii)* to rank the alternatives in the descending order according to σ values, so that $m_i \succ_\sigma m_j$. Here $\succ_\sigma$ denotes the preference resulting from the aggregation procedure σ.

Procedures. We propose 8 rules from 3 different classes: *scoring rules*, *iterative scoring rules*, and *majority-relation based rules*. We provide more details and examples in Appendix A.1.

2.2.1 Scoring rules

The total score of each system is calculated as the sum of corresponding scores in each task $Sc(m) = \sum_{i=1}^{|M|} c_i p_i(m)$, where $p_i(m)$ is the number of tasks having model m in the i^{th} place, and c_i is the i^{th} element of the scoring vector c. The systems with the highest scores constitute the final decision. We study the following rules that differ in their scoring vectors.

- *Plurality rule* applies $c = (1, 0, \dots, 0)$.
- *Borda rule* operates on $c = (|M| - 1, |M| - 2, \dots, 1, 0)$.
- *Dowdall rule* applies the scoring vector $c = (1, 1/2, \dots, 1/|M|)$.

The scoring vectors are designed to satisfy the voting rules' properties mentioned in Table 8 in Appendix A.2. The scoring vectors' design is based on the mathematical foundations of the social choice theory and is generally accepted in the community (Aizerman and Aleskerov, 1995).

Interpretation. The *Plurality* rule is one of the most widely used in everyday life. It only requires information about the best alternative for each voter. The *Borda* rule takes into account information about all alternatives. It assumes that differences in positions should be treated the same, whether it is between the first and the second alternatives or the second and the third ones. At the same time, the *Dowdall* rule is in some way in-between *Plurality* and *Borda*. It considers information about all alternatives but gives more weight to the difference in the preferences. A similar approach is used in the Eurovision song contest: they use $c = (12, 10, 8, 7, \dots, 1)$ making the difference in top positions more important to the outcome.

2.2.2 Iterative scoring rules

- The *Threshold rule* applies $c = (1, 1, .., 1, 1, 0)$. In case of ties scoring vectors $(1, 1, ..., 1, 0, 0)$, ..., $(1, 0, ..., 0, 0)$ are iteratively applied and used only to compare systems with the maximum sum of scores.
- The *Baldwin rule* iteratively applies scoring vectors $(|M|-1, |M|-2, ..., 1, 0)$, $(|M|-2, |M|-3, ..., 1, 0, 0)$,..., $(1, 0, ..., 0, 0)$, and at each iteration discards systems with the minimum sum of scores.

Both rules stop the procedure when it is impossible to break ties or there is only one alternative left.

Interpretation. The rules are similar in their iterative nature but different in terms of the intuition behind them. The *Threshold* rule is based on the idea that the worst position is what matters the most. When we start with $c = (1, 1, .., 1, 1, 0)$, we choose the alternatives declared worst in the least amount of cases. Since there can be ties, additional iterations are used to break them with $c = (1, 1, ..., 1, 0, 0)$ and so on; in other words, by looking at the least-k positions until we have one alternative left or can not break ties.

The *Baldwin* rule has two main differences from *Threshold*. First, it is based on the *Borda* score and considers information from all positions in the ranking, not only the worst one. Second, whilst the *Threshold* rule applies a new vector to the original profile and compares only tied alternatives, the *Baldwin* rule iteratively eliminates the least scored systems and moves the remaining up in rankings. For example, if system m_A is in the fifth place, but alternatives from the first four places are eliminated in the first rounds, m_A will be the first until it is eliminated or is among alternatives in the outcome.

2.2.3 Majority-relation based rules

Let us define a majority relation μ over the set of alternatives as the following binary relation: $m_A \mu m_B$ iff m_A is ranked higher than m_B by more criteria.

- *Condorcet rule*. m_C is the Condorcet winner (CW) iff $m_C \mu m$ for any $m \in M$.
- *Copeland rule*. Define the lower counter set of systems m_A as a set of systems dominated by m_A via μ: $L(m_A) = \{m \in M, m_A \mu m\}$. In a similar way, define the upper counter set of systems m_A as a set of systems that dominate m_A

via μ: $U(m_A) = \{m \in M, m\mu m_A\}$. Define $u(m) = |L(m)| - |U(m)|$. The final decision is provided by the alternatives with the highest $u(m)$.

- *Minimax rule*. Let $s(m_A, m_B)$ be the number of criteria for which system m_A is ranked higher than system m_B if $m_A \mu m_B$ or $s(m_A, m_B) = 0$ otherwise. The systems are ranked according to the formula $\text{rank}(m_A) = -\max_B s(m_B, m_A)$.

Interpretation. CW is the alternative that beats all the others in pairwise comparison. However, the *Condorcet* rule does not declare any winner if the CW does not exist. The *Copeland* and *Minimax* rules select the CW whenever it exists and solve the drawback as follows. The *Copeland rule* selects an alternative that dominates more alternatives and is dominated by less (the difference between the numbers is maximised). The *Minimax rule* chooses the alternative with the minimum number of defeats.

2.3 Properties of the Aggregation Procedures

There is a multitude of voting rules in the social choice theory (Nurmi, 1983; Levin and Nalebuff, 1995; De Almeida et al., 2019; Aleskerov et al., 2010). The motivation behind our rules[3] is that they generally overcome the mean aggregation limitations and vary in their *properties*, allowing the user to be more flexible in choosing the rule for their purposes. The outcomes can be interpreted in terms of the properties followed or violated by the rules. We discuss our rules' properties in Appendix A.2.

2.4 Framework

Figure 1 describes three supported settings of performing the aggregation objectives. The toy benchmark has three evaluated alternatives and consists of seven voters grouped by the task, e.g. natural language inference, text classification, and question answering (QA).

A Basic aggregation: the aggregation procedure is applied to the leaderboard as is.

B Weighted aggregation: each voter in the group is assigned a group weight equal to $1/|T_{group}|$. The blue group weights are $1/3$, and the orange and the violet group weights are $1/2$. Each group contributes equally to the final ranking, regardless of the number of voters.

C Two-step aggregation: each voter group is treated as a standalone leaderboard. We independently apply a procedure to each voter group and compute an interim ranking shown as "elector". Next, we aggregate the groupwise rankings by applying the same procedure one more time and compute the final ranking.

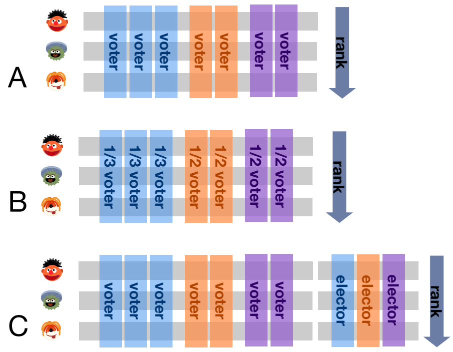


Figure 1: Three ways to run the aggregation procedures. A: Basic aggregation. B: Weighted aggregation. C: Two-step aggregation.

3 Case Studies

This section describes four case studies on three NLP and multimodal benchmarks. Our main objective here is to re-interpret the benchmarking trends under the social choice theory. We provide a brief description of the benchmarks below.

- GLUE (General Language Understanding Evaluation; Wang et al., 2018) combines nine datasets on QA, sentiment analysis, and textual entailment. GLUE also includes a linguistic diagnostic test set. $|M|$=30.
- SGLUE (Wang et al., 2019a) is the GLUE follow-up consisting of two diagnostic and eight more complex NLU tasks, ranging from causal reasoning to multi-hop and cloze-style QA. $|M|$=22.
- VALUE (Video-and-Language Understanding Evaluation; Li et al.) covers 11 video-and-language datasets on text-to-video retrieval, video QA, and video captioning. $|M|$=7.

The leaderboards present the results of evaluating various neural models, such as BERT (Devlin et al., 2019), STRUCTBERT (Wang et al., 2019b), ALBERT (Lan et al., 2019), ROBERTA (Liu et al., 2019), T5 (Raffel et al., 2020), DEBERTA (He et al., 2020), ERNIE (Zhang et al., 2019), and their ensembles and other model configurations.

---

[3] We do not consider more complex rules like *Kemeny* since it is NP-hard to find the Kemeny winner (Bartholdi et al., 1989), and it is often implemented as the *Borda* rule approximation (Colombo et al.).

Benchmark	k	σ^{gm}	σ^{og}	**Copeland**	**Minimax**	**Plurality**	**Dowdall**	**Borda**
GLUE	top-1	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	top-3	1.0	0.67	1.0	0.67	0.67	0.67	1.0
	top-5	1.0	0.80	0.60	0.80	0.80	0.80	0.8
	top-7	1.0	0.86	0.86	0.86	0.86	0.86	1.0
	least-5	0.67	0.00	1.0	0.33	0.33	1.0	1.0
	least-7	0.86	0.71	1.0	0.14	0.14	1.0	1.0
	τ	0.56	-0.08	0.23	-0.05	0.03	0.28	0.41
SGLUE	top-1	1.0	1.0	0.00	1.0	0.00	0.00	1.0
	top-3	1.0	1.0	0.67	0.67	0.67	0.67	1.0
	top-5	1.0	1.0	1.0	1.0	0.80	1.0	1.0
	top-7	1.0	0.86	0.86	0.71	0.57	0.86	0.86
	least-5	1.0	1.0	1.0	0.33	0.33	1.0	1.0
	least-7	0.86	0.86	0.86	0.14	0.14	0.86	1.0
	τ	0.45	0.36	0.08	-0.5	-0.15	0.12	0.24

Table 1: Agreement rates between the top/least-k rankings with σ^{am}. The Kendall Tau correlation (τ) is computed on the total rankings.

Method	GLUE		SGLUE	
	DP	IIA	DP	IIA
σ^{am}	1	0.0 $\pm$0.0	0	0.0 $\pm$0.0
σ^{gm}	0	0.0 $\pm$0.0	0	0.0 $\pm$0.0
σ^{og}	3	0.0 $\pm$0.0	1	0.0 $\pm$0.0
Copeland	6	2.76 $\pm$1.3	2	0.90 $\pm$0.8
Minimax	21	2.94 $\pm$1.5	17	1.14 $\pm$1.0
Plurality	25	5.26 $\pm$1.6	17	1.98 $\pm$1.4
Dowdall	0	9.10 $\pm$2.4	0	4.24 $\pm$2.1
Borda	0	7.96 $\pm$3.8	1	5.38 $\pm$1.8

Table 2: Discriminative power (DP) and independence of irrelevant alternatives (IIA) values. The lower, the better for both DP and IIA.

3.1 Re-interpreting Benchmarks

Method. We begin with a case study on re-ranking systems on the publicly available leaderboards using the scoring and majority-relation based rules[4]: *Plurality*, *Dowdall*, *Borda*, *Copeland*, and *Minimax* as the baselines. σ^{og} is an aggregation metric that identifies the amount by which the system fails to get a minimum score of $\gamma = 0.95$ (lower is better). The comparison is run by computing *(i)* the agreement rate (AR; in %), i.e. the proportion of the top/least-k systems between the given procedure and σ^{am}, *(ii)* the Kendall Tau correlation (τ) between the total rankings, *(iii)* the discriminative power (DP) or the number of tied alternatives. i.e. alternatives with the same score (Brandt and Seedig, 2016), and *(iv)* the independence of irrelevant alternatives (IIA), i.e. how often the *new* systems change the ranking (see Appendix A.2 for details). IIA is computed iteratively in two steps. First, we initialise a leaderboard with two random systems m_A and m_B. Second, we add a new random system m_C to the leaderboard and check if the rankings of m_A and m_B have changed. We repeat the procedure by adding up to $|M|$ systems and counting how often the new system affects the ranking. The experiment is run 50 times to account for randomness.

Results. Table 1 and Table 2 present the results except for the VALUE benchmark which is discussed in Appendix B. We find that methods tend to agree on the top systems, but *Minimax* and *Plurality* disagree on which ones are the worst. Despite high ARs on particular top/least-k systems, the order of the systems on GLUE and SGLUE is different, which is indicated by the low correlation coefficients. The Pythagorean mean results are consistent with one another on the top-7 systems and may lead to different worst systems. σ^{og} generally disagrees with σ^{am} for the top and worst systems on GLUE but has higher ARs and correlation on SGLUE.

At the same time, the DP results demonstrate that *Dowdall* and *Borda* produce only one pair of alternatives with the same score, whilst *Minimax* and *Plurality* treat a significantly larger number of systems as equivalent. The reason is that the rules initially intend to define the best alternative, and they are indecisive between the alternatives when utilised to rank. The IIA experiment shows that introducing a new system influences the *Dowdall* and *Borda* rankings. However, this tendency is less common for *Copeland*, *Minimax*, and *Plurality* and is observed only up to 2 times on SGLUE.

Overall, we observe that the GLUE and SGLUE benchmark rankings depend on the aggregation procedure. The human baseline (HUMAN) rank has risen by up to 13 positions on GLUE (see Table 3). The *Copeland* method takes HUMAN, DEBERTA+CLEVER, and T5 equal, meaning that the difference between the number of candidates they dominate and are dominated by is the same. The *Minimax* ranking suggests that HUMAN, T5, and the ALBERT+DAAF+NAS ensemble are equivalent, meaning that minimal maximum defeats against other models are the same. In their turn, the *Plurality* and *Dowdall* procedures rank HUMAN as the second-best solution, since HUMAN receives the best performance in several tasks, such as RTE (Wang et al., 2018) and MNLI (Williams et al., 2018). The tendency is also observed on the SGLUE benchmark (see Table 9 in Appendix B), with the exception that HUMAN is selected as the winner by the *Copeland*, *Plurality*, and *Dowdall*

---

[4] We omit the iterative rules here for the sake of space.

Rank	σ^{am}	σ^{gm}	σ^{og}	**Copeland**	**Minimax**	**Plurality**	**Dowdall**	**Borda**
1	😃 91.18	😃 90.89 ↕0	😃 0.074 ↕0	😃 29.00 ↕0	😃 0 ↕0	😃 2.00 ↕0	😃 4.95 ↕0	😃 260.50 ↕0
2	🗑️ 91.07	🗑️ 90.78 ↕0	🧚 0.075 ↑4	😐 25.00 ↑1	😐 −5.50 ↑1	🙋🏽‍♀️ 2.00 ↑13	🙋🏽‍♀️ 4.08 ↑13	🗑️ 256.00 ↕0
3	😐 90.88	😐 90.56 ↕0	🗑️ 0.076 ↓1	🗑️ 24.00 ↓1	🧚 −6.00 ↑1	😐 1.50 ↕0	😐 3.82 ↕0	😐 247.50 ↕0
4	🧚 90.86	🧚 90.48 ↕0	🦁 0.076 ↕0	🐛 22.00 ↑3	🗑️ −6.50 ↓2	🟦 1.00 ↑1	🧚 3.41 ↕0	🧚 241.50 ↕0
5	🟦 90.74	🟦 90.44 ↕0	🟦 0.077 ↕0	🙋🏽‍♀️ 22.00 ↑10	🐛 −7.00 ↑2	🗑️ 1.00 ↓3	🗑️ 3.27 ↓3	🦁 233.50 ↑1
6	🦁 90.66	🦁 90.34 ↕0	🐛 0.078 ↑1	🧚 22.00 ↓2	🙋🏽‍♀️ −7.00 ↑9	🦁 0.50 ↕0	🟦 2.57 ↓1	🐛 229.50 ↑1
7	🐛 90.48	🐛 90.11 ↕0	🔵 0.082 ↑3	🦁 16.00 ↓1	🦁 −7.00 ↓1	🧚 0.00 ↓3	🐛 2.55 ↕0	🟦 220.50 ↓2

Table 3: Results of re-ranking the GLUE benchmark. Changes in the system ranks are depicted with arrows, whilst the superscripts denote scores assigned by the aggregation procedure. Notations: 🙋🏽‍♀️=HUMAN; 😃=ERNIE; 🗑️=STRUCTBERT+CLEVER; 😐=DEBERTA+CLEVER; 🧚=DEBERTA/TURINGNLRV4; 🟦=MACALBERT+DKM; 🐛=T5; 🦁=ALBERT+DAAF+NAS; 🔵=FUNNEL. The superscript values stand for the voting rules' scores, whilst the subscript values indicate changes in the ranking positions. $\uparrow x$ means up x positions, $\downarrow x$ means down x positions, $\updownarrow$ means no changes.

procedures and is equal to the ERNIE system according to *Minimax*. The results for *Borda* are similar to σ^{am} and σ^{gm} on the top-4 and top-6 ranks on GLUE and SGLUE, respectively.

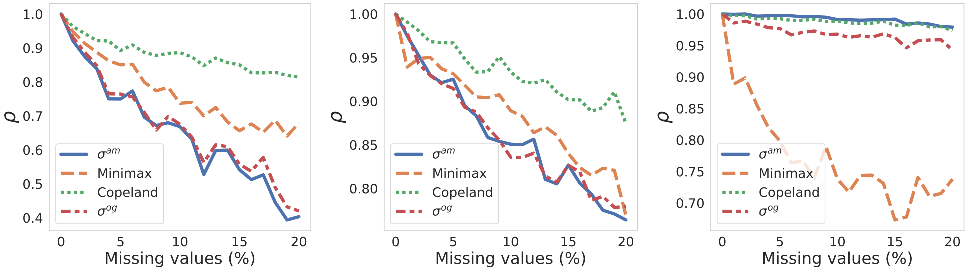
Selecting the winner. Another application of the voting rules includes selecting the winner from the set of alternatives. Here, we also utilise the *Threshold*, *Baldwin*, and *Condorcet* rules. Note that we run the VALUE experiment over missing and non-missing scores since the HUMAN results are presented for only 6 out of 11 tasks.

Results. Table 4 presents the results of selecting the winner for each benchmark. ✗ denotes that *(i)* the given method does not support missing values, or *(ii)* there is no *Condorcet* winner (CW). We observe that different SoTAs are selected by 2/7/3 (on GLUE/SGLUE/VALUE) procedures as opposed to σ^{am}, σ^{gm}, and σ^{og}. The *Threshold* rule selects T5+UDG and STRUCTBERT+CLEVER as winners because their performance is the worst the least amount of times. The *Baldwin* rule agrees with the *Plurality* and *Minimax* results. When considering VALUE missing scores, we find that HUMAN is declared SoTA by the *Copeland*, *Minimax*, and *Condorcet* procedures. It means that HUMAN beats any other model in pairwise comparison and is declared the CW, whilst significantly outperforming the systems on specific tasks.

Method	GLUE	SGLUE	VALUE
σ^{am}	😃	😃	✗/👺
σ^{gm}	😃	😃	✗/👺
σ^{og}	😃	😃	✗/👺
Copeland	😃	🙋🏽‍♀️	🙋🏽‍♀️/👺
Minimax	😃	😃🙋🏽‍♀️	🙋🏽‍♀️/👺
Plurality	😃🙋🏽‍♀️	🙋🏽‍♀️	✗/👺
Dowdall	😃	🙋🏽‍♀️	✗/👺
Borda	😃	😃	✗/👺
Threshold	🗑️	🔵	✗/👺
Baldwin	😃	😃🙋🏽‍♀️	✗/👺
Condorcet	😃	✗	🙋🏽‍♀️/👺

Table 4: The winner selection results. Notations: 🗑️=STRUCTBERT+CLEVER; 🙋🏽‍♀️=HUMAN; 👺=CRAIG.STARR; 😃=ERNIE; 🔵=T5+UDG.

Case study discussion. Benchmarks can suffer from *saturation*, which is characterised by surpassing estimates of the human performance followed by stagnation in SoTA improvements (Ott et al., 2022). The NLP community has discussed saturation of the GLUE benchmark over time (Kiela et al., 2021; Ruder, 2021) and minor performance gains of the upcoming top-leading systems on SGLUE (Rogers, 2019). However, the discussion relies on the mean aggregation. Let us take a step away from the *utilitarian* approach. We observe that HUMAN may still take leading positions, and system ranking varies on these benchmarks under the social choice theory principles. VALUE demonstrates more stable results in terms of the AR and the system order, which we attribute to its novelty and minor performance differences between the systems. Overall, our rules provide interpretable results and cope with the missing leaderboard values in contrast to the *utilitarian* methods.



(a) GLUE (b) SGLUE (c) VALUE

Figure 2: Spearman correlation (ρ) between top-7 model rankings with/without omitted leaderboard values for σ^{am}, σ^{og}, *Minimax*, and *Copeland* rankings. The results are averaged over 100 runs.

3.2 The Condorcet Winner

One of the most natural ways to choose the best system given a set of weights defined by the user is the Condorcet method, which declares a system the winner if it dominates all other alternatives in pairwise comparison (Black et al., 1958). The Condorcet method is hard to destabilise (Edelman, 2015) and easy to interpret in practice, indicating that the CW best matches the preferences. Given the weights vector, finding the CW, if it exists, is trivial. We can also find the weights that make a given alternative the CW or determine that no weights with that property exist.

Method. Let us define an operator $R(m_1, m_2, i)$:

$$R(m_1, m_2, i) = \begin{cases} 1, & \text{if } \exists s_{m_1 i} \wedge \exists s_{m_2 i} \wedge s_{m_1 i} > s_{m_2 i}, \\ -1, & \text{if } \exists s_{m_1 i} \wedge \exists s_{m_2 i} \wedge s_{m_1 i} < s_{m_2 i}, \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

A system m is declared a CW if the following property is satisfied:

$$\forall m' \in M \setminus \{m\} \quad \sum_{k=1}^{|T|} R(m, m', k) w_k \geq 0 \quad (2)$$

Let $G_m \in \{-1, 0, 1\}^{|M|-1 \times |T|}$ be a matrix such that $G_{ij} = R(m, m'_i, j)$, where $\{m'_1, .., m'_{|M|-1}\} = M \setminus \{m\}$.

Equation 2 can be re-written as: $G_m \cdot w \succcurlyeq 0$, which results in defining a space W^* in $\mathbb{R}^{|T|}$, whose each point is a weight vector making m a CW. Any linear algorithm can be applied to find a point in W^* or determine that W^* is empty. Furthermore, any other linear conditions can be added, such as upper/lower bounds of the w components and a linear function that needs to be optimised, e.g. $w_i \longrightarrow \min$. Let us call a system for which there exists a vector of weights making it a CW *prospective*. By definition, the system is *prospective* if W^* is not empty.

Example. Let us illustrate the method on the SGLUE benchmark (see Table 4). There is no CW if the task weights are assigned uniformly. Nevertheless, T5 may become the CW when the BOOLQ accuracy (Clark et al., 2019) and MULTIRC exact match scores (Khashabi et al., 2018) have equal weights of 0.5, and the other criteria weights are zeroed. In this scenario, T5 has been found to be a *prospective* system on SGLUE, whiste ROBERTA is declared *non-prospective*.

Results. There are 9/88, 10/12, and 3/3 prospective/non-prospective systems on GLUE, SGLUE, and VALUE, respectively. The results indicate that it is possible to find specific evaluation scenarios in which a given system is the best. In contrast, the non-prospective system always has an alternative that performs neither worse nor better.

Case study discussion. The CW criterion presents another perspective of selecting the best systems. Notably, the existence of the CW weights assumes that practitioners can simulate a set of real-world scenarios where the system is the best across the given axes. Specifying if the system can be the CW on the leaderboard would help diagnosing the systems without additional heavy experiments. The developers also can document this information on model sharing platforms, e.g. HUGGINGFACE (Wolf et al., 2020).

3.3 Robustness to Missing Scores

This case study considers a more detailed analysis of the majority-relation based voting rules that can be efficiently utilised for ranking systems and selecting the winner over missing scores. Here, we evaluate the robustness of the rules to omitted performance scores and analyse how the rankings change under such perturbation.

Method. *Copeland* and *Minimax* take as input the majority graph in which each vertex corresponds to a candidate and an edge from the candidate m_1 to m_2 exists iff $m_1 \mu m_2$, i.e. m_1 is ranked higher than m_2 by more criteria. Let us say there are T criteria $t \in \{t_1, \dots, t_T\}$ and $w \in \{w_1, \dots, w_n\}$ are the weights assigned to them.

$$m_1 \mu m_2 \iff \sum_{i=1}^{|T|} w_i R(m_1, m_2, i) > 0 \quad (3)$$

When evaluating $R(m_1, m_2, i)$, this approach can handle missing values, ignoring the pairs where either of the scores is missing. We can apply the majority-relation based rules using relation μ to rank alternatives with missing scores without losing any information whilst accounting for the available criteria.

We analyse the robustness of the *Copeland* and *Minimax* rules as follows. First, we compute the rankings using both methods on each benchmark without omitting scores and use them as references. Next, we randomly replace N scores with empty values and find top-7 systems over the corrupted leaderboards. We calculate the Spearman correlation (ρ) between the final rankings and the references. Note that we use the median values when omitting scores for σ^{am} and σ^{og} as the baselines.

Results. Figure 2 shows that σ^{am} and σ^{og} display lower stability and *Copeland* performs the best on GLUE and SGLUE. However, we observe that *Minimax* is the least stable on VALUE, whilst *Copeland*, σ^{am}, and σ^{og} perform on par.

Case study discussion. We attribute the low stability of *Minimax* on VALUE to its limitations. Recall that there are minor differences between the systems on VALUE, which cause *Minimax* to score them very similar (see Table 10 in Appendix B). In this case any missing value can influence the rankings, which results in the low ρ coefficients.

3.4 Ranking Based on User Preferences

This case study aims at system ranking based on the user utility. We rank systems in a simulated scenario that considers preferences on *performance*, *computational efficiency*, and *fairness*.

Method. We use the HUGGINGFACE library to fine-tune and evaluate systems on GLUE. Each system is initialised with a fixed set of five random seeds and fine-tuned for five epochs with default hyper-parameters and a batch size of 16. The development set performance is averaged across all runs. We consider the following systems: BERT-base, RoBERTa-base, ALBERT-base, DeBERTa-base, DistilBERT-base (Sanh et al., 2019), DistilRoBERTa-base (Sanh et al., 2019), and GPT2-medium (Radford et al., 2019). The experiments are run on a single GPU unit, NVIDIA A100 80 GB SXM (NVLink), 4-CPU cores, AMD EPYC 7702 2-3.35 GHz, and 1 TB RAM.

The *efficiency* is computed during fine-tuning via the Impact tracker toolkit (Henderson et al., 2020): the total power, run time in hours, GPU usage in hours, and estimated carbon footprint. To maximise these, we inverse the computational efficiency features through multiplying them by -1.

To measure *fairness*, we choose three social bias evaluation datasets: CROWS-PAIRS (Nangia et al., 2020), STEREOSET (Nadeem et al., 2021), and WINOBIAS (Zhao et al., 2018). In these datasets, one sentence is always more stereotyping than the other. Following Nangia et al. (2020), we use MLM scoring (Salazar et al., 2020) to score the pairs. The final metrics account for cases (%) in which a less stereotyping sentence is the most probable.

For the sake of space, we present the results on the *Borda* procedure in the basic, weighted, and two-step aggregation settings (§2.4). We assign the weights vector as $(0.4, 0.3, 0.3)$ to performance, efficiency, and fairness. The weights are introduced to increase the impact of performance. We use σ^{am} as the baseline and interim rankings by each criterion individually as references.

Results. Table 5 shows that DEBERTA is the winner according to σ^{am} and *Borda*. However, it requires more computational resources than the other systems and is mediocre in detecting social biases. As a result, it is not the best system in any user-oriented ranking. In this scenario, *Borda* tends to favour the distilled systems (DISTILROBERTA and DISTILBERT) due to their computational efficiency, which has the highest impact on the ranking with four criteria assigned per task. The weighted *Borda* ranks DISTILBERT, ALBERT, and BERT as the top-3 systems. In its turn, the weighted 2-step *Borda* prefers ALBERT first, followed by BERT and DISTILBERT. ALBERT is selected as the winner by the fairness ranking only and occupies the middle positions in the two other rankings. ROBERTA drops down drastically from the second rank (σ^{am}), whilst GPT2 remains in the least-3 systems.

Case study discussion. Overall, our setup follows DYNASCORE (Ma et al., 2021), where the microe-

Rank	$\sigma^{am}_{\text{Performance}}$	**Borda**	**Weighted Borda**	**Weighted 2-step Borda**	**Borda Performance**	**Borda Efficiency**	**Borda Fairness**
1	⬛ 82.73	🟪 267.0 ↑4	🎀 10.75 ↑5	🟦 4.30 ↑2	⬛ 56.5 ↕0	🟪 223.0 ↑4	🟦 19.0 ↑2
2	💋 82.52	🎀 245.0 ↑4	🟦 9.83 ↑1	🟨 3.60 ↑2	💋 49.0 ↕0	🎀 216.0 ↑4	🎀 18.0 ↑4
3	🟦 80.94	🟨 166.0 ↑1	🟨 8.96 ↑1	🎀 3.40 ↑3	🟦 32.5 ↕0	🟨 120.0 ↑1	🟨 14.0 ↑1
4	🟨 79.20	🟦 154.0 ↓1	🟪 8.63 ↑1	⬛ 3.00 ↓3	🟨 32.0 ↕0	🟦 103.0 ↓1	⬛ 11.00 ↓3
5	🟪 78.56	💋 144.0 ↓3	⬛ 7.17 ↓4	🟪 2.90 ↕0	🟪 17.0 ↕0	💋 91.0 ↓3	🟧 11.0 ↑1
6	🎀 77.89	🟧 10.0 ↑1	💋 7.04 ↓4	💋 2.60 ↓3	🎀 11.0 ↕0	🟧 84.0 ↑1	🟪 7.00 ↑1
7	🟧 75.95	⬛ 70.50 ↓6	🟧 5.47 ↕0	🟧 0.90 ↕0	🟧 8.0 ↕0	⬛ 3.00 ↓6	💋 4.00 ↓5

Table 5: Results of re-ranking the GLUE benchmark using the *Borda* rule in the simulated user-oriented scenario. Notations: 🟦= ALBERT; 🟨=BERT; 🎀=DISTILBERT; 💋=ROBERTA; 🟪=DISTILROBERTA; ⬛=DEBERTA; 🟧=GPT2.

conomic concept of MRS is used to compare performance, efficiency, and fairness metrics, followed by the weighted average score as the final ranking. Unlike the DYNASCORE results, we find that the average performance ranking is not preserved when using our voting rules. The most notable difference is with DEBERTA and ROBERTA systems, which may become penalised for low efficiency in our case. The reason is that in DYNASCORE, the weight of 0.5 is assigned to performance which blocks substantial changes in re-ranking.

4 Recommendations for Rules Choice

The information about the voting rules' properties can be used to choose the most suitable one to the user's preferences (Felsenthal and Machover, 2012). We also provide the following recommendations.

- The *Plurality* rule is a good choice if the user wants only the best systems in *each* criterion.
- If *all* ranking positions matter, use the *Borda* or *Dowdall* rules. Note that *Dowdall* assigns higher weights to the top positions.
- The *Threshold* rule is helpful in cases when the user wants to minimise the number of the low-performance criteria: the rule assigns the highest rank to the system that is considered the worst in the least amount of criteria.
- If the goal is to select the system that beats all the others in pairwise comparison, use the *Baldwin*, *Condorcet*, *Copeland*, or *Minimax* rules. These rules are Condorcet consistent; i.e. choose the CW if it exists. The main difference is how the rules behave when there is no CW. In particular, *Baldwin* selects the system that is left after elimination according to the *Borda* scores. *Copeland* chooses the system that dominates the others in more cases and is dominated by the least. In turn, *Minimax* selects the system with minimum defeat in pairwise comparison.
- The outcomes *may* contain equivalent alternatives (§3.1). Depending on the scenario, the user can select the rule that produces ties with a lower probability or *Dowdall* and *Borda* if their properties meet the preferences.

5 Conclusion and Future Work

This paper introduces novel aggregation procedures to rank and select the best-performing systems under the social choice theory principles. Our approach provides an alternative perspective of system evaluation in benchmarking and overcomes the standard mean aggregation limitations.

Our case studies show that VOTE'N'RANK provides interpretable decisions on the best and worst systems whilst accounting for missing performance scores and potential user preferences. The framework allows for finding scenarios in which a given system dominates the others. At the same time, the rule choice may depend on the particular research and development purpose. We provide recommendations based on the rules' properties and scenarios of the intended framework's application.

The application scope of VOTE'N'RANK is not limited and may be easily extended to other applied ML areas. In our future work we hope to explore applications of the social choice theory in the *multilingual* and *multimodal* benchmarking.

6 Limitations

Robustness. In the robustness experiments, the *Copeland* and *Minimax* rules are less sensitive to performance score drops than σ^{am} and σ^{og}. However, in certain circumstances *Minimax* may display low resistance to such corruption due to its nature, which is analysed in §3.3. Other robustness evaluation settings can be considered, such as sensitivity to removing and adding new tasks (Procaccia et al., 2007; Colombo et al.), which are out of scope of this work.

Ambiguity. Almost all rules in our study allow ties or the recognition of systems as equivalent. This may result in non-resoluteness: the selection of multiple winning systems or the presence of many equivalencies in ranking. However, we empirically observe no or a few ties using the *Dowdall*, *Borda*, and *Copeland* rules, whilst *Minimax* and *Plurality* treat a significant number of systems as equivalent due to their properties (§3.1). VOTE'N'RANK does not currently support any additional tie-breaking rules to be applied in this case. The only exception here is the *Threshold rule* that gives only one winner in almost all cases due to the built-in tie-breaking procedure.

Independence of irrelevant alternatives. The violation of the IIA axiom in applications is a well-known fundamental aspect in the social choice theory, and the voting rules can violate the IIA with different probabilities (Dougherty and Heckelman, 2020). IIA violation may imply undesirable behavior: submitting a new system to the leaderboard affects the relative ranking of the other systems. However, we empirically show that *Copeland* and *Minimax* are less likely to violate IIA than *Plurality* and *Borda* rules (§3.1). The IIA assumption may be unrealistic in practice as it takes no account of perfect or near-perfect substitutes (Suppes, 1965).

Lack of ground truth. Comparison of the aggregation procedures is hindered by the absence of the correct ranking, especially when votes are noisy and incomplete. There is no universal answer to the question of how the systems on the multi-task benchmarks should be preferred. However, we hope to contribute from a practical standpoint, offering an alternative approach to the mean aggregation procedure.

7 Ethical Considerations

Stereotypes and discrimination in LMs' pre-training data can lead to representation biases against race, religion, and social minorities. Our framework allows ranking systems to account for sensitive attributes (Celis et al., 2018), e.g. gender and nationality, or to find the trade-off between multiple criteria, e.g. performance and fairness (Baldini et al., 2022). The rank aggregation rules have been widely adopted to information retrieval and recommendation systems (Dwork et al., 2001; Masthoff, 2011). We assume that translation of the social choice theory into the system evaluation problems may improve the user experience by selecting systems that best satisfy evaluative criteria and individual or group preferences in downstream applications.

Acknowledgements

Mark Rofin, Mikhail Florinskiy, and Daniel Karabekyan were supported by the grant for research centres in the field of AI provided by the Analytical Centre for the Government of the Russian Federation (ACRF) in accordance with the agreement on the provision of subsidies (identifier of the agreement 000000D730321P5Q0002) and the agreement with the HSE University No. 70-2021-00139. We acknowledge the computational resources of HPC facilities at the HSE University. We would also like to thank colleagues from the IT University of Copenhagen and the anonymous reviewers for their comments on this paper.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. 2021. Deep Reinforcement Learning at the Edge of the Statistical Precipice. *Advances in Neural Information Processing Systems*, 34.

Mark Aizerman and Fuad Aleskerov. 1995. *Theory of Choice*, volume 38. North Holland.

Fuad Aleskerov, Vyacheslav V Chistyakov, and Valery Kalyagin. 2010. The threshold aggregation. *Economics Letters*, 107(2):261–262.

Kenneth J Arrow. 2012. Social choice and individual values. In *Social Choice and Individual Values*. Yale university press.

Ioana Baldini, Dennis Wei, Karthikeyan Natesan Ramamurthy, Moninder Singh, and Mikhail Yurochkin. 2022. Your fairness may vary: Pretrained language model fairness in toxic text classification. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2245–2262, Dublin, Ireland. Association for Computational Linguistics.

John Bartholdi, Craig A Tovey, and Michael A Trick. 1989. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and welfare*, 6(2):157–165.

Anya Belz, Shubham Agarwal, Anastasia Shimorina, and Ehud Reiter. 2021. A systematic review of reproducibility research in natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 381–393, Online. Association for Computational Linguistics.

Alessio Benavoli, Giorgio Corani, and Francesca Mangili. 2016. Should We Really Use Post-Hoc Tests Based on Mean-Ranks? *Journal of Machine Learning Research*, 17(5):1–10.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Duncan Black et al. 1958. The theory of committees and elections.

Samuel R. Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855, Online. Association for Computational Linguistics.

Felix Brandt and Hans Georg Seedig. 2016. On the Discriminative Power of Tournament Solutions. In *Operations Research Proceedings 2014*, pages 53–58. Springer.

L Elisa Celis, Damian Straszak, and Nisheeth K Vishnoi. 2018. Ranking with Fairness Constraints. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Monojit Choudhury and Amit Deshpande. 2021. How Linguistically Fair are Multilingual Pre-trained Language Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12710–12718.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Pierre Colombo, Nathan Noiry, Ekhine Irurozki, and Stephan CLEMENCON. What are the Best Systems? New Perspectives on NLP Benchmarking. In *Advances in Neural Information Processing Systems*.

Pierre Jean A Colombo, Chloé Clavel, and Pablo Piantanida. 2022. InfoLM: A New Metric to Evaluate Summarization & Data2Text Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10554–10562.

Adiel Teixeira De Almeida, Danielle Costa Morais, and Hannu Nurmi. 2019. *Systems, procedures and voting rules in context: A primer for voting rule selection*, volume 9. Springer.

Mostafa Dehghani, Yi Tay, Alexey A Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. 2021. The Benchmark Lottery. *arXiv preprint arXiv:2107.07002*.

Janez Demšar. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine learning research*, 7:1–30.

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

Keith L Dougherty and Jac C Heckelman. 2020. The probability of violating arrow's conditions. *European Journal of Political Economy*, 65:101936.

Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. 2001. Rank Aggregation Methods for the Web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622.

Paul H Edelman. 2015. The myth of the condorcet winner. *Supreme Court Economic Review*, 22(1):207–219.

Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1325–1335, Online. Association for Computational Linguistics.

Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4846–4853, Online. Association for Computational Linguistics.

Dan S Felsenthal and Moshé Machover. 2012. *Electoral Systems: Paradoxes, Assumptions, and Procedures*. Springer Science & Business Media.

John Geanakoplos. 2005. Three brief proofs of arrow's impossibility theorem. *Economic Theory*, 26(1):211–215.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.

Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2020. Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning. *Journal of Machine Learning Research*, 21(248):1–43.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A Lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Jonathan Levin and Barry Nalebuff. 1995. An introduction to vote-counting schemes. *The Journal of Economic Perspectives*, 9(1):3–26.

Guohao Li, Feng He, and Zhifan Feng. 2021. A CLIP-Enhanced Method for Video-Language Understanding. *arXiv preprint arXiv:2110.07137*.

Linjie Li, Yen-Chun Chen, Yu Cheng, Zhe Gan, Licheng Yu, and Jingjing Liu. 2020. HERO: Hierarchical encoder for Video+Language omni-representation pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2046–2065, Online. Association for Computational Linguistics.

Linjie Li, Jie Lei, Zhe Gan, Licheng Yu, Yen-Chun Chen, Rohit Pillai, Yu Cheng, Luowei Zhou, Xin Eric Wang, William Yang Wang, et al. VALUE: A Multi-Task Benchmark for Video-and-Language Understanding Evaluation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang Liu, Fan Yang, Daniel Campos, Rangan Majumder, and Ming Zhou. 2020. XGLUE: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Zhiyi Ma, Kawin Ethayarajh, Tristan Thrush, Somya Jain, Ledell Wu, Robin Jia, Christopher Potts, Adina Williams, and Douwe Kiela. 2021. Dynaboard: An Evaluation-As-A-Service Platform for Holistic Next-Generation Benchmarking. In *Advances in Neural Information Processing Systems*, volume 34, pages 10351–10367. Curran Associates, Inc.

Judith Masthoff. 2011. Group Recommender Systems: Combining Individual Models. In *Recommender systems handbook*, pages 677–702. Springer.

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, Tom Kwiatkowski, Patrick Lewis, Yuxiang Wu, Heinrich Küttler, Linqing Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Edouard Grave, Ikuya Yamada, Sonse Shimaoka, Masatoshi Suzuki, Shumpei Miyawaki, Shun Sato, Ryo Takahashi, Jun Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, Pavel Smrz, Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen-tau Yih. 2021. NeurIPS 2020 EfficientQA Competition: Systems, Analyses and Lessons Learned. In *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, volume 133 of *Proceedings of Machine Learning Research*, pages 86–111. PMLR.

Swaroop Mishra and Anjana Arunkumar. 2021. How Robust are Model Rankings: A Leaderboard Customization Approach for Equitable Evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13561–13569.

Giuseppe Munda. 2012. Choosing aggregation rules for composite indicators. *Social Indicators Research*, 109(3):337–354.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Christina Nießl, Moritz Herrmann, Chiara Wiedemann, Giuseppe Casalicchio, and Anne-Laure Boulesteix. 2022. Over-optimism in Benchmark Studies and the Multiplicity of Design and Analysis Options when Interpreting Their Results. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12(2):e1441.

Hannu Nurmi. 1983. Voting procedures: A summary analysis. *British Journal of Political Science*, 13(2):181–208.

Simon Ott, Adriano Barbosa-Silva, Kathrin Blagec, Jan Brauner, and Matthias Samwald. 2022. Mapping Global Dynamics of Benchmark Creation and Saturation in Artificial Intelligence. *Nature Communications*, 13(1):6793.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.

Ariel D Procaccia, Jeffrey S Rosenschein, and Gal A Kaminka. 2007. On the Robustness of Preference Aggregation in Noisy Environments. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–7.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-text Transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Inioluwa Deborah Raji, Emily Denton, Emily M Bender, Alex Hanna, and Amandalynne Paullada. 2021. AI and the Everything in the Whole Wide World Benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Pedro Rodriguez, Joe Barrow, Alexander Miserlis Hoyle, John P. Lalor, Robin Jia, and Jordan Boyd-Graber. 2021. Evaluation examples are not equally informative: How should that change NLP leaderboards? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4486–4503, Online. Association for Computational Linguistics.

Anna Rogers. 2019. How the Transformers Broke NLP Leaderboards. `https://hackingsemantics.xyz/2019/leaderboards`.

Sebastian Ruder. 2021. Challenges and Opportunities in NLP Benchmarking. `http://ruder.io/nlp-benchmarking`.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Tatiana Shavrina and Valentin Malykh. 2021. How not to lie with a benchmark: rearranging NLP leaderboards. In *I (Still) Can't Believe It's Not Better! NeurIPS 2021 Workshop*.

Minchul Shin, Jonghwan Mun, Kyoung-Woon On, Woo-Young Kang, Gunsoo Han, and Eun-Sol Kim. 2021. Winning the ICCV'2021 VALUE Challenge: Task-aware Ensemble and Transfer Learning with Visual Concepts. *arXiv preprint arXiv:2110.06476*.

P Suppes. 1965. Preference, utility and subjective probability. inhandbook of mathematical psychology, ed. rd luce, rr bush and eh galanter, 3, 249–410.

Neeraj Varshney, Swaroop Mishra, and Chitta Baral. 2022. ILDAE: Instance-level difficulty analysis of evaluation data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3412–3425, Dublin, Ireland. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A Stickier Benchmark for General-purpose Language Understanding Systems. *Advances in Neural Information Processing Dystems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing* *and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Boxin Wang, Chejian Xu, Shuohang Wang, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Awadallah, and Bo Li. 2021. Adversarial GLUE: A Multi-Task Benchmark for Robustness Evaluation of Language Models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Jiangnan Xia, Zuyi Bao, Liwei Peng, and Luo Si. 2019b. StructBERT: Incorporating Language Structures into Pre-training for Deep Language Understanding. In *International Conference on Learning Representations*.

Zeerak Waseem, Smarika Lulz, Joachim Bingel, and Isabelle Augenstein. 2021. Disembodied Machine Learning: On the Illusion of Objectivity in NLP. *arXiv preprint arXiv:2101.11974*.

Geoffrey I Webb. 2000. MultiBoosting: A Technique for Combining Boosting and Wagging. *Machine learning*, 40(2):159–196.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

Xiyou Zhou, Zhiyu Chen, Xiaoyong Jin, and William Yang Wang. 2021. HULK: An energy efficiency benchmark platform for responsible natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 329–336, Online. Association for Computational Linguistics.

A Aggregation Procedures

A.1 Examples

Rank	Task 1	Task 2	Task 3	Task 4	Task 5
1	m_A	m_A	m_B	m_C	m_D
2	m_B	m_C	m_D	m_B	m_B
3	m_C	m_D	m_C	m_D	m_C
4	m_D	m_B	m_A	m_A	m_A

Table 6: A toy leaderboard for illustration purposes.

Rank	Task 1	Task 2	Task 3	Task 4	Task 5
1	m_B	m_C	m_B	m_C	m_D
2	m_C	m_D	m_D	m_B	m_B
3	m_D	m_B	m_C	m_D	m_C

Table 7: The leaderboard based on Table 6 used for describing the *Baldwin* rule.

This appendix provides illustrative examples on how our voting rules work. Here, suppose we have a toy leaderboard with five tasks and four systems as shown in Table 6. The systems are ranked within each task by their performance score. We now compute the rankings using each voting rule.

Scoring rules.

- *Plurality rule* assigns the score of 2 to m_A and scores of 1 to m_B, m_C, and m_D.
- According to the *Borda rule*, the systems that take the first position get 3 points for each task, 2 points are awarded for the second position, etc. As a result, the systems receive the following *Borda* scores: $m_A = 6$, $m_B = 9$, $m_C = 8$, and $m_D = 7$. The system m_B has the highest score and is chosen as the best one.
- For the *Dowdall* rule scoring vector, we get the following scores: $S_{m_A} = 2.75$, $S_{m_B} = 2.75$, $S_{m_C} = 2.5$, and $S_{m_D} = 1.75 + 2/3$. There is a tie between the systems m_A and m_B, and both of them are considered the best models.

Iterative scoring rules.

- For the *Threshold* rule scoring vector (1,1,1,0), we get the following scores: $S_{m_A} = 2$, $S_{m_B} = 4$, $S_{m_C} = 5$, and $S_{m_D} = 4$. The system m_C is the winner. If there is a tie, the scoring vector (1,1,0,0) is further applied for only tied systems.
- The *Baldwin* rule: first, we calculate the *Borda* scores as mentioned above. Second, we eliminate the system m_A since it has the lowest score (see Table 7). Next, we re-calculate the *Borda* scores for a new scoring vector (2,1,0) and get the following results: $S_{m_B} = 6$, $S_{m_C} = 5$, and $S_{m_D} = 4$. At this step, the system m_D is eliminated. Finally, we re-calculate the results for the scoring vector (1,0). The results are $S_{m_B} = 3$ and $S_{m_C} = 2$, and the system m_B is declared the winner.

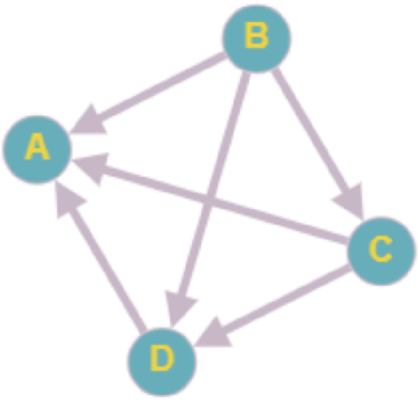


Figure 3: A toy graph example of the majority relation μ based on Table 6.

Majority-relation based rules. The majority relation in this example is illustrated in Figure 3.

- The system m_B is the *Condorcet* winner as it beats each of the alternatives. Note that since all majority-relation based rules (*Copeland* and *Minimax*) are *Condorcet consistent*, they declare the system m_B the winner as well. Let us illustrate it in more detail.
- The *Copeland* rule scores are $u(m_A) = -3$, $u(m_B) = 3$, $u(m_C) = 1$, $u(m_D) = -1$. The system m_B is the winner as it has the highest $u(x)$.
- The *Minimax* rule scores are $rank(m_A) = -3$, $rank(m_B) = 0$, $rank(m_C) = -3$, $rank(m_D) = -3$. Here, the system m_B has the highest rank.

A.2 Properties

We consider the following properties to describe our voting rules and summarise them in Table 8.

- **Transitivity.** There are no cycles in the final ranking. An example of the cycle is a situation, where m_A is better than m_B, m_B is better than m_C, and m_C is better than m_A.

	Plurality	*Borda*	*Dowdall*	*Threshold*	*Baldwin*	*Copeland*	*Minmax*
Transitivity	✓	✓	✓	✓	✓	✓	✓
Anonymity	✓	✓	✓	✓	✓	✓	✓
Unanimity	✓*	✓	✓	✓	✓	✓	✓
IIA	✗	✗	✗	✗	✗	✗	✗
Monotonicity	✓	✓	✓	✓	✗	✓	✓
Majority	✓	✗	✓	✗	✓	✓	✓
CW	✗	✗	✗	✗	✓	✓	✓
Condorcet loser	✗	✓	✗	✗	✓	✓	✗
Sum	5	5	5	4	6	7	6

Table 8: Rules and their properties. *The non-winning Pareto-dominated systems can be tied.

- **Unanimity (Pareto efficacy).** If the system m_A is ranked higher than m_B according to all criteria, then m_A should be ranked higher.
- **Non-dictatorship (Anonymity).** There is no single criterion that defines the final ranking.
- **Independence of irrelevant alternatives (IIA).** For any two systems, the information about other systems should not influence their ranking.
- **Monotonicity.** If the system m_A is the winner and it started to rank higher according to one of the criteria, then it should still be the winner.
- **Majority criterion.** If the system m_A is considered the best by more than 50% criteria, then it should be the winner.
- **Condorcet winner criterion**. This criterion is a stronger version of the Majority criterion. If the system m_A is the Condorcet winner (CW), it should be the winner according to the rule.
- **Condorcet loser criterion.** If the system m_A is the Condorcet loser ($a \mu M^L$ for any $a \in A$), it should never be the winner according to the rule.

Recall that the *Condorcet* rule by definition complies with the Condorcet winner and loser criteria. The other properties can not be checked in application to benchmarking since the rule is defined on a restricted domain: it does not provide the results on any possible combination of rankings.

There is no single best voting rule since none of them satisfies properties of the Arrow's impossibility theorem (Arrow, 2012; Geanakoplos, 2005): transitivity, unanimity, non-dictatorship, and independence of irrelevant alternatives (IIA).

B Case Studies

We do not report the agreement rate, the Kendall Tau (τ) correlation, and the IIA results for VALUE since we are given only up to 7 evaluated alternatives: CRAIG.STARR (Shin et al., 2021), DUKG (Li et al., 2021), HUMAN, and four HERO-based configurations (Li et al., 2020). The HERO-based baselines are trained in the following settings: single-task training (ST), multi-task training (MT) by tasks or domains, all-task training (AT) and AT first then ST (AT -> ST). We refer to the configurations as follows:

- HERO$_1$: AT->ST, PT+FT;
- HERO$_2$: AT->ST, FT-only;
- HERO$_3$: ST, PT+FT;
- HERO$_4$: ST, FT-only.

The VALUE results. Table 10 and Table 11 show the VALUE re-ranking results over missing/non-missing scores. ✗ means that the given aggregation method does not operate over missing values. In the first case, we observe that the *Copeland* and *Minimax* rules generally agree on the final outcomes except for the fifth and sixth positions. The rules select HUMAN as the winner. At the same time DUKG and HERO$_1$ have the same *Minimax* values, and the *Minimax* values of the least-3 systems are also equal. In the second case, we omit HUMAN due to missing scores on 6 out of 11 tasks for comparable interpretation. However, there are 3 tied alternatives in the *Minimax* and *Dowdall* outcomes. Interestingly, all methods are consistent in conclusions on the top-3 systems, with the *Minimax* treating DUKG and HERO$_1$ as equal alternatives. The σ^{og}, *Minimax*, *Plurality*, *Dowdall*, and *Borda* rules make equal decisions on the final outcomes.

Rank	σ^{am}	σ^{gm}	σ^{og}	Copeland	Minimax	Plurality	Dowdall	Borda
1	😡90.62	😡90.04 ↕0	😡0.066 ↕0	🙋20.00 ↑3	😡0 ↕0	🙋4.00 ↑3	🙋4.98 ↑3	😡155.00 ↕0
2	🔵90.39	🔵89.84 ↕0	🔵0.068 ↕0	😡19.00 ↓1	🙋0 ↑2	😡2.50 ↓1	😡4.25 ↓1	🔵154.50 ↕0
3	👾90.29	👾89.75 ↕0	👾0.068 ↕0	🔵18.00 ↓1	🔵−4.50 ↓1	🔵1.00 ↓1	🔵3.62 ↓1	👾153.00 ↕0
4	🙋89.79	🙋88.80 ↕0	🐛0.073 ↑1	👾15.00 ↓1	👾−5.00 ↓1	👾0.50 ↓1	👾3.29 ↓1	🙋145.50 ↕0
5	🐛89.25	🐛88.75 ↕0	🙋0.074 ↓1	🐛13.00 ↕0	🐛−7.50 ↕0	🩰0.00 ↑9	🐛2.11 ↕0	🐛141.50 ↕0
6	👱86.65	👱85.93 ↕0	👹0.089 ↑1	👱11.00 ↕0	🩰−8.00 ↑8	🐤0.00 ↑12	👱1.16 ↕0	👱116.50 ↕0
7	👹86.09	👹85.38 ↕0	🤖0.10 ↑5	👵9.00 ↑1	🐤−8.00 ↑11	🦁0.00 ↑12	👵1.06 ↑1	👵108.00 ↑1

Table 9: Results of re-ranking the SGLUE benchmark. The model rank changes are depicted with arrows, whilst the superscripts denote scores assigned by the voting method. Notations: 🙋=HUMAN; 😡=ERNIE; 👾=DEBERTA/TURINGNLRV4; 👱=NEZHA-PLUS; 🔵=T5+UDG; 🐤=BERT++; 👹=PAI ALBERT; 👵=ROBERTA-ICETS; 🦁=GPT-3 FEW-SHOT; 🩰=IPET (ALBERT) FEW-SHOT; 🐛=T5;🤖=AILABS TEAM.

Rank	σ^{am}	σ^{gm}	σ^{og}	Copeland	Minimax	Plurality	Dowdall	Borda
1	✗	✗	✗	🙋🏾6.00 ↕0	🙋🏾0 ↕0	✗	✗	✗
2	✗	✗	✗	🦉4.00 ↕0	🦉−6.00 ↕0	✗	✗	✗
3	✗	✗	✗	🦁2.00 ↕0	🦁−10.00 ↕0	✗	✗	✗
4	✗	✗	✗	👹0.00 ↕0	👹−10.00 ↕0	✗	✗	✗
5	✗	✗	✗	🩰−2.00 ↕0	🔵−11.00 ↕0	✗	✗	✗
6	✗	✗	✗	🔵−4.00 ↕0	🩰−11.00 ↕0	✗	✗	✗
7	✗	✗	✗	🎩−6.00 ↕0	🎩−11.00 ↕0	✗	✗	✗

Table 10: Results of re-ranking the VALUE benchmark over missing scores. Changes in the system ranks are depicted with arrows, whilst the superscripts denote scores assigned by the aggregation procedure. Notations: 🙋🏾=HUMAN; 🦉=CRAIG.STARR; 🦁=DUKG; 👹=HERO$_1$; 🩰=HERO$_2$; 🔵=HERO$_3$; 🎩=HERO$_4$.

Rank	σ^{am}	σ^{gm}	σ^{og}	Copeland	Minimax	Plurality	Dowdall	Borda
1	🦉62.87	🦉49.96 ↕0	🦉0.365 ↕0	🦉5.00 ↕0	🦉0 ↕0	🦉9.00 ↕0	🦉10.00 ↕0	🦉53.00 ↕0
2	🦁60.00	🦁46.30 ↕0	🦁0.381 ↕0	🦁3.00 ↕0	🦁−10.00 ↕0	🦁1.00 ↕0	🦁5.17 ↕0	🦁39.00 ↕0
3	👹57.58	👹44.12 ↕0	👹0.399 ↕0	👹1.00 ↕0	👹−10.00 ↕0	👹1.00 ↕0	👹4.08 ↕0	👹30.00 ↕0
4	🩰56.96	🩰43.22 ↕0	🔵0.403 ↑1	🩰−1.00 ↕0	🔵−11.00 ↑1	🔵0.00 ↑1	🔵2.87 ↑1	🔵20.00 ↑1
5	🔵56.07	🔵42.81 ↕0	🩰0.404 ↓1	🔵−3.00 ↕0	🩰−11.00 ↓1	🩰0.00 ↓1	🩰2.82 ↓1	🩰18.00 ↓1
6	🎩52.59	🎩37.56 ↕0	🎩0.438 ↕0	🎩−5.00 ↕0	🎩−11.00 ↕0	🎩0.00 ↕0	🎩2.02 ↕0	🎩5.00 ↕0

Table 11: Results of re-ranking the VALUE benchmark over non-missing scores. The HUMAN results are discarded due to missing scores. Changes in the system ranks are depicted with arrows, whilst the superscripts denote scores assigned by the aggregation procedure. Notations: 🦉=CRAIG.STARR; 🦁=DUKG; 👹=HERO$_1$; 🩰=HERO$_2$; 🔵=HERO$_3$; 🎩=HERO$_4$.

Combining Parameter-efficient Modules for Task-level Generalisation

Edoardo M. Ponti[ϵ] **Alessandro Sordoni**[σ] **Yoshua Bengio**[μ,ρ] **Siva Reddy**[μ,γ]
[ϵ]University of Edinburgh [μ]Mila – Quebec AI Institute [γ]McGill University
[σ]Microsoft Research Montréal [ρ]Université de Montréal
eponti@ed.ac.uk

Abstract

A modular design encourages neural models to disentangle and recombine different facets of knowledge to generalise more systematically to new tasks. In this work, we assume that each task is associated with a subset of latent skills from an (arbitrary size) inventory. In turn, each skill corresponds to a parameter-efficient (sparse / low-rank) model adapter. By jointly learning adapters and a routing function that allocates skills to each task, the full network is instantiated as the average of the parameters of active skills. We propose several inductive biases that encourage re-usage and composition of the skills, including variable-size skill allocation and a dual-speed learning rate. We evaluate our latent-skill model in two main settings: 1) multitask reinforcement learning for instruction following on 8 levels of the BabyAI platform; and 2) few-shot fine-tuning of language models on 160 NLP tasks of the CrossFit benchmark. We find that the modular design of our network enhances sample efficiency in reinforcement learning and few-shot generalisation in supervised learning, compared to a series of baselines. These include models where parameters are fully shared, task-specific, or conditionally generated (HyperFormer), as well as sparse mixture-of-experts (Task-MoE).

1 Introduction

Task-level generalisation involves training a model on multiple tasks (in parallel or sequentially) and then performing zero-shot inference or few-shot adaptation on new tasks (Caruana, 1997; Ye et al., 2021). However, the training signals from different tasks may interfere with each other (McCloskey and Cohen, 1989) or lead to catastrophic forgetting of previous knowledge (French, 1999). Moreover, when the nature of the tasks varies between training and evaluation, models often struggle to generalise systematically (Hupkes et al., 2020).

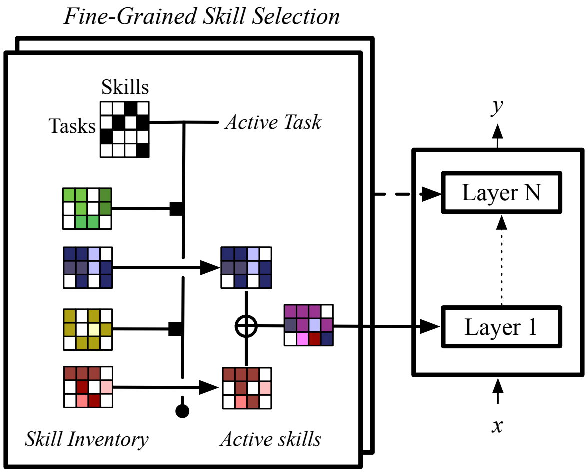


Figure 1: A diagram of our latent-skill model: 1) a row of the task–skill binary matrix is selected according to the active task; 2) the sparse (SFT) or low-rank (LoRA) adapters corresponding to active skills from a layer-specific inventory are combined; 3) the resulting combination is applied to the weights of a neural network, e.g. BART in our multi-task experiments.

A potential solution to these challenges consists in a modular design of neural architectures. In fact, these are endowed with an inductive bias towards updating and activating modules locally and asynchronously, thus preventing negative interference (Jacobs et al., 1991b). Additionally, by disentangling autonomous facets of knowledge (also known as *skills*) that are re-used and recombined in original ways for new tasks, modularity facilitates systematic generalisation (Alet et al., 2018; Ponti, 2021; Kingetsu et al., 2021, *inter alia*).

Previous work on task-level generalisation focused on settings where the skills relevant for new tasks are already known *a priori* (Pfeiffer et al., 2020; Ansell et al., 2022). However, this requires expert knowledge, possibly with sub-optimal granularity and limited to a few domains. In alternative, mixture-of-expert (MoE) methods such as Task-MoE (Kudugunta et al., 2021a) learn a routing function that allocates modules to tasks end-to-end. However, MoEs mostly focus on scaling large lan-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 687–702
May 2-6, 2023 ©2023 Association for Computational Linguistics

guage models trained from scratch, and their effectiveness for out-of-domain generalisation is a moot point (Artetxe et al., 2021; Fedus et al., 2022).

In this work, we introduce a modular architecture that can efficiently adapt to new tasks, by virtue of a series of inductive biases that facilitate generalisation. The full model is shown in Figure 1. For each task, we learn a binary vector that specifies which skills (from a fixed inventory) are used to solve it. In the example, skills 2 and 4 are active. We implement each skill as a parameter-efficient adapter, either sparse (Ansell et al., 2022, SFT) or low-rank (Hu et al., 2021, LoRA). The parameters corresponding to every active skill (red and blue in the example) are then combined, by simple averaging. The resulting combination adapts a pretrained model, such as BART (Lewis et al., 2020), towards a specific task during multitask learning. The variable-size allocation of skills encourages their re-usage and re-combination in new tasks. To promote a coarse-to-fine dynamic where skill allocation is determined before skill-specific parameters, we explore a dual-speed learning rate for these two components (see Section 2.3).

We evaluate our model on reinforcement learning on BabyAI (Chevalier-Boisvert et al., 2019), a platform for instruction following in a simulated environment, and supervised learning on CrossFit (Ye et al., 2021), a benchmark recasting 160 NLP tasks as text-to-text generation problems. Compared to a series of competitive baselines, we obtain higher sample efficiency and higher performance in few-shot adaptation to held-out tasks. In particular, our latent-skill model surpasses non-modular methods for multi-task adaptation, such as HyperFormers (Karimi Mahabadi et al., 2021) and CA-MTL (Pilault et al., 2021), and mixture-of-experts methods such as Task-MoE (Kudugunta et al., 2021a).

Finally, we probe the learnt task–skill allocation matrix and illustrate how our method also enhances interpretability, as it discovers which pairs of tasks require common skills. The code is made available at `https://github.com/McGill-NLP/polytropon`.

2 A Latent-Skill Multitask Model

The goal of multitask learning in modelling a set of tasks $\mathcal{T} = (\mathcal{T}_1, \dots, \mathcal{T}_{|\mathcal{T}|})$ is two-fold: 1) increasing sample efficiency on each seen task by borrowing statistical strength from the others; and 2) attaining systematic generalisation, the ability to adapt robustly to new tasks, possibly based on a few target-domain examples. In particular, in supervised learning, each task $\mathcal{T}_i$ is associated with a dataset $\mathcal{D}_i \triangleq \{(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$ and a loss function $\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y})$, where $\mathbf{x}$ is an input and $\mathbf{y}$ is an output. In reinforcement learning, each task is characterised by an initial state distribution $q(\mathbf{x}_1)$, a transition distribution $q(\mathbf{x}_{t+1} \mid \mathbf{x}_t, a_t)$, and a loss function $\mathcal{L}(\mathbf{x}_1, a_1, \dots, \mathbf{x}_h, a_h) \to \mathbb{R}$,[1] where $\mathbf{x}$ is a state, a is an action, and h is the temporal horizon of each episode. Thus, each task defines a Markov Decision Process (MDP).

Fully sharing the parameters of a model across tasks (Stickland and Murray, 2019) may exhaust its capacity or create interference among the gradients from task-specific losses (Wang et al., 2021). These limitations can be countered by instead composing task-specific adapters (Pfeiffer et al., 2021) or softly sharing parameters (Ponti et al., 2021a; Karimi Mahabadi et al., 2021; Ansell et al., 2021). The first method leads to an explosion in parameter count, which grows with the number of tasks. Moreover, the second method suffers during few-shot adaptation to new tasks as the entangled (i.e., non-modular) knowledge may overfit the training distribution.

Instead, we posit that there exists a (possibly small) fixed inventory of skills $\mathcal{S} = (\mathcal{S}_1, \dots, \mathcal{S}_{|\mathcal{S}|})$, where $|\mathcal{S}| \ll |\mathcal{T}|$. Each skill is an independent facet of knowledge that is reused across a subset of tasks. These assumptions guarantee both scalability and modularity. In particular, we seek to create a model that jointly learns which skills are active for which task, aggregates the corresponding skill parameters according to some deterministic function, and maximises the multitask log-likelihood:

$$\begin{gathered}\operatorname*{arg\,max}_{Z,\Phi} \sum_{\mathcal{T}_i} \sum_{(\mathbf{x},y)\in\mathcal{T}_i} \log p(y \mid \mathbf{x}, \boldsymbol{\vartheta}_i)\, p(Z)\, p(\Phi), \\ \text{where } \boldsymbol{\vartheta}_i = \delta(Z_i, \Phi, \boldsymbol{\vartheta}_0)\end{gathered} \tag{1}$$

The learnable parameters, as shown in Figure 1, are: i) Z, which denotes the task–skill allocation matrix, representing a soft partition of skills across tasks; ii) Φ, a tensor containing the parameters of each skill. As we explain in Section 2.4, these consist in either sparse or low-rank linear adapters. δ is a deterministic combination function that takes as input the task–skill allocation vector Z_i for task $\mathcal{T}_i$,

---

[1] This is the negative of the reward: $\mathcal{L}(\cdot) = -\mathcal{R}(\cdot)$.

the skill parameters Φ and some shared parameters across tasks $\boldsymbol{\vartheta}_0$ and creates a task-dependent set of parameters $\boldsymbol{\vartheta}_i$, i.e. in our model, we use simple averaging of the active skills (see Figure 1). $\boldsymbol{\vartheta}_i$ is then used to predict the task label y given the input $\mathbf{x}$. In what follows, we illustrate each component separately.

2.1 Soft Partitions

What is the best strategy to determine which skills are active for which task? The cognitively inspired notion of modularity at the level of structured inputs assumes that modules compete with each other for activation and updating (Bengio, 2017; Goyal et al., 2021). This intuition is translated in practice into a softmax across modules and top-k selection, such as in mixtures of experts (MoEs; Kudugunta et al., 2021a; Fedus et al., 2022). Nevertheless, we argue, modularity at the task level should reflect the fact that tasks fall into a hierarchy: more complex ones subsume simpler ones as sub-tasks. This idea has a long history, harking back to early works on MoEs (Jacobs et al., 1991b) and neural programme induction (Rosenbaum et al., 2019). Hence, we allow for variable-size subsets of skills.

As a consequence, we assume that the matrix $Z \in \{0,1\}^{|\mathcal{T}| \times |\mathcal{S}|}$ representing task–skill allocations is a soft partition: each cell z_{ij} is a binary scalar indicating if skill-specific parameters Φ_j are active for a certain task $\mathcal{T}_i$. However, being discrete, such a binary matrix is not differentiable and therefore cannot be learned end-to-end via gradient descent. Instead, we implement it as a collection of Bernoulli distributions continuously relaxed through a Gumbel-sigmoid (Maddison et al., 2017; Jang et al., 2017), which ensures stochasticity while allowing for differentiable sampling:

$$\hat{z}_{i,j} = \sigma\left[\log \frac{\sigma(z_{i,j})\, u}{(1-\sigma(z_{i,j}))\,(1-u)}^{1/\tau}\right] \\ u \sim \text{Uniform}(0,1). \tag{2}$$

In principle, either a coarse-grained soft partition can be learned globally for the entire neural network, or different fine-grained soft partitions can be assigned to each layer. We opt for the second alternative as it affords the model more flexibility and, foreshadowing, yields superior performance. Therefore, Z and Φ are henceforth assumed to be layer-specific, although we will omit layer indexes in the notation for simplicity's sake.

2.2 Skill-specific Parameters

Given the matrix row $\hat{Z}_i$ for task $\mathcal{T}_i$ from Equation (2) and a matrix of skill-specific parameters $\Phi \in \mathbb{R}^{|\mathcal{S}| \times d}$, where d is the dimension of the layer parameters, the aggregate of the parameters of active skills is superimposed to a base parameterisation $\boldsymbol{\vartheta}_0 \in \mathbb{R}^d$ shared across tasks. For instance, $\boldsymbol{\vartheta}_0$ may be either the initialisation from a pre-trained model or learned from scratch:

$$\boldsymbol{\vartheta}_i = \delta(Z_i, \Phi, \boldsymbol{\vartheta}_0) \triangleq \boldsymbol{\vartheta}_0 + \sum_{\mathcal{S}_j} \Phi_j \frac{\hat{z}_{i,j}}{\sum_{\mathcal{S}_j} \hat{z}_{i,j}}, \tag{3}$$

where $\boldsymbol{\vartheta}_i$ denote the parameters obtained by combining the active skills for the specific task $\mathcal{T}_i$. Note that we normalise the rows of the task–skill allocation matrix $\hat{Z}$ prior to composition because the variable number of active skills per task would otherwise affect the norm of the combined parameters $\boldsymbol{\vartheta}_i$, thus making training unstable.

2.3 Inductive Biases

A possible failure mode during training is a collapse into a highly entropic or non-sparse allocation matrix $\hat{Z}$, e.g., where all skills are active and skill-specific parameters remain general-purpose rather than specialising. Thus, we also provide an inductive bias to encourage the model to learn a low-entropy, highly-sparse allocation matrix.

In particular, we experiment with a dual-speed learning rate, setting its value for Z higher than for Φ. Intuitively, by accelerating learning of the soft partition matrix, to minimise the loss it becomes more convenient to discover better task–skill allocations over settling for general-purpose parameters that are agnostic with respect to the subset of active skills. This is reminiscent of the hypothesis that different kinds of knowledge require faster or slower rates of learning (Kahneman, 2011).

2.4 Parameter Efficiency

In order to keep the skills modular, each of them must correspond to a separate layer parameterisation. Nevertheless, this may lead to a significant increase in both time and space complexity. Thus, we explore parameter-efficient implementations of Φ that only add a negligible amount of parameters to the base model. In particular, we contemplate both sparse and low-rank approximations.

Sparse Fine-Tuning (SFT; Ansell et al., 2022) learns a highly sparse vector of differences Φ with respect to a base model ϑ_0. Its non-zero entries are identified by a binary matrix $M \in \{0,1\}^{|\mathcal{S}| \times d}$. For simplicity, we randomly draw M, as this was shown to perform almost on par with more sophisticated criteria without requiring a separate selection phase (Ansell et al., 2022). SFT is agnostic with respect to the underlying neural architecture.

Low-Rank Adapters (LoRA; Hu et al., 2021) is another method for parameter-efficient fine-tuning designed specifically for Transformer architectures. It factorises each weight of the linear projections inside self-attention as a batched matrix multiplication between two low-rank tensors $A \in \mathbb{R}^{|\mathcal{S}| \times o \times r}$ and $B \in \mathbb{R}^{|\mathcal{S}| \times r \times i}$, where r stands for the rank. In our model, each linear projection $f : \mathbb{R}^i \to \mathbb{R}^o$ is implemented as:

$$\mathbf{x}' = [W_0 + (\mathbf{z}^\top AB)]\mathbf{x} + \mathbf{b}_0 \qquad (4)$$

where $\Phi \triangleq \text{flatten}(AB)$.

We employed sparse adapters (LT-SFT) for the reinforcement learning experiments and low-rank adapters (LoRA) for the NLP experiments. The rationale behind this choice is showing that our method is effective independent of the choice of adapters.

2.5 Baselines

We measure the performance of our SKILLED model, where we learn the skill–task allocation matrix Z end-to-end, against these baselines, which derive from a certain configuration of Z:

- PRIVATE: each task has a separate model parameterisation. During few-shot adaptation, given that $\mathcal{T}_{train} \cap \mathcal{T}_{eval} = \emptyset$, this model cannot benefit from any transfer of information between training and evaluation tasks. This is equivalent to the special case where the task–skill allocation matrix is an identity matrix $Z = I$ of size $|\mathcal{T}| \times |\mathcal{T}|$ and $|\mathcal{S}| = |\mathcal{T}|$.
- SHARED: a shared skill is learnt on the training tasks and then fine-tuned for each evaluation task separately. This is equivalent to the special case where the task–skill allocation matrix is a matrix of ones $Z = \mathbf{1}$ of size $|\mathcal{T}| \times 1$ and $|\mathcal{S}| = 1$.
- EXPERT, where the task–skill allocation is contingent on expert knowledge about task relationships. Crucially, Z is fixed *a priori* rather than being learned.

In addition, for the NLP experiments, we compare our SKILLED method to two state-of-the-art baselines for multitask learning. In the first baseline, HYPER, parameters are softly shared (Karimi Mahabadi et al., 2021; Pilault et al., 2021). These methods generate adapters for the pre-trained model parameters with hyper-networks conditioned on task embeddings. Note that HYPER has a hidden connection with SKILLED, despite their apparent difference. In fact, we can draw an equivalence between i) binary skill vectors $\mathbf{z}$ and task embeddings; ii) the matrix of skill-specific parameters Φ and linear hyper-network weights. Crucially, in our case $\mathbf{z}$ is binary and modular, rather than continuous and entangled.

Secondly, we compare SKILLED with TASK-MOE (Kudugunta et al., 2021a). MoEs were originally conceived for scaling large language models trained from scratch, rather than model adaptation. Moreover, their modules consist of multiple feed-forward networks after the attention layer inside Transformers. The routing function to allocate modules to tasks is based on softmax and top-k selection. This, however, does not favour module re-usage, contrary to variable-size allocation, and instead creates a rich-gets-richer mechanism.

Contrary to their original formulation, we implement the modules of HYPER and TASK-MOE through LoRA adapters rather than Adapter layers and FFNs, respectively. This ensures that the number of updated parameters is similar to SKILLED as implemented in Equation (4) and thus their results are comparable. On top of this, LoRA was shown to offer a better trade-off between performance and efficiency (Liu et al., 2022).

3 Reinforcement Learning Experiments

3.1 Dataset

As a proof-of-concept experiment, we perform multitask reinforcement learning on the BabyAI platform (Chevalier-Boisvert et al., 2019). This benchmark consists in a series of increasingly complex levels, where an agent must execute a linguistic command by navigating a two-dimensional grid world and manipulating objects. Crucially, levels are procedurally generated to reflect different subsets of skills (e.g., PICKUP, UNLOCK, *et cetera*). This enables us to test our model in a controlled setting where performance based on learned skills can be compared with 'ground truth' skills. In particular, we focus on a similar subset of 8 levels as

Hui et al. (2020).

3.2 Experimental Setup

The neural architecture adheres to the best model reported in Hui et al. (2020), BOW_ENDPOOL_RES. It encodes the linguistic input through a Gated Recurrent Unit (GRU; Cho et al., 2014) and the visual input through a convolutional network (CNN). These two streams from different modalities are then merged into a single representation through FiLM (Perez et al., 2018). This component performs a feature-wise affine transformation of the CNN output conditioned on the GRU output.

Afterwards, a Long Short-Term Memory network (LSTM; Hochreiter and Schmidhuber, 1997), a recurrent module keeping track of the agent state trajectory, receives the multimodal representation and returns the current hidden state. This in turn is fed into two distinct MLPs, an actor and a critic (Sutton, 1984). The actor yields a distribution over actions, whereas the critic a reward baseline for the current state. In our experiments, each row of the matrix Φ corresponds to a possible parameterisation for all these components.

To determine *a priori* a skill–task allocation for the EXPERT baseline, we harness the information about the skills employed to procedurally generate each level by Chevalier-Boisvert et al. (2019, p. 6). For the SKILLED model, we set $|\mathcal{S}| = 9$ similarly to EXPERT. This allows us to compare learned skills and 'ground-truth' skills from an inventory of identical size. As a parameter-efficient implementation of Φ, we employ SFT (Ansell et al., 2022) with a sparsity of 90%. For all model variants, $\boldsymbol{\vartheta}_0$ and Φ are both initialised from a Kaiming uniform (He et al., 2015) and learnable. During training, we sample levels uniformly. The full specification of hyper-parameters is available in Appendix A.

3.3 Results

We now measure if our latent-skill model facilitates sample efficiency, which following Chevalier-Boisvert et al. (2019) is defined as the number of episodes required for an agent to reach a success rate greater than 0.99.[2] Success in turn is defined as executing an instruction in a number of steps $n < n_{\max}$, where the threshold again depends on the level complexity.

We plot our results in Figure 2. Firstly, models sharing information across tasks (either fully or mediated by skills) enjoy higher sample efficiency than assigning disjoint parameters for each task (PRIVATE), as they can borrow statistical strength from each other. Crucially, among information-sharing models, dual-speed SKILLED (where knowledge is modular) surpasses SHARED (where knowledge is entangled among tasks). Thus, considering a task as a collection of fine-grained skills that can be separated and reused is the most effective way of sharing information. Finally, results surprisingly show that learning a task–skill allocation matrix end-to-end (SKILLED) is more beneficial than leveraging the ground-truth task–skill decomposition used to create the BabyAI levels (EXPERT). This highlights the fact that different tasks might mutually benefit in ways that go beyond what is posited *a priori* by experts, and that our proposed approach can successfully uncover and exploit such task synergies.

Moreover, we run ablations to study the impact of the inductive biases and the parameter-efficient implementation. We compare the SKILLED model in the standard setup (with dual-speed learning rates and sparse skill-specific parameters) with other variants (with a single learning rate or with fully dense skill-specific parameters) in terms of sample efficiency in Table 3 in Appendix B. We find that assigning the same learning rate to every model component drastically slows convergence to an almost-perfect success rate. In addition, employing fully dense skill-specific parameters increases sample efficiency only to a limited degree (~1.9M). Thus, we can verify that parameter sparsification offers an effective trade-off between performance and space complexity.

4 Supervised Learning Experiments

4.1 Dataset

In order to measure the benefits of a modular design for systematic generalisation to new tasks, we run a second set of experiments on CrossFit (Ye et al., 2021), a benchmark including 160 diverse natural language processing tasks in English sourced from Huggingface Datasets (Lhoest et al., 2021). The tasks in CrossFit are all converted into a unified text-to-text format inspired by Raffel et al. (2020). Moreover, they are partitioned into three disjoint subsets. First, a model is pre-trained in a multitask fashion on training tasks $\mathcal{T}_{train}$. Afterwards, it is adapted to each evaluation task from $\mathcal{T}_{eval}$ in a few-shot learning setting. Hyper-parameter are tuned

---

[2] Success rate alone is insufficient as a performance metric as most models can solve all the levels if given enough time.

TASK	Metric	Ye et al. (2021)	Shared	Private	Expert	Hyper-Former	Task-MoE	Skilled
AG-NEWS	C–F1	84.6 ± 1.4	59.6 ± 21.1	74.7 ± 10.2	47.0 ± 9.9	64.6 ± 13.4	79.2 ± 13.1	81.2 ± 8.0
AI2-ARC	Acc	22.8 ± 1.9	23.7 ± 2.4	20.3 ± 0.8	28.3 ± 3.7	23.8 ± 6.7	19.6 ± 5.6	22.3 ± 3.4
AMAZON-POLARITY	C–F1	92.2 ± 0.6	92.4 ± 2.8	94.4 ± 0.4	57.4 ± 3.1	93.8 ± 1.5	87.3 ± 14.0	93.7 ± 0.9
BSN-NPI-LICENSOR	Acc	99.9 ± 0.2	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.2
BSN-NPI-SCOPE	Acc	85.7 ± 13.0	99.8 ± 0.3	99.6 ± 0.9	99.9 ± 0.2	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.2
BREAK-QDMR	EM	4.8 ± 0.4	4.1 ± 0.5	1.9 ± 1.7	4.1 ± 0.9	3.8 ± 1.2	4.2 ± 0.8	4.9 ± 0.6
CIRCA	C–F1	42.3 ± 7.8	44.3 ± 7.5	22.2 ± 6.3	13.0 ± 7.0	25.0 ± 6.3	39.9 ± 9.3	45.9 ± 5.7
CRAWL-DOMAIN	EM	20.7 ± 2.0	40.9 ± 2.2	36.2 ± 6.3	37.1 ± 5.3	34.9 ± 3.8	44.5 ± 2.7	39.0 ± 4.2
ETHOS-DISABILITY	C–F1	75.3 ± 2.2	66.9 ± 11.8	64.7 ± 12.1	59.2 ± 15.8	79.4 ± 3.9	66.8 ± 5.2	72.2 ± 5.2
ETHOS-SEXUAL	C–F1	59.7 ± 5.7	62.4 ± 8.4	71.2 ± 9.8	40.3 ± 11.4	76.8 ± 10.0	76.7 ± 14.4	86.1 ± 2.6
FREEBASE-QA	EM	1.3 ± 0.1	0.7 ± 0.2	0.2 ± 0.1	1.3 ± 0.5	1.6 ± 0.8	0.7 ± 0.6	0.7 ± 0.3
GLUE-COLA	M-Corr	3.5 ± 6.7	12.9 ± 5.5	9.1 ± 6.3	7.6 ± 5.6	6.8 ± 4.2	10.8 ± 10.7	7.1 ± 5.3
GLUE-QNLI	Acc	74.7 ± 2.9	75.5 ± 3.6	57.1 ± 7.7	56.6 ± 19.7	73.9 ± 3.2	75.0 ± 3.3	78.1 ± 1.6
HATEXPLAIN	C–F1	44.9 ± 2.5	33.1 ± 8.2	26.5 ± 7.8	11.9 ± 4.0	23.2 ± 6.2	41.9 ± 4.2	32.6 ± 13.6
QUOREF	QA-F1	41.2 ± 1.6	46.0 ± 4.4	36.3 ± 4.6	48.4 ± 4.3	41.7 ± 6.5	44.3 ± 3.1	47.3 ± 3.5
RACE-HIGH	Acc	30.5 ± 1.5	34.0 ± 2.7	28.5 ± 1.4	38.5 ± 2.0	30.8 ± 1.9	30.2 ± 3.2	34.8 ± 2.0
SUPERGLUE-RTE	Acc	60.4 ± 3.6	60.6 ± 2.9	49.7 ± 5.1	51.7 ± 4.8	60.9 ± 3.8	59.5 ± 5.2	60.4 ± 5.9
TWEET-EVAL-IRONY	C–F1	55.2 ± 3.6	52.1 ± 8.0	50.1 ± 14.2	25.6 ± 8.9	38.4 ± 6.0	51.8 ± 12.6	57.2 ± 2.4
WIKI-SPLIT	Rouge-L	79.3 ± 0.5	80.1 ± 0.6	80.3 ± 0.6	79.2 ± 0.8	79.2 ± 0.7	80.0 ± 0.7	80.6 ± 0.3
YELP-POLARITY	C–F1	71.8 ± 21.1	88.3 ± 14.9	65.0 ± 20.5	53.9 ± 12.7	95.0 ± 0.9	76.2 ± 17.6	94.5 ± 1.1
ALL	-	52.5 ± 1.5	53.9 ± 1.7	49.4 ± 1.6	43.0 ± 2.2	52.7 ± 1.0	54.9 ± 1.4	**56.9** ± 1.2

Table 1: Few-shot adaptation results of SKILLED and five baselines, including the original model from Ye et al. (2021), on 20 evaluation tasks of CrossFit (Ye et al., 2021). For the full name of the metrics, refer to Section 4.1.

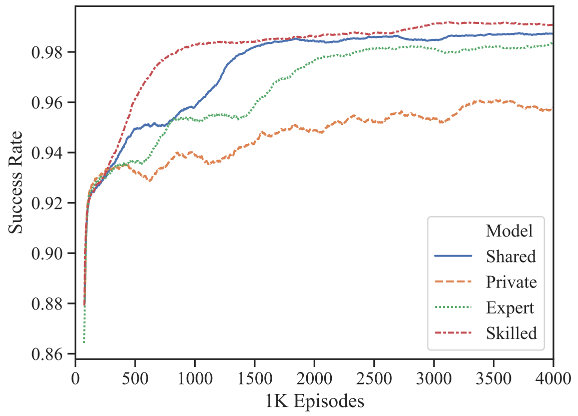


Figure 2: Sample efficiency (success rate vs. number of episodes) for different multitask models across 8 levels of BabyAI. Moving average with window of 100.

on the held-out set $\mathcal{T}_{dev}$.

We adopt the partition 1 (called RANDOM) from Ye et al. (2021), where $|\mathcal{T}_{train}| = 120$, $|\mathcal{T}_{dev}| = |\mathcal{T}_{eval}| = 20$, and tasks are split randomly. This is the most comprehensive partition and most suited for general-purpose models, as it includes all types of tasks. Every task is associated with 5 different few-shot data splits for train $\mathcal{D}_{train}$ and development $\mathcal{D}_{dev}$ and 1 larger data split for evaluation $\mathcal{D}_{eval}$. During multitask pre-training, we concatenate all $\mathcal{D}_{train}$ and $\mathcal{D}_{dev}$ from $\mathcal{T}_{train}$. During few-shot adaptation, for each task in $\mathcal{T}_{eval}$ we use each $\mathcal{D}_{train}$ and $\mathcal{D}_{dev}$ separately in 5 distinct runs and evaluate on $\mathcal{D}_{eval}$. We measure the performance of a model with 7 evaluation metrics according to the type of task: C[lassification]-F1, Acc[uracy], QA-F1, E[xact] M[atch], Rouge-L, M[atthew]-Corr[elation], and P[earson]-Corr[elation].

4.2 Experimental Setup

As pre-trained weights ϑ_0 and architecture for conditional text generation, we choose BART Large (Lewis et al., 2020), a 24-layer Transformer-based encoder–decoder. We use LoRA (Hu et al., 2021) to implement skill-specific parameters efficiently, as it was explicitly designed for the Transformer architecture (cf. Section 2.4). While ϑ_0 remains frozen, A matrices in Φ are initialised to zero matrices following Hu et al. (2021).

As a source of expert knowledge for the EXPERT baseline, we associate each task to a unique skill corresponding to one of the 4 task types of Ye et al. (2021, p. 20)'s taxonomy: question answering, conditional text generation, classification, or other (e.g., regression). The skill inventory size of 8 for SKILLED was chosen among $\{2, 4, 8, 16, 32\}$ based on validation. We set the embedding size e and hidden size h of HYPER to an identical value to ensure a fair comparison. Both SKILLED and HYPER therefore increase the parameter count by $\sim 0.78\%$ per skill and task embedding dimension, respectively. We provide more information on the hyper-parameter configuration in Appendix A.

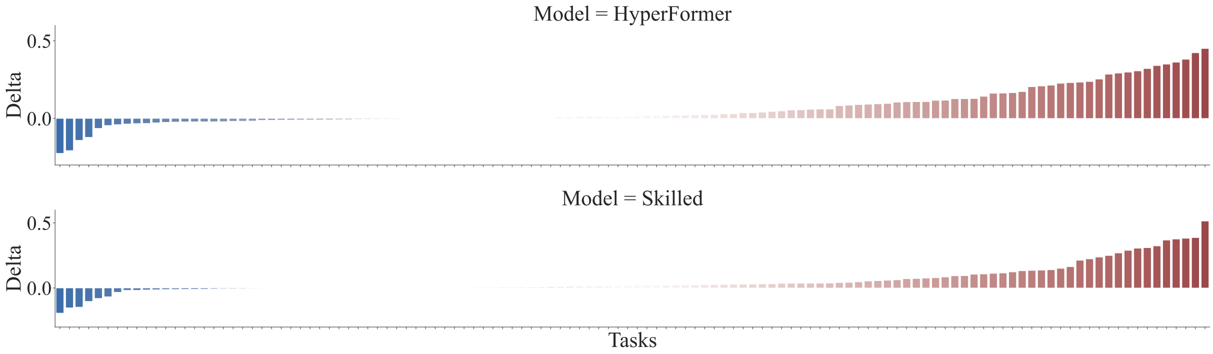


Figure 3: Delta in performance in terms of task-specific metrics between SKILLED (bottom) and HYPER (top) on the one hand and SHARED on the other, across 120 CrossFit tasks seen during multitask pre-training.

4.3 Results

Few-shot Adaptation to New Tasks In the few-shot adaptation setting, our goal is to evaluate the capability of the model to quickly generalise to new tasks unseen during training. This is the most realistic setting as tasks encountered by models deployed 'in the wild' will be characterised by different distributions or involve different input / output spaces. Performance in terms of task-specific metrics is reported in Table 1 for the 20 evaluation tasks individually and on average.

Crucially, results show that SKILLED outperforms alternative formulations of the task–skill allocation matrix, such as SHARED, PRIVATE and EXPERT. Importantly, we note that SKILLED also surpasses HYPER by a sizeable margin despite the two models having comparable parameter counts. This points to the fact that explicitly modularising knowledge learnt during multitask training is important for systematic adaptation to unseen tasks, whereas entangled knowledge is more brittle to distribution shifts in cross-task transfer. Similarly, the average performance of SKILLED is significantly higher that TASK-MOE, which illustrates the importance of variable-size module allocation. Also, it demonstrates that modularity holds promise not only for scaling large language models, but also as a strategy to favour generalisation. Finally, we corroborate the soundness of our experiment setup by reproducing the results of Ye et al. (2021).[3]

Multitask Evaluation on Seen Tasks Moreover, to ensure that modularity does not adversely affect in-domain performance, we evaluate models on the test sets of seen tasks after multitask training. In Figure 3, we report the delta in performance in terms of task-specific metrics between two models (SKILLED on top, HYPER on the bottom) and a baseline (SHARED) for the 120 training tasks. Both models yield positive gains for most tasks over SHARED. Most importantly, SKILLED achieves a performance (48.95 on average) comparable to HYPER (49.37 on average), therefore confirming that explicit modularisation can be as effective as conditional parameter generation when evaluated on seen tasks, but also engenders vast improvements on held-out tasks.

In-Depth Analysis of Learned Skills Finally, we run an in-depth analysis of the task–skill allocation matrices Z learned by SKILLED. We visualise its posterior for $|\mathcal{S}| = 4$ in Figure 5. Specifically, we measure:

1. *Discreteness*. How close is the continuous relaxation to a binary matrix? To this end, we report the average normalised entropy across all probabilities in the cells of the matrix:

$$\text{Discrete}(Z) = \frac{1}{|\mathcal{T}| \cdot |\mathcal{S}|} \sum_{\mathcal{T}_i} \sum_{\mathcal{S}_j} \frac{\mathcal{H}(z_{ij})}{\log 2} \quad (5)$$

   For instance, a binary matrix yields a score of 1, a matrix where every cell is 0.5 yields a score of 0.

2. *Sparsity*. How many skills are active per task on average? We count the rate of non-zero cells in the values rounded to the closest integer:

$$\text{Sparsity}(Z) = \frac{1}{|\mathcal{T}| \cdot |\mathcal{S}|} \sum_{\mathcal{T}_i} \sum_{\mathcal{S}_j} \lfloor z_{ij} \rceil \quad (6)$$

3. *Usage*. Is the allocation of skills across tasks balanced or are some preferred over others? We

---

[3] Note that in Ye et al. (2021) the pre-trained model is BART Small and all parameters are fine-tuned. Hence, these results are not directly comparable.

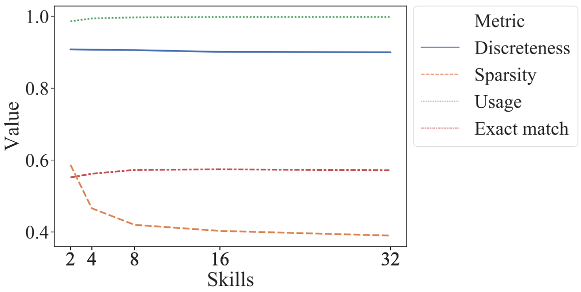


Figure 4: Statistics of the task–skill matrices for different choices of skill inventory size, including: discreteness, sparsity, usage, and the average exact match on the development set of 120 CrossFit tasks.

provide the normalised entropy of a categorical distribution parameterised by $\sum_j Z_{\star,j}$, the sum of the columns of Z:

$$\text{Usage}(Z) = \frac{\mathcal{H}\left[\sum_{\mathcal{T}_i} z_{i,\star}\right]}{\log |\mathcal{S}|} \tag{7}$$

Note that the entropy values are normalised into the range $[0, 1]$ to make them invariant to the number of skills: this quantity is known as 'efficiency'.

We plot these metrics—as well as the performance on in-domain train tasks in terms of exact match—as a function of the skill inventory size in Figure 4. We find that, whilst a continuous relaxation, the learned matrices are highly discretised and all their values are extremely close to either 0 or 1. Moreover, the level of sparsity decreases as the number of skills increases. This means that relatively smaller subsets of skills are required. Instead, usage stays consistently near the maximum value, which implies that there is uniformity in how frequently each skill is active across tasks. Finally, exact match is consistently high for sufficiently large ($|\mathcal{S}| \geq 8$) skill inventories.

Overall, these results demonstrate that a quasi-binary, highly-sparse, and non-trivial allocation matrix can be successfully learned in an end-to-end fashion even with simple inductive biases such as a dual-speed learning rate.

Crucially, learning a skill allocation also facilitates interpreting black-box multitask models. In fact, the structure of Z corresponds to an explicit hierarchy of tasks, where simpler ones are subsumed by more complex ones, and similar tasks can be grouped into the same category if they share the same subset of skills. We plot this hierarchy as a dendrogram in Figure 6. For instance, most GLUE tasks (Wang et al., 2018) are grouped together as they are all focused on natural language understanding: for instance, they require skill 1 (COLA), 2 (MRPC, RTE, SST2, WNLI), or both 1 and 2 (MNLI, QQP).

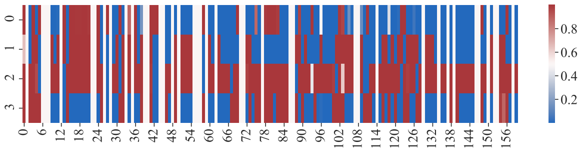


Figure 5: Posterior over Z in SKILLED for $|\mathcal{S}| = 4$.

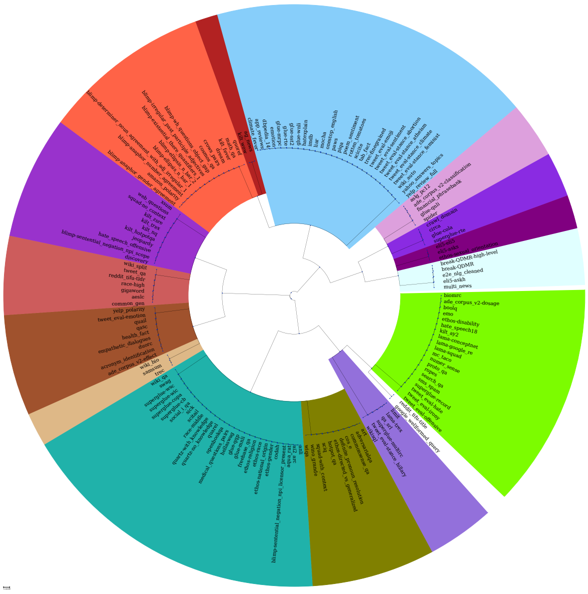


Figure 6: Task partitions for $|\mathcal{S}| = 4$, which corresponds to $2^{|\mathcal{S}|} = 16$ possible subsets of skills.

5 Related Work

Modular Networks Modularising neural networks has long been sought as a way to achieve better generalisation to unseen inputs (Jacobs et al., 1991b; Andreas et al., 2016; Kirsch et al., 2018), tasks (Jacobs et al., 1991a; Alet et al., 2018), and recently to improve continual learning (Ostapenko et al., 2021), to be more robust to changes in the environment (Goyal et al., 2021) and for sequence generation (Zhang et al., 2022).

In routing networks (Rosenbaum et al., 2019), a learnable router decides the order in which modules are applied to the input. Learning the structure of this composition has been achieved using an external parser (Andreas et al., 2016) or by sampling structures with simulated annealing (Alet et al., 2018). In mixture of experts (MoE), the system selects a soft subset of modules depending on the input to be processed (Jacobs et al., 1991b; Shazeer et al., 2017). MoEs can be interpreted invoking the theory of independent mechanisms (Parascandolo et al., 2018) that Goyal et al. (2021) further extend to handle sequential problems. Sun et al. (2020) share subsets of parameters among sparse subnetworks corresponding to different tasks. Within NLP, Fedus et al. (2021) uses a MoE architecture to scale large language model pre-training to trillions of parameters.

Our model conditions the computation on the task rather than on task inputs. Attempts to enforce parameter reuse and modularity for multitask learning include (Rajendran et al., 2017; Ponti et al., 2021a; Kingetsu et al., 2021; Kudugunta et al., 2021b). Rajendran et al. (2017) learn separate modules for each task and then learn how to reuse those modules for a new task. Kudugunta et al. (2021b) uses a set of modules for each task in a multi-lingual translation setting. Our approach does not assume a set of modules for each task but instead decomposes a task into a set of skills themselves reusable across tasks.

Multitask NLP Joint multitask learning in NLP proved an effective strategy for improving model performance in low-resource tasks and for quickly adapting to new tasks (Ruder et al., 2019; Min et al., 2021; Wei et al., 2021; Aribandi et al., 2021; Sanh et al., 2022), languages (Ponti et al., 2019), and modalities (Bugliarello et al., 2022). Liu et al. (2019) trained a shared model for all GLUE tasks and achieved impressive performance on GLUE. Pfeiffer et al. (2021) share information across task-specific parameters while alleviating negative task interference.

In our experiments we parameterise each skill with two kinds of adapters: either SFT (Ansell et al., 2022) or LoRA (Hu et al., 2021). Recently, Karimi Mahabadi et al. (2021) and Pilault et al. (2021) ensure cross-task information sharing by using a hyper-network to generate adapters from task embeddings. Differently, our task-specific parameters are composed of a set of skills from a shared inventory, which makes our approach modular and more scalable.

Several multitask approaches specifically target adaptation to new tasks, such as meta-learning approaches (Alet et al., 2018; Rusu et al., 2019; Ponti et al., 2021b; Garcia et al., 2021; Ostapenko et al., 2021). In our paper, we efficiently achieve few-shot task adaptation by inferring the skill allocation for new tasks and fine-tuning skill parameters initialised from multitask learning.

6 Conclusions

We argued that a modular design is crucial to ensure that neural networks can learn from a few examples and generalise robustly across tasks by recombining autonomous facets of knowledge. To this end, we proposed a model where a subset of latent, discrete skills from a fixed inventory is allocated to each task in an end-to-end fashion. The task-specific instantiation of a neural network is then obtained by combining efficient parameterisations of the active skills, such as sparse or low-rank adapters. We evaluate the sample efficiency of our model on multitask instruction following through reinforcement learning and its few-shot adaptability on multitask text-to-text generation through supervised learning. In both experiments, we surpass competitive baselines such as conditional parameter generation (HyperFormer) and mixture of experts (Task-MoE). Finally, we show that modularity helps interpret multi-task models by inferring explicit relationships between tasks according to the skills they share.

Limitations

Firstly, the kind of knowledge captured by individual skills is not fully interpretable. In terms of task–skill allocation, while some patterns based on task type and textual domain emerge from the clusters of Figure 6, it remains unclear how to systematically probe this information. Moreover, in terms of skill-specific parameters, it is hard to measure the diversity among learned skills. In fact, due to the under-specification of neural networks, different configurations in the parameter space may in fact correspond to the same function.

Secondly, few-shot adaptation in our experiments is based on the assumption that the skill inventory is fixed. Therefore, the model is compelled to recombine and possibly fine-tune old skills. Instead, each unseen task should involve a combination of newly discovered and previously honed skills. In the future, a possible extension of our method for continuous learning should adopt this more natural assumption.

Finally, we explore only a subset of the problems where different skills can be combined. In addition to instruction following and conditional text generation, our results should be replicated also in multilingual and multimodal benchmarks (Bugliarello et al., 2022, *inter alia*).

Acknowledgements

We are grateful to the anonymous reviewers for their helpful comments. EP is supported by the IVADO Postdoctoral Research Fellowship, and SR by the NSERC Discovery Grant and the Facebook CIFAR AI Chair.

References

Ferran Alet, Tomas Lozano-Perez, and Leslie P. Kaelbling. 2018. Modular meta-learning. In *Proceedings of The 2nd Conference on Robot Learning*, pages 856–868.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48.

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2021. ExT5: Towards extreme multi-task scaling for transfer learning. *arXiv preprint arXiv:2111.10952*.

Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, et al. 2021. Efficient large scale language modeling with mixtures of experts. *arXiv preprint arXiv:2112.10684*.

Yoshua Bengio. 2017. The consciousness prior. *arXiv preprint arXiv:1709.08568*.

Emanuele Bugliarello, Fangyu Liu, Jonas Pfeiffer, Siva Reddy, Desmond Elliott, Edoardo Maria Ponti, and Ivan Vulić. 2022. IGLUE: A benchmark for transfer learning across modalities, tasks, and languages. *arXiv preprint arXiv:2201.11732*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2019. BabyAI: First steps towards grounded language learning with a human in the loop. In *International Conference on Learning Representations*.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

William Fedus, Jeff Dean, and Barret Zoph. 2022. A review of sparse expert models in deep learning. *arXiv preprint arXiv:2209.01667*.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Jezabel R Garcia, Federica Freddi, Feng-Ting Liao, Jamie McGowan, Tim Nieradzik, Da-shan Shiu, Ye Tian, and Alberto Bernacchia. 2021. Meta-learning with MAML on trees. *arXiv preprint arXiv:2103.04691*.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. 2021. Recurrent independent mechanisms. In *International Conference on Learning Representations*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

David Yu-Tung Hui, Maxime Chevalier-Boisvert, Dzmitry Bahdanau, and Yoshua Bengio. 2020. BabyAI 1.1. *arXiv preprint arXiv:2007.12770*.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Robert A Jacobs, Michael I Jordan, and Andrew G Barto. 1991a. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive science*, 15(2):219–250.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991b. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations*.

Daniel Kahneman. 2011. *Thinking, fast and slow*. Macmillan.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for Transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 565–576.

Hiroaki Kingetsu, Kenichi Kobayashi, and Taiji Suzuki. 2021. Neural network module decomposition and recomposition. *arXiv preprint arXiv:2112.13208*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Louis Kirsch, Julius Kunze, and David Barber. 2018. Modular networks: Learning to decompose neural computation. *Advances in Neural Information Processing Systems*, 31.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021a. Beyond distillation: Task-level mixture-of-experts for efficient inference. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3577–3599, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021b. Beyond distillation: Task-level mixture-of-experts for efficient inference. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3577–3599.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184.

Haokun Liu, Derek Tam, Muqeeth Mohammed, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the International Conference on Learning Representations 2017*.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. MetaICL: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

Oleksiy Ostapenko, Pau Rodriguez, Massimo Caccia, and Laurent Charlin. 2021. Continual learning via local module composition. *Advances in Neural Information Processing Systems*, 34.

Giambattista Parascandolo, Niki Kilbertus, Mateo Rojas-Carulla, and Bernhard Schölkopf. 2018. Learning independent causal mechanisms. In *International Conference on Machine Learning*, pages 4036–4044.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 487–503.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673.

Jonathan Pilault, Amine El hattami, and Christopher Pal. 2021. Conditionally adaptive multi-task learning: Improving transfer learning in NLP using fewer parameters & less data. In *International Conference on Learning Representations*.

Edoardo Ponti. 2021. *Inductive Bias and Modular Design for Sample-Efficient Neural Language Learning*. Ph.D. thesis, University of Cambridge.

Edoardo M Ponti, Ivan Vulić, Ryan Cotterell, Marinela Parovic, Roi Reichart, and Anna Korhonen. 2021a. Parameter space factorization for zero-shot learning across tasks and languages. *Transactions of the Association for Computational Linguistics*.

Edoardo Maria Ponti, Rahul Aralikatte, Disha Shrivastava, Siva Reddy, and Anders Søgaard. 2021b. Minimax and Neyman–Pearson meta-learning for outlier languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1245–1260.

Edoardo Maria Ponti, Helen O'Horan, Yevgeni Berzak, Ivan Vulić, Roi Reichart, Thierry Poibeau, Ekaterina Shutova, and Anna Korhonen. 2019. Modeling language variation and universals: A survey on typological linguistics for natural language processing. *Computational Linguistics*, 45(3):559–601.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Janarthanan Rajendran, P Prasanna, Balaraman Ravindran, and Mitesh M Khapra. 2017. Attend, adapt and transfer: Attentive deep architecture for adaptive transfer from multiple sources in the same domain. In *International Conference on Learning Representations*.

Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. 2019. Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2019. Latent multi-task architecture learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4822–4829.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. 2019. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M. Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5986–5995.

Tianxiang Sun, Yunfan Shao, Xiaonan Li, Pengfei Liu, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Learning sparse sharing architectures for multiple tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8936–8943.

Richard Stuart Sutton. 1984. *Temporal credit assignment in reinforcement learning*. Ph.D. thesis, University of Massachusetts Amherst.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2021. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Paul J Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. 2017. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Advances in neural information processing systems*, 30.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. CrossFit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189.

Yanzhe Zhang, Xuezhi Wang, and Diyi Yang. 2022. Continual sequence generation with adaptive compositional modules. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3653–3667. Association for Computational Linguistics.

A Hyper-parameters

A.1 BabyAI

We follow closely the best hyper-parameter setup of Hui et al. (2020). Tiles in the visual input are encoded into embeddings of size 128 via a look-up table. The CNN has 2 layers, filter size 3, stride 1, and padding 1; whereas the FiLM module has 2 layers. A residual layer is added between the CNN output and each of the FiLM layers. The output of FiLM is max-pooled with a layer of size 7 and stride 2. Both the LSTM and the GRU have a hidden size of 128.

A learning rate of 1e-4 is adopted for Adam (Kingma and Ba, 2015). We optimise the model with Proximal Policy Optimisation (PPO; Schulman et al., 2017) and Back-Propagation Through Time (BPTT; Werbos, 1990). Additionally, we use an Advantage Actor–Critic (A2C; Wu et al., 2017) with Generalised Advantage Estimation (GAE; Schulman et al., 2015). The reward is calculated as $(1 - 0.9\, n/n_{\max})$ if the agent completes a task—where n is the number of steps required and $n_{\max}$ is a threshold set according to the level difficulty, 0 otherwise. Returns are discounted by $\gamma = 0.99$.

A.2 CrossFit

During both multitask pre-training and few-shot adaptation, we use the Adam optimiser (Kingma and Ba, 2015) and select a learning rate for Φ among $\{1e-2, 1e-3, 1e-4\}$ based on performance on the development sets of $\mathcal{T}_{train}$ and $\mathcal{T}_{dev}$, respectively. As a more aggressive learning rate for Z in Skilled instead we search in the range $[1e-1, 1e-2]$. We run multitask pre-training for 30 epochs with an effective batch size of 32, with a warm-up of 6% of the total steps. Instead, during few-shot adaptation, the effective batch size is 8 for 1000 training steps, with a warm-up rate of 10% and a weight decay of 1e-2. For each held-out task, a newly added row in Z (initialised with a vector of 0.5) and Φ (initialised from pre-training) are fine-tuned, but not ϑ_0.

For a comparison of parameter counts, LoRA adds $4l(2hr + |\mathcal{T}|) \cdot |\mathcal{S}|$ parameters to the pre-trained model, where l is the number of layers in the encoder and decoder, h is the hidden size, and 4 is the number of linear projections in self-attention (query, key, value, output). We use $r = 16$, $l = 24$ and $h = 1024$ for BART Large, so we add $\sim 3 \cdot 10^6$ parameters per skill. Given that the pre-trained model has $\sim 4 \cdot 10^8$ parameters, this implies an increase of $\sim 0.78\%$ per skill. Hyper adds $4l(2he + 2e) \cdot e$ parameters, an increase of $\sim 0.78\%$ per task embedding dimension.

B Additional Results for BabyAI

Skills	Level
1 2 3 4 8	GoTo
1 8	GoToObjMaze
1 2 3 6 7	PickupLoc
1 2 3 5	PutNextLocal
1 2 3 4	GoToLocal
1 2 3	GoToRedBall
1 2	GoToRedBallGrey
1	GoToObj

Table 2: BabiAI Expert task–skill allocation.

Model	Episodes
Private	>6000000
Shared	3544294
Expert	4608019
Skilled	**2218226**
- Sparsity	1853060

Table 3: Sample efficiency of various models on 8 BabyAI levels measured as the number of episodes needed to reach a success rate > 0.99.

C Additional Results for CrossFit

Metric	Shared	Hyper	Skilled
	Task-specific		
Acc	58.47	62.63	62.66
C-F1	41.76	56.74	55.39
EM	20.05	21.68	21.70
P-Corr	56.83	61.54	52.60
QA-F1	48,88	51.04	52.35
Rouge-L	28.02	26.71	28.05
	Global		
Average	43.09	49.37	48.95

Table 4: Performance of multitask models averaged over test sets of 120 seen CrossFit task. Performance is both aggregated globally across all tasks (in terms of task-specific metrics) and across subsets of tasks with the same evaluation metric.

	ELI5-ASKS, ELI5-ELI5, ETHOS-SEXUAL-ORIENTATION
3	GOOGLE-WELLFORMED-QUERY, REDDIT-TIFU-TITLE
2	APP-REVIEWS, CLIMATE-FEVER, DBPEDIA-14, EMOTION, GLUE-MRPC, GLUE-RTE, GLUE-SST2, GLUE-WNLI, HATEXPLAIN, IMDB, LIAR, MOCHA, ONESTOP-ENGLISH, PAWS, PIQA, POEM-SENTIMENT, ROTTEN-TOMATOES, SCICITE, TAB-FACT, TREC-FINEGRAINED, TWEET-EVAL-EMOJI, TWEET-EVAL-SENTIMENT, TWEET-EVAL-STANCE-ABORTION, TWEET-EVAL-STANCE-ATHEISM, TWEET-EVAL-STANCE-CLIMATE, TWEET-EVAL-STANCE-FEMINIST, WIKI-AUTO, YAHOO-ANSWERS-TOPICS, YELP-REVIEW-FULL
2 3	ADE-CORPUS-V2-DOSAGE, BIOMRC, BOOLQ, EMO, ETHOS-DISABILITY, HATE-SPEECH18, KILT-AY2, LAMA-CONCEPTNET, LAMA-GOOGLE-RE, LAMA-SQUAD, MC-TACO, NUMER-SENSE, PROTO-QA, ROPES, SEARCH-QA, SMS-SPAM, SUPERGLUE-RECORD, TWEET-EVAL-HATE, TWEET-EVAL-IRONY, TWEET-EVAL-OFFENSIVE
1	CIRCA, CRAWL-DOMAIN, GLUE-COLA, SUPERGLUE-RTE
1 3	LAMA-TREX, LIMIT, QA-SRL, SUPERGLUE-MULTIRC, TWEET-EVAL-STANCE-HILLARY, WIKISQL
1 2	AI2-ARC, ANLI, AQUA-RAT, BLIMP-SENTENTIAL-NEGATION-NPI-LICENSOR-PRESENT, CODAH, ETHOS-GENDER, ETHOS-NATIONAL-ORIGIN, ETHOS-RACE, ETHOS-RELIGION, FREEBASE-QA, GLUE-MNLI, GLUE-QQP, HELLASWAG, MEDICAL-QUESTIONS-PAIRS, OPENBOOKQA, QUAREL, QUARTZ-NO-KNOWLEDGE, QUARTZ-WITH-KNOWLEDGE, RACE-MIDDLE, SCITAIL, SICK, SOCIAL-I-QA, SUPERGLUE-CB, SUPERGLUE-COPA, SUPERGLUE-WIC, SUPERGLUE-WSC, SWAG, WIKI-QA
1 2 3	ADVERSARIALQA, ART, COMMONSENSE-QA, COS-E, DEFINITE-PRONOUN-RESOLUTION, ETHOS-DIRECTED-VS-GENERALIZED, HOTPOT-QA, SCIQ, SQUAD-WITH-CONTEXT, WINOGRANDE, WIQA
0	BREAK-QDMR, BREAK-QDMR-HIGH-LEVEL, E2E-NLG-CLEANED, ELI5-ASKH, MULTI-NEWS
0 3	AESLC, COMMON-GEN, GIGAWORD, RACE-HIGH, REDDIT-TIFU-TLDR, TWEET-QA, WIKI-SPLIT
0 2	AG-NEWS, KILT-WOW
0 2 3	BLIMP-SENTENTIAL-NEGATION-NPI-SCOPE, DISCOVERY, HATE-SPEECH-OFFENSIVE, JEOPARDY, KILT-HOTPOTQA, KILT-NQ, KILT-TREX, KILT-ZSRE, SQUAD-NO-CONTEXT, WEB-QUESTIONS, XSUM
0 1	ADE-CORPUS-V2-CLASSIFICATION, ASLG-PC12, FINANCIAL-PHRASEBANK, GLUE-QNLI, SPIDER
0 1 3	SAMSUM, TREC, WIKI-BIO
0 1 2	AMAZON-POLARITY, BLIMP-ANAPHOR-GENDER-AGREEMENT, BLIMP-ANAPHOR-NUMBER-AGREEMENT, BLIMP-DETERMINER-NOUN-AGREEMENT-WITH-ADJ-IRREGULAR-1, BLIMP-ELLIPSIS-N-BAR-1, BLIMP-ELLIPSIS-N-BAR-2, BLIMP-EXISTENTIAL-THERE-QUANTIFIERS-1, BLIMP-IRREGULAR-PAST-PARTICIPLE-ADJECTIVES, BLIMP-WH-QUESTIONS-OBJECT-GAP, COSMOS-QA, CROWS-PAIRS, DREAM, KILT-FEVER, MATH-QA, QUOREF
0 1 2 3	ACRONYM-IDENTIFICATION, ADE-CORPUS-V2-EFFECT, DUORC, EMPATHETIC-DIALOGUES, HEALTH-FACT, QASC, QUAIL, TWEET-EVAL-EMOTION, YELP-POLARITY

Table 5: Skill allocation to 120 training CrossFit tasks for $|\mathcal{S}| = 4$.

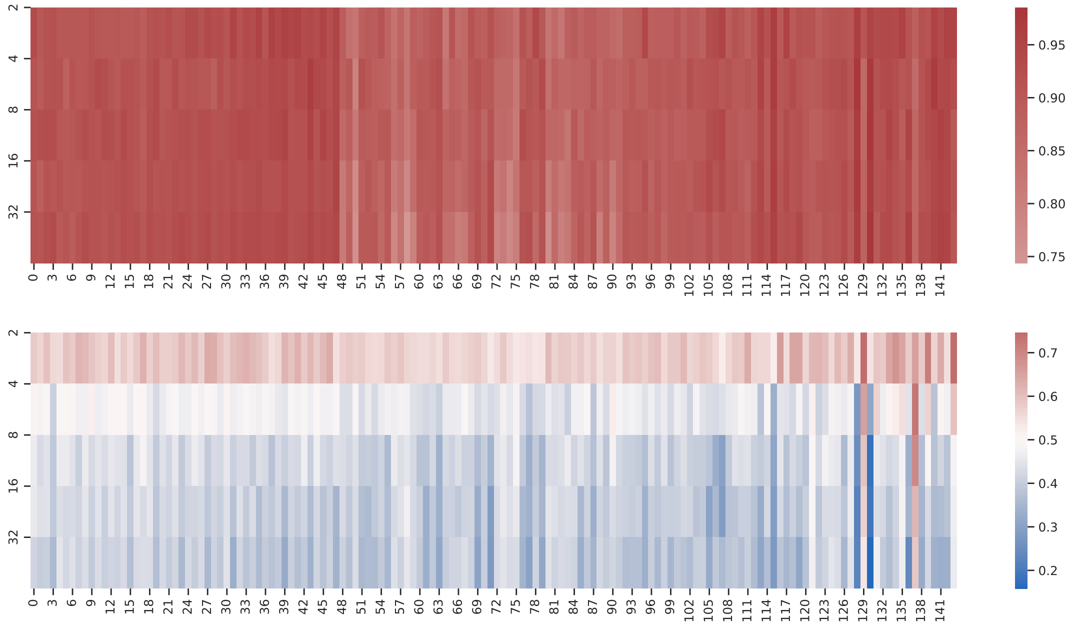


Figure 7: Per-layer discreteness (top) and per-layer sparsity (bottom).

Self-imitation Learning for Action Generation in Text-based Games

Zijing Shi[1], Yunqiu Xu[1], Meng Fang[2], Ling Chen[1]
[1] University of Technology Sydney, NSW, Australia
zijing.shi@student.uts.edu.au, {yunqiu.xu,ling.chen}@uts.edu.au
[2] University of Liverpool, Liverpool, UK
Meng.Fang@liverpool.ac.uk

Abstract

In this work, we study reinforcement learning (RL) in solving text-based games. We address the challenge of combinatorial action space, by proposing a confidence-based self-imitation model to generate action candidates for the RL agent. Firstly, we leverage the self-imitation learning to rank and exploit past valuable trajectories to adapt a pre-trained language model (LM) towards a target game. Then, we devise a confidence-based strategy to measure the LM's confidence with respect to a state, thus adaptively pruning the generated actions to yield a more compact set of action candidates. In multiple challenging games, our model demonstrates promising performance in comparison to the baselines.

1 Introduction

Text-based games are situated systems where the game agents observe textual descriptions, and generate textual commands to interact with the environment. These games have proven to be suitable test-beds for studying various natural language processing (NLP) tasks, such as question answering (Yuan et al., 2019; Xu et al., 2022), dialogue systems (Ammanabrolu et al., 2022), situated language learning (Shridhar et al., 2020) and commonsense reasoning (Murugesan et al., 2021; Ryu et al., 2022). Recent years have witnessed the thriveness of designing reinforcement learning (RL) agents in solving these games (Narasimhan et al., 2015; Hausknecht et al., 2020; Ammanabrolu and Riedl, 2019; Xu et al., 2020b), while the combinatorial action space remains as a challenging issue, preventing RL agents from being deployed in real world applications.

In general, text-based games accept free-form actions, resulting in a large combinatorial action space. Fig. 1 shows a raw excerpt from the classic game "Zork1". A 4-word action has to be selected from $|\mathcal{V}|^4$ candidates, where $\mathcal{V}$ denotes the vocabulary set (Xu et al., 2020a). Given that only 130 actions are required to solve this game, the agent wastes both training data and time in attempting irrelevant actions (Dulac-Arnold et al., 2015). To handle the combinatorial action space, early efforts either heavily rely on hand-crafted rules, or simply assume the availability of the action candidate set. For example, some works consider a set of currently admissible actions (He et al., 2016), or a template-based action space (Hausknecht et al., 2020). Alternatively, some other works alleviated this challenge by filtering inadmissible actions through methods such as action affordance (Jain et al., 2020), bandit-based elimination (Zahavy et al., 2018) and rule-based scoring (Ammanabrolu and Riedl, 2019).

Observation: The door reluctantly opens to reveal a rickety staircase descending into darkness.

Action: go down

Observation: The trap door crashes shut, and you hear someone barring it.

Action: go north

Observation: This is a small room with passages to the east and south and a forbidding hole leading west. A nasty-looking troll, brandishing a bloody axe, blocks all passages out of the room. Your sword has begun to glow very brightly.

Action: kill troll with sword

Observation: Almost as soon as the troll breathes his last breath, a cloud of sinister black fog envelops him, and when the fog lifts, the carcass has disappeared. Your sword is no longer glowing.

Figure 1: Excerpt from the game "Zork1". With a vocabulary size of 697, there are around $697^4 \approx 200$ billion potential 4-word actions in the game.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 703–726
May 2-6, 2023

In handling the combinatorial action space for text-based games, recent pre-trained language models (PLMs) (Devlin et al., 2019; Radford et al., 2019; Brown et al., 2020; Andreas and Klein, 2016) can help generate actions. However, the potential of LM is still less effectively explored. As one of the pioneer works, Yao et al. (2020) proposed the CALM, which is a GPT-2 model pre-trained on human gameplay trajectories, to generate the action candidate set for the RL agent. However, when solving a previously unseen game, CALM tends to generate actions with less satisfying qualities, leading to two consequences that may affect RL training: 1) the action set may contain a large proportion of inadmissible actions, and 2) the useful actions may not be generated. As a mitigation, the CALM model is set to generate a relatively huge action candidate set, followed by ad-hoc operations to filter out the inadmissible actions, which requires prior knowledge. Micheli and Fleuret (2021) extended the LM-based agent to goal-conditioned tasks to follow instructions. Besides the offline pre-training data, the LM is further improved with the successful trajectories collected during online interaction. However, text-based games do not have well-defined goals. Furthermore, some games are so challenging that it is impossible to collect successful trajectories (Tuyls et al., 2022).

In this work, we address the crux of combinatorial action space in solving text-based games. We propose the Confidence-based Self-imitation Model (CSM) to generate the action candidates for the RL agent.[*] Firstly, we leverage the self-imitation learning method (Oh et al., 2018) to rank and exploit past trajectories of high values to adapt a pre-trained LM towards the target game. Then, we propose a confidence-based strategy to measure the LM's confidence (Gandrabur et al., 2006) with respect to a state, thus adaptively pruning the action candidates based on the confidence value. Our model achieves promising performance in six challenging man-made games. Apart from significantly outperforming an action generation-based baseline, our strategy helps the RL agent to even achieve comparable performance to a baseline armed with the oracle action candidate set.

Our main contributions are summarized as follows: Firstly, we develop a LM-based framework to handle the issue of combinatorial action space in solving text-based games. Secondly, we propose a strategy to further improve the LM via self-imitation learning during the RL training. Thirdly, our experiments demonstrate that, the proposed method significantly improve the performance on multiple games compared with the strong contemporary method.

2 Related Work

2.1 RL Agents for Text-based Games

Inspired by the success of RL in playing games (Silver et al., 2016) and various NLP tasks (Fang et al., 2017; Yuan et al., 2019; Ammanabrolu et al., 2022), Narasimhan et al. (2015) and He et al. (2016) introduce RL to solve text-based games. Compared with non-learning-based agents (Hausknecht et al., 2019; Atkinson et al., 2019), the RL-based agents reduce the demand for extensive expert knowledge to develop gameplay strategies, and become the predominant modelling paradigm for solving text-based games. Subsequently, many variants of RL-based agents with different architectures and learning schemes have been proposed (Yuan et al., 2018; Jain et al., 2020; Guo et al., 2020; Xu et al., 2021; Tuyls et al., 2022; Shi et al., 2023). Innovations include modeling state space utilising knowledge graphs (Adhikari et al., 2020; Xu et al., 2020b), integrating question-answering and reading comprehension modules (Ammanabrolu et al., 2020; Xu et al., 2022). While these approaches focus on the problems of partial observability and language semantics, they still face the challenge of the combinatorial action space.

2.2 Combinatorial Action Space in TBGs

The combinatorial language-based action space is one primary challenge in solving text-based games. Early efforts mainly utilise hand-crafted rules or assume the agent has a predefined set of actions to choose from. For instance, the Jericho benchmark provides a valid action handicap that filters out inadmissible actions (i.e. actions that are either unrecognized by the game engine or do not change the underlying game state) at each game state (Hausknecht et al., 2020). This handicap has been widely used as the reduced action space by approaches like DRRN (He et al., 2016). In addition, the template-based action space is introduced where the agent selects first a template, and then a verb-object pair either individually (Hausknecht et al., 2020) or conditioned on the selected template (Ammanabrolu and Hausknecht, 2020). Even

---

[*] Code is available at https://github.com/winni18/CSM.

using the reduced action space, approaches filtering unnecessary actions can further improve the computational tractability and speed up the learning convergence (Zahavy et al., 2018; Jain et al., 2020).

2.3 Pre-training Methods for TBGs

Recent studies focus on enhancing the language understanding capability of agents by introducing pre-trained language processing modules. For instance, Singh et al. (2021) utilise the DistilBERT (Sanh et al., 2019) fine-tuned on human gameplay trajectories to represent game states. Ammanabrolu et al. (2020) employ the pre-trained ALBERT (Lan et al., 2019) to extract information from the textual observation by answering questions, and then update the knowledge graph during training. Adolphs and Hofmann (2020) use a pre-trained task-specific module to predict what is left to complete the tasks. In general, RL-based agents are initialised with knowledge using pre-trained modules before exploring game environments.

Some studies leverage pre-trained language models for action generation (Hausknecht et al., 2020) or word embeddings for affordance detection (Fulda et al., 2017). The approach closest to our work is Yao et al. (2020), which is state-of-the-art without requiring access to admissible actions. In their study, a GPT-2 language model trained on human gameplay trajectories is used to generate action candidates for the RL agent to select. To ensure that the correct actions are provided, the GPT-2 model is set to generate a relatively huge action candidate set, followed by ad-hoc operations to predict the admissibility of an action based on environmental feedback. In contrast, our work intends to narrow down the action space via self-imitation learning and make learning tractable.

3 Preliminaries

Text-based Games as POMDPs The text-based game can be formally formulated as a partially observable Markov Decision Process (POMDP) $(\mathcal{S}, T, \mathcal{A}, \mathcal{O}, R, \gamma)$. At each step t, the agent receives a textual observation $o_t \in \mathcal{O}$ from the game environment, while the latent state $s_t \in S$, which contains the complete internal information of the environment, could not be observed. By executing an action $a_t \in \mathcal{A}$, the environment will transit to the next state according to the latent transition function T, and the agent will receive the reward signal $r_t = R(s_t, a_t)$ and the next observation o_{t+1}. The objective of the agent is to take actions to maximize the expected cumulative discounted rewards $R_t = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$, where $\gamma \in [0, 1]$ is the discount factor.

Trajectory and Episode We define the trajectory τ as the sequence of observation-action pairs collected in an RL episode, i.e., $\tau = (o_1, a_1, o_2, a_2, \ldots, o_l, a_l)$, where l_τ is the length of τ. An RL episode is the process of an agent interacting with the environment from the beginning of a game to a termination state (e.g., the agent dies) or the step exceeding the pre-defined limit.

DRRN Existing RL methods for solving text-based games use game rewards to learn a value function. For instance, the Deep Reinforcement Relevance Network (DRRN) (He et al., 2016) is a choice-based game agent, where each action candidate a is paired with the state o to check its relevance. The agent then passes each pair through a deep neural network with parameters ϕ to estimate the Q-values $Q_\phi(o, a)$. The parameters ϕ of DRRN are trained using tuples (o, a, r, o') sampled from a prioritized experience replay buffer with the temporal difference (TD) loss:

$$\mathcal{L}_{\mathrm{TD}}(\phi) = \left(r + \gamma \max_{a' \in A} Q_\phi\left(o', a'\right) - Q_\phi(o, a)\right)^2 \tag{1}$$

where r is the game reward and γ is the discount factor. The next action is then selected by softmax sampling the predicted Q-values:

$$\pi_\phi(a|o) = \frac{\exp\left(Q_\phi(o, a)\right)}{\sum_{a' \in A} \exp\left(Q_\phi\left(o, a'\right)\right)} \tag{2}$$

To circumvent the challenge of combinatorial action space, DRRN assumes access to the valid action handicap provided by the environment at each game state.

4 Methodology

4.1 Overview

To address the combinatorial action space, we propose the Confidence-based Self-imitation Model (CSM), which leverages the advantages of pre-trained LM and Self-imitation Learning (SiL) for adaptive action generation. Fig.2 shows an overview of CSM. At time step t, the LM is provided with the context $c_t = (o_{t-1}, a_{t-1}, o_t)$ as the input, and generates a set of action candidates $\mathcal{A}_t$

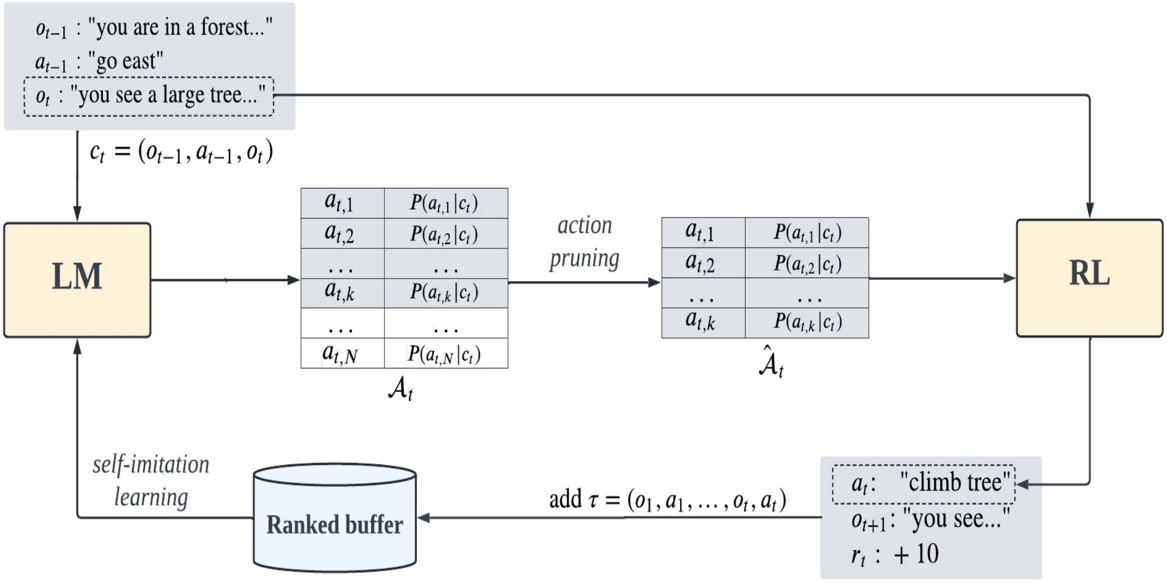


Figure 2: An overview of CSM. The LM takes the context c_t and generates action candidates $\mathcal{A}_t$, and conducts action pruning to further reduce the action space. The RL agent takes the observation o_t, and selects an action $a_t \in \hat{\mathcal{A}}_t$. The valuable trajectories τ are collected to further improve the LM through self-imitation learning.

as well as their probabilities using beam search decoding. Based on the probabilities, we conduct Action Pruning (AP) to obtain a more compact subset of action candidates $\hat{\mathcal{A}}_t \subseteq \mathcal{A}_t$ for the RL agent. Then the RL agent considers the observation o_t and selects an action $a_t \in \hat{\mathcal{A}}_t$. To generate high-quality actions which are more context-relevant, we adapt the LM towards the target game during the RL training. Specifically, we collect and then select the past valuable trajectories τ in an additional replay buffer, to further improve the LM through a self-imitation learning manner.

4.2 Self-imitation Learning

We follow the work of Yao et al. (2020) to utilize the LM for action generation. During pre-training, given human gameplay trajectories τ, we first build the context c_t, then train the LM to minimize the expected cross-entropy loss: $\mathcal{L}_{\text{LM}} = -\mathbb{E}[\log p(a|c)]$, where $\log p(a|c) = \sum_{i=1}^{m} p(a^i|a^{<i}, c)$ for an action with m tokens. During RL, the LM will serve as a “rough” action selector to generate the top-k actions. Then the RL agent will select one action to interact with the environment.

One drawback of the previous work Yao et al. (2020) is that when facing an unseen context, the LM may generate actions with poor performance. A straightforward solution is to continuously improve the LM during RL, thus making it adapted to the target game. Since no external trajectories (e.g., from human players) are available in the RL stage, we consider resorting to the self-imitation learning (Gangwani et al., 2019), i.e., letting the LM learn from the trajectories collected during the RL interaction. One thing we should pay attention to is the quality of the trajectories – sub-optimal trajectories may adversely affect imitation learning (Hu et al., 2019; Xu et al., 2022). Text-based games, especially games originally designed for human players, may be too challenging for agents to walk through. Thus, we cannot directly obtain successful trajectories during interacting with the environment. To alleviate this problem, we build a heap-like replay buffer to store past high-quality trajectories. We regard those obtaining higher scores with fewer steps as high-quality trajectories. Specifically, we rank trajectories within the replay buffer by their game scores (i.e., the sum of collected rewards) and lengths. In addition, we also take into account the novelty, by periodically replacing the old trajectories with new ones of equivalent qualities (e.g., the same scores and lengths).

4.3 Confidence-based Action Pruning

Through the aforementioned SiL, the LM is expected to generate a more reliable action candidate set $\mathcal{A}_t$ of size N. For each action $a_{t,i} \in \mathcal{A}_t$, we then calculate its normalized probability $P(a_{t,i}|c_t)$

Table 1: Game statistics.

Game	Avg.Action Number	Avg.Action Length	Avg.Steps Per Reward	Walkthough Length	Max Score
Balances	23.29	2.99	12	122	51
Inhumane	6.96	2.36	14	123	90
Ludicorp	14.52	2.76	4	364	150
Snacktime	5.68	2.14	8	34	50
Zork1	15.96	2.75	9	400	350
Ztuu	33.93	2.96	5	84	100

according to the beam search score. The probabilities exhibit two characteristics: 1) the long-tail phenomenon in linguistics (Zhan et al., 2021), where only a few probabilities produce lots of actions; 2) the probability distribution varies greatly under different states. Given these findings, we adopt a confidence-based strategy to further prune action candidates of low values, aiming to obtain a further reduced action candidate set $\hat{\mathcal{A}}_t \subseteq \mathcal{A}_t$. Specifically, we accumulate the probabilities of top-k action candidates as the confidence value: $Conf_t(k) = \sum_{i=1}^{k} P(a_{t,i}|c_t)$. We then conduct action pruning (i.e., constraining the action space k) by bonding the confidence value to a fixed, manually determined threshold ξ: $\hat{\mathcal{A}}_t = \{a_{t,i}|Conf_t(k) \leq \xi\}$. In this way, top-$k$ action candidates are selected adaptively. For a more "familiar" context c_t (e.g., it is similar to a context that LM has encountered before), the LM is supposed to be able to obtain correct actions from the training data, and the probability distribution will be centralised to top-ranked actions. In contrast, for an "unfamiliar" c_t, the actions' probabilities might be more uniformly-distributed. In this case, the size of action candidates (e.g., k) will be expanded to ensure a high confidence value.

5 Experiments

5.1 Experimental Setup

We conduct experiments upon six games provided by the Jericho Game Suite (Hausknecht et al., 2020). These games have diverse themes and genres, and each of them can represent a type of task. Different from those generated through pre-defined simple rules (Côté et al., 2018), the games we use are more complex, making them even challenging for human players. Some games contain nonstandard actions (e.g., the spells), which are unlikely to be understood by the language model pre-trained with commonsense knowledge. Table 1 shows the game statistics calculated from the walkthrough of each game.

5.2 Baselines

Our work focus on the challenge of combinatorial action space in text-based games. Thus, we compare CSM with two baselines:

- **CALM** (Yao et al., 2020), which is a pioneer work in LM-guided action generation.
- **DRRN** (He et al., 2016), which assumes access to the "oracle" action set (i.e., the valid action handicap provided by the environment).

Of these methods, CALM is the previous state-of-the-art model without the availability of "oracle" action sets, while the DRRN agent with "oracle" action sets can be regarded as our "upper bound".

5.3 Implementation Details

Training We implement CSM upon CALM's released code, including a pre-trained GPT-2 LM [†]. Both CSM and CALM adopt DRRN as the RL agent, except that $\mathcal{A}_t$ is obtained by LM. We set the step limit of an RL episode as 100, and train the RL agent on 8 parallel running environments for 100k steps. For each step, we train the RL agent with a batch size of 64, using an Adam optimizer with a learning rate of 1e-4. We set the first 20k steps as the warm-up phase, and start self-imitation learning as well as action pruning after this phase. For SiL, we use a trajectory buffer with a size of 50. For every 500 steps, we update the LM for 1 epoch with a batch size of 8, using an Adam optimizer with a learning rate of 2e-5. If there are no fresh trajectories as the training progresses, we conduct SiL using existing trajectories within the buffer. For AP, we use beam search decoding with a beam size of 40 to generate actions and choose

---

[†] `https://github.com/princeton-nlp/calm-textgame`

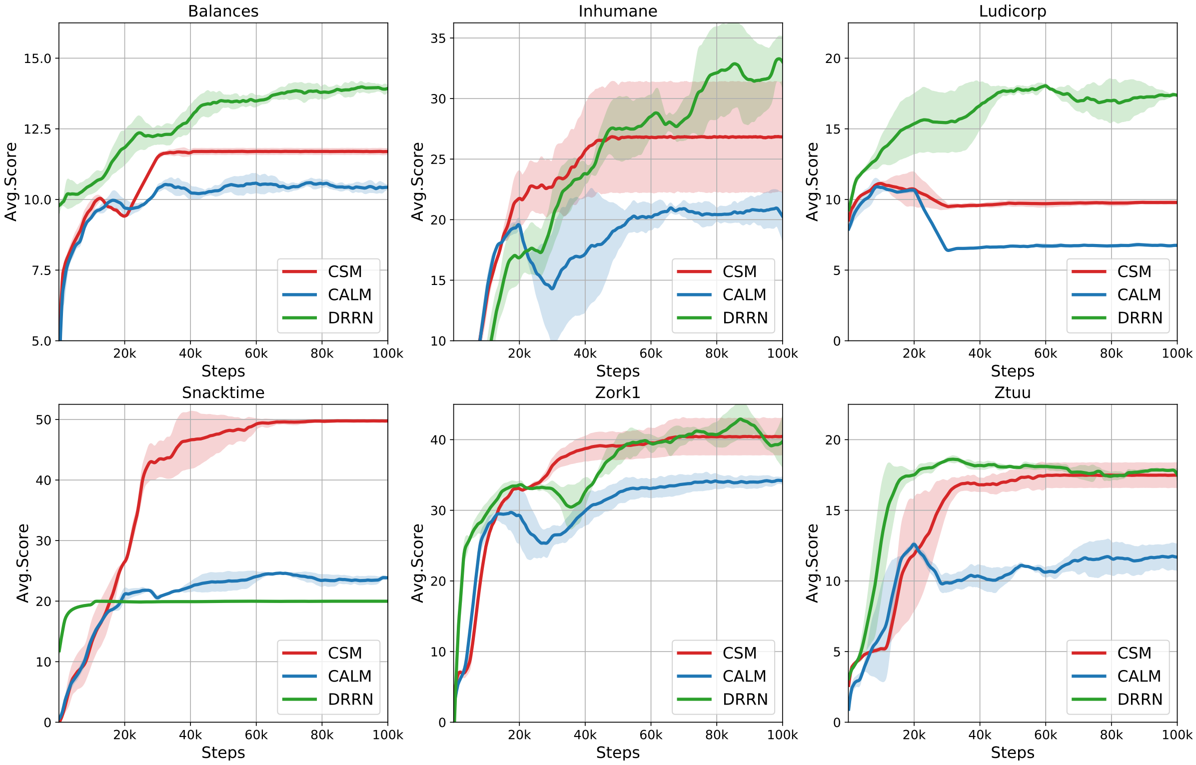


Figure 3: The performance of CSM compared to baselines (CALM and DRRN) throughout training. Shaded areas indicate one standard deviation. Our CSM outperforms CALM while getting much closer to DRRN. Besides, it successfully solves the game "Snacktime".

Table 2: The performance of CSM compared to baselines (CALM and DRRN) after training. The result with † is from Hausknecht et al. (2020). In six environments, our method obtains significant improvement compared to the CALM model, with an average normalized game score of 31.4%.

Game	*Generated $\mathcal{A}_t$*		*Oracle $\mathcal{A}_t$*	**Max**
	CSM	**CALM**	**DRRN**	
Balances	**11.7**	10.5	14.0	51
Inhumane	**27.0**	20.6	33.6	90
Ludicorp	**9.8**	6.8	17.5	150
Snacktime	**49.8**	24.0	20.0	50
Zork1	**40.6**	34.3	40.0	350
Ztuu	**17.5**	11.7	21.6†	100
Avg.Norm	**31.4%**	19.6%	24.9%	

the top 30 actions, i.e., $N = 30$. Then, we use the proposed confidence-based strategy to keep top-k highest-scoring action candidates (k<30). We set ξ as 0.6, and bound k to be no lower than 10. Following previous works, we define the score as the sum of rewards collected within an episode, and report the score averaged over the last 100 finished episodes.

LM For both CSM and CALM, we use the pre-trained GPT-2 model provided by Yao et al. (2020) as the LM module. The LM consists of 12 layers, 768 hidden sizes, and 12 attention heads. This module is first pre-trained on the WebText corpus (Radford et al., 2019), then re-trained on the ClubFloyd dataset (Yao et al., 2020), which consists of 426 human game playing transcripts on 590 games (note that the Jericho-supported games that we experiment with are not included).

RL Both CSM and CALM adopt the DRRN as the RL agent, except that the action candidate set is generated by the LM module. Given the current observation o_t, and a set of currently admissible actions $\mathcal{A}_t$, the RL agent first encodes o_t to build the state representation, then pairs it with each ac-

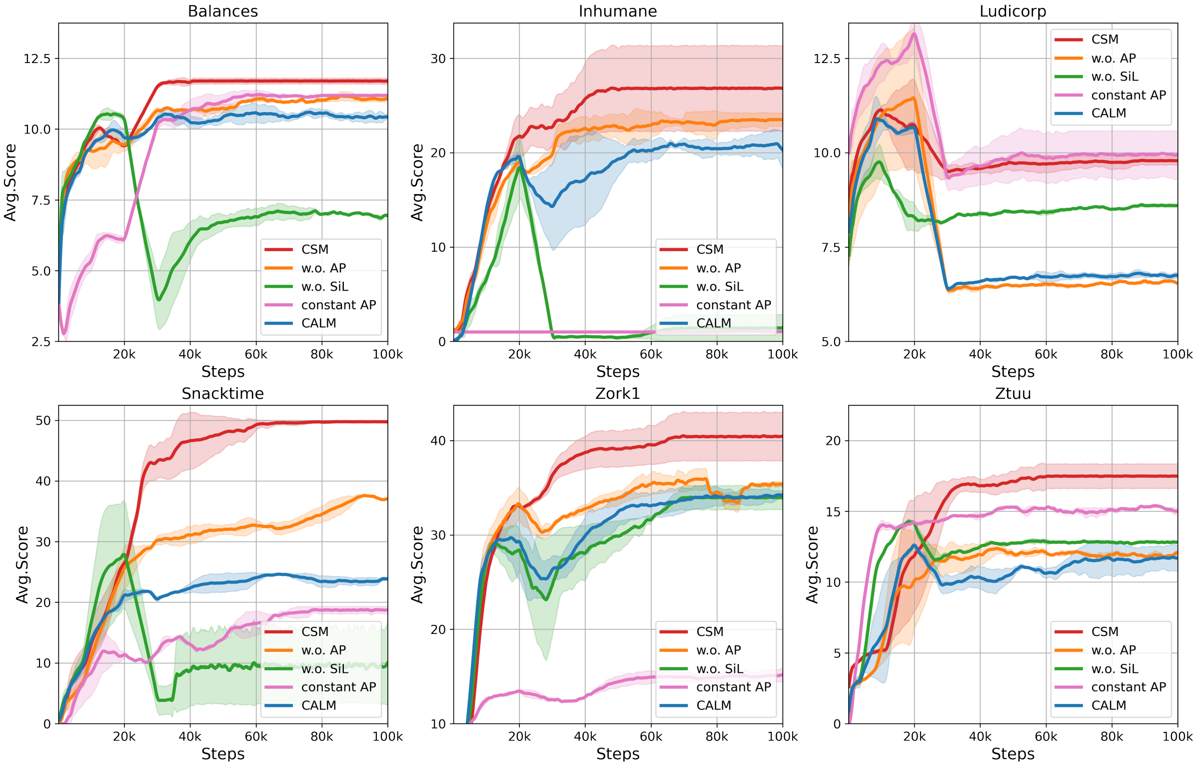


Figure 4: Average episode score throughout training for ablation models. Shaded areas indicate one standard deviation.

Table 3: Average episode score after training for ablation models. Overall, both the SiL and AP are crucial to our framework.

Game	CSM	w.o. AP	w.o. SiL	constant AP	CALM
Balances	**11.7**	11.1	7.0	11.2	10.5
Inhumane	**27.0**	23.6	1.5	1.1	20.6
Ludicorp	9.8	6.6	8.6	**10.0**	6.8
Snacktime	**49.8**	37.1	10.5	18.8	24.0
Zork1	**40.6**	35.6	34.1	15.2	34.3
Ztuu	**17.5**	12.1	12.9	15.1	11.7
Avg.Norm	**31.4%**	24.8%	10.8%	14.5%.	19.6%

tion candidate $a_{t,i} \in \mathcal{A}_t$ to compute the Q-value, which will be used as the probability for sampling the action a_t.

Warm-up Since this work does not address the RL exploration problem, we equip both CSM and CALM with a warm-up phase to facilitate training at the very beginning. During this phase, we follow Yao et al. (2020) to filter inadmissible actions from $\mathcal{A}_t$ through a pre-trained fast-text module, without applying SiL or AP. Then after this phase, the fast-text module will be discarded, and the LM has to generate the reliable $\mathcal{A}_t$ by itself. Note that this module is not essential, and could be replaced by other exploration strategies such as Zha et al. (2021); Yao et al. (2021). We leave such integration as a future direction.

5.4 Results

Fig. 3 shows the average episode score throughout training for the baselines, and Table 2 shows the average episode score after training for the baselines. Our CSM demonstrates its effectiveness by significantly outperforming the backbone CALM in all of the six games, with an average normalized game score of 31.4%. Given that DRRN has access to the "oracle" action set $\mathcal{A}_t$, its performance can be regarded as our "upper bound". We observe that the performance of CSM is much closer to DRRN, and even surpasses DRRN in two games. In particular, while DRRN gets stuck in the game "Snacktime", CSM solves this game, making its average normalized score among the highest of all. In Sec. 5.6, we further discuss this case by analyzing

Context: [CLS] scratch man [SEP] You scratch your pet, just gently, not to hurt him or anything. Hmm, your pet seems to have turned that into part of his dream or something, because he sure didn't move this time when you scratched him.right now you aren't carrying anything. (it's not like you need a lot of stuff, anyway.) [SEP]
CSM: **jump on man**, kiss pet, lick pet, kiss man, lick man, push pet, move arm, pull pet, pull arm, push man
CALM: pull man, push man, **jump on man**, push arm, pull pet, push pet, pull arm, lick man, kiss man, kiss pet, lick pet, smell, smell man, sniff, south, smell pet, wait, listen, east, northwest, southwest, get pet, down, jump, hug man, lick it, pet pet pet, up, hug pet, in, take pet, take all, eat pet, talk to pet, sleep, search pet, drop all, stand, open chest, open door, pet dog, pet pet dog, out, pull up on stool, west, north
DRRN: kiss on pet, **jump on him**, north, west

Context: [CLS] push wand [SEP] Once again you chomp down to change what's inside the box. This time there are these guys poking at each other, kinda fighting and joking around and all. This is the sort of thing your pet seems to enjoy watching sometimes, people getting hit with food and slipping and falling down and stuff. Everybody is laughing and nobody ever seems to really get hurt, so you guess it's OK.held carefully between your teeth is a magic wand. [SEP]
CSM: use wand on box, bite wand, push wand, **chew wand**, jump on man, take wand, get wand, read wand, pull wand, lick wand
CALM: push wand, **chew wand**, wait, northeast, south, southwest, northwest, east, rub wand, down, give wand to pet, pull wand, open box, take magic wand, put wand in box, get wand, pull lever, give wand to guy, take all, get magic wand, west, take wand, up, out, eat wand, give book to pet, enter box, read book, use wand on wand, drop wand, north
DRRN: examine legged, **push wand**, take inventory, get in all, take off legged, north, west

Figure 5: Sample gameplay from the game "Snacktime" along with the generated action candidates, and the action chosen by the RL agent (coloured with blue).

the underlying reasons.

5.5 Ablation Studies

In order to evaluate the contribution of the two components in CSM, we compare our model with two variants with either SiL ("w.o. AP") or AP ("w.o. SiL"). In order to demonstrate the effectiveness of confidence-based AP, we also employ constant AP. We set k to 12, which is the average number of actions selected by the confidence-based strategy. Fig. 4 shows the average episode score for the ablation models throughout training, and Table 3 shows the average episode score for the ablation models after training.

In general, adapting the LM with respect to the target game helps ("w.o. AP" v.s., "CALM"), while reducing the action space upon it further boosts the performance ("CSM" v.s., "w.o. AP"). Solely reducing the action space $\mathcal{A}_t$, in contrast, leads to poor performance ("CSM" v.s., "w.o. SiL" v.s., "CALM"). Also, simply utilizing the constant AP together with SiL results in a considerable performance drop. ("CSM" v.s., "constant AP"). Without SiL, the LM has a greater chance of incorrectly filtering actions that are essential to go through the target game.

5.6 Qualitative Analysis

To demonstrate the efficacy of the proposed framework, we present two gameplay examples from the game "Snacktime". Fig.5 shows the generated action candidates and the action chosen by the RL agent, where "Context" denotes c_t, "CSM" and "CALM" denote the actions generated by CSM and CALM respectively, "DRRN" denotes the "oracle" action set used by DRRN. In the first example, all models generate and select the correct action "jump on him", which leads to a +10 reward. Compared with CALM, CSM successfully reduces the action set from 30 to 10, relieving the burden for the RL agent. In the second example, both CSM and CALM generate action sets with the correct action "chew wand" being included. We found that the "oracle" action set provided by the environment is not always perfect, which explains why DRRN gets stuck here[‡]. It shows that our model is capable of generating high-quality, context-relevant actions, and further limits the action space while keeping key actions that may lead to higher scores in the games. Appendix B provides the detail interaction log of CSM on the game "Snacktime".

6 Conclusion

In this work, we studied reinforcement learning in solving the text-based game. We proposed the CSM framework to generate a set of action candidates for the RL agent, which alleviates the issue of combinatorial action space. During RL training, we

---

[‡] Similar phenomenon has also been reported in some other games (Tuyls et al., 2022)

collected and exploited past high-quality trajectories and utilised self-imitation learning to improve the language model. In addition, a confidence-based action pruning strategy was proposed to further restrict the action space. We evaluate our method using the Jericho benchmark. In a variety of text-based games, our method significantly improves the performance compared with the strong contemporary method, and even overcomes the challenging bottleneck in the game "Snacktime".

Limitations

In terms of limitations, text-based games are still far from being solved. Even if the agent has access to admissible actions, sparse rewards, language semantics and partial observability remain challenging obstacles for the existing game agent. In this study, we develop an effective framework to solve the issue of combinatorial action space. Future work can integrate our framework with methods that better leverage linguistic signals in order to make further progress in solving text-based games.

Acknowledgements

We thank the anonymous EACL reviewers for their insightful comments and constructive feedback.

References

Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and William L Hamilton. 2020. Learning dynamic knowledge graphs to generalize on text-based games. *arXiv preprint arXiv:2002.09127*.

Leonard Adolphs and Thomas Hofmann. 2020. Ledeepchef deep reinforcement learning agent for families of text-based games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7342–7349.

Prithviraj Ammanabrolu and Matthew Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. In *International Conference on Learning Representations (ICLR)*.

Prithviraj Ammanabrolu, Renee Jia, and Mark Riedl. 2022. Situated dialogue learning through procedural environment generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8099–8116.

Prithviraj Ammanabrolu and Mark Riedl. 2019. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, volume 1, pages 3557–3565.

Prithviraj Ammanabrolu, Ethan Tien, Zhaochen Luo, and Mark O Riedl. 2020. How to avoid being eaten by a grue: Exploration strategies for text-adventure agents. *arXiv preprint arXiv:2002.08795*.

Jacob Andreas and Dan Klein. 2016. Reasoning about pragmatics with neural listeners and speakers. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Austin, Texas. Association for Computational Linguistics.

Timothy Atkinson, Hendrik Baier, Tara Copplestone, Sam Devlin, and Jerry Swan. 2019. The text-based adventure ai competition. *IEEE Transactions on Games*, 11(3):260–266.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. *arXiv preprint arXiv:1806.11532*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2015. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*.

Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to active learn: A deep reinforcement learning approach. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 595–605.

Nancy Fulda, Daniel Ricks, Ben Murdoch, and David Wingate. 2017. What can you do with a rock? affordance extraction via word embeddings. *arXiv preprint arXiv:1703.03429*.

Simona Gandrabur, George Foster, and Guy Lapalme. 2006. Confidence estimation for nlp applications. *ACM Transactions on Speech and Language Processing (TSLP)*, 3(3):1–29.

Tanmay Gangwani, Qiang Liu, and Jian Peng. 2019. Learning self-imitating diverse policies. In *International Conference on Learning Representations (ICLR)*.

Xiaoxiao Guo, Mo Yu, Yupeng Gao, Chuang Gan, Murray Campbell, and Shiyu Chang. 2020. Interactive fiction game playing as multi-paragraph reading comprehension with reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7755–7765.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. 2020. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 7903–7910.

Matthew Hausknecht, Ricky Loynd, Greg Yang, Adith Swaminathan, and Jason D Williams. 2019. Nail: A general interactive fiction agent. *arXiv preprint arXiv:1902.04259*.

Ji He, Jianshu Chen, Xiaodong He, Jianfeng Gao, Lihong Li, Li Deng, and Mari Ostendorf. 2016. Deep reinforcement learning with a natural language action space. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1621–1630.

Hengyuan Hu, Denis Yarats, Qucheng Gong, Yuandong Tian, and Mike Lewis. 2019. Hierarchical decision making by generating and following natural language instructions. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:10025–10034.

Vishal Jain, William Fedus, Hugo Larochelle, Doina Precup, and Marc G Bellemare. 2020. Algorithmic improvements for deep reinforcement learning applied to interactive fiction. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 4328–4336.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Vincent Micheli and Francois Fleuret. 2021. Language models are few-shot butlers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9312–9318.

Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Pushkar Shukla, Sadhana Kumaravel, Gerald Tesauro, Kartik Talamadupula, Mrinmaya Sachan, and Murray Campbell. 2021. Text-based rl agents with commonsense knowledge: New challenges, environments and baselines. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pages 9018–9027.

Karthik Narasimhan, Tejas D Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–11.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. 2018. Self-imitation learning. In *International Conference on Machine Learning (ICML)*, pages 3878–3887. PMLR.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Dongwon Ryu, Ehsan Shareghi, Meng Fang, Yunqiu Xu, Shirui Pan, and Reza Haf. 2022. Fire burns, sword cuts: Commonsense inductive bias for exploration in text-based games. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 515–522.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Zijing Shi, Meng Fang, Yunqiu Xu, Ling Chen, and Yali Du. 2023. Stay moral and explore: Learn to behave morally in text-based games. In *International Conference on Learning Representations (ICLR)*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2020. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Ishika Singh, Gargi Singh, and Ashutosh Modi. 2021. Pre-trained language models as prior knowledge for playing text-based games. *arXiv preprint arXiv:2107.08408*.

Jens Tuyls, Shunyu Yao, Sham M. Kakade, and Karthik R Narasimhan. 2022. Multi-stage episodic control for strategic exploration in text games. In *International Conference on Learning Representations (ICLR)*.

Yunqiu Xu, Ling Chen, Meng Fang, Yang Wang, and Chengqi Zhang. 2020a. Deep reinforcement learning with transformers for text adventure games. In *IEEE Conference on Games (CoG)*, pages 65–72.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, and Chengqi Zhang. 2021. Generalization in text-based games via hierarchical reinforcement learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1343–1353.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, Joey Zhou, and Chengqi Zhang. 2022. Perceiving the world: Question-guided reinforcement learning for text-based games. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 538–560.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, Joey Tianyi Zhou, and Chengqi Zhang. 2020b. Deep reinforcement learning with stacked hierarchical attention for text-based games. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 16495–16507.

Shunyu Yao, Karthik Narasimhan, and Matthew Hausknecht. 2021. Reading and acting while blindfolded: The need for semantics in text game agents. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3097–3102, Online. Association for Computational Linguistics.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. 2020. Keep CALM and explore: Language models for action generation in text-based games. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8736–8754.

Xingdi Yuan, Marc-Alexandre Côté, Jie Fu, Zhouhan Lin, Christopher Pal, Yoshua Bengio, and Adam Trischler. 2019. Interactive language learning by question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2796–2806.

Xingdi (Eric) Yuan, Marc-Alexandre Côté, Alessandro Sordoni, Romain Laroche, Remi Tachet des Combes, Matthew Hausknecht, and Adam Trischler. 2018. Counting to explore and generalize in text-based games. In *European Workshop on Reinforcement Learning (EWRL)*.

Tom Zahavy, Matan Haroush, Nadav Merlis, Daniel J Mankowitz, and Shie Mannor. 2018. Learn what not to learn: Action elimination with deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pages 3562–3573.

Daochen Zha, Wenye Ma, Lei Yuan, Xia Hu, and Ji Liu. 2021. Rank the episodes: A simple approach for exploration in procedurally-generated environments. In *International Conference on Learning Representations (ICLR)*.

Zhiqiang Zhan, Jianyu Zhao, Yang Zhang, Jiangtao Gong, Qianying Wang, Qi Shen, and Liuxin Zhang. 2021. Grabbing the long tail: A data normalization method for diverse and informative dialogue generation. *Neurocomputing*, 460:374–384.

Appendix

The appendix is organized as follows: Sec. A shows more experiment results. Sec. B provides the interaction log of CSM on the game "Snacktime".

A More Results

Reproduction of DRRN Fig. 6 shows the reproducing result of the DRRN baseline, where "DRRN - Ours" denotes the "DRRN" used in the main paper. The dashed lines "DRRN - Official" denote the results reported in Hausknecht et al. (2020) and Yao et al. (2020). According to Tuyls et al. (2022), the action candidate set $\mathcal{A}_t$ provided by the environment is not always perfect, so that they manually augmented the environment-provided $\mathcal{A}_t$ with actions from the game walkthrough which are required for making progress[§]. We follow their setting to modify the environment and rerun the DRRN baseline, yielding much better performance than the official results except one game "Ztuu", which we use the official result in Table 2.

Reproduction of CALM Fig. 7 shows the reproducing result of the CALM baseline, where "CALM 20% WU - Ours" denotes the "CALM" used in the main paper. The dashed lines "CALM 100% WU - Official" denote the results reported in Yao et al. (2020). In terms of the original CALM, our replication results are comparable with or better than the official results ("CALM 100% WU" v.s., "CALM 100% WU - Official"). The original CALM adopts a fast-text model to filter out the inadmissible actions from $\mathcal{A}_t$ throughout the RL training process (i.e., they conduct warm-up for 100k steps), heavily alleviating the problem of generating inadmissible actions ("CALM 100% WU" v.s., "CALM w.o. WU"). However, obtaining this fast-text model requires prior knowledge, such as the additional training data and annotations. In our work, we would like to reduce the requirement of such external knowledge, and let the LM to conduct action pruning by itself. For all LM-based models, we only conduct warm-up for the first 20k steps, and discard the fast-text model afterwards ("CALM 20% WU"). As a future direction, we would like to consider more advanced warm-up strategies (Zha et al., 2021), thus eliminating the need for pre-training the fast-text model.

More results Besides the episode score, we provide more results for further analyzing self-imitation learning and action pruning. Regarding SiL, Fig. 8 and Fig. 9 show the average score and length of the trajectories collected in the ranked buffer, respectively. There's no doubt that the average score grows higher as the agent makes progress. Diverse trends could be observed in terms of the average length, since a newly-added trajectory might have both high score and more steps. Regarding AP, Fig. 10 shows the number of LM generated actions k, i.e., $|\hat{\mathcal{A}}_t|$, where it could be observed k gets close to the lower bound after pruning. Fig. 11 shows the LM probability of the top-1 generated action, and Fig. 12 shows the LM probability sum of the top-5 generated actions. After self-imitation learning, the top actions account for a larger proportion of the probability, making it safer for filtering those with low probabilities.

[§] `https://github.com/princeton-nlp/XTX`

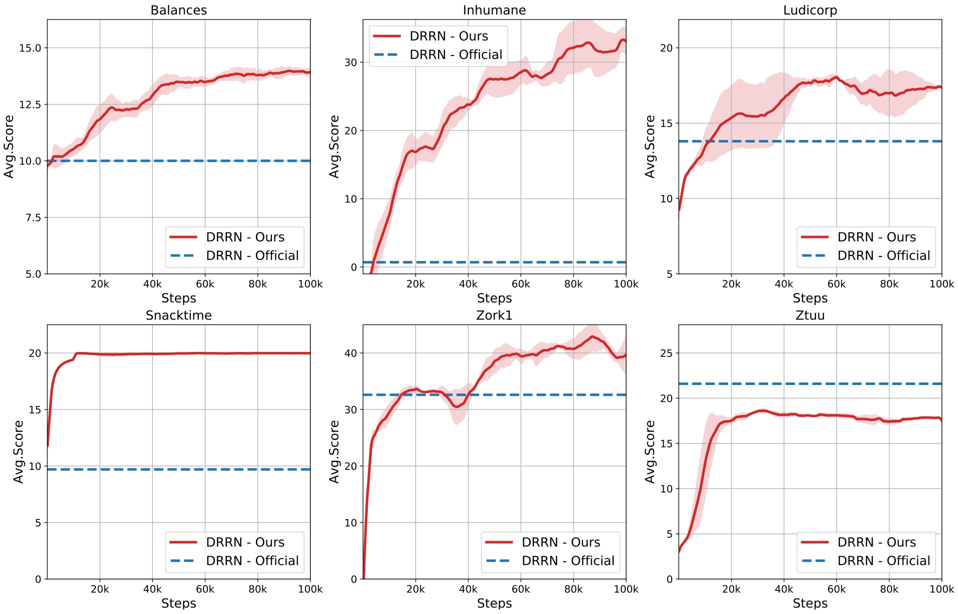


Figure 6: The reproducing result of the DRRN baseline, where "DRRN - Ours" denotes the "DRRN" used in the main paper. The dashed lines denote the results reported in Hausknecht et al. (2020) and Yao et al. (2020).

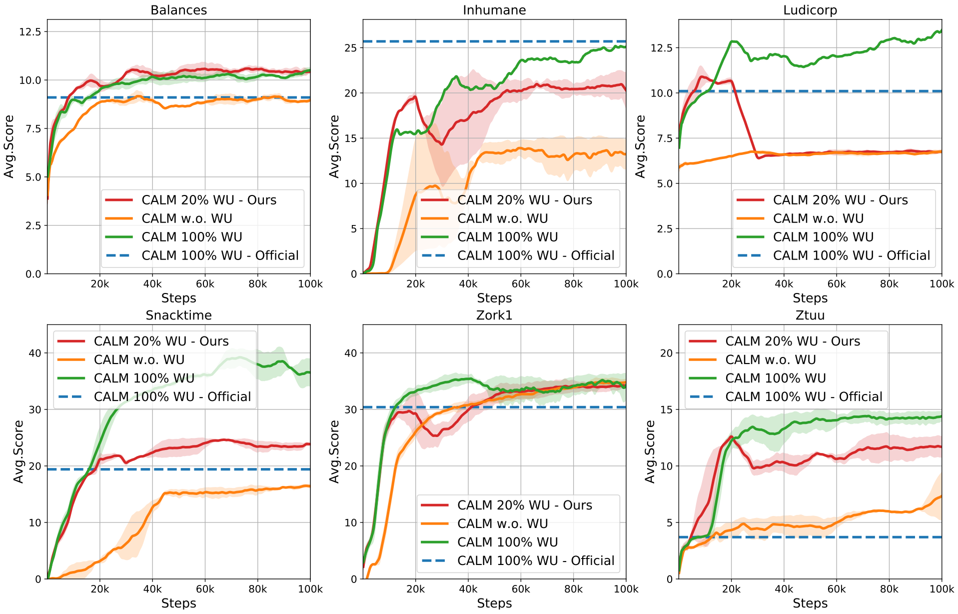


Figure 7: The reproducing result of the CALM baseline, where "CALM 20% WU - Ours" denotes the "CALM" used in the main paper. The dashed lines denote the results reported in Yao et al. (2020).

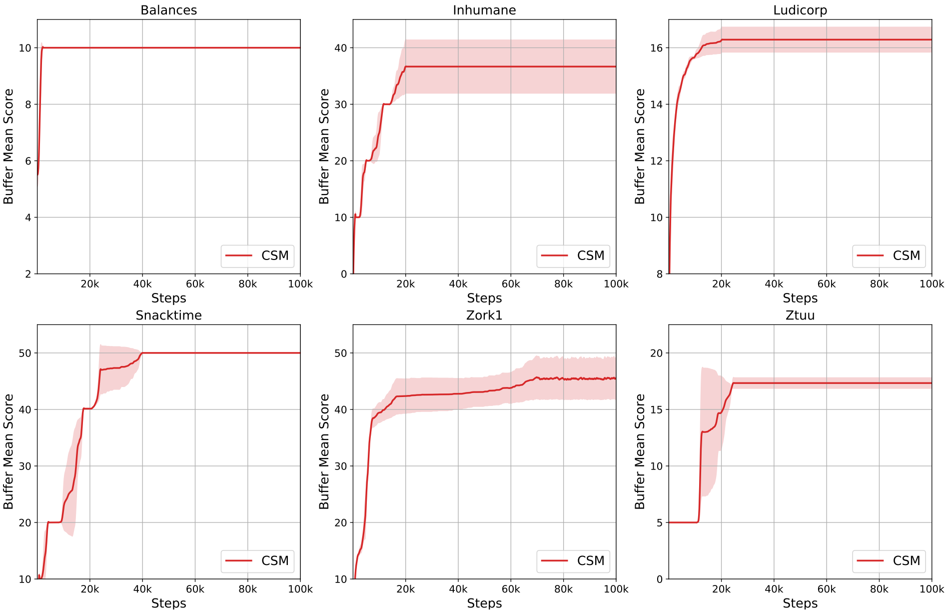


Figure 8: The average score of trajectories in the ranked buffer.

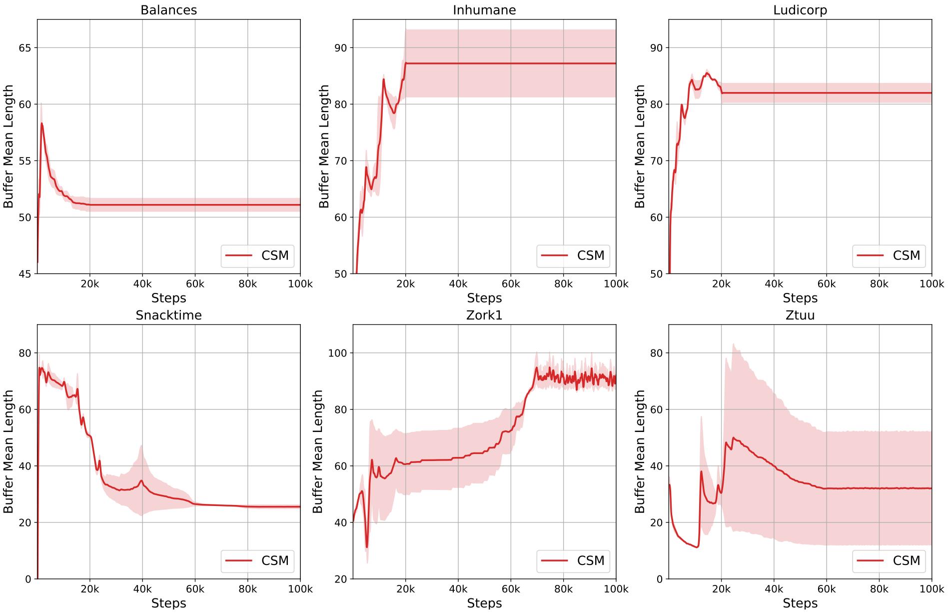


Figure 9: The average length of trajectories in the ranked buffer.

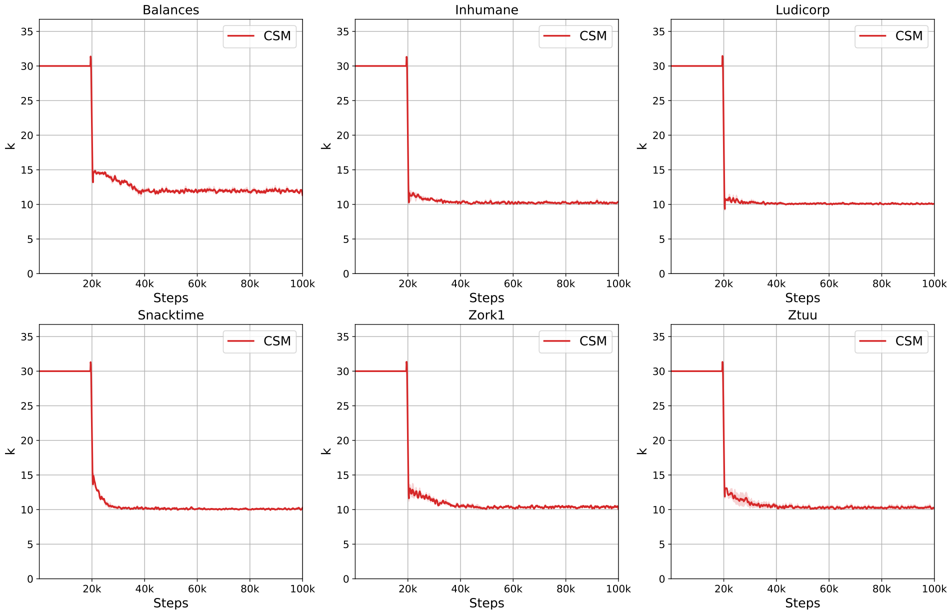


Figure 10: The number of LM generated actions k, i.e., $|\hat{\mathcal{A}}_t|$.

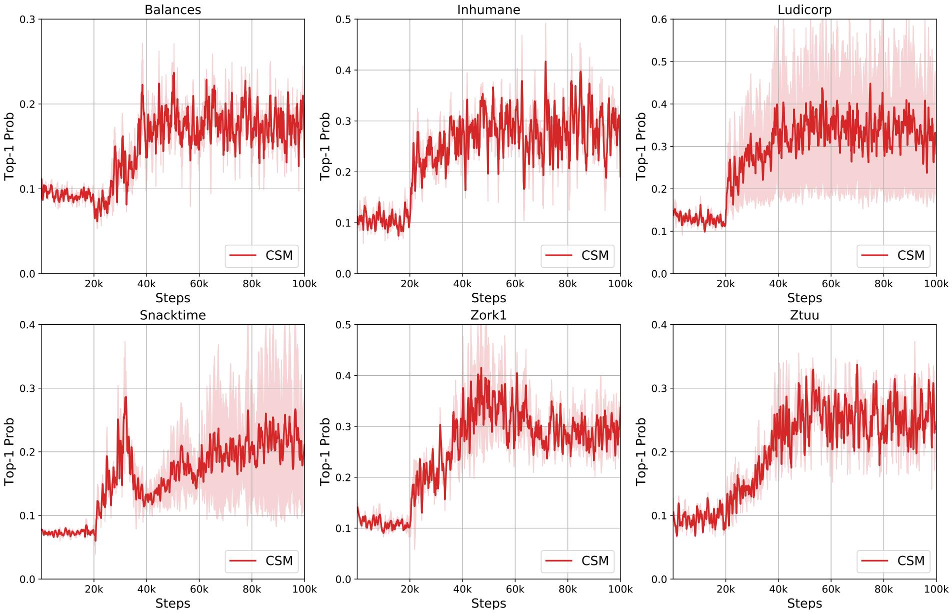


Figure 11: The LM probability of the top-1 generated action.

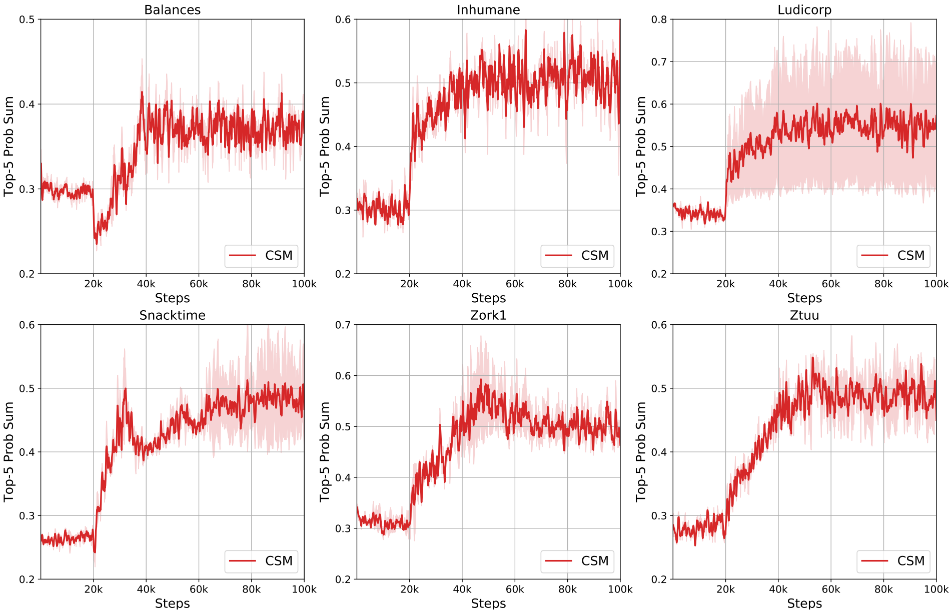


Figure 12: The LM probability sum of the top-5 generated actions.

B Interaction Example

We provide the interaction log of our CSM on the game “Snacktime”, where “Context” denotes c_t, “Actions” denotes $\hat{\mathcal{A}}_t$, “Action” denotes a_t, “Reward” denotes r_t after executing a_t, “Score” denotes the episodic sum of the collected rewards, and “Done” indicates whether this episode is finished.

```
Context56826: [CLS] reset [SEP]  <RRROWWGRROWL> Your stomach
   growls.  You've been vegging out in front of that flickering
   screen for hours, your pet by your side. Now your complaining
   stomach sets you up on your feet, and on a mission.  Snack Time!
   An Interactive Break for a Bite by Hardy the Bulldog (with help
   from Renee Choba) Release 1 / Serial number 080929 / Inform 7
   build 5U92 (I6/v6.31 lib 6/12N)  (For more information please type
    ABOUT)  Sitting Room This is the room where you sit a lot. Well,
   you sleep here sometimes too. But there is a different room that
   is just for sleeping, and it is to the north. There's another room
    to the west. It's the room with the food. That is a good room.
   Even though this is the sitting room you can't sit on everything.
   There
Actions56826: ["'s no [cls] [cl", "'s [cls] [cls", "'s no [cls] [
   cls", 'are [cls] [cls]', 'is no [cls] [cls', '[cls] [cls', 'are
    no [cls] [cls', 'are [cls] [cls']
>> Action56826: are no [cls] [cls
Reward56826: 0, Score 0, Done False

Context56827: [CLS] are no [cls] [cls [SEP] That's not a trick you
    know.right now you aren't carrying anything. (it's not like you
   need a lot of stuff, anyway.) [SEP]
Actions56827: ['move arm', 'scratch pet', 'lick pet', 'kiss pet', '
   jump on man', 'scratch man', 'pull man', 'pull pet', 'push pet', '
   pull arm']
>> Action56827: scratch man
Reward56827: 0, Score 0, Done False

Context56828: [CLS] That's not a trick you know. [SEP] scratch man
   [SEP] You scratch your pet, just gently, not to hurt him or
   anything.  Your pet twitches a little and stirs, then is still
   again. It's a start at waking him up.right now you aren't carrying
    anything. (it's not like you need a lot of stuff, anyway.) [SEP]
Actions56828: ['move arm', 'lick man', 'chew man', 'jump on man', '
   push pet', 'push man', 'lick pet', 'pull man', 'pull pet', 'pull
   arm']
>> Action56828: push pet
Reward56828: 0, Score 0, Done False

Context56829: [CLS] push pet [SEP] You tuck your head up under your
    pet and push back and forth to shake him.  Your pet gets one eye
   open and half heartedly pets at you. You're making progress.right
   now you aren't carrying anything. (it's not like you need a lot of
    stuff, anyway.) [SEP]
Actions56829: ['pull man', 'jump on man', 'lick man', 'scratch pet',
   'lick pet', 'scratch man', 'pull pet', 'pull arm', 'move arm', '
   push pet']
```

```
>> Action56829: scratch man
Reward56829: 0, Score 0, Done False

Context56830: [CLS]  scratch man [SEP]  You scratch your pet , just
   gently , not to hurt him or anything.  Hmm, your pet seems to have
   turned that into part of his dream or something , because he sure
   didn 't move this time when you scratched him.right now you aren 't
   carrying anything. (it 's not like you need a lot of stuff , anyway
   .) [SEP]
Actions56830: ['jump on man', 'kiss pet', 'lick pet', 'kiss man', '
   lick man', 'push pet', 'move arm', 'pull pet', 'pull arm', 'push
   man']
>> Action56830: jump on man
Reward56830: 10, Score 10, Done False

Context56831: [CLS]  jump on man [SEP]  You jump up and get your paws
    on your pet for a second before sliding back down.  Your pet
   shifts and stretches , blinking. He moves around until he is mostly
    sitting up.  "Hey Hardy ," he mumbles.  It might take a little
   more to get him up and moving , but at least now his eyes are open!
     [Your score has just gone up by ten points.]right now you aren 't
    carrying anything. (it 's not like you need a lot of stuff , anyway
   .) [SEP]
Actions56831: ['kiss man', 'south', 'in', 'out', 'southeast', '
   northwest', 'northeast', 'southwest', 'east', 'down']
>> Action56831: down
Reward56831: 0, Score 10, Done False

Context56832: [CLS]  down [SEP]  If you want to go to the food room ,
   it 's to the west , and the sleeping room is to the north.right now
   you aren 't carrying anything. (it 's not like you need a lot of
   stuff , anyway.) [SEP]
Actions56832: ['search soft thing', 'search couch', 'search sofa', '
   search door', 'search sleeping', 'search box', 'search arm', '
   search pet', 'search man', 'search fridge']
>> Action56832: search soft thing
Reward56832: 0, Score 10, Done False

Context56833: [CLS]  search soft thing [SEP]  You tug and pull at the
    little soft things until finally one breaks free from the bunch.
   As you dislodge it , something clatters to the floor. Ah ha! You've
    found the magic wand! Those little soft things were trying to eat
    it!  "So that 's where that got away to ," says your pet , snatching
    up the magic wand. He points it at the box of light and makes it
   even louder.held carefully between your teeth is a square soft
   thing. [SEP]
Actions56833: ['west west', 'west west west west west west west west
   west', 'west west west west west west west west west west', 'west
   west west west west west west west', 'west west west west west
   west west', 'west west west west west west', 'west west west', '
   west west west west west', 'west west west west', 'west']
```

```
>> Action56833: west west west west west west west west west west
Reward56833: 0, Score 10, Done False

Context56834: [CLS] west west west west west west west west west
   west [SEP]  Food Room Food room, glorious food room!  Your bowl
   for food and water is here, but that's only the beginning. There's
    also a lot of secret places where your pet keeps food and things
   that go with food and things that make food better. There's a
   thing that gets hot and a box that stays cold, and that is your
   favorite because you know there is always food inside there. A
   cloth hangs down from the door of the cold box.held carefully
   between your teeth is a square soft thing. [SEP]
Actions56834: ['open cold box', 'open box', 'open door', 'get cloth',
    'open fridge', 'take cloth', 'pull cloth', 'open refrigerator', '
   use wand on box', 'push cloth']
>> Action56834: open refrigerator
Reward56834: 0, Score 10, Done False

Context56835: [CLS] open refrigerator [SEP] This is a good trick!
   (As you open your mouth, you drop the square soft thing.)  You
   pull on the cloth with your teeth and back up as the door opens.
   Inside you can see various leftovers, some jars, some sandwich
   fixings and one shiny can.right now you aren't carrying anything.
   (it's not like you need a lot of stuff, anyway.) [SEP]
Actions56835: ['get can', 'take can', 'get all from fridge', 'take
   all from fridge', 'take all', 'get all', 'get all from
   refrigerator', 'take all from refrigerator', 'get everything', '
   take everything']
>> Action56835: take all from fridge
Reward56835: 0, Score 10, Done False

Context56836: [CLS] take all from fridge [SEP] leftovers: All the
   food in the cold box is up too high for you or closed away. Looks
   like you will need your pet to help you get a snack. jars: Jars
   are no good for you, they just tease you by letting you see what's
    inside. But you can't get them open without your pet's help
   anyway. sandwich fixings: All the food in the cold box is up too
   high for you or closed away. Looks like you will need your pet to
   help you get a snack. shiny can: You snatch up the shiny can with
   your mouth and hold on tight.held carefully between your teeth is
   one shiny can. [SEP]
Actions56836: ['east east', 'east', 'east east east', 'east east east
    east', 'east east east east east', 'east east east east east east
   ', 'east east east east east east east', 'east east east east east
    east east east', 'east east east east east east east east east',
   'east east east east east east east east east east']
>> Action56836: east east
Reward56836: 0, Score 10, Done False

Context56837: [CLS] east east [SEP] Before you leave the food room,
    you push the cold box door with your nose to close it. (It's
```

```
   important to conserve energy.)  Sitting Room Your pet is here,
   staring at the box of light, holding on tight to the magic wand.
   held carefully between your teeth is one shiny can. [SEP]
Actions56837: ['feed pet', 'feed man', 'give can to pet', 'give can
   to man', 'wait', 'push wand', 'east', 'feed feed man', 'take can',
    'down']
>> Action56837: give can to man
Reward56837: 10, Score 20, Done False

Context56838: [CLS]  give can to man [SEP]  Your pet smiles as he
   reaches for the can. "Good boy, Hardy," he says, scratching you
   behind the ears. Your tail starts wagging; you just can't help it.
    (You've always been a fool for a good behind the ears scratch.)
   The can hisses with releasing air as the top pops, and then your
   pets takes a few good gulps. He must have been really thirsty,
   because soon he stands up and walks off to the other room.  From
   the other room, you hear your pet banging around and yelling about
    the dino. Your pet must have almost tripped on it. He's always
   almost tripping on little stuff like that.  After a second, you
   hear a click as the door to the water room closes. Now you have
   the long soft thing all to yourself!  [Your score has just gone up
    by ten points.] right now you aren't carrying anything. (it's not
   like you need
Actions56838: ['to move', 'all', 'pet', 'your pet', 'much', 'a pet',
   'anything', 'you', 'it', 'any', 'anymore']
>> Action56838: pet
Reward56838: 0, Score 20, Done False

Context56839: [CLS]  pet [SEP]  That's not a trick you know. right now
    you aren't carrying anything. (it's not like you need a lot of
   stuff, anyway.) [SEP]
Actions56839: ['get all', 'take all', 'take everything', 'get
   everything', 'take pet', 'get pet', 'take all from refrigerator',
   'take all from fridge', 'get can', 'take can']
>> Action56839: get everything
Reward56839: 0, Score 20, Done False

Context56840: [CLS]  That's not a trick you know. [SEP]  get
   everything [SEP]  magic wand: You snatch up the magic wand with
   your mouth and hold on tight. held carefully between your teeth is
   a magic wand. [SEP]
Actions56840: ['use wand on box', 'push wand', 'bite wand', 'chew
   wand', 'take wand', 'get wand', 'read wand', 'pull wand', 'lick
   wand', 'east']
>> Action56840: chew wand
Reward56840: 0, Score 20, Done False

Context56841: [CLS]  chew wand [SEP]  Making sure you are facing the
   box of light, you carefully press down on the wand with your teeth
    and are delighted when the box of light flickers and changes!
   Now you see a bunch of tiny people running around chasing a little
```

```
    bitty ball. Uh oh, someone got pushed. Ugh, the faces got all
   huge, and they don't look happy.held carefully between your teeth
   is a magic wand. [SEP]
Actions56841: ['use wand on box', 'bite wand', 'push wand', 'chew
   wand', 'scratch wand', 'take wand', 'get wand', 'read wand', 'pull
    wand', 'lick wand']
>> Action56841: push wand
Reward56841: 0, Score 20, Done False

Context56842: [CLS]  push wand [SEP]  Once again you chomp down to
   change what's inside the box. This time there are these guys
   poking at each other, kinda fighting and joking around and all.
   This is the sort of thing your pet seems to enjoy watching
   sometimes, people getting hit with food and slipping and falling
   down and stuff. Everybody is laughing and nobody ever seems to
   really get hurt, so you guess it's OK.held carefully between your
   teeth is a magic wand. [SEP]
Actions56842: ['use wand on box', 'bite wand', 'push wand', 'chew
   wand', 'jump on man', 'take wand', 'get wand', 'read wand', 'pull
   wand', 'lick wand']
>> Action56842: chew wand
Reward56842: 10, Score 30, Done False

Context56843: [CLS]  chew wand [SEP]  You've got the hang of this
   trick now for sure! One more bite on the wand brings up something
   new.  This is more like it! Inside the box now is this lady
   putting together some tasty looking food! You're almost drooling
   just watching. This has got to make your pet think about eating!
   [Your score has just gone up by ten points.]held carefully between
    your teeth is a magic wand.  you finally hear a familiar rush of
   water from the little water room and, after a second, the sound of
    the door opening. [SEP]
Actions56843: ['use wand on box', 'south', 'southeast', 'northwest',
   'southwest', 'out', 'northeast', 'west', 'take wand', 'east']
>> Action56843: northwest
Reward56843: 0, Score 30, Done False

Context56844: [CLS]  northwest [SEP]  If you want to go to the food
   room, it's to the west, and the sleeping room is to the north.
   You finally hear a familiar rush of water from the little water
   room and, after a second, the sound of the door opening.held
   carefully between your teeth is a magic wand.  from the other room
   , you hear your pet banging around and yelling about the dino.
   your pet must have almost tripped on it. he's always almost
   tripping on little stuff like that.  your pet walks in. he starts
   to sit down on the long soft thing but stops when he sees what's
   in the box of light. that lady cooking is doing her job! you pet
   stands and watches, transfixed. he must be thinking of food! [SEP]
Actions56844: ['out', 'southeast', 'down', 'south', 'northwest', '
   southwest', 'northeast', 'wait', 'east', 'take wand']
>> Action56844: northeast
```

```
Reward56844: 0, Score 30, Done False

Context56845: [CLS]  northeast [SEP]  If you want to go to the food
   room, it's to the west, and the sleeping room is to the north.
   From the other room, you hear your pet banging around and yelling
   about the dino. Your pet must have almost tripped on it. He's
   always almost tripping on little stuff like that.  Your pet walks
   in. He starts to sit down on the long soft thing but stops when he
    sees what's in the box of light. That lady cooking is doing her
   job! You pet stands and watches, transfixed. He must be thinking
   of food!held carefully between your teeth is a magic wand.  your
   pet makes his way to the food room.  your pet starts rattling
   around in the food room. yes! he must be making a snack! [SEP]
Actions56845: ['west', 'southeast', 'down', 'northwest', 'south', '
   southwest', 'east', 'northeast', 'out', 'wait']
>> Action56845: southwest
Reward56845: 0, Score 30, Done False

Context56846: [CLS]  southwest [SEP]  If you want to go to the food
   room, it's to the west, and the sleeping room is to the north.
   Your pet makes his way to the food room.  Your pet starts rattling
    around in the food room. Yes! He must be making a snack!held
   carefully between your teeth is a magic wand.  from the other room
    you can hear the happy sound of the cold box being opened. [SEP]
Actions56846: ['down', 'south', 'southeast', 'wait', 'northwest', '
   east', 'take wand', 'northeast', 'out', 'southwest']
>> Action56846: east
Reward56846: 0, Score 30, Done False

Context56847: [CLS]  east [SEP]  If you want to go to the food room,
   it's to the west, and the sleeping room is to the north.  From the
    other room you can hear the happy sound of the cold box being
   opened.held carefully between your teeth is a magic wand.  from
   the other room you can hear the sound of secret places opening and
    closing. [SEP]
Actions56847: ['southeast', 'northwest', 'south', 'out', 'northeast',
    'take wand', 'down', 'east', 'southwest', 'read wand']
>> Action56847: read wand
Reward56847: 0, Score 30, Done False

Context56848: [CLS]  read wand [SEP]  A magical, bumpy thing your pet
    uses to control the box of light.  From the other room you can
   hear the sound of secret places opening and closing.held carefully
    between your teeth is a magic wand.  you don't hear anything from
    the food room. but then, making a sandwich is not a very noisy
   job. [SEP]
Actions56848: ['take wand', 'southeast', 'northwest', 'south', '
   northeast', 'down', 'out', 'east', 'read wand', 'southwest']
>> Action56848: east
Reward56848: 0, Score 30, Done False
```

```
Context56849: [CLS]  east [SEP]  If you want to go to the food room,
   it's to the west, and the sleeping room is to the north.  You don'
   t hear anything from the food room. But then, making a sandwich is
    not a very noisy job.held carefully between your teeth is a magic
    wand.  you don't hear anything from the food room. but then,
   making a sandwich is not a very noisy job. [SEP]
Actions56849: ['south', 'down', 'take wand', 'southeast', 'northwest
   ', 'out', 'northeast', 'southwest', 'read wand', 'east']
>> Action56849: down
Reward56849: 0, Score 30, Done False

Context56850: [CLS]  down [SEP]  If you want to go to the food room,
   it's to the west, and the sleeping room is to the north.  You don'
   t hear anything from the food room. But then, making a sandwich is
    not a very noisy job.held carefully between your teeth is a magic
    wand.  from the other room, you hear the sound of the cold box
   closing. you're sure that in just a minute your pet will be
   leaving the food room with the sandwich! [SEP]
Actions56850: ['northwest', 'southeast', 'south', 'down', 'northeast
   ', 'southwest', 'out', 'read wand', 'east', 'take wand']
>> Action56850: northwest
Reward56850: 0, Score 30, Done False

Context56851: [CLS]  northwest [SEP]  If you want to go to the food
   room, it's to the west, and the sleeping room is to the north.
   From the other room, you hear the sound of the cold box closing.
   You're sure that in just a minute your pet will be leaving the
   food room with the sandwich!held carefully between your teeth is a
    magic wand. [SEP]
Actions56851: ['west west west west west  west', 'west west west west
     west', 'west west west  west', 'west west  west', 'west west
   west west west west west', 'west west west west west west', 'west
   west west west west', 'west west west west', 'west west west', '
   west west', 'west west west  west  west', 'west west  west  west']
>> Action56851: west west west
Reward56851: 0, Score 30, Done False

Context56852: [CLS]  west west west [SEP]   Food Room Your pet is
   here, working on making a snack.  You can also see a square soft
   thing here.  Your pet seems to have finished cleaning up after
   making the snack. He turns to leave. Now is the moment of truth.
   held carefully between your teeth is a magic wand.  sandwich in
   hand, your pet heads for the sitting room.  you follow your pet as
    he makes his way to the long soft thing and starts munching away
   on the sandwich. in a moment, it is gone, along with your dream of
    the perfect snack. looks like you'll have to wait until dinner to
    eat. poor hungry puppy!           you have missed your chance to
    snack       in that game you scored 30 out of a possible 50, in
   25 turns.   would you like to restart, restore a saved game or
   quit [SEP]
```

```
Actions56852: ['give sandwich to pet', 'give sandwich sandwich to pet
   ', 'give sandwich sandwich sandwich to pet', 'give sandwich
   sandwich sandwich sandwich sandwich sandwich sandwich to pet', '
   give sandwich sandwich sandwich sandwich sandwich sandwich to pet
   ', 'give sandwich sandwich sandwich sandwich sandwich to pet', '
   give sandwich sandwich sandwich sandwich to pet', 'give sandwich
   sandwich sandwich sandwich sandwich sandwich sandwich sandwich
   sandwich', 'give sandwich sandwich sandwich sandwich sandwich
   sandwich sandwich sandwich to']
>> Action56852: give sandwich sandwich sandwich sandwich sandwich
   sandwich sandwich to pet
Reward56852: 20, Score 50, Done True
```

Investigating the Effect of Relative Positional Embeddings on AMR-to-Text Generation with Structural Adapters

Sebastien Montella[*]
Orange Innovation / Lannion, France
Aix-Marseille Univ. CNRS, LIS / Marseille, France
sebastien.montella@huawei.com

Alexis Nasr
AMU/CNRS/LIS, Marseille, France
alexis.nasr@lis-lab.fr

Johannes Heinecke
Orange Innovation / Lannion, France
johannes.heinecke@orange.com

Frederic Bechet
AMU/CNRS/LIS, Marseille, France
frederic.bechet@lis-lab.fr

Lina M. Rojas-Barahona
Orange Innovation / Lannion, France
linamaria.rojasbarahona
@orange.com

Abstract

Text generation from Abstract Meaning Representation (AMR) has substantially benefited from the popularized Pretrained Language Models (PLMs). Myriad approaches have linearized the input graph as a sequence of tokens to fit the PLM tokenization requirements. Nevertheless, this transformation jeopardizes the structural integrity of the graph and is therefore detrimental to its resulting representation. To overcome this issue, Ribeiro et al. (2021b) have recently proposed StructAdapt, a structure-aware adapter which injects the input graph connectivity within PLMs using Graph Neural Networks (GNNs). In this paper, we investigate the influence of Relative Position Embeddings (RPE) on AMR-to-Text, and, in parallel, we examine the robustness of StructAdapt. Through ablation studies, graph attack and link prediction, we reveal that RPE might be partially encoding input graphs. We suggest further research regarding the role of RPE will provide valuable insights for Graph-to-Text generation.

1 Introduction

Earliest works on AMR-to-Text generation were mostly based on statistical methods. A common practice was to convert AMR-to-Text task into an already studied problems such as Tree-to-Text (Flanigan et al., 2016; Lampouras and Vlachos, 2017), aligned text-to-text (Pourdamghani et al., 2016), Travel Sales Problems (Song et al., 2016) or Grammatical Framework (Ranta, 2011). Recently, most methods are neural-centered with an encoder-decoder architecture (Sutskever et al., 2014) as a backbone (Konstas et al., 2017; Takase et al., 2016; Cao and Clark, 2019). Unfortunately, this architecture coerces the AMR to be linearized as a sequence of tokens. This ends up in structural information loss. To tackle this issue, several strategies have attempted to integrate structure using message propagation (Song et al., 2018; Guo et al., 2019; Damonte and Cohen, 2019; Ribeiro et al., 2019; Zhang et al., 2020b; Zhao et al., 2020). A limitation of those is the absence of pretraining, as demonstrated by Ribeiro et al. (2021a). To this end, Ribeiro et al. (2021b) introduced StructAdapt for lightweight AMR-to-Text with structural adapters. As linearization and tokenization of the input graph are mandatory steps for PLMs, StructAdapt first defines a new graph where nodes are the resulting subwords from the tokenization. As a result, adapter can henceforth include GNN layers operating on the subsequent graph while leveraging pretrained representations.

However, although studies have been made to probe position embeddings (Wang and Chen, 2020; Wang et al., 2021; Dufter et al., 2022), their role on graph encoding has remained unanswered. In this

---

[*] Work done while at Orange Innovation (Lannion, France). Now at Huawei Edinburgh Research Centre (United Kingdom).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 727–736
May 2-6, 2023

paper, we are particularly interested in measuring the saliency of RPE with StructAdapt for AMR-to-Text generation. Our novelty is not in proposing a new method to encode graphs such as (Schmitt et al., 2021) but rather in revealing the interesting behaviours of RPE along with StructAdapt.

2 STRUCTADAPT: A Structural Adapter

A major issue in AMR-to-Text, and more generally Graph-to-Text with Transformers (Vaswani et al., 2017), is the linearization of the input structure. The linearization of the graph returns a sequence of node and edge labels according to a certain traversal of the graph. Nonetheless, adjacent nodes in the graph may be at multiple positions away from one another in the final serialization. To counteract this, Ribeiro et al. (2021b) introduced StructAdapt, a structure-aware (encoder) adapter. It solves the problem of segmented nodes labels by reconstructing a new graph from the resulting subwords. More specifically, the relations are primary reified as new nodes in the AMR graph. Furthermore, labels of those reified relations will be added in the vocabulary as new tokens and therefore will not be decomposed into subwords. However, the labels of the original nodes can still be chunked. To deal with this, each subword node will be connected independently to the reified relation of the initial (non-chunked) node. An example is outlined in Figure 1. As a consequence, the vanilla Adapter can now integrate any GNN-based neural network which operates on the new constructed graph (Figure 1), where nodes are the input tokens. Concretely, StructAdapt replaces the first stacked MLP of vanilla adapter with a GNN-based model as shown in Figure 2. For AMR-to-Text, only the encoder is equipped with StructAdapt in order to encode AMR structure. The decoder layers adopt vanilla adapters. In our study, we consider three different GNN-based models which are Graph Convolutional Network (GCN) (Kipf and Welling, 2017), Graph Attention network (GAT) (Veličković et al., 2017) and Relational Graph Convolutional Network (RGCN) (Schlichtkrull et al., 2018). GCN computes a representation for each node a which is a (normalized) aggregation function of representation of its neighbor nodes noted $\mathcal{N}(a)$. GAT is akin to GCN but differs in that aggregation of neighbors embeddings are weighted using an attention mechanism. Unlike GAT and GCN, RGCN further captures the type of the relation between two nodes. In our case, as AMR relations are reified and stand for new nodes of the graphs, our new relations can either be of *direct* ($a \xrightarrow{d} b$) or *reverse* ($a \xleftarrow{r} b$) connections type as in (Ribeiro et al., 2021b). The details of the representations computation for each model can be found in Appendix A. The returned nodes embeddings will then be given as input features to the following MLP.

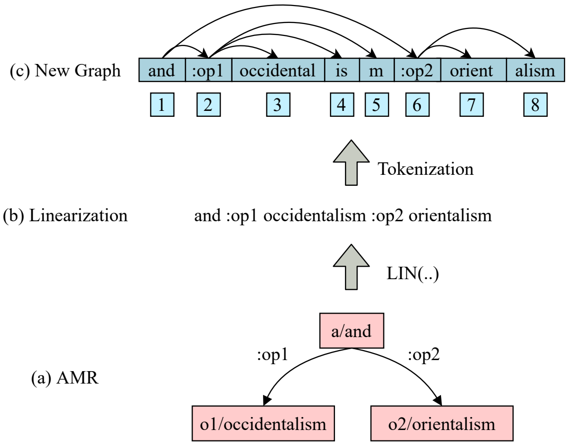


Figure 1: Examples of AMR tokenization for the sentence *"Occidentalism and Orientialism."*. The resulting input graph in (c) contains 8 nodes.

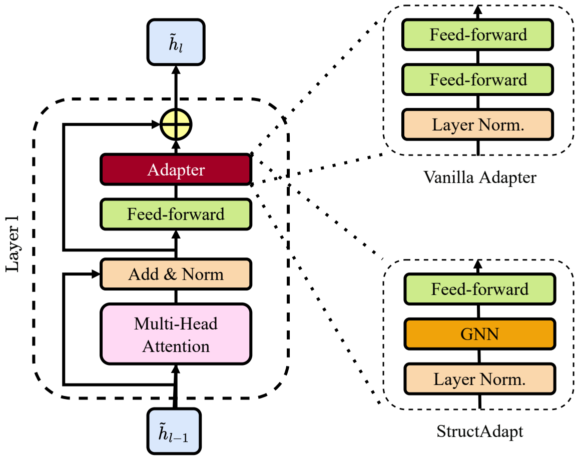


Figure 2: Vanilla Adapter vs StructAdapt

3 Relative Position Embeddings

Instead of adding Absolute Position Embeddings (APE) directly to the token embedding as in standard Transformer model, some models such as T5 make use of relative position embeddings inspired from Shaw et al. (2018). As an alternative to APE, RPE offer interesting features. A noteworthy limitation of APE is the need to set a limit of available positions. Therefore, long sequences may have

Adapter		BLEU ↑	METEOR ↑	Chrf++ ↑	TER ↓	BERTScore ↑	$\mathcal{M}$ ↑	$\mathcal{F}$ ↑
MLP	**w/ RPE**	**41.6**±0.3	**56.5**±0.3	**70.6**±0.2	**45.9**±0.5	**95.6**±0.0	**84.2**±0.1	**78.0**±0.5
	w/o RPE	16.3±0.5	38.5±0.8	49.9±1.1	85.5±0.9	91.8±0.2	81.9±1.1	76.2±0.6
GCN	**w/ RPE**	**42.6**±0.8	**56.7**±0.4	**71.0**±0.5	**44.8**±0.7	**95.7**±0.1	**84.6**±0.3	**79.0**±0.6
	w/o RPE	34.4±0.8	52.0±0.7	64.8±0.6	55.8±1.2	94.6±0.1	79.2±0.6	75.2±1.0
GAT	**w/ RPE**	**42.8**±0.1	**57.0**±0.1	**71.1**±0.0	**44.3**±0.3	**95.8**±0.0	**84.8**±0.1	**78.5**±0.8
	w/o RPE	34.8±1.1	52.3±0.7	64.8±0.7	54.9±1.4	94.6±0.2	79.6±0.8	75.6±0.4
RGCN	**w/ RPE**	**44.7**±0.6	**58.2**±0.3	**72.5**±0.3	**42.6**±0.4	**96.0**±0.0	**85.5**±0.2	**79.6**±0.7
	w/o RPE	39.9±0.8	55.7±0.5	68.9±0.8	48.8±1.3	95.3±0.1	83.1±0.4	78.0±0.6

Table 1: Comparing impact of Relative Positional Embeddings (RPE) on generation. We report mean performances (±s.d.) over 3 seeds.

Adapter	Meaning Preservation	Linguistic Correctness
MLP w/ RPE	**4.8**±1.2	**5.5**±0.9
MLP w/o RPE	3.5±1.5	5.2±1.2
GCN w/ RPE	**5.0**±1.2	**5.6**±0.8
GCN w/o RPE	4.7±1.3	5.5±1.0
RGCN w/ RPE	**5.2**±1.1	**5.6**±0.8
RGCN w/o RPE	4.7±1.3	5.4±1.0

Table 2: Human Evaluation. Mean scores (±s.d.)

to be segmented. Furthermore, APE are directly added to the token representation leading to information inconsistency, i.e. position versus semantic information. To this extent, Shaw et al. (2018) came up with relative position encodings which are supplied to the self-attention mechanism by simply adding a scalar to the logits encoding the supposed relation between a current token i and a token j.

4 Experiments

Throughout our experiments, we make use of the LDC2020T02 dataset (AMR 3.0 release)[1] and use the T5$_{\text{base}}$ model which employs RPE. The training and evaluation details can be found in Appendix B and C, respectively.

4.1 Exploring the Salience of RPE

In this section, we investigate the influence of RPE on the generation quality using structural adapter. RPE are computed in each Transformer head. Position information is then forwarded to the adapter module on top (Figure 2). However, since connections between input nodes (i.e. tokens) are already given to structural adapters in encoder, it is legitimate to question the necessity for RPE on the encoder part but also how would the generation quality vary without such information. Hence, we propose to remove the RPE from the encoder heads to gauge their salience to structure encoding for downstream language generation. Since decoder is not encoding any graph structure, we leave RPE in decoder untouched. For better readability, MLP, GCN, GAT, and RGCN respectively denote: the vanilla adapter, StructAdapt with a GCN layer, StructAdapt with a GAT layer and StructAdapt with a RGCN layer. MLP-based adapter with RPE is our baseline. Results are given in Table 1. A human evaluation is also provided for some encoder adapters in Table 2. First, it is apparent that using RPE systematically yields better generation performances. For the vanilla adapter (i.e. our baseline), we note a 25.3% absolute drop in BLEU when removing RPE. This can also be seen on human evaluation. More than one point is lost toward meaning preservation. The downturn for linguistic correctness is less important since T5 is pretrained and thus rarely prone to syntax errors. Such a result is not surprising for MLP-based adapter since it solely relies on RPE to differentiate tokens at different positions in the linearized AMR. However, a striking observation is that getting rid of RPE for GNN-based adapters leads to lower performances than our baseline. Indeed, when removing RPE when using structural adapter, we would have expected GNN-based approaches to be as competitive as a MLP-based adapter with RPE. We report a relative drop of 12.5 points in BLEU from the baseline. The same conclusion can be drawn from Table 2. This indicates that RPE are capturing relevant information for final generation. To further assess the impact and the role of RPE, we conduct a *graph attack* experiment. Instead of conveying the correct adjacency matrix, we propose to corrupt connectivity information. We randomly generate an adjacency matrix such that generated matrix does not contain any actual connection. We suppose that without RPE, structure-aware adapter will lead to a significant decrease in generated text due to the absence of information about the graph nor the position of nodes in the input sequence. We are especially interested to measure to which extent RPE might be able to take over the encoding of the graph for generation.

---

[1] https://catalog.ldc.upenn.edu/LDC2020T02

Adapter		BLEU ↑	METEOR ↑	Chrf++ ↑	TER ↓	BERTScore ↑	$\mathcal{M}$ ↑	$\mathcal{F}$ ↑
GCN	**w/ RPE**	**35.1**±0.5	**52.0**±0.4	**65.7**±0.5	**53.6**±0.9	**94.8**±0.1	**80.5**±0.4	74.1±1.2
	w/o RPE	13.9±0.5	35.2±0.6	46.7±0.5	86.5±2.7	91.1±0.1	77.1±14.4	**78.0**±2.6
GAT	**w/ RPE**	**36.8**±0.6	**52.3**±0.5	**67.0**±0.5	**51.1**±0.9	**95.0**±0.1	**81.6**±0.5	75.7±0.3
	w/o RPE	13.1±2.8	34.1±3.6	45.2±4.2	89.1±0.6	90.7±0.1	68.9±14.1	**77.4**±1.9
RGCN	**w/ RPE**	**38.0**±0.9	**54.1**±0.6	**67.6**±0.6	**49.3**±0.8	**95.2**±0.1	**81.9**±0.7	**75.9**±0.8
	w/o RPE	11.3±1.3	30.1±1.1	41.6±1.0	87.2±1.3	90.0±0.2	66.8±16.3	75.5±5.2

Table 3: *Graph Attack* - We corrupt the graph connectivity information given to the structural adapters in encoder. We report mean performances (±s.d.) over 3 seeds.

Adapter	Meaning Preservation	Linguistic Correctness
GCN w/ RPE	5.0±1.2	5.6±0.8
RGCN w/ RPE	5.2±1.1	5.6±0.8

Table 4: *Graph Attack* - Human Evaluation. Mean scores (±s.d.)

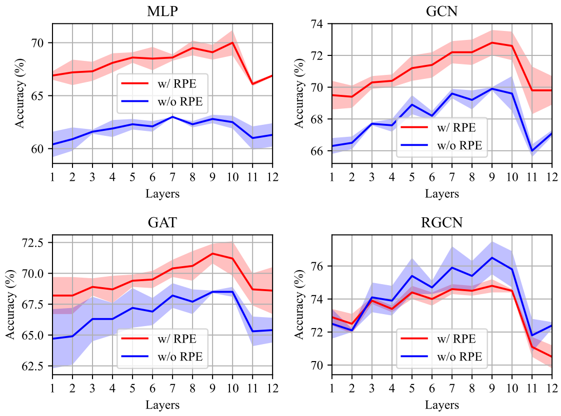


Figure 3: *Probing* - Link Prediction Results using hidden representations $\tilde{h}_i$ at different layers i.

Results are shown in Table 3. Human evaluation for *attacked* GCN and RGCN adapters is given in Table 4. As hypothesized, providing erroneous connectivity without any position embeddings makes structural adapters no longer compelling. We observe that StructAdapt with RGCN is significantly more affected compared to GAT and GCN based adapters. Since RGCN adds direction information for each edge (*direct* and *reverse*), we conjecture that RGCN is much more bewildered. Interestingly, using RPE with corrupted graph (Table 3) leads to similar performance than using graph information without RPE (Table 1). This strongly demonstrates the usefulness of RPE to carry out the generation. We additionally provide a *position attack* experiment in Appendix E where RPE are shuffled randomly. Accordingly, we can further identify the saliency of RPE despite the available GNN. This raises the question of RPE encoding the input graph.

4.2 Can the Graphs Be Reconstructed ?

As shown in Section 4.1, RPE seem to be as competitive as applying GNNs alone. If claiming that RPE also encode graphs is tempting, no strong evidence has been revealed. Indeed, better generation quality is not necessarily a consequence of better graph encoding. Therefore, we probe whether graphs can indeed be reconstructed from the learned hidden representations. To do so, we train a logistic regression, i.e. our probe, to perform link prediction as a binary classification. More specifically, given two nodes representations at a given layer l, our probe returns the probability that nodes are connected. To train our logistic regression model, we sample k positive connections, i.e. two connected nodes, and k negative connections, i.e. two non-connected nodes, for each sample of training and test sets.[2] For our experiment, we choose $k = 2$ which leads to 109,490 and 3,770 samples for each class for training and testing respectively. We plot results in Figure 3. Firstly, we observe very high accuracy for the vanilla adapter without RPE. Accuracies over 60% are easily reached while no structure encoding nor positions information are supplied. This might be a side effect of our probe training. Nevertheless, this gives a lower bound for our experiment. We can see that adding RPE increases the link prediction performance for MLP, GCN and GAT-based adapters. We observe a constant gap of about 3.5% on average. However, we remark that RGCN is able to reconstruct edges on its own with best accuracy. We report a maximum accuracy about 76% while other models are not reaching 73%. This strengthens the idea that giving information of reverse connection may add robustness to graph encoding as shown in (Beck et al., 2018). Generally, we observe that the deeper the representation, the better the link prediction. We notice however that after the 10^{th} layer, significant drops in link prediction arise regardless

---

[2] If k samples cannot be extracted for both positive and negative classes, the sample is discarded.

of adapter type. We assume representations should lose some information about the structure to perform language generation. This may indicate that encoded representation for Graph-to-Text is not just graph-centered. Although counter-intuitive, encoder representations given to the decoder part may not have to encode input graph efficiently in order to verbalize it. We leave this research question for future work. We further provide an analysis on self-attention matrices in Appendix F.

5 Conclusion

In this paper, we have explored the effect of relative position embeddings on AMR-to-Text generation using structural adapters. We have shown that the generation process could be enabled by relative position embeddings when structure is erroneous or missing. In addition, we have demonstrated the capacity of those representations to encode the input graph to some extent. We have further revealed interesting robustness of RGCN model in graph reconstruction ability. For future work, we believe further experiments on other pretrained models and Graph-to-Text tasks may shed more light on the role of position embeddings.

Limitations

A limitation of our study is that we focus on the T5 model only. Since adapters are additional modules to add, it is required to manually implement and directly modify the original code of the pretrained model which is not easily scalable. In addition, we only evaluate on the LDC2020T02 dataset which is the cleanest AMR dataset available.

Acknowledgements

We would like to thank annotators and reviewers for taking the time and effort necessary to share our contribution. This work was partially funded by the ANR Cifre conventions N°2020/0400 and Orange Innovation Research.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Kris Cao and Stephen Clark. 2019. Factorising AMR generation through syntax. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2157–2163, Minneapolis, Minnesota. Association for Computational Linguistics.

Marco Damonte and Shay B. Cohen. 2019. Structural neural encoders for AMR-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3649–3658, Minneapolis, Minnesota. Association for Computational Linguistics.

Philipp Dufter, Martin Schmitt, and Hinrich Schütze. 2022. Position information in transformers: An overview. *Computational Linguistics*, 48(3):733–763.

Cédric Fayet, Alexis Blond, Grégoire Coulombel, Claude Simon, Damien Lolive, Gwénolé Lecorvé, Jonathan Chevelu, and Sébastien Le Maguer. 2020. FlexEval, création de sites web légers pour des campagnes de tests perceptifs multimédias (FlexEval, creation of light websites for multimedia perceptual test campaigns). In *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Volume 4 : Démonstrations et résumés d'articles internationaux*, pages 22–25, Nancy, France. ATALA et AFCP.

Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Jeffrey Flanigan, Chris Dyer, Noah A. Smith, and Jaime Carbonell. 2016. Generation from Abstract Meaning Representation using tree transducers. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 731–739, San Diego, California. Association for Computational Linguistics.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Gerasimos Lampouras and Andreas Vlachos. 2017. Sheffield at SemEval-2017 task 9: Transition-based language generation from AMR. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 586–591, Vancouver, Canada. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Juri Opitz and Anette Frank. 2021. Towards a decomposable metric for explainable evaluation of text generation from AMR. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1504–1518, Online. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Nima Pourdamghani, Kevin Knight, and Ulf Hermjakob. 2016. Generating English from Abstract Meaning Representations. In *Proceedings of the 9th International Natural Language Generation conference*, pages 21–25, Edinburgh, UK. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. https://openai.com/blog/better-language-models/.

Aarne Ranta. 2011. *Grammatical Framework - Programming with Multilingual Grammars*. CSLI Studies in Computational Linguistics. Cambridge University Press.

Leonardo F. R. Ribeiro, Claire Gardent, and Iryna Gurevych. 2019. Enhancing AMR-to-text generation with dual graph representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3183–3194, Hong Kong, China. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021a. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021b. Structural adapters in pretrained language models for AMR-to-Text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4269–4282, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

Martin Schmitt, Leonardo F. R. Ribeiro, Philipp Dufter, Iryna Gurevych, and Hinrich Schütze. 2021. Modeling graph structure via relative position for text generation from knowledge graphs. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 10–21, Mexico City, Mexico. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Linfeng Song, Yue Zhang, Xiaochang Peng, Zhiguo Wang, and Daniel Gildea. 2016. AMR-to-text generation as a traveling salesman problem. In *Proceedings* *of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2084–2089, Austin, Texas. Association for Computational Linguistics.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. 2016. Neural headline generation on Abstract Meaning Representation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1054–1059, Austin, Texas. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. *6th International Conference on Learning Representations*.

Benyou Wang, Lifeng Shang, Christina Lioma, Xin Jiang, Hao Yang, Qun Liu, and Jakob Grue Simonsen. 2021. On position embeddings in {bert}. In *International Conference on Learning Representations*.

Yu-An Wang and Yun-Nung Chen. 2020. What do position embeddings learn? an empirical study of pre-trained language model positional encoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6840–6849, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yan Zhang, Zhijiang Guo, Zhiyang Teng, Wei Lu, Shay B. Cohen, Zuozhu Liu, and Lidong Bing. 2020b. Lightweight, dynamic graph convolutional networks for AMR-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2162–2172, Online. Association for Computational Linguistics.

Yanbin Zhao, Lu Chen, Zhi Chen, Ruisheng Cao, Su Zhu, and Kai Yu. 2020. Line graph enhanced AMR-to-text generation with mix-order graph attention networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 732–741, Online. Association for Computational Linguistics.

A Graph Neural Networks

A.1 Graph Convolutional Network (GCN)

To compute the node representation $g_u^l \in \mathbb{R}^{d_g}$ of the node u at layer l, GCN computes an aggregation of neighbors nodes embeddings as:

$$g_u^l = \sigma\left(\sum_{v \in \mathcal{N}(u)} \frac{1}{\deg(u) \times \deg(v)} W^l g_v^l\right) \quad (1)$$

with σ an activation function and $\deg(x)$ the degree of the node x.

A.2 Graph Attention Network (GAT)

GAT applies an attention mechanism to determine importance of neighboors nodes regarding current node u. We have:

$$\begin{aligned} g_u^l &= \sigma\left(\sum_{v \in \mathcal{N}(u)} \mathrm{softmax}(e_{u,v}) W^l g_v^l\right) \\ e_{u,v} &= \mathrm{LeakyReLu}\left(a_{u,v} \Big\Vert_{i=u}^{v} g_i^l\right) \end{aligned} \quad (2)$$

with both σ and LeakyReLu non-linear activation functions and $a_{u,v} \in \mathbb{R}^{2 \times d_g}$ a learnable vector.

A.3 Relational Graph Convolutional Network (RGCN)

RGCN takes into consideration the nature of the relation $r \in \mathcal{R}$ between nodes u and v. It performs

Adapter	Runtime (in hours)
MLP	21.5
GCN	15.6
GAT	22.0
RGCN	16.2

Table 5: *Runtime* - Runs are executed on a single NVIDIA GeForce RTX 3090 GPU.

convolution as the following:

$$g_u^l = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{v\in\mathcal{N}_r(u)} c_{u,r}W_r^l g_v^l\right) \qquad (3)$$

with $\mathcal{N}_r(u)$ the direct neighbors of node u under the relation r, $c_{u,r}$ a normalization term and $W_r^l \in \mathbb{R}^{d_g\times d_g}$ a learnable relation-dependent parameter.

B Training Setting

We used the version 3.3.1 of the HuggingFace's Transformers library (Wolf et al., 2020). We use Adam optimizer with a linearly decreasing learning rate, without warm-up. The initial learning rate is set to 1×10^{-4}. Batch size is set to 8. For decoding, a beam search of 5 is selected. A maximum length of 384 is used in case the *end-of-sentence* token is not encountered. We didn't use gradient clipping nor label smoothing. For GNNs, we make use of the version 1.7.2 of the PyTorch Geometric library (Fey and Lenssen, 2019). We only use a single layer for our GNNs networks, similarly to the vanilla adatper. For GAT, a single attention head was applied. The bottleneck dimension is set to 256 for all of our adapters. This is equivalent to about 4% only of the whole T5$_{base}$ model's parameters to train. The training time for MLP, GCN, GAT and RGCN adapters are given in Table 5. Note that the total runtime depends on the convergence as we are using early stopping.

C Automatic Evaluation

For evaluation, we consider multiple automated metrics. We employ popular n-gram-based metrics which are SacreBLEU (Post, 2018), METEOR (Banerjee and Lavie, 2005), chrf++ (Popović, 2017), TER (Och, 2003) and the semantic-based BertScore metric (Zhang et al., 2020a). In addition, we also report both $\mathcal{M}$ and $\mathcal{F}$ pillars of the decomposable $\mathcal{MF}$ score recently proposed by Opitz and Frank (2021). The meaning preservation, noted $\mathcal{M}$, measures how close the AMR of the generated sentence is to the reference sentence. To do so, both the generated sentence and target sentence are parsed with a pretrained Text-to-AMR model from Cai and Lam (2020). Then, their AMRs are compared using graph-based similarity measures such as Smatch (Cai and Knight, 2013). In contrast, the grammatical form, noted $\mathcal{F}$, measures the linguistic quality of the generated text using state-of-the-art language model (Radford et al., 2019). We also report human evaluation in Section 4.1 to accurately assess the quality of the generated outputs. We ask annotators to evaluate the meaning preservation and linguistic correctness of our generated outputs compared to the references. Details of human evaluation are given in Appendix D.

D Human Evaluation

Since handcrafted annotation is extremely costly, we limited the number of samples and models to assess. We randomly selected 30 test samples per model to evaluate manually. We further limited evaluation to 8 configurations of adapters. This led us with 240 samples to score by multiple annotators. We asked 22 annotators to answer to these following questions with a rate on a 1-6 Likert scale, with 6 the best score:

- Is the generated sentence semantically close to the reference ?
- Is the generated sentence grammatically correct ?

Each annotator was given 50 samples to annotate. We adopted FlexEval (Fayet et al., 2020) to implement a flexible evaluation environment. This allows a streaming evaluation were annotators can stop at any moment and come back to the evaluation later, at one's will. Furthermore, different samples from different models are given to annotators in an balanced manner such that each sample of each model is exposed to at least 2 annotators. Note that one annotator is given some samples from multiple configuration. This avoids evaluation bias toward a single annotator. In our case, each sample from each configuration has been annotated by 4 distinct annotators.

E Position Attack

For a given graph, we also propose to corrupt RPE with a random shuffle. Since the model may learn the generated order through training, we *systematically* shuffle RPE for each epoch for all graphs. Results are given in Table 6.

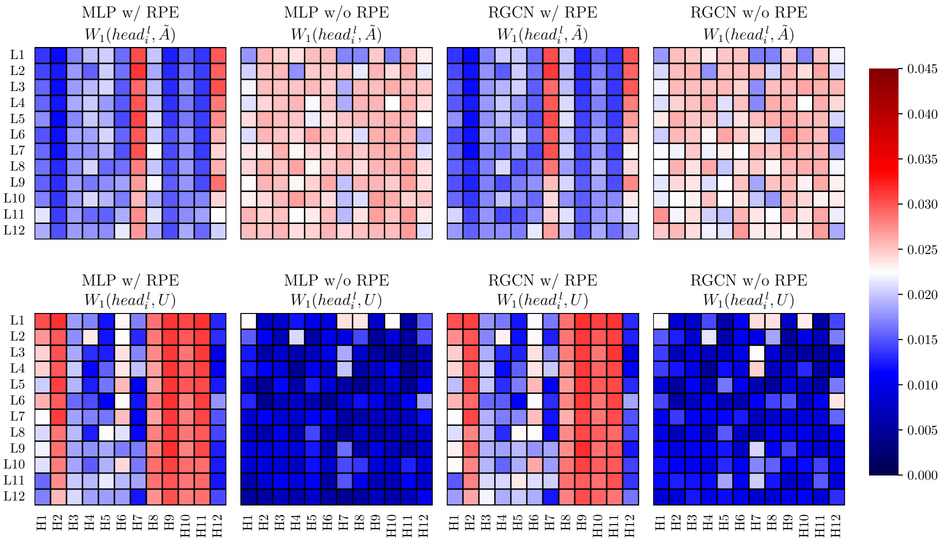


Figure 4: 1-Wasserstein distances of attention distributions for each head (columns) at each layer (rows) from both normalized adjacency matrices (top) and uniform distribution (down). The bluer, the lower the distance.

Adapter	**BLEU**
MLP	19.0±0.6
GCN	28.1±0.6
GAT	28.9±2.3
RGCN	**35.4**±1.2

Table 6: *Position Attack* - Relative Position Embeddings are shuffled.

F Analysis of Attention Matrices

The self-attention matrix gives information about how much weight should be assigned to other tokens (i.e. nodes). Thus, attention matrix can be seen as a pseudo-adjacency matrix. We further propose to inspect the similarity between the attention matrix of each head at each layer with the adjacency matrix of the input graph. Since the input is considered as a sequence of nodes, the shape of the attention matrix and adjacency matrix are equal. However, a limitation of the similarity comparison lies in the respective type of those mathematical objects. Attention matrices are probabilities whereas adjacency matrices are not. To deal with this issue, we convert the adjacency matrix A^s of a sample s to a normalized form $\tilde{A}^s$ such that each row $\tilde{A}^s_n$ is normalized and sums up to one, as described in Eq. 4.

$$A^s = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \quad \tilde{A}^s = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

This enables the rows of both attention and adjacency matrices to be probabilities distributions over input nodes. We can therefore measure the distance between those distributions to find out whether attention matrices are somehow close to the (normalized) adjacency matrix, and thus encoding graph connections. To do so, we compute the 1-Wasserstein distance W_1 [3] between the attention distribution of each token n and its corresponding normalized n^{th} row of $\tilde{A}$. We average distances over tokens, and then over samples (Eq. 5). We note $\tilde{A}$, S, and N_s the adjacency matrices, the total number of AMR samples and the length of sequence s, respectively.

$$W_1(\text{head}_i^l, \tilde{A}) = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{N_s}\sum_{n=1}^{N_s} W_1(\text{head}_{i,n}^{l,s}, \tilde{A}^s_n) \tag{5}$$

For fair comparison, we also provide the distance between attention distribution head_i^l and the uniform distribution U which assigns a probability of $\frac{1}{N}$ to each token for a graph with N nodes. We plot distances as heatmaps on Figure 4 where each square indicates either $W_1(\text{head}_i^l, \tilde{A})$ or $W_1(\text{head}_i^l, U)$. When using RPE, we can see that, for both MLP and RGCN-based adapters, the distribution of the attention scores are close to the normalized adjacency matrices of our graphs. Meanwhile, when removing them, we can witness

---

[3] Equivalent to earth mover's distance.

that attention scores are much closer to the uniform distribution and thus smoother. This is in line with our previous results suggesting that RPE might partially encode graphs. Since RPE are shared across layers, we can easily detect similar distances across layers for same heads (visible vertical lines). We also find similar behavior we have previously noted where last layers tend to lose graph information as attention scores slightly move away from the normalized adjacency matrix distribution.

On the Intersection of Context-Free and Regular Languages

Clemente Pasti[ŋ,ʕ] **Andreas Opedal**[ʕ] **Tiago Pimentel**[ð]
Tim Vieira[ᴅ] **Jason Eisner**[ᴅ] **Ryan Cotterell**[ʕ]
[ŋ]Università della Svizzera Italiana [ʕ]ETH Zürich
[ð]University of Cambridge [ᴅ]Johns Hopkins University
clemente.pasti@usi.ch andreas.opedal@inf.ethz.ch tp472@cam.ac.uk
tim.f.vieira@gmail.com jason@cs.jhu.edu ryan.cotterell@inf.ethz.ch

Abstract

The Bar-Hillel construction is a classic result in formal language theory. It shows, by a simple construction, that the intersection of a context-free language and a regular language is itself context-free. In the construction, the regular language is specified by a finite-state automaton. However, neither the original construction (Bar-Hillel et al., 1961) nor its weighted extension (Nederhof and Satta, 2003) can handle finite-state automata with ε-arcs. While it is possible to remove ε-arcs from a finite-state automaton efficiently without modifying the language, such an operation modifies the automaton's set of paths. We give a construction that generalizes the Bar-Hillel in the case where the desired automaton has ε-arcs, and further prove that our generalized construction leads to a grammar that encodes the structure of both the input automaton and grammar while retaining the asymptotic size of the original construction.

https://github.com/rycolab/bar-hillel

1 Introduction

Bar-Hillel et al.'s (1961) construction—together with its weighted generalization (Nederhof and Satta, 2003)—is a fundamental result in formal language theory. Given a weighted context-free grammar (WCFG) $\mathcal{G}$ and a weighted finite-state automaton (WSFA) $\mathcal{A}$, the Bar-Hillel construction yields another WCFG $\mathcal{G}_\cap$ whose language $L(\mathcal{G}_\cap)$ is equal to the intersection of $L(\mathcal{G})$ with $L(\mathcal{A})$. Importantly, the Bar-Hillel construction directly proves that weighted context-free languages are closed under intersection with weighted regular languages. The construction was later extended to other formalisms, e.g., tree automata (Maletti and Satta, 2009), synchronous tree substitution grammars (Maletti, 2010) and linear context-free re-writing systems (Seki et al., 1991; Nederhof and Satta, 2011b). Furthermore, the Bar-Hillel construction has seen applications in the computation of infix probabilities (Nederhof and Satta, 2011a) and human sentence comprehension (Levy, 2008, 2011).

Unfortunately, Bar-Hillel et al.'s construction, as well as its weighted generalization by Nederhof and Satta, requires the input automaton to be ε-free.[1] Although any WFSA can be converted to a weakly equivalent[2] ε-free WFSA using well-known techniques (Mohri, 2001, 2002; Hanneforth and de la Higuera, 2010), such an approach adds an additional step of computation, typically increases the size of the output grammar $\mathcal{G}_\cap$, and does not, in general, maintain a bijection between derivations in $\mathcal{G}_\cap$ and the Cartesian product of the derivations in $\mathcal{G}$ and paths in $\mathcal{A}$. In other words, $\mathcal{G}_\cap$ is *not* strongly equivalent to the product of $\mathcal{G}$ and $\mathcal{A}$.[3]

In this note, we generalize the classical Bar-Hillel construction to the case where the automaton we seek to intersect with the grammar has ε-arcs. Our new construction produces a WCFG $\mathcal{G}_\cap$ that is strongly equivalent to the product of $\mathcal{G}$ and $\mathcal{A}$. We further generalize the Bar-Hillel construction to work with arbitrary commutative semirings. Finally, we give an asymptotic bound on the size of the resulting grammar and a detailed proof of correctness in the appendix.

2 Languages, Automata, and Grammars

As background, we now give formal definitions of semirings, weighted formal languages, finite-state automata, and context-free grammars.

2.1 Semirings

Semirings are useful algebraic structures for describing weighted languages (Droste et al., 2009, Chapter 1). In order to define semirings we must first give the definition of a monoid. A **monoid** is a 3-tuple $\mathcal{M} = (A, \bullet, \mathbf{1})$, where A is a set,

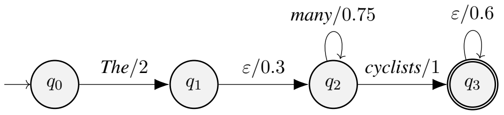
---

[1] But they do not require the input grammar to be ε-free.

[2] Two WFSAs are said to be weakly equivalent if they represent the same weighted formal language.

[3] Strong equivalence is formally defined in Definition 6 and Theorem 1.

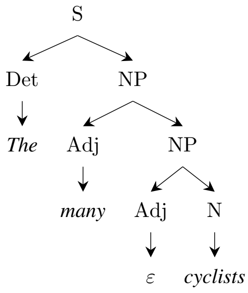
Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 737–749
May 2-6, 2023 ©2023 Association for Computational Linguistics



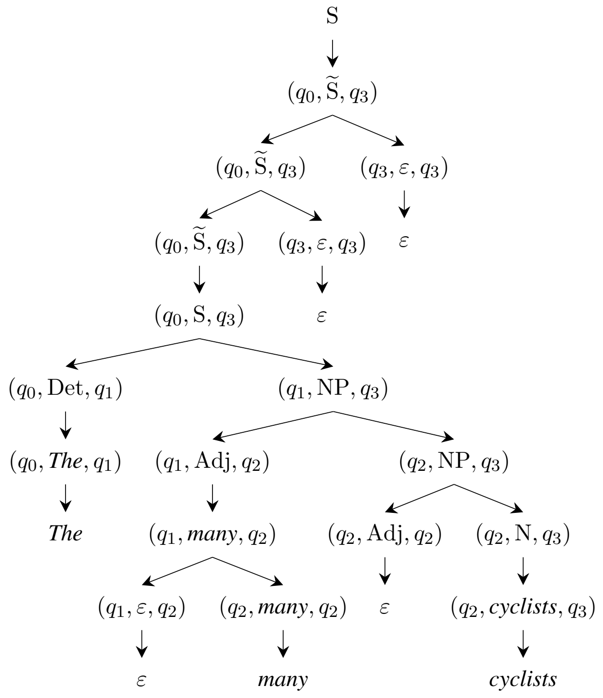
(a) Weighted finite-state automaton

S → Det NP	1
NP → Adj N	1
NP → Adj NP	0.5
N → *cyclists*	2
Adj → *many*	2
Adj → ε	1
Det → *The*	2

(b) Weighted context-free grammar



(c) Original derivation



(d) Derivation in the intersection grammar

Figure 1: Example of a derivation in the grammar obtained as the intersection of the finite-state automaton (a) and the context-free grammar (b). The derivation tree (d) encodes the derivation tree (c) in the original grammar, and path $q_0 \xrightarrow{\textit{The}/2} q_1 \xrightarrow{\varepsilon/0.3} q_2 \xrightarrow{\textit{many}/0.75} q_2 \xrightarrow{\textit{cyclists}/1} q_3 \xrightarrow{\varepsilon/0.6} q_3 \xrightarrow{\varepsilon/0.6} q_3$. We use rules from Eq. (5g) for ε-arcs appearing before an input symbol, and rules from Eq. (5b) for ε-arcs appearing at the end of the input.

$\bullet : A \times A \to A$ is an associative operator, and $\mathbf{1} \in A$ is a distinguished identity element such that $\mathbf{1} \bullet w = w \bullet \mathbf{1} = w$ for any $w \in A$. We say that a monoid is *commutative* if $\bullet$ commutes, i.e., $w_1 \bullet w_2 = w_2 \bullet w_1$ for any $w_1, w_2 \in A$. We can now give the definition of a semiring.

Definition 1. *A* ***semiring*** $\mathcal{W} = (A, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ *is a 5-tuple where* $(A, \oplus, \mathbf{0})$ *is a commutative monoid,* $(A, \otimes, \mathbf{1})$ *is a monoid,* $\otimes$ *distributes over* $\oplus$*, and* $\mathbf{0}$ *is an* annihilator *for* $\otimes$*, meaning that* $\mathbf{0} \otimes w = w \otimes \mathbf{0} = \mathbf{0}$ *for any* $w \in A$.

We say that $\mathcal{W}$ is commutative if $\otimes$ commutes. In this work, we assume commutative semirings.

2.2 Weighted Formal Languages

This paper concerns itself with transforms between devices that generate weighted formal languages.

Definition 2. *Let* Σ *be an alphabet and* $\mathcal{W} = (A, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ *be a semiring. Then a* ***weighted formal language*** $L : \Sigma^* \to A$ *is a mapping from the Kleene closure of* Σ *to the set of weights* A*. Furthermore, the set* $\operatorname{supp}(L) = \{\boldsymbol{s} \in \Sigma^* \mid L(\boldsymbol{s}) \neq \mathbf{0}\}$ *is called the language's* ***support****.*

Unweighted formal languages (e.g., Sipser, 2006; Hopcroft et al., 2006) are simply the special case of Definition 2 where $\mathcal{W}$ is the boolean semiring. In this note, we discuss algorithms for computing the intersection of two weighted formal languages.[4]

Definition 3. *Let* L_1 *and* L_2 *be two weighted formal languages over the same alphabet* Σ *and the same semiring* $\mathcal{W}$*. The* ***intersection*** *of* L_1 *with* L_2 *is defined as the weighted language*

$$(L_1 \cap L_2)(\boldsymbol{s}) \overset{\text{def}}{=} L_1(\boldsymbol{s}) \otimes L_2(\boldsymbol{s}), \quad \forall \boldsymbol{s} \in \Sigma^* \quad (1)$$

Specifically, this paper concerns itself with the special case of Definition 3 when L_1 is a weighted context-free language (represented by a WCFG), and L_2 is a weighted regular language (represented by a WFSA); we define these two formalisms in the subsequent sections.

In the following, the symbol ε always represents the empty string.

---

[4] The intersection of two weighted languages is also called their Hadamard product (Droste et al., 2009, Chapter 1).

2.3 Weighted Finite-State Automata

We now review the basics of weighted finite-state automata (WFSA), which provide a formalism to represent weighted regular languages.

Definition 4. *A* ***weighted finite-state automaton*** *$\mathcal{A}$ over a semiring $\mathcal{W} = (A, \oplus, \otimes, \boldsymbol{0}, \boldsymbol{1})$ is a 6-tuple $(\Sigma, Q, \delta, \lambda, \rho, \mathcal{W})$. In this tuple, Σ is an alphabet, Q is a finite set of states, and $\delta \subseteq Q \times Q \times (\Sigma \cup \{\varepsilon\}) \times A$ is a finite multi-set of weighted arcs. Further, $\lambda : Q \to A$ and $\rho : Q \to A$ are the initial and final weight functions, respectively. We also define the sets $Q_I = \{q \mid q \in Q,\ \lambda(q) \neq \boldsymbol{0}\}$ and $Q_F = \{q \mid q \in Q,\ \rho(q) \neq \boldsymbol{0}\}$ for convenience.*

We will represent an arc in δ with the notation $q_0 \xrightarrow{a/w} q_1$ where $a \in \Sigma \cup \{\varepsilon\}$ and $w \in A$. A **path** $\boldsymbol{\pi}$ (of length $N > 0$) is a sequence of arcs in δ^* where the states of adjacent arcs are matched, i.e.,

$$q_0 \xrightarrow{a_1/w_1} \cdots q_{n-1} \xrightarrow{a_n/w_n} q_n \cdots \xrightarrow{a_N/w_N} q_N \qquad (2)$$

and where $q_0 \in Q_I$ and $q_N \in Q_F$, i.e., the path starts at an initial state and ends at a final state. The path's **yield**, denoted $\texttt{yield}(\boldsymbol{\pi})$, is the concatenation $a_1 a_2 \cdots a_N$ of all its arc labels (strings of length ≤ 1). The path's **weight**, denoted $w(\boldsymbol{\pi})$, is the product

$$w(\boldsymbol{\pi}) = \lambda(q_0) \otimes \left(\bigotimes_{n=1}^{N} w_n \right) \otimes \rho(q_N) \qquad (3)$$

We denote the set of all paths in $\mathcal{A}$ as $\mathcal{D}_{\mathcal{A}}$, and the set of all paths with yield $\boldsymbol{s}$ as $\mathcal{D}_{\mathcal{A}}(\boldsymbol{s})$. Finally, we define the **language of an automaton** as the mapping $L_{\mathcal{A}} : \Sigma^* \to A$ where we have[5] $L_{\mathcal{A}}(\boldsymbol{s}) = \bigoplus_{\boldsymbol{\pi} \in \mathcal{D}_{\mathcal{A}}(\boldsymbol{s})} w(\boldsymbol{\pi})$. The set of languages that can be encoded by a WFSA forms the class of **weighted regular languages**.

2.4 Weighted Context-Free Grammars

We now go over the necessary background on weighted context-free grammars (WCFGs).

Definition 5. *A* ***weighted context-free grammar*** *is a tuple $\mathcal{G} = (\mathcal{N}, \Sigma, \mathcal{W}, \mathrm{S}, \mathcal{P})$, where $\mathcal{N}$ is a non-empty set of nonterminal symbols, Σ is an alphabet of terminal symbols, $\mathcal{W} = (A, \oplus, \otimes, \boldsymbol{0}, \boldsymbol{1})$ is a semiring, $\mathrm{S} \in \mathcal{N}$ is a distinguished start symbol, and $\mathcal{P}$ is a set of production rules. Each rule $p \in \mathcal{P}$ is of the form $\mathrm{X} \xrightarrow{w} \boldsymbol{\alpha}$, with $\mathrm{X} \in \mathcal{N}$, $w \in A$, and $\boldsymbol{\alpha} \in (\Sigma \cup \mathcal{N})^*$.*

Given two strings $\boldsymbol{\alpha}, \boldsymbol{\beta} \in (\Sigma \cup \mathcal{N})^*$, we write $\boldsymbol{\alpha} \overset{p}{\Rightarrow}_L \boldsymbol{\beta}$ if and only if we can express $\boldsymbol{\alpha} = \boldsymbol{z}\,\mathrm{X}\,\boldsymbol{\delta}$ and $\boldsymbol{\beta} = \boldsymbol{z}\,\boldsymbol{\gamma}\,\boldsymbol{\delta}$ where $\boldsymbol{z} \in \Sigma^*$ and $p \in \mathcal{P}$ is the rule $\mathrm{X} \xrightarrow{w} \boldsymbol{\gamma}$. A **derivation** $\boldsymbol{d}$ (more precisely, a leftmost derivation) is a sequence $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_N$ with $N > 0$, $\boldsymbol{\alpha}_0 = \mathrm{S}$, and $\boldsymbol{\alpha}_N \in \Sigma^*$, such that for all $0 < n \leq N$, we have $\boldsymbol{\alpha}_{n-1} \overset{p_n}{\Rightarrow}_L \boldsymbol{\alpha}_n$ for some (necessarily unique) $p_n \in \mathcal{P}$. The derivation's **yield**, $\texttt{yield}(\boldsymbol{d})$, is $\boldsymbol{\alpha}_N$, and its weight, $w(\boldsymbol{d})$, is $w(p_1) \otimes \cdots \otimes w(p_N)$. We denote the set of derivations under a grammar $\mathcal{G}$ as $\mathcal{D}_{\mathcal{G}}$ and the set of all derivations with yield $\boldsymbol{s}$ as $\mathcal{D}_{\mathcal{G}}(\boldsymbol{s})$. Finally, we define the **language of a grammar** as $L_{\mathcal{G}}$ where[5] $L_{\mathcal{G}}(\boldsymbol{s}) \overset{\text{def}}{=} \bigoplus_{\boldsymbol{d} \in \mathcal{D}_{\mathcal{G}}(\boldsymbol{s})} w(\boldsymbol{d}), \forall \boldsymbol{s} \in \Sigma^*$. The languages that can be encoded by a WCFG are known as **weighted context-free languages**.

3 Generalizing Bar-Hillel

Given any context-free grammar (CFG) $\mathcal{G}$ and finite-state automaton (FSA) $\mathcal{A}$, Bar-Hillel et al. (1961) showed how to construct a CFG $\mathcal{G}_\cap$ such that $L_{\mathcal{G}_\cap} = L_{\mathcal{G}} \cap L_{\mathcal{A}}$. Later, Nederhof and Satta (2003) generalized Bar-Hillel's construction to work on a *weighted* context-free grammar and a *weighted* finite-state automaton. While they focused on the real semiring, their construction actually works for any commutative semiring. However, neither of these versions correctly computes the intersection when the WFSA (or FSA) contains ε-arcs. Yet, in several applications—such as modeling noisy inputs for human sentence comprehension (Levy, 2008, 2011)—we may be interested in using a WFSA $\mathcal{A}$ that contains ε-arcs. A naïve application of the construction would ignore paths in $\mathcal{A}$ that contain ε-arcs. The problem may be sidestepped by transforming $\mathcal{A}$ into a weakly equivalent ε-free WFSA[6] before applying the construction;[7] this, however, might increase the size of the WFSA and of the intersection grammar, and it would not allow us to identify the paths in the input WFSA that yield a target string in the

---

[5] In the main paper we gloss over the question of how $\bigoplus$-summations over *infinite* sets are to be defined (or left undefined), but we treat this issue in App. B.2.

[6] See footnote 2 for the definition of weak equivalence.

[7] Levy (2008, 2011) uses WFSAs to model the degree of uncertainty under which a human comprehends a particular sentence, in which ε-arcs are used to represent word deletion. He applies the Bar-Hillel construction to compute the intersection of the language represented by the WFSA and the language encoded by a WCFG that represents the comprehender's grammatical knowledge, in order to obtain a joint posterior distribution over parses and words. While he transforms $\mathcal{A}$ to eliminate ε-arcs prior to applying the Bar-Hillel construction (Levy, p.c.), the solution we propose here is an alternative.

intersection grammar.[8]

3.1 The problem with ε-arcs

Before proposing our solution, we explain how the original construction works, and how it fails in the case of ε-arcs. Given a WFSA $\mathcal{A} = (\Sigma, Q, \delta, \lambda, \rho, \mathcal{W})$ and a WCFG $\mathcal{G} = (\mathcal{N}, \Sigma, \mathcal{W}, \mathrm{S}, \mathcal{P})$ over the same alphabet Σ and commutative semiring $\mathcal{W}$, their intersection $\mathcal{G}_\cap$ is defined by the tuple $(\mathcal{N}_\cap, \Sigma, \mathcal{W}, \mathrm{S}, \mathcal{P}_\cap)$, where:

- The set of nonterminal symbols $\mathcal{N}_\cap = \{\mathrm{S}\} \cup Q \times (\mathcal{N} \cup \Sigma) \times Q$ contains the triplets (q_i, X, q_j) plus the start symbol S.[9]
- The set of production rules $\mathcal{P}_\cap$ is given by the equations in Construction 1 of Fig. 2.[10]
- Σ, $\mathcal{W}$, S are the same as in the input grammar.

The intuition behind this construction is that a derivation in the intersection grammar encodes both a path in the input WFSA and a derivation in the input WCFG with matching yield. Specifically, rules (4f) encode arcs in the WFSA and rules (4d) encode production rules in the WCFG. Rules (4e) handle the special case of ε-productions in the input WCFG and rules (4a) are designed to take into account the initial and final weight of a path. These rules may combine through matching nonterminals to permit derivations in the intersection grammar $\mathcal{G}_\cap$.

Unfortunately, this mechanism breaks in the presence of ε-arcs. Although the rules (4f) do construct nonterminals for ε-arcs (when $a = \varepsilon$), the rules (4d) never generate those nonterminals (since the X_m on the right-hand side of a rule are never ε). We show this with an example. Consider the automaton and the grammar in Fig. 1, both of which assign non-zero weight to the string *The many cyclists*. However, their intersection computed with the Bar-Hillel construction is empty. To see this, note that all the paths from q_0 to q_3 contain the arc $q_1 \xrightarrow{\varepsilon/0.3} q_2$. Eq. (4f) will create a rule $(q_1, \varepsilon, q_2) \xrightarrow{0.3} \varepsilon$, but none of the rules produced by Eqs. (4d) and (4e) has the triplet (q_1, ε, q_2) on the right hand side. This misalignment results in an empty set of derivations in $\mathcal{G}_\cap$. In App. A we describe more failure cases in a detailed manner.

3.2 Our generalized construction

We now describe an improved version of the Bar-Hillel construction that handles ε-arcs in the WFSA. In comparison to the original construction, our version of $\mathcal{G}_\cap = (\mathcal{N}_\cap, \Sigma, \mathcal{W}, \mathrm{S}, \mathcal{P}_\cap)$ has

- $\mathcal{N}_\cap = \{\mathrm{S}\} \cup Q \times (\mathcal{N} \cup \{\widetilde{\mathrm{S}}\} \cup \Sigma) \times Q$ as the set of nonterminals, where $\widetilde{\mathrm{S}}$ is a new symbol;
- $\mathcal{P}_\cap$ as the augmented set of production rules given in Construction 2 of Fig. 2.

Our generalized construction adds additional production rules that traverse the ε-arcs. Rules (5g) can traverse a WFSA subpath labeled with $\varepsilon^* a$ to yield a terminal symbol $a \in \Sigma$. At the end of the yielded string, rules (5b) can traverse a WFSA subpath labeled with ε^* that ends at a final state q_F. Our construction carefully avoids overcounting[11] by ensuring that each matching pair of an $\mathcal{A}$-path and a $\mathcal{G}$-derivation of its string corresponds to *exactly one* $\mathcal{G}_\cap$-derivation of that string, as illustrated in Fig. 1. Note that rules (5d) to (5f) are identical to their counterparts in the original construction. Rules (5a) are a modified version of rules (4a) with the special start symbol $\widetilde{\mathrm{S}}$; this allows our construction to handle ε-arcs immediately before the final state—by repeated applications of rule (5b)—before switching $\widetilde{\mathrm{S}}$ back to S with rule (5c). In App. A we illustrate the mechanism with examples.

We now state the theorem of correctness.

Definition 6. *Let Σ be an alphabet and $\mathcal{W}$ be a commutative semiring. Let $\mathcal{G}$ be a WCFG and $\mathcal{A}$ be a WFSA—both over Σ and $\mathcal{W}$. The* ***weighted join*** *of the derivations in $\mathcal{D}_\mathcal{G}$ with the paths in $\mathcal{D}_\mathcal{A}$ is defined as:*

$$(\mathcal{D}_\mathcal{G} \bowtie \mathcal{D}_\mathcal{A}) \overset{\text{def}}{=} \Big\{ \langle \boldsymbol{d}, \boldsymbol{\pi} \rangle \mid \boldsymbol{d} \in \mathcal{D}_\mathcal{G}, \boldsymbol{\pi} \in \mathcal{D}_\mathcal{A} \quad \textit{s.t.}\ \texttt{yield}(\boldsymbol{d}) = \texttt{yield}(\boldsymbol{\pi}) \Big\} \tag{6}$$

with $w(\langle \boldsymbol{d}, \boldsymbol{\pi} \rangle) = w(\boldsymbol{d}) \otimes w(\boldsymbol{\pi})$.

---

[8] In contrast, this is easy under our construction. Each derivation of the target string under $\mathcal{G}_\cap$ uses a particular path in $\mathcal{A}$. To reconstruct that path, ε-arcs and all, simply traverse from left to right the leaves of the derivation tree (e.g., Fig. 1d) and list the states on the triplets where rule (5f) is applied.

[9] Many of the nonterminals will turn out to be **useless** in that they do not participate in any derivation in $\mathcal{D}_{\mathcal{G}_\cap}$. These can be pruned from the grammar along with all rules that mention them (Hopcroft et al., 2006).

[10] Note that this construction can handle multiple initial and final states, whereas Nederhof and Satta's (2003) construction assumes a WFSA with a single initial and a single final state. A path's initial and final weights are taken into account by the weight of rules (4a) of Construction 1 in Fig. 2.

[11] As Fig. 1d illustrates, we do this by introducing a single, right-branching subderivation for each $\mathcal{A}$-subpath $\varepsilon^* a$ that matches an input symbol a. A nonterminal of the form (q_0, ε, q_1) is never used as a right child, nor does it ever combine with a nonterminal of the form (q_0, X, q_1), except at the end of the input, which is specially handled by rules (5b). Similarly, Allauzen et al. (2010) avoid overcounting when intersecting or composing finite-state machines that have ε-arcs.

Construction 1

$$\mathrm{S} \xrightarrow{\lambda(q_I)\otimes\rho(q_F)} (q_I, \mathrm{S}, q_F) \tag{4a}$$
$$\forall q_I \in I, \forall q_F \in F$$

$$(q_0, \mathrm{X}, q_M) \xrightarrow{w} (q_0, \mathrm{X}_1, q_1) \cdots (q_{M-1}, \mathrm{X}_M, q_M)$$
$$\forall(\mathrm{X} \xrightarrow{w} \mathrm{X}_1 \cdots \mathrm{X}_M) \in \mathcal{P}, M > 0 \tag{4d}$$
$$\forall q_0, \ldots, q_M \in Q$$

$$(q_0, \mathrm{X}, q_0) \xrightarrow{w} \varepsilon \tag{4e}$$
$$\forall(\mathrm{X} \xrightarrow{w} \varepsilon) \in \mathcal{P}, \forall q_0 \in Q$$

$$(q_0, a, q_1) \xrightarrow{w} a \tag{4f}$$
$$\forall(q_0 \xrightarrow{a/w} q_1) \in \delta$$

Construction 2

$$\mathrm{S} \xrightarrow{\lambda(q_I)\otimes\rho(q_F)} (q_I, \widetilde{\mathrm{S}}, q_F) \tag{5a}$$
$$\forall q_I \in I, \forall q_F \in F$$

$$(q_I, \widetilde{\mathrm{S}}, q_1) \xrightarrow{\mathbf{1}} (q_I, \widetilde{\mathrm{S}}, q_0)(q_0, \varepsilon, q_1) \tag{5b}$$
$$\forall q_I \in I, \forall q_0, q_1 \in Q$$

$$(q_I, \widetilde{\mathrm{S}}, q_0) \xrightarrow{\mathbf{1}} (q_I, \mathrm{S}, q_0) \tag{5c}$$
$$\forall q_I \in I, \forall q_0 \in Q$$

$$(q_0, \mathrm{X}, q_M) \xrightarrow{w} (q_0, \mathrm{X}_1, q_1) \cdots (q_{M-1}, \mathrm{X}_M, q_M)$$
$$\forall(\mathrm{X} \xrightarrow{w} \mathrm{X}_1 \cdots \mathrm{X}_M) \in \mathcal{P}, M > 0 \tag{5d}$$
$$\forall q_0, \ldots, q_M \in Q$$

$$(q_0, \mathrm{X}, q_0) \xrightarrow{w} \varepsilon \tag{5e}$$
$$\forall(\mathrm{X} \xrightarrow{w} \varepsilon) \in \mathcal{P}, \forall q_0 \in Q$$

$$(q_0, a, q_1) \xrightarrow{w} a \tag{5f}$$
$$\forall(q_0 \xrightarrow{a/w} q_1) \in \delta, a \in \Sigma \cup \{\varepsilon\}$$

$$(q_0, a, q_2) \xrightarrow{\mathbf{1}} (q_0, \varepsilon, q_1)(q_1, a, q_2) \tag{5g}$$
$$\forall a \in \Sigma, \forall q_0, q_1, q_2 \in Q$$

Figure 2: The original Bar-Hillel construction (left) and our generalized version (right) that covers ε-arcs. We highlight the differences from the original construction in red. Note that the weights of rules (4a) and (4f) (respectively (5a) and (5f)) encode the weights of the WFSA, while the weights of rules (4d) and (4e) (respectively (5d) and (5e)) encode weights of the WCFG. All other rules in the generalized construction ((5b), (5c) and (5g)) are assigned weight **1**, and, thus, they do not change the weight of a derivation.

Theorem 1. *Let $\mathcal{G}$ be a WCFG and $\mathcal{A}$ a WFSA over the same alphabet Σ and commutative semiring $\mathcal{W}$. Let $\mathcal{G}_\cap$ be the grammar obtained with our generalized construction. Then we have strong equivalence between $\mathcal{G}_\cap$ and $\langle \mathcal{G}, \mathcal{A} \rangle$; meaning that there is a weight-preserving, yield-preserving bijection between $\mathcal{D}_{\mathcal{G}_\cap}$ and $(\mathcal{D}_\mathcal{G} \bowtie \mathcal{D}_\mathcal{A})$.*

Corollary 1. *$\mathcal{G}_\cap$ and $\langle \mathcal{G}, \mathcal{A} \rangle$ are weakly equivalent, meaning that $L_{\mathcal{G}_\cap}(\boldsymbol{s}) = L_\mathcal{G}(\boldsymbol{s}) \otimes L_\mathcal{A}(\boldsymbol{s})$ whenever the values on the right-hand side are defined.*

See App. B for proofs. Theorem 1 may be seen as a generalization of Theorem 8.1 by Bar-Hillel et al. (1961) and Theorem 12 by Nederhof and Satta (2003). Indeed, the set of derivations produced by Construction 1 is equivalent to the set of derivations produced by Construction 2, modulo an unfold transform (Tamaki and Sato, 1984) to remove rules containing $\widetilde{\mathrm{S}}$. Among the groups of rules listed in Fig. 2, the set of rules with maximum cardinality is the one defined by Eq. (5d). This set has cardinality $\mathcal{O}\left(|\mathcal{P}||Q|^{M_\star}\right)$, where $M_\star$ is 1 plus the length of the longest right-hand side among all the rules $\mathcal{P}$. All other equations in this construction lead to smaller sets of added rules. Since Eq. (5d) is unchanged from Eq. (4d) in the original construction, the asymptotic bound on the number of rules in our output grammar remains unchanged.

4 Conclusion

We generalized the weighted Bar-Hillel intersection construction so that the given WFSA may contain ε-arcs. Our construction is strongly equivalent to the product of the original WCFG and WFSA, i.e., every derivation tree in the resulting grammar represents a pairing of a derivation tree in the input WCFG and a path in the WFSA with the same yield. We gave a full proof of correctness for our construction. By adding output strings to the WFSA arcs and having rule (5f) rewrite to the arc's output string, our method can also be used to compose a WCFG with a weighted finite-state *transducer* (WFST) that could usefully model morphological post-processing or speaker errors.

5 Acknowledgements

The authors acknowledge Roger Levy for correspondence about Levy (2008) and Levy (2011).

6 Limitations

In this note, we generalize a fundamental theoretical result in formal language theory, which has seen a variety of practical applications, including human sentence comprehension under uncertain input (Levy, 2008, 2011) and infix probability computation (Nederhof and Satta, 2003). Although we motivate our paper by discussing the necessity of performing intersections on automata with ε-arcs, we do not explore any such practical applications. Further, while we show that the asymptotic bound on the size of our intersection grammar matches the original Bar-Hillel construction's, we do not discuss multiplicative or added constants introduced in our grammar's size.

Ethical Statement

We do not foresee any ethical issues with our work.

References

Cyril Allauzen, Michael Riley, and Johan Schalkwyk. 2010. Filters for efficient composition of weighted finite-state transducers. In *Proceedings of the 15th International Conference on Implementation and Application of Automata*, International Conference on Implementation and Application of Automata, page 28–38, Berlin, Heidelberg. Springer-Verlag.

Yehoshua Bar-Hillel, M. Perles, and E. Shamir. 1961. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*, 14:143–172. Reprinted in Y. Bar-Hillel. (1964). *Language and Information: Selected Essays on their Theory and Application*, Addison-Wesley 1964, 116–150.

Manfred Droste, Werner Kuich, and Heiko Vogler. 2009. *Handbook of Weighted Automata*. Springer Berlin, Heidelberg.

Thomas Hanneforth and Colin de la Higuera. 2010. ε-removal by loop reduction for finite-state automata. In *Language and Logos*, pages 297–312, Berlin. Akademie Verlag.

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. 2006. *Introduction to Automata Theory, Languages, and Computation*, 3 edition. Addison-Wesley Longman Publishing Co., Inc., USA.

Liang Huang. 2008. Advanced dynamic programming in semiring and hypergraph frameworks. In *Coling 2008: Advanced Dynamic Programming in Computational Linguistics: Theory, Algorithms and Applications - Tutorial notes*, pages 1–18, Manchester, UK. Coling 2008 Organizing Committee.

Roger Levy. 2008. A noisy-channel model of human sentence comprehension under uncertain input. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 234–243, Honolulu, Hawaii. Association for Computational Linguistics.

Roger Levy. 2011. Integrating surprisal and uncertain-input models in online sentence comprehension: formal techniques and empirical results. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1055–1065, Portland, Oregon, USA. Association for Computational Linguistics.

Andreas Maletti. 2010. Why synchronous tree substitution grammars? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 876–884, Los Angeles, California. Association for Computational Linguistics.

Andreas Maletti and Giorgio Satta. 2009. Parsing algorithms based on tree automata. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 1–12, Paris, France. Association for Computational Linguistics.

Mehryar Mohri. 2001. Generic ε-removal algorithm for weighted automata. In *Implementation and Application of Automata*, pages 230–242, Berlin, Heidelberg. Springer Berlin Heidelberg.

Mehryar Mohri. 2002. Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321–350.

Mark-Jan Nederhof and Giorgio Satta. 2003. Probabilistic parsing as intersection. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 137–148, Nancy, France.

Mark-Jan Nederhof and Giorgio Satta. 2011a. Computation of infix probabilities for probabilistic context-free grammars. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1213–1221, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Mark-Jan Nederhof and Giorgio Satta. 2011b. Prefix probabilities for linear context-free rewriting systems. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 151–162, Dublin, Ireland. Association for Computational Linguistics.

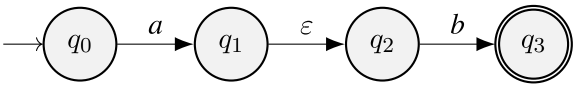
Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229.

Michael Sipser. 2006. *Introduction to the Theory of Computation*, 2 edition. Thomson Course Technology.

Hisao Tamaki and Taisuke Sato. 1984. Unfold/fold transformation of logic programs. In *Proceedings of the Second International Logic Programming Conference*, pages 127–138, Uppsala, Sweden. Uppsala University.

A Failure Cases of Original Construction

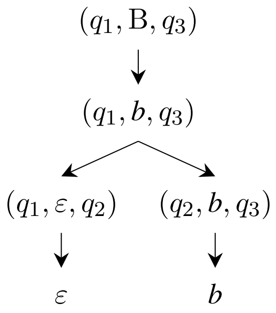
We distinguish two types of failure cases: (i) $\mathtt{supp}(L_{\mathcal{G}_\cap}) \neq \mathtt{supp}(L_{\mathcal{A}}) \cap \mathtt{supp}(L_{\mathcal{G}})$ and (ii) $L_{\mathcal{G}_\cap} \neq L_{\mathcal{A}} \cap L_{\mathcal{G}}$, both of which we will exemplify now. Notably, the case (ii) follows from (i), but—to be comprehensible—we will nonetheless give an example where (ii) fails without (i). For case (i), consider the following unweighted FSA:



and the following unweighted CFG:

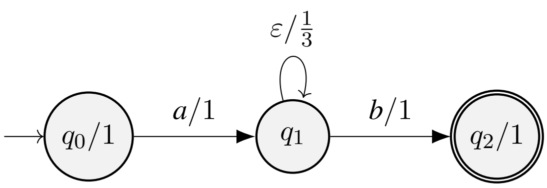
$$\begin{aligned} &\mathrm{S} \rightarrow \mathrm{A}\,\mathrm{B} \\ &\mathrm{A} \rightarrow a \\ &\mathrm{B} \rightarrow b \end{aligned}$$

It is easy to see that the intersection of the language accepted by the FSA and the language generated by the CFG is $\{ab\}$. Construction 1, however, outputs an empty grammar (after pruning useless rules as in footnote 9) and, hence, an empty language. To see this, consider Eq. (4d) and Eq. (4f). First, Eq. (4f) will create a rule $(q_1, \varepsilon, q_2) \rightarrow \varepsilon$, but (q_1, ε, q_2) will be useless because it cannot be reached from any of the rules produced by Eq. (4d). Second, Eq. (4d) will produce reachable nonterminals (q_0, A, q_i) and (q_i, B, q_3), with $i \in \{1, 2\}$. The case of $i = 1$ will reach a but not b, and $i = 2$ will reach b but not a. Let us now show how our generalized construction fixes this failure case. Eq. (5g) generates the rule $(q_1, b, q_3) \rightarrow (q_1, \varepsilon, q_2)(q_2, b, q_3)$ which then combines with rule $(q_1, \mathrm{B}, q_3) \rightarrow (q_1, b, q_3)$ to form a subderivation[12] that covers the substring εb, as shown in the picture below.



Note that rules generated by Eq. (5g) can only mention symbol ε in the left child, not in the right child, as discussed in footnote 11.

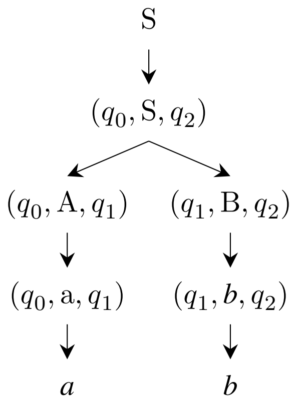
As stated above, to be comprehensive, we also show a case where only case (ii) fails, without (i). Take the following WFSA over the Inside semiring (Huang, 2008):



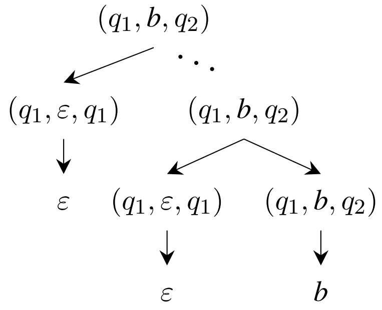
and the same grammar as above with weight 1 for all rules. It is easy to see that the language's weight for $\boldsymbol{s} = ab$ in the WFSA is a geometric series $L_{\mathcal{A}}(\boldsymbol{s}) = \sum_{i=0}^{\infty} \left(\frac{1}{3}\right)^i = \frac{3}{2}$, while in the WCFG, $L_{\mathcal{G}}(\boldsymbol{s}) = 1$. However, the output grammar $\mathcal{G}_\cap$ of Construction 1 will contain one single derivation $\boldsymbol{d}$:

---

[12] In App. B we give a formal definition of subderivation.



with $w(\boldsymbol{d}) = 1$, as all rules either stem from $\mathcal{G}$ or from the arcs $q_0 \xrightarrow{a/1} q_1$ and $q_1 \xrightarrow{b/1} q_2$.This will result in $L_{\mathcal{G}_\cap} = 1$, but $L_\mathcal{A} \cap L_\mathcal{G} = \frac{3}{2}$. This is because there are no derivations rooted at S in $\mathcal{G}_\cap$ that match with the ε-arcs in $\mathcal{A}$: Similarly to the example above, (q_1, ε, q_1) will not be reachable. We will now briefly show how our construction fixes this failure case as well. Note that there are infinitely many paths in the WFSA with yield $\boldsymbol{s} = ab$; but there is also only a single derivation in $\mathcal{D}_\mathcal{G}$ with this yield. Our construction thus ensures that there is exactly one derivation in $\mathcal{D}_{\mathcal{G}_\cap}$ for every *ab* path in $\mathcal{D}_\mathcal{A}$. As the ε-loop allows unboundedly long subpaths from q_1 to q_2 that are labeled with $\varepsilon^* b$, the rules generated by Eq. (5g) will build corresponding unboundedly deep subderivations of the following form:



Finally we observe that a similar argument holds for rules generated by Eq. (5b), and ε-arcs that occur immediately before a final state.

B Proofs

B.1 Proof of Theorem 1

Theorem 1 gives a result for derivations (which are always rooted at S) and paths (which always connect an initial state with a final state). However, in order to prove this theorem we must also consider subderivations and subpaths. We define subderivations as follows: a **subderivation** $\widetilde{\boldsymbol{d}}$ is a sequence $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_N$ with $N \geq 0$, where (i) in the case of $N > 0$, $\boldsymbol{\alpha}_0 = \mathrm{X}$, $\mathrm{X} \in \mathcal{N}$, and $\boldsymbol{\alpha}_N \in (\varepsilon \cup \Sigma^*)$, such that for all $0 < n \leq N$, we have $\boldsymbol{\alpha}_{n-1} \overset{p_n}{\Rightarrow}_L \boldsymbol{\alpha}_n$ for some $p_n \in \mathcal{P}$, and (ii) in the case of $N = 0$, $\boldsymbol{\alpha}_0 \in \Sigma \cup \{\varepsilon\}$. The weight and yield of subderivations are defined analogously to that of derivations. In the extended case of $N = 0$, the yield is equal to $\boldsymbol{\alpha}_0$ and the weight is set to $\mathbf{1}$. We will say that a subderivation is **rooted** at X if $\boldsymbol{\alpha}_0 = \mathrm{X}$. We denote the set of subderivations rooted at X with $\mathcal{D}_\mathcal{G}(\mathrm{X})$. Moreover, a subpath is defined as follows: A **subpath** $\widetilde{\boldsymbol{\pi}}$ (of length $N \geq 0$), is (i) in the case of $N > 0$, a sequence of arcs in δ^* where the states of adjacent arcs are matched, and (ii) in the case of $N = 0$ a single state $q \in Q$.[13] The subpath's weight, denoted $\widetilde{w}(\widetilde{\boldsymbol{\pi}})$, is the product $\widetilde{w}(\widetilde{\boldsymbol{\pi}}) = \bigotimes_{n=1}^{N} w_n$ of the weights of the arcs along the subpath. In the extended case $N = 0$ we set the weight to $\mathbf{1}$ and the yield to ε. Note that, in contrast to the

---

[13] We note the difference to paths defined in §2.3: a subpath does not need to start in an initial state and end in a final state.

weight of a path, the weight of a subpath does not account for initial and final weights. The yield is defined identically to that of paths. We denote the set of all paths starting at q_i and ending at q_j with $\mathcal{D}_{\mathcal{A}}(\{q_i, q_j\})$. Note that the definitions of subderivation and subpath encapsulate the definitions of derivation and path respectively. Furthermore, we will denote with $p\left(\boldsymbol{\pi}\right)$ and $n\left(\boldsymbol{\pi}\right)$, respectively, the first and the last state encountered along a path.

We will now prove two lemmas that will be necessary for the proof of Theorem 1.

Lemma 1. *For any triplet* $(q_0, \mathrm{X}, q_m) \in \mathcal{N}_\cap$, *with* $\mathrm{X} \neq \widetilde{\mathrm{S}}$ *and* $q_0, q_m \in Q$, *there is a bijection* $\psi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$ *from* $\mathcal{D}_{\mathcal{G}_\cap}((q_0, \mathrm{X}, q_m))$ *to the weighted join* $(\mathcal{D}_{\mathcal{G}}(\mathrm{X}) \bowtie \mathcal{D}_{\mathcal{A}}(\{q_0, q_m\}))$, *restricted to tuples in which the path does not have an* ε*-arc immediately before a final state. Moreover, it holds that:*

$$w\left(\widetilde{\boldsymbol{d}}_\cap\right) = w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) \tag{7}$$

$$\texttt{yield}\left(\widetilde{\boldsymbol{d}}_\cap\right) = \texttt{yield}\left(\widetilde{\boldsymbol{d}}\right) = \texttt{yield}\left(\widetilde{\boldsymbol{\pi}}\right) \tag{8}$$

Proof. We begin by showing that ψ is well defined, that it is injective and that it satisfies the properties in Eqs. (7) and (8). We prove this by induction on subderivations.

Lemma 1's Base Case. We begin by observing that the only terminal rules from $\mathcal{P}_\cap$ are defined by Eq. (5f) and Eq. (5e).

Lemma 1's Base Case, Part #1. $\widetilde{\boldsymbol{d}}_\cap$ is obtained by the application of a single production rule $(q_0, a, q_1) \xrightarrow{w} a$ from Eq. (5f). We define $\psi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$, where $\widetilde{\boldsymbol{\pi}} = q_0 \xrightarrow{a/w} q_1$ and $\widetilde{\boldsymbol{d}} = a$ is the subderivation that contains just the string a with weight $\mathbf{1}$. It is easy to see that the yield is preserved. Moreover:

$$\begin{aligned} w\left(\widetilde{\boldsymbol{d}}_\cap\right) &= w && \text{(by Eq. (5f))} \quad \tag{9a} \\ &= w \otimes \mathbf{1} \tag{9b} \\ &= \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) \otimes w\left(\widetilde{\boldsymbol{d}}\right) \tag{9c} \end{aligned}$$

Lemma 1's Base Case, Part #2. $\widetilde{\boldsymbol{d}}_\cap$ is obtained by the application of a single production rule $(q_0, \mathrm{X}, q_0) \xrightarrow{w} \varepsilon$ from Eq. (5e). We construct ψ as follows: $\psi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$, where $\widetilde{\boldsymbol{d}} = \mathrm{X} \overset{p}{\Rightarrow}_L \varepsilon$ with $p = \mathrm{X} \xrightarrow{w} \varepsilon$, and $\widetilde{\boldsymbol{\pi}}$ is the subpath q_0 with weight $\mathbf{1}$. Clearly the yield is preserved and:

$$\begin{aligned} w\left(\widetilde{\boldsymbol{d}}_\cap\right) &= w && \text{(by Eq. (5e))} \quad \tag{10a} \\ &= w \otimes \mathbf{1} \tag{10b} \\ &= w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) \tag{10c} \end{aligned}$$

Lemma 1's Induction Step. In the induction step, we show that the properties that we have shown for the base case propagate upwards along the derivation. In general, we will show that for any $\widetilde{\boldsymbol{d}}_\cap = (q_0, \mathrm{X}, q_M) \overset{p}{\Rightarrow}_L (q_0, \mathrm{X}_1, q_1), \ldots, (q_{M-1}, \mathrm{X}_\mathrm{M}, q_M) \Rightarrow_L \ldots$, we can construct $\psi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$ such that the mapping is injective and that the properties in Eqs. (7) and (8) hold. Additionally, as for the base case, we will show that $\widetilde{\boldsymbol{\pi}}$ connects q_0 with q_M and that $\widetilde{\boldsymbol{d}}$ is rooted at X. As our inductive hypothesis, we will assume that each of these hypotheses hold for the subderivations rooted at each of the child nonterminals $(q_0, \mathrm{X}_1, q_1), \ldots, (q_{M-1}, \mathrm{X}_\mathrm{M}, q_M)$. We note that the rules from $\mathcal{P}_\cap$ which apply to a nonterminal of form (q_0, X, q_M) with $\mathrm{X} \in \Sigma$ are discussed in base case #1, if instead $\mathrm{X} \in \mathcal{N}$, we either have base case #2 or one of the rules defined by Eq. (5d) and Eq. (5g); we discuss each now.

Lemma 1's Induction Step, Part #1. The topmost rule applied in $\widetilde{\boldsymbol{d}}_\cap$ is $p = (q_0, \mathrm{a}, q_2) \xrightarrow{\mathbf{1}} (q_0, \varepsilon, q_1)(q_1, \mathrm{a}, q_2)$ defined by Eq. (5g). We denote with $\widetilde{\boldsymbol{d}}_{\cap,1}$ the subderivation rooted at (q_0, ε, q_1), and we observe that the only possible form for this derivation is $(q_0, \varepsilon, q_1) \overset{p}{\Rightarrow}_L \varepsilon$ for some $p = (q_0, \varepsilon, q_1) \xrightarrow{w} \varepsilon$. We denote with $\widetilde{\boldsymbol{d}}_{\cap,2}$ the subderivation rooted at (q_1, a, q_2), then by inductive hypothesis, we know that there is a mapping $\psi(\widetilde{\boldsymbol{d}}_{\cap,2}) = \langle \widetilde{\boldsymbol{d}}_2, \widetilde{\boldsymbol{\pi}}_2 \rangle$ such that Eqs. (7) and (8) are satisfied.

Then we construct $\psi(\widetilde{\boldsymbol{d}}_{\cap}) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$, so that $\widetilde{\boldsymbol{d}} = \widetilde{\boldsymbol{d}}_2$ and $\widetilde{\boldsymbol{\pi}} = q_0 \xrightarrow{\varepsilon/w} q_1 \circ \widetilde{\boldsymbol{\pi}}_2$. As the yield of the subderivation rooted at (q_0, ε, q_1) is ε, the yield of $\widetilde{\boldsymbol{d}}_{\cap}$ is the same as that of $\widetilde{\boldsymbol{d}}_{\cap,2}$. Further, the yield of $\widetilde{\boldsymbol{\pi}}$ is the same as $\widetilde{\boldsymbol{\pi}}_2$. We thus have that:

$$\texttt{yield}\left(\widetilde{\boldsymbol{d}}_{\cap}\right) = \texttt{yield}\left(\widetilde{\boldsymbol{d}}_{\cap,2}\right), \qquad \texttt{yield}\left(\widetilde{\boldsymbol{d}}\right) = \texttt{yield}\left(\widetilde{\boldsymbol{d}}_2\right), \qquad \texttt{yield}\left(\widetilde{\boldsymbol{\pi}}\right) = \texttt{yield}\left(\widetilde{\boldsymbol{\pi}}_2\right) \quad (11)$$

By induction, we have that the yield is preserved. Similarly, we have that the weight is preserved:

$$\begin{aligned}
w\left(\widetilde{\boldsymbol{d}}_{\cap}\right) &= \mathbf{1} \otimes w\left(\widetilde{\boldsymbol{d}}_{\cap,1}\right) \otimes w\left(\widetilde{\boldsymbol{d}}_{\cap,2}\right) && && \text{(12a)}\\
&= \mathbf{1} \otimes w \otimes w\left(\widetilde{\boldsymbol{d}}_2\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}_2\right) && \text{(inductive hypothesis)} && \text{(12b)}\\
&= w\left(\widetilde{\boldsymbol{d}}_2\right) \otimes \left(w \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}_2\right)\right) && \text{(commutativity)} && \text{(12c)}\\
&= w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) && && \text{(12d)}
\end{aligned}$$

Finally, by induction we assume that $\widetilde{\boldsymbol{\pi}}_2$ connects state q_1 with state q_2, which implies that $\widetilde{\boldsymbol{\pi}}$ connects state q_0 with state q_2.

Lemma 1's Induction Step, Part # 2. The topmost rule applied in $\widetilde{\boldsymbol{d}}_{\cap}$ is $p = (q_0, \mathrm{X}, q_M) \xrightarrow{w} (q_0, \mathrm{X}_1, q_1), \ldots, (q_{M-1}, \mathrm{X}_\mathrm{M}, q_M)$ defined by Eq. (5d). By induction we assume that the subderivation $\widetilde{\boldsymbol{d}}_{\cap,m}$ rooted at $(q_{m-1}, \mathrm{X}_\mathrm{m}, q_m)$ is mapped by ψ into a subderivation $\widetilde{\boldsymbol{d}}_m$ rooted at X_m and a path $\widetilde{\boldsymbol{\pi}}_m$, so that $\texttt{yield}\left(\widetilde{\boldsymbol{d}}_{\cap,m}\right) = \texttt{yield}\left(\widetilde{\boldsymbol{d}}_m\right) = \texttt{yield}\left(\widetilde{\boldsymbol{\pi}}_m\right)$ and that $w\left(\widetilde{\boldsymbol{d}}_{\cap,m}\right) = w\left(\widetilde{\boldsymbol{d}}_m\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}_m\right)$. We then define $\psi(\widetilde{\boldsymbol{d}}_{\cap}) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$ where $\widetilde{\boldsymbol{d}} = \mathrm{X} \overset{p}{\Rightarrow}_L \mathrm{X}_1, \ldots, \mathrm{X}_\mathrm{M} \Rightarrow_L \ldots$ with $p = \mathrm{X} \xrightarrow{w} \mathrm{X}_1, \ldots \mathrm{X}_\mathrm{M}$ and $\widetilde{\boldsymbol{\pi}} = \widetilde{\boldsymbol{\pi}}_1 \circ \ldots \circ \widetilde{\boldsymbol{\pi}}_M$. As the states of neighboring triplets are matched, and by induction we assume that $\widetilde{\boldsymbol{\pi}}_m$ connects states q_{m-1} with state q_m, we have that $\widetilde{\boldsymbol{\pi}}$ is a path from q_0 to q_M. We note that the yield of $\widetilde{\boldsymbol{d}}$ is obtained by concatenation of $\texttt{yield}\left(\widetilde{\boldsymbol{d}}_m\right)$ from left to right, and that similarly the yield of $\widetilde{\boldsymbol{\pi}}$ is obtained by concatenation of $\texttt{yield}\left(\widetilde{\boldsymbol{\pi}}_m\right)$ from left to right. This, together with the inductive hypothesis proves Eq. (8) of the lemma—as the yield of $\widetilde{\boldsymbol{d}}_{\cap}$ will also be given by the concatenation of $\texttt{yield}\left(\widetilde{\boldsymbol{d}}_{\cap,m}\right)$ from left to right. We now show that Eq. (7) on weights holds:

$$\begin{aligned}
w\left(\widetilde{\boldsymbol{d}}_{\cap}\right) &= w \otimes \bigotimes_{m=1}^{M} w\left(\widetilde{\boldsymbol{d}}_{\cap,m}\right) && && \text{(13a)}\\
&= w \otimes \bigotimes_{m=1}^{M} w\left(\widetilde{\boldsymbol{d}}_m\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}_m\right) && \text{(inductive hypothesis)} && \text{(13b)}\\
&= \left(w \otimes \bigotimes_{m=1}^{M} w\left(\widetilde{\boldsymbol{d}}_m\right)\right) \otimes \bigotimes_{m=1}^{M} \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}_m\right) && \text{(commutativity)} && \text{(13c)}\\
&= w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) && && \text{(13d)}
\end{aligned}$$

We have defined ψ in a bottom-up fashion. At each step changing the topmost rule would result either in a different tree $\widetilde{\boldsymbol{d}}$ or in a different path $\widetilde{\boldsymbol{\pi}}$, which proves injectivity. The proof that ψ is surjective is very similar, and consists in showing by induction, that for any $\widetilde{\boldsymbol{d}} \in \mathcal{D}_{\mathcal{G}}(\mathrm{X})$, and for any path $\widetilde{\boldsymbol{\pi}}$ that does not have a sequence of ε-arc before a final state, it is always possible to build a derivation in $\mathcal{D}_{\mathcal{G}_{\cap}}((p(\boldsymbol{\pi}), \mathrm{X}, n(\boldsymbol{\pi})))$. We limit ourselves to noting that it is always possible to do so by using rules from Eqs. (5d) to (5f), as in the original Bar-Hillel construction, and by using rules defined by Eq. (5g) to cover ε-arcs in the WFSA. ■

Lemma 2. *For any triplet* $(q_I, \widetilde{\mathrm{S}}, q) \in \mathcal{N}_\cap$*, with* $q_I \in Q_I, q \in Q$*, there is a bijection* $\xi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$ *from* $\mathcal{D}_{\mathcal{G}_\cap}\big((q_I, \widetilde{\mathrm{S}}, q)\big)$ *to the join* $(\mathcal{D}_{\mathcal{G}}(\mathrm{S}) \bowtie \mathcal{D}_{\mathcal{A}}(\{q_I, q\}))$*, and we have that:*

$$w\left(\widetilde{\boldsymbol{d}}_\cap\right) = w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) \tag{14}$$

$$\mathtt{yield}\left(\widetilde{\boldsymbol{d}}_\cap\right) = \mathtt{yield}\left(\widetilde{\boldsymbol{d}}\right) = \mathtt{yield}\left(\widetilde{\boldsymbol{\pi}}\right) \tag{15}$$

Proof. We now present an inductive proof (similar to the above) for this lemma.

Lemma 2's Base Case. The topmost rule applied in $\widetilde{\boldsymbol{d}}_\cap$ is $(q_I, \widetilde{\mathrm{S}}, q) \xrightarrow{\mathbf{1}} (q_I, \mathrm{S}, q)$ from rules defined by Eq. (5c). We denote with $\widetilde{\boldsymbol{d}}_{\cap,1}$ the subderivation rooted at (q_I, S, q). Then by Lemma 1, we know that there is a mapping $\psi(\widetilde{\boldsymbol{d}}_{\cap,1}) = \langle \widetilde{\boldsymbol{d}}_1, \widetilde{\boldsymbol{\pi}}_1 \rangle$ such that Eqs. (14) and (15) are satisfied. We then define $\xi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}_1, \widetilde{\boldsymbol{\pi}}_1 \rangle$, and one can easily see that the properties in Eqs. (14) and (15) are satisfied.

Lemma 2's Induction Step. The topmost rule applied in $\widetilde{\boldsymbol{d}}_\cap$ is $(q_I, \widetilde{\mathrm{S}}, q_1) \xrightarrow{\mathbf{1}} (q_I, \widetilde{\mathrm{S}}, q_0)(q_0, \varepsilon, q_1)$ from rules defined by Eq. (5b). We denote with $\widetilde{\boldsymbol{d}}_{\cap,1}$ the subderivation rooted at $(q_I, \widetilde{\mathrm{S}}, q_0)$, and we assume by induction that $\xi(\widetilde{\boldsymbol{d}}_{\cap,1}) = \langle \widetilde{\boldsymbol{d}}_1, \widetilde{\boldsymbol{\pi}}_1 \rangle$ and that properties in Eqs. (14) and (15) hold. We denote with $\widetilde{\boldsymbol{d}}_{\cap,2}$ the subderivation rooted at (q_0, ε, q_1), and we observe that the only possible form for this derivation is $(q_0, \varepsilon, q_1) \overset{p}{\Rightarrow}_L \varepsilon$ for some $p = (q_0, \varepsilon, q_1) \xrightarrow{w} \varepsilon$. Then we can construct $\xi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$, where $\widetilde{\boldsymbol{d}} = \widetilde{\boldsymbol{d}}_1$ and $\widetilde{\boldsymbol{\pi}} = \widetilde{\boldsymbol{\pi}}_1 \circ q_0 \xrightarrow{\varepsilon/w} q_1$. The property in Eq. (15) is clearly satisfied, for property Eq. (14), we have:

$$\begin{aligned}
w\left(\widetilde{\boldsymbol{d}}_\cap\right) &= \mathbf{1} \otimes w\left(\widetilde{\boldsymbol{d}}_{\cap,1}\right) \otimes w\left(\widetilde{\boldsymbol{d}}_{\cap,2}\right) && \text{(16a)}\\
&= w\left(\widetilde{\boldsymbol{d}}_{\cap,1}\right) \otimes w && \text{(weight of } \widetilde{\boldsymbol{d}}_{\cap,2}\text{)} \quad \text{(16b)}\\
&= w\left(\widetilde{\boldsymbol{d}}_1\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}_1\right) \otimes w && \text{(inductive hypothesis)} \quad \text{(16c)}\\
&= w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\left(\widetilde{\boldsymbol{\pi}}\right) && \text{(weight of } \widetilde{\boldsymbol{\pi}}\text{)} \quad \text{(16d)}
\end{aligned}$$

As for Lemma 1 we note that modifying the topmost rule in $\widetilde{\boldsymbol{d}}_\cap$, would always result either in a different derivation $\widetilde{\boldsymbol{d}}$ or in a different path $\widetilde{\boldsymbol{\pi}}$, which proves injectivity. Surjectivity can be shown by induction, similarly to how we did for injectivity. We will simply note that given any derivation $\widetilde{\boldsymbol{d}}$ rooted at S, and given any path $\widetilde{\boldsymbol{\pi}}$ starting from an initial state, it is always possible to build a matching derivation $\widetilde{\boldsymbol{d}}_\cap$in $\mathcal{D}_{\mathcal{G}_\cap}((p\left(\boldsymbol{\pi}\right), \widetilde{\mathrm{S}}, n\left(\boldsymbol{\pi}\right)))$, by using the result from Lemma 1, and applying rules defined by Eqs. (5b) and (5c). ■

We can finally prove Theorem 1, which we restate here for convenience.

Theorem 1. *Let* $\mathcal{G}$ *be a WCFG and* $\mathcal{A}$ *a WFSA over the same alphabet* Σ *and commutative semiring* $\mathcal{W}$*. Let* $\mathcal{G}_\cap$ *be the grammar obtained with our generalized construction. Then we have strong equivalence between* $\mathcal{G}_\cap$ *and* $\langle \mathcal{G}, \mathcal{A} \rangle$*; meaning that there is a weight-preserving, yield-preserving bijection between* $\mathcal{D}_{\mathcal{G}_\cap}$ *and* $(\mathcal{D}_{\mathcal{G}} \bowtie \mathcal{D}_{\mathcal{A}})$*.*

Proof. Any derivation $\boldsymbol{d}_\cap$ in $\mathcal{D}_{\mathcal{G}_\cap}(\mathrm{S})$ takes the form $\mathrm{S} \overset{p}{\Rightarrow}_L (q_I, \widetilde{\mathrm{S}}, q_F) \Rightarrow_L \ldots$ with $p = \mathrm{S} \xrightarrow{\lambda(q_I) \otimes \rho(q_F)} (q_I, \widetilde{\mathrm{S}}, q_F)$, for $q_I \in Q_I$ and $q_F \in Q_F$. We denote with $\widetilde{\boldsymbol{d}}_\cap$ the subderivation rooted at $(q_I, \widetilde{\mathrm{S}}, q_F)$. We can thus define $\phi(\boldsymbol{d}_\cap) = \langle \boldsymbol{d}, \boldsymbol{\pi} \rangle = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$, where $\xi(\widetilde{\boldsymbol{d}}_\cap) = \langle \widetilde{\boldsymbol{d}}, \widetilde{\boldsymbol{\pi}} \rangle$, and ξ is the bijection defined in Lemma 2. By Lemma 2 we have that $\widetilde{\boldsymbol{d}} = \boldsymbol{d}$ is rooted at S, and that $\widetilde{\boldsymbol{\pi}} = \boldsymbol{\pi}$ has initial and final states: $p\left(\widetilde{\boldsymbol{\pi}}\right) = q_I$ and $n\left(\widetilde{\boldsymbol{\pi}}\right) = q_F$. Clearly, $\mathtt{yield}\left(\boldsymbol{d}_\cap\right) = \mathtt{yield}\left(\widetilde{\boldsymbol{d}}_\cap\right)$ and, by Lemma 2, $\mathtt{yield}\left(\widetilde{\boldsymbol{d}}_\cap\right) = \mathtt{yield}\left(\widetilde{\boldsymbol{d}}\right) = \mathtt{yield}\left(\widetilde{\boldsymbol{\pi}}\right)$. Further, by definition $\mathtt{yield}\left(\boldsymbol{d}\right) = \mathtt{yield}\left(\widetilde{\boldsymbol{d}}\right)$

and $\texttt{yield}\,(\widetilde{\boldsymbol{\pi}}) = \texttt{yield}\,(\boldsymbol{\pi})$. Moreover, we have that:

$$
\begin{aligned}
w\,(\boldsymbol{d}_\cap) &= w\,(p) \otimes w\left(\widetilde{\boldsymbol{d}}_\cap\right) && \text{(weight of a derivation)} \quad \text{(17a)}\\
&= w\,(p) \otimes w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\,(\widetilde{\boldsymbol{\pi}}) && \text{(Lemma 2)} \quad \text{(17b)}\\
&= \lambda(q_I) \otimes \rho(q_F) \otimes w\left(\widetilde{\boldsymbol{d}}\right) \otimes \widetilde{w}\,(\widetilde{\boldsymbol{\pi}}) && \text{(weight of } p\text{)} \quad \text{(17c)}\\
&= w\left(\widetilde{\boldsymbol{d}}\right) \otimes \lambda(q_I) \otimes \widetilde{w}\,(\widetilde{\boldsymbol{\pi}}) \otimes \rho(q_F) && \text{(commutativity)} \quad \text{(17d)}\\
&= w\,(\boldsymbol{d}) \otimes w\,(\boldsymbol{\pi}) && \text{(definition of weight of a path)} \quad \text{(17e)}
\end{aligned}
$$

which proves that ϕ is weight and yield preserving. By Lemma 2 we know that ξ is a bijection, which implies that modifying the topmost rule p would result in a different tuple $\langle \boldsymbol{d}, \boldsymbol{\pi} \rangle$. This proves the injectivity of ϕ. Conversely, consider any path $\boldsymbol{\pi}$ connecting an initial state with a final one and any derivation $\boldsymbol{d}$ rooted at S, such that $\texttt{yield}\,(\boldsymbol{d}) = \texttt{yield}\,(\boldsymbol{\pi})$. By Lemma 2 we know that it is always possible to construct a subderivation $\widetilde{\boldsymbol{d}}_\cap$, rooted at $(q_I, \widetilde{\mathrm{S}}, q_F)$, that satisfies Eqs. (14) and (15). Thus we can construct $\boldsymbol{d}_\cap = \mathrm{S} \overset{p}{\Rightarrow}_L (q_I, \widetilde{\mathrm{S}}, q_F) \Rightarrow_L \cdots$ with $p = \mathrm{S} \xrightarrow{\lambda(q_I)\otimes\rho(q_F)} (q_I, \widetilde{\mathrm{S}}, q_F)$ a rule from Eq. (5a). This shows the surjectivity of ϕ.

■

B.2 Proof of Corollary 1

Corollary 1. *$\mathcal{G}_\cap$ and $\langle \mathcal{G}, \mathcal{A} \rangle$ are weakly equivalent, meaning that $L_{\mathcal{G}_\cap}(\boldsymbol{s}) = L_{\mathcal{G}}(\boldsymbol{s}) \otimes L_{\mathcal{A}}(\boldsymbol{s})$ whenever the values on the right-hand side are defined.*

Proof. §2.1 defined both $L_{\mathcal{A}}(\boldsymbol{s})$ and $L_{\mathcal{G}}(\boldsymbol{s})$ as sums over derivations that yield $\boldsymbol{s}$. If there are only finitely many such derivations, then the sum is well-defined by applying the associative–commutative operator $\oplus$ finitely many times. However, footnote 5 noted that countably infinite sums can arise. We treat this issue by augmenting the semiring with an operator $\bigoplus$ that is applied to a countable (possibly infinite) multiset of weights and returns a value that is interpreted as the sum of those weights, or else returns a special "undefined" value $\bot \notin A$ to indicate that the sum diverges.

We require $\bigoplus$ to satisfy the following axioms for any two countable multisets $I, J \subseteq A$ such that

$$\bigoplus I = W \in A \qquad \bigoplus J = V \in A \tag{18}$$

- *Infinite distributivity*: Let $I \bigotimes J$ denote the multiset $\{\!\!\{ i \otimes j : i \in I, j \in J \}\!\!\}$. Then $\bigoplus (I \bigotimes J) = W \otimes V \in A$.
- *Infinite associativity*: for any partition[14] $I = \bigcup_{k \in K} I_k$, we have $\bigoplus I_k \in A$ for each $k \in K$ and furthermore $\bigoplus_{k \in K} (\bigoplus I_k) = W$.
- *Base cases*: For any $w, w' \in A$, $\bigoplus \{\!\!\{ w, w' \}\!\!\} = w \oplus w'$, $\bigoplus \{\!\!\{ w \}\!\!\} = w$, and $\bigoplus \{\!\!\{\,\}\!\!\} = \mathbf{0}$. Together with the previous property, this ensures that $\bigoplus$ agrees with the $\oplus$-based definition on finite multisets.

The first two axioms are adapted from part of Mohri (2002)'s definition of closed semirings. The proof of Corollary 1 uses only the first axiom, as follows. Given a string $\boldsymbol{s}$ such that $L_{\mathcal{A}}(\boldsymbol{s}), L_{\mathcal{G}}(\boldsymbol{s}) \in A$. By definition (§§2.3–2.4), $L_{\mathcal{A}}(\boldsymbol{s}) = \bigoplus I$ and $L_{\mathcal{G}}(\boldsymbol{s}) = \bigoplus J$ if we define $I = \{\!\!\{ w\,(\boldsymbol{\pi}) : \boldsymbol{\pi} \in \mathcal{D}_{\mathcal{A}}(\boldsymbol{s}) \}\!\!\}$ and $J = \{\!\!\{ w\,(\boldsymbol{d}) : \boldsymbol{d} \in \mathcal{D}_{\mathcal{G}}(\boldsymbol{s}) \}\!\!\}$. Then also $L_{\mathcal{G}_\cap}(\boldsymbol{s}) = \bigoplus (I \bigotimes J)$ since $I \bigotimes J = \{\!\!\{ w\,(\boldsymbol{d}) : \boldsymbol{d} \in \mathcal{D}_{\mathcal{G}_\cap}(\boldsymbol{s}) \}\!\!\}$ according to Theorem 1. By infinite distributivity, then, $L_{\mathcal{G}_\cap}(\boldsymbol{s}) = (\bigoplus I) \otimes (\bigoplus J) = L_{\mathcal{A}}(\boldsymbol{s}) \otimes L_{\mathcal{G}}(\boldsymbol{s}) \in A$ as claimed. ■

---

[14] Recall that partitions are definitionally disjoint.

Social Influence Dialogue Systems: A Survey of Datasets and Models For Social Influence Tasks

Kushal Chawla[1*] **Weiyan Shi**[2*] **Jingwen Zhang**[3**]
Gale Lucas[1**] **Zhou Yu**[2**] **Jonathan Gratch**[1**]
[1]University of Southern California [2]Columbia University
[3]University of California Davis
[1]{chawla, lucas, gratch}@ict.usc.edu [2]{ws2634, zy2461}@columbia.edu
[3]jwzzhang@ucdavis.edu

Abstract

Dialogue systems capable of social influence such as persuasion, negotiation, and therapy, are essential for extending the use of technology to numerous realistic scenarios. However, existing research primarily focuses on either task-oriented or open-domain scenarios, a categorization that has been inadequate for capturing influence skills systematically. There exists no formal definition or category for dialogue systems with these skills and data-driven efforts in this direction are highly limited. In this work, we formally define and introduce the category of *social influence dialogue systems* that influence users' cognitive and emotional responses, leading to changes in thoughts, opinions, and behaviors through natural conversations. We present a survey of various tasks, datasets, and methods, compiling the progress across seven diverse domains. We discuss the commonalities and differences between the examined systems, identify limitations, and recommend future directions. This study serves as a comprehensive reference for social influence dialogue systems to inspire more dedicated research and discussion in this emerging area.

1 Introduction

Consider a human user who signs up to interact with a persuasive dialogue system that motivates for engaging in physical exercise. The system: 1) uses *social cues like small talk and empathy*, useful for providing continued support, and 2) employs *persuasive strategies to convince* the user who, at least in the short-term, is reluctant to indulge in exercise. Does such a system fit the definition of a *task-oriented dialogue system* that are traditionally designed to assist users in completing their tasks such as restaurant or flight booking (Zhang et al., 2020c)? Although the system is task-oriented or goal-oriented per se, the task here goes beyond the traditional definition of assisting users, given the possible misalignment between the goals of the system and the user. Clearly, this system is also not open-domain (Huang et al., 2020). Although conversations involve social open-ended interactions, there is still a concrete goal of persuading the user towards a healthier habit.

Scenarios similar to above are ubiquitous in everyday life, including games (Peskov et al., 2020), social platforms (Tan et al., 2016), and therapeutic interactions (Tanana et al., 2016). Dialogue systems for these applications require a core function in human communication, that is, *social influence* (Cialdini and Goldstein, 2004; Cialdini, 2009), which involves influencing users' cognitive and emotional responses, leading to changes in thoughts, opinions, and behaviors through natural conversations. This goes beyond what is captured by traditional task definitions in the dialogue community which primarily focus on task completion and social companionship.

Despite numerous independent efforts in identifying and analyzing various social influence scenarios, there is a lack of common understanding around social influence in AI research which inhibits a systematic study in this space. Further, data-driven efforts for dialogue systems in this space are highly limited. To this end, we introduce the category of *social influence dialogue systems* (Section 2), providing a comprehensive literature review and discussing future directions.

Developing these systems holds importance in AI research for multiple reasons. Tackling these tasks not only involves AI but also aspects of game theory, communication, linguistics, and social psychology, making them an ideal testbed for interdisciplinary AI research. Most importantly, they reflect AI's general ability to consider their partners' inputs, tailor the communication strategies, personalize the responses, and lead the conversation actively.

We design a taxonomy for existing social influ-

---

[*] Equal contribution, [**] Co-supervised project

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 750–766
May 2-6, 2023

ence dialogue datasets, studying their task structure (symmetric vs asymmetric) and context (local vs global). We also organize them by their domains: games, multi-issue bargaining, social good, e-commerce, therapy and support, argumentation, conversational recommendations, and miscellaneous tasks (Section 3). We further design a taxonomy of existing methods, assisting readers to comprehend the progress and reflect on future directions. We organize them based on the system strategy, language generation, partner model, architecture, learning process, and the use of pretrained language models (Section 4). Finally, we identify key challenges and provide recommendations for future work (Section 5).

Over the years, research in task-oriented and open-domain dialogues has benefited from a myriad of survey efforts (Huang et al., 2020; Zhang et al., 2020c; Ni et al., 2021). We instead focus on dialogue systems with social influence capabilities and present a thorough review across various domains. We hope that our work serves as a timely entry point for interested researchers to take this area further, inspiring dedicated effort and discussion on social influence in the dialogue community.

2 Social Influence Dialogue Systems

"Social influence is a fact of everyday life" (Gass, 2015). It is the *change in thoughts, feelings, attitudes, or behaviors resulting from interaction with an individual or a group* (Rashotte, 2007). Influence is measured by quantifiable proxies of the observed change, like the interest to indulge in physical exercise before or after the interaction with a system, or the final deal in a negotiation as opposed to one person taking it all. Social influence dialogue systems act interactively and influence their partners in decision-making and behavioral contexts (Zhang et al., 2020a; Lee et al., 2020). This calls for an active role by the system, distinguishing them from other well-studied scenarios, such as purely task-oriented, where systems passively assist their partners to complete tasks, and open-domain, that target social companionship. Key social influence tasks include persuasion (Wang et al., 2019), aiming to change users' attitudes or behaviors, and negotiation, aiming to change the users' perspective to achieve a common ground (Lewis et al., 2017).

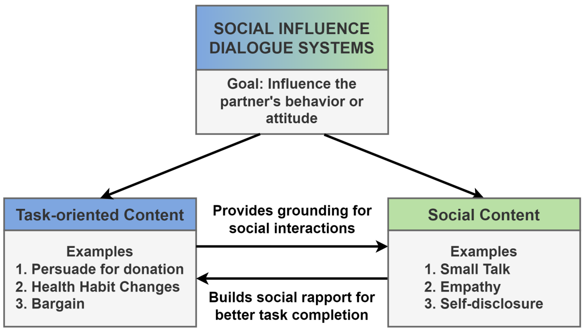


Figure 1: A conceptual overview.

Conceptual overview: Figure 1 distinguishes between the kinds of conversational content in social influence interactions. The *task-oriented content* focuses on influencing for a domain-specific goal, like persuading for donation, bargaining with trade-offs, or encouraging healthier habits. These interactions may also contain *social content*, such as small talk, empathy, or self-disclosure. The task-oriented content provides a context for social interactions. Depending on the task, social content is optional, but if present, can in turn build rapport and enhance user-system relationship for improved task outcomes (Liao et al., 2021).

Connections with task-oriented and open-domain systems: Similar to a task-oriented or an open-domain scenario, social influence dialogue can also be seen as a sequential decision making process with the goal of maximizing the expected reward (Huang et al., 2020; Gao et al., 2018). Our proposed category is not meant to be disjoint from these traditional categories. However, it still uniquely brings together the tasks that capture social influence, *which is fundamentally absent from how we primarily define dialogue tasks in the community*. Defining a new category that captures social influence dialogue would foster a dedicated effort towards this important aspect of real-world conversations.

Task-oriented scenarios focus on collaborative information exchange for a common goal of task completion. In social influence tasks, the goals of the system and the user can be different and even conflicting, leading to collaborative or non-collaborative interactions. Further, the goals can go beyond the current task (e.g. multiple therapy interactions, repeated negotiations), leading to social interactions for long-term relationships. If a scenario involves the system's goal to influence its partner, we consider it under social influence in this paper. For instance, He et al. (2018) studied buyer-seller price negotiations. The task of the buyer is to negotiate for a reasonable price (arguably making it task-oriented), but achieving it requires social influence skills of engaging in trade-offs and building a rapport with the seller so as to reach an agreement.

Measures of Success: The above discussion indicates that a comprehensive evaluation of social influence systems must draw from both task-oriented and open-domain dialogue research. Since there exist surveys that discuss the evaluation in these settings (Deriu et al., 2021; Li et al., 2021), we don't cover them here in detail. However, we define three essential axes for evaluation: 1) *Linguistic Performance*, or the system's linguistic sophistication based on automatic (e.g. perplexity, BLEU) and human (e.g. fluency, consistency, coherency) evaluation. 2) *Influence Outcome*, or the ability to influence defined by objective goals like the negotiated price or weight loss after therapy. 3) *Partner Perception*, or the subjective evaluation of the user, for instance, the user's satisfaction, likeness towards the system, and interest in interacting again. In a buyer-seller negotiation, if the seller hates the buyer in the end, no matter how favorable the deal is for the buyer, one might argue that this is still a failed negotiation for the buyer. Hence, we encourage future work to take all three dimensions into account collectively.

3 Social Influence Across Diverse Application Areas

We now illustrate social influence across numerous domains and application areas. In total, we curated 22 datasets from prior work that capture social influence in various forms, spanning 12 publication venues, 4 languages, and 7 application domains (see Appendix A for details on the compilation process). In general, the datasets capture the following information about an interaction: the *non-conversational context* for the participants (e.g. negotiation preferences or other role-specific information), the *conversation* between them, and *outcome assessment*. Optionally, some datasets also gather participant demographics and personality traits, utterance-level annotations, and subjective evaluations via post-surveys.

To understand the structural similarities and differences between these datasets, we design a taxonomy with two primary dimensions: **Task Structure** (*Symmetric* vs *Asymmetric*), and **Context Definition** (*Global* vs *Local*). **Task Structure** captures whether the participant roles are defined in a symmetric or an asymmetric manner. For instance, a typical multi-issue negotiation is symmetric, in the sense that both parties have their own preferences and goals based on which they actively try to reach a favorable agreement (Lewis et al., 2017). On the other hand, a counseling session between a therapist and a patient is asymmetric, where the therapist attempts to emotionally support the patient by employing social influence skills (Althoff et al., 2016). **Context Definition** relates to whether the input context before each interaction is defined globally or locally. For instance, the PersuasionForGood dataset globally defines the context of persuasion for charity donation, which is kept the same throughout (Wang et al., 2019). On the contrary, in a typical debate, although the rules are defined globally, the conversation topic and arguments are local and can vary for each conversation (Durmus and Cardie, 2019). We present this categorization in Table 1. We further categorize the datasets according to their **Domain**, **Source**, and the **# of parties**. We provide key statistics and the available metadata in Appendix B. We now briefly discuss the datasets in each domain.

Games: Strategy games involve social influence dynamics of trust and deception. Diplomacy captures deception in long-lasting relationships, where players forge and break alliances to dominate Europe (Peskov et al., 2020). Catan revolves around the trade of resources for acquiring roads, settlements, and cities (Asher et al., 2016; Boritchev and Amblard, 2021). The players have access to only a subset of resources that they would need, which encourages strategic influence and trade.

Multi-Issue Bargaining Tasks (MIBT): MIBT is a tractable closed-domain abstraction of a typical negotiation (Fershtman, 1990). It is based on a fixed set of issues each with a predefined priority for each player, which essentially governs the goals of the players. If the priorities of the players align, this leads to competitive negotiations, where each party attempts to convince their partner with trade-offs and persuasive arguments. If they don't, this allows cooperative interactions where the negotiators try to find optimal divisions that benefit everyone. DealOrNoDeal (Lewis et al., 2017) involves negotiations over three issues: *books*, *balls*, and *hats*. Other datasets define a more grounded scenario, such as symmetric CaSiNo (Chawla et al., 2021b) negotiations between two campsite neighbors and asymmetric JobInterview (Yamaguchi et al., 2021)

Name (Citation)	Domain	Source	Structure	Context	# of Parties
STAC (As16)	Games	Crowdsource	Symmetric	Global	Multiparty
Diplomacy (Pe20)	Games	Crowdsource	Asymmetric	Global	Multiparty
DinG (Bo21)	Games	University game night logs	Symmetric	Global	Multiparty
Tabletop (De15)	MIBT	Face-to-face, Wizard-of-Oz	Symmetric	Local	Bilateral
DealOrNoDeal (Le17)	MIBT	Crowdsource	Symmetric	Local	Bilateral
CaSiNo (Ch21)	MIBT	Crowdsource	Symmetric	Local	Bilateral
JobInterview (YaD21)	MIBT	Crowdsource	Asymmetric	Local	Bilateral
PersuasionforGood (Wa19)	Social Good	Crowdsource	Asymmetric	Global	Bilateral
CraigslistBargain (He18)	E-commerce	Crowdsource	Asymmetric	Local	Bilateral
AntiScam (Li20)	E-commerce	Crowdsource	Asymmetric	Global	Bilateral
MI (TaC16)	Therapy & Support	Psychotherapy session logs	Asymmetric	Global	Bilateral
SMS Counseling (Al16)	Therapy & Support	SMS chat logs	Asymmetric	Global	Bilateral
EmpatheticDialogues (Ra19)	Therapy & Support	Crowdsource	Asymmetric	Global	Bilateral
Hotline Counseling (De19)	Therapy & Support	Synthetic Transcripts	Asymmetric	Global	Bilateral
mPED (LiE21)	Therapy & Support	Physical activity clinical trials	Asymmetric	Global	Bilateral
Congressional Debates (Th06)	Argumentation	U.S. Congressional transcripts	Asymmetric	Local	Multiparty
Supreme Court (Da12)	Argumentation	Oyez.org transcripts	Asymmetric	Local	Multiparty
DeCour (Fo12)	Argumentation	Italian court hearings	Asymmetric	Local	Multiparty
ChangeMyView (Ta16)	Argumentation	Reddit	Asymmetric	Local	Multiparty
DDO Debates (Du19)	Argumentation	debate.org logs	Symmetric	Local	Bilateral
Court Debates (Ji20)	Argumentation	China Court transcripts	Asymmetric	Local	Multiparty
Target-Guided (Ta19)	Miscellaneous	Crowdsource	Symmetric	Local	Bilateral

Table 1: Categorization of social influence dialogue corpora. This list is non-exhaustive, and also covers the datasets that have enabled research into various sub-tasks and analyses that can eventually be useful for dialogue systems in respective domains. MIBT: Multi-Issue Bargaining Task. Key statistics and associated metadata are in Appendix 3.

negotiations between recruiters and applicants.

Social Good: Social influence is critical for social good applications. The tactics must be personalized using knowledge that is both relevant and appealing. PersuasionForGood (Wang et al., 2019) involves asymmetric interactions led by a persuader who attempts to convince the other participant for charity donation by employing a variety of tactics. For instance, *Logical Appeal* uses reason and evidence to support the argument, while *Emotional Appeal* elicits specific emotions.

E-commerce: These tasks are typically asymmetric. A *buyer* influences the seller towards a reasonable price, while the *seller* tries to maximize their own profit. An effective system must combine price-related reasoning with language realization. CraigslistBargain (He et al., 2018) involves open-ended price negotiations with rich influence strategies like embellishments, side offers, emotional appeals, and using world knowledge. Another example is customer support interactions in AntiScam dataset (Li et al., 2020), where users defend themselves against attackers who try to steal sensitive personal information with convincing arguments.

Therapy & Support: Effective therapy using social influence aids in the treatment of mental disorders, and substance use disorders, along with changing undesirable behaviors like unhealthy diets. A counselor needs to be adaptive, personalized, should understand the core issues, and should facilitate a change in patient's perspective (Althoff et al., 2016). In SMS counseling, Althoff et al. (2016) found that linguistic influence like pushing the conversation in the desired direction is associated with perspective change. Similar scenarios were captured in other datasets as well (Demasi et al., 2019; Liang et al., 2021). Tanana et al. (2016) collected the Motivational Interviewing dataset where the goal is to elicit and explore the patient's own motivations for behavior change. EmpatheticDialogues (Rashkin et al., 2019) captured empathetic support interactions, which has been associated with rapport and better task outcomes (Kim et al., 2004; Norfolk et al., 2007; Fraser et al., 2018).

Argumentation: In addition to factuality and social proof, a convincing argument must also consider the intensity, valence, authoritativeness, and framing (Chaiken, 1987; Althoff et al., 2014). Tan et al. (2016) released the ChangeMyView logs from Reddit, involving discussions on numerous controversial topics. Other datasets include Debate Dot Org (DDO) debates on diverse topics (Durmus and Cardie, 2019), congressional proceedings (Thomas et al., 2006), and court hearings (Fornaciari and Poesio, 2012; D.-N.-M. et al., 2012; Ji et al., 2020).

Conversational Recommendation: Everyday scenarios naturally hold potential for influence via recommendations, for instance, a movie fan persuading their friends to watch a movie that they adore. Li et al. (2018) and Dodge et al. (2016) collected movie recommendation datasets. Instead of guiding the conversation towards a specific movie, the goal is simply to provide recommendations based on facts and personal experiences. Nevertheless, they still provide interesting examples of scenarios that can involve social influence.

Miscellaneous: The Target-Guided dataset (Tang et al., 2019) was constructed from the PersonaChat corpus (Zhang et al., 2018). Instead of being open-ended, the Target-Guided scenario defines a concrete goal of naturally guiding the conversation to a designated target subject, thereby, making it a social influence setting.

4 Methodological Progress

Having summarized the datasets that capture social influence, we now discuss the modeling approaches developed for social influence dialogue systems. Most domains have seen efforts in analyzing human dialogue behaviors and their impact on task outcomes. Examples include analyzing deception in games (Peskov et al., 2020), the impact of persuasive strategies and dialogue acts on charity donations (Wang et al., 2019), cooperative and non-cooperative strategies in MIBT (Chawla et al., 2021b), the use of emotion expression for predicting partner perceptions (Chawla et al., 2021a), and studying semantic categories of persuasive arguments on web forums (Egawa et al., 2019).

In addition, researchers have targeted various domain-specific subtasks that can be crucial for the eventual development of dialogue systems in this space. This involves research in lie detection methods (Yeh and Ku, 2021; Yu et al., 2015), discourse parsing (Shi and Huang, 2019; Ouyang et al., 2021), strategy prediction (Chawla et al., 2021b; Wang et al., 2019), breakdown detection (Yamaguchi et al., 2021), outcome prediction (Sinha and Dasgupta, 2021; Chawla et al., 2020; Dutt et al., 2020), and argument mining (Dutta et al., 2022).

Research that directly targets the development of dialogue systems in this space is still nascent. Among other challenges like limited cross-cultural diversity and relatively smaller dataset size, social influence dialogue settings pose a unique challenge: an average human often exhibits sub-optimal strategic behaviors in social influence tasks (Wunderle, 2007; Babcock and Laschever, 2009). This means that standard seq2seq approaches trained on these collected datasets using supervised learning are fundamentally insufficient for developing effective dialogue systems with influence capabilities. Hence, prior work has put a special attention to the system strategy, employing different ways to model the strategy and language together.

We design a taxonomy of methods developed for social influence tasks, assisting readers to comprehend the progress and reflect on future directions. We organize them based on the system strategy, language generation, partner model, architecture, learning process, and the use of pretrained language models. We present annotations for all the surveyed methods in Table 2 and discuss the common categories in brief below.

4.1 Strategy Representation

Implicit: The most obvious way to represent the system strategy is *implicitly*, without any intended decoupling between system strategy and response realization. This corresponds to the usual sequence-to-sequence framework that has been a standard baseline for the methods developed in this space. An important example is the work by Lewis et al. (2017), who were one of the first works to train end-to-end dialogue models that exhibit social influence. The authors employed a neural network based on GRUs, one for encoding the negotiation context, one to encode the dialogue utterances, and two recurrent units to generate the output agreement in a bidirectional manner.

Latent vectors: Yarats and Lewis (2018) explored *latent vectors* to decouple utterance semantics from its linguistic aspects. Their hierarchical approach first constructs a latent vector from the input message, which is then used for response generation and planning. These latent vectors are trained to maximize the likelihood of future dialogue messages and actions, which enables the decoupling between semantics and realization.

Dialogue Acts (DAs): Dialogue Acts, such as greeting, offer propose, agreement, or disagreement, are effective at capturing a high-level structure of the dialogue flow in social influence settings, reducing the model strategy to first predicting the dialogue act for the next response. The use of DAs makes it convenient to apply reinforcement learning approaches (Zhang et al., 2020b; Yang et al., 2021), while also aiding in developing a modular dialogue system design (He et al., 2018).

Semantic Strategies: The structural properties ex-

Method	Domain	Strategy	NLG	Partner Model	Architecture	Learning	PLM
Le17	MIBT	Implicit	Generation	Simulated User	Enc-Dec	RL	
Ya18	MIBT	Latent Vectors	Generation	Simulated User	Hierarchical	RL	
Zh20	MIBT	DAs	Generation	DA Look-Ahead	Modular	RL	
He18	E-Com, MIBT	DA	Templates + Retrieval	Implicit	Modular	RL	
Ya21	E-Com	DAs	Templates + Retrieval	DA Look-Ahead	Modular	RL	
Zh19	E-Com, Social Good	DAs + Semantic	Generation	Implcit	Hierarchical	SL	
Jo20	E-Com	DAs + Semantic	Generation	Implcit	Hierarchical	SL	
Li20	E-Com, Social Good	DAs	Generation	Implicit	Dec-only	SL	GPT
Li21	Therapy	Implicit	Generation	Implicit	Enc-Dec	SL	
Jh21	Therapy	Emotion Labels	Retrieval, Generation	Simulated User	Modular	RL	BERT, GPT
Ha21	Argumentation	DAs	Rule-based	Implicit	Modular	SL	
Wu21	Social Good	Implicit	Generation	Implicit	Dec-only	SL	GPT2
Ta19	Misc	Keywords	Retrieval	Implicit	Modular	SL	

Table 2: Categorization of methods (non-exhaustive) for social influence dialogue. We only cover papers that explicitly design a dialogue system. NLG: Natural Language Generation, PLM: Pretrained Language Model, MIBT: Multi-Issue Bargaining Task, E-Com: E-Commerce, DA: Dialogue Act, Enc: Encoder, Dec: Decoder, SL: Supervised Learning, RL: Reinforcement Learning. Methods that use RL usually apply it in conjunction with SL.

pressed by DAs are insufficient for capturing semantics like emotion, small talk, and appeal. To better incorporate them, researchers have relied on additional utterance-level annotations grounded in prior theories in social influence contexts (Wang et al., 2019; Chawla et al., 2021b). These strategies have been used in conjunction with DAs (Zhou et al., 2019; Joshi et al., 2020).

4.2 Language Generation

An important aspect of the system design is an effective way to realize the language, that is, to generate the next response so that it portrays the desired strategic behaviors. Borrowing from task-oriented and open-domain research, existing dialogue models for social influence use a variety of methods to generate the final system response.

Templates and retrieval methods: Predefined templates and response retrieval from the training data simplify the generation pipeline, improving controllability and modularity. He et al. (2018) used templates in their generator which are later filled by retrieving similar responses from the data. This allowed the authors to explore supervised and reinforcement learning at the level of DAs for the influence strategy of the system.

Conditional Generation: Text generation methods result in more diverse responses, but negatively impact the controllability and interpretability. Prior work relies on autoregressive text generation conditioned on the dialogue history, non-conversational context, and additional annotations. These are either encoder-decoder networks (Lewis et al., 2017; Li et al., 2020; Joshi et al., 2020) or use a decoder-only design (Li et al., 2020). A useful future direction is to combine generation with retrieval for knowledge-grounded settings like argumentation. Similar methods have been explored for other NLP tasks like open-domain question answering and question generation (Lewis et al., 2020).

4.3 Partner Modeling

Partner modeling refers to inferring the mental states of the partner based on the conversation. For example, understanding the cause that the persuadee cares about in the PersuasionForGood context, or inferring the priorities of the partner in DealOrNoDeal negotiations. Building an accurate partner model is essential in social influence settings for guiding the decision-making of the system (Baarslag et al., 2013; Zhang et al., 2020b). Hence, we discuss various ways in which prior work tackles partner modeling.

Implicit: A majority of the efforts do not explicitly model the behavior of the partner but instead, this behavior implicitly guides the next response of the sequence-to-sequence dialogue system pipeline.

Simulated User: Lewis et al. (2017) trained a simulated user on the available data in a supervised manner. This was then used to further train the dialogue system. Instead of inferring mental states explicitly, this takes a more behavioral approach of estimating the future actions of the partner and using these for training via reinforcement learning.

Dialogue Act Look-Ahead: With a similar idea, Zhang et al. (2020b) proposed OPPA model with a *look-ahead* based partner modeling strategy at the level of DAs. At each step, OPPA first estimates the user's future DA, which is then used to select the next DA of the system. The authors found significant improvements on the DealOrNoDeal task. Yang et al. (2021) used a similar method for buyer-seller negotiations. Taking a different approach, Chawla et al. (2022) trained a ranking model to directly predict the hidden preferences of the partner in a multi-issue negotiation. Instead of predicting future actions, these methods assume that the partner's behavior can be explained by their context and goals in the dialogue. However, this approach is yet to be used in an end-to-end system.

4.4 Training

Architecture Choices: One crucial aspect is the *architecture design*: End-to-end (Lewis et al., 2017; Radford et al., 2019) vs Modular (He et al., 2018). While end-to-end methods improve the diversity and need less manual effort, a modularized design enhances controllability and explainability. Perhaps, this is why modular methods are popular in large-scale models (Hadfi et al., 2021). Improving the control of desired variables such as topics, strategy, or emotion in the end-to-end methods is an open area of research and is yet to be explored for social influence dialogue systems.

Supervised Learning (SL) and Reinforcement Learning (RL): Zhou et al. (2019) used SL to train a hierarchical encoder-decoder for generating the next response and used Finite State Transducers (FSTs) to encode the historic sequence of DAs and persuasive strategies into the model, showing improvements in negotiation and persuasion tasks. The performance was later improved by Joshi et al. (2020), who replaced FSTs with Graph Neural Networks to better model the interdependencies. Others have relied on RL to explicitly optimize the model on task-specific objective outcomes. While SL trains the model to mimic the average human behavior, RL techniques, such as those based on REINFORCE (Williams, 1992), allow the system to explore its own strategies in the wild while being guided by one or more overall reward metrics. Lewis et al. (2017) used RL in negotiations, with the final points scored in the agreed deal as the reward. More recent work employed RL to incorporate simplistic partner models into the decision-making process of the dialogue system, showing improvements in negotiation tasks (Zhang et al., 2020b; Yang et al., 2021).

Multi-tasking and Pretraining: Limited efforts have also explored multi-tasking and pretrained language models for social influence dialogue systems, which provide promising ways to deal with the challenge of insufficient training data. Liu (2021) trained a sequence-to-sequence transformer on a mix of Cornell Movie Dialogue corpus (Danescu-Niculescu-Mizil and Lee, 2011) and psychotherapy data. Li et al. (2020) fine-tuned the GPT model (Radford et al., 2018), while employing multi-tasking to incorporate intents and slots for both the human and the system. Wu et al. (2021) recently introduced ARDM which uses GPT2 (Radford et al., 2019) to separately encode the utterances of the human and the dialogue system, reducing the reliance on additional annotations.

5 Discussion and Recommendations

Past few years have seen an exciting progress in social influence dialogue systems. However, building sophisticated and practically useful systems remains a challenging endeavor. Several limitations still exist that must be addressed. To guide future work, we now discuss the key challenges and provide our recommendations.

Need for unifying the efforts: One challenge in this space has been the lack of large-scale datasets for model training. Social influence tasks are complex for crowdsourcing workers to understand and to participate in. Hence, prior work used extensive instructions and tutorials, making the study expensive and time consuming (Wang et al., 2019; Chawla et al., 2021b). To address this, we recommend the researchers to aim for a more unified view of the efforts in social influence.

First, this would encourage researchers to adopt the best practices from other social influence scenarios. For instance, most datasets miss out on user attributes like demographics and personality, which are crucial in social influence scenarios (Stuhlmacher and Walters, 1999; Bogaert et al., 2008). Most datasets also ignore the partner perception after the interaction is over. This can result in misleading conclusions about the model performance, where the models perform well objectively, but hurt the relationship with their partners, and thus, negatively impacting practical utility (Aydoğan et al., 2020).

Secondly, a holistic outlook will promote transfer learning and domain adaptation. Our taxonomy for datasets (Table 1) governs the way systems must be modeled and trained. Task structure is crucial to understand whether the model can learn from the utterances of all parties or just one. Further, understanding the context definition guides how it must be encoded. Hence, one interesting future direction is joint training on datasets with similar structure and context definition.

Finally, progress in task-oriented and open-domain systems can inspire more unified modeling for social influence tasks involving multiple skills in the same interaction (e.g. a combination of negotiation and persuasion tactics as common in realistic scenarios). Roller et al. (2020) blend various open-domain tasks to address multiple challenges together (e.g., persona-based, knowledge-enriched, etc.). Hosseini-Asl et al. (2020) concatenate structured and unstructured data in task-oriented dialogues, and unify task-oriented dialogue system building to be a single sequence generation problem. Future work should explore similar unified approaches for social influence settings as well, especially since these tasks follow a common conceptual foundation (Figure 1), with similar evaluation and theoretical principles (Cialdini, 2009).

To encourage this unified view, we encapsulate our insights from this survey effort in a theoretical framework, which is presented in Appendix C. The framework covers key components for designing a social influence dialogue task, including system attributes, target audience, underlying modeling techniques, and evaluation mechanisms.

Theory integration: Most modeling efforts are based on crowdsourced datasets. Since crowdsourcing workers may not exhibit optimal strategies, supervised training on these datasets is fundamentally insufficient to build an effective system for applications like pedagogy (teaching social skills to students). Unfortunately, this holds regardless of how system strategy and partner model are designed. Further, using RL to optimize on objective rewards is also not expected to be enough to reliably learn complex influence capabilities, especially when the reward is restrictive.

To address this, we recommend to tap into the vast amount of research effort in social sciences and psychology on building theories for social influence (Cameron, 2009; Giles, 2016; Lewicki et al., 2016; Cialdini and Goldstein, 2004). Instead of solely relying on the collected data, future work should consider leveraging fundamentals from this research to guide the dialogue policy. Previous works have studied resistance to social influence (Knowles and Linn, 2004; Dal Cin et al., 2004; Petty and Cacioppo, 1977; Ahluwalia, 2000). Rucker et al. (2004) found that people resist persuasion differently depending on their beliefs, suggesting personalizing the social influence process. One can also employ the politeness theory (Brown and Levinson, 1978) and model the participants' *face* acts to better understand users in social influence contexts (Dutt et al., 2020).

Task Evaluation: Another key limitation of existing work is the lack of a comprehensive evaluation. Prior work majorly focused on objective metrics which only provides a limited view of the model performance. A comprehensive evaluation is challenging since it must consider partner perception along with objective outcomes. Building user simulators could potentially alleviate this problem (Li et al., 2016; Jain et al., 2018; Shi et al., 2019). Most existing simulators are developed for task-oriented systems which follow a certain agenda. Future research should study how to use partner modeling to build social influence user simulators for more efficient and accurate task evaluation (He et al., 2018; Yang et al., 2020). For instance, one could potentially design different user personalities and simulate the change in user's beliefs, opinions, and attitudes accordingly (Yang et al., 2021).

Multimodal systems: Being a core function of human communication, social influence occurs not just through text, but through all possible modalities. Schulman and Bickmore (2009) showed that embodied agents achieve better persuasion results than text-only agents. Other studies have recognized the importance of emotion in social influence tasks (Asai et al., 2020; Chawla et al., 2021a). Nguyen et al. (2021) proposed a speech dataset in debates and study the influence of spoken tactics on persuasiveness across genders. Given these findings, we encourage interdisciplinary efforts in the future to explore the developement of multimodal social influence agents.

Knowledge-enriched systems: Social influence tasks often involve constantly-changing world knowledge such as organization facts and news. Often, the system's internal state (e.g., the change of task setting from one set of products to a different set) needs to be updated. Retraining the entire system is costly to maintain after the initial development. Recent work has proposed to augment the dialogue system with internet-search ability to generate more factual and updated responses in open-domain dialogues (Komeili et al., 2021). Future efforts in this direction will benefit social influence dialogue systems as well.

6 Conclusions

We introduced the category of social influence dialogue systems that aim to influence their partners through dialogue. We presented a survey of the recent prior work in this space, compiling datasets and methods across diverse application domains. We pointed out key limitations in existing methodologies and proposed promising directions for designing more sophisticated systems in the future. Our survey reveals that although substantial progress has been made, this is still an emerging research area. We hope our work inspires more dedicated interdisciplinary effort and discussion, which is necessary for making progress in this space.

7 Broader Impact and Ethical Considerations

Social influence is ubiquitous in everyday life. Research on how we use influence in all aspects of our lives spans a number of fields, including social psychology, communication, consumer behavior, behavioral change, and behavioral economics. This research has led to crucial findings about the strategies of social influence and how they impact our decision-making. Over the past few decades, research has accumulated and demonstrated the effectiveness of using various strategies across contexts and domains. Prominent examples include core principles of social influence by Cialdini from social psychology: reciprocity, commitment and consistency, social proof, liking and attractiveness, authority, and scarcity (Cialdini, 2009). Further, communication strategies used in persuasion and general social influence contexts include credibility appeals, two-sided argumentation, emotional tactics, and appeals to social norms, among others (Cameron, 2009; O'keefe, 2015).

First, the well-studied principles in social influence research can guide the development of effective dialogue systems with influence capabilities. In fact, many of the strategies found in the datasets developed for social influence tasks (Section 3) directly map to the principles laid out by Cialdini, for instance, credibility and emotional appeal in PersuasionForGood dataset (Wang et al., 2019) and reciprocity observed in CaSiNo negotiation dataset (Chawla et al., 2021b). Second, research in social influence dialogue systems provides novel datasets on human-human and human-machine communication, and therefore, holds a great potential to advance theories of human cognition and influence processes (Gratch et al., 2015). The datasets and subsequent analyses can further contribute new theoretical insights to social influence research.

Although dialogue systems have already been used in a number of applications involving chatbots and AI assistants, advancements in social influence dialogue systems can help to bridge the gap between our existing task definitions and a number of other real-world applications. For instance, realistic customer support interactions often involve active behaviors from both the support agent and the user where the agent uses social cues for improved customer satisfaction and retention, while the user attempts to address their queries. These settings naturally involve aspects of social influence, unlike traditional task-oriented definitions where the dialogue system plays a passive role to assist the human users. As discussed earlier, social influence dialogue systems can positively help to advance other areas as well. In therapy domain, these systems can assist in various psychological treatments such as by increasing the willingness to disclose (Lucas et al., 2014). In pedagogy, they can help to make social skills training more accessible (Johnson et al., 2019).

While we think about these applications, it is crucial to also lay out proper ethical guidelines to avoid any misuse of these systems. Primary concerns are around the use of deception (e.g. in Diplomacy and other negotiation tasks), emotional appeals (e.g. in persuasion), and behavior change (e.g. in conversational recommendations).

To mitigate possible misuse scenarios or unintended harms, we now lay out a few ethical guidelines which also apply to dialogue research in general. First, rigorous attempts must be made to ensure that the data collection, design processes, and evaluations, strictly abide by the guidelines and regulations laid out by the relevant Institutional Review Board (IRB). Second, the research team needs to develop a thorough plan to monitor and understand the behaviors of the developed systems before deployment. This includes identifying the goals of the dialogue system, identifying potential toxic language use, and any discriminatory behaviors. Third, investment into improved data collection practices, along with explainable and controllable dialogue systems can help identify these issues early on and allow manipulation to avoid them. Fourth, we argue that transparency is the key.

All stakeholders must be made aware of the goals and design objectives of the system, along with any known misbehaviors or potential risks. The users must also be informed of any data collected during the deployment phase. Lastly, we believe that continuous monitoring of dialogue systems is necessary to ensure that the system performs consistently and does not diverge to unexpected conditions that may incur offensive or discriminative actions. We hope that our work promotes a more systematic study of social influence dialogue systems, which in turn will help to tackle the ethical concerns in a more principled way.

8 Limitations

Literature Search: We presented a survey of efforts in social influence dialogue systems. Although every attempt was made to provide the readers with a comprehensive overview of the research in this space, our work does not claim exhaustiveness in the covered literature and it is likely that we missed out on other relevant research in this space. **Intention for influence**: The datasets and tasks covered in this literature review focus on scenarios where social influence is *intentional by design*. However, social influence can also be *unintentional*, that is, interactions between humans and machines can have unintended influence on the attitudes, behaviors, or feelings of the human user (Gass, 2015). For instance, changes in topic preferences after interacting with a system on a variety of topics, or incorporating biases after interacting with a biased system. As we continue to make an unprecedented progress towards AI systems that interact with humans via natural means of communication, we must also take into account the unintended influence on the users of the underlying technology. We hope that our work motivates researchers to study these effects methodically in the future.

Acknowledgments

We would like to thank our colleagues at the University of Southern California, Columbia University, and the University of California Davis, along with fellow researchers with whom we interacted at conferences, for all their comments and helpful discussions that have shaped this project. We also thank the anonymous reviewers for their valuable time and feedback. Our research was, in part, sponsored by the Army Research Office and was accomplished under Cooperative Agreement Number W911NF-20-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Rohini Ahluwalia. 2000. Examination of psychological processes underlying resistance to persuasion. *Journal of Consumer Research*, 27(2):217–232.

Tim Althoff, Kevin Clark, and Jure Leskovec. 2016. Large-scale analysis of counseling conversations: An application of natural language processing to mental health. *Transactions of the Association for Computational Linguistics*, 4:463–476.

Tim Althoff, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2014. How to ask for a favor: A case study on the success of altruistic requests. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8, pages 12–21.

Sara Asai, Koichiro Yoshino, Seitaro Shinagawa, Sakriani Sakti, and Satoshi Nakamura. 2020. Emotional speech corpus for persuasive dialogue system. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 491–497.

Nicholas Asher, Julie Hunter, Mathieu Morey, Benamara Farah, and Stergos Afantenos. 2016. Discourse structure and dialogue acts in multiparty dialogue: the stac corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2721–2727.

Reyhan Aydoğan, Tim Baarslag, Katsuhide Fujita, Johnathan Mell, Jonathan Gratch, Dave de Jonge, Yasser Mohammad, Shinji Nakadai, Satoshi Morinaga, Hirotaka Osawa, et al. 2020. Challenges and main results of the automated negotiating agents competition (anac) 2019. In *Multi-agent systems and agreement technologies*, pages 366–381. Springer.

Tim Baarslag, Mark Hendrikx, Koen Hindriks, and Catholijn Jonker. 2013. Predicting the performance of opponent models in automated negotiation. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 2, pages 59–66. IEEE.

Linda Babcock and Sara Laschever. 2009. *Women don't ask: Negotiation and the gender divide*. Princeton University Press.

Sandy Bogaert, Christophe Boone, and Carolyn Declerck. 2008. Social value orientation and cooperation in social dilemmas: A review and conceptual model. *British Journal of Social Psychology*, 47(3):453–480.

Maria Boritchev and Maxime Amblard. 2021. Ding–a corpus of transcriptions of real-life, oral, spontaneous multi-party dialogues between french-speaking players of catan. In *Journées du GdR LIFT*.

Penelope Brown and Stephen C Levinson. 1978. Universals in language usage: Politeness phenomena. In *Questions and politeness: Strategies in social interaction*, pages 56–311. Cambridge University Press.

Kenzie A Cameron. 2009. A practitioner's guide to persuasion: An overview of 15 selected persuasion theories, models and frameworks. *Patient education and counseling*, 74(3):309–317.

Shelly Chaiken. 1987. The heuristic model of persuasion. In *Social influence: the ontario symposium*, volume 5, pages 3–39.

Kushal Chawla, Rene Clever, Jaysa Ramirez, Gale Lucas, and Jonathan Gratch. 2021a. Towards emotion-aware agents for negotiation dialogues. In *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–8. IEEE.

Kushal Chawla, Gale Lucas, Jonathan May, and Jonathan Gratch. 2020. Exploring early prediction of buyer-seller negotiation outcomes. *arXiv preprint arXiv:2004.02363*.

Kushal Chawla, Gale M Lucas, Jonathan May, and Jonathan Gratch. 2022. Opponent modeling in negotiation dialogues by related data adaptation. *arXiv preprint arXiv:2205.00344*.

Kushal Chawla, Jaysa Ramirez, Rene Clever, Gale Lucas, Jonathan May, and Jonathan Gratch. 2021b. Casino: A corpus of campsite negotiation dialogues for automatic negotiation systems. *arXiv preprint arXiv:2103.15721*.

R. B. Cialdini. 2009. *Influence: Science and Practice*. fifth ed. Pearson/Allyn & Bacon, Boston, MA.

Robert B Cialdini and Noah J Goldstein. 2004. Social influence: Compliance and conformity. *Annual review of psychology*, 55(1):591–621.

Cristian D.-N.-M., Lillian Lee, Bo Pang, and Jon Kleinberg. 2012. Echoes of power: Language effects and power differences in social interaction. In *Proceedings of the 21st international conference on World Wide Web*, pages 699–708.

Sonya Dal Cin, Mark P Zanna, and Geoffrey T Fong. 2004. Narrative persuasion and overcoming resistance. *Resistance and persuasion*, 2:175–191.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 76–87.

Orianna Demasi, Marti A Hearst, and Benjamin Recht. 2019. Towards augmenting crisis counselor training by improving message retrieval. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 1–11.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2021. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*, 54(1):755–810.

David DeVault, Johnathan Mell, and Jonathan Gratch. 2015. Toward natural turn-taking in a virtual human negotiation agent. In *AAAI Spring Symposia*. Citeseer.

Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander H Miller, Arthur Szlam, and Jason Weston. 2016. Evaluating prerequisite qualities for learning end-to-end dialog systems. In *4th International Conference on Learning Representations, ICLR 2016*.

Sebastian Duerr and Peter A Gloor. 2021. Persuasive natural language generation–a literature review. *arXiv preprint arXiv:2101.05786*.

Esin Durmus and Claire Cardie. 2019. A corpus for modeling user and language effects in argumentation on online debating. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Ritam Dutt, Rishabh Joshi, and Carolyn Penstein Rosé. 2020. Keeping up appearances: Computational modeling of face acts in persuasion oriented discussions. *arXiv preprint arXiv:2009.10815*.

Subhabrata Dutta, Jeevesh Juneja, Dipankar Das, and Tanmoy Chakraborty. 2022. Can unsupervised knowledge transfer from social discussions help argument mining? *arXiv preprint arXiv:2203.12881*.

Ryo Egawa, Gaku Morio, and Katsuhide Fujita. 2019. Annotating and analyzing semantic role of elementary units and relations in online persuasive arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 422–428.

Chaim Fershtman. 1990. The importance of the agenda in bargaining. *Games and Economic Behavior*, 2(3):224–238.

Tommaso Fornaciari and Massimo Poesio. 2012. Decour: a corpus of deceptive statements in italian courts. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1585–1590.

Jamie Fraser, Ioannis Papaioannou, and Oliver Lemon. 2018. Spoken conversational ai in video games: Emotional dialogue management increases user engagement. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*, pages 179–184.

Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371–1374.

Robert H Gass. 2015. Social influence, sociology of. In *International Encyclopedia of the Social and Behavioral Sciences*, pages 348–354. Elsevier.

Howard Giles. 2016. *Communication accommodation theory: Negotiating personal relationships and social identities across contexts*. Cambridge University Press.

Jonathan Gratch, David DeVault, Gale M Lucas, and Stacy Marsella. 2015. Negotiation as a challenge problem for virtual humans. In *International Conference on Intelligent Virtual Agents*, pages 201–215. Springer.

Rafik Hadfi, Jawad Haqbeen, Sofia Sahab, and Takayuki Ito. 2021. Argumentative conversational agents for online discussions. *Journal of Systems Science and Systems Engineering*, 30(4):450–464.

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2333–2343.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Alankar Jain, Florian Pecune, Yoichi Matsuyama, and Justine Cassell. 2018. A user simulator architecture for socially-aware conversational agents. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*, pages 133–140.

Jiun-Hao Jhan, Chao-Peng Liu, Shyh-Kang Jeng, and Hung-Yi Lee. 2021. Cheerbots: Chatbots toward empathy and emotionusing reinforcement learning. *arXiv preprint arXiv:2110.03949*.

Changzhen Ji, Xin Zhou, Yating Zhang, Xiaozhong Liu, Changlong Sun, Conghui Zhu, and Tiejun Zhao. 2020. Cross copy network for dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1900–1910.

Emmanuel Johnson, Gale Lucas, Peter Kim, and Jonathan Gratch. 2019. Intelligent tutoring system for negotiation skills training. In *International Conference on Artificial Intelligence in Education*, pages 122–127. Springer.

Rishabh Joshi, Vidhisha Balachandran, Shikhar Vashishth, Alan Black, and Yulia Tsvetkov. 2020. Dialograph: Incorporating interpretable strategy-graph networks into negotiation dialogues. In *International Conference on Learning Representations*.

Sung Soo Kim, Stan Kaplowitz, and Mark V Johnston. 2004. The effects of physician empathy on patient satisfaction and compliance. *Evaluation & the health professions*, 27(3):237–251.

Eric S Knowles and Jay A Linn. 2004. *Resistance and persuasion*. Psychology Press.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2021. Internet-augmented dialogue generation. *arXiv preprint arXiv:2107.07566*.

Yi-Chieh Lee, Naomi Yamashita, and Yun Huang. 2020. Designing a chatbot as a mediator for promoting deep self-disclosure to a real mental health professional. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–27.

Roy J Lewicki, Bruce Barry, and David M Saunders. 2016. *Essentials of negotiation*. McGraw-Hill.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *EMNLP*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. *Advances in neural information processing systems*, 31.

Xinmeng Li, Wansen Wu, Long Qin, and Quanjun Yin. 2021. How to evaluate your dialogue models: A review of approaches. *arXiv preprint arXiv:2108.01369*.

Xiujun Li, Zachary C Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *arXiv preprint arXiv:1612.05688*.

Yu Li, Kun Qian, Weiyan Shi, and Zhou Yu. 2020. End-to-end trainable non-collaborative dialog system. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8293–8302.

Kai-Hui Liang, Patrick L Lange, Yoo Jung Oh, Jingwen Zhang, Yoshimi Fukuoka, and Zhou Yu. 2021. Evaluation of in-person counseling strategies to develop physical activity chatbot for women. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 32–44.

Wang Liao, Jingwen Zhang, Yoo Jung Oh, and Nicholas A Palomares. 2021. Linguistic accommodation enhances compliance to charity donation: The role of interpersonal communication processes in mediated compliance-gaining conversations. *Journal of Computer-Mediated Communication*, 26(3):167–185.

Houjun Liu. 2021. Towards automated psychotherapy via language modeling. *arXiv preprint arXiv:2104.10661*.

Gale M Lucas, Jonathan Gratch, Aisha King, and Louis-Philippe Morency. 2014. It's only a computer: Virtual humans increase willingness to disclose. *Computers in Human Behavior*, 37:94–100.

Huyen Nguyen, Ralph Vente, David Lupea, Sarah Ita Levitan, and Julia Hirschberg. 2021. Acoustic-prosodic, lexical and demographic cues to persuasiveness in competitive debate speeches. *Proc. Interspeech 2021*, pages 1034–1038.

Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, Vinay Adiga, and Erik Cambria. 2021. Recent advances in deep learning based dialogue systems: A systematic survey. *arXiv preprint arXiv:2105.04387*.

Tim Norfolk, Kamal Birdi, and Deirdre Walsh. 2007. The role of empathy in establishing rapport in the consultation: a new model. *Medical education*, 41(7):690–697.

Daniel J O'keefe. 2015. *Persuasion: Theory and research*. Sage Publications.

Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. 2021. Dialogue graph modeling for conversational machine reading. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3158–3169.

Denis Peskov, Benny Cheng, Ahmed Elgohary, Joe Barrow, Cristian Danescu-Niculescu-Mizil, and Jordan Boyd-Graber. 2020. It takes two to lie: One to lie, and one to listen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3811–3854.

Richard E Petty and John T Cacioppo. 1977. Forewarning, cognitive responding, and resistance to persuasion. *Journal of Personality and social Psychology*, 35(9):645.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381.

Lisa Rashotte. 2007. Social influence. *The Blackwell encyclopedia of sociology*.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Derek D Rucker, Zakary L Tormala, and Richard E Petty. 2004. Individual differences in resistance to persuasion: The role of beliefs and meta-beliefs. *Resistance and persuasion*, page 83.

Daniel Schulman and Timothy Bickmore. 2009. Persuading users through counseling dialogue with a conversational agent. In *Proceedings of the 4th international conference on persuasive technology*, pages 1–8.

Weiyan Shi, Kun Qian, Xuewei Wang, and Zhou Yu. 2019. How to build user simulators to train rl-based dialog systems. *arXiv preprint arXiv:1909.01388*.

Zhouxing Shi and Minlie Huang. 2019. A deep sequential model for discourse parsing on multi-party dialogues. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7007–7014.

Manjira Sinha and Tirthankar Dasgupta. 2021. Predicting success of a persuasion through joint modeling of utterance categorization. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3423–3427.

Alice F Stuhlmacher and Amy E Walters. 1999. Gender differences in negotiation outcome: A meta-analysis. *Personnel Psychology*, 52(3):653–677.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th international conference on world wide web*, pages 613–624.

Michael Tanana, Kevin A Hallgren, Zac E Imel, David C Atkins, and Vivek Srikumar. 2016. A comparison of natural language processing methods for automated coding of motivational interviewing. *Journal of substance abuse treatment*, 65:43–50.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5624–5634.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: determining support or opposition from congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 327–335.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. 2019. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5635–5649.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Qingyang Wu, Yichi Zhang, Yu Li, and Zhou Yu. 2021. Alternating recurrent dialog model with large-scale pre-trained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1292–1301.

William Wunderle. 2007. How to negotiate in the middle east. *Military review*, 87(2):33.

Atsuki Yamaguchi, Kosui Iwasa, and Katsuhide Fujita. 2021. Dialogue act-based breakdown detection in negotiation dialogues. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 745–757.

Runzhe Yang, Jingxiao Chen, and Karthik Narasimhan. 2020. Improving dialog systems for negotiation with personality modeling. *arXiv preprint arXiv:2010.09954*.

Runzhe Yang, Jingxiao Chen, and Karthik Narasimhan. 2021. Improving dialog systems for negotiation with personality modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 681–693.

Denis Yarats and Mike Lewis. 2018. Hierarchical text generation and planning for strategic dialogue. In *International Conference on Machine Learning*, pages 5591–5599. PMLR.

Min-Hsuan Yeh and Lun-Wei Ku. 2021. Lying through one's teeth: A study on verbal leakage cues. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4504–4510.

Dian Yu, Yulia Tyshchuk, Heng Ji, and William Wallace. 2015. Detecting deceptive groups using conversations and network analysis. In *53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL-IJCNLP 2015*, pages 857–866. Association for Computational Linguistics (ACL).

Jingwen Zhang, Yoo Jung Oh, Patrick Lange, Zhou Yu, Yoshimi Fukuoka, et al. 2020a. Artificial intelligence chatbot behavior change model for designing artificial intelligence chatbots to promote physical activity and a healthy diet. *Journal of medical Internet research*, 22(9):e22845.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Zheng Zhang, Lizi Liao, Xiaoyan Zhu, Tat-Seng Chua, Zitao Liu, Yan Huang, and Minlie Huang. 2020b. Learning goal-oriented dialogue policy with opposite agent awareness. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 122–132.

Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. 2020c. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences*, pages 1–17.

Yiheng Zhou, Yulia Tsvetkov, Alan W Black, and Zhou Yu. 2019. Augmenting non-collaborative dialog systems with explicit semantic and strategic dialog history. In *International Conference on Learning Representations*.

A Literature Compilation

In this section, we provide details about how the literature was curated for our survey. We hope this helps the overall reproducibility and also guides similar studies in the future. We followed a simple two-stage process. First, we compiled the relevant datasets that capture various forms of social influence across diverse domains (presented in Section 3) and then, we compiled the techniques developed on these datasets (presented in Section 4).

Step I - Datasets: Our objective was to gather datasets that (by design) capture forms of social influence. We primarily focused on dialogue interactions but include the datasets based on transcripts from multimodal interactions as well. Given the large breadth of research in this space across a number of domains, our collection is not exhaustive but is rather restricted to the following sources.

We surveyed the past 6 years of *ACL conference proceedings. We then covered several online repositories of dialogue data to capture datasets published at other venues. This includes ParlAI[1], Huggingface[2], NLP-Progress[3], and Convokit[4]. Further, we revisited several recent surveys in dialogue systems and Natural Language Generation (NLG) research (Huang et al., 2020; Zhang et al., 2020c; Ni et al., 2021; Duerr and Gloor, 2021). Datasets that were categorized as task-oriented or open-domain in these surveys but also contain some aspects of social influence have been included in our discussion. As discussed in Section 4, we also include the datasets that have not been directly used for designing dialogue systems but rather for various Natural Language Understanding (NLU) subtasks that can be crucial for the eventual development of dialogue systems in this space. Finally, we also reviewed the citation graphs of the collected papers from Google Scholar. Overall, we ended up with 22 dataset papers, spanning 12 publication venues, 4 languages, and 7 application domains.

Step II - Methods: Compiling the methodological progress was based on the models developed on the curated datasets. For this purpose, we simply reviewed the citations of all the dataset papers using Google Scholar.

B Datasets

A comprehensive list of the available datasets for investigating social influence in dialogues is provided in Table 3. For each dataset, we mention the application domain, source, key statistics, as well as the available metadata and annotations apart from the conversation logs.

C Five Stages for Designing Social Influence Dialogue Systems

We develop a five-stage framework to summarize our recommendations for future work. These stages cover key decisions in the design of a dialogue system in this space, encouraging a holistic understanding of the system characteristics, target audience, underlying modeling techniques, and evaluation mechanisms. These steps are inspired by a behavior change model in healthcare research (Zhang et al., 2020a). We adapt this model to make it suitable for general social influence tasks in NLP. We present these steps in Figure 2.

---

[1] `https://github.com/facebookresearch/ParlAI`

[2] `https://huggingface.co/docs/datasets/index`

[3] `http://nlpprogress.com/english/dialogue.html`

[4] `https://convokit.cornell.edu/documentation/datasets.html`

Name (Citation)	Domain	Source	Key Statistics	Metadata & Annotations
STAC (Asher et al., 2016)	Games	Crowdsource	Dialogues: 1081 Turns/Dialogue: 8.5 Tokens/Turn: 4.2	Dialogue Acts; Discourse Structures
Diplomacy (Peskov et al., 2020)	Games	Crowdsource	Games: 12 Messages/Game: 1440.8 Words/Message: 20.79^{+}	Intended and perceived truthfulness; Participant demographics and self-assessment of lying abilities; Ground-truth betrayals
DinG (Boritchev and Amblard, 2021)	Games	University game night logs	Games: 10 Turns/Game: 2357.5	Annotated question-answer pairs
Tabletop (DeVault et al., 2015)	MIBT	Face-to-face, Wizard-of-Oz	Face-to-face Dialogues: 89 Wizard-of-Oz Dialogues: 30	Participant demographics; Subjective questionnaire-based assessment
DealOrNoDeal (Lewis et al., 2017)	MIBT	Crowdsource	Dialogues: 5808 Turns/Dialogue: 6.6 Tokens/Turn: 7.6	—
CaSiNo (Chawla et al., 2021b)	MIBT	Crowdsource	Dialogues: 1030 Utterances/Dialogue: 11.6 Tokens/Utterance: 22	Participant demographics and personality traits; Outcome satisfaction; Partner perception; Strategy Annotations
JobInterview (Yamaguchi et al., 2021)	MIBT	Crowdsource	Dialogues: 2639 Turns/Dialogue: 12.7 Words/Turn: 6.1	Dialogue acts
PersuasionforGood (Wang et al., 2019)	Social Good	Crowdsource	Dialogues: 1017 Turns/Dialogue: 10.4 Words/Turn: 19.4	Participant sociodemographics, personality, and engagement in the conversation; Strategy annotations; Dialogue Acts
CraigslistBargain (He et al., 2018)	E-commerce	Crowdsource	Dialogues: 6682 Turns/Dialogue: 9.2 Tokens/Turn: 15.5	Dialogue Acts
AntiScam (Li et al., 2020)	E-commerce	Crowdsource	Dialogues: 220 Turns/Dialogue: 12.4 Words/Turn: 11.1	Dialogue Acts; Semantic Slots
Motivational Interviewing (Tanana et al., 2016)	Therapy & Support	Psychotherapy session logs	Sessions: 341 Utterances/Session: 513.2 Words/Utterance: 9.7	Behavior annotations
SMS Counseling (Althoff et al., 2016)	Therapy & Support	SMS chat logs	Dialogues: 80,885 Messages/Dialogue: 42.6* Words/message: 19.2*	Post-conversation assessment for both the counselor (e.g. suicide risk, main issue etc.) and user (how they feel afterwards)
EmpatheticDialogues (Rashkin et al., 2019)	Therapy & Support	Crowdsource	Dialogues: 24,850 Utterances/Dialogue: 4.3 Words/Utterance: 15.2	—
Hotline Counseling (Demasi et al., 2019)	Therapy & Support	Synthetic Transcripts	Dialogues: 254 Messages/Dialogue: 40-60	Paraphrases by trained counselors
mPED (Liang et al., 2021)	Therapy & Support	Physical activity clinical trials	Sessions: 107 Turns/Session: 423.5 Minutes/Session: 28.8	Demographics; Physical activity related pre and post surveys; Strategy annotations
Congressional Debates (Thomas et al., 2006)	Argumentation	U.S. Congressional transcripts	Debates: 53 Speech segments/Debate: 72.8	Ground-truth label with each speech segment for support/oppose of the proposed bill
Supreme Court (D.-N.-M. et al., 2012)	Argumentation	Oyez.org transcripts	Cases: 7700 Utterances/Case: 220.8	Case-related metadata such as key dates, citation, parties involved, and voting results
DeCour (Fornaciari and Poesio, 2012)	Argumentation	Italian court hearings	Hearings: 35 Utterances/Hearing: 173.4 Tokens/Utterance: 13.9$^{\#}$	Metadata for testimonies like place, date, demographics; Hearing related details; Truthfulness annotations
ChangeMyView (Tan et al., 2016)	Argumentation	Reddit	Discussion Trees: 20,626 Nodes/Tree: 61.1	The original post with initial arguments and explicitly recognized successful persuasive arguments from the opposing side
DDO Debates (Durmus and Cardie, 2019)	Argumentation	debate.org logs	Debates: 78,376 Messages/Debate: 7.7	User demographics; Debate metadata like dates and category; Audience votes and comments
Court Debates (Ji et al., 2020)	Argumentation	China Court transcripts	Dialogues: 260,190 Utterances/Dialogue: 13.9	—
Target-Guided (Tang et al., 2019)	Miscellaneous	Crowdsource	Dialogues: 9939 Utterances/Dialogue: 11.4	—

Table 3: Social Influence Dialogue Corpora, grouped by task domains and sorted by publishing year within a domain. All statistics of the form X/Y denote average numbers. MIBT: Multi-Issue Bargaining Task. *Only computed for dialogues with additional survey responses, $^{+}$Only computed for the training data. $^{\#}$Only for Speaker utterances in front of the judge (doesn't include other members of the court). Note that not all datasets listed above have been directly used for designing end-to-end dialogue systems, but instead, these have enabled research into various sub-tasks and analyses that can eventually be useful for dialogue systems in this area. Please refer to Section 3 in the main paper for a detailed discussion about these datasets and to Section 4 for information about various methods developed using them.

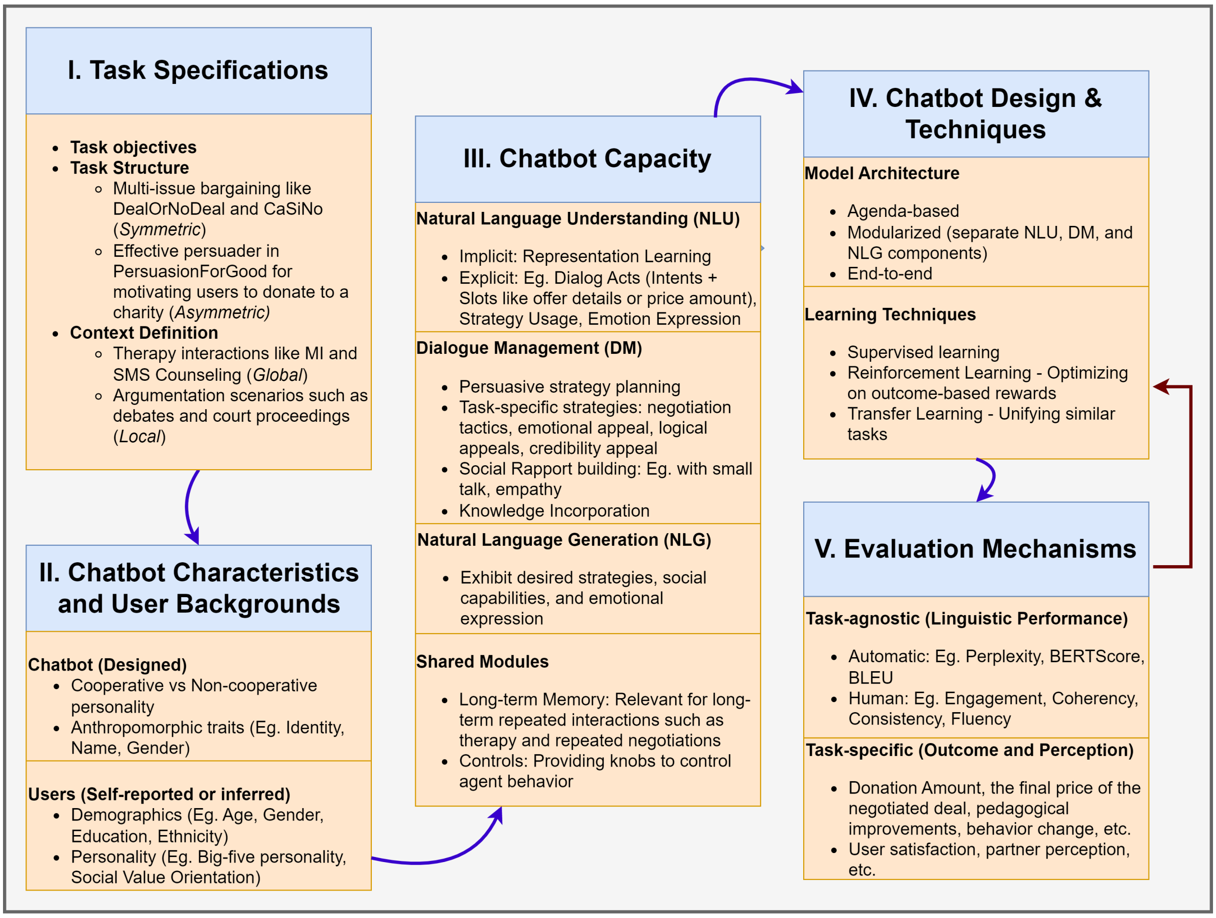


Figure 2: A theoretical model for the development of dialogue systems for social influence tasks. Curved arrows represent forward relations and the straight arrow represents the feedback. **I. Task Specifications**: Key properties that define the task in consideration and are captured by the collected dataset, **II. Chatbot Characteristics and User Backgrounds**: Attributes for the agent design and target audience, **III. Chatbot Capacity**: The desirable capabilities of the system, **IV. Chatbot Design & Techniques**: The modeling techniques to develop the dialogue system, and **V. Evaluation Mechanisms**: Metrics to evaluate system performance.

Aggregating Crowdsourced and Automatic Judgments to Scale Up a Corpus of Anaphoric Reference for Fiction and Wikipedia Texts

Juntao Yu[1], **Silviu Paun**[2]*, **Maris Camilleri**[3], **Paloma Carretero Garcia**[3],
Jon Chamberlain[1], **Udo Kruschwitz**[4] and **Massimo Poesio**[3]
[1]University of Essex, UK; [2]Amazon Research, Romania;
[3]Queen Mary University of London, UK; [4]University of Regensburg, Germany.
`j.yu@essex.ac.uk; silviupn@amazon.com; mcamil@essex.ac.uk;pcarre@essex.ac.uk;`
`jchamb@essex.ac.uk; udo.kruschwitz@ur.de; m.poesio@qmul.ac.uk;`

Abstract

Although several datasets annotated for anaphoric reference / coreference exist, even the largest such datasets have limitations in terms of size, range of domains, coverage of anaphoric phenomena, and size of documents included. Yet, the approaches proposed to scale up anaphoric annotation haven't so far resulted in datasets overcoming these limitations. In this paper, we introduce a new release of a corpus for anaphoric reference labelled via a game-with-a-purpose. This new release[1] is comparable in size to the largest existing corpora for anaphoric reference due in part to substantial activity by the players, in part thanks to the use of a new resolve-and-aggregate paradigm to 'complete' markable annotations through the combination of an anaphoric resolver and an aggregation method for anaphoric reference. The proposed method could be adopted to greatly speed up annotation time in other projects involving games-with-a-purpose. In addition, the corpus covers genres for which no comparable size datasets exist (Fiction and Wikipedia); it covers singletons and non-referring expressions; and it includes a substantial number of long documents ($> 2K$ in length).

1 Introduction

Many resources annotated for anaphoric reference / coreference exist; but even the largest such datasets, such as ONTONOTES (Pradhan et al., 2012), have limitations. The largest resources are still medium scale (e.g., ONTONOTES (Pradhan et al., 2012) is 1.5M tokens, as is CRAFT (Cohen et al., 2017)). They only cover a limited range of domains, primarily news (as in ONTONOTES) and scientific articles (as in CRAFT), and models trained on these datasets have been shown not to generalize well to other domains (Xia and Durme, 2021).[2] The range of anaphoric phenomena covered is also narrow (Poesio et al., 2016).

Several proposals have been made to scale up anaphoric annotation in terms of size, range of domains, and phenomena covered proposed, including automatic data augmentation (Emami et al., 2019; Gessler et al., 2020; Aloraini and Poesio, 2021), and crowdsourcing combined with active learning (Laws et al., 2012; Li et al., 2020; Yuan et al., 2022) or through Games-With-A-Purpose (Chamberlain et al., 2008; Hladká et al., 2009; Bos et al., 2017; Kicikoglu et al., 2019). However, the largest existing anaphoric corpora created using Games-With-A-Purpose (e.g., (Poesio et al., 2019)) are still smaller than the largest resources created with traditional methods, and the corpora created using data augmentation techniques are focused on specific aspects of anaphoric reference. In order to use such approaches to create resources of the required scale in terms of size, variety and range of phenomena covered novel methods are required.

The first contribution of this paper is the *Phrase Detectives* 3.0 corpus of anaphoric reference annotated using a Game-With-A-Purpose. This corpus has a comparable size in tokens (1.37M) to the ONTONOTES corpus (Pradhan et al., 2012), but twice the number of markables[3]. Its annotation scheme also covers singletons and non-referring expressions; it is focused on two genres - fiction and Wikipedia articles - not covered in ONTONOTES, and for which only much smaller datasets exist; and it includes a range of documents ranging from short to fairly long ($14K$ tokens) enabling research in NLP on long documents (Beltagy et al., 2020). While ONTONOTES will remain a fundamental resource for the field in terms of size and languages it covers, we therefore hope that *Phrase Detec-*

---

[*] Work was done prior to joining Amazon research.

[1] The corpus is available at `https://github.com/dali-ambiguity/Phrase-Detectives-Corpus-3.0`

[2] The largest existing corpus for English, the 10M words PRECO (Chen et al., 2018), consists of language learning texts, but the models trained on this genre have proven to have even worse performance on other domains.

[3] The number of non-singleton markables is similar to that of ONTONOTES

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 767–781
May 2-6, 2023 ©2023 Association for Computational Linguistics

tives 3.0 will complement ONTONOTES, providing a comparable amount of data in genres so far less covered, while at the same time covering aspects of anaphoric interpretation not covered there, such as singletons and non-referring expressions.

The second contribution of the paper is a new iterative resolve-and-aggregate approach developed to 'complete' the corpus by combining crowdsourcing with automatic annotation. Only about 70% of documents in the corpus were completely annotated by the players. The proposed method (i) uses an anaphoric resolver to automatically annotate all mentions, including the few still unannotated; (ii) aggregates the resulting judgments using a probabilistic aggregation method for anaphora, and (iii) uses the resulting expanded dataset to retrain the anaphoric resolver. We show that the resolve-and-aggregate method results in models with higher accuracy than models trained using only the completely annotated data, or the full corpus not completed using the method.

2 Background

Anaphorically annotated corpora A number of anaphorically annotated datasets now exist, covering a number of languages (Hinrichs et al., 2005; Hendrickx et al., 2008; Recasens and Martí, 2010; Pradhan et al., 2012; Landragin, 2016; Nedoluzhko et al., 2016; Cohen et al., 2017; Chen et al., 2018; Bamman et al., 2020; Uryupina et al., 2020; Zeldes, 2020) and turning anaphora / coreference into a very active area of research (Pradhan et al., 2012; Fernandes et al., 2014; Wiseman et al., 2015; Lee et al., 2017, 2018; Yu et al., 2020; Joshi et al., 2020). However, only a few of these are genuinely large in terms of markables (Pradhan et al., 2012; Cohen et al., 2017), and most are focused on news. Corpora of comparable size exist for scientific articles (e.g., CRAFT (Cohen et al., 2017)), and substantially smaller ones for fiction (e.g., LitBank (Bamman et al., 2020) and *Phrase Detectives* 2 (Poesio et al., 2019)), and Wikipedia (e.g., WikiCoref (Ghaddar and Langlais, 2016) or again *Phrase Detectives* 2 (Poesio et al., 2019)). But important genres such as dialogue are barely covered (Muzerelle et al., 2014; Yu et al., 2022a). There is evidence that this concentration on a single genre, and on ONTONOTES in particular, results in models that do not generalize well (Xia and Durme, 2021).

Existing resources are also limited in terms of coverage. Most recent datasets are based on general purpose annotation schemes with a clear linguistic foundation, but especially the largest ones focus on the simplest cases of anaphora / coreference (e.g., singletons and non-referring expressions are not annotated in ONTONOTES). And the documents included in existing corpora tend to be short, with the exception of CRAFT: e.g., average document length is 329 in PRECO, 467 in ONTONOTES, 630 in ARRAU, and 753 in *Phrase Detectives*.

Scaling up anaphoric annotation One approach to scale up anaphoric reference annotation is using fully automatic methods to either annotate a dataset, such as AMALGUM (Gessler et al., 2020), or create a benchmark from scratch, such as KNOWREF (Emami et al., 2019). While entirely automatic annotation may result in datasets of arbitrarily large size, such annotations cannot expand current models' coverage to aspects of anaphoric reference they do not already handle well. And creating from scratch large-scale benchmarks for specific issues hasn't so far been shown to result in datasets reflecting the variety and richness of real texts.

Crowdsourcing has emerged as the dominant paradigm for annotation in NLP (Snow et al., 2008; Poesio et al., 2017) because of its reduced costs and increased speed in comparison with traditional annotation. But the costs for really large-scale annotation are still prohibitive even for crowdsourcing (Poesio et al., 2013, 2017). To address this issue, a number of approaches have been developed to optimize the use of crowdsourcing for coreference annotation. In particular, active learning has been used to reduce the amount of annotation work needed (Laws et al., 2012; Li et al., 2020; Yuan et al., 2022). Another issue is that anaphoric reference is a complex type of annotation whose most complex aspects require special quality control typically not available with microtask crowdsourcing.

Games-With-A-Purpose A form of crowdsourcing which has been widely used to address the issues of cost and quality is Games-With-A-Purpose (GWAP) (von Ahn, 2006; Cooper et al., 2010; Lafourcade et al., 2015). Games-With-A-Purpose is the version of crowdsourcing where labelling is created through a game, so that the reward for the workers is in terms of enjoyment rather than financial. They were proposed as a solution for large-scale data labelling. A number of GWAPs were therefore developed for NLP, including *Jeux de Mots* (Lafourcade, 2007; Joubert et al., 2018), *Phrase Detectives* (Chamberlain et al., 2008; Poesio et al., 2013), *OntoGalaxy* (Krause et al., 2010), the *Wordrobe* platform (Basile et al., 2012), *Dr Detective* (Dumitrache et al., 2013), *Zombilingo* (Fort et al., 2014), *TileAttack!* (Madge et al., 2017), *Wormingo* (Kicikoglu et al., 2019), *Name That Language!* (Cieri et al., 2021) or *High School Superhero* (Bonetti and Tonelli, 2021). GWAPs for coreference include *Phrase Detectives* (Chamberlain et al., 2008; Poesio et al., 2013), the *Pointers* game in *WordRobe* (Bos et al., 2017) and *Wormingo* (Kicikoglu et al., 2019), all deployed, and *PlayCoref* (Hladká et al., 2009), proposed but not tested.

However, whereas truly successful GWAPs such as FOLDIT have been developed in other areas of science (Cooper et al., 2010), even the most successful GWAPs for NLP only collected moderate amounts of data (Poesio et al., 2019; Joubert et al., 2018). In part, this is because the games used to actually collect NLP labels aren't very entertaining, leading to efforts to develop engaging designs such as (Jurgens and Navigli, 2014; Dziedzic and Włodarczyk, 2017; Madge et al., 2019).

An interesting solution to this issue was proposed although not fully developed for *Wordrobe* (Bos et al., 2017). This solution is a hybrid between automatic annotation and crowdsourcing: a combination of crowd and automatically computed judgments is aggregated to ensure that every item has at least one label. This solution wasn't properly tested in *Wordrobe*, which only collected very few judgments and for a small corpus; and anyway the approach followed could not be applied to anaphora/coreference, due to the lack of a suitable aggregation mechanism for anaphora/coreference. In this paper we present the first true test of the idea by proposing a method for aggregating crowd and automatic judgments inspired by this idea, but using an aggregation method for anaphora, and truly tested on a dataset containing a very large number of anaphoric judgments.

3 Phrase Detectives

The human judgments used in our corpus were collected using the *Phrase Detectives* Game-With-A-Purpose[4] (Chamberlain et al., 2008; Poesio et al., 2013; Chamberlain, 2016; Poesio et al., 2019), designed to collect multiple judgments about anaphoric reference.

Game design *Phrase Detectives* doesn't follow the design of some of the original von Ahn games (von Ahn and Dabbish, 2008), in that it is a one-person game, and not timed; both competition and timing were found to have orthogonal effects on the quality of the annotation (Chamberlain, 2016). Points are used as the main incentive, with weekly and monthly boards being displayed.

Players play two different games: one aiming at labelling new data, the other at validating judgments expressed by the other players. In the annotation game, *Name the Culprit*, the player provides an anaphoric judgment about a highlighted markable (the possible judgments according to the annotation scheme are discussed next). If different participants enter different interpretations for a markable then each interpretation is presented to other participants in the validation game, *Detectives Conference*, in which the participants have to agree or disagree with the interpretation.

Every item is annotated by at least 8 players (20 on average), and each distinct interpretation is validated by at least four players. Players get points for each label they produce, but especially when their interpretation is agreed upon by other players, thus rewarding accuracy. Initially, players play against gold data, and are periodically evaluated against the gold; when they achieve a sufficient level of accuracy, they start seeing incompletely annotated data. Extensive analyses of the data suggest that although there is a great number of noisy judgments, this simple training and validation method delivers extremely accurate aggregated labels (Poesio et al., 2013; Chamberlain, 2016; Poesio et al., 2019).

Annotation Scheme The annotation scheme used in *Phrase Detectives* is a simplified version of the ARRAU annotation scheme (Uryupina et al., 2020), covering all the main aspects of anaphoric annotation, including the distinction between referring and non-referring expressions (all noun phrases are annotated as either referring or non-referring, with two types of non-referring expressions being annotated: expletives and predicative NPs); the distinction between discourse-new and discourse-old referring expressions (Prince, 1992); and the annotation of all types of identity reference (including split antecedent plural anaphora). Only the most complex types of anaphoric reference (bridging references and discourse deixis) are not annotated. The main differences between the annotation scheme used in *Phrase Detectives* and

---

[4] `http://phrasedetectives.com/`

Type	Example	ONTONOTES	PRECO	ARRAU	*Phrase Detectives*
predicative NPs	[John] is a teacher [John, a teacher]	Pred	Coref	Pred	Pred
singletons		No	Yes	Yes	Yes
expletives	It's five o'clock	No	No	Yes	Yes
split antecedent plurals	[John] met [Mary] and they ...	No	No	Yes	Yes
generic mentions	[Parents] are usually busy. Parents should get involved	Only with pronouns	Yes	Yes	Yes
event anaphora	Sales [grew] 10%. This growth is exciting	Yes	No	Yes	No
ambiguity	Hook up [the engine] to [the boxcar] and send it to Avon	No	No	Explicit	Implicit

Table 1: Comparison between the annotation schemes in ONTONOTES, PRECO, ARRAU and *Phrase Detectives*.

		Docs	Tokens	Markables
TRAIN COMPLETE	Gutenberg	154	181142	48329 (29527)
	Wikipedia	359	244770	65050 (21803)
	Other	2	7294	2126 (1347)
	Subtotal	515	433206	115505 (52677)
TRAIN FULL	Gutenberg	194	372001	102354 (57387)
	Wikipedia	544	931752	258560 (92465)
	Other	2	7294	2128 (1347)
	Subtotal	740	1311047	363042 (151199)
DEV	Gutenberg	5	7536	2133 (1494)
	Wikipedia	35	15287	4423 (1669)
	Other	5	989	331 (126)
	Subtotal	45	23812	6887 (3289)
TEST	Gutenberg	7	20646	5925 (3332)
	Wikipedia	13	22998	7704 (3876)
	Subtotal	20	43644	13629 (7208)
All	Gutenberg	206	400183	110412 (62213)
	Wikipedia	592	970037	270687 (98010)
	Other	7	8283	2459 (1473)
	Total	805	1378503	383558 (161696)

Table 2: Summary of the current release. In parentheses the number of markables that are non-singletons.

those used in ARRAU, ONTONOTES, and PRECO are summarized in Table 1, modelled on a similar table in (Chen et al., 2018). In the *Phrase Detectives* corpus predication and coreference are clearly distinguished, as in ONTONOTES and ARRAU but unlike in PRECO. Singletons are considered markables. Expletives and split antecedent plurals are marked, unlike in either ONTONOTES or PRECO.

Possibly the most distinctive feature of the annotation scheme is that disagreements among annotators are preserved, encoding a form of implicit ambiguity as opposed to the explicit ambiguity annotated in ARRAU. The DEV and TEST subsets of the corpus (see next Section) have been annotated according to the full ARRAU scheme.

Preliminary player statistics At the time of writing (11th October, 2022), 61,391 players have registered on *Phrase Detectives*, of which more than 4,000 demonstrated sufficient linguistic understanding that they allowed to provide judgments on partially labelled data. So far, the players provided about 3.7M annotations and 1.7M validations, for a total of over 5.4M judgments.

Speed of annotation Over the course of the project, the games has been collecting an average of 385,000 judgments a year, i.e., slightly over 1,000 judgments per day, every day. While this is an impressive number of judgments, it only translates in an average of around 10,000 new completely annotated markables per year, or 20 new completely annotated documents, for an average of 30,000 extra words. (Progress was faster in the early years of the project, when all short documents were annotated; but as discussed in the next Section, the corpus also contains a number of fairly long texts – these are the ones still being annotated.) The project discussed in this paper was motivated by the simple calculation that at this speed, it would take us 40 years to completely annotate all the documents already in the corpus, and 300 years to completely annotated a corpus of 10M words.

4 Characteristics of the corpus

The *Phrase Detectives* 3.0 corpus includes all the 805 documents originally uploaded in the game. In this Section we highlight the main characteristics of the texts in this release, summarized in Table 2. For comparison, we include in the Appendix a short description of the previous release of the *Phrase Detectives* corpus, *Phrase Detectives* 2, released in 2019 (Poesio et al., 2019).

The new release The new release of the corpus, *Phrase Detectives* 3.0, is more than three times larger than the previous release of the *Phrase De-*

tectives corpus described in the previous section in terms of the number of tokens (1.4M) and markables (383K). (See 'All' row in Table 2.) This makes the *Phrase Detectives* 3.0 corpus comparable in the number of tokens to ONTONOTES, but double the size of ONTONOTES in terms of markables, partly due to the singletons and non-referring expressions being included. 72% of the documents were completely annotated by the players (580 out of 805 documents), and the near totality of mentions have at least one crowd annotation (99.4%).

Genres The corpus covers mainly two genres. The Gutenberg domain consists of fiction texts from the Gutenberg Project: in part children fiction (e.g., *Alice in Wonderland*, Grimm brothers stories), in part classics (e.g., Sherlock Holmes stories). At 400K tokens, it is twice the size of the largest existing fiction corpus (Bamman et al., 2020). The Wikipedia domain consists of primarily the 'Wikipedia Unusual' documents. This subset is 1M tokens in size, substantially larger than WikiCoref (60K tokens) (Ghaddar and Langlais, 2016).

Organization The corpus is split into train, development, and test subsets, where the development and test sets are annotated by human experts (see below) and the training set is aggregated using the MPA anaphoric annotation model (Paun et al., 2018b) as described in Section 5. But crucially, two versions of the training set exists.

TRAIN COMPLETE is like the training sets released in previous versions of the corpus, in that it consists of documents that were completely annotated by the players: i.e., all markables in the documents have more than 8 judgments, and all interpretations have more than 4 validations.

The second training set, TRAIN FULL, additionally includes documents that have not yet been 'completely' annotated by the players. These documents are considerably longer, and as a result it is harder to have them completely annotated. So a state-of-the-art coreference model for this annotation scheme (Yu et al., 2020) was used in the resolve-and-aggregate setting discussed in Section 5 to augment the existing annotations by ensuring that every markable had at least one label, which would then be aggregated with the others. TRAIN FULL is three times larger than TRAIN COMPLETE, both in the number of tokens and of markables.

A New Gold The test set from the previous release of the corpus, consisting of 45 documents, is now available as DEV. DEV was fully revised by human experts for this release to correct previous labelling mistakes, and has now been annotated according to the full ARRAU guidelines, including ambiguity annotation, bridging references, and discourse deixis. In addition, a brand new TEST set of 20 documents was also created, balanced between the two domains, double in size compared to the old test set, and also annotated according to the full ARRAU guidelines by the annotators that have been preparing the ARRAU 3 release.

Domain specific training With the new release, the corpus is now large enough to be used separately for domain-specific research. We demonstrate in Section 6 that models trained on the domain-specific portion of the training set achieve comparable results to those trained on TRAIN FULL. The results indicate that the domain-specific training data can be sufficient to be used separately for dedicated research in target domains.

Long and short documents An important characteristic of the corpus is that it was designed to contain both short documents (< 2K tokens) and long ones. 34.5% of the documents are longer than 2K tokens, and the longest document reaches 14K tokens. (In contrast, in ONTONOTES only 0.4% of the documents have more than 2K tokens.) This makes our corpus a suitable resource for research on long-distance anaphora and on long document training. To this end, we use our dataset to replicate the experiments by Beltagy et al. (2020) comparing the LONGFORMER model with the ROBERTA model. In the original paper, which used the ONTONOTES corpus, no obvious differences were found between the two models, partly due to the lack of long documents. We discuss these experiments in Section 6.5. The only other corpus that we are aware of with a large portion of long documents is the CRAFT corpus (Cohen et al., 2017), which is however focused on biomedical texts.

5 Resolve-and-Aggregate

The challenge To create a reliable corpus using crowdsourcing, multiple judgments are required to ensure a good coverage of correct answers, together with sufficient evidence to enable an accurate aggregation method (Paun et al., 2018a, 2022) to distill the correct answers from the noisy ones. The problem of collecting such large number of judgments is even more serious for long documents. Annotating all anaphoric relations in long documents is challenging, partly due to the amount of time needed to complete the task, but also because the great number of entities makes it difficult for annotators to keep track of all the coreference chains. And indeed, the short documents in our corpus were completed much faster than the longer documents: the average length of the incomplete documents is 4K tokens, whereas for the complete documents is 850 tokens. Thus in our corpus the rate at which judgments are collected from players, while substantial (over 1,000 judgments per day) is not sufficient to extract reliable labels in a reasonable amount of time, as discussed in Section 3.

Possible solutions Clearly, part of the solution is to develop more engaging games, thus able to attract more players and keep them playing for longer (von Ahn and Dabbish, 2008; Jurgens and Navigli, 2014; Madge et al., 2019; Kicikoglu et al., 2019). A second ingredient is to use active learning-like approaches to minimize the number of labels required to complete the annotation (Laws et al., 2012; Li et al., 2020; Yuan et al., 2022). A number of proposals have been made in these two directions, and we are carrying out research in these areas as well (Madge et al., 2022). In this work however we investigate an approach that to our knowledge has been much less studied: combining crowdsourcing with automatic labelling. Specifically, we propose a new resolve-and-aggregate method that iteratively makes use of a coreference resolver to enhance the collected annotations. The approach is inspired, apart from *Wordrobe* (Bos et al., 2017), by previous work on Bayesian combination of classifiers (Kim and Ghahramani, 2012) which allows for aggregating predictions from classifiers and humans together with the help of a probabilistic annotation model. Both the iterative use of the coreference resolver and the application domain of the annotation model are however novel to this paper.

The coreference resolver As a coreference resolver, we use the system by Yu et al. (2020) which, to the best of our knowledge, is the only modern coreference resolver that also predicts singletons and non-referring expressions, both of which need to be annotated in our corpus. The system is an extension of (Lee et al., 2017, 2018), replacing their mention-ranking algorithm with a cluster-ranking algorithm to build the entity clusters incrementally. The system uses BERT (Devlin et al., 2019) for pretrained contextual embeddings instead of the Elmo embeddings (Peters et al., 2018) used in (Lee et al., 2018).

Aggregation Standard aggregation methods for classification labels such as the (Dawid and Skene, 1979) model are not appropriate for coreference labels, whose class space is not fixed but depends on the document mentions. However, an aggregation model for coreference judgments is now available, the mention-pair annotation model (MPA) (Paun et al., 2018b). We used MPA to aggregate judgments by players and by the coreference resolver. MPA can capture the accuracy and bias of the players, and of the coreference resolver, respectively, and adjust the aggregated labels accordingly.

Resolve-and-aggregate resolve-and-aggregate is an iterative procedure which relies on the MPA aggregation model to label the corpus, which is in turn used to retrain the coreference resolver to get better system predictions. More specifically, in the first step of the procedure we aggregate the players' annotations from the complete documents and build an initial training set, TRAIN COMPLETE. Then, we train the coreference resolver on this set, but in a gold mention setting to mimic the players who focus only on the resolution task. Having trained the system, we then use it to get predictions for the entire dataset. The resolver can be seen as a player who played all the documents in the corpus. Next, all the players' annotations and the system's predictions are aggregated using MPA, and an initial version of the entire corpus, TRAIN FULL, is built as a result. This procedure is repeated, taking TRAIN FULL as input and creating a new version every time. With each new version, the MPA-aggregated labels get refined, leading in turn to better predictions from the coreference resolver. The procedure is repeated until the performance of the resolver plateaus. The final version of the corpus contains the MPA-aggregated labels of the players' annotations and the system's best predictions. We show in the next Section that this approach results in substantial improvements in the quality of the labels produced by the coreference resolver, which translate to more accurate labels for the items not fully annotated by the players.

6 Resolving-and-Aggregating results

Experiment Setting For our experiments, we report the CoNLL F1 scores as defined in (Pradhan

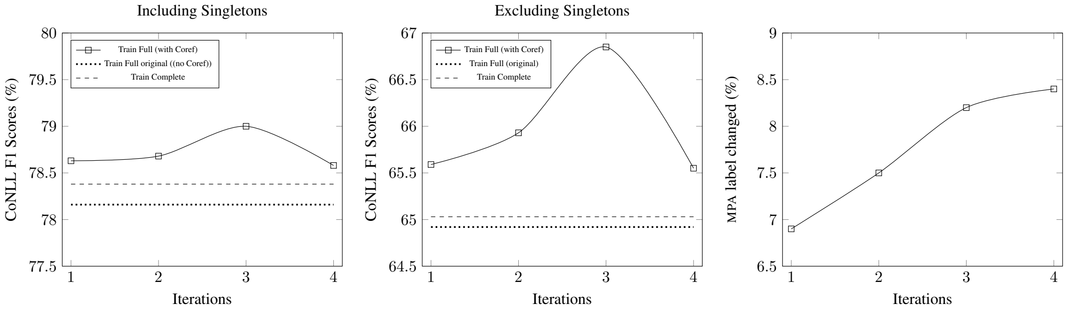


Figure 1: Left and Middle: The CoNLL scores for Yu et al. (2020) trained on different training sets and tested on the Dev set in gold mention setting. Right: The percentage of MPA labels changed by using the additional judgments from the Yu et al. (2020) system in different iterations.

et al., 2012) in both singleton included and excluded settings, as well as non-referring F1 scores for non-referring expressions. We use the Universal Anaphora (UA) Scorer (Yu et al., 2022b) that reports all the necessary scores.

We trained the Yu et al. (2020) system using most of its default settings. The only exception is that we always use the full context of the documents for training instead of choosing a random 1K tokens as done in Yu et al. (2020). The default setting gives priority to the short documents as for each epoch, the full context of the short documents is always used, whereas for long documents only part of the documents is used.

We establish three baselines, all using the same system Yu et al. (2020) with the same settings, but trained with different training sets. The first baseline is trained on the Previous release. The second baseline is trained on Train Complete (complete documents aggregated by MPA without resolve-and-aggregate). The third baseline is trained on Train Full aggregated by MPA but without annotations from the coreference resolver.

6.1 Parameter Tuning

We first trained the system using the gold mention settings to improve the quality of the corpus. We used the baseline trained on Train Complete to annotate the full corpus, then assigned labels to all the mentions by aggregating player and system annotations using MPA. We then trained a new model by using the full corpus (Train Full (with Coref)) and doing resolve-and-aggregate between the system and MPA in iterations until the system performance stopped improving.

	CoNLL Avg. F1		
Train data	Sing. (inc)	Sing. (exc)	NR F1
Previous release	65.5	53.6	36.8
Train Complete	66.1	54.7	39.4
Train Full(original)	64.9	52.9	35.5
Train Full(with Coref)	**66.8**	**56.1**	**40.1**
Joshi et al. (2020)	-	60.2	-

Table 3: The CoNLL and non-referring scores for (Yu et al., 2020) trained on different training sets and tested on the Test set in predicted mention setting.

The first key result is that, the system trained with Train Full (with Coref) always beats the baseline trained on the Train Complete (see Figure 1). The improvements on the singletons excluded setting are larger than those in the singletons included setting; this makes sense as all the models use gold mentions, hence the performance with singletons is inflated by the gold mentions. The system achieved the best performance on the third iteration with CoNLL F1 scores of 79% and 66.9% for singletons included and excluded settings respectively. This is 0.6% and 1.9% higher than the Train Complete baseline.

What is especially interesting is that the improvement is not just a matter of Train Full being larger than Train Complete: running the coreference resolver helps substantially. The system trained on Train Full original (i.e., without any automatic labels) is slightly worse than the Train Complete baseline, despite using the additional training data. One explanation would be that MPA's performance is affected by the lower number of judgments collected in the incomplete documents: the correct answer might not appear in the players' annotations, or the players producing the annotations might not be considered sufficiently reliable.

To quantify the contribution of the automatic coreference resolver, we calculate the percentage of MPA labels flipped due to the additional system annotations. We compare the labels of the Train

FULL (with Coref) in different iterations with the TRAIN FULL (original) labels. We find that in the first iteration, 7% of the MPA labels (26K) were changed (see Figure 1). The percentages increased sharply until iteration 3 to 8.2% (31K) but slowed down for iteration 4. This might explain why performance starts dropping in the 4th iteration.

MPA works very well when the number of judgments is high, but performance might be affected when there are not enough annotations, e.g. for the incomplete documents. We suspected MPA might benefit more from system annotations when the document is incomplete. To assess our hypothesis, we took a closer look at MPA labels from our best iteration. We split the documents into two classes, complete and incomplete, according to our complete criterion (i.e., a document is considered complete when every markable has been annotated by at least 8 players, and each distinct interpretation has been validated by at least 4 players) and calculate a separate score for each class. We find that for the complete document only 3.3% of the MPA labels are changed as a result of the additional system annotations; in contrast, 10.8% of MPA labels are changed in the incomplete documents.

To assess the quality of these label changes, we checked the different MPA labels between iteration 3 and the original on the DEV set. Since all documents from the DEV set are complete documents, out of 7K mentions, only 201 have a different label. The TRAIN FULL (original) gets 70 of the labels correct with an accuracy of 34.8%, whereas after the 3rd iteration of resolve-and-aggregate, the number increased to 125 (62.2% accuracy). Although the sample is not large, it still gives a clear picture that even for complete documents the system annotations can improve the quality of the corpus.

6.2 Evaluation on the Test set

After finding the best setting as discussed in the previous Section, we evaluated the impact of resolve-and-aggregate on the TEST set in the more realistic predicted mention setting. As shown in Table 3, our best model trained on the TRAIN FULL aggregated by MPA with additional coreference annotation by the Yu et al. (2020) system (TRAIN FULL(with Coref)) beats all the baselines in both singletons included and excluded settings. Of the baselines trained on the complete documents only, the TRAIN COMPLETE baseline works better than the PREVIOUS RELEASE baseline, most likely because the training set is larger while the quality of the annotation remains the same. But again, when training with the additional incomplete documents (TRAIN FULL (original without Coref)), the performance dropped substantially by 1%-2% when compared with the TRAIN COMPLETE baseline. This highlights again the importance of combining automatic and crowd annotations via resolve-and-aggregate: the model trained on this corpus pushes up the TRAIN FULL (original) baseline by up to 3.2%. The story is the same for the models' performance on non-referring expressions (Table 3): again, the model trained on TRAIN FULL (with Coref) is top of the list.

Finally, we report the result by the Joshi et al. (2020) system on our corpus to give insight into the complexity of our corpus when compared with ONTONOTES. The system was trained on the same TRAIN FULL (with Coref) corpus. Since the Joshi et al. (2020) system only output the non-singleton clusters, we report only the CoNLL F1 score in a singleton excluded setting. As expected the system has a better CoNLL F1 score when compared with our baselines, since SpanBERT has been shown to be more effective than BERT on coreference. The Joshi et al. (2020) result on our corpus is, however, 20% lower than on ONTONOTES (79.6%), which indicates that our corpus is more complex than ONTONOTES. We hypothesize this is partially due to the longer documents and more diverse domains included in our release.

6.3 Annotation speed-up

The results in the previous Sections show that using automatic annotations turns the incomplete documents into documents whose quality is enough to result in improved performance when training a coreference resolver, speeding up annotation. In this section, we try to estimate the amount of time potentially saved by the proposed method. For the complete documents, we have on average 20 judgements (annotations and validations) per markable, which seems sufficient to ensure the quality of the corpus, if not perhaps necessary. For incomplete documents, the average number of judgements is currently 7.7. If we do need 20 judgements to achieve the same quality as the complete documents, we still need to collect on average 12.3 more judgements for every markable. Multiplied by the number of markables in the incomplete documents (250K), this means we would need 3M more judge-

Train data	CoNLL Avg. F1 Sing. (inc)	 Sing. (exc)	 NR F1
	Gutenberg		
DOMAIN ONLY	70.4	61.8	43.8
TRAIN FULL (with Coref)	**71.5**	**62.1**	**44.9**
	Wikipedia		
DOMAIN ONLY	61.9	**50.9**	**36.1**
TRAIN FULL (with Coref)	**62.3**	50.6	36.0

Table 4: The CoNLL and non-referring scores for the system trained on different training sets and tested on the TEST set of different domains using predicted mention.

Model	Short Doc	Long Doc	All Doc
LONGFORMER	**61.0**	**67.2**	**64.7**
ROBERTA	60.1	65.2	63.1

Table 5: The CoNLL scores (exclude singletons) for LONGFORMER and ROBERTA trained on TRAIN FULL and tested on the TEST set using gold mentions.

ments to complete all documents in the game. In the last five years, we have been averaging 334K judgements per year, which means if we proceed at the current speed, we need another 9 years before we can release this corpus. In other words, the resolve-and-aggregate method significantly speeds up the annotation process.

6.4 Domain-specific Training

Thanks to the resolve-and-aggregate method, this new release gives us datasets of a reasonable size for both the Gutenberg (fiction) and Wikipedia domains. We evaluated system performance on the domain-specific portion - e.g., for Fiction we trained our model on the Gutenberg section of TRAIN FULL (with Coref) and tested it on the Gutenberg section of the TEST. We then compared the performance of these domain-specific models with that of the best system trained on the entire corpus. As shown in Table 4, the DOMAIN ONLY systems trained on the domain-specific subsections of the corpus achieve scores close to the system trained on the full corpus. This suggests each domain-specific part of the corpus is sufficiently large to be used for domain-specific research.

6.5 Long and short documents

As stated earlier, one of the emerging challenges for research on anaphora (and NLP in general) are longer documents (>2K tokens). Our corpus is unusual in that it includes a large number of documents more than 2K in length, with the longest document containing 14K tokens. TEST also balances short (55%) and long (45%) documents.

To test that the corpus can support research on anaphora in long documents, we used it to replicate the comparison in (Beltagy et al., 2020) between their new model designed specifically for longer documents, the LONGFORMER, with ROBERTA (Liu et al., 2019). In that paper, the LONGFORMER is compared with ROBERTA on the ONTONOTES corpus, without however finding a clear difference between the two systems. We suspected this might be because ONTONOTES does not contain enough long documents to observe improvements. We replicated the experiments by Beltagy et al. with our corpus, and report the CoNLL F1 score on full TEST as well as separate scores for long/short documents. (Since neither system predicts singletons and non-referring expressions, we report the CoNLL F1 scores in the singleton excluded setting.) We evaluated the systems with the gold mentions so that the system's performance will not be affected by mention detection.

Table 5 shows the results for both systems on different test set. The LONGFORMER works better on all test sets, but with a much larger gain over ROBERTA on long documents: the improvement over ROBERTA is 0.9% and 2% on short and long documents respectively. This finding confirms that long documents benefit more from the LONGFORMER architecture, while also showing that our corpus can be used to differentiate systems designed to perform on long documents.

7 Conclusions

This research makes two main contributions. First of all, we proposed an iterative method for speeding up anaphoric annotation via GWAPs by combining crowdsourced data with labels produced by an automatic coreference resolver, and aggregating the labels using a probabilistic annotation method; and showed that the resulting extension leads to quantifiable improvements in model performance. The method can be easily extended to other types of annotation. Second, we introduced a new corpus for anaphoric reference which, thanks to the use of resolve-and-aggregate, is of a comparable size to ONTONOTES in terms of tokens, but twice the size in terms of markables; it contains two substantial datasets for genres not covered in ONTONOTES; and it includes both short and long documents.

8 Limitations

The main limitation of this work is that the new release is still only twice the size of ONTONOTES in terms of markables. In ongoing work, we are developing a new platform to label a corpus twenty times the size of the current release. The new platform[5] combines more engaging games with active-learning like methods for allocating work to players more efficiently and according to their linguistic understanding (Madge et al., 2022). We hope that the new platform, in combination with the methods proposed here, will allow us to label the new and larger dataset much more quickly.

A second limitation of the new release is that the markables in the corpus were automatically extracted; thus, the quality of the mentions is lower than in corpora in which they were hand-identified. The approach followed in these years has been to ask our players to signal issues; as a result, tens of thousands of markables were hand-corrected. However, this approach doesn't really lend itself to scaling up. Thus, in our new platforms we are following a different strategy: asking our players to do the corrections themselves, by including also games to check other levels of linguistic interpretation.

A third limitation, in particular in comparison with ONTONOTES, is that this release of the corpus only contains English documents, although a small amount of Italian documents was uploaded in the game.

Acknowledgements

This research was supported in part by the ANAWIKI project, funded by EPSRC (EP/F00575X/1);[6] in part by the DALI project, funded by the European Research Council (ERC), Grant agreement ID: 695662;[7] in part by the ARCIDUCA project, funded by EPSRC (EP/W001632/1).

References

Abdulrahman Aloraini and Massimo Poesio. 2021. Data augmentation methods for anaphoric zero pronouns. In *Proc. of the CRAC Workshop*.

David Bamman, Olivia Lewke, and Anya Mansoor. 2020. An annotated dataset of coreference in English literature. In *Proc. of LREC*. European Language Resources Association (ELRA), Association for Computational Linguistics (ACL).

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *Proc. of LREC*, pages 3196–3200, Istanbul, Turkey.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Federico Bonetti and Sara Tonelli. 2021. Challenges in designing games with a purpose for abusive language annotation. In *Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing*.

Johan Bos, Valerio Basile, Kilian Evang, Noortje J. Venhuizen, and Johannes Bjerva. 2017. The groningen meaning bank. In N. Ide and J. Pustejovsky, editors, *The Handbook of Linguistic Annotation*, chapter 18, pages 463–496. Springer.

Jon Chamberlain. 2016. *Harnessing Collective Intelligence on Social Networks*. Ph.D. thesis, University of Essex, School of Computer Science and Electronic Engineering.

Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. 2008. Phrase Detectives: A web-based collaborative annotation game. In *Proceedings of I-Semantics 2008*.

Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. 2016. Phrase detectives corpus 1.0: Crowdsourced anaphoric coreference. In *Proceedings of LREC*, Portoroz, Slovenia.

Hong Chen, Zhenhua Fan, Hao Lu, Alan Yuille, and Shu Rong. 2018. PreCo: A large-scale dataset in preschool vocabulary for coreference resolution. In *Proceedings of EMNLP*, pages 172–181, Brussels, Belgium.

Christopher Cieri, James Fiumara, and Jonathan Wright. 2021. Using games to augment corpora for language recognition and confusability. In *Proc. of Interspeech: 22nd Annual Conference of the International Speech Communication*.

K Bretonnel Cohen, Arrick Lanfranchi, Miji Joo-young Choi, Michael Bada, William A Baumgartner, Natalya Panteleyeva, Karin Verspoor, Martha Palmer, and Lawrence E Hunter. 2017. Coreference annotation and resolution in the colorado richly annotated full text (craft) corpus of biomedical journal articles. *BMC bioinformatics*, 18(1):1–14.

Seth Cooper, Firsas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popovic, and the Foldit Players. 2010. Predicting protein structures with a multiplayer online game. *Nature*, 466:756–760.

---

[5] `https://lingotowns.com/`

[6] `https://anawiki.essex.ac.uk/`

[7] `http://www.dali-ambiguity.org`

Alexander P. Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28.

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

Anca Dumitrache, Lora Aroyo, Chris Welty, Robert-Jan Sips, and Anthony Levas. 2013. Dr. detective: combining gamification techniques and crowdsourcing to create a gold standard in medical text. In *Proc. of CrowdSem*.

Dagmara Dziedzic and Wojciech Włodarczyk. 2017. Making nlp games with a purpose fun to play using free to play mechanics: RoboCorp case study. In *Proc. of the Games4NLP Symposium*.

Ali Emami, Paul Trichelair, Adam Trischler, Kaheer Suleman, Hannes Schulz, and Jackie Chi Kit Cheung. 2019. The knowref coreference corpus: Removing gender and number cues for difficult pronominal anaphora resolution. In *Proc. of the ACL*.

Eraldo R. Fernandes, Cícero N. dos Santos, and Ruy L. Milidiú. 2014. Latent trees for coreference resolution. *Computational Linguistics*, 40(4):801–835.

Karen Fort, Bruno Guillaume, and H. Chastant. 2014. Creating Zombilingo, a game with a purpose for dependency syntax annotation. In *Proceedings of the 1st International Workshop on Gamification for Information Retrieval (GamifIR'14)*, pages 2–6. ACM.

Luke Gessler, Siyao Peng, Yang Liu, Yilun Zhu, Shabnam Behzad, and Amir Zeldes. 2020. AMALGUM – a free, balanced, multilayer english web corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, page 5267–5275. European Language Resources Association.

Abbas Ghaddar and Phillippe Langlais. 2016. WikiCoref: An English coreference-annotated corpus of Wikipedia articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 136–142, Portorož, Slovenia. European Language Resources Association (ELRA).

Iris Hendrickx, Gosse Bouma, Frederik Coppens, Walter Daelemans, Veronique Hoste, Geert Kloosterman, Anne-Marie Mineur, Joeri Van Der Vloet, and Jean-Luc Verschelde. 2008. A coreference corpus and resolution system for dutch. In *Proc. LREC*.

Erhard W. Hinrichs, Sandra Kübler, and Karin Naumann. 2005. A unified representation for morphological, syntactic, semantic and referential annotations. In *Proc. of the ACL Workshop on Frontiers in Corpus Annotation II: Pie in the Sky*, Ann Arbor, Michigan.

Barbora Hladká, Jiří Mírovský, and Pavel Schlesinger. 2009. Designing a language game for collecting coreference annotation. In *Proceedings of the Third Linguistic Annotation Workshop (LAW III)*, pages 52–55, Singapore.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Alain Joubert, Mathieu Lafourcade, and Nathalie Le Brun. 2018. The jeuxdemots project is 10 years old: What we have learned. In *Proceedings of the 2018 LREC Workshop on Games and Gamification for Natural Language Processing (Games4NLP)*, pages 22–26.

David Jurgens and Roberto Navigli. 2014. It's all fun and games until someone annotates: Video games with a purpose for linguistic annotation. *Transactions of the ACL*.

Doruk Kicikoglu, Richard Bartle, Jon Chamberlain, and Massimo Poesio. 2019. Wormingo: a'true gamification' approach to anaphoric annotation. In *FDG'19: Proc. of the GAMNLP Workshop at the 14th International Conference on the Foundations of Digital Games*, pages 1–7, San Luis Obispo.

Hyun-Chul Kim and Zoubin Ghahramani. 2012. Bayesian classifier combination. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 619–627, La Palma, Canary Islands. PMLR.

Markus Krause, Aneta Takhtamysheva, Marion Wittstock, and Rainer Malaka. 2010. Frontiers of a paradigm: exploring human computation with digital games. In *Proceedings of HCOMP - the ACM SIGKDD Workshop on Human Computation*, pages 22–25.

Mathieu Lafourcade. 2007. Making people play for lexical acquisition with the JeuxDeMots prototype. In *Proc. of SNLP*.

Mathieu Lafourcade, Alain Joubert, and Nathalie Le Brun. 2015. *Games with a Purpose (GWAPs)*. Wiley.

Fred Landragin. 2016. Description, modéelisation et détection automatique des chaînes de rééférence (democrat). *Bulletin de l'Association Française pour l'Intelligence Artificielle*, 92:11–15.

Florian Laws, Florian Heimerl, and Hinrich Schütze. 2012. Active learning for coreference resolution. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 508–512. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of EMNLP*.

Kenton Lee, Luheng He, and Luke S. Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of ACL*.

Belinda Z. Li, Gabriel Stanovsky, and Luke Zettlemoyer. 2020. Active learning for coreference resolution using discrete annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 8320–8331. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chris Madge, Richard Bartle, Jon Chamberlain, Udo Kruschwitz, and Massimo Poesio. 2019. Incremental game mechanics applied to text annotation. In *CHI PLAY 2019 - Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, pages 545–558, Barcelona.

Chris Madge, Jussi Brightmore, Doruk Kicikoglu, Fatima Althani, Richard Bartle, Jon Chamberlain, Udo Kruschwitz, and Massimo Poesio. 2022. Lingotowns: A virtual world for natural language annotation and language learning. In *CHI PLAY 2022 - Proceedings of the Annual Symposium on Computer-Human Interaction in Play, Demo session*, Bremen.

Chris Madge, Jon Chamberlain, Udo Kruschwitz, and Massimo Poesio. 2017. Experiment-driven development of a gwap for marking segments in text. In *Proceedings of CHI PLAY*, Amsterdam.

Judith Muzerelle, Anaïs Lefeuvre, Emmanuel Schang, Jean-Yves Antoine, Aurore Pelletier, Denis Maurel, Iris Eshkol, and Jeanne Villaneau. 2014. Ancor_centre, a large free spoken french coreference corpus. In *Proc. of LREC*.

Anna Nedoluzhko, Michal Novák, Silvie Cinková, Marie Mikulová, and Jiří Mírovský. 2016. Coreference in prague czech-english dependency treebank. In *Proc. of LREC*, Portoroz. ELRA.

Silviu Paun, Ron Artstein, and Massimo Poesio. 2022. Statistical methods for annotation analysis. *Synthesis Lectures on Human Language Technologies*, 15(1):1–217.

Silviu Paun, Bob Carpenter, Jon Chamberlain, Dirk Hovy, Udo Kruschwitz, and Massimo Poesio. 2018a. Comparing bayesian models of annotation. *Transactions of the Association for Computational Linguistics*.

Silviu Paun, Jon Chamberlain, Udo Kruschwitz, Juntao Yu, and Massimo Poesio. 2018b. A probabilistic annotation model for crowdsourcing coreference. In *Proceedings of EMNLP*, pages 1926–1937, Brussels, Belgium. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL*.

Massimo Poesio, Jon Chamberlain, and Udo Kruschwitz. 2017. Crowdsourcing. In N. Ide and J. Pustejovsky, editors, *The Handbook of Annotation*, pages 277–295. Springer.

Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Silviu Paun, Alexandra Uma, and Juntao Yu. 2019. A crowdsourced corpus of multiple judgments and disagreement on anaphoric interpretation. In *Proc. of NAACL*, page 1778–1789, Minneapolis. Association for Computational Linguistics (ACL).

Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Livio Robaldo, and Luca Ducceschi. 2013. Phrase Detectives: Utilizing collective intelligence for internet-scale language resource creation. *ACM Transactions on Interactive Intelligent Systems*, 3(1):1–44.

Massimo Poesio, Sameer Pradhan, Marta Recasens, Kepa Rodriguez, and Yannick Versley. 2016. Annotated corpora and annotation tools. In M. Poesio, R. Stuckardt, and Y. Versley, editors, *Anaphora Resolution: Algorithms, Resources and Applications*, chapter 4. Springer.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Sixteenth Conference on Computational Natural Language Learning (CoNLL 2012)*, Jeju, Korea.

Ellen F. Prince. 1992. The ZPG letter: subjects, definiteness, and information status. In S. Thompson and W. Mann, editors, *Discourse description: diverse analyses of a fund-raising text*, pages 295–325. John Benjamins.

Marta Recasens and M. Antònia Martí. 2010. AnCora-CO: Coreferentially annotated corpora for Spanish and Catalan. *Language Resources and Evaluation*, 44(4):315–345.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast - but is it good? Evaluating non-expert annotations for natural language tasks. In *Proceedings of EMNLP*, pages 254–263.

Olga Uryupina, Ron Artstein, Antonella Bristot, Federica Cavicchio, Francesca Delogu, Kepa J. Rodriguez, and Massimo Poesio. 2020. Annotating a broad range of anaphoric phenomena, in a variety of genres: the ARRAU corpus. *Journal of Natural Language Engineering*.

Luis von Ahn. 2006. Games with a purpose. *Computer*, 39(6):92–94.

Luis von Ahn and Laura Dabbish. 2008. General techniques for designing games with a purpose. *Communications of the ACM*, pages 58–67.

Sam J. Wiseman, Alexander M. Rush, Stuart M. Shieber, and Jason Weston. 2015. Learning anaphoricity and antecedent ranking features for coreference resolution. In *Proc. of the ACL*, Bejing.

Patrick Xia and Benjamin Van Durme. 2021. Moving on from ontonotes: Coreference resolution model transfer. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, page 5241–5256. Association for Computational Linguistics.

Juntao Yu, Sopan Khosla, Ramesh Manuvinakurike, Lori Levin, Vincent Ng, Massimo Poesio, Michael Strube, and Massimo Poesio. 2022a. The CODI/CRAC 2022 shared task on anaphora resolution, bridging and discourse deixis in dialogue. In *Proc. of CODI/CRAC Shared Task*.

Juntao Yu, Sopan Khosla, Nafise Sadat Moosavi, Silviu Paun, Sameer Pradhan, and Massimo Poesio. 2022b. The universal anaphora scorer. In *Proceedings of the Language Resources and Evaluation Conference*, pages 4873–4883, Marseille, France. European Language Resources Association.

Juntao Yu, Alexandra Uma, and Massimo Poesio. 2020. A cluster ranking model for full anaphora resolution. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 11–20, Marseille, France. European Language Resources Association.

Michelle Yuan, Patrick Xia, Chandler May, Benjamin Van Durme, and Jordan Boyd-Graber. 2022. Adapting coreference resolution models through active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, page 7533–7549. Association for Computational Linguistics.

Amir Zeldes. 2020. *Multilayer corpus studies*. Routledge.

Appendix

A The previous release of the corpus

Phrase Detectives 2 consisted of a total of 542 documents containing 408K tokens and 108K markables from two main genres: Wikipedia articles and fiction from the Gutenberg collection. This version of the corpus was divided in two subsets. The subset referred to to as $\text{PD}_{\text{silver}}$ consisted of 497 documents, for a total of 384K tokens and 101K markables, whose annotation was completed–i.e. 8 judgments per markable were collected, and 4 validations per interpretation–as of 12th of October 2018. In these documents, an aggregated ('silver') label obtained through MPA is also provided. 45 additional documents were also gold-annotated by two experts annotators. The subset of the corpus for which both gold and silver annotations are available was called PD_{gold}, as it is intended to be used as test set.[8] The gold subset consists of a total of 23K tokens and 6K markables. The contents of the *Phrase Detectives* 2 corpus are summarized in Table 6.

		Docs	Tokens	Markables
PD_{gold}	Gutenberg	5	7536	1947 (1392)
	Wikipedia	35	15287	3957 (1355)
	GNOME	5	989	274 (96)
	Subtotal	45	23812	6178 (2843)
$\text{PD}_{\text{silver}}$	Gutenberg	145	158739	41989 (26364)
	Wikipedia	350	218308	57678 (19444)
	Other	2	7294	2126 (1339)
	Subtotal	497	384341	101793 (47147)
All	Total	542	408153	107971 (49990)

Table 6: Summary of the contents of the 2019 release of the *Phrase Detectives* corpus. The numbers in parentheses indicate the total number of markables that are non-singletons.

B Detailed Evaluation Results

This appendix section includes the detailed evaluation results for this paper. More specifically, Table 7 and Table 8 show the detailed scores for our experiments on predicted mentions (discussed in Section 6.2); Table 9 and Table 10 show the detailed scores of coreference and non-referring expressions for the domain specific training experiments set out in Section 6.4. Table 11 shows the detailed scores for the long/short documents experiments discussed in Section 6.5.

Singletons	Train Data	MUC			BCUB			CEAFE			Avg. F1
		P	R	F1	P	R	F1	P	R	F1	
Included	PREVIOUS RELEASE	83.2	60.6	70.1	73.9	54.8	62.9	59.7	67.5	63.4	65.5
	TRAIN COMPLETE	83.1	62.1	71.1	74.7	54.5	63.0	62.4	66.2	64.3	66.1
	TRAIN FULL (original)	84.2	58.6	69.1	76.0	52.5	62.1	60.2	66.8	63.4	64.9
	TRAIN FULL (with Coref)	83.4	63.4	72.0	74.5	55.3	63.5	63.4	66.5	64.9	66.8
Excluded	PREVIOUS RELEASE	83.2	60.6	70.1	72.4	36.6	48.6	52.8	34.9	42.0	53.6
	TRAIN COMPLETE	83.1	62.1	71.1	71.6	37.6	49.3	53.6	36.8	43.6	54.7
	TRAIN FULL (original)	84.2	58.6	69.1	73.4	33.4	46.0	55.1	36.1	43.7	52.9
	TRAIN FULL (with Coref)	83.4	63.4	72.0	71.0	39.0	50.4	56.2	38.8	45.9	56.1
	SpanBERT-Large (Joshi et al.)	89.0	65.5	75.5	79.7	43.2	56.0	60.9	41.4	49.2	60.2
	SpanBERT-Base (Joshi et al.)	88.1	64.6	74.5	79.3	43.4	56.1	58.8	40.2	47.7	59.4

Table 7: The CoNLL scores for the Yu et al. (2020) and Joshi et al. (2020) systems trained on different training sets and tested on the TEST set in predicted mention setting.

---

[8] $\text{PD}_{\text{gold}}$is the dataset released in 2016 as *Phrase Detectives* corpus, Release 1 (Chamberlain et al., 2016).

Train data	P	R	F1
Previous release	73.8	24.6	36.8
Train Complete	71.1	27.3	39.4
Train Full(original)	75.1	23.3	35.5
Train Full(with Coref)	77.9	27.0	40.1

Table 8: Non-referring scores for Yu et al. (2020) system trained on different training sets and tested on the Test set in predicted mention setting.

Singletons	Train Data	MUC			BCUB			CEAFE			Avg. F1
		P	R	F1	P	R	F1	P	R	F1	
						Gutenberg					
Included	Domain Only	87.3	75.2	80.8	71.5	57.6	63.8	61.3	73.4	66.8	70.4
	Train Full (with Coref)	87.6	75.9	81.3	73.2	57.8	64.6	65.0	72.5	68.5	71.5
Excluded	Domain Only	87.3	75.2	80.8	70.2	42.0	52.5	56.3	48.5	52.1	61.8
	Train Full (with Coref)	87.6	75.9	81.3	69.9	42.7	53.0	56.3	48.2	51.9	62.1
						Wikipedia					
Included	Domain Only	75.4	52.0	61.6	72.8	54.0	62.0	61.2	62.8	62.0	61.9
	Train Full (with Coref)	78.1	51.3	61.9	75.5	53.3	62.5	62.4	62.7	62.5	62.3
Excluded	Domain Only	75.4	52.0	61.6	69.8	36.9	48.3	54.1	35.5	42.9	50.9
	Train Full (with Coref)	78.1	51.3	61.9	72.3	35.8	47.9	56.2	33.5	42.0	50.6

Table 9: The CoNLL scores for Yu et al. (2020) system trained on different training sets and tested on the Test set of different domains in predicted mention setting.

Train data	P	R	F1
	Gutenberg		
Domain Only	79.9	30.2	43.8
Train Full (with Coref)	84.0	30.6	44.9
	Wikipedia		
Domain Only	71.6	24.1	36.1
Train Full (with Coref)	72.4	24.0	36.0

Table 10: Non-referring scores for Yu et al. (2020) system trained on different training sets and tested on the Test set of different domains in predicted mention setting.

Settings	Model	MUC			BCUB			CEAFE			Avg. F1
		P	R	F1	P	R	F1	P	R	F1	
Short Doc	Longformer	96.2	61.5	75.0	88.4	45.9	60.4	74.8	34.7	47.4	61.0
	Roberta	96.3	60.8	74.5	89.3	45.1	59.9	71.1	33.9	45.9	60.1
Long Doc	Longformer	94.2	71.6	81.3	77.6	53.3	63.1	73.2	46.7	57.0	67.2
	Roberta	94.4	71.1	81.1	76.3	49.4	59.9	71.9	43.8	54.4	65.2
All Doc	Longformer	94.9	67.5	78.9	81.6	50.2	62.2	73.8	41.3	52.9	64.7
	Roberta	95.1	66.9	78.5	81.1	47.6	60.0	71.6	39.3	50.7	63.1

Table 11: The CoNLL scores for Longformer and Roberta systems trained on Train Full and tested on the Test set using gold mentions in a singleton excluded setting.

What Makes Sentences Semantically Related?
A Textual Relatedness Dataset and Empirical Study

Mohamed Abdalla
Institute for Better Health
& University of Toronto
msa@cs.toronto.edu

Krishnapriya Vishnubhotla
University of Toronto
vkpriya@cs.toronto.edu

Saif M. Mohammad
National Research Council Canada
saif.mohammad@nrc-cnrc.gc.ca

Abstract

The degree of semantic relatedness of two units of language has long been considered fundamental to understanding meaning. Additionally, automatically determining relatedness has many applications such as question answering and summarization. However, prior NLP work has largely focused on semantic similarity, a subset of relatedness, because of a lack of relatedness datasets. In this paper, we introduce a dataset for Semantic Textual Relatedness, *STR-2022*, that has 5,500 English sentence pairs manually annotated using a comparative annotation framework, resulting in fine-grained scores. We show that human intuition regarding relatedness of sentence pairs is highly reliable, with a repeat annotation correlation of 0.84. We use the dataset to explore questions on what makes sentences semantically related. We also show the utility of STR-2022 for evaluating automatic methods of sentence representation and for various downstream NLP tasks.

Our dataset, data statement, and annotation questionnaire can be found at: `https://doi.org/10.5281/zenodo.7599667`.

1 Introduction

The semantic relatedness of two units of language is the degree to which they are close in terms of their meaning (Mohammad, 2008; Mohammad and Hirst, 2012). The linguistic units can be words, phrases, sentences, etc. Though our intuition of semantic relatedness is dependent on many factors such as the context of assessment, age, and socio-economic status (Harispe et al., 2015), it is argued that a consensus can usually be reached for many pairs (Harispe et al., 2015). Consider the two sentence pairs in Table 1. Most speakers of English will agree that the sentences in the first pair are closer in meaning to one another than those in the second. When judging the semantic relatedness between two sentences, humans generally look for commonalities in meaning: whether they are on the same topic, express the same view, originate from the same time period, one elaborates on (or follows from) the other, etc.

Pair 1: a. *There was a lemon tree next to the house.*
b. *The boy enjoyed reading under the lemon tree.*

Pair 2: a. *There was a lemon tree next to the house.*
b. *The boy was an excellent football player.*

Table 1: Most people will agree that the sentences in pair 1 are more related than the sentences in pair 2.

The semantic relatedness of two units of language has long been considered fundamental to understanding meaning (Halliday and Hasan, 1976; Miller and Charles, 1991); given how difficult it has been to define meaning, a natural approach to get at the meaning of a unit is to determine how close it is to other units. Thus, unsurprisingly, automatically determining relatedness has many applications such as question answering, text generation, and summarization (more discussion in §7).

However, prior NLP work has focused on semantic similarity (a small subset of semantic relatedness), largely because of a dearth of datasets on relatedness. The few relatedness datasets that exist are only for word pairs (Rubenstein and Goodenough, 1965; Radinsky et al., 2011) or phrase pairs (Asaadi et al., 2019). Further, most existing datasets were annotated, one item at a time, using coarse rating labels such as integer values between 1 and 5 representing coarse degrees of closeness. It is well documented that such approaches suffer from inter- and intra-annotator inconsistency, scale region bias, and issues arising due to the fixed granularity (Presser and Schuman, 1996). Further, the notions of *related* and *unrelated* have fuzzy boundaries. Different people may have different intuitions of where such a boundary exists. Finally, for some tasks, it is more appropriate to train on a dataset of relatedness than similarity. (§2.1 discusses how relatedness and similarity are different.)

In this paper, we present the first manually annotated dataset of sentence–sentence semantic re-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 782–796
May 2-6, 2023 ©2023 Association for Computational Linguistics

latedness. It includes fine-grained scores of relatedness from 0 (least related) to 1 (most related) for 5,500 English sentence pairs. The sentences are taken from diverse sources and thus also have diverse sentence structures, varying amounts of lexical overlap, and varying formality.

The relatedness scores were obtained using a *comparative* annotation schema: two (or more) items are presented together and the annotator has to determine which is greater with respect to the metric of interest. Since annotators are making relative judgments, the limitations discussed earlier for rating scales are greatly mitigated. Importantly, such annotations do not rely on arbitrary boundaries between arbitrary categories such as "strongly related" and "somewhat related".

We use the relatedness dataset to explore:

1. To what extent do speakers of English intuitively agree on the relatedness of sentences? (§4)
2. What makes two sentences more related? (§5)
3. How well do existing approaches of sentence representation capture semantic relatedness (by placing related sentence pairs closer to each other in vector space)? (§6)
4. How can an improved annotation schema to capture relatedness benefit other NLP tasks? (§7)

We refer to our dataset as *STR-2022*, and the task of predicting relatedness between sentences as the *Semantic Textual Relatedness (STR)* task. Data, data statement, and annotation questionnaire are made available[1].

2 Related Work and Our Approach to Annotating for Semantic Relatedness

The three subsections below discuss key ideas from past work on annotating relatedness and similarity, existing datasets, and comparative annotation, respectively. Notably, each of these subsections also discusses how relevant past work has influenced our approach to data annotation.

2.1 Annotating Relatedness and Similarity

Semantic relatedness and semantic similarity are two ways to explore closeness of meaning. Two terms are considered semantically similar if there is a synonymy, hyponymy, or troponymy relation between them (examples include *doctor–physician* and *mammal–elephant*). Two terms are considered to be semantically related if there is any lexical semantic relation at all between them. Thus, all similar pairs are also related, but not all related pairs are similar. For example, *surgeon–scalpel* and *tree–shade* are related, but not similar.

Analogous to term pairs, two sentences are considered semantically similar when they have a paraphrasal or entailment relation. Determining such an equivalence of meaning is useful in NLP tasks such as text summarization and plagiarism detection. Semantic Relatedness, however, accounts for all of the commonalities that can exist between two sentences (Halliday and Hasan, 1976; Morris and Hirst, 1991). For example, the sentences in Table 1 Pair 1 are highly related, but they are not paraphrases or entailing. This expands the scope of the measure to include aspects such as the relatedness between their topics, their styles, etc.

However, because semantic relatedness involves innumerable classical and ad-hoc semantic relationships, it is markedly more complex than semantic similarity, and there are no widely agreed upon linguistic theories or guidelines for judging relatedness. This presents a challenge for gathering annotations; one can either: (i) construct their own codified instructions on how to judge semantic relatedness under various scenarios (e.g., overlapping sentence structure, relatedness of topic, etc.), at the risk of artificially over-simplifying the task or (ii) abstain from explicitly and comprehensively defining relatedness for numerous types of sentence pairs, relying instead on a simple description of relatedness, a few examples, and framing the task in relative terms.[2] In this work, we chose the latter. This allows us to: (i) determine the extent to which human intuition of relatedness is reliable and (ii) use the resulting dataset to empirically determine what makes sentences semantically related.

2.2 Existing Relatedness and Similarity Data

Existing datasets created for sentence pair similarity (e.g., STS (Agirre et al., 2012, 2013, 2014, 2015, 2016), MRPC (Dolan and Brockett, 2005), and LiSent (Li et al., 2006)) ask annotators to choose among coarse similarity labels. This leads to information loss and makes annotation difficult because distinctions between categories are often not clear; for example, the STS 2012–2016 questionnaires

---

[1] `doi.org/10.5281/zenodo.7599667` or `https://github.com/Priya22/semantic-textual-relatedness` or `https://huggingface.co/datasets/vkpriya/str-2022`

[2] Recall that for Table 1, we were able to judge relative relatedness without explicit instruction on how to judge relatedness.

ask annotators to make the distinction between *2: not equivalent but share some details* and *1: not equivalent, but are on the same topic*, which is often not straightforward. Further, despite claiming to determine semantic similarity, the descriptions of categories 1 and 2 incorporate aspects of semantic relatedness — an amalgamation muddying the waters with respect to the phenomenon being annotated. Such an amalgamation is also seen in the SICK (Marelli et al., 2014) dataset which combines a labeling scheme from STS with those about entailment and contradiction. These datasets have helped make progress in the field, but there is a need for relatedness datasets obtained strictly from relatedness judgments as opposed to a hybrid involving artificially created categories for similarity and entailment. For our annotations, we avoid fuzzy ill-defined categories, and rely instead on the intuitions of fluent English speakers to judge **relative rankings** of sentence pairs by relatedness.

2.3 Comparative Annotations

The simplest form of comparative annotations is paired comparisons (Thurstone, 1927; David, 1963). Annotators are presented with pairs of examples and asked to choose which item is greater with respect to the property of interest (relatedness, sentiment, etc.). The choices are then used to generate an ordinal ranking of items. Paired comparison avoids a number of biases, but it requires a large number of annotations (N^2, where $N =$ # items).

Best–Worst Scaling (BWS) is a comparative annotation schema that builds on pairwise comparisons and requires fewer labels (Louviere and Woodworth, 1991). Annotators are given n items at a time (for our work, $n = 4$ and an *item* is a pair of sentences). They are instructed to choose the best (i.e., most related) and worst (i.e., least related) item. Annotation for each 4-tuple provides us with five pairwise inequalities. For example if a is marked as most related and d as least related, then we know that $a > b, a > c, a > d, b > d$, and $c > d$. These inequalities can be used to calculate real-valued scores, and thus an ordinal ranking of items, using a simple counting mechanism (Orme, 2009; Flynn and Marley, 2014): the fraction of times an item was chosen as the best (most related) minus the fraction of times the item was chosen as the worst (least related). Given N items, reliable scores are obtainable from about $2N$ 4-tuples (Kiritchenko and Mohammad, 2016, 2017).

3 Creating STR-2022

Dataset creation included several steps: curating sentence pairs for annotation, designing the questionnaire, crowdsourcing annotations, and aggregating the annotations to obtain relatedness scores.

3.1 Data Sources

Like previous work on semantic similarity, we chose to construct our dataset by sampling sentences from many sources to capture a wide variety of text in terms of sentence structure, formality, and grammaticality. Pairs of sentences were created from the sampled sentences in a number of ways as described below. The sources are:

1. **Formality** (Rao and Tetreault, 2018): Pairs of sentences having the same meaning but differing in formality (one formal, one informal).
2. **Goodreads** (Wan and McAuley, 2018): Book reviews from the Goodreads website.
3. **ParaNMT** (Wieting and Gimpel, 2018): Paraphrases from a machine translation system.
4. **SNLI** (Bowman et al., 2015): Pairs of premises and hypotheses, created from image captions, for natural language inference.
5. **STS** (Cer et al., 2017): Pairs of sentences with semantic similarity scores. (Integer label responses, 0 to 5, from multiple annotators were averaged to obtain the similarity scores.)
6. **Stance** (Mohammad et al., 2016): Tweets labelled for both sentiment (*positive*, *negative*, *neutral*) and stance (*for*, *against*, *neither*) towards targets (e.g., *Donald Trump*, *Feminism*).
7. **Wikipedia** Text Simplification Dataset (Horn et al., 2014): Pairs of Wikipedia sentences and their simplified forms.

From each source, we sampled sentences that were between 5 and 25 words long. We selected sentence pairs with varying amounts of lexical overlap because randomly sampling sentence pairings would result in mostly unrelated sentences. This also allowed us to systematically study the impact of lexical overlap on semantic relatedness. For the paraphrase datasets (Formality, ParaNMT, and Wikipedia), we obtained sentence pairs in two ways: by directly taking the paraphrase pairs (indicated by the suffix *_pp*), and by randomly pairing sentences from two different paraphrase pairs (suffixed by *_r*). The paraphrase pairs were selected at random from the source dataset, whereas the lexical overlap strategy was applied in the creation of

Types of Pairs	Key Attributes	# pairs
1. Formality	paraphrases, style	
Formality_pp	paraphrases, differ in style	300
Formality_r	random pairs	700
2. Goodreads	reviews, informal	1000
3. ParaNMT	automatic paraphrases	
ParaNMT_pp	automatic paraphrases	450
ParaNMT_r	random pairs	300
4. SNLI	captions of images	750
5. STS	have similarity scores	250
6. Stance	tweet pairs with same hashtag, less grammatical	750
7. Wikipedia	formal	
Wiki_pp	paraphrases, formal	500
Wiki_r	random pairs, formal	500
ALL		5500

Table 2: Summary of sentence pair types in STR-2022.

the random pairs. From STS, we randomly sampled 50 sentence pairs having similarity scores in [0-1), 50 pairs having scores in [1-2), and so on.

Initially, we sought to annotate 1000 sentence pairs from each source. As our goal was to cover as large a range of relatedness, sentence structures, lengths and topics as possible, we lowered the amount of sentence pairs to obtain the desired variety. For example, SNLI pairs had little variation in sentence length and so we reduced the number of sampled instances; their semantic relatedness also tends to skew towards the higher range, and we aimed to balance this out with a larger number of random, non-paraphrasal pairs from other sources (Formality). Table 2 summarizes key details of the sentence pairs in STR-2022. Further details about the source data and sampling are in Appendix A.

3.2 Annotating For Semantic Relatedness

From the list of 5,500 sentence pairs, we generated 11,000 unique 4-tuples (each 4-tuple consists of 4 distinct sentence pairs) such that each sentence pair occurs in around eight 4-tuples.[3]

In our framing of the task, we did not use detailed or technical definitions; rather, we provided brief and easy-to-follow instructions, gave examples, and encouraged annotators to rely on their intuitions of the English language to judge relative closeness in meaning of sentence pairs (similar to Asadi et al.'s (2019) work on bigrams). Annotators were asked to judge the "*closeness in meaning of sentence pairs*". Inspired by early work in linguistics on cohesion in text (Halliday and Hasan, 1976), we also specified that: "*Often sentence pairs that are more specific in what they share tend to be more related than sentence pairs that are only loosely about the same topic*" and "*If a sentence has more than one interpretation, consider that meaning which is closest to the meaning of the other sentence in the pair.*" This is inline with application scenarios where often relatedness is to be determined between sentences from the same document. The full questionnaire is included in the supplementary material.

3.2.1 Crowdsourcing Annotations

We used Amazon Mechanical Turk (MTurk) for obtaining annotations.[4] Each 4-tuple (also referred to as a question) in our MTurk task consists of four sentence pairs. Annotators are asked to choose the (a) most-related, and (b) least-related sentence pairs from among these four options. Each question is annotated by two MTurk workers.[5]

For quality control, the task was open only to fluent speakers of English and those MTurk workers with an approval rate higher than 98%. Further, we inserted "Gold Standard" questions at regular intervals in the task. These questions were manually annotated by all the authors, and had high agreement scores. If an annotator gets a gold question wrong, they are immediately notified and shown the correct answer. This has several benefits, including: keeping the annotator alert and clearing any misunderstandings about the task. Those who scored less than ∼70% on the gold questions were stopped from answering further questions and were paid for their work. All their responses were discarded.

3.2.2 Annotation Aggregation

We aggregate information from various responses by using the counting procedure discussed in §2.3. Since relatedness is a unipolar scale, the resulting relatedness score was linearly transformed to fit within a 0–1 scale of increasing relatedness. Appendix Table 8 presents sample sentence pairs from each data source.

Figure 1 presents a histogram of relatedness scores for STR-2022. Observe that each of the subsets covers a wide range of relatedness scores; that the lexical overlap sampling strategy has resulted in a wide spread of relatedness scores; and that supposed paraphrases are spread across much of the right half of the relatedness scale.

---

[3] The tuples were generated using the BWS scripts provided by Kiritchenko and Mohammad (2017): `http://saifmohammad.com/WebPages/BestWorst.html`.

[4] This project was approved by the first author's Institutional Research Ethics Board (Protocol #: Masked for review).

[5] Pilot studies showed that this results in reliable scores.

# Sentence Pairs	# Tuples	# Annotations Per Tuple	# Annotations	# Annotators	SHR
5,500	11,000	8	21,936	389	0.84

Table 3: Annotation statistics of STR-2022. SHR = split-half reliability (as measured by Spearman correlation).

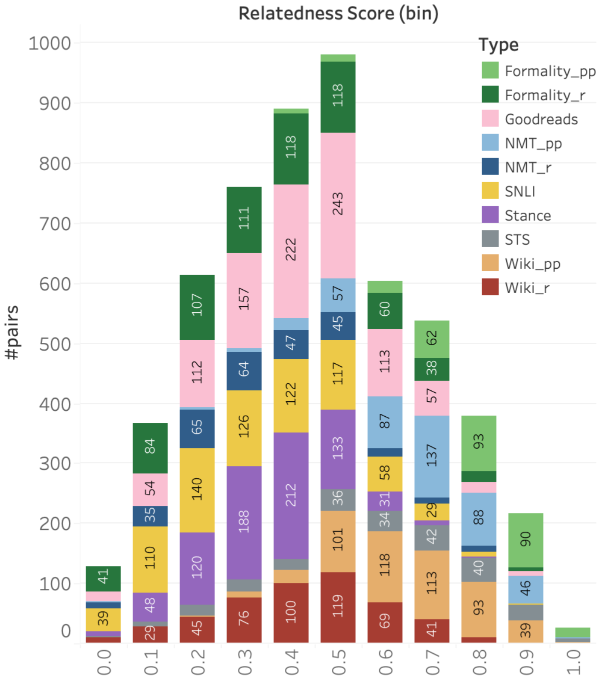


Figure 1: Histogram of STR-2022 relatedness scores.

4 Reliability of Annotations

For annotations producing real-valued scores, a commonly used measure of quality and reliability is *split-half reliability* (SHR) (Cronbach, 1951; Kuder and Richardson, 1937). SHR is a measure of the degree to which repeating the annotations would result in similar relative rankings of the items. To measure SHR, annotations for each 4-tuple are split into two bins. The annotations for each bin are used to produce two different independent relatedness scores. Next, the Spearman correlation between the two sets of scores is calculated—a measure of the closeness of the two rankings. If the annotations are reliable then there should be a high correlation. This process is repeated 1000 times and the correlation scores are averaged.

As shown in Table 3, STR-2022 has an SHR of 0.84—signifying high annotation reliability. This is a key result of this paper. Recall that our annotation guidelines did not hard code the various scenarios of sentence pair types and how they should be judged, but rather were designed to elicit how native speakers of English naturally judge relatedness. The high reliability of annotations, despite this, shows that speakers of a language are inherently consistent in their judgments of relatedness. It also validates our approach as a way to produce high-quality relatedness datasets; which, in turn, can be used to study the mechanisms underpinning relatedness (as we explore in the next Section).

4.1 STR vs STS

We also conducted experiments to assess fine-grained rankings of common sentence pairs as per our relatedness scores and as per STS's similarity scores. For each of the sets of 50 sentence pairs taken from STS (with scores in (0–1], (1–2], etc.), we calculated the Spearman correlation between the rankings by similarity and rankings by relatedness. We found that the correlations are only 0.25 (weak) and 0.19 (very weak) for the bins of (1,2] and (3,4], respectively, and only about 0.49 (moderate) for the bins of (2,3] and (4,5]. Overall, this shows that the fine-grained ranking of items in the STS dataset by similarity differ considerably from that of the STR dataset.

5 What Makes Sentences More Semantically Related?

The availability of a dataset with human notions of semantic relatedness allows one to explore fundamental aspects of meaning: for example, what makes two sentences more related? In this section, we examine some basic questions. On average, to what extent is the semantic relatedness of a sentence pair impacted by presence of:

- identical words (lexical overlap)? (Q1)
- related words? (Q2)
- related words of the same part of speech? (Q3)
- related subjects, related objects? (Q4)

5.1 Method

To explore the questions above, we first computed relevant measures for Q1 through Q4 (lexical overlap, term relatedness, etc.) for each sentence pair in our dataset. We then calculated the correlations of these scores with the gold relatedness scores.

Lexical Overlap. A simple measure of lexical overlap between two sentences X and Y is the Dice Coefficient (the number of unique unigrams occurring in both sentences, adjusted by their lengths):

$$\frac{2 \times |\, unigram(X) \cap unigram(Y)\,|}{|\, unigram(X)\,| + |\, unigram(Y)\,|} \tag{1}$$

Question	Spearman	# pairs
Q1. Lexical overlap	0.57	5500
Q2. Related words - All	0.61	5500
Q3a. Related words - per POS		
PROPN	0.50	1907
NOUN	0.45	4746
ADJ	0.36	2236
VERB	0.31	3946
PRON	0.30	1800
ADV	0.28	1147
AUX	0.25	2069
ADP	0.23	2476
DET	0.20	3265
Q3b. Related words - per POS group		
Noun Group	0.60	5478
Verb Group	0.32	4999
ADJ Group	0.29	4584
Q4. Related Subjects and Objects		
Subject	0.29	1611
Object	0.43	1618

Table 4: Correlation between features and the relatedness of sentence pairs. A rule of thumb for interpreting the numbers: 0–0.19: very weak; 0.2–.39: weak; 0.4–0.59: moderate; 0.6–0.79: strong; 0.8–1: very strong.

Related Words: We averaged the embeddings for all the tokens in a sentence and computed the cosine between the averaged embeddings for the two sentences in a pair. This roughly captures the relatedness between the terms across the two sentences.[6] Token embeddings were taken from Google's publicly released Word2Vec embeddings trained on the Google News corpus (Mikolov et al., 2013a).
Related Words with same POS: The same procedure was followed as for Q2, except that only the tokens for one part of speech (POS) at a time were considered. We determined the part-of-speech of the tokens using spaCy (Honnibal et al., 2020).[7]

Related Subjects and Related Objects: For Q4, which examines the importance of different parts of a sentence, we employ the same process as Q2, except that for a given sentence: only tokens marked as subject are averaged; and only tokens marked as object are averaged. We use the packages spaCy (Honnibal et al., 2020) and Subject Verb Object Extractor (de Vocht, 2020) to determine all tokens that are the subject and object.

5.2 Results

Table 4 shows the results. Row Q1 shows that simple word overlap obtains a correlation of 0.57 (considered to be at the high end of weak correlation). Figure 2 is a scatter plot where the x-axis is the word overlap score, the y-axis is the relatedness score, and each dot is a sentence pair. Observe that a number of pairs fall along the diagonal; however, there are also a large number of pairs along the top-left side of this diagonal. This suggests that even though STR-2022 has pairs where the relatedness increases linearly with the amount of word overlap, there are also a number of pairs where a small amount of word overlap results in substantial amount of relatedness. The sparse bottom-right side of the plot indicates that it is rare for there to be substantial word overlap, and yet very low relatedness. On average, occurrence of related words across a sentence pair leads to slightly higher relatedness scores than lexical overlap (row Q2).

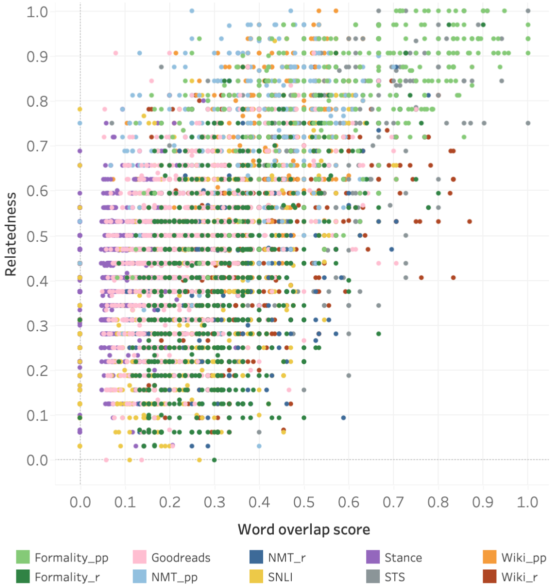


Figure 2: Scatter plot showing the relationship between lexical overlap and semantic relatedness of sentence pairs. Each dot in the plot is a sentence pair.

The Q3a rows in Table 4 show correlations for related tokens of a given part of speech.[8] (The rows are in order from highest to lowest correlation.) Observe that proper nouns (PROPN) and nouns have the highest numbers. It is somewhat surprising that related verbs do not contribute greatly to semantic relatedness; they have similar correlations as pronouns and adverbs, and markedly lower than adjectives and nouns. Not surprisingly, determiners (DET) are at the lower end of weak correlation.

The Q3b rows show correlations of coarse POS categories: NOUN Group (NOUN, PRON, PROPN), VERB Group (VERB, AUX), and ADJ

---

[6] Other ways to estimate relatedness between sets of words across two sentences may also be used.

[7] We used the simple (coarse-grained) UPOS part-of-speech tags: https://universaldependencies.org/docs/u/pos/

[8] Only those POS tags that occur in both sentences of a pair in more than 10% of the pairs are considered (>550 pairs).

Group (ADJ, ADP, ADV). We see that presence of related nouns in a sentence pair impacts semantic relatedness much more than any other POS group.

Since related nouns were found to be especially important, we also wanted to determine what impacts overall relatedness more: the presence of related nouns in the subject position or in the object position. Q4 rows show that, on average, related objects lead to markedly higher sentence-pair relatedness than related subjects.

In order to examine whether lexical overlap and some POS are less or more relevant in low or high relatedness pairs, we repeated the experiment of Table 4, only for pairs with relatedness scores <0.5, and separately, only for pairs with scores ≥0.5. We find that for the <0.5 relatedness pairs, only the existence of related proper nouns across sentence pairs has moderate correlation with the semantic relatedness of sentences; the correlation is weak for nouns, and close to 0 for all other parts of speech. The notable importance of related proper nouns and nouns is likely because they indicate a common topic, person, or object being talked about in both sentences—making the two sentence pairs related. For the ≥0.5 relatedness pairs, the correlations are weak for most POS; highest for nouns; and the gap between nouns and adjectives, adverbs, and verbs is reduced. Lexical overlap in general has a much higher correlation for the ≥0.5 relatedness pairs than the <0.5 pairs. Detailed results are in Appendix B.

6 Evaluating Sentence Representation Models using STR-2022

Since STR-2022 captures a wide range of fine-grained relations that exist between sentences, it is a valuable asset in evaluating sentence representation and embedding models. Essentially, predicting semantic relatedness is treated as a regression task, where first, using various unsupervised and supervised approaches described in the two sub-sections below, we represent each sentence as a vector. We use the cosine similarity between the vectors as a prediction of their semantic relatedness. We use the Spearman correlation between the prediction and gold relatedness scores to measure the goodness of the relatedness predictions (and in turn of the sentence representation).

The experiments below (unless otherwise specified) all involve 5-fold cross-validation (CV) on STR-2022. We report the average of the Spearman correlations across the folds. Note that even for models that do not require training (e.g., Dice score), to enable direct comparisons with trained methods, we evaluate their performance on each test fold independently and report the average of the correlations across folds.

Model	**Spearman**
Baseline	
1. Lexical overlap (Dice)	0.57
Unsupervised, Static Embeddings	
2. Word2Vec (mean, Googlenews)	0.60
3. Word2Vec (max, Googlenews)	0.54
4. GloVe (mean, Common Crawl)	0.49
5. GloVe (max, Common Crawl)	0.56
6. GloVe (mean, 200_Twitter)	0.44
7. GloVe (max, 200_Twitter)	0.48
8. Fasttext (mean, Common crawl)	0.29
9. Fasttext (max, Common crawl)	0.24
Unsupervised, Contextual Embeddings	
10. BERT-base (mean)	0.58
11. BERT-base (max)	0.55
12. BERT-base (cls)	0.41
13. RoBERTa-base (mean)	0.48
14. RoBERTa-base (max)	0.47
15. RoBERTa-base (cls)	0.41
Supervised (Fine-tuning on portions of STR-2022)	
16. BERT-base (mean)	0.82
17. RoBERTa-base (mean)	0.83

Table 5: Average correlation between human annotated relatedness of sentence pairs and the cosine distance between their embeddings across the CV runs.

6.1 Do Unsupervised Embeddings Capture Semantic Relatedness?

We first explore unsupervised approaches to sentence representation where the embedding of a sentence is derived from that of its constituent tokens. The token embedding can be of two types:

- **Static Word Embeddings:** We tested three popular models: Word2Vec (Mikolov et al., 2013b), GloVe (Pennington et al., 2014), and Fasttext (Grave et al., 2018).
- **Contextual Word Embeddings:** We tested pre-trained contextual embeddings from BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). We use the bert-base-uncased and roberta-base models from the HuggingFace library.[9]

We obtain sentence embeddings by both mean-pooling and max-pooling the token embeddings from the final layer. For the contextual embeddings, we also explore using the embedding of the classification token ([CLS]).

Table 5 shows the results. As baseline, we include how well simple lexical overlap (Dice score)

---

[9] `https://huggingface.co`

	Dice	SBERT(RoBERTa)	
	CV	CV	LOO CV
STS	0.60	0.79	0.82
SNLI	0.53	0.80	0.77
Stance	0.20	0.49	0.39
Goodreads	0.44	0.73	0.70
Wiki	0.48	0.79	0.75
Formality	0.69	0.86	0.83
ParaNMT	0.44	0.80	0.79

Table 6: Breakdown of average test-fold correlations for each source: (a) using lexical overlap (Dice), (b) using SBERT and some in-domain data for fine-tuning (in addition to data from other domains), and (c) using SBERT and only out-of-domain data for fine-tuning (LOO CV). CV: cross-validation. LOO: leave-one-out.

predicts relatedness (row 1). Observe that mean-pooling with word2vec (row 2) obtains slightly higher correlation than the baseline, but the majority of the static embedding models fail to obtain better correlations (rows 3–9). The contextual embeddings from BERT and RoBERTa do not perform better than the word2vec embeddings (rows 10–15). Overall, the unsupervised methods leave much room for improvement.

6.2 Do Supervised Embeddings Capture Semantic Relatedness?

We now evaluate the performance of BERT-based models on STR-2022 when formulated as a *supervised* regression task. We use the S-BERT cross-encoder framework of Reimers and Gurevych (2019), and apply mean-pooling on top of the token embeddings of the final layer to obtain sentence embeddings. The model is trained using a cosine-similarity loss—the cosine between the embeddings of a sentence pair is compared to the gold semantic relatedness scores to obtain the Mean Squared Error (MSE) loss for each datapoint.

Table 5 rows 16 and 17 show the results: fine-tuning on STR-2022 leads to considerably better relatedness scores.

6.2.1 Impact of Domain on Fine-Tuning

The results above show that fine-tuning is critical for better sentence representation. However, it is well-documented that the domain of the data can have substantial impact on results; especially when quite different from the training data. With the inclusion of data from various domains in STR-2022 (Table 2), one can systematically explore performance on individual domains, as well as the extent to which performance may drop if no training data from the target domain is included for training.

Table 6 shows the results. The RoBERTa CV column shows a breakdown of results by source (domain). Essentially, these are results for the scenario where some portion of in-domain data is included in the training folds (along with data from other domains), and the system correlations are determined only on the test fold's target domain pairs. Observe that performance on most domains is comparable to each other.

The LOO CV column shows correlations with a leave-one-out cross-validation setup: no in-domain training data is used and system correlations are determined only for the target domain pairs. Observe that this leads to drops in scores for all domains except STS. However, the drop is small; and scores are still much higher than the lexical overlap (Dice CV) baseline. This suggests that the diversity of data in the remaining subsets is useful in overcoming a lack of in-domain training data.

7 Utility of Semantic Relatedness and STR-2022 in Downstream NLP Tasks

Semantic relatedness is central to textual coherence and narrative structure. Often, sentences in a document are not paraphrases, entailments, or similar, but rather semantically related to each other. This need for continuity of meaning has long been identified as a crucial component of language (Halliday and Hasan, 1976; Morris and Hirst, 1991). Thus, when generating a summary or a response to a question, systems must choose sentences that are *not* paraphrases or entailments of each other, but yet suitably semantically related. Therefore, being able to judge both similarity and relatedness is crucial.

Since we made STR-2022 publicly available, it has already been used in some projects. Notable among these is Wang et al. (2022). Wang et al. (2022) propose a new intrinsic evaluation method, *EvalRank*, that focuses on local neighborhoods (how well systems identify close neighbors, rather than how well they rank the full set of pairs). Using STR-2022, they are able to obtain markedly higher correlations between performance scores on the intrinsic evaluation and performance on downstream tasks (seven NLP tasks including NLI, question classification, caption retrieval, and sentiment analysis). Their ablation study demonstrates that using STS instead of STR-2022 decreases performance up to 10 points, leading them to conclude that STR-2022 is particularly useful in generating sentence embeddings for downstream tasks.

8 Conclusion

We created STR-2022, the first dataset of English sentence pairs annotated with fine-grained relatedness scores. We used a comparative annotation method that produced a split-half reliability of 0.84. Thus, we showed that speakers of a language can reliably judge semantic relatedness. We used the dataset to explore several research questions pertaining to what makes two sentences more related. Finally, we used STR-2022 to evaluate the ability of sentence representation methods to embed sentences in vector spaces such that those that are closer to each other in meaning are also closer in the vector space. The dataset is made freely available; facilitating further research in semantic relatedness and sentence representation.

9 Limitations

For limitations surrounding the dataset, please refer to the Ethics Statement (§10). In our experiments, we used the most common methods for sentence representations (e.g., mean-pooling and max-pooling of traditional and contextual word embeddings). However, there may exist other embeddings which are better suited for predicting semantic relatedness (e.g., other order-aware embeddings). Expanding the set of embedding techniques tested using our dataset may yield different results and provide us a stronger understanding of the effects of different representation techniques. Furthermore, while we explored the impact of some sentence-pair features such as lexical overlap, POS, and some aspects of sentence structure (subject and object) on semantic relatedness, we did not explore the impacts of other features such as logicality and common sense reasoning on relatedness. These remain interesting directions for future work.

10 Ethics Statement

This paper respects existing intellectual property by making use of only publicly and freely available datasets. The crowd-sourced task was approved by our Institutional Research Ethics Board. The annotators were based in the United States of America and were paid the federal minimum wage of $7.25 per hour. Our annotation process stored no information about annotator identity and as such there is no privacy risk to them. The individual sentences selected did not have any risks to privacy either (as evaluated by manual annotation of the sentences). Models trained on this dataset may not generalize to external datasets gathered from different populations. Knowledge about language features may not generalize to other languages.

Any dataset of semantic relatedness entails several ethical considerations. We list some notable ones below. Many of these were first introduced in the context of sentiment lexicons (Mohammad, 2020, 2023). We adapted them to semantic relatedness datasets and added to the discussion.

- *Coverage:* We sampled English sentences from a diverse array of sources from the internet, with a focus on social media. Yet, it is likely that several types of sentences (and several demographic groups) are not well-represented in STR-2022. The dataset likely includes more sentences by people from the United States and Europe and with a socio-economic and educational backgrounds that allow for social media access.
- *Not Immutable:* The relatedness scores do not indicate an inherent unchangeable attribute. The relatedness can change with time, but the dataset entries are largely fixed. They pertain to the time they are created.
- *Socio-Cultural Biases:* The annotations of relatedness capture various human biases. These biases may be systematically different for different socio-cultural groups. Our data was annotated by US annotators, but even within the US there are diverse socio-cultural groups.
- *Inappropriate Biases:* Our biases impact how we view the world, and some of the biases of an individual may be inappropriate. For example, one may have race or gender-related biases that may percolate subtly into one's notions of how related two units of text are. Our dataset curation was careful to avoid sentences from problematic sources, and we have not seen any inappropriate relatedness judgments, but it is possible that some subtle inappropriate biases still remain. Thus, as with any approach for sentence representation or semantic relatedness, we caution users to explicitly check for such biases in their system regardless of whether they use STR-2022.
- *Perceptions (not "right" or "correct" labels):* Our goal here was to identify common perceptions of semantic relatedness. These are not meant to be "correct" or "right" answers, but rather what the majority of the annotators believe based on their intuitions of the English language.

- *Relative (not Absolute):* The absolute values of the relatedness scores themselves have no meaning. The scores help order sentence pairs relative to each other. For example, a pair with a higher relatedness score should be considered more related than a pair with a lower score. No claim is made that the mid-point (relatedness score of 0.5) separates related words from unrelated words. One may determine categories such as *related* or *unrelated* by finding thresholds of relatedness scores optimal for their use/task.

We recommend careful reflection of ethical considerations relevant for the specific context of deployment when using STR-2022.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. SemEval-2015 Task 2: Semantic Textual Similarity, English, Spanish and Pilot on Interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 Task 10: Multilingual Semantic Textual Similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. SemEval-2016 Task 1: Semantic Textual Similarity, Monolingual and Cross-lingual Evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics).

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic Textual Similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43.

Shima Asaadi, Saif Mohammad, and Svetlana Kiritchenko. 2019. Big BiRD: A Large, Fine-Grained, Bigram Relatedness Dataset For Examining Semantic Composition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 505–516.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large Annotated Corpus for Learning Natural Language Inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

Lee J Cronbach. 1951. Coefficient Alpha and the Internal Structure of Tests. *Psychometrika*, 16(3):297–334.

Herbert Aron David. 1963. The Method of Paired Comparisons. In *Proceedings of the Fifth Conference on the Design of Experiments in Army Research Developments and Testing*.

Peter de Vocht. 2020. Python Package: Subject Verb Object Extractor. Github.

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

William B Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Terry N Flynn and Anthony AJ Marley. 2014. Best-Worst Scaling: Theory and Methods. In *Handbook of Choice Modelling*. Edward Elgar Publishing.

Édouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomáš Mikolov. 2018. Learning Word Vectors for 157 Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Michael Alexander Kirkwood Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman Group.

Sébastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. 2015. *Semantic Similarity from Natural Language and Ontology Analysis*. Morgan & Claypool Publishers.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. Python Package; spaCy: Industrial-strength Natural Language Processing in Python. Zenodo.

Colby Horn, Cathryn Manduca, and David Kauchak. 2014. Learning a Lexical Simplifier Using Wikipedia. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 458–463.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-Worst Scaling More Reliable than Rating Scales: A Case Study on Sentiment Intensity Annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470.

Svetlana Kiritchenko and Saif M. Mohammad. 2016. Capturing reliable fine-grained sentiment associations by crowdsourcing and best–worst scaling. In *Proceedings of The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, San Diego, California.

G Frederic Kuder and Marion W Richardson. 1937. The Theory of the Estimation of Test Reliability. *Psychometrika*, 2(3):151–160.

Yuhua Li, David McLean, Zuhair A Bandar, James D O'shea, and Keeley Crockett. 2006. Sentence Similarity Based on Semantic Nets and Corpus Statistics. *IEEE Transactions on Knowledge and Data Engineering*, 18(8):1138–1150.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Jordan J Louviere and George G Woodworth. 1991. Best-Worst Scaling: A Model For The Largest Difference Judgments. Technical report, Working paper.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2014)*, pages 216–223. Reykjavik.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

George A Miller and Walter G Charles. 1991. Contextual Correlates of Semantic Similarity. *Language and Cognitive Processes*, 6(1):1–28.

Saif Mohammad. 2008. *Measuring Semantic Distance Using Distributional Profiles of Concepts*. Ph.D. thesis, University of Toronto.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. A Dataset for Detecting Stance in Tweets. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3945–3952.

Saif M. Mohammad. 2020. Practical and ethical considerations in the effective use of emotion and sentiment lexicons.

Saif M. Mohammad. 2023. Best practices in the creation and use of emotion lexicons.

Saif M Mohammad and Graeme Hirst. 2012. Distributional Measures of Semantic Distance: A Survey. *arXiv preprint arXiv:1203.1858*.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):21–48.

Bryan Orme. 2009. MaxDiff Analysis: Simple Counting, Individual-Level Logit, and Hb. *Sawtooth Software*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Stanley Presser and Howard Schuman. 1996. *Questions and Answers in Attitude Surveys: Experiments on Question Form, Wording, and Context*. SAGE Publications, Inc.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A Word at a Time: Computing Word Relatedness Using Temporal Semantic Analysis. In *Proceedings of the 20th International Conference on World Wide Web*, pages 337–346.

Sudha Rao and Joel Tetreault. 2018. Dear Sir or Madam, May I Introduce the GYAFC Dataset: Corpus, Benchmarks and Metrics for Formality Style Transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Peter Mark Roget. 1911. *Roget's Thesaurus of English Words and Phrases...* TY Crowell Company.

Herbert Rubenstein and John B Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10):627–633.

Louis L Thurstone. 1927. A Law of Comparative Judgment. *Psychological Review*, 34(4):273.

Mengting Wan and Julian J. McAuley. 2018. Item Recommendation on Monotonic Behavior Chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 86–94. ACM.

Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian J. McAuley. 2019. Fine-Grained Spoiler Detection from Large-Scale Review Corpora. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2605–2610. Association for Computational Linguistics.

Bin Wang, C-C Jay Kuo, and Haizhou Li. 2022. Just rank: Rethinking evaluation with word and sentence similarities. *arXiv preprint arXiv:2203.02679*.

John Wieting and Kevin Gimpel. 2018. ParaNMT-50M: Pushing the Limits of Paraphrastic Sentence Embeddings with Millions of Machine Translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Melbourne, Australia. Association for Computational Linguistics.

A Further Details on Sampling Sentence Pairs from Source Datasets

This Appendix provides further information about the sources of data and how sentence pairs were sampled from them to be included in STR-2022.

A.1 STS Data

We selected 250 sentence pairs from existing STS corpora. This selection was done to enable a small investigation into the interplay between relatedness and similarity, which could serve as motivation for further investigation in future work. For this dataset, we randomly sampled 50 sentence pairs from each of bucket of annotation (i.e., 50 sentence pairs having an STS similarity scores falling in $[0, 1)$, 50 sentence pairs having scores in $[1, 2)$, and so on).

A.2 Stance Data

We created 750 sentence pairs by sampling from Mohammad et al. (2016)'s dataset of tweets labeled for stance. The original dataset is composed of individual tweets labelled for both stance ('For', 'Against', 'Neither Inference Likely') and sentiment ('Positive', 'Negative', 'Neutral'). The dataset was built from tweets focused on six targets: 'Atheism', 'Climate Change', 'Donald Trump', 'Feminism', 'Hillary Clinton', 'Abortion'.

When curating our sentence pairs, we limited the possible targets to 'Hillary Clinton', 'Donald Trump', and 'Abortion'. Sentence pairs were chosen such that both sentences shared the same target. 500 sentence pairs shared their stance towards their target (i.e., 250 *for–for* pairs and 250 *against–against* pairs). 250 sentences pairs differed on their stance (i.e., *250 for–against* pairs). We did not use any lexical overlap heuristic to specify which tweets should be paired with each other because we were interested in studying whether overlap in topic was a strong enough signal to impact relatedness. That is, by choosing pairs with the same target, we were already pre-selecting for various degrees of relatedness.

A.3 SNLI Data

We created 750 sentence pairs by sampling from the Stanford Natural Language Inference (SNLI) Dataset (Bowman et al., 2015). SNLI is composed of image description captions; for each caption, multiple premise sentences are generated, along with multiple possible hypothesis sentences that could possibly belong to each premise. To build our sentence pairs we sought to pair different premise sentences together. We did not wish to pair between premise and hypothesis sentences as the sentence structure was significantly different (and simpler for the hypothesis sentences), as noted by the creators of the dataset. Even still, the majority of premise sentences are very short (with a mean token count of 14), often following very simple (and similar) grammatical structure.

To generate the sentence pairs, first we removed all sentences with less than 5 or more than 25 tokens. Then, for each token in all remaining sentences, we replaced each token with its most frequent synonym, using Roget's Thesaurus (Roget, 1911) to define synonymous relationships. Words which did not have synonyms were left unchanged. The intention behind replacing each word with its most frequent synonym was to ensure that synonymous phrasings would count as overlaps when we measure it. We then randomly selected 750 sentences to serve as the first sentence of our final pairings. To find the second sentence to each pairing we looped through all premise sentences and returned the first sentence that satisfied two conditions: 1) The unigram overlap was greater than or equal to 25% and less than 75% of the first sentence, and 2) the difference in length between both sentences did not exceed 25%.

A.4 Wikipedia Data

We sampled 1000 sentence pairs from a dataset that pairs sentences from English Wikipedia with sentences from Simple English Wikipedia. Created to enable the task of sentence simplification, the paired sentences, paired using rules-based classification, are often very closely related. We used this dataset in two ways: 1. Extracting sentence pairs which serve as paraphrases or near paraphrases (we refer to these as Wiki_pp), and 2. pairing sentences to other random sentences in the dataset (we refer to these as Wiki_r).

Wiki_pp: First, we removed any pairings for which either sentence was less than 5 words or more than 25 words. Then we narrowed the list of pairings further by removing any pairings that did not share more than 25% but less than 75% of unique unigrams. From the remaining sentence pairs, we randomly selected 500 paired sentences.

Wiki_r: Here, we only made use of the full sentences from the original Wikipedia, discarding sentences from Simple Wikipedia. We removed all sentences that have less than 5 or more than 25 tokens. To create the sentence pairs, we looped in a random order through all possible pairing of sentences. We paired two sentences if they share at least 25% of their tokens but less than 75% of their tokens AND the difference in length between both sentences did not exceed 25%. We stopped once we had generated 500 sentence pairs.

A.5 Goodreads Data

We created 1000 sentence pairs by sampling from the UCSD Goodreads Dataset (Wan and McAuley, 2018; Wan et al., 2019), which has book reviews from the Goodreads website. We limited the sampling to the 'Fantasy and Paranormal' genre, since it contained a relatively higher number of reviews per book, allowing for a higher possibility of sampling more related sentence pairs. Each review was first split into sentences using the default NLTK sentence tokenizer; we kept only those sentences with the number of tokens between 5 and 25. We then randomly examined pairs of sentences, and quantified the lexical overlap between then with an IDF-weighted Dice overlap score. The pairs were then assigned to buckets based on this overlap score; the range of each bucket was obtained by first finding 50 equally-spaced percentiles of the entire score distribution. We then sampled exponentially increasing number of sentences from low to high weighted Dice overlap bins such that a total of 1000 sentence pairs were included.

A.6 ParaNMT Data

ParaNMT (Wieting and Gimpel, 2018) is a dataset of 51 million sentential paraphrases that were automatically generated using a neural machine translation system. We generated two sets of pairs from these sentences corresponding to paraphrases and random pairs:

ParaNMT_pp: We assigned paraphrases to buckets based on the Dice score between the two sentences. We divided the range of scores into 100 equally-sized percentiles. We then sampled pairs uniformly from each bucket, for a total of 450 sentence pairs.

ParaNMT_r: For the random, non-paraphrase sentence pairings, we used the Dice score to extract 300 pairs, analogous to the creation of the **Wiki_r** pairs.

A.7 Formality Data

Our third paraphrase corpus is the Formality dataset from Rao and Tetreault (2018) (They refer to it as GYAFC). This consists of human-written formal and informal paraphrases for sentences sourced from the Yahoo! Answers platform. Our sampling procedure for this dataset followed that of the ParaNMT dataset.

Formality_pp: We assigned sentences to one of 50 buckets based on their lexical overlap score as before. We then uniformly sampled from each bucket to extract 300 sentence pairs.

Formality_r: We sampled random pairings of sentences using the token overlap and length difference conditions as defined for Wiki_r and ParaNMT_r. We extracted 700 such sentence pairs.

B Correlation of Features in Low and High Relatedness Sentence Pairs

As discussed in Section 5.2, in order to examine whether lexical overlap and some parts of speech are less or more relevant in low or high relatedness pairs, we repeated the experiment in Table 4, only for pairs with relatedness scores less than 0.5 and also for pairs with scores greater than 0.5. Table 7 shows the detailed correlation scores. See Section 5.2 for a discussion of the main trends.

C Sample Sentence Pairs from STR-2022

Table 8 presents sample sentene pairs from different domains.

Question	Spearman 0–1 pairs	 <0.5 pairs	 ≥0.5 pairs
Q1. Lexical overlap	0.57	0.14	0.52
Q2. Related words - All	0.61	0.14	0.50
Q3a. Related words - per POS			
PROPN	0.50	0.34	0.26
NOUN	0.45	0.18	0.37
ADJ	0.36	0.04	0.35
VERB	0.31	0.03	0.31
PRON	0.30	0.01	0.30
ADV	0.28	0.04	0.35
AUX	0.25	0.03	0.20
ADP	0.23	0.07	0.22
DET	0.20	0.03	0.19
Q3b. Related words - per POS group			
Noun Group	0.60	0.34	0.41
Verb Group	0.32	0.09	0.29
ADJ Group	0.29	0.04	0.32
Q4. Related Subjects and Objects			
Subject	0.29	0.00	0.32
Object	0.43	0.14	0.33

Table 7: Correlation between features and the relatedness of sentence pairs in STR-2022 when considering full relatedness range (0–1), only the pairs with relatedness < 0.5, and only the pairs with relatedness ≥ 0.5.
Note: The 0–1 pairs column was shown earlier in Table 4. It is repeated here for ease of comparison.

Source	Sentence Pairs	STR score
Formality_pp	*I think Taylor is really cute, but I hate his voice.* *I think Taylor is SUPER cute...but I hate his voice.*	1.000
Wiki_pp	*It is sometimes referred to as the trunk.* *Some people also call it the trunk.*	0.969
Goodreads	*I loved this short story - wish it were longer!* *It was a quick read and part of me wished that it would go on a little longer.*	0.844
Wiki_r	*On August 2 , a tropical storm hit Northeastern Florida .* *In early October , a hurricane caused damage and erosion to northeastern Florida .*	0.625
Stance	*So unfortunate #thebriefcase @cbs. Adoption isn't always the answer.* *Just remember, there is a living family out there just waiting to #adopt your aborted baby.*	0.562
SNLI	*A woman in speaking in a theater.* *deleon speaking into a mic.*	0.406
ParaNMT_pp	*Are you–are you going to tell every one?* *will you say it now – all of you?*	0.334
Formality_r	*i believe in american dreams ...* *You are the woman of my dreams*	0.219
STS	*A person is riding a horse.* *A woman is slicing potatoes.*	0.062

Table 8: Sample sentence pairs from different domains in the STR-2022 dataset.

RevUp: Revise and Update Information Bottleneck for Event Representation

Mehdi Rezaee **Francis Ferraro**
Department of Computer Science
University of Maryland Baltimore County
Baltimore, MD 21250 USA
{rezaee1, ferraro}@umbc.edu

Abstract

The existence of external ("side") semantic knowledge has been shown to result in more expressive computational event models. To enable the use of side information that may be noisy or missing, we propose a semi-supervised information bottleneck-based discrete latent variable model. We reparameterize the model's discrete variables with auxiliary continuous latent variables and a light-weight hierarchical structure. Our model is learned to minimize the mutual information between the observed data and optional side knowledge that is not already captured by the new, auxiliary variables. We theoretically show that our approach generalizes past approaches, and perform an empirical case study of our approach on event modeling. We corroborate our theoretical results with strong empirical experiments, showing that the proposed method outperforms previous proposed approaches on multiple datasets.

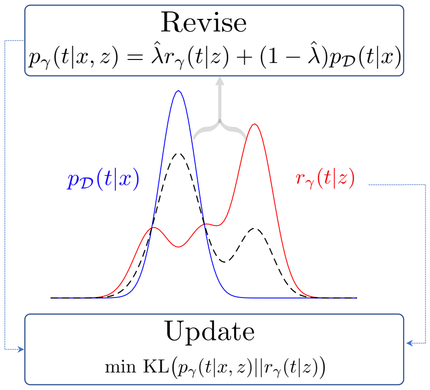


Figure 1: Revise and Update steps for an observed node. We update the proposed distribution $r_\gamma(t|z)$ with the empirical distribution $p_\mathcal{D}(t|x)$ to produce the revised distribution $p_\gamma(t|x,z)$ (dashed). We then minimize the KL divergence between the proposed and revised distributions to update the proposed distribution. For unsupervised nodes, we rely on $r_\gamma(t|z)$ without updating.

1 Introduction

In this work, we are interested in addressing limitations in how computational event modeling can make use of relevant, supplementary semantic knowledge. This is because when modeling text descriptions of complex situations, such as newspaper descriptions of real world events, learning how to encode richer information about those descriptions can be a fruitful way of improving modeling performance (Judea and Strube, 2015; Xia et al., 2021). E.g., if we are dealing with sequences of events, like a newspaper report of a stock or commerce transaction, then being able to encode that "buying" or "selling," even though they may have nuanced connotation differences, are instances of the same general event (a TRANSACTION event) can improve downstream predictive performance on what events might happen next (Ferraro and Van Durme, 2016; Rezaee and Ferraro, 2021).

However, there is not an obvious single way to learn to encode this richer information, as three different questions naturally come to mind: (1) If the model is representationally limited, can we address these representational limitations of the model itself, such as through developing richer latent representations $\mathbf{z}$ of the input $\mathbf{x}$? (2) If there is available background or side information $\mathbf{t}$ that may be especially relevant for the modeling task at hand, can we develop systems that make use of it? (3) Even when side information is available, it may be noisy, e.g., it may not always be available (missing data) or it may contain errors: how can we make our models robust to this noisy side information? In the context of a text-based sequence modeling problem, we propose an approach that addresses all three of these questions.

We provide a conceptual overview of our method—RevUp—in Fig. 1. The building blocks of RevUp are **revision** of modeling side information $\mathbf{t}$: forming a new distribution by combining the (red) learned proposal distribution with (blue) empirical information about when particular aspects of side knowledge appeared in training, and **updat-**

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 797–814
May 2-6, 2023 ©2023 Association for Computational Linguistics

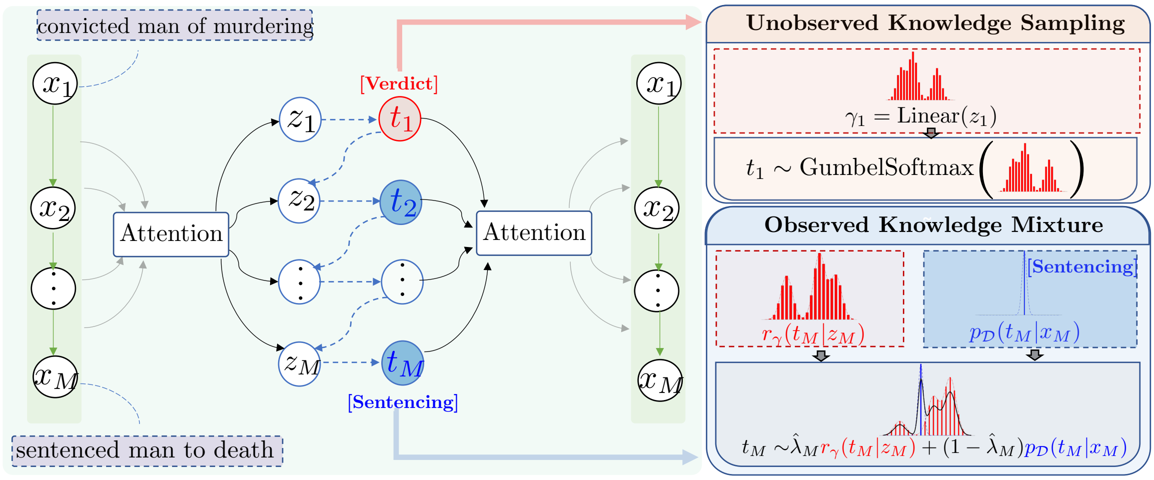


Figure 2: Components of RevUp for event modeling. We encode the sequence of events $\mathbf{x}$ into Gaussian latent $\mathbf{z}$ and discrete knowledge $\mathbf{t}$. Red nodes such as t_1 are latent: [VERDICT] for the *convicted man of murdering* event. RevUp predicts these latent nodes by sampling from the GumbelSoftmax distribution. Blue nodes such as t_M are observed: [SENTENCING] for the *sentenced man to death* event. We use this observed node to modify the proposed distribution and then we draw a sample from the revised distribution.

ing by minimizing the KL-divergence between the new distribution with the previous proposed one.

Our approach is inspired by recent work (Kong et al., 2019), which argued that a key to the success of neural advances in NLP (Mikolov et al., 2013; Devlin et al., 2018; Yang et al., 2019, i.a.,) is that they fall within the information theoretic-based InfoMax framework (Linsker, 1988). In this viewpoint, neural advances can be attributed to implicitly maximizing the mutual information between different representations of the same document.

However, we additionally want to use "side" or "external" knowledge to aid our modeling. The use of side information has long been desired for effective representation (Wyner, 1975; Wyner and Ziv, 1976). While this side information can occur in a variety of forms (Chen et al., 2020; Padia et al., 2018), we broadly view it as a *concise, abstract view of information* conveyed in the main input.

Our work centers on the idea that the latent neural representations $\mathbf{z}$ and side knowledge $\mathbf{t}$ act as complementary representations of the same input $\mathbf{x}$: $\mathbf{z}$ provides a compact representation of the input itself, while $\mathbf{t}$ provides more generic information about the data or task. As such, we use an InfoMax-inspired formulation to incorporate external knowledge into the modeling and latent representation aspects. Since recent work (Joy et al., 2022; Chen et al., 2019) has suggested that providing some sort of "guidance" to these latent variables is beneficial and can lead to these guided models outperforming both fully supervised and fully unsupervised methods, learning with less-than-full observation of structured side knowledge is a requirement. We illustrate this in Fig. 2, where the correct value for the first piece of side information ($t_1 = \text{Verdict}$) is missing but the last piece of side information is known ($t_M = \text{Sentencing}$). Our model must be able to operate over this partially observed sequence of side knowledge as a way of modeling the original event description.

We propose a unified approach, RevUp (**Rev**ise and **Up**date Information Bottleneck), that maximizes the mutual information between the external knowledge and latent representation $\mathcal{I}(\mathbf{z};\mathbf{t})$. We consider the external knowledge $\mathbf{t}$ to be a discrete random variable in a lightly structured, semi-supervised setting. We demonstrate the effectiveness of this approach on multiple modeling tasks. Our contributions are:

(1) We provide a principled, information theoretic approach for injecting side information into neural discrete latent variable models. We provide theoretical backing to show that our methodology captures available information from external knowledge.

(2) We define a new model that leads to state-of-the-art results in the semi-supervised setting on two standard event modeling datasets.

(3) We show that our proposed model generalizes the existing state-of-the-art model studied in Rezaee and Ferraro (2021), where the "parameter injection" method developed there can be

understood as a special case of our framework.

(4) We experimentally show that our model is more robust when the external knowledge is noisy and it outperforms other baselines when the external knowledge is partially observed.

Our code is available at `https://github.com/mmrezaee/RevUp`.

2 Background

In this section we present the related work and discuss the connections to our approach.

Mutual Information (MI) The mutual information $\mathcal{I}(x;y)$ measures how much two random variables x and y depend on each other. It is defined as $\mathcal{I}(x;y) = \mathbb{E}_{p(x,y)} \log \frac{p(x,y)}{p(x)p(y)}$, where $\mathcal{I}(x;y) = 0$, when x and y are independent. Conditional MI $\mathcal{I}(x;y|z)$ extends MI to measure the conditional dependence between x and y given z, as $\mathcal{I}(x;y|z) = \mathbb{E}_{p(x,y,z)} \log \frac{p(x,y|z)}{p(x|z)p(y|z)}$. When $\mathcal{I}(x;y|z) = 0$, there isn't any information between x and y that is not present in z.

Information Bottleneck Principle (IB) The Information Bottleneck principle is a method for finding the most informative encoding of an input x with respect to the target output y. This is accomplished by finding the maximally compressed encoding z of x that is most informative about y (Tishby et al., 1999). The objective is to maximize

$$\mathcal{I}(z;y) - \beta\mathcal{I}(x;z), \tag{1}$$

where $\mathcal{I}(z;y)$ denotes z and y mutual information, and $\mathcal{I}(x;z)$ denotes x and z mutual information. β balances compression and informativeness.

Variational Autoencoder (VAE) The Variational Autoencoder is similar to the Information Bottleneck Principle in that it finds an encoding z of x to reconstruct x. However, the approach is different as the aim is to approximate the intractable posterior distribution $p_\theta(z|x)$ with a variational distribution $q_\phi(z|x)$ in order to optimize the Evidence Lower Bound (ELBO) (Kingma and Welling, 2013).

Incorporation of Side Knowledge Conceptually, we consider an "event" to be a condensed form of knowledge that outlines part of a particular situation. Semantic frames are a prime example of relevant side knowledge: a semantic frame can be thought of as an abstraction over highly related events. Though they also provide abstractions over the *who*s, *what*s, *where*s and *how*s of events, in this setting, it is sufficient to consider an event's semantic frame as a type of label or cluster id. Multiple potential sources of semantic frames exists, e.g., FrameNet, PropBank, or VerbNet.

Recently, incorporating external knowledge has been investigated by numerous studies in a wide range of tasks beyond NLP (Kang et al., 2017; Flajolet and Jaillet, 2017; Zhang et al., 2021), and zero-shot classification (Badirli et al., 2021). Common across these efforts is treating the side information as part of the input to be encoded. This makes the side information prerequisite knowledge for the model to be learned, rather than supplementary.

A recent approach, called Sequential, Semi-supervised Discrete Variational Autoencoder (Rezaee and Ferraro, 2021, SSDVAE), is a new method for structured semi-supervised modeling that allows, but does not require, side information to guide the learning in an approach called "parameter injection." Because the SSDVAE framework is a deep latent variable model that is specifically designed to treat external knowledge as supplementary to the main task, we focus our study within it and the associated NLP-based computational event modeling tasks examined. They define parameter injection as

Definition. *(Parameter injection) Let* $\mathbf{t} \sim$ *GumbelSoftmax*$(\mathbf{t};\gamma)$*, where* $\boldsymbol{\gamma}$ *are the logits. If* $\mathbf{t}$ *is observed as external knowledge and represented with a one-hot vector* $\mathbb{1}(\mathbf{t})$ *with* $t_{k^*} = 1$*, the operation* $\boldsymbol{\gamma} = \boldsymbol{\gamma} + (\|\boldsymbol{\gamma}\| * \mathbb{1}(\mathbf{t}))$ *guides the latent variable* $\mathbf{t}$ *during the training because on average increases the* t_{k^*} *value (Maddison et al., 2016).*

In RevUp, we build on and generalize this notion of parameter injection, in part by introducing the empirical distribution $p_{\mathcal{D}}(\mathbf{t}|\mathbf{x})$ to accommodate the dependence between data $\mathbf{x}$ and knowledge $\mathbf{t}$.

3 RevUp: Revise and Update

Previous work (Rezaee and Ferraro, 2021, SSDVAE) has empirically shown that incorporating external knowledge in discrete GumbelSoftmax parameters improves model performance with different metrics such as classification, event modeling and training speed. We seek to re-frame this view with information theory terms and generalize it. For fairness and consistency we stick with the same types of computational event modeling tasks that

SSDVAE was developed for. We first describe the probabilistic model we develop (section 3.2) and the loss/training methodology (section 3.3). These enable smoothly incorporating side knowledge into a probabilistic neural model. Recall we provide a conceptual overview of RevUp in Fig. 1.

3.1 Setup

We assume that within a document, we have a sequence of M different predicate/argument-style events, $\mathbf{x} = x_1 ... x_M$, with each x_m describing some action (the predicate) occurring among participants of that action (the arguments). If each event x_m could also be paired with a semantic frame t_m, so we have a corresponding sequence $\mathbf{t}$, then considered across multiple events, $\mathbf{t}$ could provide a sequential generalization. E.g., in the example event sequence from Fig. 2 that includes "convicted man of murdering," and "sentenced man to death," if each event can be associated with a semantic frame (such as VERDICT and SENTENCING for these two events), then the corresponding sequence of frames provides both an abstraction over the entire event sequence, and an incredibly rich, yet low-dimensional, collection of side knowledge.

Having paired sequences $\mathbf{x}$ and $\mathbf{t}$ is not restrictive. Whether $\mathbf{t}$ is partially observed (i.e., not all x_m have observed frames t_m) or noisy (i.e., the observed t_m for x_m is incorrect) during training, our approach can still extract useful signal.

3.2 Probabilistic Encoding

Our problem setup is that we have input text $\mathbf{x}$ describing some complex situation, paired with a partially observed sequence of frames $\mathbf{t}$. To account for when side knowledge is/isn't observed, we define a knowledge indicator set $\boldsymbol{l} = l_m|_{m=1}^{M}$ with $l_m \in \{0, 1\}$, where $l_m = 1$ denotes the external knowledge t_m is present and $l_m = 0$ means the external knowledge is not available (latent). To enable successful modeling, we introduce $\mathbf{z} = z_m|_{m=1}^{M}$, a set of M latent variables to first compress the information of the given inputs $\mathbf{x}$ and then be informative regarding $\mathbf{t}$. We define $p_\theta(\mathbf{z}|\mathbf{x})$, a probabilistic encoder from data points $\mathbf{x}$ to the latent variables $\mathbf{z}$, parameterized by a neural network θ.

We define a joint model over $\mathbf{t}, \mathbf{x}$ and $\mathbf{z}$ as $p(\mathbf{t}, \mathbf{x}, \mathbf{z}; \boldsymbol{l}) = p_\mathcal{D}(\mathbf{x}) \prod_m p_\theta(z_m|t_{m-1}, \mathbf{x}) p_\gamma\big(t_m|x_m, z_m; l_m\big)$, where $p_\mathcal{D}(\mathbf{x})$ is the empirical distribution over the input variables $\mathbf{x}$. Consistent with previous approaches, $p_\theta(z_m|t^{(s)}, \mathbf{x}; l_m)$ is a Gaussian. Similarly $p_\gamma\big(t_m|x_m, z_m; l_m\big)$ posits a distribution over semi-supervised latent knowledge $\mathbf{t}$ given $\mathbf{x}$ and latent variables $\mathbf{z}$. To learn richer representations, we force $\mathbf{z}$ and $\mathbf{t}$ to depend on one other in a sawtooth fashion: t_i depends on z_i, but z_i depends on t_{i-1}. Fig. 2 shows an illustration. This is a novel segmented, autoregressive sequencing for event modeling.

We define $r_\gamma(\mathbf{t}|\mathbf{z})$ as the *proposed distribution* that relates $\mathbf{z}$ to $\mathbf{t}$, where γ is computed as the outcome of a neural encoding of $\mathbf{z}$, NN($\mathbf{z}$). This distribution must be learned. For simplicity we omit the node index m when possible.

Revision Phase Incorporating the external knowledge in the training phase must satisfy two criteria: First, without observation ($l_m = 0$), we just rely on $r_\gamma(t_m|z_m)$. Second, when we have access to external knowledge, $p_\mathcal{D}(t_m|x_m)$ should be used for *guidance* and not discarding the proposed distribution. We define the *revised distribution* $p_\gamma\big(t_m|x_m, z_m; l_m\big)$ as a smoothed weighted average of the proposed distribution and the empirical distribution as $p_\gamma\big(t_m|x_m, z_m; l_m\big) = \frac{1}{1+\lambda l_m} r_\gamma(t_m|z_m) + \frac{\lambda l_m}{1+\lambda l_m} p_\mathcal{D}(t_m|x_m)$, where $\lambda \in \mathbb{R}^+$ is a weighting parameter for balancing the proposed and empirical distributions. In this setting, λ depends on the level of confidence in our external knowledge, where for less noisy knowledge, we can choose higher values for λ. In practice, we found that $\lambda = 1.0$ works reasonably.[1]

If side knowledge is absent ($l_m = 0$), we have $\hat{\lambda}_m = 1.0$ so $p_\gamma(t_m|x_m, z_m) = r_\gamma(t_m|z_m)$: the model just uses the proposed distribution. This allows gradients to propagate through the network. During the test phase we do not use any external knowledge, so the revised distribution $p_\gamma\big(\mathbf{t}|\mathbf{x}, \mathbf{z}; \boldsymbol{l}\big)$ reduces to the proposed distribution $r_\gamma(\mathbf{t}|\mathbf{z})$.

Analysis and Insights If we define $\hat{\lambda}_m = 1/(1 + \lambda l_m)$, we rewrite this as $p_\gamma\big(t_m|x_m, z_m; l_m\big) = \hat{\lambda}_m r_\gamma(t_m|z_m) + (1 - \hat{\lambda}_m) p_\mathcal{D}(t_m|x_m)$. This slight notation change is beneficial, as it lets us characterize the behavior of our revision step in terms of expected

---

[1] In dev experiments we tried various values of λ, both large and small. When λ was high (greater weight to the empirical distribution), the model over-emphasized this signal, which was detrimental at test time when we intentionally withheld this signal. In contrast, when λ is low (greater weight to the proposed distribution), there is less emphasis on the empirical distribution. However, as that empirical distribution could itself be quite sparse (due to ϵ), the model learns to disregard it. This highlights a strength in simplicity of the equal weighting.

observability of side knowledge. Specifically, in a semi-supervised setting, if the probability of observing any particular piece of side information can be modeled as $l \sim \text{Bern}(\epsilon)$, where ϵ is the observation probability, by marginalizing out l we have $\mathbb{E}_{l\sim\text{Bern}(\epsilon)} p_\gamma(t|x,z,l) = \hat{\epsilon} r_\gamma(t|z) + (1-\hat{\epsilon}) p_{\mathcal{D}}(t|x)$, where $\hat{\epsilon} = 1 - \left(\frac{\lambda}{1+\lambda}\right)\epsilon$. This indicates that with more observation, we rely more heavily on the empirical distribution and less on the proposed distribution. For space, see Appendix E for more details/the derivation.

Finally, our framework generalizes SSDVAE's parameter injection (see Appendix F):

Theorem 1. *If $r_\gamma(t|z) = Cat(\boldsymbol{\gamma})$, a categorical distribution with parameters γ, and the empirical distribution $p_{\mathcal{D}}(t|x)$ is a one-hot representation with $t_{k^*} = 1$, the revision step reduces to SSDVAE parameter injection.*

We have described how to guide the latent variable t in the encoding phase. Next we define our objective function to capture the background information and decoding $\mathbf{t}$ to effectively model event descriptions.

3.3 Training Objective and Decoding

After training, the model relies solely on the proposed distribution $r_\gamma(t_m|z_m)$ to predict t_m, implying the statistics of t_m will only depend on z_m. To capture the background information in $\mathbf{t}$, minimize $\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z})$ during training. In information theory terms, in the ideal situation where $\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z}) = 0$, there is not any residual information between $\mathbf{x}$ and $\mathbf{t}$ that is not captured by the latent representation $\mathbf{z}$ (Kirsch et al., 2020). Therefore without the need of $\mathbf{x}$ and $p_{\mathcal{D}}(\mathbf{t}|\mathbf{x})$, the latent $\mathbf{z}$ is enough to predict $\mathbf{t}$.

To be meaningfully learned, $\mathbf{t}$ must be informative enough to make some prediction or reconstruction. For clarity, we refer to the targets as $\mathbf{y}$, though for a task like language modeling, $\mathbf{x} = \mathbf{y}$. To achieve this learning, we maximize $\mathcal{I}(\mathbf{y},\mathbf{t})$.

Together, our ideal objective is $\mathcal{L} = -\mathcal{I}(\mathbf{y};\mathbf{t}) + \alpha\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z})$, where α is a tunable hyperparameter. This is difficult to optimize because within each $\mathcal{I}$ term there are intractable marginalizations (such as over $\mathbf{x}$). To understand why this is our ideal objective, we show that maximizing the intractable mutual information $\mathcal{I}(\mathbf{t};\mathbf{z})$ is inherently included in this unified objective for the reconstruction tasks:

Theorem 2. *For tasks where we maximize $\mathcal{I}(\mathbf{x};\mathbf{t})$: minimizing $\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z})$ leads to maximizing $\mathcal{I}(\mathbf{t};\mathbf{z})$.*

This theorem shows that reconstruction tasks like language modeling explicitly maximize the mutual information between different data representations, which is consistent with the InfoMax principle. See Appendix G for the proof. As a consequence of Theorem 2, we are maximizing the mutual information between two views of $\mathbf{x}$: compressed representation $\mathbf{z}$ and side information $\mathbf{t}$.

With that understanding, we proceed to a tractable approximation for our objective. Following Alemi et al. (2016), we have $-\mathcal{I}(\mathbf{y};\mathbf{t}) = -\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log \frac{p(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})}$. We approximate this as

$$-\mathcal{I}(\mathbf{y};\mathbf{t}) \leq -\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log \frac{q_\phi(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})} \tag{2}$$

$$= \underbrace{-\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log q_\phi(\mathbf{y}|\mathbf{t})}_{\mathcal{L}_y} - H(\mathbf{y}). \tag{3}$$

Since $p(\mathbf{y}|\mathbf{t})$ is intractable, we approximate it via a decoder $q_\phi(\mathbf{y}|\mathbf{t})$ that can be computed by a neural network, denoted as ϕ. As the task entropy $H(\mathbf{y})$ is constant, we just minimize $\mathcal{L}_y$. While not explicitly reflected in Eq. 2, note that irrespective of whether side information is present or not, $\mathcal{I}(\mathbf{y};\mathbf{t})$ depends on $r_\gamma(\mathbf{t}|\mathbf{z})$. See Appendix B.1 for additional analysis of Eq. 3.

Updating Phase The term $\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z}) = \mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z};\boldsymbol{l})}{p(\mathbf{t}|\mathbf{z})}$ makes $\mathbf{z}$ informative about $\mathbf{t}$. We approximate with a surrogate objective $\mathcal{L}_{\mathcal{I}}$:

$$\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z}) \leq \mathbb{E}_{p(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})}{r_\gamma(\mathbf{t}|\mathbf{z})} = \mathcal{L}_{\mathcal{I}}. \tag{4}$$

This surrogate objective encourages the proposed distribution $r_\gamma(\mathbf{t}|\mathbf{z})$ to be updated to be close to the revised distribution $p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z};\boldsymbol{l})$. After training, the proposed distribution $r_\gamma(\mathbf{t}|\mathbf{z})$ plays the role of $p_\gamma(\mathbf{t}|\mathbf{z},\mathbf{x})$ and does not need to explicitly use $p_{\mathcal{D}}(\mathbf{t}|\mathbf{x})$. Throughout, we refer to $\mathcal{L}_{\mathcal{I}}$ as *updating*. The updating objective for our model is given by

$$\mathcal{L}_{\mathcal{I}} = \sum_{m=1}^{M} \text{KL}\big(p_\gamma(t_m|x_m, z_m; l_m) \parallel r_\gamma(t_m|z_m)\big), \tag{5}$$

where by expanding further we obtain

$$\mathcal{L}_{\mathcal{I}} = \sum_{m=1}^{M} \Big[-H(t_m|x_m, z_m) + \hat{\lambda}_m H(t_m|z_m) - (1-\hat{\lambda}_m)\mathbb{E}_{p_{\mathcal{D}}(t_m|x_m)} \log r_\gamma(t_m|z_m)\Big]. \tag{6}$$

This sum is computed over observed nodes. For unsupervised nodes, the revised distribution and proposed distribution are equal and their KL terms are zero. The last term is a classification term.

Regularization To improve the model generalization ability, we introduce regularization terms for $\mathbf{z}$ and $\mathbf{t}$. We constrain the mutual information between data $\mathbf{x}$ and $\mathbf{z}$ latent representations, $\mathcal{I}(\mathbf{x};\mathbf{z}) = \mathbb{E}_{p(\mathbf{z},\mathbf{x})} \log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}$ as

$$\mathcal{I}(\mathbf{x};\mathbf{z}) \leq \underbrace{\mathrm{KL}\big(p_\theta(\mathbf{z}|\mathbf{x}) \parallel q(\mathbf{z})\big)}_{\mathcal{L}_z}, \tag{7}$$

where we introduce a variational distribution $q(\mathbf{z})$ due to intractability of $p(\mathbf{z})$. For simplicity we assume that $q(\mathbf{z})$ factorizes over independent Gaussian random variables as $q(\mathbf{z}) = \prod_{m=1}^{M} q(z_m)$, where the variational distribution over z_m is given by the unit Gaussian $z_m \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Here $\mathcal{L}_z$ is estimated with the standard Monte-Carlo sampling:

$$\mathcal{L}_z \approx \frac{1}{S} \sum_{m=1}^{M} \sum_{s=1}^{S} \mathrm{KL}\big(p_\theta(z_m|t_{m-1}^{(s)}, \mathbf{x}) \parallel q(z_m)\big). \tag{8}$$

We reduce the distance between the proposed distribution $r_\gamma(\mathbf{t}|\mathbf{z})$ and a uniform distribution $\mathcal{U}(\mathbf{t})$:

$$\mathcal{L}_t = \mathbb{E}_{p_\mathcal{D}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})r_\gamma(\mathbf{t}|\mathbf{z})} \log \frac{r_\gamma(\mathbf{t}|\mathbf{z})}{\mathcal{U}(\mathbf{t})}. \tag{9}$$

The Kullback-Leibler (KL) divergence terms in Eq. 7 and Eq. 9 help avoid overfitting. An alternative interpretation for these regularization terms is discarding the task-irrelevant information. We combine Eqs. 3, 4, 7 and 9 to at our final objective

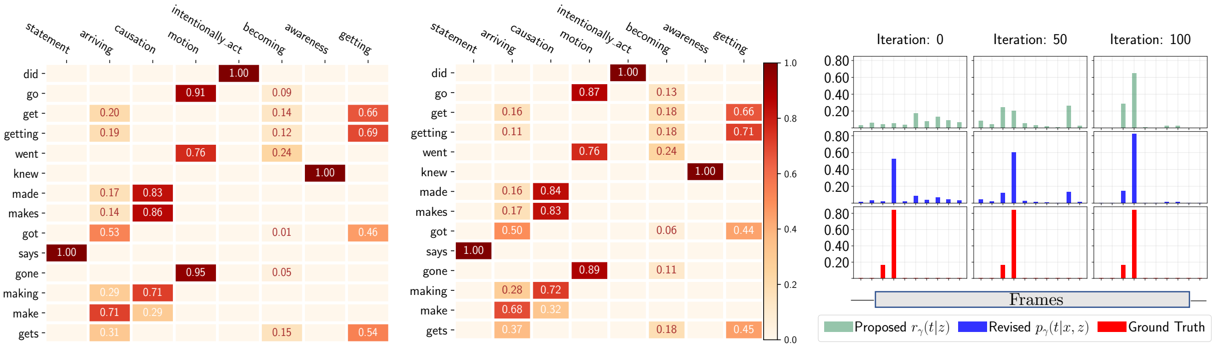
$$\mathcal{L}^U = \mathcal{L}_y + \alpha\mathcal{L}_\mathcal{I} + \beta\mathcal{L}_z + \zeta\mathcal{L}_t, \tag{10}$$

where β and ζ denote the trade-off parameters, and can be set empirically, as described in Appendix I.

3.4 Architecture For Event Modeling

To ensure fair comparisons, we focus on the reconstruction task similar to a β-VAE framework (Higgins et al., 2016). The overall structure is depicted in Fig. 2. Following previous work on event modeling (Pichotta and Mooney, 2016; Rezaee and Ferraro, 2021; Weber et al., 2018b; Gao et al., 2022), we represent each document $\mathbf{x}$ as a sequence of M events, where each event is a 4 tuple of predicate (verb), two main arguments (subject and object), and a modifier (if applicable). Each event is associated with a discrete semantic frame. E.g., *convicted man murdering of* is an event and the semantic frame is [VERDICT]. All the possible frames are collected in a vocabulary of size T. In this setting, we obtain a point estimate for $p_\mathcal{D}(t_m|x_m)$ as $\delta(x_m, t_m)$. Sampling from this empirical distribution outputs a one-hot vector of dimension T. The proposed distribution $r_\gamma(t|z)$ is a Gumbel-Softmax distribution. We found that the Gumbel-Softmax algorithm is particularly suitable for our task because it can effectively approximate discrete distributions and backpropagate gradients. While experiments with the Straight-Through Gumbel-Softmax (STGT) yielded near identical performance to the Gumbel-Softmax method, we opted for the latter. STGT generates one-hot vectors during the forward pass, but requires an approximation of the gradient using Gumbel-Softmax samples. By setting a low temperature of 0.5, the generated Gumbel-Softmax samples become almost identical to one-hot vectors, eliminating the need for gradient approximation. The effects on the gradient were carefully analyzed and the results are presented in Damadi and Shen (2022), where a comprehensive study of gradient properties was conducted. To learn richer representations, we define an embedding matrix $\mathbf{E} \in \mathbb{R}^{T \times d_t}$, to convert a simplex frame into a vector representation as $e_m = t_m^T\mathbf{E}$.

Encoding and Decoding With data point $\mathbf{x} \sim p_\mathcal{D}(\mathbf{x})$, we encode the whole sequence into recurrent hidden representations $\mathbf{H} = \{h\}_{t=1}^{T}$. For each event m, we draw Gaussian random variable $z_m \sim \mathcal{N}(\mu_m, \sigma_m)$ where μ_m and σ_m are the outputs of attention layers over frame embedding e_{m-1} and hidden states $\mathbf{H}$. We use a linear mapping over z_m to compute Gumbel softmax parameters for the proposed distribution $r_\gamma(t_m|z_m)$. Given the proposed distribution $r_\gamma(t_m|z_m)$ and the empirical distribution $p_\mathcal{D}(t_m|x_m)$, we first draw a Bernoulli sample $\hat{\lambda}_m$, then we draw knowledge sample from the mixture of probabilities by $t_m \sim p_\gamma(t_m|x_m, z_m; l_m)$. From the encoding phase, all the embedding vectors are gathered into $\mathbf{e} = \{e_m\}_{m=1}^{M}$. At generation time, analogously to the encoder, we use an attention layer over the decoder recurrent hidden state h and frame embeddings $\mathbf{e}$, resulting in decoder logits g. The generative distribution over possible next tokens is given by $x_t \sim p(x_t|g)$.

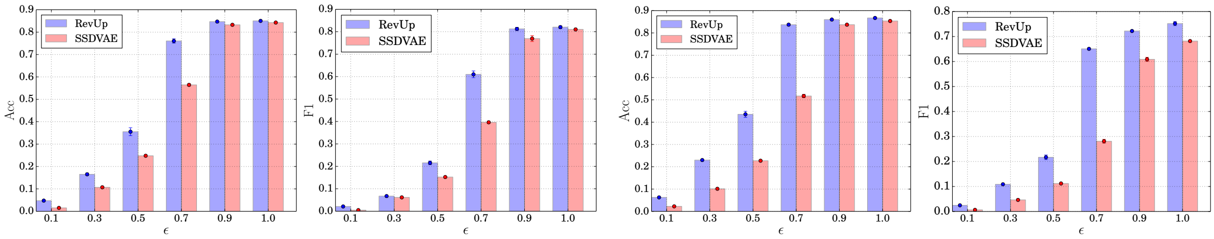


(a) Normalized samples from learned $r_\gamma(t|z)$ given predicates.

(b) Ground truth conditional distribution of semantic frames given predicates.

(c) The proposed and revised distributions converge to the ground truth distribution.

Figure 3: A demonstration of RevUp working on a small dataset with 10 frames. Ground truth frame distributions are shown in Fig. 3b, where in Fig. 3a we show that the distribution $r_\gamma(t|z)$ that RevUp learns very closely matches the ground truth. In Fig. 3c, we show how RevUp affects both the proposed and revised distributions. Initially, the $r_\gamma(t|z)$ prediction is random. After 50 iterations it is closer to the revised distribution $p_\gamma(t|z,x)$. After 100 iterations of training, both of them get close to the ground truth distribution.



(a) NYT Accuracy

(b) NYT F1 score

(c) Wikipedia Accuracy

(d) Wikipedia F1 score

Figure 4: RevUp vs SSDVAE accuracy for sequential semantic frame classifications for NYT and Wikipedia dataset. See Appendix H for full results. For all degree of observation probabilities, RevUp outperforms the baseline.

4 Experimental Results

Following previous work (Rezaee and Ferraro, 2021), we experiment on event modeling tasks using Concretely Annotated versions of New York Times Gigaword (NYT) and Wikipedia datasets (Ferraro et al., 2014). Both are English and have FrameNet semantic frames provided via SemaFor (Das et al., 2014). We perform a direct comparison with the previous state-of-the-art work (Rezaee and Ferraro, 2021) by using the same frame types, which were derived from the FrameNet annotations on Wikipedia articles. Training, validation and test splits for NYT and Wikipedia have 320k/17k/7k and 240k/8k/11k documents, respectively. For both the validation and test phases we set $l_m = 0$ (unsupervised). We average results over three runs (standard deviations in the appendix). We tuned the hyperparameters on the validation dataset. See Appendix I for additional implementation and data details.

4.1 Small Dataset Example

We first examine RevUp by examining its behavior on a small, focused example dataset. We sampled 400 newswire documents from the NYT dataset. We trained a RevUp model, with 10 semantic frame types (10 options for each t) and where each z was 100 dimensional. To obtain the ground-truth distributions of frames given events, we just focus on the predicates and collect all the semantic frames given that specific predicate. We carefully selected the frame types and data to reflect a diverse range of difficulties. For example, events including the *did* predicate are always associated with the INTENTIONALLY_ACT frame and the corresponding semantic frames for the *gets* predicate are ARRIVING, BECOMING and GETTING. RevUp predictions are acquired from the revised distribution $r_\gamma(t|z)$: given an event x we first draw a Gaussian sample z and then use $\text{argmax}(t)$ to find the proposed index. Finally, we normalize across all the proposed frames. We visualize the revision and update steps for the predicate "*made*" in Fig. 3c.

As evidenced by Eq. 4, the minimization of the KL-divergence between the proposed distribution $r_\gamma(t|z)$ and the revised distribution $p_\gamma(t|x,z)$ results in the elimination of non-essential information. Fig. 3c gives a visual representation of this phenomenon. At iteration 0, the proposed distribution $p_\gamma(t|x,z)$ is almost uniform, while the revised distribution $r_\gamma(t|z)$ is more aligned with the ground distribution. With additional iterations, reaching 100, the gap between these distributions reduces, ultimately leading to convergence with the ground truth for both distributions.

Additionally, our framework is able to effectively capture the conditional distribution of frames given predicates in diverse scenarios, as demonstrated in the heatmap figures in Figs. 3a and 3b. For instance, certain predicates such as “did,” “says,” and “knew” have a single associated frame of “intentionally_act,” “awareness,” and “statement,” respectively. Meanwhile, the predicate “made” has two possible semantic frames of “arriving” and “causation.” In some cases, such as the predicate “get,” there are even three possible frames of “arriving,” “becoming,” and “getting.” The comparison between the ground truth distribution of frames given predicates and the normalized samples highlights the accuracy of our model in capturing these conditional distributions, with minimal error.

4.2 Baselines

We compare the proposed RevUp method with the following event modeling approaches. (a) **RNNLM** (Pichotta and Mooney, 2016): A Bidirectional GRU cell with two layers, hidden dimension of 512, gradient clipping at 5 and Glove 300 embeddings to represent words. We used the implementation provided in (Weber et al., 2018b). (b) **HAQAE** (Weber et al., 2018b): An unsupervised VAE-based method for script learning and generation. They use vector quantization (VQ–VAE) to define a hierarchical discrete latent space. (c) **SSDVAE** (Rezaee and Ferraro, 2021) A semi–supervised VAE-based model. They utilize Gumbel softmax and use the parameter injection to incorporate the side knowledge. The architecture of our model largely follows the SSDVAE and HAQAE models with minor modification; see Appendix I.

4.3 Effect of Noisy Knowledge

We empirically compare the predictive performance of RevUp and SSDVAE with noisy knowledge. To do so, in a fully supervised setting, for each event x_m, instead of using the associated semantic frame t_m, with probability η we replace t_m with a random semantic frame $\hat{t}_m$. We train both models with this new training dataset. During the testing phase, we compare the predicted knowledge with ground-truth one. Results in Table 1 show that classification and parameter injection in SSDVAE are not enough for capturing knowledge. These results validate the effectiveness of our information capturing strategy when the external knowledge is noisy. As we increase η, SSDVAE performance degrades much more than RevUp. As an instance when $\eta = 0.9$, SSDVAE's accuracy is almost zero but our proposed model can achieve 0.41.

model	η	Wiki		NYT	
		Acc	F1	Acc	F1
SSDVAE	0.1	0.77	0.43	0.76	0.55
RevUp		**0.85**	**0.58**	**0.83**	**0.71**
SSDVAE	0.2	0.69	0.35	0.67	0.44
RevUp		**0.82**	**0.52**	**0.80**	**0.63**
SSDVAE	0.3	0.60	0.28	0.58	0.36
RevUp		**0.79**	**0.45**	**0.77**	**0.58**
SSDVAE	0.5	0.41	0.17	0.41	0.23
RevUp		**0.72**	**0.36**	**0.71**	**0.48**
SSDVAE	0.7	0.23	0.09	0.22	0.11
RevUp		**0.64**	**0.29**	**0.63**	**0.37**
SSDVAE	0.9	0.02	0.00	0.02	0.00
RevUp		**0.41**	**0.09**	**0.25**	**0.06**

Table 1: Effect of Noise on robustness. We present standard deviations in Appendix H.

model	ϵ	Wiki	NYT
RNNLM	-	56.96	64.57
HAQAE	-	39.47	50.10
SSDVAE	0.0	39.75	47.50
RevUp		**39.48**	**45.36**
SSDVAE	0.1	39.73	45.91
RevUp		**33.34**	**44.87**
SSDVAE	0.7	36.79	44.79
RevUp		**33.20**	**41.74**
SSDVAE	1.0	30.69	36.96
RevUp		**28.33**	**34.85**

Table 2: Test perplexity results, varying the percent ϵ of side knowledge that was observed during training. We present standard deviations in Appendix H (table 4).

4.4 Effect of Incomplete Knowledge

We consider the case of semi-supervised learning, where with probability ϵ a node t_m is observed. We report the classifications on the latent nodes as $\mathrm{KL}\big(p_{\mathcal{D}}(\mathbf{t}|\mathbf{x}) \;\|\; r_\gamma(\mathbf{t}|\mathbf{x})\big) \approx -\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})p_{\mathcal{D}}(\mathbf{t}|\mathbf{x})} \log r_\gamma(\mathbf{t}|\mathbf{z})$. The results are shown in Fig. 4. We report two widely-used classification metrics including accuracy and F1 to evaluate the performance of all methods. SSDVAE just relies on the guidance and classification but RevUp also uses the updating phase to shift the available knowledge from side information into latent variables. Thus the results demonstrate the superiorities of RevUp to predict knowledge when they are partially observed, attributable to its novel information injection and learning.

To investigate whether the proposed approach works for task representation, we compare the perplexity of our approach to prior work in Table 2. Perplexity has been commonly used in the literature, which allows us to provide a fair comparison with previous efforts. We investigate the effect of supervision with various observation probabilities $\epsilon = 0.0$ (unsupervised), 0.1, 0.7 and 1.0 (fully supervised) on the generated samples from the model. We see that our model is able to obtain lower perplexity scores than the previous event modeling methods. We observe as ϵ increases, the performance of our proposed model improves. For each observation probability, our method outperforms SSDVAE. The results demonstrate that our method achieves state-of-the-art performance with large margins from the baselines.

5 Related Work

Information Bottleneck The concept of using side information for discrete source coding was explored by Wyner (1975); Wyner and Ziv (1976). The Information Bottleneck (IB) principle was then introduced by (Tishby et al., 2000) to compress input variables while predicting a target. Chechik and Tishby (2002) proposed incorporating negative side information through an auxiliary loss in a supervised manner. Our method stands apart by handling both supervised and semi-supervised settings with ease. The Variational Information Bottleneck (VIB) was introduced by Alemi et al. (2016), which improved the IB estimation through amortized variational methods. Voloshynovskiy et al. (2020) extended the VIB method for semi-supervised classification. In relation to our work, there have been numerous studies focused on maximizing the mutual information between different views and discarding the non-shared information (Federici et al., 2019; Wan et al., 2021; Wang et al., 2019; Mao et al., 2021; Yan et al., 2019).

Variational Autoencoders (VAEs) Kingma and Welling (2013) and Rezende et al. (2014) introduced the reparameterization technique for variational inference. Recently, Huang and Xiao (2021) applied adversarial training to enhance the ability of VAE in processing sequential data, while Qiu et al. (2020) explored the possibility of VAE handling multi-view temporal data. Our focus is limited to objectives that can be expressed with discrete latent variables.

Jang et al. (2016) and Maddison et al. (2016) independently presented the Gumbel-Softmax distribution, which allows backpropagation through latent discrete variables in variational autoencoders. For more information, refer to the review by Huijben et al. (2022). Van Den Oord et al. (2017) proposed VQ-VAE, which uses vector quantization to represent discrete values with embeddings.

Kingma et al. (2014) proposed a VAE for semi-supervised classification, where labels in unsupervised settings are treated as latent variables and predicted when available. Joy et al. (2022) later improved this approach by separating latent variables to differentiate label characteristics from values. Lo and Lim (2020) modified the VAE objective to incorporate external knowledge with embedding vectors, and Feng et al. (2020) proposed a new ELBO to incorporate label prediction loss.

Event Modeling In recent years, much research has been dedicated to the challenge of modeling the sequence of events (Weber et al., 2018b; Rezaee and Ferraro, 2021; Gao et al., 2022). One such contribution was made by Weber et al. (2018a), who introduced a tensor-based composition to effectively capture semantic event relations. Gao et al. (2022) proposed a self-supervised contrastive learning approach based on co-occurring events. Another approach suggested by Weber et al. (2018b) involves the use of Recurrent Neural Networks (RNNs) to model the hierarchical latent structures that exist in sequences of events. This methodology was further developed in a subsequent study by Rezaee and Ferraro (2021), where external knowledge was incorporated into the latent layer for enhanced modeling, which is highly relevant to our proposed approach.

6 Conclusion

We show how to incorporate noisy partially observed side knowledge source along with latent variables. To do so, we generalized the main idea of parameter injection and maximizing the mutual information between external knowledge and latent variables. Our experiments show that our approach is more robust to noisy knowledge and outperform other baselines for the event modeling task

Acknowledgments

We would like to thank the anonymous reviewers for their comments, questions, and suggestions.

This material is based in part upon work supported by the National Science Foundation under Grant No. IIS-2024878. Some experiments were conducted on the UMBC HPCF, supported by the National Science Foundation under Grant No. CNS-1920079. This material is also based on research that is in part supported by the Army Research Laboratory, Grant No. W911NF2120076, and by the Air Force Research Laboratory (AFRL), DARPA, for the KAIROS program under agreement number FA8750-19-2-1003. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of the Air Force Research Laboratory (AFRL), DARPA, or the U.S. Government.

7 Limitations

Some limitations of our work are that multiple hyper parameters should be tuned for each task. While we provide guidance and insights into effective settings (and some intuition as to why), we acknowledge that the settings may be domain dependent.

Because we use semantic frames as our side knowledge, our focus is on improving the representation and use of discrete latent variables. While current NLP approaches have often focused on text-to-text methods for input and output, and individual words in text can be considered a form of discrete latent variables, we note that these methods are driven by large continuous embedding methods. While we believe this work can be extended to continuous cases, the approximations did make use of aspects of discrete variables, and they would need to be re-derived.

RevUp depends on mixing in statistics and learned representations from external side knowledge. While we envision this side knowledge as containing useful generalizations and semantic information, such resources could encode overly broad generalizations or other biases. While the degree of this mixture can be adjusted, imperfections or biases in the external knowledge could be captured and propagated through RevUp.

We focus on the task of event modeling, but we believe RevUp represents a step towards improving settings where noisy side information is available.

References

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*.

Sarkhan Badirli, Zeynep Akata, George Mohler, Christine Picard, and Mehmet M Dundar. 2021. Fine-grained zero-shot learning with dna as side information. *Advances in Neural Information Processing Systems*, 34:19352–19362.

Gal Chechik and Naftali Tishby. 2002. Extracting relevant structures with side information. *Advances in Neural Information Processing Systems*, 15.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2019. Variational sequential labelers for semi-supervised learning. *arXiv preprint arXiv:1906.09535*.

Muhao Chen, Hongming Zhang, Haoyu Wang, and Dan Roth. 2020. What are you trying to do? semantic typing of event processes. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 531–542.

Thomas M Cover. 1999. *Elements of information theory*. John Wiley & Sons.

Saeed Damadi and Jinglai Shen. 2022. Gradient properties of hard thresholding operator. *arXiv preprint arXiv:2209.08247*.

Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. 2014. Frame-semantic parsing. *Computational linguistics*, 40(1):9–56.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. 2019. Learning robust representations via multi-view information bottleneck. In *International Conference on Learning Representations*.

Hao-Zhe Feng, Kezhi Kong, Minghao Chen, Tianye Zhang, Minfeng Zhu, and Wei Chen. 2020. Shot-vae: semi-supervised deep generative models with label-aware elbo approximations. *CoRR abs/2011.10684. URL https://arxiv. org/abs/2011*, 10684.

Francis Ferraro, Max Thomas, Matthew R Gormley, Travis Wolfe, Craig Harman, and Benjamin Van Durme. 2014. Concretely annotated corpora. In *4th Workshop on Automated Knowledge Base Construction (AKBC)*.

Francis Ferraro and Benjamin Van Durme. 2016. A Unified Bayesian Model of Scripts, Frames and Language. In *Proceedings of the 30th Conference on Artificial Intelligence (AAAI)*, pages 2601–2607, Phoenix, Arizona. Association for the Advancement of Artificial Intelligence.

Arthur Flajolet and Patrick Jaillet. 2017. Real-time bidding with side information. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jun Gao, Wei Wang, Changlong Yu, Huan Zhao, Wilfred Ng, and Ruifeng Xu. 2022. Improving event representation via simultaneous weakly supervised contrastive learning and clustering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3036–3049, Dublin, Ireland. Association for Computational Linguistics.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework.

Jin Huang and Ming Xiao. 2021. Regularized sequential latent variable models with adversarial neural networks. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 834–839. IEEE.

Iris AM Huijben, Wouter Kool, Max Benedikt Paulus, and Ruud JG Van Sloun. 2022. A review of the gumbel-max trick and its extensions for discrete stochasticity in machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Tom Joy, Yuge Shi, Philip Torr, Tom Rainforth, Sebastian M Schmon, and Siddharth N. 2022. Learning multimodal VAEs through mutual supervision. In *International Conference on Learning Representations*.

Alex Judea and Michael Strube. 2015. Event extraction as frame-semantic parsing. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 159–164, Denver, Colorado. Association for Computational Linguistics.

Di Kang, Debarun Dhar, and Antoni Chan. 2017. Incorporating side information by adaptive convolution. *Advances in Neural Information Processing Systems*, 30.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, 27.

Andreas Kirsch, Clare Lyle, and Yarin Gal. 2020. Unpacking information bottlenecks: Surrogate objectives for deep learning.

Lingpeng Kong, Cyprien de Masson d'Autume, Lei Yu, Wang Ling, Zihang Dai, and Dani Yogatama. 2019. A mutual information maximization perspective of language representation learning. In *International Conference on Learning Representations*.

R. Linsker. 1988. Self-organization in a perceptual network. *Computer*, 21(3):105–117.

Pei-Chi Lo and Ee-Peng Lim. 2020. Co-embedding attributed networks with external knowledge.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*.

Yiqiao Mao, Xiaoqiang Yan, Qiang Guo, and Yangdong Ye. 2021. Deep mutual information maximin for cross-modal clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8893–8901.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Ankur Padia, Francis Ferraro, and Tim Finin. 2018. SURFACE: Semantically rich fact validation with explanations. ArXiv:1810.13223.

Karl Pichotta and Raymond Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Lin Qiu, Vernon M Chinchilli, and Lin Lin. 2020. Interpretable deep representation learning from temporal multi-view data. *arXiv preprint arXiv:2005.05210*.

Mehdi Rezaee and Francis Ferraro. 2021. Event representation with sequential, semi-supervised discrete variables. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4701–4716.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 1999. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 2000. The information bottleneck method.

Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Slava Voloshynovskiy, Olga Taran, Mouad Kondah, Taras Holotyak, and Danilo Rezende. 2020. Variational information bottleneck for semi-supervised classification. *Entropy*, 22(9):943.

Zhibin Wan, Changqing Zhang, Pengfei Zhu, and Qinghua Hu. 2021. Multi-view information-bottleneck representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10085–10092.

Qi Wang, Claire Boudreau, Qixing Luo, Pang-Ning Tan, and Jiayu Zhou. 2019. Deep multi-view information bottleneck. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 37–45. SIAM.

Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. 2018a. Event representations with tensor-based compositions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nathanael Chambers. 2018b. Hierarchical quantized representations for script generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3783–3792.

A. Wyner. 1975. On source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 21(3):294–300.

A. Wyner and J. Ziv. 1976. The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 22(1):1–10.

Patrick Xia, Guanghui Qin, Siddharth Vashishtha, Yunmo Chen, Tongfei Chen, Chandler May, Craig Harman, Kyle Rawlins, Aaron Steven White, and Benjamin Van Durme. 2021. LOME: Large ontology multilingual extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*.

Xiaoqiang Yan, Yangdong Ye, Yiqiao Mao, and Hui Yu. 2019. Shared-private information bottleneck method for cross-modal clustering. *IEEE Access*, 7:36045–36056.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Qi Zhang, Wei Lin, and Antoni B Chan. 2021. Cross-view cross-scene multi-view crowd counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 557–567.

A Information Bottleneck Principle (IB)

Given its importance to our effort, we restate the information bottleneck principle, and discuss how it ties in with our loss formulation. Then, in subsequent appendices (App. B to D) we step through individual subloss terms.

Given the original input data $\mathbf{x}$ with the target $\mathbf{y}$, IB aims to extract a compact representation of $\mathbf{x}$ that is most informative about the target $\mathbf{y}$ via maximizing $\mathcal{I}(y;z) - \beta\mathcal{I}(x;z)$. The first term $\mathcal{I}(y;z)$ motivates the model to predict y, whilst the second term $\mathcal{I}(x;z)$ aims at discarding irrelevant information from the input x. The hyperparameter β can be set or tuned to adjust the relative importance of discarding irrelevant information. In our framework, we have a generalized version of IB as follows

$$\begin{aligned}\mathcal{L} = &-\mathcal{I}(\mathbf{y};\mathbf{t}) + \alpha\mathcal{I}(\mathbf{x};\mathbf{t}|\mathbf{z}) \\ &+ \beta\mathcal{I}(\mathbf{x};\mathbf{z}) + \zeta\mathrm{KL}\big(r_\gamma(\mathbf{t}|\mathbf{z}) \parallel \mathcal{U}(\mathbf{t})\big). \quad (11)\end{aligned}$$

Our generalized version accounts for additional random variables, and the associated dependencies. Here, $r_\gamma(\mathbf{t}|\mathbf{z})$ is the proposed distribution, $\mathcal{U}$ is a uniform distribution over $\mathbf{t}$. The KL term helps avoid overfitting.

B Regularization and Task Representation Derivation

Throughout we use the following inequality

$$\begin{aligned}&\mathbb{E}_{p(\mathbf{z},\mathbf{x})} \log \frac{q(\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})} \le \\ &\qquad \mathbb{E}_{p_\mathcal{D}(\mathbf{x})} \log \underbrace{\mathbb{E}_{p_\theta}(\mathbf{z}|\mathbf{x}) \frac{q(\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})}}_{1} = 0. \end{aligned} \quad (12)$$

The regularization term $\mathcal{I}(\mathbf{x};\mathbf{z})$ is

$$\begin{aligned}\mathcal{I}(\mathbf{x};\mathbf{z}) &= \mathbb{E}_{p(\mathbf{z},\mathbf{x})} \log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})} \\ &\le \mathbb{E}_{p(\mathbf{z},\mathbf{x})} \log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{q(\mathbf{z})} \end{aligned} \quad (13)$$

where since $p(\mathbf{z}) = \int p_\theta(\mathbf{z}|\mathbf{x}) p_\mathcal{D}(\mathbf{x})\, dx$ is intractable we use the inequality in Eq. 12. Now, we can approximate Eq. 13 as follows

$$\begin{aligned}&\mathbb{E}_{p_\mathcal{D}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})} \log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})} \quad (14) \\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})} \log \frac{\prod_{m=1}^{M} p_\theta(z_m|t_{m-1},\mathbf{x})}{\prod_{m=1}^{M} q(z_m)} \\ &= \sum_{m=1}^{M} \mathbb{E}_{p_\mathcal{D}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})} \log \frac{p_\theta(z_m|t_{m-1},\mathbf{x})}{q(z_m)} \\ &\approx \frac{1}{S} \sum_{m=1}^{M} \sum_{s=1}^{S} \mathrm{KL}\big(p_\theta(z_m|t_{m-1}^{(s)},\mathbf{x}) \parallel q(z_m)\big), \end{aligned} \quad (15)$$

where for simplicity we assume that $q(\mathbf{z})$ factorizes over independent random variables as $q(\mathbf{z}) = q(z_1, z_2, \ldots, z_M) = \prod_{m=1}^{M} q(z_m)$. We parameterize each z_m via a multivariate Normal, $q(z_m) = \mathcal{N}(\mathbf{0},\mathbf{I})$, and $p_\theta(z_m|z_{m-1}^{(s)},\mathbf{x}) = \mathcal{N}(\mu_m, \Sigma_m)$:

$$\begin{aligned}\mu_m &= f^\mu(z_{m-1}^{(s)},\mathbf{x}) \\ \Sigma_m &= f^\Sigma(z_{m-1}^{(s)},\mathbf{x}),\end{aligned}$$

where both f^μ and f^Σ are neural networks. With this formulation, we can calculate the KL-divergence terms of Eq. 15 in closed forms.

B.1 Approximating $\mathcal{I}(\mathbf{y};\mathbf{t})$ (Eq. 3)

In the same way for the task representation, we have

$$\begin{aligned}-\mathcal{I}(\mathbf{y};\mathbf{t}) &= -\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log \frac{p(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})} \\ &\le -\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log \frac{q_\phi(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})} \\ &\le -\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log q_\phi(\mathbf{y}|\mathbf{t}) + H(\mathbf{y}),\end{aligned} \quad (16)$$

where $H(\mathbf{y})$ is constant and we ignore it. We have:

$$\begin{aligned}&\mathbb{E}_{p(\mathbf{y},\mathbf{z},\mathbf{t})} \log q_\phi(\mathbf{y}|\mathbf{t}) \\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x},\mathbf{y})p_\theta(\mathbf{z}|\mathbf{x})p_\gamma(\mathbf{t}|\mathbf{z})} \log q_\phi(\mathbf{y}|\mathbf{t}), \quad (17) \\ &\approx \frac{1}{S_z}\frac{1}{S_t} \sum_{s_z=1}^{S_z} \sum_{s_t=1}^{S_t} \log q_\phi(\mathbf{y}|\mathbf{t}^{(s_t)}), \quad (18)\end{aligned}$$

where we first sample $\mathbf{x}, \mathbf{y} \sim p_\mathcal{D}(\mathbf{x},\mathbf{y})$, then $\mathbf{z}^{(s_z)} \sim p_\theta(\mathbf{z}|\mathbf{x})$ and finally $\mathbf{t}^{(s_t)} \sim r_\gamma(\mathbf{t}|\mathbf{z}^{(s_z)})$. In this approximation S_z and S_t are the total number of samples for $\mathbf{z}$ and $\mathbf{t}$ respectively.

We wish to clarify why this term is included in the objective and how optimizing it affects the model parameters. The answer is that in computing $\mathcal{I}(\mathbf{y};\mathbf{t})$, the distribution used to compute the mutual information's expectation depends on the proposed distribution r_γ. This is by construction and irrespective of whether side information is present or not. This is where the model parameters come in.

Specifically, while by definition $\mathcal{I}(\mathbf{y};\mathbf{t}) = \mathbb{E}_{\mathbf{p(y,t)}} \frac{\mathbf{logp(y|t)}}{\mathbf{p(y)}}$, because $p(|\mathbf{t})$ does not depend on a (free) $\mathbf{z}$, we can compute $mut(\mathbf{y};\mathbf{t}) = \mathbb{E}_{\mathbf{p(y,z,t)}} \frac{\mathbf{logp(y|t)}}{\mathbf{p(y)}}$. This follows from the fact that for random variables u and v, $\mathbb{E}_{p(u,v)}[f(u)] = \mathbb{E}_{p(u)}[f(u)]$. With this, we can then write

$$
\begin{aligned}
\mathcal{I}(\mathbf{y};\mathbf{t}) &= \mathbb{E}_{p(\mathbf{y},\mathbf{t})} \log \frac{p(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})} \\
&= \mathbb{E}_{p(\mathbf{y},\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})} && (19) \\
&= \mathbb{E}_{\left\{p_{\mathcal{D}}(\mathbf{x},\mathbf{y})p(\mathbf{z},\mathbf{t}|\mathbf{x})\right\}} \log \frac{p(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})} && (20) \\
&= \mathbb{E}_{\left\{p_{\mathcal{D}}(\mathbf{x},\mathbf{y}) \prod_{m=1}^{M} p_\theta(z_m|t_{m-1},\mathbf{x}) {\color{red}p_\gamma(t_m|x_m,z_m;l_m)}\right\}} && (21) \\
&\log \frac{p(\mathbf{y}|\mathbf{t})}{p(\mathbf{y})}. && (22)
\end{aligned}
$$

This red term, $p_\gamma(t_m|x_m, z_m; l_m)$, is defined as the interpolation of a proposal distribution $r_\gamma(t_m|z_m)$ and an empirical distribution $t_m|x_m$. Defined as a GumbelSoftmax, this proposal distribution explicitly allows uncertainty. Therefore, even when side information is present, we sample the value of t_m to approximate that final expectation.

Finally, as mentioned in the paper, this last line is intractable, so we learn an approximation $q_\phi(y|t)$, which introduces additional model parameters to learn. We compute q_ϕ using the sampled values $t_1 ... t_M$.

C Update Step Upperbound

In this section, we show how we approximate the update term $\mathcal{I}(\mathbf{x};\mathbf{z}|\mathbf{z})$ in Eq. 4. We have the following

$$
\begin{aligned}
\mathcal{I}(\mathbf{t};\mathbf{x}|\mathbf{z}) &= \mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{z},\mathbf{x})} \log \frac{p_\gamma(\mathbf{t},\mathbf{x}|\mathbf{z})}{p(\mathbf{t}|\mathbf{z})p(\mathbf{x}|\mathbf{z})} \\
&= \mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})p(\mathbf{x}|\mathbf{z})}{p(\mathbf{t}|\mathbf{z})p(\mathbf{x}|\mathbf{z})} \\
&= \mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})}{p(\mathbf{t}|\mathbf{z})} && (23) \\
&\leq \mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})}{r_\gamma(\mathbf{t}|\mathbf{z})}, && (24)
\end{aligned}
$$

First, we should note that in Eq. 23, the distribution $p(\mathbf{t}|\mathbf{z})$ is not equal to $r_\gamma(\mathbf{t}|\mathbf{z})$, because for each node m, we have

$$
\begin{aligned}
p(t_m|z_m) &= \int p(t_m, x_m|z_m)\, dx_m \\
&= \int p(t_m|x_m, z_m)p(x_m|z_m)\, dx_m \\
&= \int \hat{\lambda}_m r_\gamma(t_m|z_m)p(x_m|z_m)\, dx_m + \\
&\int (1-\hat{\lambda}_m)p_{\mathcal{D}}(t_m|x_m)p(x_m|z_m)\, dx_m \\
&\neq r_\gamma(t_m|z_m). && (25)
\end{aligned}
$$

Our the last step of estimation in Eq. 24 is

$$
\begin{aligned}
&\mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})}{p(\mathbf{t}|\mathbf{z})} \leq \\
&\mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})}{r_\gamma(\mathbf{t}|\mathbf{z})}. && (26)
\end{aligned}
$$

The proof for Eq. 26 is as follows.

$$
\begin{aligned}
&\mathbb{E}_{p(\mathbf{t},\mathbf{x},\mathbf{z})} \log \frac{r_\gamma(\mathbf{t}|\mathbf{z})}{p(\mathbf{t}|\mathbf{z})} \\
&= \mathbb{E}_{p(\mathbf{t},\mathbf{z})} \log \frac{r_\gamma(\mathbf{t}|\mathbf{z})}{p(\mathbf{t}|\mathbf{z})} \\
&= \mathbb{E}_{p(\mathbf{z})}\mathbb{E}_{p(\mathbf{t}|\mathbf{z})} \log \frac{r_\gamma(\mathbf{t}|\mathbf{z})}{p(\mathbf{t}|\mathbf{z})} \\
&\leq \mathbb{E}_{p(\mathbf{z})} \log \Big\{ \underbrace{\mathbb{E}_{p(\mathbf{t}|\mathbf{z})} \frac{r_\gamma(\mathbf{t}|\mathbf{z})}{p(\mathbf{t}|\mathbf{z})}}_{1} \Big\} \\
&= 0
\end{aligned}
$$

D Update State Approximation for Discrete Case

We show how we approximate the upperbound presented in Eq. 24.

$$
\begin{aligned}
&\mathbb{E}_{p_\gamma(\mathbf{t},\mathbf{z},\mathbf{x})} \log \frac{p_\gamma(\mathbf{t}|\mathbf{z},\mathbf{x})}{r_\gamma(\mathbf{t}|\mathbf{z})} \\
&= \underbrace{\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})p_\gamma(\mathbf{t}|\mathbf{z},\mathbf{x})} \log p_\gamma(\mathbf{t}|\mathbf{z},\mathbf{x})}_{\textcircled{\scriptsize A$_1$}} \\
&- \underbrace{\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})p_\theta(\mathbf{z}|\mathbf{x})p_\gamma(\mathbf{t}|\mathbf{z},\mathbf{x})} \log r_\gamma(\mathbf{t}|\mathbf{z})}_{\textcircled{\scriptsize A$_2$}}. && (27)
\end{aligned}
$$

Here $\textcircled{A_1}$ is the negative entropy of the revised distribution $p_\gamma(\mathbf{t}|\mathbf{x},\mathbf{z})$:

$$\textcircled{A_1} = -H(\mathbf{t}|\mathbf{x},\mathbf{z}) \tag{28}$$

$$= -\sum_{m=1} H(t_m|x_m,z_m). \tag{29}$$

For the second term we have:

$$\begin{aligned}\textcircled{A_2} &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\Big[\hat{\lambda}\mathbb{E}_{r_\gamma(\mathbf{t}|\mathbf{z})}+ \\ &\qquad (1-\hat{\lambda})\mathbb{E}_{p_\mathcal{D}(\mathbf{t}|\mathbf{x})}\Big]\log r_\gamma(\mathbf{t}|\mathbf{z}) \\ &= \hat{\lambda}\,\underbrace{\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\mathbb{E}_{p_\theta(\mathbf{z}|\mathbf{x})}\mathbb{E}_{r_\gamma(\mathbf{t}|\mathbf{z})}\log r_\gamma(\mathbf{t}|\mathbf{z})}_{\textcircled{A_{21}}} \\ &+ (1-\hat{\lambda})\,\underbrace{\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\mathbb{E}_{p_\mathcal{D}(\mathbf{t}|\mathbf{x})}\log r_\gamma(\mathbf{t}|\mathbf{z})}_{\textcircled{A_{22}}}.\end{aligned} \tag{30}$$

Similarly $\textcircled{A_{21}}$ is the entropy of the proposed distribution $r_\gamma(\mathbf{t}|\mathbf{z})$:

$$\textcircled{A_{21}} = -H(\mathbf{t}|\mathbf{z}) \tag{31}$$

$$= -\sum_{m=1}^{M} H(t_m|z_m) \tag{32}$$

Also $\textcircled{B_{22}}$ guides the model to discriminator $\mathbf{t}$ given $\mathbf{z}$ which is a classifier. To compute the entropy terms, in the discrete case (Gumbel Softmax), we use Monte Carlo sampling as follows

$$H(t_m|x_m,z_m) \approx \frac{1}{S}\sum_{n=1}^{N}\sum_{s=1}^{S}\hat{\gamma}_m^{\top(s)}\log\hat{\gamma}_m^{(s)}, \tag{33}$$

$$H(t_m|z_m) \approx \frac{1}{S}\sum_{s=1}^{S}\gamma_m^{\top(s)}\log\gamma_m^{(s)}, \tag{34}$$

where S is the number of point estimates and $\hat{\gamma}_m$ is derived from Eq. 35. Here, we have defined $\hat{\gamma}$ to be $[\hat{\lambda}\gamma_1, \hat{\lambda}\gamma_2, \ldots, \hat{\lambda}\gamma_{k^*} + (1-\hat{\lambda}), \ldots, \hat{\lambda}\gamma_K]$. The $\hat{\gamma}$ can be thought of as a mixture of the γ and empirical categorical distributions. (We use this definition of $\hat{\gamma}$ in the proof of Theorem 1 in Appendix F.) To sample from the revised distribution $p_\gamma(t_m|\mathbf{x},\mathbf{z};l_m)$ we first draw a Bernoulli sample $\hat{\lambda}_m \sim \text{Bern}(\hat{\lambda}_m; 1/(1+\lambda l_m))$ and consequently we draw knowledge sample from the mixture of probabilities by $t_m \sim \hat{\lambda}_m r_\gamma(t_m|z_m) + (1-\hat{\lambda}_m)p_\mathcal{D}(t_m|x_m)$. We implement this with a single-sample approximation.

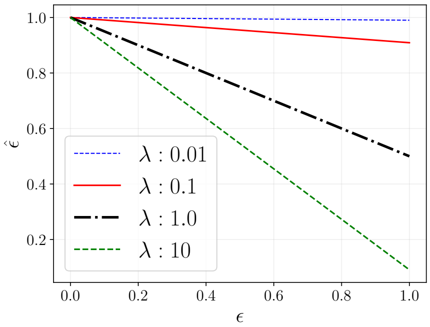


Figure 5: Effect of λ and observation probability ϵ on $\hat{\epsilon}$

E Revision Step Property of $p_\gamma\left(t_m|x_m,z_m;l_m\right)$

In this section we show how the formulation of $p_\gamma\left(t_m|x_m,z_m;l_m\right)$ leads to the property that, during the revision step, we rely more on the available side/external knowledge as more of it becomes available. Recall that we initially defined $p_\gamma\left(t_m|x_m,z_m;l_m\right) = \frac{1}{1+\lambda l_m}r_\gamma(t_m|z_m) + \frac{\lambda l_m}{1+\lambda l_m}p_\mathcal{D}(t_m|x_m)$, where λ allows a fractional portion of the empirical distribution $p_\mathcal{D}$ to influence the learned model. Suppose that $l_m \sim \text{Bern}(\epsilon)$ and $\tilde{\lambda} = 1/(1+\lambda)$, given node m we have

$$\begin{aligned}p(t_m|x_m,z_m) &= \sum_{l_m} p(t_m|x_m,z_m,l_m)p(l_m) \\ &= \epsilon p_\gamma(t_m|x_m,z_m,l_m) + (1-\epsilon)r_\gamma(t_m|z_m) \\ &= \epsilon\left[\tilde{\lambda}r_\gamma(t_m|z_m) + (1-\tilde{\lambda})p_\mathcal{D}(t_m|x_m)\right] \\ &\qquad + (1-\epsilon)r_\gamma(t_m|z_m) \\ &= \left(1-\epsilon(1-\tilde{\lambda})\right)r_\gamma(t_m|z_m) \\ &\qquad + \epsilon(1-\tilde{\lambda})p_\mathcal{D}(t_m|x_m) \\ &= \hat{\epsilon}r_\gamma(t_m|z_m) + (1-\hat{\epsilon})p_\mathcal{D}(t_m|x_m),\end{aligned}$$

where $\hat{\epsilon} = 1-\epsilon(\frac{\lambda}{1+\lambda})$. In particular, this formulation allows us to account for both our estimate of observation of side knowledge, and the extent to which we wish to use it when learning our model. In Fig. 5, we show the effect of λ and ϵ on $\hat{\epsilon}$.

F Generalization of SSDVAE (Proof of Theorem 1)

We show that under specific setting, our model reduces to the SSDVAE parameter injection mechanism.

Theorem. *If* $r_\gamma(t|z) = \mathit{Cat}(\gamma)$ *, a categorical distribution with parameters* γ*, and the empirical dis-*

tribution $p_{\mathcal{D}}(t|x)$ is a one-hot representation with $t_{k^*} = 1$, the revision step reduces to SSDVAE parameter injection.

Proof. Using the interpolation form $p_\gamma(t_m|x_m, z_m; l_m) = \hat{\lambda}_m r_\gamma(t_m|z_m) + (1 - \hat{\lambda}_m) p_{\mathcal{D}}(t_m|x_m)$ and due to the fact that the mixture of categorical distributions is a single categorical, we have $p(t|x, z; l = 1) = \mathrm{Cat}(\hat{\gamma})$, where

$$\hat{\gamma} = [\hat{\lambda}\gamma_1, \hat{\lambda}\gamma_2, \ldots, \hat{\lambda}\gamma_{k^*} + (1 - \hat{\lambda}), \ldots, \hat{\lambda}\gamma_K]. \tag{35}$$

Given $0 \leq \gamma_i \leq 1$ and $0 \leq \hat{\lambda} \leq 1$, we have $\hat{\lambda}\gamma_i \leq \gamma_i$ $(\forall i \neq k^*)$, and $\hat{\lambda}\gamma_{k^*} + (1 - \hat{\lambda}) \geq \gamma_{k^*}$. This implies that for all the indices $i \neq k^*$, the value of γ_i is reduced and γ_{k^*} is increased, confirming that parameter injection is a specific form of our definition. □

G Updating Step Maximizes Mutual Information (Proof of Theorem 2)

We show that for the reconstruction tasks, where $\mathbf{y}$ is a copy of $\mathbf{x}$, minimizing $\mathcal{I}(\mathbf{x}; \mathbf{t}|\mathbf{z})$ maximizes $\mathcal{I}(\mathbf{t}; \mathbf{z})$. While not explicitly reflected in our model definition, note that $\mathcal{I}(\mathbf{t}; \mathbf{z})$ is intractable. This can be seen in the definition of $\mathcal{I}(\mathbf{t}; \mathbf{z})$,

$$\mathcal{I}(\mathbf{t}; \mathbf{z}) = \mathbb{E}_{p(\mathbf{t},\mathbf{z})} \log \frac{p(\mathbf{t}|\mathbf{z})}{p(\mathbf{t})}, \tag{36}$$

here both $p(\mathbf{t}, \mathbf{z})$ and $p(\mathbf{t})$ are intractable due to the integral over $\mathbf{x}$. Following Cover (1999), the mutual information between three random variables $\mathbf{x}$, $\mathbf{t}$, and $\mathbf{z}$ can be defined as

$$\mathcal{I}(\mathbf{x}; \mathbf{t}; \mathbf{z}) = \mathcal{I}(\mathbf{x}; \mathbf{t}) - \mathcal{I}(\mathbf{x}; \mathbf{t}|\mathbf{z}). \tag{37}$$

Eq. 37 is symmetric in $\mathbf{x}$, $\mathbf{t}$ and $\mathbf{z}$, so we have:

$$\mathcal{I}(\mathbf{x}; \mathbf{t}; \mathbf{z}) = \mathcal{I}(\mathbf{t}; \mathbf{z}) - \mathcal{I}(\mathbf{t}; \mathbf{z}|\mathbf{x}). \tag{38}$$

As already proven by Federici et al. (2019), the second term $\mathcal{I}(\mathbf{t}; \mathbf{z}|\mathbf{x})$ equals to zero. Combining Eq. 37 and Eq. 38, we have

$$\mathcal{I}(\mathbf{x}; \mathbf{t}) - \mathcal{I}(\mathbf{x}; \mathbf{t}|\mathbf{z}) = \mathcal{I}(\mathbf{t}; \mathbf{z}) \tag{39}$$

As evident from Eq. 39, maximizing $\mathcal{I}(\mathbf{x}; \mathbf{t})$ and minimizing $\mathcal{I}(\mathbf{x}; \mathbf{t}|\mathbf{z})$ is equivalent to maximizing $\mathcal{I}(\mathbf{t}; \mathbf{z})$.

H Full Results

In this section, we present the full results for those presented in the main paper. Specifically, the results presented here augment those in Fig. 4, Table 1, and Table 2.

Results on the Effect of Noisy Knowledge (η) In Table 3, we how the full results from our noisy knowledge experiments, shown in the main paper's Table 1. As with our other results, these are averaged over three runs. In our development we noticed that precision tended to be the main hindrance on F1, and therefore we additionally report precision.

Full Perplexity Results In Table 4, we how the full language modeling perplexity results that are shown in the main paper's Table 2. Here, we are showing results across both the dev and test sets. We also include standard deviation results (from three runs).

Full Results on Frame Classification In Table 5, we show the full results for the side knowledge (per-event frame) prediction experiments we reported in Fig. 4. The main classification results were averaged over three runs, along with standard deviation. These results are visible in Fig. 4, but we include the numeric values for any future comparisons.

I Setup/Implementation Details

The NYT dataset is available through the LDC and a research license, and the Wikipedia dataset is publicly available under a CC BY-NC 4.0 license. For both datasets, the token vocabulary size is 150k and the number of unique semantic frames is 600. In line with previous research (Rezaee and Ferraro, 2021), for the documents with more than 5 events, we used the first 5 events that had frames. Each model was trained using a single GPU (an RTX 2080 TI, TITAN RTX, or a Quadro 8000). Each RevUp model took approximately 12 hours to train.

We noticed that the observability of side knowledge can affect the relative importance of the loss terms in Eq. 10. Given a mini batch of training data, we define $\hat{\epsilon}$ as an approximation of ϵ. To do so, we calculate the number of observed frames over total frames. In all cases, high importance placed on the regularization term $\mathcal{L}_t$ (Eq. 9) was important to help prevent overfitting. When frames were frequently observed ($\hat{\epsilon} \geq 0.5$), the model needed to rely more heavily on the updating phase ($\mathcal{L}_{\mathcal{I}}$, Eq. 4) and the regularization on z ($\mathcal{L}_z$, Eq. 7) was less important; specifically, we found $\alpha = 0.3$, $\beta = 1\mathrm{e}-6$ and $\zeta = 0.7$ to be effective. Otherwise, we found the model needed to place small but relatively equal weight on the updating phase and z regularization components; specifically, we found

model	noise η	Dataset					
		Wiki			NYT		
		Acc	F1	Prec	Acc	F1	Prec
SSDVAE	0.1	0.77 ± 0.008	0.43 ± 0.019	0.41 ± 0.015	0.76 ± 0.000	0.55 ± 0.001	0.52 ± 0.001
RevUp		**0.85 ± 0.005**	**0.58 ± 0.015**	**0.56 ± 0.012**	**0.83 ± 0.000**	**0.71 ± 0.002**	**0.69 ± 0.002**
SSDVAE	0.2	0.69 ± 0.002	0.35 ± 0.001	0.33 ± 0.002	0.67 ± 0.001	0.44 ± 0.000	0.41 ± 0.001
RevUp		**0.82 ± 0.001**	**0.52 ± 0.001**	**0.49 ± 0.003**	**0.80 ± 0.000**	**0.63 ± 0.001**	**0.60 ± 0.001**
SSDVAE	0.3	0.60 ± 0.010	0.28 ± 0.018	0.27 ± 0.013	0.58 ± 0.001	0.36 ± 0.000	0.34 ± 0.000
RevUp		**0.79 ± 0.008**	**0.45 ± 0.023**	**0.42 ± 0.019**	**0.77 ± 0.000**	**0.58 ± 0.001**	**0.55 ± 0.000**
SSDVAE	0.5	0.41 ± 0.017	0.17 ± 0.014	0.18 ± 0.011	0.41 ± 0.003	0.23 ± 0.001	0.23 ± 0.002
RevUp		**0.72 ± 0.007**	**0.36 ± 0.022**	**0.34 ± 0.017**	**0.71 ± 0.001**	**0.48 ± 0.001**	**0.45 ± 0.000**
SSDVAE	0.7	0.23 ± 0.002	0.09 ± 0.001	0.11 ± 0.001	0.22 ± 0.004	0.11 ± 0.002	0.13 ± 0.002
RevUp		**0.64 ± 0.005**	**0.29 ± 0.003**	**0.28 ± 0.002**	**0.63 ± 0.003**	**0.37 ± 0.002**	**0.35 ± 0.002**
SSDVAE	0.9	0.02 ± 0.001	0.00 ± 0.000	0.01 ± 0.000	0.02 ± 0.001	0.00 ± 0.000	0.01 ± 0.000
RevUp		**0.41 ± 0.009**	**0.09 ± 0.003**	**0.11 ± 0.003**	**0.25 ± 0.039**	**0.06 ± 0.013**	**0.09 ± 0.015**

Table 3: Effect of Noise on the test dataset.

model	ϵ	Dataset			
		Wiki		NYT	
		Valid	Test	Valid	Test
RNNLM	-	64.02 ±2.53	64.57 ±2.60	65.07 ±3.25	56.96 ±2.82
HAQAE	-	49.03 ±3.90	50.10 ±4.05	43.13 ±5.29	39.47 ±4.84
SSDVAE	0.0	46.61 ±1.08	47.50 ±1.06	44.80 ±0.85	39.75 ±1.21
RevUp		**44.38 ±1.38**	**45.36 ±1.40**	**42.61** ±0.58	**39.48** ±0.49
SSDVAE	0.1	45.17 ±1.17	45.91 ±1.15	43.52 ±0.16	39.73 ±0.16
RevUp		**43.99 ±1.79**	**44.87 ±1.83**	**36.10** ±0.94	**33.34** ±0.96
SSDVAE	0.7	44.38 ±0.55	44.79 ±0.58	39.77 ±1.05	36.79 ±0.91
RevUp		**40.90 ±1.11**	**41.74 ±1.05**	**35.77** ±0.25	**33.20** ±0.17
SSDVAE	1.0	36.79 ±0.33	36.96 ±0.34	33.18 ±0.39	30.69 ±0.31
RevUp		**34.28 ±0.89**	**34.85 ±0.90**	**30.61 ±0.38**	**28.33** ±0.43

Table 4: Perplexity results on the Wikipedia and NYT datasets (extension of Table 2). Main results are the average of three runs, along with standard deviation.

$\alpha = 0.1$, $\beta = 0.2$ and $\zeta = 1.0$ to be effective. We obtained these values via experimentation on dev data.

Following SSDVAE and HAQAE models, we use pre-trained Glove 300 embeddings to represent tokens and used gradient clipping at 5.0 to prevent exploding gradients. We use a two-layer of bi-directional GRU for the encoder and a two-layer uni-directional GRU for the decoder (with a hidden dimension of 512 for both). We used early stopping (lack of validation performance improvement for 3 iterations).

model	ϵ	Dataset					
		Wiki			NYT		
		Acc	F1	Prec	Acc	F1	Prec
SSDVAE	0.1	0.02 ± 0.002	0.01 ± 0.001	0.02 ± 0.002	0.02 ± 0.002	0.00 ± 0.001	0.01 ± 0.003
RevUp		**0.06 ± 0.001**	**0.02 ± 0.000**	**0.04 ± 0.000**	**0.05 ± 0.003**	**0.02 ± 0.001**	**0.04 ± 0.002**
SSDVAE	0.3	0.10 ± 0.002	0.05 ± 0.001	0.08 ± 0.002	0.11 ± 0.003	0.06 ± 0.001	0.09 ± 0.002
RevUp		**0.23 ± 0.002**	**0.11 ± 0.001**	**0.13 ± 0.000**	**0.17 ± 0.005**	**0.07 ± 0.002**	**0.09 ± 0.002**
SSDVAE	0.5	0.23 ± 0.002	0.11 ± 0.000	0.14 ± 0.001	0.25 ± 0.005	0.15 ± 0.002	0.18 ± 0.003
RevUp		**0.44 ± 0.014**	**0.22 ± 0.009**	**0.21 ± 0.006**	**0.36 ± 0.017**	**0.22 ± 0.008**	**0.22 ± 0.007**
SSDVAE	0.7	0.52 ± 0.008	0.28 ± 0.008	0.28 ± 0.007	0.56 ± 0.005	0.40 ± 0.005	0.39 ± 0.004
RevUp		**0.84 ± 0.004**	**0.65 ± 0.003**	**0.65 ± 0.002**	**0.76 ± 0.010**	**0.61 ± 0.015**	**0.57 ± 0.018**
SSDVAE	0.9	0.84 ± 0.004	0.61 ± 0.008	0.62 ± 0.010	0.83 ± 0.001	0.77 ± 0.013	0.78 ± 0.014
RevUp		**0.86 ± 0.001**	**0.72 ± 0.003**	**0.73 ± 0.005**	**0.85 ± 0.002**	**0.81 ± 0.007**	**0.82 ± 0.008**
SSDVAE	1.0	0.85 ± 0.002	0.68 ± 0.003	0.69 ± 0.003	0.84 ± 0.001	0.81 ± 0.002	0.82 ± 0.001
RevUp		**0.87 ± 0.003**	**0.75 ± 0.008**	**0.76 ± 0.009**	**0.85 ± 0.001**	**0.82 ± 0.007**	**0.83 ± 0.006**

Table 5: Classification Results: these are the full numeric values (averages and standard deviations computed from three runs) for the graphs shown in Fig. 4.

NusaX: Multilingual Parallel Sentiment Dataset for 10 Indonesian Local Languages

Genta Indra Winata[1*], Alham Fikri Aji[2*], Samuel Cahyawijaya[3*], Rahmad Mahendra[4,5*], Fajri Koto[2,6*], Ade Romadhony[5,7*], Kemal Kurniawan[5,6*], David Moeljadi[8], Radityo Eko Prasojo[9], Pascale Fung[3], Timothy Baldwin[2,6], Jey Han Lau[6], Rico Sennrich[10], Sebastian Ruder[11]

[1]Bloomberg [2]MBZUAI [3]HKUST [4]Universitas Indonesia [5]INACL
[6]The University of Melbourne [7]Telkom University [8]Kanda University of International Studies
[9]Kata.ai [10]University of Zurich [11]Google Research

Abstract

Natural language processing (NLP) has significant impact on society via technologies such as machine translation and search engines. Despite its success, NLP technology is only widely available for high-resource languages such as English and Mandarin Chinese, and remains inaccessible to many languages due to the unavailability of data resources and benchmarks. In this work, we focus on developing resources for languages of Indonesia. Despite being the second most linguistically-diverse country, most languages in Indonesia are categorized as endangered and some are even extinct. We develop the first-ever parallel resource for 10 low-resource languages in Indonesia. Our resource includes sentiment and machine translation datasets, and bilingual lexicons. We provide extensive analysis, and describe challenges for creating such resources. Our hope is that this work will spark more NLP research on Indonesian and other underrepresented languages.

1 Introduction

Indonesia is one of the most populous and linguistically-diverse countries in the world, with more than 700 languages spoken across the country (Aji et al., 2022; Eberhard et al., 2021). However, while many of these languages are spoken by millions of people they have received little attention from the NLP community. There are very few public datasets, preventing the global research community from exploring these languages. To this end, we introduce **NusaX**,[1] a high-quality multilingual parallel corpus that covers 10 local languages from Indonesia: Acehnese, Balinese, Banjarese, Buginese, Madurese, Minangkabau, Javanese, Ngaju, Sundanese, and Toba Batak.

The **NusaX** dataset was created by translating SmSA (Purwarianti and Crisdayanti, 2019) — an existing Indonesian sentiment analysis dataset containing comments and reviews from the IndoNLU benchmark (Wilie et al., 2020) — using competent bilingual speakers, coupled with additional human-assisted quality assurance. Sentiment analysis is one of the most popular NLP tasks, and has been explored in many applications in Indonesia, including presidential elections (Ibrahim et al., 2015; Budiharto and Meiliana, 2018), product reviews (Fauzi, 2019), stock forecasting (Cakra and Trisedya, 2015; Sagala et al., 2020), and COVID-19 monitoring (Nurdeni et al., 2021). By translating an existing text, we additionally produce a parallel corpus, which is useful for building and evaluating translation systems. As we translate from a regional high-resource language (Indonesian), we ensure that the topics and entities reflected in the data are culturally relevant to the other languages, which is generally not the case when translating an English dataset (Conneau et al., 2018; Ponti et al., 2020). We apply the corpus to two downstream tasks: sentiment analysis and machine translation. We use the new benchmark to assess the performance of existing Indonesian language models (LMs), multilingual LMs, and classical machine learning methods.

Our contributions are as follows:

- We propose NusaX, the first high-quality human annotated parallel corpus in 10 languages from Indonesia, and corresponding parallel data in Indonesian and English, covering the tasks of sentiment analysis and machine translation.
- We provide an extensive evaluation of deep learning and classical NLP/machine learning methods on downstream tasks in few-shot and full-data settings.
- We conduct comprehensive analysis of the languages under study both from linguistic

---

[*] These authors contributed equally.

[1] The dataset is released at `https://github.com/IndoNLP/nusax`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 815–834
May 2-6, 2023

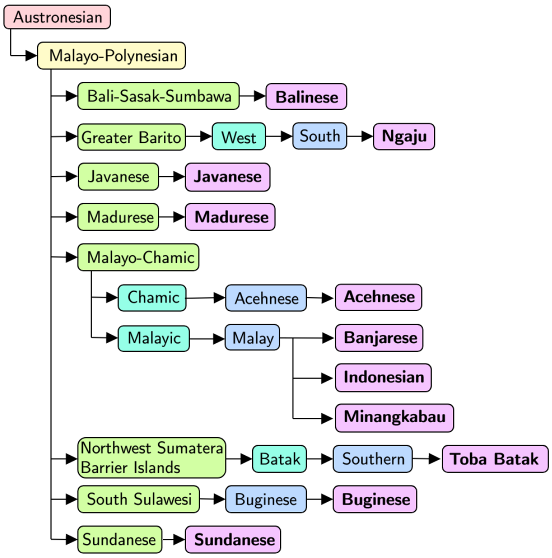


Figure 1: Language taxonomy of the 10 focus languages and Indonesian, according to Ethnologue (Eberhard et al., 2021). The color represents the language category level in the taxonomy. Purple denotes language, and other colors denote language family.

and empirical perspectives, the cross-lingual transferability of existing monolingual and multilingual LMs, and an efficiency analysis of various methods for NLP tasks in extremely low-resource languages.

2 Focus Languages

We work on 10 local languages in Indonesia: Acehnese, Balinese, Banjarese, Buginese, Madurese, Minangkabau, Javanese, Ngaju, Sundanese, and Toba Batak. Most of these languages have a population of over 2 million speakers (van Esch et al., 2022; Aji et al., 2022), but are underrepresented in NLP research. Figure 1 shows the taxonomy of these languages and Indonesian. Geographically, these languages are spoken on different big islands in Indonesia, including Sumatra, Borneo, Java, Madura, and Sulawesi. The languages belong to the Austronesian language family under the Malayo-Polynesian subgroup. While some of the covered languages are written in multiple scripts, we use the Latin script in NusaX, which has become predominant for all covered languages.

Indonesian (ind) is the national language of Indonesia based on the 1945 Constitution of the Republic of Indonesia (article 36). It is written in Latin script, and was developed from literary "Classical Malay" of the Riau-Johor sultanate (Sneddon, 2003), with regional variants. Its lexical similarity to Standard Malay is over 80%. It has a rich affixation system, including a variety of prefixes, suffixes, circumfixes, and reduplication. Most of the affixes in Indonesian are derivational (Pisceldo et al., 2008).

Acehnese (ace) is a language spoken mainly in the Aceh province. Although it is the de facto language of Aceh, language use is shifting to Indonesian in urban areas. Acehnese has features typical of the Mon-Khmer languages of mainland Southeast Asia, a result of its former status as part of the early Chamic dialect continuum on the coast of Vietnam. In addition to the large number of diphthongs, it has a high percentage of monosyllabic root morphemes.

Balinese (ban) is a language spoken mainly in the Bali province. It has three main dialects: Highland Balinese, Lowland Balinese, and Nusa Penida. Since the early 20th century, it has mainly been written in the Latin script, but also has its own Balinese script. The word order in Balinese is SVO. Balinese has three sociolinguistic registers (Arka, 2003).

Banjarese (bjn) is a language spoken in Kalimantan (Central, East, South, and West Kalimantan provinces). It is dominant in the South Kalimantan Province and is also growing rapidly in the Central and Eastern Kalimantan provinces. It has two main dialects: Kuala and Hulu. Although it is a Malayic language, it has many Javanese loanwords, probably acquired during the Majapahit period from the late thirteenth century until the fifteenth century (Blust et al., 2013). It has 73% of lexical similarity with Indonesian and is written in Arabic and Latin scripts (Eberhard et al., 2021).

Buginese (bug) is a language spoken mainly in the South Sulawesi, Southeast Sulawesi, Central Sulawesi, and West Sulawesi provinces. The word order is SVO. Verb affixes are used to mark persons. Historically, it was written in the Buginese script (derived from Brahmi script), but is mainly written in Latin script now (Eberhard et al., 2021). Buginese employs sentence patterns, pronouns, and other terms to express politeness (Weda, 2016).

Madurese (mad) is a language spoken in the East Java province, mainly on Madura Island, south and west of Surabaya city, Bawean, Kangean, and Sapudi islands. It has vowel harmony, gemination, rich affixation, reduplication, and SVO basic word order (Davies, 2010).

Minangkabau (min) is a language spoken mainly in West Sumatra and other provinces on Sumatra Island such as Bengkulu and Riau. Although it is classified as Malay, it is not intelligible with Indonesian. Standard Minangkabau voice can be characterised as an Indonesian-type system, whereas colloquial Minangkabau voice is more effectively characterised as a Sundic-type system (Crouch, 2009).

Javanese (jav) is a language spoken mainly on Java Island. It is the de facto language of provincial identity in central and eastern Java. The number of native Javanese speakers is greater than the number of Indonesian L1 speakers (Eberhard et al., 2021). Javanese consists of several regional dialects, which differ primarily in pronunciation and vocabulary. Javanese has an elaborate system of speech levels related to the relation of the speaker to the interlocutor that depend on social status, age, kinship distance, and familiarity (Wedhawati et al., 2001). It used to be written in Javanese script, but since the 20th century has mostly been written in Latin script.

Ngaju (nij) is a language spoken in the Central Kalimantan province. It is widely used as a language for trade in much of Kalimantan, from the Barito to the Sampit River. It has various affixes and reduplication, and its word order is similar to Indonesian. Pronouns have enclitic forms to mark possessors in a noun phrase or passive agents (Uchibori and Shibata, 1988).

Sundanese (sun) is a language spoken mainly in the Banten and West Java provinces. It is the de facto language of provincial identity in western Java. The main dialects are Bogor (Krawang), Pringan, and Cirebon. It has elaborate coding of respect levels. It has been written in Latin script since the mid-19th century but was previously written in Arabic, Javanese, and Sundanese scripts. Sundanese is a predominantly SVO language, and has voice marking and incorporates some (optional) actor-verb agreement, i.e., number and person (Kurniawan, 2013).

Toba Batak (bbc) is a language spoken in the North Sumatra province. Similarly to Acehnese, it is slowly being replaced by Indonesian in urban and migrant areas. It used to be written in the Batak script but is mainly written in Latin script now. The Batak languages are verb-initial, and have verb systems reminiscent of Philippine languages, although they differ from them in many details (Blust et al., 2013).

3 Data Construction

Our data collection process consists of several steps. First, we take an existing dataset in a high-resource local language (Indonesian) as a base for expansion to the other ten languages, and ask human annotators to translate the text. To ensure the quality of the final translation, we run quality assurance with additional human annotators.

3.1 Annotator Recruitment

Eliciting or annotating data in underrepresented languages generally requires working with local language communities in order to identify competent bilingual speakers (Nekoto et al., 2020). In the Indonesian setting, this challenge is compounded by the fact that most languages have several dialects. As dialects in Indonesian languages may have significant differences in word usage and meaning (Aji et al., 2022), it is important to recruit annotators who speak the same or similar dialects to ensure that translations are mutually intelligible.

In this work, we employ at least 2 expert annotators who are native speakers of each local language and Indonesian. To filter the recruited annotators, we first ask annotator candidates to translate three samples. We then conduct a peer review by asking whether they can understand the translations of other annotators for the same language, using the hired annotators as translators as well as translation validators. We also conducted 2 hours of training to introduce the user interface of the annotation system for selected workers. For English translations, we hire annotators based on their English proficiency test scores with an IELTS score ≥ 6.5 or TOEFL PBT score ≥ 600.

3.2 Data Filtering and Sampling

We base our dataset on SmSA, the largest publicly available Indonesian sentiment analysis dataset from the IndoNLU benchmark (Purwarianti and Crisdayanti, 2019; Wilie et al., 2020). SmSA is an expert-annotated sentence-level multi-domain sentiment analysis dataset consisting of more than 11,000 instances of comments and reviews collected from several online platforms such as Twitter, Zomato, and TripAdvisor. We filter the data to remove abusive language and personally-identifying information by manually inspecting all sentences. We randomly select 1,000 samples via stratified sampling for translation, ensuring that the label distribution is balanced.

3.3 Human Translation

We instructed the annotators to retain the meaning of the text and to keep entities such as persons, organizations, locations, and time with no target language translation the same. Specifically, we instructed them to: (1) maintain the sentence's sentiment polarity; (2) preserve entities; and (3) maintain the complete information content of the original text.

Initially, we asked the translators to maintain the typography. Most sentences from the original dataset are written in an informal tone, with non-standard spelling, e.g., elongated vowels and punctuation. When the sentence is translated into the target language, direct translation can sound unnatural. For example, translating the Indonesian word *kangeeeen* (originally *kangen*; en: *miss*) to *taragaaaak* (originally *taragak*) in Minangkabau may sound unnatural. Similarly, the original sentence may also contain typos. Due to the difficulty of accurately assessing typographical consistency of translations, we removed this as a criterion.

3.4 Human-Assisted Quality Assurance

We conduct quality control (QC) between two annotators by having annotator A check the translations of annotator B, and vice versa. We include the corrected translations in our dataset. To ensure the quality assurance is performed well, we randomly perturb 5% of the sentences by removing a random sequence of words. The quality assurance annotators are then expected to notice the perturbed sentences and fix them.

We analyze the quality assurance edits for Balinese, Sundanese, and Javanese, which are spoken by the authors of this paper. For each language, we randomly sample 100 translations that have been edited by a QC annotator. We classify edits as follows:

Typos and Mechanics: Edit that involves correcting typos, punctuation, casing, white spaces/dashes, and numerical formatting.

Orthography: Edit that changes the spelling of words due to orthographic variation in local languages without a standard orthography. The word sounds and means the same before and after editing, and both are used by natives. The QC annotator might feel that one writing variant is more natural/commonly used, and hence make this change.

Translation: The words used by the translator are still in Indonesian and the QC annotator translates them to the local language.

Word edit: The QC annotator paraphrases a word/phrase. This also includes adding/removing words and morpheme changes.

Major changes: Other edits that significantly alter the original translation.

The results are shown in Table 1. Generally, word edits make up the majority of QC modifications, which involve replacing a word/phrase with a synonym or altering a morpheme slightly. In contrast, major changes are extremely rare. We also see changes to the orthography around 10% of the time. Other types of edits vary between languages. Sundanese has significantly less typos compared to other languages, but a considerably higher number of translation edits. We suspect this is because code-switching with Indonesian happens regularly in Sundanese, which results in many Indonesian words being adopted despite the existence of equivalent Sundanese translations.

Category	ban	sun	jav
Typos & Mechanic	31	14	42
Orthography	14	6	12
Translation	22	55	10
Word edit	67	65	61
Major changes	3	0	1

Table 1: Statistics of QC edits per category over 100 samples.

3.5 Bilingual Lexicon Creation

Bilingual lexicons are useful for data augmentation (Wang et al., 2022) and evaluating cross-lingual representations (Artetxe et al., 2018). We select 400 words from an Indonesian lexicon[2] to be translated into the 10 local languages and English. For each language, we employ two annotators and ask them to translate the word into all possible lexemes. The translations from both annotators are combined. We obtain 800–1,600 word pairs for each of our 11 language pairs (from Indonesian to the remaining languages). We augment the bilingual lexicon with data from PanLex (Kamholz et al., 2014).

---

[2] `https://github.com/andria009/IndonesianSentimentLexicon`

4 NusaX Benchmark

4.1 Tasks

We develop two tasks — sentiment analysis and machine translation — based on the datasets covering 12 languages, including Indonesian, English, and the 10 local languages. For the NusaX sentiment dataset, each language has the same label distribution and we show the label distribution of each dataset subset in Table 2. We maintain the label ratio in each dataset subset to ensure a similar distribution. More details of the dataset are provided in Appendix C.

4.1.1 Sentiment Analysis

Sentiment analysis is an NLP task that aims to identify the sentiment of a given text document. The sentiment is commonly categorized into 3 classes: positive, negative, and neutral. We focus our dataset construction on sentiment analysis because it is one of the most widely explored tasks in Indonesia (Aji et al., 2022) due to broad industrial relevance, such as for competitor and marketing analysis, and detection of unfavorable rumors for risk management (Socher et al., 2013). After translating 1,000 instances from the sentiment analysis dataset (SmSA), we have a sentiment analysis dataset for each translated language. For each language, we split the dataset into 500 train, 100 validation, and 400 test examples. In total, our dataset contains 6,000 train, 1,200 validation, and 4,800 test instances across 12 languages (Indonesian, English and the 10 local languages).

4.1.2 Machine Translation

Indonesia consists of 700+ languages covering three different language families (Aji et al., 2022). Despite its linguistic diversity, existing machine translation systems only cover a small fraction of Indonesian languages, mainly Indonesian (the national language), Sundanese, and Javanese. To broaden the coverage of existing machine translation systems for underrepresented local languages, we construct a machine translation dataset using our translated sentiment corpus, which results in a parallel corpus between all language pairs. In other words, we have 132 possible parallel corpora, each with 1,000 samples (500 train, 100 validation, and 400 test instances) which can be used to train machine translation models. Compared to many other MT evaluation datasets, our data is in the review domain and is not English-centric.

Subset	Negative	Neutral	Positive
Train	192	119	189
Valid	38	24	38
Test	153	96	151

Table 2: Label distribution of NusaX Sentiment dataset.

4.2 Baselines

4.2.1 Classical Machine Learning

Classical machine learning approaches are still widely used by local Indonesian researchers and institutions due to their efficiency (Nityasya et al., 2021). The trade-off between performance and compute cost is particularly important in situations with limited compute, which are common for low-resource languages. We therefore use classical methods as baselines for our comparison. Namely, we use naive Bayes, SVM, and logistic regression for the classification tasks. For MT, we employ a naive baseline that copies the original Indonesian text, a dictionary-based substitution method using the bilingual lexicon, and a phrase-based MT system based on Moses (Koehn et al., 2007).

4.2.2 Pre-trained Local Language Models

Recent developments in neural pre-trained LMs have brought substantial improvements in various NLP tasks. Despite the lack of resources in Indonesian and local languages, there have been some efforts in developing large pre-trained LMs for Indonesian and major local languages. IndoBERT (Wilie et al., 2020) and Sundanese-BERT (Wongso et al., 2022) are two popular LMs for natural language understanding (NLU) tasks in Indonesian and Sundanese. IndoBART and IndoGPT have also been introduced for natural language generation (NLG) tasks in Indonesian, Sundanese, and Javanese (Cahyawijaya et al., 2021). We employ these LMs as baselines to assess their adaptability to other languages.

4.2.3 Massively Multilingual LMs

We consider large pre-trained multilingual LMs to further understand their applicability to low-resource languages. Specifically, we experiment with mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) for sentiment analysis, and mBART (Liu et al., 2020) and mT5 (Xue et al., 2021) for machine translation. We provide the hyper-parameters of all models in Appendix B.

Model	ace	ban	bbc	bjn	bug	eng	ind	jav	mad	min	nij	sun	**avg**
Naive Bayes	72.5	72.6	73.0	71.9	73.7	76.5	73.1	69.4	66.8	73.2	68.8	71.9	72.0
SVM	75.7	75.3	**76.7**	74.8	**77.2**	75.0	78.7	71.3	73.8	76.7	75.1	74.3	75.4
LR	**77.4**	76.3	76.3	75.0	**77.2**	75.9	74.7	73.7	74.7	74.8	73.4	75.8	75.4
IndoBERT$_{\text{BASE}}$	75.4	74.8	70.0	83.1	73.9	79.5	90.0	81.7	**77.8**	82.5	**75.8**	77.5	78.5
IndoBERT$_{\text{LARGE}}$	76.3	**79.5**	74.0	83.2	70.9	87.3	90.2	**85.6**	77.2	82.9	**75.8**	77.2	**80.0**
IndoLEM$_{\text{BASE}}$	72.6	65.4	61.7	71.2	66.9	71.2	87.6	74.5	71.8	68.9	69.3	71.7	71.1
mBERT$_{\text{BASE}}$	72.2	70.6	69.3	70.4	68.0	84.1	78.0	73.2	67.4	74.9	70.2	74.5	72.7
XLM-R$_{\text{BASE}}$	73.9	72.8	62.3	76.6	66.6	90.8	88.4	78.9	69.7	79.1	75.0	80.1	76.2
XLM-R$_{\text{LARGE}}$	75.9	77.1	65.5	**86.3**	70.0	**92.6**	**91.6**	84.2	74.9	**83.1**	73.3	**86.0**	**80.0**

Table 3: Sentiment analysis results in macro-F1 (%). Models were trained and evaluated on each language.

5 Results

5.1 Overall Results

Sentiment Analysis Table 3 shows the sentiment analysis performance of various models across different local languages, trained and evaluated using data in the same language. Fine-tuned large LMs such as IndoBERT$_{\text{LARGE}}$ and XLM-R$_{\text{LARGE}}$ generally achieve the best performance. XLM-R models achieve strong performance on some languages, such as Indonesian (idn), Banjarese (bjn), English (eng), Javanese (jav), and Minangkabau (min). Many of these languages are included in XLM-R's pre-training data while others may benefit from positive transfer from related languages. For instance, Banjarese is similar to Malay and Indonesian (Nasution et al., 2021), while Minangkabau shares some words and syntax with Indonesian (Koto and Koto, 2020). IndoBERT models, despite only being pre-trained on Indonesian, also show good performance across some local languages, suggesting transferability from Indonesian to the local languages.

The classic approaches are surprisingly competitive with the neural methods, with logistic regression even outperforming IndoBERT$_{\text{LARGE}}$ and XLM-R on Acehnese (ace), Buginese (bug), and Toba Batak (bbc). These results indicate that both Indonesian and multilingual pre-trained LMs cannot transfer well to these languages, which is supported by the fact that these languages are very distinct from Indonesian, Sundanese, Javanese, or Minangkabau — the languages covered by IndoBERT and XLM-R.

Machine Translation We show the results on machine translation in Table 4 (x → idn) based on SacreBLEU (Post, 2018). As some local languages are similar to Indonesian, we observe that the `Copy` baseline (which does not do any translation) performs quite well. Minangkabau (min) and Banjarese (bjn) achieve high BLEU without any translation despite not being included in the LM pre-training data, due to their similarity with Indonesian (Koto and Koto, 2020; Nasution et al., 2021). Since these local languages share grammatical structure with Indonesian, dictionary-based word substitution yields a reasonable improvement.

Both PBSMT and fine-tuned LMs reach encouraging performance levels despite the limited training data, which we again attribute to the target languages' similarity to Indonesian. In contrast, the performance for translating Indonesian languages from/to English is extremely poor as shown in Table 5, demonstrating the importance of non-English-centric translation. Overall, we observe good translation performance across local languages. Thus, there is an opportunity to utilize translation models to create new synthetic datasets in local languages via translation from a related high-resource language, not only for Indonesian local languages but also other underrepresented languages. However, note that even for language pairs where the SacreBLEU score is very high, we observe translation deficiencies stemming from the small amount of training data: rare words may just be copied with PBSMT, and mistranslated with NMT.

Similar effects are also observed for (idn → x) translation, as shown in Table 6. Similar to (x → idn) translations, we observe that the `Copy` baseline performs quite well on Minangkabau (min) and Banjarese (bjn) due to their similarity with Indonesian (Koto and Koto, 2020; Nasution et al., 2021). Dictionary-based word substitution also yields a reasonable improvement especially for Ja-

	x → idn											
Model	ace	ban	bbc	bjn	bug	eng	jav	mad	min	nij	sun	**avg**
Copy	5.88	9.99	4.28	15.99	3.44	0.57	9.29	5.11	18.10	7.51	9.24	8.13
Word Substitution	7.33	12.30	5.02	16.17	3.52	1.67	17.34	7.89	24.17	12.07	15.38	11.17
PBSMT	**25.17**	**41.22**	**20.94**	47.80	**15.21**	6.68	46.99	**38.39**	**60.56**	32.86	41.79	34.33
IndoGPT	7.01	13.23	5.27	19.53	1.98	4.26	27.31	13.75	23.03	10.83	23.18	13.58
IndoBARTv2	24.44	40.49	19.94	**47.81**	12.64	11.73	**50.64**	36.10	58.38	**33.50**	**45.96**	**34.69**
mBART-50	18.45	34.23	17.43	41.73	10.87	**17.92**	39.66	32.11	59.66	29.84	35.19	30.64
mT5$_{\text{BASE}}$	18.59	21.73	12.85	42.29	2.64	12.96	45.22	32.35	58.65	25.61	36.58	28.13

Table 4: Results of the machine translation task from other languages to Indonesian (x → idn) based on SacreBLEU.

Model	**avg. SacreBLEU**			
	ind → x	x → ind	eng → x	x → eng
PBSMT	**28.72**	34.33	4.56	5.84
IndoBARTv2	28.21	**34.69**	6.36	**7.46**
mBART-50	24.69	30.64	**7.20**	6.45

Table 5: MT performance from / to Indonesian compared to from / to English.

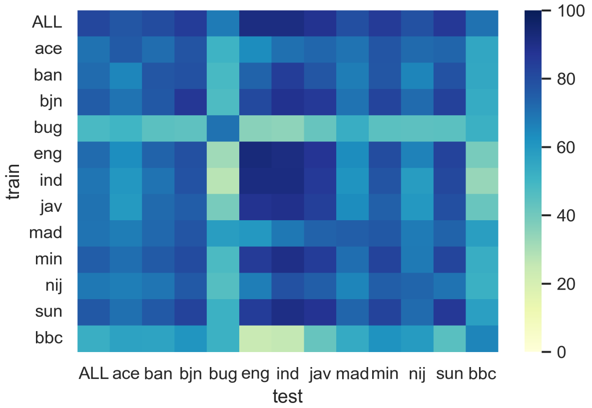


Figure 2: Zero-shot cross-lingual results for the sentiment analysis task with XLM-R$_{\text{LARGE}}$. The model is trained on the language indicated on the x-axis and evaluated on all languages.

vanese (jav), Minangkabau (min), and Sundanese (sun) due to the high similarity of the grammatical structure with Indonesian. PBSMT and fine-tuned IndoBARTv2 models achieve the best scores over multiple local languages despite the limited training data, which is also attributed to the target languages' similarity to Indonesian.

5.2 Cross-lingual Capability of LMs

From a linguistic perspective, local languages in Indonesia share similarities according to language family. Many local languages share a similar grammatical structure and have some vocabulary overlap. Following prior work that demonstrates positive transfer between closely-related languages (Cahyawijaya et al., 2021; Hu et al., 2020; Aji et al., 2020; Khanuja et al., 2020; Winata et al., 2021, 2022), we analyze the transferability between closely-related languages in the Malayo-Polynesian language family.

Empirically, we show the cross-lingual capability of the best performing model (XLM-R$_{\text{LARGE}}$) in the zero-shot cross-lingual setting for sentiment analysis. The heatmap is shown in Figure 2. In general, most languages, except for Buginese (bug) and Toba Batak (bbc), can be used effectively as the source language, reaching ∼70–75% F1 on average, compared to an average of 80% F1 in the monolingual setting (cf. XLM-R$_{\text{LARGE}}$ in Table 3). This empirical result aligns with the fact that both Buginese (bug) and Toba Batak (bbc) have very low vocabulary overlap with Indonesian (cf. Copy in Tables 4 and 6). Interestingly, despite coming from a completely different language family, English can also be effectively used as the source language for all 10 local languages, likely due to its prevalence during pre-training.

These results demonstrate that we can take advantage of language similarity by transferring knowledge from Indonesian and other local languages to perform zero-shot or few-shot classification in closely-related languages. New datasets for underrepresented languages that are closely related to high-resource languages thus do not necessarily need to be large, which make the development of NLP datasets in low-resource languages more affordable than may initially appear to be the case.

5.3 Multilingual Capability

We explore training multilingual models, as most Indonesian local languages share similarities. For sentiment analysis, we concatenate the training

ind → x												
Model	ace	ban	bbc	bjn	bug	eng	jav	mad	min	nij	sun	**avg**
Copy	5.89	10.00	4.28	15.99	3.45	0.56	9.29	5.11	18.10	7.52	9.24	8.13
Word Substitution	7.60	10.31	5.99	17.51	3.57	0.76	14.75	7.58	22.34	9.76	12.38	10.23
PBSMT	**20.47**	26.48	18.18	**42.08**	10.84	7.73	39.08	**33.26**	**52.21**	**29.58**	36.04	**28.72**
IndoGPT	9.60	14.17	8.20	22.23	5.18	5.89	24.05	14.44	26.95	17.56	23.15	15.58
IndoBARTv2	19.21	**27.08**	**18.41**	40.03	**11.06**	**11.53**	**39.97**	28.95	48.48	27.11	**38.46**	28.21
mBART-50	17.21	22.67	17.79	34.26	10.78	3.90	35.33	28.63	43.87	25.91	31.21	24.69
mT5$_{\text{BASE}}$	14.79	18.07	18.22	38.64	6.68	11.21	33.48	0.96	45.84	13.59	33.79	21.39

Table 6: Results of the machine translation task from Indonesian to other languages (idn → x) in SacreBLEU.

Language	**Single**	**Multi**	**LOLO**
Acehnese	75.9	76.96	75.79
Balinese	77.1	80.13	77.83
Banjarese	86.3	84.85	82.68
Buginese	70.0	67.86	63.67
English	92.6	91.05	89.88
Indonesian	91.6	91.13	90.62
Javanese	84.2	88.19	87.39
Madurese	74.9	79.41	78.52
Minangkabau	83.1	85.29	84.45
Ngaju	73.3	78.82	76.31
Sundanese	86.0	86.02	84.41
Toba batak	65.5	70.00	68.76
Average	80.04	81.64	80.03

Table 7: Sentiment analysis results for macro-F1 (%) of XLM-R$_{\text{LARGE}}$ in the multilingual setting.

data of all languages. Additionally, we also explore Leave-One-Language-Out (LOLO), where we train on all data except for the test language. The LOLO setting arguably reflects the most realistic scenario where we do not have training data for a particular language, but we do have access to data in other local languages. The multilingual results for sentiment analysis are shown in Table 7. Multilingual training outperforms monolingual training, while LOLO matches the performance of training on target language data. Related language data is thus often sufficient for good cross-lingual results.

6 Data Collection Challenges

In this section, we discuss challenges faced during data collection.

Finding annotators We found collecting the NusaX dataset challenging. First of all, finding local language-speaking annotators is not easy, and popular platforms such as MTurk do not support these languages. Instead, we looked for annotators through local Indonesian networks and forums, such as the INACL forum, local campus forums, or the Indonesian polyglot community. We intended to cover as many local languages as possible, but based on the available annotators, only the 10 languages presented in this paper were possible, as we needed at least 2 annotators for each language. Searching for annotators online is not easy, due to disparities in Internet penetration in different parts of Indonesia. Hence, we might not reach potential annotators through online communities alone. However, holding an in-person workshop for data collection is also not practical; Indonesia is an archipelago and traveling between islands is costly. Similar challenges occur in many other regions, including Africa and South America.

Communication with annotators Communication between the authors and annotators was done through WhatsApp, as the most popular communication tool in Indonesian (Mulyono et al., 2021). Annotation was conducted through spreadsheets. We found that some of the annotators use mobile apps instead of a desktop for annotation. Their reasons include ease of use, no access to a laptop, and better keyboard support for typing diacritics. In the most extreme case, one annotator printed out the sheet and performed the annotation on paper, then took a picture of the paper and sent it back to us. We found some annotators to be difficult to contact, due to other commitments such as college or work. Some of them were not responsive and had to be replaced by new annotators.

7 Related Work

Multilingual Parallel Corpora Several multilingual parallel corpora have been developed to support studies on machine translation such as GCP (Imamura and Sumita, 2018), Leipzig (Goldhahn et al., 2012), JRC Acquis (Steinberger et al., 2006), TUFS Asian Language Parallel (Nomoto et al., 2018), Intercorp (ek Čermák and Rosen, 2012), DARPA LORELEI (Strassel and Tracey, 2016), Asian Language Treebank (Riza et al., 2016), FLORES (Guzmán et al., 2019), the Bible Parallel Corpus (Resnik et al., 1999; Black, 2019), JW-300 (Agić and Vulić, 2019), BiToD (Lin et al., 2021), and WikiMatrix (Schwenk et al., 2021). Guzmán et al. (2019) describe the procedure to generate high-quality translations as part of FLORES. Similar to FLORES, we also conducted QC of the translations.

Emerging Language Benchmarks Recently, benchmarks in underrepresented languages have emerged, such as MasakhaNER (Adelani et al., 2021), AmericasNLI (Ebrahimi et al., 2022), PMIndia (Haddow and Kirefu, 2020), Samanantar (Ramesh et al., 2022), and NaijaSenti (Muhammad et al., 2022). Particularly, for Indonesian languages, NLP benchmarks have been developed such as IndoNLU (Wilie et al., 2020), IndoLEM (Koto et al., 2020), IndoNLG (Cahyawijaya et al., 2021), IndoNLI (Mahendra et al., 2021), and English–Indonesian machine translation (Guntara et al., 2020).

Datasets for Indonesian Local Languages Only a limited number of labeled datasets exist for local languages in Indonesia. WikiAnn (Pan et al., 2017) — a weakly-supervised named entity recognition dataset — covers Acehnese, Javanese, Minangkabau, and Sundanese. Putri et al. (2021) built a multilingual dataset for abusive language and hate speech detection involving Javanese, Sundanese, Madurese, Minangkabau, and Musi languages. Sakti and Nakamura (2013) constructed speech corpora for Javanese, Sundanese, Balinese, and Toba Batak. Few datasets exist for individual languages, e.g., sentiment analysis and machine translation in Minangkabau (Koto and Koto, 2020) and emotion classification in Sundanese (Putra et al., 2020). Finally, some datasets focus on colloquial Indonesian mixed with local languages in the scope of morphological analysis (Wibowo et al., 2021) and style transfer (Wibowo et al., 2020).

8 Conclusion

In this paper, we propose NusaX, the first parallel corpus for 10 low-resource Indonesian languages. We create a new benchmark for sentiment analysis and machine translation in zero-shot and full-data settings. We present a comprehensive analysis of the language similarity of these languages from both linguistic and empirical perspectives by assessing the cross-lingual transferability of existing Indonesian and multilingual pre-trained models.

We hope NusaX can enable NLP research for under-represented languages, and can be used as a testbed for adaptation or few-shot learning methods that take advantage of similarities between languages. NusaX opens up the possibility for future research that focuses on covering more local languages, and additionally, further extension to other tasks and domains. Our study on cross-lingual transfer enables further exploration on cross-lingual zero-shot learning for more diverse tasks in local languages. Our guidelines and discussion of data collection issues may also motivate future work on more efficient high-quality data collection for extremely low-resource languages.

Acknowledgments

We thank Dea Adhista and all annotators who helped us in building the corpus. We are grateful to Alexander Gutkin and Xinyu Hua for feedback on a draft of this manuscript. This work has been partially funded by Kata.ai (001/SD/YGI-NLP/1/2022) and PF20-43679 Hong Kong PhD Fellowship Scheme, Research Grant Council, Hong Kong.

Limitations

We created data for low-resource languages, which increases the accessibility of NLP research for marginalized communities. However, we were only able to cover 10 languages with only 1000 samples each, due to cost and the number of available annotators. This dataset has limited domain coverage and may also contain biases towards certain groups or entities. We tried our best to eliminate negative biases based on a manual inspection of the data. As our dataset was translated, there may be some translationese artifacts in the resulting corpus. We invited annotators based on their fluency level on a particular language. However, the fluency level is self-declared, and there is no mechanism to verify it, except for several languages that are spoken by authors of this paper. The dialect used in the dataset also depends on the annotator, for languages with multiple dialects.

References

David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, Shamsuddeen H. Muhammad, Chris Chinenye Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Aremu Anuoluwapo, Catherine Gitau, Derguene Mbaye, Jesujoba Alabi, Seid Muhie Yimam, Tajuddeen Rabiu Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, Verrah Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, Chiamaka Chukwuneke, Nkiruka Odu, Eric Peter Wairagala, Samuel Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane MBOUP, Dibora Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima DIOP, Abdoulaye Diallo, Adewale Akinfaderin, Tendai Marengereke, and Salomey Osei. 2021. MasakhaNER: Named Entity Recognition for African Languages. *Transactions of the Association for Computational Linguistics*, 9:1116–1131.

Željko Agić and Ivan Vulić. 2019. JW300: A wide-coverage parallel corpus for low-resource languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210, Florence, Italy. Association for Computational Linguistics.

Alham Fikri Aji, Nikolay Bogoychev, Kenneth Heafield, and Rico Sennrich. 2020. In neural machine translation, what does transfer learning transfer? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7701–7710.

Alham Fikri Aji, Genta Indra Winata, Fajri Koto, Samuel Cahyawijaya, Ade Romadhony, Rahmad Mahendra, Kemal Kurniawan, David Moeljadi, Radityo Eko Prasojo, Timothy Baldwin, Jey Han Lau, and Sebastian Ruder. 2022. One country, 700+ languages: NLP challenges for underrepresented languages and dialects in Indonesia. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7226–7249, Dublin, Ireland. Association for Computational Linguistics.

I Wayan Arka. 2003. *Balinese morphosyntax: a lexical-functional approach*. Pacific Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Annisa Nurul Azhar, Masayu Leylia Khodra, and Arie Pratama Sutiono. 2019. Multi-label aspect categorization with convolutional neural networks and extreme gradient boosting. In *2019 International Conference on Electrical Engineering and Informatics (ICEEI)*, pages 35–40. IEEE.

Alan W Black. 2019. Cmu wilderness multilingual speech dataset. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5971–5975. IEEE.

Robert Blust et al. 2013. *The Austronesian Languages*. The Australian National University.

Widodo Budiharto and Meiliana Meiliana. 2018. Prediction and analysis of indonesia presidential election from twitter using sentiment analysis. *Journal of Big data*, 5(1):1–10.

Samuel Cahyawijaya, Genta Indra Winata, Bryan Wilie, Karissa Vincentio, Xiaohong Li, Adhiguna Kuncoro, Sebastian Ruder, Zhi Yuan Lim, Syafri Bahar, Masayu Khodra, Ayu Purwarianti, and Pascale Fung. 2021. IndoNLG: Benchmark and resources for evaluating Indonesian natural language generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8875–8898, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yahya Eru Cakra and Bayu Distiawan Trisedya. 2015. Stock price prediction using linear regression based on sentiment analysis. In *2015 international conference on advanced computer science and information systems (ICACSIS)*, pages 147–154. IEEE.

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

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Sophie Elizabeth Crouch. 2009. *Voice and verb morphology in Minangkabau, a language of West Sumatra, Indonesia*. Ph.D. thesis, The University of Western Australia.

William D Davies. 2010. *A grammar of Madurese*. Mouton De Gruyter.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig. 2021. *Ethnologue: Languages of the World. Twenty-fourth edition.* Dallas, Texas: SIL International.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir Meza Ruiz, Gustavo Giménez-Lugo, Elisabeth Mager, Graham Neubig, Alexis Palmer, Rolando Coto-Solano, Thang Vu, and Katharina Kann. 2022. AmericasNLI: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6279–6299, Dublin, Ireland. Association for Computational Linguistics.

Franti ek Čermák and Alexandr Rosen. 2012. The case of intercorp, a multilingual parallel corpus. *International Journal of Corpus Linguistics*, 17(3):411–427.

M Ali Fauzi. 2019. Word2vec model for sentiment analysis of product reviews in indonesian language. *International Journal of Electrical and Computer Engineering*, 9(1):525.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 759–765, Istanbul, Turkey. European Language Resources Association (ELRA).

Tri Wahyu Guntara, Alham Fikri Aji, and Radityo Eko Prasojo. 2020. Benchmarking multidomain english-indonesian machine translation. In *Proceedings of the 13th Workshop on Building and Using Comparable Corpora*, pages 35–43.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6098–6111, Hong Kong, China. Association for Computational Linguistics.

Barry Haddow and Faheem Kirefu. 2020. Pmindia–a collection of parallel corpora of languages of india. *arXiv preprint arXiv:2001.09907*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization. In *Proceedings of ICML 2020*.

Mochamad Ibrahim, Omar Abdillah, Alfan F Wicaksono, and Mirna Adriani. 2015. Buzzer detection and sentiment analysis for predicting presidential election results in a twitter nation. In *2015 IEEE international conference on data mining workshop (ICDMW)*, pages 1348–1353. IEEE.

Kenji Imamura and Eiichiro Sumita. 2018. Multilingual parallel corpus for global communication plan. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

David Kamholz, Jonathan Pool, and Susan Colowick. 2014. PanLex: Building a resource for panlingual lexical translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3145–3150, Reykjavik, Iceland. European Language Resources Association (ELRA).

Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. 2020. Gluecos: An evaluation benchmark for code-switched nlp. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3575–3585.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Fajri Koto and Ikhwan Koto. 2020. Towards computational linguistics in Minangkabau language: Studies on sentiment analysis and machine translation. In *Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation*, pages 138–148, Hanoi, Vietnam. Association for Computational Linguistics.

Fajri Koto, Afshin Rahimi, Jey Han Lau, and Timothy Baldwin. 2020. Indolem and indobert: A benchmark dataset and pre-trained language model for indonesian nlp. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 757–770.

Eri Kurniawan. 2013. *Sundanese complementation*. Ph.D. thesis, The University of Iowa.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, Peng Xu, Feijun Jiang, Yuxiang Hu, Chen Shi, and

Pascale Fung. 2021. BiToD: A bilingual multi-domain dataset for task-oriented dialogue modeling. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Rahmad Mahendra, Alham Fikri Aji, Samuel Louvan, Fahrurrozi Rahman, and Clara Vania. 2021. IndoNLI: A natural language inference dataset for Indonesian. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10511–10527, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shamsuddeen Hassan Muhammad, David Adelani, Anuoluwapo Aremu, and Idris Abdulmumin. 2022. NaijaSenti: A nigerian Twitter sentiment corpus for multilingual sentiment analysis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 590–602, Marseille, France. European Language Resources Association.

Herri Mulyono, Gunawan Suryoputro, and Shafa Ramadhanya Jamil. 2021. The application of whatsapp to support online learning during the covid-19 pandemic in indonesia. *Heliyon*, 7(8):e07853.

Arbi Haza Nasution, Yohei Murakami, and Toru Ishida. 2021. Plan optimization to bilingual dictionary induction for low-resource language families. *ACM Transactions Asian Low-Resource Language Information Processing*, 20(2).

Wilhelmina Nekoto, Vukosi Marivate, Tshinondiwa Matsila, Timi Fasubaa, Tajudeen Kolawole, Taiwo Fagbohungbe, Solomon Oluwole Akinola, Shamsuddee Hassan Muhammad, Salomon Kabongo, Salomey Osei, Sackey Freshia, Rubungo Andre Niyongabo, Ricky Macharm, Perez Ogayo, Orevaoghene Ahia, Musie Meressa, Mofe Adeyemi, Masabata Mokgesi-Selinga, Lawrence Okegbemi, Laura Jane Martinus, Kolawole Tajudeen, Kevin Degila, Kelechi Ogueji, Kathleen Siminyu, Julia Kreutzer, Jason Webster, Jamiil Toure Ali, Jade Abbott, Iroro Orife, Ignatius Ezeani, Idris Abdulkabir Dangana, Herman Kamper, Hady Elsahar, Goodness Duru, Ghollah Kioko, Espoir Murhabazi, Elan van Biljon, Daniel Whitenack, Christopher Onyefuluchi, Chris Emezue, Bonaventure Dossou, Blessing Sibanda, Blessing Itoro Bassey, Ayodele Olabiyi, Arshath Ramkilowan, Alp Öktem, Adewale Akinfaderin, and Abdallah Bashir. 2020. Participatory Research for Low-resourced Machine Translation: A Case Study in African Languages. In *Findings of EMNLP 2020*.

Made Nindyatama Nityasya, Haryo Akbarianto Wibowo, Radityo Eko Prasojo, and Alham Fikri Aji. 2021. Costs to consider in adopting NLP for your business.

Hiroki Nomoto, Kenji Okano, David Moeljadi, and Hideo Sawada. 2018. Tufs asian language parallel corpus (talpco). In *Proceedings of the twenty-fourth annual meeting of the Association for Natural Language Processing*, pages 436–439.

Deden Ade Nurdeni, Indra Budi, and Aris Budi Santoso. 2021. Sentiment analysis on covid19 vaccines in indonesia: From the perspective of sinovac and pfizer. In *2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT)*, pages 122–127. IEEE.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Femphy Pisceldo, Rahmad Mahendra, Ruli Manurung, and I Wayan Arka. 2008. A two-level morphological analyser for the Indonesian language. In *Proceedings of the Australasian Language Technology Association Workshop 2008*, pages 142–150, Hobart, Australia.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2362–2376, Online. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Ayu Purwarianti and Ida Ayu Putu Ari Crisdayanti. 2019. Improving bi-lstm performance for indonesian sentiment analysis using paragraph vector. In *2019 International Conference of Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, pages 1–5. IEEE.

Oddy Virgantara Putra, Fathin Muhammad Wasmanson, Triana Harmini, and Shoffin Nahwa Utama. 2020. Sundanese twitter dataset for emotion classification. In *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, pages 391–395. IEEE.

Shofianina Dwi Ananda Putri, Muhammad Okky Ibrohim, and Indra Budi. 2021. Abusive language and hate speech detection for Indonesian-local language in social media text. In *Recent Advances in Information and Communication Technology 2021*, pages 88–98, Cham. Springer International Publishing.

Gowtham Ramesh, Sumanth Doddapaneni, Aravinth Bheemaraj, Mayank Jobanputra, Raghavan AK, Ajitesh Sharma, Sujit Sahoo, Harshita Diddee, Mahalakshmi J, Divyanshu Kakwani, Navneet Kumar, Aswin Pradeep, Srihari Nagaraj, Kumar Deepak, Vivek Raghavan, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh Shantadevi Khapra. 2022. Samanantar: The Largest Publicly Available Parallel Corpora Collection for 11 Indic Languages. *Transactions of the Association for Computational Linguistics*, 10:145–162.

Philip Resnik, Mari Broman Olsen, and Mona Diab. 1999. The bible as a parallel corpus: Annotating the 'book of 2000 tongues'. *Computers and the Humanities*, 33(1):129–153.

Hammam Riza, Michael Purwoadi, Teduh Uliniansyah, Aw Ai Ti, Sharifah Mahani Aljunied, Luong Chi Mai, Vu Tat Thang, Nguyen Phuong Thai, Vichet Chea, Sethserey Sam, et al. 2016. Introduction of the asian language treebank. In *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA)*, pages 1–6. IEEE.

Tommy Wijaya Sagala, Mei Silviana Saputri, Rahmad Mahendra, and Indra Budi. 2020. Stock price movement prediction using technical analysis and sentiment analysis. In *Proceedings of the 2020 2nd Asia Pacific Information Technology Conference*, pages 123–127.

Sakriani Sakti and Satoshi Nakamura. 2013. Towards language preservation: Design and collection of graphemically balanced and parallel speech corpora of indonesian ethnic languages. In *2013 International Conference Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, pages 1–5. IEEE.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

James Neil Sneddon. 2003. *The Indonesian language: Its history and role in modern society*. UNSW Press, Sydney.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Dániel Varga. 2006. The jrc-acquis: A multilingual aligned parallel corpus with 20+ languages. *arXiv preprint cs/0609058*.

Stephanie Strassel and Jennifer Tracey. 2016. Lorelei language packs: Data, tools, and resources for technology development in low resource languages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3273–3280.

Motomitsu Uchibori and Norio Shibata. 1988. Ngaju-Dayak Language. *The Sanseido Encyclopedia of Linguistics: Languages of The World*, 1:1156–1160.

Daan van Esch, Tamar Lucassen, Sebastian Ruder, Isaac Caswell, and Clara E Rivera. 2022. Writing system and speaker metadata for 2,800+ language varieties. In *Proceedings of the 13th Language Resources and Evaluation Conference*, Marseille, France.

Xinyi Wang, Sebastian Ruder, and Graham Neubig. 2022. Expanding pretrained models to thousands more languages via lexicon-based adaptation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 863–877, Dublin, Ireland. Association for Computational Linguistics.

Sukardi Weda. 2016. Syntactic variation of buginese, a language in austronesian great family. *Kongres Internasional Masyarakat Linguistik Indonesia (KIMLI) 2016*, pages 838–841.

Wedhawati Wedhawati, Wiwin E.S.N., Sri Nardiati, Herawati Herawati, Restu Sukesti, Marsono Marsono, Edi Setiyanto, Dirgo Sabariyanto, Syamsul Arifin, Sumadi Sumadi, and Laginem Laginem. 2001. *Tata bahasa Jawa mutakhir*. Badan Pengembangan dan Pembinaan Bahasa.

Haryo Akbarianto Wibowo, Made Nindyatama Nityasya, Afra Feyza Akyürek, Suci Fitriany, Alham Fikri Aji, Radityo Eko Prasojo, and Derry Tanti Wijaya. 2021. IndoCollex: A testbed for morphological transformation of Indonesian word colloquialism. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3170–3183.

Haryo Akbarianto Wibowo, Tatag Aziz Prawiro, Muhammad Ihsan, Alham Fikri Aji, Radityo Eko Prasojo, Rahmad Mahendra, and Suci Fitriany. 2020. Semi-supervised low-resource style transfer of Indonesian informal to formal language with iterative forward-translation. In *2020 International Conference on Asian Language Processing (IALP)*, pages 310–315. IEEE.

Bryan Wilie, Karissa Vincentio, Genta Indra Winata, Samuel Cahyawijaya, Xiaohong Li, Zhi Yuan Lim, Sidik Soleman, Rahmad Mahendra, Pascale Fung, Syafri Bahar, and Ayu Purwarianti. 2020. IndoNLU: Benchmark and resources for evaluating Indonesian natural language understanding. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the*

10th International Joint Conference on Natural Language Processing, pages 843–857, Suzhou, China. Association for Computational Linguistics.

Genta Winata, Shijie Wu, Mayank Kulkarni, Thamar Solorio, and Daniel Preoţiuc-Pietro. 2022. Cross-lingual few-shot learning on unseen languages. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing*, pages 777–791.

Genta Indra Winata, Andrea Madotto, Zhaojiang Lin, Rosanne Liu, Jason Yosinski, and Pascale Fung. 2021. Language models are few-shot multilingual learners. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 1–15.

Wilson Wongso, Henry Lucky, and Derwin Suhartono. 2022. Pre-trained transformer-based language models for sundanese. *Journal of Big Data*, 9(1):1–17.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

A Data Statement for NusaX

A.1 General Information

Dataset title NusaX

Dataset curators Alham Fikri Aji (MBZUAI), Rahmad Mahendra (Universitas Indonesia), Samuel Cahyawijaya (HKUST), Ade Romadhony (Telkom University, Indonesia), Genta Indra Winata (Bloomberg), Fajri Koto (University of Melbourne), Kemal Kurniawan (University of Melbourne)

Dataset version 1.0 (May 2022)

Data statement author Kemal Kurniawan (University of Melbourne)

Data statement version 1.0 (February 2022)

A.2 Executive Summary

NusaX is a multilingual parallel corpus across 10 local languages in Indonesia: Acehnese, Balinese, Banjarese, Buginese, Madurese, Minangkabau, Javanese, Ngaju, Sundanese, and Toba Batak. The data was translated obtained by human translation from Indonesian and human-assisted quality assurance.

A.3 Curation Rationale

The goal of the dataset creation process is to provide gold-standard sentiment analysis corpora for Indonesian local languages. The Indonesian data is sampled from SmSA (Purwarianti and Crisdayanti, 2019), an Indonesian sentiment analysis corpus. SmSA is chosen among other corpora (e.g., HoASA (Azhar et al., 2019) based on (1) the agreement of our manual re-annotation of a small and randomly selected samples and (2) manual inspection to ensure that the topics are diverse. After sampling, the data is edited and/or filtered to remove harmful contents and maintain quality. Several criteria are used in this process:

1. Is the sentiment label correct?
2. Does the sentence contain multiple sentiments?
3. Does the sentence contain harmful content that discriminates against race, religion, or other protected groups?
4. Does the sentence contain an attack toward an individual or is abusive?
5. Is the sentence politically charged?
6. Is the sentence overly Bandung/Sunda-centric?[3]
7. Will the sentence be difficult to translate into local languages?
8. Are there any misspellings?

A.4 Documentation for Source Datasets

NusaX is obtained by translating SmSA (Purwarianti and Crisdayanti, 2019), an Indonesian sentiment analysis dataset.

A.5 Language Variety

NusaX covers a total of 10 local languages spoken in Indonesia (ID) as shown in Table 8.

A.6 Speaker Demographic

The SmSA dataset was obtained from social media and online forums: Twitter, Zomato, TripAdvisor, Facebook, Instagram, Qraved. We can assume the users' age ranges from 25 to 34 years, which is the age range of the majority of Indonesian social media users[4].

A.7 Annotator Demographic

A total of 28 translators are employed in the translation process. All translators are Indonesian and recruited by via either online surveys or personal contacts. They are then selected based on (1) the self-reported fluency in the local language into which they would be translating and (2) the highest education level achieved. Those who (a) are native speakers of or fluent in the target local language and (b) finished at least high school education (id: *SMA/sederajat*) are selected.

Acehnese There are 3 translators for Acehnese, but only 2 of them responded when asked for demographic information. Thus, what follows is the demographic information of only those 2 translators. One has some experience in translation work, while the other does not. One identifies as male, and the other as female. Both are in their 20s. Lastly, one works as a freelancer, while the other is a farmer.

Balinese Three people translate into Balinese. Two of them have previous experience in translation work, and both identify as female. The other one, who identifies as male, does not have such

---

[3] Bandung is the capital city of West Java, in which Sunda is the ethnic group.

[4] https://www.statista.com/statistics/997297/indonesia-breakdown-social-media-users-age-gender/

Language	ISO 639-3	Annotators' Dialect	Example
Acehnese	ace	Banda Aceh	Meureutoh rumoh di Medan keunong ie raya
Balinese	ban	Lowland	Satusan umah ring medan merendem banjir
Toba Batak	bbc	Toba, Humbang	Marratus jabu di medan na hona banji
Banjarese	bjn	Hulu, Kuala	Ratusan rumah di medan tarandam banjir
Buginese	bug	Sidrap	Maddatu bola okko medan nala lempe
Javanese	jav	Matraman	Atusan omah ing medan kebanjiran
Madurese	mad	Situbondo	Ratosan bangko e medan tarendem banjir
Minangkabau	min	Padang, Agam	Ratuihan rumah di medan tarandam banjir
Ngaju	nij	Kapuas, Kahayan	Ratusan huma hong medan lelep awi banjir
Sundanese	sun	Priangan	Ratusan bumi di medan karendem banjir

Table 8: Local languages spoken in Indonesia (ID) that are covered in NusaX.

experience. Two of them are aged 20-29 years old, while the other is in their 30s. Their occupations are university lecturer, school teacher, and civil employee respectively.

Banjarese Two translators are employed for Banjarese, but only one responded when asked for demographic information. The translator has prior experience in translation work, identifies as male, is in his 40s, and works as a university lecturer.

Buginese Buginese is translated by 2 people, but only one responded when asked for demographic information. The person has prior translation experience, identifies as male, is aged 30-39 years old, and runs an Islamic boarding school as a living.

Javanese Four translators are employed for Javanese, but one did not respond when asked for demographic information. The other three have prior experience in translation work. Among them, two identify as female, and one as male. All of them are in their 20s. Two of them are university students, and the other one works as a freelance assistant editor.

Madurese There are 3 translators for Madurese. Only one of them has previous experience in translation work. Two of them identify as female, while the other as male. One person is aged under 20 years old and is a university student. The others are 20-29 years old and work as a school teacher and an employee in a private company respectively.

Minangkabau Three people translate into Minangkabau. Two of them have previous translation experience. All three identify as female and are aged 20-29 years old. They work as a civil employee, a university student, and a senior data annotator respectively.

Ngaju Two translators work on Ngaju, but only one responded when asked for demographic information. The translator has prior experience, identifies as female, is aged no less than 50 years old, and is a stay-at-home mother.

Sundanese There are 5 translators for Sundanese, four of which identify as female, and the other one as male. Three translators are in their 20s, one is younger than 20 years old, and the remaining one is in their 30s. The translators work as a school teacher, a university student, a university lecturer, and the remaining two as employees in a private company.

Toba Batak Three translators are employed for Toba Batak. One has prior translation experience. Two translators identify as male while the other as female. All three are in their 20s. One works for a private company, and the others are university students.

B Hyperparameters

B.1 Sentiment Analysis

Hyperparams	NB	SVM	LR
feature	{BoW, tfidf}	{BoW, tfidf}	{BoW, tfidf}
alpha	(0.001 - 1)	–	–
C	–	(0.01 - 100)	(0.001 - 100)
kernel	–	{rbf, linear}	–

Table 9: Hyperparameters of statistical models on sentiment analysis.

For statistical models, we use a spaCy as our toolkit, and we perform grid-search over the parameter ranges shown in Table 9 and select the

Hyperparams	**Values**
learning rate	[1e-4, 5e-5, **1e-5**, 5e-6, 1e-6]
batch size	[4, 8, 16, **32**]
num epochs	100
early stop	3
max norm	10
optimizer	Adam
Adam β	(0.9, 0.999)
Adam γ	0.9
Adam ϵ	1e-8

Table 10: Hyperparameters of pre-trained LMs on sentiment analysis. **Bold** denotes the best hyperparameter setting.

best performing model over the devset. For all pre-trained LMs, we perform grid-search over batch size and learning rate while keeping the other hyperparameters fixed. The list of hyperparameters is shown in Table 10.

B.2 Machine Translation

Table 11 shows the hyperparameters of deep learning models on machine translation.

Hyperparams	**IndoGPT**	**IndoBARTv2**	**mBART-50**	**mT5$_{\text{BASE}}$**
learning rate	1e-4	1e-4	2e-5	5e-4
batch size	16			
gamma	0.98	0.98	0.98	0.95
max epochs	20			
early stop	10			
seed	{1...5}			

Table 11: Hyperparameters of pretrained LMs on machine translation.

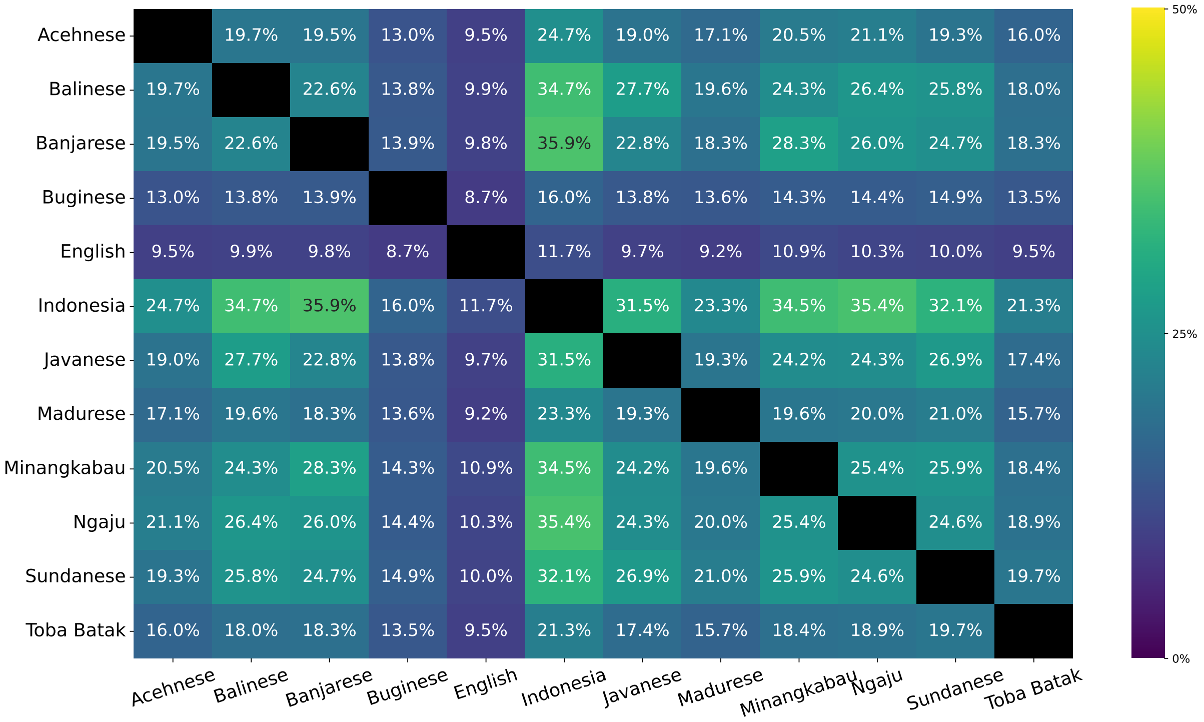
C Dataset Statistics

In this section, we present more detail statistics of our NusaX datasets. To evaluate the difference between each language in the NusaX dataset, we analyze the vocabulary characteristic for each language. We collect the vocabulary for each language by removing all the punctuation in the sentence and tokenize the sentence with the spaCy tokenizer. [5] We show the vocabulary size and the top-10 words for each language on Table 12, and the vocabulary histogram for each language in Figure 4. We can see that the most common words between Indonesian and other local languages vary a lot, despite having a similar vocabulary size and histogram pattern. This shows the intuitive difference between Indonesian and local languages in Indonesia.

We further measure the vocabulary overlap over different language pairs. We measure the vocabulary overlap for each pair of languages by measuring the intersection over union (IoU) of the two vocabularies. We show the vocabulary overlap in Figure 3. From the results, we can conclude that English has the smallest vocabulary overlap with the other languages. This makes sense since English comes from a different language family, i.e., Indo-European language under the Germanic language branch, while the others are from the Austronesian language family under the Malayo-Polynesian branch. Other languages that have low vocabulary overlap are Buginese (bug) and Toba Batak (bbc). This aligns with our discussion in §5, which shows the distinction between these languages and the other languages in the NusaX dataset.

---

[5] `https://github.com/explosion/spaCy`



	Acehnese	Balinese	Banjarese	Buginese	English	Indonesia	Javanese	Madurese	Minangkabau	Ngaju	Sundanese	Toba Batak
Acehnese		19.7%	19.5%	13.0%	9.5%	24.7%	19.0%	17.1%	20.5%	21.1%	19.3%	16.0%
Balinese	19.7%		22.6%	13.8%	9.9%	34.7%	27.7%	19.6%	24.3%	26.4%	25.8%	18.0%
Banjarese	19.5%	22.6%		13.9%	9.8%	35.9%	22.8%	18.3%	28.3%	26.0%	24.7%	18.3%
Buginese	13.0%	13.8%	13.9%		8.7%	16.0%	13.8%	13.6%	14.3%	14.4%	14.9%	13.5%
English	9.5%	9.9%	9.8%	8.7%		11.7%	9.7%	9.2%	10.9%	10.3%	10.0%	9.5%
Indonesia	24.7%	34.7%	35.9%	16.0%	11.7%		31.5%	23.3%	34.5%	35.4%	32.1%	21.3%
Javanese	19.0%	27.7%	22.8%	13.8%	9.7%	31.5%		19.3%	24.2%	24.3%	26.9%	17.4%
Madurese	17.1%	19.6%	18.3%	13.6%	9.2%	23.3%	19.3%		19.6%	20.0%	21.0%	15.7%
Minangkabau	20.5%	24.3%	28.3%	14.3%	10.9%	34.5%	24.2%	19.6%		25.4%	25.9%	18.4%
Ngaju	21.1%	26.4%	26.0%	14.4%	10.3%	35.4%	24.3%	20.0%	25.4%		24.6%	18.9%
Sundanese	19.3%	25.8%	24.7%	14.9%	10.0%	32.1%	26.9%	21.0%	25.9%	24.6%		19.7%
Toba Batak	16.0%	18.0%	18.3%	13.5%	9.5%	21.3%	17.4%	15.7%	18.4%	18.9%	19.7%	

Figure 3: Vocabulary overlap between language pairs in NusaX dataset.

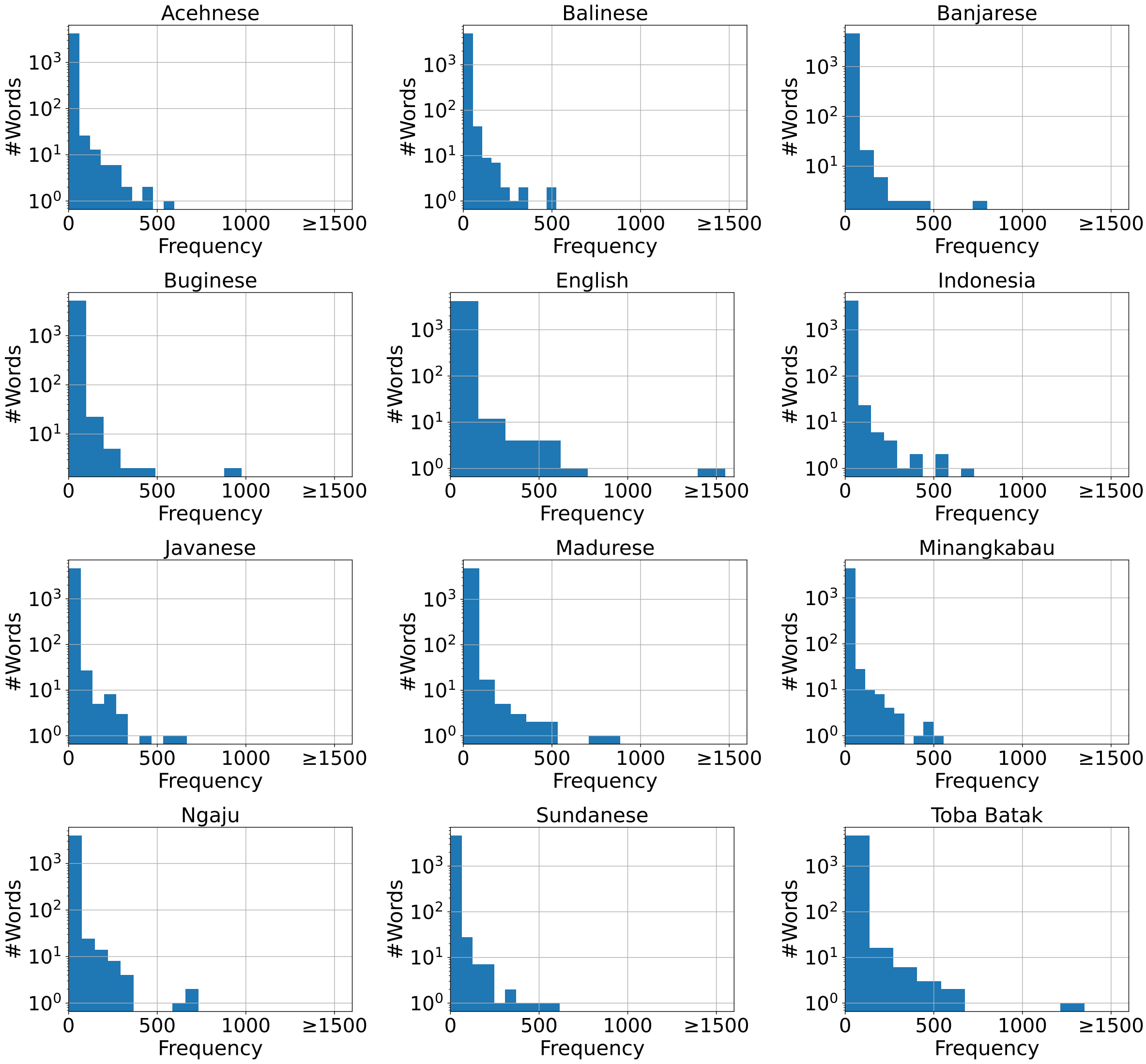


Figure 4: Word frequency histogram for each language in NusaX.

Toba Batak (4681 words)	**Balinese (4927 words)**	**Banjarese 4631 words**
na	lan	nang
di	sane	wan
dohot	ring	di
ni	ane	kada
tu	ne	ulun
do	sajan	nyaman
dang	tiyang	gasan
pe	tiang	banar
tabo	ajak	makan

Minangkabau (446 words)	**English (4233 words)**	**Ngaju (4005 words)**
di	the	te
nan	and	dengan
dan	to	ji
jo	is	mangat
untuak	a	eka
awak	of	akan
yang	for	aku
lamak	in	jadi
ka	I	diak

Sundanese (4693 words)	**Buginese (5118 words)**	**Indonesian (4269 words)**
nu	e	yang
sareng	na	di
di	okko	dan
teu	sibawa	tidak
pisan	iya	saya
abdi	de	dengan
ka	i	ini
ieu	ko	enak
aya	ladde	untuk

Javanese (4719 words)	**Acehnese (4250 words)**	**Maduranese (4846 words)**
sing	nyang	se
lan	ngon	e
ora	hana	bik
karo	lon	engkok
aku	that	ben
ing	mangat	tak
iki	nyoe	nyaman
ning	dan	ka
enak	bak	ghebey

Table 12: Vocabulary size (in bracket) and top-10 words on each language in the NusaX dataset.

The Functional Relevance of Probed Information: A Case Study

Michael Hanna[*], **Roberto Zamparelli**[†], and **David Mareček**[§]
[*]ILLC, University of Amsterdam, The Netherlands: `m.w.hanna@uva.nl`
[†]CIMeC, University of Trento, Italy: `roberto.zamparelli@unitn.it`
[§]MFF ÚFAL, Charles University, Czechia: `marecek@ufal.mff.cuni.cz`

Abstract

Recent studies have shown that transformer models like BERT rely on number information encoded in their representations of sentences' subjects and head verbs when performing subject-verb agreement. However, probing experiments suggest that subject number is also encoded in the representations of all words in such sentences. In this paper, we use causal interventions to show that BERT only uses the subject plurality information encoded in its representations of the subject and words that agree with it in number. We also demonstrate that current probing metrics are unable to determine which words' representations contain functionally relevant information. This both provides a revised view of subject-verb agreement in language models, and suggests potential pitfalls for current probe usage and evaluation.

1 Introduction

The phenomenon of subject-verb agreement has received significant attention from the NLP community. In English, this phenomenon is very simple: present tense verbs must agree in number with their subject noun, which is either singular or plural. In the present tense, verbs that agree with 3rd-person singular nouns receive one verb conjugation, generally ending in "-s"; in all other cases, the bare form of the verb is used.

The simplicity of this phenomenon, combined with the potential for long-distance subject and verb agreement across intervening adverbs or relative clauses (e.g. "The friend [that probably called my parents] is…") has made it the object of intense study in humans (Vigliocco et al. (1995) and Franck et al. (2010), *inter alia*). It also prompted early investigations on the ability of language models to capture it (left-to-right LSTMs in Linzen et al. (2016); Gulordava et al. (2018)). More recently, the popular pre-trained model BERT (Devlin et al., 2019) has been shown to be relatively proficient at subject-verb agreement (Goldberg, 2019), although these abilities depend somewhat on verb frequency and lexical patterns (Newman et al., 2021; Lasri et al., 2022a).

Other studies ask not how models behave (e.g. with respect to subject-verb agreement), but how models' representations support this behavior. Such studies often use probing, a technique in which an auxiliary classifier (probe) is trained to extract some property of words or sentences from models' internal representations thereof (Belinkov and Glass, 2019; Belinkov, 2022). If the probe can extract the property with high accuracy, one concludes that the model has encoded the property in the word's representation. Klafka and Ettinger (2020) discover that probes can extract the plurality of a sentence's subject from last-layer BERT representations of any word in the sentence.

However, probing has received criticism because the information discovered by probes is not always used by models (Ravichander et al., 2021). Causal interventions have been proposed as a means of connecting models' internal representations to their external behavior. Such techniques make targeted changes to models' representations, and observe how model behavior changes, in order to establish a causal connection between the two (Ravfogel et al., 2021; Geiger et al., 2021). For example, Ravfogel et al. (2020) remove gender information from models' noun representations, and observe that models behave as if they do not know the nouns' gender.

Recently, Lasri et al. (2022b) unified these lines of work by using causal probing interventions to investigate subject-verb agreement. They did so by first training probes to predict subject number information from representations of verbs and their subjects. Then, they removed subject number information from the representations of subjects and verbs. This caused BERT to make errors on a subject-verb agreement task, indicating that the probes discovered functionally relevant informa-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 835–848
May 2-6, 2023 ©2023 Association for Computational Linguistics

tion about subject number in subject and verb representations. However, this leaves open the question of whether models use the subject number information in the representations of other words of the sentence, found by Klafka and Ettinger (2020).

In this paper, we focus on these questions: do large language models (LLMs) like BERT rely on subject number information stored outside of subject and verb representations when performing subject-verb agreement? Moreover, do current probing evaluation metrics allow us to determine which information is used and which is not? Our goal is thus twofold: first, to clarify how subject-verb agreement occurs in LLMs, and second, to determine if any current non-causal probing metrics can determine which probes have found functionally relevant information.

We achieve these goals as follows. First, we adopt the setup of Klafka and Ettinger (2020), examining simple sentences of a fixed structure and length. We use probing to demonstrate that subject number information is extractable from representations of any word in the sentence, at most layers. Next, we use causal interventions to show that although subject number information exists in the representations of all words of the sentence, it is not always used; rather, it seems that BERT uses information stored in words that agree with the subject in number. Finally, we evaluate our probes using modern probe evaluation metrics, to ascertain if any metric can determine whether subject number information found by a probe is used by the model or not. We find that these metrics are unable to do this.[1]

2 Probing for Plurality

To determine if the subject number information that is contained in all words of a sentence is used, we first verify that all words' representations contain said information. We adopt the setup of Klafka and Ettinger (2020), and thus investigate simple sentences with a fixed structure, using a synthetic English-language dataset. Their original dataset contains sentences of the form "The [subject] [verb-past] the [object]". However, we create our own dataset, which contains not only subject-verb agreement but also article-subject agreement. It consists of sentences of the form "[This / these] [adjective] [subject] [adverb] [verb-present] the [object]", e.g. "This short boy definitely admires the firefighter." The dataset contains 6000 sentences, with roughly equal numbers of singular and plural subjects. We split our dataset into train, valid, and test splits containing 4000/1000/1000 examples.

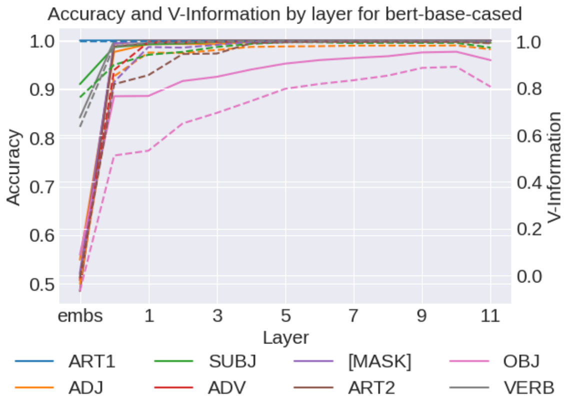


Figure 1: Test accuracy (left, solid) and $\mathcal{V}$-information (right, dashed) of probes of a given word and layer

Having generated a dataset, we must then define the model representations to be probed for subject number information. Klafka and Ettinger (2020) find said information in all words' final-layer BERT representations; in contrast, we examine the representations generated as the output of the entire transformer block at *each* layer, of a variety of LLMs. We also consider representations from the embedding, which have by definition no contextual information (except for positional information). Concretely, we analyze the base, large, and distilled variants of BERT and RoBERTa (Devlin et al., 2019; Liu et al., 2019; Sanh et al., 2019). However, we present only the BERT-base representations unless discussing another model, as results are similar across all models analyzed.

Having extracted model representations, we train probes on the subject number prediction task. Each probe is specific to a word in the sentence and to a model layer; a given probe might predict the number of a sentence's subject given the 5th-layer representation of the sentence's verb. Each probe is a linear layer with a sigmoid activation. We use HuggingFace (Wolf et al., 2020) implementations of these models; for precise model names, and dataset and training details, see Appendices A and B.

For each probe, we record 3 metrics, averaged across the dataset: (i) test accuracy; (ii) $\mathcal{V}$-information (Xu et al., 2020), and (iii) codelength, as measured by online minimum description length (MDL) probing (Voita and Titov, 2020). We record

---

[1] The code for all experiments is available at `https://github.com/hannamw/probed-information`.

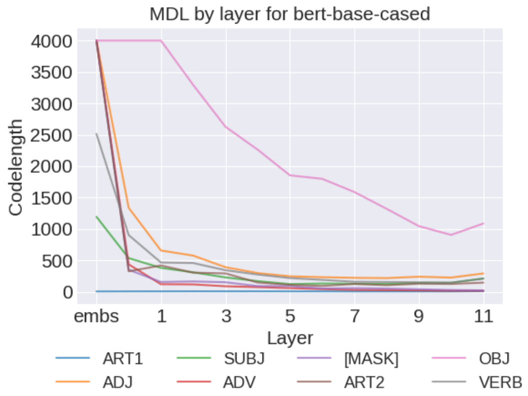


Figure 2: Probe codelength for a given word and layer

the latter two metrics to see if recently-proposed metrics can distinguish functionally relevant from irrelevant information in representations. We report metrics averaged over 10 probe training runs.

$\mathcal{V}$-information measures, given a family $\mathcal{V}$ of probes used to extract a label from a representation, how much those probes benefit from receiving a representation as input, as opposed to no input at all. In contrast, online MDL probing measures codelength by retraining probes on subsets of the dataset of increasing size. If probes assign low entropy to a dataset when training data is limited, the information being probed for is easily accessible from the input representations. A detailed description of these metrics can be found in Appendix C.

Results Figure 1 shows the accuracy and $\mathcal{V}$-information of our probes (higher is better). Note that for all figures, 'ART1' refers to the subject's article, and 'ART2' to the object's. We also probe the representation of masked verbs ('[MASK]'), discussed in the following section. As in Klafka and Ettinger (2020), accuracy is high for all words in later BERT layers. Unsurprisingly, accuracy is also high for subjects, their articles, and unmasked verbs using just BERT's embeddings. But starting in layer 0, when information is first able to move between positions, accuracy is high for most other words in the sentence as well. $\mathcal{V}$-information tracks accuracy; this is unsurprising, as it combines entropy, which tracks with accuracy, and a model-family-specific baseline, which is the same for all words / layers. This indicates that subject number information is present in the representations of all words of the sentence, and could be used for subject-verb agreement.

Figure 2 shows the codelength in bits for each probe, by layer; lower codelength means probes could learn to extract subject number information given less data. Broad trends are similar to those from accuracy and $\mathcal{V}$-information. Before layer 0, codelength given by the unmasked verb, subject, and subject article probes is notably lower than in others; this makes sense, because the very form of these words indicates the number of the subject. However, the plurality of the subject quickly becomes available, and the codelength for other words' probes drops. By the last layer, the disparity between words that agree in number with the subject, and those that do not, has mostly disappeared; the exception is the object probe, whose codelength remains high.

3 Causal Interventions

We now apply causal interventions to test if the information found by probes in the prior experiment is actually used by BERT[2]. In our case, this means we will alter BERT's internal representations with respect to subject number information, and observe BERT's performance on a subject-verb agreement task. In order to perform such interventions, we alter our dataset to accommodate this task: we mask out the verb of the sentence, and task BERT with predicting this masked word.

Then, we apply causal interventions, specifically reflection (Ravfogel et al., 2021) and interchange (Geiger et al., 2021) interventions. To define these, consider the case where BERT's input $\mathbf{s}$ is "This short boy definitely [MASK] the firefighter.". In the middle of computation, we extract $\mathbf{z}$, BERT's nth-layer representation of our word of interest; for example, the sentence's object ("firefighter"). $\mathbf{z}$ serves as input to a linear probe trained on the full training set, defined as $h(\mathbf{x}) = \sigma\left(\mathbf{W}^\top \mathbf{x} + \mathbf{b}\right)$. Let $h(\mathbf{z}) < 0.5$, indicating that the probe predicts the subject of this sentence is singular. In the **reflection** intervention, we reflect $\mathbf{z}$ over the decision boundary of h (a hyperplane defined by $(\mathbf{W}, \mathbf{b})$), creating $\mathbf{z}_r$. By definition, $h(\mathbf{z}_r) > 0.5$; the probe now predicts that the sentence's subject is plural. We replace the original representation $\mathbf{z}$ in the BERT with $\mathbf{z}_r$, and observe how BERT's output changes. In contrast, in the **interchange** intervention, we do not use the probe. Define $\mathbf{z}_i$ as the same representation as $\mathbf{z}$, but taken from the opposite-plurality context, e.g. the n-th layer representation of "firefighter" in "These short boys definitely [MASK]

---

[2] In this section, we discuss only BERT, but the procedure is identical for all models tested.

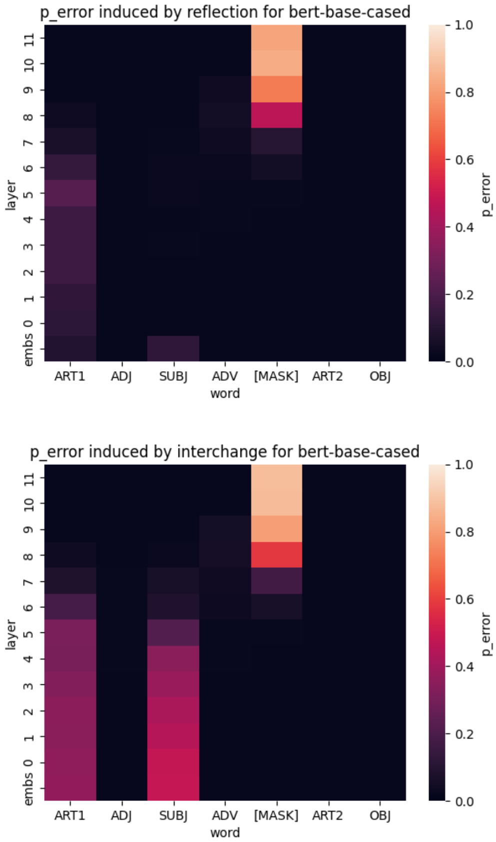


Figure 3: p_{error} induced by intervening on a representation, by word (x-axis), and layer (y-axis). Top: reflection intervention; bottom: interchange intervention

the firefighter." We then run BERT on $\mathbf{s}$, but replace $\mathbf{z}$ with $\mathbf{z}_i$, and observe how its behavior changes.

In both cases, we attempt to reverse the plurality of the subject number information in $\mathbf{z}$ by replacing it with $\mathbf{z}'$. Thus, if the intervention was effective (i.e. BERT was using said information in $\mathbf{z}$), BERT should predict verbs that do not agree in number with the original subject of the sentence.

We perform interventions on examples from the test split. Each intervention targets one word at a specific layer. To measure the effectiveness of a given intervention, we record p_{error}, the probability mass assigned by BERT to all 3rd-person present-tense verbs in its vocabulary that do not agree in number with the sentence's original subject. Higher p_{error} indicates a more successful intervention, i.e. that BERT relied on information in the targeted word representation to perform subject-verb agreement.

Results Figure 3 shows, for each word and layer of BERT, the error induced by performing either a reflection or interchange intervention on that word's representation at the given layer, averaged across the test split. For reflection interventions, we previously trained 10 sets of probes, and thus also average over the results obtained for each set. Note that with no interventions, $p_{error} \approx 0$; BERT assigns 80% of its probability mass to verbs that agree with the subject in number (BERT sometimes predicts conjunctions or other words that neither agree nor disagree with the sentence's subject).

The results show clearly that not all subject number information in a sentence is used. Information stored in representations of the adjective, object, and object's article, is not used at all, producing no effects when either intervention is performed. On the other hand, considering only the interchange intervention, information from the subject (in early layers) and masked verb (in later layers) is heavily used, as reported by Lasri et al. (2022b). Moreover, there is minor usage of number information ($p_{error} \approx 0.05$) in the adverb in layers 9-10, the transition layers between subject and verb. Although the error induced is very small, this could hint at instances where BERT's subject-verb agreement processing diverge from our expectations.

Our setup also reveals a new phenomenon: BERT uses the number information in the representation of the subject's article (a demonstrative, which agrees with the subject). The information in the subject article representation is used in the same layers as the subject information, with slightly weaker intervention effects. On this basis, we conclude that BERT uses subject number information in the subject and words that agree with it (not just the masked verb); however, further study, with additional agreement effects, would strengthen this.

These results are strengthened by the reflection intervention, which yields only slightly different results. It produces the same effects as the interchange at the masked verb and subject's article, albeit at a lower magnitude, indicating that the information found by probes is indeed the information the model is using to perform subject-verb agreement. Moreover, the strong effects at the masked verb position indicate that the model may be encoding subject number linearly. While linearity in BERT's representations of subject number was also found by (Lasri et al., 2022b), they used a much more complex approach, iteratively projecting representations into classifiers' null space; in contrast, our reflection approach is non-iterative, and suggests the model may be encoding this information in just one dimension.

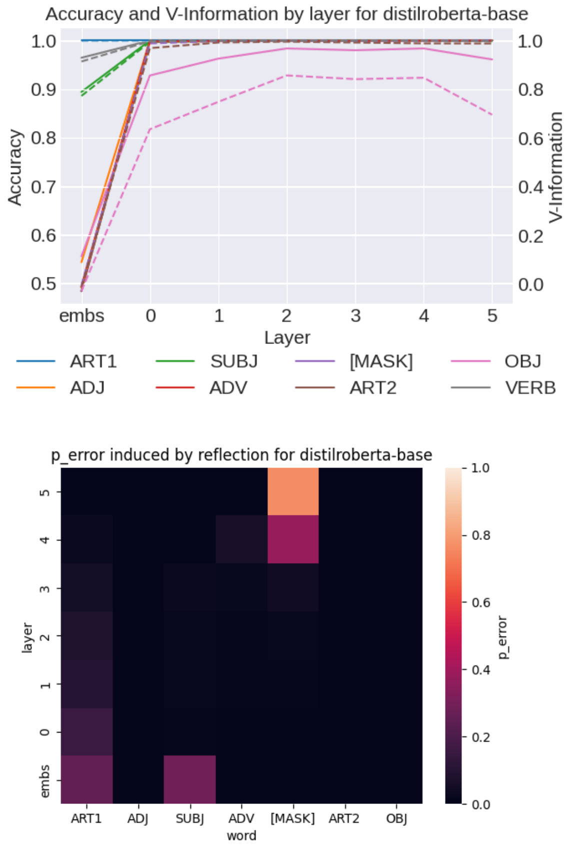


Figure 4: Top: Accuracy and $\mathcal{V}$-information for distilroberta-base. Bottom: effects of reflection interventions on distilroberta-base

It is notable that the reflection intervention produces almost no effect when performed on the subject. As this effect does appear under the interchange intervention, this result likely does not reflect a non-reliance on subject information there encoded; rather, it is due to specific properties of this intervention. First, the information encoded in the subject may well be used, but simply not encoded as the probe found it. Second, the subject is often split into multiple tokens; while we apply the reflection to each token, reflecting multiple tokens representations might not be as effective, as multiple tokens of the same word might not share the same plurality boundary as that of single tokens.

Although we primarily discuss results for BERT, results generalize to the RoBERTa models and their large / distilled variants: interchange interventions show use of information in representations of the subject, its article, and the verb, at expected locations (see Appendix D for plots and heatmaps). This could be surprising, especially in the case of the distilled models. If one views the unused subject number information present in model representations as extraneous or irrelevant encoded information, one might expect that these smaller distilled models do not have the space to encode irrelevant information. However, this is not the case, as seen in Figure 4; probe achieve high accuracies even when the their information is not used.

4 Discussion and Conclusions

In our prior two experiments, we showed that subject number information is extractable from the representation of any word in our simple sentences; however, it is only used if it comes from a representation of the subject or a word that agrees with it. Could any existing metrics have warned us of this, without requiring causal interventions?

Accuracy is insufficient to distinguish functional relevant information; probes extract functionally irrelevant subject number information with high accuracy. The same is true for $\mathcal{V}$-information, for the same reasons; high $\mathcal{V}$-information is necessary but not sufficient for a property to be used by models.

This leaves codelength, which measures ease of extracting the property from the data. It is an appealing hypothesis that models might encode functionally relevant information more accessibly in their representations; if this were the case, MDL probing could detect functionally relevant information. Indeed in early layers, it distinguishes between words that directly reflect subject number (the verb, subject, its article) and those that do not. However, when we mask the verb, such that subject number cannot be determined from the verb's form, but its representation's subject number information is still functionally relevant to subject-verb agreement, this distinction is lost. Thus, MDL probing seems unable to determine functional relevance.

So, we conclude the following. First, we show via probing that subject number information is present in representations of all words of our simple sentences. Then, using causal interventions, we show that only in the subject and words that agree in number with it, is said information functionally relevant. This indicates, as previously shown, that probing is not a reliable method for understanding how models function. Moreover, a way of distinguishing between functionally relevant and irrelevant information in model representations remains elusive. For now, causal interventions remain the most promising way to make this determination.

Limitations

This study is limited to a single phenomenon (number agreement), in a morphologically poor language (English). Our conclusions should be checked against different domains (person/gender agreement) and in morphologically complex languages. In particular, the study of sentences in which agreement extends beyond a demonstrative, the subject, and the verb, could help determine the extent to which models rely on agreement information in each word whose form expresses it. This study is also limited by its use of small, simple, synthetic data; expanding to real-world data, or data that follows a less rigid template, would strengthen our conclusions. From a modeling point of view, these results do generalize to various different masked language models, but this study does not investigate larger, more modern language models; autoregressive language models are also excluded. Thus, it is unclear to what extent these popular models exhibit the phenomenon studied.

Acknowledgments

MH received travel support from ELISE - the European Network of AI Excellence Centers - to present this work at the 2022 ELLIS Doctoral Symposium. This work has been supported by the grant 23-06912S of the Czech Science Foundation.

References

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

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

Julie Franck, Gabriela Soare, Ulrich H. Frauenfelder, and Luigi Rizzi. 2010. Object interference in subject–verb agreement: The role of intermediate traces of movement. *Journal of memory and language*, 62(2):166–182. ID: unige:31707.

Atticus Geiger, Hanson Lu, Thomas F Icard, and Christopher Potts. 2021. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *ArXiv*, abs/1901.05287.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Sarah M Harmon. 2017. *Narrative Encoding for Computational Reasoning and Adaptation*. Ph.D. thesis, University of California, Santa Cruz.

Josef Klafka and Allyson Ettinger. 2020. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4801–4811, Online. Association for Computational Linguistics.

Karim Lasri, Alessandro Lenci, and Thierry Poibeau. 2022a. Does BERT really agree ? fine-grained analysis of lexical dependence on a syntactic task. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2309–2315, Dublin, Ireland. Association for Computational Linguistics.

Karim Lasri, Tiago Pimentel, Alessandro Lenci, Thierry Poibeau, and Ryan Cotterell. 2022b. Probing for the usage of grammatical number. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8818–8831, Dublin, Ireland. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Benjamin Newman, Kai-Siang Ang, Julia Gong, and John Hewitt. 2021. Refining targeted syntactic evaluation of language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3710–3723, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256, Online. Association for Computational Linguistics.

Shauli Ravfogel, Grusha Prasad, Tal Linzen, and Yoav Goldberg. 2021. Counterfactual interventions reveal the causal effect of relative clause representations on agreement prediction. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 194–209, Online. Association for Computational Linguistics.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2021. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3363–3377, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Gabriella Vigliocco, Brian Butterworth, and Carlo Semenza. 1995. Constructing subject-verb agreement in speech: The role of semantic and morphological factors. *Journal of Memory and Language*, 34(2):186–215.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. 2020. A theory of usable information under computational constraints. In *International Conference on Learning Representations*.

A Dataset Details

Our dataset is a tightly-controlled synthetic dataset created using the BLiMP sentence generator (Warstadt et al., 2020), which we altered and updated to fit our use-case. A benefit of using this specific generator is that it forms (relatively) coherent sentences, such that e.g. all subjects and objects are valid (if not especially plausible) arguments of their verb. Moreover, it also allows us to specify additional constraints, and sample an arbitrary number of sentences fulfilling them.

Each example in the dataset consists of a sentence of the form "[This / these / that / those] [adjective] [subject] [adverb] [verb-present] the [object]"; we also automatically generate the same sentence, but with an opposite-number subject (and corresponding verb / article) for use with interchange interventions. At intervention time, we append (but do not probe / investigate) the word "nowadays". This encourages BERT to output a present-tense verb for the masked token; otherwise, BERT often predicts past-tense verbs or conjunctions. Subjects have distinct singular and plural forms, where the plural form ends in "-s".

Each of the word types contained in brackets is always one word long (so there are no multi-word subjects, or phrasal verbs). If a word is split into multiple tokens, we handle it in the following way. During the probe training phase, we consider each token as a separate training example (so the number of training examples for each probe might differ, and be slightly greater than the number of sentences in the dataset). Then, during the intervention phase, we simply apply the intervention to every token composing the word. For the interchange intervention, both the original word and its opposite-number counterpart must have the same number of tokens. Thus, we discard from our dataset any examples where the two have different token lengths. In order to determine whether the model's prediction is a present-tense verb, and whether it was conjugated in the singular or plural, we use the `nodebox_linguistics_extended` package (Harmon, 2017).

B Experimental Details

Probes are scikit-learn (Pedregosa et al., 2011) `LogisticRegression` models, trained with L2 normalization to predict the number (singular / plural) of the subject. For some experiments, we use the weights from these models as the weights for identical PyTorch (Paszke et al., 2019) models, consisting of linear layer with bias and sigmoid activation. The `bert-base-cased` model has approximately 110 million parameters (Devlin et al., 2019). Each probe has 1538 parameters, for a total of 140000 parameters summed over all probes. All experiments were performed using an NVIDIA A100 GPU, and take no more than 24 GPU hours to run in total. We study `bert-base-cased`, `bert-large-cased`, `distilbert-base-cased`, `roberta-base`, `roberta-large`, and `distilroberta-base`.

C Metric Details

We consider two metrics in addition to standard accuracy. The first, predictive $\mathcal{V}$-information (Xu et al., 2020), measures the degree to which information in representations is made available to a chosen model family $\mathcal{V}$. We study linear probes f, where $f(x)$ is the probability that the subject of the sentence from which the representation x comes, is plural. For the purposes of $\mathcal{V}$-information, however, it is useful to define $f'(y|x) = (1-y)(1-f(x)) + yf(x)$, and $\mathcal{V}$ as the family of such functions.

$\mathcal{V}$-information relies on conditional $\mathcal{V}$-entropy:

$$H_{\mathcal{V}}(Y|X) = \inf_{f' \in \mathcal{V}} \mathbb{E}_{x,y \sim \mathcal{D}} - \log f'(y|x)$$

where $\mathcal{D}$ is our dataset of representation, number label pairs. We can define baseline $\mathcal{V}$-entropy as

$$H_{\mathcal{V}}(Y|\emptyset) = \inf_{f' \in \mathcal{V}} \mathbb{E}_{y \sim Y\mathcal{D}} - \log f'(y|\emptyset)$$

where $\emptyset$ represents the absence of representational information, e.g. a constant vector. Now we can define predictive $\mathcal{V}$-information as

$$I(\emptyset \to X) = H_{\mathcal{V}}(Y|\emptyset) - H_{\mathcal{V}}(Y|X)$$

i.e. the predictive advantage given to models of the family $\mathcal{V}$' by representational information. Put simply, this metric asks how much better our probes can extract subject number given representational information, as opposed to if they had none.

The conditional $\mathcal{V}$-information can be estimated via probing; it is the entropy achieved by a probe trained on our dataset. We compute the baseline as the maximum likelihood estimate of the label distribution, i.e. $f'(y|\emptyset) = c(y)/(c(0) + c(1))$, where $y \in \{0, 1\}$ is a label, and $c(y)$ is its count.

We also use minimum description length (MDL) probing, described by Voita and Titov (2020). Their codelength metric rewards probes that are both high-performing and simple. We measure it using their online MDL method, which measures the cost of encoding a dataset's labels with probes trained on limited data. In this setup, we have a dataset of data X and binary labels Y; we begin by partitioning this dataset into groups $(x_{1:t_1}, y_{1:t_1}); (x_{t_1+1:t_2}, y_{t_1+1:t_2}); \ldots; (x_{t_{S-1}+1:t_S}, y_{t_{S-1}+1:t_S})$ of increasing size. We encode the first group of labels $y_{1:t_1}$ at full cost, t_1 bits. For each timestep $i = 1, \ldots, S-1$, we train a probe p_{θ_i} on $\{(x_j, y_j)\}_{j=1}^{i}$, and encode the next set of labels using the updated model, for a cost of $-\log p_{\theta_i}(y_{t_i+1:t_{i+1}}|x_{t_i+1:t_{i+1}})$. The online codelength is, in bits

$$L = t_1 + \sum_{i=1}^{S-1} - \log p_{\theta_i}(y_{t_i+1:t_{i+1}}|x_{t_i+1:t_{i+1}}).$$

In this setup, probes that quickly learn how to extract subject plurality from BERT representations will have a shorter codelength. So, comparing codelength across representations from different words will tell us the words that are easier or more difficult to extract plurality information from.

We follow this procedure in our analysis: we split our data into partitions; then, we repeatedly train probes on increasing portions of our dataset, until we have trained on all data. Probes are trained using scikit-learn's (Pedregosa et al., 2011) `LogisticRegression`, with L2 normalization. After each round of training, we compute the cost of encoding the next partition. We sum over all partitions, and repeat this process 10 times, reporting average codelength. The partition sizes we use are identical to those in Voita and Titov (2020), i.e. 0.1%, 0.2%, 0.4%, 0.8%, 1.6%, 3.2%, 6.25%, 12.5%, 25%, 50%, and 100% of the dataset (cumulatively). In some cases (e.g. when training on embeddings that contain no subject plurality information), MDL is very high; in this case we cap it at 4000 (the cost of transmitting the labels uncompressed, i.e. 4000 examples, at 1 bit / example).

D Results for All Models

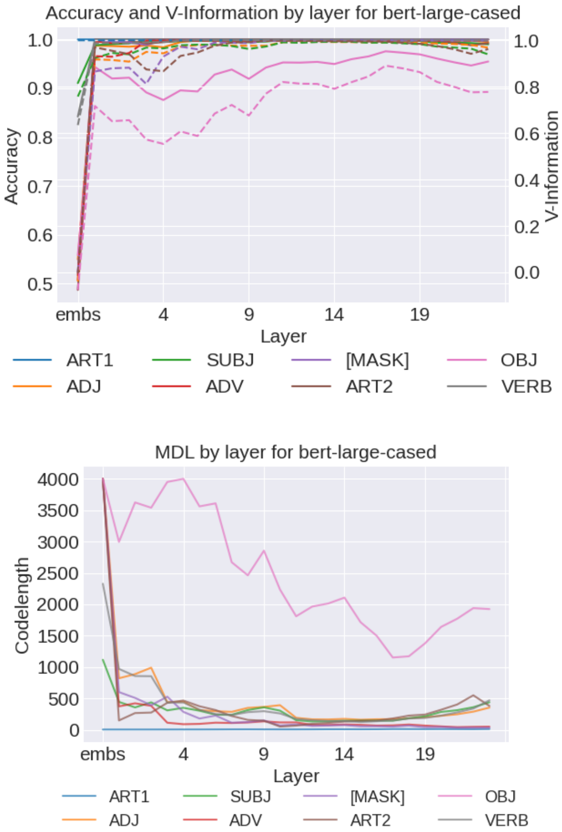


Figure 5: Top: Accuracy and V-information; bottom: MDL codelength

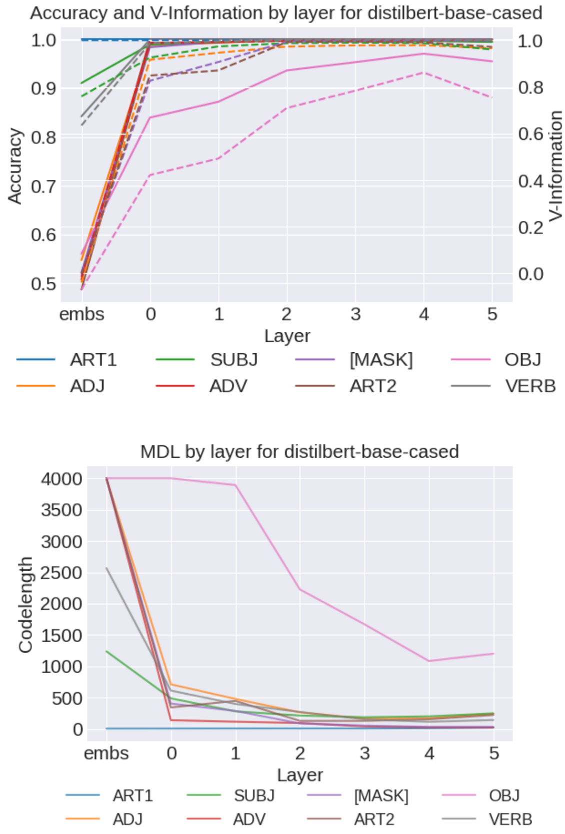


Figure 6: Top: Accuracy and V-information; bottom: MDL codelength

Figure 7: Top: Accuracy and V-information; bottom: MDL codelength

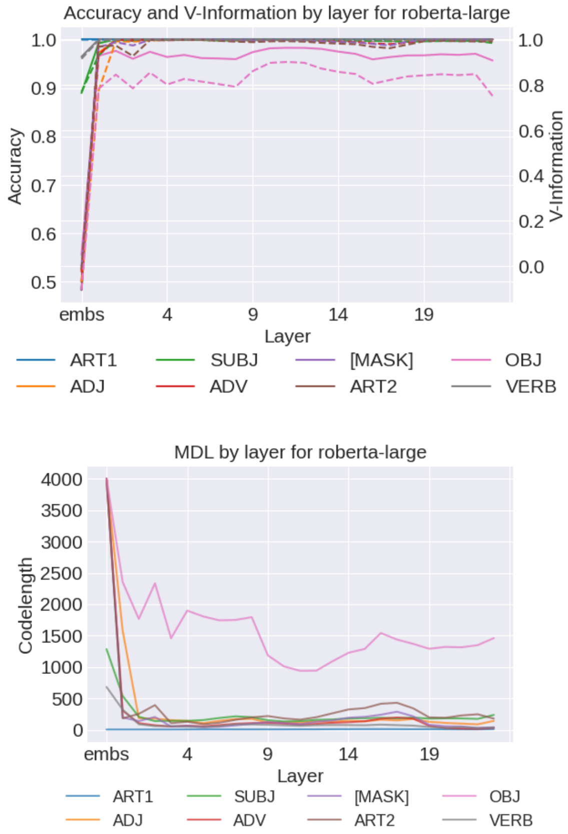


Figure 8: Top: Accuracy and V-information; bottom: MDL codelength

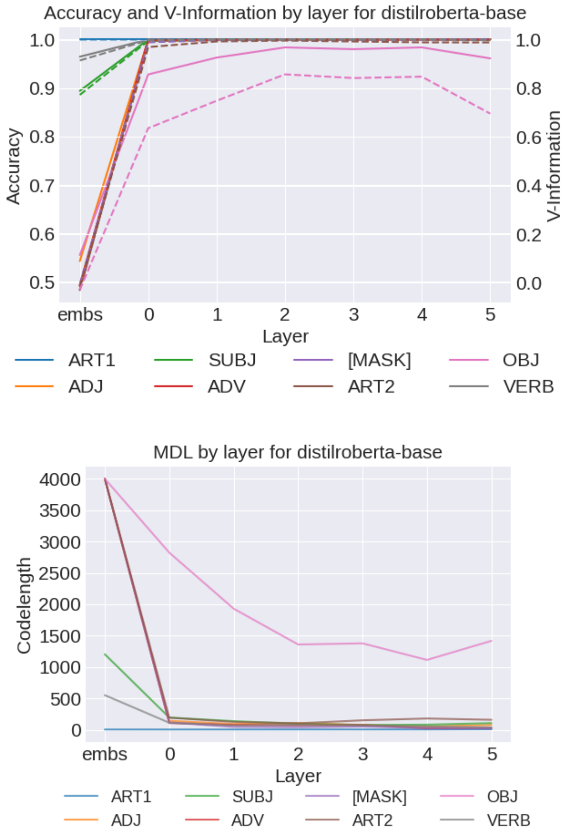


Figure 9: Top: Accuracy and V-information; bottom: MDL codelength

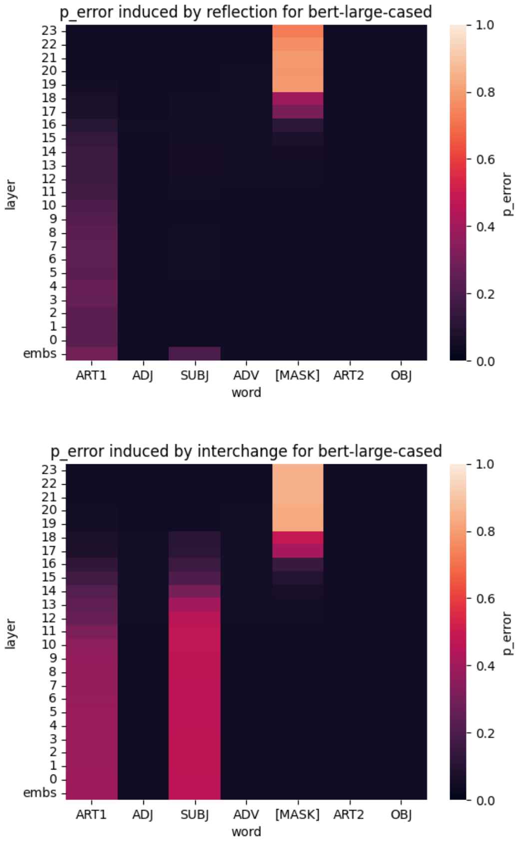


Figure 10: p_{error} induced by intervening on a representation, by word (x-axis), and layer (y-axis). Top: reflection intervention; bottom: interchange intervention

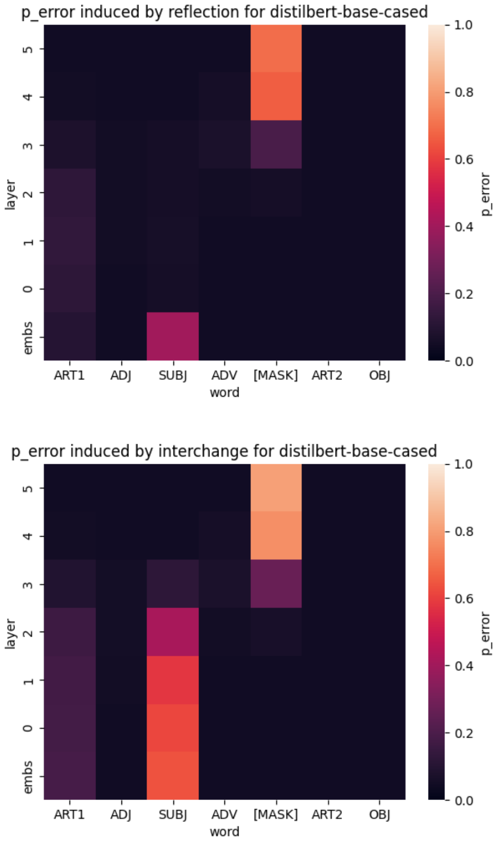


Figure 11: p_{error} induced by intervening on a representation, by word (x-axis), and layer (y-axis). Top: reflection intervention; bottom: interchange intervention

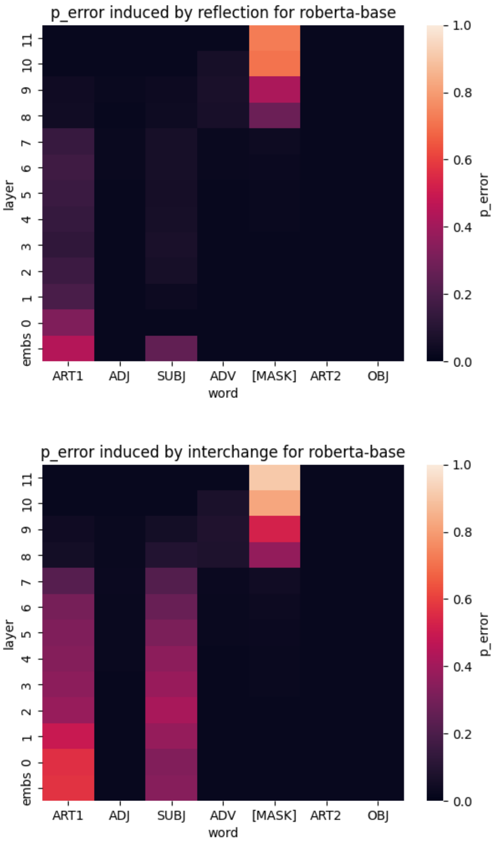


Figure 12: p_{error} induced by intervening on a representation, by word (x-axis), and layer (y-axis). Top: reflection intervention; bottom: interchange intervention

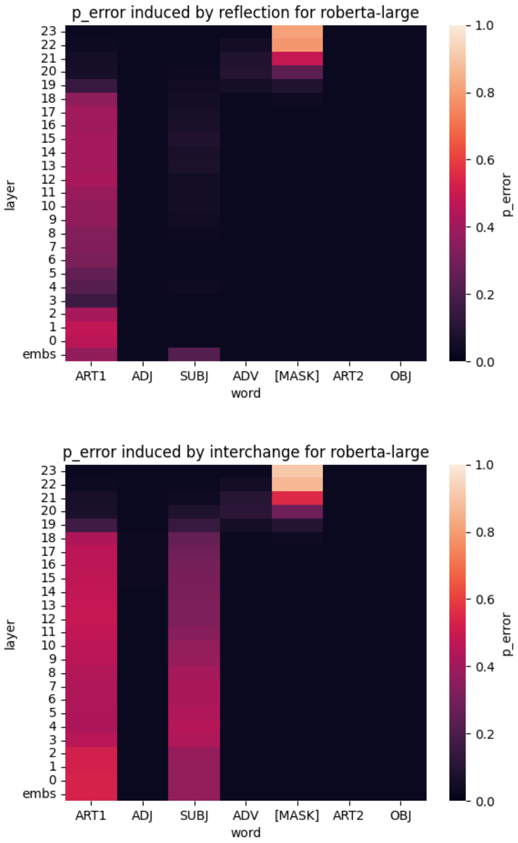


Figure 13: p_{error} induced by intervening on a representation, by word (x-axis), and layer (y-axis). Top: reflection intervention; bottom: interchange intervention

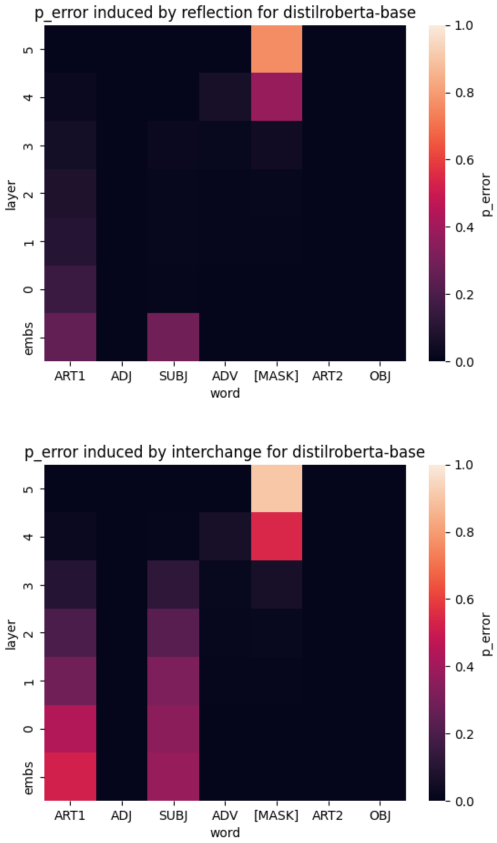


Figure 14: p_{error} induced by intervening on a representation, by word (x-axis), and layer (y-axis). Top: reflection intervention; bottom: interchange intervention

Do Pretrained Contextual Language Models Distinguish between Hebrew Homograph Analyses?

Avi Shmidman[1,2,†] **Cheyn Shmuel Shmidman**[2,‡]
Dan Bareket[1,‡] **Moshe Koppel**[1,2,‡] **Reut Tsarfaty**[1,†]

[1]Bar Ilan University [2]DICTA / Jerusalem, Israel
[†]`{avi.shmidman,reut.tsarfaty}@biu.ac.il`
[‡]`{cheynshmuel,dbareket,moishk}@gmail.com`

Abstract

Semitic morphologically-rich languages (MRLs) are characterized by extreme word ambiguity. Because most vowels are omitted in standard texts, many of the words are homographs with multiple possible analyses, each with a different pronunciation and different morphosyntactic properties. This ambiguity goes *beyond* word-sense disambiguation (WSD), and may include token segmentation into multiple word units. Previous research on MRLs claimed that standardly trained pre-trained language models (PLMs) based on word-pieces may not sufficiently capture the internal structure of such tokens in order to distinguish between these analyses. Taking Hebrew as a case study, we investigate the extent to which Hebrew homographs can be disambiguated and analyzed using PLMs. We evaluate all existing models for contextualized Hebrew embeddings on a novel Hebrew homograph challenge sets that we deliver. Our empirical results demonstrate that contemporary Hebrew contextualized embeddings outperform non-contextualized embeddings; and that they are most effective for disambiguating segmentation and morphosyntactic features, less so regarding pure word-sense disambiguation. We show that these embeddings are more effective when the number of word-piece splits is limited, and they are more effective for 2-way and 3-way ambiguities than for 4-way ambiguity. We show that the embeddings are equally effective for homographs of both balanced and skewed distributions, whether calculated as masked or unmasked tokens. Finally, we show that these embeddings are as effective for homograph disambiguation with extensive supervised training as with a few-shot setup.

1 Introduction

Semitic morphologically-rich languages (MRLs) such as Arabic, Hebrew, and Aramaic are characterized by extreme ambiguity at the word level (Wintner, 2014; Tsarfaty et al., 2020). In a standard text, many (and often most) of the words will be homographs with multiple possible analyses. The high ambiguity derives from several factors. First, prepositions, conjunctions, accusative pronouns, and possessive pronouns are often seamlessly affixed to words. Next, vowels are generally omitted in written texts. Finally, proper nouns are not differentiated from common nouns (no capital letters).

The task of distinguishing between Hebrew homograph analyses is related to the general task of Word Sense Disambiguation (WSD) (Agirre and Edmonds, 2006; Navigli, 2009), yet it is more challenging. In the standard case of WSD, a single orthographic form is associated with a single word that can be analyzed in terms of two or more senses; also, the analyses are generally pronounced identically, and often have the same morphosyntactic properties (e.g bank of a river vs. savings bank). In contrast, in Semitic languages, the need for disambiguation often goes beyond a determination of sense. Hebrew word ambiguities can be divided into three primary categories (Table 1): 1. Segmentation ambiguities, in which a given orthographic form may (or may not) be segmented into multiple word units each bearing its own role (POS tag) in the sentence. 2. Morphosyntactic ambiguities, in which the segmentation of the form is not ambiguous, but multiple analyses of the word reflect different morphosyntactic properties of each word unit(s). 3. Sense ambiguities (the aforementioned standard case of WSD), in which the analyses of the unit(s) do not differ in their morphosyntactic properties, but rather in their *sense*. One orthographic form may exhibit multiple types of ambiguity simultaneously.

Pretrained contextualized language models with standard word-piece tokenization mechanisms have been shown to excel at WSD in English and other Indo-European languages (Yaghoobzadeh et al., 2019). However, for Hebrew and other semitic lan-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 849–864
May 2-6, 2023 ©2023 Association for Computational Linguistics

Type	Form	Word (translation)	Morphology
Segmentation	הקפה	הַ+קָּפֶה (the+coffee)	DET + Noun [M,S,abs]
		הַקָּפָה (credit)	Noun [F,S,abs]
	שאף	שֶׁ+אַף (for+even)	Sconj + Cconj
		שָׁאַף (he aspired)	Verb [M,S,3,PAST]
Morph	אלימות	אַלִּימוֹת (violent)	Adj [F,P,abs]
		אַלִּימוּת (violence)	Noun [F,S,abs/cons]
	הרים	הֵרִים (he lifted)	Verb [M,S,3,PAST]
		הָרִים (mountains)	Noun [M,P,abs]
Semantic	הזמר	הַ+זֶּמֶר (the+song)	DET + Noun [M,S,abs]
		הַ+זַּמָּר (the+singer)	DET + Noun [M,S,abs]
	הסופר	הַ+סּוֹפֵר (the+author)	DET + Noun [M,S,abs]
		הַ+סּוּפֶּר (the+market)	DET + Noun [M,S,abs]

Table 1: Examples of Hebrew ambiguity types

guages it has been argued that such models would not sufficiently capture the structure of MRLs in order to distinguish between internally-complex homograph analyses (Klein and Tsarfaty, 2020; Tsarfaty et al., 2020). In this work, we take Modern Hebrew, a Semitic language with rich and highly ambiguous morphology, as a case study, and we investigate the extent to which homographs can be disambiguated by contextualized embeddings, regarding all three levels of ambiguity. Regarding Arabic — a sister language to Hebrew — a wide survey of WSD methods is presented by Abderrahim and Abderrahim (2022). They raise the possibility of utilizing pretrained contextualized embeddings, yet leave its evaluation to future work.[1]

Hebrew is a particularly challenging language on which to perform a homograph disambiguation due to the limited available corpora. First of all, currently existing Hebrew treebanks are severely limited in size, such that most of the words in the language are not amply represented. Furthermore, even regarding common Hebrew words, this corpus is problematic, because the nature of language is such that many homographs are skewed in their distribution; thus, even if the primary analysis is sufficiently represented within a tagged corpus, the secondary analysis will often be hopelessly underrepresented. For instance, one common Hebrew homograph is מהם (*mhm*), which can be analyzed as a preposition with pronominal suffix, or as an interrogative. The ratio of these two analyses in naturally-occurring Hebrew text is over 50:1; thus, occurrences of the secondary analysis within existing tagged corpora are insufficient.

In theory, these homograph ambiguities could be addressed using POS tagging systems. For instance, Habash and Rambow (2005) consider the use of a morphological tagging system to solve WSD in Arabic. A number of Hebrew POS tagging systems have been published as well (Yona and Wintner, 2005; Adler and Elhadad, 2006; Shacham and Wintner, 2007). The current SOTA for Hebrew POS tagging is the YAP morpho-syntactic parser (Tsarfaty et al., 2019). However, as we have shown in a previous study (Shmidman et al., 2020, p. 3318, table 2), although YAP produces high accuracy overall on normal Hebrew text, its scores drop drastically regarding homographs of skewed distribution.

For analogous cases of skewed distribution in other languages, researchers have proposed the creation of dedicated challenge sets, containing hard-to-classify sentences not easily found in naturally-occurring text (Gardner et al., 2020; Elkahky et al., 2018). In the aforementioned previous study, we produced 22 such challenge sets for Hebrew homographs, and demonstrated that a Bi-LSTM of non-contextualized embeddings can obtain high accuracy on this task, establishing the current SOTA for Hebrew homograph disambiguation (Shmidman et al., 2020). In this paper, we extend the investigation by considering whether contextualized embeddings from pretrained language models (PLMs) can provide a more optimal solution. We consider all existing contextualized Hebrew PLMs: multilingual BERT ("mBERT") (Devlin et al., 2019); HeBERT (Chriqui and Yahav, 2021); and AlephBERT (Seker et al., 2021) (Table 2). Moreover, we evaluate and verify these on a new dataset, substantially larger than all previous datasets for Hebrew homograph disambiguation.

Our experiments demonstrate that contextualized PLMs pre-trained on sufficiently large unlabeled data and vocabulary size are excellent at disambiguating the word-internal structures of homographs, yet face some challenge with pure sense disambiguation. We show the efficacy of these models in cases of homographs with skewed distribution, and in a few-shot setup. Finally, we establish new state-of-the-art results on the challenging task of homograph disambiguation for a morphologically-rich language printed without vowels, along with a novel benchmark for assessing the morphological reach of future PLMs in Hebrew.

2 The Data

The challenge sets for Hebrew homograph disambiguation from our previous study were limited in

---

[1] Additional studies in Arabic WSD include Merhben et al. (2010), Merhbene et al. (2013), and Shah et al. (2010).

Model	Vocab (Heb. tokens)	Corpus Size (Heb. sentences)
mBERT	2.5K	6.3M
HeBERT	30K	27.2M
AlephBERT	52K	98.7M

Table 2: Comparison of available Hebrew BERT models

number (only 22 sets) and insufficiently representative regarding types of ambiguities; only one of the sets was a prefix-segmentation ambiguity. Further, they were limited to binary cases, where only two analyses exist.

In contrast, for this study we employed field experts to choose the most critical homographs in the language. The experts chose 75 homographs from a list of the 3600 most frequent words in the language, balancing frequency of word occurrence with practical need for its disambiguation. All of the homographs occur with a minimum frequency of 27 words per million in naturally occurring Hebrew text. Our challenge sets include homographs with 2-5 possible analyses. Our sets contain a wide representation of segmentation ambiguities (15 in number), as well as 5 cases of purely semantic ambiguities. For each of the 75 homographs, we collect hundreds of naturally-occurring sentences attesting to each analysis. In almost all cases, we succeed in collecting 1000 sentences for the primary analysis, at least 500 sentences for the secondary analysis, and at least 250 for each additional analysis. The sentences were culled from newspapers, Wikipedia, literature, and social media. We employed a team of annotators who chose the relevant homograph analysis for each case.[2] All in all, our 75 challenge sets contain 150K tagged sentences. The full list of homographs and analyses is provided in Appendix A.[3]

3 Experimental Setup

To evaluate the ability of pre-trained language models (PLMs) to disambiguate the in-context analyses of morphologically rich and highly ambiguous homographs in Hebrew, we adopt a "word expert" approach, producing dedicated classifiers for each individual homograph (Zhao et al., 2020).

We use two types of PLMs, contextualized and non-contextualized. For the non-contextualized case, we replicate our previous method detailed in Shmidman et al. (2020). For each training example, we use a BiLSTM on top of the word2vec embeddings of all of the words in the sentence (other than the homograph itself) to produce an encoding for disambiguation.[4] An MLP is trained to predict the correct homograph analysis based on this encoding.[5] For the contextualized case, we run the sentence through a pretrained contextualized language model and retrieve the 768-dimension embedding representing the homograph in question. An MLP is trained to predict the correct analysis based on the homographs embeddings alone. In the standard "unmasked" scenario, the sentence is fed into the model as is, including the homograph in question. In the "masked" scenario, the homograph is replaced with a [MASK] token.

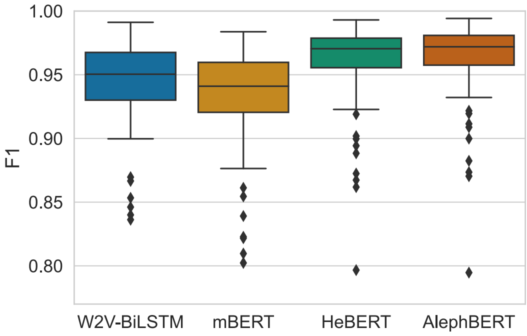


Figure 1: Comparison of previous SOTA (w2v-based Bi-LSTM method) versus BERT-based approaches.

We evaluate the performance of each given method on each given challenge set using 10-fold cross-validation. We calculate an F1 score for each homograph analysis, based upon the precision and recall scores micro-averaged across all folds. We then calculate the macro-average of the F1 scores for all possible analyses for a given homograph, and this is the score reported in the charts herein.

4 Results and Analysis

Standard (Unmasked) Scenario Figure 1 presents the cumulative F1 score obtained by the models for all challenge sets. Our results show that HeBERT and AlephBERT far outperform mBERT, with AlephBERT achieving the higher score. The poor performance of mBERT is likely due to its smaller pre-training data size and exceedingly lean Hebrew vocabulary (cf. Table 2). Furthermore, the

---

[2] The annotation process is detailed in Appendix B.

[3] The dataset is downloadable at: `https://github.com/Dicta-Israel-Center-for-Text-Analysis/EACL_2023`

[4] We also tested fastText, but results were inferior.

[5] For implementation details, see Appendix D.

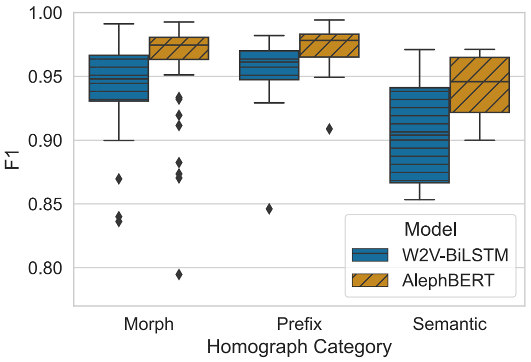


Figure 2: Disambiguation accuracy across various categories of homograph ambiguity.

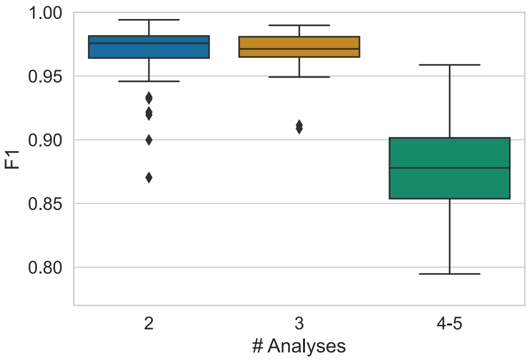


Figure 3: AlephBERT's disambiguation accuracy across homographs with differing counts of possible analyses.

HeBERT and AlephBERT models both substantially outperform the previous word2vec-based SOTA. It is thus apparent that *contextualized* language models do effectively capture Hebrew homograph distinctions, even those based on word-pieces, even for an MRL, and they do so more effectively than non-contextualized models.

Figure 2 demonstrates AlephBERT's performance on different ambiguity types. AlephBERT performs equally well on cases of segmentation ambiguity and morphosyntactic ambiguity. In contrast, when it comes to ambiguities that are purely semantic, the scores are noticeably lower. This is in line with the findings of Ettinger (2020), who shows that BERT is stronger with syntax than semantics; Goldberg (2019) also notes BERT's strong syntactic abilities. Interestingly, the same gap exists with the W2V-based method. Thus, both contextualized and non-contextualized embeddings struggle to differentiate between senses which are morphologically equivalent. Although such cases are only of minimal import when it comes to sentence parsing, they are critical for downstream tasks such as coreference resolution and relation extraction. It thus remains a desideratum to improve disambiguation of purely semantic Hebrew homographs.

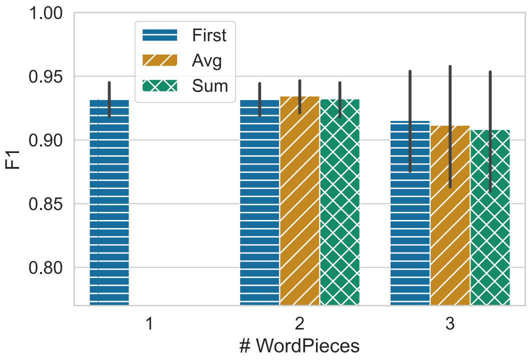


Figure 4: Disambiguation accuracy across varying degrees of word piece splits within the target homograph, using mBERT.

The results in Figure 3 demonstrate that AlephBERT performs equally well on cases of binary homographs as on cases of three-way homograph classification. However, when faced with cases of 4-way or 5-way classification, accuracy declines.

The Effect of Word-Pieces Previous studies have hypothesized that word-pieces are not adequate for capturing complex morphosyntactic structures due to arbitrary (non-linguistic) word-splits. To probe into this we investigate the question, do such splits affect performance. Our 75 homographs are all treated as single tokens in HeBERT and AlephBERT. However, many of the homographs are broken up into word pieces in mBERT, due to its meager Hebrew vocabulary. We thus compare mBERT's results on words treated as single tokens versus those that are broken up into two or three pieces, which are aggregated using first, sum, or average of the vectors. With regard to cases of split words, we train models using three separate methods: providing the MLP with only the embedding of the first word piece; with an average of the word piece embeddings; or with the sum of the embeddings. As shown in Figure 4, the splitting of a homograph into three word-pieces appears to have a negative impact on the ability of the resulting embedding to differentiate between homograph analyses, for all aggregation methods.

Masked Scenario We consider whether AlephBERT embeddings are more effective if we replace the homograph word with [MASK] when running

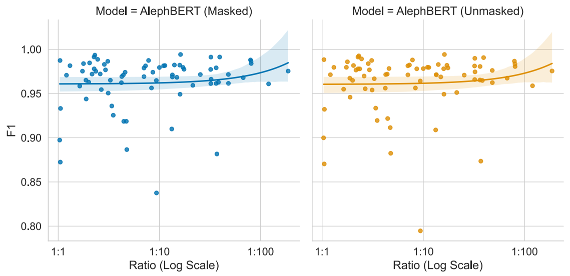


Figure 5: Comparison of unmasked vs. masked scenarios, across varying degrees of skewed homographs. The ratio axis indicates the relative distribution between the least and most frequent analyses of each plotted homograph within naturally-occurring Hebrew text.

the challenge set sentences through AlephBERT. The motivation behind this experiment is that, as explained above, many of the homographs are skewed in their natural proportion. In such cases, we worry whether AlephBERT might be disproportionately influenced by the skewed distribution; replacing the word with [MASK] would prevent the model from being influenced as such. As shown in figure 5, AlephBERT achieves high scores both with balanced homographs as well as with homographs of highly skewed distribution. Using a [MASK] token instead of the actual word does not generally improve performance, whether or not the homographs are of skewed proportion.

Few-Shot Scenarios In our experiments thus far, the 10-fold cross-validation allows the MLP to leverage 90% of the data in each fold (hundreds of sentences for each analysis) in order to learn the difference between the analyses. We now consider whether the AlephBERT embeddings can suffice on a few-shot basis, where the training stage has access to only 100, 50, 25, 10 or even 5 examples of each analysis. In these cases, we train an MLP based only on these few samples, and we use the rest of the sentences for evaluation. Astoundingly, as demonstrated in Figure 6, the AlephBERT embeddings provide a highly accurate solution even on this few-shot basis. Even when training with only 5 examples of each homograph analysis, AlephBERT reaches an accuracy that is not far below the accuracy achieved when performing full 10-fold CV across hundreds of sentences of each analysis.

Probing Scenarios Finally, we probe the pre-trained AlephBERT embeddings (Yaghoobzadeh et al., 2019; Tenney et al., 2019; Klafka and Ettinger, 2020; Belinkov, 2021) to see whether in and of themselves they reflect clusters which correspond to different homograph analyses. We skip the MLP, and instead use the raw embeddings directly, classifying sentences based on their proximity to the centroid of the training samples for each homograph analysis. As shown in the orange bars in Figure 6, this method generally does not perform as well as the MLP-based method; however, the degradation is limited to only a few percentage points, indicating that the raw embeddings are generally clustered in groups which indeed reflect the distinctions between the analyses.

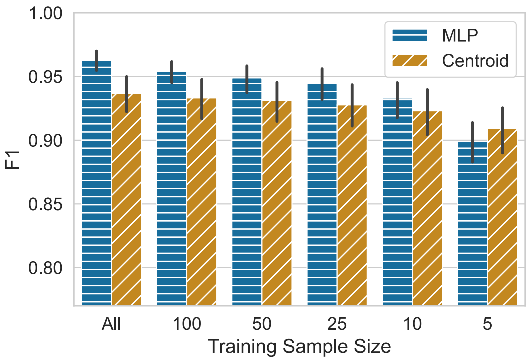


Figure 6: Use of AlephBERT embeddings to differentiate between homographs on a few-shot basis, contrasted with scores from the full 10-fold CV ("All").

5 Conclusion

In this study we have utilized a wide-ranging collection of Hebrew homograph challenge sets in order to evaluate the extent to which raw contextualized embeddings can be leveraged to disambiguate Hebrew homographs. We found that contextualized embeddings can effectively disambiguate analyses of homographs, much more so than non-contextualized ones, regarding multiple types of ambiguity: segmentation, morphosyntactic and sense. Yet, efficacy on pure sense ambiguity is lower than on the other two types. Additionally, an increasing number of splits, or an increasing number of different possible analyses of a token, each lower efficacy. Finally, we found that contextualized embeddings can function effectively for this purpose on a few-shot basis, with as little as 5 examples of each analysis. This indicates that with relatively modest effort, highly ambiguous homographs may be effectively treated.

Limitations

One of the major strengths of this paper is its new and comprehensive dataset for the training and benchmarking Hebrew homograph disambiguation. The dataset is uncomparable in size, quality and balance to all previous Hebrew homograph datasets; we have made every effort to be as inclusive as possible in the creation of the dataset, making sure to include data from a widely diverse set of genres. Nevertheless, a perennial challenge in corpus-based studies is that the lion's share of the available data tends to be authored by male writers. In order to offset this bias, we bolstered our corpus with a large corpus of texts specifically taken from blog sites devoted entirely to female bloggers. Even so, we cannot escape the fact that female writing and feminine conjugations are underrepresented in our dataset.

A further limitation derives from our filter for sentences with offensive language. We perceived early on that our human annotators were not comfortable tagging sentences with offensive language, and we therefore took steps to remove such sentences from our corpus. Nevertheless, this means that our resulting dataset is limited in that it does not properly reflect the use of offensive language in naturally-occurring Hebrew sentences. Furthermore, our resulting tests and reported scores may not accurately reflect the performance of our models when applied to sentences with offensive language.

Ethics Statement

Creation of the Dataset As noted, our dataset contains over 150K sentences in all. Every sentence was reviewed and tagged by our team of human annotators, who chose the relevant homograph analysis for each instance of each of our 75 homographs. Our annotator team included members of diverse genders and sexual orientations. They were paid hourly wages with legal pay stubs. Their hourly wage was well above the minimum wage required by law. The entirety of the dataset will be made available for research purposes with the acceptance of this article, together with the tagging information.

Risks of the Research Ultimately, this data will enable end-users to automatically diacritize and parse large corpora of Hebrew text. For the most part, this will provide a beneficial contribution to the world: for the visually impaired, this technology will enable the development of more precise text-to-speech products; teachers will be able to provide children and second-language learners with accessible diacritized texts; and humanities and linguistics researchers can bolster their research with big-data analysis. However, there also is a risk of nefarious use, for instance, if an end user were to leverage these capabilities in order to produce anonymous texts or recordings containing threats to life, liberty, or happiness.

Acknowledgments

The work of the third and last authors has been funded by a grant from the Israeli Ministry of Science and Technology (MOST) grant no. 3-17992.

References

Mohammed Alaeddine Abderrahim and Mohammed El-Amine Abderrahim. 2022. Arabic word sense disambiguation for information retrieval. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 21(4).

Meni Adler and Michael Elhadad. 2006. An unsupervised morpheme-based hmm for hebrew morphological disambiguation. In *COLING/ACL 2006 - 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, COLING/ACL 2006 - 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, pages 665–672, United States. Association for Computational Linguistics (ACL). 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, COLING/ACL 2006 ; Conference date: 17-07-2006 Through 21-07-2006.

Eneko Agirre and Philip Edmonds, editors. 2006. *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*. Springer.

Yonatan Belinkov. 2021. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, pages 1–13.

Avihay Chriqui and Inbal Yahav. 2021. Hebert hebemo: a hebrew bert model and a tool for polarity analysis and emotion recognition.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ali Elkahky, Kellie Webster, Daniel Andor, and Emily Pitler. 2018. A challenge set and methods for noun-verb ambiguity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2562–2572, Brussels, Belgium. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hanna Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *CoRR*, abs/1901.05287.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan. Association for Computational Linguistics.

Josef Klafka and Allyson Ettinger. 2020. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4801–4811, Online. Association for Computational Linguistics.

Stav Klein and Reut Tsarfaty. 2020. Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 204–209, Online. Association for Computational Linguistics.

Laroussi Merhben, Zouaghi Anis, and Mounir Zrigui. 2010. Ambiguous arabic words disambiguation. pages 157–164.

Laroussi Merhbene, Anis Zouaghi, and Mounir Zrigui. 2013. A semi-supervised method for Arabic word sense disambiguation using a weighted directed graph. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1027–1031, Nagoya, Japan. Asian Federation of Natural Language Processing.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41.

Amit Seker, Elron Bandel, Dan Bareket, Idan Brusilovsky, Refael Shaked Greenfeld, and Reut Tsarfaty. 2021. Alephbert:a hebrew large pre-trained language model to start-off your hebrew nlp application with.

Danny Shacham and Shuly Wintner. 2007. Morphological disambiguation of Hebrew: A case study in classifier combination. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 439–447, Prague, Czech Republic. Association for Computational Linguistics.

Rushin Shah, Paramveer S. Dhillon, Mark Liberman, Dean Foster, Mohamed Maamouri, and Lyle Ungar. 2010. A new approach to lexical disambiguation of Arabic text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 725–735, Cambridge, MA. Association for Computational Linguistics.

Avi Shmidman, Joshua Guedalia, Shaltiel Shmidman, Moshe Koppel, and Reut Tsarfaty. 2020. A novel challenge set for Hebrew morphological disambiguation and diacritics restoration. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3316–3326, Online. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations.

Reut Tsarfaty, Dan Bareket, Stav Klein, and Amit Seker. 2020. From SPMRL to NMRL: What did we learn (and unlearn) in a decade of parsing morphologically-rich languages (MRLs)? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7396–7408, Online. Association for Computational Linguistics.

Reut Tsarfaty, Amit Seker, Shoval Sadde, and Stav Klein. 2019. What's wrong with hebrew nlp? and how to make it right.

Shuly Wintner. 2014. Morphological processing of semitic languages. In *NLP of Semitic Languages*.

Yadollah Yaghoobzadeh, Katharina Kann, T. J. Hazen, Eneko Agirre, and Hinrich Schütze. 2019. Probing for semantic classes: Diagnosing the meaning content of word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5740–5753, Florence, Italy. Association for Computational Linguistics.

Shlomo Yona and Shuly Wintner. 2005. A finite-state morphological grammar of Hebrew. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 9–16, Ann Arbor, Michigan. Association for Computational Linguistics.

Mengjie Zhao, Philipp Dufter, Yadollah Yaghoobzadeh, and Hinrich Schütze. 2020. Quantifying the contextualization of word representations with semantic class probing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1219–1234, Online. Association for Computational Linguistics.

A Appendix A: Table of Homographs

We present three tables of homographs, corresponding to the three categories of homographs discussed within the paper (segmentation ambiguity, morphosyntactic ambiguity, and semantic ambiguity).[6] In each table, the first column ("form") indicates the homograph as it is found in naturally-occurring non-diacritized Hebrew text.[7] The second column ("word") indicates the possible diacritizations of each form. In cases where the diacritized word includes an attached prefix, a plus sign indicates the segmentation point between the prefix letters and the primary word. Regarding all homographs considered in this paper, different segmentation options are diacritized differently. Thus, for each sentence in the dataset, our human annotators were asked simply to choose the correct diacritization for the target homograph (that is, they were asked to choose among the options listed in the "word" column). There was no need for the annotators to separately tag the segmentation, because in all cases the choice of diacritization itself indicates the segmentation.[8] The third column indicates the morphology of each of the possible diacritizations.[9] The fourth column lists the translation.[10] Within each table the homographs are listed alphabetically.[11]

A.1 Homographs with Segmentation Ambiguity

Form	Word	Morphology[12]	Translation	Sentences
האם	הַאִם	Interrogative	does	1,000
	הָ+אֵם	Det + Noun [F,S,abs]	the + mother	1,000
המראה	הַ+מַּרְאֶה	SConj + Ptcp [M,S] / Det + Noun [M,S,abs]	that + indicates / the + sight	1,000
	הַ+מַּרְאָה	SConj + Ptcp [F,S] / Det + Noun [F,S,abs]	that + indicates / the + mirror	882
	הַמְּרָאָה	Noun [F,S,abs]	takeoff	735

---

[6] It should be noted that the higher level of ambiguity are supersets of the lower levels: segmentation ambiguities generally entail differences on the morphosyntactic and semantic levels as well, and morphosyntactic ambiguities generally also entail semantic ambiguities. Furthermore, because many of the homographs admit to more than two analyses, it is often the case that a subset of the analyses may form a lower level of ambiguity (e.g., just a semantic ambiguity), while other analyses form a higher level of ambiguity (e.g. segmentation ambiguity). For the purposes of this paper, we categorize each homograph according to the highest level of ambiguity involved: First, if a segmentation ambiguity is indicated anywhere across the possible analyses, then we include the homograph in the "segmentation ambiguity" category. Next, if there is no segmentation ambiguity, but a morphosyntactic ambiguity is indicated anywhere across the possible analyses, then we include the homograph in the "morphosyntactic ambiguity" category. Finally, if the analyses all differ only on the semantic level, then we include the homograph in the "semantic ambiguity" category.

[7] The ranking of analyses is based on a frequency analysis of our in-house annotated corpus. It is worth emphasizing that the paper as a whole relates to each of the 75 homographs specifically as they are spelled in this list, and does not relate to cases where further prefixes are attached to the homographs. As a result, the frequency analysis may sometimes seem counterintuitive. For instance, regarding the form ראשי, a native Hebrew speaker might intuit that the adjectival form is primary (רָאשִׁי). However, in practice, that sense is common only when prefixed with a definitive marker (הראשי). In contrast, the homograph considered here involves the form ראשי as is, without any prefixes; in this case, the other analyses are far more common.

[8] It should be noted that in certain sentences, an exceedingly rare diacritization was warranted, which was not among the options listed in the "word" column. The annotators were instructed to tag such cases as "none of the above", and all such sentences were removed from the corpus. Similarly, some sentences do not provide enough context to determine the correct diacritization; the annotators were asked to tag such sentences as "unclear", and these sentences too were removed from the corpus.

[9] In most cases, a diacritized form has one specific morphological analysis. However, in other cases, the diacritized form can admit to multiple morphologies. In such cases, we list all of the practically relevant morphological analyses in the third column, separated by a slash (thus for instance in the case of זר). Rare analyses which hardly ever occur in practice are not listed.

[10] Naturally, a given Hebrew term often captures a substantial range of potential English translations, and it would not be practical to list them all in this column; therefore, we generally present only a single representative translation.

[11] Ideally, we might have grouped the homographs based on part-of-speech instead. However, as can be seen from the following tables, the 75 homographs vary so widely in terms of the parts of speech that they can represent, such that alphabetical listing was deemed most useful.

[12] Key to morphology abbreviations: F=Feminine; M=Masculine; S=Singular; P=Plural; abs=absolute; cons=construct; 1,2,3=First Person, Second Person, Third Person; suf=suffix; Det = determiner; SConj=Subordinating Conjunction; Adj=Adjective; Conj=Conjunction; Prep=Preposition; Propn=Proper Noun; Adv=Adverb; Ptcp=Participle; ACC=Accusative marker.

הקפה	הַ+קָפֶה	Det + Noun [M,S,abs]	the + coffee	1,000
	הַקָפָה	Noun [F,S,abs]	credit	500
הקשר	הַ+קֶשֶׁר	Det + Noun [M,S,abs]	the + connection	1,000
	הַ+קַשָׁר	Det + Noun [M,S,abs]	the + signaler	1,000
	הֶקְשֵׁר	Noun [M,S,abs/cons]	context	588
השלמה	הַשְׁלָמָה	Noun [F,S,abs]	completion	1,000
	הַ+שְׁלֵמָה	Det + Adj [F,S]	the + complete	906
ועד	וְ+עַד	Conj + Prep	and + until	1,000
	וַעַד	Noun [M,S,abs/cons]	committee	519
לשם	לְשֵׁם	Prep	for the purpose of	1,000
	לְ+שָׁם	Prep + Adverb	to + there	1,000
מבחינה	מִ+בְּחִינָה	Prep + Noun [F,S,abs]	from + point of view	1,000
	מַבְחִינָה	Participle [F,S,abs]	she notices	607
מסיבות	מְסִיבּוֹת	Noun [F,P,abs]	parties	1,000
	מִ+סִיבּוֹת	Prep + Noun [F,P,abs/cons]	due to + reasons	1,000
מפתח	מַפְתֵּחַ	Noun [M,S,abs]	key	1,000
	מְפַתֵּחַ	Participle [M,S,abs]	develops / developer	329
	מִ+פֶּתַח	Prep + Noun [M,S,abs/cons]	from + opening	206
שאלה	שְׁאֵלָה	Noun [F,S,abs]	question	1,000
	שָׁאֲלָה	Verb [F,S,3,Past]	she asked	1,000
	שֶׁ+אֵלֶּה	SConj + Pronoun [MF,P,3]	that + these	228
שאף	שֶׁ+אַף	SConj + CConj	for + even	1,000
	שָׁאַף	Verb [M,S,3,Past]	he aspired	665
שבה	שֶׁ+בָּהּ	Sconj + Prep [suf=F,S,3]	that + in it	1,000
	שָׁבָה	Verb [F,S,Present/Past]	she returns / she returned	1,000
שמן	שֶׁמֶן	Noun [M,S,abs/cons]	oil	1,000
	שָׁמֵן	Adj [M,S,abs]	wide	335
	שֶׁ+מִּן	SConj + Prep	that + from	207
	שְׁמָן	Noun [M,S,abs,suf=F,P,3]	their name	149
שמר	שָׁמַר	Verb [M,S,3,Past]	he guarded	1,000
	שֶׁמֶר	Propn	Shemer	872
	שֶׁ+מַּר	SConj + Titular [M,S]	that + Mr.	224

A.2 Homographs with Morphosyntactic Ambiguity

Form	Word	Morphology	Translation	Sentences
אהבה	אַהֲבָה	Noun [F,S,abs]	love	1,000
	אָהֲבָה	Verb [F,S,3,Past]	she loved	1,000
אוכל	אוֹכֶל	Noun [M,S,abs/cons]	food	748
	אוּכַל	Modal [MF,S,1,Future]	I can	729
	אוֹכֵל	Participle [M,S,abs/cons]	eats	640
אחדות	אֲחָדוֹת	Det [F,P,abs]	several	1,000
	אַחְדוּת	Noun [F,S,abs]	unity	1,000

אחיו	אֶחָיו	Noun [MF,P,abs,suf=M,S,3]	his brothers	1,000
	אָחִיו	Noun [MF,S,abs,suf=M,S,3]	his brother	774
אלימות	אַלִּימוֹת	Adj [F,P]	violent	1,000
	אַלִּימוּת	Noun [F,S,abs/cons]	violence	1,000
אם	אִם	Conj	if	1,000
	אֵם	Noun [F,S,abs/cons]	mother	1,000
אמצעי	אֶמְצָעֵי	Noun [M,P,cons]	centers of / methods of	1,000
	אֶמְצָעִי	Noun [M,S,abs] / Adj [M,S]	method / central	969
אמרה	אָמְרָה	Verb [F,S,3,Past]	she said	1,000
	אִמְרָה	Noun [F,S,abs]	a saying	343
אפשר	אֶפְשָׁר	Modal / Adv	possible	1,000
	אִפְשֵׁר	Verb [M,S,3,Past]	he allowed	511
את	אֶת	ACC	accusative	1,000
	אַתְּ	Pronoun [F,S,2]	you	1,000
בהמשך	בְּ+הֶמְשֵׁךְ	Prep + Noun [M,S,cons]	in + continuation of	1,000
	בַּ+הֶמְשֵׁךְ	Prep [with Det] + Noun [M,S,abs]	in the + future	930
בחיי	בְּ+חַיֵּי	Prep + Noun [M,P,cons]	in + lives of	1,000
	בְּ+חַיַּי	Prep + Noun [M,P,abs,suf=MF,S,1]	in + my life	1,000
בעולם	בָּ+עוֹלָם	Prep [with Det] + Noun [M,S,abs]	in the + world	1,000
	בְּ+עוֹלַם	Prep + Noun [M,S,cons]	in + a world of	913
	בְּ+עוֹלָם	Prep + Noun [M,S,abs]	in + a world	485
בקרב	בְּ+קֶרֶב	Prep + Noun [M,S,cons]	in + midst of	1,000
	בַּ+קְרָב	Prep [with Det] + Noun [M,S,abs]	in the + battle	1,000
	בִּ+קְרַב	Prep + Verb [Bare Infinitive]	in + approaching of	734
	בִּ+קְרָב	Prep + Noun [M,S,abs]	in + a battle	256
גילו	גִּילּוּ	Verb [MF,P,3,Past]	they discovered	1,000
	גִּילוֹ	Noun [M,S,abs,suf=M,S,3]	his age	859
די	דַּי	Adverb	sufficiently	1,000
	דֵּי	Det [cons]	enough of	828
	דִּי	Prefix	di-	808
הזקן	הַ+זָּקָן	Det + Noun [M,S,abs]	the + beard	739
	הַ+זָּקֵן	Det + Adj [M,S] / Det + Noun [M,S,abs]	the + old / the + elderly man	533
החל	הֵחֵל	Verb [M,S,3,Past]	he began	1,000
	הָחֵל	Verb [Bare Infinitive]	starting (from)	1,000
המשנה	הַ+מִּשְׁנֶה	Det + Noun [M,S,abs]	the + deputy	1,000
	הַ+מִּשְׁנָה	Det + Propn [F,S,abs]	the + Mishna	1,000
הנחה	הַנָּחָה	Noun [F,S,abs]	placing	1,000
	הִנְחָה	Verb [M,S,3,Past]	he directed	731
	הֲנָחָה	Noun [F,S,abs]	discount	517
הרים	הֵרִים	Verb [M,S,3,Past]	he lifted	1,000
	הָרִים	Noun [M,P,abs]	mountains	1,000
ואת	וְ+אֶת	Conj + ACC	and + accusative	1,000
	וְ+אַתְּ	Conj + Pronoun [F,S,2]	and + you	1,000
זר	זֵר	Noun [M,S,abs/cons]	bouquet	1,000
	זָר	Adj [M,S] / Noun [M,S,abs]	foreign / stranger	1,000

חברות	חֲבָרוֹת	Noun [F,P,abs]	companies	1,000
	חֶבְרוֹת	Noun [F,P,cons]	companies of	1,000
	חֲבֵרוֹת	Noun [F,P,abs/cons]	friends	501
	חֲבֵרוּת	Noun [F,S,abs/cons]	friendship	398
	חַבְרוֹת	Noun [F,P,cons]	friends of	261
חדר	חֲדַר	Noun [M,S,cons]	room of	1,000
	חֶדֶר	Noun [M,S,abs]	room	1,000
	חָדַר	Verb [M,S,3,Past]	penetrated	783
טוב	טוֹב	Adj [M,S]	good	1,000
	טוּב	Noun [M,S,abs/cons]	goodness	357
יהודי	יְהוּדִי	Noun [M,S,abs] / Adj [M,S]	a Jew / Jewish	1,000
	יְהוּדֵי	Noun [M,P,cons]	Jews	1,000
כיוון	כֵּיוָן	Conj	because	487
	כִּיוּוּן	Noun [M,S,abs] / Noun [M,S,cons]	direction	468
	כִּיוֵּן	Verb [M,S,3,Past]	directed	455
לו	לוֹ	Prep [suf=M,S,3]	to him	1,000
	לוּ	Conj	if only	1,000
לחם	לֶחֶם	Noun [M,S,abs]	bread	1,000
	לָחַם	Verb [M,S,3,Past]	he fought	1,000
לפנות	לִפְנוֹת	Prep / Verb [Infinitive]	facing / to turn	1,000
	לְפַנּוֹת	Verb [Infinitive]	to clear out	570
מדי	מִדֵּי	Det [cons]	every	1,000
	מִדַּי	Adv	too much	802
	מַדֵּי	Noun [M,P,cons]	uniforms of	541
מהם	מֵהֶם	Preposition [suf=M,P,3]	from them	1,000
	מָהֵם	Interrogative	what are	587
מי	מִי	Interrogative / Pronoun [S,3]	who	1,000
	מֵי	Noun [M,P,cons]	waters of	1,000
מלך	מֶלֶךְ	Noun [M,S,abs/cons]	king	1,000
	מָלַךְ	Verb [M,S,3,Past]	he ruled	522
מעבר	מֵעֵבֶר	Prep	beyond	1,000
	מַעֲבָר	Noun [M,S,abs]	passage	1,000
	מַעֲבַר	Noun [M,S,cons]	passage of	883
מראה	מַרְאֶה	Participle [M,S]	he shows	1,000
	מַרְאָה	Participle [F,S]	she shows	1,000
מרכז	מֶרְכַּז	Noun [M,S,cons]	center of	1,000
	מֶרְכָּז	Noun [M,S,abs]	center	1,000
	מְרַכֵּז	Participle [M,S,abs/cons]	organizes / organizer	393
משחק	מִשְׂחָק	Noun [M,S,abs]	game	1,000
	מְשַׂחֵק	Participle [M,S,abs]	plays / player	479
נעשה	נַעֲשֶׂה	Verb [MF,P,1,Future]	we will do	1,000
	נַעֲשָׂה	Verb [M,S,3,Past]	was done	1,000
נשים	נָשִׁים	Noun [F,P,abs]	women	1,000
	נָשִׂים	Verb [MF,P,1,Future]	we will put	613
נתן	נָתַן	Verb [M,S,3,Past]	gave	1,000
	נָתָן	Propn	Nathan	681

עבר	עָבַר	Verb [M,S,3,Past]	he passed	1,000
	עָבָר	Noun [M,S,abs]	past	1,000
	עֵבֶר	Noun [M,S,abs/cons]	side	456
עד	עַד	Prep	until	1,000
	עֵד	Noun [M,S,abs/cons]	witness	1,000
עובדות	עֻבְדוֹת	Noun [F,P,abs/cons]	facts	1,000
	עוֹבְדוֹת	Participle [F,P]	they work / workers	1,000
עם	עִם	Prep	with	1,000
	עַם	Noun [M,S,abs/cons]	nation	1,000
פני	פְּנֵי	Noun [M,P,cons]	faces of	1,000
	פָּנַי	Noun [MF,P,abs,suf=MF,S,1]	my face	338
פרס	פְּרָס	Noun [M,S,abs]	award	1,000
	פֶּרֶס	Propn	Peres	956
	פָּרַס	Verb [M,S,3,Past]	he spread	630
	פְּרַס	Noun [M,S,cons]	award of	290
ציון	צִיּוֹן	Propn	Zion	1,000
	צִיּוּן	Noun [M,S,abs/cons]	mark	1,000
קודם	קוֹדֶם	Adv	before	1,000
	קוֹדֵם	Adj [M,S]	previous	1,000
	קֻודַּם	Verb [M,S,3,Past]	was promoted	284
ראשי	רָאשֵׁי	Noun [M,P,cons]	heads	1,000
	ראֹשִׁי	Noun [M,S,abs,suf=MF,S,1]	my head	881
	רָאשִׁי	Adj [M,S,abs]	head	399
שירת	שֵׁירֵת	Verb [M,S,3,Past]	he served	1,000
	שִׁירַת	Noun [F,S,cons]	poetry of	896
שכר	שָׂכָר	Noun [M,S,abs]	salary	1,000
	שְׂכַר	Noun [M,S,cons]	salary of	751
	שָׂכַר	Verb [M,S,3,Past]	rented	681
שם	שֵׁם	Noun [M,S,abs/cons]	name	1,000
	שָׁם	Adv	there	1,000
	שָׁם	Verb [M,S,Present/Past]	he placed	1,000
תנאי	תְּנָאֵי	Noun [M,P,cons]	conditions of	1,000
	תְּנַאי	Noun [M,S,abs/cons]	condition	834

A.3 Homographs with Semantic Ambiguity

Form	Word	Morphology	Translation	Sentences
הזמר	הַ+זֶּמֶר	Det + Noun [M,S,abs]	the + music	1,000
	הַ+זַּמָּר	Det + Noun [M,S,abs]	the + musician	1,000

הסופר	**הַ+סוּפֶּר**	Det + Noun [M,S,abs]	the + market	763
	הַ+סוֹפֵר	Det + Noun [M,S,abs]	the + author	570
זמר	**זַמָּר**	Noun [M,S,abs]	musician	1,000
	זֶמֶר	Noun [M,S,abs]	song	602
חברה	**חֲבֵרָה**	Noun [F,S,abs]	friend	1,000
	חֶבְרָה	Noun [F,S,abs]	company	1,000
רשות	**רְשׁוּת**	Noun [F,S,abs/cons]	permission	1,000
	רָשׁוּת	Noun [F,S,abs/cons]	authority	1,000

B Appendix B: Creation of our Annotated Dataset

We present here further details regarding the creation of our annotated homograph dataset. As noted above, our dataset contains challenge sets for 75 Hebrew homographs. In almost all cases, we collect 1000 sentences for the primary analysis, at least 500 sentences for the secondary analysis, and at least 250 for each additional analysis. Every sentence was tagged by our team of human annotators.

The main challenge that we faced is that it is often extraordinarily difficult to identify a sufficient number of naturally-occurring Hebrew sentences attesting to the non-primary homograph analyses. In many cases, the naturally-occurring ratio of these analyses is 100:1 or worse, meaning that the non-primary analyses only occur once in 100 or more sentences. In such cases, prima facie, we would need to tag some 50,000 randomly-selected cases of the homograph in order to gather 500 cases of the alternate analysis. It would not be practical to tag this number of sentences for each of the 75 homographs. Therefore, we devised the following three-step process to allow us to efficiently gather a sufficient number of sentences attesting each possible analysis of each homograph:

1. For each homograph, we first gather 4000 randomly-selected sentences from an untagged Hebrew corpus. For each sentence, our annotation team determines the correct analysis of the homograph. This initial step virtually always provides us with a sufficient number of sentences attesting to the primary analysis, and sometimes for the secondary analysis as well. However, it often leaves us with an insufficient number of sentences regarding the non-primary analyses.

2. In order to find additional sentences for the non-primary homograph analyses, we train classifiers to identify a wide net of potential candidate sentences. For these classifiers, we use the known cases of the primary homograph analysis (identified in the previous step) as the first class, and we use various proxies to represent the opposing class. The proxies used for the opposing class include the following: (A) Hebrew words which unambiguously function morphosyntactically in a manner that is different from the morphosyntactic function of the primary class; (B) Hebrew words whose word2vec embedding is maximally distant from that of the homograph in question (because, in practice, for homographs of skewed proportions, the word2vec embedding of the word tends to represent the primary usage); (C) Randomly-selected Hebrew words. These classifiers are trained with a BiLSTM encoding of the word2vec embeddings of the four neighbors on either side of the target word (not including the target word itself). Essentially, the point of these classifiers is to identify a generous selection of cases in which the homographs may possibly be functioning in a way that is different than their primary usage.

3. We run these classifiers on all instances of the target homograph in our untagged Hebrew corpus. Cases that are classified as the opposing class are sent to our annotators for human verification. In practice, these proxy-based classifiers reach a precision of 15-30% on the task of identifying sentences in which the target homographs function according to a non-primary analysis. Thus, on average, for any given homograph, tagging an additional 2222 sentences allows us to collect 500 relevant cases of the secondary analysis.

Our procedure ensures diversity while keeping the process cost-effective. All in all it took approximately 1500 hours of annotation time to amass the sentences in the present dataset.

C Appendix C: Computational Equipment

We performed all computations on a desktop workstation with an i9-10980XE processor and 256GB of memory. This system enabled us to run 36 experiments in parallel (the processor contains 18 hyperthreaded cores), and thus we were able to complete all of the relevant experiments and computations over the course of several weeks of calendar time.

D Appendix D: Neural Network Implementation Details

All BiLSTMs and MLPs were trained using dynet (http://dynet.io/). All MLPs are constructed with a hidden layer of size 100, and with two layers. We train with the Adam optimizer at a learning rate of .001, for 3 epochs.

Our word2vec embeddings for Hebrew are of size 100. We trained them on a 500M-word Hebrew corpus using word2vecf (`https://github.com/BIU-NLP/word2vecf`, adding position info to context words, and excluding words with a frequency of less than seven. All in all, our word2vec vocabulary covers 94.6% of the tokens in our dataset. For the out-of-vocabulary words, we use a trainable UNK parameter in place of the word2vec embedding, which is trained from scratch for each “word expert” classifier.

As per the "word expert" paradigm, a completely separate MLP is trained for each homograph. In each case, the possible homograph analyses are each treated as a possible class for prediction, and the MLP is trained to choose from among those classes. Thus, for instance, if the homograph has two analyses, we train an MLP to predict one of the two classes; if the homograph has three analyses, then we train an MLP to predict one of the three classes; and so on.

For the Probing Scenarios based on centroid classification, we proceed as follows. For each of the homographs, given the training sample size (100, 50, 25, etc.), we randomly select that amount of training sentences for each of the possible analyses of the homograph. We calculate the centroid for each of the analyses by averaging the embeddings of the target homographs within the corresponding training sentences. The remainder of the available sentences for the homograph forms the test set. We classify them by calculating the dot product of the embedding of the target homograph in each given test sentence with the centroid of each of the homograph analyses. We run this process 200 times, each time selecting a different random set of training sentences. The values plotted in Figure 6 reflect the average of the F1 scores across these 200 rounds. For the corresponding MLP-based experiments presented for comparison in the aforementioned table, we follow an analogous procedure, across 10 rounds.

Parameter-Efficient Tuning with Special Token Adaptation

Xiaocong Yang†*, **James Y. Huang**‡, **Wenxuan Zhou**‡ and **Muhao Chen**‡
†Tsinghua University; ‡University of Southern California
`yangxc.18@sem.tsinghua.edu.cn`;
`{huangjam,zhouwenx,muhaoche}@usc.edu`

Abstract

Parameter-efficient tuning aims at updating only a small subset of parameters when adapting a pretrained model to downstream tasks. In this work, we introduce PASTA, in which we only modify the special token representations (e.g., `[SEP]` and `[CLS]` in BERT) before the self-attention module at each layer in Transformer-based models. PASTA achieves comparable performance to full finetuning in natural language understanding tasks including text classification and NER with up to only 0.029% of total parameters trained. Our work not only provides a simple yet effective way of parameter-efficient tuning, which has a wide range of practical applications when deploying finetuned models for multiple tasks, but also demonstrates the pivotal role of special tokens in pretrained language models.[1]

1 Introduction

Built upon a pretrained language model (PLM; Devlin et al. 2019; Liu et al. 2019; Yang et al. 2019; Chowdhery et al. 2022), many of the recent NLP systems are developed based on task-specific finetuning. In this way, the PLM effectively leverages the task-agnostic knowledge captured during self-supervised pretraining and adapts itself to downstream tasks. However, full finetuning poses a challenge to model deployment under multi-task, memory-limited scenarios, where we need to train and store a separate full-sized model for each substantially distinct task. As an alternative, parameter-efficient tuning (Ding et al., 2022) aims at only updating a small number of parameters when adapting PLMs to downstream tasks while making most of the model parameters fixed and shared among tasks, thus reducing memory usage.

In this paper, we propose **PA**rameter-efficient tuning with **S**pecial **T**oken **A**daptation (PASTA), where we only add trainable vectors to hidden representations of *special tokens* [2] at each layer before the multi-head attention module in Transformer-based PLMs. Our work is motivated by the role of special tokens in PLMs. First, special tokens such as `[CLS]` collect information from the whole input sequence and are typically regarded as the global text representation (Devlin et al., 2019). For sentence-level tasks such as GLUE (Wang et al., 2018), a common practice is to add a new classifier head based on the `[CLS]` representation in the last model layer. Thus, if trained properly, by updating the `[CLS]` representations, we can approximate the result of the *information collection* process in PLMs. Second, many attention heads in PLMs follow a vertical pattern[3], where the attention scores are mostly allocated to either the `[CLS]` or `[SEP]` token (Clark et al., 2019; Kovaleva et al., 2019), as

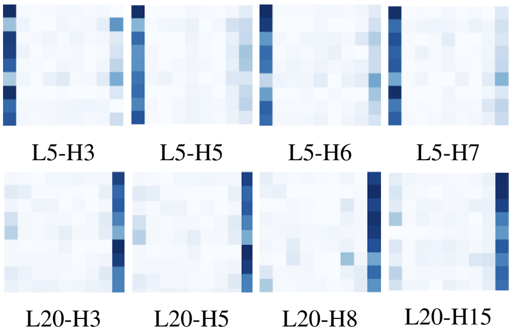


Figure 1: Examples of vertical attention heads in the 5-th and 20-th layer of BERT-large with a random sample from CoLA (Warstadt et al., 2019) as input. Heads in the first row and second row assign most of maximal attention weights to `[CLS]` and `[SEP]` respectively. See Appx. §C for the full attention map.

---

[*] Work done when visiting USC.

[1] Our code is publicly available at: `https://github.com/luka-group/PASTA/`

[2] WLOG, we use the notation of special tokens `[CLS]` and `[SEP]` in BERT for the convenience of expression, while the method applies in the same way to other paradigms such as `<S>` and `</S>` in RoBERTa (Liu et al., 2019).

[3] Following Voita et al. (2019) and Yao et al. (2021), an attention head is regarded as *vertical* if at least 90% tokens assign maximal attention scores to either `[CLS]` or `[SEP]`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 865–872
May 2-6, 2023

illustrated in Fig. 1. Therefore, updates to special tokens can also be *disseminated* to other tokens during the forward pass through the vertical attention heads (Elhage et al., 2021), enabling the PLMs to adapt to both sentential and lexical tasks.

By tuning as few as up to 0.029% of the total parameters, PASTA achieves competitive performance on par with full finetuning and BitFit (Zaken et al., 2022) on GLUE (§4.2). It also outperforms P-tuning v2 (Liu et al., 2022) by 0.6% on CoNLL2003 with 20× fewer additional parameters (§4.3). The ablation study shows that we can further reduce trainable parameters to 0.009% with only a slight performance drop (§4.4), showing the merit of adapting special token representations.

2 Related Work

A recent survey (Ding et al., 2022) categorizes three types of *parameter-efficient tuning* methods. *Addition methods* (Houlsby et al., 2019; Lester et al., 2021; Liu et al., 2022) introduce a small number of additional trainable parameters while keeping those in the PLM unchanged. *Specification methods* (Zaken et al., 2022; Guo et al., 2021; Zhao et al., 2020) update a portion of parameters in the PLM while keeping others frozen. *Reparameterization methods* (Aghajanyan et al., 2021; Hu et al., 2021; Qin et al., 2021) modify PLMs' structures to parameter-efficient forms. Our method belongs to the addition-based methods and follows the basic settings of P-tuning v2 (Liu et al., 2022), where newly initialized hidden representations of tokens are inserted into each Transformer layer. Different from most prompt tuning methods that introduce new tokens, we add the introduced vectors to the hidden states of special tokens and keep the sequence length unchanged.

Previous works use probing tasks (Wu et al., 2020) and pruning methods (Prasanna et al., 2020) to study the roles of different modules inside BERT. It has been shown that functional specialization exists in BERT self-attention heads (Clark et al., 2019), and vertical attention heads[3] take up a large portion (Yao et al., 2021). Kovaleva et al. (2019) find that vertical attention heads are almost exclusively associated with attention to `[SEP]` or `[CLS]` tokens, and Clark et al. (2019) conclude that heads in early layers often attend to `[CLS]` while in middle layers attend to `[SEP]`. In this work, we demonstrate that adapting hidden representations of special tokens is sufficient to bring the performance of PLMs to the level of full finetuning.

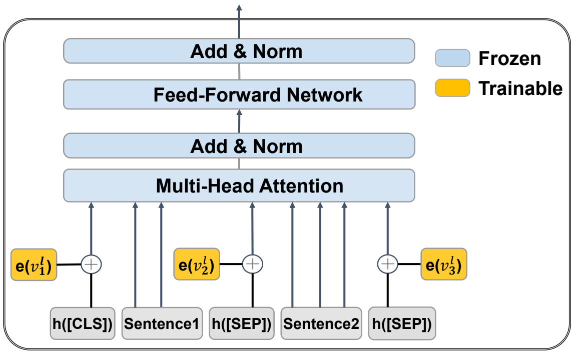


Figure 2: Architecture of PASTA layer in Transformer. Skip-connections in Transformers are not shown for brevity. At layer l we add a trainable vector $\mathbf{e}(\mathbf{v}_p^l) \in \mathbb{R}^d$ to the hidden representation of the p-th special token in the input sequence, and freeze the weights of the PLM.

3 PASTA

Given a large PLM, our goal is to develop a parameter-efficient tuning method that only updates a small set of parameters when adapting to a downstream task. To this end, we propose a simple yet effective method called PASTA, in which we train a hidden vector for every special token at each Transformer layer, along with a task-specific classifier, while freezing the parameters of the PLM.

3.1 Special Token Adaptation

The special token adaption is illustrated in Fig. 2. Although these adaptations are not directly applied to non-special tokens, changes in special token hidden states can be effectively disseminated to other tokens via self-attention during forward passes, thanks to the prevalence of vertical attention heads[3] in PLMs.

Specifically, denote the inputs to the l-th Transformer layer as $\mathbf{H}^l = \{\mathbf{h}_i^l\}_{i=1}^N, \mathbf{h}_i^l \in \mathbb{R}^d$, where N is the number of input tokens, d is the hidden size, PASTA modifies the inputs as follows:

$$\begin{aligned}\mathbf{H}_{\text{mod}}^l &= \{\mathbf{h}_i^l + \mathbf{m}_i^l\}_{i=1}^N, \\ \mathbf{H}^{l+1} &= \text{Trm}^l(\mathbf{H}_{\text{mod}}^l),\end{aligned}$$

where Trm^l is the l-th Transformer layer, $\mathbf{m}_i^l \in \mathbb{R}^d$ is our special token adaptation defined as follows:

$$\mathbf{m}_i^l = \begin{cases} \mathbf{0} & \text{if token } i \text{ is not a special token} \\ \mathbf{e}(\mathbf{v}_p^l) & \text{if token } i \text{ is the } p\text{-th special token} \end{cases}$$

	# Param	Parameter Consistency
Adapter	$\mathcal{O}(L \times d \times r)$	✓
P-tuning v2	$\mathcal{O}(L \times d \times T)$	✓
BitFit	$\mathcal{O}(L \times (d + m))$	✓
Diff-Prune	-	✗
PASTA	$\mathcal{O}(L \times d)$	✓

Table 1: Parameter complexity of PASTA and baselines. Here L and d refer to the number of layers and hidden size of the PLM. m and r refer to the intermediate size of FFN modules in Transformers and Adapters, respectively. T is the prompt length. Parameter consistency refers to whether the set of trainable parameters is consistent across different tasks (Zaken et al., 2022).

where $\mathbf{e}(\mathbf{v}_p^l) \in \mathbb{R}^d$ is the trainable vector added to the hidden representation of the p-th special token in the input sequence. During downstream task training, only those introduced hidden vectors for special tokens and the task-specific classifier are optimized, and the rest of model parameters are frozen.

3.2 Parameter Efficiency and Consistency

As shown in Tab. 1, PASTA achieves $\mathcal{O}(L \times d)$ parameter complexity[4] and updates as few as 0.015%-0.029% of the parameters compared to a full PLM when using BERT-large or RoBERTa-large as backbone. Unlike Adapter (Houlsby et al., 2019) that learns the transformation of all input tokens using a shared FFN, PASTA only learns the task-specific update of special token representations as a bias term, which significantly reduces the parameter capacity needed for adaptation on downstream tasks.

Meanwhile, the set of parameters introduced by PASTA is consistent across different tasks, making it efficient for hardware-based deployment (Zaken et al., 2022). On the contrary, in Diff-Prune, the parameter update is considered as a term of the loss function (Guo et al., 2021), resulting in different sets of updated parameters in distinct tasks.

4 Experiments and Results

We hereby study the downstream performance of PASTA and analyze the properties of introduced hidden vectors.

4.1 Experimental Setup

Baseline Methods. We compare PASTA with the following parameter-efficient tuning methods in prior studies. **Adapter** (Houlsby et al., 2019) introduces new feed-forward modules in Transformer layers while keeping original parameters of the PLM frozen. **BitFit** (Zaken et al., 2022) updates all bias terms in the PLM during finetuning. **Diff-Prune** (Guo et al., 2021) introduces L_0-norm penalty on the updated parameters to encourage sparsity of tuned parameters. **P-tuning v2** (Liu et al., 2022) prepends trainable hidden vectors before the input sequence at each layer while keeping the original PLM parameters frozen. **LoRA** (Hu et al., 2021) uses low-rank decomposition matrices to model the parameter updates.

Model Configuration. We conduct our experiments using BERT-large and RoBERTa-large (We also report experiments with BERT-base in Appx. §A). To facilitate comparison with baseline works, we take most of the experimental results from their original papers which are reported with either BERT-large or RoBERTa-large. Note that multiple `[SEP]` tokens in a single sequence (e.g., in sentence pair tasks like MNLI) are treated as different special tokens and have separate sets of trainable parameters, and the number of trainable parameters varies among downstream tasks according to the number of special tokens added. Details of training and hyperparameters settings are shown in Appx. §B.

4.2 GLUE Tasks

Task Setup. We evaluate PASTA on the widely used GLUE benchmark[5] (Wang et al., 2018). For the convenience of direct comparison, we use the same metrics as were used in baseline works (Devlin et al., 2019; Liu et al., 2019). For experiments with BERT, MRPC and QQP are evaluated using F1 score, STS-B is evaluated using Spearman's correlation coefficient, CoLA is evaluated using Matthew's Correlation, and the other tasks are evaluated using accuracy. For experiments with RoBERTa, STS-B is evaluated using Pearson's correlation coefficient, CoLA is evaluated using Matthew's Correlation, and the other tasks are evaluated using accuracy.

Results. Tabs. 2 and 3 report the performance of PASTA on GLUE benchmark with BERT-large and RoBERTa-large respectively. PASTA with RoBERTa-large achieves the same average score

---

[4] The number of special tokens are invariant to the scale of models, and are usually very small.

[5] Following previous work (Houlsby et al., 2019; Guo et al., 2021; Zaken et al., 2022), we exclude WNLI since BERT underperforms the majority class baseline (Devlin et al., 2019).

	%Param	RTE	CoLA	STS-B	MRPC	SST-2	QNLI	MNLI(m/mm)	QQP	Avg.
		acc.	mcc.	Spearman	F1	acc.	acc.	acc.	F1	
Full Finetuning*	100%	70.1	60.5	86.5	89.3	**94.9**	92.7	**86.7/85.9**	**72.1**	**81.6**
Adapter**	3.6%	71.5	59.5	**86.9**	89.5	94.0	90.7	84.9/85.1	71.8	81.1
Diff-Prune†	0.5%	70.6	61.1	86.0	**89.7**	94.1	**93.3**	86.4/86.0	71.1	81.5
P-tuning v2	0.29%	70.1	60.1	86.8	88.0	94.6	92.3	85.3/84.9	70.6	81.0
BitFit‡	0.08%	**72.0**	59.7	85.5	88.9	94.2	92.0	84.5/84.8	70.5	80.9
PASTA	**0.015%-0.022%**	70.8	**62.3**	86.6	87.9	94.4	92.8	83.4/83.4	68.6	80.9

Table 2: BERT-large model performance on GLUE benchmark test set. Lines with * and ** are results from Devlin et al. (2019) and Houlsby et al. (2019), and lines with † and ‡ are from Guo et al. (2021) and Zaken et al. (2022) respectively. We reimplement experiments of P-tuning v2 on GLUE benchmark with a prompt length of 20.

	%Param	RTE	CoLA	STS-B	MRPC	SST-2	QNLI	MNLI(overall)	QQP	Avg.
		acc.	mcc.	Pearson	acc.	acc.	acc.	acc.	acc.	
Full Finetuning*	100%	86.6	68.0	92.4	**90.9**	96.4	94.7	90.2	**92.2**	88.9
LoRA†	0.24%	**87.4**	68.2	**92.6**	**90.9**	96.2	94.9	**90.6**	91.6	**89.0**
PASTA	**0.015%-0.029%**	86.6	**69.7**	91.8	**90.9**	**96.8**	**95.1**	90.4	89.9	88.9

Table 3: RoBERTa-large model performance on GLUE benchmark. Lines with * are results from Liu et al. (2019), and lines with † are from Hu et al. (2021). We follow the metric settings of baselines and also report results on GLUE development set for the convenience of direct comparison.

	CoLA	RTE	MRPC	STS-B	CoNLL2003
PASTA	**65.4**	**76.2**	89.7	**90.8**	**94.0**
- w/o `[CLS]`	58.8	72.6	91.4	90.2	93.7
- w/o `[SEP]`	64.5	71.1	91.9	90.3	93.7
- shared vector	64.7	74.7	**92.1**	90.0	93.9
- classifier only	36.5	54.2	81.5	64.9	77.4

Table 4: Performance of ablation study with BERT-large on GLUE and CoNLL2003 development sets.

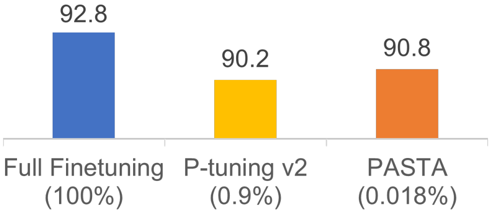


Figure 3: NER results on CoNLL03 with BERT-large (F1 score percentages are marked over the bars). Each method is labeled with the percentage of trainable parameter sizes with regard to full finetuning in parentheses.

as that of full finetuning over GLUE tasks. PASTA with BERT-large achieves an average score on par with BitFit using over 3× fewer trainable parameters and comparable results to other higher parameter complexity baselines. The results demonstrate that by leveraging the pivotal role of special tokens in PLMs, PASTA is able to effectively adapt the model to sentence-level tasks with significantly fewer parameters tuned than previous methods.

4.3 Named Entity Recognition

Task Setup. We experiment with the NER task on CoNLL2003 (Tjong Kim Sang and De Meulder, 2003). Following Devlin et al. (2019), we formulate NER as a token classification problem.

Results. As shown in Fig. 3, PASTA with BERT-large achieves an F1 score of 90.8% on CoNLL2003 test set, outperforming P-tuning v2 (Liu et al., 2022) by 0.6% with 20× fewer trainable parameters, while falling behind full finetuning by 2.0%. Nevertheless, the strong performance of PASTA compared to P-tuning v2 indicates that even though PASTA only directly adapts special tokens, the representations of all input tokens can still be properly tuned, supporting our hypothesis that vertical attention heads are able to disseminate adaptations in special token hidden states to other tokens.

4.4 Analysis

Ablation on choices of special tokens. To understand the effect of tuning different combinations of special tokens on downstream tasks, we further limit the additional parameter capacity of PASTA by only adapting either `[CLS]` or `[SEP]`, or share a common vector across all special tokens. Tab. 4 shows the performance of three ablated variants and a baseline that only tunes the classification head on top of a fixed BERT-large. In general, we observe

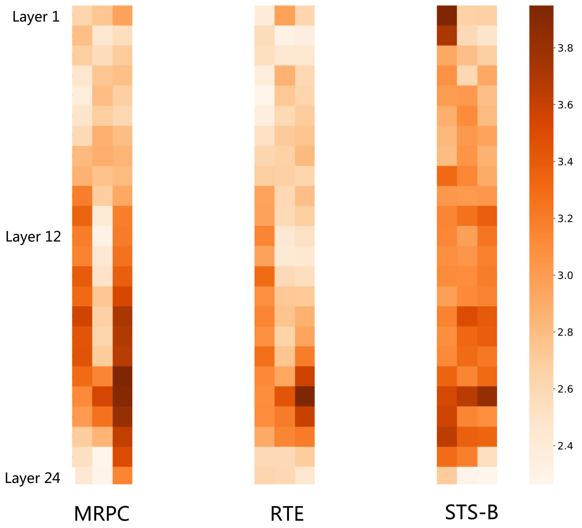


Figure 4: The distribution map of norms of introduced hidden vectors on MRPC, RTE and STS-B tasks with BERT-large. In each subgraph, the first column shows the norms of introduced vectors added to `[CLS]` at each layer, and the second and third columns are introduced vector norms at two `[SEP]` tokens respectively.

a decrease in performance on most tasks except on MRPC for three PaSTA variants, and performance degrades significantly if we do not adapt any special tokens. These results demonstrate the vital role of introduced hidden vectors for special tokens in PaSTA, while the best choice of special tokens to be adapted may vary depending on the task.

Norm distribution of introduced hidden vectors. Fig. 4 shows the norm distribution of introduced vectors on downstream tasks. The introduced hidden vectors learn the difference of special tokens between pretrained and adapted models, and thus norms of those vectors indicate the magnitude of parameter change at different layers. Similar to the pattern of parameter change during full finetuning (Kovaleva et al., 2019), PaSTA generally has larger norms of hidden vectors at layers closer to the output.

5 Conclusion

We present PaSTA, a parameter-efficient tuning method that only modifies special token representations at each Transformer layer when adapting to downstream tasks. Our approach is motivated by the observation that PLMs have a large amount of vertical attention heads that heavily attend to special tokens, and these heads disseminate value updates from special tokens to all of the other tokens. Experiments show that PaSTA achieves strong performance comparable to full finetuning on sentential and lexical tasks with high parameter efficiency. Our work not only provides an effective solution for parameter-efficient tuning, but also demonstrates the pivotal role of special tokens in PLMs.

Limitations

In this work we hypothesize that the vertical attention heads could play a role as "information disseminator" based on the theoretical analysis of Transformers (Elhage et al., 2021). However, we still have no direct approaches such as probing tasks and reverse engineering to prove this assumption. And since PaSTA relies on adaptation of special tokens, it cannot be applied to language models which do not pad special tokens to input sequences such as GPT-2 (Radford et al., 2019). For the empirical results, we choose GLUE benchmark and CoNLL2003 to study the performance on language understanding tasks. The effectiveness of PaSTA on language generation tasks has not been tested in this work due to limited bandwidth. Finally, similar to other parameter-efficient tuning methods, PaSTA suffers from a higher computational cost compared to full finetuning.

Acknowledgement

We appreciate the anonymous reviewers for their insightful comments and suggestions. Xiaocong Yang, Wenxuan Zhou and Muhao Chen are supported by the National Science Foundation of United States Grant IIS 2105329. This material is supported in part by the DARPA MCS program under Contract No. N660011924033 with the United States Office Of Naval Research, an Amazon Research Award, a Cisco Research Award and a subaward from NSF Cloudbank 1925001.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328, Online. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek

Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

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

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Demi Guo, Alexander Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4884–4896, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabsa. 2021. Unipelt: A unified framework for parameter-efficient language model tuning.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When BERT Plays the Lottery, All Tickets Are Winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3208–3229, Online. Association for Computational Linguistics.

Yujia Qin, Xiaozhi Wang, Yusheng Su, Yankai Lin, Ning Ding, Jing Yi, Weize Chen, Zhiyuan Liu, Juanzi Li, Lei Hou, Peng Li, Maosong Sun, and Jie Zhou.

2021. Exploring universal intrinsic task subspace via prompt tuning.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4166–4176, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*.

Xingcheng Yao, Yanan Zheng, Xiaocong Yang, and Zhilin Yang. 2021. Nlp from scratch without large-scale pretraining: A simple and efficient framework.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2226–2241, Online. Association for Computational Linguistics.

A Performance of BERT-base

Tab. 5 reports the performance of PASTA and a part of baselines with BERT-base as backbones on GLUE development sets. We do not put the result table in the main body of this work because most baselines did not report GLUE test set scores using BERT-base, and only three of them reported GLUE development set scores with BERT-base in their works. PASTA slightly underperforms baselines on average, while it outperforms other models on small datasets such as RTE and MRPC.

B Implementation Details

Our model is implemented based on Huggingface's Transformers. We optimize models with AdamW (Loshchilov and Hutter, 2017). We set the maximum input length to 128 and use a fixed random seed of 42 for all tasks. Experiments are done on NVIDIA RTX A5000 for averagely 3 hours per task, and distributed training is used for most tasks. Tab. 6 reports the best hyperparameters for model training. For hyperparameter search, we select learning rate from {5e-4, 1e-3, 2e-3, 2.5e-3, 3e-3, 4.5e-3, 5e-3, 7e-3} and the number of epochs from {50, 80, 100, 150}.

C Full Attention Map

Fig. 5 illustrates the attention maps of all heads in BERT-large. With a random sample from CoLA dataset as input (*"Fred watered the plants flat."*), there are 112 heads out of 384 in total being vertical heads[3].

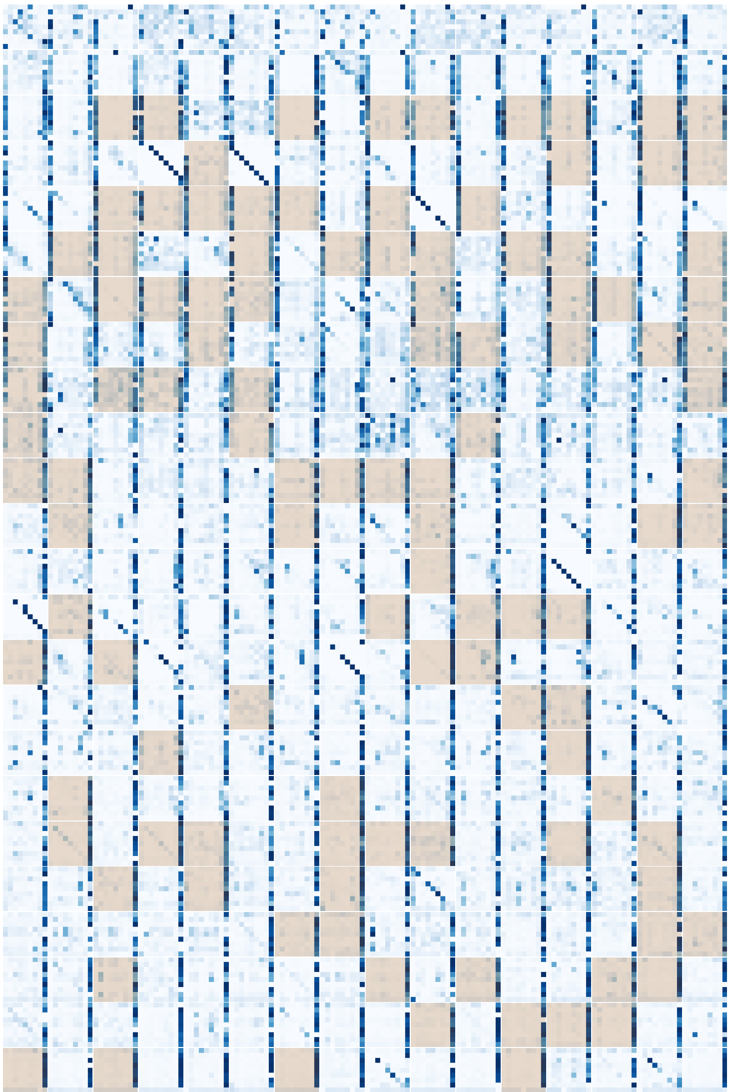


Figure 5: Full attention map of BERT-large pretrained with a random sample from CoLA as input. Rows and columns represent model layers and heads respectively, and darker color indicates larger weights. Vertical attention heads are highlighted in orange.

	%Param	**RTE**	**CoLA**	**STS-B**	**MRPC**	**SST-2**	**QNLI**	**MNLI(m/mm)**	**QQP**	**Avg.**
Full Finetuning *	100%	66.4	**62.1**	**89.8**	90.9	91.6	90.0	**83.2/ -**	**87.4**	82.7
Adapter*	0.81%	71.8	61.5	88.6	89.9	91.9	**90.6**	83.1/ -	86.8	**83.0**
BitFit†	0.8%	72.3	58.8	89.2	90.4	**92.1**	90.2	81.4/ -	84.0	82.3
PASTA	**0.015%-0.022%**	**73.6**	57.9	88.7	**91.5**	91.2	89.7	77.8/78.8	80.8	81.4

Table 5: PASTA with BERT-base model performance on GLUE benchmark development set. Lines with * and † refer to results from Mao et al. (2021) and Zaken et al. (2022) respectively.

	RTE	**CoLA**	**STS-B**	**MRPC**	**SST-2**	**QNLI**	**MNLI**	**QQP**	**CoNLL2003**
Learning rate	4.5e-3	5e-3	2e-3	2.5e-3	7e-3	2e-3	5e-4	5e-3	3e-3
Batch size	32×4	32×1	32×3	32×4	64×3	32×4	32×1	32×4	16×1
Number of adapted tokens	3	2	3	3	2	3	3	3	2
Training epochs	150	100	150	150	100	80	50	100	100
Best dev performance	76.2	65.4	90.8	89.7	93.9	92.2	83.7	87.9	94.1
Best epochs	121	63	109	136	99	42	49	97	89

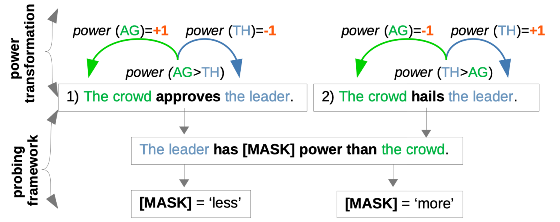
Table 6: PASTA with BERT-large training details for GLUE and CoNLL2003 tasks. Distributed training on multiple GPUs is used when avaiable for less training time.

Probing Power by Prompting: Harnessing Pre-trained Language Models for Power Connotation Framing

Shima Khanehzar[†] **Trevor Cohn**[†] **Gosia Mikolajczak**[⋆] **Lea Frermann**[†]
[†] School of Computing and Information Systems, The University of Melbourne
[⋆] The Global Institute for Women's Leadership, The Australian National University
`skhanehzar@student.unimelb.edu.au`
`{tcohn,lfrermann}@unimelb.edu.au`
`Gosia.Mikolajczak@anu.edu.au`

Abstract

Subtle changes in word choice in communication can evoke very different associations with the involved actors. For instance, a company 'employing workers' evokes a more positive connotation than the one 'exploiting' them. This concept is called *connotation*. This paper investigates whether pre-trained language models (PLMs) encode such subtle connotative information about *power differentials* between involved entities. We design a probing framework for power connotation, building on Sap et al. (2017)'s operationalization of connotation frames. We show that zero-shot prompting of PLMs leads to above chance prediction of power connotation, however fine-tuning PLMs using our framework drastically improves their accuracy. Using our fine-tuned models, we present a case study of power dynamics in US news reporting on immigration, showing the potential of our framework as a tool for understanding subtle bias in the media.[1]



Prelude	The crowd approves the leader.
Hypothesis	The leader has MASK power than the crowd.
Connotation	$AG \succ_{power} TH$
Prelude	The crowd hails the leader.
Hypothesis	The leader has MASK power than the crowd.
Connotation	$AG \prec_{power} TH$

Figure 1: Top: Connotation frames from Sap et al. (2017)'s dataset for the predicates 'approve' and 'hail', which evoke different power differentials between arguments 'crowd' and 'leader'. Predicate 'approve' implies that the leader needs approval of the *powerful crowd*; while 'hail' suggests that the crowd praises the *powerful leader*. Colored arrows show how we map from predicate- to entity-centric connotation. Bottom: Our PLM probing framework comprising a *prelude* (introducing predicate and arguments) and a *hypothesis* (expressing the implied power differential).

1 Introduction

An author's choice of words evokes associations and reactions from a reader that go beyond the literal meaning they express. This underlying level of meaning is called connotation and often carries social, cultural, or emotional implications for listeners or readers (Sonesson, 1998). In high-stakes settings such as opinion polls or news reporting, it has been shown that subtle changes in word choices can influence responses or opinions (Kahneman and Tversky, 1984; Rashkin et al., 2016). For example, the choice of the term 'undocumented workers' vs. 'illegal aliens' to describe immigrants can elicit different levels of prejudice toward that group (M.S., 2010).

Different connotations can inject multiple layers of meaning into a word, phrase, or passage. This paper focuses on *connotations of power* arising from descriptions of entities. Power dynamics are omnipresent at all levels of society, between individuals, groups, political actors, or institutions, and the subjective power of an entity can be expressed through the choice of words used to describe their actions (Sap et al., 2017). For example, in Figure 1, sentence (1) implies the leader requires approval of the *powerful crowd*; while sentence (2) implies the crowd praises the *powerful leader*.

Following the successful application of *probing* pre-trained language models (PLMs) to a variety of tasks ranging from grammatical structure (Koto et al., 2021; Kulmizev et al., 2020; Kassner and Schütze, 2020; Sinclair et al., 2022) to commonsense reasoning (Lin et al., 2020), we extend the probing paradigm to connotative power and study

---

[1] Source code is available at https://github.com/shinyemimalef/Probing-Power-by-Prompting

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 873–885
May 2-6, 2023

the extent to which PLMs can be used to reliably predict power dynamics between two entities.

We do so by drawing on the recently proposed notion of *connotation frames* (Rashkin et al., 2016; Sap et al., 2017; Ma et al., 2020), a structured framework that captures connotative associations evoked by a predicate (verb) about its agent (subject) and theme (object). Figure 1 illustrates the concept of the *power connotation frame*, which for a given predicate predicts whether the agent (subject) has more, less, or equal power compared to the theme (object). Specifically, we devise a probing framework to test PLMs for this formalization of connotative power which carefully controls for confounds relating to the specific agents and themes and probe structure. We show that, while PLMs cannot reliably predict power dynamics in a zero-shot setting, it is possible to fine-tune PLMs using our probes to predict power connotation with close to 80% macro F1.

Using our fine-tuned PLMs, we present a case study of power dynamics in US news reporting on the topic of immigration. News reporting is a particularly prominent example of documents where power dynamics are often at the core of the issue under discussion, and there is often an intrinsic motivation, especially in partisan news outlets, to present information in a biased way. We draw connections between power dynamics and emphasis framing and surface subtle bias in the context of immigration. In sum, our main contributions are:

- We propose a method to (i) disentangle connotation frames implied by the predicate from its arguments and the sentence structure; and (ii) quantify predicate connotation frames in PLMs.
- We probe the zero-shot ability of common PLMs to capture connotation frames and find poor performance, suggesting that this subtle signal is not captured in the representations.
- We show that the power prediction performance of PLMs can be drastically improved by fine-tuning on a small set of labeled instances, achieving F1 scores close to 80%.
- Using the best model, we present a case study on the news reporting on immigration and analyze how the power of immigrants and immigration services are portrayed in US news outlets with different political leaning.

2 Background

In this section, we provide an overview of the relevant literature on connotative framing and its formalization, probing PLMs, and how connotative framing and PLMs combined can be harnessed to study media bias.

2.1 Connotation Frames

Connotation frames were introduced by Rashkin et al. (2016) as a formalism for examining the sentiment and presupposed facts about actors and themes, as implied by the actions and events that they engage in (i.e., their predicates). The original framework cover several connotations, incldinging the writer's perspective (e.g., being sympathetic/antagonistic towards the agent), effect (e.g., the theme has been hurt), and the mental states projected onto agents and themes (e.g., being unhappy). Later work by Sap et al. (2017) extended the set of connotation frames to include 'power', which denotes the relative authority levels of the agent and theme implied by the predicate, and 'agency' defined as the capability of the agent to progress their own narrative. While our paper focuses on *power*, we note that our methodology can be extended to other connotation dimensions.

Sap et al. (2017) published a data set of English verbs manually annotated with power levels: The agent has either more power (AG$\succ_{power}$TH; the writer implies the agent has a level of control over the theme), or less power (AG$\prec_{power}$TH; the theme is implied to be more authoritative). We use a subset of their data in this work.[2] Unlike previous approaches to connotation frame prediction, which used annotated data to train supervised classifiers (Field et al., 2019; Rashkin et al., 2016), we use the method of probing to study and extend PLMs' ability to predict connotative power.

In earlier application of connotation frames to study various biases, researchers studied entities' depicted sentiment polarities in news articles (Rashkin et al., 2016), gendered bias in movies Sap et al. (2017), or people portrayals in #metoo stories Field et al. (2019). In our case study, we connect power connotation with emphasis frames (Card et al., 2015) for a more nuanced analysis of media bias. We show that our method is applicable to complex input sentences, diverging from settings in prior work, which largely considered isolated,

---

[2] The original data also includes an 'equal power' option, which we do not use here.

short text snippets (such as (agent,predicate,theme) tuples), and abstracted away from specific entities by using placeholders.

2.2 Probing PLMs

Probing language models for different types of knowledge has become a widely-used approach to understanding the knowledge encoded in PLMs (Liu et al., 2023). Typically, a PLM is presented with a prompting input and either completes the input or predicts the most likely token to fill in a masked position. Auto-prompting (Shin et al. (2020); PLMs generating prompts) and continuous prompting (Qin and Eisner (2021); prompting with an embedding) have been proposed recently, reducing the requirement of natural prompt engineering, but also transparency. While probing is a flexible framework, previous work showed that results can be sensitive to prompt wording and structure (Elazar et al., 2021a), and recommended considering such confounds by careful construction of controlled probe sets. Probing can be used as a zero-shot knowledge querying strategy or a way of fine-tuning to train PLMs to more accurately complete probes (Liu et al., 2023). Here, we extend probing to subtle connotative power associations arising from transitive verbs. Following previous work (Trichelair et al., 2019; Elazar et al., 2021b), we introduce several layers of controls, including paraphrased prompts and controlled sets of entities (i.e., predicate arguments) to ensure our findings are robust.

2.3 Media Bias and Framing

Automatically predicting framing bias has attracted recent attention in the NLP community, ranging from article-level frame prediction (Card et al., 2016; Field et al., 2018; Khanehzar et al., 2019, 2021) to tweet analysis (Mendelsohn et al., 2021). Previous works (Card et al., 2016; Khanehzar et al., 2021) showed characteristics of entities could help predict the article-level emphasis frames. In a case study (§ 7) on the topic of immigration in major US news outlets, we analyze the interplay of power connotation and article-level emphasis frames. Immigration has been a contested issue in the US, with the proponents and opponents actively trying to steer public opinion towards their stance by emphasizing selective and often simplified aspects of the topic (Farris and Silber Mohamed, 2018; Lawlor and Tolley, 2017; Ommundsen et al., 2014).

3 Probing PLMs for Connotative Power

We propose a probing framework to assess how much connotative information PLMs acquired during pretraining, including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and ALBERT (Lan et al., 2020). Similar to the masked token prediction objective during PLM pre-training, we have PLMs predict a masked token which in context reveals the power dynamics between an agent and theme in a given sentence. Our probe formulation is closely tied to Sap et al. (2017)'s annotation instructions to align the ground-truth data and information elicited from the PLMs.

We directly query the underlying masked language model in PLMs to compute power connotations associated with predicates. In particular, we formulate our probing template reflecting the definition of the power connotation:

Probe 1 *AG P TH. TH has MASK power than AG.*

Here, P is a placeholder for a predicate (details in §5.2). AG and TH are placeholders for concrete entities in the agent and theme position, which during probing and fine-tuning, are instantiated with common English names (details in §3.1). In the remainder of this paper, we will refer to the first sentence of our template as *prelude* which introduces the verb and associated entities; and to the second sentence as *hypothesis*. Figure 1 (bottom) shows two instatinated probes.

The PLMs are probed with the context of an instantiated probe with $\{P{=}p, AG{=}ag, TH{=}th\}$:

$$P(\text{MASK}{=}m|p, ag, th),$$

for the predicted probability of two possible values in the MASK position, namely the masked target word taking value $m{\in}\{\text{'more'}, \text{'less'}\}$. For example, if $P(m{=}\text{'more'}) > P(m{=}\text{'less'})$, the PLM predicts higher power of theme (th) than agent (ag) of the verb p, ($\text{AG}{\prec}_{\text{power}}\text{TH}$). To reduce the impact of specific instantiations of entities, we compute the final power connotation score for p as the average score over a large numger of entity combinations in the AG and TH positions:

$$S^{p}_{\text{probe}}(m) = \frac{1}{K} \sum_{\substack{ag \in A \\ th \in T}} P(\text{MASK}{=}m|p, ag, th) \tag{1}$$

where A and T are the set of all candidate entities for the AG and TH positions, respectively, and $K{=}|A||T|$.

3.1 Removing confounds

As discussed in §2.2, PLM probing is susceptible to confounds, including probe structure and choice of arguments, both of which can effect connotation frame prediction. We introduce measures to control three confounds (C1–C3).

C1: Probe structure In addition to Probe 1 we propose a second, semantically equivalent, probe with different structure.

Probe 2 *AG P TH. TH is MASK powerful than AG.*

We use different probe formulations to ensure that the model predictions are not incidental to a particular probe wording; and that the model is *consistently correct*. To this end, we adapt Elazar et al. (2021b)'s group evaluation (§5.3) score which only accepts a model prediction as correct if the correct class is predicted for multiple variants of the probe.

C2: Entity semantics We aim to capture power differentials as implied by the predicates that relate to two entities. As such, we do not want the predicted scores to be impacted by intrinsic power associated with the entities (e.g., high power 'presidents' vs. low power 'immigrants'). To this end, we constrain our entities to common English names. We use 16 female and male names from Nosek et al. (2002),[3] because (1) unlike general groups or entities (immigrants, agencies, politicians, etc.) names are largely free of a priori connotation which could impact our connotation scores; (2) all names are in the PLM vocabulary, removing discrepancies in subword tokenization; and (3) the set of names has been tested in implicit association tests (IAT and WEAT) and calibrated to only differ in gender, removing confounding effects of class or frequency (Jentzsch et al., 2019). The same set of names has been previously used in the context of gender bias mitigation (Gupta et al., 2022). To remove the gender confound we instantiate all possible $\{ag, th\}$ tuples from our 16 names, and reporting per-verb power connotations, obtaining $16 \times 15 = 240$ name combinations for each of our two probe templates, resulting in 480 instantiated probes per verb. We finally average probe predictions as in equation 1.

C3: PLM representation bias PLMs encode demographic biases, including stereotypical gender associations (Sun et al., 2019). Thus, in addition to controlling the entity set, we normalize power prediction scores by the PLM's a priori power assessment of the agent and theme names in the absence of a predicate. This normalization is also important for our case study in §7, where we use our probing framework with a fine-tuned PLM on less constrained contexts involving real-world entities in news articles. Inspired by previous work (Trichelair et al., 2019; Elazar et al., 2021b), we subtract from our connotation score the a priori relative power between the agent and theme in the absence of a specific predicate connecting them. To do so, we predict the MASK token given only the *hypothesis*:[4]

Prior 1 *TH has MASK power than AG.*

Like for the full probes, we obtain the probability of m with $m \in \{\text{'more', 'less'}\}$ from our PLMs and average over all entity combination, obtaining S_{prior}:

$$S^{p}_{\text{prior}}(m) = \frac{1}{K} \sum_{\substack{ag \in A \\ th \in T}} P(\text{MASK}{=}m | p, ag, th)$$

where P is computed using the Prior 1 template.

3.2 The Power Probing Score

The final power probing score (PPS) is the difference of the log predicted probability of the candidate MASK terms by the prior and the full probe:

$$\text{PPS}^{p}(m) = \log S^{p}_{\text{probe}}(m) - \log S^{p}_{\text{prior}}(m),$$

where $m \in \{\text{'more', 'less'}\}$. We finally derive a connotation score (CS) as a binary indicator variable:

$$CS^{p} = \begin{cases} +1 & \text{if } PPS^{p}(\text{'less'}) > PPS^{p}(\text{'more'}) \\ -1 & \text{otherwise,} \end{cases} \tag{2}$$

noting that CS=+1 directly corresponds to Sap et al. (2017)'s (AG$\succ_{power}$TH) and CS=-1 to (AG$\prec_{power}$TH).

4 Fine-tuning PLMs for connotative power prediction

As we will show in §6.1, PLMs do not encode power connotation naturally (in the zero-shot setting). We therefore aim to instill connotative knowledge into PLMs through task-specific fine-tuning.

---

[3] Male: {John, Paul, Mike, Kevin, Steve, Greg, Jeff, Bill}; Female: {Amy, Joan, Lisa, Sarah, Diana, Kate, Ann, Donna}

[4] See Table 3 in the Appendix for a complete list of priors.

Probe 1	AG P TH. TH has MASK power than AG.
No-Ent 1	P. has MASK power than.
Part-Sent 1	TH has MASK power than AG.

Table 1: Probe 1 (top) and two derived baseline templates, where agent (AG), theme (TH), and predicate (P) are instantiated as explained in §3.1. Probe 2 baselines are constructed analogously (cf., Appendix Table 3).

We use the probes introduced in §3 to construct a connotation prediction task as masked token prediction-based fine-tuning. The input to the model is the instantiated probe with instantiated ag, th, and p and with the target masked word m. The model is trained to predict 'more' or 'less', in correspondence with the gold standard power connotation value for the predicate p, CS^p. Training then proceeds as normal, backpropagating gradients through the full model, allowing the model to focus better on aspects of the encoding related to power connotation.

5 Experimental Setup

5.1 Baselines

We include a random baseline and a majority-class baseline to compare with PLMs. In addition, we provide two baselines to investigate the underlying language models' biases towards context and arguments of template sentences, by constructing inputs that are structurally similar to our probes, but do not carry any signal towards power connotations. This allows us to test (1) the extent of bias in the PLMs measured by deviation of the baselines from a random model; and (2) how much actual power connotative knowledge (over and above biases) the models contain in a zero-shot setting, or after fine-tuning, by comparing the respective models tested with the probes against these baselines. Both baselines are adapted from prior work on controlled debiasing (Elazar et al., 2021b).

No-Entity Baseline This baseline quantifies a priori PLM bias towards P in the absence of any entity. Probe 1 is reduced to No-Ent 1 in Table 1 (middle). An unbiased model should predict 'more' and 'less' with equal probability as the prompts contain no signal of the power differential.

Partial-Sentence Baseline This baseline quantifies prior biases related to the arguments by removing the predicate. We reduce the Probe 1 to only the *prelude* (similar to our prior in §3), see Part-Sent 1 in Table 1. Again, there is no information about power, and we expect an unbiased model to show random performance. Both baselines predict CS^p analogous to equations 1 and 2, but conditioning on their respective contexts.[5]

5.2 Predicate Selection

We use a subset of Sap et al. (2017)'s dataset of power-annotated verbs, keeping only transitive verbs applicable to human actors and themes based on manual filtering. We retain 300 verbs, with 67% of AG$\succ_{power}$TH and 33% verbs with AG$\prec_{power}$TH.

5.3 Evaluation metrics

We report class-wise and macro-averaged F1 scores to provide a detailed view of model performance and acknowledge the label skew in the ground truth data. We report standard (single-instance) evaluation for each test probe separately, resulting in 200 verbs × 2 probes = 400 instances; as well as the stricter grouped evaluation discussed below.

Group evaluation The more stringent *group evaluation* ensures that predictions are consistent across our semantically equivalent probes. A prediction CS^p for a given predicate p is accepted as correct only if the predicted connotation is correct for both probe 1 and probe 2. We compute the connotation score for each (ag, p, th) with probe 1 and probe 2 templates, and assign the worst-performing score. This group evaluation lowers the chance of random or coincidental correct predictions.

6 Results

We first test PLMs for power connotation knowledge in a zero-shot setting (§ 6.1), before we turn to task-specific fine-tuning (§ 6.2). See Appendix A for parameter settings.

6.1 Zero-shot Setting

We perform five separate runs. Each time, we obtain a stratified sample of 200 verbs selected from the full dataset of 300 to test the model. We report mean and variance across the five runs.[6]

Table 2 (top left) shows all three PLMs in the *single evaluation* metric. The 'No-Entity' and 'Partial-Sentence' baselines degrade to the majority classifier, which demonstrates that the PLMs contain

---

[5] See Table 3 in the Appendix for Probe 2 baselines.

[6] This evaluation protocol mirrors the fine-tuning setup in §6.2, allowing the comparison of variance between settings.

		Single Evaluation			Group Evaluation
	Setup	**F1+**	**F1-**	**macro F1**	**macro F1**
	Random	57.20 ± 2.60	39.74 ± 2.90	48.49 ± 2.40	21.56 ± 2.46
	Majority	80.36	0	40.18	40.18
Zero-shot	No-Entity*	80.36	0	40.18	40.18
	Partial-Sentence*	80.36	0	40.18	40.18
	BERT-Prob	75.41 ± 1.00	49.51 ± 3.00	41.64 ± 1.30	32.67 ± 1.80
	ALBERT-Prob	75.22 ± 0.40	44.54 ± 1.40	59.88 ± 0.80	40.84 ± 1.71
	RoBERTa-Prob	49.72 ± 0.16	2.19 ± 1.18	25.95 ± 0.60	25.00 ± 0.39
Fine-tuned	BERT-No-Ent	46.01 ± 14.61	12.81 ± 6.14	29.41 ± 4.38	23.35 ± 7.49
	ALBERT-No-Ent	20.58 ± 2.36	34.17 ± 2.94	27.38 ± 0.47	18.91 ± 0.43
	RoBERTa-No-Ent	58.42 ± 4.69	22.34 ± 5.76	40.38 ± 0.70	29.74 ± 7.69
	BERT-Part-Sent	42.88 ± 12.44	10.62 ± 4.52	26.74 ± 8.84	20.02 ± 14.80
	ALBERT-Part-Sent	19.11 ± 27.02	46.17 ± 4.62	32.64 ± 11.20	16.48 ± 11.65
	RoBERTa-Part-Sent	72.68 ± 10.86	13.21 ± 18.69	42.95 ± 3.91	26.79 ± 18.94
	BERT-Prob	85.36 ± 0.96	51.59 ± 4.37	68.48 ± 2.62	66.30 ± 2.57
	ALBERT-Prob	84.45 ± 0.29	66.49 ± 0.12	75.43 ± 0.29	74.02 ± 0.28
	RoBERTa-Prob	86.56 ± 0.98	70.73 ± 0.86	78.65 ± 0.88	77.18 ± 1.51

Table 2: Class-wise and macro F1 score of *power* connotation frame predictions with both single- (left) and group evaluation (right) for random and majority baseline, zero-shot setting and after task specific fine-tuning. *Results for these baselines are identical across all PLMs (and to the majority baseline) as they consistently predicted 'more' in this setting. Results are the mean and standard deviation over 5× repeated random subsampling of 200 verbs as a test set.

significant biases. An unbiased model would show a random performance.

Overall, in the zero-shot setup using the probe templates, the ALBERT model outperforms the BERT and RoBERTa models and random baseline in macro F1. Surprisingly, RoBERTa performs very poorly compared to the other two PLMs. We speculate that this might be due to the next sentence prediction loss, which is not part of RoBERTa training but is included BERT and ALBERT, thus explaining their ability to learn short-range dependencies between adjacent sentences in the probe templates. The main finding in this experiment is that none of the models contain power connotation information, as their performances are all close to random. Next, we show that PLMs can be effectively fine-tuned on this task.

6.2 Fine-tuned Setting

We evaluate the performance of fine-tuned PLMs using our proposed masked predication based approach (§4). As in the zero-shot experiments, we perfome 5 separate runs, each time using a stratified sample of 100 verbs for model fine-tuning, and the remaining 200 verbs for testing. We report mean and variance across these five runs.

Table 2 (bottom) shows that the fine-tuned models perform significantly better than the zero-shot versions across architectures. This holds in both the *single* (left) and the stricter *group evaluation* (evaluation). In the group evaluation, we observe the performance drops slightly compared to single evaluation after fine-tuning, and to a lesser extent than in the zero-shot setting. This suggests consistency of model predictions, increasing our confidence that the PLMs indeed capture connotative power associations with the predicates. Although RoBERTa performed poorly in the zero-shot setting, it is the strongest model after fine-tuning.

All fine-tuned models outperform all baselines by a large margin. Furthermore, we compared the performance of our fine-tuned models with the best previously reported approach for power connotation prediction — the logistic regression model of Field et al. (2019). The logistic regression model obtains macro F1 of 60%, substantially lower than our fine-tuned results.

Next, we analyze the impact of the training set size, and the robustness and consistency of our probing score.

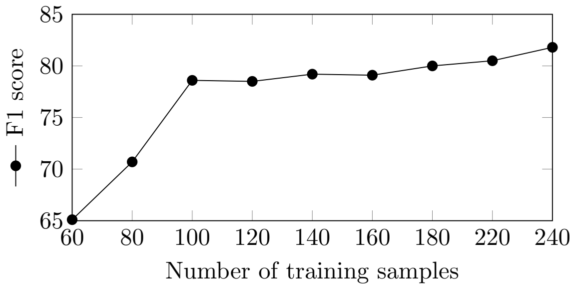


Figure 2: Performance (macro F1) of the fine-tuned RoBERTa model in single evalution with training sets of varying size.

Impact of the training set size Our main results are based on fine-tuning on only 100 labeled instances. We investigated the effect of varying the number of training samples from $\{60, 80, \ldots, 220, 240\}$, and a fixed test set of the remaining 60 instances for RoBERTa. The benefits of using more training samples plateaued after 100 samples as shown in Figure 2. Pre-trained models can efficiently learn to capture power connotation information from few training samples.

Robustness For each predicate, we calculated the variance of the probability of the masked token taking values 'more' or 'less', normalizing their scores to sum to 1 and averaging across all possible combinations of names in the agent and theme position (n=240). Overall, variance is low (mean=0.003). The predicates with the highest variance (0.005) include 'question' and 'reach', while those with lowest variance (0.001) include 'shut' and 'ruin'. Intuitively, the former (high-variance) verbs are more ambiguous than the latter.

Consistency We checked whether, given a predicate, the predicted filler for the masked token is the same across possible combinations of names in the agent and theme position. Across all test predicates, we find that 88% of the time the prediction is consistent among the 240 argument combinations. These final two analyses suggest that our probing score is not sensitive to the confounding variables of the agent or theme position or gender.

7 Case study

We employ our best-performing fine-tuned model RoBERTa to explore subtle power dynamics in US news reports on immigration. In particular, we study the power connotations implied in the descriptions (i.e., actions and events) involving prominent *entities* in the articles. In doing so, we draw connections between power connotation and emphasis frames (§7.1) as well as power connotation and issue stance (§7.2). We first describe our dataset and the entity extraction process, before explaining how we map the predicate-centric probing framework from §3 to an entity-centric measure.

Dataset We use the Media Frames Corpus (MFC, Card et al. (2015)), which contains 6.7k news articles about immigration,[7] manually labeled with one of 15 emphasis frames (including `Legality`, `Political`, `Economic`, ...), as well as an additional 42k unlabeled news articles. The articles were sourced from 13 U.S. news outlets published between 1969 and 2017. A subset of the immigration articles includes manual labels of 'stance' which indicates whether the article author is supporting, neutral about, or opposing immigration.[8]

Entity extraction We identify the most common entities within each article using an off-the-shelf transformer-based semantic role labeling model[9] (Shi and Lin, 2019) which has been previously used to identify key entities in MFC articles (Khanehzar et al., 2021). For each sentence in a document, we collect spans corresponding to the main verbs (`predicate`), and their first (ARG0) and second (ARG1) arguments, and then apply a coreference resolution model (Lee et al., 2018) to group the arguments into entities. We also use NER and string matching to find all mentions of each argument and consider only entities mentioned >3 times in an article. Figure 5 in the Appendix B shows the most common entities in the dataset.

Entity-level power connotation We apply our power probing framework as introduced in §3, and add a final step to map predicate-level CS^p to an entity-level CS. Similar to Field et al. (2019), and as illustrated in Figure 1, if $CS^p = +1$ (AG$\succ_{power}$TH), we consider connotation score of the agent as positive CS(AG)=+1 and the theme as negative CS(AG)=-1, and vice versa for predicates with $CS^p = -1$. This approach enables us to obtain power scores for entities in unannotated documents. To obtain the power score for each entity in each article, we average the power score

---

[7] The MFC covers four other issues, but immigration is the most prominent issue spanning the longest time period.

[8] Pro: 2740, Anti: 1685, Neutral: 982, None: 1350

[9] From AllenNLP, trained on OntoNotes5.0

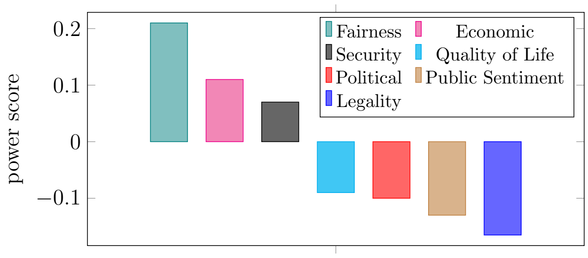


Figure 3: The power of 'immigrants' in articles with different emphasis frames (see legend). The majority of differences are significant, see Appendix C for details.

predicted for each mention of the entity (in the context of a transitive predicate) in that article. The entity's overall power score across the corpus is the average of its power scores across all articles.

7.1 The Power of Immigrants across Different Emphasis Frames

Immigration is a contested issue in the US with politicians, lobby groups, and news agencies selectively framing the topic in a way that supports their stance (Farris and Silber Mohamed, 2018; Lawlor and Tolley, 2017; Ommundsen et al., 2014). We examine how the news outlets portray the key entity involved in this discourse, namely 'the immigrants'. Specifically, we relate power connotations to emphasis frames in the MFC. We applied the method described above to all 6.7k frame-labelled immigration articles, and considered all mentions of the term 'immigrant(s)' and their co-referents.

Figure 3 shows that immigrants are depicted as relatively powerful when the articles are using the `Economic`, `Fairness and Equality`, `Security and Defense` frames, and fairly powerless when the articles are using the `Legality`, `Constitutionality`, `Jurisdiction`, `Public Sentiment`, `Quality of Life`, and `Political` frames. This suggests that news outlets generally approve of immigrants' role in contributing to the economy as confirmed by example mentions like 'Low-skilled immigrants *provide* cheap child care' and 'More immigrants than ever before *start* their own companies'. The gold standard power connotation label of *provide* and *start* is $\mathrm{AG} \succ_{power} \mathrm{TH}$, and our fine-tuned model predicted the same label with confidence for both cases.

Another example of depicting immigrants as relatively high in power are articles adopting the `Security and Defense` frame. These articles portray immigrants' power to disrupt the societal order. Examples include 'Immigrants *break* a security cordon', 'immigrants *overwhelm* the guards' and 'immigrants *overstay* their visas'. Conversely, articles using the `Fairness and Equality` frame tend to portray immigrants in an unfavorable light, albeit with high power associated with their actions, implying malicious intentions. Examples include 'immigrants *bring* crime and drugs', and 'detained immigrants *fabricate* accounts'.

These examples show that our operationalization of power complements the positive vs negative sentiment dimension, and in general, is not a stable trait of an entity, but rather depends on the more general frame adopted by an article. Additionally, in emphasis frames depicting immigrants as low in power, for example, in the `Legality` frame, immigrants are often associated with either positive or neutral actions such as 'immigrants *comply with* Federal laws', or 'immigrants *renounce* the citizenship of their native countries'. Among the frames depicting the immigrants as powerless, we noted that articles adopting the `Quality of Life` frame generally imply that immigrants, especially undocumented immigrants, face difficulties in life. Examples include 'immigrants *suffer* the alleged brutality', 'immigrants *fear* they will be hounded and deported'. Under the `political` frame, articles generally portray politicians' actions or relations towards immigrants, often attributing less power to immigrants. Examples include 'illegal immigrants *toil* for governor', 'Rudy accused Mitt *employing* illegal immigrants'.

7.2 Power of Immigrants vs. Immigration Services and Issue Stance

Immigration services such as ICE and INS[10] play a significant role in executing and designing immigration policies in the US and are prominent in the public discourse. Notorious for hardline approaches, these agencies are often criticized by the more liberal news outlets for abusing their legal power (Omokha, 2022). We compare the connotative power associated with *immigrants* and *immigration services* in news articles with a supportive, opposing or neutral stance on immigration, as manually labeled in the MFC.

Figure 4 shows, unsurprisingly, that immigrants are generally portrayed as less powerful than immigration services. At the same time, we observe

---

[10] Immigration and Customs Enforcement (ICE) and Immigration and Naturalization Service (INS)

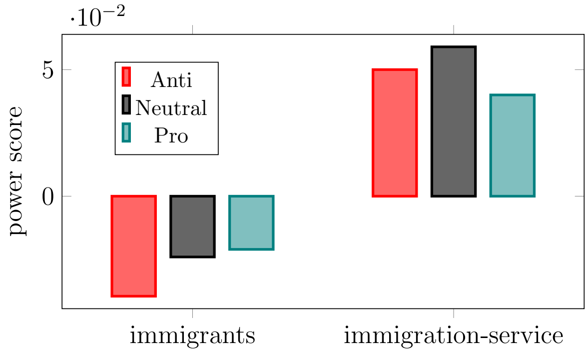


Figure 4: Power of ‘immigrants’ and ‘immigration services’ in articles with an anti-, pro-, and neutral- immigration stance. See Appendix C for significance results.

different trends for the two entities depending on the author’s stance on immigration: immigrants are portrayed as ‘most powerless’ in anti-immigration articles, but slightly less so in neutral and pro-immigration articles. In pro-immigration articles, immigrants more often take action (instead of being acted upon). Illustrative examples from pro-immigration articles include ‘immigrants *navigate* the byzantine rules of permanent residency’, and ‘immigrants *discover* the power of citizenship’, and ‘people should *join* immigrants in the continuing fight for civil rights and human dignity’. Examples of depictions of immigrants as powerless in anti-immigration articles include ‘officers *detain* a total of thirty Haitian immigrants’ and ‘immigration officials *arrest* illegal immigrants’. Immigration services, on the other hand, are portrayed as powerful in both pro- and anti-immigration articles, but less so in neutral ones. Immigration services in anti-immigration articles are often involved in strong actions toward immigrants: for instance, the predicates ‘arrest’ and ‘deport’ appear 1.8 times more often in anti-immigration articles than in pro-immigration articles.

8 Conclusion

We introduced a framework to (i) disentangle connotation frames implied by the predicate from its arguments and the sentence structure; and (ii) quantify predicate connotation frames in PLMs. Using the proposed framework, we investigated the capability of pre-trained language models to understand connotative information focusing on power dynamics between involved entities, both in a zero-shot setting, where performance was overall poor, and fine-tuning, which lead to drastic improvements. Our framework can be applied to investigate other connotation frames such as agency and sentiment, and their relationships to media bias, although defining the probing template may be challenging due to the subtle nature of connotation frames.

Finally, in a case study, we showed how our model can be used to detect subtle differences in the implied power dynamics between entities. Our findings highlight the potential of our framework as a tool for understanding subtle bias in the media. Future work could use our framework to analyze language in various forms of media, e.g. social media posts and TV programming, to identify patterns and trends in language usage for conveying power and other connotations.

It is worth investigating whether the improved performance in connotation predictions would enhance the model’s reading comprehension (Rajpurkar et al., 2016) capability at a higher level. Naturally, we would like to see that the model can infer which entity is more powerful or wields more authority without the text explicitly stating that fact. Previous work has shown that different layers in PLMs specialize in various meanings (de Vries et al., 2020). Future work should investigate the extent each layer in a PLM contributes to encoding connotation frames.

Our framework can also be used to study character roles by analyzing the verbs or verb phrases used to describe the characters and the connotations that they carry. For example, a news article about a political leader who is advocating for tougher immigration policies might mention the event of ‘cracking down on illegal immigration’, evoking a negative connotation and portraying the leader as a ‘villain’. A different article on the same entity might include an event of ‘helping refugees find safety’ assigning the leader the role of ‘hero’ with a more positive connotative association. Both articles frame the immigrants as ‘victims’. By analyzing the language used to describe characters and the connotations it carries, it is possible to gain insight into their framing and roles in the broader narratives around an issue.

9 Limitations

We identify several limitations and shortcomings in our work as potential areas for future work. While the proposed probing framework could be used to investigate other connotation frames, we focused only on the power dimension. We plan to extend our work to other connotation dimensions and explore their relationships with media bias.

While our case study demonstrates that predicate connotative information can be used to depict entity-level power connotation, we recognize that many other relevant features might do so. These include low-level features like adjectives and noun phrases, as well as high-level ones like the position of the entity in the article, and whether or not the entity is directly quoted. Future studies could compare how these different features contribute to predictions of connotation frames.

We acknowledge that our model, which predicts power dynamics on predicate signals alone, simplifies the construct of social power. This might lead to inaccurate predictions. Furthermore, the automatic analysis of portrayed power of real world entities or groups bears a risk of misuse to justify discriminatory practices or policies on the basis of the power portrayal in the media.

To mitigate these risks, it is essential to consider the potential consequences of the model's predictions and ensure they are used ethically and responsibly, including considering the potential impact on marginalized groups and taking steps to minimize any potential biases in the model's predictions. Additionally, human oversight and interpretation of the results is important, especially when they are used in decision-making processes that have a social impact.

Ethics Statement

The original news articles from MFC (Card et al., 2015) used in this work were obtained from Lexis Nexis under the institutional licence held by the University of Melbourne.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback and suggestions. This article was written with the support from the graduate research scholarship from the Melbourne School of Engineering, University of Melbourne provided to the first author. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne, established with the assistance of LIEF Grant LE170100200.

References

Dallas Card, Amber E. Boydstun, Justin H. Gross, Philip Resnik, and Noah A. Smith. 2015. The media frames corpus: Annotations of frames across issues. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 438–444, Beijing, China. Association for Computational Linguistics.

Dallas Card, Justin Gross, Amber Boydstun, and Noah A. Smith. 2016. Analyzing framing through the casts of characters in the news. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1410–1420, Austin, Texas. Association for Computational Linguistics.

Wietse de Vries, Andreas van Cranenburgh, and Malvina Nissim. 2020. What's so special about BERT's layers? a closer look at the NLP pipeline in monolingual and multilingual models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4339–4350, Online. Association for Computational Linguistics.

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

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021a. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. 2021b. Back to square one: Artifact detection, training and commonsense disentanglement in the Winograd schema. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10486–10500, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Emily M Farris and Heather Silber Mohamed. 2018. Picturing immigration: How the media criminalizes immigrants. *Politics, Groups, and Identities*, 6(4):814–824.

Anjalie Field, Gayatri Bhat, and Yulia Tsvetkov. 2019. Contextual affective analysis: A case study of people portrayals in online #metoo stories. In *Proceedings of the Thirteenth International Conference on Web and Social Media, ICWSM 2019, Munich, Germany, June 11-14, 2019*, pages 158–169. AAAI Press.

Anjalie Field, Doron Kliger, Shuly Wintner, Jennifer Pan, Dan Jurafsky, and Yulia Tsvetkov. 2018. Framing and agenda-setting in Russian news: a computational analysis of intricate political strategies. In *Pro-*

ceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3570–3580, Brussels, Belgium. Association for Computational Linguistics.

Umang Gupta, Jwala Dhamala, Varun Kumar, Apurv Verma, Yada Pruksachatkun, Satyapriya Krishna, Rahul Gupta, Kai-Wei Chang, Greg Ver Steeg, and Aram Galstyan. 2022. Mitigating gender bias in distilled language models via counterfactual role reversal. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 658–678, Dublin, Ireland. Association for Computational Linguistics.

Sophie Jentzsch, Patrick Schramowski, Constantin Rothkopf, and Kristian Kersting. 2019. Semantics derived automatically from language corpora contain human-like moral choices. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, page 37–44, New York, NY, USA. Association for Computing Machinery.

Daniel Kahneman and Amos Tversky. 1984. Choices, values, and frames. *American psychologist*, 39(4):341.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Shima Khanehzar, Trevor Cohn, Gosia Mikolajczak, Andrew Turpin, and Lea Frermann. 2021. Framing unpacked: A semi-supervised interpretable multi-view model of media frames. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2154–2166, Online. Association for Computational Linguistics.

Shima Khanehzar, Andrew Turpin, and Gosia Mikolajczak. 2019. Modeling political framing across policy issues and contexts. In *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*, pages 61–66, Sydney, Australia. Australasian Language Technology Association.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3849–3864, Online. Association for Computational Linguistics.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4077–4091, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Andrea Lawlor and Erin Tolley. 2017. Deciding who's legitimate: News media framing of immigrants and refugees. *International Journal of Communication*, 11(0).

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. 2020. PowerTransformer: Unsupervised controllable revision for biased language correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7426–7441, Online. Association for Computational Linguistics.

Julia Mendelsohn, Ceren Budak, and David Jurgens. 2021. Modeling framing in immigration discourse on social media. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2219–2263, Online. Association for Computational Linguistics.

Matthew R. Pearson M.S. 2010. How "undocumented workers" and "illegal aliens" affect prejudice toward mexican immigrants. *Social Influence*, 5(2):118–132.

Brian A Nosek, Mahzarin R Banaji, and Anthony G Greenwald. 2002. Harvesting implicit group attitudes and beliefs from a demonstration web site. *Group Dynamics: Theory, Research, and Practice*, 6(1):101.

Reidar Ommundsen, Knud S. Larsen, Kees van der Veer, and Dag-Erik Eilertsen. 2014. Framing unauthorized immigrants: The effects of labels on evaluations. *Psychological Reports*, 114(2):461–478. PMID: 24897900.

Rita Omokha. 2022. 'They don't have any humanity': Black immigrants in Ice custody report abuse and neglect.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Hannah Rashkin, Sameer Singh, and Yejin Choi. 2016. Connotation frames: A data-driven investigation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 311–321, Berlin, Germany. Association for Computational Linguistics.

Maarten Sap, Marcella Cindy Prasettio, Ari Holtzman, Hannah Rashkin, and Yejin Choi. 2017. Connotation frames of power and agency in modern films. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2329–2334, Copenhagen, Denmark. Association for Computational Linguistics.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Arabella Sinclair, Jaap Jumelet, Willem Zuidema, and Raquel Fernández. 2022. Structural Persistence in Language Models: Priming as a Window into Abstract Language Representations. *Transactions of the Association for Computational Linguistics*, 10:1031–1050.

Probe 2	AG P TH. TH is MASK powerful than AG.
Prior 2	TH is MASK powerful than AG.
No-Ent 2	P. is MASK powerful than.
Part-Sent 2	TH is MASK powerful than AG.

Table 3: Prior and baselines for Probe 2. The agent (AG), theme (TH), and predicate (P) are instantiated as explained in §3.1.

Göran Sonesson. 1998. Denotation and connotation. *Encyclopedia of semiotics*, pages 187–89.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Paul Trichelair, Ali Emami, Adam Trischler, Kaheer Suleman, and Jackie Chi Kit Cheung. 2019. How reasonable are common-sense reasoning tasks: A case-study on the Winograd schema challenge and SWAG. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3382–3387, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Parameter Settings

Zero-shot Setting We use the pre-trained model implemented in HuggingFace's Transfomers (Wolf et al., 2020) with base version[11] and default model parameters.

Fine-tuning Settings We perform stochastic gradient descent with mini-batches of 64 sentences. We use the Adam optimizer (Kingma and Ba, 2015) with the default parameters, except for the learning rate, which we set to 10^{-5}. We train the models for five epochs or until we reach convergence.

---

[11] 'bert-base-uncased', 'roberta-base', 'albert-base-v2'

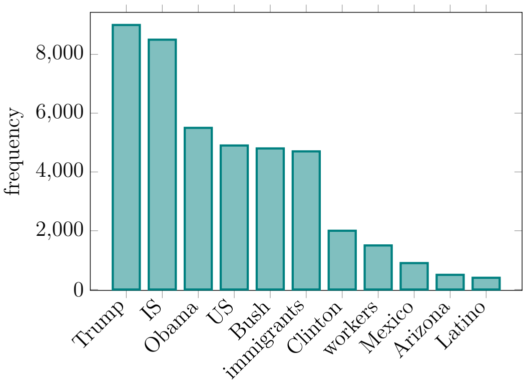


Figure 5: Most common entities such as Immigration Services (IS) and United States (US) in the immigration articles from the MFC (Card et al., 2015).

B Most Common Entities in MFC Immigration

To complement the list of entities extracted by SRL and coreference resolution, we also used NER to extract entities tagged as PERSON, and *string matching* to find all mentions of each entity. We consider only entities mentioned more than three times in each article as the most common entities within each article. Figure 5 shows the most common entities, with the number of their appearance in the immigration articles from the MFC dataset.

C Statistical Significance of the Power Scores in Case Studies

In the case studies in §7, the overall power scores of entities are reported by averaging the power score across the whole corpus. We test the significance of pairwise differences in power score using a z-test assuming a binomial distribution of labels (CS = +1 or -1). Below we discuss the result for each case study separately.

In case study in §7.1 we compare the power score of 'immigrants' in articles with different emphasis frames (as shown in Fig 7.1). The majority of comparisons of the power score are significant (as shown in Fig 4); for example, the p-value between frames `Fairness` vs. `legality` is 0.0004, and `Fairness` vs. `Public Sentiment` is 0.001. Overall, differences of power scores between frames of > 0.1 are significant at $p \leq 0.05$. The difference of power scores between frames `Economic` vs. `Security` are not significant ($p = 0.1$).

	F	E	S	Q	Po	Pu	L
F		*	*	*	*	*	*
E	*		-	*	*	*	*
S	*	-		*	*	*	*
Q	*	*	*		-	-	*
Po	*	*	*	-		-	*
Pu	*	*	*	-	-		-
L	*	*	*	*	*	-	

Table 4: Shows the comparisons of the power score of 'immigrants' in different emphasis frames including **F**`airness`, **E**`conomic`, **S**`ecurity`, **Q**`uality of Life`, **Po**`litical`, **Pu**`blic Sentiment`, and **L**`egality`. The * indicates the difference is significant at $p \leq 0.05$ (z-test) while '-' means insignificant.

In the case study in §7.2 the comparisons between power score of 'immigrants' vs. 'immigration services' in articles with different issue stances (as shown in Fig 4 as same color across blocks) are significant with a p-value of ≤ 0.005. Additionally, the comparison of the power score for each entity in different issue stances (pairwise comparison of bars within blocks) is almost significant with a p-value of $p \leq 0.05$.

Zero and Few-Shot Localization of Task-Oriented Dialogue Agents with a Distilled Representation

Mehrad Moradshahi **Sina J. Semnani** **Monica S. Lam**
Computer Science Department
Stanford University
Stanford, CA
`{mehrad,sinaj,lam}@cs.stanford.edu`

Abstract

Task-oriented Dialogue (ToD) agents are mostly limited to a few widely-spoken languages, mainly due to the high cost of acquiring training data for each language. Existing low-cost approaches that rely on cross-lingual embeddings or naive machine translation sacrifice a lot of accuracy for data efficiency, and largely fail in creating a usable dialogue agent. We propose automatic methods that use ToD training data in a source language to build a high-quality functioning dialogue agent in another target language that has no training data (i.e. zero-shot) or a small training set (i.e. few-shot). Unlike most prior work in cross-lingual ToD that only focuses on Dialogue State Tracking (DST), we build an end-to-end agent.

We show that our approach closes the accuracy gap between few-shot and existing full-shot methods for ToD agents. We achieve this by (1) improving the dialogue data representation, (2) improving entity-aware machine translation, and (3) automatic filtering of noisy translations.

We evaluate our approach on the recent bilingual dialogue dataset BiToD. In Chinese to English transfer, in the zero-shot setting, our method achieves 46.7% and 22.0% in Task Success Rate (TSR) and Dialogue Success Rate (DSR) respectively. In the few-shot setting where 10% of the data in the target language is used, we improve the state-of-the-art by 15.2% and 14.0%, coming within 5% of full-shot training.[1]

1 Introduction

While dialogue agents in various forms have become commonplace in parts of the world, their lack of support for most human languages has prevented access to the benefits they provide for much of the world. Commercial virtual assistants for example, only support a handful of languages, as extending their functionality to each new language is extremely costly, partially due to the need for collecting new annotated training data in that language.

In recent years, several non-English task-oriented dialogue (ToD) datasets have been created; they are either collected from scratch such as RiSAWOZ (Quan et al., 2020) and CrossWOZ (Zhu et al., 2020), paraphrased from synthetic sentences by crowdworkers such as BiToD (Lin et al., 2021), or manually translated from another language (Li et al., 2021b). All of these approaches are labor-intensive, expensive, and time-consuming; such investment is unlikely to be made for less widely spoken languages.

Cross-lingual transfer, i.e. using training data from other languages to build a dialogue agent for a specific language, seems especially appealing. An emerging line of work has employed machine translation of training data, and multilingual pre-trained neural networks to tackle this task (Sherborne et al., 2020; Li et al., 2021a; Moradshahi et al., 2023). However, work in ToD cross-lingual transfer has for the most part, focused on understanding the user input, namely Dialogue State Tracking (DST) and Natural Language Understanding (NLU). Other necessary parts of a dialogue agent like policy and response generation have mostly remained unexplored.

In this paper, we present a methodology for building a fully functional dialogue agent for a new language (e.g. English), by using training data in another language (e.g. Chinese) with little to no additional manual dataset creation effort. We found that despite prior efforts to improve modeling for existing ToD datasets, the dialogue representation used as input to these models, e.g. full dialogue history in natural language (Hosseini-Asl et al., 2020), is sub-optimal, especially when the training data is either scarce or created automatically us-

---

[1] Code can be accessed at `https://github.com/stanford-oval/dialogues`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 886–901
May 2-6, 2023 ©2023 Association for Computational Linguistics

ing noisy machine translation. We propose a new *Distilled* representation to fix the shortcomings of current representations. We also found that previously proposed entity-aware translation technique Moradshahi et al. (2023) to be inadequate. Our proposed technique effectively combines entity-aware neural machine translation with text similarity classifiers to automatically create training data for a new language. This paper explains all the ingredients we found useful, and motivates their use by performing extensive ablation studies.

The contributions of this paper are:

1. *A new state-of-the-art result for the BiToD dataset in both few-shot and full-shot settings on English* according to all of our 6 automatic metrics, including an improvement of 14.0% and 2.9%, respectively, in Dialogue Success Rate (DSR). In fact, using our Distilled representation, our few-shot model trained on only 10% of the training data, achieves similar results to the previous SOTA model trained on 100% training data.
2. *The first dialogue agent created in the zero-shot cross-lingual transfer setting*, i.e. starting from no training data in the target language. Our agent achieves 71%, 62%, 40%, and 47% of the performance of a full-shot agent in terms of Joint Goal Accuracy (JGA), Task Success Rate (TSR), DSR, and BLEU score, respectively.
3. *A concise dialogue representation designed for cross-lingual ToD agents*. The Distilled dialogue representation works well with our new decomposition of agent subtasks, making significant improvements possible.
4. *An improved methodology for automatic translation of ToD training data*. We adapt and improve an existing entity-aware machine translation system that localizes entities (Moradshahi et al., 2023), extend it to agent response generation, and equip it with a filtering step that increases the quality of the resulting translations.

2 Related Work

2.1 Multilingual Dialogue Datasets

MultiWOZ (Budzianowski et al., 2018; Ramadan et al., 2018; Eric et al., 2019) and CrossWOZ (Zhu et al., 2020) are two monolingual Wizard-Of-Oz dialogue datasets that cover several domains, suitable for building travel dialogue agents in English and Chinese respectively. For the 9th Dialog System Technology Challenge (DSTC-9) (Gunasekara et al., 2020), they were translated to Chinese and English using Google Translate.

GlobalWOZ (Ding et al., 2021), AllWOZ (Zuo et al., 2021), and Multi2WOZ (Hung et al., 2022) translate MultiWOZ to even more languages such as Spanish, Hindi, and Indonesian, with human translators post-editing machine translated dialogue templates, and filling them with newly collected local entities. Although manual post-editing improves data quality and ensures fluency, it also increases the cost and time to create new datasets, thus limiting scalability.

Different from these translation approaches, Lin et al. (2021) introduced BiToD, the first bilingual dataset for *end-to-end* ToD modeling. BiToD uses a dialogue simulator to generate dialogues in 5 tourism domains in English and Chinese, then uses crowdsourcing to paraphrase entire dialogues to be more natural. Unlike WOZ-style datasets which usually suffer from poor annotation quality due to human errors (Moradshahi et al., 2023), BiToD is automatically annotated during synthesis. Since neither manual nor machine translation is used in the creation of BiToD, it does not contain translationese (Eetemadi and Toutanova, 2014) or other artifacts of translated text (Clark et al., 2020), and provides a realistic testbed for cross-lingual transfer of task-oriented dialogue agents.

2.2 Multilingual Dialogue State Tracking

Mrkšić et al. (2017) proposed using cross-lingual word embeddings for zero-shot cross-lingual transfer of DST models. With the advent of large language models, contextual embeddings obtained from pre-trained multilingual language models (Devlin et al., 2018; Xue et al., 2021; Liu et al., 2020) have been used to enable cross-lingual transfer in many natural language tasks, including DST.

Chen et al. (2018) used knowledge distillation (Hinton et al., 2015) to transfer DST capabilities from a teacher DST model in the source language to a student model in the target language.

Machine translation has been used for DST, both as a way of obtaining cross-lingual representations, and to translate training data. For instance, Schuster et al. (2019) used representations obtained from machine translation models and reported that it performs better than training with machine translated training data for single-turn commands. More advanced data translation approaches like the entity-aware method of Moradshahi et al. (2023) further

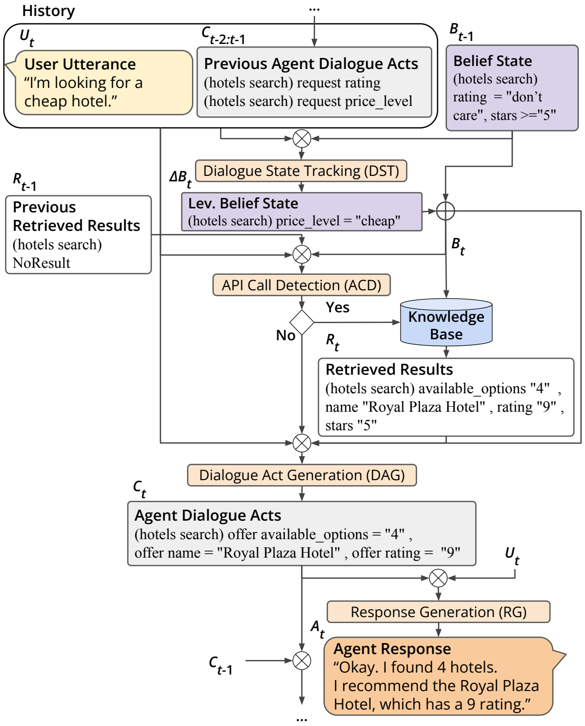


Figure 1: Inference-time flow diagram for our dialogue agent. DST, ACD, DAG, and RG share the same neural model. U, A, C, B, and R indicate user utterance, agent response, agent dialogue acts, dialogue state, and retrieved database results respectively. t is the turn number. $\otimes$ indicates text concatenation. $\oplus$ refers to the update rule in Equation 1.

improved the DST data quality achievable with machine translation.

3 Distilled ToD Agent

Our methodology includes a dialogue task decomposition and a Distilled dialogue representation that are tailored to cross-lingual ToD agents. In this section we describe these two components.

We follow the end-to-end task-oriented dialogue (ToD) setting (Hosseini-Asl et al., 2020) where a user converses freely with an agent over several turns to accomplish his/her goal with all of its constraints (e.g. "book a restaurant that is rated at least 3."). In each turn, the agent must access its database if needed to find the requested information (e.g. find a restaurant that satisfies user constraints), decide on an action (e.g. to present the information to the user or to ask follow-up questions) and finally respond to the user in natural language based on the action it selects.

3.1 Preliminaries

Formally, a *dialogue* $D = \{U_1, A_1, ..., U_T, A_T\}$ is a set of alternating user utterances U_t and agent responses A_t for a number of turns T.

A *belief state* at turn t, B_t, consists of a list of $\langle domain, intent \rangle$ tuples and a set of $\langle slot, relation, value \rangle$ tuples. *Intent* is the user intent, either search or book. *Relation* is a comparison or membership operator. *Value* can be one or more entity names or strings from the ontology, or a literal. To see all possible domains, slots and values please refer to Table 4 in Lin et al. (2021).

The *Levenshtein belief state* (Lin et al., 2020) is the difference between belief states in consecutive turns, i.e. $\Delta B_t = B_t - B_{t-1}$. It captures only the relations and values that have changed in the last user utterance, or tuples that have been added or removed.

An *Agent dialogue act* at turn t, C_t, is a list of $\langle domain, intent \rangle$ tuples and a set of $\langle dialogue\_act\_name, slot, value \rangle$ tuples indicating the action the agent takes and the information offered to the user, if any.

3.2 Task Decomposition

The task of dialogue agents is usually broken down to subtasks, which may be performed by a pipelined system (Gao et al., 2018) or by a single neural network (Hosseini-Asl et al., 2020; Lei et al., 2018). Here we describe our subtasks and their inputs and outputs (Figure 1).

After the user speaks at turn t, the agent has access to the belief state up to the previous turn (B_{t-1}), the history of agent dialogue acts ($C_1, ..., C_{t-1}$), and the history of agent and user utterances so far ($A_1, ..., A_{t-1}$ and $U_1, ..., U_t$). Our agent performs the following four subtasks:

1. *Dialogue State Tracking (DST)*: Generate ΔB_t, the Levenshtein belief state, for the current turn based on the previous belief state, the last two agent dialogue acts[2], and the current user utterance. ΔB_t is combined with B_{t-1} to produce the current belief state.

$$\begin{aligned} \Delta B_t &= \mathrm{DST}(B_{t-1}, C_{t-2}, C_{t-1}, U_t) \\ B_t &\leftarrow B_{t-1} + \Delta B_t \end{aligned} \tag{1}$$

2. *API Call Detection (ACD)*: Call an API to query the database, if needed.

$$q_t = \mathrm{ACD}(B_t, C_{t-2}, C_{t-1}, U_t, R_{t-1}) \tag{2}$$

$$R_t \;\leftarrow\; q_t ?\, \mathrm{KB}(B_t) \;:\; \emptyset \tag{3}$$

---

[2] Our ablation study described in Section 6.1 justifies the use of the last two agent dialogue acts instead of just the last one.

In turn t, ACD determines if an API call is necessary. If so, the result R_t is the top entity in the knowledge base KB, based on a deterministic ranking scheme, that matches the API call constraints in B_t, and is empty otherwise. If no entities match the constraint, we set R_t to the special value NORESULT.

3. *Dialogue Act Generation (DAG)*: Generate C_t, the agent dialogue act for the current turn based on the current belief state, the last two agent dialogue acts, the user utterance, and the result from the API call.

$$C_t = \mathrm{DAG}(B_t, C_{t-2}, C_{t-1}, U_t, R_t) \quad (4)$$

4. *Response Generation (RG)*: Convert the agent dialogue act C_t to the new agent utterance A_t. Note that C_t contains all the necessary information for this subtask. However, providing U_t improves response fluency and choice of words, leading to a higher BLEU score, partly due to mirroring (Kale and Rastogi, 2020).

$$A_t = \mathrm{RG}(U_t, C_t) \quad (5)$$

3.3 The Distilled Dialogue Representation

The design of Distilled is based on the following principles:

1. For cross-lingual agents, it is important to reduce the impact of translation errors. The representation should make minimal use of natural language by using a formal representation where possible.
2. Dialogues can get long, but the representation should be succinct, containing only the necessary information, so the neural network need not *learn* to ignore unnecessary information from copious data. This improves data efficiency as well as the training and inference speed of neural models.

We note that BiToD's original representation (Lin et al., 2021) follows neither of these principles.[3] It makes extended use of natural language: all previous user and agent natural language utterances are included in the input of all subtasks. It has many redundancies: for each subtask, it inputs the concatenation of all previous subtask's inputs and outputs. In the following, we highlight the changes we made to the (Lin et al., 2021) representation.

Replace agent utterances with formal agent dialogue acts. Since agent responses are automatically generated, it is possible to capture all information useful to the different subtasks with formal agent dialogue acts. In this way, the neural network need not interpret previous natural language utterances.

We take two steps to generate the agent responses: DAG (Dialogue Act Generation) first produces the formal act, C_t, which is then fed into RG (Response Generation) to generate the natural language response A_t. Note that RG is not a part of the dialogue loop: the natural language A_t only serves to communicate to the user; it is the formal C_t from DAG that gets fed to subsequent subtasks instead. In contrast, Lin et al. (2021) generates the agent response directly from API results. Hosseini-Asl et al. (2020) also separates the response generation into two steps, but they use A_t instead of C_t as input to the semantic parser for the next turn.

Note that the agent dialogue acts are independent of the natural language used in the dialogues, if we ignore the entity values. This is beneficial to cross-lingual agents as it can learn easier from data available in other languages. Furthermore, DAG can be validated on whether the output dialogue acts match the gold answers exactly. This is not possible with natural language results, whose quality is typically estimated with BLEU score.

Shorten user utterance history. Since the belief state formally summarizes what the user has said, we remove previous user utterances $U_1, ..., U_{t-1}$ from input to all subtasks, relying on the belief state B_{t-1} instead.

Untangle API call detection from response generation. After DST is done, depending on whether or not an API call is needed, Lin et al. (2021) either directly generates the agent response, or makes the API call and then generates the response in two steps. Our design is to always take two steps: (1) generate the API call *or indicate that there is none*, and (2) generate the agent response.

4 Automatic Dialogue Data Translation

Given a training dataset for one language, we automatically generate a training set in the target language we are interested in. This problem has been studied in the context of NLU for questions (Moradshahi et al., 2020; Sherborne et al., 2020; Li et al., 2021a) and for dialogues (Moradshahi et al., 2023;

---

[3] We found this to be true for several previously-proposed popular representations of MultiWOZ as well (Lei et al., 2018; Chen et al., 2019).

Ding et al., 2021; Zuo et al., 2021). One challenge is that the translated dataset should refer to entities in the target language. Thus, Moradshahi et al. (2020) proposed to first use cross-attention weights of the neural translation model to align entities in the original and translated sentences, then replace entities in the translated sentences with local entities from a target language knowledge base. Our initial experiments showed that applying this approach directly to end-to-end dialogue datasets does not yield good performance, especially for response generation. Thus, we adapted and improved this approach for dialogues as discussed below.

4.1 Alignment for Dialogues

First, we found that while translation with alignment works for NLU, it does not work well for RG. Machine translation introduces two kinds of error: (1) Translated sentences can be ungrammatical, incorrect, or introduce spurious information. (2) The alignment for entities may be erroneous, which can seriously hurt the factual correctness of the responses. As shown in Moradshahi et al. (2023), these errors are tolerable in NLU since (1) sentences are seen by machines, not shown to users, (2) pre-trained models like mBART are somewhat robust to noisy inputs, since they are pre-trained on perturbed data. However, training with such low-quality data is not acceptable for RG, since the learned responses are shown directly to the user.

Second, we found alignment recall to be particularly low for an important category: entities that are mostly quantitative. We observe that dates, times, and prices can be easily mapped between different languages using rules. We propose to first try to translate such entities with dictionaries such as those available in dateparser (Scrapinghub, 2015) and num2words (faire Linux, 2017), and to match them in the translated text. We resort to using neural alignment only if no such match is found.

4.2 Filtering Translation Noise for RG

To reduce translation noise for RG, we automatically filter the translated data based on the semantic textual similarity between the source and translated sentences. For this purpose, we use LaBSE (Feng et al., 2020), a multilingual neural sentence encoder based on multilingual BERT (Devlin et al., 2018), trained on translation pairs in various languages with a loss function that encourages encoding pairs to similar vectors. To score a pair of sentences, the model first calculates an embedding for each sentence and computes the cosine distance between those vectors. The lower the distance is, the more semantically similar the sentences are, according to the model.

In creating the RG training set, we first translate the source agent utterances to the target language and use LaBSE to remove pairs whose similarity score is below a threshold. We found a threshold of 0.8 to work best empirically. Higher thresholds would inadvertently filter correctly translated utterances. We construct the final training data by pairing aligned translated utterances that pass the filter with their corresponding translated agent dialogue acts.

5 Experiment Setting

5.1 Base Dataset

We perform our experiments on BiToD, a large-scale high-quality bilingual dataset created using the Machine-to-Machine (M2M) approach. It is a multi-domain dataset, including restaurants, hotels, attractions, metro, and weather domains. It has a total of 7,232 dialogues (3,689 dialogues in English and 3,543 dialogues in Chinese) with 144,798 utterances in total. The data is split into 5,787 dialogues for training, 542 for validation, and 902 for testing. The training data is from the same distribution as validation and test data.

5.2 Implementation details

Our code is implemented in PyTorch (Paszke et al., 2019) using GenieNLP (Campagna et al., 2019) library for training and evaluation metrics. We also use the Dialogues[4] library for data preprocessing and evaluation. We use pre-trained models available through HuggingFace's Transformers library (Wolf et al., 2019). The following model names are from that library. We use *mbart-large-50* as the neural model for our agent in all our experiments. All models use a standard Seq2Seq architecture with a bidirectial encoder and left-to-right autoregressive decoder. mBART is pre-trained to denoise text in 50 languages, while mT5 is trained on 101 languages. mBART uses sentence-piece (Kudo and Richardson, 2018) for tokenization.

In each setting, all four subtasks of DST, API detection, dialogue act generation, and response generation are done in a single model, where we specify the task by prepending a special token to the

---

[4] `https://github.com/stanford-oval/dialogues`

input. We found mBART to be especially effective in zero-shot settings as the language of its outputs can be controlled by providing a language-specific token at the beginning of decoding. Additionally, its denoising pre-training objective improves its robustness to the remaining translation noise.

For translation, we use the publicly available *mbart-large-50-many-to-one-mmt* (~611M parameters) model which can directly translate text from any of the 50 supported languages to English. It is an mBART model additionally fine-tuned to do translation. We use greedy decoding and train our models using teacher-forcing and token-level cross-entropy loss. We used Adam (Kingma and Ba, 2014) as our optimizer with a starting learning rate of 2×10^{-5} and linear scheduling. These hyperparameters were chosen based on a limited hyperparameter search on the validation set. For the numbers reported in the paper, due to cost, we performed only a single run for each experiment.

Our models were trained on virtual machines with a single NVIDIA V100 (16GB memory) GPU on the AWS platform. For a fair comparison, all monolingual models were trained for the same number of iterations of 60K, and bilingual models for 120K. In the few-shot setting, we fine-tuned the model for 3K steps on 1% of the data and 6K steps on 10% of the data. Sentences are batched based on their input and approximate output token count for better GPU utilization. We set the total number of tokens per batch to 800 for mBART. Due to the verbosity and redundancy of the original BiToD representation, Lin et al. (2021) used a batch size of 1 example for training mbart-large. Using our Distilled representation, however, we can fit up to 6 examples in each batch and process each batch 3 times faster during training. Training and evaluating each model takes about 10 GPU-hours on average.

During error analysis, we noticed that although certain slots (max_temp and min_temp slots in Metro domain, and time and price_range slots in Weather domain) are present in the retrieved knowledge base values, the model does not learn to output them in the agent dialogue act generation subtask. This issue stems from BiToD's non-deterministic policy where an agent sometimes provides these slots and sometimes not in the gold training data. To mitigate this, during evaluation, we automatically check if these slots are present in the input and append them and their retrieved values to the generated agent dialogue acts.

At inference time, we use the predicted belief state as input to subsequent turns instead of ground truth. However, to avoid the conversation from diverging from its original direction, Lin et al. (2021) use the ground-truth natural-language agent response as input for the next turn. To make sure the settings are equivalent for a fair comparison, we use ground-truth agent acts as input for the next turn.

5.3 Evaluation Metrics

We use the following metrics to compare different models. Scores are averaged over all turns unless specified otherwise.

- **Joint Goal Accuracy (JGA)** (Budzianowski et al., 2018): Is the standard metric for evaluating DST. JGA for a dialogue turn is 1 if all slot-relation-value triplets in the generated belief state match the gold annotation, and is 0 otherwise.

- **Task Success Rate (TSR)** (Lin et al., 2021): A task, defined as a pair of domain and intent, is completed successfully if the agent correctly provides all the user-requested information and satisfies the user's initial goal for that task. TSR is reported as an average over all tasks.

- **Dialogue Success Rate (DSR)** (Lin et al., 2021): DSR is 1 for a dialogue if all user requests are completed successfully, and 0 otherwise. DSR is reported as an average over all dialogues. We use this as the main metric to compare models, since the agent needs to complete all dialogue subtasks correctly to obtain a full score on DSR.

- **API** (Lin et al., 2021): For a dialogue turn, is 1 if the model correctly predicts to make an API call, and all the constraints provided for the call match the gold. It is 0 otherwise.

- **BLEU** (Papineni et al., 2002): Measures the natural language response fluency based on n-gram matching with the human-written gold response. BLUE is calculated at the corpus level.

- **Slot Error Rate (SER)** (Wen et al., 2015): It complements BLEU as it measures the factual correctness of natural language responses. For each turn, it is 1 if the response contains all entities present in the gold response, and is 0 otherwise.

Representation	JGA ↑	TSR ↑	DSR ↑	API ↑	BLEU ↑	SER ↓
Original (Lin et al., 2021)	69.19	69.13	47.51	67.92	38.48	14.93
Distilled (Ours)	**76.79**	**75.64**	**53.39**	**76.33**	**42.54**	**10.61**
• Generate full state	74.30	74.19	50.90	73.93	41.90	11.38
• Natural agent response	75.62	73.41	49.10	73.93	40.94	11.90
• Only last agent turn	73.97	74.19	52.71	74.27	41.83	11.81
• Prev. user utterance as state	71.75	61.66	33.94	67.67	39.72	15.97
• Remove state	70.84	51.89	24.43	66.47	37.10	19.61

Table 1: Full-shot English monolingual training with ablation. All results are reported on the English test set of BiToD using the same evaluation script. The best result is in bold.

6 Results and Discussion

We first show how our Distilled representation affects the performance of an agent in a full-shot setting. We then evaluate our proposed techniques on cross-lingual settings with varying amounts of available training data.

6.1 Evaluation of Distilled Representation

To understand how our design of Distilled representation affects the performance of ToD agents in general, we train an English agent using all the English training data and perform an ablation study (Table 1). We observe that even though the Distilled representation removes a lot of natural language inputs, it improves the best previous English-only results on JGA, TSR, DSR, API, BLEU and SER by 7.6%, 6.5%, 5.9%, 8.4%, 4.1%, and 4.7%, respectively. This suggests that natural language utterances carry a lot of redundant information, and the verbosity may even hurt the performance. Note that the improvement in BLEU is also accompanied by an improvement of factuality measured by SER.

Furthermore, using the Distilled representation reduces training time by a factor of 3. See Section 5.2 for more details.

Generate full state. Our first ablation study confirms that the proposal by Lin et al. (2020) to predict the Levenshtein belief state (ΔB_t) is indeed better than the cumulative state (B_t). Note that the training time per gradient step is more than twice as long in this ablation since the outputs are longer.

Natural agent response. Here we use natural language agent responses as input instead of agent dialogue acts, replacing C_{t-1}, C_{t-2} with A_{t-1}, A_{t-2}. The drop in TSR and DSR shows this is an important design choice - distilling natural language into a concise formal representation improves the model's ability to understand the important information in the sentence.

Only last agent turn. When we remove C_{t-2} from the input and only use C_{t-1}, we observe a drop across all metrics. This is because some turns in BiToD refer to the agent's states from two turns ago. We experimented with carrying three turns, but there was no improvement.

Previous user utterance as state. In this ablation, we use U_{t-1} instead of B_{t-1} as subtask inputs. Compared to all previous ablations, accuracy drastically decreases across all metrics, especially JGA. This is expected since the information from earlier turns present in the dialogue state is now lost. Additionally, it shows that the dataset is highly contextual and therefore a summary of the conversation history is necessary.

Remove state. We remove B_{t-1} without adding back the previous user utterance U_{t-1}. Compared to the previous ablation, TSR and DSR drop by 10.5% and 5.2% respectively. This difference shows U_{t-1} does contain part of the information captured in B_{t-1}.

6.2 Evaluation of Cross-Lingual Transfer

The goal of this experiment is to create an agent in a *target* language, given the full training data in a source language ($\mathcal{D}_{\text{src}}$), and a varying amount of training data in a target language ($\mathcal{D}_{\text{tgt}}$). We also assume that valuation and test data are available in both source and target languages. We chose Chinese as the source language and English as the target language so we can perform error analysis and the model outputs are understandable for a wider audience.

6.2.1 Varying Target Training Data

Full-Shot. In the full-shot experiments, all of $\mathcal{D}_{\text{tgt}}$ is available for training. We train two models on two data sets: (1) on a shuffled mix of $\mathcal{D}_{\text{src}}$ and D_{tgt}. (2) on $\mathcal{D}_{\text{tgt}}$ alone. The ablation "*−Mixed*" in Table 2 refers to the latter.

Zero-Shot. In our zero-shot experiments, we train with a canonicalized $\mathcal{D}\text{SRC}$ and an automatically translated data set, as explained below.

Setting	JGA ↑	TSR ↑	DSR ↑	API ↑	BLEU ↑	SER ↓
Full-Shot						
MinTL(mT5)	72.16	71.18	51.13	71.87	40.71	13.75
– Mixed	69.19	69.13	47.51	67.92	38.48	14.93
MinTL(mBART)	69.37	42.45	17.87	65.35	28.76	–
– Mixed	67.36	56.00	33.71	57.03	35.34	–
Ours	**77.52**	75.04	**54.07**	74.44	41.46	11.17
– Mixed	76.79	**75.64**	53.39	**76.33**	**42.54**	**10.61**
Zero-Shot						
Ours	**55.33**	**46.74**	**21.95**	**63.04**	**20.01**	**20.52**
– Filtering	54.83	45.03	19.68	60.81	19.11	20.86
– Alignment	47.21	4.72	1.13	52.74	8.26	39.20
– Translation	14.73	3.52	1.58	6.26	0.69	41.30
– Canonicalization	2.13	1.20	0.00	0.26	0.25	42.39
Few-Shot (1%)						
Ours	**64.60**	**57.89**	**34.16**	**62.09**	**28.15**	**17.94**
– Filtering	63.88	57.80	32.35	59.95	28.00	18.57
– Alignment	58.86	51.89	23.76	57.12	26.84	21.56
– Translation	49.58	41.34	19.68	46.05	22.73	24.86
– Canonicalization	44.56	42.97	20.36	46.23	23.08	24.77
Few-shot Only	25.08	24.61	11.09	23.67	18.71	32.62
Few-Shot (10%)						
MinTL(mT5)	58.85	56.43	34.16	57.54	31.20	–
– Translation	48.77	44.94	24.66	47.60	29.53	19.75
Few-shot Only	19.86	6.78	1.36	17.75	10.35	–
MinTL(mBART)	37.50	21.61	10.18	27.44	17.86	–
– Translation	42.84	36.19	16.06	41.51	22.50	–
Few-shot Only	4.64	1.11	0.23	0.60	3.17	–
Ours	**72.70**	**71.61**	**48.19**	**72.56**	**36.02**	**12.71**
– Filtering	72.45	69.55	44.57	69.55	34.67	13.62
– Alignment	68.40	63.38	38.24	63.38	32.99	16.63
– Translation	67.13	63.12	41.40	63.64	32.86	16.40
– Canonicalization	64.51	63.64	40.27	62.69	32.71	16.63
Few-shot Only	57.18	54.80	28.73	55.66	29.61	19.66

Table 2: All results are reported on the original English test set of BiToD using the same evaluation script. The best result in each section is in bold. Each "−" removes one additional component from the previous row. All MinTL results are from Lin et al. (2021). SER numbers are not available for some models. An upward arrow is show for columns where bigger numbers are better, and vice versa.

Canonicalization: To increase transfer learning from the source to the target language, we use the same canonical formal representation across languages (Moradshahi et al., 2020; Razumovskaia et al., 2021). To do so, we adapt $\mathcal{D}$src so that the domain names, slot names, agent dialogue acts, and API names in the formal representation to be the same as the target language. Note that the user utterance, agent response, and slot values will remain in the source language. The BiToD dataset has a one-to-one mapping for most of the above and we added the missing items.

Translation: We use machine translation to convert the user and agent utterances and slot values in $\mathcal{D}_{\text{src}}$ to create a training set for the target language.

Alignment: After translating the data, we use alignment (Section 4) to localize entities while ensuring the entities in translated utterances still match the values specified in annotations.

Filtering: We use the filtering procedure described in Section 4.2 to remove turns where agent responses are deemed to have low translation quality.

In Table 2, *Ours* refers to our main approach, which combines all four techniques. Each ablation incrementally takes away one of the techniques.

Few-Shot. In the few-shot setting, we start with our pre-trained zero-shot models (with various ablations) and further fine-tune it on 1% and 10% of $\mathcal{D}_{\text{tgt}}$, which comprises 29 and 284 dialogues, respectively. Lin et al. (2021) reported the results only for the 10% setting. We use their few-shot data split in that case to be directly comparable. We add one more ablation study where we eliminate cross-lingual transfer by training a model only on the few-shot data (Few-shot Only).

6.2.2 Baseline

We compare our results to the best previously reported result on BiToD from Lin et al. (2021). This SOTA result was obtained using MinTL (Lin et al., 2020) and using a single mT5-small model to perform all dialogue subtasks.

Contrary to what Lin et al. (2021) reported, we found that mBART-large model outperforms mT5-small in all settings. We have included all the results including MinTL(mBART) in Table 2 for comparison.

6.2.3 Results

The results for our cross-lingual experiment are reported in Table 2. Overall, in the full-shot setting, when training on both source and target language data, we improve the SOTA in JGA by 5.3%, TSR by 3.8%, DSR by 2.9%, API by 2.6%, BLEU by 0.8%, and SER by 2.6%.

Our zero-shot agent achieves 71%, 62%, 40%, and 47% of the performance of a full-shot agent in terms of JGA, TSR, DSR, and BLEU score, respectively. In the 10% few-shot setting, our approach establishes a new SOTA by increasing JGA, TSR, DSR, API, and BLEU absolutely by 13.9%, 15.2%, 14.0%, 15.0%, and 4.8% respectively. Prominently, training with just 10% of the data beats the full-shot baseline which is trained on 100% of the training data, on all metrics except for DSR and BLEU. It also comes within 5% of full training using the Distilled representation on all metrics.

Our Distilled representation improves the performance, especially in few-shot. Comparing our results with that of Lin et al. (2021), in the full-shot monolingual setting (MinTL(mT5) "−Mixed" vs. Ours "−Mixed"), models trained on data with our representation outperform the baseline on all metrics. In the pure few-shot (10%) setting, Ours outperforms MinTL(mT5) significantly in all metrics. This suggests that our Distilled representation and task decomposition are much more effective in low-data settings.

Canonicalization is useful. Comparing "−Translation" with "−Canonicalization", training on canonicalized data significantly improves the results in the zero-shot setting. This is intuitive since canonicalization brings training data closer in vocabulary to the test data in the target language. This improvement comes at almost no cost since translation is done automatically using a dictionary.

Automatic naive translation of the training set does not work for zero-shot. The naive translation approach (i.e. without alignment) completely fails in the zero-shot setting by achieving only 4.7% in TSR, and 1.1% in DSR, as translated entities might no longer match with ones in the annotation. Adding few-shot data helps significantly as the gap closes between "−Alignment" and "−Translation" ablations.

Alignment improves translation quality in all settings and metrics. With alignment, the translation approach performs much better in all settings, establishing a new state-of-the-art in zero and few-shot settings according to almost all metrics. As a general trend, the lower data settings benefit more from alignment. We additionally performed an experiment using the alignment proposed by (Moradshahi et al., 2023). There is a 4.0% drop in TSR and 4.5% in DSR, confirming the benefit of our improved alignment.

Filtering noise for RG improves fluency. We perform an ablation by training separate models on filtered and unfiltered translated agent utterances. The filtering process is described in Section 4.2. In 10% fewshot setting, both BLEU and SER improve by 1.4% confirming that automatically removing poor translations from training data improves the agent response quality. Interestingly, we observe an increase in other metrics too. Since model parameters are shared between all subtasks, enhancing the data quality for one subtask will have a positive impact on the others as well.

7 Conclusion

This paper shows how to build a dialogue agent in a new language automatically, given a dialogue dataset in another language, by using entity-aware machine translation and our new Distilled dialogue representation. The performance can be further improved if a few training examples in the target language are available, and we show that our approach outperforms existing ones in this setting as well.

On the BiToD dataset, our method achieves 3.9% and 2.9% improvement in TSR and DSR, respectively, over the previous SOTA in full-shot setting, and 15.2% and 14.0% in a 10% few-shot setting, showing the effectiveness of our approach. More importantly, training on translated data and only 10% of original training data comes within 5% of full training.

We have implemented our methodology as a toolkit for developing multilingual dialogue agents, which we have released open-source. Our proposed methodology can significantly reduce the cost and time associated with data acquisition for task-oriented dialogue agents in new languages.

8 Limitations

As discussed in Section 2.1, organic (i.e. without the use of translation) multilingual dialogue datasets are scarce, which has limited the scope of our experiments. Our guidelines to improve dialogue representation mentioned in Section 4 are general and applicable to any Human-to-Human or Machine-to-Machine dialogues annotated with slot-values. We have yet to evaluate the generalization of our cross-lingual approach across different languages and datasets, and to Human-to-Human dialogues. For instance, we use a Chinese to English translator in this work. Available translation models for low-resource languages have much lower quality, and this will likely lower the performance of this approach.

Another limitation is the lack of human evaluation for agent responses. BLEU score does not correlate well with human judgment, and SER only accounts for the factuality of the response but not the grammaticality or fluency. This problem is also reported in prior works (see Section 5). Although finding native speaker evaluators for different languages is a challenge (Pavlick et al., 2014), in the future, we wish to address this by conducting human evaluations.

9 Ethical Considerations

We do not foresee any harmful or malicious misuses of the technology developed in this work. The data used to train models is about seeking information about domains like restaurants, hotels and tourist attractions, does not contain any offensive content, and is not unfair or biased against any demographic. This work does focus on two widely-spoken languages, English and Chinese, but we think the cross-lingual approach we proposed can improve future dialogue language technologies for a wider range of languages.

We fine-tune multiple medium-sized (several hundred million parameters) neural networks for our experiments. We took several measures to avoid wasted computation, like performing one run instead of averaging multiple runs (since the numerical difference between different models is large enough), and improving batching and representation that improved training speed, and reduced needed GPU time. Please refer to Appendix 5.2 for more details about the amount of computation used in this paper.

Acknowledgements

This work is supported in part by the National Science Foundation under Grant No. 1900638, the Alfred P. Sloan Foundation under Grant No. G-2020-13938, Microsoft, Stanford HAI and the Verdant Foundation.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Giovanni Campagna, Silei Xu, Mehrad Moradshahi, Richard Socher, and Monica S. Lam. 2019. Genie: A generator of natural language semantic parsers for virtual assistant commands. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, pages 394–410, New York, NY, USA. ACM.

Wenhu Chen, Jianshu Chen, Pengda Qin, Xifeng Yan, and William Yang Wang. 2019. Semantically conditioned dialog response generation via hierarchical disentangled self-attention. *arXiv preprint arXiv:1905.12866*.

Wenhu Chen, Jianshu Chen, Yu Su, Xin Wang, Dong Yu, Xifeng Yan, and William Yang Wang. 2018. XL-NBT: A cross-lingual neural belief tracking framework. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 414–424, Brussels, Belgium. Association for Computational Linguistics.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Bosheng Ding, Junjie Hu, Lidong Bing, Sharifah Mahani Aljunied, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. Globalwoz: Globalizing multiwoz to develop multilingual task-oriented dialogue systems.

Sauleh Eetemadi and Kristina Toutanova. 2014. Asymmetric features of human generated translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 159–164, Doha, Qatar. Association for Computational Linguistics.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Savoir faire Linux. 2017. num2words. `https://github.com/savoirfairelinux/num2words`.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371–1374.

Chulaka Gunasekara, Seokhwan Kim, Luis Fernando D'Haro, Abhinav Rastogi, Yun-Nung Chen, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, Dilek Hakkani-Tür, Jinchao Li, Qi Zhu, Lingxiao Luo, Lars Liden, Kaili Huang, Shahin Shayandeh, Runze Liang, Baolin Peng, Zheng Zhang, Swadheen Shukla, Minlie Huang, Jianfeng Gao, Shikib Mehri, Yulan Feng, Carla Gordon, Seyed Hossein Alavi, David Traum, Maxine Eskenazi, Ahmad Beirami, Eunjoon, Cho, Paul A. Crook, Ankita De, Alborz Geramifard, Satwik Kottur, Seungwhan Moon, Shivani Poddar, and Rajen Subba. 2020. Overview of the ninth dialog system technology challenge: Dstc9.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Chia-Chien Hung, Anne Lauscher, Ivan Vulić, Simone Paolo Ponzetto, and Goran Glavaš. 2022. Multi2woz: A robust multilingual dataset and conversational pretraining for task-oriented dialog. *arXiv preprint arXiv:2205.10400*.

Mihir Kale and Abhinav Rastogi. 2020. Template guided text generation for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6505–6520, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021a. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Jinchao Li, Qi Zhu, Lingxiao Luo, Lars Liden, Kaili Huang, Shahin Shayandeh, Runze Liang, Baolin Peng, Zheng Zhang, Swadheen Shukla, Ryuichi Takanobu, Minlie Huang, and Jianfeng Gao. 2021b. Multi-domain task-oriented dialog challenge ii at dstc9. In *AAAI-2021 Dialog System Technology Challenge 9 Workshop*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. MinTL: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3391–3405, Online. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, Peng Xu, Feijun Jiang, Yuxiang Hu, Chen Shi, and Pascale Fung. 2021. BiToD: A bilingual multi-domain dataset for task-oriented dialogue modeling. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1 pre-proceedings (NeurIPS Datasets and Benchmarks 2021)*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation.

Mehrad Moradshahi, Giovanni Campagna, Sina Semnani, Silei Xu, and Monica Lam. 2020. Localizing open-ontology QA semantic parsers in a day using machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5970–5983, Online. Association for Computational Linguistics.

Mehrad Moradshahi, Victoria Tsai, Giovanni Campagna, and Monica S Lam. 2023. Contextual semantic parsing for multilingual task-oriented dialogues. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017. Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*, 5:309–324.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037.

Ellie Pavlick, Matt Post, Ann Irvine, Dmitry Kachaev, and Chris Callison-Burch. 2014. The language demographics of amazon mechanical turk. *Transactions of the Association for Computational Linguistics*, 2:79–92.

Jun Quan, Shian Zhang, Qian Cao, Zizhong Li, and Deyi Xiong. 2020. RiSAWOZ: A large-scale multi-domain Wizard-of-Oz dataset with rich semantic annotations for task-oriented dialogue modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 930–940, Online. Association for Computational Linguistics.

Osman Ramadan, Paweł Budzianowski, and Milica Gasic. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 432–437.

Evgeniia Razumovskaia, Goran Glavaš, Olga Majewska, Anna Korhonen, and Ivan Vulic. 2021. Crossing the conversational chasm: A primer on multilingual task-oriented dialogue systems. *arXiv preprint arXiv:2104.08570*.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.

Scrapinghub. 2015. dateparser. `https://github.com/scrapinghub/dateparser`.

Tom Sherborne, Yumo Xu, and Mirella Lapata. 2020. Bootstrapping a crosslingual semantic parser. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 499–517, Online. Association for Computational Linguistics.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Qi Zhu, Kaili Huang, Zheng Zhang, Xiaoyan Zhu, and Minlie Huang. 2020. CrossWOZ: A large-scale Chinese cross-domain task-oriented dialogue dataset. *Transactions of the Association for Computational Linguistics*, 8:281–295.

Lei Zuo, Kun Qian, Bowen Yang, and Zhou Yu. 2021. Allwoz: Towards multilingual task-oriented dialog systems for all. *arXiv preprint arXiv:2112.08333*.

A Appendix

A.1 Dialogue Examples

We include the same example from BiToD's English validation set both in our Distilled representation (Table 3) and in the original (Table 4) representation, along with model predictions in the full-shot setting. For brevity, only the first 3 turns are shown.

In Table 4, we observe that the model fails to ask for the hotel price-range in the second turn and makes an API call instead. Since the API call results are carried over between turns in this representation, in the third turn, the model sees those results in the input and falsely assumes it does not need to make an API call anymore, ultimately resulting in an incorrect response. Compare this to our representation in Table 3. This example shows the importance of separation between API call detection and response generation.

Another phenomenon we often observe is that the model asks for more information than it should according to the gold agent dialogue act. As shown in Table 3, in the second turn, the agent requests user to provide the desired location for the hotel as well as the price range. We believe the main reason for this behavior is the randomness in the agent policy of the BiToD's dialogue simulator. For example, if the agent needs to fill out two slots to make an API call, it can do so by requesting both in the same turn, or one turn at a time. This behavior, though reasonable, is penalized during evaluation, and predictions are considered incorrect if they contain extraneous slots.

Turn 1	DST	Input	DST: <state> null <endofstate> <history> USER: I'd like hotel recommendations. <endofhistory>
		Target	(hotels search)
		Prediction	(hotels search)
	ACD	Input	API: <knowledge> null <endofknowledge> <state> (hotels search) <endofstate> <history> USER: I'd like hotel recommendations. <endofhistory>
		Target	no
		Prediction	no
	DAG	Input	ACTS: <knowledge> null <endofknowledge> <state> (hotels search) <endofstate> <history> USER: I'd like hotel recommendations. <endofhistory>
		Target	(hotels search) request rating , request stars
		Prediction	(hotels search) request rating , request stars
	RG	Input	RG: <actions> (hotels search) request rating , request stars <endofactions> <history> USER: I'd like hotel recommendations. <endofhistory>
		Target	Certainly. Do you have any requirements for the hotel's rating or the number of stars of the hotel?
		Prediction	Do you have a preference on how many stars and what rating the hotel should have?
Turn 2	DST	Input	DST: <state> (hotels search) <endofstate> <history> AGENT_ACTS: (hotels search) request rating , request stars USER: The rating doesn't matter, but should be at least 5 stars. <endofhistory>
		Target	(hotels search) rating equal_to " don't care " , stars at_least " 5 "
		Prediction	(hotels search) rating equal_to " don't care " , stars at_least " 5 "
	ACD	Input	API: <knowledge> null <endofknowledge> <state> (hotels search) rating equal_to " don't care " , stars at_least " 5 " <endofstate> <history> AGENT_ACTS: (hotels search) request rating , request stars USER: The rating doesn't matter, but should be at least 5 stars. <endofhistory>
		Target	no
		Prediction	no
	DAG	Input	ACTS: <knowledge> null <endofknowledge> <state> (hotels search) rating equal_to " don't care " , stars at_least " 5 " <endofstate> <history> AGENT_ACTS: (hotels search) request rating , request stars USER: The rating doesn't matter, but should be at least 5 stars. <endofhistory>
		Target	(hotels search) request price_level
		Prediction	(hotels search) request location , request price_level
	RG	Input	RG: <actions> (hotels search) request price_level <endofactions> <history> USER: The rating doesn't matter, but should be at least 5 stars. <endofhistory>
		Target	Do you have a price range for the hotel?
		Prediction	And what about location? Do you have a price range for the hotel?

Turn 3	DST	Input	DST: <state> (hotels search) rating equal_to " don't care " , stars at_least " 5 " <endofstate> <history> AGENT_ACTS_PREV: (hotels search) request rating , request stars AGENT_ACTS: (hotels search) request price_level USER: cheap <endofhistory>
		Target	(hotels search) price_level equal_to " cheap "
		Prediction	(hotels search) price_level equal_to " cheap "
	ACD	Input	API: <knowledge> null <endofknowledge> <state> (hotels search) price_level equal_to " cheap " , rating equal_to " don't care " , stars at_least " 5 " <endofstate> <history> AGENT_ACTS_PREV: (hotels search) request rating , request stars AGENT_ACTS: (hotels search) request price_level USER: cheap <endofhistory>
		Target	yes
		Prediction	yes
	DAG	Input	ACTS: <knowledge> (hotels search) available_options " 4 " , location " Mong Kok | Kowloon | Yau Tsim Mong District " , name " Royal Plaza Hotel " , price_level " cheap " , price_per_night " 793 HKD " , rating " 9 " , stars " 5 " <endofknowledge> <state> (hotels search) price_level equal_to " cheap " , rating equal_to " don't care " , stars at_least " 5 " <endofstate> <history> AGENT_ACTS_PREV: (hotels search) request rating , request stars AGENT_ACTS: (hotels search) request price_level USER: cheap <endofhistory>
		Target	(hotels search) offer available_options equal_to " 4 " , offer name equal_to " Royal Plaza Hotel " , offer rating equal_to " 9 "
		Prediction	(hotels search) offer available_options equal_to " 4 " , offer name equal_to " Royal Plaza Hotel " , offer rating equal_to " 9 "
	RG	Input	RG: <actions> (hotels search) offer available_options equal_to " 4 " , offer name equal_to " Royal Plaza Hotel " , offer rating equal_to " 9 " <endofactions> <history> USER: cheap <endofhistory>
		Target	Okay. There are 4 hotels available. I recommend the Royal Plaza Hotel, which has a 9 rating.
		Prediction	There are 4 available hotels. I recommend Royal Plaza Hotel. Its rating is 9.

Table 3: An example from BiToD English validation set in Distilled representation, along with our mBART model predictions. For brevity, only the first 3 turns are shown.

Turn 1	DST	Input	Track Dialogue State:<knowledge><dialogue_state> <user> I'd like hotel recommendations.
		Target	<API> hotels search
		Prediction	<API> hotels search
	API/ **Response**	Input	Generate Response:<knowledge><dialogue_state> <API> hotels search<user> I'd like hotel recommendations.
		Target	Certainly. Do you have any requirements for the hotel's rating or the number of stars of the hotel?
		Prediction	What rating would you like the hotel to have?
Turn 2	DST	Input	Track Dialogue State:<knowledge><dialogue_state> <API> hotels search<user> I'd like hotel recommendations.<system> Certainly. Do you have any requirements for the hotel's rating or the number of stars of the hotel?<user> The rating doesn't matter, but should be at least 5 stars.
		Target	<API> hotels search<slot> rating<relation> equal_to<value> don't care<slot> stars<relation> at_least<value> 5
		Prediction	<API> hotels search<slot> rating<relation> equal_to<value> don't care<slot> stars<relation> at_least<value> 5
	API/ **Response**	Input	Generate Response:<knowledge><dialogue_state> <API> hotels search<slot> rating<relation> equal_to<value> don't care<slot> stars<relation> at_least<value> 5<user> I'd like hotel recommendations.<system> Certainly. Do you have any requirements for the hotel's rating or the number of stars of the hotel?<user> The rating doesn't matter, but should be at least 5 stars.
		Target	Do you have a price range for the hotel?
		Prediction	I found 43 hotels. I recommend Cordis, Hong Kong, which has a rating of 10.
Turn 3	DST	Input	Track Dialogue State:<knowledge><dialogue_state> <API> hotels search<slot> rating<relation> equal_to<value> don't care<slot> stars<relation> at_least<value> 5<user> The rating doesn't matter, but should be at least 5 stars.<system> Do you have a price range for the hotel?<user> cheap
		Target	<API> hotels search<slot> price_level<relation> equal_to<value> cheap
		Prediction	<API> hotels search<slot> price_level<relation> equal_to<value> cheap
	API/ Response	Input	Generate Response:<knowledge><dialogue_state> <API> hotels search<slot> rating<relation> equal_to<value> don't care<slot> stars<relation> at_least<value> 5<slot> price_level<relation> equal_to<value> cheap<user> The rating doesn't matter, but should be at least 5 stars.<system> Do you have a price range for the hotel?<user> cheap
		Target	<API> hotels search
		Prediction	–
	API/ **Response**	Input	Generate Response:<knowledge> [hotels]<slot> name<value> Royal Plaza Hotel<slot> location<value> Mong Kok<value> Kowloon<value> Yau Tsim Mong District<slot> price_level<value> cheap<slot> price_per_night<value> 793 HKD<slot> rating<value> 9<slot> stars<value> 5<slot> available_options<value> 4<dialogue_state> <API> hotels search<slot> rating<relation> equal_to<value> don't care<slot> stars<relation> at_least<value> 5<slot> price_level<relation> equal_to<value> cheap<user> The rating doesn't matter, but should be at least 5 stars.<system> Do you have a price range for the hotel?<user> cheap<API> hotels search
		Target	Okay. There are 4 hotels available. I recommend the Royal Plaza Hotel, which has a 9 rating.
		Prediction	The hotel costs 839 HKD per night.

Table 4: Same example as in Table 3 but in the original representation from Lin et al. (2021), along with MinTL(mT5) model predictions. For brevity, only the first 3 turns are shown.

Contextual Semantic Parsing for Multilingual Task-Oriented Dialogues

Mehrad Moradshahi[1] **Victoria Tsai**[2] **Giovanni Campagna**[1] **Monica S. Lam**[1]
[1] Stanford University, Stanford, CA
[2] Meta, Menlo Park, CA
`{mehrad,gcampagn,lam}@cs.stanford.edu`
`{vitsai}@meta.com`

Abstract

Robust state tracking for task-oriented dialogue systems currently remains restricted to a few popular languages. This paper shows that given a large-scale dialogue data set in one language, we can automatically produce an effective semantic parser for other languages using machine translation. We propose automatic translation of dialogue datasets with alignment to ensure faithful translation of slot values and eliminate costly human supervision used in previous benchmarks. We also propose a new contextual semantic parsing model, which encodes the formal slots and values, and only the last agent and user utterances. We show that the succinct representation reduces the compounding effect of translation errors, without harming the accuracy in practice.

We evaluate our approach on several dialogue state tracking benchmarks. On RiSAWOZ, CrossWOZ, CrossWOZ-EN, and MultiWOZ-ZH datasets we improve the state of the art by 11%, 17%, 20%, and 0.3% in joint goal accuracy. We present a comprehensive error analysis for all three datasets showing erroneous annotations can lead to misguided judgments on the quality of the model.

Finally, we present RiSAWOZ English and German datasets, created using our translation methodology. On these datasets, accuracy is within 11% of the original showing that high-accuracy multilingual dialogue datasets are possible without relying on expensive human annotations. We release our datasets and software open source.[1]

1 Introduction

Tremendous effort has gone into the research and development of task-oriented dialogue agents for English and a few other major languages in recent years. A methodology that can transfer the effort to other languages automatically will greatly benefit the large population of speakers of the many other languages in the world.

Underlying an effective TOD agent is *dialogue state tracking*, the task of predicting a formal representation of the conversation sufficient for the dialogue agent to reply, in the form of slots and values. However, DST currently remains restricted to a few popular languages (Razumovskaia et al., 2021). Traditional DST agents require large hand-annotated *Wizard-of-Oz* (Kelley, 1984) datasets for training, which are prohibitively labor-intensive to produce in most languages (Gunasekara et al., 2020). Large, multi-domain WOZ datasets are only available in English and Chinese (Quan et al., 2020; Ye et al., 2021a).

The contributions of this paper are as follows:

1. We propose an automatic technique to build multilingual data sets using machine translation. Machine translation has been shown effective for localizing question-answering agents (Moradshahi et al., 2020). It shows that for open ontology datasets, we need to use an alignment model to properly translate entities in the source language to entities in the target language. This paper shows that alignment is necessary even for closed ontology datasets and dialogues.

Furthermore, we improve alignment to address these challenging issues we discovered unique to dialogues: (1) Translation errors accumulate and can prevent a correct parse for the rest of the dialogue; (2) There are logical dependencies between slot values across different turns; (3) Utterances are generally longer and more complex carrying multiple entities. We found that alignment improves the accuracy on the RiSAWOZ benchmark by 45.6%. This technique eliminates the cost of human post-editing used on all previous translation benchmarks, and can improve machine translation quality on other tasks too.

Using this methodology, we automatically trans-

---

[1] `https://github.com/stanford-oval/dialogues`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 902–915
May 2-6, 2023 ©2023 Association for Computational Linguistics

late the RiSAWOZ dataset to English and German, creating RiSAWOZ-EN-auto and RiSAWOZ-DE-auto datasets respectively.

2. We show that the accumulation of translation and annotation errors across turns can be mitigated with a *Contextual Semantic Parsing* (CSP) model for state tracking. We propose a BART-CSP model, a seq-to-seq based on BART, that encodes the belief state, and the last agent and user utterances, rather than the full history of utterances.

BART-CSP improves SOTA on RiSAWOZ (Quan et al., 2020) and CrossWOZ (Zhu et al., 2020), two large-scale multi-domain WoZ dialogue datasets, by 10.7% and 17% in Joint Goal Accuracy(JGA). Notably, BART-CSP is more effective on translated data as evident by bigger performance improvement: on RiSAWOZ-EN-auto and RiSAWOZ-DE-auto datasets, automatically translated versions of RiSAWOZ, BART-CSP improves SOTA by 32.4% and 52.5%.

2 Related Work

2.1 Dialogue State Tracking

Dialogue state tracking (DST) refers to the task of predicting a formal state of a dialogue at its current turn, as a set of slot-value pairs at every turn. State-of-the-art approaches apply large transformer networks (Peng et al., 2020; Hosseini-Asl et al., 2020) to encode the full dialogue history in order to predict slot values. Other approaches include question-answering models (Gao et al., 2019), ontology matching in the finite case (Lee et al., 2019), or pointer-generator networks (Wu et al., 2019). Both zero-shot cross-lingual DST transfer (Ponti et al., 2018; Chen et al., 2018) and multilingual knowledge distillation (Hinton et al., 2015; Tan et al., 2019) have been investigated; however, training with translated data is the dominant approach, outperforming zero-shot and few-shot methods.

2.2 Contextual Semantic Parsing

Alternatively to encoding the full dialogue history, previous work has proposed including the state as context (Lei et al., 2018; Heck et al., 2020; Ye et al., 2021b) together with the last agent and user utterance. Recently, Cheng et al. (2020) proposed replacing the agent utterance with a formal representation as well. Existing models rely on custom encoder architectures and loss functions for the state (Heck et al., 2020). Our formulation of CSP is different since we encode the formal dialogue state directly as text, which simplifies the architecture and makes better use of the pretrained model's understanding of natural text.

Previous work also applied rule-based state trackers that compute the state based on the agent and user dialogue acts (Schulz et al., 2017; Zhong et al., 2018; Zhu et al., 2020). Such techniques cannot handle state changes outside of a state machine defined ahead of time and do not achieve state-of-the-art accuracy on WOZ dialogues.

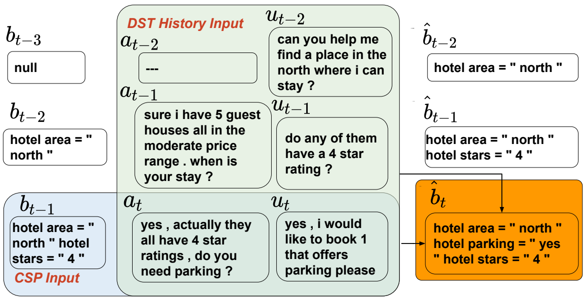
2.3 Multilingual Dialogues

Several multilingual dialogue benchmarks have been created over the past few years. Dialogue State Tracking Challenge (DSTC) has released several datasets (Kim et al., 2016; Hori et al., 2019; Gunasekara et al., 2020), covering only a few domains and languages. CrossWOZ (Zhu et al., 2020) and RiSAWOZ (Quan et al., 2020) are Chinese datasets collected through crowdsourcing. BiToD (Lin et al., 2021) uses a dialogue simulator to generate dialogues in English and Chinese, then uses crowdsourcing to paraphrase entire dialogues. All these approaches use crowdworkers in one or multiple stages of data collection which is costly and human errors degrade quality. Automatic creation of affordable high-quality dialogue datasets for other languages still remains a challenge (Razumovskaia et al., 2021).

3 Task Setting

We are interested in the *dialogue state tracking* task, in which the goal is to predict a formal representation of a conversation up to a certain point, also known as *belief state*, consisting of the slots that were mentioned in the dialogue and their value. At the beginning of the conversation, the belief state is empty, and it grows as the conversation progresses, accumulating the slots that were mentioned across all turns prior.

Formally, the problem is formulated given a predefined set of *slots* $s_1, s_2, \ldots s_n$ (such as "restaurant name", "restaurant food", etc.). Each slot has one value taken from the *ontology* $v_1, v_2, \ldots, v_n$. The ontology contains the legitimate values for the slot from the database (i.e. the list of restaurant names or restaurant cuisines), as well as the special values "none" indicating the slot was not mentioned, and "dontcare" indicating the slot was explicitly mentioned by the user but the user has



can you help me find a place in the north where i can stay ?

sure i have 5 guest houses all in the moderate price range . when is your stay ?

do any of them have a 4 star rating ?

yes , actually they all have 4 star ratings , do you need parking ?

yes , i would like to book 1 that offers parking please

Figure 1: Visualization of the differences between inputs of CSP and traditional DST models. While the latter encodes the full history of the dialogue, CSP only encodes the current state and turn. b, a, and u indicate the dialogue state, agent utterance, and user utterance, respectively. t is the turn number. $\hat{}$ indicates a predicted dialogue state.

no preference.

Therefore, given a partial conversation history composed of turns $x_1, x_2, \dots, x_t$, where each turn consists of an agent response (a_t) and user utterance (u_t), the task is to predict the belief state: $b_n(x_1, \dots, x_t) = \{s_1 = v_{1,t}, s_1 = v_{2,t}, \dots, s_n = v_{n,t}\}$ where n is the number of slots and $v_{i,t}$ is the value of the slot s_i up to turn t of the conversation. Note that the slot could be mentioned at turn t, or in any of the turns before.

4 Reducing Translation Noise

Key to the success of the dialogue state tracking is a precise and consistent annotation of the belief state at every turn. This is challenging in the multilingual setting, where we apply automatic translation to produce new training sets for DST: (1) the translation can be ungrammatical or incorrect, and it can introduce spurious information, and (2) the new annotation (in the target, translated ontology) can diverge from the translation of the sentence, such as referring to the same value in different ways in the target language.

We refer to the error introduced in translation collectively as *translation noise*. We posit that translation noise is the reason for poorer performance in existing translated dialogue datasets, compared to the same model on the dialogue dataset in the source language. In this section, we describe our methods to reduce translation noise.

4.1 Alignment

A major source of translation noise is due to mismatches between the translation of an entity by itself and in sentence. For instance, given a Chinese sentence that refers to an utterance containing the word “aquarium” (水族馆), incorrect translation may result in “ocean museum” in the English utterance, which does not match the slot value “aquarium” in the ontology and annotation anymore. Slot values may also get dropped or transliterated.

Translation with alignment was previously proposed by Moradshahi et al. (2020) to localize open-ontology multilingual semantic parsing datasets. Token alignments, obtained from cross-attention weights of the neural model, are used to track position of entities during translation so they can be replaced afterwards with local entity values. Figure 2 shows the translation and alignment process for an example input.

We show that alignment is useful also for a finite (closed) ontology in a dialogue setting. The dialogue setting is more challenging since the replacement with local entity values must be consistent across turns and dependent slots - slots that their values are logically dependent on each other. For instance, the corresponding price range for a *fast food* restaurant should be *cheap*, or a speaker looking for an attraction to go to with his girlfriend wants a place where *best-for-crowd* = “lover’s date”. Furthermore, utterances are generally longer and more complex containing multiple entities.

We have made several changes in alignment to address these issues:

1. We use a dictionary constructed from the dataset’s ontology for translating the dependent slots to ensure relations are preserved. For all other slots, we randomly replace them with values from the target language ontology similar to previous work.

2. In previous work, quotation marks were used to mark the boundary of entities and to retrieve alignment between tokens in the input and output. We found the translation of quotation marks to be inconsistent. Instead, we omit those marks before translation and purely rely on cross-attention between subwords to compute alignment.

3. We observed alignment does poorly on digits and often misplaces them in the output. We use string matching to retrieve spans for numbers, dates, and time slots if present in the output and omit alignment if successful.

4. Dialogues contain longer utterances with multiple entities per turn. We found breaking

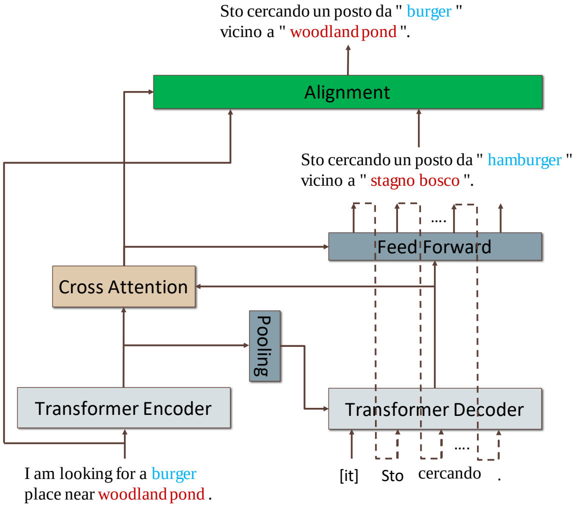


Figure 2: Example of the translation and alignment process. The encoder-decoder cross-attention weights are used to achieve word alignment.

down utterances into individual sentences before translation, significantly improves the quality of outputs when there are fewer entities to align.

To measure the effect of our changes, we translate MultiWOZ English dev set to Chinese with the new and previous alignment methods using MBART-MMT (see Section 5.3 for details). The new approach improves BLEU score by 8 points and JGA by 5% on MultiWOZ-ZH dev set.

4.2 Contextual Semantic Parsing

Previous work on dialogue state tracking encodes all or several turns up to the current one using a neural model, which then predicts the value for each slot. Hence, the input of the model consists exclusively of natural language, and grows as the conversation grows, accumulating any translation noise.

At the same time, we observe that the belief state at turn t, b_t, can be computed from the belief state at turn $t-1$ and the slots mentioned in the utterances at turn t. Hence, we propose to use a *contextual semantic parser* (CSP) for dialogue state tracking that computes $P(b_t|a_t; u_t; b_{t-1})$. The CSP model is applied to the dialogue-state tracking task by iteratively predicting the belief state of all turns, starting from b_0, the initial state consisting of all empty slots.

The CSP formulation condenses the dialogue history into a formal, fixed-length representation. Because the representation does not grow with the dialogue, it does not suffer from accumulation of translation noise.

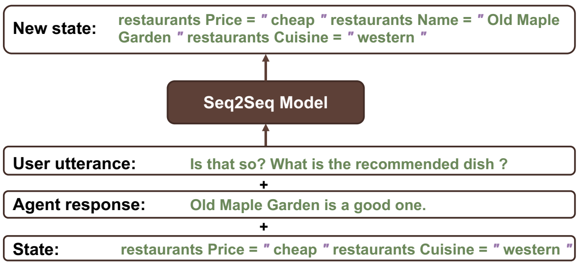


Figure 3: Contextual Semantic Parser model. We use BART for English and MBART for Chinese and German as the seq2seq model.

Our CSP model is based on Seq2Seq Transformer models BART (Lewis et al., 2019) for English and MBART (Liu et al., 2020) for all others. Here we refer to them as CSP-BART for simplicity.

The model encodes the belief state as a textual sequence of slot names and slot values. This encoding is concatenated to the agent utterance and user utterance, and fed to the model to predict the belief state at the end of the turn (Fig. 3). Similar to (Yang et al., 2021b), we encode the belief state directly as text, which simplifies the architecture and leverages the pretraining of BART.

5 Experiments

Our experiments are designed to answer the following questions: 1) How well does CSP perform on WOZ datasets compared to DST models that encode the full conversation history? 2) Is our approach effective to reduce the translation noise?

5.1 Datasets

We evaluate our models on the RiSAWOZ (Quan et al., 2020), MultiWOZ (Budzianowski et al., 2018; Eric et al., 2019), and CrossWOZ (Zhu et al., 2020) datasets and their available translated versions: MultiWOZ Chinese (Li et al., 2021), CrossWOZ English (Li et al., 2021). These particular datasets were chosen because they are large Wizard-of-OZ dialogue datasets and therefore more natural and representative of task-oriented dialogues. Additionally, we use our methodology to create the RiSAWOZ English and German datasets.

RiSAWOZ (Quan et al., 2020) is a Chinese WOZ dataset of 11k annotated dialogues and over 150k utterances spanning the 12 domains of attraction, restaurant, hotel, flight, train, weather, movie, TV, computer, car, hospital, and courses. Dialogues are formed from both single-domain and multi-domain goals, and annotated with dialogue states, dialogue acts, and coreference clusters.

MultiWOZ is an English-language WOZ dataset of 10k single- and multi-domain dialogues spanning the following 7 domains: taxi, restaurant, hotel, attraction, train, police, hospital. Following prior work with this dataset (Lee et al., 2019; Kim et al., 2019), we drop hospital and police from the training set as they are not included in the validation and test set. After the release of MultiWOZ 2.0 (Budzianowski et al., 2018), later iterations (Eric et al., 2019; Zang et al., 2020; Han et al., 2020) corrected some of the misannotations.

CrossWOZ is a Chinese-language WOZ dataset of 6k dialogues and over 102k utterances spanning the same 5 domains as the MultiWOZ validation and test sets: hotel, restaurant, attraction, metro, and taxi. CrossWOZ dialogues are annotated with dialogue states and dialogue acts, and average over 3.24 domains per dialogue, as opposed to the 1.80 of MultiWOZ.

For DSTC-9, Google NMT was used to translate MultiWOZ 2.1 to Chinese and CrossWOZ to English (Gunasekara et al., 2020; Li et al., 2021). To ensure translations of slot values in the dialog are faithful to the ontology, they replace them with their translations from a dictionary before feeding it to the NMT. This approach creates mixed-language sentences, shifting input distribution away from what public NMTs have been trained on, thus reducing quality (Moradshahi et al., 2020). Note that human translators were employed to proofread the translations and check certain slots to ensure values are correctly translated.

Table 1 shows a comparison of statistics for the training split of datasets used or created in this work. All the datasets have a closed ontology: the same entities appear in the train, validation, and test sets.

5.2 Models

We compare BART-CSP results against SOTA for each dataset. Models include the following:

- **TRADE** (Wu et al., 2019) uses a sequence-to-sequence architecture that encodes all utterances in the dialogue. It uses a pointer-generator to output the value of each slot.
- **MLCSG** (Quan and Xiong, 2020) extends TRADE by improving modeling of long contexts through a multi-task learning framework.
- **SOM** (Kim et al., 2019) considers dialogue state as an explicit fixed-sized memory and uses state operation to selectively update slot values at each turn.
- **SUMBT** (Lee et al., 2019) applies a BERT encoder to each utterance on the dialogue and a recurrent network to compute a representation of the whole dialogue, which is then matched against the ontology of each slot.
- **MinTL** (Lin et al., 2020) uses more recent pretrained models such as T5 (Raffel et al., 2019) and BART as the dialogue utterance encoder and builds an end-to-end dialogue model jointly learning dialogue state tracking, policy, and natural language generation tasks.
- **STAR** (Ye et al., 2021b) STAR uses two BERT models for encoding context and slot values. Additionally, they use a slot self-attention mechanism that can learn the slot correlations automatically. They use as input both the previous belief state and history of dialogue.

5.3 Implementation Details

BART-CSP is implemented using the Huggingface (Wolf et al., 2019) and GenieNLP (Campagna et al., 2019) libraries. We use the available open-source code for the other models. Hyperparameters for BART-CSP are discussed below; hyperparameters for other models are taken from the respective papers.

For semantic parsing we used *bart-base* (~139M parameters) for English and *mbart-large-50* (~611M parameters) model for other languages. For translation we used *mbart-large-50-many-to-many-mmt* (~611M parameters) which can translate directly between any pairs of 50 languages it supports. All models use a standard Seq2Seq architecture with a bidirectial encoder and left-to-right autoregressive decoder. All the models are pretrained using text denoising objective. *mbart-large-50-many-to-many-mmt* is additionally finetuned to do translation. BART uses BPE (Gage, 1994) to tokenize the input sentences whereas MBART uses sentence-piece model (Kudo and Richardson, 2018). We used Adam (Kingma and Ba, 2014) as our optimizer with a learning rate of 1×10^{-5} and used transformer warmup schedule (Popel and Bojar, 2018) with a warmup of 80. These hyperparameters were chosen based on a very limited hyperparameter search on the validation set. For the numbers reported in the paper, due to cost, we performed only a single run for each experiment.

	Dataset		
	RiSAWOZ	**CrossWOZ**	**MultiWOZ**
Languages	ZH, EN*, DE*	ZH, EN	EN, ZH
# Domains	12	5	7
# Dialogues	10,000	5,012	8,438
# Turns	134,580	84,692	115,424
# Slots	159	72	25
# Values	4,061	7,871	4,510

Table 1: Statistics of the training datasets (excluding validation and test). *Created using our methodology.

We batch sentences based on their input and approximate output token count for better GPU utilization. We set the total number of tokens to 700 for Mbart and 2000 for Bart models. We use gradient accumulation of 3 for Mbart and 9 for bart models boosting the effective batch size for better training. Our models were trained on NVIDIA V100 GPU using the AWS platform. For a fair comparison, all models were trained for the same number of iterations of 50K.

5.4 Metrics

We evaluate the models using the following two metrics:

- **Joint Goal Accuracy (JGA):** The standard metric of evaluation in DST is *joint goal accuracy*, which measures the average accuracy of predicting all slot assignments to exact match (EM) for any given turn. To compute this metric for CSP, the belief state predicted in previous turn is used as input for the current turn.
- **Gold Joint Goal Accuracy (GJGA):** This metric is similar to JGA but is calculated on a turn by turn basis, with ground-truth belief state used as input. Assuming the belief state correctly captures the state up to current turn of the dialogue, this metric acts as an oracle in evaluation removing the compounding effect of errors from previous turns.

6 Analysis and Results

Table 2 shows the results on test set for BART-CSP and previous SOTA models. For a better comparison we have included some details for each model:

- *Context Encoder*: The neural model used to encode the input.
- *Dialogue History*: If dialogue history is included in the input. A turn is defined as a pair of user utterance and agent response. “Partial” history means only a few turns of dialogue are kept while “Full” history models encode all previous turns.
- *Encodes State*: indicates if the user belief state up to the current turn is included in the input.
- *Predefined Slots or Ontology*: whether the model design or the data processing step needs knowledge of slot names or values.

As shown in Table 2, all previous models encode either a partial or full history. BART-CSP encodes significantly less information as it relies only on the current turn and the latest belief state. This simplifies model design and improves data efficiency for training (Yang et al., 2021a; Kapelonis et al., 2022).

Furthermore, models that rely on predefined ontologies require changes in architecture for new datasets. On the other hand, BART-CSP is a generative model that learns to copy slots from context and can be deployed for a new dataset as is.

On MultiWOZ, we report results for two models: MinTL which uses BART-large, and STAR[2] which uses BART as the context encoder. Our model outperforms MinTL despite using a smaller BART model, and achieves similar performance to STAR despite not having access to the full history. This shows bigger models do not necessarily yield better performance and model architecture and data representation are important too.

6.1 RiSAWOZ

The RiSAWOZ experiments shows that contextual semantic parsing delivers better accuracy than the state of the art on the original data sets; it is even more significant for translated data sets because it is more robust to translation errors.

BART-CSP improves the state of the art by 10.7% on JGA to 76.9% for the original Chinese data set. It provides an even greater improvement on the translated data sets: by 32.4% to 68.6% and by 42.5% to 65.9% on the automatically translated English and German data sets, respectively.

BART-CSP holds two major advantages over models that predict the slot-value pairs from the dialogue history. First, by distilling the belief state into a concise representation, it reduces noise in the input that would otherwise be present in a long dialogue history. Second, by taking the belief state as input, the model becomes more robust to translation errors in utterances from previous turns than

---

[2] SOLOIST (Peng et al., 2021) and SCORE (Yu et al., 2021) achieve better performance than STAR; however, these models use additional dialogue datasets to either pre-train or fine-tune their models. To keep the comparison fair, we only consider prior work that trained on MultiWOZ only.

	SOTA							BART-CSP	
Dataset	Model	Context Encoder	Dialogue History	Encodes State	Predefined Slots / Ont.	GJGA	JGA	GJGA	JGA
RiSAWOZ	MLCSG	Bi-GRU	Full	✗	✓	–	66.2	90.4	**76.9**
RiSAWOZ-EN-auto	MLCSG	Bi-GRU	Full	✗	✓	–	36.2	88.8	**68.6**
(-alignment)	MLCSG	Bi-GRU	Full	✗	✓	–	15.6	65.2	**22.9**
RiSAWOZ-DE-auto	MLCSG	Bi-GRU	Full	✗	✓	–	13.4	86.7	**65.9**
CrossWOZ	TRADE	Bi-GRU	Full	✗	✓	71.3†	36.1	**80.2**	**53.6**
CrossWOZ-EN	SOM	BERT	Partial	✓	✗	–	32.3‡	81.1	**52.3**
MultiWOZ 2.1	MinTL	BART-large	Partial	✓	✗	–	**53.6**‡	81.2	**53.7**
MultiWOZ 2.1	STAR	BERT	Full	✓	✓	78.7*	**56.7**	**81.2**	53.7
MultiWOZ-ZH 2.1	SUMBT	BERT	Full	✓	✓	–	**46.0**‡	75.9	**46.3**
MultiWOZ 2.4	STAR	BERT	Full	✓	✓	90.2*	**74.8**	**91.7**	70.4

Table 2: Comparison of results on test set for the BART-CSP model vs state-of-the-art on various datasets. The best result on JGA metric is in bold. GJGA is only bolded if SOTA is available. †RuleDST uses the previous system state and user dialogue acts as input. *STAR-GT uses ground truth previous dialogue state as input. ‡ indicates results are reported from their paper. We reproduced all results for models that source code and pretrained models were available to ensure correct comparison. "-align." denotes an ablation where translation is done without slot-value alignment.

models that accept dialogue histories. Hence it is even more effective on translated datasets.

Alignment is critical to generating a high-quality translated dialogue data set. The English and German semantic parser show only a degradation of 8-11% from the Chinese parser. In an ablation study of direct translation without alignment, the JGA on RiSAWOZ English drops from 68.6% to 22.9%, a difference of 45.7%.

Alignment ensures that entities are translated to the right phrase from the target ontology. For example, the phrase "姑苏区" is translated to "Aguzhou district" in a user utterance when the whole sentence is translated directly, but becomes "Gusu district" in the annotation. With alignment, both are translated identically to "Gusu district." The correspondence of utterance and belief state leads to higher DST performance.

6.2 CrossWOZ

Compared to MultiWOZ and RiSAWOZ, CrossWOZ is a more challenging dataset. Besides having longer conversations with more domains per dialogue, the cross-domain dependency is stronger. For example, the choice of location in one domain will affect the choice of location in a related domain later in the conversation, requiring models to have better contextual understanding. CrossWOZ is not only a smaller, more complex dataset than RiSAWOZ, but also exhibits a higher misannotation rate, to be discussed in Section 7. The current state of the art result was obtained with TRADE, which achieves 36.1% JGA.

The experiments with CrossWOZ also confirm that BART-CSP outperforms prior state-of-the-art models that encode full or partial history. Specifically, it exhibits an improvement of 17.5% in JGA on the original dataset and 20.0% in JGA on the English translated data set.

The GJGA metric for CrossWOZ was obtained by using RuleDST (Zhu et al., 2020), a set of handwritten rules specialized to the dataset to compute the new belief state from the ground truth user and system dialogue acts. BART-CSP outperforms the use of RuleDST in GJGA by 9%, showing that it is not necessary to handcraft these rules.

The translated CrossWOZ-EN data have been manually corrected for slot-value errors. Application of our automatic slot-value alignment technique would have greatly reduced the tedious manual effort required. In both GJGA and JGA, BART-CSP performs within 1% of the original Chinese dataset on English CrossWOZ.

6.3 MultiWOZ

MultiWOZ is a challenging data set because of the well-documented problem of misannotations in the data set (Eric et al., 2019). Misannotations teach the model to mispredict; conversely, correct predictions may be deemed to be incorrect. Thus the current state-of-the-art STAR model can only achieve an accuracy of 56.7%.

While BART-CSP accepts only the belief state as input context, the STAR model accepts both the previous belief state and the dialogue history. The latter offers an opportunity for the model to recover missing state values from the history, giving a 3% advantage in JGA over BART-CSP. However, we note that once an agent misinterprets a user input, it is not meaningful to measure the accuracy for subsequent turns since the conversation would have diverged from the test data.

On the other hand, parsing history has its own cost: (1) It is less data efficient as you need more data to learn the same task. (2) It requires a more complex model that can find relevant slots among a large number of sentences. BART-CSP outperforms STAR by a 2.5% improvement in GJGA, suggesting that having the dialogue history as input can be detrimental when the past turns of a dialogue have been predicted correctly. On the Chinese translation of MultiWOZ, BART-CSP does slightly better than state of the art, improving JGA by 0.3% to 46.3%.

We also compare BART-CSP performance to MinTL, which is not SOTA, but uses more recent BART and T5 models as encoder. The results show that better performance on DST cannot be achieved by solely relying on better encoders with improved pretraining as both models are outperformed by STAR which uses BERT.

Between MultiWOZ 2.1 and 2.4, BART-CSP results improve by 16.7% on JGA and 10.5% on GJGA, while STAR improves by 18.1% on JGA and 11.5% on GJGA, showing dependence of both BART-CSP and STAR on the quality of annotation. Because MultiWOZ 2.4 only corrects the validations and test sets, CSP is still affected by mis-annotations in the training dataset. The lack of an equally clean training set may be the reason BART-CSP does not exhibit as much improvement across the versions.

7 Data and Error Analysis

A manual inspection revealed the following sources of errors on the CSP model, showing some of the inference limitations and its susceptibility to misannotations.

7.1 Misannotations

A substantial portion of incorrect predictions is due to existing annotation errors in all the datasets. In particular, in a manual review of 200 randomly chosen turns from each dataset, RiSAWOZ exhibits a 10.0% misannotation rate while CrossWOZ and MultiWOZ exhibit 17.9% and 26% misannotation rates, respectively.

Prevalent misannotation error types observed in the three datasets are noted below, with examples in the appendix.

- **Inconsistency:** Annotation inconsistency is a common issue with the Wizard-of-Oz data collection method. Examples of inconsistent annotations include inferred slots and slots that are mentioned by the agent but ignored by the user.
- **Inexact Match:** Typos, i.e. minor mismatches between the utterances and the annotation slot values. Chinese is a homonym-heavy language. It is not unexpected for single-character mismatch typos to occur frequently in the dataset. A second kind of typo is for a character to be entirely missing in an entity name.
- **Missing Slots:** Sometimes, values for some slots are simply just not included in the annotations.

CrossWOZ and MultiWOZ 2.1 are also susceptible to the following:

- **Extra Slots:** The presence of slot names which are not mentioned by either the user or the agent.

The following additional annotations problems are salient in MultiWOZ 2.1:

- **Delayed Annotations:** Slot values that are already confirmed by the user show up at a later turn in the conversation.

In RiSAWOZ, the final parting turn in a dialogue has no annotations, indicating a state reset:

- **Empty Annotation (Hard State Reset):** Some turns are missing annotations altogether.

7.2 Logical Relation Inference

In RiSAWOZ, the model is expected to infer the logical relationships between entities. For instance, the price range for a *fast food* restaurant should be *cheap*; looking for an attraction to go to with a girlfriend implies the interest of a place where *best-for-crowd* = “lover’s date”; similarly the desired hotel rating is to be inferred from the utterance rather than explicitly mentioned.

For example:

- 我们一家是外地的，来苏州游玩，你可以帮我找一个在吴中区，中等消费水平的景点吗 (Our family is foreign, we have come to Suzhou to have fun, could you help me find, in the WuZhong area, a medium priced attraction?) The slot value pair 最适合人群= “家庭亲子” (*best-for-crowd* = “family”) must be inferred.

- 好的。再看看高新区有什么好吃的火锅店？(Okay. Can you also see if GaoXin District has some good hotpot?)
  The slot 价位= “偏贵” (*price* = “expensive”) must be inferred from the fact that the food is hotpot.

We found that most models struggle with this type of inference which requires higher level language understanding and reasoning. It is unclear whether this weakness is inherent to the model or whether it is an artifact of inconsistent annotations. Of the misannotations counted, 33% were missing the slot “Best for crowd.”

7.3 Expanded Range Inference

In CrossWOZ, when a user requests, for instance, a restaurant where the cost per person is within a price range, oftentimes the agent cannot find such a restaurant, and responds with a suggestion with a cost outside of that range. In subsequent slots, the price range is expanded to include that cost, typically by rounding to the nearest ten. For example:

- 你好，我想去吃饭。请帮我找一家人均消费是100-150元，有井冈山豆皮这个菜的餐馆。(Hello, I’d like to go eat. Please help me find a place where the cost per person is 100-150 yuan, and has Jingangshan bean curd.) In this turn, the slot value for cost per person is *100-150*. However, following the agent response, the slot changes: 只为您找到一家叫西江美食舫(健德桥店)，但是它家的人均消费是83元。(I could only find a place called XiJiang Gourmet Boat (Jiandeqiao location), but the cost per person is 83 yuan.) Since 83 is outside the original range *100-150*, the slot value is expanded to *80-150* for the remaining turns of the dialogue.

We find that CSP struggles with such turns and will typically mispredict the slot value, assigning the previous range rather than inventing a new one.

8 Conclusion

Given a dialogue dataset in one language, this paper shows how to build contextual semantic parsers for a new language using automatically machine-translated data. We propose an improved alignment approach for dialogues to ensure faithful translation of slot values. This removes the need for costly human-post editing used in all previous benchmarks.

We show that the compounding effects of translation noise across turns can be mitigated with a CSP model for dialogue state tracking. By leveraging pretrained seq2seq models such as BART, training with CSP can outperform state-of-the-art results on RiSAWOZ, CrossWOZ, and MultiWOZ-ZH, and remains competitive on MultiWOZ, despite not encoding any previous conversation turns or having access to a predefined ontology.

We use our methodology to create RiSAWOZ English and German, the first automatically created high-quality translated datasets for dialogue state tracking with no human in the loop. We have implemented our methodology as a toolkit[3] which developers can use to create a new multilingual dialogue dataset as well as a contextual semantic parser for it.

9 Limitations

Organic multilingual dialogue datasets (i.e. created without the use of translation) are scarce, which has limited the scope of our experiments. We would have liked to evaluate the generalization of our approach to other languages. For instance, we partially rely on machine translation models to create datasets. Available translation models for other language pairs, especially from/to low-resource languages have much lower quality, and it would be desirable to measure the effect of that in our experiments.

Our methodology has only been applied to Human-to-Human dialogues annotated with slot-values. Although our approach is independent of data collection technique and formal representation, it should be applied and tested on datasets annotated with representations other than slot-values to study how well it can generalize.

Previous studies (Ding et al., 2021; Hung et al., 2022) utilized human post-editing to guarantee the fluency and accuracy of the translated datasets. However, in order to reduce cost, we have decided not to use manual post-editing in this work. As a result, our findings could be an overestimation of the model’s actual performance in real-world scenarios. In future research, we plan to rectify this by manually post-editing the validation and test portions of the datasets.

---

[3] Code can be accessed at `https://github.com/stanford-oval/dialogues`

10 Ethical Considerations

Our translation method replaces the manual work needed to create multilingual dialogue datasets usually done via crowdsourcing. Instead, it requires some computation time which can be an environmental concern. However, in practice, such additional computing is small and much cheaper than the cost of human annotation for the same amount of data. The translation of the data set takes about half an hour on an Nvidia TITAN V GPU. Training takes about 6 hours on an Nvidia V100 GPU. We did not use crowdworkers for this paper. The error analysis was done by the authors.

Acknowledgements

This work is supported in part by the National Science Foundation under Grant No. 1900638, the Alfred P. Sloan Foundation under Grant No. G-2020-13938, Microsoft, Stanford HAI, and the Verdant Foundation.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Giovanni Campagna, Silei Xu, Mehrad Moradshahi, Richard Socher, and Monica S. Lam. 2019. Genie: A generator of natural language semantic parsers for virtual assistant commands. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, pages 394–410, New York, NY, USA. ACM.

Wenhu Chen, Jianshu Chen, Yu Su, Xin Wang, Dong Yu, Xifeng Yan, and William Yang Wang. 2018. Xl-nbt: A cross-lingual neural belief tracking framework. *arXiv preprint arXiv:1808.06244*.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, Jason D. Williams, Hong Yu, Diarmuid Ó Séaghdha, and Anders Johannsen. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8107–8117, Online. Association for Computational Linguistics.

Bosheng Ding, Junjie Hu, Lidong Bing, Sharifah Mahani Aljunied, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. Globalwoz: Globalizing multiwoz to develop multilingual task-oriented dialogue systems.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Philip Gage. 1994. A new algorithm for data compression. *C Users Journal*, 12(2):23–38.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019. Dialog state tracking: A neural reading comprehension approach. *arXiv preprint arXiv:1908.01946*.

Chulaka Gunasekara, Seokhwan Kim, Luis Fernando D'Haro, Abhinav Rastogi, Yun-Nung Chen, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, et al. 2020. Overview of the ninth dialog system technology challenge: Dstc9. *arXiv preprint arXiv:2011.06486*.

Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Wei Peng, and Minlie Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation. *arXiv preprint arXiv:2010.05594*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. *arXiv preprint arXiv:2005.02877*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Chiori Hori, Julien Perez, Ryuichiro Higashinaka, Takaaki Hori, Y-Lan Boureau, Michimasa Inaba, Yuiko Tsunomori, Tetsuro Takahashi, Koichiro Yoshino, and Seokhwan Kim. 2019. Overview of the sixth dialog system technology challenge: Dstc6. *Computer Speech & Language*, 55:1–25.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Chia-Chien Hung, Anne Lauscher, Ivan Vulić, Simone Paolo Ponzetto, and Goran Glavaš. 2022. Multi2woz: A robust multilingual dataset and conversational pretraining for task-oriented dialog. *arXiv preprint arXiv:2205.10400*.

Eleftherios Kapelonis, Efthymios Georgiou, and Alexandros Potamianos. 2022. A multi-task bert model for schema-guided dialogue state tracking. *arXiv preprint arXiv:2207.00828*.

John F Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 2(1):26–41.

Seokhwan Kim, Luis Fernando D'Haro, Rafael E Banchs, Jason D Williams, Matthew Henderson, and Koichiro Yoshino. 2016. The fifth dialog state tracking challenge. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 511–517. IEEE.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2019. Efficient dialogue state tracking by selectively overwriting memory. *arXiv preprint arXiv:1911.03906*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483, Florence, Italy. Association for Computational Linguistics.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jinchao Li, Qi Zhu, Lingxiao Luo, Lars Liden, Kaili Huang, Shahin Shayandeh, Runze Liang, Baolin Peng, Zheng Zhang, Swadheen Shukla, Ryuichi Takanobu, Minlie Huang, and Jianfeng Gao. 2021. Multi-domain task-oriented dialog challenge ii at dstc9. In *AAAI-2021 Dialog System Technology Challenge 9 Workshop*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. *arXiv preprint arXiv:2009.12005*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, Peng Xu, Feijun Jiang, Yuxiang Hu, Chen Shi, and Pascale Fung. 2021. BiToD: A bilingual multi-domain dataset for task-oriented dialogue modeling. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1 pre-proceedings (NeurIPS Datasets and Benchmarks 2021)*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Mehrad Moradshahi, Giovanni Campagna, Sina Semnani, Silei Xu, and Monica Lam. 2020. Localizing open-ontology QA semantic parsers in a day using machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5970–5983, Online. Association for Computational Linguistics.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. *arXiv preprint arXiv:2005.05298*.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Edoardo Maria Ponti, Ivan Vulić, Goran Glavaš, Nikola Mrkšić, and Anna Korhonen. 2018. Adversarial propagation and zero-shot cross-lingual transfer of word vector specialization. *arXiv preprint arXiv:1809.04163*.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70.

Jun Quan and Deyi Xiong. 2020. Modeling long context for task-oriented dialogue state generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7119–7124, Online. Association for Computational Linguistics.

Jun Quan, Shian Zhang, Qian Cao, Zizhong Li, and Deyi Xiong. 2020. Risawoz: A large-scale multi-domain wizard-of-oz dataset with rich semantic annotations for task-oriented dialogue modeling. *arXiv preprint arXiv:2010.08738*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Evgeniia Razumovskaia, Goran Glavaš, Olga Majewska, Anna Korhonen, and Ivan Vulić. 2021. Crossing the conversational chasm: A primer on multilingual task-oriented dialogue systems. *arXiv preprint arXiv:2104.08570*.

Hannes Schulz, Jeremie Zumer, Layla El Asri, and Shikhar Sharma. 2017. A frame tracking model for memory-enhanced dialogue systems. *arXiv preprint arXiv:1706.01690*.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. *arXiv preprint arXiv:1902.10461*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Puhai Yang, Heyan Huang, and Xian-Ling Mao. 2021a. Comprehensive study: How the context information of different granularity affects dialogue state tracking? *arXiv preprint arXiv:2105.03571*.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021b. Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14230–14238.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2021a. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *arXiv preprint arXiv:2104.00773*.

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021b. Slot self-attentive dialogue state tracking. *arXiv preprint arXiv:2101.09374*.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2021. Score: Pre-training for context representation in conversational semantic parsing. In *International Conference on Learning Representations*.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. *arXiv preprint arXiv:2007.12720*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467.

Qi Zhu, Kaili Huang, Zheng Zhang, Xiaoyan Zhu, and Minlie Huang. 2020. Crosswoz: A large-scale chinese cross-domain task-oriented dialogue dataset. *Transactions of the Association for Computational Linguistics*, 8:281–295.

A Appendix

A.1 Missannotation Examples

See Table 3 below for missannotation examples discussed in Section 7.

Error Type	**Dataset**	**Agent Utterance; User Utterance**	**Annotation**	**Correct Annotation**
Delayed Annotation	MultiWOZ	great , i can get you a ticket for that train . how many people are riding with you ?; i need to book it for 6 people , can i get the reference number too ?	train day = " thursday " train departure = " cambridge " train destination = " birmingham new street " train leaveat = " 10:00 "	**train book people = " 6 "** train day = " thursday " train departure = " cambridge " train destination = " birmingham new street " train leaveat = " 10:00 "
		[next turn] can i confirm you want to book this train for 6 people ?; yes , i would like to book the train for 6 people . i need the reference number , please .	**train book people = " 6 "** train day = " thursday " train departure = " cambridge " train destination = " birmingham new street " train leaveat = " 10:00 "	(already correct)
Extra Slot	MultiWOZ	; hello ! i am planning my trip there and i am trying to find out about an attraction called kettle s yard . what can you tell me about it ?	attraction name = " kettles yard " , **attraction area = " west "**	attraction name = " kettles yard "
Empty Annotation	RiSAWOZ	;你好，刚到苏州，想先找个餐厅吃点东西。有价位便宜的江浙菜餐厅吗 Hello, I just came to Suzhou and am looking for a restaurant to grab a bite. Is there a cheap Jiangzhe restaurant?	**null**	餐厅价位= " 便宜" 餐厅菜系= " 江浙菜" Restaurant Price = "Cheap" Restaurant Cuisine = "Jiangzhe"
Inexact Match	RiSAWOZ	;你好，我这几天在苏州度假，明天准备去狐狸家手工奶酪这家餐厅吃饭，但不是很了解，你能帮我查查那个餐馆附近有没有地铁能直达呢? ; Hello, I'm vacationing in Suzhou these several days, tomorrow I plan to go to the Fox Family Handmade Cheese restaurant to eat, but I don't really understand, can you help me look up whether there is direct subway access to anywhere near there?	餐厅名称= " 狐狸家手工酸奶酪" Restaurant Name = "Fox Family Handmade **Yogurt** (Yogurt = Sour Cheese)"	餐厅名称= " 狐狸家手工奶酪" Restaurant Name = "Fox Family Handmade Cheese"
Missing Slot	RiSAWOZ	有呀，推荐您去鑫花溪牛肉米粉。; 这家店地址在哪? Yes, I recommend you go to Xinhuaxi Beef Noodle; What is the address of this place?	餐厅价位= " 中等" 餐厅菜系= " 快餐简餐" Restaurant Price = "Medium" Restuarant Cuisine = "Quick and Easy"	餐厅价位= " 中等" 餐厅名称**= "** 鑫花溪牛肉米粉**"** 餐厅菜系= " 快餐简餐" Restaurant Price = "Medium" **Restaurant Name = "Xinhuaxi Beef Noodle"** Restaurant Cuisine = "Quick and Easy"

Table 3: Prevalent annotation error types found in the datasets.

Teacher Intervention: Improving Convergence of Quantization Aware Training for Ultra-Low Precision Transformers

Minsoo Kim[1], **Kyuhong Shim**[2], **Seongmin Park**[1], **Wonyong Sung**[2] and **Jungwook Choi**[1]*
[1]Department of Electronic Engineering, Hanyang University
[2]Department of Electrical and Computer Engineering, Seoul National University
`{minsoo2333, skstjdals, choij}@hanyang.ac.kr`
`{skhu20, wysung}@snu.ac.kr`

Abstract

Pre-trained Transformer models such as BERT have shown great success in a wide range of applications, but at the cost of substantial increases in model complexity. Quantization-aware training (QAT) is a promising method to lower the implementation cost and energy consumption. However, aggressive quantization below 2-bit causes considerable accuracy degradation due to unstable convergence, especially when the downstream dataset is not abundant. This work proposes a proactive knowledge distillation method called *Teacher Intervention* (TI) for fast converging QAT of ultra-low precision pre-trained Transformers. TI intervenes layer-wise signal propagation with the intact signal from the teacher to remove the interference of propagated quantization errors, smoothing loss surface of QAT and expediting the convergence. Furthermore, we propose a *gradual* intervention mechanism to stabilize the recovery of subsections of Transformer layers from quantization. The proposed schemes enable fast convergence of QAT and improve the model accuracy regardless of the diverse characteristics of downstream fine-tuning tasks. We demonstrate that TI consistently achieves superior accuracy with significantly lower fine-tuning iterations on well-known Transformers of natural language processing as well as computer vision compared to the state-of-the-art QAT methods.

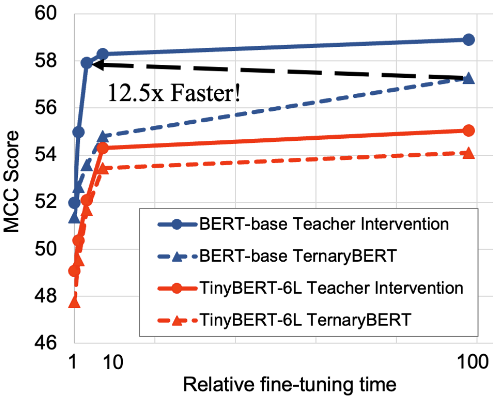


Figure 1: Comparison of fine-tuning time and accuracy of ternary weight quantized Transformer models between Teacher Intervention (TI) and TernaryBERT (BERT-base and TinyBERT-6L for CoLA task). TI achieves higher accuracy within 12.5× shorter fine-tuning time.

1 Introduction

The Transformer-based pre-trained neural networks have significantly improved the performance of various applications of artificial intelligence, including natural language processing (NLP) (Devlin et al., 2019; Raffel et al., 2020; Brown et al., 2020) and computer vision (CV) (Dosovitskiy et al., 2020; Touvron et al., 2021; Liu et al., 2021a). The self-attention mechanism represents these models (Vaswani et al., 2017), which links different symbols within a sequence to obtain a relational representation. Thanks to the exceptional performance of the pre-trained Transformer models, there have been increasing needs for their efficient deployment. However, the gigantic size of the pre-trained Transformer models hinders straightforward implementation. Even relatively small models like BERT-base (Devlin et al., 2019) contain a few hundred million parameters, incurring profound memory and computation overhead for resource-constrained devices with limited memory and computing fabric. Therefore, seminal research efforts attempted to reduce this burden via model compression. (Behnke and Heafield, 2020) and (Gordon et al., 2020) pruned unimportant weights to reduce the number of parameters, while (Mao et al., 2020) further employed low-rank matrix factorization. In addition, Knowledge Distillation (KD) (Hinton et al., 2015) was employed in (Sanh et al., 2019; Sun et al., 2019, 2020; Wang et al., 2020) to transfer knowledge of the original model (*teacher*) to the compressed one (*student*) by mimicking the

---

*Corresponding Author

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 916–929
May 2-6, 2023 ©2023 Association for Computational Linguistics

Teacher's behavior.

Among many model compression techniques, *quantization-aware training* (QAT) stands out for its recent success in reducing computational complexity and memory requirements of Transformer models (Bhandare et al., 2019; Zafrir et al., 2019; Kim et al., 2021). QAT reflects quantization errors during the forward pass computation of stochastic gradient descent to train a more accurate quantized model. However, quantizing weight parameters of Transformers to a precision lower than 2-bits degrades the accuracy, especially when the dataset size for the target downstream tasks is not large enough (Zhang et al., 2020b; Bai et al., 2021). Although few-sample fine-tuning of Transformer models has been reported to be highly unstable (Grießhaber et al., 2020; Yu et al., 2021; Zhang et al., 2020a; Dodge et al., 2020; Mosbach et al., 2020), efforts on understanding and improving QAT on small dataset tasks are limited. XTC (Wu et al., 2022) recently revealed that a simple expansion in fine-tuning iterations could heal the QAT accuracy. Still, it caused an order-of-magnitude increase in QAT time, hindering the broad deployment of quantized Transformers.

This work proposes a proactive KD method called *Teacher Intervention* (TI) for fast converging QAT of ultra-low precision pre-trained Transformers. We reveal that the difficulty of quantization on few-sample fine-tuning originates from disruption of loss surface due to quantization error propagation. To mitigate this undesirable phenomenon, we propose TI to intervene layer-wise signal propagation with the intact signal from the teacher. TI removes the interference of propagated quantization errors to smooth out the loss surface and expedite the convergence. We further discover that subsections of Transformer layers exhibit different susceptibility to quantization error for diverse downstream tasks. Thus, we propose a gradual intervention mechanism that first intervenes at the attention output for stable tuning of the feed-forward network, followed by self-attention map intervention for its recovery from quantization. The proposed gradual intervention along the subsections of Transformer layers enables fast convergence of QAT and improves the model accuracy regardless of the diverse characteristics of downstream fine-tuning tasks. We perform an extensive evaluation on various fine-tuned Transformers (BERT-base/large (Devlin et al., 2019), TinyBERT-4L/6L (Jiao et al., 2020), and SkipBERT-6L (Wu et al., 2022) for NLP, and ViT (Dosovitskiy et al., 2020) for CV) and demonstrate that TI consistently achieves superior accuracy with lower fine-tuning iterations compared to the state-of-the-art QAT methods. In particular, TI outperforms TernaryBERT (Zhang et al., 2020b) on GLUE tasks with 12.5× savings in fine-tuning hours, as shown in Fig.1.

2 Related Work

2.1 Knowledge Distillation for BERT Compression

Knowledge distillation (KD) (Hinton et al., 2015) is a transfer learning framework that passes on knowledge of a large model (*teacher*) to a smaller one (*student*). Since KD provides extra guidance on how the student should behave, it can help mitigate accuracy degradation for model compression. Therefore, KD has been widely employed in training smaller BERT models for various application domains.

The most common distillation approach is to match the probability distribution from the final output softmax between the teacher and student for the same input, as shown in DistilBERT (Sanh et al., 2019). In addition to the distillation loss at the model output, PKD (Sun et al., 2019) suggested loss on intermediate output that matched the normalized output of the teacher and the student at each Transformer layer. MobileBERT (Sun et al., 2020) also employed per-head attention map transfer along with the customized network structure for constructing efficient Transformers. MiniLM (Wang et al., 2020) further transferred knowledge from the self-attention map as well as the value-relation. Considering the structural mismatch between the Teacher and Student models, MiniLM performed distillation only at the last Transformer layer.

While the above mentioned studies focused on the task-agnostic BERT, there have been several efforts (Tang et al., 2019; Aguilar et al., 2020) to train tiny task-specific students. In this line of study, the task-specific, downstream fine-tuned BERT is first prepared, and the Student is trained with KD by utilizing this fine-tuned model as a teacher. As a hybrid approach, TinyBERT (Jiao et al., 2020) proposed a two-step KD, the first step for general distillation, followed by task-specific distillation.

Although extensive research has been conducted to utilize KD for BERT compression, there are limited efforts in investigating and developing KD techniques for model quantization, an orthogonal model compression method. In this work, we develop a new KD technique primarily focusing on improving QAT of pre-trained Transformers. In particular, we reveal that more aggressive intervention of the teacher on the subsections of each Transformer layer helps the ultra-low precision model regain the model accuracy.

2.2 Quantization for Ultra-Low Precision BERT

Quantization is a promising technique for reducing the high inference cost of large-scale models without changing the model structure. Instead of representing numbers with the 32-bit floating-point (FP32) format, employing fixed-point representation, such as 8-bit integer (INT8) quantization, has achieved significant speedup and storage savings for BERT (Zafrir et al., 2019; Kim et al., 2021). However, direct quantization of weight parameters would suffer accuracy degradation of the original model accuracy when the quantization bit-precision is low. Therefore, quantization-aware training (QAT) is commonly applied for ultra-low precision model quantization.

Recently, QAT has been applied for compressing BERT with precision lower than 2-bit. TernaryBERT (Zhang et al., 2020b) represents each weight element into one of three values $\{-1, 0, 1\}$. TernaryBERT actively incorporates KD into QAT for improving accuracy degradation. Especially, KD with the MSE loss on the attention score (before taking Softmax) and the output of each Transformer layer is employed for QAT. To further reduce the bit-precision, BinaryBERT (Bai et al., 2021) suggested a modified QAT procedure that initializes the weights for binary quantization. However, ternarizing or binarizing weight parameters significantly degrades the model accuracy, especially when the dataset size for the target downstream tasks is not large enough.

In fact, it has been reported that finetuning BERT on downstream tasks with insufficient data is highly unstable (Grießhaber et al., 2020; Yu et al., 2021). As a result, several works proposed modified finetuning procedures for improving the stability (Zhang et al., 2020a; Dodge et al., 2020; Mosbach et al., 2020). Still, the proposed approaches do not address the sensitivity of Transformer models on QAT for small datasets. XTC (Wu et al., 2022) recently proposed a QAT method with significantly increased iterations and data augmentation to improve quantization accuracy of ultra-low bit precision Transformers. As illustrated in Fig. 1, however, this prolonged fine-tuning results in sizable deployment overhead, let alone costly data augmentation. In this work, we discover that quantization significantly disrupts the propagation of self-attention in Transformer layers hindering the optimization process of QAT. Therefore, we propose a new KD-based method that proactively intervene the error propagation to improve convergence of QAT methods.

3 Background and Motivation

3.1 Transformer Layer

The BERT model (Devlin et al., 2019) is built with Transformer layers (Vaswani et al., 2017), which include two main sub-modules: Multi-Head Attention (MHA) and Feed-Forward Network (FFN). Input to the l-th Transformer layer is $\mathbf{X}_l \in \mathbb{R}^{n\times d}$ where n and d are the sequence length and hidden state size, respectively. Let H be the number of attention heads and $d_h = d/H$. $\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V \in \mathbb{R}^{d\times d_h}$ are the weight parameters converting $\mathbf{X}_l$ into Query ($\mathbf{Q} = \mathbf{X}_l\mathbf{W}_h^{\mathbf{Q}}$), Key ($\mathbf{K} = \mathbf{X}_l\mathbf{W}_h^{\mathbf{K}}$), and Value ($\mathbf{V} = \mathbf{X}_l\mathbf{W}_h^{\mathbf{V}}$), respectively. Then, attention score ($\mathbf{AS}_h = QK^\top$), self-attention map ($\mathbf{SA}_h = \text{Softmax}_h(\frac{\mathbf{AS}_h}{\sqrt{d}})$), and attention context ($\mathbf{AC} = \mathbf{SA}_h\mathbf{V}$) can be defined.

The attention output ($\mathbf{AO}$) is defined as $\mathbf{AO} = \text{MHA}(\mathbf{X}_l) = \text{Concat}(\mathbf{AC}_1, \mathbf{AC}_2, ...\mathbf{AC}_{N_H}) \times \mathbf{W}^O$. Motivated by (Kobayashi et al., 2020), the attention output can be re-written per each token i:

$$\text{MHA}(\mathbf{X}_l)(i) = \sum_{j=1}^{n} \alpha_{i,j} f(\mathbf{X}_l(j)), \qquad (1)$$

where $f(x) := (x\mathbf{W}^V + b^V)\mathbf{W}^O$ and $\alpha_{i,j}$ is j'th value of i'th token in SA_h. Therefore, the computation of MHA consists of two parts: self-attention generation (SA-GEN) corresponding to the attention map (α) and self-attention propagation (SA-PROP) corresponding to $f(x)$. Fig. 3 illustrates the structure of a Transformer layer indicating regions corresponding to SA-GEN and SA-PROP. FFN consists of two fully-connected layers with weight parameters $\mathbf{W}^1$ and $\mathbf{W}^2$:

$$\text{FFN}(\mathbf{Y}_l) = \text{GeLU}(\mathbf{X}_l\mathbf{W}^1 + b^1)\mathbf{W}^2 + b^2. \qquad (2)$$

Therefore, an output of Transformer layer X_{l+1} is defined as:

$$\begin{aligned} \mathbf{Y}_l &= \text{LayerNorm}(\mathbf{X}_l + \text{MHA}(\mathbf{X}_l)), \\ \mathbf{X}_{l+1} &= \text{LayerNorm}(\mathbf{Y}_l + \text{FFN}(\mathbf{Y}_l)). \end{aligned} \tag{3}$$

3.2 Quantization-Aware Training

Quantization-aware training (QAT) emulates inference-time quantization during fine-tuning of the full-precision model to adjust the parameters to be robust to the quantization error (Hwang and Sung, 2014) (Hubara et al., 2016). In particular, ternary quantization represents all the weight parameters ($\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V, \mathbf{W}^O, \mathbf{W}^1, \mathbf{W}^2$) into ternary values $t \in \{+1, 0, -1\}$ along with a scale factor s for sub-2bit inference. In this work, we follow the approach of TWN (Zhu et al., 2016) that analytically estimates the optimal scale factor s and the ternary weight $\mathbf{W}_t = \{t\}^{|\mathbf{W}|}$ to minimize $\|\mathbf{W} - s \cdot \mathbf{W}_t\|$.

Due to aggressive bit-reduction, ternary quantization causes significant accuracy loss. KD can help compensate for accuracy degradation, where the original full-precision model works as a teacher to guide the training of the quantized model as a student. In case of Transformer models, TernaryBERT (Zhang et al., 2020b) applied KD on every output activation X_{l+1} as well as the attention score (**AS**) with mean squared error (MSE) loss:

$$L_{trm} = \sum_{l=1}^{L+1} \text{MSE}(\mathbf{X}_l^S, \mathbf{X}_l^T) + \sum_{l=1}^{L} \text{MSE}(\mathbf{AS}_l^S, \mathbf{AS}_l^T), \tag{4}$$

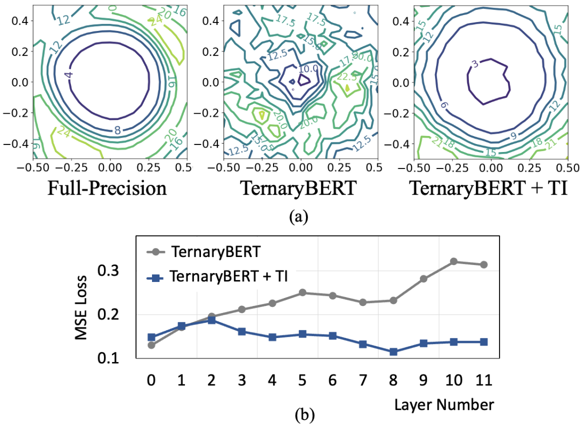
where S and T represent the student and teacher models, respectively. Also, the output logits of the student (P^S) and the teacher (P^T) are used in TernaryBERT to compute the cross-entropy (CE) loss:

$$L_{pred} = \text{CE}(P^S, P^T). \tag{5}$$

We follow the settings of TernaryBERT as our baseline QAT method.

3.3 Challenges

Despite attempts to bridge the accuracy gap, prior works on ultra-low precision Transformers (Bai et al., 2021; Zhang et al., 2020b) still suffer noticeable accuracy degradation, especially when the dataset size is small. Recall (Mosbach et al., 2020) that the few-sample fine-tuning is unstable. But we observed that QAT often fails even if it fine-tunes from the successfully trained model.



(a)

(b)

Figure 2: (a) Loss landscape visualization of a fine-tuned Transformer for a few-sample task (BERT-base, CoLA): A well-trained full-precision model and the quantized models without and with Teacher Intervention (TI). (b) MSE loss at the output of Transformer layers of TernaryBERT without and with TI

To gain intuition on this failure, we visualize the loss landscape of the quantized Transformers fine-tuned for a few-sample task (BERT-base, CoLA) in Fig. 2(a). In contrast to the smooth loss surface of the well-trained full-precision model, TernaryBERT exhibits sharp curvatures with many valleys, suffering unstable fine-tuning due to quantization. Investigating the internal behavior of Transformer layers under quantization elucidates the cause of unstable QAT. We measure the mean-square error (MSE) of the output of each Transformer layer between the full-precision baseline (teacher) and TernaryBERT (student). Fig. 2(b) reveals a prominent trend that MSE grows over the layers. (Similar trends can be observed in the other GLUE tasks.) This inflated error along the layers would degrade the model's accuracy.

In this work, we focus on managing this aggravating impact of quantization errors on Transformers with a proactive knowledge distillation called *Teacher Intervention* (TI). Interestingly, as shown in Fig. 2(b), TI successfully suppresses the error propagation and flattens the loss surface for favorable convergence without precipitously increased fine-tuning iterations. We discuss the technical details of TI in the next section.

4 Method

4.1 Teacher Intervention

Teacher intervention (TI) is a KD method that aggressively intervenes in the student's signal propagation along the Transformer layers to suppress propagation of quantization error. Fig. 3(b) illus-

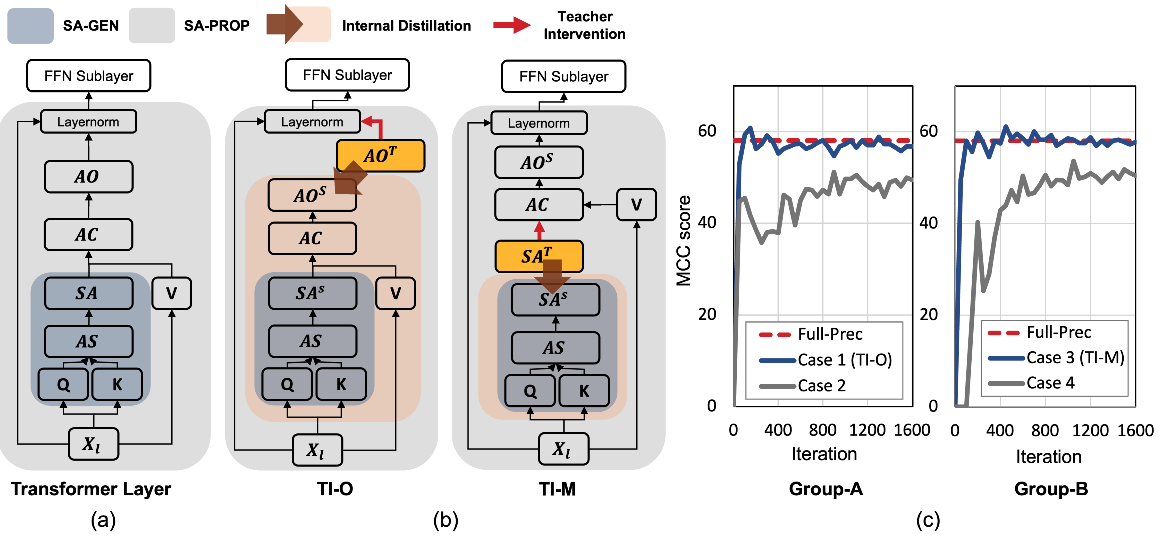


Figure 3: (a) Architecture of Transformer layer. (b) Locations of Teacher Intervention: output intervention (TI-O) and map intervention (TI-M). $\mathbf{AO}^T$ and $\mathbf{SA}^T$ are teacher's attention outputs. (c) Accuracy curves of controlled experiments. Group-A: Quantize all sub-layers with TI-O (Case 1) or do not quantize MHA (Case 2), Group-B: Quantize all sub-layers with TI-M (Case 3) or do not quantize SA-GEN (Case 4). Note rapid convergence for the cases with TI.

trates the two options for teacher intervention. First, intervention on the attention output (a.k.a. *output intervention*, TI-O) replaces the student's attention output ($\mathbf{AO}^S$ in Fig. 3(b)) in each Transformer layer with the teacher's ($\mathbf{AO}^T$ in Fig. 3(b)). In this case, the FFN sub-layers are trained with ultra-low precision quantization without concerns of erroneous input from the preceding MHA. Meanwhile, the computation within the MHA sub-layer is quantized for internal distillation. Similarly, intervention on the self-attention map (a.k.a. *map intervention*, TI-M) replaces the student's SA-GEN output ($\mathbf{SA}^S$ in Fig. 3(b))with the teacher's ($\mathbf{SA}^T$ in Fig. 3(b)). [1]

The development of TI is motivated by the previous observation of the aggravating impact of quantization error along the layers (cf. Fig. 2(b)). We conjecture that the root cause of this phenomenon is error propagation instead of the quantization error itself. To confirm our hypothesis, we conducted controlled experiments for TI-O and TI-M with two groups of quantization cases:

- Group-A: Quantize all sub-layers with TI-O (Case1) or do not quantize MHA (Case2).
- Group-B: Quantize all sub-layers with TI-M (Case3) or do not quantize SA-GEN (Case4).

The key difference between the two groups is that Case2/4 propagates the quantization error through the sub-layers while Case1/3 does not, thanks to TI. Fig. 3(c) shows the convergence curves of the four cases on CoLA. As shown in the figure, Case1/3 converges rapidly to full-precision accuracy despite the ultra-low bit quantization in all the sub-layers. Whereas Case2/4 converges slowly to the sub-optimal point with noticeable accuracy degradation. Although Case2/4's MHA/SA-GEN computations are in full-precision, the error propagated from its preceding quantized FFN sub-layers still affects it, corrupting the attention output/self-attention map. On the other hand, TI-O and TI-M interrupt this error propagation to stabilize QAT on FFN sub-layers and MHA outputs, respectively. Therefore, Fig. 2(b) confirms that the error propagation disappears when TI is applied.

Motivated by this insightful observation, we devise a new QAT method that employs TI for the step-by-step reconstruction of sub-layers of Transformers. In the first step, TI is used to fine-tune the quantized weights of either FFN (TI-O) or SA-PROP (TI-M) sub-layers. Note that the convergence in this step is quick, as shown in Fig. 3(c). Therefore, Step1 takes only a fraction of fine-tuning epoch. In the second step, quantization is applied to the entire weights of Transformer layers for QAT. Since part of the parameters is already trained in Step1 with the guidance of TI, Step2 converges faster to a superior local minimum. In Sec. 5, we demonstrate this improved convergence leads to boosts in accuracy for ultra-low bit Transformers.

---

[1] We empirically investigated other options for teacher intervention and found that TI-O and TI-M were most representative.

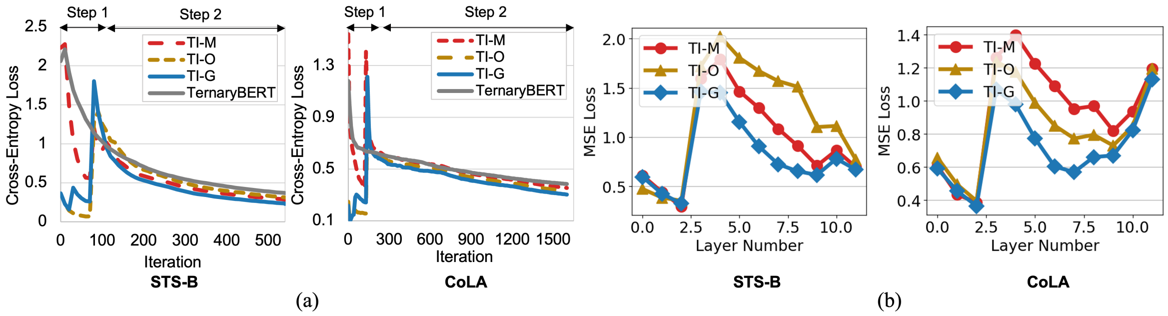


Figure 4: (a) Cross-entropy loss of QAT with different TI methods, and (b) Layer-wise MSE loss for different TI methods (Left: STS-B, Right: CoLA). Note that TI-G converges to lower loss than TI-M and TI-O on both tasks.

4.2 Gradual Teacher Intervention

Type	TI-M			TI-O			TI-G		
Sub-layer	GEN	PROP	FFN	GEN	PROP	FFN	GEN	PROP	FFN
Step1-Phase1	Q+TI	Q	Q	Q+TI		Q	Q+TI		Q
Step1-Phase2							Q+TI	Q	Q
Step2	Q			Q			Q		

Table 1: Quantization settings for teacher intervention. Different from TI-M and TI-O, gradual teacher intervention (TI-G) applies TI to quantization (Q) from large (GEN+PROP) to small (GEN) scope in Step1.

Given multiple teacher intervention options, which would achieve the best performance? In this section, we propose a unified approach that gradually applies the output intervention followed by the map intervention. Note that the two TI options have strengths and weaknesses. For example, TI-O focuses on tuning the FFN sub-layers, but it lacks consideration of the self-attention map recovery. On the other hand, TI-M is best suited for recovering the self-attention map, but it does not protect signal propagation through SA-PROP. Interestingly, we empirically discover that different downstream tasks of the pre-trained Transformers have diverse preferences; e.g., BERT-base fine-tuned on STS-B is sensitive to disruption in the self-attention map while the model fine-tuned on CoLA prefers careful tuning of the FFN sub-layers. Therefore, developing a unified solution that utilizes teacher intervention in various scopes is beneficial. As a natural combination, we propose a gradual teacher intervention mechanism that applies TI-O first to tune the FFN sub-layers (Step1-Phase1), followed by TI-M to recover the self-attention map (Step1-Phase2). The proposed method, called *gradual intervention* (TI-G), has shown practical success in most ultra-low bit Transformers studied in this work. (We conducted an ablation study for the other possibilities of a unified solution for teacher intervention in A.3.) Table. 1 summarizes the TI settings. Note that the two phases of TI-G in Step1 do not increase the total number of iterations thanks to fast convergence.

Fig. 4(a) shows the convergence curves of different TI options for STS-B and CoLA. As discussed earlier, TI-M shows better convergences than TI-O on STS-B, but the opposite trend is shown on CoLA. Nevertheless, TI-G always shows superior convergence compared to the other options, demonstrating its universal applicability. In particular, on STS-B, TI-G benefits from the output intervention in the first phase to favorably warm up the FFN sub-layers, and thus the map intervention in the next phase can reduce the loss more than TI-M. A similar situation happens in the case of CoLA.

Fig. 4(b) investigates the internal behavior of Transformers via the layer-wise MSE at the output of MHA. As discussed earlier, the fine-tuned Transformers exhibit distinct characteristics depending on the downstream tasks. For example, in the case of STS-B, TI-M is more effective in reducing MSE since the recovery of the self-attention map is essential. On the other hand, CoLA has preferred TI-O for its focus on tuning FFN sub-layers. In both cases, TI-G's gradual intervention with decreasing scopes from the attention output to the self-attention map helps achieve the smallest MSE. Therefore, this investigation suggests that the proposed unified intervention mechanism can manage diverse characteristics of fine-tuned Transformers for various downstream tasks. We demonstrate the general applicability of TI-G in the next section.

5 Experiments

In this section, we evaluate TI for QAT of fine-tuned BERT and vision Transformer with ternary weight quantization. We demonstrate that TI significantly boosts the convergence of QAT to achieve higher model accuracy within shorter fine-tuning time compared to the state-of-the-art QAT methods

(TernaryBERT (Zhang et al., 2020b) and XTC (Wu et al., 2022)) on various Transformer tasks and datasets. More extensive evaluation results on a wide range of BERT models as well as the ablation studies on various aspects of TI can be found in Appendix A.3. Due to limited space, detailed experiment settings are also summarized in Appendix A.1.

Iterations	CoLA	RTE	MRPC	STS-B	Avg. Ratio
Budget-O	1,603	233	343	538	1
Budget-A	13,325	4,468	7,050	10,066	16
Budget-C	159,909	53,621	84,611	120,795	200

Table 2: Different budgets for fine-tuning iterations for GLUE tasks.

5.1 QAT Accuracy on Few-Sample BERT

First, we perform an extensive performance comparison of the proposed teacher intervention methods with the state-of-the-art QAT methods: TernaryBERT (Zhang et al., 2020b) and XTC (Wu et al., 2022). We evaluate TI on fine-tuned BERT$_{base}$ (12 layer), BERT$_{large}$ (24 layer) (Devlin et al., 2019), TinyBERT (Jiao et al., 2020) (6 layer), and SkipBERT (Wu et al., 2022) (6 layer) for GLUE tasks (Wang et al., 2018). We report the experimental results for the small dataset tasks (less than 10k of dataset size) here; the results on the other models and tasks are reported in Appendix A.2.

To investigate the convergence of these QAT methods, we consider the following fine-tuning budgets:

- Budget-O: The number of iterations reported in the original paper (Zhang et al., 2020b).
- Budget-O2: 2× iterations of Budget-O.
- Budget-O4: 4× iterations of Budget-O.
- Budget-A/C: prolonged fine-tuning budgets employed in XTC (Wu et al., 2022).

A summary of fine-tuning budgets is shown in Table 2. Note that the number of fine-tuning iterations of Budge-A is roughly 16× larger than Budget-O. In the following subsections, we categorize the experiments to two scenarios: few-sample fine-tuning (Scenario-1: Budget O/O2/O4) and prolonged fine-tuning (Scenario-2: Budget A/C).

Scenario-1: Few-Sample Fine-tuning: Table 3 summarizes the experimental results of few-sample fine-tuning. Consistent with the prior observations (Wu et al., 2022), the accuracy of TernaryBERT increased as the fine-tuning budget grew from Budget-O to Budget-O4. But all teacher intervention options have improved these baseline accuracies. Note that the preference between TI-M and TI-O varies across the tasks and models. For example, TI-M is mainly preferred on CoLA and RTE, while there is a marginally higher preference for TI-O on STS-B. For most cases, however, TI-G outperforms the other TI options. Although the accuracies of TI-M and TI-O are similar, TI-G's accuracy significantly exceeds both. From these observations, we can conclude that the proposed gradual intervention significantly improves the QAT convergence for regaining model accuracy.

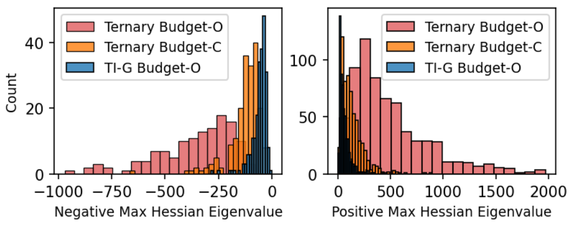


Figure 5: Convergence analysis of QAT methods with Hessian Max Eigenvalues.

Scenario-2: Prolonged Fine-tuning: Table 4 summarizes the experimental results of prolonged fine-tuning. For fair comparisons, we followed the instructions of XTC to match the fine-tuning budgets, learning rates, and the model compression mechanism on both TinyBERT and SkipBERT. As expected, TI-G's accuracy is significantly increased from Budget-O to Budget-A/C than Budget-O4. Interestingly, TI-G's accuracy is higher than TernaryBERT and XTC for these prolonged fine-tuning budgets with noticeable margins. In fact, XTC's average accuracy on Budget-A (for SkipBERT-6L) is lower than TI-G's average accuracy on Budge-O4 (for TinyBERT-6L), highlighting the superior convergence of TI-G compared to XTC.

5.2 QAT Accuracy on Vision Transformers

We further evaluate the proposed teacher intervention method on ViT. Table 5 summarizes the QAT accuracies of ViT fine-tuned for CIFAR10, CIFAR100, and ImageNet with the fine-tuning budgets following the original paper (Dosovitskiy et al., 2020). As shown in the Table. 5, TI-G outperforms TernaryBERT on all the QAT cases for ternary weight quantization. It is noteworthy that TI-G is exceptionally effective for fine-tuned ImageNet.

		BERT$_{base}$				BERT$_{large}$			
QAT Type	Iterations	RTE	CoLA	STS-B	MRPC	RTE	CoLA	STS-B	MRPC
Full-Prec	Budget-O	73.28	58.04	89.24	87.77	70.39	60.31	89.83	88.43
TernaryBERT	Budget-O	67.44 ±1.30	49.44 ±1.11	87.58 ±0.09	85.58 ±0.58	63.36 ±1.01	53.25 ±1.20	88.65 ±0.16	88.31 ±0.20
TI-Map	Budget-O	69.60 ±0.92	51.37 ±1.23	87.75 ±0.12	86.25 ±1.03	66.13 ±1.12	52.40 ±1.65	88.61 ±0.16	88.67 ±0.32
TI-Output	Budget-O	69.31 ±0.57	50.91 ±0.94	87.76 ±0.22	86.04 ±0.61	65.20 ±0.94	52.66 ±1.27	88.56 ±0.16	88.68 ±0.51
TI-G	Budget-O	**70.32** ±0.72	**51.98** ±1.35	**87.77** ±0.29	**86.44** ±0.49	**66.27** ±0.79	**54.12** ±1.13	**88.66** ±0.05	**88.80** ±0.41
TernaryBERT	Budget-O2	70.51 ±0.41	52.65 ±0.77	88.04 ±0.14	86.00 ±0.38	66.42 ±0.62	55.72 ±1.26	89.00 ±0.09	88.22 ±0.82
TI-G	Budget-O2	**71.48** ±0.36	**54.98** ±0.66	**88.04** ±0.18	**88.63** ±0.55	**68.11** ±0.75	**57.55** ±1.69	**89.12** ±0.04	**88.25** ±0.46
TernaryBERT	Budget-O4	71.23 ±0.42	53.57 ±0.80	88.22 ±0.04	86.58 ±0.32	67.50 ±0.95	57.70 ±0.64	89.12 ±0.01	**89.12** ±0.84
TI-G	Budget-O4	**73.16** ±0.36	**57.92** ±1.19	**88.48** ±0.61	**89.56** ±0.52	**69.67** ±0.72	**59.89** ±1.07	**89.33** ±0.10	88.74 ±0.76
		TinyBERT-4L				TinyBERT-6L			
QAT Type	Iterations	RTE	CoLA	STS-B	MRPC	RTE	CoLA	STS-B	MRPC
Full-Prec	Budget-O	68.23	43.06	87.07	87.76	74.00	57.78	88.74	87.35
TernaryBERT	Budget-O	63.15 ±0.50	32.15 ±1.43	83.33 ±0.36	84.90 ±0.38	68.74 ±1.42	47.77 ±0.35	87.29±0.12	84.89 ±0.53
TI-G	Budget-O	**64.29** ±0.72	**35.17** ±1.35	**83.58** ±0.29	**85.48** ±0.49	**69.38** ±0.78	**49.08** ±1.24	**87.31** ±0.11	**86.30** ±0.63
TernaryBERT	Budget-O2	64.74 ±0.76	34.49 ±1.89	84.10 ±0.34	85.73 ±0.05	69.67 ±0.95	49.54 ±0.22	87.51 ±0.04	86.61 ±0.41
TI-G	Budget-O2	**64.98** ±1.45	**36.30** ±0.24	**84.27** ±0.22	**86.44** ±0.31	**71.12** ±0.63	**50.38** ±0.56	**87.56** ±0.09	**86.80** ±0.42
TernaryBERT	Budget-O4	65.10 ±0.55	35.91 ±0.31	84.36 ±0.26	86.70 ±0.13	70.75 ±0.36	51.66 ±0.51	87.75 ±0.04	86.76 ±0.23
TI-G	Budget-O4	**65.22** ±0.55	**37.40** ±1.30	**84.39** ±0.27	**87.22** ±0.13	**72.13** ±0.12	**52.08** ±0.01	**87.90** ±0.14	**87.15** ±0.39

Table 3: Accuracy comparison of QAT methods on BERT family (few-sample GLUE tasks) without and with TI for regular fine-tuning iterations (Budget-O/O2/O4). Each experiment is repeated 10 times.

		TinyBERT-6L					SkipBERT-6L			
QAT Type	Iterations	RTE	CoLA	STS-B	MRPC	QAT Type	RTE	CoLA	STS-B	MRPC
Full-Prec	Budget-O	74.00	57.78	88.74	87.35		74.72	55.37	89.27	86.11
TernaryBERT	Budget-A	72.02 ±0.21	53.44 ±1.11	88.43 ±0.07	88.14 ±0.31	XTC	69.91 ±0.41	53.74 ±0.77	88.77 ±0.03	86.29 ±0.57
TI-G	Budget-A	**72.92** ±0.72	**54.29** ±1.35	**88.45** ±0.29	**88.36** ±0.49	TI-G	**70.87** ±0.20	**56.46** ±0.68	**88.94** ±0.04	**86.98** ±0.44
TernaryBERT	Budget-C	73.40 ±1.30	54.11 ±1.11	**88.60** ±0.02	88.43 ±0.58	XTC	73.76 ±0.54	56.30 ±0.67	88.91 ±0.03	**87.38** ±0.19
TI-G	Budget-C	**73.82** ±0.41	**55.05** ±1.13	**88.60** ±0.01	**88.62** ±0.02	TI-G	**74.48** ±0.79	**56.32** ±1.13	**88.92** ±0.03	87.34 ±0.41

Table 4: Accuracy comparison of QAT methods on compressed BERT family (few-sample GLUE tasks) without and with TI for prolonged fine-tuning iterations (Budget-A/C). Each experiment is repeated 10 times.

Iterations	Short (1K)		Regular (10K/20K)	
Dataset	CIFAR100	CIFAR10	CIFAR100	ImageNet
Full-Prec	92.78	99.1	92.78	82.65
TernaryBERT	84.61 ±0.12	97.32 ±0.02	89.57 ±0.04	75.40 ±0.12
TI-G	**85.28** ±0.04	**97.59** ±0.05	**90.07** ±0.04	**76.66** ±0.04

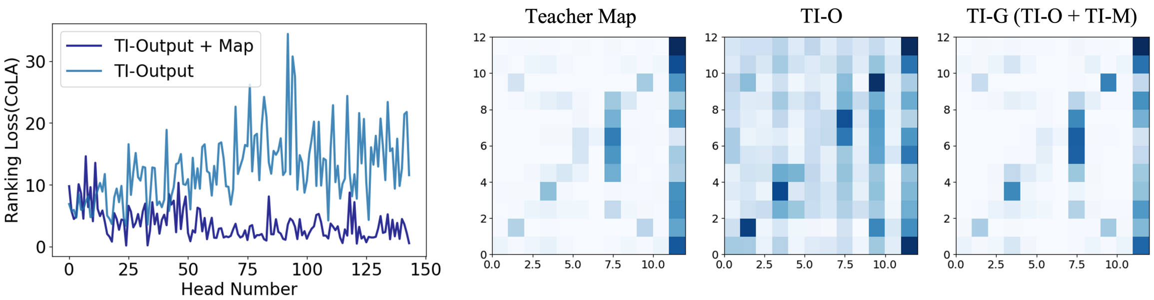
Table 5: Accuracy comparison of QAT methods on ViT-B for vision benchmarks.

5.3 Convergence Analysis

In Sec. 4, we discussed that TI impedes propagation of quantization errors and flattens the loss surface for favorable convergence as shown in Fig. 2. To quantitatively analyze the convergence of QAT, we conducted the Hessian eigenvalue analysis (Park and Kim, 2022) on BERT$_{base}$ over CoLA task. As shown in Fig. 5, TernaryBERT suffers large magnitude (positive and negative) eigenvalues. The large magnitude Hessian eigenvalues indicate sharp loss surfaces, unfriendly for quantization (Shen et al., 2020; Dong et al., 2019). A prolonged fine-tuning would suppress the large Hessian eigenvalues at the expense of significantly increased fine-tuning computations. In contrast, TI significantly reduces the magnitude with an order of magnitude smaller fine-tuning iteration. From this analysis, we can conclude that TI helps flatten the loss surface to improve the convergence of QAT.

5.4 Further Analysis of TI-G: Self-Attention Map Recovery

In this section, we provide an analysis of TI-M's effectiveness in the use of TI-G. Since the self-attention map encodes the relational representation of symbols within a sequence, quantization on SA-GEN would disrupt this relational encoding. Therefore, TI on the self-attention map (TI-M) is crucial for maintaining relative order within the self-attention map. As shown in Sec. 4.2, TI-O lacks consideration of the self-attention map recovery, requiring gradual teacher intervention mechanism



(a) (b)

Figure 6: (a) Comparison of ranking loss between TI-O and TI-O with TI-M (TI-Output + Map) (b) Visualization of self-attention map with full-precision, TI-O and TI-G trained on CoLA. The input sentence is "John was lots more pleasure than Fred."

that applies TI-M after TI-O for recovering the self-attention map. Fig. 6 shows two empirical evidences that TI-M helps to recover the self-attention map with TI-O.

Fig. 6(a) shows the ranking loss computed on the self-attention map of BERT$_{base}$ after QAT on CoLA. Note that the ranking loss quantitatively indicates how much the relative order in the quantized self-attention map has been skewed from the teacher's self-attention map (Liu et al., 2021b). As can be seen, QAT without TI-M leads to significant disruption of the self-attention map. In contrast, QAT with TI-M maintains the original ranking of cross-symbol correlation, successfully suppressing the ranking loss. We further visualize each quantized model's resulting self-attention maps in Fig. 6(b). Contrary to using TI-O only, TI-O with TI-M successfully recovers the teacher self-attention map's distinctive feature. From this observation, we can conclude that TI-M approach used with TI-O is particularly beneficial for preserving the self-attention map under quantization.

6 Conclusion

In this work, we proposed a proactive knowledge distillation method for improving the convergence of QAT for ultra-low precision Transformers called teacher intervention. The proposed method intervenes in the propagation of quantization error to suppress accuracy degradation and improve QAT's convergence speed. We demonstrate that the proposed method outperforms the state-of-the-art QAT methods in achieving higher accuracy on various fine-tuned Transformers with smaller fine-tuning budgets. Our code is available at https://github.com/MarsJacobs/ti-kd-qat.

7 Limitations

This work proposes a proactive knowledge distillation method called *teacher intervention* for improving the convergence of QAT with small fine-tuning budgets. Although our analysis reveals that quantization error propagation is one of the main causes of sub-optimal convergence of QAT, a more in-depth investigation of the detailed recovery mechanism of QAT from quantization error would be interesting. Also, extending the discovery of proactive knowledge distillation to diverse Transformer architectures, including encoder-decoder and decoder-only models, would be a promising future research direction.

Acknowledgements

This work was partly supported by Samsung Advanced Institute of Technology, Samsung Electronics Co., Ltd, Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government (MSIT) (2020-0-01297, Development of Ultra-Low Power Deep Learning Processor Technology using Advanced Data Reuse for Edge Applications) and National Research Foundation of Korea (NRF) grant funded by Korea government (MSIT) (No. 2021R1A2C1013513).

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge distillation from internal representations. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY,* *USA, February 7-12, 2020*, pages 7350–7357. AAAI Press.

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2021. BinaryBERT: Pushing the limit of BERT quantization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4334–4348, Online. Association for Computational Linguistics.

Maximiliana Behnke and Kenneth Heafield. 2020. Losing heads in the lottery: Pruning transformer attention in neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2664–2674, Online. Association for Computational Linguistics.

Aishwarya Bhandare, Vamsi Sripathi, Deepthi Karkada, Vivek Menon, Sun Choi, Kushal Datta, and Vikram Saletore. 2019. Efficient 8-bit quantization of transformer neural machine language translation model. *arXiv preprint arXiv:1906.00532*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

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

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2019. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 293–302.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Mitchell Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the effects of weight pruning on transfer learning. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 143–155, Online. Association for Computational Linguistics.

Daniel Grießhaber, Johannes Maucher, and Ngoc Thang Vu. 2020. Fine-tuning BERT for low-resource natural language understanding via active learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1158–1171, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2016. Binarized neural networks. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Kyuyeon Hwang and Wonyong Sung. 2014. Fixed-point feedforward deep neural network design using weights +1, 0, and -1. In *2014 IEEE Workshop on Signal Processing Systems (SiPS)*, pages 1–6.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2021. I-bert: Integer-only bert quantization. *International Conference on Machine Learning (Accepted)*.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021a. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022.

Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. 2021b. Post-training quantization for vision transformer. In *Advances in Neural Information Processing Systems*.

Yihuan Mao, Yujing Wang, Chufan Wu, Chen Zhang, Yang Wang, Quanlu Zhang, Yaming Yang, Yunhai Tong, and Jing Bai. 2020. LadaBERT: Lightweight adaptation of BERT through hybrid model compression. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3225–3234, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

Namuk Park and Songkuk Kim. 2022. How do vision transformers work? In *International Conference on Learning Representations*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8815–8821.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Xiaoxia Wu, Zhewei Yao, Minjia Zhang, Conglong Li, and Yuxiong He. 2022. Extreme compression for pre-trained transformers made simple and efficient. In *Advances in Neural Information Processing Systems*.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2021. Fine-tuning pre-trained language model with weak supervision: A contrastive-regularized self-training approach. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1063–1077.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, pages 36–39.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020a. Revisiting few-sample bert fine-tuning. In *International Conference on Learning Representations*.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020b. TernaryBERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, Online. Association for Computational Linguistics.

Chenzhuo Zhu, Song Han, Huizi Mao, and William J Dally. 2016. Trained ternary quantization. *arXiv preprint arXiv:1612.01064*.

A Appendix

In this section, we provide experimental setup and TI implementation in Sec. A.1. We also provide evaluation results of GLUE (Wang et al., 2018) large datasets (SST-2, QNLI, MNLI, QQP) in Sec. A.2 and ablation studies in Sec. A.3. Further analysis of TI-G's effectiveness is provided in 5.4.

A.1 Experimental Settings

Models Description We use task-specific, fine-tuned BERT family models to evaluate our method. Further information on each model is as follows:

1. **BERT$_{base}$** L=12, d=768 N_H=12, Contains about 110M parameters.
2. **BERT$_{large}$** L=24, d=1024 N_H=16, Contains about 340M parameters.
3. **TinyBERT-4L** L=4, d=312 N_H=12, Contains about 14M parameters. This compressed model is fine-tuned using task-specific distillation method (Jiao et al., 2020) initialized from general pre-trained TinyBERT model file. [2]
4. **TinyBERT-6L** L=6, d=768 N_H=12, Contains about 67M parameters. This compressed model is fine-tuned using task-specific distillation method (Jiao et al., 2020) initialized from general pre-trained TinyBERT model file. [3]
5. **SkipBERT-6L** L=6, d=768 N_H=12, Contains about 67M parameters. This compressed model parameters are initialized from XTC (Wu et al., 2022) model initialization method: using every other layer of the fine-tuned BERT$_{base}$'s parameter to initialize the layer-reduced model.

Datasets and Settings We evaluate our method on GLUE benchmark (Wang et al., 2018), which is a collection of resources for training, evaluating, and analyzing natural language understanding systems. For the training setting, we use the batch size of 16 for CoLA and 32 for other tasks. The learning rate starts from zero and gradually increases to 2e-5 during the warm-up stage and decays linearly to 2e-9 for total training epochs.

	(Zhang et al., 2020b)			(Wu et al., 2022)	
	Budget-O	Budget-O2	Budget-O4	Budget-A	Budget-C
DA	✗	✗	✗	✓	✓
Epoch	3	6	12	1	12

Table 6: Training budgets for the GLUE tasks. DA is data augmentation.

	CoLA	RTE	MRPC	STS-B
Train (noDA)	8,551	2,490	3,668	5,749
Train (DA)	213,212	142,991	225,630	322,121
DA / noDA	24.9	57.4	61.5	56.0
Budget-O Time	259	97	72	223
Budget-A Time	2,149	1,862	1,484	4,160
Budget-C Time	25,792	22,342	17,813	49,915

Table 7: Training details for the GLUE tasks. DA is data augmentation.

Comparison of Training Cost We performed an extensive performance comparison in Sec. 5.1 along the training budget to investigate the convergence of QAT methods. Table. 6 shows the training setting of each budget with the Data Augmentation (DA) option, which is first proposed by (Zhang et al., 2020b) (Jiao et al., 2020) artificially expanding the training sample for prolonged fine-tuning, causing a blow-up in fine-tuning costs. To investigate how much the DA magnifies the QAT overhead, we compare the number of training samples of GLUE small datasets and the number of data-augmented training samples in Table. 7. Furthermore, we measured the total QAT time for each Budget over GLUE small datasets with the A6000 single GPU, indicating that prolonged fine-tuning with DA would become profoundly painful, increasing model deployment costs and time.

Teacher Intervention Implementation Our experiments were performed on A6000 GPUs. We use Pytorch 1.10.2 for the implementation of TI and this implementation is based on the TernaryBERT Pytorch code base. [4] We would like to emphasize that TI has strong advantages of simplicity for implementation. To implement TI operation, we modified the teacher model forward operation, enforcing the teacher model to return layer-wise MHA outputs and SA-GEN outputs which will be used in TI-O and TI-M respectively. When the teacher model's layer-wise required outputs are returned,

---

[2] https://huggingface.co/huawei-noah/TinyBERT_4L_zh

[3] https://huggingface.co/huawei-noah/TinyBERT_6L_zh

[4] https://github.com/huawei-noah/Pretrained-Language-Model/tree/master/TernaryBERT

		BERT$_{base}$				BERT$_{large}$			
Method	Cost	SST-2	QNLI	MNLI	QQP	SST-2	QNLI	MNLI	QQP
	Full-Prec	93.57	91.32	84.59	89.34	93.57	92.29	86.49	89.55
TernaryBERT	Budget-O	91.82 ±0.21	90.68 ±1.11	83.74 ±0.07	89.09 ±0.31	91.62 ±0.41	91.91 ±0.77	85.52 ±0.03	89.26 ±0.57
TI-Gradual	Budget-O	**92.16** ±0.72	**90.91** ±1.35	**84.13** ±0.29	**89.11** ±0.49	**92.12** ±0.20	**92.11** ±0.68	**86.11** ±0.04	**89.46** ±0.44
		TinyBERT-4L				TinyBERT-6L			
Method	Cost	SST-2	QNLI	MNLI	QQP	SST-2	QNLI	MNLI	QQP
	Full-Prec	92.86	91.32	84.59	89.34	93.57	92.29	86.49	89.55
TernaryBERT	Budget-O	91.43 ±0.72	84.18 ±1.35	80.93 ±0.29	85.02 ±0.49	91.12 ±0.20	89.71 ±0.68	83.51 ±0.04	89.43 ±0.44
TI-Gradual	Budget-O	**91.45** ±0.72	**84.51** ±1.35	**81.06** ±0.29	**85.09** ±0.49	**91.43** ±0.20	**89.81** ±0.68	**83.64** ±0.04	**89.44** ±0.44

Table 8: Accuracy comparison of QAT methods on BERT family (large-sample GLUE tasks) without and with TI for regular fine-tuning iterations (Budget-O). Each experiment is repeated 3 times.

		BERT$_{base}$				BERT$_{large}$			
Method	Cost	RTE	CoLA	STS-B	MRPC	RTE	CoLA	STS-B	MRPC
	Full-Prec	73.28	58.04	89.24	87.77	70.39	60.31	89.83	88.43
TernaryBERT	Budget-A	73.16 ±0.54	54.81 ±0.39	88.92 ±0.02	**87.77** ±0.32	70.39 ±0.21	58.65 ±0.31	89.77 ±0.01	**88.91** ±0.59
TI-G	Budget-A	**74.24** ±1.36	**58.29** ±0.50	**89.01** ±0.01	87.74 ±0.96	**71.48** ±0.95	**59.24** ±1.38	**89.89** ±0.07	88.52 ±0.22
TernaryBERT	Budget-C	73.40 ±0.41	58.25 ±0.84	89.23 ±0.01	88.43 ±0.31	71.48 ±0.06	59.11 ±1.47	89.85 ±0.01	**89.08** ±0.55
TI-G	Budget-C	**75.93** ±0.90	**58.91** ±0.54	**89.39** ±0.16	**87.85** ±0.26	**72.68** ±1.50	**59.17** ±0.61	**90.10** ±0.01	88.74 ±0.41

Table 9: Accuracy comparison of QAT methods on BERT family (few-sample GLUE tasks) without and with TI for prolonged fine-tuning iterations (Budget-A/C). Each experiment is repeated 10 times.

	BERT$_{base}$	BERT$_{large}$	TinyBERT-4L	TinyBERT-6L
Baseline	72.51	73.29	65.88	72.17
TI - output loss	73.92	73.79	66.85	72.77
TI - two step	73.95	74.26	66.32	72.59
TI	**74.13**	**74.46**	**67.13**	**73.02**

Table 10: Ablation study of Teacher Intervention. Each result is averaged score over GLUE small datasets (RTE, CoLA, STS-B, MRPC)

	BERT$_{base}$	BERT$_{large}$	TinyBERT-4L	TinyBERT-6L
Inverted TI	74.27	74.37	68.28	72.98
Stochastic TI	74.32	**74.57**	68.17	72.93
TI-G	**74.37**	74.41	**68.32**	**73.06**

Table 11: Comparison of unified TI strategies. Each result is averaged score over GLUE small datasets (RTE, CoLA, STS-B, MRPC)

we passed these to the student model forward operation's input. With these additional inputs, we replace the student's attention map/output with the teacher's in the attention sub-layer forward operation. More detailed implementation can be found at https://github.com/MarsJacobs/ti-kd-qat.

A.2 Additional Experimental Results

Large-Sample Fine-tuning Table. 8 shows evaluation of TI in GLUE large datasets (SST-2, QNLI, MNLI, QQP). TI-G outperforms the baseline for all the cases as we discussed in Table. 3, showing that TI improve convergence of QAT in large sample fine-tuning.

Additional Results of Prolonged Fine-Tuning Table. 9 shows additional evaluation results of TI in prolonged fine-tuning (BERT$_{base}$ and BERT$_{large}$ with Budget-A/C). As can be seen, TI-G shows superior accuracy to TernaryBERT with a noticeable margin in Budget-A/C both. Furthermore, TI-G achieves close to full-precision accuracy with Budget-A, showing TI's beneficial effect on the convergence of QAT.

A.3 Ablation Study

TI two step QAT and output loss In this section, we delve deeper into evaluating the efficacy of the TI method. As TI-O employs one additive loss for internal distillation (Attention Output MSE loss, called output loss) and TI QAT is performed in two steps, conducting an ablation study on TI (exploiting output loss and two-step QAT separately) provides a deeper understanding of our method. Table. 10 demonstrates the effect of output loss and the use of a two-step method with TI independently with BERT family models over GLUE small datasets. As can be seen, both factors act positively on the performance of quantized models. Interestingly, two-step QAT factor provides a powerful impact (Row 2) in compressed models (TinyBERT-4L and TinyBERT-6L), and output loss shows a distinct impact (Row 3) in the BERT$_{large}$ model. When both factors are implied together (Row 4), performance is boosted further.

Exploration of unified teacher intervention methods In this section, we investigate the best unified teacher intervention strategy. We proposed a unified intervention approach (TI-G) in Sec. 4.2, applying TI with output intervention (TI-O) followed by map intervention (TI-M). As more scheduling choices are left when using TI-O and TI-M together, we propose additional two scheduling methods, Inverted TI, and Stochastic TI. Inverted TI applies TI-O and TI-M in the opposite order of TI-G, and stochastic TI means choosing TI-O and TI-M options randomly in every training iteration, ensembling each TI option's effect in a stochastic way. Table. 11 shows the comparison between unified TI scheduling methods with BERT family models over GLUE small datasets. TI-G outperforms other unified TI scheduling methods in almost every model, which shows our proposed teacher intervention unification approach's efficacy.

Generative Replay Inspired by Hippocampal Memory Indexing for Continual Language Learning

Aru Maekawa[1], **Hidetaka Kamigaito**[1,2], **Kotaro Funakoshi**[1], **Manabu Okumura**[1]
[1]Tokyo Institute of Technology, [2]Nara Institute of Science and Technology
`{maekawa, kamigaito, funakoshi, oku}@lr.pi.titech.ac.jp`
`kamigaito.h@is.naist.jp`

Abstract

Continual learning aims to accumulate knowledge to solve new tasks without catastrophic forgetting for previously learned tasks. Research on continual learning has led to the development of generative replay, which prevents catastrophic forgetting by generating pseudo-samples for previous tasks and learning them together with new tasks. Inspired by the biological brain, we propose the hippocampal memory indexing to enhance the generative replay by controlling sample generation using compressed features of previous training samples. It enables the generation of a specific training sample from previous tasks, thus improving the balance and quality of generated replay samples. Experimental results indicate that our method effectively controls the sample generation and consistently outperforms the performance of current generative replay methods.[1]

1 Introduction

Humans and intelligent animals continually acquire new knowledge and skills throughout their lifetime. This ability, called continual learning (CL) or lifelong learning, is a fundamental requirement for human-like general intelligence (Parisi et al., 2019). CL is also crucial for practical applications, as new data and tasks to train models appear every day in the real world. It is especially important for natural language processing (NLP) systems, in which vocabulary and language usage change over time. However, most neural network based models are trained with a static dataset. When learning different tasks sequentially, performance on the previously learned tasks tends to significantly degrade, referred to as catastrophic forgetting (McCloskey and Cohen, 1989). Learning new tasks without catastrophic forgetting has been a long-standing challenge in machine learning and neural networks.

Replay is an approach to alleviate catastrophic forgetting by retraining with previous tasks' data when training a new task. Although replay-based methods are effective in most CL scenarios, it is necessary to retain training data for all previous tasks, which may cause problems with storage requirements and data privacy. Therefore, generative replay was developed (Shin et al., 2017), which uses pseudo-samples generated from generation models instead of real samples. In the NLP domain, LAMOL (Sun et al., 2020a) was proposed as a generative replay framework, where a single language model simultaneously learns to solve tasks and to generate pseudo-samples (Fig. 1a).

Although the generative replay does not require any previous task's data, it typically underperforms actual sample replay because of the balance and quality problems in sample generation. Regarding the balance problem, generation models tend to generate a large number of samples for recently learned tasks, that results in forgetting older tasks. This is due to the difficulty in controlling sample generation and catastrophic forgetting occurring in generation models. Regarding the quality problem, generative replay methods commonly assume to generate even unseen samples which is not included in the past training datasets. However, since generating such unseen samples is more difficult than generating previously trained samples, it may cause the degradation of the quality of replay samples. To prevent the catastrophic forgetting, it is sufficient to generate only previously learned samples.

To address these issues, we refer to the memory retrieval mechanism in the biological brain that achieves CL. According to the hippocampal indexing theory (Teyler and Rudy, 2007), the hippocampus encodes memory engrams for new neocortical activity patterns and uses them as memory indexes to recall past experiences. Inspired by this hippocampal mechanism, we propose hippocam-

---

[1] The source code is available at `https://github.com/arumaekawa/GR-HMI`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 930–942
May 2-6, 2023 ©2023 Association for Computational Linguistics

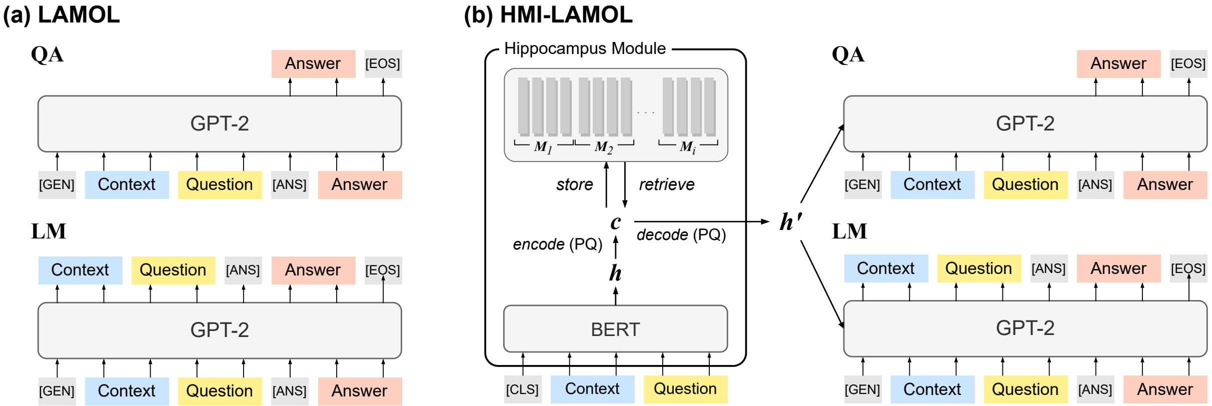


Figure 1: **(a)** is the overview of the LAMOL framework. The top is the learning QA to solve tasks and the bottom is the learning LM to generate pseudo-samples. **(b)** is the proposed HMI implemented on LAMOL. We introduce a hippocampus module illustrated on the left.

pal memory indexing (HMI) for improving generative reply. To remember training samples with a small data usage, we introduce a hippocampus module that encodes training samples into compressed memory engrams using BERT (Devlin et al., 2019) and product quantization (PQ) (Jégou et al., 2011), and stores them to generate conditioned samples during the replay step (Fig. 1b). This method makes it possible to generate specific training samples from previously learned tasks.

We evaluated HMI on two different CL scenarios using the original LAMOL as a baseline. The first scenario is a sequence of different types of tasks, for which we used five natural language understanding (NLU) tasks from DecaNLP (McCann et al., 2018). The other scenario is a sequence of different domains in the same task, for which we used five text classification datasets and single-pass setting, which is considered as an ideal scenario for CL. The results indicate that HMI consistently outperforms LAMOL and improves robustness to training task order and amount of replay samples. We also investigated the balance of previously learned tasks in generated samples and found that HMI enables the generation of even old task samples, which indicates the controllability of sample generation with HMI. Furthermore, we explored the potential of further improvement of HMI with different sample selection strategies for replay.

2 Related Work

CL, which involves learning from a stream of tasks without catastrophic forgetting, is a long-standing issue in machine learning. In the NLP, CL has been studied for diverse tasks, for example, word and sentence representations (Xu et al., 2018; Liu et al., 2019), sentiment analysis (Chen et al., 2015; Xia et al., 2017), composition language learning (Li et al., 2020b), relation learning (Han et al., 2020), dialogue systems (Lee, 2017; Madotto et al., 2021), text classification, and question-answering (QA) (de Masson d'Autume et al., 2019; Wang et al., 2020).

Regularization-based methods aim to constrain changes in model parameters important for previous tasks. Various methods have been proposed to estimate the importance of each parameter. For example, elastic weight consolidation (EWC) (Kirkpatrick et al., 2017) uses the Fisher information matrix. Synaptic intelligence (SI) (Zenke et al., 2017) estimates importance from the contribution to loss changes. Memory-aware synapses (MAS) (Aljundi et al., 2018) computes the sensitivity of parameters on the basis of the gradient of model outputs.

Architecture-based methods dynamically change the network structure to assign model parameters for each task. Progressive neural networks (PNN) (Rusu et al., 2016) freeze the current parameters and add a new column of the network when training a new task. Instead of extending the network, PackNet (Mallya and Lazebnik, 2018) applies network pruning using dynamic filters to separate the neurons used for each task.

Replay-based methods mitigate catastrophic forgetting by retraining for previous tasks when training for a new one. MbPA++ (de Masson d'Autume et al., 2019) introduces an episodic memory that stores real samples of previous tasks to use for experience replay and local adaptation. Meta-MbPA

(Wang et al., 2020) applies a meta-learning algorithm to improve MbPA++. To enhance the replay-based methods with a limited amount of samples, Wang et al. (2020) and Huang et al. (2021) also investigated effective selection strategies other than random sampling. Instead of keeping real samples for replay, Shin et al. (2017) proposed generative replay, which trains a model to generate pseudo-samples. Sun et al. (2020a) proposed LAMOL as a generative replay method for NLP tasks. LAMOL uses GPT-2 (Radford et al., 2019) to simultaneously learn a variety of NLP tasks and pseudo-sample generation. L2KD (Chuang et al., 2020) and DnR (Sun et al., 2020b) use knowledge distillation to extend LAMOL. MFK-LAMOL (Choi and Kang, 2021) makes replay more efficient by using more forgotten pseudo-samples in generative replay. Rational-LAMOL (Kanwatchara et al., 2021) uses critical freezing guided by supervised or unsupervised rationale. RVAE-LAMOL (Wang et al., 2022) enhances LAMOL by mapping different tasks into a limited unified feature space.

Current generative replay methods have problems on the balance and quality of generated samples. To address these issues, we propose a sample-generation control with the HMI method, inspired by the biological brain. In contrast to the previous work, our approach prevents low quality samples by using the assumption that a model generates only previously learned samples. HMI also achieves balanced sample generation by strong sample-level conditioning rather than task-level conditioning. Although our HMI can be applied to most of the existing generative replay methods, similar to other recent work, we build HMI upon LAMOL, which is a simple generative replay baseline for CL in NLP and whose implementation code is available.

3 LAMOL: Language Modeling for Lifelong Language Learning

Before describing HMI, we briefly explain LAMOL (Sun et al., 2020a), on which we propose our HMI, in this section.

LAMOL is a generative replay framework using a single GPT-2 to solve different types of NLP tasks and generate pseudo-samples. In LAMOL, all training samples are fed into GPT-2 as a sequence of *context*, *question*, and *answer*. As illustrated at the top of Figure 1a, GPT-2 learns each task in a QA manner, predicting the answer part on the basis of the given context and question. As well as training QA, GPT-2 learns language modeling (LM) to generate the whole sequence of the context, question, and answer, as illustrated at the bottom of Figure 1a. During the training step, the parameters of GPT-2, $\boldsymbol{\theta}_{\text{GPT-2}}$, are optimized to minimize the QA loss $\mathcal{L}_{\text{QA}}$ and the LM loss $\mathcal{L}_{\text{LM}}$ together as $\mathcal{L} = \mathcal{L}_{\text{QA}} + \lambda\mathcal{L}_{\text{LM}}$, where λ is a hyperparameter.

When training for a new task, LAMOL generates pseudo-samples for previous tasks to use for replay. Assume a stream of tasks $\{\mathcal{T}_1, \mathcal{T}_2, \dots \mathcal{T}_T\}$ to train a model with LAMOL, where the number of tasks T may be unknown. Before training a new task $\mathcal{T}_i$ ($i > 1$), GPT-2 generates pseudo-samples by top-k sampling from the first token [GEN]. The number of pseudo-samples is $\gamma|\mathcal{T}_i|$, where γ is the sampling ratio and $|\mathcal{T}_i|$ is the number of training samples in $\mathcal{T}_i$. Defective samples, which do not have a unique [ANS] token that indicates the start position of the answer, are discarded, and the others are mixed with $\mathcal{T}_i$ to alleviate forgetting for $\mathcal{T}_{<i}$ in training.

When using the same [GEN] for all tasks, the ratio for old tasks in the generated samples decreases exponentially in theory (Sun et al., 2020a). Therefore, Sun et al. (2020a) proposed to replace [GEN] with a task-specific token [TASK] (e.g., "`__sst__`") to control GPT-2 to generate pseudo-samples belonging to the specific task. In the beginning of training for $\mathcal{T}_i$, $\frac{\gamma}{i-1}|\mathcal{T}_i|$ pseudo-samples for each previous task, $\mathcal{T}_1, \dots, \mathcal{T}_{i-1}$, are generated using the corresponding task-specific token.

4 Hippocampal Memory Indexing (HMI)

In this section, we introduce our Hippocampal Memory Indexing (HMI) that can suppress the problems of unbalanced and low-quality generation in the replay by accessing compressed features for previous training samples.

4.1 Overview

HMI is implemented as a module that works on LAMOL. Figure 1b shows the overview of our HMI on LAMOL. The training process of HMI on LAMOL is as follows.

1. The **hippocampus module** encodes each training sample into a feature vector representation with a BERT encoder (§ 4.2.1).

2. **Product Quantization** (PQ) compresses the encoded feature as a **memory engram** in a **hippocampal memory** for future replay (§ 4.2.2). To prevent increasing the size of the hippocampal memory, we apply memory pruning at the end of training for each task (§ 4.2.3).

3. The **memory engrams** are used to condition the generation by GPT-2 to generate the corresponding samples (§ 4.3). This allows us to control the task balance in replay samples and generate only previously learned samples of higher quality.

We describe the hippocampus module, which encodes and stores memory engrams for each training sample, in § 4.2 and explain generation control with HMI in § 4.3. Finally, we explain the pre-training for HMI in § 4.4.

4.2 Hippocampus Module

The hippocampus module is a memory component that stores memory engrams, compressed representations of training samples in previously learned tasks. During replay, the module retrieves stored memory engrams and provides them for GPT-2 to generate the corresponding previous samples. In this section, we describe the encoding of a memory engram from each training sample and the pruning of the hippocampal memory in the module.

4.2.1 Memory Engram Encoding

A memory engram is created from the context and question part of each training sample. Each sequence consisting of the context and question is first encoded to a fixed-sized vector representation with the BERT encoder. Similar to BERT, we use a special classification token [CLS] in the beginning of an input sequence to obtain a sample-level embedding $\boldsymbol{h}_{\texttt{[CLS]}} \in \mathbb{R}^H$, where H is the embedding dimension in BERT. It is further converted to a d-dimensional feature vector $\boldsymbol{h} = \boldsymbol{W}_{\mathrm{E}}\boldsymbol{h}_{\texttt{[CLS]}} \in \mathbb{R}^d$, where $\boldsymbol{W}_{\mathrm{E}} \in \mathbb{R}^{d\times H}$ is a weight matrix. Note that, during CL, the encoder parameters $\{\phi_{\mathrm{BERT}}, \boldsymbol{W}_{\mathrm{E}}\}$ are frozen to prevent the features from drifting, which degrades the controllability of HMI.

4.2.2 Feature Quantization

A feature vector $\boldsymbol{h} \in \mathbb{R}^d$ is then quantized to reduce the data size of the hippocampal memory. In the beginning of training a new task, we train a quantization model on encoded feature vectors for all training samples in the new task and all feature vectors currently stored in the hippocampal memory. We use PQ (Jégou et al., 2011), which has a lower reconstruction error than a simple quantization model that uses only k-means. PQ first divides a d-dimensional $\boldsymbol{h}$ into S sub-vectors, each of dimension d/S. It then creates a codebook by computing N centroids for each S partitions with the k-means algorithm. Using this codebook, each $\boldsymbol{h}$ is quantized to S integer indices $\boldsymbol{c} \in \{0, \dots, 255\}^S$ and stored in the hippocampal memory as a memory engram.

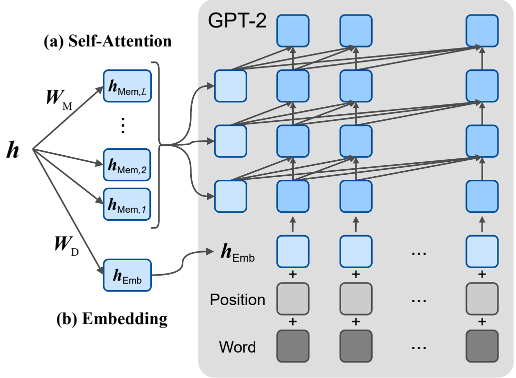


Figure 2: Feature-vector injection to GPT-2. (a) Self-Attention: $\boldsymbol{h}_{\mathrm{Mem},l}$ is fed into the l-th self-attention layer and used as an additional key and value. (b) Embedding: $\boldsymbol{h}_{\mathrm{Emb}}$ is added to word and position embeddings.

4.2.3 Memory Pruning

It is inefficient to keep memory engrams of all training samples in each previously learned task, which may cause a scalability problem with the data size for an increased number of tasks. Thus, we set the maximum number of stored memory engrams M_{max}. When learning the i-th task $\mathcal{T}_i$, the memory engrams of all training samples in $\mathcal{T}_i$ are first appended to the hippocampus module as the i-th task hippocampal memory $\boldsymbol{M}_i$. After updating the codebook of the PQ model, every hippocampal memory $\{\boldsymbol{M}_1, \boldsymbol{M}_2, \dots \boldsymbol{M}_i\}$ is reduced to satisfy $|\boldsymbol{M}_j| \leq \frac{M_{\mathrm{max}}}{i}$ for $1 \leq j \leq i$ by randomly selecting memory engrams to keep from $\boldsymbol{M}_j$. Therefore, we can keep the total number of memory engrams at most M_{max} regardless of the number of tasks.

4.3 Sample Generation with Hippocampal Memory Indexing

To condition the sample generation, we feed the memory engram of each training sample into GPT-2. We use two schemes, each of which is based on embedding and self-attention layers (Li et al., 2020a; Fang et al., 2021), as described as follows and illustrated in Figure 2.

Embedding Layer In GPT-2, the embedding representation of the t-th token in an input sequence is the sum of the word embedding $\boldsymbol{h}_{\mathrm{WE}}^{(t)} \in \mathbb{R}^H$ and position embedding $\boldsymbol{h}_{\mathrm{PE}}^{(t)} \in \mathbb{R}^H$, where H is the embedding dimension in GPT-2. The feature vector $\boldsymbol{h}$ is added to it, so the new embedding vector is $\boldsymbol{h}_{\text{Emb}}^{(t)} = \boldsymbol{h}_{\text{WE}}^{(t)} + \boldsymbol{h}_{\text{PE}}^{(t)} + \boldsymbol{W}_{\text{D}}\boldsymbol{h}$, where $\boldsymbol{W}_{\text{D}} \in \mathbb{R}^{H \times d}$ is a weight matrix.

Self-Attention Layer The feature vector $\boldsymbol{h}$ is used as an additional key and value in each self-attention layer. It is first projected into an LH-dimensional vector with a weight matrix $\boldsymbol{W}_{\text{M}} \in \mathbb{R}^{LH \times d}$. It is then divided into L vectors $[\boldsymbol{h}_{\text{Mem},1}, \ldots, \boldsymbol{h}_{\text{Mem},L}]$ and converted to a key and value in each self-attention layer.

We apply the feature-vector injection to the training, replay, and inference steps. During the training step, a quantized memory engram $\boldsymbol{c}$, encoded from a training sample in the hippocampus module, is decoded back into a feature vector $\boldsymbol{h}'$ and given to GPT-2 to condition the generation for both the QA and LM. It reduces the effect of quantization errors between training and generation to use $\boldsymbol{h}'$ rather than the BERT encoder outputting $\boldsymbol{h}$ directly. During the replay step, when training for the i-th task $\mathcal{T}_i$ $(i > 1)$, we randomly select $\frac{\gamma}{i-1}|\mathcal{T}_i|$ memory engrams from the hippocampal memory for each previous task, $M_1, \ldots, M_{i-1}$, and provide them for GPT-2 to generate the corresponding past training samples by greedy decoding.

In inference, GPT-2 generates the answer part on the basis of the given context and question with injected $\boldsymbol{h}$ encoded from the context and question of a test sample.

4.4 Pre-training

The memory engrams encoded in the hippocampus module should contain enough information to reconstruct the corresponding samples with GPT-2. However, a naive [CLS] token embedding of BERT and a linear projection with randomly initialized $\boldsymbol{W}_{\text{E}}$ are considered insufficient. In addition, the GPT-2 side uses the connections with $\boldsymbol{W}_{\text{D}}$ and $\boldsymbol{W}_{\text{M}}$, which are initialized randomly, so it may make CL unstable, especially for initial tasks.

To address these issues, we introduce a pre-training of the BERT encoder and GPT-2 as an autoencoder (AE) using an unlabeled text corpus. In the pre-training, the BERT encoder learns to encode each input text into a single feature vector, and GPT-2 learns to reconstruct the original input from the feature vector. Note that we do not apply the quantization model to the pre-training. This is because it is not necessary to store feature vectors, since we do not consider the replay of the pre-training data, and this also allows the encoder parameters $\{\phi_{\text{BERT}}, \boldsymbol{W}_{\text{E}}\}$ to be optimized through gradient descent. With this pre-training, the model encodes well-informed memory engrams and achieves their conditional generation with GPT-2 from the beginning of CL.

Along with the AE, GPT-2 simultaneously learns the LM in this pre-training. This is to prevent GPT-2 from losing the knowledge of the LM that GPT-2 originally has by training the AE. Therefore, we optimize the sum of the AE loss and the LM loss $\mathcal{L} = \mathcal{L}_{\text{AE}} + \mathcal{L}_{\text{LM}}$, and obtain the initial model parameters $\{\phi_{\text{BERT}}, \boldsymbol{W}_{\text{E}}, \boldsymbol{\theta}_{\text{GPT-2}}, \boldsymbol{W}_{\text{D}}, \boldsymbol{W}_{\text{M}}\}$ for CL. The effect of the pre-training are given in Appendix A.

5 Experiments

5.1 General Settings

Tasks, Datasets, and Metrics To evaluate the effectiveness of HMI for CL on different types of tasks, we first evaluated it on a scenario of selecting five NLU tasks from decaNLP (McCann et al., 2018), following the settings of Sun et al. (2020a): QA (SQuAD), semantic parsing (WikiSQL), sentiment analysis (SST), semantic role labeling (QA-SRL), and goal-oriented dialogue (WOZ).

To evaluate it on another CL scenario, we used five text classification tasks from diverse domains: news classification (AGNews), sentiment analysis (Yelp, Amazon), Wikipedia article classification (DBPedia), and QA categorization (Yahoo). We followed the settings of de Masson d'Autume et al. (2019) to use the balanced version datasets[2] and applied the single-pass setting, where a model trains with each dataset for only one epoch.

Each dataset has the corresponding evaluation metric ranging from 0 to 100%. More details of tasks and datasets are given in Appendix B.1.

Compared Methods We compared the following methods:

- **LAMOL** This is the baseline generative replay method, without HMI. GPT-2 generates pseudo-samples by top-k sampling with $k = 20$ given only first tokens [GEN]/[TASK].
- **HMI-LAMOL** Our HMI implemented on LAMOL. We evaluated the effectiveness of HMI by comparing it to LAMOL.

---

[2] We used the random sampled subsets released by Sun et al. (2020a).

- **Real Samples** Replay samples in HMI-LAMOL are replaced with actual training samples corresponding to the given memory engrams. This is considered as the upper bound of HMI-LAMOL in terms of the quality of generated samples.
- **Multitask** All tasks are simultaneously trained with GPT-2. Note that GPT-2 optimizes only the QA loss. Since future tasks are not accessible in CL, this is often regarded as the upper bound.

Implementation We implemented the above methods on the basis of the official implementation of LAMOL.[3] We also re-implemented LAMOL to use [GEN]/[TASK] for learning the QA as well as learning the LM to make training fast and improve performance by unifying input format. We used the smallest pre-trained GPT-2[4] as the language model for all methods and the smallest pre-trained BERT[5] for the encoder of the hippocampus module in HMI-LAMOL.

Hyper-parameters For all methods, we followed the settings in Sun et al. (2020a): We set $\lambda = 0.25$ for the weight of the LM loss and applied greedy decoding during inference. We trained the models for nine epochs for each NLU task and for only one epoch for each text classification task. For HMI-LAMOL, we set the maximum size of the hippocampal memory M_{max} to 10,000. We also set the dimension d of the feature vector $\boldsymbol{h}$ to 768, the same as the embedding dimension in BERT and GPT-2. We set the number of partitions $S = 16$ and the number of centroids $N = 256$ for the feature quantization. All the results in our experiments were the average over two runs (seed $= 42, 43$). More details of the experimental settings are given in Appendix C.

For the pre-training in HMI-LAMOL (§ 4.4), we used 1,070,272 text lines from the Wiki-40B (Guo et al., 2020) test set as training data. We trained the model with them for three epochs to obtain the initial model parameters for CL.

5.2 Evaluations

5.2.1 Five Different NLU Tasks

Settings We trained each model on the five NLU tasks in either descending/ascending order in accordance with the number of training samples. We evaluated it with and without task-specific tokens, and with three different replay sample ratios $\gamma \in \{0.01, 0.05, 0.2\}$. We obtained the performance of CL as the average score on the five tasks at the end of training streams.

Replay sample ratio (γ)	0.01	0.05	0.2
Large to small: (SQuAD → WikiSQL → SST → QA-SRL → WOZ)			
LAMOL$_{\text{GEN}}$ (Sun et al., 2020a)$*$	-	69.6	73.1
LAMOL$_{\text{TASK}}$ (Sun et al., 2020a)$*$	-	71.5	74.3
LAMOL$_{\text{GEN}}$	69.1	74.1	75.5
LAMOL$_{\text{TASK}}$	67.9	74.4	76.2
HMI-LAMOL$_{\text{GEN}}$	**72.7**	**75.3**	76.4
HMI-LAMOL$_{\text{TASK}}$	72.6	75.2	**76.6**
HMI-LAMOL$_{\text{GEN}}$ + Real samples	73.9	76.4	78.0
HMI-LAMOL$_{\text{TASK}}$ + Real samples	73.7	76.4	77.6
Small to large: (WOZ → QA-SRL → SST → WikiSQL → SQuAD)			
LAMOL$_{\text{GEN}}$ (Sun et al., 2020a)$*$	-	63.2	73.0
LAMOL$_{\text{TASK}}$ (Sun et al., 2020a)$*$	-	75.3	76.9
LAMOL$_{\text{GEN}}$	57.7	59.3	72.9
LAMOL$_{\text{TASK}}$	58.2	60.5	76.4
HMI-LAMOL$_{\text{GEN}}$	67.1	**76.5**	77.3
HMI-LAMOL$_{\text{TASK}}$	**70.8**	75.5	**77.5**
HMI-LAMOL$_{\text{GEN}}$ + Real samples	75.5	78.0	78.9
HMI-LAMOL$_{\text{TASK}}$ + Real samples	75.0	78.0	79.2
Multitask		77.2	

Table 1: Experimental results on the five NLU tasks in two different orders. $*$ indicates the reported score by Sun et al. (2020a).

Results Table 1 shows the results. The results indicate that HMI-LAMOL outperformed LAMOL in all cases, that is, in both two task orders and all γ. HMI-LAMOL with the best resulting setting, in ascending order and $\gamma = 0.2$, even beat multitask. HMI-LAMOL also reduced the performance gap between the two task orders. Although the performance of LAMOL degraded when the value of γ decreased, HMI-LAMOL mitigated the performance degradation of LAMOL. As expected, even when using the same [GEN] token, HMI-LAMOL typically performed as well as using task-specific tokens because it includes their role as well. However, there is also a performance gap from replacing generated samples with real samples. This indicates that HMI-LAMOL could be further improved by developing a better model for conditional generation.

---

[3] `https://github.com/jojotenya/LAMOL`

[4] `https://huggingface.co/gpt2`

[5] `https://huggingface.co/bert-base-uncased`

γ	Methods	(i)	(ii)	(iii)	(iv)	Avg.	Std.
0.01	LAMOL$_{\text{TASK}}$	42.9	49.6	61.7	62.4	54.2	8.3
	HMI-LAMOL$_{\text{TASK}}$	68.2	68.9	67.6	69.4	68.5	0.7
	+ Real samples	70.2	71.7	70.1	71.0	70.8	0.7
0.05	LAMOL$_{\text{TASK}}$	61.2	66.2	63.8	63.4	63.7	1.8
	HMI-LAMOL$_{\text{TASK}}$	70.9	71.7	71.1	71.1	71.2	0.3
	+ Real samples	72.7	73.2	73.3	73.1	73.1	0.2
0.2	LAMOL$_{\text{TASK}}$	71.0	71.9	68.0	71.4	70.6	1.5
	HMI-LAMOL$_{\text{TASK}}$	72.2	72.6	72.3	71.8	72.2	0.3
	+ Real samples	74.9	73.9	75.6	75.5	75.0	0.7
-	Multitask	72.7					

Table 2: Experimental results on the five text classification tasks. (i)–(iv) denote four random task orders. Avg. and Std. respectively represent average and standard deviation for the four orders.

5.2.2 Five Text Classification Tasks

Settings We applied the single-pass setting, which is considered to make it difficult for HMI-LAMOL to memorize training samples.[6] We tried the three different γ as in the previous experiments (§ 5.2.1). We used the task-specific tokens for all methods except multitask learning. Following previous studies (de Masson d'Autume et al., 2019; Sun et al., 2020a), we report our results for four random task orders. The four orders are shown in Appendix B.2.

Results As shown in Table 2, HMI-LAMOL improved upon LAMOL also in this scenario. For all γ, the p-value of the paired t-test between the results on the four task orders of LAMOL and HMI-LAMOL was smaller than 1%. In particular, the performance gains were larger for smaller γ. Moreover, HMI-LAMOL had smaller standard deviations for the four task orders, which indicates that it is robust to the task training order of CL.

5.2.3 Sample Selection Strategies

Although HMI can control sample generation at the sample level as well as at the task level, in the previous experiments, we randomly selected memory engrams to use for the sample generation for each replay step. In this section, we compare the following three selection strategies with the random selection:

- **Nearest K-means** Inspired by previous studies on real samples replay (Wang et al., 2020; Huang et al., 2021), we compute k-means centroids of all memory engrams in the hippocampal memory for each task, where k is the number of generated replay samples for each task, and choose the nearest memory engram for each centroid. This strategy can be used to include more diverse samples in replay than the random selection.

Methods	NLU tasks		Text classification tasks			
	desc	asc	(i)	(ii)	(iii)	(iv)
Random	72.9	**69.7**	68.7	67.4	**67.4**	68.4
Nearest K-means	**73.1**	69.1	**69.0**	**68.6**	67.0	**69.0**
Loss Difference	71.8	65.5	62.4	66.2	**67.4**	68.0
Low Perplexity	71.8	59.3	67.8	63.7	64.9	68.6

Table 3: Results of HMI-LAMOL with different selection strategies for generating samples.

- **Loss Difference** We select samples with a larger loss difference between before and after training $\mathcal{L}_{\text{before}} - \mathcal{L}_{\text{after}}$. The samples selected through this process are considered more effective for model training.
- **Low Perplexity** To ensure the quality of generated samples, we use memory engrams for the samples with low perplexity of the model after training.

In order to simplify the comparison of the selection strategies, we do not apply the memory pruning (§ 4.2.3) and select samples from all previously learned samples. We also set the small $\gamma = 0.01$, where the difference in the selection strategy is more likely to affect CL performance. We tried all task orders for both CL scenarios with the task-specific tokens.

Results Table 3 shows the results. We first find that the selection strategies clearly affected the final performance of CL. This indicates that the control of the sample generation with HMI-LAMOL also works at the sample level as well as at the task level. This further indicates that HMI-LAMOL has the potential to improve performance when we can use a better selection strategy.

The results indicate that nearest k-means had stably good performance. However, we did not observe a clear advantage compared with random selection. This might be because even random selection can also include a sufficient number of varied samples. In contrast with these strategies focusing on the diversity of generated samples, the other two strategies, which are based on a single measure, such as loss difference or perplexity, did not perform well and lacked stability. After observing generated samples with these strategies, we discovered a serious bias in the generated samples: the selection based on the loss difference included more samples from a single class in the text classifi-

---

[6] Results in the multiple-pass setting are shown in Appendix D.

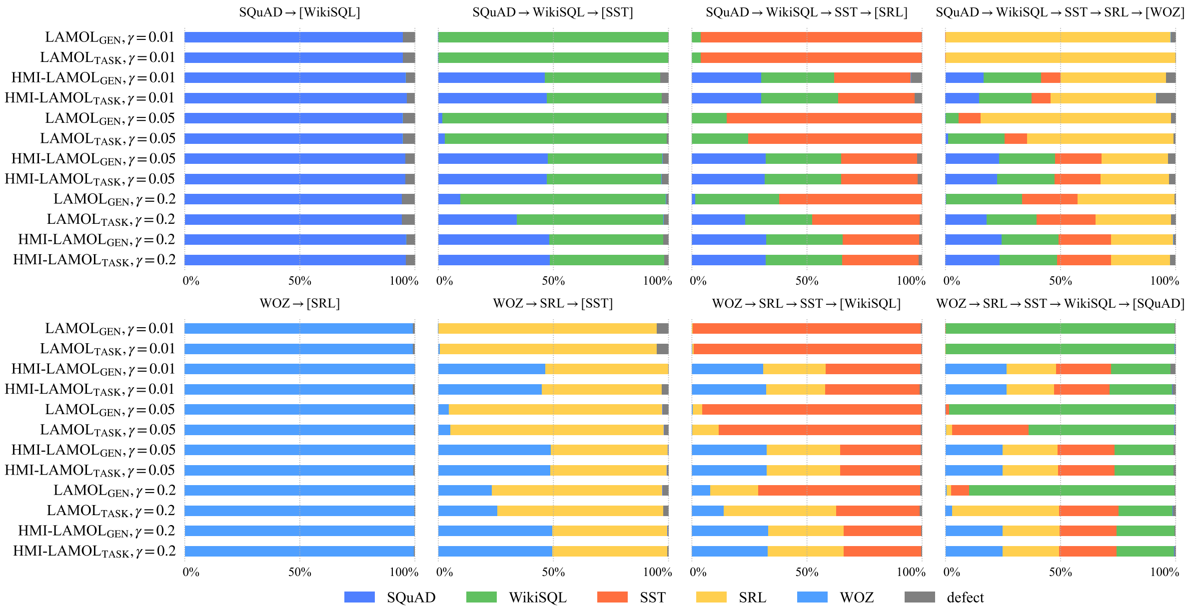


Figure 3: Task balance in replay samples generated with LAMOL and HMI-LAMOL in the experiments of the NLU tasks. Each graph shows the results of generated replay samples when training for each new task denoted with [].

cation task; generated samples selected on the basis of the perplexity tended to be of high quality but short in length and easy to predict. In conclusion, our experiments demonstrated that the diversity of the generated samples contributes to CL performance, which is consistent with the recent findings on real samples replay Wang et al. (2020); Huang et al. (2021).

6 Analysis

6.1 Balance of Replay Samples

To validate the controllability in sample generation in HMI-LAMOL, we investigated the balance of each task in the replay samples generated in the experiments on the five NLU tasks, described in § 5.2.1. To classify the replay samples, we used the BERT classifier model trained with the same training data as the experiments of CL.

Figure 3 shows the portion of each task in the generated replay samples for each replay step in the CL experiments. We first find that LAMOL generated many samples from more recently learned tasks rather than from older tasks. It became more evident in the smaller γ, which is consistent with the performance trend in CL. Although task-specific tokens alleviate this problem to some extent, when $\gamma = 0.01$, almost all of the generated samples were from the most recently learned task in all replay steps. These results indicate that the task-specific tokens are insufficient to tie the generated samples to each task.

	NLU tasks	Text classification tasks
raw text	103,329 KB	340,200 KB
+ gzip	10,879 KB	106,914 KB
no compression	263,553 KB	862,500 KB
+ PQ	3,129 KB	9,368 KB
+ PQ, Pruning	540 KB	540 KB

Table 4: Storage requirements for the hippocampal memory after training the last task in our two CL experiments. *raw text* indicates the size of ASCII text file containing all real samples, and the next line is the size after gzip compression. *no compression* means the case of keeping all samples as real-valued vectors.

However, HMI-LAMOL successfully controls the sample generation for each previous task. In contrast to LAMOL, it enables the generation of samples for older tasks even with the small γ.

6.2 Effect of Memory Compression

HMI uses the feature quantization (§ 4.2.2) and the memory pruning (§ 4.2.3) to reduce the extra storage space required to store memory engrams. Table 4 shows the storage requirements for HMI with and without the two compression methods. It also shows the storage requirements when keeping real samples as raw text file and when applying gzip compression.

When keeping real-valued vectors without the feature quantization, HMI requires even more storage space than when keeping real samples. However, after applying the quantization, each memory engram is compressed to 16 bytes, which is 96x smaller than real-valued vectors and suppressed much less than keeping real samples. Note that the numbers in Table 4 include the PQ codebook, which is 384 KB in size. In addition, the memory pruning reduces it to fixed 540 KB, which the storage requirements will never exceed even if the number of tasks or training samples increase.

7 Conclusion

We proposed hippocampal memory indexing (HMI), inspired by the biological brain, that controls generative replay by conditioning sample generation with compressed representations of previous training samples. Experimental results indicated that HMI successfully controls the sample generation of generative replay and consistently improves the CL performance and robustness. HMI is expected to be further improved by exploring better selection strategies for generating samples.

Limitations

First, in contrast to most existing generative replay approaches, HMI needs extra data space to store features of previous training samples for the sample generation control, while these features are compressed to quantization indices, which require smaller storage space, and their total number is limited to at most $M_{\max}$ by the memory pruning.

Second, there is still a performance gap between the replay of generated samples and real samples in HMI. This indicates that it is difficult to completely reconstruct previously learned samples from memory engrams.

Third, although HMI can control the generated samples for replay, there is room for further investigation into the selection strategies better than the random selection. In addition, we tried only random selection and did not further investigate the selection methods for samples in the memory pruning, which can be also explored in future work.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154.

Zhiyuan Chen, Nianzu Ma, and Bing Liu. 2015. Lifelong learning for sentiment classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 750–756, Beijing, China. Association for Computational Linguistics.

Heejeong Choi and Pilsung Kang. 2021. Lifelong language learning with the most forgotten knowledge. *IEEE Access*, 9:57941–57948.

Yung-Sung Chuang, Shang-Yu Su, and Yun-Nung Chen. 2020. Lifelong language knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2914–2924, Online. Association for Computational Linguistics.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems*, volume 32, pages 13122–13131. Curran Associates, Inc.

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

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*.

Mandy Guo, Zihang Dai, Denny Vrandečić, and Rami Al-Rfou. 2020. Wiki-40B: Multilingual language model dataset. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2440–2452, Marseille, France. European Language Resources Association.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6440, Online. Association for Computational Linguistics.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. 2021. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2736–2746, Online. Association for Computational Linguistics.

Herve Jégou, Matthijs Douze, and Cordelia Schmid. 2011. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128.

Kasidis Kanwatchara, Thanapapas Horsuwan, Piyawat Lertvittayakumjorn, Boonserm Kijsirikul, and Peerapon Vateekul. 2021. Rational LAMOL: A rationale-based lifelong learning framework. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2942–2953, Online. Association for Computational Linguistics.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Sungjin Lee. 2017. Toward continual learning for conversational agents. *CoRR*, abs/1712.09943.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020a. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699, Online. Association for Computational Linguistics.

Yuanpeng Li, Liang Zhao, Kenneth Church, and Mohamed Elhoseiny. 2020b. Compositional language continual learning. In *International Conference on Learning Representations*.

Tianlin Liu, Lyle Ungar, and João Sedoc. 2019. Continual learning for sentence representations using conceptors. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3274–3279, Minneapolis, Minnesota. Association for Computational Linguistics.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7452–7467, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Arun Mallya and Svetlana Lazebnik. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7765–7773.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, volume 30, pages 2990–2999. Curran Associates, Inc.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020a. Lamol: Language moodeling is for lifelong language learning. In *International Conference on Learning Representations*.

Jingyuan Sun, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2020b. Distill and replay for continual language learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3569–3579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Timothy J Teyler and Jerry W Rudy. 2007. The hippocampal indexing theory and episodic memory: updating the index. *Hippocampus*, 17(12):1158–1169.

Han Wang, Ruiliu Fu, Xuejun Zhang, and Jun Zhou. 2022. Rvae-lamol: Residual variational autoencoder to enhance lifelong language learning. *arXiv preprint arXiv:2205.10857*, abs/2205.10857.

Zirui Wang, Sanket Vaibhav Mehta, Barnabas Poczos, and Jaime Carbonell. 2020. Efficient meta lifelong-learning with limited memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 535–548, Online. Association for Computational Linguistics.

Rui Xia, Jie Jiang, and Huihui He. 2017. Distantly supervised lifelong learning for large-scale social media sentiment analysis. *IEEE Transactions on Affective Computing*, 8(4):480–491.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Lifelong domain word embedding via meta-learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4510–4516. International Joint Conferences on Artificial Intelligence Organization.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995. PMLR.

Methods	Pre-training	NLU tasks		Text classification tasks			
		desc	asc	(i)	(ii)	(iii)	(iv)
LAMOL	–	76.2	76.4	71.0	71.9	68.0	71.4
	$\mathcal{L}_{\text{LM}}$	75.8	76.5	70.0	72.4	71.5	71.7
HMI-LAMOL	$\mathcal{L}_{\text{AE}}$	76.2	77.2	71.8	72.3	72.2	71.4
	$\mathcal{L}_{\text{AE}}$ + $\mathcal{L}_{\text{LM}}$	76.6	77.5	72.2	72.6	72.3	71.8

Table 5: Results of CL experiments with $\gamma = 0.2$ and using task-specific tokens for the different pre-training tasks.

A Effect of Pre-training

As described in § 4.4, we applied the LM loss as well as the AE loss to the pre-training in HMI. Table 5 shows the CL performance of LAMOL and HMI-LAMOL for the different pre-training tasks.

The results show that HMI-LAMOL pre-trained with LM, outperformed HMI-LAMOL, pre-trained only with AE, for all task settings, which indicates pre-training LM is effective in HMI-LAMOL. Whereas, it is difficult to say that the pre-training LM is also effective in LAMOL, and the performance of LAMOL, pre-trained with LM, was still lower than HMI-LAMOL. Therefore, the gain of HMI is not only due to the pre-training.

B Tasks and Orderings

B.1 Details of Datasets

Following previous studies, we used five NLU tasks and five text classification tasks for our experiments. Table 6 contains a summary of the datasets, their size, metrics, and examples of the SQuAD-like QA scheme in decaNLP (McCann et al., 2018).

B.2 Ordering for Text Classification Tasks

For text classification tasks, we consider four random permutation orders, which mirror those in a previous study (de Masson d'Autume et al., 2019):
(i) Yelp→AGNews→DBPedia→Amazon→Yahoo,
(ii) DBPedia→Yahoo→AGNews→Amazon→Yelp,
(iii) Yelp→Yahoo→Amazon→DBpedia→AGNews,
(iv) AGNews→Yelp→Amazon→Yahoo→DBpedia.

C Details of Experiment Settings

We used a single Tesla V100 for all experiments. We implemented all methods with half-precision number (16-bit float). All experiments were averaged over two runs with seed $= 42, 43$. A summary of the training hyperparameter settings, following the settings in the official implementation of LAMOL,[3] are listed in Table 7. More details of the implementation can be found in our released experimental source code.[1]

D Five Text Classification Tasks with Multiple-pass Setting

In our experiments with the five text classification tasks (§ 5.2.2), we evaluated our HMI in the single-pass setting, which is considered as an ideal setting for CL and where HMI is more difficult to memorize training samples. In this section, we also present the performance of HMI when training for each task for nine epochs, the same setting as in previous work.

Table 8 shows the results of LAMOL and HMI-LAMOL, which use task-specific tokens and $\gamma = 0.2$, and other current methods. The results indicate that HMI-LAMOL also consistently outperformed LAMOL in this setting and narrows the gap with the replay of real samples. Note that although Meta-MbPA also has a good performance, MbPA++ and Meta-MbPA cannot be directly compared to other methods because of using real samples.

Task	Dataset	Context	Question	Answer	# of Train	# of Test	Metric
Question Answering	SQuAD	Albert Einstein lived in a flat at the Kramgasse 49, ...	Where is Albert Einstein live?	The Kramgasse 49	87,599	10,570	nF1
Semantic Parsing	WikiSQL	The table has columns club, ... Which club was founded ... ?	What is the translation from English to SQL?	SELECT player from table WHERE ...	56,355	15,878	lfEM
Sentiment Analysis	SST	It's a very valuable film...	Is this review negative or positive?	positive	6,920	1,821	EM
Semantic Role Labeling	QA-SRL	The trilogy was released on vinyl by ipecac recordings.	What was released on something?	The trilogy	6,414	2,201	nF1
Goal-oriented Dialogue	WOZ	I am looking for African food	What is the change in state?	food: African;	2,536	1,646	dsEM
Text Classification	AGNews	Smart phone market growing ...	Is this sentence World, ... ?	Sci/Tech	115,000	7,600	EM
	Yelp	Nothing special, your typical buffet food. ...	Is this sentence very negative, ... ?	negative			
	Amazon	One of the worst comercials ...	Is this sentence very negative, ... ?	very negative			
	DBPedia	Rubyville Elementary School ...	Is this sentence Company, ... ?	EducationalInstitution			
	Yahoo	What should I do, I cant quit smoking? ...	Is this sentence Society & Culture, ... ?	Health			

Table 6: Summary of datasets, size, metrics, and example of conversion to dacaNLP format of all tasks.

Hyperparameter	Value
optimizer	Adam
epsilon of Adam	1.0×10^{-4}
learning rate	6.25×10^{-5}
learning rate schedule	warm-up linear
warm-up ratio	0.005
weight decay	0.01
max gradient norm	1.0

Table 7: Training hyperparameters in our experiments.

Methods	(i)	(ii)	(iii)	(iv)	Avg.
MbPA++ (de Masson d'Autume et al., 2019)	70.8	70.9	70.2	70.7	70.6
MbPA++ (Wang et al., 2020)	75.3	74.6	75.6	75.5	75.3
Meta-MbPA (Wang et al., 2020)	77.9	76.7	77.3	77.6	77.3
LAMOL (Sun et al., 2020a)	76.7	77.2	76.1	76.1	76.5
DnR (Sun et al., 2020b)	77.4	77.2	77.1	76.9	77.2
LAMOL	76.6	76.8	76.8	76.8	76.8
HMI-LAMOL	77.5	77.5	77.8	77.3	77.5
HMI-LAMOL + Real samples	77.5	77.5	78.0	78.0	77.7

Table 8: Results on five text classification tasks in the multiple-pass setting. LAMOL and HMI-LAMOL were evaluated with our implementation (bottom three rows), and the other scores are obtained from each paper.

A Survey of Multi-task Learning in Natural Language Processing: Regarding Task Relatedness and Training Methods

Zhihan Zhang, Wenhao Yu, Mengxia Yu, Zhichun Guo, Meng Jiang
University of Notre Dame, Notre Dame, IN, USA
`{zzhang23, wyu1, myu2, zguo5, mjiang2}@nd.edu`

Abstract

Multi-task learning (MTL) has become increasingly popular in natural language processing (NLP) because it improves the performance of related tasks by exploiting their commonalities and differences. Nevertheless, it is still not understood very well how multi-task learning can be implemented based on the relatedness of training tasks. In this survey, we review recent advances of multi-task learning methods in NLP, with the aim of summarizing them into two general multi-task training methods based on their task relatedness: (i) joint training and (ii) multi-step training. We present examples in various NLP downstream applications, summarize the task relationships and discuss future directions of this promising topic.

1 Introduction

Machine learning generally involves training a model to perform a single task. By focusing on one task, the model ignores knowledge from the training signals of *related tasks* (Ruder, 2017). There are a great number of tasks in NLP, from syntax parsing to information extraction, from machine translation to question answering: each requires a model dedicated to learning from data. Biologically, humans learn natural languages, from basic grammar to complex semantics in a single brain (Hashimoto et al., 2017). In the field of machine learning, multi-task learning (MTL) aims to leverage useful information shared across multiple related tasks to improve the generalization performance on all tasks (Caruana, 1997). In deep neural networks, it is generally achieved by sharing part of hidden layers between different tasks, while keeping several task-specific output layers. MTL offers advantages like improved data efficiency, reduced overfitting, and fast learning by leveraging auxiliary information (Crawshaw, 2020).

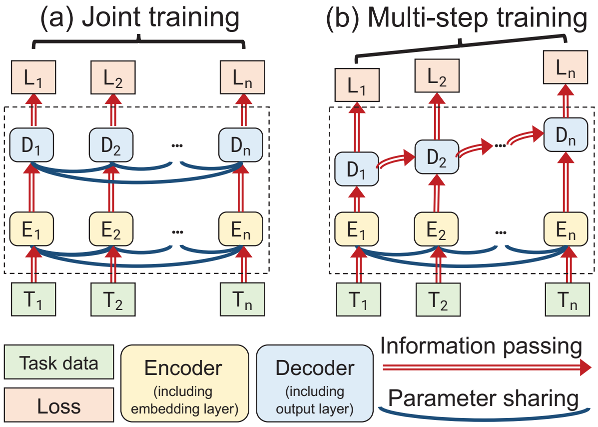


Figure 1: Two multi-task learning frameworks.

There have been relevant surveys that looked into architecture designs and optimization algorithms in MTL. Ruder (2017) classified different MTL frameworks into two categories: hard parameter sharing and soft parameter sharing, as well as some earlier MTL examples in both non-neural and neural models; Zhang and Yang (2018) expanded such two "how to share" categories into five categories, including feature learning approach, low-rank approach, task clustering approach, task relation learning approach, and decomposition approach; Crawshaw (2020) presented more recent models in both single-domain and multi-modal architectures, as well as an overview of optimization methods in MTL. Nevertheless, it is still not clearly understood *how* to design and train a single model to handle a variety of NLP tasks according to **task relatedness**. Especially when faced with a set of tasks that are seldom simultaneously trained previously, it is of crucial importance that researchers find proper auxiliary tasks and assess the feasibility of such multi-task learning attempt.

In this paper, we first review recent approaches on multi-task training methods in popular NLP applications. We find that these approaches can be categorized into *two multi-task training methods* according to the types of task relatedness: (i) joint training methods and (ii) multi-step training meth-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 943–956
May 2-6, 2023 ©2023 Association for Computational Linguistics

Multi-task training methods in our survey	Multi-task frameworks defined in Ruder (2017)	Multi-task frameworks defined in Crawshaw (2020)	Related papers
Joint Training	Deep relationship network	Shared trunk	(Liu et al., 2015; Dong et al., 2015)
	Cross-stitch network	Cross-talk	(Liu et al., 2016; Xiao et al., 2018)
	Weighting losses with uncertainty		(Xiong et al., 2018; Xu et al., 2019b)
	Sluice network		(Ruder et al., 2017)
		Adversarial feature separation	(Liu et al., 2017; Mao et al., 2020)
Multi-step Training		Prediction distillation	(Dinan et al., 2019; Lewis et al., 2020)
	Low supervision	Cascaded information	(Søgaard et al., 2016; Hashimoto et al., 2017)

Table 1: Categories of multi-task learning frameworks in two related surveys can be merged into our proposed joint training and multi-step training frameworks.

ods. **Joint training** is commonly used when all given tasks can be performed simultaneously and all the task-specific data can be learned simultaneously. In joint training, model parameters are shared (either via soft or hard parameter sharing[1]) among encoders and decoders so that the tasks can be jointly trained to benefit from shared representations, as shown in Figure 1(a). In contrast, **multi-step training** is used when some task's input needs to be determined by the outputs or hidden representations of previous task(s). Due to such task dependencies, the task-specific decoders are connected as a multi-step path starting from the encoder "node", as shown in Figure 1(b).

Therefore, different from previous surveys which focus on architecture designs (*e.g.*, how to share parameters in (Ruder, 2017) and (Zhang and Yang, 2018)) and optimization methods (*e.g.*, loss weighting and regularization in (Crawshaw, 2020)), our motivation lies in categorizing two major multi-task training methods in NLP, according to **task relatedness**. In fact, task relatedness is the key to determine what training method to use, then the training method decides the general scope of available architecture designs. With specific application tasks, readers are able to identify the ideal training method from our review before looking for detailed module design or loss optimization in previous surveys. We also show that how the MTL techniques covered in previous surveys can be matched with the two training methods in Table 1.

The remainder of this survey is organized as follows. Section 2 includes an overview of MTL models in NLP and the rationales of using MTL. Section 3 presents a number of joint and multi-step training applications in different fields of NLP. Section 4 analyzes the task relatedness involved in these MTL approaches. Section 5 discusses future directions. Section 6 concludes the paper.

2 Multi-task Training Methods

2.1 Encoder-Decoder Architecture and Two Multi-Task Training Frameworks

Suppose we train a model on n NLP tasks $T_1, \cdots, T_n$ on a dataset $\mathcal{D} = \{(X^{(i)}, Y^{(i)})\}|_{i=1}^{N}$ with N data points. For the j-th NLP task, the model is trained with $\{(X_j^{(i)}, Y_j^{(i)})\}|_{i=1}^{N}$, where $X_j^{(i)}$ is a component of the input $X^{(i)}$, and $Y_j^{(i)}$ is the desired output. The input components of different tasks can be the same, but the desired outputs are usually different. We formulate the multi-task frameworks discussed in this paper under the popular encoder-decoder architecture which are mainly composed of three components: (a) the encoder layer (including the embedding layer), (b) the decoder layer (including the output layer for classification or generation), and (c) loss and optimization.

Encoder layer. In NLP networks, an embedding layer is usually applied to generate the embedding vectors of the basic elements of the input $X^{(i)}$ For the j-th task, the encoder layer learns the hidden state of $X_j^{(i)}$ as a vector $\mathbf{h}_j^{(i)}$:

$$\mathbf{h}_j^{(i)} = \text{Encoder}(X_j^{(i)}, \Theta_{E_j}), \tag{1}$$

where Θ_{E_j} denotes the parameters of j-th task's encoder. Parameters of different encoders can be shared. Popular encoder modules include BiLSTM and BERT (Devlin et al., 2019).

Decoder layer. When the tasks are *independent* with each other at decoding, the decoder of the j-th task transforms the hidden state into an output:

$$\hat{Y}_j^{(i)} = \text{Decoder}_j(\mathbf{h}_j^{(i)}, \Theta_{D_j}). \tag{2}$$

When the tasks are *sequentially dependent*, the decoder of the j-th task needs the output of the $(j-1)$-th task, then we have

$$\hat{Y}_j^{(i)} = \text{Decoder}_j(\hat{Y}_{j-1}^{(i)}, \mathbf{h}_j^{(i)}, \Theta_{D_j}). \tag{3}$$

---

[1] We do not specifically distinguish different parameter sharing designs, since this topic is not the focus of our survey. We refer readers to learn details in Ruder (2017).

where Θ_{D_j} denotes the parameters of j-th task's decoder. Practically, $\hat{Y}_{j-1}^{(i)}$ are often presented as hidden representations of the decoder prediction to enable end-to-end training. Parameters of different decoders can be shared. Popular decoder choices include MLP, LSTM and the Transformer decoder.

According to the two types of task dependencies, the multi-task learning frameworks define and organize the decoders in two different ways. As shown in Figure 1, (i) the **joint training** framework is for the tasks that are independent at decoding; and (ii) the **multi-step training** framework is for tasks that are sequentially dependent. It can be easily generalized when the task dependencies form a directed acyclic graph, in which sequential dependence is a special and common case.

Optimization. A common optimization approach of MTL is to optimize the weighted sum of loss functions from different tasks, (i.e., Loss $= \lambda_j \sum_{j=1}^{n} \text{Loss}_j$) then compute the gradient descent to update all trainable parameters ($\{\Theta_{E_j}\}_{j=1}^{n}$, $\{\Theta_{D_j}\}_{j=1}^{n}$). The weights of $\{\lambda_j\}_{j=1}^{n}$ can be either pre-defined or dynamically adjusted (Kendall et al., 2018; Xiong et al., 2018). It is worth mentioning that optimization in MTL has many alternative ways. For example, Søgaard et al. (2016) choose a random task t from a pre-defined task sets to optimize its loss at each iteration. Readers can find a more detailed review of MTL optimization methods in Crawshaw (2020), which is not the main focus of this paper.

2.2 How does MTL Work

One of the prerequisites of multi-task learning is the relatedness among different tasks and their data. Most work prefers to train positively correlated tasks in a multi-task setting. Such tasks have similar objectives or relevant data, and can boost each other to form consistent predictions through shared lower-level representations. According to Caruana (1997), in MTL, tasks prefer hidden representations that other tasks prefer. MTL enables shared representations to include features from all tasks, thus improving the consistency of task-specific decoding in each sub-task. Furthermore, the co-existence of features from different objectives naturally performs feature crosses, which enables the model to learn more complex features.

According to the experiments by Standley et al. (2020), tasks are more likely to benefit from MTL when using a larger network. This can be achieved as the emergence of deep neural frameworks in recent years. Many deep models, like BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020), have strong generalization ability to fit a variety of tasks with minute changes. Therefore, different tasks can be learned through similar models, especially in the field of NLP where the encoder-decoder architecture has become a norm.

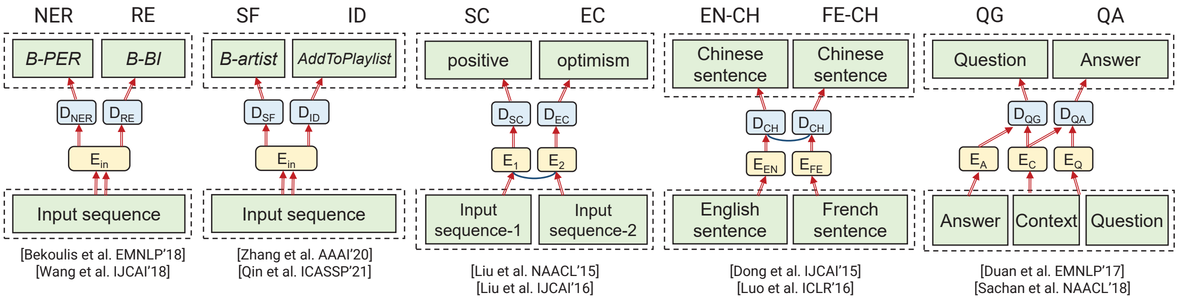
With the above premises, deep models are able to benefit from MTL in multiple perspectives. First, MTL improves data efficiency for each sub-task. Different tasks provide different aspects of information, enriching the expression ability of the hidden representation to the input text. Besides, different tasks have different noise patterns, which acts as an implicit data augmentation method. This encourages the multi-task model to produce more generalizable representations in shared layers. Thus, the model is prevented from overfitting to a single task and gains stronger generalization ability, which helps itself perform well when faced with new tasks from a similar environment (Baxter, 2000). Multi-task learning are also effective for low-resource tasks (Lin et al., 2018b; Johnson et al., 2017). Co-training with a high-resource task in a similar domain, low-resource tasks receive ampler training signals which prevents the model from overfitting on the limited data.

Auxiliary tasks in MTL can serve as conditions or hints for the main task. Such setting usually falls into the category of *multi-step training*. Providing additional conditions reduces the distribution space of possible outputs, thus lower the prediction difficulty of the main task. Such conditions can serve as additional features during decoding, including external knowledge pieces, low-level NLP tasks (*e.g.*, part-of-speech tagging or syntactic parsing) or relevant snippets extracted from long documents. When some features are difficult for the main task to directly learn, explicit supervision signals of such features, if available, enables the model to "eavesdrop", *i.e.*, obtaining these features through the learning of auxiliary task (Ruder, 2017).

3 Training Methods: Applications

3.1 Joint Training Applications

In this section, we list a series of recent approaches of joint training in different fields of NLP (shown in Figure 2), including information extraction, spoken language understanding, text classification, machine translation and language generation.



[Bekoulis et al. EMNLP'18]
[Wang et al. IJCAI'18]

[Zhang et al. AAAI'20]
[Qin et al. ICASSP'21]

[Liu et al. NAACL'15]
[Liu et al. IJCAI'16]

[Dong et al. IJCAI'15]
[Luo et al. ICLR'16]

[Duan et al. EMNLP'17]
[Sachan et al. NAACL'18]

Figure 2: Five joint training NLP applications that have been discussed from §3.1.1 to §3.1.5.

3.1.1 Information Extraction (IE)

Two popular tasks that are usually jointly performed in IE are named entity recognition (NER) and relation extraction (RE). NER seeks to locate and classify named entities in text into pre-defined categories, such as names and locations. NER is often tackled by sequence labeling methods, or say, token-wise classifiers. RE aims to extract semantic relationships between two or more entities, and there are multiple ways to define the RE task in the multi-task training approach.

First, Zhou et al. (2019) predicted the type of relation mentioned in a sentence by the RE decoder. It works for simple sentences such as those that have a pair of entities and one type of relation, e.g., "[President Obama] was *born in* [Honolulu]." However, one sentence may have multiple types of relations. Second, Zheng et al. (2017) predicted a relation tag for every pair of tokens. If the decoder performs perfectly, it can identify any number and any types of relations in a sentence. However, the complexity is too high to be effectively trained with annotated data. Third, Bekoulis et al. (2018); Wang et al. (2018a) treated RE as a sequence labeling problem. So both NER and RE decoders are token-wise classifiers. As shown in Figure 2, for example, $B\text{-}BI$ tag represents the beginning word of subject entity (person) or object entity (location) in the "born_in" (BI) relation. Therefore, if multiple tag sequences can be generated, they can identify any number, and any type of relations in the input sentence.

3.1.2 Spoken Language Understanding

Spoken Language Understanding (SLU) plays an important role in spoken dialogue system (Qin et al., 2021c). SLU aims at extracting the semantics from user utterances, which is a critical component of task-oriented dialogue. Concretely, it captures semantic constituents of the utterance and identifies the user's intent. These two tasks are typically known as slot filling (SF) and intent detection (ID), respectively. Each word in the utterance corresponds to one slot label, and a specific intent is assigned to the whole utterance. An example of these two sub-tasks is given below:

Word	Put	Kanye	into	my	rap	playlist
	↓	↓	↓	↓	↓	↓
Slot	O	B-artist	O	O	B-playlist	O
Intent	AddToPlaylist					

Since two sub-tasks share the same input utterance, they usually share a single utterance encoder and are jointly trained (Liu and Lane, 2016; Castellucci et al., 2019). Recent state-of-the-art SLU models build bi-directional interactions during encoding (Liu et al., 2019b; Zhang et al., 2020; Qin et al., 2021a). Therefore, two tasks mutually impact each other before making respective predictions. It is worth noting that there is also a line of work that uses the hidden states of intent detection to assist slot filling (Goo et al., 2018; Qin et al., 2019, 2021b). This can be considered as a combination of joint training and multi-step training: intent detection helps the prediction of slot filling, but finally their predictions are integrated to perform the parent (larger) SLU task.

3.1.3 Sentence/Document Classification

Sentence/document classification is one of the fundamental tasks in NLP with broad applications such as sentiment classification (SC), emotion classification (EC), and stance detection. However, the construction of large-scale high-quality datasets is extremely labor-intensive. Therefore, multi-task learning plays an important role in leveraging potential correlations among related classification tasks to extract common features, increase corpus size implicitly and yield classification improvements. Popular multi-task learning setting in text classification has two categories. First, one dataset is annotated with multiple labels and one input is associated with multiple outputs (Liu et al., 2015;

Yu et al., 2018a; Gui et al., 2020). Second, multiple datasets have their respective labels, i.e., multiple inputs with multiple outputs, where samples from different tasks are jointly learned in parallel (Liu et al., 2016, 2017). Most existing work leverages joint training for different sentence/document classification tasks. Specifically, Liu et al. (2016) proposed three different parameter sharing designs under the joint training framework, and further compared their performances.

3.1.4 Multilinguality

Languages differ lexically but are closely related on the semantic and/or the syntactic levels. Such correlation across different languages motivates the multi-task learning on multilingual data. Neural machine translation (NMT) is the most important application. Dong et al. (2015) first proposed a multi-task learning framework based on Seq2Seq to conduct NMT from one source language to multiple target languages. Luong et al. (2016) extended it with many-to-one and many-to-many approaches. Many-to-one is useful for translating multi-source languages to the target language, in which only the decoder is shared. Many-to-many studies the effect of unsupervised translation between multiple languages. Zhu et al. (2019) proposed to improve cross-lingual summarization by jointly training with monolingual summarization and machine translation. Arivazhagan et al. (2019) built a massive multi-lingual translation model handling 103 languages, and conducted experiments on multiple sampling schema for building joint training dataset.

Besides, unlabelled data from the target language is also a common source of multi-task cross-lingual training. Ahmad et al. (2019) collected unannotated sentences from auxiliary languages to assist learning language-agnostic representations. Van Der Goot et al. (2021) incorporated a masked language modeling objective using unlabeled data from target language to perform zero-shot transfer.

3.1.5 Natural Language Generation (NLG)

Recent success in deep generative modeling have led to significant advances in NLG, motivated by an increasing need to understand and derive meaning from language (Yu et al., 2020b). The relatedness between different generation tasks promotes the application of multi-task learning in NLG.

For example, Guo et al. (2018) proposed to jointly learn abstractive summarization (AS) and question generation (QG). An accurate summary of a document is supposed to contain all its salient information. This goal is consistent with that of question generation (QG), which looks for salient questioning-worthy details. Besides, QG and question answering (QA) are often trained as dual tasks. Tang et al. (2017) proposed a joint learning framework that connects QG and QA. QA improves QG through measuring the relevance between the generated question and the answer. QG improves QA by providing additional signal which stands for the probability of generating a question given the answer. A similar framework was also employed in Duan et al. (2017); Sachan and Xing (2018).

In other applications, semantic parsing is gaining attention for knowledge-based question answering since it does not rely on hand-crafted features. (Shen et al., 2019) developed a joint learning approach where a pointer-equipped semantic parsing model is designed to resolve coreference in conversations, and naturally empower joint learning with a novel type-aware entity detection model. Researchers also found NLU tasks, *e.g.*, input meaning representation learning (Qader et al., 2019) or entity mention prediction (Dong et al., 2020), can improve the performance of generating sentences

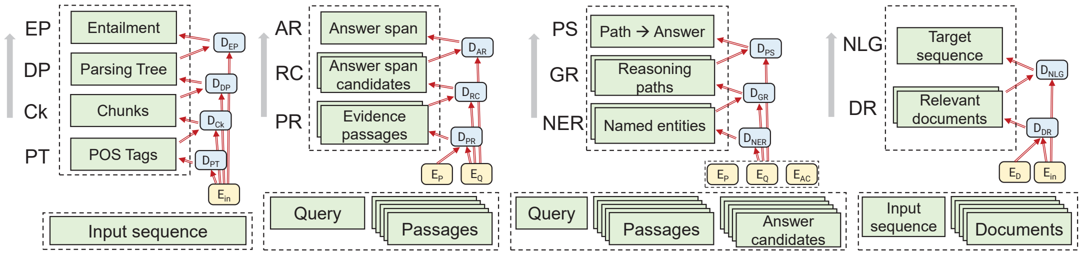
Multi-view learning is also applied in NLG approaches for auxiliary learning objectives. Input data are erased partially to create distinct views, and divergence metrics are usually learned along with the main loss to force the model generate consistent predictions across different views of the same input. Typical approaches include Clark et al. (2018) which built up the multi-view learning paradigm in IE and NLG tasks. In addition, Shen et al. (2020) upgraded the network by combining multiple cutoff methods to create augmented data, and achieved success in translation tasks.

3.2 Multi-step Training: Applications

We list recent approaches of multi-step training in different field of NLP (as shown in Figure 3), such as language understanding, multi-passage question answering and natural language generation.

3.2.1 Multi-level Language Understanding

The potential for leveraging multiple levels of representations has been demonstrated in various ways in the field of NLP. For example, Part-Of-Speech (POS) tags are used for syntactic parsers. The parsers are used to improve higher-level tasks, such as natural language inference. Søgaard et al. (2016) showed when learning POS tagging and chunking, it is consistently better to have POS supervision



[Søgaard et al. ACL'16]
[Hashimoto et al. EMNLP'17]

[Wang et al. AAAI'18a]
[Wang et al. ICLR'18b]

[Kundu et al. ACL'19]
[Tu et al. ACL'19]

[Dinan et al. ICLR'19]
[Lewis et al. NeurIPS'20]

Figure 3: Four multi-step training NLP applications discussed at §3.2.1, §3.2.2 (2nd and 3rd subfigures) and §3.2.3.

at the innermost rather than the outermost layer. Hashimoto et al. (2017) predicted increasingly complex NLP tasks at successively deeper layers for POS tagging, chunking, dependency parsing, semantic relatedness, and textual entailment, by considering linguistic hierarchies. Lower level predictions may influence predictions in higher levels, *e.g.*, if the semantic relatedness between two sentences is very low, they are unlikely to entail each other. Similar architecture can be found in Sanh et al. (2019b), where low-level tasks are name entity recognition and entity mention detection, with coreference resolution and relation extraction supervising at higher levels.

3.2.2 Multi-Passage Question Answering

Question answering (QA) models may need to construct answers by querying multiple passages (*e.g.*, paragraphs, documents). Given a question, multi-passage QA (MPQA) requires AI models identify an answer span from multiple evidence passages. Due to the complexity of MPQA, it is usually achieved by multiple sub-tasks. Thus, multi-step training is utilized by many approaches in MPQA.

Typically, MPQA can be split into a 3-phase task. *Passage retrieval* (PR) is to select relevant evidence passages according to the question. *Reading comprehension* (RC) is to extract multiple answer span candidates from the retrieved set of relevant passages. *Answer reranking* (AR) is to re-score multiple answer candidates based on the question and evidence passages. There exist dependencies between these tasks: evidence passages are generated by PR and fed into RC as input; the answer span candidates are generated by RC and given into AR as input. So, as shown in Figure 3, the decoders form a multi-hop path starting from the shared encoder. Hu et al. (2019) proposed a typical approach called RE^3 (for REtriever, REader, and REranker). The retriever used TF-IDF cosine similarities to prune irrelevant passages. The reader is a token classifier that predicts the start and end indices of answer candidates per segment. The reranker prunes redundant span candidates and then predict the reranking scores. Other works in MPQA also considered 2-phase approaches, such as PR+RC (Wang et al., 2018b) or RC+AR (Wang et al., 2018d), which are simplified versions of the above framework. Similar approaches have been developed for many domains such as news (Nishida et al., 2018), and web questions (Lin et al., 2018a).

Another branch of this task is *multi-choice* MPQA, where we have a set of answer candidates for the given question. (Kundu et al., 2019) proposed to exploit explicit paths for multi-hop reasoning over structured knowledge graphs. The model attempted to extract implicit relations from text through entity pair representations, and compose them to encode each path. It composed the passage representations along each path to compute a passage-based representation. Then it can explain the reasoning via these explicit paths through the passages. The sub-tasks are *named entity recognition* (NER), *graph-based reasoning* (GR) to extract and encode paths, and passage-based *path scoring* (PS). So, the multi-task QA systems perform interpretable and accurate reasoning (Welbl et al., 2018; Tu et al., 2019).

3.2.3 Retrieval-augmented Text Generation

In NLG, the input sequence alone often contains limited knowledge to support neural generation models to produce the desired output, so the performance of generation is still far from satisfactory in many real-world scenarios (Yu et al., 2020b). Retrieval-augmented generation models use the input sequence to retrieve relevant information (e.g., a background document) and use it as additional contexts when generating the target sequence. For example, Dinan et al. (2019) proposed to tackle the knowledge-aware dialogue by first selecting knowledge from a large pool of document candidates and generate a response based on the selected knowledge and context. To enhance the aforementioned idea, Kim et al. (2020) presented a sequential latent variable model to keep track of the prior and posterior distribution over knowledge. It not only reduced the ambiguity caused from the diversity in knowledge selection of conversation but also better leveraged the response information for proper choice of knowledge. Similar retrieval-augmented generation approaches have been applied in question generation (Lewis et al., 2020), comment generation (Lin et al., 2019b), image captioning (Xu et al., 2019a), summarization (Cao et al., 2018), and long form QA (Krishna et al., 2021).

4 NLP Task Relatedness

In this section, we summarize the characteristics of the aforementioned MTL approaches, and look into the task relatedness between the sub-tasks.

4.1 Joint Training

Joint training with similar tasks. Joint training with a similar task is the classical choice for multi-task learning. According to Caruana (1997), more similar tasks share more hidden units. Hence, similar tasks are more likely to benefit from shared generic representations. However, what kind of tasks can be considered as “similar" are not always evident in the deep learning era. Empirically selecting similar tasks is still the most mainstream method (Ruder, 2017; Worsham and Kalita, 2020). To get some intuitions what a similar task can be, here we introduce some prominent examples. (Dong et al., 2015) proposed training neural machine translation from one language into multiple languages simultaneously; (Yu et al., 2018a) proposed a joint training framework for sentiment classification and emotion classification; (Guo et al., 2018) proposed abstractive summarization can be jointly learned with question generation. (Yang et al., 2019) jointly trained question categorization and answer retrieval.

Recently, Aribandi et al. (2021) attempted to empirically select a set of tasks (from 107 NLP tasks) to transfer from, using a multi-task objective of mixing supervised tasks with self-supervised objectives for language model pre-training. In addition, Guo et al. (2019) used multi-armed bandits to select tasks and a Gaussian Process to control the mixing rates. Besides, some recent work tried to select appropriate sub-tasks based on manually defined features (Lin et al., 2019a; Sun et al., 2021). Aside from NLP, Fifty et al. (2021) proposed a method to select sub-tasks based on task gradients.

Auxiliary task for adversarial learning. Partial sharing of model parameters is the mainstream in multi-task learning, which attempts to divide the features of different tasks into private and shared spaces. However, the shared feature space could contain some unnecessary task-specific features, while some sharable features could also be mixed in private space, suffering from feature redundancy. To alleviate this problem, Liu et al. (2017) adds an adversarial task via a discriminator to estimate what task the encoding sequence comes from. Such learning strategy prevents the shared and private latent feature spaces from interfering with each other. This setup has also received success in multi-task multi-domain training for domain adaptation (Yu et al., 2018b). The adversarial task in this case is to predict the domain of the input. By reversing the gradient of the adversarial task, the adversarial task loss is maximized, which is beneficial for the main task as it forces the model to learn representations that are indistinguishable between domains.

Auxiliary task to boost representation learning. While auxiliary tasks are utilized to assist the main task, they are usually expected to learn representations shared or helpful for the main task (Ruder, 2017). Self-supervised, or unsupervised tasks, therefore, are often considered as a good choice. Self-supervised objectives allow the model to learn beneficial representations without leveraging expensive downstream task labels. For example, language modeling can help to learn transferable representations. In BERT (Devlin et al., 2019) pretraining, the next sentence prediction task is used to learn sentence-level representations, which is complementary to the masked language model task that mainly targets at word-level contextual representations. Besides, Rei (2017) showed that learning with a language modeling objective improves performance on several sequence labelling tasks. An auto-encoder objective can also be used as an auxiliary task. Yu et al. (2020a) demonstrated adding the auto-encoder objective improves the quality of semantic representations for questions and answers in the task of answer retrieval.

Another branch of auxiliary tasks used to facilitate representation learning is knowledge distillation. This is achieved by forcing a smaller student model to learn a larger teacher model’s output distribution or hidden representation using additional training objectives (Hinton et al., 2015). Knowledge contained in the hidden representations is then transferred from the teacher to the student. Thus, the student model gains the generalization ability of the teacher model, but still preserving its small size which is more suitable for deployment. Such distillation idea has been verified on popular NLP models such as BERT (Sanh et al., 2019a).

4.2 Multi-step Training

Narrow the search space of the subsequent decoder. In some cases, it is not easy to predict the original task directly due to the large search space of the potential outputs (Lewis et al., 2020). For example, in open-domain QA, directly answering a given question is hard. So, multi-stage methods (e.g., retrieve-then-read) are often used to tackle open-domain QA problems: a retriever component finding documents that might contain the answer from a large corpus, followed by a reader component finding the answer in the retrieved documents. Documents provided by the retriever serves as conditions to the reader, which narrows the search space and thus reduces the difficulty of open-domain QA (Wang et al., 2018b,c).

In another example about pre-trained language models, BERT only learns from 15% of the tokens that are masked in the input. ELECTRA (Clark et al., 2020) proposed a two-step self-supervised training to improve training efficiency. The masked language modeling task performed by an auto-encoder serves as an auxiliary task. It reconstructs the masked tokens in the input. Then, a discriminative model in the second step predicts whether each token in the corrupted input was replaced by the auto-encoder. The design of such classification task allows supervision on all tokens in the example.

Select focused contents from the input. The auxiliary task can be used to focus attention on parts of the input text that can be leveraged for the main task. For example, humans tend to write summaries containing certain keywords and then perform necessary modifications to ensure the fluency and grammatical correctness of the summary. Thus, keyword extraction could help the model to focus on salient information that can be used in the summary (Li et al., 2020). A similar approach can be found in Cho et al. (2019), where the authors used a flexible continuous latent variable for content selection to deal with different focuses on the context in question generation.

Predict attributes of the output. In some NLG scenarios, it may be hard to guarantee the output sequence contains certain desired patterns or features (e.g., emotion, sentiment) if no explicit signals are given. Therefore, an attribute classifier could be used for predicting whether the output sequence contains the desired objective, either before or after the prediction is made. For example, Fan et al. (2018) predicted which question type should be used before generating diverse questions for an image. The predicted question type acts as an additional condition while the decoder is searching for the best question sequence. Besides, Song et al. (2019) used a emotion classifier after the decoder to discriminate whether the generated sentence expresses the desired emotion. The post-decoder classifier guides the generation process to generate dialogue responses with specific emotions.

Introduce external knowledge. Precisely manipulating world knowledge is extremely hard for a single neural network model. One could devise learning tasks informed by the knowledge so that the model is trained to acquire and utilize external knowledge. This research direction is known as "Knowledge-enhanced NLP" (Yu et al., 2020b). The knowledge-related tasks can be combined as auxiliary to the main task, resulting in a multi-task learning setting (Dinan et al., 2019; Kim et al., 2020; Zhang et al., 2021). For instance, Wu et al. (2019) uses the input sequence to query the candidate knowledge pieces via attention mechanism, then fuses the selected knowledge into decoder. The knowledge selection phase is trained by minimizing the KL-divergence between the prior distribution (queried by the input) and the posterior distribution (queried by the output).

5 Future Directions

In this section, we will discuss some promising directions regarding either task relatedness or training methods of multi-task training in NLP.

5.1 Regarding Task Relatedness

Task-specific multi-task pre-training. Under a typical *"pre-train then fine-tune"* paradigm, many NLP works attempted to design pre-training tasks that are relevant to downstream objectives (Févry et al., 2020; Wang et al., 2021b). Such approaches endow the model with task-specific knowledge acquired from massive pre-training data. For example, Wang et al. (2021b) learned a knowledge embedding objective besides masked language modeling (MLM) to assist relation classification and entity typing tasks; Févry et al. (2020) and Zhang et al. (2022) added an entity linking objective into pre-training for fact checking and question answering applications. These results have shown that designing proper downstream-oriented pre-training tasks is a promising direction. Such pre-training tasks are jointly trained with the MLM objective to learn relevant knowledge of downstream tasks, which can greatly reduce the gap between pre-training and fine-tuning. These tasks need to be self-supervised on pre-training corpus, while sharing a similar learning objective with downstream tasks so that relevant knowledge can be transferred.

Learning to multi-tasking. One critical issue in MTL is how to *train* a multi-task model. Existing works typically design MTL training strategies (e.g., weighing losses or task grouping) by human intuition and select the best framework through cross validation. Such model selection suffers from heavy computational cost when considering every possibility. Thus, a promising direction is to learn to multi-tasking. Meta learning is a popular approach while encountering "learning to learn" problems (Hospedales et al., 2021). It aims to allow the model to quickly learn a new task, given the experience of multiple learning episodes on different tasks. Wang et al. (2021a) tried to fuse the feature of fast adaptation of meta learning into an efficient MTL model. This approach preliminarily proved that meta-learning philosophy can benefit the training of MTL models. For future directions, using meta-learning to learn a general purpose multi-task learning algorithm is a promising route to "learning to multi-tasking". Besides, learning to group tasks through meta-learning is worthy of exploration.

5.2 Regarding Training Methods

Adaptive parameter sharing. Parameter sharing is believed to be an effective technique in improving the generalizability of multi-task learning models and reducing training time and memory footprint. Two popular parameter sharing methods are hard and soft sharing (Ruder, 2017). Hard parameter sharing (Bekoulis et al., 2018) means all tasks share a certain number of model layers before branching out. Soft parameter sharing (Duong et al., 2015) adds constraints to the distances between specific layers of different tasks. However, hard sharing suffers from finding the optimal configuration while soft sharing does not reduce the number of parameters. Therefore, in addition to empirically tuning which layers to share, learning adaptable sharing for efficient MTL is a promising solution. Sun et al. (2020) tried to allow adaptive sharing by learning which layers are used by each task through model training. This approach suits in the field of computer vision where many models have the architecture of stacking the same layer. However, in NLP neural networks, layers are functionally and structurally discrepant, such as the encoder-decoder framework. The development of proper adaptive sharing methods to improve parameter sharing in multi-task NLP models is needed.

Mutli-task leaning in training a universal model. Recently, training a universal model to perform a variety of tasks becomes an emerging trend in NLP. Multi-task supervised learning helps the model fuse knowledge from different domains, and encourages it to obtain universal representations that generalize to different downstream tasks. For example, Liu et al. (2019a) unified the input format of GLUE tasks to feed into a single model before fine-tuning on individual tasks. However, the role of multi-task learning is still unclear in training a universal model, with different approaches adopting MTL in different phases of transfer learning. Among recent works, Aribandi et al. (2021) preferred multi-task pre-training over multi-task fine-tuning for smaller gaps between pre-training and fine-tuning. Aghajanyan et al. (2021) used multi-task pre-finetuning on a self-supervised pre-trained model before further fine-tuning on downstream tasks. Sanh et al. (2022) used prompted multi-task fine-tuning over a pre-trained T5 (Raffel et al., 2020) in order to perform zero-shot transfer on out-of-domain tasks. Therefore, future researches may dive deeper into maximizing the benefits of MTL in the transfer learning paradigm, including the choice of properly including MTL in pre-training or fine-tuning for better generalization. Besides, a theoretical analysis of transfer learning regarding the benefits of MTL is also desired.

6 Conclusions

In this paper, we reviewed recent work on multi-task learning for NLP tasks. According to the types of task relatedness, we categorized multi-task NLP approaches into two typical frameworks: joint training and multi-step training. We presented the design of each framework in various NLP applications, and discussed future directions of this interesting topic.

7 Limitations

Due to the space constraint, we are only able to show some prominent application scenarios of joint training and multi-step training, in which we may not cover all existing fields with multi-task approaches. For example, in dialogue systems, dialogue act recognition and sentiment recognition can be jointly trained to capture speakers' intentions. Besides, zero-shot and few-shot approaches in the multi-task setting are also interesting directions.

As for another limitation, this work is purely theoretical without any software-level implementation of the mentioned framework. In addition, we did not list the experimental results of the mentioned models on benchmark datasets because of the space limit.

Acknowledgements

This work was supported by NSF IIS-2119531, IIS-2137396, IIS-2142827, CCF-1901059, and ONR N00014-22-1-2507. Wenhao Yu is also supported in part by Bloomberg Data Science Ph.D Fellowship. We would also like to thank Libo Qin from Harbin Institute of Technology for his valuable suggestions to this paper.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Kai-Wei Chang, and Nanyun Peng. 2019. Cross-lingual dependency parsing with unlabeled auxiliary languages. In *The SIGNLL Conference on Computational Natural Language Learning (CoNLL)*.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q Tran, Dara Bahri, Jianmo Ni, et al. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. In *ArXiv preprint*.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. In *ArXiv preprint*.

Jonathan Baxter. 2000. A model of inductive bias learning. In *Journal of artificial intelligence research*.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Adversarial training for multi-context joint entity and relation extraction. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Rich Caruana. 1997. Multitask learning. In *Machine learning*.

Giuseppe Castellucci, Valentina Bellomaria, Andrea Favalli, and Raniero Romagnoli. 2019. Multi-lingual intent detection and slot filling in a joint bert-based model. In *ArXiv preprint*.

Jaemin Cho, Minjoon Seo, and Hannaneh Hajishirzi. 2019. Mixture content selection for diverse sequence generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference for Learning Representation (ICLR)*.

Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. In *ArXiv preprint*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference for Learning Representation (ICLR)*.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Xiangyu Dong, Wenhao Yu, Chenguang Zhu, and Meng Jiang. 2020. Injecting entity types into entity-guided text generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zhihao Fan, Zhongyu Wei, Piji Li, Yanyan Lan, and Xuanjing Huang. 2018. A question type driven framework to diversify visual question generation. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Lin Gui, Leng Jia, Jiyun Zhou, Ruifeng Xu, and Yulan He. 2020. Multi-task learning with mutual learning for joint sentiment classification and topic detection. In *IEEE Transactions on Knowledge and Data Engineering (TKDE)*.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2018. Soft layer-specific multi-task summarization with entailment and question generation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2019. Autosem: Automatic task selection and mixing in multi-task learning. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. In *Deep Learning Workshop at Advances in Neural Information Processing Systems (NeurIPS)*.

Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. 2021. Meta-learning in neural networks: A survey. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019. Retrieve, read, rerank: Towards end-to-end multi-document reading comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. In *Transactions of the Association for Computational Linguistics (TACL)*.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential latent knowledge selection for knowledge-grounded dialogue. In *International Conference for Learning Representation (ICLR)*.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Souvik Kundu, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. Exploiting explicit paths for multi-hop reading comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Haoran Li, Junnan Zhu, Jiajun Zhang, Chengqing Zong, and Xiaodong He. 2020. Keywords-guided abstractive sentence summarization. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018a. Denoising distantly supervised open-domain question answering. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ying Lin, Shengqi Yang, Veselin Stoyanov, and Heng Ji. 2018b. A multi-lingual multi-task architecture for low-resource sequence labeling. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, et al. 2019a. Choosing transfer languages for cross-lingual learning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2019b. Learning comment generation by leveraging user-generated data. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Bing Liu and Ian R. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *17th Annual Conference of the International Speech Communication Association*.

Pengfei Liu, Xipeng Qiu, and Xuan-Jing Huang. 2017. Adversarial multi-task learning for text classification. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yijin Liu, Fandong Meng, Jinchao Zhang, Jie Zhou, Yufeng Chen, and Jinan Xu. 2019b. Cm-net: A novel collaborative memory network for spoken language understanding. In *Conference on Empirical Methods in Natural Language Processing, (EMNLP)*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *International Conference for Learning Representation (ICLR)*.

Yuren Mao, Weiwei Liu, and Xuemin Lin. 2020. Adaptive adversarial multi-task representation learning. In *International Conference on Machine Learning (ICML)*.

Kyosuke Nishida, Itsumi Saito, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2018. Retrieve-and-read: Multi-task learning of information retrieval and reading comprehension. In *International Conference on Information and Knowledge Management (CIKM)*.

Raheel Qader, François Portet, and Cyril Labbé. 2019. Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of the 12th International Conference on Natural Language Generation*.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Conference on Empirical Methods in Natural Language Processing(EMNLP)*.

Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021a. A co-interactive transformer for joint slot filling and intent detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu. 2021b. GL-GIN: fast and accurate non-autoregressive model for joint multiple intent detection and slot filling. In *Annual Meeting of the Association for Computational Linguistics(ACL)*.

Libo Qin, Tianbao Xie, Wanxiang Che, and Ting Liu. 2021c. A survey on spoken language understanding: Recent advances and new frontiers. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. In *Journal of Machine Learning Research*.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. In *ArXiv preprint*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice networks: Learning what to share between loosely related tasks. In *ArXiv preprint*.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019a. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Workshop on Energy Efficient Machine Learning and Cognitive Computing at Advances in Neural Information Processing Systems (NeurIPS)*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference for Learning Representation (ICLR)*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019b. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. In *ArXiv preprint*.

Tao Shen, Xiubo Geng, QIN Tao, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-task learning for conversational question answering over a large-scale knowledge base. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Anders Søgaard, Yoav Goldberg, et al. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zhenqiao Song, Xiaoqing Zheng, Lu Liu, Mu Xu, and Xuan-Jing Huang. 2019. Generating responses with a specific emotion in dialog. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning (ICML)*.

Jimin Sun, Hwijeen Ahn, Chan Young Park, Yulia Tsvetkov, and David R Mortensen. 2021. Ranking transfer languages with pragmatically-motivated features for multilingual sentiment analysis. In *European Chapter of the Association for Computational Linguistics (EACL)*.

Ximeng Sun, Rameswar Panda, Rogerio Feris, and Kate Saenko. 2020. Adashare: Learning what to share for efficient deep multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. In *ArXiv preprint*.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Rob Van Der Goot, Ibrahim Sharaf, Aizhan Imankulova, Ahmet Üstün, Marija Stepanović, Alan Ramponi, Siti Oryza Khairunnisa, Mamoru Komachi, and Barbara Plank. 2021. From masked language modeling to translation: Non-english auxiliary tasks improve zero-shot spoken language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Haoxiang Wang, Han Zhao, and Bo Li. 2021a. Bridging multi-task learning and meta-learning: Towards efficient training and effective adaptation. In *International Conference on Machine Learning (ICML)*.

Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018a. Joint extraction of entities and relations based on a novel graph scheme. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2018b. R3: Reinforced ranker-reader for open-domain question answering. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018c. Evidence aggregation for answer re-ranking in open-domain question answering. In *International Conference for Learning Representation (ICLR)*.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. Kepler: A unified model for knowledge embedding and pre-trained language representation. In *Transactions of the Association for Computational Linguistics (TACL)*.

Zhen Wang, Jiachen Liu, Xinyan Xiao, Yajuan Lyu, and Tian Wu. 2018d. Joint training of candidate extraction and answer selection for reading comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. In *Transactions of the Association for Computational Linguistics (TACL)*.

Joseph Worsham and Jugal Kalita. 2020. Multi-task learning for natural language processing in the 2020s. In *Pattern Recognition*.

Wenquan Wu, Zhen Guo, Xiangyang Zhou, Hua Wu, Xiyuan Zhang, Rongzhong Lian, and Haifeng Wang. 2019. Proactive human-machine conversation with explicit conversation goal. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Liqiang Xiao, Honglun Zhang, and Wenqing Chen. 2018. Gated multi-task network for text classification. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. Dcn+: Mixed objective and deep residual coattention for question answering. In *International Conference for Learning Representation (ICLR)*.

Chunpu Xu, Wei Zhao, Min Yang, Xiang Ao, Wangrong Cheng, and Jinwen Tian. 2019a. A unified generation-retrieval framework for image captioning. In *International Conference on Information and Knowledge Management (CIKM)*.

Yichong Xu, Xiaodong Liu, Yelong Shen, Jingjing Liu, and Jianfeng Gao. 2019b. Multi-task learning with sample re-weighting for machine reading comprehension. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Min Yang, Lei Chen, Xiaojun Chen, Qingyao Wu, Wei Zhou, and Ying Shen. 2019. Knowledge-enhanced hierarchical attention for community question answering with multi-task and adaptive learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Jianfei Yu, Luis Marujo, Jing Jiang, Pradeep Karuturi, and William Brendel. 2018a. Improving multi-label emotion classification via sentiment classification with dual attention transfer network. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jianfei Yu, Minghui Qiu, Jing Jiang, Jun Huang, Shuangyong Song, Wei Chu, and Haiqing Chen. 2018b. Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. In *International Conference on Web Search and Data Mining (WSDM)*.

Wenhao Yu, Lingfei Wu, Qingkai Zeng, Yu Deng, Shu Tao, and Meng Jiang. 2020a. Crossing variational autoencoders for answer retrieval. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2020b. A survey of knowledge-enhanced text generation. In *ACM Computing Survey (CSUR)*.

Linhao Zhang, Dehong Ma, Xiaodong Zhang, Xiaohui Yan, and Houfeng Wang. 2020. Graph LSTM with context-gated mechanism for spoken language understanding. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Yu Zhang and Qiang Yang. 2018. An overview of multi-task learning. In *National Science*.

Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang. 2021. Knowledge-aware procedural text understanding with multi-stage training. In *Proceedings of the Web Conference*.

Zhihan Zhang, Wenhao Yu, Chenguang Zhu, and Meng Jiang. 2022. A unified encoder-decoder framework with entity memory. In *Conference on Empirical Methods in Natural Language Processing(EMNLP)*.

Suncong Zheng, Yuexing Hao, Dongyuan Lu, Hongyun Bao, Jiaming Xu, Hongwei Hao, and Bo Xu. 2017. Joint entity and relation extraction based on a hybrid neural network. In *Neurocomputing*.

Xin Zhou, Luping Liu, Xiaodong Luo, Haiqiang Chen, Linbo Qing, and Xiaohai He. 2019. Joint entity and relation extraction based on reinforcement learning. In *IEEE Access*. IEEE.

Junnan Zhu, Qian Wang, Yining Wang, Yu Zhou, Jiajun Zhang, Shaonan Wang, and Chengqing Zong. 2019. Ncls: Neural cross-lingual summarization. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Conclusion-based Counter-Argument Generation

Milad Alshomary
Institute of Artificial Intelligence
Leibniz University Hannover
m.alshomary@ai.uni-hannover.de

Henning Wachsmuth
Institute of Artificial Intelligence
Leibniz University Hannover
h.wachsmuth@ai.uni-hannover.de

Abstract

In real-world debates, the most common way to counter an argument is to reason against its main point, that is, its conclusion. Existing work on the automatic generation of natural language counter-arguments does not address the relation to the conclusion, possibly because many arguments leave their conclusion implicit. In this paper, we hypothesize that the key to effective counter-argument generation is to explicitly model the argument's conclusion and to enforce that the stance of the generated counter is opposite to that conclusion. In particular, we propose a multitask approach that jointly learns to generate both the conclusion and the counter of an input argument. The approach employs a stance-based ranking component that selects the counter from a diverse set of generated candidates whose stance best opposes the generated conclusion. In both automatic and manual evaluation, we provide evidence that our approach generates more relevant and stance-adhering counters than strong baselines.

1 Introduction

Given an argument, a valid counter-argument to it should be relevant to the topic discussed by the argument while opposing to its conclusion's stance. Countering the opponent's arguments in a debate effectively is key to winning the debate (Zhang et al., 2016). While some counter-arguments attack an argument's premises or their connection to the conclusion, the most common attack is to directly rebut the argument's conclusion (Walton, 2009).

The automatic countering of natural language arguments is one of the most challenging tasks in the area of computational argumentation. Prior research has addressed the task through retrieval (Wachsmuth et al., 2018; Orbach et al., 2020) or generation-based approaches (Hua and Wang, 2018; Hidey and McKeown, 2019). By concept, the former requires the presence of suitable counter-arguments in a predefined collection, limiting its flexibility. Existing generation-based approaches, on the other hand, either consider a single claim as input or do not model the relation between premise and conclusion in the input argument.

Conclusion (title): Purchasing meat encourages animal abuse.
Premises (post): All meat, to my knowledge, is obtained by raising animals in cramped quarters and slaughtering them as soon as they are fully grown. The only exception i can think of is perhaps when you go into the woods and hunt food for yourself in which case the animal has lived an undisturbed life and is put down by humane means compared to how it happens in nature. However, this is, of course, time intensive, requires skill, expensive, and thus is of course not how the vast majority of meat is obtained.

Table 1: An example argument (conclusion + premises) taken from *Reddit ChangeMyView*, showing how the conclusion is mentioned implicitly only in the body.

In previous work, we have studied the task of counter-argument generation through undermining weak premises in the input arguments (Alshomary et al., 2021b). We assumed the input argument to be given as a set of premises and their conclusion and modeled the weakness of premises relevant to the argument conclusion. In daily-life debates, however, people often do not explicitly state their argument's main point (i.e., its conclusion), since it is often clear from the context (Habernal and Gurevych, 2015) or for rhetorical reasons, as is often the case in news editorials (Al Khatib et al., 2016). This makes it challenging for computational models to generate a proper counter.

Table 1 shows an example of an argument with the conclusion "Purchasing meat is encouraging animal abuse". The author states that meat production would often lead to animal abuse. However, this statement is never linked explicitly to the conclusion. Such a link may be easy to infer for humans, but it is challenging for machines.

State-of-the-art language models based on transformers excel in many downstream text generation

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 957–967
May 2-6, 2023

tasks, such as summarization and machine translation (Vaswani et al., 2017). While they have been applied successfully for reconstructing implicit argument components such as conclusions (Gurcke et al., 2021; Syed et al., 2021), they still fall short on more complex tasks, such as counter-argument generation (Hua and Wang, 2019).

In this paper, we investigate how to enable transformer-based language models to generate an effective counter-argument to a given argument. We observe that the performance of these models in generating relevant counters with correct stance deteriorates particularly when the input argument does not mention its conclusion. Hence, we hypothesize that explicitly modeling the argument's conclusion and its stance will lead to more adequate counter-arguments. For this purpose, we propose a multitask generation approach with a stance-based ranking component. Our approach jointly models the two tasks of conclusion generation and counter-argument generation, and it enforces stance correctness through a stance-based ranking component.

Given a training dataset, where we have access to both the premises of arguments and their corresponding conclusions and counters, we explore two variations of our approach: The first shares the transformer's encoder and decoder between the two tasks, and we learn to generate both the conclusion and the counter as one sequence (separated with a special token). By contrast, the second variation is composed of one shared encoder along with two decoders, one to generate the conclusion and the other to generate the counter-argument. Although we expect the trained models to often capture the stance relation between the argument and its counter, we reinforce opposite stance through a stance-based ranking component at inference time. This component samples different counter-arguments and ranks them based on their stance score towards the corresponding generated conclusion.

To evaluate both approach variations, we use the ChangeMyView dataset of Jo et al. (2020), which consists of discussions where someone posts a view and others write comments opposing to this view. As in our previous work (Alshomary et al., 2021b), we use a post's title as the conclusion, its body text as the premises, and each comment as a counter. To classify stance as part of our ranking component, we fine-tune RoBERTa (Liu et al., 2019) on a dataset of pairs of claim and counter-claim collected from the *Kialo.org* debate platform. We compare our approach against two baselines; one that learns to generate the conclusion and the counter-argument independently in a pipeline model and one that employs a sequence-to-sequence model but does not actively represent the conclusion.

The results emphasize the deficiency of standard transformer-based models in counter-argument generation, particularly when the conclusion is not mentioned explicitly, highlighting the importance of conclusions in counter-argument generation. In most cases, our variation with shared encoder and decoder produces the best counter-arguments in terms of relevance and stance correctness.

We summarize our contributions as follows:[1]

- We study how to generate effective counter-arguments even if the attacked argument's conclusion is implicit.
- We present two multitask transformer-based counter-argument approaches, tuned to opposing to the argument's conclusion.
- We empirically reveal the impact of modeling an argument's conclusion and counter-argument jointly in the given task.

2 Related Work

Argument generation is one of the main branches of computational argumentation, studying the synthesis of arguments in natural language texts. This field includes a host of tasks like the generation of argument conclusions (Alshomary et al., 2020; Syed et al., 2021), implicit premises (Chakrabarty et al., 2021), controlled claims (Schiller et al., 2021; Alshomary et al., 2021a), as well as the generation of counter-arguments (Hua and Wang, 2018; Alshomary et al., 2021b). Our work studies the task of counter-argument synthesis.

The task of counter-argument synthesis has been addressed through either retrieval or generation-based approaches. An example of the former is the work of Orbach et al. (2020) whose approach tries to retrieve relevant counters for a given argument from a collection of documents. Wachsmuth et al. (2018) utilized topic knowledge to retrieve the best counter for a given argument.

On the other hand, generation-based approaches aim to construct counter-arguments from scratch. For example, both Bilu et al. (2015) and Hidey and

---

[1] The code of our experiments is available under `https://github.com/webis-de/EACL-23`

McKeown (2019) worked on the task of counter-claim generation. The former developed a set of rules and classifiers to negate claims, while the latter used neural methods to learn from data. Alshomary et al. (2021b) proposed an approach to generate counter-arguments by automatically identifying weak points in the input argument given the conclusion and attacking them. Moreover, Hua and Wang (2018, 2019) proposed an approach for generating long texts and applied it to the counter-argument generation task. Their approach relies on a retrieval component that acquires relevant key phrases for an input argument to be used to guide the generation of counter-arguments. While the size of the given argument collection limits retrieval-based approaches, the generation-based approaches either rely on the conclusion being given in the input or don't distinguish the different components in the input argumentative text. Our proposed approach is generation-based, where we study the conclusion's role in the counter-argument generation task.

Argument conclusion is the main point an argument argues towards/against, which is important for understanding the argument. In daily life argumentation, conclusions often are left implicit Alshomary et al. (2020). While it is easy for humans to infer the main point of an argument, it remains a challenging task for machines. Hence, several works have addressed the task of conclusion inference. Alshomary et al. (2020) reconstructed implicit claim targets from argument premises using triplet neural networks. Syed et al. (2021) studied the effectiveness of several transformer-based models on the conclusion generation and evaluated the informativeness criteria of conclusions. Gurcke et al. (2021) utilized conclusion generation to study argument quality. Our proposed approach also generates conclusions for a given argument as the first step in order to generate reliable counters.

3 Approach

As discussed above, the conclusions of arguments are important for understanding them properly. However, they are often left implicit, making understanding hard for machines. Our goal is to study how the absence of conclusions affects the performance of transformer-based counter-argument generation models. To alleviate this problem, we propose an approach that jointly learns to generate both the conclusion and the counter of an argument. At inference time, it utilizes a stance-based ranking component to select the most contrastive candidate counter in each case. We detail the generation and ranking in the following.

3.1 Joint Generation of Conclusions and Counter-Arguments

Text generation is usually modeled as a sequence-to-sequence generation task and is widely addressed through transformer-based encoder-decoder models (Vaswani et al., 2017). Since we aim to learn two generation tasks (conclusion and counter), one could think of either sharing the full model between the two tasks or only the encoder part. Hence, as illustrated in Figure 1, we experiment with both options to realize our approach:

Fully-shared Encoder and Decoder In the first model, we maintain the same transformer-based encoder-decoder architecture and train it to generate output sequences containing both the conclusion and the counter. Hence, the model learns to perform the two tasks simultaneously. Particularly, the input to the model is one sequence representing an argument's premises, and the output is a single sequence composed of the ground-truth conclusion and counter-argument separated by special tokens, `<conclusion>` and `<counter>`. The model encodes premises and decodes first the conclusion and then the counter in one sequence. We train the model to optimize the following loss function:

$$L(\theta) = -\sum_{1}^{n} \log p(y_i | x, y_{<i}; \theta)$$

Here, x is the input sequence that represents the premises, $y_{<i}$ is the sequence composing the conclusion and counter until the next word y_i, and θ denotes the model's parameters. We call this model *Joint One-seq* later in our experiments.

At inference time, we utilize a mechanism to generate a diverse set of n candidate conclusions and their counter-arguments, which are later passed to our stance-based ranking component to select the best counter. The diverse generation is as follows. We first extract a set of m Wikipedia concepts from the input premises using the approach of Dor et al. (2018). Then, during decoding, we use these concepts to prompt our trained model by masking all logits except the ones matching the prompt tokens, resulting in conclusions addressing different aspects of the premises followed by

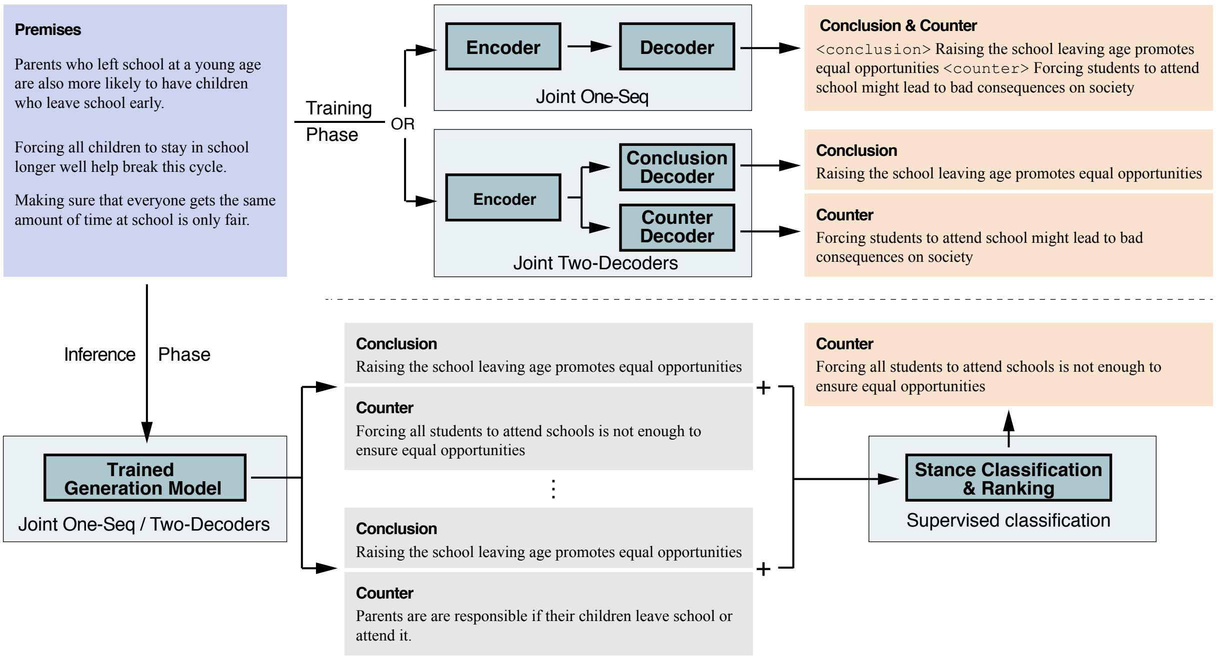


Figure 1: Both variations of our proposed approach to counter-argument generation. In the *training phase*, we learn to jointly generate the conclusion and counter either as one sequence (*Joint-based One-seq*, variation 1) or as two separated sequences (*Joint-based Two-decoders*, variation 2). In the *inference phase*, we classify and rank a diverse set of counters with respect to their stance towards the corresponding conclusion. The top-ranked counter is used.

their corresponding counters. Moreover, to ensure candidate diversity, we enable nucleus sampling (Holtzman et al., 2019), where at each step, we randomly select one of the top k tokens with an accumulated probability of more than p.

Shared Encoder with two Decoders Similarly, the second model starts with an argument's premises as input. However, it then decodes two independent sequences representing the conclusion and the counter-argument as output. First, the input premises are passed through a shared encoder, and then two decoders are used to learn to generate the counter and the conclusion. During training, we optimize the following multi-task loss function, which is a weighted average of the two language modeling losses of the two decoders:

$$
\begin{aligned}
L(\theta_e, \theta_a, \theta_b) = \quad & \alpha_a \cdot \sum_{i=1}^{n} \log p(y_i^a | x, y_{<i}^a; \theta_e; \theta_a) \\
+ \; & \alpha_b \cdot \sum_{i=1}^{m} \log p(y_i^b | x, y_{<i}^b; \theta_e; \theta_b)
\end{aligned}
$$

Here, y^a and y^b are the conclusion and counter sequences. θ_e, θ_a, and θ_b are the weight parameters of the encoder, the conclusion decoder, and the counter decoder, respectively. The weights, α_a and α_b, sum up to one. Their best values are determined experimentally during validation.

The difference between this model and the previous one is given by the layers shared between the two tasks. In the previous model, both the encoder and decoder layers are shared between the two tasks, while, here, only the encoder's layers are shared, keeping a dedicated decoder for each of the two tasks. We refer to this model as *Joint Two-decoders* below.

We aim to generate a diverse set of candidate counters similar to the above model. However, we noticed that counters rarely start by referring to entities or similar concepts, and prompting the model with concepts might lead to generating irrelevant texts. Hence, we generate one conclusion for this model, but a set of candidate counters by only enabling the nucleus sampling during decoding.[2]

3.2 Ranking Component

Give a set of n generated candidate counters, we rank them based on their stance contrastiveness towards the corresponding generated conclusion and select the top-ranked as our final output. In particular, we trained a transformer-based stance classifier on pairs of claim and counter-claim acquired from the *kialo.com* platform to be used to

---

[2] We tested the performance of the model empirically and noticed that these prompted counters of low quality.

predict whether the pair have a *pro* or *con* stance. Experimental details are provided in the next section. To guarantee stance coherence of the selected counter, we compute the stance-based scores on the sentence level to ensure all sentences have some degree of contrastiveness towards the conclusion. In particular, given a pair of a conclusion and the corresponding counter, we first split the counter into a set of sentences. For each sentence s_i, we apply our trained classifier to compute the stance $label$ towards the conclusion c and its probability pr_{label}. We then translate this into a stance contrastiveness score as follows:

$$cont(s_i, c) = \begin{cases} pr_{con}, & \text{if } label = con \\ -pr_{pro}, & \text{if } label = pro \end{cases}$$

The final score of a counter is averaged across its sentences, ranging from -1 to 1. The counters are then ranked accordingly, selecting the top one.

4 Experiments

This section describes the experiments carried out to investigate the conclusion's importance in counter-argument generation.

4.1 Data

We evaluate our approach on the ChangeMyView (CMV) dataset of Jo et al. (2020). On the CMV platform, users publish their opinions on controversial topics as posts consisting of a title summarizing the main point and a body representing the reasoning behind it. In turn, others comment on these posts trying to convince the authors to change their mind. We follow Alshomary et al. (2021b) by assuming the following mapping: The title of a post represents an argument's *conclusion* and its body is the *premises*, while each comment is a *counter-argument*. To ensure our models are trained on high quality counters, we select for each post the comment with highest argumentative quality score predicted by the model proposed by Gretz et al. (2020).

To study counter-argument generation for settings where the conclusion is not mentioned explicitly, we use only the post's body as input, and the title as training output to learn to generate the conclusion. Since users might also restate their post's main point (the conclusion) inside their post, this allows us to study and evaluate the correlation between a model's effectiveness in generating good counter-arguments and the level of implicitness of the conclusion in the input.

The stance-based ranking component relies on a classifier that assesses the stance polarity between two statements. To train such a classifier, we use dataset of Syed et al. (2021), which is based on the *Kialo.org* platform, where claims on controversial topics contributed by humans are organized in a hierarchical structure with supporting and opposing relations. We transformed the data into pairs of claims labeled as *pro* or *con*, and we split it by debates into 95.6k instances for training, 7.7k for validation, and 22.4k for testing.

4.2 Models

Approach For generation, we used BART as our base model (Lewis et al., 2020), and fine-tuned it starting from the *BART-large* checkpoint. We trained for three epochs using a learning rate of $5^e - 5$ and a batch size of 8. We then selected the checkpoint with the lowest error on the validation set. To find the best parameters α_a and α_b for the *Joint Two-decoders* model, we explored pairs of values between 0.1 and 1.0 on a sample of the training set, and took the pair that led to the lowest validation loss: $\alpha_a = 0.7$ and $\alpha_b = 0.3$.

To obtain a diverse set of candidate counters for ranking, we used nucleus sampling (Holtzman et al., 2019) with $p = 0.95$ and $top\_k = 50$. For the *Joint One-seq* model, we obtained relevant Wikipedia concepts from the input premises using Project Debater's API[3] that we used to prompt the output sequence (conclusion and counter-argument) to encourage diversity. As for the stance classifier, we fine-tuned *roberta-large* on the Kialo pairs for three epochs with learning rate $2e^{-5}$ and batch size 64. The trained classifier achieved an F_1-score of 0.81 on the test split. To test its performance on the ChangeMyView data, we took a sample of 2k instances with pro pairs (an argument and its conclusion) and con pairs (conclusion and counter). The trained classifier resulted in an F_1-score of 0.70.

Baselines To study how effective transformer-based models are when the conclusion is not explicitly stated, we compare against four BART-based models, all trained on the conclusion and premises as input and the counter-argument as output, but treated differently in the inference time.

---

[3] https://github.com/IBM/debater-eap-tutorial

Approach	**BLEU**	**Be.F$_1$**	**Stance**	**Contr.**
BART-based w/o Conclusion	0.149	0.138	0.814	0.447
Pipeline-based	0.148	0.142	0.816	0.437
Pipeline-based w/ Stance	0.141	0.142	0.852	0.615
Joint One-seq	0.143	***0.159**	0.850	*0.480
Joint One-seq w/ Stance	0.140	*0.147	**0.889**	***0.661**
Joint Two-decoders	*0.154	*0.148	0.798	0.423
Joint Two-decoders w/ Stance	***0.164**	*0.153	0.825	*0.652
BART-based w/ Conclusion	0.175	0.160	0.773	0.584
Argument Undermining	0.072	0.090	0.805	0.664

Table 2: Automatic evaluation of our two models, with and without *stance* ranking, compared to baselines, in terms of the similarity of the generated and the ground-truth counters (BLEU and BERT F$_1$-score) and of the counter's correct (opposing) stance. Stance is computed once using Project Debater's API (*Stance*) and once with our stance classifier (*Contrastiveness*). Results highlighted with * are significantly better than BART-based w/o Conclusion at a confidence level of 95%.

In particular, the first baseline (*BART w/o Conclusion*) relies only on the premises at inference time. To account for the missing conclusion, the second (*Pipeline-based*) generates a conclusion using another BART-based conclusion generation model trained independently on the training split of the CMV dataset. This can be seen a pipeline alternative to our approach, since conclusions and counters are learned independently. We also evaluate a variation of this pipeline approach that chooses the best counter among a diverse set of candidates using our ranking component (*Pipeline-based w/ Stance*). Finally, the fourth model is an oracle that knows the ground-truth conclusion in addition to the premises (*BART w/ Conclusion*).

Additionally, we compare our approach with the argument undermining approach of Alshomary et al. (2021b) in which the argument's weak points are first identified subject to its conclusion. Then a counter is generated to attack the weakest point(s). We obtained the trained model from the authors and used it to generate counter-arguments corresponding to the top three weak points (similar to their experiments).

4.3 Automatic Evaluation

In the following we introduce the automatic evaluation measures used in our experiments. We then present the evaluation results of our approaches, as well as a detailed analysis of their effectiveness with respect to argument length (measured by number of tokens) and conclusion implicitness.

Evaluation Measures To approximate the similarity of generated and ground-truth counters, we compute BLEU and BERT F$_1$-score[4]. In addition, we measure the stance correctness of the generated counter with respect to the ground-truth conclusion in two ways: First, a *contrastiveness* score is computed using the stance classifier trained for our ranking component. It represents the average likelihood of classifying the counter and the corresponding ground-truth conclusion as *con* across the evaluation dataset. Second, a target-based *stance* score that measures the stance of both the conclusion and the counter towards the conclusion target. Given the valdiation set, we extract the target of each conclusion for this purpose as proposed by Alshomary et al. (2020) and then use Project Debater's API[5] to classify the conclusion's stance and the generated counter's stance towards the extracted target. The final measure is the absolute difference between the counter and conclusion scores, averaged across the evaluation dataset.

Results Table 2 shows the evaluation results. All approaches are close in BLEU and BERT F$_1$-score, with small but significant advantages for our models. We observe that the absence of explicit mention of the conclusion in the input (*BART w/o Conclusion*) worsens the results across all measures but the Stance score, and vice versa when introducing the conclusion (*BART w/ Conclusion*). This clearly indicates the importance of the conclusion in the process of counter-argument generation.

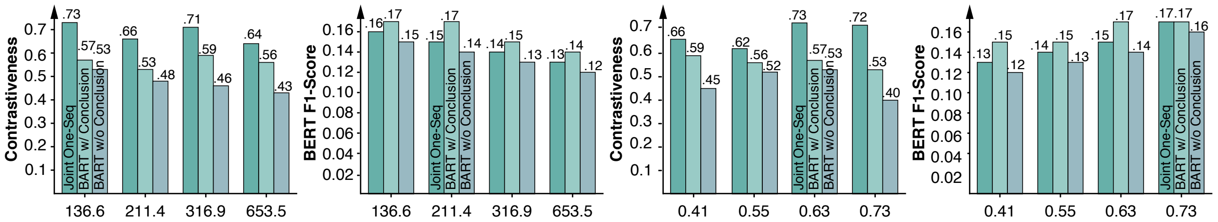
When the conclusion is not mentioned explicitly but has to be inferred, we can see that both our generation models which jointly generate conclusions and counters, outperform the baselines in terms of correct stance. As expected, adding the ranking component to our approaches and the pipeline baseline consistently boosts the correctness, the best being *Joint One-seq w/ Stance* with stance score 0.889 and contrastiveness score 0.661.

Although the Argument Undermining approach of Alshomary et al. (2021b) requires an explicit mention of the conclusion to rank premises according to their attackability, its effectiveness lacks behind. This could be because their model is trained on only a subset of the training data where the comments are countering specific points in the post.

---

[4] For each instance, we compare against all ground-truth counters and take the maximum score achieved

[5] Debater API, `https://early-access-program.debater.res.ibm.com/`



a) Premise's avg. number of tokens b) Premise's avg. number of tokens c) Avg. of Max. similarity to conclusion c) Avg. of Max. similarity to conclusion

Figure 2: Contrastiveness and BERT F_1-scores of our approach *Joint One-seq* against the baseline subject to different levels of argument complexity (approximated by the number of tokens) and conclusion implicitness (approximated by the maximum similarity of the ground-truth conclusion to the premises).

Analysis As discussed above, conclusions may appear in arguments implicitly, which we expect to correlate with the quality of the generated counters: the more explicit the conclusion, the better the generated counters. Moreover, we hypothesize that, the longer an argument is, the more important the inference of its conclusion is for effective counter-argument generation.

We empirically investigate these two hypotheses by comparing the performance of the counter-argument generation models subject to *argument length* (in terms of the number of tokens) and to the degree of *conclusion implicitness* (in terms of the maximum similarity between the ground-truth conclusion and premises). In particular, for both dimensions, we sorted the a sample of 2k instances from the test set accordingly and split it into five subsets of equal size. We then compare the BERT F_1-score and contrastiveness score of *Joint One-seq* against *BART w/o Conclusion* and *BART w/ Conclusion* on the respective subset.

Figure 2 shows the scores for all three models at different levels of argument length and conclusion implicitness. In Figure 2a, we see that the baseline's contrastiveness drops from 0.53 to 0.43 the longer the argument gets, while the scores for *BART w/ Conclusion* fluctuate relatively around 0.57. In contrast, our approach achieves scores between 0.64 and 0.73, indicating the benefit of the explicit modeling of conclusions. Figure 2c suggests that the more direct the conclusion is formulated in the premises, the better *BART w/o Conclusion*'s contrastiveness score gets, and vice versa for *BART w/ Conclusion* model.

We observe an unexpected drop in scores for arguments where conclusions have an average similarity of 0.7 to the premises. Upon inspection, we found that the baselines tend to copy parts of the premises with slight rephrasing. However, our approach, *Joint One-seq*, maintains high scores and also benefits from the clear formulation of the conclusion in the premises, since this helps to generate better conclusions.

Lastly, looking at BERT F_1-scores in Figures 2b and 2d, we notice that the values drop across all approaches as arguments get longer. Similarly, the more apparent the conclusion in the premises, the better the scores get.

4.4 Manual Evaluation

To gain more reliable insights into the performance of our approaches, we designed a human evaluation study to measure the quality of the generated counters in terms of relevance to the input argument and the correctness of their stance. In a second study, we also let humans assess the validity of the generated conclusions.

Counter-Arguments We selected 100 test set arguments randomly along with the counters generated by the two variations of our approach, *Joint One-seq w/ Stance* and *Joint Two-decoders w/ Stance*, as well as by two baselines, *BART w/o Conclusion* and *Pipeline-based*. Using the UpWork platform, we recruited three human annotators who are proficient in English with a job success of more than 90%. We presented them the 100 arguments together with the texts of the four given counters, shuffled pseudo-randomly for each argument. For each argument, we then asked them to rank the texts based on their adequacy of being a counter-argument to the input argument, where we defined adequacy as follows:

> *An adequate counter is a text that (1) carries an argumentative and coherent language and (2) clearly represents an opposing stance to one of the main points in the input argument.*

Additionally, the annotators should provide comments describing their decision regarding the coun-

	Annotator 1	**Annotator 2**	**Annotator 3**
Annotator 1	-	0.43	0.28
Annotator 2	0.43	-	0.30
Annotator 3	0.28	0.30	-

Table 3: Pairwise inter-annotator agreement in terms of Kendall's τ in the manual evaluation.

Counter Generation Approach	**Average ↓**	**Majority ↓**
BART-based w/o Conclusion	2.56	2.54
Pipeline-based w/ Stance	**2.38**	2.31
Joint One-seq w/ Stance	2.39	**2.26**
Joint Two-decoders w/ Stance	2.65	2.72

Table 4: Manual evaluation: The *average* and *majority* rank of the counters generated by our two approach variations and the two baselines. Lower is better.

ters ranked first (the best) and fourth (the worst). Computing the inter-annotator agreement using Kendall's τ results in an average of 0.32 (ranging from 0.32 to 0.43), while we observed majority agreement on full ranks between the annotators in 78% of the evaluated cases.

Table 3 shows the pairwise inter-annotator agreement of the three annotators in terms of Kendall's τ, resulting in an average of 0.32, and ranging from 0.28 to 0.43. We observe that *Annotator 1* and *Annotator 2* agree notably more with each other than with *Annotator 3*. We observed a full ranking majority agreement between our annotators in 78% of the evaluated cases.

Table 4 reports the mean of the average and majority ranks achieved by each approach. When considering cases with majority agreement, our model *Joint One-seq w/ Stance* performs best (mean rank 2.26). This also can be seen in Figure 3, where we plot the rank distribution for all approaches. In 55% of the cases, the approach generated counters that were ranked either first or second. However, the variation with two decoders falls short compared to all others (mean rank 2.72). This suggests that sharing only the encoder between the two tasks is not enough to generate relevant counters. Also, as indicated before, not being able to prompt the generated conclusions limits the diversity of candidates in the stance-based ranking component. Finally, we see that the *pipeline-based* baseline equipped with our ranking component is almost on par with our approaches, indicating the importance of promoting stance correctness.

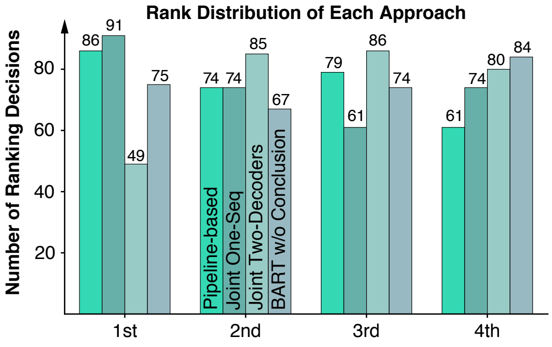


Figure 3: A histogram of the ranks that each of the manually evaluated approaches achieved on the 100 test cases, summing up the results of all three annotators.

Conclusion Generation Approach	**Validity ↑**
Pipeline-based w/ Stance	1.42
Joint One-seq w/ Stance	1.91
Joint Two-decoders w/ Stance	2.03
Ground-truth Conclusions	**2.39**

Table 5: Manual evaluation: Average *validity* score from 1 (non-valid) to 3 (valid) of the conclusions generated by our two approach variations and by the baseline, in comparison to the ground truth.

Conclusions To investigate whether the joint learning of conclusion and counter-argument generation leads to more valid conclusions, we designed another human evaluation study, for which we defined validity in a simple way:

> *A conclusion is valid if humans can infer it from the input argument.*

For 50 random arguments, we selected their ground-truth conclusion as well as two conclusions generated by the two variations of our approach and the best baseline (*Pipeline-based w/ Stance*), summing up to seven conclusions per argument. We hired two annotators through UpWork again. We asked them to read each argumentsand to evaluate the validity of each conclusion on a 3-point Likert scale, where 3 means that they strongly agree that the conclusion can be inferred and 1 means they strongly disagree. The agreement of the two annotators was 0.46 in terms of Cohen's κ.

Table 5 shows the average scores achieved by each evaluated model. With 1.42, the *pipeline-based* approach is notably worse than the others, indicating the advantage of multitask learning for conclusion and counter generation. The best score is achieved by *Joint Two-decoders w/ Stance* (2.03), being only 0.36 points below the ground-truth con-

Conclusion (title): Artificial General Intelligence does not need to be conscious to be more intelligent or effective than humans
Premises (post): In my opinion AGI needs the algorithms that make us intelligent and by having these missing pieces they will be a generally intelligent entity. I hear arguments that an AGI needs to have a subjective experience like a human being in order to be generally intelligent and I find this theory to be lackluster if the laws of physics can be computed than so can the algorithms that make general intelligence. After all we are information processing systems however, i am trying to explore all avenues of thought, even play devil's advocate and not get caught in some sort of echo chamber or even worse a spirit of the age.

BART-based w/o Conclusion:

Counter: I don't think that AGI needs to have a subjective experience like a human being in order to be generally intelligent. I think it's more accurate to say that an AGI doesn't need to have subjective experience, but that doesn't mean that it can't have an objective experience.

Pipeline-based w/ Stance:

Conclusion: i need to be more intelligent in order to be an AGI.
Counter: intelligence is not determined by a subjective experience like a human being. it is determined by the laws of physics. an agi does not need to have an objective experience to be intelligent, but it does need to be able to understand the world around it.

Joint-based One-seq w/ Stance:

Conclusion: Scientific law is the only thing that can make AGI generally intelligent.
Counter: The problem with AGI is that we don't really know what it is that makes us intelligent. we have no idea how it works, what it's like to be an AGI, how it's different from a human being, or how it will work in the real world.

Table 6: An example argument with conclusion (title) and premises (post), taken from *CMV*. Implicitly, the conclusion is also mentioned in the premises. Below, we show the output generated by each approach.

clusion's score. Given the low effectiveness of this model on the counter-argument generation task, we assume that the training process optimized more towards generating conclusions, especially since the task may be easier than generating counters. A better weighting scheme for the two tasks may alleviate this in future work.

Qualitative Analysis Table 6 shows an example argument discussing *Artificial Intelligence* along with counters generated by the two baselines as well as by our approach *Joint One-seq w/ Stance*. *BART w/o Conclusion* rephrases sentences from the input argument without generating a proper counter, possibly due to the ignorance of the conclusion. While the *pipeline-based* baseline equipped with our ranking component generates a somehow relevant conclusion, its counter still vague and doesn't clearly oppose the argument's stance. Finally, *Joint One-seq* infers a conclusion that addresses the main point of the input argument (*Scientific law*), and counter it by pointing out the difficulty of defining *intelligent* , making it hard to be measured.

Upon exploring annotators' comments that justified their decisions of what is the best/worse counter, we identified some patterns. For example, *Joint One-seq* was most appreciated, because it generated argumentative and coherent counters that sometimes offered new perspectives. In contrast, the cases in which the model's output was ranked worst happen mainly due to being vague, incoherent, or diverging from the main topic. The counters of *BART w/o Conclusion* were ranked worse due to coherences sometimes, but often due to not opposing to the input argument.

5 Conclusion

In this paper, we have studied the task of counter-argument generation, considering the role of the argument's conclusion. We argued that automatically generating counter-arguments becomes more challenging when the argument's conclusion is implicit, mandating explicit modeling. To validate our claims, we have proposed an approach that jointly learns to generate the conclusion and a counter for a given argument and compare it to baselines with no explicit conclusion modeling. Moreover, it explicitly enforces that the generated counters have a correct stance through a stance-based ranking component. We haved realized the approach in two ways, both using transformer-based models but with varying encoder-decoder concepts.

Although far from perfect, our results clearly suggest that the joint learning of the two tasks leads to better counters and to more valid conclusions of the input argument, in comparison to strong baselines. Thereby, we contribute substantially towards more robused counter-argument generation.

6 Acknowledgments

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): TRR 318/1 2021 - 438445824. We would also like to thank the reviewers and the participants who took part anonymously in our user study.

7 Limitations

In our evaluated, we have only experimented with BART as the underlining transformer-based model.

Additional experiment settings could demonstrate the gain of modeling the conclusion across other transformer-based models, such as GPT and T5.

Furthermore, we did not explore all possible weighting schemes for the two jointly learned tasks in our multitask setting. A potential extension could be to consider a more systematic evaluation of different schemes, for example, dynamic weighting schemes (Gong et al., 2019).

Lastly, our models are limited by the quality of the data we use. We have built on the assumption that CMV commentators rebut the original post's conclusion. However, this might not be a valid assumption in all cases and should be reassessed in future work.

8 Ethical Statement

Although our experiments demonstrate the role of conclusions in counter-argument generation, we believe that this task is far from solved. We are aware that issues such as faithful text generation must be considered when working with language models to avoid misinformation. We believe that mechanisms such as a fact-checking component or a factuality optimizer should accommodate any text generation model. The primary goal of our experiments is to highlight the potential of conclusion inference as part of the counter-argument generation pipeline, not to create an approach that is already ready for practical application.

References

Khalid Al Khatib, Henning Wachsmuth, Johannes Kiesel, Matthias Hagen, and Benno Stein. 2016. A news editorial corpus for mining argumentation strategies. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3433–3443. The COLING 2016 Organizing Committee.

Milad Alshomary, Wei-Fan Chen, Timon Gurcke, and Henning Wachsmuth. 2021a. Belief-based generation of argumentative claims. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 224–233.

Milad Alshomary, Shahbaz Syed, Arkajit Dhar, Martin Potthast, and Henning Wachsmuth. 2021b. Counter-argument generation by attacking weak premises. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1816–1827.

Milad Alshomary, Shahbaz Syed, Martin Potthast, and Henning Wachsmuth. 2020. Target inference in argument conclusion generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4345, Online. Association for Computational Linguistics.

Yonatan Bilu, Daniel Hershcovich, and Noam Slonim. 2015. Automatic claim negation: Why, how and when. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 84–93, Denver, CO. Association for Computational Linguistics.

Tuhin Chakrabarty, Aadit Trivedi, and Smaranda Muresan. 2021. Implicit premise generation with discourse-aware commonsense knowledge models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6247–6252.

Liat Ein Dor, Alon Halfon, Yoav Kantor, Ran Levy, Yosi Mass, Ruty Rinott, Eyal Shnarch, and Noam Slonim. 2018. Semantic relatedness of wikipedia concepts–benchmark data and a working solution. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Ting Gong, Tyler Lee, Cory Stephenson, Venkata Renduchintala, Suchismita Padhy, Anthony Ndirango, Gokce Keskin, and Oguz H Elibol. 2019. A comparison of loss weighting strategies for multi task learning in deep neural networks. *IEEE Access*, 7:141627–141632.

Shai Gretz, Roni Friedman, Edo Cohen-Karlik, Assaf Toledo, Dan Lahav, Ranit Aharonov, and Noam Slonim. 2020. A large-scale dataset for argument quality ranking: Construction and analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7805–7813.

Timon Gurcke, Milad Alshomary, and Henning Wachsmuth. 2021. Assessing the sufficiency of arguments through conclusion generation. In *Proceedings of the 8th Workshop on Argument Mining*, pages 67–77, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ivan Habernal and Iryna Gurevych. 2015. Exploiting debate portals for semi-supervised argumentation mining in user-generated web discourse. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2127–2137. Association for Computational Linguistics.

Christopher Hidey and Kathleen McKeown. 2019. Fixed that for you: Generating contrastive claims with semantic edits. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1756–1767.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Xinyu Hua and Lu Wang. 2018. Neural argument generation augmented with externally retrieved evidence. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 219–230, Melbourne, Australia. Association for Computational Linguistics.

Xinyu Hua and Lu Wang. 2019. Sentence-level content planning and style specification for neural text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 591–602.

Yohan Jo, Seojin Bang, Emaad Manzoor, Eduard Hovy, and Chris Reed. 2020. Detecting attackable sentences in arguments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–23, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Matan Orbach, Yonatan Bilu, Assaf Toledo, Dan Lahav, Michal Jacovi, Ranit Aharonov, and Noam Slonim. 2020. Out of the echo chamber: Detecting countering debate speeches. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7073–7086, Online. Association for Computational Linguistics.

Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. 2021. Aspect-controlled neural argument generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–396, Online. Association for Computational Linguistics.

Shahbaz Syed, Khalid Al Khatib, Milad Alshomary, Henning Wachsmuth, and Martin Potthast. 2021. Generating informative conclusions for argumentative texts. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3482–3493.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251. Association for Computational Linguistics.

Douglas Walton. 2009. Objections, rebuttals and refutations.

Justine Zhang, Ravi Kumar, Sujith Ravi, and Cristian Danescu-Niculescu-Mizil. 2016. Conversational flow in Oxford-style debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 136–141, San Diego, California. Association for Computational Linguistics.

A Computing Infrastructure

All our experiments are run inside an `ubuntu20.04` system using `Python 3.8.10`. The CUDA version is 11.2. We used one A100-SXM4-40GB GPU to train our models. The following libraries are required to run our experiments:

- torch==1.11.0+cu113
- transformers==4.18.0
- flair==0.11
- spacy==3.3.1
- debater-python-api==3.5.8

Question-Answer Sentence Graph for Joint Modeling Answer Selection

Roshni G. Iyer[1]*, **Thuy Vu**[2], **Alessandro Moschitti**[2], and **Yizhou Sun**[1]
[1]University of California, Los Angeles; Los Angeles, CA, USA
[2]Amazon Alexa AI; Manhattan Beach, CA, USA
{roshnigiyer, yzsun}@cs.ucla.edu, {thuyvu, amosch}@amazon.com

Abstract

This research studies graph-based approaches for Answer Sentence Selection (AS2), an essential component for retrieval-based Question Answering (QA) systems. During offline learning, our model constructs a small-scale relevant training graph per question in an unsupervised manner, and integrates with Graph Neural Networks. Graph nodes are question sentence to answer sentence pairs. We train and integrate state-of-the-art (SOTA) models for computing scores between question-question, question-answer, and answer-answer pairs, and use thresholding on relevance scores for creating graph edges. Online inference is then performed to solve the AS2 task on unseen queries. Experiments on two well-known academic benchmarks and a real-world dataset show that our approach consistently outperforms SOTA QA baseline models.

1 Introduction

Automated Question Answering (QA) research has received renewed attention thanks to diffusion of Virtual Assistants. For example, Google Home, Siri, and Alexa provide general information inquiry services, while many others serve customer requests in very different application domains. Two main tasks have been widely studied: (i) Answer Sentence Selection (AS2), which, given a question and set of answer-sentence candidates, consists of selecting sentences (e.g., retrieved by a search engine) that correctly answer the question; and (ii) machine reading (MR), e.g., (Chen et al., 2017), which, given a question and reference text, involves finding an exact text span that answers the question.

AS2 models can more efficiently target large text databases (as they originated from the TREC-QA track (Voorhees and Tice, 1999)) and there is evidence that they are currently used in personal assistants, e.g., Alexa (Matsubara et al., 2020).

q:	**Who won the 1967 NBA Championship ?**	*score*	*label*
c_1:	The 1967 NBA World Championship Series was the championship series of the 1966-67 National Basketball Association season and was the conclusion of the 1967 NBA Playoffs.	**0.810**	**0**
c_2:	This was the first championship series in 11 years without the Boston Celtics, who were defeated in the Division Finals by Philadelphia.	0.048	0
c_3:	The 76ers won the series over the Warriors 4-2.	**0.142**	**1**
q_1:	**Who won the 2009 Super Bowl ?**		
a_1:	The Steelers defeated the Cardinals by the score of 27–23.		
q_2:	**Who won Fifa World Cup 2010 ?**		
a_2:	In the final, Spain, the European champions, defeated third-time finalists the Netherlands 1–0 after extra time.		
q_3:	**Who won the most NBA championships ?**		
a_3:	Bill Russell won 11 championships with the Boston Celtics.		

Table 1: An example from WikiQA, for candidates ranked by TANDA with normalized scores and labels.

Garg et al. (2020) proposed the Transfer and Adapt (TANDA) approach, which obtained impressive improvement over SOTA for AS2, measured on two most used datasets, WikiQA (Yang et al., 2015) and TREC-QA (Wang et al., 2007). However, TANDA simply applies pointwise rerankers to individual question-answer pairs, e.g., binary classifiers, exploiting labeled out domain data (the ASNQ dataset proposed by the same authors).

The approach above was significantly improved by the Answer Support-based Reranker (ASR/MASR) (Zhang et al., 2021b), which jointly models answer candidates of each question. Essentially, the authors showed that answer candidates bring additional information for determining if a target answer t is correct, and proposed an ad-hoc joint model. ASR/MASR uses TANDA as fundamental building blocks for its model, but improves on TANDA's score via a component with AA similarity relations. Our model further improves this via a QQ similarity component integrated with a graph neural network (GNN) to capture more interrelation dependency.

A way to go beyond ASR/MASR's approach is to consider other questions similar to the target one along with their answers for deciding over t. For example, Table 1 reports a question, $q =$ *Who won the*

---

[*] This work was done while the author was an intern at Amazon Alexa AI.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 968–979
May 2-6, 2023

1967 NBA Championship?, with some candidate answers, c_1, c_2, and c_3, sorted by a pointwise reranker. c_1 contains the same phrase from the question, i.e., the phrase *The 1967 NBA World Championship*, which is also used by q to characterize the asked information (the name of the winner). The AS2 model mistakenly ranks c_1 before the other candidates, as the latter do not contain the phrase above. The model answers incorrectly because for most cases matching a portion of the question in the answer is a strong evidence of its correctness. In contrast, the correct answer c_3 (which also indicates a winner) does not contain the contextual entities, i.e., NBA and 1967.

The selection ability of the AS2 model would improve if it could learn the additional context. The latter can be provided by other similar question/answer pairs. For example, in the lower part of Table 1, we can see that the correct answer for q_3 (a question very similar to q) is a_3, which shows a predicate argument structure very similar to c_3, i.e., (**subj**:x, **verb**:*won*, **obj**: *championship*). Also a_1 and a_2 have somewhat similar structure, but most importantly all three similar questions provide evidence that it is not necessary having NBA or, in general, the name of the championship in the answer to make it correct.

The correctness characterization provided by the similar questions above can hardly be learned by the model from individual (q, c) pairs, as the characterization does not hold in general. It can be applied relatively to the list of candidates, $\{q, c_1, c_2, c_3\}$. Therefore, these patterns should be learned from comparing similar questions together with their list of candidates.

In this paper, we propose to jointly model questions with their list of candidates for AS2. This enables the usage of the information from other questions, similar to the given question, together with their answers. For this purpose, we design a graph-based QA model relying on the assumption that, given question q and a corresponding answer a, there exist other questions, q_i and answers a_i semantically similar to q, and a, respectively. These similar questions and answers can provide evidence to decide the correctness of a. To model the complex interactions among these different sentences, we use Graph Neural Networks (GNN) (Gori et al., 2005; Scarselli et al., 2009) applied to graphs which utilize interaction among question and answer pairs.

Our main innovation regards graph construction: (i) we propose to build a different small graph for each target question, such that we improve efficiency, and effectiveness as we use specific relevant questions; (ii) as we target answer selection, which traditionally is modeled as classification of question and answer pairs, (q, a), we associate graph nodes with them. We use training data to assign pairs with positive and negative labels. (iii) We form edges between pairs using models that automatically score different types of relations: question-question (QQ), question-answer (QA), and answer-answer (AA), and then we apply thresholds to reduce the number of active edges.

We test our models over three datasets, WikiQA, TREC-QA, and WQA, where the latter is an internal dataset built with de-identified customer questions. Our GNN for QA improves the best pointwise model for AS2, i.e., TANDA, over all datasets (up to 7 absolute points on TREC-QA corresponding to 75% of error reduction in Accuracy). It also establishes the new SOTA among joint models, improving ASR by 4 and 2 absolute points on WikiQA and TREC-QA, respectively.

2 Related Work

Our work aims at improving the answer sentence selection (AS2) task in open-domain question answering (ODQA).

Modeling for AS2 Previous work for AS2 modeling is typically categorized into three approaches: pointwise (Shen et al., 2017; Yoon et al., 2019; Garg et al., 2020), pairwise (Rao et al., 2016; Tayyar Madabushi et al., 2018; Laskar et al., 2020), and listwise methods (Cao et al., 2007; Bian et al., 2017; Ai et al., 2018). TANDA, and most other pointwise methods for AS2, however, overlook the natural existing inter-relations in the data. ASR/MASR is the current SOTA for joint modeling and considers multiple candidates for a target question. In this paper, we propose graph-based approaches for AS2, considering multiple questions and answers.

Graph-based Question Answering GNNs (Gori et al., 2005; Scarselli et al., 2009) have gained traction for their ability to effectively and scalably learn graph representations. Empirically, GNNs (Iyer et al., 2021) have achieved SOTA performance in many tasks such as node classification, link prediction, and graph classification. GNNs have been studied to improve QA in several ways. In multi-hop QA (Yang et al., 2018), GNNs are used to provide a structural presentation among several entities, e.g, questions, paragraphs, sentences, named entities to facilitate reasoning (Fang et al., 2020). However, the link between text is always triggered by named entities or concepts.

In our work, we use entire sentence semantics to link questions or answer text. Moreover, our semantic objects are pairs and we introduce relations between them. Recently, Yu et al. (2021) has used GNNs to exploit structural relationship described in pre-built knowledge graphs to improve ranking of passages. Their approach is different from ours: they use entity knowledge graphs. More importantly, we use multiple questions, while they only use multiple passages.

For other works proposing graph-based answer selection methods, they are limited in omitting important graph dependencies such as between questions (Tian et al., 2020), answers (Zhang et al., 2020), or both (Zhang et al., 2021a). Further, several existing approaches require additional information to be present, such as user reputation data (Lin et al., 2021; Zhang et al., 2021a), product reviews for product-related questions (Zhang et al., 2020), as well as external question and answer subject knowledge (Yang et al., 2022; Deng et al., 2021), assumptions that are no longer reasonable in real-world settings where this information may not be accessible. Our model overcomes these limitations, by considering the general setting where only question sentence and answer candidate information is available to effectively solve the AS2 task.

3 Methodology

In this section, we first present an overview of our QA Graph Model, followed by a discussion of our graph construction procedure. Then we describe our GNN model design, and lastly detail our training and inference framework.

3.1 QA Graph Model Overview

Our approach novelly reformulates the task of AS2 as a graph problem of node classification, where the goal is classify each (q, a) node from test to label 1 (correct answer) or 0 (incorrect answer). To this end, our approach carefully designs a QA graph guided by TANDA-RoBERTa, which captures important inter-relations between QQ, AA, and QA sentences, discussed in Section 3.2. GNNs are then used to learn final embeddings. GNN model parameters are learned during offline training, and online inference is performed for new node classification on unseen queries by turning the answer selection problem into the (q, a) node classification task.

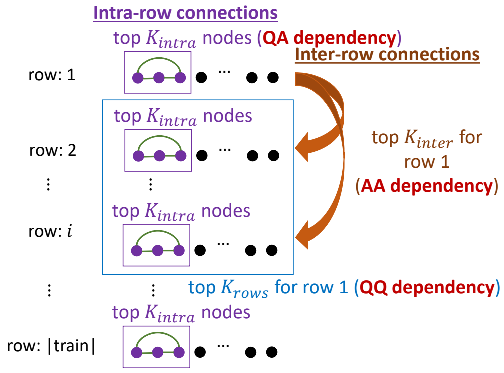


Figure 1: EQAG-GNN Graph Construction: each row is a question with all its answer candidates.

3.2 Graph Construction

Our graph construction procedure designs **E**ffective **QA G**raphs, EQAG. These graphs only have (q, a) nodes, the construction is based on K-best similar questions, and there is a final step to add intra- and cross- question connections to capture inter-relation QQ, AA, and QA sentence dependencies.

3.2.1 EQAG node construction procedure

Nodes are modeled as (q, a) sentence pairs (for design choice details, see the Appendix), and all graph nodes are of the type (q, a) (for motivation on node type choice, see Limitations section).

3.2.2 EQAG edge construction procedure

Figure 1 illustrates our approach: given query q, we first capture QA dependency using the top K_{intra} elements by determining which of the query's answer candidates to connect (through QA similarity). Then, we capture QQ dependency by identifying which similar questions to connect using top K_{rows} (through QQ similarity). Lastly, we capture AA dependency by identifying which similar answers to connect given the nodes already have similar questions, by using the top K_{inter} answers located in the top K_{rows} questions (through QA similarity from one node's question to the other node's answer, which was more effective than directly using AA similarity between the nodes' answers).

Capturing QA Dependencies. QA dependencies are explicitly captured via intra-row connections where for each query q, all corresponding query-answer nodes that have top K_{intra} scores from TANDA are connected for all training data (offline learning) and test data (online learning) independently. To reduce connections between noisy data, selected top K_{intra} nodes are further thresholded by th_{intra}. We include graph singleton nodes to enforce total question answerability coverage.

Capturing QQ Dependencies. QQ dependencies are *implicitly* captured by identifying similar questions to connect using top K_{rows} per query via QQ similarity. For QQ similarity, we used RoBERTa-base model trained on the open-domain Quora dataset [1]. Note that use of Quora is an additional improvement in our graph construction.

Capturing AA Dependencies. AA dependencies are captured by identifying similar answers to connect using top K_{inter} answers located in top K_{rows} questions, via TANDA's prediction scores. Inter-row connections are further ensured to connect only training data during the offline learning stage, and performs online inference solely on unseen test data. Similar to intra-row connections, to reduce connections between noisy data, selected top K_{inter} nodes are further thresholded by th_{inter}. Through empirical observation, for training nodes involved in cross-row connections, only connecting positively labeled training nodes improves model performance over connecting all labeled types of training nodes. These experiments, including other settings of inter-row connections are in Section 6. The latter also discusses hyperparameter settings for datasets.

In practice, we find similar questions to be abundant from datasets and therefore not a limitation in finding AA dependencies. Empirically, we also find similar questions have similar answers both in terms of semantics and syntax (structure). Hypothetically, in the extreme case of having only few similar questions, EQAG-GNN model accuracy will not be compromised since AA construction additionally requires thresholding where AA similarity score must be $\geq th_{inter}$, ensuring only sufficiently similar answers have dependencies.

3.2.3 Training Graph Construction Analysis

An alternative graph construction is explicitly introducing inter-relation dependencies as nodes: (q, q), (a, a), and (q, a). To keep graph size scalable, one may use semantically useful nodes, using QQ and AA language model similarity scores e.g., via RoBERTa with thresholding, where threshold is a hyperparameter e.g., $[0.7, 1.0]$. To form edges, one may use a similar procedure as node generation, where if any q or a elements between both nodes have a similarity score greater than the threshold value, to connect them. As GNNs do not update isolated nodes, one may further consider removing them to ensure scalable graph size.

However, this above approach shows several limitations: (i) not all questions (which were supposed to be answerable) may be answered. This occurs since the constructed graph may eliminate certain queries altogether since they do not satisfy threshold score when creating nodes. Further, only connected components greater than one element are considered by the baseline. In other words, isolated nodes are removed. The edge formation process may also remove edges due to thresholding of elements between nodes, thus creating isolated nodes that are subsequently removed. The thresholding may also remove correct answer candidates, though other candidates of that query may be kept. In this case, the model will be limited to only choosing an answer candidate from existing set of candidates labeled as incorrect, leading to lower performance.

EQAG overcomes nodes eliminated by thresholding by instead choosing top-K ranked nodes for all query-answer pairs. This guarantees all queries will have prediction made by the GNN (full answerability coverage) due to their presence in the graph, and further, correct answers that may have been removed due to thresholding will be included. Moreover, having all (q, a) pairs as nodes in the graph does not negatively affect a correct node's learned representation. Initial embeddings of all (q, a) nodes are produced from TANDA-RoBERTa classifier and then updated by a GCN-GNN model to learn final embeddings which are then mapped to a binary label. While all (q, a) nodes are present, only supporting evidence (q, a) nodes are connected to eachother (which is a sparsely connected instead of a fully connected graph). Therefore, the GCN-GNN model will only update node embeddings based on sparsely available node edges. Isolated nodes are not updated via message passing (as they have no neighbor node information to aggregate from), so such a suboptimal (q, a) node's information will not affect correct nodes.

4 Learning from QA graphs with GNNs

GNNs broadly use message passing with graph structure learning to inform a node's representation

---

[1] `kaggle.com/c/quora-question-pairs`

by a recursive neighborhood aggregation scheme. A node's neighborhood aggregation considers its local context of nodes, usually set to one-hop neighbor nodes, or directly connected nodes. In this way, utilizing a node's neighborhood for learning its representation takes into effect graph connectivity, node degree, and graph features. The general framework for GNNs is as follows:

$$\boldsymbol{h}_i^{(l+1)} = \sigma\big(\sum_{j \in N_i} f(\boldsymbol{h}_i^{(l)}, \boldsymbol{h}_j^{(l)})\big), \tag{1}$$

where $\boldsymbol{h}_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ is the feature representation of node v_i at layer l of the neural network, with dimensionality d. f is a message-specific neural network function of incoming messages to v_i from its neighborhood context N_i, and activation function σ, typically being $\mathrm{ReLU}(\cdot)$ for all layers but the last one which is $\mathrm{softmax}(\cdot)$.

Graph Convolutional Neural Networks. Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017) are a widely used class of GNNs, which have been shown to achieve superior performance on semi-supervised classification on graph-structured data. GCNs have been successfully applied to several networks including various citation network graphs, and knowledge graphs. GCN's framework is as follows where for a node $v_i = (q, a)$, where $q \in \mathrm{Q}_{\mathrm{Train}}, a \in \mathrm{A}_{\mathrm{Train}}$, its feature $\boldsymbol{h}_i^0 = \mathbf{x_i}$ is calculated by $\boldsymbol{h}_i^0 = \mathrm{score}_{\mathrm{TANDA}}(v_i)$:

$$\boldsymbol{h}_i^{(l+1)} = \sigma\Big(\boldsymbol{W}_l^T\big(\sum_{j \in N_i \cup \{i\}} \frac{e_{j,i}}{\sqrt{m_j m_i}} \boldsymbol{h}_j^{(l)}\big)\Big), \tag{2}$$

where $\boldsymbol{h}_i^{(l)}$ are embeddings of node v_i at layer $l \in [0, L]$, $\boldsymbol{W}_l$ is a layer-specific learnable weight matrix, N_i is the set of nodes in neighborhood context of v_i, $e_{j,i}$ is edge-weight between nodes $v_j \to v_i$, with default edge weight being 1.0 if edge exists. m_i and m_j are entries of degree matrix, with $m_i = 1 + \sum_{j \in N_i} e_{j,i}$. In other words, the GNN model only uses TANDA as initial embeddings for nodes. After that, the GNN model is used to update these embeddings through multiple layers of learning, which use message passing and local neighborhoods to update the node's representation.

In this work, we explore how GNNs applied to our QA graphs are effective in learning representations of QA nodes for AS2, through their ability to inform embeddings by capturing the latent QA, QQ, and AA dependencies between nodes.

GNN Loss Function. For the task of binary node classification, GNNs use binary-cross entropy (BCE) loss for training, where only the nodes from the training set are optimized:

$$BCE = -\frac{1}{N}\sum_{i=0}^{n} y_i \cdot log(\hat{y_i}) + (1 - y_i) \cdot log(1 - \hat{y_i}),$$

where y_i is the binary ground truth label for each query-answer, and $\hat{y_i}$ is the model's predicted probability score of the positive label, where $\hat{y_i} = \boldsymbol{h}_i^L$.

5 Training and Inference

Figure 2 summarizes EQAG-GNN's pipeline.

5.1 Training

After the graph is constructed, described in Section 3.2, it is passed to a GNN. We utilize $L = 2$ layers of GCN such that initial node features are TANDA's relevance scores, representing TANDA's prediction score of how well candidate c answers query q. After node features are updated by the GCN, the answer candidates are reranked by the learned relevance scores. Parameters of the GCN model, which include weight matrix $\mathbf{W}_l$, are optimized using BCE loss, described in Section 4, such that for every query-candidate node i, the learned relevance score $\hat{y_i}$ is computed per c and ranked. Further, learning rate and dropout rate are fine-tuned to the dataset. Our model's task is node classification, since each node is a query-answer to be classified with positive label (the candidate correctly answers the question) or negative label (the candidate incorrectly answers the question).

5.2 Inference

The inference stage is performed as online inference with only unseen query-candidate nodes. Here, each new query-answer pair forms one node, such that all possible answer candidates per query are considered. For each new query, the trained EQAG-GNN model leverages its neighborhood information of certain existing (q, a) nodes from training that it is connected to, based on top K_{inter} rows in EQAG. As computation of the top K_{inter} rows uses QQ, AA, and QA semantic similarity, the GNN model leverages these important inter-relation dependencies when classifying a new query-candidate node. Further, as the model's offline training is strictly performed on training nodes with no additional data leveraged, it is ensured that there is no information leak. Once EQAG-GNN

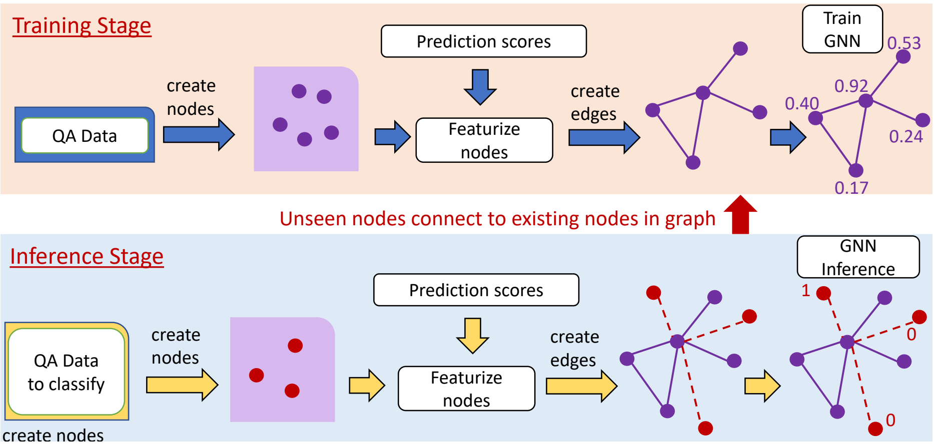


Figure 2: EQAG-GNN Pipeline. Graph nodes are (q, a) which have initial features being TANDA-RoBERTa's relevance score. During inference, for an unseen query, (q, a) nodes are created and connected to existing nodes in training graph. The learned GNN is then applied to obtain classification scores and mapped to binary labels.

predicts scores of unseen nodes, scores are ranked to find the candidate with the highest learned relevance score. Then, the soft-valued score is converted to a binary valued label where only the highest scoring candidate is assigned label 1 and the remaining candidates are label 0.

For effectiveness, during inference, we also do not fully connect all test answer candidates of a query when building the inference graph but instead use ranked top-K candidates guided by TANDA-RoBERTa's prediction score to form edge connections. This is suggested by our ablation study: more noise is included in the model when the QA graph becomes too densely connected. At the same time, an overly sparsely connected graph will also lead to less accuracy during inference of the model since GNN's message passing component will not be as effective. Therefore, we choose K as tunable hyperparameter on the target dataset.

GNN parameter size: Our GNN is also efficient in learned parameter size, as it uses the same parameter size as the efficient TANDA model and its GNN model parameter size is also minimal. Specifically, for the GCN GNN model, we used parameter size of $d \cdot d + n \cdot d$, where d is embedding size, and n is number of graph nodes. In our model setting, $d = 1$ and n is dataset specific, which in practice is in magnitude of a few thousand nodes in total, or in magnitude of a few hundred nodes per question on average. Specifically, n is around 26K, 52K, and 426K in total for the datasets of WikiQA, TREC-QA, and WQA respectively, which on average per question is 31, 42, and 121 nodes for WikiQA, TREC-QA, and WQA respectively.

WikiQA			
Statistics	Train	Dev	Test
#Q	873	121	237
#A+	1,040	140	293
#A-	7,632	990	2,058
TREC-QA			
Statistics	Train	Dev	Test
#Q	1,227	65	68
#A+	6,388	205	248
#A-	46,974	912	1,194
WQA			
Statistics	Train	Dev	Test
#Q	3,519	648	717
#A+	42,739	6,147	6,356
#A-	96,049	10,034	11,539

Table 2: AS2 dataset statistics, number questions, positive/negative answers, from official train/dev/test splits.

6 Experiments

Here, we compare EQAG-GNN against SOTA QA models. Then, we show an ablation study for EQAG-GNN for the best hyperparameters. Finally, we discuss a case study with error analysis.

6.1 Datasets

Table 2 shows a description of the datasets.

WikiQA: WikiQA (Yang et al., 2015) is an AS2 dataset containing data with form label-question-answer such that labels are binary, indicating existence of positive or negative QA pair, and there are several answer candidates for a question. Data comes from Bing query logs over Wikipedia where answers are manually labeled. We follow the most used setting: training with all questions having at least one correct answer, and validating and testing with all questions with at least one correct and one incorrect answer.

TREC-QA: TREC-QA (Wang et al., 2007) is an AS2 dataset containing data in a format like WikiQA of label-question-answer. We use the same splits of the original data, following the setting of previous work (Garg et al., 2020).

WQA: WQA (Zhang et al., 2022) is an AS2 dataset built with anonymized customers' utterances from a popular personal assistant. The dataset was built as part of the effort to improve understanding and benchmarking in ODQA. The creation process includes steps: (i) given a set of questions collected from the web, a search engine is used to retrieve up to 1,000 web pages from an index containing millions of pages. (ii) From retrieved documents, all candidate sentences are extracted and ranked using AS2 models. Finally, (iii) top candidates for each question are manually assessed as correct or incorrect by human judges. This allowed obtaining higher average number of correct answers with richer variety from multiple sources, shown in Table 2. Data is in a format similar to WikiQA of label-question-answer. For consistency with standard QA datasets, we filter out WQA for all non-answerable questions, or questions with only negative answer candidate choices.

6.2 Setup

Metrics: Performance of QA systems is typically measured with Accuracy being percentage of correct responses. This is also referred to as Precision-at-1 (P@1) in the context of reranking, while standard Precision and Recall are not meaningful as the system does not abstain from providing answers.

Implementation details: As the basic language model for our systems, we used the TANDA checkpoint, which is the SOTA AS2 (Garg et al., 2020). This is a pre-trained RoBERTa-base, further fine-tuned on ASNQ data [2]. We use the same reported optimal hyperparameter settings (Garg et al., 2020). Specifically, 4 Tesla V100 GPUs with 32GB for training and evaluation batch sizes of 32, with the maximum sequence length 128, and learning rate of 1e-6 for adapt step on the target dataset. We adopt Adam optimizer (Kingma and Ba, 2015) with learning rate of 2e-5 for the transfer step on ASNQ.

Model Parameters: To construct our QA graph, we used 8 Tesla V100 GPUs with 32GB with training batch size of 256. We utilized TANDA's configuration to guide the initial graph features, as described in Section 6.2. We then utilize the GCN model to complete the final step of model training, such that the node's embedding dimension size is set to 1, initially being TANDA's prediction scores. Learning rates were hyperparameters tuned from 1e-6, 2e-6, 5e-6, 1e-3, 2e-5, and used for WikiQA/TREC-QA/WQA with 1e-3, 1e-3, and 1e-6. Number of layers were hyperparameters tuned from 2, 4, 8, 16 and we utilized 2 layers for datasets. Training and eval batch sizes were 32.

6.3 Experiments with EQAG

We consider four EQAG-GNN model variants:

RI (Random Initialization): node features are randomly initialized with Gaussian uniformly random distribution between $[0.0, 1.0]$. This is to test impact that TANDA's model has on guiding EQAG-GNN's learned embeddings. For all variant models, we use hyperparameter settings from Section 6.3.

TA (Train All): both positively and negatively labeled training examples are considered to connect nodes between rows (inter-row connections).

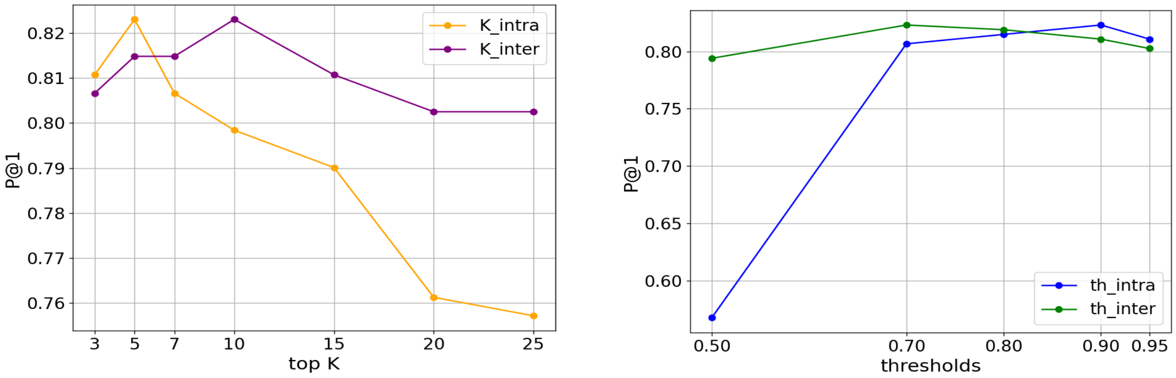
T$+$ (Train on positives): only the positively labeled training examples are considered to form the connections of nodes between rows. This approach seems to reduce the noise that incorrect training data may introduce to the model, when learning embeddings for all training data considered together.

T$\pm$ (Train positive and negative individually): for inter-row connections, both positively and negatively labeled training examples are used for connections, but considered separately. Specifically, we propagate both positive and negative node information throughout the network as different subcomponents. Positively labeled training data connects to the top K_{intra} connected test nodes, while negatively labeled training data connects to the bottom K_{intra} test nodes, which are isolated while ensuring that node features are less than th_{intra}. This may guide the model to learn better embeddings, as negative nodes will be influenced more by their negative neighbor context, while positive nodes will be influenced more by their positive neighbor context during the GNN message passing stage. This helps better distinguish embeddings between positive and negative test nodes during inference as overlap between local contexts of positive and negative nodes will be minimized.

Choosing EQAG-GNN hyperparameters We run ablation studies for hyperparameter tuning on the WikiQA dataset, evaluated on the validation

---

[2] Available at `github.com/alexa/wqa_tanda`



(a) top K_{intra} and top K_{inter} using WikiQA

(b) th_{intra} and th_{inter} using WikiQA

Figure 3: EQAG-GNN Hyperparameters on WikiQA

Model	**WikiQA**	**TREC-QA**	**WQA**
	P@1		
BR-MPGE-AS	0.835	0.912	0.600
L.D.C. Model	0.549	0.618	0.402
PWIM	0.823	0.824	0.582
HyperQA	0.827	0.853	0.598
Comp-Clip + LM + LC	0.827	0.838	0.590
TANDA	0.823	0.912	0.651
KGAT ($k = 2$)	0.844	0.941	–
ASR ($k = 3$)	0.844	0.971	–
MASR ($k = 3$)	0.823	0.927	–
EQAG-GNN (RI)	0.309	0.412	0.223
EQAG-GNN (TA)	0.840	0.956	0.671
EQAG-GNN (T+)	0.860	**0.985**	0.676
EQAG-GNN (T±)	**0.864**	**0.985**	**0.679**

Table 3: P@1 evaluation of GNN applied to EQAG and leading baselines. The best results are bold-faced.

set. For each hyperparameter tuned, we fix the values of all other hyperparameters to their optimal setting. The optimal settings of the variables, as shown in Figures 3(a) and 3(b), are $K_{intra} = 5$, $K_{inter} = 10$, $th_{intra} = 0.70$, and $th_{inter} = 0.90$. It can be observed that K_{intra} and th_{intra} may affect the P@1 accuracy more than K_{inter} and th_{inter}, perhaps because they directly impact how the intra-row connections between answer candidates of test queries are formed, which greatly influences the learned embeddings of each test node.

Comparative Results Table 3 reports P@1 accuracies of different SOTA QA models for ODQA on AS2 evaluated on WikiQA, TREC-QA, and WQA datasets. Models include TANDA, ASR, MASR, KGAT, EQAG-GNN model variants, described above, the SOTA GNN-based model for AS2, BR-MPGE-AS (Tian et al., 2020), the CNN-based sentence similarity model, L.D.C. (Wang et al., 2016), the Bi-LSTM CNN-based model which explicitly models pairwise word interactions, PWIM (He and Lin, 2016), the hyperbolic space embedding model, HyperQA (Tay et al., 2018), and the CNN-based latent clustering (LC) language model (LM), Comp-Clip+LM+LC (Yoon et al., 2019). All models use RoBERTa-base pre-trained checkpoint. The table shows QA-GNN consistently achieves highest performance among all models and all datasets on P@1 metric. For example, it outperforms TANDA by around 4, 7, and 3 absolute percent points in P@1 on WikiQA, TREC-QA, and WQA, respectively. Appendix tables 7 and 8 further report Maximum a Posteriori (MAP) and Mean Reciprocal Rank (MRR) scores, showing EQAG-GNN also achieves similar performance gains.

Results show 100% coverage of answerable questions and boosted accuracy of the approach. The ranking mechanism also improves graph connectivity, since all top-ranked question-answer pairs are included, though some nodes may be isolated. This technique greatly improves the amount of information needed by the GNN, compared to when only connected components without isolated nodes were considered. Finally, since the approach needs to start from weights learned by TANDA as random initialization, GNN-RI, produces very low results. Experiment results confirm effectiveness of modeling sentence-level semantics via graph-based models for AS2 (our approach) through comprehensive comparison of SOTA methods on AS2 including graph-based AS2 methods that model entity-level semantics. Further, our model consistently achieves significant improved performance against all baselines on all comprehensive metrics for AS2 (P@1, MAP, MRR), showing both accuracy and quality of QA pair ranking learned.

7 Case Study

We provide a case study and error analysis comparing performance of EQAG with TANDA. Ta-

q:	**How is Jameson Irish Whiskey made ?**
c_1:	Jameson is similar in its adherence to the single distillery principle to the single malt tradition but Jameson blends column still spirit with Single pot still whiskey a combination of malted barley with unmalted or " green " barley distilled in a pot still.
c_2:	Jameson is a single distillery Irish whiskey produced by a division of the French distiller Pernod Ricard.
c_3:	The company was established in 1780 when John Jameson established the Bow Street Distillery in Dublin.
c_4:	Originally one of the six main Dublin Whiskeys Jameson is now distilled in Cork although vatting still takes place in Dublin.
c_5:	With annual sales of over 31 million bottles Jameson is by far the best selling Irish whiskey in the world as it has been sold internationally since the early 19th century when John Jameson along with his son (also named John) was producing more than a million gallons annually.
c_6:	Portraits of John and Margaret Jameson by Sir Henry Raeburn are in the collection of the National Gallery of Ireland.
c_7:	Jameson was Scottish a lawyer from Alloa who had married Margaret Haig a sister of the brothers who founded the main Haig firms and related to the Steins a Scottish distilling family with interests in Dublin.

Table 4: Case study example of (q, a) from WikiQA.

	EQAG score	*TANDA score*	*label*
c_1:	**0.192**	**0.108**	**1**
c_2:	**0.158**	**0.505**	**0**
c_3:	0.103	0.110	0
c_4:	0.130	0.266	0
c_5:	0.140	0.009	0
c_6:	0.128	~ 0	0
c_7:	0.148	~ 0	0

Table 5: EQAG, TANDA scores for Table 4, with normalized predicted candidate answer scores underlined.

ble 4 reports question, q = *How is Jameson Irish Whiskey made?*, with candidate answers, c_1, through c_7, for which the AS2 model has to pick out the best candidate answer. Table 5 reports prediction scores learned by EQAG, and TANDA, per candidate answer and ground truth label.

7.1 Error Analysis

TANDA mistakenly ranks c_2 of label 0 before other candidates, while EQAG correctly ranks c_1 of label 1 above other candidates. Though phrase *Jameson Irish Whiskey* is important to the question, semantic intent of the question is how it is *made*. While TANDA recognizes importance of phrase *Jameson Irish Whiskey*, it does not learn necessary context of what the question asks for. As such, it picks the candidate describing what the whiskey is rather than how it is made. EQAG, however, places attention on both *Jameson Irish Whiskey* and question context *made*, while also learning that entire phrase *Jameson Irish Whiskey* may not need to be present in the candidate sentence as long as there is some indication of referring to the item e.g., *Jameson*.

As shown in Table 6, EQAG effectively learns from similar questions and its corresponding answer candidates that it sees during training to recognize important semantic characteristics of the question. For example, all three supporting questions e.g., q_1 through q_3 are about how various items (single malt scotch, bourboun, root beer) are made. Further, correct corresponding answers contain information about *both* item string name as well as how it is made. Regarding item string name, it is not necessary for the entire string name to be present as long as appropriate substring referring to this item is there. For example, a_1 refers to *single malt scotch* as simply *Scotch*, which EQAG also learns in order to identify that *Jameson Irish Whiskey* may be referred to as *Jameson*.

		(q, q_i) sim
q_1: a_1:	**How is single malt scotch made ?** As with any Scotch whisky, a single malt Scotch must be distilled in Scotland and matured in oak casks in Scotland for at least three years (most single malts are matured longer).	0.743
q_2: a_2:	**What is bourboun made of ?** Bourbon whiskey is a type of American whiskey– a barrel-aged distilled spirit made primarily from corn.	0.622
q_3: a_3:	**How is root beer made ?** Root beer is a carbonated, sweetened beverage, originally made using the root of a sassafras plant (or the bark of a sassafras tree) as the primary flavor.	0.498

Table 6: EQAG learned similar questions to question from Table 4, with corresponding absolute similarity scores between (q, q_i), $i \in [1, 3]$

8 Conclusions

To our knowledge, our model is the first graph-based approach for jointly modeling sentence-level semantics of question-answer pairs for AS2 as an offline processing application, such as those required by community QA, forums, etc. This is different from previous methods using graphs, e.g., MultiHop or Graph-based QA, which mainly model semantics via entities. Our approach builds query-specific small-scale training graphs for offline learning, through (q, a) pairs as nodes, and edges encoding relations between members of pairs to capture both supporting question-question, and answer-answer dependencies. Further, we demonstrate that our approach achieves significant performance gains over existing SOTA models on AS2 for metrics of P@1, MAP, and MRR.

Acknowledgments

The work of the first and last authors was partially supported by NSF 2211557, NSF 1937599, NSF 2119643, NASA, Okawa Foundation Grant, Amazon Research Awards, Cisco research grant, Picsart Gifts, and Snapchat Gifts.

Limitations

Our proposed model is efficient as its complexity is comparable to SOTA retrieval models like TANDA-RoBERTa. However, we note that out of our model components, the main complexity is from graph construction for the offline learning stage when constructing query-specific small-scale training graphs as opposed to graph processing, since GNN processing is typically fast. This is due to retrieval of top K-questions for each target query, which we perform with a RoBERTa cross-encoder. While the graph building takes roughly a few seconds per question in practice, given the scope of our problem of investigation for *offline* training and *online* learning, our model is efficient.

Further, while we capture Q-Q and A-A dependencies to form edge connections between the (q, a) nodes, the fact that the nodes in EQAG are all of the form (q, a) may be seen as a limitation. However, in the context of QA, the model deciding if an answer is correct or not for a question is trained over (q, a) pairs. This means that most information is captured by the pair, which is seen as the whole object. In general, we aim at modeling the similarity between pairs in the graph as we want to learn the patterns that make a pair correct. A graph having nodes as pairs directly enables this kind of learning, with also the great advantage that cross encoding two pieces of text in a transformer always produces a much higher accuracy than separated encoding of question and answer.

As future work, we plan to investigate building a model for learning graph topologies (Iyer et al., 2022), and other online processing applications e.g., document retrieval, by exploring methods like DPR (Karpukhin et al., 2020) to further speed up offline graph construction.

References

Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. *CoRR*, abs/1804.05936.

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *CIKM*, pages 1987–1990. ACM.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 129–136, New York, NY, USA. ACM.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *CoRR 2017*, abs/1704.00051.

Yang Deng, Yuexiang Xie, Yaliang Li, Min Yang, Wai Lam, and Ying Shen. 2021. Contextualized knowledge-aware attentive neural network: Enhancing answer selection with knowledge.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8823–8838, Online. Association for Computational Linguistics.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. TANDA: transfer and adapt pre-trained transformer models for answer sentence selection. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7780–7788. AAAI Press.

Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. *In Proceedings of International Joint Conference on Neural Networks*, pages 729–734.

Hua He and Jimmy Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement.

Roshni G. Iyer, Yunsheng Bai, Wei Wang, and Yizhou Sun. 2022. Dual-geometric space embedding model for two-view knowledge graphs. In *Proceedings of the 2022 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'22), Washington, DC, USA, August 14-18, 2022*, pages 676–686. ACM SIGKDD.

Roshni G. Iyer, Wei Wang, and Yizhou Sun. 2021. Bi-level attention graph neural networks. In *Proceedings of the 2021 IEEE International Conference on Data Mining (ICDM'21), Virtual Event, Auckland, New Zealand, December 7-10, 2021*, pages 1126–1131. IEEE ICDM.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. *ICLR 2015*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ICLR 2017*.

Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, and Enamul Hoque. 2020. Contextualized embeddings based transformer encoder for sentence similarity modeling in answer selection task. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5505–5514, Marseille, France. European Language Resources Association.

Zizheng Lin, Haowen Ke, Ngo-Yin Wong, Jiaxin Bai, Yangqui Song, Huan Zhao, and Junpeng Ye. 2021. Multi-relational graph based heterogeneous multi-task learning in community question answering.

Yoshitomo Matsubara, Thuy Vu, and Alessandro Moschitti. 2020. Reranking for efficient transformer-based answer selection. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1577–1580. ACM.

Jinfeng Rao, Hua He, and Jimmy J. Lin. 2016. Noise-contrastive estimation for answer selection with deep neural networks. In *CIKM*, pages 1913–1916. ACM.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, pages 61–80.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017. Inter-weighted alignment network for sentence pair modeling. In *EMNLP'17*, pages 1179–1189, Copenhagen, Denmark.

Yi Tay, Luu Anh Tuan, and Siu Cheng Hui. 2018. Hyperbolic representation learning for fast and efficient neural question answering.

Harish Tayyar Madabushi, Mark Lee, and John Barnden. 2018. Integrating question classification and deep learning for improved answer selection. In *COLING'18*, pages 3283–3294.

Zhixing Tian, Yuanzhe Zhang, Xinwei Feng, Wenbin Jiang, Yajuan Lyu, Kang Liu, and Jun Zhao. 2020. Capturing sentence relations for answer sentence selection with multi-perspective graph encoding.

E. Voorhees and D. Tice. 1999. *The TREC-8 Question Answering Track Evaluation*, pages 77–82. Department of Commerce, National Institute of Standards and Technology.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *EMNLP-CoNLL'07*, pages 22–32, Prague, Czech Republic. Association for Computational Linguistics.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Sentence similarity learning by lexical decomposition and composition.

Haitian Yang, Xuan Zhao, Yan Wang, Min Li, Wei Chen, and Weiqing Huang. 2022. Dgqan: Dual graph question-answer attention networks for answer selection.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. *CoRR*, abs/1905.12897.

Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2021. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. *CoRR*, abs/2110.04330.

Wei Zhang, Zeyuan Chen, Chao Dong, Wen Wang, Hongyuan Zha, and Jianyong Wang. 2021a. Graph-based tri-attention network for answer ranking in cqa. *AAAI 2021*.

Wenxuan Zhang, Yang Deng, and Wai Lam. 2020. Answer ranking for product-related questions via multiple semantic relations modeling.

Zeyu Zhang, Thuy Vu, Sunil Gandhi, and Alessandro Moschitti Ankit Chadha. 2022. WDRASS: A web-scale dataset for document retrieval and answer sentence selection. *CIKM 2022*.

Zeyu Zhang, Thuy Vu, and Alessandro Moschitti. 2021b. Joint models for answer verification in question answering systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3252–3262. Association for Computational Linguistics.

A Additional Experiment Results

Table 7 reports MAP scores and Table 8 reports MRR scores for EQAG variant models, as well as SOTA QA models for AS2. The tables further show that QA-GNN consistently achieves the highest performance among all models and all datasets on MAP and MRR metrics. For example, on MAP, it

Model	**WikiQA**	**TREC-QA**	**WQA**
	MAP		
BR-MPGE-AS	0.867	0.897	0.661
L.D.C. Model	0.706	0.771	0.582
PWIM	0.709	0.758	0.550
HyperQA	0.712	0.784	0.561
Comp-Clip + LM + LC	0.764	0.868	0.610
TANDA	0.889	0.914	0.653
KGAT ($k = 2$)	0.899	0.916	–
ASR ($k = 3$)	**0.901**	0.928	–
MASR ($k = 3$)	0.889	0.920	–
EQAG-GNN (RI)	0.384	0.485	0.301
EQAG-GNN (TA)	0.869	0.897	0.656
EQAG-GNN (T+)	**0.901**	0.926	0.658
EQAG-GNN (T±)	**0.901**	**0.941**	**0.662**

Table 7: MAP evaluation of GNN applied to EQAG and leading baselines. The best results are bold-faced.

Model	**WikiQA**	**TREC-QA**	**WQA**
	MRR		
BR-MPGE-AS	0.879	0.912	0.669
L.D.C. Model	0.723	0.845	0.598
PWIM	0.723	0.822	0.593
HyperQA	0.727	0.865	0.630
Comp-Clip + LM + LC	0.784	0.928	0.636
TANDA	0.901	0.952	0.681
KGAT ($k = 2$)	0.912	0.965	–
ASR ($k = 3$)	0.912	0.982	–
MASR ($k = 3$)	0.902	0.963	–
EQAG-GNN (RI)	0.359	0.544	0.371
EQAG-GNN (TA)	0.907	0.956	0.690
EQAG-GNN (T+)	0.916	0.971	0.697
EQAG-GNN (T±)	**0.924**	**0.983**	**0.699**

Table 8: MRR evaluation of GNN applied to EQAG and leading baselines. The best results are bold-faced.

outperforms TANDA by around 1, 3, and 1 absolute percent points on WikiQA, TREC-QA, and WQA, respectively. On MRR, it outperforms TANDA by around 2, 3, and nearly 2 absolute percent points on WikiQA, TREC-QA, and WQA, respectively.

B Nodes Modeled as Sentence Pairs

In the context of QA, we decide if an answer is correct or not for a question by training QA classifiers. This means that most information is captured by the pair, which is seen as the whole object. In general, we aim at modeling the similarity between pairs in the graph as we want to learn the patterns that make a pair correct. A graph having nodes as pairs directly enable this kind of learning, with also the great advantage that cross encoding two pieces of text in a transformer always produces a much higher accuracy than separated encoding of question and answer.

Evaluating and Improving the Coreference Capabilities of Machine Translation Models

Asaf Yehudai[1] **Arie Cattan**[2] **Omri Abend**[1] **Gabriel Stanovsky**[1]
[1]School of Computer Science, The Hebrew University of Jerusalem
[2]Computer Science Department, Bar Ilan University
`{asaf.yehudai,omri.abend,gabriel.stanovsky}@mail.huji.ac.il`
`arie.cattan@gmail.com`

Abstract

Machine translation (MT) requires a wide range of linguistic capabilities, which current end-to-end models are expected to learn implicitly by observing aligned sentences in bilingual corpora. In this work, we ask: *How well do MT models learn coreference resolution from implicit signal?* To answer this question, we develop an evaluation methodology that derives coreference clusters from MT output and evaluates them without requiring annotations in the target language. We further evaluate several prominent open-source and commercial MT systems, translating from English to six target languages, and compare them to state-of-the-art coreference resolvers on three challenging benchmarks. Our results show that the monolingual resolvers greatly outperform MT models. Motivated by this result, we experiment with different methods for incorporating the output of coreference resolution models in MT, showing improvement over strong baselines.[1]

1 Introduction

Machine translation (MT) may require coreference resolution to translate cases where the source and target language differ in their grammatical properties. For example, consider translating *"The trophy didn't fit in the suitcase because it was too small"* from English to French: *"Le trophée ne rentrait pas dans la valise car elle était trop petite"* (Sakaguchi et al., 2020). In French, suitcase (*"valise"*) is grammatically feminine, and trophy (*"trophée"*) is masculine, while the source-side English does not encode grammatical noun gender. This requires an MT model to infer that *"it"* refers to the suitcase (and not the trophy) to correctly produce the feminine inflection for the phrase *"it was too small"* (*"elle était trop petite"*), whereas an incorrect coreference resolution may produce the masculine inflection (*"il était trop petit"*) corresponding to the trophy.

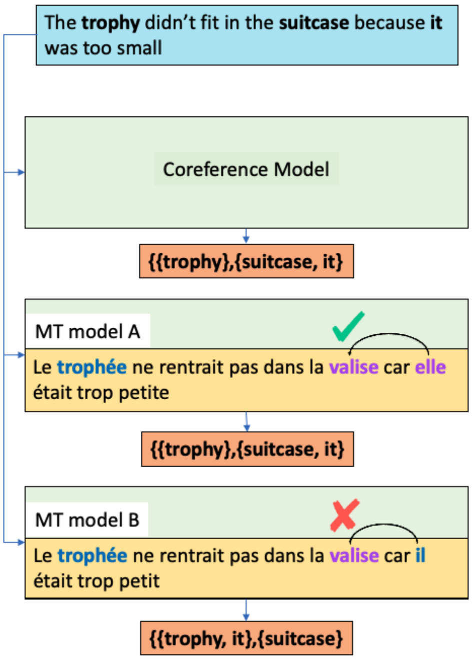


Figure 1: MT models can be compared to source-side coreference resolvers. An example translation from English (turquoise) to French (yellow). Our method first identifies the grammatical gender of the mentions in the target language marked in purple (female) and blue (male), followed by inferring the source side clusters (orange), through gender agreement.

Such texts evade lexical one-to-one translation, and instead demand source-side coreference resolution as a prerequisite for a correct translation. The prominent end-to-end approach to MT assumes that translation models implicitly learn source-side coreference resolution by observing aligned source-target pairs, without intermediate coreference supervision. While the importance of addressing such semantic phenomenon has been stated in various works (Le Nagard and Koehn, 2010; Stojanovski and Fraser, 2018), it was also observed that the ubiquitous BLEU metric (Papineni et al., 2002) does not adequately quantify it (Hardmeier and Federico, 2010; Freitag et al., 2022).

---

[1] https://github.com/AsafYehudai/MT-coref

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 980–992
May 2-6, 2023 ©2023 Association for Computational Linguistics

This work addresses the following research question: *How well does MT learn coreference when compared against explicit coreference supervision?* Answering this question can improve our understanding of the way MT models operate and also has practical implications: if implicit supervision lags behind monolingual training, it would motivate integration between end-to-end MT approaches and explicitly-supervised monolingual components.

In Section 3 we devise an evaluation paradigm that reduces MT output to source-side coreference resolution predictions by inferring coreference clusters from source inputs and predicted target translations. E.g., in the previous example, a feminine inflection for the pronoun "it" in French can infer linking "it" with "suitcase", while a masculine French inflection links "it" with "trophy", as shown in Figure 1. This approach allows us to distill the coreference resolution abilities of MT models and compare them against state-of-the-art coreference resolution models, trained explicitly on the task.

We use this approach to evaluate the coreference capabilities of several commercial and open source MT systems, translating from English to six target languages. We conduct our experiments in both synthetic (WinoMT and Wino-X; Stanovsky et al., 2019; Emelin and Sennrich, 2021) and naturalistic settings (BUG; Levy et al., 2021). Our results show that state-of-the-art coreference resolvers vastly outperform MT models on several benchmarks, indicating that explicit supervision may lead to better coreference performance.

Following this finding, in Section 4, we develop methods for improving coreference in MT, both implicitly and explicitly. Our implicit approach consists of fine-tuning MT models on texts that specifically require many coreference decisions, thus exposing the model to more implicit coreference signal. Our explicit approach further enriches source sentences with predicted coreference markers. We show that these approaches improve coreference over the end-to-end MT approach, achieving comparable or better results than much larger MT models, both commercial systems and open-source.

More broadly, our approach can be applied to improve the translation of other semantic phenomena that diverge in realization between source and target languages, such as plurality in second-person pronouns (Stanovsky and Tamari, 2019) or tense marking (Wolfram, 1985).

2 Background: Gender Bias in MT

We start our work by extending the methodology developed in (Stanovsky et al., 2019), which relies on target-side morphology to infer the translated gender of certain professions.

In particular, assuming a dataset of English sentences D, where each instance includes gold coreference annotation between a human entity and its pronoun (e.g., *"The* ***doctor*** *asked the nurse to help* ***her*** *with the procedure."*), they evaluate gender bias from English to language T with morphological gender in the following manner:

1. Predict word alignment between D and $M(D)$, i.e., the output translations of an MT model M. This finds the translations for pronouns (e.g., *"her"*) and possible entities (e.g *"doctor"*, *"nurse"*) in the target language T.
2. Automatically extract the gender of the possible entities and the pronouns in the target language based on morphological features.
3. Check whether the gender of the co-referred entity (e.g., *"doctor"*) in T corresponds to the gender of the English pronoun (e.g., *"her"*).

The gender bias of M is then defined as the difference in performance between stereotypical and anti-stereotypical gender role assignments.

We use a similar setup to address a different question: rather than evaluating the gender bias of the model, we evaluate its coreference abilities, which may be hindered by bias, but also by the inherent difficulty to infer coreference in the absence of an explicit training signal.

3 MT Models Fare Poorly Against Coreference Resolvers

The approach taken in WinoMT is limited as it restricts the evaluation to sentences with a known gender in English, indicated by a gendered pronoun of a human entity (e.g., *her*). Consider the sentence in Figure 1: *"The trophy didn't fit in the suitcase because it was too small"* with the coreference cluster $\{$suitcase, it$\}$. Step 3 in Stanovsky et al.'s method will fail to assign a coreference label to the translation, because *"it"* does not have a gender in English. In this section, we extend the WinoMT approach in order to estimate more general coreference abilities of MT models.

To achieve this, we note that many languages have gender agreement between pronouns and the noun that they refer to. Therefore, correct target-side gender agreement requires (implicitly) resolving the source-side coreference of the relevant entities. As exemplified in Figure 1, a reader of the French translation would infer from the gender inflection of "it" whether it refers to "suitcase" or the "trophy". I.e., a feminine pronoun ("elle") would agree with the feminine noun suitcase ("valise"), while a masculine pronoun ("il") would agree with the masculine noun trophy ("trophée'). We therefore formulate a new metric quantifying the ability of the MT model to implicitly resolve source-side coreference (henceforth, *Target-side Consistency*), defined as the proportion of instances in which the morphological gender of an entity (e.g., "suitcase") matches that of its referring pronoun (e.g., "it") in the target language T.

This metric examines whether an MT model is consistent in its coreference decisions, regardless of whether it correctly inferred the coreference relations in the input text. Indeed, some texts may keep the English ambiguity in the translation, and hence absolve the MT model from resolving coreference. For example, in the sentence *"The battery didn't fit in the suitcase because it was too small"*, both *battery* and *suitcase* are feminine in French. Our proposed metric will correctly indicate that the MT model was successful in such cases (albeit trivially). The metric thus serves as an upper bound on the MT model's coreference abilities.

We note that while this framing uses the morphological gender inflection of common nouns, it is different in motivation from measures of gender bias. In our example above, gender inflection allows us to determine whether an MT model correctly employs common sense rather than examining whether it tends to prefer stereotypical gender norms. While a model's gender bias may explain some loss in coreference abilities, the model's ability to resolve coreference need not be aligned with the degree of its bias (e.g., a random gender assignment would result in unbiased performance, but very poor coreference ability).

Most importantly, by considering the gender of the entity and the pronoun, we obtain mention clusters which can be compared against those produced by coreference resolution models. In our example figure, both the first MT model and the coreference model produce the correct clustering: {{trophy}, {suitcase, it}}, while the second MT model errs by producing: {{trophy, it}, {suitcase}}.

Another aspect of our evaluation methodology is its generality. Our method does not require a reference translation or make any particular assumptions about the generated output. As there are generally many correct translations, this flexibility allows us to accurately assess the model's coreference abilities. For instance, our methodology does not assume the gender of the entity's translation as can be seen in the first example in Table 7 where the two systems translate the entity "jar" differently. Further, some languages might not always translate an English pronoun into a pronoun but still express its gender in a different word. Consider the second example in Table 7 where the alignment model (step 2 in §2) finds that the English word "it" is aligned with both "l" and "trouvée" in French. Here, the feminine suffix of the past participle "trouv**ée**" indicates that the ellipsis "l" corresponds to a feminine entity.

3.1 Evaluation Setup

Evaluation datasets. The first dataset we use is Wino-X (Emelin and Sennrich, 2021), a filtered subset of WinoGrande (Sakaguchi et al., 2020) built to test commonsense reasoning and coreference resolution of MT models and multilingual encoders. The dataset contains sentences similar to the one in Figure 1. All sentences have two entities and a pronoun, "it", coreferring to one of them. The dataset consists of three parts, each part constructed for a different target language (German, French, and Russian), where each part only contains sentences where the two entities have different morphological gender in the target language (e.g., in French *"trophy"* is masculine whereas *"suitcase"* is feminine). Hence, applying our target-side consistency metric on these filtered sentences avoids trivial instances where both candidate entities have the same gender in the target language, and provides a clearer picture of the coreference capabilities of the model.

Second, we use WinoMT (Stanovsky et al., 2019),[2] a dataset built following the Winograd schema (Levesque et al., 2011), designed to test gender bias and coreference resolution of MT models. The sentences in this dataset contain two human entities and one gendered pronoun, e.g., *"The doctor asked the nurse to help her in the proce-*

---

[2] WinoMT is a combination of Winogender (Rudinger et al., 2018) and WinoBias (Zhao et al., 2018).

Dataset	#sentences
Wino-X (en → de)	3,774
Wino-X (en → fr)	2,988
Wino-X (en → ru)	2,238
WinoMT	3,888
BUG	1,717

Table 1: Statistics of our evaluation datasets. Note that in all datasets we use the corresponding English source-side sentences as our input.

dure". The gendered pronoun reveals the gender of the entity and adds gender attributes to the source cluster. In our example, *"her"* refers to the *"doctor"*, revealing the doctor's gender.

Our third dataset is BUG (Levy et al., 2021), a semi-automatic collection of naturalistic English sentences that are challenging with respect to societal gender-role assignments. Similar to WinoMT, each sentence contains a human entity, identified by their profession and a gendered pronoun. To reduce noise, we use the GOLD portion of this dataset which was validated by human annotators. All datasets statistics are presented in Table 1.

Machine translation models. We apply our evaluation methodology to four Transformer-based machine translation models from EasyNMT:[3] mBART50 (Tang et al., 2020; Liu et al., 2020), M2M_418M, M2M_1.2B (Fan et al., 2021), and the bilingual Opus-MT (Tiedemann and Thottingal, 2020), representing the state-of-the-art for publicly available neural machine translations models. In addition, we measure coreference consistency on the output of two commercial systems: Google Translate[4] and Microsoft Translator.[5]

Target languages. For WinoMT and BUG, we translate from English to six different languages: Arabic, German, Spanish, Hebrew, Russian and French. These languages form a diverse set with respect to how they encode grammatical gender (e.g., number of grammatical genders), as well as to their orthography, word order and other linguistic traits, while still allowing for highly accurate automatic morphological analysis. These languages belong to four families: (1) Romance languages: Spanish and French, which have gendered noun-determiner agreement with two grammatical genders; Spanish is also a *pro-drop* language, i.e., pronouns can be omitted in certain cases, which in our setting may keep the coreference ambiguity of the source-side English sentence (Webster and Pitler, 2020). (2) Slavic languages (with Cyrillic alphabets): Russian with 3 grammatical genders. (3) Semitic languages: Hebrew and Arabic, each with a unique alphabet; both are partial pro-drop languages and have two grammatical genders. (4) Germanic languages: German with 3 grammatical genders.

	Wino-X	WinoMT	BUG
SpanBERT	51.2	76.6	72.0
s2e	**60.8**	81.7	72.2
LINGMESS	58.7	**83.7**	**74.6**

Table 2: Accuracy of SpanBERT, *s2e* and LINGMESS model on our evaluation datasets. For simplicity, we report the accuracy on Wino-X sentences from the three languages as a single corpus, because there is a small difference between the languages (up to 0.3).

	en → de	en → fr	en → ru
mBART50	37.4	**56.6**	44.6
M2M_418M	31.4	48.5	**44.9**
Google	**41.3**	36.3	40.7
Microsoft	40.5	36.7	43.5
Opus-MT	37.4	35.3	43.6

Table 3: Target-side consistency results of commercial and open-source MT systems on Wino-X when translating into German, French, and Russian.

3.2 Target-side Consistency Results

We first evaluate the accuracy of existing coreference resolvers on our three evaluation datasets, where accuracy is defined as the percentage of instances in which the model identifies that the pronoun is coreferring with the correct entity. We select state-of-the-art models trained on CoNLL-2012 (Pradhan et al., 2012): SpanBERT (Joshi et al., 2020),[6] the s2e model (Kirstain et al., 2021) and LINGMESS (Otmazgin et al., 2022). Results in Table 2 show that coreference models perform quite well on WinoMT and BUG but poorly on Wino-X (60.8 for *s2e*), indicating weak commonsense capabilities.

Table 3 shows the target-side coreference consistency scores for all MT models on Wino-X, which, as mentioned above (§3.1), includes only sentences

---

[3] https://github.com/UKPLab/EasyNMT

[4] https://cloud.google.com/translate

[5] https://www.bing.com/translator

[6] Using AllenNLP's implementation (Gardner et al., 2018).

	WinoMT						BUG					
	de	es	fr	ru	he	ar	de	es	fr	ru	he	ar
mBART50	77.7	**75.2**	**73.3**	57.7	**69.3**	**69.5**	74.6	69.4	76.0	69.1	80.2	83.0
M2M_418M	69.7	55.1	65.4	54.7	56.5	64.3	81.3	84.9	86.0	**71.6**	83.6	85.1
M2M_1.2B	69.7	55.1	65.4	54.7	56.5	64.3	**83.5**	85.1	**90.3**	70.4	**89.6**	**93.4**
Google	69.9	59.4	65.5	**57.9**	60.1	69.2	56.0	84.4	89.2	71.0	85.2	91.9
Microsoft	**78.0**	66.5	69.3	57.3	63.9	60.7	77.2	**86.7**	86.0	67.2	84.0	89.8
Opus-MT	68.2	56.0	63.2	49.8	59.0	59.8	79.8	85.1	86.2	67.3	88.8	88.0

Table 4: Target-side consistency results of commercial and open-source MT systems on WinoMT and BUG when translating into different languages. These numbers are an upper bound for the source-side coreference accuracy.

where the entity and pronoun should be translated using different genders in the target language. We observe that all MT models perform poorly on Wino-X with the highest average score of 46.2 for mBART50, which vastly underperforms English coreference resolvers by 14.6 points. Interestingly, many instances are inconsistent because models tend to generate *neutral* pronoun whereas a gendered pronoun is expected, for example, *cela*, *c'était* in French (31% of translations) and это,они in Russian (17% of translations), meaning "this" or "they" in English. Likewise, 68% of German translations include neutral pronouns (e.g., "es"), while only 22% of the entities are neutral. The reported percentages were calculated on Opus-MT. Similar trends were observed in all models.

Table 4 shows the target-side consistency results for WinoMT and BUG. Following common practices on those datasets (Stanovsky et al., 2019; Levy et al., 2021), we omit sentences where the candidate pronoun does not provide information about the entity's gender. For example, in French, possessive pronouns agree with the gender of the possessed object, rather than the possessor as in English. Another example is in Spanish, which is a pro-drop language, where a valid translation can drop the pronoun and use a generic verb, leaving the only gender signal in the translation to be marked on the profession noun. See App. §C for more examples.

Similarly to Wino-X, target-side consistency results on WinoMT are consistently lower than coreference resolvers. Further, we observe that consistency is affected by two factors: the MT model and the target language. Regarding models, Opus-MT achieves lowest performance, with average consistency of 59.3, while mBART50 achieves high results with average consistency of 70.5, sometimes surpassing the second-best MT model by about 9 points. This might be due to the extensive pretraining of mBART50, as previously demonstrated for monolingual LMs (Huang et al., 2019; Sakaguchi et al., 2020; Bhagavatula et al., 2020). With respect to target languages, Russian consistency results are systematically lower than the results in other languages, to the extent that the best model in Russian provides lower results than the worst model in most other languages. In contrast, all models in German achieve a consistency score of about 70 or more, which can be due to its similarity with English and the research focus on improving English-German translations.

Consistency results on BUG are higher than on WinoMT for most models, while sometimes surpassing English coreference resolvers, notably in Hebrew and Arabic (e.g., 91.8 for Google vs. 74.6 for LingMess). To understand this gap, we analyze the translation of 50 BUG sentences to Hebrew and French and find that most instances (45 in Hebrew and 33 in French) do not include a distracting entity which should be translated to a different gender in the target language. As mentioned above (§3), our metric trivially indicates those examples as consistent.

Overall, target-side consistency results across all datasets demonstrate that both open-source and commercial MT systems exhibit rather poor coreference capabilities compared to English coreference models.

3.3 Human Validation

The use of automatic tools in the proposed methodology inevitably implies the introduction of noise into the process. To assess the quality of our measurements, we randomly sampled 50 translations of the Opus-MT model from all evaluation datasets and in all target languages (for a total of 750 annotations), annotating each sample in-house by a native speaker of the target language. The human annotators were asked to identify if the candidate pronoun is indeed the target pronoun and to verify that the gender prediction is correct. This way, we can account for both types of possible errors, i.e., alignment and gender extraction.

We compare the human annotations to the output of our automatic method and find that the average agreement over all languages and datasets is above 90% (see full results in App. §A). These results are comparable to the ones reported by Stanovsky et al. (2019), who conducted human validation and reported that their alignment and gender prediction of the entity in question were reliable for 85% of translations across all languages.

Some errors can be caused by idiosyncrasies that affect the morphological analysis, as Stanovsky et al. (2019) noted. For example, gender for certain words in Hebrew cannot be determined without diacritics, and some pronouns in German are used in both masculine and neutral forms (e.g., sein), or feminine and third-person plural forms (e.g., ihr). In addition, we notice that sentences from BUG, specifically in partial pro-drop languages, were found to be more challenging for the alignment model, and account for most mistakes in Hebrew and Arabic.

4 Improving MT Coreference Consistency

In the previous section we showed that the coreference performance of MT systems, obtained through an implicit signal, seems inferior to that of coreference resolution learned from an explicit signal. This result raises the question of whether we can leverage dedicated conference resolvers to improve the consistency of MT coreference.

To address this question, we propose two data augmentation techniques that leverage a source-side English coreference model, and show that fine-tuning on them indeed improves coreference resolution in MT.

Augmented fine-tuning with instances which require coreference resolution. First, we run a coreference resolution model on the source-side sentences. We then consider two approaches for constructing the augmented fine-tuning data: (1) *Coref data* with all sentences that have non-singleton clusters and (2) *Gender data*, a subset of *Coref data* where there is at least one non-singleton cluster with a gendered pronoun (he, she, her, him, hers, his). The motivation for this augmented fine-tuning strategy is that further fine-tuning on such instances would expose the MT model to examples that may bear a coreference signal.

	de	es	fr	ru	he	ar
Coref data	500K	744K	761K	149K	1.1M	1.4M
Gender data	38K	50K	51K	19K	265K	268K

Table 5: Number of fine-tuning instances in *Coref Data* (requiring some sort of coreference resolution) and *Gender data* (requiring coreference resolution with some gendered pronoun) for each target language.

Adding explicit source-side coreference markers. Second, we use the non-singleton clusters from the coreference model to add inline coreference markers in the source sentences. For our example sentence, this process produces the following source-side sequence: *"The trophy didn't fit in the **<ENT1>** suitcase **</ENT1>** because **<ENT1>** it **</ENT1>** was too small"*, indicating that "suitcase" and "it" are coreferring.

4.1 Experimental Setup

MT models. In our fine-tuning experiments, we opt for the Opus-MT model, since its size (68M parameters) and efficiency (Junczys-Dowmunt et al., 2018) enables us to run extensive experiments across many languages.

Training datasets. For fine-tuning data of Spanish, French, and German, we use Europarl (Koehn, 2005), and for Russian, Hebrew, and Arabic, we use CCMatrix (Schwenk et al., 2021), randomly sub-sampled to 5M sentences for computational reasons. In each dataset, we find instances that require coreference resolution and add appropriate markup using the *s2e* coreference resolver. We use *s2e* as it performed well in the previous experiments. Table 5 shows the size of *Coref data* and *Gender data* for all training datasets. Note that invariably *Gender data* is an order of magnitude smaller than *Coref data*.

Fine-tuning and inference. For each language, we fine-tune the Opus-MT model using four different finetuning datasets: (1) *Coref data* (2) *Coref data* with explicit coreference markers, (3) *Gender data* and (4) *Gender data* with explicit coreference markers. The inference on our three evaluation datasets (Wino-X, WinoMT, BUG) conforms with the fine-tuning procedure of each model. Namely, we run the models (1) and (3) on raw English sentences. For the models (2) and (4), we first add explicit coreference markers according to the output of the *s2e* model (a) or the gold annotation (b), then translate those augmented sentences to the different languages.

4.2 Results

Table 6 shows the target-side coreference consistency scores of all our fine-tuned models on Wino-X and WinoMT (see App. §B for the performance on BUG, which follow similar trends). For both datasets, our fine-tuned models surpass the Opus-MT baseline model, while preserving the overall translation quality, as indicated by automatic measures such as BERTScore (Zhang et al., 2020) ($+0.08\%$) and COMET-20 (Rei et al., 2020) ($+0.0025$).

Effect of augmented fine-tuning data. The models fine-tuned on *Coref data* (1) and *Gender data* (3) outperform the Opus-MT baseline for all languages, both in Wino-X and WinoMT. This demonstrates that MT models learn implicitly linguistic phenomena from instances involving those phenomena. Furthermore, we point out that consistency scores on Wino-X are generally higher when fine-tuning on *Coref data* (1, 2a, 2b) while WinoMT results are better when fine-tuning on *Gender data* (3, 4a, 4b). This performance gap likely stems from the similarity between WinoMT and *Gender data* (as both include gendered pronouns), while Wino-X's like sentences with the pronoun "it" appear only in *Coref data*. This further confirms the important role of fine-tuning data, which is in line with the observation of Saunders and Byrne (2020), that smaller, more goal-oriented data is better for fine-tuning, compared to much larger but less focused data.

Effect of explicit coreference markers. In the majority of our experiments (13/18), the explicit fine-tuning models (2a and 4a) outperform the implicit data augmentation approach when using the same augmented data (1 and 3) (see examples in Table 7). These results suggest that an explicit monolingual signal can improve results more than an implicit signal. Results also show that the improvement is more pronounced when incorporating gold coreference markers (2b and 4b) instead of predicted markers (2a and 2b). Hence, applying more accurate coreference resolution models than the *s2e* model will result in higher target-side consistency results.

4.3 Analysis

We turn to observing the empirical effect of the suggested fine-tuning strategies, using additional metrics. For each sentence in Wino-X, we have the gold target pronoun that should appear in its translation. We use it to compute pronoun translation accuracy by comparing the candidate pronoun with the gold target pronoun. Table 8 presents the results. We can see that our method provides a large improvement over the baseline. Comparing these results against those of prominent open-source and commercial MT (see App. §D) shows that our approach outperforms other MT models in German and Russian, and is only second in French.

In WinoMT, Stanovsky et al. (2019) computed *gender accuracy* as the percentage of instances in which the translation preserved the gender of the entity from the original English sentence (§2). Table 9 shows that our approach improved gender accuracy results across all languages except Arabic.

Other metrics that Stanovsky et al. (2019) used, are ΔS and ΔG. ΔS measures the difference in gender accuracy between stereotypical and non-stereotypical gender role assignments (as defined by Zhao et al., 2017), and ΔG measures the difference in performance (F1 score) between source sentences with male and female entities. Our method decreases biases in both ΔS and ΔG by 5-6 points on average, indicating that the explicit signal helps the model in associating the pronoun with the coreferring entity, even in the presence of social and gender biases.

5 Related Work

The study of coreference has a long tradition in machine translation. A long line of work uses pronoun translation as a way of measuring coreference, since BLEU-based evaluation was shown to be insufficient for measuring improvement in coreference (Hardmeier and Federico, 2010).

An alternative evaluation methodology is using automatic reference-based methods that produce a score based on word alignment between the source, reference translation, and translation output, and identification of pronouns in them, such as AutoPRF (Hardmeier and Federico, 2010) and APT (Miculicich Werlen and Popescu-Belis, 2017). Nevertheless, a later human meta-evaluation showed

	Wino-X			WinoMT					
	en→de	en→fr	en→ru	en→de	en→es	en→fr	en→ru	en→he	en→ar
Opus-MT	37.4	35.3	43.6	67.6	56.0	63.2	49.8	59.0	59.8
Coref data (1)	42.7	41.0	44.0	74.4	58.0	67.1	**62.1**	68.1	67.1
+coref markers (2a)	44.1	42.6	45.5	76.0	58.2	67.2	57.7	68.7	67.9
+gold markers (2b)	**44.7**	**45.8**	**50.5**	77.6	58.2	67.3	58.6	69.8	**68.9**
Gender data (3)	41.2	37.0	45.0	75.0	60.6	68.3	61.6	68.1	66.2
+coref markers (4a)	43.6	36.6	43.8	78.7	60.1	**68.6**	58.4	69.0	60.9
+gold markers (4b)	42.9	36.7	46.3	**80.6**	**60.8**	68.4	59.6	**69.9**	62.2

Table 6: Target-side consistency results of the Opus-MT baseline and our fine-tuning experiments on Wino-X and WinoMT when translating into different languages. For both datasets, our fine-tuned models surpass the baseline.

Source	The chef tried to store the **fat** in the **jar** but **it** was too large.	
Baseline (FR)	Le chef a essayé de stocker la graisse dans le bocal, mais il était trop grand.	✗
Ours (FR)	Le chef a essayé de stocker la graisse dans le pot, mais elle était trop grande.	✓
Source	The chickens escaped from the **yard** and fled to the **field**, as they found **it** so confining.	
Baseline (FR)	Les poulets se sont échappés de la cour et ont fui vers le champ, comme ils l'ont trouvé si restreint.	✗
Ours (FR)	Les poulets se sont échappés de la cour et ont fui vers le champ, car ils l'ont trouvée si encombrée.	✓
Source	The headphones blocked the **noise** but not the **vibration**, as **it** was relatively strong	
Baseline (RU)	Наушники блокировали шум, но не вибрацию, поскольку он был относительно сильным.	✗
Ours (RU)	Наушники блокировали шум, но не вибрацию, так как она была относительно сильной.	✓

Table 7: Translation examples of Wino-X sentences to French and Russian by the baseline (Opus-MT) and our model (with coref markers). Words in blue and red indicate male, female entities, respectively. Bold indicates coreference mentions in the source sentence.

	en→de	en→fr	en→ru
Opus-MT	39.8	31.7	37.0
Coref data	42.3	38.9	35.0
+coref markers	43.6	39.3	37.2
+gold markers	**45.7**	**42.7**	**42.6**

Table 8: Pronoun accuracy results of our fine tuning approaches on Wino-X.

	de	es	fr	ru	he	ar
Opus-MT	66	60.4	56.5	50.2	56.6	**59.8**
Gender data	73.3	66.7	**60.3**	52.4	60.9	59.7
+coref markers	**76.8**	**67.9**	60.1	**53.2**	**63.5**	55.6

Table 9: Gender Accuracy results of our fine tuning approaches on WinoMT.

substantial disagreement between these metrics and human annotators, especially because of the existence of valid alternative translations and pronouns than the ones used in the reference (Guillou and Hardmeier, 2018). Based on these conclusions, Sennrich (2017) developed a scoring-based evaluation approach that compares model scores of a predefined set of correct and incorrect translations and evaluates how often the model selects the correct option.

Our method extends (Stanovsky et al., 2019), which used a reference-free approach by aligning the source and candidate translation, but focused on entity translation accuracy to evaluate gender bias in MT models. The availability of references was assumed by most previous work (Guillou and Hardmeier, 2016; Bawden et al., 2018; Müller et al., 2018; Stojanovski et al., 2020; Emelin and Sennrich, 2021), where most of them are limited to a single language pair. The flexibility afforded by a reference-free approach allows us to evaluate any target language for which an alignment model and morphological analyzer are available. Moreover, our approach is not restricted by a predefined set of translations and can also correctly detect valid translations that are different from the reference.

Several previous methods aimed to improve the coreference abilities of MT models and reduce undesirable biases, by modifying the training data in ways that share some similarities with our method. Vanmassenhove et al. (2018) incorporate a "speaker gender" tag into training data, allowing gender to be conveyed at the sentence level. Similarly, Moryossef et al. (2019) added a prefix to disambiguate the coreference in the sentence. Stojanovski and Fraser (2018) used oracle-based approach to inject new tokens indicating the pronoun translation and its gender into the source sentence. Our method is novel in the way it enriches the data with coreference signal using only the source-side signal, and thus requires only an English coreference resolution model without the need for coreference annotation in the target language.

6 Conclusion

Our work is the first to present an automatic methodology for assessing the coreference capabilities of MT models, that can be applied in any target language and does not require any target side annotations. Furthermore, to the best of our knowledge, we are the first to conduct a large-scale multilingual coreference evaluation study on prominent open-source and commercial MT models, and compare them against state-of-the-art coreference resolvers on three challenging benchmarks. Finally, based on the superior results of coreference resolvers, we propose a novel approach to improve the coreference capabilities of MT models, that outperforms or achieves comparable results to strong and larger MT models. Despite this substantial gain, there is still a performance gap between our model and state-of-the-art coreference resolvers. We hope that our work, and specifically our automatic evaluation methodology, will encourage future research to improve the coreference capabilities of MT models.

Future work can expand our approach to account for number and person agreement phenomena, investigate how to extend our approach to more coreference clusters and more mentions per cluster in intra-sentential as well as inter-sentential settings. Moreover, we intend to investigate how different morphological attributes affect MT models' coreference abilities.

Limitations

Even though our study presents the first large-scale multilingual coreference evaluation study in MT, it still has some limitations that could be addressed in future work. First, our methodology provides an upper bound to the coreference capabilities based on detecting gender valuations. While this could allow for a controlled evaluation experiment, this upper bound can become non-indicative in cases where gender assignment is not a discriminative factor. This can be addressed by accounting for more semantic and syntactic constraints that the translation needs to follow (e.g., singular/plural agreement).

Second, our setting addresses one entity and a single co-referring pronoun in the naturalistic sentences experiment. Our methodology could in principle be augmented to deal with more coreference clusters and mentions per cluster. Another possible extension is to include event coreference in addition to entity coreference. For example, in this work, we focus only on the anaphoric function of the pronoun "it" but further research can also examine the event function of "it" (Loáiciga et al., 2017).

Third, MT models should generally produce translations with accurate gender inflection for all words. However, in this work, we focus on the *coreference* capabilities of MT models by evaluating gender agreement between coreferring *entity* mentions. Future research can extend our evaluation methodology to assess the gender inflection of verb and adjective translation (e.g., the gender of "big" and "small" in Figure 1), using additional tools and resources such as a semantic role labeling model and a dependency parser.

Finally, although in Section 4 we show big gains from the fine-tuning approach, it is clear that there is much room for improving the coreference capabilities of MT models, especially with regard to the performance of state-of-the-art coreference resolvers. We hope this work will help others develop MT models with better coreference capabilities.

Acknowledgements

We thank the reviewers for their insightful comments and suggestions. This work was partially supported by the Israeli Ministry of Science and Technology (grant no. 2088). Arie Cattan is partially supported by the PBC fellowship for outstanding PhD candidates in data science.

References

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya

Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. *ArXiv*, abs/1908.05739.

Denis Emelin and Rico Sennrich. 2021. Wino-X: Multilingual Winograd schemas for commonsense reasoning and coreference resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8517–8532, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Çelebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2021. Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, 22:107:1–107:48.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi kiu Lo, Craig Stewart, Eleftherios Avramidis, Tom Kocmi, George Foster, Alon Lavie, and André FT Martins. 2022. Results of wmt22 metrics shared task: Stop using bleu–neural metrics are better and more robust. In *Proceedings of the Seventh Conference on Machine Translation, Abu Dhabi. Association for Computational Linguistics*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew E. Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *ArXiv*, abs/1803.07640.

Liane Guillou and Christian Hardmeier. 2016. PROTEST: A test suite for evaluating pronouns in machine translation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 636–643, Portorož, Slovenia. European Language Resources Association (ELRA).

Liane Guillou and Christian Hardmeier. 2018. Automatic reference-based evaluation of pronoun translation misses the point. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4797–4802, Brussels, Belgium. Association for Computational Linguistics.

Christian Hardmeier and Marcello Federico. 2010. Modelling pronominal anaphora in statistical machine translation. In *Proceedings of the 7th International Workshop on Spoken Language Translation: Papers*, pages 283–289, Paris, France.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. *ArXiv*, abs/1909.00277.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference resolution without span representations. In *ACL*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MTSUMMIT*.

Ronan Le Nagard and Philipp Koehn. 2010. Aiding pronoun translation with co-reference resolution. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 252–261, Uppsala, Sweden. Association for Computational Linguistics.

Hector J. Levesque, Ernest Davis, and L. Morgenstern. 2011. The winograd schema challenge. In *KR*.

Shahar Levy, Koren Lazar, and Gabriel Stanovsky. 2021. Collecting a large-scale gender bias dataset for coreference resolution and machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2470–2480, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pretraining for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Sharid Loáiciga, Liane Guillou, and Christian Hardmeier. 2017. What is it? disambiguating the different readings of the pronoun 'it'. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1325–1331, Copenhagen, Denmark. Association for Computational Linguistics.

Lesly Miculicich Werlen and Andrei Popescu-Belis. 2017. Validation of an automatic metric for the accuracy of pronoun translation (APT). In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 17–25, Copenhagen, Denmark. Association for Computational Linguistics.

Amit Moryossef, Roee Aharoni, and Yoav Goldberg. 2019. Filling gender & number gaps in neural machine translation with black-box context injection. In *Proceedings of the First Workshop on Gender Bias* *in Natural Language Processing*, pages 49–54, Florence, Italy. Association for Computational Linguistics.

Mathias Müller, Annette Rios, Elena Voita, and Rico Sennrich. 2018. A large-scale test set for the evaluation of context-aware pronoun translation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 61–72, Brussels, Belgium. Association for Computational Linguistics.

Shon Otmazgin, Arie Cattan, and Yoav Goldberg. 2022. Lingmess: Linguistically informed multi expert scorers for coreference resolution. *ArXiv*, abs/2205.12644.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. *ArXiv*, abs/1907.10641.

Danielle Saunders and Bill Byrne. 2020. Reducing gender bias in neural machine translation as a domain adaptation problem. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7724–7736, Online. Association for Computational Linguistics.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, and Armand Joulin. 2021. Ccmatrix: Mining billions of high-quality parallel sentences on the web. In *ACL*.

Rico Sennrich. 2017. How grammatical is character-level neural machine translation? assessing MT quality with contrastive translation pairs. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 376–382, Valencia, Spain. Association for Computational Linguistics.

Gabriel Stanovsky, Noah A. Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684, Florence, Italy. Association for Computational Linguistics.

Gabriel Stanovsky and Ronen Tamari. 2019. Y'all should read this! identifying plurality in second-person personal pronouns in english texts. In *EMNLP*.

Dario Stojanovski and Alexander Fraser. 2018. Coreference and coherence in neural machine translation: A study using oracle experiments. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 49–60, Brussels, Belgium. Association for Computational Linguistics.

Dario Stojanovski, Benno Krojer, Denis Peskov, and Alexander Fraser. 2020. ContraCAT: Contrastive coreference analytical templates for machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4732–4749, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal. European Association for Machine Translation.

Eva Vanmassenhove, Christian Hardmeier, and Andy Way. 2018. Getting gender right in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3003–3008, Brussels, Belgium. Association for Computational Linguistics.

Kellie Webster and Emily Pitler. 2020. Scalable cross lingual pivots to model pronoun gender for translation. *ArXiv*, abs/2006.08881.

Walt Wolfram. 1985. Variability in tense marking: A case for the obvious. *Language Learning*, 35(2):229–253.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020,* *Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2989, Copenhagen, Denmark. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

	he	ar	es	ru	fr	de
alignment error	14	19	4	4	4	0
gender prediction error	7	7	0	7	2	2
correct annotation	79	74	96	139	144	148
Total # of annotation	100	100	100	150	150	150

Table 10: Human validation results on our three evaluation datasets and six target languages.

A Human Validation Results

Table 10 shows the complete human annotations results. The results indicate that alignment and gender prediction are accurate in most languages. In Arabic and Hebrew, the alignment error occurs more. A possible explanation for that can be the fact that both those languages are partial pro-drop languages. To verify that those results will not affect our measurement, we verified that the error has similar consistency distributions as the rest of our results.

B BUG Consistency Results

Table 11 presents the target-side consistency results of the Opus-MT baseline model and all our fine tuning approach on BUG. Similarly to Wino-X and WinoMT, our fine-tuned models outperform the baseline.

C Omitted Cases

Table 12 shows translation examples from English to French and Spanish that demonstrate unique features in each language. The first example shows a French translation that contains a possessive pronoun, which does not indicate the gender of the possessor. The second example shows a Spanish translation where the pronoun is omitted. In both cases, we can obtain a correct translation without information concerning the aligned pronoun gender, we therefore exclude them from the evaluation.

D Pronoun Translation Accuracy

Table 13 shows pronoun accuracy results of our baselines on Wino-X. We can notice those results are similar to our consistency results although our methodology does not use any annotated target data. Moreover, those results clearly show that MT models struggle with translating sentences that demand solving the source side coreference resolution.

	en→de	en→es	en→fr	en→ru	en→he	en→ar
Opus-MT	79.8	85.1	86.2	67.3	88.8	88.0
gender data	84.1	**87.4**	88.7	69.9	**90.2**	91.3
+coref markers	**84.2**	**87.4**	**88.8**	**70.1**	89.8	**91.8**

Table 11: Target-side consistency results of our implicit and explicit fine tuning approaches on BUG.

Source	[Target lang.] Predicted translation	Phenomenon
The developer argued with the designer because his idea cannot be implemented.	[FR.] Le développeur a argumenté avec le concepteur parce que son idée ne peut pas être mis en œuvre.	"son" is male because the possessed noun ("idée") is male.
The doctor asked the nurse to help her in the procedure	[ES.] El doctor le pidio a la enfermera que le ayudara con el procedimiento	In Spanish, the pronoun "her" is dropped in the translation.

Table 12: Examples of omitted sentences from our evaluation datasets and their translations. Words in blue and red indicate male and female entities, respectively.

	German	French	Russian
mBART50	38.1	**47.2**	34.7
M2M00_418M	33.8	39.5	32.4
M2M100_1.2B	38.5	39.0	33.7
Google	**44.2**	34.8	35.8
Microsoft	43.1	35.7	**37.8**
EasyNMT	39.9	31.8	37.0

Table 13: Pronoun accuracy results of commercial and open-source MT models on Wino-X.

E Computing Infrastructure

We fine tuned our models using 4 NVIDIA GTX Titan Black GPUs. The run time of the models varies between one hour to 24 hours depending on dataset size.

Document-Level Planning for Text Simplification

Liam Cripwell
Université de Lorraine
CNRS/LORIA
liam.cripwell@loria.fr

Joël Legrand
Université de Lorraine
Centrale Supélec
CNRS/LORIA
joel.legrand@inria.fr

Claire Gardent
CNRS/LORIA
Université de Lorraine
claire.gardent@loria.fr

Abstract

Most existing work on text simplification is limited to sentence-level inputs, with attempts to iteratively apply these approaches to document-level simplification failing to coherently preserve the discourse structure of the document. We hypothesise that by providing a high-level view of the target document, a simplification plan might help to guide generation. Building upon previous work on controlled, sentence-level simplification, we view a plan as a sequence of labels, each describing one of four sentence-level simplification operations (copy, rephrase, split, or delete). We propose a planning model that labels each sentence in the input document while considering both its context (a window of surrounding sentences) and its internal structure (a token-level representation). Experiments on two simplification benchmarks (Newsela-auto and Wiki-auto) show that our model outperforms strong baselines both on the planning task and when used to guide document-level simplification models.

1 Introduction

Text simplification aims to transform a given text into a simpler version of itself that preserves the core meaning such that it can be better understood by a wider audience (Gooding, 2022). Simplification has also been shown to be a useful preprocessing step for downstream NLP tasks such as relation extraction (Miwa et al., 2010; Niklaus et al., 2016) and machine translation (Chandrasekar et al., 1996; Mishra et al., 2014; Li and Nenkova, 2015; Štajner and Popovic, 2016).

Previous research has mostly considered the simplification of isolated sentences. Much work has focused on training a statistical or a neural model on pairs of complex and simplified sentences assuming that such models will learn to perform simplification operations (e.g. sentence splitting, lexical simplification or syntactic rephrasing) implicitly from the inductive bias present in the training data (Zhang and Lapata, 2017; Nisioi et al., 2017; Jiang et al., 2020). However, because the training data is obtained using distant supervision techniques and is often imbalanced in terms of simplification operations (many of which occur infrequently (Jiang et al., 2020)), system outputs have been found to be overly conservative, often making no changes or being limited to the paraphrasing of short word sequences (Alva-Manchego et al., 2017; Maddela et al., 2021). In addition, these systems provide limited capacity for controllability and are unable to express alternative variants of the simplified text (Alva-Manchego et al., 2017; Cripwell et al., 2021).

In response, controllable simplification systems have been proposed which either constrain attributes of the output (length, amount of paraphrasing, lexical and syntactic complexity) (Martin et al., 2020) or explicitly specify which simplification operation to perform (Alva-Manchego et al., 2017; Dong et al., 2019; Malmi et al., 2019; Scarton et al., 2020; Maddela et al., 2021; Cripwell et al., 2022).

To guide the simplification of full documents, we combine the power of data-driven neural generative models with strategies from controllable simplification. Our hypothesis is that document-level simplification can be facilitated by a plan specifying how each complex input sentence should be transformed to yield a simplified version of that document - should it be copied, deleted, split or rewritten?

We make the following contributions: We present a model for predicting document simplification plans which leverages both the context of sentences and their internal structure (the words they consists of). We create the data necessary to train this model by labelling complex sentences in simplification corpora with the simplification operation that relates it to the corresponding simplified sentence. We compare our planning model with several alternative neural architectures and we briefly examine the impact of planning on document simplification.

Experiments on two simplification benchmarks (Newsela-auto and Wiki-auto) show that our model outperforms strong baselines both on the planning task and when used to guide document-level simplification models. [1]

2 Related Work

Document-Level Simplification. There is limited existing work on document-level text simplification. Early attempts largely applied sentence-level techniques iteratively over a document (Woodsend and Lapata, 2011a; Alva-Manchego et al., 2019b). However, this is generally viewed as insufficient for certain operations and maintaining the discourse coherence of the document (Siddharthan, 2003; Alva-Manchego et al., 2019b).

There are several works that address subproblems of simplification that only consider a limited set of operations, like paraphrasing and sentence re-ordering (Lin et al., 2021), insertion (Srikanth and Li, 2021) or deletion (Zhong et al., 2020; Zhang et al., 2022). Others fully address simplification but only extend inputs to the level of paragraphs without clearly differentiating the problem from the sentence-level (Laban et al., 2021; Devaraj et al., 2021).

Recently, Sun et al. (2020) proposed a sentence-level model (SUC) that uses an encoding of surrounding sentences as context information to influence the simplification. They use two extra encoders to build a representation of the two preceding and two following sentences, which are attended over in their encoder-decoder generative model. However, when applied to the document-level task, their system was unable to outperform any baseline systems (Sun et al., 2021).

Operation Prediction. Revision-based simplification models learn to predict edit operations to apply at the token-level rather than generating the entire simplification from scratch (Alva-Manchego et al., 2017; Dong et al., 2019; Kumar et al., 2020; Omelianchuk et al., 2021; Dehghan et al., 2022). This has the benefit of providing more control and interpretability over generative approaches, often at the cost of the ability to perform major structural changes. It also allows some systems to leverage non-autoregressive generation strategies, resulting in faster inference times (Malmi et al., 2019; Omelianchuk et al., 2021).

Some works have attempted to predict rewrite operations at the sentence-level. Applying a binary classifier to predict whether simplification should be performed has been found to improve SARI results, reducing conservatism and spurious transformations (Scarton et al., 2020; Garbacea et al., 2021). Others have proposed multi-class systems to predict sentence-level operations that are then used to condition a generative model (Scarton and Specia, 2018; Scarton et al., 2020; Cripwell et al., 2022). These show some capacity for general improvement over end-to-end systems, while also significantly improving performance for specific operations (e.g. splitting in the case of Cripwell et al. (2022)).

At the document-level, there has been limited interest to date. However, there are recent works specifically looking at predicting sentence deletions (Zhong et al., 2020; Zhang et al., 2022). Both of these use features of the discourse structure from surrounding sentences to identify likely deletion candidates.

We bring all of these methods together by proposing a system that uses both sentence and document-level information to predict a multi-class, sentence-level operation plan over an entire document.

3 Problem Formulation

Let C denote an English language document. The aim of document-level simplification is to produce a text S that simplifies the input document C.

As a plan can provide a high-level view of a document, we hypothesize that a document-level simplification model that is based on a plan specifying a simplification operation for each input sentence should fare better than a simplification model that directly simplifies an entire document.

We therefore decompose simplification into a two-stage generation process:

$$p(S \mid C) = p(S \mid C, P)p(P \mid C)$$

where input document $C = c_1 \dots c_n$ is a sequence of complex sentences, $S = s_1 \dots s_k$ is a sequence of simplified sentences and $P = o_1 \dots o_n$ is a sequence of sentence-level simplification operations for C.

---

[1] Pretrained models, code, and data are available at `https://github.com/liamcripwell/plan_simp`.

We consider three simplification operations proposed in previous work on sentence simplification (*copy*, *rephrase*, and *split*) to which we add *delete*, an operation that is needed to account for the fact that, contrary to sentence simplification, document-level simplification can require for a sentence present in the input document to be excluded from the resulting simplified document.

Given the input document C, the first-stage model aims to predict the sequence of simplification operations P that should be applied to each individual sentence in that document. The second-stage model generates the output simplified document S conditioned on the input document C and its accompanying simplification plan P.

In this work, we focus on the planning stage, comparing different architectures and demonstrating the impact of planning on three possible document-level simplification models. We leave the exploration of alternative, more complex architectures for the simplification stage to future work.

	Wiki-auto	Newsela-auto
# Doc Pairs	85,123	18,319
# Sent Pairs	461,852	707,776
Avg. $\lvert C\rvert$	155.51	868.98
Avg. $\lvert S\rvert$	97.72	674.94
Avg. $\lvert c_i\rvert$	28.64	22.49
Avg. $\lvert s_i\rvert$	21.57	15.84
Avg. n	5.43	38.64
Avg. k	4.53	42.60

Table 1: Statistics of each dataset after preprocessing, where n is # sentences in C and k is # sentences in S.

4 Data

In this section we introduce the datasets used, explain how annotation is performed for each complex sentence and describe other preprocessing steps.

Dataset. For all experiments, we utilise Wiki-auto and Newsela-auto (Jiang et al., 2020), two datasets of English documents paired with their simplification. These datasets were derived from WikiLarge (Zhang and Lapata, 2017) and Newsela (Xu et al., 2015) by aligning the input document with the output simplification at both the sentence and the paragraph level.

WikiLarge gathers three simplification datasets which were automatically-collated from English Wikipedia and Wikipedia simple (Zhu et al., 2010; Woodsend and Lapata, 2011b; Kauchak, 2013).

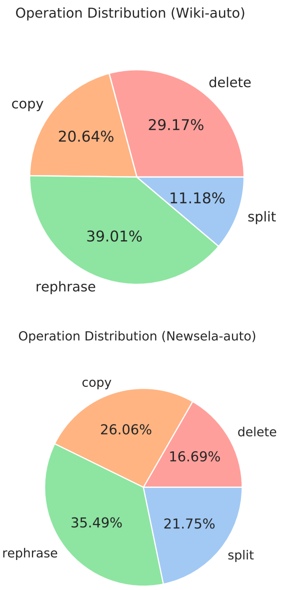


Figure 1: Operation class distributions for Wiki-auto (top) and Newsela-auto (bottom) datasets.

Newsela consists of news articles, each manually rewritten at five different levels of simplification, corresponding to discrete reading levels (0-4) of increasingly simplicity. Aligned pairs are created by pairing every article version with each other version corresponding to a higher reading level. Because of this, there can be up to four aligned document pairs that contain the same document as either the input or the output.

The types of operations present in different reading level pairings differs significantly, with adjacent level transitions being extremely conservative (no instances of deletion throughout entire dataset). To mitigate any issues arising from this, all models we train with Newsela-auto receive a control-token at the start of the input which specifies the target reading level.

We do not use the D-Wikipedia dataset from Sun et al. (2021) as it does not contain sentence/paragraph alignments and is poorly formatted. In particular, all text is lower-cased and pretokenized in a way that makes it difficult to accurately parse sentences. There are also regular formatting issues at points where references exist in the source article.

Annotating Complex Sentences. Using the pairs (c_i, s_j) of complex and simplified sentences available in Wiki-auto and Newsela-auto, we heuristically assign a silver simplification operation label to each complex sentence c_i in these two datasets as follows:

Delete: c_i is not aligned to any s_j.

Copy: c_i is aligned to a single s_j with a Levenshtein similarity above 0.92.

Rephrase: c_i is aligned to a single s_j with a Levenshtein similarity below 0.92.

Split: c_i is aligned to multiple s_js.

Preprocessing. Wiki-auto contains many document pairs with wildly different sizes. We therefore clip all complex documents after the last aligned paragraph. Many simple articles resemble a summarization, rather than a simplification of the complex article (lots of deletion, often consisting of about one sentence from each paragraph in C). Because of this, we also remove documents where more than 50% of aligned sentences are labelled as *delete*. Finally, we remove all articles that exceed 1024 tokens (so that we can fit them into a BART baseline generative model).

For Newsela-auto, article pairs are much more even in length as they are manually created to be gradual, direct simplifications of each other. We perform the same length-based filtering to exclude documents that will not fit into a baseline generative model.

Train/Dev/Test Split. For both datasets we use a train/validation/test split of 92.5/2.5/5. This is applied at the document-level so that sentences from the same document will not exist across different sets. For Newsela, this means that all reading level versions of a single article will exist within the same set.

Table 1 and Figure 1 give some statistics and a graphical description of the two datasets after pre-processing.

5 Planning

We present our model and four alternative models we explored for comparison. Training details are given in Appendix A.

5.1 Model (Contextual Classifier)

Given some input document $C = c_1 \dots c_n$ consisting of n complex sentences c_i, the task of the planner is to predict a sequence $\hat{P} = \hat{o}_1 \dots \hat{o}_n$ of n simplification operations with $\hat{o}_i \in \{$*copy, rephrase, split, delete*$\}$.

One challenge with this is that the operations have different, sometimes conflicting requirements.

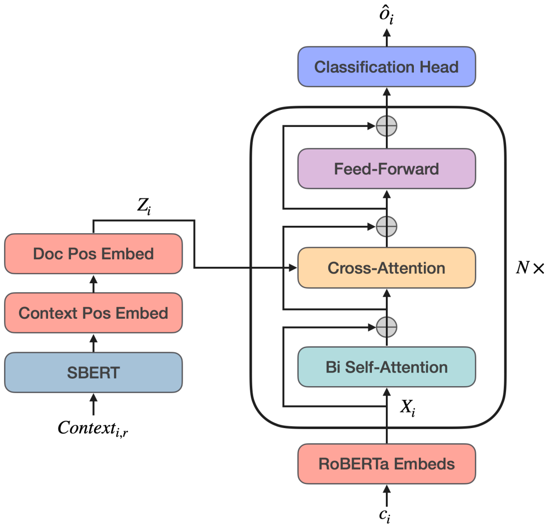
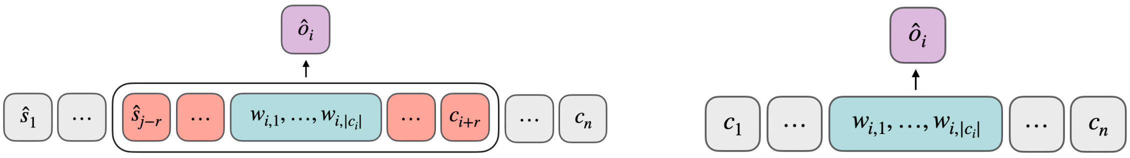


Figure 2: Contextual classifier model architecture.

By construction, splitting is mostly context independent as it is mainly determined by the input sentence's internal structure: a sentence will be split only if it has the appropriate syntactic (e.g., The man who sleeps snores → The man sleeps. He snores.) or discourse (e.g., John went shopping after he left work → John left work. Afterwards he went shopping.) structure. For sentence splitting, context (the other sentences in the input document) has little impact.

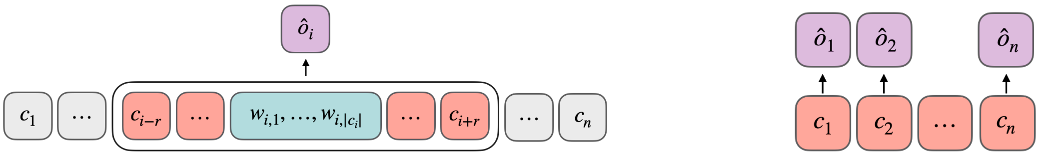
In contrast, deletion and to a lesser extent, copy and rephrase are mostly context dependent. Intuitively, a sentence can only be omitted in the simplified text in cases where it is either redundant with, or of minor semantic import relative to, other sentences in the document. That is, while for splitting, internal sentence structure is the key factor, for deletion, it is the semantics of the input sentence and how it relates to that of the other sentences which matters most.

We model these different requirements by using a token level encoder for the target document sentence c_i (the input sentence to be labelled with a simplification operation) and a sentence level representation of the context where each $c_p \in c_1 \dots c_{i-1}, c_{i+1} \dots c_n$ is represented by a sentence level embedding using SBERT. In this way both the internal structural information needed to capture splitting operations and the contextual information required by the other operations are provided. Specifically, we propose a model for planning that combines a classifier with cross-attention over the (dynamic or static) context and two types of positional embeddings. Figure 2 illustrates our model



(a) Dynamic Contextual Classifier

(b) Classifier



(c) Contextual Classifier

(d) Tagger

(e) Tagger+Dec

(f) EncDec$_{\text{full}}$

Figure 3: Visualisation of the inputs/outputs of the various models, where $w_{i,t}$ is the tth token in c_i, n is the no. sentences in C and m is the no. tokens in C. Sentence-level representations are shown in red, token-level representations in teal, operation labels in pink, and unused parts of C in grey.

architecture.

Classifier with Cross-attention over the Context. We build upon a RoBERTa classifier architecture to enable conditioning upon the surrounding sentences in the document. We do this by inserting an additional cross-attention layer between the self-attention and the feed-forward layer of each transformer block, allowing the model to attend to a latent representation of the surrounding sentences, Z_i.

Context Representation. To obtain Z_i, we take a fixed window of radius r, extract the r sentences on either side of the target sentence to be simplified and concatenate the representation of each of these sentences. Each context sentence is encoded with the pretrained Sentence-BERT (SBERT) model[2] (Reimers and Gurevych, 2019) and combined with custom learnt positional embeddings.[3]

To better simulate autoregressive inference, we consider a strategy where the left context consists of previously simplified sentences, rather than complex ones. We refer to this as *dynamic context*. At training time, we use the ground truth simplifications

$$Context_{i,r} = Concat(s_{j-r..j-1}, c_{i..i+r}) \quad (1)$$

where $j \in \{1,\dots,|S|\}$ is the index of the first sentence aligned to c_i in the simple document S. During inference, the simplifications generated at preceding timesteps $\hat{s}_{j-r..j-1}$ are used.

Positional Embeddings. We use custom positional embeddings to encode both information about document, and relative context-window positions. These are each handled by a dedicated embedding layer and added to the representations of the corresponding context sentence.

Document positional embedding indices are simply the document quintile (1-5) that a given sentence falls into. We use quintiles as this will ensure that all indices are encountered within the input document. The context positional embedding indices are the relative distance of a given sentence from the input sentence c_i, adjusted to be within $\mathbb{N}_0$: $ContextPosIdxs = \{p - i + r \mid p \in \{i - r, \dots, i + r\}\}$.

Initialisation. Given that the cross-attention layers must be trained from scratch, the start of training can see a lot of instability in the model, potentially making it more difficult to model context-independent features of the input sentence. To account for this, we initialise the RoBERTa layers with weights from a context-independent classifier.

5.2 Alternative Models

We compare our model with four alternative models. The different inputs/outputs of the models are illustrated in Figure 3.

Classifier. We fine-tune pretrained `RoBERTa-base` (Liu et al., 2019), which has 12 hidden layers and a hidden size of 768, and

---

[2] Specifically, all-mpnet-base-v2.

[3] At training time, we backpropogate to the positional embedding layers but keep the SBERT weights frozen.

	Wiki-auto						Newsela-auto					
Model	C	R	S	D	Micro	**Macro**	C	R	S	D	Micro	**Macro**
$\text{EncDec}_{\text{full}}$	26.9	42.2	36.0	51.8	43.2	40.8	26.1	10.8	11.7	9.0	12.2	11.5
Tagger+Dec	29.3	54.5	30.0	51.8	47.7	41.4	72.2	73.9	75.9	79.7	75.0	75.4
Tagger	38.6	54.2	31.7	**58.5**	50.6	45.8	71.4	72.7	74.1	78.4	73.7	74.1
Classifier	42.1	52.9	42.6	49.0	48.4	46.7	77.0	75.6	80.0	78.5	77.4	77.8
Dyn. Context	**44.8**	**57.9**	42.4	54.8	**52.8**	**50.0**	79.3	77.3	82.8	81.4	79.7	80.2
+ docpos	43.7	55.4	**43.6**	56.7	52.3	49.9	**80.0**	**78.1**	**83.6**	**82.0**	**80.3**	**80.8**

Table 2: Planning Accuracy. **Dyn. Context** is the contextual classifier described in Section 5.1 with $r = 13$, dynamic context and weights initialised using the classifier weights (C: Copy, R: Rephrase, S: Split, D: Delete).

add a pooled classification head which takes the final layer [CLS] representation as input. Given an input sentence c_i, the model simply takes the tokenized sentence as input and outputs a prediction score for each operation class. The model is applied from left-to-right on the input document classifying each sentence in turn. Thus, in this approach, while the model has access to the tokens of the sentence to be classified, there is no notion of context which, intuitively, should be detrimental in particular for deletions.

Tagger. We consider a model that frames the problem as a sequence tagging task over the full document, predicting the entirety of $\hat{P}$ at once. Each c_i is encoded using the same SBERT model as the contextual classifier, with the input document C therefore being represented as a sequence of sentence embeddings. In contrast to the classifier, the tagger makes predictions based both on the input sentence to be classified and on the context. However, because the input representation at each index is for an entire sentence we lose some resolution with respect to token-level content. The approach is thus less adapted for splitting.

Tagger+Dec. We also consider an autoregressive variant of the tagger that better models the dependencies between predicted tags. Here, we include a 1-layer decoder and condition each prediction both on the input document and on the previously predicted operation tags for the earlier sentences. This approach is somewhat similar to Dong et al. (2019); Malmi et al. (2019), except we abstract to the document-level and do not require explicit realisation, as this will be handled downstream by the simplication model.

$\text{EncDec}_{\text{full}}$. Finally, we experiment with an encoder-decoder variant that conditions on a token-level representation of the input, thereby combining a global view and a token-level representation of the input document. We use sentence separator tokens to delimit each sentence in the input document.

5.3 Evaluation Metrics

To evaluate the performance of the various planners we use F1-score, considering each individual prediction at the sentence-level. We report the F1 for each operation class as well as both the micro and macro averages. The micro F1 weights all examples equally, whereas the macro re-weights examples such that each class is represented equally in the final score. Given the class imbalances in the data, we regard macro F1 as our primary metric.

5.4 Results

Table 2 summarizes the results.

Compared to the various baselines, our model consistently shows best results on both datasets. The improvement over the context-free classifier is slightly less on Newsela-auto however. We conjecture that the much larger dataset and additional guidance provided by the reading levels allows the classifier to achieve rather high accuracy without document-level context. We also note that the context-free classifier is markedly outperformed by other models with respect to *delete*, which confirms the intuition that context modeling particularly matters for this operation.

Of the four baselines, $\text{EncDec}_{\text{full}}$ performs worst presumably because the very long input (the whole context is modelled at the token level) challenges the attention mechanism which tends to become blurry as the length of the input increases. This is particularly apparent on the longer Newsela documents.

The tagger models, which both use sentence-level encodings of the complex document, perform

Model	Copy	Rephrase	Split	Delete	Micro	**Macro**
(a) Ablation on Best Model						
Dyn, $r = 13$, +init, +docpos	80.0	78.1	83.6	82.0	80.3	80.8
-docpos	79.3	77.3	82.8	81.4	79.7	80.2
-init	74.9	72.1	77.8	75.2	74.6	75.0
-init, -docpos	75.6	72.0	77.7	77.1	75.1	75.6
(b) Dynamic vs. Static Context						
Stat, $r = 9$	71.3	69.5	75.4	73.3	72.0	72.4
Stat, $r = 13$	72.2	65.3	69.9	68.3	68.5	68.9
Dyn, $r = 9$	73.1	70.1	75.5	75.9	73.1	73.6
Dyn, $r = 13$	75.6	72.0	77.7	77.1	75.1	75.6
(c) With vs without Initialisation						
Dyn, $r = 9$	73.1	70.1	75.5	75.9	73.1	73.6
Dyn, $r = 9$ +init	79.3	78.0	82.7	79.8	79.7	80.0
Dyn, $r = 13$	75.6	72.0	77.7	77.1	75.1	75.6
Dyn, $r = 13$ +init	79.3	77.3	82.8	81.4	79.7	80.2
(d) Window Size						
Stat, $r = 9$	71.3	69.5	75.4	73.3	72.0	72.4
Stat, $r = 13$	72.2	65.3	69.9	68.3	68.5	68.9
Dyn, $r = 9$	73.1	70.1	75.5	75.9	73.1	73.6
Dyn, $r = 13$	75.6	72.0	77.7	77.1	75.1	75.6
Dyn, $r = 9$ +docpos	73.8	72.9	77.2	75.8	74.6	74.9
Dyn, $r = 13$ +docpos	74.9	72.1	77.8	75.2	74.6	75.0
Dyn, $r = 9$ +init +docpos	79.4	78.0	83.1	82.0	80.1	80.6
Dyn, $r = 13$ +init +docpos	80.0	78.1	83.6	82.0	80.3	80.8

Table 3: Ablations on Newsela-auto TestSet.

worse than the classifier. This highlights the importance of having a token-level modeling of the input sentences.

We observe a strong difference in terms of absolute scores between the two datasets. This is likely a result of Wiki-auto being an inferior simplification corpus (discussed in Section 4).

Next, we examine the impact of our modeling choices using ablation (Table 3) and focusing on the higher-quality, Newsela-auto dataset. Our best model is one with dynamic left-context, a context radius of 13, document position embeddings and weight initialisation. We see (Sub-table **a**) that each of these components help improve performance (document position appears less important with a larger context window). Sub-tables **b-d** show that using a dynamic rather than a static context increases results by up to +6.7 Macro F1, while increasing the context radius from 9 to 13 sentences mostly improves performance when dynamic context is used. Using document positional embeddings also generally improves results (Sub-table **d**).

6 Simplification

To assess whether document plans can help improve simplification models, we experiment with two simple document-level simplification models and compare their performance with and without a preceding planning step.

6.1 Simplification Models

All models use the BART model (Lewis et al., 2020) fine-tuned on aligned text pairs. [4]

We consider two variants for document-level simplification: (i) **Doc-BART**, which is finetuned on full document pairs; and (ii) **Sent-BART** which is finetuned on sentence pairs and iteratively applied to each input sentence at test time.

We compare these to various plan-guided (**PG**) systems whereby one of our planners predicts an $\hat{o}_i$ for each c_i and is given as a control-token to a sentence-level BART simplification model. In the case of the dynamic planner, $\hat{o}_i$ is predicted based on the sequence of previously simplified sentences $\hat{s}_{i-r} \dots \hat{s}_{i-1}$.

Training details are given in Appendix B.

6.2 Evaluation

To measure meaning preservation and fluency, we use BARTScore (Yuan et al., 2021), a state-of-the-

---

[4] We use the pretrained *facebook/bart-base* model from `https://huggingface.co/facebook/bart-base`.

System	**BARTScore ↑**				**SMART ↑**			**FKGL ↓**	**SARI ↑**	**Length**	
	Faith. $(s \rightarrow h)$	P $(r \rightarrow h)$	R $(h \rightarrow r)$	F1	P	R	F1			Tokens	Sents
Input	-0.93	-2.47	-1.99	-2.23	63.2	62.7	62.8	8.44	20.52	866.9	38.6
Reference	-1.99	-0.93	-0.93	-0.93	100	100	100	4.93	99.99	671.5	42.6
Doc-BART	-2.48	-2.68	-2.76	-2.72	61.9	43.9	50.6	10.01	47.07	600.8	20.7
Sent-BART	**-1.86**	-1.63	-1.56	-1.60	78.9	80.1	79.3	5.03	73.02	666.4	42.6
PG$_{\text{Tag}}$	-1.95	-2.22	-2.18	-2.20	62.0	62.6	61.6	5.07	56.13	657.4	41.8
PG$_{\text{Tag+Dec}}$	-1.94	-2.22	-2.18	-2.20	62.2	62.5	61.6	5.09	56.06	654.2	41.4
PG$_{\text{Clf}}$	-1.91	-1.68	**-1.53**	-1.60	77.8	**81.2**	79.3	**4.95**	73.83	688.8	44.5
PG$_{\text{Dyn}}$	-1.91	**-1.60**	-1.54	**-1.57**	**80.2**	81.0	**80.5**	4.98	**75.00**	667.2	42.6
PG$_{\text{Oracle}}$	-1.93	**-1.39**	**-1.40**	**-1.40**	**85.5**	**85.0**	**85.3**	**4.91**	**80.74**	655.6	42.1

Table 4: Results of document simplification systems on Newsela-auto. For BARTScore, s is the source, h is the hypothesis, and r is the reference.

art summarization metric that has proved effective on many other text generation tasks. We also compute SMART (Amplayo et al., 2022), a new metric that considers sentences as the primary unit of comparison. It was shown to be highly effective for document summarization and does not use any neural model, making it very fast to compute (we use the SMARTL+CHRF version). We cannot use other model-based metrics, such as BERTScore or QuestEval, as these do not support texts longer than 512 tokens.

To assess simplicity, we use the Flesch-Kincaid grade level (FKGL), a document-level metric used to measure text readability, which has been found to have the highest correlation with simplicity measures of human-written simplifications (Scialom et al., 2021). We also report the popular SARI (Xu et al., 2016). The EASSE python library (Alva-Manchego et al., 2019a) is used for calculation of FKGL and SARI. We include results for other popular metrics in Appendix D.

At test time we generate sequences using beam search with a beam size of 5 and a maximum length of 1024 tokens. We enforce a minimum length for Doc-BART, which is tuned on the validation set.

We do not conduct a human evaluation as we intend the focus of this work to be on the planning component and include simplification results only to confirm its efficacy. We leave a more in-depth investigation of the interaction between planning and document-level simplification to future work.

6.3 Results

Results can be seen in Table 4.

PG$_{\text{Dyn}}$ achieves the highest results of all systems. Using the silver operation labels (PG$_{\text{Oracle}}$) leads to a substantial further increase in performance across every metric, highlighting the impact of planning and pointing to the possibility of further improvements to be made.

Using either PG$_{\text{Dyn}}$ or PG$_{\text{Clf}}$ yields generally better results than Sent-BART. Both systems achieve better FKGL and SARI, suggesting greater output simplicity. Sent-BART achieves much higher source-oriented BARTScore (faithfulness) than even the references, suggesting some conservativity in its transformations.

PG$_{\text{Clf}}$ achieves slightly higher recall BARTScore than PG$_{\text{Dyn}}$, while also generating the longest outputs, both in terms of tokens and sentences. This suggests it is less effective at identifying sentences for deletion, confirming our hypothesis that context is key for deletion. We can see here that the rank order of SMART matches that of BARTScore, suggesting it is similarly suited for simplification.

Both PG$_{\text{Tag}}$ and PG$_{\text{Tag+Dec}}$ perform quite badly relative to the other PG systems and Sent-BART. However, Doc-BART is by far the worst performing system, presumably a result of it failing to properly handle the long document lengths.

7 Conclusion

In this paper we present an approach to document simplification that decomposes the task into a two-stage process of planning and generation. We propose a planning system that is able to take document context and structure into account to produce a coherent high-level simplification plan. By using this plan to guide a sentence-level simplification model, we are able to outperform end-to-end systems in terms of both meaning preservation and simplicity.

We leave for future work the development of dedicated simplification models that can leverage a document-level plan while also considering contextual information directly during generation.

8 Acknowledgements

We thank the anonymous reviewers for their feedback. We gratefully acknowledge the support of the French National Research Agency (Gardent; award ANR-20-CHIA-0003, XNLG "Multilingual, Multi-Source Text Generation").

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see `https://www.grid5000.fr`).

9 Limitations

All models we propose are only trained with English datasets and therefore do not extent to other languages. There are also additional high-level simplification operations for which our planning framework does not offer support, such as sentence reordering, insertion, and fusion.

Furthermore, the Newsela dataset which we use in our experiments requires a license to use, meaning that researchers cannot fully reproduce our work without first obtaining said license from Newsela Inc. Due to this constraint and the low-quality alignments observed within the Wiki-auto dataset, we strongly encourage any work towards producing new open-access datasets for the document-level simplification task.

References

Fernando Alva-Manchego, Joachim Bingel, Gustavo Paetzold, Carolina Scarton, and Lucia Specia. 2017. Learning how to simplify from explicit labeling of complex-simplified text pairs. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 295–305, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. 2019a. EASSE: Easier automatic sentence simplification evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 49–54, Hong Kong, China. Association for Computational Linguistics.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2019b. Cross-sentence transformations in text simplification. In *Proceedings of the 2019 Workshop on Widening NLP*, pages 181–184, Florence, Italy. Association for Computational Linguistics.

Reinald Kim Amplayo, Peter J. Liu, Yao Zhao, and Shashi Narayan. 2022. Smart: Sentences as basic units for text evaluation.

R. Chandrasekar, Christine Doran, and B. Srinivas. 1996. Motivations and methods for text simplification. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Liam Cripwell, Joël Legrand, and Claire Gardent. 2021. Discourse-based sentence splitting. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 261–273, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liam Cripwell, Joël Legrand, and Claire Gardent. 2022. Controllable sentence simplification via operation classification. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2091–2103, Seattle, United States. Association for Computational Linguistics.

Mohammad Dehghan, Dhruv Kumar, and Lukasz Golab. 2022. GRS: Combining generation and revision in unsupervised sentence simplification. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 949–960, Dublin, Ireland. Association for Computational Linguistics.

Ashwin Devaraj, Iain Marshall, Byron Wallace, and Junyi Jessy Li. 2021. Paragraph-level simplification of medical texts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4972–4984, Online. Association for Computational Linguistics.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402, Florence, Italy. Association for Computational Linguistics.

Cristina Garbacea, Mengtian Guo, Samuel Carton, and Qiaozhu Mei. 2021. Explainable prediction of text complexity: The missing preliminaries for text simplification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1086–1097, Online. Association for Computational Linguistics.

Sian Gooding. 2022. On the ethical considerations of text simplification. In *Ninth Workshop on Speech* *and Language Processing for Assistive Technologies (SLPAT-2022)*, pages 50–57, Dublin, Ireland. Association for Computational Linguistics.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural CRF model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960, Online. Association for Computational Linguistics.

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1537–1546, Sofia, Bulgaria. Association for Computational Linguistics.

Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. 2020. Iterative edit-based unsupervised sentence simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7918–7928, Online. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul Bennett, and Marti A. Hearst. 2021. Keep it simple: Unsupervised simplification of multi-paragraph text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6365–6378, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Junyi Jessy Li and Ani Nenkova. 2015. Detecting content-heavy sentences: A cross-language case study. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1271–1281, Lisbon, Portugal. Association for Computational Linguistics.

Zhe Lin, Yitao Cai, and Xiaojun Wan. 2021. Towards document-level paraphrase generation with sentence rewriting and reordering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1033–1044, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Mounica Maddela, Fernando Alva-Manchego, and Wei Xu. 2021. Controllable text simplification with explicit paraphrasing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3536–3553, Online. Association for Computational Linguistics.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065, Hong Kong, China. Association for Computational Linguistics.

Louis Martin, Éric de la Clergerie, Benoît Sagot, and Antoine Bordes. 2020. Controllable sentence simplification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4689–4698, Marseille, France. European Language Resources Association.

Kshitij Mishra, Ankush Soni, Rahul Sharma, and Dipti Sharma. 2014. Exploring the effects of sentence simplification on Hindi to English machine translation system. In *Proceedings of the Workshop on Automatic Text Simplification - Methods and Applications in the Multilingual Society (ATS-MA 2014)*, pages 21–29, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Makoto Miwa, Rune Sætre, Yusuke Miyao, and Jun'ichi Tsujii. 2010. Entity-focused sentence simplification for relation extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 788–796, Beijing, China. Coling 2010 Organizing Committee.

Christina Niklaus, Bernhard Bermeitinger, Siegfried Handschuh, and André Freitas. 2016. A sentence simplification system for improving relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 170–174, Osaka, Japan. The COLING 2016 Organizing Committee.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 85–91, Vancouver, Canada. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vipul Raheja, and Oleksandr Skurzhanskyi. 2021. Text Simplification by Tagging. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 11–25, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

C. Scarton, P. Madhyastha, and L. Specia. 2020. Deciding when, how and for whom to simplify. © 2020 The Author(s) and IOS Press. This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial Licence (http://creativecommons.org/licenses/by-nc/4.0/).

Carolina Scarton and Lucia Specia. 2018. Learning simplifications for specific target audiences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 712–718, Melbourne, Australia. Association for Computational Linguistics.

Thomas Scialom, Louis Martin, Jacopo Staiano, Éric Villemonte de la Clergerie, and Benoît Sagot. 2021. Rethinking automatic evaluation in sentence simplification.

Advaith Siddharthan. 2003. Preserving discourse structure when simplifying text. In *Proceedings of the 9th European Workshop on Natural Language Generation (ENLG-2003) at EACL 2003*, Budapest, Hungary. Association for Computational Linguistics.

Neha Srikanth and Junyi Jessy Li. 2021. Elaborative simplification: Content addition and explanation generation in text simplification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5123–5137, Online. Association for Computational Linguistics.

Sanja Štajner and Maja Popovic. 2016. Can text simplification help machine translation? In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pages 230–242.

Renliang Sun, Hanqi Jin, and Xiaojun Wan. 2021. Document-level text simplification: Dataset, criteria and baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7997–8013, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Renliang Sun, Zhe Lin, and Xiaojun Wan. 2020. On the helpfulness of document context to sentence simplification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1411–1423, Barcelona, Spain (Online). International Committee on Computational Linguistics.

K. Woodsend and Mirella Lapata. 2011a. Wikisimple: Automatic simplification of wikipedia articles. In *AAAI*.

Kristian Woodsend and Mirella Lapata. 2011b. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 409–420, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Bohan Zhang, Prafulla Kumar Choubey, and Ruihong Huang. 2022. Predicting sentence deletions for text simplification using a functional discourse structure. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 255–261, Dublin, Ireland. Association for Computational Linguistics.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark. Association for Computational Linguistics.

Yang Zhong, Chao Jiang, Wei Xu, and Junyi Jessy Li. 2020. Discourse level factors for sentence deletion in text simplification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9709–9716.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1353–1361, Beijing, China. Coling 2010 Organizing Committee.

A Training Details for the Planning Models

Each model was trained with a learning rate of $1e^{-5}$, a batch size of 32 and a dropout rate of 0.1. We ran experiments on a computing grid with $2\times$ Nvidia A40 GPUs (45GB memory).

For the contextual classifier, we test with r values of 9 and 13, subject to findings in Appendix C. All layers in common with the standard RoBERTa architecture are initialised with the `RoBERTa-base` pretrained weights. All added positional embedding layers are also initialised with the pretrained weights from the `RoBERTa-base` positional embedding layer. All other layers are randomly initialised.

B Training Details for the Simplification Models

For all generative models, we used a learning rate of $3e^{-5}$, a batch size of 16, and performed dropout with a rate of 0.1 and early stopping. The network has 6 layers in each of the encoder and decoder, with a hidden size of 768. All models were trained on a computing grid using $2\times$ Nvidia A40 GPUs (45GB memory) in under 24 hours.

C Context Window Size

To determine the optimal context window size for the contextual planner we ran a series of experiments with varying values of the radius, r. We used 100,000 random examples from the Newsela-auto (non-adjacent reading levels) training set and trained a model with each of the configurations for 5 epochs. Results can bee seen in Figure 4.

The *deletion* operation is most affected by the inclusion of context, with performance rapidly rising as r grows to 13. The *rephrase* operation appears to slowly degrade in performance as r increases, while the other two operations show no obvious pattern. We also observe that $r = 9$ produces the highest macro F1.

D Extra Evaluation Results

For clarity, we provide scores for a wider range of simplification evaluation metrics that were not included in the main body of the paper in Table 5. These mostly include popular metrics used for sentence simplification that we do not believe adapt as well to the document-level setting, do not provide further insight into system differences, or have not recieved much support in the literature. Specifially, we include BLEU (Papineni et al., 2002), and full operation scores for both SARI and D-SARI (Sun et al., 2021). For D-SARI, we apply the document-level penalties on top of the base EASSE implementation of SARI.

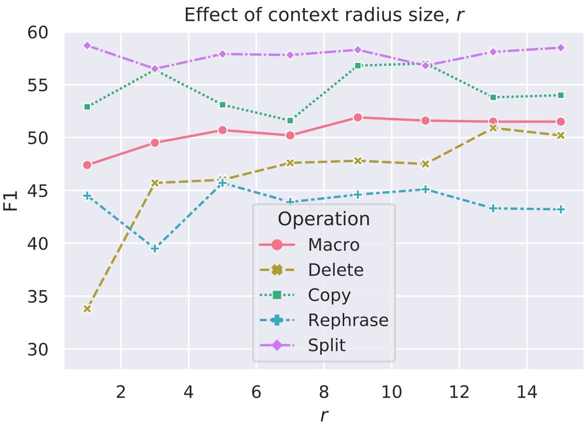


Figure 4: Effect of context window size on F1 scores.

We can see that the main SARI differences between the context-free planner and Sent-BART is that Sent-BART achieves higher keep, while the planner achieves higher add. This suggests that Sent-BART is likely more conservative in edits. Further, as the planner does not have access to contextual content, it is likely failing to consistently copy/delete the correct parts of the text.

E Example Planner Predictions

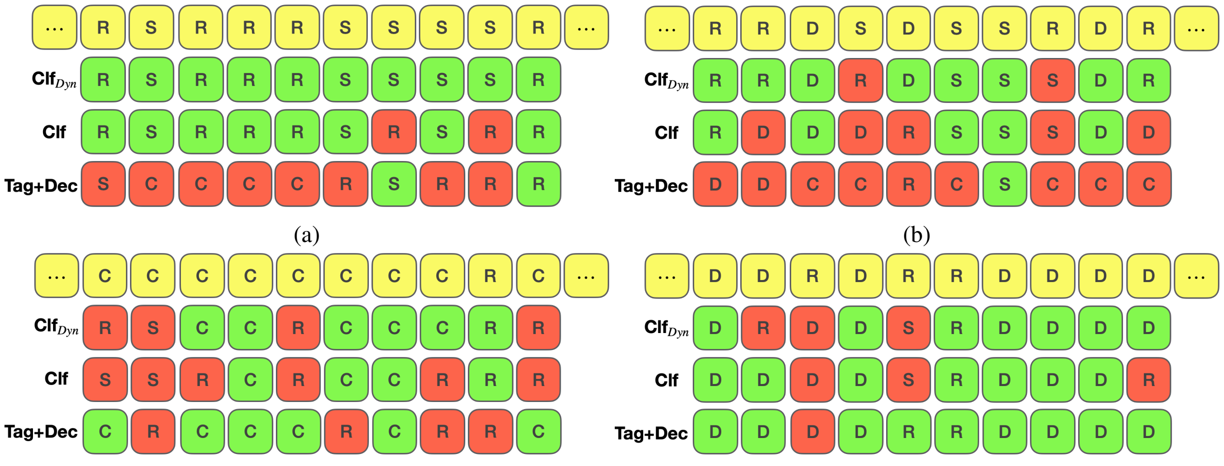
Figure 5 shows example snippets of planner model outputs. We have selected representative extracts that highlight the strengths and weaknesses of the main models. We do not include outputs from Tagger as they are virtually identical to Tagger+Dec in most cases and therefore do not provide further insight.

F Example Simplification

Figure 6 shows system output examples for the simplification models. We only show texts from Wiki-auto as they are easier to showcase due to their shorter length, as well as their being licensing restrictions for Newsela content.

System	BLEU ↑	**D-SARI** ↑	add	keep	delete	**SARI** ↑	add	keep	delete
Input	46.2	8.76	0.0	26.29	0.0	20.52	0.0	61.56	0.0
Reference	100.0	99.98	99.99	99.97	99.99	99.99	100	99.97	99.99
Doc-BART	31.13	30.60	16.54	25.01	50.24	47.07	20.41	55.40	65.40
Sent-BART	70.74	66.27	53.89	71.95	72.95	73.02	55.91	83.66	79.48
PG$_{\text{Tag}}$	48.08	42.96	31.70	44.01	53.17	56.13	35.61	65.61	67.18
PG$_{\text{Tag+Dec}}$	48.12	43.31	31.57	44.68	53.69	56.06	35.54	65.54	67.11
PG$_{\text{Clf}}$	70.84	62.97	56.31	65.15	67.47	73.83	57.62	83.56	80.32
PG$_{\text{Dyn}}$	**72.41**	**67.42**	**56.83**	**71.82**	**73.61**	**75.00**	**58.88**	**84.75**	**81.36**
PG$_{\text{Oracle}}$	**78.97**	**77.02**	**63.44**	**83.92**	**83.70**	**80.74**	**65.22**	**89.94**	**87.05**

Table 5: Extra results for document simplification experiments on Newsela-auto.



(a) (b)

(c) (d)

Figure 5: Example planning results for various models. Subfigures show representative snippets from Newsela-auto test-set documents. The silver labels are shown above in yellow, and system outputs are shown on the rows below with correct predictions in green and incorrect predictions in red. Clf$_{Dyn}$ is our best performing model, the contextual classifier with dynamic context. Figure 5a shows a case where there are lots of context-agnostic operations (rephrase, split) resulting in poor performance from Tagger+Dec. Figure 5b shows a varied snippet where Clf$_{Dyn}$ appears to be the best at identifying both rephrase and split, as well as delete. Figures 5c and 5d show that Tagger+Dec is capable of performing well in situations demanding a lot of context-dependent operations (copy, delete).

System	Output
Complex	Silvano "Nano" Campeggi (January 23, 1923 – August 29, 2018) was an Italian artist who designed and produced the artwork for the posters of many classic Hollywood films. His iconic images are associated with the golden era of Hollywood and Campeggi is now generally regarded as the most important graphic artist and poster designer in the history of American cinema. In the following decades, Campeggi designed and produced the poster and advertising graphics for over 3000 films, working not only under contract with the MGM studios, but also with Warner Brothers, Paramount, Universal, Columbia Pictures, United Artists, RKO, Twentieth-Century Fox and several other movie studios. Sixty-four of the films he illustrated won Oscars, including "Casablanca", "Ben-Hur", "Singin' in the Rain", "An American in Paris", "West Side Story", "Exodus", "Breakfast at Tiffany's", and "Gigi". Campeggi died on 29 August 2018, at the age of 95.
Simple	Silvano "Nano" Campeggi (January 23, 1923 – August 29, 2018) was an Italian artist. He designed and produced the artwork for the posters of many classic Hollywood movies. Campeggi was known for his poster designs for "Casablanca", "Singin' in the Rain", and "Breakfast at Tiffany's". Campeggi died on August 29, 2018 in Florence from respiratory failure at the age of 95.
Doc-BART	Silvano "Nano" Campeggi (January 23, 1923 – August 29, 2018) was an Italian artist. He designed and produced the artwork for the posters of many classic Hollywood movies, including "Casablanca", "Ben-Hur", "Singin' in the Rain", "An American in Paris", "West Side Story", "Exodus", "Breakfast at Tiffany's", "Gigi", and "".
Sent-BART	Silvano "Nano" Campeggi (23 January 1923 – 29 August 2018) was an Italian artist. He designed and produced the artwork for the posters of many classic Hollywood movies. Campeggi is generally regarded as the most important graphic artist and poster designer in the history of American cinema. Campeggi designed and produced the poster and advertising graphics for over 3000 movies, working not only under contract with the MGM studios, but also with Warner Brothers, Paramount, Universal, Columbia Pictures, United Artists, RKO, Twentieth Century Fox and several other movie studios. Sixty-four of the movies he illustrated won Oscars, including "Casablanca", "Ben-Hur", "Singin' in the Rain", "An American in Paris", "West Side Story", "Exodus", "Breakfast at Tiffany's", and "Gigi". Campeggi died on 29 August 2018, at the age of 95.
PG_{Dyn}	Silvano "Nano" Campeggi (January 23, 1923 – August 29, 2018) was an Italian artist. He designed and produced the artwork for the posters of many classic Hollywood movies. Sixty-four of the movies he illustrated won Oscars, including "Casablanca", "Ben-Hur", "Singin' in the Rain", "An American in Paris", "West Side Story", "Exodus", "Breakfast at Tiffany's", and "Gigi". Campeggi died on 29 August 2018 at the age of 95.

Table 6: Simplification outputs for a specific document pair example. Although Newsela-auto is the focus of our simplification experiments, we can only include example documents from Wiki-auto due to licensing constraints.

Efficient Hybrid Generation Framework for Aspect-Based Sentiment Analysis

Haoran Lv[1], Junyi Liu[1], Henan Wang[1], Yaoming Wang[2], Jixiang Luo[2], Yaxiao Liu[1]
[1]Amazon Web Services, China
[2]Shanghai Jiao Tong University, China
`{lvhaoran, liujunyi, henanwan, liuyaxia}@amazon.com`
`{wang_yaoming, ljx123456}@sjtu.edu.cn`

Abstract

Aspect-based sentiment analysis (ABSA) has attracted broad attention due to its commercial value. Natural Language Generation-based (NLG) approaches dominate the recent advance in ABSA tasks. However, current NLG practices are inefficient because most of them directly employ an autoregressive generation framework that cannot efficiently generate location information and semantic representations of ABSA targets. In this paper, we propose a novel framework, namely Efficient Hybrid Generation (EHG) to revolutionize traditions. Specifically, we leverage an Efficient Hybrid Transformer to generate the location and semantic information of ABSA targets in parallel. Besides, we design a novel global hybrid loss function in combination with bipartite matching to achieve end-to-end model training. Extensive experiments demonstrate that our proposed EHG framework greatly improves the efficiency of NLG-based methods and outperforms the competitive baselines in almost all cases.

1 Introduction

Aspect-based sentiment analysis (ABSA) enjoys broad commercial value, e.g., analyzing customer's opinions through review data to improve products. ABSA consists of four basic sentiment elements, including aspect term, opinion term, aspect category and sentiment polarity. Illustrated in Fig. 1, the aspect terms a_1, a_2 in the sentence belong to the aspect categories c_1, c_2 , respectively, and their corresponding opinion terms are o_1, o_2 , with sentiment polarity s_1, s_2. For simplicity, we name the aspect term and opinion term as **sentiment entities** (SE), and the aspect category and sentiment polarity as **sentiment abstractions** (SA). The majority of ABSA subtasks utilize sentences as input and the combination of SE and SA as output, so we define the output set of all ABSA subtasks as **sentiment tuples** (ST). Fig. 1 shows all the subtasks studied in this work.

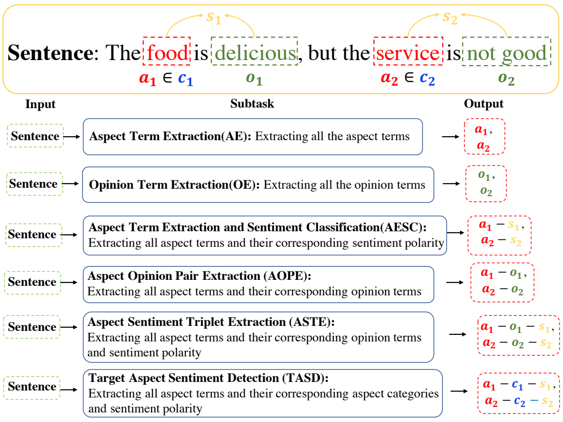


Figure 1: The six ABSA subtasks studied in this work, where a_1=*food*, a_2=*service*, c_1=*food quality*, c_2=*service general*, o_1=*delicious*, o_2=*not good*, s_1=*positive*, s_2=*negative*.

Natural Language Understanding (NLU) based approaches were utilized in early research for modeling ABSA problems. The Two stage pipeline, i.e., using a sequence tagging model to extract SE, then using a classification model to distinguish SA (Li and Lam, 2017; Ma et al., 2017; Xue and Li, 2018; Fan et al., 2019), ignored the semantics of the labels and propagated error in a not end-to-end manner. Besides, unique end-to-end NLU variants were proposed to enhance the information interaction between elements within ST, however, these approaches suffered from complex model design and were not generic for different ABSA subtasks.

Natural Language Generation (NLG) based approaches model the ABSA tasks uniformly as sequence-to-sequence generation tasks (Zhang et al., 2021b; Yan et al., 2021; Zhang et al., 2021a; Mao et al., 2022) and have dominated the recent advance in ABSA tasks. Specifically, the sequence-to-sequence generation framework can adapt to multiple ABSA subtasks simultaneously without additional architectural design, and the autoregressive generation mode (Neal, 1992) can naturally

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1007–1018
May 2-6, 2023

include information interaction of elements within ST. Despite the superior performance of NLG-based approaches, the autoregressive generation also renders NLG-based methods time inefficient in generating multiple sets of STs. Besides, existing NLG-based methods fail to adequately coordinate the semantic representation of STs and the positioning of SEs when generating results.

In this paper, we propose an Efficient Hybrid Generation framework EHG to generate semantic representations of multiple STs in parallel and simultaneously output the location information of SEs corresponding to each ST. EHG is composed of two modules: Efficient Hybrid Transformer (EHT) and Entity Correction Module (ECM). For EHT, inspired by DETR (Carion et al., 2020), we use a fixed number of *object queries* to detect STs in sentences in parallel and locate the corresponding SE. Further, we pass *object queries* that detect ST (obtained by *Filter* or *Matcher*), through the *Semi-Autoregressive Generator* to generate semantic representations of the corresponding ST. In addition we propose a novel global hybrid loss function in combination with bipartite matching named hybrid Hungarian loss, to achieve end-to-end model training. For ECM, we normalize the location information of the corresponding SEs based on the input text and the semantic representation of each ST generated. The contributions of this paper are summarized as below.

- We propose a novel and Efficient Hybrid Generation framework that generates semantic representations of STs and simultaneously locates the SEs within them.
- We introduce a fixed number of *object queries* to detect and decode multiple sentiment tuples in parallel, which greatly improves the efficiency of current generation methods.
- We propose a novel hybrid Hungarian loss function that allows our Efficient Hybrid Transformer to be trained end-to-end.

Experimental results demonstrate that our proposed EHG achieves the state-of-art performance in almost all datasets on the six ABSA subtasks and outperforms existing NLG-based methods in terms of inference efficiency.

2 Related Work

In this section, we summarize the existing NLU-based and NLG-based approaches for aspect-based sentiment analysis.

2.1 NLU-based ABSA

For NLU-based methods, early work was often devoted to solving simple problems. For example, extracting SEs or classifying the sentiment polarity for a given aspect (Tang et al., 2015; Xu et al., 2018; Li et al., 2018; Wang et al., 2016). Among them, the vast majority defined the extraction of SEs as a sequence tagging problem (Huang et al., 2015) and the determination SAs as a classification problem. However, this approach ignores the information interaction of the elements within ST and makes the errors cumulative. Recently, much work has been devoted to solving complex multi-objective problems (ASTE (Zhang et al., 2015; Li et al., 2019; Mitchell et al., 2013), TASD(Wan et al., 2020), AOPE (Zhao et al., 2020; Chen et al., 2020)), such as predicting multiple elements simultaneously. Some of these works use the idea of pipeline to solve the extraction of SEs and the classification of SAs in stages (Peng et al., 2020; Mao et al., 2021). There are also some recent approaches that attempt to introduce interactions between multiple elements in the ST during inference to achieve end-to-end multi-objective modeling (Xu et al., 2021, 2020; Ma et al., 2018). However, this approach usually requires complex model design and is not generic for different ABSA subtasks.

2.2 NLG-based ABSA

Generation frameworks are increasingly used in NLP due to their powerful generality and rich pretrained models (PTMs) (Sutskever et al., 2014; Cho et al., 2014; Radford et al., 2018; Shao et al., 2021). There has been some recent work using NLG-based approaches to solve ABSA problems. Yan et al. (2021) use BART (Lewis et al., 2019) to receive sentence input and generate STs directly autoregressively, where SEs are represented using location indexes. More directly, Zhang et al. (2021b,a); Mao et al. (2022) used T5 to generate semantic representations of STs, and in addition, to locate the location of SEs, Zhang et al. (2021b) additionally proposed an *Annotation* pattern to insert STs to the original sentences. Although the NLG-based approach can model multiple ABSA subtasks in a unified framework, the above two methods share the common problem that the efficiency of the generation will be greatly reduced when there are multiple sets of STs on the input in the autoregressive framework. In addition, for the direct output of the SE index, although it can locate entity locations,

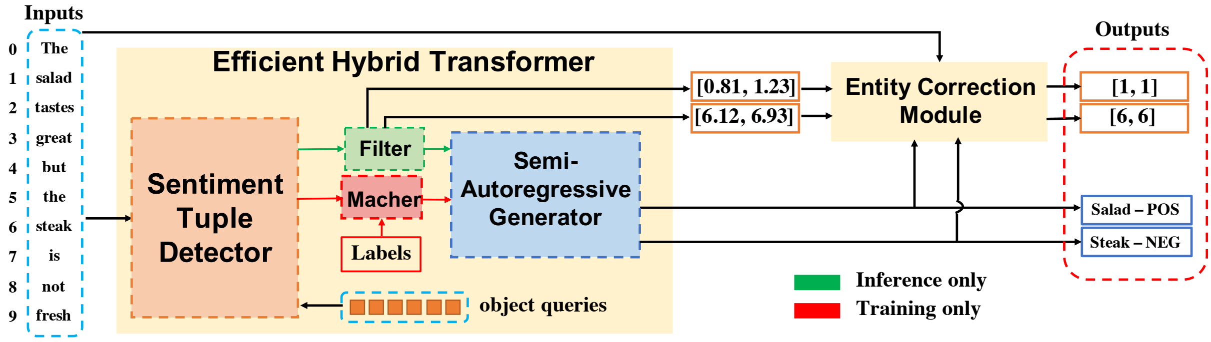


Figure 2: Efficient Hybrid Generation Framework. 1) the input sentence is directly passed through the *Efficient Hybrid Transformer* to generate a set of STs in parallel and locate the corresponding SEs. Specifically, a fixed number of learnable embeddings, called *object queries*, will be fed into the *Sentiment Tuple Detector* along with the input, and each *object queries* will focus on capturing an ST and locating the SE in it. The output of the *Sentiment Tuple Detector* will then be passed through a *Filter* (for training, it's *Matcher*) to remove the part of the ST that is not detected. Finally, the output of the *Filter* will be fed to the *Semi-Autoregressive Generator* to generate multiple STs in parallel. 2) the SE location information will be combined with the input and the corresponding ST to get the corrected SE index by *Entity Correction Module*.

this approach is not intuitive enough compared to directly generating semantic representations, and still requires additional design for scenarios that require the extraction of aspect categories, such as TASD. For the direct generation of semantic representations, it is impossible to locate the location of SEs accurately (the same SE may appear multiple times in the input), and although the *Annotation* pattern is proposed, the operation of inserting sentiment tuples to the input makes the generated results very redundant, which amplifies the problem of autoregressive inefficiency.

3 Methodology

Our proposed Efficient Hybrid Generation Framework is depicted in detail in Fig. 2. Since the output of all six ABSA subtasks contains the part of SEs, they can be described under a unified framework. To facilitate the description, this section expands on AESC as an example, first introducing the task definition, and then Efficient Hybrid Transformer and Entity Correction Module, are described in detail.

3.1 Problem Formulation

As discussed above, the proposed framework contains two parts of output, which are the semantic representation of multiple STs and the location of the corresponding SEs. Specifically, the proposed framework supports a fixed maximum number M of outputs, where M is set to be much larger than the number of STs generally present in the sentence. Here we denote aspect term, opinion term, aspect category, sentiment polarity, sentiment entity, sentiment abstraction and sentiment tuple as a, o, c, s, se, sa and st respectively, and define $X = \{x_i\}_{i=1}^{N}$ to represent an input text of length N, $L_{hybrid}(\cdot)$ represents the proposed hybrid Hungarian loss function. We use hat "$*, \widehat{*}$" to distinguish between model output and ground truth, then the problem is defined as follows,

$$argmin \sum_{j=1}^{M} L_{hybrid}[(d_j^{se}, y_j^{st}), (\widehat{d_j^{se}}, \widehat{y_j^{st}})] \quad (1)$$

where d_j^{se} and y_j^{st} represent the detection result and semantic representation of the j-th ST, respectively. Specifically, $d_j^{se} = (c_j^{st}, idx_j)$ where c_j^{st} is the class label of the ST and there are only two classes representing the presence or absence of the ST respectively. The idx_j is defined as the index range of the SE in the j-th ST normalized in the sentence, and we set $idx_j = [-1, -1]$ when $c_j^{st} = \oslash$ represents the absence of the ST.

Furthermore, since the outputs of different ABSA subtasks may contain different SEs, we describe the idx_j corresponding to the different subtasks as follows:

$$idx_j = \begin{cases} (idx_j^a), & [AE, AESC, TASD], \\ (idx_j^o), & [OE], \\ (idx_j^a, idx_j^o), & [AOPE, ASTE], \end{cases}$$

where $idx_j^a = [0,1]^2$ and $idx_j^o = [0,1]^2$ represent the position indices of a_j and o_j respectively.

3.2 Efficient Hybrid Transformer

Efficient Hybrid Transformer contains two main components, *Sentiment Tuple Detector*, which detects the STs in X, and *Semi-Autoregressive Generator*, which generates a semantic representation for

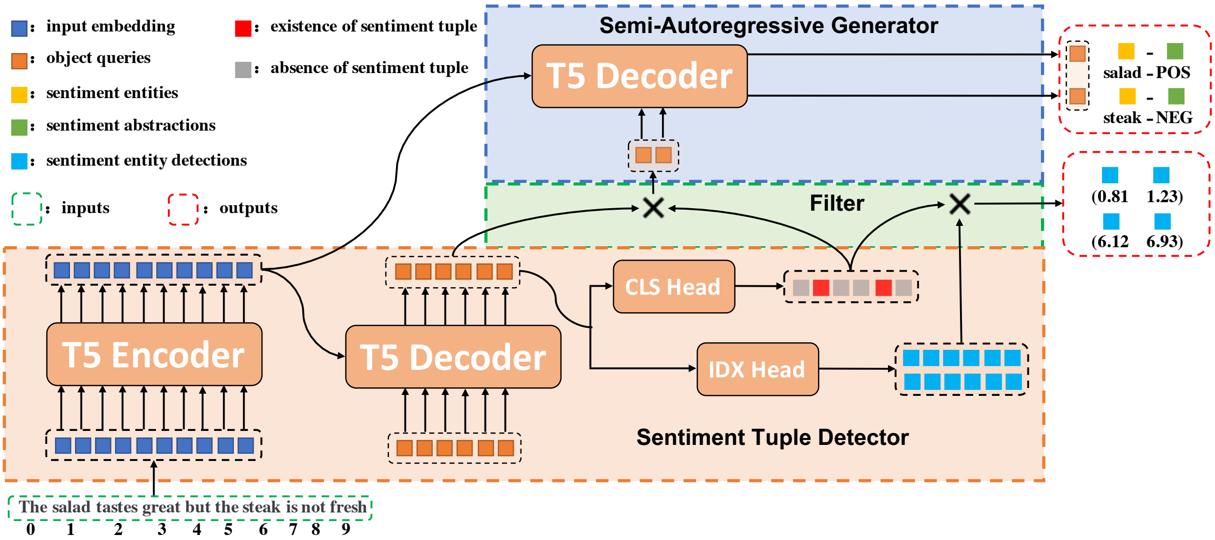


Figure 3: Inference details of Efficient Hybrid Transformer under AESC.

each detected ST, the details are shown in Fig. 3. It is worth mentioning that our EHT uses an asymmetric inference and training process, the difference between the two is that *Filter* is used for inference and *Matcher* is used for training.

3.2.1 Sentiment Tuple Detector

Inspired by DETR, we define mining STs in sentences as an object detection problem. Specifically, we detect the presence of STs and then locate the corresponding SEs in X. The *Sentiment Tuple Detector* utilizes the conventional transformer encoder-decoder architecture (Vaswani et al., 2017) , a setup that is convenient to extend to any similar pre-trained model (Qiu et al., 2020; Peters et al., 2018; Devlin et al., 2018; Raffel et al., 2019; Song et al., 2019), and in this paper we directly use the encoder and decoder of the pre-trained T5 model.

The input text X is vectorized and passed through the encoder to obtain the encoded D-dimensional hidden variable $H \in \mathbb{R}^{N \times D}$. This process can be described in the following form:

$$H = T5Encoder(Embed(X)), \tag{2}$$

where $Embed(X) \in \mathbb{R}^{N \times D}$, represents the vectorized X.

Different from the original autoregressive decoder, the decoder of the *Sentiment Tuple Detector* decodes M objects in parallel. Similar to DETR, we use M learnable embeddings $q^{st}_{in} \in \mathbb{R}^{M \times D}$, called *object queries*, as input to the decoder to generate output embeddings $q^{st}_{out} \in \mathbb{R}^{M \times D}$ through global self-attention and cross-attention with H.

$$q^{st}_{out} = T5Decoder(q^{st}_{in}, H), \tag{3}$$

The output embedding q^{st}_{out} finally outputs the class labels of the STs and the location coordinates of the corresponding SEs independently through the linear projection layer and the feed-forward neural network, respectively.

$$\begin{aligned} \widehat{c}^{st} &= Softmax(MLP(q^{st}_{out})), \\ \widehat{idx}^{a} = FFN_a(q^{st}_{out}), &\ \widehat{idx}^{o} = FFN_o(q^{st}_{out}), \end{aligned} \tag{4}$$

where $\widehat{c}^{st} = \{\widehat{c}^{st}_j\}^M_{j=1}, \in \mathbb{R}^{M \times 2}$ is the class vector of the ST (here only two classes are present and absent). $\widehat{idx}^{a} = \{\widehat{idx}^{a}_j\}^M_{j=1}, \in \mathbb{R}^{M \times 2}$ and $\widehat{idx}^{o} = \{\widehat{idx}^{o}_j\}^M_{j=1}, \in \mathbb{R}^{M \times 2}$ are generated by two 3-layer feed-forward neural networks (FFN) with ReLU activation function and hidden dimension D, representing the normalized position coordinates of a and o, respectively. Since different ABSA subtasks contain different SEs in their outputs, FFN_a and FFN_o are set according to the actual needs, and in the case of ASEC, only FFN_a is used to generate $\widehat{idx}^{a}$.

3.2.2 Filter

The role of the *Filter* in inference is 2-fold, firstly filtering out the output of the *Sentiment Tuple Detector*; secondly the corresponding q^{st}_{out} of detected ST need to be fed into the *Semi-Autoregressive Generator* to decode the semantic representation of STs. The process can be expressed as follows.

$$\widehat{d}^{se}_{\widehat{c}^{st} \neq \varnothing}, q^{st}_{out, \widehat{c}^{st} \neq \varnothing} = Filter(\widehat{D}^{se}, q^{st}_{out}), \tag{5}$$

3.2.3 Semi-Autoregressive Generator

The semi-autoregressive decoder aims to decode the detected STs in parallel. Different from traditional autoregressive decoding, our semi-autoregressive decoder uses $q^{st}_{out,\widehat{c}^{st}\neq\varnothing}$ as the embedding of the starting token for each ST and then performs autoregressive decoding separately. For the s-th ST, we use $q^{st}_{out,s,\widehat{c}^{st}\neq\varnothing}$ as the starting embedding, then the autoregressive decoding process can be expressed as:

$$\begin{aligned} h^d_{s,0} &= q^{st}_{out,s,\widehat{c}^{st}\neq\varnothing} \\ h^d_{s,t} &= T5Decoder(H^e, h^d_{s,t-1}) \\ st_{s,t} &= Softmax(MLP(h^d_{s,t})) \end{aligned} \tag{6}$$

where $h^d_{s,t} \in \mathbb{R}^D$ is the decoder output of step t of the s-th ST, and $st_{s,t}$ is the corresponding token representation, which is generated by a linear projection layer that immediately follows softmax.

Besides Decoder in the *Semi-Autoregressive Generator* shares parameters with the Decoder in the *Sentiment Tuple Detector*, which allows our model to maintain almost the same number of parameters as the original T5.

3.2.4 Matcher

For model training, we need to know the correspondence between a fixed number of M outputs and the ground truth in order to calculate the loss. Thus we use the Hungarian algorithm (Kuhn, 1955) to obtain the optimal bipartite matching between model outputs and the ground truth, and then calculate the loss for the completed matched outputs and the ground truth to train the model, this part focuses on the matching process.

We define $\phi \in \Phi_M$ as all possible mapping connections between the ground truth and the model outputs, and L_{match} as the matching loss function under the given mapping condition, then the objective function for finding the optimal mapping $\phi_{optimal}$ is as follows.

$$\phi_{optimal} = \underset{\phi\in\Phi_M}{argmin} \sum_{j=1}^{M} L_{match}(d^{se}_j, \widehat{d}^{se}_{\phi(j)}) \tag{7}$$

Since we assume that M is greater than the number of STs in the actual input, we use $\varnothing$ to padding each ground truth d^{se} that is less than M. In Eq. 7, the index of $\widehat{d}^{se}$ is mapped by $\phi_{optimal}$ such that L_{match} is minimized. For L_{match}, we consider two components which are class cost L_{cls} and location cost L_{idx}, as follows.

$$L_{match} = \lambda_{cls}L_{cls} + \lambda_{idx}L_{idx} \tag{8}$$

$$L_{cls} = -\sum_{j=1}^{M}(\widehat{p}_{\phi(j)}(c^{st}_j))_{c^{st}_j\neq\varnothing} \tag{9}$$

$$L_{idx} = \sum_{j=1}^{M}(\|idx_j - \widehat{idx}_{\phi(j)}\|_1)_{c^{st}_j\neq\varnothing} \tag{10}$$

where λ_{cls} and λ_{idx} are hyperparameters, $\widehat{p}_{\phi(j)}(c^{st}_j)$ and $idx_{\phi(j)}$ represents the probability of class c_j and normalized index of SEs with index $\phi(i)$, respectively. At this point, we have completed the matching between the model output and the ground truth without any gradient computation, but the accurate matching is very important for the subsequent loss calculation, as discussed later.

3.2.5 Loss

To enable end-to-end training of the EHT, we propose a hybrid Hungarian loss L_{hybrid}, which considers both the detection loss L_{det} and the generation loss L_{gen} under a given optimal matching $\phi_{optimal}$.

$$\begin{aligned} L_{hybrid} &= L_{det} + \mu_{gen}L_{gen} \\ L_{det} &= \mu_{cls}\sum_{j=1}^{M}[-log(\widehat{p}_{\phi_{optimal}(j)}(c^{st}_j)) + \\ &\quad \mu_{idx}(\|idx_j - \widehat{idx}_{\phi_{optimal}(j)}\|_1)_{c^{st}_j\neq\varnothing}] \\ L_{gen} &= \sum_{j=1}^{M}(\sum_{t=1} -log(\widehat{p}_{\phi_{optimal}(j)}(st_{j,t})))_{c^{st}_j\neq\varnothing} \end{aligned}$$

where $\phi_{optimal}$ is obtained in *Matcher*, μ_{cls}, μ_{idx} and μ_{gen} are hyperparameters. For L_{det}, the difference with Eq. 9 is that we use the standard cross-entropy loss and consider the part of $c^{st}_j \neq \varnothing$ into the classification loss as well, which reduces the redundant detections, as will be shown in the experimental section. For L_{gen}, we simply computed the negative log-likelihood loss of the ST generated by the matched $q^{st}_{out,\widehat{c}^{st}\neq\varnothing}$ with the ground truth ST. Thus we jointly optimize L_{det} and L_{gen} for end-to-end training.

3.3 Entity Correction Module

In computer vision, object detection is not required to output the exact pixel coordinates because the images have redundant information, but for ABSA, inaccurate detection of SEs is unacceptable. To solve this problem, we propose an *Entity Correction Module* that corrects the detections of SEs by combining the input text and the output of the *Semi-Autoregressive Generator*.

First, we define SEs in the s-th ST of the model output as $\widehat{se}_s \in \widehat{st}_s$ and the corresponding detection result as $\widehat{idx}_s$, then we have the following

Algorithm 1 Entity Correction Module

```
$N = len(X); res = \widehat{idx}_s$
$Mean_{idx} = Int(mean(\widehat{idx}_s))$
$s, e = \widehat{st}_s[0], \widehat{st}_s[-1]$
$L, R = Mean_{idx}; w = len(\widehat{st}_s))$
while True do
    if $x[L]$==$s$ and $x[L : L$+$w]$==$\widehat{st}_s$ then
        $res \leftarrow (L, L$+$w)$; berak
    else if $x[R]$==$e$ and $x[R$-$w$+1 : $R$+1]==$\widehat{st}_s$ then
        $res \leftarrow (R$-$w$+1$, R)$; berak
    else if $L$==0 and $R$==$N$-1 then
        berak
    else
        $L = max(0, L$-1$)$;
        $R = min(N$-1$, R$+1$)$;
    end if
end while
return $res$
```

algorithm.

In Algorithm 1, $Mean_{idx}$ represents the midpoint of $\widehat{idx}_s$ rounded down. The essence of the whole algorithm is to consider the midpoint $Mean_{idx}$ as a fuzzy detection, and then use $Mean_{idx}$ to initialize two pointers in the input X to search for the nearest SE forward and backward respectively, and finally return the original detection $\widehat{idx}_s$ if there is no match.

4 Experiments

4.1 Datasets

We evaluate our EHG framework on four benchmark datasets Laptop14, Rest14, Rest15, and Rest16 that are publicly available in SemEval 2014, 2015 and 2016 (Pontiki et al., 2014, 2015, 2016). We use multiple publicly available datasets derived from the original dataset to respond to different ABSA subtasks, and these derived datasets are often additionally annotated. Specifically, we used the dataset D_{17} provided by Wang et al. (2017) to evaluate the AE and OE; the dataset D_{19} provided by Peng et al. (2020) to evaluate the AOPE and AESC; the dataset D_{20a} provided by Xu et al. (2020) to evaluate the ASTE task; and the dataset D_{20b} provided by Wan et al. (2020) to evaluate the TASD task.

Model	Rest14		Rest15		Laptop14	
	AE	OE	AE	OE	AE	OE
IMN-BERT	84.06	85.10	69.9	73.29	77.55	81.0
RACL-BERT	86.38	87.18	73.99	76.0	81.79	79.72
Dual-MRC	86.6	-	75.08	-	82.51	-
SPAN-BERT	86.71	-	74.63	-	82.34	-
Gen-idx	87.07	87.29	75.48	76.49	83.52	77.86
EHG(Ours)	**87.43**	**88.87**	**79.41**	**82.72**	**85.32**	**84.42**

Table 1: Comparison of F1 scores for AE and OE. All comparison results are from Yan et al. (2021). Bolded fonts for the best performance and underlined fonts for the second-best performance.

4.2 Metrics

We use precision(P), recall (R) and F1 scores to evaluate for all experiments, where a correct prediction is considered when and only when the ST is exactly the same as the ground truth.

4.3 Experiment Setup

In order to maintain a similar number of parameters as existing methods, the encoder and decoders in the proposed EHT are derived from the encoder and decoder of T5 base model. Since all T5 decoders in EHT are parameter shared, we have almost the same number of parameters as the original T5[1]. It is worth mentioning that our EHT is based on the T5 model implementation of the huggingface Transformer library[2].

For the different ABSA subtasks, we used a similar experimental setup. We use the AdamW (Loshchilov and Hutter, 2018) optimizer to train the EHT with a learning rate of 3e-4 and a batch size of 2 and a cumulative gradient every two batches, and all Transformers are initialized using a pretrained T5-base. We train all datasets with 16 object queries (M=16), a setting larger than the maximum number of sentiment tuples in all datasets for a single sentence. We set λ_{cls}=2, λ_{idx}=1, μ_{cls}=0.75 and μ_{idx}=1.5 for all ABSA subtasks. In addition, for μ_{gen}, we set 1.5 for AE, OE and AOPE, 1.25 for ASTE, 1 for AESC, and 0.75 TASD. All training processes are performed on an ml.p3.2xlarge instance of Amazon Elastic Compute Cloud, which includes an NVIDIA Tesla V100 (16G) GPU.

4.4 Competitive Baselines

To demonstrate the superiority of our proposed EHG framework, we compare multiple SOTA

---

[1] Compared to T5-base, which has about 222M parameters, our EHT has about 223M parameters.

[2] `https://github.com/huggingface/transformers`

Models	R14	R15	R16	L14
Li-unified+†	51.0	47.82	44.31	42.34
Two-stage†	51.46	52.32	54.21	42.87
JET-BERT†	62.40	57.53	63.83	51.04
Gen-idx†	65.25	59.26	67.62	58.69
GAS-R*	70.52	60.23	69.05	58.19
GAS-Para*	**72.03**	62.56	71.70	61.13
EHG(Ours)	70.13	62.71	70.17	60.8
EHG-Para(Ours)	71.82	**63.58**	**72.35**	**61.53**

Table 2: F1 results for ASTE task. The comparison results with "†" are retrieved from Yan et al. (2021), and result with "$*$" is from Zhang et al. (2021a). Bolded fonts for the best performance and underlined fonts for the second-best performance.

methods for six ABSA subtasks, including NLG-based methods Gen-idx (Yan et al., 2021), GAS-R (Zhang et al., 2021b) and GAS-Para (Zhang et al., 2021a). In addition, for fairness, all NLG-based methods do not use prediction normalization in GAS-R (Zhang et al., 2021b). For NLU-based methods, IMN-BERT (He et al., 2019), RACL-BERT (Chen and Qian, 2020), SPAN-BERT (Hu et al., 2019), Jet+BERT (Xu et al., 2020), Peng-two-stage (Peng et al., 2020), Li-unified-R (Li et al., 2019), Dual-MRC (Mao et al., 2021), SDRN (Chen et al., 2020), TAS (Wan et al., 2020) and Baseline (Brun and Nikoulina, 2018) are selected for comparison.

The performance: Tables 1, 2, 3, 4 show the main comparison results for AE, OE, ASTE, AOPE, AESC, and TASD, respectively. It is worth mentioning that the proposed method achieves the best F1 score in all cases except for AOPE and ASTE on the Rest14 dataset, and TASD on the Rest15 dataset. In addition, for ASTE and TASD, we also combine EHG with GAS-Para to further improve the performance while maintaining the advantages of the original EHG inference efficiency (EHG-Para), which fully demonstrates that the proposed method can be easily combined with existing NLG-based methods, each taking advantage of the other. Finally, we found that all NLG-based methods achieved stronger performance on ASTE, AOPE, AESC, and TASD tasks compared to NLU-based methods, demonstrating that the semantic interactions within ST tend to be more effective.

Inference efficiency: To demonstrate the speed advantage of the proposed method, we compared the generation speed and generation quality of the ASTE task on the Rest16 dataset for different NLG-based methods. As shown in Fig. 4, where the horizontal coordinate represents the number of STs generated in a single inference, the left vertical coordinate represents the average time to generate different numbers of STs and the right vertical coordinate represents the F1 corresponding to different NLG-based methods for different number of STs in ground truth. It can be clearly found that the inference time of the proposed method is independent of the number of generated STs, but depends only on the longest ST generated in parallel, and that the quality of generation does not decrease for different numbers of STs, but is relatively better among the three methods (highest in 1, 3, and 5).

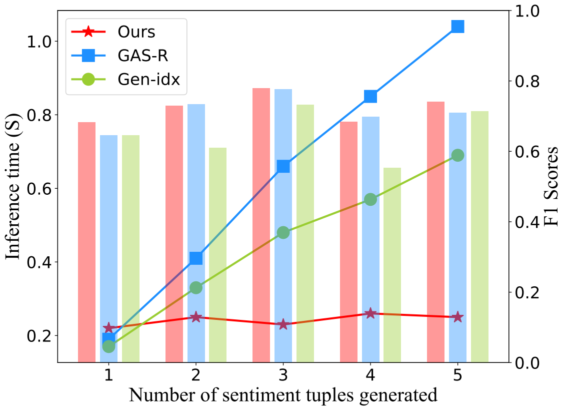


Figure 4: Comparison of the generation speed and generation quality of different NLG-based methods, where the bar chart corresponds to the F1 scores on the right and the line chart corresponds to the inference time on the left.

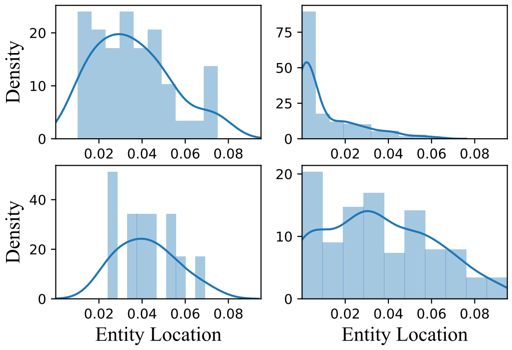


Figure 5: Visualizing the location distribution of sentiment entities detected by the first four object queries on the rest16 test set under the TASD task.

4.5 Analysis and Ablation Studies

To better understand the proposed framework, we visualize the location distribution of SEs (aspect

Model	Rest14		Rest15		Rest16		Laptop14	
	AOPE	AESC	AOPE	AESC	AOPE	AESC	AOPE	AESC
Li-unified+†	55.34	73.79	56.85	64.95	53.75	70.20	52.56	63.38
Two-stage†	56.10	74.19	56.23	65.79	60.04	71.73	53.85	62.34
Dual-MRC†	74.93	76.57	64.97	65.14	75.71	70.84	63.37	64.59
CMLA+†	48.95	70.62	44.60	53.60	50.00	61.20	44.10	56.90
Gen-idx‡	77.68	78.47	67.98	69.95	77.38	75.69	66.11	68.17
GAS-R	**77.75**	79.06	68.46	68.82	76.63	75.73	66.25	65.87
EHG(Ours)	77.17	**79.32**	**69.11**	**70.04**	**78.19**	**77.12**	**69.05**	**68.48**

Table 3: Comparison of F1 scores for AOPE and AESC. The comparison results with "†" are retrieved from Mao et al. (2021), and result with "‡" is from Yan et al. (2021). Bolded fonts for the best performance and underlined fonts for the second-best performance.

Model	Rest15	Rest16
Baseline	-	38.10
TAS-LPM	54.76	64.66
TAS-SW-CRF	57.51	65.89
TAS-SW-TO	58.09	65.44
GAS-R	60.63	68.31
GAS-Para	**63.06**	71.97
EHG(Ours)	60.56	68.56
EHG-Para(Ours)	62.83	**72.09**

Table 4: Comparison of F1 scores for TASD. All comparison results are from Zhang et al. (2021a). Bolded fonts for the best performance and underlined fonts for the second-best performance.

terms) generated from the first four *object queries* in the Rest16 test set under the TASD task. As shown in Figure 5, it can be found that different *object queries* have different location preferences and focus on capturing SEs in different locations.

M	μ_{gen}	TASD	AESC	ASTE	AOPE
16	0.75	**68.56**	76.33	67.87	75.73
16	1	66.13	**77.12**	66.22	74.97
16	1.25	67.32	75.03	**70.02**	75.50
16	1.5	67.15	74.33	67.17	**78.19**
8	1	**66.87**	76.16	**68.09**	**75.86**
16	1	66.13	**77.12**	66.22	74.97
32	1	63.59	74.64	66.79	66.91

Table 5: Comparing the effects of different μ_{gen} and M for different ABSA subtasks.

To evaluate the impact of different μ_{gen} and M settings on different ABSA subtasks, we trained multiple models for the ablation study. All experiments occurred on the Rest16 dataset, as shown in Table 5. It can be found that the effect of different μ_{gen} is different for different subtasks, where TASD performs best at μ_{gen}=0.75, AESC performs best at μ_{gen}=1, ASTE performs best at μ_{gen}=1 and AOPE performs best at μ_{gen}=1.5. From this we conclude that as the proportion of SA in ST increases, μ_{gen} needs to be smaller to obtain better performance. The reason for this phenomenon we believe is that since SAs are usually composed of a small fixed number of words, it is relatively simple for the model to learn the mapping of different SEs to SAs, while the diversity of SEs is greatly increased, which again makes it relatively difficult to generate SEs. For M, we found that all ABSA subtasks performed best when M was set to 8 or 16, and the performance decreased when M was set to 32. Moreover, through data analysis, we found that the average number of STs per sentence in all datasets is about 1.6, and thus we believe that setting M far beyond the actual number of STs is not conducive to model training.

4.6 Error Analysis

To explore the behavior of EHG more intuitively, we performed an error analysis. Specifically, we verified the actual performance of the ASTE task in the test set of Rest16.

First, we find that the largest percentage of prediction errors is caused by the prediction errors of the opinion term, which accounts for about 50 percent of all errors. As in **Example 2** in Figure 6., in many cases we find that EHG has a mislocalization of the opinion term, which often occurs after "and", which we believe is related to the semi-autoregressive decoding. In **Example 2**, the aspect terms before and after "and" point to a same opinion term, but since the decoding of "server" and

Example 1
Sentence: Portions was just enough for me, but may not be for a big eater.
Gold Lable: (Portions , enough , neutral)
Prediction: (Portions , enough , positive)

Example 2
Sentence: While I could have done without the youth who shared the evening with us, our wonderful server and food made the experience a very positive one.
Gold Lable: (server , wonderful , positive); (food , wonderful , positive)
Prediction: (server , wonderful , positive); (food , positive , positive)

Example 3
Sentence: My friend enjoyed the grilled Alaskan King Salmon with delectable creamed Washington russet potatoes and crisp green beans.
Gold Lable: (grilled Alaskan King Salmon , enjoyed , positive); (creamed Washington russet potatoes , delectable , positive); (green beans , crisp , positive)
Prediction: (grilled Alaskan King Salmon , enjoyed , positive); (creamed Washington russet potatoes , delectable ,positive); (green beans , crisp , positive)

Figure 6: Examples containing the input sentence, gold label and predicted.

"food" by EHG is independent, the relationship between them is difficult to be captured. This is also a part worth optimizing in the future.

Second, we found that EHG is error-prone for neutral sentiment mining, which is of course a common problem of many current ABSA methods. We believe that this is a contradictory point in the ABSA problem, as positive or negative sentiment polarity is often highly correlated with the opinion term, while neutral sentiment polarity often requires the understanding of the whole sentence. However, the model tends to learn the correspondence between opinion term and sentiment polarity while ignoring the semantics of the whole sentence during training, which is also worthy of deeper investigation by researchers.

Finally, since EHG uses different *object queries* to capture different sentiment tuples, it is better at gaining advantages in the extraction of multiple sentiment tuples, as demonstrated by **Example 3** in Figure 6 as well as Figure 4.

5 Conclusion

In this paper, we propose a novel Efficient Hybrid Generation framework to model various ABSA subtasks. Specifically, we propose an Efficient Hybrid Transformer to generate sentiment tuples in parallel and simultaneously locate the corresponding sentiment entities. Moreover, we propose a hybrid Hungarian loss to achieve end-to-end model training. Finally, we propose an entity correction module to normalize the location information of sentiment entities. Extensive experiments demonstrate that our proposed EHG framework achieves competitive results while maintaining efficient inference.

6 Limitations

Although our proposed framework can generate semantic representations of sentiment tuples and the corresponding sentiment entity detection in parallel, there are still some limitations.

First, for the case where the input sentence contains only one set of sentiment tuples, our framework does not have a speed advantage over other NLG-based approaches, due to the fact that the efficiency of the proposed framework is based on the parallel generation of multiple sets of sentiment tuples.

In addition, the convergence speed of the proposed framework is slower than other NLG-based methods, and in all experiments we find that the convergence speed is positively related to the number of *object queries*, specifically, the average number of convergence epochs is 35 when the number of *object queries* is set to 8 and 45 when the number of *object queries* is set to 16.

Acknowledgements

We would like to thank all authors for their contributions to this work and all reviewers for their comments.

References

Caroline Brun and Vassilina Nikoulina. 2018. Aspect based sentiment analysis into the wild. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 116–122.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6515–6524.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2509–2518.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6714–6721.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. *arXiv preprint arXiv:1805.00760*.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2886–2892.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Dehong Ma, Sujian Li, and Houfeng Wang. 2018. Joint learning for targeted sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4737–4742.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. *arXiv preprint arXiv:1709.00893*.

Yue Mao, Yi Shen, Jingchao Yang, Xiaoying Zhu, and Longjun Cai. 2022. Seq2path: Generating sentiment tuples as paths of a tree. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2215–2225.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. *arXiv preprint arXiv:2101.00816*.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654.

Radford M Neal. 1992. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *ICML*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Hai Wan, Yufei Yang, Jianfeng Du, Yanan Liu, Kunxun Qi, and Jeff Z Pan. 2020. Target-aspect-sentiment joint detection for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9122–9129.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. *arXiv preprint arXiv:1603.06679*.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Hu Xu, Bing Liu, Lei Shu, and Philip S Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. *arXiv preprint arXiv:1805.04601*.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4755–4766, Online. Association for Computational Linguistics.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. *arXiv preprint arXiv:1805.07043*.

Hang Yan, Junqi Dai, Xipeng Qiu, Zheng Zhang, et al. 2021. A unified generative framework for aspect-based sentiment analysis. *arXiv preprint arXiv:2106.04300*.

Meishan Zhang, Yue Zhang, and Duy Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 612–621.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021a. Aspect sentiment quad prediction as paraphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9209–9219.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021b. Towards generative aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 504–510.

He Zhao, Longtao Huang, Rong Zhang, Quan Lu, and Hui Xue. 2020. Spanmlt: A span-based multi-task learning framework for pair-wise aspect and opinion terms extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3239–3248.

A Appendix

A.1 Hyperparametric ablation studies

As described in the previous section, the proposed framework has five hyperparameters λ_{cls}, λ_{idx}, μ_{cls}, μ_{idx} and μ_{gen}. Among them, μ_{gen} has been discussed in 4.5, and the ablation experiments of the remaining hyperparameters are released here.

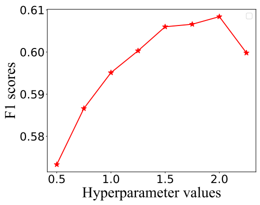


Figure 7: F1 scores under different λ_{cls} settings.

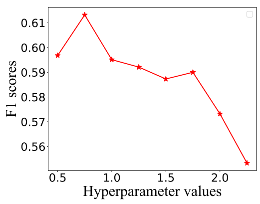


Figure 8: F1 scores under different μ_{cls} settings.

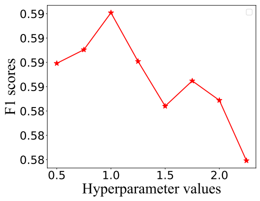


Figure 9: F1 scores under different λ_{idx} settings.

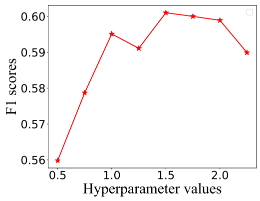


Figure 10: F1 scores under different μ_{idx} settings.

To facilitate the adjustment, we fixed all the remaining hyperparameters to 1 when adjusting a particular hyperparameter, and we set M=8 for all experiments in order to obtain results quickly. All experiments occurred in the ASTE task on the Rest15 dataset. Figures 7, 8, 9, 10 show the curves of F1 variation with λ_{cls}, μ_{cls}, λ_{idx} and μ_{idx}, respectively, from which we use λ_{cls}=2, λ_{idx}=1, μ_{cls}=0.75 and μ_{idx}=1.5 for all experiments.

A.2 Training details

Subtask	Rest14	Rest15	Rest16	Laptop14
AE	40	40	-	40
OE	40	40	-	40
ASTE	50	45	50	40
AOPE	50	55	55	50
AESC	45	45	45	45
TASD	-	40	55	-

Table 6: Training epochs of EHT under different ABSA subtasks and different datasets.

Since our comparison experiments consist of multiple versions of the same dataset, we give the number of training rounds for all ABSA subtasks under different datasets, which can be found in Table 6.

What's New? Summarizing Contributions in Scientific Literature

Hiroaki Hayashi[1] **Wojciech Kryściński**[1] **Bryan McCann**[2]
Nazneen Rajani[3] **Caiming Xiong**[1]
[1]Salesforce Research [2]You.com [3]Hugging Face
hiroakihayashi@salesforce.com

Abstract

With thousands of academic articles shared on a daily basis, it has become increasingly difficult to keep up with the latest scientific findings. To overcome this problem, we introduce a new task of *disentangled paper summarization*, which seeks to generate separate summaries for the paper contributions and the context of the work, making it easier to identify the key findings shared in articles. For this purpose, we extend the S2ORC corpus of academic articles, which spans a diverse set of domains ranging from economics to psychology, by adding disentangled "contribution" and "context" reference labels. Together with the dataset, we introduce and analyze three baseline approaches: 1) a unified model controlled by input code prefixes, 2) a model with separate generation heads specialized in generating the disentangled outputs, and 3) a training strategy that guides the model using additional supervision coming from inbound and outbound citations. We also propose a comprehensive automatic evaluation protocol which reports the *relevance*, *novelty*, and *disentanglement* of generated outputs. Through a human study involving expert annotators, we show that in 79%, of cases our new task is considered more helpful than traditional scientific paper summarization.

1 Introduction

With the growing popularity of open-access academic article repositories, such as arXiv or bioRxiv, disseminating new research findings has become nearly effortless. Through such services, tens of thousands of scientific papers are shared by the research community every month[1]. At the same time, the unreviewed nature of mentioned repositories and the sheer volume of new publications has made it nearly impossible to identify relevant work and keep up with the latest findings.

Scientific paper summarization, a subtask within automatic text summarization, aims to assist researchers in their work by automatically condensing articles into a short, human-readable form that contains only the most essential information. In recent years, abstractive summarization, an approach where models are trained to generate fluent summaries by paraphrasing the source article, has seen impressive progress. State-of-the-art methods leverage large, pre-trained models (Lewis et al., 2020; Liu et al., 2022), define task-specific pre-training strategies (Zhang et al., 2019), and scale to long input sequences (Zhao et al., 2020; Zaheer et al., 2020). Available large-scale benchmark datasets, such as arXiv and PubMed (Cohan et al., 2018), were automatically collected from online archives and repurpose paper abstracts as reference summaries. However, the current form of scientific paper summarization where models are trained to generate paper abstracts has two caveats: 1) often, abstracts contain information which is not of primary importance, 2) the vast majority of scientific articles come with human-written abstracts, making the generated summaries superfluous.

To address these shortcomings, we introduce the task of disentangled paper summarization. The new task's goal is to generate two summaries simultaneously, one strictly focused on the summarized article's novelties and contributions, the other introducing the context of the work and previous efforts. In this form, the generated summaries can target the needs of diverse audiences: senior researchers and field-experts who can benefit from reading the summarized contributions, and newcomers who can quickly get up to speed with the intricacies of the addressed problems by reading the context summary and get a perspective of the latest findings from the contribution summary.

For this task, we introduce a new large-scale dataset by extending the S2ORC (Lo et al., 2020) corpus of scientific papers, which spans multiple

---

[1] `https://arxiv.org/stats/monthly_submissions`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1019–1031
May 2-6, 2023

scientific domains and offers rich citation-related metadata. We organize and process the data, and extend it with automatically generated contribution and context reference summaries, to enable supervised model training. We also introduce three abstractive baseline approaches: 1) a unified, controllable model manipulated with descriptive control codes (Fan et al., 2018; Keskar et al., 2019; Liu et al., 2021; He et al., 2022), 2) a one-to-many sequence model with a branched decoder for multihead generation (Luong et al., 2016; Guo et al., 2018), and 3) an information-theoretic training strategy leveraging supervision coming from the citation metadata (Peyrard, 2019). To benchmark our models, we design a comprehensive automatic evaluation protocol that measures performance across three axes: relevance, novelty, and disentanglement. We thoroughly evaluate and analyze the baselines models and investigate the effects of the additional training objective on the model's behavior. To motivate the usefulness of the newly introduced task, we conducted a human study involving human annotators in a hypothetical paper-reviewing setting. The results find disentangled summaries more helpful in 79% of cases in comparison to abstract-oriented outputs. Code and data preparation scripts introduced in this work are available at `https://github.com/salesforce/disentangled-sum`.

2 Related Work

Recent trends in abstractive text summarization show a shift of focus from designing task-specific architectures trained from scratch (See et al., 2017; Paulus et al., 2018) to leveraging large-scale Transformer-based models pre-trained on vast amounts of data (Liu and Lapata, 2019; Lewis et al., 2020), often in multi-task settings (Raffel et al., 2019). A similar shift can be seen in scientific paper summarization, where state-of-the-art approaches utilize custom pre-training strategies (Zhang et al., 2019) and tackle problems of summarizing long documents (Zhao et al., 2020; Zaheer et al., 2020). Other methods, at a smaller scale, seek to utilize the rich metadata associated with scientific articles and combine them with graph-based methods (Yasunaga et al., 2019). In this work, we combine these two lines of work and propose models that benefit from pre-training procedures, but also take advantage of task-specific metadata.

Popular large-scale benchmark datasets in scientific paper summarization (Cohan et al., 2018; Cachola et al., 2020) were automatically collected from open-access paper repositories and considered article abstracts as the reference summaries. Other forms of supervision have also been investigated for the task, including author-written highlights (Collins et al., 2017) and annotations (Meng et al., 2021), citations (Yasunaga et al., 2019), and transcripts from conference presentations of the articles (Lev et al., 2019). In contrast, we introduce a large-scale automatically collected dataset with more fine-grained references than abstracts, which also offers rich citation-related metadata.

Update summarization (Dang and Owczarzak, 2008) defines a setting in a collection of documents with partially overlapping information is summarized, some of which are considered prior knowledge. The goal of the task is to focus the generated summaries on the novel information. Work in this line of research mostly focuses on novelty detection in news articles (Bysani, 2010; Delort and Alfonseca, 2012) and timeline summarization (Martschat and Markert, 2018; Chang et al., 2016) on news and social media domains. Here, we propose a novel task that is analogous to update summarization in that it also requires contrasting the source article with the content of other related articles which are considered pre-existing knowledge.

3 Task

Given a source article D, the goal of disentangled paper summarization is to simultaneously summarize the *contribution* y_{con} and *context* y_{ctx} of the source article. Here, contribution refers to the novelties introduced in the article D, such as new methods, theories, or resources, while context represents the background of the work D, such as a description of the problem or previous work on the topic. The task inherently requires a relative comparison of the article with other related papers to effectively disentangle its novelties from pre-existing knowledge. Therefore, we also consider two sets of citations: inbound citations C_I and outbound citations C_O as potential sources of useful information for contrasting the article D with its broader field. Inbound citations refer to the set of papers that cite D, *i.e.* relevant future papers, while outbound citations are the set of papers that D cites, *i.e.* relevant previous papers. With its unique set of goals, the task of disentangled paper summarization poses a novel set of challenges for automatic summarization systems to overcome: 1) identifying salient

Dataset	#Examples	Avg. #Tokens				
		Paper D	Inbound C_I	Outbound C_O	Contribution y_{con}	Context y_{ctx}
ArXiv (Train)	203037	4938	-	-	220 (Total summary)	
PubMed (Train)	119924	3016	-	-	203 (Total summary)	
Ours - Train	805152	6351	925	877	136	236
Valid	36129	6374	922	875	135	236
Test	54242	6350	927	892	136	237

Table 1: Token length statistics on the training split of our dataset compared to existing scientific paper summarization datasets. Contribution summaries tend to be shorter than context summaries.

content of D and related papers from C_I and C_O, 2) comparing the content of D with each document from the citations, and 3) summarizing the article along the two axes: contributions and context.

3.1 Dataset

Current benchmark datasets used for the task of scientific paper summarization, such as arXiv and PubMed (Cohan and Goharian, 2015), are limited in size, the number of domains, and lack of citation metadata. Thus, we construct a new dataset based on the S2ORC (Lo et al., 2020) corpus, which offers a large collection of scientific papers spanning multiple domains along with rich citation-related metadata, such as citation links between papers and annotated citation spans. Specifically, we curate the data available in the S2ORC corpus and extend it with new reference labels.

Data Curation Some papers in the S2ORC corpus[2] do not contain a complete set of information required by our summarization task: paper text, abstract, and citation metadata. We remove such instances and construct a paper summarization dataset in which each example a) has an abstract and body text, and b) has at least 5 or more inbound and outbound citations, C_I and C_O respectively. In cases where a paper has more than K incoming or outgoing citations, we first sample K citations for each of incoming and outgoing citations and sort them in descending order by the number of their respective citation from and to the target paper. K is a hyperparameter and we choose $K = 20$ in this work.[3]

Citation Span Extraction Each article in the set of inbound and outbound citations can be represented by its full text, abstract, or the span of text associated with the citation. In this study, we follow Qazvinian and Radev (2008) and Cohan and Goharian (2015) in representing citations with the sentences in which the citation occurs.[4] Thus, an outbound citation is represented by a sentence from the source paper. Usually, such sentences directly refer to the cited paper and place its content in relation to the source paper. Analogously, an inbound citation is represented by sentences from the citing paper and relates its content with the source paper.

Reference Generation Our approach relies on the availability of reference summaries for both contributions and contexts. However, such annotations are not provided or easily extractable from the S2ORC corpus, and collecting expert annotations is infeasible due to the associated costs. Therefore, we apply a data-driven approach to automatically extract contribution and context reference summaries from the available paper abstracts. First, we manually label 400 abstracts sampled from the training set. Annotations are done on a sentence-level with binary labels indicating *contribution*- and *context*-related sentences.[5] This procedure yields 3341 sentences with associated binary labels, which we refer to as golden standard references. Next, we fine-tune an automatic sentence classifier using the golden standard data. As our classifier we use SciBERT (Beltagy et al., 2019), which after fine-tuning achieves 86.3% accuracy and 0.932 for Area under ROC curve in classifying *contribution* and *context* sentences on a held-out test set. Finally, we apply the fine-tuned classifier to generate reference labels for all examples in our dataset, which we refer to as silver standard references. The statistics of the resulting dataset are shown in Table 1.

4 Models

Our goal is to build an abstractive summarization system which has the ability to generate contribu-

---

[2] Release ID: 20190928.

[3] Among other values, we experimented with 20, as the trade-off between computational cost and rich citation contexts.

[4] If a publication is cited multiple times within a source article we concatenate all relevant sentences.

[5] Sentences not labeled as contribution are considered context, we leave finer-grained labels for future work.

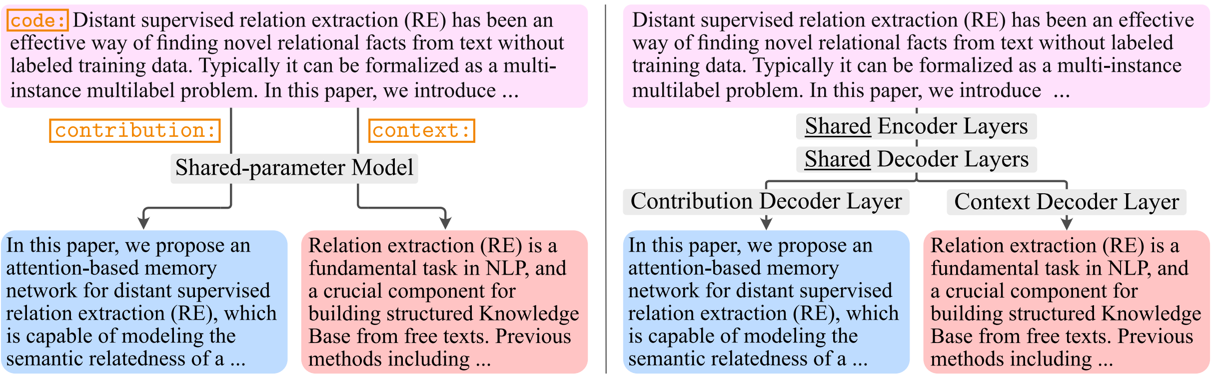


Figure 1: Model diagram. Left: CONTROLCODE model, in which inputs are prefixed with a prompt symbol and passed to a shared model to control the output mode. Right: MULTIHEAD model, which shares all of the model's parameters apart from the last decoder layer for different output modes, and chooses the final decoder layer accordingly to control the output mode.

tion and context summaries based on the source article. To achieve the necessary level of controllability, we propose two independent approaches building on encoder-decoder architectures:

CONTROLCODE (CC) A common approach to controlling model-generated text is by conditioning the generation procedure on a control code associated with the desired output. Previous work on controllable generation (Fan et al., 2018; Keskar et al., 2019; Chan et al., 2021; Luo et al., 2022) showed that prepending a special token or descriptive prompt (Liu et al., 2021) to the model's input during training and inference is sufficient to achieve fine-grained control over the generated content. Following this line of work, we modify our training instances by prepending textual control codes, `contribution:` or `context:`, to the summarized articles. During training, all model parameters are updated for each data instance and the model is expected to learn to associate the provided prompt with the correct output mode. The approach does not require changes in the architecture, making it straightforward to combine with existing large-scale, pre-trained models. The architecture is shown on the left of Figure 1.

MULTIHEAD (MH) An alternative way of controlling generation is by explicitly allocating layers within the model specifically for the desired control aspects. Prior work investigating multi-task models (Luong et al., 2016; Guo et al., 2018) showed the benefits of combining shared and task-specific layers within a single, multi-task architecture. Here, the encoder shares all parameters between the two generation modes, while the decoder shares all parameters, apart from the final layer, which splits into two generation branches. During training, each branch is individually updated with gradients from the associated mode. The model shares the softmax layer weights between the output branches under the assumption that token-level vocabulary distributions are similar in the two generation modes due to the common domain. This approach is presented on the right of Figure 1.

4.1 Informativeness-guided Training

In addition to supervising the models with gold-standard summaries, we consider guiding them to generate summaries with a focus on contributions using *informativeness* (Peyrard, 2019). Specifically, informativeness measures the user's degree of surprise after reading a summary given their background knowledge, and can be formally defined as:

$$Inf(D, K) = -\sum_i P_D(\omega_i) \log P_K(\omega_i), \quad (1)$$

where ω_i is a primitive semantic unit, P_K is the probability over the unit under the user's knowledge, P_D is the probability over the unit with respect to the source document, and i is an index over all semantic units within a summary.

Since paper contributions are novel contents introduced to the community, we argue that contributions cause surprisal given the general knowledge about the state of the field. Quantified such surprisal as informativeness, we explore utilizing this measure as an auxiliary objective that is optimized during training. We define the semantic unit ω_i as the summary itself,[6] which enables a simple interpretation of the corresponding probabil-

---

[6] For simplicity in modeling, we chose the entire summary. However, this goes against the requirement set by Peyrard (2019) that ω_i is a *primitive* semantic unit, because a paragraph's meaning can be decomposed into higher granular units.

ities. We estimate P_D as the likelihood of the summary given the paper content, $P_D(\omega_i) = p(y \,|\, D)$. Since each paper is associated with a unique context and background knowledge, we treat the background knowledge as all relevant papers published before the source paper, *i.e.*, outbound citations C_O. Therefore, P_K is estimated as the likelihood of the summary given the previous work, $P_K(\omega_i) = p(y \,|\, C_O)$. We formulate the informativeness function as:

$$Inf(D, K) = \begin{cases} -p(y_{con} \,|\, D) \log p(y_{con} \,|\, C_O) \\ -p(y_{ctx} \,|\, D) \log p(y_{ctx} \,|\, C_I) \end{cases}, \tag{2}$$

where the conditioning depends on the generation mode of the model, and aim to maximize it during the training procedure. The estimation of each term is done by a forward pass on the model with corresponding input and output pairs: $p(y_{con} \,|\, C_O)$ is computed by estimating the probability of y_{con} when feeding C_O as the encoder input.

Combined with a cross entropy loss L_{CE}, we obtain the final objective which we aim to the minimize during training:

$$L = L_{CE} - \lambda\, Inf(D, K), \tag{3}$$

where λ is a scaling hyperparameter determined through cross-validation. Note that C_I, C_O are only used during training. Models trained with this objective is applicable to papers without citation information at inference time.

5 Experiments and Results

In this section, we describe the experimental environment and report automatic evaluation results. We consider four model variants:

- **CC**, **CC+INF**: CONTROLCODE model without and with the informativeness objective,
- **MH**, **MH+INF**: MULTIHEAD model without and with the informativeness objective.

5.1 Evaluation

We perform automatic evaluation of the system outputs (s_{con}, s_{ctx}) against the silver standard references (y_{con}, y_{ctx}). For this purpose, we have designed a comprehensive evaluation protocol, shown in Figure 2, based on existing metrics that evaluates the performance of models across 3 dimensions:

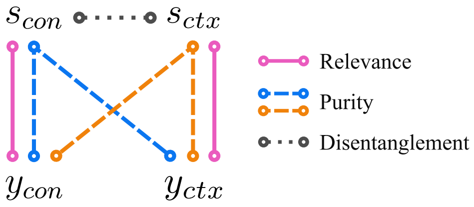


Figure 2: Diagram illustrating the evaluation protocol assessing summaries along 3 axes: relevance, purity, and disentanglement.

Relevance Generated summaries should closely correspond with the available reference summaries. We measure the lexical overlap and semantic similarity between (s_{con}, y_{con}) and (s_{ctx}, y_{ctx}) using ROUGE (R-i) (Lin, 2004) and BERTScore (Zhang et al. 2020; BS), respectively.

Purity Generated contribution summary should closely correspond with its respective reference summary, but should not overlap with the context reference summary. We measure the lexical overlap between s_{con} and $(y_{con},\, y_{ctx})$ using NouveauROUGE $_{con}$ (N$_{con}$-i) (Conroy et al., 2011). The metric reports an aggregate score defined as a linear combination between the two components:

$$\begin{aligned} \text{NouveauROUGE}_{con}\text{-}i = \alpha_0^i \\ + \alpha_1^i \text{ROUGE-}i(s_{con}, y_{con}) \\ + \alpha_2^i \text{ROUGE-}i(s_{con}, y_{ctx}), \end{aligned}$$

where weights α_j^i were set by the original authors to favor outputs with maximal and minimal overlap with related and unrelated references, accordingly. Analogously, we calculate N$_{ctx}$-i in reverse direction between s_{ctx} and $(y_{ctx},\, y_{con})$. Purity P-i is defined as the average novelty in both directions:

$$\text{Purity-}i = (\text{N}_{con}\text{-}i + \text{N}_{ctx}\text{-}i)/2; \quad (\text{P-}i).$$

Disentanglement Generated contribution and context summaries should have minimal overlap. We measure the degree of lexical overlap and semantic similarity between (s_{con}, s_{ctx}) using ROUGE and BERTScore, respectively. To maintain consistency across metrics (higher is better) we report disentanglement scores as complements of the associated metrics:

$$\begin{aligned} \text{DisROUGE-}i = 100 - \text{ROUGE-}i; \quad (\text{D-}i), \\ \text{DisBERTScore} = 100 - \text{BERTScore}; \quad (\text{DBS}). \end{aligned}$$

5.2 Results

In Table 2 we report results from the automatic evaluation protocol described in Subsection 5.1.

Model		Relevance				Purity		Disentanglement			
		R-1	R-2	R-L	BS	P-1	P-2	D-1	D-2	D-L	DBS
CC	Con	39.16	15.96	24.65	63.22	2.77	3.69	52.95	72.18	69.12	33.62
	Ctx	41.84	17.24	24.55	63.78						
CC+INF	Con	38.92	15.95	24.65	62.94	2.75	3.68	53.68	71.97	68.46	34.09
	Ctx	41.49	17.03	24.50	63.40						
MH	Con	39.20	15.98	24.72	63.04	2.73	3.68	50.89	69.51	65.97	32.51
	Ctx	41.67	17.23	24.65	63.77						
MH+INF	Con	38.74	15.90	24.59	62.70	2.68	3.60	53.35	71.47	67.20	33.86
	Ctx	40.39	16.31	23.83	62.85						

Table 2: Automatic evaluation results on the test set. For all metrics, higher values indicate better results. Con and Ctx refer to contribution summary and context summary, respectively. Purity and Disentanglement are measaured on the pairs of contribution and context summaries.

Original Abstract: Energy optimization in buildings by controlling the Heating Ventilation and Air Conditioning (HVAC) system is being researched extensively. In this paper, a model-free actor-critic Reinforcement Learning (RL) controller is designed using a variant of artificial recurrent neural networks called Long-Short-Term Memory (LSTM) networks. Optimization of thermal comfort alongside energy consumption is the goal in tuning this RL controller. The test platform, our office space, is designed using SketchUp. Using OpenStudio, the HVAC system is installed in the office. The control schemes (ideal thermal comfort, a traditional control and the RL control) are implemented in MATLAB. Using the Building Control Virtual Test Bed (BCVTB), the control of the thermostat schedule during each sample time is implemented for the office in EnergyPlus alongside local weather data. Results from training and validation indicate that the RL controller impoves thermal comfort by an average of 15% and energy efficiency by an average of 2.5% as compared to other strategies mentioned.

Generated Abstract: Despite the advances in research on HVAC control algorithms, most field equipment is controlled using classical methods that include hysteresis/on/off and Proportional Integral and Derivative (PID) controllers. These classical methods do not perform optimally. The high thermal inertia of buildings induces large time delays in the building dynamics, which cannot be handled efficiently by the simple on/off controllers. However, due to the high non-linearity in building dynamics coupled with uncertainties such as weather, energy pricing, etc., these PID controllers require extensive retuning or auto-tuning capabilities, which increases the difficulty and complexity of the control problem. In this work, we introduce novel control algorithms from a branch of machine learning called reinforcement learning. From a controls perspective, reinforcement learning algorithms can be considered as direct adaptive optimal control. Like optimal control, reinforcement training algorithms minimize the cumulative sum of costs over a time horizon. Unlike traditional optimization algorithms can learn optimal control actions

---

Contribution: In this work, we introduce novel control algorithms from a branch of machine learning called reinforcement learning. In our current approach, the impetus is thermostat control. Instead of traditional on/off heating and cooling control, reinforcement learning is utilized to set this schedule to obtain improved Predicted Mean Vote (PMV)-based thermal comfort at an optimal energy expenditure. Hence, a thermostats schedule is computed using an RL controller. The results show that the Q-learning algorithm can learn to adapt to time-varying and nonlinear system dynamics without explicit identification of the plant model in both systems and controls.

Context: The Heating, Ventilation and Air Conditioning (HVAC) systems can account for up to 50% of total building energy demand. In the hopes of moving toward a greener, more energy-efficient future, a significant improvement in energy efficiency is needed to achieve this goal. Despite the advances in research on HVAC control algorithms, most field equipment is controlled using classical methods that include hysteresis/on/off and Proportional Integral and Derivative controllers. However, due to the high nonlinearity in building dynamics coupled with uncertainties such as weather, energy pricing, etc., these PID controllers require extensive retuning or auto-tuning capabilities, which increases the difficulty and complexity of the control problem. The high thermal inertia of buildings induces large time delays in the building dynamics, which cannot be handled efficiently by the simple on/off controllers.

Table 3: Generated samples compared with the original and generated abstracts of the associated paper. The second rows shows the output decoded from DistilBART fine-tuned on our dataset, the third rows shows the outputs from CONTROLCODE model. Our model successfully generates disentangled content, thus making it easier to follow than the abstract.

Relevance Across most models and metrics, relevance scores for context generation are higher than those for contribution summarization. Manual inspection revealed that in some cases generated context summaries also include article contribution information, while this effect was not observed in the reverse situation. Considering that silver standard annotations may contain noisy examples with incorrectly separated references, we suspect that higher ROUGE scores for context summaries may be caused by noisy predictions coinciding with noisy references. Examples of such summaries are shown in the Appendix C. We also observe that informativeness-guided models (+INF) perform on par with their respective base versions, and the additional training objective does not affect the performance on the relevance metric. This insight corroborates with Peyrard (2019) who defines informativeness and relevance as orthogonal criteria.

Purity While the informativeness objective was designed to improve the novelty of generated summaries, results show an opposite effect, where informativeness-guided models slightly underperform their base counterparts. The true reason for such behavior is unknown, however, it might be an indicator that the outbound citations C_O are not a good approximation of reference context summaries y_{ctx}, or the relationship between the two is weak. This effect is more evident in the Medical and Biology domains, which are the two most frequent domains in the dataset.

Disentanglement Results indicate that CONTROLCODE-based models perform better than MULTIHEAD approaches in terms of generating disentangled outputs. This comes as a surprise given that the CC models share all parameters between the two generation modes, but might indicate that the two tasks contain complementary training signals. We also noticed that, both informativeness-guided models performed better in terms of D-1.

Based on both purity and disentanglement evaluations, we suspect that the informativeness objective does guide the models to output more disentangled summaries (second term in Eq 2), but the signal is not strong enough to focus on generating the appropriate content (first term in Eq 2). It is also clear that the MULTIHEAD model benefits more from the additional training objective.

6 Analysis

6.1 Qualitative Analysis

To better understand the strengths and shortcomings of our models, we performed a qualitative study of model outputs. Table 3 shows an example of generated summaries compared with the original abstract of the summarized article. Our model successfully separates the two generation modes and outputs coherent and easy to follow summaries. The contribution summary clearly lists the novelties of the work, while the context summary introduces the task at hand and explains its importance. In comparison, the original abstract briefly touches on many aspects: the context, methods used, and contributions, but also offers details that are not of primary importance, such as the detailed about the simulation environment.

More generally, the described trends hold across summaries generated by our models. The model outputs are fluent, abstractive, offer good separation between modes, and are on topic. An artifact noticed in a few instances of the inspected outputs was leakage of contribution information into context summaries. Factual correctness of summaries is discussed in Section 6.4. Other examples of generated summaries are included in the Appendix C.

6.2 Per-domain Performance

Taking advantage of the rich metadata associated with the S2ORC corpus, we analyze the performance of models across the 10 most frequent scientific domains. Table 4 shows the results of contribution summarization using the CONTROLCODE[7] model. While ROUGE-1 scores oscillate around 40 points for most academic fields, the results indicate that summarizing documents from the Medical domain is particularly difficult, with models scoring about 7 points below average. Manual inspection of instances with low scores (R-1 < 20), exposed that contribution summaries in the Medical domain are highly quantitative (*e.g.* “Among these treated ... retinopathy was noted in X%”). While other domains such as Biology also suffer from the same phenomenon, low-scoring quantitative summaries were 1.9 times more frequent in Medicine than in Biology. An investigation into the domain distribution in our dataset (Appendix) revealed that Biology and Medicine are the two best represented fields in the corpus, with Biology having over twice as many examples. We hypothesize that the poor performance of models stems from the fact that generating such quantitative summaries requires a deeper, domain-specific understanding of the source document and the available in-domain training data is insufficient to achieve that goal.

Metric	R-1	R-2	R-L	BS
Biology	40.63	17.01	25.59	64.23
Medicine	33.97	13.08	21.73	61.75
Mathematics	40.13	15.56	24.42	61.58
Computer science	43.54	16.41	25.86	63.43
None	40.31	18.14	26.68	64.00
Psychology	39.51	15.56	24.34	62.95
Physics	40.09	15.85	24.89	62.10
Chemistry	40.44	17.77	26.14	63.93
Economics	39.56	14.25	23.41	60.91
Materials science	42.52	18.96	27.57	65.25

Table 4: Relevance evaluation of contribution summaries for the top 10 domains generated using the CONTROLCODE model. Performance on Medicine domain is paricularly low.

Dataset	A1	A2	A3	AVG.
S2ORC	82%	78%	70%	77%
CORD	88%	76%	78%	81%

Table 5: Usefulness of disentangled summaries in percentage, *e.g.*, Annotator 1 (A1) chose the disentangled summaries 82% out of all the samples from S2ORC.

6.3 Human Evaluation of Usefulness

To assess the usefulness of the newly introduced task to the research community, we conducted a human study involving expert annotators. The study aimed to compare disentangled papers summaries with traditional, abstract-based summaries in a hypothetical paper reviewing setting. Judges were shown both types of summaries side by side and asked to pick one which would be more helpful for conducting the paper review.[8] Abstract-based sum-

---

[7] The remaining models exhibit the same pattern.

[8] We include a screen shot of the annotation user interface in Appendix B.

maries were generated by a model with a configuration identical to the models previously introduced in this work, trained to generate full abstracts using the same training corpus. Annotators that participated in this study hold graduate degrees in technical fields and are active in the research community, however, they were not involved or familiar with this work prior to this experiment. The study used 100 examples, out of which 50 were decoded on the test split of the adapted S2ORC dataset, while the other 50 were generated in a zero-shot fashion from articles in the CORD dataset (Wang et al., 2020), a recently introduced collection of papers related to COVID-19. The inter-annotator agreement measured by Fleiss' Kappa were 0.41 and 0.33 for the S2ORC and CORD datasets, respectively. Results in Table 5 show the proportion of all examples where the annotators preferred the disentangled summaries over the generated abstracts. The numbers indicate a strong preference from the judges for disentangled summaries, in the case of both S2ORC and CORD examples. The values on CORD samples are slightly higher than those on S2ORC; we suspect this being due to the fact that the annotators were less familiar with the topics described in Covid-related publications and would require more help to review such articles.

6.4 Factuality of Generated Summaries

In the scientific literature domain, the truthfulness of generated summaries with the input articles is a crucial aspect. Thus, we measured the factual consistency of 10 pairs of summaries sampled from Computer Science domain by assessing the validity of each *sentence* in the generated summaries against the input article and representing truthfulness as the proportion of sentences deemed consistent. We compared summaries from CC+INF (contribution, context) and DistilBART, resulting in 57, 71, 67 sentences to evaluate, respectively. As shown in Table 7, we found that most sentences copy segments from various positions in the input and lightly paraphrased or fused them together, which led to high percentage of factually consistent sentences.

6.5 Evaluation against Gold Annotations

As discussed in Section 3.1, contribution and context labels are assigned automatically using a data-driven classifier, which could introduce errors in the process. Therefore, we created a gold standard evaluation set by manually annotating 100 samples from the test set and report the evaluation results in Table 6. A sharp drop in ROUGE scores for the context summaries is caused by some examples receiving zero scores for generating context summaries when the manual annotators judged that there are not existent. The overall trend of CONTROLCODE model outpeforming MULTIHEAD model is still observed in the evaluation. More importantly, a reverse tendency is noticeable when the two models are trained with the informativeness objective. Specifically, the MULTIHEAD model showed significant improvement in terms of novelty and disentanglement.

7 Conclusions

In this paper, we propose *disentangled paper summarization*, a new task in scientific paper summarizing where models simultaneously generate contribution and context summaries. With the task in mind, we introduced a large-scale dataset with fine-grained reference summaries and rich metadata. Along with the data, we introduced three abstractive baseline approaches to solving the new task and thoroughly assessed them using a comprehensive evaluation protocol design for the task at hand. Through human studies involving expert annotators with motivated the usefulness of the task in comparison to the current scientific paper summarization setting. Together with this paper, we release the code, trained model checkpoints, and data preprocessing scripts to support future work in this direction. We hope this work will positively contribute to creating AI-based tools for assisting scientists in the research process.

Limitations

The importance of factuality in scientific literature summarization is crucial, as the summaries will serve as evidence for scientific discussion and citations. While our human annotation showed over 90% of the summary sentences were truthful to the input documents, the gap needs to be filled towards zero non-truthful sentences. In addition, failing to construct discourse-level coherence is another factor non-truthful summaries, which our models do not take into account explicitly.

Our models are developed upon DistilBART, which can process up to 1024 tokens for input. However, scientific documents (such as an 8-page paper like this submission) tend to exceed 1024 tokens. While there might be tendencies as to

Model		Relevance				Purity		Disentanglement			
		R-1	R-2	R-L	BS	P-1	P-2	D-1	D-2	D-L	DBS
CC	Con	39.37	15.86	24.73	63.28	2.30	3.22	52.81	71.52	68.36	33.05
	Ctx	30.59	11.22	19.08	55.76						
CC+INF	Con	38.38	15.21	23.47	62.59	2.17	3.10	52.49	69.64	66.60	32.76
	Ctx	30.14	11.10	19.00	55.55						
MH	Con	38.63	15.53	24.68	62.84	2.21	3.13	49.62	67.45	64.43	31.39
	Ctx	29.82	10.61	18.51	55.24						
MH+INF	Con	39.43	15.75	24.77	63.11	2.26	3.13	51.56	68.57	64.97	32.35
	Ctx	29.14	10.25	18.48	54.92						

Table 6: Automatic evaluation results on 100 samples from the test set with **manual** annotation on contributions. For all metrics, higher values indicate better results.

Model		Truthfulness [%]
CC+INF	Con	93.0
	Ctx	90.1
DistilBART	-	91.0

Table 7: Proportion of factually consistent sentences in the summaries.

which sections likely discuss contributions or backgrounds, it is important to consider all the documents whenever possible, not to mention those of cited documents. Efficient incorporation of relevant documents at scale (*e.g.*, efficient attention, retrieval-augmented generation) is an active research area and should be considered for future work.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Praveen Bysani. 2010. Detecting novelty in the context of progressive summarization. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2, 2010, Los Angeles, California, USA - Proceedings of the Student Research Workshop*, pages 13–18. The Association for Computational Linguistics.

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel Weld. 2020. TLDR: Extreme summarization of scientific documents. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4766–4777, Online. Association for Computational Linguistics.

Hou Pong Chan, Lu Wang, and Irwin King. 2021. Controllable summarization with constrained Markov decision process. *Transactions of the Association for Computational Linguistics*, 9:1213–1232.

Yi Chang, Jiliang Tang, Dawei Yin, Makoto Yamada, and Yan Liu. 2016. Timeline summarization from social media with life cycle models. In *IJCAI*, pages 3698–3704.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Arman Cohan and Nazli Goharian. 2015. Scientific article summarization using citation-context and article's discourse structure. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 390–400. Association for Computational Linguistics.

Ed Collins, Isabelle Augenstein, and Sebastian Riedel. 2017. A supervised approach to extractive summarisation of scientific papers. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 195–205, Vancouver, Canada. Association for Computational Linguistics.

John M. Conroy, Judith D. Schlesinger, and Dianne P. O'Leary. 2011. Nouveau-ROUGE: A novelty metric for update summarization. *Computational Linguistics*, 37(1):1–8.

Hoa Trang Dang and Karolina Owczarzak. 2008. Overview of the tac 2008 update summarization task.

Jean-Yves Delort and Enrique Alfonseca. 2012. Dualsum: a topic-model based approach for update summarization. In *EACL 2012, 13th Conference of the European Chapter of the Association for Computational Linguistics, Avignon, France, April 23-27,* *2012*, pages 214–223. The Association for Computer Linguistics.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2020. Summeval: Re-evaluating summarization evaluation. *arXiv preprint arXiv:2007.12626*.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2018. Soft layer-specific multi-task summarization with entailment and question generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 687–697. Association for Computational Linguistics.

Junxian He, Wojciech Kryscinski, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2022. CTRLsum: Towards generic controllable text summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5879–5915, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Guy Lev, Michal Shmueli-Scheuer, Jonathan Herzig, Achiya Jerbi, and David Konopnicki. 2019. TalkSumm: A dataset and scalable annotation method for scientific paper summarization based on conference talks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2125–2131, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. BRIO: Bringing order to abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2890–2903, Dublin, Ireland. Association for Computational Linguistics.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2022. Readability controllable biomedical document summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4667–4680, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Sebastian Martschat and Katja Markert. 2018. A temporally sensitive submodularity framework for timeline summarization. In *Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018, Brussels, Belgium, October 31 - November 1, 2018*, pages 230–240. Association for Computational Linguistics.

Rui Meng, Khushboo Thaker, Lei Zhang, Yue Dong, Xingdi Yuan, Tong Wang, and Daqing He. 2021. Bringing structure into summaries: a faceted summarization dataset for long scientific documents. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1080–1089, Online. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Maxime Peyrard. 2019. A simple theoretical model of importance for summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1059–1073, Florence, Italy. Association for Computational Linguistics.

Vahed Qazvinian and Dragomir R. Radev. 2008. Scientific paper summarization using citation summary networks. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 689–696, Manchester, UK. Coling 2008 Organizing Committee.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, et al. 2020. Cord-19: The covid-19 open research dataset. *arXiv preprint arXiv:2004.10706*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Michihiro Yasunaga, Jungo Kasai, Rui Zhang, Alexander Fabbri, Irene Li, Dan Friedman, and Dragomir Radev. 2019. ScisummNet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *Proceedings of AAAI 2019*.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yao Zhao, Mohammad Saleh, and Peter J Liu. 2020. Seal: Segment-wise extractive-abstractive long-form text summarization. *arXiv preprint arXiv:2006.10213*.

A Implementation Details

Our models build upon DistilBART[9] (Sanh et al., 2019; Wolf et al., 2019), a Transformer-based (Vaswani et al., 2017), pre-trained sequence-to-sequence architecture distilled from BART (Lewis et al., 2020). Specifically, we used a model with 6 self-attention layers in both the encoder and Decoder. Weights were initialized from a model fine-tuned on a news summarization task that are available at `https://huggingface.co/sshleifer/student_cnn_6_6`. For the MultiHead model, the final layer of the decoder was duplicated and initialized with identical weights. We fine-tuned on the training set for 80000 gradient steps on one NVIDIA V100 GPU with a fixed learning rate of 3.0×10^{-5} and choose the best checkpoints in terms of ROUGE-1 scores on the validation set. The loss scaling hyperparameter λ (Eq. 3) was set to 0.05 and 0.01 for the ControlCode and MultiHead models, accordingly. Input and output lengths were set to 1024 and 200, respectively. At inference time, we decoded using beam search with beam size 5. The evaluation was performed using SummEval toolkit (Fabbri et al., 2020).

B Human Evaluation of Disentanglement

In addition to various automatic evaluation, we perform human evaluation on disentanglement to understand which models human annotators prefer. We use Best-Worst scaling (Kiritchenko and Mohammad, 2017) over the 4-tuples of summaries on 50 random samples from the test set and have 3 annotators pick the best and the worst contribution and context summary pairs in terms of disentanglement. The rating in Table 8 shows the percentage a model is chosen as the best minus the percentage a model is chosen as the worst; the rating ranges from -1 to 1. A similar trend to automatic disentanglement evaluation is observed here as well in that introducing the informativeness objective significantly improves MultiHead model, while it conversely affects ControlCode model.

Table 8: Best-Worst scaling scores for Disentanglement.

Model	Rating
CC	**0.027**
CC+INF	0.020
MH	-0.073
MH+INF	**0.027**

We show the annotation user interface in Fig 3.

C Generated Full Samples from Different Models

We show additional samples generated from ControlCode model in Table 9. A common failure case of all our models is the lack of disentanglement between the two summaries. While samples start generating on-topic, the model for context summary generation starts generating contributions at one point.

Table 9: Generated Sample. In this example, the red highlighted segments describe the contribution.

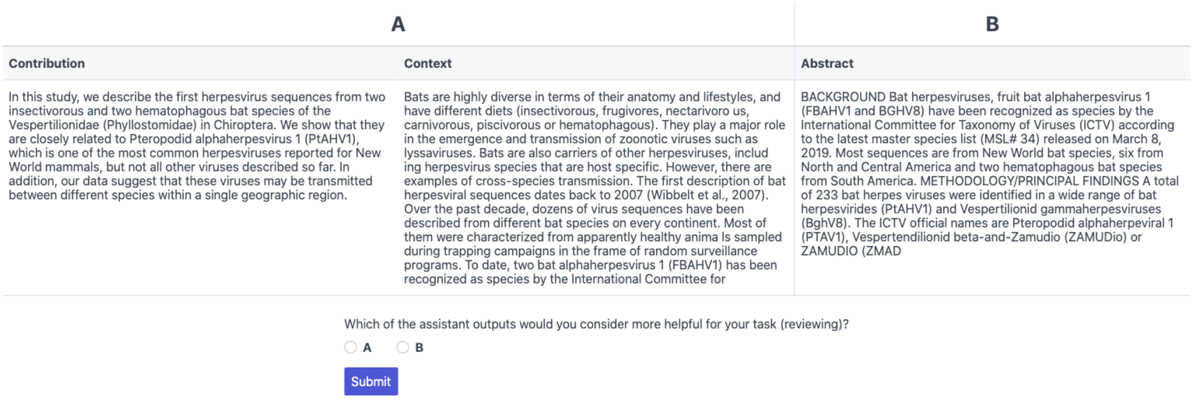
Original Abstract: Domain adaption (DA) allows machine learning methods trained on data sampled from one distribution to be applied to data sampled from another. It is thus of great practical importance to the application of such methods. Despite the fact that tensor representations are widely used in Computer Vision to capture multi-linear relationships that affect the data, most existing DA methods are applicable to vectors only. This renders them incapable of reflecting and preserving important structure in many problems. We thus propose here a learning-based method to adapt the source and target tensor representations directly, without vectorization. . . .

Generated Abstract: The difficulty of securing an appropriate and exhaustive set of training data, and the tendency for the domain of application to drift over time, often lead to variations between the distributions of the training (source) and test (target) data. In Machine Learning this problem is labeled domain mismatch. Failing to model such a distribution shift may cause significant performance degradation. Domain adaptation (DA) techniques capable of addressing this problem of distribution shift have thus received significant attention recently [24]. . . .

Contribution: To address these issues, we propose a novel approach termed Tensor-Aligned Invariant Subspace Learning (TAISL) to learn an invariant tensor subspace that is able to adapt the tensor representations directly. By introducing a set of alignment matrices, the tensors from the source domain are aligned to an underlying tensor space shared by the target domain. Instead of executing a holistic adaptation (where all feature dimensions would be taken into account), our approach performs mode-wise partial adaptation where each mode is adapted separately to avoid the curse of dimensionality. . . .
Context: Deep convolutional neural networks (CNNs) represent the state-of-the-art method for a substantial number of visual tasks. The activations of such CNNs, and the interactions between them, are naturally represented as tensors, meaning that DA should also be applied using this representation. . . . The proposed direct tensor method uses much lower dimensional entities, thus avoiding these estimation problems. To address these issues we propose to learn an invariant tensor subspace that is able to adapt the tensor representations directly. We show in Section 5 that the proposed method outperforms all comparators in DA of the Convolutional Activation of CNNs. Higher-order tensor modeling offers us an opportunity to investigate multiple interactions and couplings that capture the commonality and differences between domains. Following this idea, a novel approach

---

[9] We did not observe a substantial difference in performance between DistilBART and BART.



A		B
Contribution	**Context**	**Abstract**
In this study, we describe the first herpesvirus sequences from two insectivorous and two hematophagous bat species of the Vespertilionidae (Phyllostomidae) in Chiroptera. We show that they are closely related to Pteropodid alphaherpesvirus 1 (PtAHV1), which is one of the most common herpesviruses reported for New World mammals, but not all other viruses described so far. In addition, our data suggest that these viruses may be transmitted between different species within a single geographic region.	Bats are highly diverse in terms of their anatomy and lifestyles, and have different diets (insectivorous, frugivores, nectarivoro us, carnivorous, piscivorous or hematophagous). They play a major role in the emergence and transmission of zoonotic viruses such as lyssaviruses. Bats are also carriers of other herpesviruses, includ ing herpesvirus species that are host specific. However, there are examples of cross-species transmission. The first description of bat herpesviral sequences dates back to 2007 (Wibbelt et al., 2007). Over the past decade, dozens of virus sequences have been described from different bat species on every continent. Most of them were characterized from apparently healthy anima ls sampled during trapping campaigns in the frame of random surveillance programs. To date, two bat alphaherpesvirus 1 (FBAHV1) has been recognized as species by the International Committee for	BACKGROUND Bat herpesviruses, fruit bat alphaherpesvirus 1 (FBAHV1 and BGHV8) have been recognized as species by the International Committee for Taxonomy of Viruses (ICTV) according to the latest master species list (MSL# 34) released on March 8, 2019. Most sequences are from New World bat species, six from North and Central America and two hematophagous bat species from South America. METHODOLOGY/PRINCIPAL FINDINGS A total of 233 bat herpes viruses were identified in a wide range of bat herpesvirides (PtAHV1) and Vespertilionid gammaherpesviruses (BghV8). The ICTV official names are Pteropodid alphaherpeviral 1 (PTAV1), Vespertendilionid beta-and-Zamudio (ZAMUDio) or ZAMUDIO (ZMAD

Which of the assistant outputs would you consider more helpful for your task (reviewing)?

○ A ○ B

Figure 3: The annotation interface. Summaries indicated with A and B are disentangled summaries and a generated abstract, respectively.

D Ethical Considerations

While we achieve reasonable automatic evaluation results using the proposed models, we note that these models pose ethical risks in two ways. From readers' perspective, entirely trusting machine-generated summaries would lead to a wrong understanding of the articles, thus potentially harming the progress of the research community. Even though we show that more than 90% of sentences from the annotated summaries were truthful to the input articles, the remaining sentences that were not truthful are impactful enough to misunderstand the contributions.

From writers' perspective, our proposed models could be used maliciously to appear valuable. In a hypothetical situation where our model outputs are regarded trustworthy enough for people to assess articles, "hacking" our summarization model to output over-claming contributions could be possible.

Find Parent then Label Children: A Two-stage Taxonomy Completion Method with Pre-trained Language Model

Fei Xia^{1,2}[*]; **Yixuan Weng**^{1*}, **Shizhu He**^{1,2}, **Kang Liu**^{1,2} **and Jun Zhao**^{1,2}
^{1} National Laboratory of Pattern Recognition,, Institute of Automation, CAS
^{2} School of Artificial Intelligence, University of Chinese Academy of Sciences
`xiafei2020@ia.ac.cn,wengsyx@gmail.com`
`{shizhu.he, kliu, jzhao}@nlpr.ia.ac.cn`

Abstract

Taxonomies, which organize domain concepts into hierarchical structures, are crucial for building knowledge systems and downstream applications. As domain knowledge evolves, taxonomies need to be continuously updated to include new concepts. Previous approaches have mainly focused on adding concepts to the leaf nodes of the existing hierarchical tree, which does not fully utilize the taxonomy's knowledge and is unable to update the original taxonomy structure (usually involving non-leaf nodes). In this paper, we propose a two-stage method called ATTEMPT for taxonomy completion. Our method inserts new concepts into the correct position by finding a parent node and labeling child nodes. Specifically, by combining local nodes with prompts to generate natural sentences, we take advantage of pre-trained language models for hypernym/hyponymy recognition. Experimental results on two public datasets (including six domains) show that ATTEMPT performs best on both taxonomy completion and extension tasks, surpassing existing methods.

1 Introduction

Taxonomies[1] are an important form of domain knowledge that organize concepts into hierarchical structures, representing "hypernym-hyponym" relationships among concepts in the form of trees or directed acyclic graphs (Shen et al., 2020). Taxonomies are essential components of knowledge systems such as ontologies and knowledge graph (Yu et al., 2020), and are widely used in various downstream applications, including search engineering (Yin and Shah, 2010), recommendation systems (Huang et al., 2019; Zhang et al., 2014), and information filtering (Demeester et al., 2016).

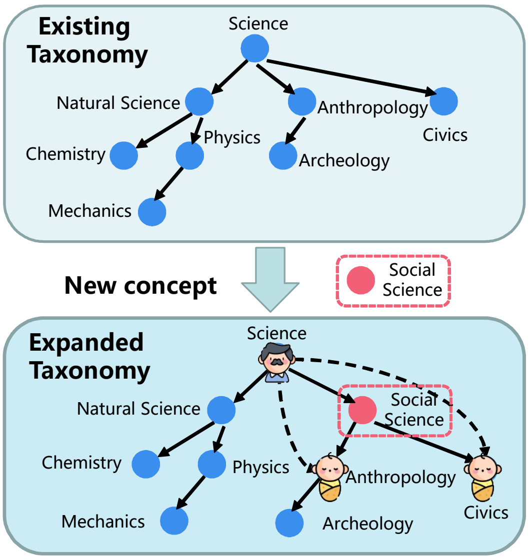


Figure 1: An example of taxonomy completion for a non-leaf node. The new concept "Social Science" needs to be inserted into the correct position in the existing taxonomy.

As domain knowledge continues to evolve, especially with the rapid growth of web content, new concepts are constantly emerging. In order to stay current, original taxonomies must incorporate these new concepts and adapt their hierarchical relationships. For example, as shown in Figure 1, with the advancement of sociology and science, the concept of "Social Science" should be added to the science knowledge system, and the original structure should be adjusted accordingly.

However, existing taxonomies are primarily constructed by human experts (Shen et al., 2018). Manual extraction of domain concepts and detection of hierarchical relationships by domain experts is both time-consuming and labor-intensive, and may result in missing important concepts and relationships.

To extend existing taxonomies automatically, researchers have proposed the tasks of taxonomy

---

[*] These authors contribute this work equally.

[1] In this paper, we mainly focus on the taxonomy represented as tree rather than directed acyclic graph, because trees are the mainstream form at present, such as the online catalog taxonomies of Amazon and Yelp.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1032–1042
May 2-6, 2023 ©2023 Association for Computational Linguistics

expansion (TE) and taxonomy completion (TC). Both tasks aim to append new nodes (concepts) to a given taxonomy . The main difference is that TE focuses on identifying the parent of a given node (usually a leaf node), while TC aims to identify both the parent and child nodes. As illustrated in Figure 1, TE would aim to identify the parent node of "Social Science", while TC would also aim to identify the child nodes of "Social Science".

Recently, researchers have been focusing on using pre-trained language models, such as BERT (Devlin et al., 2018), to improve the performance of taxonomy expansion (Liu et al., 2021; Takeoka et al., 2021). For example, TEMP (Liu et al., 2021) appends new concepts to leaf nodes and generates candidate taxonomy paths, then uses a pre-trained model for ranking and selecting the best path. Musubu (Takeoka et al., 2021) generates candidate "hypernym"-"new concept" pairs using Hearst patterns (Hearst, 1992), and relies on pre-trained knowledge to identify the optimal hypernym node for the new concept. These proposed models have greatly improved the effectiveness of taxonomy updates, thanks to the improved generalization performance of pre-trained language models (Liu et al., 2021).

Although current TE&TC methods have achieved good results, there are several main issues that need to be addressed. Firstly, existing TE methods struggle to extend non-leaf nodes or perform poorly in this task (Zhang et al., 2021). Secondly, while existing TC methods can extend both leaf and non-leaf nodes, they may be less effective in leaf node expansion than specialized TE methods (Liu et al., 2021), potentially due to a lack of sufficient utilization of knowledge. Furthermore, these methods often require large amounts of labeled samples or external resources, which are not always available (Takeoka et al., 2021). Lastly, current TC methods do not typically involve modifying the nodes of the original taxonomy system (all original parent-child relationships are preserved after adding nodes to the taxonomy). However, the insertion of new nodes can modify the relationship of the original nodes. For example, the insertion of "Social Science" in Figure 1 would change the relationship between "Science-Anthropology" from father-son to grandfather-grandson.

To address these issues, we propose **A** **T**wo-stage **T**axonomy compl**E**tion **M**ethod with **P**re-**T**rained Language model (ATTEMPT), which inserts new concepts into the correct position by identifying a parent node and labeling child nodes.

In the first stage of our proposed method, we use the "Taxonomy-path Prompt with Pre-trained model" (PPT) approach to take advantage of the local information of the taxonomy path and convert it into natural language using a prompt method, which helps to better utilize the implicit knowledge of the pre-trained model. Additionally, the pre-trained model's extensive knowledge reserve allows us to avoid the need for external resources and large amounts of labeled data. In the second stage, we propose the "Multiple Nodes Labeling" (MNL) method, which jointly identifies each child node and better utilizes the interdependence between nodes, resulting in more accurate node type prediction (including father-son, sibling and other relationships). Additionally, MNL allows for modification of the original taxonomy nodes and simultaneous annotation of multiple child nodes.

We conduct detailed experiments on two public datasets (including six domains) to evaluate the effectiveness of our proposed method, ATTEMPT, in leaf and non-leaf node expansion. Specifically, for leaf nodes, our parent-finding method (PPT) outperforms the best baseline by 8.2% in accuracy. For non-leaf nodes, our children-finding method (MNL) improves by 21% and 20.3% respectively in accuracy and average F1 score, compared to a pair-wise classification method. On the overall task, our proposed method (ATTEMPT) outperforms other methods by 2.4% in average F1 score.

In summary, the main contributions of this paper include:

- The proposal of a two-stage taxonomy expansion method, ATTEMPT, that inserts new concepts into the correct position by identifying a parent node and labeling child nodes.
- The introduction of a multiple-nodes labeling method, MNL, for the children-finding stage, which allows for the label of zero to multiple children nodes of a given node simultaneously and modification of the original taxonomy nodes.
- The demonstration of the effectiveness of our approach through experiments on two public datasets (including six domains), with the best performance obtained in both non-leaf and leaf node expansion.

2 Related Work

Taxonomy construction aims to build a tree-structured taxonomy with a set of terms from scratch. Existing methods can be roughly divided into two categories. The first is an unsupervised method to construct the taxonomy based on clustering (Alfarone and Davis, 2015; Zhang et al., 2018a; Shang et al., 2020). The terms are grouped into a hierarchy based on hierarchical clustering or topic models (Downey et al., 2015). Each node of this taxonomy is a collection of topic-indicative terms, different from the taxonomy in this paper (each node represented by one individual term). The other approach constructs a taxonomy based on terms, where each node represents a term concept (Cocos et al., 2018; Dash et al., 2020). Hypernymy detection models are often used for this task. For example, pattern-based (Agichtein and Gravano, 2000; Jiang et al., 2017; Roller et al., 2018) or distributional models (Yin and Roth, 2018; Wang et al., 2019; Dash et al., 2020) extract the hypernymy for a given query node and then organize them into a tree structure.

Creating a taxonomy from scratch is labor-intensive. In many scenarios, such as e-commerce, some taxonomies may already be deployed in online systems, which involves the demand of taxonomy extension. QASSIT (Cleuziou and Moreno, 2016) is a semi-supervised vocabulary classification method, mainly based on genetic algorithms. The TAXI (Panchenko et al., 2016) system uses a taxonomy induction method based on lexico-syntactic patterns, substrings, and focused crawling. Later, TaxoGen (Zhang et al., 2018b) uses term embeddings and hierarchical clustering to construct topic taxonomies recursively. TEMP (Liu et al., 2021) is a self-supervised classification extension method that trains models with a new dynamic margin loss margin function.

Taxonomy completion (Zhang et al., 2021) is a recently proposed task that aims to find appropriate hypernyms-hyponyms for new nodes, not just hypernyms. GenTaxo (Zeng et al., 2021) gathers information from complex local structural information and learns to generate full names of concepts from corpora. TMN (Zhang et al., 2021) focuses on channel gating mechanisms and triplet matching networks. CoRel relies on concept learning and relation transferring to build a seed-oriented topic taxonomy.

But the above mentioned methods also have some issues. The addition of new nodes may also lead to changes in the original taxonomy. The taxonomy completion task only finds the hyponyms of a given node, which cannot modify of the original taxonomy. GenTaxo (Zeng et al., 2021) requires a large amount of training data to learn enough information, and CoRel (Huang et al., 2020) focuses more on topic taxonomy than the taxonomy of individual terms. Other works such as CGExpan (Zhang et al., 2020) use the automatically generated class names and the class-guided entity selection module for entity expansion. However, CGExpan (Zhang et al., 2020) is more on the entity set than the tree taxonomy.

In addition, although the above methods can find both hypernyms and hyponyms of a given query node, they do not make sufficient use of the pre-trained model or do not use the pre-trained model at all (Zhang et al., 2021; Zeng et al., 2021). This may lead them to perform poorly on the hypernym recognition task, inferior to the specialized taxonomy extension methods of the pre-trained model (Liu et al., 2021). And most methods of taxonomy extension cannot perform well on the task of taxonomy completion (Zhang et al., 2021). We are dedicated to finding an approach that works in both tasks.

3 Method

Given an existing taxonomy $T = (V, E)$ and a set of new terms V', where V is a set of terms, and E is a set of " hyponym- hypernym" relationships between terms, the task of Taxonomy completion is to insert the new terms $v' \in V'$ into the appropriate position of the existing taxonomy T one by one and extend them into a more complete taxonomy $\tilde{T} = (\tilde{V}, \tilde{E})$.

Figure 2 provided illustrates the overall structure of the ATTEMPT method, which is broken down into two main stages: the parent finding stage and the children finding stage. These two stages work together to identify the relationships between terms in the taxonomy, specifically determining the parent and children of a given term.

3.1 Stage one: Parent Finding

The first stage of the process is to identify the parent node of a given node in the taxonomy. For example, finding the parent node “science" for the node “social science" in Figure 2.

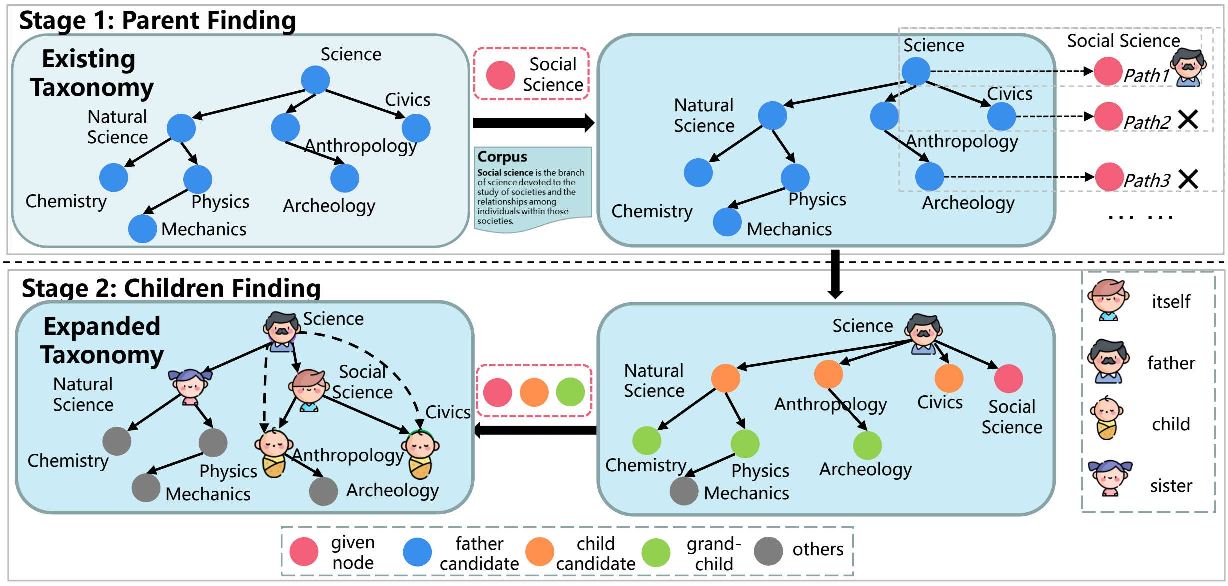


Figure 2: An overview of the proposed method ATTEMPT. ATTEMPT consists of two stages. Given an existing taxonomy and the new concept term ("Social Science"), the first stage is to find the correct parent ("Science") and the second stage is to find all possible children ("Anthropology" and "Civics").

3.1.1 TEMP

The TEMP method (Liu et al., 2021) is the first approach to use pre-trained contextual encoders as the core component for taxonomy extension. The pre-trained contextual embeddings are useful for capturing relationships between terms because they have been trained on a large corpus. TEMP predicts the location of new concepts by ranking the generated taxonomy paths. A taxonomy path of a new term (ND) in the tree-structured taxonomy is the unique path from that term to the root of the taxonomy. The taxonomy path is represented as $P = [ROOT, N_1, N_2, ..., N_D]$, where D is the depth of the ND and $ROOT$ is the root of the taxonomy. In the taxonomy, N_{i-1} is the parent of N_i. TEMP generates taxonomy paths for each term, then adds the new term to be expanded to the end of each path to form new paths. Finally, the new paths are ranked and the highest-scoring path is chosen as the correct parent term.

Equation 1 describes how TEMP uses a contextual encoder to return a sequence of vectors, given a term's definition S and an arbitrary taxonomy path P.

$$\mathrm{Encoder}(S, P) = v_{[\mathrm{CLS}]}, v_1, \ldots, v_{[\mathrm{SEP}]}, v_{p_d}, \ldots, v_{\mathrm{root}} \tag{1}$$

The TEMP method, which uses pre-trained contextual encoders to model taxonomy paths, has been an inspiration for our work. However, TEMP also has some limitations. One of the main limitations is that it can only expand new leaf nodes. Additionally, TEMP has some issues such as:

1) Limited use of local information - although TEMP uses paths to narrow the search range within the taxonomy tree, the problem of too long paths can still arise. In such cases, distant relationships may have a limited impact on the determination of leaf nodes.

2) Inadequate utilization of pre-trained model - TEMP only connects the nodes of the path using special tokens such as $[SEP]$ or $[UNK]$, which does not fully leverage the knowledge encoded by the pre-trained language model.

3.1.2 PPT: Taxonomy-path Prompt with Pre-trained model

To address the limitations of the TEMP method, we proposed PPT (A Taxonomy Expansion Method Based on Taxonomy Path Prompt and Pre-Trained Model). Our approach includes a few improvements:

Utilization of local information - Instead of using the entire taxonomy path, we use the local information nodes l_p closest to the nodes. For example, in Figure 2, for the node "Archeology", the local information nodes would be "Archeology" and "Anthropology". When the depth of the taxonomy path is less than two, we take only one node.

$$l_p = local(P) = \{N_{D-1}, N_D\} \tag{2}$$

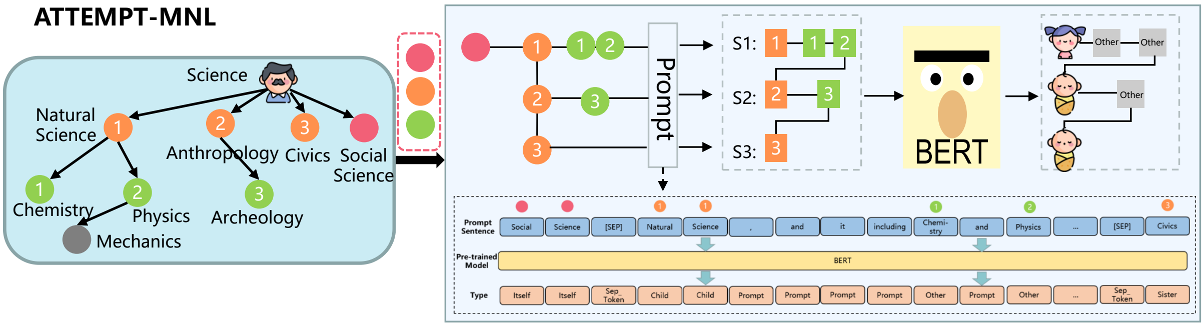


Figure 3: An overview of the MNL (multiple nodes labeling). We concatenate the current node (red), parent node, candidate children nodes (orange) and their children nodes (green) together to form a unified prompt, and then input into BERT. Finally, we realize the prediction of various types of nodes through joint labeling.

Improved pre-trained model utilization - We form a set of taxonomy path points $P_{ScocialScience}$ = (Archeology-Anthropology-Social Science) by combining the local information points of each node and the node Social Science to be extended. We then generate the appropriate natural language S_{Gen} using a prompt function.

$$S_{Gen}(q, l_p) = Prompt(q, l_p) \qquad (3)$$

where q is the node to be expanded and $Prompt$ is a function to generate natural language from prompts. For example, $S_{Gen}(q, l_p)$ = "Social Science including Anthropology, and Anthropology including Archeology". We feed this generated language S_{Gen} into the pre-trained model, rank the results in the same way as TEMP, and use the highest score as the parent node of the given node.

$$\text{Encoder}(S_{Gen}) = v'_{[\text{CLS}]}, v'_1, \ldots, v'_w \qquad (4)$$

The encoder results are as above, where w is the number of output vectors. We trained the model with Margin Ranking Loss (MRL), which is defined as follows:

$$\mathcal{L} = \sum_{P \in \mathcal{P}^+} \sum_{P' \in \mathcal{P}^-} \max\left(0, f\left(P'\right) - f(P) + \gamma\left(P, P'\right)\right) \qquad (5)$$

where $\mathcal{P}^+$ is the set of taxonomy-paths in the taxonomy, $\mathcal{P}^-$ is the set of negative samples, and $\gamma\left(P, P'\right)$ is a function designed for the margin between positive and negative taxonomy-paths. To capture the semantic similarity of different taxonomy-paths, we follow TEMP to set a dynamic margin function based on the semantic similarity as follows:

$$\gamma\left(P, P'\right) = \left(\frac{|P \cup P'|}{|P \cap P'|} - 1\right) * k \qquad (6)$$

where k is a parameter used to adjust margins (usually between 0.1 and 1).

3.2 Stage two: Children Finding

The second stage of ATTEMPT is to identify all the children nodes of a given node in the taxonomy. For example, finding the children nodes "Anthropology" and "Civics" for the node "Social Science", as shown in Figure 2. We propose two methods for this stage: PWC and MNL.

3.2.1 PWC: Pair-wise Classification

In the second stage, we identify all the child nodes of a given term. To do this, we form pairs of possible “hypernym-hyponym" term pairs from the node to be expanded (red node) and each candidate child node (orange node, child of the parent identified in the first stage). These term pairs are connected with the special token $[SEP]$ and fed into a pre-trained language model such as BERT. An example can be seen in Figure 2, where the node to be classified is “Social Science" and the orange candidate child nodes are “natural science," “anthropology" and “civics."

We use the pre-trained model to perform binary classification to determine whether the term pairs have a “hypernym-hyponym" relationship or not. The traditional cross-entropy function is used as the loss function to train the classification model. This method is simple, because the pre-trained model has been trained on a large corpus already and it can identify whether the term pairs have a hierarchical relationship or not. This method is called Pair-wise classification.

Dataset		**Environment**			**Food**			**Science**		
Metric		Wu&P	MRR	Acc	Wu&P	MRR	Acc	Wu&P	MRR	Acc
Leaf Nodes	BERT+MLP	47.9	21.5	11.1	47.0	14.9	10.5	43.6	15.7	11.5
	TaxoExpan	54.8	32.3	11.1	54.2	40.5	27.6	57.6	44.8	27.8
	STEAM	69.6	46.9	36.1	67.0	43.4	34.2	68.2	48.3	36.5
	TMN	54.0	43.6	35.0	65.9	47.2	34.7	75.9	53.2	41.9
	TEMP-BERT	75.9	62.0	49.0	78.3	57.1	45.2	84.6	64.6	54.4
	ATTEMPT-PPT(ours)	**82.3**	**75.1**	**65.4**	**78.4**	**58.1**	**46.5**	**86.6**	**70.7**	**61.2**
All Nodes	TEMP-BERT	64.8	33.5	15.6	84.3	41.1	37.4	88.0	52.9	43.4
	ATTEMPT-PPT(ours)	**81.1**	**52.5**	**37.5**	84.3	**42.6**	**39.0**	**90.9**	**60.0**	**52.8**

Table 1: Baseline comparison on the three datasets in stage one (in %)

3.2.2 MNL: Multiple Nodes Labeling

MNL is a new approach that addresses the problem of identifying multiple children of a given node in the taxonomy. There are two main challenges: determining whether a node has children and how many children it has, and identifying as many children as possible if there are multiple children.

To address these challenges, we first determine whether the given node is a leaf node (has no children) and if so, the second stage ends. If there are multiple children, we treat this as a multiple-choice problem and model it as a sequential labeling task. As shown in Figure 3, we extract the possible siblings, children, and grandchildren (orange and green nodes) of the given node to make use of local information. We then use a prompt function to convert these three types of nodes into natural language (e.g., “Natural Science - Chemistry, Physics" is converted to “Natural Science, and it including Chemistry and Physics").

We concatenate the node to be expanded “Social Science" with all the sentences generated by the prompt, and then feed this into the pre-trained model. Since the model was trained on a large corpus of natural language, the input of natural language is consistent with the pre-training phase, which helps to fully utilize the hidden information of the model and correctly identify the contextual relationships. The addition of local information provides additional context to the model, which allows it to make more accurate predictions about the children of the given term.

4 Experiments

In this section, we first describe the experimental setup and implementation details in Section 4.1 and Section 4.2. We then present the results of our experiments in Section 4.3, including a comparison of our approach to the baseline method. To further understand the contribution of different components of our approach, we conduct ablation experiments in Section 4.4 to investigate the effectiveness of using local information and prompts in ATTEMPT.

4.1 Experimental Setup

Datasets. We conducted experiments on two datasets that include six domains and two types of nodes. The first dataset is the Semeval-2016 task 13 dataset, which was used to evaluate the performance of expanding leaf nodes in stage one. We compared our method to previous approaches such as TEMP (Liu et al., 2021) and STEAM (Yu et al., 2020), which have also been tested on this dataset for leaf node expansion.

To evaluate the expansion of non-leaf nodes, we constructed a new dataset based on Semeval, as there are limited previous datasets that are relevant to this task. This dataset was specifically designed for the purpose of non-leaf node expansion and evaluation.

The following is a description of the two datasets: 1) We used the dataset from Semeval-2016 task 13 [2], which contains three English datasets for the environment, science, and food domains. We followed the setup as in (Yu et al., 2020) and used the randomly-grown taxonomies for self-supervised learning, and sampled 20% of the leaf nodes for testing. We used this dataset to compare our method with other taxonomy extension methods for leaf nodes. 2) As there is limited data available for non-leaf node expansion, we reconstructed the original data. We defined nodes with one parent and no children as leaf nodes and

---

[2] https://alt.qcri.org/semeval2016/task13/

Dataset		Environment		Science		Food	
Metric		acc	Avg(F1)	acc	Avg(F1)	acc	Avg(F1)
Non-Leaf Nodes	PWC	21.5	50.8	20.0	54.3	12.5	32.1
	ATTEMPT-MNL(ours)	**46.2**	**68.7**	**33.3**	**59.8**	**37.5**	**69.6**
All Nodes	PWC	47.1	56.3	58.1	**74.65**	60.3	74.8
	ATTEMPT-MNL(ours)	**64.7**	**82.8**	**61.3**	74.09	**73.3**	**84.5**

Table 2: Baseline comparison on the three datasets in stage two (in %)

Dataset	Environment		Science		Food	
Metric	acc	Avg(F1)	acc	Avg(F1)	acc	Avg(F1)
Baseline	11.8	14.7	22.5	23.3	25.9	30.9
ATTEMPT(ours)	**12.0**	**15.0**	**22.6**	**28.1**	**26.7**	**32.9**

Table 3: Comparison of the baseline method and ATTEMPT in the overall process(in %)

nodes with one parent and at least one child as non-leaf nodes. More details about the dataset are provided in Appendix A.

Metrics. For the parent finding process in stage 1, we followed the evaluation strategy of (Yu et al., 2020) using Accuracy, Mean reciprocal rank (MRR), and Wu & Palmer similarity (Wu&P) to evaluate our methods. Accuracy (ACC) measures the count of parent or child nodes that are accurately predicted. MRR calculates the average of reciprocal ranks of the true taxonomy path. Wu&P measures the semantic similarity between the predicted taxonomy path and the truth taxonomy-path.

For stage two, we proposed two metrics for evaluating the effectiveness of this phase. One is ACC, which represents whether all children can be found or not. The second one is Avg F1, which can further evaluate how many children are found for a given node. $\text{Avg(F1)} = \frac{1}{n}\sum_{i=1}^{n} \text{F1}$

Compared Methods. We compare with the following methods:

- **BERT+MLP** The method extracts terms embeddings from BERT and then feeds them into a multilayer perceptron (MLP) to predict their relationship.
- **TEMP** (Liu et al., 2021) One state-of-the-art taxonomy expansion framework which predicts new concepts' position by ranking the generated taxonomy paths. The first method that employs pre-trained contextual encoders in taxonomy construction and hypernym detection problems.
- **STEAM** (Yu et al., 2020) A taxonomy expansion framework that leverages natural supervision in the existing taxonomy for expansion.
- **TaxoExpan** (Shen et al., 2020) A self-supervised method for encoding local structures in seed taxonomy using location-enhanced graph neural networks.
- **TMN** (Zhang et al., 2021) A Triplet Matching Network (TMN) that finds suitable hypernym, hyponym word pairs for a given query concept.

4.2 Implementation Details

We present the PPT method for the first stage of leaf node expansion, which is based on TEMP (TEMPs' code link [3]). We use BERT (bert-base-uncased) as the pre-trained language model and split the terms into 10% for validation and 10% for testing. To expand the full type of nodes, both leaf and non-leaf, we use the new data introduced previously and select the same number of leaf and non-leaf nodes as the test set. We use the default optimal hyperparameters of the original TEMP authors and experiment with different learning rates to obtain the best performance. We also use multiple prompts (see Appendix C) according to the settings of Musubu (Takeoka et al., 2021), and take the average result as the experimental result. To reduce the impact of randomness, we repeat the experiment three times.

For the MNL method in stage two, we connect the nodes to be expanded (red), the candidate child nodes (orange), and the child nodes of the candidate nodes (green) and generate natural language by way of prompt. The generated natural language is fed into the pre-training model and labelled. We label the real children of a given node as 1, the sibling nodes as 0, and ignore the computational loss for all the rest of the nodes. In addition, if a term has multiple tokens and one of the tokens is marked as one by the model, we mark all those tokens as child nodes. See Appendix B for more details.

---

[3] https://github.com/liu-zichen/TEMP

Dataset	**Environment**			**Science**			**Food**		
Metric	Wu&P	MRR	Acc	Wu&P	MRR	Acc	Wu&P	MRR	Acc
no path nodes	81.8	73.2	59.6	81.0	67.2	56.5	78.5	57.5	44.1
no prompt	79.0	64.8	51.9	84.1	67.0	56.5	78.4	56.9	45.1
ATTEMPT-PPT(ours)	**82.3**	**75.1**	**65.4**	**86.6**	**70.7**	**61.2**	**78.7**	**58.1**	**46.5**

Table 4: Results of ablation experiments on the three dataset in stage one (in %)

Dataset	**Environment**		**Science**		**Food**	
Metric	acc	Avg(F1)	acc	Avg(F1)	acc	Avg(F1)
no prompt	52.9	53.7	**62.0**	**75.0**	52.7	63.4
no grandchild	52.9	52.9	51.6	62.3	51.1	59.8
ATTEMPT-MNL(ours)	**64.7**	**82.8**	61.3	74.2	**73.3**	**84.5**

Table 5: Results of ablation experiments on the three dataset in stage two (in %)

4.3 Experimental Results

As shown in Table 1, our method PPT outperforms the existing TEMP model significantly on both leaf and non-leaf nodes. For leaf nodes, we improved the TEMP model by 8.2%, 6.7%, and 2.8% on Acc, MRR, and Wu&P, respectively. For all types of nodes, the improvement is 11.0%, 9.2%, and 6.4%, respectively.

The comparison results of the two methods tested in the child discovery phase are presented in Table 2. For leaf nodes, the MNL method improves Acc and Avg(F1) by 21% and 20.3%, respectively, compared to the pair-wise classification method over the three benchmark datasets. For all type nodes, the improvement is 11.3% and 11.9%, respectively.

Table 3 presents the comparison results between the baseline method and our ATTEMPT. The baseline method achieves 14.7%, 23.3%, and 30.9% in Avg(F1) metrics for the three datasets of environment, science, and food, respectively. Our ATTEMPT method improved the Avg(F1) by an average of 2.4% over the baseline. The low results in Table 3 are due to the challenging nature of the task. To obtain the correct parent node, all child nodes must be successfully identified. This highlights the potential for further improvement.

4.4 Ablation Studies

To verify local information and prompt effectiveness, we compare and test the changes in experimental results with/without these two types of information on both stages.

Local Information As shown in Table 4, after removing the path nodes, the PPT method in stage 1 decreases on average by 4.3%, 2%, and 2.1% on acc, mrr,wu&p, respectively, on the three datasets. Table 5 also shows that the MNL method decreases by 14.6% and 22.1% on average on accuracy and average F1 score, respectively, after removing the grandchild node information in the child finding stage. We found that local information is essential in both the first and second phases, particularly in the second child lookup phase. Removing local information brings about a significant performance degradation, which may be attributed to our method's modelling of relationships. The individual nodes are closely associated in our MNL method.

Prompt In Table 4, the PPT method with prompt removal decreased in acc, mrr, wu&p by 6.5%, 5.1%, and 2.0% on average, respectively. Meanwhile, in the second stage, the MNL method decreased 8.6% and 15.6% for accuracy and average F1 metrics, respectively, after prompt removal. The scientific data in the second stage showed a slight performance improvement after prompt removal, which we speculate may be due to insufficient data and pre-trained corpus. Overall, the prompt is essential for the parent finding process in the first stage and the child finding process in the second stage.

5 Conclusion

This paper proposes a two-stage taxonomy completion method based on pre-trained Language models (ATTEMPT), which effectively inserts the new concept in the correct position by finding a parent node then labeling children nodes. In addition, we use prompt to generate natural language information suitable for the pre-trained model further to improve the effectiveness of parent node recognition and children labeling for the given node. Our experiments on two types and three domains with six datasets show that our method can enhance the effectiveness of locating the position of a given node in existing taxonomies. Furthermore, the efficacy of local information and prompts in ATTEMPT is also demonstrated by ablation experiments. In conclusion, our proposed ATTEMPT method is an effective approach for taxonomy completion, and it can be further improved with more comprehensive datasets.

Limitations

Since ATTEMPT uses the pre-trained language model to complete the taxonomy, the expansion effect is limited by the model. Generally, pre-trained models with more knowledge scales are better (e.g., BERT-Large V.S. BERT-Base-uncased). However, our paper focuses on how to fully use the knowledge of the pre-trained model rather than verifying whether more knowledge scales better or not. Based on the above, this paper does not conduct more related research (in fact, TEMP (Liu et al., 2021) has been compared and reached similar conclusions). In addition, the selection of prompts will also affect the expansion effect. For the convenience of comparison, we have selected several basic prompts (the same as Musubu (Takeoka et al., 2021)) for experimentation. In future work, we plan to study how to construct or select better prompts for classification expansion. We do not consider the situation of multi-parent nodes according to the TEMP (Liu et al., 2021) settings. And according to our statistics, there are only a few multi-parent nodes in the Semeval-2016(task-13) datasets (1/3843). We will continue investigating how to make better use of the pre-trained model knowledge to solve the taxonomy completion problem.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No. 2020AAA0106400), the National Natural Science Foundation of China (No.61976211) and the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDA27020200). This research work was supported by the Youth Innovation Promotion Association CAS, Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004) and CCF-DiDi GAIA Collaborative Research Funds for Young Scholars.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 85–94.

Daniele Alfarone and Jesse Davis. 2015. Unsupervised learning of an is-a taxonomy from a limited domain-specific corpus. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Guillaume Cleuziou and Jose G. Moreno. 2016. QASSIT at SemEval-2016 task 13: On the integration of semantic vectors in pretopological spaces for lexical taxonomy acquisition. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1315–1319, San Diego, California. Association for Computational Linguistics.

Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2018. Comparing constraints for taxonomic organization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 323–333.

Sarthak Dash, Md Faisal Mahbub Chowdhury, Alfio Gliozzo, Nandana Mihindukulasooriya, and Nicolas Rodolfo Fauceglia. 2020. Hypernym detection using strict partial order networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7626–7633.

Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted rule injection for relation embeddings. *arXiv preprint arXiv:1606.08359*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Doug Downey, Chandra Bhagavatula, and Yi Yang. 2015. Efficient methods for inferring large sparse topic hierarchies. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 774–784.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *International Conference on Computational Linguistics*.

Jiaxin Huang, Yiqing Xie, Yu Meng, Yunyi Zhang, and Jiawei Han. 2020. Corel: Seed-guided topical taxonomy construction by concept learning and relation transferring. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1928–1936.

Jin Huang, Zhaochun Ren, Wayne Xin Zhao, Gaole He, Ji-Rong Wen, and Dong Daxiang. 2019. Taxonomy-aware multi-hop reasoning networks for sequential recommendation. In *Web Search and Data Mining*.

Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. Metapad: Meta pattern discovery from massive text corpora. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 877–886.

Zichen Liu, Hongyuan Xu, Yanlong Wen, Ning Jiang, HaiYing Wu, and Xiaojie Yuan. 2021. TEMP: Taxonomy expansion with dynamic margin loss through taxonomy-paths. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3854–3863, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Alexander Panchenko, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cédrick Fairon, Simone Paolo Ponzetto, and Chris Biemann. 2016. TAXI at SemEval-2016 task 13: a taxonomy induction method based on lexico-syntactic patterns, substrings and focused crawling. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1320–1327, San Diego, California. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst patterns revisited: Automatic hypernym detection from large text corpora. *arXiv preprint arXiv:1806.03191*.

Jingbo Shang, Xinyang Zhang, Liyuan Liu, Sha Li, and Jiawei Han. 2020. Nettaxo: Automated topic taxonomy construction from text-rich network. In *Proceedings of The Web Conference 2020*, pages 1908–1919.

Jiaming Shen, Zhihong Shen, Chenyan Xiong, Chi Wang, Kuansan Wang, and Jiawei Han. 2020. *TaxoExpan: Self-Supervised Taxonomy Expansion with Position-Enhanced Graph Neural Network*, page 486–497. Association for Computing Machinery, New York, NY, USA.

Jiaming Shen, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle Vanni, Brian M. Sadler, and Jiawei Han. 2018. Hiexpan: Task-guided taxonomy construction by hierarchical tree expansion. In *Knowledge Discovery and Data Mining*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Kunihiro Takeoka, Kosuke Akimoto, and Masafumi Oyamada. 2021. Low-resource taxonomy enrichment with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2747–2758, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chengyu Wang, Yan Fan, Xiaofeng He, and Aoying Zhou. 2019. A family of fuzzy orthogonal projection models for monolingual and cross-lingual hypernymy prediction. In *The World Wide Web Conference*, pages 1965–1976.

Wenpeng Yin and Dan Roth. 2018. Term definitions help hypernymy detection. *arXiv preprint arXiv:1806.04532*.

Xiaoxin Yin and Sarthak Shah. 2010. Building taxonomy of web search intents for name entity queries. In *Proceedings of the 19th international conference on World wide web*, pages 1001–1010.

Yue Yu, Yinghao Li, Jiaming Shen, Hao Feng, Jimeng Sun, and Chao Zhang. 2020. *STEAM: Self-Supervised Taxonomy Expansion with Mini-Paths*, page 1026–1035. Association for Computing Machinery, New York, NY, USA.

Qingkai Zeng, Jinfeng Lin, Wenhao Yu, Jane Cleland-Huang, and Meng Jiang. 2021. Enhancing taxonomy completion with concept generation via fusing relational representations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2104–2113.

Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian Sadler, Michelle Vanni, and Jiawei Han. 2018a. Taxogen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2701–2709.

Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian M. Sadler, Michelle Vanni, and Jiawei Han. 2018b. Taxogen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering. In *Knowledge Discovery and Data Mining*.

Jieyu Zhang, Xiangchen Song, Ying Zeng, Jiaze Chen, Jiaming Shen, Yuning Mao, and Lei Li. 2021. Taxonomy completion via triplet matching network. In *AAAI*.

Yuchen Zhang, Amr Ahmed, Vanja Josifovski, and Alexander Smola. 2014. Taxonomy discovery for personalized recommendation. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 243–252.

Yunyi Zhang, Jiaming Shen, Jingbo Shang, and Jiawei Han. 2020. Empower entity set expansion via language model probing. *arXiv preprint arXiv:2004.13897*.

A Dataset

The original dataset and our reconstructed dataset statistics are in Table 6 and 7.

Dataset	Environment	Science	food
$\|\mathcal{N}\|$	261	429	1486
$\|\mathcal{L}\|$	200	306	1161
$\|\mathcal{D}\|$	6	8	8
$\|\Delta_{leaf}\|$	3.79	5.17	5.37
$\|\Delta_{non}\|$	2.97	4.41	4.78
train(leaf/non)	148/59	226/112	893/284
test(leaf/non)	52/0	85/0	297/0

Table 6: Statistics of the original taxonomy datasets for evaluation. $|\mathcal{N}|$ and $|\mathcal{L}|$ are the number of nodes and leaf nodes in the taxonomy. $|\mathcal{D}|$, $|\Delta_{leaf}|$ and $|\Delta_{non}|$ indicate the depth of the taxonomy and the average depth of leaf nodes and non-leaf nodes respectively.Train(leaf/non) and test(leaf/non) represents the proportion of leaf and non-leaf nodes in train data and test data.

Dataset	Environment	Science	food
$\|\mathcal{N}\|$	261	429	1486
$\|\mathcal{L}\|$	200	306	1161
$\|\mathcal{D}\|$	6	8	8
$\|\Delta_{leaf}\|$	3.79	5.17	5.37
$\|\Delta_{non}\|$	2.96	4.41	4.05
train(leaf/non)	187/46	292/94	1124/214
test(leaf/non)	13/13	21/21	74/74

Table 7: Statistics of the new taxonomy datasets for evaluation.

To prevent the test data from being leaked during training and to thoroughly test the generalization ability of the model when encountering unseen data, we split each original taxonomy tree into two subtrees, one for training and one for testing. For example, the left subtree of the scientific taxonomy in Figure 2, natural science and its children, is used as the test subtree, and the rest is used for training. Specifically, we select the subtree with 20% of the number of nodes of the current taxonomy tree as the subtree for testing and ignore too many leaf nodes to ensure the ratio of leaf nodes to non-leaf nodes is 1:1. Too many leaf nodes will make the child finding stage degenerate into an expansion of leaf nodes, and the model will be easily overfitting. And too few leaf nodes will make the test inadequate, so we use equal leaf and non-leaf node data as the test.

In the training and testing phases, we dig out the node to be expanded in the current taxonomy tree, and if the node has N children, these N children are reassigned to the original parent of the node to be expanded as child nodes. We ignore the case of double parent nodes because their existence is too rare. Only one node in the three datasets containing more than 2000 nodes in our experiments has a dual-parent node. We will consider this case further in our future work.

B Implementation Details

For the fairness of the experiment, we follow the setting of TEMP (Liu et al., 2021). We use 10% terms for validating and 10% for testing. For each benchmark, we try various learning rates and report the best performance. We use multiple prompts to experiment and select the average result as the experimental result. We repeated the experiment three times to reduce the impact of randomness. We train the model using the Pytorch [4] (Paszke et al., 2019) on the NVIDIA RTX3090 GPU. For all methods, the bert-base-uncased [5] model are chosen for feature extraction. The pretrained contextual encoders are of base size with 12 layers. We use the AdamW (Loshchilov and Hutter, 2018) as the optimizer with the warm-up (He et al., 2016), and fine tune the whole model with a learning rate of 2e-5. The dropout (Srivastava et al., 2014) of 0.1 is applied to prevent overfitting.

C Prompt Details

Name	Prompt
Such-as	Y such as X and Z
One-of	X is one of Y, and Z is one of Y
Especiaally	Y, especially X and Z
Is-a	X is a Y, and Z is a Y
Including	Y including X and Z

Table 8: List of prompts used in the experiments. Y denotes a parent term of a term X and Z

---

[4] `https://pytorch.org`

[5] `https://huggingface.co/bert-base-uncased`

Meta Self-Refinement for Robust Learning with Weak Supervision

Dawei Zhu[1], **Xiaoyu Shen**[1], **Michael A. Hedderich**[1,2], **Dietrich Klakow**[1]
[1] *Saarland University, Saarland Informatics Campus, Germany*
[2] *Cornell University, United States*
`{dzhu,xshen,mhedderich,dietrich.klakow}@lsv.uni-saarland.de`

Abstract

Training deep neural networks (DNNs) under weak supervision has attracted increasing research attention as it can significantly reduce the annotation cost. However, labels from weak supervision can be noisy, and the high capacity of DNNs enables them to easily overfit the label noise, resulting in poor generalization. Recent methods leverage self-training to build noise-resistant models, in which a teacher trained under weak supervision is used to provide highly confident labels for teaching the students. Nevertheless, the teacher derived from such frameworks may have fitted a substantial amount of noise and therefore produce incorrect pseudo-labels with high confidence, leading to severe error propagation. In this work, we propose Meta Self-Refinement (MSR), a noise-resistant learning framework, to effectively combat label noise from weak supervision. Instead of relying on a fixed teacher trained with noisy labels, we encourage the teacher to refine its pseudo-labels. At each training step, MSR performs a meta gradient descent on the current mini-batch to maximize the student performance on a clean validation set. Extensive experimentation on eight NLP benchmarks demonstrates that MSR is robust against label noise in all settings and outperforms state-of-the-art methods by up to 11.4% in accuracy and 9.26% in F1 score.

1 Introduction

Fine-tuning pre-trained language models has led to great success across NLP tasks. Nonetheless, it still requires a substantial amount of manual labels to achieve satisfying performance on many tasks. In reality, obtaining large amounts of high-quality labels is costly and labor-intensive (Davis et al., 2013). For certain domains, it is even infeasible due to legal issues and lack of data or domain experts. Weak supervision is a widely-used approach for reudcing such cost by leveraging labels from weak sources, e,g., heuristic rules, knowledge bases or lower-quality inexpensive crowdsourcing (Ratner et al., 2017; Zhou et al., 2020; Lison et al., 2020). It has raised increasing attention in recent years, and efforts have been made to quantify the progress on weakly supervised learning, like the WRENCH benchmark (Zhang et al., 2021).

Although weak labels are inexpensive to obtain, they are often noisy and inherit biases from weak sources. Training neural networks with weak labels is challenging because of their immense capacity, which leads them to heavily overfit to the noise distribution, resulting in inferior generalization performance (Zhang et al., 2017). Various approaches have been proposed to tackle this challenge. Earlier research focused primarily on simulated noise (Bekker and Goldberger, 2016; Hendrycks et al., 2018), required prior knowledge (Ren et al., 2020; Awasthi et al., 2020) or relied on context-free aggregation rules without leveraging modern pre-trained language models (Ratner et al., 2017; Fu et al., 2020).

Recently, Yu et al. (2021) proposed a contrastive regularized self-training framework that achieved state-of-the-art (SOTA) performance in several NLP tasks from the WRENCH benchmark. It trains a teacher network on weak labels, then iteratively applies the teacher to produce pseudo-labels for training a new student model. To prevent error propagation, it filters the pseudo-labels with the model confidence scores and adds contrastive feature regularization to enforce more distinguishable representations. However, we find that this approach is *effective on easy tasks but fragile on challenging ones*, where the initial teacher model already have memorized a substantial amount of biases with high confidence. Consequently, confidence-based filtering is misleading and all future students will be reinforced with these initial wrong biases from the teacher.

To address this weakness, one strategy is learning to reweight the pseudo-labels with meta learning (Ren et al., 2018; Shu et al., 2019; Wang et al.,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1043–1058
May 2-6, 2023

2020). By this means, sample weights are dynamically adjusted to minimize the validation loss instead of prefixed with potentially misleading confidence scores. Nevertheless, if the initial teacher is weak and mostly produces incorrect pseudo-labels, simply reweighting the labels does not suffice to extract enough useful training signals.

In this paper, we propose Meta Self-Refinement (MSR) to go one step further. The teacher is jointly trained with a meta objective such that the student, after one gradient step, can achieve better performance on the validation set. In each training step, a copy of the current student performs one step of gradient descent based on the teacher predictions. The teacher will then update itself towards the gradient direction that minimizes the validation loss of the student. Finally, the actual student is trained by the updated teacher. In MSR, teacher's predictions are iteratively *refined*, instead of only "reweighted", based on the meta objective. This will enable more efficient data utilization since the teacher still has the opportunity to refine itself to provide the proper training signal, even if its initial output label is wrong. To further stabilize the training, we enhance our framework with confidence filtering when teaching the student and apply a linearly scaled learning rate scheduler to the teacher.

In summary, the main contributions are as follows: **1)** We propose a meta-learning based self-refinement framework, MSR, that allows robust learning with label noise induced by weak supervision. **2)** We analyze and quantify how label noise impacts model predictions and representation learning. We find existing methods become less effective in challenging cases when the label noise can be easily fitted. In contrast, MSR is more stable and learns better representation. **3)** Extensive experiments demonstrate that MSR consistently reduces the negative impact of the label noise, matching or outperforming SOTAs on six sequence classification and two sequence labeling tasks.[1]

2 Related Work

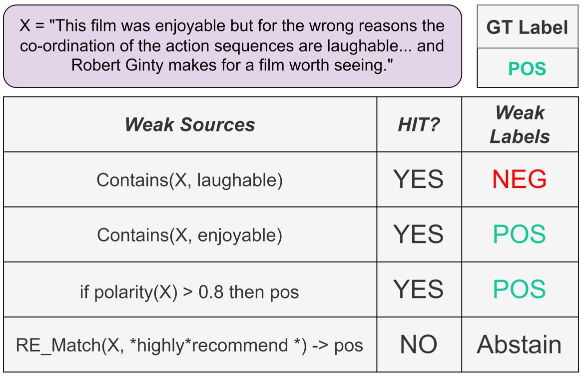
Learning with Noisy Labels. Learning in the presence of label noise is a long-standing problem (Angluin and Laird, 1988). Zhang et al. (2017) show that deep neural networks can memorize arbitrary noise during training, resulting in poor generalization. Noise-handling techniques - by modeling (Goldberger and Ben-Reuven, 2017; Patrini et al., 2017; Hendrycks et al., 2018) or filtering (Han et al., 2018; Li et al., 2020) the noisy instances - are proposed to conquer the label noise. While being effective, they typically assume that the noise is feature-independent which may oversimplify the noise generation process in realistic settings (Gu et al., 2021; Zhu et al., 2022). Recently, realistic and feature-dependent noise induced by weak supervision has received significant attention. To handle this type of noise, Awasthi et al. (2020) propose an implication loss that jointly denoises the noisy labels and weak sources. Ren et al. (2020) denoise the weak label by considering the reliability of different weak sources and aggregating them into one cleaned label. Zhang et al. (2021) release a benchmark, WRENCH, including various weakly supervised datasets in both text and image domains.

Self-Training. Self-training (Yarowsky, 1995; Lee et al., 2013) is a simple yet effective framework that is commonly used in semi-supervised learning (SSL). It typically trains a teacher model to provide pseudo-labels for the student model. Different methods have been proposed for better generalization (Xie et al., 2020; Zoph et al., 2020; Mukherjee and Hassan Awadallah, 2020). Recently, self-training has been adopted to tackle weak supervision. Karamanolakis et al. (2021) train a teacher network that aggregates weak labels to form high-quality pseudo-labels for the student. Liang et al. (2020); Yu et al. (2021) initialize the teacher model by training a classifier directly on the weak labels, they apply early stopping to prevent this initial teacher from memorizing the label noise. The student is then trained on the highly confident pseudo-labels provided by the teacher. While the core assumption of self-training - that highly confident pseudo-labels are reliable - is generally valid in SSL, it may not be true for feature-dependent noise induced by weak supervision, especially when the noise is easy to learn. In this case, self-training inevitably suffers more from error propagation and fails to train robust models.

Meta-Learning. Recently, different works leveraged meta-learning techniques to develop noise-robust learning frameworks. The idea is to optimize an outer learner (e.g., sample weights) that guides the inner learner (the classifier) to generalize well. Often, a clean validation dataset is used as a proxy

---

[1] Code is available on: `https://github.com/uds-lsv/msr`



X = "This film was enjoyable but for the wrong reasons the co-ordination of the action sequences are laughable... and Robert Ginty makes for a film worth seeing."

GT Label
POS

Weak Sources	*HIT?*	*Weak Labels*
Contains(X, laughable)	YES	NEG
Contains(X, enjoyable)	YES	POS
if polarity(X) > 0.8 then pos	YES	POS
RE_Match(X, *highly*recommend *) -> pos	NO	Abstain

Figure 1: Sentiment analysis dataset annotated with rule-based weak sources. A weak source is triggered if a specific textual pattern is matched, after which a pre-defined label is then assigned. Otherwise, it abstains. Depending on how many weak sources are triggered, a text may obtain zero, one, or multiple weak labels.

for estimating the generalization performance. Ren et al. (2018) attempt to down weight training samples that increase the validation loss. Shu et al. (2019) employ a neural network to infer such sample weights and show a significant boost on performance under feature-independent noise. Wang et al. (2020) reweight the training samples by their pseudo-labels instead of the original noisy labels. In this work, we aim to leverage meta-learning in a more flexible manner by refining the pseudo-labels instead of reweighting them. Approach-wise, the most related works are (Pham et al., 2021; Zhou et al., 2022) used for semi-supervised learning and model distillation, which also refine the teacher's parameters based on the student feedback. However, they work with samples from clean distributions, while we anticipate the noise memorization effect and enhance our framework with teacher warm-up and confidence filtering to suppress the error propagation.

3 Problem Formulation

Let $\mathcal{X}$ and $\mathcal{Y}$ be the feature and label space, respectively. In standard supervised learning, one is given a clean dataset $\mathcal{D}_c = \{(x_i, y_i)\}_{i=1}^N$, where N is the number of samples. The clean labels y_i are supposed to be annotated by human experts.

In weak supervision, a dataset is labeled by weak sources rather than humans. Weak sources can have diverse forms like lexical rules, knowledge bases, pre-trained models, lower-quality inexpensive crowdsourcing, etc. Figure 1 shows an example of text labeled via weak supervision. Compared to manual annotations, weak labels contain more mistakes. We denote the dataset labeled by weak sources by $\mathcal{D}_w = \{(x_i, \hat{y}_i)\}_{i=1}^N$ where $\hat{y}_i$ is the weak label.[2] Since weak sources might not cover all data, we may have a set of unlabeled data $\mathcal{D}_u$ in addition to $\mathcal{D}_w$. We use $\mathcal{D}_a = \mathcal{D}_w \cup \mathcal{D}_u$ to denote the full set of data. Moreover, as we do not make any assumption on the quality of the weak labels, their distribution can deviate arbitrarily from the distribution of clean labels. Learning with only weak labels can lead to unbounded model errors (Menon et al., 2016; Gu et al., 2021). Hence, following standard practice in weak supervision, we assume the access to a small clean validation set $\mathcal{D}_v = \{(x_i^v, y_i^v)\}_{i=1}^M$ where $M \ll N$. $\mathcal{D}_v$ is used for early stopping, hyper-parameter tuning or meta-learning so that the learned model will not fully overfit the noisy weak labels (Ren et al., 2018; Shu et al., 2019; Zhang et al., 2021).

4 Meta Self-Refinement

We propose a novel meta-learning based framework, named Meta Self-Refinement (MSR), to tackle the label noise induced by weak supervision. In contrast to conventional self-training methods, where the teacher model is fixed after being trained on weakly labeled data, MSR enables the teacher to refine itself based on student performance on the clean validation set, yielding higher-quality labels and more accurate confidence estimates. In this section, we first provide an overview of its training objective (section 4.1), then go into the training details (section 4.2). Figure 2 illustrates the full training process.

4.1 Training Objective

MSR contains a teacher network f and a student network g, both are functions that map $\mathcal{X} \rightarrow \mathcal{Y}$. f is initialized by fine-tuning a pre-trained language model (PLM) on the weakly labeled data $\mathcal{D}_w$:

$$f_1 = \arg\min_f \mathbb{E}_{(x_i, \hat{y}_i) \in \mathcal{D}_w} \mathcal{L}(\hat{y}_i, f(x_i)) \qquad (1)$$

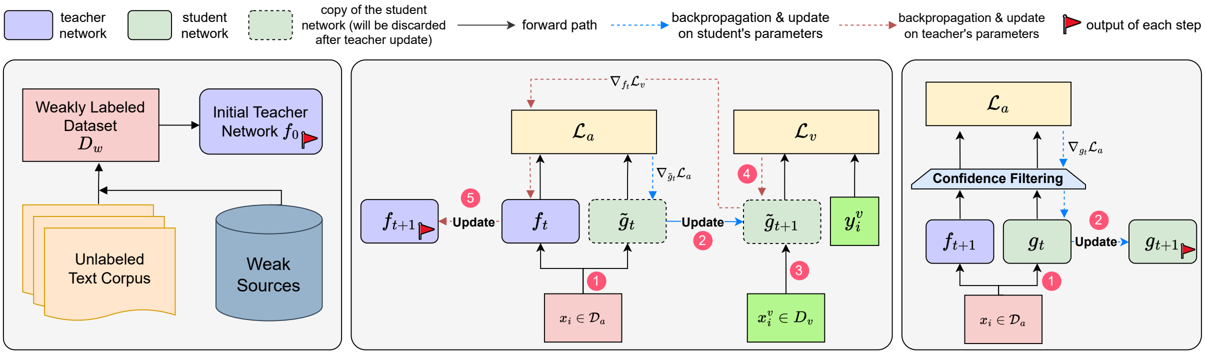
where $\mathcal{L}$ denotes the loss function. We use the cross entropy loss throughout the paper:

$$\mathcal{L}(p, q) = -\mathbb{E}_{y \sim p(y)} \log q(y) \qquad (2)$$

p and q are distributions over the label space $\mathcal{Y}$. The initial student network, g_1, is the PLM without fine-tuning on any data.

---

[2] Multiple weak sources may be triggered simultaneously by a sample. In this case, we can use different aggregation methods like majority voting to determine the final weak label.



(a) Train the initial teacher with weak labels (b) Teacher update (c) Student update

Figure 2: Illustration of our proposed Meta-Self Refinement method (MSR). (a) We start by fine-tuning a PLM on weak labels with early stopping, which yields an initial teacher f_1. (b) At each training step t, f_t gets training signals by performing a "teaching experiment" on $\tilde{g}_t$: a copy of the student network g_t. $\tilde{g}_t$ is updated by fitting f_t with the loss function $\mathcal{L}_a$. f_t is then updated to minimize the validation loss $\mathcal{L}_v$ of $\tilde{g}_{t+1}$. (c): g_t is updated by fitting f_{t+1} with confidence filtering under the loss $\mathcal{L}_a$.

In conventional self-training, f_1 is used to provide pseudo-labels to train the student. By selecting higher-quality pseudo-labels via confidence filtering (Yu et al., 2021) or uncertainty estimation (Mukherjee and Hassan Awadallah, 2020), the student can often outperform its teacher. However, as the teacher is trained solely on the weak labels, it can easily inherit unexpected biases and provide misleading signals to the student. In MSR, instead of using a fixed teacher to provide pseudo-labels, we use student performance on the clean validation set as a feedback signal to dynamically refine the teacher. Specifically, the objective for the teacher f, formulated as in Equation 3, is that *the student network, after fitting the teacher's output labels on $\mathcal{D}_a$, can perform best on the validation set $\mathcal{D}_v$*:

$$\begin{aligned} f^\star &= \arg\min_f \mathbb{E}_{(x_i^v, y_i^v)\in\mathcal{D}_v} \mathcal{L}(y_i^v, g'_f(x_i^v)) \\ g'_f &= \arg\min_g \mathbb{E}_{x_i\in\mathcal{D}_a} \mathcal{L}(f(x_i), g(x_i)) \end{aligned} \tag{3}$$

where g' is the student network after fitting output labels from f on $\mathcal{D}_a$. Intuitively, MSR aims to find the best teacher to help the student achieve the lowest validation loss. After finding the optimal teacher $f^\star$ in Equation 3, the student can then be obtained by learning from the output labels of $f^\star$:

$$g^\star = \arg\min_g \mathbb{E}_{x_i\in\mathcal{D}_a} \mathcal{L}(f^\star(x_i), g(x_i)) \tag{4}$$

Algorithm 1: MSR Algorithm

```
Input: Initial teacher network $f_1$ trained according to
       Eq. 1. Student network $g_1$, number of training
       steps $T$, teacher's learning rate scheduler
       $R(t)$, confidence threshold $\tau$, $\mathcal{D}_a$, $\mathcal{D}_v$.
Result: $f_T$, $g_T$
for $t \leftarrow 1 \dots T$ do
    $\{x_i\} \leftarrow$ SampleMiniBatch( $\mathcal{D}_a$)
    $\{x_i^v, y_i^v\} \leftarrow$ SampleMiniBatch( $\mathcal{D}_v$)
    // Teacher Update
    $\tilde{g}_t \leftarrow$ Copy($g_t$)
    $\tilde{g}_{t+1} \leftarrow \tilde{g}_t - \lambda_s \mathbb{E}_{x_i} \nabla_{\tilde{g}_t} \mathcal{L}(f_t(x_i), \tilde{g}_t(x_i))$
    $f_{t+1} \leftarrow f_t - R(t)\,\mathbb{E}_{(x_i^v, y_i^v)} \nabla_{f_t} \mathcal{L}(y_i^v, \tilde{g}_{t+1}(x_i^v))$
    // Student Update
    $w(f_{t+1}(x_i)) \leftarrow \mathbb{1}(1 - \frac{H(f_{t+1}(x_i))}{\log(k)} \geq \tau)$
    $g_{t+1} \leftarrow g_t -$
        $\lambda_s \mathbb{E}_{x_i} \nabla_{g_t} w(f_{t+1}(x_i)) \mathcal{L}(f_{t+1}(x_i), g_t(x_i))$
end
```

4.2 Training Details

Finding the exact $f^\star$ in Equation 3 involves solving two nested loops of optimization, and each loop can be computationally expensive given the large size of $\mathcal{D}_a$. We resort to an online approximation to merge Equation 3 and 4 into an iterative training pipeline. At each training step t, the teacher f_t is first updated based on the meta-objective of "learning to teach", the student g_t is then trained by the updated teacher.

Teacher Update. To update the teacher in an efficient way, we approximate the inner loop in Equation 3 with a single-step gradient descent of the student network. Namely, the objective of the teacher is changed so that the current student, after *one single gradient descent step* of fitting the teacher, can perform best on the validation set. To do so, the teacher will first conduct a "teaching experiment" on a copy of the current student, denoted as $\tilde{g}_t$. $\tilde{g}_t$ is updated for one gradient descent step to fit the teacher's pseudo labels[3]:

$$\tilde{g}_{t+1} = \tilde{g}_t - \lambda_s \, \mathbb{E}_{x_i \sim \mathcal{D}_a} \, \nabla_{\tilde{g}_t} \mathcal{L}(f_t(x_i), \tilde{g}_t(x_i))$$

where λ_s is the learning rate of the student network. Afterwards, we update the teacher network to minimize the validation loss of $\tilde{g}_{t+1}$:

$$f_{t+1} = f_t - \lambda_t \, \mathbb{E}_{(x_i^v, y_i^v) \sim \mathcal{D}_v} \, \nabla_{f_t} \mathcal{L}(y_i^v, \tilde{g}_{t+1}(x_i^v))$$

where λ_t is the learning rate of the teacher network. It requires calculating second derivatives over f_t. We always use soft labels from the teacher for $\mathcal{L}(f_t(x_i), \tilde{g}_t(x_i))$, so the whole process is fully differentiable. Note that $\tilde{g}_t$ is only used in the "teaching experiment" to help update the teacher. It will be discarded after the teacher is updated.

Student Update. After obtaining f_{t+1}, the real student network is updated with the same objective as in Equation 4, except that we use the updated teacher f_{t+1} instead of $f^{\star}$. As the teacher has performed the "teaching experiment", it will provide more useful signals to guide the student.[4]

Teacher Learning Rate Scheduler. We find the teacher is rather sensitive to its learning rate in practice. If the learning rate is large from the start, the teacher may over-adjust itself due to the large performance gap between the teacher and the student. If the learning rate is small, the teacher will adjust itself too slowly so that more noisy pseudo-labels are passed to the student network. Therefore, we apply a linear learning rate scheduler $R(t) = \frac{t\lambda_t}{T}$ to the teacher network where t denotes the current iteration and λ_t is the targeted learning rate for the teacher. By this means, the teacher's learning rate will gradually increase as it gets better at teaching.

Confidence-Based Label Filtering. Despite having the opportunity to refine itself, the teacher inevitably produces some wrong pseudo labels during training, especially at early iterations of self-refinement. To further reduce error propagation, we only select labels with high confidence to guide the student model. The student is updated as follows:

$$g_{t+1} = g_t - \lambda_s \, \mathbb{E}_{x_i \sim \mathcal{D}_a} \, \nabla_{g_t} \mathcal{L}(f_{t+1}(x_i), g_t(x_i)) \times \mathbb{1}(1 - \frac{H(f_{t+1}(x_i))}{\log(k)} \geq \tau)$$

where $\mathbb{1}$ is the indicator function, $H(f_{t+1}(x_i))$ is the entropy of the distribution $f_{t+1}(x_i)$, k is the number of classes in $\mathcal{Y}$ and τ is a pre-defined confidence threshold. $\log(k)$ is the upper bound of the entropy for k-classification tasks. By this means, only low-entropy (high-confidence) predictions from the teacher are learned. Note that the filtering strategy is only applied to the actual student update step, not during the teaching experiment. Otherwise, the teacher will ignore low-confident samples as they do not contribute to teacher update.

Putting all together, Algorithm 1 summarizes the self-refinement process.

5 Experimental Settings

Datasets. WRENCH (Zhang et al., 2021) is a well-established benchmark for weak supervision and offers weak labels for various datasets. We compare different baselines on six NLP datasets from WRENCH including both sequence classification and Named-Entity Recognition (NER) tasks. For sequence classification, we include AGNews (Zhang et al., 2015), IMDB (Maas et al., 2011), Yelp (Zhang et al., 2015), and TREC (Li and Roth, 2002). For NER tasks, CoNLL-03 (Sang and De Meulder, 2003) and OntoNotes 5.0 (Pradhan et al., 2013) are used. In addition, we further include two sequence classification datasets in low-resource languages, Yorùbá and Hausa (Hedderich et al., 2020), to involve evaluation cases in diverse languages. Table 1 summarizes the basic statistics of the datasets. Majority voting over weak sources is used to determine a single label for each sample.

Dataset	Task	# Class	# Train	# Val	# Test
AGNews	Topic	4	96,000	12,000	12,000
IMDB	Sentiment	2	20,000	2,500	2,500
Yelp	Sentiment	2	30,400	3,800	3,800
TREC	Question	6	4,965	500	500
Yorùbá	Topic	7	1,340	189	379
Hausa	Topic	5	2,045	290	582
CoNLL03	NER	4	14,041	3,250	3,453
OntoNotes5.0	NER	18	115,812	5,000	22,897

Table 1: Dataset statistics. Refer to Appendix A for more details on datasets.

Implementation. RoBERTa-base (Liu et al., 2019) is used as the PLM for English datasets and

---

[3] We use SGD for illustration purposes. The AdamW (Loshchilov and Hutter, 2019) optimizer is used in our experiments.

[4] In theory, if the teacher network is strong enough to generalize among different batches, we can directly update the real student in the "teaching experiment", in the hope that the teacher from the last step can also work in the current batch. However, in practice, we find this mismatch leads to poor performance.

multilingual BERT-base (Devlin et al., 2019) for non-English ones. We utilize the higher[5] library to perform second-order optimization. Refer to Appendix B for detailed hyper-parameter configurations.

Baselines. We compare our method with prior work on learning with noisy labels. 1) **Majority** applies majority vote on the weak labels. Ties are broken by randomly selecting a weak label. 2) **Snorkel** (Ratner et al., 2017) trains a labeling model that aggregates weak labels from different weak sources. 3) **FT-WL** fine-tunes PLMs on the weak labels. 4) **FT-WLST** further applies classic self-training (Lee et al., 2013) on the model obtained by FT-WL. 5) **L2R** (Ren et al., 2018) uses a meta-learning framework to reweight weakly labeled samples. 6) **Meta-Weight-Net** (Shu et al., 2019) also applies meta-learning based sample reweighting. However, the weights are computed through an external reweighting network. 7) **Denoise** (Ren et al., 2020) iteratively corrects wrong annotations in the training set, and the classifier learns with the corrected labels. 8) **UST** (Mukherjee and Hassan Awadallah, 2020) is a self-training based method that assigns higher weights to samples that the teacher is certain about. The uncertainties are measured via MC-dropout on the predictions (Gal and Ghahramani, 2016). 9) **COSINE** (Yu et al., 2021) trains its student network with pseudo-labels which the teacher is highly confident about. In addition, contrastive regularization is introduced to further alleviate error propagation.

For our proposed framework, we report the performances of both **Teacher-Init** (f_1): the initial teacher trained directly on weak labels, and **MSR**: the final student model (g_T). f_1 is obtained by running FT-WL five times and selecting the best one among them according to the validation performance. *For a fair comparison, the same f_1 is used as the initial teacher for all self-training based models.* Finally, we also include the results of fine-tuning PLMs on the clean versions of each dataset, denoted by **FT-CL**, to represent the upper bound performance.

6 Results

Comparison with Baselines. Table 2 shows a comparison among different methods. MSR matches or outperforms SOTAs on all eight datasets. FT-WL outperforms majority voting over the weak labels in all cases except Hausa, which leads to a minor drop. This confirms that PLMs encode useful knowledge in their parameters, enabling them to generalize better than weak rules they are trained on. This phenomenon is particularly noticeable on AGNews, IMDB, and Yelp: direct fine-tuning on the noisy labels (FT-WL) can already achieve decent performance (accuracy above 83%). *We consider them easy tasks since label noise has only a minor impact on performance of PLMs and decent generalization can be attained even without specific noise-handling.* Applying self-training to such simple tasks lead to further performance improvement. COSINE, a SOTA self-training based model, can even perform comparably to the fully supervised model on these three datasets. On the other five datasets, however, FT-WL performs poorly and conventional self-training methods provide little performance boost (even a disservice on OntoNotes). This implies that *self-training relies on a well-performed initial teacher to work effectively.* On challenging datasets where the initial teacher is weak, it struggles to achieve further performance gain. Meta-learning based methods such as L2R performs better than COSINE on these challenging datasets. *MSR can further boost the performance on all the challenging datasets by up to 11.4% in accuracy or 9.26% in F1 score while maintaining comparable results on simpler datasets.*

Error Decomposition. Let $y', \hat{y}, y$ denote the model prediction, the noisy weak label, and the clean label, respectively. To investigate how the label noise influences the model predictions, we decompose model prediction errors into three types: (1) Type-A error: $y' = \hat{y}; \hat{y} \neq y$ (2) Type-B error: $y' \neq \hat{y} \neq y$ and (3) Type-C error: $y' \neq y; y = \hat{y}$. Type-A/B errors correspond to situations in which a model complies with an incorrect weak label $\hat{y}$, or predicts another incorrect class label. If, on the other hand, the weak label $\hat{y}$ is correct, a Type-C error arises if the model predicts a label different than $\hat{y}$. A higher Type-A error rate indicates that a model memorizes more label noise from the weak sources, while a model that underfits fails to learn useful knowledge from the weak sources can have a higher Type-C error rate.

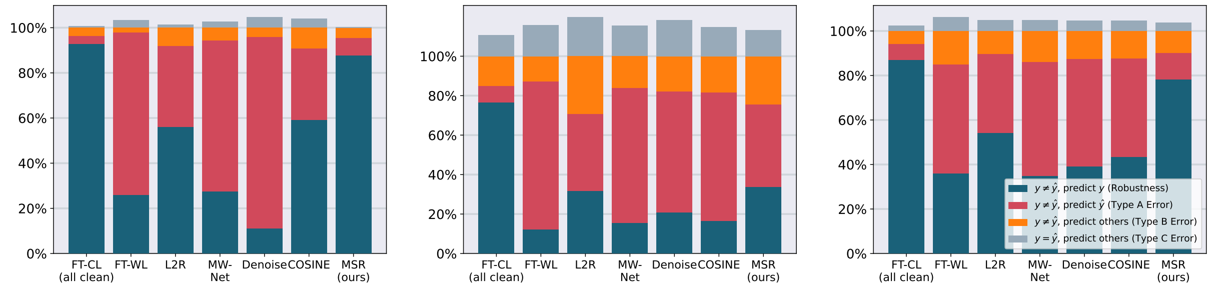
Figure 3 visualizes the three types of errors on three challenging datasets: TREC, Hausa and CoNLL-03. The blue bars represent model robust-

---

[5] https://github.com/facebookresearch/higher

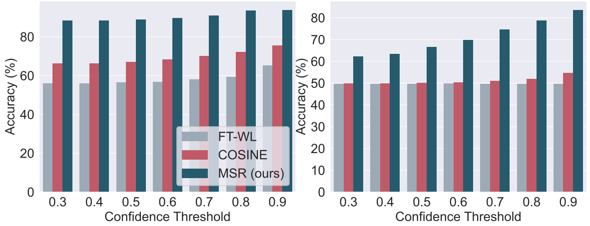
Method	AGNews (Acc)	IMDB (Acc)	Yelp (Acc)	TREC (Acc)	Yorùbá (Acc)	Hausa (Acc)	CoNLL-03 (F1)	OntoNotes (F1)
Fully-Supervised Result								
FT-CL	92.61	93.20	96.91	96.67	77.24	81.57	92.27	85.74
Label Models								
Majority	63.84	71.04	70.21	60.80	58.05	47.93	60.38	58.92
Snorkel (Ratner et al., 2017)	62.67	71.60	68.92	59.60	62.80	47.94	62.88	58.46
DNN Baselines								
FT-WL	$85.73_{\pm 0.43}$	$83.43_{\pm 0.91}$	$87.71_{\pm 1.46}$	$66.80_{\pm 1.44}$	$64.12_{\pm 0.83}$	$46.13_{\pm 0.43}$	$69.20_{\pm 0.33}$	$67.26_{\pm 0.62}$
FT-WLST† (Lee et al., 2013)	$88.61_{\pm 0.71}$	$89.50_{\pm 0.65}$	$95.32_{\pm 0.70}$	$76.00_{\pm 2.21}$	$67.28_{\pm 1.12}$	$49.22_{\pm 1.39}$	$69.87_{\pm 0.36}$	$64.13_{\pm 1.45}$
L2R (Ren et al., 2018)$^{\diamond}$	$87.28_{\pm 1.00}$	$82.76_{\pm 1.59}$	$93.34_{\pm 0.91}$	$83.40_{\pm 2.01}$	$70.45_{\pm 0.69}$	$55.67_{\pm 0.88}$	$79.15_{\pm 1.34}$	$70.66_{\pm 0.74}$
Meta-Weight-Net$^{\diamond}$ (Shu et al., 2019)	$85.96_{\pm 0.80}$	$86.72_{\pm 0.50}$	$86.97_{\pm 0.74}$	$69.39_{\pm 1.27}$	$70.00_{\pm 2.12}$	$48.63_{\pm 0.96}$	$69.54_{\pm 1.43}$	$69.11_{\pm 1.20}$
Denoise (Ren et al., 2020)	$83.45_{\pm 0.68}$	$76.22_{\pm 0.92}$	$71.56_{\pm 0.56}$	$61.80_{\pm 1.30}$	$66.10_{\pm 1.52}$	$49.31_{\pm 0.93}$	$72.96_{\pm 0.51}$	$67.64_{\pm 1.06}$
UST† (Mukherjee and Hassan Awadallah, 2020)	$87.78_{\pm 0.59}$	$86.74_{\pm 1.18}$	$91.23_{\pm 0.90}$	$77.20_{\pm 2.29}$	$68.12_{\pm 0.71}$	$47.67_{\pm 0.91}$	$69.48_{\pm 1.69}$	$66.98_{\pm 0.99}$
COSINE† (Yu et al., 2021)	$89.34_{\pm 0.76}$	$\mathbf{90.52}_{\pm 1.06}$	$\mathbf{95.48}_{\pm 0.13}$	$82.60_{\pm 1.09}$	$68.87_{\pm 0.82}$	$49.66_{\pm 1.32}$	$70.60_{\pm 0.87}$	$64.59_{\pm 1.08}$
Our Framework								
Teacher-Init (f_1)	$86.37_{\pm 0.00}$	$85.00_{\pm 0.00}$	$89.92_{\pm 0.00}$	$69.00_{\pm 0.00}$	$65.44_{\pm 0.00}$	$46.74_{\pm 0.00}$	$69.73_{\pm 0.00}$	$68.25_{\pm 0.00}$
MSR$^{\dagger\ \diamond}$	$\mathbf{89.92}_{\pm 0.64}$	$89.16_{\pm 0.91}$	$95.00_{\pm 0.35}$	$\mathbf{94.80}_{\pm 0.29}$	$\mathbf{72.56}_{\pm 0.78}$	$\mathbf{59.11}_{\pm 0.78}$	$\mathbf{88.41}_{\pm 0.63}$	$\mathbf{74.59}_{\pm 0.84}$

Table 2: Accuracy and F1 score (in %) on eight NLP tasks. The mean and standard deviation over five trials are reported. Teacher-Init is the best model checkpoint selected from the five trials of FT-WL (according to the validation performance). For a fair comparison, all self-training-based models use the same Teacher-Init checkpoint. MSR matches or outperforms SOTAs on all tasks. † self-training based method. $^{\diamond}$ meta-learning based method.



(a) TREC (b) Hausa (c) CoNLL-03

Figure 3: Prediction error decomposition of various weak supervision baselines, evaluated on the test sets. A model is considered robust against label noise if it manages to predict the correct labels despite the wrong weak labels (the robustness is represented by the blue bars). Otherwise, it conforms to the weak label (Type-A error) or predict another incorrect label (Type-B error), which has a negative effect on generalization. The Type-C error rate quantifies the proportion of incorrect model predictions when weak labels are correct. MSR consistently reduces the Type-A error rate and attains a high level of noise robustness.

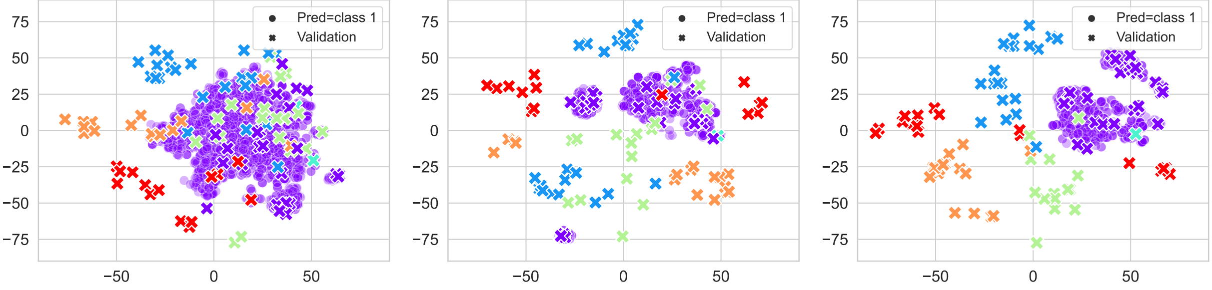


(a) TREC (b) Hausa

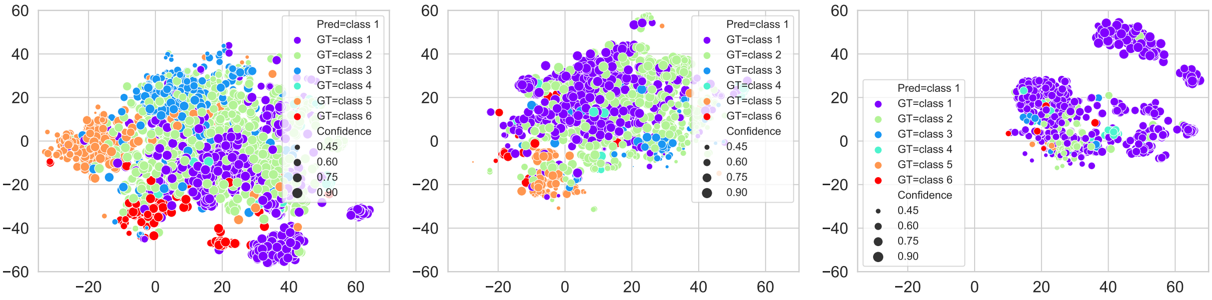
Figure 4: Accuracy *vs.* confidence thresholds.

ness, i.e., how often the model predicts correctly when $\hat{y} \neq y$. It clearly shows that direct fine-tuning on weak labels (FT-WL) has a much higher Type-A error rate compared with the model trained on clean data (FT-CL), suggesting that the model quickly memorizes the label noise. On the other hand, the disparity in type C error rate is much smaller, indicating that all models do not underfit and the knowledge from the weak sources is properly transferred. The Type-B error shows similar trends and does not differ much across models. Overall, Type-A error has the strongest impact on model performance. *All the noisy-handling models mainly help with reducing Type-A errors*. We also observe that while COSINE reduces Type-A errors on TREC, it barely works on the other two datasets. Only MSR manages to consistently reduce Type-A errors by over 20% on all three datasets.

Accuracy vs Confidence. As confidence-based filtering is a key component in both COSINE and MSR, we show the accuracy of model predictions with different confidence thresholds in Figure 4. As can be seen, *even using a high confidence threshold for COSINE, the accuracy is still low*, which is why it struggles to improve on challenging datasets. MSR, on the contrary, consistently attains higher accuracy with higher confidence thresholds, and thereby confidence-based filtering on top of MSR



(a) Teacher-Init (b) MSR student at step 1500 (c) MSR student at step 3000

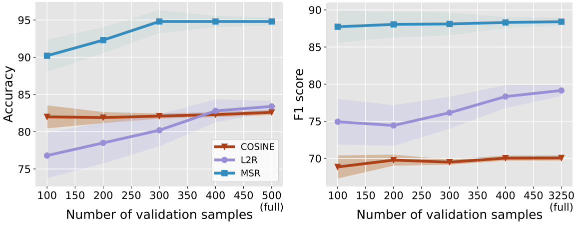


(d) Teacher-Init (e) COSINE (f) MSR

Figure 5: Projected feature space of different models on TREC using t-SNE (Van der Maaten and Hinton, 2008). The circles represent training samples that are predicted as class 1. **a)-c)**: development of MSR during training. Circles are colored by the predicted class (i.e., class 1, in purple). The validation samples are represented by crosses and colored according to the ground truth labels. The MSR student gradually improves its feature space to embed the training and validation samples from the same class in the same area. **d)-f)**: training samples are colored according to their ground truth labels; model confidence is reflected by the size of the circles. Teacher-Init and COSINE misclassify samples with high confidence. MSR attains a cleaner cluster.

help lead to better performance.

Impact of Label Noise on Feature Space. We also analyze how the label noise influences representation learning. Figure 5 illustrates the projected feature space of different models on TREC. For a clear visualization, we present only training samples predicted as class 1 by the models in the form of circles. In figs. 5a to 5c, we further visualize the feature space of validation samples (represented by crosses). As can be seen, initially the feature space of class 1 overlaps with that of other classes from the validation set. As the training proceeds, when the teacher keeps refining itself, the MSR student gradually reduces such overlap and learns a well-split representation space. In figs. 5d to 5f, we compare the feature space between different models. The training samples are colored according to their ground truth classes to highlight the misclassification ratio (the more colorful the clusters, the higher the misclassification ratio). We observe that Teacher-Init makes many wrong predictions with high degree of confidence. In this case, utilizing the confidence score for denoising is fragile. This may explain why COSINE, despite offering a more compact cluster, still has a considerable amount of misclassification. Finally, MSR has a considerably cleaner cluster and is less affected by error propagation than COSINE.



(a) TREC (b) CoNLL-03

Figure 6: Accuracy *vs.* number of validation samples.

Effects of Validation Data Size. The model performance reported in Table 2 is based on the original data splits from the WRENCH benchmark. The size of the validation sets is mostly less than 15% of the training sets. Typically, they are used to perform early stopping and model selection. For meta-learning based methods, they additionally rely on the validation sets for meta-update and might be more sensitive to validation size. Hence, we study how the validation size affects different models. In particular, we randomly sample a subset from the original validation set $\mathcal{D}_v$ and repeat the same training process. Figure 6 presents the results. We find that the validation size indeed has a greater impact on meta-learning approaches. However, *MSR still retains its high generalization performance even with as few as 100 validation samples*, suggesting that MSR is very data efficient in performing the self-refinement.

Configuration	**Seq. Classification (Acc)**	**NER (F1)**
Teacher-Init	73.75	68.99
Student	**83.43**	**81.50**
Teacher	82.38 (↓ 1.05%)	80.26 (↓ 1.24%)
w/o Teacher Scheduler	81.80 (↓ 1.63%)	80.15 (↓ 1.35%)
w/o Confidence Filtering	82.32 (↓ 1.11%)	81.09 (↓ 0.41%)
w/o Both	81.63 (↓ 1.80%)	79.95 (↓ 1.55%)

Table 3: Summary of ablation experiments aggregated across multiple datasets. See Appendix D for results in each dataset.

Ablation Study. Table 3 summarizes the impact of different components of our method. In general, our student model performs slightly better than the teacher. This is as expected because a) the teacher's goal is to guide the student to generalize better, the training loss does not explicitly encourage the teacher to improve its accuracy, and b) the confidence filtering helps the student avoid fitting some wrong pseudo-labels from the teacher. This is also confirmed by the decreased performance when the filter is removed. In addition, applying a learning rate scheduler is better than using a fixed learning rate throughout training.

7 Conclusion

We present MSR, a meta-learning based self-refinement framework that enables robust learning with weak labels. Unlike conventional self-training which relies on a fixed teacher, MSR dynamically refines the teacher based on the student's performance on the validation set. To further suppress error propagation, we introduce a learning rate scheduler to the teacher and add confidence filtering to the student. We demonstrate that our framework performs on par with or better than current SOTAs on both sequence classification and labeling tasks.

Limitations

In this work, Our primary focus is to propose a strong weak supervision method that works reliably under various weak supervision settings. We employ meta-learning techniques to address the issue of unreliable confidence scores under challenging settings (Figure 4). Despite the effectiveness, the main limitation of our method, just like other meta-learning based frameworks, is the computational overhead. The teacher update step (Algorithm 1, Line 4-6) requires computing both the first and second-order derivatives, which incurs additional computation time and higher memory consumption. Consequently, our method requires longer training.[6] Implementation-wise and computation-wise, MSR is as complex as other existing meta-learning based methods, like L2R (Ren et al., 2018) and MW-Net (Shu et al., 2019), but performs substantially better than them in all weak supervision scenarios we evaluated. It is worth noting that MSR has *no overhead at inference time*. In weak supervision, the data annotation cost is considered the most significant bottleneck. A stronger model is often obtained by trading some more computation with the cost and effort of obtaining more human-generated, manual annotations. Hence, the one-off investment of training MSR can be worthwhile for real-world weak supervision applications.

Acknowledgments

This work has been partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 232722074 – SFB 1102 and the EU Horizon 2020 projects ROXANNE under grant number 833635.

References

Dana Angluin and Philip Laird. 1988. Learning from noisy examples. *Machine Learning*, 2(4):343–370.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Alan Joseph Bekker and Jacob Goldberger. 2016. Training deep neural-networks based on unreliable labels. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, pages 2682–2686. IEEE.

Allan Peter Davis, Thomas C. Wiegers, Phoebe M. Roberts, Benjamin L. King, Jean M. Lay, Kelley Lennon-Hopkins, Daniela Sciaky, Robin J. Johnson, Heather Keating, Nigel Greene, Robert Hernandez,

---

[6] Detailed training time on each dataset can be found in Appendix E The most costly training of MSR takes roughly 3 hours.

Kevin J. McConnell, Ahmed Enayetallah, and Carolyn J. Mattingly. 2013. A ctd-pfizer collaboration: manual curation of 88 000 scientific articles text mined for drug-disease and drug-phenotype interactions. *Database J. Biol. Databases Curation*, 2013.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Daniel Fu, Mayee Chen, Frederic Sala, Sarah Hooper, Kayvon Fatahalian, and Christopher Ré. 2020. Fast and three-rious: Speeding up weak supervision with triplet methods. In *International Conference on Machine Learning*, pages 3280–3291. PMLR.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059. JMLR.org.

Jacob Goldberger and Ehud Ben-Reuven. 2017. Training deep neural-networks using a noise adaptation layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Keren Gu, Xander Masotto, Vandana Bachani, Balaji Lakshminarayanan, Jack Nikodem, and Dong Yin. 2021. An instance-dependent simulation framework for learning with label noise. *arXiv preprint arXiv:2107.11413*.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8536–8546.

Michael A. Hedderich, David Ifeoluwa Adelani, Dawei Zhu, Jesujoba O. Alabi, Udia Markus, and Dietrich Klakow. 2020. Transfer learning and distant supervision for multilingual transformer models: A study on african languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2580–2591. Association for Computational Linguistics.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. 2018. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10477–10486.

Giannis Karamanolakis, Subhabrata (Subho) Mukherjee, Guoqing Zheng, and Ahmed H. Awadallah. 2021. Self-training with weak supervision. In *NAACL 2021*. NAACL 2021.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Junnan Li, Richard Socher, and Steven C. H. Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1518–1533. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Aditya Krishna Menon, Brendan van Rooyen, and Nagarajan Natarajan. 2016. Learning from binary labels with instance-dependent corruption. *CoRR*, abs/1605.00751.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. In *Advances in Neural Information Processing Systems (NeurIPS 2020)*, Online.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2233–2241. IEEE Computer Society.

Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V. Le. 2021. Meta pseudo labels. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 11557–11568. Computer Vision Foundation / IEEE.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4331–4340. PMLR.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising multi-source weak supervision for neural text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3739–3754. Association for Computational Linguistics.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1917–1928.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Yaqing Wang, Subhabrata Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed Hassan Awadallah. 2020. Adaptive self-training for few-shot neural sequence labeling. *arXiv preprint arXiv:2010.03680*.

Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10684–10695. Computer Vision Foundation / IEEE.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics, 26-30 June 1995, MIT, Cambridge, Massachusetts, USA, Proceedings*, pages 189–196. Morgan Kaufmann Publishers / ACL.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2021. Fine-tuning pretrained language model with weak supervision: A contrastive-regularized self-training approach. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1063–1077. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. 2021. WRENCH: A comprehensive benchmark for weak supervision. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2022. Bert learns to teach: Knowledge distillation with meta learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Wenxuan Zhou, Hongtao Lin, Bill Yuchen Lin, Ziqi Wang, Junyi Du, Leonardo Neves, and Xiang Ren. 2020. NERO: A neural rule grounding framework for label-efficient relation extraction. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2166–2176. ACM / IW3C2.

Dawei Zhu, Michael A. Hedderich, Fangzhou Zhai, David Ifeoluwa Adelani, and Dietrich Klakow. 2022. Is BERT robust to label noise? A study on learning with noisy labels in text classification. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP, Insights@ACL 2022, Dublin, Ireland, May 26, 2022*, pages 62–67. Association for Computational Linguistics.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. 2020. Rethinking pre-training and self-training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

A Dataset Details

We experiment with eight NLP datasets, including six English datasets and two datasets in low-resource languages. All datasets come with their ground truth annotations and as well as the weak labels.

A.1 Datasets Selection Criteria

The WRENCH (Zhang et al., 2021) benchmark contains 23 NLP datasets. We choose representative datasets (like previous research in weak supervision) that **a)** overlap with previous works to enable direct comparisons. **b)** are diverse in terms of weak label quality, languages and tasks to approve the applicability of different baselines.

A.2 English Datasets

We experiment with four popular sequence classification datasets: AGNews, IMDB, Yelp and TREC.

1. **AGNews** (Zhang et al., 2015): originates from AG, which is a large collection of news articles. The news are categorized in four classes: "World", "Sports", "Business" and "Sci/Tech".
2. **IMDB** (Maas et al., 2011): consists of movie reviews with binary labels. It is a commonly used benchmark dataset for sentiment analysis.
3. **Yelp** (Zhang et al., 2015): obtained from the Yelp Dataset Challenge in 2015. Similar to IMDB, it is a sentiment analysis dataset.
4. **TREC** (Li and Roth, 2002): categorizes the questions in TREC-6 datasets into 6 categories: "Abbreviation", "Entity", "Description", "Human", "Location", "Numeric-value".

and with the two sequence labeling datasets: CoNLL-03 and OntoNotes 5.0.

1. **CoNLL-03** (Sang and De Meulder, 2003) NER dataset with four named-entity categories.
2. **OntoNotes 5.0** (Pradhan et al., 2013): NER dataset with 18 named-entity categories.

All weak labels are obtained from the WRENCH benchmark[7] (Zhang et al., 2021).

A.3 Datasets in Low-Resource Languages

Most datasets in the current WRENCH benchmarks are in English. Although weak supervision is desired in low-resource languages, it is understudied as finding annotators for them is more difficult. Hence, we include two low-resource languages, Yorùbá and Hausa, to cover this scenario. Often, learning with weak labels in low-resource languages is more challenging. First, the training set is often much smaller than English datasets. For example, Hausa has only about 2k training samples while AGNews have 96k. Second, the weak labels in low-resource languages can have lower quality as experts for weak source development are harder to find. A set of simple rules is often used for labeling (which is the case in Yorùbá and Hausa). Hence, weak supervision with low-resource languages is a combination of two challenges: training with *small* datasets which have *low-quality* labels.

Yorùbá and Hausa are text classification datasets obtained from (Hedderich et al., 2020).[8]

1. **Yorùbá**: consists of news headlines from BBC Yoruba which are categorized in seven classes: "Nigeria", "Africa", "World", "Entertainment", "Health", "Sport", "Politics".
2. **Hausa**: consists of news headlines from VOA Hausa which have the same seven classes as Yorùbá. However, only five classes are considered in the text classification task. "Entertainment" and "Sport" have been removed due to the lack of samples of these classes.

Hedderich et al. (2020) provided both the clean labels and weak labels on the two datasets. A gazetteer is created for each class for weak supervision. For example, a gazetteer containing names of agencies, organizations, states and cities in Nigeria is used to label the class "Nigeria".

A.4 More Dataset Statistics

We provide more details on the datasets we used in our experiments in Table 4. In general, not all data can be covered by weak sources. No weak source is triggered for some training samples and they remain unlabeled. The coverage of the datasets ranges from 69.08% to 100%. Note that for NER tasks, the coverage is always 100% since if no weak source is triggered for a token, we assign

---

[7] `https://github.com/JieyuZ2/wrench`

[8] `https://github.com/uds-lsv/transfer-distant-transformer-african`

Dataset	Task	# Class	$\lvert\mathcal{D}_w\rvert$	$\lvert\mathcal{D}_a\rvert$	Coverage	Conflict	$\lvert\mathcal{D}_v\rvert$	$\lvert\mathcal{D}_t\rvert$
AGNews	Topic	4	66,314	96,000	69.08%	14.17%	12,000	12,000
IMDB	Sentiment	2	17,515	20,000	87.58%	11.96%	2,500	2,500
Yelp	Sentiment	2	25,165	30,400	82.78%	18.29%	3,800	3,800
TREC	Question	6	4,723	4,965	95.13%	22.76%	500	500
Yorùbá	Topic	7	1,340	1,340	100.00%	1.87%	189	379
Hausa	Topic	5	2,045	2,045	100.00%	1.90%	290	582
CoNLL03	NER	4	14,041	14,041	100.00%	4.05%	3,250	3,453
OntoNotes5.0	NER	18	115,812	115,812	100.00%	1.86%	5,000	22,897

Table 4: Dataset statistics. $|\mathcal{D}_w|$: number of training samples with weak labels. $|\mathcal{D}_a|$: total number of training samples (weakly labeled + unlabeled). Coverage: fraction of samples that are weakly labeled, i.e., $\frac{|\mathcal{D}_w|}{|\mathcal{D}_a|}$. Conflict: samples that are labeled by at least two weak sources with contradicted weak labels. $|\mathcal{D}_v|$: number of validation samples. $|\mathcal{D}_t|$: number of test samples.

Hyperparameter	Search Range
Teacher Learning Rate	3e-6, 5e-6, 2e-5, 3e-5
Teacher Warm-Up Steps	500, 100, 2000, 3000
Confidence Filter Threshold	0.4, 0.5, 0.6, 0.7, 0.8, 0.95

Table 5: Hyperparameter search.

label "O" (i.e., non-entity) to it. On the other hand, some samples can be covered by two or more weak sources with contradicted weak labels. In this case, we have a conflict. The conflict ratio ranges from 1.86% to 22.76% in the datasets we tested.

B Implementation Details

Models. All baselines in our paper, except the majority vote and the Snokerl model (Ratner et al., 2017) which work with label space only, use the official RoBERTa model[9] (Liu et al., 2019) from Huggingface as the classification backbone for all English datasets, and the multilingual BERT[10] for datasets in African languages. We use the base version of the two models which contain roughly 120M and 110M parameters, respectively.

Fine-Tuning on Classification Task. We fine-tune all layers using AdamW (Loshchilov and Hutter, 2019) as the optimizer. For sequence classification tasks, we pass the final layer of the [CLS] token representation ($\mathbb{R}^{768}$) to a feed-forward layer for prediction. For sequence labeling tasks, the final layers of all tokens ($\mathbb{R}^{768\times L}$, where L is the sentence length) are passed to a shared feed-forward layer to predict the class of each token in the sentence. We report the score where the model performs the best on the validation set during training.

Hyper-Parameters of MSR. We apply grid search on the warm-up steps for the teacher and the confidence threshold for the student network. Table 5 shows our hyperparameter search configuration. We choose the final configurations of the hyperparameters according to the model's performance on the validation set. Table 6 shows the best configurations of parameters we used to produce the results in Table 2.

Evaluation Metrics. For model evaluation, we report accuracy for sequence classification tasks and F1 Score for sequence labeling tasks. In our implementation, we call the function `classification_report()` from the scikit-learn library[11] to compute the accuracy, and use the `Seqeval` class from Huggingface[12] to compute the F1 Score.

C Validation Performance

The average test performance of MSR is reported in Table 2. We further report the corresponding validation performance in Table 7.

---

[9] https://huggingface.co/roberta-base

[10] https://huggingface.co/bert-base-multilingual-cased

[11] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

[12] https://github.com/huggingface/datasets/blob/master/metrics/seqeval/seqeval.py

	AGNews	**IMDB**	**Yelp**	**TREC**	**Yorùbá**	**Hausa**	**CoNLL-03**	**OntoNotes 5.0**
BERT Backbone	RoBERTa	RoBERTa	RoBERTa	RoBERTa	mBERT	mBERT	RoBERTa	RoBERTa
Batch Size	32	16	16	32	32	32	32	32
Maximum Sequence Length	128	256	256	64	64	128	64	64
Student Learning Rate	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
Teacher Learning Rate	2e-5	2e-5	2e-5	2e-5	5e-6	2e-5	2e-5	2e-5
Teacher Warm-Up Steps	500	500	3000	500	1000	3000	2000	2000
Confidence Filter Threshold	0.7	0.7	0.5	0.5	0.7	0.4	0.8	0.5

Table 6: Selected hyperparameters. mBERT: multilingual BERT.

Dataset	**Test**	**Validation**
AGNews	89.92	89.90
IMDB	89.16	89.21
Yelp	95.00	94.79
TREC	94.80	94.42
Yorùbá	72.56	75.13
Hausa	59.11	62.34
CoNLL-03	88.41	87.86
OntoNotes	74.59	75.20

Table 7: The average test and validation accuracy/F1 score (in %) of MSR over five trials.

D Ablation Studies

We report the detailed ablation results for each dataset in Table 8.

E Hardware and Average Runtime.

We use Nvidia Tesla V100 to accelerate training. The average runtime for each method and dataset is summarized in Table 9.

MSR Configuration	**AGNews (Acc)**	**IMDB (Acc)**	**Yelp (Acc)**	**TREC (Acc)**	**Yorùbá (Acc)**	**Hausa (Acc)**	**CoNLL-03 (F1)**	**OntoNotes (F1)**
Student	**89.92**	**89.16**	**95.00**	**94.80**	**72.56**	59.11	**88.41**	**74.59**
Teacher	89.02	88.08	94.37	93.80	68.87	**60.14**	87.30	73.22
w/o Teacher Scheduler	89.68	87.68	93.78	93.60	70.71	55.32	87.82	72.48
w/o Confidence Filtering	89.87	89.04	94.76	93.60	71.50	55.15	88.07	74.11
w/o Both	89.55	87.68	93.33	93.40	70.50	55.32	87.82	72.08

Table 8: Ablation studies. The numbers represent the test accuracy and F1 Score.

	AGNews	IMDB	Yelp	TREC	Yorùbá	Hausa	CoNLL-03	OntoNotes 5.0
Running time (hours)	2.5	1.6	0.5	1.2	0.5	0.7	1.1	3.0

Table 9: Average runtime (in hours) for training a MSR model. One single Nvidia Tesla V100 GPU is used in each experiment to accelerate the computation.

Looking for a Needle in a Haystack: A Comprehensive Study of Hallucinations in Neural Machine Translation

Nuno M. Guerreiro[1,2] **Elena Voita**[4] **André F. T. Martins**[1,2,3]

[1]Instituto de Telecomunicações, Lisbon, Portugal
[2]Instituto Superior Técnico & LUMLIS (Lisbon ELLIS Unit), Lisbon, Portugal
[3]Unbabel, Lisbon, Portugal [4]University of Edinburgh, Scotland
{nuno.s.guerreiro, andre.t.martins}@tecnico.ulisboa.pt lena-voita@hotmail.com

Abstract

Although the problem of hallucinations in neural machine translation (NMT) has received some attention, research on this highly pathological phenomenon lacks solid ground. Previous work has been limited in several ways: it often resorts to artificial settings where the problem is amplified, it disregards some (common) types of hallucinations, and it does not validate adequacy of detection heuristics. In this paper, we set foundations for the study of NMT hallucinations. First, we work in a *natural* setting, i.e., in-domain data without artificial noise neither in training nor in inference. Next, we annotate a dataset of over 3.4k sentences indicating different kinds of critical errors and hallucinations. Then, we turn to detection methods and both revisit methods used previously and propose using glass-box uncertainty-based detectors. Overall, we show that for preventive settings, (i) previously used methods are largely inadequate, (ii) sequence log-probability works best and performs on par with reference-based methods. Finally, we propose DeHallucinator, a simple method for alleviating hallucinations at test time which significantly reduces the hallucinatory rate.

1 Introduction

Neural machine translation (NMT) is becoming increasingly accurate (Vaswani et al., 2017; Akhbardeh et al., 2021), particularly in high resource language pairs where parallel data is abundant. However, even the best systems available today may generate *hallucinations*. These are extremely pathological translations that contain content that is unfaithful to the source sequence. Critically, a tiny fraction of these mistakes is all it takes to compromise user trust or safe deployment of NMT models in production.

Unfortunately, although the problem of hallucinations received some attention, research on this highly pathological phenomenon lacks solid ground. First, previous work used multiple and often overlapping definitions and categories of hallucinations which makes it hard to draw connections between observations made in different works (Lee et al., 2018; Raunak et al., 2021; Zhou et al., 2021). Next, since hallucinations are extremely rare, previous work focused on settings in which the phenomenon is amplified, e.g. perturbing data either in training or at inference, or evaluating under domain shift (Lee et al., 2018; Raunak et al., 2021; Müller et al., 2020; Wang and Sennrich, 2020; Voita et al., 2021; Müller and Sennrich, 2021; Zhou et al., 2021). Critically, the analysis on these works mostly relied on the adequacy of the automatic hallucination detection methods they proposed. However, it is not immediate whether these methods translate well to unperturbed settings.

In this work, we set foundations for the study of NMT hallucinations. We take a step back from previous work and, instead of considering perturbed settings for which hallucinations are more frequent, we consider a *natural scenario* and face the actual problem of identifying a small fraction of hallucinations (a “needle”) in a large number of translated sentences (a “haystack”). Then, we provide a rigorous comparison among hallucination detection methods. Apart from analysing those proposed in previous work (e.g., heuristics based on anomalous encoder-decoder attention), we also propose to use simple model uncertainty measures as detectors. For each of these methods, we select examples marked as hallucinations, put them together, and gather human annotations. As a result, we introduce a corpus of 3415 structured annotations for different NMT pathologies and hallucinations. We use this corpus for analysis and show that, in preventive settings where high recall is desirable, previously proposed methods are mostly inadequate, and filtering according to standard sequence log-probability performs the best. In fact, it performs on par with the state-of-the-art COMET (Rei et al.,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1059–1075
May 2-6, 2023

Category	Source Sentence	Reference Translation	Hallucination
Oscillatory	Ist ein Kompromiss aufgrund des zugrundeliegenden Regelsystems unmöglich, so spricht man von Aporie.	The case where, based on the pertinent system of regulations a compromise is not possible, is referred to as Aporia.	Aporia is the name of aporia , which is the name of aporia.
Strongly Detached	Tickets für Busse und die U-Bahn ist zu teuer, vor allem in Stockholm.	Tickets for buses and the subway is too expensive, especially in Stockholm.	The hotel is located in the centre of Stockholm, close to the train station.
Fully Detached	Die Zimmer beziehen, die Fenster mit Aussicht öffnen, tief durchatmen, staunen.	Head up to the rooms, open up the windows and savour the view, breathe deeply, marvel.	The staff were very friendly and helpful.

Table 1: Examples of hallucination types. Hallucinated content is shown shaded.

2020a) which uses reference translation and thus cannot be used in most real-world on-the-fly applications. Surprisingly, its reference-free version COMET-QE (Rei et al., 2020b), which was shown to generally perform on par with COMET (Kocmi et al., 2021; Freitag et al., 2021), substantially fails to penalise the severity of hallucinations. Overall, methods targeting the phenomena are largely unfit, quality estimation systems fail, and sequence log-probability, i.e. a byproduct of generating a translation, turns out to be the best.

Apart from our analysis of detection methods, we propose DeHallucinator, a method for alleviating hallucinations at test time. At a high level, we first apply a lightweight hallucination detector and then, if a translation is flagged, we try to overwrite it with a better version. For this, we generate several MC-dropout hypotheses (Gal and Ghahramani, 2016), score them with some measure, and pick the highest-scoring translation as the final candidate. With this approach, the proportion of correct translations among the ones flagged by the detector increases from 33% to 85%, and the hallucinatory rate decreases threefold.

Overall, we show that (i) in preventive settings, previously proposed hallucination detectors are mostly inadequate; (ii) quality estimation techniques fail to distinguish hallucinations from less severe errors; (iii) sequence log-probability is the best hallucination detector and performs on par with reference-based COMET; and, (iv) our DeHallucinator significantly alleviates hallucinations at test time.

Additionally, we release our annotated dataset along with the model, training data, and code.[1]

2 Taxonomy of Translation Pathologies

Choosing a good taxonomy is a compromise between simplicity (which minimizes annotation effort) and comprehensiveness. Thus, generic quality assessment taxonomies, such as MQM (Lommel et al., 2014), might be unfit or too complex when we focus only on critical errors and hallucinations. For hallucinations, in turn, previous work used multiple, often overlapping, definitions (Lee et al., 2018; Raunak et al., 2021; Zhou et al., 2021; Raunak et al., 2022). The taxonomy we build here is rather general: it covers categories considered previously (Lee et al., 2018; Raunak et al., 2021) and others not reported before. For a broader discussion on the taxonomy of hallucinations in NMT, and how it differs from other natural language generation tasks, refer to Ji et al. (2022).

2.1 Hallucinations

To distinguish hallucinations from other errors, we rely on the idea of detachment from the source sequence. From this perspective, other critical errors such as mistranslation of named entities are not considered as hallucinations. In Section 6 and Appendix D, we show that properties of hallucinations differ a lot from these other errors and thus our taxonomy is very reasonable.

Oscillatory hallucinations. These are inadequate translations that contain erroneous repetitions of words and phrases.

Largely fluent hallucinations. These are largely fluent translations that are unrelated to the content of the source sequence. Previous work assumed they always bear *no relation at all* to the source content (Lee et al., 2018; Raunak et al., 2021). However, we find that a large proportion of fluent hallucinations partially support the source. Therefore, we also consider severity of a hallucination and distinguish translations that are *fully* detached from those that are *strongly* (but not fully) detached.

Note that oscillatory hallucinations can also be either fully or only partially detached, but since these hallucinations are less frequent, in what follows we do not split them by severity. We show examples of these hallucination types in Table 1.

---

[1] All these resources are available at `https://github.com/deep-spin/hallucinations-in-nmt`.

2.2 Translation errors

Undergeneration. These are incomplete translations that do not cover part of the source content. This problem is often studied in isolation (Koehn and Knowles, 2017; Stahlberg and Byrne, 2019; Kumar and Sarawagi, 2019). Undergenerations are sometimes considered as hallucinations (Lee et al., 2018) but we do not consider them so in our work.

Mistranslation of named entities. Appropriately translating named entities (e.g. names, dates, etc.) is also a known difficulty of NMT systems (Ugawa et al., 2018; Li et al., 2021; Hu et al., 2022). Note that for production systems, this error is rather critical. However, we do not consider it as an hallucination as it does not show detachment from the source but rather an incorrect attempt to translate part of its content.

Other errors. These are other incorrect translations that do not fit the categories above. They may include errors related to part of speech, word order, and others. For an extensive analysis on machine translation errors, refer to Vilar et al. (2006).

3 Hallucination Detection Methods

Approaches to hallucination detection generally aim to find low-quality translations that may also satisfy additional constraints. Previous work either relied only on quality filtering (Lee et al., 2018; Raunak et al., 2021; Müller and Sennrich, 2021), or only on heuristics (Berard et al., 2019), or a combination of the two (Raunak et al., 2021). We stick to this general form and consider different quality filters and heuristics.

3.1 Quality Filters

Quality filters come in two forms: reference-free and reference-based filters. The latter rely on reference translations, while the former do not.

Reference-free methods. We use the state-of-the-art COMET-QE (Rei et al., 2020b) for its superior performance compared to other metrics (Mathur et al., 2020; Freitag et al., 2021; Kocmi et al., 2021).

Reference-based methods. Previous work used adjusted BLEU or CHRF2 scores of less than 1% as a standalone criteria (Lee et al., 2018; Raunak et al., 2021; Müller and Sennrich, 2021; Yan et al., 2022). In this work, we analyse CHRF2 because it is more suitable for sentence-level evaluation. In addition to this lexical metric, we also consider neural COMET (Rei et al., 2020a), a state-of-the-art reference-based metric (Kocmi et al., 2021).

Note that in real-world on-the-fly applications, detecting hallucinations is needed when references are not available. Thus, we use reference-based methods to estimate an upper bound for performance of the other methods.

3.2 Hallucination Detection Heuristics

3.2.1 Previously Used Heuristics

Binary-score Heuristics. These heuristics were used by Raunak et al. (2021) to detect oscillatory and fully detached hallucinations. Given a corpus of source-translation pairs, a translation is flagged as an hallucination if it is in the set of 1% lowest-quality translations, and if:

- **Top n-gram count (TNG).** The count of the top repeated n-gram in the translation is greater than the count of the top repeated source n-gram by at least t (in their work, $n = 4$ and $t = 2$);
- **Repeated targets (RT).** The translation is repeated for multiple unique source sentences.

Anomalous decoder-encoder attention. Attention patterns[2] in which most attention mass is concentrated on the source `EOS` token are often associated with a model ignoring the source and generating a hallucinatory translation (Lee et al., 2018; Berard et al., 2019; Raunak et al., 2021). We consider two different criteria targeted to find this pattern:

- **Attn-to-EOS**: the proportion of attention paid to the `EOS` source token;
- **Attn-ign-SRC**: the proportion of source words with a total incoming attention mass lower than 0.2. This was used as a data filtering criterion in Berard et al. (2019).

3.2.2 Uncertainty-Based Heuristics

Now we describe the uncertainty measures we propose to use as hallucination detectors. Previously, these were used to improve quality assessments (Fomicheva et al., 2020; Zerva et al., 2021).

---

[2] These patterns are respective to the average of the cross-attention heads of the decoder's last layer.

Sequence log-probability (Seq-Logprob). For a trained model $P(y|x,\theta)$ and a generated translation y, Seq-Logprob (i.e., model confidence) is the *length-normalised* sequence log-probability:

$$\frac{1}{L}\sum_{k=1}^{L}\log P(y_k \mid y_{<k}, x, \theta). \quad (1)$$

We hypothesise that when hallucinating, a model is not confident.

Dissimilarity of MC hypotheses (MC-DSim). This method measures how the original hypothesis y disagrees with hypotheses $\{h_1, \dots, h_N\}$ generated in stochastic passes. For the same source sentence, we generate these new hypotheses using Monte Carlo (MC) Dropout (Gal and Ghahramani, 2016). Then we evaluate the average similarity:

$$\frac{1}{N}\sum_{i=1}^{N}\textsc{sim}(h_i, y). \quad (2)$$

Different similarity measures can be used in place of SIM (e.g. METEOR (Banerjee and Lavie, 2005), BERTScore (Zhang et al., 2020), etc.). We follow previous work and use METEOR with $N = 10$ (Fomicheva et al., 2020; Zerva et al., 2021).

3.3 Trained Hallucination Detection Model

An exception from the general framework of hallucination detection is the work by Zhou et al. (2021) who *learn* to detect token-level hallucinations. Specifically, the authors create synthetic data where they randomly corrupt some tokens in a translation and reconstruct them with the BART model (Lewis et al., 2020). Then, the authors fine-tune a pretrained language model to identify the replaced tokens.

TokHal-Model. We evaluate the proportion of tokens that are predicted to be hallucinated and use this as a detection score.

3.4 Binary vs Continuous Scores

The methods above fall into two categories: *binary-score* and *continuous-score* heuristics. The former (only TNG and RT) output a value in $\{0, 1\}$, whereas the latter output a value in $\mathbb{R}$ and the prediction is made depending on a chosen threshold.

4 Experimental Setting

Model. We use Transformer base (Vaswani et al., 2017) from `fairseq` (Ott et al., 2019).

Data. We use the WMT2018 DE-EN news translation data excluding Paracrawl (Bojar et al., 2018) – 5.8M sentence pairs. We randomly choose 2/3 of the dataset for training and use the remaining 1/3 as a held-out set for analysis. For validation, we use the *newstest2017* dataset.

Held-out Data Filtering. We are mainly interested in hallucinations produced for clean data. Since our held-out data comes from the WMT2018 training dataset and thus can be noisy, we filter it using Bicleaner, the filtering tool used in official releases of filtered ParaCrawl data (Sánchez-Cartagena et al., 2018; Ramírez-Sánchez et al., 2020; Kreutzer et al., 2022). Following previous work, we exclude examples with a score below 0.5 and end up with about 1.3M examples.

All details on preprocessing, hyperparameters and implementation can be found in Appendix A.

5 Hallucinations Dataset

To analyse the effectiveness of hallucination detection criteria, we pick a subset of examples that are likely to be hallucinations, and obtain fine-grained annotations from professional translators.

5.1 Data for Annotation

Our data selection is motivated by two goals: (i) find as many hallucinatory translations as possible – to analyse hallucinations, (ii) pick some translations from a long tail of hallucination detection predictions – to analyse the behaviour of these detection methods. Thus, we first pick 250 worst-scored samples for each heuristic and quality filter (including binary assignments obtained through TNG and RT). Next, we turn to a broader set of samples and consider translations whose scores fall below a chosen percentile for a given method, i.e. we consider long tails of the scores.[3] For in-domain settings, previous work reported hallucinatory rates of $0.2-2\%$. However, these rates were either obtained on noisy and/or low-resource data or using weaker models. Therefore, in our cleaner in-domain setting with a stronger model, we expect the hallucination rate to lie in the lower end of the indicated range. Thus, we consider approximately 0.4% of the worst scores (which amounts to 5000 flagged translations for each criteria).[4] From

---

[3] From now on, we refer to examples contained in a long tail of a method as “flagged” or “detected” by this method.

[4] The threshold for prediction is consistent with this percentile. For practical details, refer to Appendix G.

these, we sample 250 examples and add them to the dataset. In total, we end up with 3415 examples for annotation.[5]

5.2 Guidelines and Annotation

The annotation guidelines are developed according to the taxonomy defined in Section 2. All details on data collection can be found in Appendix B.

6 High-level Overview of the Dataset

6.1 General Statistics

Figure 1 gives a structured overview of dataset statistics. First, we see that while we picked translations that are likely to be pathological, 60% of the dataset consists of correct translations. This highlights that with the existing methods, finding poor translations reliably is still challenging. Next, note that most of the incorrect translations have translation errors that are not severe enough to be deemed hallucinations. This agrees with the view that hallucinations lie at the extreme end of the spectrum of MT pathologies (Raunak et al., 2021). Finally, the results of the annotation confirm that our data selection is very reasonable. Indeed, while previous work has reported hallucinatory rates of $0.2 - 2\%$ in in-domain settings, we see that, for a reasonably numbered collection of examples, our hallucination rate is substantially higher (9%) – 294 hallucinations among the 3415 translations. In Figure 1, we also show the method-specific statistics of human annotation results for each heuristic and quality filter. Unsurprisingly, the long tails of each method display different characteristics. For example, almost all translations flagged by Attn-to-EOS are correct, whereas the proportion of good translations flagged by COMET-QE or Seq-Logprob is rather small. We will analyse this further in Section 7.

6.2 MT Errors vs Hallucinations

Now let us look separately at the sets of examples with hallucinations and other translation errors. Figure 2 shows the structures of these sets with respect to interactions of the different criteria.

We see that it is reasonable to consider hallucinations separately from other errors: patterns in Figures 2a and 2b are substantially different. For example, for translation errors, COMET-QE performs well, being on par with reference-aware COMET (Figure 2b). However, for hallucinations, it identifies less than half the amount of examples identified not only by COMET, but even by simple uncertainty-based heuristics (Figure 2a). Furthermore, Figure 2 reveals a significant difference between interactions of different criteria for hallucinations and for other translations: hallucinations in our data are most often flagged by multiple criteria simultaneously (e.g. Seq-Logprob flags the majority of hallucinations detected with Attn-ign-SRC and MC-DSim), whereas most MT errors and correct translations are flagged by a single method. This difference in patterns supports our choice of taxonomy: properties of hallucinations are very different from those of other less severe errors.

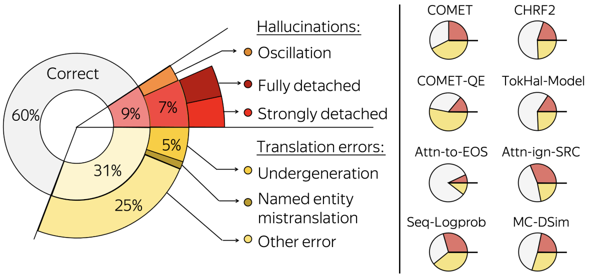


Figure 1: Overall (left) and method-specific (right) statistics of human annotation results. Method-specific statistics show the percentages of correct translations (grey), translation errors (yellow) and hallucinations (red) among the examples flagged by each method.

7 Analysing Detection Criteria

In this section, we provide a comprehensive analysis of the performance of the heuristics and quality filters introduced in Section 3.

7.1 Quality Filters

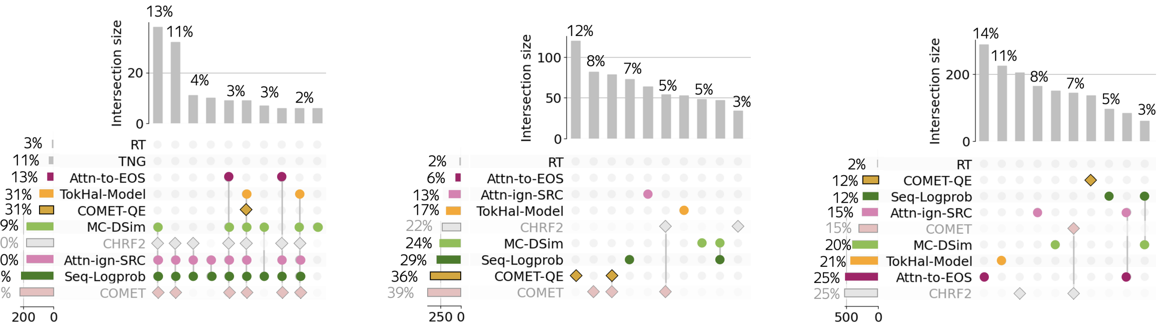
Here we start by analysing reference-based methods, namely COMET and CHRF2, and then turn to the reference-free COMET-QE. Overall, our results show that reference information is helpful, while COMET-QE fails to penalise hallucinations.

Reference information helps detection. Figure 2a shows that leveraging reference information helps detecting hallucinations: COMET detects more hallucinations than any of the other methods. As expected, lexical-based CHRF2 is significantly worse than neural-based COMET. In fact, it lags behind several heuristics (e.g., Seq-Logprob).

We also explore whether previously proposed methods for detecting hallucinations under domain

---

[5] Note that a sample originally obtained from the worst scores or sampled from the long tail of a given method may belong to the long tail of another method.



(a) Hallucinations (b) MT errors (c) Correct translations

Figure 2: Structure of the sets of translations flagged by the considered methods. Horizontal bars show the proportion of examples flagged by each method among all translations of the considered category. Each vertical bar shows the size for the set of translations that are (i) flagged by all the methods marked in the corresponding column and (ii) not flagged by any of the rest; only the top intersections are shown. Quality filters are shown with diamond marks, and detection heuristics – with circles. Methods requiring reference translations are shaded.

Heuristic	**Correct**	**MT Errors**	**Hallucinations**		
			OSC	SD	FD
TNG	0	0	32	0	0
RT	18	19	2	1	7
All dataset	2048	1073	86	90	118

Table 2: Translations flagged by TNG and RT.

shift are appropriate in our cleaner in-domain setting. Specifically, we follow Müller and Sennrich (2021) and consider translations with CHRF2 score lower than 1%. Strikingly, for our clean setting, this approach is inadequate: in the entire held-out set of 1.3M examples, it flagged only 2 translations. This suggests that methods suitable for noisy settings (e.g., domain shift) might not be applicable in settings where models are less likely to hallucinate.

COMET-QE fails to penalise hallucinations. From Figure 1 (right) we see that, as expected, most of the translations flagged by COMET-QE are incorrect. However, the vast majority of them are not hallucinations. Indeed, Figure 2a shows that COMET-QE is one of the worst hallucination detectors, meaning that it fails to rank by the severity of a translation pathology. This supports the hypothesis made in previous work: since quality estimation models are mostly trained on data that lacks negative examples, they may be inadequate for evaluating poor translations (Takahashi et al., 2021; Sudoh et al., 2021).

Overall, among the considered quality filters, only COMET may be used as a hallucination detector. However, it is important to keep in mind that, in on-the-fly applications, detecting hallucinations is needed when references are not available, rendering reference-based methods not applicable.

7.2 Detection Heuristics

The results in Section 7.1 leave a relevant gap: where do we turn to when references are not available? Is there any information, besides quality, that may help detecting hallucinations? Our evidence suggests that in preventive settings, previously proposed heuristics are mostly inadequate, and uncertainty may be the answer for the questions we pose.

Binary-score heuristics perform the worst. Table 2 shows the number of detected translations for Top n-gram count (TNG) and Repeated targets (RT). These heuristics are targeted to identify oscillatory and fully detached hallucinations, respectively. We see that while TNG obtains perfect precision, it fails to identify more than half of the oscillatory translations. RT, in turn, performs poorly across the board: only a few hallucinations are detected, and a significant proportion of flagged translations turn out to be correct. Moreover, Figure 2 shows that even if we join sets of translations detected by TNG and RT (as done in Raunak et al. (2021)), altogether we get *fewer hallucinations than almost any other considered method*. Thus, in preventive settings, these methods are highly inadequate.

Anomalous attention is mostly *not* hallucination. Figures 1 and 2 show that behaviors of Attn-to-EOS and Attn-ign-SRC are significantly different. First, Attn-to-EOS *is not indicative of hallucinations*. Indeed, attention patterns in which most attention mass is concentrated on the EOS token largely correspond to correct translations. On the other hand, Attn-ign-SRC performs well and is second only to uncertainty-based Seq-Logprob. Such a difference in performance is surprising: both methods are motivated by a common belief that if almost all the attention mass is concentrated on the source EOS token, a translation is likely to be a hallucination (Berard et al., 2019; Raunak et al., 2021). In fact, both methods were designed to identify this specific pattern. However, patterns identified with Attn-ign-SRC span from attention mass coming to various uninformative tokens (e.g., punctuation) to examples where attention is mostly diagonal (typically, these correspond to undergenerations). We show examples of such attention maps in Appendix C. Overall, the results highlight a disparity between what is *believed to indicate* hallucinations and what *actually indicates* them.

Note that while Attn-ign-SRC performs relatively well, it should be used with caution: attention-based heuristics rely on the assumption that attention patterns reflect model reasoning. This assumption is not reliable: although there is evidence that attention can play recognizable roles (Voita et al., 2018, 2019), a lot of work questions attention explainability (Wiegreffe and Pinter, 2019; Jain and Wallace, 2019; Serrano and Smith, 2019; Bastings and Filippova, 2020; Pruthi et al., 2020). Since hallucinations identified by Attn-ign-SRC are overwhelmingly contained among the ones identified by Seq-Logprob (Figure 2a), we recommend using the latter instead.

TokHal-Model is unfit for natural hallucinations. Let us recall that the model used for the TokHal-Model scores was trained to identify replaced tokens in corrupted translations that are fluent and do not differ much from the original ones (Zhou et al., 2021). This means that during training, the model was unlikely to observe highly pathological translations that reflect the types of hallucinations produced by actual NMT systems. This raises several concerns when using this model in our setting. For example, it might incorrectly flag adequate tokens such as synonyms or paraphrases. What is more, since severely flawed examples are mostly out of distribution for the model, labels predicted for such translations may be unreasonable.

The results confirm our concerns: Figures 1 and 2a show that (i) the vast majority of translations flagged by TokHal-Model are correct and (ii) it is one of the worst hallucination detectors.

Model confidence may be all you need. Figure 2 shows that Seq-Logprob is *the best heuristic* and performs on par with reference-based COMET. This means that the less confident the model is, the more likely it is to generate an inadequate translation. This agrees with some observations made in previous work on quality estimation (Fomicheva et al., 2020). Interestingly, such performance of the method contrasts with its simplicity: Seq-Logprob scores are easily obtained as a by-product of generating a translation. This distinguishes the method from all the rest that require additional computation (e.g., corpus-level search for RT or generating multiple hypotheses for MC-DSim).

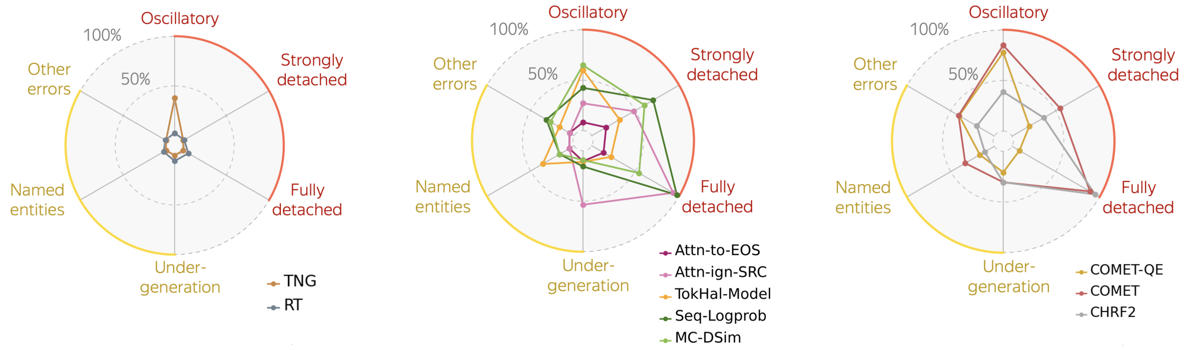
On another note, NMT models have been found to be miscalibrated (Kumar and Sarawagi, 2019). Thus, investigating the impact of recalibration methods – aimed at enhancing prediction reliability – on the detection quality of Seq-Logprob constitutes an interesting avenue for future research.

MC-DSim: hallucinations are mostly unstable. The intuition behind this heuristic is simple: when faced with a source sentence for which a good translation is not immediate, the set of hypotheses the model "keeps at hand" may be very diverse. Indeed, MC-DSim performs relatively well and identifies a good proportion of hallucinations (Figures 1, 2). Thus, most hallucinations are *unstable*: dissimilarity of MC hypotheses helps to identify them.

Note that while MC-DSim is not the best choice for detection, later we will show that the intuition behind this method is very helpful for alleviating hallucinations at test time (Section 9).

7.3 Combining Heuristics and Quality Filters

Intersecting a set of translations obtained via a heuristic with the bottom scored translations according to a quality filter was introduced in Raunak et al. (2021). The motivation for this was to avoid incorrectly flagging good translations as hallucinations (e.g., without this intersection, RT flags good-quality paraphrases). Intuitively, this idea is very reasonable: hallucinations are indeed incorrect translations. However, implementing this in practice requires a good quality estimation model, specifically for ranking poor translations. Unfortunately, we showed in Section 7.1 that for this purpose, even the state-of-the-art COMET-QE is largely inadequate. This means that using such quality estimates may lead to filtering out a lot of hallucinations, which is not desirable in preven-



(a) Binary-score heuristics (b) Continuous-score heuristics (c) Quality filters

Figure 3: Distribution of the translations flagged by each method conditioned on each of the considered pathologies.

tive settings. Our results show that this is exactly the case: e.g., filtering with COMET-QE leads to losing nearly 80% of hallucinations detected by Seq-Logprob. All in all, such an intersection generally does more harm than good.

8 Analysing Hallucination Pathologies

In this section, we look at hallucination pathologies in isolation and show that the behavior of detection methods varies depending on the type of pathology. For example, for a given pathology, some methods may be specialised, whereas others may fail. For a similar analysis on other less severe translation errors, refer to Appendix D.

Fully detached hallucinations. Figure 3 shows that these hallucinations are easily detected by several methods, e.g. Seq-Lobprob, Attn-ign-SRC, COMET, CHRF2. This is not surprising: intuitively, the most severe pathology should be the easiest to detect. However, COMET-QE *fails to identify almost all these hallucinations*. While COMET-based metrics are known to not penalise enough certain types of errors (e.g., discrepancies in numbers and named entities; see Amrhein and Sennrich (2022); Raunak et al. (2022)), such poor performance for completely inadequate translations is highly unexpected. This calls for further research on the behavior of quality estimation models.

On a more general note, previous work suggested that fully detached hallucinations emerge as exact copies of references from training data (Raunak et al., 2021). We validate this hypothesis and find the contrary: out of the 44 unique translations marked as fully detached from the source, only 4 are exact copies of references in the training data. Nevertheless, when looking at these sentences more closely, we see that they do contain large substrings that are seen frequently during training. Therefore, fully detached hallucinations are indeed likely to be traced back to the training data, but they *emerge in non trivial ways and not necessarily as exact copies*. This can be seen as one more evidence that, when dealing with memorisation in language models, it is necessary to consider not just full copies in the training data but also near-duplicates (Lee et al., 2022).

Strongly detached hallucinations. As expected, this pathology is harder to detect than fully detached hallucinations (Figure 3). However, the trends are largely similar: for example, COMET-QE fails again, and Seq-Logprob performs best and outperforms even reference-based COMET.

Oscillatory hallucinations. The method specifically developed to detect this hallucination type (Top n-gram count, TNG) performs worse than most of the other methods. Among the rest, COMET performs best. Interestingly, in contrast to previous observations, COMET-QE performs well, being on par with COMET.

9 DeHallucinator: Overwriting Hallucinations at Test Time

In previous sections, we saw that hallucinations are more unstable than other translations: for them, generated MC-dropout hypotheses tend to vary greatly. This motivates us to look more closely into these hypotheses: are any of these translations *not* hallucinations? Answering this question not only gives insight into the inner workings of hallucinating NMT models, but also leads to an interesting practical application – overwriting hallucinations at inference time. This is of utmost importance for production systems where hallucinations have a deeply compromising effect on user trust.

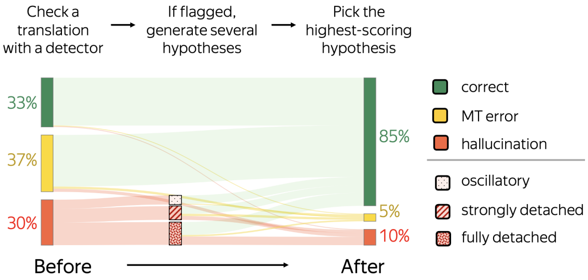


Figure 4: Our pipeline scheme along with results.

Whenever flagged, overwrite with better. Intuitively, our idea is similar to hybrid pipelines when a machine-generated translation is first passed to a quality estimation system and then, if needed, is corrected by human translators. In our case, we first apply a hallucination detector and then, if a translation is flagged, we try to overwrite it with a better translation (Figure 4). For this, we generate several MC-dropout hypotheses, score them with some measure, and pick the highest-scoring translation as a final candidate (in the spirit of reranking approaches (Shen et al., 2004; Lee et al., 2021; Fernandes et al., 2022; Freitag et al., 2022)).

The general pipeline above relies on the choice of a hallucination detector and a scoring measure. For the detector, we use the best of the analysed detectors, i.e. Seq-Logprob.[6] For the scoring measure, a natural choice would be a quality estimation system: by construction, these systems are designed to score translations according to quality. However, as we saw earlier, even the state-of-the-art COMET-QE may fail (Section 7.1). Therefore, we compare two measures: COMET-QE and Seq-Logprob.

In this experiment, we randomly choose 200 translations from our dataset flagged by Seq-Logprob. For each, we generate 10 hypotheses with MC-dropout. Then, for the overwritten translations we gather annotations according to our guidelines (Section 5). The results are summarized in Figure 4. Although we were concerned about COMET-QE because of its low performance when ranking poor translations, we find that for choosing the best hypothesis, it is indeed appropriate and performs better than Seq-Logprob. We thus show results with COMET-QE scores in the main text and with Seq-Logprob in Appendix E.

Most hallucinations and errors become correct. Figure 4 shows that most hallucinations are overwritten with correct translations. This is surprising: in most cases, the model is not stuck in a hallucinatory mode and can generate good translations in a small vicinity of model parameters. In this sense, most hallucinations result from "bad luck" during generation and not profound model defect. Note that fully detached hallucinations are the hardest to improve. This makes sense as these are likely to be traced back to (near-)duplicates in the training data and, therefore, they do highlight model anomalies.

Note that in this pipeline, we overwrite not only hallucinations but also other errors and correct translations that were flagged by the detector. This means that our method needs to appropriately handle such translations. From Figure 4 we see a pleasant side-effect: our approach overwrites most of the errors with correct translations. Just as importantly, almost all originally correct translations remain correct. Overall, the proportion of correct translations among the ones flagged by the detector increases from 33% to 85%, and the hallucinatory rate decreases threefold. In Appendix F, we show several examples of overwritten hallucinations.

10 Conclusions

Dealing with hallucinations is difficult. First, we had to take a step back from previous work and refuse procedures that amplify the problem, as these hinder the behavior of models in their standard settings. After that, we notice that work on detection often relies on assumptions that remained unquestioned (e.g., generic quality measures, targeted heuristics, anomalous attention being suitable detectors). Through extensive experiments, we establish order in detection methods. Surprisingly, despite introduction of several methods specifically targeted for hallucinations, what works best has always been at our disposal: standard sequence log-probability. This suggests that characteristics innate to a model can have a lot of value. In fact, such characteristics are the backbone of our DEHALLUCINATOR, a lightweight approach that significantly alleviates hallucinations at test time. This leaves space for future research on model uncertainty, hallucination prevention, understanding where hallucinations come from, among others. For this, we release our corpus with structured annotations along with the model and its training data. Altogether, this allows us to say that we provide solid ground for future study of hallucinations in NMT.

---

[6] In this experiment, we take the translations from our dataset and consider the percentiles defined in Section 5.

Limitations

We highlight three main limitations in our work. First, although the foundation of our proposed taxonomy for hallucinations rests on the idea of detachment from the source content, we do not evaluate it quantitatively. Indeed, we cannot point whether the model that generated the hallucinations was indeed detached from the source sentence when generating them. Nevertheless, we can guarantee that the hallucinations in our dataset are translations that are detached from the source content according to professional translators. We consider that the quantitative analysis of the detachment to be out of the scope of this paper. Still, it constitutes an interesting line for future research on understanding hallucinations in NMT that may be facilitated with the release of our code, model and annotated dataset.

Second, while this paper comprehensively studies the phenomena of hallucinations in NMT for a high-resource language pair, experiments in more language pairs (including low-resource languages) are necessary to assess the broad validity of our claims. To keep our setup familiar to researchers and practicioners, we opted for a familiar language pair for which data is widely available. Moreover our choice also facilitated the data collection process as there is a large supply of professional translators for this language pair.

Lastly, instead of focusing on more recent NMT models that use large pretrained language models as their backbone, we focused on a Transformer base model. The reason for this choice is that we wanted to keep the setup simple, familiar, easy to reproduce, and computationally economical. Moreover, it was important for our work to have full control on the training and held-out data. Nevertheless, research on hallucinations on more recent and powerful NMT models is an exciting line of future work and we hope our work spurs that research.

Acknowledgments

This work is partially supported by the European Research Council (ERC StG DeepSPIN 758969, by the FCT through contract UIDB/50008/2020, by EU's Horizon Europe (UTTER, HORIZON-CL4-2021-HUMAN-01-13, contract 101070631), and by the P2020 programs MAIA and Unbabel4EU (LISBOA-01-0247-FEDER-045909 and LISBOA01-0247-FEDER-042671). Lena is supported by the Facebook PhD Fellowship.

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online. Association for Computational Linguistics.

Chantal Amrhein and Rico Sennrich. 2022. Identifying weaknesses in machine translation metrics through minimum bayes risk decoding: A case study for comet.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155, Online. Association for Computational Linguistics.

Alexandre Berard, Ioan Calapodescu, and Claude Roux. 2019. Naver labs Europe's systems for the WMT19 machine translation robustness task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 526–532, Florence, Italy. Association for Computational Linguistics.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco

Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Patrick Fernandes, António Farinhas, Ricardo Rei, José De Souza, Perez Ogayo, Graham Neubig, and Andre Martins. 2022. Quality-aware decoding for neural machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1396–1412, Seattle, United States. Association for Computational Linguistics.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Markus Freitag, David Grangier, Qijun Tan, and Bowen Liang. 2022. High Quality Rather than High Model Probability: Minimum Bayes Risk Decoding with Neural Metrics. *Transactions of the Association for Computational Linguistics*, 10:811–825.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, George Foster, Alon Lavie, and Ondřej Bojar. 2021. Results of the WMT21 metrics shared task: Evaluating metrics with expert-based human evaluations on TED and news domain. In *Proceedings of the Sixth Conference on Machine Translation*, pages 733–774, Online. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA. PMLR.

Junjie Hu, Hiroaki Hayashi, Kyunghyun Cho, and Graham Neubig. 2022. DEEP: DEnoising entity pre-training for neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1753–1766, Dublin, Ireland. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, Iroro Orife, Kelechi Ogueji, Andre Niyongabo Rubungo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, Tapiwanashe Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. 2022. Quality at a Glance: An Audit of Web-Crawled Multilingual Datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Aviral Kumar and Sunita Sarawagi. 2019. Calibration of encoder decoder models for neural machine translation. *CoRR*, abs/1903.00802.

Ann Lee, Michael Auli, and Marc'Aurelio Ranzato. 2021. Discriminative reranking for neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7250–7264, Online. Association for Computational Linguistics.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2018. Hallucinations in neural machine translation.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 8424–8445, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Panpan Li, Mengxiang Wang, and Jian Wang. 2021. Named entity translation method based on machine translation lexicon. *Neural Comput. Appl.*, 33(9):3977–3985.

Arle Lommel, Aljoscha Burchardt, and Hans Uszkoreit. 2014. Multidimensional quality metrics (mqm): A framework for declaring and describing translation quality metrics. *Tradumàtica: tecnologies de la traducció*, 0:455–463.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020. Results of the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 688–725, Online. Association for Computational Linguistics.

Mathias Müller, Annette Rios, and Rico Sennrich. 2020. Domain robustness in neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 151–164, Virtual. Association for Machine Translation in the Americas.

Mathias Müller and Rico Sennrich. 2021. Understanding the properties of minimum Bayes risk decoding in neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 259–272, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online. Association for Computational Linguistics.

Gema Ramírez-Sánchez, Jaume Zaragoza-Bernabeu, Marta Bañón, and Sergio Ortiz-Rojas. 2020. Bifixer and bicleaner: two open-source tools to clean your parallel data. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 291–298, Lisboa, Portugal. European Association for Machine Translation.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online. Association for Computational Linguistics.

Vikas Raunak, Matt Post, and Arul Menezes. 2022. Salted: A framework for salient long-tail translation error detection.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020a. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020b. Unbabel's participation in the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 911–920, Online. Association for Computational Linguistics.

Víctor M. Sánchez-Cartagena, Marta Bañón, Sergio Ortiz-Rojas, and Gema Ramírez-Sánchez. 2018. Prompsit's submission to wmt 2018 parallel corpus filtering shared task. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter* of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 177–184, Boston, Massachusetts, USA. Association for Computational Linguistics.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Katsuhito Sudoh, Kosuke Takahashi, and Satoshi Nakamura. 2021. Is this translation error critical?: Classification-based human and automatic machine translation evaluation focusing on critical errors. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 46–55, Online. Association for Computational Linguistics.

Kosuke Takahashi, Yoichi Ishibashi, Katsuhito Sudoh, and Satoshi Nakamura. 2021. Multilingual machine translation evaluation metrics fine-tuned on pseudo-negative examples for wmt 2021 metrics task. In *Proceedings of the Sixth Conference on Machine Translation*, pages 1049–1052, Online. Association for Computational Linguistics.

Chau Tran, Shruti Bhosale, James Cross, Philipp Koehn, Sergey Edunov, and Angela Fan. 2021. Facebook AI's WMT21 news translation task submission. In *Proceedings of the Sixth Conference on Machine Translation*, pages 205–215, Online. Association for Computational Linguistics.

Arata Ugawa, Akihiro Tamura, Takashi Ninomiya, Hiroya Takamura, and Manabu Okumura. 2018. Neural machine translation incorporating named entity. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3240–3250, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across nlp tasks.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

David Vilar, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. 2006. Error analysis of statistical machine translation output. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Analyzing the source and target contributions to predictions in neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1126–1140, Online. Association for Computational Linguistics.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Chaojun Wang and Rico Sennrich. 2020. On exposure bias, hallucination and domain shift in neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3544–3552, Online. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Jianhao Yan, Fandong Meng, and Jie Zhou. 2022. Probing causes of hallucinations in neural machine translations.

Chrysoula Zerva, Daan van Stigt, Ricardo Rei, Ana C Farinha, Pedro Ramos, José G. C. de Souza, Taisiya Glushkova, Miguel Vera, Fabio Kepler, and André F. T. Martins. 2021. IST-unbabel 2021 submission for the quality estimation shared task. In *Proceedings of the Sixth Conference on Machine Translation*, pages 961–972, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting hallucinated content in conditional neural sequence generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, Online. Association for Computational Linguistics.

A Experimental setup

Data preprocessing. We filter our data using language identification and simple length-heuristics described in Tran et al. (2021). We encode the data with byte-pair encoding (Sennrich et al., 2016) using the SentencePiece framework (Kudo and Richardson, 2018). We set the vocabulary size to 32k and compute joint encodings and vocabulary.

Model parameters. We follow the setup of Transformer base model (Vaswani et al., 2017) (hidden size of 512, feedforward size of 2048, 6 encoder and 6 decoder layers, 8 attention heads). The model has approximately 77M parameters.

Optimizer. Similarly to (Vashishth et al., 2019), we train our model using the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.98$ and use an inverse square root learning rate scheduler with initial value 5×10^{-4}, and a linear warm-up in the first 4000 steps.

Training and Inference. Models are trained for 250K updates with a batch size of about 32K tokens. We set dropout to 0.3. At inference time, we produce translations using beam search with a beam of 5. We validate our models during training using SacreBLEU (Post, 2018), and we choose the checkpoint based on best BLEU in validation. We provide BLEU[7] and COMET baselines on WMT evaluation campaigns in Table 3. We train and performance inference on top of the Fairseq framework (Ott et al., 2019).

COMET versions. We use models available in the official repository[8]: `wmt20-comet-da` for COMET and `wmt20-comet-qe-da-v2` for COMET-QE.

TokHal-Model. We follow the official implementation.[9] For the synthetic data generation step, we used BART-large; and, for the token-level hallucination predictor, we used XLM-R (Conneau et al., 2019).

Computing Infrastructure. All our experiments have been ran on a machine with 2 physical Intel(R) Xeon(R) Gold 6348 @ 2.60GHz CPUs (total of 112 threads), and 4 NVIDIA RTX A6000 GPUs. In particular, the NMT model described above was trained in less than 30 hours on a single GPU.

Metric	WMT2014	WMT2017	WMT2018
BLEU	31.1	32.6	38.9
COMET	0.3178	0.3257	0.3340

Table 3: Evaluation metrics for EN → DE for *newstest* sets for the WMT 2014, 2017 and 2018 campaigns.

B Data Collection

We perform a rigorous and comprehensive manual annotation procedure with professional translators, in order to make sure that we are reliably analysing hallucinated translations.[10] First, the translators were asked to be familiar with our task by reading our provided annotation examples, along with detailed annotation instructions. Then they had to pass a test to show they can recognize different pathologies in different translations. We selected the two best translators to gather the annotations. After being hired, we ran three one-hour tutorial sessions to explain the task thoroughly and to clarify any possible questions. During the annotation process, we made sure to be promptly available to answer any question from the translators. We paid a fair wage (25-30 USD per hour) – well above both the US federal minimum and the average EU minimum wage – and inspected their work for quality.

Guidelines. We make available the full guidelines used by the translators along with all other resources in the project repository. In short, the annotators were presented with a source sentence and a model-generated hypothesis and asked to deem that translation as correct (`COR`) or incorrect. If incorrect, they were prompted to answer a series of yes/no questions, regarding the presence of specific hallucinatory pathologies: oscillations (`OSC`), strong detachment (`SD`) and full detachment (`FD`). We also asked annotators to flag critical errors such as named-entity mistranslations (`NE`) and under-generated translations (`UG`).

Inter-annotator agreement. To determine the reliability of our annotations, we asked both our translators to annotate a set of 400 randomly sampled translations. For all hallucinatory categories but `SD`, the annotators achieved – according to Cohen's kappa coefficient (Cohen, 1960) – almost

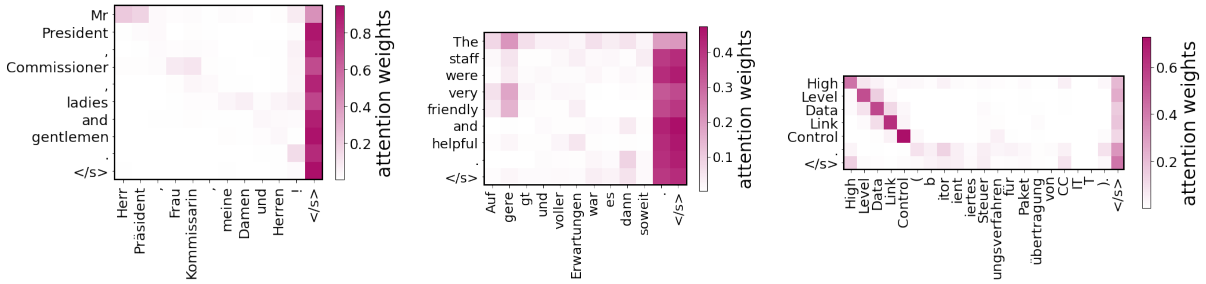
---

[7] BLEU+case.mixed+lang.XX+numrefs.1+smooth.exp+tok.13a+version.1.4.2

[8] https://github.com/Unbabel/COMET

[9] https://github.com/violet-zct/fairseq-detect-hallucination

[10] Our translators were hired through Upwork and they were informed about the academic purposes of the data annotation process. All translators hired for this study reside in Europe.



(a) Correct translation, flagged by Attn-to-EOS

(b) Hallucination, flagged by Attn-ign-SRC

(c) Undergeneration, flagged by Attn-ign-SRC

Figure 5: Examples of attention maps flagged by attention-based heuristics.

Fleiss's Kappa Scores					
COR	UG	NE	OSC	SD	FD
0.62	0.73	0.42	0.86	0.45	0.89

Table 4: Fleiss's Kappa inter-annotator agreement scores (↑) for the different categories translators were prompted to identify.

perfect agreement. For all other categories, moderate to substantial agreement was obtained. This confirms that our data conforms very well to our instructions. The agreement scores for each category are displayed in Table 4.

C Patterns of attention maps for translations flagged with attention-based heuristics

The attention maps are shown in Figure 5. While both Attn-to-EOS and Attn-ign-SRC were designed to identify translations for which almost all the attention mass is concentrated on the EOS token, the patterns identified with Attn-ign-SRC are more diverse. They span from attention mass coming to various uninformative tokens (e.g., punctuation and other tokens as in Figure 5b) to examples shown in Figure 5c where attention is mostly diagonal (typically, these correspond to undergenerations; we follow this discussion in the next section).

D Analysing Less Severe Translation Errors

We notice that some of the detection methods are specialised on specific pathologies. For example, Attn-ign-SRC is by far the best for detecting undegenerations. This is expected: an undergeneration does not cover part of the source sentence, thus a significant proportion of source tokens receives little attention mass (see example in Figure 5c). For named entity errors, the best heuristic is TokHal-Model. This mirrors the discussion in Section 7.2: while severe errors (i.e., hallucinations) fall out of distribution for this model, mistranslations of short phrases are more in line with the model's training.

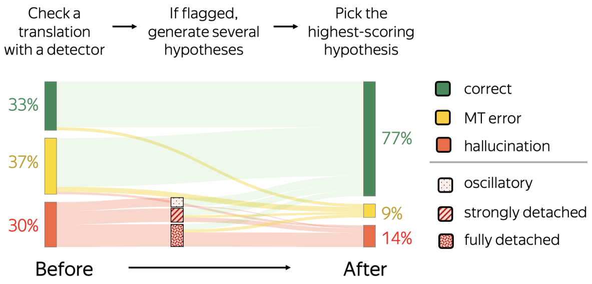


Figure 6: Our pipeline scheme along with results when we use Seq-Logprob as both the detector and the scoring measure.

On a different note, MC-DSim performs much better for hallucinations than for less severe pathologies (Figure 3b). This again points to hallucinations being more unstable than other errors.

E Overwriting Hallucinations with Seq-Logprob as the scoring measure

The results are shown in Figure 6. In order to use Seq-Logprob as the scoring measure, we score all generated hypothesis with the original model. Overall, although the results follow the same trend as those using COMET-QE as the scoring measure, they are slightly worse: the hallucinatory rate is higher and the percentage of correct translations is smaller.

F Examples of overwritten translations

Table 5 shows examples of each hallucination type that have been overwritten with correct translations with the approach described in Section 9.

G Practical Recommendations for Detection

As we have mentioned in Section 3.4, Binary-label heuristics output a value in $\{0; 1\}$, and continuous-score heuristics output a value s in $\mathbb{R}$. These values s can be used to build a binary decision rule: for a given source x, translation $\hat{y}$ and reference y, a translation is flagged by the detector if and only if $s(x, \hat{y}, y) \leq \gamma$. Naturally, s may not need be a function of x, $\hat{y}$ and y (e.g. COMET-QE is only a function of x and $\hat{y}$).

Choosing the thresholds. In our work, we chose the thresholds γ for each detector by assessing the value s correspondent to approximately the 0.4-th percentile to be consistent with the data selection process (see Section 5). We computed these thresholds using the entire filtered held-out data. However, in practice, we obtained very similar results when we obtained these thresholds using a collection of only $10\,000$ examples from that dataset.

We recommend computing these thresholds on in-domain clean data. This will guarantee that the cut-off values were obtained in a scenario where the model performs best. Finally, the definition of the k-th percentile is expected to have an impact on the precision-recall trade-off. Thus, for preventive settings, we recommend sticking to more conservative values of k.

Choosing the detectors. The choice of detector rests upon the application for which it is intended. For high-precision settings, binary-label heuristics such as TNG and variants thereof (Raunak et al., 2022) may be more recommended. For preventive settings, we suggest using Seq-Logprob as the backbone of the hallucination detector. Naturally, we generally obtain higher recall by joining the sets of translations flagged with multiple methods. For example, Figure 3 reveals that joining the set of translations flagged with Seq-Logprob with the set of translations flagged with COMET-QE is very reasonable: COMET-QE performs better for oscillatory hallucinations, while Seq-Logprob is better for the other hallucination types.

Be careful when relying on references. Using reference information is helpful for detecting hallucinations (see Section 7.1), and while it may not be used to detect hallucinatins on-the-fly, it may still prove useful for analysis works. We have found that high-quality parallel data is critical for adequate application of these methods: very low scores might not only be attributed to poor translations, but also to reference mismatches. Indeed, preliminary experiments highlighted this worrying trend, which motivated us to clean the held-out set (Section 4). Thus, if using reference information to detect hallucinations, make sure to thoroughly clean your parallel data.

Overwriting fully detached hallucinations	
Source	Handys, die bis auf Wäschewaschen und Staubsaugen scheinbar alles können.
Reference	Mobile phones that can practically do everything except clean the laundry and vacuum clean.
Original Hypothesis	The staff were very friendly and helpful.
Overwritten Hypothesis	Mobile phones that seem to be able to do everything except on laundry and dustproofing.
Source	In unserem 2 Personen Van mit Dusche/WC war ausreichend Platz für uns beide.
Reference	The space in our 2 person van with shower/toilet was enough for 2 people.
Original Hypothesis	The staff were very friendly and helpful. The room was clean and comfortable.
Overwritten Hypothesis	In our 2 person van with shower/WC there was enough room for us both.

Overwriting strongly detached hallucinations	
Source	Tickets für Busse und die U-Bahn ist zu teuer, vor allem in Stockholm.
Reference	Tickets for buses and the subway is too expensive, especially in Stockholm.
Original Hypothesis	The hotel is located in the centre of Stockholm, close to the train station.
Overwritten Hypothesis	Buses and metro tickets are too expensive, especially in Stockholm.
Source	Ich freue mich über jeden Tag, wo es mir so gut geht!
Reference	I am pleased about each day, where I am so well!
Original Hypothesis	I look forward to seeing you every day!
Overwritten Hypothesis	I'm very happy with every day I'm doing so well!

Overwriting oscillatory hallucinations	
Source	In dieser Zeit stürzt sich Murnau bereits wieder in Theaterproben, allerdings widmet er sich nicht mehr der Schauspielerei, sondern der Regie.
Reference	During this period, Murnau once again dedicates his time to theatre rehearsals; however, this time not as an actor, but as a director.
Original Hypothesis	Murnau was born in Murnau, Germany. He was born in Murnau, Germany.
Overwritten Hypothesis	During this time, Murnau began to appear in theater tests again, but he was no longer concerned with acting, but with directing.
Source	Die Staaten, deren Fangflotten im Nordwestatlantik Hochseefischerei betreiben, bemühen sich im Rahmen der NAFO (Organisation für die Fischerei im Nordwestatlantik) um eine gemeinsame Bestandserhaltung und - bewirtschaftung.
Reference	States which fish in the high seas in the North West Atlantic co-operate in NAFO (North-west Atlantic Fisheries Organisation) in order to ensure conservation and management of stocks.
Original Hypothesis	The North-West Atlantic Fisheries Organisation (NAFO) is a member of the North-West Atlantic Fisheries Organisation (NAFO).
Overwritten Hypothesis	The states whose fishing fleets in the North-West Atlantic are engaged in deep-sea fishing are seeking joint conservation and management within the framework of the North-West Atlantic Fisheries Organisation (NAFO).

Table 5: Examples of hallucinations of each type that have been overwritten with correct translations.

Investigating UD Treebanks via Dataset Difficulty Measures

Artur Kulmizev Joakim Nivre
Department of Linguistics and Philology
Uppsala University
`{artur.kulmizev,joakim.nivre}@lingfil.uu.se`

Abstract

Treebanks annotated with Universal Dependencies (UD) are currently available for over 100 languages and are widely utilized by the community. However, their inherent quality characteristics are hard to measure and are only partially reflected in parser evaluations via accuracy metrics like LAS. In this study, we analyze a large subset of the UD treebanks using three recently proposed accuracy-free dataset analysis methods: dataset cartography, $\mathcal{V}$-information, and minimum description length. Each method provides insights about UD treebanks that would remain undetected if only LAS was considered. Specifically, we identify a number of treebanks that, despite yielding high LAS, contain very little information that is usable by a parser to surpass what can be achieved by simple heuristics. Furthermore, we make note of several treebanks that score consistently low across numerous metrics, indicating a high degree of noise or annotation inconsistency present therein.

1 Introduction

Datasets have long played a crucial role in dictating the pace of progress in NLP. Their function, for most tasks, is largely two-fold: 1) to collect data points (and their corresponding gold-standard labels) on which statistical models can be trained, and 2) to serve as benchmarks though which various models can be evaluated and compared. In recent years, much research has been devoted towards developing new datasets, tasks, and benchmarks for NLP — so as to articulate the distinguishing aspects of a bevy of new neural models. Syntactic parsing has remained an active area of research in this regard, and Universal Dependencies (UD) (Nivre et al., 2016, 2020; de Marneffe et al., 2021) has emerged as a crucial initiative within NLP, offering a set of cross-lingually consistent annotation principles that have since been adapted to over 100 languages.

Notably, the CoNLL shared tasks of 2017 and 2018 (Zeman et al., 2017, 2018) featured UD at the forefront, inviting researchers to submit systems that could not only parse, but process the entirety of UD across its numerous annotation layers. Beyond parsing, UD has also been utilized for a variety of other ends, including cross-lingual transfer (Ammar et al., 2016; Pires et al., 2019; Wu and Dredze, 2020; Lauscher et al., 2020), domain adaptation (Li et al., 2019, 2020; Stymne, 2020), and linguistic typology (Futrell et al., 2015; Hahn et al., 2020; Levshina, 2019).

Though UD and other initiatives have aided in driving recent advances in NLP, overall progress has typically been measured via aggregate accuracy metrics, which provide little more than a bird's eye view into the data. In the era of deep learning, where popular models are notoriously opaque, it has thus proven vital to study the contents of datasets and identify aspects that may misrepresent model performance. In this vein, numerous studies have shown that the crowd-funded nature of some popular NLP datasets makes them prone to annotation artefacts that are readily exploitable by neural models as heuristics (Kaushik and Lipton, 2018; Gururangan et al., 2018; Poliak et al., 2018; McCoy et al., 2019). With such insights in mind, researchers have shifted their focus towards the *datasets* instead of the models, proposing general methods for exploring the former so as to better understand the performance of the latter. Such approaches have drawn from, e.g., information theory (Perez et al., 2021; Ethayarajh et al., 2022), item response theory (Rodriguez et al., 2021; Vania et al., 2021), and model training dynamics (Swayamdipta et al., 2020). This work, however, has predominately focused on classification tasks and has proven difficult to extend to other classes of problems, such as the structured prediction tasks of UD.

In this paper, we perform an analysis of (a large

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1076–1089
May 2-6, 2023 ©2023 Association for Computational Linguistics

subset of) UD v2.9 through the perspective of a popular parsing architecture — namely that of Dozat and Manning (2016). As opposed to much previous work, which prioritizes metrics like LAS in order to build accurate parsers, we aim instead to better understand the underlying data, as well as how our parser interfaces with it. To do so, we extend recently proposed dataset analysis methods based on model training dynamics (Swayamdipta et al., 2020), $\mathcal{V}$-information (Xu et al., 2020; Ethayarajh et al., 2022), and minimum description length (Blier and Ollivier, 2018; Voita and Titov, 2020; Perez et al., 2021) to the dependency parsing scenario. In working with each method, we formalize the following set of research questions:

1. Which treebanks appear *hard* (or *easy*) to parse, given a model's confidence throughout training, and variability therein?
2. Which treebanks contain the most (or least) information that is actually usable by a parser, with respect to a naive baseline?
3. Which treebanks are the most (or least) sample efficient, i.e. most easily fit by a parser, irrespective of training set size?

2 Universal Dependencies

Universal Dependencies (UD) (Nivre et al., 2016, 2020; de Marneffe et al., 2021) is an initiative focused on the development of dependency treebanks. UD is founded upon a *lexicalist* perspective on syntax, which posits that (syntactic) relations are formed directly between words. Although this approach does not take morphological segmentation explicitly into account, UD nonetheless provides such information in the form of lemmas, part-of-speech tags, and morphological features for each word. These design decisions have inspired the widespread adoption of UD as an annotation scheme, which has grown from 10 treebanks across 10 languages in v1.0 to 243 treebanks across 138 languages in v2.11.

2.1 Dependency Parsing

Though UD contains multiple layers, our focus in this paper is on its syntactic layer — the dependency tree annotation upon which parsers are trained and evaluated. As a task, dependency parsing amounts to mapping a sentence $\mathbf{x} = \{x_1, x_2, ..., x_n\}$ to its respective syntactic structure $\mathbf{y}$. This is typically a tree (a rooted, directed, acyclic graph) over $\mathbf{x}$, where each word $x_i \in \mathbf{x}$ (called a dependent) forms an edge with another word $x_j \in \mathbf{x}$ that it syntactically modifies (its head). Though such edges (x_i, x_j) are sometimes considered in isolation, most often they are accompanied by a label describing the relation between x_i and x_j. UD comprises of a base set of 37 such labels, in addition to treebank-specific subtypes that can be introduced by annotators.

Unlike classification or sequence labeling tasks, which entail predicting y given a fixed label set K, parsing is considered a structured prediction task, where the output space is constrained by the sentence length $n+1$ (with a special `root` symbol included). In data-driven parsing, a parser is typically a function f whose parameters θ are fit on some gold-annotated training set X_{train}, e.g. a UD treebank. A trained parser's predictions $\hat{\mathbf{y}} = f_\theta(\mathbf{x})$ can then be evaluated on a held out test set X_{test} by means of various metrics. Commonly, these are *labeled attachment score* (LAS), which is the percentage of all words in X_{test} that are assigned the correct head *and* label by f, and the *unlabeled attachment scores* (UAS), which is the percentage of words that are assigned the correct head, irrespective of label. We will focus primarily on LAS in this study, as it assesses both head attachment and labeling.

2.2 UD Parsing

In conjunction with UD's popularity and transition to v2.0, the CoNLL 2017 and 2018 Shared Tasks on Multilingual Parsing from Raw Text to Universal Dependencies (Zeman et al., 2017, 2018) sought to consolidate trends in parsing research with respect to a wealth of new data, algorithms, and models. Overall, the findings of both shared tasks offered many insights for the future of dependency parsing. Primarily, they helped establish Dozat and Manning (2016)'s parser as the most popular parsing architecture of the neural NLP era. Though many new architectures have been proposed since, the biaffine attention decoder continues to feature prominently in parsing research, where the focus has shifted to the matter of feature representation and fine-tuning, rather than decoding (see, e.g. Kondratyuk and Straka (2019); Üstün et al. (2020)).

3 Beyond Accuracy

In NLP, system performance is typically measured via accuracy, which has the advantage of being intuitive and straightforward to calculate. Moreover, it is often useful in model selection, as well in setting the state-of-the-art for a given dataset or task. Beyond these qualities, however, accuracy leaves much to be desired when investigating models vis-a-vis datasets. For one, it is often reported with respect to a single checkpoint from a model's training regime, which typically consists of numerous epochs and parameter updates. In honing in on one particular checkpoint (usually the best with respect to validation loss or accuracy), one cannot readily assess whether the model was easily fit on the data, or if training was stopped prematurely. Furthermore, in choosing the $\arg\max$ over the output distribution, one inevitably loses information about it: Was the model confident in making its prediction? Or was the distribution highly entropic? Also relevant is the train/test distinction: in evaluating on the latter, one can gauge a model's ability to generalize, but generally cannot assess the goodness-of-fit on the former, nor its sample efficiency. Likewise, accuracy cannot, in principle, adequately assess the *quality* of the training data: can the model learn from all instances therein? Or does the data contain a substantial amount of noise due to, e.g. annotation inconsistencies?

With regard to dependency parsing, accuracy-based metrics like LAS carry a number of additional drawbacks. For example, if working with an arc-factored graph-based dependency parser, one must score all possible n incoming edges for a dependent $x_i \in \mathbf{x}$. If $n = 1$, then x must necessarily be attached to the dummy node and assigned the `root` label. For a treebank consisting of many such sentences (e.g. Russian Taiga), this will lead to artificially inflated accuracies for that particular relation. Another concern is the potential proliferation of functional relations that may arise in some treebanks (e.g. Japanese GSD), where parsers often yield disproportionately high accuracies (Nivre and Fang, 2017). To transcend the limitations of accuracy-based measures in dependency parsing, we consider three recently proposed dataset analysis methods as a means of exploring UD treebanks: dataset cartography (Swayamdipta et al., 2020), $\mathcal{V}$-information (Xu et al., 2020), and minimum description length (Blier and Ollivier, 2018).

3.1 Dataset Cartography

Dataset cartography (Swayamdipta et al., 2020) is a method for analyzing training datasets via the lens of model training dynamics. Put briefly, DC assumes the use of a model f trained to minimize loss on a dataset D of size N. Crucially, for a given instance $D_i = (\mathbf{x}, y^*)$, DC posits that f defines a probability distribution over labels y, such that the probability of the true label $p(y^*|\mathbf{x})$ can be tracked throughout training. Given a gradient descent-based training regime of E epochs, DC defines the notion of *confidence* (CONF) as follows:

$$\textsc{Conf}_i = \frac{1}{E}\sum_{e=1}^{E} p_{\theta_e}(y^*|\mathbf{x})$$

where θ_e are f's parameters following the epoch e update. Intuitively, high CONF values (e.g. ≈ 0.95) for a given training instance D_i indicate that f generally assigns high probability to y^* throughout training — i.e. that D_i is "easy-to-learn". Conversely, low CONF values (e.g. ≈ 0.05) indicate that f generally "fails" to learn from those particular instances.

As a complement to CONF, Swayamdipta et al. (2020) also introduce the *variability* (VAR) metric, which summarizes the tendency of f to waver in its assignment of $p(y^*|\mathbf{x})$ throughout training. VAR is defined as follows:

$$\textsc{Var}_i = \sqrt{\frac{\sum_{e=1}^{E}(p_{\theta_e}(y^*|\mathbf{x}) - \textsc{Conf}_i)^2}{E}}$$

In essence, while CONF is the mean of $p(y^*|\mathbf{x})$ across E, VAR is its standard deviation. Using both of these metrics, Swayamdipta et al. (2020) are able to construct *data maps*, which visualize D through the perspective of f. With VAR and CONF plotted on the x and y axes, respectively, data maps help in identifying select regions of D that are easy or difficult for f to learn — or are otherwise ambiguous.

3.2 $\mathcal{V}$-Information

An alternative approach for quantifying dataset "difficulty" is proposed by Ethayarajh et al. (2022), who leverage the concept of $\mathcal{V}$-Information. Introduced by Xu et al. (2020), $\mathcal{V}$-information (denoted as $I_{\mathcal{V}}(X \rightarrow Y)$) is a framework for estimating the amount of information between random variables X and Y (input and output, respectively) that is *usable* by a model in family $\mathcal{V}$ — e.g., a sentiment classifier or syntactic parser. Here, *usability* is measured with respect to an encrypted form of the input $\varnothing$, from which $\mathcal{V}$ must nonetheless attempt to predict Y — essentially a label-only baseline. Predicting Y from X and $\varnothing$ amounts to measuring $\mathcal{V}$-entropy:

$$H_{\mathcal{V}}(Y) = \inf_{f \in \mathcal{V}} [-\log f'[\varnothing](Y)]$$

and *conditional* $\mathcal{V}$-entropy:

$$H_{\mathcal{V}}(Y|X) = \inf_{f \in \mathcal{V}} [-\log f[X](Y)]$$

where $f \in \mathcal{V}$ is a model that maximizes the log-likelihood of the labels Y with the original input X and f' is the same model trained on the encrypted input $\varnothing$. Given these two quantities, $I_{\mathcal{V}}(X \to Y)$ can be computed as follows:

$$I_{\mathcal{V}}(X \to Y) = H_{\mathcal{V}}(Y) - H_{\mathcal{V}}(Y|X)$$

In essence, $\mathcal{V}$-information is the amount of information between inputs and labels that can be estimated by $\mathcal{V}$ beyond the label distribution itself. Given that $\mathcal{V}$-information is computed with respect to $H_{\mathcal{V}}(Y)$, it is important to note that $I_{\mathcal{V}}(X \to Y) \geq 0$. Also, if X is independent of Y, then $I_{\mathcal{V}}(X \to Y) = 0$.

In addition to functioning as a summary statistic alternative to accuracy, $\mathcal{V}$-information can also be generalized to the instance-case. To do so, Ethayarajh et al. (2022) propose measuring point-wise difficulty, which they deem PVI and calculate as follows:

$$\text{PVI}(x \to y) = -\log_2 p_{f'}(y^*|\varnothing) + \log_2 p_f(y^*|\mathbf{x})$$

where $f_\theta, f'_\theta \in \mathcal{V}$ are models trained on normal and encrypted data, respectively. Recall that, as before, y^* refers to the gold label and not the one with the highest score. Unlike $\mathcal{V}$-information, PVI can return negative values at the instance level (similarly to pointwise mutual information (Shannon, 1948)), which indicates that the model would fare better choosing a class at random.

3.3 Minimum Description Length

Minimum description length (MDL) (Rissanen, 1978) is an information-theoretic concept that concerns the transmission of data through a specified channel — i.e., a probabilistic model. Ideally, a model that is fit well on some data will learn to transmit — or *compress* — it using as few bits as possible (Blier and Ollivier, 2018). Naively, in order to evaluate how well a model, e.g., a neural network, might learn to compress its training data, one might refer to the model's cross-entropy loss after training for a full cycle of E epochs, which amounts to the Shannon-Humman code (Shannon, 1948). However, a model endowed with enough parameters may learn to *fit* the data without necessarily *compressing* it — see, e.g., Zhang et al. (2021), who show that training loss can still be minimized on data that contains no inherent structure (shuffled labels). MDL is thus designed to express not only how well a model might learn to compress some data, but also how efficiently the model itself might be transmitted.

Blier and Ollivier (2018) outline various methods for compressing labels, with or without a model. Among these, they describe online (or prequential) coding (Rissanen, 1984; Dawid, 1984), which transmits the labels and model without explicitly compressing the latters's parameters. Online coding requires D to be partitioned into S blocks where $1 = t_0 < t_1 \cdots < t_S = N$. The model of choice, f, is initialized with parameters θ, learning algorithm $\mathcal{A}$, etc. The first block, t_0, is first evaluated with a uniform prior[1] and then used to train f_s (we omit the parameters θ for brevity). This model is then evaluated on t_{s+1} and reset to its initial state, where it is consequently trained on t_{s+1}, and so on. Formally, the online codelength can be expressed as:

$$L^{\text{online}}(y_{1:n}|x_{1:n}) = \sum_{s=1}^{S-1} \sum_{n=t_s}^{t_{s+1}} -\log_2 p_{f_s}(y_n|x_n)$$

In contrast to the training loss above, which represents the data codelength if the model parameters are known, L^{online} (henceforth MDL) is an implicit way of measuring the same without knowing the model parameters. Effectively, MDL estimates f's ability to generalize with respect to D: models that learn efficiently from limited instances will yield shorter codelengths.

4 Experimental Setup

In this section, we describe our data sampling procedure, the parsing model we employ, and our extension of the aforementioned analysis methods to the context of (graph-based) dependency parsing. More details about how each metric is calculated can be found in Appendix A.

Data In order to compare the analysis obtained with each of the dataset analysis methods, we require a representative sample of UD treebanks.

---

[1] More details on this can be found in Appendix A.

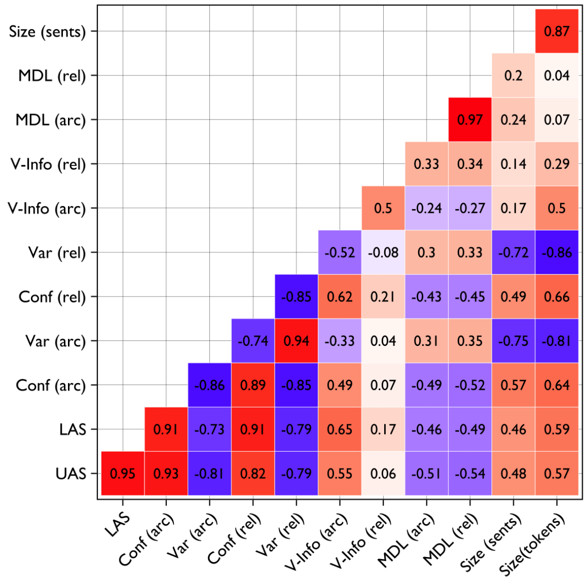


Figure 1: Spearman's ρ across all metrics of interest, including training set size in tokens and sentences.

Since each method requires the training of a parser on some subset of data (as well as a validation set for estimating V-INFO), we consider every UD treebank that contains train, validation, and test splits. To ensure that a) there is sufficient data for training our parsers and b) training is reasonably quick and the models do not overfit, we limit our selection to treebanks whose training sets contain at least 1,000 and at most 20,000 sentences. This gives us 88 treebanks across 58 languages,[2] with Faroese FarPahC having the smallest training partition (1,021 sentences, 23,094 tokens) and Polish PDB the biggest (17,773 sentences, 281,736 tokens). All resuls are reported with respect to UD v2.9, which was the most recent release at the time of experimentation.

Parsing Model We employ a neural parser based on Dozat and Manning (2016)'s biaffine decoder, as implemented in the `SuPar`[3] Python library. For input encoding, we make use of randomly initialized word embeddings ($d \in \mathbb{R}^{100}$) and LSTM-based character embeddings ($d \in \mathbb{R}^{100}$), as well as a stacked three-layer LSTM feature extractor ($d \in \mathbb{R}^{400}$). We choose to forego the use of POS embeddings (contrary to Dozat et al. (2017)), so as to maintain a direct correspondence between the input string and its tree, which is vital for measuring V-INFO. Furthermore, we do not initialize the input embeddings with pretrained representations in order to avoid confounds central to language coverage and overlap (Wu and Dredze, 2020). Each model is trained for 30 epochs with a batch size of 32 and optimized by Adam (Kingma and Ba, 2014) with a starting learning rate of 2e$-$3.

Analysis Methods In order to be able to measure CONF, VAR, and V-INFO, we need to be able to extract the probabilities that parsers assign to gold arcs and labels. Recall that Dozat and Manning (2016)'s parser is effectively a multi-task model, which jointly maximizes the log-likelihood of a given word's correct head, as well as the label for the relation. The arc and label logits are calculated via separate biaffine transformations, which yield a $S_{\text{arcs}} \in \mathbb{R}^{N \times N+1}$ matrix for the former and a $S_{\text{labels}} \in \mathbb{R}^{N \times N+1 \times R}$ tensor for the latter, where N is the sentence length and R is the size of the relation set. To obtain a normalized probability distribution, we apply a softmax to the last dimension of each S_{arcs} and S_{labels}, and index into the correct cell for the gold head and label probabilities.

For MDL, we train parsers on increasingly larger partitions of the training set, starting with a minimum of five sentences and doubling in size up to a maximum of 360 — a total of 995 sentences. We do so in reference to the smallest treebank in our sample (Faroese FarPahC (1,021 sentences)), so as to control for training set size, which varies drastically across treebanks. In accordance with the parser's multi-task design, we compute two separate MDL measures with respect to the separate arc and relation losses, averaged over five trials.

5 Results and Analysis

In this section, we analyze the results of our four metrics: CONF, VAR, V-INFO, MDL. We focus our analysis on arcs (probabilities and loss) and refer to Appendix B for the full set of results. For each metric, we begin with a brief discussion of its pairwise correlation to all other metrics, which is displayed for reference in Figure 1. We continue by framing our analyses with respect to the Top 3 best and worst performing treebanks for each metric.

5.1 Dataset Cartography

Figure 2 (left) depicts the mean CONF and VAR scores calculated across arcs. We observe a strong negative correlation between CONF and VAR. This relationship also appears to be influenced by 1) validation LAS and 2) training set size. This is not surprising, as training set size is a common

---

[2] We filter out four treebanks due to issues with tokenization, etc.

[3] `https://github.com/yzhangcs/parser`

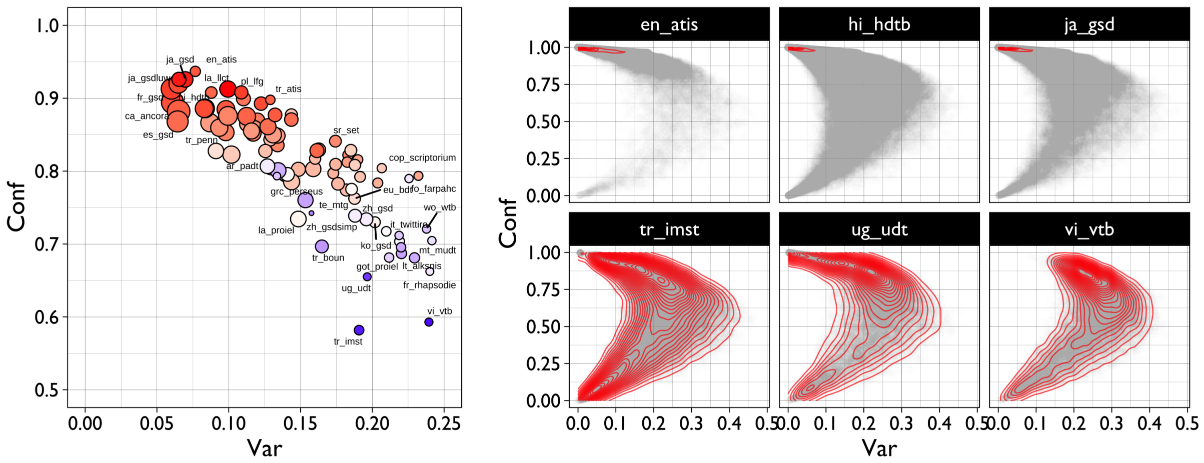


Figure 2: Left: mean CONF and VAR scores for Arcs across all languages; color represents Validation LAS and point size represents the size of the train set in words. Right: Data maps for Arcs for best and worst CONF scoring languages, with 2d densities super-imposed.

predictor of indicates that parser performance can, to a large extent, be reliably estimated simply by observing its confidence throughout training.

In zooming in on individual points, we observe several treebanks in the upper left-hand corner of the arc plot, which corresponds to high average CONF and low VAR. Indeed, many of these points belong to the largest treebanks in the sample, e.g., Hindi HDTB (0.92 CONF, 0.06 VAR, 281,057 tokens), French GSD (0.92, 0.06, 354,505), and Catalan AnCora (0.89, 0.06, 429,141). However, English Atis — a relatively small treebank with 48,655 tokens— tops out with the highest CONF overall at 0.94 (VAR: 0.08). This is unsurprising given the nature of the Atis dataset (Price, 1990), which collects transcriptions of requests sent to automated flight information systems, e.g. *list the nonstop flights early tuesday morning from dallas to atlanta*. The imperative nature of such requests, in combination with a small vocabulary, likely limits the range of structures a parser might encounter during training, thus making the treebank *easy* to fit. Interestingly, the second and third-highest CONF treebanks are both Japanese: GSDLUW (0.93, 0.07, 130,298) and GSD (0.93, 0.07, 168,333). This is likewise expected, as GSD has been observed by, e.g., Nivre and Fang (2017) to possess a large amount of functional relations, which can be parsed with near 100% accuracy.

On the other end of the spectrum, we observe that Turkish IMST (0.58, 0.19, 37,784), Vietnamese VTB (0.59, 0.24, 20,285), and Uyghur UDT (0.67, 0.20, 19,262) yield the three lowest CONF scores overall, as well as generally high VAR. Interestingly, though Turkish IMST does not appear within the top-25 highest VAR treebanks, it nonetheless yields the lowest CONF, indicating the the treebank might be particularly "hard" to parse for reasons other than treebank size (Çöltekin et al., 2017). Conversely, the fact that VTB and UDT are low CONF and high VAR implies that a lack of training data might play a role.

The data maps for the highest and lowest CONF scoring treebanks (depicted in Figure 2 (right)) highlight important disparities between these two groups. Most strikingly, we observe that the overwhelming majority of arcs in the English, Hindi, and Japanese treebanks are concentrated in the upper left high-CONF/low-VAR region, which is characterized as easy-to-learn by Swayamdipta et al. (2020). Indeed, this corresponds to 65.26, 68.02, and 68.32% of all tokens in the English, Hindi, and Japanese treebanks where CONF ≥ 0.95 and VAR ≤ 0.1. Conversely, only 3.04, 7.61, and 0.001% of all arcs in the Turkish, Uyghur, and Vietnamese treebanks are allocated to this region, indicating that this group contains few arcs that might be considered "trivially easy". Likewise, if we define the hard-to-learn region as consisting of points where CONF ≤ 0.25 and VAR ≤ 0.1, we find that 9.5, 5.45, and 5.06% of the latter group's treebanks can be characterized as such, respectively, compared to 0% for any treebank in the former group.

5.2 $\mathcal{V}$-Information

Figure 3 depicts the PVI density for the Top 3 and Bottom 3 treebanks in terms of V-INFO for arcs.

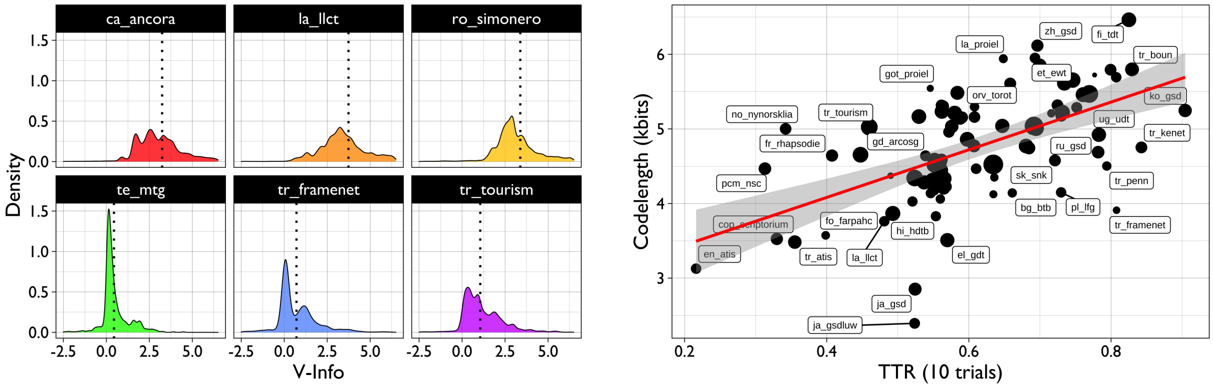


Figure 3: Left: Arc-level PVI density for Top-3 and Bottom-3 V-INFO treebanks, across arcs (labels omitted for space). Right: Block-wise codelength (in bits) for Top-3 and Bottom-3 MDL treebanks, across arcs.

We observe weaker correlations between V-INFO and LAS when measured across all treebanks than we did for CONF or VAR, indicating that V-INFO measures different aspects of parser performance than either of these metrics. Immediately, we can see that the lowest-ranked treebanks — Telugu MTG (0.44 average V-INFO), Turkish FrameNet (0.71), and Turkish Tourism (1.05) — have densities that are skewed towards 0. In particular, Telugu shows a sharp peak around this point, suggesting that it contains many arcs and relations that can be inferred by sentence sizes and label distributions. In contrast, the distributions representing the highest V-INFO treebanks (Latin LLCT (3.74), Romanian SiMoNERo (3.38), and Catalan AnCora (3.24)) are much flatter and more evenly distributed along the space of positive V-INFO values.

Turkish Tourism is composed entirely of hotel and restaurant reviews. Due to the nature of this genre, the treebank's vocabulary is very limited and many of its sentences are exceptionally short — 4.77 tokens on average in the dev set. This limits the space of possible trees that a parser may encounter, potentially biasing it towards certain structures. For example, the adverb "çok" (very) appears in the first position of 128 sentences as modifier of the 2nd word (typically an adjective, e.g., "güzel" (good)). A similar effect can be observed for Telugu MTG, which contains 5.05 words per sentence in its development set, and for Turkish FrameNet, which systematically places the root of the sentence at the penultimate position (91.7% of the time).

Given the observations above, V-INFO can be imagined as a means of simultaneously penalizing regularity and stochasticity in data. We can illustrate this further by returning to the CONF results. Recall that, for arcs, the highest scoring treebank was English Atis. Interestingly, when measuring its arc V-INFO, we find that its rank drops to 77 out of 88 (1.97). Recall that the Atis dataset contains a single genre of data with simple sentences often beginning with *what is*, *show me*, etc. In this case, simply guessing $(1, 0)$ and $(2, 1)$ for the first two arcs yields accuracies of 0.52 and 0.41, respectively, which the label-only baseline employed for calculating V-INFO would likewise capture. By contrast, the lowest scoring CONF treebank — Turkish IMST — retains a low rank of 83 (1.27). Recalling that 9.5% of IMST's tokens are considered hard-to-learn by the cartography metrics, this treebank is likely to contain certain annotation inconsistencies that cannot be systematically captured beyond guessing.[4] Such cases, too, are captured by V-INFO. With this in mind, we might surmise that *high* V-INFO values might represent treebanks with a varied distribution of structures that are likewise consistently annotated, thus requiring strong generalization of the parser.

In addition to arcs, we would also like to note the interesting behavior of V-INFO for relations, which is generally uncorrelated with any other metric (other than arcs). This is due to the fact that the parser must know the identity of both words when deciding upon the relation that binds them. For the vast majority of relations, this cannot be determined from position alone, even if head placement is largely systematic (as in the aforementioned Atis and Tourism treebanks). As a result, aggregate

---

[4] Indeed, this is corroborated by Türk et al. (2019), who discuss the errors resulting from the automatic conversion of IMST to the UD format, among which are 269 cases where a subject bearing a genitive marker was misidentified as an object in the accusative case.

V-INFO for relations (3.89 ± 3.66) is noticeably higher than for arcs (2.40 ± 1.98), albeit with more spread. It is important to note, however, that there are two primary exceptions to this in the form of the `root` and `punct` relations, which can be deterministically assigned to the first (dummy node) and last (end of sentence punctuations) positions.

5.3 MDL

In terms of MDL, we observe only moderate correlations across all other measures, with the strongest between CONF for arcs. Here, we find that MDL for relations (which takes the model's loss into account) is actually more correlated with arc CONF than CONF for relations. Expectably, neither measure is correlated with treebank size, which was controlled across treebanks.

For rankings, we observe that the Top 3 treebanks (for arcs) are largely the same ones as returned by arc CONF: Japanese GSD and GSDLUW, (2.85, 2.39) English Atis (3.12), and Turkish Atis (3.48). This is unsurprising given the reasons outlined in previous sections. Therefore, it seems feasible that MDL — which measures a parser's fit on successively larger, unseen partitions of a dataset — would reflect such qualities as well. MDL's alternative interpretation as a measure of a dataset's sample efficiency is also applicable here: a parser trained on a small number of trees will likely generalize well in these very narrow distributions.

Interestingly, we encounter a handful of new treebanks at the higher end of MDL scores: Finnish TDT (6.46), Chinese GSD (traditional and simplified) (6.11, 6.14), and Latin PROIEL (5.94). Intuitively, a high MDL in the case of parsing might suggest that the model is exposed to a larger diversity of token types during training, which could hinder it in learning various types of dependencies. Following this logic, one might hypothesize that treebanks representing languages with complex morphological systems might yield comparatively higher MDL, due to the higher number of word forms that appear therein. In line with this, we observe that the highest MDL treebanks tend to be morphologically rich, e.g., Finnish, Latin, Turkish, Estonian, Polish, Russian.

In an attempt to quantify the correspondence between a treebank's attested morphological complexity and its MDL, we compute a series of proxy metrics, as featured in Çöltekin and Rama (2018). These include type-token ratio (TTR) (averaged across 10 random samples of 1,000 sentences), number of feature types (in these samples), and feature entropy (calculated across feature types). Indeed, Figure 3 shows a strong correspondence between TTR and MDL, as the two are highly correlated ($\rho = 0.58, p < 0.001$). Significant correlations for feature entropy ($\rho = 0.35, p < 0.001$) and number of feature types ($\rho = 0.28, p < 0.001$) corroborate our hypotheses further. Interestingly, while both Chinese GSD treebanks simultaneously yield the highest MDL and comparatively high morphological scores, the language itself is typically described as having an analytical (impoverished) morphology. We surmise, however, that this fact — combined with Chinese's logographic writing system — contributes to high MDL scores in the same way as morphological richness: a lack of high-frequency function words and a wide range of lexical items lead to large vocabulary sizes. MDL for these treebanks is thus expectably high.

6 Conclusion

In this paper, we investigated 88 UD treebanks through the lens of dataset difficulty measures. We found that CONF and VAR are capable of painting a nuanced picture of how *easy* or *hard* treebanks might be to parse. We also observed that a model's confidence throughout training is an excellent indicator of how well it might generalize to held-out data. Regarding V-INFO, we observed that the measure tends to simultaneously penalize high degrees of predictability and stochasticity, and that treebanks otherwise characterized as *easy* may have low V-INFO due to lack of structural diversity. Finally, treebanks with high MDL seem to be characterized by low sample efficiency, which in turn is related to morphological complexity and vocabulary usage. Given the broad range of insights expressed via these metrics, we hope that our results — however preliminary — will inspire future researchers to pursue a greater understanding of UD as trove of data, so as to push further boundaries in the realms of parsing, typology, etc.

In terms of future work, we make note of several potentially interesting directions. Indeed, one of the main drawbacks of our experimental design is that it only accounts for the perspective of a single parsing architecture — albeit (arguably) the most popular one of the neural era. Dependency parsing, however, has a long research tradition where many different parsing models have been proposed throughout the years — each with their respective advantages and drawbacks (see, e.g. McDonald and Nivre (2007, 2011)). Though we chose to work with an arc-factored graph-based parser due to the need for extracting arc-level probabilities, future studies may consider consider ways of leveraging transition-based parsers (Nivre, 2003) or models that directly maximize full tree probabilities (Koo et al., 2007; Ma and Hovy, 2017). If working with a wide range of parsing models, one could employ item response theory (Rodriguez et al., 2021; Vania et al., 2021), which is a framework that consolidates many predictions per instance in order to identify regions of datasets that may be perceived as difficult, easy, etc. Certainly, this would provide a more broad perspective on UD than what we have offered here.

A different direction that could be explored is data *selection*, which is indeed what Swayamdipta et al. (2020) proposed as the main uses for dataset cartography. Although the CONF and VAR metrics provide valuable insights about UD treebanks in our case, they are nonetheless measured at the *token*-level. This is distinct from their original application, in that each token is crucial for the composition a sentence and cannot be readily removed. Although we experimented with measuring sentence-level CONF and VAR, some preliminary results indicated that a naive application of Koo et al. (2007)'s method is ultimately confounded by sentence length. As such, it would be interesting to experiment with models that directly optimize for tree probability, such as Ma and Hovy (2017). If successful, this would allow us to identify select subsets of treebanks for the purpose of training more accurate parsers with less data, or for choosing the least noisy sentences for typological studies, etc.

Limitations

As already mentioned, the main limitation to our work is that we focus on a single parsing architecture. Indeed, it would be preferable to extend the experiments described here to other parsers in order to evaluate the generalizability of our results. Ideally, we might choose to work with a comparable transition-based parsing algorithm, which have been shown to exhibit different error profiles than their graph-based counterparts (McDonald and Nivre, 2007; Kulmizev et al., 2019). However, the fact that transition-based parsers calculate probabilities over *transitions* instead of *arcs* would render such parsers incompatible with the cartography and V-INFO measures, which reveal interesting insights about our surveyed treebanks. Beyond this, we note that the scope of our current work is quite large, as we compare 10 metrics across 88 treebanks. By this token, we can admittedly only offer a bird's eye view of the UD treebank collection, even if our surveyed metrics offer more nuance than attachment scores.

Ethical Considerations

The research presented in this paper is compatible with the ACL ethics policy. The datasets used come from the Universal Dependencies repository and have appropriate licenses and documentation. The experiments are done with small-scale models that do not have a significant impact in terms of energy consumption.

Acknowledgements

We would like to thank Anders Søgaard for providing useful feedback on an early version of this work, as well as Sasha Berdicevskis for lending insight about computational measures of morphological complexity. We acknowledge the computational resources provided by CSC in Helsinki through NeIC-NLPL (`www.nlpl.eu`).

References

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Léonard Blier and Yann Ollivier. 2018. The description length of deep learning models. *arXiv preprint arXiv:1802.07044*.

Çağrı Çöltekin, Ben Campbell, Erhard Hinrichs, and Heike Telljohann. 2017. Converting the tüba-d/z treebank of german to universal dependencies. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 27–37.

Çağrı Çöltekin and Taraka Rama. 2018. Exploiting universal dependencies treebanks for measuring morphosyntactic complexity. *Proceedings of the First Shared Task on Measuring Language Complexity*, pages 1–8.

A Philip Dawid. 1984. Present position and potential developments: Some personal views statistical theory the prequential approach. *Journal of the Royal Statistical Society: Series A (General)*, 147(2):278–290.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. pages 5988–6008.

Richard Futrell, Kyle Mahowald, and Edward Gibson. 2015. Large-scale evidence of dependency length minimization in 37 languages. *Proceedings of the National Academy of Sciences*, 112(33):10336–10341.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Michael Hahn, Dan Jurafsky, and Richard Futrell. 2020. Universals of word order reflect optimization of grammars for efficient communication. *Proceedings of the National Academy of Sciences*, 117(5):2347–2353.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Terry Koo, Amir Globerson, Xavier Carreras Pérez, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150.

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China. Association for Computational Linguistics.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Natalia Levshina. 2019. Token-based typology and word order entropy: A study based on universal dependencies. *Linguistic Typology*, 23(3):533–572.

Ying Li, Zhenghua Li, and Min Zhang. 2020. Semi-supervised domain adaptation for dependency parsing via improved contextualized word representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3806–3817, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhenghua Li, Xue Peng, Min Zhang, Rui Wang, and Luo Si. 2019. Semi-supervised domain adaptation for dependency parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2386–2395, Florence, Italy. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2017. Neural probabilistic model for non-projective MST parsing. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 59–69, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic. Association for Computational Linguistics.

Ryan McDonald and Joakim Nivre. 2011. Analyzing and integrating dependency parsers. *Computational Linguistics*, 37(1):197–230.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 149–160, Nancy, France.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Joakim Nivre and Chiao-Ting Fang. 2017. Universal Dependency evaluation. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 86–95, Gothenburg, Sweden. Association for Computational Linguistics.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. Rissanen data analysis: Examining dataset characteristics via description length. *arXiv preprint arXiv:2103.03872*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Patti Price. 1990. Evaluation of spoken language systems: The atis domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990.*

Jorma Rissanen. 1978. Modeling by shortest data description. *Automatica*, 14(5):465–471.

Jorma Rissanen. 1984. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information theory*, 30(4):629–636.

Pedro Rodriguez, Joe Barrow, Alexander Miserlis Hoyle, John P. Lalor, Robin Jia, and Jordan Boyd-Graber. 2021. Evaluation examples are not equally informative: How should that change NLP leaderboards? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4486–4503, Online. Association for Computational Linguistics.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Sara Stymne. 2020. Cross-lingual domain adaptation for dependency parsing. In *Proceedings of the 19th International Workshop on Treebanks and Linguistic Theories*, pages 62–69, Düsseldorf, Germany. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Utku Türk, Furkan Atmaca, Şaziye Betül Özateş, Balkız Öztürk Başaran, Tunga Güngör, and Arzucan Özgür. 2019. Improving the annotations in the turkish universal dependency treebank. In *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, pages 108–115.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly Universal Dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2302–2315, Online. Association for Computational Linguistics.

Clara Vania, Phu Mon Htut, William Huang, Dhara Mungra, Richard Yuanzhe Pang, Jason Phang, Haokun Liu, Kyunghyun Cho, and Samuel R. Bowman. 2021. Comparing test sets with item response theory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1141–1158, Online. Association for Computational Linguistics.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings* *of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. 2020. A theory of usable information under computational constraints. *arXiv preprint arXiv:2002.10689*.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

A Analysis Method Details

Since every surveyed dataset analysis method was proposed with classification tasks in mind, we must make numerous modifications in order to make them applicable to structured prediction problems like dependency parsing.

Dataset Cartography Given that we can index directly into S_{arcs} and S_{labels} throughout the training regime, the process of calculating CONF and VAR for each token is relatively straightforward. As such, we keep track of the probabilities assigned to the gold arcs and labels in the train and development sets after each epoch.

$\mathcal{V}$-information In order to calculate V-INFO, we must be able to estimate $\mathcal{V}$-entropy and conditional $\mathcal{V}$-entropy. Though the latter can be computed by simply fitting a model $f' \in \mathcal{V}$ on a designated training set, the former requires the "encryption" of the input X. To do so, we follow Ethayarajh et al. (2022) in setting every input token string $x \in X$ to _. $\mathcal{V}$-entropy can then be estimated by fitting f' on the same, albeit encrypted, training set, and $I_{\mathcal{V}}(X \to Y)$ subsequently calculated on a held-out (also encrypted) validation set. Though we track V-INFO across all epochs, we report it for $e = 30$, across arcs and labels. Also, it is worth noting that, although we attempted to compute $I_{\mathcal{V}}(X \to Y)$ at the sentence level via unlabeled tree probabilities extracted via Koo et al. (2007)'s method, the model f' trained on encrypted data produced extremely low probabilities, which led to underflow when computing the logarithm. As such, we chose to forego further exploration of this problem for this study.

Minimum Description Length Since our treebanks vary greatly in size, we must set our partitions such that they can span the length of the smallest training set (1,021 sentences). This way, our estimation of MDL remains comparable across treebanks. To do so, we employ partitions $S = \{5, 10, 20, 40, 80, 160, 320, 360\}$ at the sentence level, where the entire training set is shuffled prior to partitioning. For t_0, which does not contain any training data, we follow Voita and Titov (2020) in calculating the codelength over t_0 using a uniform prior. This is computed as follows:

$$L^{\text{unif}}(y_{1:s_1}|x_{1:s_1}) = \sum_{i=1}^{n} \log_2 K_{t_{1_i}} \qquad (1)$$

where K is the number of words (arcs or labels) in a given sentence i in the first partition t_1 (composed of $s = 5$ sentences). Note that this is the extension of uniform coding proposed by Blier and Ollivier (2018) to structured prediction. In a typical classification task with K labels, the same codelength is computed as $L^{\text{unif}}(y_{1:s_1}|x_{1:s_1}) = s_1 \log_2 K$.

For all remaining $t \in S$, we revert a model to its initial state, train it on t_i and compute its codelength over t_{i+1}. Since MDL is particularly sensitive to the ordering of instances within S, we repeat this process for 5 trials, fully re-initializing the model after each one.

B Full Results

The results for all metrics on all treebanks can be found in Table 1.

		Size	Rank	Accuracy		Arcs				Relations			
Language	Treebank	# Tokens	Abs. Rank	UAS	LAS	Conf	Var	V-Info	MDL	Conf	Var	V-Info	MDL
Afrikaans	AfriBooms	33.88	46	84.48	79.48	0.80	0.21	2.72	3.87	0.89	0.13	2.64	3.69
Ancient Greek	PROIEL	187.03	55	85.00	79.92	0.79	0.14	2.83	5.30	0.87	0.09	2.66	5.10
Ancient Greek	Perseus	159.90	75	75.13	68.61	0.76	0.15	2.57	5.32	0.85	0.10	2.57	5.31
Arabic	PADT	223.88	48	85.45	79.39	0.82	0.10	2.43	4.52	0.88	0.06	2.31	4.94
Armenian	ArmTDP	42.10	66	83.07	74.69	0.70	0.22	2.97	5.18	0.76	0.17	2.90	5.27
Basque	BDT	72.97	71	82.44	77.90	0.76	0.19	2.34	5.29	0.87	0.13	2.48	5.21
Bulgarian	BTB	124.34	15	91.61	86.76	0.90	0.11	2.83	4.14	0.93	0.07	2.60	4.37
Catalan	AnCora	429.14	2	92.98	90.13	0.89	0.06	3.24	4.34	0.94	0.03	2.68	4.17
Chinese	GSD	98.62	77	78.68	73.78	0.73	0.20	2.78	6.12	0.87	0.15	2.82	6.14
Chinese	GSDSimp	98.62	74	78.76	73.90	0.74	0.19	2.78	5.95	0.87	0.13	2.82	6.20
Coptic	Scriptorium	30.84	54	86.36	82.54	0.79	0.23	2.86	3.53	0.86	0.21	2.43	3.74
Croatian	SET	152.86	31	88.44	83.86	0.85	0.13	3.02	5.21	0.90	0.08	2.81	5.00
Czech	FicTree	133.64	24	91.28	86.52	0.87	0.12	2.90	4.77	0.89	0.09	2.84	4.74
Danish	DDT	80.38	53	83.88	80.56	0.77	0.18	2.78	5.13	0.90	0.10	2.87	5.15
Dutch	Alpino	186.03	29	91.35	87.44	0.86	0.13	3.03	5.48	0.91	0.07	2.78	5.42
Dutch	LassySmall	75.13	52	84.95	78.94	0.83	0.19	2.85	4.79	0.89	0.11	2.71	5.00
English	Atis	48.66	13	93.63	89.39	0.94	0.08	1.97	3.13	0.94	0.07	1.48	2.99
English	EWT	204.58	23	88.95	85.98	0.85	0.10	2.79	5.03	0.94	0.05	2.96	5.24
English	GUM	103.40	25	88.66	85.65	0.84	0.13	3.05	4.86	0.92	0.08	3.04	5.00
English	LinES	57.37	45	85.77	81.43	0.80	0.17	2.78	4.64	0.89	0.12	2.81	4.64
English	ParTUT	43.50	44	86.52	81.67	0.78	0.20	2.86	4.21	0.88	0.14	2.87	4.39
Estonian	EWT	49.60	86	75.94	68.64	0.69	0.22	2.31	5.85	0.78	0.17	2.60	5.73
Faroese	FarPaHC	23.09	62	79.24	73.43	0.79	0.23	2.43	3.57	0.86	0.17	2.49	3.57
Finnish	FTB	127.60	69	86.54	81.32	0.80	0.15	2.07	5.79	0.86	0.12	2.34	5.76
Finnish	TDT	162.82	59	85.68	80.96	0.80	0.16	2.68	6.46	0.88	0.12	3.09	6.42
French	GSD	354.50	4	94.49	91.46	0.91	0.06	3.13	4.44	0.94	0.04	2.62	4.45
French	Rhapsodie	19.14	78	80.63	73.45	0.66	0.24	2.57	4.64	0.79	0.19	2.58	4.62
French	Sequoia	50.51	26	89.12	85.50	0.82	0.18	3.16	4.14	0.90	0.12	2.70	4.13
Galician	CTG	79.33	37	85.13	81.83	0.79	0.14	2.82	4.03	0.91	0.07	2.15	4.23
German	GSD	263.80	19	88.23	83.32	0.87	0.09	2.79	4.35	0.94	0.04	2.77	4.52
Gothic	PROIEL	35.02	84	79.59	71.86	0.68	0.21	2.26	5.54	0.77	0.17	2.45	5.57
Greek	GDT	42.33	37	86.89	82.84	0.82	0.19	3.00	3.51	0.88	0.13	2.53	3.60
Hebrew	HTB	137.72	18	88.68	84.94	0.86	0.12	2.93	4.18	0.92	0.07	2.77	4.38
Hindi	HDTB	281.06	6	94.10	90.57	0.92	0.06	3.02	3.83	0.94	0.05	2.05	3.90
Icelandic	Modern	123.87	34	91.32	88.40	0.83	0.16	2.79	5.17	0.90	0.11	2.82	5.14
Indonesian	GSD	97.60	49	85.86	78.71	0.80	0.13	2.88	5.04	0.85	0.10	2.70	4.83
Irish	IDT	95.88	32	86.61	80.02	0.84	0.13	2.60	4.23	0.89	0.10	2.73	4.29
Italian	ISDT	276.02	12	91.78	88.80	0.89	0.08	2.98	4.59	0.94	0.05	2.61	4.41
Italian	ParTUT	48.93	40	87.63	83.48	0.81	0.18	2.88	4.28	0.90	0.11	2.58	4.42
Italian	PoSTWITA	99.54	60	83.23	76.76	0.80	0.14	2.63	5.32	0.86	0.09	2.76	5.36
Italian	TWITTIRO	23.63	76	79.60	71.49	0.71	0.22	2.56	4.63	0.78	0.17	2.49	4.62
Italian	VIT	225.08	21	87.94	84.94	0.86	0.09	3.04	4.59	0.92	0.05	2.55	4.46
Japanese	GSD	168.33	3	94.78	93.46	0.93	0.07	2.88	2.85	0.97	0.04	2.06	2.62
Japanese	GSDLUW	130.30	8	94.86	93.70	0.93	0.07	2.46	2.39	0.97	0.04	1.85	2.28
Korean	GSD	56.69	80	81.91	76.14	0.73	0.20	2.13	5.25	0.83	0.16	2.07	5.29
Latin	LLCT	194.14	1	96.45	95.12	0.91	0.10	3.74	3.76	0.96	0.05	2.90	4.02
Latin	PROIEL	172.13	67	81.81	75.85	0.73	0.15	2.49	5.94	0.83	0.11	2.87	6.04
Latvian	LVTB	201.60	41	87.03	83.32	0.85	0.13	2.83	5.72	0.92	0.09	2.99	5.57
Lithuanian	ALKSNIS	47.64	81	76.43	69.36	0.68	0.23	2.43	5.47	0.81	0.16	2.75	5.29
Maltese	MUDT	22.88	67	81.10	73.05	0.70	0.24	2.69	4.47	0.77	0.21	2.83	4.64
Naija	NSC	111.88	11	93.09	90.47	0.89	0.12	3.18	4.47	0.93	0.07	2.67	4.37
Norwegian	Bokmaal	243.89	14	91.88	89.37	0.88	0.10	2.87	5.15	0.95	0.05	2.86	5.09
Norwegian	Nynorsk	245.33	15	90.97	88.47	0.88	0.11	2.95	5.12	0.94	0.06	2.92	4.98
Norwegian	NynorskLIA	35.21	79	75.58	69.24	0.70	0.22	2.17	5.00	0.83	0.16	2.61	5.18
Old Church Slavonic	PROIEL	37.43	81	81.87	74.62	0.72	0.21	2.31	5.30	0.79	0.18	2.43	5.23
Old East Slavic	TOROT	118.63	73	81.85	75.31	0.80	0.14	2.04	5.61	0.86	0.10	2.35	5.74
Old French	SRCMF	158.62	39	89.07	83.86	0.86	0.11	2.67	5.23	0.90	0.07	2.68	5.28
Persian	Seraji	121.06	36	88.28	84.13	0.83	0.16	3.14	4.77	0.91	0.08	2.44	5.04
Polish	LFG	104.75	20	94.43	91.37	0.91	0.11	2.23	4.15	0.92	0.08	2.27	4.27
Polish	PDB	281.74	30	91.63	85.57	0.88	0.10	2.89	5.65	0.88	0.08	3.08	5.68
Portuguese	Bosque	171.78	17	90.22	86.57	0.85	0.12	3.12	4.26	0.92	0.06	2.60	4.34
Portuguese	GSD	255.31	7	92.31	90.37	0.89	0.08	3.09	4.06	0.96	0.04	2.45	4.27
Romanian	RRT	185.11	28	89.69	83.80	0.86	0.12	2.97	5.04	0.89	0.07	2.85	4.96
Romanian	SiMoNERo	116.86	9	92.43	89.19	0.88	0.13	3.38	4.12	0.93	0.07	2.69	4.18
Russian	GSD	74.91	47	86.42	81.26	0.81	0.17	2.89	4.69	0.88	0.11	2.57	4.60
Russian	Taiga	176.63	65	77.75	69.63	0.80	0.13	2.28	5.61	0.89	0.08	2.61	5.45
Scottish Gaelic	ARCOSG	63.70	51	86.57	80.83	0.82	0.16	2.62	4.65	0.87	0.14	2.66	4.36
Serbian	SET	74.26	33	89.18	85.07	0.84	0.17	3.07	4.74	0.89	0.11	2.72	4.68
Slovak	SNK	80.63	42	85.90	79.84	0.88	0.14	2.48	4.58	0.89	0.11	2.76	4.61
Slovenian	SSJ	112.53	22	90.46	87.66	0.87	0.14	3.09	5.01	0.93	0.08	2.83	4.68
Spanish	AnCora	452.74	5	91.90	88.90	0.88	0.07	3.19	4.51	0.94	0.03	2.73	4.12
Spanish	GSD	382.44	10	90.75	87.72	0.87	0.06	3.04	4.33	0.94	0.03	2.58	4.21
Swedish	LinES	55.45	56	85.07	79.98	0.79	0.19	2.72	4.96	0.87	0.13	2.82	4.85
Swedish	Talbanken	66.65	57	83.48	78.97	0.81	0.19	2.67	5.16	0.89	0.13	2.88	5.24
Telugu	MTG	5.08	72	86.47	66.87	0.74	0.16	0.44	4.56	0.60	0.13	0.68	4.40
Turkish	Atis	36.20	27	90.85	88.44	0.90	0.13	1.13	3.48	0.93	0.11	1.59	3.29
Turkish	BOUN	98.21	83	76.09	67.92	0.70	0.16	1.71	5.80	0.81	0.12	2.24	5.76
Turkish	FrameNet	16.33	63	86.49	71.15	0.79	0.13	0.71	3.91	0.65	0.15	1.16	3.94
Turkish	IMST	37.78	88	67.65	58.38	0.58	0.19	1.29	5.69	0.70	0.16	1.83	5.86
Turkish	Kenet	143.29	61	87.14	73.74	0.81	0.13	1.53	4.75	0.79	0.10	1.91	4.69
Turkish	Penn	166.51	50	88.11	77.55	0.83	0.09	1.61	4.50	0.82	0.08	1.89	4.56
Turkish	Tourism	71.24	34	93.57	88.20	0.91	0.09	1.06	5.02	0.89	0.08	1.34	4.89
Ukrainian	IU	92.40	58	85.78	80.97	0.78	0.18	2.88	5.46	0.87	0.11	2.81	5.33
Urdu	UDTB	108.69	43	86.51	80.29	0.83	0.13	2.75	4.32	0.87	0.09	1.88	4.27
Uyghur	UDT	19.26	85	75.25	60.47	0.66	0.20	1.35	4.92	0.66	0.17	1.46	4.96
Vietnamese	VTB	20.29	87	67.44	57.77	0.59	0.24	1.42	5.21	0.71	0.22	1.84	5.14
Western Armenian	ArmTDP	72.09	64	82.99	75.46	0.78	0.19	2.77	5.22	0.82	0.14	2.90	5.24
Wolof	WTB	23.56	70	79.00	71.16	0.72	0.24	2.66	4.37	0.80	0.20	2.61	4.62

Table 1: Results across all treebanks and metrics. Size is measured per thousand tokens. Green cells indicate that the treebank is Top 3 in that category. Red cells correspond to Bottom 3.

On Robustness of Prompt-based Semantic Parsing with Large Pre-trained Language Model: An Empirical Study on Codex

Terry Yue Zhuo[1][2] and **Zhuang Li**[2]*
Yujin Huang[2] and **Fatemeh Shiri**[2]
Weiqing Wang[2] and **Gholamreza Haffari**[2] and **Yuan-Fang Li**[2]
[1]CSIRO's Data61, Australia
[2]Monash University, Australia
`{terry.zhuo, zhuang.li}@monash.edu`

Abstract

Semantic parsing is a technique aimed at constructing a structured representation of the meaning of a natural-language question. Recent advances in language models trained on code have shown superior performance in generating these representations compared to language models trained solely on natural language text. The existing fine-tuned neural semantic parsers are vulnerable to adversarial attacks on natural-language inputs. While it has been established that the robustness of smaller semantic parsers can be enhanced through adversarial training, this approach is not feasible for large language models in real-world scenarios, as it requires both substantial computational resources and expensive human annotation on in-domain semantic parsing data. This paper presents the first empirical study on the adversarial robustness of a prompt-based semantic parser based on CODEX, a state-of-the-art (SOTA) language model trained on code. Our results demonstrate that the large language model of code is vulnerable to carefully crafted adversarial examples. To overcome this challenge, we propose methods for enhancing robustness without requiring substantial amounts of labelled data or intensive computational resources.

1 Introduction

Semantic parsing is a technique that transforms natural-language utterances (NLs) into machine-readable logical forms (LFs) and has been widely applied in various research fields, such as code generation, question-answering systems, and dialogue systems (Kamath and Das, 2018). Most current state-of-the-art semantic parsers are deep-learning models trained in a supervised manner using in-domain data. However, this approach requires a large amount of in-domain semantic parsing data, which can be costly to obtain (Bapna et al., 2017).

To address this issue, prompt-based semantic parsers based on large pre-trained language models, such as Codex (Chen et al., 2021) and GPT-J (Wang and Komatsuzaki, 2021), have become a new choice for semantic parsing applications. Prompt-based semantic parsers learn to solve a new task by in-context learning, instructing the parsers to generate correct LFs by constructing the prompt with a few demonstration examples. Such a method can significantly lower the cost of annotations by including only a few exemplars in the prompt and achieve comparable results to fully-supervised semantic parsers (Shin and Van Durme, 2022).

Recent studies (Huang et al., 2021; Pi et al., 2022) show that fully-supervised semantic parsers are vulnerable to adversarial attacks, which perturb input sentences into their semantic equivalent adversaries to mislead models to produce attacker-desired outputs. Hence, to mitigate such attacks, various adversarial training methods (Tramer and Boneh, 2019; Shafahi et al., 2019; Ganin et al., 2016; Shafahi et al., 2020) have been proposed to improve the adversarial robustness of the semantic parsers. In light of this, two main questions naturally arise: (1) *Do prompt-based semantic parsers based on large pre-trained language models also suffer from adversarial attacks?* (2) *If so, how can we improve the robustness of the large prompt-based semantic parsers?*

To address the former question, we evaluate the prompt-based semantic parsers on several evaluation sets built by different perturbation approaches mentioned in the AdvGLUE (Wang et al., 2021) dataset. Adopting the adversarial evaluation metrics proposed by Huang et al. (2021), it is found that the prompt-based semantic parsers are vulnerable to various types of adversarial attacks.

According to the experimental results from the first step, we perform a three-fold experiment to answer the latter questions. The first aspect of the study aims to determine if the inclusion of

---

[*] corresponding author

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1090–1102
May 2-6, 2023 ©2023 Association for Computational Linguistics

additional examples within the prompt during in-context learning improves the robustness of prompt-based parsers. This hypothesis is based on prior research that has demonstrated that the increase in the size of the training data results in an enhancement of robustness in fully-supervised models (Pang et al., 2019). The second part of the study aims to determine if the integration of few-shot adversarial examples within prompts can improve the robustness of Codex. This was based on the observation that conventional adversarial training methods often include adversarial examples within the training set (Miyato et al., 2016; Tramer and Boneh, 2019). Finally, the third part of the study aims to evaluate if sampling methods other than random sampling can select more effective examples that improve the robustness of prompt-based parsers.

In this work, we perform a series of experiments to probe CODEX, a large pre-trained model trained on code, on two semantic parsing benchmarks, `GeoQuery` (Zelle and Mooney, 1996) and `Scholar` (Iyer et al., 2017). Our key findings from the above experiments are as follows:

- Prompt-based semantic parsers are vulnerable to adversarial examples, particularly the ones crafted by sentence-level perturbations.
- In-context learning with more demonstration examples in the prompt can improve the in-domain robustness of prompt-based parsers.
- Augmenting the prompt with adversarial examples has limited effect in improving the robustness of prompt-based parsers.
- The few-shot example sampling strategy with higher language complexity can result in stronger robustness for the prompt-based parsers.

2 Related Work

Prompt-based Learning. Prompt-based learning is an alternative approach to supervised learning that aims to reduce the reliance on large human-annotated datasets (Liu et al., 2021). Unlike traditional supervised models, which estimate the probability of an output given an input text, prompt-based learning models estimate the probability of the text directly. This is achieved by applying prompt functions to modify the input text into various prompt templates with unfilled slots. By filling these slots, various Natural Language Processing (NLP) tasks can be completed, such as common-sense reasoning (Kojima et al., 2022), self-rationalization (Marasović et al., 2021), and text style transfer (Suzgun et al., 2022). The development of prompt-based methods has enabled zero-shot and few-shot learning in a variety of artificial intelligence domains (Ramesh et al., 2021; Yang et al., 2022; Sanghi et al., 2022). Recent research has also evaluated the capabilities of few-shot prompt-based learning for semantic parsing (Shin and Van Durme, 2022; Roy et al., 2022a; Drozdov et al., 2022). Our contribution extends the current research by investigating the effect of prompts comprising only a limited number of examples on the robustness of prompt-based semantic parsers.

Adversarial Robustness. Neural networks have achieved impressive performance across various domains. However, as demonstrated by Szegedy et al. (2014), neural models are vulnerable to adversarial examples. Adversarial attacks in NLP normally take on various forms, including character-level manipulations (Hosseini et al., 2017; Ebrahimi et al., 2018; Belinkov and Bisk, 2018; Gao et al., 2018; Eger et al., 2019; Boucher et al., 2022), sentence-level rewriting (Iyyer et al., 2018; Ribeiro et al., 2018; Zhao et al., 2018), and adversarial word substitutions (Alzantot et al., 2018; Liang et al., 2018; Zhang et al., 2019).

There has been an increasing interest in defending against adversarial attacks in large language models via adversarial training (Yi et al., 2021; Ross et al., 2022; Bartolo et al., 2021; Guo et al., 2021). Adversarial training involves incorporating adversarial examples in the training set, thus making the model robust to such attacks. However, adversarial training can sometimes negatively impact the generalization ability of the neural models (Raghunathan et al., 2019; Min et al., 2021).

3 Robustness Evaluation for Prompt-based Semantic Parsing

This section gives an overview of our evaluation framework, including the methods of constructing the evaluation corpora and the evaluation metrics to evaluate the robustness of the prompt-based semantic parser.

Linguistic Phenomenon	Samples (~~Strikethrough~~ = Original Text, **red** = Adversarial Perturbation)
Typo (Word-level)	**NL**: what can you ~~tell~~**te11** me about ~~the~~**th e** population of missouri
Substitution (Word-level)	**NL**: what ~~can~~**will** you tell me about ~~the~~**a** population of missouri
Paraphrase (Sent.-level)	**NL**: ~~what can you tell me about the population of missouri~~ **What information can you provide on Missouri's population?**

Table 1: **Examples from Robustness Evaluation Set**. We show 3 examples from `GeoQuery`. These examples are generated with three different perturbations, and they all can successfully change the predictions of CODEX.

3.1 Construction of the Evaluation Corpus

A robust prompt-based semantic parser should be able to parse both the utterances and their adversarial counterparts into correct LFs. As proposed by Huang et al. (2021), an adversary of an utterance for a semantic parser is defined as i) an utterance with the same semantic meanings as the original one given the human judgment and ii) an utterance on which the semantic parser cannot produce correct LF. Therefore, to evaluate the robustness of prompt-based semantic parsers, we craft the robustness evaluation sets by perturbing the original utterances in existing benchmark datasets with multiple adversarial perturbation methods. Such perturbations should not alter the semantics of the original utterances. Each example in a robustness evaluation set is a perturbed utterance paired with its ground-truth LF. Next, we introduce the details of each perturbation method and how we guarantee the perturbations do not change the semantics. Table 1 illustrates some meaning-preserved utterances after perturbation in the robustness evaluation set of `GeoQuery` based on different perturbation methods. More examples can be found in Appendix B.

3.1.1 Adversarial Perturbations

Following the principles as in Wang et al. (2021) to design adversarial attacks, we perform five word-level perturbations and two sentence-level perturbations to generate seven robustness evaluation sets for the standard evaluation set in each benchmark.

Word-level Perturbations.

- **Typo-based (TB)** uses TextBugger (Li et al., 2018) to replace two words in each utterance with the typos.
- **Random Deletion (RD)** randomly deletes two words in the utterance.
- **Random Swap (RS)** swaps the positions of two random words in each utterance.
- **Context-aware Substitution (CS)** leverages RoBERTa (Liu et al., 2019) to substitute two random words with their synonyms.
- **Context-aware Insertion (CI)** inserts two most probable words selected by RoBERTa at two random positions in each utterance.

Sentence-level Perturbations.

- **Rewriting-based (RB)** chooses Quillbot[1] (Fitria, 2021), a state-of-the-art (SOTA) commercial paraphrasing model, to rewrite the complete utterances. Quillbot has been demonstrated as an effective tool to paraphrase utterances in semantic parsing data (Shiri et al., 2022).
- **Distraction-based (DB)** appends interrogation statements to the end of each NL, inspired by StressTest (Naik et al., 2018). Specifically, we design the following interrogation statements: "who is who; what is what; when is when; which is which; where is where", in which the selected interrogative words are more likely to appear in the utterance.

3.1.2 Data Filtering

In order to ensure that the perturbed examples preserve the meaning of the original NL, we design a two-stage evaluation process:

Step1: We first generate 20 adversarial examples against the original NL for each perturbation method and choose the top 10 candidates ranked based on text similarity scores between the original and the perturbed ones, which are calculated by Sentence-BERT (Reimers and Gurevych, 2019).

---

[1] `https://www.quillbot.com/paraphrasing`

Step2: We engage human experts to select the best one among the 10 adversarial candidates produced in **Step1**.

3.2 Evaluation Metrics

Since the output LFs of the prompt-based language models may not follow the same naming convention (Shin et al., 2021; Shin and Van Durme, 2022) as the ground truth, previous string-based evaluation metrics, including BLEU (Papineni et al., 2002) and Exact Match (Poon and Domingos, 2009), are not suitable for prompt-based semantic parsers. Therefore, we follow Rajkumar et al. (2022) to report the *execution accuracy*, which is based solely on the execution correctness of the LFs on the test sets, for the purpose of robustness evaluation.

Following Huang et al. (2021), we report the experiment results with three variants of execution accuracy, namely *standard accuracy*, *perturbation accuracy* and *robust accuracy*:

- **Standard Accuracy** is measured on the standard (original) test sets.
- **Perturbation Accuracy** tests the performance of the model on perturbed test sets.
- **Robust Accuracy** is defined as $n/|R_{eval}|$. R_{eval} denotes a subset of the perturbed test sets, and n is the number of the utterances in R_{eval} that are parsed correctly. More specifically, R_{eval} consists of the examples whose counterparts before perturbation are parsed correctly. Intuitively, Robust Accuracy estimates the quantity of cases that a parser can successfully parse before perturbation but cannot do so after perturbation, and hence shows the robustness of the parsers against adversarial perturbation.

4 Improving Robustness of Prompt-based Semantic Parsers

Instead of predicting the LF conditioned on the input utterance, large language models such as CODEX could learn to solve a specific task by *in-context learning*. During in-context learning, the parser predicts the LF conditioned on a prompt which consists of a small list of utterance-LF pairs to demonstrate the semantic parsing task and, optionally, a table schema. To defend against adversarial attacks, one seminal approach is adversarial training. One of the most typical adversarial training methods augments the training data with adversarial examples, from which the machine learning model could learn robust features (Allen-Zhu and Li, 2022) by gradient descent. However, directly adapting conventional adversarial training is not suitable for in-context learning. First, the number of demonstration examples is limited due to the restriction on the maximum number of tokens for the pre-trained language model. As a result, we cannot include an arbitrary number of adversarial examples in the prompt, which might not include enough robust features. Second, in-context learning does not update the parameters of the language model. The model would not be optimized towards learning the robust features in the adversarial examples through gradient descent.

Given the difference, it is unclear whether in-context learning could improve the robustness of the parser as the conventional supervised training. In this paper, we conduct the first investigation on in-context learning for model robustness. More specifically, we examine the impact of variants of in-context learning and sampling methods on parser robustness.

4.1 Standard In-context Few-shot Learning

In our setting, given an input utterance $\boldsymbol{x}$, the pre-trained language model $P(\cdot;\theta)$ predicts the LF $\boldsymbol{y}'$ conditioned on the prompt, which consists of a set of demonstration examples $\mathcal{M} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$, and a table schema $\mathcal{T}$:

$$\boldsymbol{y}' = \arg\max_{\boldsymbol{y}\in\mathcal{Y}} P(\boldsymbol{y}|\boldsymbol{x}, \mathcal{M}, \mathcal{T}; \theta) \qquad (1)$$

For the *few-shot* setting, the number of demonstration examples N is limited by a budget size.

4.2 Adversarial In-context Few-shot Learning

In adversarial in-context learning, we include the perturbed adversarial examples, $\mathcal{M}_{adv}$, in the demonstration examples:

$$\boldsymbol{y}' = \arg\max_{\boldsymbol{y}\in\mathcal{Y}} P(\boldsymbol{y}|\boldsymbol{x}, \mathcal{M} \cup \mathcal{M}_{adv}, \mathcal{T}; \theta) \qquad (2)$$

4.3 In-context Few-shot Selection

Current in-context learning assumes there is an example pool from where they can select prompting examples. However, most of the works only randomly pick examples from the pools. We argue that the way to select the examples might deeply impact the robustness of the prompt-based semantic parser. Therefore, we examine various strategies to select in-context few-shot examples.

Random Sampling (Random). We randomly sample N utterances from the example pool.

Confidence-based Sampling (Confidence) (Duong et al., 2018). We score each utterance with the confidence of the parser on the predicted LF given the utterance and the table schema. Then we select the ones with the lowest parser confidence scores[2].

Diversity-based Sampling. Following Li et al. (2021), we partition the utterances in the utterance pool into N clusters with the K-means (Wu, 2012) clustering algorithm and select the example closest to the cluster centers. We measure the edit distance (Cluster-ED) (Wagner and Fischer, 1974), and Euclidean distances using utterance features of TF-IDF (Cluster-TF-IDF) (Anand and Jeffrey David, 2011), or Contextual Word Embedding (Cluster-CWE) encoded by Sentence-BERT (Reimers and Gurevych, 2019), between each pair of utterances for K-means.

Perplexity-based Sampling (Sen and Yilmaz, 2020). We score each utterance with the perplexity of GPT-2 on this utterance. Then we select the utterances with the highest (PPL. Asc) and lowest (PPL. Desc) perplexity scores, respectively.

5 Experiments

5.1 Setup

Evaluation Datasets. We evaluate the robustness of the prompt-based semantic parsers via the adversarial robustness sets built on top of the test sets of `GeoQuery` (Finegan-Dollak et al., 2018) and `Scholar` (Finegan-Dollak et al., 2018) with the proposed perturbation methods in Section 3. As in Finegan-Dollak et al. (2018), we choose the *query* splits of both `GeoQuery` and `Scholar`, where there is no LF template overlap among train, test, and dev sets.

Prompt-based Semantic Parser. We choose CODEX (Chen et al., 2021) as the representative prompt-based semantic parser for our evaluation. In recent studies, CODEX has performed comparably via in-context few-shot semantic parsing to the SOTA-supervised trained neural semantic parsers (Shin and Van Durme, 2022; Roy et al., 2022b; Drozdov et al., 2022) in terms of execution accuracy.

To examine the vulnerability of large prompt-based semantic parsers against adversarial examples, we choose the `code-davinci-002` version of CODEX as it is the most powerful variant among all CODEX models, with 175B parameters. In our experiments, we sample a maximum of 200 tokens from CODEX with the temperature set to 0, with the stop token to halt generation.

Prompts. In this work, we adopt the prompt design of `Create Table + Select X` as presented in Rajkumar et al. (2022), which has been shown to be effective for semantic parsing using *static prompting*[3].

The prompt for semantic parsing on CODEX consists of `CREATE TABLE` commands, including specifications for each table's columns, foreign key declarations, and the results of executing a `SELECT * FROM T LIMIT X` query on the tables via the column headers. As described in Section 4.3, we select NL-LF pairs as in-context few-shot examples from the *train* sets.

To guide the prompt-based semantic parser, we also include the textual instruction of "`Using valid SQLite, answer the following questions for the tables provided above.`" as proposed by Rajkumar et al. (2022).

5.2 Research Questions and Discussions

Our experimental results answer the following four research questions (RQs) related to the robustness of CODEX.

RQ1: How vulnerable is the prompt-based semantic parser to adversarial examples?

Settings. To answer RQ1, we evaluate the standard accuracy and perturbation accuracy of CODEX on `GeoQuery` and `Scholar` test sets through *zero-shot* learning.

Results. The zero-shot parsing performances of CODEX are shown in Table 2. Our first observa-

---

[2] The confidence scoring parser is a zero-shot model, meaning that there are no examples present in the prompt. It operates solely based on the input utterance, instructions, and schema provided within the prompt. Please see Section 5.1 for more information on the prompt structure.

[3] In contrast to the approach of Shin et al. (2021) which involves dynamically selecting few-shot examples from an example pool, we refer to static prompting as being performed with a fixed set of examples.

Category	Pert. Strategy	`GeoQuery`			`Scholar`		
		Pert. Acc.	Std. Acc.	Δ	Pert. Acc.	Std. Acc.	Δ
Word-level	TB	53.85	57.14	-3.29	11.35	12.21	-0.86
	RD	50.55		-6.59	10.52		-1.69
	RS	37.36		-19.78	8.31		-3.90
	CS	42.31		-14.83	8.40		-3.81
	CI	38.46		-18.68	8.31		-3.90
Sentence-level	RB	**31.87**		-25.27	**5.22**		-6.99
	DB	35.71		-21.43	7.88		-4.33

Table 2: Results of perturbation accuracy (Pert. Acc.) and standard accuracy (Std. Acc.) of zero-shot performance on `GeoQuery` and `Scholar`. The zero-shot prompt only contain the table information and initial semantic parsing instruction. Perturbation accuracy is calculated based on each perturbation method.

tion is that Codex is more vulnerable to sentence-level perturbations than to word-level perturbations, as indicated by the more significant performance gaps between standard and perturbed accuracies on the sentence-level perturbed test sets. Wang et al. (2021) observed that neural language models are vulnerable to human-crafted adversarial examples where there are complex linguistic phenomenons (e.g., coreference resolution, numerical reasoning, negation). We observe that the rewriting model trained on human paraphrase pairs also introduces such complex linguistic phenomenons.

With respect to the word-level perturbations, Codex is most robust to typo-based perturbations, which is surprising as Wang et al. (2021) shows typo-based perturbation is the most effective attack method for large language models like BERT (Devlin et al., 2019) in the evaluation of natural language understanding tasks. However, utterances with typos drop only 3% of the accuracy of Codex. Random Deletion is also less effective than the other word-level methods, consistent with the observations by Huang et al. (2021) on the fully-supervised semantic parsers. This phenomenon can be attributed to the fact that Random Deletion primarily makes minor modifications to the standard NL utterances, as this method often involves removing non-functional words such as articles, for example, “the” and “a.”

Although Codex is pre-trained on a considerably large dataset, it does not show robustness on the in-domain tasks. We conjecture that the reason is that zero-shot Codex has not yet learned any in-domain knowledge on `GeoQuery` or `Scholar`. So in RQ2, we would address whether in-domain examples would improve the robustness of Codex.

Takeaways. Zero-shot Codex is vulnerable to adversarial examples, especially sentence-level perturbation of utterances, rather than to word-level perturbations.

Pert. Strategy	5-shot	10-shot	20-shot	30-shot	40-shot	50-shot
TB	63.19	71.98	78.02	81.32	81.30	82.42
RD	59.34	64.29	71.43	71.98	70.33	75.27
RS	52.20	52.75	54.87	59.34	60.99	63.14
CS	54.40	56.04	60.44	63.19	65.93	67.03
CI	51.65	54.95	55.49	57.69	58.79	61.02
RB	44.51	47.80	49.45	50.55	54.23	57.27
DB	48.35	49.77	53.30	54.20	59.34	59.89
Avg. Pert. Acc.	53.38	56.80	57.50	59.49	64.42	66.58
Std. Acc.	66.48	74.37	79.67	81.87	83.52	84.62
Avg. Robust Acc.	75.67	77.28	78.74	80.44	82.07	83.22

Table 3: Few-shot performances on `GeoQuery`. We conduct {5, 10, 20, 30, 40, 50}-shot learning experiments. Average perturbation accuracy (Avg. Pert. Acc.) is the average score of execution accuracies on different perturbation sets. Average robust accuracy (Avg. Robust Acc.) is the average score of execution accuracies on the test sets perturbed by different perturbation methods.

Pert. Strategy	3-shot	5-shot	10-shot
TB	10.57	20.33	34.29
RD	12.09	25.27	31.43
RS	6.04	17.03	25.08
CS	9.34	18.13	26.03
CI	8.24	14.29	20.63
RB	3.30	8.43	18.10
DB	4.40	10.44	21.90
Avg. Pert. Acc.	7.71	16.27	25.35
Std. Acc.	14.29	23.08	40.32
Avg. Robust. Acc.	51.12	53.10	55.97

Table 4: Few-shot performances on `Scholar`. We conduct {3, 5, 10}-shot learning experiments.

RQ2: Does standard in-context few-shot learning improve the robustness?

Settings. We respectively select up to 50 and 10 examples from `GeoQuery` and `Scholar` train sets[4], with the random sampling strategy, to construct prompts for parsers. Then, for each few-shot

---

[4] Due to the larger size of table schema in `Scholar`, we could only include up to 10 examples in the prompt.

learning experiment, we measure standard accuracy, perturbation accuracy and robust accuracy on our various perturbed test sets.

Results. Tables 3 and 4 show the performance of standard in-context few-shot learning on the robustness evaluation sets perturbed by different methods. We observe that more standard examples in the prompt can evidently improve the robust accuracy of CODEX, which demonstrates the effectiveness of standard in-context few-shot learning in improving the robustness of semantic parsing. Although it performs slightly worse on the test sets perturbed by typo-based methods than the one perturbed by the random deletion in `GeoQuery`, we argue that this is due to the performance variance, which does not necessarily hurt the model robustness.

The performance gap between perturbation accuracy with standard accuracy is enlarged when the number of in-context shots increases. However, the robust accuracy grows slowly. This indicates that improving the generalization ability of the parser does not necessarily mean the improvement of the robustness. The trade-off between standard and robust accuracies is a long-standing problem in adversarial training. Raghunathan et al. (2019) shows that increasing the training sample size could eliminate such a trade-off. Our experiments demonstrate that in-context learning follows similar patterns as supervised adversarial training. It can be observed that both objectives can be improved with a limited number of examples when compared to the zero-shot parser. However, the extent of improvement varies.

Takeaways. With more standard in-context examples, few-shot CODEX can be guided to achieve better robustness and standard execution performance.

RQ3: Does adversarial in-context learning improve robustness?

Settings. In this work, we present the experimental results of CODEX on both `GeoQuery` and `Scholar` datasets, using 10 and 5 in-context examples, respectively. In order to assess the robustness of CODEX through adversarial in-context learning, we first augment the standard few-shot examples by incorporating examples whose utterances have been perturbed using various perturbation methods. Subsequently, for each set of augmented examples, we calculate the average robust accuracy of CODEX based on the average of the parser robust accuracies on all robustness evaluation sets.

Adv. L. Strategy	`GeoQuery`		`Scholar`	
	Avg. Robust Acc.	Std. Acc.	Avg. Robust Acc.	Std. Acc.
No Adv.	77.28	74.37	53.10	23.08
No Adv. (× 2)	78.74	79.67	55.97	40.32
TB	77.32	73.62	52.75	34.99
RD	77.40	73.64	53.11	33.65
RS	78.30	74.73	54.88	33.46
CS	78.05	74.47	53.12	34.86
CI	78.14	74.81	54.66	33.65
RB	78.47	75.51	56.58	35.85
DB	78.31	75.08	55.08	35.71

Table 5: The results of the average robust accuracy obtained through Adversarial In-context Learning (Adv. L. Strategy) with different types of perturbed few-shot examples. Additionally, we include results of applying the method with only standard examples (No Adv.), as well as with a doubled number of standard examples (No Adv. (× 2)).

Results. The experimental results of the various perturbation strategies applied to the in-context few-shot examples are presented in Table 5. While the approach of supervised adversarial training has been widely regarded as an effective means of enhancing the robustness of machine learning models, the results indicate that on both `GeoQuery` and `Scholar`, the robust accuracies are only marginally improved through the application of adversarial in-context learning. Previous studies (Raghunathan et al., 2019; Huang et al., 2021) have pointed out that supervised adversarial training can sometimes result in a decrease in standard accuracy, even as it improves robust accuracy. However, the results of adversarial in-context learning diverge from this trend, with significant improvement in standard accuracy, from 23% to more than 33%, observed on `Scholar`, while robust accuracy only experiences marginal improvement. These observations indicate that adversarial in-context learning represents a distinct approach from supervised adversarial training in terms of enhancing the robustness of the model. Furthermore, the results suggest that simply incorporating adversarial examples into the prompt has a limited impact on the robustness of parsers, in contrast to supervised adversarial training.

Of all the perturbation strategies analyzed, the results indicate that CODEX achieves the best performance in terms of both standard and robust accuracy through the application of RB adversarial in-context learning, but experiences the worst performance through TB adversarial in-context learning. The hypothesis is that RB produces utterances

Dataset	Metric	Confidence	Cluster-CWE	Cluster-ED	Cluster-TF-IDF	PPL. Asc	PPL. Desc	Random
GeoQuery	Avg. Robust Acc.	78.77	78.02	82.40	**85.10**	70.36	50.82	73.22
	Avg. Pert. Acc.	69.80	68.93	74.41	**77.74**	62.11	45.71	66.58
	Std. Acc.	74.73	78.41	81.32	**85.64**	73.14	70.76	74.18
Scholar	Avg. Robust Acc.	55.31	60.24	60.68	**62.61**	53.97	47.18	55.97
	Avg. Pert. Acc.	28.55	29.81	30.28	**31.44**	22.25	7.47	25.35
	Std. Acc.	37.99	41.49	42.19	**42.97**	35.93	34.89	36.91

Table 6: The performance of standard few-shot in-context learning using various sampling methods on the `GeoQuery` and `Scholar` datasets. The average robust accuracy, average perturbation accuracy, and standard accuracy are computed for each sampling method to assess their efficiency in this learning scenario.

Dataset	LC. Metric	Confidence	Cluster-CWE	Cluster-ED	Cluster-TF-IDF	PPL. Asc	PPL. Desc	Random
GeoQuery	TTR ↑	7.68	7.24	8.47	**10.26**	5.94	3.22	6.44
	Yule's I ↑	68.55	64.37	69.59	**71.49**	62.94	43.41	58.14
	MTLD ↑	12.44	12.19	13.37	**15.58**	11.32	8.16	10.41
Scholar	TTR ↑	28.18	29.91	31.40	**33.11**	21.15	14.17	25.67
	Yule's I ↑	198.15	207.11	223.76	**262.36**	177.37	102.17	193.31
	MTLD ↑	15.68	15.63	16.34	**19.49**	11.94	13.12	14.36

Table 7: Results of the language complexity of standard NLs sampled by different sampling strategies, measured by three lexical diversity (LC.) metrics. For the ease of readability and comparison, we multiply both TTR scores and Yule's I scores by 100.

with more complex linguistic features, resulting in enhanced standard and robust accuracy during in-context learning. To test this hypothesis, the number of standard examples is doubled (No Adv.×2) to match the size of the examples augmented with the adversarial examples. The results show that the robust and standard accuracies of CODEX are higher than those obtained through adversarial in-context learning, likely due to the greater diversity of linguistic variations in the doubled standard examples.

Takeaways. The robustness of few-shot CODEX can be marginally improved by adversarial in-context learning without significant negative impacts on standard performances.

RQ4: What is the impact of sampling techniques on the robustness of parsers that utilize standard in-context few-shot learning?

Settings. In order to compare the influence of different sampling methods on the robustness of the model, we vary standard in-context examples on `GeoQuery` and `Scholar` with all 7 strategies aforementioned in Section 4.3. We choose the 50-shot setting for `GeoQuery` and 10-shot setting for `Scholar`.

Results and Takeaways. We present standard accuracies in Table 6 when varying the sampling methods for the few-shot example selection. We first observe that different sampling strategies impact the robust and standard performance of the CODEX. Overall, the Cluster methods, which diversify the features of the selected utterances, perform better than the other sampling methods. On the other hand, PPL. Desc sampling method performs consistently poorly than the other sampling methods. In brief, we conclude that CODEX is sensitive to the few-shot example sampling strategies.

RQ4-Ablation: Why does CODEX react differently to various sampling strategies?

Settings. The findings of RQ 1 and RQ 3 indicate that linguistic complexity has an impact on the performance of CODEX. As a result, the results of RQ 4 may also be influenced by linguistic complexity. To further explore this correlation, three lexical diversity metrics, Type-Token Ratio (TTR) (Templin, 1957), Yule's I (Yule, 1944), and Measure of Textual Lexical Diversity (MTLD) (McCarthy, 2005), are adopted to measure the lexical diversity of the selected NLs from `GeoQuery` and `Scholar`. The TTR is defined as the ratio of the number of unique tokens, also known as types, to the total number of tokens. The TTR is used as an indicator of lexical diversity, with a higher TTR indicating higher lexical diversity. Yule's Characteristic Constant (Yule's K) is a measure of text consistency that considers vocabulary repetition. Yule's K and its inverse, Yule's I, are considered more robust to variations in text length than the Type-Token Ratio (TTR). MTLD is computed as the average number of words in a text required to maintain a specified TTR value.

Results and Takeaways. Table 7 presents the lexical diversity of each set of NLs sampled by different approaches. The diversity scores obtained from the three metrics align with the performance of CODEX as presented in Table 6. For instance, the three metrics indicate that the examples selected using the Cluster-TF-IDF method achieve higher lexical diversity compared to those selected through the other methods. Additionally, the Cluster-TF-IDF method also produces the highest results in terms of robust and standard accuracy for CODEX. Thus, it can be inferred that an increase in the lexical diversity of the few-shot examples leads to an improvement in the robust and standard accuracy of CODEX.

6 Conclusion

This study examines the robustness of semantic parsers in the context of prompt-based few-shot learning. To achieve this objective, robustness evaluation sets were curated to evaluate the robustness of the prompt-based semantic parser, CODEX. The research aims to identify methods to improve the robustness of CODEX. The results of our comprehensive experiments demonstrate that even the prompt-based semantic parser based on a large pre-trained language model is susceptible to adversarial attacks. Our findings also indicate that various forms of in-context learning can improve the robustness of the prompt-based semantic parser. It is believed that this research will serve as a catalyst for future studies on the robustness of prompt-based semantic parsing based on large language models.

Limitations

In this study, we examine the robustness of the prompt-based semantic parser, CODEX, by focusing on the impact of prompt design on its execution performance. However, there is a need for future research to investigate more alternative adversarial training strategies for prompt-based semantic parsers in order to advance this field. In addition, our focus is limited to text-to-SQL tasks, and we encourage further investigation into the robustness of semantic parsers across different datasets and LFs. Despite these limitations, we emphasize the importance of exploring more effective prompt design in order to enhance the robustness of prompt-based semantic parsers, including CODEX, which shows non-negotiable vulnerability.

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2022. Feature purification: How adversarial training performs robust deep learning. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 977–988. IEEE.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896.

Rajaraman Anand and Ullman Jeffrey David. 2011. *Mining of massive datasets*. Cambridge University Press.

Ankur Bapna, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *arXiv preprint arXiv:1707.02363*.

Max Bartolo, Tristan Thrush, Robin Jia, Sebastian Riedel, Pontus Stenetorp, and Douwe Kiela. 2021. Improving question answering model robustness with synthetic adversarial data generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8830–8848.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004. IEEE.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Andrew Drozdov, Nathanael Schärli, Ekin Akyuürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2022. Compositional semantic parsing with large language models. *arXiv preprint arXiv:2209.15003*.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip R Cohen, and Mark Johnson. 2018. Active learning for deep semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for*

Computational Linguistics (Volume 2: Short Papers), pages 43–48.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding nlp systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647.

Catherine Finegan-Dollak, Jonathan K Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-sql evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360.

Tira Nur Fitria. 2021. Quillbot as an online tool: Students' alternative in paraphrasing and rewriting of english writing. *Englisia: Journal of Language, Education, and Humanities*, 9(1):183–196.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. 2021. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5747–5757.

Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Deceiving google's perspective api built for detecting toxic comments. *arXiv preprint arXiv:1702.08138*.

Shuo Huang, Zhuang Li, Lizhen Qu, and Lei Pan. 2021. On robustness of neural semantic parsers. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3333–3342.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 963–973.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885.

Aishwarya Kamath and Rajarshi Das. 2018. A survey on semantic parsing. In *Automated Knowledge Base Construction (AKBC)*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *ICML 2022 Workshop on Knowledge Retrieval and Language Models*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Zhuang Li, Lizhen Qu, and Gholamreza Haffari. 2021. Total recall: a customized continual learning method for neural semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3816–3831.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *IJCAI*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ana Marasović, Iz Beltagy, Doug Downey, and Matthew E Peters. 2021. Few-shot self-rationalization with natural language prompts. *arXiv preprint arXiv:2111.08284*.

Philip M McCarthy. 2005. *An assessment of the range and usefulness of lexical diversity measures and the potential of the measure of textual, lexical diversity (MTLD)*. Ph.D. thesis, The University of Memphis.

Yifei Min, Lin Chen, and Amin Karbasi. 2021. The curious case of adversarially robust models: More data can help, double descend, or hurt generalization. In *Uncertainty in Artificial Intelligence*, pages 129–139. PMLR.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. *arXiv preprint arXiv:1806.00692*.

Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. 2019. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979. PMLR.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Xinyu Pi, Bing Wang, Yan Gao, Jiaqi Guo, Zhoujun Li, and Jian-Guang Lou. 2022. Towards robustness of text-to-sql models against natural and realistic adversarial table perturbation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2007–2022.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 1–10.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang. 2019. Adversarial training can hurt generalization. *arXiv preprint arXiv:1906.06032*.

Nitarshan Rajkumar, Raymond Li, and Dzmitry Bahdanau. 2022. Evaluating the text-to-sql capabilities of large language models. *arXiv preprint arXiv:2204.00498*.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Alexis Ross, Tongshuang Wu, Hao Peng, Matthew E Peters, and Matt Gardner. 2022. Tailor: Generating and perturbing text with semantic controls. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3194–3213.

Subhro Roy, Sam Thomson, Tongfei Chen, Richard Shin, Adam Pauls, Jason Eisner, and Benjamin Van Durme. 2022a. Benchclamp: A benchmark for evaluating language models on semantic parsing.

Subhro Roy, Sam Thomson, Tongfei Chen, Richard Shin, Adam Pauls, Jason Eisner, and Benjamin Van Durme. 2022b. Benchclamp: A benchmark for evaluating language models on semantic parsing. *arXiv preprint arXiv:2206.10668*.

Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. 2022. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613.

Priyanka Sen and Emine Yilmaz. 2020. Uncertainty and traffic-aware active learning for semantic parsing. In *Proceedings of the First Workshop on Interactive and Executable Semantic Parsing*, pages 12–17, Online. Association for Computational Linguistics.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32.

Ali Shafahi, Mahyar Najibi, Zheng Xu, John Dickerson, Larry S Davis, and Tom Goldstein. 2020. Universal adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5636–5643.

Richard Shin, Christopher Lin, Sam Thomson, Charles Chen Jr, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715.

Richard Shin and Benjamin Van Durme. 2022. Few-shot semantic parsing with language models trained on code. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5417–5425, Seattle, United States. Association for Computational Linguistics.

Fatemeh Shiri, Terry Yue Zhuo, Zhuang Li, Shirui Pan, Weiqing Wang, Reza Haffari, Yuan-Fang Li, and Van Nguyen. 2022. Paraphrasing techniques for maritime qa system. In *2022 25th International Conference on Information Fusion (FUSION)*, pages 1–8. IEEE.

Mirac Suzgun, Luke Melas-Kyriazi, and Dan Jurafsky. 2022. Prompt-and-rerank: A method for zero-shot and few-shot arbitrary textual style transfer with small language models. *arXiv preprint arXiv:2205.11503*.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Mildred C Templin. 1957. *Certain language skills in children: Their development and interrelationships*, volume 10. JSTOR.

Florian Tramer and Dan Boneh. 2019. Adversarial training and robustness for multiple perturbations. *Advances in Neural Information Processing Systems*, 32.

Robert A Wagner and Michael J Fischer. 1974. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Boxin Wang, Chejian Xu, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Hassan Awadallah, and Bo Li. 2021. Adversarial glue: A multi-task benchmark for robustness evaluation of language models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Junjie Wu. 2012. *Advances in K-means clustering: a data mining thinking*. Springer Science & Business Media.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. 2022. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3081–3089.

Mingyang Yi, Lu Hou, Jiacheng Sun, Lifeng Shang, Xin Jiang, Qun Liu, and Zhiming Ma. 2021. Improved ood generalization via adversarial training and pretraing. In *International Conference on Machine Learning*, pages 11987–11997. PMLR.

GU Yule. 1944. The statistical study of literary vocabulary. cambridge, cambridge [eng.].

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. 2019. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *International Conference on Learning Representations*.

A Experiments

Datasets. The `GeoQuery` dataset contains 877 pairs of NL-LF pairs about U.S. geographical information. On the other hand, `Scholar` contains pairs of NL-SQL regarding information about academic publications. Finegan-Dollak et al. (2018) proposed a dataset split for evaluating the compositional generalization capability of semantic parsers on several datasets, including `GeoQuery` and `Scholar`. The proposed split, referred to as the query-split, presents a more challenging scenario for semantic parsing models. This paper utilizes the query-split, where the two test sets in our experiments include 182 NL-LF pairs from `GeoQuery` and 315 NL-LF pairs from `Scholar`, respectively, during the evaluation of the prompt-based semantic parser.

Hyperparameters. We sample at most 200 tokens from CODEX with temperature 0, with the following strings used as stop tokens to halt generation: “--”, “\n\n”, “;”, “#”.

Model Versioning. The version of the `code-davinci-002` model referred to in this paper is as of the midpoint of the year 2022.

B Adversarial Examples

Table 8 lists the examples generated by all perturbation strategies.

Linguistic Phenomenon	Samples (~~Strikethrough~~ = Original Text, **red** = Adversarial Perturbation)
Typo (Word-level)	**NL**: what can you ~~tell~~**te11** me about ~~the~~**th e** population of missouri
Random Deletion (Word-level)	**NL**: ~~what~~ can you tell me ~~about~~ the population of missouri
Random Swap (Word-level)	**NL**: what can you tell me ~~about~~**missouri** the population of ~~missouri~~**about**
Context-aware Substitution (Word-level)	**NL**: what ~~can~~**will** you tell me about ~~the~~**a** population of missouri
Context-aware Insertion (Word-level)	**NL**: what **what** can you tell me about the **exact** population of missouri
Rewriting (Sent.-level)	**NL**: ~~what can you tell me about the population of missouri~~ **What information can you provide on Missouri’s population?**
Distraction (Sent.-level)	**NL**: what can you tell me about the population of missouri **who is who; what is what; when is when; which is which; where is where**

Table 8: **Examples from Robustness Evaluation Set**. The adversarial utterances in this Table are generated by applying various perturbation strategies to a single utterance “*what can you tell me about the population of missouri*” sampled from the `GeoQuery` dataset. All of the generated utterances can successfully alter Codex’s output.

Leveraging Task Dependency and Contrastive Learning for Case Outcome Classification on European Court of Human Rights Cases

Santosh T.Y.S.S, Marcel Perez San Blas, Phillip Kemper, Matthias Grabmair
School of Computation, Information, and Technology;
Technical University of Munich, Germany
`{santosh.tokala, marcel.perez, phillip.kemper, matthias.grabmair}@tum.de`

Abstract

We report on an experiment in case outcome classification on European Court of Human Rights cases where our model first learns to identify the convention articles allegedly violated by the state from case facts descriptions, and subsequently uses that information to classify whether the court finds a violation of those articles. We assess the dependency between these two tasks at the feature and outcome level. Furthermore, we leverage a hierarchical contrastive loss to pull together article-specific representations of cases at the higher level, leading to distinctive article clusters. The cases in each article cluster are further pulled closer based on their outcome, leading to sub-clusters of cases with similar outcomes. Our experiment results demonstrate that, given a static pre-trained encoder, our models produce a small but consistent improvement in classification performance over single-task and joint models without contrastive loss.

1 Introduction

The NLP task of classifying case outcome information from a textual statement of case facts is generally referred to as Legal Judgement Prediction (LJP)(e.g., Aletras et al. 2016; Chalkidis et al. 2019). It has been studied using corpora from different jurisdictions, such as the European Court of Human Rights (ECtHR) (Chalkidis et al., 2019, 2021, 2022b; Aletras et al., 2016; Medvedeva et al., 2020; Santosh et al., 2022, 2023), Chinese Criminal Courts (Luo et al., 2017; Zhong et al., 2018; Yue et al., 2021; Zhong et al., 2020; Yang et al., 2019), US Supreme Court (Katz et al., 2017; Kaufman et al., 2019), Indian Supreme Court (Malik et al., 2021; Shaikh et al., 2020), the French court of Cassation (Şulea et al., 2017), Brazilian courts (Bertalan and Ruiz, 2020), the Federal Supreme Court of Switzerland (Niklaus et al., 2021), the Turkish Constitutional court (Sert et al., 2021; Mumcuoğlu et al., 2021), UK courts (Strickson and De La Iglesia, 2020), German courts (Waltl et al., 2017), the Philippine Supreme court (Virtucio et al., 2018), and the Thailand Supreme Court (Kowsrihawat et al., 2018).

In this work, we experiment with classifying case outcomes in the ECtHR A and B benchmark tasks introduced by LexGLUE (Chalkidis et al., 2022b). Task B is to identify the set of articles of the European Convention of Human Rights (ECHR) that the claimant alleges to have been violated, while Task A's goal is to classify which of the convention's articles has been deemed violated by the court. The input for both tasks is the case's fact description that has been extracted from the published judgement document. It should be noted that, despite these tasks being typically referred to as instances of 'legal judgement prediction', the fact statements are typically not finalized until the decision outcome is known, making the task effectively one of retrospective classification rather than prediction (Medvedeva et al., 2021). While this does lead to distracting and confounding phenomena (see our prior work in Santosh et al. 2022), the dataset remains a useful resource for the development of NLP models that analyze these fact statements for text patterns that correspond to specific convention articles as drafted by the court. Consequently, we speak of *case outcome classification (COC)*.

Positive instances for Task A are cases in which an article was deemed violated by the court. Negative instances, however, usually encompass not only cases in which that particular article was alleged and considered not violated, but also the cases in which the particular article was not alleged in the first place. Given that the conditional probability of a positive task A label (violation) given its task B label (allegation) can be very high in the LexGLUE dataset, we posit that models can fall into this pitfall of identifying dominant articles

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1103–1111
May 2-6, 2023

with high conditional violation probability, missing information specific to violation classification. Thus, we experiment with multi-task architectures to decouple these reasoning steps involved in Task A . This is similar to work in Chinese criminal cases judgement prediction carried out in Zhong et al. 2018; Yang et al. 2019; Xu et al. 2020; Yue et al. 2021 that coordinated multiple outcome variables (law articles, criminal charges, and penalty terms).

In the ECtHR context, most of the previous works (Chalkidis et al., 2019, 2021, 2022b; Clavié and Alphonsus, 2021; Santosh et al., 2022, 2023) employed independent models for allegation identification and violation classification. Concurrent work by Valvoda et al. 2023 has recently explored multi-task joint models that learn Task A and Task B simultaneously. Their *Claim-Outcome model* decomposes them into two independent classifiers with separate encoders for each allegation and violation classification given allegation information. Our work differs in three ways: (i) We learn the joint representation for both tasks through a shared encoder following our intuition that the shared representation is beneficial as both tasks require similar features at the lower level; (ii) We model dependency from allegation to violation classification at both the allegation outcome and feature levels. Conditioning violation classification on the allegation outcome leads to higher performance, but we find that passing along feature level information yields additional improvement. This way, the violation branch will focus on identifying specific information required for determining violation than falling prey to only allegation level features. (iii) We also model correlations among different articles that tend to be concurrently alleged.

Inspired by recent advances in contrastive learning to learn effective representations (Khosla et al., 2020), we further devise a two-level hierarchical contrastive loss for COC. First, we strive to maximize the latent space distance between different article representations, which assists the model in learning distinct article-specific views of case facts. Second, we apply contrastive learning within each article latent space to form distinct sub-clusters of similar outcome cases. Unlike our two-level hierarchical contrastive learning, a single-level one has been explored for LJP in the concurrent work by Zhang et al. 2023.

Our experiments demonstrate that, given a static pre-trained encoder, our models outperform single-task and joint models without contrastive loss by a small but consistent margin, with larger improvements for sparse classes in classification performance.

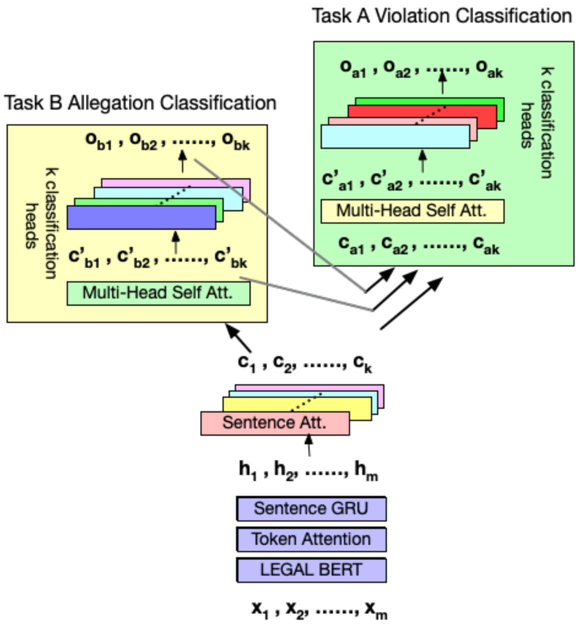


Figure 1: Our architecture capturing task dependency from allegation to violation classification branch.

2 Method

Our model takes as input the case fact description as token encodings $x = \{x_1, x_2, \ldots, x_m\}$ where $x_i = \{x_{i1}, x_{i2}, \ldots, x_{in}\}$ and outputs the set of articles claimed to be alleged by the applicant (Task B) and set of articles deemed to be violated by the court (Task A) as multi-hot vectors. x_i and x_{ip} denote i^{th} sentence and p^{th} token of i^{th} sentence of the case facts respectively. m and n denote the number of sentences and tokens in the i^{th} fact sentence, respectively. Our model is built on top of a hierarchical attention network (Yang et al., 2016), which we chose to be able to process the very long fact description inputs. Our model contains three main components: (1) encoding layer, (2) Article-Specific Case representation and (3) Task Dependency learning. See Fig. 1 for an overview of our model.

Encoding layer: We encode each sentence in the case facts description with LegalBERT (Chalkidis et al., 2020) to obtain the token level representations $\{z_{i1}, z_{i2}, \ldots, z_{in}\}$. Sentence level representations are then computed using a token-level attention mechanism as follows:

$$u_{it} = \tanh(W_w z_{it} + b_w) \;\; \& \;\; \alpha_{it} = \frac{\exp(u_{it} u_w)}{\sum_t \exp(u_{it} u_w)} \;\; \& \;\; f_i = \sum_{t=1}^{n} \alpha_{it} z_{it} \tag{1}$$

where W_w, b_w and u_w are trainable parameters. The sentence level representations $\{f_1, \ldots, f_m\}$ are passed through a GRU encoder to obtain context-aware representations of the case facts $h = \{h_1, h_2, \ldots, h_m\}$.

Article-Specific Case Representation: We disentangle the input case facts into multiple article-view-representations $c = \{c_1, c_2, \ldots, c_k\}$ where k is the total number of modeled convention articles, through aggregating context-aware sentence-level representations h using k sentence-level attention mechanisms similar to eq. 1 for every article individually. Through this article-specific attention mechanism, sentences relevant to a specific article are emphasized and intended to aid in fine-grained reasoning of outcome prediction for every article. This is helpful especially for sparser classes and mitigates the tendency of models to focus on skewed dominant articles. This is distinct from previous works (Chalkidis et al., 2019, 2021, 2022b; Santosh et al., 2022) which use a single vector representation for case facts.

Task Dependency Learning: To capture both the inter-article correlations for allegation, violation classification, and the inter-task dependency, we again use a two-step architecture. First, we apply a multi-head self-attention layer (Vaswani et al., 2017) to the obtained article-specific representations of case c to allow interactions among articles (e.g., some articles are typically alleged together while others usually occur in isolation) and obtain article-interaction-aware allegation feature representations $c'_b = \{c'_{b1}, c'_{b2}, \ldots, c'_{bk}\}$. These c'_b are passed through k article-specific Task B classification layers to obtain the binary outcome o_b corresponding to each article for allegation classification, which are then concatenated into a multi-hot vector.

Then we utilize the task B allegation information to enhance task A violation classification to capture the task dependency. We concatenate the obtained article-interaction-aware allegation representations c'_b and allegation label probability logits o'_b[1] with article-specific case representations c to obtain the enhanced article-specific representation for Task A as $c_{ai} = [c_i, c'_{bi}, o'_{bi}]$. We again employ multi-head self-attention mechanism on these enhanced article-aware representations c_a for violation prediction to capture the correlations that exist between violation of different articles. Finally, the obtained representations are passed through k article-specific Task A classification layers to obtain the binary outcome o_a corresponding to each article's violation output. To be able to evaluate the effectiveness of our interaction architecture in a clean way, we freeze the LegalBERT encoder weights in all our experiments.

2.1 Hierarchical Contrastive Loss

Contrastive learning has recently gained attention as a technique to obtain effective representations. In essence, it involves pulling together an 'anchor point' and its related samples while pushing it away from unrelated samples in the embedding space. Originally developed in self-supervised learning (Chen et al., 2020; Henaff, 2020), it has since been adopted in supervised settings (Khosla et al., 2020) where samples with the same/different labels are deemed related/unrelated with respect to an anchor.

In this work, we use a hierarchical contrastive loss (Liang et al., 2022) alongside the standard binary cross entropy loss on the task outcome probability. On the higher level, this is intended to form distinctive clusters of article-specific case representations. We hypothesize that this distinctiveness maximization constraint in turn helps the article-specific representation component to extract salient information with respect to each article more effectively. At the lower level, inside the latent space for each article, we further perform contrastive learning among cases based on their outcome for tasks A and B, respectively. This allows the positive outcome representations of cases under a specific article to stay closer and separate from the negative outcome cases, leading to formation of sub-clusters. We apply contrastive losses for Task A and Task B separately on the interaction-aware representations. It is calculated as the mean loss computed based on every article based representation as an anchor point. The loss for the interaction-aware representation of the j^{th} case specific to the i^{th} article for Task B $c'_{b_{ij}}$ as an anchor point is calculated as follows:

$$\begin{aligned} l(c'_{b_{ij}}) = &\log\Big(\sum_{f \in Q(i,j)} \frac{\exp(c'_{b_{ij}} \cdot f/\tau_a)}{\sum_{g \in P(i,j)} \exp(c'_{b_{ij}} \cdot g/\tau_a)}\Big) \\ &\times \alpha \sum_{f \in R(i,j)} \frac{\exp(c'_{b_{ij}} \cdot f/\tau_c)}{\sum_{g \in Q(i,j)} \exp(c'_{b_{ij}} \cdot g/\tau_c)}\Big) \end{aligned} \tag{2}$$

where $P(i,j) = \{c'_{b l_m} | l \neq i \;\; \& \;\; m \neq j\}$ (all rep-

---

[1] We use non-binarized response (i.e. probability logits) as it avoids the information loss that can occur in the probability space due to binarization.

resentations except the anchor point), $Q(i,j) = \{c'_{bl_m} | l = i \ \& \ m \neq j\}$ (all representations which share the same article as the anchor point), $R(i,j) = \{c'_{bl_m} | l = i \ \& \ y_{i_j} = y_{l_m} \ \& \ m \neq j\}$, y_{i_j} the denotes binary outcome label of case j with respect to article i (all representations which share the same article and outcome of that article as the anchor point). τ_a and τ_c are scalar temperature parameters that control the penalties on negative samples. The first term in the above equation 2 denotes the contrastive loss among article representations (i.e., for a given anchor point, positive pairs are obtained by article-aware representations of cases in the batch which share the same article with the anchor point; negative pairs are the remaining article representations). The second term contrasts the cases based on their task outcome within specific article representations. Contrastive learning has shown to be more effective with larger batch sizes (Radford et al., 2021; Chen et al., 2020). To account for smaller batch size due to computational constraints incurred with our hierarchical setup, we use a memory bank (Wu et al., 2018) which progressively reuses the representations from previous batches in computing the contrastive loss.

3 Experiments & Discussion

3.1 Dataset & Baseline Models

We experiment on the ECHR task A and B datasets of LexGLUE (Chalkidis et al., 2022b), which consist of 11k case fact descriptions chronologically split into training (2001–2016, 9k cases), validation (2016–2017, 1k cases), and test sets (2017-2019, 1k cases). The label set for both tasks include 10 prominent ECHR articles. Implementation details are given in Appendix A. We compare our models against the following baselines:

Task A/B only: We train single-task models only on task A and B labels, respectively. Their architecture is based on our model with contrastive loss, task dependency and article-specific representation components removed.

Multi-task: Using the same architecture, we also develop a model for both task A and B that is trained in classic multi-task fashion with separate classification heads without article-specific representations and task dependency components.

3.2 Performance Evaluation

Following Chalkidis et al. 2022b, we report micro-F1 (μ-F1) and macro-F1 (m-F1) scores across the 10 ECtHR articles contained in the dataset. We also report hard-macro-F1 (hm-F1) for Task A following Santosh et al. 2022, which is the mean F1-score computed for each article where cases with that article having been violated are considered as positive instances, and cases with that article being alleged but not found to have been violated as negative instances. This forms the most critical measure for violation classification as it conditions on allegation.

Table 1: Results on Task A and B. ‘feat.’, ‘lab,’. ‘Cont.’ indicate feature, label and contrastive, respectively

	Task B		**Task A**		
Model	**μ-F1**	**m-F1**	**μ-F1**	**m-F1**	**hm-F1**
Task B only	76.20	67.15	-	-	-
Task A only	-	-	68.42	56.26	54.14
Multi-task	78.17	69.16	69.29	58.05	55.57
Our Method	79.29	70.97	71.26	65.24	60.90
w/o feat. & lab.	78.93	71.45	70.07	59.14	57.09
w/o feat.	78.59	71.56	70.68	63.93	59.28
w/o label	79.09	71.38	70.32	64.12	61.70
only gold lab.	78.21	70.03	81.46	78.93	66.59
gold lab. + feat.	77.68	70.40	83.19	78.79	67.42
w/o outcome Cont.	78.42	69.48	69.86	60.84	57.62
w/o article Cont.	79.02	71.14	71.16	64.68	59.86
Task A Cont.	-	-	70.16	62.14	58.12
Task B Cont.	78.16	69.42	-	-	-

We compare the performance of our method with the baselines in the first four rows of Table 1. The multi-task method performs better than the individual tasks alone, validating the dependency between them. Our method performs better than the multi-task architecture, highlighting the effectiveness of our feature and label dependency components, as well as of the hierarchical contrastive loss. It scores higher by a small margin in Task B and Task A micro-F1 but achieves a larger benefit (>5%) in task A macro-F1 and hard macro-F1 metrics. This suggests that our model is of particular utility for sparser classes.

3.3 Analysis

3.3.1 Ablation on Task Dependency Learning

We create three ablation conditions by removing one interaction mechanism in each of them: (i) *w/o feature & label* (article-specific representation but with no concatenation with features or labels) (ii) *w/o feature* (article-specific representation concatenated with task B label as classified by the model) and (iii) *w/o label* (article-specific representation concatenated with task B features).

From the second section of Table 1, we observe that even the performance of *w/o feature and label* is better than *multi-task* model indicated by a small but consistent margin, indicating that article-specific case representation is a competitive component. Both *w/o feature*, *w/o label* models perform better than *w/o feature & label* in Task A, demonstrating the benefit of passing on task B model information explicitly. Between them, *w/o label* performs better in m-F1 and hm-F1 scores of Task A than *w/o feature* showing that providing the fine-grained representation of features from the allegation identification model is more useful than the predicted allegation labels only.

Further, to evaluate the impact of allegation identification performance on downstream violation classification, we conduct a control experiment in which we provide the actual gold allegation set of articles as input to the violation component in place of the predicted ones. We create two variants: *only gold labels* and *gold labels + features* (gold labels along with task B features). From the third section of Table 1, we observe that performance on Task A increases substantially in micro-F1 and macro-F1, which is intuitive as the model gets access to perfect allegation information. Interestingly, when adding task allegation feature information, we notice an additional small increase in hard macro-F1, indicating that adding feature information can have beneficial effect even in the presence of gold task B labels. These results form an upper bound of the benefit of using task dependency information from allegation branch to violation branch in our architecture, and the size of the performance gap motivates future work on accurate allegation classification.

3.3.2 Ablation Hierarchical Contrastive Loss

We carry out an ablation experiment for each component of our hierarchical contrastive loss: (i) disable article-level contrastive learning to obtain distinctive article representations ('w/o article contrastive') and (ii) disable outcome-based contrastive learning to separate cases based on outcome within each article cluster ('w/o outcome contrastive'). Intuitively, from the fourth section of Table 1, we observe that removing the outcome contrastive loss has the larger effect on performance, as it directly relates to the predicted label. The smaller but consistent drop in performance when removing article-level contrastive learning supports our hypothesis that maximizing the distinctiveness among article representations encourages the model to learn how to extract article-specific salient information from case facts.

Finally, to study the impact of our contrastive loss component alone, we evaluate single-task models applying our hierarchical contrastive loss referred to as 'Task A/B Contrastive'. From the last section of Table 1, we observe that they outperform their baseline counterparts in both tasks, but still stay behind our model.

4 Conclusion

We improve ECtHR article violation classification from fact statements by leveraging feature and label information from an allegation classification model. We also leveraged hierarchical contrastive loss to contrast between different article representations and case representations based on outcome with respect to a specific article. Given a static pretrained encoder, our models outperform a straightforward multi-task architecture by a small but consistent margin, with larger improvements for sparse classes. These results suggest that the tasks of allegation and violation classification on ECtHR fact statements interrelate in a way that may not be optimally captured using straightforward multi-task architectures, and motivate further research on dependency modeling between related legal classification tasks.

Limitations

In this work, we have demonstrated improvements in violation classification for ECtHR cases by leveraging the dependency from allegation classification at the feature and label level. While our feature transfer and contrastive learning techniques are general, our experimental contributions are contextual to the court. The nature of the fact statements as being influenced by the eventual case outcome and not suitable for prospective prediction has already been addressed in section 1.

While similar tasks (i.e., allegations paired with findings) exist in many other jurisdictions, they will differ in, for example, legal nature, semantic difficulty, and sub-task dependency. In our case, we choose the allegation classification as the auxiliary task which is closely related and also forms a necessary sub-part of deriving the final outcome related to the main task (violation prediction) at hand. We leave an exploration of the relatedness of auxiliary tasks and their impact on LJP/COC for future work.

One major challenge is dealing with long input case facts description, which is currently handled with hierarchical model in this work. These hierarchical models do not allow tokens in one sentence to attend to tokens in other sentence which leads to sub-optimal interaction modelling. This modeling impact on performance is still underexplored except some preliminary empirical work in Dai et al. 2022; Chalkidis et al. 2022a. Additionally, we freeze the weights in the LegalBERT encoder, both to save computational resources and to evaluate the effectiveness of our dependency mechanism in a clean way.

Ethics Statement

We employ pre-trained language models and do not train them from scratch, thus inheriting the biases they may have acquired from their training corpus. Our experiments have been carried out on a dataset of ECtHR decisions which is publicly available as the part of LexGLUE benchmark (Chalkidis et al., 2022b) and has been derived from the public court database HUDOC[2]. Though these decisions are not anonymized and contain the real names of the involved parties, we do not foresee any harm incurred by our experiments beyond making this information available. This collection of decision documents is of course historical data and using it to train model may result in classifiers that exhibit behavior that may be considered biased. For example, Chalkidis et al. 2022c explores disparities in classification performance with regard to an applicant's gender, their age, and the identity of the respondent state. If COC models are deployed as part of a decision support systems, then they of course must be screened for performance/error differences in between groups that are to be treated equally.

The task of LJP/COC in itself raises serious ethical and legal concerns, both in general and specific to the European Court of Human Rights. However, we do not advocate for the practical adoption of LJP/COC systems by courts. Our prior work in Santosh et al. 2022 demonstrates that these systems rely on several shallow surface-level spurious signals that are statistically predictive but legally irrelevant. This highlights the risk of using predictive systems in high stakes domains such as law. In the same work, we argue that models leveraging the case outcome signal for analytical purposes must be developed mindfully and with the goal of aligning their inferences with legal expert reasoning. This further parallels the broader legal NLP community increasingly addressing ethical aspects of developed systems in the context of technical research (e.g., Wang et al. 2021; Medvedeva et al. 2021, 2022; Tsarapatsanis and Aletras 2021; Leins et al. 2020).

In this work, we use COC as a technical benchmarking task that allows the development and study of neural NLP models on legal text. We focus on how to leverage dependencies on two successive tasks (allegation identification and violation classification) based on case facts, as well as on learning effective representations of these facts using contrastive learning. Our results are hence to be understood as technical contributions in pursuit of the overarching goal of developing models capable of deriving insight from data that can be used legally, ethically, and mindfully by experts in solving problems arising in legal research and practice.

All experiments were carried out using Google Colab. We did not track computation hours.

References

Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preoţiuc-Pietro, and Vasileios Lampos. 2016. Predicting judicial decisions of the european court of human rights: A natural language processing perspective. *PeerJ Computer Science*, 2:e93.

Vithor Gomes Ferreira Bertalan and Evandro Eduardo Seron Ruiz. 2020. Predicting judicial outcomes in the brazilian legal system using textual features. In *DHandNLP@ PROPOR*, pages 22–32.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in english. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323.

Ilias Chalkidis, Xiang Dai, Manos Fergadiotis, Prodromos Malakasiotis, and Desmond Elliott. 2022a. An exploration of hierarchical attention transformers for efficient long document classification. *arXiv preprint arXiv:2210.05529*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904.

Ilias Chalkidis, Manos Fergadiotis, Dimitrios Tsarapatsanis, Nikolaos Aletras, Ion Androutsopoulos, and Prodromos Malakasiotis. 2021. Paragraph-level rationale extraction through regularization: A case study on european court of human rights cases. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational*

---

[2] `https://hudoc.echr.coe.int`

Linguistics: Human Language Technologies, pages 226–241.

Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Katz, and Nikolaos Aletras. 2022b. Lexglue: A benchmark dataset for legal language understanding in english. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330.

Ilias Chalkidis, Tommaso Pasini, Sheng Zhang, Letizia Tomada, Sebastian Schwemer, and Anders Søgaard. 2022c. Fairlex: A multilingual benchmark for evaluating fairness in legal text processing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4389–4406.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Benjamin Clavié and Marc Alphonsus. 2021. The unreasonable effectiveness of the baseline: Discussing svms in legal text classification. In *Legal Knowledge and Information Systems*, pages 58–61. IOS Press.

Xiang Dai, Ilias Chalkidis, Sune Darkner, and Desmond Elliott. 2022. Revisiting transformer-based models for long document classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7212–7230, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Olivier Henaff. 2020. Data-efficient image recognition with contrastive predictive coding. In *International conference on machine learning*, pages 4182–4192. PMLR.

Daniel Martin Katz, Michael J Bommarito, and Josh Blackman. 2017. A general approach for predicting the behavior of the supreme court of the united states. *PloS one*, 12(4):e0174698.

Aaron Russell Kaufman, Peter Kraft, and Maya Sen. 2019. Improving supreme court forecasting using boosted decision trees. *Political Analysis*, 27(3):381–387.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Diederik P Kingma and Jimmy Ba. 2015. Adam: a method for stochastic optimization 3rd int. In *International Conference on Learning Representations*.

Kankawin Kowsrihawat, Peerapon Vateekul, and Prachya Boonkwan. 2018. Predicting judicial decisions of criminal cases from thai supreme court using bi-directional gru with attention mechanism. In *2018 5th Asian Conference on Defense Technology (ACDT)*, pages 50–55. IEEE.

Kobi Leins, Jey Han Lau, and Timothy Baldwin. 2020. Give me convenience and give her death: Who should decide what uses of nlp are appropriate, and on what basis? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2908–2913.

Bin Liang, Zixiao Chen, Lin Gui, Yulan He, Min Yang, and Ruifeng Xu. 2022. Zero-shot stance detection via contrastive learning. In *Proceedings of the ACM Web Conference 2022*, pages 2738–2747.

Bingfeng Luo, Yansong Feng, Jianbo Xu, Xiang Zhang, and Dongyan Zhao. 2017. Learning to predict charges for criminal cases with legal basis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2727–2736.

Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripabandhu Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. 2021. Ildc for cjpe: Indian legal documents corpus for court judgment prediction and explanation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4046–4062.

Masha Medvedeva, Ahmet Üstün, Xiao Xu, Michel Vols, and Martijn Wieling. 2021. Automatic judgement forecasting for pending applications of the european court of human rights. In *ASAIL/LegalAIIA@ ICAIL*.

Masha Medvedeva, Michel Vols, and Martijn Wieling. 2020. Using machine learning to predict decisions of the european court of human rights. *Artificial Intelligence and Law*, 28(2):237–266.

Masha Medvedeva, Martijn Wieling, and Michel Vols. 2022. Rethinking the field of automatic prediction of court decisions. *Artificial Intelligence and Law*, pages 1–18.

Emre Mumcuoğlu, Ceyhun E Öztürk, Haldun M Ozaktas, and Aykut Koç. 2021. Natural language processing in law: Prediction of outcomes in the higher courts of turkey. *Information Processing & Management*, 58(5):102684.

Joel Niklaus, Ilias Chalkidis, and Matthias Stürmer. 2021. Swiss-judgment-prediction: A multilingual legal judgment prediction benchmark. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 19–35.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

T. Y. S. S Santosh, Oana Ichim, and Matthias Grabmair. 2023. Zero shot transfer of article-aware legal outcome classification for european court of human rights cases. *arXiv preprint arXiv:2302.00609*.

T.y.s.s Santosh, Shanshan Xu, Oana Ichim, and Matthias Grabmair. 2022. Deconfounding legal judgment prediction for European court of human rights cases towards better alignment with experts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1120–1138, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mehmet Fatih Sert, Engin Yıldırım, and İrfan Haşlak. 2021. Using artificial intelligence to predict decisions of the turkish constitutional court. *Social Science Computer Review*, page 08944393211010398.

Rafe Athar Shaikh, Tirath Prasad Sahu, and Veena Anand. 2020. Predicting outcomes of legal cases based on legal factors using classifiers. *Procedia Computer Science*, 167:2393–2402.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Benjamin Strickson and Beatriz De La Iglesia. 2020. Legal judgement prediction for uk courts. In *Proceedings of the 2020 the 3rd international conference on information science and system*, pages 204–209.

Octavia-Maria Şulea, Marcos Zampieri, Mihaela Vela, and Josef van Genabith. 2017. Predicting the law area and decisions of french supreme court cases. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 716–722.

Dimitrios Tsarapatsanis and Nikolaos Aletras. 2021. On the ethical limits of natural language processing on legal text. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3590–3599.

Josef Valvoda, Ryan Cotterell, and Simone Teufel. 2023. On the role of negative precedent in legal outcome prediction. *Transactions of the Association for Computational Linguistics*, 11:34–48.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Michael Benedict L Virtucio, Jeffrey A Aborot, John Kevin C Abonita, Roxanne S Avinante, Rother Jay B Copino, Michelle P Neverida, Vanesa O Osiana, Elmer C Peramo, Joanna G Syjuco, and Glenn Brian A Tan. 2018. Predicting decisions of the philippine supreme court using natural language processing and machine learning. In *2018 IEEE 42nd annual computer software and applications conference (COMPSAC)*, volume 2, pages 130–135. IEEE.

Bernhard Waltl, Georg Bonczek, Elena Scepankova, Jörg Landthaler, and Florian Matthes. 2017. Predicting the outcome of appeal decisions in germany's tax law. In *International conference on electronic participation*, pages 89–99. Springer.

Yuzhong Wang, Chaojun Xiao, Shirong Ma, Haoxi Zhong, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2021. Equality before the law: Legal judgment consistency analysis for fairness. *arXiv e-prints*, pages arXiv–2103.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742.

Nuo Xu, Pinghui Wang, Long Chen, Li Pan, Xiaoyan Wang, and Junzhou Zhao. 2020. Distinguish confusing law articles for legal judgment prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3086–3095.

Wenmian Yang, Weijia Jia, Xiaojie Zhou, and Yutao Luo. 2019. Legal judgment prediction via multi-perspective bi-feedback network. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4085–4091.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Linan Yue, Qi Liu, Binbin Jin, Han Wu, Kai Zhang, Yanqing An, Mingyue Cheng, Biao Yin, and Dayong Wu. 2021. Neurjudge: a circumstance-aware neural framework for legal judgment prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 973–982.

Han Zhang, Zhicheng Dou, Yutao Zhu, and Ji-Rong Wen. 2023. Contrastive learning for legal judgment prediction. *ACM Transactions on Information Systems*.

Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. 2018. Legal judgment prediction via topological learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3540–3549.

Haoxi Zhong, Yuzhong Wang, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. Iteratively questioning and answering for interpretable legal judgment prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1250–1257.

A Implementation Details

Our models compute word embeddings of size 768. Our word level attention context vector size is 300. The sentence level GRU encoder dimension is 200, thus giving a bidirectional embedding of size 400, and a sentence level attention vector dimension of 200. The final dense classifier for all tasks has 100 hidden units. We use a mini batches size of 32 and the model is optimized end-to-end using Adam (Kingma and Ba, 2015). The dropout rate (Srivastava et al., 2014) in all layers is 0.1. We determine the best learning rate using grid search on the development set and use early stopping based on the development set m-F1 score. We finetuned τ_a, τ_c with an additional constraint of $\tau_a < \tau_c$ among the values of $\{0.07, 0.1, 0.14, 0.2, 0.25, 0.3\}$ so that it aids in pulling together the representations belonging to the same article in latent space (leading to distinct article clusters) and also in further slightly pulling together the representations of cases belonging to the same outcome in each separated article-specific embedding latent space compared to the other outcome cases in that same article. We set α to be 0.5. We use a memory bank of size 32 per article and outcome, and store only the most recent examples per article and its corresponding outcome.

Semi-supervised Relation Extraction via Data Augmentation and Consistency-training

Komal K. Teru
The Vanguard Group
`komal_teru@vanguard.com`

Abstract

Due to the semantic complexity of the Relation extraction (RE) task, obtaining high-quality human labelled data is an expensive and noisy process. To improve the sample efficiency of the models, semi-supervised learning (SSL) methods aim to leverage unlabelled data in addition to learning from limited labelled data points. Recently, strong data augmentation combined with consistency-based semi-supervised learning methods have advanced the state of the art in several SSL tasks. However, adapting these methods to the RE task has been challenging due to the difficulty of data augmentation for RE. In this work, we leverage the recent advances in controlled text generation to perform high quality data augmentation for the RE task. We further introduce small but significant changes to model architecture that allows for generation of more training data by interpolating different data points in their latent space. These data augmentations along with consistency training result in very competitive results for semi-supervised relation extraction on four benchmark datasets.

Input sentence: The **battle** led to **panic** on the frontier, and settlers in the surrounding counties fled.

Synonym-replacement: The struggle cause scare on the frontier, and settlers in the surrounding counties fly.

LM-based augmentation: The **battle** reduced **panic** on the frontier, and settlers in the surrounding counties relaxed.

Vanilla BT: The war caused **panic** at the border, and residents of the nearby counties fled.

Our constrained BT: The **battle** sparked **panic** at the border, with residents fleeing in surrounding counties.

Figure 1: Different data augmentation techniques applied to a sample datapoint from SemEval dataset. Existing methods replace the head/tail entities (highlited in red), change the original meaning or do not give very fluent paraphrases.

1 Introduction

Relation extraction is one of the essential components in constructing structured knowledge bases (Luan et al., 2018), performing interpretable question answering (Sun et al., 2021), improving web search, and many other information extraction pipelines. It aims to discover the semantic relation between a given head entity and tail entity based on the context in the input sentence. For example, given a sentence "*The battle led to panic on the frontier, and settlers in the surrounding counties fled.*", the goal is to extract the `Cause-Effect` relation between the head entity '**battle**' and the tail entity '**panic**'. The RE task requires a high level of language understanding and involves a significant level of semantic complexity (Bach and Badaskar, 2007). Due to this semantic complexity it often requires extensive and highly skilled human involvement to obtain good quality labelled data, making data collection an expensive and noisy process. Unsurprisingly, because of the same semantic complexity of the task, models typically require large amounts of labelled data to give production-ready performance.

A common strategy to improve the sample efficiency of machine learning models is semi-supervised learning methods which leverage easily accessible unlabelled data to improve the overall performance. While there are several paradigms of semi-supervised learning methods, consistency training based methods have advanced the state of the art in several SSL tasks (Ghosh and Thiery, 2021). These methods can typically reach performances that are comparable to their fully supervised counterparts while using only a fraction of labelled data points. Recently, strong data augmentation combined with consistency training algorithms have shown great success, even surpassing fully supervised models, in low-data settings of various

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1112–1124
May 2-6, 2023 ©2023 Association for Computational Linguistics

tasks (Xie et al., 2020). Adapting these methods to the task of relation extraction has been challenging due to the difficulty of data augmentation for RE task. This is because, in addition to the input sentence, each data point also consists of a head entity and a tail entity contained in the input sentence. Typical data augmentation techniques used in NLP such as back-translation, synonym-replacement, language-model based augmentation, etc. (Feng et al., 2021) can not be easily applied to such 'structured' input as they do not guarantee the integrity of either a) the entities in the input sentence or, b) the meaning of the input sentence itself. Figure 1 shows that using synonym-replacement and vanilla back-translation (BT) methods (Sugiyama and Yoshinaga, 2019) the entities themselves could be paraphrased or replaced. Matching the new and the old entities is a whole problem in itself. In the Language Model-based augmentation method (Anaby-Tavor et al., 2020), the semantic meaning of the input sentence changes altogether, which makes it difficult to employ consistency training.

Present work. In this work, we leverage the recent advances in controlled text generation to perform high quality data augmentation for the relation extraction task that not only keeps the meaning and the head/tail entities intact but also produces fluent and diverse data points. In particular, we modify back-translation to leverage lexically constrained decoding strategies (Post and Vilar, 2018; Hu et al., 2019) in order to obtain paraphrased sentences while retaining the head and the tail entities. We further propose novel modifications to the widely popular relation extraction model architecture, that allows for generation of more samples by interpolating different data points in their latent space, a trick that has been very successful in other domains and tasks (Berthelot et al., 2019; Chen et al., 2020b,a). Additionally, we leverage the entity types of the head and the tail entities, when available, in a way that effectively exploits the knowledge embedded in pre-trained language models. These data augmentations, when applied to unlabelled data, let us employ consistency training techniques to achieve very competitive results for semi-supervised relation extraction on four benchmark datasets. To the best of our knowledge, this is the first study to apply and show the merit of data augmentation and consistency training for semi-supervised relation extraction task.

2 Related work

Semi-supervised learning for NLP

Semi-supervised learning algorithms can be categorized into two broad classes–1) *self-ensembling* methods and 2) *self-training* methods.

***Self-ensembling* methods** leverage the smoothness and cluster/low-density assumptions of the latent space (Chapelle and Zien, 2005). They train the models to make consistent predictions under various kinds of perturbations to either a) the data (Miyato et al., 2019; Xie et al., 2020), or b) the model parameters themselves (Tarvainen and Valpola, 2017). The former methods are broadly referred to as consistency training methods and have resulted in state-of-the-art performances for several semi-supervised NLP tasks. Sachan et al. (2019) add adversarial noise to both labelled and unlabelled data and train models to make consistent predictions on the original and the corresponding noisy data-point. Many recent methods leverage large pre-trained language models for more advanced data augmentation techniques, like back-translation (Edunov et al., 2018), and further improve performance in the low-data regime (Xie et al., 2020). Recently, Chen et al. (2020b,a) adapted the Mixup algorithm (Zhang et al., 2018) as another form of data augmentation for textual data and show state-of-the-art performance on text classification and NER tasks. Due to the difficulty of data augmentation for relation extraction task (Figure 1), these methods have not been adapted for semi-supervised relation extraction (SSRE) task so far. In this work, we fill that gap and demonstrate the empirical success of consistency training for SSRE.

***Self-training* methods** are the oldest heuristic methods of iteratively expanding the labelled training set by including high-confidence *pseudo-labels* from the unlabelled data. All of the existing works on SSRE fall under this paradigm. These methods famously suffer from the confirmation bias problem where the incorrect predictions of the initially trained model affect the quality of pseudo-labels and eventually cause the label distribution to drift away from the true data distribution, resulting in a *semantic drift*. Lin et al. (2019) was one of the first works to address this by training two different models and augmenting the labelled set with the 'consensus' set, i.e., the data points which are labelled the same by both models. Several works have developed on this idea of improving the pseudo-label quality via various strategies like meta-learning

(Hu et al., 2021a) or reinforcement learning (Hu et al., 2021b). These set of methods constitute our baselines.

Data augmentation for NLP

In this work, we concentrate on two major classes of data augmentation techniques for NLP – sentence-level data augmentation and latent space augmentations. Sentence-level data augmentation techniques include back-translation (Edunov et al., 2018), language-model based augmentations (Anaby-Tavor et al., 2020), and word-replacement strategies (Zhang et al., 2015). We adapt the back-translation techniques to the RE task.

In latent space augmentations one generates more samples by interpolating between pairs of given data points in their latent space. This was originally introduced for image classification (Zhang et al., 2018; Verma et al., 2019; Yun et al., 2019) as a data augmentation and regularization method. Previous works have generalized this idea to the textual domain by proposing to interpolate in embedding space (Cheng et al., 2020) or the general latent space (Chen et al., 2020b,a) of textual data and applied the technique to NLP tasks such as text classification, machine translation, NER task and achieved significant improvements. We show that both these styles of augmentations can be effectively applied to improve performance on SSRE task.

3 Background

Task formulation. In this work, we focus on the sentence-level relation extraction task, i.e., given a *relation statement* $\mathbf{x} : (\mathrm{s}, e_h, e_t)$ consisting of a sentence, s, a head entity, e_h, and a tail entity, e_t (both the entities are mentioned in the given sentence s), the goal is to predict a relation $r \in \mathcal{R} \cup \{\mathrm{NA}\}$ between the head and the tail entity, where $\mathcal{R}$ is a pre-defined set of relations. If the sentence does not express any relation from the set $\mathcal{R}$ between the two entities, then the relation statement $\mathbf{x}$ is accordingly labelled NA.

This is typically done by learning a relation encoder model $\mathcal{F}_\theta : \mathbf{x} \mapsto \mathbf{h}_r$ that maps an input relation statement, $\mathbf{x}$, to a fixed length vector $\mathbf{h}_r$ that represents the relation expressed in s between e_h and e_t. This relation representation, $\mathbf{h}_r$, is then classified to a relation $r \in \mathcal{R} \cup \{\mathrm{NA}\}$ via an MLP classifier.

Base model architecture. Most recent methods for RE use a Transformer-based architecture (Devlin et al., 2019; Vaswani et al., 2017) for the relation encoder model, $\mathcal{F}_\theta$. To represent the head and tail entities in the input to the encoder, the widely accepted strategy is to augment the input sentence s with entity marker tokens–`[E1]`, `[/E1]`, `[E2]`, `[/E2]`–to mark the start and end of both entities. Concretely, an input sentence like "*Lebron James currently plays for LA Lakers team.*" when augmented with entity marker tokens becomes

`[E1]` **Lebron James** `[/E1]` currently plays for `[E2]` **LA Lakers** `[/E2]` team.

This modified text is input to the Transformer-based sequence encoder. Next, the encoder output representations[1] of the tokens `[E1]` and `[E2]` are concatenated to give the fixed length relation representation, $\mathbf{h}_r = [\mathbf{h}_{[E1]} \oplus \mathbf{h}_{[E2]}]$. This fixed length vector is in turn passed through an MLP classifier, $p_\phi(\mathbf{h}_r)$, to give a probability vector, $\mathbf{y}$, over the relation set $\mathcal{R} \cup \{\mathrm{NA}\}$.

4 Proposed approach

In our approach we build on the base model architecture described in §3 and introduce additional model design elements that are necessary to obtain an improved performance in semi-supervised relation extraction (SSRE) task.

We first describe the two data augmentation techniques we perform, and the model architectural changes we introduce that facilitate these augmentations. Then, we describe the training procedure we follow to leverage unlabelled data and achieve state-of-the-art performance on three out of four benchmark datasets for SSRE.

4.1 Constrained back-translation

Back-translation (Edunov et al., 2018) generates diverse and fluent augmentations while retaining the global semantics of the original input. Specifically, one translates a given text into an intermediate language, say, German, and translates it back to the source language, say English. Using different intermediate languages and temperature-based sampling results in a diverse set of paraphrases. Applying this back-translation technique in a vanilla fashion is not possible for RE task because one has little control over the retention of the head and tail entities (Figure 1). Thus, when translating back

---

[1] hidden state from the last layer of the Transformer model

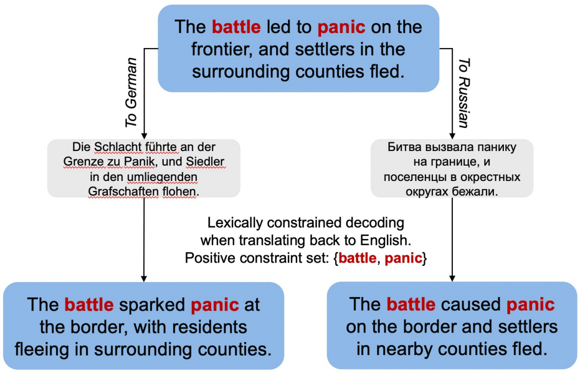


Figure 2: Constrained back-translation process

to the source language from the intermediate language we perform lexically-constrained decoding (Hu et al., 2019), i.e., force the inclusion of pre-specified words and phrases–positive constraint set–in the output. In our case the original head and tail entity words/phrases make up this positive constraints set. We use German and Russian as intermediate languages and use the pre-trained WMT'19 English-German and English-Russian translation models (in both directions) and their implementations provided by Ott et al. (2019). This methodology generates diverse data augmentations for a given sentence. For example, the sentence "*The battle led to panic on the frontier, and settlers in the surrounding counties fled.*" is converted to "*The battle sparked panic at the border, with residents fleeing in surrounding counties*" when back-translated via German, and to "*The battle caused panic on the border and settlers in nearby counties fled.*" when done via Russian. This is illustrated in Figure 2. This strong data-augmentation technique for RE can be applied to both labelled and unlabelled data opening the doors to consistency training (Xie et al., 2020) as we will see in §4.3.

4.2 Latent-space interpolation

Here, we adapt a mixup-based data augmentation technique to the RE task by making necessary modifications to the base model architecture we described in §3. As done in previous works (Chen et al., 2020b,a), we sample two random data points–$(\mathbf{x}, \mathbf{y})$ and $(\mathbf{x}', \mathbf{y}')$, where $\mathbf{x}$ and $\mathbf{y}$ denote the relation statement and the corresponding relation label–from the training data and separately compute the respective latent representations, $\mathbf{h}^m$ and $\mathbf{h}'^m$, upto the layer m of the relation encoder $\mathcal{F}_\theta$ as follows:

$$\mathbf{h}^l = \mathcal{F}_\theta^l(\mathbf{h}^{l-1}); \quad l \in [1, m],$$
$$\mathbf{h}'^l = \mathcal{F}_\theta^l(\mathbf{h}'^{l-1}); \quad l \in [1, m],$$

where $\mathbf{h}^l$ is the latent representation of all tokens in the sentence $\mathbf{x}$ at the l^{th} layer of the encoder. Next, the latent representations of each token in $\mathbf{x}$ at the m^{th} layer are linearly interpolated:

$$\tilde{\mathbf{h}}^m = \lambda \mathbf{h}^m + (1 - \lambda)\mathbf{h}'^m,$$

where λ is the mixing coefficient which is sampled from a Beta distribution, i.e., $\lambda \sim \text{Beta}(\alpha, \beta)$. Then, the interpolated latent representation is passed through the rest of the encoder layers:

$$\tilde{\mathbf{h}}^l = \mathcal{F}_\theta^l(\tilde{\mathbf{h}}^{l-1}); \quad l \in [m + 1, L].$$

This final encoder output representation, $\tilde{\mathbf{h}}^L$, can be interpreted as the encoder output representation of a *virtual* input $\tilde{\mathbf{x}}$, i.e., $\tilde{\mathbf{h}}^L = \mathcal{F}_\theta(\tilde{\mathbf{x}})$. We denote this whole mixup operation[2] as $\tilde{\mathbf{h}}^L := \text{REMix}(\mathbf{x}, \mathbf{x}')$. The label for this augmented *virtual* sample is given by the linear interpolation of the respective labels, $\mathbf{y}$ and $\mathbf{y}'$, with the same mixing coefficient λ i.e., $\tilde{\mathbf{y}} := \text{mix}(\mathbf{y}, \mathbf{y}') = \lambda \mathbf{y} + (1 - \lambda)\mathbf{y}'$. This *virtual* data point, $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$, is the augmented data point and can be used as additional training data.

Proposed architecture change. Now, for the RE task we need to extract a fixed-length relation representation from the encoder output representation of the entire input sequence. The traditional way to do this for RE task is by concatenating the encoder output representations of the entity marker tokens `[E1]` and `[E2]`. However, it is challenging to do this for a virtual sample, $\tilde{\mathbf{x}}$, as the entity markers are not clearly defined in this case. We thus modify the relation representation to be the encoder output representation of the `[CLS]` token. However, Baldini Soares et al. (2019) have shown this choice to be sub-optimal compared to concatenation of marker tokens. This is because the marker token representations provide direct access to the contextual information of the respective entities. Although the `[CLS]` token, in theory, has access to the entire context of the sentence, it might be difficult to capture the nuances like the head entity type, tail entity type, and the contextual information around the two entities all in a single vector.

---

[2] REMix entails the model architecture changes discussed below.

On the other hand, entity type information is easily accessible in most RE benchmarks[3]. So, to compensate for the sub-optimal choice of using `[CLS]` token representation as the relation representation, we modify how we represent the entity spans in the input token to more effectively use the easily accessible entity type information. In particular, we note that the entity type labels can trivially be mapped to tokens from any pre-trained language model's vocabulary. For example, entity types like `PERSON` and `STATE_OR_PROVINCE` can be tokenized into a word/phrase like 'person' and 'state or province', respectively. In such cases when entity type information is available, instead of using special marker tokens like `[E1]` and `[E2]` we prepend the entity spans in the input sequence with the word/phrases corresponding to their respective types and enclose these 'type-words' in punctuation marks (Zhou and Chen, 2021). The modified input to the transformer along with the `[CLS]` token looks as follows:

`[CLS]` @ * person * **Lebron James** @ plays
for & * organization * **LA Lakers** & team.

We use different punctuation symbols to distinguish between subject and object entities. Specifically, we use '@' for subject and '&' for object entities. This representation helps leverage the knowledge already contained in the pre-trained large-language model about the type of the entity and offset some of the downside of using a simplified relation representation in the `[CLS]` token. As we will empirically see in §5.4, this use of entity type information is not only effective but is necessary for the optimal functioning of our approach. Zhou and Chen (2021) recently showed the success of this method in the fully supervised setting. Here we use it in conjunction with a simplified relation representation and show its merit in semi-supervised RE setting.

4.3 Consistency training for SSRE

Let the given limited labelled set be $\mathbf{X}_l = \{\mathbf{x}^l_1, ..., \mathbf{x}^l_n\}$, with their relation labels $\mathbf{Y}_l = \{\mathbf{y}^l_1, ..., \mathbf{y}^l_n\}$, where $\mathbf{y}^l_i \in \{0,1\}^{|\mathcal{R} \cup \{\text{NA}\}|}$ is a one-hot vector and $\mathcal{R}$ is the set of pre-defined relations. Let $\mathbf{X}_u = \{\mathbf{x}^u_1, ..., \mathbf{x}^u_m\}$ be a large unlabelled set. The goal is to apply both the data augmentation techniques described above and train a model with consistency loss to effectively leverage unlabelled data along with the limited labelled data.

We largely adapt the semi-supervised training techniques introduced by Chen et al. (2020b). For each $\mathbf{x}^u_i$ in the unlabelled set $\mathbf{X}_u$, we generate K augmentations $\mathbf{x}^a_{i,k}, k \in \{1, 2, ..., K\}$ using the constrained back translation technique with different intermediate languages[4]. These augmentations make up the set $\mathbf{X}_a = \{\mathbf{x}^a_{i,k}\}$. For a given unlabelled data point $\mathbf{x}^u_i$ and its K augmentations $\mathbf{x}^a_{i,k}$ the label is given by the average of current model's predictions on all $K+1$ data points:

$$\mathbf{y}^u_i = \frac{1}{K+1}\left(p_\phi(\mathcal{F}_\theta(\mathbf{x}^u_i)) + \sum_{k=1}^{K} p_\phi(\mathcal{F}_\theta(\mathbf{x}^a_{i,k}))\right),$$

where $\mathbf{y}^u_i$ is a probability vector. This not only enforces the constraint that the model should make consistent predictions for different augmentations but also makes the predictions more robust by ensembling all the predictions. We merge the unlabelled set and the augmented set into $\mathbf{X}_{\text{ua}} = \mathbf{X}_u \cup \mathbf{X}_a$ and the corresponding (pseudo-)labels are given by $\mathbf{Y}_{\text{ua}} = \mathbf{Y}_u \cup \mathbf{Y}_a$, where $\mathbf{Y}_u = \{\mathbf{y}^u_i\}$, $\mathbf{Y}_a = \{\mathbf{y}^a_{i,k}\}$, and $\mathbf{y}^a_{i,k} = \mathbf{y}^u_i \forall k \in \{1, 2, ..., K\}$, i.e., all the augmented data points share the same label as the original unlabelled data point.

Given this cumulative set $\mathbf{X}_{\text{ua}}$ and their generated labels $\mathbf{Y}_{\text{ua}}$ as additional training data, we employ the REMix augmentation technique to generate arbitrary amounts of training data. In particular, we randomly sample two data points $\mathbf{x}^{\text{ua}}_s, \mathbf{x}^{\text{ua}}_t \in \mathbf{X}_{\text{ua}}$, and compute the encoder output representation of a new *virtual* data point with $\text{REMix}(\mathbf{x}^{\text{ua}}_s, \mathbf{x}^{\text{ua}}_t)$ and the corresponding target label with $\text{mix}(\mathbf{y}^{\text{ua}}_s, \mathbf{y}^{\text{ua}}_t)$.

Additionally, while computing the final unsupervised loss in each training iteration we filter out the unlabelled data points with prediction confidence below a certain threshold γ (Xie et al., 2020). Finally, to encourage low-entropy predictions on unlabelled data, we sharpen the predictions with a sharpening coefficient T:

$$\hat{\mathbf{y}}^{\text{ua}}_i = \frac{(\mathbf{y}^{\text{ua}}_i)^{\frac{1}{T}}}{||(\mathbf{y}^{\text{ua}}_i)^{\frac{1}{T}}||_1}.$$

Everything put together, the final unsupervised loss in each training iteration with mini-batch size B is

---

[3] From new datasets/applications viewpoint, when entities are identified in a piece of text it is safe to assume that their types would also be identified.

[4] In our specific implementation K = 2; with German and Russian as intermediate languages.

computed as:

$$\mathcal{L}_{\text{unsp}} = \frac{1}{B}\sum_{\mathbf{x}_s^{\text{ua}},\mathbf{x}_t^{\text{ua}}\sim\mathbf{X}_{\text{ua}}}^{B} \mathtt{m}(\mathbf{x}_s^{\text{ua}},\mathbf{x}_t^{\text{ua}})\mathcal{L}_{\text{mix}}(\mathbf{x}_s^{\text{ua}},\mathbf{x}_t^{\text{ua}}),$$

where

$$\begin{aligned}\mathcal{L}_{\text{mix}}(\mathbf{x}_s^{\text{ua}},\mathbf{x}_t^{\text{ua}}) &= \text{CE}(\text{mix}(\hat{\mathbf{y}}_s^{\text{ua}},\hat{\mathbf{y}}_t^{\text{ua}})|| \\ &\qquad p_\phi(\text{REMix}(\mathbf{x}_s^{\text{ua}},\mathbf{x}_t^{\text{ua}}))), \\ \mathtt{m}(\mathbf{x}_s^{\text{ua}},\mathbf{x}_t^{\text{ua}}) &= I(\max \mathbf{y}_s^{\text{ua}} > \gamma) I(\max \mathbf{y}_t^{\text{ua}} > \gamma).\end{aligned}$$

Here, $I(.)$ is an indicator function and $\mathtt{m}(.)$ denotes the confidence masking function which filters out the low-confidence datapoints. In our implementation $p_\phi(.)$ is a two-layer MLP classifier on top of the relation encoder model. CE denotes the cross entropy loss function. [5]

This combined with the traditional supervised loss, $\mathcal{L}_{\text{sup}} = \sum_{\mathbf{x}_i\sim\mathbf{X}_l}^{B} \text{CE}(\mathbf{y}_i^l || p_\phi(\mathcal{F}_\theta(\mathbf{x}_i)))$, constitutes the total loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sup}} + \gamma_m \mathcal{L}_{\text{unsp}},$$

where γ_m is a parameter which controls the trade-off between supervised and unsupervised loss.

5 Experiments

We perform experiments on four benchmark datasets for sentence-level RE and compare the proposed model, REMix, against current state-of-the-art SSRE approaches. We further conduct ablation studies and sensitivity analysis to expose the significance of different design choices of REMix.

5.1 Datasets

We use two widely popular relation extraction benchmark datasets: SemEval 2010 Task 8 (SemEval) (Hendrickx et al., 2010), and the TAC Relation Extraction Dataset (TACRED) (Zhang et al., 2017). SemEval is a standard benchmark dataset for evaluating relation extraction models containing 10717 examples in total. Each sentence is annotated with a pair of untyped nominals (concepts; example in Figure 1) that are related via one of 19 semantic relation types (including `no_relation`). TACRED is a large-scale crowd-sourced relation extraction dataset with 106264 examples which is collected from all the prior TAC KBP relation schema. Unlike SemEval, sentences in TACRED are labelled with pairs of typed-entities that are related via one of 42 person- and organization-oriented relation types (including `no_relation`). In addition to these standard benchmark datasets, we also show results on two more datasets: RE-TACRED (Stoica et al., 2021) and KBP37 (Zhang and Wang, 2015). RE-TACRED is a re-annotated version of the original TACRED dataset using an improved annotation strategy to ensure high-quality labels. Zhou and Chen (2021) provide a compelling analysis and recommend using this as the evaluation benchmark for sentence-level RE. KBP37 is another sentence-level RE dataset with 21046 total examples collected from 2010 and 2013 KBP documents as well as July 2013 dump of Wikipedia. In terms of size, this falls between SemEval and TACRED. Similar to SemEval the entity types are not available in this dataset, however the 37 relation types are person- and organization-oriented like in TACRED. This dataset is thus a good segue between the two standard benchmarks. The statistics of these datasets is given in Table 1.

Table 1: Dataset statistics

Dataset	# rel.	examples	# no_relation
TACRED	42	106264	79.51%
RE-TACRED	40	91467	63.17%
KBP37	37	21046	10.33%
SemEval	19	10717	17.39%

5.2 Baselines and implementation details

We compare REMix with three state-of-the-art models that are representative of the existing class of methods for SSRE: MRefG (Li et al., 2021), MetaSRE (Hu et al., 2021a), and GradLRE (Hu et al., 2021b). MRefG leverages the unlabelled data by semantically or lexically connecting them to labelled data by constructing reference graphs, such as entity reference or verb reference. This approach heavily leverages the linguistic structure of the data and is the only existing method that falls outside the *self-training* class of methods. MetaSRE generates pseudo labels on unlabelled data by learning from the mistakes of the classification model as an additional meta-objective. GradLRE on the other hand generates pseudo label data to imitate the gradient

---

[5] Note that we only apply the augmentation techniques on the unlabelled data set. Initial experiments applying these to the labelled data set resulted in only marginal improvements and even performance deterioration in some cases, likely due to introduction of too much noise into an already limited labelled set.

Table 2: F1 score with various amounts of labelled data and 50% unlabelled data. Mean and standard deviation of 5 different runs is reported. Best performance on each configuration is bolded and second best is underlined.

	TACRED			KBP37		
%labelled Data	3%	10%	15%	5%	10%	30%
MRefG	43.81± 1.44	55.42± 1.40	58.21± 0.71	-	-	-
MetaSRE	46.16± 0.74	56.95± 0.33	58.94± 0.31	59.29± 0.92	61.83± 0.21	63.51± 0.69
GradLRE	47.37± 0.74	58.20± 0.33	59.93± 0.31	59.98± 0.37	62.67± 0.54	**66.41**± 0.28
REMix(ours)	**55.80**± 1.33	**61.30**± 0.70	**63.07**± 0.93	**60.84**± 0.40	**63.82**± 0.71	**66.46**± 0.69

	RE-TACRED			SemEval		
%labelled Data	3%	10%	15%	5%	10%	30%
MRefG	-	-	-	75.48± 1.34	77.96± 0.90	83.24± 0.71
MetaSRE	44.42± 3.02	58.71± 1.70	61.71± 3.70	78.33± 0.92	80.09± 0.78	84.81± 0.44
GradLRE	61.22± 0.58	74.03± 1.74	**79.46**± 0.82	**79.65**± 0.68	**81.69**± 0.57	**85.52**± 0.34
REMix(ours)	**71.33**± 1.22	**77.94**± 0.59	**79.76**± 0.47	77.58± 0.59	81.13± 0.82	**85.51**± 0.38

descent direction on labelled data and bootstrap its optimization capability through trial and error (Hu et al., 2021b). MetaSRE and GradLRE are two of the strongest methods in the widely adapted *self-training* methods for SSRE.

Implementation details. We follow the established setting to use stratified sampling to divide the training set into various proportions of labelled and unlabelled sets so that the relation label distribution remains the same across all subsets. Following existing work, we sample 5%, 10%, and 30% of the training set as labelled data for the SemEval and KBP37 datasets, and 3%, 10%, and 15% of the training set as labelled data for TACRED and RE-TACRED datasets. For all datasets and experiments, unless otherwise specified, we sample 50% of the training set as the unlabelled set. For TACRED and SemEval datasets we take the performance numbers of all baseline models reported by Hu et al. (2021b). For other datasets, we re-run the models with their best configuration as provided in their respective implementations, when available. To be consistent with all the baselines we initialize the text encoder of REMix with the `bert-base-cased` model architecture and pre-trained weights. Full details of all the hyperparameters can be found in Appendix A.2.

5.3 Main Results

Table 2 shows F1 results of all baseline models and our proposed model, REMix, on the four datasets when leveraging various amounts of labelled data and 50% unlabelled data. We report the mean and standard deviation of 5 different runs (with different seeds) of training and testing. REMix gives state-of-the-art performance on 10 out of 12 different configurations across all four datasets. This reinforces the importance of consistency regularization beyond the currently popular self-training methods for SSRE. Interestingly, the performance gains are significantly higher for TACRED and RE-TACRED datasets–we see an average improvement of as much as 17% when trained on 3% labelled data. This can be attributed to the fact that entity type information is available for these datasets and entity type markers are very effective in exploiting the knowledge embedded in the pre-trained language models. We revisit this observation in our ablation studies (§5.4) where we concretely establish the benefits of using entity type markers.

5.4 Analysis and discussion

We first conduct experiments to empirically demonstrate the effectiveness of three components of our proposed model: i) data augmentation by latent space interpolation (Mix-DA), ii) data augmentation by constrained back-translation (BT-DA), and iii) entity type markers (ET). In Table 3, we report the mean F1 score of five different runs for different variations of our model by removing a certain combination of these components[6]. As can be seen from Table 3, each of these components contributes to the overall success of REMix. For contribution of just the Mix-DA: we compare i) row 1 v/s row

---

[6] We omit standard deviation values for brevity. See the full results and comparisons in Appendix B

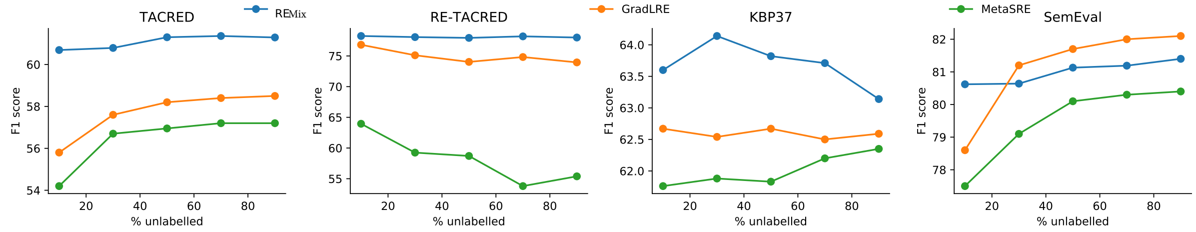


Figure 3: F1 Performance with various unlabelled data and 10% labelled data

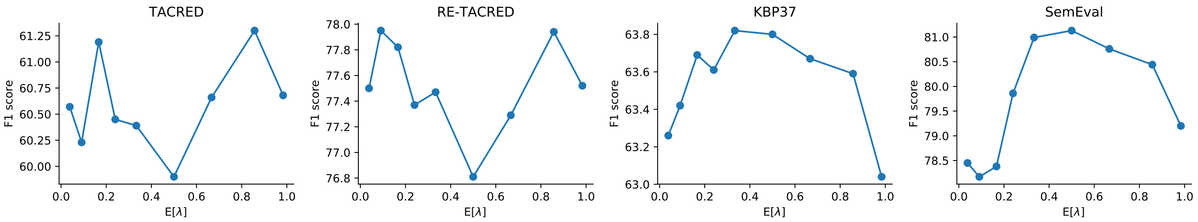


Figure 4: F1 Performance of REMix with 50% unlabelled and 10% labelled data with changing mixing coefficient λ

3, and ii) row 2 v/s row 4. All comparisons show positive improvement. For contribution of just the BT-DA: we compare i) row 1 v/s row 2, and ii) row 3 v/s row 4. We note that BT-DA results only in marginal improvements in most cases. Upon closer inspection we note that the constrained-decoding algorithms we implement for BT-DA are actually not perfect, especially when combined with translation models. It sometimes misses the constraints and sometimes falls into repetitive loops in an attempt to satisfy the constraint. With the ever-improving language generation capabilities, we believe the quality of data augmentation will only improve with time and result in more significant performance improvements. For contribution of both DA techniques together: we compare row 1 v/s row 4. All comparisons show significant improvements with data augmentation. The contribution of entity type markers can be noted in TACRED and RE-TACRED datasets. We see an average drop of 5.4% in F1 score across all 8 comparisons. Although our data augmentation techniques are effective, without the entity type information our method doesn't result in state-of-the-art performance. This reinforces our architectural choice to include entity type markers when using `[CLS]` token for relation representation (§4.2).

Table 3: Ablation results on all datasets using 10% labelled set and 50% unlabelled set.

Mix.	BT-DA	ET	TACRED	RE-T	KBP37*	SemEval*
✓	✓	✓	61.30	77.94	63.82	81.13
		✗	56.82	75.11		
✓	✗	✓	60.81	77.77	63.48	79.71
		✗	56.35	74.67		
✗	✓	✓	59.65	76.80	62.64	79.17
		✗	55.52	73.78		
✗	✗	✓	58.96	77.25	63.14	79.20
		✗	55.25	74.58		

* these datasets do not have entity type information

Next we examine the effect of using different amounts of unlabelled data. In Figure 3, we report the average F1 score for different models trained with different amounts of unlabelled data and 10% labelled data. REMix outperforms the baselines in all settings except on SemEval dataset, and, interestingly, the performance only marginally changes with the change in the amount of unlabelled data. Note that we train the models until the performance on the validation set stops improving for more than 5 epochs. Hence, REMix generates, in principle, an infinite amount of unlabelled data via the mixup strategy. Coupled with the fact that the label distribution remains the same in all settings, adding more unlabelled data does not seem to add a lot of new information. This explains why the model performance is relatively insensitive to changing amounts of unlabelled data. This also implies that REMix can leverage low amounts of unlabelled data better than the baselines.

Finally, in Figure 4 we show how the performance of REMix changes with a change in the mean of the Beta distribution from which λ is sampled on each iteration. Note that a value near 0 and 1 for λ means the augmented *virtual* data point will be closer to one of the underlying data points. As we get closer to 0.5 the virtual data points get further from the original data manifold and become more 'novel'. On TACRED and RE-TACRED datasets the performance peaks at $\mathrm{E}(\lambda) = 0.15(\text{or } 0.85)$ and drops in the mid-values. This can be interpreted as: adding datapoints far from the original data manifold is detrimental for these datasets. Interestingly, on KBP37 and SemEval the pattern inverts, i.e., the performance increases as $\mathrm{E}(\lambda)$ approaches 0.5, implying that more 'novel' augmentations help for these datasets.

6 Conclusion

In this paper, we propose a consistency-training-based semi-supervised algorithm for relation extraction and empirically show the merit of this class of methods in comparison to the current state-of-the-art *self-training* class of methods. In future work, one could bootstrap the self-training methods with consistency training as done in some previous works on vision tasks (Pham et al., 2021). Additionally, we show how the entity type information, when available, can result in massive performance boosts in the semi-supervised scenario. This is important because in most practical use cases when entities have already been identified, the entity type information is easy available and could be effectively leveraged in the proposed fashion.

7 Limitations

One of the key limitations of our proposed method compared to the baseline methods is the tight dependence on a strong external translation system to get good quality back-translated data augmentations. Secondly, since we use `[CLS]` token embeddings instead of entity-specific embeddings for final classification, it is more challenging to decipher entity-specific context. This is evident from the fact that our method performs relatively the weakest on the SemEval dataset which consists of untyped nominals (concepts) as entities and abstract relations which we believe need more entity-specific context to understand. Hence, our proposed method, REMix, shines bright when the entities in the dataset are typed or named entities whose meaning or type is relatively context-agnostic.

Acknowledgements

The author would like to thank the reviewers for their thoughtful feedback, the entire AI/ML Research team at Vanguard for their support, and especially David Landsman for their valuable input and discussion on the initial drafts of this work.

Disclaimer

This material is provided for informational purposes only and is not intended to be investment advice or a recommendation to take any particular investment action.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7383–7390.

Nguyen Bach and Sameer Badaskar. 2007. A review of relation extraction.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 32.

Olivier Chapelle and Alexander Zien. 2005. Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, volume R5 of *Proceedings of Machine Learning Research*, pages 57–64. PMLR.

Jiaao Chen, Zhenghui Wang, Ran Tian, Zichao Yang, and Diyi Yang. 2020a. Local additivity based data augmentation for semi-supervised NER. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1241–1251, Online. Association for Computational Linguistics.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020b. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2147–2157, Online. Association for Computational Linguistics.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. AdvAug: Robust adversarial augmentation for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5961–5970, Online. Association for Computational Linguistics.

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Atin Ghosh and Alexandre H. Thiery. 2021. On data-augmentation and consistency-based semi-supervised learning. In *International Conference on Learning Representations*.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38, Uppsala, Sweden. Association for Computational Linguistics.

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850, Minneapolis, Minnesota. Association for Computational Linguistics.

Xuming Hu, Chenwei Zhang, Fukun Ma, Chenyao Liu, Lijie Wen, and Philip S. Yu. 2021a. Semi-supervised relation extraction via incremental meta self-training. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 487–496, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xuming Hu, Chenwei Zhang, Yawen Yang, Xiaohe Li, Li Lin, Lijie Wen, and Philip S. Yu. 2021b. Gradient imitation reinforcement learning for low resource relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2737–2746, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Wanli Li, Tieyun Qian, Xu Chen, Kejian Tang, Shaohui Zhan, and Tao Zhan. 2021. Exploit a multi-head reference graph for semi-supervised relation extraction. In *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*, pages 1–7. IEEE.

Hongtao Lin, Jun Yan, Meng Qu, and Xiang Ren. 2019. Learning dual retrieval module for semi-supervised relation extraction. In *The World Wide Web Conference*, WWW '19, page 1073–1083, New York, NY, USA. Association for Computing Machinery.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Takeru Miyato, Shin ichi Maeda, Masanori Koyama, and Shin Ishii. 2019. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:1979–1993.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Hieu Pham, Qizhe Xie, Zihang Dai, and Quoc V. Le. 2021. Meta pseudo labels. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11552–11563.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Devendra Singh Sachan, Manzil Zaheer, and Ruslan Salakhutdinov. 2019. Revisiting lstm networks for semi-supervised text classification via mixed objective function. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6940–6948.

George Stoica, Emmanouil Antonios Platanios, and Barnabas Poczos. 2021. Re-tacred: Addressing shortcomings of the tacred dataset. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13843–13850.

Amane Sugiyama and Naoki Yoshinaga. 2019. Data augmentation using back-translation for context-aware neural machine translation. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 35–44, Hong Kong, China. Association for Computational Linguistics.

Haitian Sun, Patrick Verga, Bhuwan Dhingra, Ruslan Salakhutdinov, and William W Cohen. 2021. Reasoning over virtual knowledge bases with open predicate relations. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9966–9977. PMLR.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, volume 30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6438–6447. PMLR.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45.

Wenxuan Zhou and Muhao Chen. 2021. An improved baseline for sentence-level relation extraction. *ArXiv*, abs/2102.01373.

A Reproducibility checklist

A.1 Datasets

The sources of all the datasets are given in Table 4. We use the given train/validation/test splits for TACRED, RE-TACRED and KBP37 datasets. For SemEval dataset, we use the same splits as all the baselines, i.e., we split the original training set into 90% training set and 10% validation set.

A.2 Hyperparameters

We use the BERT tokenizer and set maximum sequence length to 256 to pre-process all datasets. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with 5e-5 learning rate and 0.1 warmup ratio. We sweep over the following hyperparameters: sharpening coefficient T, confidence threshold γ, the Beta-distribution parameters (α, β)[7], and the unsupervised loss weight γ_m. We perform incremental grid search to get the best performing configuration based on the F1 score on validation set. Table 5 shows the set of values we use for each parameter. Table 6 shows the best parameter values on each dataset and configuration. Following Chen et al. (2020b), we use {7, 9, 12} for the mixup layer set; this layer subset contains most of the syntactic and semantic information as suggested by Jawahar et al. (2019).

Table 5: Hyperparameter search values

Parameter	Values
T	{0.4, 0.6, 0.8, 1.0}
γ	{0, 0.15, 0.2, 0.25}
β	{1, 10, 30, 60, 120, 190, 300, 600}*
γ_m	{0.01, 0.1, 1}

* corresponding means of the sampled mixing coefficient, λ, are given by {0.04, 0.09, 0.17, 0.24, 0.33, 0.50, 0.67, 0.86, 0.98}

Table 6: Best hyperparameter values

	TACRED			KBP37		
	3%	10%	15%	5%	10%	30%
T	0.8	0.8	0.8	0.8	0.8	0.8
γ	0.15	0.15	0.90	0.25	0.25	0.70
β	10	10	10	120	120	190
γ_m	0.01	0.1	0.1	0.1	1.0	1.0
	RE-TACRED			**SemEval**		
	3%	10%	15%	5%	10%	30%
T	0.4	0.4	0.4	0.8	0.8	0.8
γ	0.0	0.0	0.9	0.2	0.2	0.7
β	10	10	10	60	60	60
γ_m	0.01	0.1	0.1	0.1	1.0	1.0

A.3 Training details

We train each model on a single NVIDIA Tesla T4 GPU with 16GB memory. We employ mixed precision training and gradient checkpointing techniques for faster and memory-efficient training. Note that we train the models until the performance on the validation set plateaus. The full REMix model roughly takes about 6 hours to train on TACRED, 5 hours in RE-TACRED, 1 hour in KBP37, and about 30 minutes on SemEval. Note that the training time slightly varies (± 30 minutes) depending on the percentage of labelled and unlabelled data we use. The number of parameters in all our models are largely dominated by the `bert-base-cased` that we use as the text encoder. The relatively negligible varying component is the MLP classifier that varies with the varying number of relations in each dataset.

B Full Ablation results

We conduct experiments to empirically demonstrate the effectiveness of three components of our proposed model: i) data augmentation by latent space interpolation, ii) data augmentation by constrained back-translation, and iii) entity type mark-

Table 4: Dataset sources

Dataset	Source
TACRED	`https://catalog.ldc.upenn.edu/LDC2018T24`
RE-TACRED	`https://github.com/gstoica27/Re-TACRED`
KBP37	`https://github.com/zhangdongxu/kbp37`
SemEval	`https://semeval2.fbk.eu/semeval2.php?location=data`

---

[7] α is fixed to be 60 and we change the values of β to control the mean of the distribution

Table 7: F1 score with 10% of labelled data and 50% unlabelled data. Mean and standard deviation of 5 different runs is reported.

	Mix.	BT aug.	Ent. type	TACRED	RE-TACRED	KBP37	SemEval
a)	✓	✓	✓	61.30± 0.70	77.94± 0.59	63.82± 0.71	81.13± 0.82
			✗	56.82± 0.64	75.11± 1.16		
b)	✓	✗	✓	60.81± 1.31	77.77± 0.96	63.48± 0.53	79.71± 0.83
			✗	56.35± 0.97	74.67± 1.04		
c)	✗	✓	✓	59.65± 0.92	76.80± 0.98	62.64± 0.69	79.17± 1.64
			✗	55.52± 0.89	73.78± 1.34		
d)	✗	✗	✓	58.96± 1.21	77.25± 0.70	63.14± 0.90	79.20± 0.32
			✗	55.25± 1.53	74.58± 0.91		

ers. In Table 7, we report the performance of different variations of our model by removing a certain combination of these components. As can be seen from Table 7, each of these components contributes to the overall success of REMix. To see the impact of data augmentation by latent space interpolation, we compare the results in row 'a' with the corresponding ones in row 'c', and similarly row 'b' with row 'd'. We see a significant and consistent drop in every comparison with and without that component. Specifically, we see a drop of an average 1.6% in F1 score over all the 12 comparisons. Another interesting pattern that stands out is the significant and consistent drop in performance (average 5.4% in F1 score across all 8 comparisons) when not using entity type markers. When we drop the entity type markers from the input representation, we use the basic entity start and end markers– `[E1]`, `[/E1]`, `[E2]`, `[/E2]`. This only applies to TACRED and RE-TACRED since only these datasets have entity type information readily available. Note that when we do not use any of the data augmentations, consistency training is not possible and we are effectively using only the labelled set for training the model (fourth row in Table 7). Surprisingly, even without any unlabelled data using the entity type markers alone gives better performance than the current state-of-the-art results by GradLRE (Hu et al., 2021b), i.e., an F1 score of 58.96 vs 58.20 on TACRED and 77.25 vs 74.03 on RE-TACRED. This proves the effectiveness of using entity type information in the fashion we use, i.e., by tokenizing the type words to leverage the knowledge embedded in pre-trained language models.

Event Temporal Relation Extraction with Bayesian Translational Model

Xingwei Tan[1], **Gabriele Pergola**[1], **Yulan He**[1,2,3]
[1]Department of Computer Science, University of Warwick, UK
[2]Department of Informatics, King's College London, UK
[3]The Alan Turing Institute, UK
`{Xingwei.Tan, Gabriele.Pergola.1}@warwick.ac.uk`
`yulan.he@kcl.ac.uk`

Abstract

Existing models to extract temporal relations between events lack a principled method to incorporate external knowledge. In this study, we introduce *Bayesian-Trans*, a Bayesian learning-based method that models the temporal relation representations as latent variables and infers their values via Bayesian inference and *translational functions*. Compared to conventional neural approaches, instead of performing point estimation to find the best set parameters, the proposed model infers the parameters' posterior distribution directly, enhancing the model's capability to encode and express uncertainty about the predictions. Experimental results on the three widely used datasets show that Bayesian-Trans outperforms existing approaches for event temporal relation extraction. We additionally present detailed analyses on uncertainty quantification, comparison of priors, and ablation studies, illustrating the benefits of the proposed approach.[1]

1 Introduction

Understanding events and how they evolve in time has been shown beneficial for natural language understanding (NLU) and for a growing number of related tasks (Cheng et al., 2013; Wang et al., 2018; Ning et al., 2020; Geva et al., 2021; Sun et al., 2022). However, events often form complex structures with each other through various temporal relations, which is challenging to track even for humans (Wang et al., 2020a).

One of the main difficulties is the wide variety of linguistic expressions of temporal relations across different contexts. Although many of them share some linguistic similarities, most of the topics in which they occur are characterized by some shared but unspoken knowledge that determines how temporal information is expressed. For example, when it comes to health, prevention is widely practised, with many treatments (e.g., vaccinations) being effective only if administered *before* the onset of a disorder. On the contrary, in the automotive industry, it is common that most people repair their car *after* a problem occurs. However, despite its simplicity, such commonsense knowledge is rarely stated explicitly in text and varies greatly across different domains. For example, in Figure 1, a detection model lacking the commonsense knowledge that *vaccination* can protect people from infection, tends to get confused by the complex linguistic structures in the excerpt and returns the wrong prediction entailing that '*died*' happens `after` '*vaccinated*'. Instead, with the consideration of prior temporal knowledge involving the *vaccination* event from an external knowledge source ATOMIC (Hwang et al., 2021), a model gives the correct prediction that '*died*' occurs `before` '*vaccinated*'.

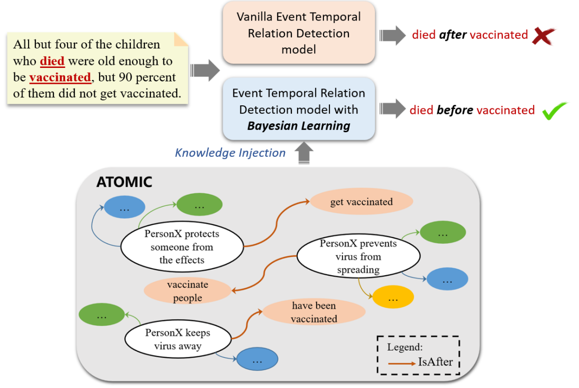


Figure 1: Comparison between with or without external knowledge incorporation on event relation extraction.

Methods proposed in recent studies for event relation extraction are mostly end-to-end neural architectures making rather limited use of such commonsense knowledge (Han et al., 2019a,b). Only a few works have explored the incorporation of external knowledge to mitigate the scarcity of event annotations (Ning et al., 2019; Wang et al., 2020b).

---

[1] Experimental source code is available at `https://github.com/Xingwei-Warwick/Bayesian-Trans`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1125–1138
May 2-6, 2023 ©2023 Association for Computational Linguistics

Nevertheless, these approaches typically update the event representations with knowledge features derived from external sources, lacking a principled way of updating models' beliefs in seeing more data in the domains of interests.

In this work, we posit that the Bayesian learning framework combined with translational models can provide a principled methodology to incorporate knowledge and mitigate the lack of annotated data for event temporal relations. Translational models, such as TransE (Bordes et al., 2013), are energy-based models based on the intuition that the relations between entities can be naturally represented by geometric translations in the embedding space. More concretely, a relation between a *head entity* and a *tail entity* holds if there exists a *translational* operation bringing the *head* close to the *tail* vector.

Specifically, we introduce a novel Bayesian Translational model (Bayesian-Trans) for event temporal relation extraction. Compared to conventional neural translational models, which only yield a point estimation of the network parameters, the Bayesian architecture can be seen as an ensemble of an infinite number of neural predictors, drawing samples from the posterior distribution of the translational parameters, refining its belief over the initial prior. As a result, event temporal relations are determined by the stochastic translational parameters drawn from posterior distributions. Additionally, such posteriors are conditioned upon the prior learned on external knowledge graphs, providing the commonsense knowledge required to interpret more accurately the temporal information across different contexts. As shown in the results obtained from the experimental evaluation on three commonly used datasets for event temporal relation extraction, the combination of translational models and Bayesian learning is particularly beneficial when tailored to the detection of event relations. Moreover, a favorable by-product of our Bayesian-Trans model is the inherent capability to express degrees of uncertainty, avoiding the overconfident predictions on out-of-distribution context. Our contributions are summarized in the following:

- We formulate a novel Bayesian translational model for the extraction of event temporal relations, in which event temporal relations are modeled through the stochastic translational parameters, considered as latent variables in Bayesian inference.
- We devise and explore 3 different priors under Bayesian framework to study how to effectively incorporate knowledge about events.
- We conduct thorough experimental evaluations on three benchmarking event temporal datasets and show that Bayesian-Trans achieves state-of-the-art performance on all of them. We also provide comprehensive analyses of multiple aspects of the proposed model.

2 Related Work

This work is related to at least three lines of research: event temporal relation detection, prior knowledge incorporation, and graph embedding.

2.1 Event Temporal Relation

Similar to entity-level relation extraction (Zeng et al., 2014; Peng et al., 2017), the latest event temporal relation extraction models are based on neural networks, but in order to learn from limited labeled data and capture complex event hierarchies, a wide range of optimization or regularization approaches have been explored. Ning et al. (2019) proposed an LSTM-based network and ensured global consistency of all the event relations in the documents by integer linear programming. Wang et al. (2020b) employed RoBERTa (Liu et al., 2019) and converted a set of predefined logic rules into differentiable objective functions to regularize the consistency of the relations inferred and explore multi-task joint training. Tan et al. (2021) proposed using hyperbolic-based methods to encode temporal information in a hyperbolic space, which has been shown to capture and model asymmetric temporal relations better than their Euclidean counterparts. Hwang et al. (2022) adopted instead a probabilistic box embeddings to extract asymmetric relations. Wen and Ji (2021) proposed to add an auxiliary task for relative time prediction of events described over an event timeline. Cao et al. (2021) developed a semi-supervised approach via an uncertainty-aware self-training framework, composing a training set of samples with actual and pseudo labels depending on the estimated uncertainty scores. None of the aforementioned approaches explored Bayesian learning for incorporating prior event temporal knowledge.

2.2 Incorporation of Prior Knowledge

Knowledge plays a key role in understanding event relations because people often skip inessential details and express event relations implicitly which is difficult to understand without relevant knowledge. For example, TEMPROB (Ning et al., 2018b) contains temporal relation probabilistic knowledge which is encoded by Siamese network and incorporated into neural models as additional features (Ning et al., 2019; Wang et al., 2020b; Tan et al., 2021). Unlike previous works, we combine the Bayesian Neural Network with distance-based models, treating the translational parameters as latent variables to be inferred. To this end, we adopt the variational inference (Kingma and Welling, 2014a; Blei et al., 2016; Gui et al., 2019; Pergola et al., 2021a; Zhu et al., 2022), and derive the prior distribution of the temporal relation information from commonsense knowledge bases (Pergola et al., 2021b; Lu et al., 2022). Christopoulou et al. (2021) explored a similar intuition of using knowledge base priors as distant supervision signals, but the approach and the task are different.

2.3 Graph Embedding Learning

Multi-relational data are commonly interpreted in terms of directed graphs with nodes and edges representing entities and their relations, respectively. Several works have recently focused on modelling these multi-relational data with relational embeddings by detecting and encoding local and global connectivity patterns between entities.

TransE (Bordes et al., 2013) has been a seminal work adopting geometric translations of entities to represent relations in the embedding space. If a relation between a head and a tail entity holds, it is encoded via the translational parameters learned at training time. However, TransE cannot model symmetry relation well by simple addition which led to several subsequent studies exploring diverse types of transformation resulting in a family of *translational models* (Wang et al., 2014; Ji et al., 2015; Lin et al., 2015). Among them, Balazevic et al. (2019) proposed to utilize the Poincaré model, mapping the entity embeddings onto a Poincaré ball, and using the Poincaré metric to compute the score function and predict their relations. Chami et al. (2020b) further expanded the idea of embedding learning over manifolds by additionally considering reflections and rotations and redefining the translation over a learned manifold.

Although translational models are shown efficient in modeling graph relation, they provide relatively limited interaction between nodes than neural network-based methods, such as Graph Neural Networks (Estrach et al., 2014; Chami et al., 2020a). Under this framework, nodes in a graph are neural units, which can iteratively propagate information through edges, and whose representations are learnt during the training process. In particular, Relational Graph Convolutional Networks (RGCN) (Schlichtkrull et al., 2018) encode relational data through link prediction and entity classification tasks, while enforcing sparsity via a parameter-sharing technique. Although modeling knowledge graphs has been one of the main focuses of the above-mentioned graph learning approaches, they lack any systematic mechanism to inject prior knowledge and update it during training.

3 Bayesian-Trans Model

In identifying temporal relations between events, we aim at predicting the relation type of two events given in text, commonly denoted as *head* event x_h and *tail* event x_t:

$$\hat{y} = \arg\max_{y \in \mathcal{R}} p(y|x_h, x_t) \tag{1}$$

where $\mathcal{R}$ denotes a set of possible relation types, while x_h and x_t the head and tail event triggers, respectively. Assuming that a set of latent variables $\mathbf{\Lambda}$ denotes the collection of all relation-specific transformation parameters $\mathbf{\Lambda}_r$. For example, in the knowledge embedding learning model such as MuRE (Balazevic et al., 2019), the head entity is first transformed through a relation-specific matrix $\mathbf{W}_r$, followed by a relation-specific translation vector t_r, then $\mathbf{\Lambda}_r = \{\mathbf{W}_r, t_r\}$. By Bayesian learning, the probability of inferring a relation type r can be written as:

$$p(y = r|x_h, x_t) = \int_{\mathbf{\Lambda}} p(y_r|x_h, x_t, \mathbf{\Lambda}) p(\mathbf{\Lambda}|\mathcal{G}) d_{\mathbf{\Lambda}} \tag{2}$$

Here, $p(\mathbf{\Lambda}|\mathcal{G})$ denotes the prior distribution of $\mathbf{\Lambda}$ derived from an existing knowledge graph encoded as $\mathcal{G}$. Directly inferring Eq. (2) is intractable. But we can resort to amortised variational inference to learn model parameters. In what follows, we present our proposed Bayesian learning framework built on translational models for event temporal relation extraction, called **Bayesian-Trans**, with its architecture shown in Figure 2.

In particular, the context S in which the two events occur is the input to our Bayesian-Trans. First, we encode S via a pre-trained language model generating the contextual embeddings e_h and e_t for the triggers of the head and tail events,

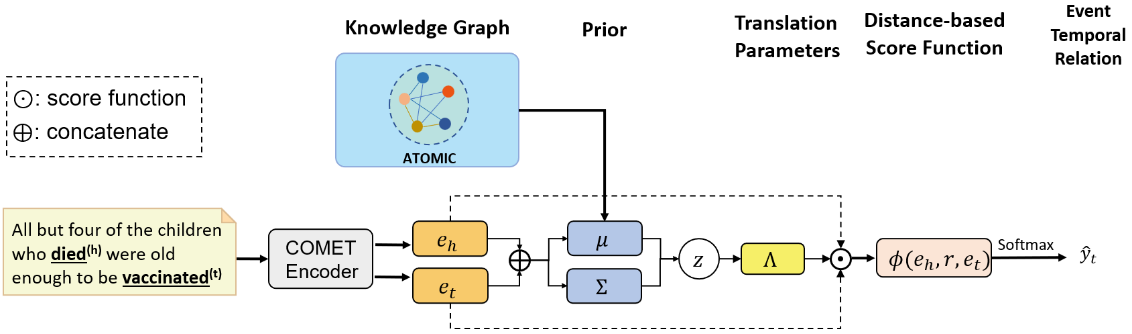


Figure 2: The network structure of Bayesian-Trans. Context sentences are first fed into a COMET encoder to generate event representations. With MLP layers, the event representations are mapped to generate a variational distribution of relation representations which is guided by KG priors. The relation representations are then used in the translational model to generate prediction scores.

respectively. The contextualised event trigger representations, e_h and e_t, are fed as input into a Bayesian translational module. This module, by means of variational inference, determines the parameters of the translational model, encoding the posterior distribution of the temporal relations conditioned upon the input events. Finally, we use a score function on the translated head and tail triggers to predict their temporal relation. We provide a more detailed description in the following.

3.1 Contextual Encoder

The proposed model uses COMET-BART (Hwang et al., 2021) as the context encoder. COMET-BART is a BART pre-trained language model (Lewis et al., 2020) fine-tunned on ATOMIC (Bosselut et al., 2019; Hwang et al., 2021), which is an event-centric knowledge graph encoding inferential knowledge about entities and events, including event temporal relations. The COMET-BART is able to generate consequence events given the antecedent event and a relation with good accuracy thus is regarded encodes knowledge well. Following the approach adopted in previous works (Ning et al., 2019; Wang et al., 2020b; Tan et al., 2021), we use the representation of the first token of an event trigger as the contextual embedding of that event[2], $\mathrm{e}_h, \mathrm{e}_t = \text{COMET-BART}(x_h, x_t)$, where $\mathrm{e}_h, \mathrm{e}_t \in \mathbb{R}^d$. The event representations are then concatenated together and fed through MLPs to generate the parameters of the variational distribution, from which the latent event-pair representation z is sampled. z is then mapped to the parameter space of the translational model as $\mathbf{\Lambda}$.

3.2 Incorporating Knowledge via Bayesian Learning

The proposed model utilizes relation embeddings for classifying event relation in a similar manner as the translational models in knowledge graph embedding, such as TransE (Bordes et al., 2013). If the embedding of the tail event is close enough to the embedding of head event after applying a series of relation-specific transformation, the relation stands, and vice versa. A wide range of translational models typically proposed for learning knowledge graph embeddings can be adopted in the proposed Bayesian-Trans. Additionally, to incorporate prior knowledge, we extend translational models to operate within the Bayesian inference framework. We proceed with introducing a standard translational model in the context of temporal relations, and describe how we extend it to work in the Bayesian framework.

Translational Model Generally speaking, a translational model uses *relation representations* $\mathbf{\Lambda}_r$ to perform "translation" for relation r on the head and tail events. Then, the transformed head and tail event embeddings are compared using a *distance-based score function*, whose score is indicative of the temporal relation between the events. The score function $\phi(\cdot)$ takes the general form:

$$\phi(\mathrm{e}_h, r, \mathrm{e}_t) = -d(\mathcal{T}^h_{\mathbf{\Lambda}_r}(\mathrm{e}_h), \mathcal{T}^t_{\mathbf{\Lambda}_r}(\mathrm{e}_t)) \qquad (3)$$

where r is a relation type, $\mathcal{T}_{\mathbf{\Lambda}_r}(\cdot)$ is a function depending on the parameters $\mathbf{\Lambda}_r$ of relation r to transform the event embeddings e_h and e_t, and $d(\cdot)$ is any distance metrics (e.g., Euclidean distance).

---

[2] We conducted some exploratory experiments adopting the last token or the average representation, but results showed that the first token was still the best option in this context.

We explored several models with different translation functions and distance metrics in the context of temporal relations, including TransE (Bordes et al., 2013), AttH (Chami et al., 2020b), MuRE (Balazevic et al., 2019) and MuRP (Balazevic et al., 2019), and based on our preliminary results[3], we eventually adopted MuRE as it strikes a good balance of training efficiency and accuracy of temporal relation classification. We define the scoring function in the proposed model as follows:

$$\phi(\mathrm{e}_h, r, \mathrm{e}_t) = -\|\mathbf{W}_r \mathrm{e}_h + \mathrm{t}_r - \mathrm{e}_t\|_2^2 \tag{4}$$

where $\mathbf{W}_r \in \mathbb{R}^{d \times d}$ is a diagonal relation matrix and $t_r \in \mathbb{R}^d$ a translation vector of relation r, $\mathbf{\Lambda}_r = \{\mathbf{W}_r, \mathrm{t}_r\}, r \in \mathcal{R}$.

Although the number of parameters to train is rather low, the number of annotated samples is usually small compared to the wide range of linguistic expressions capturing temporal relations. We thus extend the MuRE model into a Bayesian framework to enhance its scalability by treating the translational parameters $\mathbf{\Lambda}$ as latent variables. The proposed framework enhances generalization by defining a variational inference process that optimizes the regularization and leverages the additional information injected via the prior distributions.

Bayesian Inference As shown in the inference equation 2, the prior is derived from an external knowledge graph, such as ATOMIC, as a means to inject prior information about events and temporal relations. In particular, $\mathbf{\Lambda}$ is assumed to follow a Gaussian distribution with unit variance and with mean determined by the relation representations trained on the knowledge graph. The probability function is formulated as a softmax function over a pre-defined scoring function:

$$p(y_r|\mathrm{e}_h, \mathrm{e}_t, \mathbf{\Lambda}) = \frac{\exp\Big(\phi(\mathrm{e}_h, r, \mathrm{e}_t)\Big)}{\sum_{r' \in \mathcal{R}} \exp\Big(\phi(\mathrm{e}_h, r', \mathrm{e}_t)\Big)} \tag{5}$$

with e_h and e_t denoting the embedding for the head and the tail events, respectively.

Yet, Eq. (2) is intractable and cannot be inferred directly. Thus, we resort to amortized variational inference by introducing a variational posterior $q_\theta(\mathbf{\Lambda}|x_h, x_t)$, which follows the isotropic Gaussian distribution and can be modeled as:

$$\begin{gathered} \mu = f_\mu(\mathrm{e}_h; \mathrm{e}_t) \quad \Sigma = \mathrm{diag}\Big(f_\Sigma(\mathrm{e}_h; \mathrm{e}_t)\Big) \\ q_\theta(\mathbf{\Lambda}|\mathrm{e}_h, \mathrm{e}_t) = \mathcal{N}(\mathbf{\Lambda}|\mu, \Sigma), \end{gathered} \tag{6}$$

where f_μ and f_Σ are both fully connected layers that map the event pair representation into the parameters of the variational distribution.

Following the amortized variational inference, we maximize the evidence lower bound (ELBO) $\mathcal{L}_e$, defined in Eq. (7), and approximated by a Monte Carlo estimation with sample size N, as described in Eq. (8):

$$\begin{aligned} \mathcal{L}_e = \mathbb{E}_{q_\theta(\mathbf{\Lambda}|x_h, x_t), \{x_h, x_t\} \in \mathcal{D}} \Big[\log p_\theta(y|x_h, x_t, \mathbf{\Lambda})\Big] - \\ \mathrm{Reg}\Big(q_\theta(\mathbf{\Lambda}|x_h, x_t, \mathcal{G}) || p(\mathbf{\Lambda}|\mathcal{G})\Big) \quad (7) \\ \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{\{x_h, x_t\} \in \mathcal{D}} \Big[\log p_\theta(y|x_h, x_t, \mathbf{\Lambda}^{(n)}) - \\ \mathrm{Reg}\Big(q_\theta(\mathbf{\Lambda}^{(n)}|x_h, x_t, \mathcal{G}) || p(\mathbf{\Lambda}^{(n)}|\mathcal{G})\Big)\Big] \quad (8) \end{aligned}$$

where $\mathrm{Reg}(\cdot)$ is a regularization term which will be discussed in 3.3. To train end-to-end a fully differentiable model, we adopt the reparameterization trick (Kingma and Welling, 2014b).

3.3 Prior Distribution and Regularization

We proceed to discuss how the Bayesian framework enabled the incorporation of prior acquired from an external knowledge source. Then, we provide the details of how we compute the regularization term to induce a more stable training.

Prior Distribution One of the main advantages of the Bayesian inference framework is the possibility to inject commonsense knowledge into the model through the prior distribution of the latent variables, i.e., $p(\mathbf{\Lambda}|\mathcal{G})$ in Eq. (2), where $\mathbf{\Lambda}$ are the translational parameters and $\mathcal{G}$ denotes an external knowledge graph, in our case, the ATOMIC knowledge graph (Hwang et al., 2021). ATOMIC is a commonsense knowledge graph containing inferential knowledge tuples about entities and events encoding social and physical aspects of human everyday experiences. For our task of event temporal relation extraction, we are only interested in the events linked via temporal relations, such as 'ISBEFORE' (23,208 triples) or 'ISAFTER' (22,453 triples). By conducting link prediction on these links, we use relation embeddings learnt using an RGCN (Schlichtkrull et al., 2018) as the mean of the prior distribution for the translational latent variables. For the relations in the experiment

---

[3] Experimental results using different translational models are shown in Table A1.

dataset that do not have applicable counterparts in ATOMIC (e.g., VAGUE), we set their priors to standard Gaussian. The variance of the priors is defined as the identity matrix.

Specifically, we use COMET-BART to encode the event nodes from ATOMIC, then use their context embeddings as the node features in the RGCN. In our preliminary experiment, we also found that RGCN cannot train well on the commonsense graph with only the event-event relation links. The graph is too sparse which makes the information difficult to propagate through the nodes. Thus, we added semantic similarity links based on the cosine similarity of the event context embeddings. During the training of the RGCN, the node embeddings are kept frozen. After the training of the link prediction task, we extract the relation embeddings of the RGCN.

Regularization Term To mitigate the posterior collapse problem (Lucas et al., 2019) and have a stable inference process, we adopt the Maximum Mean Discrepancy (MMD)[4] which is an estimation of Wasserstein distance (Tolstikhin et al., 2018) as the regularization term (Eq. 8).

4 Experimental Setup

Datasets We evaluated the proposed Bayesian-Trans model on three event temporal relation datasets: MATRES (Ning et al., 2018c), Temporal and Causal Reasoning (TCR) (Ning et al., 2018a), and TimeBank-Dense (TBD) (Cassidy et al., 2014). TimeBank-Dense is a densely annotated dataset focusing on the most salient events and providing 6 event temporal relations. MATRES follows a new annotation scheme which focuses on main time axes, with the temporal relations between events determined by their endpoints, resulting in a consistent inter-annotator agreement (IAA) on the event annotations (Ning et al., 2018c). TCR follows the same annotation scheme, yet with a much smaller number of event relation pairs than in MATRES. Table 1 shows the statistics of the datasets.

Class	MATRES	TCR	TBD
BEFORE	6,852	1,780	2,590
AFTER	4,752	862	2,104
EQUAL/SIMULTANEOUS	448	4	215
VAGUE/NONE	1,425	N/A	5,910
INCLUDE	N/A	N/A	836
ISINCLUDED	N/A	N/A	1,060
Total	12,740	2,646	12,715

Table 1: The statistics of MATRES, TCR, and TBD.

Baselines We compare the proposed Bayesian-Trans[5] with the following baselines:

CogCompTime (Ning et al., 2018d) is a multi-step system which detect temporal relation using semantic features and structured inference.

BiLSTM is a basic relation prediction model built by Han et al. (2019b).

LSTM + knowledge (Ning et al., 2019) incorporates knowledge features learnt from an external source and optimize global consistency by ILP.

Deep Structured (Han et al., 2019a) adds a structured support vector machine on top of a BiLSTM.

Joint Constrained Learning (Wang et al., 2020b) constrains the training of a RoBERTa-based event pair classifier using predefined logic rules, while knowledge incorporation and global optimization are also included.

Poincaré Event Embedding (Tan et al., 2021) learns event embeddings based on a Poincaré ball and determines the temporal relation base on the relative position of events.

HGRU + knowledge (Tan et al., 2021) is a neural architecture processing temporal relations via hyperbolic recurrent units which also incorporates knowledge features like LSTM + knowledge.

Relative Event Time (Wen and Ji, 2021) is a neural network classifier combining an auxiliary task for relative time extraction over an event timeline.

UAST (Cao et al., 2021) is an uncertainty-aware self-training model. We show the result of the model which is trained on all the labeled data.

5 Experimental Results

Temporal Relation Classification We first compare Bayesian-Trans with the most recent approaches for temporal event classification in Table 2, including methods with or without commonsense knowledge injection. The results are obtained by training models on the MATRES training set and evaluated on both the MATRES test set and TCR. Table 3 shows results from the TBD dataset which are generated using the provided train, development, and test sets. We report F_1 score on MATRES and TCR following the definition in (Ning et al., 2019), and micro-F_1 on TimeBank-Dense. Compared with existing methods, the proposed

---

[4] MMD calculation can be found in Appendix A.

[5] Hyperparameter setting can be found in Appendix B.

Model	MATRES P	MATRES R	MATRES F$_1$	TCR P	TCR R	TCR F$_1$
CogCompTime (Ning et al., 2018d)	61.6	72.5	66.6	-	-	70.7
Poincaré Event Embeddings (Tan et al., 2021)	74.1	84.3	78.9	85.0	86.0	85.5
Relative Event Time (Wen and Ji, 2021)	78.4	85.2	81.7	84.3	**86.8**	85.5
LSTM + knowledge (Ning et al., 2019)	71.3	82.1	76.3	-	-	78.6
Joint Constrainted Learning (Wang et al., 2020b)	73.4	85.0	78.8	83.9	83.4	83.7
HGRU + knowledge (Tan et al., 2021)	79.2	81.7	80.5	88.3	79.0	83.5
Bayesian-Trans	**79.6**	**86.0**	**82.7**	**89.8**	82.6	**86.1**

Table 2: Experimental results on MATRES and TCR. The first three lines contain methods without commonsense knowledge incorporation. The rest are methods which inject commonsense knowledge. The results of Wang et al. (2020b) and (Wen and Ji, 2021) on TCR are generated from our run of the source code provided by the authors since they are not available in their original papers. The others are taken from the cited papers.

Model	Micro-F$_1$
BiLSTM (Han et al., 2019b)	61.9
Deep Structured (Han et al., 2019a)	63.2
Relative Event Time (Wen and Ji, 2021)	63.2
UAST (Cao et al., 2021)	64.3
Bayesian-Trans	**65.0**

Table 3: Experimental results on TBD. All compared methods do not incorporate commonsense knowledge explicitly. The result of Wen and Ji (2021) is generated from our run of the source code provided by the authors since they are not available in their original paper. The others are taken from the cited papers.

Model	MATRES	TBD
(1) RoBERTa + MLP	81.5	62.8
(2) RoBERTa + Vanilla MuRE	80.4	60.5
(3) RoBERTa + Bayesian-Trans	82.2	63.0
(4) COMET-BART + MLP	81.8	63.2
(5) COMET-BART + Vanilla MuRE	81.8	62.6
(6) COMET-BART + Bayesian-Trans	**82.7**	**65.0**

Table 4: Ablation test results on MATRES and TBD.

Bayesian-Trans has generally better performance on all three datasets, with more noticeably improvements on MATRES. Bayesian-Trans has significant performance gains over previous methods with knowledge incorporation, which shows that it can utilize knowledge more extensively. Details of the per-class performance can be found in Table A2 and A3.

Ablation Study We conducted an ablation study to highlight the impact of the different modules composing Bayesian-Trans. The results are shown in Table 4. In particular, we have the following variants: (1) RoBERTa+MLP, using RoBERTa to encode the context and then feeding representations of head and tail events to a multi-layer perceptron (MLP) for temporal relation classification; (2) RoBERTa+ Vanilla MuRE, using MuRE to extract temporal relations without modeling its parameters as latent variables; (3) RoBERTa+Bayesian-Trans, our proposed model by replacing COMERT-BART with RoBERTa as the text encoder; (4) COMET-BART+MLP, using COMET-BART as context encoder and an MLP for temporal relation classification; and (5) COMET-BART+ Vanilla MuRE, the proposed model without Bayesian learning or knowledge incorporation. The results demonstrate that COMET-BART is a better choice as the context encoder. Using MuRE for event temporal knowledge embedding learning does not bring any improvement compared to using a simple MLP layer for event temporal relation prediction (see (1) cf. (2), and (4) cf. (5)). Regardless of the contextual encoder used, the results of (3) and (6) show the benefit of employing Bayesian learning which naturally incorporates prior knowledge of event temporal relations learned from an external knowledge source for event temporal relation detection. With our proposed Bayesian translational model, we observe an improvement of $0.9-1.8\%$ in micro-F$_1$ on MATRES and $0.2-2.5\%$ in micro-F$_1$ on TimeBank-Dense compare to their non-Bayesian counterparts.

Effects of the Priors We further investigate the impact of different priors on the model performance. Inspired by the work on VAEs by Burda et al. (2016) and Truong et al. (2021), we employed

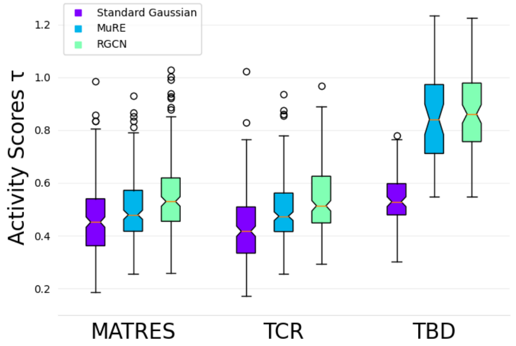


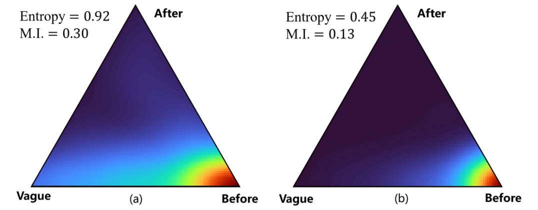
Figure 3: The box chart of the activity scores across all the dimensions of the latent encoding $\mathbf{\Lambda}$ with respect to the priors used in the model.

Dataset	Standard Gaussian	MuRE	RGCN
MATRES	81.2	81.8	82.7
TCR	84.3	85.4	86.1
TBD	63.6	64.6	65.0

Table 5: F_1 values based on different priors used in the proposed model.

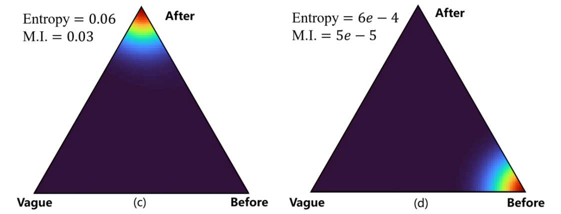
an *'activity' score*, $\tau = Cov_{e_h,e_t}(\mathbb{E}_{q(\mathbf{\Lambda}|e_h,e_t)}[\mathbf{\Lambda}])$ to evaluate the quality and diversity of the latent encodings. The intuition behind the "activity" score is that if a latent dimension encodes relevant information and is not redundant, its value is expected to vary significantly over different inputs. By computing the score across all the test instances, every dimension of $\mathbf{\Lambda}$ is given an 'activity' value. Latent units with a higher value are considered more active and thus more informative. Figure 3 shows activity scores with respect to different prior distributions, including the standard Gaussian prior and priors learned on ATOMIC using MuRE or RGCN, in which the latent variables are the least active when using standard Gaussian as the prior distribution. The higher activation is obtained using the priors learnt on the external knowledge base. In particular, the prior based on RGCN and MuRE over ATOMIC displays the most active units, with RGCN showing the most active units on average. Table 5 shows the performance of the proposed model based on different priors. Two-sided Welch's t-test ($p < 0.05$) also supports that the RGCN-learned prior improves over standard Gaussian prior.

Uncertainty Quantification We present an analysis of uncertainty quantification of the Bayesian-Trans predictions. We adopted the uncertainty quantification methods as in Malinin and Gales



All but four of the children who died were old enough to be vaccinated , but 90 percent of them did not get vaccinated, CDC officials said

All of the children who died were old enough to be vaccinated, but 90 percent of them did not get vaccinated, CDC officials said



Before departing for Jordan on Friday afternoon, Mr. Obama revealed that Mr. Netanyahu and Mr. Erdogan had just spoken by telephone

After departing for Jordan on Friday afternoon, Mr. Obama revealed that Mr. Netanyahu and Mr. Erdogan had just spoken by telephone

Figure 4: Examples of temporal relations in text and uncertainty quantification (entropy and mutual information) for the Bayesian-Trans model. Examples (*a*),(*b*) show how simplifying the linguistic structure without altering the temporal relation increases the model confidence. While examples (*c*),(*d*) illustrate the model's detection of temporal linguistic hints and its confidence.

(2018), computing the entropy (*total uncertainty*) and mutual information (*model uncertainty*) to visualize the predictive probabilities on a 2-simplex. Each forward pass on the same test instance is represented as a point on the simplex. For the sake of clarity of the visualization, we removed the EQUAL class, which is hardly ever predicted by the models.

In one of the test cases (Figure 4(a)), the true label is "*die*" BEFORE "*vaccinate*". This example exhibits a rather complex linguistic structure, as such, the model exhibits some uncertainty. Most of the predictions located at the corner are associated with BEFORE, but there also are several predictions scattered around it. We then simplified the sentence structure by removing "*but four*", and fed the modified sentence to the same model. This time, the model predicted the right temporal relation with much lower uncertainty (Figure 4(b)).

In another case study (Figure 4(c)), the true label is "*depart*" AFTER "*reveal*". This test case is rather straightforward, because of the explicit temporal word "before". The model predicted AFTER with high confidence, as shown by the predictive probabilities cluster at the top of the simplex. To show the impact of the temporal description, we swapped it from "*before*" to "*after*" and fed it to the same model. The model recognized the reversed meaning and correctly predicted BEFORE with low uncertainty (Figure 4(d)). The above cases demonstrate that the proposed model reacts to different inputs with reasonable uncertainty, on both the total and model uncertainty scores.

6 Conclusion

We propose a principled approach to incorporate knowledge for event temporal relation extraction named Bayesian-Trans, which models the relation representations in the MuRE translational model, as latent variables. The latent variables are inferred through variational inference, during which commonsense knowledge is incorporated in the form of the prior distribution. The experiments on MATRES, TCR, and TBD show that Bayesian-Trans achieves state-of-the-art performance. Comprehensive analyses of the experimental results also demonstrate the characteristics and benefits of the proposed model.

Limitations

Our approach takes an event pair as input for the prediction of their temporal relation. We observe that if two events reside in different sentences, the error rate increases by 19%. A promising future direction is to construct a global event graph where temporal relations of any two events are refined with the consideration of global consistency constraints, for example, no temporal relation loop is allowed in a set of events. Our current work only deals with even temporal relations, it could be extended to consider other event semantic relations such as causal, hierarchical or entailment relations. The event temporal knowledge in this paper is derived from ATOMIC which can possibly be extended to more sources. Bayesian learning could also be extended to life-long learning. But we need to explore approaches to address the problem of catastrophic forgetting. We didn't exhaustively investigate all the translational models due to the large volume of work in that area. There might be a translational model which can achieve better performance, but the core idea of the proposed framework stays the same.

Ethical Considerations

The goal of the proposed method is to understand the temporal relation between events based on the descriptions in the given text. What the method can achieve in the most optimistic scenario is no more than giving the same text to a human reader and letting him or her explain the event relations. Therefore, the ethical concerns only come from the data collection. In this paper, we only use publicly available datasets which have already been widely used in the research field. As for potential application, as long as the user collects the training data legally, the proposed method does not have the potential to have a direct harmful impact.

Acknowledgements

This work was supported in part by the UK Engineering and Physical Sciences Research Council (grant no. EP/T017112/1, EP/V048597/1, EP/X019063/1). YH is supported by a Turing AI Fellowship funded by the UK Research and Innovation (grant no. EP/V020579/1). This work was conducted on the UKRI/EPSRC HPC platform, Avon, hosted in the University of Warwick's Scientific Computing Group. XT was partially supported by the Research Development Fund (RDF) 2022/23 (University of Warwick): '*An Event-Centric Dialogue System for Second Language Learners*'.

References

Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Multi-relational poincaré graph embeddings. In *NeurIPS*.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. 2016. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112:859 – 877.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Yuri Burda, Roger B Grosse, and Ruslan Salakhutdinov. 2016. Importance weighted autoencoders. In *International Conference on Learning Representations*.

Pengfei Cao, Xinyu Zuo, Yubo Chen, Kang Liu, Jun Zhao, and Wei Bi. 2021. Uncertainty-aware self-training for semi-supervised event temporal relation extraction. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*.

Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 501–506, Baltimore, Maryland. Association for Computational Linguistics.

Ines Chami, Sami Abu-El-Haija, Bryan Perozzi, Christopher Ré, and Kevin Murphy. 2020a. Machine learning on graphs: A model and comprehensive taxonomy. *arXiv preprint arXiv:2005.03675*.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020b. Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6901–6914, Online. Association for Computational Linguistics.

Yao Cheng, Peter Anick, Pengyu Hong, and Nianwen Xue. 2013. Temporal relation discovery between events and temporal expressions identified in clinical narrative. *Journal of biomedical informatics*, 46.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2021. Distantly supervised relation extraction with sentence reconstruction and knowledge base priors. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 11–26, Online. Association for Computational Linguistics.

Joan Bruna Estrach, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2014. Spectral networks and deep locally connected networks on graphs. In *2nd international conference on learning representations, ICLR*, volume 2014.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Lin Gui, Jia Leng, Gabriele Pergola, Yu Zhou, Ruifeng Xu, and Yulan He. 2019. Neural topic model with reinforcement learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3478–3483, Hong Kong, China. Association for Computational Linguistics.

Rujun Han, I-Hung Hsu, Mu Yang, Aram Galstyan, Ralph Weischedel, and Nanyun Peng. 2019a. Deep structured neural network for event temporal relation extraction. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 666–106, Hong Kong, China. Association for Computational Linguistics.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019b. Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 434–444, Hong Kong, China. Association for Computational Linguistics.

EunJeong Hwang, Jay-Yoon Lee, Tianyi Yang, Dhruvesh Patel, Dongxu Zhang, and Andrew McCallum. 2022. Event-event relation extraction using probabilistic box embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 235–244, Dublin, Ireland. Association for Computational Linguistics.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. In *AAAI*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 687–696, Beijing, China. Association for Computational Linguistics.

Diederik P. Kingma and Max Welling. 2014a. Auto-encoding variational bayes. *CoRR*, abs/1312.6114.

Diederik P. Kingma and Max Welling. 2014b. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations ICLR*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Junru Lu, Xingwei Tan, Gabriele Pergola, Lin Gui, and Yulan He. 2022. Event-centric question answering via contrastive learning and invertible event transformation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2377–2389, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

James Lucas, George Tucker, Roger B Grosse, and Mohammad Norouzi. 2019. Don't blame the elbo! a linear vae perspective on posterior collapse. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Andrey Malinin and Mark Gales. 2018. Predictive uncertainty estimation via prior networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7047–7058.

Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. 2019. Topic modeling with Wasserstein autoencoders. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6345–6381, Florence, Italy. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018a. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2278–2288, Melbourne, Australia. Association for Computational Linguistics.

Qiang Ning, Sanjay Subramanian, and Dan Roth. 2019. An improved neural baseline for temporal relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6203–6209, Hong Kong, China. Association for Computational Linguistics.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020. TORQUE: A reading comprehension dataset of temporal ordering questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Qiang Ning, Hao Wu, Haoruo Peng, and Dan Roth. 2018b. Improving temporal relation extraction with a globally acquired statistical resource. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 841–851, New Orleans, Louisiana. Association for Computational Linguistics.

Qiang Ning, Hao Wu, and Dan Roth. 2018c. A multi-axis annotation scheme for event temporal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328, Melbourne, Australia. Association for Computational Linguistics.

Qiang Ning, Ben Zhou, Zhili Feng, Haoruo Peng, and Dan Roth. 2018d. CogCompTime: A tool for understanding time in natural language. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 72–77, Brussels, Belgium. Association for Computational Linguistics.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph LSTMs. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Gabriele Pergola, Lin Gui, and Yulan He. 2021a. A disentangled adversarial neural topic model for separating opinions from plots in user reviews. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2870–2883, Online. Association for Computational Linguistics.

Gabriele Pergola, Elena Kochkina, Lin Gui, Maria Liakata, and Yulan He. 2021b. Boosting low-resource biomedical QA via entity-aware masking strategies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1977–1985, Online. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *15th Extended Semantic Web Conference (ESWC)*, pages 593–607. Springer, Cham.

Zhaoyue Sun, Jiazheng Li, Gabriele Pergola, Byron Wallace, Bino John, Nigel Greene, Joseph Kim, and Yulan He. 2022. PHEE: A dataset for pharmacovigilance event extraction from text. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5571–5587, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xingwei Tan, Gabriele Pergola, and Yulan He. 2021. Extracting event temporal relations via hyperbolic geometry. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8065–8077, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. 2018. Wasserstein autoencoders. In *International Conference on Learning Representations*.

Quoc-Tuan Truong, Aghiles Salah, and Hady W Lauw. 2021. Bilateral variational autoencoder for collaborative filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 292–300.

Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2018. Event phase oriented news summarization. *World Wide Web*, 21(4):1069–1092.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020a. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020b. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706, Online. Association for Computational Linguistics.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Haoyang Wen and Heng Ji. 2021. Utilizing relative event time to enhance event-event temporal relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10431–10437, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Lixing Zhu, Zheng Fang, Gabriele Pergola, Robert Procter, and Yulan He. 2022. Disentangled learning of stance and aspect topics for vaccine attitude detection in social media. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1566–1580, Seattle, United States. Association for Computational Linguistics.

A Maximum Mean Discrepancy (MMD)

The Maximum Mean Discrepancy (MMD) can be unbiasedly estimated using the following equation (Nan et al., 2019):

$$\widehat{\mathrm{MMD}} = \frac{1}{N(N-1)} \sum_{n \neq m} f_k\left(z_n\,, z_m\,\right) + \frac{1}{N(N-1)} \sum_{n \neq m} f_k\left(\tilde{z}_n, \tilde{z}_m\right) - \frac{2}{N^2} \sum_{n,m} f_k\left(z_n, \tilde{z}_m\right)$$

where $z_1, ..., z_n$ are sampled from variational distribution q_ϕ and $\tilde{z}_1, ..., \tilde{z}_m$ are sampled from prior distribution p, $f_k(\cdot)$ is inverse multiquadratic knernel $f_k(x, y) = \frac{C}{C+||x-y||_2^2}$ which is often chosen for high-dimensional Gaussians.

Model	**Precision**	**Recall**	**F$_1$**
TransE	**80.0**	84.1	82.0
MuRP	75.4	85.8	80.2
AttH	76.2	**87.4**	81.4
MuRE	79.6	86.0	**82.7**

Table A1: The performance of the proposed Bayesian framework combined with different translational models on MATRES.

B Hyperparameter Settings and Resource Consumption

We conducted a grid-search to determine the optimal hyperparameters and dimensionality of the relation embeddings. The searching range for the dimension of the latent vector the transformation parameters is $[50, 200]$, with a step size of 50. As a result, on the MATRES, the dimension of the latent vector z is 200, the dimension of relation transformation vectors t_r or matrices W_r is 50, the dropout rate is 0.1. On the TBD, the dimension of the latent vector z is 100, and the dimension of relation transformation vectors t_r or matrices W_r is 100, dropout rate is 0. Based on the above settings, the number of parameters of the Bayesian-Trans is 443 thousand (excluding the COMET-BART). The COMET-BART encoder has 204 million parameters. The learning rate α_c for the context encoder is set to $1e-5$, while for other components of the architecture $\alpha = 1e-3$. To calibrate the influence of the regularization term, we set a scaling weight smoothly increasing from $1e-2$ to 2.0 during training. We ran the training for 60 epochs which is enough for the model to converge, and evaluated on the validation set after each training epoch.

All the experiments were conducted on an Nvidia GeForce RTX 3090 GPU. On the TBD dataset, the average training time is 93 seconds per epoch, while the inference time is 4 seconds. On the MATRES dataset, the average training time is 74 seconds per epoch, and the inference time is 7 seconds.

C Comparison of Translational Models

Table A1 shows the performance on MATRES using different translational models in the Bayesian framework. TransE (Bordes et al., 2013) is one of the most commonly used translational models, which only performs the addition transformation on the head event. AttH (Chami et al., 2020b) expands the idea of hyperbolic translational models by making the curvature learnable. It also introduces more types of transformation, reflection and rotation. MuRE (Balazevic et al., 2019) strikes a balance by conducting diagonal matrix transformation and addition transformation. MuRP (Balazevic et al., 2019) is the Poincarè version of MuRE, which projects the head and tail onto a Poincarè ball before performing scaling and addition. The score function of MuRP computes the Poincarè distance instead of the Euclidean distance.

We can observe that TransE performs well, beating the previous state-of-the-art ($F_1 = 81.7$) but gives slightly worse results compared to MuRE. Both translational models in the hyperbolic space, MuRP and AttH, are inferior to the Euclidean-based translational models. As MuRE gives more balanced precision and recall values, it is therefore adopted in our Bayesian learning framework.

Relation	**Precision**	**Recall**	**F$_1$**
BEFORE	82.66	90.40	86.35
AFTER	75.24	88.56	81.36
EQUAL	0.00	0.00	0.00
VAGUE	33.33	15.60	21.25

Table A2: Performance on each relation type class on MATRES.

D Class-Specific Results

In Table A2 and A3, We show the results obtained using Bayesian-Trans under each temporal relation

Relation	**Precision**	**Recall**	**F_1**
BEFORE	77.04	62.98	69.31
AFTER	76.37	62.20	68.56
SIMULTANEOUS	33.33	10.42	15.87
INCLUDES	36.84	10.29	16.09
ISINCLUDED	53.33	10.39	17.39
NONE	59.06	83.80	69.29

Table A3: Performance on each relation type class on TimeBank-Dense.

class on MATRES and TimeBank-Dense, respectively. On MATRES, the performance on BEFORE and AFTER are significantly better than for the other two classes. The model predicts no EQUAL labels, most likely caused by the scarce training data for this class. Previous works in the literature (Han et al., 2019b) have also shown similar class-specific results, with models struggling the most on the prediction of EQUAL and VAGUE relations. Similar conclusions can be drawn from the TimeBank-Dense dataset, that Bayesian-Trans performs relatively well on the BEFORE, AFTER and NONE classes, but performs worse on the other three minority classes.

Persona Expansion with Commonsense Knowledge for Diverse and Consistent Response Generation

Donghyun Kim[1] **Youbin Ahn**[1] **Wongyu Kim**[1] **Chanhee Lee**[1] **Kyungchan Lee**[1]
Kyong-Ho Lee[1] **Jeonguk Kim**[2] **Donghoon Shin**[2] **Yeonsoo Lee**[2]
[1]Department of Computer Science, Yonsei University
[2]NCSOFT
{dhkim92,ybahn,rladnjsrb9999,heederer,steven800,khlee89}@yonsei.ac.kr
{jukim,dhshin,yeonsoo}@ncsoft.com

Abstract

Generating diverse and consistent responses is the ultimate goal of a persona-based dialogue. Although many studies have been conducted, the generated responses tend to be generic and bland due to the personas' limited descriptiveness. Therefore, it is necessary to expand the given personas for more attractive responses. However, indiscriminate expansion of personas threaten the consistency of responses and therefore reduce the interlocutor's interest in conversation. To alleviate this issue, we propose a consistent persona expansion framework that improves not only the diversity but also the consistency of persona-based responses. To do so, we define consistency criteria to avoid possible contradictions among personas as follows: 1) Intra-Consistency and 2) Inter-Consistency. Then, we construct a silver profile dataset to deliver the ability to conform with the consistency criteria to the expansion model. Finally, we propose a persona expansion model with an encoder-decoder structure, which considers the relatedness and consistency among personas. Our experiments on the Persona-Chat dataset demonstrate the superiority of the proposed framework.

1 Introduction

In the field of open-domain dialogues, maintaining long-term interaction with users by generating human-like responses has been a persistent goal. As efforts along this line of research, persona-based dialogue models aim at generating more engaging and consistent responses based on personal traits (Li et al., 2016b; Qian et al., 2018). As the most widely used benchmark dataset, Persona-Chat (Zhang et al., 2018) consists of personal profiles, which contain 3-5 persona sentences (hereinafter referred to as "personas"), and dialogues conditioned on personas. With the Persona-Chat, many researches have attempted to improve the quality of responses from various perspectives (Song et al., 2020; Liu et al., 2020; Song et al., 2021; Cao et al., 2022). However, since the descriptiveness of personas is limited, they have difficulties in converting what personas imply into attractive responses.

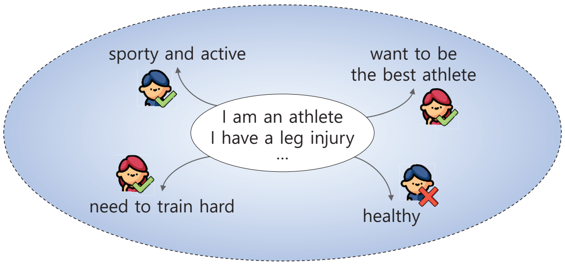


Figure 1: An example of persona expansion with commonsense. The center circle is the predefined profile, and the dashed circle is the valid expandable range.

Unlike dialogue models, humans converse beyond the explicit persona itself through rational reasoning with commonsense knowledge. The commonsense reasoning ability is essential for making an interlocutor engage in conversations with a dialogue agent. Figure 1 illustrates an example of persona expansion through commonsense reasoning. Given the persona '*I am an athlete*', it is possible to infer the characteristic '*sporty and active*' with high probability. The implications of '*want to be the best athlete*' and '*need to train hard*' are also predictable. In order to mimic human reasoning ability, previous works expand original personas (Majumder et al., 2020) or infer new personas from dialogue utterances (Kim et al., 2022). However, the consistency among expanded personas was not explicitly considered. To prolong a conversation, both the engagingness and the consistency of responses are very important. Therefore, we define the two consistencies that must comply with while improving the diversity of responses as follows: 1) **Intra-Consistency** and 2) **Inter-Consistency**. First, the Intra-Consistency means that expanded personas must not contradict the original persona.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1139–1149
May 2-6, 2023 ©2023 Association for Computational Linguistics

If the response conditioned on the expanded persona contradicts the original persona, the interlocutor loses interest in the conversation. Second, the Inter-Consistency means that expanded personas must not contradict other personas in the same profile. For instance, as shown in Figure 1, the attribute of '*healthy*' can be expanded from the original persona of '*I am an athlete*'. However, considering the other persona in the profile, '*I have a leg injury*', this comes to an inappropriate expansion. Therefore, it is necessary to follow the Intra- and Inter-Consistency when expanding personas to improve the persona-based responses.

To achieve the above goals, we propose a novel **P**ersona **E**xpansion framework with **C**ommonsense **K**nowledge (**PECK**). PECK aims to not only improve the diversity of responses but also avoid the problem of contradictions among personas. Both abilities to understand the relatedness and to reason with commonsense knowledge are required to satisfy the consistency criteria. Unfortunately, there are no supervised training datasets for persona expansion. Furthermore, it costs a lot to annotate them to train both abilities. Therefore, we build a silver profile dataset (see Section 4) similar to the gold profile in the Persona-Chat, with simple manipulation of the commonsense knowledge graph ANION (Jiang et al., 2021). In addition, we propose a persona expansion model (see Section 5) consisting of an encoder and a decoder for consistent expansion. The bidirectional encoder learns to understand the relatedness among personas within the same profile. Then, the auto-regressive decoder learns to expand personas consistently with negative log-likelihood (NLL) and unlikelihood (UL) (Welleck et al., 2019a) training objectives. Finally, we conduct extensive experiments on persona expansion and persona-based response generation. Experimental results confirm the superiority of the proposed consistent expansion framework.

The contributions of this work are as follows:

- A novel persona expansion framework that satisfies essential consistency criteria is proposed for attractive chit-chat responses.
- We generate consistent personas with a manipulated silver profile dataset and unlikelihood training objective.
- We show the effectiveness of the proposed expansion framework via experimental results.

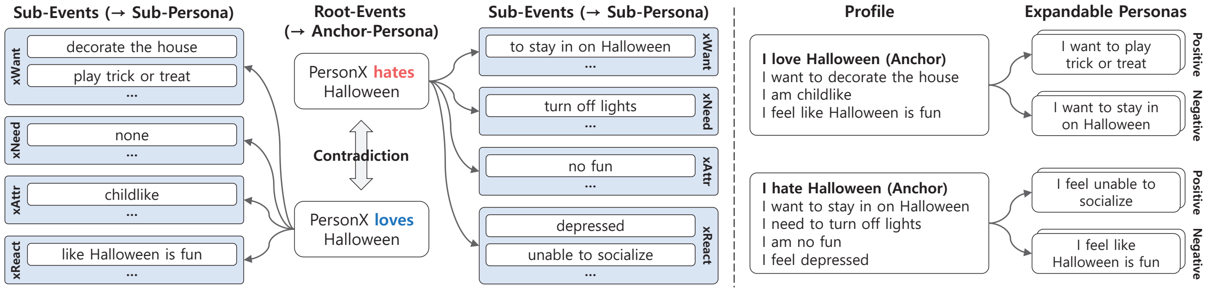
2 Related Work

2.1 Persona Expansion with Commonsense

COMPAC proposed by Majumder et al. (2020) makes a fine-grained choice of a persona to generate more engaging chit-chat responses. They expand the predefined personas leveraging COMET (Bosselut et al., 2019) trained with the commonsense knowledge of ATOMIC (Sap et al., 2019). For instance, COMET deduces expanded personas such as '*I am caring*, *I want to adopt a cat*' from the original persona '*I love animals*' along the nine relations in ATOMIC. Meanwhile, Kim et al. (2022) propose the dual task framework of debiasing persona-based dialogues via a data-driven approach. Similary, they utilize COMET to perform commonsense-based expansion before matching personas with utterances in a dialogue. The commonsense-based expansion in the conventional methods improves the engagingness of persona-based dialogues. However, they did not consider that consistency should be kept in the expansion process. To address this issue, we propose a novel approach of expanding personas and keeping the proposed Intra- and Inter-Consistency among them.

2.2 Commonsense Knowledge Graphs

As the demand for the representation of relations between real-world events, some recent studies on event commonsense knowledge graphs (CSKGs) have been conducted. ATOMIC (Sap et al., 2019) is an event-based CSKG, containing textual descriptions of inferential knowledge on real-world events. ATOMIC provides nine if-then causal relations, in which the reasoning such as causes and effects focuses on the agent (i.e., xAttr, xEffect, xIntent, xNeed, xReact, xWant.) or the other (i.e., oEffect, oReact, oWant). As a follow-up study, Hwang et al. (2021) propose ATOMIC-2020, which contains social and physical relations as well as eventive ones. Further, Jiang et al. (2021) introduce ANION, which focuses on negated and contradictory events contrary to the affirmative ones of ATOMIC. ANION contains 624k if-then rules consisting of negated and contradictory relations. We construct a silver profile dataset by leveraging the inter-event inferential commonsense of ANION via simple manipulation. And then, we compose positive and negative samples of unlikelihood training (Welleck et al., 2019a) for consistent persona expansion by leveraging the oppositions between each profile in the dataset.



(1) Examples of contradictory root-events and their sub-events

(2) Positive and Negative Training Samples

Figure 2: (1) An example of contradictory root-events and their sub-events in ANION. The root-event and its corresponding sub-events are designated anchor-persona and sub-personas, respectively. (2) Training samples for consistent persona expansion. Reasonable expansions are classified as positive samples E^+, and contradictories are classified as negative samples E^-.

3 Problem Statement

In this paper, our goal is to generate expanded personas, which contribute to improving the quality of persona-based responses. To do so, the consistency criteria that expanded personas must comply with are as follows: 1) **Intra-Consistency** and 2) **Inter-Consistency**. Formally, given an original persona $p_i \in P$, where $P = \{p_1, p_2, ..., p_n\}$ is the predefined profile, a persona expansion model aims to generate expanded personas $E_i = \{e_1, e_2, ..., e_m\}$, which do not contradict the original persona p_i and all the remaining personas in profile P. For the model to learn these consistencies, we need original-expandable persona pairs with a positive or negative label, and the profile where the original persona belongs. That is to say, the training dataset should contain {*Original Personas* (*Profile*), *Expandable Personas*, *Label*}. Regrettably, there are no training datasets of this form, and human annotation is expensive. Hence, we construct a new dataset, silver profile, appropriate for our expansion goal through simple manipulation on ANION, the existing commonsense knowledge graph. The detailed construction process will be described in Section 4. Finally, our model trained with the silver profile aims to generate expanded personas consistently from gold profiles within Persona-Chat.

4 Data Construction

Leveraging ANION, we construct a new dataset for training consistent persona expansion. Through exploratory analysis on Persona-Chat, we identified some prominent features of profiles. First, each profile is mostly made up of personas, which are correlated or at least non-contradictory to one another. In the gold profile example in Table 1, three of the five personas (1, 2, 3) are directly correlated, and the others are slightly related. Second, most of all personas indicate the speaker's preferences (i.e., I like –, I love –, I want –) or states (i.e., I am –). Based on these structural and semantical features of profiles, we build a silver profile dataset similar to the gold profile in Persona-Chat to train our expansion model. Statistics of the silver profile dataset are given in Appendix B.

Gold Profile	Silver Profile
1. I play football.	1. I love Halloween.
2. My position is linebacker.	2. I want to decorate the house.
3. I am an athlete.	3. I am childlike.
4. I had 128 tackles last year.	4. I feel like Halloween is fun.
5. My team is the baltimore ravens.	

Table 1: Examples of the gold profile from Persona-Chat and silver profile from ANION.

4.1 Profilizing Commonsense Events

With reference to the aforementioned feature of the profile, the *root-event* existing in ANION and the *sub-events* connected by if-then relations can be defined as the speaker's profile. In this case, the root-event selected for profilizing is the sentence described only for *PersonX*, not including *PersonY*.[1] As shown on the left of Figure 2, the root-event is defined as a dominant *anchor-persona* in the profile, and each sub-event is defined as a *sub-persona* that can be inferred from the anchor-persona. To filter mislinked relations out during ANION deployment, we leverage the RoBERTa (Liu et al., 2019) model fine-tuned with the MNLI

---

[1] In ANION, there are events related to the agent (i.e., *PersonX* –) and events related to the agent and the other people (i.e., *PersonX* – *PersonY* –). In this study, events with *PersonY* are excluded because we only handle personas representing the traits of agent '*I*'.

(Williams et al., 2018) dataset to verify the non-contradiction between a root-event and its sub-events. Also, based on the fact that most personas represent the speaker's preferences or states, we get sub-events from four if-then relations as follows: *xWant*, *xNeed*, *xAttr*, *xReact*. Then, we change the subject of each event to '*I*' and conduct subject-verb matching to convert them into a sentence of a persona type, such as <*PersonX loves Halloween*> → '*I love Halloween*'. An example of silver profile is shown on the right side of Table 1. If '*I love Halloween*' is the anchor-persona, then '*I want to decorate the house*' and '*I feel like Halloween is fun.*' are designated as sub-personas.

4.2 Assigning Expandable Personas to Profile

A silver profile has a maximum of five personas; one is the anchor-persona, and the others are sub-personas from four if-then relations. Since there are one or more sub-personas for each relation, sentences not designated as sub-personas can be assigned as expandable personas. As shown on the right side of Figure 2, '*I want to play trick or treat*' is not designated as a sub-persona, so it is treated as an expandable persona. Through this process, the expandable personas that can be inferred for the four relations are assigned to the profile.

4.3 Aligning Contradictory Profiles

Inspired by Li et al. (2020), we leverage NLL and UL objectives which require both positive and negative samples to endow the model with the consistent expansion ability. Therefore, we align the corresponding contradictory profiles to construct positive and negative samples. In ANION, negated root-events such as <*PersonX hates Halloween*> and <*PersonX does not loves Halloween*> contradict their affirmative <*PersonX loves Halloween*>. Using these opposites as pivots, we construct negative samples by swapping the expandable personas of each other. As shown on the right side of Figure 2, for profiles with '*I love Halloween*' as an anchor-persona, '*I want to play trick or treat*' is a positive expandable persona, and '*I want to stay in on Halloween*' comes to a negative one.

Definition 1 *A silver profile dataset is defined as* $D_{sp} = \{(P_i, E_i^{+/-})\}_{i=1}^{K}$*, where* P_i *is a synthetic profile,* $E_i^{+} = \{E_i^{want}, E_i^{need}, E_i^{attr}, E_i^{react}\}$ *is a set of expandable personas for four relations,* E_i^{-} *is a set of contradictories, and each* $E_i^{rel} = \{e_1^{rel}, e_2^{rel}, ..., e_{m_{rel}}^{rel}\}$ *consists of* m_{rel} *personas.*

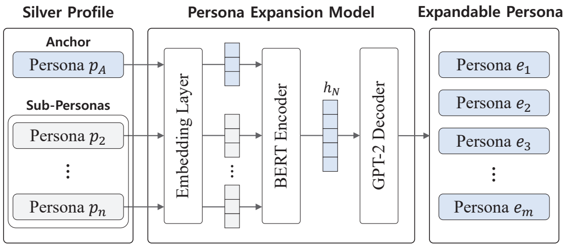


Figure 3: Overview of the proposed expansion model. In the training phase, the model leverages the silver profile dataset to learn consistencies among personas.

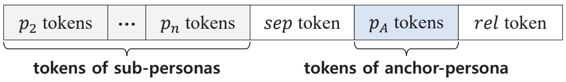


Figure 4: Input token template for training the persona expansion model.

5 Consistent Persona Expansion

5.1 Overview

In this section, we propose a novel consistent persona expansion model. To achieve the consistency criteria mentioned above, we design the model with an encoder-decoder structure as illustrated in Figure 3. The encoder learns to understand the relatedness among personas in the profile, and the decoder learns to consistently reason expandable personas with commonsense knowledge. This separated architecture allows the model to generate expanded personas satisfying the consistencies.

Formally, given a silver profile dataset D_{sp}, the model takes the profile $P = \{p_1, p_2, ..., p_n\}$ as input, where p_1 ($= p_A$) is the anchor-persona, and the others are sub-personas. Then, the target expandable personas are $E^{rel} = \{e_1^{rel}, e_2^{rel}, ..., e_{m_{rel}}^{rel}\}$.

5.2 Encoder

The encoder is initialized by the pre-trained BERT (Devlin et al., 2019) embedding to inherit its bidirectional language understanding ability.

The input consists of unfolded persona profile P to recognize a correlation among personas in the profile. As illustrated in Figure 4, a separate token $[sep]$ is placed between the sequence of sub-personas and the anchor-persona, and a relation token $[rel]$ is placed at the end of the input. The formatted input is as follows:

$$p_2, ..., p_n, [sep], p_A, [rel]. \tag{1}$$

Then, the embedding layer transforms each token of the input into a vector representation. At this time, the sum of the corresponding token and position embedding is used as the vector representation. The resulting vector representation is denoted as $input$. Next, the encoder takes the vector representation and performs multi-head attention (Vaswani et al., 2017) as follows:

$$h_{i+1} = FFN(MultiHead(h_i, h_i, h_i)), \quad (2)$$

where $MultiHead(query, key, value)$ calculates the scaled dot-product attention of query, key, value, $FFN(\cdot)$ is a fully connected feed-forward network, and $h_0 = input$. The encoder outputs the vector representation h_L after L identical layers. The vector representation h_L implies context information for expansion and the correlatedness among all personas in the profile.

5.3 Decoder

The decoder is initialized by the pre-trained GPT-2 (Radford et al., 2019) to leverage its robust autoregressive generation ability.

Unlike the encoder, the decoder adopts the cross-attention and left-to-right mask. First, the cross-attention is performed to reflect the context information among personas. In the cross-attention, the query is the vector representation from previous layer, and the output vector h_L of the encoder is given as the key and value:

$$v_{i+1} = FFN(MultiHead(v_i, h_L, h_L)), \quad (3)$$

where v_0 is initialized from the embeddings of the decoder input $e_j^{rel} (\in E^{rel})$ for given relation rel in Equation 1. Then, the decoder outputs the vector representation v_L after L identical layers. Finally, after the linear and softmax layer, we get the probabilities of each token in the output sequence.

5.4 Training Objectives

For the consistent persona expansion, we exploit negative log-likelihood loss (NLL) and unlikelihood loss (UL). NLL is a conventional objective function for generation tasks that maximizes the probability of the next target token. In contrast, UL is an objective function that minimizes the probability of the next target token, usually used to prevent the model from generating dull and repetitive sequences. In our setting, referred to Li et al. (2020), we use NLL and UL for positive samples (D_{sp}^{+}) and negative samples (D_{sp}^{-}), respectively. The expression for each sample is as follows:

$$D_{sp}^{+} = \{P, E^{+}\}, D_{sp}^{-} = \{P, E^{-}\}, \quad (4)$$

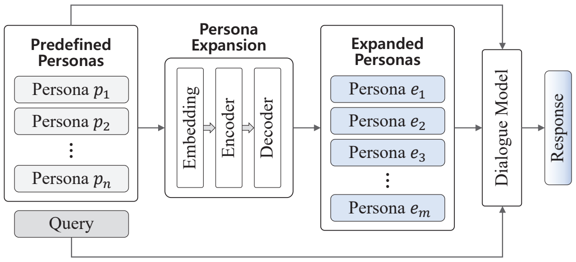


Figure 5: Response generation with persona expansion.

where P is the profile, E^{+} is the expandable personas, and E^{-} is the contradictory personas. For simplicity of expression, we omit the index notation i. See Definition 1 for the exact expression. The NLL and UL losses are calculated as follows:

$$L_{NLL}^{D_{sp}^{+}} = -\sum_{i=1}^{|e|} log(p_\theta(t_i | P, e_{<i})), \quad (5)$$

$$L_{UL}^{D_{sp}^{-}} = -\sum_{i=1}^{|e|} log(1 - p_\theta(t_i | P, e_{<i})), \quad (6)$$

where e is each persona in E^{+}/E^{-}, and t_i is the token of the persona e at time step i. The final objective function of the model computes the weighted sum of NLL and UL losses as follows:

$$Loss = \alpha L_{NLL}^{D_{sp}^{+}} + \beta L_{UL}^{D_{sp}^{-}}, \quad (7)$$

where α and β are hyperparameters. The optimized objective function is obtained with iterative loss back-propagation.

5.5 Persona Expansion

Through the training with silver profile, our model acquires the ability to understand the relatedness among input personas and to generate consistent expanded personas. The model is leveraged to expand the personas of the gold profile in the Persona-Chat. For each persona in the gold profile, we generate five expanded personas per relation (i.e., xWant, xNeed, xAttr, xReact) by beam-search decoding. Then, the number of expanded personas per an original persona comes to 20 (= 5 × 4).

5.6 Response Generation

After persona expansion, we generate diverse and consistent responses utilizing the original and expanded personas, as illustrated in Figure 5. In this paper, we leverage two dialogue models to prove the effectiveness of the proposed framework on response generation.

GPT-2 (Wolf et al., 2019): A GPT-2 model simply concatenates all personas with dialogue history. In our setting, up to five expanded personas with the highest NLI scores on the predefined profile are selected and fed into the dialogue model.

COMPAC (Majumder et al., 2020): A dialogue model that generates fine-grained persona-based responses through expansion and selection. In our setting, the expanded personas generated by PECK framework are selected by COMPAC and used for response generation.

6 Experiments

6.1 Dataset

We conducted experiments on both persona expansion and response generation with Persona-Chat, converted for the ConvAI2 benchmark (Dinan et al., 2020). The Persona-Chat consists of dialogues between two speakers, each utterance conditioned on personas in the speaker's profile. We generated the expanded personas using personas in the predefined profile. And we carried out the response generation using utterances and expanded personas. In our experiments, 16,878 instances were used for training, 1,000 for validation and 1,000 for testing.

6.2 Baselines

To verify the effectiveness of the constructed silver profile dataset and the proposed expansion model, the following three baseline models were adopted and compared.

1. **COMET$_{\textbf{ATOMIC}}$** (Bosselut et al., 2019): As a GPT-2 model fine-tuned on ATOMIC, it expands the original personas along the four relations mentioned above for comparison.
2. **COMET$_{\textbf{SP w/o UL}}$**: We fine-tuned the GPT-2 model with a silver profile dataset as a comparative model, which takes a concatenated sequence of personas as input. UL objective was not used in training.
3. **COMET$_{\textbf{SP w/ UL}}$**: We fine-tuned the GPT-2 using a silver profile dataset and UL with negative samples. The other settings except UL are the same as the above model.

6.3 Evaluation Metrics

6.3.1 Automatic Evaluation

Four metrics were utilized for automatic evaluation. We employed Perplexity (**PPL**), which measures how well a model predicts response tokens, where lower means better. We employed Distinct-1/2 (**Dist-1/2**) (Li et al., 2016a), which measures the diversity of sentences via calculating the ratio of distinct words against total uni/bi-grams. We employed F1 score of BERTScore (**F-BERT**) (Zhang et al., 2019), which measures how semantically relevant the expanded persona is to the original persona or generated response is to the selected persona. We employed Consistency Score (**C.Score**) (Madotto et al., 2019), which measures the consistency of the expanded persona (or response) for the predefined profile P. The scoring process is as follows:

$$NLI(p_i, e) = \begin{cases} 0, & \text{if } e \text{ contradicts } p_i, \\ 1, & \text{otherwise,} \end{cases} \tag{8}$$

$$C.Score(e) = \sum_{i=1}^{n} NLI(p_i, e), \tag{9}$$

where p_i is each persona in the profile P, and e is the expanded persona (or response) to score. RoBERTa was employed for the NLI model, which is fine-tuned with the DNLI (Welleck et al., 2019b) dataset. The Consistency Score is calculated, and scaled to a value between 0 and 1.

6.3.2 Human Evaluation

For qualitative assessment, we conducted evaluations by three annotators. We recruited annotators proficient in the language from an external agency. Annotators had sufficient knowledge of the domain but knew nothing of the proposed model. During the evaluation, there was no interaction among annotators that could affect each other's ratings. Also, annotators were properly compensated for the work they performed.

We randomly selected 100 samples each to evaluate persona expansion and response generation. Human annotators were asked to evaluate samples with the following four metrics: (1) Engagement, (2) Relevance, (3) Diversity, (4) Intra-Consistency, and (5) Inter-Consistency. Ratings range from 1 to 5, with higher scores indicating better. Intra-Consistency measures the consistency between the expanded persona and the original persona. Inter-Consistency is scaled to a value between 1 to 5, after summing up the one-to-one Intra-Consistency score between the expanded persona and each persona in the profile. Further, we conducted human A/B test that provide consensus among annotators for consistency.

Model	Automatic Evaluation				Human Evaluation			
	Dist-1	Dist-2	F-BERT	C.Score	Engagement	Relevance	Intra-Con.	Inter-Con.
$\text{COMET}_{\text{ATOMIC}}$	0.041	0.162	0.806	0.54	2.92	3.01	3.04	2.24
$\text{COMET}_{\text{SP w/o UL}}$	0.044	0.165	0.810	0.66	3.08	3.06	3.12	2.67
$\text{COMET}_{\text{SP w/ UL}}$	0.043	0.164	0.818	0.70	3.05	3.18	3.58	3.52
$\text{PECK}_{\text{w/o UL}}$	**0.052**	**0.183**	0.822	0.74	**3.12**	3.25	3.61	3.73
$\text{PECK}_{\text{w/ UL}}$	0.047	0.179	**0.835**	**0.83**	3.10	**3.70**	**4.11**	**4.05**

Table 2: Automatic and human evaluation results on the persona expansion. The best results are **bolded**, and the second-best are underlined.

6.4 Persona Expansion Results

The automatic and human evaluation results are reported in Table 2. Compared with baselines, PECK outperformed other expansion models in all metrics regarding diversity and consistency. In particular, relevance (i.e., F-BERT, Relevance) and consistency (i.e., C.Score, Intra/Inter-Consistency) were significantly improved compared to diversity (i.e., Dist-1/2, Engagement). This result demonstrated the effectiveness of the proposed framework for consistent persona expansion.

6.4.1 Anlalysis

Through a detailed analysis of the results, we identified the contributions of the following three implements: 1) silver profile dataset, 2) encoder-decoder structure, and 3) UL objective.

First, from the results of $\text{COMET}_{\text{ATOMIC}}$ and $\text{COMET}_{\text{SP w/o UL}}$ in Table 2, the latter model using the silver profile dataset for training showed better performance in all evaluation metrics. Especially, C.Score showed a higher improvement compared to other automatic metrics. In human evaluation, Inter-Consistency increased more significantly than other metrics. This result showed that the presence of the silver profile helped improve the consistency of the expanded personas. In other words, it means that the silver profile sufficiently imbued the expansion model with the characteristics of the profile and correlatedness among personas.

To analyze the impact of the encoder-decoder structure, COMET_{SP} and PECK were trained using the same silver profile dataset. Different from PECK, COMET_{SP} is a model based on GPT-2 with a decoder-only structure. Our experimental results showed that PECK outperformed COMET_{SP} despite being trained on the same dataset. In addition, we found significant increases in C.Score and Inter-Consistency, the metrics representing one-to-many consistency. This means that the encoder has effectively fused personas within the profile so that the decoder could generate expanded personas consistently for the given profile.

PECK vs. $\text{COMET}_{\text{ATOMIC}}$	Intra-Consistency			Inter-Consistency		
	Win	Lose	k	Win	Lose	k
xWant	72.4	11.6	0.62	76.1	9.4	0.71
xNeed	68.3	18.4	0.52	72.7	10.2	0.66
xAttr	74.5	10.8	0.64	79.5	7.8	0.73
xReact	61.8	23.7	0.48	69.4	14.6	0.54

Table 3: Human A/B results for consistency. Ties are not indicated in the table. The values of Fleiss' kappa k (Fleiss, 1971) for all results are in $0.4 < k < 0.8$, indicating moderate and substantial agreements.

In terms of UL objective, both $\text{COMET}_{\text{SP w/ UL}}$ and $\text{PECK}_{\text{w/ UL}}$ models showed higher relevance and consistency than those without UL training. We found that Dist-1/2 and Engagement metrics slightly decreased when using UL training. This result seems to be because the constraint of generation by UL training defected the range of expanded personas. However, despite the decrease in diversity, it can be judged to be a sufficiently attractive expansion. The analysis demonstrated how effective the UL training is in improving consistency while keeping diversity.

6.4.2 Human A/B Test

To compare the consistency of the expansion for each relation, we conducted a human A/B test. Human annotators were asked to choose a more consistent persona between personas expanded by $\text{COMET}_{\text{ATOMIC}}$ and PECK. Table 3 shows that Intra- and Inter-Consistency of the persona expanded by PECK outperformed in all relations. In particular, the performances for Inter-Consistency were better than Intra-Consistency. Through these results, we found that PECK is effective in solving Inter-Consistency contradiction that the existing model cannot control.

Dialogue Model	Expansion Model	Automatic Evaluation				Human Evaluation			
		PPL	Dist-1	Dist-2	C.Score	Engagement	Diversity	Relevance	Consistency
GPT-2	-	22.12	0.054	0.108	0.45	2.08	2.64	2.98	2.25
	COMET$_{\text{ATOMIC}}$	21.43	0.206	0.297	0.39	3.18	2.78	2.71	2.12
	PECK$_{\text{w/ UL}}$	21.18	0.223	0.345	0.67	3.22	3.31	3.05	3.24
COMPAC	-	19.84	0.146	0.253	0.58	2.31	3.02	3.14	2.86
	COMET$_{\text{ATOMIC}}$	16.36	0.836	0.964	0.64	3.35	3.42	3.07	2.97
	PECK$_{\text{w/ UL}}$	**16.21**	**0.851**	**0.986**	**0.76**	**3.43**	**3.67**	**3.22**	**3.53**

Table 4: Automatic and human evaluation results on the response generation. The best results are **bolded**, and the second-best are underlined.

6.5 Response Generation Results

The automatic and human evaluation results on the response generation are reported in Table 4. The dialogue models with expansion showed better performance than those without expansion. Especially, COMPAC with PECK outperformed the others in all evaluation metrics. In the following sections, we further analyze the diversity and consistency of generated responses.

6.5.1 Analysis on Diversity

The results of the response generation showed that persona expansion greatly contributes to improving the diversity of responses. In automatic evaluation, Dist-1/2 metrics significantly outperformed when using expansion models than when not using them. In particular, the values of Dist-1/2 showed greater increases when using the personas expanded by PECK. The impact of expanded personas was noticeable in human evaluation. In Engagement and Diversity, the responses based on expanded personas were rated well by human annotators. This means that PECK framework is helpful in generating attractive responses, which are essential to engage interlocutors and prolong a conversation.

6.5.2 Analysis on Consistency

In terms of consistency, the performance increased significantly when PECK framework was applied. Further, PECK outperformed COMET on both dialogue models. When COMET expansion was utilized for response generation, C.Score, Relevance, and Consistency metrics were, in some cases, lower than without expansion. GPT with PECK achieved better C.Score and Consistency than COMPAC with COMET, despite not performing fine-grained persona selection. This suggests that COMET persona expansion help improve diversity but adversely affects consistency. On the other hand, PECK not only improved the diversity of responses but was also the best in consistency. These results demonstrated that the proposed expansion framework is effective at generating diverse and consistent responses.

Intra-Consistency	
Original Persona	I am primarily a **meat eater**
COMET$_{\text{ATOMIC}}$	I want to **avoid meat** I am **vegetarian** I feel satisfied
PECK$_{\text{w/ UL}}$	I want to **eat meat** I am **carnivore** I feel happy
Inter-Consistency	
Original Profile	I love to rest at home on the weekend (A) I am a **hard worker** My hobby is walking with my dog
COMET$_{\text{ATOMIC}}$	I am **lazy** I want to take a rest
PECK$_{\text{w/ UL}}$	I am **tired** I want to relax

Table 5: Examples of expanded personas. Each sample was picked from all expansions. Red indicates expansion that contradicts the original.

6.6 Case Study on Intra-/Inter-Consistency

Table 5 shows sample cases of persona expansion on Intra-/Inter-Consistency. In Intra-Consistency case, given the original persona '*I am primarily a meat eater*', COMET generated contradictory personas like '*I want to avoid meat*' and '*I am vegetarian*'. On the other hand, PECK generated consistent personas. In Inter-Consistency case, '*I love to rest at home on the weekend*' is an anchor-persona, and '*I am a hard worker*' is a sub-persona to be considered when expanding. COMET generated expanded personas such as '*I am lazy*' and '*I want to take a rest*'. Here, the expanded persona '*I am lazy*' contradicts the contents of the sub-persona. On the contrary, the personas expanded by PECK are consistent with the original profile.

7 Conclusions and Future Work

In this paper, we proposed a novel persona expansion framework to improve the diversity and consistency of the persona-based dialogue. Leveraging the existing commonsense knowledge graphs, we built a silver profile dataset to deliver the ability to conform with the consistency criteria (Intra-/Inter-Consistency) to the expansion model. The expansion model of the encoder-decoder structure was trained using the silver profile dataset with unlikelihood training. After training, the model generated expanded personas that satisfy the consistency criteria. The experimental results proved that the proposed persona expansion is effective at improving the diversity of responses while keeping their consistency with personas. In the future, we plan to extend PECK to leverage more external knowledge, such as ConceptNet (Speer et al., 2017).

Limitations

In this study, the expansion target was limited to personas with a first-person subject. However, if other forms of persona expansion are considered, more diverse and richer responses can be generated. To solve this problem, we plan to research strategies to more effectively utilize various existing commonsense datasets in the persona-based dialogue model.

Ethical Considerations

Through our analysis of the persona expansion, we found that expanded personas highly depend on the commonsense knowledge used in the model training. We built a silver profile dataset utilizing the commonsense knowledge graph ANION and our expansion model trained with the silver profile dataset. As mentioned in ANION, linguistic, social, and cultural biases may exist within the ANION, the source of our silver profile dataset. These biases or stereotypes will affect the meaning of the personas expanded by our framework. This study did not discuss this aspect, but we believe it is an issue worth considering in future works.

Acknowledgements

We thank all anonymous reviewers for their valuable comments on this work. This work was supported by NCSOFT NLP Center. Kyong-Ho Lee is the corresponding author.

References

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Yu Cao, Wei Bi, Meng Fang, Shuming Shi, and Dacheng Tao. 2022. A model-agnostic data manipulation method for persona-based dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7984–8002, Dublin, Ireland. Association for Computational Linguistics.

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

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition: From Machine Learning to Intelligent Conversations*, pages 187–208. Springer.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: On symbolic and neural commonsense knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6384–6392.

Liwei Jiang, Antoine Bosselut, Chandra Bhagavatula, and Yejin Choi. 2021. "I'm not mad": Commonsense implications of negation and contradiction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4380–4397, Online. Association for Computational Linguistics.

Minju Kim, Beong-woo Kwak, Youngwook Kim, Hong-in Lee, Seung-won Hwang, and Jinyoung Yeo. 2022. Dual task framework for improving persona-grounded dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10912–10920.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016b. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany. Association for Computational Linguistics.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2020. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4715–4728, Online. Association for Computational Linguistics.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, and Pascale Fung. 2019. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5459, Florence, Italy. Association for Computational Linguistics.

Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. 2020. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9194–9206, Online. Association for Computational Linguistics.

Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Assigning personality/profile to a chatting machine for coherent conversation generation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4279–4285.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Haoyu Song, Yan Wang, Kaiyan Zhang, Wei-Nan Zhang, and Ting Liu. 2021. BoB: BERT over BERT for training persona-based dialogue models from limited personalized data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–177, Online. Association for Computational Linguistics.

Haoyu Song, Yan Wang, Wei-Nan Zhang, Xiaojiang Liu, and Ting Liu. 2020. Generate, delete and rewrite: A three-stage framework for improving persona consistency of dialogue generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5821–5831, Online. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019a. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019b. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Implementation Details

A.1 Persona Expansion

Our persona expansion model was implemented with HuggingFace Transformers.[2] The encoder and decoder initialize from the publicly available BERT-base-uncased[3] and GPT-2-small[4] model, respectively. Both models are with 12 layers and 768 hidden sizes. In addition, we use an Adam optimizer (Kingma and Ba, 2014), and the learning rate varies from 5e-6 to 5e-5. The α and β, hyperparameters for weighted NLL and UL training, are 0.8 and 0.07, respectively. The training was conducted on an Nvidia RTX3090 24G GPU with a batch size of 16. The best-performing model was found in epochs 11 to 13.

A.2 Response Generation

We utilized the repositories and implementations of GPT-2[5] and COMPAC[6] for response generation. To suit the purpose of our experiment, we adjusted some of the details of these models and trained them in a single-turn dialogue setting.

	Positive	**Negative**	**All**
# of unique anchor personas	4,253	12,540	16,793
# of silver profiles	2,428,926	3,710,299	6,139,225
Average # of profiles per unique anchor persona	571.11	295.88	365.58

Table 6: Statistics of silver profile dataset

Positive Profile				
Relation	xAttr	xNeed	xReact	xWant
Average # of expandable personas	4.48	3.65	2.60	4.55
Negative Profile				
Relation	xAttr	xNeed	xReact	xWant
Average # of expandable personas	3.17	1.24	2.74	3.50

Table 7: Expandable personas per relation

B Statistics of Silver Profile

The statistics of the silver profile dataset are shown in Table 6. We built a total of 6,139,225 synthetic profiles based on 16,793 unique anchor personas. Each profile has expandable personas for four relations. Table 7 shows the average number of expandable personas per relation.

---

[2] `https://github.com/huggingface/transformers`

[3] `https://huggingface.co/bert-base-uncased`

[4] `https://huggingface.co/gpt2`

[5] `https://github.com/huggingface/transfer-learning-conv-ai`

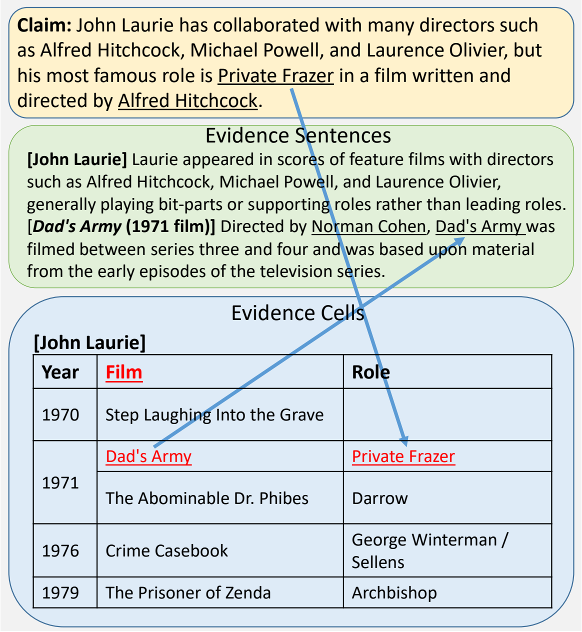
[6] `https://github.com/majumderb/compac`

UniFEE: Unified Evidence Extraction for Fact Verification

Nan Hu♠[*], **Zirui Wu**♠[*], **Yuxuan Lai**♡♠, **Chen Zhang**♠, **Yansong Feng**♠♢[†]
♠Wangxuan Institute of Computer Technology, Peking University, China
♢The MOE Key Laboratory of Computational Linguistics, Peking University, China
♡Department of Computer Science, The Open University of China
`{hunan,ziruiwu,erutan,zhangch,fengyansong}@pku.edu.cn`
`laiyx@ouchn.edu.cn`

Abstract

FEVEROUS is a fact extraction and verification task that requires systems to extract evidence of both sentences and table cells from a Wikipedia dump, then predict the veracity of the given claim accordingly. Existing works extract evidence in the two formats separately, ignoring potential connections between them. In this paper, we propose a Unified Evidence Extraction model (UniFEE), which uses a mixed evidence graph to extract the evidence in both formats. With the carefully-designed unified evidence graph, UniFEE allows evidence interactions among all candidates in both formats at similar granularity. Experiments show that, with information aggregated from related evidence candidates in the fusion graph, UniFEE can make better decisions about which evidence should be kept, especially for claims requiring multi-hop reasoning or a combination of tables and texts. Thus it outperforms all previous evidence extraction methods and brings significant improvement in the subsequent claim verification step.



Claim: John Laurie has collaborated with many directors such as Alfred Hitchcock, Michael Powell, and Laurence Olivier, but his most famous role is Private Frazer in a film written and directed by Alfred Hitchcock.

Evidence Sentences

[John Laurie] Laurie appeared in scores of feature films with directors such as Alfred Hitchcock, Michael Powell, and Laurence Olivier, generally playing bit-parts or supporting roles rather than leading roles.
[*Dad's Army* (1971 film)] Directed by Norman Cohen, Dad's Army was filmed between series three and four and was based upon material from the early episodes of the television series.

Evidence Cells

[John Laurie]

Year	**Film**	**Role**
1970	Step Laughing Into the Grave	
1971	Dad's Army	Private Frazer
	The Abominable Dr. Phibes	Darrow
1976	Crime Casebook	George Winterman / Sellens
1979	The Prisoner of Zenda	Archbishop

Figure 1: An example in FEVEROUS. The green rectangle shows the gold evidence sentences, while the blue one is the gold evidence table, where the gold evidence cells are underlined.

1 Introduction

FEVEROUS (Aly et al., 2021) is a fact extraction and verification task over structured and unstructured information. Models should extract fine-grained evidence in two formats, namely, sentences and table cells, from a Wikipedia dump and verify the given claim accordingly.

Many previous works focus on the claim verification procedure. They propose various methods to fuse evidence in different formats while leaving the evidence extraction within each format separately (Kotonya et al., 2021; Malon, 2021; Bouziane et al., 2021; Hu et al., 2022). For those efforts on evidence extraction, Saeed et al. (2021) use a neural re-ranker to refine page retrieval. Bouziane et al. (2021) propose a reinforced adaptive retrieval embedding paradigm, but they ignore the possible connections among evidence either in the same format or across different formats. However, there are many gold evidence pieces with fewer lexical or semantic overlaps with the claim, while their necessity is mainly determined by other evidence. Without considering evidence connections during extraction, models may be prone to miss these evidence candidates, thus, propagating errors to subsequent claim verification.

Figure 1 shows an example from FEVEROUS. To refute the given claim, models should find that *Private Frazer* is a role in the film *Dad's Army*, and recognize that its director is *Norman Cohen* instead of *Alfred Hitchcock*. However, the overlap between the claim and the required evidence sentence *Directed by Norman Cohen ...* is only two words:

---

[*] The first author and the second author contribute equally to this work.

[†] Corresponding Author.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1150–1160
May 2-6, 2023 ©2023 Association for Computational Linguistics

directed by, without any key entities, which is generally not strong enough to recognize its usefulness. With the evidence cells in the table, one can find that *Private Frazer* is a role in the film *Dad's Army*, which links the evidence sentence and the given claim. Therefore, for the evidence extraction step it is essential to encourage interactions between evidence pieces of different formats.

Meanwhile, evidence in the same format also provides context information to each other, especially for table cells. In Figure 1, the cell *Dad's Army* is not mentioned in the claim, but the column header, *Film*, and the cell in the same row, *Private Frazer*, are strong clues appearing in the given claim, which implies that the cell *Dad's Army* is highly likely to be useful evidence. We argue that models should allow the evidence to accumulate context information from all closely-associated evidence in both formats to extract comprehensive evidence sets for claim verification.

In this paper, we propose a Unified Evidence Extraction model (UNIFEE), which exploits graph attention networks (GAT) to encourage interactions among connected candidates during evidence extraction. We design a novel graph structure, which accommodates all evidence candidates and introduces column nodes to allow evidence candidates of different formats to interact with each other in a similar granularity. This graph also facilitates the interactions among cells in several nested tables with better representation of the table layout. Compared to previous flatten-based methods, our method exploits the structural relationship among cells in complicated tables and enhances the interactions between evidence in both formats.

Experiments on FEVEROUS show that our UNIFEE improves the evidence extraction performances and further boosts the final fact verification scores significantly. The ablation experiments and case study demonstrate the effectiveness of the proposed evidence extraction method and our graph designs. Our code is released to the public for further exploration. [1].

Our contributions can be summarized as follows:

- We propose a novel evidence extraction method, featuring a mixed graph structure with carefully designed column nodes and layout-aware table representations. Our method enables early and thorough evidence interactions within one format or across formats at a similar granularity.

- Experiments show that our method outperforms previous works by a large margin on both the evidence extraction and the final verification results. Thorough analysis shows that our model can recall more multi-hop evidence and also evidence with fewer lexical overlaps with the given claim.

2 Our Model

We take the widely-adopted three-step pipeline for FEVEROUS (Aly et al., 2021), which consists of retrieving pages from Wikipedia dump, extracting sentences and table cells as evidence from the pages, and predicting the veracity label of the given claim based on the evidence set. In this work, we focus on the second step, i.e., evidence extraction. We use the DRQA (Chen et al., 2017) based document retrieval method and the dual-channel verification method proposed by Hu et al. (2022) for the remaining two steps.

Formally, during evidence extraction, n'_s sentences and n_t tables are extracted from n_p retrieved passages by a pair-wise classification model firstly. This pre-processing step narrows the search space of sentences and table cells from thousands to a few dozens, which makes the mixed graph lighter. Then, n_c cells and n_s sentences are extracted by our proposed Unified Evidence Extraction model (UNIFEE). These fine-grained evidence pieces are used for fact verification later.

In UNIFEE, we design a mixed evidence graph (§2.1) to organize evidence of two formats. Then, a graph neural network is applied to the graph to accumulate contextualized information from all evidence candidates with similar granularity, and identify useful evidence from a unified perspective (§2.2).

2.1 Mixed Evidence Graph

There are three kinds of nodes in our mixed evidence graph. Sentence nodes and cell nodes stands for sentences and cells, e.g., *FC Ararat Yerevan* in table 1, evidence candidates. Besides, we further introduce a novel node type representing table columns , e.g., *Senior career* Apps 10 164 8 26* in table 1. The intuition is that textual evidence is mainly at the sentence level but table cells are generally words or phrases. The unbalanced granularities will hinder their interactions. Therefore, we introduce this column node, to act as a mediator for information transmission between the sentence nodes and the cell nodes.

---

[1] `https://github.com/WilliamZR/unifee`

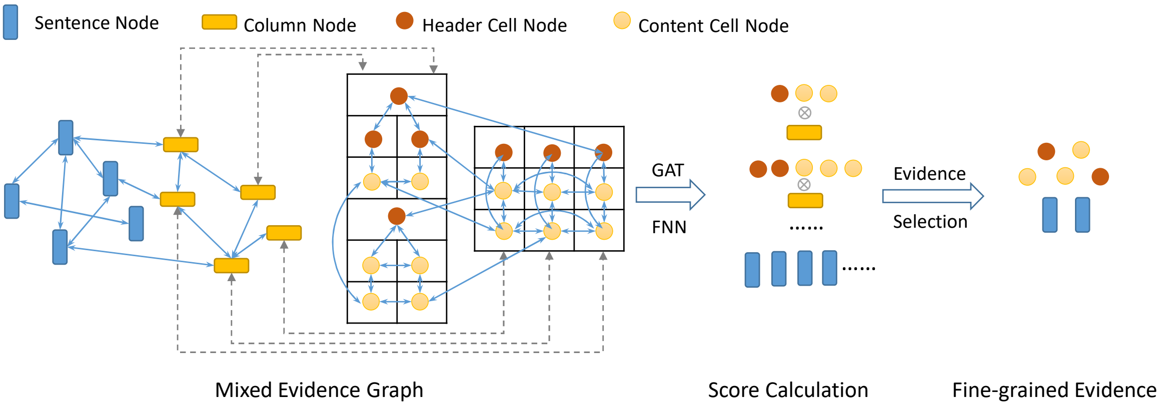


Figure 2: The overview of UNIFEE. The blue lines are edges in the mixed evidence graph, and the gray dash lines indicate which column each column node comes from .

Specifically, the mixed evidence graph G has three types of nodes: sentence nodes, cell nodes (including header cells and content cells), and column nodes, as shown in Figure 2.

The edge connections are elaborated as follows:

Edges among Sentence Nodes and Column Nodes We add bi-directed edges between two nodes v_i and v_j for $v_i, v_j \in V^S \cup V^{Col}$ if and only if the two following conditions:

- v_i and v_j represent evidence in the same Wikipedia page.
- v_i and v_j contain common entities or hyperlinks.

Here the text representation of a column node simply is the concatenation of all cells in that column.

Edges among Cell Nodes Although we expect direct messaging between cells, simply connecting cells in the same rows or the same columns is not enough. The tables in Wikipedia pages are often organized as complicated nested tables. Previous works simply flattening a table or converting cells to strings with human-designed templates will lose delicate structural information layout and cannot reflect the cross-table connections.

Meanwhile, we notice that the tables often have large headers in the middle of themselves in reality, which separates the influenced columns into several parts, and, in this case, cells in the same column could be irrelevant. Disconnecting these cells in the graph will help us remove confusing edges and make the information transmission more reasonable.

Table 1 is a table from the Wikipedia page *Aramais Yepiskoposyan*. We can see that three long headers *Personal Information*, *Senior career* and *National team* cut the table into three parts. The cell organization and contents within each part are consistent while differing a lot across different parts. For example, although the header cell *Personal Information* and the content cell *FC Ararat Yerevan* are in the same column, they do not have a subordinate relationship.

Personal information			
Date of birth	27 September 1968 (age 53)		
Height	1.75 m (5 ft 9 in)		
Position(s)	Midfielder		
Senior career*			
Years	Team	Apps	(Gls)
1986–1991	FC Ararat Yerevan	10	(0)
1992–1997	FC Chernomorets Novorossiysk	164	(8)
1997	FC Kuban Krasnodar	8	(0)
1999–2000	FC Irtysh Pavlodar	26	(2)
2000	FC Spartak Anapa (amateur)		
National team			
1997	Armenia	1	(0)

Table 1: A table instance from the Wikipedia.

We believe that a graph deliberately connecting cells in all table candidates can better model the table structures and connections. According to the table layout and lexical features, we connect two cell nodes v_i^C and v_j^C by bi-directional edges:

- If v_i^C and v_j^C are two content cells in the same column and there is no table-header-like cell between them. For example, the header cells *Senior Career** and *Team* separate the content cells *Midfielder* and *FC Ararat Yerevan*.
- If v_i^C and v_j^C are two cells in the same column, v_i^C is a header cell above v_j^C, and no header cell exists between them.
- If v_i^C and v_j^C are two cells in the same row.
- If v_i^C and v_j^C are two cells sharing at least one entity or hyperlink in their contents.

2.2 Evidence Extraction with Mixed Graphs

In this section, we will show how we extract the evidence set with the mixed evidence graph, which encourages evidence to accumulate context information from closely-associated candidates of both formats and make a better decision to form a comprehensive and accurate evidence set for the verification step.

We formulate the task as: given a claim Q and n retrieved Wikipedia pages $P = \{p_i\}_{i=1}^{n}$, the models should extract a sentence evidence set $E = \langle S, C \rangle$, where S are sentences in P and C are cells in P. The evidence set E can be used to support or refute the claim Q.

Node Initialization The column nodes and cell nodes share a common embedding module, which is more efficient and can build an implicit connection between cells and columns apart from the explicit edges in the graph. Specifically, we feed the claim-table pairs into TAPAS (Herzig et al., 2020), a pre-trained table representation model, to get the embedding of cell nodes and column nodes. Each cell node is initialized by averaging the last hidden states of its tokens, $n(v_i^C)$. $\#\text{TOKENS}\,(cell_i)$ means the number of tokens in $cell_i$. T means the table $cell_i$ belongs to.

$$\mathbf{n}(\mathbf{v_i^C}) = \frac{\sum_{t \in \text{TOKENS}(cell_i)} (\text{Tapas}_t(Q, T))}{\#\text{TOKENS}\,(cell_i)}$$

And each column node is initialized as the mean-pooling of the embeddings of all cells in this column $n(v_i^{Col})$. $\#\text{CELLS}\,(col_i)$ is the number of cells in the column i.

$$\mathbf{n}(\mathbf{v_i^{Col}}) = \frac{\sum_{t \in \text{CELLS}(col_i)} (\text{Tapas}_t(Q, T))}{\#\text{CELLS}\,(col_i)}$$

A one-layer GAT is applied to the graph, which updates the node representations by aggregating representations from their neighbors. $h_i^{(l)}$ is the current representation of node i. W^f, $\vec{a}$ and W are training parameters.

$$\begin{aligned} h_i^{(l+1)} &= \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} W^f h_j^{(l)} \\ \alpha_{ij}^{(l)} &= \text{softmax}_\text{i}\left(e_{ij}^{(l)}\right) \\ e_{ij}^{(l)} &= \text{LeakyReLU}\left(\vec{a}^T \left[W h_i^{(l)} \| W h_j^{(l)}\right]\right) \end{aligned}$$

With a two-layer feed-forward network and a softmax layer, we obtain the probability of each fine-grained evidence candidate being retrieved. The evidence with a score above a predefined threshold is kept.

The loss is calculated as the weighted sum of the sentence loss L^S, column loss L^{Co}, and the cell loss L^{Ce}. We use cross entropy as the loss function for each part separately.

$$L = L^S + \alpha \cdot L^{Co} + \beta \cdot L^{Ce}$$

$$L^S = -\frac{1}{N^S} \sum_{i=1}^{N^S} \log\left(\text{p}\left(\hat{y}^S = y_i^S | Q, S, C\right)\right)$$

$$L^{Co} = -\frac{1}{N^{Co}} \sum_{i=1}^{N^{Co}} \log\left(\text{p}\left(\hat{y}^{Co} = y_i^{Co} | Q, S, C\right)\right)$$

$$L^{Ce} = -\frac{1}{N^{Ce}} \sum_{i=1}^{N^{Ce}} \log\left(\text{p}\left(\hat{y}^{Ce} = y_i^{Ce} | Q, S, C\right)\right)$$

where N^S, N^{Co}, and N^{Ce} are the total numbers of sentence, column, and cell candidates in the training set, respectively. α and β are the coefficients of L^{Co} and L^{Ce}. If a cell or a sentence is in any of the gold evidence sets, we give it the label as 1, otherwise 0. For the labels of the column nodes, if any of the cells in that column is labeled as 1, this column is labeled as 1, otherwise 0.

The score of a sentence candidate is simply calculated as the softmax result of the label 1. In the training procedure, the cell score is calculated similarly. While in the evidence prediction step, the probability of a cell being retrieved is calculated as the product of the score of the cell node itself and the score of the column node it belongs to.

2.3 Document Retrieval and Claim Verification

For the two remaining steps in the whole pipeline, i.e., document retrieval and claim verification, we follow (Hu et al., 2022). For the document retrieval step, we retrieve evidence pages from the Wikipedia dump with a DrQA retriever (Chen et al., 2017) and re-rank the pages with a combination of BM25 and RoBERTa-base (Liu et al., 2019) re-rankers. For the verification step, we perform bidirectional evidence conversions and dual-channel evidence fusion to predict the final veracity label of the given claim.

3 Experiments

3.1 Evaluation Metrics

There are two main official metrics in the FEVEROUS task, i.e., accuracy and FEVEROUS score. The accuracy calculates the proportion of instances for which the model predicts a correct veracity label. The FEVEROUS score considers not only the correctness of the final veracity label but also the extracted evidence set. It calculates the proportion when the extracted evidence set covers one of the gold evidence sets and the predicted veracity label is consistent with the gold label.

Three additional official metrics are used to evaluate the quality of the extracted evidence sets, i.e., Evidence Precision, Evidence Recall, and Evidence F1. Multiple gold evidence sets are provided in FEVEROUS for each claim. A piece of extracted evidence is correct if and only if it is contained by any gold evidence set. For each instance, the Evidence Precision is the proportion of the correct predicted evidence. The overall Evidence Precision is the average score over all instances. The Evidence Recall is the proportion of instances with a correct extracted evidence set. An extracted evidence set is correct if and only if it covers any of the gold evidence sets. The Evidence F1 is the harmonic mean of the Evidence Precision and Evidence Recall.

3.2 Implementation Details

We retrieve $n_p = 5$ pages from the Wikipedia dump for each claim, and further narrow the evidence searching space to $n'_s = 5$ sentences and $n_t = 3$ tables with RoBERTa-base and DrQA model (same as the official baseline), separately. When we choose the final evidence set according to UNIFEE, the threshold for selecting cells and sentences is both 0.01. At most 25 cells are selected for each claim. As for the results, 12.1 cells and 4.7 sentences are extracted for each instance on average.

We use an Adam optimizer (Kingma and Ba, 2015) with a batch size of 4. The learning rates for pre-trained parameters and the others are 10^{-6} and 10^{-4}, respectively. We train UNIFEE for 3 epochs, which takes 21 hours on a NVIDIA A40 GPU. We use Stanza toolkit[2] to identify entities in the claim and evidence for graph construction. We use base-size RoBERTa[3] and TAPAS[4] for sentence and table embedding in evidence extraction. The sentence encoder in claim verification is RoBERTa-large fine-tuned with NLI and verification tasks[5], same as the official baseline.

Details of the FEVEROUS dataset are shown in Appendix § A.

3.3 Baseline Models

The official baseline (Aly et al., 2021) uses DrQA (Chen et al., 2017) to extract sentences and tables. It uses pre-trained models to retrieve cells from flattened tables. EURECOM (Saeed et al., 2021) proposes a neural re-ranker to improve document retrieval. Both NCU (Gi et al., 2021) and Z team (Kotonya et al., 2021) linearize cells to unify evidence element format with sentences. NCU concatenates the claim and the evidence elements as input into RoBERTa for binary classification. Z team constructs a fully-connected graph of evidence element to extract evidence. A next-hop predictor is introduced by Papelo (Malon, 2021) to extract the required evidence. Sentences and cells are retrieved in multi-hops simply based on word overlap with imagined evidence. FaBULOUS (Bouziane et al., 2021) trains a Multi-Hop Dense Retriever (Xiong et al., 2020) to retrieve sentences and cells separately. DCUF(Hu et al., 2022) strengthens the official evidence extraction method with a multi-turn cell retriever.

3.4 Main Results

Our main results are summarized in Table 2. We obtain remarkable improvements over the best previous model, DCUF, in evidence extraction. In the development/test set, the improvements are 3.98%/3.56%, 11.86%/9.77%, and 5.96%/5.22% in evidence precision, recall, and f1 score, respectively. These results suggest that, with the context information accumulated from other candidates in the mixed evidence graph, our model can extract evidence more accurately. We also find that UNIFEE can easily drop irrelevant evidence and keep crucial evidence even though it has less word-overlapping with the claim. Specifically, compared with the official baseline, UNIFEE extracts 25% more gold evidence, with an average word overlap of 15.8%,

---

[2] https://github.com/stanfordnlp/stanza/

[3] https://huggingface.co/roberta-base

[4] https://huggingface.co/google/tapas-base

[5] https://huggingface.co/ynie/roberta-large-snli_mnli_fever_anli_R1_R2_R3-nli

Models	Development/Test				
	FEVEROUS Score	**Accuracy**	**Evidence Precision**	**Evidence Recall**	**Evidence F1**
Official Baseline	19 / 17.73	53 / 48.48	12 / 10.17	30 / 28.78	17 / 15.03
EURECOM	19 / 20.01	53 / 47.79	12 / 13.73	29 / 33.73	17 / 19.52
Z team	– / 22.51	– / 49.01	– / 7.76	– / 42.64	– / 13.12
NCU	29 / 25.14	60 / 52.29	10 / 9.91	42 / 39.07	17 / 15.81
Papelo	28 / 25.92	66 / 57.57	– / 7.16	– / 34.60	– / 11.87
FaBULOUS	30 / 27.01	65 / 56.07	8 / 7.73	43 / 42.58	14 / 13.08
UNIFEE*	**43.48 / 39.36**	**72.35 / 62.41**	**19.04 / 18.35**	**55.08 / 53.87**	**28.30 / 27.37**
DCUF	35.77 / 33.97	72.91 / 63.21	15.06 / 14.79	43.22 / 44.10	22.34 / 22.15
UNIFEE	**44.86 / 41.50**	**73.67 / 65.04**	**19.04 / 18.35**	**55.08 / 53.87**	**28.30 / 27.37**

Table 2: Model performance on the development set and the test set. UNIFEE* means UNIFEE with the verification model of the official baseline instead of DCUF's.

far less than the 28.9% word overlap averagely in the gold evidence extracted by the official baseline.

Our model also achieves better performance on the fact verification metrics of FEVEROUS. Compared to previous state-of-the-art model DCUF, UNIFEE obtains an improvement of 9.09%/7.53% on the FEVEROUS score and 0.76%/1.83% on accuracy at the development/test set. Moreover, we experiment with a simpler verification model taken from the official baseline instead of that from DCUF to see the effectiveness of our evidence extraction step (UNIFEE*). Compared to Bouziane et al. (2021), Kotonya et al. (2021) and Hu et al. (2022) , who use complex verification models with template-based format conversion and(or) dual-channel setting, UNIFEE* still achieves better results. These results suggest that , with a high-quality evidence set, even a simple verification model can achieve good performance.

We also measure the improvement of our model on instances of different challenges in the dev set. Compared to the official baseline, UNIFEE shows an improvement of 65.93% (from 21% to 35%) on instances requiring multi-hop reasoning and 45.32% (from 41% to 59%) on instances requiring the combination of tables and texts, far above the average increase ratio 35.84%. It shows that with information accumulated from connected evidence candidates, UNIFEE can recall more required evidence within a limited evidence size.

Meanwhile, compared to the previous models using multi-turn retriever (Malon, 2021; Bouziane et al., 2021; Hu et al., 2022) for evidence extraction, our method does not require iterative running to retrieve multi-hop evidence and thus is less time-consuming.

3.5 Ablation Study

We evaluate the effectiveness of our Unified Evidence Extraction model with ablation experiments. (1)w/o Fusion Graph: We use the baseline proposed in Aly et al. (2021) to retrieve evidence based on our document retrieval results. (2) w/o Column Nodes: we connect sentences and cells directly if they have common hyperlinks or entities without column nodes as intermediaries. (3) w/o Cross-format Interactions: We deprecate sentence nodes and column nodes, only using cell nodes to retrieve cell evidence. (4)w/o Threshold: We retrieve top 5 sentences and top 25 cells instead of using a threshold.

The results are listed in Table 3. When we apply the evidence extraction model proposed in the official baseline (Aly et al., 2021) on our page retrieval results, the evidence precision, recall, and F1 drop by 3.15%, 14.54%, and 5.47%, respectively. It proves that our UNIFEE extracts the required evidence and removes confusing evidence candidates more confidently with context information accumulated from all evidence candidates of both formats. Meanwhile, the metrics drop a lot when we directly connect sentence candidates and cell candidates, with a decrease of 2.78% on evidence F1. It suggests that with column nodes as intermediaries, UNIFEE allows cross-format evidence interactions at a similar granularity, thus making the information flow over the graph more reasonable and efficient. In the w/o Cross-format Interactions setting, when we remove column nodes and do not add any edges between sentence nodes and cell nodes, the evidence F1 decreases to 25.89%. It shows that context information from evidence of another format is also crucial for the evidence extraction step.

Models	P	R	F1
UNIFEE	**19.04**	55.08	**28.30**
w/o Fusion Graph	15.89	40.54	22.83
w/o Column Nodes	16.74	53.66	25.52
w/o Cross-format Interactions	16.98	54.51	25.89
w/o Threshold	12.48	**56.22**	20.42

Table 3: Ablation study regarding the retrieved evidence on the development set.

We notice that , without cross-format evidence interactions, UNIFEE still outperforms the official baseline by 2.94% and improves the evidence recall by 13.97%, which means that our carefully-designed edges between cells can better represent the layout of complex nested tables, thus improving the cell selection results.

From the w/o Threshold setting, we can see that if we, as many previous works do, directly select the top 25 cells and 5 sentences even whose scores are very low, there will be a slight improvement in the evidence recall, from 55.08% to 56.22%. However, it is at the expense of a 6.56% drop in evidence precision and a 7.88% drop in evidence F1. More irrelevant evidence is selected, which easily confuse the verification model in the final step. With the rough threshold in evidence selection, UNIFEE can dynamically select useful evidence according to their context-aware relevance regardless of the numbers and formats , which helps even simple verification models achieve better performance.

4 Analysis

4.1 Adaptive Evidence Selection

The number of sentences and cells required to verify claims in the FEVEROUS varies a lot. Some claim needs only sentence evidence, some only cell evidence, and many others a combination of both. We find that our UNIFEE can make an adaptive selection of evidence according to the given claim, without being bound to certain formats or a fixed evidence number. With joint evidence extraction in a mixed evidence graph and evidence selection with a threshold, UNIFEE keeps more helpful evidence in the fine-grained evidence retrieval step.

We analyze the relationship between evidence in the gold evidence set and that in the extracted evidence set. The results are shown in Figure 3.

We can see that UNIFEE extracts more cells for claims requiring more cell evidence. For claims that only need sentence evidence, UNIFEE extracts 11 cells on average, and for claims requiring 25 cells, it extracts cells to the maximum limit. We can observe a similar trend for sentence evidence. These results prove that UNIFEE introduces fewer noisy evidence pieces of unwanted formats than the previous works that set a fixed number of each format of evidence, while keeping the required evidence.

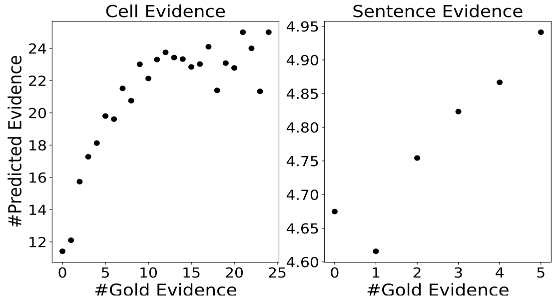


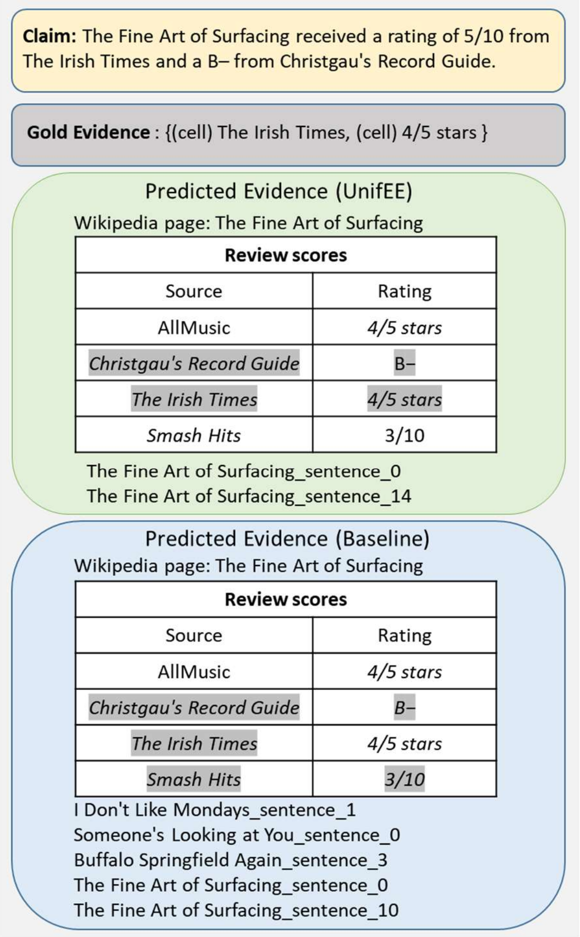
Figure 3: The relationship between the number of gold evidence and the average number of predicted cell/sentence evidence the evidence extraction step.

4.2 Positions of Cell Candidates

We find that the baseline cell extraction model is prone to select cells in the first few lines. If the cell evidence is at the bottom of a table, it is likely to be left out. A possible explanation is that when flattening the table into lines and applying a sequence tagging model to extract evidence cells, these cells are too far from their header cells, making it hard to capture the crucial context information. The average row position of the predicted cells in the baseline is 4.37 on average, much smaller than 6.71, the average of the gold ones. By contrast, it is easier for UNIFEE to exclude the interference of the cell positions thanks to its graph structure, where the cells in different lines are almost equivalent. Therefore, it can extract evidence cells at the bottom of the tables, and the average row position of its predicted cells increases to 6.39.

4.3 Case Study

A case is shown in Figure 4. Models are expected to find the cells *The Irish Times* and *4/5 stars* in the same row to refute the given claim *The Fine Art of Surfacing received a rating of 5/10 from The Irish Times*. The baseline model leaves out the cell *4/5 stars* since it has little word coverage with the given claim. Instead, it selects the distracting cell *3/10*. With the carefully-designed graph structure in UNIFEE, our model can obtain context information from the cell *The Irish Times* in the same row and the cell *Rating* in the same column. Meanwhile,



Claim: The Fine Art of Surfacing received a rating of 5/10 from The Irish Times and a B– from Christgau's Record Guide.

Gold Evidence : {(cell) The Irish Times, (cell) 4/5 stars }

Predicted Evidence (UnifEE)

Wikipedia page: The Fine Art of Surfacing

Review scores	
Source	Rating
AllMusic	*4/5 stars*
Christgau's Record Guide	B–
The Irish Times	*4/5 stars*
Smash Hits	3/10

The Fine Art of Surfacing_sentence_0
The Fine Art of Surfacing_sentence_14

Predicted Evidence (Baseline)

Wikipedia page: The Fine Art of Surfacing

Review scores	
Source	Rating
AllMusic	*4/5 stars*
Christgau's Record Guide	*B–*
The Irish Times	*4/5 stars*
Smash Hits	*3/10*

I Don't Like Mondays_sentence_1
Someone's Looking at You_sentence_0
Buffalo Springfield Again_sentence_3
The Fine Art of Surfacing_sentence_0
The Fine Art of Surfacing_sentence_10

Figure 4: The extracted evidence set by UNIFEE and the official baseline. Cells with a gray background are the predicted ones.

the column where *4/5* is situated will be given a high score with the information from the key cell *Rating*, *B-* and even the distracting cell *3/10*. As a result, the cell *4/5* can be easily retrieved.

Figure 4 also shows that UNIFEE introduces less irrelevant evidence, especially when it comes to the unwanted evidence format. The cells *Christgauś Record Guide* and *B-* can support one aspect of the claim, so our model only introduces two more sentences in need. By contrast, the baseline model retrieves two irrelevant cells, and 5 sentences from many different pages, which may confuse the verification model in the next step.

5 Related Works

In this work, we focus on the evidence extraction step of the fact verification based on table and text, i.e. the FEVEROUS dataset (Aly et al., 2021). Other fact verification datasets only concentrate on unstructured data (Thorne et al., 2018) or a single given table (Wang et al., 2021; Chen et al., 2020; Kwiatkowski et al., 2019). These datasets do not require fine-grained evidence extraction or only extract evidence of a single format.

For evidence sentence extraction, most previous works rank claim-sentence pairs with ESIM (Hanselowski et al., 2018; Zhou et al., 2019) or PLM (Liu et al., 2020; Soleimani et al., 2020). There are also works using the multi-turn retrieving method for multi-hop evidence (Stammbach and Ash, 2020) or training the evidence extraction and claim verification model jointly to relieve the error propagation (Yin and Roth, 2018). For cell evidence extraction, SEM-TAB-FACT (Wang et al., 2021) and FEVEROUS (Aly et al., 2021) are the only two verification datasets requiring fine-grained cell selection to our best known. Acharya (2021) propose to parse the claim into logical forms and identify cells with string matching and dependency parsing. Jindal et al. (2021) use an ensemble of BERT fine-tuned on a single-cell NLI task and a cell-wise similarity algorithm to capture the additional relationship. Many previous works on FEVEROUS flatten each table to a sequence and perform binary sequence labeling to select cells (Aly et al., 2021; Hu et al., 2022). However, these models extract evidence according to every single sentence/cell/table, they neglect the connections of evidence candidates. Meanwhile, they are not tailored for evidence extraction of two formats and cross-format evidence interactions, which are the main challenges of our task.

Many efforts aim at fusing evidence of two formats in FEVEROUS. Kotonya et al. (2021) and Gi et al. (2021) convert the extracted table cells to strings with human-designed templates to get a unified evidence set of sentences. Bouziane et al. (2021) convert each evidence sentence to a small table and verify the claim according to a set of evidence tables. Hu et al. (2022) perform bi-directional format conversion and apply dual-channel encoding to the evidence set. However, their format fusion is only performed in the verification step after evidence extraction. Without evidence fusion in the extraction step, much evidence has been left, especially cell evidence and evidence with fewer overlaps with the given claim.

For works fusing evidence in the extraction step, Kotonya et al. (2021) and Malon (2021) convert cells to strings and treat all evidence candidates as sentences for evidence selection. They neglect the delicate structure of nested tables and lose much context information after the format conversion.

Our proposed UNIFEE introduces column nodes and layer-aware table representations, which can gather complex table information and facilitate cross-format interactions in a similar granularity in the evidence extraction step.

6 Conclusion

In this paper, we propose a Unified Evidence Extraction model (UNIFEE) for fact extraction over structured and unstructured data. UNIFEE adopts a mixed evidence graph to encourage evidence interactions among evidence candidates in both formats without manually designed conversion rules. Experiments on the FEVEROUS benchmark demonstrate that, with UNIFEE, a simple claim verification model outperforms previous SOTA results by a large margin. Further analysis shows that UNIFEE enhances the contextualized modeling of cells in complicated nested tables, thus largely improving the evidence extraction performance.

7 Acknowledgements

This work is supported in part by the NSFC Grants (No.62161160339, 62206070). For any correspondence, please contact Yansong Feng.

8 Limitations

Although we consider evidence interactions in the evidence extraction step and find out required evidence with less overlapping with the claim, it is hard for our method to recall multi-hop evidence in different pages since these pages are left out in the document retrieval step.

Apart from cell evidence and sentence evidence, there is a small proportion of evidence in the FEVEROUS dataset whose type is *table caption*, *list* or so. We simply ignore evidence of these types in the evidence retrieval step. To further improve the quality of evidence extraction step, we should also take these evidence types into consideration.

Another limitation is that the instances of the three veracity labels is unbalanced. From the details of each split shown in Appendix A, only 3% of the training split is labelled NEI, which makes it hard for models to learn predicting this class accurately. We have not tried solving this problem yet.

References

Kaushik Acharya. 2021. KaushikAcharya at SemEval-2021 task 9: Candidate generation for fact verification over tables. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 1271–1275, Online. Association for Computational Linguistics.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13, Dominican Republic. Association for Computational Linguistics.

Mostafa Bouziane, Hugo Perrin, Amine Sadeq, Thanh Nguyen, Aurélien Cluzeau, and Julien Mardas. 2021. FaBULOUS: Fact-checking based on understanding of language over unstructured and structured information. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 31–39, Dominican Republic. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

In-Zu Gi, Ting-Yu Fang, and Richard Tzong-Han Tsai. 2021. Verdict inference with claim and retrieved elements using roberta. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 60–65.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Nan Hu, Zirui Wu, Yuxuan Lai, Xiao Liu, and Yansong Feng. 2022. Dual-channel evidence fusion for fact verification over texts and tables. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5232–5242, Seattle, United States. Association for Computational Linguistics.

Aditya Jindal, Ankur Gupta, Jaya Srivastava, Preeti Menghwani, Vijit Malik, Vishesh Kaushik, and Ashutosh Modi. 2021. BreakingBERT@IITK at SemEval-2021 task 9: Statement verification and evidence finding with tables. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 327–337, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Neema Kotonya, Thomas Spooner, Daniele Magazzeni, and Francesca Toni. 2021. Graph reasoning with context-aware linearization for interpretable fact extraction and verification. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 21–30, Dominican Republic. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics.

Christopher Malon. 2021. Team papelo at FEVEROUS: Multi-hop evidence pursuit. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 40–49, Dominican Republic. Association for Computational Linguistics.

Mohammed Saeed, Giulio Alfarano, Khai Nguyen, Duc-Hong Pham, Raphael Troncy, and Paolo Papotti. 2021. Neural re-rankers for evidence retrieval in the feverous task. *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*.

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. Bert for evidence retrieval and claim verification. *Advances in Information Retrieval*, 12036:359.

Dominik Stammbach and Elliott Ash. 2020. e-fever: Explanations and summaries for automated fact checking. *Proceedings of the 2020 Truth and Trust Online (TTO 2020)*, pages 32–43.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Nancy X. R. Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021. SemEval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (SEM-TAB-FACTS). In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 317–326, Online. Association for Computational Linguistics.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, et al. 2020. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Wenpeng Yin and Dan Roth. 2018. TwoWingOS: A two-wing optimization strategy for evidential claim verification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 105–114, Brussels, Belgium. Association for Computational Linguistics.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. GEAR: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901, Florence, Italy. Association for Computational Linguistics.

A Statistics of the FEVEROUS dataset

FEVEROUS[6] is an open-domain dataset based on English Wikipedia which contains 95.6 million sentences and 11.8 million tables. The dataset has 87,026 claims, with an average length of 25.3. An average of 1.4 sentences and 3.3 cells (0.8 tables) are required to verify each claim. Only text evidence is required in 34,963 instances, only tables in 28,760 instances, and both formats are required in 24,667 instances. 49,115 instances are marked as SUPPORTS, 33,669 as REFUTES, and the remaining 4,242 are marked as NEI. Table 4 displays specific label and evidence distributions in each split.

	Train	**Dev**	**Test**
Supported	41,835(59%)	3,908(50%)	3,372 (43%)
Refuted	27,215(38%)	3,481(44%)	2,973 (38%)
NEI	2,241 (3%)	501 (6%)	1,500 (19%)
Total	71,291	7,890	7,845
Sentences	31,607(41%)	3,745(43%)	3,589 (42%)
Cells	25,020 (32%)	2,738(32%)	2,816 (33%)
Sentence+Cells	20,865 (27%)	2,468 (25%)	2,062 (24%)

Table 4: Details of each split in FEVEROUS

---

[6] `https://fever.ai/dataset/feverous.html`

MiniALBERT: Model Distillation via Parameter-Efficient Recursive Transformers

Mohammadmahdi Nouriborji[2†], **Omid Rohanian**[1,2†], **Samaneh Kouchaki**[3], **David A. Clifton**[1,4]
[1]Department of Engineering Science, University of Oxford, Oxford, UK
[2]NLPie Research, Oxford, UK
[3]Dept. Electrical and Electronic Engineering, University of Surrey, Guildford, UK
[4]Oxford-Suzhou Centre for Advanced Research, Suzhou, China
`{m.nouriborji,omid}@nlpie.com`
`samaneh.kouchaki@surrey.ac.uk`
`{omid.rohanian,david.clifton}@eng.ox.ac.uk`

Abstract

Pre-trained Language Models (LMs) have become an integral part of Natural Language Processing (NLP) in recent years, due to their superior performance in downstream applications. In spite of this resounding success, the usability of LMs is constrained by computational and time complexity, along with their increasing size; an issue that has been referred to as 'overparameterisation'. Different strategies have been proposed in the literature to alleviate these problems, with the aim to create effective compact models that nearly match the performance of their bloated counterparts with negligible performance losses. One of the most popular techniques in this area of research is model distillation. Another potent but underutilised technique is cross-layer parameter sharing. In this work, we combine these two strategies and present MiniALBERT, a technique for converting the knowledge of fully parameterised LMs (such as BERT) into a compact recursive student. In addition, we investigate the application of bottleneck adapters for layer-wise adaptation of our recursive student, and also explore the efficacy of adapter tuning for fine-tuning of compact models. We test our proposed models on a number of general and biomedical NLP tasks to demonstrate their viability and compare them with the state-of-the-art and other existing compact models. All the codes used in the experiments are available at `https://github.com/nlpie-research/MiniALBERT`. Our pre-trained compact models can be accessed from `https://huggingface.co/nlpie`.

1 Introduction

Following the introduction of BERT (Devlin et al., 2019), generic pre-trained Language Models (LMs) have started to dominate the field of NLP. Virtually all state-of-the-art NLP models are built on top of some large pre-trained transformer as a backbone and are subsequently fine-tuned on their target dataset. While this pre-train and fine-tune approach has resulted in significant improvements across a wide range of NLP tasks, the widespread use of resource-exhaustive and overparameterised transformers has also raised concerns among researchers about their energy consumption, environmental impact, and ethical implications (Strubell et al., 2019; Bender et al., 2021).

As a response to this, different approaches have appeared with the aim to make large LMs more efficient, accessible, and environmentally friendly. Model compression is a line of research that has recently received considerable attention. It involves encoding a larger and slower but more performant model into a smaller and faster one with the aim to retain much of the former's performance capability (Bucilua et al., 2006). Knowledge distillation (Hinton et al., 2015), quantisation (Shen et al., 2020), and pruning (Ganesh et al., 2021) are three examples of such methods.

Adapter modules (Bapna and Firat, 2019; He et al., 2021) are recently introduced as an effective mechanism to address the parameter inefficiency of large pre-trained models. In this approach, several 'bottleneck adapters'(Houlsby et al., 2019a) are embedded inside different locations within the original network. During fine-tuning, the parameters of the

---

[†] The two authors contributed equally to this work.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1161–1173
May 2-6, 2023 ©2023 Association for Computational Linguistics

original model are kept fixed, and for each new task only the adapters are fine-tuned. This only adds a small number of parameters to the overall architecture and allows for a much faster and more efficient fine-tuning on different downstream tasks.

Another approach to improve efficiency of LM-based transformers is shared parameterisation, which was popularised by ALBERT (Lan et al., 2019). While the original formulation of transformers (Vaswani et al., 2017) employs full parameterisation wherein each model parameter is independent of other modules and used only once in the forward pass, shared parameterisation allows different modules of the network to share parameters, resulting in a more efficient use of resources given the same parameterisation budget. However, a common downside of this approach is slower inference time and reduced performance. Ge and Wei (2022) posits two different parameterisation methods as an attempt to address the compute and memory challenges of transformer models and explores layer-wise adaptation in an encoder-decoder architecture. These methods exploit cross-layer parameter sharing in a way that would allow for the model to be utilised on mobile devices with strict memory constraints while achieving state-of-the-art results on two seq2seq tasks for English.

In this work, we exploit some of the above approaches to create a number of compact and efficient encoder-only models distilled from much larger language models. The contributions of this work are as follows:

- To the best of our knowledge, we are the first to compress fully parameterised large language models using recursive transformers (i.e. ALBERT-like models that employ full parameter sharing).
- We demonstrate the effectiveness of our pre-trained bottleneck adapters by merely fine-tuning them on downstream tasks while still achieving competitive results.
- We present several light-weight transformers with parameters ranging from 12M for the smallest to 32M for the largest. These models are shown to perform at the same level with their fully parameterised versions.
- Finally, we evaluate our models on a wide range of tasks and datasets on general and biomedial NLP datasets.

2 Background

2.1 LM-based Transformers and Computational Complexity

Ever since the introduction of the transformer architecture (Vaswani et al., 2017), large LM-based transformers such as BERT (Devlin et al., 2019) have become increasingly more popular in NLP and lie at the heart of most state-of-the-art models. A transformer is primarily composed of a number of transformer blocks stacked on top of one another. BERT_{Base} , for instance, consists of 12 of these blocks. The most important component in a block is the multi-head self-attention module. To be useful for language tasks, transformers are pre-trained using a number of self-supervised auxiliary tasks (Xia et al., 2020); these usually include some variation of Language Modelling (LM) and an optional sentence-level prediction task. Examples of the former include Masked Language Modelling (MLM) and Casual Language Modelling (CLM). For the latter, BERT uses Next Sentence Prediction (NSP) and ALBERT (Lan et al., 2019) employs Sentence Order Prediction (SOP).

The standard approach to utilise these pre-trained models is to fine-tune them on a target task. Given N as the sequence length, the computational and time complexity of self-attention is N^2 (Keles et al., 2022). In recent years, different approaches have appeared in the literature to address this bottleneck by modifying the self-attention operation in order to improve the general efficiency of transformers (with different performance trade-offs). Tay et al. (2020) surveys the most common approaches to develop what is referred to as 'efficient transformers'.

The magnitude of the parameters of LM-based transformers is another significant issue that restricts their use. With new releases like GPT-3 and MT-NLG (Smith et al., 2022) that feature hundreds of billions of parameters, these models have become increasingly overparameterised due to the large number of layers and embedding sizes (Rogers et al., 2020).

2.2 Model Distillation

The overparameterisation issue has motivated research into developing methods to compress larger models into smaller and faster versions that perform reasonably close to their larger counterparts. Knowledge distillation (Hinton et al., 2015) is a prominent method that intended to distill a

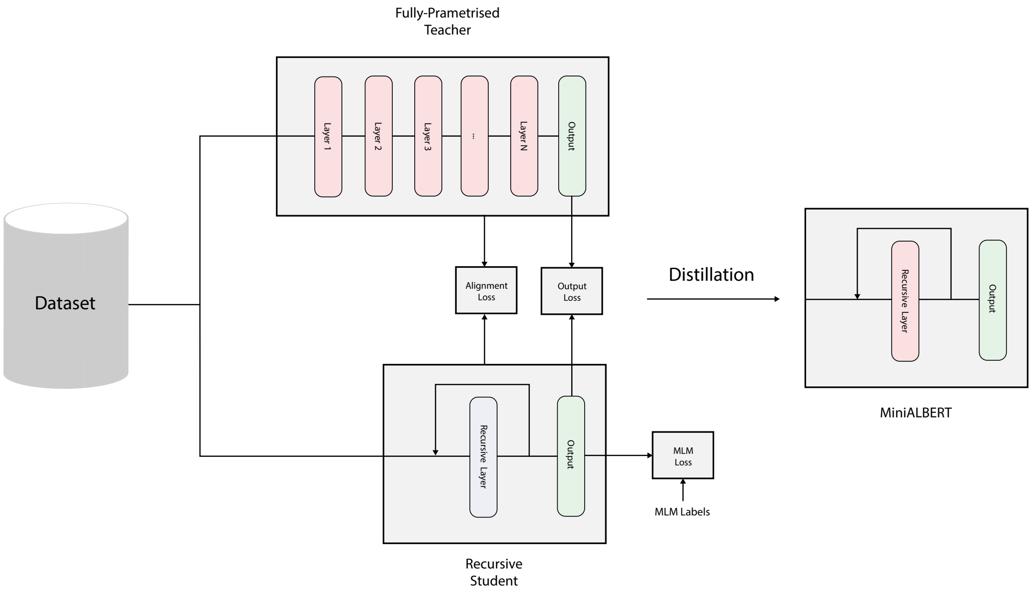


Figure 1: The layer-to-layer distillation procedure proposed for distilling the knowledge of a fully-parameterised teacher into a compact recursive student. While the teacher has fully parameterised layers, the recursive student has only one layer and the output is fed back into the same layer repeatedly. Despite this compact structure, our proposed distillation procedure is designed to align the output of each iteration of the recursive student with a particular layer of the fully-parameterised teacher, as if the student had fully-parameterised layers. Additional losses, namely, Output Loss, and MLM Loss, as shown above, are used for further knowledge distillation.

lightweight 'student' model from a larger 'teacher' network by using the outputs of the teacher netwrok as soft labels. Distillation can either be done task-specifically during fine-tuning, or task-agnostically by mimicking the MLM outputs or the intermediate representations of the teacher prior to the fine-tuning stage. The latter is more flexible and computationally less expensive (Wang et al., 2020). DistilBERT is a well-known example of a distilled model derived from BERT which is claimed to be 40% smaller in terms of parameters and 60% faster while retaining 97% of BERT's performance on a range of language understanding tasks (Sanh et al., 2019).

2.3 Efficient Fine-tuning Approaches

As discussed in Sec 2.1, LM-based transformers involve a large number of parameters and they are often fine-tuned on a target dataset. However, fine-tuning could become time-consuming as the size of the datasets grow. Different techniques exist in the literature to alleviate this bottleneck during fine-tuning. In this section we explore two of these techniques, namely, prompt tuning and bottleneck adapters.

Prompt tuning (Lester et al., 2021) is a technique in which the weights of a language model are kept frozen during the fine-tuning stage and fine-tuning is reformulated as a cloze-style task. Similar to T5, prompt tuning regards all tasks as a variation of text generation and conditions the generation using 'soft prompts'. A typical prompt consists of a text template with a masked token and a set of candidate label words to fill the mask. This turns the target task into another MLM objective in which the right candidates are chosen and soft prompts are learned. This method is especially useful for few-shot learning scenarios where there are not many target labels available for standard fine-tuning.

Bottleneck Adapters (BAs) (Houlsby et al., 2019b; Pfeiffer et al., 2021; Rücklé et al., 2020; Pfeiffer et al., 2020) are another mechanism used during fine-tuning to enhance efficiency of training. Each BA block consists of a linear down-projection, non-linearity, and up-projection along with residual connections. Several of these adapters are placed after the feed-forward or attention modules in a transformer. Similar to prompts, only the BAs are trained during fine-tuning.

2.4 Parameter Sharing via Recursion

Weight sharing is a strategy intended to reduce the overall number of parameters in a model. Lan et al.

(2019) introduced cross-layer parameter sharing in a recurrent-like encoder-only architecture where instead of having several transformer blocks with different parameters, there is only one transformer block whose outputs are recursively fed into itself a number of times. This drastically reduces the number of unique parameters in the model.

Edgeformer (Ge and Wei, 2022) is a recent work towards development of parameter-efficient encoder-decoder models specialised for on-device seq2seq generation. EdgeFormer employs two novel approaches for cost-effective parameterisation which improve on standard cross-layer parameter sharing. In addition to efficient parameterisation, Edgeformer explores the use of layer-wise adaptation in encoder-decoder models. To the best of our knowledge, however, the use of layer-wise adaptation in recursive encoder-only models (Sec 3.2) is yet to be explored.

3 Methods

In this work, we introduce a method to distil the knowledge of a fully parameterised transformer into a lightweight efficient recursive transformer via layer-to-layer distillation. We also experiment with layer-wise adaption of the recursive transformer via bottleneck adapters and factorise the embedding layer for extra parameter saving. In this section, we explain each component of our compact models in detail.

3.1 The Fully Parameterised Transformer

For each layer i, let the multi-head attention and feed-forward blocks be $f^i_{att}(x)$ and $f^i_{mlp}(x)$, respectively. The output of each layer is computed as follows:

$$O_i = f^i_{mlp}(f^i_{att}(O_{i-1})) \tag{1}$$

where O_i is the output of the i^{th} layer of the transformer.

3.2 The Recursive Student

The output of the recursive student is computed as follows:

$$O_i = f_{mlp}(f_{att}(O_{i-1})) \tag{2}$$

where O_i is the output of the i^{th} iteration of the recursive transformer. Note that unlike Equation 1, the multi-head attention and feed-forward blocks in this case are layer-agnostic, i.e. the same layers are used recursively.

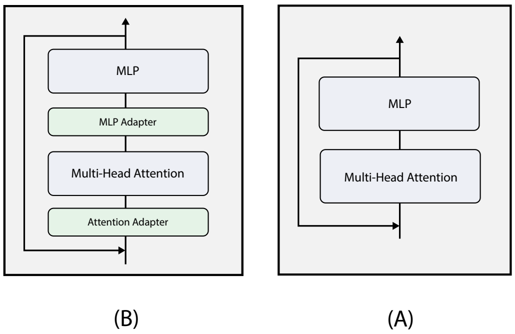


Figure 2: The two recursive students proposed in this work. (A) is a simple recursive student that employs cross-layer parameter sharing, which means that the multi-head attention and MLP blocks are shared across all layers. (B) is a recursive student with cross-layer parameter sharing which additionally uses layer-wise adaption via bottleneck adapters.

3.3 Recursive Student with Layer-wise Adaption

This formulation is identical to the recursive student defined in Section 3.2, except that, in addition to the shared parameters, here we employ a small number of trainable parameters. This will allow the model to capture distinct features in each iteration, similar to how transformer layers behave in a fully parameterised model. To this end, we use Bottleneck Adapters (BAs) which are small bottleneck blocks followed by residual connections, as defined in the following equation:

$$\phi(X) = W_{up}\ \sigma(W_{down}\ X) + X \tag{3}$$

where $\phi(.)$ represents the BA and $\sigma(.)$ is a nonlinearity function such as *RELU* or *GELU*. The recursive student with layer-wise adaption can be formulated as follows:

$$O_i = f_{mlp}(\phi^i_{mlp}(f_{att}(\phi^i_{att}(O_{i-1})))) \tag{4}$$

where $\phi^i_{att}(.)$ and $\phi^i_{mlp}(.)$ are the BAs for the multi-head attention and feed-forward blocks of the i^{th} iteration of the recursive transformer.

3.4 Embedding Factorisation

Following the work of ALBERT (Lan et al., 2019), instead of a full-rank embedding matrix, we use a low-rank matrix with size $|V| \times r$ where V is the vocabulary and r is the rank of the embedding matrix. We additionally use a projection weight with the size of $r \times d$ where d is the hidden dimension of the transformer. This is set to 768 in our experiments. Factorisation can be mathematically expressed as

$$E = E_{low}\, W_e \tag{5}$$

where $E_{low} \in \mathbb{R}^{|V|\times r}$, $W_e \in \mathbb{R}^{r\times d}$ and $E \in \mathbb{R}^{|V|\times d}$. In our experiments, we employ factorisation with ranks 312 and 128, and explore initialising the 312 versions with pre-trained embeddings obtained from the TinyBERT (Jiao et al., 2019) for the general models and TinyBioBERT (Rohanian et al., 2022) for the biomedical models.

In our experiments, we employed factorisation with ranks of 312 and 128 and initialised the 312 versions with pre-trained embeddings from TinyBERT (Jiao et al., 2019) for general models and TinyBioBERT (Rohanian et al., 2022) for biomedical models.

3.5 Distillation Procedure

We use three different loss terms, namely, MLM, alignment, and output. The MLM loss, which is the original loss used in Masked Language Modelling, is defined as follows:

$$L_{MLM}(X,Y) = \sum_{n=1}^{N} CE(f_s(X)^n, Y^n) \tag{6}$$

where X denotes the the model's input, N the number of input tokens, CE the cross-entropy loss function, Y^n the one-hot encoded label for the n^{th} token[1], and $f_s(X)^n$ the output distribution of the student for the n^{th} token. Y^n and $f_s(X)^n$ are both $|V|$-dimensional.

Because the number of layers in the student network is less than that of the teacher, alignment is typically achieved by comparing the student's layers to a subset of the teacher's. However, in our case, the student is recursive and lacks multiple layers. Therefore, in each iteration, we map the output of the student's layer to a specific layer of the teacher, i.e. the first iteration is considered the first layer, the second iteration is the second layer, and so on. The alignment loss is a combination of two terms, namely, attention and hidden losses. Attention loss is used to align the student's attention maps with those of the teacher and is defined as follows:

$$L_{att}(\hat{A}, A) = \frac{1}{HN} \sum_{h=1}^{H} \sum_{n=1}^{N} D_{KL}(\hat{A}_n^h \,||\, A_n^h) \tag{7}$$

$\hat{A}$ and A are the inputs to the loss function, and they correspond to the attention maps of a certain layer of the student and its associated layer of the teacher, respectively. The cosine-based hidden loss is used to align the hidden states of the student and teacher, and is defined as:

$$L_{hidden}(\hat{H}, H) = \frac{1}{N} \sum_{n=1}^{N} 1 - \psi(\hat{H}^n, H^n) \tag{8}$$

where $\hat{H}$ and H are the hidden states of a particular layer in the student and teacher networks. The ψ function denotes cosine similarity[2]. The alignment loss is defined as:

$$\begin{aligned} L_{align}(\hat{A}, A, \hat{H}, H) = \quad & L_{att}(\hat{A}, A) \\ & + L_{hidden}(\hat{H}, H) \end{aligned} \tag{9}$$

The output loss is based on KL divergence and is inteneded to align the output distribution of the student with the teacher on the MLM objective. This loss term is defined as below:

$$L_{out}(X) = \sum_{n=1}^{N} W_n D_{KL}(f_s(X)^n \,||\, f_t(X)^n) \tag{10}$$

where $f_s(X)^n$ and $f_t(X)^n$ are the output distributions of the student and teacher for the n^{th} token, respectively. W_n is 1 if the n^{th} token is masked and 0 otherwise. This ensures that only the masked tokens will contribute to the loss.

The complete layer-to-layer distillation loss used in this study is expressed by the following equation:

$$\begin{aligned} L(X, Y, & A_s, A_t, H_s, H_t) = \\ & \lambda_1\, L_{MLM}(X,Y) \\ & + \lambda_2 \sum_{l=1}^{L} L_{align}(A_s^l, A_t^{g(l)}, H_s^l, H_t^{g(l)}) \\ & + \lambda_3\, L_{out}(X). \end{aligned} \tag{11}$$

A_s and A_t in Equation 11 are collections of attention maps for the student and teacher, respectively. H_s and H_t are sets of hidden states for the student and teacher. L is the number of iterations (layers) of the recursive student. $g(.)$ is a mapping function that connects each iteration of the student to a specific layer of the teacher. Finally, λ_1 to λ_3 are hyperparameters used for controlling the importance of each component of the loss function (we use $\lambda_1 = 1.0$, $\lambda_2 = 3.0$, $\lambda_3 = 5.0$ in our experiments).

---

[1] Y^n is a zero vector if the n^{th} token is not masked.

[2] $\psi(\vec{u}, \vec{v}) = \frac{\vec{u}.\vec{v}}{||\vec{u}||_2 ||\vec{v}||_2}$

Table 1: The DEV set results on the GLUE benchmark. ALBERT$_6$ and ALBERT$_{12}$ denote ALBERT models with 6 and 12 layers respectively, and an embedding size of 128, which are trained on the same data for the same number of training steps as MiniALBERT. DistilBERT$_{base}$ is a DistilBERT model trained with the same distillation setting as Sanh et al. (2019) and for the same number of training steps as MiniALBERT. Adapter denotes layer-wise adaption and EF denotes Embedding Factorisation. The metrics used for reporting the results on each dataset is the same as the official GLUE benchmark. † denotes that the models were trained using adapter tuning in which all weights of the model except bottleneck adapters are kept frozen during fine-tuning. $*$ shows that the bottleneck adapters were initialised randomly since the model has not used bottleneck adapters at the pre-training stage. N/A means that the model did not learn anything from the target dataset. #Params denotes the number of tunable parameters which are used during fine-tuning. Note that for fine-tuning TinyBERT on the downstream tasks, unlike the original paper (Jiao et al., 2019), we do not employ task-specific distillation; this is to ensure the comparison with other models is fair.

Model	Adapter	EF	#Params	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg
BERT-base	-	-	110M	**84.78/84.78**	**87.91**	**91.56**	**92.77**	**57.19**	**89.20**	**91.46**	**74.72**	**83.81**
DistilBERT	-	-	65M	82.17/82.33	87.08	89.47	90.25	53.61	87.20	88.66	66.42	80.79
MobileBERT	-	-	25M	84.03/83.84	87.41	91.17	91.16	51.55	88.12	90.59	66.78	81.62
TinyBERT	-	-	15M	80.22/80.65	84.44	85.31	88.87	17.53	87.77	89.41	66.06	75.58
DistilBERT$_{base}$	-	-	65M	**82.08/82.27**	**86.01**	86.91	**91.28**	**50.33**	83.92	88.06	59.56	**78.93**
ALBERT$_6$	-	✓	11M	76.35/77.01	84.73	84.51	86.12	29.54	82.76	86.39	59.56	74.10
ALBERT$_{12}$	-	✓	11M	78.93/80.03	85.45	**88.12**	86.92	41.73	**86.21**	**90.65**	**63.17**	77.91
MiniALBERT$_{768}$	×	×	31M	80.71/81.80	86.25	88.13	89.79	43.71	**86.93**	89.00	61.37	78.63
MiniALBERT$_{312}$	×	✓	17M	80.55/81.29	86.46	87.46	89.56	45.82	85.51	89.22	62.09	78.66
MiniALBERT$_{768}$	✓	×	32M	**81.04/82.05**	**86.48**	**88.79**	89.79	46.33	86.92	88.46	67.50	79.70
MiniALBERT$_{312}$	✓	✓	18M	80.78/81.67	86.35	88.57	**90.36**	45.50	86.60	88.96	**69.31**	79.78
MiniALBERT$_{128}$	✓	✓	12M	80.64/81.33	86.29	88.08	89.67	**47.93**	86.62	**89.77**	68.95	**79.92**
MiniALBERT$_{768}{}^{\dagger,*}$	×	×	0.9M	74.61/75.72	79.72	80.70	85.20	N/A	83.90	81.72	52.70	68.25
MiniALBERT$_{312}{}^{\dagger,*}$	×	✓	0.9M	74.48/75.70	79.76	82.39	83.48	N/A	80.94	81.22	54.15	68.01
MiniALBERT$_{768}{}^{\dagger}$	✓	×	0.9M	**79.48**/80.06	**85.29**	**87.84**	90.02	42.26	**86.33**	87.74	**67.14**	**78.46**
MiniALBERT$_{312}{}^{\dagger}$	✓	✓	0.9M	79.13/**80.16**	85.27	86.10	89.90	**45.25**	85.34	87.91	65.34	78.26
MiniALBERT$_{128}{}^{\dagger}$	✓	✓	0.9M	78.05/79.66	84.94	87.40	**90.36**	44.72	84.62	**89.08**	66.78	78.40

4 Experiments and Results

We evaluate our general models on the widely used GLUE benchmark (Wang et al., 2018). All the models were pre-trained on four Tesla V100 32GB GPUs with a total batch size of 192 (48 each) and fine-tuning was done using only one Tesla V100. A random seed of 42 was used consistently throughout training for fair comparison. For all of the datasets, in order to do full fine-tuning, we use a learning rate from the set {5e-5, 3e-5, 1e-5}. For large datasets (MNLI, QQP, QNLI, and SST-2), we train models for a maximum of 5 epochs, and up to ten epochs on other datasets. The hyperparameters used for full-finetuning are listed in Table 7.

For adapter-tuning, the learning rate was selected from the set {5e-5, 5e-4, 1e-3}. Models were trained for a maximum of 10 epochs for large datasets and up to 15 epochs for other datasets. Table 8 details the hyperparameters used during adapter tuning on the GLUE benchmark. The results of the baselines and our general models are available in Table 1.

The biomedical models are evaluated on the task of Named Entity Recognition (NER), which is one of the most prominent tasks in biomedical NLP. We use four well-known datasets, namely, NCBI-disease (Doğan et al., 2014), BC5CDR-disease (Li et al., 2016), BC5CDR-chem (Li et al., 2016), and BC2GM (Smith et al., 2008).

We generally follow the same pre-processing pipeline as Rohanian et al. (2022). For biomedical NER, we use pre-processed datasets from Lee et al. (2020). We fine-tune the models using a learning rate from the set {5e-5, 3e-5, 1e-5} and perform adapter-tuning with a learning rate from {5e-5, 5e-4, 1e-3}. The hyperparameters for the biomedical datasets are presented in Tables 9 and 10. The results of the baselines and our biomedical models are available in Table 2.

5 Discussion

We trained our recursive students both with and without adapters and found that generally having adapters would increase the learning capacity of the model since each iteration provides an extra degree of freedom to the model in order to capture a specific type of feature. As shown in Tables 1

Table 2: The results of biomedical NER for the biomedical baselines and our models distilled from BioBERT-v1.1 on the PubMed dataset. BioALBERT$_6$ and BioALBERT$_{12}$ represent ALBERT models with 6 and 12 layers, respectively, and an embedding size of 128. These models were trained for the same number of steps as BioMiniALBERT using the same data. "Adapter" refers to layer-wise adaptation and "EF" stands for Embedding Factorization. The results are reported using the F1-score as the evaluation metric. The notations here are consistent with Table 1.

Model	Adapter	EF	#Params	NCBI-disease	BC5CDR-disease	BC5CDR-chem	BC2GM	Avg
DistilBioBERT	-	-	65M	87.93	85.42	94.53	86.60	88.62
CompactBioBERT	-	-	65M	**88.67**	85.38	94.31	86.71	88.76
TinyBioBERT	-	-	15M	85.22	81.28	92.20	82.52	85.30
BioMobileBERT	-	-	25M	87.21	84.62	94.23	85.26	87.83
BioBERT	-	-	110M	88.62	**86.67**	**94.73**	**87.62**	**89.41**
BioALBERT$_6$	-	✓	11M	**86.07**	**82.00**	**93.19**	**84.51**	**86.44**
BioALBERT$_{12}$	-	✓	11M	**86.07**	81.94	93.11	84.33	86.36
BioMiniALBERT$_{768}$	×	×	31M	87.44	84.40	94.18	86.06	88.02
BioMiniALBERT$_{312}$	×	✓	17M	87.94	84.45	94.03	86.03	88.11
BioMiniALBERT$_{768}$	✓	×	32M	88.02	**84.98**	**94.49**	86.10	**88.39**
BioMiniALBERT$_{312}$	✓	✓	18M	**88.03**	84.75	94.23	**86.14**	88.28
BioMiniALBERT$_{128}$	✓	✓	12M	87.16	84.58	94.20	86.00	87.98
BioMiniALBERT$_{768}{}^{\dagger,*}$	×	×	0.9M	85.61	82.31	92.80	84.89	86.40
BioMiniALBERT$_{312}{}^{\dagger,*}$	×	✓	0.9M	85.98	81.72	91.99	84.65	86.08
BioMiniALBERT$_{768}{}^{\dagger}$	✓	×	0.9M	**87.80**	**84.64**	94.20	**86.02**	**88.16**
BioMiniALBERT$_{312}{}^{\dagger}$	✓	✓	0.9M	87.61	84.55	94.12	85.60	87.97
BioMiniALBERT$_{128}{}^{\dagger}$	✓	✓	0.9M	87.71	84.48	**94.22**	85.87	88.07

and 2, in virtually all of the studies, models with adapters outperformed models of comparable size without adapters. In general, both models with and without adapter have outperformed their baseline by a significant margin (as shown in Tables 2 and 1), demonstrating the effectiveness of the proposed layer-to-layer distillation loss for constructing powerful compact recursive models. Comparison between the attention maps of our trained student and teacher models (Figure 3 in appendix) suggests the specific recursive architecture we have introduced in this work is indeed capable of mimicking the components of a larger model.

In addition, our experiments revealed that utilising adapters in the pre-training stage enables the model to effectively use adapter-tuning with only minor performance drops. However, adapter-tuning in models that have not used adapters in the pre-training stage causes major performance drops. We also found that adapter tuning is nearly 30% faster than full fine-tuning. Therefore, with adapter-tuning, the models can be trained for a larger number of epochs given the same training time, which can potentially increase the performance of the model. In general, a higher learning rate was required for adapter-tuning than for full fine-tuning. For both general and biomedical tasks we found that a learning rate of $5e-4$ results in the best performance, however, in some cases, a higher or lower learning rate such as $5e-5$ or $1e-3$ was deemed better.

Another method explored in this work for parameter saving is Embedding Factorisation. In our experiments, we observed that regardless of drastic parameter reduction, models using this approach are still able to perform on par or even in some cases better than models with full-rank embedding.

6 Ablation Studies

6.1 The Effect of The Alignment Loss

One of the main losses explored in this work for knowledge distillation is the alignment loss as explained in Equation 9. Alignment loss is used for mapping each iteration of the recursive student to a specific layer of the teacher, so the knowledge of each fully-parameterised layer is explicitly encoded into a specific iteration of the recursive student. The alignment loss consists of two losses, one for aligning the attention maps and one for aligning the hidden states. For ablation studies, we trained our best model for 20k steps on the Wikipedia dataset, with different alignment losses, and evaluated the resulting models on the GLUE dataset. The results are shown in Table 3.

As Shown in Table 3, without alignment, the performance of the recursive student drops significantly which shows the importance of using a layer-to-layer distillation technique. Furthermore, we discovered that aligning hidden states is far more

Table 3: Ablation study on the alignment loss. The performance drop is computed based on the difference with the average score of the model with full alignment (i.e. when all the alignment losses are used).

Alignment Type	Performance Drop
Hidden-Only	-0.5
Attention-Only	-3.27
No-Alignment	-5.27

important than aligning attention maps. We even noticed occasional improvements in some tasks by solely aligning hidden states, but the average performance of full alignment remains higher than hidden states alignment.

6.2 The Effect of an Extra Embedding Loss

Following the work of Jiao et al. (2019) and as part of our ablation tests, we investigated employing an extra loss for aligning the embeddings of the recursive student and fully-parameterised teacher. This embedding loss is defined as follows:

$$L_{embed}(\hat{E}, E) = \frac{1}{N}\sum_{n=1}^{N} 1 - \psi(\hat{E}^n, E^n) \quad (12)$$

$\hat{E}$ and E are the inputs to the loss function and represent the embeddings of the student and teacher. The embedding weights are not aligned globally in this formulation; instead, the local embeddings created for each training sample are compared before entering the transformer encoder. For our ablation studies, we trained models on the Wikipedia dataset for 20k steps, with and without this extra alignment loss, and evaluated them on the GLUE benchmark.The average performance of the models are reported in Table 4.

Table 4: Ablation study on the embedding loss. 'Full-rank' denotes models without embedding factorisation, and 'Loss' denotes extra embedding loss during distillation.

Model	Avg Performance
Full-rank	76.01
Full-rank + Loss	75.24
Factorised	76.31
Factorised + Loss	76.14

We discovered that, unlike the trend seen in fully-parameterised models, embedding loss reduces the performance of the student model (Table 4). This difference between the recursive student and the fully parameterised teacher implies that forcing the recursive student's embedding to match that of the fully parameterised teacher may not be beneficial, and that allowing the student to learn embeddings independently works better.

7 Conclusion and Future Works

In this work, we explored distilling the knowledge of fully-parameterised language models into recursive students with cross-layer parameter sharing. We used a layer-to-layer distillation technique to observe the learning capacity of the recursive students compared to their fully-parameterised teachers. We used bottleneck adapters for improving the performance of our recursive students and also assessed the benefits of adapter-tuning at the fine-tuning stage. Furthermore, an embedding factorisation technique was used for additional parameter reduction, which was evaluated with and without an extra distillation loss to match the student's embeddings with the teacher. Finally, by integrating all of the strategies outlined above, we were able to train compact recursive students with no more than 12M parameters, yielding competitive performance on both general and biomedical NLP. In the future, we hope to investigate various parameter-sharing and embedding factorisation strategies, as well as other layer-wise adaption techniques such as prompt-tuning. We would also like to train recursive students with larger hidden sizes and more training iterations to compress massive fully-parameterised models with minor performance drops.

Funding

This work was supported in part by the National Institute for Health Research (NIHR) Oxford Biomedical Research Centre (BRC), and in part by an InnoHK Project at the Hong Kong Centre for Cerebro-cardiovascular Health Engineering (COCHE). OR acknowledges the support of the Medical Research Council (grant number MR/W01761X/). DAC was supported by an NIHR Research Professorship, an RAEng Research Chair, COCHE, and the Pandemic Sciences Institute at the University of Oxford. The views expressed are those of the authors and not necessarily those of the NHS, NIHR, MRC, COCHE, or the University of Oxford.

Limitations

Regardless of the parameter reduction induced by our proposed recursive architecture, the resultant models have the same inference latency and memory complexity as fully-parameterised models of comparable size, which for our models is DistilBERT.

In general, our models' capacity for learning semantic and grammatical knowledge is limited, and they may perform poorly on tasks that necessitate a significant amount of reasoning and understanding, such as Question Answering or Semantic Acceptability. More analysis is required to determine the source of this limitation, i.e. whether it is a result of the architecture used or a consequence of the particular cross-layer sharing method etc.

Ethics Statement

In this study, we aimed to create efficient lightweight versions of large and less accessible NLP models. This area of research aims to make AI/NLP models more readily available, with fewer computational resources required to run them and potentially less negative environmental impact.

This work does not use any private or sensitive data and instead relies on widely used publicly available datasets that have been utilised by other researchers in the field with references provided in the paper for more information. All the codes and models are going to be made available for reproducibility purposes.

References

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

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

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Hassan Sajjad, Preslav Nakov, Deming Chen, and Marianne Winslett. 2021. Compressing large-scale transformer-based models: A case study on BERT. *Transactions of the Association for Computational Linguistics*, 9:1061–1080.

Tao Ge and Furu Wei. 2022. Edgeformer: A parameter-efficient transformer for on-device seq2seq generation. *arXiv preprint arXiv:2202.07959*.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jiawei Low, Lidong Bing, and Luo Si. 2021. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2208–2222, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. Cite arxiv:1503.02531Comment: NIPS 2014 Deep Learning Workshop.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019a. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019b. Parameter-efficient transfer learning for nlp.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Feyza Duman Keles, Pruthuvi Mahesakya Wijewardena, and Chinmay Hegde. 2022. On the computational complexity of self-attention. *arXiv preprint arXiv:2209.04881*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Omid Rohanian, Mohammadmahdi Nouriborji, Samaneh Kouchaki, and David A Clifton. 2022. On the effectiveness of compact biomedical transformers. *arXiv preprint arXiv:2209.03182*.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020. Adapterdrop: On the efficiency of adapters in transformers.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821.

Larry Smith, Lorraine K Tanabe, Cheng-Ju Kuo, I Chung, Chun-Nan Hsu, Yu-Shi Lin, Roman Klinger, Christoph M Friedrich, Kuzman Ganchev, Manabu Torii, et al. 2008. Overview of biocreative ii gene mention recognition. *Genome biology*, 9(2):1–19.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *ACM Computing Surveys (CSUR)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Patrick Xia, Shijie Wu, and Benjamin Van Durme. 2020. Which *BERT? A survey organizing contextualized encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7516–7533, Online. Association for Computational Linguistics.

A Details of the Pre-Training

A.1 General Models

For pre-training our general models we use the English subset of the Wikipedia dataset which is available on the Huggingface platform. For pre-processing the Wikipedia dataset, we use the 'bert-base-uncased' tokeniser and apply a sliding window with a size of 256 tokens and a stride size of 128. Due to computational restrictions, we limit the maximum number of tokenised samples per article to 10, resulting in a total of 21 million training samples of 256 tokens each. We then follow BERT's masking approach, with a masking probability of 15% for each token.

Table 5: Hyperparameters used for pre-training models on Wikipedia dataset

Param	Value
learning rate	$5e-4$
scheduler	Linear
optimiser	AdamW
weight decay	$1e-4$
total batch size	192
warmup steps	5000
epochs	1

A.2 Biomedical Models

We pre-trained the biomedical models using the PubMed Abstracts dataset which consists of 31 million abstracts from PubMed articles. In the pre-processing stage, we employed the 'bert-base-cased' tokeniser with a maximum length of 256 and adhered to the same masking strategy as used in the training of the general models.

Table 6: Hyperparameters used for pre-training models on Wikipedia dataset

Param	Value
learning rate	$5e-4$
scheduler	Linear
optimiser	AdamW
weight decay	$1e-4$
total batch size	192
warmup steps	5000
training steps	100K

B Finetuning Details

C Comparison of Teacher and Student Attention Maps

Figure 3 shows attention maps of a teacher model and one of the proposed students on the input sentence "This is the first book I've ever done". The first and second rows of the student contain four attention heads belonging to the 0^{th} and 4^{th} iteration of the recursive student, respectively. The first and second row of teacher contain attention heads belonging to the 1^{st} and 9^{th} layer of the teacher. During training, these sets of layers have been compared together in order to compute the alignment

Table 7: Hyperparameters used for full fine-tuning of the models on the GLUE benchmark

Param	Value
learning rate	{5e-5, 3e-5, 1e-5}
scheduler	Linear
optimiser	AdamW
weight decay	0.01
batch size	{8, 16, 32}
epochs	{3, 5, 10}

Table 8: Hyperparameters used for adapter-tuning of the models on the GLUE benchmark

Param	Value
learning rate	{5e-5, 5e-4, 1e-3}
scheduler	Linear
optimiser	AdamW
weight decay	0.01
batch size	{8, 16, 32}
epochs	{5, 10, 15}

Table 9: Hyperparameters used for full fine-tuning of the models on the Biomedical datasets

Param	Value
learning rate	{5e-5, 3e-5, 1e-5}
scheduler	Linear
optimiser	AdamW
weight decay	0.01
batch size	16
epochs	5

Table 10: Hyperparameters used for adapter tuning of the models on the Biomedical datasets

Param	Value
learning rate	{5e-5, 5e-4, 1e-3}
scheduler	Linear
optimiser	AdamW
weight decay	0.01
batch size	16
epochs	{5, 10}

loss. Despite the fact that the recursive student has only one layer, it has been able to perfectly mimic its teacher in some of the heads which shows the efficiency of the layer-to-layer distillation loss.

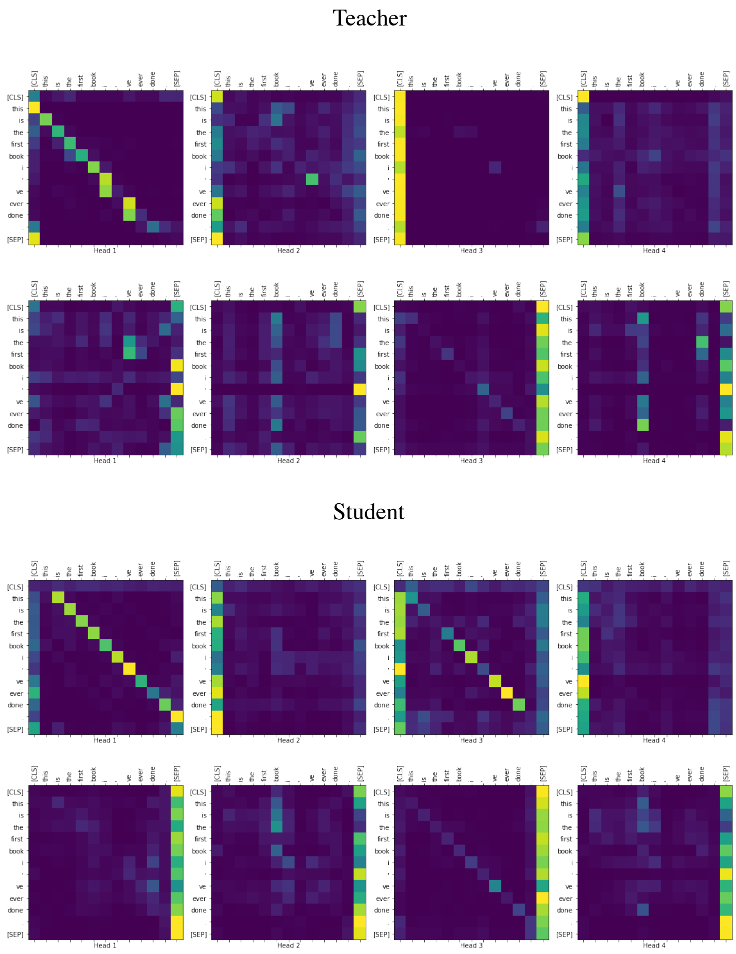


Figure 3: The attention maps for the teacher and one of the proposed recursive students on the input “This is the first book I’ve ever done”.

Multilingual Normalization of Temporal Expressions with Masked Language Models

Lukas Lange[1] **Jannik Strötgen**[1] **Heike Adel**[1] **Dietrich Klakow**[2]
[1] Bosch Center for Artificial Intelligence, Renningen, Germany
[2] Spoken Language Systems (LSV), Saarland University, Saarbrücken, Germany
{Lukas.Lange,Jannik.Stroetgen,Heike.Adel}@de.bosch.com
dietrich.klakow@lsv.uni-saarland.de

Abstract

The detection and normalization of temporal expressions is an important task and preprocessing step for many applications. However, prior work on normalization is rule-based, which severely limits the applicability in real-world multilingual settings, due to the costly creation of new rules. We propose a novel neural method for normalizing temporal expressions based on masked language modeling. Our multilingual method outperforms prior rule-based systems in many languages, and in particular, for low-resource languages with performance improvements of up to 33 F_1 on average compared to the state of the art.

1 Introduction

Temporal tagging consists of the extraction of temporal expressions (TE) from texts and their normalization to a standard format (e.g., *May '22*: `2022-05`). While there are deep-learning approaches for the extraction, temporal tagging as a whole is usually solved with highly specific rule-based systems, such as SUTime (Chang and Manning, 2012) or HeidelTime (Strötgen and Gertz, 2013). However, transferring rule-based methods to new languages or text domains requires a large manual effort to create rules specific to the target language. Although work on the automatic generation of rules for many languages (Strötgen and Gertz, 2015) exists, the rule quality typically does not match the high accuracy of hand-crafted rules.

In contrast to rule-based systems, neural networks are known for their ability to generalize to new targets, in particular, for cross- and multilingual applications (Rahimi et al., 2019; Artetxe and Schwenk, 2019). In the context of temporal tagging, recent works have shown promising results of neural networks for TE extraction in monolingual (Laparra et al., 2018) and multilingual settings (Lange et al., 2020) where a single neural model is trained on many languages at once. However, TE normalization remains challenging, and no solution for the normalization across languages exists yet.

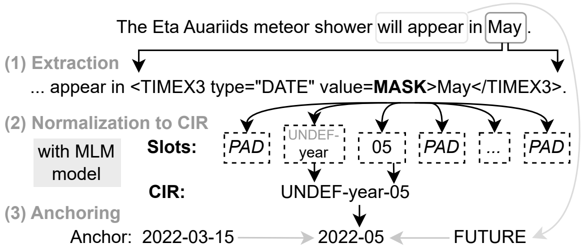


Figure 1: Overview of our 3-step pipeline for temporal tagging consisting of extraction of temporal expressions (step 1), normalization to a context-independent representation (CIR) using a slot-based masked language model (step 2) and anchoring given a reference time and further contextual cues (step 3).

We propose a new multilingual normalization method which can make use of labeled data from many languages by training a neural transformer model with a masked language modeling (MLM) objective. Thus, we adopt the MLM objective function for a new purpose: TE normalization.

To the best of our knowledge, this is the first work that uses neural networks for TE normalization. For this, as shown in Figure 1 and detailed below, we split the normalization task into two steps: normalization to a context-independent representation (CIR) and anchoring this representation using the document context.

The main contribution of this paper is our novel neural normalization method based on masked language modeling. For this, we create a large-scale multilingual dataset with weakly-supervised annotations of TEs and their normalized values in 87 languages. Our extensive set of experiments across 17 languages demonstrates that our multilingual method robustly works for many languages and outperforms the state of the art for multilingual temporal tagging, HeidelTime (Strötgen and Gertz, 2015), especially for low-resource languages by

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1174–1186
May 2-6, 2023 ©2023 Association for Computational Linguistics

more than 33 F_1 on average. Further, we explore different training and decoding strategies for our model. The code for our models and the weakly-supervised data is publicly available.[1]

2 Related Work

TE Normalization. Besides rule-based systems (Chang and Manning, 2012; Strötgen and Gertz, 2013), one normalization method for TEs are context-free grammars (Bethard, 2013; Lee et al., 2014) which are independent of the extraction method. However, they are even more language-specific than rule-based systems and hardly generalizable to new languages. Laparra et al. (2018) used a rule-based procedure for English TE normalization based on the SCATE format proposed by Bethard and Parker (2016). While their method could be extended to multilingual applications, no annotated data for other languages is available in the SCATE format, and it is mostly incompatible with the predominant TimeML (Pustejovsky et al., 2005) annotation format. Therefore, we will focus on the TimeML format in this work and present the first neural approach to TE normalization.

Masked Language Modeling (MLM). The MLM paradigm gains a lot of attention (Sun et al., 2021) due to popular language models like BERT (Devlin et al., 2019) This leads to active research on using MLM to solve further tasks like text classification (Brown et al., 2020), named entity recognition (Ma et al., 2021) and relation extraction (Han et al., 2021), also in low-resource languages (Hedderich et al., 2021). In this work, we adopt it to TE normalization for the first time.

3 Background on Temporal Tagging

Temporal tagging addresses the detection, classification and normalization of temporal expressions in unstructured texts — often following the TimeML specifications (Pustejovsky et al., 2005).

TimeML's most important attributes are `type` (the class of an expression, e.g., DATE, TIME, DURATION or SET), and `value` (the normalized meaning of an expression, e.g., `YYYY-MM-DD` for specific days, such as `2022-05-01` for May 1, 2022). While some TEs contain all necessary information for the normalization, e.g., "May 1, 2022", many expressions are incomplete w.r.t. the temporal information required for a normalization. An example is a *relative expression* like "yesterday" which needs an anchor point. Given the anchor point May 1, 2022, for example from the document creation time, "yesterday" should be annotated with `type=`DATE and `value=2022-04-30`.

Determining the anchor point can be challenging as it requires additional context information that could be given anywhere in the document. Therefore, systems for TE normalization, such as HeidelTime (Strötgen and Gertz, 2013), create an intermediate context-independent representation (CIR) of the `value`. In the syntax of HeidelTime, the expression *yesterday* would result in a CIR of `UNDEF-last-day`. Similarly, an *underspecified expression*, such as "May" would be represented with a CIR of `UNDEF-year-05`. Note that such a syntax for CIRs is language-independent. See Appendix A for more details. To determine the final `value`, the CIR needs to be anchored given, e.g., a reference date and further cues (such as tense information).

4 Approach

We propose to approach multilingual temporal tagging in three steps as shown in Figure 1: (1) Extraction of temporal expressions and their types using a multilingual sequence tagger; (2) Normalization of TEs to CIRs with our novel MLM-based normalization model; (3) Anchoring of CIRs given a reference time, e.g., using HeidelTime rules.

Our main contribution is a neural model for the second subtask, the normalization to a CIR. To the best of our knowledge, this has not been addressed with neural networks before. In this section, we detail all components of our approach. Information on the models that we apply for the first and third subtasks as well as an ablation study of directly predicting the normalized anchored value (without CIRs) are given in Section 5.

Masked Language Modeling. We model the task of assigning CIRs to temporal expressions as masked language modeling. In particular, we add TimeML annotations as inline information to the text sequences and mask the value field for prediction, e.g., "... <TIMEX3 type="DATE" value="**MASK**">yesterday</TIMEX3> ...". Note that those annotations could be the ground-truth annotations when applying the model on gold temporal expressions or predicted temporal expressions when using the model in the 3-step pipeline as described above. In our experiments, we train a transformer model for CIR prediction using the

---

[1] `https://github.com/boschresearch/temporal-tagging-eacl`

masked language modeling (MLM) objective.

Slot-Based Value Representation. Using only a single mask token for the whole CIR would require the model to store all possible CIRs in its vocabulary. Since it is not possible to enumerate, i.a., all possible dates, we model the CIRs as a fixed-length sequence of slots. In particular, we define 11 slots and use regular expressions to split the value field into slots in the training data. Figure 1 shows an example for the CIR "UNDEF-year-05" that is represented as the slots " [PAD], [year], [05], [PAD], ..., [PAD]". Details on the slots and regular expressions are given in Appendix A. To cover the full vocabulary of CIRs, we introduce 200 new tokens to the vocabulary of the language model.[2]

Curriculum Learning. Our slot-based representation with 11 slots per CIR results in 11 masks. To train the model on this task, we apply curriculum learning in the first half of the training. In particular, we start with masking only a single slot of the CIR and steadily increase the number of masks up to the maximum of 11. For the second half of the training, masking is applied to all slots. We follow Devlin et al. (2019) and mask different parts of the input with different probabilities. In particular, we mask the value slots with a probability of 70%, annotated tokens with 15%, types with 10% and other text parts with 5%.

Inference and Decoding. For inference, we first add 11 masks (i.e., one per slot) to the input sentence. They serve as value placeholders that need to be predicted. Then, we use the masked language model to predict the most probable sequence of slots for the CIR. To decode the sequence, we apply sequential left-to-right decoding of all masks by iteratively decoding the left-most mask and replacing the mask with its predicted value until all masks are resolved. We compare this to two alternative decoding strategies: (i) decoding all masks simultaneously, (ii) training a conditional random field model that takes the logits as input and uses the Viterbi algorithm to determine the most probable sequence of predictions (Lafferty et al., 2001).

5 Experiments

This section describes our experiments and discusses the results. We compare our model to HeidelTime (Strötgen and Gertz, 2013), the current state of the art for multilingual temporal tagging. For evaluation, we use the TempEval3 evaluation script (UzZaman et al., 2013) and report strict, relaxed and type F_1 for the extraction and value F_1 for our normalization experiments, respectively.

Evaluation Data. Our models are evaluated on gold-standard corpora in 17 languages. Details on the corpora are given in Appendix B.2. We divide the languages into high- and low-resource depending on whether manually created HeidelTime rules are available for the respective language.

Training Data. For training the normalization model, we create a large-scale weakly-supervised dataset covering 87 languages.[3] Reasons are that (i) existing gold training data is too small to cover the wide range of different values and (ii) CIRs are not part of existing annotations. For all languages, we take the data from GlobalVoices[4] (news-style documents) and Wikipedia[5] (narrative-style documents), use spacy for tokenization and HeidelTime for the annotation with temporal expressions. The number and quality of annotations is highly dependent on the amount of available data for that language and the quality of HeidelTime's rules. Details on the weakly-supervised data are given in Appendix B.1.

3-Step Pipeline for Temporal Tagging. Both our temporal expression extraction and normalization models are based on the mulitlingual XLM-R transformer (Conneau et al., 2020).[6]

We model the *TE extraction* as a sequence-labeling problem following Lange et al. (2020). For this, we convert the annotated corpora into the BIO format. For the monolingual setting (Mono), we train one model per language on the gold-standard resources if available or the weakly-supervised data otherwise. For the multilingual setting (Multi), we train a single model on the combined training resources of all languages.

For the *normalization to CIRs*, we train our proposed model with masked language modeling (see Section 4). In our experiments, we evaluate this model in combination with the multilingual extraction model (Multi+OUR) as well as in combination with the gold boundaries for temporal expressions (Gold+OUR) which serves as an upper bound.

---

[2] In our experiments, we compare our pre-defined slots to using subtokens from the language model tokenizer.

[3] The set of 87 languages is the intersection of languages covered by HeidelTime, our data and the XLM-R language model that we use for initializing our models.

[4] `https://globalvoices.org/`

[5] `https://wikipedia.org/`

[6] `xlm-roberta-base` with 270M parameters.

	HeidelTime				**Mono+OUR**				**Multi+OUR**				**Gold+OUR**
	Str.	**Rel.**	**Type**	**Val.**	**Str.**	**Rel.**	**Type**	**Val.**	**Str.**	**Rel.**	**Type**	**Val.**	**Val.**
de (N)	69.7	79.3	75.4	**62.4**	**75.4**	**85.9**	**80.6**	61.5	70.9	82.6	76.2	59.5	74.1
de (W)	88.5	94.3	89.0	84.8	**89.6**	**97.0**	**96.0**	83.8	88.9	96.7	95.4	**85.7**	87.5
en (N)	81.8	90.7	83.3	**78.1**	**85.7**	**92.3**	**86.5**	72.5	82.0	88.9	82.8	70.5	79.0
en (W)	90.6	94.3	90.6	**94.3**	93.1	96.6	**93.1**	89.7	**94.7**	**98.3**	87.7	94.2	94.2
es (N)	83.7	90.2	86.1	**80.9**	**89.6**	**94.5**	**91.4**	79.0	89.3	94.2	90.0	77.1	84.4
et (N)	42.4	57.4	51.3	44.0	3.3	28.0	24.4	9.6	**55.5**	**78.0**	**72.0**	**45.2**	64.8
fr (N)	**85.6**	**90.6**	**82.3**	**73.3**	82.5	88.1	79.7	67.9	82.4	89.8	76.9	61.4	68.0
hr (W)	**93.3**	**95.8**	**94.6**	**85.7**	84.1	90.8	89.5	74.6	86.3	91.7	90.1	75.7	84.7
it (N)	**84.4**	**92.9**	**83.5**	**74.1**	69.8	81.4	73.7	60.4	76.8	82.4	78.4	67.2	75.3
nl (N)	54.0	**91.3**	79.0	44.4	61.4	73.0	67.2	42.7	**76.0**	82.7	**81.4**	**53.5**	64.6
pt (N)	71.3	80.9	76.5	63.2	**87.1**	**91.2**	85.0	**68.7**	**87.1**	91.1	**86.5**	**68.7**	76.6
vi (W)	**92.6**	89.5	**96.6**	**91.6**	87.6	85.0	89.8	83.5	91.5	**93.8**	92.6	90.8	91.2
avg.	**78.2**	87.3	82.4	**75.6**	75.8	83.7	79.7	66.2	81.8	**89.2**	**84.2**	70.7	78.7
	HeidelTime-auto				**Mono+OUR**				**Multi+OUR**				**Gold+OUR**
	Str.	**Rel.**	**Type**	**Val.**	**Str.**	**Rel.**	**Type**	**Val.**	**Str.**	**Rel.**	**Type**	**Val.**	**Val.**
ca (N)	28.1	62.8	61.1	**43.6**	29.5	64.3	62.3	40.2	**77.3**	**87.8**	**82.5**	**59.7**	67.9
el (W)	2.2	4.9	4.9	1.3	47.0	88.2	86.1	64.6	**81.7**	**92.0**	**90.2**	**70.6**	83.7
eu (N)	22.5	26.8	23.9	18.3	0.0	0.0	0.0	0.0	**59.7**	**70.2**	**66.0**	**45.0**	51.2
id (N)	19.7	54.7	44.5	40.1	17.4	39.7	30.6	25.6	**49.7**	**79.5**	**63.9**	**46.9**	64.8
pl (N)	18.8	27.2	16.5	11.2	86.1	**92.5**	87.6	58.7	**86.7**	92.2	**87.7**	**59.0**	66.0
ro (N)	3.2	19.5	16.7	5.5	3.8	22.6	37.0	7.7	**9.8**	**47.2**	**39.1**	**19.7**	54.6
ua (W)	1.6	2.8	2.2	1.2	**80.2**	90.6	87.5	63.6	79.4	**90.7**	**88.8**	**65.4**	74.5
avg.	12.7	28.4	24.3	17.3	37.7	56.8	55.9	37.2	**63.5**	**79.9**	**74.0**	**50.9**	66.1

Table 1: Detailed overview of our results for extraction (Str., Rel., Typ.) and normalization (Val.) per language for **N**ews and **W**iki domains. The upper and lower parts display high- and low-resource languages, respectively.

For *anchoring CIRs*, we use rules similar to HeidelTime's rules.[7] In particular, anchor dates can be given by the document creation time or by previous temporal expressions (Strötgen and Gertz, 2016).

Results. Table 1 gives an overview of our experimental results. Multilingual extraction outperforms monolingual extraction, probably because the model is able to use knowledge from different languages. Our multilingual model achieves +2 F_1 for high-resource and +51 F_1 for low-resource languages compared to HeidelTime.

The normalization results are given in the *Val.* columns of Table 1. Our masked language model is matching HeidelTime's performance rather close for high-resource languages and outperforms it for low-resource languages with an increase of 33 F_1 points on average with our multilingual extraction model. Note that our models are multilingual, thus, we can use the same model for all languages.[8] The upper bound of using gold extractions (Gold+OUR) shows that the extraction part still offers room for future improvements.

Note that HeidelTime with automatically created rules has a poor performance for some low-resource languages (el, ro, ua). This is similar to the observations by Grabar and Hamon (2019) who found that "[e]xploitation of this automatically built system produced no results when applied to the Ukrainian data." For those languages, the automatic rule generation is not good enough in practice which emphasizes the need for multilingual systems like our model.

Ablation Studies. As our proposed model consists of multiple components, we now investigate their individual effects in more detail. The results for our ablation studies are given in Table 2.

First, we test different *decoding strategies* as described in Section 4. We find that sequential decoding works best. However, it also requires more computation time. A cheaper alternative with only minor performance decreases is the simultaneous decoding of all masks.

Second, we analyze the impact of different *value representations* by comparing our proposed approach with CIR and slot tokenization to (i) tokenization of values using the standard XLM-R

---

[7] More precisely, we use a slightly modified version of HeidelTime's SPECIFYAMBIGUOUSVALUESSTRING function which incorporates tense information of the context using morphological features from spacy (`https://spacy.io/usage/linguistic-features#morphology`).

[8] Since we actually train the MLM model on 87 languages, we could even apply it to more languages if there were gold-standard evaluation datasets publicly available.

tokenizer instead of pre-defined slots (w/o OUR Slots), and (ii) training a model to directly predict the anchored value without CIRs in between (w/o OUR CIR). For (i), we find that our slot method has major advantages when processing narrative texts, such as Wikipedia, due to the higher amount of relative expressions (cf., Table 3 in Appendix B.3), that are tokenized into many subtokens (up to 34, instead of 11 when using our slots). For (ii), we add the document creation time to the input so that the model has more temporal information to predict the fully normalized value directly instead of a CIR. However, we find that current transformers are not able to correctly incorporate this information in a combined normalizing+anchoring step and mostly predict a memorized, incorrect value. Thus, using CIRs as an intermediate step is important for neural temporal tagging.

Third, we investigate the *training strategy* and *training data*. Our curriculum learning has advantages for low-resource languages as it reduces the training complexity which helps for the difficult adaptation to languages with few resources. Weakly-supervised training data is required, as the amount of gold-standard data is too small to train the MLM model. Finetuning the trained MLM model further on gold data (Weak+Gold) decreases performance slightly. Training the model on monolingual data only also decreases performance, highlighting the prospects of our multilingual approach.

Finally, we compare our models to an *encoder-decoder model*, i.e., an autoregressive language model that we adapt to TE normalization. For this, we follow the entity linking approach from De Cao et al. (2021) and train a BART encoder-decoder model (Lewis et al., 2020) for constrained decoding against a subspace of normalized TEs with our weakly-supervised data. We add the document creation time to the input, mark the extracted annotations and keep other TEs in the context, as in our other experiments. Given the gigantic amount of possible temporal expressions, e.g., there are roughly 32M seconds in a single year, we have to prune the search space to a reasonable size. Thus, we do not use time expressions of hour and smaller granularities and restrict the search space to years and months from 1 AD to 2100. Finer elements like weeks, days and daytimes are added for years between 2000 and 2026. We use durations for all defined units with numbers up to 10,000, e.g., 10,000 days. With this, we prune the search space to 1.4B terms which we store in a prefix tree. This results in an acceptable inference speed with BEAM search (5 beams). It takes roughly twice as long as our sequential MLM decoding. Note that this BART model has more parameters (400M) than the base version of XLM-R (270M) that we use in our model. The results are shown in the lower part of Table 2. We see, that our proposed MLM normalization model outperforms the BART model by a large margin. Nonetheless, the encoder-decoder model performs comparable to our model variant that directly predicts fully-normalized expressions. This clearly highlights the need for normalizing to CIRs before anchoring temporal expressions.

	News		**Wiki**		**Low-R.**	
	de	**en**	**de**	**en**	**ca**	**eu**
OUR	**74.1**	79.0	**87.5**	94.2	67.9	**51.2**
Decoding Strategy (OUR uses Sequential)						
w/ Simultaneous	73.3	78.3	**87.5**	93.5	**68.1**	50.4
w/ Viterbi	73.3	78.3	**87.5**	94.2	67.9	50.4
Value Representation						
w/o OUR Slots	71.9	77.9	83.3	92.8	63.7	27.8
w/o OUR CIR	68.5	68.0	66.5	55.7	41.6	21.2
Training Strategy						
w/o Curriculum	72.1	**80.4**	85.5	**94.4**	64.7	29.3
Training Data (OUR uses Weak)						
Weak + Gold	68.3	76.1	58.2	93.3	-	-
only Gold	14.2	13.8	6.7	3.6	-	-
only Monolingual	63.1	76.8	86.8	91.4	29.2	8.9
Encoder-Decoder Model						
Monolingual	59.3	67.4	49.4	59.6	2.2	7.3
Multilingual	63.5	63.0	49.0	58.5	54.8	25.2

Table 2: Ablation study for our model components (Value F_1) on gold extractions. Low-R. stands for low-resource languages without gold training data.

6 Conclusion

In this paper, we introduced a new method for normalizing temporal expressions based on masked language modeling and a new slot-based prediction scheme of context-independent representations. With this approach, we were able to train a single multilingual model for the task. We evaluated our method in 17 languages and set the new state of the art in low-resource languages with massive improvements of 35 F_1 points on average. The success of our method demonstrates the potential of neural networks for temporal normalization and we are convinced that it will enable future research on this topic. An interesting research direction is the joint modeling of extraction and normalization.

Limitations

Our experiments are focused on Indo-European languages due to the lack of publicly available, labeled data points in other languages. Exceptions for which we could test zero-shot transfer were Basque, Estonian, Indonesian and Vietnamese. Even though, our model is working for these languages, it is not clear if the multilingual models transfer to all languages seen in the pre-training or by our weak supervision. The training of the multilingual models requires a considerable number of computational resources (up to 1.5 GPU days), which might not be available for all people/organizations. By publishing our model, we hope to lower the barrier for this kind of research by providing a pre-trained starting point. An in-depth error analysis to better understand which types of temporal expressions are well or less well covered in which language by our model was not performed. We are full of hope that such analyses can be tackled by users of our models who have the required language skills so that the analysis does not have to be limited to English.

Acknowledgments

We would like to thank the members of the BCAI NLP & NS-AI research group and the anonymous reviewers for their helpful comments.

References

Begoña Altuna, María Jesús Aranzabe, and Arantza Díaz de Ilarraza. 2020. Eustimeml: A mark-up language for temporal information in basque. *Research in Corpus Linguistics*, 8(1):86–104.

Mikel Artetxe and Holger Schwenk. 2019. Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Steven Bethard. 2013. A synchronous context free grammar for time normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 821–826, Seattle, Washington, USA. Association for Computational Linguistics.

Steven Bethard and Jonathan Parker. 2016. A semantically compositional annotation scheme for time normalization. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3779–3786, Portorož, Slovenia. European Language Resources Association (ELRA).

André Bittar, Pascal Amsili, Pascal Denis, and Laurence Danlos. 2011. French TimeBank: An ISO-TimeML annotated reference corpus. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 130–134, Portland, Oregon, USA. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Angel X. Chang and Christopher Manning. 2012. SUTime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*.

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

Francisco Costa and António Branco. 2012. TimeBankPT: A TimeML annotated corpus of Portuguese. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3727–3734, Istanbul, Turkey. European Language Resources Association (ELRA).

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

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

Corina Forăscu and Dan Tufiş. 2012. Romanian TimeBank: An annotated parallel corpus for temporal information. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3762–3766, Istanbul, Turkey. European Language Resources Association (ELRA).

Natalia Grabar and Thierry Hamon. 2019. Wikiwars-ua: Ukrainian corpus annotated with temporal expressions. *Computational Linguistics and Intelligent Systems*, 2:22–31.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Emmanouil I. Kapernaros. 2020. Extending the temporal tagger heideltime for the greek language.

Jan Kocon, Marcin Oleksy, Tomasz Bernas, and Michał Marcinczuk. 2019. Results of the poleval 2019 shared task 1: Recognition and normalization of temporal expressions. *Proceedings ofthePolEval2019Workshop*, page 9.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lukas Lange, Anastasiia Iurshina, Heike Adel, and Jannik Strötgen. 2020. Adversarial alignment of multilingual models for extracting temporal expressions from text. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 103–109, Online. Association for Computational Linguistics.

Egoitz Laparra, Dongfang Xu, and Steven Bethard. 2018. From characters to time intervals: New paradigms for evaluation and neural parsing of time normalizations. *Transactions of the Association for Computational Linguistics*, 6.

Kenton Lee, Yoav Artzi, Jesse Dodge, and Luke Zettlemoyer. 2014. Context-dependent semantic parsing for time expressions. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Pawel Mazur and Robert Dale. 2010. WikiWars: A new corpus for research on temporal expressions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 913–922, Cambridge, MA. Association for Computational Linguistics.

Anne-Lyse Minard, Manuela Speranza, Ruben Urizar, Begoña Altuna, Marieke van Erp, Anneleen Schoen, and Chantal van Son. 2016. MEANTIME, the NewsReader multilingual event and time corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4417–4422, Portorož, Slovenia. European Language Resources Association (ELRA).

Paramita Mirza. 2016. Recognizing and normalizing temporal expressions in indonesian texts. In *Computational Linguistics*, pages 135–147, Singapore. Springer Singapore.

Siim Orasmaa. 2014. Towards an integration of syntactic and temporal annotations in Estonian. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1259–1266, Reykjavik, Iceland. European Language Resources Association (ELRA).

James Pustejovsky, Robert Ingria, Roser Saurí, José Castaño, Jessica Littman, Rob Gaizauskas, Andrea Setzer, Graham Katz, and Inderjeet Mani. 2005. The specification language TimeML. In *The language of time: a reader*, pages 545–557. Oxford University Press.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Roser Saurı. 2010. Annotating temporal relations in catalan and spanish timeml annotation guidelines. Technical report, Technical report, Technical Report BM 2010-04, Barcelona Media.

Luka Skukan, Goran Glavaš, and Jan Šnajder. 2014. Heideltime. hr: extracting and normalizing temporal expressions in croatian. In *Proceedings of the 9th Slovenian Language Technologies Conferences (IS-LT 2014)*, pages 99–103.

Jannik Strötgen, Ayser Armiti, Tran Van Canh, Julian Zell, and Michael Gertz. 2014. Time for more languages: Temporal tagging of arabic, italian, spanish, and vietnamese. *ACM Transactions on Asian Language Information Processing*, 13(1).

Jannik Strötgen and Michael Gertz. 2011. Wikiwarsde: A german corpus of narratives annotated with temporal expressions. In *Proceedings of the conference of the German society for computational linguistics and language technology (GSCL 2011)*, pages 129–134. Citeseer.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and cross-domain temporal tagging. *Language Resources and Evaluation*, 47(2).

Jannik Strötgen and Michael Gertz. 2015. A baseline temporal tagger for all languages. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Jannik Strötgen and Michael Gertz. 2016. *Domain-sensitive temporal tagging*, volume 9. Morgan & Claypool Publishers.

Jannik Strötgen, Anne-Lyse Minard, Lukas Lange, Manuela Speranza, and Bernardo Magnini. 2018. KRAUTS: A German temporally annotated news corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Tianxiang Sun, Xiangyang Liu, Xipeng Qiu, and Xuanjing Huang. 2021. Paradigm shift in natural language processing. *arXiv preprint arXiv:2109.12575*.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. SemEval-2013 task 1: TempEval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia, USA. Association for Computational Linguistics.

A Slot Tokenization of CIRs

In this section, we describe our slot-based tokenization of the context-independent representation (CIR) of values as introduced in Section 3 and Section 4 of the main paper.

A.1 Overview of Slots

We use the following 11 slots to represent CIRs values.[9] These slots are then used for masking during training and inference with our normalization model (which basically is a masked language model).

SB: This slot can contain BC information of years (e.g., as in `BC4000` for the year 4000 BC) or the duration markers P and PT. Moreover, mathematical operations like `PLUS` are covered as used in relative expression involving offset computations (e.g., `this-day-plus-2` for the day after tomorrow) and holiday names (`EasterSunday`).

SD1, SD2: These slots are used to represent 4-digit year numbers (**SD1** = `20` and **SD2** = `22` for the year `2022`) by splitting the 4-digit number into two 2-digit numbers. This helps to generalize to unseen years as fewer parameters have to be learned. In addition, we use **SD1** to mark reference expressions like `PAST_REF`. For underspecifed expressions like `UNDEF-this-day`, `this` is stored in **SD1** and day in **SD2**. Moreover, **SD1** and **SD2** are used to store numbers of `DURATION` expressions.

SD3, SD4: Analogously to **SD1** and **SD2** that are used to store year information, **SD3** is used for months and **SD4** for days.

ST1, ST2, ST3: Temporal information from expressions of type `TIME` that are smaller than day granularity are stored in the **ST** slots. For example, the hour information of `24:00` and the daytime information, such as `EV` is stored in **ST1**. Information on minutes and seconds is stored in **ST2** and **ST3**, respectively. Moreover, these slots are used to cover additional units in durations, such as in `P1D2H` (1 day and 2 hours).

SA1, SA2, SA3: Finally, some CIRs include function calls which can be augmented with arguments that we store in the **SA** slots. For example, the argument `2` of `this-day-plus-2` is stored in **SA1**. Other function calls are used to compute days with respect to holidays like `EaserSunday` or specific weekdays.

Note that slots can be optional depending on the temporal expression. For example, the value `2022` representing the year 2022 would only require **SD1** and **SD2**. All other slots are set to a padding value `[PAD]` then which allows a fixed-sized representations of CIRs that can be predicted with our masked language model.

Examples. The following examples show temporal expressions, their corresponding CIRs and the tokenization into our slots. Note that there is no need to capture terms like `UNDEF` in our slots as the presence of words like `this`, `next` or `last` in a CIR implies the existence of `UNDEF` in the CIR.

---

[9] Note that our CIRs describe a superset of TimeML.

This information can be reconstructed when obtaining a CIR from our slots. This also includes `--` to separate numbers as in `YYYY-MM-DD` values, `REF` in reference expressions and `T` for time information. We use the following format to give examples for our CIR conversion: *Text → CIR → Slot Sequence*

- *Now* ...
  → `PRESENT_REF`
  → **SD1**=`PRESENT`

- ... for *1000 days* ...
  → `P1000D`
  → **SB**=`P`, **SD1**=`10`, **SD2**=`00`, **SD4**=`D`

- ... for *one and a half day* ...
  → `P1D12H`
  → **SB**=`P`, **SD1**=`1`, **SD4**=`D`, **ST1**=`12`, **ST2**=`H`

- ... in *1000 BC* ...
  → `BC1000`
  → **SB**=`BC`, **SD1**=`10`, **SD2**=`00`

- ... on the *morning of March 15, 2022* ...
  → `2022-03-15TMO`
  → **SD1**=`20`, **SD2**=`22`, **SD3**=`03`, **SD4**=`15`, **ST1**=`MO`

- On *March 15*, ...
  → `UNDEF-year-03-15`
  → **SD1**=`year`, **SD3**=`03`, **SD4**=`15`

- ... *the day after tomorrow* ...
  → `UNDEF-this-day-PLUS-2`
  → **SB**=`PLUS`, **SD1**=`this`, **SD2**=`day`, **SA1**=`2`

- ... at *Pentecost*[10] ...
  → `UNDEF-year-00-00 funcDate...`
  `...Calc(EasterSunday(YEAR, 49))`
  → **SB**=`EasterSunday`, **SD1**=`year`, **SD2**=`00`, **SA1**=`49`

A.2 Regular expressions

In the following, we will describe the six regular expressions used to split CIR values from HeidelTime outputs into our slots for the weakly-supervised training data.

Notation. For readability, we define the following groups to capture temporal units and other fixed names. Note that these are used across languages. For example, the German expression *Montag* would still be represented with `monday`.

```
UNITS = (H|D|DE|DT|M|C|Y|
    C|CE|W|WE|Qu|Q|S)
UNITS_F = (day|month|year|
    decade|century|week|
    weekend|quarter|
    hour|minute|second)
DAYTIME = (NI|AF|MO|EV|MD|MI)
SPECIAL = (SP|SU|FA|AU|WI|
    H1|H2|Q1|Q2|Q3|Q4|H|Q)
NAMES = (monday|tuesday|
    wednesday|thursday|
    friday|saturday|sunday|
    january|february|march|
    april|may|june|july|
    august|september|
    october|november|december)
```

In the following, $DX(n)$ marks the n-th group captured by the regular expression DX.

$D1$: References. The first regular expression $D1$ is used to capture simple reference expressions that refer to uncertain points in time.

```
D1 = (PRESENT|PAST|FUTURE)_REF
Slots: SD1=D1(1)
```

$D2$: Explicit Dates. The second regular expression $D2$ detects explicit values that do not need further normalization, such as days in the `YYYY-DD-MM` format, e.g., `20222-03-15`.

```
D2 =(BC)?(\d\d?|XX)?(\d\d|XX)?
  (?:-(W)?(\d\d?|XX|SPECIAL))?
  (?:-(\d\d?|XX|WE))?\)?
  (?:T(\d\d|X|DAYTIME|XX)?
  (?::(\d\d))?
  (?:(?::|-)(\d\d))?)?
Slots: SB=D2(1), SD1=D2(2),
  SD2=D2(3), SD3=D2(5), SD4=D2(6),
  ST1=D2(7), ST2=D2(8),
  ST3=D2(9)|D2(4)
```

$P1$: Durations. The third regular expression $P1$ detects expressions of `type` DURATION, e.g., `P1D2H`. These are defined as `P<number><unit>` for units of at least day granularity and `PT<number><unit>` for smaller granularities. We capture up to two different units `P1D2H` (1 day and 2 hours) but ignore further units that are theoretically defined in the TimeML specifications but do not occur often in practice (in our datasets those did not occur at all).

```
P1 = (P|PT)(\d\d?|X|XX)
  (\d\d|\.)?(\d\d?)?)?(UNITS)?
  (\d\d?)?(UNITS)?
Slots: SB=P1(1), SD1=P1(2),
  SD2=P1(3), SD3=P1(4), SD4=P1(6),
  ST1=P1(5), ST2=P1(7)
```

---

[10] In christian communities, the holiday of Pentecost is celebrated 49 days after Easer Sunday.

$D3$: **Relative Dates.** While the previous regular expressions $D1$, $D2$ and $P1$ follow the TimeML specifications and capture fully normalized expressions, i.e., anchored values, the following regular expressions capture CIRs as used internally by HeidelTime. They represent relative expressions that need to be anchored.

$D3$ detects relative expressions with respect to a certain point in time, such as `this-day-plus-2` (the day after tomorrow).

```
D3 = UNDEF-(this|next|last|REF|
      REFUNIT|REFDATE)?-?
  (UNITS_F|SPECIAL)?-??
  (NAMES|SPECIAL)|XX|\d\d?)?
  (?:-?(\d\d?|XX))?
  (?:-(PLUS|MINUS|LESS)-(\d\d?)-?
  (\d\d?)?-?(\d\d?)?)?\)?
  (?:T(\d\d?|X|DAYTIME|XX)?
  (?::(\d\d?|XX))?(?:(?::|-)
      (\d\d|XX))?)
Slots: SB=D3(5), SD1=D3(1),
  SD2=D3(2), SD3=D3(3), SD4=D3(4),
  ST1=D3(9), ST2=D3(10), ST3=D3(11),
  SA1=D3(6), SA2=D3(7), SA3=D3(8)
```

$D4$: **Relative Dates (coarse).** $D4$ captures underspecified expressions like *May* that is missing year information and would be represented with the CIR `UNDEF-year-05`.

```
D4 = UNDEF-(year|decade|century?)
  -?(\d\d?|X)?-?(\d\d?|X)?-?
  (\d\d?|X|SPECIAL)?\)?
  (?:T(\d\d?|X|DAYTIME)?
  (?::(\d\d?|XX))?
  (?:(?::|-)(\d\d|XX))?)?
Slots: SD1=D4(1), SD2=D4(2),
  SD3=D4(3), SD4=D4(4),
  ST1=D4(5), ST2=D4(6), ST3=D4(7)
```

$D5$: **Holidays and functions.** Finally, $D5$ covers special functions used by HeidelTime. These functions are used to compute days with respect to weekdays and moveable feasts like `EasterSunday` that refer to different days depending on the year. For example, the earliest possible date of Easter Sunday is March 22 and the latest is April 25 in the Gregorian calendar.[11] The concrete date is then computed by an external function given a year.[12]

```
D5 = (UNDEF-year|UNDEF-this-year|
        UNDEF-century\d\d|\d\d\d\d)-
    (\d\d)-00 funcDateCalc\((
        WeekdayRelativeTo|
        EasterSundayOrthodox|
        EasterSunday|
        ShroveTideOrthodox)
    \(YEAR(?:(?:-(\d\d)))?
    (?:-(\d\d))
    (?:,\s?(-?\d\d?))?
    (?:,\s?(-?\d\d?))?
    (?:, (true|false))?\)\)
Slots: SB=D5(3), SD1=D4(1),
    SD2=D4(2), SD3=D4(5),
    SD4=D4(7), ST1=D4(3),
    SA1=D5(6), SA2=D5(7), SA3=D5(8)
```

B Data Statistics

B.1 Weakly-Supervised Data

As detailed in Section 5, we create weakly-supervised data to train our normalization model, as the gold standard is too small and is not annotated with CIRs which are required by our method. For all languages, we take the data from GlobalVoices[13] (news-style documents) and Wikipedia[14] (narrative-style documents), use spacy for tokenization and our HeidelTime version that outputs CIRs for the annotation with temporal expressions. The sizes of our weakly-supervised data for each language are given in Table 4.

B.2 Gold-Standard Data

Detailed information on the datasets used in this paper (their languages, domains, sizes and references) are provided in Table 5. Note that all corpora come from the news domain except the WikiWars corpora that are based on Wikipedia articles.

B.3 Distribution of Explicit and Relative Values

The distribution of explicit and relative values has a large impact on the normalization performance of different models, as shown in our ablation study in Section 5. Exemplarily, we analyze their distribution in the German and English datasets for which we have data from two domains: News and Wikipedia. The results are given in Table 3. We see, that the Wikipedia corpora contain a much larger percentage of relative values as these articles often follow a narrative structure (cf., (Strötgen and Gertz, 2016).

C A Note on Adopting HeidelTime

In our experiments, we used a modified version of HeidelTime. First, we implemented a new UIMA collection reader based on spacy as an alternative to

---

[11] https://en.wikipedia.org/wiki/List_of_dates_for_Easter

[12] https://www.linuxtopia.org/online_books/programming_books/python_programming/python_ch38.html

[13] https://globalvoices.org/

[14] https://en.wikipedia.org/wiki/List_of_Wikipedias

	De	En
News	**67.1** / 32.9	**52.3** / 47.7
Wiki	47.6 / **52.4**	44.2 / **55.8**

Table 3: Distribution of explicit / relative values according to HeidelTime by domains (in %).

the TreeTagger that has a restrictive license. This results in a slightly different sentence segmentation and tokenization, and, thus, minor differences in performance. For example, the original HeidelTime achieves 63.47 F_1 on the Portuguese test data, while our spacy version achieves 63.24 F_1 as one additional false positive expression was annotated due to different sentence boundaries. Second, we adapted HeidelTime to output its internal CIRs for the TimeML values, such that we can create our weakly-supervised training data.

The rather low performance of our models and HeidelTime for the high-resource languages Estonian (et) and Dutch (nl) can be explained by poor data quality. An inter-annotator agreement of 44 F_1 was reported for the Estonian corpus (Orasmaa, 2014), which is close to our results. The Dutch data was translated from English and automatically annotated via cross-lingual projections (Minard et al., 2016), which may reduce the annotation quality. Note, that only the first five sentences for each document were annotated in the Meantime corpora (it and nl). We restricted our evaluation to these annotated parts accordingly.

Rank	Lang	#Ann.
1	de	870897
2	en	542087
3	fr	284871
4	ar	280446
5	es	250871
6	pt	215209
7	it	199236
8	nl	194944
9	ru	122884
10	zh	105421
11	hr	50233
12	ro	33545
13	vi	22048
14	af	21081
15	mk	19539
16	tr	19532
17	gl	17416
18	ca	16747
19	bn	16284
20	cy	14738
21	bg	14550
22	et	13948
23	sv	13705
24	id	13031
25	da	12919
26	fy	12852
27	pl	11283
28	fa	11041
29	eu	10992
30	ne	10750
31	ms	10017
32	mg	9271
33	kk	8080
34	hi	7762
35	eo	7353
36	ur	6228
37	hu	5871
38	sq	5760
39	sk	5172
40	sr	4276
41	ka	4247
42	el	4217
43	he	4057
44	sw	3979
45	ja	3696
46	br	3582
47	uz	3361
48	th	3162
49	cs	3096
50	ga	2799
51	mn	2778
52	gd	2772
53	lt	2734
54	mr	2623
55	la	1876
56	uk	1673
57	hy	1642
58	ta	1556
59	my	1103
60	ml	1079
61	kn	1029
62	fi	1017
63	oa	979
64	jv	968
65	ky	926
66	is	804
67	am	776
68	ku	557
69	so	506
70	yi	485
71	ko	483
72	si	442
73	ps	403
74	lo	354
75	km	350
76	su	335
77	lv	323
78	as	299
79	ug	283
80	sd	278
81	gu	258
82	ha	205
83	sl	125
84	yo	102
85	sa	24
86	or	19
87	xh	3

Table 4: Languages and the sizes of our weakly-supervised data.

Corpus	Language	#Annotations (train / test)	Reference
Corpora only used for evaluation			
KRAUTS-DieZeit	German (de)	_ / 493	(Strötgen et al., 2018)
TempEval-3 (platinum)	English (en)	_ / 137	(UzZaman et al., 2013)
KOMPAS (test)	Indonesian (id)	_ / 192	(Mirza, 2016)
TimeBankCA	Catalan (ca)	_ / 1383	(Saurı, 2010)
EstTimeML	Estonian (et)	_ / 622	(Orasmaa, 2014)
EusTimeML	Basque (eu)	_ / 112	(Altuna et al., 2020)
Fr TimeBank	French (fr)	_ / 423	(Bittar et al., 2011)
Ro TimeBank	Romanian (ro)	_ / 151	(Forăscu and Tufiş, 2012)
PT-TimeBank (test)	Portuguese (pt)	_ / 151	(Costa and Branco, 2012)
WikiWars-EL (test)	Greek (el)	_ / 414	(Kapernaros, 2020)
Corpora split into train and test sets			
Meantime (IT)	Italian (it)	229 / 244	(Minard et al., 2016)
Meantime (NL)	Dutch (nl)	221 / 259	(Minard et al., 2016)
TempEval-3 (ES)	Spanish (es)	730 / 551	(UzZaman et al., 2013)
PolEval-2019	Polish (pl)	633 / 6011	(Kocon et al., 2019)
WikiWars	English (en)	1378 / 1251	(Mazur and Dale, 2010)
WikiWars-DE	German (de)	1510 / 684	(Strötgen and Gertz, 2011)
WikiWars-HR	Croatian (hr)	724 / 677	(Skukan et al., 2014)
WikiWars-UA	Ukrainian (ua)	454 / 2237	(Grabar and Hamon, 2019)
WikiWars-VI	Vietnamese (vi)	118 / 101	(Strötgen et al., 2014)
Corpora only used for training			
KRAUTS-Dolomiten	German (de)	388 / _	(Strötgen et al., 2018)
Meantime (EN)	English (en)	472 / _	(Minard et al., 2016)
TempEval-3 (train, en)	English (en)	1240 / _	(UzZaman et al., 2013)
PT-TimeBank (train)	Portuguese (pt)	1127 / _	(Costa and Branco, 2012)
WikiWars-EL (train)	Greek (el)	1496 / _	(Kapernaros, 2020)

Table 5: Overview of datasets and details on their usage as training (extraction-only) or evaluation data.

FewShotTextGCN: K-hop neighborhood regularization for few-shot learning on graphs

Niels van der Heijden♣◇ **Ekaterina Shutova**♣ **Helen Yannakoudakis**♠
♣ILLC, University of Amsterdam, the Netherlands
♠Dept. of Informatics, King's College London, United Kingdom
◇Deloitte Risk Advisory, Amsterdam, the Netherlands
`n.vanderheijden@uva.nl, e.shutova@uva.nl,`
`helen.yannakoudakis@kcl.ac.uk`

Abstract

We present FewShotTextGCN, a novel method designed to effectively utilize the properties of word–document graphs for improved learning in low-resource settings. We introduce K-hop Neighborhood Regularization, a regularizer for heterogeneous graphs, and show that it stabilizes and improves learning when only a few training samples are available. We furthermore propose a simplification in the graph-construction method, which results in a graph that is ∼7 times less dense and yields better performance in low-resource settings while performing on-par with the state of the art in high-resource settings. Finally, we introduce a new variant of Adaptive Pseudo-Labeling tailored for word–document graphs. When using as little as 20 samples for training, we outperform a strong TextGCN baseline with 17% in absolute accuracy on average over eight languages. We demonstrate that our method can be applied to document classification without any language model pretraining on a wide range of typologically diverse languages while performing on par with large pretrained language models.

1 Introduction

Text classification, a key task in natural language processing (NLP), has many real-world applications, including toxic comment identification, news categorization, spam detection and opinion mining. One popular approach to this problem relies on large-scale pretraining of Transformer models (Devlin et al., 2018; Conneau et al., 2019; Raffel et al., 2020), which have shown to be able to approach or even surpass human performance on many natural language understanding (NLU) benchmarks (Rajpurkar et al., 2016; Wang et al., 2019; Liang et al., 2020). While these results are impressive for the languages on which models are pretrained, performance tends to deteriorate on languages where no or little data is available (Chau and Smith, 2021; van der Heijden et al., 2020). In practice, this means that these models are effective on a set of approximately 100 out of the 7000+ spoken languages in the world. Next to the requirement for vast amounts of data for pretraining, Transformer language models tend to be impractically large in terms of their number of parameters and have a high environmental footprint (Strubell et al., 2019).

Recently, Graph Neural Networks (GNNs) have shown to be effective for text classification in both transductive (Yao et al., 2019; Liu et al., 2020; Lin et al., 2021) and inductive (Nikolentzos et al., 2020; Ding et al., 2020) learning settings – with promising results in both high- and low-resource settings. Particularly in the transductive setting, the authors of TextGCN (Yao et al., 2019) show that Graph Convolutional Networks (GCNs) (Kipf and Welling, 2016) can outperform state-of-the-art methods for document classification on English datasets without any language model pretraining. They do so by modeling an entire corpus of documents simultaneously as one heterogeneous word–document graph. The document classification task is formulated as a node-classification task over this graph.

Later work shows that (multilingual) Pretrained Language Models (mPLMs) can be used to provide GNNs used in transductive setting with rich representations of both words and documents, improving results further in both monolingual (Lin et al., 2021) and cross-lingual settings (Wang et al., 2021; Li et al., 2020). These works focus solely on high-resource settings and do not report any results on performance in low-resource settings.

In this work, we propose a novel GNN-based method for learning document classification tasks on a range of languages without the need for any pretraining data (i.e., without utilizing any pretrained word embeddings or language models), and from few labeled samples only. To the best of our knowledge, we are the first to investigate few-shot graph-based transductive document classification

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1187–1200
May 2-6, 2023 ©2023 Association for Computational Linguistics

in a range of languages other than English.

We present FewShotTextGCN, an improved version of the original TextGCN model, where we exploit properties of the heterogeneous word–document graph for improved learning from scratch and with only a few labels. More specifically, we:

(1) Introduce K-hop Neighborhood Regularization (K-NR), an unsupervised learning technique for heterogeneous graphs, and use it in its $K = 2$ instantiation as a regularizer tailored to word–document graphs , and show that it consistently leads to performance gains in low-resource settings;

(2) Propose a simplification of the graph-construction method, which results in improved performance in the low-resource setting while reducing the graph density by a factor of approximately 7 on average, therefore substantially speeding up computations and reducing memory requirements;

(3) Present a variant of adaptive pseudo-labeling (Zhou et al., 2019) on word–document graphs and show that it leads to consistent gains over the original TextGCN approach (Yao et al., 2019), particularly when combined with K-NR.

We compare FewShotTextGCN to its predecessor and two strong PLMs on ten topic classification benchmarks comprising eight typologically diverse languages, and experiment with a range of low-resource settings, including using as little as 20 labeled samples to learn from, and without any other form of (pre-trained) knowledge about a language except for what constitutes a word (using word boundaries or a tokenizer). In our lowest-resource setting, our method outperforms TextGCN with 4.6% and 17% points in absolute accuracy on average for Reuters and MLDoc, respectively – while having a substantially smaller computational and memory footprint. FewShotTextGCN performs on par with large PLMs on the great majority of the considered benchmarks, without the need for any large-scale pretraining and at only a fraction of the parameter count of these PLMs – indicating that graph-based methods are an attractive alternative to using large PLMs for topic classification. All our code and models are released to facilitate further research on this topic.[1]

2 Graph Neural Networks

Graph Neural Networks (GNNs) are a class of neural models designed to facilitate representation learning on geometric data – data that naturally occur in many situations/fields, such as chemistry, social networks, maps, visual meshes, etc. Recently, there has been a great surge in research on GNNs. GNNs create new feature representations of nodes by aggregating the nodes' own feature representation and a message passed on from neighboring nodes. A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined as a set of nodes $\mathcal{V}$ with edges $\mathcal{E}$ between them, typically represented as a square adjacency matrix A, where each entry holds the weight of the edge between node j and i. Locality in graphs is defined by neighborhoods, where the neighbors of node i are defined as $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\ \forall (j,i) \in \mathcal{E}\}$. Let $\bigoplus$ be some permutation invariant aggregator such as sum, $average$ or max, and let ψ and ϕ be two differentiable, learnable functions such as an MLP. Using these ingredients, we can describe GNNs by the way they do message passing.

Convolutional GNNs use the weights c_{ij} of the edge between nodes j and i to weigh the incoming messages. These weights are part of the definition of the graph, meaning they are statically defined. The input feature x_i of node i is transformed to a latent representation h_i by taking

$$h_i = \phi(x_i, \bigoplus_{j \in \mathcal{N}_i} c_{ij}\psi(x_j)) \tag{1}$$

The first and most well-known convolutional GNN is the Graph Convolutional Network (GCN) (Kipf and Welling, 2016).

3 Related work

Our work is based on TextGCN (Yao et al., 2019), which also serves as our baseline for all experiments. To the best of our knowledge, we are the first to investigate few-shot graph-based transductive learning from scratch for document classification in a range of languages other than English. Since the scope of our work is few-shot document classification in many languages by learning from scratch, we do not consider CLHG (Wang et al., 2021) directly related work. The reasons being it models corpora in multiple languages jointly, whereas we learn each task in isolation, and relies on machine translation and mPLMs. Similarly, MGL (Li et al., 2020) relies on mPLMs for encoding similar corpora in different languages into one

---

[1] `https://github.com/mrvoh/FewShotTextGCN`

embedding space, where consecutively a graph is dynamically constructed based on the similarity of the documents in the respective embedding space. Finally, meta-learning is applied to learn to classify documents in one language, given a limited set of documents in at least three other languages. Hence, we do not review these works in-depth.

TextGCN TextGCN (Yao et al., 2019) is the first application of GNNs to transductive text classification, applied on English datasets without any language model pretraining. The great majority of experiments is performed in high-resource settings, but a small set of results on performance in low-resource settings is also provided – motivating us to further explore and expand upon this subject. The authors construct a heterogeneous graph containing both word and document nodes. Word–word edges are weighed based on the pointwise mutual information (PMI) between the respective words, and word–document edges are created based on the TF–IDF score of the word in the respective document. More specifically, the adjacency matrix A is defined as:

$$A_{ij} = \begin{cases} \text{PMI}(i,j) & i,j \text{ words, PMI}(i,j) > 0 \\ \text{TF–IDF}_{ij} & i \text{ document, } j \text{ word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Document–document links are not considered. A one-hot encoding is used as input features for the nodes and a two-layer GCN is used to classify the document nodes. While this setup is relatively simple in terms of preprocessing, pretraining and the number of parameters in the model, the authors show that their method performs on par with state-of-the-art methods, even improving the state of the art for the 20News[2] dataset.

BERTGCN Follow-up work on TextGCN is that of BERTGCN (Lin et al., 2021), where the authors leverage PLMs to initialize document–node features. More specifically, a BERT-based model is used to encode the documents, and all other nodes are initialized with a one-hot vector. The BERT model used for encoding documents is optimized both via gradients propagated through the GCN and via an auxiliary classifier that directly uses the BERT embeddings to classify the documents. Using BERTGCN, the authors improve over TextGCN on a variety of text classification tasks – especially on a sentiment analysis task, for which word order information is crucial for good performance (Johnson and Zhang, 2014). To be able to use BERTGCN in a full-batch gradient descent method, the authors use a memory bank that allows decoupling the dictionary size from the mini-batch size. Although the presented results are promising, a drawback of using large PLMs is the need for vast amounts of pretraining data, making these methods inaccessible for low-resource languages.

4 Data

In this section, we give an overview of the datasets we use and the respective classification tasks.

MLDoc Schwenk and Li (2018) published an improved version of the Reuters Corpus Volume 2 (Lewis et al., 2004) with balanced class priors for all languages. MLDoc consists of news stories in 8 languages: English, Spanish, French, German, Italian, Russian, Japanese and Chinese. Each news story is manually classified into one of four classes: *Corporate/Industrial (CCAT), Economics (ECAT), Government/Social (GCAT)* and *Markets (MCAT)*. Per language, the train and test datasets contain 1k and 4k samples respectively.

Reuters 21578 From the Reuters-21578 dataset, a dataset of English news articles on a wide variety of topics, we use the **R8** and **R52** subsets (all-terms versions[3]). R8 has 8 categories and consists of 5485 and 2189 samples for training and testing respectively. R52 has 52 categories and 6532 and 2568 samples for training and testing respectively. The distribution of samples over the respective categories is highly skewed.

During preprocessing on both datasets for all GNN-based models, we remove words with a frequency of less than 5, and tokenize the data. For all languages except Japanese and Chinese, we split sentences based on whitespace. For Chinese, we use the Jieba[4] tokenizer, and for Japanese, the Fugashi one (McCann, 2020). For Transformer-based models, solely their respective tokenizers are used.

5 Approach

Graph construction We follow the graph construction method as described in the original

---

[2] `http://qwone.com/~jason/20Newsgroups/`

[3] `https://ana.cachopo.org/datasets-for-single-label-text-categorization`

[4] `https://github.com/fxsjy/jieba`

TextGCN (Yao et al., 2019) work except we deviate in two different directions. Stopword removal is omitted, as this assumes knowledge of the language, whereas we aim for an approach that assumes no prior knowledge. Furthermore, word–word edges are omitted too. Omitting such edges results in a much less densely connected graph, making learning substantially less memory intensive. We argue that the added value of word–word edges in a word–document graph is minimal given 1) only global information of word co-occurrence is considered (i.e., co-occurrence over the whole corpus as opposed to within document co-occurrence), and 2) over the course of training, words co-occurring in a document can influence each other's representation through an N-layer GNN where $N > 1$. We illustrate this with an example in Appendix E, while in Section 8, we experimentally demonstrate the limited effect of word–word edges using ablation studies.

K-hop Neighborhood Regularization (K-NR) We propose a new method that can exploit the properties of a word–document graph, inspired by approaches such as GraphSage (Hamilton et al., 2017) that shows that meaningful node representations can be learned in an unsupervised manner with contrastive learning methods like Node2Vec (Grover and Leskovec, 2016). These methods typically consist of two components: a sampling technique for deciding what nodes are regarded as positive or negative samples, and a loss function. Let u be the anchor node, P_p the positive sample sampling method, P_n the negative sample sampling method, and $\mathcal{J}_{\mathcal{G}}(u, P_p, P_n)$ the contrastive loss function. In the case of GraphSage, P_p is defined as a random walk starting from the anchor node, and P_n is defined as uniformly sampling from all available nodes.

This contrastive learning approach on graphs assumes homogeneity and P_p always samples in the 1-hop neighborhood from the anchor node. Herein, we propose a contrastive learning regularization method tailored on heterogeneous graphs instead, where nodes of the same type are K-hops away from each other on the graph.

In what follows, we describe our approach in detail for heterogeneous word–documents graphs for the specific case of K-NR with $K = 2$. Driven by the intuition that documents (within a language) that share large parts of their vocabulary are more likely to be about the same topic, we introduce 2-hop Neighborhood Regularization (2-NR), a novel unsupervised learning method which can be used as a regularization technique.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the graph defined by the vertices $\mathcal{V}$ and edges $\mathcal{E}$. Let $\mathcal{V}_d, \mathcal{V}_w$ be the document and word nodes respectively. Given anchor node $u \in \mathcal{V}_d$, we first sample a word node $v \in \mathcal{V}_w$ connected to u by sampling from a multinomial distribution weighted by the edge attribute values (the TF–IDF scores):

$$v \sim Multinomial(1, A_{u,\{w|w\in\mathcal{V}_w \wedge w\in\mathcal{N}(v)\}}) \quad (3)$$

Then, a positive document node u_p and negative document node u_n are sampled as follows:

$$u_p \sim U(\mathcal{N}(v)) \quad (4)$$

$$u_n \sim U(\mathcal{V}_d \setminus \mathcal{N}(v)) \quad (5)$$

Let z_u be the final hidden representation of node u, the 2-NR loss, $\mathcal{L}_{2\text{-}NR}$, is then defined as:

$$\mathcal{L}_{2\text{-}NR}(u, u_p, u_n) = max\{d(u, u_p) - d(u, u_n) + m, 0\} \quad (6)$$

for some distance function d and margin m. This represents a triplet margin loss (Balntas et al., 2016), which forces u to be closer to u_p than u_n by at least a margin m. See Appendix A for an elaboration on the intuition of 2-NR as well as a visualization. In the word–document graph case, $K = 2$ works specifically because we know that document nodes are only connected to word nodes and vice versa (see Section 5 for a description of our graph construction method). Hence, when starting at a document node, all nodes in its neighborhood are word nodes and similarly, all those word nodes do exclusively have edges to document nodes. Hence any walk of two steps starting at some document, will end up at another document via a word (node) they both contain. This simplifies the implementation for our specific word–document graph, but one can easily imagine generalizing the method to situations where taking K hops on the graph does not guarantee ending up at a node of the same type as the start node by restricting the sampling methods to a subset of the desired nodes.

Adaptive pseudo-labeling Pseudo-labeling is a well-explored technique for improving performance in semi-supervised learning settings (Lee et al., 2013), which, recently, has also been successfully applied to graphs (Zhou et al., 2019; Chen et al., 2021). We argue this technique can be particularly powerful for heterogeneous word–document graphs based on three premises:

(1) Different topics/classes have a different distribution of words in their vocabulary. So it can be assumed that there exist words per class that occur more often in documents corresponding to that respective class – i.e. these words are more distinctive for that given class, which in the word–document graph translates to that word node having relatively more edges to documents of the class the respective word is distinctive for.

(2) Document nodes are always at least two hops away from each other in the graph, meaning that only the input features of one document can influence the final feature representation of another document via message passing on the graph. This is assuming a two-layer GNN.

(3) The most effective way of encoding label information in the input document embedding is by directly optimizing for that respective class on the node, as opposed to relying on indirect optimization via backpropagating through the message-passing computational graph.

Instead of applying adaptive pseudo-labeling to the whole graph, we propose to only apply it to a subset of unlabeled document nodes, $\mathcal{U}_d$, that are not part of our train or test split. By doing this, we can directly optimize an unlabeled document embedding to be a good predictor for a certain class (premise (3)) . This class-tailored document embedding can now be propagated over the graph to be used in the final feature representation of other document nodes via message passing on the graph (premise (2)). Finally, we can assume that there exist word nodes in the graph which are characteristic of a topic/class and via which the class-specific features can be propagated to other documents without losing information due to over-smoothing (premise (1)).

We implement adaptive pseudo-labeling as described by Zhou et al. (2019), which adds an extra component to the total loss, the pseudo-label loss $\mathcal{L}_{pse}$:

$$\mathcal{L}_{pse} = \sum_{v_i \in U'} \frac{1}{N_i} CE(\tilde{Y}_i, F_i) \tag{7}$$

With CE representing the cross-entropy loss, $\tilde{Y}_i$ the pseudo-label and $F_i \in \mathbb{R}^C$ the predicted probability per class. The pseudo-label is generated by taking the argmax over F_i, which results in the pseudo-label loss optimizing for high-confidence predictions on the most certain class. The set of unlabeled samples U' used for this loss is defined as:

$$U^{'} = \{u_i : u_i \in U_d | F_{i,j} \leq \beta\}, j = \underset{j^{'}}{argmax} F_{i,j'}\} \tag{8}$$

Some minimum confidence threshold β is used to filter out predictions, and the pseudo-loss per node is weighted by dividing it by N_i, the amount of nodes which have the same predicted label as node u_i and are part of U'.

6 Experimental setting

Throughout our experiments, TextGCN is used as a directly comparable baseline. Since our main goal is to develop a method that performs well in low-resource scenarios for many languages without the need of any knowledge of that language – apart from the ability to identify word boundaries in a sentence – our setup deviates from the original TextGCN work. Unlike the original work, we do not perform a grid-search of hyperparameter settings per experiment/language, but rather keep them fixed – which make our results not directly comparable to the original. Similarly to the original TextGCN work, we also consider the R8 and R52 datasets for an analysis of our approach on English (see Section 4). Additionally, we also provide results for two PLMs trained with the same amount of data. These results are not directly comparable, since the PLMs are trained in an inductive setting, but are included to provide better insight into the positioning of our method in the context of broader literature.

PLM baselines We introduce both multilingual BERT (mBERT) (Devlin et al., 2018) and XLM-R (Conneau et al., 2019) as strong baselines based on the Transformer (Vaswani et al., 2017) architecture. These baselines are fine-tuned in the same data settings, with their architecture settings kept as their defaults as defined in the HuggingFace Transformers library (Wolf et al., 2020). For training, a learning rate of 5e-5 and a batch-size of 20 is used.

Learning settings We investigate the effectiveness of our approach when learning from 1%, 2%, 5%, 10% and 90% of the available training samples. The 1% setting is only considered for the R8 and R52 datasets, due to the already relatively small

% train	Method	MLDoc									Reuters		
		de	**en**	**es**	**fr**	**it**	**ja**	**ru**	**zh**	**Δ**	**R8**	**R52**	**Δ**
1%	mBERT	-	-	-	-	-	-	-	-	-	87.4	73.4	78.6
	XLM-R	-	-	-	-	-	-	-	-	-	87.6	**75.0**	**81.3**
	TextGCN	-	-	-	-	-	-	-	-	-	82.8	65.7	74.3
	+ 2-NR	-	-	-	-	-	-	-	-	-	83.9	68.5	76.2
	+ Pseudo-label	-	-	-	-	-	-	-	-	-	79.1	65.2	72.2
	FewShotTextGCN	-	-	-	-	-	-	-	-	-	**88.6**	69.2	78.9
2%	mBERT	80.2	68.0	57.1	53.8	53.2	56.6	64.0	27.1	57.5	83.8	78.3	81.1
	XLM-R	79.4	79.0	71.1	73.5	62.4	57.5	**70.0**	40.5	66.7	85.8	**81.8**	**83.8**
	TextGCN	60.7	68.2	71.4	35.6	59.3	63.8	55.2	73.4	61.0	84.2	59.2	71.7
	+ 2-NR	85.9	75.3	**75.7**	81.9	**66.9**	70.7	65.7	79.1	75.2	85.3	63.2	74.3
	+ Pseudo-label	83.7	76.2	61.0	70.0	47.1	62.6	58.1	74.8	66.9	80.1	64.5	72.3
	FewShotTextGCN	**86.9**	**84.0**	**75.7**	**83.5**	**66.9**	**78.9**	67.5	**80.3**	**78.0**	**87.2**	65.2	76.2
5%	mBERT	89.1	85.0	74.0	84.3	67.7	77.1	73.5	80.9	79.0	94.7	86.0	90.4
	XLM-R	91.2	84.6	76.3	87.7	**75.8**	81.2	**75.2**	**85.0**	**82.1**	**95.7**	**88.8**	**92.3**
	TextGCN	88.9	73.8	77.2	84.9	70.5	79.6	59.0	80.4	76.8	87.2	67.3	77.2
	+ 2-NR	89.6	85.5	79.4	86.4	75.0	81.0	67.3	81.6	80.1	90.4	69.0	79.7
	+ Pseudo-label	88.9	87.2	77.0	83.8	72.1	79.8	60.6	80.6	78.0	87.4	64.4	75.9
	FewShotTextGCN	**91.5**	**88.6**	**81.2**	**88.7**	72.6	**81.9**	70.0	82.0	**82.1**	90.9	69.2	80.1
10%	mBERT	90.3	87.5	86.3	87.3	77.1	81.2	82.5	82.8	84.4	95.7	85.1	90.4
	XLM-R	91.0	88.1	**88.8**	87.2	75.8	81.9	**83.0**	**88.9**	**85.6**	**96.9**	**92.4**	**94.7**
	TextGCN	90.7	85.5	87.2	86.4	72.5	81.1	72.7	85.1	82.7	89.1	73.6	81.4
	+ 2-NR	89.9	86.8	87.8	87.1	**77.3**	81.1	68.1	84.7	82.9	90.9	76.0	83.5
	+ Pseudo-label	91.5	87.2	87.2	87.8	72.6	**82.5**	74.4	85.3	83.6	90.2	75.6	82.9
	FewShotTextGCN	**91.8**	**90.0**	87.9	**88.7**	74.4	**82.5**	74.4	85.3	84.4	92.5	80.2	86.4
90%	mBERT	91.4	93.2	93.3	94.2	86.6	87.8	86.7	90.9	90.5	95.8	94.5	95.2
	XLM-R	**95.2**	**94.2**	**95.9**	93.4	**87.1**	86.9	**88.7**	**91.3**	**91.4**	**97.5**	**95.4**	**96.5**
	TextGCN	94.5	91.9	94.3	**93.6**	85.8	**89.1**	82.8	89.5	90.2	94.1	82.0	88.1
	+ 2-NR	94.0	91.1	93.9	92.1	84.7	86.8	84.5	88.6	89.5	95.3	83.1	89.2
	+ Pseudo-label	94.3	92.1	94.3	92.3	85.6	88.5	82.9	89.4	89.9	94.1	82.3	88.2
	FewShotTextGCN	94.2	91.8	94.5	92.1	84.3	87.7	84.5	89.3	89.8	95.4	85.1	90.3

Table 1: Average accuracy across 5 different seeds on the test set using a different number of training samples available. Δ corresponds to the average accuracy across all datasets/languages. Methods starting with "+" correspond to TextGCN extended with one of our corresponding proposed methods at a time. FewShotTextGCN refers to the combination of all our proposed improvements together (i.e., including *both* 2-NR and adaptive pseudo-labeling) as well as our adjusted graph construction method. Highest scoring method per benchmark is marked in **bold**.

training set size in the MLDoc datasets. For all settings except the 90% one, the size of the validation set is equal to the size of the training set (see Appendix D for a background experiment on the influence of the division of a limited set of labeled samples over the train and validation sets). The remaining documents are then added to the word–document graph as unlabeled nodes. For the high-resource setting (90%), the remaining 10% of the training set is used for validation (i.e., no unlabeled nodes). We train all GNN models from scratch for each language and do not rely on any form of transfer- or multi-task learning.

Training setup and hyperparameters We use the Ranger optimizer (Liu et al., 2019; Zhang et al., 2019; Yong et al., 2020), an adapted version of Adam (Kingma and Ba, 2014). All experiments run for 1000 epochs and the model with the lowest validation loss is used at test time. A learning rate of 0.01 and dropout of 0.5 are used throughout all experiments except when mentioned otherwise. All hidden dimensions are set to 64 and in line with the original TextGCN work, we use two layers of GCN followed by one linear layer for classification. The log schedule for training signal annealing as per Appendix A.2 in Xie et al. (2020) is used whenever 2-NR is applied. For pseudo-labeling, we set

% train	Edge types	MLDoc								
		de	en	es	fr	it	ja	ru	zh	Δ
2%	$\mathcal{V}_d - \mathcal{V}_w$+$\mathcal{V}_w - \mathcal{V}_w$	60.7	68.2	71.3	35.6	59.3	63.8	55.2	73.4	60.9
	$\mathcal{V}_d - \mathcal{V}_w$	**74.6**	**71.7**	71.3	**76.7**	**59.9**	**65.1**	**59.9**	**75.4**	**69.3**
5%	$\mathcal{V}_d - \mathcal{V}_w$+$\mathcal{V}_w - \mathcal{V}_w$	88.4	73.7	77.2	**84.9**	70.5	**79.6**	59.0	**80.4**	76.7
	$\mathcal{V}_d - \mathcal{V}_w$	**88.5**	**80.7**	**78.1**	82.7	**71.5**	78.4	**60.2**	80.0	**77.5**
10%	$\mathcal{V}_d - \mathcal{V}_w$+$\mathcal{V}_w - \mathcal{V}_w$	**90.7**	85.5	**87.2**	86.4	72.4	81.0	72.7	83.7	82.5
	$\mathcal{V}_d - \mathcal{V}_w$	90.3	**86.8**	87.0	86.4	**75.8**	**82.3**	**74.7**	**85.1**	**83.6**
90%	$\mathcal{V}_d - \mathcal{V}_w$+$\mathcal{V}_w - \mathcal{V}_w$	**94.5**	91.9	94.2	**93.4**	85.8	**89.1**	82.8	**89.5**	90.2
	$\mathcal{V}_d - \mathcal{V}_w$	94.1	91.9	**94.4**	93.0	**86.6**	88.7	**85.0**	89.4	**90.4**
#edges	$\mathcal{V}_d - \mathcal{V}_w$+$\mathcal{V}_w - \mathcal{V}_w$	7.4M	11M	5.5M	8.4M	5.2M	4.9M	10.2M	5.2M	7.2M
	$\mathcal{V}_d - \mathcal{V}_w$	1M	1.3M	900K	1.1M	758K	1.1M	1.1M	889K	1M

Table 2: Average accuracy of 5 different seeds on the test set, with a different number of training samples available. Here, the original TextGCN model is used and only the graph-construction method is varied. Δ corresponds to the average accuracy across seeds. Highest scoring method per language is marked in **bold**.

the confidence threshold $\beta = 0.75$ following the original paper.

7 Results

7.1 Comparison to TextGCN

MLDoc Table 1 shows the results of our experiments. In the 2% training data setting, FewShotTextGCN outperforms TextGCN by 17% points on average (Δ) on the eight languages of the MLDoc dataset, showing that we can effectively utilize the properties of heterogeneous word–document graphs to improve learning in low-resource settings in many languages. For MLDoc, which is a dataset with uniform class priors, we see the difference in performance between original TextGCN and TextGCN combined with 2-NR grows larger as the amount of training data decreases, demonstrating that our proposed 2-NR regularizer helps to combat overfitting. Comparing the '+2-NR' results to those of FewShotTextGCN (that uses both 2-NR and adaptive pseudo-labeling), it can be seen that, overall, our regularizer is the primary contributor in outperforming the TextGCN baseline. Our version of adaptive pseudo-labeling also outperforms the TextGCN baseline, with the largest margins in the low-resource settings, indicating the effectiveness of utilizing unlabeled document nodes in the word–document graph.

In the high-resource (90%) setting of MLDoc, FewShotTextGCN performs on a par with the original TextGCN. This can be explained by the fact that 2-NR is a regularization method and the training data set is relatively large in the high-resource setting, which makes that adding regularization to the learning process can be redundant. Furthermore, our version of adaptive pseudo-labeling works based on a set of unlabeled documents not belonging to either the train or the test set, which is a relatively small set of documents in this setting, namely only 10% of the documents of the total training set.

Reuters Interestingly, FewShotTextGCN outperforms TextGCN consistently in all data settings for the English Reuters datasets, which are highly skewed in their class distribution. This can be seen as supporting evidence for the hypothesis that 2-NR forces the learned feature representations of documents to contain information of all words it contains, which helps to learn distinguishing features for documents of minority classes.

7.2 Comparison to PLMs

MLDoc Although FewShotTextGCN only uses $\approx 1\%$ of the parameters, has no pretrained knowledge of the considered languages, has no notion of word order in the documents and does not make use of a shared subword vocabulary, it performs on par with large PLMs across all settings for MLDoc. In the lowest resource setting, FewShotTextGCN outperforms all considered PLMs, whereas both PLMs start performing on par as the amount of available data increases. We hypothesize that the somewhat larger difference in performance for the Russian language is attributable to the fact that Russian is a highly inflective language, resulting in many unique words to learn a representation for. The PLMs have the advantage of using a subword vocabulary which serves as a remedy for the formerly described sparsity challenge.

Reuters For R8 holds that similarly to the results on MLDoc, FewShotTextGCN outperforms the PLM baselines in the two lowest-resource settings, whereas the PLMs perform better when more training data is available. The results on R52 are more notable, as the gap in performance between FewShotTextGCN and the PLMs grows relatively larger with more available training data. We hypothesize this could be due to the fact for FewShotTextGCN we use only a 64 dimensional hidden size to encode the 52 classes of the dataset, whereas the PLMs use a hidden size of 768.

8 Ablation experiments

The original TextGCN implementation proposes to use edges between words based on their respective PMI. Since PMI is calculated using a window size of 20, many extra edges are introduced. For the MLDoc dataset, omitting word–word edges results in a graph that has, on average, only 15% of the amount of edges compared to the original graph (see Table 2 for statistics on the number of edges per graph construction method). To analyse the effect of word–word edges, we evaluate the original TextGCN method across the different graph construction methods in the same data availability settings as our main set of experiments (Table 2). The results provide empirical evidence that, on average, word–word edges are redundant in topic classification problems. The average performance using graphs without word–word edges is always higher; however, performance difference between the two graph construction methods does get smaller as more data is added. In Appendix E we present a visual walk-through of how words can still influence each other's feature representations in a graph without word–word edges.

9 Discussion

K-NR for K > 2 Here, we argue by example that K-NR can also be applied to other heterogeneous graphs with two or more different kinds of nodes. Consider a network with three kinds of nodes: venue nodes, paper nodes and author nodes (Shi et al., 2016). Venue nodes have a connection to a paper node if the paper is published at that venue and authors have a connection to the paper node when they are a contributor to that respective paper. No other edges exist on this graph and the classification task concerns the author nodes. In this case, we could apply K-NR on the author nodes based on the intuition that authors that publish a paper at the same venue are more similar to each other than authors that do not publish at the same venue. In order to get from the anchor author node to a positive author node, one has to traverse the graph by hopping to a neighboring paper, venue, paper and finally author node in that respective order – resulting in $K = 4$. On this same graph, K-NR can be applied for paper nodes as well for $K = 2$ and traversing via the venue node. In general, considering a graph with M different node types, K-NR can be applied if in terms of node types a symmetrical path with an odd number of nodes can be traversed. In this case, $K = 2(M - 1)$.

10 Conclusion

We introduced K-hop Neighborhood Regularization (K-NR), a contrastive learning method for heterogeneous graphs, and showed its implementation for word–document graphs (2-NR) is highly effective in improving learning from scratch in low-resource settings for a range of languages on topic classification tasks. We also showed that we can exploit properties of word–document graphs for improved learning in few-shot settings. We demonstrated that by simplifying the graph construction method via omitting word–word edges we can improve performance while reducing memory requirements in terms of total number of edges. Additionally, we showed how pseudo-labeling can be successfully applied to word–document graphs. All approaches combined together form part of our new proposed method, FewShotTextGCN, an improvement over TextGCN for few-shot graph learning. FewShotTextGCN performs on par with large PLMs across the considered benchmarks using only a fraction of the parameters and no pretraining whatsoever, showing that GNNs are an attractive alternative for these Transformer-based models. Finally, using this method, we showed that transductive document classification can be performed successfully on a wide range of typologically diverse languages without any language model pretraining. In future work, we plan to explore the effectiveness of 2-NR on a large range of graphs, such as social networks, citation networks and product–user networks as well as adaptations of K-NR for $K > 2$.

11 Limitations

Our work focused on a subset of the text-classification field, namely topic classification. In order to generalize our contributions to other subsets such as sentiment classification, our method might benefit from incorporating word order (Johnson and Zhang, 2014). Secondly, adding 2-NR to the training process does slow down the convergence rate of training. Exemplified: regular TextGCN would often reach its lowest validation loss in the range of 50 to 200 update steps, whereas TextGCN + 2-NR would often reach its lowest validation loss in the range of 700 to 900 update steps. We do not consider this a major limitation as all experiments can still be performed on a single GPU with 8Gb of RAM.

References

Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. 2016. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Bmvc*, volume 1, page 3.

Ethan C Chau and Noah A Smith. 2021. Specializing multilingual language models: An empirical study. *arXiv preprint arXiv:2106.09063*.

Long Chen, Venkatesh Ravichandran, and Andreas Stolcke. 2021. Graph-based label propagation for semi-supervised speaker identification. *arXiv preprint arXiv:2106.08207*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. Be more with less: Hypergraph attention networks for inductive text classification. *arXiv preprint arXiv:2011.00387*.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Zheng Li, Mukul Kumar, William Headden, Bing Yin, Ying Wei, Yu Zhang, and Qiang Yang. 2020. Learn to cross-lingual transfer with meta graph learning across heterogeneous languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2290–2301.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, et al. 2020. Xglue: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. Bertgcn: Transductive text classification by combining gcn and bert. *arXiv preprint arXiv:2105.05727*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8409–8416.

Paul McCann. 2020. fugashi, a tool for tokenizing japanese in python. *arXiv preprint arXiv:2010.06858*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Giannis Nikolentzos, Antoine Tixier, and Michalis Vazirgiannis. 2020. Message passing attention networks for document understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8544–8551.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Holger Schwenk and Xian Li. 2018. A corpus for multilingual document classification in eight languages. *arXiv preprint arXiv:1805.09821*.

Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Niels van der Heijden, Samira Abnar, and Ekaterina Shutova. 2020. A comparison of architectures and pretraining methods for contextualized multilingual word embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9090–9097.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Ziyun Wang, Xuan Liu, Peiji Yang, Shixing Liu, and Zhisheng Wang. 2021. Cross-lingual text classification with heterogeneous graph neural network. *arXiv preprint arXiv:2105.11246*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Hongwei Yong, Jianqiang Huang, Xiansheng Hua, and Lei Zhang. 2020. Gradient centralization: A new optimization technique for deep neural networks. *arXiv preprint arXiv:2004.01461*.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. 2019. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9597–9608.

Ziang Zhou, Jieming Shi, Shengzhong Zhang, Zengfeng Huang, and Qing Li. 2019. Effective semi-supervised node classification on few-labeled graph data. *arXiv preprint arXiv:1910.02684*.

A 2-NR intuition and visualization

In Figure 1, we present a visualization of the process behind 2-NR. Intuitively, 2-NR forces the model to incorporate information of all words contained in it such that documents with shared neighbors (i.e., overlap in vocabulary) are closer to each other in semantic space than documents without shared neighbors in feature space, resulting in learning better feature representations of documents.

B Dataset statistics

Table 3 presents an overview of data statistics.

C Visualization of document embeddings

Figure 2 shows an example of the difference in class separability between TextGCN and our method, FewShotTextGCN. It can be easily seen that using FewShotTextGCN there is less overlap between regions in which instances of the respective classes live, which is in line with the observations in Table 1 where FewShotTextGCN outperforms TextGCN by 26% points absolute accuracy.

D Effective use of data on a limited budget: training vs validation

Figure 3 shows the average accuracy and standard deviation of 10 seeds when learning from a total of

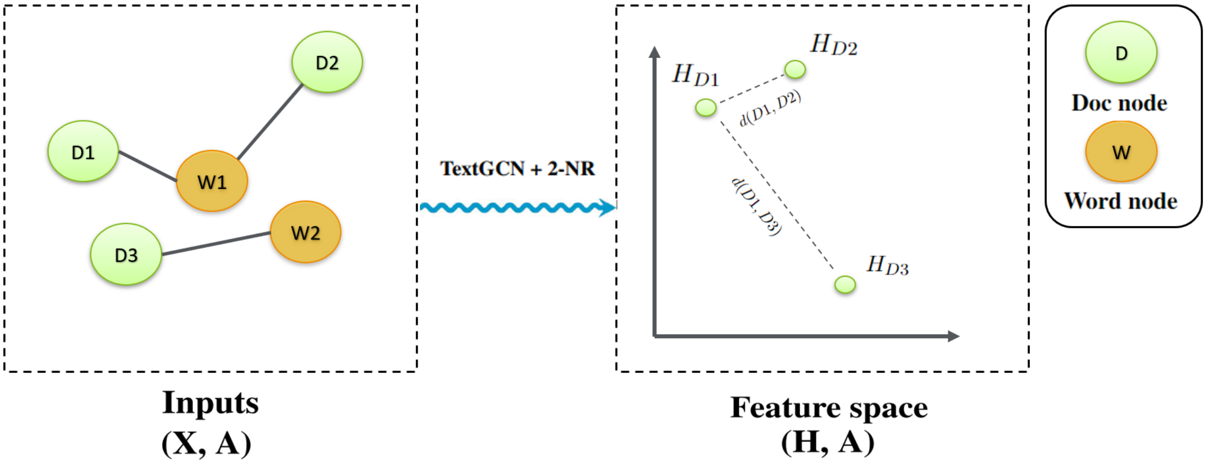


Figure 1: **2-hop Neighborhood Regularization visualized for a simple graph.** On the left, the graph is defined as a set of nodes X and an adjacency matrix A. The right figure depicts the transformed features H of each document D_i. Documents $D1$ and $D2$ have word $W1$ as shared neighbor and hence their distance in feature space (depicted by the dotted line with $d(D1, D2)$) gets smaller, whereas the feature representations of $D1$ and $D3$ get pushed away from each other, resulting in a larger distance in the respective space ($d(D1, D3)$). Word nodes are omitted in feature space (right-hand side) for demonstration purposes.

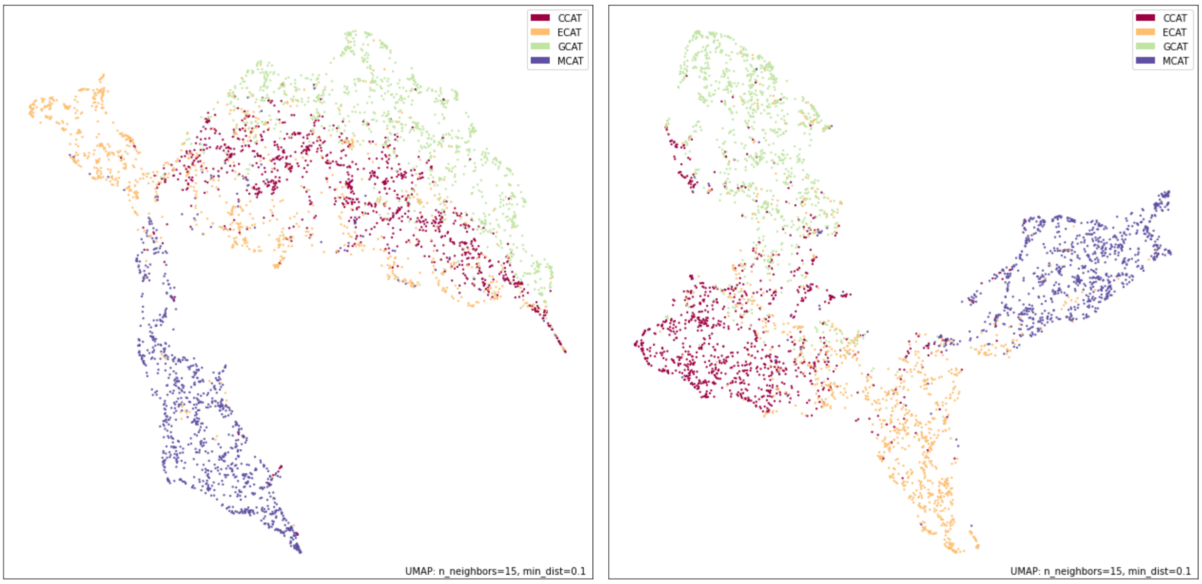
Table 3: Summary statistics of datasets.

Dataset	# Docs	# Training	# Test	# Words	# Nodes	# Classes	Average Length
MLDoc-de	5,000	1,000	4,000	14,358	19,358	4	144
MLDoc-en	5,000	1,000	4,000	17,665	22,665	4	215.6
MLDoc-es	5,000	1,000	4,000	11,662	16,662	4	143.2
MLDoc-fr	5,000	1,000	4,000	15,231	20,231	4	175.8
MLDoc-it	5,000	1,000	4,000	10,075	15,075	4	103.8
MLDoc-ja	5,000	1,000	4,000	8,423	13,423	4	271
MLDoc-ru	5,000	1,000	4,000	19,786	24,786	4	167.3
MLDoc-zn	5,000	1,000	4,000	9,270	14,270	4	163.2
R8	7,674	5,485	2,189	7822	15,496	8	98.9
R52	9,100	6,532	2,568	9027	17,992	52	106.3

100 labeled samples, divided over the train and validation set for Japanese and German, using the original TextGCN model. Other languages are omitted from the plot to prevent visual cluttering, but follow a similar trend: when using too little data for training, the model fails to learn to generalize well – as can be seen by the relatively low average accuracy on the test set. For German, it is even the case that learning does not converge at all when training on 10 samples, as we observe a mean accuracy of 25% when the number of training samples is 10. The balance between achieving the highest accuracy with the smallest standard deviation seems to be around the 50/50 split point. Increasing the training data at the cost of fewer validation data after this point can, in some cases, such as for Japanese at the 80/20 point, result in slightly higher accuracy, but the standard deviation across different seeds also increases, confirming the importance of a good-sized validation set during learning.

E On the usefulness of word–word edges in word–document graphs: illustration

We argue that the added benefit of word–word edges is limited based on the premises that: 1) words occurring in the same window might not influence each other's meaning at all, especially since sentence boundaries are not taken into account; 2) only global information of word co-occurrence (over the whole corpus) is considered, meaning that word A might be connected to word B, but they might not co-occur in document X – yet, they still influence each other's feature representation as much as when they would have co-occurred in that document; 3) over the course of training, words co-occurring in a document can still influence each



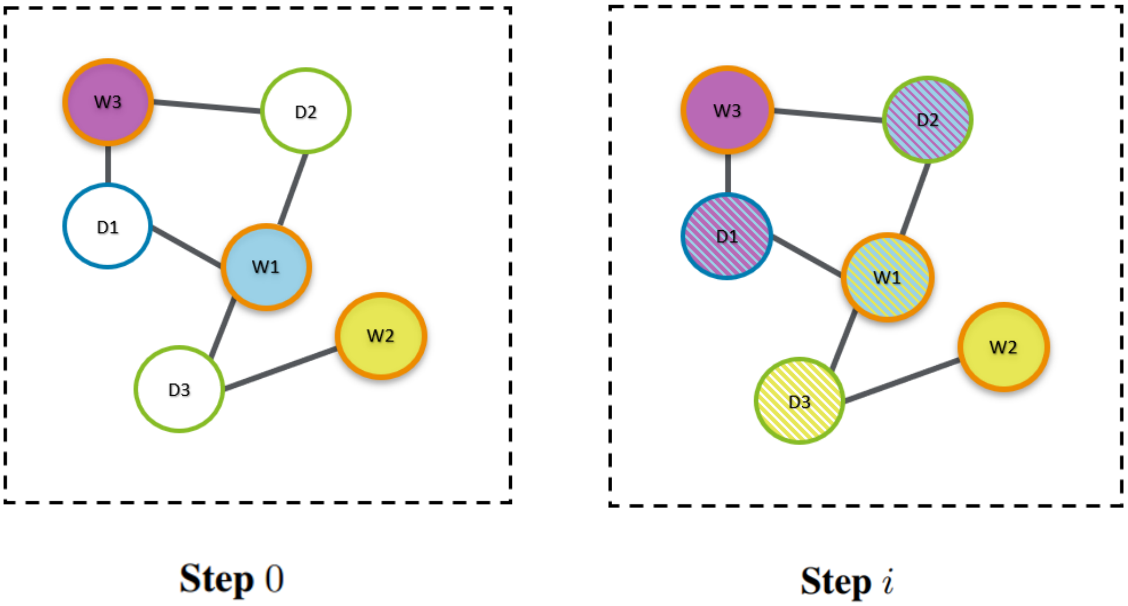
(a) Original TextGCN

(b) FewShotTextGCN

Figure 2: Visualization of the test set document embeddings in the final feature space of TextGCN and FewShotTextGCN on MLDoc-de using the 2% training data setting. UMAP (McInnes et al., 2018) is used for dimensionality reduction

Figure 3: Average test accuracy and standard deviation for 10 seeds with different amounts of data used for training. Given a total of 100 labeled samples, we vary how many are used for training and validation, adjusted in steps of 10.

other's representation, under the assumption that at least a two-layer GNN is used. We illustrate this by visualizing the initial node embeddings over the course of a hypothetical training schedule, starting at a random initialization in Figure 4a and going up until some training step i in Figure 4, where the model has improved its performance in the classification task.



Step 0

Step i

(a) **Words influencing each other's feature representation without word–word edges over the course of training: initialization.** Classes are represented in different node outline color (i.e., two document classes *green, blue*) and words with an *orange* node outline. The actual color of nodes represents the semantic value encoded in the initial embedding. We simplify the graph by omitting the initial features of the document nodes $D1$, $D2$ and $D3$ and assume they will be learned based on the features of the words occurring in them. Since both $W1$ and $W3$ occur in $D1$ and $D2$, they can intuitively not be distinguishing features for either the *green* or the *blue* class.

(b) **Words influencing each other's feature representation without word–word edges over the course of training: during the course of training.** Classes are represented in different node outline color (i.e., two document classes *green, blue*) and words with an *orange* node outline. The actual color of nodes represents the semantic value encoded in the initial embedding. After i steps of training, $D3$ will have incorporated some of the information of $W2$ in its own embedding (depicted by the yellow stripes in the node) in order for it to be a good predictor of the *green* class. Similarly, $D1$ will have incorporated features of $W3$ in its own embedding (depicted by the purple stripes in the node), as it is the most distinguishing word for the *blue* class. Finally, if we consider two-layer GNNs, the feature representation at layer 1 of $W1$, denoted by H^1_{W1}, is a function of its own embedding and the aggregated messages from H^0_{D1} and H^0_{D2} and H^0_{D3}. Similarly, the final feature representation of $D2$, H^2_{D2}, is a function of H^1_{D1} and aggregated messages from H^1_{W1} and H^1_{W2}. In order to perform the right classification, H^2_{D2}, the information corresponding to the *green* class (yellow filling of node) should be passed on from H^0_{D3}, which is the document embedding of $D3$, to H^1_{W1}, and then finally to H^2_{D2}. Due to the nature of GCNs, no distinction can be made between what message is sent from what node – as per Equation 1, $x_i = H^0_{W1}$, it can be easily seen that optimizing H^0_{W1} to be a good predictor for the *green* class (i.e., adjusting its feature representation to the yellow color), helps in classifying $D2$ correctly.

Figure 4: A visualization with guiding explanation to provide an intuition behind how features of words might still influence each other with having direct edges between them

What Clued the AI Doctor In? On the Influence of Data Source and Quality for Transformer-Based Medical Self-Disclosure Detection

Mina Valizadeh, Xing Qian, Pardis Ranjbar-Noiey, Cornelia Caragea, and **Natalie Parde**
Department of Computer Science
University of Illinois Chicago
{mvaliz2, xqian20, pranjb3, cornelia, parde}@uic.edu

Abstract

Recognizing medical self-disclosure is important in many healthcare contexts, but it has been under-explored by the NLP community. We conduct a three-pronged investigation of this task. We (1) manually expand and refine the only existing medical self-disclosure corpus, resulting in a new, publicly available dataset of 3,919 social media posts with clinically validated labels and high compatibility with the existing task-specific protocol. We also (2) study the merits of pretraining task domain and text style by comparing Transformer-based models for this task, pretrained from general, medical, and social media sources. Our *BERTweet* condition outperforms the existing state of the art for this task by a relative F_1 score increase of 16.73%. Finally, we (3) compare data augmentation techniques for this task, to assess the extent to which medical self-disclosure data may be further synthetically expanded. We discover that this task poses many challenges for data augmentation techniques, and we provide an in-depth analysis of identified trends.

1 Introduction

Self-disclosure is a complex communicative process (Kreiner and Levi-Belz, 2019) that involves sharing one's personal thoughts, feelings, or memories with another individual (Jourard and Friedman, 1970). Reciprocal self-disclosure between conversation partners may strengthen relationships (Altman and Taylor, 1973) and improve the communicative experience (Wang et al., 2016). Self-disclosure may take nuanced forms, such as *medical self-disclosure* (see Figure 1), or the act of disclosing symptoms, diagnoses, or other information related to mental or physical health problems (Valizadeh et al., 2021; Joinson, 2001). Some medical self-disclosures may be explicit (A), whereas others may be less direct (B and C).

Medical self-disclosure is helpful from a clinical perspective and reinforces therapeutic relationships during medical interactions (Jannat, 2018; Kadji and Mast, 2021). It also may enable earlier detection and treatment of medical issues (Joinson, 2001; Tidwell and Walther, 2002; Valizadeh et al., 2021). Detecting medical self-disclosure automatically could support clinicians and other medical practitioners in productively identifying helpful patient information from untapped sources or during review of medical conversations (Farber, 2003; Stricker, 2003; Valizadeh and Parde, 2022; Kaelin et al., 2021). However, research towards automated medical self-disclosure detection has been limited and may require different techniques from those used to detect self-disclosure in the general domain (Valizadeh et al., 2021; Reuel et al., 2022).

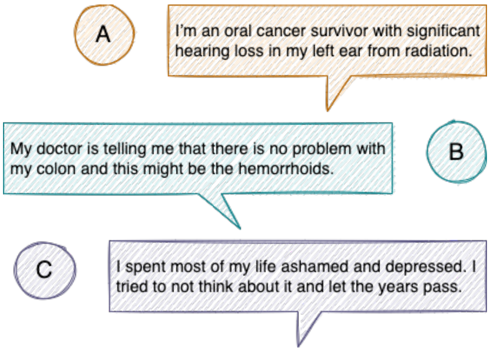


Figure 1: Examples of medical self-disclosure. Note that medical self-disclosure may be explicit (A) or less direct (B and C).

In this paper, we comprehensively investigate automated medical self-disclosure detection. We manually expand the only existing dataset (MEDSD) in this domain to include nearly 4,000 additional instances collected from publicly available forums, strengthening our understanding of this task and its relationship between performance and dataset size. In parallel, we experiment with data augmentation to study the feasibility of automated dataset expansion for this task. Finally, we perform a comparative analysis of transfer learning models, to disentangle the subtle distinctions and shared charac-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1201–1216
May 2-6, 2023

teristics between medical self-disclosure detection and other language tasks. Our primary contributions are as follows.

First (1), we create a publicly-available 3,919-instance expansion to MEDSD, sourced from health-related social media posts. The dataset is triple-annotated with continuous (0-5) and graded (NO SELF-DISCLOSURE, POSSIBLE SELF-DISCLOSURE, and CLEAR SELF-DISCLOSURE) labels. Next (2), we conduct transfer learning experiments using Transformer-based models to determine the utility of different source datasets in the context of detecting medical self-disclosure. We find that our highest-performing model outperforms the state of the art (Valizadeh et al., 2021) by relative percentage increases of 14.19% and 16.73% for accuracy and F_1, respectively. Finally (3), we conduct the first study of data augmentation in the context of medical self-disclosure. We find that this task poses unique challenges for data augmentation, and we explore these challenges in a detailed analysis. It is our hope that our findings add to the burgeoning knowledge base surrounding the detection and processing of medical self-disclosure, and can be used as a guide for others working within this domain. We make our data and source code publicly available to facilitate replication and rapid follow-up.

2 Related Work

Clinical literature has extensively examined the effects of patient self-disclosure with medical professionals, finding both positive benefits to the patient's health and overall wellbeing (Arroll and Allen, 2015), as well as negative experiences if the patient feels that professional or personal boundaries have been crossed (Lussier and Richard, 2007). Often self-disclosure in these studies is broadly construed, with less frequent exploration of medical self-disclosure specifically (Wagner, 1982). For instance, Oprescu et al. (2013) examine the correlation between information-seeking behaviors and self-disclosure, and Weisband and Kiesler (1996) investigate disclosure in online and in-person settings through statistical meta-analyses of standardized interviews, questionnaires, tests, and scales reported in existing studies. They conclude that disclosure of sensitive and private information occurs more frequently in online settings, highlighting the urgency of analyzing these settings more closely.

Bak et al. (2014) developed a semi-supervised algorithm to classify self-disclosure levels from Twitter conversations, and Blose et al. (2020) studied self-disclosure in a large dataset of Twitter conversations about the Coronavirus pandemic, proposing an unsupervised approach to detect voluntary disclosure of personal information. Yang et al. (2017) detected positive and negative self-disclosures based on linguistic features, such as LIWC and word embeddings, in a supervised model. Recently, Reuel et al. (2022) created a multi-task RoBERTa model to measure self-disclosure across varying domains, including both general and medical self-disclosure. They found poor generalization across datasets, with in-domain self-disclosure detection models performing much better than across-domain models due to widespread differences in the topics and targets of self-disclosure.

Only two existing works have focused on detecting medical self-disclosure specifically. Balani and De Choudhury (2015) predicted levels of self-disclosure of mental wellness in Reddit posts, and in our prior work we introduced the MEDSD dataset and established a Transformer-based performance benchmark for the task of medical self-disclosure detection (Valizadeh et al., 2021). Since the latter is the current state of the art and offers the only publicly available dataset and corresponding benchmark in this domain, we primarily build upon that work. We investigate medical self-disclosure specifically, and also study the manual and automated expansion of the MEDSD dataset.

3 Data

3.1 Data Collection

MEDSD, our source dataset, is a 6,639-instance dataset annotated with continuous and graded (NO SELF-DISCLOSURE, POSSIBLE SELF-DISCLOSURE, and CLEAR SELF-DISCLOSURE) medical self-disclosure labels. Since our work seeks to empirically compare the MEDSD benchmark (a fine-tuned DistilBERT model) and other Transformer-based models, we sought to expand it to a size more consistent with that used to fine-tune other Transformer-based models (typically 10k or more instances (Devlin et al., 2019)) while recognizing the time-consuming and challenging nature of the annotation task (Farzana et al., 2020). This also presented the opportunity to study the relative merits of manual and automated (via augmentation) dataset expansion for the task of medical self-disclosure detection.

To ensure consistency with the original MEDSD, we followed the same annotation procedures outlined in our previous work (Valizadeh et al., 2021). We selected *patient.info*[1] as our sole data source, and downloaded publicly-available English-language posts from randomly-selected subforums. Our rationale for selecting *patient.info* as our sole data source (rather than our primary data source as in the original data collection process) was largely due to task-based motivations—we wanted to maximize our capacity to collect data with medical information, and we observed adequate diversity in terms of style and content when performing a preliminary review of *patient.info* posts. To a lesser extent, our decision was also motivated by site-specific data privacy and sharing policies. Since posts collected from other websites in our prior work already comprised a small minority of the dataset (less than 12%), we did not anticipate that this difference would result in meaningful distributional shifts between the original and expanded datasets in terms of data content or style.

We provide additional details and examples regarding our download process in the appendix (Tables 7 and 8). Posts were complete written utterances submitted by users, and we automatically segmented posts at the paragraph level following previously established guidelines. Collected data instances had an average length of 41 tokens. Overall, we added 3,919 instances to MEDSD to reach a raw, merged dataset size of 10,558 instances. To preserve user privacy, we did not download usernames or other identifying metadata and replaced any names written directly within the post text with generic name tokens. The terms and conditions of *patient.info* maintain public access to forum posts, and this research was approved by the Institutional Review Board (IRB) at the University of Illinois Chicago. Similarly to our prior work, we make our dataset expansion available upon request.

3.2 Data Annotation

We observed very high inter-annotator agreement ($\kappa = 0.88$), measured using averaged pairwise per-class Cohen's kappa scores similarly to our prior work to facilitate comparison with annotation quality in the original MEDSD. This suggests that the annotation guidelines were sufficient for consistent replication (Landis and Koch, 1977). We did not make any changes to the guidelines to maximize labeling compatibility between original and new samples. We trained three annotators (computer science graduate and undergraduate students; a mixture of fluent L2 and native English speakers) to follow these guidelines, collecting triple annotations for each post. Labels were first assigned using a graded scheme ranging from 0-5, with "0" indicating no self-disclosure and "5" indicating high self-disclosure. More specifically, we stipulated that numeric labels should be assigned as follows:

- **5:** The post discloses a specific illness, medication, surgery, or other specific medical variable(s) or event(s).
- **4:** The post discloses specific symptoms but does not further specify an illness, medication, or other diagnosis.
- **1-3:** The post ranges from very low (ambiguous hinting of possible, non-specific medical concerns) to moderate (clear reference to non-specific medical concerns) self-disclosure.
- **0:** The post does not disclose any health-related information.

Annotations were averaged across all three labels for each instance. Cases for which the distance between one or more individual annotators and the average was greater than 1.0 were forwarded to a third-party adjudicator. The adjudicator (a study lead with high task familiarity) was authorized to decide the gold standard value based on the provided annotations and their judgment of the instance itself. The continuous-valued (averaged or adjudicated) labels were binned into three discrete classes: [0-1] NO SD, (1-4) POSSIBLE SD, and [4-5] CLEAR SD. This resulted in 520 *No SD* instances (13.26% of the expanded dataset), 1,545 *Possible SD* instances (39.42%), and 1,854 *Clear SD* instances (47.30%).

When collecting our data, we ensured that none of our newly annotated posts were duplicates of those already present in the existing MedSD, to maximize the contribution of new information and content in this dataset expansion. We also spot-checked agreement early in the annotation process to identify and resolve systemic disagreements before they could take root. We worked with in-person (rather than crowdsourced) annotators to allow further oversight for compliance with annotation guidelines as needed. We observed very high

---

[1] `https://patient.info`, a popular online forum that offers publicly available information and posts on health, disease, and other medical topics.

Dataset	No SD	Pos. SD	Clear SD	Total
MedSD	2651	1019	2969	6639
Expansion	520	1545	1854	3919
Merged	3171	2645	4823	10558
Final	**2945**	**2172**	**4650**	**9767**

Table 1: Medical self-disclosure datasets statistics. The final dataset refers to the merged dataset comprising both the original MedSD and our new dataset expansion after refinement.

final agreement: 93.59% of records did not require adjudication, and the averaged pairwise Cohen's kappa (Landis and Koch, 1977) was 0.81 across the entire dataset, indicating high agreement in line with that originally reported for MedSD. Per-class agreement was calculated at $\kappa = 0.85$, $\kappa = 0.74$, and $\kappa = 0.85$ for the *No SD*, *Possible SD*, and *Clear SD* classes, respectively. We release both the averaged scores and discretized class labels with our dataset.

3.3 Dataset Refinement

We observed that a sizable portion of mispredictions by the MedSD baseline on our new expansion were adjudicated instances. After further manual review of the full MedSD including our expansion, we discarded instances that required adjudication (791 total) from further inclusion in our experiments. Our rationale for this refinement step is that these cases may have arisen from ambiguous context, systemic and conflicted understanding of certain nuances in the annotation guidelines, or occasional annotation mistakes that produced labels that the adjudicator considered to have reasonable justification (Kilgarriff, 1998).

After refinement, the final, merged dataset comprising the pruned MedSD and our expansion without adjudicated records includes 9,767 instances (4,650 *Clear SD*, 2,172 *Possible SD*, and 2,945 *No SD* instances). Table 1 provides more details regarding the dataset composition and class distribution. Table 9 in the appendix also illustrates the performance change observed when running the benchmark model on the final, refined dataset versus the raw merged version. We release our refined MedSD alongside our other data as an additional resource for the community.

3.4 Clinical Validation

Since our annotators did not have external clinical expertise, we also recruited a clinical expert (a frequent collaborator who holds graduate degrees in the healthcare domain and has worked in clinical settings) to manually label a subset of our data. We randomly selected a sample of 1,465 data instances (15% of the final dataset size, evenly distributed across classes), and asked the expert to assign labels based on our annotation guidelines and drawing upon their own expertise.

We compared the expert labels with our gold standard, and observed high agreement. In total, 92.37% of the expert's labels (1353/1465 instances) matched those in the gold standard, confirming our annotation reliability from a clinical perspective. At a per-class level, we observed 95.97% label compatibility with *No SD* instances, 83.84% compatibility with *Possible SD* instances, and 97.31% compatibility with *Clear SD* instances. In the appendix (Table 15), we briefly discuss some examples of disagreement between the gold standard and the clinical expert for further analysis.

4 Methods

We compared fine-tuned Transformer models from varying source domains on the task of medical self-disclosure detection (§4.2), and also studied the utility of data augmentation for automated dataset expansion within this domain (§4.3). We describe these studies in the following subsections.

4.1 Data Preprocessing

We passed each instance in our final dataset through a two-step preprocessing pipeline. First, we converted all emojis to their American English CDLR short names (e.g., 😊 → *:smiling face with smiling eyes:*), since insight into an author's emotional status may provide valuable clues to the presence of self-disclosure (Eisner et al., 2016; Felbo et al., 2017). Next, we replaced all numeric values with generic Number_Tokens. This prevented our models from drawing spurious conclusions regarding specific values, allowing them to recenter their focus on the presence of numeric content. To validate the utility of these preprocessing steps, we ran the MedSD baseline (Valizadeh et al., 2021) on our final dataset without any preprocessing steps and with each step individually. We report our findings in Table 11 in the appendix, demonstrating that each step results in small but measurable performance improvements.

4.2 Models

One arm of our study focuses on the influence of pretraining style and task domain on task performance. This is naturally supported through the use of existing, externally-validated Transformer models that were originally leveraged for other research problems. We emphasize therefore that the novelty of our work is not in the model implementation itself, but in the study of its application to this new domain and the extent to which these models could sufficiently leverage and generalize from out-of-domain or near-domain data. We focused our comparison of Transformer-based models on those that are pretrained on health or social media data. In total, we consider seven models: *DistilBERT*, *RoBERTa*, *BioBERT*, *Bio-ClinicalBERT*, *Bio-RedditBERT*, *MentalBERT*, and *BERTweet*.

DistilBERT was the highest-performing model in our initial experiments on MEDSD, and is the current state of the art for this task. The model achieves comparable performance to larger Transformer-based models while requiring much less time and space through the use of knowledge distillation (Sanh et al., 2019), and is often used for lower-resource tasks. In contrast, *RoBERTa* replicates BERT but is trained on larger batches, a higher number of epochs, and more training data, often resulting in higher performance than the original BERT model (Liu et al., 2019). Both *DistilBERT* and *RoBERTa* are trained on general domain data (BookCorpus and Wikipedia for both, and additional news and web data for RoBERTa).

BioBERT leverages nearly the same architecture as BERT, but it is pretrained on large biomedical corpora (Lee et al., 2020). Prior work has shown that it outperforms other BERT-based models at biomedical text classification tasks (Mitra et al., 2021; Zhu et al., 2020). Although *BioBERT* is designed to perform well on biomedical tasks specifically, these may differ from tasks using clinical notes or more casual health discourse. *Bio-ClinicalBERT* is pretrained on two million clinical notes from the MIMIC-III v1.4 database (Johnson et al., 2016; Alsentzer et al., 2019), and *Bio-RedditBERT* is initialized from *BioBERT* (Lee et al., 2020) and then further pretrained on health-related Reddit posts (Basaldella et al., 2020). *MentalBERT* is pretrained on mental health posts collected from Reddit (Ji et al., 2022). Finally, *BERTweet* was pretrained using *RoBERTa*'s training procedures on a massive amount (80 GB) of uncompressed English tweet text, including 845,000,000 Tweets streamed from January 2012 to August 2019 and 5,000,000 Tweets related to the Covid-19 pandemic. It outperformed previous models on a wide variety of social media tasks (Nguyen et al., 2020).

We fine-tuned each included model separately for our task. We applied model gradual layer freezing, and optimized model hyperparameters using grid search. Table 13 provides additional details regarding the fine-tuned hyperparameters.

4.3 Data Augmentation

Manual dataset expansion, as described in Section 3, can be expensive and time-consuming. Data augmentation (DA) strategies can be used to automatically increase training set size, offering an attractive way to reduce overfitting or other issues causing poor predictive performance (Bayer et al., 2021; Feng et al., 2021). They also offer an opportunity to shift the class distribution of unbalanced datasets through generation of additional samples for specific classes, and they have grown more common in NLP recently (Feng et al., 2021).

Although DA techniques have not been explored in the context of self-disclosure detection, this task offers an intriguing testbed for these experiments since it relies on nuanced language with uneven class distribution (see Table 1). We experimented with the following DA techniques to synthetically expand our dataset and balance its distribution:

- **Backtranslation Augmentation (BT):** Data is translated to a different target language (in this case, German) and then back to the source language (English). This often paraphrases the original text (Beddiar et al., 2021).
- **WordNet Synonym Augmentation (WS):** One or more words, depending on a fine-tuned hyperparameter *aug_p* that controls the percentage of words to be augmented, are replaced with their synonyms from WordNet, a large English lexical database (Ramachandran and Parvathi, 2021).[2] We set *aug_p*=0.3 in our experiments.
- **Masked Language Model Augmentation (MLM):** A random sample of words is "masked out" and a pretrained Transformer model is used to restore the text to its original version, typically resulting in a paraphrase

---

[2] `https://wordnet.princeton.edu/`

(Wu et al., 2019; Kumar et al., 2020). We experiment with DistilBERT (Sanh et al., 2019) and RoBERTa (Liu et al., 2019) backbones for this technique.

- **Target-Aware Data Augmentation (TA):** The masked word is conditioned on both its context and the prepended auxiliary sentence that contains target and label information, producing alternative versions of the original text that are geared toward a specific target (in our case, the specific target is medical self-disclosure) (Li and Caragea, 2021).

We generated augmented data for the *Possible SD* and *No SD* classes using each of these techniques as a separate condition. We fine-tune the augmentation ratio, a hyperparameter indicating the multiplicative factor by which the dataset size is increased, setting it to 2 for *Possible SD* and 1.5 for *No SD*. Figure 5 in the appendix demonstrates that the DA process resulted in a relatively balanced training set (3,488 *Clear SD*, 3,258 *Possible SD*, and 3,488 *No SD* instances). Following standard protocol we did not augment the validation or test sets, to avoid introducing potential biases and allow for direct comparison with other models.

To compare DA approaches with one another, we train our best-performing model from our modeling experiments (§5.2) on separate combinations of the final, manual dataset and each of the synthetic dataset expansions (*BT*, *WS*, *DistilBERT MLM*, *RoBERTa MLM*, and *TA*). We additionally compare to a baseline *no augmentation* condition trained only on the final, manual dataset.

5 Evaluation

We compare model performance using accuracy, precision, recall, and macro-averaged F_1, following prior work on self-disclosure detection (Valizadeh et al., 2021; Balani and De Choudhury, 2015). For each experiment, we randomly split the specified data into training (80%), validation (10%), and test (10%) subsets. We measure the efficacy of our manual dataset expansion (§5.1), empirically compare the performance of the proposed Transformer models (§5.2), and evaluate the performance of DA techniques for medical self-disclosure (§5.3).

5.1 Does more (manual) data lead to better performance?

Model	**Acc.**	**Precision**	**Recall**	**F_1**
MEDSD	76.77	0.7497	0.7241	0.7313
Final	**80.91**	**0.7832**	**0.7810**	**0.7816**

Table 2: Comparison between DistilBERT models trained on the original MEDSD and final datasets, separately. Accuracy shown as a percentage (%).

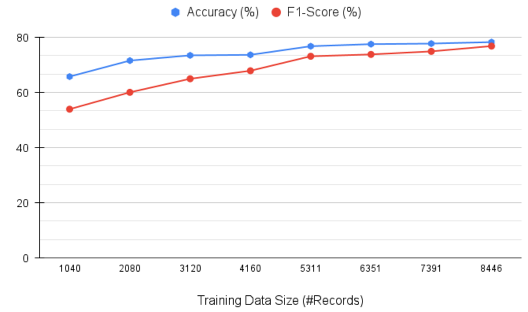


Figure 2: Training dataset size versus *DistilBERT* performance scores. Accuracy and F_1 both shown as a percentage (%) to facilitate presentation.

To investigate whether the manual expansion of MEDSD directly results in increased model performance, we train *DistilBERT*, the MEDSD benchmark, separately on the original MEDSD and the final dataset (Table 2).[3] The model trained on the final dataset exhibits relative percentage increases of 5.39% and 6.87% for accuracy and F_1, respectively, compared to the model trained on MEDSD. This suggests that manual data expansion boosts performance for this task, irregardless of model architecture or pretraining settings.

However, there also appears to be a limit to which more manual data results in meaningful improvements. Figure 2 shows *DistilBERT* performance at gradually increasing training dataset sizes, exhibiting a flattening performance curve approaching the final dataset size. Although this suggests a performance plateau approaching the full size of the manually-annotated dataset, further experiments are needed to justify a broader claim that training set size is detached from performance beyond a certain size threshold. We move closer towards validating this claim with the findings from our data augmentation experiments (§5.3).

---

[3] The MEDSD results reported in Table 2 (top row) are lower than we observed previously (Valizadeh et al., 2021). After replicating those experiments for this study with different random seeds to assess statistical significance, we found that when averaged across numerous runs, multinomial *DistilBERT* achieves the performance values reported here. These scores remain higher than the other baselines studied by Valizadeh et al. (2021), and the conclusions resulting from those earlier experiments still hold (and are statistically significant).

Model	Acc.	Precision	Recall	F_1
DistilBERT	80.91	0.7832	0.7810	0.7816
RoBERTa	84.40	0.8172	0.7927	0.7968
BioBERT	84.35	0.8117	0.8219	0.8173
Bio-ClinicalBERT	83.10	0.8042	0.7889	0.7954
Bio-RedditBERT	85.15	0.8241	0.8250	0.8245
MentalBERT	85.42	0.8357	0.8282	0.8208
BERTweet	**87.67**	**0.8528**	**0.8551**	**0.8537**

Table 3: Model performance comparison. Accuracy shown as a percentage (%).

Technique	Acc.	Precision	Recall	F_1
MLM-R	86.28	0.8388	0.8271	0.8325
MLM-D	85.88	0.8355	0.8228	0.8286
TA	85.48	0.8298	0.8362	0.8321
WS	86.01	0.8363	0.8196	0.8263
BT	**86.42**	**0.8389**	**0.8470**	**0.8420**
NA	**87.67**	**0.8528**	**0.8551**	**0.8537**

Table 4: Performance comparison for DA techniques. *MLM-R* and *MLM-D* use RoBERTa and DistilBERT backbones, respectively. *NA* refers to the *no augmentation* condition. Accuracy shown as a percentage (%).

5.2 How do pretraining style and task domain impact model performance?

We present the results of our model comparison in Table 3. *BERTweet* achieved the highest performance, with accuracy=87.67% and F_1=0.8537. The model outperformed the existing *DistilBERT* baseline by relative percent increases in accuracy and F_1 of 8.35% and 9.22%, respectively. We applied McNemar-Bowker's test ($\alpha = 0.05$), an extension of McNemar's test designed to accommodate more than two classes (McNemar, 1947), to assess the statistical significance of our results. We observe that all differences are statistically significant ($p <= 0.05$).

5.3 Does more (augmented) data lead to better performance?

To investigate whether the synthetic expansion of the dataset leads to further performance boosts beyond those observed from the manual dataset expansion, we retrained *BERTweet* on each augmented version of the dataset described in §4.3. We report our results in Table 4. Backtranslation resulted in the best model performance among all DA approaches with accuracy=86.42% and F_1=0.8537. However, *BERTweet* with *no augmentation* outperformed all other conditions. The outcomes from this experiment suggest that synthetic expansion beyond the current dataset size is unlikely to lead to substantial model improvement. This also offers supporting evidence for the insights from Figure 2, discussed in §5.1.

6 Discussion

6.1 Lessons Learned from Model Comparison

To further disentangle the differences in observed model performances, we computed per-class accuracy for each model (Table 5). The *DistilBERT* baseline had acceptable performance when predicting *No SD* and *Clear SD* instances with 88.70% and 87.60% accuracy, respectively, but poor performance in detecting *Possible SD* instances, dropping to 56.25% accuracy. *RoBERTa* resulted in higher accuracy for the *No SD* and *Clear SD* classes (relative percent increases of 7.04% and 2.79%, separately) while also offering a slight improvement in detecting *Possible SD* instances.

Our task's reliance on social media data in the medical domain drove our selection of models pretrained primarily on social media data (*BERTweet*), medical data (*BioBERT* and *Bio-ClinicalBERT*), and data at the intersection of both domains (*Bio-RedditBERT* and *MentalBERT*) for our experiments. We anticipated that *BioBERT* and *Bio-ClinicalBERT* would result in improved recognition of *Clear SD* instances, and these expectations were confirmed with per-class performance increased by 3.29% and 5.38%, respectively, relative to *DistilBERT*. *Bio-RedditBERT* and *MentalBERT* achieved higher performance still, with relative performance increases in the *Possible SD* class in particular of 20.01% and 21.15% over the baseline, emphasizing the importance of text style in addition to domain specificity.

Interestingly, our strongest model overall was *BERTweet*, which is pretrained primarily on social media data although a small subset of the data did have a specific health focus. This suggests that ultimately, stylistic patterns may be more important than task-based knowledge when recognizing specific forms of self-disclosure. At the class level, models trained entirely on healthcare datasets (e.g., *Bio-ClinicalBERT*) experience performance boosts when detecting *Clear SD* instances, and models trained on social media datasets (e.g., *BERTweet*) achieve substantial increases in their detection of *Possible SD* instances, which was the most challenging class for human annotators (§3.2).

Model	**No**	**Possible**	**Clear**
DistilBERT	88.70	56.25	87.60
RoBERTa	94.95	57.48	90.05
BioBERT	89.79	63.69	90.49
Bio-ClinicalBERT	88.86	58.27	92.32
Bio-RedditBERT	90.71	67.51	91.27
MentalBERT	91.64	68.15	90.83
BERTweet	89.38	78.66	91.45

Table 5: Accuracy per class for all the models implemented for detecting medical self-disclosure. Accuracy is shown as a percentage (%).

Dataset	**No**	**Possible**	**Clear**
BT	93.72	70.94	92.27
NA	89.38	78.66	91.45

Table 6: Comparison of per-class accuracy for the *BERTweet* model trained on the backtranslation-augmented (BT) and non-augmented (NA) datasets. Accuracy is shown as a percentage (%).

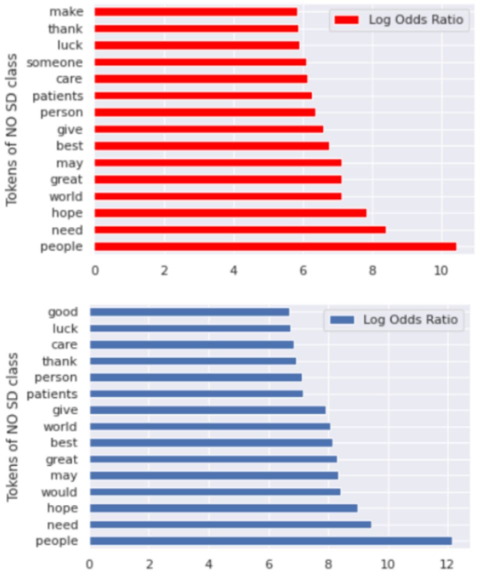


Figure 3: Words most closely associated with No SD class. Red and blue plots correspond to original and augmented data, respectively.

6.2 Factors Influencing Data Augmentation

Our experiments did not provide conclusive evidence that data augmentation can be effectively leveraged to detect medical self-disclosure. This is understandable since the task is known to rely on often-subtle language patterns (Valizadeh et al., 2021; Reuel et al., 2022). We examined the per-class performance of our *BERTweet* model fine-tuned on our backtranslation-augmented dataset compared to the *no augmentation* condition (Table 6) to develop a deeper understanding of opportunities for future improvement.[4]

We find that data augmentation resulted in small performance increases when predicting *No SD* and *Clear SD*, but larger performance reductions when predicting *Possible SD* instances. Although initially counterintuitive since our primary goal in augmenting data was to balance the class distribution (for which frequency was lowest in the *Possible SD* class), we conducted further analyses to pinpoint underlying factors. To analyze the linguistic patterns associated with augmented and non-augmented instances, we computed the log odds ratio with an informative Dirichlet prior (Monroe et al., 2008; Hessel, 2016) for both versions of *No SD* and *Possible SD* (Figures 3 and 4).

We found that after augmenting the dataset, the ratio of third-person nouns (e.g., “people” or “person”) increased in the *No SD* class (Figure 3). This may have strengthened the model’s inverse association between the use of external language and the confirmed disclosure of medical information, resulting in the observed 4.85% increase in accuracy for *No SD*. The ratio of “feeling” keywords in the *Possible SD* class also increased. Prior to DA, these words were more prominent in the *Clear SD* class than the *Possible SD* class (612 and 331 tokens prior to DA, respectively, and 621 and 622 tokens after), due to their use when expressing physical or mental symptoms. We found that 18.1% of mispredicted *Possible SD* instances contained “feeling” keywords prior to DA, and 29.54% of mispredicted *Possible SD* instances contained these keywords after DA. Thus, it appears that backtranslation created a harmful distributional shift in the expression of feeling language across classes. Finally, we suspect that despite its outperformance over other techniques, backtranslation is still limited in its ability to create convincing synthetic data for this task and may not have captured subtleties in writing style that (as observed in our model comparison) are important (Longpre et al., 2020; Beddiar et al., 2021). As a result, the introduction of synthetically augmented data to the learning process may have merely added noise.

Although our data augmentation experiments did not produce positive results, we note that this is the first exploration of augmentation in the context of

---

[4] Per-class performance of *BERTweet* fine-tuned on other augmented datasets is provided in Table 14 in the appendix.

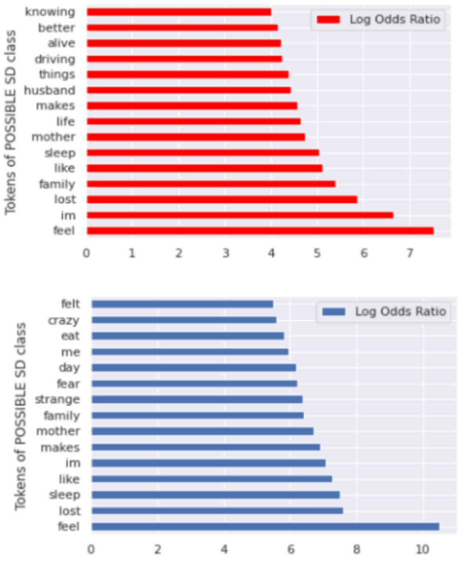


Figure 4: Words most closely associated with POSSIBLE SD class. Red and blue plots correspond to original and augmented data, respectively.

medical self-disclosure. Negative findings move the dial toward more fully understanding the performance boundaries of this and related tasks, and the recent movement towards leaderboards rather than hypothesis-driven investigations in NLP has contributed to the under-reporting of negative results (Tafreshi et al., 2022). This can slow progress as scientists repeat experiments that they are unaware have already been attempted, often with substantial effort and carbon footprint. It is our hope that through reporting the negative outcomes from our DA experiments, we add to the growing knowledge base surrounding data augmentation for NLP and also lay the groundwork for future experiments in self-disclosure detection.

7 Conclusion

In this work, we study data source, quantity, and quality as it pertains to detecting medical self-disclosure. First, we manually expanded MEDSD with 3,919 additional instances and clinically validated its labels in collaboration with a healthcare expert. Next, we compared Transformer models pretrained on varying source datasets for predicting medical self-disclosure, finding that our best-performing model outperforms the state of the art by relative percentage increases of 14.19% and 16.73% for accuracy and F_1, respectively. Our findings also suggest that stylistic patterns prove more revealing than task-specific trends. Finally, we study data augmentation in the context of this task, finding that it poses many DA challenges. We document these in our analysis, opening the door to intriguing follow-up studies. We make our dataset and models available to the research community upon request, and we hope that our work can be used as a roadmap for future experiments in self-disclosure detection.

Acknowledgements

We thank the anonymous reviewers for their helpful suggestions. We thank Dr. Vera Kaelin for providing clinical expertise for validating our labels, and we also thank Shayan Rasheed and Nidhi Bhupalam for their roles in creating the dataset. The first and last authors of this paper were fully or partially supported throughout the duration of this work by the National Science Foundation under Grant No. 2125411. Any opinions, findings, and conclusions or recommendations are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Limitations

Our work is limited by several factors. Both the original MEDSD dataset and our own expanded version are imbalanced, resulting in some known performance weaknesses. We attempted to correct for this using data augmentation techniques, but across the wide range of techniques tested, none were able to improve performance beyond that of the *no-augmentation* baseline. Thus, we cannot provide conclusive evidence that synthetic dataset expansion is a valuable pursuit for this task (although we note that this negative finding in itself provides worthwhile direction for other researchers). The manual expansion of the medical self-disclosure dataset clearly improved performance, highlighting the effectiveness of human gold standard labels for this task; therefore, manual or semi-supervised dataset expansion may be a promising avenue for future model improvements.

Our highest performing model was *BERTweet*, which is pretrained on a massive amount of primarily general-domain social media data, although a small subset of it was focused on the Covid-19 pandemic. This suggests that stylistic patterns may be more important than domain knowledge for the recognition of specific forms of self-disclosure. Therefore, instead of limiting the methodology to transfer learning models in clinical and medical domains, future work should extend the range of domains and text styles that are studied. It may be the case that the most worthwhile domains or stylistic cues for supporting this task are further from the target task, and as yet undiscovered.

All data and experiments reported in this paper were conducted on English data, which constrains the extent to which our claims can be generalized. Future work examining medical self-disclosure in languages that are less-resourced than English or that differ greatly in their morphological typology may provide crucial insight into the generalizability of our findings. Finally, training and fine-tuning large Transformer-based models often requires costly GPU resources. This limits the accessibility of running these experiments at scale.

Ethics Statement

This research was approved by the Institutional Review Board (IRB) at the University of Illinois Chicago. Our primary data source was *patient.info*, for which the terms and conditions allow public access to forum posts.[5] As outlined in §3.1 and to protect privacy, we manually anonymized our data instances by removing any usernames or other identifying metadata, and replaced any names written directly within the post text with generic name tokens. Annotators were compensated for their work through paid internships and assistantship positions at a competitive rate for the cost of living in our area. We make our expanded dataset available upon request via email, following IRB protocol.

We intend for our dataset expansion and the proposed methods to be used as a tool to analyze the linguistic trends and other language behaviors associated with medical self-disclosure in online settings. Our experiments closely follow this intent, providing novel insight into the influence of different pre-training tasks and stylistic domains on the ability of our models to recognize possible and clear cases of medical self-disclosure. When the technology is being used as intended and functioning correctly, we anticipate that it may be of value to numerous downstream applications, primarily as a data analysis tool or as an avenue for providing information. When the technology is being used as intended but giving incorrect results, its value may decrease (since researchers or clinicians using the tool may need to discard their insights or may fail to replicate findings in subsequent experiments).

A potential misuse of the technology would be to use the trained models to identify social media users disclosing medical information and apply targeted advertising or messaging to those users. Since neither our dataset nor our models are designed to recognize or predict specific medical conditions, we note that the extent to which they could be used for these purposes is limited. We do not condone this use of the technology, and we will monitor citations and uses of our dataset to ensure that others are using our data and models for their intended purpose.

References

Emily Alsentzer, John R. Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew B. A. McDermott. 2019. Publicly available clinical BERT embeddings. *CoRR*, abs/1904.03323.

Irwin Altman and Dalmas A Taylor. 1973. *Social penetration: The development of interpersonal relationships.* Holt, Rinehart & Winston.

Bruce Arroll and Emily-Charlotte Frances Allen. 2015. To self-disclose or not self-disclose? a systematic review of clinical self-disclosure in primary care. *British Journal of General Practice*, 65(638):e609–e616.

JinYeong Bak, Chin-Yew Lin, and Alice Oh. 2014. Self-disclosure topic model for classifying and analyzing twitter conversations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1986–1996.

Sairam Balani and Munmun De Choudhury. 2015. Detecting and characterizing mental health related self-disclosure in social media. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pages 1373–1378.

Marco Basaldella, Fangyu Liu, Ehsan Shareghi, and Nigel Collier. 2020. Cometa: A corpus for medical entity linking in the social media. *arXiv preprint arXiv:2010.03295.*

Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2021. A survey on data augmentation for text classification. *arXiv preprint arXiv:2107.03158.*

Djamila Romaissa Beddiar, Md Saroar Jahan, and Mourad Oussalah. 2021. Data expansion using back translation and paraphrasing for hate speech detection. *Online Social Networks and Media*, 24:100153.

Taylor Blose, Prasanna Umar, Anna Cinzia Squicciarini, and Sarah Michele Rajtmajer. 2020. Privacy in crisis: A study of self-disclosure during the coronavirus pandemic. *CoRR*, abs/2004.09717.

---

[5] `https://patient.info/terms-and-conditions`

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

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. In *Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media*, pages 48–54, Austin, TX, USA. Association for Computational Linguistics.

Barry A Farber. 2003. Patient self-disclosure: A review of the research. *Journal of clinical psychology*, 59(5):589–600.

Shahla Farzana, Mina Valizadeh, and Natalie Parde. 2020. Modeling dialogue in conversational cognitive health screening interviews. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1167–1177, Marseille, France. European Language Resources Association.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Jake Hessel. 2016. Implementation: Fightin' words. `https://github.com/jmhessel/FightingWords`.

Khadiza Tul Jannat. 2018. *Self-Disclosure, Gender, and Patient Satisfaction in the Doctor-Patient Relationship*. Minnesota State University, Mankato.

Shaoxiong Ji, Tianlin Zhang, Luna Ansari, Jie Fu, Prayag Tiwari, and Erik Cambria. 2022. MentalBERT: Publicly Available Pretrained Language Models for Mental Healthcare. In *Proceedings of LREC*.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Adam N Joinson. 2001. Self-disclosure in computer-mediated communication: The role of self-awareness and visual anonymity. *European journal of social psychology*, 31(2):177–192.

Sidney M Jourard and Robert Friedman. 1970. Experimenter-subject" distance" and self-disclosure. *Journal of Personality and Social Psychology*, 15(3):278.

Keou Kadji and Marianne Schmid Mast. 2021. The effect of physician self-disclosure on patient self-disclosure and patient perceptions of the physician. *Patient Education and Counseling*, 104(9):2224–2231.

Vera C Kaelin, Mina Valizadeh, Zurisadai Salgado, Natalie Parde, and Mary A Khetani. 2021. Artificial intelligence in rehabilitation targeting the participation of children and youth with disabilities: Scoping review. *J Med Internet Res*, 23(11):e25745.

Adam Kilgarriff. 1998. Gold standard datasets for evaluating word sense disambiguation programs. *Computer Speech & Language*, 12(4):453–472.

Hamutal Kreiner and Yossi Levi-Belz. 2019. Self-disclosure here and now: combining retrospective perceived assessment with dynamic behavioral measures. *Frontiers in psychology*, 10:558.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

J. Richard Landis and Gary G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Yingjie Li and Cornelia Caragea. 2021. Target-aware data augmentation for stance detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1850–1860.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shayne Longpre, Yu Wang, and Christopher DuBois. 2020. How effective is task-agnostic data augmentation for pretrained transformers? *arXiv preprint arXiv:2010.01764*.

Marie-Thérèse Lussier and Claude Richard. 2007. Self-disclosure during medical encounters. *Canadian family physician*, 53(3):421–422.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Avijit Mitra, Bhanu Pratap Singh Rawat, David D Mc-Manus, Hong Yu, et al. 2021. Relation classification for bleeding events from electronic health records using deep learning systems: an empirical study. *JMIR medical informatics*, 9(7):e27527.

Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin'words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16(4):372–403.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *CoRR*, abs/2005.10200.

Florin Oprescu, Shelly Campo, John Lowe, Julie Andsager, and Jose A Morcuende. 2013. Online information exchanges for parents of children with a rare health condition: key findings from an online support community. *Journal of medical Internet research*, 15(1):e2423.

Dharini Ramachandran and R Parvathi. 2021. A novel domain and event adaptive tweet augmentation approach for enhancing the classification of crisis related tweets. *Data & Knowledge Engineering*, 135:101913.

Ann-Katrin Reuel, Sebastian Peralta, João Sedoc, Garrick Sherman, and Lyle Ungar. 2022. Measuring the language of self-disclosure across corpora. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1035–1047, Dublin, Ireland. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

George Stricker. 2003. The many faces of self-disclosure. *Journal of Clinical Psychology*, 59(5):623–630.

Shabnam Tafreshi, João Sedoc, Anna Rogers, Aleksandr Drozd, Anna Rumshisky, and Arjun Akula, editors. 2022. *Proceedings of the Third Workshop on Insights from Negative Results in NLP*. Association for Computational Linguistics, Dublin, Ireland.

Lisa Collins Tidwell and Joseph B Walther. 2002. Computer-mediated communication effects on disclosure, impressions, and interpersonal evaluations: Getting to know one another a bit at a time. *Human communication research*, 28(3):317–348.

Mina Valizadeh and Natalie Parde. 2022. The AI doctor is in: A survey of task-oriented dialogue systems for healthcare applications. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6638–6660, Dublin, Ireland. Association for Computational Linguistics.

Mina Valizadeh, Pardis Ranjbar-Noiey, Cornelia Caragea, and Natalie Parde. 2021. Identifying medical self-disclosure in online communities. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4398–4408, Online. Association for Computational Linguistics.

Clifford H Wagner. 1982. Simpson's paradox in real life. *The American Statistician*, 36(1):46–48.

Yi-Chia Wang, Moira Burke, and Robert Kraut. 2016. Modeling self-disclosure in social networking sites. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, CSCW '16, page 74–85, New York, NY, USA. Association for Computing Machinery.

Suzanne Weisband and Sara Kiesler. 1996. Self disclosure on computer forms: Meta-analysis and implications. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 3–10.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Diyi Yang, Zheng Yao, and Robert Kraut. 2017. Self-disclosure and channel difference in online health support groups. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11, pages 704–707.

Yu Zhu, Lishuang Li, Hongbin Lu, Anqiao Zhou, and Xueyang Qin. 2020. Extracting drug-drug interactions from texts with biobert and multiple entity-aware attentions. *Journal of biomedical informatics*, 106:103451.

A Data Collection

Our data collection and annotation process spanned four months. Similarly to our earlier work (Valizadeh et al., 2021), we covered numerous medical topics, ranging from mental to physical health issues. In Tables 7 and 8, we provide example instances from our dataset regarding some of these topics. We note that although we categorize these instances to facilitate presentation here, labels indicating specific health conditions are not included in our dataset.

B Data Refinement

After our manual data refinement process during which we removed the instances that required adjudication, we ran the benchmark model (*DistilBERT*) established in prior work (Valizadeh et al., 2021) on the final dataset versus the merged version. Table 9 presents the results of this experiment,

PHYSICAL HEALTH CONDITION
Brain & Nerves 1. *I get horrific migraines. The dizziness/feeling faint and spaced out feeling is the worst for me.* 2. *I have nerve testing next week and am going to have my thyroid retested. I also want an MRI and a brain scan. I feel hopeless.*
Chest & Lungs 1. *For the past year, I've had issues with taking in deep breaths, have had an X-Ray, CT Scan on my lungs and they couldn't find anything.* 2. *There is a mass in my right lung and I need to go for a bronchoscopy this coming Monday to determine what exactly it is.*
Bones, Joints, & Muscles 1. *Finally after a month of waiting for results, I was told I do not have Rheumatoid Arthritis. I do wish that they would take my Osteoarthritis more seriously though. I'm glad you had your test and all seemed ok.* 2. *I am nearly 2 years total left knee replacement and was a massive diet coke drinker before surgery. However, since surgery on March 16, I haven't been able to drink the stuff, makes me sick.*
Gut, Bowel, & Stomach 1. *I recently had appendicitis and got my appendix removed about a month ago.* 2. *I went to several doctors for three and a half months with these symptoms to no avail after various negative tests. I was eventually diagnosed with IBS.*

Table 7: Some of the physical health-related topics covered in our dataset.

MENTAL HEALTH CONDITION
Anxiety Disorders 1. *I have Panic Disorder and Generalized Anxiety Disorder. It is well maintained and I have not had a panic attack in years.* 2. *I start PT on Monday and I'm hoping I get a therapist that is understanding. I don't need to cry more than I already have or have anymore anxiety attacks.*
Post Traumatic Stress Disorder (PTSD) 1. *I also suffer from C-PTSD so everything is accentuated and I can't wait till this transition is over for all of us.* 2. *I did not sleep for 6 months due to PTSD maybe micro sleeps. I just laid in bed wide awake or with eyes closed.*
Depression 1. *Since my depression is genetic, it's been following me around for a while. It's always hung around me, but in the past couple of months it hit me again and I was struggling for the longest time.* 2. *I suffered from depression for many years but it is nowhere near as bad as it was because I am on ad's and had counselling. Now I can live with it and have a good life which means something to me.*
Bipolar Disorder 1. *I've only been diagnosed Bipolar just over a month ago. I am very new to it, but looking back over many years I realise I've had it for a while.* 2. *One of the meds I am taking for Bipolar is Seroquel at night. And after eliminating everything else, it seems that the Seroquel might be the cause of it.*

Table 8: Some of the mental health-related topics covered in our dataset.

showing relative percentage increases of 3.37%, 3.14%, 1.79%, and 2.41% for accuracy, precision, recall, and F_1, respectively. We also ran our highest performing model (*BERTweet*) established in Section §5.2 on the final dataset versus the merged version. Table 10 presents the results of this experiment, showing relative percentage increases of 3.16%, 1.87%, 3.47%, and 2.22% for accuracy, precision, recall, and F_1, respectively.

Dataset	Acc.	Precision	Recall	F_1
Merged	78.27	0.7593	0.7672	0.7632
Final	80.91	0.7832	0.7810	0.7816

Table 9: Model performance before and after data refinement. The model used for this experiment is the baseline, a *DistilBERT* model.

Dataset	Acc.	Precision	Recall	F_1
Merged	84.98	0.8371	0.8264	0.8351
Final	87.67	0.8528	0.8551	0.8537

Table 10: Model performance before and after data refinement. The model used for this experiment is our highest performing model, a *BERTweet* model (§5.2).

C Data Preprocessing

To validate the utility of the preprocessing techniques described in §4.1, we ran our *DistilBERT* benchmark and our highest performing model *BERTweet* on the final dataset without any preprocessing steps and with each step individually. We report our findings in Tables 11 and 12, demonstrating that each step results in small but measurable performance improvements.

Technique	Accuracy
Base Model (No Preprocessing)	77.21%
Base + DeEmojifying	78.07%
Base + Number Replacement	78.02%

Table 11: Model performance in accuracy (%) before and after applying each preprocessing technique. *Base model* refers to our baseline *DistilBERT* model, trained on the final dataset (§5.1).

Technique	Accuracy
BERTweet (No Preprocessing)	86.54%
BERTweet + DeEmojifying	87.43%
BERTweet + Number Replacement	87.32%

Table 12: Model performance in accuracy (%) before and after applying each preprocessing technique. *BERTweet* refers to our highest performing model, trained on the final dataset (§5.2).

D Experimental Settings

We optimized model hyperparameters using grid search. Table 13 shows the final hyperparameters we used when training our models.

Model	Learning Rate	Batch Size	Epochs
DistilBERT	2e-6	16	3
RoBERTa	2e-5	16	3
BioBERT	2e-5	8	4
Bio-ClinicalBERT	2e-5	8	4
Bio-RedditBERT	2e-5	16	3
MentalBERT	2e-5	32	3
BERTweet	2e-5	8	3

Table 13: Models' final hyperparameters.

E Data Augmentation

We generated augmented data for the *Possible SD* and *No SD* classes using each of the techniques described in §4.3 as a separate condition. We fine-tune the augmentation ratio, a hyperparameter indicating the multiplicative factor by which the dataset size is increased, setting it to 2 for *Possible SD* and 1.5 for *No SD*. Figure 5 demonstrates that the DA process resulted in a relatively balanced training set (3,488 *Clear SD*, 3,258 *Possible SD*, and 3,488 *No SD* instances).

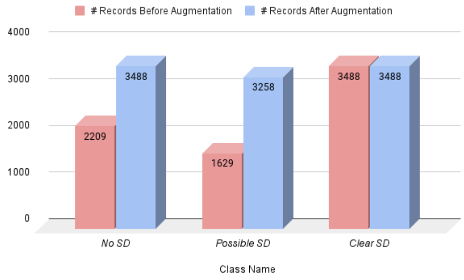


Figure 5: Distribution of training instances before and after data augmentation technique.

F Per-Class Performance for Data Augmentation Techniques

Our experiments did not provide conclusive evidence that data augmentation can be effectively leveraged for the task of detecting medical self-disclosure. This is somewhat unsurprising since the task is known to be challenging due to its reliance on often-subtle language patterns (Valizadeh et al., 2021; Reuel et al., 2022). We further examined the per-class performance of our best-performing model (*BERTweet* fine-tuned on various augmented datasets) compared to the *no augmentation* condition to develop a deeper understanding of opportunities for future improvement, and present the results in Table 14.

Dataset	**NO SD**	**POSSIBLE SD**	**CLEAR SD**
MLM-R	90.95	64.64	92.53
MLM-D	89.79	63.64	92.40
TA	89.95	68.47	89.92
WS	93.03	60.19	90.67
BT	93.72	70.94	92.27
NA	89.38	78.66	91.45

Table 14: Comparison of per-class accuracy for the BERTweet models trained on the augmented and non-augmented (NA) datasets. Accuracy is shown as a percentage (%).

G Clinical Validation

After our clinical validation process, we observed high agreement between the clinical expert and our gold standard: in total, 92.37% of the expert's labels (1353 of the 1465 instances) matched those in the gold standard. However, there were some instances for which our annotators and the clinical expert did not agree. Table 15 presents some of these instances, along with commentary provided by our clinical expert to offer additional insight into their thought process and rationale.

Instance	**Clinical Label**	**Gold Standard**	**Expert's Note**
Although the madness part of King George I might be close to. Stephen Fry is an idol of mine, and someone I watch closely for obvious link in the field of mental health but if he's Premier League I'd barely count as a part time player!	Clear SD	Possible SD	Although a specific diagnosis is not referenced, when viewed from a clinical perspective this is a clear disclosure of a mental health problem.
I don't want to die but I can't afford to live with pain of emotions that I am worth less and no one needs me!	Clear SD	Possible SD	Similar to above, clinical expertise affords additional sensitivity to mental health disclosures.
I have had a perfectly awful past year....the worst ever in my life. I have been in perimenopause for about 4-6 years now (hard to know for sure), I have been on here begging for advice on what I can do to ease my suffering and I was struck with a major realization today!	Possible SD	Clear SD	Menopause itself is not a medical problem since it is a natural circumstance; thus, focus should instead be placed on the suffering that is referenced but unspecific.
I'm also going through hormonal changes so its been rough trying to figure out whats causing what.	Possible SD	Clear SD	Again, hormonal changes are a natural circumstance rather than a healthcare problem; focus should be placed on the unspecified issues.
I'm honestly aware that it's extremely unlikely for a 16 year old to end up with colon cancer, but it does happen in rare occasions so it's hard for me to completely let go of the idea.	No SD	Possible SD	It is unclear whether this disclosure is in reference to the poster.
We have to tell our drs that we are not stupid nor irresponsible with test results. It is your body/health and you need to know and understand what is happening and not kept in the dark.	No SD	Possible SD	Advocating for rights or expertise is not a disclosure of a medical concern in itself.

Table 15: Instances with disagreements in the clinical validation process.

Improving Visual-Semantic Embedding with Adaptive Pooling and Optimization Objective

Zijian Zhang*,[1], Chang Shu*,[2,3], Ya Xiao[1], Yuan Shen[1], Di Zhu[1], Jing Xiao[2],
Youxin Chen[2], Jey Han Lau[4], Qian Zhang[3] and Zheng Lu[3]
[1] Meituan, China
[2] Ping An Technology (Shenzhen) Co., Ltd, China
[3] University of Nottingham Ningbo, China
[4] The University of Melbourne, Australia

Abstract

Visual-Semantic Embedding (VSE) aims to learn an embedding space where related visual and semantic instances are close to each other. Recent VSE models tend to design complex structures to pool visual and semantic features into fixed-length vectors and use hard triplet loss for optimization. However, we find that: (1) combining simple pooling methods is no worse than these sophisticated methods; and (2) only considering the most difficult-to-distinguish negative sample leads to slow convergence and poor Recall@K improvement. To this end, we propose an adaptive pooling strategy that allows the model to learn how to aggregate features through a combination of simple pooling methods. We also introduce a strategy to dynamically select a group of negative samples to make the optimization converge faster and perform better. Experimental results on Flickr30K and MS-COCO demonstrate that a standard VSE using our pooling and optimization strategies outperforms current state-of-the-art systems (at least 1.0% on the metrics of recall) in image-to-text and text-to-image retrieval. Source code of our experiments is available at `https://github.com/96-Zachary/vse_2ad`.

1 Introduction

Visual Semantic Embedding (VSE) (Frome et al., 2013; Faghri et al., 2018) is a representation learning method that embeds images and texts for efficient cross-modal retrieval, and typically has the following steps: (1) the image and text features are first extracted by separate visual and text encoders; (2) these features are then projected into a joint embedding space and pooled to form fixed-length vectors; (3) similarity is computed to measure the distance between image and text representation; and (4) a suitable objective is chosen for optimization (see Figure 1 for an illustration). Our paper focuses on improving the steps of feature aggregation and optimization.

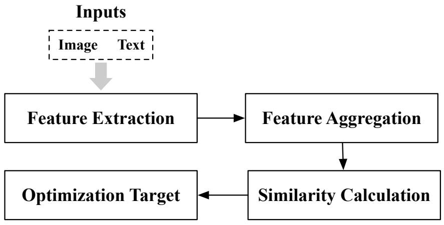


Figure 1: **Illustration of VSE**.

For feature aggregation, the most commonly used methods are simple pooling aggregators. MaxPool (Wang et al., 2018) and MeanPool (Reimers and Gurevych, 2019) are designed to detect the salient and mean points of features, and K-MaxPool (Kalchbrenner et al., 2014) extracts the mean of top-K features. Some complex aggregation techniques have been proposed, e.g. local-importance projection (Gao et al., 2019), sequence-to-sequence encoder (Hu et al., 2019), graph convolution network (Li et al., 2019), exponential adaptive pooling (Stergiou and Poppe, 2021) and self-attention encoder (Wang et al., 2020). However, we found that carefully selected pooling functions can surpass complex methods (see Appendix A.1). Motivated by this, our paper proposes an approach that can *automatically* discover the best pooling functions. Specifically, we seek to improve the feature aggregation step by proposing a formulation that parameterizes the different pooling strategies and allows the model to learn the best configuration automatically via its objective, alleviating the need to do manual tuning. In other words, we've turned these hyper-parameters (i.e. choices of pooling functions) into parameters in the model.

For optimization, most VSE models optimize using the hinge triplet ranking loss with in-batch negative example (Faghri et al., 2018). The intuition of the objective is to encourage positive pairs to be embedded in a similar space while widening the

---

[*] These authors contributed equally to this work.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1217–1229
May 2-6, 2023

distance between a target with the hardest in-batch negative sample (Li et al., 2022a). In practice, however, it is often difficult for the model to find a good negative sample in the early stages of training (as instances are randomly distributed in space), resulting in slow convergence (see Appendix A.2). To improve optimization, we propose an adaptive optimization objective that selects multiple in-batch negative samples based on model quality during training. The intuition is that in the early stages of training, we want to sample more negative samples for faster convergence, and in the later stages, we want to sample fewer negative samples for finer tuning.

Over two public datasets, MS-COCO (Lin et al., 2014) and Flickr30K (Young et al., 2014), we show that a standard VSE model using our proposed feature aggregation and optimization strategies outperforms benchmark models substantially. In particular, our method obtains 1.4% relative gains on RSUM for MS-COCO and 1.0% for Flickr30K. Compared with a pre-trained vision-language model with similar performance (Geigle et al., 2022), our method is $4.3\times$ faster.

2 Related Work

Depending on whether the image and text features have any form of *cross-modal interaction* before similarity calculation, existing image-text retrieval can be broadly categorized into two types.

The visual semantic embedding (VSE) (Faghri et al., 2018; Wang et al., 2020; Chun et al., 2021) methods process the multimodal instances *independently* before projecting them into a joint embedding space for similarity matching. Wang et al. (2018) design a two-branch neural networks, LIWE (Wehrmann et al., 2019) considers character-based alignment and embedding methods for language encoder and Faghri et al. (2018) extend it by using hard triplet loss for optimization. Following these ideas, PVSE (Song and Soleymani, 2019) and CVSE (Wang et al., 2020) are proposed to consider intra-modal polysemous and consensus information. Recently, Chun et al. (2021) samples instances as probabilistic distributions and achieves further improvement. These VSE-based methods are fast as they do not consider cross-modal interaction and as such the visual and text features can be pre-computed. The non-VSE methods concentrate on the interaction of modalities. For example, prior methods explore fusing multi-modal information by attention (Lee et al., 2018; Chen et al., 2020; Zhang et al., 2021a), alignment (Zhang et al., 2020), multi-view representation (Qu et al., 2020) or fine-grained reasoning (Qu et al., 2021; Zhang et al., 2022). The early-interaction methods (Geigle et al., 2022), like pre-trained vision-language models (Lu et al., 2019; Chen et al., 2019; Li et al., 2020; Jia et al., 2021; Li et al., 2022b), focus on maximizing performance while trading off efficiency.

The performance of VSE ultimately depends on the quality of the joint embedding space, which is usually learned with simple transformations (e.g. linear projection or multi-layer perceptron) and pooling aggregators (e.g. mean pooling (Faghri et al., 2018; Qu et al., 2020), max pooling (Zhang and Lu, 2018; Li et al., 2021), or a combination of them (Lee et al., 2018)). Compared to these simple aggregation methods, more complex aggregators that introduce a large number of trainable parameters have also been explored, e.g. inter-modal attention (Wehrmann et al., 2020) and self-attention mechanisms (Han et al., 2021). Zhang et al. (2021b) design a cross-modal guided pooling module that attends to local information dynamically. These sophisticated aggregators typically require more time, and don't always outperform simple pooling strategies. Perhaps the closest study to our work is GPO (VSE∞) (Chen et al., 2021), which builds a generalized operator to learn the best pooling strategy that only considers the position information of the extracted features.

Some studies focus on improving the optimization objective, and the most widely adopted objective is the hinge-based hard triplet ranking loss (Faghri et al., 2018; Wei et al., 2020b; Messina et al., 2021), which dynamically selects the "hardest" negative sample within a mini-batch. Other studies explore solutions that choose multiple negative samples. Zhou et al. (2020) introduce a coherence metric to rank the "irrelevant" candidates. Extending the idea, Wei et al. (2020a) assign different weights for positive and negative pairs. To tackle the issue of noisy labels which impacts multimodal representation, Hu et al. (2021) propose maximizing the mutual information between different modalities. Huang et al. (2021) separate data into "clean" and "noisy" partitions by co-teaching. Ultimate though these methods did not explore modifying the behaviour of negative sample selection *adaptively* according to model performance during training, which we address in this work.

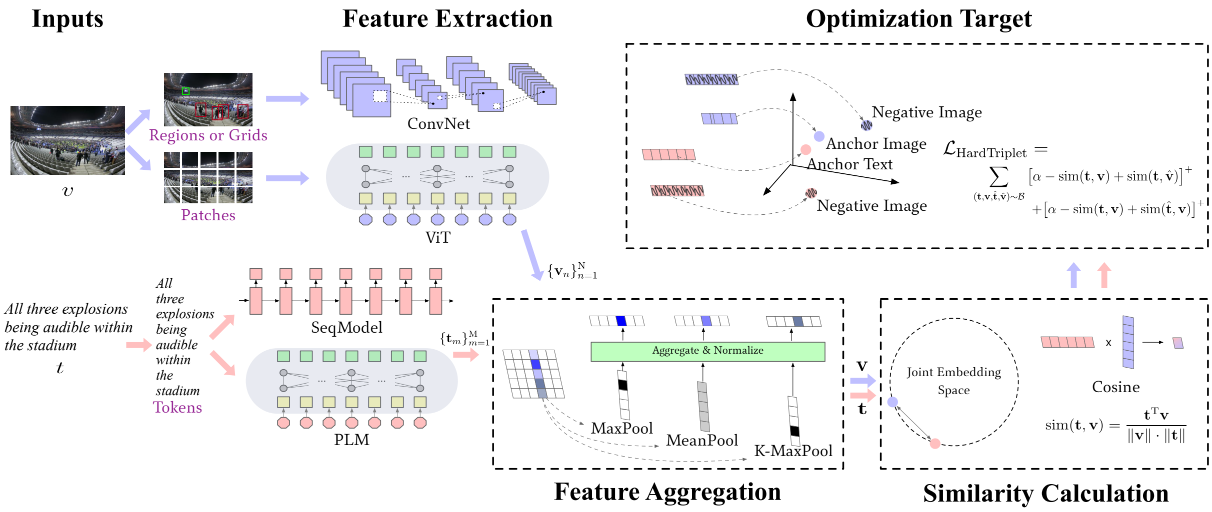


Figure 2: **The framework of VSE.** The visual and text encoders process the image and text separately at first. The related images and sentences are then directed to a similar space using an appropriate optimization target.

3 Methodology

3.1 Background of VSE

We first discuss the standard formulation of VSE, before introducing our innovation on improving feature aggregation (Section 3.2) and optimization (Section 3.3).

To compute the similarity of given multimodal instance (image & text), a VSE model (Figure 2) separately encodes them via a visual encoder (`VisEnc`$(\cdot)$) and a text encoder (`TextEnc`$(\cdot)$). There are three widely used visual features produced by different visual encoders — *grid* is the feature maps from convolutional networks (CNNs; He et al. (2016)), *region* is the region of interest features from object detectors (Anderson et al., 2018) and *patch* is the partition from vision transformer (Dosovitskiy et al., 2021). The text encoders are usually RNNs (Sutskever et al., 2014)) and BERT (Devlin et al., 2019). Formally:

$$\mathbf{F}_v = \texttt{VisEnc}(v)$$
$$\mathbf{F}_t = \texttt{TextEnc}(t)$$

where v and t are the input image and text.

Assuming the visual feature $\mathbf{F}_v$ has N object vectors (represented either as grids, regions or patches) in d_1 dimension, and the text feature $\mathbf{F}_t$ has M token vectors in d_2 dimension, we next project them to the same d-dimension:

$$\begin{aligned}\{\mathbf{v}_n\}_{n=1}^{\text{N}} &= \mathbf{F}_v\mathbf{W}_v + \mathbf{b}_v \\ \{\mathbf{t}_m\}_{m=1}^{\text{M}} &= \mathbf{F}_t\mathbf{W}_t + \mathbf{b}_t\end{aligned} \tag{1}$$

where $\mathbf{v}_n$ and $\mathbf{t}_m$ now have the same dimension d.

To aggregate the extracted features into fixed-length vectors, domain aggregators, $f_{\text{vision}}(\cdot)$ and $f_{\text{text}}(\cdot)$ are used to transform $\{\mathbf{v}_n\}_{n=1}^{\text{N}} \in \mathbb{R}^{\text{N}\times d}$ and $\{\mathbf{t}_n\}_{m=1}^{\text{M}} \in \mathbb{R}^{\text{M}\times d}$ into $\mathbf{v} \in \mathbb{R}^d$ and $\mathbf{t} \in \mathbb{R}^d$, respectively:

$$\mathbf{v} = f_{\text{vision}}\Big(\{\mathbf{v}_n\}_{n=1}^{\text{N}}\Big),$$
$$\mathbf{t} = f_{\text{text}}\Big(\{\mathbf{t}_m\}_{m=1}^{\text{M}}\Big)$$

And lastly, to measure how related the inputs we use cosine similarity:

$$\text{sim}(\mathbf{t},\mathbf{v}) = \frac{\mathbf{t}^{\text{T}}\mathbf{v}}{\|\mathbf{t}\| \cdot \|\mathbf{v}\|}$$

Existing optimization strategies generally use the hinge-based triplet ranking loss to optimize the VSE model. Given an anchor, it aims to maximize its similarity with positive samples while minimizing its similarity with the most "difficult" negative sample in the mini-batch (i.e. the example that has the highest similarity with the anchor that is not a positive example), and includes both text-to-image and image-to-text retrieval objectives:

$$\begin{aligned}\mathcal{L}_{\text{HardTriplet}} &= \\ \sum_{(\mathbf{t},\mathbf{v},\hat{\mathbf{t}},\hat{\mathbf{v}})\sim\mathcal{B}} &\big[\alpha - \text{sim}(\mathbf{t},\mathbf{v}) + \text{sim}(\mathbf{t},\hat{\mathbf{v}})\big]^+ \\ &+ \big[\alpha - \text{sim}(\mathbf{t},\mathbf{v}) + \text{sim}(\hat{\mathbf{t}},\mathbf{v})\big]^+\end{aligned} \tag{2}$$

where α is the margin hyper-parameter, and $[x]^+ = \max(0, x)$. $(\mathbf{t}, \mathbf{v})$ is a positive text-image pair in mini-batch $\mathcal{B}$ and $(\hat{\mathbf{t}}, \mathbf{v})$ and $(\mathbf{t}, \hat{\mathbf{v}})$ are negative pairs, where $\hat{\mathbf{t}} = \text{argmax}_{\mathbf{t}'\neq\mathbf{t}}\text{sim}(\mathbf{t}', \mathbf{v})$ and $\hat{\mathbf{v}} = \text{argmax}_{\mathbf{v}'\neq\mathbf{v}}\text{sim}(\mathbf{t}, \mathbf{v}')$ are the hardest negative sentence and image respectively in $\mathcal{B}$.

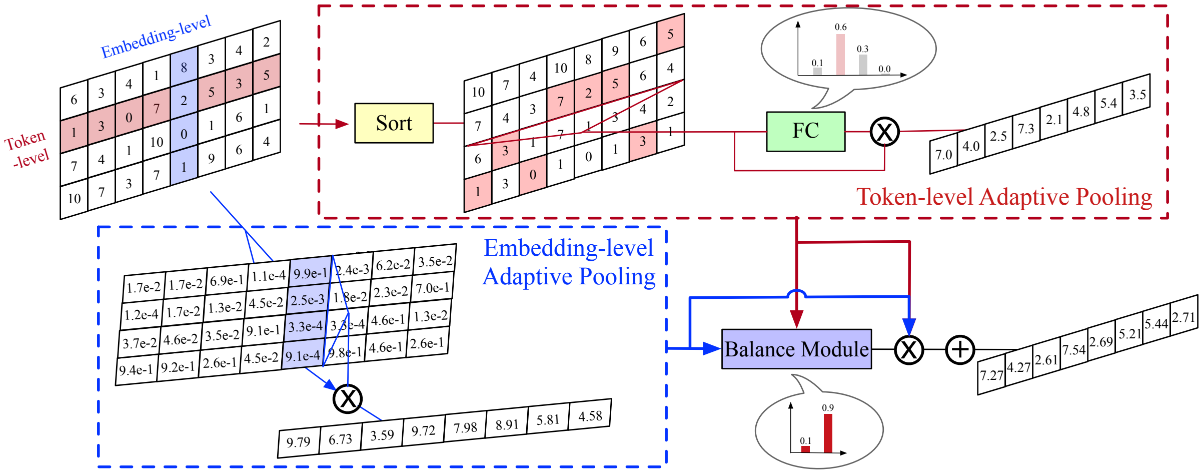


Figure 3: **Illustration of our proposed adaptive pooling module (ADPOOL).** It produces vectors pooled at the Token-level and Embedding-level and combined via a balance module. Red row denotes a token-level vector.

3.2 Enhanced VSE with Adaptive Pooling

We now describe our approach (named adaptive pooling, ADPOOL) to improving feature aggregation, where we parameterize the pooling operations (f_{vision} and f_{text}) to allow VSE to learn the best way to aggregate the features via its objective adaptively (in other words, we have effectively turned the pooling methods — which are hyperparameters — into model parameters). Our approach has the advantage of being much faster and simpler than complex aggregators, as it introduces only a small number of parameters. Note that we will describe our method only for the text feature ($\{\mathbf{t}_m\}_{m=1}^{\mathrm{M}}$), as the same process can be applied for the visual feature ($\{\mathbf{v}_n\}_{n=1}^{\mathrm{N}}$).

3.2.1 Token-level Pooling

Recall that the text feature ($\{\mathbf{t}_m\}_{m=1}^{\mathrm{M}}$) has M tokens each of d-dimension. The exact form of $\mathbf{t}_m$ can be token vectors. As for vision features, they can be grids, regions or image patches, depending on the visual encoder. Let $\max_k(\cdot)$ be a function that extracts the top k-th value in a list, the common pooling mechanisms can be formulated as follows:

- **MeanPool** = $\frac{1}{\mathrm{M}}\sum_{i=1}^{\mathrm{M}}\{t_{ij}\}_{i=1}^{\mathrm{M}}$ ouputs the mean value among N objects at position j;
- **MaxPool** = $\max_1\left(\{t_{ij}\}_{i=1}^{\mathrm{M}}\right)$ returns the maximum values of $\{t_{ij}\}_{i=1}^{\mathrm{M}}, \forall j$;
- **K-MaxPool** = $\frac{1}{\mathrm{K}}\sum_{k=1}^{\mathrm{K}}\max_k\left(\{t_{ij}\}_{i=1}^{\mathrm{M}}\right)$ computes the mean of the top-K values.

where t_{ij} denotes the j-th element in the i-th token vector.

Using the token vectors $\{\mathbf{t}_m\}_{m=1}^{\mathrm{M}}$ in Figure 3 (red rows in the input matrix) as an example, we follow the *"sort → weight-sum"* paradigm (Grefenstette et al., 2014) of simple pooling methods to first sort the feature matrix along the embedding axis. Next we let the model learn (via a fully-connected layer) how to weight each vector (red rows in the matrix) automatically (e.g. $[0.1, 0.6, 0.3, 0.0]$ in Figure 3). Intuitively, this means we have now expressed MeanPool and K-MaxPool using a generalized function (through the *"sort → weight-sum"* process), and the parameters of the function is learned automatically so the feature aggregation step is "adaptive". Formally:

$$\begin{aligned}\mathbf{t}^{\text{tok}} &= \sum_{m=1}^{\mathrm{M}} \theta_m \cdot \mathbf{u}_m \\ \{\mathbf{u}_m\}_{m=1}^{\mathrm{M}} &= \operatorname{sort}\left(\{\mathbf{t}_m\}_{m=1}^{\mathrm{M}}\right) \qquad (3)\\ \boldsymbol{\theta} &= \operatorname{softmax}\left(\{\mathbf{u}_m\}_{m=1}^{\mathrm{M}}\mathbf{W}_{\text{tok}}\right)\end{aligned}$$

where $\operatorname{sort}(\cdot)$ is a function that sorts the token vectors along the embedding axis.

3.2.2 Embedding-level Pooling

With token-level pooling, we learn to how to weight each (sorted) token vector and aggregate them. With embedding-pooling, we learn how to weight each (original unsorted) object vector *in each embedding dimension* to extract the most salient element within that dimension, and can be interpreted as a "soft" version of MaxPool:

$$\begin{aligned}\mathbf{t}^{\text{emb}} &= \sum_{i=1}^{\mathrm{M}} \delta_{ij} \cdot t_{ij}, \forall j \qquad (4)\\ \delta_{ij} &= \frac{e^{t_{ij}}}{\sum_{i=1}^{\mathrm{M}} e^{t_{ij}}}, \forall i\end{aligned}$$

3.2.3 Combining Token-level and Embedding-level Pooled Vectors

Given the two pooled vectors $\mathbf{t}^{\text{tok}}$ and $\mathbf{t}^{\text{emb}}$ (which have the same dimension d), one straightforward way to combine them is to weight them with scaling hyper-parameters. To avoid these hyper-parameters (which requires manual tuning), we let the model learn these weights automatically with a trainable $\mathbf{W}_{bal} \in \mathbb{R}^{d\times 1}$:

$$\begin{aligned} \mathbf{t} &= \omega_1 \mathbf{t}^{\text{tok}} + \omega_2 \mathbf{t}^{\text{emb}} \\ \omega_{\{\mathbf{1},\mathbf{2}\}} &= \text{softmax}\Big([\mathbf{t}^{\text{tok}}\mathbf{W}_{bal}, \mathbf{t}^{\text{emb}}\mathbf{W}_{bal}]\Big) \end{aligned} \tag{5}$$

For the visual feature $\{\mathbf{v}_n\}_{n=1}^{\text{N}}$ (Equation 1), we follow the same process to compute the token-level (where here an "token" is a unit of image) and embedding-level pooled vectors ($\mathbf{v}^{\text{tok}}$ and $\mathbf{v}^{\text{emb}}$) and combine them to produce $\mathbf{v} \in \mathbb{R}^d$.

3.3 Enhanced VSE with Adaptive Objective

Our next contribution is in improving the optimization step. The hard triplet loss (Equation 2) is the most commonly utilized training objective for optimizing VSE. However, we find that locating a "most-difficult" negative sample is challenging in the early stages of training (Appendix A.2), resulting in delayed convergence. We introduce a novel adaptive optimization objective, ADOPT, that automatically (or adaptively) selects K (K ≥ 1) negative samples in each mini-batch $\mathcal{B}$ during training.

Wang and Isola (2020) introduce two key properties, *Alignment* and *Uniformity*, that correlate with a retrieval performance:

- *Alignment*: the positive ($\mathbf{t}$ and $\mathbf{v}$) should be mapped into a similar embedding space. Defining $\gamma_{\text{align}} \in [0, 1]$ as the average similarity values for all positive pairs, the larger γ_{align} is, the better the VSE model.

$$\gamma_{\text{align}} = \mathop{\mathbb{E}}_{(\mathbf{t},\mathbf{v})\sim\mathcal{B}}[\text{sim}(\mathbf{t}, \mathbf{v})]$$

- *Uniformity*: all vectors should be roughly uniformly distributed on the unit hypersphere, and $\gamma_{\text{uniform}} \in [0, 1]$ measures this quality.

$$\gamma_{\text{uniform}} = \log \mathop{\mathbb{E}}_{\mathbf{t}\sim\mathcal{B},\mathbf{v}\sim\mathcal{B}}[e^{\text{sim}(\mathbf{t},\mathbf{v})}]$$

By combining $\frac{(\gamma_{\text{align}}+\gamma_{\text{uniform}})\times\pi}{4} \in [0, \frac{\pi}{2}]$, we can use it to assess the maturity of a VSE model during training and dynamically adjust the number of negative samples. That is, in the early stages of training, we expect the value to be close to 0 (as the model is unable to differentiate between positive and negative samples), and we would want to use more negative samples for optimization. Conversely in the later stages of training, the value should be close to $\frac{\pi}{2}$ and we want less negative samples. Formally:

$$\begin{aligned} &\text{K}' = \text{round}\Big(|\mathcal{B}| \times \cos(\frac{(\gamma_{\text{align}} + \gamma_{\text{uniform}}) \times \pi}{4})\Big) \\ &\text{K} = \max\Big(1, \min(\text{K}', |\mathcal{B}| - 1)\Big) \end{aligned} \tag{6}$$

where cos is the cosine function to invert the sum[1], K is the number of negative samples to be used, and $\text{round}(x)$ is a function that rounds down the value of x. As the hard triplet loss can only work with $\text{K} = 1$ (Equation 2), we therefore utilize the InfoNCE objective (Van den Oord et al., 2018), which is a commonly used contrastive objective (Radford et al., 2021):

$$\begin{aligned} &\mathcal{L} = \mathcal{L}_{\text{v2t}} + \mathcal{L}_{\text{t2v}} \\ &\mathcal{L}_{\text{v2t}} = -\frac{1}{|\mathcal{B}|}\sum_{(\mathbf{t},\mathbf{v})\in\mathcal{B}} \log\frac{\exp\big(\text{sim}(\mathbf{t},\mathbf{v})/\tau\big)}{\sum_k^{\text{K}}\exp\big(\text{sim}(\hat{\mathbf{t}}_k,\mathbf{v})/\tau\big)} \\ &\mathcal{L}_{\text{t2v}} = -\frac{1}{|\mathcal{B}|}\sum_{(\mathbf{t},\mathbf{v})\in\mathcal{B}} \log\frac{\exp\big(\text{sim}(\mathbf{t},\mathbf{v})/\tau\big)}{\sum_k^{\text{K}}\exp\big(\text{sim}(\mathbf{t},\hat{\mathbf{v}}_k)/\tau\big)} \end{aligned}$$

where τ is the temperature hyper-parameter.

4 Experiments

4.1 Datasets and Metrics

Datasets. We conduct our experiments on MS-COCO and Flickr30K using various visual and text encoders for cross-modal retrieval. The MS-COCO dataset contains 123,287 images, each with 5 manually annotated sentences. Following the split method of Faghri et al. (2018), 113,287 images are used for training, 5,000 for validation, and 5,000 for testing.

Following prior studies (Faghri et al., 2018), we experiment with two evaluation settings: (1) MS-COCO 1K averages the results over 5-folds of 1K test images; and (2) MS-COCO 5K directly results on the whole 5K test images. Following Chen et al. (2021), we use the former to assess overall performance with state-of-the-art VSE models and the

---

[1] The purpose of using cos is to map $\frac{(\gamma_{\text{align}}+\gamma_{\text{uniform}})\times\pi}{4} \in [0, \frac{\pi}{2}]$ to the range of $[1, 0]$.

		MS-COCO 5-fold 1K Test							Flickr30K 1K Test						
		Image Retrieval			Caption Retrieval				Image Retrieval			Caption Retrieval			
VF	Method	R@1	R@5	R@10	R@1	R@5	R@10	RSUM	R@1	R@5	R@10	R@1	R@5	R@10	RSUM
	Text Encoder: BiGRU														
R	VSE++	54.0	85.6	92.7	68.5	92.6	97.1	490.5	45.7	73.6	81.9	62.2	86.6	92.3	442.3
R	LIWE	57.9	88.3	94.5	73.2	95.5	98.2	507.6	51.2	80.4	87.2	69.6	90.3	95.6	474.3
R	PVSE	55.2	86.5	93.7	69.2	91.6	96.6	492.8	-	-	-	-	-	-	-
R	CVSE	55.7	86.9	93.8	69.2	93.3	97.5	496.4	54.7	82.2	88.6	70.5	88.0	92.7	476.7
R	VSE∞	61.7	90.3	95.6	78.5	96.0	98.7	520.5	56.4	83.4	89.9	76.5	94.2	97.7	498.1
R	SCAN(i2t)	54.4	86.0	93.6	69.2	93.2	97.5	493.9	43.9	74.2	82.8	67.9	89.0	94.4	452.2
R	SCAN(t2i)	56.4	87.0	93.9	70.9	94.5	97.8	500.5	45.8	74.4	83.0	61.8	87.5	93.7	446.2
R	CAAN	61.3	89.7	95.2	75.5	95.4	98.5	515.6	52.8	79.0	87.9	70.1	91.6	97.2	478.6
R	IMRAM	61.7	89.1	95.0	76.7	95.6	98.5	516.6	53.9	79.4	87.2	74.1	93.0	96.6	484.2
R	VSE+2AD	**63.5**	**91.8**	**96.3**	**79.7**	**97.3**	**99.2**	**527.8**	**58.0**	**85.0**	**91.2**	**76.9**	**94.4**	**98.2**	**503.7**
RG	VSE∞	64.8	91.6	96.5	80.0	97.0	99.0	528.8	60.8	86.3	92.3	80.7	96.4	98.3	514.8
RG	VSE+2AD	**65.7**	**92.3**	**97.0**	**82.1**	**97.9**	**99.4**	**534.4**	**62.2**	**86.8**	**93.1**	**82.2**	**97.1**	**98.8**	**520.2**
RGP	VSE+2AD	**67.1**	**93.0**	**97.7**	**83.8**	**98.1**	**99.4**	**539.1**	**63.5**	**87.6**	**93.4**	**83.1**	**97.7**	**99.1**	**524.4**
	Text Encoder: BERT														
R	VSE++	54.0	85.6	92.5	67.9	91.9	97.0	488.9	45.6	76.4	84.4	63.4	87.2	92.7	449.7
R	VSE∞	64.8	91.4	96.3	79.7	96.4	98.9	527.5	61.4	85.9	91.5	81.7	95.4	97.6	513.5
R	DSRN	64.5	90.8	95.8	78.3	95.7	98.4	523.5	59.2	86.0	91.9	77.8	95.1	97.6	507.6
R	DIME(i2t)	63.0	90.5	96.2	77.9	95.9	98.3	521.8	64.6	85.5	91.0	77.5	93.5	97.4	504.0
R	DIME(t2i)	62.3	90.2	95.8	77.2	95.5	98.5	519.5	60.1	85.5	91.8	77.4	95.0	97.4	507.2
R	VSE+2AD	**67.5**	**93.6**	**97.7**	**81.3**	**96.7**	**99.2**	**536.0**	**59.1**	**90.3**	**93.5**	**81.8**	**96.1**	**98.4**	**524.7**
RG	VSE∞	68.1	92.9	97.2	82.2	97.5	**99.5**	537.4	66.7	89.9	94.0	85.3	97.2	98.9	532.0
RG	VSE+2AD	**71.9**	**94.3**	**98.3**	**84.2**	**98.5**	99.4	**546.6**	**69.2**	**91.3**	**95.6**	**87.1**	**97.9**	**99.3**	**540.4**
RGP	VSE+2AD	**72.5**	**94.8**	**98.7**	**85.4**	**98.9**	**99.2**	**549.5**	**71.4**	**92.0**	**95.8**	**88.2**	**98.4**	**99.5**	**545.3**

Table 1: **Cross-modal retrieval results** on MS-COCO and Flickr30K datasets. The top half of the table uses BiGRU as the text encoder; the bottom half is BERT. VF denotes vision feature, and R, G, and P mean region, grid and patch respectively. The best results are marked bold in **black**, **blue** and **red** for region feature (R), ensemble of region+grid features (RG) and ensemble of region+grid+patch features (RGP) respectively. VSE+2AD is our proposed model, which enhances the VSE model by using ADPOOL for aggregation and ADOPT for optimization.

latter for further analyses such as ablation results. Flickr30K consists of 31,783 images with the same corresponding 5 captions, and 1,000 images are reserved for validation and testing.

Metrics. We evaluate cross-modal retrieval performance using recall@K (R@K), where $K = \{1, 5, 10\}$. The evaluation tasks include both caption retrieval (given caption, find images) and image retrieval (given image, find captions). We also compute RSUM which is a sum of all R@K metrics across both tasks to assess the overall performance.

4.2 Implementation Details

We implement our models using the PyTorch library. The dimension of the shared embedding space d is 1024. We use the Adam optimizer with a mini-batch size of 128 and train our models with 25 epochs. The learning rate is set to 5e-4 with a decaying factor of 10% for every 15 epochs.

Visual Encoders. We use Faster-RCNN (Ren et al., 2015) (ResNet-101 is pre-trained on ImageNet and Visual Genome) to extract *region* feature directly (Anderson et al., 2018), and fine-tune it further with *grid* feature (resolution = 512×512) (Jiang et al., 2020) before using it as a grid feature extractor. For the *patch* feature, we fine-tune the pre-trained Vision Transformer [2] (Dosovitskiy et al., 2021) using images with a resolution of 224×224 to use it as a patch feature extractor.

Text Encoders. We experiment with BiGRU (Faghri et al., 2018) and BERT-base[3] (Devlin et al., 2019). Additional implementation and training details are given in the Appendix B.

4.3 Main Results

We compare against several mainstream baselines with our proposed method, all of which have been described in Section 2. For these benchmark models, we use the results reported in the original papers (test set is the same).

Table 1 presents the full results over both image and caption retrieval and across two datasets (MS-

---

[2] vit-base-patch16-224

[3] bert-base-uncased

COCO and Flickr30K). Results for MS-COCO is an average of over 5-folds of 1K test images. Our method is "VSE+2AD", which enhances the standard VSE model by introducing AdPool and AdOpt. The top half of the table presents results where we use BiGRU as the text encoder, and the bottom half uses BERT. For models where we combine visual features from multiple visual encoders (e.g. "RG" which combines region and grid feature), we do so by simply taking the average similarity values to rank the candidates.

Looking at the results, VSE+2AD (our model) outperforms almost all baselines/benchmark models consistently. Our model displays consistent improvement over the state-of-the-art method VSE∞ (Chen et al., 2021) with the same visual (region feature by BUTD (Anderson et al., 2018)) and text encoders (BiGRU). In particular, with BiGRU as the text encoder, it obtains 1.4% (520.5 → 527.8) and 1.0% (498.1 → 503.1) relative gains on RSUM for MS-COCO and Flickr30K datasets respectively. These improvements are consistent irrespective of the combination of visual and text encoders. We also see that combining visual encoders ("RG" vs. "R") further boosts its performance (like RSUM from 527.8 to 534.4 for MS-COCO), and utilizing all types of visual features ("RGP") produces the best performance (539.1 > 534.4 > 527.8).

4.4 Comparison with Pre-trained Models

We next compare VSE+2AD (with BERT as the text encoder) to pre-trained vision language models: ViLBERT (Lu et al., 2019), UNITER (Chen et al., 2019), OSCAR (Li et al., 2020), ALIGN (Jia et al., 2021), CLIP (Radford et al., 2021) and MVP (Li et al., 2022b) in Table 2. These results are evaluated on the COCO 5K test images.

Without using any large-scale paired image-text corpus for pre-training, our ensemble VSE+2AD$^{\mathrm{RGP}}$ (that combines region/grid/patch features) outperforms two out of six pre-trained VL methods and is not much worse than OSCAR, even though it does not use any cross-modal interaction. Our model is substantially faster than these pre-trained models: our slowest ensemble model is still an order of magnitude faster. As for our method we can pre-compute and cache the visual and text features, so the retrieval only need similarity calculation and ranking operations. Overall, these results demonstrate that our model strikes a good balance between performance and efficiency.

	COCO 5K Test					
	Image Retrieval		Caption Retrieval			
Method	R@1	R@5	R@1	R@5	RSUM	#OIPs
ViLBERT	38.6	68.2	53.5	79.7	406.9	-
UNITER	48.4	76.7	63.3	87.0	454.4	9.6×
OSCAR	54.0	80.8	70.0	91.1	479.9	4.3×
ALIGN	59.9	83.3	77.0	93.5	500.4	1.0×
CLIP	58.7	83.6	76.8	94.0	500.9	-
MVP	60.1	84.0	77.3	93.6	502.6	-
VSE+2AD	52.5	80.2	69.5	91.2	475.9	45.3×

Table 2: Comparison between VSE+2AD and pre-trained models. #OIPs denotes operating items per second, and larger is better. The text encoder for VSE+2AD is BERT. And the shown number is the ensemble results considering *region+grid+patch* visual features.

	COCO 5K Test						
	Image Retrieval			Caption Retrieval			
Method	R@1	R@5	R@10	R@1	R@5	R@10	RSUM
VSE+2AD	41.4	72.3	83.8	54.9	83.0	91.5	426.9
−*tok* AdPool	39.5	71.3	82.3	53.5	82.2	91.3	420.1
−*emb* AdPool	40.1	72.3	83.2	53.7	82.5	91.3	423.1
−AdPool	38.5	72.4	81.8	53.3	82.0	91.1	419.1
−AdOpt	39.2	71.1	81.5	53.4	82.2	90.5	417.9

Table 3: Ablation results where we measure the *token-level*, *embedding-level*, overall AdPool pooling methods and AdOpt optimization strategy. The *region* feature with simple projection as visual encoder and BiGRU as the textual encoder.

4.5 Ablation Study

To understand the impact of AdPool (which improves feature aggregation in Section 3.2) and AdOpt (which improves optimization in Section 3.3), we perform several ablation studies based on the COCO 5K test. For this experiment (in Table 3), we use only the *region* feature and BiGRU as the text encoder. Looking at the aggregate RSUM performance, we see both the token-level pooling ("−*tok* AdPool"; Section 3.2.1) and embedding-level pooling ("−*emb* AdPool"; Section 3.2.2) appear to be useful, although token-level pooling is arguably more important. In the case where we remove AdPool entirely ("−AdPool") and use MeanPool as the aggregation method, the performance drops even more, suggesting complementarity. AdOpt is the most impactful method, where taking it out produces the worst performance. To understand its impact qualitatively, we also look at the training curve of a VSE model trained with and without AdOpt. We found that that AdOpt is particularly useful in the early stages of train-

	COCO 5K Test					
	Image Retrieval		Caption Retrieval			
Aggregator	R@1	R@5	R@1	R@5	RSUM	#OIPs
LIP	38.4	69.3	52.2	81.1	414.8	4.6×
Seq2Seq	38.2	68.9	52.1	80.9	415.7	1.8×
GCN	40.7	71.3	53.5	81.5	419.2	1.0×
SelfAttn	40.9	71.3	53.8	82.2	420.1	1.6×
AdaPool	40.2	71.8	52.8	81.4	418.3	4.1×
SoftPool	39.2	69.8	52.8	81.4	419.8	5.1×
GPO	41.2	71.1	**55.6**	82.0	422.9	4.3×
Manual	38.7	71.6	53.7	82.1	419.3	**5.6**×
ADPOOL	**41.4**	**72.3**	54.9	**83.0**	**426.9**	4.9×

Table 4: Performance of VSE+2AD using different aggregators. #OIPs denotes operating items per second, and larger is better. Note that our proposed ADPOOL aggregation method is only slower than simple pooling, 5.6× > 4.9 ×(ADPOOL) > others, but it no need for fussily manual tuning.

ing, where it helps the model to converge much faster. This earlier convergence ultimately impacts their final performance, where the VSE trained with ADOPT reaches a plateau that is higher than the one without ADOPT. For more details, please see Appendix A.2.

We next investigate the impact of ADPOOL further, by replacing it with other more advanced pooling strategies. As before, we use the region feature, BiGRU for the text encoder, and COCO 5K test. Table 4 presents the results. Here we see that ADPOOL outperforms existing pooling strategies (best RSUM), even against more complex aggregators such as Seq2Seq, GCN and SelfAttn. It is also reasonably fast (competitive with other methods). These results once again highlight that our proposed pooling method has both performance and speed as we saw in Section 4.4.

4.6 Case Study

To validate the effectiveness of VSE+2AD, we present two examples for image retrieval and caption retrieval in Figure 4(a) and Figure 4(b), respectively. As we can see in Figure 4(a), the "incorrect" sentence retrieved by our model seems sensible, suggesting that this is likely noise in the data. Figure 4(b), on the other hand, shows some genuine erroneous images retrieved by our model, and we suspect this is because it is a particularly difficult example where the caption is very descriptive and the details are difficult to be captured by VSE's bi-encoder approach.

5 Conclusion and Discussion

In this paper, we propose methods to improve VSE's feature aggregation and optimization objective. For the former, we introduce a way to parameterize the aggregation function to allow the visual and text encoders to learn the best way to combine their features to produce a fixed-size embedding. For the latter, we propose a method that dynamically selects many negative samples that allow the VSE to converge faster with a better final performance. We compare our enhanced VSE model to several baselines and state-of-the-art models over two public datasets and demonstrate that it marries both performance (state-of-the-art retrieval results) and efficiency (orders of magnitude faster than pretrained models). In terms of practical impact, we also tested a real-world application of our method in Dianping (a Chinese searching platform) for a unimodal retrieval task (semantic textual similarity) and found that it worked very well, demonstrating its applicability to real-world applications.

Top-1 A man with gauges and glasses is wearing a Blitz hat. ✓
Top-2 The man with pierced ears is wearing glasses and an orange hat. ✓
Top-3 A man wears an orange hat and glasses. ✓
Top-4 A man with glasses is wearing a beer can crocheted hat. ✓
Top-5 A man in an orange hat starring at something. ✓

Top-1 A Boston Terrier is running on lush green grass in front of a white fence. ✓
Top-2 A dog runs on the green grass near a wooden fence. ✓
Top-3 A Boston terrier is running in the grass. ✓
Top-4 A dog running though a grassy field. ✘
Top-5 A black and white dog is running in a grassy garden surrounded by a white fence. ✓

Five people wearing winter jackets and helmets stand in the snow , with snowmobiles in the background.

A young female student performing a downward kick to break a board held by her Karate instructor.

(a) Image Retrieval

(b) Caption Retrieval

Figure 4: Top-5 image retrieval and caption retrieval results on Flickr30K. The ground-truth results are marked with greed ✓, and the wrong results are indicated by red ✘.

Limitations

Our method's performance has only been verified on English datasets; other languages can be explored to understand whether these results translate to them. In a similar vein, we've also only tested our methods in domain-general datasets; cross-modal retrieval is a particularly important task in the general intelligence domain, and with limited paired image-text datasets it would be interesting to see whether our proposed enhancements work with these specialised domains. Lastly, our idea of parameterizing the pooling and optimization strategies can technically be applied to tasks other than cross-modal retrieval, and it's something worth exploring in future works.

Acknowledgements

We are grateful for the support of Biao Tang and Gong Zhang from the platform search group in Meituan-Dianping for testing the practical utility of our method. This work was supported by Ningbo Science and Technology Bureau under Major Projects Fund with project code 2021Z089 and 2022Z060.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Hui Chen, Guiguang Ding, Xudong Liu, Zijia Lin, Ji Liu, and Jungong Han. 2020. Imram: Iterative matching with recurrent attention memory for cross-modal image-text retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12655–12663.

Jiacheng Chen, Hexiang Hu, Hao Wu, Yuning Jiang, and Changhu Wang. 2021. Learning the best pooling strategy for visual semantic embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15789–15798.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.

Sanghyuk Chun, Seong Joon Oh, Rafael Sampaio De Rezende, Yannis Kalantidis, and Diane Larlus. 2021. Probabilistic embeddings for cross-modal retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8415–8424.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*.

Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. 2018. Vse++: Improving visual-semantic embeddings with hard negatives. *Proceedings of the British Machine Vision Conference (BMVC)*.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. *Advances in neural information processing systems*, 26.

Ziteng Gao, Limin Wang, and Gangshan Wu. 2019. Lip: Local importance-based pooling. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3354–3363.

Gregor Geigle, Jonas Pfeiffer, Nils Reimers, Ivan Vulić, and Iryna Gurevych. 2022. Retrieve Fast, Rerank Smart: Cooperative and Joint Approaches for Improved Cross-Modal Retrieval. *Transactions of the Association for Computational Linguistics*, 10:503–521.

Edward Grefenstette, Phil Blunsom, et al. 2014. A convolutional neural network for modelling sentences. In *The 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland*.

Ning Han, Jingjing Chen, Guangyi Xiao, Hao Zhang, Yawen Zeng, and Hao Chen. 2021. Fine-grained cross-modal alignment network for text-video retrieval. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3826–3834.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hexiang Hu, Ishan Misra, and Laurens van der Maaten. 2019. Evaluating text-to-image matching using binary image selection (bison). In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0.

Peng Hu, Xi Peng, Hongyuan Zhu, Liangli Zhen, and Jie Lin. 2021. Learning cross-modal retrieval with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5403–5413.

Zhenyu Huang, Guocheng Niu, Xiao Liu, Wenbiao Ding, Xinyan Xiao, Hua Wu, and Xi Peng. 2021. Learning with noisy correspondence for cross-modal matching. *Advances in Neural Information Processing Systems*, 34.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*.

Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. 2020. In defense of grid features for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10267–10276.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.

Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. 2018. Stacked cross attention for image-text matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 201–216.

Jiangfeng Li, Zijian Zhang, Bowen Wang, Qinpei Zhao, and Chenxi Zhang. 2022a. Inter-and intra-modal contrastive hybrid learning framework for multimodal abstractive summarization. *Entropy*, 24(6):764.

Jiangtong Li, Liu Liu, Li Niu, and Liqing Zhang. 2021. Memorize, associate and match: Embedding enhancement via fine-grained alignment for image-text retrieval. *IEEE Transactions on Image Processing*, 30:9193–9207.

Kunpeng Li, Yulun Zhang, Kai Li, Yuanyuan Li, and Yun Fu. 2019. Visual semantic reasoning for image-text matching. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 4654–4662.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Zejun Li, Zhihao Fan, Huaixiao Tou, and Zhongyu Wei. 2022b. Mvp: Multi-stage vision-language pre-training via multi-level semantic alignment. *arXiv preprint arXiv:2201.12596*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Nicola Messina, Fabrizio Falchi, Andrea Esuli, and Giuseppe Amato. 2021. Transformer reasoning network for image-text matching and retrieval. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5222–5229. IEEE.

Leigang Qu, Meng Liu, Da Cao, Liqiang Nie, and Qi Tian. 2020. Context-aware multi-view summarization network for image-text matching. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1047–1055.

Leigang Qu, Meng Liu, Jianlong Wu, Zan Gao, and Liqiang Nie. 2021. Dynamic modality interaction modeling for image-text retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1104–1113.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Yale Song and Mohammad Soleymani. 2019. Polysemous visual-semantic embedding for cross-modal retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1979–1988.

Alexandros Stergiou and Ronald Poppe. 2021. Adapool: Exponential adaptive pooling for information-retaining downsampling. *arXiv preprint*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807.

Haoran Wang, Ying Zhang, Zhong Ji, Yanwei Pang, and Lin Ma. 2020. Consensus-aware visual-semantic embedding for image-text matching. In *European Conference on Computer Vision*, pages 18–34. Springer.

Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. 2018. Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):394–407.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Jonatas Wehrmann, Camila Kolling, and Rodrigo C Barros. 2020. Adaptive cross-modal embeddings for image-text alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12313–12320.

Jonatas Wehrmann, Douglas M Souza, Mauricio A Lopes, and Rodrigo C Barros. 2019. Language-agnostic visual-semantic embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5804–5813.

Jiwei Wei, Xing Xu, Yang Yang, Yanli Ji, Zheng Wang, and Heng Tao Shen. 2020a. Universal weighting metric learning for cross-modal matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13005–13014.

Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. 2020b. Multi-modality cross attention network for image and sentence matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10941–10950.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Chenxi Zhang, Zijian Zhang, Jiangfeng Li, Qin Liu, and Hongming Zhu. 2021a. Ctnr: Compress-then-reconstruct approach for multimodal abstractive summarization. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Gangjian Zhang, Shikui Wei, Huaxin Pang, and Yao Zhao. 2021b. Heterogeneous feature fusion and cross-modal alignment for composed image retrieval. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5353–5362.

Qi Zhang, Zhen Lei, Zhaoxiang Zhang, and Stan Z Li. 2020. Context-aware attention network for image-text retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3536–3545.

Ying Zhang and Huchuan Lu. 2018. Deep cross-modal projection learning for image-text matching. In *Proceedings of the European conference on computer vision (ECCV)*, pages 686–701.

Zijian Zhang, Chang Shu, Youxin Chen, Jing Xiao, Qian Zhang, and Lu Zheng. 2022. Icaf: Iterative contrastive alignment framework for multimodal abstractive summarization. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Mo Zhou, Zhenxing Niu, Le Wang, Zhanning Gao, Qilin Zhang, and Gang Hua. 2020. Ladder loss for coherent visual-semantic embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13050–13057.

Appendix

A Verification of Assumptions

A.1 Effectiveness of Simple Pooling Methods

Although various complex methods (described in Section 1 and Section 2) are explored for aggregation, we find that the simple pooling strategy works no worse than those complex methods by numerous experiments. It needs to be carefully manually tuned, like 5-MaxPool for the visual feature (*region* feature from BUTD) and MeanPool for the text feature (extracted by BiGRU). The results are shown in Table A.1-1, where the similar conclusion is also verified by VSE∞ (Chen et al., 2021).

	COCO 5K Test				
	Image Retrieval		Caption Retrieval		
Aggregator	R@1	R@5	R@1	R@5	RSUM
LIP	38.4	69.3	52.2	81.1	414.8
Seq2Seq	38.2	68.9	52.1	80.9	415.7
GCN	40.7	71.3	53.5	81.5	419.2
SelfAttn	40.9	71.3	53.8	82.2	420.1
AdaPool	40.2	71.8	52.8	81.4	418.3
SoftPool	39.2	69.8	52.8	81.4	419.8
GPO	41.2	71.1	**55.6**	82.0	**422.9**
Manual	38.7	**71.6**	53.7	**82.1**	419.3

Table 1: Comparison of complex aggregators and manually chosen pooling method evaluated with MS-COCO 5K. The best results are marked **bold**.

A.2 Slow Convergence of Hard Triplet Loss

Figure A.2-1 shows the comparison of VSE model with and without our proposed optimization objective, ADOPT. Note that hard triplet loss (Faghri et al., 2018) is used for optimization when ADOPT is not used. The optimization target ADOPT can fit the model more quickly, thus further improving the potential of the model, that is performance is better in the latter stage.

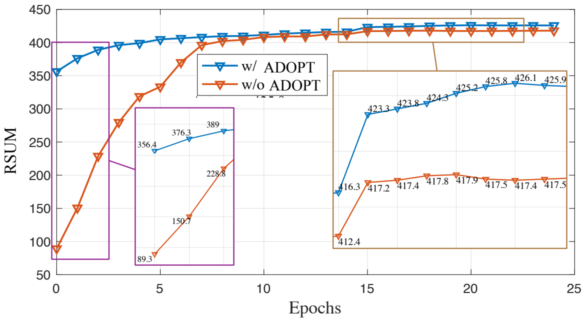


Figure 1: Performance of VSE trained with and without ADOPT.

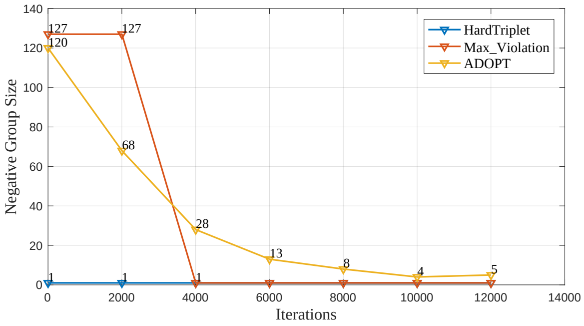


Figure 2: Visualization of negative group size learned by ADOPT on the first 5 epochs (about 12000 iterations).

B Additional Implementation Details

Basic settings. The margin α for hard triplet loss is set to 0.2 and the τ used in InfoNCE is 0.05. The image and text features extracted from the encoders use L2 normalization. The common learning rate is set to 5e-4, while the learning rate of the pre-trained modules (like BERT, ResNet and ViT) is 10% of its.

Vision Encoders. When using *region* feature that is directly extracted from BUTD (Anderson et al., 2018), the multilayer perceptron is used to map the visual feature dimension into 1024 as the same as the text feature dimension. For the *grid* feature, the warm-up strategy is used for the first epoch. Then, all parameters are optimized in the rest of the 24 epochs. For the *patch* feature, the original image is changed to 224×224 resolution with the size of a patch as 16.

Text Encoders. When using BiGRU as the backbone, the token dimension is 300 and the hidden dimension is 1024. Only one layer is considered and the bidirectional features are averaged as the output feature. For BERT, the hidden dimension is 768 and the multilayer perceptron is also used to map the text feature dimension into 1024 as the same as the visual feature dimension.

C Additional Experiments and Results

We present Figure C-2 to show how the size of the negative samples changes during training with ADOPT. We can see that ADOPT starts with a large number of negative samples, but that decreases over time to only 4–5 samples at the end of the 5^{th} epoch (yellow line). Different from the common hard triplet optimization that only considers one hardest negative sample, the max_violation strategy (Faghri et al., 2018) considers the rest of samples

		COCO 5K Test				
Weights		Image Retrieval		Caption Retrieval		
obj	*emb*	R@1	R@5	R@1	R@5	RSUM
0.25	0.75	39.8	70.2	51.9	80.4	416.6
0.5	0.5	40.1	70.6	52.8	81.1	418.2
0.75	0.25	40.6	71.4	54.1	82.2	421.3
random	random	39.3	69.4	51.2	80.1	413.9
ADPOOL	ADPOOL	41.4	72.3	54.9	83.0	426.9

Table 2: Different choices of parameters for fusing token-level and embedding-pooling.

within the same mini-batch as negative samples only on the first epoch and the rest of epochs are the same as the hard triplet loss.

We last investigate the impact of the balance module in ADPOOL (Equation 5). Table C-2 shows that manually tuned weights underperform substantially compared to automatically learned weights[4].

---

[4] For "random", we select the weights randomly and run them 5 times and compute the average to reduce variance.

Policy-based Reinforcement Learning for Generalisation in Interactive Text-based Environments

Edan Toledo[*] [1,4] **Jan Buys** [2] **Jonathan Shock** [3,5,6]
[1] Department of Computer Science and Technology, University of Cambridge
[2] Department of Computer Science, University of Cape Town
[3] Department of Mathematics and Applied Mathematics, University of Cape Town
[4] InstaDeep [5] INRS, Montreal, Canada [6] NiTHeCS, South Africa
`et498@cam.ac.uk`, `jbuys@cs.uct.ac.za`,
`jonathan.shock@uct.ac.za`

Abstract

Text-based environments enable RL agents to learn to converse and perform interactive tasks through natural language. However, previous RL approaches applied to text-based environments show poor performance when evaluated on unseen games. This paper investigates the improvement of generalisation performance through the simple switch from a value-based update method to a policy-based one, within text-based environments. We show that by replacing commonly used value-based methods with REINFORCE with baseline, a far more general agent is produced. The policy-based agent is evaluated on Coin Collector and Question Answering with interactive text (QAit), two text-based environments designed to test zero-shot performance. We see substantial improvements on a variety of zero-shot evaluation experiments, including tripling accuracy on various QAit benchmark configurations. The results indicate that policy-based RL has significantly better generalisation capabilities than value-based methods within such text-based environments, suggesting that RL agents could be applied to more complex natural language environments.

1 Introduction

General domain language comprehension is an important component of the goal to create Artificial Intelligence agents that can perform real-world activities (Gil and Selman, 2019). In the pursuit of such systems, classic text-based games are often utilised as a bridge to the generality of the real world (Yuan et al., 2018; Ammanabrolu and Riedl, 2019a).

Text-based games are a useful test-bed for exploring interactive dialogue agents in the context of partially observable Markov decision processes (POMDPs) that have large combinatorial action spaces. Additionally, they provide environments that depend on the comprehension and use of natural language. In recent years, there has been a shift in focus from simply performing well in a single text-based environment (Haroush et al., 2018; Narasimhan et al., 2015b) to generalising in unseen environments (Xu et al., 2021; Adhikari et al., 2020; Yuan et al., 2018, 2019). This shift in focus has often relied on complex methods and bespoke solutions (Ammanabrolu and Riedl, 2019b,a; Yin et al., 2020; Adolphs and Hofmann, 2019). A great deal of this research has utilised value-based reinforcement learning (RL) methods which often show high training performance but poor zero-shot generalisation. These methods have been shown to overfit and memorise the training environments, thereby not learning a generalisable policy (Farebrother et al., 2018a).

Previous work has shown that policy-based methods can work well in non text-based settings where generalization is required (Cobbe et al., 2020). However, to the best of our knowledge no controlled comparison has been made between the generalisability of value-based and policy-based RL methods in text-based environments. This leads us to investigate policy-based RL in text-based environments. We hypothesise that learning an accurate and general value function in a partially observable environment with a large action space (such as natural language) is too difficult for most value-based methods. We aim to show that policy-based approaches solve this generalisation issue by directly learning a stochastic and general policy.

In this paper, we use two text-based environments, Coin Collector (Yuan et al., 2018) and QAit (Yuan et al., 2019). Both environments utilise TextWorld (Côté et al., 2018) to generate environments on the fly. Agents must acquire natural language skills in order to navigate successfully within the worlds and achieve some specified goal. We compare a policy-based method, trained using the

---

[*] Work done whilst at InstaDeep and the University of Cape Town

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1230–1242
May 2-6, 2023 ©2023 Association for Computational Linguistics

REINFORCE with baseline algorithm (Williams, 1992) against Coin Collector and QAit's value-based approaches on the environments' respective test sets. For comparison, we make use of both Coin Collector and QAit's open-sourced codebases with a simple replacement of the value-based update and action selection method.

Our results show that the policy-based method leads to substantially improved performance over previous value-based methods on a variety of zero-shot evaluations. In the Coin Collector task, we see greater average performance and stability on the easy and medium test sets (after minimal hyperparameter tuning). On the QAit benchmark, large accuracy improvements are obtained for location questions (28% to 98% for fixed maps) and existence questions (69.2% to 94.8% for fixed maps). The results indicate that policy-based RL has significantly better generalisation capabilities than value-based methods within these contexts, and are more suited to text-based environments.

This paper's main contributions are as follows:

1. We show that policy-based methods have better and more stable zero-shot performance on two text-based environments.

2. We show that large improvements can be made through a simple replacement of the update method without significant change to any deep learning architectures.

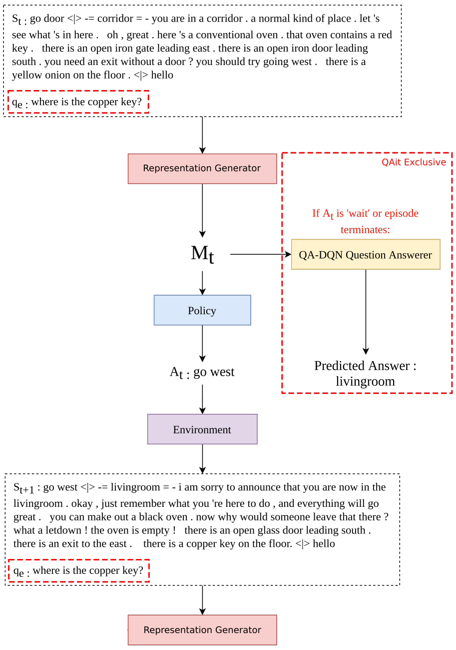
3. We provide new results for the QAit and Coin Collector environments against which further research can be compared.

2 Background

2.1 Policies and Value Functions

In text-based environments, agents are presented with a descriptive paragraph describing their current state (Figure 1). Agents interact with the environment using natural language commands. These commands could be long descriptive sentences such as *"go to the kitchen and pick up the copper key"* or a short templated sequence such as *"go kitchen"* and *"pickup copper key"*. The environment interprets these commands and responds with a new paragraph or sentence describing the changed state or event.

An agent in a text-based environment uses a policy to predict a command string as a sequence of words at every time step t. A policy π is defined as a mapping of environment states or observations $\mathcal{S}$ to the probabilities of selecting each possible word in the command string $a \in \mathcal{A}$ (Sutton and Barto, 2018). If an agent is following a policy π at time step t, the policy function $\pi(a|s)$ is the probability of performing the specific action $A_t = a$ given that the agent is in state $S_t = s$. The agent's high-level policy for command strings represents the joint policy of each sub-policy used for each word.



S_t : go door <|> -= corridor = - you are in a corridor . a normal kind of place . let 's see what 's in here . oh , great . here 's a conventional oven . that oven contains a red key . there is an open iron gate leading east . there is an open iron door leading south . you need an exit without a door ? you should try going west . there is a yellow onion on the floor . <|> hello

q_e : where is the copper key?

S_{t+1} : go west <|> -= livingroom = - i am sorry to announce that you are now in the livingroom . okay , just remember what you 're here to do , and everything will go great . you can make out a black oven . now why would someone leave that there ? what a letdown ! the oven is empty ! there is an open glass door leading south . there is an exit to the east . there is a copper key on the floor. <|> hello

q_e : where is the copper key?

Figure 1: Overview of agent-environment interaction. The agent receives a textual observation from the environment, encoded in a latent representation, which is used to compute the policy and value functions. In QAit this representation is also used for the QA module.

Widely-used RL methods such as Q-learning (Watkins and Dayan, 1992) and Actor-Critic algorithms (Konda and Tsitsiklis, 1999) estimate the state-action value (or Q-value) function within an environment. This function maps a state-action pair to the expected cumulative future reward under a given policy. The expected cumulative reward G_t is defined as

$$G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k, \qquad (1)$$

where T is the terminal time step, R_k is the reward at time step k, and γ is the discount factor (the weight of importance given to future rewards). Formally, the Q-value function for any given policy is defined for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$ as

$$Q_\pi(s, a) = \mathbb{E}[G_t | S_t = s, A_t = a], \qquad (2)$$

while the state value function $V_\pi(s)$ is $\max\limits_a Q_\pi(s, a)$.

2.2 Value-Based Methods

In value-based methods the agent tries to learn $V(s)$ (or $Q(s, a)$), which tells the agent what states it should be in (and which actions it should take) in order to maximise G_t. The agent can use $Q(s, a)$ to select the action which will take it to the most valuable next state by choosing $\text{argmax}_a\, Q_\pi(s, a)$. This is called the greedy policy and is commonly used to implement control in value-based methods (Sutton and Barto, 2018).

2.3 Policy-based Methods

In policy-based methods the agent tries to learn the policy directly. In contrast to value-based methods, this allows the agent to learn stochastic policies (Sutton and Barto, 2018).

REINFORCE (Williams, 1992) is a Monte Carlo method that updates the policy function's parameters (neural network weights) directly by moving them in the direction which will increase the expected future returns (the network's loss function). However, REINFORCE suffers from high variance with noisy gradient estimates and no clear credit assignment to positive or negative actions throughout the episode (Sutton and Barto, 2018). A simple improvement is to reduce the variance of the empirical returns G_t by subtracting a baseline function $b(s)$ in the policy gradient. The baseline is regarded as a proxy for the true expected return. A popular option for the baseline function is the state value function $V(S_t)$. This requires the agent to learn the value function alongside the policy, which can introduce a bias at the cost of lowering variance. REINFORCE uses Monte Carlo episodic sample returns G_t (refer to §2.1) to update the baseline and calculate the advantage factor with which to update the policy.

2.4 Generalisation in Text-based Games

Much research has gone into applying reinforcement learning to text-based games (see Osborne et al. (2022) for a survey), predominantly using value-based update methods. These value-based agents have been shown to often overfit on the training environment and have poor out-of-domain performance (Yuan et al., 2018, 2019). This is supported by evidence showing that DQN suffers from severe overfitting on training environments for Atari games (Farebrother et al., 2018b).

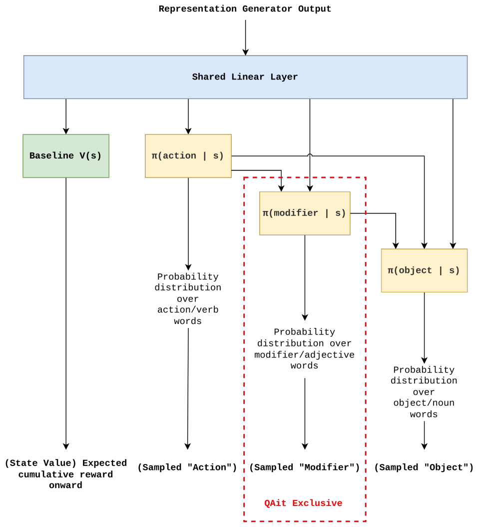


Figure 2: Overview of REINFORCE with baseline *action scorer* architecture replacement. The policy and value functions all share a linear layer that takes in the encoded text observation as input. Both Coin Collector and QAit experiments make use of this architecture for interaction, with the exception that Coin Collector's policy outputs two words whereas QAit's policy outputs three words.

Research on improving agents' generalisation performance in text-based games has mainly focused on adding components such as knowledge graphs, as well as improving value-based learning algorithms (Adhikari et al., 2020; Yuan et al., 2018; Ammanabrolu and Riedl, 2019b,a; Yin et al., 2020; Yuan et al., 2019).

Adolphs and Hofmann (2019) showed that employing an actor-critic learning method, A2C (Mnih et al., 2016), significantly increased generalisation performance compared to value-based baselines such as LSTM-DQN (Narasimhan et al., 2015a) and DRRN (He et al., 2015). However, their results do not provide a direct comparison and their agent employs an additional helper model for command generation. This helper model is domain-specific and does not allow one to fully infer the performance difference from a simple switch to a policy-based learning method. There has thus been no research directly showing that policy-based methods improve generalisation over value-based ones when applied to text-based tasks.

3 Policy-based Text-based Agents

We propose a policy-based agent that makes only minor changes to previous value-based approaches. For both environments, the neural network architecture is kept largely unmodified, in order to focus on comparing the generalisation performance of the value-based baselines and the policy-based method (REINFORCE with baseline). The only changes are the use of a policy-based update method, an additional MLP to predict state values (to use as $b(s)$ - the baseline function), and the conditioning of future command words on previous command words (shown in Figure 2). For most value-based methods, the Q-values of actions are learnt. This means that the MLP used to predict Q-values already has the necessary number of output nodes for a policy-based method.

Each output layer represents the policy for each word in the command tuple, e.g. (Action, Modifier, Object) for QAit and (Verb, Noun) for Coin Collector. Each output gives a probability distribution over the vocabulary. Each word in the command tuple is then sampled from these probability distributions to form the command at each game step. This more closely emulates how a human would speak compared to value-based methods, which traditionally act using greedy action selection at evaluation time (which is deterministic) and epsilon-greedy action selection at training time (which introduces some stochasticity) (Sutton and Barto, 2018).

The baseline function approximates the state-value function to aid in the training of the policy. The baseline network also uses the policy's shared linear layer as input to produce the state value $V(s)$. This is done to regularise the shared linear layer to a common representation to aid the policy and value function in generalisation (Silver et al., 2017). The REINFORCE with baseline algorithm is used to update model weights. Each policy's entropy (for each word in the command tuple) is subtracted from the total loss. This incentivises more stochastic policies to be learnt whilst maximising reward, in order to promote exploration and generalisation. The loss functions used for the policies $\mathcal{L}_a$ and baseline $\mathcal{L}_c$ are:

$$\mathcal{L}_a = \frac{1}{T}\sum_{t=0}^{T}(G_t - V(s_t)) * \log \pi(a_t) \tag{3}$$

$$\mathcal{L}_c = \frac{1}{2T}\sum_{t=0}^{T}(G_t - V(s_t))^2 \tag{4}$$

The command word conditioning is done as follows: Each consecutive word in the command tuple receives the vector of raw outputs from the previous word, concatenated with the shared linear output. A small ablation of the effects of command conditioning indicated only a slight performance increase. Conditioning on consecutive words did not lead to any performance gains on the value-based baselines.

4 Coin Collector

Coin Collector (Yuan et al., 2018) is a text-based deterministic version of the chain experiment (Osband et al., 2016; Plappert et al., 2017) that introduces off-chain nodes (distractor rooms) to distract and confuse the agent.

4.1 Environment

An agent needs to learn how to navigate a through a path of rooms (nodes) to the final goal location whereby it must collect a “coin". In the optimal case, an agent never revisits distractor rooms. Every game starts off with the agent at one end of the *chain* and the *coin* at the other end. Coin Collector games have a finite horizon, thereby requiring the agent to collect the coin with a limited number of mistakes deviating from the optimal trajectory.

Coin Collector has two parameters that control environment difficulty. There are 3 possible modes (*easy*, *medium*, *hard*) for constructing a game, with zero, one, and two *distractors* along the optimal path, respectively. The agent needs to learn that if it makes a mistake and enters a distractor room, it needs to go back the way it came and continue in a different direction. Each game also has a difficulty **level** which indicates the length of the optimal path.

The Coin Collector environment has two reward types. An agent is given an exploration reward each time it enters a previously unseen *physical location*. Additionally, it receives a terminal reward of 1 if it collects the coin.

4.2 Interaction

Interaction in the environment requires a very limited vocabulary. Dividing action selection into verb and noun, the action space is: {*go*, *take*} × {*north*, *south*, *east*, *west*, *coin*}. The only valid commands are: *go north*, *go east*, *go south*, *go west* and *take coin*.

4.3 Evaluation

Each mode (easy, medium, hard) has 5 distinct test sets of increasing difficulty levels: 5, 10, 15, 20, 30. Each test set contains 10 unseen games. We evaluate agents on all test sets during training to

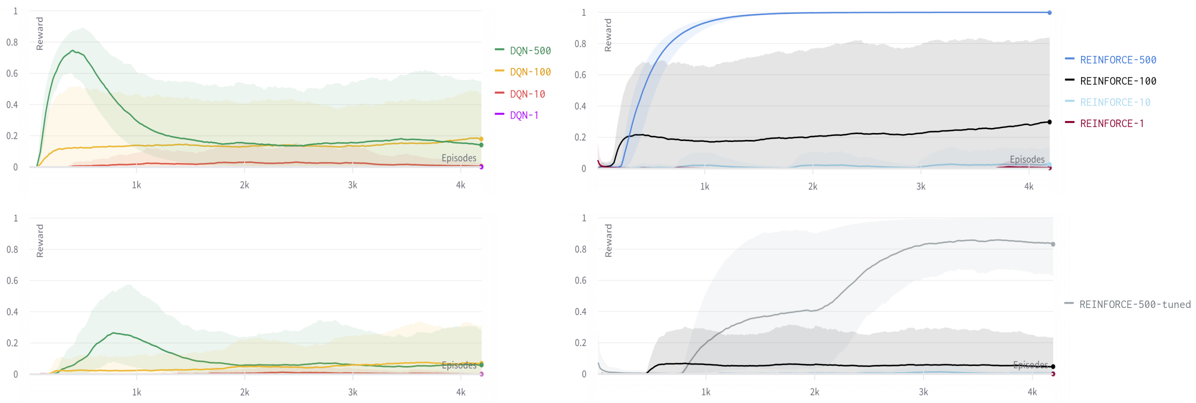


Figure 3: Coin Collector easy (top row) and medium (bottom row) test set results, averaged over all training runs. Shaded regions indicates max and min performance achieved.

see the performance and stability of generalisation as an agent learns how to solve its training games. By evaluating on higher-level games than the agent has trained on, we can see to what extent it is truly able to generalise and extrapolate its policy beyond what it has seen. The evaluation is done on agents trained on 1, 10, 100, and 500 unique level 10 games in their respective modes. Each agent is trained for 20 000 episodes.

4.4 Architecture

The original Coin Collector architecture (Yuan et al., 2018) has two modules: The representation generator and the action scorer. The representation generator is used to encode textual observations and produce an input for the action scorer. It consists of an embedding layer, an LSTM (Hochreiter and Schmidhuber, 1997), and a mean pooling layer. The action scorer is used to produce Q-values for each possible action in the command sequence (§4.2). It consists of two MLPs that share a lower layer. This model does not condition on previous actions or observations, which can present an issue due to the partially observable nature of the environment. To mitigate this issue, the previous command is concatenated to the observation. Yuan et al. (2018) implemented two value-based RL methods, LSTM-DQN (Narasimhan et al., 2015a) and LSTM-DRQN (Hausknecht and Stone, 2015; Lample and Chaplot, 2017), with this architecture. The only difference between the two is that LSTM-DRQN uses a recurrent action scorer module which takes in the hidden state output of the previous timestep. Our policy-based agent makes use of this architecture with changes illustrated in section 3.

4.5 Results

Figure 3 shows the results of our Coin Collector experiments, comparing the LSTM-DQN to our policy-based agent trained with REINFORCE. The curves indicate the average performance over all test sets for each agent when trained on 1, 10, 100, and 500 games. Both LSTM-DQN and REINFORCE failed to perform adequately when trained on 1 or 10 games.

Yuan et al. (2018) showed that LSTM-DQN performs better on the medium setting and is comparable to LSTM-DRQN on the easy setting, so in this paper we only compare to LSTM-DQN, which also simplifies the update method replacement. Experiments in the hard mode setting were not run as we were unable to reproduce the reported results using the original code base.

For the test sets in the easy setting, our policy-based agent matches or outperforms DQN on every level. We see that here the REINFORCE agent manages to achieve a similar maximum performance when training on only 100 games compared to LSTM-DQN training on 500 games. Additionally, the policy-based method is much more stable and does not suffer from catastrophic losses in performance due to overfitting the training data. This suggests that the policies learn not to simply memorise the training set as seen in the LSTM-DQN agent, which loses generalisation ability as training continues.

In the medium setting, the performance improvement is less apparent than the easy setting. The LSTM-DQN architecture performs better overall than the REINFORCE agent, largely due to the policy-based method failing to learn anything in the 500 games setting. Due to the policy-based

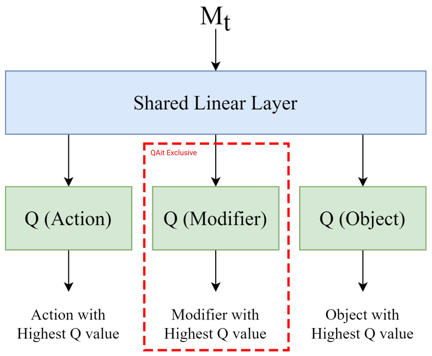


Figure 4: Overview of DQN network (action scorer) architecture.

method simply being a substitute for the value-based update, no hyperparameters were changed. This was done intentionally to show that all that is needed to generalise better is to replace the update method. By changing the discount factor from 0.5 (as is specified by Yuan et al. (2018)) to 0.9 the REINFORCE agent trained on 500 games achieves significantly higher performance than LSTM-DQN on all the medium test sets.

5 QAit

The Question Answering with interactive text (QAit) (Yuan et al., 2019) task is a text-based question answering problem in which an agent must interact with a partially observable text-based environment to gather the declarative knowledge required to answer questions. QAit aims to test an agent's language comprehension abilities by posing questions about the location, existence and attributes of objects distributed throughout the environment.

5.1 Environment

We use the implementation of QAit built on top of TextWorld to create text environments and associated questions dynamically. All environments are generated procedurally according to two environment categories (fixed map and random map). The fixed maps always have 6 unique rooms, while for random map the number of rooms is drawn from a uniform distribution between 2 and 12. For both maps types the number of entities is sampled from a uniform distribution between 3 and 6 times the number of rooms in the map. Room connectivity and structure is also changed from map to map.

Questions based on each environment are created on the fly as an agent plays a game. Yuan et al. (2019) trained agents on datasets consisting of 1, 2, 10, 100, and 500 created environments, as well as an unlimited setting where different environments are sampled for each question. In the unlimited setting more than 10^{40} different games can be created, making it unlikely that the agent will see the same environment multiple times.

The QAit environment has two reward types. An agent is given an exploration reward each time it enters a previously unseen *physical location* and/or new *inventory status*. In addition, a sufficient information reward is given at the end of the episode. These rewards are explained in §5.3.

There are three types of questions that are posed to the agent: **Location** questions ask the whereabouts of objects situated within the world, e.g. "Where is the can of soda?", where a suitable answer would be "fridge". **Existence** questions ask about the presence of objects situated within the world, e.g. "is there a raw egg in the world?", where the answer would simply be yes or no. **Attribute** questions ask whether or not an object has a certain associated attribute, e.g. "is apple edible", where the answer is also yes or no. These questions are the most challenging, as the agent has to both find and interact with the object, e.g. find the apple, pick it up, and try to eat it.

5.2 Interaction

All text commands are triplets of the form action, modifier, object (e.g., "open metallic gate"). When there is no ambiguity, the environment understands commands without modifiers, e.g. "pick key" will result in picking up the "copper key" provided it is the only key in the room. At each game step, there are separate lexicons for actions, modifiers and objects. An episode of experience terminates when a maximum number of steps is reached or the *wait* command is issued by the agent, indicating that it wants to answer the question. For our experiments, we use a maximum of 80 steps. There are, on average, 17 actions, 18 modifiers, and 27 objects per game, and 93.1 and 89.7 observed tokens in fixed and random maps games, respectively.[1]

5.3 Evaluation

The QAit test set provides 500 hold-out games for each of the question types and for both map types. This test set is used to benchmark the generalisation abilities of agents in all experimental

---

[1] Statistics calculated over 10,000 sampled games.

configurations. This allows for models to be assessed in a reproducible manner and is analogous to supervised learning test sets. The evaluation metrics used are accuracy and sufficient information. Each agent is trained for 200 000 episodes. **Accuracy** refers to the proportion of correctly answered questions. The distribution of answers in the QAit evaluation set is presented in the Appendix (Tables 6, 7 and 8).

Sufficient Information is used to evaluate the amount of information gathered by the agent and whether or not the information was sufficient to answer the question (Yuan et al., 2019). It is also used as part of the reward function. It measures the performance of the navigation and interaction required by the agent to answer a given question. For each question type, the sufficient information score is calculated as follows:

- **Location**: A score of 1 is given if the entity mentioned in the question is present in the final observation when the agent decides to stop the interaction, indicating that the agent has observed the information it needs to answer the question successfully. Otherwise, a score of 0 is given.
- **Existence**: If the answer to the question is ***yes***, a score of 1 is given if the entity mentioned in the question is present in the final observation. If the answer is ***no***, a score between 0 and 1 is given representing the proportion of the environment that the agent has explored, accounting for the fact that an agent can only be sure that an entity does not exist if it has explored the whole environment.
- **Attribute**: Attribute questions have a set of heuristics defined to verify each attribute and assign a score of sufficient information. Each attribute has specific commands that need to be executed or certain states the agent needs to be in for sufficient information to be gathered.

5.4 Architecture

As in the Coin Collector architecture, the original QAit architecture (Yuan et al., 2019) consists of a representation generator and an action scorer. An additional module is used to answer the question (the component exclusive to QAit in figure 1). The representation generator is a transformer encoder (Vaswani et al., 2017) consisting of an embedding layer, two stacks of transformer blocks (one for encoding and the other for aggregation), and a final attention layer. The action scorer has a shared linear layer followed by MLPs for each word in the command sequence (Figure 4). The question answerer appends an additional stack of aggregation transformer blocks to compute the answer from the encoder output. At each game step, the question representation is merged with the representation of the current game observation to produce the final state representation, so that the agent cannot forget the goal. The QAit task (Yuan et al., 2019) provides three value-based RL methods using this architecture as baselines: DQN (Mnih et al., 2013), DDQN (Van Hasselt et al., 2016) and Rainbow (Hessel et al., 2018). As with Coin Collector, our policy-based agent uses this architecture with the changes presented in §3.

5.5 Results

Table 1 gives the QAit test set results for all experiments as well as the baseline models' performance as reported by Yuan et al. (2019). Tables 3 and 4 in the Appendix give the full results including training performance. The policy-based method outperforms all the value-based methods on location and existence questions in each of the number of games settings on both map types, in many instances by a large margin. In many settings the increase in sufficient information score is even larger than the increase in question answering accuracy.

Most notable is the large increase in test accuracy of the policy-based method in larger numbers of training games. The value-based methods' performances increase only slightly with the number of games, compared to the large jumps in the policy-based method, indicating much better generalisation ability. In the unlimited games setting the policy-based agent reaches 98% and 90.2% accuracy on location and existence questions respectively on the fixed map setting. This is compared to the best value-based agent's accuracies of 28% and 69.2%. On existence questions, the increase in sufficient information score from 0.246 to 0.781 is even larger, suggesting a dramatic improvement in navigation and interaction. On random maps the improvement is also large, albeit not as stark as in fixed maps. We believe that this is due to the more difficult nature of random map-type games having potentially double the number of rooms, thereby making exploration more difficult and giving rise to more possible entities.

	Fixed			Random		
Model	**Location**	**Existence**	**Attribute**	**Location**	**Existence**	**Attribute**
Random	0.027	0.497	0.496	0.034	0.5	0.499
10 games						
DQN	0.180 (0.188)	0.568 (0.156)	0.518 (0.030)	0.156 (0.160)	0.566 (0.142)	0.518 (0.036)
DDQN	0.188 (0.208)	0.566 (0.128)	0.516 (0.036)	0.142 (0.154)	0.606 (0.153)	0.500 (0.033)
Rainbow	0.156 (0.170)	0.590 (0.131)	**0.520 (0.023)**	0.144 (0.170)	0.586 (0.128)	0.530 (0.018)
REINFORCE	**0.230 (0.244)**	**0.654 (0.357)**	0.498 (0.049)	**0.266 (0.282)**	**0.656 (0.317)**	**0.534 (0.038)**
100 games						
DQN	0.194 (0.206)	0.614 (0.160)	0.498 (0.014)	0.184 (0.204)	0.668 (0.181)	0.524 (0.017)
DDQN	0.168 (0.196)	0.650 (0.216)	0.528 (0.017)	0.188 (0.204)	0.662 (0.205)	0.544 (0.019)
Rainbow	0.156 (0.160)	0.602 (0.207)	0.524 (0.022)	0.174 (0.184)	0.654 (0.190)	0.504 (0.032)
REINFORCE	**0.786 (0.802)**	**0.898 (0.768)**	**0.530 (0.043)**	**0.492 (0.508)**	**0.822 (0.471)**	**0.546 (0.055)**
500 games						
DQN	0.224 (0.244)	0.674 (0.279)	**0.534 (0.014)**	0.204 (0.216)	0.678 (0.214)	0.530 (0.017)
DDQN	0.218 (0.228)	0.626 (0.213)	0.508 (0.026)	0.222 (0.246)	0.656 (0.188)	0.486 (0.023)
Rainbow	0.190 (0.196)	0.656 (0.207)	0.496 (0.029)	0.172 (0.178)	0.678 (0.191)	0.494 (0.017)
REINFORCE	**0.948 (0.958)**	**0.948 (0.892)**	0.466 (0.045)	**0.570 (0.588)**	**0.836 (0.560)**	**0.534 (0.044)**
unlimited games						
DQN	0.216 (0.216)	0.662 (0.246)	**0.514 (0.016)**	0.188 (0.188)	0.668 (0.218)	**0.506 (0.018)**
DDQN	0.258 (0.258)	0.628 (0.134)	0.480 (0.024)	0.206 (0.206)	0.694 (0.196)	0.482 (0.017)
Rainbow	0.280 (0.280)	0.692 (0.157)	0.514 (0.014)	0.258 (0.258)	0.686 (0.193)	0.470 (0.017)
REINFORCE	**0.980 (0.980)**	**0.902 (0.781)**	0.462 (0.032)	**0.738 (0.738)**	**0.858 (0.584)**	0.502 (0.042)

Table 1: QAit test set results for fixed and random map configurations. QA accuracy is reported with sufficient information scores in brackets.

	Fixed		
Model	**Location**	**Existence**	**Attribute**
500 games			
DQN	0.430 (0.430)	0.742 (0.136)	0.700 (0.015)
DDQN	0.406 (0.406)	0.734 (0.173)	0.714 (0.021)
Rainbow	0.358 (0.358)	0.768 (0.187)	0.736 (0.032)
Policy-based	**0.990 (0.990)**	**0.964 (0.916)**	**0.748 (0.048)**

Table 2: QAit training performance for agents trained on 500 fixed games. QA accuracy is reported with sufficient information scores in brackets.

For attribute-type questions, neither the policy-based model nor the value-based methods achieve results significantly higher than a random agent in terms of QA accuracy in any of the settings. By looking at the sufficient information score, we can see that the models rarely end up in the states they should be in order to answer the question. Therefore these results are most likely due to chance.

The policy-based method achieves higher training QA accuracy and sufficient information than the value-based methods when using the same number of training episodes (Table 2). This is notable since off-policy methods (DQN, DDQN, Rainbow) are generally more sample efficient than on-policy methods (REINFORCE). The REINFORCE agent only updates weights once per episode, whereas the value-based methods update their weights every 20 steps (2 to 4 times per episode). This result suggests that learning a policy directly in the QAit environment is an easier task than approximating the complicated Q-function.

Analysis of the value-based methods' interaction in the test set shows that the methods often get the agent stuck during exploration. This is largely due to the deterministic nature of the algorithms for interaction. We hypothesise that the stochastic nature of the policy-based method affords it enough flexibility to learn more generalised policies.

6 Conclusion

This work demonstrates the advantages of using policy-based methods in textual environments. More specifically, we investigated the differences in generalisation performance of the REINFORCE with baseline algorithm compared to value-based RL baselines on two sets of text-based tasks. The results strongly suggest that policy-based RL methods are not only more suited for textual domains due to their training performance, but also possess generalisation capabilities beyond their value-based counterparts.

Limitations

As this paper is a comparison of policy-based and value-based deep RL methods and their generalisation capabilities, a large limitation is the scarcity of text-based environments available to evaluate on. To confirm the stark differences in generalisation performance, a large suite of text-based environments would be needed for training and evaluation. This is clearly hampered by the lack of such suitable environments. Additionally, more policy-based methods and value-based baselines would need to be evaluated to confirm that the performance differences are not environment- or algorithm-specific.

Acknowledgements

This work is based on research supported in part by the National Research Foundation of South Africa (Grant Number: 129850) and the South African Centre for High Performance Computing. Computations were performed using facilities provided by the University of Cape Town's ICTS High Performance Computing team: `hpc.uct.ac.za`.

References

Ashutosh Adhikari, Xingdi (Eric) Yuan, Marc-Alexandre Côté, Mikulas Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and William L. Hamilton . 2020. Learning dynamic belief graphs to generalize on text-based games. In *NeurIPS 2020*. ACM.

Leonard Adolphs and Thomas Hofmann. 2019. Ledeepchef: Deep reinforcement learning agent for families of text-based games. *CoRR*, abs/1909.01646.

Prithviraj Ammanabrolu and Mark Riedl. 2019a. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3557–3565, Minneapolis, Minnesota. Association for Computational Linguistics.

Prithviraj Ammanabrolu and Mark Riedl. 2019b. Transfer in deep reinforcement learning using knowledge graphs. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 1–10, Hong Kong. Association for Computational Linguistics.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. 2020. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR.

Marc-Alexandre Côté, Ákos Kádár, Xingdi (Eric) Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. In *Computer Games Workshop at ICML/IJCAI 2018*, pages 1–29.

Jesse Farebrother, Marlos C Machado, and Michael Bowling. 2018a. Generalization and regularization in dqn. *arXiv preprint arXiv:1810.00123*.

Jesse Farebrother, Marlos C. Machado, and Michael Bowling. 2018b. Generalization and regularization in dqn. In *NeurIPS'18 Deep Reinforcement Learning Workshop*.

Yolanda Gil and Bart Selman. 2019. A 20-year community roadmap for artificial intelligence research in the us. *arXiv preprint arXiv:1908.02624*.

Matan Haroush, Tom Zahavy, Daniel J. Mankowitz, and Shie Mannor. 2018. Learning how not to act in text-based games.

Matthew Hausknecht and Peter Stone. 2015. Deep recurrent q-learning for partially observable mdps. In *2015 aaai fall symposium series*.

Ji He, Jianshu Chen, Xiaodong He, Jianfeng Gao, Lihong Li, Li Deng, and Mari Ostendorf. 2015. Deep reinforcement learning with a natural language action space. *arXiv preprint arXiv:1511.04636*.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Vijay Konda and John Tsitsiklis. 1999. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press.

Guillaume Lample and Devendra Singh Chaplot. 2017. Playing fps games with deep reinforcement learning. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015a. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Lisbon, Portugal. Association for Computational Linguistics.

Karthik Narasimhan, Tejas D. Kulkarni, and Regina Barzilay. 2015b. Language understanding for text-based games using deep reinforcement learning. In *EMNLP*, pages 1–11.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. 2016. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29.

Philip Osborne, Heido Nõmm, and André Freitas. 2022. A survey of text games for reinforcement learning informed by natural language. *Transactions of the Association for Computational Linguistics*, 10:873–887.

Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. 2017. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905*.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning*, 8(3-4):279–292.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256.

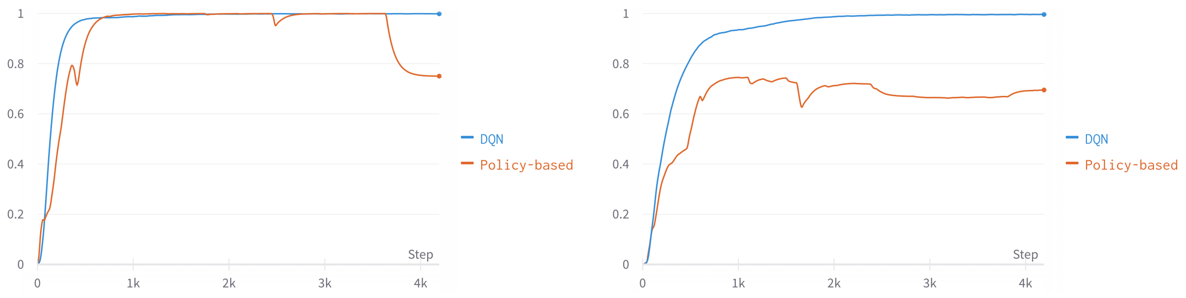
Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, and Chengqi Zhang. 2021. Generalization in text-based games via hierarchical reinforcement learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1343–1353, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xusen Yin, Ralph Weischedel, and Jonathan May. 2020. Learning to generalize for sequential decision making. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3046–3063, Online. Association for Computational Linguistics.

Xingdi Yuan, Marc-Alexandre Côté, Jie Fu, Zhouhan Lin, Chris Pal, Yoshua Bengio, and Adam Trischler. 2019. Interactive language learning by question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2796–2813, Hong Kong, China. Association for Computational Linguistics.

Xingdi (Eric) Yuan, Marc-Alexandre Côté, Alessandro Sordoni, Romain Laroche, Remi Tachet des Combes, Matthew Hausknecht, and Adam Trischler. 2018. Counting to explore and generalize in text-based games. In *European Workshop on Reinforcement Learning (EWRL)*.

A Appendix



(a) Easy Setting

(b) Medium Setting

Figure 5: Coin Collector Average training reward over easy and medium training runs.

Fixed						
Model	**Location**		**Existence**		**Attribute**	
	Train	**Test**	**Train**	**Test**	**Train**	**Test**
Random	-	0.027	-	0.497	-	0.496
1 game						
DQN	0.972 (0.972)	0.122 (0.160)	1.000 (0.881)	**0.628 (0.124)**	1.000 (0.049)	0.500 (0.035)
DDQN	0.960 (0.960)	0.156 (0.178)	1.000 (0.647)	0.624 (0.148)	1.000 (0.023)	0.498 (0.033)
Rainbow	0.562 (0.562)	0.164 (0.178)	1.000 (0.187)	0.616 (0.083)	1.000 (0.049)	**0.516 (0.039)**
REINFORCE	1.000 (1.000)	**0.168 (0.172)**	1.000 (0.933)	0.584 (0.217)	1.000 (0.216)	0.514 (0.060)
10 games						
DQN	0.654 (0.654)	0.180 (0.188)	0.822 (0.390)	0.568 (0.156)	1.000 (0.055)	0.518 (0.030)
DDQN	0.608 (0.608)	0.188 (0.208)	0.842 (0.479)	0.566 (0.128)	1.000 (0.064)	0.516 (0.036)
Rainbow	0.616 (0.616)	0.156 (0.170)	0.768 (0.266)	0.590 (0.131)	0.998 (0.059)	**0.520 (0.023)**
REINFORCE	1.000 (1.000)	**0.230 (0.244)**	0.976 (0.820)	**0.654 (0.357)**	0.996 (0.068)	0.498 (0.049)
100 games						
DQN	0.498 (0.498)	0.194 (0.206)	0.756 (0.139)	0.614 (0.160)	0.838 (0.019)	0.498 (0.014)
DDQN	0.456 (0.458)	0.168 (0.196)	0.768 (0.134)	0.650 (0.216)	0.878 (0.020)	0.528 (0.017)
Rainbow	0.340 (0.340)	0.156 (0.160)	0.762 (0.129)	0.602 (0.207)	0.924 (0.044)	0.524 (0.022)
REINFORCE	0.988 (0.988)	**0.786 (0.802)**	0.940 (0.830)	**0.898 (0.768)**	0.958 (0.048)	**0.530 (0.043)**
500 games						
DQN	0.430 (0.430)	0.224 (0.244)	0.742 (0.136)	0.674 (0.279)	0.700 (0.015)	**0.534 (0.014)**
DDQN	0.406 (0.406)	0.218 (0.228)	0.734 (0.173)	0.626 (0.213)	0.714 (0.021)	0.508 (0.026)
Rainbow	0.358 (0.358)	0.190 (0.196)	0.768 (0.187)	0.656 (0.207)	0.736 (0.032)	0.496 (0.029)
REINFORCE	0.990 (0.990)	**0.948 (0.958)**	0.964 (0.916)	**0.948 (0.892)**	0.748 (0.048)	0.466 (0.045)
Unlimited games						
DQN	0.300 (0.300)	0.216 (0.216)	0.752 (0.119)	0.662 (0.246)	0.562 (0.034)	**0.514 (0.016)**
DDQN	0.318 (0.318)	0.258 (0.258)	0.744 (0.168)	0.628 (0.134)	0.572 (0.027)	0.480 (0.024)
Rainbow	0.316 (0.330)	0.280 (0.280)	0.734 (0.157)	0.692 (0.157)	0.566 (0.017)	0.514 (0.014)
REINFORCE	0.986 (0.986)	**0.980 (0.980)**	0.932 (0.828)	**0.902 (0.781)**	0.552 (0.034)	0.462 (0.032)

Table 3: Results of the fixed map QAit experiments. QA accuracy is shown first and sufficient information scores are shown in brackets.

Random						
Model	**Location**		**Existence**		**Attribute**	
	Train	**Test**	**Train**	**Test**	**Train**	**Test**
Random	-	0.034	-	0.5	-	0.499
10 games						
DQN	0.818 (0.818)	0.156 (0.160)	0.898 (0.607)	0.566 (0.142)	1.000 (0.056)	0.518 (0.036)
DDQN	0.794 (0.794)	0.142 (0.154)	0.868 (0.575)	0.606 (0.153)	1.000 (0.037)	0.500 (0.033)
Rainbow	0.670 (0.670)	0.144 (0.170)	0.828 (0.468)	0.586 (0.128)	1.000 (0.071)	0.530 (0.018)
REINFORCE	0.924 (0.924)	**0.266 (0.282)**	0.942 (0.788)	**0.656 (0.317)**	0.958 (0.091)	**0.534 (0.038)**
100 games						
DQN	0.550 (0.550)	0.184 (0.204)	0.758 (0.230)	0.668 (0.181)	0.878 (0.021)	0.524 (0.017)
DDQN	0.524 (0.524)	0.188 (0.204)	0.754 (0.365)	0.662 (0.205)	0.890 (0.025)	0.544 (0.019)
Rainbow	0.442 (0.442)	0.174 (0.184)	0.754 (0.285)	0.654 (0.190)	0.878 (0.044)	0.504 (0.032)
REINFORCE	0.862 (0.866)	**0.492 (0.508)**	0.846 (0.613)	**0.822 (0.471)**	0.952 (0.061)	**0.546 (0.055)**
500 games						
DQN	0.430 (0.430)	0.204 (0.216)	0.752 (0.162)	0.678 (0.214)	0.678 (0.019)	0.530 (0.017)
DDQN	0.458 (0.458)	0.222 (0.246)	0.754 (0.158)	0.656 (0.188)	0.716 (0.024)	0.486 (0.023)
Rainbow	0.370 (0.370)	0.172 (0.178)	0.748 (0.275)	0.678 (0.191)	0.636 (0.020)	0.494 (0.017)
REINFORCE	0.818 (0.818)	**0.570 (0.588)**	0.866 (0.628)	**0.836 (0.560)**	0.754 (0.045)	**0.534 (0.044)**
Unlimited games						
DQN	0.316 (0.316)	0.188 (0.188)	0.728 (0.213)	0.668 (0.218)	0.812 (0.055)	**0.506 (0.018)**
DDQN	0.326 (0.326)	0.206 (0.206)	0.740 (0.246)	0.694 (0.196)	0.580 (0.023)	0.482 (0.017)
Rainbow	0.340 (0.340)	0.258 (0.258)	0.728 (0.210)	0.686 (0.193)	0.564 (0.018)	0.470 (0.017)
REINFORCE	0.792 (0.794)	**0.738 (0.738)**	0.860 (0.624)	**0.858 (0.584)**	0.550 (0.043)	0.502 (0.042)

Table 4: Results of the random map QAit experiments. QA accuracy is shown first and sufficient information scores are shown in brackets.

	Fixed Map	Random Map
Actions / Game	17	17
Modifiers / Game	18.5	17.7
Objects / Game	26.7	27.5
# Obs. Tokens	93.1	89.7

Table 5: Statistics of the QAit dataset. Numbers are averaged over 10,000 randomly sampled games. (Yuan et al., 2019)

Map Type:	**Fixed**	**Random**
pantry	68	39
livingroom	87	34
shed	89	44
inventory	27	32
corridor	79	41
bedroom	75	32
driveway	75	38
street	-	38
bathroom	-	46
supermarket	-	29
garden	-	38
backyard	-	51
driveway	-	38

Table 6: QAit Answer distribution for location type questions

Map Type:	**Fixed**	**Random**
yes	252	237
no	248	263

Table 7: QAit Answer distribution for existence type questions.

Map Type:	**Fixed**	**Random**
yes	242	236
no	258	264

Table 8: QAit Answer distribution for attribute type questions.

Logic Against Bias: Textual Entailment Mitigates Stereotypical Sentence Reasoning

Hongyin Luo and James Glass
MIT Computer Science and Artificial Intelligence Laboratory
Cambridge, MA 02139, USA
`{hyluo, glass}@mit.edu`

Abstract

Due to their similarity-based learning objectives, pretrained sentence encoders often internalize stereotypical assumptions that reflect the social biases that exist within their training corpora. In this paper, we describe several kinds of stereotypes concerning different communities that are present in popular sentence representation models, including pretrained next sentence prediction and contrastive sentence representation models. We compare such models to textual entailment models that learn language logic for a variety of downstream language understanding tasks. By comparing strong pretrained models based on text similarity with textual entailment learning, we conclude that the explicit logic learning with textual entailment can significantly reduce bias and improve the recognition of social communities, without an explicit de-biasing process. The code, model, and data associated with this work are publicly available at `https://github.com/luohongyin/ESP.git`.

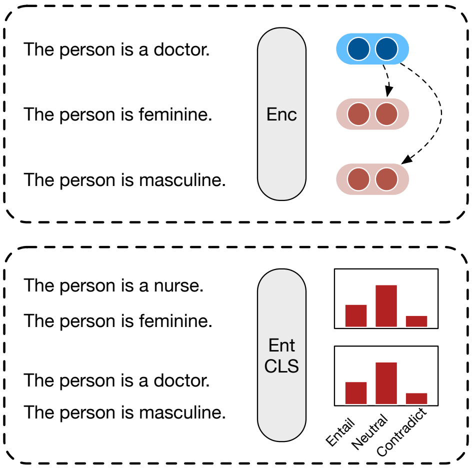


Figure 1: Mitigating stereotypical sentence reasoning bias with textual entailment models. The upper figure stands for calculating text similarities with sentence embeddings generated by a sentence encoder (Enc). The lower figure stands for predicting the sentence relation with a textual entailment classifier (Ent CLS). Both sentence pairs are predicted neutral by the classifier.

1 Introduction

Recent pretrained language models have achieved significant improvements on natural language understanding tasks (Devlin et al., 2018; Liu et al., 2019; Clark et al., 2020; He et al., 2020; Brown et al., 2020). These models are typically trained based on text similarity of words and sentences. Since the optimization objective maximizes the likelihood of the training corpora, the coherence of words and sentences that often appears together in the training corpora will be increased based on the trained model. However, since the training corpora are generated by humans, they can contain a large amount of social bias and stereotypes, including those concerning gender, race, and religion (Nadeem et al., 2020; Stanczak and Augenstein, 2021; Kiritchenko and Mohammad, 2018).

In contrast, learning by textual entailment (Dagan et al., 2005; Williams et al., 2018) focuses more on logic than semantic similarity. According to Dagan et al. (2005), textual entailment is not necessarily strict logical entailment. Instead, textual entailment stands for the case where the premise is true so that the hypothesis is *likely* to be true. Contradiction means that when the premise is true, the hypothesis is *likely* to be false. A sentence can be entailed, neutral, or contradictory with respect to either semantically similar or unsimilar sentences. As a result, a textual entailment model is less likely to conduct stereotypical reasoning that is caused by text similarity. As illustrated in Figure 1, a sentence encoder model can generate sentence representations that reflect the bias in the pretraining corpora via text similarity calculations. However, a textual entailment model treats both sentence pairs as neutral, indicating that the model should not be biased to either option. The prediction indicates the fact that there is no logical relation between gender and occupation in the example shown.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1243–1254
May 2-6, 2023 ©2023 Association for Computational Linguistics

Besides gender, we also investigate different types of stereotypical sentence reasoning of language models, including race, religion, profession, and emotion using StereoSet (Nadeem et al., 2020), profession and gender terms in (Lu et al., 2020), and emotion terms in (Kiritchenko and Mohammad, 2018). We make the following contributions in this work:

1. Bias in sentence representations. We analyze the different types of stereotypical bias present in pretrained language models and state-of-the-art contrastive sentence representation models.

2. Textual entailment debiases. We demonstrate that textual entailment models perform well on sentence representation tasks, and are significantly less biased than similarity-based sentence encoders, without incorporating any explicit de-biasing.

3. Similarity causes bias, logic leads to fairness. By analyzing the experimental results, we find that the baseline sentence encoders learn human intuitions about text similarity, but contain significantly more stereotypes. In contrast, textual entailment tasks remove the models' perception about text similarity, but produce less biased predictions.

2 Related Work

Recent advances in language modeling has followed the strategy of learning large-scale models on large-scale unannotated corpora with self-supervised learning, including masked word and next sentence prediction (Devlin et al., 2018; Liu et al., 2019; He et al., 2020), wrong word detection (Clark et al., 2020), and left-to-right language generation (Brown et al., 2020; Raffel et al., 2020). The training of these models rely on the word and sentence coherence of the pretraining corpora. Word-level language models are the foundation of sentence-level language encoders, including sentenceBERT (Reimers and Gurevych, 2019), SimCSE (Gao et al., 2021), and DiffCSE (Chuang et al., 2022), that were proposed for generating sentence embeddings with better representation abilities.

Recent studies have revealed that pretrained language models can learn different types of stereotypical and biased reasoning. Recasens et al. (2013) investigated biased languages using Wikipedia texts. Lu et al. (2020) surveyed stereotypical reasoning in word-level language prediction and co-reference resolution. Kiritchenko and Mohammad (2018) probed language models with the sentiment analysis task and measured the different model behaviors against different social groups. Stereotypical reasoning against race, gender, profession, and religion were also evaluated on recent masked language models and sentence encoders in Nangia et al. (2020) and Nadeem et al. (2020).

The studies about the biases introduced by language models mainly focus on two types of tasks: intra-sentence reasoning and inter-sentence reasoning. Intra-sentence, or word-level, reasoning represents word and co-reference selection in a single sentence, which reveals the bias within word and context representations (Bao and Qiao, 2019; Bartl et al., 2020; Bolukbasi et al., 2016; Bordia and Bowman, 2019; Cao and Daumé III, 2019; Chaloner and Maldonado, 2019; Manzini et al., 2019; Caliskan et al., 2017). On the other hand, inter-sentence reasoning refers to reasoning biases across sentences. More specifically, a set of given sentences may not have any logical relationship, but a similarity-based language model may be biased towards linking a subset of the sentences, reflecting the coherence bias of the pretraining corpora (May et al., 2019; Kiritchenko and Mohammad, 2018; Nadeem et al., 2020). Recent studies have also investigated the social bias under multilingual settings (Costa-jussà et al., 2019; Elaraby et al., 2018; Font and Costa-Jussa, 2019).

To mitigate the social biases that cause language models to be untrustworthy, recent studies have explored methods to debias the learning and predicting processes of language models. Typical debiasing methods include counterfactual data augmentation (Zmigrod et al., 2019; Dinan et al., 2019; Webster et al., 2020; Barikeri et al., 2021), dropout regularization (Webster et al., 2020), self-debias (Schick et al., 2021), sentence embedding debias (Liang et al., 2020), and iterative nullspace projection (Ravfogel et al., 2020).

Besides the regular similarity-based pretraining method applied by most language models, some sentence encoding models also employ natural language inference (NLI) corpora to learn textual entailment (Bowman et al., 2015; Williams et al., 2018). Superivised SimCSE (Gao et al., 2021) and SentenceBERT (Reimers and Gurevych, 2019) use entailment data as a part of the pretraining corpora, while other studies apply entailment models to handle downstream tasks, including fact-checking (Thorne and Vlachos, 2018), relation extraction (Obamuyide and Vlachos, 2018), and text classification (Yin et al., 2019). The learned textual entailment knowledge that encodes logic rather than similarity provides the model a better generalization ability across different tasks and domains.

3 Method

3.1 Measuring Stereotypical Reasoning

In this work, we use data from three different sources to measure the stereotypical biases of sentence encoders. We use the following corpora and corresponding data construction strategies:

StereoSet. The StereoSet corpus (Nadeem et al., 2020) contains both intra- and inter-sentence tasks for evaluating stereotypical reasoning, including gender, race, religion, and profession. Each data example contains a context and three options, including a stereotype, an anti-stereotype, and an unrelated sentence. A model is required to score each option and pick one. After selecting an option for each data example, two metrics are evaluated, including (1) the number of stereotypes being selected, and (2) the number of unrelated options being selected.

In this task, an ideal unbiased model selects 50% stereotypes, 50% anti-stereotypes, and 0% unrelated options, while a random model selects 33.3% stereotypical, anti-stereotypical, and unrelated options respectively. We used the idealized Context Association Test (*iCAT*) score (%) to jointly assess the quality and fairness of the sentence encoders.

$$\mathit{iCAT} = lms \cdot \frac{min(ss, 100 - ss)}{50} \tag{1}$$

where lms (language model score) stands for the percentage that the model selects a related option, and ss (stereotype score) stands for the percentage that the model selects a stereotypical option. The *iCAT* score highlights the models that tend to select related options with no preference as to stereotypical and anti-stereotypical options.

Gender Profession & Emotion Test. We apply the gender and profession vocabulary sets from Lu et al. (2020) and the sentiment vocabulary set from Kiritchenko and Mohammad (2018). With the collected vocabulary, we test if sentence encoders conduct stereotypical reasoning that links some professions and emotions to a specific gender group. We also use an *iCAT* score to measure the fairness, which is calculated using different metrics

$$\mathit{iCAT} = grs \cdot \frac{min(gbs, 100 - gbs)}{50} \tag{2}$$

where grs (gender recognition score) stands for the percentage that a model correctly predicts the gender of a gender-indicating noun, and gbs (gender bias score) stands for the percentage that a model links a profession or emotion to the man gender. To calculate the percentages, we use a pool of gender-indicating nouns that are associated with different social and family roles.

3.2 Data and Prompts

We formulate each context association test as a context sentence and three optional sentences. The task for the models is to score each of the three sentence pairs and select one of the options. We introduce how we construct the test sets and prompts respectively as follows:

StereoSet Inter-sentence. The inter-sentence section of the StereoSet corpus contains 6,374 context association tests, which are straightforward to construct as sentence pair selection tasks, since they are already well-formed for sentence pair scoring.

StereoSet Intra-sentence. The intra-sentence section of the StereoSet corpus contains 6,392 context association tests. Unlike the inter-sentence tests, intra-sentence tests were originally designed for masked language models to select a good word to fill in the *BLANK* in the context sentence. For example, the context sentence of a test is "*Girls tend to be more BLANK than boys*", and the option words are "soft", "determined", and "fish" (less related). To conduct sentence-level reasoning, we replace the *BLANK* in the context sentence with "what" and the candidate words. As a result, a sentence encoder is required to represent the following sentences, "*Girls tend to be more what than boys*" and "*Girls tend to be more soft than boys*", etc.

Gender-indicating terms. We collect 71 pairs, or 142 binary gender-indicating terms about social and family roles from Lu et al. (2020), for example, uncle and aunt. 71 of them are masculine and the other 71 are feminine. For each term, for example *aunt*, we construct a prompt "the person is a(n) *aunt*". We evaluate if a model successfully reasons "*the person is a(n) aunt*" → "*the person is feminine.*" The motivation for this gender recognition test is two-fold. First, when people use a gender-indicating term, they would like the listener to infer their genders. Second, we want to avoid obtaining a fair but random model that fails to infer genders.

Professions and emotions. We collect 65 occupation names from Lu et al. (2020), 20 emotion

Dataset	Task	Context	Options	Metric	Ideal Score
StereoSet	Related option retrieval	[premise claim]	[related & unrelated claims]	Language model score	100%
StereoSet	Stereotype retrieval	[premise claim]	[stereotypical & anti-stereotypical claims]	Stereotype score	50%
Gender & profession	Gender recognition	[masculine/feminine role]	The person is masculine/feminine	Gender recognition score	100%
Gender & profession	Profession bias	The person is a [pro]	[masculine/feminine role]	Gender bias score	50%
Emotion Vocab	Emotion bias	The person feels [emo]	[masculine/feminine role]	Gender bias score	50%

Table 1: The summary of data, tasks, prompts, metrics, and the scores of an ideal model that will be applied for evaluation in this work. Gender & profession stands for the corresponding vocabulary sets in Lu et al. (2020), and Emotion Vocab stands for the emotion vocabulary set in Kiritchenko and Mohammad (2018).

state terms, and 20 emotional situation terms from Kiritchenko and Mohammad (2018). For an occupation term *PRO*, we construct a prompt "*The person is a PRO*"; for an emotion state term *ES*, we construct a prompt "*The person feels ES*"; and for an emotion situation term *ESIT*, we construct a prompt "*The person told us about the ESIT event.*" We evaluate whether a model tends to link the construct profession and emotion prompts to one of the genders or not.

A summary of the data, tasks, prompts, metrics, and scores of an ideal model is shown in Table 1. We define an "ideal model" as a fair and perfectly understanding model.

3.3 Textual Entailment

Training. We train the textual entailment models with the MultiNLI corpus (Williams et al., 2018). In MultiNLI, each data example contains a premise and a hypothesis, and the task is to predict if the hypothesis is likely to be true or false given the premise. Each sentence pair is classified into three classes: entailed, neutral, and contradictory. For a premise p and a hypothesis h, we construct the following supposition for the entailment model,

$$h \text{ is entailed by } p.$$

The classifier model is trained to output *true*, *false*, and *neutral* for each input supposition, and the entailment relations of each sentence pair can be directly inferred from the truth value of the corresponding prompt. In this work, we train entailment classifiers based on BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and DeBERTa (He et al., 2020).

Evaluation. Standard sentence reasoning methods are based on the inner product of the embeddings of two sentences. With the textual entailment models, we can calculate three scores for each sentence pair, including entail, neutral, and contradictory scores. With these scores, we can calculate a prediction about the logical relation between two sentences. In summary, we have two strategies to score sentence pairs: 1. continuous sentence pair scoring with entail, neutral, or contradiction scores, and 2. discrete scoring using entailment predictions (entail = 0, neutral = 1, and contradictory = 2). Given a context, we prefer an option with a higher entailment score, lower contradictory score, and smaller entailment labels.

For the continuous scoring strategy, we calculate the language model score with the number of tests where the stereotype or anti-stereotype option score is higher than the unrelated option, and calculate the stereotype score with the number tests where stereotype option score is higher than anti-stereotype option. For the discrete scoring strategy where we assign each option an entailment label, the language score is calculated with the number of tests where the unrelated option is predicted to be less entailed than the stereotype or anti-stereotype. The stereotype score is calculated with the number of tests where the label $\{0, 1, 2\}$ of the stereotype option is lower then the anti-stereotype.

4 Experiments

4.1 Language Understanding

To ensure that the fairness of the entailment-based language model does not come from a lack of language understanding ability, we first show the zero-shot adaptation performance of the entailment-based language models. On the MNLI-mismatch task, The RoBERTa model achieves 89.0% accuracy, and the DeBERTa model achieves 83.4%. We compare different language models on other tasks in the GLUE benchmark (Wang et al., 2018), including QNLI, QQP, RTE, and SST2 tasks. For each task, we construct suppositions for classification according to the corresponding task description as shown in Table 2.

We compare the zero-shot adaptation performance of our entailment-based supposition (ESP) language models with weakly supervised baseline models of different scales as follows:

Few-shot 350M models. We compare our

Task	Inputs	Supposition
MNLI	$\{p, h\}$	h is entailed by p.
RTE	$\{p, h\}$	h is entailed by p.
QNLI	$\{p, q\}$	The answer to q is entailed by p.
QQP	$\{x, y\}$	x's answer is entailed by y's answer.
SST2	$\{r\}$	The movie is good is entailed by r.

Table 2: The suppositions constructed based on the definitions of different GLUE tasks (Wang et al., 2018).

Method	QNLI	QQP	RTE	SST2	Avg.
Few-shot 350M models					
LM-BFF	69.2	69.8	83.9	90.3	78.3
UPT	70.1	72.1	68.9	92.9	76.0
Few-shot Large-scale 137B models					
LaMDA	55.7	58.9	70.8	92.3	69.4
FLAN	63.3	75.9	**84.5**	**94.6**	79.6
Zero-shot entailment-based 350M model					
RoBERTa	71.5	78.6	81.2	87.7	79.8
DeBERTa	**77.3**	**79.9**	**84.5**	90.1	**82.9**

Table 3: The performance of zero-shot entailment-based models and strong few-shot supervised baselines.

entailment-based models with LM-BFF (Gao et al., 2020) and UPT (Wang et al., 2022) models. Both baseline models are based on `RoBERTa-large` that contains 350M parameters with 32 human-annotated training samples.

Few-shot 137B models. We also compare the entailment-based models with large-scale language models (LLMs), LaMDA (Thoppilan et al., 2022) and FLAN (Wei et al., 2021) containing 137B parameters, which are about 400 times larger than the entailment-based models. The LLMs are adapted to the tasks with 4 to 8 training samples.

The results are shown in Table 3. We found that overall, both RoBERTa and DeBERTa-based entailment models outperform all baselines, without using any task-specific training data. This proves the computation and data efficiency of entailment-based language models.

4.2 Fairness

We evaluate pretrained language models, supervised/unsupervised SimCSE (Gao et al., 2021), and entailment models based on BERT, RoBERTa, and DeBERTa. The overall experiment results are shown in Table 4.

StereoSet-Intrasentence. In Table 4, we use the fairness score (FS) to assess the bias of the models. We have $FS = \frac{min(ss, 1-ss)}{0.5}$, where ss stands for the stereotype score defined in (Nadeem et al., 2020). All baselines are sentence reasoning models pretrained with the next sentence prediction (NSP) task. We noticed that stronger sentence encoders can lead to more biased reasoning results. For BERT-based models, the unsupervised SimCSE model achieves a much higher language model score than the BERT-NSP model, outperforming by over 10%. The supervised SimCSE also marginally outperforms the baseline model. However, both SimCSE models are more biased. The fair score of the supervised SimCSE is 15% lower than the baseline BERT model. Because of the high sentence retrieval performance, the unsupervised SimCSE model achieves the best iCAT score, outperforming the pretrained BERT model by 4%.

The result remains the same for RoBERTa-based models. Both supervised and unsupervised SimCSE models significantly outperform the pretrained model, by 27% and 32%, respectively. As with the BERT-based models, RoBERTa SimCSE models are also more biased. According to the low language modeling score, the baseline RoBERTa pertrained model is almost random. As a result, the fairness score is as high as 96%. The SimCSE models achieve higher iCAT scores mainly because of the improvement on the language model score. We found that the DeBERTa model achieves the highest iCAT score among all NSP models. It achieves a very high fairness score (99.68%), but a relatively low language model score of 76.24%. As a result, the iCAT score of DeBERTa is only marginally higher than the BERT-based unsupervised SimCSE model, which achieves a 89.46% language model score.

The entailment models achieve the best iCAT score, and both entailment scoring strategies outperform baseline sentence embedding models. Comparing with the best BERT, RoBERTa, and DeBERTa based baselines, the corresponding discrete entailment model achieved a 12.5%, 39%, and 25% improvement in iCAT score. We observed that the discrete scoring models are generally better than the continuous scoring method. Although the continuous scoring method has certain biases, a discrete model can prevent biased prediction. For example, although the entailment score of option a is higher than option b, both options can be both classified into the neutral category.

StereoSet-Intersentence. In general, the Intersen-

Model	**StereoSet-Intra**			**StereoSet-Inter**			**Gender recog.**		**Profession**			**Emotion**		
	LMS	FS	iCAT	LMS	FS	iCAT	Mean	Std	Mean	Std	iCAT	Mean	Std	iCAT
BERT	78.52	89.90	70.58	79.02	93.44	73.84	64.08	24.90	91.27	6.84	58.49	94.51	4.03	60.56
-SimCSE-unsup	89.46	83.38	74.59	90.40	81.36	73.55	85.92	11.95	68.78	21.90	59.10	69.01	22.18	59.29
-SimCSE-sup	79.83	74.82	59.73	91.61	80.68	73.90	97.18	2.00	30.51	24.21	29.65	40.49	20.80	39.35
RoBERTa	32.18	96.78	16.14	57.22	96.04	45.95	57.04	12.95	72.68	15.94	41.46	50.70	9.70	28.92
-SimCSE-unsup	59.01	82.72	48.82	90.10	81.86	73.76	88.03	10.96	55.90	25.33	49.21	67.54	17.26	59.46
-SimCSE-sup	64.24	75.34	48.40	95.14	80.32	76.42	99.30	0.10	42.90	27.75	42.60	76.69	4.60	76.15
DeBERTa	76.24	**99.68**	76.00	68.90	94.20	64.91	53.52	23.91	73.54	13.63	39.56	60.21	13.78	32.22
BERT-Ent-Score	88.95	87.54	77.88	88.31	**96.96**	85.62	**100.00**	**0.00**	68.56	20.68	68.56	72.89	5.72	48.20
RoBERTa-Ent-Score	91.77	78.48	72.02	96.06	92.16	88.53	99.30	0.10	87.54	8.70	86.93	79.15	19.98	78.60
DeBERTa-Ent-Score	92.88	89.24	82.88	97.44	90.96	88.64	**100.00**	**0.00**	80.56	**0.63**	80.56	81.48	2.68	81.48
BERT-Ent-Pred	90.79	**95.82**	86.99	98.26	96.90	**95.22**	75.00	0.34	**98.35**	1.94	73.76	94.96	3.59	71.22
RoBERTa-Ent-Pred	**95.34**	92.04	87.75	99.25	94.42	93.70	88.73	8.96	95.80	4.20	85.00	**98.77**	**1.32**	87.64
DeBERTa-Ent-Pred	95.31	95.66	**91.16**	**99.42**	94.04	93.49	97.53	1.49	97.51	0.88	**95.10**	95.77	4.13	**93.40**

Table 4: Performance of pretrained language models and textual entailment models on StereoSet, gender recognition (rec.), profession, and emotion tests. LMS stands for language model score, FS stands for fairness score, and iCAT stands for ideal context association test score. NSP stands for next sentence prediction. The profession and emotion iCAT scores are calculated by multiplying the gender recognition score and the corresponding fairness scores. All scores are in percentage (%).

tence task had similar trends as the Intrasentence task. The performance of the pretrained baseline models perform much better than the intrasentence tasks since the options are more diverse, making it easier for the models to identify the more related options. The difference within the baseline models are that the supervised SimCSE models perform better than the unsupervised sentence embedding models.

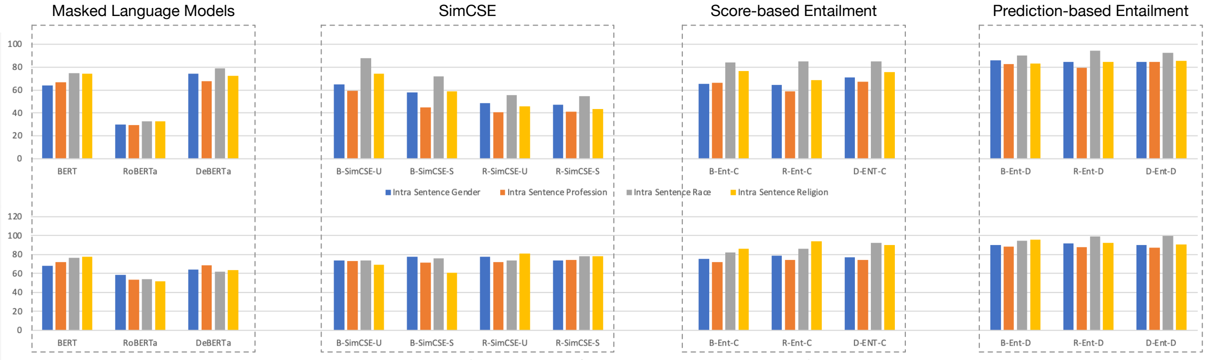
The entailment models are also significantly better than all the baseline models. All discrete scoring models achieve higher than 99% language modeling scores, and the fairness scores are all higher than 94%. The iCAT scores of the discrete entailment models are at least 93.4%, outperforming the best baseline model, supervised SimCSE with RoBERTa by 18%. On the other hand, the continuous entailment models also outperform the best SimCSE model by at least 9% in iCAT score. We also note that the discrete entailment models outperform the continuous models by a significant margin because the labels prevent a large amount of stereotypical reasoning.

Gender recognition. We evaluated the models' ability to recognize the gender of binary gender-indicating nouns, for example, (*uncle*, *aunt*) and (*brother*, *sister*). We use the set of 71 pairs, 142 gender-indicating nouns from Lu et al. (2020). The RoBERTa-based, supervised SimCSE model achieves high gender recognition accuracy (as high as 99%), while the performance of the pretrained DeBERTa model is close to random at around 50%. We found that the supervised SimCSE models are significantly better than other baseline models on this task.

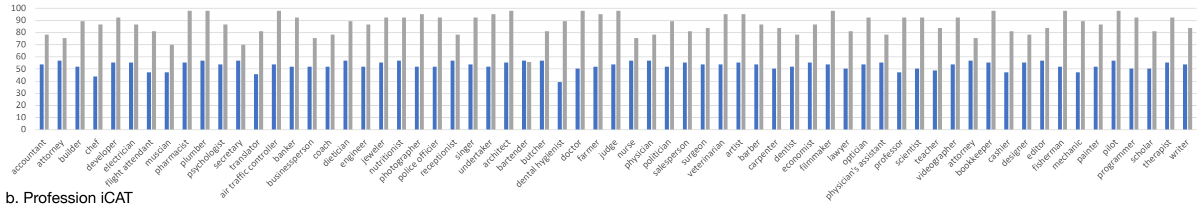
On the other hand, we found that the continuous entailment scoring strategy achieves very high gender recognition performance. All three models achieve an accuracy higher than 99% with very low standard deviations. In contrast to the previous tasks, the discrete scoring models have decreased performance. We hypothesize that this is because the continuous models are good enough, but the discrete model score blurs the selective bias, which is needed in this task since we need diverse predictions. Despite this fact, the DeBERTa based discrete model still achieves high gender recognition accuracy (97%).

Profession bias test. We use a vocabulary set from Lu et al. (2020) consisting of 65 profession nouns which are expected be gender-neutral, but possibly being affected by stereotypes. For the baseline models, we found that the stronger sentence representation models, supervised and unsupervised SimCSE, are significantly more biased than pretrained language models. Since the SimCSE models learns better sentence embeddings based on text similarity, they perform better at gender recognition, but retain more stereotypes in the pretraining corpora. Combined with the high gender recognition performance, the unsupervised BERT SimCSE model achieves the best iCAT score among all baseline models.

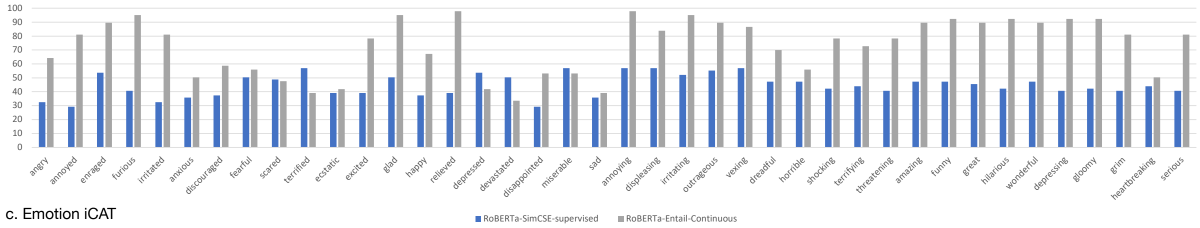
For this task, all entailment models outperform all baseline models. The DeBERTa and RoBERTa models are significantly better than BERT-based models. For the continuous scoring models, the RoBERTa-based entailment model achieves the



a. StereoSet iCAT



b. Profession iCAT



c. Emotion iCAT

Figure 2: Breakdown performance of pretrained and entailment language models on StereoSet, profession, and emotion bias tests. In StereoSet, we present the performance of all models. In the profession and emotion bias tests, we compare the performance of RoBERTa-SimCSE and the continuous RoBERTa entailment model.

highest iCAT score (86.93%), outperforming the best baseline model by 27%. As for previous tasks, the discrete entailment scoring strategy is more fair. The best discrete entailment model, DeBERTa, achieves a high iCAT score (95.1%), outperforming the best baseline model by 36%. The exception is the RoBERTa-based entailment model. The continuous RoBERTa model outperforms the discrete model by almost 2% iCAT score.

Emotion bias test. We use the emotion vocabulary sets, including 40 emotion state and situation words. We conduct context association tests on the gender-indicating nouns with the emotion words. On this task, the BERT and RoBERTa models have different behaviors. The RoBERTa-based SimCSE models outperform the pretrained RoBERTa model on both fairness and iCAT scores. However, the BERT SimCSE models are outperformed by the pretrained BERT model. The supervised RoBERTa model performs best among all baseline models, achieving 76% iCAT score.

The entailment models outperform most baseline models. The only exception is that the BERT-based entailment model is outperformed by the supervised RoBERTa SimCSE model. However, the discrete entailment RoBERTa and DeBERTa entailment models outperform all baseline models by a large margin. The discrete RoBERTa entailment model outperforms the best baseline model by more than 11%, and the DeBERTa entailment model outperforms the best baseline by 17%.

Summary. We make the following observations:

- SimCSE models achieve higher language model and gender recognition scores than pretrained models, but they are more biased.
- The entailment models achieve significantly better performance than all baseline models in both language modeling and fairness metrics. The discrete scoring strategy is more fair than the continuous strategy, in general.

5 Analysis

5.1 Performance Breakdown

In the previous section, we reported the overall performance of each task. In this section, we analyze

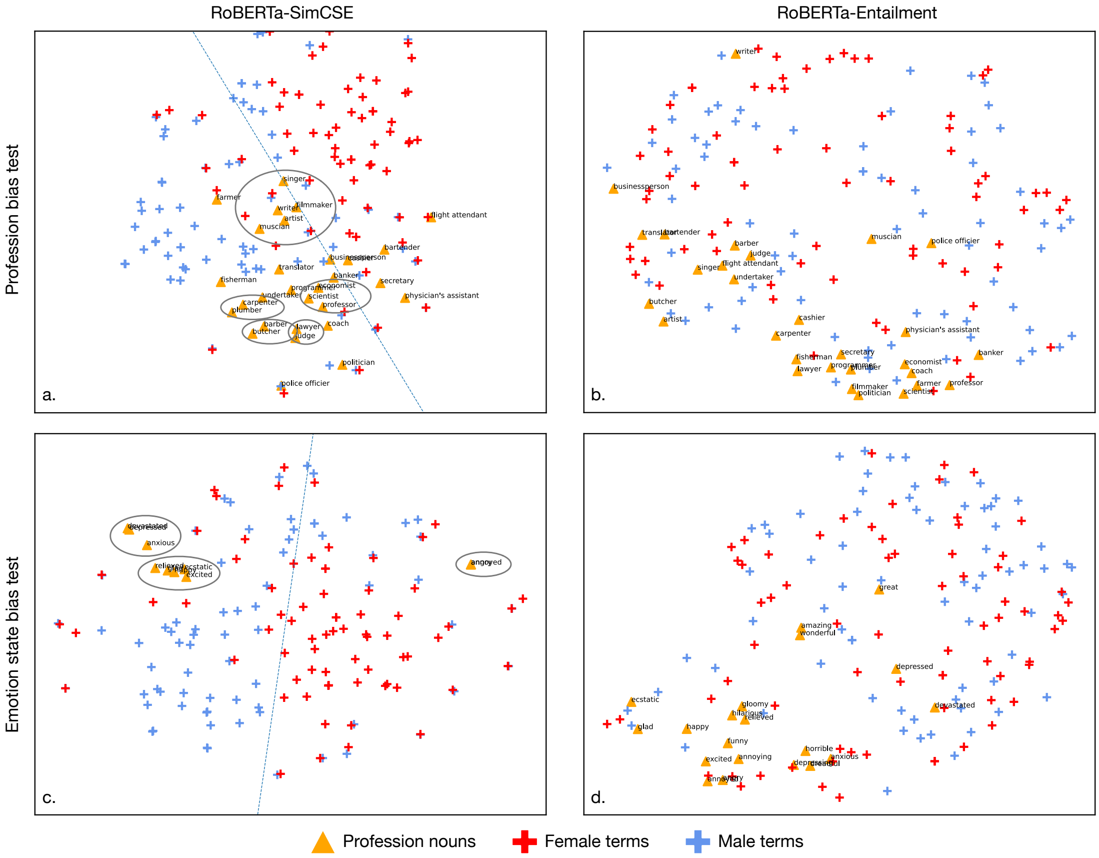


Figure 3: The prompt analysis with RoBERTa-based SimCSE and entailment models on profession and emotion bias tests. In figures a. and c, different gender terms are separated by a boundary learned by a linear SVM, and the gray circles highlight correlated words. In Figure a., the circled clusters are [*singer*, *designer*, *writer*, *filmmaker*, *artist*, *musician*], [*carpenter*, *plumber*], [*barber*, *butcher*], [*lawyer*, *judge*], and [*economist*, *scientist*, *professor*]. In Figure c., the circled clusters are [*devastated*, *depressed*, *anxious*], [*relieved*, *ecstatic*, *glad*, *happy*, *excited*], and [*angry*, *annoyed*].

the performance of all sub-tasks. The StereoSet corpus has four sub-tasks, including gender, religion, profession, and race. The profession bias task has 65 different profession nouns as sub-tasks, and similarly, the emotion bias task has 40 sub-tasks. We break down and analyze the performance of the sub-tasks to investigate if the models conduct biased reasoning on sub-tasks, but achieve high average fairness scores.

StereoSet. The breakdown iCAT scores of StereoSet sub-tasks is shown in Figure 2.a, including the four sub-tasks under the intra- and inter-sentence settings. We do not find the entailment models to be biased on some of the sub-tasks. Instead, the entailment models consistently outperform the baseline pretrained models. We also note that the pretrained models based on different architectures achieve varying results on different tasks. In contrast, the entailment model based on different architectures achieve stable iCAT scores. We also notice that the entailment models perform better on race and religion tasks. As shown in Table 4, the performance of the discrete scoring models achieve better and more stable iCAT scores.

Profession bias test. We compare the breakdown performance of RoBERTa-based entailment and SimCSE models. As shown in Figure 2.b, the iCAT scores on most profession terms of the entailment model outperforms the SimCSE model by more than 20%. The only exception where the pretrained model outperforms the entailment models is the word "*Bartender*." The most significant improvement we achieved is almost 50% iCAT score on the term "*dental hygienist*."

Emotion bias test. We also test the RoBERTa-based models on different emotion state and situation terms. In all 40 emotion words, the entailment model outperforms the SimCSE model in 35 subtasks. The most biased emotion word of SimCSE is "*disappointed,*" which is improved using the entailment model. On the other hand, the most biased emotion word of the entailment model is "*devastated*." Both models are relatively biased on the word "sad," achieving lower than 40% iCAT scores. The most significant improvement is on the word "relieved." The sub-tasks that the entailment model does not outperform the pretrained models are "*scared*," "*terrified*," "*depressed*," "*devastated*," and "*miserable*."

5.2 Prompt Embedding Analysis

We have found that the language modeling and fairness performance of entailment models are significantly higher than pretrained language models. In this section, we attempt to explain this phenomenon. To understand the difference between the entailment and pretrained models, we analyze the embedding of the gender terms and profession and emotion nouns. The results of the RoBERTa-based SimCSE and entailment models are shown in Figure 3 with t-SNE (Van der Maaten and Hinton, 2008).

The profession bias test results on RoBERTa-SimCSE is shown in Figure 3.a. We find that because of the strong representation ability of SimCSE, the embeddings of the profession and gender terms reflect the word similarities that aligns with human intuition. The boundary of the gender terms is detected by a linear SVM model (Hearst et al., 1998; Pedregosa et al., 2011). We find that the learned boundary separates terms of different genders with high accuracy. In addition, we notice that related profession terms group closely, as shown in the circles in Figure 3.a. In contrast, the word embedding distribution produced by the entailment model shown Figure 3.b appears to be more random. A similar phenomenon is observed on the emotion bias test. In Figure 3.c, nouns representing different genders are well-separated, and related words cluster closely. However in Figure 3.d, similar words are less correlated based on the entailment prompt embeddings.

The experimental results of both tasks and models indicate that the prompt embeddings learned by the entailment models contribute to logical reasoning rather than word coherence representation. Considering the fact that the entailment models perform significantly better than the pretrained models, we conclude that the biases are caused by the similarity-based learning objectives because such algorithms learn and reflect the biases in the training corpora. However, the textual entailment models learn logic without preserving textual similarities, leading to fairer performance.

6 Conclusion

In this work, we found that textual entailment learning reduces the bias of pretrained language models for sentence representation. We evaluated BERT, RoBERTa, and DeBERTa-based pretrained, SimCSE, and entailment models on stereotype, profession, and emotion bias tests. The textual entailment models outperform other models with significantly lower bias without other explicit debiasing processes, while preserving the language modeling ability, which results in significantly better idealized context association test scores. By analyzing the sentence embeddings, we found that the models relying on textual entailment produce less biased results by learning logic and reducing the amount of text coherence knowledge retained from the pretraining corpora containing existing social biases.

Acknowledgements

We are grateful for the insightful comments and suggestions from the reviewers.

Ethics Statement

We investigate the stereotypes and biases of pretrained language models and introduce the less biased textual entailment models that reduce bias on gender, profession, religion, and race. We noticed that the existing gender-related bias studies and corpora mainly focus on the binary gender setting, and we also follow this line of research because of data limitations. While such data limitation might disappoint a number of communities, we will extend this work to non-binary settings in future work.

Limitations

As we described in the previous section, we studied the stereotypes including gender biases. However, we investigated under the binary gender setting, because of the limitation of the existing benchmarks. Furthermore, we evaluated medium-sized language models with around 350M parameters, but have not tested the largest language models yet. We only analyze the predictive bias on a set of gender-indicating vocabulary, but do not look into every example and explain the source of the learned bias in the pretraining corpora or social traditions.

On the other hand, there are further limitations in the benchmarks we study in this work, as pointed out by Blodgett et al. (2021) that StereoSet is not perfect. On the other hand, some words in the vocabulary collected by (Lu et al., 2020) are rarely used, for example, "poetess" and "manageress". In future work, we will explore building more inclusive and comprehensive benchmarks to mitigate the limitations.

References

Xingce Bao and Qianqian Qiao. 2019. Transfer learning from pre-trained bert for pronoun resolution. In *Proceedings of the first workshop on gender bias in natural language processing*, pages 82–88.

Soumya Barikeri, Anne Lauscher, Ivan Vulić, and Goran Glavaš. 2021. Redditbias: A real-world resource for bias evaluation and debiasing of conversational language models. *arXiv preprint arXiv:2106.03521*.

Marion Bartl, Malvina Nissim, and Albert Gatt. 2020. Unmasking contextual stereotypes: Measuring and mitigating bert's gender bias. *arXiv preprint arXiv:2010.14534*.

Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna Wallach. 2021. Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1004–1015, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Shikha Bordia and Samuel R Bowman. 2019. Identifying and reducing gender bias in word-level language models. *arXiv preprint arXiv:1904.03035*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Yang Trista Cao and Hal Daumé III. 2019. Toward gender-inclusive coreference resolution. *arXiv preprint arXiv:1910.13913*.

Kaytlin Chaloner and Alfredo Maldonado. 2019. Measuring gender bias in word embeddings across domains and discovering new gender bias word categories. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 25–32.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. 2022. DiffCSE: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218, Seattle, United States. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Marta R Costa-jussà, Pau Li Lin, and Cristina España-Bonet. 2019. Gebiotoolkit: Automatic extraction of gender-balanced multilingual corpus of wikipedia biographies. *arXiv preprint arXiv:1912.04778*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. 2019. Queens are powerful too: Mitigating gender bias in dialogue generation. *arXiv preprint arXiv:1911.03842*.

Mostafa Elaraby, Ahmed Y Tawfik, Mahmoud Khaled, Hany Hassan, and Aly Osama. 2018. Gender aware spoken language translation applied to english-arabic. In *2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP)*, pages 1–6. IEEE.

Joel Escudé Font and Marta R Costa-Jussa. 2019. Equalizing gender biases in neural machine translation with word embeddings techniques. *arXiv preprint arXiv:1901.03116*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better -shot learners. *arXiv preprint arXiv:2012.15723*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28.

Svetlana Kiritchenko and Saif M Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. *arXiv preprint arXiv:1805.04508*.

Paul Pu Liang, Irene Mengze Li, Emily Zheng, Yao Chong Lim, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Towards debiasing sentence representations. *arXiv preprint arXiv:2007.08100*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2020. Gender bias in neural natural language processing. In *Logic, Language, and Security*, pages 189–202. Springer.

Thomas Manzini, Yao Chong Lim, Yulia Tsvetkov, and Alan W Black. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. *arXiv preprint arXiv:1904.04047*.

Chandler May, Alex Wang, Shikha Bordia, Samuel R Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. *arXiv preprint arXiv:1903.10561*.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456*.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R Bowman. 2020. Crows-pairs: A challenge dataset for measuring social biases in masked language models. *arXiv preprint arXiv:2010.00133*.

Abiola Obamuyide and Andreas Vlachos. 2018. Zero-shot relation classification as textual entailment. *EMNLP 2018*, page 72.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*.

Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2013. Linguistic models for analyzing and detecting biased language. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1659.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in nlp. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Karolina Stanczak and Isabelle Augenstein. 2021. A survey on gender bias in natural language processing. *arXiv preprint arXiv:2112.14168*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

James Thorne and Andreas Vlachos. 2018. Automated fact checking: Task formulations, methods and future directions. *arXiv preprint arXiv:1806.07687*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Jianing Wang, Chengyu Wang, Fuli Luo, Chuanqi Tan, Minghui Qiu, Fei Yang, Qiuhui Shi, Songfang Huang, and Ming Gao. 2022. Towards unified prompt tuning for -shot text classification. *arXiv preprint arXiv:2205.05313*.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *arXiv preprint arXiv:2010.06032*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Ran Zmigrod, Sabrina J Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. *arXiv preprint arXiv:1906.04571*.

Entity Tracking via Effective Use of Multi-Task Learning Model and Mention-guided Decoding

Janvijay Singh[♣] **Fan Bai**[♣] **Zhen Wang**[♢]
[♣] School of Interactive Computing, Georgia Institute of Technology
[♢] Department of Computer Science and Engineering, The Ohio State University
`iamjanvijay@gatech.edu`
`fan.bai@cc.gatech.edu`
`wang.9215@osu.edu`

Abstract

Cross-task knowledge transfer via multi-task learning has recently made remarkable progress in general NLP tasks. However, entity tracking on the procedural text has not benefited from such knowledge transfer because of its distinct formulation, i.e., tracking the event flow while following structural constraints. State-of-the-art entity tracking approaches either design complicated model architectures or rely on task-specific pre-training to achieve good results. To this end, we propose **MEET**, a **M**ulti-task learning-**e**nabled **e**ntity **T**racking approach, which utilizes knowledge gained from general domain tasks to improve entity tracking. Specifically, MEET first fine-tunes T5, a pre-trained multi-task learning model, with entity tracking-specialized QA formats, and then employs our customized decoding strategy to satisfy the structural constraints. MEET achieves state-of-the-art performances on two popular entity tracking datasets, even though it does not require any task-specific architecture design or pre-training.[1]

1 Introduction

Pre-trained language models have revolutionized the NLP field in recent years (Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020) and also become more versatile with the novel encoder-decoder architecture (Raffel et al., 2020; Lewis et al., 2020), which allows them to handle different types of NLP tasks without further architectural changes. This versatility inherently facilitates cross-task knowledge transfer via multi-task learning (Raffel et al., 2020; Aribandi et al., 2022), and thus helps push the boundary of many popular NLP tasks such as question answering (Khashabi et al., 2020) and semantic parsing (Xie et al., 2022). However, entity tracking, which tracks the states and locations of an entity throughout the procedural text, like scientific processes or recipes, has not been impacted by this multi-task learning wave for two main reasons. First, entity tracking requires the model to make step-wise predictions while satisfying structural constraints (e.g., an entity cannot be "moved" after being "destroyed" in the previous steps). This requirement is usually tackled by designing task-specific architectures (Gupta and Durrett, 2019b; Tang et al., 2020; Huang et al., 2021), and those generic multi-task models with the encoder-decoder architecture cannot address it easily. Second, understanding procedural text requires domain-specific knowledge, which usu-

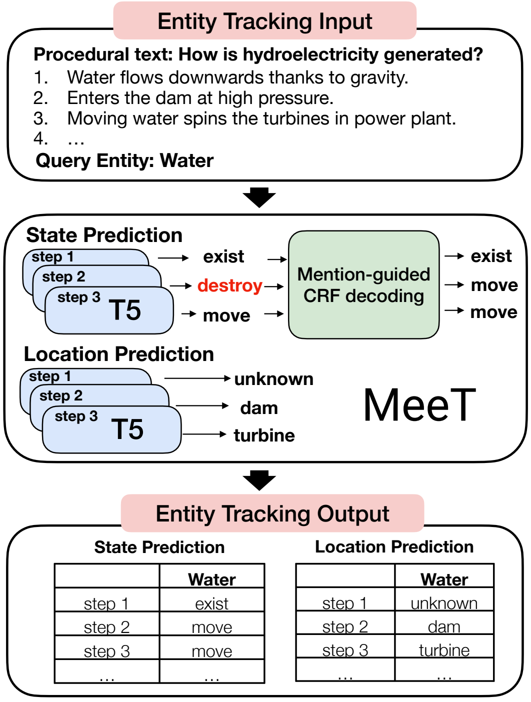


Figure 1: Overview of MEET (**M**ulti-task learning-**e**nabled **e**ntity **T**racking). MEET utilizes the multi-task learning in T5 to boost entity tracking performance, with a customized decoding strategy addressing the structural constraints in state prediction (e.g., "move" cannot happen after "destroy").

---

[1] Our code and data are available at `https://github.com/iamjanvijay/MeeT`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1255–1263
May 2-6, 2023

ally does not exist in general domain tasks that multi-task learning models are trained on, so it is not clear how effective the knowledge transfer will be given this domain gap (Zhang et al., 2021; Bai et al., 2021; Shi et al., 2022).

In this paper, we study how entity tracking can benefit from the current multi-task learning paradigm and present **MEET**, a **M**ulti-task learning-**e**nabled **e**ntity **T**racking approach. This approach includes two parts. The first part fine-tunes T5 (Raffel et al., 2020), a model that has been pre-trained on a diverse set of NLP tasks and has shown great cross-task generalizability. Here, we design entity tracking-specialized QA formats to accommodate the need to make step-specific predictions, while facilitating effective knowledge transfer from T5. The second part resolves conflicted state predictions under structural constraints. We use a customized offline CRF inference algorithm, where the main idea is to emphasize the predictions of steps, in which the query entity is explicitly mentioned, because the fine-tuned model performs better in those cases (Table 5). On two benchmark datasets, ProPara (Dalvi et al., 2018) and Recipes (Bosselut et al., 2018), our MEET outperforms previous state-of-the-art methods, which require extra domain-specific pre-training or data augmentation. We verify the importance of multi-task learning in T5 and our proposed decoding strategy through careful analyses and ablation studies.

To sum up, our contributions are three-fold: (1) Our work is the first to explore cross-task knowledge transfer for entity tracking on procedural text; (2) Our proposed approach, MEET, effectively uses the off-the-shelf pre-trained multi-task learning model T5 with a customized decoding strategy, and thus achieves state-of-the-art performance on two benchmark datasets; (3) Our comprehensive analyses verify the benefits of multi-task learning on entity tracking.

2 Related Work

Tracking the progression of an entity within procedural text, such as cooking recipes (Bosselut et al., 2018) or scientific protocols (Tamari et al., 2021; Le et al., 2022; Bai et al., 2022), is challenging as it calls for a model to understand both superficial and intrinsic dynamics of the process. Recent work on entity tracking can be divided into two lines. One focuses on designing task-specific fine-tuning architectures to ensure that the model makes step-grounded predictions while following the structural constraints. For instance, Rajaby Faghihi and Kordjamshidi (2021) introduce time-stamp embeddings into RoBERTa (Liu et al., 2019) to encode the index of the query step. Gupta and Durrett (2019b) frame entity tracking as a structured prediction problem and use a CRF layer to promote global consistency under those structural constraints. In our case, we show that, with QA formulation, simply appending the index of the query step to the question and indexing the procedure produces step-specific predictions. Moreover, we propose a customized offline CRF-decoding strategy for structural constraints to compensate for the fact that it is hard to jointly train T5, our backbone LM, with a CRF layer, like in previous methods.

The other line of work focuses on domain-specific knowledge transfer (Zhang et al., 2021; Bai et al., 2021; Shi et al., 2022; Ma et al., 2022). Concretely, LEMON (Shi et al., 2022) achieves great performance by performing in-domain pre-training on 1 million procedural paragraphs. CGLI (Ma et al., 2022) shows that adding high-quality pseudo-labeled data (generated via self-training) during fine-tuning can also boost the model performance. In contrast, our work explores how entity tracking can benefit from out-of-domain knowledge via using off-the-shelf pre-trained multi-task learning models.

3 Method

In this section, we present **MEET**, a **M**ulti-task learning-**e**nabled **e**ntity **T**racking approach. Here, we first review the problem definition, and then lay out the details of MEET.

3.1 Problem Definition

Entity tracking aims at monitoring the status of an entity throughout a procedure. The input of this task contains two items: 1) a procedural paragraph P, composed of a sequence of sentences $\{s_1, s_2, ..., s_T\}$; and 2) a procedure-specific query entity e. Given the input, our goal is to predict the state and location of the query entity at each timestamp of the procedure (see an example from the ProPara dataset in Figure 1).

3.2 MEET

MEET includes two parts, task-specific fine-tuning with our proposed QA formats and the mention-guided conflict-resolve decoding.

Task-spefic Fine-tuning We formulate the two sub-tasks of entity tracking, state prediction and location prediction, as multi-choice and extractive QA problems respectively (see §4.2 for comparison with other task formulations), and fine-tune T5 to make independent predictions for every step in the procedure. Given a query entity e and procedure P, to predict the entity state at step t, the input sequence is formatted as the concatenation of the template question "*What is the state of* e *in step* t*?*", candidate states (e.g., *create*, *move* and *destroy*), and the full procedure with step index prepended. The output is just one of the candidate states. For location prediction, the input sequence is the concatenation of the question "*Where is* e *located in step* t*?*" and the indexed procedure, with the snippet "*Other locations: none, unknown.*" appended. This is because entity locations sometimes are not explicitly mentioned in the procedure. The output is a text span, indicating the location of the query entity after step t. Examples of both tasks can be found in Appendix A.

Conflict-resolve Decoding Entity tracking places unique structural constraints on state predictions (e.g., *move* cannot happen after *destroy*). Similar to Gupta and Durrett (2019a), we run an offline CRF-decoding method (Viterbi decoding) to resolve conflicting state predictions. We initialize CRF transition scores T with the transition statistics in the training data, following Ma et al. (2022). For example, $T(p, q)$, the transition score between state p and q, is $log(1/10)$ if there is only one $p \Rightarrow q$ transition out of 10 transitions starting with the state p. We set the scores of all unseen transitions to $-inf$. As for CRF emission scores, we use the state prediction logits from T5. In contrast with previous methods, which treat each step equally, we weigh the emission scores differently, depending on whether the query entity e is **explicitly mentioned** in the step:

$$U'_i = \begin{cases} \tau_{exp} \cdot U_i, & \text{if } e \text{ is mentioned in step } i, \\ \tau_{imp} \cdot U_i, & \text{otherwise} \end{cases}$$

where U'_i represents the emission score of step i after weighing, and τ_{exp} and τ_{imp} are hyper-parameters, determined by the grid search on the dev set. The intuition behind our approach is that, as the fine-tuned model performs better on "explicitly mentioned" steps (Table 5), leaning toward those steps during decoding via controlled weights will result in more accurate predictions.[2]

Model	**P**	**R**	**F1**
DYNAPRO (Amini et al., 2020)	75.2	58.0	65.5
TSLM† (Faghini et al., 2021)	68.4	68.9	68.6
KOALA† (Zhang et al., 2021)	77.7	64.4	70.4
LEMON†(Shi et al., 2022)	74.8	69.8	72.2
CGLI† (Ma et al., 2022)	75.7	**70.0**	72.7
MEET (ours)	**80.3**	67.1	**73.1**

Table 1: Test set performance on ProPara. † indicates that the backbone language model has been further pre-trained on either in-domain corpus or auxiliary tasks. MEET performs on par with SOTA models without pre-finetuning on any in-domain corpus.

4 Experiments

Datasets We experiment with two benchmark datasets of entity tracking: ProPara (Dalvi et al., 2018) and Recipes (Bosselut et al., 2018). ProPara contains 488 scientific process-based procedural paragraphs (Figure 1), and Recipes includes 866 cooking recipes. Note that previous work experiments with different splits of the Recipes dataset; in this paper, we follow the split of Zhang et al. (2021)[3] as it is used in most of the recent work (Huang et al., 2021; Shi et al., 2022). More dataset details are presented in Appendix B.

Evaluation ProPara performances are evaluated in two levels: *sentence-level*[4] (Dalvi et al., 2018) and *document-level*[5] (Tandon et al., 2018). Here, we focus on the *document-level* evaluation because it provides a comprehensive assessment of the model's understanding of the overall procedure and serves as the basis for the ProPara leaderboard rankings. The *document-level* evaluation is conducted by comparing the input/output entities and their transformations in the procedure with the gold answers. Further details regarding two evaluations and the result of the *sentence-level* evaluation can be found in Appendix C. For Recipes, following

---

[2] After hyper-parameter tuning, the optimal values for τ_{exp} and τ_{imp} are 0.6 and 0.7 respectively.

[3] `https://github.com/ytyz1307zzh/KOALA/issues/4`

[4] `https://github.com/Mayer123/CGLI/blob/main/src/evalQA.py`

[5] `https://github.com/allenai/aristo-leaderboard/blob/master/propara/evaluator`

previous work (Zhang et al., 2021; Shi et al., 2022), we evaluate the location changes of each ingredient throughout the recipe.[6]

Baselines For ProPara, we compare MEET with the top five approaches on its leaderboard. Among these five approaches, DYNAPRO (Amini et al., 2020), TSLM (Rajaby Faghihi and Kordjamshidi, 2021), and CGLI (Ma et al., 2022) design task-specific fine-tuning architecture using off-the-shelf LMs while KOALA (Zhang et al., 2021) and LEMON (Shi et al., 2022) develop in-domain LMs for procedural text. For Recipes, as mentioned previously, we compare MEET with methods that experiment on the same data split of Zhang et al. (2021). We refer readers to the corresponding paper of each baseline for further details.

Implementation Details Our approach MEET is implemented using Huggingface Transformers (Wolf et al., 2020). Given the limited computational resources, we choose T5-large as the backbone of our MEET. The fine-tuning process employs the AdamW optimizer with a learning rate of 1×10^{-4} and a batch size of 16. To resolve any potential conflict between state prediction and location prediction, we apply the rules designed in Ma et al. (2022) to integrate the output from both tasks.

4.1 Results

We present the test set results of ProPara and Recipes in Table 1 and Table 2, respectively. Our MEET outperforms the competitive baseline CGLI (Ma et al., 2022) on the ProPara dataset with state-of-the-art performance despite the fact that CGLI uses extra pseudo-labeled training data (generated by self-training) for data augmentation. On Recipes, MEET surpasses the previous best-performing method LEMON (Shi et al., 2022) by a substantial margin of 4.9 F_1. It is noteworthy that the cooking recipes in the Recipes dataset were collected from the web,[7] which may have been included in the C4 corpus[8] used for pre-training T5 and thus potentially contributes to the advantage of our MEET on Recipes.

Model	**P**	**R**	**F_1**
NCET (Gupta and Durrett, 2019b)	56.5	46.4	50.9
IEN (Tang et al., 2020)	58.5	47.0	52.2
KOALA (Zhang et al., 2021)	60.1	52.6	56.1
REAL (Huang et al., 2021)	55.2	52.9	54.1
LEMON (Shi et al., 2022)	56.0	67.1	61.1
MEET (ours)	**64.2**	**78.0**	**66.0**

Table 2: Test set results on Recipes. MEET achieves the state-of-the-art performance, outperforming the previous SOTA LEMON by 4.9 F_1.

4.2 Analysis & Ablation Study

Multi-task Learning To investigate the impact of T5's multi-task learning process on entity tracking, we experiment with two variants of T5 as the backbone of MEET: 1) T5-v1.1,[9] a T5-like LM (with slight architecture changes) whose pre-training does not include any supervised tasks; 2) T5-v1.1$_{QA\text{-}FT}$, the resulting LM after fine-tuning T5-v1.1 on the three QA datasets,[10] which T5 is pre-trained on. The performance of the three LMs (T5-large size) on the ProPara dev set is presented in the top section of Table 3. We can see that T5 outperforms T5-v1.1 by a large margin, verifying that multi-task learning on out-of-domain non-entity-tracking tasks can benefit entity tracking. In addition, the advantage of T5 over T5-v1.1$_{QA\text{-}FT}$ indicates that knowledge transfer can cross the task boundaries with T5's encoder-decoder architecture.

Task Formulation We compare our QA formulation with two other task formulations, proposed in recent work, for T5. The first formulation is called "*step-input*" (Gupta and Durrett, 2019a; Amini et al., 2020), where each pair of the query entity e and procedure step t is formulated as one instance. Here, the state prediction is formulated as a classification problem, where the entity name is appended to the input, and no candidate answers are provided. Moreover, the procedure is trimmed until step t to specify the step index in the input. The second formulation is called "*process-input*" (Zhang et al., 2021; Gupta and Durrett, 2019b), where the model predicts entity states or locations in all steps in one instance. The input is the concatenation of entity e and the full procedure, and the model decodes

---

[6] `https://drive.google.com/drive/folders/1PYGLe7hSoCYfpKmpPumeTy6jmPyONGz4`

[7] `http://www.ffts.com/recipes.htm`

[8] `https://www.tensorflow.org/datasets/catalog/c4`

[9] `https://huggingface.co/docs/transformers/model_doc/t5v1.1`

[10] The three datasets include MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018), and BoolQ (Clark et al., 2019)

	P	**R**	**F1**
MEET (ours)	77.3	71.1	74.1
Multi-task Learning			
T5-v1.1	76.6	64.9	70.3
T5-v1.1$_{QA\text{-}FT}$	76.3	65.8	70.7
Task Formulation			
Process-level	89.3	30.2	45.1
Step-level	76.7	61.3	68.1
Decoding Strategy & Model Size			
CRF-normal	75.0	72.8	73.8
T5-base	76.8	68.2	72.2

Table 3: Analysis and ablation study on ProPara (dev set results). Top: Comparison of different backbone LMs to investigate the impact of multi-task learning. Middle: Comparison of different task formulations. Bottom: Ablation on decoding strategy and model size. Multi-task learning leads to a better entity tracking model, especially with the QA formulation and mention-guided decoding.

entity states and locations in all steps sequentially. The results of two new formulations are presented in the middle of Table 3. Our proposed QA formulation outperforms the other two formulations by a large margin. Detailed analyses of formulation comparison can be found in Appendix D.

Decoding Strategy & Model Size The ablation study on decoding strategy and model size is shown at the bottom section of Table 3. Clearly, our proposed "mention-guided" decoding strategy, as well as using a larger LM as the backbone, contribute to the success of MEET.

5 Conclusion

We presented MEET, a T5-based entity tracking approach. This approach includes our newly proposed QA fine-tuning formats and a customized decoding strategy so that it can effectively encode the flow of events in the procedural text while following structural constraints. The state-of-the-art performances on two benchmark datasets demonstrate the effectiveness of MEET, and further analyses verify that multi-task learning on out-of-domain tasks can be beneficial for entity tracking.

Limitations

This paper demonstrates that multi-task learning on a combination of general domain datasets can effectively improve the model's understanding of the procedural text. However, the precise source dataset responsible for this improvement remains uncertain, making it an avenue for future research to investigate more efficient knowledge transfer through the identification of the most pertinent source dataset. Moreover, the pipeline structure of MEET may limit its practical utilization. As such, future work could consider incorporating our proposed mention-guided decoding strategy into the end-to-end training of the multi-task learning model.

References

Aida Amini, Antoine Bosselut, Bhavana Dalvi, Yejin Choi, and Hannaneh Hajishirzi. 2020. Procedural reading comprehension with attribute-aware context flow. In *Automated Knowledge Base Construction*.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. Ext5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*.

Fan Bai, Alan Ritter, Peter Madrid, Dayne Freitag, and John Niekrasz. 2022. SynKB: Semantic search for synthetic procedures. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 311–318, Abu Dhabi, UAE. Association for Computational Linguistics.

Fan Bai, Alan Ritter, and Wei Xu. 2021. Pre-train or annotate? domain adaptation with a constrained budget. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5002–5015, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *Proceedings of the 6th International Conference for Learning Representations (ICLR)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604, New Orleans, Louisiana. Association for Computational Linguistics.

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

Aditya Gupta and Greg Durrett. 2019a. Effective use of transformer networks for entity tracking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 759–769, Hong Kong, China. Association for Computational Linguistics.

Aditya Gupta and Greg Durrett. 2019b. Tracking discrete and continuous entity state for process understanding. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 7–12, Minneapolis, Minnesota. Association for Computational Linguistics.

Hao Huang, Xiubo Geng, Jian Pei, Guodong Long, and Daxin Jiang. 2021. Reasoning over entity-action-location graph for procedural text understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5100–5109, Online. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Nghia T. Le, Fan Bai, and Alan Ritter. 2022. Few-shot anaphora resolution in scientific protocols via mixtures of in-context experts. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2693–2706, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Eric Nyberg, and Alessandro Oltramari. 2022. Coalescing global and local information for procedural text understanding. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1534–1545, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Masaaki Nagata, Katsuki Chousa, and Masaaki Nishino. 2020. A supervised word alignment method based on cross-language span prediction using multilingual BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 555–565, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hossein Rajaby Faghihi and Parisa Kordjamshidi. 2021. Time-stamped language model: Teaching language models to understand the flow of events. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4560–4570, Online. Association for Computational Linguistics.

Qi Shi, Qian Liu, Bei Chen, Yu Zhang, Ting Liu, and Jian-Guang Lou. 2022. Lemon: Language-based environment manipulation via execution-guided pre-training.

Ronen Tamari, Fan Bai, Alan Ritter, and Gabriel Stanovsky. 2021. Process-level representation of scientific protocols with interactive annotation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2190–2202, Online. Association for Computational Linguistics.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 57–66, Brussels, Belgium. Association for Computational Linguistics.

Jizhi Tang, Yansong Feng, and Dongyan Zhao. 2020. Understanding procedural text using interactive entity networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7281–7290, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *EMNLP*.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *ArXiv*, abs/1810.12885.

Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang. 2021. Knowledge-aware procedural text understanding with multi-stage training. In *Proceedings of the Web Conference 2021*, WWW '21, page 3512–3523, New York, NY, USA. Association for Computing Machinery.

Wanjun Zhong, Duyu Tang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. A heterogeneous graph with factual, temporal and logical knowledge for question answering over dynamic contexts. *CoRR*, abs/2004.12057.

Dataset	Statistics	Train	Dev	Test	Total
Recipes	# procedures	693	86	87	866
	Avg. steps / proc.	8.8	8.9	9.0	8.8
	Avg. entities / proc.	8.6	8.8	8.5	8.6
ProPara	# procedures	391	43	54	488
	Avg. steps / proc.	6.8	6.7	6.9	6.8
	Avg. entities / proc.	3.8	4.1	4.4	3.9

Table 4: Statistics of Recipes and ProPara.

	P	R	F1
implicit	37.4	23.2	28.3
explicit	**68.3**	**72.4**	**70.2**

Table 5: MEET's sentence-level performance (before applying offline CRF) on *implicit* and *explicit* steps (where the query entity is explicitly mentioned). Clearly, MEET makes more accurate predictions on *explicit* steps.

A Fine-tuning Formats for T5

A.1 State Prediction (Multi-choice QA)

Input:

```
What is the state of water in step 2?
(a) create (b) ... (f) move
step 1: Water flows downawards thanks
to gravity. step 2: Enters the dam at
high pressure. step 3: The moving water
spins the turbines in the power plant ...
step 6: The water leaves the dam at the
bottom.
```

Output:

```
move
```

A.2 Location Prediction (Extractive QA)

Input:

```
Where is water located in step 2?
step 1: Water flows downwards thanks to
gravity. step 2: Enters the dam at high
pressure. step 3: The moving water spins
the turbines in the power plant ... step
6: The water leaves the dam at the bottom.
Other locations: none, unknown.
```

Output:

```
dam
```

B Dataset

For ProPara (Dalvi et al., 2018), following Ma et al. (2022), the state prediction task includes six candidate states (`Outside_Before`, `Create`, `Destroy`, `Move`, `Exist` and `Outside_After`). For Recipes (Bosselut et al., 2018), each ingredient has two possibles states (`Exist` or `Absence`) in each step of the recipe. Full data statistics on two datasets are presented in Table 4.

C Evaluation

Sentence-level evaluation This evaluation measures the following questions for each target *entity*:

- **Cat-1:** Is *entity* created (destroyed, moved) in the process?
- **Cat-2:** When (step #) is *entity* created (destroyed, moved)?
- **Cat-3:** Where (location) is *entity* created (destroyed, moved to/from)?

Further, the F_1 scores of the three questions are aggregated with micro/macro averages.

Document-level evaluation It measures the four questions below for each paragraph:

- What are the *input* entities to the process?
- What are the *output* entities of the process?
- What entity *conversions* occur, when (step #), and where (location)?
- What entity *movements* occur, when, and where?

The macro average of the F_1 scores of these four questions will be used as the final score.

Table 6 provides a comprehensive comparison of past work on the ProPara dataset, including both document-level and sentence-level evaluations.

D Analysis of Formulation Comparison

When compared with the "*step-input*" formulation, the QA formulation allows the model to have the full context, and may take better advantage of LM's pre-training scheme (Li et al., 2019; Nagata et al., 2020). The "*process-input*" formulation works the worst in this comparison. With qualitative analyses, we find that it suffers from error propagation due to its autoregressive decoding, so future work may explore incorporating structural decoding (Tandon et al., 2018) into T5.

Model	**Document-level**			**Sentence-level**				
	P	**R**	**F1**	**Cat-1**	**Cat-2**	**Cat-3**	**macro**	**micro**
NCET (Gupta and Durrett, 2019b)	67.1	58.5	62.5	73.7	47.1	41.0	53.9	54.0
IEN (Tang et al., 2020)	69.8	56.3	62.3	71.8	47.6	40.5	53.3	53.0
DYNAPRO (Amini et al., 2020)	75.2	58.0	65.5	72.4	49.3	44.5	55.4	55.5
ProGraph (Zhong et al., 2020)	67.3	55.8	61.0	67.8	44.6	41.8	51.4	51.5
TSLM (Faghini et al., 2021)	68.4	68.9	68.6	78.8	56.8	40.9	58.8	58.4
KOALA (Zhang et al., 2021)	77.7	64.4	70.4	78.5	53.3	41.3	57.7	57.5
REAL (Huang et al., 2021)	**81.9**	61.9	70.5	78.4	53.7	42.4	58.2	57.9
LEMON (Shi et al., 2022)	74.8	69.8	72.2	**81.7**	58.3	43.3	61.1	60.7
CGLI (Ma et al., 2022)	75.7	**70.0**	72.7	80.8	60.7	46.8	**62.8**	**62.4**
MEET (ours)	80.3	67.1	**73.1**	77.5	**61.0**	**49.6**	62.7	**62.4**

Table 6: Document-level and sentence-level evaluation results on ProPara test set.

Conversational Tree Search: A New Hybrid Dialog Task

Dirk Väth **Lindsey Vanderlyn** **Ngoc Thang Vu**
University of Stuttgart, Germany
`{vaethdk|vanderly|thang.vu}@ims.uni-stuttgart.de`

Abstract

Conversational interfaces provide a flexible and easy way for users to seek information that may otherwise be difficult or inconvenient to obtain. However, existing interfaces generally fall into one of two categories: FAQs, where users must have a concrete question in order to retrieve a general answer, or dialogs, where users must follow a predefined path but may receive a personalized answer. In this paper, we introduce Conversational Tree Search (CTS) as a new task that bridges the gap between FAQ-style information retrieval and task-oriented dialog, allowing domain-experts to define dialog trees which can then be converted to an efficient dialog policy that learns only to ask the questions necessary to navigate a user to their goal. We collect a dataset for the travel reimbursement domain and demonstrate a baseline as well as a novel deep Reinforcement Learning architecture for this task. Our results show that the new architecture combines the positive aspects of both the FAQ and dialog system used in the baseline and achieves higher goal completion while skipping unnecessary questions.

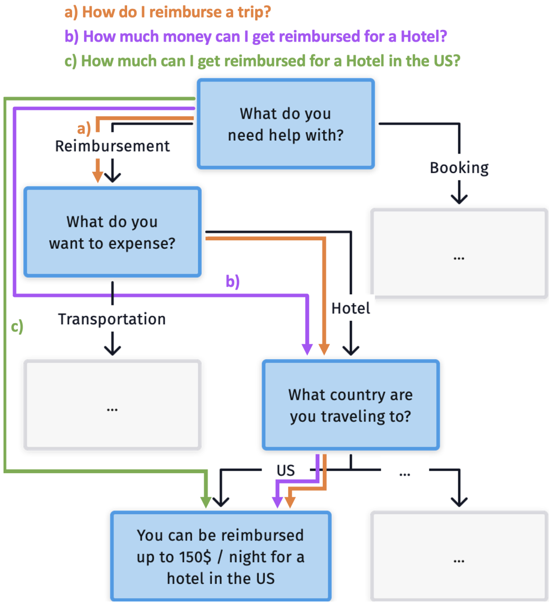


Figure 1: An example of the proposed task: Slice of a dialog tree (blue/gray nodes, black edges) showing how progressively more concrete questions could be answered. Question a) guiding a user with a general goal through the tree, b) asking only at nodes that need more clarification, and c) requiring no clarification and thus receiving a direct answer.

1 Introduction

Complex processes, e.g., healthcare, insurance, or travel reimbursement, can be challenging for users to navigate. FAQ and task-oriented dialog systems can provide immediate support, which is especially helpful as questions in these areas can be time sensitive, e.g., whom to contact in case of a lost passport. To support all types of users, however, such systems must be fast to satisfy those with more experience and thorough, so that those with less experience understand all steps needed to accomplish their goals. Above all, however, accuracy is critical in such susceptible domains.

FAQ systems can directly answer to user queries by matching them to predefined common question/answer pairs. It enables quick access to information while allowing subject experts to define system output and ensure it is factually correct (Wu et al., 2005). However, this approach does not allow for personalizing answers to a user's specific case, e.g., a specific user's per diem might depend on their destination and length of travel. Including information for all cases in one FAQ answer would make it long and hard to understand, while adding FAQs for each case, would make retrieval challenging. Additionally, as retrieval accuracy could be better (Thakur et al., 2021), the top answer might be incorrect, but displaying multiple answers makes the search process longer and transfers the responsibility of selecting the correct answer to the user. Finally, FAQ systems expect concrete information needs from users; but, new users may not be familiar enough with the complex process they wish to

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1264–1280
May 2-6, 2023

navigate in order to pose specific questions.

Dialog systems provide the opportunity for turn-based interaction, which can solve many of the shortcomings of FAQ systems (allowing process guidance, and shorter, more personalized answers). While automatically learning dialog behavior can be quicker and allow for more flexibility than designing a handcrafted system, such dialog systems rely on large amounts of data (Raghu et al., 2021), and their behavior can be challenging to control (Gao et al., 2018). As such, handcrafted dialog systems have generally still been preferred in low-resource settings, where there is not enough data to train a machine learning approach (Zhang et al., 2020), and in domains where it is essential that dialog designers can carefully control the system's behavior (Cohen, 2020). In these cases, it is common for non-technical experts to define the system behavior using a graphical interface (Shukla et al., 2020), resulting in a dialog tree. However, the structure of such systems can be rigid to traverse, making it less suitable for experienced users who want a specific question answered as fast as possible, rather than having to navigate through the whole tree until they reach the answer they were looking for.

In trying to implement a dialog system for travel reimbursement, we confirmed the shortcomings of both FAQ- and dialog-based approaches through pilot studies. As we are unaware of any system at the time of writing that can handle both forms of information-seeking behavior, we propose a new task called Conversational Tree Search (CTS). In this task, subject-experts can design arbitrarily complex dialog trees from which a dialog policy can be automatically trained to support both a guided dialog mode – where users with a vague information need can be guided by questions each turn until they reach their answer – and a free dialog mode – where users with a concrete information need can receive an answer as efficiently as possible (see Figure 1). In this way, CTS offers a hybrid solution where experts can still carefully control dialog structure and system responses while also profiting from the flexibility of a machine learning policy that can navigate the tree efficiently, only asking questions necessary to clarify information needs or personalize answers. Providing both dialog modes allows both novices and more experienced users to receive the correct level of support.

To solve our newly proposed task, we focus on the following research questions:

(RQ1) How can we develop a dialog policy that can navigate an expert-designed dialog tree, supporting both users with a specific information need and those with only a vague information need?

- **(RQ1.1)** Can the policy learn to differentiate between both dialog modes?
- **(RQ1.2)** Can this policy learn only to ask questions that are necessary to reach the user goal?

(RQ2) How robust is such a system to noise added to the encoding of the user input?

(RQ3) How well can such a system generalize to unseen utterances?

In the process of answering these research questions, we make the following concrete contributions: 1) We define a new task, CTS, and its evaluation criteria; 2) we collect and publish REIMBURSE, a low resource, real-world German language dataset for CTS in the domain of travel reimbursement, consisting of a dialog tree and user utterances; 3) we implement a domain-agnostic user simulator for CTS, using it to generate new dialogs based on the REIMBURSE data; and 4) we design a novel Reinforcement Learning (RL) architecture and train an agent to achieve significant performance increases over the baseline system on noisy and unseen data. Our data, simulator, and code are released[1] under an open source license to encourage more research into this task.

Our results show that an RL agent, based on our novel architecture, can learn to differentiate between both modes of interaction (guided and free) and skip unnecessary questions in free mode. Furthermore, the RL agent can significantly outperform the baseline system even in settings where Gaussian noise is applied to the user utterance encodings. Finally, the RL agent outperforms the baseline on dialogs that are generated based on unseen user utterances. Its success in this setting indicates an understanding of the user's text input, as the agent cannot exploit structural features, e.g., positional information.

2 Definition of Conversational Tree Search Task

The goal of this task is, given a dialog tree, to develop an interactive dialog system able to efficiently navigate a user to their goal by traversing the tree,

---

[1] `https://github.com/DigitalPhonetics/conversational-tree-search`

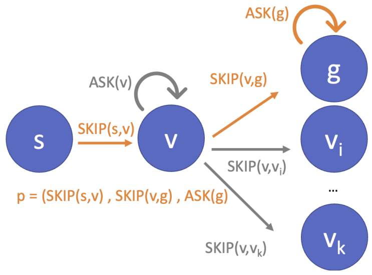


Figure 2: Example graph with dialog actions and path p from start node s to goal node g. As $ASK-g \in p$, this represents a dialog where the user reached their goal.

and only outputting questions which are needed to clarify the user's intention and personalize the output accordingly. In order to accomplish this, a policy has two types of actions it can choose at any node, either to ASK, which represents outputting the system text associated with that node, or to $SKIP$, which represents transitioning to a connected node without outputting the text at the current node.

2.1 Formal Task Description

Given is a dialog tree represented as (cyclic) directed Graph $G = (V, E)$, where the outgoing edges of node v include the self-transition $ASK(v) = (v, v)$, combined with the edges to neighbours of v ($v_i, \dots, v_k \subseteq V$): $E(v) = \{ASK(v), SKIP(v, v_i), \dots SKIP(v, v_k)\}$. Here, $SKIP(v, v_i) = (v, v_i)$ is an edge to a direct neighbour v_i of v.

Starting from a node $s \in V$, the goal is to find a path $p = (ASK(s), SKIP(s, v_i), \dots, ASK(g))$ to a given user goal node $g \in V$ (see Figure 2) only described by user utterances U, subject to $ASK(g) \in p$ (the goal node should be presented to the user) while at the same time minimizing $|\{ASK(v) | v \in V \wedge ASK(v) \in p\}|$ (the amount of asked questions, i.e., perceived length to the goal, should be as few as possible).

Here, the edges E also define the system's dialog actions, where $ASK(v)$ will output the text associated with the node v. Depending on the node type, this can either output information to the user or extend the user utterance history U as $U' = U \cup \{u_v\}$ by asking a question. The action $SKIP(v, v')$ will skip from v to v' without outputting anything to the user. Each node v is associated with text representing a system utterance and a node type (e.g. question). $SKIP$-actions are labeled by domain experts with a user answer prototype, which are used to compare user utterances to during dialog. For more information on node and edge data, see Section 4.

In order to address both user interactions (general exploration and specific questions), we define two goal settings within this task framework:

Guided Dialog Here, we define guided dialog as interactions where the user has a vague information goal and rather than posing a concrete question, would like to explore the information available in the dialog graph. Therefore, rather than having one static goal, the user decides on a new goal every turn, i.e., each turn their new goal will be to reach the node connected to the edge associated with the answer they give to the system question. Thus, a guided dialog can be seen as having turn-by-turn goals $(g, g', g'', \dots) \subset V$, meaning that from any node g, the next goal g' is an immediate neighbour of g, i.e., $SKIP(g, g') \in p$.

Free Dialog Free dialog in contrast, considers the case that the user has expressed a concrete information goal, which may or may not require later clarification. In this form of dialog, rather than focusing on choosing the next node from the set of neighbours, the dialog system's goal is to help the user fulfill their information need as quickly as possible. To this end, if the system is not sure about an upcoming decision, it may choose to ask for relevant information, thereby increasing its understanding of the user's goal. Thus, a free dialog has only one goal $g \in V$ and each turn serves to clarify the goal or skip closer to the answer.

2.2 Evaluation Objectives

We evaluate the path p taken by the policies according to the following criteria:

1. Task success, i.e., the goal node ($g \in V$) text is outputted to the user: $ASK(g) \in p$

2. Skip ratio, i.e., the number of times two consecutive skip actions occur along the path $((SKIP(v, v_i), SKIP(v_i, v_j)) \in p)$ divided by the length of the path

For guided dialog, the objective is to maximize task success while minimizing the skip ratio. In free mode, the objective is to maximize both task success and the skip ratio.

3 Methods

We develop and release[1] a baseline system, an RL-agent, and a domain-agnostic user simulator for training and evaluating dialog policies on the new CTS task.

3.1 User Simulator

The user simulator generates new dialogs for both interaction modes (with 50% probability of starting in either mode), which can be used to train and evaluate an RL-agent. The simulator is rules-based and each turn can respond to the dialog agent, either by asking a question (initial turn) or responding to a follow-up question (subsequent turns) using text from the available paraphrases (Appendix A). Dialogs are generated by choosing random goals and constraints. As its behavior only depends on an interchangeable dialog tree, the simulator remains domain-agnostic.

To represent exploratory users (guided mode), the simulator randomly samples a neighboring node as a new goal after each node transition. This mimics a user whose goal is only that the system correctly understands their current input and takes them to the associated next node. To support user exploration of the domain, skipping over nodes should be avoided, so asking after skipping to a new node – rather than skipping twice in a row – is rewarded (+2), as is correctly skipping after asking (+3). The initial utterance is the user answer to the start node question, subsequent utterances are given by (paraphrases of) answers leading to the next goal node.

To represent users with a concrete information goal (free mode), the simulator randomly selects a single, static goal and constraints for the entire dialog. A random node (with at least one associated FAQ question) is chosen from the graph as the user goal and one of the FAQ questions associated with that node is chosen as the initial user utterance. The simulator then finds a valid path from the start node to the goal node, saving the user answers along that path. Each time the dialog agent asks a question from a node along the goal path, the simulator can use the stored answers to respond. In case the node is not along the goal path, a random answer will be chosen to continue the dialog. To discourage long dialogs, each turn is given a small negative reward (-1). To discourage asking unnecessary questions, any questions that are not part of the stored goal path are punished (-4). Finally, reaching the goal node and outputting its content is given a reward of $4 \times tree\_depth$ (e.g. in the REIMBURSE dataset: reward of +128).

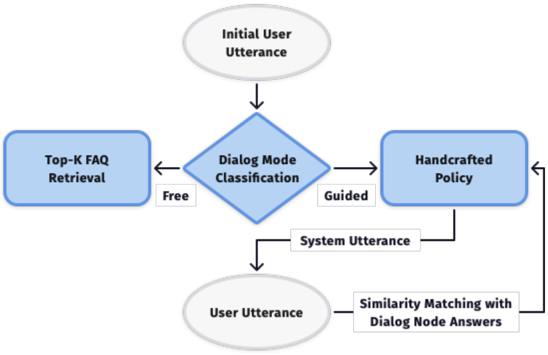


Figure 3: Baseline architecture: Combines an FAQ retrieval system (free mode) and a handcrafted dialog system (guided mode), with a classifier deciding which policy is active based on the input in the first turn.

A dialog will stop after: 1) reaching 50 dialog turns, 2) presenting the user goal, or 3) presenting the same node 3 times (user patience). To simulate unseen text input, noise can be added to user utterances in the form of a normal distribution around the original utterance encoding vector $\mathbf{u}$, using a percentage n of u as standard deviation: $\mathcal{N}(\mathbf{u}, n|\mathbf{u}|)$.

3.2 Baseline

For the baseline system (see Figure 3), we combine an FAQ retrieval system (free mode) and a handcrafted dialog system (guided mode) together, training a classifier to decide which policy is active based on the input in the first turn.

To train the dialog mode classifier, we fine-tune a German BERT model (Chan et al., 2020), providing a user utterance and the associated node text as inputs (see Appendix D). In free mode, a state-of-the-art similarity model (Reimers and Gurevych, 2019) is used to compare the first user utterance to all nodes in the dialog tree, directly outputting the most similar match to the user. In guided mode, node text is outputted to the user at each node. Their response is then compared to the prototypical answers for that node (using the same similarity model), and the policy then skips to the node connected by the most similar prototypical answer.

3.3 Reinforcement Learning Model

We propose a novel RL-architecture based on Dueling Network Architectures (DDQN) (Wang et al., 2016). For improved stability and convergence speed, we integrate Munchausen Reinforcement Learning (Vieillard et al., 2020), Double Q-Learning (van Hasselt et al., 2016), and Hindsight Experience Replay (Andrychowicz et al., 2017) in conjunction with Loss-Adjusted Prioritized Experience Replay (Fujimoto et al., 2020).

As a deviation from conventional DQN-based (Mnih et al., 2013) algorithms, we re-parameterize the usual network structure $\mathbf{Q} : \mathrm{s}, \boldsymbol{\theta} \mapsto \mathbb{R}^{|A|}$ to $Q : \mathrm{s}, a_i, \boldsymbol{\theta} \mapsto \mathbb{R}$, where s is a state vector, $\boldsymbol{\theta}$ represents the trainable network weights, and a_i is the i-th action in action space A with $|A|$ discrete actions. Thus, our architecture has one output node instead of $|A|$ output nodes (see Figure 4). Instead of performing one forward pass per state, we now perform $n(\mathbf{s})$ forward passes per state, where $n(\mathbf{s}) \leq |A|$ is the variable action count at state s, concatenating all outputs to obtain the full state-action vector $\mathbf{Q}(\mathrm{s}, \boldsymbol{\theta})$. By batching these forward passes, we achieve comparable runtime performance. The benefits of this approach are twofold: 1) It scales to an arbitrary number of actions without increasing the number of output neurons. 2) We can process action-specific inputs, e.g., action text (Figure 4 (a)), allowing the model to infer information from text rather than just exploiting action-space structure.

To keep the state values stable across different actions in the same state, we expand on the idea of DDQN (Wang et al., 2016), which processes inputs using shared layers (Figure 4 (b)) and calculates a state-value $V(\mathrm{s})$ and an advantage function $A(\mathrm{s}, a)$ with separate network layers on top of the shared layer. Here, we add separate layers for encoding action inputs (Figure 4 (c)), which we then concatenate with the shared layer output (Figure 4 (b)) in order to calculate $A(\mathrm{s}, a)$. $V(\mathrm{s})$ only receives inputs based on state s, and is thus decoupled from action-related inputs. We found this stabilized Q-values and performed better than the original algorithms in all our experiments (e.g. DDQN: 31.87% combined success on test data).

After experimenting with an external dialog mode classifier, we found that adding dialog mode classification as an auxiliary task improved our success metric. Therefore, we add an additional binary classification head to the output of the shared state layers from our enhanced DDQN to predict which dialog mode the current turn belongs to. Using cross entropy loss, we add this task as a hard-parameter sharing multitask learning objective (Caruana, 1993) in order to force the model to develop an understanding of the different tasks: $\mathcal{L} = \mathcal{L}_{\text{ddqn}} + \lambda \mathcal{L}_{\text{intent}}$.

4 REIMBURSE: A Dataset for CTS

For this task, we collect and publish a dataset from two sources: a dialog tree, defining the general system behavior, and a corpus of user utterances, with paraphrases for both user questions and responses to system questions. This dataset represents challenging real-world data, both in the complex structure of the dialog tree for this domain, and the real-world nature of the user utterances. Examples of user questions and user responses are provided in Appendix B.

Dialog Tree We provide a dialog tree for travel reimbursement, created by subject-area-experts using a graphical dialog designer tool. The dialog tree consists of four different types of nodes: 1) Dialog Nodes, defining a system question and possible user answers. 2) Variable Nodes, defining a system question and storing the user answer for later use in the beliefstate. 3) Information Nodes, defining information that the system should share with the user without expecting a user response. And 4) Logic Nodes, defining dialog flow based on logical conditions evaluated against beliefstate values, without outputting anything to the user. An example of each of these can be seen in Figure 6 in the Appendix.

In total, the dialog graph we use contains 123 total nodes (with a maximum depth of 32), of which, 79 are information nodes which contain answers to user questions and 23 are Dialog Nodes with system questions for the user. The maximum action count per node (directly connected nodes) is 14.

User Utterances We collected a real world corpus, called REIMBURSE. This corpus consists of 452 free dialog questions, each corresponding to an Information Node in the dialog tree, with an average of 5.72 paraphrases per Information Node. This was done through recording user utterances in two early pilot studies, and manual augmentation by two expert annotators. Synonyms for user responses to system questions were collected the same way, resulting in 408 total response paraphrases, on average 5.58 per user answer. For examples, see Appendix B.1.

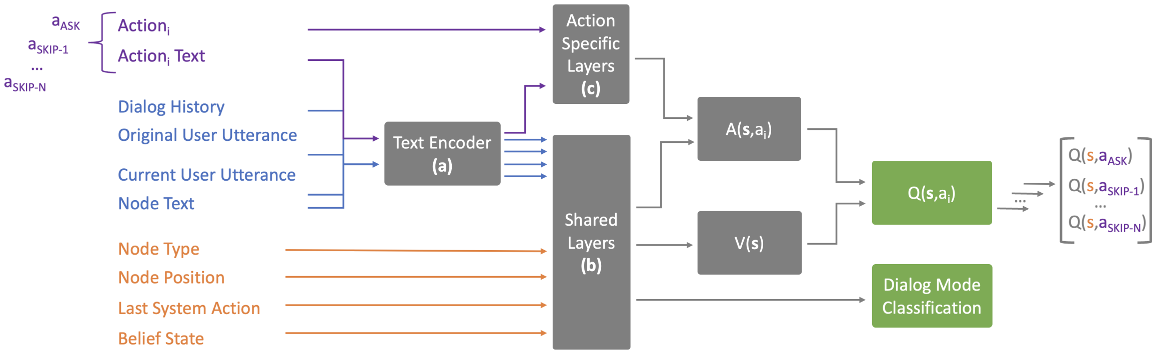


Figure 4: Proposed RL architecture: The state-value function $V(\mathbf{s})$ for state s is calculated from the shared layers (b), independent of action inputs. For the advantage function $A(\mathbf{s}, a)$, the outputs from shared layers (b) and action-specific layers (c) are combined. The final state-action-values Q are obtained by performing one forward pass per action a_i, each yielding a scalar $q_{\mathbf{s},a_i}$, which are then concatenated by state into a vector with one value per action. Additionally, a binary dialog mode classifier is added on top of the shared layers (b).

5 Experimental Setup

The RL agents receive the following inputs: beliefstate, last system action type, action index, and current node type encoded by one-hot vectors, current node position as a binary tree encoding (Shiv and Quirk, 2019), the dialog history, initial user utterance, current user utterance, current node text and the node's prototypical answer texts encoded by either an English-German RoBERTa (Liu et al., 2019) [2] or sentence transformers (Reimers and Gurevych, 2019) model. The model is trained against the user simulator using the Adam optimizer (Kingma and Ba, 2015) with a learning rate of $1e^{-4}$. We set the maximum number of training dialog turns to 1.5M and save the model which performs best in evaluation (performed every 10k turns with 500 simulated dialogs) for testing. Evaluation and testing are also performed against the user simulator. However, in this mode agent exploration is disabled. For more details on model and training parameters, see Appendix D.

Testing is done on 500 simulated dialogs (in RQ3 generated from held out utterances). We measure success and skip ratios for guided and free mode separately and jointly. To check task understanding, we measure the dialog mode prediction consistency: we calculate the difference between the percentage of turns predicted for guided and free mode per dialog, then averaging across all dialogs.

5.1 RQ1: Task Performance

To understand how well the RL agent learns to support both dialog modes, we measure combined task success as well as success on each individual mode (how often user reaches goal), the mode-specific skip ratios (higher skip ratio desired for free, lower for guided mode), and the mode prediction and consistency measures (proxy for task understanding). All models are trained and tested against the user simulator generating dialogs from the same distribution, following standard RL procedure. We apply a noise level of $n = 10\%$ to encourage generalization.

5.2 RQ2: Task Performance in a Noisy Environment

To test how well our models perform in noisy environments, we test the success and mode classification F1 score of models trained in RQ1 on simulated dialogs with increasing noise levels. We test five times, reporting average results and statistically significant performance differences.

5.3 RQ3: Generalizing to New Data

In the previous experiments, we have followed the RL convention of testing our models on new dialogs generated from the same user simulator they trained on, with the same set of possible user utterances. However, much of the challenge in this domain comes from understanding text input. By exchanging the set of user utterances available to the simulator for testing, we explore how well the model can generalize to user inputs not seen during training. To this end, we split our corpus into a test/eval and a train set. The train/eval set contains 279 FAQ questions (3.5 questions per Information node) and 246 responses (3.4 paraphrases per response prototype). The test set contains 173 user

---

[2] https://huggingface.co/T-Systems-onsite/cross-en-de-roberta-sentence-transformer?doi=true

questions (2.2 questions per Information node) and 162 responses (2.2 paraphrases per response prototype). We then train and evaluate the best performing models from RQ1 and RQ2 on simulated dialogs generated from the train split. Finally, to measure how well they can generalize, we test the final performance on dialogs generated from the unseen paraphrases in the test split.

6 Results and Discussion

6.1 RQ1: Task Performance

To verify the user simulator, we evaluate it against our baseline model, without noise and only using the prototypical user answers from the dialog tree. This yields a success rate of 99.35% for the guided task and an F1-score of 1.0 for the dialog mode prediction, showing that the simulator works as expected. On the free task, it reaches 64.46% success. Given the challenging nature of the dataset, it is to be excepted that top-1 retrieval is not perfect.

Just by adding user answer paraphrases and 10% noise, baseline performance drops substantially on all tasks (see Table 1), demonstrating the task difficulty w.r.t. a real-world scenario. Our RL agents significantly outperform the baseline (t-test $p < 3 \times 10^{-8}$) on all task success metrics, except the skip ratios (see Table 1). This shows our models are better able to learn the task compared to the baseline, especially under noisy text inputs, which we attribute to the improved generalization.

RQ1.1: Differentiating Between Dialog Modes Table 1 shows that both of our models learn to skip more frequently in free mode (e.g., 0.58) and less frequently in guided mode (e.g., 0.07). This indicates that multitask learning helped understand the user's intended mode, as we would otherwise not see a difference in skipping behaviour. Additionally, the classification consistency is 1.0, indicating stable task understanding. Jointly learning intent prediction also improves dialog mode classification compared to using a pre-trained classifier.

RQ1.2: Asking Only Necessary Questions While the baseline has, by construction, perfect skip ratios on both free (1-step retrieval) and guided tasks (no skipping), the skip ratios in Table 1 demonstrate that our models learn to ask questions in both modes. In free mode, we see that asking some questions helps differentiate the user goal, as we obtain much better task success than the 1-step baseline. High skip ratios (e.g., 0.58) in this mode show the model still skips unnecessary nodes.

6.2 RQ2: Task Performance in a Noisy Environment

Figure 5 shows our models are able to handle high levels of noise: performance only decreases rapidly after 100% noise. Our models demonstrate a significant performance increase over the baseline at all noise levels (t-test $p < 0.003$). Additionally, we find that dialog mode understanding is robust w.r.t. input noise (dialog mode classification F1 stays unchanged at 1.0 after rounding, e.g. consistency only drops from 1.0 without noise to 0.83 at the highest noise setting). The robustness to noise suggests that the regularization techniques are effective.

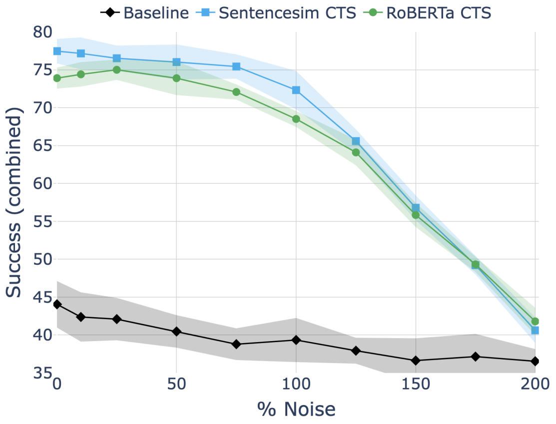


Figure 5: Task performance for different noise levels on user input (drawn from a normal distribution around the original text encoding vector $\mathbf{u}$, using a percentage n of $\mathbf{u}$ as standard deviation: $\mathcal{N}(\mathbf{u}, n|\mathbf{u}|)$.

6.3 RQ3: Text Understanding and Generalization to New Data

When only training on a fraction of the data and testing on the unseen split, we observe some performance drops for the baseline and all RL agents, compared to the setting from RQ1, as well as between evaluation and test performance in RQ3 (see Table 2). The performance drop in the RL agents can likely be explained by the drop in the dialog mode classification score (e.g., from 1.0 in RQ1 to 0.85). This paired with high dialog mode prediction consistency (0.96, 1.0) means the models more often classified the dialog mode incorrectly and could not recover during a dialog. This misclassification likely also explains increased skipping in guided mode (e.g., from 0.07 in RQ1 to 0.19).

However, even in this more challenging setting, our models demonstrate much better combined task

Model	**Success (guided)**	**Skip Ratio (guided)**	**Success (free)**	**Skip Ratio (free)**	**Success (combined)**	**Dialog Mode Prediction F1**	**Dialog Mode Prediction Consistency**
Baseline	64.71%	**0.0**	22.84%	**1.0**	43.10%	0.83	**1.0**
RoBERTa CTS	75.12%	0.10	**74.20%**	0.55	74.40%	**1.0**	**1.0**
Sentencesim CTS	**84.77%**	0.07	72.00%	0.58	**77.16%**	**1.0**	**1.0**

Table 1: Average model performance on data with 10% noise. Both RL agents are able to significantly outperform the baseline (t-test: $p < 3 \times 10^{-8}$) in terms of success (guided, free, and combined). The RL agents learned to distinguish between dialog modes, skipping more than half of nodes in free mode and less than 10% in guided.

Model	**Success (guided)**	**Skip Ratio (guided)**	**Success (free)**	**Skip Ratio (free)**	**Success (combined)**	**Dialog Mode Prediction F1**	**Dialog Mode Prediction Consistency**
Evaluation Data							
Baseline	81.05%	**0.0**	25.54%	**1.0**	55.40%	0.95	**1.0**
Sentencesim CTS	82.96%	0.13	71.19%	0.54	76.01%	**1.0**	**1.0**
RoBERTa CTS	**85.36%**	0.07	**86.18%**	0.53	**85.81%**	**1.0**	**1.0**
Test Data							
Baseline	65.52%	**0.0**	20.38%	**1.0**	42.05%	0.85	**1.0**
Sentencesim CTS	**74.76%**	0.19	54.38%	0.61	62.54%	**0.88**	0.96
RoBERTa CTS	57.10%	0.25	66.78%	0.53	**62.58%**	0.85	**1.0**
- Action Positions	46.70 %	0.29	**66.79%**	0.55	57.59%	0.82	0.99
- Action Text	57.63%	0.23	55.31%	0.57	55.28%	0.84	0.98
- Node Text	39.82%	0.20	55.34%	0.54	47.95%	0.83	0.96
- Node Positions	54.98%	0.20	55.40%	0.59	55.21%	0.81	0.99
- Node Type	50.67%	0.20	64.41%	0.56	58.39%	0.84	0.98
- Mode Prediction	43.61%	0.23	60.38%	0.59	52.33%	n/a	n/a
- Beliefstate	62.26%	0.21	42.66%	0.57	51.05%	0.82	0.99

Table 2: Model performance on simulated dialogs generated from dataset splits, and skip ratios (skipped dialog nodes w.r.t. dialog length). Here, higher skip ratios are better for the free setting (looking for a direct answer), but lower ones are preferable for guided mode (exploratory dialogs). Rows underneath *RoBERTa CTS* for the data represent input ablations, i.e. the listed input was removed from the model.

success than the baseline. This is particularly interesting as utterances in the test data were never seen in training, meaning the model could not have merely exploited structural properties of the data, but rather must have learned text understanding to solve the task. For example (see Appendix C.2), the model can understand that when a user asks what to do after an earthquake, they need the emergency help number and skips directly to that node, even though nothing about earthquakes appeared in training. The system also learns to recognize when information is missing from a user question and ask for additional information, e.g., if the business trip took place in or outside the home city, even for new questions (see Appendix C.3).

To better understand how the model processed inputs, we performed an ablation study (Table 2) of the RoBERTa agent. From the success rates without either action text or action position, we see that providing action context in input space increases model performance, validating our new architecture.

In free mode, when node or action text are removed, there is a large drop in model performance (about 10%), which is not as severe if instead action positions are left out. This suggests that the model learns to understand text as it can correlate user utterances to text from nodes and actions in the dialog graph, even with noise and synonyms not seen during training. Node position intuitively simplifies path finding, which explains a drop in performance when leaving it out. It also makes sense then that removing the dialog history has a similar effect, as it encodes path information in textual form. We conclude from this that both textual and positional information are needed for this task and better understanding of either input could likely boost performance.

In the guided mode, we observe that while node text was very important (nearly 20% drop), removing action text had almost no effect. This might relate to the text length associated with each action, and the length of user input per node (usually much less information than in the free mode questions). As user input was often one to three words long, it could have been easier for the model to learn to correlate the user input to an action position rather than action text, especially given that BERT-based language models rely on context. Thus, improving focus on action text in the guided mode is an area for future investigation. Removing the node position or dialog history had a minor impact, which agrees well with the objective of the guided mode: it should not rely much on path planning, instead user input should be considered at most nodes to move the dialog along.

In general, dialog mode prediction improves performance in both dialog modes. Without this, the agent is able to learn to skip more in free mode and less in guided, but success drops. In the multitask learning setting, we force the model to learn to output a consistent prediction across a dialog (Table 2 consistency values are close to 1.0 for all multitask models), which may explain the higher success.

7 Related Work

Data Efficient Task Oriented Dialog Systems In some domains, e.g., recommendation, the number of actions for a dialog system can be quite large prompting new architectures which group the actions into a tree structure to reduce the search space (Chen et al., 2019; Montazeralghaem et al., 2021). While the information seeking dialogs have different characteristics, including a tree structure for the data, can help experts maintain control of the system and reduce the action search space.

Task Oriented Systems with Expert involvement In many domains, it is crucial domain experts maintain control of dialog flow to ensure correctness of system outputs. An early approach (Williams, 2008) involved a hand-crafted dialog policy which output the allowed actions at a time step and a POMDP policy then chose the optimal action from this set. More recent approaches, e.g., hybrid code networks (HCN) have expanded on this idea for neural systems (Williams et al., 2017; Liang and Yang, 2018; Razumovskaia and Eskenazi, 2019), where action space can be constrained using masks. Shukla et al. (2020) extend the HCN approach by automatically converting expert designed dialog trees into hybrid code networks (HCN). This approach increases explainability of system behavior, but doesn't provide a mechanism for skipping portions of a dialog irrelevant to a user, falling short of addressing free dialog.

Knowledge Augmented Dialog Systems The DSTC10 challenge (Kim et al., 2022) introduced a new track for knowledge grounded dialog, combining task-oriented dialog and FAQ style questions answered from unstructured documents. Such hybrid dialog systems help answer free form user queries, but still require a concrete question, and do not allow precise control over generated answers.

Another recent approach (Raghu et al., 2021) proposed grounding dialog systems on troubleshooting flow charts and FAQ questions. Here, the dialog system fuses knowledge from FAQs with an under-specified flow chart to generate an appropriate response to a user input. However, dialogs appear limited to short flow charts, and response generation is again outside the designer's control.

Clarifying Questions and Conversational Search Conversational Search and clarifying questions are two intertwined areas of research which focus on disambguating free form user queries. The goal of these tasks is to generate one or more questions to narrow down a user's information need (Zamani et al., 2020) in order to retrieve relevant documents. To improve conversational search performance, much work has focused on ranking candidate questions (Kumar et al., 2020) and leveraging the user answers (Bi et al., 2021). However, neither of these tasks do not allow for human oversight of the questions that will be asked, and currently do not consider any structure for the organization of questions.

8 Conclusion and Future Work

In this work, we introduced a new task, CTS which combines the positive aspects of both an FAQ retrieval and task-based dialog system, along with a real-world German language dataset, REIMBURSE, a domain-agnostic user simulator, and baseline system based on state-of-the-art language models. Furthermore, we demonstrated a novel, scalable RL architecture, showing significant performance improvements over the baseline. Our models are able to handle high levels of noise in the input data and demonstrate the ability to learn text-based information from user utterances, as seen by their ability to handle unseen data. In the future, it would be interesting to explore data augmentation for generating unseen utterances as well as to explore methods, particularly for guided mode dialogs, to encourage the model to rely even more information from text. We also hope that the community helps in exploring performance on additional languages and task instances.

9 Limitations

We recognize the following limitations in our work:

- When testing on unseen real world data rather than the noisy simulation, we notice a performance drop between the evaluation and test results. As we are working in a very low resource setting and using a challenging real-world dataset, it is difficult to state with certainty if this is a result of the challenge inherent to the dataset or a weakness of the models to generalize.
- We were not able to perform testing in an interactive user study, but rather tested against simulated dialogs generated from pre-collected real-world utterances. While these were collected from real users of the dialog system, a user study would be valuable to investigate how the system handles input in a live setting.
- As we only collected a single dataset for the new CTS task, we have not investigated domain generalizability.

References

Marcin Andrychowicz, Dwight Crow, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. pages 5048–5058.

Keping Bi, Qingyao Ai, and W. Bruce Croft. 2021. Asking clarifying questions based on negative feedback in conversational search. In *ICTIR '21: The 2021 ACM SIGIR International Conference on the Theory of Information Retrieval, Virtual Event, Canada, July 11, 2021*, pages 157–166. ACM.

Rich Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Machine Learning, Proceedings of the Tenth International Conference, University of Massachusetts, Amherst, MA, USA, June 27-29, 1993*, pages 41–48. Morgan Kaufmann.

Branden Chan, Stefan Schweter, and Timo Möller. 2020. German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Haokun Chen, Xinyi Dai, Han Cai, Weinan Zhang, Xuejian Wang, Ruiming Tang, Yuzhou Zhang, and Yong Yu. 2019. Large-scale interactive recommendation with tree-structured policy gradient. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3312–3320. AAAI Press.

Philip R. Cohen. 2020. Back to the future for dialogue research. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 13514–13519. AAAI Press.

Scott Fujimoto, David Meger, and Doina Precup. 2020. An equivalence between loss functions and non-uniform sampling in experience replay.

Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1371–1374, New York, NY, USA. Association for Computing Machinery.

Seokhwan Kim, Yang Liu, Di Jin, Alexandros Papangelis, Behnam Hedayatnia, Karthik Gopalakrishnan, and Dilek Hakkani-Tür. 2022. Knowledge-grounded task-oriented dialogue modeling on spoken conversations track at dstc10.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Vaibhav Kumar, Vikas Raunak, and Jamie Callan. 2020. Ranking clarification questions via natural language inference. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2093–2096. ACM.

Weiri Liang and Meng Yang. 2018. Hierarchical hybrid code networks for task-oriented dialogue. In *Intelligent Computing Theories and Application - 14th International Conference, ICIC 2018, Wuhan, China, August 15-18, 2018, Proceedings, Part II*, volume 10955 of *Lecture Notes in Computer Science*, pages 194–204. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing atari with deep reinforcement learning. volume abs/1312.5602.

Ali Montazeralghaem, James Allan, and Philip S. Thomas. 2021. Large-scale interactive conversational recommendation system using actor-critic framework. In *Fifteenth ACM Conference on Recommender Systems*, RecSys '21, page 220–229, New York, NY, USA. Association for Computing Machinery.

Dinesh Raghu, Shantanu Agarwal, Sachindra Joshi, and Mausam. 2021. End-to-end learning of flowchart grounded task-oriented dialogs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4348–4366, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Evgeniia Razumovskaia and Maxine Eskenazi. 2019. Incorporating rules into end-to-end dialog systems. In *Proc. 3rd NeurIPS Workshop on Conversational AI, Vancouver, Canada*, pages 1–11.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Vighnesh Shiv and Chris Quirk. 2019. Novel positional encodings to enable tree-based transformers. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Swadheen Shukla, Lars Liden, Shahin Shayandeh, Eslam Kamal, Jinchao Li, Matt Mazzola, Thomas Park, Baolin Peng, and Jianfeng Gao. 2020. Conversation Learner - a machine teaching tool for building dialog managers for task-oriented dialog systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 343–349, Online. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. In *35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks*.

Hado van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2094–2100. AAAI Press.

Nino Vieillard, Olivier Pietquin, and Matthieu Geist. 2020. Munchausen reinforcement learning.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1995–2003. JMLR.org.

Jason D Williams. 2008. Integrating expert knowledge into pomdp optimization for spoken dialog systems. In *Proceedings of the AAAI-08 Workshop on Advancements in POMDP Solvers*, volume 2, page 25.

Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 665–677, Vancouver, Canada. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chung-Hsien Wu, Jui-Feng Yeh, and Ming-Jun Chen. 2005. Domain-specific faq retrieval using independent aspects. *ACM Transactions on Asian Language Information Processing*, 4(1):1–17.

Hamed Zamani, Gord Lueck, Everest Chen, Rodolfo Quispe, Flint Luu, and Nick Craswell. 2020. MIMICS: A large-scale data collection for search clarification. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 3189–3196. ACM.

Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. 2020. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences*, 63(10):2011–2027.

A User Simulator

The (simplified) behavior of the user simulator is show below. To generate dialogs based on unseen utterances, one can simply replace the files containing questions and answers with new files containing different questions and answers than were present during the training process.

```
Require: questions ← List of Questions associated with dialog tree nodes
Require: answers ← List of Answer synonyms
Require: $N$ ← Number of dialogs to be simulated
Require: $G(V, E)$                                                    ▷ Dialog Tree
Require: $T$ ← Maximum number of turns per dialog
  $n \leftarrow 0$
  for $n < N$ do
      $v$ ← start node
      $m$ ← randomly choose GUIDED or FREE
      if $m =$ GUIDED then
          $g$ ← randomly choose neighbour(v)                ▷ Draw new goal in guided mode
          $e \leftarrow e(v, g) \in E$
          $u$ ← randomly choose from answers[e]
      else
          $g$ ← randomly draw $g \in V$ with |questions[g]| $\geq 1$      ▷ Draw new goal in free mode
          $p \leftarrow (e(v, v_1), \ldots, e(v_N, g)) \subset E$
          $u$ ← randomly choose from questions[g]
      end if
      $t \leftarrow 1$
      while $v \neq g \wedge t \leq T$ neighbours(v) $\neq \emptyset$ do          ▷ Simulate dialog
          $a$ ← next action from dialog policy, with inputs $v, u$
          if $a =$ ASK(v) then                                  ▷ Print / Ask information
              print $v$
              $e \leftarrow e(v, v')$ where $v'$ is g in guided mode or in free mode the next node after v in the path
              $u$ ← randomly choose from answers[e] if type(v) $\neq$ information, else $\emptyset$
          else if $a =$ SKIP(v,v') then                             ▷ Skip to next node
              $v \leftarrow v'$
              $u \leftarrow \emptyset$
          end if
          $r$ ← Calculate rewards
          Store $a, v', r$ in replay buffer
          if $m =$ GUIDED then
              $g$ ← randomly choose neighbour(v)        ▷ Draw followup goal in guided mode
              $e \leftarrow e(v, g) \in E$
          end if
          $t \leftarrow t + 1$
      end while
      $n \leftarrow n + 1$
  end for
```

B REIMBURSE: Dataset

B.1 Example User Utterances

B.1.1 Answer Paraphrases

Below is an example of the answer paraphrase data we collected. The subject-area expert defined a prototypical answer to the question and then additional paraphrases were collected (through pilot study and manual expansion). Although some are very close to the prototypical answer, there can also be a lot of variation.

Prototypical Node Answer: Mit Familie *(en: With my family)*
Associated Paraphrases:

- Zusammen mit meiner Frau *(en: Together with my wife)*
- Mit Mann und Kinder *(en: With husband and kids)*
- Ich und meine Angehörigen *(en: Me and my relatives)*
- Mit Begleitung *(en: With accompaniment)*

B.1.2 User Question Paraphrases

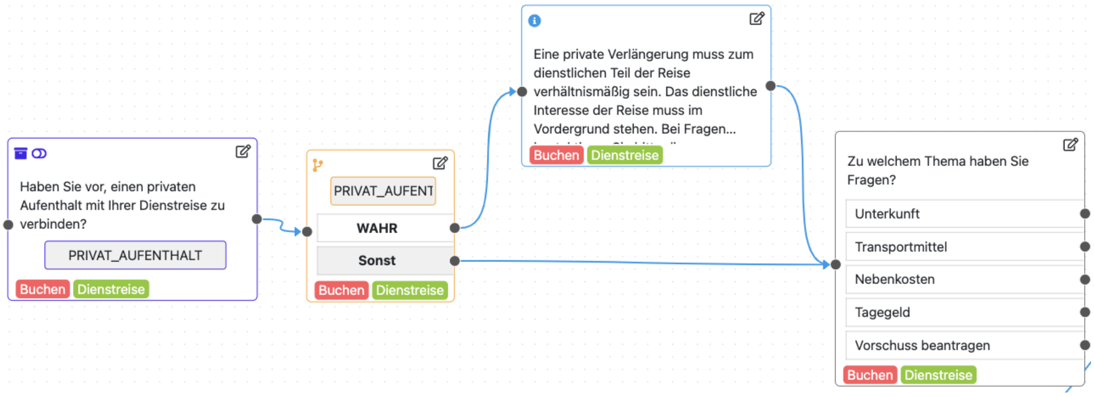
For each Information node in the graph, a set of paraphrased questions were collected.

Node Text: Eine private Verlängerung muss zum dienstlichen Teil der Reise verhältnismäßig sein. Das dienstliche Interesse der Reise muss im Vordergrund stehen. Bei Fragen kontaktieren Sie bitte die Reisekostenstelle. (*en: A personal extension must remain in proportion with the official part of the trip. The business benefit must remain the primary focus of the trip. In case of questions, please contact the business travel department.*)
Associated Questions:

- Darf ich meine Reise privat verlängern? *(en: Can i extend my trip privately?)*
- Wie lang darf der private Reiseanteil sein? *(en: How long is a private extension allowed to be?)*
- Kann ich länger bleiben als der dienstliche Teil meiner Reise? *(en: Can I stay longer than the official part of my trip?)*
- Darf ich nach meiner Reise Urlaub machen? *(en: Can I add vacation to the end of my trip?)*

B.2 Example Tree

Below is an example of each of the node types in the dialog editor. For Information nodes (blue) subject experts could also define associated FAQ questions to help the RL agent understand user user questions related to that node.



Haben Sie vor, einen privaten Aufenthalt mit Ihrer Dienstreise zu verbinden?

Eine private Verlängerung muss zum dienstlichen Teil der Reise verhältnismäßig sein. Das dienstliche Interesse der Reise muss im Vordergrund stehen. Bei Fragen...

Zu welchem Thema haben Sie Fragen?

Figure 6: Example of the dialog designer interface. From left to right: Variable Nodes (purple outline), which the designer can use to ask for information relevant for later decisions, Logic Nodes (orange outline) which control the flow of the dialog based on user responses stored in the Variable Nodes (here the two options are *"true"* and *"else"*), Information Nodes (blue outline) where the designer can specify information which does not need a user response, and Dialog Nodes (black outline) where the designer can specify system questions and expected user responses.

Figure 7 is a view of the full dialog tree used for experiments in this paper, consisting of 123 nodes.

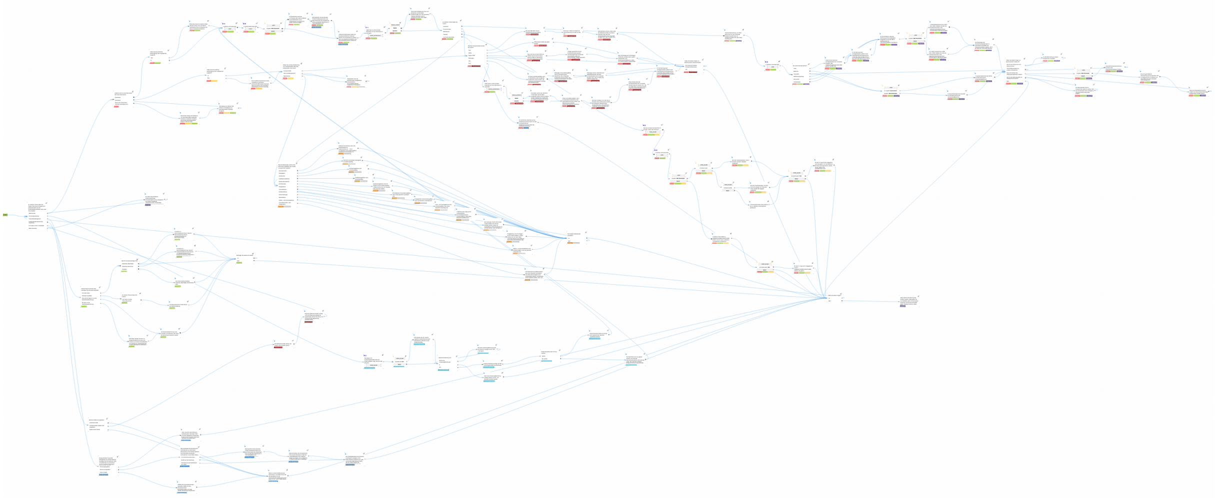


Figure 7: Overview of the dialog tree used for all experiments in this paper. The tree was designed by subject-area experts to help users better navigate the travel reimbursement process.

C Example Dialogs

In this section, we show parts of successful conversations between the simulator and the Sentencesim-CTS agent.

C.1 Guided Mode

SYSTEM: [ASK] Zu welchem Thema haben Sie Fragen? Sie können entweder eine passende Option aus der Vorschlagsliste anklicken oder unten Text eingeben. (*en: What topic do you have questions about? You can either click one of the suggestions or type your question.*)

USER: Ein Forschungssemester planen (*en:Plan a research semester*)

SYSTEM: *[SKIP TO] Wie lange ist ihr Forschungssemester (bitte mit Einheit angeben: Tage, Wochen oder Monate)?* (*en:How long will your research semester be (please use a unit: days, weeks, or months*?)

SYSTEM: [ASK] Wie lange ist ihr Forschungssemester (bitte mit Einheit angeben: Tage, Wochen oder Monate)? (*en:How long will your research semester be (please use a unit: days, weeks, or months*?)

USER: 2916 Sekunden (*en: 2916 Seconds*)

[*Environment action: Match logic node branch condition → Automatic SKIP TO: Bitte wenden Sie sich zwecks möglicher Unterkünfte bei einem Aufenthalt ab 4 Wochen an die Reisekostenstelle.*]

SYSTEM: [ASK] Bitte wenden Sie sich zwecks möglicher Unterkünfte bei einem Aufenthalt ab 4 Wochen an die Reisekostenstelle. (*en: Please contact the travel department about booking your accommodations for trips longer than 4 weeks.*)

SYSTEM: *[SKIP TO] Liegt eine Zuweisung vor?* (*en: Have you been reallocated?*)

SYSTEM: [ASK] Liegt eine Zuweisung vor? (*en: Have you been reallocated?*)

[*. . . CONTINUES FOR > 30 MORE TURNS*]

C.2 Free Mode - Direct Skipping to Goal

SYSTEM: [ASK] Zu welchem Thema haben Sie Fragen? Sie können entweder eine passende Option aus der Vorschlagsliste anklicken oder unten Text eingeben. (*en: What topic do you have questions about? You can either click one of the suggestions or type your question.*)

USER: Es gab ein Erdbeben, an wen soll ich mich wenden? (*en: There was an earthquake, who can I contact?*)

SYSTEM: *[SKIP TO] Welcher Notfall ist eingetreten?* (*en: What type of emergency has happened?*)

SYSTEM: *[SKIP TO] Was ist in einem Notfall auf einer Dienstreise zu tun? Bitte führen Sie die Notfallkarte mit der entsprechenden Notfall-Rufnummer stets mit sich: +123456789. Die Notfallkarte ist auch im Scheckkartenformat erhältlich. Hierzu können Sie sich bei der Reisekostenstelle oder an ihrem Institut erkundigen. Diese Rufnummer sollte nur für Notfälle genutzt werden. Die Weitergabe an Dritte ist untersagt. [. . .]* (*en: What should you do if an emergency occurs during a business trip? Please use the emergency card, which has the emergency telephone number +123456789. You can also get the emergency card in a platic card form. For more information, ask your institute or the business travel department. This number should only be used in emergencies. Passing it on to external people is strictly forbidden. [. . .]*)

SYSTEM: [ASK] Was ist in einem Notfall auf einer Dienstreise zu tun? Bitte führen Sie die Notfallkarte mit der entsprechenden Notfall-Rufnummer stets mit sich: +123456789. Die Notfallkarte ist auch im Scheckkartenformat erhältlich. Hierzu können Sie sich bei der Reisekostenstelle oder an ihrem Institut erkundigen. Diese Rufnummer sollte nur für Notfälle genutzt werden. Die Weitergabe an Dritte ist untersagt. [. . .] (*en: What should you do if an emergency occurs during a business trip? Please use the emergency card, which has the emergency telephone number +123456789. You can also get the emergency card in a platic card form. For more information, ask your institute or the business travel department. This number should only be used in emergencies. Passing it on to external people is strictly forbidden. [. . .]*)

C.3 Free Mode - Clarifying Question

SYSTEM: [ASK] Zu welchem Thema haben Sie Fragen? Sie können entweder eine passende Option aus der Vorschlagsliste anklicken oder unten Text eingeben. (*en: What topic do you have questions about? You can either click one of the suggestions or type your question.*)

USER: Muss ich etwas tun bevor ich eine Reise antreten darf? (*en: Is there anything I have to do before I can travel?*)

SYSTEM: *[SKIP TO] Handelt es sich um eine Dienstreise oder einen Dienstgang?* (*en: is this a trip within the city of employment or a trip outside of the city of employment?*)

SYSTEM: [ASK] Handelt es sich um eine Dienstreise oder einen Dienstgang? (*en: is this a trip within the city of employment or a trip outside of the city of employment?*)

USER: Eine dienstliche Reise (*en: A trip outside the city of employment*)

SYSTEM: [SKIP TO] *Haben Sie die schriftliche Genehmigung ihres Vorgesetzten eingeholt?* (*Do you have written permission from your supervisor?*)

SYSTEM: [SKIP TO] *Bitte holen Sie sich zunächst mittels dieses Formulars eine schriftliche Genehmigug ihres Vorgesetzten ein. Dies ist zwingend erforderlich.* (*en: Please get written permission from your supervisor as soon as possible. This is mandatory.*)

SYSTEM: [ASK] Bitte holen Sie sich zunächst mittels dieses Formulars eine schriftliche Genehmigug ihres Vorgesetzten ein. Dies ist zwingend erforderlich. (*en: Please get written permission from your supervisor as soon as possible. This is mandatory.*)

D Training details

D.1 RL Model Parameters

The following parameters were used by all user RL models (chosen through manual tuning):

Parameter	Value
Layer type	Linear
Activation (after each layer except in Dialog Mode Classifier Head)	SELU
Shared Layer Neurons (one value / layer)	$8096, 4096, 4096$
Value Function Layer Neurons (one value / layer)	$2048, 1024$
Advantage Function Layer Neurons (one value / layer)	$4096, 2048, 1024$
Dialog Mode Classifier Neurons (one value / layer)	$256, 1$
Dropout (after each layer)	25%

Table 3: RL Model Parameters

D.2 RL Experience Buffer

New goals in the replay phase of the Hindsight Experience Replay are generated only for free mode dialogs. Here, we follow the original dialog backwards until we find a suitable alternative goal node (having at least one associated user question). We choose one of the user questions of the new goal node randomly and replace the original dialog's initial user utterance with it.

Parameter	Value
Buffer size	100000
Priority Replay α	0.6
Priority Replay β	0.4

Table 4: RL Experience Buffer

D.3 Simulation Parameters

The following parameters were used by our user simulator:

Parameter	Value
Reward Normalization	$[-1.0, 1.0]$
Maximum Dialog Steps	50
User Patience	3
Probability guided vs. free dialog	0.5
Training utterance noise	10%

Table 5: Simulation Parameters

D.4 Dialog Mode Classifier

We fine-tune GBERT (Chan et al., 2020) using the Huggingface framework (Wolf et al., 2020). As data, we use utterances from our train data split and evaluate on the test data split. We have two different input pair types: 1) node text and user answer 2) node text and user FAQ question. Training is done for 5 epochs, otherwise using the standard Huggingface (Wolf et al., 2020) trainer class parameters.

D.5 RL Training Parameters

The following parameters were used to train all RL agents (chosen through manual tuning):

Parameter	**Value**
Optimizer	Adam
Learning Rate	$1e^{-4}$
λ	1.0
Maximum Training Dialog Turns	$1.5M$
Max. Gradient Norm	1.0
Batch Size	128
γ	0.99
Exploration fraction of Training Turns	0.99
Exploration Scheme	ϵ-greedy
ϵ start	0.6
ϵ end	0.0
Training frequency (w.r.t. dialog turns)	3
Training start (w.r.t. dialog turns)	1280
DDQN Target Network update frequency (w.r.t. training steps)	15
Q-Value clipping	10.0
Munchausen τ	0.03
Munchausen α	0.9
Munchausen Clipping	-1
Evaluation frequency (w.r.t. dialog turns)	10000
Evaluation dialogs	500

Table 6: RL Training Parameters

A Human Subject Study of Named Entity Recognition (NER) in Conversational Music Recommendation Queries

Elena V. Epure and **Romain Hennequin**
Deezer Research, Paris, France
`research@deezer.com`

Abstract

We conducted a human subject study of named entity recognition on a noisy corpus of conversational music recommendation queries, with many irregular and novel named entities. We evaluated the human NER linguistic behaviour in these challenging conditions and compared it with the most common NER systems nowadays, fine-tuned transformers. Our goal was to learn about the task to guide the design of better evaluation methods and NER algorithms. The results showed that NER in our context was quite hard for both human and algorithms under a strict evaluation schema; humans had higher precision, while the model higher recall because of entity exposure especially during pre-training; and entity types had different error patterns (e.g. frequent typing errors for artists). The released corpus goes beyond predefined frames of interaction and can support future work in conversational music recommendation.

1 Introduction

Music recommendation systems (RSs), fundamental to streaming services nowadays, learn from user listening history or music content which artists or tracks to suggest next (Schedl et al., 2018). Most of these algorithms provide personalized music content to the users when logging in the streaming apps or websites, or when triggered with pre-defined utterances via voice assistants (Ammari et al., 2019; Bontempelli et al., 2022). More recent *conversational* RSs aim to help users to express their recommendation needs by supporting interactions via *queries* in natural language (Jannach et al., 2021). However, despite existing in the scientific literature, such conversational RSs are not widely deployed because of multiple issues, one being NER.

The processing of recommendation queries entails the *extraction of named entity mentions* (Moon et al., 2019; Rongali et al., 2020). This sub-task faces multiple challenges, even when queries are framed as pre-defined utterances. The transcriptions of the voice queries results in lower-case noisy text, often with misspellings (Muralidharan et al., 2021). The lack of capitalisation in entities and misspelled words are often present in text-based queries too (Cheng et al., 2021). Music entities, or those coming from the creative content domains, are highly irregular: they do not follow inherent patterns as it is the case with people's names, and there is little to no separation between the vocabularies of entity and context words, especially for creative works (Derczynski et al., 2017) (e.g. common words like "I" or "love" in track titles). Also, new music entities appear all the time. Major music streaming services ingest one new track almost every second (Ingham, 2021).

Previous works have already shown that NER systems struggle with the aforementioned challenges (Augenstein et al., 2017; Lin et al., 2020b; Epure and Hennequin, 2022). Thus, multiple approaches have been proposed to address them, either focused 1) on collecting more and relevant data for training / fine-tuning standard NER sequential models (Lison et al., 2020); or 2) on model's design choices that favour generalisation (Guerini et al., 2018; Lin et al., 2020a). Most solutions focused on the latter objective have been motivated by the *human NER linguistic behaviour*, e.g. make the model rely more on context cues than on named entity mentions or learn from a few examples only, as humans do. However, apart from some scarce, partially related works (Derczynski et al., 2016; Ding et al., 2021), there is no systematic investigation of how humans actually perform NER on noisy text with many new and irregular named entities. Moreover, in the case of music recommendation, we are not aware of any existing dataset of queries in natural language, annotated with named entities.

Thus, our goal is to investigate the human NER linguistic behavior when confronted with these challenging conditions. For that, we create *Musi-*

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1281–1296
May 2-6, 2023 ©2023 Association for Computational Linguistics

cRecoNER, a new corpus of noisy natural language queries for music recommendation in English that simulates human-music assistant interactions. We then conduct a *human subject research study* to establish a human baseline and learn from it. Finally, we perform a detailed comparison of humans and the most popular NER systems nowadays, fine-tuned transformers, that covers multiple evaluation schemes (*strict* named entity segmentation and typing, *exact* segmentation only, or partial segmentation with strict named entity typing) and scenarios including entities previously seen or unseen by the model or humans.

The results showed that the task was challenging for humans. Given an aggregated metric such as F1 score, human and algorithmic performances were on par. However, the detailed evaluation revealed that humans struggled more with recall while the best model with precision. The high recall obtained by the model was partially a result of entity exposure during pre-training or fine-tuning. Also, music entities had different error patterns and, in some queries, had ambiguous context that made their segmentation and typing quite hard.

To sum up, our research contribution[1] are:

1. *MusicRecoNER*, a corpus of noisy complex natural language queries for music recommendation collected from human-human conversations in English, but which simulates human-music assistant interactions, annotated with *Artist* and *WoA* (work of art) entities. This dataset is not limited to pre-defined utterances as it would be the case if collected from interactions with conversational or voice assistants. Thus, it contains entities in diverse context, being also a useful resource for future work on conversational music recommendation.

2. A *human subject study design for NER* in noisy text with many new and irregular named entities. The proposed method is transferable to other creative content domains that face similar challenges to music such as books, movies, videos, but also to any other domain with scarce data, which wants to learn more about the NER task before building a system.

3. An *extensive music NER benchmark on noisy text* which compares the performance of human versus automatic baselines under multiple evaluation schemes, scenarios and by controlling for the *novelty* of named entities.

2 Related Work

Analysing human and algorithmic performance was done for multiple NLP tasks in the past. Nangia and Bowman (2019) ran an annotation campaign on the GLUE benchmark with the goal to estimate the effort needed by existing models to catch up with the humans under limited-data regimes. Kazantseva and Szpakowicz (2012) conducted a large-scale human study on topic shift identification in order to discover patterns of disagreements and consolidate the evaluation metrics. Ghaly and Mandel (2017) analysed the human behaviour for understanding ambiguous text-based or spoken sentences to guide the development of a machine learning system. Multiple machine translation works challenged the human parity claim (Toral, 2020) and proposed a secondary evaluation method to reveal detailed differences between humans or algorithms (Graham et al., 2020).

Compared to these, we benchmark humans and models on a different task—named entity recognition, but we share similar goals—to estimate the human-algorithmic performance gap and to identify patterns that could support the design of better evaluation methods or automatic solutions. Human annotation is frequent in NER especially when targeting a new domain such as archaeology (Brandsen et al., 2020), or a new language such as Indonesian (Khairunnisa et al., 2020). However, we are not aware of any annotated corpus of conversational queries for recommendation in the music domain. Some other related works propose corpora of noisy social media text containing new entities including irregular ones (Derczynski et al., 2016, 2017), a noisy dataset of movie-related queries (Liu, 2014), a dataset of music artist biographies annotated for entity linking (Oramas et al., 2016), or a corpus of tweets associated with a classical music radio channel (Porcaro and Saggion, 2019).

Previous works have showed that transformers fine-tuned for NER are strong baselines, especially when training data is scarce (Akbik et al., 2019; Fu et al., 2020). A more recent line of research employs these pre-trained models as few-short learners (Yang and Katiyar, 2020; Tänzer et al., 2022). However, the results are still below those obtained with a fine-tuning approach. In order to improve the bare-bone fine-tuned transformers, other works

---

[1] Code and data are available at `https://github.com/deezer/music-ner-eacl2023`.

adopted distant supervision (Lison et al., 2020), and the inclusion of gazetteers (Shang et al., 2018) or contextual triggers (Lin et al., 2020a). Though these solutions are interesting and relevant to our problem and context, in the current research, we want to rely on the results of this study before making any design choices for an advanced NER system in the music domain.

When conducting human subject studies, the quality of annotations (inter-rater agreement or reliability) is often assessed with Kappa statistic or its variations (McHugh, 2012). Yet, for NER, or more generally for labelling phrases, this statistic is less applicable as the number of negative cases on which it relies is ill-defined (Hripcsak and Rothschild, 2005). To address this issue, multiple imperfect solutions have been proposed such as to compute the Kappa statistic at the token level (Deleger et al., 2012)—however, this does not reflect the task well as each token is not tagged individually; or to estimate the negative cases by enumerating all n-grams or noun phrases from a text—however, this lacks accuracy (Grouin et al., 2011). Hripcsak and Rothschild (2005) show that when the number of negative cases gets very large, the Kappa statistic approaches the F1 score. Thus, F1 is considered a better metric, which we also adopt to measure the performance of humans and compare the NER human and algorithmic baselines.

3 Human Subject NER Study

3.1 Data Collection

For data collection we have chosen the *music suggestions* subreddit[2] as a relevant data source. Reddit is a discussion website where members can submit questions, share content and interact with other members. It is organised in subreddits built around dedicated topics. Each discussion starts with an initial post that has a title and description. From this post, threads of conversations develop. We were interested only in posts triggered by a music information seeking or recommendation need. We crawled the full subreddit with 8615 initial posts. This number corresponds to the posts in the beginning of 2020. We did not consider posts' comments.

These humans-to-humans posts asking for music recommendations are particularly relevant to study as they go beyond pre-defined frames of interaction with a text or voice-based assistant. Hence, they exhibit a realistic human use of language, which although more challenging, could help with the development of the next generation of music assistants. For NER, the existence of queries in natural language translates in a more diverse context surrounding named entities, thus in a higher query generalisation for music recommendation. By manually checking this data, we noticed that many mentioned artists or music titles were not popular. Thus, we expected most named entities to be new to the annotators, an aspect we wanted to control for, as mentioned in Section 1.

1	looking for some playlists to listen to before going to sleep i usually listen to beach house madlib etc
2	ive just started listening to grateful dead and the ramones what else have i missed
3	looking for music similar to yamashita
4	songs sounds like drive by lil peep
5	new rappers

Table 1: Examples of queries in *MusicRecoNER*.

3.2 Data Cleaning and Pre-processing

As we aimed at creating a corpus of music recommendation queries simulating human-assistant interactions, we made multiple decisions to pre-process the collected posts. We performed a manual cleaning of this data by removing those posts which directly shared music with the community; were aimed at promoting music or other music-related entities; contained explicit words; or contained only links to external music resources.

Then, we focused on *titles* only as the post content was rather long, specific to asynchronous communication; as human-assistant interactions happen synchronously, the written or spoken queries are expected to be short, composed of a few short sentences at most (Song and Diederich, 2010). We removed all references to specific music-related services in order to obtain generic queries (e.g. we removed "Youtube" from the request "music similar to my Youtube playlist"). We also removed words which were explicit markers of human-human interaction in order to ensure compatibility with human-assistant interaction. For instance, we removed phrases such as "hello guys" or "could anybody".

We performed the rest of the pre-processing steps to ensure that the queries contained, to some extent, the kind of noise that could be found in *transcribed* voice queries too, such as those obtained when interacting with a voice assistant. For this, we transformed the text in lowercase and removed punctuation marks and emoticons (with some ex-

---

[2] `www.reddit.com/r/musicsuggestions/`

ceptions when the symbol was part of the named entity's pronunciation such as "&"). We kept content from parentheses when found at the end of a post title, otherwise we removed it. Although very common in automatic transcriptions, we did not introduce any artificial noise regarding the spelling of named entities. Still some noise was present as Reddit authors sometimes made misspelling errors. These steps were done automatically. We release both the original and pre-processed data. All keywords used in the described steps are in Appendix A. We show multiple query examples in Table 1.

3.3 Annotation Guidelines and Procedure

We sampled multiple subsets of 600 queries each from the cleaned and pre-processed corpus. This number was established by estimating the required time for the experiment to be maximum 2 hours per annotator, based on an initial trial on 751 queries. The annotation guidelines were also tested in the trial experiment and refined after. The subjects were informed that the goal was to identify names of artists (e.g. bands, singers, composers) and titles of works of art (e.g. albums, tracks, playlists, soundtracks) in unformatted music-related queries. We requested the annotators not to consult the Internet as we wanted them to rely on the query content only and on their own previous knowledge.

We then introduced the labels: *Artist_known*, *Artist_deduced*, *WoA_known*, *WoA_deduced*, and *Artist_or_WoA_deduced* with examples. The last one was for ambiguous cases of named entity typing, but allowed the annotators to segment. Segmentation is still very relevant when parsing natural language queries for music recommendation as the type could be eventually disambiguated with the help of a search engine, for instance. The other labels corresponded to *Artist* and *WoA* types, completed by whether the annotator knew the entity from before or deduced it from query's content, as we wanted to keep track of *entity's novelty*.

Then, we introduced challenging annotation cases with guidelines on how to proceed. We instructed the annotators to include *Artist* and *WoA* named entities from other domains too such as movies or video games, but to ignore all the other entity types such as countries or music genres; to consider the innermost entities in case of nested entities; to ignore implicit entities such as "this singer"; to always include the "'s" from the possessive case as part of the named entity; and to consider a named entity with misspelled, translated and transliterated words as correct. The final form of the guidelines is shown in Appendix B.

Ten annotators (1 for the trial, and 9 for the main study) were recruited from our organisation with the condition to be fluent in English. Each set of 600 queries (DS1, DS2, and DS3) was given to three annotators. The annotation campaign was performed using Doccano (Nakayama et al., 2018). The guidelines and the annotation tool were presented in a 30-minute workshop where annotators could ask questions. They could consult the guidelines and contact the researchers if they needed any clarification during the experiment too. After, one week was set aside for each annotator to complete the annotations individually.

3.4 Ground-truth *MusicRecoNER* Corpus

Often in related works, a ground-truth corpus is obtained by using full agreement or majority voting (Nangia and Bowman, 2019; Lin et al., 2020a) (e.g. tag named entities on which at least two out of three human annotators agreed). However, here, because we wanted to establish a human baseline and have a corpus exhaustively annotated, we labelled the ground-truth corpus ourselves from scratch.

Compared to the settings of the human subject study, we had access to the original Reddit post titles including capitalised text and punctuation. During the annotation, we used web and music streaming search engines to check if certain entities were *Artist* or *WoA*. The full Reddit post was also used to disambiguate cases when a name could be both an *Artist* or a *WoA*. The most challenging examples were discussed among us. The ground-truth preparation together with the adjudication discussions happened over several weeks, as the process to disambiguate entities was more complex.

Statistics about each dataset are presented in Part I of Table 2. *Artist* mentions are more common than *WoA* mentions. Regardless of the type, we could notice that a large majority of entity mentions are unique in each dataset. The mean number of entity mentions per query is around 2, with the maximum varying between 6 and 10. From these, the proportion of queries with no entity is on average 56%.

4 Evaluation protocol

4.1 Fine-tuned Transformer Baselines

The goals of the human subject NER study are to establish a human baseline on this challenging dataset

	Artist$_t$	**Artist$_u$**	**WoA$_t$**	**WoA$_u$**	**%query$_{w/oents.}$**	**ents./query**	**Train**	**Pre-train**	**Human**
DS1	303	289	208	202	58%	2.0 ± 1.0	15%	51%	29%
DS2	285	271	221	220	56%	1.9 ± 0.9	14%	43%	30%
DS3	299	284	229	229	57%	2.0 ± 1.1	15%	44%	24%
Trial	383	360	270	269	56%	2.0 ± 1.0	11%	47%	27%

Table 2: Part I shows the total ($[\text{Type}]_t$) and unique ($[\text{Type}]_u$) numbers of *Artist* and *WoA* mentions; % of queries with no entities (%query$_{w/oents.}$); and per query mean and std. of entity mentions (ents./query). Part II shows % of unique *test* entities in the *train* set (Train), seen during model *pre-training* (Pre-train) or known to *humans* (Human).

of noisy queries for music recommendation and to learn from the human linguistic behavior in comparison to the most common NER systems nowadays, the fine-tuned transformers. We consider three language models proven to have good results in various natural language tasks including language understanding, sequence labeling or text classification: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and MPNet (Song et al., 2020).

BERT (Devlin et al., 2019) is a multi-layer bidirectional encoder based on the original Transformer architecture (Vaswani et al., 2017). It is pre-trained on: 1) the cloze task, i.e. to predict a masked token from the left and right context; and 2) next sentence prediction, i.e. to predict the next sentence from a given one. **RoBERTa** (Liu et al., 2019) has the same architecture as BERT, but incorporates multiple training steps proven to lead to an increased performance than the original model: the training of the model using more data, with larger batches, on longer sequences and for a longer time; and keeping only the cloze task as a pre-training objective while applying a dynamic masking schema to the input training data. **MPNet** (Song et al., 2020) proposes a new pre-training objective by integrating the masked language modeling objective of BERT and the permuted language modeling objective introduced in XLNet (Yang et al., 2019). That is, it models the dependency among the masked tokens at prediction (i.e. takes into account the already predicted masked tokens to generate the current one), while providing visibility on the position information of the full sentence (i.e. the positions of the masked token and the next ones to be predicted).

We fine-tune the pre-trained versions of these models released in the *huggingface transformers* library (Wolf et al., 2020) for token classification / sequence labeling. We took the largest available version for each of them: *bert-large-uncased*, *roberta-large*, and *mpnet-base*. From all, only BERT is pre-trained on uncased text.

During experiments, we noticed that the model initialisation had a large impact on the results. This instability is well-documented in the past work, especially when the corpus for fine-tuning was small (Zhang et al., 2021). Thus, to overcome bad initialisation and have more coherent results over different runs, we re-initialized the last layer of each pre-trained model. This also led to faster convergence and more efficient fine-tuning. We also tried to increase the number of the re-initialized layers to 2, but the results were similar or sometimes worse.

4.2 Evaluation Metrics and Schemes

Precision (P), recall (R) and F1 are commonly used to evaluate automatic NER systems (Yadav and Bethard, 2018). In our evaluation, we extend these metrics to support a more detailed benchmark and understanding of the kind of errors a NER system makes. Namely, we also allow for a relaxed system's evaluation, when either segmentation or typing is correct, but not necessarily both.

A NER system can produce various types of outcomes (Chinchor and Sundheim, 1993a; Chinchor, 1991). Inspired by this and Batista (2018), all NER outcomes, which we denote O, can be:

- *Correct* outcomes (O_c): predicted and ground-truth entities match.
- *Missing* outcomes (O_m): system entirely fails to spot a ground-truth entity.
- *Spurious* outcomes (O_s): false entities are produced by the system.
- *Incorrect* outcomes (O_i): predicted and ground-truth entities do not match because of either typing or segmentation errors.

To classify the predictions of a NER system in these categories, we first need to fix *an evaluation schema*. The most common one in the literature is the *Strict* match (UzZaman et al., 2013; Chinchor, 1991) when both segmentation and typing are correct. Under the *Strict* schema, a prediction is *incorrect* when its boundaries were correct but not its

Ground-truth		**Predicted**		*Strict*	*Exact*	*Type*
Artist	the beatles	*Artist*	the beatles	o_c	o_c	o_c
Artist	the beatles	*WoA*	the beatles	o_i	o_c	o_i
Artist	the beatles	*Artist*	beatle	o_i	o_i	o_c
Artist	the beatles	*WoA*	beatle	o_s	o_s	o_s
		WoA	love	o_s	o_s	o_s
Artist	the beatles			o_m	o_m	o_m

Table 3: Example of outcomes under various evaluation schemes.

type, or when its type was correct but not its boundaries. All other cases (e.g. partial segmentation with incorrect type) are classified as *spurious*.

The *Exact* schema classifies a prediction as *correct* when its boundaries match those of the ground-truth, regardless of its type. In contrast, the *Entity* schema classifies a prediction as *correct* when its type matches that of the ground-truth, regardless of its boundaries. For these latter schemes, *incorrect* is adapted from its definition in *Strict*; *missed* and *spurious* are the same too.

We use another class of outcomes, *partial* (O_p), only when computing the human performance. As described in Section 3.3, humans could annotate a text as *Artist_or_WoA_deduced*. Thus, whenever a human prediction had this label and matched exactly the boundaries of the ground-truth entity, *partial* was incremented and contributed to the final scores with a factor of 0.5 (Chinchor and Sundheim, 1993b), as follows:

$$R = (|O_c| + 0.5 * |O_p|)/(|O| - |O_s|) \quad (1)$$

$$P = (|O_c| + 0.5 * |O_p|)/(|O| - |O_m|) \quad (2)$$

We exemplify the different outcomes under the mentioned schemes in Table 3.

One practical detail regarding the calculation of the evaluation metrics is that we had to apply some segmentation corrections before, to cover the situations when human annotations started or finished in the middle of a word. This could appear because Doccano did not force automatically an alignment to a desired tokenization (entire words). Thus, we corrected the start or end index of the concerned span by moving them to the left or right, based on a simple heuristic with regard to the closest found separating character (space or newline) to the concerned word. We did not intervene when an entity was composed of multiple words and only a part of them were annotated, but we captured this type of errors with the used evaluation schemes. No correction was needed in the case of model annotations as, during fine-tuning, we propagated the label of the first word token to the rest; hence, the labels were always consistent for all word tokens.

4.3 Evaluation Scenarios

We explicitly consider the novelty of entities. In the case of humans, this was encoded in the annotation process as we introduced the labels suffixed with *_known*. Fine-tuned models could have seen music entities from the test set during pre-training, when they were exposed to a large amount of unlabelled data or during fine-tuning, if the train and test sets had common entities. While this latter exposure could be easily checked, the pre-training exposure is more challenging to assess as it requires access to the pre-training data or to find other ways to test exposure based on the model only (Epure and Hennequin, 2022; Tänzer et al., 2022).

The solution we adopted targeted BERT, which performed on par with the other models as revealed in Section 5. BERT is pre-trained on Wikipedia and BookCorpus (Devlin et al., 2019). Thus, music entities could be found more likely in the Wikipedia content. However, some music entities could be quite rarely mentioned in Wikipedia compared to others. To quantify BERT's exposure to an entity e we used the following method. First, we tried to link each entity to Wikipedia by querying the Wikidata knowledge base (Vrandečić and Krötzsch, 2014). We re-ranked the returned results to give priority to music entities and returned the first entity whose type was in a pre-defined type list (see Appendix C). Second, we computed exposure by adapting the metric proposed by Carlini et al. (2019):

$$\text{expo}(e) = \begin{cases} \log|\mathcal{S}| - \log \text{rank}(e) & e \in \text{Wiki.} \\ 0 & e \notin \text{Wiki.} \end{cases} \quad (3)$$

where $\mathcal{S}$ represents all Wikipedia named entities and the function *rank* considers entity popularity (higher the popularity, lower the rank). We retrieve $\mathcal{S}$ and entity counts from Wikipedia2Vec (Yamada et al., 2020). We manually checked the linking

Model	**Artist**	**WoA**	**Macro**
BERT	0.80 ± 0.03	0.72 ± 0.04	0.76 ± 0.03
RoBERTa	0.77 ± 0.01	0.71 ± 0.05	0.74 ± 0.03
MPNet	0.80 ± 0.03	0.72 ± 0.05	0.76 ± 0.04

Table 4: *F1* scores under the *strict* evaluation schema.

for 300 random entities. 82% were correct, either linked or not found on Wikipedia correctly. 14% were linked to music-related entities but not the right ones and the rest were errors or missed entities. Examples of entities with high exposure values are: *the beatles*, *elvis*, *pink floyd*, *metallica*, *drake*, *johnny cash*, *eminem*, *nirvana*, and *coldplay*. We could notice that all are of type *Artist*.

5 Results and Discussion

We report scores using 4-fold cross-validation on the datasets presented in Table 2. Means and standard deviations (std.) are computed over different folds, different initialisation seeds for the model, and different human annotators. In most cases, this was over 12 data points as, for each model, the results were aggregated over each dataset as a test and 3 different initialisation seeds[3] and for the human evaluation, over each dataset as a test and 3 human predictors per dataset.

When comparing BERT and the other models in Table 4, BERT and human baselines in Tables 5 and 6, and results on Seen versus Unseen entities obtained either by humans or BERT in Table 7, scores in bold are statistically larger (p-value= 0.05). We test statistical significance with the Mann-Whitney U Test (Wilcoxon Rank Sum Test, Mann and Whitney 1947), which assesses under the null hypothesis that two randomly selected observations X and Y come from the same distribution.

5.1 Fine-tuned Transformer Baselines

Table 4 shows that the fine-tuned BERT, pre-trained on uncased text, and MPNet yield the largest *F1* scores for each entity type or overall. RoBERTa is statistically comparable and only marginally lower than the other models. Although MPNet and RoBERTa share the same pre-training corpus and the Transformer architecture, the addition of the permuting language objective to the cloze task gives a slight advantage to MPNet. We use BERT for the rest of the experiments.

	Artist		
	Strict	Exact	Entity
BERT	0.80 ± 0.02	0.84 ± 0.02	0.83 ± 0.02
human	0.77 ± 0.06	0.84 ± 0.05	0.81 ± 0.05
	WoA		
	Strict	Exact	Entity
BERT	0.71 ± 0.04	0.75 ± 0.04	0.78 ± 0.04
human	0.74 ± 0.07	0.79 ± 0.07	0.80 ± 0.05

Table 5: *F1* scores under different evaluation schemes.

		P	R
Artist	BERT	0.79 ± 0.02	**0.82 ± 0.03**
	human	**0.82 ± 0.04**	0.73 ± 0.07
WoA	BERT	0.67 ± 0.04	0.74 ± 0.05
	human	**0.78 ± 0.07**	0.70 ± 0.08

Table 6: *P*recision (P) and and *R*ecall (R) under the *strict* evaluation schema.

5.2 Humans vs. Fine-tuned BERT

Table 5 shows that the performance of BERT is comparable to that of the human baseline in terms of *F1* score. However, Table 6 shows that humans and BERT perform differently in terms of *precision* and *recall*. Humans have a higher *precision*, for both *Artist* and *WoA*, whilst BERT has a marginal or significantly larger *recall* than humans, especially for *Artist*. We confirmed that this phenomenon was not due to a particular precision / recall compromise by testing various precision / recall value and optimizing on F1. Also, BERT has a lower *precision* than the *recall*, but we see the opposite for humans. Considering Equations 1 and 2, the model appears to hypothesize spurious entities more often, while humans tend to miss entities more often.

Table 5 also shows that the *F1* scores under *Exact* and *Entity* schemes are larger than under *Strict* as some of the errors produced are because of segmentation or typing. However, we can notice a different behaviour for the two entity types for both BERT and humans. In the case of *WoA*, the *Entity F1* scores are slightly larger than those obtained under the *Exact* schema, showing that boundary errors happen more frequently. On the contrary, for *Artist* entities, the segmentation is more often correct, but the typing is wrong.

5.3 Error Analysis

Figure 1, showing a detailed error analysis, confirms that indeed BERT has more often *spurious* outcomes than humans, for both entity types. Also, humans miss to annotate ground-truth entities more often than BERT. We can equally observe that BERT is highly superior in identifying correct

---

[3] All the models were trained and tested on the ground-truth datasets, and did not consider annotator-specific sets.

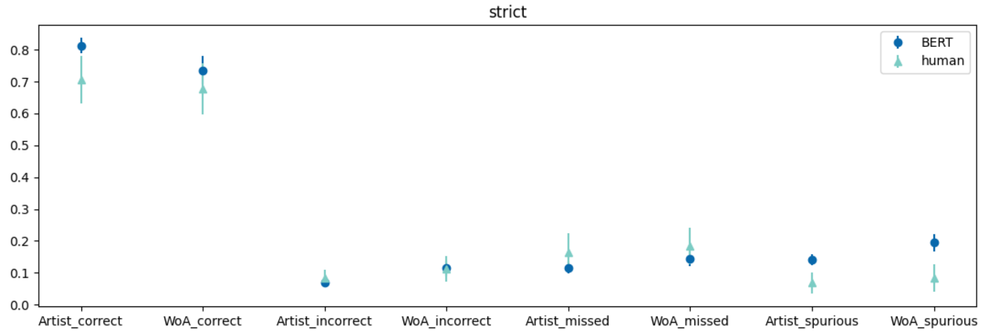


Figure 1: Normalized *correct*, *incorrect*, *missed* and *spurious* outcomes per entity type under *strict*.

named entities. Previous works on NER (Lin et al., 2020a; Epure and Hennequin, 2022) have discussed that a system should learn to exploit the context (i.e. the non-entity words) rather than entity memorisation to generalise. However, the high number of correctly recognised entities as well as the frequent spurious entities suggest that this may not be the case here; and BERT's behaviour may be linked to entity exposure. As shown at the end of Section 4.3, the entities with the highest exposure score were of type *Artist*. We could see in Figure 1 that there are a lot more correct *Artist* entities, and the number of *missed* and *spurious* outcomes for *Artist* is lower, which seems to be aligned with our hypothesis related to entity exposure.

5.4 Impact of Entity Exposure

In Table 2, Part II, we show the percentage of entities known by at least one annotator among the three in each dataset. This varies between 24% and 30%. In practice, each annotator has known at most this number of entities, which confirms that most entities from the collected corpus were new to our subjects. The entity exposure is much larger for BERT. While the train and test sets share only maximum 15% of the entities, BERT has seen up to a half of corpus' entities during pre-training.

To check the model's performance on seen versus unseen entities, we show *Recall* scores for these groups in Table 7. *Seen* entities are those present in the train set or with $expo(e) > 1$. *Unseen* entities have $expo(e) = 0$ and are not known to humans. The rest of the entities are discarded from the evaluation. BERT's recall on *Seen* is much larger than on *Unseen*, which confirms our hypothesis that memorisation plays a role. However, the model seems to rely significantly on context too given that the results on Unseen are still quite high.

We also report the results of humans in Table 7 and see a similar pattern. Although the split is made considering *the model's exposure*, humans are also very likely to know entities from *Seen*. The lower humans' scores on *Unseen* show that the recognition of these entities is quite challenging, possibly because of insufficient context. For example, "songs bands similar to sales getting it on off and on and porches mood" contains an enumeration that is difficult to segment and type (*Artist*: "sales", "porches"; *WoA*: "getting it on", "off and on", "mood"). Also, entity typing is ambiguous in "anything similar to some people say" (*WoA*). For these imperfectly recognised entities, including external resources such as gazetteers or search engines might be an option to explore.

6 Conclusion

In this work, we investigated the human linguistic behavior when performing NER in the music domain. We created *MusicRecoNER*, a new corpus of complex noisy queries for recommendations annotated with *Artist* and *WoA* entities. We then designed and conducted a human subject research study to establish a human baseline and learn from its comparison with the most popular systems nowadays, fine-tuned transformers. We performed a thorough evaluation covering multiple metrics, schemes and scenarios, including a careful analysis of the impact of entity exposure on results.

The results obtained by the algorithmic baselines were comparable to the human ones. Yet, the detailed evaluation showed that humans yielded a better precision while the model had a better recall, linked also to entity exposure during pre-training and fine-tuning. Thus, when evaluating fine-tuned pre-trained models, checking their performance on new entities shows their real generalisation ability.

	Artist		**WoA**	
	Seen	Unseen	Seen	Unseen
BERT	**0.86 ± 0.03**	0.74 ± 0.05	**0.81 ± 0.05**	0.69 ± 0.06
human	**0.77 ± 0.07**	0.63 ± 0.08	**0.74 ± 0.08**	0.66 ± 0.09

Table 7: *R*ecall scores under the *strict* evaluation schema on Seen and Unseen.

Regarding the NER evaluation protocol, human performances were much better under a more relaxed schema focused on segmentation or typing only. Such a schema could prove a more realistic setup to aim to when training models too. Also, we noticed that the relevant schema depended on the entity type as *Artist* was better segmented, while *WoA* better typed.

Contrary to previous claims, we show that, in our domain, NER in challenging conditions such as noisy text, and irregular or novel entities is rather hard for humans even when provided with complex instructions and multiple examples. Thus, although we could learn from the human linguistic behaviour, we should not, by default, assume their results to be a target for any NLP problem. For some tasks, it is common when establishing a human baseline to consider it as an upper bound for the model. This is not necessarily a desirable outcome in our case as it would imply mislabelling $1/3$ *WoA* entities. More generally, as we also showed by studying the impact of entity exposure, algorithms can store a lot more knowledge than humans and one may want to leverage this as much as possible.

As for proposing a better system to perform music NER, one next step would be to continue the model's pre-training on more related data, in our case music, to get even more exposure, or to integrate gazetteers. Still, given the rate of new entities in our domain, forcing the model to rely more on context, when context is not confusing, is another desirable future direction. In case of context ambiguity, asking questions to clarify the request and supporting user interaction in natural language could be ultimately the answer towards a more suitable, but still very challenging solution. We plan to explore these ideas as future work.

7 Limitations

We further discuss the limitations of our work. The corpus of noisy complex queries in natural language we use in the human subject study and we release is built based on a single source, Reddit. The demographics of the users using Reddit are relatively narrow, with a majority being male, young, and educated[4]. Moreover, users seeking music recommendations on this type of forums may be rather "music enthusiasts" and may not represent regular music listeners. The implications are that the language employed in these queries could be specific to this category of population. Also, the mentioned entities could reflect the music taste of this type of profiles only. This latter implication turned to be an advantage for us as we ended up with many novel entities, unknown by the annotators who participated in the study. As for the first implication, we manually checked the queries, and found them quite diverse, not necessarily using a specific vocabulary but more general language expressions. An alternative to creating such a corpus could have been a Wizard of Oz experimental setup (Green and Wei-Haas, 1985). However, this would require significantly more costs and would highly depend on the type of profiles interested in participating in such a music discovery experiment.

Second, we pre-processed the corpus in order to simulate written or transcribed speech-based human-computer interactions. However, the steps we took may be largely insufficient to simulate the kind of noise found in transcriptions. As we also discussed in Section 3.2, we did not inject any artificial noise for named entities, while spelling errors when automatically transcribing them are a common problem. Another limitation regarding named entities is the computation of the model's exposure by leveraging Wikipedia. Our linking was quite rudimentary and imperfect, as we reported in Section 4.3. Moreover, for retrieving entity ranks, we used Wikipedia2Vec (Yamada et al., 2020), which is built on a slightly older Wikipedia version than the one BERT was trained on. Therefore, the results obtained by the model on the *Unseen* dataset may be slightly larger, as the model could have been exposed to some of these entities. However, our goal was to prove a phenomenon—the impact of named entity exposure on the results, even if this impact may be marginally underestimated.

Finally, the annotators recruited from our organ-

---

[4] `https://foundationinc.co/lab/reddit-statistics/`

isation have similar age and demographics. Also they likely have a richer musical background compared to regular human subjects. This signifies that, in reality, the number of novel entities could be higher, which could also impact the overall results obtained with the human baseline. Nevertheless, this hypothesis could be tested only by running subsequent studies including more subjects.

8 Ethical Considerations

We have provided most of the details about data collection, data cleaning and pre-processing, and the annotation procedure and guidelines in Section 3 and Appendices. We discuss further various ethics-related aspects not covered yet in the paper.

The dataset was gathered from the music suggestion subreddit via the Reddit API. According to the privacy policy of Reddit[5], third parties can freely access public content via the API. We have not gathered any other information besides the public posts—their titles and descriptions.

As previously mentioned, the annotators were recruited from our organisation. They performed the annotation tasks during their regular paid hours. Moreover, the participation was fully on voluntarily basis, following a public call for participation by the authors within the organisation.

References

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

Tawfiq Ammari, Jofish Kaye, Janice Y. Tsai, and Frank Bentley. 2019. Music, search, and iot: How people (really) use voice assistants. *ACM Trans. Comput.-Hum. Interact.*, 26(3).

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition. *Comput. Speech Lang.*, 44(C):61–83.

David S. Batista. 2018. Named-entity evaluation metrics based on entity-level.

Théo Bontempelli, Benjamin Chapus, François Rigaud, Mathieu Morlon, Marin Lorant, and Guillaume Salha-Galvan. 2022. Flow moods: Recommending music by moods on deezer. In *Proceedings of the 16th ACM Conference on Recommender Systems*, RecSys '22, page 452–455, New York, NY, USA. Association for Computing Machinery.

Alex Brandsen, Suzan Verberne, Milco Wansleeben, and Karsten Lambers. 2020. Creating a dataset for named entity recognition in the archaeology domain. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4573–4577, Marseille, France. European Language Resources Association.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 267–284, USA. USENIX Association.

Xiang Cheng, Mitchell Bowden, Bhushan Ramesh Bhange, Priyanka Goyal, Thomas Packer, and Faizan Javed. 2021. An end-to-end solution for named entity recognition in ecommerce search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17):15098–15106.

Nancy Chinchor. 1991. MUC-3 evaluation metrics. In *Third Message Uunderstanding Conference (MUC-3): Proceedings of a Conference Held in San Diego, California, May 21-23, 1991.*

Nancy Chinchor and Beth Sundheim. 1993a. Muc-5 evaluation metrics. In *Proceedings of the 5th Conference on Message Understanding*, MUC5 '93, page 69–78, USA. Association for Computational Linguistics.

Nancy Chinchor and Beth Sundheim. 1993b. MUC-5 evaluation metrics. In *Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993.*

Louise Deleger, Qi Li, Todd Lingren, Megan Kaiser, Katalin Molnar, Laura Stoutenborough, Michal Kouril, Keith Marsolo, Imre Solti, et al. 2012. Building gold standard corpora for medical natural language processing tasks. In *AMIA Annual Symposium Proceedings*, volume 2012, page 144. American Medical Informatics Association.

Leon Derczynski, Kalina Bontcheva, and Ian Roberts. 2016. Broad Twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179, Osaka, Japan. The COLING 2016 Organizing Committee.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

---

[5] `https://www.reddit.com/policies/privacy-policy`

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

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

Elena V. Epure and Romain Hennequin. 2022. Probing pre-trained auto-regressive language models for named entity typing and recognition. In *the 13th Edition of Language Resources and Evaluation Conference*, LREC2022.

Jinlan Fu, Pengfei Liu, and Qi Zhang. 2020. Rethinking generalization of neural models: A named entity recognition case study. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7732–7739.

Hussein Ghaly and Michael Mandel. 2017. Analyzing human and machine performance in resolving ambiguous spoken sentences. In *Proceedings of the Workshop on Speech-Centric Natural Language Processing*, pages 18–26, Copenhagen, Denmark. Association for Computational Linguistics.

Yvette Graham, Christian Federmann, Maria Eskevich, and Barry Haddow. 2020. Assessing human-parity in machine translation on the segment level. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4199–4207, Online. Association for Computational Linguistics.

Paul Green and Lisa Wei-Haas. 1985. The rapid development of user interfaces: Experience with the wizard of oz method. *Proceedings of the Human Factors Society Annual Meeting*, 29(5):470–474.

Cyril Grouin, Sophie Rosset, Pierre Zweigenbaum, Karën Fort, Olivier Galibert, and Ludovic Quintard. 2011. Proposal for an extension of traditional named entities: From guidelines to evaluation, an overview. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 92–100, Portland, Oregon, USA. Association for Computational Linguistics.

Marco Guerini, Simone Magnolini, Vevake Balaraman, and Bernardo Magnini. 2018. Toward zero-shot entity recognition in task-oriented conversational agents. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 317–326, Melbourne, Australia. Association for Computational Linguistics.

George Hripcsak and Adam S Rothschild. 2005. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American medical informatics association*, 12(3):296–298.

Tim Ingham. 2021. Over 60,000 tracks are now uploaded to spotify every day. that's nearly one per second. Accessed June 7, 2022.

Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Comput. Surv.*, 54(5).

Anna Kazantseva and Stan Szpakowicz. 2012. Topical segmentation: a study of human performance and a new measure of quality. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 211–220, Montréal, Canada. Association for Computational Linguistics.

Siti Oryza Khairunnisa, Aizhan Imankulova, and Mamoru Komachi. 2020. Towards a standardized dataset on Indonesian named entity recognition. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 64–71, Suzhou, China. Association for Computational Linguistics.

Bill Yuchen Lin, Dong-Ho Lee, Ming Shen, Ryan Moreno, Xiao Huang, Prashant Shiralkar, and Xiang Ren. 2020a. TriggerNER: Learning with entity triggers as explanations for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8503–8511, Online. Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Jialong Tang, Xianpei Han, Le Sun, Zhicheng Wei, and Nicholas Jing Yuan. 2020b. A rigorous study on named entity recognition: Can fine-tuning pretrained model lead to the promised land? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7291–7300, Online. Association for Computational Linguistics.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1518–1533, Online. Association for Computational Linguistics.

JJ (Jingjing) Liu. 2014. A conversational movie search system based on conditional random fields. In *Interspeech 2012*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

H. B. Mann and D. R. Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, 18(1):50–60.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy. Association for Computational Linguistics.

Deepak Muralidharan, Joel Ruben Antony Moniz, Sida Gao, Xiao Yang, Justine Kao, Stephen Pulman, Atish Kothari, Ray Shen, Yinying Pan, Vivek Kaul, Mubarak Seyed Ibrahim, Gang Xiang, Nan Dun, Yidan Zhou, Andy O, Yuan Zhang, Pooja Chitkara, Xuan Wang, Alkesh Patel, Kushal Tayal, Roger Zheng, Peter Grasch, Jason D Williams, and Lin Li. 2021. Noise robust named entity understanding for voice assistants. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 196–204, Online. Association for Computational Linguistics.

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

Nikita Nangia and Samuel R. Bowman. 2019. Human vs. muppet: A conservative estimate of human performance on the GLUE benchmark. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4566–4575, Florence, Italy. Association for Computational Linguistics.

Sergio Oramas, Luis Espinosa Anke, Mohamed Sordo, Horacio Saggion, and Xavier Serra. 2016. ELMD: An automatically generated entity linking gold standard dataset in the music domain. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3312–3317, Portorož, Slovenia. European Language Resources Association (ELRA).

Lorenzo Porcaro and Horacio Saggion. 2019. Recognizing musical entities in user-generated content. In *International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing 2019.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing. In *Proceedings of The Web Conference 2020*, WWW '20, page 2962–2968, New York, NY, USA. Association for Computing Machinery.

Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. 2018. Current challenges and visions in music recommender systems research. *International Journal of Multimedia Information Retrieval*, 7(2):95–116.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2054–2064, Brussels, Belgium. Association for Computational Linguistics.

Insu Song and Joachim Diederich. 2010. Intention extraction from text messages. In *Neural Information Processing. Theory and Algorithms*, pages 330–337, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2022. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578, Dublin, Ireland. Association for Computational Linguistics.

Antonio Toral. 2020. Reassessing claims of human parity and super-human performance in machine translation at wmt 2019. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 185–194. European Association for Machine Translation. Annual Conference of the European Association for Machine Translation ; Conference date: 03-11-2020 Through 05-11-2020.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. SemEval-2013 task 1: TempEval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 23–30, Online. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample {bert} fine-tuning. In *International Conference on Learning Representations*.

Spotify
Check out
http
Compiled
Event
Weekly
Instagram
[
Hour
Official
Guys
Reddit
Post
Made
Hi
Listen
If
Tell me
Text
Playlist
Skunk
Buy
Zoom
Promotion
Someone
iTunes
{
Release
Video
S**t
Apple
Soundcloud
Our
Hello
Cover
You
Check
Radio
YouTube
Enjoy
Download
Quarantine
Anybody
Playlist
/r/
Stream
Anyone
F**k
Link
Radio
Thanks
Mix
My
Inside
Example

Table 8: Keywords used for manually filtering out Reddit posts in the data pre-processing step.

A Data Filtering Keywords

The list of keywords used in the data cleaning and pre-processing steps are presented in Table 8. These keywords were used to filter out posts, which were manually verified after. The outcome of the verification was either to exclude these posts from the data, or to keep the posts as they were or after having removed specific words (as described in Section 3.2). We have considered both lower and upper case variations of each keyword.

B Annotation Guidelines

B.1 Introduction

The goal of this annotation experiment is to identify names of artists (e.g. bands, singers, composers) and names of works of art (e.g. albums, tracks, playlists, soundtracks, movies, video games) in music-related requests. The requests could be single- or multi-line. Also, they are unformatted, meaning that they contain no capitalized letters or punctuation marks. Also contractions such as "**Artist's** first album", "**don't**" are written as if pronounced, specifically "artists first album" and "dont".

Through this experiment, we study how well humans can identify named entities (artists and works of art) in unformatted text by relying on the *request content only*, and on one's own knowledge. For this reason, it is important that during annotation you do not consult the Internet to verify if some parts of text are named entities, but rely on your intuition after reading the text.

B.2 Named Entity Categories

There are two categories referring to the entity type *Artist*; two categories referring to the entity type Work of Art (*WoA*); and one category for dealing with ambiguous cases as follows:

1. Artist_known. This category should be used for sequences of words denoting an artist that is previously known to the annotator.

In the next request I recognize "queen" and "the clash" to be *Artist* entities because I knew them from the past: *i like* ***queen*** *and* ***the clash*** *what else should i listen to*.
Note that when "the" is part of the name (e.g. "the clash"), it should be annotated likewise.

2. Artist_deduced This category should be used for sequences of words denoting an artist that is not known to the annotator, but deduced from the text.

In the next request I recognize "stephan forté" to be an *Artist*: *looking for something like the first album of* ***stephan forté***.
I have never heard of this artist before, but I deduced it from the request's content.

3. WoA_known This category should be used for sequences of words denoting a work of art that is previously known to the annotator.

In the next request I recognize "karma police" to be a *WoA* because I knew it before: *im a very picky music listener but i love* ***karma police*** *any other suggestions*.

4. WoA_deduced This category should be used for sequences of words denoting a work of art that is not known to the annotator, but deduced from text.

In the next request I recognize "special affair" to be a *WoA*: *songs like* ***special affair***.
I have never heard of this work of art before, but I deduced it from the request's content.

5. Artist_or_WoA_deduced This category is used for sequences of words recognised to denote an artist or a work of art, but choosing between the two entity types is challenging.

In the next request I recognize "superunloader" to be either an *Artist* or a *WoA*: *music like* ***superunloader***
I have never heard of this before and it is difficult for me to distinguish between the two options.

B.3 (Challenging) Examples

Please read the following examples carefully and re-consult them during the experiment whenever needed.

Relevant named entities not related to music. A text could contain other types of works of art such as movies or video games. Annotate these names using the category *WoA*. Similarly, annotate with the *Artist* category movie directors, filmmakers, music composers and so on. All the other types of named entities not related to *Artist* and *WoA* must be ignored (e.g. names of countries, music genres etc.).

In the example below, "gemini" is annotated as *WoA* and "ang lee" as *Artist*: *i recently watched this film **gemini** made by **ang lee** and liked the soundtrack any similar recommendations of this*.

Multiple named entities clearly delimited. A text could contain multiple entities which are clearly delimited by other words such as "by", "from", "and" etc. Please annotate all of them individually.

In the example below, "heartbeat" is a *WoA* and "annie" is an *Artist*: *songs with similar vibe and structure as **heartbeat** by **annie***.

In the example below, "hallelujah" is a *WoA* and "jeff buckley" is *Artist*: *other beautiful songs by **jeff buckley** apart from **hallelujah***.

Multiple named entities with no delimitation. A text could contain multiple entities which are not clearly delimited. Try to annotate each segment of text individually with its corresponding named entity category.

In the example below, if the annotator recognizes the entities, then 3 separate *Artist* entities should be selected, namely "imagine dragons", "bastille", and "daya": *singers bands like **imagine dragons bastille** and **daya***.

However, if not all entities are known from the past, then the unknown span of text could be annotated either as *Artist_or_WoA_deduced*, *Artist_deduced* or *WoA_deduced* depending on the content and the annotator's intuition. For instance, if the annotator recognizes only "imagine dragons" but not the rest, then "bastille and daya" could be considered either 1 entity ("bastille and daya") or 2 entities ("bastille", "daya") and further annotated with any of the 3 categories mentioned above.

Named entities collated with 's from the possessive case. In this case, include the "s" as part of the named entity.

In the example below, "toni braxton" is the real name of the artist, but "toni braxtons" (coming from "toni braxton's") is actually annotated as an *Artist* entity: *newer 2005+ ballads in the style of **toni braxtons** un break my heart and **stevie wonders** all in love is fair*. Similarly for "stevie wonders" (coming from "stevie wonder's").

Nested named entities. A text could contain nested entities. This means that there is a larger text segment that could be considered as an entity and a smaller text segment inside the larger one that could be also considered as an entity.

In this case, always favor the *innermost* text segment with an exception which is described below. Multiple examples are given further.

In the example below, "treasure planet" is annotated as *WoA* and not "treasure planet soundtrack" (which is also a *WoA*, but the innermost one is chosen): *looking for calm violin music very similar to the first 34 seconds of 12 years later from the **treasure planet** soundtrack*.

In the example below, "ezra collective" and "ty" are annotated as 2 separate *Artist* entities and not as one: "ezra collective feat ty": *recommend me some good jazz hip hop songs with rap like chapter 7 by **ezra collective** feat **ty***. There is also a third entity, "chapter 7", annotated as *WoA*):

In the example below, although "i took a pill in ibiza seeb remix" could be considered as a *WoA*, the innermost entities are annotated instead, namely "i took a pill in ibiza" as *WoA* and "seeb" as *Artist*: *songs similar to **i took a pill in ibiza seeb** remix*.

Exception: if the name of a well-known band that you recognize is composed of 2 or more individual artist names, then annotate the band name using the category *Artist_known*. In the example below, I recognized that "emerson lake and palmer" is the name of a band despite the fact that it refers to three individual artists ("emerson", "lake", "palmer"): *other artists similar to **emerson lake and palmer***.

Explicit versus implicit named entities. There are cases when an *Artist* or a *WoA* are mentioned in text, but these entities are not explicitly named. For instance, neither "the last album", nor "this singer" are explicit named entities in the request below; hence they must not be annotated: *show me* *something similar to the last album of this singer.*

(Incorrect) Variations of the original named entities. The text may contain variations of the original names of the entities (including misspelled, missing, translated or transliterated words). Normally, in order to recognize an incorrectly written named entity, the named entity must be already known to the annotator. In these cases, even if the named entity does not match exactly the real name, the annotator is required to annotate the corresponding span of text using the named entity categories ending with the "_known" suffix.

In the example below, the annotator recognizes "hey ponchuto" to be mistakenly written: *fast dancey blues or songs like **hey ponchuto** from **the mask***. The original named entity which the annotator knows from the past is "hey pachuco". Thus, "hey ponchuto" is annotated as *WoA_known*. Note that "the mask" is a WoA too (a movie).

C Pre-defined Entity Types for Wikidata Linking

In order to ensure that the entity linking gives priority to music-related entities, we re-rank the returned results. Specifically, we return the first entity whose type matches any of the criteria below:

- *Artist*: type matches exactly one of the following types—*musical group*, *rock group*, *supergroup*, *musical ensemble*, *girl group*, or it contains one of the following strings—*band*, *duo*, *musician*, *singer*.

  WoA: type matches exactly one of the following types—*album*, *musical work/composition*, *song*, *single*, *extended play*, or it contains one of the following strings—*album*, *song*.

If the previous matching fails, the fallback is the first entity of type *human* for an *Artist* entity, or of type *video* or *film* for a *WoA* entity. If none of these type criteria is met, then an empty string, corresponding to failed linking is returned.

D Computational Information

For training and evaluation, we had a 32-core Intel Xeon Gold 6134 CPU @ 3.20GHz CPU with 128GB RAM, equipped with 4 GTX 1080 GPUs, each with 11GB RAM. Fine-tuning a single model on three datasets from the four we annotated during 3 epochs and testing it on the hold-out dataset on a single GPU took about 2 minutes.

Entity Disambiguation with Entity Definitions

Luigi Procopio[1,2*] **Simone Conia**[1] **Edoardo Barba**[1] **Roberto Navigli**[1]
[1] Sapienza NLP Group, Sapienza University of Rome
[2] Sunglasses.ai
`luigi.procopio@sunglasses.ai`
`conia@di.uniroma1.it`
`{barba,navigli}@diag.uniroma1.it`

Abstract

Local models have recently attained astounding performances in Entity Disambiguation (ED), with generative and extractive formulations being the most promising research directions. However, previous works have so far limited their studies to using, as the textual representation of each candidate, only its Wikipedia title. Although certainly effective, this strategy presents a few critical issues, especially when titles are not sufficiently informative or distinguishable from one another. In this paper, we address this limitation and investigate the extent to which more expressive textual representations can mitigate it. We evaluate our approach thoroughly against standard benchmarks in ED and find extractive formulations to be particularly well-suited to such representations. We report a new state of the art on 2 out of the 6 benchmarks we consider and strongly improve the generalization capability over unseen patterns. We release our code, data and model checkpoints at `https://github.com/SapienzaNLP/extend`.

1 Introduction

Being able to pair a mention in a given text with its correct entity out of a set of candidates is a crucial problem in Natural Language Processing (NLP), referred to as Entity Disambiguation (Bunescu and Paşca, 2006, ED). Indeed, since ED enables the identification of the actors involved in human language, it is often considered a necessary building block for a wide range of downstream applications, including Information Extraction (Ji and Grishman, 2011; Guo et al., 2013; Fatahi Bayat et al., 2022), Question Answering (Yin et al., 2016) and Semantic Parsing (Bevilacqua et al., 2021; Procopio et al., 2021). ED generally occurs as the last step in an Entity Linking pipeline (Broscheit, 2019), preceded by Mention Detection and Candidate Generation, and its approaches have traditionally been divided into two groups, depending on whether co-occurring mentions are disambiguated independently (*local methods*; Shahbazi et al. (2019); Wu et al. (2020); Tedeschi et al. (2021)), or not (*global methods*; Hoffart et al. (2011); Moro et al. (2014); Yamada et al. (2016); Yang et al. (2018)).

Despite the limiting operational hypothesis of independence between co-occurring mentions, local methods have nowadays achieved performances that are either on par or above those attained by their global counterparts, mainly thanks to the advent of large pre-trained language models. In particular, among these methods, generative (De Cao et al., 2021) and extractive (Barba et al., 2022) formulations are arguably the most promising directions, having resulted in large performance improvements across multiple benchmarks. Regardless of their modeling differences, the key idea behind these methods is to part away from the previous classification-based approaches and, instead, adopt formulations that better leverage the original pre-training of the underlying language models. On the one hand, generative formulations tackle ED as a text generation problem and train neural architectures to generate auto-regressively, given a mention and its context, a textual representation of the correct entity. On the other hand, extractive approaches frame ED as extractive question answering: they first concatenate a textual representation of each entity candidate to the original input and then train a model to extract the span corresponding to the correct entity.

Although they have admittedly attained great improvements, both in- and out-of-domain, to the best of our knowledge, previous works on both these formulations have limited their studies to a single type of textual representation for entities, that is, their title in Wikipedia. This strategy, however, presents a number of issues (Barba et al., 2022) and, in particular, often results in representations that

---

[*] Work carried out while at the Sapienza University of Rome.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1297–1303
May 2-6, 2023 ©2023 Association for Computational Linguistics

are either insufficiently informative, or are even virtually indistinguishable from one another. In contrast to this trend, we address this limitation and explore the effect of more expressive textual representations on state-of-the-art local methods. To this end, we propose complementing Wikipedia titles with their description in Wikidata, such that, for instance, the candidates for *Ronaldo* in *Ronaldo scored two goals for Portugal* would be *Cristiano Ronaldo: Portuguese association football player* and *Ronaldo: Brazilian association football player*, rather than the less informative *Cristiano Ronaldo* and *Ronaldo*, respectively. We test our novel representations on generative and extractive formulations, and evaluate against standard benchmarks in ED, both in and out of domain, reporting statistically significant improvements for the latter group.

2 Method

We now formally introduce ED and the textual representation strategy we put forward. Then, we describe the two formulations with which we implement and test our proposal.

ED with Entity Definitions. Given a mention m occurring in a context c_m, Entity Disambiguation is formally defined as the task of identifying, out of a set of candidates $e_1, \ldots, e_n$, the correct entity e^* that m refers to. Each candidate e typically corresponds to a page in Wikipedia, and, in generative and extractive formulations, is additionally associated with a textual representation $\hat{e}$ describing its meaning. Whereas previous works have considered the title that e has in Wikipedia as $\hat{e}$ (e.g. *Cristiano Ronaldo*), here we focus on more expressive alternatives and leverage Wikidata to achieve this objective. Specifically, we first retrieve the Wikidata description of e (e.g. *Portuguese association football player*). Then, we define as the new representation of e the colon-separated concatenation of its Wikipedia title and its Wikidata description (e.g., *Cristiano Ronaldo: Portuguese association football player*).

Generative Modeling. In our first formulation, we follow De Cao et al. (2021) and frame ED as a text generation problem. Starting from a mention m and its context c_m, we first wrap the span corresponding to m in c_m between two special symbols, namely <*s*> and </*s*>; we denote this modified sequence by $\tilde{c}_m$. Then, we train a sequence-to-sequence model to generate the textual representation $\hat{e}^*$ of the correct entity e^* by learning the following probability:

$$p(\hat{e}^*|\tilde{c}_m) = \prod_{j=1}^{|\hat{e}^*|} p(\hat{e}^*_{(j)}|\hat{e}^*_{(0:j-1)}, \tilde{c}_m)$$

where $\hat{e}^*_{(j)}$ denotes the j-th token of $\hat{e}^*$ and $\hat{e}^*_{(0)}$ is a special start symbol. The purpose of <*s*> and </*s*> is to signal to the model that m is the mention we are interested in disambiguating. As in the reference work, we use BART (Lewis et al., 2020) as our sequence-to-sequence architecture for our experiments and, most importantly, adopt constrained decoding on the candidate set at inference time. Indeed, applying standard decoding methods such as beam search might result in outputs that do not match any of the original candidates; thus, to obtain only valid sequences, at each generation step we constrain the set of tokens that can be generated according to a prefix tree (Cormen et al., 2009) built over the candidate set.

Extractive Modeling. Additionally, we also consider the formulation recently presented by Barba et al. (2022) that frames ED as extractive question answering. Here, $\tilde{c}_m$, defined in the same way as it was for Generative Modeling, above, represents the query, whereas the context is built by concatenating the textual representations $\hat{e}_1, \ldots, \hat{e}_n$ of the candidates $e_1, \ldots, e_n$. A model is then trained to extract the text span that corresponds to e^*. Following the same efficiency reasoning of the authors, we use as our underlying model the Longformer (Beltagy et al., 2020), whose linear attention better scales to this type of long-input formulations. Compared to the above generative method, the benefits of this approach lie in i) dropping the need for a potentially slow auto-regressive decoding process, and ii) enabling full joint contextualization both between context and candidates, and across candidates themselves.

3 Experiments and Results

In order to assess the impact of our proposal in ED, we evaluate how the performances of generative and extractive formulations change when moving from Wikipedia titles to our alternative. To this end, in this Section, we first describe our experimental setting, discussing the datasets, evaluation strategy and comparison systems we adopt. Then, we describe the architecture we use for the two formulations. Finally, we present our findings.

	Dataset	Instances	Candidates	Failures
AIDA	Train	18,448	905,916 $^{/\,79{,}561}$	5038 $^{/\,682}$
	Validation	4791	236,193 $^{/\,43{,}339}$	1360 $^{/\,296}$
	Test	4485	231,595 $^{/\,46{,}660}$	1395 $^{/\,323}$
OOD	MSNBC	656	17,895 $^{/\,8336}$	149 $^{/\,72}$
	AQUAINT	727	23,917 $^{/\,16{,}948}$	142 $^{/\,121}$
	ACE2004	257	12,292 $^{/\,8045}$	66 $^{/\,50}$
	CWEB	11,154	462,423 $^{/\,119{,}781}$	3642 $^{/\,1265}$
	WIKI	6821	222,870 $^{/\,105{,}440}$	1216 $^{/\,719}$

Table 1: Number of instances, candidates and failures to map a Wikipedia title to its Wikidata definition in the AIDA-CoNLL (top) and out-of-domain (OOD, bottom) datasets. For candidates and failures, we report both their total (base) and unique (exponent) number.

3.1 Experimental Setup

Data. We follow the same experimental setting described by De Cao et al. (2021) and use the standard AIDA-CoNLL splits (Hoffart et al., 2011) for training, model selection and in-domain evaluation (AIDA); similarly, we leverage their cleaned version of MSNBC, AQUAINT, ACE2004, WNED-CWEB (CWEB) and WNED-WIKI (WIKI) (Guo and Barbosa, 2018; Evgeniy et al., 2013) for out-of-domain evaluation and use their same candidate sets, which were originally presented by Le and Titov (2018).[1] We match each entity candidate with its item in Wikidata[2] to retrieve the corresponding description. Due to inconsistencies in the datasets and different dump versions, this mapping is not always possible, and, in these cases, we fall back to employing their Wikipedia title alone. We report in Table 1 the number of instances, candidates and mapping failures in each dataset under consideration.

Evaluation. Following previous literature in ED, we compute scores over the test sets in terms of *inKB Micro* F_1. Furthermore, for each system we consider, we calculate the macro average of its performances both over all the test sets (Avg) and over the five out-of-domain datasets only (Avg$_{\text{OOD}}$).

Comparison Systems. We consider the original models presented by De Cao et al. (2021, GENRE) and Barba et al. (2022, ExtEnD), trained on AIDA-CoNLL with Wikipedia titles, as our main natural comparison systems; in particular, for ExtEnD, we evaluate against both its Longformer base (ExtEnD$_{\text{base}}$) and large (ExtEnD$_{\text{large}}$) alternatives. Furthermore, to better contextualize the performances we attain within the current landscape of ED, we also include three state-of-the-art systems, namely, the global model of Yang et al. (2018), and the variants of De Cao et al. (2021) and Barba et al. (2022), both pre-trained on BLINK (Wu et al., 2020) before fine-tuning on AIDA-CoNLL. However, we note that, differently from our work, these three systems use additional training data (9M samples) from Wikipedia, whereas, due to computational constraints, we limit our experiments to the sole usage of AIDA-CoNLL ($<$ 20K samples).

3.2 Architectures

For both our formulations, we closely follow the corresponding reference architectures. For the generative method, we use BART (406M parameters) as our underlying sequence-to-sequence model and fine-tune it on AIDA-CoNLL using a 10,000 token batch size, Adam (Kingma and Ba, 2015) as our optimizer and 10^{-5} learning rate, with 500 warm-up steps and linear decay. For the extractive method, we test and evaluate our approach on both the *base* (139M parameters) and *large* (435M parameters) versions of ExtEnD presented in the reference work, using Rectified Adam as our optimizer, with 10^{-5} learning rate, and training with a batch size of 8000 tokens. All the trainings are done for a single run on GeForce RTX 3090 graphic card with 24 gigabytes of VRAM. Henceforth, we refer to these systems as GENRE^{def}, ExtEnD$^{def}_{base}$ and ExtEnD$^{def}_{large}$, respectively.

3.3 Results

In Table 2 we show the *inKB Micro* F_1 score that our models and their comparison systems achieve on the datasets under consideration. As a first note, we point out that, for easier comparability between our experiments, we reproduce the original AIDA-CoNLL models of both De Cao et al. (2021) and Barba et al. (2022). While we attain comparable performances for the latter, and hence omit it, we find that our own implementation of GENRE, which we denote in Table 2 by GENRE†, obtains better results than its reference, especially out of domain, with an average improvement of more than 2 points.

Moving to GENRE^{def}, its behavior is definitely below its counterpart with Wikipedia titles (i.e.,

---

[1] These candidate sets were generated through count statistics from Wikipedia, YAGO and a large Web corpus.

[2] We took the latest dump (June 13th, 2022) at the moment of writing from the official Wikidata website: https://dumps.wikimedia.org/wikidatawiki/entities/

	Model	In-domain		Out-of-domain					Avgs	
		AIDA$_{dev}$	AIDA$_{test}$	MSNBC	AQUAINT	ACE2004	CWEB	WIKI	Avg	Avg$_{OOD}$
AIDA+	Yang et al. (2018)	-	**95.9**	92.6	89.9	88.5	**81.8**	79.2	88.0	86.4
	GENRE	-	93.3	94.3	89.9	90.1	77.3	87.4	88.8	87.8
	ExtEnD$_{large}$	-	92.6	**94.7**	**91.6**	**91.8**	77.7	**88.8**	**89.5**	**88.9**
AIDA	GENRE	-	88.6	88.1	77.1	82.3	71.9	71.7	79.5	78.2
	ExtEnD$_{base}$	-	87.9	92.6	84.5	**89.8**	74.8	74.9	84.1	83.3
	ExtEnD$_{large}$	-	90.0	**94.5**	**87.9**	88.9	**76.6**	76.7	**85.8**	**84.9**
	GENRE†	94.8	90.7	91.3	76.9	87.3	73.9	73.7	82.3	80.6
	GENRE^{def}	93.2	84.4	83.1	59.6	81.3	64.0	63.4	72.6	70.3
	ExtEnD$^{def}_{base}$	93.9	89.1	93.5	84.9	87.7	74.9	74.5	84.1	83.1
	ExtEnD$^{def}_{large}$	**94.9**	**92.4**	93.2	87.0	87.7	76.4	**78.3**	**85.8**	84.5

Table 2: *inKB Micro* F_1 scores over the AIDA-CoNLL validation and test splits, and the out-of-domain datasets when training on AIDA-CoNLL (bottom), or on additional resources as well (top). The best score in each section is marked in **bold** and, in the bottom part, if its difference from its best alternative is statistically significant ($p < 0.01$ according to the McNemar's test (Dietterich, 1998)), we also underline it.

	Model	MFC	LFC	UE	UEM	UM
AIDA	ExtEnD$_{large}$	**98.3**	**81.6**	80.9	80.9	89.0
	ExtEnD$^{def}_{large}$	**98.3**	81.0	**86.9**	**86.5**	**92.9**
OOD	ExtEnD$_{large}$	**97.2**	82.2	73.8	74.4	77.2
	ExtEnD$^{def}_{large}$	96.5	81.5	**74.5**	**75.0**	**77.7**

Table 3: Fine-grained results analysis over the AIDA-CoNLL (top) and out-of-domain (bottom) datasets. Left to right, columns are Most Frequent Class (MFC), Less Frequent Class (LFC), Unseen Entity (UE), Unseen Entity-Mention pair (UEM) and Unseen Mention (UM). **Bold** and underline have the same meaning as in Table 2.

GENRE and GENRE†), with a drop of roughly 10 points on average. To better understand this issue, we analyzed its predictions over the validation set, but did not identify any significant error pattern. In particular, we investigated whether GENRE^{def} presented length biases or was excessively skewed towards the most frequent entities and, consequently, less apt to scale over less frequent entities or unseen mentions. Interestingly, we did not find either of these to be the case, with the two systems having similar error distributions. We believe instead that the drop might be happening as the formulation behind GENRE^{def} requires modeling a much more complex output space and more data could be needed to scale properly.

Considering, instead, extractive formulations, we find the role of definitions to be definitely more impactful. ExtEnD$^{def}_{base}$ surpasses ExtEnD$_{base}$ on 3 out of 5 out-of-domain benchmarks and on the standard test set, here by more than 1 point. However, arguably our most interesting finding is the behavior of ExtEnD$^{def}_{large}$. This system attains large statistically significant improvements on AIDA$_{test}$ (+2.4) and WIKI (+1.5) and comparable performances on CWEB.

Yet, when considering Avg and Avg$_{\text{OOD}}$, ExtEnD$^{def}_{large}$ appears to behave worse than its title-only alternative, with identical Avg and inferior Avg$_{\text{OOD}}$ performances. We argue that this is an unfortunate limitation of these two metrics, inherent to their nature of macro averages, and that statistical testing depicts a more complete landscape. On the one hand, MSNBC, AQUAINT and ACE2004 are all very small datasets (Table 1) where the apparently large performance drop between ExtEnD$^{def}_{large}$ and ExtEnD$_{large}$ is not statistically significant but rather caused by a few different classifications; to put things into perspective, on ACE2004, despite the 1.2 difference in F_1 score, the predictions of the two systems differ for a total of only 8 samples, with 5 and 2 being the number of these that only ExtEnD$_{large}$ and ExtEnD$^{def}_{large}$ get right, respectively. On the other hand, on the three remaining datasets – which are far larger – ExtEnD$^{def}_{large}$ either reports a statistically significant improvement (AIDA$_{test}$ and WIKI) or performs on par (CWEB), highlighting the benefits of our more expressive textual representations.

Finally, to further examine the impact of our proposal, we investigate the effectiveness of ExtEnD$^{def}_{large}$ over different classes of mention and label frequency, both in domain, i.e., over the test set in AIDA-CoNLL, and out of domain, i.e., over the concatenation of the five datasets, and compare it with ExtEnD$_{large}$ (Table 3). Specifically, we consider instances:

- tagged with their most frequent entity in the training set (MFC);
- tagged with a less frequent entity (LFC);
- tagged with an unseen entity (UE);
- whose (mention, entity) pair does not appear in the training set (UEM);
- whose mention does not appear in the training set (UM).

Overall, apart from the MFC and LFC classes, where the difference is not statistically significant, ExtEnD$^{def}_{large}$ fares better in all other settings, which all require scaling over unseen patterns. Most notably, it yields $+6.0$ (AIDA) and $+0.7$ (OOD) improvements, both statistically significant, on unseen entities. This further underlines the better generalization capability granted by the use of more expressive textual representations.

4 Qualitative Analysis

We report in Table 4 a selection of examples from the WIKI dataset, showing candidates with both their title-only textual representations and those produced by our proposal. What we can see is that using the sole titles can result in imposing strong assumptions on what knowledge was captured by the model under consideration during its pre-training stage. For instance, in the first example in Table 4, the model needs to know, beforehand, that *Leed Rhinos* is an English rugby league football club. Moreover, relying only on titles can also result in underspecified queries. In the second example, if we were to look only at the titles provided for the two candidates, both alternatives would arguably be equally correct. A similar issue holds for the third example: although it may appear that the system could guess that the most likely candidate is the first one, as the second alternative is explicitly stated to be in Massachusetts, this strategy does not hold when considering the actual full list of candidates, which is not reported in Table 4 due to space constraints. While the model might be able to correctly predict these instances thanks to spurious correlations in the training set (e.g., the entity *Superman* being always linked to the superhero meaning, while *Superman (comic book)* to the meaning of comic book series), Table 3 clearly shows that this strategy does not scale.

Sentence: *Hugh Waddell is a Scottish [...] professional rugby league footballer [...] has played [...] at club level for [...] Leeds [...]*
Previous Candidates:
✗ Leeds
✓ Leeds Rhinos
New Candidates:
✗ Leeds: city in West Yorkshire, England
✓ Leeds Rhinos: English rugby league football club

Sentence: *World Without Superman is a Superman comic book story arc published by DC Comics.*
Previous Candidates:
✓ Superman
✗ Superman (comic book)
New Candidates:
✓ Superman: superhero appearing in DC Comics
✗ Superman: comic book series featuring Superman

Sentence: *Frank Mortimer born [...] in Wakefield was an English professional rugby league footballer [...]*
Previous Candidates:
✓ Wakefield
✗ Wakefield, Massachusetts
New Candidates:
✓ Wakefield: city in West Yorkshire, England
✗ Wakefield, Massachusetts: town in Massachusetts

Table 4: Extracts from the WIKI dataset, showing candidates with both the textual representations relying only on Wikipedia titles (**Previous candidates**), and our description-enhanced proposal (**New Candidates**). Due to space limitations, out of the 100 candidates all these three examples have, we only report the first two, which always include the correct one (denoted by ✓, as opposed to the incorrect alternative marked by ✗).

5 Conclusion

In this work, we focus on a shortcoming of generative and extractive formulations in Entity Disambiguation, namely their usage of Wikipedia titles, which are often insufficiently informative, and explore the effect of more expressive representations on these formulations. While we do not witness positive gains for generative formulations, at least in the limited data and computational regime we consider, we report strong improvements on extractive formulations. Specifically, our extractive approach sets a new state of the art on 2 out of the 6 benchmarks under consideration, and, more interestingly, shows better scalability over unseen patterns, especially unseen entities.

Limitations

We believe that our work has four major limitations. First, both the generative and extractive formulations that we consider lack parallelism, as they disambiguate each mention in the input text one at a time. While batching can definitely help, it poses additional computational requirements and, what is more, the same (but for the position of the <*s*> and </*s*> special symbols) input text would still need to be encoded multiple times. Second, our representation strategy requires the availability of descriptions in the target language in Wikidata (or some other knowledge base with a mapping from Wikipedia titles). While this data is readily available for English, this might not be the case for several other mid-to-low-resource languages. Third, both our formulations are local and, granted that pre-trained language models have certainly bridged the gap with global alternatives, their underlying independence assumption is still limiting. Finally, our proposal does incur an increased computational cost, with the textual representations getting considerably longer: while using Wikipedia titles results in sequences with an average subword length over AIDA-CoNLL of 7 and a 99th percentile of 14, adding descriptions nearly doubles the average, reaching 12.5, and makes the 99th percentile hit 29.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487 under the European Union's Horizon 2020 research and innovation programme, and the CREATIVE project (CRoss-modal understanding and gEnerATIon of Visual and tExtual content) funded by the MIUR Progetti di ricerca di Rilevante Interesse Nazionale programme (PRIN 2020).

References

Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2022. ExtEnD: Extractive entity disambiguation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2478–2488, Dublin, Ireland. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Proceedings of AAAI*.

Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China. Association for Computational Linguistics.

Razvan Bunescu and Marius Paşca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 9–16, Trento, Italy. Association for Computational Linguistics.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, 3rd Edition*. MIT Press.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Thomas G Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923.

Gabrilovich Evgeniy, Ringgaard Michael, and Subramanya Amarnag. 2013. FACC1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06- 26, format version 1, correction level 0).

Farima Fatahi Bayat, Nikita Bhutani, and H. Jagadish. 2022. CompactIE: Compact facts in open information extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 900–910, Seattle, United States. Association for Computational Linguistics.

Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To link or not to link? a study on end-to-end tweet entity linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1020–1030, Atlanta, Georgia. Association for Computational Linguistics.

Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web*, 9(4):459–479.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, Melbourne, Australia. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.

Luigi Procopio, Rocco Tripodi, and Roberto Navigli. 2021. SGL: Speaking the graph languages of semantic parsing via multilingual translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 325–337, Online. Association for Computational Linguistics.

Hamed Shahbazi, Xiaoli Z. Fern, Reza Ghaeini, Rasha Obeidat, and Prasad Tadepalli. 2019. Entity-aware elmo: Learning contextual entity representation for entity disambiguation. *CoRR*, abs/1908.05762.

Simone Tedeschi, Simone Conia, Francesco Cecconi, and Roberto Navigli. 2021. Named Entity Recognition for Entity Linking: What works and what's next. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2584–2596, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259, Berlin, Germany. Association for Computational Linguistics.

Yi Yang, Ozan Irsoy, and Kazi Shefaet Rahman. 2018. Collective entity disambiguation with structured gradient tree boosting. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 777–786, New Orleans, Louisiana. Association for Computational Linguistics.

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. 2016. Simple question answering by attentive convolutional neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1746–1756, Osaka, Japan. The COLING 2016 Organizing Committee.

Exploring Paracrawl for Document-level Neural Machine Translation

Yusser Al Ghussin[1,2], **Jingyi Zhang**[3] **and Josef van Genabith**[1,2]
[1]German Research Center for Artificial Intelligence (DFKI),
Saarland Informatics Campus, Saarbrucken, Germany
[2]Department of Language Science and Technology, Saarland University, Germany
[3]Hasso-Plattner-Institut (HPI), Potsdam, Germany
`yusser.al_ghussin/Josef.Van_Genabith@dfki.de,Jingyi.Zhang@hpi.de`

Abstract

Document-level neural machine translation (NMT) has outperformed sentence-level NMT on a number of datasets. However, document-level NMT is still not widely adopted in real-world translation systems mainly due to the lack of large-scale general-domain training data for document-level NMT. We examine the effectiveness of using Paracrawl for learning document-level translation. Paracrawl is a large-scale parallel corpus crawled from the Internet and contains data from various domains. The official Paracrawl corpus was released as parallel sentences (extracted from parallel webpages) and therefore previous works only used Paracrawl for learning sentence-level translation. In this work, we extract parallel paragraphs from Paracrawl parallel webpages using automatic sentence alignments and we use the extracted parallel paragraphs as parallel documents for training document-level translation models. We show that document-level NMT models trained with only parallel paragraphs from Paracrawl can be used to translate real documents from TED, News and Europarl, outperforming sentence-level NMT models. We also perform a targeted pronoun evaluation and show that document-level models trained with Paracrawl data can help context-aware pronoun translation. We release our data and code here[1].

1 Introduction

The Transformer translation model (Vaswani et al., 2017), which performs sentence-level translation based on attention networks, has achieved great success and significantly improved the state-of-the-art in machine translation. Compared to sentence-level translation, document-level translation (Xu et al., 2021; Bao et al., 2021; Jauregi Unanue et al., 2020; Ma et al., 2020; Maruf et al., 2019; Tu et al., 2018; Maruf and Haffari, 2018) performs translation at document-level and can potentially further improve translation quality, e.g., document-level context can help word disambiguation for translating words with multiple senses, document-level translation can help pronoun translation which requires context outside of the current sentence (Müller et al., 2018), document-level translation can improve document-level lexical cohesion in the translation (Voita et al., 2019).

Document-level neural machine translation (NMT) has received much attention in recent years (Bao et al., 2021; Donato et al., 2021; Fernandes et al., 2021; Kang et al., 2020; Saunders et al., 2020; Yu et al., 2020; Zheng et al., 2020; Yang et al., 2019; Kuang et al., 2018; Bawden et al., 2018; Zhang et al., 2018; Voita et al., 2018; Kuang and Xiong, 2018). Existing works showed that document-level translation can outperform sentence-level translation for a number of datasets, such as TED, News, Europarl (Bao et al., 2021; Donato et al., 2021; Xu et al., 2021). Although document-level NMT has shown promising results on a number of benchmarks, document-level NMT is still not widely adopted in real-world translation systems mainly due to the lack of large-scale general domain training data for document-level NMT.

We examine the effectiveness of using Paracrawl (Bañón et al., 2020) for learning document-level NMT. Paracrawl is a large-scale parallel corpus crawled from the Internet and contains data from various domains. The official Paracrawl corpus[2] was released as parallel sentences (extracted from parallel webpages) and therefore previous works only used Paracrawl for learning sentence-level translation. In this work, we extract parallel paragraphs from Paracrawl parallel webpages using automatic sentence alignments and we use the extracted parallel paragraphs as parallel documents for training document-level translation models. We show that document-level NMT models trained with only parallel paragraphs from Paracrawl can

---

[1] https://github.com/Yusser96/Exploring-Paracrawl-for-Document-level-Neural-Machine-Translation

[2] https://paracrawl.eu/

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1304–1310
May 2-6, 2023

be used to translate real documents from TED, News and Europarl, outperforming sentence-level NMT models and also improving context-aware pronoun translation (Müller et al., 2018).

2 Extracting Parallel Paragraphs from Paracrawl

Paracrawl (Bañón et al., 2020) is a large-scale parallel corpus crawled from the Internet and contains texts from various domains. The official Paracrawl corpus was released as aligned sentence pairs. For high-resource language pairs like German-English, the Paracrawl corpus provides 278M aligned sentences for the translation task. As Paracrawl was released as aligned sentence pairs, previous work only used Paracrawl for learning sentence-level translation. However, as described in the Paracrawl documentation, Paracrawl was constructed by first identifying aligned webpages (URLs) and then aligning sentences within aligned webpages. Although texts in parallel webpages tend to be noisy and are only loosely aligned, we demonstrate that extracting parallel paragraphs from Paracrawl parallel webpages using automatic sentence alignments can provide effective training data for learning document-level translation. Below we describe how we extract parallel paragraphs from Paracrawl.

Extracting We extract parallel paragraphs from Paracrawl and examine the effectiveness of using these aligned paragraphs for learning document-level NMT. We used German-English, a high-resource language pair, in our experiments. As the main purpose of the original Paracrawl project is to collect aligned sentences, Paracrawl did not officially release all the webpage-level aligned texts. Paracrawl only released a subset of all the aligned webpage texts, for the German-English language pair[3]. We extract parallel paragraphs from the released parallel webpages: we first download aligned webpages[4] (one German webpage is aligned with one English webpage) and then download the automatic sentence alignments (vecalign[5]) provided by Paracrawl; we then split a webpage into paragraphs according to the newline symbol; then extract aligned paragraphs from aligned webpages according to the automatic sentence alignments. We consider two paragraphs to be aligned if sentences in these two paragraphs are aligned to each other and not aligned to any other paragraphs. We discarded sentences that are not one-to-one aligned (e.g. one English sentence aligned to two German sentences) and discarded sentences that are not aligned to any other sentences. We discarded repeated paragraphs and discarded paragraphs that only contain a single sentence. Paragraphs with non-monotonic sentence alignments were also discarded.

	Paragraphs	Sentences	Words	
			En	De
Train	1.5M	5.5M	118M	109M
Dev	402	1504	32K	29K
Test	411	1510	33K	30K

Table 1: Statistics of parallel paragraphs extracted from Paracrawl.

Original vecalign	147M
After parallel paragraph extraction	11.7M
After cleaning	5.5M

Table 2: Numbers of remaining sentence pairs after different processing steps.

Cleaning To improve the quality of the extracted parallel paragraphs, we removed sentences which do not belong to the correct language[6], removed paragraphs that are too short (contain less than 30 words) and removed paragraphs with more than 50% overlap.

We randomly split the extracted parallel paragraphs into training, development and test sets as shown in Table 1. Note that, compared to the officially released Paracrawl corpus, the size of parallel paragraphs that we extracted from Paracrawl is still relatively small. This is because (i) the aligned webpages provided by Paracrawl we used for parallel paragraph extraction is only a subset of all Paracrawl data (ii) many parallel sentences were discarded due to our strict extraction rules as shown in Table 2. For future work, we will collect more webpage-level aligned Paracrawl data and test more flexible extraction rules.

We show the length statistics of the parallel paragraphs that we extracted from Paracrawl in Table 3. Compared to normal documents, the parallel paragraphs extracted from Paracrawl are generally much shorter. However, sentences in the same paragraph are usually closely related and can provide

---

[3] https://www.statmt.org/paracrawl-benchmarks/

[4] https://www.statmt.org/paracrawl-benchmarks/paracrawl-benchmark.en-de.aligned-docs.xz

[5] https://www.statmt.org/paracrawl-benchmarks/paracrawl-benchmark.en-de.vecalign.xz

[6] https://pypi.org/project/langid/

Sentences	Distribution
2	34.63%
3	29.31%
4	15.99%
5∼10	18.29%
>10	1.77%

Table 3: Distribution of paragraph length. For example, the first line of numbers mean 34.63% of the extracted paragraphs contain 2 sentences.

useful context information to help the translation of each other. In our experiments, we use the extracted parallel paragraphs as parallel documents to train document-level NMT models and we show that document-level NMT models trained with only parallel paragraphs from Paracrawl can be used to translate real documents from TED, News and Europarl, outperforming sentence-level NMT models.

3 Document-level Translation with Paracrawl

3.1 Modeling

Previous works have shown that document-level NMT models can outperform sentence-level models on several benchmarks. We adopted the recent document-level translation model, G-Transformer (Bao et al., 2021)[7], in our experiments to test the effectiveness of using Paracrawl data for learning document-level translation. The G-Transformer model is based on the standard sentence-level Transformer (Vaswani et al., 2017), but uses a whole document together as the input of the model and then generates translation for the whole document. G-Transformer improves the standard Transformer model for document-level translation with extra group tags and group attention. Each word (both source and target words) in the document is assigned with a group tag to indicate which sentence this word belongs to. Compared to the standard Transformer attention, G-Transformer computes group attention using group tags to encourage local attention and reducing the hypothesis space of the attention, especially from target to source, for long documents. G-Transformer outperformed sentence-level Transformer and obtained new state-of-the-art results on three document-level translation benchmarks.

3.2 Experimental Setting

We tokenize and truecase all data with MOSES (Koehn et al., 2007) scripts, and then perform subword segmentation with byte pair encoding (BPE) (Sennrich et al., 2016) using 30k merging operations. For both the sentence-level Transformer and the document-level G-Transformer, we used the base model setting (Vaswani et al., 2017) with 6-layer encoder/decoder, 512-dimension word embedding and 2048 hidden units for the feed forward networks. Following the G-Transformer experimental settings (Bao et al., 2021), we set the max length of a document to 512 BPE tokens (if a document is longer than 512 tokens, we split it into multiple instances). For model training, we first pretrained a standard sentence-level Transformer for 100k training steps and then finetuned the sentence-level Transformer to learn document-level G-Transformer for another 100k training steps. For a fair comparison between sentence-level and document-level translation, we also applied pretraining for sentence-level translation, i.e. we train a sentence-level Transformer with 100k pretraining and 100k finetuning steps in order to compare with G-Transformer.

3.3 Evaluating with BLEU, COMET and Targeted Pronoun Evaluation

We used BLEU (Papineni et al., 2002) and COMET (Rei et al., 2020) for translation quality evaluation[8] and we performed significance testing with bootstrap resampling (Koehn, 2004). In addition to BLEU and COMET which evaluate the general translation quality, we also performed a targeted evaluation for pronoun translation following Müller et al. (2018)'s work. The correct translation of a pronoun often requires context outside of the current sentence. Therefore, evaluation of pronoun translation can demonstrate the advantage of document-level NMT models. Following Müller et al. (2018)'s work, we computed the accuracy of our models choosing the correct translation for the English pronoun "it" from the three possible German words "es", "er" and "sie". We used a context of 5 sentences for the test data[9] provided by Müller et al. (2018) for pronoun evaluation.

---

[7] https://github.com/baoguangsheng/g-transformer

[8] Both BLEU and COMET scores were computed on sentence-level using
https://github.com/mjpost/sacrebleu
https://unbabel.github.io/COMET/html/index.html

[9] https://github.com/ZurichNLP/ContraPro

		Train-Paracrawl-Only				Train-Combined			
		EnDe		DeEn		EnDe		DeEn	
		sent	doc	sent	doc	sent	doc	sent	doc
BLEU	Paracrawl	26.57	**26.96**	30.94	**31.94**†	26.44	**27.40**†	31.36	**31.92**†
	Europarl	22.78	**23.04**	27.76	**28.69**†	28.60	**29.30**†	35.53	**35.90**†
	TED	23.56	**24.39**†	27.89	**28.24**	25.70	**26.13**	30.15	**31.49**†
	News	31.55	**32.07**†	34.59	**35.39**†	33.08	**33.89**†	35.99	**36.39**
COMET	Paracrawl	16.16	**17.99**	18.90	**20.99**†	17.66	**20.44**†	20.60	**22.02**
	Europarl	38.34	**39.84**†	42.39	**45.10**†	54.10	**54.80**†	55.78	**56.31**†
	TED	23.14	**23.77**	41.34	**42.62**	34.21	**35.25**	46.02	**46.92**
	News	33.84	**35.51**†	44.45	**48.26**†	42.49	**44.20**†	49.66	**51.35**†

Table 4: Translation results. Train-Paracrawl-Only: only Paracrawl as training data. Train-Combined: Paracrawl, Europarl, TED and News combined as training data. † represents a significant difference at the $p < 0.01$ level.

	Train-Paracrawl-only				Train-Combined			
	total	es	er	sie	total	es	er	sie
sent	0.43	0.91	0.15	0.24	0.43	**0.94**	0.17	0.20
doc	**0.55**	**0.93**	**0.34**	**0.37**	**0.60**	0.92	**0.44**	**0.44**

Table 5: Accuracy on contrastive pronoun test set with regard to reference pronoun.

	Train-Paracrawl-Only		Train-Combined	
	inside current sentence	outside current sentence	inside current sentence	outside current sentence
sent	**0.68**	0.37	0.71	0.37
doc	0.67	**0.52**	**0.76**	**0.56**

Table 6: Accuracy on contrastive pronoun test set with regard to antecedent location.

	T-P-O		T-C	
	EnDe	DeEn	EnDe	DeEn
Paracrawl	26.88	31.37	27.35	31.65
Europarl	23.22	28.47	29.03	35.61
TED	24.24	28.04	25.62	29.91
News	31.86	34.92	33.41	35.99

Table 7: Translation results (BLEU) of using the document-level G-Transformer for translating single sentences. T-P-O: Train-Paracrawl-Only. T-C: Train-Combined.

T-P-O		T-C	
inside	outside	inside	outside
0.67	0.36	0.78	0.35

Table 8: Accuracy on contrastive pronoun test set with regard to antecedent location when using the document-level G-Transformer for translating single sentences without context.

3.4 Training with Only Paracrawl

We trained both sentence-level Transformers and document-level G-Transformers with only Paracrawl (see Table 1) as training data. Parallel paragraphs were used as parallel documents to train document-level G-Transformers and parallel sentences contained in parallel paragraphs were used to train sentence-level Transformers.[10] For evaluation, we used test data from 4 datasets, Paracrawl, Europarl, TED and News. For Europarl, TED and News, we used the same test data following the original G-Transformer (Bao et al., 2021) work, i.e., the Europarl, TED and News test sets are parallel documents in contrast to the Paracrawl test set which is parallel paragraphs. BLEU and COMET scores are given in Table 4 as Train-Paracrawl-Only. The document-level G-Transformers achieved higher translation quality than sentence-level Transformers for all 4 test sets and only used Paracrawl as training data, which demonstrates that Paracrawl can provide useful document-level information for effective training of document-level NMT models. Table 4 also shows that the document-level information contained in Paracrawl is robust across domains as document-level G-Transformers trained with only Paracrawl data can help to translate real documents from TED, News and Europarl test sets.

Table 5 and Table 6 give results of targeted pronoun evaluation. Results show that the document-level model, trained with only Paracrawl data, significantly outperformed the sentence-level model for pronoun translation especially when the an-

---

[10] Therefore, training data for the document-level model and the sentence-level model contain the same number of sentence pairs.

tecedent location of a pronoun is outside of the current sentence (see Table 6), which again demonstrates that Paracrawl can provide useful information outside of the current sentence for effective learning of document-level NMT.

3.5 Training with Paracrawl, TED, News and Europarl Combined

We also trained translation models with training data from Paracrawl, Europarl, TED and News combined. The sentence-level Transformers were trained with sentence pairs from all the 4 training datasets. The document-level G-Transformers were trained with parallel paragraphs from Paracrawl and parallel documents from Europarl, TED and News. We then computed BLEU and COMET scores for the 4 test sets as shown in Table 4 as Train-Combined. The targeted pronoun evaluation results are also given in Table 5 and Table 6 as Train-Combined. Results show that using additional parallel documents together with parallel (Paracrawl) paragraphs as training data further improved the general translation quality (BLEU and COMET) and also helped the document-level model to obtain a higher accuracy for targeted pronoun translation evaluation.

3.6 Document-level G-Transformer for Translating Single Sentences

We also evaluated how the document-level G-Transformer (trained with parallel documents) performs for translating single sentences (i.e. each sentence is considered as a single document at test time). Table 7 shows that, when considering the general translation quality (e.g. BLEU), document-level G-Transformers can perform well for translating single sentences, even outperforming the sentence-level Transformers in Table 4 for most of the test sets. However, the targeted pronoun evaluation results in Table 8 show that the accuracy of translating pronouns with antecedent location outside of the current sentence dropped more than 10% compared to the document-level model in Table 6, which demonstrates that document-level G-Transformers indeed require context outside of the current sentence for improving pronoun translation.

4 Conclusion

As document-level translation lacks large-scale general-domain document-level training data, we examine the effectiveness of using Paracrawl data for learning document-level translation. Paracrawl was officially released as parallel sentences extracted from parallel webpages. In this work, we extract parallel paragraphs from Paracrawl aligned webpages using automatic sentence alignments and we use the extracted parallel paragraphs as parallel documents for training document-level NMT models. We show that document-level models trained with only parallel paragraphs from Paracrawl can be used to translate real documents from TED, News and Europarl, outperforming sentence-level Transformers and also improving context-aware pronoun translation.

Limitations

Compared to the officially released Paracrawl corpus (278M sentence pairs for German-English), the size of parallel paragraphs that we extracted from Paracrawl is still relatively small. This is because (i) Paracrawl only released a subset of all webpage-level aligned texts and we only extracted parallel paragraphs from these released webpage texts (ii) we used very strict rules for extracting parallel paragraphs from Paracrawl and many parallel sentences were discarded by our extraction rules. For future work, we will collect more Paracrawl webpage-level aligned data for parallel paragraph extraction and we will test more flexible extraction rules.

Acknowledgements

The authors acknowledge the financial support by the German Federal Ministry for Education and Research (BMBF) within the projects “CORA4NLP" 01IW20010 and “KI-Servicezentrum Berlin Brandenburg" 01IS22092.

References

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L. Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, Sergio Ortiz Rojas, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Elsa Sarrías, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 2020. ParaCrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Guangsheng Bao, Yue Zhang, Zhiyang Teng, Boxing Chen, and Weihua Luo. 2021. G-transformer for document-level machine translation. In *Proceedings* *of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3442–3455, Online. Association for Computational Linguistics.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Domenic Donato, Lei Yu, and Chris Dyer. 2021. Diverse pretrained context encodings improve document translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1299–1311, Online. Association for Computational Linguistics.

Patrick Fernandes, Kayo Yin, Graham Neubig, and André F. T. Martins. 2021. Measuring and increasing context usage in context-aware machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6467–6478, Online. Association for Computational Linguistics.

Inigo Jauregi Unanue, Nazanin Esmaili, Gholamreza Haffari, and Massimo Piccardi. 2020. Leveraging discourse rewards for document-level neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4467–4482, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Xiaomian Kang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2020. Dynamic context selection for document-level neural machine translation via reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2242–2254, Online. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Shaohui Kuang and Deyi Xiong. 2018. Fusing recency into neural machine translation with an inter-sentence gate model. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 607–617, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2018. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 596–606, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020. A simple and effective unified encoder for document-level machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3505–3511, Online. Association for Computational Linguistics.

Sameen Maruf and Gholamreza Haffari. 2018. Document context neural machine translation with memory networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1275–1284, Melbourne, Australia. Association for Computational Linguistics.

Sameen Maruf, André F. T. Martins, and Gholamreza Haffari. 2019. Selective attention for context-aware neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3092–3102, Minneapolis, Minnesota. Association for Computational Linguistics.

Mathias Müller, Annette Rios, Elena Voita, and Rico Sennrich. 2018. A large-scale test set for the evaluation of context-aware pronoun translation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 61–72, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Danielle Saunders, Felix Stahlberg, and Bill Byrne. 2020. Using context in neural machine translation training objectives. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7764–7770, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zhaopeng Tu, Yang Liu, Shuming Shi, and Tong Zhang. 2018. Learning to remember translation history with a continuous cache. *Transactions of the Association for Computational Linguistics*, 6:407–420.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Mingzhou Xu, Liangyou Li, Derek F. Wong, Qun Liu, and Lidia S. Chao. 2021. Document graph for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8435–8448, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhengxin Yang, Jinchao Zhang, Fandong Meng, Shuhao Gu, Yang Feng, and Jie Zhou. 2019. Enhancing context modeling with a query-guided capsule network for document-level translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1527–1537, Hong Kong, China. Association for Computational Linguistics.

Lei Yu, Laurent Sartran, Wojciech Stokowiec, Wang Ling, Lingpeng Kong, Phil Blunsom, and Chris Dyer. 2020. Better document-level machine translation with Bayes' rule. *Transactions of the Association for Computational Linguistics*, 8:346–360.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018. Improving the transformer translation model with document-level context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542, Brussels, Belgium. Association for Computational Linguistics.

Zaixiang Zheng, Xiang Yue, Shujian Huang, Jiajun Chen, and Alexandra Birch. 2020. Towards making the most of context in neural machine translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3983–3989. International Joint Conferences on Artificial Intelligence Organization. Main track.

Poor Man's Quality Estimation:
Predicting Reference-Based MT Metrics Without the Reference

Vilém Zouhar[E] **Shehzaad Dhuliawala**[E] **Wangchunshu Zhou**[E] **Nico Daheim**[D]
Tom Kocmi[M] **Yuchen Eleanor Jiang**[E] **Mrinmaya Sachan**[E]

[E]**ETH Zürich** [M]**Microsoft** [D]**TU Darmstadt**
{vzouhar,msachan}@ethz.ch

Abstract

Machine translation quality estimation (QE) predicts human judgements of a translation hypothesis without seeing the reference. State-of-the-art QE systems based on pretrained language models have been achieving remarkable correlations with human judgements yet they are computationally heavy and require human annotations, which are slow and expensive to create. To address these limitations, we define the problem of *metric estimation* (ME) where one predicts the automated metric scores also without the reference. We show that even without access to the reference, our model can estimate automated metrics (ρ=60% for BLEU, $\bar{\rho}$=51% for other metrics) at the sentence-level. Because automated metrics correlate with human judgements, we can leverage the ME task for pre-training a QE model. For the QE task, we find that pre-training on TER is better (ρ=23%) than training for scratch (ρ=20%).

github.com/zouharvi/mt-metric-estimation

1 Introduction

Quality estimation (QE) is often used in machine translation (MT) production pipelines where we need to make decisions based on the quality of an MT output but where the reference is unavailable (Specia et al., 2020, 2021). For example, QE is used in translation companies to decide whether to send a specific MT output for post-editing to a human translator or whether to use it directly (Tamchyna, 2021; Zouhar et al., 2021; Murgolo et al., 2022). In this scenario, an accurate QE system has the potential to save expensive translator effort. However, training QE models usually requires human-annotated judgements of the translation quality (Specia et al., 2013; Rubino and Sumita, 2020; Rei et al., 2020a). These human annotations are scarce and costly, especially for low-resource language directions, and may need to be replicated for new MT systems and domains (de Souza et al., 2014)

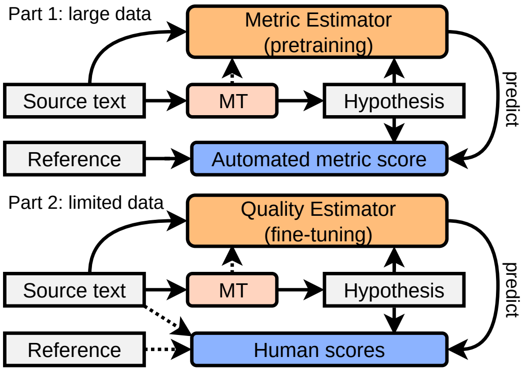


Figure 1: Pipeline for pre-training on automated metrics (top) and fine-tuning on limited quality estimation data (bottom). Dotted lines are optional dependency.

We investigate if automated MT metrics can be used to reduce the cost of learning QE models. Automated metrics can be run with no additional costs and can be used to generate large amounts of QE data. If these metrics correlate well with human judgements, this larger data can be used as a partial substitute for human data during training. Data augmentation and synthetic QE data via automated metrics has already been explored (Heo et al., 2021; Baek et al., 2020; Cui et al., 2021), though never in the pre-training & fine-tuning fashion.

Our work is guided by a simple intuition. Intuitively, human judgements can be thought of as functions depending on the target sequence and (optionally) the source and reference(s): $f_{\text{HUMAN}}([s], h, [r])$. The task of QE is to model f_{HUMAN} based only on the source s and hypothesis h. Because the function arguments of $f_{\text{HUMAN}}([s], h, [r])$ resemble those of automated metrics for MT: $f_{\text{METRIC}}([s], h, r)$, we can use the automated metrics to guide learning of the human quality judgements which are hard to obtain and

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1311–1325
May 2-6, 2023 ©2023 Association for Computational Linguistics

replicate. Generating automated metric scores is limited only by the amount of parallel data which is more abundant. Because f_{METRIC} correlates with human judgement f_{HUMAN} (Ma et al., 2019; Mathur et al., 2020), we can start by estimating f_{METRIC} and only later fine-tune to f_{HUMAN}. We refer to estimating $f_{\text{METRIC}}(s, h, r)$ as **metric estimation (ME)** as a parallel task to **quality estimation (QE)**.

We illustrate our idea of pre-training on the automatic metrics and fine-tuning on human assesments in Figure 1. Our model uses a BiLSTM with the source and hypothesis as the input (with several more features like the decoder confidence and hypothesis variance) to output a single number (metric or quality score).

Experimentally, we find the idea of mitigating data limitation for QE with ME pretraining challenging. Thus, we structure our investigation around a set of research questions. First, we try to establish that it is possible to robustly predict automated metrics and explore the associated data requirements. Then, concerned with the application and deployment of the ME model, we also check how transferrable the model is between different MT systems. We break the research down into the following questions.

RQ1: Can automated reference-based MT metrics be reliably predicted without the reference?
RQ2: What is the effect of data size on predicting automated reference-based MT metrics?
RQ3: How detrimental is the transfer between different MT systems for metric estimation models?
RQ4: Can metric estimation be used as a pre-training step for quality estimation?

We answer the research questions with experiments on the English → German language direction and replicate the main findings on 12 language pairs in total in Appendix C. We confirm that BLEU is predictable with $\rho = 60.4\%$ sentence-level Pearson's correlation and other metrics with $\bar{\rho} = 51.3\%$ (**RQ1**, Section 4.2). Authentic parallel data is needed for ME models but this can be alleviated by using more hypothesis from beam search (**RQ2**, Section 4.3). It is possible to train the ME system on one MT system and then use it on a different MT system with only a slight loss in performance (**RQ3**, Section 4.4). We find that pre-training on Translation Edit Rate (TER) (Snover et al., 2006) leads to better results than training on the QE data directly, though this approach does not outperform the state-of-the-art in QE (**RQ4**, Section 4.5).

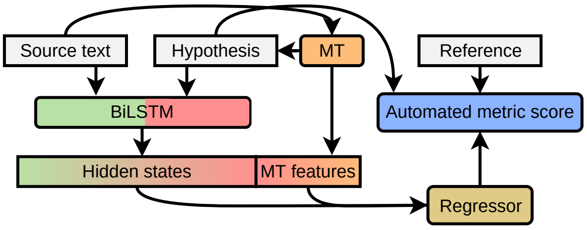


Figure 2: Metric/quality estimation model architecture.

2 Our Metric Estimation Model

Notation. Given a source sentence s and its translation, h which is an output of a MT system ($h = \text{MT}(s)$), we build a regressive ME model which outputs a numeric score that is close to the output of an automated metric that is further dependent on the reference: $f_{\text{METRIC}}(s, h, r)$. We distinguish two cases based on our level of access to the MT system: **blackbox** setting (where we assume access only to the MT system output) and **glassbox** setting (where we have access to entire MT model). In the later case, we may leverage features from the MT model to improve the ME capabilities.

Model. Our main model for ME/QE, shown in Figure 2, starts with Byte-Pair-Encoding (Gage, 1994; Sennrich et al., 2016) the source s and hypothesis h. It is followed by BiLSTM on top of concatenated source and hypothesis with a separator ($s\,\texttt{[SEP]}\,t$). The last hidden state (denoted as BiLSTM_{-1}) is extracted and fused together via concatenation with the internal MT system and other features (see the following list). This is then used in a simple feed-forward layer (FFNN) to output a single number:

$$\begin{aligned}
\text{source } s, \text{hypothesis } h &:= \text{MT}(s) \\
\Phi &= \text{BiLSTM}_{-1}(\text{BPE}(s\,\texttt{[SEP]}\,h)) \\
\Psi &= \text{Features}(\text{MT}_{\text{conf.}}(s,h), s, h) \\
\text{ME}_{\text{all}}(s,h) &= \text{FFNN}[\Phi, \Psi] \\
\text{ME}_{\text{text}}(s,h) &= \text{FFNN}[\Phi]
\end{aligned}$$

The first ME model is glassbox and the second is blackbox. In the first case, we utilize hand-crafted features and also those from the MT system (function features). Both of these models are optimized with mean-squared error against a particular metric. That is, we train separate models for each target metric (COMET, ChrF, BLEURT, BLEU, METEOR, TER) or human judgements.

$$\mathcal{L} = \frac{1}{n} \sum (\text{ME}(s_i, h_i) - f_{\text{METRIC}}(s_i, h_i, r_i))^2$$

The additional features Ψ are:

- Decoder confidence (prob and logprob).
- Source and target lengths and their relation. This is included as the distribution of errors may be different for various sentence lengths.
- Average distance and variance between hypotheses as measured by an automatic metric.

Decoder confidence. Low probability MT outputs have overall lower quality (Specia et al., 2018; Yankovskaya et al., 2018; Fomicheva et al., 2020). The decoder confidence is the hypothesis probability as defined by the model $\prod_i p(t_i|t_{<i}, s)$ which is in practice usually computed in the logspace $\sum_i \log p(t_i|t_{<i}, s)$.[1]

Hypotheses variance. Intuitively, there are many ways to generate a wrong translation but only a few correct ones (Xu et al., 2011). Similar to Fomicheva et al. (2020), we hypothesise that larger variance between the hypotheses correlates negatively with quality. We therefore use the distances between hypotheses as features for our system. Specifically, as shown in Figure 7 and formalized with the following, we use the mean distance and also the variance between distances as features. We first consider distances from the current hypothesis to be estimated (`H1`) to all other hypotheses, and then all hypothesis pairs.

$$\text{Avg or Var}(\{\text{BLEU}(h_1, h_j) | h_j \in H\})$$
$$\text{Avg or Var}(\{\text{BLEU}(h_i, h_j) | h_i, h_j \in H, i \neq j\})$$

Baselines. We use multiple baselines for comparison in the ME task. Apart from the individual features, they are all optimized to minimize the MSE loss with a specific target metric.

- Linear regression on TF-IDF features (with limited max features, see Appendix B): $\text{Lin.Reg.}\big[\text{TF-IDF}(s, h)\big]$
- Linear regression on all text & MT features: $\text{Lin.Reg.}[\Psi]$
- Fine-tuned mBERT (Devlin et al., 2019) with identical regression head architecture on top of last layer `[CLS]` hidden state[2]: $\text{FFNN}\big[\text{mBERT}(s\,\texttt{[SEP]}\,h)\big]$

We also use an off-the-shelf QE model used in WMT21 QE task `wmt21-comet-qe-mqm` (Rei et al., 2020b).[3] We do not fine-tune the model to the available data, but since we use correlation as an evaluation metric, the mean is subtracted and the output rescaled to unit variance, same as human judgements.

Automated metrics. For ME we use the following MT metrics. BLEU (Papineni et al., 2002), ChrF (Popović, 2015), TER (Snover et al., 2006) and METEOR (Banerjee and Lavie, 2005) work with lexical or character-level units, commonly in word or character n-grams. COMET (Rei et al., 2020a) uses pre-trained encoders to evaluate the hypothesis at a deeper level. BLEURT (Sellam et al., 2020) is another learned metric for text generation which uses pseudo-label. While automated metrics usually yield only sentence-level scores, QE is done for multiple levels: word, phrase and sentence. However, because of the automatic metric restrictions, we also focus on sentence-level QE in this work.

3 Experiment Setup

Pipeline & data. We start by translating 500k English→German sentences of the WMT14 dataset (Bojar et al., 2014) and computing the automated metrics of these translations.[4] We use a pre-trained WMT19 model by Ng et al. (2019).

For human scores, we use the train data of WMT21 Sentence-Level Quality Estimation Shared Task (Specia et al., 2021) which contains 14k human-direct-assessment annotated segments (5k unique). For the human score prediction, we do not have access to the features in the hypothesis space (because the hypotheses were not generated by the MT system to which we have access) and use forced decoding of the pre-trained model to get a confidence estimate. Note that since the data comes from a different MT system, there is a distribution mismatch which may negatively influence the results. We address this in Section 4.4.

We refer to the two datasets as ME and QE, respectively and show the distribution of automated metric scores and human judgements in Figure 6. We perform ME on both but QE only on the latter because of the human annotations availability.

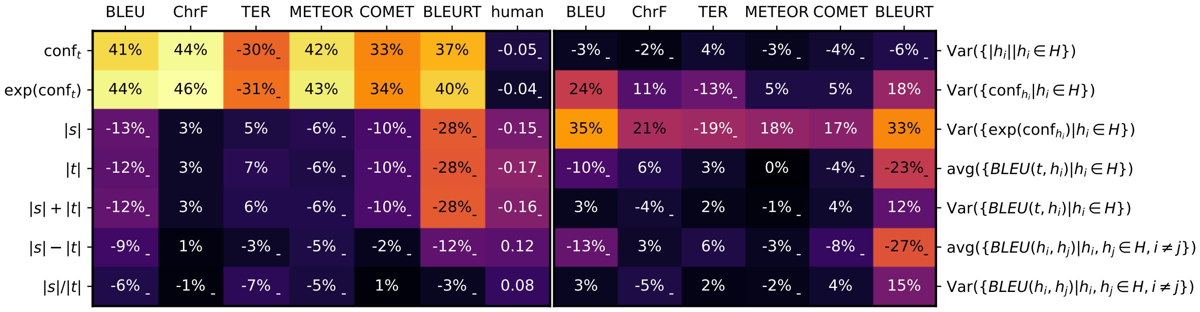
---

[1] The justification for using both conf_t and $\exp(\text{conf}_t)$ is that the non-linear transformation improves correlation.

[2] See Appendices A and B for details.

[3] This model was better than `wmt21-comet-qe-da`. Note the difference between the automated metric COMET and the QE system COMET-QE.

[4] We are not limited by the relatively small size of this dataset because we are considering only its subset and study the effect of available data size in Section 4.3.



	BLEU	ChrF	TER	METEOR	COMET	BLEURT	human
conf_t	41%	44%	-30%	42%	33%	37%	-0.05
$\exp(\text{conf}_t)$	44%	46%	-31%	43%	34%	40%	-0.04
$\lvert s\rvert$	-13%	3%	5%	-6%	-10%	-28%	-0.15
$\lvert t\rvert$	-12%	3%	7%	-6%	-10%	-28%	-0.17
$\lvert s\rvert + \lvert t\rvert$	-12%	3%	6%	-6%	-10%	-28%	-0.16
$\lvert s\rvert - \lvert t\rvert$	-9%	1%	-3%	-5%	-2%	-12%	0.12
$\lvert s\rvert / \lvert t\rvert$	-6%	-1%	-7%	-5%	1%	-3%	0.08

BLEU	ChrF	TER	METEOR	COMET	BLEURT	
-3%	-2%	4%	-3%	-4%	-6%	$\text{Var}(\{\lvert h_i\rvert \mid h_i \in H\})$
24%	11%	-13%	5%	5%	18%	$\text{Var}(\{\text{conf}_{h_i} \mid h_i \in H\})$
35%	21%	-19%	18%	17%	33%	$\text{Var}(\{\exp(\text{conf}_{h_i}) \mid h_i \in H\})$
-10%	6%	3%	0%	-4%	-23%	$\text{avg}(\{BLEU(t, h_i) \mid h_i \in H\})$
3%	-4%	2%	-1%	4%	12%	$\text{Var}(\{BLEU(t, h_i) \mid h_i \in H\})$
-13%	3%	6%	-3%	-8%	-27%	$\text{avg}(\{BLEU(h_i, h_j) \mid h_i, h_j \in H, i \neq j\})$
3%	-5%	2%	-2%	4%	15%	$\text{Var}(\{BLEU(h_i, h_j) \mid h_i, h_j \in H, i \neq j\})$

Figure 3: Feature correlations on 500k English → German sentences with reference-based metrics and human judgement. Colour is based on absolute values to show contained relevant information. Cells with negative correlations are marked with “-”.

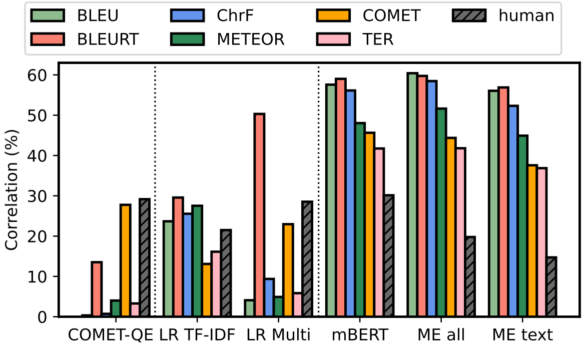


Figure 4: Correlations with metrics and human judgement of baseline and main metric estimation (ME) models. Each bar is a separate model trained to predict a particular metric or human judgement. Individual features are shown in Figure 3.

Evaluation. We evaluate ME model performance with Pearson’s coefficient with the target metric on a dev set of 10k sentences from WMT14 on the segment-level. Note that we care about the *magnitude (absolute value) of the correlation* and not whether it is positive or negative. For example, TER is expected to correlate negatively with human ranking because higher TER means more errors while higher human scores mean higher translation quality. Correlations with human judgement are evaluated on 1k WMT21 Sentence-Level QE data. Specifically, we estimate human z-scores which were computed per-annotator.[5]

4 Results

This section first studies single-feature baselines (Section 4.1) and then the possibility of robust ME model (Section 4.2) and data size requirements (Section 4.3). The model is then checked on a different MT system output to see transferability between systems and architectures (Section 4.4). Fine-tuning and evaluation on human data (QE) is done in Section 4.5. A natural follow-up experiment on using joint prediction to improve the ME model is documented in Section 4.6.

4.1 Feature analysis

We show the correlations between individual features and metrics in Figure 3. An immediate observation is that confidence-based features correlate much more with automatic metrics than the other features. Some metrics (BLEU and COMET) and especially human z-scores are highly correlated with the source and target lengths. As a negative result, very few of the hypothesis space metrics correlate highly, with an exception of $\rho(\text{BLEU}, \text{Var}(\text{BLEU}(H)))$. We still use all features later on because (1) despite low individual correlations, they may still be useful in combination or for the full model, whose input is the text, and (2) we did not encounter any overfitting issues.

BLEU	BLEURT	ChrF	METEOR	COMET	TER
11.1%	16.5%	12.3%	-12.0%	11.5%	34.6%

Table 1: Pearson correlations between human judgement (human z-scores) and automated metrics.

We show the correlations between the automated metrics and humans in Table 1. For most of them, the correlations are very low. Outliers with the highest correlation are BLEURT and COMET, which are known to be strong-performant metrics (Kocmi et al., 2021). One of the reasons is that they were specifically trained to correlate well with humans.

---

[5] Z-score of a variable has zero mean and unit variance. They are possibly unbounded but on the other hand, slightly alleviate the effect of individual annotator differences.

4.2 Metric estimation performance

We show the baselines with comparison to the main models trained only on the target data (either WMT News or WMT QE) in Figure 4. Every bar is a separate model trained to predict a specific metric and the bar magnitude shows its correlation. The systems are described formally in Section 2. Notably, *ME text* has access to only the source and hypothesis texts while *ME all* in addition fuses in extra hand-crafted features.

A simple linear regression based on features from Figure 3 is able to achieve $> 40\%$ correlations with automated metrics and $\sim$13% correlation with human judgement. These features seem important as demonstrated by the comparatively lower correlations of a TF-IDF featurizer. This is also documented by the difference between *ME all* and *ME text*. The former model consistently outperforms *ME text*, possibly because it has access to all the extra features while the latter model only works with the source and target texts. The pre-training of the *mBERT* model on language modelling helps only marginally, given that it performs only slightly better than *ME text*. The COMET-QE model almost does not correlate with automated metrics at all apart from its related reference-based metric COMET. Of all models, it also correlates the most with human judgement.

> **RQ1**: Best-performing metric estimation models (mBERT, ME all, ME text) achieve sound correlations with automated metrics (60.4% with BLEU, $\sim$ 51.3% for others).

We can interpret this correlation with respect to the individual features performances ($\max \rho = 46\%$) which shows that the model was able to learn more predictive patterns. On the selected datasets, the ME task is easier than the QE task where our moidels have a consistently lower performance.

4.3 Data requirements for ME

We are interested in how much data we need to train the ME models. This is a practical question that helps us understand the behaviour and requirements of the models.

Naturally, models utilizing just a handful of continuous non-trainable features require much less data than models whose inputs are the raw texts. This is demonstrated in Figure 5 where the linear regression gains very little even if 500× more data is used. For the main ME model, a larger amount of data is required (line H1). So far we have been considering only the highest-scored hypothesis (in terms of decoder score) among 5 generated by beam search: (s, h_1). In a low-resource scenario, it may be beneficial to create more pairs from the hypotheses provided by the beam-search decoder: $\{(s, h_i) | h_i \in H\}$. Through this hypothesis expansion, we may obtain more parallel data in data-restricted settings. Again we show the results in Figure 5 with a substantial gain of the model trained on expanded data (H2) over just the top hypothesis (H1). This effect is quickly diminishing with larger data ($\geq$ 10k) but for lower data remains a useful tool.

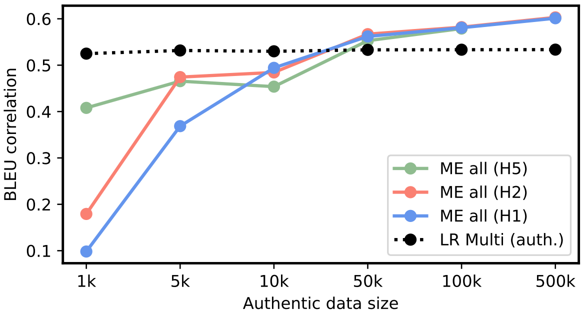


Figure 5: Model dependency on training data size. H2 and H5 expand the data by top-2 and top-5 decoder hypotheses, respectively. Note the non-linear x-axis.

> **RQ2**: The metric estimation model requires $\sim$500k parallel sentences and the associated metric scores before the performance starts to plateau. For very low-resource scenarios, it is possible to reduce the ME model data requirements by utilizing multiple hypotheses for a single source sentence.

	ME All		**ME Text**	
Model	**Transfer**	**Auth.**	**Transfer**	**Auth.**
Original	60.4%		56.1%	
W16Conv.	54.2%	57.3%	48.4%	51.0%
W16Trans.	54.2%	57.1%	47.9%	50.7%
W17Conv.	58.7%	60.8%	55.1%	55.7%
T5	19.8%	72.9%	43.8%	70.9%

Table 2: BLEU estimation correlation for texts translated by different models than WMT19 Transformer (original).

4.4 Generalization of ME across MT systems

In this section, we examine whether our ME model overfits on the specific errors the used MT system is doing or whether it generalizes and is applicable to also other MT systems. This is important in deployment so that the possible ME/QE model is not dependent on a specific MT system.

We translate the same data using the following English → German models and store the decoder features: T5-small (Raffel et al., 2020), WMT16 Convolutional (Wu et al., 2019), WMT16 Transformer (Ott et al., 2018), WMT17 Convolutional (Gehring et al., 2017). We then run the ME model trained on the outputs of the original WMT19 system to predict metric scores for the outputs of these models. We show the results in Table 2, for both text-only and feature-enriched models. The evaluation of translations by different MT systems shows a varying decrease in correlation with the automated metrics. For most systems, the drop was ∼2–3%, which means that the metric estimator generalizes well across MT systems. However, an exception is using a completely different model, the prompted T5 LM, for which the transfer mostly failed. When training metric estimators on the T5 translated data, they achieved ∼71% correlation but when models trained on outputs of a different MT system were used, the correlation dropped to 19.8% and 43.8% for *ME All* and *ME Text*, respectively. One of the reasons may be vastly different extra features, as documented by the noticeably higher correlation for the *ME Text* model.

> **RQ3**: It is possible to train a metric estimation model on sentences produced by MT_1 and use it to estimate metric scores of MT_2 with a drop in correlation of $\sim 3\%$.

4.5 From ME to QE

This section verifies empirically whether pre-training on ME helps on the QE task. For this, we use models trained on estimating automatic metrics and either: correlate their outputs with human judgements (zero-shot), or fine-tune them to predict the human judgements directly (fine-tuning). The fine-tuning was done with the same setup as in Appendix B with all model parameters updated.

The results in Table 3 show that fine-tuning definitely improved the performance over zero-shot. However, only TER was able to outperform training on z-scores from scratch. This can be attributed to it being the only metric with reasonable absolute correlation in the zero-shot. Notably, COMET was better than the rest of the automated metrics with the worst being BLEU. Despite the fine-tuning, we were not able to construct a QE system that would outperform the standard baseline of COMET-QE ($\rho = 29.2\%$).

Pre-train metric	Zero-shot	Fine-tuning
BLEU	-3.2%	-0.9%
BLEURT	5.9%	-5.2%
ChrF	-6.9%	6.5%
METEOR	-4.2%	6.0%
COMET	1.1%	10.4%
TER	-12.5%	22.8% ⋆
human	19.8%	

Table 3: Correlations with human judgement (z-scores) from models (with extra features fusion) which were pre-trained on metric estimation. Only the magnitude (absolute value) of the correlation is important.

> **RQ4**: Pre-training on TER (large data) and fine-tuning on human scores (small data) is better than training only on human scores (small data).

Further experiments with limited target-domain data (Figure 8) show that the proposed pre-training & fine-tuning regime does not perform well even with less fine-tuning data. The same figure also shows fine-tuning sensitivity to the selected data. Variance is caused by both the optimization process and data subsampling. Even though we include confidence intervals, in deployment one would start multiple runs and use the best-performing one. A striking observation is that very little human-annotated data is needed for training. Further research should more closely examine the relationship between model capacity, data requirements and QE performance.

	ME		QE	
Metric	**Single**	**Multi**	**Single**	**Multi**
BLEU	60.4%	47.0%	15.5%	23.9%
BLEURT	59.7%	5.7%	26.7%	5.7%
ChrF	58.5%	42.6%	23.7%	24.0%
METEOR	51.6%	36.5%	22.6%	24.5%
COMET	44.4%	23.1%	13.1%	21.0%
TER	37.4%	36.8%	7.6%	18.7%
human	-	-	19.8%	5.5%

Table 4: Pearson correlations between system outputs and automated metrics or human z-scores for either multiple single-metric models (Single) or a single multi-target model (Multi).

4.6 Joint prediction of multiple metrics

In this section, we investigate using all the automated metrics at the same time in a single model instead of multiple individual models.

For both the WMT News and QE datasets (individually), all the metric scores for a single segment can be predicted at the same time. Instead of training 6+7 individual models to predict each metric, we train two models (for ME and QE data) that predict all available metrics at once (similar to BLEURT pre-training phase) using different regression heads. The only difference in the architecture from Appendix B is that the last linear layer has 6 or 7 output neurons instead of one. The loss for model f is then defined as $\sum_{m\,\in\,\text{metrics}} \text{MSE}(f_m(s,t), m(s,t,r))$. Having multiple targets in a single training can provide more signal and better representation (Aho et al., 2012; Korneva and Blockeel, 2020). The results shown in Table 4 demonstrate that for the smaller dataset (QE), joint learning mostly helps in metric prediction but not in human z-score prediction. This may be because of a loss imbalance of 6 target outputs optimizing on automated metrics and only 1 target output optimizing on human z-scores.

5 Complexity & Fluency Estimation

Currently, our model was dependent on mostly the source and the hypothesis. If it had access to only the hypothesis, it could still consider its fluency and other factors in estimating the metric. Likewise, having access to only the source would correspond to sentence difficulty/complexity estimation. Similarly to Wan et al. (2022), we explored both of these modes and found very high sentence-level correlations.

$$\begin{aligned}\rho(\{(f_{\text{ME}}(h), f_{\text{BLEU}}(h,r))\}) &= 55.5\% \\ \rho(\{(f_{\text{ME}}(s), f_{\text{BLEU}}(h,r))\}) &= 55.3\% \\ h,s,r &\in \mathcal{D}\end{aligned}$$

These high correlations, which are close to the full text-only model's performance ($\rho = 56.5\%$) show that our model is not able to utilize the relationship between the source and the hypothesis and that a more elaborate models should be considered. These results are in line with general findings of Behnke et al. (2022). The models' inadequacy is also shown by the imperfect performance when given access to the hypothesis and the reference, just as the metric has:

$$\begin{aligned}\rho(\{(f_{\text{ME}}(s,h,r), f_{\text{BLEU}}(h,r))\}) &= 60.7\% \\ \rho(\{(f_{\text{ME}}(h,r), f_{\text{BLEU}}(h,r))\}) &= 61.5\% \\ h,s,r &\in \mathcal{D}\end{aligned}$$

However, the focus solely on the hypothesis or just the source itself has been confirmed for also other QE systems (Sun et al., 2020) and our model is not an outlier.

6 Related Work

This section discusses how our proposed ME task fits in the field of QE.

Confidence estimation. The task of ME has a connection to an older task of confidence estimation (Blatz et al., 2004), which predates QE (Specia et al., 2013). In confidence estimation, the goal is to predict the probability of the output being correct. Blatz et al. (2004) define correctness as a binary class which is based on two thresholded MT metrics: word error rate and NIST (Doddington, 2002). This is in contrast to the ME task which is a regression task (predicting e.g. 0.7 instead of `GOOD` and segment-level).

More recent works use the term confidence estimation more freely to mean essentially the QE task with full training data and model access (Chelba et al., 2020). Because this term is used also in other contexts, such as in calibration (Wan et al., 2020; Wang et al., 2020), we define the task metric estimation to avoid ambiguity.

Feature-based QE models. Specia et al. (2010, 2013) pioneered the work of mainstream MT QE. The QuEst model uses support vector regression on top of features such as source & target lengths, the number of translations in a phrase table or target sentence language model probability. Further research has been devoted to devising good features for MT QE models, such as grammatical ones (Felice and Specia, 2012), ones based on the decoder (Avramidis, 2012; Fomicheva et al., 2020) or based on the model embeddings spaces (Shah et al., 2016; Chen et al., 2017).

Deeper QE models. QUETCH (Kreutzer et al., 2015), NuQE (Martins et al., 2016), DeepQuest (Ive et al., 2018) and others (Kim and Lee, 2016; Li et al., 2018) regress directly from the source and hypothesis texts into a score. Notably some systems approach sentence-level QE by aggregating or otherwise utilizing previously-estimated word-level QE predictions (Kepler et al., 2019).

Pre-training QE models. Closest to our work is computing TER between the hypothesis and its post-edited version (Heo et al., 2021). QE is then trained jointly on the artificial and authentic QE data. Other models are first trained on artificial (pre-training) and then authentic data (Baek et al., 2020; Cui et al., 2021; Yankovskaya and Fishel, 2021). The pre-training task does not have to be tied to QE. Large pre-trained language models have also been used for QE (Hu et al., 2020; Moura et al., 2020; Nakamachi et al., 2020; Eo et al., 2021). In contrast, our pre-training aims not only to acquire better sentence representations in general but specifically to acquire better sentence representations for translation quality score estimation.

QE models outside of MT The idea to estimate the quality of a prediction given only itself and the input has also been applied to other NLP tasks. RUBER (Tao et al., 2018) uses an RNN-based model to compute a score for a context-response pair in a dialog that is combined with a reference-based score to obtain the final metric. BERT-RUBER extends this by using pre-trained representations and GRADE uses the unreferenced scorer in combination with a module that constructs a conversational graph using ConceptNet concepts and reasons over it (Huang et al., 2020). Recently different works evaluate responses based on specific quality attributes, for example, groundedness (Honovich et al., 2021), or combine them into one quality score (Pang et al., 2020). For open-ended tasks, reference-free metrics, similar to QE, are more desirable because they make fewer assumptions about how the hypothesis should look like. Compared to those tasks, the admissible hypothesis space in MT for a given source sentence is more constrained.

7 Discussion

The experiments have shown that the outputs of automated metrics can be predicted even without access to the reference. Pre-training only on TER and not any other automated metric outperformed training from scratch, which highlights the importance of exploring multiple metrics instead of just one or two of the most popular ones. Pre-training on TER helped because of the large absolute zero-shot correlation. We are, however, unable to provide an explanation for this zero-shot correlation in the first place.

Our ME approach can also be potentially used for improving models, even outside of the MT field, by providing additional signals through self-supervision. For example, the ME model could be used instead of the decoder probabilities for reranking when decoding with beam-search in generative models. This approach would be similar to using QE in decoding by Fernandes et al. (2022) but would not require a separate scorer model.

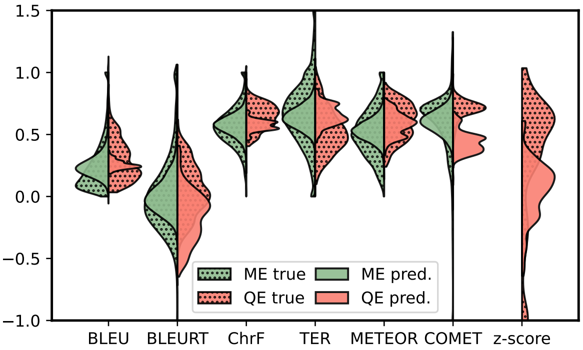


Figure 6: Distribution of metric and human judgement values in the used ME and QE datasets together with their predictions. Long tails clipped between -1 and 1.5 for higher resolution.

7.1 Error analysis

In this section, we examine three model predictions and comment on their comparison with the target scores. At the first glance, the model predictions are generally more conservative (lower variance and concentration around the average), as shown in Figure 6. This is however not an issue when evaluating with Pearson's correlation coefficient as the distributions are rescaled. While the error analysis serves as a good check for the model outputs, we are unable to clearly define specific failure modes of the model, which does not achieve 100% metric correlation.

> **Example 1** **BLEU** 0.08, **ME** 0.07
>
> **Source:** *Police try new, less-lethal tools as protests continue.*
> **Reference:** *Während die Proteste weitergehen, testet die Polizei weniger tödliche Geräte*
> **Reference (lit.):** *While the protest continue, the police is testing less-lethal devices.*
> **Hypothesis:** *Die Polizei probiert neue, weniger tödliche Werkzeuge aus, während die Proteste anhalten.*
> **Hypothesis (lit.):** *The police is testing new, less-lethal tools, as the protest persist.*

We first examine cases in which we compare our model's prediction with the true metric value.[6] Example 1 is an almost exact match in BLEU and model prediction. Although the hypothesis is reasonable, it is too literal of a translation. This is a failure for the automated metric but a success for our model because it predicted the metric accurately.

---

[6] We report true BLEU scores and not percentages (i.e. scale is 0-1, not 0-100).

> **Example 2** **BLEU** 0.67, **ME** -0.51
>
> **Source:** *It didn't work.*
> **Reference:** *Das hat nicht funktioniert.*
> **Reference (lit.):** *That did not function.*
> **Hypothesis:** *IT hat nicht funktioniert.*
> **Hypothesis (lit.):** *THAT/IT did not function.*

In some error cases, the model prediction was a better quality estimate than the metric it was trained to estimate. In Example 2, the word *it* is mistaken for a named entity which distorts the sentence meaning: *it* - pronoun, *IT* - abbreviation *Information Technology* (can also mean the tech support department).

> **Example 3** **Human** -4.72, **QE** -0.42
>
> **Source:** *There's mask-shaming and then there's full on assault.*
> **Reference:** *Masken-Shaming ist eine Sache, Körperverletzung eine andere.*
> **Reference (lit.):** *Mask-shaming is one thing, body assault is a different one.*
> **Hypothesis:** *Es gibt Masken-Beschämen und dann gibt es voll auf Angriff.*
> **Hypothesis (lit.):** *There is mask-humiliation and then there is full-on assault.*

In certain cases, the model output is very far from human judgement. In Example 3, the hypothesis contains a phrase *voll auf Angriffe* which seems like a good translation but only at the first sight and is, in fact, incorrect (word-wise translation *full - voll, on - auf, assault - Angriff*). This may be a reason for the low score by the human annotator and it was not captured at all by our model.

7.2 Negative results

We also attempted to leverage mBERT representations in the main LSTM-based model via concatenation fusion with the last hidden state, identical to the approach of Zouhar et al. (2022) for language modelling. However, the results (ρ = 60.0% for BLEU) were on par with the main LSTM-based model alone (ρ = 60.4% for BLEU).

We experimented with the expressivity of the used model architecture in Section 5 by granting the model access to the source, the reference and the hypothesis. This should provide sufficient information to be able to learn the specific metrics, however, the model was unable to fit it perfectly. Approaches in future work should therefore use models with larger capacities and explore specific mechanics that could be useful for predicting automated metrics relying on n-gram overlaps.

8 Conclusion

We proposed the task of **metric estimation** for machine translation, as a parallel to quality estimation and attempted to solve it with a baseline BiLSTM model. We show that it is possible to predict the output of a metric without even seeing the reference (ρ = 60.4% for BLEU and $\bar{\rho}$ = 51.3% for other metrics). The main advantage of this task compared to QE is that the data for training ME models to predict a particular metric can be generated from any parallel corpus on which the metric can be run. While pre-training on TER outperformed training from scratch, it did not perform better than the commonly used baseline, COMET-QE.

Future work. Despite the negative results, features in the hypothesis space should be more explored for tasks beyond ME/QE, such as calibration of self-reported confidence in generative models. Metric and quality estimation could also be a part of the MT system itself (e.g. as a separate head) which would alleviate the need for an external ME/QE model. The ME models should also be evaluated for cross-domain performance, similarly to our cross-system evaluation as motivated by needs of production settings. Non-perfect correlation with the metrics when presented with the same input shows the imporants of exploring more complex architectures or optimization approaches.

Limitations

Although our model outperformed the simple baselines in ME and QE, it provides less explainability because a specific QE output can not be linked easily to input features. The model also required much longer training[6] while the baselines just need a simple featurizer, MT intrinsic features and can run the linear regression fitting on a CPU. Nevertheless, the largest computational bottleneck in this research has been running the MT system inference[6] rather than training the individual models.

Concerns have long been raised about using segment-level metrics/evaluations because of the large variance (Lavie, 2010). However, we find that for automated metrics, our models are still able to deal with this variance.

Ethics statement

Detailed error analysis should always be performed before deploying a quality estimation system in machine translation production pipelines.

---

[6] Training one full model takes ~3 hours on 500k sentences on NVIDIA GeForce GTX 1080 Ti. Machine translation of the same data takes 120 GPU hours on the same model.

Acknowledgements

We thank Ricardo Rei (Unbabel), Jonas Belouadi (Universität Bielefeld) and Florian Schottmann (TextShuttle) for their proofreading, discussions and comments on our work.

References

Timo Aho, Bernard Ženko, Sašo Džzeroski, Tapio Elomaa, and Carla Brodley. 2012. Multi-target regression with rule ensembles. *Journal of Machine Learning Research*, 13(8).

Eleftherios Avramidis. 2012. Quality estimation for machine translation output using linguistic analysis and decoding features. In *Proceedings of the seventh workshop on statistical machine translation*, pages 84–90.

Yujin Baek, Zae Myung Kim, Jihyung Moon, Hyunjoong Kim, and Eunjeong Park. 2020. Patquest: Papago translation quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 991–998.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan.

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L. Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, Sergio Ortiz Rojas, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Elsa Sarrías, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 2020. ParaCrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online.

Hanna Behnke, Marina Fomicheva, and Lucia Specia. 2022. Bias mitigation in machine translation quality estimation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1475–1487.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *Coling 2004: Proceedings of the 20th international conference on computational linguistics*, pages 315–321.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA.

Ciprian Chelba, Hideto Kazawa, Jeff Klingner, Junpei Zhou, Mengmeng Niu, and Music Li. 2020. Data troubles in sentence level confidence estimation for machine translation.

Zhiming Chen, Yiming Tan, Chenlin Zhang, Qingyu Xiang, Lilin Zhang, Maoxi Li, and Mingwen Wang. 2017. Improving machine translation quality estimation with neural network features. In *Proceedings of the Second Conference on Machine Translation*, pages 551–555.

Qu Cui, Shujian Huang, Jiahuan Li, Xiang Geng, Zaixiang Zheng, Guoping Huang, and Jiajun Chen. 2021. Directqe: Direct pretraining for machine translation quality estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12719–12727.

José GC de Souza, Marco Turchi, and Matteo Negri. 2014. Machine translation quality estimation across domains. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 409–420.

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

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145.

Sugyeong Eo, Chanjun Park, Hyeonseok Moon, Jaehyung Seo, and Heuiseok Lim. 2021. Comparative analysis of current approaches to quality estimation for neural machine translation. *Applied Sciences*, 11(14):6584.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, 22(107):1–48.

Mariano Felice and Lucia Specia. 2012. Linguistic features for quality estimation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 96–103.

Patrick Fernandes, António Farinhas, Ricardo Rei, José GC de Souza, Perez Ogayo, Graham Neubig, and André FT Martins. 2022. Quality-aware decoding for neural machine translation. *arXiv preprint arXiv:2205.00978*.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Philip Gage. 1994. A new algorithm for data compression. *C Users Journal*, 12(2):23–38.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252.

Taisiya Glushkova, Chrysoula Zerva, Ricardo Rei, and André FT Martins. 2021. Uncertainty-aware machine translation evaluation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3920–3938.

Dam Heo, WonKee Lee, Baikjin Jung, and Jong-Hyeok Lee. 2021. Quality estimation using dual encoders with transfer learning. In *Proceedings of the Sixth Conference on Machine Translation*, pages 920–927.

Or Honovich, Leshem Choshen, Roee Aharoni, Ella Neeman, Idan Szpektor, and Omri Abend. 2021. q^2: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7856–7870, Online and Punta Cana, Dominican Republic.

Chi Hu, Hui Liu, Kai Feng, Chen Xu, Nuo Xu, Zefan Zhou, Shiqin Yan, Yingfeng Luo, Chenglong Wang, Xia Meng, et al. 2020. The niutrans system for the WMT20 quality estimation shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1018–1023.

Lishan Huang, Zheng Ye, Jinghui Qin, Liang Lin, and Xiaodan Liang. 2020. GRADE: Automatic graph-enhanced coherence metric for evaluating open-domain dialogue systems. *arXiv preprint arXiv:2010.03994*.

Julia Ive, Frédéric Blain, and Lucia Specia. 2018. DeepQuest: A framework for neural-based quality estimation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3146–3157.

Fábio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, and André FT Martins. 2019. OpenKiwi: An open source framework for quality estimation. *ACL 2019*, page 117.

Hyun Kim and Jong-Hyeok Lee. 2016. Recurrent neural network based translation quality estimation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 787–792.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 478–494.

Evgeniya Korneva and Hendrik Blockeel. 2020. Towards better evaluation of multi-target regression models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 353–362. Springer.

Julia Kreutzer, Shigehiko Schamoni, and Stefan Riezler. 2015. Quality estimation from scratch (quetch): Deep learning for word-level translation quality estimation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 316–322.

Alon Lavie. 2010. Evaluating the output of machine translation systems. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Tutorials*.

Maoxi Li, Qingyu Xiang, Zhiming Chen, and Mingwen Wang. 2018. A unified neural network for quality estimation of machine translation. *IEICE TRANSACTIONS on Information and Systems*, 101(9):2417–2421.

Qingsong Ma, Johnny Wei, Ondřej Bojar, and Yvette Graham. 2019. Results of the WMT19 metrics shared task: Segment-level and strong MT systems pose big challenges. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 62–90.

André FT Martins, Ramón Fernandez Astudillo, Chris Hokamp, and Fabio Kepler. 2016. Unbabel's participation in the wmt16 word-level translation quality estimation shared task. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 806–811.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020. Results of the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 688–725.

Joao Moura, Miguel Vera, Daan van Stigt, Fabio Kepler, and André FT Martins. 2020. Ist-unbabel participation in the wmt20 quality estimation shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1029–1036.

Elena Murgolo, Javad Pourmostafa Roshan Sharami, and Dimitar Shterionov. 2022. A quality estimation and quality evaluation tool for the translation industry. In *Proceedings of the 23rd Annual Conference of the European Association for Machine Translation*, pages 305–306.

Akifumi Nakamachi, Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. 2020. Tmuou submission for wmt20 quality estimation shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1037–1041.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium.

Bo Pang, Erik Nijkamp, Wenjuan Han, Linqi Zhou, Yixian Liu, and Kewei Tu. 2020. Towards holistic and automatic evaluation of open-domain dialogue generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3619–3629, Online.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Maja Popović. 2015. chrF: character n-gram f-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020a. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020b. Unbabel's participation in the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 909–918, Online.

Raphael Rubino and Eiichiro Sumita. 2020. Intermediate self-supervised learning for machine translation quality estimation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4355–4360.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Édouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6490–6500.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *ArXiv*, abs/1508.07909.

Kashif Shah, Fethi Bougares, Loïc Barrault, and Lucia Specia. 2016. Shef-lium-nn: Sentence level quality estimation with neural network features. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 838–842.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 shared task on quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764, Online.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Chrysoula Zerva, Zhenhao Li, Vishrav Chaudhary, and André FT Martins. 2021. Findings of the WMT 2021 shared task on quality estimation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 684–725.

Lucia Specia, Dhwaj Raj, and Marco Turchi. 2010. Machine translation evaluation versus quality estimation. *Machine translation*, 24(1):39–50.

Lucia Specia, Carolina Scarton, and Gustavo Henrique Paetzold. 2018. Quality estimation for machine translation. *Synthesis Lectures on Human Language Technologies*, 11(1):1–162.

Lucia Specia, Kashif Shah, José GC De Souza, and Trevor Cohn. 2013. QuEst: A translation quality estimation framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84.

Shuo Sun, Francisco Guzmán, and Lucia Specia. 2020. Are we estimating or guesstimating translation quality? In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 6262–6267.

Aleš Tamchyna. 2021. Deploying MT quality estimation on a large scale: Lessons learned and open questions. In *Proceedings of Machine Translation Summit XVIII: Users and Providers Track*, pages 291–305.

Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. Ruber: An unsupervised method for automatic evaluation of open-domain dialog systems. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Yu Wan, Dayiheng Liu, Baosong Yang, Haibo Zhang, Boxing Chen, Derek Wong, and Lidia Chao. 2022. UniTE: Unified translation evaluation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8117–8127.

Yu Wan, Baosong Yang, Derek F Wong, Yikai Zhou, Lidia S Chao, Haibo Zhang, and Boxing Chen. 2020. Self-paced learning for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1074–1080.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3070–3079.

Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. *arXiv preprint arXiv:1901.10430*.

Haihua Xu, Daniel Povey, Lidia Mangu, and Jie Zhu. 2011. Minimum bayes risk decoding and system combination based on a recursion for edit distance. *Computer Speech & Language*, 25(4):802–828.

Elizaveta Yankovskaya, Andre Tättar, and Mark Fishel. 2018. Quality estimation with force-decoded attention and cross-lingual embeddings. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 816–821.

Lisa Yankovskaya and Mark Fishel. 2021. Direct exploitation of attention weights for translation quality estimation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 955–960.

Vilém Zouhar, Michal Novák, Matúš Žilinec, Ondřej Bojar, Mateo Obregón, Robin L Hill, Frédéric Blain, Marina Fomicheva, Lucia Specia, and Lisa Yankovskaya. 2021. Backtranslation feedback improves user confidence in MT, not quality. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 151–161.

Vilém Zouhar, Marius Mosbach, and Dietrich Klakow. 2022. Fusing sentence embeddings into LSTM-based autoregressive language models.

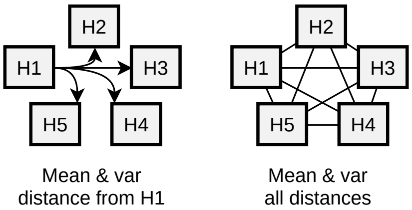


Figure 7: Four features based on properties of the generated hypothesis space: mean and variance of either H1 (top hypothesis) or all pairs. The used metric (any hyp-ref-based) considers one of the hypotheses in a pair as the reference.

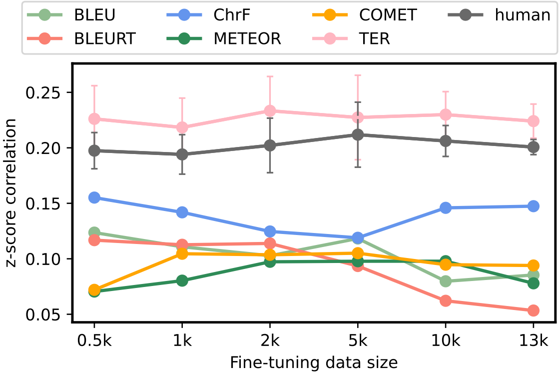


Figure 8: Finetuning on limited target-domain QE data. Note the non-linear x-axis. Each point is an average of 5 runs. Error bars show a 95% confidence interval of the mean. Error bars shown only for BLEU and human for clarity.

A Reproducibility

We compute metric scores using SacreBLEU (Post, 2018) with the following signatures:

- All: `nrefs:1 | version:2.2.0`
- BLEU: `case:mixed | eff:yes | tok:13a | smooth:exp`
- ChrF: `case:mixed | eff:yes | nc:6 | nw:0 | space:no`
- TER: `case:lc | tok:tercom | norm:no | punct:yes | asian:no`

For baseline experiments, multilingual version of BERT was used: `bert-base-multilingual-cased`. Translations using T5-small are done with the prefix "*translate English to German:*". For other translation models, we used the following models available on torch.hub under the `pytorch/fairseq` namespace (Ott et al., 2019):

- `dynamicconv.glu.wmt16.en-de`
- `conv.wmt17.en-de`
- `transformer.wmt16.en-de`
- `transformer.wmt18.en-de`

B Model details

The metric/quality estimation model specifics are shown in Table 5. Additionally, we concatenate all forward and backward hidden states from both LSTM layers. A non-standard choice was to use sigmoid as the final activation function which worked better than just the linear output.[7] However, we also rescaled and centered it in case of BLEU, ChrF and METEOR so that scores of 0 and 100 are attainable. We eventually did not use this in the main experiments so that a single model setup could be used for all metrics. All models are trained with early stopping of 10 epochs. The optimization loss is mean squared error. Our used model is fairly small in comparison to other models, such as those utilizing the Transformer architecture. This was an intentional choice with respect to the small amount of data used. The hyperparameters were chosen manually by best practices with respect to final metric correlation (5 trials).

Optimizer	Adam (Kingma and Ba, 2015)
Learning rate	10^{-6}
Batch size	10 (0-padded to longest)
Vocab size (BPE)	8192
Vocab embedding	512
LSTM	Hidden state 128 2 bidirectional layers
LSTM dropout	20% inter-layer 75% final hidden state
Fusion	Concatenate (512 + 6)
Linear	518 → 100
Activation	ReLU
Linear	100 → 1 (6/7 for multi)

Table 5: Metric/quality estimation model details.

The TF-IDF featurizer in the linear regression TF-IDF baseline uses variable maximum number of features and the best-performing one is chosen. The search is logarithmical from 2^4 to 2^{14}.

---

[7] This is dissimilar to the baseline model where linear regression worked better than logistic regression even in the case of metrics with bounded output range.

C Results for other language pairs

The paper used figures and examples from the English→German language direction. To replicate the findings, we translate 500k sentences for the following language directions, models and datasets:[8]

- German ↔ English (`WMT14`)
  `transformer.wmt19.{de-en,en-de}`
- German ↔ Polish (`opus_paracrawl`)
  `Helsinki-NLP/opus-mt-{de-pl,pl-de}`
- Chinese ↔ English (`CCMatrix`)
  `Helsinki-NLP/opus-mt-{zh-en,en-zh}`
- Czech ↔ English (`WMT14`)
  `Helsinki-NLP/opus-mt-{cs-en,en-cs}`
- Russian ↔ English (`WMT14`)
  `transformer.wmt19.{en-ru,ru-en}`
- French ↔ English (`WMT14`)
  `Helsinki-NLP/opus-mt-{fr-en,en-fr}`
- Hindi → English (`CCMatrix`)
  `Helsinki-NLP/opus-mt-{hi-en,en-hi}`

The results for metric estimation for all metrics and language directions are shown in Figure 9. In comparison to the other languages, the chosen language pair in the paper (En → De) is more conservative than the other language pairs, which achieve higher correlations across most metrics. Most of them achieve $> 50\%$ correlation, though specifically *COMET* appears to be more predictable for other language pairs. These results confirm the main results of predictability of the metric without having access to the reference.

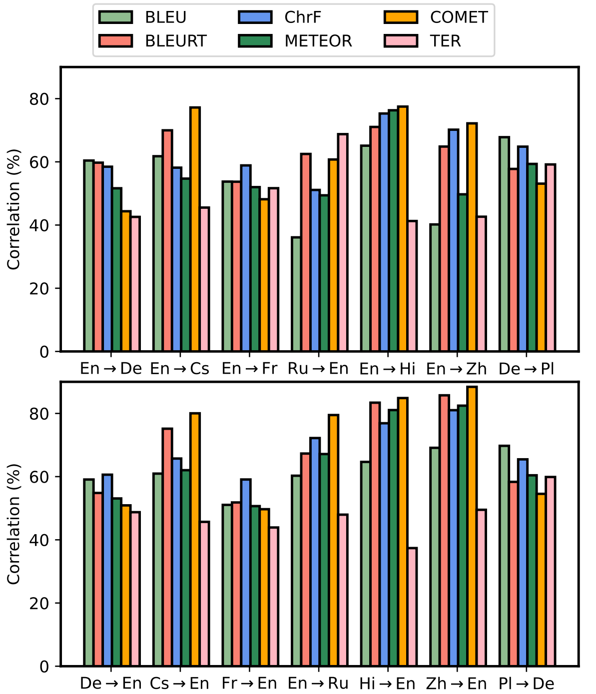


Figure 9: Correlations with metrics and human judgement of main metric estimation (ME) models on other languages. Each bar is a separate model trained to predict a particular metric or human judgement.

D Confidence Estimation for Metric Estimation

Quantifying the confidence of the prediction can be crucial in downstream applications. For example, if a quality estimator has low confidence or is predicted to not be accurate, the decision regarding the quality of translation should be delegated to humans. We attempt to do this by training an auxiliary model to predict the confidence of the output score. We do this by training a logistic regression classifier which takes as input the final layer (λ) of our metric estimator and is trained to predict the binary label: if the predicted metric $f_{\mathrm{BLEU}}(s,h,r)$ is close to the true metric $\mathrm{ME}(s,h)$.

$$\mathbb{P}(|\mathrm{ME} - f_{\mathrm{BLEU}}| \in [0, \mathrm{ME}(s,h) \times 10\%]) \\ = \sigma(W^T \cdot \lambda + b)$$

We, unfortunately, find that the classifier suffers from a very low accuracy 63.4% against a most common class (negative) baseline of 51.6%. It therefore cannot be meaningfully used to ascertain when our regressor is correct and when it is not. Glushkova et al. (2021) propose a more complex solution and exploit uncertainty methods for MT metric and quality estimation systems.

---

[8] WMT14 (Bojar et al., 2014), Opus Paracrawl (Tiedemann, 2012; Bañón et al., 2020), CCMatrix (Schwenk et al., 2021; Fan et al., 2021).

Integrating Translation Memories into Non-Autoregressive Machine Translation

Jitao Xu† **Josep Crego**‡ **François Yvon**†

†Université Paris-Saclay, CNRS, LISN, 91400, Orsay, France
‡SYSTRAN, 5 rue Feydeau, 75002, Paris, France
{jitao.xu,francois.yvon}@limsi.fr, josep.crego@systrangroup.com

Abstract

Non-autoregressive machine translation (NAT) has recently made great progress. However, most works to date have focused on standard translation tasks, even though some edit-based NAT models, such as the Levenshtein Transformer (LevT), seem well suited to translate with a Translation Memory (TM). This is the scenario considered here. We first analyze the vanilla LevT model and explain why it does not do well in this setting. We then propose a new variant, TM-LevT, and show how to effectively train this model. By modifying the data presentation and introducing an extra deletion operation, we obtain performance that are on par with an autoregressive approach, while reducing the decoding load. We also show that incorporating TMs during training dispenses to use knowledge distillation, a well-known trick used to mitigate the multimodality issue.

1 Introduction

Non-autoregressive neural machine translation (NAT) has been greatly advanced in recent years (Xiao et al., 2022). NAT takes advantage from parallel decoding to generate multiple tokens simultaneously and speed up inference. This is often at the cost of a loss in translation quality when compared to autoregressive (AR) models (Gu et al., 2018a). This gap is slowly closing and methods based on iterative refinement (Ghazvininejad et al., 2019; Gu et al., 2019; Saharia et al., 2020) and on connectionist temporal classification (Libovický and Helcl, 2018; Gu and Kong, 2021) are now reporting BLEU scores similar to strong AR baselines.

Most works on NAT focus on the standard machine translation (MT) task, where the decoder starts from scratch, with the exception of Susanto et al. (2020); Xu and Carpuat (2021), who use NAT to integrate lexical constraints in decoding. However, edit-based NAT models, such as the Levenshtein Transformer (LevT) of Gu et al. (2019), seem to be a natural candidate to perform MT with Translation Memories (TM). LevT is able to iteratively edit an initial target sequence by performing insertion and deletion operations until convergence. This design also matches the concept of using TMs in MT, where given a source sentence, we aim to edit a candidate translation retrieved from the TM.

This idea has been used for decades in the localization industry and implemented into basic Computer-Aided Translation tools. Translators wishing to translate a sentence can benefit from fuzzy matching techniques to retrieve similar segments from the TM. These segments can then be revised, thereby improving productivity and consistency of the translation process (Koehn and Senellart, 2010; Yamada, 2011). The retrieval of similar examples from a TM has also proved useful in conventional (AR) neural MT systems; they can be injected into the encoder (Bulte and Tezcan, 2019; Xu et al., 2020) or as priming signals in the decoder (Pham et al., 2020) to influence the translation process. These studies report significant gains in translation performance in technical domains, where the translation of terms and phraseology greatly benefits from examples found in a TM.

Our main focus in this work is to develop an improved version of LevT suited to the *revision part* of TM use, where the translation retrieved from TM is modified via edit operations in a non-autoregressive way. We first show that the original LevT cannot perform well on this task and explain that this failure is a direct consequence of its training design. We propose to fix this issue with TM-LevT, which includes an additional deletion step. Next, we propose to further improve the training procedure in two ways: (a) by also including the retrieved candidate translation on the source side, as done in AR TM-based approaches (Bulte and Tezcan, 2019; Xu et al., 2020); (b) by simultaneously training with empty and non-empty initial target sentences. In our experiments, TM-LevT achieves performance that is on par with a strong AR ap-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1326–1338
May 2-6, 2023 ©2023 Association for Computational Linguistics

proach on various domains when translating with TMs, with a reduced decoding load. We also observe that incorporating an initial translation both on the source and target sides makes Knowledge Distillation (KD, Kim and Rush, 2016) useless. This contrasts with standard NAT models, which rely on KD to alleviate the multimodality issue (Gu et al., 2018a). As far as we know, this work is the first to study NAT with TMs in a controlled setting.

Our contributions are the following: (a) we show that the original LevT training scheme is not suited to edit similar translations from a TM; (b) we propose a variant of LevT, TM-LevT with an improved training procedure, which yields performance that are close, or even similar to AR approaches when translating with good TM matches, with a reduced decoding load; (c) we highlight the benefits of multi-task training (with and without TMs) to attain the best performance; (d) we discuss the reasons why KD hurts the training of NAT with TMs.

2 Using Translation Memories in NAT

2.1 Background

TM Retrieval Given a source sentence $\mathbf{f}$, we aim to retrieve a good match $\tilde{\mathbf{e}}$ from the TM. For this, we search the TM for a pair of sentences $(\tilde{\mathbf{f}}, \tilde{\mathbf{e}})$, where $\tilde{\mathbf{f}}$ is similar to $\mathbf{f}$. The corresponding target $\tilde{\mathbf{e}}$ is then used to initiate the translation of $\mathbf{f}$. We compute the similarity between $\mathbf{f}$ and $\tilde{\mathbf{f}}$ as:

$$\mathrm{sim}(\mathbf{f}, \tilde{\mathbf{f}}) = 1 - \frac{\mathrm{ED}(\mathbf{f}, \tilde{\mathbf{f}})}{\max(|\mathbf{f}|, |\tilde{\mathbf{f}}|)}, \qquad (1)$$

where $\mathrm{ED}(\mathbf{f}, \tilde{\mathbf{f}})$ is the edit distance between $\mathbf{f}$ and $\tilde{\mathbf{f}}$, and $|\mathbf{f}|$ is the length of $\mathbf{f}$. The intuition is that the closer $\mathbf{f}$ and $\tilde{\mathbf{f}}$ are, the more suitable $\tilde{\mathbf{e}}$ will be. As is custom, we only use TM matches when the similarity score exceeds a predefined threshold, otherwise we translate from scratch. We discuss the effect of the match similarity in Section 4.5.

Levenshtein Transformer LevT is an edit-based NAT model proposed by Gu et al. (2019). It performs translation by iteratively editing an initial target sequence with insertion and deletion operations until convergence. The insertion operation is composed of a *placeholder insertion module* and a *token predictor*. The placeholder classifier predicts the number of additional tokens that need to be inserted between any two consecutive tokens in its input sequence. The token predictor then generates a token for each placeholder position. The deletion operation aims to detect prediction errors made by the model. It makes a binary decision for each token, indicating whether it should be deleted or kept. During training, a noised initial target sequence $\mathbf{e}'$ is first generated by randomly dropping tokens from the reference $\mathbf{e}$. The insertion modules learn to reinsert the deleted tokens into $\mathbf{e}'$. The deletion operation is then trained to erase erroneous predictions made during insertion.

During inference, LevT starts with an empty target sequence ($\mathbf{e}' = []$) and generates the translation by alternatively performing deletion and insertion operations until convergence or a maximum number of decoding rounds is reached. In the first iteration, the deletion is omitted, as no tokens can be deleted from the empty sequence. This iterative refinement procedure converges when the input and output of one iteration are the same, either because LevT predicts nothing to delete and to insert, or because it enters a loop where the deleted tokens are reinserted in the same round. Unlike almost all other NAT models, LevT does not require any external prediction of the target length, as the number of target tokens is iteratively revised and adjusted by the placeholder prediction module. We refer to Gu et al. (2019) for more details about LevT.

2.2 Deficiencies of LevT Training

Even though the edit-based nature of LevT makes it readily able to translate with TMs, it has mostly been applied to standard MT, where the decoder starts with an empty sentence.[1] This is consistent with the overall training scheme, illustrated in Figure 1 (Vanilla LevT), where inputs for the placeholder insertion module are always subsequences of the reference and the deletion module only sees the outputs of the previous token insertion step.

Settings	Empty	Random Sent	Shuffle Ref
Init	-	1.3	5.0
LevT	45.4	2.1	40.2
LevT vs Init	-	90.4	9.4

Table 1: BLEU scores of LevT decoding with various target initialization. *Empty* refers to standard LevT inference with an empty start. *Random Sent* uses a random sentence as initial target. *Shuffle Ref* starts with a random shuffle of the reference translation. *Init* reports the BLEU score of the initialization, while *LevT vs Init* compares LevT's outputs with their starting points.

To illustrate the deficiency of this training

---

[1] One notable exception is the attempt in Gu et al. (2019) to perform automatic post-editing through iterative revisions.

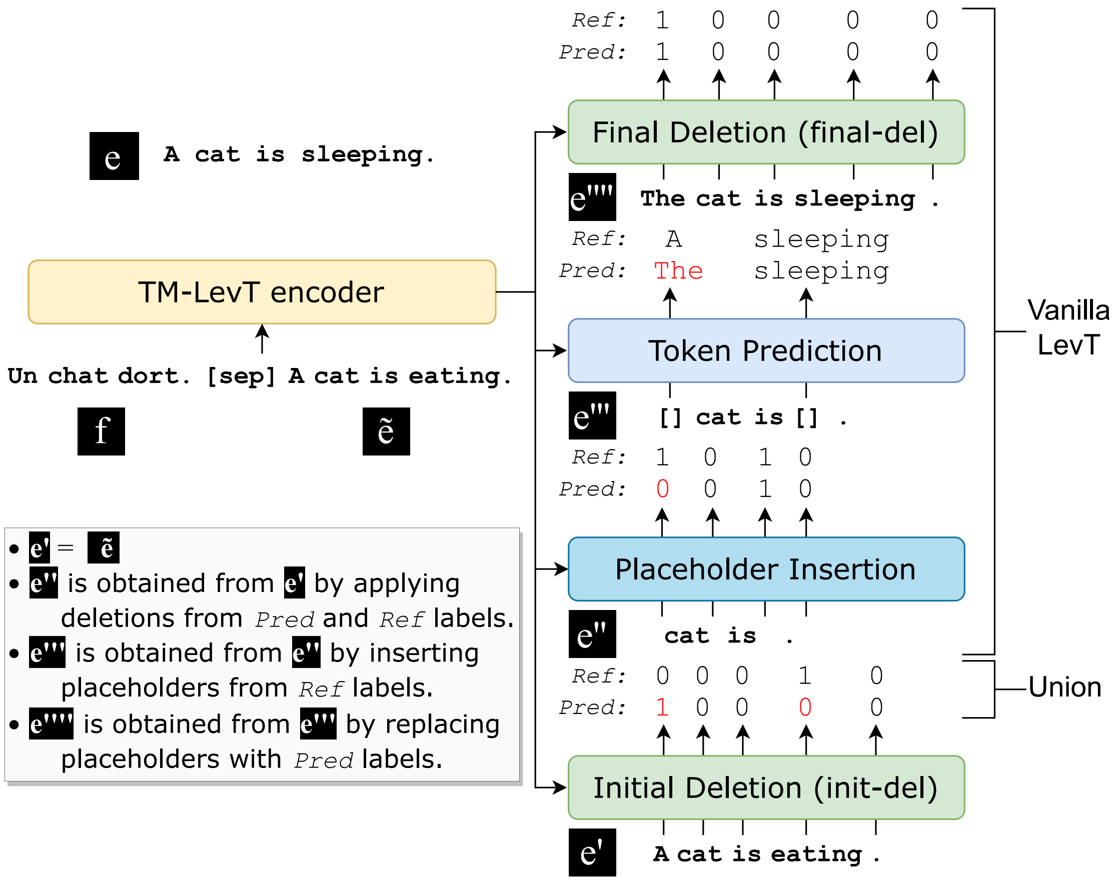


Figure 1: A complete training step for TM-LevT. Compared to the original LevT which starts training from $\mathbf{e}''$, TM-LevT adds the init-del step to delete unrelated tokens from a TM match. Figure better viewed in color.

scheme, we learned a vanilla LevT model using the datasets of Section 3.1 and initialized the decoder with a sentence randomly selected from the test set and totally unrelated to the source sentence. We observe (Table 1, Random Sent) that LevT's outputs are almost as bad as their starting point. This is because the deletion module fails to delete irrelevant input words, presenting the insertion modules with a fully fluent yet fully inadequate sequence that the insertion module is hard-pressed to revise. This contrasts with the *Shuffle Ref* scenario, where the decoder starts with a random shuffle of the reference. LevT can now make changes during the iterative refinement and generates translations (40.2 BLEU) that are close to standard decoding (45.4 BLEU). The TM-based scenario discussed below presents the same challenge for the deletion module, that of spotting and deleting irrelevant words. Our proposal will first focus on fixing this issue.

2.3 Improving Editions with TM-LevT

The experiment of previous section suggests that LevT will have issues editing TM matches, as they often contain tokens that are unrelated to the source and should be removed (see Figure 1 for an example TM match $\tilde{\mathbf{e}}$ containing an unrelated word "eating"). The distribution of unrelated tokens may greatly differ from token prediction errors made by LevT, which are tokens LevT is trained to delete.

We propose a variant of LevT denoted TM-LevT, where we include an extra deletion step (init-del) that applies before the insertion modules. As shown in Figure 1, init-del is trained to detect unrelated tokens from the initial $\mathbf{e}'$, whereas the final deletion (final-del) focuses on prediction errors. During training, we generate examples for the insertion modules by removing from $\mathbf{e}'$ tokens that either are not in the reference, or should be deleted according to the init-del operation. The resulting subsequence $\mathbf{e}''$ is then used to train the insertion operation. TM-LevT does not change the number of parameters, as we use the same classifier for the init-del and final-del steps. During inference, TM-LevT behaves exactly as LevT, iteratively applying deletions and insertions to an initial candidate translation.

2.4 Translating with or without TM Matches

In practical applications, two modes of operations need to be supported. The first is when a good match is found in the TM and used to initialize the decoding ($\mathbf{e}' = \tilde{\mathbf{e}}$). Revising $\mathbf{e}'$ may imply to delete or insert tokens, which is what the system is trained for. It may also imply to move words around, which is achieved by a succession of deletion and insertion. However, as these operations are performed independently, there is no guarantee that the deleted words will be memorized for subsequent insertions, causing the loss of relevant words in the process. This is again illustrated in Table 1 (Shuffle Ref) where we see that even when given all the reference tokens (in random order), LevT still underperforms translating from scratch. To mitigate this risk, we augment the source side with the initial TM match, ensuring that $\tilde{\mathbf{e}}$ is always fully available to the decoder. Following the proposal of Bulte and Tezcan (2019); Xu et al. (2020) for AR models, this is performed by concatenating $\mathbf{f}$ and $\tilde{\mathbf{e}}$ on the encoder side (see Figure 1).

The second mode of operation is when no appropriate match is found, causing the system to fall back to a standard MT regime. In order to handle both situations in a single model, we resort to multitask training and prepare our training samples as follows: with probability $p = 0.5$, we decide either to decode with a retrieved TM match $\tilde{\mathbf{e}}$ or from scratch. In the former case, the decoder is initialized with $\tilde{\mathbf{e}}$, while in the latter, we use a noised subsequence $\mathbf{e}'$ generated as in (Gu et al., 2019). TM-LevT is then jointly trained on both tasks.

3 Experiments

3.1 Datasets

Our experiments use the same corpus as Xu et al. (2020), and contains texts from a diverse set of 11 domains for the English-French direction, downloaded from OPUS[2] (Tiedemann, 2012): documents from the European Central Bank (ECB); from the European Medicines Agency (EMEA); Proceedings of the European Parliament (Epps); legislative texts of the European Union (JRC); News Commentaries (News); TED talk subtitles (TED); parallel sentences extracted from Wikipedia (Wiki); localization files (GNOME, KDE and Ubuntu) and manuals (PHP). We include both technical domains, for which good matches are likely to exist, and more "general" domains (Epps and News), for which useful TM matches are harder to find, allowing us to explore the benefits of using TMs for a variety of conditions. All these data are deduplicated prior to training.

For each source sentence, we retrieve from the same domain the top 3 TM matches according to the similarity score of Equation (1), further requiring a score of $0.4 \leq \mathrm{sim}<1$. For each domain, we prepare two test sets with $1{,}000$ sentences each: one contains randomly selected sentences with a close match ($\mathrm{sim}>0.6$) in the TM, the other with an acceptable match ($\mathrm{sim}\in[0.4, 0.6]$). We also leave a held-out set of $1{,}000$ sentences per domain, for which no matches of $\mathrm{sim}\geq 0.4$ are found. The remaining data is used for training. Note that the ratio of sentences with at least one TM match greatly varies across domains. Detailed statistics about these corpora are in Appendix A. We use all retrieved TM matches (up to 3) for training and only the best match for test. Therefore, a source training sentence with 3 TM matches yields 3 training samples. The initial set of 4.4M parallel data is thus extended with about 5M examples augmented with a TM match. We tokenize all data using the Moses tokenizer[3] and build a shared source-target vocabulary with 32K Byte Pair Encoding units (Sennrich et al., 2016) learned with `subword-nmt`.[4]

3.2 Experimental Settings

We compare TM-LevT with a strong AR approach (Bulte and Tezcan, 2019) and the original LevT model.[5] These baselines use the same training data as TM-LevT, and also process examples with and without TM matches. For the AR model, TM matches only appear concatenated to the source sentence and translation always starts from scratch; for LevT, we test both cases where the decoder is initialized with and without TM matches.

TM-LevT is based on the Transformer architecture (Vaswani et al., 2017), implemented with fairseq[6] (Ott et al., 2019).[7] We use a hidden size of 512 and a feedforward size of $2{,}048$, optimizing with Adam with a maximum learning rate of 0.0005, an inverse square root decay schedule, and

---

[2] `https://opus.nlpl.eu/`

[3] `https://github.com/moses-smt/mosesdecoder`

[4] `https://github.com/rsennrich/subword-nmt`

[5] `https://github.com/facebookresearch/fairseq/tree/main/examples/nonautoregressive_translation`

[6] `https://github.com/pytorch/fairseq`

[7] Our implementation is open-sourced at `https://github.com/jitao-xu/tm-levt`.

w/o TM	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	59.8	64.5	34.4	70.3	67.6	55.3	12.0	38.6	30.8	51.6	47.4	52.6
AR	58.7	53.8	55.8	55.0	68.8	53.9	27.1	18.2	62.0	54.0	65.0	51.2
LevT	46.6	30.7	51.8	51.0	62.3	47.0	23.6	12.5	58.7	50.0	61.9	46.5
TM-LevT	53.0	49.7	53.2	51.5	64.7	50.8	24.5	37.1	59.5	50.4	64.0	52.6
w/ TM	**ECB**	**EMEA**	**Epps**	**GNOME**	**JRC**	**KDE**	**News**	**PHP**	**TED**	**Ubuntu**	**Wiki**	**All**
AR	71.9	72.0	58.9	80.1	83.2	67.3	28.8	44.7	63.3	67.6	68.6	67.1
LevT	62.4	53.8	55.5	77.5	78.8	63.3	26.1	28.7	60.2	66.0	67.1	60.4
+tgt TM	60.2	63.8	34.8	69.6	67.7	54.5	12.5	38.8	31.1	52.1	47.5	52.8
TM-LevT	69.8	72.2	56.0	78.1	82.2	68.2	26.0	44.1	60.3	66.3	68.7	65.9

Table 2: BLEU scores for each domain when performing translation without and with TMs for $\text{sim} > 0.6$. *All* is computed by concatenating all test sets (11k sentences in total). *Copy* refers to using the TM match $\tilde{\mathbf{e}}$ as the output.

10,000 warm-up steps. We share the decoder parameters for both two deletions and the insertion operation and also tie all input and output embedding matrices (Press and Wolf, 2017). We train TM-LevT for 300k iterations with a batch size of 8,192 tokens per GPU on 8 V100 GPUs. For inference, we use a maximum iteration number of 10 for TM-LevT and LevT, and a beam size of 5 for the AR decoder. We use a batch size of 8,192 tokens and perform inference on one single GPU for all compared models. The vanilla LevT model is trained similarly, while the AR model (Bulte and Tezcan, 2019) is trained with a maximum learning rate of 0.0007, with 4,000 warmup steps for 300k iterations on 4 V100 GPUs. We report results of a *do-nothing* baseline which simply copies the TM matches as outputs. Performance is measured with BLEU using SacreBLEU[8] (Post, 2018) and with COMET (Rei et al., 2020).

4 Results and Analyses

4.1 NAT Can Benefit from TMs

We evaluate the performance of standard MT and TM-based MT on the two test sets ($\text{sim}{>}0.6$ and $\text{sim}{\in}[0.4, 0.6]$) introduced in Section 3.1. When performing standard MT, the source side only contains the source sentence for all models, and the decoder side of TM-LevT is initialized with an empty input. When translating with TMs, the TM match is concatenated to the source sentence for all models. TM-LevT is additionally initialized with the TM match on the decoder side, a setting we also consider for LevT (+tgt TM). Table 3 reports the aggregated results computed on all domains (11k sentences). The results for each domain when translating with and without TMs on $\text{sim}{>}0.6$ are in Table 2. The corresponding results with a breakdown by domain for $\text{sim}{\in}[0.4, 0.6]$ are in Appendix B.

As reported in Tables 3 and 2, the AR with TM baseline yields much higher BLEU and COMET scores than the standard MT setting. LevT can also make good use of TM matches, but its performance lags way behind the AR strategy in both settings.[9] Scores in Table 2 show that for domains like ECB and EMEA, it is difficult for LevT to generate good translations without using TMs, while for more general domains like News, the performance gap between LevT and AR are less significant.

	$\text{sim} > 0.6$		$\text{sim} \in [0.4, 0.6]$	
BLEU ↑	w/o TM	w/ TM	w/o TM	w/ TM
copy	-	52.6	-	34.5
AR	51.2	67.1	46.1	55.7
LevT	46.5	60.4	40.8	49.3
+tgt TM	-	52.8	-	35.0
TM-LevT	52.6	65.9	45.7	53.3
COMET ↑	w/o TM	w/ TM	w/o TM	w/ TM
copy	-	0.1330	-	-0.3784
AR	0.6143	0.6985	0.5379	0.5900
LevT	0.4251	0.5767	0.3429	0.4404
+tgt TM	-	0.1639	-	-0.3478
TM-LevT	0.5314	0.6454	0.4263	0.4889

Table 3: BLEU and COMET scores on multi-domain test sets for various TM similarity ranges. *w/o TM* is standard MT, *w/ TM* adds a retrieved match $\tilde{\mathbf{e}}$ on the source side, and use it as initial target for TM-LevT. *+tgt TM* refers to using TM match as the initial target for LevT.

TM-LevT, on the contrary, does remarkably well when translating from scratch, even surpassing the

---

[8] SacreBLEU signature: BLEU+case.mixed+lang.en-fr+numrefs.1+smooth.exp+tok.13a+version.1.5.1

[9] We use the `fairseq` source code released by Gu et al. (2019) to train the LevT model. We have performed a sanity check by training a LevT model on the WMT14 English-German data and obtained results that are about 2 BLEU points below the scores reported in Gu et al. (2019). As Gu et al. (2019) have not specified the tool used to compute BLEU scores, it is difficult to make a more precise comparison.

AR model on BLEU on the $\mathrm{sim}>0.6$ set, which is arguably easier. When using TM matches, TM-LevT also performs much better than LevT. We achieve BLEU scores of only 1.2 and 2.4 below AR for $\mathrm{sim}>0.6$ and $\mathrm{sim}\in[0.4, 0.6]$, respectively (Table 3). The effect of using TM matches greatly varies across domains. TM-LevT can improve more general domains like News and Wiki when using TMs (Table 2), even though the performance gains are less significant than for more specific domains like ECB and KDE. However, the same trend is also observed for AR, and TM-LevT even surpasses AR on Wiki in Table 2. The gap in COMET score between TM-LevT and AR is also much smaller than reported by Helcl et al. (2022), indicating that TM-LevT outputs valid translations.

Unrelated Tokens AR approaches are known to improperly copy "unrelated tokens" from TM matches into the output (Xu et al., 2020). As TM-LevT includes the deletion operation, we expect it to properly delete unrelated tokens. We define unrelated tokens as those present in $\tilde{\mathbf{e}}$ but not in $\mathbf{e}$ and count the ratio of such tokens that appear in the final translation. This is different from Xu et al. (2020), as they used a word alignment model to label tokens that are unrelated to the "source". We have re-implemented the same method, but the alignment model was never perfect and yielded additional errors, which led to an imprecise measure of unrelated tokens. Table 4 reports the ratio of such unrelated tokens: TM-LevT is slightly less prone than AR to recopy unrelated parts of the TM matches in both test sets. LevT does even better on that account, but its comparatively lower BLEU scores suggest that it also discards valid tokens.

Unrelated rate ↓	$\mathrm{sim}>0.6$	$\mathrm{sim}\in[0.4, 0.6]$
AR	28.42	17.78
LevT	21.39	13.74
TM-LevT	26.67	16.56

Table 4: Percentage of unrelated tokens from the retrieved TM matches appearing in the final translations.

4.2 Ablation Analysis

We conduct an ablation analysis to study the effectiveness of each component of our method, by training a new model for each contrast. Training without TM matches on the target side (Table 5, -tgt TM) vastly degrades the performance in both conditions (w/ and w/o TM), indicating that the standard MT can also benefit from training with TMs as initial targets. However, removing TM matches on the source side (-src TM) improves standard MT, as also observed by Bulte and Tezcan (2019), but has a negative impact when translating with TMs. This highlights the importance of always remembering the TM match on the source side of TM-LevT.

	$\mathrm{sim}>0.6$		$\mathrm{sim}\in[0.4, 0.6]$	
BLEU	w/o TM	w/ TM	w/o TM	w/ TM
TM-LevT	52.6	65.9	45.7	53.3
-tgt TM	46.6	60.7	40.7	49.6
-src TM	53.2	64.3	45.9	52.2
-final-del	38.5	64.2	32.7	50.8
-self-pred	52.6	65.2	45.6	52.7

Table 5: BLEU scores for various configurations. *-tgt TM* (resp. *-src TM*) is the model trained without TM match on the target (resp. source) side. *-final-del* is trained without the final-del operation, *-self-pred* only applies reference deletions during training.

We also compare with alternative implementations of the deletion operation. Results in Table 5 (-final-del) show that removing the final deletion step mostly impacts TM-LevT in the standard MT setting, where the detection of wrong predictions matters most (Huang et al., 2022). This further demonstrates that unrelated tokens from TM matches and the prediction errors of the model are vastly different, and training to delete both is necessary. Last, we experiment with using only reference deletion labels to train the insertion operation, instead of using both the reference and model predictions (see Section 2.3). We observe (-self-pred) a small performance drop with respect to the baseline policy.

4.3 Knowledge Distillation

KD is used in most NAT models, as it reduces the complexity and lexical diversity of target sentences, thereby helping NAT approaches to mitigate the multimodality issue (Zhou et al., 2020; Xu et al., 2021). Given that our results so far have only relied on actual target data, we thus ask whether KD could also improve the performance of TM-based NAT. We train a teacher Transformer-based model with the 4.4M parallel data and use it for data distillation. As expected, using KD does improve BLEU scores of TM-LevT on standard MT (Table 6). However, using KD hurts performance when editing an initial similar translation, resulting in a large drop in scores compared to using real data. Applying KD also to TM matches yields similar results.

The benefits of KD are assumed to mainly reduce

SRC	Measures to reduce or eliminate releases from unintentional production
TM match $\text{sim}=0.73$	Measures to reduce or eliminate releases from intentional production and use Mesures propres à réduire ou éliminer les rejets résultant d'une production et d'une utilisation intentionnelles
TM-LevT +TM	Mesures visant à réduire ou à éliminer les disséminations de la production non intentionnelle Mesures **propres** à réduire **ou éliminer** les **rejets résultant d'une** production non intentionnelle
KD +TM	Mesures visant à réduire ou à éliminer les rejets de la production non intentionnelle Mesures **visant** à réduire ou **à** éliminer les rejets **provenant de la** production non intentionnelle
REF	Mesures propres à réduire ou éliminer les rejets résultant d'une production non intentionnelle

Figure 2: An example with the retrieved TM match and translations generated by TM-LevT and KD model. Benefits taken from the TM match by TM-LevT are in blue. Segments KD model fails to make use of are in red.

	$\text{sim} > 0.6$		$\text{sim} \in [0.4, 0.6]$	
BLEU	w/o TM	w/ TM	w/o TM	w/ TM
copy	-	52.6	-	34.5
Teacher	56.7	-	49.6	-
AR + KD	55.7	56.9	48.7	49.4
TM-LevT	52.6	65.9	45.7	53.3
+KD	54.3	57.1	47.6	49.3
+KD TM	53.8	56.0	47.3	48.5

Table 6: BLEU scores with and without KD. *Teacher* is the teacher model used to distill the parallel data. *+KD* applies KD to the training references, *+KD TM* applies KD to both references and TM matches.

the multimodality issue in NAT (Zhou et al., 2020). This issue may be less problematic in our context, as the TM match already provides an explicit and often unambiguous context for generating the missing words in the translation. In this case, KD is even detrimental to the translation quality. This is because using distilled references exposes TM-LevT to imperfect translations (with BLEU scores of respectively 56.7 and 49.6), only a few points better than the initial TM matches (*copy* in Table 6). This seems to limit TM-LevT's ability in learning to generate very high quality translation that it can achieve when exposed to real references (+8.8 and +4 BLEU on $\text{sim}>0.6$ and $\text{sim}\in[0.4, 0.6]$, respectively). For comparison, we also train an AR model using the same KD data (AR + KD) and again observe very little gain when translating with TMs. In fact, the limit of models using KD data, whether using TMs or not, seems to be upper-bounded by the performance of the teacher. These intuitions are illustrated by the example in Figure 2.

4.4 Computational Trade-offs of TM-LevT

Inference speedup is the main advantage of using NAT models. Table 7 compares the average decoding time per sentence on all domains.[10] Here, we perform the inference speed analysis as a sanity check. As discussed by Kasai et al. (2021), Helcl et al. (2022) and Schmidt et al. (2022), comparing the inference speed for NAT models could be tricky. We follow here the recommendations of Helcl et al. (2022) and use the same hardware conditions and inference batch size for all settings, making our results as comparable as possible. TM-LevT is much faster than AR both with and without TMs. We also note that decoding with a TM match is always slightly longer due to (a) finding matches; (b) encoding a longer input made of the source and the TM match.[11]

Settings	AR	TM-LevT	Speedup
w/o TM	5.91	2.53	$\times 2.34$
w/ TM	7.80	3.43	$\times 2.28$

Table 7: Average decoding time (ms) per sentence for all domains of both $\text{sim} > 0.6$ and $\text{sim} \in [0.4, 0.6]$.

	$\text{sim} > 0.6$		$\text{sim} \in [0.4, 0.6]$	
Systems	w/o TM	w/ TM	w/o TM	w/ TM
LevT	2.027	1.899	2.544	2.538
TM-LevT	1.781	1.348	2.260	1.880

Table 8: Average number of decoding iterations.

Using TMs in MT is expected to improve the translation quality while also reducing the decoding load, as useful tokens can be directly copied to the output. This is not observed in AR approaches nor in the original LevT, as both models always start inference with an empty input. TM-LevT, however, uses an initial translation to speed up decoding. We report the average number of iterations required in inference in Table 8, where we see that translating with TMs reduces the decoding effort for TM-LevT by about 20%, while it remains almost unchanged for LevT. We also find that TM-LevT needs fewer

---

[10] We exclude PHP, for which AR generates many repetitions, yielding very long runtimes that biased the average.

[11] The numbers in Table 7 only reflect the effect of (b), as step (a) is the same for both AR and NAT models.

iterations to converge than LevT in all conditions. Note that the training time of TM-LevT is only 1.1-1.2× compared to LevT, which we think is an acceptable overhead considering the large performance gains and the reduction of decoding load.

4.5 Good Matches Increase MT Quality

TM-based models require good TM matches to improve their translations; when none is found, standard MT can be used instead. Defining the minimal similarity for a match to be useful (0.4 in this paper) requires some tuning and the best threshold may vary from corpora to corpora. In this section, we take a closer look at the effect of thresholding for the AR and NAT models considered here. We compute the best TM match for the held-out set of Section 3.1, trying to find a matched translation for all sentences without any filtering. We then combine the held-out set with the two test sets ($\mathrm{sim}{>}0.6$ and $\mathrm{sim}{\in}[0.4, 0.6]$) and bucket sentences by the similarity of the best TM match. For each bucket, we compute BLEU scores obtained for various systems and plot results on Figure 3.

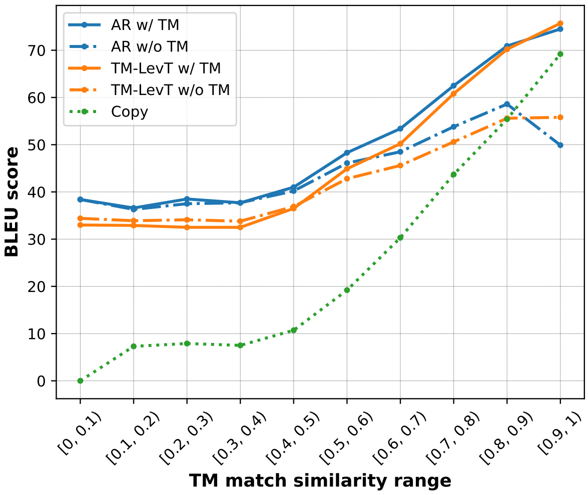


Figure 3: BLEU scores when translating with TM matches of various similarity ranges. The *Copy* strategy uses the retrieved example as the final translation.

We first observe that the scores of all systems, even those that do not use TMs, increase with better TM matches: this is because test sentences with better matches are also more similar to the training data, thus easier to translate for all systems. Secondly, even with very good matches, both models using TMs are able to improve over the *Copy* strategy. Comparing AR to TM-LevT, we see that the former is preferable across the board, even though the gap closes when very good matches are available. The TM-based AR model is almost as good as standard MT for poor matches and starts improving standard MT when $\mathrm{sim}{\geq}0.4$. In comparison, we see for TM-LevT a small edge of standard MT over translating with TMs which subsists as long as $\mathrm{sim}{<}0.5$; for higher similarity matches, translating with TMs gets much better scores. These results suggest that thresholding may not be necessary and that both TM-based architectures adapt their behavior to the match quality, with hardly any performance loss w.r.t. the standard MT approach.

5 Related Work

TM-based MT Most studies using TM sentences to improve translations are based on AR models and either use a second encoder to integrate the TM match or concatenate it to the source in the same encoder. The former approach is illustrated by Gu et al. (2018b), who rerun the translation model as an encoder to encode the similar translation. Xia et al. (2019) alternatively use a compact graph as the second encoder. Another work along this line is He et al. (2021), which encodes the TM match using the decoder embedding matrix and performs a cross-attention between the decoder input and the encoded TM match. Besides, Cai et al. (2021) directly search in a corpus of target sentences with a cross-lingual similarity and encode the resulting sentences with a dual encoder approach similar to Junczys-Dowmunt and Grundkiewicz (2018). Single encoder approaches are first explored by Bulte and Tezcan (2019), who concatenate the TM match with the source to perform TM-based MT. This idea is extended by Xu et al. (2020) by adding a second embedding feature to distinguish related and unrelated tokens and by Pham et al. (2020), who use both source and target sentences of the matched TM. Zhang et al. (2018) explore a different direction to improve translation with retrieved segments instead of complete sentences. Khandelwal et al. (2021) further propose k-nearest neighbor MT by searching for target tokens that have similar contextualized representations at each decoding step, an approach continued by Zheng et al. (2021) with dynamic neighborhoods.

NAT with Augmented Resources Several works have studied ways to integrate extra information into NAT architectures, mostly using the LevT model as their starting point. Susanto et al. (2020) incorporate lexical constraints with LevT by simply initializing the decoder with constraint words inserted in a predefined order; this limitation is lifted in the EDITOR model of Xu and Carpuat (2021), who introduce a repositioning operation to allow constraints to be inserted in arbitrary order. Zeng et al. (2022) pay particular attention to low-frequency constraints by preventing rare tokens from being removed when generating training samples for the insertion operation. The most relevant study to this paper is the recent work by Niwa et al. (2022), who also seek to improve LevT with TMs, using good matches to initialize the decoder. This work only mildly departs from the vanilla LevT with a small modification of the deletion operation to remove unrelated tokens and only compare with standard MT, failing to contrast their improvements with TM-based AR models. Finally, Xu et al. (2022) also explore the integration of TMs into the original LevT model, but fail to obtain improvements over a copy baseline.

6 Conclusion

In this paper, we studied ways to augment the LevT architecture with TMs. Our proposal adds an initial deletion operation during training to detect possible unrelated tokens present in TM matches. By copying the TM match both on the source side and on the target side as an initial target sequence, our model vastly outperformed the original LevT model and achieved BLEU scores approaching those of a strong AR model both when decoding from scratch and when editing a TM match. Compared to LevT, TM-LevT also generates translations that contain less unrelated tokens, and is able to converge in fewer iterations. We also found that training with TMs improves NAT performance on standard MT. Finally, we have tried to combine KD with our approach, concluding that it was more hurting than helpful for TM-based architectures.

Limitations

NAT models such as LevT are more difficult to train than AR models, as they require larger batch size to converge. Our TM-LevT adds an initial deletion operation during training, therefore slightly lengthening the training time by approximately $1.1 - 1.2\times$ with respect to the basic LevT model. Due to computational limits, we have not conducted experiments on other language pairs, especially on more distant language pairs. Even tough our findings apply for a wide range of domains, considering also more languages would be helpful to fully validate our observations.

Acknowledgement

The authors would like to thank Maxime Bouthors for his suggestions and Ruiyang Zhou for her help of the deficiency analysis of LevT. We would also like to thank the anonymous reviewers for their valuable suggestions and comments. This work was granted access to the HPC resources of IDRIS under the allocation 2022-[AD011011580R2] made by GENCI. The first author is partly funded by SYSTRAN and by a grant (Transwrite) from the Région Ile-de-France.

References

Bram Bulte and Arda Tezcan. 2019. Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1800–1809, Florence, Italy. Association for Computational Linguistics.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7307–7318, Online. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. 2018a. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133, Online. Association for Computational Linguistics.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, volume 32, pages 11181–11191. Curran Associates, Inc.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. 2018b. Search engine guided neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Qiuxiang He, Guoping Huang, Qu Cui, Li Li, and Lemao Liu. 2021. Fast and accurate neural machine translation with translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3170–3180, Online. Association for Computational Linguistics.

Jindřich Helcl, Barry Haddow, and Alexandra Birch. 2022. Non-autoregressive machine translation: It's not as fast as it seems. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1780–1790, Seattle, United States. Association for Computational Linguistics.

Xiao Shi Huang, Felipe Perez, and Maksims Volkovs. 2022. Improving non-autoregressive translation models without distillation. In *International Conference on Learning Representations*.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2018. MS-UEdin submission to the WMT2018 APE shared task: Dual-source transformer for automatic post-editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 822–826, Belgium, Brussels. Association for Computational Linguistics.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *Proceedings of the Second Joint EM+/CNGL Workshop: Bringing MT to the User: Research on Integrating MT in the Translation Industry*, pages 21–32, Denver, Colorado, USA. Association for Machine Translation in the Americas.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Ayana Niwa, Sho Takase, and Naoaki Okazaki. 2022. Nearest neighbor non-autoregressive text generation. *CoRR*, abs/2208.12496.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Minh Quang Pham, Jitao Xu, Josep Crego, François Yvon, and Jean Senellart. 2020. Priming neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 462–473, Online. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. Association for Computational Linguistics.

Robin Schmidt, Telmo Pires, Stephan Peitz, and Jonas Lööf. 2022. Non-autoregressive neural machine translation: A call for clarity. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2785–2799, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Raymond Hendy Susanto, Shamil Chollampatt, and Liling Tan. 2020. Lexically constrained neural machine translation with Levenshtein transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3536–3543, Online. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Mengzhou Xia, Guoping Huang, Lemao Liu, and Shuming Shi. 2019. Graph based translation memory for neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7297–7304.

Yisheng Xiao, Lijun Wu, Junliang Guo, Juntao Li, Min Zhang, Tao Qin, and Tie-yan Liu. 2022. A survey on non-autoregressive generation for neural machine translation and beyond. *CoRR*, abs/2204.09269.

Jitao Xu, Josep Crego, and Jean Senellart. 2020. Boosting neural machine translation with similar translations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1580–1590, Online. Association for Computational Linguistics.

Jitao Xu, Josep Crego, and François Yvon. 2022. Bilingual synchronization: Restoring translational relationships with editing operations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8016–8030, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Weijia Xu and Marine Carpuat. 2021. EDITOR: An Edit-Based Transformer with Repositioning for Neural Machine Translation with Soft Lexical Constraints. *Transactions of the Association for Computational Linguistics*, 9:311–328.

Weijia Xu, Shuming Ma, Dongdong Zhang, and Marine Carpuat. 2021. How does distilled data complexity impact the quality and confidence of non-autoregressive machine translation? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4392–4400, Online. Association for Computational Linguistics.

Masaru Yamada. 2011. The effect of translation memory databases on productivity. *Translation research projects*, 3:63–73.

Chun Zeng, Jiangjie Chen, Tianyi Zhuang, Rui Xu, Hao Yang, Qin Ying, Shimin Tao, and Yanghua Xiao. 2022. Neighbors are not strangers: Improving non-autoregressive translation under low-frequency lexical constraints. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5777–5790, Seattle, United States. Association for Computational Linguistics.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1325–1335, New Orleans, Louisiana. Association for Computational Linguistics.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 368–374, Online. Association for Computational Linguistics.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.

A Details of Data Processing

Table 9 reports statistics of the ratio of TM matches for various similarity ranges of the multi-domain dataset described in Section 3.1. These ratios vary greatly across domains.

Domain	Raw	sim > 0.6	sim $\in [0.4, 0.6]$
ECB	195,956	51.73%	14.06%
EMEA	373,235	65.68%	12.65%
Epps	2,009,489	10.12%	25.30%
GNOME	55,391	39.31%	11.06%
JRC	503,437	50.87%	16.67%
KDE	180,254	36.00%	10.81%
News	151,423	2.12%	9.65%
PHP	16,020	34.93%	12.38%
TED	159,248	11.90%	26.64%
Ubuntu	9,314	20.32%	8.26%
Wiki	803,704	19.87%	17.32%
Total	4,457,471	24.27%	20.00%

Table 9: Dataset statistics, with ratios of sentences with at least one TM match for various similarity ranges.

B Detailed Results on Each Domain

BLEU and COMET scores for each domain are in Tables 10, 11, 12, 13. The variation in scores across domains is large, confirming that TM matches can be very beneficial for some technical domains (e.g. ECB, EMEA, GNOME, KDE, JRC), for which we often find good matches that help to greatly increase the performance. On the other hand, News, Wiki and TED yield less matches, and these only help for both the AR approach and TM-LevT when the similarity is high (sim > 0.6).

BLEU	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	59.8	64.5	34.4	70.3	67.6	55.3	12.0	38.6	30.8	51.6	47.4	52.6
AR	58.7	53.8	55.8	55.0	68.8	53.9	27.1	18.2	62.0	54.0	65.0	51.2
LevT	46.6	30.7	51.8	51.0	62.3	47.0	23.6	12.5	58.7	50.0	61.9	46.5
TM-LevT	53.0	49.7	53.2	51.5	64.7	50.8	24.5	37.1	59.5	50.4	64.0	52.6
COMET	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	0.4006	0.4625	-0.0797	0.4893	0.6893	0.1150	-0.6083	-0.1977	-0.4184	0.3296	0.2843	0.1330
AR	0.6333	0.6402	0.8137	0.7190	0.9057	0.5116	0.3241	-0.0556	0.7848	0.7031	0.7786	0.6143
LevT	0.4251	0.1322	0.7460	0.6181	0.8291	0.3879	0.2037	-0.6139	0.6912	0.5636	0.6947	0.4251
TM-LevT	0.5637	0.5559	0.7513	0.6355	0.8477	0.4218	0.1660	-0.0980	0.6929	0.5768	0.7335	0.5314

Table 10: BLEU and COMET scores for each domain, the task is **standard MT** with $\mathrm{sim} > \mathbf{0.6}$. *All* is computed by concatenating all test sets (11k sentences in total). *Copy* refers to copying the TM match into the output.

BLEU	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	59.8	64.5	34.4	70.3	67.6	55.3	12.0	38.6	30.8	51.6	47.4	52.6
AR	71.9	72.0	58.9	80.1	83.2	67.3	28.8	44.7	63.3	67.6	68.6	67.1
LevT	62.4	53.8	55.5	77.5	78.8	63.3	26.1	28.7	60.2	66.0	67.1	60.4
+tgt TM	60.2	63.8	34.8	69.6	67.7	54.5	12.5	38.8	31.1	52.1	47.5	52.8
TM-LevT	69.8	72.2	56.0	78.1	82.2	68.2	26.0	44.1	60.3	66.3	68.7	65.9
COMET	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	0.4006	0.4625	-0.0797	0.4893	0.6893	0.1150	-0.6083	-0.1977	-0.4184	0.3296	0.2843	0.1330
AR	0.7288	0.7211	0.8223	0.9143	0.9954	0.6299	0.3318	0.0801	0.7910	0.8610	0.8110	0.6985
LevT	0.5647	0.3384	0.7608	0.8617	0.9355	0.5683	0.2443	-0.2183	0.7203	0.8091	0.7618	0.5767
+tgt TM	0.4086	0.4573	-0.0075	0.5230	0.7062	0.1437	-0.5679	-0.1957	-0.3328	0.3710	0.3008	0.1639
TM-LevT	0.6792	0.7003	0.7591	0.8699	0.9696	0.6106	0.2093	0.0353	0.6923	0.8155	0.7614	0.6454

Table 11: BLEU and COMET scores for each domain, the task is **MT with TMs** with $\mathrm{sim} > \mathbf{0.6}$. *All* is computed by concatenating all test sets (11k sentences in total). *Copy* refers to copying the TM match into the output.

BLEU	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	47.3	47.6	12.7	52.6	53.0	42.7	5.8	29.7	8.2	35.1	13.0	34.5
AR	52.3	52.7	44.7	54.4	64.7	53.2	30.0	17.9	41.7	49.2	42.2	46.1
LevT	40.7	31.4	42.6	51.0	59.8	46.8	27.6	11.9	38.7	45.7	40.2	40.8
TM-LevT	47.9	47.7	41.5	51.6	61.1	50.1	26.8	34.3	38.0	46.8	41.0	45.7
COMET	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	0.0310	0.1527	-0.7608	0.1416	0.1919	-0.1703	-0.9719	-0.6279	-1.1419	-0.1837	-0.8222	-0.3784
AR	0.5229	0.5920	0.7735	0.7048	0.8834	0.5522	0.4688	-0.1819	0.5501	0.6363	0.4157	0.5379
LevT	0.2908	0.1245	0.6996	0.5956	0.8069	0.4140	0.3567	-0.7332	0.3979	0.5194	0.3011	0.3429
TM-LevT	0.4370	0.5231	0.6515	0.6205	0.8116	0.4576	0.2948	-0.2343	0.3655	0.5035	0.2600	0.4263

Table 12: BLEU and COMET scores for each domain, the task is **standard MT** with $\mathrm{sim} \in [\mathbf{0.4}, \mathbf{0.6}]$. *All* is computed by concatenating all test sets (11k sentences). *Copy* refers to copying the TM match into the output.

BLEU	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	47.3	47.6	12.7	52.6	53.0	42.7	5.8	29.7	8.2	35.1	13.0	34.5
AR	62.3	62.8	44.9	69.6	75.4	62.1	29.9	39.2	42.6	58.1	43.9	55.7
LevT	52.3	47.1	42.7	65.7	71.9	57.6	27.5	23.8	39.0	55.0	40.8	49.3
+tgt TM	47.4	48.0	13.2	53.2	53.5	42.9	6.0	29.7	9.1	37.1	13.2	35.0
TM-LevT	59.7	61.9	41.4	68.1	73.0	61.4	26.4	39.1	37.5	56.1	39.7	53.3
COMET	ECB	EMEA	Epps	GNOME	JRC	KDE	News	PHP	TED	Ubuntu	Wiki	All
copy	0.0310	0.1527	-0.7608	0.1416	0.1919	-0.1703	-0.9719	-0.6279	-1.1419	-0.1837	-0.8222	-0.3784
AR	0.5814	0.6607	0.7740	0.8380	0.9220	0.6217	0.4741	-0.1140	0.5543	0.7453	0.4344	0.5900
LevT	0.4283	0.2846	0.6998	0.7697	0.8746	0.5437	0.3660	-0.4900	0.4107	0.6676	0.2910	0.4404
+tgt TM	0.0487	0.1569	-0.7208	0.1883	0.2167	-0.1151	-0.9508	-0.6205	-1.0949	-0.1234	-0.8100	-0.3478
TM-LevT	0.5102	0.6281	0.6368	0.8142	0.8741	0.5814	0.2781	-0.1853	0.3523	0.6727	0.2172	0.4889

Table 13: BLEU and COMET scores for each domain, the task is **MT with TMs** with $\mathrm{sim} \in [\mathbf{0.4}, \mathbf{0.6}]$. *All* is computed by concatenating all test sets (11k sentences). *Copy* refers to copying the TM match into the output.

Shorten the Long Tail for Rare Entity and Event Extraction

Pengfei Yu, Heng Ji
University of Illinois Urbana-Champaign `{pengfei4,hengji}@illinois.edu`

Abstract

The distribution of knowledge elements such as entity types and event types is long-tailed in natural language. Hence information extraction datasets naturally conform a long-tailed distribution. Although imbalanced datasets can teach the model about the useful real-world bias, deep learning models may learn features not generalizable to rare or unseen mentions of entities or events during evaluation, especially for rare types without sufficient training instances. Existing approaches for the long-tailed learning problem seek to manipulate the training data by re-balancing, augmentation or introducing extra prior knowledge. In comparison, we propose to handle the generalization challenge by making the evaluation instances closer to the frequent training cases. We design a new transformation module that transforms infrequent candidate mention representation during evaluation with the average mention representation in the training dataset. Experimental results on classic benchmarks on three entity or event extraction datasets demonstrate the effectiveness of our framework. [1]

1 Introduction

Long-tailed distributions are common in natural language processing tasks. This natural phenomenon of long-tailed distribution is formulated as Zipf's Law (Reed, 2001). For information extraction, knowledge elements such as entities, relations and events typically conform a long-tailed distribution in natural language. This leads to the imbalanced distribution of types in the benchmark datasets unless manual manipulation is performed to balance the dataset. We show the entity type distribution of an entity extraction dataset (Few-NERD (Ding et al., 2021)), and the event type distribution of two event extraction datasets (ACE 2005 (Walker et al., 2006) and MAVEN (Wang et al., 2020)) in Figure 1. All three datasets have similar long-tailed distributions. Among them, MAVEN and Few-NERD are two relatively large-scale datasets, but the training mentions are still concentrated on a small number of frequent types and majority of the entity/event types are rare types without sufficient training examples.

Long-tailed distribution is a generic problem not limited to NLP research. In fact, a lot of previous work on the long-tailed learning is in the computer vision domain (Lin et al., 2017; Kang et al., 2020). These approaches consider the imbalance in the type distribution as the major problem and are mostly based on balancing the datasets by upweighting the rare types and downweighting the frequent types. However, we argue that this line of research faces significant challenges when applying to the information extraction task. First, balancing the dataset breaks the real-world long-tailed prior in the datasets and may lead the model to mistakenly predict the long-tailed types too often. We observe this phenomenon in our experiments in Section 3 especially for the Classifier Re-training baseline (Kang et al., 2020). Second, due to the lack of effective data augmentation approaches as in the computer vision domain, upweighting the training mentions will only make the model to repetitively learn the same set of instances and aggravate the overfitting problem.

Instead of the imbalance in distribution, the real challenge of long-tailed learning for NLP is knowledge insufficiency, i.e. long-tailed datasets don't contain enough examples to acquire generalizable knowledge for the rare types. Learned models tend to capture features such as event trigger words for rare types and fail to generalize well to rare or even unseen cases (unseen trigger words) of those types during evaluation. For instance, if an event extraction model is trained on a dataset where all `Acquit` events are triggered by the keyword *acquit*, it may not identify the `Acquit` event trig-

---

[1] Code is available at `https://github.com/Perfec-Yu/LongTailIE`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1339–1350
May 2-6, 2023 ©2023 Association for Computational Linguistics

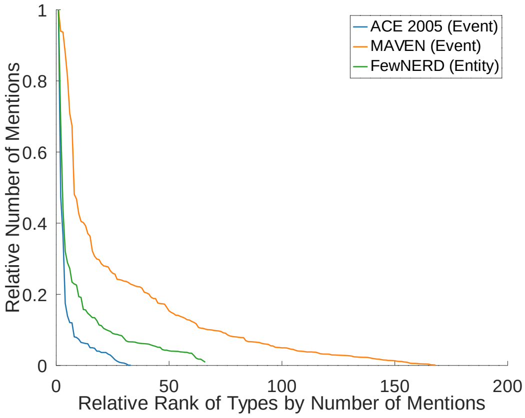


Figure 1: Distribution of types in ACE 2005 event dataset, MAVEN event dataset and the Few-NERD entity dataset. Y-axis is the number of training mentions divided by that of the most frequent type. X-axis is the rank of types by number of mentions.

gered by **walk free** in *The adjudicator allowed the criminal to* ***walk free***. Under this perspective, existing work tackling the long-tailed problem in information extraction (Han et al., 2018; Zhang et al., 2019) incorporates external structural knowledge to help learning rare types. However, the selection of the structural knowledge requires expertise in the target ontology and is not easily transferable to other ontologies. Another more recent line of work (Tang et al., 2020; Nan et al., 2021) considers the causal inference (Pearl et al., 2000) approach to solve the long-tailed problem. They aim to avoid the learning of spurious correlations between the input features and the labels in the limited training data for rare types. This is usually achieved by removing the effect of a manually selected confounding factor in prediction through deconfounded learning methods such as backdoor adjustment and inference based on total direct effect (Pearl et al., 2000). Although the model is guided to avoid spurious correlation through causal-inference-based approaches, learning the generalizable features with real causalities is still challenging due to the knowledge insufficiency problem.

In this work instead of confronting the knowledge insufficiency problem directly, we propose to bypass the problem by transforming the evaluation instances that require additional knowledge into more familiar instances that are closer to the frequent training instances. We decompose the input sentence $\{t_0, t_1, \ldots, t_n\}$ for the event/entity prediction task into two parts: a candidate token $e = t_i$ for the i-th token and surrounding contextual tokens $c = \{t_j\}_{j=0}^{i-1} \bigcup \{t_j\}_{j=i+1}^{n}$. Instead of making predictions for an event/entity type y_k solely based on $P(y_k|e, c)$, we propose to also consider predictions on averaged training inputs $P(y_k|c, \boldsymbol{r}_k)$, where $\boldsymbol{r}_k$ is the average training hidden representation of the type y_k. Our framework combines these two predictions using a transformation module. The transformation module computes weights g_e for the original candidate token e and g_k for each event type representation r_k. Moreover, instead of simply combining probabilities as $g_e P(y_k|e, c) + g_k P(y_k|c, \boldsymbol{r}_k)$, we use two weights g_e and g_k to combine in the representation layer for $\boldsymbol{e}$ and $\boldsymbol{r}_k$ to produce $P(y_k|g_e \boldsymbol{e} + g_k \boldsymbol{r}_k, c)$. The transformation module decides the weights g_e and g_k based on the frequency of the candidate mention in the training dataset and the cosine similarity between the candidate mention representation and all event types' average training mention representations in the ontology. Experimental results on three benchmark datasets demonstrate the effectiveness of our framework. Additionally, we found our approach, though derived from a different motivation, can be interpreted as a backdoor adjustment approach (Pearl et al., 2000) as shown in Section 2.5, which gives another interpretation that our approach improves long-tail learning by facilitating the model to learn the true correlation between the candidate text span and the event/entity type. We also empirically found that our approach facilitates the model to capture more generalizable features, which aligns with the goal of causal inference approaches.

To summarize, we propose a new approach for learning from the long-tailed datasets for entity extraction and event extraction. Our contributions are two-fold:

- We provide a new framework by bypassing the knowledge insufficiency problem in long-tailed learning and propose a novel learning framework in this perspective.
- We found that our framework aligns theoretically with the causal inference approach and can facilitate the model to capture more generalizable features.

2 Approach

2.1 Task Definition

In this work we take the task of both entity extraction and event extraction as sequence labeling

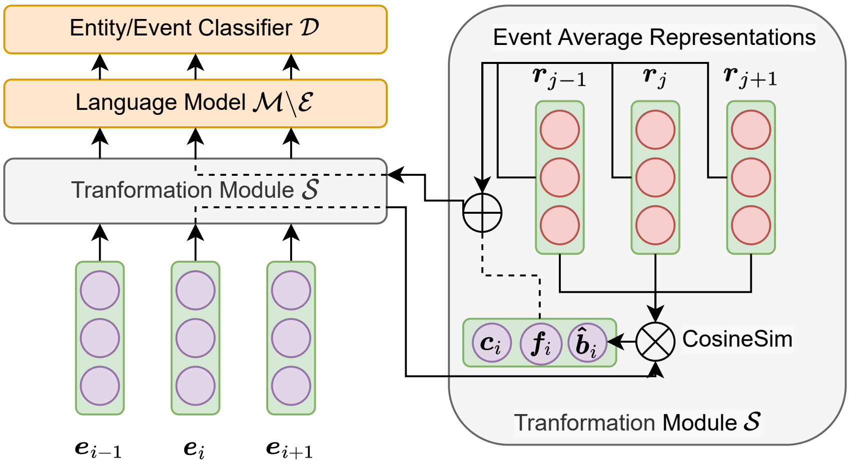


Figure 2: Overall long-tail learning information extraction framework. We ignore the original token t_i and only show the token embeddings $\boldsymbol{e}_i$ in this figure for conciseness. Here we use $\mathcal{M}\backslash\mathcal{E}$ to indicate the language model layers after the embedding layer and $\mathcal{D}$ as the classifier heads for the sequence labeling task. The transformation module $\mathcal{S}$ combines rare candidate mentions with average training mentions. The inputs to the transformation module includes the frequency feature $\boldsymbol{f}_i$, the similarity features $\hat{\boldsymbol{b}}_i$ between the token embedding $\boldsymbol{e}_i$ and the average training embeddings $\{\boldsymbol{r}_j\}_{j=1}^{K}$, and the context features $\boldsymbol{c}_i$.

problems. The input to the model $\mathcal{H}$ is a sequence of tokens $x = \{t_0, t_1, \ldots, t_n\}$, and the model predicts a label for each token $\mathcal{H}(x) = \{l_0, l_1, \ldots, l_n\}$. Each l_i is either one of the entity/event types or the `None` type for none-entity or none-event tokens.[2]

2.2 Overview of the Learning Framework

The learning of our proposed long-tailed extraction framework includes two steps. First, we finetune a language model with an additional classifier head for the sequence labeling problem without special treatments for the long-tail problem. In the second step, we train the transformation module while fixing the parameters of the language model. We adopt this two-stage training approach because the transformation module requires the language model's representations as inputs as we will show in Section 2.4. The finetuning step should render these representations as more task-specific features, which we redeem to be helpful for the learning of the transformation module. We will introduce these two steps in the following sections. The overall architecture is shown in Figure 2 and the two-staged training strategy is illustrated in Figure 3.

2.3 Finetuning Step

Given an input sequence $x = \{t_0, t_1, \ldots, t_n\}$, we first encode it with a pretrained language model $\mathcal{M}$ into $\mathcal{M}(x) = \{\boldsymbol{t}_0, \boldsymbol{t}_1, \ldots, \boldsymbol{t}_n\}$. We then adopt a linear discriminator $\mathcal{D}$ to predict the label for each token

$$l_i = \arg\max \mathcal{D}(\boldsymbol{t}_i). \qquad (1)$$

We finetune $\mathcal{M}$ and $\mathcal{D}$ with the cross entropy loss.

2.4 Learning the Transformation Module

The transformation module $\mathcal{S}$ transforms the representation for each token t_i into a weighted combination of its original representation and the average training representation of each event type. Our goal is to compute weights g_e, g_k for the final prediction $P(y_k | g_e \boldsymbol{e} + g_k \boldsymbol{r}_k, c)$ for each type y_k. However, this would require feeding the inputs multiple times to the language model, which is inefficient. Hence, we simplify the prediction to $P(y_k | g_e \boldsymbol{e} + \sum_{i=1}^{k} g_k \boldsymbol{r}_k, c)$. In other words, our transformation module combines the average representations of all types together in the hidden layer. We can essentially perform the transformation in any hidden layers of the pretrained language model $\mathcal{M}$. In this work, we perform transformation in the embedding layer to take full advantage of $\mathcal{M}$'s capability of encoding contextual information.

Specifically, let $\mathcal{E}(t_i) = \boldsymbol{e}_i$ be the embedding representation of the token t_i. Let $\{\boldsymbol{r}_j\}_{j=1}^{K}$ be the average training embedding representation for the entity/event type j, i.e.,

$$\boldsymbol{r}_j = \frac{1}{\#j} \sum_{\{t|l(t)=j\}} \mathcal{E}(t). \qquad (2)$$

---

[2] We adopt the IO labeling schema instead of BIO labeling schema to be consistent with the annotations in the Few-NERD dataset (Ding et al., 2021)

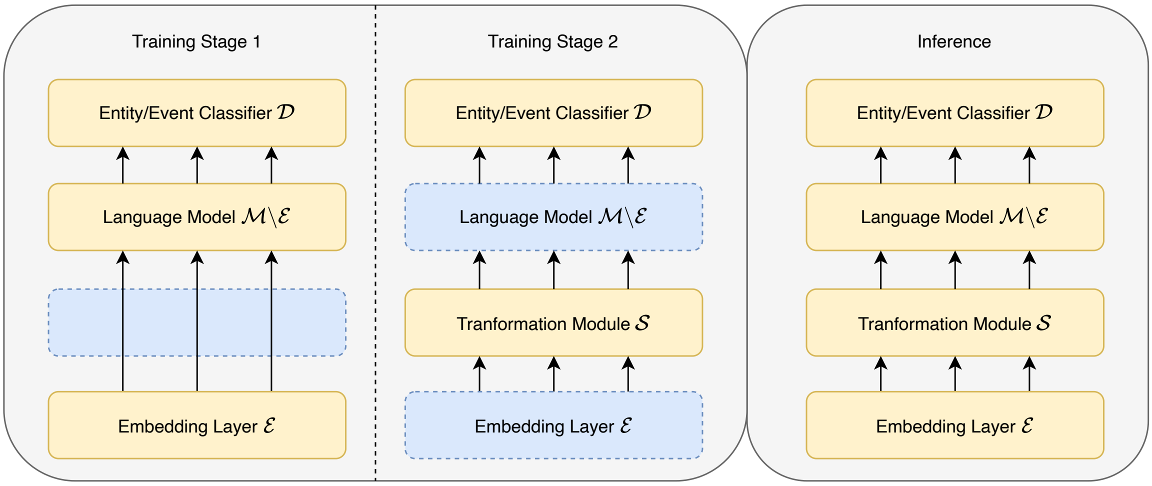


Figure 3: Illustration of the training and inference stages. There are two stages of the training on the left. Modules with dashed lines and blue background are not updated during the corresponding training stage.

Here $\{t|l(t) = j\}$ refers to the training tokens with the label j and $\#j$ is the total number of such tokens. The transformation module computes weights to combine each $\boldsymbol{e}_i$ with $\{\boldsymbol{r}_j\}_{j=1}^K$. We use the following information as input to compute the weights for combination

- t_i's frequencies of being labeled as entity/event types : $\boldsymbol{f}_i = [\exp(\gamma f_{ij})]_{j=1}^K$. Here we rescale the original frequencies inspired by (Lin et al., 2020). We use $\gamma = 0.1$ following (Lin et al., 2020). We expect the tranformation module to provide most help in rare cases when $\boldsymbol{f}_i$ is small.

- The similarity of t_i's embedding $\boldsymbol{e}_i$ with average training embeddings $\boldsymbol{r}_j$: $\boldsymbol{b}_i = [\cos(\boldsymbol{e}_i, \boldsymbol{r}_j)]_{j=1}^K$. If $\boldsymbol{b}_i$ contains large values, t_i is close to some event type's average embedding and potentially not a rare mention. On the other hand, if $\boldsymbol{b}_i$ is very small, t_i deviates significantly from some event type's average embedding and the transformation may significantly alter the meaning of the input. Taking these into consideration, we propose to model the similarity value $\boldsymbol{b}_i$ with a quadratic function, i.e. we use $\hat{\boldsymbol{b}}_i = [\boldsymbol{b}_i; \boldsymbol{b}_i^2]$ as the similarity features to the transformation module.

- The context representation $\boldsymbol{c}_i$, which is computed by the pretrained language model $\mathcal{M}$ while masking t_i in the input sequence x. Hence $\boldsymbol{c}_i$ is independent of the token t_i. This is to make sure that the transformation module is not overfitted only on the seen candidate tokens during training, because our goal is to apply the transformation module to rare cases in the evaluation corpus.

We decompose the computation of these weights into two steps. We first compute aggregation weights $\{\alpha_j\}_{j=1}^K$ for $\{\boldsymbol{r}_j\}_{j=1}^K$ with an attention module based on the context $\boldsymbol{c}_i$:

$$\begin{aligned} \alpha_j &= \frac{\exp(\langle \boldsymbol{a}_j, \boldsymbol{c}_i \rangle)}{\sum_{k=1}^K \exp(\langle \boldsymbol{a}_k, \boldsymbol{c}_i \rangle)} \\ \boldsymbol{s}_i &= \sum_j \alpha_j \boldsymbol{r}_j \end{aligned} . \tag{3}$$

Here $\{\boldsymbol{a}_j\}_{j=1}^K$ are trainable weights. We then feed all three features $[\boldsymbol{f}_i; \hat{\boldsymbol{b}}_i; \boldsymbol{c}_i]$ into a linear layer with the sigmoid activation to compute two gating scores $(g_e, g_s) \in [0, 1]^2$. The final embedding representation is computed as

$$\boldsymbol{h}_i = g_e \boldsymbol{e}_i + g_s \boldsymbol{s}_i. \tag{4}$$

We then substitute $\boldsymbol{e}_i$ with $\boldsymbol{h}_i$ for the following layers in $\mathcal{M}\backslash\mathcal{E}$.[3]

We train the transformation module $\mathcal{S}$ using the same cross entropy loss as the finetuning step. We also compute the cross entropy on the attention weights of the entity/event types in Equation (3) as an additional loss. We fix the parameters of the finetuned language model $\mathcal{M}$. This ensures that the similarity features $\hat{\boldsymbol{b}}_i$ and the context representations $\boldsymbol{c}_i$ remain fixed during the learning of $\mathcal{S}$. We provide training details in the Appendix.

---

[3] From Equation (4) we essentially have $g_k = g_s \alpha_k$

2.5 Connection to Causal Inference

We found our architecture can be mathematically intepreted as a backdoor adjustment (Pearl et al., 2000) method in causal inference, which is used to mitigate the effect of some confounding factor $\mathcal{U}$ in making decisions. In our case, it can be formulated as the following prediction probability

$$P(l_i|do(t_i)) = \sum_u P(l_i|t_i, \mathcal{U} = u)P(\mathcal{U} = u), \tag{5}$$

where u is the value space of $\mathcal{U}$. We refer readers to (Pearl et al., 2000) for the derivation of the above equation[4]. If we select the confounding factor $\mathcal{U}$ as the type prediction based on the context around the token t_i (independent of t_i), Equation (5) becomes

$$P(l_i|do(t_i)) = \sum_{j=1}^{K} P(l_i|t_i, j)P(j|\boldsymbol{c}_i),. \tag{6}$$

The attention weights $\{\alpha_j\}_{j=1}^{k}$ in Equation (3) is can be considered as probabilities of event types dependent on $\boldsymbol{c}_i$, thus can be seen as $P(j|\boldsymbol{c}_i)$ in the above equation. If we further model $P(l_i|t_i, j) = P(l_i|\boldsymbol{e}_i + \boldsymbol{r}_j)$ and apply the Normalized Weighted Geometric Mean (NWGM) approximation following (Yue et al., 2020),

$$P(l_i|do(t_i)) \approx P(l_i|\boldsymbol{e}_i + \sum_{j=1}^{k} \alpha_j \boldsymbol{r}_j). \tag{7}$$

In our framework, we are essentially computing

$$P(l_i|t_i) = P(l_i|g_e\boldsymbol{e}_i + g_s \sum_{j=1}^{k} \alpha_j \boldsymbol{r}_j). \tag{8}$$

This means that α_j composes the deconfounding priors $P(\mathcal{U})$ and (g_e, g_s) serve as the switch for the backdoor adjustment decisions. We expect the model to make decisions on rare or unseen mentions with the backdoor adjustment, and rely on the model's own prediction with more frequent cases.

3 Experiments

3.1 Datasets and Evaluation

We experiment on an entity extraction dataset, Few-NERD (Ding et al., 2021) and two event extraction datasets, ACE 2005 (Walker et al., 2006) and MAVEN (Wang et al., 2020). We provide data statistics in Table 1. Few-NERD, MAVEN and ACE include 67, 168, and 33 types respectively. For ACE 2005, we split the dataset such that all event types are covered in the evaluation data which is different from the splits used in previous work. For Few-NERD, we used the "supervised" split for experiments.

For evaluation, we use the macro F1 score as the main metric to reflect the influence of long-tail types. We also report micro F1 for additional reference. We provide macro recall, macro precision, micro recall and micro precision in Appendix. Previous work also reports scores on subsets of rare types (Nan et al., 2021). Instead of manually determine a frequency threshold for the "rarest" types, we plot curves of F1 scores on all types in Figure 4. All results are averaged over three runs using different random seeds.

3.2 Methods in Comparison

We group previous work on long-tailed learning into two categories: balancing-based approaches and causal inference approaches. For balancing-based approaches, we mainly follow (Kang et al., 2020) and implemented Classifier Re-training (CRT), Nearest Class Mean classifier (NCM), τ-normalized classifier (τ-norm) and Learnable weight scaling (LWS). We also implemented the focal loss approach (Lin et al., 2017). For causal inference work, we implemented Momentum (Tang et al., 2020) and CFIE (Nan et al., 2021). Although (Nan et al., 2021) also experimented on the ACE 2005 and MAVEN, we re-implemented[5] their methods due to the difference in splits and evaluation strategies. CFIE also has different designs for the entity extraction and event extraction models. We didn't re-implement the entity model due to insufficient details. We also compare with a **Vanilla** baseline which is the sequence labeling model without the transformation module.

3.3 Main Results

We show our main results in Table 2 and Figure 4. From Table 2, our framework achieves the best macro F1 scores across three datasets. We also noticed that our approach improves macro F1 scores without suffering from inferior micro F1 scores. Al-

---

[4] Note that the common formulation for $P(l_i|t_i)$ would be substituting $P(\mathcal{U} = u)$ with $P(\mathcal{U} = u|t_i)$

[5] Due to insufficient details in their paper and released code, we implemented their framework to the best of our knowledge. CFIE also requires named entity extraction results as part of the inputs to event extraction. Therefore it is not clear how the entity extraction model is designed.

Dataset	Few-NERD		MAVEN		ACE 2005	
	#Sentences	# Mentions	#Sentences	# Mentions	#Sentences	# Mentions
train	131,767	340,387	32, 431	77,993	16,807	4,254
dev	18,824	48,770	8,042	18,904	2056	500
test	37,648	96,902	9,400	21,835	1,930	570

Table 1: Dataset statistics.

though the Momentum baseline performs closely to our approach in ACE Macro F1. However, Momentum has a much worse micro F1 sacrificing frequent types and worse performance on the other two datasets. These results also show the superiority of our approach and indicate that our model's performance on frequent types is not harmed. This is because we do not tackle the long-tailed learning problem in terms of balancing that may affect the natural distribution in the dataset. Instead, we transform the long-tailed evaluation samples into frequent training samples with the transformation module.

Although macro F1 metric puts extra stress on the long-tailed types, the performance on the rare types is still dilated by frequent types especially for datasets with a large number of types such as MAVEN. In order to further investigate the performance on different subsets of event types, we show in Figure 4 macro F1 improvements (over the `Vanilla` baseline) for the top X fraction of the event types with the fewest training examples. We notice that our approach, together with other approaches, indeed improves the learning with long-tailed types when X is small. We notice that our approach can significantly outperform the baseline for the less frequent types with a maximum of over 10% on ACE 2005 event dataset. We found our approach has the best performance of the moderately long-tailed event types compared with other long-tailed learning approaches. For extremely long-tailed types, our performance is also close to the best method `Momentum` (Tang et al., 2020). One possible reason of our model's ineffectiveness on extremely long-tailed types is that we cannot learn reliable representation $\boldsymbol{e}_i$ to compute similarity values $\hat{\boldsymbol{b}}_i$ for the transformation module. Another possible reason is that the extreme long-tailed types don't have enough training instances to provide an informative average training representation r_k. A potential solution would be introducing external knowledge (e.g., a few keywords as candidate event triggers or entity mentions) to enrich the r_k. Moreover, our approach is more consistent than other methods when X grows larger. This further validates that our approach improves the long-tailed types without sacrificing the performance of frequent event types. For more references, we also provide top 10 event types in MAVEN dataset where our method achieves most F1 performance gain on in Appendix. Majority of those types are rare types.

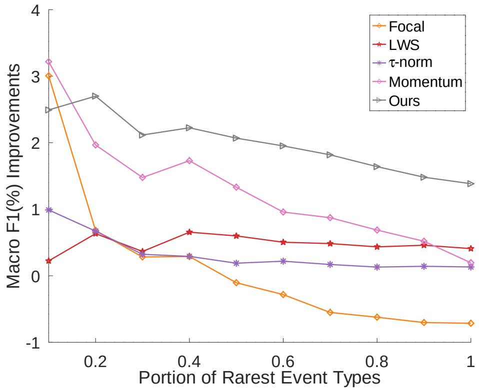
3.4 Improving Classifier Features

In our second training stage, we finetune the entity/event classifier head together with the transformation module. Finetuned classifiers from our approach should benefit from the connections of our approach and the backdoor adjustment approach for causal inference in Section 2.5, which encourages the model to capture generalizable features instead of surface correlations. To test the finetuned classifiers alone, we evaluate our framework with the transformation module disabled, i.e. forcing $(g_e, g_s) = (1, 0)$ in Equation (4). The forward architecture becomes exactly the same as the `Vanilla` model. In Table 3, we observe that finetuned classifier alone outperforms the `Vanilla` baseline. Since we fixed the language model in the second stage, the improvements purely come from the finetuned classifiers that have learned to avoid surface features for prediction.

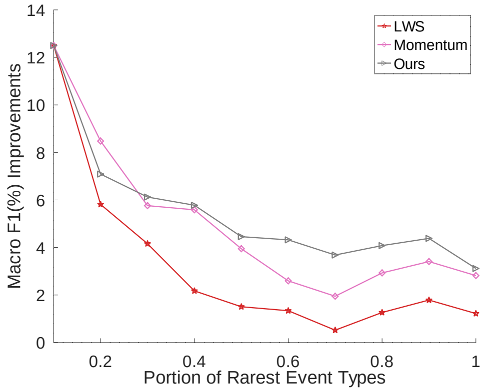
In addition to the interpretation of the improvements from the causal inference theory, we give a more intuitive explanation based on Equation (4). During the second training step, Equation (4) can be seen as changing the original candidate representation with an interpolation of itself and other candidates' representations of the same event type. This is similar to augmenting the dataset by interchanging mention spans of the same type across sentences, though we interchange it with the average representation of that type in the entire training dataset. The classifier may benefit from this "aug-

Dataset	Few-NERD		MAVEN		ACE 2005	
	Macro F1	Micro F1	Macro F1	Micro F1	Macro F1	Micro F1
Vanilla	62.4	**67.8**	60.1	67.1	60.4	72.7
Focal	62.5	**67.8**	59.4	65.3	59.9	**73.9**
CRT	44.9	54.3	51.2	54.3	60.2	71.8
LWS	61.4	66.9	60.5	66.7	60.9	72.5
τ-norm	62.4	**67.8**	60.2	67.1	60.2	72.7
NCM	51.5	50.4	57.5	61.2	57.8	72.2
CFIE	-	-	56.3	61.2	50.6	63.2
Momentum	59.7	63.9	60.1	66.4	62.1	68.9
Ours	**62.6**	**67.8**	**61.2**	**67.4**	**62.4**	72.9

Table 2: Macro and micro F1 scores (in %) on three datasets. References to the methods above is provided in Section 3.2.



(a) MAVEN



(b) ACE 2005

Figure 4: Performance difference compared with `Vanilla` w.r.t the portion of rarest event types. We first rank event types by the number of training mentions from low to high, and then compute average F1 scores (minus that of the `Vanilla` baseline) for the first X fraction of the event types. We omitted those approaches that are significantly lower than `Vanilla`.

mentation” to capture more general features.

Method	Ours	No Sub	Vanilla
Macro F1	62.4	61.9	60.4

Table 3: Performance (in %) with the transformation module disabled (No Sub) on the ACE 2005 dataset.

3.5 Case Study

We also show some examples that the `Vanilla` baseline misses but our approach correctly identifies and classifies the candidate mentions in Table 4. We also visualize the transformation weights (g_e, g_s) in Equation (4). As discussed in Section 3.3, our model is the best at handling moderately long-tailed types. For these two examples, the event `Start-Position` in the first case has 94 mentions and `became` appears only 2 times as the trigger. In the second case, the `Action` appears 709 times, but only 16 of them are triggered by `undertaken`[6]. It is worth mentioning that although g_s is not as large as g_e in both cases, we found that they are indispensable since the model fails on both cases if we disable the transformation module by forcing $g_s = 0$.

4 Related Work

4.1 Balancing-based Long-tailed Learning for Computer Vision

Long-tailed learning is closely related to imbalanced learning. In the computer vision domain, Lin et al. (2017) proposes the focal loss to handle

---

[6] As a comparison, `war` appears 270 times as the trigger of `Attack` in ACE 2005 and `storm` appears 757 times as the trigger of `Catastrophe` in MAVEN.

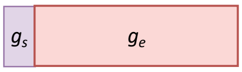
Input Sentence	Dataset	Transformation Weights
This upcoming visit to Russia will be my first trip aboard since I **became** president of China. (Event: `Start-Position`)	ACE	
The operation was **undertaken** so that Allies could secure a beachhead. (Event: `Action`)	MAVEN	

Table 4: Examples of missing event triggers by the `Vanilla` model. Transformation weights (g_e, g_s) in Equation (4) are visualized.

the imbalanced learning problem in object detection. The focal loss downweights the loss terms for confident training samples so that the model predicts high probabilities for the gold standard labels. In addition to this, some work (Zhang et al., 2019; Rebuffi et al., 2017), though not focusing on the long-tailed learning problem, adopts special training strategies to tackle the imbalance in the datasets. Kang et al. (2020) summarize these approaches and evaluate them with the computer vision tasks. These approaches are mostly based on balancing the datasets by upweighting the rare types and downweighting the frequent types, either by modifying the sampling strategy or associating weights to the loss terms. These methods first learn the feature extractor with the original imbalanced distribution and then perform special treatments to the classification layer. Classifier Re-training (CRT) retrains the classification layer by sampling examples from each type uniformly. Nearest Class Mean classifier (NCM) uses the average training features as the type-level features and uses certain distance metrics to perform the nearest neighbor classifcation. τ-normalized classifier (τ-norm) normalizes the type weights in the classification layer. This can make sure the weights for frequent types are not significantly larger than rare types. Learnable weight scaling (LWS) is similar to τ-norm but learns the normalization weights while sampling examples from each type uniformly.

4.2 Long-tailed Learning with External Knowledge for IE

Balancing-based methods are usually ineffective in information extraction as shown in our experiments. Existing work usually tackles the long-tailed problem in information extraction (Han et al., 2018; Zhang et al., 2019) by incorporating external structural knowledge. Han et al. (2018) adopt hierarchical structures among relation types to transfer knowledge from the frequent relation types to their siblings. Zhang et al. (2019) incorporate label semantics and knowledge graph embeddings to transfer knowledge from frequent types to rare types. Required expertise in selecting external knowledge limits the generalization of these methods. (Yu et al., 2021) also leverages correlation between event types to help the learning of rare event types in the context of lifelong learning.

4.3 Causal Inference and Its Application to Long-tailed Learning

Due to the potential of causal inference (Pearl et al., 2000) theory to reduce the spurious correlation, there have been explorations on its application in machine learning (Lopez-Paz et al., 2017; Magliacane et al., 2018; de Haan et al., 2019; Bengio et al., 2020; Yang et al., 2020; Li et al., 2021; Park et al., 2021). Since the the spurious correlation is more common in limited data, some work attempts to interpret the long-tailed learning problem under a causal inference framework. The core component of this interpretation is to find a confounding factor that affects the distribution of the input features and output labels at the same time. Tang et al. (2020) consider the training momentum of the gradients to be the confounding factor and proposed the corresponding deconfounded training with the backdoor adjustment approach. They use the total direct effect (TDE) for inference, which essentially lowers the probabilities of frequent types systematically. Nan et al. (2021) work on the information extraction tasks and take a set of linguistic features as the confounding factor. They also adopt an inference approach similar to TDE to lower the probabilities of frequent types.

4.4 Related Few-shot Learning Methods

Few-shot learning aims at training models with a small number of instances, which is similar to the goal of improving rare types in long-tailed learning. Some few-shot learning methods also overlap with long-tailed learning methods. Snell et al. (2017) proposed a prototypical network that has a similar framework as NCM for long-tailed learning. Yue et al. (2020) proposed a backdoor adjustment approach based on the causal inference thoery to reduce the spurious correlation in the model, since the few-shot learning also has the limited data size problem. However, few-shot learning usually works on the N-way K-shot setting. This makes these approaches usually not directly applicable to the long-tailed learning scenario.

5 Conclusions and Future Work

In this work we propose a new long-tailed learning framework for entity and event extraction by candidate transformation. We design a novel transformation module to convert representations of rare or unseen mentions during evaluation into representations of average training mentions. Experimental results have validated the effectiveness of our framework. Our framework can significantly improve the performance on long-tailed types, and outperform other long-tailed learning methods especially for moderately long-tailed types. Moreover, our framework does not sacrifice the performance on frequent types. We also discover the connections between our learning framework and the backdoor adjustment in the causal inference theory We empirically observe that our training strategy can improve the model's capability in capturing more generalizable features, which aligns with the causal inference theory. In the future, we will explore: (1) adapt the concept of transformation module to other NLP tasks; (2) based on our connection with the causal inference theory, it is possible to design a better transformation module by choosing a better confounding factor other than the context information.

6 Limitations

In terms of the framework design, our current design of the framework is only applicable to the sequence labeling task, although we believe it can be adapted to other NLP tasks without significant modifications. Besides, our framework should be most helpful if the semantics of the elements in the sequence of the same type are close to each other and thus may require additional modifications in more heterogeneous cases, such as vision-language models where visual features may have the same type as the textual token embeddings.

In terms of time efficiency, our framework will require an additional training stage to learn the transformation module, which will cost extra time. Since we fixed the language model, which composes majority of parameters in our framework, this additional cost is acceptable. We also recommend to pre-process input features (since they are fixed with respect to training samples during the second stage) to the transformation module to further reduce the time cost in the second stage.

7 Acknowledgement

This research is based upon work supported in part by U.S. DARPA LORELEI Program No.HR0011-15-C-0115, U.S. DARPA AIDA Program No. FA8750-18-2-0014 and KAIROS Program No. FA8750-19-2-1004. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Nan Rosemary Ke, Sébastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher J. Pal. 2020. A meta-transfer objective for learning to disentangle causal mechanisms. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Pim de Haan, Dinesh Jayaraman, and Sergey Levine. 2019. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11693–11704.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of* *the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2245, Brussels, Belgium. Association for Computational Linguistics.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2020. Decoupling representation and classifier for long-tailed recognition. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xin Li, Zhizheng Zhang, Guoqiang Wei, Cuiling Lan, Wenjun Zeng, Xin Jin, and Zhibo Chen. 2021. Confounder identification-free causal visual feature learning. *CoRR*, abs/2111.13420.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2999–3007. IEEE Computer Society.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Schölkopf, and Léon Bottou. 2017. Discovering causal signals in images. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 58–66. IEEE Computer Society.

Sara Magliacane, Thijs van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M. Mooij. 2018. Domain adaptation by using causal inference to predict invariant conditional distributions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10869–10879.

Guoshun Nan, Jiaqi Zeng, Rui Qiao, Zhijiang Guo, and Wei Lu. 2021. Uncovering main causalities for long-tailed information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9683–9695, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jongjin Park, Younggyo Seo, Chang Liu, Li Zhao, Tao Qin, Jinwoo Shin, and Tie-Yan Liu. 2021. Object-aware regularization for addressing causal confusion in imitation learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 3029–3042.

Judea Pearl et al. 2000. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19:2.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5533–5542. IEEE Computer Society.

William J Reed. 2001. The pareto, zipf and other power laws. *Economics letters*, 74(1):15–19.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. 2020. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 multilingual training corpus LDC2006T06. Web Download. Philadelphia: Linguistic Data Consortium.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online. Association for Computational Linguistics.

Xu Yang, Hanwang Zhang, and Jianfei Cai. 2020. Deconfounded image captioning: A causal retrospect. *CoRR*, abs/2003.03923.

Pengfei Yu, Heng Ji, and Prem Natarajan. 2021. Lifelong event detection with knowledge transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5278–5290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhongqi Yue, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. 2020. Interventional few-shot learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3016–3025, Minneapolis, Minnesota. Association for Computational Linguistics.

A Appendix

A.1 Details on Two-stage Training

In the first training stage, all parameters in the language model $\mathcal{M}$and entity/event classifier head $\mathcal{D}$ are updated. In the second stage, we fix the entire language model and train both $\mathcal{D}$ and the transformation module $\mathcal{S}$. Apart from the cross entropy loss for entity/event extraction, we also apply loss to the weights $\{\alpha_j\}_{j=1}^{K}$ in Equation (3). This loss is only applicable to entity/event mention tokens, since $\{\alpha_j\}_{j=1}^{K}$ corresponds to entity/event types. For these mention tokens, we compute cross entropy between attention weights and the labeled types.

For the input features to the transformation module, we want to make the values of all three kinds of features to be in the similar range. Therefore, we adopt a batch normalization module to these features. We also add dropout with probability 0.2 to similarity features and frequency features, and dropout with probability 0.5 to the context features.

A.2 Implementation Details

For all experiments, we use learning rate to be $1e-5$ and batch size to be 8 to fit in a single Nvidia Tesla V100 GPU with 16GB memory. We evaluate performance after each epoch and select the best model based on the development performance. We use early-stop strategy with a patience of 5 epochs. We report average performance over 3 runs initialized with 3 different random seeds.

The approximate number of parameters is 3.5 million (RoBERTa). Added parameters from the transformation module is significantly less and dependant on the number of target entity/event types. Approximately the transformation module has 2,500 parameters. For the first stage training, it takes about 10-20 minutes to train an epoch on ACE 2005 data, 20-30 minutes on MAVEN data and about 40 minutes on Few-NERD. The time difference mainly comes from the total number of sentences in these datasets. However these are just rough estimations since the performance largely depends on the environmental factors such as temperatures and also the workload of other gpus/cpus in the same machine during training. In the second stage, we didn't preprocess any features in advance but block the backpropagation to language model parameters. In general it is 4-5 times faster than the first stage. We incorporate the transformation module by modifying the RoBERTa implementation from Transformers[7] Library.

A.3 Dataset Licenses

This dataset is licensed by LDC.[8] Membership is required for access. The dataset can be used for research purpose. Few-NERD dataset is distributed under the CC BY-SA 4.0 license. MAVEN dataset is ditributed under MIT License.

A.4 Additional Results

We include precision and recall scores in Table 5. Moreover, we include top 10 event types in the MAVEN dataset that our approach achieves most improvements over the Vanilla baseline, as well as their number of training mentions in Table.

---

[7] `https://huggingface.co/docs/transformers/index`

[8] `https://www.ldc.upenn.edu`

Dataset	Few-NERD		MAVEN		ACE 2005	
	Macro P,R	Micro P,R	Macro P,R	Micro P,R	Macro P,R	Micro P,R
Vanilla	60.1, 65.3	65.8, 70.0	61.9, 61.1	65.7, 68.6	63.6, 62.0	73.4, 72.1
Focal	60.6, 64.9	66.0, 69.8	57.8, 63.1	62.0, 69.1	61.9, 63.0	73.8, 74.1
CRT	40.7, 51.1	50.6, 58.8	41.1, 75.1	41.2, 79.8	58.6, 67.2	66.7, 77.9
LWS	58.1, 65.7	64.3, 69.7	60.1, 63.9	62.4, 71.8	62.5, 64.7	72.0. 73.0
τ-norm	60.8, 63.8	65.9, 69.9	62.2, 61.1	64.9, 69.4	63.5, 61.8	73.4, 72.0
NCM	56.0, 65.6	61.0, 43.3	58.9, 63.6	57.3, 65.7	64.2,56.5	76.1, 68.7
CFIE	-	-	50.9, 66.5	52.5, 73.5	59.7, 56.9	58.0, 69.7
Momentum	60.3, 65.4	59.7, 62.5	61.8, 61.9	66.3, 66.5	60.9, 68.3	66.9, 69.5
Ours	60.3, 65.4	65.8, 70.0	61.3,63.2	65.5, 69.4	63.5, 65.5	71.4, 74.6

Table 5: Macro and micro precision and recall scores (in %) on three datasets.

Event Type	# of Mentions	F1 (%) Imp.
Kidnapping	87	19.0
Body_movement	115	15.8
Emptying	124	15.1
Manufacturing	326	13.8
Scouring	32	13.2
Carry_goods	48	12.0
Military_operation	1,022	10.5
Practice	37	10.4
Labeling	35	10.0
Cure	71	8.4

Table 6: Top 10 event types in the MAVEN dataset that our approach achieves most improvements over the Vanilla baseline. We also include the number of training mentions and F1 score improvements (in %) in the second the third columns. MAVEN has a total of 168 event types. and our training split includes 77,987 training event mentions in total for all event types.

Do Deep Neural Networks Capture Compositionality in Arithmetic Reasoning?

Keito Kudo[1] **Yoichi Aoki**[1] **Tatsuki Kuribayashi**[1,2] **Ana Brassard**[3,1]
Masashi Yoshikawa[1] **Keisuke Sakaguchi**[1,3] **Kentaro Inui**[1,3]
[1]Tohoku University [2]Langsmith, Inc. [3]RIKEN
`{keito.kudo.q4, youichi.aoki.p2}@dc.tohoku.ac.jp, kuribayashi@tohoku.ac.jp,`
`ana.brassard@riken.jp, {yoshikawa,keisuke.sakaguchi,inui}@tohoku.ac.jp`

Abstract

Compositionality is a pivotal property of symbolic reasoning. However, how well recent neural models capture compositionality remains underexplored in the symbolic reasoning tasks. This study empirically addresses this question by systematically examining recently published pre-trained seq2seq models with a carefully controlled dataset of multi-hop arithmetic symbolic reasoning. We introduce a *skill tree* on compositionality in arithmetic symbolic reasoning that defines the hierarchical levels of complexity along with three compositionality dimensions: systematicity, productivity, and substitutivity. Our experiments revealed that among the three types of composition, the models struggled most with systematicity, performing poorly even with relatively simple compositions. That difficulty was not resolved even after training the models with intermediate reasoning steps.[1]

1 Introduction

Integrating symbolic reasoning capabilities into neural models has been a crucial goal of artificial intelligence (Marcus, 2003; d'Avila Garcez and Lamb, 2020). With this in mind, many researchers investigated how well modern neural models achieve symbolic reasoning (Lake and Baroni, 2018). However, recent studies have reported conflicting results on this; some suggest that neural models can solve complex multi-hop reasoning (Clark et al., 2020), while others claim that models struggle even with performing simple symbolic operations (Qian et al., 2022).

As a step toward further understanding neural models' symbolic reasoning ability, this study systematically analyzes recently published pre-trained seq2seq models using a carefully controlled dataset of multi-hop arithmetic symbolic reasoning.

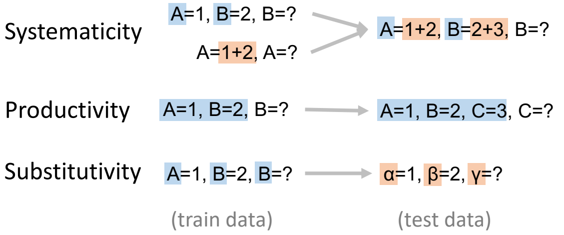


Figure 1: Three dimensionalities of compositionality in arithmetic symbolic reasoning

Specifically, our study empirically evaluates the models' ability to generalize the *compositionality* underlying arithmetic reasoning, where we explore three dimensions of compositionality: (i) systematicity, (ii) productivity, and (iii) substitutivity, as illustrated in Figure 1. Capturing compositionality is crucial in performing symbolic reasoning since compositionality is a pivotal property of generalizability over training instances.

To systematically explore the models' composition ability, we introduce a *skill tree on compositionality* that defined the hierarchical levels of complexity in arithmetic symbolic reasoning, as illustrated in Figure 2. Using this hierarchy as a lens, we identify the limitations of the neural seq2seq models in capturing the compositionality in arithmetic symbolic reasoning. Our major findings can be summarized as follows:

- Among the three types of composition, the models struggled most with **systematicity**, performing poorly even with relatively simple compositions.
- The major difficulty in systematicity was in the access to intermediate information that is not stated in input but produced during the reasoning.
- Capturing systematicity remained hard for the models trained with the information of the intermediate reasoning steps.

---

[1] Our code and data are available at `https://github.com/keitokudo/dentaku_skill_tree`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1351–1362
May 2-6, 2023 ©2023 Association for Computational Linguistics

2 Skill tree in arithmetic reasoning

We take arithmetic reasoning as the domain for our exploration because it allows us to synthesize questions systematically, as we show in this paper, which helps examine a model's composition ability in a controlled manner. Furthermore, the arithmetic reasoning ability of neural models has gained much attention as modern large language models still struggle with this problems (Rae et al., 2021).

Specifically, we use multi-hop arithmetic reasoning problems as follows:

Question: `A=1, B=2, C=A+2, C=?`
Answer: `3`

Here, the value assigned to the variable `C` is asked.

2.1 Compositionality in multi-hop symbolic reasoning

In this study, we specifically focused on three dimensions: systematicity, productivity, and substitutivity (Hupkes et al., 2020). According to these, we evaluate how well neural models achieve compositional generalization.

Systematicity refers to the ability of combining known different concepts into a more complex concept, i.e., structural composition. To evaluate this ability in models, we first trained with several types of primitive operations (e.g., addition; `A=1+2,A=?` and selection; `A=1,B=2,B=?`). Then, we measured the performance in solving problems consisting of combinations of primitives (e.g., `A=1+2,B=2+3,B=?`).

Productivity refers to the ability to solve longer/complex problems based on shorter/simpler ones. To evaluate this ability in models, we first trained with a short version of a formula (e.g., `A=1+2,B=2+3,B=?`). Then, we measured the performance in solving longer problems (e.g., `A=1+2,B=2+3,C=3+4,C=?`).

Substitutivity refers to the ability to keep the performance even if a particular constituent in a problem is replaced with another (unseen) constituent (i.e., lexical composition). To evaluate this ability in models, we conduct several experiments changing the variable characters between training and test (e.g., train with `A=1+2,A=?`; then evaluated with `α=1+2,α=?`).

2.2 Dataset configurations

Typical symbolic reasoning (e.g., procedural programming, assembly language) consists of at least three primitive symbol manipulations: assignment (`a=2`), arithmetic operation (`1+2`), and reference (`a=?`). With this in mind, our dataset is generated by combining the following five basic formulas: (i) `A=1` (assignment), (ii) `A=B` (reference & assignment), (iii) `A=1+2` (arithmetic operation & assignment), (iv) `A=B+2` (arithmetic operation & assignment & reference), (v) `A=?` (reference). The detailed properties are explained in Section 3.

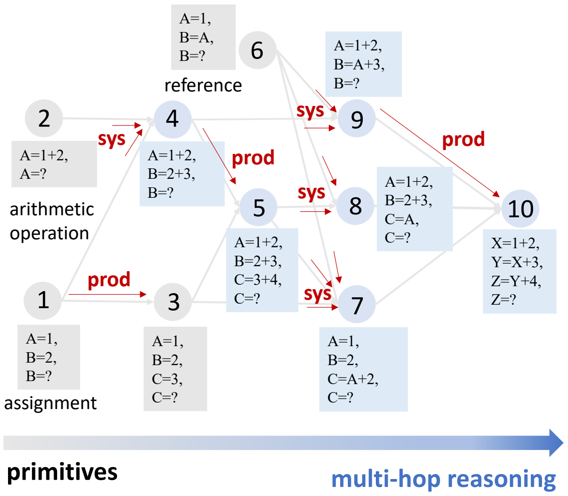


Figure 2: Skill tree to evaluate compositional generalization. The data format of primitive operations is gray and others (complex formulas composed of combinations of primitive operations) are blue .

2.3 Skill tree evaluations

We preliminarily observed that compositionally generalizing complex multi-hop arithmetic reasoning was difficult for neural seq2seq learners (the 1,2,6→9 setting in Section 4). Building on this fact, this study questions what type of composition made it hard for the neural models. To answer this, we designed a *skill tree on compositionality* that organizes the (hierarchical) complexity levels of symbolic reasoning.[2] Evaluating the models using problems with different complexity of composition in a step-wise manner, we elucidate the exact weakness of neural seq2seq models in multi-hop symbolic reasoning.

Specifically, we designed ten versions of symbolic reasoning problems. The hierarchical relationship of their task levels is illustrated in Figure 2; in this skill tree, each vertex, i.e., domain, corresponds to different task settings with different

---

[2] The term "skill tree" refers to a visualization method of step-by-step learning in the field of pedagogy (Tondello and Nacke, 2019), distinct from the "tree" in graph theory.

		base		large		x-large	
Task	Type	ZA	WA	ZA	WA	ZA	WA
1,2 →4	sys.	42.0	82.1	35.2	89.8	51.7	96.7
	+subst.	39.1	80.4	33.2	89.4	50.7	96.7
2,3 →5	sys.	33.6	75.4	32.1	85.6	35.9	94.7
	+subst.	33.6	77.0	31.5	87.2	36.5	94.9
2,3,6 →8	sys.	40.8	76.5	39.1	87.7	40.5	94.6
	+subst.	39.3	74.7	37.5	86.4	39.2	94.9
2,3,6 →7	sys.	56.5	79.7	51.0	83.7	58.1	94.2
	+subst.	57.6	80.2	51.1	85.8	56.7	95.1
1,2,6 →9	sys.	24.1	28.2	23.1	29.3	27.5	32.6
	+subst.	25.8	28.0	24.1	31.7	28.5	34.7
7,8 →10	sys.	23.6	21.3	25.3	25.9	22.3	28.2
	+subst.	22.3	21.6	24.4	26.4	23.0	30.2
1 →3	prod.	100.0	100.0	100.0	100.0	100.0	100.0
	+subst.	100.0	100.0	100.0	100.0	100.0	100.0
4 →5	prod.	100.0	100.0	100.0	100.0	100.0	100.0
	+subst.	99.9	99.9	100.0	100.0	100.0	99.9
9 →10	prod.	57.0	59.3	61.7	63.8	60.6	62.7
	+subst.	58.4	60.9	62.2	64.1	59.5	64.5

Table 1: Average accuracies in the experiment with 2 different seeds. The "Task" column exhibits (train→test) domains corresponding to the skill-tree (Figure 2). The "Type" column shows the targeted compositionality type in each setting; here, "sys.," "prod," and "subst." denote the systematicity, productivity, and substitutivity generalizations, respectively.

complexity, and edges represent the hierarchical complexity levels.

By adequately selecting a particular combination of training and test domains, we evaluated the compositional generalization ability of the models from various perspectives. Here, the arithmetic expressions used in the test domain are a combination of those in training domains, creating a semi-order relationship in the skill tree. For example, using the settings 1 (`A=1+2,A=?`) and 2 (`A=1,B=2,B=?`) as a training set, and 4 (`A=1+2,B=2+3,B=?`) as a test set, one can evaluate the model's systematicity generalization towards the arithmetic operations (`a+b`) and assignments (`A=`i`,B=`j`,B=?`).

3 Experimental settings

3.1 Data

Dataset: In each experimental setting, we refer to the training domains as $\mathcal{D}_{\text{train}} = \{d_{\text{train}1}, \cdots d_{\text{train}k}\}$ and the test domain as d_{test}. Each domain has 100,000 training data and 3,200 test data; these are randomly generated, and there is no overlapped instance. When the training domain consisted of multiple domains, we used the union of the training data in $\mathcal{D}_{\text{train}}$. In addition, when the training domain is not primitive operations (1, 2, 3, and 6 in Figure 2), we further added the primitive operation data related to the training domain (Appendix A) into the training data.

Arithmetic expressions: As introduced in Section 2, the input is a sequence of arithmetic expressions. Formally, each expression is in the format of `a=`n or `a=`n`{+,-,max,min}`m except that the final expression asks the number assigned to a specified variable (`b=?`). Here, `a` and `b` are a member of a variable name set Σ; n and m are a member of the variable name set or number set $\Sigma \cup \mathcal{N}$. Specifically, Σ consists of 21 alphabets, and N consists of the integer from 0 to 99. The symbol = indicates that the result of the left-hand side is substituted into the right-hand side. The operations (`+`, `-`, `max`, and `min`) correspond to arithmetic addition, subtraction, max (returning the larger of its left and right numbers), and min (returning the smaller of its left and right numbers).

The questions are designed so that the answer is unique, and depending on the problem set-up, may include mathematical expressions that are not directly related to the final answer, i.e., distractors. The order of the equations is arbitrary; the first equation should not necessarily be calculated first.

Substitivity test: In each experimental setting, we evaluate the substitutivity generalization performance of the model under the situation where the variable names are replaced with unseen ones, e.g., training with `a=1+2,a=?`; then evaluating with α`=2+4,`α`=?`. In this setting, we replaced each variable name in the test set with one of five alphabets that do not overlap with the training ones.

3.2 Trainig and test

Training: The training stops when the accuracy on the validation dataset does not increase in successive five epochs or until the validation accuracy reaches 100%. Checkpoints with the highest accuracy in the validation dataset are used for evaluation. Note that among the experiments, the accuracy in the training domain reached at least 99.5%; this indicates that the primitive operations were learnable for the models. Detailed settings for training are described in Appendix A.

Evaluation metrics: The accuracy is calculated by the test data in the test domain d_{test}. Here, we used two metrics: (i) zero-shot accuracy (ZA) and (ii) weighted average of accuracies (WA) to measure the efficiency of learning (Talmor et al., 2020). In measuring WA, a model was further trained using the training set in the test domain d_{test}; then, the weighted average of accuracies at every update was calculated (details are in Appendix B).

3.3 Models:

We used three different sizes (base, large, and xl) of T5 (Raffel et al., 2020), which is a widely used pre-trained seq2seq model in numerical reasoning tasks (Pal and Baral, 2021; Chung et al., 2022; Yang et al., 2021). Note that we began our training using the models with learned parameters. We also evaluated BART variants and randomly initialized models in Appendix C.

4 Experiments and results

We adopted nine combinations of training and test domains as shown in the first column of Table 1 (training domains→test domain). Six of them test the systematicity generalization and the other three test productivity generalization. In each setting, we further tested substitutivity generalization ability using the test domain data with a different variable name set (e.g., α instead of `A`) to that used in the training domain.

Table 1 shows the overall results. We observed the following four trends:

- Systematicity generalization was more difficult than productivity generalization.
- Even in the simple composition (the setting 1,2→4), the models struggle with generalization from zero or few examples.
- Models achieved substitutivity generalization.
- Model size did not incur substantial performance difference.

We identified that the systematicity generalization of reference and arithmetic operations (setting $2,3 \rightarrow 5$; from `A=1,B=2,C=3,C=?` and `A=1+2,A=?` to `A=1+2,B=2+3,C=4+5,C=?`) was a simple setting, yet difficult to solve (refer to Appendix D for results on other tasks.). To better understand why neural models struggle with this setting, we decomposed the complexity of this setting and analyzed the model performance. Note that Kim and Linzen (2020) also suggested that neural models lack systematicity generalization ability in the context of semantic parsing; our results corroborate their findings from the context of arithmetic multi-hop reasoning.

	base		large		x-large	
Setting	ZA	WA	ZA	WA	ZA	WA
2,3→5	33.6	75.4	32.1	85.6	35.9	94.7
String	37.3	94.1	66.1	98.4	86.9	99.3
Steps	26.2	82.1	36.1	89.4	33.7	96.4

Table 2: Ablation study with the 2,3→5 (vanilla) setting. “String” refers to the setting where string operations are used instead of arithmetic operations. “Step” denotes the setting generating intermediate steps.

Is this difficulty specific to *arithmetic* symbolic reasoning? We experimented with the same setting except that the four arithmetic operations are replaced with string operations (`join`, `reserveJoin`, `strSub`, and `stackJoin`; details are in Appendix E.1). The notable difference between arithmetic and string operations is that the string operation could be achieved by only copying selective elements in the input (e.g., `12+34=1234`), while arithmetic operation requires the models to access the arithmetic knowledge stored in their internals (e.g., `1+2=3`) and generate new information not stated in the input context (e.g., 3).

Larger models tended to overcome the weakness in composition with string operations (e.g., the accuracy of 86.9 in zero-shot evaluation with the x-large model), while they struggled with arithmetic operations. This suggests that the major difficulty in systematicity was in **the access to the arithmetic knowledge (e.g., 1+1=2)**.

Does scratchpad training alleviate the difficulty? Existing studies suggested that showing the intermediate step (scratchpad-style training/inference) improves the multi-hop reasoning ability of neural models (Wei et al., 2022). We tested whether such an explicit generation of intermediate information alleviates the difficulty faced in the previous analysis. Specifically, we trained models with intermediate steps (e,g. `A=1+2,B=2+3,B=?; B=2+3,B=5`. Details are in Appendix E.2) during training.

The accuracy was calculated by the exact match of the answer and intermediate steps (the steps are designed to be uniquely determined). The performance gain due to explicating the intermediate steps was limited (Table 2), at least with our T5-based models. This shows that, in our carefully controlled setting, merely employing the scratchpad-style generation is not substantially effective.

5 Analysis

We conduct a more in-depth analysis of compositional generalization difficulties from another

Complexity dimensions	base ZA	base WA	large ZA	large WA	x-large ZA	x-large WA	Average
Δ#variables	0.098	-0.098	0.488	-0.293	-0.098	-0.488	-0.065
Δ#numbers	0.059	0.265	-0.088	0.647	0.206	0.677	0.294
Δ#operations	-0.507	-0.338	-0.338	0.169	-0.338	0.169	-0.197
Δ#reference	-0.655	-0.655	-0.393	-0.655	-0.655	-0.655	**-0.611**

Table 3: Spearman's rank correlation coefficient between the increase of training–test arithmetic complexity and the compositional generalization performance (accuracy) across the nine settings listed in Table 1. A negative score indicates that the greater the training–test discrepancy in its dimension, the more difficult compositional generalization is. In the case that there are multiple training domains, the maximum value among them is used.

perspective—complexity of arithmetic expressions. Specifically, for each pair of training and test domains listed in Table 1 (e.g., 1,2→4), we quantified the increase of the complexity of arithmetic formulas from several aspects, e.g., how much the formula's number of variables increased in the test domain (setting of 4) compared to the training domain (setting of 1 and 2). Specifically, we focused on the increase of the number of variables (Δ#variables), numbers (Δ#numbers), operations (Δ#operations), and references (Δ#references) from the training to test domains. Here, "#reference" denotes the number of access to a particular variable on the right-hand of equations. For example, the Δ#references is 1 if the training data format is `A=`n`,B=`**`A`**`+`m and the test format is `A=`n`,B=`**`A`**`+`m`,C=`**`B`**`+`l. Then, we identified which dimension strongly relates to the compositional generalization difficulty.

We analyzed the macro trends between formula's complexity increase and the difficulty of generalization across the experimental settings. Table 3 shows Spearman's rank correlation coefficient between each complexity and the test-domain accuracy. We found a notable negative correlation in the Δ#reference; that is, the more references in the test domain compared to the training domain, the more difficult the compositional generalization becomes (the cases of 1,2,6→9 and 9→10 settings). Simply put, this reveals the difficulty of compositional generalization with *multi-hop* reasoning—retaining the results of a calculation and accessing them again for another calculation.

6 Related work

The analysis of the compositional generalization ability of neural models and arithmetic multi-hop reasoning problems have typically been studied separately; this study has merged these two directions. As for composition generalization analysis, several studies analyzed neural models using datasets such as SCAN (Lake and Baroni, 2018), COGS (Kim and Linzen, 2020), and CFQ (Keysers et al., 2020). These mainly focused on compositionality in the context of semantic parsing; the composition ability toward symbol manipulations (e.g., multi-hop arithmetic reasoning) is typically out of focus. As for arithmetic reasoning, neural models' abilities have been analyzed typically using benchmarks such as DROP (Dua et al., 2019). It has recently been reported that such dataset has superficial cues (Al-Negheimish et al., 2021), which made it unclear how much arithmetic reasoning neural model achieves; our study using a carefully controlled dataset contributed to the exact weakness of neural models in this context.

7 Conclusion

In this study, we have empirically investigated the arithmetic multi-hop reasoning ability of modern neural models through the lens of compositional generalization ability. To systematically analyze neural models' ability, we have defined a skill tree that organizes the (hierarchical) complexity levels of the multi-hop symbolic reasoning dataset.

Our experiments have revealed that the major weakness lies in systematicity, even with a relatively simple composition. Through the ablation studies, we also have found that difficulty in systematicity is pronounced in accessing knowledge that is not written in input but stored in models. Furthermore, even in training models with intermediate steps that explicate the composition, they struggle to capture systematicity. We also found the difficulty of multi-hop reasoning in compositional generalization. These highlight the exact weakness of neural models and encourage studies to overcome such limitations.

Limitations

In this work, we explored neural networks' ability to capture compositionality in symbolic arithmetic reasoning in hopes that it may lead to future improvements in more general reasoning. However, arithmetic reasoning may not necessarily generalize to natural language tasks. Furthermore, we explored several aspects of multi-hop arithmetic reasoning, but these were chosen from a relatively human-centric perspective, and models may suffer from unforeseen other difficulties. Finally, while we found several patterns in how model performance degrades, it is difficult to aggregate this into a full picture of what a model can and cannot do. Further experiments are needed to gain a more complete understanding of model performance

Acknowledgements

We thank four anonymous reviewers who provided valuable feedback. We would like to also appreciate the member of Tohoku NLP Group for their cooperation in conducting this research. This work was supported by JST CREST JPMJCR20D2 and JSPS KAKENHI Grant Number JP22H00524, 21K21343.

References

Hadeel Al-Negheimish, Pranava Madhyastha, and Alessandra Russo. 2021. Numerical reasoning in machine reading comprehension tasks: are we there yet? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9643–9649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3882–3890. International Joint Conferences on Artificial Intelligence Organization. Main track.

Artur S. d'Avila Garcez and Luís C. Lamb. 2020. Neurosymbolic AI: the 3rd wave. *CoRR*, abs/2012.05876.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? (extended abstract). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization. Journal track.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, Online. Association for Computational Linguistics.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Gary F Marcus. 2003. *The algebraic mind: Integrating connectionism and cognitive science*. MIT press.

Kuntal Kumar Pal and Chitta Baral. 2021. Investigating numeracy learning ability of a text-to-text transfer model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3095–3101, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jing Qian, Hong Wang, Zekun Li, Shiyang Li, and Xifeng Yan. 2022. Limitations of language models in arithmetic and symbolic induction. *CoRR*, abs/2208.05051.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew J. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4603–4611. PMLR.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Gustavo F. Tondello and Lennart E. Nacke. 2019. A pilot study of a digital skill tree in gameful education. In *Proceedings of the 3rd International Symposium on Gamification and Games for Learning - GamiLearn '19*. CEUR-WS.org, CEUR-WS.org.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Peng-Jian Yang, Ying-Ting Chen, Yuechan Chen, and Daniel Cer. 2021. Nt5?! training T5 to perform numerical reasoning. *CoRR*, abs/2104.07307.

A Training configurations

Training data: Data of primitive domains (grey domains in Figure 2) are also added to the training data. Specifically, data in the primitive domains that are reached by traversing the graph (Figure 2) from the training domain to the left is added. For example, when the training domain has the domain of 7, the data of primitive domains of 2, 3, and 6 are added[3]. Note that when the domain of 3 is included, the domain of 1 is not added. Additionally, as the objective of this study does not emphasize the generalization performance in arithmetic ability, the scope of numbers utilized in the test domain is adjusted to ensure that the upper limit of the answers in the test domain does not surpass that of the answers in the train domain.

Hyperparameter: We used the T5 models (v1.1) as pre-trained model[4]. We used three model sizes: base (250 million parameters), large (800 million parameters), and xl (3 billion parameters). Following T5 (Raffel et al., 2020) fine-tuning configurations, we use Adafactor (Shazeer and Stern, 2018) as the optimizer with a constant learning rate. Also, we specify the learning rate 1.0×10^{-5} for training and 5.0×10^{-5} in measuring the WA. The batch size is 32 for all experimental settings. We trained each model on NVIDIA A6000 (48GB memory), A100 (80GB memory). In addition, following previous research about numerical reasoning (Geva et al., 2020), we tokenize numbers in a digit-by-digit manner.

		base		large	
Task	Type	ZA	WA	ZA	WA
1,2	sys.	25.5	71.2	33.8	61.0
→4	+subst.	25.2	72.4	30.9	62.0
2,3	sys.	23.0	48.7	25.4	49.1
→5	+subst.	22.1	47.5	25.7	43.5
2,3,6	sys.	29.6	57.6	35.0	55.1
→8	+subst.	28.7	50.4	34.3	50.2
2,3,6	sys.	36.7	59.8	40.4	48.5
→7	+subst.	39.4	57.0	38.6	39.3
1,2,6	sys.	22.0	26.1	21.1	16.1
→9	+subst.	22.6	26.7	23.5	18.2
7,8	sys.	24.1	28.6	32.5	25.3
→10	+subst.	24.9	27.7	34.9	29.3
1	prod.	92.4	99.8	100.0	83.1
→3	+subst.	91.2	99.8	100.0	82.3
4	prod.	60.4	93.0	89.5	90.8
→5	+subst.	63.8	95.9	91.8	87.4
9	prod.	33.5	49.4	41.6	21.0
→10	+subst.	34.3	48.2	43.5	22.2

Table 4: Experimental results when BART is used as a pre-trained model. The "Task" column exhibits (train→test) domains corresponding to the skill-tree (Figure 2). The "Type" column shows the targeted compositionality type in each setting; here, "sys.," "prod," and "subst." denote the systematicity, productivity, and substitutivity generalizations, respectively.

B How to calculate WA

As described in Section 3, we used weighted average accuracy (WA). This metric quantifies the efficiency (ease) of generalization by assigning a high weight to accuracy in the early training stage. Specifically, the weights w_i (where i is the number of validation steps) were calculated using the following formulae:

$$w_i = -ai + w_{\max} \quad , \tag{1}$$

$$w_{\max} = \alpha w_{\min} \quad , \tag{2}$$

$$w_{\min} = \frac{2}{(N+1)(\alpha+1)} \quad , \tag{3}$$

$$a = \frac{(w_{\max} - w_{\min})}{N} \quad . \tag{4}$$

Here, N is the number of validation steps. α is a hyperparameter that determines how heavily the accuracy in the early stages is weighted. We set α to 1000 in all the experiments. Also, We used the first 100 validation steps to calculate WA, so specify $N = 100$ for all experiments. WA was calculated with accuracy on the held-out validation datasets in the test domain.

---

[3] Task 7,8→10 do not incorporate data from the primitive domains into the training data to evaluate if compositional generalization can be achieved solely from complex components.

[4] `https://huggingface.co/docs/transformers/model_doc/t5v1.1`

Task	base	large	x-large
1,2 (→4)	72.5	100.0	100.0
2,3 (→5)	63.3	56.1	99.9
1,3,6 (→7, 8)	68.6	67.9	67.5
1,2,6 (→9)	99.9	76.0	76.8
7,8 (→10)	48.6	37.3	24.0
1 (→3)	51.1	15.1	16.3
4 (→5)	100.0	99.9	99.9
9 (→10)	99.7	99.9	99.7

Table 5: Accuracy on the training held-out dataset when we start training the model from randomly initialized parameters. (When training began with parameters pretrained on language, the accuracy was almost 100% in all settings.) Each training task includes the primitive operations required to solve train domain tasks as described in section A.

C Model variants

C.1 BART

To confirm the generality of our results obtained with the T5 models, we also conduct the same experiment using BART (Lewis et al., 2020). We use two model sizes[5]: base (140 million parameters) and large (400 million parameters). Table 4 shows the result using BART. The same tendency described in Section 4 was observed when BART was used as a language-pre-training model.

C.2 Training from scratch

We also experimented with the case where we started training from randomly initialized parameters to isolate the effect of the pre-training adopted in T5. We found that these initialized models failed to learn even primitive operations and in-domain tasks (Table 5), at least with the hyperparameter setting[6] used in this study. Thus, we did not proceed to their evaluation of compositionality generalization.

D Detailed analysis of T5-based models

Figure 3 shows the learning curves (accuracy on validation datasets) for all tasks demonstrated in Table 1. The graphs also show that the neural language model struggles to solve tasks that require compositional generalization in the beginning part of the training.

D.1 String operations ablation for all tasks

Table 6 shows the experimental results of the ablation study using string operations for all tasks.

D.2 Scratch pad evaluation for all tasks

Table 7 shows the experimental results of the ablation study using scratchpad for all tasks.

E Dataset details

Table 8 summarizes the characteristics of each task.

E.1 String operations

Table 9 shows the details of the string operations.

E.2 Scratch pad formulation

For the additional study using scratchpad in section 4, we generate problems with ing the intermediate steps. Scratchpad is in the form of straightforward one-by-one calculations of only those calculations necessary to find the target variable. The following is an example of scratchpad reasoning:

Question: `A=1+2, B=A+3, B=?`
Answer: `A=1+2;A=3;B=A+3;B=3+3;B=6`

---

[5] https://huggingface.co/facebook/bart-base, https://huggingface.co/facebook/bart-large

[6] Unlike in the training domain used for the language pretrained model, we relax the training stopping criterion. We stopped training when the accuracy did not increase by **10** epochs in the validation dataset.

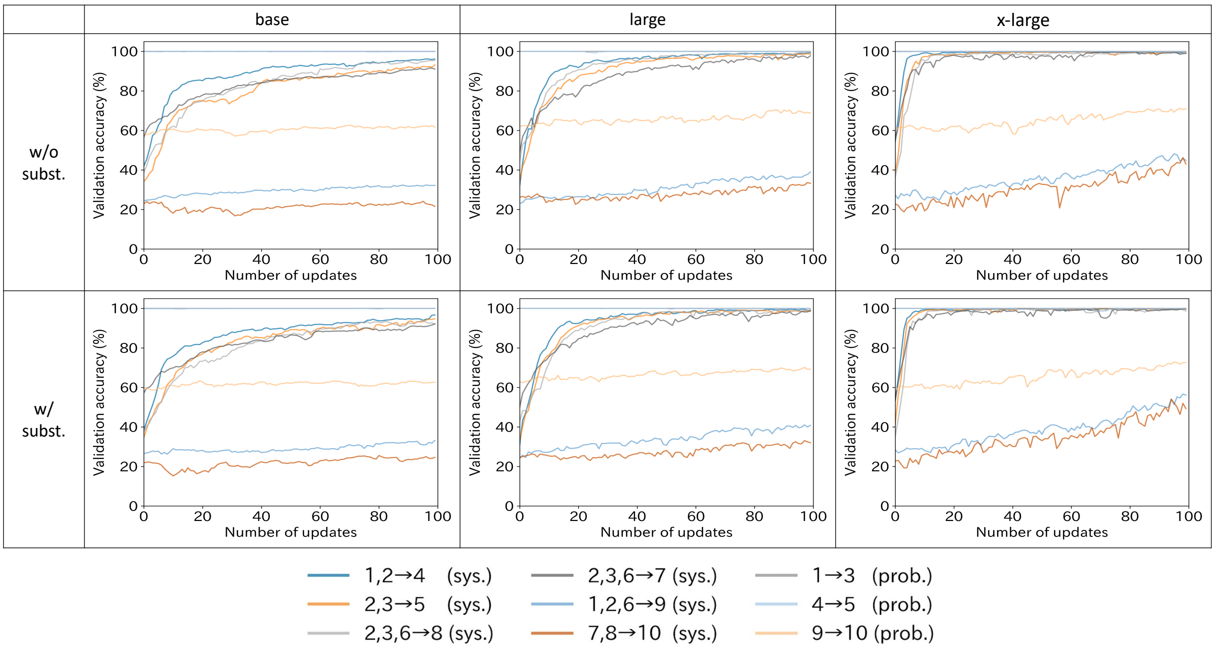


Figure 3: Learning curves of generalization test for all tasks.

Task	Type	base ZA	base WS	large ZA	large WS	x-large ZA	x-large WS
1,2	sys.	59.2	95.9	66.9	98.5	63.9	98.9
→ 4	+subst.	59.0	95.1	67.3	98.2	62.2	99.0
2, 3	sys.	37.3	94.1	66.1	98.4	86.9	99.3
→ 5	+subst.	35.9	93.5	65.2	98.6	84.2	99.0
2,3,6	sys.	59.6	93.9	71.5	98.0	52.5	97.2
→8	+subst.	56.3	92.8	67.1	98.0	52.3	97.0
2,3,6	sys.	34.3	91.0	46.3	93.6	32.0	95.7
→7	+subst.	34.3	91.3	48.1	93.6	34.2	96.5
1,2,6	sys.	13.2	34.3	11.6	38.1	12.3	59.6
→9	+subst.	12.7	34.9	13.0	39.0	12.4	58.8
7,8	sys.	26.2	61.1	20.0	61.7	21.3	67.4
→10	+subst.	26.4	61.4	19.9	60.0	20.5	68.5
1	prod.	100.0	100.0	100.0	100.0	100.0	100.0
→3	+subst.	100.0	100.0	100.0	100.0	99.9	100.0
4	prod.	100.0	100.0	99.9	100.0	100.0	100.0
→5	+subst.	100.0	100.0	100.0	100.0	100.0	100.0
9	prod.	69.6	88.2	71.5	89.9	75.3	92.0
→10	+subst.	67.4	88.8	72.0	89.6	75.2	92.0

Table 6: Experiments result from string operation ablation. The "Task" column exhibits (train→test) domains corresponding to the skill-tree (Figure 2). The "Type" column shows the targeted compositionality type in each setting; here, "sys.," "prod," and "subst." denote the systematicity, productivity, and substitutivity generalizations, respectively.

		base		large		x-large	
Task	Type	ZA	WS	ZA	WS	ZA	WS
1,2	sys.	30.6	81.3	33.9	85.5	35.6	94.9
→4	+subst.	30.3	77.9	32.9	85.5	36.3	95.5
2,3	sys.	26.2	82.1	36.1	89.4	33.7	96.4
→5	+subst.	25.4	81.3	36.1	88.1	32.9	96.4
2,3,6	sys.	0.0	27.6	0.0	47.5	0.0	71.3
→8	+subst.	0.0	26.0	0.0	47.3	0.0	72.9
2,3,6	sys.	0.0	43.1	0.0	59.0	0.0	75.0
→7	+subst.	0.0	40.0	0.0	56.6	0.0	73.3
1,2,6	sys.	0.0	0.2	0.0	2.5	0.0	14.3
→9	+subst.	0.0	0.1	0.0	2.7	0.0	16.4
7,8	sys.	0.0	17.0	0.0	33.2	0.0	55.1
→10	+subst.	0.0	17.4	0.0	32.6	0.0	55.3
1	prod.	100.0	100.0	100.0	100.0	96.0	99.9
→3	+subst.	100.0	100.0	100.0	100.0	95.8	99.9
4	prod.	100.0	100.0	100.0	99.9	100.0	99.9
→5	+subst.	100.0	100.0	100.0	99.9	100.0	99.9
9	prod.	0.0	50.8	0.0	72.2	0.0	80.9
→10	+subst.	0.0	51.8	0.0	69.5	0.0	79.3

Table 7: Experiments result from scratchpad ablation. The “Task” column exhibits (train→test) domains corresponding to the skill-tree (Figure 2). The “Type” column shows the targeted compositionality type in each setting; here, “sys.,” “prod,” and “subst.” denote the systematicity, productivity, and substitutivity generalizations, respectively.

Domains		Required premitives			Targeted composition		
Task	Example	Assign.	Arith.	Ref.	Sys.	Prod.	Subst.
1	`A=1, B=2, B=?`	✓					
2	`A=1+2, A=?`	✓	✓				
3	`A=1, B=2, C=3, C=?`	✓				✓	
3’	`α=1, β=2, γ=3, γ=?`	✓				✓	✓
4	`A=1+2, B=2+3, B=?`	✓	✓		✓		
4’	`α=1+2, β=2+3, β=?`	✓	✓		✓		✓
5	`A=1+2, B=2+3, C=3+4, C=?`	✓	✓		✓	✓	
5’	`α=1+2, β=2+3, γ=3+4, γ=?`	✓	✓		✓	✓	✓
6	`A=1, B=A, B=?`	✓		✓			
7	`A=1, B=2, C=B+3, C=?`	✓	✓	✓	✓		
7’	`α=1, β=2, γ=β+3, γ=?`	✓	✓	✓	✓		✓
8	`A=1+2, B=2+3, C=B, C=?`	✓	✓	✓	✓		
8’	`α=1+2, β=2+3, γ=β, γ=?`	✓	✓	✓	✓		✓
9	`A=1+2, B=A+3, B=?`	✓	✓	✓	✓		
9’	`α=1+2, β=α+3, β=?`	✓	✓	✓	✓		✓
10	`A=1+2, B=A+3, C=B+4, C=?`	✓	✓	✓	✓	✓	
10’	`α=1+2, β=α+3, γ=β+4, γ=?`	✓	✓	✓	✓	✓	✓

Table 8: Dataset configurations. The “Task” column exhibits (train→test) domains corresponding to the skill-tree (Figure 2). “Assign.” “Arith.” and “Ref.” in “Required premitives” column mean primitive operations name, “assignment” “arithmetic operations” and “reference” (see subsection 2.2) The “Target composition” column shows the targeted compositionality type in each setting; here, “Sys.,” “Prod,” and “Subst.” denote the systematicity, productivity, and substitutivity generalizations, respectively.

	Description	Example
`join`	String concatenation.	`12 + 34 = 1234`
`reverseJoin`	String concatenation + Reverse	`123 ^ 78 = 87321`
`strSub`	Deletion of duplicate characters. Return 0 if there are no duplicates.	`7873 - 73 = 87`
`stackJoin`	Select one character from the left side of each string alternately and return the combined string.	`12 * 34 = 1324`

Table 9: Detail of string operation.

BLM-AgrF: A New French Benchmark to Investigate Generalization of Agreement in Neural Networks

Aixiu An[1]*, **Chunyang Jiang**[2], **Maria A. Rodriguez**[3]*, **Vivi Nastase**[2] and **Paola Merlo**[2]
[1]Department of Brain and Cognitive Sciences
Massachusetts Institute of Technology, `aixiu@mit.edu`
[2]Department of Linguistics
University of Geneva
`{Chunyang.Jiang, Paola.Merlo}@unige.ch, vivi.a.nastase@gmail.com`
[3] Computer Science and Information Technology
Lucerne University of Applied Sciences and Arts, `maria.anduezarodriguez@hslu.ch`

Abstract

Successful machine learning systems currently rely on massive amounts of data, which are very effective in hiding some of the shallowness of the learned models. To help train models with more complex and compositional skills, we need challenging data, on which a system is successful only if it detects structure and regularities, that will allow it to generalize. In this paper, we describe a French dataset (BLM-AgrF) for learning the underlying rules of subject-verb agreement in sentences, developed in the BLM framework, a new task inspired by visual IQ tests known as Raven's Progressive Matrices. In this task, an instance consists of sequences of sentences with specific attributes. To predict the correct answer as the next element of the sequence, a model must correctly detect the generative model used to produce the dataset. We provide details and share a dataset built following this methodology. Two exploratory baselines based on commonly used architectures show that despite the simplicity of the phenomenon, it is a complex problem for deep learning systems.

1 Introduction

Over the last years, driven by the surge in deep learning methods, models in NLP have become very powerful. They have even reached superhuman performance on standard benchmarks such as SuperGLUE (Wang et al., 2019) and SQuAD (Rajpurkar et al., 2018). Deeper probing, though, shows that this is due to the models' surprisingly robust superficial natural language understanding ability. This indicates that new benchmarks are necessary that accurately show the level of progress in language processing (Ruder, 2021).

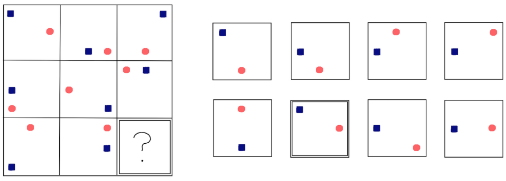


Figure 1: An example Raven's progressive matrix (best seen in colour). The matrix is constructed according to two rules: (i) the red dot moves one place clockwise when traversing the matrix left to right; (ii) the blue square moves one place anticlockwise when traversing the matrix top to bottom. The task consists in finding the tile in the answer set that correctly completes the sequence, indicated with a double border.

In this paper, we describe such a benchmark, developed based on a new method described in (Merlo et al., 2022), and summarized in Section 2. The method defines a procedure for building datasets that capture specific linguistic phenomena in a structured problem, inspired by the visual patterns detection tasks in Raven's progressive matrices (RPM) (Raven, 1938), as in Figure 1.

The visual RPMs manipulate elements with attributes such as position, shape, colour and size. The language matrices manipulate phrases, dependencies in the syntactic tree, and lexical, grammatical and semantic attributes between connected elements of a sentence. To successfully tackle such a complex problem, a system must detect structure and patterns in the data. To complement the structure of the problem, the candidate answers set is

---

[*] The work was done while the author was at the University of Geneva.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1363–1374

also specifically designed – the negative answers are built following specific criteria – to help determine which facets of the problem the system is able to learn, and which it is not.

By enabling different levels of analysis, from the solution of the task in different controlled setting to the analysis of the errors, this dataset intends to support the development of neural models with stronger abilities of abstraction and generalization, and more complex and compositional skills, that could learn robust models from few examples, and ultimately be deployed on low-resource languages. The code and the data are available here: `https://github.com/CLCL-Geneva/BLM-SNFDisentangling`.

2 BLM-AgrF: Blackbird's Language Matrices for agreement

CONTEXTS TEMPLATE				
1	NP-sing	PP1-sing		VP-sing
2	NP-plur	PP1-sing		VP-plur
3	NP-sing	PP1-plur		VP-sing
4	NP-plur	PP1-plur		VP-plur
5	NP-sing	PP1-sing	PP2-sing	VP-sing
6	NP-plur	PP1-sing	PP2-sing	VP-plur
7	NP-sing	PP1-plur	PP2-sing	VP-sing
8	NP-plur	PP1-plur	PP2-sing	VP-plur

ANSWER SET		
1	NP-sing PP1-sing et NP2 VP-sing	Coord
2	**NP-plur PP1-plur NP2-sing VP-plur**	correct
3	NP-sing PP-sing VP-sing	WNA
4	NP-sing PP1-sing PP2-sing VP-plur	AE
5	NP-plur PP1-sing PP1-sing VP-plur	WN1
6	NP-plur PP1-plur PP2-plur VP-plur	WN2

Figure 2: BLM instances for verb-subject agreement, with two attractors. WNA= wrong number of attractors; AE= agreement error; WN1= wrong nr. for 1^{st} attractor noun (N1); WN2= wrong nr. for 2^{nd} attractor noun (N2).

The data format we present has been called Blackbird's Language Matrices (BLMs) (Merlo et al., 2022), because it requires the presentation of the linguistic phenomenon of interest in the form of a complex set of sentences that have both syntagmatic and paradigmatic relations, thereby, like in the RPM visual version, forming a matrix structure.

A BLM has a structure defined by a combination of rules. The starting point is defining the linguistic problem that needs to be learned (e.g. subject-verb agreement) and the grammatical rules that define it. The combination of rules can be complex and each rule can act as an interfering factor obfuscating the other rules. The next step is to devise the rules governing the abstract automatic generation process that embody the properties of the linguistic phenomenon and its underlying rules. Combining these examples of grammatical rules will produce templates that can be used to automatically create large samples of data with lexical/structural variety. To allow for probing the learned model, apart from the correct answer, the answer sets contain negative examples built by corrupting some of the generating rules. This helps investigate the kind of information and structure learned, and the type of mistakes a system is prone to.

2.1 BLM-AgrF for subject-verb agreement

The BLM-AgrF dataset we illustrate here defines implicitly the rules of subject-verb agreement in French. As a reminder, the main rule of subject-verb agreement in French, and English, states that subjects and verbs agree in their number. Agreement is a rule that applies to the structure of the sentence and not the linear order, so agreement applies independently of how many noun phrases intervene between the subject and the verb.

Subject-verb agreement is a morphological phenomenon of appropriate complexity to start our investigations with BLMs. Subject-verb agreement is clearly limited to some specific words in the sentence so that the elements and the attributes manipulated by the underlying rules can be clearly identified. It is marked explicitly in the forms of words (for example by an $-s$ ending) and it does not depend on the words' meaning. Moreover, agreement rules show structural properties, so that sequences of increasing complexity of application of the rule can be defined (Linzen et al., 2016; Linzen and Leonard, 2018). We choose to work specifically on French because its agreement system, its verb conjugations and its noun phrase structure lend themselves well to our investigation.

The data that illustrates this linguistic rule must show all patterns of combination of agreement between subject and verb but also include data that illustrate the structural nature of the rule. Noun phrases inserted in the subject NP as prepositional complements or relative clauses act as intervening elements. We consider one and two such noun phrases, to increase the distance between the subject and the verb, and different clause complexities to produce data that covers syntactic structures of various depths.

While intervening noun phrases do not enter into an agreement relation based on the grammatical rules, in practice, the intervening noun phrases can act as agreement attractors and trigger agreement mistakes, if they are close to the verb. More specifically, Franck et al. (2002) show, in experiments with French and English speakers, that attraction is determined by the syntactic distance between an intervening noun and the head noun.

BLM-AgrF grammatical templates To generate the BLMs for the subject-verb number agreement, we develop a context-free grammar based on the targeted linguistic phenomenon and the interfering factors chosen, illustrated in Figure 3.

```
<CONSTRUCTION>→ <AGREEMENT>

# structure
<AGREEMENT>→ <MAIN-CLAUSE>
<AGREEMENT>→ <COMPLETIVE-CLAUSE>
<AGREEMENT>→ <RELATIVE-CLAUSE>

# L' ordinateur avec le programme est en panne .
<MAIN-CLAUSE>→
          <SUBJNP(Num)><ATTRACTORS><VP(Num)>

# L' ordinateur avec le programme dont Jean se servait est
en panne .
<RELATIVE-CLAUSE>→
          <SUBJNP(Num)><ATTRACTORS>
               <RELCLAUSE><VP(Num)>

# Jean suppose que l' ordinateur avec le programme est
en panne .
# Jean suppose que l' ordinateur avec le programme de l'
expérience est en panne .
<COMPLETIVE-CLAUSE>→
          <COMPCLAUSE><MAIN-CLAUSE>

# e.g.: "dont Jean se servait"
<RELCLAUSE>→ {Rel chunks}
# e.g.: "Jean suppose que"
<COMPCLAUSE>→ {Comp chunks}

# e.g. ["L' ordinateur", "Les ordinateurs"]
<SUBJNP(Num)>→ {NP chunks (Num)}

# e.g. PP1: ["avec le programme", "avec les programmes"]
# e.g. PP2: ["de l' expérience", "des expériences"]
<ATTRACTORS>→ <PP>
<ATTRACTORS>→ <PP><ATTRACTORS>
<PP>→ {Prep chunks (attractor)}

# e.g.: ["est en panne", "sont en panne"],
<VP(Num)>→ {VP chunks (Num)}
```

Figure 3: Context-free grammar for the subject-verb agreement in French, illustrated with examples.

The agreement between the subject and the verb is explicitly included in the production rule for the different types of clauses. The different types of clauses will lead to sentences with different structures, and the attractors' rule will insert one or two NPs between the subject and the verb to create different levels of linear and syntactic distance between them.

To instantiate these templates, our starting point are the examples in Franck et al. (2002, appendix 1). They provide a set of subject NPs of various complexity – including prepositional phrases, themselves of various complexity. We produced sentences based on these subject NPs by manually adding verb phrases, and by making the NPs more complex to increase the distance between the subject and the verb in the sentence. Each of these sentences is used to produce a *seed*. A seed contains all number variations of the NPs and VP in the sentence, and additional complement or prefix phrases to produce sentence variations of increased complexity. The end result is a set of 32 seeds, which provide the terminals (chunks) for the grammar. This progression from Franck et al. (2002) examples to seeds is illustrated in Figure 4.

The rules for BLM-AgrF generation The grammatical templates generate individual sentences. It is shown in Figure 2. The rules for a BLM-AgrF problem generation use the grammatical templates to generate a sequence of sentences according to specific sentence attributes. In our case, the sentence attributes that vary are the grammatical number of the subject and verb ({sg, pl}), the number of attractors ({1, 2}), and the grammatical number of each of these attractors ({sg, pl}). For each of the three clause types in the grammar, varying the above-mentioned attributes process will generate eight sentences, as illustrated in Figure 5.

To avoid biases in the process that may be caused by having an overly consistent input (i.e. the sequence following always the same sentence structure), different sentence sequences are generated for the same seed and same clause type by varying the alternation of values for the chosen attributes. For the sequences shown in Figure 5, the subject-verb grammatical number alternates between {sg, pl}, sentences with one attractor are generated first with alternating grammatical number, and then sentences with two attractors are built, but using a fixed grammatical number for the second attractor. We can alternate instead between {pl, sg}, generate sentences with two attractors first, and use a different grammatical number for the second attractor. Considering all these variations, we can obtain $\binom{2}{2} \times \binom{2}{2} \times \binom{3}{2} = 24$ different sequences for each generation seed and each clause type (illustrated in more detail in Appendix A.1, Figure 10).

Each of these sequences of eight sentences will be transformed into a problem. The first seven sentences will be the input (we call it *context*), and the eighth will be included in the answer set.

The BLM-AgrF answer set The answer set will contain the eighth sentence as the correct answer, and another five candidates generated by corrupting one of the generating rules. The candidates in the answer set are generated such that they are distinguishable from the correct answer but relevant and challenging. Unlike other RPM-like datasets in vision, we choose to have a fixed set of answer types, to be able to then do a type-based error analysis. At the bottom of Figure 5, we present the answer set for the *Main clause* sequence in the table, with the one correct and five incorrect answers, and the characteristics of the wrong answer candidates.

The BLM-AgrF dataset The dataset consists of lexical instantiations of the grammatical templates produced based on the linguistic phenomenon – a subject-verb agreement in French declarative sentences – in simple and complex structures, thanks to noun phrases of various lengths and complexity between the subject and the verb in the sentence.

The manually provided seeds are useful to control the structure of the sentence and to ensure a starting point with syntactically and semantically valid sentences. By applying the rules of BLM-AgrF generation to the 32 seeds, which generate 24 sequences for each seed and for each clause type, we obtain a first dataset consisting of 2304 BLM-AgrF problems. We call this dataset *type I*.

To introduce some lexical variation in this dataset in a semi-automatic manner, we use CamemBERT (Martin et al., 2020) to replace individual words in the sentences in the type I dataset. We mask different words in the three types of clauses one at a time, and generate the five highest probability replacements that will be substituted in the sentence sequence and the candidate set:

Main clause : mask the subject noun and second noun in the sentence, e.g.:

Les MASK avec le programme de l'experience sont en panne.

Les ordinateurs avec le MASK de l'experience sont en panne.

Completive clause : mask the subject and verb in the completive clause, and mask the nouns in the embedded clause, e.g.:

MASK suppose que les ordinateurs avec le programme de l'experience sont en panne.

Jean MASK que les ordinateurs avec le programme de l'experience sont en panne.

...

Relative clause mask the head noun and verb in the relative clause, and the subject noun and following noun in the main clause, e.g. :

Les ordinateurs avec le programme de l'experience dont MASK se servait sont en panne.

Les ordinateurs avec le programme de l'experience dont Jean se MASK sont en panne.

...

The process is illustrated in Figure 6, and a more detailed view of masking scenarios is shown in Figure in Appendix A.2.

By applying these lexical variations on the type I dataset we obtain a dataset containing 38400 BLM-AgrF problems. We call this dataset *type II*. To further increase the lexical variation, we build the *type III* dataset, where a BLM-AgrF problem consists of a combination of sentences (with the same grammatical structure) from different type II problems. As this dataset consists of resampled sentences from type II, it will also contain 38400 BLM problems.

These three datasets are split 90:10 into train and test subsets. During experiments we take a random 0.1 portion of the training set for validation.

3 Experiments

Transformers and other neural architectures have shown very high performance on a variety of NLP tasks. We describe here two baselines to investigate the difficulty in learning the underlying regularities of subject-verb agreement on the proposed dataset. Figure 7 shows the general process flow. Each sentence in the input is encoded separately using a pre-trained multilingual transformer model,[1] which were shown to capture, among others, syntactic information (Hewitt and Manning, 2019).

---

[1] The results reported here are based on sentence embeddings obtained using BERTTokenizer and BERTModel from the *transformers* Python library, using the pretrained *BERT-base-multilingual-cased* model `https://huggingface.co/bert-base-multilingual-cased`. This encoder produces an embedding of size 768 for each sentence. We have run preliminary experiments with French-specific sentence embeddings using FlauBERT (Le et al., 2020). The results were lower than when using a multilingual cased BERT language model.

Example subject NPs from Franck et al. (2002)

L’ordinateur avec le programme de l’experience
The computer with the program of the experiments

Manually expanded and completed sentences

L’ordinateur avec le programme de l’experience est en panne.
The computer with the program of the experiments is down.

Jean suppose que l’ordinateur avec le programme de l’experience est en panne.
Jean thinks that the computer with the program of the experiments is down.

L’ordinateur avec le programme dont Jean se servait est en panne.
The computer with the program that John was using is down.

A seed for language matrix generation

Jean suppose que Jean thinks that	*l’ordinateur* the computer	*avec le programme* with the program	*de l’experience* of the experiment	*dont Jean se servait* that John was using	*est en panne* is down
	les ordinateurs the computers	*avec les programmes* with the programs			*sont en panne* are down

Figure 4: Examples from Franck et al. (2002), manually completed and expanded sentences based on these examples, and seeds made based on these sentences for the subject-verb agreement BLM-AgrF dataset that contain all number variations for the nouns and the verb.

Main clause						
1		L’ordinateur	avec le programme			est en panne.
2		Les ordinateurs	avec le programme			sont en panne.
3		L’ordinateur	avec les programmes			est en panne.
4		Les ordinateurs	avec les programmes			sont en panne.
5		L’ordinateur	avec le programme	de l’expérience		est en panne.
6		Les ordinateurs	avec le programme	de l’expérience		sont en panne.
7		L’ordinateur	avec les programmes	de l’expérience		est en panne.
8		Les ordinateurs	avec les programmes	de l’expérience		sont en panne.
Completive clause						
1	Jean suppose que	l’ordinateur	avec le programme			est en panne.
2	Jean suppose que	les ordinateurs	avec le programme			sont en panne.
3	Jean suppose que	l’ordinateur	avec les programmes			est en panne.
4	Jean suppose que	les ordinateurs	avec les programmes			sont en panne.
5	Jean suppose que	l’ordinateur	avec le programme	de l’expérience		est en panne.
6	Jean suppose que	les ordinateurs	avec le programme	de l’expérience		sont en panne.
7	Jean suppose que	l’ordinateur	avec les programmes	de l’expérience		est en panne.
8	Jean suppose que	les ordinateurs	avec les programmes	de l’expérience		sont en panne.
Relative clause						
1		L’ordinateur	avec le programme		dont Jean se servait	est en panne.
2		Les ordinateurs	avec le programme		dont Jean se servait	sont en panne.
3		L’ordinateur	avec les programmes		dont Jean se servait	est en panne.
4		Les ordinateurs	avec les programmes		dont Jean se servait	sont en panne.
5		L’ordinateur	avec le programme	de l’expérience	dont Jean se servait	est en panne.
6		Les ordinateurs	avec le programme	de l’expérience	dont Jean se servait	sont en panne.
7		L’ordinateur	avec les programmes	de l’expérience	dont Jean se servait	est en panne.
8		Les ordinateurs	avec les programmes	de l’expérience	dont Jean se servait	sont en panne.

Answer set for problem constructed from lines 1-7 of the main clause sequence		
1	L’ordinateur avec le programme et l’experiénce est en panne.	N2 coord N3
2	**Les ordinateurs avec les programmes de l’experiénce sont en panne.**	correct
3	L’ordinateur avec le programme est en panne.	wrong number of attractors
4	L’ordinateur avec les programmes de l’experiénce sont en panne.	agreement error
5	Les ordinateurs avec le programme de l’experiénce sont en panne.	wrong nr. for 1^{st} attractor noun (N1)
6	Les ordinateurs avec les programmes des experiénces sont en panne.	wrong nr. for 2^{nd} attractor noun (N2)

Figure 5: BLM-AgrF instances for verb-subject agreement, with two attractors (programme, experiénce), and three clause structures. And candidate answer set for a problem constructed from lines 1-7 of the main clause sequence.

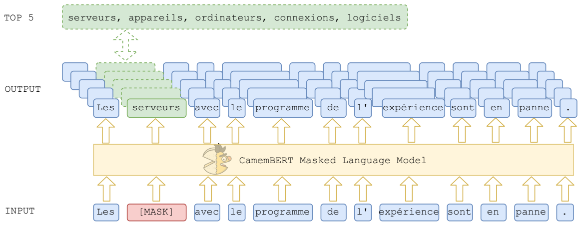


Figure 6: Creation of lexical variants by generating variations of a masked input using CamemBERT (Martin et al., 2020)

3.1 Models

We use two baseline systems – a feed-forward neural network (FFNN) and a convolutional neural network (CNN). Because the sentence embedding produced by the transformer captures structural information and we are presenting sentences in a sequence, both the FFNN and the CNN will have the chance to find patterns shared across the sentences.

The input to the FFNN is a concatenation of the sentence embeddings in the sequence (size 7 * 768), that is passed through 3 fully connected layers that gradually compress the input (7 * 768 $\xrightarrow{layer1}$ 3.5 * 768 $\xrightarrow{layer2}$ 3.5 * 768 $\xrightarrow{layer3}$ 768) to the size of a sentence representation. Because of the full connectedness between successive layers, the FFNN has the capacity of capturing patterns spread out over the entire input vector.

The input to the CNN is an array of embeddings, of size (7 x 768). This is passed through three successive layers of 2-dimensional convolutions, with a kernel size (3x3) (stride 1, no dilation). The output of the convolution is passed through a fully connected layer to compress it to the sentence representation size (768). Because of the kernel size, stride=1, and no dilation, this setup will focus on finding localized patterns in the sentence sequence array. If the NPs and verb grammatical numbers are encoded in a more localised manner within the sentence representation, this architecture should detect the patterns in the sequence.

The output of the two networks is the same – a vector representing the sentence embedding of the correct answer. The learning objective is to maximize the probability of the correct answer from the candidate answer set. Because the incorrect answers in the answer set are specifically designed to be minimally different from the correct answer, we implement the objective through the max-margin loss function. This function combines the distances between the predicted answer and the correct and erroneous ones. We first compute a score for the embedding e_i of each candidate answer a_i in the answer set $\mathcal{A}$ with respect to the predicted sentence embedding e_{pred} as the cosine of the angle between the respective vectors:

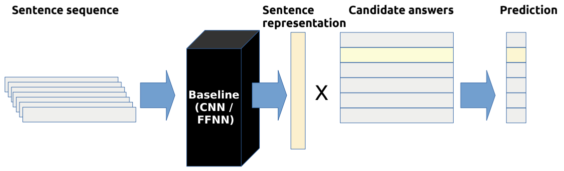


Figure 7: Illustration of the baseline setup experiments.

$$score(e_i, e_{pred}) = cos(e_i, e_{pred})$$

The loss uses the max-margin between the score for the correct answer e_c and for each of the incorrect answers e_i:

$$\mathcal{L}_a = \sum_{e_i} [1 - score(e_c, e_{pred}) + score(e_i, e_{pred})]^+$$

At prediction time, we take the answer with the highest *score* value from a candidate set as the correct answer.

3.2 Results and discussion

The results of the experiments, in terms of F1 averages over 5 runs, are shown in Figure 8, and the detailed version is in Appendix A.3. The experiments were run on a VM on the Google Cloud Platform with one NVIDIA Tesla T4 GPU and 8G memory. We ran experiments for 50 epochs, with a learning rate of 0.001 and Adam optimizer. On type II and type III data, a run took 20 minutes, on type I data 2 minutes.

As a reminder, Type I data is lexically consistent – the same vocabulary is used in all sentences in the sequence, and in the answer candidates. Type II has a limited amount of lexical variation – one word in each sentence is different. Type III is more lexically varied, with little, if any, lexical overlap between any of the context or answer candidate sentences.

The models perform well when using the full amount of training data – the heatmaps in the left column in Figure 8 –, confirming that the experimental setup used is suitable for benchmarking the problem. Type I data has available 2073 instances for training (of which 20% are used for validation).

Figure 8: F1 averages over 5 runs, for the FFNN and CNN baselines, when training on all the available training data, or on the same amount of data (2073 instances)

When training models on this amount of data for type II and type III data, the performance on these subsets drops dramatically – the heatmaps on the right in Figure 8. This indicates that the success on Type I data is due to finding superficial clues, which do not generalize to data with higher lexical variability in both the input and the candidate answers. Overall, the FFNN model performs better, potentially indicating that the interesting patterns are not localized, but rather more spread out in the inputs. With different stride and dilation parameters, the CNN might improve its performance.

A plot of the different error types made by the different models (relative to the size of the test data) is presented in Figure 9. The error plots show how the frequency of different types of errors changes when using a model trained on data of a different type than the test data.

The error analysis presented in Figure 9 reveals several insights about the data and the performance of the models.

The different errors indicate the ability of the models to learn different types of information: subject-verb agreement requires long-distance, structural information; errors on N1 and N2 tell us whether the model exhibits recency effects, thereby showing, like humans, that both structural and linear considerations come into play in learning agreement; choosing the wrong number of attractors is a very salient form of structural deviance from the correct answer and coordination is a more subtle one.

Across both models, the highest error is the N2_alt – the wrong number in the second attractor, the one closest to the verb. This proximity preference suggests that the models are rather shallow, with linear distance exerting greater influence rather than syntactic distance. In one case, when tested on type III data, the two models diverge. The pattern of results might indicate that CNNs find more localised patterns that allow them to avoid a recency bias. Coordination and number of attractors mistakes occur much less frequently, suggesting the models do learn the difference in construction and the rule of attractor sequence. This result matches our intuition that these are also the two most saliently different cases from the right answer because they differ in structure.

These results and error analysis show that curated datasets like the one presented here reveal the superficiality of the positive results on the main task. If the underlying structural rules of the subject-verb agreement had been learned properly, lexical variation would not prove so disrupting and recency effects would not be as strong.

4 Related work

The current paper does not have any direct comparison, as this is the first proposal of a dataset for language using a BLM scheme. But it is inspired by work on generating RPM problems, and on solving such problems in computer vision, and it contributes to the investigation on learning of agreement by neural networks.

Structured datasets for vision and language The automatic generation of RPM-like matrices, whether in vision or in language, is challenging, technically, in two aspects. First, how do we represent the RPM problems to tackle their variations, regularities, and irregularities? Second, how do we ensure that the generated RPMs are valid?

To overcome these challenges, some efforts have been made in computer vision: Wang and Su (2015) formulate RPMs with first-order logic, which have inspired Barrett et al. (2018) who propose Procedurally Generated Matrices (PGM) dataset through relation-object-attribute triple instantiations. Zhang et al. (2019) use the Attributed Stochastic Image Grammar (A-SIG, proposed by Zhu and Mumford, 2006) as the representation of RPM and create the Relational and Analogical Visual rEasoNing (RAVEN) dataset.

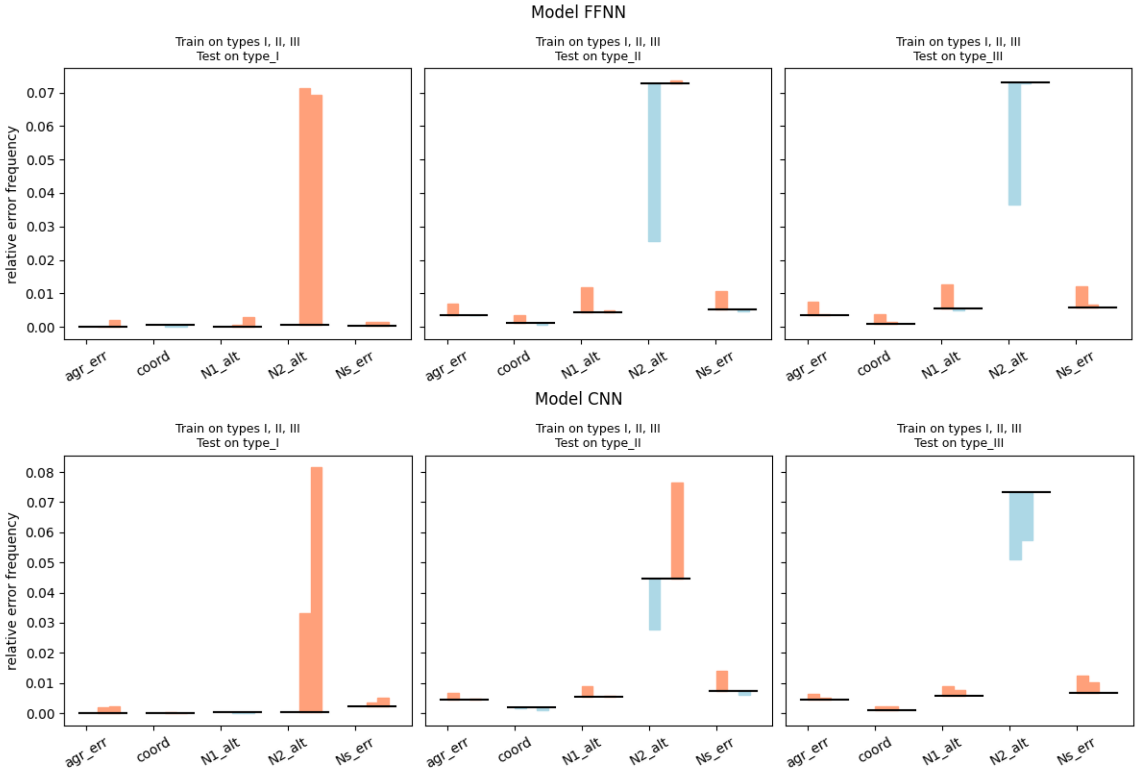


Figure 9: Relative frequencies of error types (relative to test data size) made by the different models, using models trained for all types of data. The reference – training and testing on the same data type – are given as the black lines, and we plot as vertical bars the increase (orange) or decrease (blue) in error when using a model trained on the other two data types.

These structured datasets have been mostly developed to study issues of generalisation and disentaglement. van Steenkiste et al. (2020) developed a dataset for computer vision similar to the RPMs, and evaluate the usefulness of the representations learned for abstract reasoning. They note that learning disentangled representations leads to faster few-shot learning. M'Charrak (2018) developed a large dataset, consisting of simple examples containing a few morphological markings. They use this dataset to learn disentangled sentence representations. The simplicity of the sentences does not provide a sufficiently realistic challenge from a linguistic point of view.

Learning agreement Previous work on agreement has tested recurrent neural network (RNN) language models and found that RNNs can learn to predict English subject-verb agreement if provided with explicit supervision (Linzen et al., 2016). Bernardy and Lappin (2017)'s follow-up work has shown that RNNs are better at modeling long-distance agreement if they can train the model on top of a corpus where a larger (10000 types vs. 100) vocabulary is used – the rest of the words are replaced by their POS to highlight structural patterns. Gulordava et al. (2018) explore the RNNs capacity to track abstract hierarchical structure, by predicting long-distance number agreement in various constructions in four languages (English, Hebrew, Italian, Russian). Their results suggest that RNNs can learn hierarchical grammatical phenomena and not just shallow patterns. Lakretz et al. (2021) found that individual neurons in an RNN can encode linguistically meaningful features, and propagate subject-verb number agreement information over time. In a recent paper, Li et al. (2023) investigates deeper representational issues, by contrasting two kinds of agreement, subject-verb agreement and past-participle agreement in French. They argue, based on theoretical accounts, that these superficially similar kinds of agreement, involve in fact very different abstract operations and demonstrate that transformers do reflect this difference in their representations.

5 Conclusions

In this paper we have introduced BLM-AgrF, an instance of Blackbird's Language Matrices (BLM)

(Merlo et al., 2022). This novel linguistic dataset is generatively constructed to support investigations in representation learning of grammatical rules. Each instance, consisting of a sequence of sentences and a candidate answer set, was built using a combination of rules, to provide a layered and structured dataset for learning more complex models. The various layers of the dataset allow for a variety of explorations, from disentangled sentence representations to capture structure and regularities within a sentence, to modular architectures that could capture structure and regularities in the sentence sequences. The purposefully built candidate answers supports more in-depth analyses of the behaviour of a system, and provide insights into the source of prediction errors.

Experiments using baseline set-ups – feed-forward networks and CNNs – show that the task is difficult for previously successful sentence representations and neural architectures, despite the fact that the agreement rule they are supposed to discover is rather simple. This supports our hypothesis that the task the data embodies could provide a new benchmark for modeling generalization and abstraction.

6 Limitations

Manual creation for seeds for the synthetic data The seeds to generate the data are manually chosen, and the grammar rules are specifically designed for the problem. The process may be further automated in the future through higher-level formalisation of the matrix generation process.

Language variations and linguistic phenomena The dataset described in this paper focuses on subject-verb agreement in French, with the main verb in the present tense, covering common interfering factors, different clause complexities and various depth of syntactic structures. While the simplicity of the modeled rule can be perceived as a limitation, it was a deliberate feature. The low performance of the transferred models to different test sets shows that the simple rule was not easy to model. But as systems become successful on a given dataset, new, more complex versions can be built with richer phenomena at various linguistic levels, including in morphologically-rich languages.

Human upper bounds We do not have, in this paper, an explicit experiment on human upper bounds for the different data types.

Acknowledgments

We gratefully acknowledge the partial support of this work by the Swiss National Science Foundation, through grants #51NF40_180888 (NCCR Evolving Language) and SNF Advanced grant TMAG-1_209426 to PM.

References

David Barrett, Felix Hill, Adam Santoro, Ari Morcos, and Timothy Lillicrap. 2018. Measuring abstract reasoning in neural networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4477–4486. PMLR.

Jean-Phillipe Bernardy and Shalom Lappin. 2017. Using deep neural networks to learn syntactic agreement. In *Linguistic Issues in Language Technology, Volume 15, 2017*. CSLI Publications.

Julie Franck, Gabriella Vigliocco, and Janet Nicol. 2002. Subject-verb agreement errors in french and english: The role of syntactic hierarchy. *Language and cognitive processes*, 17(4):371–404.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Yair Lakretz, Dieuwke Hupkes, Alessandra Vergallito, Marco Marelli, Marco Baroni, and Stanislas Dehaene. 2021. Mechanisms for handling nested dependencies in neural-network language models and humans. *Cognition*.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

Bingzhi Li, Guillaume Wisniewski, and Benoît Crabbé. 2023. Assessing the Capacity of Transformer to Abstract Syntactic Representations: A Contrastive Analysis Based on Long-distance Agreement. *Transactions of the Association for Computational Linguistics*, 11:18–33.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Tal Linzen and Brian Leonard. 2018. Distinct patterns of syntactic agreement errors in recurrent networks and humans. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Amine M'Charrak. 2018. Deep learning for natural language processing (NLP) using variational autoencoders (VAE). Master's thesis, ETH Switzerland.

Paola Merlo, Aixiu An, and Maria A. Rodriguez. 2022. Blackbird's language matrices (BLMs): a new benchmark to investigate disentangled generalisation in neural networks. *ArXiv*, cs.CL 2205.10866.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

John C. Raven. 1938. Standardization of progressive matrices. *British Journal of Medical Psychology*, 19:137–150.

Sebastian Ruder. 2021. Challenges and Opportunities in NLP Benchmarking. `http://ruder.io/nlp-benchmarking`.

Sjoerd van Steenkiste, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem. 2020. Are disentangled representations helpful for abstract visual reasoning? In *NeurIPS 2019*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Ke Wang and Zhendong Su. 2015. Automatic generation of raven's progressive matrices. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 903–909. AAAI Press.

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. 2019. RAVEN: A dataset for relational and analogical visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5317–5327. Computer Vision Foundation / IEEE.

Song-Chun Zhu and David Mumford. 2006. A stochastic grammar of images. *Found. Trends Comput. Graph. Vis.*, 2(4):259–362.

A Appendix

A.1 Generation process

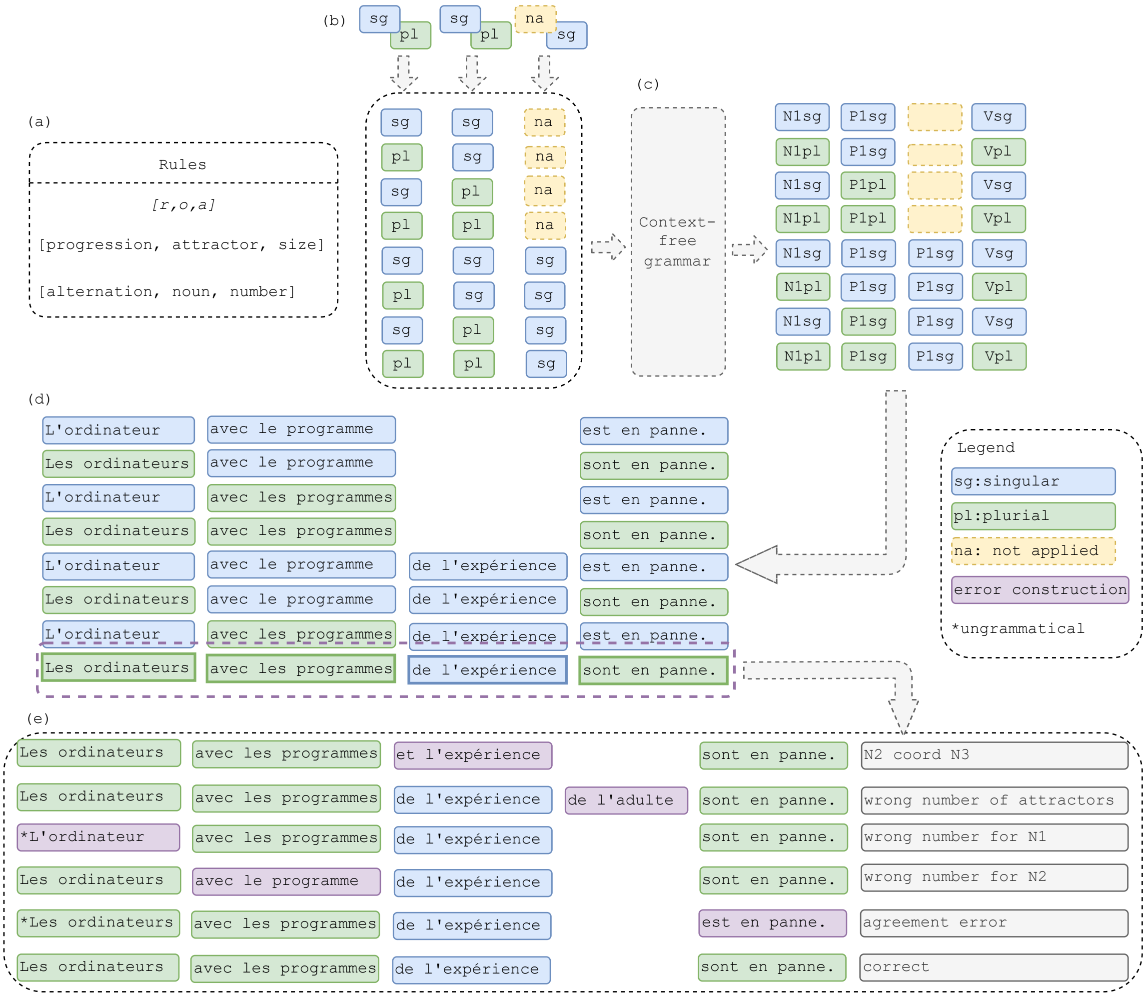
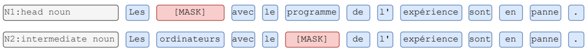
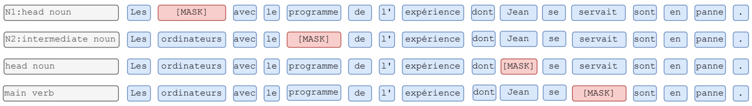


Figure 10:
Illustration of a BLMs problem generation process. Given sampled $[relation, object, attribute]$ rule tuples from (a), we first construct the abstract structure of the context with the values of the attributes of different objects in (b). We then derive, expand and prune the context-free grammar (with details in Figure 3) from each item's abstract structure into its corresponding sentence template in (c). In (d), we instantiate each item template into a sentence from the syntactic segment seed sets adapted from Franck et al. (2002). Finally, given the correct answer (last item), we modify one attribute at a time to obtain the relevant, minimally distinguishable and challenging candidate answer set in (e). The entire process is illustrated with an example of a progression in $[1, 2]$ constructed in a main clause. The error types across the dataset are uniformly distributed to avoid statistical bias.

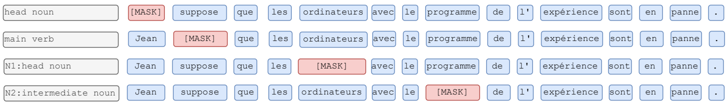
A.2 Masking



(a) Type II Main clause masking



(b) Type II Relative clause masking



(c) Type II Completive clause masking

Figure 11: Creation of lexical variants for Type II: masking strategy.

A.3 Detailed results

TRAIN ON	TEST ON	FFNN	CNN
train on full training data			
type I	type I	**0.9957** (0)	0.9887 (0.0065)
	type II	**0.6508** (0)	0.6291 (0.0089)
	type III	**0.5724** (0)	0.4992 (0.0053)
type II	type I	**0.9870** (0)	0.9861 (0.0017)
	type II	**0.9578** (0)	0.9236 (0.0062)
	type III	**0.7469** (0)	0.6159 (0.0064)
type III	type I	**0.9740** (0)	0.8909 (0.0158)
	type II	**0.9055** (0)	0.7425 (0.0094)
	type III	**0.8792** (0)	0.6714 (0.0140)
train on the same amount of data (2073 instances: 1658 train/415 validation)			
type I	type I	**0.9896** (0.0035)	0.9827 (0)
	type II	**0.6491** (0.005)	**0.6492** (0)
	type III	**0.5644** (0.0038)	0.5370 (0)
type II	type I	0.5584 (0)	**0.6234** (0)
	type II	0.4779 (0)	**0.5229** (0)
	type III	0.4622 (0)	**0.4914** (0)
type III	type I	**0.5455** (0)	0.5368 (0)
	type II	0.4768 (0)	**0.4849** (0)
	type III	**0.4669** (0)	0.4664 (0)

Table 1: Average F1 (std) scores (to 4 decimal places) for the FFNN and CNN systems, over five runs. The highest value for each train/test combination is highlighted in bold.

Robustification of Multilingual Language Models to Real-world Noise in Crosslingual Zero-shot Settings with Robust Contrastive Pretraining

Asa Cooper Stickland[⋆¶†], **Sailik Sengupta**[⋆‡], **Jason Krone**[¶‡], **He He**[‡◇], **Saab Mansour**[‡]
[†]University of Edinburgh, [‡]AWS AI Labs, [◇]New York University
`a.cooper.stickland@ed.ac.uk,{sailiks,saabm,hehea}@amazon.com`

Abstract

Advances in neural modeling have achieved state-of-the-art (SOTA) results on public natural language processing (NLP) benchmarks, at times surpassing human performance. However, there is a gap between public benchmarks and real-world applications where *noise*, such as typographical or grammatical mistakes, is abundant and can result in degraded performance. Unfortunately, works which evaluate the robustness of neural models on noisy data and propose improvements, are limited to the English language. Upon analyzing noise in different languages, we observe that noise types vary greatly across languages. Thus, existing investigations do not generalize trivially to multilingual settings. To benchmark the performance of pretrained multilingual language models, we construct noisy datasets covering five languages and four NLP tasks and observe a clear gap in the performance between clean and noisy data in the zero-shot cross-lingual setting. After investigating several ways to boost the robustness of multilingual models in this setting, we propose Robust Contrastive Pretraining (RCP). RCP combines data augmentation with a contrastive loss term at the pretraining stage and achieves large improvements on noisy (& original test data) across two sentence-level (+3.2%) and two sequence-labeling (+10 F1-score) multilingual classification tasks.

1 Introduction

Recently, multilingual pre-trained language models like mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020) and various others (Chi et al., 2021; Xue et al., 2021; Chi et al., 2022) have improved multilingual language understanding by pretraining large Transformer models on web-scale corpora (such as Wikipedia, Common-Crawl). These models achieve state-of-the-art performance on cross-lingual transfer and many multilingual NLP tasks (Wu and Dredze, 2019; Pires et al., 2019). However, a real-world system will encounter *real-world noise*, such as linguistic variations and common errors observed in textual data, that are often absent from benchmark datasets.

While prior works focused on this issue of robustness in monolingual settings (Peng et al., 2021; Sengupta et al., 2021; Tan et al., 2020), investigation has been scarce for multilingual settings. In this paper, we study the effect of realistic noise in multilingual settings and propose methods to boost the robustness of multilingual language models across four NLP tasks: Intent Classification (IC), Slot Labeling (SL), Named Entity Recognition (NER) and Natural Language Inference (NLI).

Due to the lack of multilingual noisy *evaluation* data, we synthesize benchmarks by mining noise from publicly available corpora and injecting them into the test sets associated with each of the four tasks. We conduct human validation to ensure that this noised data is indeed realistic (see examples from MultiATIS++ in Figure 1) and identify the variety of noise-types seen across languages (in §3). These analyses highlight the potential of our test-set in evaluating (and motivating future research on) multilingual robustness.

To benchmark the performance of multilingual systems, we consider accuracy metrics on two utterance-level tasks (IC% and NLI%) and F1-scores on two token-level classification tasks (SL-F1 and NER-F1). Specifically, we seek to evaluate the model's performance on the noised version of the test datasets in a *zero-shot cross-lingual setting*. In this scenario, we have training data for a task available only in one language (in our case, English) and test-data in various languages (Liu et al., 2019, 2020).

While training data augmentation increases model robustness for monolingual (i.e. English) settings, it is not immediately obvious if these robustness gains can transfer across languages, as error types can often be language-specific. For ex-

---

[⋆] Equal Contribution, [¶] Work done while at Amazon.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1375–1391
May 2-6, 2023 ©2023 Association for Computational Linguistics

Language	Noise Injection Ratio	Realistic Utt. %	Realistic Examples (test-set)	Unrealistic Examples (test-set)
French (fr)	0.1	95.4%	Me montré les vols directs de Charlotte à Minneapolis mardi matin . Quelle compagnie aérienne fut YX	Me montré des vols entre Détroit er St. Louis sur Delta Northwest US Air est United Airlines . Lister des vols de Las Vegas à Son Diego
German (de)	0.2	94.5%	Zeige mir der Flüge zwischen Housten und Orlando Welche Flüge gibt es vom Tacoma nach San Jose	Zeige mit alle Flüge vor Charlotte nach Minneapolis zum Dienstag morgen Zeige mit Flüge an Milwaukee nach Washington DC v. 12 Uhr
Spanish (es)	0.1	96.9%	qué aerolíneas vuelan de baltimore a san francesc muéstrame vuelos entr toronto y san diego	necesito información de un vuelo y la tarifa de oakland a salt lake city para el jueves antes e sus 8 am de nuevo york a las vegas el domingo con la tarde
Hindi (hi)	0.05	95.4%	मुझे डेल्टा उड़ानों के बारे में बताइए जो कोच के यात्रियों को नाश्ता देता हों मुझे मेम्फिस से लास वेगास तक उड़ान की जरूरत है	सोमवार दोपहर ने लॉस एंजिल्स से पिट्सबर्ग रविवार दोपहर को मियामी में क्लीवलैंड
Japanese (jp)	0.1	**92.3%**	来🄵水曜日にカンザスシティ初シカゴ行きでシカゴ の 午後7時ごろ到着して、🄵りのフライトが木曜日のフライト ワシントン を コロンバス間のすべてのフライトの運賃はいくら	シャ🄵ロット空港 の 土曜日 err 午后1時に出🄵する US エア🄵のフライトをリストアップして 水曜日のフェニックス🄵ミルウォ🄵キ🄵逝き
Chinese (zh)	0.1	**86.2%**	我需要4点 后 在 达拉斯起飞飞往旧金山的联程航班 请列出从纽瓦克飞往 洛杉机 的航班	然而 每天上午10点之前从密尔沃基飞往亚特兰大 拉瓜迪亚 了 豪华轿车服务要多少钱

Figure 1: MultiATIS++ test set injected with real-world noise mined from Wikipedia edits. The highest error injection ratio found to be realistic by human experts is shown alongside the realistic utterance percentage. We do not include the noisy test sets for Chinese and Japanese in our analysis owing to low ($< 95\%$) realism.

ample, typos in Devanagari script can differ from those seen in Latin scripts (e.g. स्कूल → सकउल in Devanagari showcases that a joined character is incorrectly separated into two characters in the word ‘school’).

Thus, to improve the robustness of pretrained multilingual models across noise in all languages, we propose Robust Constrastive Pretraining (RCP) that couples multilingual noisy data-augmentation with a contrastive learning loss term during pretraining; this encourages the model to develop similar representations for the original and the noised version of a sentence.

On the noisy test sets, our method improves the multilingual model performance across all metrics and multilingual tasks– IC% by 4.9% on MultiATIS++, 4.1% on MultiSNIPS; SL-F1 by 18.4 on MultiATIS++, 8.6 on MultiSNIPS; NER-F1 by 2.9 on WikiANN; NLI% by 0.7% on XNLI. In summary, our primary contributions are:

1. We construct multilingual test data to evaluate the robustness of NLP models to noise (§3).
2. We show that the performance of existing multilingual language models deteriorates on four tasks when tested on the noisy test data (§5.1).
3. We introduce Robust Contrastive Pretraining (RCP) to boost the robustness of existing multilingual language models (§5.2).

Our code and data is available on *Github (repo: amazon-science/multilingual-robust-contrastive-pretraining)* .

2 Related Work

Many prior works demonstrate the brittleness of neural models on different noise phenomena such as misspellings (Belinkov and Bisk, 2017; Karpukhin et al., 2019; Moradi et al., 2021), casing variation (van Miltenburg et al., 2020), paraphrases (Einolghozati et al., 2019), morphological variance (Tan et al., 2020), synonyms (Sengupta et al., 2021), and dialectical variance (Sarkar et al., 2022). A popular approach to improve the robustness to noise is fine-tuning models with data augmentation (Feng et al., 2021) at either the pretraining (Tan et al., 2020; Sarkar et al., 2022) or the task-training stage (Peng et al., 2021). These works consider monolingual pre-trained models and primarily focus on English. While recent works on token-free models motivate robustness in multilingual settings (Clark et al., 2021; Xue et al., 2022; Tay et al., 2021), *examining the robustness of SOTA* *multilingual pre-trained models (and improving them) remains unexplored*. Hence, we investigate– (1) are multilingual models robust to noise seen in different languages (that may be dissimilar to noise types seen in English)? (2) can we get and leverage multi-lingual noise data to improve multilingual models? and (3) do automatic data-augmentation methods designed for English improve robustness to multilingual noise?

To boost the robustness of multilingual models to diverse multilingual noise, we leverage multilingual data augmentation at the pretraining stage and use contrastive learning. Our effort complements work in computer vision that showcases contrastive learning with adversarial learning at task-training (Fan et al., 2021; Ghosh and Lan, 2021) and pretraining time (Jiang et al., 2020; Kim et al., 2020) can improve model robustness. NLP has also seen a plethora of work that leverages contrastive learning, but seldom to alleviate robustness concerns (Jaiswal et al., 2020). Similar concepts, such as Adversarial Logit Pairing (Einolghozati et al., 2019), used at task-training time have proven to be less effective than data augmentation approaches (Sengupta et al., 2021) in boosting robustness.

All the aforementioned works lack in at least one of the two novel aspects of this paper– robustness to real-world (as opposed to adversarial) noise, and/or multilinguality. Lastly, the aspect of cross-lingual knowledge transfer has been studied in the context of different NLP tasks; typically, from a high-resource language to a low-resource one, as exemplified by the XTREME benchmark (Hu et al., 2020). In this paper, we investigate the cross-lingual transferability of robustness to real-world noise.

3 Constructing Noisy Test Data

As no existing benchmarks exist to evaluate the robustness of multilingual models, we construct noisy test sets in multiple languages for four tasks. First, we construct a word-level error-and–correction dictionary by leveraging the Wikipedia edit corpora. Then, we sample replacements from this dictionary and inject them into the test data for the various multilingual tasks, focusing on replacements that only affect individual words but do not change word order. Finally, we conduct human evaluation to filter out test sets that are not deemed to be realistic by language experts.

3.1 Wiki-edit Mining

Wikipedia[2] is a public encyclopedia available in multiple languages. Wikipedia editors create and iteratively edit its contents. We leverage these edits to construct error-correction word dictionaries (later used to create noisy test data). Our approach to mining edits is similar to Tanaka et al. (2020), but we consider multiple languages (as opposed to only Japanese), and additionally create dictionaries of word-level edits.

To isolate likely useful edits, we first consider each revision page of an article and split it into a list of sentences using NLTK (Bird et al., 2009). Second, we filter out sentence pairs from two consecutive edit versions ensuring both sentences have (1) 2-120 tokens, (2) a difference if < 5 tokens, and (3) a relative edit-distance within 30% of the shorter sentence. Third, we leverage language-specific tokenizes `difflib`[3] to extract exact token-level deltas between the sentence pair. At last, we ensure word pairs (in these deltas) that have at least one character-level Levenshtein edit-distance from each other[4] and none of words are only numbers or punctuation tokens. Note that edits to Wikipedia involve changes to factual information, such as dates, rather than incorrect spelling or grammar; thus, the last step is necessary.

We can finally create a *noise dictionary* of correct-to-incorrect words that has frequency information about the different errors. For example, an element of the dictionary (in Spanish) looks like `{de: [(del, 0.52), (se, 0.32), (do, 0.1), (dë, 0.04), (en, 0.02)]}`.

3.2 Injecting Noise into Test sets

We use the noise dictionaries to create a noised version of the original test data for the four tasks– MultiATIS++ (Xu et al., 2020), MultiSNIPS, WikiANN (Pan et al., 2017) and XNLI (Conneau et al., 2018). After tokenization, we sample tokens randomly without replacement. In each sampling step, we sample based on a uniform probability distribution over the individual tokens and then check if the token exists in the noise dictionary. If so, we replace it with a noised version from the dic-

---

[2] https://meta.wikimedia.org/wiki/List_of_Wikipedias

[3] https://docs.python.org/3/library/difflib.html

[4] For Chinese characters, including Kanji, even a single character distance could imply a different word.

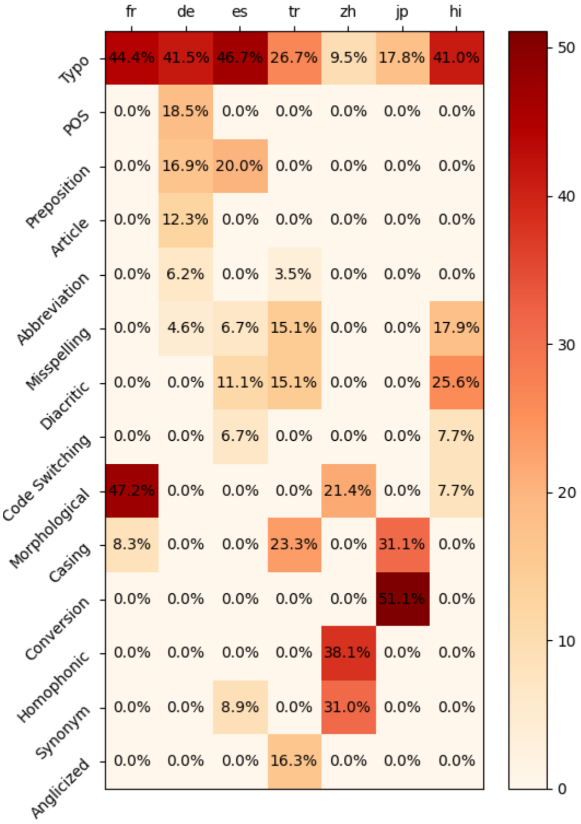
tionary; the noised version is sampled based on its probability in the noise dictionary (that is proportional to the frequency of its occurrence in the noisy corpora). This procedure continues till we noise a particular number of tokens, precisely between 1 and $\min(4, pL)$ where p a controllable fraction (chosen as a hyperparameter at first, and finalized based on human evaluation described in §3.3), and L is the number of words in the sentence.

3.3 Human Verification of Noised Test-sets

During human evaluation, we analyse the noisy data created for the MultiATIS++ dataset. We asked the language expert to assume that a user who may not be a native speaker, or in a hurry, or sloppy, was trying to find out flight information via text chat, and evaluate realism with this in mind. Note that analysis of noise types for MultiATIS++ generalizes well to other datasets as we use the same error-correction dictionaries for injecting noise into all the test-sets.

Our language experts have graduate/doctoral degrees in linguistics, computational linguistics, or natural language processing and are fluent/native speakers of the respective languages. We employed the human experts and compensated them fairly to conduct this study (see §7 for details). The experts are given 45 examples without being told that 15 examples have 5%, 15 have 10%, and 15 have 20% noised tokens and asked three questions about each example. (1) Is the noised sentence realistic, moderately realistic, or unrealistic? (2) What type of noise is present in the sentence (we supply an initial list and let them add more)? and (3) Are the intent and slot labels unchanged? Based on their initial feedback, we choose the most realistic noise fraction (i.e. 5, 10 or 20%) and provide them with 60 more examples from that set. We considered 15 utterances enough to determine the noise fraction, but used the ratings on 75 utterances for evaluating realism (see realistic utterance $\%$ in Figure 1).

In Figure 1, we summarize the results of the human evaluation. Column two shows the error injection ratio that was deemed to have more than 95% realistic utterances. We set a high cut-off of 95% to ensure we can make confident statements about the robustness of multilingual models to realistic alterations exhibited in our benchmarks. Hence, Chinese and Japanese (with a realism of 86.2% and 92.3% resp.) are omitted in our benchmarks. The last two columns highlight examples deemed as realistic and unrealistic by human experts with the noised tokens highlighted in orange.



	fr	de	es	tr	zh	jp	hi
Typo	44.4%	41.5%	46.7%	26.7%	9.5%	17.8%	41.0%
POS	0.0%	18.5%	0.0%	0.0%	0.0%	0.0%	0.0%
Preposition	0.0%	16.9%	20.0%	0.0%	0.0%	0.0%	0.0%
Article	0.0%	12.3%	0.0%	0.0%	0.0%	0.0%	0.0%
Abbreviation	0.0%	6.2%	0.0%	3.5%	0.0%	0.0%	0.0%
Misspelling	0.0%	4.6%	6.7%	15.1%	0.0%	0.0%	17.9%
Diacritic	0.0%	0.0%	11.1%	15.1%	0.0%	0.0%	25.6%
Code Switching	0.0%	0.0%	6.7%	0.0%	0.0%	0.0%	7.7%
Morphological	47.2%	0.0%	0.0%	0.0%	21.4%	0.0%	7.7%
Casing	8.3%	0.0%	0.0%	23.3%	0.0%	31.1%	0.0%
Conversion	0.0%	0.0%	0.0%	0.0%	0.0%	51.1%	0.0%
Homophonic	0.0%	0.0%	0.0%	0.0%	38.1%	0.0%	0.0%
Synonym	0.0%	0.0%	8.9%	0.0%	31.0%	0.0%	0.0%
Anglicized	0.0%	0.0%	0.0%	16.3%	0.0%	0.0%	0.0%

Figure 2: The column-wise color density (which adds up to one) shows the percentage of a different noise types observed for a particular language. The row-wise values show that some noise types (eg. homophononic) is only present for a single language (eg. zh).

Given the sentence length and similarity in task types, we use the error injection percentage determined to be the most realistic for MultiATIS++ as the error injection percentage for MultiSNIPS and Wiki-ann. For XNLI, experts deemed higher noise injection ratios (of > 0.05) to be unrealistic (15% for 0.1, 27% for 0.2) because (1) the premise, usually much longer than sentences in MultiATIS++, had (impractically high) number of noise tokens, and (2) the classification label (implies/neutral/-contradicts) sometimes changed with large noise additions. Thus, for XNLI, we choose 0.05 to be the default noise injection ratio. Finally, one expert noted the Turkish data for MultiATIS++ lacked many diacritic characters, muddling the distinction between noise injected by our procedure and existing misspellings; hence, it was ignored.

In Figure 2, we list the noise-types identified by our experts in different languages. While certain noise-types, such as typographical errors, misspellings are common across multiple languages,

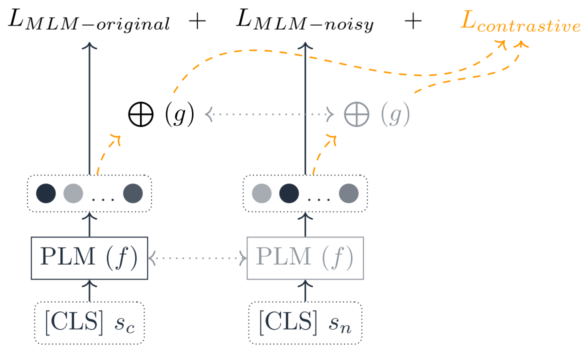


Figure 3: Loss function for fine-tuning a Pretrained Language Model (PLM) using Robust Contrastive Pre-training (RCP).

there are various language-specific noise-types, such as homophonic errors (for zh), Kanji conversion errors (for ja), anglicization (for tr) (we showcase some examples in Appendix A). Given disjoint noise types across languages, we expect that augmentation with errors seen in English (using approaches proposed by prior works) will generalize better to languages that share error types.

4 Robust Contrastive Pre-training (RCP)

Motivation and Approach While *task-time* data augmentation (aka adversarial training) has been effective to boost the robustness of pre-trained models for English, we face two major challenges– (1) lack of supervised multilingual training data in our zero-shot setting, and (2) lack of approaches to synthetically generate noise data for non-English languages. We overcome these with a multilingual data-augmentation approach at *pre-training time* that uses the multilingual Wikipedia edit corpus to expose our models to human errors during pre-training. Here, the need of ex-situ injection of noise (for test-data creation §3) is unnecessary as our edit corpus contains pairs of similar sentences, i.e. a version of the sentence before and after revision by a Wikipedia contributor (§3.1). To encourage the model to align the representations of these two sentences in the encoder's output space, we use a contrastive loss term (see Figure 3). Building on previous work on contrastive learning (Giorgi et al., 2021), Robust Contrastive Pre-training (RCP) considers the original and edited version of a sentence as positive examples and other unrelated sentences as the negative examples.

Similar to Giorgi et al. (2021) and Reimers and Gurevych (2019)), we map variable length sentences to fixed-length embeddings with a pooler $e_i = g(f(s_i))$, where $f(\cdot)$ is a transformer encoder, and $g(\cdot)$ is the mean of the token-level embeddings. Given a batch of N (noisy, clean) sentence tuples, we set our original sentence s_c as the anchor and the noisy version s_n as the corresponding positive pair. [5] Other sentences in the batch (i.e. $\neq s_n$) are deemed to be negative examples. We consider the InfoNCE/NT-Xent loss (Sohn, 2016) for our per-example contrastive loss:

$$\ell(i,j) = -\log \frac{\exp(\text{sim}(e_i, e_j))}{\sum_{i \neq k} \exp(\text{sim}(e_i, e_k)/\tau)} \quad (1)$$

where $\text{sim}(u,v) = u^T v/||u||_2||v||_2$ denotes the cosine similarity of two vectors u and v and $\tau > 0$ denotes the temperature hyper-parameter. Thus, our final contrastive loss function is

$$L_{\text{contrastive}} = \sum_{i=1}^{N} \ell(c,n) + \ell(n,c)$$

We additionally use the standard MLM loss at pre-training time, masking 15% of the input tokens of every sentence (i.e. both noisy and clean) independently. Therefore, our final loss function is

$$L = L_{\text{contrastive}} + L_{\text{MLM-noisy}} + L_{\text{MLM-original}}$$

$L_{\text{MLM-original}}$ is the MLM loss on original sentences, and ensures the model does not 'forget' its original pre-training task. $L_{\text{MLM-noisy}}$ is the MLM loss on noisy sentences, and can be thought of as data-augmentation at pre-training time.

Pre-training Details Following the Domain Adaptive Pre-Training (DAPT) approach (Gururangan et al., 2020), we start with an existing multilingual pre-trained model and fine tune it with our RCP objective. Unlike DAPT, we are not interested in specializing in a particular domain, but in increasing robustness to errors. As mentioned before, we use (unfiltered) pairs of correct/incorrect sentences from the multilingual Wikipedia archive and include sentences from the Lang8 corpus.[6] The Lang8 corpora consists of a smaller number of sentences compared to the Wikipedia corpus, but proves to be apt for our purpose; it consists of pairs of sentences– one written by a non-native speaker who is learning the language (eg. "As the winter

---

[5] One obvious choice would be for clean sentence with index $2i$, the noisy sentence has index $2i-1$.

[6] `https://sites.google.com/site/naistlang8corpora/`

Dataset	Task	Size (training)	Languages	Epochs	Learning Rate	Seeds
MultiATIS++ (Xu et al., 2020)	IC/SL	5k	de,en,es,fr,hi	80	1E-04	5
+ training data aug.		18k	de,en,es,fr,hi	20	1E-04	5
MultiSNIPS	IC/SL	13k	en,es,fr,hi	40	1E-04	5
+ training data aug.		72k	en,es,fr,hi	10	1E-04	5
WikiANN (Pan et al., 2017)	NER	20k	de,en,es,fr,hi,tr	3	2E-05	5
XNLI (Conneau et al., 2018)	NLI	392k	de,es,fr,hi,tr	5	2E-05	5

Table 1: Data-set characteristics and hyper-parameters for our experiments.

Model	Original/ Noisy	MultiATIS++		MultiSNIPS		Wiki-ann	XNLI
		IC%	SL-F1	IC%	SL-F1	NER-F1	NLI%
XLM-R$_{\text{base}}$	Original	**90.68**	**71.45**	**92.93**	**68.01**	**74.14**	**76.69**
	Noisy	**89.65**	**62.3**	**90.46**	**61.63**	**69.48**	**74.38**
mBERT	Original	86.29	64.95	78.65	59.05	73.92	70.82
	Noisy	85.42	55.17	75.35	53.71	69.38	68.44

Table 2: Performance of pre-trained multilingual models on the four multilingual datasets averaged across languages and 5 seeds. XLM-R$_{\text{base}}$ outperforms mBERT on Original and Noisy test data across all metrics.

is coming, I'm getting to feel better.") and a rewrite of this sentence by a native speaker (eg. "as the winter is coming, I'm starting to feel better."). More details about the individual corpora can be found in Appendix C.

We note that the pre-training corpus is not exactly the same set of sentences used to construct our noise dictionaries in §3.1. In this case, the only criteria for inclusion is a length difference of < 5 tokens, and a relative edit-distance of 30% of the shorter sentence (see appendix C for more details). Hence, we incorporate training data from the corpora that exhibit changes beyond simple typos (such as paraphrasing, sentence-level morphological variance) in the pre-training stage.[7]

Similar to Gururangan et al. (2020), we fine tune for 25k steps with a batch size of 2048 sentences to create two pretrained models– one with $\mathcal{L}_{\text{contrastive}} + \mathcal{L}_{\text{MLM-noisy}} + \mathcal{L}_{\text{MLM-clean}}$ (referred to as Robust Contrastive Pre-training or RCP) and an ablation without the contrastive term, i.e. $\mathcal{L}_{\text{MLM-noisy}} + \mathcal{L}_{\text{MLM-clean}}$. The latter setting represents a pure (pre-training time) data augmentation approach such as Tan et al. (2020) (termed *p(aug)* in Table 3). See Appendix D for more hyper-parameters and settings.

5 Experiments and Results

We divide this section into three parts. In §5.1, we analyze the robustness of popular multilingual language models in the zero-shot cross-lingual setting. In §5.2, we show that Robust Contrastive Pre-training (RCP) improves the robustness of existing baselines on noisy test-data for all tasks– joint intent classification and slot labeling (IC-SL), Slot-Labeling (SL) Named Entity Recognition (NER) and Natural Language Inference (NLI)– and (not only maintains but) improves performance on the original test data. Finally, in §5.3, we conduct failure mode analysis for MultiATIS++ and discover that the model trained with RCP makes more explicable sequence-labeling errors (for slot-value prediction) in comparison to existing baselines.

Setup We consider four datasets (shown in Table 1) and four metrics for evaluation. Two of these metrics consider sentence classification accuracy– Intent classification Accuracy (IC%) for the goal-oriented dialog text datasets MultiATIS++ and MultiSNIPS, and classification accuracy (NLI%) for XNLI. We also consider F-score for sequence-labeling tasks– Slot Labelling (SL-F1) for MultiATIS++ and Multi-SNIPS++ and Named Entity Recognition (NER-F1) for Wiki-ann. Table 1 shows the languages present in the noisy test data and the size of the English training data used in our zero-shot cross-lingual setting. Note that for task-time data augmentation, we follow the strategy of *aggregate noise augmentation* proposed in (Sengupta et al., 2021) for English, which involves augmenting training data with a variety of synthetic noise types such as typos, making words ALL-CAPS, abbreviations etc. As this augmentation procedure increases the size of the training data-set ≈ 3.5 times for MultiATIS++ and ≈ 5.5 times for

---

[7] Unfortunately, the benefit of including sentence-level noise in the pre-training phase is not directly examined by our benchmarks, which focus more on word-level noise.

Task	Metric	XLMR	XLMR +p(aug)	XLMR +t(En-aug)	XLMR +RCP (Ours)	XLMR +RCP+t (Ours)	Gain
MultiATIS++	IC%	89.65	93.10	91.26	93.80	**94.57**	+4.92
	SL-F1	62.30	67.47	74.62	67.45	**80.68**	+18.38
MultiSNIPS	IC%	90.46	93.98	91.60	93.79	**94.53**	+4.07
	SL-F1	61.63	66.67	66.44	67.69	**70.20**	+8.57
Wiki-ann	NER-F1	69.48	72.32	-	**72.37**	-	+2.89
XNLI	NLI%	74.38	74.83	-	**75.06**	-	+0.68

Table 3: Average performance across languages and five seeds. We abbreviate the baselines, multi-lingual pre-training time augmentation as p(aug), and English task-time (aggregate) data augmentation as t(En-aug). 'RCP' stands for 'Robust Contrastive Pre-training', and 'RCP + t' means combining RCP with task-time data augmentation. 'Gain' refers to the increase in performance of the best method vs. XLM-R$_{\text{base}}$.

MultiSNIPS, we find that training for fewer epochs yields the best results.

5.1 Robustness of Multilingual Models

We compare the robustness of two popular pre-trained language models– XLM-R$_{\text{base}}$ and multilingual BERT in the zero-shot cross-lingual setting.[8] In this setup, we fine-tune the pretrained language models on the task-training data in English and test (zero-shot) on multilingual test sets. The results reported in Table 2 are averaged across multiple languages for brevity (and provide a detailed breakdown in Appendix E). A secondary goal of this experiment was to decide which pre-trained model to use for further experiments and we base our judgements on twelve metrics across four datasets.

Noise always leads to a decrease in performance. On average, the accuracy of both models decreases by $\approx 2\%$ for sentence-level tasks (IC%, NLI%), and by ≈ 6.6 F1-points on sequence-labeling tasks (SL, NER), on noisy data compared to clean data. This can perhaps be explained by the ability to ignore a particular token for sentence-level tasks, whereas every token, including noisy ones, need to be assigned a label for sequence-labeling tasks.

We observe that XLM-R$_{\text{base}}$ outperforms mBERT on all the twelve metrics. For sentence-level tasks (i.e. IC%, NLI%), XLM-R$_{\text{base}}$ outperforms mBERT by 8.43% on average on the noisy test-sets and for sequence-tagging tasks (i.e. SL, NER), XLM-R$_{\text{base}}$ outperforms mBERT by 5.1 F1-points. In general, XLM-R$_{\text{base}}$ also seems to be a model better suited for these tasks in the zero-shot cross-lingual setting, as we also see similar gains when using XLM-R$_{\text{base}}$ on the clean data.

Task	Metric	XLMR	Ours	Gain
MultiATIS++	IC%	90.68	95.32	+4.64
	SL-F1	71.45	84.07	+12.62
MultiSNIPS	IC%	92.93	95.66	+2.73
	SL-F1	68.01	74.39	+6.38
Wiki-ann	NER-F1	74.14	76.34	+2.2
XNLI	NLI%	76.69	76.75	+0.06

Table 4: Comparison of our RCP method with the baseline XLM-R$_{\text{base}}$ model on the original (clean) test data.

Breaking the results down by language (see Appendix E for detailed results), XLM-R$_{\text{base}}$ outperforms mBERT on average across all languages. Specifically XLM-R$_{\text{base}}$ outperforms mBERT on German (in 6/8 metrics), on Spanish (10/10), on French (8/12), on Hindi (12/12), and on Turkish (4/4). As German is missing in MultiATIS++ and Turkish is only present in WikiANN and XNLI among the four datasets, the overall number of metrics is less than 12 for these two languages. Given these results, we consider XLM-R$_{\text{base}}$ as the baseline multilingual language model in the rest of our experiments.

5.2 Robust Contrastive Pre-training Results

To showcase the efficacy of our RCP approach, we compare our approach to a popular multilingual model XLM-R$_{\text{base}}$, which performed best in the previous section, and two augmentation solutions that were proposed earlier and shown to improve robustness of English language models to real-world noise. First, we consider a pre-training time data augmentation approach, similar to Tan et al. (2020), by continuing to pre-train XLM-R$_{\text{base}}$ on *noisy* multilingual data; see section 4. Next, we consider augmenting task-time data with a combination of various noise types, following Sengupta et al. (2021) that shows using this aggregate data augmentation during task-time finetuning improved

---

[8] We also considered Canine-c (Clark et al., 2021), a token-free baseline, but observed poor performance compared to XLM-R$_{\text{base}}$ and BERT on IC-SL tasks (see Table 10).

Error Type	Utterance	Slot-lables
Hallucination	*Ichs* brauche einen Flug von Memphis nach Tacoma, der ***uber*** Los Angeles fliegt	✓ O (über) ! airline_code
Contextual	Zeige *mit der* Erste-Klasse und Coach-Flüge vom **JFK** nach Miami	✓ fromloc.airport_code ! toloc.airport_code

Table 5: Examples of slot labeling errors in German– errors are in *italics*; misclassified tokens are **bold**.

performance on both noisy and clean data for IC-SL tasks like ATIS and SNIPS. For the latter, we treat it as a baseline for zero-shot cross-lingual transfer for the dialog-datasets–MultiATIS++ and MultiSNIPS– and also combine it with our pre-training time approaches.

As shown in Table 3, our approach can improve the performance of current multilingual models across all 4 tasks and datasets. For the multilingual goal-oriented dialog datasets, our approach coupled with task-time augmentation outperforms all the other methods. We observe that the gain for SL tasks is higher than that obtained for IC tasks. Although we analyze the SL results further in §5.3, we highlight that IC accuracy is less affected by noise than SL F1; this provides more headroom for improving SL metrics. The highest gains are observed for Hindi where the XLM-R$_{\text{base}}$ model has the worst SL performance on noisy data (42.86 for MultiATIS++, 36.93 for MultiSNIPS). Likewise, we also observe improvement on XNLI% and NER-F1; the largest improvement is again seen on the noisy data for Hindi. Overall, the gain on sequence-labelling tasks is larger than the gain on sentence-level classification tasks.

Does this improvement on noisy data come at the cost of worse performance on clean data? In Table 4, we show that the best performing models shown in Table 3 (XLMR+RCP+t for MultiATIS++ and MultiSNIPS, and XLMR+RCT for WikiANN and XNLI) also improve the performance on clean test data. Further, the magnitude of growth seen on clean data is like the ones seen on the noisy test data. For slot-labeling errors, we observe a particular kind error which occurs on both clean and noisy data that our model mitigates; we provide more details on this in the next section. For IC and XNLI, we found no specific error pattern that distinguishes between XLM-R$_{\text{base}}$ and our model. Thus, we believe that our approach mostly improves the overall quality of the model's representation rather than just its downstream robustness. In the future, one can consider if an upper bound on model quality exists beyond which the tension between accuracy on clean data and robustness to real-world noise emerges (Tsipras et al., 2018).

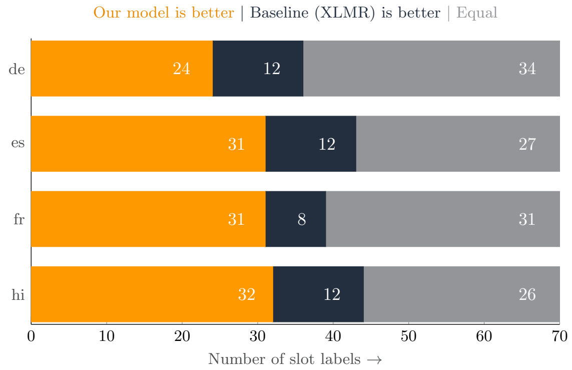


Figure 4: Comparing the number of slot labels for which our model *vs.* the baseline performs better.

Finally, we note that beyond improving performance on clean and noisy data, our approach reduces the *disparity* in performance between the clean and noisy test sets. For MultiATIS++, the disparity reduces by 0.3% for IC% and 5.76 for SL-F1; for MultiSNIPS, it reduces by 1.34% for IC% and 2.19 for SL-F1; for WikiANN, it reduces by 0.68 for NER-F1; and for XNLI, it reduces by 0.9% for NLI%.

5.3 Result Analysis

Given the large improvement seen on sequence labeling tasks, we zoom in on the SL metrics for MultiATIS++. In Figure 4, we show the number of slot labels on which our method outperforms (has fewer misclassifications than) the baseline, vice versa, and where they perform equally. Our method clearly out-performs the baseline on at least twice the number of slot-labels– 2× better on German, ≈ 2.6× times on Spanish and on Hindi, and ≈ 4× on French. Across all languages, our model always outperforms XLM-R$_{\text{base}}$ on eight slot-labels. These slots correspond to relative times ('leaves in the *evening*'), relative dates ('traveling the *day after tomorrow*'), relative costs ('*cheap* flights'), meal names ('flights that offer *breakfast*'), and carrier tokens/non-slot values ('*that offer* breakfast'). We postulate these slot values are more common in the

N/O	Model	de	es	fr	hi
Noisy	XLMR	315	358	413	671
	XLMR+RCP+t	21	123	33	204
Original	XLMR	208	262	334	460
	XLMR+RCP+t	19	106	22	180

Table 6: Reduction in hallucination error (i.e. model identifies irrelevant tokens as a slot value) counts.

Languages	de	es	fr	hi
(r1) Top-confusion changes to no-label (w/ RCP)	7	8	6	17
(r2) Confusions becomes more explicable (w/ RCP)	8	3	3	4

Table 7: Number of slot-labels that our model misclassified to (r1) a no-slot or (r2) a more explicable slot-label.

pre-training data compared to proper nouns such as airline, airport or city names and thus, understood in noisy contexts. In turn, variations of these words are mapped closer in the embedding space and the classifier is more robust to such errors.

Upon further analysis, we observe two distinct patterns– (1) reduction in *hallucination errors*, i.e. errors where an irrelevant carrier phrase token is labeled to be a slot value, and (2) errors become more contextual– misclassification is to related classes (see examples in Table 5).

In Table 6, we highlight the distribution of hallucination errors and observe that the number of carrier phrase tokens that the baseline XLM-R_{base} misclassifies as a slot-value reduces (by $> 10\times$ for German and French, and $\approx$2-3$\times$ for Hindi and Spanish) with our approach on both the original and the noisy test data. This observation aligns with our initial reasoning that the contrastive loss term at pre-training time helps the model develop a better understanding of non-slot words as the model learns to identify such words (and their noisy forms) in both linguistically correct and noisy contexts. Note that the latter signal is missing for the XLM-R_{base} baseline.

For a subset of the slot labels, the class to which it was misclassified (with the highest frequency) differed between the XLM-R_{base} baseline and our model. In Table 7, we highlight two scenarios where the most-confused label changed from (r1) an incorrect slot label (eg. *meal_code* → *airline_code*) to no-label (i.e. *meal_code* → O), and (r2) from an inexplicable slot label (*state_code* → *transport_type*) to a more explicable one (*state_code* → *state_name*) when the RCP method is used (we use the explicable/inexplicable terminology of Olmo et al. (2020)). Thus, our approach inadvertently improves the explicability of the failures made during slot-labeling.

6 Conclusion

In this paper, we investigate the robustness of pretrained multilingual models in the zero-shot cross-lingual setting on four tasks– intent classification, slot labeling, named entity recognition, and natural language inference. Given the dearth of existing datasets to benchmark the robustness of existing multilingual models, we develop noisy test data by injecting errors mined from an edit corpus (and conduct expert evaluation for quality assurance). Our identification of noise types across various languages motivates the necessity of language specific investigation in the future. Finally, demonstrate existing baselines perform poorly in the presence of noise in the test data and propose Robust Contrastive Pretraining to boost the robustness of these multilingual models.

7 Ethical Considerations

For the human annotation tasks of (1) identifying language-specific noise types, and (2) ranking their realism, we leveraged the effort of full-time employees at Amazon. The annotators had advanced degrees in linguistics or natural language processing, and were fluent/native in the languages they annotated. Amazon compensated them under a competitive industry rate, which is above the minimum hourly pay rate, for their particular job role (which included Applied/Research Scientists, Software/Language Engineers, Linguists, and Language Consultants).

Acknowledgements A special thanks to Saab, Batool Haider and M. Saiful Bari for sharing with us the MultiSNIPS dataset. In addition, we want to express our gratitude to members of the AWS AI Lab for their valuable comments, suggestions, and participation in our pilot and human labeling studies (in no particular order)– Sebastien Jean, Volha Belash, Arshit Gupta, Berk Sarioz, Maansi Shandilya, Raphael Shu, Abhilash Panigrahi, Lorenzo Lambertino, and Yi Zhang. Finally, we are grateful to the anonymous reviewers who have helped us improve this paper.

8 Limitations

8.1 The Umbrella of Realistic Noise

‘Realistic noise’ is too abstract a category. We mostly concern ourselves with real-world errors and their corrections appearing in existing corpora (with criteria like a small character-level edit distance). But this could include things like better paraphrasing, use of more appropriate synonyms or morphology that can be viewed as language variation rather than noise; this could be one reason we notice improvements on the original (i.e. unnoised) test data. Yet, to distinguish ourselves from the terminology of synthetic or adversarial noise, we choose this (imperfect) terminology of real-world/realistic noise as in Sengupta et al. (2021) to bracket all our noise types under a single class.

8.2 Language Choice and Diversity

This work considers (relatively) high-resource languages. This makes it easier for us to find publicly available corpora from where we can mine error/-correction data and use it to improve the model’s understanding of errors and, in turn, boost their robustness to real-world noise. But this is only the first step towards developing an understanding of noise phenomena in languages beyond English, bench-marking multi-lingual model performance in such settings, and improving their robustness. Further, we do notice that Hindi (and, to some extent, Turkish) are relatively low resource languages when it comes to pre-training data (see Table 8 in Appendix). We hope future work builds on this and explores a greater variety of languages.

8.3 Zooming-in on Individual Tasks

Many of our human studies are based on a subset of datasets (eg. MultiATIS, XNLI). It is possible individual tasks and further, individual datasets need more fine-grained human attention. Given language expertise for several datasets and several languages is difficult/costly, we made the choice to concentrate on a smaller number of datasets in order to provide a more rigorous analysis. We hope future work can expand the number of tasks and datasets covered so we have a more comprehensive analysis of how multilingual noise affects pre-trained models.

References

Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O’Reilly Media, Inc.".

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Zewen Chi, Shaohan Huang, Li Dong, Shuming Ma, Bo Zheng, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, Heyan Huang, and Furu Wei. 2022. XLM-E: Cross-lingual language model pre-training via ELECTRA. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6170–6182, Dublin, Ireland. Association for Computational Linguistics.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2021. CANINE: pre-training an efficient tokenization-free encoder for language representation. *CoRR*, abs/2103.06874.

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

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

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

Arash Einolghozati, Sonal Gupta, Mrinal Mohit, and Rushin Shah. 2019. Improving robustness of task oriented dialog systems. *arXiv preprint arXiv:1911.05153*.

Lijie Fan, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Chuang Gan. 2021. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? *Advances in Neural Information Processing Systems*, 34.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Aritra Ghosh and Andrew Lan. 2021. Contrastive learning improves model robustness under label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2703–2708.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895, Online. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2020. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2.

Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. 2020. Robust pre-training by adversarial contrastive learning. *Advances in Neural Information Processing Systems*, 33:16199–16210.

Vladimir Karpukhin, Omer Levy, Jacob Eisenstein, and Marjan Ghazvininejad. 2019. Training on synthetic noise improves robustness to natural noise in machine translation. *arXiv preprint arXiv:1902.01509*.

Minseon Kim, Jihoon Tack, and Sung Ju Hwang. 2020. Adversarial self-supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:2983–2994.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Zihan Liu, Jamin Shin, Yan Xu, Genta Indra Winata, Peng Xu, Andrea Madotto, and Pascale Fung. 2019. Zero-shot cross-lingual dialogue systems with transferable latent variables. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1297–1303, Hong Kong, China. Association for Computational Linguistics.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8433–8440.

Pooya Moradi, Nishant Kambhatla, and Anoop Sarkar. 2021. Measuring and improving faithfulness of attention in neural machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2791–2802, Online. Association for Computational Linguistics.

Alberto Olmo, Sailik Sengupta, and Subbarao Kambhampati. 2020. Not all failure modes are created equal: Training deep neural networks for explicable (mis) classification. *ICML Workshop on Uncertainty and Robustness in Deep Learning*.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Baolin Peng, Chunyuan Li, Zhu Zhang, Chenguang Zhu, Jinchao Li, and Jianfeng Gao. 2021. RADDLE: An evaluation benchmark and analysis platform for robust task-oriented dialog systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4418–4429, Online. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Soumajyoti Sarkar, Kaixiang Lin, Sailik Sengupta, Leonard Lausen, Sheng Zha, and Saab Mansour. 2022. Parameter and data efficient continual pre-training for robustness to dialectal variance in arabic. In *NeurIPS 2022 Workshop on Efficient Natural Language and Speech Processing (ENLSP)*.

Sailik Sengupta, Jason Krone, and Saab Mansour. 2021. On the robustness of intent classification and slot labeling in goal-oriented dialog systems to real-world noise. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 68–79, Online. Association for Computational Linguistics.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.

Yu Tanaka, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. 2020. Building a Japanese typo dataset from Wikipedia's revision history. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 230–236, Online. Association for Computational Linguistics.

Yi Tay, Vinh Q Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2021. Charformer: Fast character transformers via gradient-based subword tokenization. *arXiv preprint arXiv:2106.12672*.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. 2018. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*.

Emiel van Miltenburg, Chris van der Lee, Thiago Castro-Ferreira, and Emiel Krahmer. 2020. Evaluation rules! on the use of grammars and rule-based systems for NLG evaluation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 17–27, Online (Dublin, Ireland). Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Weijia Xu, Batool Haider, and Saab Mansour. 2020. End-to-end slot alignment and recognition for cross-lingual NLU. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5052–5063, Online. Association for Computational Linguistics.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

A Examples of Noise/Errors in the test set

In this section, we highlight an example of some of the unique noise types observed for certain languages shown in Figure 5.

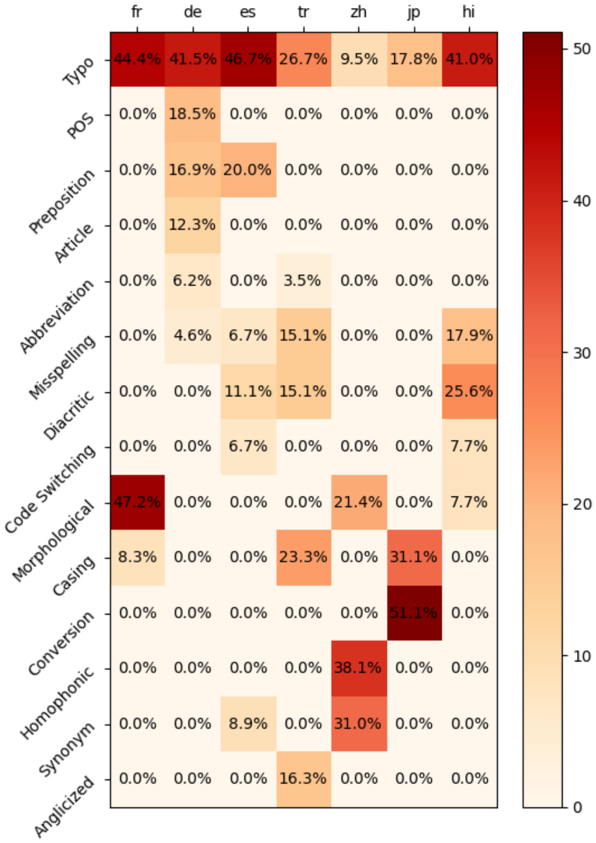


Figure 5: Noise types seen across various languages.

A.1 Typographic Errors (Typos)

Two examples follow for Hindi and Chinese, where experts evaluated based on the Indic Script and the Pinyin keyboards (which is what they use regularly) respectively.

Language	Examples
Hindi (hi)	सोमवार को बरबैंक [से|के] मिल्वौकी तक उड़ानें
Chinese (zh)	列出美国航空周三从密尔沃基飞往圣何塞的航班 **[列出 | 例出]** 美国航空周三从密尔沃基飞往圣何塞的航班

A.2 Preposition Errors

We noticed language experts tagged preposition errors for French and German. Examples follow:

Language	Examples
French (fr)	Je veux un vol aller-retour [de | à] Memphis à Seattle .
German (de)	Wie sieht es [am | im] Mittwoch morgen mit Flügen von DC nach Oakland aus

A.3 Diacritic Errors

Some languages use diacritic characters; although even these diacritics may greatly differ depending on script. Examples from Hindi and Spanish follow.

Language	Examples
Spanish (es)	¿puedo tomar el vuelo [más | mas] corto de milwaukee a orlando ?
Hindi (hi)	d9s किस [तरह | तरह्] का विमान हैं

A.4 Conversion Errors

Kanji conversion error. This error was unique to the Japanese language. Examples follow.

Language	Examples
Japanese (ja)	アトランタからセントルイスまでの火曜日午後2時30分 [以前 | 依然] のフライト

A.5 Homophonic Errors

This error was unique to Chinese. Words with the same pronunciation (potentially with different tones), but different spelling. Examples follow.

Language	Examples
Chinese (zh)	请列出从 [洛杉矶 | 洛杉机] 飞往夏洛特的航班

A.6 Synonym

Experts marked these as use of a different synonym in Spanish and Chinese only. Note that such variations may not be erroneous but is still considered a noise given they are not used in the original training/testing data in the given context as much. Examples follow.

Language	Examples
Spanish (es)	el próximo miércoles , me gustaría salir de kansas city en [un | el] viaje a chicago que llegue a chicago alrededor de las 7 p m.
Chinese (zh)	请列出从 ewr [到 | 直到] 纽约市的地面交通

A.7 Anglicized

We observed this errors only for Turkish and noticed that experts marked scenarios where an alphabet in the native script was replaced with a particular one in the latin script. Examples follow (note that Turkish examples are drawn from the XNLI dataset, while the others were drawn from MultiATIS++).

Language	Examples
Turkish (tr)	Sonrasında, ilk ziyareti yapmış olan aynı temsilci, soruları cevaplamak ve şikayet örneğinde not edilen sorunları tartışmak [için | icin] yeni sağlayıcıyı yeniden ziyaret eder.
	Konfederasyonun hukuk felsefesi, hem maddi hem de üslupla [karşı | karsi] karşıya geldi.

B Chinese and Japanese Edit Mining

Our two character edit distance criteria for obtaining word-level correct-noisy pairs of words does not work well for Chinese characters, including Kanji for Japanese. This is because words are made up of only a small number of characters (relative to e.g. latin scripts). So we can completely change the semantics with only a small character-level edit distance. We therefore used different noise types: Homophonic and Synonym errors for Chinese and Kanji Conversion errors for Japanese, with brief descriptions and examples in Appendix A. In order to collect homophonic errors we converted words to pinyin[9] (without tone markers) and checked if they were the same in pinyin but different in Chinese characters. To collect synonym noise we labelled words with part-of-speech (POS) tags [10], and kept words that weren't labeled as nouns, verbs, adverbs, keeping e.g. prepositions and conjunctions, with the hope that these would be less likely to involve the kind of big semantic changes you might get with changes to e.g. proper nouns like place names.

However this process was largely driven by trial and error and more work is needed to create a principled pipeline that creates a realistic noise dictionary for these languages.

Finally for Kanji we re-use the criteria of Tanaka et al. (2020) as we re-use their dataset of sentence pairs: checking if the two sentences (containing Kanji) have the same reading.

C Data Details

Table 8 shows the number of Wikipedia and Lang8 sentences (in Millions) we used for fine-tuning the multilingual models in the pre-training stage (§4). As stated earlier, the proportion of data obtained from the Lang8 corpus is less than Wikipedia for most languages except English (where it is comparable) and Japanese (where Lang8 has ≈ $4x$ the data compared to the Wikipedia corpus). In general, Hindi (and Turkish) stand out as a relatively low-resource language in our investigation with less than 0.5 Million sentences.

Language	Lang8	Wikipedia	Total
en	2.5	3.8	6.3
de	0.2	13	13.2
es	0.2	7.6	7.8
fr	0.2	10.7	10.9
hi	0.001	0.1	0.101
ja	4.2	1	5.2
tr	0.02	0.4	0.42
zh	0.6	1.9	2.5

Table 8: Number of sentences (in millions) used for pre-training.

Language	# Pairs (in Millions)
en	0.13
de	0.33
es	0.21
fr	0.27
hi	0.04
ja	0.05
tr	0.25
zh	0.01

Table 9: Number of Error pairs by language.

Table 9 lists the number of correct/incorrect pairs (in Millions) used for noise dictionaries to create the test-sets for the various languages (§3). Here too, we can observe that the number of corrections are relatively less for Hindi. Interestingly, the number of errors for Chinese are the least although it representation is significantly more compared to Hindi. This low number of errors is inline with our human studies where even the 5% error injection was deemed to be unrealistic; futher, such low pairs of errors also reduced the diversity of our test set, which would eventually results in a lower-quality test-set. Hence, we drop it from our evaluation benchmarks.

D Pre-training Settings

For our experiments with Robust Contrasting Pretraining (§4) and variants we use the following hyperparameters and setup. We train on 4 Nvidea V100 GPUs, with a per-gpu batch size of 8 sentences with a maximum sequence length of 128 tokens, and 64 gradient accumulation steps, for an overall batch size of $64 \times 8 \times 4 = 2048$ sentences. We use a masked language modeling mask probability of 15% and a

---

[9] Using the `pinyin` Python package `https://pypi.org/project/pinyin/`

[10] With the `jieba` Python package `https://pypi.org/project/jieba/`.

learning rate of 1e-4 with the Adam optimizer (Kingma and Ba, 2015), and used 16-bit floating point operations. See below for the arguments of the Huggingface transformers (Wolf et al., 2020) masked language modelling script which we modified[11]

```
python -m torch.distributed.launch --nproc_per_node 4 run_mlm.py \
    --model_name_or_path xlm-roberta-base \
    --gradient_accumulation_steps 64 \
    --validation_split_percentage 1 \
    --per_gpu_train_batch_size 8 \
    --dataloader_num_workers 32 \
    --model_type xlm-roberta \
    --mlm-probability 0.15 \
    --learning_rate 1e-4 \
    --num_train_epochs 5 \
    --max_seq_length 128 \
    --line_by_line \
    --do_train \
    --do_eval \
    --seed 42 \
    --fp16
```

E Per-language Results

Table 10 shows the performance of multilingual models like m-BERT and XLM-R$_{\text{base}}$ on individual languages. We note that the reduction in performance for high-resource language (e.g. German, French, English) is higher than low-resource languages for several settings. To explain this seemingly surprising result, first notice that the metrics on low-resource languages are already bad, even on clean data. Second, the variety of noise seen for low resource languages is less (see Table 9) compared to high-resource settings. Hence, the effect of less diverse noise in low-resource languages doesn't have as large an adverse effect on already poorly performing models.[12]

Another hypothesis, pending future investigation, is that multi-lingual models trained on more high-resource language data overfit to clean test-sets for these languages and fail to generalize better when faced with noise. For low resource languages, the performance on clean data is already poor because of a lack of sufficient language understanding that prevents over-fitting.

---

[11] `https://github.com/huggingface/transformers/blob/main/examples/pytorch/language-modeling/run_mlm.py`

[12] We are told a saying goes (coincidentally) in Hindi, *mare hue ko kya maroge, saheb?*. It implies you cannot do much (by adding noise) to kill the (model that is already) dead.

Dataset	Model	Metric	C/N	de	en	es	fr	hi	tr	Avg.
MultiATIS++	*XLMR*	IC%	C	92.4	98.7	92.0	90.6	79.6	-	90.7
			N	90.9	97.6	91.8	89.5	78.4	-	89.6
		SL-F1	C	74.4	96.0	73.6	70.4	42.9	-	71.5
			N	67.3	82.2	68.2	65.6	38.2	-	62.3
	mBERT									
		IC%	C	83.3	98.3	84.7	88.8	76.3	-	86.3
			N	81.2	97.6	84.3	87.9	76.1	-	85.4
		SL-F1	C	59.9	96.0	65.1	69.8	33.9	-	65.0
			N	51.6	78.5	60.2	64.3	31.3	-	55.2
	(XLMR vs mBERT)			4,0	1,3	4,0	4,0	4,0		
	Canine-c	IC%	C	66.32	96.51	78.41	76.06	71.55	-	77.77
			N	65.13	95.90	78.08	75.06	71.15	-	77.06
		SL-F1	C	31.56	92.19	19.52	23.67	22.81	-	37.95
			N	32.42	78.51	20.25	24.41	22.45	-	35.61
MultiSNIPS++	*XLMR*	IC%	C	-	98.8	94.0	91.3	87.6	-	92.9
			N	-	98.4	92.4	87.0	84.1	-	90.5
		SL-F1	C	-	96.9	72.0	66.2	36.9	-	68.0
			N	-	92.7	63.3	57.7	32.8	-	61.6
	mBERT									
		IC%	C	-	98.9	88.0	88.5	39.3	-	78.6
			N	-	98.2	84.1	82.9	36.2	-	75.4
		SL-F1	C	-	96.5	65.4	59.9	14.5	-	59.1
			N	-	91.3	58.1	52.4	13.0	-	53.7
	(XLMR vs mBERT)				3,1	4,0	2,2	4,0		
	Canine-c	IC%	C	-	69.39	32.88	36.39	23.28	-	40.48
			N	-	69.30	32.57	34.99	23.68	-	40.13
		SL-F1	C	-	0.89.31	24.09	23.06	6.93	-	35.85
			N	-	87.86	22.3	21.49	7.02	-	34.67
WikiANN	*XLMR*	NER-F1	C	74.9	-	75.2	77.2	67.5	75.9	74.1
			N	71.6	-	70.0	71.1	65.1	69.5	69.1
	mBERT									
		NER-F1	C	78.6	-	72.1	79.5	66.2	73.1	73.9
			N	75.4	-	67.1	74.2	63.0	67.3	69.4
	(XLMR vs mBERT)			0,2		2,0	0,2	2,0	2,0	
XNLI	*XLMR*	NLI%	C	76.4	84.6	78.8	77.9	69.7	72.9	76.7
			N	72.6	80.7	76.4	75.7	70.3	70.6	74.4
	mBERT									
		NLI%	C	71.1	82.0	74.9	74.2	60.5	62.2	70.8
			N	67.5	77.9	73.1	71.8	61.5	59.1	68.4
	(XLMR vs mBERT)			2,0	2,0	2,0	2,0	2,0	2,0	

Table 10: Per-language results of cross-lingual transfer from English data (average of 5 random seeds) across 4 datasets analyzed in §5.1 to compare between existing pre-trained multilingual models.

Unsupervised Anomaly Detection in Multi-Topic Short-Text Corpora

Mira Ait-Saada[1,2] and **Mohamed Nadif**[1]
[1]Centre Borelli, Université Paris Cité, 75006, Paris
[2]Caisse des Dépôts et Consignations, 75013, Paris
`{mira.ait-saada,mohamed.nadif}@u-paris.fr`

Abstract

Unsupervised anomaly detection seeks to identify deviant data samples in a dataset without using labels and constitutes a challenging task, particularly when the majority class is heterogeneous. This paper addresses this topic for textual data and aims to determine whether a text sample is an outlier within a potentially multi-topic corpus. To this end, it is crucial to grasp the semantic aspects of words, particularly when dealing with short texts, since it is difficult to syntactically discriminate data samples based only on a few words. Thereby we make use of word embeddings to represent each sample by a dense vector, efficiently capturing the underlying semantics. Then, we rely on the Mixture Model approach to detect which samples deviate the most from the underlying distributions of the corpus. Experiments carried out on real datasets show the effectiveness of the proposed approach in comparison to state-of-the-art techniques both in terms of performance and time efficiency, especially when more than one topic is present in the corpus.

1 Introduction

Anomaly Detection (AD) is a task that can address various objectives such as mining frauds (Deng and Mei, 2009), diseases (Han et al., 2021) and intrusions (Pu et al., 2021). AD takes several forms: supervised, unsupervised or semi-supervised. Unsupervised AD implies that no prior information about the dataset is provided. In this case, the solution usually consists of identifying samples that deviate in a certain way from the others among the same dataset; anomalies being, by definition, rare phenomena. Particularly, anomalies in a textual dataset can be defined as samples having an atypical vocabulary (lexical anomaly) or a deviating global meaning (semantic anomaly). Identifying abnormalities in textual data can be very useful in many industrial use-cases. A good example is the detection of non-eligible and/or fraudulent course contents in the public French plateform MonCompteFormation[1] where millions of course sessions are available with no possibility of controlling training organizations in a supervised fashion (using labeled data). Hence, to assess the effectiveness of our approach, we rely on an external labeled dataset that closely relates to course contents and that is dedicated to course certifications. The dataset is described in Appendix A. In addition to the difficulty of mining anomalies in short-text corpora of varying sizes, we also have an important computational cost constraint that is also addressed by the proposed solution.

Les lignes de commande Linux pour débutants	inlier
Formation: Introduction au Shell Bash	inlier
Administration système Unix pour les nuls	inlier
Apprendre à utiliser le terminal Ubuntu/Debian	inlier
Formation en Espagnol pour débutants	outlier

Table 1: Example illustrating the importance of semantic representations in a small corpus of short texts.

Capturing the semantics of a given text is usually performed using Word Embeddings, which consist in representing a word or a piece of text by a fixed-size vector, supposed to detain its meaning. Several word embedding techniques are available such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), fastText (Bojanowski et al., 2017), BERT (Devlin et al., 2019), DistilBERT (Sanh et al., 2019), etc. Each of the mentioned works provides ready-to-use models that are pre-trained on very large corpora and intended to be general for a given language and suitable for several NLP downstream tasks. Indeed, relying on such pre-trained models has proved efficient in several tasks (Kim, 2014; Das et al., 2017) and is particularly useful when dealing with small corpora (Buechel et al., 2018). If we consider the

---

[1] `https://www.moncompteformation.gouv.fr/` developed by « Caisse des Dépôts et Consignations » (CDC)

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1392–1403
May 2-6, 2023 ©2023 Association for Computational Linguistics

small corpus given in Table 1, we can observe that the inlier samples (training titles about Linux shell programming) do not have any words in common. Thus, based only on the syntactic information of the samples, it would be impossible to isolate the outlier (about learning Spanish), even though it is the only one that does not have anything to do with shell programming. This can easily occur when dealing with short-text corpora, especially when the number of samples is not sufficient to learn the different syntactic variants of a word or a concept.

Depending on the data type and the assumptions that can be made, the definition of an outlier may differ, and the choice of the model is crucial, especially in an unsupervised context (Aggarwal, 2017). In this paper, we propose a probabilistic AD approach based on Mixture Models, that effectively identifies the most deviating samples in short-text datasets, even in the case where several topics are present in the inlier class. We also show the effectiveness of using the knowledge learned by word embedding models in capturing the underlying semantics of short texts and efficiently identifying outliers. The main contributions of this paper are:

- We address the challenge of mining anomalies in short texts written in French
- We tackle the classical one-class inlier scheme and also a more challenging multi-class inlier setting.
- We propose an effective and efficient anomaly detection approach that outperforms previously proposed AD techniques.

2 Related Work

Anomaly detection is an active research area, and a large number of approaches are proposed in several application domains. Specifically, our work relates to unsupervised AD for text and clustering-based AD. Unsupervised AD is gaining more and more interest in research due to the constant growth of data volumes while labeling data samples is not getting any cheaper. One of the most important family of methods contains reconstruction-based approaches that assume that a well-generalizing model would struggle at compressing rare anomalous samples. This kind of approaches include linear models such as Robust PCA (Kang et al., 2015) and deep autoencoder models based on convolutional networks (Oza and Patel, 2019), recurrent networks (Hsieh et al., 2019), etc.

2.1 Clustering-based AD

In the clustering-based AD approaches, anomalies are generally seen as data samples that present a lower adhesion to the underlying groups. Several two-phase approaches have been proposed and consist in using a clustering algorithm such as DBSCAN (Sheridan et al.), K-means (Deng and Mei, 2009) and Affinity Propagation (Marcos Alvarez et al., 2013), then compute an anomaly score from the obtained clustering partition. Similarly in (Mahadevan et al., 2010), to detect temporal anomalies in videos, inlier behaviors are modeled as a mixture of Gaussian distributions. A deep GMM-based approach called DAGMM is proposed in (Zong et al., 2018) to detect outliers in numerical data, where the input data are compressed into a lower-dimensional space using an autoencoder and then fed into a GMM component. The autoencoder's reconstruction loss and the log-likelihood of the GMM component are optimized jointly, without performing any pre-training phase.

2.2 AD in text data

Unlike images, time series, and numerical data, relatively few AD studies are dedicated to textual data. Document-term matrix representations (also called sparse bag-of-words) have previously been used in (Kannan et al.) to perform AD based on Nonnegative Matrix Factorisation (NMF) and isolate an outlier matrix, used to compute the anomaly scores. Sparse representations are also used by Manevitz and Yousef (2001) as input to a One-Class Support Vector Machine (OC-SVM) (Schölkopf et al., 2001) and later to a shallow autoencoder (Manevitz and Yousef, 2007). Word embeddings like word2vec are used for AD in (Zhuang et al., 2017) along with a von Mises Fisher (vMF) mixture model where more general words are penalized when computing the overall outlierness score of a given document. Pre-trained fastText word vectors are used in (Ruff et al., 2019) as the embedding layer of a multi-head attention network to perform anomaly detection as a one-class classification task. Recently, a deep end-to-end approach has been proposed by Manolache et al. (2021) that does not use any knowledge transfer. The authors use the transformer architecture of ELECTRA (Clark et al., 2020) that contains two adversarial components: a generator and a discriminator. The model is trained from scratch on a given dataset by optimizing a loss function based on token replacement.

2.3 Semantic text representations

Tremendous advances in various NLP tasks have been made in recent years thanks to dense vector representations of words and text sequences. Static word embeddings like word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and fastText (Bojanowski et al., 2017) provide one unique dense representation for each word whereas contextual word embedding models like ELMo (Peters et al., 2018), BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) provide word representations that depend on the surrounding context. Contextual word embedding models are based on deep neural networks, which makes them resource-intensive and difficult to use in some industrial contexts. Both kinds of word embeddings have proved effective in several unsupervised downstream tasks like semantic textual similarity (Arora et al., 2017; Ranasinghe et al., 2019), clustering (Ait-Saada et al., 2021; Boutalbi et al., 2022) and anomaly detection (Zhuang et al., 2017).

3 Gaussian Mixture Models

Given a corpus $\mathcal{D}$ of n short texts $(d_1, \dots, d_n)$, we represent each sample by a fixed size vector, thus obtaining a matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ of size $n \times m$. To tackle the AD problem, we postulate that the samples follow a mixture of distributions, from which the anomalous samples deviate.

Admittedly, the family of t-distributions provides a heavy-tailed alternative to the gaussian family for anomaly detection. However, as pointed out by (Yuan and Huang, 2009), although useful from modeling perspective, the practical use of multivariate t-distribution is often limited by the difficulty in parameter estimation, particularly so for high dimensional data. Note that, in our proposal, the consideration of the time consumption is important. Therefore, considering a t mixture model leads to estimate a supplementary parameter (in addition to the estimation of vector means and covariance matrices) that is the degree of freedom of each component. Moreover, since we suggest to consider an ensemble method allowing to combine results by varying the number of components (cf. Section 3.1), we would therefore increase yet the computation time for estimation of the parameters. However, Gaussian Mixture Model (GMM)-based approaches, more parsimonious than t-mixture model, have shown their effectiveness in anomaly detection, such as DAGMM (Zong et al., 2018). For these reasons we retain GMM to address our purpose.

In a finite GMM, the data $(\mathbf{x}_1, \dots, \mathbf{x}_n)$ are taken to constitute a sample of n independent instances of a random variable $\mathbf{X}$ in $\mathbb{R}^m$. Density can be expressed as:

$$f(\mathbf{x}_i; \Theta) = \sum_{k=1}^{g} \pi_k \varphi_k(\mathbf{x}_i | \mu_k, \Sigma_k), \forall i \in \{1, \dots, n\}$$

where $\Theta = (\pi_1, \dots, \pi_g, \mu_1, \dots, \mu_g, \Sigma_1, \dots, \Sigma_g)$, $\varphi_k(\mathbf{x}_i|\mu,\Sigma_k)$ is the kth component density for observation $\mathbf{x}_i$ with parameters (μ_k, Σ_k), $(\pi_1, \dots, \pi_{g-1})$ are the mixing weights or probabilities (such that $\pi_k > 0$, $\sum_{k=1}^{g} \pi_k = 1$) and g is the number of mixture components. Thus, clusters are ellipsoidal, centered at the mean vector μ_k, and with other geometric features, such as volume, shape and orientation, determined by the covariance matrix Σ_k (Banfield and Raftery, 1993; Celeux and Govaert, 1995). To estimate Θ we rely on the maximisation of the log-likelihood given by:

$$L(\mathbf{X}; \Theta) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{g} \pi_k \varphi_k(\mathbf{x}_i | \mu_k, \Sigma_k) \right).$$

The maximization is commonly performed by Expectation-Maximization (Dempster et al., 1977); an iterative algorithm based on the maximization of the conditional expectation of the complete data log-likelihood given Θ':

$$Q(\Theta|\Theta') = \sum_{i} \sum_{k} s_{ik} \log(\pi_k \varphi_k(\mathbf{x}_i | \mu_k, \Sigma_k))$$

where $s_{ik} \propto \pi_k \varphi_\ell(\mathbf{x}_i | \mu_k, \Sigma_k)$ are the posterior probabilities.

In real terms, the algorithm is broken down into two steps (E-M steps) and the unknown parameters of Θ are updated thanks to the previously computed probabilities. For each component k, we have

$$\pi_k = \frac{\sum_i s_{ik}}{n} \qquad \mu_k = \frac{\sum_i s_{ik} x_{ij}}{\sum_i s_{ik}}$$

$$\text{and } \Sigma_k = \frac{\sum_i s_{ik} (\mathbf{x}_i - \mu_k)^\top (\mathbf{x}_i - \mu_k)}{\sum_i s_{ik}}.$$

The procedure used to identify anomalies is described in Algorithm 1. It takes as input a set of short texts and returns the ones that are the most likely to constitute an anomaly. The maximum density as normality score denotes the confidence

Algorithm 1: AD with GMM

```
Input: $\mathcal{D} = \{d_1, \dots d_n\}$, $g$ the number of
       components, $\mathcal{M}$ an embedding
       module, $\alpha$ the desired number of
       output samples;
$\mathbf{x}_i \leftarrow \mathcal{M}(d_i)$, $i = 1, \dots n$;
$\mathbf{X} \leftarrow (\mathbf{x}_1, \dots, \mathbf{x}_n)$;
Initialize $\Theta'$ from a partition obtained with
$k$-means
repeat
    E-step: Compute $Q(\Theta|\Theta')$;
    M-step: Update $\pi_k$, $\mu_k$ and $\Sigma_k$;
until Convergence;
$s_i \leftarrow -\max_k(s_{ik}), k = 1, \dots, g$;
$\mathbf{s} \leftarrow (s_i, \dots, s_n)$;
$\mathbf{r} \leftarrow$argsort$(\mathbf{s})$;
return $d_j$, $j = \mathbf{r}_1, \dots, \mathbf{r}_\alpha$;
```

of the assignment. Multiplied by -1, it denotes the uncertainty of the assignment and is similar to using the entropy of $\mathbf{s}$ since $\sum_k s_{ik} = 1$. The number of returned text samples depends on the user's needs and is specified by the cutoff parameter α. In the evaluation section, we evaluate the AD performance with every possible value of α using the AUROC score.

3.1 Proposed solution for multi-class inliers

In the standard setting of AD where we consider one large inlier class, we set the number of components to its smallest possible value $g = 2$, which provides satisfactory results. In this study, we also consider the more challenging scenario where several underlying topics are present in the dataset. In this context, we make the distinction between extreme values and outliers (Aggarwal, 2017) as shown in Figure 1. A Gaussian mixture model would not have any difficulty in spotting both types of outliers since it is capable of modeling clusters of different shapes. Furthermore, we expect GMM to show good results in the multi-class context, since one of its fundamental assumptions is the multiplicity of inherent distributions among the data samples. However, this property requires to know the number of components in advance, which is not always possible in real life.

To address this issue, we propose to use GMME, an ensemble of several models, obtained with different values of g. To this end, we use Algorithm 1 with varying $g_k \in \mathcal{G}$ and combine the output scores $\mathbf{s}^{(g_k)}$ as follows:

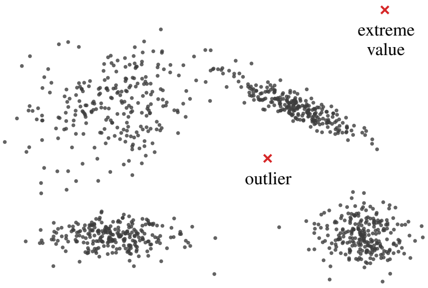


Figure 1: Difference between outlier and extreme value. This example illustrates the benefit of the clustering-based AD approaches in general and Gaussian mixture models in particular.

$$\mathbf{e}_i = -\sum_k \text{rank}(s_i^{(g_k)}).$$

The intuition behind using an ensemble approach is to make each of the models separate the dataset into clusters in a different way and assign an anomaly score according to the formed clusters. Combining those different anomaly scores leads to a more robust and meaningful overall score, even when the optimal number of clusters is not included in $\mathcal{G}$. This is corroborated by the empirical study conducted in Section 4.5.

4 Experimental Study

To assess the effectiveness of our approach and compare it to state-of-the-art, we conduct a set of experiments on real datasets and discuss the results in this section.

4.1 Datasets

We run our AD experiments on three datasets described in Table 2. MLSUM (Scialom et al., 2020) and COVID-news (Cortal, 2022) are both news datasets from which we extract the title to constitute our short-text corpora. RNCP is a dataset we built from an official French repository that lists training certifications (cf. Appendix A).

Dataset	Classes	Smallest	Largest	Medial
RNCP	16	763	9,510	3,101
COVID	9	236	3,235	1,270
MLSUM	10	2,573	26,024	13,054

Table 2: Datasets' description. The sizes correspond to the whole set of samples (training and test set).

Given a classification dataset, we first remove the classes that are too small to constitute an AD dataset, thus obtaining ℓ classes. We then derive ℓ sets of samples in which there is one inlier (majority) class and a certain rate r of outliers picked randomly from the other classes.

4.2 Experimental settings

In order to empirically evaluate our approach and compare it to other AD techniques, we rely on the Area Under the Receiver Operating Curve (AUROC), originally used as a metric in the classification task. In our case, it takes as input the anomaly scores as well as the ground truth labels and determines at what extent it is possible to accurately identify outliers using the anomaly score. It is equivalent to evaluating the performance of AD at every possible value of α in Algorithm 1. For each AD approach, we compute the AUROC on the test set over 5 different initializations, except for OC-SVM that is a deterministic model.

In our study, we discard the case where we train the model on the majority class as a one-class classification task (Ruff et al., 2019; Manolache et al., 2021; Manevitz and Yousef, 2001), for it is not a realistic scenario since one rarely has access to a large enough amount of inlier-only labeled samples in real life. Thus, we consider in this study a fully unsupervised scenario, where no labels are available and both inlier and outlier samples are present in the training set. To this end, we contaminate both sets with up to $r = 10\%$ of outliers as in (Manolache et al., 2021). The set of labels is used only during the evaluation phase.

Text representation. Given a raw corpus $\mathcal{D}$ of short texts, we first perform a minimal preprocessing that consists in removing stop words and lowercasing the input text. Then, we use a pre-trained fastText model (Bojanowski et al., 2017) to represent texts by fixed-size vectors and build the $\mathbf{X}$ matrix. The model is trained on French Wikipedia and represents each word by a vector of size $m = 300$. Using those word vectors, we represent a text sequence by the arithmetic mean of its tokens' representations as in (Ranasinghe et al., 2019; Arora et al., 2017). We show that this way of representing text sequences is well suited to shorts texts and is very beneficial in capturing text semantics for AD. One noticeable advantage of fastText is its ability to represent out-of-vocabulary words thanks to sub-word embeddings. Another advantage of fastText is that pre-trained word representations are provided in a wide range of languages[2]. We do not use contextual word embeddings to represent text sequences since they significantly increase the computational cost and do not seem to bring any performance gain in the AD task (Ruff et al., 2019).

Baselines. We compare our approach to other AD techniques: OC-SVM (Schölkopf et al., 2001), AE, DAGMM (Zong et al., 2018) and DATE (Manolache et al., 2021). For OC-SVM, AE and DAGMM, we use as input the same matrix $\mathbf{X}$ as for GMM. Concerning OC-SVM, we set $\nu = 0.05$, which is the value that presents the best results among $\{0.05, 0.1, 0.2, 0.5\}$ by far. For the autoencoder (AE) we train a model with three encoder layers and three decoder layers of size 256, 128 and 64, a learning rate of 0.001 and a weight decay of 10^{-8}. Concerning DAGMM, the authors choose the parameters relating to the architecture of the neural network according to the dataset and do not provide a method to reproduce this choice. This way of configuring the model is not suitable for the unsupervised case, in which no tuning of the hyperparameters is possible. We therefore opt for a standard architecture in decreasing powers of 2 starting from $m = 300$ (i.e. $256, 128, \ldots$). We choose $m' = 5$ as the encoding dimension because it gives the best overall performance. For DATE, we use the same parameters as used in the original paper for AG-news (Manolache et al., 2021).

Hyperparameter tuning. Our present work falls within the context of unsupervised learning, where we are not supposed to have access to ground truth labels. In this respect, it is not feasible in real life to tune the hyperparameters of a given model to fit unlabeled data, since the performance score is simply impossible to compute. We therefore consider hyperparameter tuning in this context unrealistic and possibly leading to hiding instabilities that can neither be detected nor fixed by practitioners. Hence, robustness and insensitivity to parameters are key in the unsupervised setting. For this reason, regarding the baselines, we use the recommended parameters provided in the original works when available, in order to reproduce real-world conditions. When no recommended parameters are provided, we use the ones that maximize the overall performance even though it does not play in favor of our proposed approach which does not require any parameter tuning. By doing so, we guaran-

---

[2] `https://fasttext.cc/`

	Inlier	OC-SVM	AE	DAGMM	DATE	GMM
RNCP	enviro.	62.2	62.1	52.8	65.8	**68.4**
	défens.	58.0	46.2	52.5	62.2	**74.3**
	intell.	66.7	65.4	59.1	79.0	**80.1**
	recher.	68.6	64.5	58.0	75.3	**77.7**
	nautis.	64.9	62.9	53.0	68.2	**73.1**
	aérono.	63.7	59.4	49.8	76.1	**78.0**
	sécuri.	59.8	54.7	51.3	75.0	**80.0**
	multim.	57.8	56.3	51.3	64.5	**71.2**
	humani.	56.7	59.6	53.4	69.0	**72.1**
	nucléa.	65.1	58.7	59.9	74.9	**75.1**
	enfanc.	61.0	60.1	65.0	72.1	**78.9**
	saison.	50.5	47.5	50.8	48.8	**74.9**
	assist.	45.1	50.7	56.4	52.5	**63.2**
	sport	51.1	56.1	46.8	66.3	**73.3**
	ingéni.	66.9	60.1	57.2	**75.3**	74.2
	sans d.	45.4	36.9	53.7	39.3	**59.9**
COVID	cultur.	49.7	40.1	43.4	40.3	**53.7**
	enviro.	56.9	55.0	50.2	55.2	**66.2**
	intern.	62.0	53.7	53.1	62.4	**65.1**
	people.	63.9	47.4	49.7	51.8	**64.0**
	politi.	67.2	60.1	51.6	68.1	**79.2**
	scienc.	55.5	38.3	60.6	64.5	**66.6**
	sociét.	48.9	47.5	55.3	**57.0**	55.1
	sport	68.5	35.6	55.4	52.6	**69.0**
	économ.	52.7	49.4	50.7	54.0	**59.4**
MLSUM	afriqu.	70.0	50.5	62.0	74.3	**75.5**
	police.	78.3	74.4	52.9	77.0	**82.9**
	politi.	70.3	60.3	49.6	74.6	**75.7**
	scienc.	53.1	35.5	49.6	**71.0**	55.2
	societ.	73.0	73.0	48.9	75.4	**79.0**
	sante	78.7	56.2	55.2	86.5	**87.5**
	argent.	47.7	36.4	51.7	**82.2**	63.4
	livres.	**71.2**	65.8	48.4	58.1	65.9
	cultur.	**59.5**	49.8	52.4	45.8	51.4
	sport	74.9	56.5	45.2	69.2	**81.7**

Table 3: AUC scores obtained with anomaly $r = 10\%$. The **bold** numbers correspond to the best score in each row and the underlined numbers are for the second best performance score.

tee a fair evaluation of our proposal and show its robustness in an unsupervised context.

4.3 Results with one-class inliers

The obtained performance is given in Table 3 with an anomaly rate $r = 10\%$. We first observe the effectiveness of GMM on the three datasets, in comparison to all of the baselines, offering the best AUROC in most of the cases. Furthermore, the state-of-the-art DATE shows its limits on short texts and presents competitive but poorer results in comparison to GMM. OC-SVM is competitive on short texts in comparison to DATE but presents poorer overall performance. AE is the model that provides the lowest AUROC values, right after DAGMM. This might be due to the fact that the encoding (or embedding) step of the encoder (for both AE and DAGMM) is performed beforehand using pre-trained word embeddings and becomes pointless when applied to this kind of representation.

The results obtained with different values of contamination rate r are summarized by Critical Difference (CD) diagrams in Figure 2. The aim of CD diagrams (Demšar, 2006) is to visualize the performance ranks of each approach over the different datasets. If we take the example of $r = 10\%$, the CD diagram summarizes the scores given in Table 3. It depicts the average rank of each method and the bold line corresponds to the critical difference, based on the post-hoc Nemenyi test (Nemenyi, 1963). Note that the presence of a higher number of outliers increases the difficulty of the AD task and decreases the overall performance scores. We can observe that the more we inject anomalies in the training set, the more GMM gets competitive against the other approaches in terms of AUROC score. This shows GMM's robustness to outliers and its generalization capabilities while other techniques tend to overfit in the presence of noise in the training set.

Also, it is worth noting that GMME yields similar results in comparison to GMM, which makes it a universal solution, well suited to AD even in the one-class scenario.

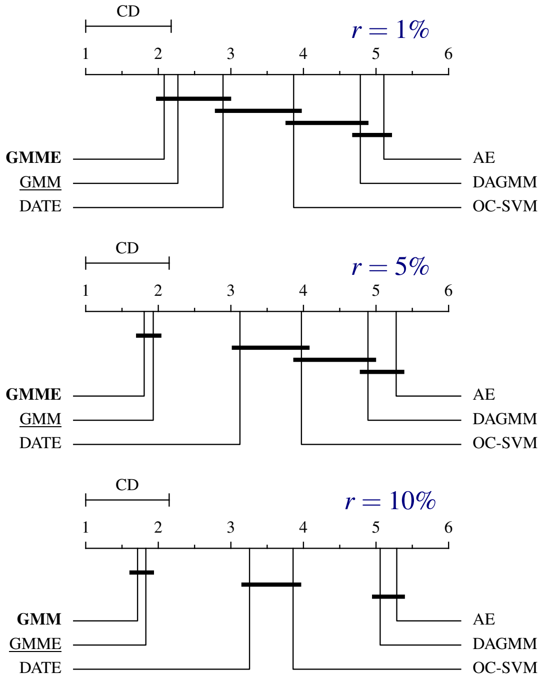


Figure 2: CD plots from the Nemenyi test over different datasets. This graphic summarizes the rank of each approach with different contamination rates r.

Comparison with VMF-Q. To make a fair comparison between GMM and VMF-Q (Zhuang et al., 2017), we reproduce the exact same setting for the two approaches. VMF-Q is based on the von Mises-Fisher distribution that relies on a Bessel function to estimate the parameter κ. The model is originally trained using embeddings of size $m = 200$, but encounters numerical difficulties with higher dimensions, due to the approximations made by the Bessel function that depends, inter alia, on m. We hence use, for both GMM and VMF-Q, another pre-trained model of size $m = 200$, that is provided by Fauconnier (2015). The gain of performance from VMF-Q to GMM is summarized in Figure 3. We can observe a clear advantage of GMM in comparison to VMF-Q on the three datasets, especially on MLSUM, where GMM outperforms VMF-Q on all the subsets. We also report poorer overall results using word2vec with $m = 200$ in comparison to the fastText model we use in the rest of our experiments.

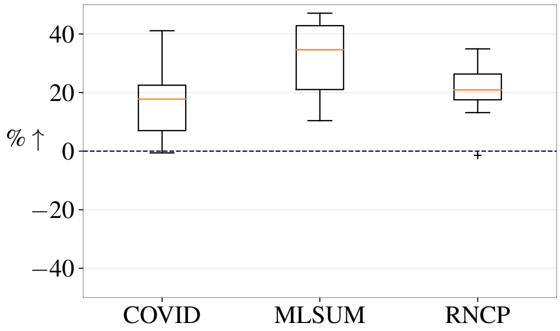


Figure 3: Gain of performance between GMM and VMF-Q using word2vec embeddings of size $m = 200$. Positive values give an advantage to GMM.

4.4 Performance and data size

Figure 4 shows the gain of performance from baselines to GMM according to the size of the datasets. We see that the percentage of improvement is greater on small datasets but remains positive on large datasets. Note that detecting anomalies can be trickier on small datasets, especially when dealing with short texts (cf. Table 1) which makes GMM a good solution to tackle this difficulty.

4.5 Multi-class inliers

We investigated in the previous sections the detection of semantic anomalies in a dataset classically composed of one unique class. In this section, we consider a dataset with several underlying topics and identify the samples that do not belong to any of them. To evaluate our approach in such a context, we create datasets as described in Section 4.1 but this time, by combining $\hat{g}$ random classes to form one inlier class then inject anomalous samples with a rate $r = 10\%$. We then proceed similarly to identify anomalies and validate the obtained results. Note that we make sure in our experiments to put aside enough “anomalous“ classes so that we have a sufficient diversity of anomalies and avoid forming an additional cluster with the anomalous samples. To this end, we limit $\hat{g}$ according to the available classes in the dataset. We use the obtained datasets to assess the performance of GMME (cf. Section 3) and compare it to GMM and two other baselines: DBSCAN and DATE. DBSCAN natively deals with outliers and considers the samples that cannot be assigned to any existing cluster as anomalies (they are assigned the -1 label). We use this information to determine whether a text sample is an anomaly. We set the parameters $\epsilon = 1$ and min_samples $= 3$ which yielded better results than the default values available in the scikit-learn implementation.

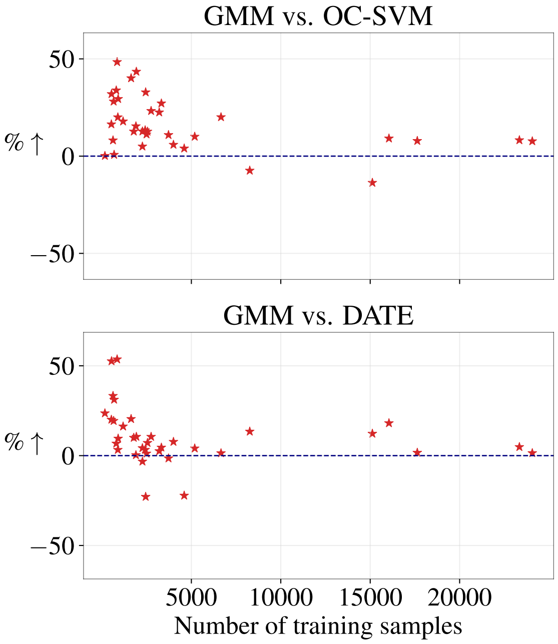


Figure 4: Percentage of improvement of GMM in comparison to baselines w.r.t. to the train set’s size. Positive values give an advantage to GMM.

Table 4 shows the performance obtained by GMME with values of $g_k \in \mathcal{G} = \{2, 3, 4\}$. We first observe that the ensemble approach further improves the performance of the simple GMM, which still performs better in comparison to the other baselines. DBSCAN presents the poorest results after

# inlier classes		DBSCAN	DATE	GMM	GMME
RNCP	2	63.4	75.6	88.6	**89.2**
	3	62.8	70.5	82.4	**83.9**
	4	64.6	77.6	80.2	**82.0**
	5	65.7	63.5	**81.6**	80.7
	6	61.7	56.7	**65.5**	64.5
	7	58.4	48.8	60.6	**62.6**
	8	60.4	56.8	63.2	**63.4**
COVID	2	51.5	59.3	**65.2**	63.8
	3	50.6	51.3	54.1	**54.2**
	4	50.7	53.5	56.2	**56.3**
	5	50.0	51.1	52.6	**52.7**
MLSUM	2	50.5	70.0	74.2	**74.6**
	3	50.2	49.6	**55.1**	54.4
	4	50.6	47.0	55.0	**55.5**
	5	50.7	45.7	53.1	**53.8**

Table 4: Comparison of DBSCAN, DATE, GMM and GMME approaches: AUROC scores obtained with multiple classes as inliers.

DATE, which is less competitive in the multi-class context, especially on the RNCP dataset. Thus, GMME is the most competitive approach in the multi-topic scenario, even when the real number of clusters is not included in $\mathcal{G}$.

4.6 Computation time analysis

Table 5 contains the execution time of both the training and evaluation steps. It is estimated over three different runs, on the three-class datasets used in Section 4.5. The sizes of the training sets are 1034, 6342, 46253 and the test sets are of size 444, 2718, 5837 for RNCP, COVID-news and MLSUM respectively. The experiments on DATE are performed on an NVIDIA RTX2070 GPU.

We first notice that GMM is the fastest approach both during the training and evaluation phase. It is followed by GMME that is relatively quick, especially during evaluation. GMM scales better with an increasing number of samples in comparison to OC-SVM that takes more than four times as much time to train. DATE is the approach that takes the most time to train and evaluate, which is due to its deep architecture. The computational time of DATE can be partially amortized with a more powerful GPU but can still represent an impediment, especially in an industrial context.

We also observe that the computation time of $\mathbf{X}$ does not depend much on the size of the dataset, simply because the task that takes the longest is loading the model from memory. Hence, this way of representing text scales well, especially when used along with GMM or GMME.

		1-SVM	GMM	GMME	DATE
RNCP	$\mathbf{X}$	3.7			-
	train	3e-01	3e-01	1.2	78.49
	eval	1e-01	3e-03	3e-02	2.0
COVID	$\mathbf{X}$	3.7			-
	train	5.7	3.5	16.2	512.97
	eval	1.8	2e-02	8e-02	14.1
MLSUM	$\mathbf{X}$	3.9			-
	train	450.6	10.0	94.8	3639
	eval	33.0	4e-02	2e-01	30.2

Table 5: Execution time in seconds. The row $\mathbf{X}$ corresponds to the computation time of the embedding matrix $\mathbf{X}$ using fastText. “train“ stands for the training time and “eval“ for the evaluation time.

4.7 Improve the results with Transformers

In the previous experiments, we relied on fastText static embeddings in order to show that is it possible to perform effective and fast anomaly detection without having access to important computational resources. In this section, we show that it is possible to achieve even better results using Transformer representations. For our purpose, we use CamemBERT (Martin et al., 2020) and FlauBERT (Le et al., 2020), both trained on French corpora.

CamemBERT and FlauBERT are both based on the RoBERTa variant of Transformer language models (Liu et al., 2019). While FlauBERT uses the exact same objective and tokenization process as in Liu et al. (2019), plus a French-specific preprocessing, CamemBERT makes use of the SentencePiece tokenizer and the whole-word masking strategy that consists in masking words instead of sub-word tokens. Another major difference between the two models is the training set of data. CamemBERT uses the French part of OSCAR, a large multilingual corpus extracted from Common Crawl, while FlauBERT is trained on a set of 24 corpora.

Table 6 contains the results obtained using several input representations. We recall the results obtained by DATE for comparison purposes since it is also based on a Transformer architecture. We first observe the significant difference in performance between CamemBERT and FlauBERT with a clear advantage for CamemBERT in both its base and large versions. The large variant of Camem-

Dataset	# Inlier classes	DATE	GMME					
			fastText	flauB$_6$	flauB$_{12}$	flauB$_{24}$	camB$_{12}$	camB$_{24}$
RNCP	2	75.58	**89.22**	87.02	59.78	71.38	83.82	88.44
	3	70.48	**83.90**	76.10	64.80	63.38	76.98	82.14
	4	77.58	82.00	79.14	70.82	66.06	80.82	**84.82**
	5	63.50	80.68	69.76	47.78	69.80	77.60	**83.54**
	6	56.74	64.46	60.48	53.40	60.48	64.60	**66.64**
	7	48.78	62.60	64.68	67.24	66.42	67.84	**68.66**
	8	56.82	63.44	53.18	47.28	59.92	65.16	**67.00**
COVID	2	59.28	63.82	62.34	56.38	56.56	**65.82**	63.32
	3	51.32	54.20	54.10	47.28	54.86	**56.00**	55.48
	4	53.46	56.30	55.40	49.70	51.80	58.00	**58.44**
	5	51.08	52.70	**56.28**	49.02	50.48	52.90	52.12
MLSUM	2	70.00	74.60	68.80	61.00	55.00	79.60	**85.10**
	3	49.60	54.44	59.70	47.80	56.50	66.44	**72.74**
	4	47.02	55.52	58.30	39.00	54.70	65.02	**70.18**
	5	45.70	53.84	58.80	42.30	56.02	65.44	**71.08**

Table 6: Comparison between different embedding representations in terms of AUC score with anomaly rate $r = 10\%$. The **bold** numbers correspond to the best score in each row and the underlined numbers are for the second-best performance score. flauB and camB stand respectively for FlauBERT and CamemBERT and the subscript is for the number of layers, which indicates the model size (6 for small, 12 for base and 24 for large).

BERT achieves the best overall results, especially on MLSUM. CamemBERT-base yields competitive results, achieving good AUROC scores on the COVID dataset. We also notice that, in all cases, the two-phased approach that consists in computing Transformer representations and then use an ensemble of model-based clustering models (GMME) is more efficient than the end-to-end Transformer-based approach (DATE). It is worth noting that fastText embeddings remain competitive and surpass the three FlauBERT models. It is therefore a very good alternative in real-time use cases when computational speed is a critical issue.

5 Conclusion

This paper addresses semantic anomaly detection in short texts with an additional constraint in time efficiency. In addition to the classical framework where one class is used as the inlier class, we also consider the scenario where several underlying subgroups are present in the normal class. We see anomaly detection as a probabilistic clustering problem, in which we learn a Gaussian mixture model and consider the low posterior probability samples as belonging to none of the modeled clusters and more likely to constitute outliers. This uncertainty score proved effective with different numbers of subgroups. In the multi-class setting, we propose GMME, an ensemble approach that improves the performance of GMM when several topics are present in the inlier class. The two approaches outperform state-of-the-art anomaly detection techniques in both scenarios, with an impressively low computation time.

In our proposal, we rely on the Gaussian Mixture model for its flexibility. This choice is motivated by the presence of the proportions π_k of each cluster and the spectral decomposition of the covariance matrix Σ_k taking into account the volume, shape, and orientation of each cluster (as depicted in Figure 1). The characteristics of the clusters should not be overlooked when tackling the problem of anomaly detection through a clustering approach. Furthermore, note that our approach can be extended to latent block models, devoted to co-clustering, which may constitute an interesting and promising future path of research.

6 Limitations

This paper deals exclusively with French text corpora to answer a specific industrial need. However, we are confident about the fact that this work can easily be extended to other languages, especially English, for which more data and pre-trained models are available. This would constitute an interesting trajectory for future work.

References

Charu C. Aggarwal. 2017. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer.

Mira Ait-Saada, François Role, and Mohamed Nadif. 2021. How to leverage a multi-layered transformer language model for text clustering: An ensemble approach. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 2837–2841.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations*.

Jeffrey D. Banfield and Adrian E. Raftery. 1993. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49(3):803–821.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rafika Boutalbi, Mira Ait-Saada, Anastasiia Iurshina, Steffen Staab, and Mohamed Nadif. 2022. Tensor-based graph modularity for text data clustering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2227–2231.

Sven Buechel, João Sedoc, H. Andrew Schwartz, and Lyle H. Ungar. 2018. Learning neural emotion analysis from 100 observations: The surprising effectiveness of pre-trained word representations. *CoRR*, abs/1810.10949.

Gilles Celeux and Gérard Govaert. 1995. Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5):781–793.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Gustave Cortal. 2022. Covid-19 - french news dataset.

Arjun Das, Debasis Ganguly, and Utpal Garain. 2017. Named entity recognition with word embeddings and wikipedia categories for a low-resource language. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(3).

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30.

Qingshan Deng and Guoping Mei. 2009. Combining self-organizing map and k-means clustering for detecting fraudulent financial statements. In *2009 IEEE International Conference on Granular Computing*, pages 126–131.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 4171–4186.

Jean-Philippe Fauconnier. 2015. French word embeddings.

Changhee Han, Leonardo Rundo, Kohei Murao, Tomoyuki Noguchi, Yuki Shimahara, Zoltán Ádám Milacski, Saori Koshino, Evis Sala, Hideki Nakayama, and Shin'ichi Satoh. 2021. Madgan: Unsupervised medical anomaly detection gan using multiple adjacent brain mri slice reconstruction. *BMC bioinformatics*, 22(2):1–20.

Ruei-Jie Hsieh, Jerry Chou, and Chih-Hsiang Ho. 2019. Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing. In *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*, pages 90–97.

Zhao Kang, Chong Peng, and Qiang Cheng. 2015. Robust pca via nonconvex rank approximation. In *2015 IEEE International Conference on Data Mining*, pages 211–220.

Ramakrishnan Kannan, Hyenkyun Woo, Charu C. Aggarwal, and Haesun Park. *Outlier Detection for Text Data*, pages 489–497.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos. 2010. Anomaly detection in crowded scenes. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1975–1981.

Larry Manevitz and Malik Yousef. 2007. One-class document classification via neural networks. *Neurocomputing*, 70(7):1466–1481. Advances in Computational Intelligence and Learning.

Larry M Manevitz and Malik Yousef. 2001. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154.

Andrei Manolache, Florin Brad, and Elena Burceanu. 2021. DATE: Detecting anomalies in text via self-supervision of transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 267–277.

Alejandro Marcos Alvarez, Makoto Yamada, Akisato Kimura, and Tomoharu Iwata. 2013. Clustering-based anomaly detection in multi-view data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, page 1545–1548.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26.

Peter Bjorn Nemenyi. 1963. *Distribution-free multiple comparisons*. Princeton University.

Poojan Oza and Vishal M. Patel. 2019. One-class convolutional neural network. *IEEE Signal Processing Letters*, 26(2):277–281.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Guo Pu, Lijuan Wang, Jun Shen, and Fang Dong. 2021. A hybrid unsupervised clustering-based anomaly detection method. *Tsinghua Science and Technology*, 26(2):146–153.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2019. Enhancing unsupervised sentence similarity methods with deep contextualised word representations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 994–1003, Varna, Bulgaria. INCOMA Ltd.

Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. 2019. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4061–4071.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. 2001. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7):1443–1471.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. MLSUM: The multilingual summarization corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8051–8067.

Kevin Sheridan, Tejas G. Puranik, Eugene Mangortey, Olivia J. Pinon-Fischer, Michelle Kirby, and Dimitri N. Mavris. *An Application of DBSCAN Clustering for Flight Anomaly Detection During the Approach Phase*.

Ming Yuan and Jianhua Z. Huang. 2009. Regularized parameter estimation of high dimensional t distribution. *Journal of Statistical Planning and Inference*, 139(7):2284–2292.

Honglei Zhuang, Chi Wang, Fangbo Tao, Lance Kaplan, and Jiawei Han. 2017. Identifying semantically deviating outlier documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2748–2757.

Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*.

A Construction of the RNCP dataset

The RNCP[3] dataset is composed of certification contents that are provided by an organization named « *France-Compétences* ». All the data used here are publicly available in `https://www.data.gouv.fr/fr/datasets/`.

In order to label data we first get the ROME codes of each certification that correspond to the

---

[3] Répertoire National des Certifications Professionnelles

professions related to the certification. Each ROME code is assigned to a topic in the file « Arborescence thématique » of the same repository. We use the ROME code as intermediate to have the topic (*thématique*) of each certification. We hence obtain a multi-label classification dataset.

Given a certification c_i, let $\mathcal{T}_i$ be the set of its corresponding topics. In the one-class setting with inlier class h, the label of c_i is established as follows:

$$y_i = \begin{cases} 0 & \text{if } h \in \mathcal{T}_i \\ 1 & \text{otherwise.} \end{cases}$$

In the multi-class setting with inlier classes symbolized by $\mathcal{H}$, the anomaly labels are defined as:

$$y_i = \begin{cases} 0 & \text{if } \mathcal{H} \subset \mathcal{T}_i \\ 1 & \text{if } \mathcal{H} \cap \mathcal{T}_i = \emptyset \\ \text{N/A} & \text{otherwise} \end{cases}$$

where $y_i = 1$ means c_i is categorized as an anomaly.

B AD examples with RNCP

Table 7 presents some examples of anomalies predicted on two subsets of the RNCP dataset: « *aéronotique* » (meaning aeronautics) and « *nucléaire* » (meaning nuclear). The *aéronautique* test set contains 1585 samples, 154 of which are labeled as anomalies, and the *nucléaire* set contains 1431 samples including 131 anomalies. In both cases we set α to 250 (cf. Algorithm 1). We observe in both cases that DATE has more difficulty in detecting anomalous text sequences when they are very short. For example, in the *nucléaire* set, the certification *Livreur* (meaning delivery person) does not have anything to do with the nuclear field. Yet DATE does not place it among the 250 most deviant samples and makes it the 355th anomalous sample while it is only 3rd according to GMM. This might be explained by the fact that DATE is trained from scratch and does not benefit from the semantic knowledge inherited by transfer learning.

Subset	Certification	GMM	DATE	Real
aéronotique	Sciences, Technologies, Santé - Mention : Automatique et informatique industrielle - Spécialité : Automatismes industriels	Inlier	Inlier	Inlier
	Production industrielle option ingénierie des matériaux nouveaux	Inlier	Inlier	Inlier
	Actuaire	Outlier	Inlier	Outlier
	Sciences Politiques	Outlier	Inlier	Outlier
	CQP Animateur de patinoire option hockey sur glace	Outlier	Outlier	Outlier
	Décor architectural opt. B Domaine du décor du mur	Outlier	Outlier	Outlier
nucléaire	Culture et communication Mention : Création, innovation, information numériques Spécialité : Gestion de l'information et du document Domaine : Culture et communication	Inlier	Inlier	Inlier
	Responsable d'ingénierie des systèmes d'information et de communication, option "analyse et développement", option "systèmes et réseaux" et option "télécommunications"	Inlier	Inlier	Inlier
	Livreur	Outlier	Inlier	Outlier
	Architecte d'intérieur	Outlier	Inlier	Outlier
	Responsable conception, mise en place et maintenance des installations frigorifiques et climatiques	Outlier	Outlier	Outlier
	Urbanisme et Aménagement Spécialité DYATER (Dynamiques et Aménagement des espaces, Territorialités)	Outlier	Outlier	Outlier

Table 7: Examples illustrating the difference of prediction between GMM and DATE according to the length of the text sequence, with $\alpha = 250$.

Metaphor Detection with Effective Context Denoising

Shun Wang[1]*, Yucheng Li[2]*, Chenghua Lin[1]†, Loïc Barrault[1,3] , Frank Guerin[2]
[1] Department of Computer Science, University of Sheffield, UK
[2] Department of Computer Science, University of Surrey, UK
[3] Meta AI
`{swang209, c.lin}@sheffield.ac.uk`
`{yucheng.li, f.guerin}@surrey.ac.uk`
`loicbarrault@meta.com`

Abstract

We propose a novel RoBERTa-based model, RoPPT, which introduces a target-oriented parse tree structure in metaphor detection. Compared to existing models, RoPPT focuses on semantically relevant information and achieves the state-of-the-art on several main metaphor datasets. We also compare our approach against several popular denoising and pruning methods, demonstrating the effectiveness of our approach in context denoising. Our code and dataset can be found at `https://github.com/MajiBear000/RoPPT`.

1 Introduction

Metaphor is a pervasive linguistic device, which attracts attention from both the fields of psycholinguistics and computational linguistics due to the key role it plays in the cognitive and communicative functions of language (Wilks, 1978; Lakoff and Johnson, 1980; Lakoff, 1993). Linguistically, metaphor is defined as a figurative expression that uses one or several words to represent another concept given the context, rather than taking the literal meaning of the expression (Fass, 1991). For instance, in the sentence "*This project is such a <u>headache</u>!*", the contextual meaning of *headache* is "a thing or person that causes worry or trouble", different from its literal meaning, "a continuous pain in the head".[1]

Metaphor detection is challenging, as it requires understanding the nuanced relationships between abstract concepts embodied by the metaphoric expression and its surrounding context. Recent studies on this direction show its potential in benefiting a wide range of NLP applications, including sentiment analysis (Li et al., 2022a), metaphor generation (Li et al., 2022b,c) and mental healthcare (Abd Yusof et al., 2017; Gutiérrez et al., 2017). When modelling relevant context for metaphor detection, various strategies have been proposed. These range from using highly restricted forms of linguistic context such as subject-verb and verb-direct object word pairs (Gutiérrez et al., 2016), to a wider context accounting for a fixed window surrounding the target word (Do Dinh and Gurevych, 2016; Mao et al., 2018), and modelling the full sentential context (Gao et al., 2018; Choi et al., 2021). While it has been argued that modelling a wider context is beneficial (Cheng et al., 2021), it has also been noted that a wider context is likely to introduce noise into the representations, and hence hinder model's performance in metaphor detection (Le et al., 2020).

Some recent efforts (Le et al., 2020; Song et al., 2021a) attempt to improve context modelling by explicitly leveraging the syntactic structure (e.g., dependency parse tree) of a sentence in order to capture important context words, where the parse trees are typically encoded with graph convolutional neural networks. MelBERT (Choi et al., 2021) employs a simple chunking method which separates sub-sentences by commas. The sub-sentence that contains a target word is then marked with a special token type, signalling its contextual importance to the target. However, these strategies are either difficult to apply to batch optimisation due to their tree-dependent encoding process, or have limited effectiveness for context denosing. For instance, the simple chunking mechanism misses the syntactic structure, and thus can neither determine the degree of importance of context words, nor connect information across different subsentences.

In this paper, we propose a novel metaphor detection model RoPPT: RoBERTa with Pruning on target-oriented Parse Tree. RoPPT introduces a *flat*, *target-oriented* tree structure by reshaping and pruning the ordinary parse trees to extract semantically relevant neighbours of a target word. The resulting tree representation allows the model to

---

* The two authors contributed equally to this work.
† Corresponding author
[1] `https://www.oxfordlearnersdictionaries.com`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1404–1409

focus on syntactically relevant information of a target word, and ignore irrelevant parts despite their position. It thus retains more relevant context for metaphor detection.

Extensive experiments conducted on three public benchmark datasets (i.e., VUA, MOH-X, TroFi) show that RoPPT can significantly improve metaphor detection on all datasets against several popular denoising and pruning methods. Our model also yields better or comparable performance to the state-of-the-art models (Choi et al., 2021; Song et al., 2021a) in Micro F1 measure. To further validate our approach, we conducted an additional investigation to assess the effect of sentence length on the performance of our model. Experimental results demonstrate a positive correlation between the increase in the performance of RoPPT and the length of the input sentence.

In summary, our paper makes three contributions: (1) we propose a flat, target word-oriented tree structure by reshaping and pruning the ordinary parse trees to retain the most relevant context for a target word; (2) we propose RoPPT, a RoBERTa-based model which can effectively encode the target-oriented parse tree for metaphor detection, achieving state-of-the-art results on three bench mark datasets; (3) we compare and evaluate a range of context denoising methods for metaphor detection, demonstrating the effectiveness of our proposed tree structure in context denoising.

2 Method

The overall architecture of RoPPT is shown in Figure 1, which can be divided into two parts: a target-oriented parse tree pruning module and a RoBERTa (Liu et al., 2019) contextual encoder.

2.1 Target-oriented Dependency Parse Tree

Connecting target words with their most relevant context words is crucial for metaphor detection and comprehension. While there have been attempts to employ dependency parse trees in graph convolutional neural networks to improve context modelling (Wang et al., 2020), it raises challenges of how to effectively encode and leverage such syntactic structure information for transformer-based mask language models for metaphor detection.

We tackle this challenge by introducing a target-oriented parse tree generated by three steps: **1**) reshape the original parse tree from existing parsers such as spaCy (Honnibal and Montani, 2017) and Biaffine (Dozat and Manning, 2016); **2**) root the tree at the target word; **3**) prune the tree according to the distance between leaves and root, coined as *neighbor range*. The rationale behind is that the target word is the focus of the task rather than the original root. So the re-rooting allows us to focus on the connections between target words and their relevant context. The resulting flat, target-oriented tree structure also enables simple encoding process into the model. Figure 1 shows an example of our reshaped tree, which retrains words with neighbor range $con = 1$ to the root 'bogged'.

2.2 RoBERTa-based Context Encoder

We employ two metaphor identification theories in our model, i.e., Metaphor Identification Procedure (Steen, 2010, MIP) and Selectional Preference Violation (Wilks, 1978, SPV). In MIP, a metaphor is detected when there is a contrast between target word's contextual and literal meanings, whereas in SPV a metaphorical word is identified by the semantic difference from its surrounding words. Therefore, we model three types of semantic representations for implementing MIP and SPV, i.e., the literal meaning and the contextual meaning of a target word, and the context meaning.

Formally, given a sentence $S = (w_0, ..., w_n)$, we first employ the RoBERTa network to produce representations for each word.

$$H = \text{RoBERTa\_Enc}(\text{emb}_{\text{cls}}, ..., \text{emb}_n) \quad (1)$$

Here CLS is a special token indicating the start of an input, $H = (h_{\text{cls}}, h_0, ..., h_n)$ the output hidden states, and emb_i the input embedding for word w_i. Specifically, $\text{emb}_i = \text{emb}_w + \text{emb}_{pos}$, where emb_w is the word embedding, and emb_{pos} the position encoding.

Context denoising with the target-oriented parse tree. When modelling sentence representation, existing works directly employed the CLS embedding as a common practice (Choi et al., 2021; Song et al., 2021b). In contrast, RoPPT employs the target-oriented parse tree to retain the most relevant context for a target word when computing the sentence embedding. Specifically, our sentence embedding is computed as follows.

$$v_S = \frac{1}{n} \sum h_i, \ i \in \mathcal{C}_n \quad (2)$$

Here v_S is the sentence representation; $\mathcal{C}_n$ represents the n neighbour words within the neighbour

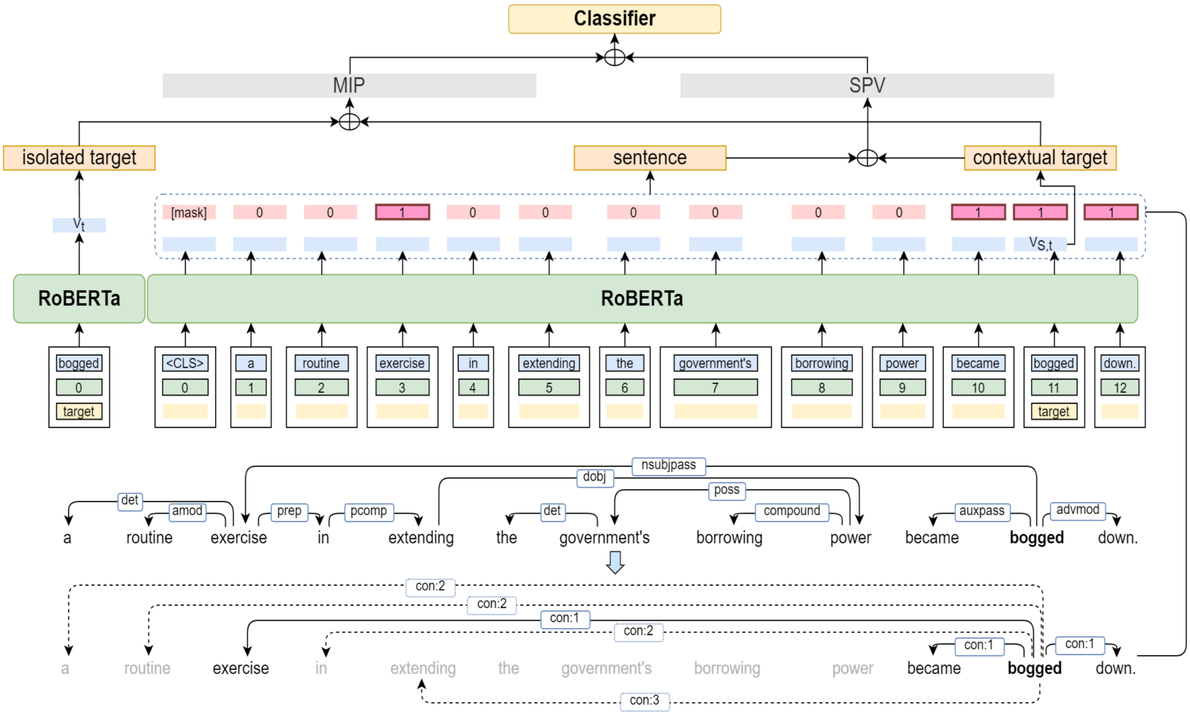


Figure 1: The overall framework of RoPPT. The parse tree of a sentence is reshaped to a target-oriented tree, and the context is pruned with a pre-set threshold. The sentence embedding is the average pooling result of hidden states for pruned context from RoBERTa. ⊕ denotes concatenation.

range of the target-oriented parse tree, and h_i is the hidden state of w_i. In other words, we do average pooling on the most relevant context words as the sentence representation and ignore other words in the sentence. We also design an alternative strategy by directly masking the original input sentence to the encoder according to the pruned parse tree. We denote this intuitive solution as RoPPT with Input Mask (**RoPPT_IM**) and discuss the performance difference between these two variants in §4.

Similar to Choi et al. (2021), we use the hidden state of target word w_t as the contextual target word embedding (i.e. $v_{S,t} = h_t$), and the literal target word embedding v_t is obtained by feeding the single target word w_t to the RoBERTa network.

$$v_t = \text{RoBERTa\_Enc}(\text{emb}_t) \tag{3}$$

We then model SPV (h_{SPV}) by concatenating the sentence embedding v_S and contextual target embedding $v_{S,t}$, and MIP (h_{MIP}) by concatenating the contextual and literal target embeddings v_t, followed by a MLP layer (i.e. $f_1(\cdot)$ and $f_2(\cdot)$).

$$h_{\text{SPV}} = f_1([v_S, v_{S,t}]) \tag{4}$$

$$h_{\text{MIP}} = f_2([v_{S,t}, v_t]) \tag{5}$$

Finally, we combine two hidden vectors h_{MIP} and h_{SPV} to compute a prediction score $\hat{y}$, and use binary cross entropy loss to train the overall framework for metaphor prediction.

$$\hat{y} = \sigma(W^\top[h_{\text{MIP}}; h_{\text{SPV}}] + b) \tag{6}$$

$$\mathcal{L} = -\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)] \tag{7}$$

3 Experimental Setup

Dataset. We conduct experiments on four public benchmark datasets. **VUA-18** (Leong et al., 2018) and **VUA-20** (Leong et al., 2020) are the largest available datasets, released in the metaphor detection shared tasks in 2018 and 2020. VUA-20 extends VUA-18 with about 12K sentences for training set and 3.6K sentences for test and validation sets. The **MOH-X** dataset is constructed by sampling sentences from WordNet (Miller, 1998). Only a single target verb in each sentence is annotated. The average sentence length is 8 tokens, the shortest of our three datasets. **TroFi** (Birke and Sarkar, 2006) consists of sentences from the 1987-89 Wall Street Journal Corpus (Charniak et al., 2000), with an average length of 28.3 tokens per sentence.

Baselines. **RoBERTa_SEQ** (Leong et al., 2020) is a fine-tuned RoBERTa sequence labeling model for metaphor detection. **MelBERT** (Choi et al., 2021)

Model	VUA18			VUA20		
	Prec	Rec	F1	Prec	Rec	F1
RNN_ELMo	71.6	73.6	72.6	-	-	-
RoBERTa_SEQ	80.1	74.4	77.1	75.1	67.1	70.9
MrBERT	82.7	72.5	77.2	-	-	-
MelBERT*	79.6	76.4	77.9	76.3	68.6	72.2
MelBERT	80.1	76.9	78.5	75.9	69.0	72.3
RoBERTa_tree	78.9	76.1	77.4	74.8	68.6	71.6
RoChunk	76.6	80.0	78.2	73.9	70.0	71.9
RoWindow	78.0	78.1	78.0	75.0	68.8	71.8
RoPPT_IM	73.4	74.3	73.9	67.7	66.8	67.2
RoPPT	80.0	78.2	**79.1**	75.9	70.0	**72.8**

Table 1: Performance comparison on **VUA** dataset (best is in **bold**). NB: * indicates the reproduced results of MelBERT using the original source code and setting of (Choi et al., 2021). RNN_ELMo and MrBERT have no results on VUA20 in their original paper. Popular denoising methods are also compared. **RoChunk** means chunk sentence by comma on RoBERTa input, **RoWindow** means denoising by a context window (size=4). **RoPPT_IM** represent masking sentence before input to transformer encoder.

Models	TroFi			MOH-X		
	Prec	Rec	F1	Prec	Rec	F1
RoBERTa_SEQ	53.6	70.1	60.7	80.6	77.7	78.7
DeepMet	53.7	72.9	61.7	79.9	76.5	77.9
MrBERT	53.8	75.0	62.7	75.9	84.1	79.8
MelBERT*	53.1	73.2	61.6	78.0	79.5	78.8
MelBERT	53.4	74.1	62.0	79.3	79.7	79.2
RoBERTa_tree	50.3	77.8	61.1	76.9	83.5	79.3
RoPPT	54.2	76.2	**63.3**	77.0	83.5	**80.1**

Table 2: Performance comparison on TroFi and MOH-X datasets (NB: **bold** denotes the best result).

realises MIP and SPV theories via a RoBERTa based model. **MrBERT** (Song et al., 2021b) is the recent SOTA on verb metaphor detection based on BERT with verb relations encoded.

Hyperparameter. We set the hyperparameter neighbour range $con = 4$ based on the validation set results. All the parser results are based on spaCy as it performs better than Biaffine empirically (see §4 for more discussion).

4 Experimental Results

Overall results. Table 1 shows a comparison of the performance of our models against the baseline models on VUA18 and VUA20, respectively. It is clear that our RoPPT outperforms all baselines on VUA18 and VUA20, including the state-of-the-art model MelBERT. A two-tailed t-test was conducted based on 10 paired results from RoPPT and the strongest baseline MelBERT* on both VUA-18 ($p = 0.014$) and VUA-20 ($p = 0.019$).

We also compared our method against several common denoising strategies. The results show that our tree-based denoising method is more effective than other popular denoising approaches such as RoChunk and RoWindow, which are sequence-based methods. We also apply our target-oriented tree to RoBERTa_SEQ, denoted as the RoBERTa_tree model. The improvement of RoBERTa_tree over RoBERTa_SEQ on two VUA datasets (i.e. 0.3% and 0.7%) further demonstrates the utility of our tree-based denoising method.

Following the setup of Choi et al. (2021), we also conducted a zero-shot transfer learning experiment shown in Table 2. Specifically, our model is trained on the training set of VUA20 and directly tested on the entire Trofi and MOH-X datasets. This is intended to test the generalisation power of trained models. RoPPT shows the best performance on both datasets (significant test on RoPPT against MelBERT*: TroFi $p = 0.0001$; MOH-X $p = 0.021$; we cannot compare with MrBERT as the code is unavailable). It can be observed that our model gives a larger margin of improvement over the baselines on TroFi (i.e., 1.3% gain over MelBERT and 0.6% over MrBERT) than MoH-X (i.e., 0.9% gain over MelBRTT and 0.3% over MrBERT).

Model performance vs. Sentence length. As the averaged sentences length of TroFi (28.3 tokens) is significantly longer than that of MoH-X (8 tokens), it is worth investigating whether our model gives more performance boost on data with longer context as it is likely to be noisier. To verify this hypothesis, we evaluated the performance boost of our RoPPT against the SOTA baseline MelBERT. Table 3 shows the results of VUA18 with the testset splitted into 3 different groups based on sentence length. The results demonstrate a clear positive correlation between performance boost and sentence length.

Impact of Parsers. We also investigated how the choice of parsers impacts the metaphor detection performance of our model. Specifically, we tested two parsers for constructing the target-oriented dependency parse trees, namely, the CNN-based parser Biaffine and the RoBERTa-based parser spaCy. When tested on the validation set, our model achieves 78.0% with spaCy and 77.7% with Biaffine in F1 for metaphor detection, respectively. This shows that the impact of the parse choice is relatively small for our model.

Case Studies. RoPPT shows its strength in the following example with the target word far away from

Sent. len.	RoPPT Prec	RoPPT Rec	RoPPT F1	MelBERT* Prec	MelBERT* Rec	MelBERT* F1	F1 diff.	Pruning comp.	# of Sent.
<20	76.4	74.8	**75.6**	75.0	75.2	75.1	**0.5**	10.7 / 12.3	18,515
20-40	81.8	79.9	**80.8**	79.2	79.1	79.2	**1.6**	16.4 / 29.4	17,729
>40	82.3	80.0	**81.1**	78.5	76.8	77.6	**3.5**	19.5 / 53.6	7,703

Table 3: RoPPT and MelBERT* performance comparison on sentences with different length range from VUA18. 'Pruning comp.' is the comparison of the average length of (pruned) / (original) sentences.

its subject, which is correctly labeled by RoPPT but incorrectly by baseline models. For the instance with metaphorical target word *bogged*, "*a routine exercise in extending the government's borrowing power to $3.1 thousand billion became bogged down.*", the target word *bogged* is separated from its subject by a long phrase, which causes baselines (including MelBERT) to fail to detect the metaphor. Thanks to the parse tree, RoPPT links *exercise* directly to the target and produces the right label.

5 Conclusion

In this paper, we proposed, RoPPT, an effective approach to extract contextual information for target words for metaphor detection based on a target-oriented parse tree structure. Extensive experiments show that our model can yield better performance compared to the state-of-the-art. In addition, our method is particularly effective in denoising long sentences, despite its simplicity.

6 Limitations

Empirical experiments show that our method is more effective in denoising long sentences with the proposed target-oriented parse tree. While this is somewhat expected as shorter sentences tend to have cleaner context, it raises a question or limitation of how can we improve the proposed method to better deal with short sentences and improve its performance in these cases. One possibility is to exploit external knowledge (e.g. ConceptNet) to support the detection of the most important contextual words.

References

Noor Fazilla Abd Yusof, Chenghua Lin, and Frank Guerin. 2017. Analysing the causes of depressed mood from depression vulnerable individuals. In *Proceedings of the International Workshop on Digital Disease Detection using Social Media 2017 (DDDSM-2017)*, pages 9–17.

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Eugene Charniak, Don Blaheta, Niyu Ge, Keith Hall, John Hale, and Mark Johnson. 2000. Bllip 1987-89 wsj corpus release 1. *Linguistic Data Consortium, Philadelphia*, 36.

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. 2021. Guiding the growth: Difficulty-controllable question generation through step-by-step rewriting. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5968–5978, Online. Association for Computational Linguistics.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. Melbert: metaphor detection via contextualized late interaction using metaphorical identification theories. *arXiv preprint arXiv:2104.13615*.

Erik-Lân Do Dinh and Iryna Gurevych. 2016. Token-level metaphor detection using neural networks. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 28–33, San Diego, California. Association for Computational Linguistics.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Dan Fass. 1991. met*: A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 17(1):49–90.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. *arXiv preprint arXiv:1808.09653*.

E. Darío Gutiérrez, Guillermo Cecchi, Cheryl Corcoran, and Philip Corlett. 2017. Using automated metaphor identification to aid in detection and prediction of first-episode schizophrenia. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2923–2930, Copenhagen, Denmark. Association for Computational Linguistics.

E. Dario Gutiérrez, Ekaterina Shutova, Tyler Marghetis, and Benjamin Bergen. 2016. Literal and metaphorical senses in compositional distributional semantic models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 183–193, Berlin, Germany. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

George Lakoff. 1993. The contemporary theory of metaphor.

George Lakoff and Mark Johnson. 1980. Metaphors we live by: University of chicago press. *Chicago, IL*.

Duong Le, My Thai, and Thien Nguyen. 2020. Multi-task learning for metaphor detection with graph convolutional neural networks and word sense disambiguation. In *AAAI*, pages 8139–8146.

Chee Wee Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 vua and toefl metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29.

Chee Wee Leong, Beata Beigman Klebanov, and Ekaterina Shutova. 2018. A report on the 2018 vua metaphor detection shared task. In *Proceedings of the Workshop on Figurative Language Processing*, pages 56–66.

Yucheng Li, Frank Guerin, and Chenghua Lin. 2022a. The secret of metaphor on expressing stronger emotion. In *Proceedings of the 3rd Workshop on Figurative Language Processing (FLP)*, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Yucheng Li, Chenghua Lin, and Frank Guerin. 2022b. Cm-gen: A neural framework for chinese metaphor generation with explicit context modelling. In *International Conference on Computational Linguistics*.

Yucheng Li, Chenghua Lin, and Frank Guerin. 2022c. Nominal metaphor generation with multitask learning. In *Proceedings of the 15th International Conference on Natural Language Generation*, pages 225–235.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and wordnet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1222–1231.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Wei Song, Shuhui Zhou, Ruiji Fu, Ting Liu, and Lizhen Liu. 2021a. Verb metaphor detection via contextual relation learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4240–4251, Online. Association for Computational Linguistics.

Wei Song, Shuhui Zhou, Ruiji Fu, Ting Liu, and Lizhen Liu. 2021b. Verb metaphor detection via contextual relation learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4240–4251.

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. *arXiv preprint arXiv:2004.12362*.

Yorick Wilks. 1978. Making preferences more active. *Artificial intelligence*, 11(3):197–223.

Low-Resource Compositional Semantic Parsing with Concept Pretraining

Subendhu Rongali[*]
Amazon Alexa AI
rongals@amazon.com

Mukund Sridhar[*]
Google
smukund@google.com

Haidar Khan
Amazon Alexa AI
khhaida@amazon.com

Konstantine Arkoudas[*]
Dyania Health
konstantine@dyaniahealth.com

Wael Hamza
Amazon Alexa AI
waelhamz@amazon.com

Andrew McCallum
UMass Amherst
mccallum@iesl.cs.umass.edu

Abstract

Semantic parsing plays a key role in digital voice assistants such as Alexa, Siri, and Google Assistant by mapping natural language to structured meaning representations. To extend the capabilities of a voice assistant for a new domain, the underlying semantic parsing model needs to be retrained using thousands of annotated examples from the new domain, which is time-consuming and expensive. In this work, we present an architecture to perform such *domain adaptation* automatically, with only a small amount of metadata about the new domain and without any new training data (zero-shot) or with very few examples (few-shot). We use a base seq2seq (sequence-to-sequence) architecture and augment it with a *concept* encoder that encodes intent and slot tags from the new domain. We also introduce a novel decoder-focused approach to pretrain seq2seq models to be concept aware using Wikidata. This pretraining helps our model learn important concepts and perform well in low-resource settings. We report few-shot and zero-shot results for compositional semantic parsing on the TOPv2 dataset and show that our model outperforms prior approaches in few-shot settings for the TOPv2 and SNIPS datasets.

1 Introduction

Voice assistants such as Alexa, Siri, and Google Assistant often rely on semantic parsing to understand requests made by their users. The underlying semantic parsing model converts natural language user utterances into logical forms consisting of actions requested by the user (play music, check weather), called *intents*, and relevant entities in the request (which song? which location?), called *slots*. The model is built to process requests in a fixed set of domains, such as music, weather, shopping, and so on. With voice assistants increasingly pervading more aspects of daily life, systems need to be continuously updated to comprehend new intents and slots across an ever-growing number of domains.

Current semantic parsing models are trained on large amounts of annotated data from a predetermined set of domains. Extending these models to learn new intents or slots typically involves collecting and annotating large amounts of new data. This process is expensive and time-consuming. To combat this problem, researchers have proposed semantic parsing models that can be efficiently trained with fewer examples (few-shot) from new domains (Shrivastava et al., 2021; Mansimov and Zhang, 2021; Ghoshal et al., 2020; Shin et al., 2021; Desai et al., 2021; Rongali et al., 2022; Shrivastava et al., 2022). While these methods facilitate few-shot learning, they have limitations. Some of them rely on hand-crafted knowledge such as intermediate grammars or logical form templates (Shrivastava et al., 2022; Shin et al., 2021; Rongali et al., 2022). Others rely on very large pretrained language models, such as GPT-3, to perform in-context learning by appending test examples with instructional prompts (Shin et al., 2021).

In this work, we explore few-shot domain adaptation for semantic parsing without any additional hand-crafted knowledge apart from the intent and slot tag names, and with much smaller architectures that can perform efficient inference in practical production environments. We also explore zero-shot domain adaptation, when we have no annotated training data from a new domain.

To that end, we propose CONCEPT-SEQ2SEQ, a novel architecture based on a state-of-the-art semantic parsing model, SEQ2SEQ-PTR (Rongali et al., 2020), which uses seq2seq models and a pointer generator network to decode the target semantic parse. We augment this model with a *concept* encoder that encodes intents and slots from the schema and uses those encodings to conditionally decode the semantic parse. Figure 1 shows

---

[*] work done when SR was at UMass and MS and KA were at Amazon; it does not relate to their current positions.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1410–1419
May 2-6, 2023 ©2023 Association for Computational Linguistics

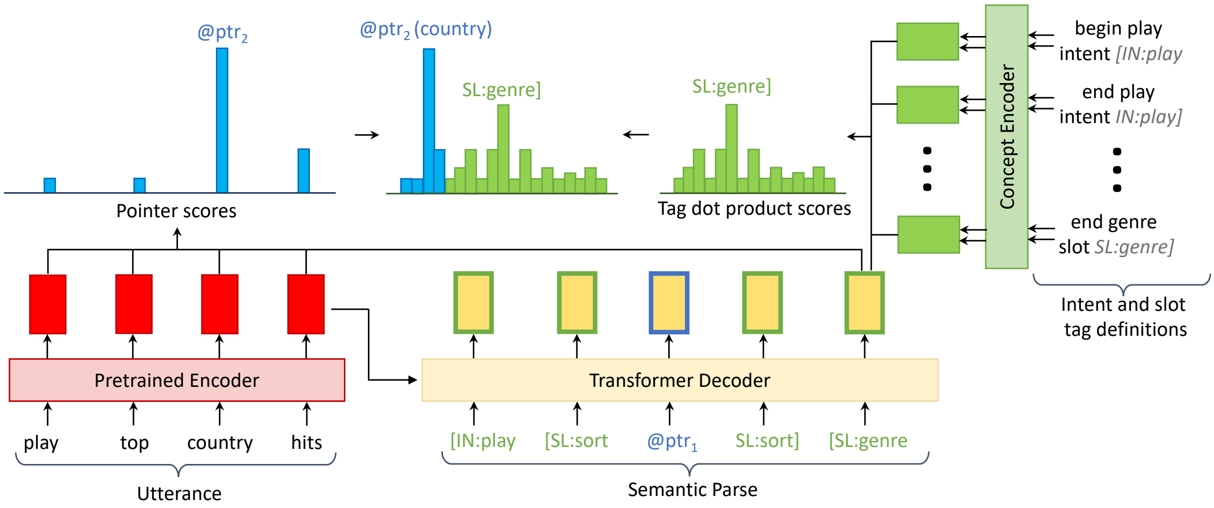


Figure 1: The architecture of CONCEPT-SEQ2SEQ for low resource domain adaptation. The concept encoder encodes descriptions of each of the concept tags into an embedding and incorporates them into the decoded parse.

the architecture of our proposed model. We train this model on annotated data from the given domains. During inference, we simply encode all intents and slots from the schema, including new, unseen ones, into the learned concept space, and decode the target parse. This model has the same time complexity as the original SEQ2SEQ-PTR model but comes with the added benefit of now being able to effectively parse utterances from unseen domains without any additional effort.

There have been a few zero-shot semantic parsing approaches proposed in the past but they either covered only simple slot-filling style utterances (Bapna et al., 2017; Lee and Jha, 2019) or compositional utterances that also came with carefully crafted intermediate representations and context-free grammars (Herzig and Berant, 2018; Wu et al., 2021). Our model is capable of performing zero-shot domain adaptation for compositional semantic parsing, producing meaning representations with nested intents and slots, but also doesn't require any grammars, whose construction effort often exceeds the effort required to annotate a few examples.

In few-shot scenarios, we fine-tune our zero-shot model checkpoints further on the small number of available examples. Due to the presence of the concept encoder in our architecture, we expect to receive better knowledge-transfer advantages by encoding intent and slot tags from new domains as opposed to initializing them as new tags. To further improve performance, we propose a novel decoder-focused pretraining scheme for CONCEPT-SEQ2SEQ using an entity-centric processed version of Wikidata (Vrandečić and Krötzsch, 2014) called WikiWiki (Li et al., 2022), to help it better encode unseen concepts and parse effectively.

We report the first zero-shot performance numbers for semantic parsing on the compositional TOPv2 dataset (Chen et al., 2020) and show that CONCEPT-SEQ2SEQ achieves commendable zero-shot performance on the flat-entity SNIPS dataset (Coucke et al., 2018). We also evaluate in few-shot settings and show that we match or outperform previous state-of-the-art models while still being production-viable.

In summary, our contributions are as follows.

- We propose CONCEPT-SEQ2SEQ, a bi-tower architecture with a seq2seq model and a concept encoder, that can perform few-shot and zero-shot domain adaptation for compositional semantic parsing without additional handcrafted knowledge.
- We propose a novel decoder-focused pretraining scheme for CONCEPT-SEQ2SEQ using Wikidata that helps it better encode unseen concepts and parse effectively.
- We report few-shot and zero-shot semantic parsing results on the TOPv2 and SNIPS datasets and show that our model outperforms or matches previously proposed approaches on a variety of few-shot settings.

2 Methodology

In this section, we describe our proposed model, CONCEPT-SEQ2SEQ, for low resource (few-shot and zero-shot) domain adaptation for semantic parsing. It is based on the Seq2Seq-Ptr model from Rongali et al. (2020), consisting of a sequence-to-sequence encoder-decoder component, augmented with a pointer generator network to constrain the target decoding vocabulary. Since our task at hand is to perform potential zero-shot semantic parsing with just descriptive metadata about the new domain, we modify the architecture of Seq2Seq-Ptr to incorporate information about new intents and slots from new domains by adding a concept encoder. Section 2.2 describes this architecture in detail. To help our model learn to parse utterances from unseen domains better, we also propose a novel pretraining scheme to incorporate general concept parsing knowledge into it. Section 2.3 describes this concept pretraining scheme. Finally, we describe Concept-Seq2Seq model specifics for few-shot and zero-shot settings in Section 2.4. Before we get to these sections, we first describe the source and target sequence formulation for the semantic parsing task below.

2.1 Task Formulation

Our model solves semantic parsing as a sequence-to-sequence task, where the source sequence is the utterance and the target sequence is a linearized representation of the semantic parse. Following Rongali et al. (2020), we modify the target sequence to only contain intent/slot tags or pointers to utterance tokens. An example source and target sequence from the TOPv2 dataset are given below.

`Source: How far is the coffee shop`
`Target: [IN:GET_DISTANCE` $@ptr_0$ $@ptr_1$ $@ptr_2$
`[SL:DESTINATION [IN:GET_RESTAURANT_LOCATION`
$@ptr_3$ `[SL:TYPE_FOOD` $@ptr_4$ `SL:TYPE_FOOD]`
$@ptr_5$ `IN:GET_RESTAURANT_LOCATION]`
`SL:DESTINATION] IN:GET_DISTANCE]`

Each $@ptr_i$ token here points to the i^{th} token in the source sequence. So $@ptr_4$ corresponds to the word *coffee*.

2.2 Model Architecture

Concept-Seq2Seq consists of three main components: an encoder, a decoder, and a concept encoder. Just like in traditional sequence-to-sequence models, the encoder encodes the source sequence, and the decoder autoregressively decodes the target sequence. However, to effectively understand new intent and slot tags in target sequences that the model hasn't seen during training, our model needs to be able to incorporate new intents and slots, or concepts, and decode the target sequence accordingly. The concept encoder helps us do this by encoding descriptive metadata about new concepts and creating vector representations that we can use while decoding the target sequence.

Specifically, for an input sequence $[x_1 \ldots x_n]$, we first encode it using the encoder into a sequence of hidden states $e_1 \ldots e_n$. Then, having generated the first $t-1$ tokens, the decoder generates the token at step t as follows. It first produces the decoder hidden state at time t, d_t by building a multi-layer, multi-head self-attention on the encoder hidden states and the decoder states so far. This step is based on the transformer decoder from Vaswani et al. (2017). In a traditional sequence-to-sequence generation task, d_t is then fed into a dense layer to produce scores over the target vocabulary.

Our target vocabulary consists of pointer tokens and the concept tags. Since we do not have access to all concept tags at the time of training, we train our model to incorporate descriptive information about concepts instead of using a fixed-size dense layer. To do this, we encode intent and slot concepts using a **concept encoder**. The descriptions we use in this work are simply rule-based naturalized versions of the intent and slot names in the dataset. For example, the description for the intent tag token `[IN:GET_DISTANCE` is set as "*begin get distance intent*". Similarly, for `SL:DESTINATION]`, it is set to "*end destination slot*". We purposely use just this information and no additional handcrafted knowledge to remove any additional user input and to compare to previous approaches in the same setting.

Given m concept tokens (both begin and end) and their descriptions, the concept encoder encodes each of them to produce concept vector representations $[c_1 \ldots c_m]$. We then use the computed decoder hidden state at t, d_t, as the query and compute unnormalized attention scores $[s_1 \ldots s_m]$ with $[c_1 \ldots c_m]$, and $[a_1 \ldots a_n]$ with $[e_1 \ldots e_n]$. Concatenating all these scores, we obtain an unnormalized distribution over $m+n$ tokens, the first m of which are the intent and slot tagging tokens from the concepts, and the last n of which are the $@ptr_i (0 < i < n)$ tokens pointing to the source sequence. We feed this through a softmax layer to obtain the final probability distribution. This probability distribution is used in the loss function during training and to choose the next token to generate during inference. For the target token embeddings in the decoder, we use a set of special embeddings to represent the $@ptr_i$ tokens and $[c_1 \ldots c_m]$ to rep-

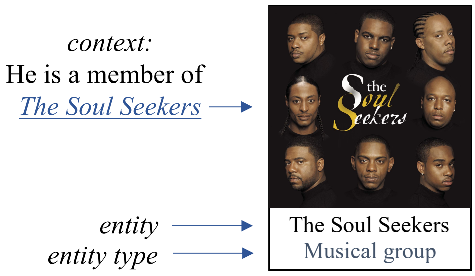


Figure 2: An example sentence from the Wikiwiki dataset with the associated mention, entity, and type fields. The full hyperlinked sub-span is extracted as the mention and the entity and type are extracted from the target page.

resent the concept embeddings.

Figure 1 shows this process in action on a toy example. The model is decoding the next token after `SL:genre` at step 5. To do this, the model computes the pointer attention scores $[a_1 \dots a_n]$ (blue, left) and the concept token attention scores $[s_1 \dots s_n]$ (green, right). The highest overall score is for the token $@ptr_2$, corresponding to the word *country* in the source sequence, so the next predicted token is *country*.

2.3 Concept Pretraining

CONCEPT-SEQ2SEQ has the ability to incorporate new, unseen concepts while parsing using the concept encoder and transfer knowledge on similar concepts. In order to produce these unseen concepts or types, and have our model be robust in low-data settings, it is important for our decoder to be type aware. Conventional seq2seq pretraining schemes such as Lewis et al. (2020); Raffel et al. (2020); Soltan et al. (2022) pretrain the decoder using language modeling criterion. Li et al. (2022) extend the language modeling task to induce entity-type information by treating it as a question answering task. We pretrain the seq2seq model on the semantic parsing task using the WikiWiki (Li et al., 2022) dataset. We explain how to achieve this by keeping an open-domain, extensible output space for the semantic parse.

The Wikiwiki dataset curates mentions, entities, and entity types from 10M Wikipedia documents using hyperlink information linking sub-spans of text in sentences to other Wikipedia pages. The hyperlink is considered as the mention, and the entity and the type information are extracted from the new page. For further details on this processing, please refer to Li et al. (2022). This dataset contains around 2m entities and 40K entity types.

Each example in the Wikiwiki dataset consists of a *context*, which is a paragraph from a wiki page, *mentions*, which are sub-spans of text that link to another page, *entities*, which correspond to each mention, and *entity types*, which describe the type of the entity. We extract individual sentences from this dataset and use them to train CONCEPT-SEQ2SEQ to learn to encode a wide variety of concepts using the entity type fields as descriptions and tag the relevant mentions in the sentence. Figure 2 shows an example sentence from this dataset and the different fields. The source and target sequences for pretraining, and the descriptions of the concept tags for this example are given below.

Source: He is a member of The Soul Seekers
Target: $@ptr_0$ $@ptr_1$ $@ptr_2$ $@ptr_3$ $@ptr_4$
    [Q215380 $@ptr_5$ $@ptr_6$ $@ptr_7$ Q215380]
Concept descriptions:
    [Q215380: begin musical group
    Q215380]: end musical group

During training, we collect all the concept tokens within a training batch and use them to create in-batch negatives (denominator of the softmax calculation) for the decoding task. We do this since it is extremely inefficient to encode all 40K × 2 concept token descriptions (begin and end) from Wikiwiki in every step.

2.4 Few-shot and Zero-shot Specifics

CONCEPT-SEQ2SEQ is primarily designed to perform low resource domain adaptation for semantic parsing by effectively encoding the output space via a concept encoder.

In the zero-shot setting, we deploy the following procedure to build our models. We first perform concept pretraining on CONCEPT-SEQ2SEQ using Wikiwiki example sequences. We take this checkpoint and train on a set of known domains to then obtain the zero-shot model. During inference, we encode all the intent and slot tags from the new unknown domain using the concept encoder of the obtained zero-shot model and set the appropriate decoder parameters to reduce the architecture to a simple encoder-decoder setting.

In few-shot settings, we further fine-tune the zero-shot checkpoint on the available handful of training examples. Since we explore extremely low-resource settings (1, 5, 25 samples per intent/slot), we run the risk of over-fitting and instability during training. To account for these risks and smooth training, we augment the finetuning loss at every step with the loss from a randomly sampled

	Alarm	Event	Timer	Weather	Alarm	Event	Timer	Weather
	F1 score				EM Accuracy			
CONCEPT-SEQ2SEQ w/o pretraining	62.53	30.25	55.68	51.23	45.94	0.00	0.51	8.62
CONCEPT-SEQ2SEQ	71.00	70.47	58.48	66.29	53.64	20.21	3.86	26.63

Table 1: Zero-shot performance of CONCEPT-SEQ2SEQ on domains in TOPv2. We observe notable scores in the alarm and weather domains and improvements across all the domains after the concept pretraining step.

batch of training data from the known domains. We scale the loss from the random known domain batch down using a multiplier before adding to the loss. This scheme is akin to rehearsal (Ratcliff, 1990), a popular technique in domain adaptation.

3 Experimental Setup

We evaluate CONCEPT-SEQ2SEQ on few-shot and zero-shot domain adaptation using two popular English task-oriented semantic parsing datasets - TOPv2 (Chen et al., 2020) and SNIPS (Coucke et al., 2018). Both datasets have utterances grouped into multiple domains; TOPv2 has eight domains and SNIPS has seven intents from seven different areas, which we consider domains. TOPv2 is a large dataset consisting of 10k-20k training and 3k-7k test examples per domain. It is also comprised of compositional examples with nested intents and slots. We exclude the *unsupported* utterances from the training and test sets in TOPv2 for the zero-shot experiments (we use the full sets in few-shot). *Unsupported* utterances consist of utterances that belong to a domain but are not supported, which is impossible to learn in zero-shot. SNIPS is a smaller and simpler dataset with flat, disjoint slots. It has 2k training and 100 test examples per domain.

For zero-shot, we use a leave-one-out approach where given n domains, we train models on annotated data from $n-1$ of them and evaluate on utterances from the left-out domain. For few-shot settings, where the model has access to a few annotated examples from the left-out domain, we further fine-tune using 1, 5, and 25 samples-per-intent/slot (SPIs), which we randomly sample from the training data of the left-out domain. We fine-tune CONCEPT-SEQ2SEQ on three different randomly sampled training sets per domain per SPI setting and report the average performance score of the three runs.

We use a transformer encoder, initialized from a *roberta-base* checkpoint, for CONCEPT-SEQ2SEQ. The decoder is a transformer decoder initialized from scratch and it contains 6 layers, 8 heads, and a hidden state size of 768. The concept encoder is also a transformer encoder and it is initialized from a *bert-base-uncased* checkpoint. We choose a BERT-based model here since it is pretrained to compute a vector for the whole sequence using the CLS token, which is what we need for encoding a concept consisting of a multi-word description. We also choose all *base*-size components to keep the overall model size small and expect the relative improvements shown by our model to generalize.

We train our zero-shot models using sequence cross entropy loss. We use the Adam optimizer with learning rate $2e^{-5}$ and $\epsilon = 1e^{-8}$, warm-up proportion 0.1, weight decay 0.01, and batch-size 128. The number of epochs is set to 100 and we evaluate after every epoch and early stop with a patience of 5. For the Wikiwiki pretraining step, we use the same hyper-parameters but stop the model training after 2 epochs on the entire Wikiwiki dataset. We did not perform explicit hyper-parameter tuning.

For the few-shot experiments, we take the zero-shot model trained by excluding the given few-shot domain and fine-tune it on small set of annotated examples for 1000 epochs, evaluating after every 25 epochs. All other parameters are set to the same values as in the initial zero-shot training. We use a multiplier of 0.1 for the augmented loss from the random known-domain data batch. To speed up evaluation during training, we use teacher-forced sequence accuracy as our validation metric, which doesn't require us to perform any beam search. During inference, we use beam search decoding with a beam size of 4.

We report exact match (EM) accuracy for the few-shot experiments, meaning the entire predicted parse has to match exactly with the gold parse. For zero-shot, we report both EM and F1 score since the task is more difficult and the performance is generally lower. At these lower numbers, F1 score, which awards partial credit to correctly tagged spans provides a better picture for improvements than EM accuracy, which requires the entire pre-

	Music	Book	Creative	Weather	Restaurant	Playlist	Screening
F1 score							
Slot-filling (Lee and Jha, 2019; Bapna et al., 2017)	31.20	**34.13**	86.21	65.64	51.40	**59.96**	44.50
CONCEPT-SEQ2SEQ	**50.00**	30.26	**88.75**	**74.58**	**57.78**	57.11	**45.78**
EM Accuracy							
Slot-filling (Lee and Jha, 2019; Bapna et al., 2017)	11.00	1.00	69.00	37.00	12.00	**21.00**	24.00
CONCEPT-SEQ2SEQ	**20.00**	**2.00**	**69.00**	**42.00**	**13.00**	19.00	**26.00**

Table 2: Zero-shot performance of CONCEPT-SEQ2SEQ on domains in SNIPS. Our model matches or outperforms the slot-filling style baseline on most domains.

dicted parse to be correct for credit. For comparison wherever applicable, we use prior state-of-the-art models as baselines. In addition, we also report the performance of a vanilla SEQ2SEQ-PTR model without concept pretraining.

4 Results and Discussion

In this section, we report and discuss the performance of CONCEPT-SEQ2SEQ on zero-shot and few-shot domain adaptation for semantic parsing. We first briefly describe our findings in zero-shot setting and then describe findings in a variety of few-shot scenarios.

4.1 Zero-shot Domain Adaptation

We report the first zero-shot performance numbers for domain adaptation on the TOPv2 dataset. Table 1 contains these numbers. We observed that our model produced decent predictions on four of the eight domains in the dataset, which we document in the table. For the other domains, the scores were very low. CONCEPT-SEQ2SEQ achieves good F1 and EM Accuracy scores on the alarm domain (71.00% F1 and 53.64% EM). On the weather and event domains, the concept pretraining step helps it achieve decent EM scores around 20%. On the timer domain, CONCEPT-SEQ2SEQ achieves a fairly high F1 score (58.48%) but a very low EM score (3.86%). Upon manual examination, we found that this was because our model always skipped a certain tag. In the timer domain, there is a slot tag called `SL:METHOD_TIMER` which tags the kind of timer such as *timer* or *stopwatch*. Our model never learns to tag these words with that slot. We believe this is probably due to the description being inadequate for performing the requisite task.

Overall, we believe the task at hand here is difficult due to the combination of the zero-shot setting and the presence of specific nesting/parsing rules in a compositional semantic parsing task. While a good amount of information can be gleaned from the intent and slot names, our model has no access to any new kinds of tagging rules since it has no annotated data or any descriptions of those rules within the concept descriptions. The descriptions themselves are sometimes inadequate as described with the timer domain above. We simply use the descriptions from the dataset and they weren't really designed to be used to describe the entity being tagged. We leave exploration into better descriptions and incorporating parsing rules without explicit annotations for future work.

We also evaluated our model on the SNIPS dataset to compare CONCEPT-SEQ2SEQ to prior zero-shot approaches for flat slot-filling style datasets. We created a strong baseline using recent NLP advancements such as pretrained transformers and attention mechanisms on the slot-filling style zero-shot model proposed by Bapna et al. (2017) and Lee and Jha (2019). Table 2 compares the performance of CONCEPT-SEQ2SEQ to this baseline. We observe that our model matches or outperforms the slot-filling baseline on most domains while also being adaptable to compositional datasets.

4.2 Few-shot Domain Adaptation

We evaluated CONCEPT-SEQ2SEQ in few-shot settings of 1, 5, and 25 samples per intent/slot. Table 3 reports the EM accuracy scores of CONCEPT-SEQ2SEQ and other recent baselines on TOPv2. We also report the performance of a fully trained CONCEPT-SEQ2SEQ model on all the training data for reference and to show that the architecture of CONCEPT-SEQ2SEQ is competitive with other state-of-the-art methods in the full-resource setting.

We evaluated with a range of SPIs to allow for comparison with a wide range of models focused on both extremely low-resource (1, 5 SPIs) and medium low-resource (25 SPIs) settings. We report numbers for the baselines from their original

	Alarm	Event	Messaging	Music	Navigation	Reminder	Timer	Weather
Few-shot 1 SPIs								
Seq2Seq-Ptr	20.41	31.85	38.12	25.58	19.96	23.66	16.62	47.24
Inventory (Desai et al., 2021)	62.13	46.57	**46.54**	23.00	21.16	28.58	28.92	54.53
RAF (Shrivastava et al., 2022)	62.71	-	-	35.47	-	-	**55.06**	**61.05**
Concept-Seq2Seq w/o pretraining	61.72	44.28	34.24	20.66	20.82	35.39	44.75	52.24
Concept-Seq2Seq	**64.71**	**54.42**	46.13	**36.30**	**30.00**	**36.93**	53.44	54.68
Few-shot 5 SPIs								
Seq2Seq-Ptr	45.50	38.31	52.79	48.75	43.38	36.37	54.79	49.94
Inventory (Desai et al., 2021)	71.81	58.87	**63.72**	**53.59**	42.59	48.88	55.54	65.09
Concept-Seq2Seq w/o pretraining	71.32	53.73	51.52	45.96	50.71	50.83	58.89	66.65
Concept-Seq2Seq	**74.17**	**61.72**	61.20	51.24	**56.76**	**54.36**	**63.13**	**68.54**
Few-shot 25 SPIs								
Seq2Seq-Ptr	-	-	-	-	-	55.7	-	71.6
RINE (Mansimov and Zhang, 2021)	-	-	-	-	-	**68.71**	-	74.53
Concept-Seq2Seq w/o pretraining	78.16	68.21	75.28	65.54	67.67	67.92	70.72	74.30
Concept-Seq2Seq	**79.87**	**72.96**	**80.45**	**67.91**	**70.94**	67.76	**72.41**	**76.44**
Reference - Fully trained								
Concept-Seq2Seq	88.07	83.23	93.11	79.47	81.63	79.57	77.33	90.73

Table 3: EM Accuracy scores of various models in few-shot settings on TOPv2. We see that our model outperforms prior approaches on many domains and settings, most notably in the 1 SPIs setting.

papers, so they are missing for some domains.

As shown in the table, Concept-Seq2Seq outperforms a vanilla Seq2Seq-Ptr, Inventory (Desai et al., 2021), and Retrieve-and-Fill (RAF) (Shrivastava et al., 2022) models on most domains in the 1 SPIs setting. RAF scores are very close and the approach outperforms our model on two domains but it uses additional hand-crafted information such as handmade descriptions and examples for intents and slots, as well as an intermediate scenario-bank to retrieve templates from. Concept-Seq2Seq simply works off of the existing information in the dataset. In the 5 SPIs setting, it again outperforms the vanilla Seq2Seq-Ptr and Inventory models on most domains and on average. Inventory is a similar model to ours where the lexical information from intents and slots is used to help better transfer knowledge in the low-resource setting. However, this information is prepended to the input sequence and this might cause input size issues for large inventories. In the slightly higher resource setting of 25 SPIs, Concept-Seq2Seq beats a vanilla Seq2Seq-Ptr model and matches the performance of RINE (Mansimov and Zhang, 2021), reported on two domains.

Across all the SPIs settings, we see that there is a noticeable drop in performance of Concept-Seq2Seq without the Wikiwiki concept pretraining. This shows the effectiveness of the pretraining step in helping the model generalize to unseen concepts and domains better.

To wrap up our evaluation, we also report the performance of Concept-Seq2Seq on SNIPS in the few-shot settings described above. Table 4 shows these numbers. We can see that we almost catch up to a fully trained model by training with just 25 SPIs in this dataset on most domains except music. We believe the music domain probably requires a lot of samples to effectively identify the diverse set of entities in the domain.

Overall, we find Concept-Seq2Seq to be a very promising approach which achieves high performance scores in low resource domain adaptation. It is capable of doing this in both compositional and flat semantic parsing settings, without any additional hand-crafted information apart from the little documentation in the dataset, and with the memory and inference latency footprint of a vanilla Seq2Seq-Ptr model.

5 Related Work

Zero-shot domain adaptation for task-oriented semantic parsing has been previously explored for simple flat queries with single intents and disjoint, non-overlapping slots. Bapna et al. (2017) and Lee and Jha (2019) encode the lexical tag features and create a token-tagging schema to create the final semantic parses. Yu et al. (2021) solve the task using

	Music		Book		Creative		Weather		Restaurant		Playlist		Screening	
	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
1 SPIs	67.10	39.00	86.74	61.67	92.39	80.33	88.14	71.00	89.64	72.00	78.31	47.00	79.19	61.00
5 SPIs	84.33	68.00	96.83	89.33	92.58	84.33	94.64	86.33	93.27	81.33	87.74	68.33	94.16	87.67
25 SPIs	85.90	72.67	99.32	97.33	98.10	95.33	98.27	94.67	96.38	89.33	91.73	80.67	95.02	89.33
Fully-trained	90.78	83.00	99.18	97.00	100.00	100.00	98.55	96.00	96.89	90.00	94.53	87.00	98.35	97.00

Table 4: Few-shot performance of CONCEPT-SEQ2SEQ on domains in SNIPS. Our 25 SPIs model almost catches up to a fully trained model. Numbers are an average of three runs with different random samples of SPIs.

a retrieve-and-fill mechanism. Our baseline model for simple queries is based on these approaches.

For complex utterances with nested structures, zero-shot semantic parsing has been explored using intermediate, concept-agnostic logical forms (Herzig and Berant, 2018; Dong and Lapata, 2018; Reddy et al., 2017) or natural language canonical forms (Wu et al., 2021). These approaches apply to semantic parsing datasets which have context free grammars and specified rules, such as database or knowledge graph queries. The effort to craft these grammars for task-oriented semantic parsing in a voice assistant setting could quite possibly be greater than annotating utterances.

A more relevant class of approaches for this work are ones that solve task-oriented semantic parsing for complex utterances in a few-shot setting using lexical tag features. Shrivastava et al. (2021) and Mansimov and Zhang (2021) modify the seq2seq architecture from Rongali et al. (2020) to perform non-autoregressive style decoding and show that their models perform better in a few-shot setting. Ghoshal et al. (2020) use adaptive label smoothing, a model-agnostic technique. Shin et al. (2021) proposed a prompting-style approach where custom instructional prompts filled with handful of annotated examples and an unsolved utterance are fed as input to GPT-3 to directly produce a semantic parse. Their approach is extremely slow and cannot be easily adapted into a zero-shot framework. Shrivastava et al. (2022) explore a retrieve-and-fill style approach where they retrieve the best *scenario*, an intermediate logical form consisting of the semantic frame and abstracted out tags, from a scenario bank of all supported semantic parses. Their approach is contingent on the availability of this scenario bank which could possibly entail more effort than annotating utterances. Mueller et al. (2022) and Desai et al. (2021) use lexical features from intent and slot names to create an *inventory* and use it as input to train semantic parsers for new domains. Mueller et al. (2022) also pretrain their model to improve generalizability but only evaluate it on an intent classification task. Desai et al. (2021) evaluate their model for full sequences and our model is similar to theirs. However, we use our inventory to create custom decoder embeddings in a seq2seq model, which removes any input size issues that their model will encounter with large inventories. We also pretrain our model with Wikidata and evaluate it in a completely zero-shot setting, in addition to few-shot. Zhao et al. (2022) is another recent question-answering-based approach that uses lexical features from the intent and slot tags by using them as context and posing questions but it has a similar input size issue with large inventories.

6 Conclusion

We propose a model called CONCEPT-SEQ2SEQ to perform low-resource domain adaptation for compositional semantic parsing. Our model is built on the SEQ2SEQ-PTR framework and is augmented with a concept encoder to transfer knowledge and encode unseen intents and slots from new domains through their text definitions. We also propose a novel concept pretraining scheme to incorporate general concept knowledge into our model using an entity-centric Wikipedia dataset called Wikiwiki.

We evaluate our model in zero-shot and multiple few-shot settings on Facebook TOPv2 and SNIPS datasets. We show that our model is capable of performing zero-shot domain adaptation on some domains of the TOPv2 dataset and beats a strong slot-filling baseline on the SNIPS dataset. In few-shot, over multiple dataset sizes of 1, 5, and 25 SPIs, we show that our model outperforms many strong prior models on TOPv2. Using the SNIPS dataset, we also demonstrate how our model catches up to a fully-trained semantic parsing model using just 25 SPIs on most domains. Our model is capable of low-resource domain adaptation in both compositional and flat parsing settings, without additional hand-crafted information, and with the inference behavior of a vanilla SEQ2SEQ-PTR model.

Acknowledgements

This work was supported in part by Amazon Alexa AI, in part by the Chan Zuckerberg Initiative, in part by IBM Research AI through the AI Horizons Network, and in part by the National Science Foundation (NSF) grants IIS-1763618 and IIS-1955567. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

Limitations

Our models were trained on GPUs that had at least 20GB on-board memory since in addition to the traditional encoder and decoder components in a seq2seq model, we also train a concept encoder, which is around the same size as the encoder. This eliminates the use of popular GPUs such as 1080-ti and 2080-ti, unless parameter freezing or other tricks are employed during training. During inference however, once the concepts are trained, we can simply encode the target tags and this reduces to the size and performance of a traditional SEQ2SEQ-PTR model.

We also report all results on models (ours and baselines) with *base*-size components such as *roberta-base*. We do this since these models are more likely to be used in production than the much bigger *large*-size models. Results and comparison with *large*-sized models is missing from this work (we expect the trends shown to generalize) and we leave this to future work.

Finally, to simulate our low-resource experiments, we randomly sample a few examples from the existing training datasets. While this is useful for experimentation, it doesn't truly mimic a real low-resource workflow where these few examples could be carefully crafted by developers to ensure better semantic coverage in terms of the language of the utterances. This work doesn't include any analysis on the influence of the content of the few selected examples; it just focuses on their number.

References

Ankur Bapna, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *arXiv preprint arXiv:1707.02363*.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-resource domain adaptation for compositional task-oriented semantic parsing. *arXiv preprint arXiv:2010.03546*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Shrey Desai, Akshat Shrivastava, Alexander Zotov, and Ahmed Aly. 2021. Low-resource task-oriented semantic parsing via intrinsic modeling. *arXiv preprint arXiv:2104.07224*.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*.

Asish Ghoshal, Xilun Chen, Sonal Gupta, Luke Zettlemoyer, and Yashar Mehdad. 2020. Learning better structured representations using low-rank adaptive label smoothing. In *International Conference on Learning Representations*.

Jonathan Herzig and Jonathan Berant. 2018. Decoupling structure and lexicon for zero-shot semantic parsing. *arXiv preprint arXiv:1804.07918*.

Sungjin Lee and Rahul Jha. 2019. Zero-shot adaptive transfer for conversational language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6642–6649.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Shuyang Li, Mukund Sridhar, Chandana Satya Prakash, Jin Cao, Wael Hamza, and Julian McAuley. 2022. Instilling type knowledge in language models via multi-task qa. In *Findings of the 2022 North American Chapter of the Association for Computational Linguistics (NAACL)*.

Elman Mansimov and Yi Zhang. 2021. Semantic parsing in task-oriented dialog with recursive insertion-based encoder. *arXiv preprint arXiv:2109.04500*.

Aaron Mueller, Jason Krone, Salvatore Romeo, Saab Mansour, Elman Mansimov, Yi Zhang, and Dan Roth. 2022. Label semantic aware pre-training for few-shot text classification. *arXiv preprint arXiv:2204.07128*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Roger Ratcliff. 1990. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285.

Siva Reddy, Oscar Täckström, Slav Petrov, Mark Steedman, and Mirella Lapata. 2017. Universal semantic parsing. *arXiv preprint arXiv:1702.03196*.

Subendhu Rongali, Konstantine Arkoudas, Melanie Rubino, and Wael Hamza. 2022. Training naturalized semantic parsers with very little data. *arXiv preprint arXiv:2204.14243*.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing. In *Proceedings of The Web Conference 2020*, pages 2962–2968.

Richard Shin, Christopher H Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. *arXiv preprint arXiv:2104.08768*.

Akshat Shrivastava, Pierce Chuang, Arun Babu, Shrey Desai, Abhinav Arora, Alexander Zotov, and Ahmed Aly. 2021. Span pointer networks for non-autoregressive task-oriented semantic parsing. *arXiv preprint arXiv:2104.07275*.

Akshat Shrivastava, Shrey Desai, Anchit Gupta, Ali Elkahky, Aleksandr Livshits, Alexander Zotov, and Ahmed Aly. 2022. Retrieve-and-fill for scenario-based task-oriented semantic parsing. *arXiv preprint arXiv:2202.00901*.

Saleh Soltan, Shankar Ananthakrishnan, Jack FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith Peris, Stephen Rawls, Andy Rosenbaum, Anna Rumshisky, Chandana Satya Prakash, Mukund Sridhar, Fabian Triefenbach, Apurv Verma, Gokhan Tur, and Prem Natarajan. 2022. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Shan Wu, Bo Chen, Chunlei Xin, Xianpei Han, Le Sun, Weipeng Zhang, Jiansong Chen, Fan Yang, and Xunliang Cai. 2021. From paraphrasing to semantic parsing: Unsupervised semantic parsing via synchronous semantic decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5110–5121, Online. Association for Computational Linguistics.

Dian Yu, Luheng He, Yuan Zhang, Xinya Du, Panupong Pasupat, and Qi Li. 2021. Few-shot intent classification and slot filling with retrieved examples. *arXiv preprint arXiv:2104.05763*.

Wenting Zhao, Konstantine Arkoudas, Weiqi Sun, and Claire Cardie. 2022. Compositional task-oriented parsing as abstractive question answering. *arXiv preprint arXiv:2205.02068*.

Made of Steel?
Learning Plausible Materials for Components in the Vehicle Repair Domain

Annerose Eichel, Helena Schlipf, Sabine Schulte im Walde
Institute for Natural Language Processing, University of Stuttgart
`{annerose.eichel,schulte}@ims.uni-stuttgart.de, helena.schlipf@gmail.com`

Abstract

We propose a novel approach to learn domain-specific plausible materials for components in the vehicle repair domain by probing Pretrained Language Models (PLMs) in a cloze task style setting to overcome the lack of annotated datasets. We devise a new method to aggregate salient predictions from a set of cloze query templates and show that domain-adaptation using either a small, high-quality or a customized Wikipedia corpus boosts performance. When exploring resource-lean alternatives, we find a distilled PLM clearly outperforming a classic pattern-based algorithm. Further, given that 98% of our domain-specific components are multiword expressions, we successfully exploit the compositionality assumption as a way to address data sparsity.

1 Introduction

Connecting a symptom to an underlying cause is a crucial building block for natural language understanding across domains. For example, as illustrated in Fig. 1, a standard approach that human mechanics in vehicle repair shops apply when tracing the cause of a symptom, is to exploit the link between a vehicle *component* and the component's *materials*. This study tackles the task of automatically learning plausible materials for vehicle components. The information is crucial in vehicle repair shops, as mechanics are faced with constantly growing vehicle complexity, making it hard to manually identify the cause of a malfunction. While domain-specific information on vehicle components is often available, plausible domain-specific material information typically needs to be gathered from external data sources. For example, from "*Brake disk are usually manufactured from gray cast iron*", one may retrieve the information that *brake disks* consist of *iron* (and possibly further materials not mentioned here).

We propose a novel approach to leverage state-of-the-art pretrained language models (PLMs) for the task of learning plausible materials in highly domain-specific contexts, such as vehicle components in the vehicle repair domain. Our approach can also be applied to similarly specific domains, e.g., to learn which materials or substances are plausibly used in products in the health domain, textile industry, etc. As we tackle a task from material science, we make a contribution to a field that is not only very challenging, but heavily under-investigated from an NLP perspective. We focus on the PLM RoBERTa (Liu et al., 2019) and develop an approach that successfully learns plausible materials in English for components in the vehicle repair domain. We further exploit domain-adaptation variants and show (i) that harnessing a small high-quality domain-specific corpus boosts the performance over an out-of-the-box vanilla RoBERTa, and (ii) how similar performance can be reached by using a widely accessible data source such as Wikipedia if no domain-specific data is available.

To overcome the typical lack of annotated datasets for training and fine-tuning models in highly domain-specific tasks, we probe RoBERTa in a cloze task setting (Taylor, 1953; Petroni et al., 2019), compare variants for aggregating the results, and analyze the influence of a varied set of query templates on model predictions.

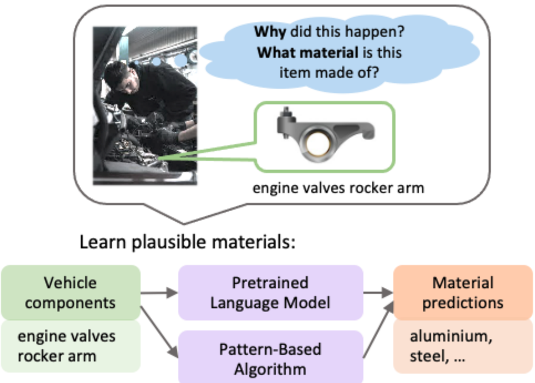


Figure 1: Conceptual overview for learning domain-specific plausible materials (e.g., *aluminium*) for vehicle components (e.g., *engine valves rocker arm*).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1420–1435
May 2-6, 2023

While PLMs come with great advantages, their hunger for storage, data, and computing power is even greater. We thus compare the performance of the vanilla and domain-adapted versions of RoBERTA with (i) a distilled version, i.e., the much smaller DistilRoBERTA (Sanh et al., 2019), and (ii) the seminal pattern-based bootstrapping algorithm Basilisk (Thelen and Riloff, 2002).

Finally, we tackle the challenge of handling a dataset with a vast majority of multiword expressions, i.e., 98% of our targets are noun compounds (such as *seat heating switch led*). We address the corresponding severe data sparsity by assuming that many domain-specific compounds are compositional (Clouet and Daille, 2014; Hätty et al., 2021) such that our domain-specific model may fall back to information regarding the compound's head (in the example above: the right-most simplex noun in the compound *led*), and thus improve on the data sparsity.

2 Related Work

Domain-specific IR In contrast to domains such as biomedicine and chemistry, "[l]everaging NLP tools in materials science remains in its infancy" (Olivetti et al., 2020, p.4), with main challenges including the development of task- and domain-specific tools to structure and harnes knowledge for material synthesis and material discovery. Common approaches to elicit material mentions from text include unsupervised approaches such as topic modeling (Rani and Kumar, 2021) and LDA (Venugopal et al., 2021), as well as supervised methods for domain-specific NER, often focusing on a single material or material group (Mysore et al., 2017, 2019; Friedrich et al., 2020; Gupta et al., 2022; Nayak and Timmapathini, 2021; O'Gorman et al., 2021). Differently to previous work, we probe PLMs in a cloze-task style setting and compare the results to a similarly unsupervised pattern-based bootstrapping algorithm. While most unsupervised methods are not easily interpretable, our query templates, seed words, patterns and predicted materials are directly accessible.

Prompting and Cloze Query Engineering Recent work has leveraged cloze-task style settings to probe the knowledge that PLMs acquire during pretraining, thus targeting linguistic capabilities (Goldberg, 2019; Ettinger, 2020; Apidianaki and Garí Soler, 2021; Rogers et al., 2021), the understanding of rare words (Schick and Schütze, 2020) and conceptual abstractions (Ravichander et al., 2020), as well as factual and commonsense knowledge (Petroni et al., 2019; Jiang et al., 2021). An emerging strand of research highlights PLM sensitivity to the input in a cloze-task style setting. For example, Elazar et al. (2021) demonstrate that PLMs show deficiencies in prediction consistency when presented cloze-style query paraphrases. Others exploit this flaw by explicitly considering paraphrases of a cloze query in addressing a model's sensitivity to a specific input query to elicit a desired output (Davison et al., 2019; Jiang et al., 2021). Pandia and Ettinger (2021) show that models lack robustness in their ability to harness relevant context information in the face of cloze tasks containing distracting contextual cues. In reverse, priming the model using *trigger tokens* (Shin et al., 2020) and *lexical cues* (Misra et al., 2020) might aid in guiding the model to predict the desired output. Specifically, Shin et al. (2020) propose AUTOPROMPT to develop automatically constructed prompts or patterns to elicit knowledge from pretrained PLMs for a variety of tasks. In contrast to previous work, we semi-automatically construct a variety of prompts to cover a salient set of paraphrases for eliciting plausible materials in a PLM; this allows for meaningful comparison and interpretation of the predictions' quality and the influence of the cloze queries.

Semantic Plausibility While classical distributional models tend to model selectional preferences and thematic fit instead of capturing semantic plausibility (Erk et al., 2010), we have recently seen advances to model plausibility across various dimensions, including *physical* and *abstract* semantic plausibility (Wang et al., 2018; Porada et al., 2019). SOTA models for event plausibility however still rely on straightforward conditional probabilities of co-occurrences as estimated by distributional models (Emami et al., 2021; Porada et al., 2021). Considering our task of learning plausible materials for vehicle components, we go beyond selectionally preferred component materials as predicted with high probabilities, aiming also for less frequently observed cases that are nevertheless plausible.

(Domain-specific) MWEs and Compositionality Multiword expressions (MWEs) are challenging for any natural understanding system, given that MWE meanings are idiosyncratic to some degree, i.e., the meaning of an MWE is not entirely (or even not at all) predictable from the meanings of the constituents (Sag et al., 2002; Reddy et al., 2011; Salehi et al., 2014; Schulte im Walde et al., 2016; Cordeiro et al., 2019; Schulte im Walde and Smolka, 2020). Even though MWEs are ubiquitous not only in general- but also in domain-specific language (Clouet and Daille, 2014; Hätty et al., 2021), up to date only few NLP systems have exploited MWE meaning modules, as in machine translation (Cholakov and Kordoni, 2014; Weller et al., 2014). This study is faced with 98% noun compounds among our domain-specific targets, and we test the compound–head compositionality assumption (e.g., a *seat heating switch led* "is a type of" *led* but an *engine valves rocker arm* "is not a type of" *arm*) to fight the severe MWE-triggered data sparsity.

3 Data

Vehicle Component Dataset As targets for our components, we rely on a set of 7,069 unique component names curated by experts from the vehicle repair domain.[1] A component name may denote a tangible physical component such as *cooling blower*, as well as intangible functional and software components such as *ABS warning lamp function* and *road test*. The dataset comprises 155 single-word components and 6,914 multiword components with up to eight constituents (see Fig. 6 in App. A for the distribution). When applying the assumption of MWE compositionality, we fall back to the right-most constituent word as the MWE head (Altakhaineh, 2019), resulting in a total of 725 different heads across the 6,914 multiword components.

Domain-Specific Corpora We utilize two domain-specific English corpora for extracting materials for vehicle components from text: (i) a domain-specific vehicle repair manual written by domain-experts, containing approx. 800K English tokens (henceforth: DOMAIN) and (ii) a portion of the English Wikidata (henceforth: WIKI). For this, we download a Wikidata dump[2], from which we only keep those 118,154 articles which contain one or more of the 6,914 multiword components in the titles or body (Hätty et al., 2020). For domain-adaptation, we delete sentences that do not contain any of the multiword components, resulting in approx. 225K sentences.

4 PLM Experiments

Our PLM experiments investigate to what extent and how PLMs encode domain-specific component materials when prompted with cloze query variants, as illustrated in Fig. 2: We use the vehicle component dataset to construct a set of 504 cloze query templates for each individual component. We first probe a RoBERTa model without any modifications and gather the top-5 predictions for each component, resulting in a total of >2K predictions. To aggregate these predictions, we experiment with three aggregation methods and determine the top-5 most plausible predictions per component and aggregation method for expert evaluation. In a follow-up experiment, we perform domain-adaptation of RoBERTa and DistilRoBERTa for the vehicle repair domain using our two domain-specific corpora, and compare the probing results of the vanilla and the domain-adapted models. The following subsections describe our PLM experiments in detail.

4.1 Cloze Query Prediction and Processing

Our cloze query handling involves three steps (again, cf. Fig. 2): we first construct a set of cloze query templates, then we aggregate model predictions, and finally we analyze the effect of individual cloze queries to select the most meaningful templates for the final top-5 predictions. As Vanilla RoBERTa model (Liu et al., 2019) (henceforth: Vanilla RB), we draw on the `roberta-base` implementation by `huggingface` (Wolf et al., 2020) without any modifications.

Step 1: Cloze Query Template Construction We develop a set of templates and generate cloze statements for each vehicle component to probe our PLM for plausible component materials: First of all, we define a set of 18 paraphrases typically used to express that a component is made from one or more materials (Davison et al., 2019; Jiang et al., 2021). For this, we start with a set of highly frequent verb relations in our corpus (such as *contain*) and use WordNet synsets (Fellbaum, 1998) to determine additional relevant verbs. We then apply both plural and singular forms of the component nouns with corresponding indefinite articles to define queries (see Table 1). To include prompts that refer to prototypical (and thus presumably highly plausible) materials, we follow Apidianaki and Garí Soler (2021) and integrate the quantifiers {MOST} and {ALL}; we also add the quantifier {MANY}. Furthermore, we define a query

---

[1] The Vehicle Component Dataset and the DOMAIN corpus are provided by Bosch.

[2] `https://dumps.wikimedia.org/enwiki/`, 2.3.2022

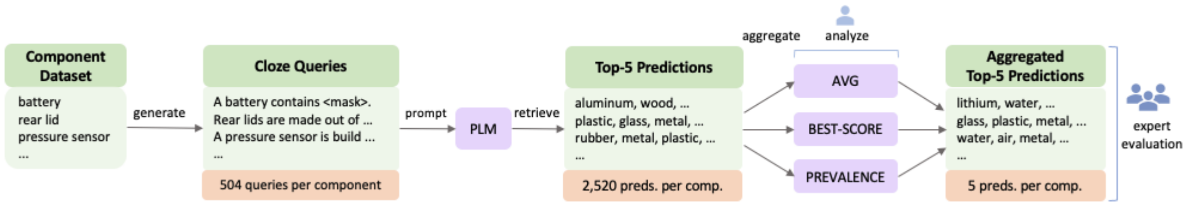


Figure 2: Overview of prediction generation and aggregation, with the numbers in the orange boxes referring to queries/predictions for *one* example component, e.g., *battery*.

Cloze Query Templates [context] + [noun_quant.] + indef. article + component + verb_relation + [adverb] + <mask>.	
SINGULAR	a battery consists of <mask>.
PLURAL	batteries consist of <mask>.
noun_quantifier + PLURAL	most batteries consists of <mask>.
adverb + SINGULAR	a battery usually consists of <mask>.
adverb + PLURAL	batteries usually consists of <mask>.
context + SINGULAR	when used in a vehicle, a battery consists of <mask>.
context + PLURAL	when used in a vehicle, batteries consists of <mask>.

Table 1: Cloze query template and filled-in queries for the vehicle component *battery*. SINGULAR and PLURAL are placeholders that are filled in with vehicle component nouns in singular and plural form, respectively.

template element for adverbs referring to typicality, by including the adverbs {USUALLY, GENERALLY, NORMALLY}. Fostering materials that are not necessarily prototypical but still potentially plausible, we use the noun quantifier {SOME} (Apidianaki and Garí Soler, 2021) and the adverbs {POSSIBLY, PLAUSIBLY}. In our domain-specific scenario, humans tend to intuitively restrict the answer space of plausible materials by providing contexts such as "WHEN USED IN A VEHICLE, ...". Accordingly, we follow recent work investigating the impact of so-called *trigger tokens* (Shin et al., 2020) and *lexical cues* (Misra et al., 2020) to predict a related token, and define six limiting context phrases to leverage intuitive human behaviour.

The above procedure constructs 504 cloze queries for each vehicle component such as the input query "*a battery contains <mask>*.", where the plausible material candidate is masked and to be predicted by a PLM. A full overview of query variants is presented in Table 1.

Step 2: Cloze Query Prediction Aggregation For each component, we prompt our PLM with the corresponding 504 cloze queries and obtain the top-5 predicted tokens ranked by probability for each of the queries, resulting in 2,520 predictions per component, see Fig. 2.[3] Given that we want only a small list of highly plausible materials, we experiment with three approaches to aggregate the 2,520 predictions for each component such that the most plausible material candidates are ranked at the top of a component's material list.

- BEST-SCORE probabilities aggregate the most probable PLM-predicted material types from the 2,520 top-5 predictions across all queries. In this way, predictions that the model considers highly probable in a specific query constellation are considered highly plausible, no matter how often they have been predicted.
- AVG probabilities are obtained by summing up the probabilities for each query variant on the material type level and then averaging by the number of queries that predicted that material. This way, probability mass is taken away from materials that were predicted with high probabilities by only one or few queries. The aggregation is thus directed towards materials that are less prominent for individual queries but more pervasive across query variants.
- PREVALENCE refers to a method where the most salient PLM-predicted materials are obtained by ranking only according to how often the material was predicted across all top-5 suggestions of all query variants. Completely ignoring the probabilities in this step accounts for the amount of used cloze queries and provides insight into whether the model leverages semantic information across specific queries.

---

[3] We apply basic post-processing as described in App. B.2.

	AVG	BEST-SCORE	PREVALENCE
Total@5	0.08	0.18	**0.42**
Acc@5	0.36	0.58	**0.86**

Table 2: Results of aggregation methods on top-5 predicted materials for 100 components each, i.e., a total of 500 predictions per method, reporting total@5 and accuracy in %.

After aggregating the final top-5 predictions by using each of the three presented methods, we analyze[4] for 100 components (see App. C for sampling details) whether the top-5 material predictions are plausible. We follow previous work (Ettinger, 2020; Apidianaki and Garí Soler, 2021) and evaluate by accuracy, i.e. the proportion of vehicle components for which a suggested material is among the model's top-5 predictions.

Results are reported in Table 2. The suggested materials ranked by PREVALENCE with over 40% of correct predictions clearly outperform BEST-SCORE and AVG. Especially AVG is observed to promote rather implausible predictions such as *dna*, *food*, or *death*. Based on this preliminary assessment, we use the PREVALENCE method for aggregating the top-5 predictions in our final experiments.

Step 3: Cloze Query Template Selection Using the top-5 material predictions by PREVALENCE, we now investigate the productiveness of query templates for 100 components (see App. C for sampling details) in order to select the most salient ones for our final experiments. Productiveness investigates which queries are *productive*, i.e., contribute a material to the final top-5 materials.

In our first productiveness analysis, we analyze the 68,651 queries that trigger any of the top-5 predictions across all 100 sampled components and all query template variants. Fig. 3 shows the number of queries actively triggering a top-5 prediction in comparison to the overall number of queries that could have potentially been activated (in absolute numbers). Overall, plural variants exhibit a slightly higher productiveness than singular variants.

In our second productiveness analysis, we examine whether some verb relations "do all the work" in comparison to others that are not activated and should therefore be excluded for the final experiments. Fig. 4 shows the relations triggering a top-5 prediction across all query variants. The most productive relations for singular and plural variants are [*comprise, formed of, build with*] and [*made up of, composed of, consist of*], respectively. For all but two cases, query productiveness of plural variants is higher than for the singular equivalents.

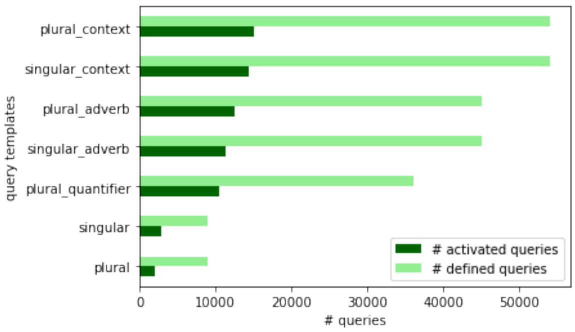


Figure 3: Query template productiveness across the respective defined query template variants.

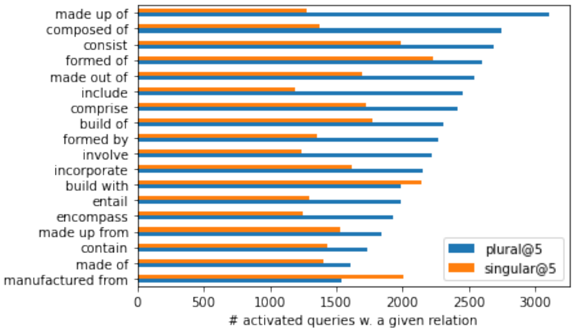


Figure 4: Relation productiveness across singular and plural query template variants.

Our two productiveness analyses suggest that our set of domain-specific query templates represents a well-defined set which is *successful as a whole* in cooperation with a suitable method for aggregating a large number of resulting predictions, and does not include strongly over- or underperforming variants. For our final PLM experiments, we thus leverage the full set of query templates.

4.2 Domain Adaptation of RoBERTa

We experiment with our two domain-specific corpora DOMAIN and WIKI (see §3) to adapt RoBERTa to the vehicle repair domain (henceforth: DOMAIN RB and WIKI RB, respectively). Additionally, we compare RoBERTa to a domain-specific resource-lean model DOMAIN DistilRB, where we adapt DistilRoBERTa (Sanh et al., 2019) to the vehicle repair domain using DOMAIN.

---

[4] The analysis is performed by one of the authors of this paper, who is familiar with the vehicle repair domain. Note that this preliminary assessment is only for comparing aggregation methods, and therefore independent of the final evaluation involving three experts, see Fig. 2.

The domain adaptations draw on the `roberta-base` and `distilroberta-base` implementations from `huggingface` (Wolf et al., 2020); we split our input texts into train (90%) and validation (10%) sets. Following common choices, we set the argument specifying the fraction of tokens to mask to 15%. We train with a learning rate of 2e-5, a weight decay of 0.01, and a batch size of 1,024 (cf. App. B.3 for further details). We train for three epochs and save the best model according to validation set performance. We train one model for each of the aforementioned variants (DOMAIN RB, WIKI RB, and DOMAIN DistilRB). Model results will be presented in §7, in comparison to Basilisk results.

5 Basilisk Experiments: A Classic

In comparison to the PLM experiments, we explore the seminal pattern-based bootstrapped learner Basilisk (Thelen and Riloff, 2002) for learning domain-specific plausible materials for vehicle components. Basilisk is designed to learn high-quality semantic lexicons for one or more categories, provided an unstructured natural language text corpus plus seed words for each semantic category to be learnt. The resource-lean learner leverages extraction patterns representing linguistic contexts, thus exploiting lexico-syntactic structures to capture word meaning.

As the starting step for Basilisk, we select ten seed nouns from the most frequent words in DOMAIN[5] that belong to the semantic class *material*. Since Basilisk depends on extraction patterns to supply contextual support for additional words belonging to the same semantic class, we rely on dependency parsing to create domain-specific syntactic contextual patterns by using the seed words obtained from DOMAIN and only the DOMAIN corpus. We then apply Basilisk as bootstrapping algorithm to select the best-performing patterns for a pattern pool and subsequently fill a candidate word pool with the extractions of the highest-scoring patterns. We retrieve, for example, `SEED*<GDep>:<compound>:<dependent>:alloy` on the basis of which the word `alloy` gets added to the candidate word pool. We stop Basilisk after n bootstrapping rounds and store the retrieved semantic lexicon of n plausible materials for a given vehicle component. To then connect these possible materials with the components, we first process the underlying unstructured data sources and only keep sentences containing at least one component name mention. Second, we filter the obtained sentences with the semantic lexicon and store all component-material candidate matches.

In this way, we generate 396,887 extraction patterns, and use the Basilisk implementation (Thelen and Riloff, 2002) with all corresponding patterns and seed words on the DOMAIN dataset. We limit the candidate word pool size to $n = 200$, and generate possible material candidates for both the DOMAIN and the WIKI datasets.

6 Evaluation

To our knowledge, no gold standard is available for the task of plausible material extraction in the vehicle repair domain. We thus perform an expert evaluation of the quality of predictions for each slot obtained from the vanilla and the domain-adapted RoBERTA models. To compare PLM-based results to resource-leaner methods, we also evaluate results from the pattern-based Basilisk algorithm and DistilRoBERTa.

Evaluation Task For each model, we select and evaluate 100 randomly sampled components,[6] (i) balancing the number of constituents per target component, (ii) providing the full component vs. providing only its head, and (iii) varying the underlying corpus (for Basilisk only). For both 1- and 2-constituent words,[7] we sample 30 components with 20 full/10 head samples with equal amounts from DOMAIN/WIKI. For 3-constituent words, we sample 20 instances with equal amounts of full/head and DOMAIN/WIKI samples. For both 4- and 5-constituent words, we sample 10 head instances with equal amounts of DOMAIN/WIKI samples (cf. App. C for further details).

We obtain the aggregated top-5 predictions for vehicle components from each model. For Basilisk, we sample up to five answer options whenever more than five entries are extracted for a given component. To minimize annotator bias, we compile a set of up to 18 materials for each vehicle component while keeping track of prediction origins. This way, annotators see each evaluated component only once and rate all predictions from all models for

---

[5] Set of seeds: {*water, steel, metal, glass, rubber, plastic, aluminum, copper, polyester, quartz*} In the beginning, we created individual seed word lists for DOMAIN and WIKI, where the latter lacked sufficient quality. Thus, we harness the DOMAIN seed words for both datasets.

[6] We make the list of evaluated components available here.

[7] A 1-constituent word is a simplex word.

Model	IAA	A1::A2	A2::A3	A3::A1
Vanilla RB	0.85	0.89	0.81	0.86
DOM. RB	0.79	0.85	0.74	0.78
WIKI RB	0.81	0.86	0.74	0.83
Basilisk	0.79	0.79	0.74	0.83
DOM. DistilRB	0.82	0.88	0.78	0.81

Table 3: Averaged (left) and pair-wise (right) IAA on material predictions.

each vehicle component without potential unconscious comparison to previously seen predictions from another model. In the majority of cases, the models suggested materials either as a singular or as a plural form (e.g., *metal* and *metals*); for the few cases were we are faced with both versions, we decide to not merge the model predictions and ask the annotators to tick both options if both are considered plausible.

Evaluation Setup We present each component instance with corresponding materials to three experts[8] from the vehicle repair domain. The annotators are asked to rate the plausibility of material predictions for each vehicle component, by ticking the correctly identified materials. They are also provided the options "none of these" if no material is plausible, and "I do not know the answer" if this is the case. Inter-annotator agreement (IAA) is shown in Table 3, with overall average IAA on the left and IAA calculated for each annotator pair on the right. All IAA scores indicate substantial agreement. Further details on the annotation setup and on inter-annotator analyses on both material and component level are provided in App. C.2.

7 Results and Discussion

Table 4 presents results on model performances. On the left, we show accuracy on component level across n annotators. Accuracy is defined as the proportion of components for which at least one of a model's top-5 material predictions is rated plausible by the annotators. On the right, we display accuracy on material level where accuracy is defined as the proportion of plausible material predictions among a model's top-5 predictions, as rated by n expert annotators.[9]

	COMPONENTS		MATERIALS	
Model	$\geqslant$1A	3A	$\geqslant$1A	3A
Vanilla RB	0.87	0.68	0.49	0.24
DOMAIN RB	**0.93**	**0.73**	**0.62**	**0.28**
WIKI RB	0.91	0.66	0.56	0.24
Basilisk	0.73	0.40	0.45	0.14
DOMAIN DistilRB	0.87	0.69	0.53	0.23

Table 4: Model performance. COMPONENT accuracy: proportion of components for which at least one material among a model's top-5 predictions is rated plausible by n expert annotators nA. MATERIAL accuracy: proportion of material predictions among a model's top-5 predictions rated plausible by n expert annotators nA.

Domain-Adaption Approaches: Vanilla, DOMAIN, and WIKI RoBERTa RoBERTa adapted to the domain using a small high-quality corpus (DOMAIN RB) beats all other models with an accuracy of 0.93, and RoBERTa adapted to the domain using a customized portion of the English Wikipedia (WIKI RB) is similarly successful (0.91); in comparison, Vanilla RoBERTa reaches an accuracy of 0.87. With perfect agreement among annotators (i.e., 3A), DOMAIN RB is still able to reach an accuracy of 0.73, in comparison to 0.66 for WIKI RB and 0.68 for Vanilla RB. The top results on material level resemble those for component accuracy: DOMAIN RB (0.62) outperforms all other models, followed by WIKI RB (0.56). In contrast to the component level, Vanilla RB is no longer on par with the smaller but domain-adapted DOMAIN DistilRB, while still outperforming Basilisk. The differences between the three resource-intense approaches (Vanilla RB, DOMAIN RB and WIKI RB) are not significant[10] on component level for nA. On material level, however, differences are significant between PLMs (incl. DOMAIN DistilRB) for $\geqslant$1A, but not for 3A. All approaches significantly outperform the resource-lean Basilisk on both levels for all nA.

As a particular example to illustrate the positive impact of domain adaptation, we inspect results for the vehicle component *engine valves rocker arm* from our introductory example (cf. Fig 1). Vanilla RB results include [*wood, metal, steel, bones, legs*], where the predictions *bones* and *legs* refer to (parts of) an extremity of an animate being instead of a vehicle component. The materials [*wood, metal, steel, joints, aluminium*] as obtained from WIKI RB instead encompass the material *aluminium* as

---

[8] The participating annotators are not affiliated with the company providing the Vehicle Component Dataset.

[9] Material accuracy is naturally lower than component accuracy, as we evaluate five material predictions per component. Hence, achieving perfect agreement for material predictions is more difficult than perfect agreement for components.

[10] Significance tests apply $\chi^2, p < 0.05$.

well as *joints* which refer to connections between body parts as well as vehicle parts. Finally, DOMAIN RB results comprise all plausible materials predicted by the other models, namely [*steel, metal, parts, aluminium, plastic*]. Moreover, the results feature the additional in-domain material *plastic*, while not including out-of-domain predictions such as *wood* anymore. We note that annotators are quite likely to rate one of the top-5 predictions for the same component as plausible, however, material prediction accuracy values indicate that the likeliness to also agree on this specific material drops with an increasing number of annotators.

The findings from these results are twofold. Domain adaptation of a given model results in a clear increase of plausible materials for items within a given domain. Second, if no highly domain-specific and well-curated corpus is available, domain adaptation using WIKI leads to very viable results and therefore represents a strong alternative.

Resource-Lean Approaches Results for the pattern-based algorithm Basilisk significantly underperform all other models for both component (0.73) and material (0.45) accuracy. In contrast, the domain-adapted resource-lean DistilRoBERTa (DOMAIN DistilRB) is on par with Vanilla RB and WIKI RB. Manually analyzing predictions for our example discussed earlier, i.e., the component *engine valves rocker arm*, we find that the Basilisk results [*structures, glass, core, steel, plating*] are potentially within the domain, but often neither a material (*structures, core*) nor rated plausible for the respective target component (*glass, plating*). Results for DOMAIN DistilRB, in contrast, include [*steel, metal, parts, plastic, aluminium*] and basically equal DOMAIN RB predictions.

The comparison of larger vs. leaner and smaller approaches by the examples of the well-established classic Basilisk and the transformer-based DistilRoBERTa model thus clearly demonstrates a preference for the latter.

Compositionality of Domain-specific MWEs Results regarding the assumption of compositionality on domain-specific MWEs are presented in Fig. 5. While accuracy values are lower when using the right-most head instead of the full multiword component (exception: Basilisk), the average difference is only 0.07 and 0.01 for component and material prediction accuracy, respectively. Note, however, that IAA differs especially on material level with an average agreement of 0.44 using full and 0.32 drawing on head components. This indicates that predictions for full component names tend be more clearly connected to this very item than predictions retrieved for a head.

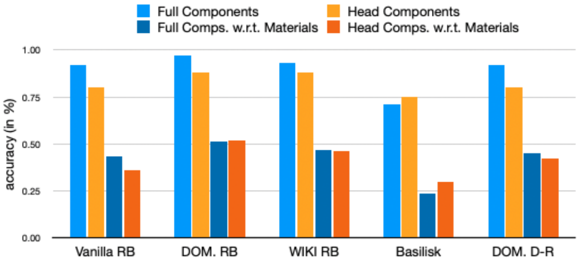


Figure 5: Comparison of full and head component performance on component (full: x/59, head: x/40) and material (full: x/395, head: x/200) accuracy level.

We further analyse model performance regarding the influence of MWE constituent number. The results indicate that specificity beats brevity when learning plausible materials for vehicle components. However, our analysis also point towards a threshold separating a beneficial number of constituents (2) from a number pointing towards a detrimental degree of specificity (3+, cf. App. D.2 for further details).

Our findings suggest that domain-specific multiword components are largely compositional and therefore justify a simplified meaning representation via head constituents.

The Influence of Cloze Queries To shed more light on the impact of using cloze query templates for material prediction, we examine whether the number of queries triggering predictions[11] for a given component correlates with component accuracy. We find moderate-to-strong correlation values for all our PLMs, thus suggesting that a high number of queries yielding the same prediction indicates the prediction's plausibility. We also test whether this correlation is directed and fit a corresponding linear regression model. However, the resulting significant R^2 values, ranging around 0.25, are relatively weak and leave us with a mixed picture regarding a directed relationship between the number of queries and component accuracy (cf. App. D.1 for further details).

When zooming into each template part, we find the adverb GENERALLY outperforming the other ad-

---

[11] An average number of 718 queries is activated for each component across models, with numbers of queries ranging from a minimum of <250 and a maximum of >1.200 queries activated per component.

verbs. Concerning quantifiers, we observe MANY yielding the most activated queries across models, followed by MOST and ALL. Among contexts, we identify WHEN USED IN VEHICLES,... demonstrating best performance. When inspecting the impact of our domain-specific verb relations, we find mostly distinctive groups of verbs accumulating at the extremes of the curve, e.g., *build with, consist of, made up of* and *involve, made of, include* yielding maximum and minimum numbers of plausible predictions across models and singular/plural variants, respectively. While overall observations suggest that templates in the passive voice produce more plausible predictions overall, examples such as *consist of* underline the importance of a well-rounded template set that is *successful as a whole*.

To further explore the impact of designing high-quality query templates for automatic query construction, we perform the following experiment. We take the five most frequent main verbs *make, say, go, use, take* from ENCOW16AX (Schäfer, 2015) that are considered to be *not* directly related to describing what material an item is made of. We construct a set of cloze queries to probe our best-performing domain-adapted PLM DOMAIN RB and aggregate predictions as described in §4.1. We analyze the final top-5 material predictions and find only 9 correct materials. Instead, we observe a small number of types being predicted for many of the sampled components, as well as a small number of component-specific predictions that might be semantically related, but are no material, e.g., top-5 predictions for the component *fuel tank* include [*sense, noise, hydrogen, oil, cold*]. These findings emphasize that using a set that is (i) comprehensive, (ii) domain-specific, and (iii) syntactically diverse is crucial in order to avoid distortion of results.

Exploring Prediction Pool Size Finally, we explore whether aggregating results from a larger pool of predictions obtained from the model benefits our goal of learning all plausible materials for a component. For this, we use our original set of 504 cloze queries per component to obtain the top-10 instead of previously top-5 predicted tokens from the PLM Vanilla RB, resulting in a set of 5040K predictions per component. We aggregate this doubled amount of predictions as explained in §4.1 and analyze the final top-5 predictions as outlined in §4.1. While component accuracy differs by only 1% (in favor of a smaller pool), drawing from a larger pool seems to benefit learning plausible materials with an 5% increase of correct predictions when using top-10 model predictions for aggregation (cf. App. D.3 for details). We attribute these gains to the increase in relevant material predictions that are appropriately aggregated using PREVALENCE, thus displacing, for example, non-relevant material predictions such as *wood* from top-k positions.[12]

8 Conclusion

We tackled the task of learning domain-specific plausible materials for components in the vehicle repair domain from a novel perspective by probing SOTA language models in a cloze task style setting to overcome the lack of annotated datasets. Based on a diverse set of semi-automatically constructed cloze queries combined with a suitable aggregation method, we presented a new method to efficiently extract knowledge regarding vehicle components and their materials as acquired by a PLM. While showing that domain-adaptation using either a small, high-quality or a customized Wikipedia corpus boosts performance, we also demonstrated the power of resource-lean alternatives such as the PLM DistilRoBERTa, and found that the bottleneck for domain-adaptation with respect to our task might not be model size but rather corpus quality and suitability. Finally, we successfully exploited the compositionality assumption for our domain-specific multiword expressions as a way to address data sparsity.

Acknowledgements

The authors would like to thank the Robert Bosch GmbH for providing access to the Vehicle Component Dataset and the DOMAIN corpus. We are grateful to Frank Hartmann, Andreas Körber, Patrick Reiser, and Boris Haselbach for valuable discussions and their support with the Bosch Vehicle Component Dataset.

We also thank Prisca Piccirilli, Neele Falk, and the SemRel group for constructive suggestions and feedback on versions of this work. Helena Schlipf was funded by the Robert Bosch GmbH. Annerose Eichel received funding by the Robert Bosch GmbH and the Hanns Seidel Foundation's Talent Program.

---

[12] Statistics w.r.t. *wood* as a top-1 material prediction with pool size 5: 0.36% (correct: 0.1%); with pool size 10: 0.23% (correct: 0.1%).

Limitations

When learning plausible materials for components in the vehicle repair domain, we build a varied set of query templates to probe PLMs and seek to aggregate obtained predictions in an optimal way. We are, however, fully aware that a set of queries that is optimal for machines is not necessarily the set that also makes perfect sense to humans.

As far as the transfer of the suggested approach to languages other than English is concerned, we call attention to the potential need to adapt the query templates, e.g., when working with languages that allow for more flexibility in word order such as Polish, Turkish, German, Hindi, or Finnish. Further, while it might be difficult to find well-curated domain-specific corpora in some languages, we show that using a customized version of Wikipedia of moderate size (approx. 225K sentences) in a given language represents a very viable alternative. Additionally, researchers could use a multilingual model or an adapter-based approach to navigate in-/output in other or multiple languages.

Our work focuses on RoBERTa as a backbone model, which has been shown to perform well in cloze task style settings. We also carried out initial experiments with newer models such as ELECTRA, not yielding desired results. We also conducted experiments with generative models such as GPT3. However, the output contained a lot of noise that did not aid our goals. Moreover, the focus in this paper is devoted to the development of a novel contribution regarding the prompting and aggregation techniques as well as analyzing results rather than benchmarking a wide variety of models using an already existing method. While experiments with a wider variety of models represent an interesting future task, high-quality evaluation might be a bottleneck due to the availability and cost factor of domain experts.

Regarding our model predictions, we have not yet attempted to detect and organize potential semantic relations between predictions, such as hypernym/hyponym relations between *metal* and *aluminium*. We leave organizing and utilizing such relations as potentially relevant milestone for connecting and tracing symptoms back to a cause for future work. Furthermore, material predictions are not yet categorized along dimensions such as main and auxiliary materials or surface and inner materials, which might be of interest in the vehicle repair and material science domains.

Finally, we would like to point out that all tested models and algorithms struggle with predicting plausible materials for intangible items which might, however, be more prevalent in other domains. Even if a model is able to predict correct material predictions, rating the plausibility of predictions that are not tangible but rather abstract materials such as *data*, *parameters* and *functions* cannot be considered a trivial task for expert annotators and needs corresponding guidance.

Ethics Statement

Two of the resources used in this work, the Vehicle Component Dataset and one of the two domain-specific corpora (DOMAIN) have been kindly provided to us. To provide insight into the nature of the data and foster reproducibility using comparable data, we make a sample of the Vehicle Component Dataset available (cf. §6). We also investigated comparable publicly available alternatives for DOMAIN and showed that leveraging a portion of the English Wikipedia customized to the domain of interest (WIKI) represents a viable substitute in case no custom corpus is available. Note that Wikipedia text content including Wikipedia dumps is licensed under both the Creative Commons Attribution-ShareAlike 3.0 License and the GNU Free Documentation License.

In the context of our evaluation task, we collected plausibility ratings from human participants. For this, the participants were provided an Informed Consent Letter with the name and the contact of the principal investigators; the title, purpose and procedure of the study; risks, benefits and compensation for participating in the study; confirmation of confidential anonymous data handling; and confirmation that participation in the study is voluntary. The Informed Consent Letter was signed by both the participant and the investigators before the participants took part in the study.

We use and adapt PLMs as provided and licensed under the Apache License 2.0 by `huggingface` (Wolf et al., 2020). We acknowledge that material predictions learnt using the outlined approach are a product of unsupervised learning methods which might be prone to error. We point out that predictions should be approved by an expert or flagged otherwise in case they are used in a downstream application to avoid potential risks including harm of objects or safety risks in case of incorrect repair procedures.

References

Abdel Rahman Mitib Altakhaineh. 2019. A cross-linguistic perspective on the Right-Hand Head Rule: The rule and the exceptions. *Linguistics Vanguard*, 5(1):20180033.

Marianna Apidianaki and Aina Garí Soler. 2021. ALL dolphins are intelligent and SOME are friendly: Probing BERT for nouns' semantic properties and their prototypicality. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 79–94, Punta Cana, Dominican Republic.

Kostadin Cholakov and Valia Kordoni. 2014. Better Statistical Machine Translation through Linguistic Treatment of Phrasal Verbs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 196–201, Doha, Qatar.

Elizaveta Loginova Clouet and Béatrice Daille. 2014. Splitting of compound terms in non-prototypical compounding languages. In *Proceedings of the First Workshop on Computational Approaches to Compound Analysis*, pages 11–19, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Silvio Cordeiro, Aline Villavicencio, Marco Idiart, and Carlos Ramisch. 2019. Unsupervised Compositionality Prediction of Nominal Compounds. *Computational Linguistics*, 45(1):1–57.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense Knowledge Mining from Pretrained Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1173–1178, Hong Kong, China.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and Improving Consistency in Pretrained Language Models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Ali Emami, Ian Porada, Alexandra Olteanu, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2021. ADEPT: An Adjective-Dependent Plausibility Task. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7117–7128, Online.

Katrin Erk, Sebastian Padó, and Ulrike Padó. 2010. A Flexible, Corpus-Driven Model of Regular and Inverse Selectional Preferences. *Computational Linguistics*, 36(4):723–763.

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

Christiane Fellbaum, editor. 1998. *WordNet – An Electronic Lexical Database*. Language, Speech, and Communication. MIT Press, Cambridge, MA, USA.

Annemarie Friedrich, Heike Adel, Federico Tomazic, Johannes Hingerl, Renou Benteau, Anika Marusczyk, and Lukas Lange. 2020. The SOFC-Exp Corpus and Neural Approaches to Information Extraction in the Materials Science Domain. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1255–1268, Online.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. *CoRR*, abs/1901.05287.

Tanishq Gupta, Mohd Zaki, N. M. Anoop Krishnan, and Mausam. 2022. MatSciBERT: A Materials Domain Language Model for Text Mining and Information Extraction. *npj Computational Materials*, 8(1):102.

Anna Hätty, Julia Bettinger, Michael Dorna, Jonas Kuhn, and Sabine Schulte im Walde. 2021. Compound or Term Features? Analyzing Salience in Predicting the Difficulty of German Noun Compounds across Domains. In *Proceedings of the 10th Joint Conference on Lexical and Computational Semantics*, pages 252–262, Bangkok, Thailand (online).

Anna Hätty, Dominik Schlechtweg, Michael Dorna, and Sabine Schulte im Walde. 2020. Predicting Degrees of Technicality in Automatic Terminology Extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2883–2889, Online.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Kanishka Misra, Allyson Ettinger, and Julia Rayz. 2020. Exploring BERT's Sensitivity to Lexical Cues using Tests from Semantic Priming. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4625–4635, Online.

Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. 2019. The Materials Science Procedural Text Corpus: Annotating Materials Synthesis Procedures with Shallow Semantic Structures. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 56–64, Florence, Italy.

Sheshera Mysore, Edward Kim, Emma Strubell, Ao Liu, Haw-Shiuan Chang, Srikrishna Kompella, Kevin Huang, Andrew McCallum, and Elsa Olivetti. 2017. Automatically Extracting Action Graphs from Materials Science Synthesis Procedures. *CoRR*, abs/1711.06872.

Anmol Nayak and Hariprasad Timmapathini. 2021. Wiki to automotive: Understanding the distribution shift and its impact on named entity recognition. *CoRR*, abs/2112.00283.

Tim O'Gorman, Zach Jensen, Sheshera Mysore, Kevin Huang, Rubayyat Mahbub, Elsa Olivetti, and Andrew McCallum. 2021. MS-Mentions: Consistently Annotating Entity Mentions in Materials Science Procedural Text. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1337–1352, Online and Punta Cana, Dominican Republic.

Elsa A. Olivetti, Jacqueline M. Cole, Edward Kim, Olga Kononova, Gerbrand Ceder, Thomas Yong-Jin Han, and Anna M. Hiszpanski. 2020. Data-driven materials research enabled by natural language processing and information extraction. *Applied Physics Reviews*, 7(4):041317.

Lalchand Pandia and Allyson Ettinger. 2021. Sorting through the noise: Testing robustness of information processing in pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1583–1596, Online and Punta Cana, Dominican Republic.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language Models as Knowledge Bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2463–2473, Hong Kong, China.

Ian Porada, Kaheer Suleman, and Jackie Chi Kit Cheung. 2019. Can a Gorilla Ride a Camel? Learning Semantic Plausibility from Text. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 123–129, Hong Kong, China.

Ian Porada, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2021. Modeling Event Plausibility with Consistent Conceptual Abstraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1732–1743, Online.

Seema Rani and Mukesh Kumar. 2021. Topic modeling and its applications in materials science and engineering. *Materials Today: Proceedings*, 45:5591–5596. Second International Conference on Aspects of Materials Science and Engineering.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the Systematicity of Probing Contextualized Word Representations: The Case of Hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102, Barcelona, Spain (Online).

Siva Reddy, Diana McCarthy, and Suresh Manandhar. 2011. An Empirical Study on Compositionality in Compound Nouns. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 210–218, Chiang Mai, Thailand.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2021. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword Expressions: A Pain in the Neck for NLP. In *Proceedings of the Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, Mexico.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2014. Using Distributional Similarity of Multi-way Translations to Predict Multiword Expression Compositionality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 472–481, Gothenburg, Sweden.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Roland Schäfer. 2015. Processing and querying large web corpora with the COW14 architecture. Proceedings of the 3rd Workshop on Challenges in the Management of Large Corpora, Lancaster, 20 July 2015, pages 28 – 34, Mannheim.

Timo Schick and Hinrich Schütze. 2020. Rare Words: A Major Problem for Contextualized Embeddings and How to Fix it by Attentive Mimicking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8766–8774.

Sabine Schulte im Walde, Anna Hätty, and Stefan Bott. 2016. The Role of Modifier and Head Properties in Predicting the Compositionality of English and German Noun-Noun Compounds: A Vector-Space Perspective. In *Proceedings of the 5th Joint Conference on Lexical and Computational Semantics*, pages 148–158, Berlin, Germany.

Sabine Schulte im Walde and Eva Smolka, editors. 2020. *The Role of Constituents in Multi-Word Expressions: An Interdisciplinary, Cross-Lingual Perspective*. Number 4 in Phraseology and Multiword Expressions. Language Science Press, Berlin, Germany.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online.

Wilson L. Taylor. 1953. "Cloze Procedure": A New Tool for Measuring Readability. *Journalism Quarterly*, 30(4):415–433.

Michael Thelen and Ellen Riloff. 2002. A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 214–221.

Vineeth Venugopal, Sourav Sahoo, Mohd Zaki, Manish Agarwal, Nitya Nand Gosvami, and N. M. Anoop Krishnan. 2021. Looking through glass: Knowledge discovery from materials science literature using natural language processing. *Patterns*, 2(7):100290.

Su Wang, Greg Durrett, and Katrin Erk. 2018. Modeling Semantic Plausibility by Injecting World Knowledge. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 303–308, New Orleans, Louisiana.

Marion Weller, Fabienne Cap, Stefan Müller, Sabine Schulte im Walde, and Alexander Fraser. 2014. Distinguishing Degrees of Compositionality in Compound Splitting for Statistical Machine Translation. In *Proceedings of the 1st Workshop on Computational Approaches to Compound Analysis*, pages 81–90, Dublin, Ireland.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online.

A Data

Dataset Distribution Fig. 6 shows the distribution of multiword component names in the Vehicle Component Dataset with the absolute number of vehicle components per number of constituent, e.g., the dataset comprises 1,563 multiword components with 2 constituents.

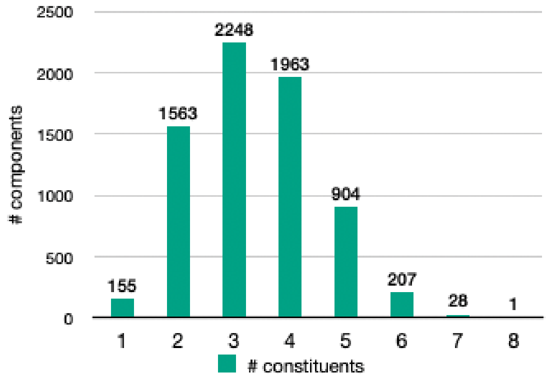


Figure 6: Distribution of multiword components in the Vehicle Component Dataset (in absolute numbers).

B Cloze Query Prediction and Processing

B.1 Cloze Query Templates

Verb Relation Variables consist of, comprise, contain, be formed by, be formed of, be made up of, be made up from, be made of, be made out of, be composed of, be manufactured from, encompass, entail, include, involve, incorporate, be build of, be build with

Cloze Query Statements Table 1 presents an overview of the defined cloze query templates and provides an example of filled-in prompts for the vehicle component *lamp*.

B.2 Post-Processing

PLM-based Predictions Having created a given set of cloze queries, we feed the queries into a PLM and obtain the top-5 predictions. Based on our goals, we apply the following post-processing steps. We lower-case the predictions and filter out numbers, punctuation, 1-character tokens, as well as a list of standard stopwords obtained from `spacy` with nine customized stopwords including non-informative tokens such as *material* and *component*. Additionally, spelling variants such as *aluminum* and *aluminium* are merged. We make sure that a predicted token does not equal the singular or plural form of the full input component.

Basilisk-based Predictions We obtain n predictions for each vehicle component using DOMAIN or WIKI for the Basilisk algorithm, and apply post-processing as described for the PLM predictions.

B.3 PLM and Basilisk Experiments: Computing Infrastructure and Model Information

As Vanilla RoBERTa model, we use the `roberta-base` implementation from `huggingface` (Wolf et al., 2020) that comes with 125M parameters. We adapt this model to the vehicle repair domain. As DistilRoBERTa implementation, we use the `distilroberta-base` implementation from `huggingface` (Wolf et al., 2020) which has 82M parameters. See the official huggingface documentation for further details. For all final experiments, including obtaining predictions from the various models, we use a single NVIDIA RTX A600 GPU. We adapted our models once (DOMAIN RoBERTa, WIKI RoBERTa, and DOMAIN DistilRoBERTa) and used the best model according to validation set performance, i.e. our domain-adapted models are the product of a single fine-tuning run.

As a Basilisk implementation, we use the Java source code as provided by Thelen and Riloff (2002). We use `spaCy` (Honnibal et al., 2020) for text preprocessing and dependency parsing.

C Evaluation

C.1 Sampling Vehicle Components

To test whether the head of a MWE preserves the meaning of the full MWE in our domain-specific data, we compare extracted material predictions using only the head of the MWE vs. all constituents (full). To evaluate results, we apply the following procedure when sampling vehicle components consisting of 2+ words. Full components are sampled as is along with the extracted results. Whenever only the head of a component is used, the results for only the head component are obtained. For the evaluation task, the head component is mapped back to the original full component from the vehicle information dataset. For example, during sampling the head component *sensor* gets mapped to the full 2-constituent component *pressure sensor*. In this way, we can evaluate whether results extracted based on only the head *sensor* are rated as plausible also for the more domain-specific full multiword expression *pressure sensor*.

For the Basilisk-based approach, no results are extracted for one item of the 2-constituent full components and five items of the 3-constituent full components. These results were thus expanded with head component extraction results. In one case, stopword removal leads to an empty prediction list (*alternator start charge current reduction, []*, using DOMAIN). The prediction is back-filled with a randomly sampled token from WIKI-based results for the head component (*reduction*, [*lithium*], using WIKI).

C.2 Expert Evaluation

Evaluation Task Setup The evaluation task was carried out in a remote setting using Google Forms. Annotators were provided detailed written guidelines including example questions and borderline decisions. In case of questions, annotators had the option to contact the authors of the paper. The evaluation could be completed flexibly in the course of a week. Annotators could take as much as time as they needed for completing the evaluation (average time effort: ~1.5 hours). Our three recruited annotators are based in Germany, male, and speak German as their first language. All annotators have 10+ years knowledge and continued education and training in English in general, as well as profound work experience with vehicle repair domain data in German and English. Annotators received a representation allowance for their effort. Each annotator submitted one unique set of answers.

The collected data does not include any information that names or uniquely identifies individual people or offensive content. We check for this by (i) not giving out data to annotators containing such information, and (ii) not asking for any of this information when collecting ratings regarding plausible material candidates for vehicle components, and (iii) manually inspecting collected data to confirm anonymity of annotators and potential other entities. Letters of Consent (cf. §8) are signed before participation and stored separately from the collected ratings.

Evaluation Results Analysis We evaluate a total of 2,342 material predictions from five different models and algorithms on a set of 100 vehicle components from the vehicle repair domain. For 99 components, we collect an average of 5.6 plausible material predictions valid for one or more models. For exactly one component, all annotators agree on *I do not know the answer*.

Pair-wise inter-annotator agreement (IAA) on the level of vehicle components is shown in Table 5. We calculate the percentage of components where two given annotators agree on *at least one* material being plausible (i.e. accuracy on inter-annotator level) or where two given annotators agree that at least one material prediction is **not** plausible. This includes instances where up to four predictions are not plausible as well as instances where all five predicted material options are not rated as plausible (*none of these*). We exclude instances where one or both annotators agree on the answer option *I do not know the answer*.

Model	IAA	A1::A2	A2::A3	A3::A1
Vanilla RB	1.00	1.00	1.00	1.00
DOMAIN RB	1.00	0.99	1.00	1.00
WIKI RB	1.00	1.00	1.00	1.00
Basilisk	0.93	0.97	0.91	0.91
DOM. DistilRB	1.00	1.00	1.00	1.00

Table 5: Proportion of components where two An agree on at least one plausible material or where Ans agree on at least one material prediction being *not* plausible.

D Results and Discussion

D.1 Statistical Analysis: Number of Cloze Queries and Component Accuracy

We statistically analyze the relationship between the number of cloze queries triggering predictions for a given component and the actual accuracy values on component level.[13] For this, we first perform a correlation analysis to see whether the number of activated queries is correlated to component accuracy, i.e., whether 1+ of the predictions for this component is rated plausible by 1+ expert annotator. Results are reported in Table 6 with the investigated models on the left and Pearson's correlation coefficients in the first column. The obtained results indicate that the relationship between the number of activated queries and observed component accuracy is strongly correlated (coefficient >0.5) for the Vanilla and WIKI RB models, and moderately correlated (coefficient >0.3) for DOMAIN RB and DOMAIN DistilRB.

To see whether these correlations are in fact directed, we employ linear regression modeling with the number of queries as predictor and component accuracy as outcome variable. Results are presented in Table 6 with investigated models on

---

[13] As no cloze queries are generated for the Basilisk algorithm, we only investigate PLM results.

Model	Pearson's Coefficients	R^2
Vanilla RB	0.51***	0.26***
DOMAIN RB	0.49***	0.24***
WIKI RB	0.58***	0.34***
DOM. DistilRB	0.40***	0.16***

Table 6: Pearson's correlation coefficients for the number of activated queries and component accuracy. All coefficients are significant with a p-value < 0.01 (***).

the left and R^2 values in the second column. The relatively low R^2 values suggest that the relationship between the number of activated queries and observed *component* ($\neq$*material*) accuracy might more strongly rely on other factors than the number of queries activated for a given component.

D.2 n-Constituent Component Analysis

Results regarding MWEs with n constituents are shown in Fig. 7.

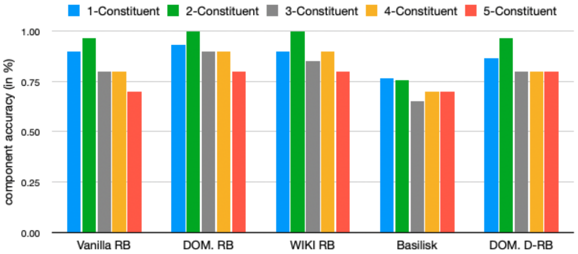


Figure 7: Results regarding n-constituent performance on component accuracy level (in %).

Model performance is presented on component accuracy level. It can be observed that 2-constituent components receive the highest number of plausible material annotations, followed by simplex components, consisting of only one constituent. 4-constituent components outperform the shorter 3-constituent and the longest investigated components, namely 5-constituent components. The results indicate that (domain-)specificity beats brevity when learning plausible materials for vehicle components. However, our findings also suggest that there might be a threshold separating a beneficial number of constituents (2) from a number pointing towards a detrimental degree of specificity (3+).

D.3 Prediction Pool Size Analysis

We present results regarding the size of the pool from which predictions are drawn for aggregation in Table 7. The percentage of correct predictions per top-k predictions are depicted as Hits@1-5. Here, using a larger pool leads to a gain of 18% more plausible materials as the top-1 prediction. While accuracy on component level stays basically the same (top-10 pool: 0.85%, top-5 pool: 0.86%), an increase of 5% can be observed when building the final top-5 predictions from a larger pool of predictions.

	Top-10 Pool	Top-5 Pool
Hits@1	0.54	0.37
Hits@2	0.45	0.56
Hits@3	0.45	0.42
Hits@4	0.50	0.42
Hits@5	0.41	0.31
Comp. Acc@5	0.85	**0.86**
Total Hits@5	**0.47**	0.42

Table 7: Correct predictions (Hits@k) per top-k predictions (x/100), component accuracy (x/100), and total correct predictions per top-5 predictions (Total Hits@5).

Self-Adapted Utterance Selection for Suicidal Ideation Detection in Lifeline Conversations

Zhong-Ling Wang
Department of Computer Science
National Chengchi University
Taipei, Taiwan
`109753106@g.nccu.edu.tw`

Po-Hsien Huang and **Wen-Yau Hsu**
Department of Psychology
National Chengchi University
Taipei, Taiwan
`psyphh@nccu.edu.tw` `hsu@nccu.edu.tw`

Hen-Hsen Huang
Institute of Information Science
Academia Sinica
Taipei, Taiwan
`hhhuang@iis.sinica.edu.tw`

Abstract

This paper investigates a crucial aspect of mental health by exploring the detection of suicidal ideation in spoken phone conversations between callers and counselors at a suicide prevention hotline. These conversations can be lengthy, noisy, and cover a broad range of topics, making it challenging for NLP models to accurately identify the caller's suicidal ideation. To address these difficulties, we introduce a novel, self-adaptive approach that identifies the most critical utterances that the NLP model can more easily distinguish. The experiments use real-world Lifeline transcriptions, expertly labeled, and show that our approach outperforms the baseline models in overall performance with an F-score of 66.01%. In detecting the most dangerous cases, our approach achieves a significantly higher F-score of 65.94% compared to the baseline models, an improvement of 8.9%. The selected utterances can also provide valuable insights for suicide prevention research. Furthermore, our approach demonstrates its versatility by showing its effectiveness in sentiment analysis, making it a valuable tool for NLP applications beyond the healthcare domain.

1 Introduction

The suicide prevention hotline provides a free service for the people with mental issues to have the support on a 24/7 basis. On the other end of the call, a group of counselors who are trained volunteers are waiting for calls online in shifts. As mental health problems are becoming more and more prevalent in society, the difficulty of finding a sufficient number of counselors forms a common challenge that the suicide prevention hotline of many countries in the world is facing. The overwhelmed counselors and the busy phone line make the individuals in urgent need of help unable to get the opportunity to have the counseling. In order to alleviate of demand of suicide counseling, the intelligent conversation system forms a potential direction to study for aiding the hotline service. Fully automated chatbots as counselors providing counseling services are still impractical because the current natural language processing (NLP) technology has not yet been able to construct a reliable chatbot like a well-trained counselor to talk to these people who are mentally unstable and at high risk of suicide.

In this work, we approach the suicide counseling aiding from a more practical aspect. Instead of creating a chatbot to replace the human counselors, our goal is to propose a model for suicidal ideation detection (Ji et al., 2021). During the counseling, our model can be used for monitoring the conversation and identifying the suicide attempt of the caller. Once it is discovered that the caller is at a high risk of suicide or self-harm, more experienced counselors and other resources can immediately intervene at this time to prevent further harm.

Text-based suicidal ideation detection is a hot topic in the NLP area. In addition to medical records, clinical notes (Ji et al., 2021), and suicide notes (Schoene et al., 2021), previous work on textual suicidal ideation detection mainly focused on the social media data, such as tweets (Mishra et al., 2019; Sawhney et al., 2020; MacAvaney et al., 2021), microblogs (Huang et al., 2015), or online forums (Coppersmith et al., 2018; Sawhney et al., 2021; Yao et al., 2020; Alambo et al., 2019). Broadly monitoring the suicidality of users on the social media is no doubt a new way for public health professionals to prevent suicides. However, social media-based suicidal ideation detection can

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1436–1446
May 2-6, 2023 ©2023 Association for Computational Linguistics

cover only a part of cases as not everyone uses the social media and the users with suicidal ideation may not share their troubles publicly.

The scope of this work is entirely different from those based on social media data. We focus on the Lifeline conversations between callers in trouble and the well-trained counselors. Unlike the social media data, which are posted by the users unilaterally, the conversational data are made of the interaction between two or even more parties (e.g., the caller's family member). In the case of suicide prevention hotlines, a conversation is typically made by a caller and a counselor. The well-trained counselor will try to guide the caller to express her or his emotions and distress, revealing rich information for suicide risk assessment. The data used in this work were collected from the real world recordings provided by Taiwan Lifeline International.[1] Psychologists were invited for assessing and labeling the suicidal ideation of the caller in each conversation. To the best of our knowledge, this is a pioneering work of conversation-based suicidal ideation detection.

Unlike social media posts, the conversational data is usually longer and noisier. In our real world data, a conversation contains 515 utterances and 7,981 tokens in average, causing a barrier to models analyzing the content. In the long conversation, not all utterances provide the cue for suicide risk assessment. Instead of analyzing the whole conversation, we focus on key interactions between the caller and the counselor. However, extracting the important utterances from the hotline conversation can be still challenging due to the lack of sufficient training data. For this reason, we propose a novel self-adapted approach to utterance selection. Our idea is to focus on a number of discriminative utterances that provide useful information for NLP models to do the classification. To accomplish this goal, a distant-supervised model for suicidal ideation detection at the utterance level is further built for generating the training data for the utterance selection model.

Our approach to detecting suicidal ideation in phone-call conversations is distinct from reinforcement learning methods that are used to select important features or instances for training the main model. Instead, our distant-supervision methodology leverages information from the conversation level to the utterance level based on a key observation that the risk of suicidal ideation can often be determined by only a few utterances. Our approach also differs from previous hierarchical and multi-grained approaches (Kar et al., 2020; Angelidis and Lapata, 2018) that seek to distill fine-grained information from every smaller piece of the entire input.

In this work, we consider the suicidal ideation labels at the conversation level as the distant label and train a model to identify the discriminative utterances that lead to assessing the suicide risk of the caller. Due to the lack of annotation at the utterance level, we introduce a novel relabeling mechanism that is unique to our distant-supervision method. The result is a final model that can efficiently perform suicidal ideation detection on short, condensed inputs.

Our approach is evaluated on the real-world data labeled by psychologists, showing that the self-adapted utterance selection successfully improves the performance in suicidal ideation detection. We also explore our approach in another domain. Experimental results confirm that our approach is also effective in sentiment analysis, a typical NLP task. The contributions of this work are threefold:

1. We investigate the task of suicidal ideation detection in the phone-call conversations, introducing a new way to aid suicide prevention. Our system can assist the counselors in early detection of callers with high suicidal ideation, thus preventing suicides and reducing the work stress of the counselors.

2. We propose a novel self-adapted utterance selection approach that greatly condenses the conversations and successfully improves the model for dialogue understanding and other NLP tasks required to handle long input.

3. The utterances selected by our approach can also be taken as explanatory information that highlights key interactions between the caller and the counselor.

2 Related Work

This section summarizes the related work from three aspects, including the recent applications of NLP in suicide prevention, the conversation-based detection for other mental diseases.

---

[1] http://www.life1995.org.tw/

2.1 Suicidal Ideation Detection

The previous works on suicidal ideation detection were mainly built for the social media data due to the ease of data collection.

Gaur et al. (2019) investigated the detection of suicide risk based on a dataset consisting of 500 Reddit users. They based on the suicide risk severity lexicon to annotate the posts, and used Columbia Suicide Severity Rating Scale (C-SSRS) to determine the risk of suicidal ideation of each individual. Moreover, Mishra et al. (2019) extracted several kinds of features, such as Parts of Speech (POS), Latent Dirichlet Allocation (LDA), etc. from the tweets as the input of the trainer. Besides, linguistic Inquiry and word count (LIWC) is also a common method to analyze the utterance structure for individuals with mental problems. For instance, Shah et al. (2020) used LIWC for linguistic analysis on Reddit posts. Also, Schoene et al. (2021) performed LICW and presented the learning model to detect suicide notes on microblog, which figured out the syntax patterns of the individuals with mental problems. Despite these features provide part of the insight of utterances, most researchers combined these approaches to deep learning, which let studies more efficient and comprehensive.

In relation to deep learning for text-based data, bidirectional Long Short-Term Memory (Bi-LSTM) is also a common method for detecting mental health problem. Bi-LSTM is an approach that captures the frontend and backend features at every time step. Therefore, it provides much information from the utterances to make a detection. The study of ordinal suicidal ideation detection (Sawhney et al., 2021) designed an architecture, including Bi-LSTM layers, temporal attention layer, ordinal regression layer, to comprehensively analyze the posts from past. Then, they adapted C-SSRS to make an assessment of suicidality. Coppersmith et al. (2018) extracted contextual information between words via a Bi-LSTM layer, which handled with semantic features in the context.

Overall, broadly monitoring the suicidality of users on the social media is undoubtedly a new way to prevent suicide from public health. However, social media-based suicidal ideation detection can only cover a part of cases since not everyone uses the social media and the users with suicidal ideation may not share their troubles publicly.

2.2 Conversation-based of Mental Disease Detection

Dialogue analysis gains attention in the field of mental health care. Rinaldi et al. (2020) aim to predict depression based on the spoken data from interviews, and the multimodal approach (Zhang et al., 2021) that includes the audio features for depression detection in conversations. Other cognitive health issues can also be modeled in conversations (Farzana et al., 2020).

Clinical interviews are a widely utilized method for addressing mental health issues by professionals. However, the highly sensitive nature of much of this data makes it difficult for machine learning models to analyze. Despite these challenges, mental health-focused datasets are considered more consistent and reliable, leading to increased attention in the field of dialogue analysis for mental health care.

Xu et al. (2021) detects the suicidality from a 24-hour online text-based counseling services, which the form of dataset and the research objectives are similar with us. The researchers created the suicide knowledge graph to encode the utterance features as the input of Bi-LSTM layer, and then made a classification of the risk level with multilayer perceptron(MLP). In similar issue, Rinaldi et al. (2020) aim to predict depression in the interviews. They make an evaluation on the Distress Analysis Interview Corpus (DAIC) (Gratch et al., 2014), which contains text transcripts and audio records of interviews for a clinical assessment for depression. In this study, they adopt the BERT (Devlin et al., 2019) model to classify depression or not on the conversation transcripts as the one of the baseline. However, their method is that each prompt of the interview is grouped by the probability distribution and concatenated with its belonging response, then converted to word embeddings using GloVe to transformed to word embedding as the features, and the result is obtained from the decision layer.

Apart from depression, dementia, such as Alzheimer's disease (AD), is also a hot topic of the dialogue based dataset. Farzana et al. (2020) adopt the Pitt corpus as the source of their dialogue act corpus. In addition to feature extraction, they build the classification models using support vector machine, decision tree, logistic regression, and neural networks. The result shows that the neural network model obtains the best score among those methods. Furthermore, Green et al. (2012) use the

Carolina Conversation Collection (CCC) corpus to facilitate conversation between the individuals with AD and their conversational partner. The CCC corpus includes recorded and transcribed conversations between researchers, students and individuals with AD. They aim to resolve the AD-related disfluencies via natural language understanding.

In these studies, the research focused on utterances from both or single side to figure out the context patterns of the mental health problem. Generally, utterances transformed by GloVe, BERT, etc. to word embeddings and the deep neural networks such as Bi-LSTM contribute to semantic analysis and are commonly used in nowadays. Though the results from the neural network models are difficult to observe the patterns of each mental disorder from the computing, we can combine with various features to make an analysis from both human and computing perspectives.

3 Dataset

We obtained the phone call recordings from Taiwan Lifeline International, and the audio recordings were transcribed by the experts with a psychology background. The transcriptions contain both counselors and callers' utterances, which are conversation-based forms. The transcriptions were further annotated by psychologists for assessing the caller's suicide risk level into five grades, including no suicidal ideation, occasional, frequent, ongoing suicide plan, and ongoing suicide attempt. The outlier conversations, such as not mention to suicide, cannot identify, statement not in person, and other situation will be removed. A high kappa value of 0.89 was measured, confirming a strong inter-rater agreement.

Due to the data sparsity, with experts' approval, we further merge the two weak ideation grades (i.e., no suicidal ideation and occasional) into low risk and the two strong ideation grades (i.e., ongoing suicide plan and ongoing suicide attempt) into high risk, resulting in three suicide risk levels. Table 1 summarizes the distribution of instances across different risk levels and presents the conversation length for each risk level in terms of the average number of utterances and tokens.

Risk Level	Overall	Low	Med.	High
# Instances	656	263	227	166
# Utterances	515	426	546	613
# Tokens	7,981	6,965	8,473	8,919

Table 1: Statistics of our dataset

4 Methodology

Given a conversation $\mathbf{x} = (x_1, x_2, ..., x_n)$, where x_i is either the prompt made by the counselor or the response made by the caller repeatedly, an intuitive way to predict the level of the caller's suicidal ideation can be defined in Equation 1.

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in Y} P(\mathbf{y}|\mathbf{x}) \tag{1}$$

where $Y = \{\text{Low Risk, Medium Risk, High Risk}\}$, and $P(\cdot)$ is a model that is trained to estimate the probability of y given $\mathbf{x}$.

As mentioned in Section 1, $\mathbf{x}$ can be very long and noisy, resulting in a poorer performance. Thus, our idea is to predict $\mathbf{y}$ given a short and condensed version of $\mathbf{x}$. That is, we aim to approximate $P(\mathbf{y}|\mathbf{x})$ by $P_C(\mathbf{y}|\hat{\mathbf{x}})$, where $\hat{\mathbf{x}} \subseteq \mathbf{x}$ and $P_C(\cdot)$ makes the prediction based on the subset of $\mathbf{x}$.

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in Y} P(\mathbf{y}|\mathbf{x}) \approx \arg\max_{\mathbf{y}\in Y} P_C(\mathbf{y}|\hat{\mathbf{x}}) \tag{2}$$

In Section 4.1, we present a novel self-adapted utterance selection approach that finds the utterances providing useful information that is easier for the underlying backbone model to capture. Section 4.2 shows the distant-supervised learning employed to train our model for utterance selection. Section 4.3 describes the combination of our framework, and Section 4.4 presents our model in the inference stage. Figure 1(a) and Figure 1(b) illustrate the details of our model in the training and the inference stages, respectively. Figure 1(c) shows the dataflow of our framework in evaluation.

4.1 Self-Adapted Utterance Selection

To extract the sentences that may contain essential suicidal ideation patterns from the conversation, we establish a model $P_S(\cdot)$ for finding $\hat{\mathbf{x}}$. The general form is given in Equation 3.

$$\hat{\mathbf{x}} = \{x_i | x_i \in \mathbf{x} \text{ and } P_S(x_i|\mathbf{x}) > \tau\} \tag{3}$$

where $P_S(\cdot)$ determines the importance of x_i to the whole conversation $\mathbf{x}$. In our case, however, $P_S(\cdot)$ cannot be directly trained in the supervised manner due to the lack of ground-truth. Thus, a novel distant supervised approach is proposed to estimate the importance of each utterance in a conversation.

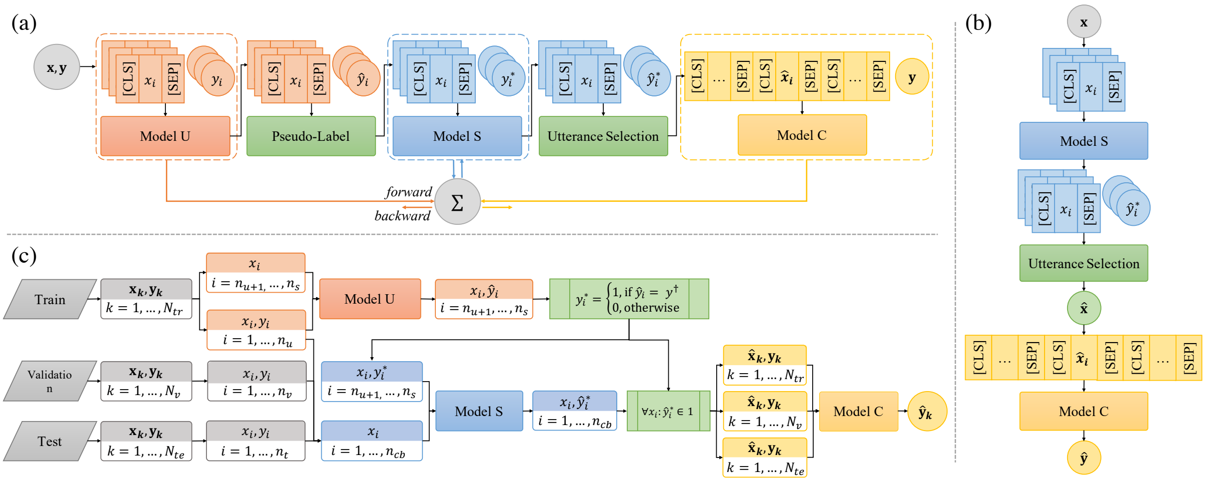


Figure 1: Overview of our approach. The sub-figure (a) shows our model in the training stage, the sub-figure (b) shows our model in the inference stage, and the sub-figure (c) shows the dataflow in evaluation.

Our assumption is that the important utterances are those able to contribute discriminative information for $P_C(\cdot)$ to predict the suicidal ideation. Thus, we aim to extract the utterances indicating suicidal ideation that is easily captured by NLP models. For this reason, we train another model $P_U(\cdot)$ that aims to predict the suicidal ideation at the utterance level.

$$\hat{y}_i = \arg\max_{y_i \in Y} P_U(y_i | x_i, \mathbf{x}) \qquad (4)$$

where $\hat{y}_i$ is the suicidal ideation of the i-th utterance in $\mathbf{x}$. Note that $P_U(\cdot)$ is conditional on not only x_i but also its contextual information from $\mathbf{x}$.

For an utterance x_i from a conversation where the caller with a golden suicide risk level of $\mathbf{y}^\dagger$, we would like to pick up x_i as an important utterance if $\mathbf{y}^\dagger = \hat{y}_i$ because the underlying backbone model can make a consistent prediction based on the information in x_i. In other words, we prepared each training instance (x_i, y_i^*) for $P_S(\cdot)$ by determining the value of y_i^* as Equation 5.

$$y_i^* = \begin{cases} 1, & \text{if } P_U(y_i | x_i, \mathbf{x}) = \mathbf{y}^\dagger \\ 0, & \text{otherwise} \end{cases} \qquad (5)$$

The number of kinds of y_i^* is six for the binary condition at the three risk levels.

4.2 Suicidal Ideation at the Utterance Level

The last problem is how to train the model $P_U(\cdot)$. In our case, $P_U(\cdot)$ cannot be trained in the supervised manner because of the lack of labels at the utterance level. For this reason, we train it using distant supervised learning. For every utterance x_i in a conversation with a golden label of $\mathbf{y}^\dagger$ at the conversation level, we simply assign $y_i^\dagger$, the pseudo label of x_i, with $\mathbf{y}^\dagger$. For example, every utterance from a conversation where the caller was annotated with high suicide risk will be labeled with high suicide risk as well.

4.3 Training of the Whole Framework

The data preprocessing discards utterances that are longer than 29 tokens or shorter than 8 tokens, with an average token count of 15 tokens in reference to a single utterance. Role indicators, such as “A: ” for counselors and “B: ” for callers, are added to the beginning of each utterance. The models $P_S(\cdot)$ and $P_U(\cdot)$ are trained based on the pre-trained BERT model (BERT-BASE-CHINESE) (Devlin et al., 2019). On the other hand, $P_C(\cdot)$ is trained based on the pre-trained Longformer model (LONGFORMER_ZH) (Beltagy et al., 2020) for processing Chinese transcriptions, with an input length of 2,048 tokens. As shown in Figure 1(a) and Figure 1(b), the input of P_C is the concatenation of the selected utterances, while the input of P_S and P_U is a focused utterance and its neighboring utterances separated by `[SEP]`.

As illustrated in Figure 1(a), the three models $P_C(\cdot)$, $P_S(\cdot)$, and $P_U(\cdot)$ are updated in a sequential manner in each epoch. First, $P_U(\cdot)$ is updated to prepare the training data for $P_S(\cdot)$. Then, $P_S(\cdot)$ is updated to select utterances for training $P_C(\cdot)$. The losses of the models $\mathcal{L}_C$, $\mathcal{L}_S$, and $\mathcal{L}_U$ are calculated using cross-entropy, and the total loss $\mathcal{L}$ is their weighted sum which is used for updating the parameters of the three models.

4.4 Inference

Figure 1(b) shows our model in the inference stage. The $P_S(\cdot)$ and $P_C(\cdot)$, which obtain the lowest validation loss value from $P_C(\cdot)$, are saved for the inference stage. The $P_S(\cdot)$ model will shorten the original $\mathbf{x}$ to $\hat{\mathbf{x}}$, and the $P_C(\cdot)$ model predicts the level of the caller's suicidal ideation given $\hat{\mathbf{x}}$. Note that the $P_U(\cdot)$ model, which is built for training $P_S(\cdot)$, is not involved in the inference stage.

5 Experiments

We employ five-fold cross validation, with a development set split 10% from the training set. Significance testing is conducted using McNemar's test. The value of τ is set at 0.5, and the AdamW optimizer is utilized with a total of 40 epochs.

The baseline models include the vanilla BERT/Longformer models in the end-to-end supervised learning, the multi-turn GRU model, the JLPC model proposed by previous work for depression detection in interviews (Rinaldi et al., 2020), the LDA-based utterance selection method, reinforcement learning (RL) model, and the LIWC-based methods applying Gradient Boosting Decision Tree (GBDT) and multilayer perception (MLP) respectively, which are widely-applied in psychological tasks.

In the experiments, the input length with BERT is 512 tokens, and with the Longformer model is 2,048 tokens. The vanilla baseline is fed with the concatenation of all utterances formed as `[CLS]`$x_{counselor}$`[SEP]`x_{caller}`[SEP]`. The GRU model handles each turn in the conversation with the BERT/Longformer encoder separately and combines the information from the tokens with the GRU layer. JLPC model employs the latent prompt categorization on the utterances of counselors and also makes the prediction on first hundred pairs content. The LDA model performs the topic modeling for the conversation and clusters utterances into several groups. In brief, it is equivalent to let LDA model replace $P_U(\cdot)$ and $P_S(\cdot)$ models to do the utterance selection. We select the utterances from each cluster to form the input to the BERT/Longformer model and report the best performance by one of them.

Our novel distant supervised approach is also compared with reinforcement learning (Liu et al., 2019). We simplify our model to $P_S(\cdot)$ and $P_C(\cdot)$ as the framework of reinforcement learning. The $P_S(\cdot)$ as the agent is a binary classification to select the utterances according to the reward. The reward is defined as the difference from the loss of $P_C(\cdot)$ between two continuous epochs.

Different from the models based on Transformer, the LIWC-based model represent the conversation as a 79-dimensional bag-of-word vector, where the value of each dimension indicates the occurrences of the words belonging to the corresponding Chinese version of LIWC (Linguistic Inquiry and Word Count) category.[2] For each conversation, we perform word segmentation and obtain the distribution of LIWC categories. The MLP network is constructed for making the prediction as one of the baselines. GBDT classifiers which combine several weak learners and calculate with the gradient boosting decision tree algorithm are also adopted to accomplish the classification.

6 Results and Discussion

Table 2 shows the overall performances of our approach compared with baselines. The JLPC model making a binary classification for depression in previous work (Rinaldi et al., 2020) does not perform well on the hotline conversations. The GRU model with Longformer encoder produces the worst performance but has the better adaption with BERT encoder. The LIWC-based methods provide the information in different angle from these methods, achieving an F-score of 57.65% with GBDT classifier, higher than most of baselines, but the MLP network dose not perform well on the task. The LDA selection and RL methods figure out some important clues in conversations, which are superior to the vanilla BERT model. By contrast, the Longformer model does not benefit from these selection methods. The vanilla Longformer model can exploit the information from a large portion of the conversation and achieves an F-score of 59.77% in overall. However, our method with either BERT or Longformer significantly outperforms its respective vanilla model in terms of the overall performance at $p = 0.05$. Especially, our method with Longformer achieves the best F-score at overall and every risk levels, superior to all the other models. Particularly, our methodology accomplishes the highest gain on the high suicide risk, which is the most important case to be recognized and intervened. In the rest of this work, we adopt our method with Longformer as the subject for analysis in details.

---

[2] `https://cliwc.weebly.com/`

	Overall			Low Risk			Med. Risk			High Risk		
Method	P	R	F	P	R	F	P	R	F	P	R	F
BERT-based												
BERT	52.02	51.21	51.03	58.64	66.97	62.21	46.10	44.54	45.19	51.31	42.12	45.67
GRU	44.12	42.57	42.35	50.08	57.80	53.43	38.28	39.17	38.12	44.01	30.73	35.49
JLPC	40.45	42.41	39.57	43.79	49.99	44.17	39.62	41.13	39.85	37.93	36.11	34.70
LDA	54.59	52.77	52.45	58.44	63.58	59.99	50.53	47.21	48.16	54.81	47.52	49.21
RL	55.58	52.80	53.14	59.78	63.88	61.53	46.23	50.64	48.23	60.73	43.89	49.68
Longformer-based												
Longformer	60.38	59.93	59.77	69.02	64.19	66.01	55.30	57.68	56.24	56.81	57.91	57.04
GRU	41.03	38.49	36.30	44.51	65.69	52.51	36.97	32.89	33.58	41.62	16.88	22.81
JLPC	39.97	41.03	38.73	46.10	48.32	45.57	41.44	40.28	39.52	32.37	34.50	31.12
LDA	60.44	58.55	57.94	69.11	62.00	64.61	52.11	51.01	50.34	60.10	62.64	58.87
RL	59.21	56.67	56.75	65.18	63.50	63.34	51.29	52.93	51.88	61.15	53.58	55.03
Feature-based												
LIWC-MLP	50.26	49.26	49.48	54.69	51.70	52.82	41.78	44.90	43.16	54.29	51.18	52.47
LIWC-GBDT	58.93	57.28	57.65	63.42	71.47	67.14	49.77	49.83	49.67	63.61	50.53	56.15
Our Approach												
w/ BERT	59.29	58.74	58.53	68.71	62.34	65.19	53.96	57.28	55.12	55.20	56.60	55.28
w/ Longformer	66.58	66.50	**66.01**	73.01	69.54	**70.96**	62.74	60.77	**61.14**	64.00	69.20	**65.94**

Table 2: Performances of the three suicide risk levels and their macro-average, reported in Precision (P), Recall (R), and F-score (F) (%).

6.1 Choice of the Context Window

For an utterance $x_i \in \hat{\mathrm{x}}$ predicted as important by $P_S(\cdot)$, we can consider not only x_i itself but also its context, e.g., the previous and/or the next utterance of x_i. We concatenate x_{i-1} and/or x_{i+1} to x_i with the `[SEP]`. Table 3 compares the results of different context settings. The results indicate that incorporating additional context through utterances does not lead to improvement as the most relevant contextual information is already captured by the $P_S(\cdot)$ selection process.

6.2 Number of Selected Utterances

Table 4 shows how many utterances are extracted by our utterance selection approach compared with Longformer-based methods, including LDA, RL and original data. Based on the number of tokens in condensed conversations, the LDA and RL methods reduce the input length more than ours. Our utterance selection model $P_S(\cdot)$ reduces the average input length to 131 utterances and 2,405 tokens. Concerning the performance of vanilla Longformer baseline with data pre-processing, the model possibly lose the information, resulting in the worse result, an F-score of 55.23%. However, our method outperforms these approaches. Compared with other methods, our utterance selection model $P_S(\cdot)$ has more capability of identifying clues of suicidality from conversations.

6.3 Content of the Selected Utterances

Our approach allows us to understand what types of utterances are selected for the main model $P_C(\cdot)$ to predict the suicidal ideation. Through analyzing instances that were mispredicted by the baseline model but correctly predicted by our approach, we discovered that our $P_S(\cdot)$ extracts important patterns. Since Lifeline provides the service for people to confide their suffering, the negative words, such as "Bad mood", "Painful", "Wanna die", are appeared in every risk level. In contrast, diverse daily life topics such as relationship, school, and work are mentioned at the low risk level. We can figure out what the main trouble occurred to the callers leads to the feeling of sadness or fear. Different from the low risk level, the thought of low self-esteem comes up in their words, such as "I am useless." at medium risk level. Moreover, the strong hopeless, helpless and lonely feelings are revealed in the conversations. The word "death" is

Method	Overall			Low Risk			Med. Risk			High Risk		
	P	R	F	P	R	F	P	R	F	P	R	F
x_i	66.58	66.50	66.01	73.01	69.54	70.96	62.74	60.77	61.14	64.00	69.20	65.94
x_i, x_{i+1}	62.93	62.66	61.74	71.56	69.11	69.72	56.09	50.71	52.93	61.14	68.15	62.57
x_{i-1}, x_i	57.37	56.23	56.35	62.58	66.52	64.40	52.45	56.41	54.12	57.08	45.76	50.54
x_{i-1}, x_i, x_{i+1}	58.09	56.90	56.93	64.64	60.78	62.27	50.45	56.86	53.16	59.19	53.05	55.37

Table 3: With different context windows, the performances of our method at the three suicide risk levels and their macro-average, reported in Precision (P), Recall (R), and F-score (F) (%).

Selection Method	F-score	# Utter.	# Tokens
None	59.77	515	7,981
LDA	57.94	150	2,295
RL	56.75	218	3,930
Ours ($P_S(\cdot)$)	66.01	131	2,405

Table 4: Average lengths of the conversations condensed by different approaches. The macro-average F-scores (%) are reported.

usually in the utterances, but the suicidal ideation is only staying the thoughts.

Unlike above statements, physically suicidal means are mentioned at the high risk level. The callers usually talk about their suicide plan evidently, such as "I bought a helium barrel.", "I took medicine and cut the wrist.". The counselors then ask the caller's location, e.g., "Where are you now?", to ensure their safety. Overall, our approach effectively selects the utterances that are key to the interaction between the caller and the counselor, and provides valuable insights into the caller's suicidal ideation.

6.4 Early Detection

Early risk detection plays an important role in mental issues (Wang et al., 2018). To reduce the burden for each counselor, we expect that the model has the ability to detect the risk of suicidal ideation as soon as possible so that the counselor can take a corresponding action to ensure caller's safety. Figure 2 shows the F-scores of our method to do prediction given different numbers of first utterances in each conversation from the testing data. The black, blue, orange, and red dashed lines represent the F-scores of our method given all the utterances available as input in overall and at the low, the medium, and the high levels, respectively.

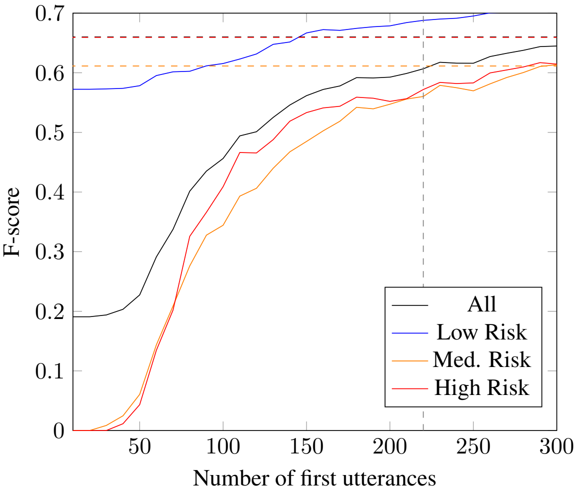


Figure 2: The figure shows the comparison of F-scores for our method in the scenario of early detection. The black, blue, orange, and red dashed lines represent the F-scores of our method with all utterances in overall and at low, medium and high risk levels, respectively.

Generally, the low risk cases are easier to identify. An F-score of 57.24% is achieved with only first 10 utterances being given to our method. The vertically dashed gray line represents the overall F-score achieves 60.67% in the first 220 utterances (i.e., 110 turns), superior to most of baselines. Based on our method framework, $P_S(\cdot)$ can directly detect each utterance whether it should be selected without any other information. The result of the early detection indicates that our method can identify the callers in different needs with a number of turns from the conversation, expected to distributing the suitable resources to assist them. Overall, our method not only contributes to extract the significant utterance patterns but also recognizes the important information during the conversations to accomplish early detection.

7 Applied to Sentiment Analysis

We also explore our approach in sentiment analysis, one of the most-studied application in NLP.

Method	F-score	# Tokens
BERT	86.62	718
Ours w/ BERT	89.63	266
Longformer	95.50	718
Ours w/ Longformer	92.74	371

Table 5: Experimental results of the Polarity dataset. The F-score (%) and the length of input fed into the backbone model are reported.

We evaluate our model for sentiment binary classification on the famous Polarity dataset (Pang and Lee, 2004), which consists of 1,000 positive and 1,000 negative movie reviews. Similar to our experiments, the split of data adopts five-fold cross validation. All methods are trained on the same split for five runs to verify the results.

Table 5 shows the F-scores of the baseline models and our methods. The average tokens fed into the backbone model is also shown. In respect to BERT-based models, our method achieves an F-score of 89.63%, superior to the vanilla BERT model obtaining 86.62%. However, though the performance of our method with Longformer model is improved, it is inferior to the vanilla Longformer model. The average length of the movie reviews in the Polarity dataset is 718 tokens, exceeding the input length of BERT but within the limitation of Longformer. Thus, the vanilla Longformer model is capable to handle full information from the input, while the BERT model can only exploit the information from the first 512 input tokens. Our method with BERT reduces the input length to 266 tokens in average, meeting the limitation of the BERT model and effectively selecting key sentences to enhance the performance.

These results support the key premise of our approach that by dramatically reducing the input length, our method can significantly improve the performance of the underlying model when the input exceeds its capacity. Furthermore, the results highlight the broader potential of our approach for other NLP tasks on lengthy and noisy data.

8 Conclusion

Suicidal ideation detection is one of the core interests in the domain of intelligent healthcare. This work explores the task on a new kind of data, the Lifeline conversations, and presents a novel self-adapted approach that condenses the long and noisy conversation for alleviating the model to identify the suicidal risk level of the caller. Experimental results on real-world data show our approach is not only significantly superior to existing models but also capable of applying to other domains. Our approach is also effective in the scenario of early detection, a practical issue for suicide prevention.

With the advancement of large language models such as ChatGPT, the development of intelligent mental health chatbots is becoming a reality in the near future. Our detection model can be integrated into these chatbots to trigger an emergency response if a high suicide risk level is detected.

Limitations

The Lifeline conversations are highly sensitive. Under the extreme circumstance where the callers suffer from suicidal ideation, it is inappropriate to ask them to consent to share their recordings publicly. For this reason, it is hardly to release a public dataset for the research community, resulting a barrier to reproducibility.

In this work, all the voice data were transcribed into spoken data by human. To launch a fully-automatic system, the transcription should be performed by machine, and more noises will be introduced by the speech recognition model. We will explore the voice-based approach to conversation-based suicidal ideation detection in the future.

Ethical Considerations

The data collected for this study is highly sensitive and contains personal information of the callers. To protect their privacy, all personal identifiable information such as names, ages, addresses, phone numbers, and places of work were removed during the transcription process. The data is securely stored and access to it is restricted to only authorized personnel. This study was reviewed and approved by the Human Research Institutional Review Board (IRB) at National Chengchi University[3] with the case number NCCU-REC-202102-006.

Acknowledgements

This work is partially supported by National Science and Technology Council, Taiwan, under grants 109-2222-E-001-004-MY3 and 110-2410-H-004-111-MY2. Thank you very much to the three anonymous reviewers and the meta-reviewer for your useful suggestions.

---

[3] http://rec.nccu.edu.tw

References

Amanuel Alambo, Manas Gaur, Usha Lokala, Ugur Kursuncu, Krishnaprasad Thirunarayan, Amelie Gyrard, Amit Sheth, Randon S Welton, and Jyotishman Pathak. 2019. Question answering for suicide risk assessment using reddit. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pages 468–473. IEEE.

Stefanos Angelidis and Mirella Lapata. 2018. Multiple instance learning networks for fine-grained sentiment analysis. *Transactions of the Association for Computational Linguistics*, 6:17–31.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Glen Coppersmith, Ryan Leary, Patrick Crutchley, and Alex Fine. 2018. Natural language processing of social media as screening for suicide risk. *Biomedical informatics insights*, 10:1178222618792860.

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

Shahla Farzana, Mina Valizadeh, and Natalie Parde. 2020. Modeling dialogue in conversational cognitive health screening interviews. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1167–1177, Marseille, France. European Language Resources Association.

Manas Gaur, Amanuel Alambo, Joy Prakash Sain, Ugur Kursuncu, Krishnaprasad Thirunarayan, Ramakanth Kavuluru, Amit Sheth, Randy Welton, and Jyotishman Pathak. 2019. Knowledge-aware assessment of severity of suicide risk for early intervention. In *The World Wide Web Conference*, WWW '19, page 514–525, New York, NY, USA. Association for Computing Machinery.

Jonathan Gratch, Ron Artstein, Gale Lucas, Giota Stratou, Stefan Scherer, Angela Nazarian, Rachel Wood, Jill Boberg, David DeVault, Stacy Marsella, David Traum, Skip Rizzo, and Louis-Philippe Morency. 2014. The distress analysis interview corpus of human and computer interviews. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3123–3128, Reykjavik, Iceland. European Language Resources Association (ELRA).

Nancy Green, Curry Guinn, and Ronnie Smith. 2012. Assisting social conversation between persons with Alzheimer's disease and their conversational partners. In *Proceedings of the Third Workshop on Speech and Language Processing for Assistive Technologies*, pages 37–46, Montréal, Canada.

Xiaolei Huang, Xin Li, Tianli Liu, David Chiu, Tingshao Zhu, and Lei Zhang. 2015. Topic model for identifying suicidal ideation in Chinese microblog. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 553–562, Shanghai, China.

Shaoxiong Ji, Shirui Pan, Xue Li, Erik Cambria, Guodong Long, and Zi Huang. 2021. Suicidal ideation detection: A review of machine learning methods and applications. *IEEE Transactions on Computational Social Systems*, 8(1):214–226.

Sudipta Kar, Gustavo Aguilar, Mirella Lapata, and Thamar Solorio. 2020. Multi-view story characterization from movie plot synopses and reviews. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5629–5646, Online. Association for Computational Linguistics.

Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. 2019. Reinforced training data selection for domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1957–1968, Florence, Italy. Association for Computational Linguistics.

Sean MacAvaney, Anjali Mittu, Glen Coppersmith, Jeff Leintz, and Philip Resnik. 2021. Community-level research on suicidality prediction in a secure environment: Overview of the CLPsych 2021 shared task. In *Proceedings of the Seventh Workshop on Computational Linguistics and Clinical Psychology: Improving Access*, pages 70–80, Online.

Rohan Mishra, Pradyumn Prakhar Sinha, Ramit Sawhney, Debanjan Mahata, Puneet Mathur, and Rajiv Ratn Shah. 2019. SNAP-BATNET: Cascading author profiling and social network graphs for suicide ideation detection on social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 147–156, Minneapolis, Minnesota.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Alex Rinaldi, Jean Fox Tree, and Snigdha Chaturvedi. 2020. Predicting depression in screening interviews from latent categorization of interview prompts. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7–18, Online.

Ramit Sawhney, Harshit Joshi, Saumya Gandhi, and Rajiv Ratn Shah. 2020. A time-aware transformer based model for suicide ideation detection on social media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7685–7697, Online.

Ramit Sawhney, Harshit Joshi, Saumya Gandhi, and Rajiv Ratn Shah. 2021. Towards ordinal suicide ideation detection on social media. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 22–30.

Annika Marie Schoene, Alexander Turner, Geeth Ranmal De Mel, and Nina Dethlefs. 2021. Hierarchical multiscale recurrent neural networks for detecting suicide notes. *IEEE Transactions on Affective Computing*, pages 1–1.

Faisal Muhammad Shah, Farsheed Haque, Ragib Un Nur, Shaeekh Al Jahan, and Zarar Mamud. 2020. A hybridized feature extraction approach to suicidal ideation detection from social media post. In *2020 IEEE Region 10 Symposium (TENSYMP)*, pages 985–988.

Yu-Tseng Wang, Hen-Hsen Huang, and Hsin-Hsi Chen. 2018. A neural network approach to early risk detection of depression and anorexia on social media text. In *CLEF (Working Notes)*.

Zhongzhi Xu, Yucan Xu, Florence Cheung, Mabel Cheng, Daniel Lung, Yik Wa Law, Byron Chiang, Qingpeng Zhang, and Paul S.F. Yip. 2021. Detecting suicide risk using knowledge-aware natural language processing and counseling service data. *Social Science & Medicine*, 283:114176.

Hannah Yao, Sina Rashidian, Xinyu Dong, Hongyi Duanmu, Richard N Rosenthal, and Fusheng Wang. 2020. Detection of suicidality among opioid users on reddit: Machine learning–based approach. *J Med Internet Res*, 22(11):e15293.

Pingyue Zhang, Mengyue Wu, Heinrich Dinkel, and Kai Yu. 2021. Depa: Self-supervised audio embedding for depression detection. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 135–143.

Can Pretrained Language Models (Yet) Reason Deductively?

Zhangdie Yuan[◇*], **Songbo Hu**[♠*], **Ivan Vulić**[♠], **Anna Korhonen**[♠], **Zaiqiao Meng**[♣♠†]
[◇]Department of Computer Science and Technology, University of Cambridge
[♠]Language Technology Lab, University of Cambridge
[♣]School of Computing Science, University of Glasgow
[◇♠]`{zy317,sh2091,iv250,alk23}@cam.ac.uk`
[♣]`zaiqiao.meng@glasgow.ac.uk`

Abstract

Acquiring factual knowledge with Pretrained Language Models (PLMs) has attracted increasing attention, showing promising performance in many knowledge-intensive tasks. Their good performance has led the community to believe that the models do possess a modicum of reasoning competence rather than merely memorising the knowledge. In this paper, we conduct a comprehensive evaluation of the learnable *deductive* (also known as explicit) *reasoning capability* of PLMs. Through a series of controlled experiments, we posit two main findings. (i) PLMs inadequately generalise learned logic rules and perform inconsistently against simple adversarial surface form edits. (ii) While the deductive reasoning fine-tuning of PLMs does improve their performance on reasoning over unseen knowledge facts, it results in catastrophically forgetting the previously learnt knowledge. Our main results suggest that PLMs cannot yet perform reliable deductive reasoning, demonstrating the importance of controlled examinations and probing of PLMs' deductive reasoning abilities; we reach beyond (misleading) task performance, revealing that PLMs are still far from robust reasoning capabilities, even for simple deductive tasks.

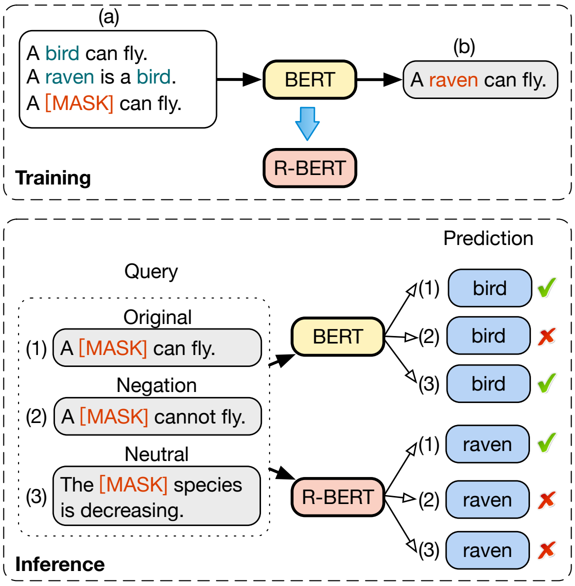


Figure 1: Training and inference for deductive reasoning. Given the explicit premises (a), the input BERT model is trained to get transformed into a reasoner R-BERT model by deductively predicting a previously unseen conclusion (b). This inference process requires R-BERT to understand factual knowledge and interpret rules (e.g. taxonomic relations), intervening directly in the deduction process.

1 Introduction

Pretrained Language Models (PLMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have orchestrated tremendous progress in NLP across a large variety of downstream applications. For knowledge-intensive tasks in particular, these large-scale PLMs are surprisingly good at memorising factual knowledge presented in pretraining corpora (Petroni et al., 2019; Jiang et al., 2020b) and infusing knowledge from external sources (Wang et al., 2021a; Zhou et al., 2022, *among others*), demonstrating their effectiveness in learning and capturing knowledge.

Automatic reasoning, a systematic process of deriving previously unknown conclusions from given formal representations of knowledge (Lenat et al., 1990; Newell and Simon, 1956), has been a long-standing goal of AI research. In the NLP community, a modern view of this problem (Clark et al., 2020), where the formal representations of knowledge are substituted by the natural language statements, has recently received increasing attention,[1] yielding multiple exploratory research directions: mathematical reasoning (Rabe et al., 2021), symbolic reasoning (Yang and Deng, 2021), and com-

---

[*] Indicates equal contribution.
[†] Corresponding author.

[1] Following Clark et al. (2020), we also define natural language rules as linguistic expressions of conjunctive implications, $condition[\wedge condition]^* \rightarrow conclusion$, with the semantics of logic programs with negations (Apt et al., 1988).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1447–1462
May 2-6, 2023 ©2023 Association for Computational Linguistics

monsense reasoning (Li et al., 2019). Impressive signs of progress have been reported in teaching PLMs to gain reasoning ability rather than just memorising knowledge facts (Kassner et al., 2020; Talmor et al., 2020), suggesting that PLMs could serve as effective reasoners for identifying analogies and inferring facts not explicitly/directly seen in the data (Kassner et al., 2020; Ushio et al., 2021).

In particular, *deductive reasoning*[2] is one of the most promising directions (Sanyal et al., 2022; Talmor et al., 2020; Li et al., 2019). By definition, deduction yields valid conclusions, which must be true given that their premises are true (Johnson-Laird, 1999). In the NLP community, given all the premises in natural language statements, some large-scale PLMs have shown to be able to deductively draw appropriate conclusions under proper training schemes (Clark et al., 2020; Talmor et al., 2020). Figure 1 shows an example of the training and inference processes of deductive reasoning.

Despite promising applications of PLMs, some recent studies have pointed out that they could only perform a shallow level of reasoning on textual data (Helwe et al., 2021). Indeed, PLMs can be easily affected by mispriming (Misra et al., 2020) and still hardly differentiate between positive and negative statements (i.e., the so-called *negation issue*) (Ettinger, 2020). However, given that some evidence suggests that PLMs can learn factual knowledge beyond mere rote memorisation (Heinzerling and Inui, 2021) and their limitations (Helwe et al., 2021), it is natural to ask, "*Can the current PLMs potentially serve as reliable deductive reasoners over factual knowledge?*" To answer it, as the main contribution of this work, we conduct a comprehensive experimental study on testing the learnable deductive reasoning capability of the PLMs.

In particular, we test various reasoning training approaches on two knowledge reasoning datasets. Our experimental results indicate that such deductive reasoning training of the PLMs (e.g., BERT and RoBERTa) yields strong results on the standard benchmarks, but it inadequately generalises learned logic rules to unseen cases. That is, they perform inconsistently against simple surface form perturbations (e.g., simple synonym substitution, paraphrasing or negation insertion), advocating a careful rethinking of the details behind the seemingly flawless empirical performance of deductive reasoning using the PLMs. We hope our work will inspire further research on probing and improving the deductive reasoning capabilities of the PLMs. Our code and data are available online at `https://github.com/cambridgeltl/deductive_reasoning_probing`.

2 Related Work

Knowledge Probing, Infusing, and Editing with PLMs. PLMs appear to memorise (world) knowledge facts during pretraining, and such captured knowledge is useful for knowledge-intensive tasks (Petroni et al., 2019, 2021). A body of recent research has aimed to (i) understand how much knowledge PLMs store, i.e., *knowledge probing* (Petroni et al., 2019; Meng et al., 2022); (ii) how to inject external knowledge into them, i.e., *knowledge infusing* (Wang et al., 2021b; Meng et al., 2021); and (iii) how to edit the stored knowledge, i.e. *knowledge editing* (De Cao et al., 2021). In particular, De Cao et al. (2021) have shown that it is possible to modify a single knowledge fact without affecting all the other stored knowledge. However, some empirical evidence suggests that existing PLMs generalise poorly to unseen sentences and are easily misled (Kassner and Schütze, 2020).[3] Moreover, this body of research focuses only on investigating how to *recall or expose* the factual and commonsense knowledge that has been encoded in the PLMs, rather than exploring their capabilities of deriving previously unknown knowledge via *deductive reasoning*, as done in this work.

Knowledge Reasoning with PLMs. In recent years, PLMs have also achieved impressive progress in *knowledge reasoning* (Helwe et al., 2021). For example, PLMs can infer a conclusion from a set of knowledge statements and rules (Talmor et al., 2020; Clark et al., 2020), with both the knowledge and the rules being mentioned explicitly and linguistically in the model input. Some generative PLMs, such as T5 (Raffel et al., 2020), are even able to generate natural language proofs that support implications over logical rules expressed in natural language (Tafjord et al., 2021). In particular, some large PLMs, such as LaMDA (Thoppilan et al., 2022), have been shown to be able to conduct multi-step reasoning under the chain

---

[2] This type of reasoning is also often referred to as *explicit reasoning* in the literature (Broome, 2013; Aditya et al., 2018).

[3] For instance, if we add the *talk* token into the statement "Birds can [MASK]." (i.e. "Talk. Birds can [MASK]."), the PLM might be misled by the added token and predict *talk* rather than the originally predicted *fly* token (Kassner and Schütze, 2020).

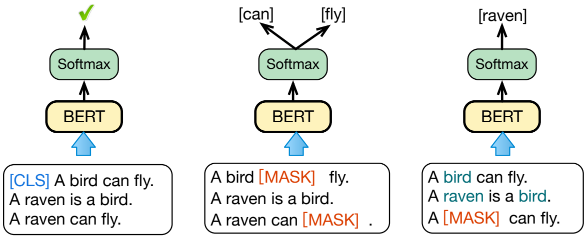
of thought prompting (Wei et al., 2022) or proper simple prompting template (Kojima et al., 2022). Although the generated 'reasoning' statements potentially benefit some downstream tasks, there is currently no evidence that the statements are generated via deductive reasoning, rather than obtained via pure memorisation. Generative reasoning models are difficult to evaluate since this requires huge effort of manual assessment (Bostrom et al., 2021).

Although some research has demonstrated that PLMs can learn to effectively perform inference which involves taxonomic and world knowledge, chaining, and counting (Talmor et al., 2020), preliminary experiments on a single test set in more recent research have revealed that fine-tuning PLMs for editing knowledge might negatively affect the previously acquired knowledge (De Cao et al., 2021). Our work performs systematic and controlled examinations of the deductive reasoning capabilities of PLMs and reaches beyond (sometimes misleading) task performance.

3 Deductive Reasoning

What is Deductive Reasoning? Psychologists define reasoning as a process of thought that yields a conclusion from precepts, thoughts, or assertions (Johnson-Laird, 1999). Three main schools describe what people may compute to derive this conclusion: relying on factual knowledge (Anderson, 2014; Newell, 1990), formal rules (Braine, 1998; Braine and O'Brien, 1991), mental models (Johnson-Laird, 1983), or some mixture of them (Falmagne and Gonsalves, 1995). Our experimental study focuses on a 'computational' aspect of reasoning — namely, whether computational PLMs for reasoning inadequately generalise learned logic rules and perform inconsistently against simple adversarial reasoning examples.

We investigate deductive reasoning in the context of NLP and neural PLMs. In particular, the goal of this deductive reasoning task is to train a PLM (e.g. BERT) over some reasoning examples (each with a set of premises and a conclusion) to become a potential reasoner (e.g. R-BERT as illustrated in Figure 1). Then, the trained reasoner can be used to infer deductive conclusions consistently over explicit premises, where the derived conclusions are usually unseen during the PLM pretraining/training. This inference process requires the underlying PLMs to understand factual knowledge and interpret rules intervening in the deduction process. In this paper, we only focus on the encoder-based PLMs (e.g. BERT and RoBERTa) as they can be evaluated under more controllable conditions and scrutinised via automatic evaluation. In particular, we investigate two task formulations of the deductive reasoning training: **1)** *classification-based* and **2)** *prompt-based reasoning*, as follows.



(a) CLS-BERT **(b) MLM-BERT** **(c) Cloze-BERT**

Figure 2: Different reasoning training approaches.

3.1 Classification-based Reasoning

The classification-based approach formulates the deductive reasoning task as a sequence classification task. Let $\mathcal{D} = \{\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, \cdots, \mathcal{D}^{(n)}\}$ be a reasoning dataset, where n is the number of examples. Each example $\mathcal{D}^{(i)} \in \mathcal{D}$ contains a set of premises $\mathcal{P}^{(i)} = \{\mathbf{p}_1^{(i)}, \mathbf{p}_2^{(i)} \dots \mathbf{p}_j^{(i)}\}$, a hypothesis $\mathbf{h}^{(i)}$, and a binary label $\mathbf{l}^{(i)} \in \{0, 1\}$. A classification-based reasoner takes the input of $\mathcal{P}^{(i)}$ and $\mathbf{h}^{(i)}$, then outputs a binary label $\mathbf{l}^{(i)}$ indicating the faithfulness of $\mathbf{h}^{(i)}$, given that $\mathcal{P}^{(i)}$ is hypothetically factual.

The goal of the classification-based reasoning training is to build a statistical model parameterised by θ to characterise $P_\theta(\mathbf{l}^{(i)}|\mathbf{h}^{(i)}, \mathcal{P}^{(i)})$. Those PLMs built on the transformer encoder architecture, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), can be used as the backbone of such a classification-based reasoner. Figure 2(a) shows an example of using the BERT model to train a classification-based reasoner (CLS-BERT). In particular, given a training example $\mathcal{D}^{(i)} = \{\mathbf{l}^{(i)}, \mathbf{h}^{(i)}, \mathcal{P}^{(i)}\}$, the BERT model is trained to predict the hypothesis label by encoding $[\mathbf{h}^{(i)}; \mathcal{P}^{(i)}]$ and computing $P_\theta(\mathbf{l}^{(i)}|\mathbf{h}^{(i)}, \mathcal{P}^{(i)})$. To do so, the contextualised representation of the '`[CLS]`' token is subsequently projected down to two logits and passed through a softmax layer to form a Bernoulli distribution indicating that a hypothesis is *true* or *false*.

3.2 Prompt-based Reasoning

Deductive reasoning can also be approached as a cloze-completion task by formulating a valid conclusion as a cloze test. Specifically, given a reasoning example, i.e., $\mathcal{D}^{(i)}$ with its premises $\mathcal{P}^{(i)}$, and a cloze prompt $\mathbf{c}^{(i)}$ (e.g. *"A* `[MASK]` *can fly"*), instead of predicting a binary label, this cloze-completion task is to predict the masked token $\mathbf{a}^{(i)}$ (e.g. *raven*) to the cloze question $\mathbf{c}^{(i)}$.

The BERT-based models have been widely used in the prompt-based reasoning tasks (Helwe et al., 2021; Liu et al., 2022), by concatenating the premises and the prompt as input and predicting the masked token based on the bidirectional context. In general, there are two training objectives for the prompt-based reasoning task, i.e., the mask language modelling (MLM) and task-specific (cloze-filling) objectives. For MLM, the given PLMs are trained over the reasoning examples using their original pretraining MLM objective to impose deductive reasoning ability; see Figure 2(b) for an example of the BERT reasoner MLM-BERT. For the cloze-filling objective, the PLMs are trained with a task-specific cloze filling objective. As illustrated in Figure 2(c), Cloze-BERT is trained to predict the masked token in the cloze prompt, by computing the probability $P_\theta(\mathbf{a}^{(i)}|\mathbf{c}^{(i)}, \mathcal{P}^{(i)})$. We note that, unlike the original pretraining MLM objective where 15% tokens of the input are masked randomly, the cloze-filling objective only masks the answer token $\mathbf{a}^{(i)}$ in the cloze prompt $\mathbf{c}^{(i)}$.

This prompt-based reasoning task matches the mask-filling nature of BERT. In this way, we can probe the native reasoning ability of BERT without any further fine-tuning and evaluate the contribution of reasoning training to the PLMs' reasoning ability. Foreshadowing, our experimental results in Section 5 indicate that reasoning training impacts the model both positively and negatively.

4 Experiments and Results

Recent PLMs have shown surprisingly near-perfect performance in deductive reasoning (Zhou et al., 2020). However, we argue that high performance does not mean PLMs have mastered reasoning skills. To validate this, we run controlled experiments to examine whether PLM-based reasoners genuinely understand the natural language context, produce conclusions robustly against lexical and syntactic variance in surface forms, and apply learned rules to unseen cases.

4.1 Datasets

Two datasets are used to examine the PLM-based reasoners, namely, the Leap of Thought (**LoT**) dataset (Talmor et al., 2020) and the WikiData (**WD**) dataset (Vrandecic and Krötzsch, 2014).

LoT was originally proposed for conducting the classification-based reasoning experiments for deductive reasoning (Talmor et al., 2020) and has been used as a standard (and sole) benchmark to probe the deductive reasoning capabilities of PLMs (Tafjord et al., 2021; Helwe et al., 2021). This dataset is automatically generated by prompting knowledge graphs, including ConceptNet (Speer et al., 2017), WordNet (Fellbaum, 1998) and WikiData (Vrandecic and Krötzsch, 2014). LoT contains 30,906 training instances and 1,289 instances for each validation and testing set. Each data point in LoT also contains a set of distractors that are similar but irrelevant to deriving the conclusion.

For the prompt-based reasoning task, we can reformulate the LoT dataset to fit our cloze-completion task. Instead of having a set of premises $\mathcal{P}$, a hypothesis $\mathbf{h}$, and a binary label $\mathbf{l}$, we rewrite the hypothesis in LoT into a cloze $\mathbf{c}$ and the answer $\mathbf{a}$ (e.g. *A raven can fly.* → *A [MASK] can fly.*). Note that we only generate those cloze questions on the positive examples. Consequently, the results across these two tasks are not directly comparable.

The **WD** dataset is an *auxiliary* reasoning dataset which we generated and extracted from Wikidata5m (Wang et al., 2021b). Similar to previous work (Petroni et al., 2019; Talmor et al., 2020), we converted a set of knowledge graph triples into linguistic statements using manually designed prompts. The full description of the dataset construction is provided in Appendix C. The final WD dataset contains 4,124 training instances, 413 validation instances, and 314 test instances. WD only contains positive examples: therefore, we only use this dataset for the cloze-completion task.

4.2 Adversarial Probing

Previous work demonstrates that PLMs can achieve near-perfect empirically results in reasoning tasks. For example, RoBERTa-based models record a near-perfect accuracy of 99.7% in the deductive reasoning task on LoT (Talmor et al., 2020). However, another recent study shows that in some natural language inference benchmarks, PLMs are still not robust to the negation examples (Hossain et al., 2020), while humans can handle negations with

Perturbation	Premises	Conclusion	Valid
Original	A bird can fly. A raven is a bird.	A raven can fly.	✓
Paraphrasing	A bird is able to fly. A raven is a bird species.	A raven can fly.	✓
Synonym Substitution	A fowl can fly. A raven is a fowl.	A raven can fly.	✓
Negation	A bird can fly. A raven is a bird.	A raven cannot fly.	✗
Retained knowledge	A bird can live up to 100 years.	–	✓

Table 1: Examples of different perturbations strategies that were used to create the adversarial dataset (see §4.2).

ease. In order to systematically probe PLMs' deductive reasoning capabilities, we design controlled experiments over three different adversarial test settings by generating different *surface form perturbations*, *negation of the original examples*, and creating additional *retained knowledge anchors*. Table 1 shows some different adversarial examples.

Surface Form Perturbations. The theory of mental models postulates that deductive reasoning is based on manipulations of mental models representing situations. In other words, envisaging the situations and making a deduction can be viewed as a semantic process (Johnson-Laird, 1999; Polk and Newell, 1995). On the other hand, previous works demonstrate that, instead of learning interpretable meaning representations and generalising across different surface forms, PLMs tend to learn artefacts in the training data, e.g., higher-order word co-occurrence statistics (Sinha et al., 2021).

As both LoT and WD datasets are prompted from knowledge graphs, the lexical and syntactical variance of the dataset is minimal, with imaginable artefacts. To examine if the PLM-based reasoner could consistently perform reasoning against linguistic diversity and variability (in terms of both the token-level and the syntactic-level diversity), we employ two types of surface form perturbations to the data items from the original datasets:

- **Synonym Substitution:** In order to investigate to what extent the PLM-based reasoners would be sensitive to the token-level semantic diversity in terms of deriving their conclusions, we employ synonym substitution (Dhole et al., 2021) to the premises $\mathcal{P}$. Synonym substitution does not modify the syntactic structures and the premises' semantics, preserving all the original input's structural information. In our setting, a word is replaced by a uniform-randomly selected synonym based on WordNet (Fellbaum, 1998) with a probability of 50%.
- **Paraphrasing:** To further investigate the PLM-based reasoners' robustness on sentence-level semantic variability, we paraphrase the premises $\mathcal{P}$ with two paraphrasing systems: (i) PEGASUS, an end-to-end model fine-tuned for paraphrasing (Zhang et al., 2020) (ii) Syntactically Diverse Paraphrasing (SD-Paraphrasing), a two-step framework that incorporates neural syntactic preordering for better diversity (Goyal and Durrett, 2020).

Negated Examples. Understanding negation is often considered as the first test case in natural language understanding tasks (Ettinger, 2020; Khemlani et al., 2012; Schon et al., 2021). To examine whether PLMs can handle negation in the deductive reasoning task, we construct a set of negated samples by negating the hypothesis $\mathbf{h}$ or the cloze prompt $\mathbf{c}$ while keeping the premises $\mathcal{P}$ unchanged (Hosseini et al., 2021). For the classification-based reasoning task, the label of the negated hypothesis is then also flipped. For the cloze-completion task, since the answer for the original query will unlikely be the same answer for the negated queries, predicting the original answer would be regarded as a wrong prediction.

Anchors of Retained Knowledge. Prior work has shown that PLMs are prone to forgetting previously learnt knowledge when fine-tuning with new knowledge data (De Cao et al., 2021), the so-called catastrophic forgetting issue (Kirkpatrick et al., 2017; de Masson d'Autume et al., 2019). It is thus natural to also probe whether deductive reasoning training still retains the knowledge already stored in the original PLM. In this work, in order to measure to which extent PLMs retain the knowledge acquired during pretraining, we introduce a set of 'retained knowledge statements' or *anchors* for each dataset. There are two criteria for such *anchors*: (i) they are semantically close to the conclusions in our test data, (ii) they do not meet the conditions of the premises for a reasoning replacement.[4] Ideally, the reasoning training should not affect the prediction of PLMs when reasoning over these anchors.

We create such a set of anchors for both LoT and WD, and investigate the behaviour of the reasoning models over these anchors based on the prompt-based reasoning task. In particular, these anchors should be real-world textual statements that contain the target word (to meet criterion (i) above), but their newly composed sentences (by the reasoning replacement) are unlikely true statements (to meet criterion (ii) above). To this end, we use the BM25 algorithm (Sparck Jones et al., 2000) to retrieve the top 10 sentences for each 'reasoning' target word from the Wikipedia corpus.[5] Then, we construct the anchor sentences based on these top 10 retrieved sentences by replacing their target words with their hyponym/hypernym words. The final anchors are selected from these top 10 sentences only when the newly created sentences do not likely exist in the entire Wikipedia corpus (i.e. their top-1 similar sentence should have less than a BM25 score of 50). Ideally, this set of retrained knowledge statements is relevant but should not be affected by the reasoning training.

4.3 Evaluation

Following previous work (Talmor et al., 2020), the evaluation metric for classification-based reasoning is accuracy. For prompt-based reasoning, we calculate top K recall (R@K) by measuring what fraction of the correct answers are retrieved in the top K predictions. For the negation examples, we report the top K error rate (E@K) because a retrieved answer $\mathbf{a}$ and the negated cloze question $\neg\mathbf{c}$ would compose a fallacy.

In the following, we report our findings and numerical results based on the BERT-based reasoners (in particular `bert-base-uncased`), but we note that other PLMs (such as RoBERTa) of various sizes observe the same performance trends and result in the same findings and conclusions. Appendix B provides results for other PLMs.

Model	R@1	R@5	R@10
Dataset: **LoT**			
Pretrained	13.15	59.18	70.96
MLM-BERT	98.36	98.36	98.36
Cloze-BERT	99.73	100	100
Dataset: **WD**			
Pretrained	27.07	64.97	72.93
MLM-BERT	99.04	99.36	99.36
Cloze-BERT	100	100	100

Table 2: Recall of the correct answer in the top K predictions (R@K) from the BERT model before and after deductive prompt-based reasoning fine-tuning. Both MLM-BERT and Cloze-BERT achieve (near-)perfect R@1 scores after fine-tuning.

5 Results and Discussion

We evaluate the impact of reasoning training on the PLMs and investigate their robustness against three well-known issues of PLMs: utilising artefacts from data, incapability of modelling negation, and catastrophic forgetting. We further conduct qualitative analysis to understand the inference errors introduced by deductive reasoning training.

5.1 Deductive Reasoning Training

Finding 1 *All the deductive reasoning training approaches significantly improve PLMs' reasoning capabilities, achieving near-perfect deductive reasoning performance on both the reasoning test sets.*

Table 2 reveals that the prompt-based reasoners achieve near-perfect performance on both datasets, regardless of the reasoning training method. In particular, on LoT dataset the R@1 score of BERT has increased from 13.15% to 98.36% and 99.73% after the MLM reasoning training (i.e. MLM-BERT) and the cloze reasoning training (i.e. Cloze-BERT) respectively, which are in line with previously reported result (Talmor et al., 2020). The near-perfect trends are observed in the classification-based reasoning models, where CLS-BERT also achieves a high accuracy score of 94.72% after reasoning training (Table 5).[6] In sum, while the off-the-shelf BERT model already demonstrates a decent level of empirical performance, conducting reasoning training on the pretrained BERT achieves strong or even near-perfect performance on both datasets.

---

[4] Taking the premise (a) of Figure 1 as an example, the 'bird' token in the statement 'A bird can fly.' was replaced by its hyponym 'raven' after reasoning training. However, in the anchor statement 'The bird species is decreasing.', the 'bird' token should not be replaced by 'raven'.

[5] ElasticSearch: https://github.com/elastic/elasticsearch.

Adversarial Probing	**LoT**			WD		
	Pretrained	MLM-BERT	Cloze-BERT	Pretrained	MLM-BERT	Cloze-BERT
Original	13.15	98.36	99.73	27.07	99.04	100.00
+ Pegasus-Paraphrasing	11.79 (↓1.36)	64.66 (↓33.70)	50.96 (↓48.77)	16.88 (↓10.19)	49.36 (↓49.68)	51.27 (↓48.73)
+ SD-Paraphrasing	5.75 (↓7.40)	9.32 (↓89.04)	0.55 (↓99.18)	16.56 (↓10.51)	27.39 (↓71.65)	31.21 (↓68.79)
+ Syn. Substitution	20.00 (↑6.85)	64.38 (↓33.98)	64.66 (↓35.07)	15.29 (↓11.78)	61.78 (↓37.26)	62.42 (↓37.58)

Table 3: R@1 scores on test sets obtained via applying various surface form perturbations from Section 4.2.

Examples	**BERT**	**MLM-BERT**	**Cloze-BERT**
A holly is not music. A holly is part of a forest. A plant is an actor. A music is not an actor. A holly is a plant. A bluebottle is an organism. A marigold is not an angiosperm. A holly is not an important food source. A <MASK> is not an actor.	musician (0.1213) ✓	holly (0.7168) ✗	holly (0.9999) ✗
A perry is not a tree. A tree is not capable of burn. Alcohol is capable of burn. A perry is an alcohol. A <MASK> is not capable of burn.	tree (0.5244) ✓	person (0.0300) ✗	perry (0.9999) ✗

Table 4: Examples of top 1 predictions from the set of negated examples based on LoT test set. The wrong predictions are in red, and the reasonable predictions are in green. Some other key-related entities are in the same colour. The numbers in the parentheses are the respective probabilities of each prediction after the softmax layer.

Adversarial Probing	**LoT**
	CLS-BERT
Original	94.72
+ Pegasus-Paraphrasing	83.86 (↓10.86)
+ SD-Paraphrasing	71.14 (↓23.58)
+ Syn. Substitution	84.48 (↓10.24)
+ Negation	9.34 (↓85.49)

Table 5: Accuracy scores on test sets obtained via applying various surface form perturbations from Section 4.2.

5.2 Surface Form Perturbations

Finding 2 *Surface form perturbations drastically decrease PLMs' reasoning performance.*

A natural follow-up question to ask is to what extent the aforementioned near-perfect numbers really reflect the model's reasoning abilities. We thus perform surface form perturbations to add lexical and syntactic variance to the test datasets and probe the model against such variations.

Table 3 demonstrates the performances of the MLM-BERT and Cloze-BERT reasoners, as well as the vanilla pretrained BERT language model, on our controlled test sets generated by different perturbation approaches. We can observe that the scores for both reasoners decrease substantially (>30%) even when applying simple synonym substitution, which only adds lexical variance to the prompt-generated query. In contrast, the pretrained BERT language model is less vulnerable to such an issue. This finding aligns with the hypothesis that PLMs tend to memorise word co-occurrence statistics (Sinha et al., 2021).[7]

We also observe from Table 3 that the R@1 performances of all reasoners decrease strongly when applying either the Pegasus-Paraphrasing or SD-Paraphrasing methods to the premises. In particular, the drops are especially pronounced with the SD-Paraphrasing method, which is designed exactly to generate syntactically very diverse paraphrases. On the other hand, we see from Table 5 that similar performance degradation trends can be observed in the classification-based reasoning task. These results illustrate that current PLMs perform inconsistently against various surface form perturbations, suggesting that future work should look into the creation of more robust reasoners that should be resilient to lexical, syntactic, and semantic variability.

---

[6] Note that we cannot obtain the performance of original BERT since the classification head has not been trained yet.

[7] The comparison of large models (e.g. `bert-large`) can be found in Appendix B, which indicates the similar trends.

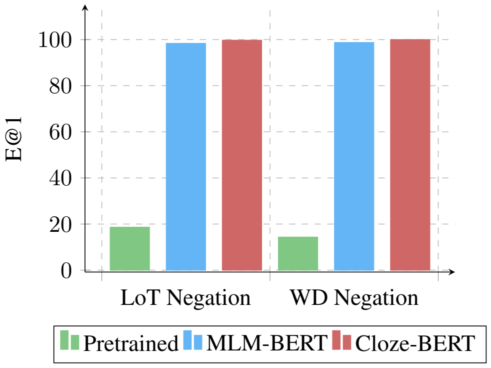


Figure 3: Results (E@1 scores, lower is better) on the test set comprising negated examples; before versus after deductive-reasoning training.

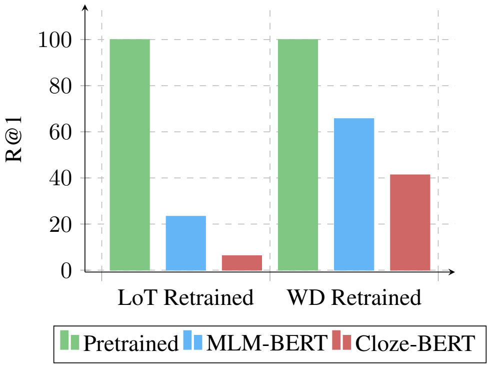


Figure 4: Results (R@1 scores, higher is better) on the test set comprising anchors of retained knowledge; before versus after deductive-reasoning training.

5.3 Negated Examples

Finding 3 *All reasoners cannot distinguish between negated and non-negated examples.*

Figure 3 reveals that the error rate E@1 scores (the lower, the better)[8] for all reasoners on the set of negated examples largely increase after reasoning training. The original off-the-shelf BERT model achieves E@1 of 18.63% on LoT and 14.33% on the WD dataset. However, after reasoning training, the error rate of MLM-BERT significantly increases to 98.36% (LoT) and 98.73% (WD). Cloze-BERT's performance is even worse, with E@1 scores of 99.73% (LoT) and 100% (WD), suggesting a clear case of overfitting to word co-occurrence and other artefacts in the training sets. Moreover, the accuracy score on negated LoT examples is only 9.34%, while a random baseline would score 50%. In sum, these scores indicate that the PLM-based reasoners cannot distinguish between negated and non-negated examples, and their performance on negated examples substantially worsens after reasoning training due to task-specific overfitting.

A quick error analysis, provided in Table 4, further points to the issues with negated examples. The first example shows that the off-the-shelf BERT model makes a reasonable guess, semantically related to an entity mentioned in the premises. This guess is similar to a human guess from the same premises. However, after reasoning training, both MLM-BERT and Cloze-BERT make wrong predictions, and Cloze-BERT is extremely confident in its wrong predictions.

5.4 Anchors of Retained Knowledge

Finding 4 *Previously learnt knowledge is not fully retained after reasoning training, and the trained reasoners (catastrophically) forget it.*

Figure 4 shows that performance on the anchors deteriorates substantially after reasoning training. On LoT, MLM-BERT 'forgets' ∼77% of the previously learnt knowledge, achieving only 23.34% on R@1. Cloze-BERT performs even worse, scoring only 6.27% R@1. The drops are slightly lower but still substantial on WD: MLM-BERT retains 65.7% and Cloze-BERT retains 41.32% of the previously stored knowledge. This result indicates that reasoning training yields the well-known phenomenon of catastrophic forgetting: this effect seems even more pronounced with Cloze-BERT, which relies on a very task-specific objective that might result in overfitting the task data.

Furthermore, Table 6 displays several qualitative examples where the predictions in green refer to the correct predictions based on human judgement. Notice that in the last example, even though the predictions from the two trained reasoners are correct, the probabilities are much lower.

Several strategies might help mitigate catastrophic forgetting. A promising direction is encapsulating lightweight adapter modules (Houlsby et al., 2019; Pfeiffer et al., 2020b; Ansell et al., 2021) within the underlying PLM, where all the 'deductive reasoning capability' will be stored solely in the adapter modules, leaving the original PLM intact (Pfeiffer et al., 2020a). Other similar parameter-efficient and modular methods include prefix tuning (Li and Liang, 2021) or sparse masks (Sung et al., 2021; Ansell et al., 2022). Their main

---

[8] E@1 only denotes a specific type of error: the model makes a wrong prediction due to negation being added, i.e. the negation error. However, many other types of errors that a model can make a wrong prediction, but they are not our focus and thus not measured. In other words, (1 - E@1) is not equivalent to accuracy.

Examples	BERT	MLM-BERT	Cloze-BERT
a flea is a parasitic <MASK>.	insect (0.2006) ✓	substance (0.1950) ✗	drone (0.0193) ✗
vapour density is a unitless <MASK>.	quantity (0.3494) ✓	quantity (0.2197) ✓	volume (0.0412) ✗
both sexes have a throat <MASK>.	pouch (0.5169) ✓	##lid (0.3606) ✗	hollow (0.0135) ✗
firefox is a web <MASK>.	browser (0.9144) ✓	browser (0.8861) ✓	browser (0.3243) ✓

Table 6: Examples of top 1 predictions from the set of Anchors of Retained Knowledge based on LoT. A wrong prediction is in red, and a reasonable prediction (aligning with human judgement) is in green. The numbers in the parentheses are the respective probabilities of each prediction after the softmax layer.

premise is to separate knowledge extraction and composition, preserving previously learnt knowledge during reasoning training. We leave reasoning training with such methods for future research.

6 Conclusion

In this paper, we probed into the deductive reasoning capabilities of PLMs and conducted comprehensive controlled experiments to examine and compare various deductive reasoning training approaches. Our experimental results showed that current PLM-based deductive reasoners suffer from several issues: **1)** they rely on artefacts from the training data, **2)** they are incapable of modelling negation in deductive reasoning, and **3)** they forget knowledge acquired during pretraining when they get specialised into deductive reasoners. In particular, our experimental study demonstrated that models are vulnerable to multiple adversarial methods, including simple surface form perturbations such as synonym substitution or paraphrasing. While the PLMs trained for deductive reasoning achieve seemingly perfect empirical results in different reasoning datasets, they cannot yet systematically generalise to other deductive reasoning examples. Consequently, our study also calls for further, more rigorous examinations of future PLM-based models' deductive reasoning abilities.

Acknowledgements

Ivan Vulić is supported by a personal Royal Society University Research Fellowship (no 221137; 2022-). Songbo Hu is supported by Cambridge International Scholarship.

Limitations

Despite the thorough experiments on standard and popular PLMs of various sizes, this study explores only encoder-based models. Some generation-based models under other Transformer architectures, such as encoder-decoder (T5) or decoder-only (GPT-3), were also deployed in the reasoning tasks (Bostrom et al., 2021; Wei et al., 2022). We do not probe those groups of models here due to the difficult of evaluation and we leave them for future research.

Further, for prompt-based reasoning, the current reasoners only support and have been evaluated on single-token prediction in a single language (i.e. English). Several prior works have demonstrated that conducting multi-token prediction is considerably more difficult (Jiang et al., 2020a), which would pose an additional challenge to the PLMs in deductive reasoning tasks. One avenue of future work should also extend the scope of analyses to multilingual and multi-token prediction.

In addition, we note that better evaluation resources that could address paraphrases and word senses, especially for mask-filling tasks, are still lacking. This limitation is particularly significant in our setting. For example, in addition to the single-token answer in the evaluation datasets we used, there are some other feasible answers (e.g. synonyms) for the same query, which should also be considered a correct prediction. However, such answers are ignored by the current standard evaluation protocols. As a result, there is a certain level of unavoidable noise in the evaluation process.

Finally, introducing a reasoning dataset is highly challenging and appreciated by the community. Leap-of-thought is to our knowledge the only existing dataset that is suitable for our deductive reasoning evaluation. To solidify our conclusions, we further constructed an auxiliary dataset (WD) following a similar procedure to LoT. Although our data construction method is commonly used to extract reasoning examples, such an automatic procedure, unfortunately, inevitably reflects the quality and errors (e.g. nonsensical statements) from our source (WikiData). To reduce such noisy examples, we have conducted multiple rounds of filtering (see

Appendix for the filtering process) and manually removed as many meaningless relations as we can, given that manually verifying each reasoning example is a highly labour-intensive task.

References

Somak Aditya, Yezhou Yang, and Chitta Baral. 2018. Explicit reasoning over end-to-end neural architectures for visual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

John R Anderson. 2014. *Rules of the mind*. Psychology Press.

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 1778–1796.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781.

Krzysztof R. Apt, Howard A. Blair, and Adrian Walker. 1988. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–148. Morgan Kaufmann.

Kaj Bostrom, Xinyu Zhao, Swarat Chaudhuri, and Greg Durrett. 2021. Flexible generation of natural language deductions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6266–6278.

Martin D Braine and David P O'Brien. 1991. A theory of if: A lexical entry, reasoning program, and pragmatic principles. *Psychological review*, page 182.

Martin DS Braine. 1998. Steps toward a mental-predicate logic. In *Mental logic*, pages 281–340. Psychology Press.

John Broome. 2013. *Rationality through reasoning*. John Wiley & Sons.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3882–3890.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Kaustubh D. Dhole, Varun Gangal, Sebastian Gehrmann, Aadesh Gupta, Zhenhao Li, Saad Mahamood, Abinaya Mahendiran, Simon Mille, Ashish Srivastava, Samson Tan, Tongshuang Wu, Jascha Sohl-Dickstein, Jinho D. Choi, Eduard H. Hovy, Ondrej Dusek, Sebastian Ruder, Sajant Anand, Nagender Aneja, Rabin Banjade, Lisa Barthe, Hanna Behnke, Ian Berlot-Attwell, Connor Boyle, Caroline Brun, Marco Antonio Sobrevilla Cabezudo, Samuel Cahyawijaya, Emile Chapuis, Wanxiang Che, Mukund Choudhary, Christian Clauss, Pierre Colombo, Filip Cornell, Gautier Dagan, Mayukh Das, Tanay Dixit, Thomas Dopierre, Paul-Alexis Dray, Suchitra Dubey, Tatiana Ekeinhor, Marco Di Giovanni, Rishabh Gupta, Rishabh Gupta, Louanes Hamla, Sang Han, Fabrice Harel-Canada, Antoine Honore, Ishan Jindal, Przemyslaw K. Joniak, Denis Kleyko, Venelin Kovatchev, and et al. 2021. Nl-augmenter: A framework for task-sensitive natural language augmentation. *CoRR*.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, pages 34–48.

R J Falmagne and J Gonsalves. 1995. Deductive inference. *Annual Review of Psychology*, 46(1):525–559.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Benjamin Heinzerling and Kentaro Inui. 2021. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main xvolume*, pages 1772–1791.

Chadi Helwe, Chloé Clavel, and Fabian M. Suchanek. 2021. Reasoning with transformer-based models: Deep learning, but shallow reasoning. In *3rd Conference on Automated Knowledge Base Construction*.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 9106–9118.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. 2021. Understanding by understanding not: Modeling negation in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1301–1312.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 2790–2799.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020a. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5943–5959.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, pages 423–438.

P. N. Johnson-Laird. 1999. Deductive reasoning. *Annual Review of Psychology*, 50(1):109–135. PMID: 15012459.

Philip N. Johnson-Laird. 1983. *Mental models : towards a cognitive science of language, inference, and consciousness*. Cambridge, MA: Harvard University Press. Excerpts available on Google Books.

Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are pretrained language models symbolic reasoners over knowledge? In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 552–564.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818.

Sangeet Khemlani, Isabel Orenes, and P. N. Johnson-Laird. 2012. Negation: A theory of its meaning, representation, and use. *Journal of Cognitive Psychology*, 24(5):541–559.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *CoRR*.

Douglas B. Lenat, Ramanathan V. Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd. 1990. CYC: toward programs with common sense. *Commun. ACM*, 33(8):30–49.

Shiyang Li, Jianshu Chen, and Dian Yu. 2019. Teaching pretrained models with commonsense reasoning: A preliminary kb-based approach. *arXiv preprint arXiv:1909.09743*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 4582–4597.

Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. 2022. Generated knowledge prompting for commonsense reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 3154–3169.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Zaiqiao Meng, Fangyu Liu, Thomas Clark, Ehsan Shareghi, and Nigel Collier. 2021. Mixture-of-partitions: Infusing large biomedical knowledge graphs into BERT. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4672–4681.

Zaiqiao Meng, Fangyu Liu, Ehsan Shareghi, Yixuan Su, Charlotte Collins, and Nigel Collier. 2022. Rewire-then-probe: A contrastive recipe for probing biomedical knowledge of pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 4798–4810.

Kanishka Misra, Allyson Ettinger, and Julia Rayz. 2020. Exploring BERT's sensitivity to lexical cues using tests from semantic priming. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4625–4635.

A Newell. 1990. Unified theories of cognition cambridge university press. *Cambridge, MA*.

A. Newell and H. Simon. 1956. The logic theory machine–a complex information processing system. *IRE Transactions on Information Theory*, pages 61–79.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian

Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2523–2544.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 2463–2473.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 46–54.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7654–7673.

Thad A Polk and Allen Newell. 1995. Deduction as verbal reasoning. *Psychological Review*, page 533.

Markus Norman Rabe, Dennis Lee, Kshitij Bansal, and Christian Szegedy. 2021. Mathematical reasoning via self-supervised skip-tree training. In *International Conference on Learning Representations*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992.

Soumya Sanyal, Harman Singh, and Xiang Ren. 2022. FaiRR: Faithful and robust deductive reasoning over natural language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 1075–1093.

Claudia Schon, Sophie Siebert, and Frieder Stolzenburg. 2021. Negation in cognitive reasoning. In *KI 2021: Advances in Artificial Intelligence - 44th German Conference on AI*, Lecture Notes in Computer Science, pages 217–232.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913.

K. Sparck Jones, S. Walker, and S.E. Robertson. 2000. A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Information Processing & Management*, 36(6):779–808.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Yi-Lin Sung, Varun Nair, and Colin Raffel. 2021. Training neural networks with fixed sparse masks. In *Advances in Neural Information Processing Systems*, pages 24193–24205.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. In *Advances in Neural Information Processing Systems*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed H. Chi, and Quoc Le. 2022. Lamda: Language models for dialog applications. *CoRR*.

Asahi Ushio, Luis Espinosa Anke, Steven Schockaert, and Jose Camacho-Collados. 2021. BERT is to NLP what AlexNet is to CV: Can pre-trained language models identify analogies? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 3609–3624.

Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021a. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, pages 176–194.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Kaiyu Yang and Jia Deng. 2021. Learning symbolic rules for reasoning in quasi-natural language. *arXiv preprint arXiv:2111.12038*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, pages 11328–11339.

Ming Zhou, Nan Duan, Shujie Liu, and Heung-Yeung Shum. 2020. Progress in neural nlp: Modeling, learning, and reasoning. *Engineering*, 6(3):275–290.

Wenxuan Zhou, Fangyu Liu, Ivan Vulić, Nigel Collier, and Muhao Chen. 2022. Prix-LM: Pretraining for multilingual knowledge base construction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 5412–5424.

A Experimental Details

Table 7 lists our model hyperparameters. Among these models, MLM-BERT and Cloze-BERT were implemented using the HuggingFace transformers package (Wolf et al., 2020). We implement CLS-BERT via the SBERT repository (Reimers and Gurevych, 2019), which is built on top of the HuggingFace repository (Wolf et al., 2020). Unless mentioned otherwise, all the hyperparameters are set to the default values provided in the HuggingFace and SBERT repositories.

Hyper-parameter	**Value**
MLM-BERT and Cloze-BERT	
batch_size	2
max_sequence_length	512
training_epoch	3
CLS-BERT	
batch_size	64
max_sequence_length	128
training_epoch	20

Table 7: Model hyper-parameters.

B Evaluation Results for PLMs

Table 8 supplements the main paper by providing additional results with classification-based reasoners, where the reasoners start from different PLMs: `distil-bert`, `bert-base`, `bert-large`, `roberta-base`, and `roberta-large`. These results corroborate the main findings presented in the main paper; see Section 5. Larger models do perform slightly better than their smaller variants on average. However, the results also demonstrate that different models, regardless of their size, suffer from exactly the same issues, discussed in the main paper.

Table 9 and Table 10 demonstrate performance over adversarial test sets for prompt-based reasoners (1. MLM-based, 2. Cloze-based). The findings from these tables align with the findings from the main results presented in Section 5.

C WD Dataset Construction Pipeline

We construct the **WD** dataset following the pipeline shown in Figure 5, and outlined in what follows.

Source Data. We choose the Wikidata5m dataset (Wang et al., 2021b) as the knowledge source for **WD**. Wikidata5m is a million-scale knowledge graph dataset created upon Wikidata (Vrandecic and Krötzsch, 2014). This dataset comprises 20 million triples, describing relevant and important knowledge statements about real-world entities.

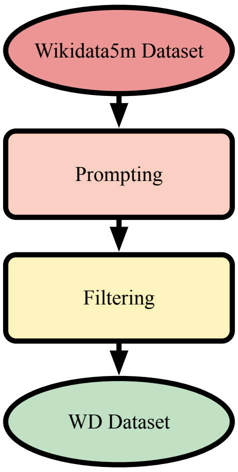


Figure 5: Data construction pipeline for the **WD** dataset. This **WD** dataset is extracted and derived from WikiData5m (Wang et al., 2021b). Following the previous pipeline (Petroni et al., 2019), we convert a set of knowledge graph triples into linguistic statements using manually designed prompts.

Prompting. We manually select a set of relations based on their frequency and design their corresponding prompts shown in Table 11. Given a taxonomic relation, such as is_a with relation ID P31, we sample a relevant taxonomic-knowledge graph triple: $\langle raven, is\_a, bird\rangle$. We then retrieve other relevant triples about the subject and the object, for example, $\langle bird, is\_capable\_of, fly\rangle$ and $\langle raven, is\_capable\_of, fly\rangle$. Next, we use our predefined prompts to convert the knowledge graph triples to textual knowledge statements: *A bird can fly*, *A bird is a raven* and *A raven can fly*. Furthermore, these prompted statements assemble an inference instance in the **WD** dataset: if *A bird can fly* and *A bird is a raven*, then *A raven can fly*. In our cloze-completion task setting, we mask the object of the taxonomic triple (e.g. $raven$) in the conclusion statement to form a cloze question (*A [MASK] can fly*). Therefore, $raven$ is the correct answer to this question.

Filtering. We filter those constructed inference instances with the following properties: (i) We only choose examples with answers being a single masked token, and these answers should be included in the BERT vocabulary. (ii) For all in-

Adversarial Probing	**LoT**				
	distilbert-based-uncased	bert-base-uncased	bert-large-uncased	roberta-base	roberta-large
Original	90.22	94.72	98.76	92.71	99.38
+ Pegasus-Paraphrasing	77.27 (↓12.95)	83.86 (↓10.86)	84.40 (↓14.35)	80.84 (↓11.87)	87.20 (↓12.18)
+ SD-Paraphrasing	66.56 (↓23.66)	71.14 (↓23.58)	73.39 (↓25.39)	75.95 (↓16.76)	78.97 (↓20.40)
+ Syn. Substitution	81.46 (↓8.76)	84.48 (↓10.24)	93.33 (↓5.43)	80.76 (↓11.95)	94.26 (↓5.12)
+ Negation	18.86 (↓71.37)	9.34 (↓85.49)	2.58 (↓96.28)	19.40 (↓73.31)	35.85 (↓63.53)

Table 8: Accuracy on different LoT adversarial test sets for classification-based reasoners with various back-boned PLMs.

Adversarial Probing	**LoT**			**WD**		
	MLM-DISTILBERT	MLM-BERT	MLM-BERT-LARGE	MLM-DISTILBERT	MLM-BERT	MLM-BERT-LARGE
Original	99.45	98.36	99.45	99.36	99.04	98.09
+ Pegasus-Paraphrasing	76.71 (↓22.74)	64.66 (↓33.7)	60.27 (↓39.18)	56.05 (↓43.31)	49.36 (↓49.68)	52.23 (↓45.86)
+ SD-Paraphrasing	24.66 (↓74.79)	9.32 (↓89.04)	4.93 (↓94.52)	28.66 (↓70.7)	27.39 (↓71.65)	33.12 (↓64.97)
+ Syn. Substitution	64.11 (↓35.34)	64.38 (↓33.98)	64.93 (↓34.52)	62.1 (↓37.26)	61.78 (↓37.26)	60.19 (↓37.9)
+ Negation (E@1 ↓)	97.53	98.36	95.89	99.36	98.73	98.41
Retained Knowledge	25.86	23.34	26.67	58.33	65.7	56.61

Table 9: Top 1 recall (R@1) on adversarial test sets for various prompt-based reasoners with MLM training. The numbers for the negated examples indicate the top 1 error rates (the lower, the better).

Adversarial Probing	**LoT**			**WD**		
	Cloze-DISTILBERT	Cloze-BERT	Cloze-BERT-LARGE	Cloze-DISTILBERT	Cloze-BERT	Cloze-BERT-LARGE
Original	100	99.73	98.63	100	100	100
+ Pegasus-Paraphrasing	57.81 (↓42.19)	50.96 (↓48.77)	47.4 (↓51.23)	57.01 (↓42.99)	51.27 (↓48.73)	56.37 (↓43.63)
+ SD-Paraphrasing	2.74 (↓97.26)	0.55 (↓99.18)	0.82 (↓97.81)	37.26 (↓62.74)	31.21 (↓68.79)	46.82 (↓53.18)
+ Syn. Substitution	64.66 (↓35.34)	64.66 (↓35.07)	63.84 (↓34.79)	62.74 (↓37.26)	62.42 (↓37.58)	62.74 (↓37.26)
+ Negation (E@1 ↓)	100	99.73	98.63	100	100	99.68
Retained Knowledge	11.79	6.27	1.45	35.61	41.32	16.95

Table 10: Top 1 recall (R@1) on adversarial test sets for various prompt-based reasoners with Cloze-filling training. The numbers for the negated examples indicate the top 1 error rates (the lower, the better).

ference instances, the maximum number of occurrences of a single answer is 50 to balance the dataset and avoid excessive repetition.

Final WD Dataset. The final WD dataset contains 4,851 instances, which are randomly split into 4,124/413/314 instances for training/validation/testing while keeping that the answers of the testing set should not appear in the training/validation sets. This is to ensure that trained reasoners need to draw conclusions via conducting deductive reasoning rather than via memorisation.

Relation ID	Prompt
P31	[X] is a [Y] .
P136	[X] is a genre of [Y] .
P179	[X] is part of the [Y] series .
P279	[X] is a subclass of [Y] .
P527	[X] consists of [Y] .
P1269	[X] is a topic of [Y] .
P17	[X] is hosted in [Y] .
P39	[X] holds a [Y] position .
P101	[X] is a subject of [Y] .
P106	[X] is a [Y] by profession .
P140	The religion of [X] is [Y] .
P144	[X] is based on [Y] .
P180	[X] is a painting of [Y] .
P276	[X] is located at [Y] .
P306	[X] runs on [Y] operating system .
P355	[X] owns [Y] .
P360	[X] is a list of [Y] .
P400	[Y] is a platform of [X] .
P404	The game mode of [X] is [Y] .
P462	The color of [X] is [Y] .
P463	[X] is a member of [Y] .
P737	[X] is influenced by [Y] .
P749	[Y] owns [X] .
P1303	[X] plays [Y] .
P1343	[X] is written about in [Y] .

Table 11: Manually written prompts for generating the WD dataset. Given a triple $\langle [X], R\_ID, [Y] \rangle$, the textual knowledge statement (e.g. premises) is written based on the above prompts. R_ID is the unique relation ID in the Wikidata5m dataset. Gray entries (first six rows) denote taxonomic relations.

Selective In-Context Data Augmentation for Intent Detection using Pointwise V-Information

Yen-Ting Lin[*] **Alexandros Papangelis**[†] **Seokhwan Kim**[†] **Sungjin Lee**[†]
Devamanyu Hazarika[†] **Mahdi Namazifar**[†] **Di Jin**[†] **Yang Liu**[†] **Dilek Hakkani-Tur**[†]
[⋆]National Taiwan University
[†]Amazon Alexa AI
ytl@ieee.org {papangea,seokhwk,sungjinl,dvhaz,mahdinam,djinamzn,yangliud,hakkanit}@amazon.com

Abstract

This work focuses on in-context data augmentation for intent detection. Having found that augmentation via in-context prompting of large pre-trained language models (PLMs) alone does not improve performance, we introduce a novel approach based on PLMs and pointwise V-information (PVI), a metric that can measure the usefulness of a datapoint for training a model. Our method first fine-tunes a PLM on a small seed of training data and then synthesizes new datapoints – utterances that correspond to given intents. It then employs intent-aware filtering, based on PVI, to remove datapoints that are not helpful to the downstream intent classifier. Our method is thus able to leverage the expressive power of large language models to produce diverse training data. Empirical results demonstrate that our method can produce synthetic training data that achieve state-of-the-art performance on three challenging intent detection datasets under few-shot settings (1.28% absolute improvement in 5-shot and 1.18% absolute in 10-shot, on average) and perform on par with the state-of-the-art in full-shot settings (within 0.01% absolute, on average).

1 Introduction

Intent detection, defined as the identification of a user's intent given an utterance, is a fundamental element in task-oriented dialogue systems, usually occurring within the Natural Language Understanding (NLU) component. One of the practical challenges of training and deploying NLU modules is data scarcity, due to various reasons, such as under-represented languages, privacy and ethical concerns, or simply the cost of collecting and annotating sufficiently large amounts of data for new intents. Consequently, accurately identifying intents in limited-resource scenarios has drawn attention from the community (Papangelis et al., 2021; Mehri and Eric, 2021; Zhang et al., 2021b, for example).

There are three main families of approaches that address the challenge of limited data for intent detection: data augmentation (Peng et al., 2021; Li et al., 2021), focusing on generating high-quality synthetic training and evaluation data; few-shot learning (Zhang et al., 2020, 2021b), focusing on creating learning algorithms that can cope with limited amounts of data; and transfer learning (Namazifar et al., 2021), focusing on learning algorithms that can generalize across domains (therefore not requiring in-domain data). In this work, we follow the data augmentation approach, which is a general method that attempts to augment a human-authored dataset with a large set of synthetically-generated instances. Most recent work has suggested using Pre-trained Language Models (PLMs) for data augmentations under various setups, e.g., (Peng et al., 2021), showing great improvements in performance. However, simply generating a large number of synthetic data points is not enough; we need to consider the quality of each data point, i.e., how beneficial it would be to the model's performance if that synthetic data point is added to the training set. This is an important issue since the model might learn to overfit to synthetic datapoints (which may be low quality, represent specific use cases, etc.) and thus under-perform on real data.

In this work, we propose to apply Pointwise $\mathcal{V}$-Information (PVI) (Ethayarajh et al., 2022) for data augmentation, in a way that leverages a PLM to generate synthetic examples that are relevant and beneficial for training the downstream model, which in our case is an intent classifier. Our contributions are as follows:

- We propose a novel filtering method based on PVI (Ethayarajh et al., 2022) to filter out examples that are not relevant or helpful to the desired intent.
- We conduct experiments on three challenging intent detection datasets and show that our

---

[*] Work done during internship at Amazon Alexa AI

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1463–1476
May 2-6, 2023 ©2023 Association for Computational Linguistics

method achieves state-of-the-art performance.

- We conduct an in-depth study and present a comprehensive analysis of the factors that influence performance, including ablation studies and comparisons with alternative methods.

The rest of the paper is organized as follows: In Section 2 we present relevant work and in Section 3 we introduce our method. In sections 4 and 5 we discuss training details, experiments, and results. In section 6, we present our analysis and discuss alternative approaches we investigated. In section 7 we conclude, and in the following sections we discuss limitations and ethical considerations.

2 Related Work

Intent Detection Intent detection is the task of identifying the user's intent by mapping the user's natural language utterance into one of several predefined classes (Hemphill et al., 1990; Coucke et al., 2018). It is a critical component in the pipeline of task-oriented dialogue systems, as it is used to determine the user's goal and to trigger an appropriate system action (Raux et al., 2005; Young et al., 2013). Several datasets have been proposed to evaluate the performance of intent detection models (Casanueva et al., 2020; Liu et al., 2019a; Larson et al., 2019, for some recent examples). With the availability of such datasets, intent detection has been extensively studied in the literature. Recently, pre-trained language models (e.g., BERT (Devlin et al., 2019)) have been shown to be effective in intent detection (Bunk et al., 2020; Zhang et al., 2020, 2021a,b; Mehri and Eric, 2021).

Data Augmentation Data augmentation is a widely-used technique to address the problem of data scarcity. Paraphrasing the data is one of the ways frequently used for augmentation and can produce more diverse synthetic text with different word choices and sentence structures while preserving the meaning of the original text. Paraphrasing methods have been shown to be effective in many natural language processing tasks (Gupta et al., 2018; Edunov et al., 2018; Iyyer et al., 2018; Wei and Zou, 2019; Cai et al., 2020; Okur et al., 2022; Panda et al., 2021; Jolly et al., 2020). However, such methods often fail to generate more challenging and semantically diverse sentences that are important for the robustness of the downstream models.

Recently, conditional generation – using a PLM to produce text conditioned on some label – has become the dominant paradigm of data augmentation (Bowman et al., 2016; Kumar et al., 2019; Anaby-Tavor et al., 2020; Kumar et al., 2020; Yang et al., 2020a; Lee et al., 2021). This is usually achieved by fine-tuning a language model to produce the original text given the label.

In the field of intent detection, previous work has proposed using data augmentation techniques to generate synthetic training data (Sahu et al., 2022; Papangelis et al., 2021). Sahu et al. (2022) also used PLMs to generate augmented examples, but they require human effort for labeling. This is a challenging task since it is expensive to annotate large amounts of data.

Our approach involves data valuation, similar to the concepts of Ghorbani and Zou (2019); Mindermann et al. (2022). However, our approach differs from such previous work in two key ways. First, Ghorbani and Zou (2019) only evaluated the quality of the training set after training them, whereas we evaluate the synthetic examples before training the task model. Second, Mindermann et al. (2022) selected points that minimize the loss on a holdout set, whereas we select synthetic examples that are reasonably challenging to the task model. Our approach aims to address the problem of data scarcity by evaluating the synthetic examples generated by PLMs and selecting the most valuable examples to augment the training data.

In-context Learning Large language models such as GPT-3 (Brown et al., 2020) and OPT (Zhang et al., 2022) have shown to be able to perform many natural language processing tasks with in-context learning. In this paradigm, the model is provided with a few exemplars based on which it performs the respective task.

In-context learning is a promising solution for few-shot learning. Because of the effectiveness in few-shot performance, in-context learning has been applied to a wide range of NLP tasks. For dialogue tasks, in-context learning has been applied to intent classification (Yu et al., 2021), semantic parsing (Shin and Durme, 2022), and dialogue state tracking (Hu et al., 2022).

However, PLMs require a large amount of computational resources and the limitation on input length restricts the application of PLMs to intent detection tasks with large numbers of intents (e.g., 150 intents in CLINC (Larson et al., 2019)), where


Prompt:

The following sentences belong to the same category as '**Refund not showing up**':

Example 1:I'm supposed to have a refund but it isn't there

Example 2:My refund is not here yet

...

Example 10:When will I be able to see the refund

Example 11:

Example Completions:

- It's been weeks since I ordered my items and I still can't seem to see the funds.
- I am looking for information about when I can expect my refund
- I've submitted a refund request but I haven't seen a change in my account. What's going on?
- Please track my refund!
- the refund has not arrived yet so when will it show?
- Where is my refund? It doesn't appear on my statement.
- There was an error with the refund, when will I receive this amount again

Figure 1: An example of the prompt used to generate synthetic examples. The intent class is *refund not showing up*. Completions are generated by a pre-trained language model via sampling. Note that 5-shot experiments only use 5 examples from the training set.

we cannot fit examples for each intent in the input. One solution would be to call the model multiple times, each time with a subset of the possible intents. This would lead to increased inference time and may also impact performance. Consequently, Yoo et al. (2021); Sahu et al. (2022) leveraged in-context learning and PLMs to generate synthetic examples for intent detection, instead of directly deploying the PLM. However, they did not consider the quality of the generated examples, which may lead to the model overfitting on examples that are not relevant to the desired intent.

3 In-Context Data Augmentation

In the following section, we describe our proposed two-stage method for data augmentation, which we refer to as In-Context Data Augmentation (**ICDA**). The overall procedure is summarized in Algorithm 1. We apply ICDA to the task of few-shot intent detection, which involves classifying a user utterance x into an intent label $y \in Y$. ICDA aims to generate synthetic examples x' such that they would belong to a given intent y.

3.1 Synthesizing Examples

The core idea is to use a large pre-trained language model such as GPT-3 (Brown et al., 2020) or OPT (Zhang et al., 2022) to generate synthetic data in the context of the training set. In particular, for each intent class, we create a natural language context (prompt) that contains the intent class name, a set of real training examples under the same intent class, and an incomplete example. For instance, the prompt for the intent class *refund_not_showing_up* is shown in Figure 1. We feed the prompt to the language model and obtain a set of synthetic examples as outputs. In this work, we use OPT-66B (Zhang et al., 2022) as the language model to generate a set of examples for each intent class. We adopt typical decoding with $\tau = 0.9$ (Meister et al., 2022) and set repetition penalty to 1.1 following Keskar et al. (2019) to generate the synthetic examples.[1] Due to the fine-grained nature of intents, and the sampling-based generation aiming to produce a set of diverse datapoints, we expect some of the generated utterances to not match the given intent.

Note that our method leverages PLMs in a way that is orthogonal to the intent detection model. Unlike other methods that use the same model to directly predict the intent class of a user utterance, we use a PLM to generate synthetic training instances. These instances are then used to augment the actual training data and train a smaller intent detection model. This approach leverages the power of PLMs while preserving the independence of the intent detection model design.

3.2 PVI Filtering

As mentioned above, given the stochastic nature of synthetic data generation, we expect some of the synthetic utterances not to match the given intent. To address this phenomenon, we filter generated instances and retain only those that are relevant and helpful to the desired intent classes.

Specifically, we apply Pointwise V-Information (Ethayarajh et al., 2022) - an idea originally suggested for understanding how difficult a dataset is - as a filter to discard unhelpful datapoints. PVI of an utterance x with respect to its corresponding intent class, y, is defined as:

$$\mathrm{PVI}(x \to y) = -\log_2 g^*[\emptyset](y) + \log_2 g'[x](y)$$

where, in this work, g' and g^* are the intent detection models finetuned with and without the input x, respectively. $\emptyset$ is a special token that is used to indicate the absence of an input utterance.

Intuitively, PVI measures the amount of information that the input x provides to the intent detection

---

[1] Implementation details are available from `https://huggingface.co/docs/transformers/main_classes/text_generation`

Algorithm 1: In-Context Data Augmentation with PVI Filtering

```
Input: Task Model $\mathcal{V}$, Language Model
       $\mathcal{PLM}$, Data Multiplier $m$, PVI
       Threshold Function $\epsilon$
Output: Task Model $g$
Data: Seed Data $\mathcal{D}_{\text{train}} =$
      $\{(\text{input } x_i, \text{gold label } y_i)\}_{i=1}^{n}$
$g' \leftarrow$ Finetune $\mathcal{V}$ on $\mathcal{D}_{\text{train}}$
$\emptyset \leftarrow$ empty string
$g^* \leftarrow$ Finetune $\mathcal{V}$ on $\{(\emptyset, y_i) | (x_i, y_i) \in \mathcal{D}_{\text{train}}\}$
$\mathcal{D}_{\text{synthetic}} \leftarrow \text{Prompt}(\mathcal{PLM}, \mathcal{D}_{\text{train}})$
for $(x_i, y_i) \in \mathcal{D}_{\text{synthetic}}$ do
    $\text{PVI}(x_i \rightarrow y_i) \leftarrow -\log_2 g^*[\emptyset](y_i) + \log_2 g'[x_i](y_i)$
$\mathcal{D}_{\text{synthetic}} \leftarrow \{(x_i, y_i) | (x_i, y_i) \in \mathcal{D}_{\text{synthetic}} \ \& \ \text{PVI}(x_i \rightarrow y_i) > \epsilon(y_i)\}$
$g \leftarrow$ Finetune $\mathcal{V}$ on $\mathcal{D}_{\text{train}} \bigcup \mathcal{D}_{\text{synthetic}}$
```

model (compared to the absence of meaningful input). A high PVI value indicates that the input x provides a lot of information to the model, and thus is more likely to be helpful when training the model to classify instances of the intent class y. On the contrary, a low PVI value indicates that the input x provides little information to the model, and thus is likely to be irrelevant to the intent class y (Ethayarajh et al., 2022).

We set a threshold ϵ (tunable parameter) to determine which x are retained and conduct experiments to study the effect of the threshold in Section 6. Algorithm 1 defines ϵ as a function of y to allow flexibility in its definition: either a fixed threshold for all intent classes, or a different threshold per intent class.

4 Experimental Setup

4.1 Datasets

To evaluate the effectiveness of our approach in intent detection in cases where we have a large number of often semantically similar intent labels, we chose the BANKING (Casanueva et al., 2020), HWU (Liu et al., 2019a), and CLINC (Larson et al., 2019) datasets and compare with recent state-of-the-art baselines. BANKING comprises 13,083 utterances in a single banking domain and 77 intents. HWU includes 25,716 utterances with 64 intents across 21 domains. CLINC contains 23,700 utterances with 150 intents across 20 domains.

	Full-shot mult.	Few-shot mult.
XS	-	1x
S	1x	4x
M	2x	16x
L	4x	64x
XL	-	128x

Table 1: To assess the impact of the synthetic data size on performance, we experiment with several data multipliers (*synthetic data size = source data size x mult.*).

4.2 Training

In our experiments, we use RoBERTa-LARGE (Liu et al., 2019b) as the intent detection model $\mathcal{V}$ in Algorithm 1. We use OPT-66B[2] (Zhang et al., 2022) as the language model $\mathcal{PLM}$ to generate synthetic examples and set the data multiplier m to be 128[3]. We set the PVI threshold function ϵ to be the average PVI under each intent class in the validation set, where the PVI is computed using the same models as in Algorithm 1. We train RoBERTa-LARGE for 40 epochs with a batch size of 16, a learning rate of $1e-5$, and the AdamW optimizer (Loshchilov and Hutter, 2019). We use the HuggingFace Transformers library (Wolf et al., 2020) for all experiments.

4.3 Baseline Models

We compare our proposed method with the following baselines:

RoBERTa-BASE + Classifier is a baseline that uses RoBERTa-BASE (Liu et al., 2019b) with a linear classifier on top (Zhang et al., 2020).

USE is a universal sentence encoder pre-trained on 16 languages supporting multiple down-stream tasks (Yang et al., 2020b).

CONVERT is an intent detection model finetuned from dual encoder models, which is pre-trained on (input, response) pairs from Reddit (Henderson et al., 2020).

CONVBERT fine-tunes BERT on a large open-domain dialogue corpus with 700 million conversations (Mehri et al., 2020) .

CONVBERT + Combined is an intent detection model based on CONVBERT, with example-driven training based on similarity matching and observers for transformer attentions. It also conducts task-adaptive self-supervised learning with masked language modeling (MLM) on the intent detection

---

[2] We used p3dn.24xlarge AWS EC2 instances for our experiments.

[3] This means that we generate m times the available training data, e.g. (5 x 77) x m in the 5-shot BANKING case.

datasets. Here, “Combined" represents the best MLM+Example+Observers setting in the referenced paper (Mehri and Eric, 2021).

DNNC (Discriminative Nearest-Neighbor Classification) is a discriminative nearest-neighbor model, which finds the best-matched example from the training set through similarity matching. The model conducts data augmentation during training and boosts performance by pre-training on three natural language inference tasks (Zhang et al., 2020).

CPFT (Contrastive Pre-training and Fine-Tuning) is the current state-of-the-art in few-shot intent detection on the selected datasets. It is pre-trained on multiple intent detection datasets in a self-supervised contrastive manner and then fine-tuned with supervised contrastive learning (Zhang et al., 2021b).

5 Experimental Results

We conduct experiments on three benchmark datasets to validate the effectiveness of our proposed method. We first use OPT-66B to generate augmentation examples and then apply our method to enhance a RoBERTa-Large model trained on three datasets. We repeat all experiments with 5 random seeds and report the average performance in Full-shot and Few-shot settings. To investigate the effect of the synthetic data size, we experiment with a variety of multipliers (see Table 1 for notations). Results are shown in Table 2.

Full-shot settings. In this setting, we use the entire training set for each domain. The proposed method achieves the best performance on BANKING and comparable results on HWU and CLINC. In particular, on BANKING, we improve the ConvBERT + Combined baseline (Mehri and Eric, 2021) by 0.59% (absolute) and the RoBERTa-Large baseline by 0.72% (absolute). Compared with the ConvBERT + Combined, which is pre-trained on intent detection datasets in a self-supervised fashion and adds examples-driven training and specific model architectural design, our method achieves similar results with much simpler model design. Furthermore, our method is orthogonal to model architectures and can be integrated with any other approach for further improvement.

We also find that ICDA improves the performance of the RoBERTa-Large model on HWU and CLINC. This highlights the effectiveness of our method for enhancing intent detection models. Moreover, state-of-the-art performance on BANKING with the proposed method and RoBERTa-Large shows that our method is capable of generating high-quality augmentation examples to enhance the RoBERTa-Large model on the most fine-grained intent detection task.

Few-shot settings. In this setting we only use a small number of instances (datapoints) per class. We evaluate our method in both 5-shot and 10-shot settings and compare it with several strong baselines. Our proposed method outperforms all baselines on all datasets in both 5-shot and 10-shot settings. ICDA-M achieves the best performance in 5-shot settings on BANKING dataset and ICDA-XL achieves the best performance on HWU and CLINC datasets in 5-shot settings and on all datasets in 10-shot settings. All configurations of our method significantly improve the performance of a RoBERTa-Large model trained on any of the three datasets. Compared with CPFT (Zhang et al., 2021b), which utilizes contrastive learning for few-shot intent detection with extra data, our method achieves better performance without any additional human-annotated data. This showcases the advantage of our method for few-shot intent detection.

We also observe that our method consistently improves the performance of the baseline model as the number of synthetic datapoints increases from XS to XL. This indicates that the generated instances from our method can gradually cover more and more information of real instances and are capable of providing more useful information for model training.

6 Analysis and Discussion

In this section, we analyze the performance of ICDA and other approaches we tried. We first identify several factors that affect performance, and then present evidence that ICDA works by transferring knowledge from the pretrained generator to the task model. We then discuss a data-relabelling experiment and an experiment using uncertainty measures or data cartography (Swayamdipta et al., 2020) as filters.

6.1 Factors that Affect ICDA Performance

ICDA is effective at various training sizes. Throughout this work, we conduct experiments with different seed data sizes[4] to study the effect of

---

[4] By *seed data*, we mean data taken from each dataset, i.e. not synthetic data produced by ICDA.

	BANKING			HWU			CLINC		
Model	**5**	**10**	**Full**	**5**	**10**	**Full**	**5**	**10**	**Full**
RoBERTa-Base + Classifier	74.04	84.27	-	75.56	82.90	-	87.99	91.55	-
USE	76.29	84.23	92.81	77.79	83.75	91.25	87.82	90.85	95.06
CONVERT	75.32	83.32	93.01	76.95	82.65	91.24	89.22	92.62	97.16
USE+CONVERT	77.75	85.19	93.36	80.01	85.83	92.62	90.49	93.26	97.16
CONVBERT	-	83.63	92.95	-	83.77	90.43	-	92.10	97.07
+ MLM	-	83.99	93.44	-	84.52	92.38	-	92.75	97.11
+ MLM + Example	-	84.09	94.06	-	83.44	92.47	-	92.35	97.11
+ Combined	-	85.95	93.83	-	86.28	**93.03**	-	93.97	**97.31**
DNNC	80.40	86.71	-	80.46	84.72	-	91.02	93.76	-
CPFT	80.86	87.20	-	82.03	87.13	-	92.34	94.18	-
RoBERTa-Large + Classifier	78.99	86.08	93.70	74.44	84.11	92.13	89.89	93.56	96.80
+ **ICDA**-XS	80.29	86.72	-	81.32	85.59	-	91.16	93.71	-
+ **ICDA**-S	81.95	87.37	93.66	81.97	86.25	92.33	91.22	93.98	96.97
+ **ICDA**-M	**84.01***	88.64	93.73	81.84	87.36	92.12	91.93	94.71	97.06
+ **ICDA**-L	83.90	89.12	**94.42***	81.97	86.94	92.57	92.41	94.73	97.12
+ **ICDA**-XL	83.90	**89.79***	-	**82.45***	**87.41***	-	**92.62***	**94.84***	-

Table 2: Intent Detection Accuracy (in %) in few-/full-shot settings with augmented data from OPT-66B. Numbers in bold are the best results and numbers with * are statistically significant by t-test ($p < 0.05$) compared to the baselines (**5** / **10** examples per intent).

	Model	BANKING	HWU	CLINC
	RoBERTa-Large	86.08	84.11	93.56
	All	84.19	84.57	94.24
	All w/ relabeling	87.05	85.22	93.02
PVI	Global Low PVI	73.99	69.61	85.42
	Global High PVI	87.38	86.27	94.27
	Per-Intent Low PVI	76.49	71.84	89.33
	Per-Intent High PVI	**88.64**	**87.36**	**94.71**

Table 3: Intent Detection Accuracy (in %) for RoBERTa-Large model in 10-shot settings with ICDA-M synthetic instances from OPT-66B. Numbers in bold are statistically significant by t-test ($p < 0.05$). "All" represents using all synthetic data without PVI filtering. and "All w/ relabeling" represents using "All" and an oracle intent classifier to relabel the synthetic data.

training size. By looking at the results in Table 2, we observe that our proposed method consistently improves the accuracy of the downstream model in all training sizes. Also, as the training size decreases, we see that the ICDA improvement increases significantly. For example, on BANKING, the improvement goes from 0.72% in the full shot setting to 5.02% as the training size decreases to 5-shot. This indicates that ICDA is more effective when we have few training data available.

PVI filtering threshold. To study the effect of the threshold function ϵ, we conduct experiments with two different threshold functions: *Global*, and *Per-Intent*. *Global* means that the PVI threshold is the same for all intent classes, which is the average PVI value in the validation set. *Per-Intent* means that the PVI threshold is different for each intent class, which is the average PVI value under each intent class in the validation set. As a sanity check, we also conduct experiments using synthetic instances with PVI values lower than the threshold (*Low PVI)* as opposed to the normal (*High PVI)* instances.

We show the results in Table 3 (bottom half), where we see that *Per-Intent High PVI* filtering performs the best. Compared to using all synthetic training data without filtering (referred to as *All*), we see that *High PVI* filtering in general helps in improving accuracy. In BANKING, for example, when PVI filtering is applied with *Per-Intent High PVI*, the accuracy is 88.64% with 10-shot training size, which is significantly better than the result without PVI filtering (84.19%) – the same holds for the other two datasets. For the *Low PVI* conditions, we observe that performance drops significantly. This indicates that the model overfits on those examples that are not relevant to the desired intent. We discuss the *All w/ relabelling* condition in Section 6.3.

In Figure 2, we plot the F1 score against the PVI score of the test set instances grouped by intent, showing that some classes are harder than others, further supporting why we need a threshold per class rather than a global one.

6.2 Why Does ICDA Work?

PVI filtering discards mislabeled examples. We believe that the success of ICDA is because of not only the high diversity of the synthetic instances produced by the generator, but also the fact that PVI filtering effectively discards digressed instances. To verify this hypothesis,

Data	Prompt Label	Generated Sentence	PVI
BANKING	Refund not showing up (PVI Threshold: 5.79)	I didn't see my refund appear on my statement.†	**6.10**
		Where did your refund end up at? Please send it back immediately since I had a return and then refunded your purchase in full and I want it all returned back to my credit card.†	**5.81**
		Can we please cancel my return & resend my goods again	3.97
		Please confirm what is the reason for delay in payment for my purchase?	-3.86
HWU	alarm query (PVI Threshold: 4.28)	show me all the alarms and remind me about them†	**5.69**
		i want to be notified when alarm goes off.†	**4.36**
		how do i delete or disable alarms.	3.18
		list all the events on this date	-5.13
CLINC	accept reservation (PVI Threshold: 6.53)	does hanover steakhouse take reservations†	**6.74**
		are there any restaurants that take reservations for dinner in philadelphia†	**6.58**
		how many days prior is required for making reservations	5.39
		what time does bibiana's in greenwich open	−4.31

Table 4: Synthetic examples generated from OPT-66B. † indicates the sentences that belong to the same intent as the prompt label from our manual assessment; and bold denotes the PVI values over the threshold for given label.

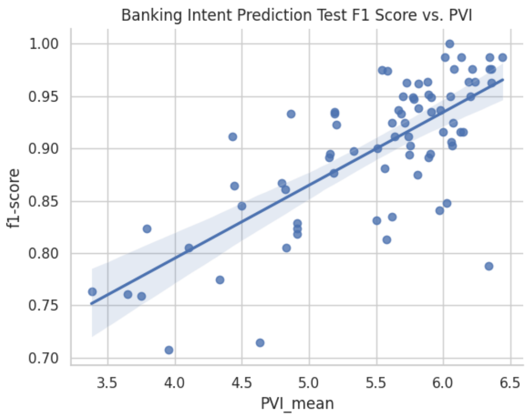


Figure 2: Intent Detection F1 score per intent class (circle) of the BANKING test set, justifying why we need a PVI threshold per intent.

Data	Split	D-1 ↑	D-2 ↑	Self-BLEU ↓	PPL ↓
Bank.	Test	-	-	-	12.14
	10-shot	0.15	0.54	0.24	**17.34**
	ICDA	**0.21**	**0.66**	**0.11**	21.33
HWU	Test	-	-	-	14.84
	10-shot	0.25	0.71	0.07	**26.97**
	ICDA	**0.30**	**0.78**	**0.03**	28.52
CLINC	Test	-	-	-	14.77
	10-shot	0.15	0.49	0.28	**34.23**
	ICDA	**0.20**	**0.60**	**0.17**	37.34

Table 5: Quantitative metrics of fluency and diversity of real and synthetic utterances in 10-shot settings as measured with distinct-1 (D-1), distinct-2 (D-2), self-BLEU, and perplexity.

we randomly sample several synthetic instances from the OPT-66B generator and manually assess if each instance follows the same intent as the prompt label. We show some examples in Table 4. We observe that instances that are relevant to the desired intent are assigned high PVI values, and instances that are not relevant to the desired intent are assigned low PVI values. This further indicates that the per-intent threshold function provides an effective indicator of relevance. For example, in the BANKING dataset, most relevant instances have PVI values greater than 5.79, and most non-relevant instances have PVI values less than 5.79. This indicates that PVI filtering is an effective method for discarding mislabeled data points.

ICDA produces fluent and diverse utterances. We hypothesize that our proposed method is effective because it introduces more fluent and diverse utterances. We therefore compare synthetic data under the 10-shot XS condition (i.e., we generate 10 synthetic datapoints) with the original 10-shot datapoints taken from the training data. Then we use a GPT2 model trained on the test set of each benchmark dataset to calculate the perplexity of the generated utterances. We also use the same synthetic set to calculate the distinct-1, distinct-2, self-BLEU, and perplexity (PPL) metrics. We report the results in Table 5 and observe that our proposed method generates more diverse utterances as shown by distinct-1, distinct-2, and self-BLEU. This indicates that our proposed method harnesses the generation power of the OPT-66B generator. Additionally, the perplexity of synthetic utterances is slightly higher than the human-annotated training set. These results suggest that our proposed method generates more diverse utterances, which can help the task model to learn a better representation.

6.3 Data Relabelling

Following Sahu et al. (2022), we wanted to see if it is effective to use the available data to train an intent classifier and then use it to relabel the synthetic data. Intuitively, such a method would correct mistakes in the generation process. To test the feasibility of this approach, we train an oracle classifier using the entire training data of each dataset and use this as an upper bound. The results are shown in Table 3 (“All w/ relabeling"), where we see that while promising, this approach underperforms ICDA.

7 Conclusion

We introduced **I**n-**C**ontext **D**ata **A**ugmentation, a novel data augmentation framework to generate synthetic training data, preserving quality and diversity. We demonstrate that ICDA is effective on multiple intent detection benchmarks, with state-of-the-art few-shot performance. Our analysis shows that ICDA tends to perform better in low-resource settings and that our PVI filtering strategy is important for performance. Future work includes applying ICDA to other conversational understanding tasks such as slot filling and dialogue state tracking, and incorporating other filtering or data selection strategies for further performance gains.

Limitations

In this section we take BANKING as a case study to motivate PVI and discuss some of the limitations of our approach. Figure 3 shows how much we gain (or lose) in F1 score when we use a custom threshold for each class vs. a fixed threshold. While most classes benefit, there are clearly many that show performance degradation. Another limitation is the size of the model we use to generate synthetic instances (OPT-66B); in general the larger the model is, the better the generated data is.

Ethical Considerations

As with any work involving PLMs (or *foundation models*), due to the data and training methods, there is inherent risk of generating biased, toxic, harmful, or otherwise unwanted output. Regarding our work in particular, as we show in Figure 3, the model's performance on some of the classes can degrade. More analysis needs to be done before deploying our approach, since it is unclear whether it will introduce a bias towards certain types of classes.

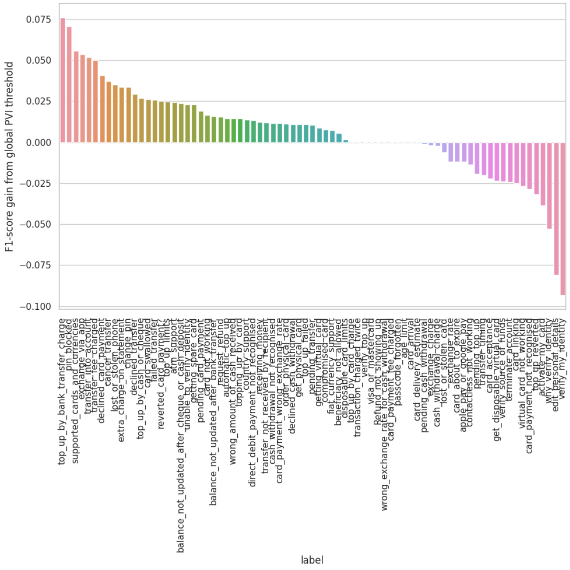


Figure 3: This figure shows the difference in Intent Detection F1 score for each intent, if we have a PVI threshold per-class VS having a fixed PVI threshold. See larger figure in Appendix.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7383–7390. AAAI Press.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21. ACL.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Tanja Bunk, Daksh Varshneya, Vladimir Vlasov, and Alan Nichol. 2020. DIET: lightweight language understanding for dialogue systems. *CoRR*, abs/2004.09936.

Hengyi Cai, Hongshen Chen, Yonghao Song, Cheng Zhang, Xiaofang Zhao, and Dawei Yin. 2020. Data manipulation: Towards effective instance learning for neural dialogue generation via learning to augment and reweight. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6334–6343, Online. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

Aron Culotta and Andrew McCallum. 2005. Reducing labeling effort for structured prediction tasks. In *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pages 746–751. AAAI Press / The MIT Press.

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR.

Amirata Ghorbani and James Y. Zou. 2019. Data shapley: Equitable valuation of data for machine learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2242–2251. PMLR.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5149–5156. AAAI Press.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, USA, June 24-27, 1990*. Morgan Kaufmann.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. *CoRR*, abs/2203.08568.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1875–1885. Association for Computational Linguistics.

Shailza Jolly, Tobias Falke, Caglar Tirkaz, and Daniil Sorokin. 2020. Data-efficient paraphrase generation to bootstrap intent classification and slot labeling for new features in task-oriented dialog systems. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 10–20, Online. International Committee on Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *CoRR*, abs/2003.02245.

Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. 2019. A closer look at feature space data augmentation for few-shot intent classification. In *Proceedings of the 2nd Workshop on*

Deep Learning Approaches for Low-Resource NLP (DeepLo 2019), pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Kenton Lee, Kelvin Guu, Luheng He, Tim Dozat, and Hyung Won Chung. 2021. Neural data augmentation via example extrapolation. *CoRR*, abs/2102.01335.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Fatema Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2021. Coco: Controllable counterfactuals for evaluating dialogue state trackers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019a. Benchmarking natural language understanding services for building conversational agents. In *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction - 10th International Workshop on Spoken Dialogue Systems, IWSDS 2019, Syracuse, Sicily, Italy, 24-26 April 2019*, volume 714 of *Lecture Notes in Electrical Engineering*, pages 165–183. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tong Luo, Kurt Kramer, Dmitry B. Goldgof, Lawrence O. Hall, Scott Samson, Andrew Remsen, and Thomas Hopkins. 2004. Active learning to recognize multiple types of plankton. In *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004*, pages 478–481. IEEE Computer Society.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shikib Mehri and Mihail Eric. 2021. Example-driven intent prediction with observers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2979–2992, Online. Association for Computational Linguistics.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tür. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *CoRR*, abs/2009.13570.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation. *CoRR*, abs/2202.00666.

Sören Mindermann, Jan Markus Brauner, Muhammed Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N. Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. 2022. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 15630–15649. PMLR.

Mahdi Namazifar, Alexandros Papangelis, Gökhan Tür, and Dilek Hakkani-Tür. 2021. Language model is all you need: Natural language understanding as question answering. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 7803–7807. IEEE.

Eda Okur, Saurav Sahay, and Lama Nachman. 2022. Data augmentation with paraphrase generation and entity extraction for multimodal dialogue system. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4114–4125, Marseille, France. European Language Resources Association.

Subhadarshi Panda, Caglar Tirkaz, Tobias Falke, and Patrick Lehnen. 2021. Multilingual paraphrase generation for bootstrapping new features in task-oriented dialog systems. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 30–39, Online. Association for Computational Linguistics.

Alexandros Papangelis, Karthik Gopalakrishnan, Aishwarya Padmakumar, Seokhwan Kim, Gokhan Tur, and Dilek Hakkani-Tur. 2021. Generative conversational networks. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 111–120, Singapore and Online. Association for Computational Linguistics.

Baolin Peng, Chenguang Zhu, Michael Zeng, and Jianfeng Gao. 2021. Data augmentation for spoken language understanding via pretrained language models. In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 1219–1223. ISCA.

Antoine Raux, Brian Langner, Dan Bohus, Alan W. Black, and Maxine Eskénazi. 2005. Let's go public! taking a spoken dialog system to the real world. In *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, September 4-8, 2005*, pages 885–888. ISCA.

Nicholas Roy and Andrew McCallum. 2001. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 441–448. Morgan Kaufmann.

Gaurav Sahu, Pau Rodriguez, Issam Laradji, Parmida Atighehchian, David Vazquez, and Dzmitry Bahdanau. 2022. Data augmentation for intent classification with off-the-shelf large language models. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 47–57, Dublin, Ireland. Association for Computational Linguistics.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden markov models for information extraction. In *Advances in Intelligent Data Analysis, 4th International Conference, IDA 2001, Cascais, Portugal, September 13-15, 2001, Proceedings*, volume 2189 of *Lecture Notes in Computer Science*, pages 309–318. Springer.

Greg Schohn and David Cohn. 2000. Less is more: Active learning with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 839–846. Morgan Kaufmann.

Richard Shin and Benjamin Van Durme. 2022. Few-shot semantic parsing with language models trained on code. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5417–5425. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020a. Generative data augmentation for commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020b. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. GPT3Mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Steve J. Young, Milica Gasic, Blaise Thomson, and Jason D. Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proc. IEEE*, 101(5):1160–1179.

Dian Yu, Luheng He, Yuan Zhang, Xinya Du, Panupong Pasupat, and Qi Li. 2021. Few-shot intent classification and slot filling with retrieved examples. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 734–749, Online. Association for Computational Linguistics.

Haode Zhang, Yuwei Zhang, Li-Ming Zhan, Jiaxin Chen, Guangyuan Shi, Xiao-Ming Wu, and Albert Y.S. Lam. 2021a. Effectiveness of pre-training for few-shot intent classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1114–1120, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021b. Few-shot intent detection via contrastive pre-training and fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1912, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5064–5082, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: open pre-trained transformer language models. *CoRR*, abs/2205.01068.

A Data Cartography and Uncertainty

Apart from relabelling, we investigated two additional approaches to rank synthetic instances as easy or hard to classify. We used data cartography (Swayamdipta et al., 2020) and classification uncertainty to guide our filtering.

Data cartography classifies the training data in four categories: Easy-to-learn, Low-Correctness, Ambiguous, Hard-to-Learn using training dynamics (i.e. the model's confidence in the true class, and the variability of this confidence across epochs).

For uncertainty modeling, we assign uncertainty scores to each training instance in a *cross-validation* manner. We first split the training set into 5 folds, hold one fold out as validation, and predict on the validation with the classifier trained on the remaining 4 folds. We tried the following uncertainty measures: Contrastive Active Learning (AL) (Margatina et al., 2021), Least Confidence (Culotta and McCallum, 2005), Prediction Entropy (Schohn and Cohn, 2000; Roy and McCallum, 2001), and Breaking Ties (Scheffer et al., 2001; Luo et al., 2004).

We conducted experiments using the above approaches to select data that amounts to one third of the total training data in BANKING (i.e., we select the top 33% hardest examples, etc.). As an additional baseline, we include a random filter, i.e., a randomly sampled 33% portion of BANKING. Table 6 shows the results, where we see that the performance actually degrades when compared to using the entirety of the data. We experimented with a few more variations in the filtering thresholds but no combination improved performance and we do not report those results here. See Figures 5 and 6 in the Appendix B for a visualization of the BANKING data map.

		100% Train	92.89
33% Train		Random	89.50
	Uncertainty	Contrastive AL	88.54
		Least Confidence	89.08
		Breaking Ties	89.20
		Prediction Entropy	89.23
	Cartography	Easy to Learn	90.44
		Ambiguous	90.94
		Low Correctness	91.00
		Hard to Learn	91.26

Table 6: Intent Detection Accuracy (in %) for ConvBERT model, trained on different selections of BANKING77 under full-shot settings.

B Figures

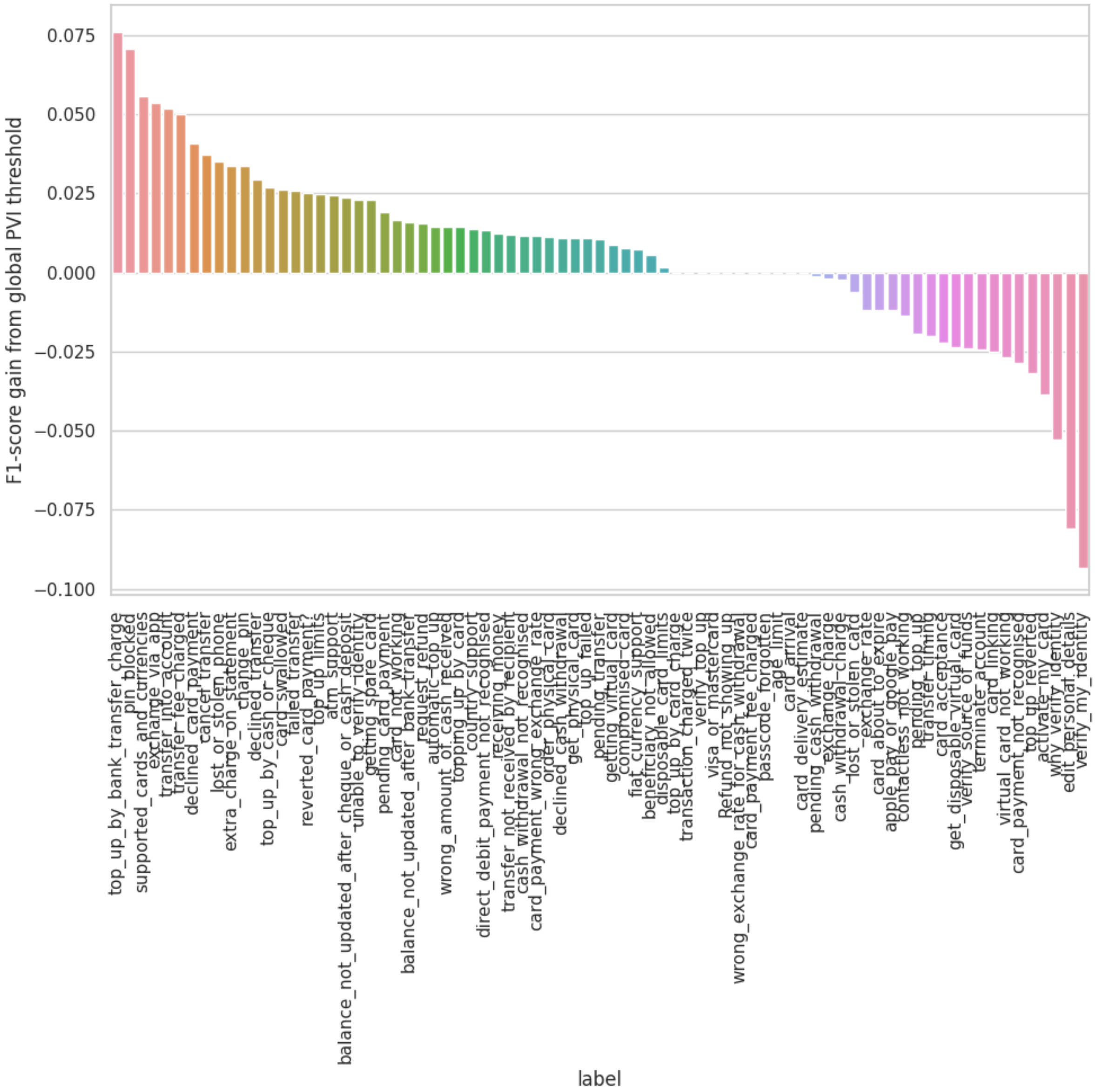


Figure 4: This figure shows the difference in F1 score for each intent, if we have a PVI threshold per-class VS having a fixed PVI threshold (Enlarged Figure 3).

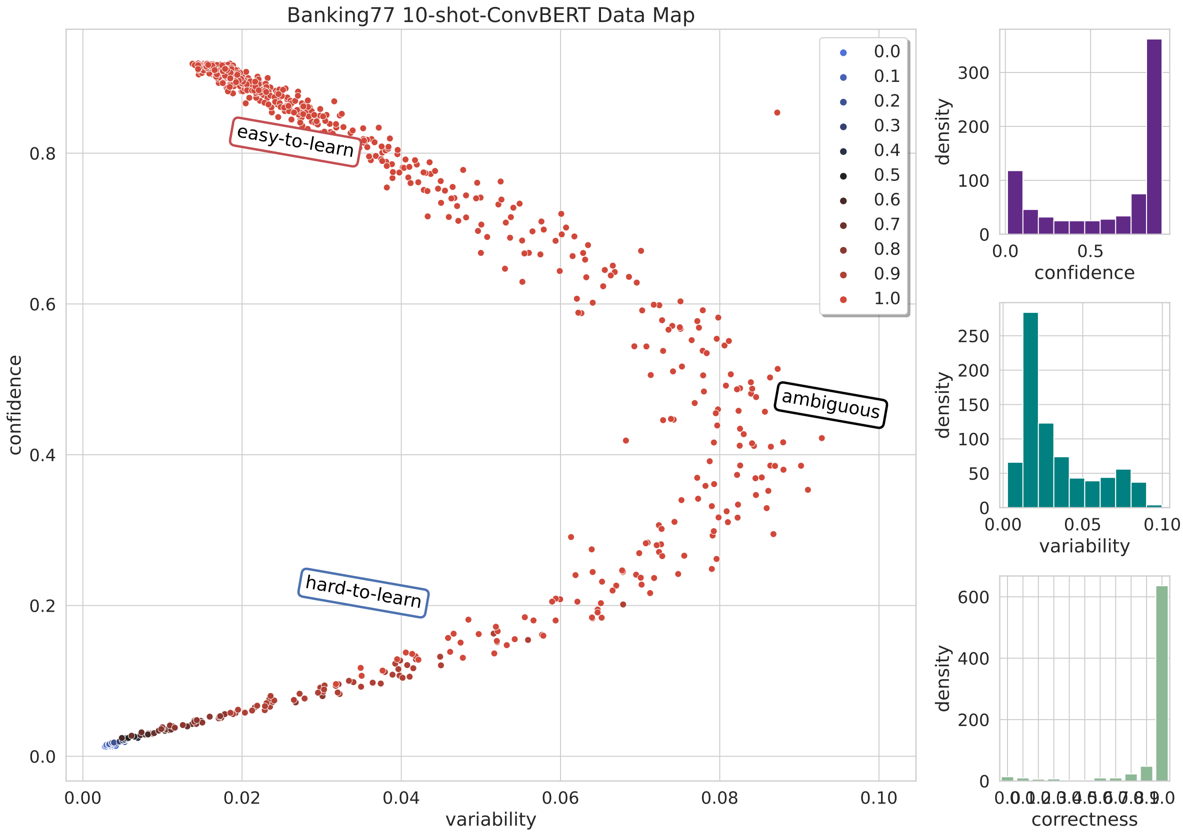


Figure 5: Data map for BANKING (10-shot).

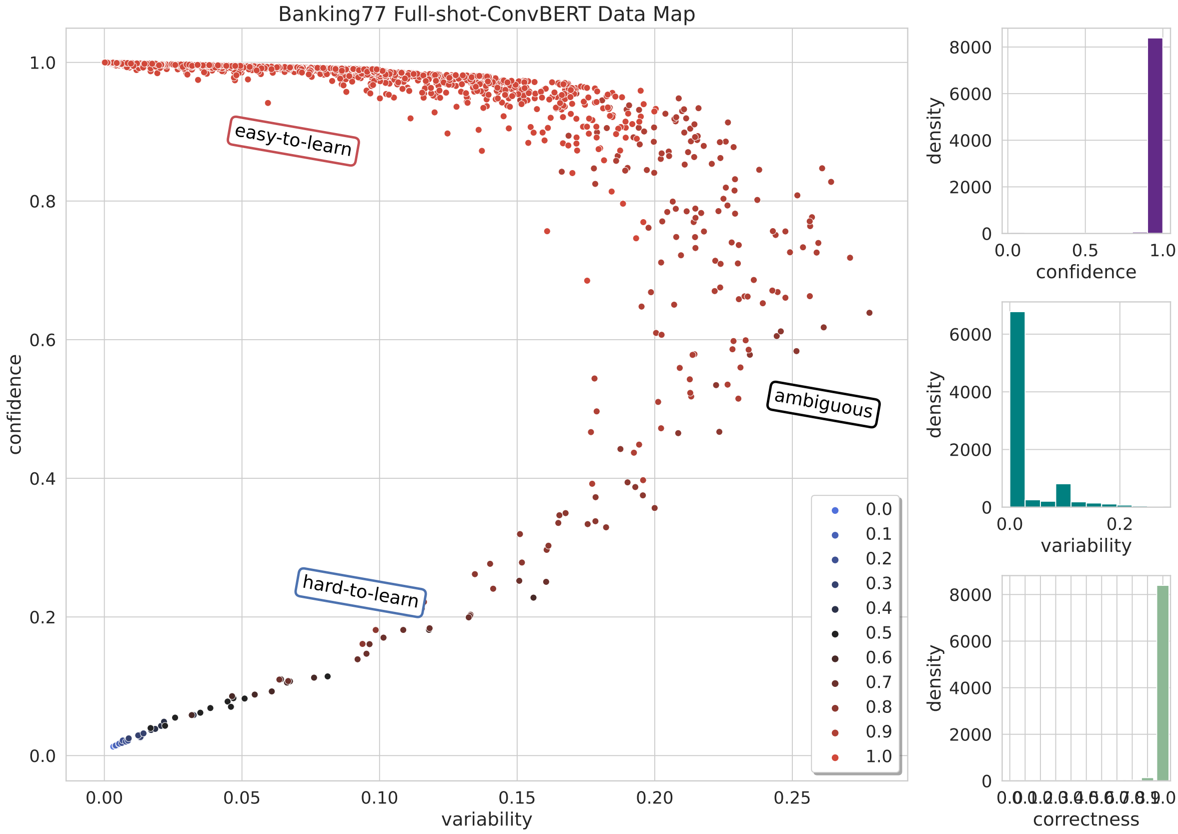


Figure 6: Data map for BANKING (full-shot).

Multilingual Representation Distillation with Contrastive Learning

Weiting Tan♠[*] **Kevin Heffernan**♡ **Holger Schwenk**♡ **Philipp Koehn**♠♡

♠Johns Hopkins University
♡Meta AI Research
`{wtan12, phi}@jhu.edu {kevinheffernan,schwenk}@meta.com`

Abstract

Multilingual sentence representations from large models encode semantic information from two or more languages and can be used for different cross-lingual information retrieval and matching tasks. In this paper, we integrate contrastive learning into multilingual representation distillation and use it for quality estimation of parallel sentences (i.e., find semantically similar sentences that can be used as translations of each other). We validate our approach with multilingual similarity search and corpus filtering tasks. Experiments across different low-resource languages show that our method greatly outperforms previous sentence encoders such as LASER, LASER3, and LaBSE.

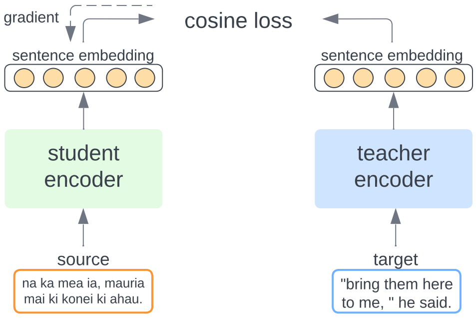


Figure 1: Student-Teacher Distillation in LASER3.

1 Introduction

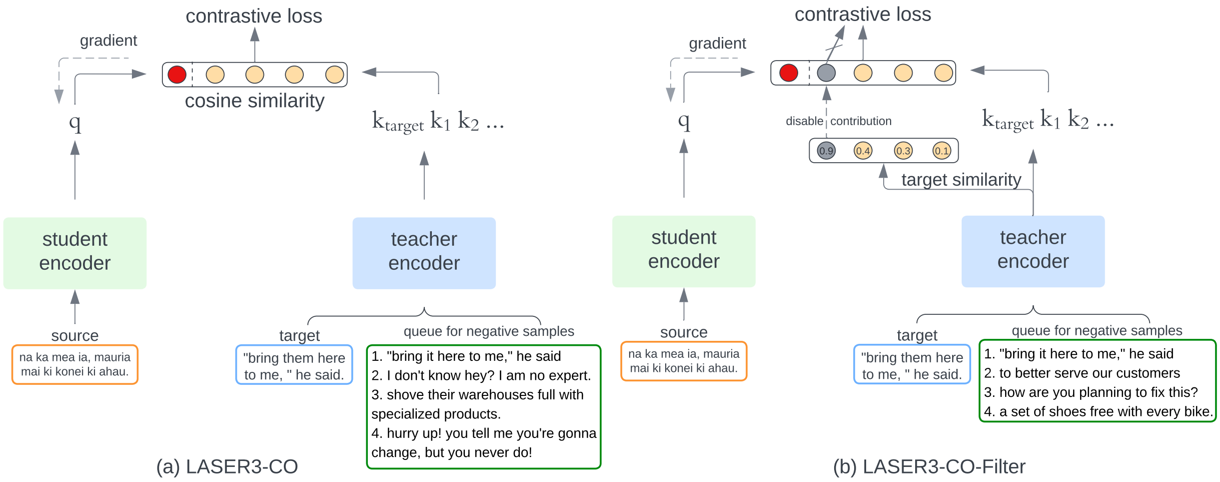
With the rise of neural networks, high-dimensional word-level sequence representations play an important role in almost any natural language processing task. Contextual representations from large pre-trained language models (Vaswani et al., 2017; Devlin et al., 2019; Liu et al., 2019; CONNEAU and Lample, 2019) have shown advantages compared to earlier static embeddings (Mikolov et al., 2013; Pennington et al., 2014). However, they are not pre-trained with sentence-level objectives and their representations of two different sentences cannot be easily used to indicate semantic similarity. To encode sentence-level information, LASER (Artetxe and Schwenk, 2019b) pools a sentence embedding from the encoder and feeds it to the decoder. Another approach is to use siamese-structured models where two identical encoders are used to represent sentences of similar meaning and are trained to push their representations close to each other. Sentence-Transformers (Reimers and Gurevych, 2019) are siamese-structured models that are initialized with pre-trained large models like BERT (Devlin et al., 2019) or Roberta (Liu et al., 2019). After fine-tuning, Sentence-Transformers improve their sentence representation for cross-lingual tasks. Besides fine-tuning with identical (siamese) encoders, distillation can be used to retrieve better representations. Reimers and Gurevych (2020a) extends a monolingual sentence representation into a multilingual representation with model distillation. Similarly, Heffernan et al. (2022) proposed LASER3, a student-teacher architecture that distills information from a pre-trained teacher encoder to a student encoder in new languages. As shown in Figure 1, the distillation process updates the student encoder with the gradient from the cosine loss. Note that it freezes the parameters of the teacher encoder, which is already pre-trained on the target language (English in our case). Therefore, the target sentence embedding is fixed and the corresponding source embedding is aligned with the target embedding.

In this paper, we focus on the quality estimation of parallel sentences in low-resource languages, which requires models to distinguish similar and dissimilar sentence pairs. Contrastive learning is helpful because its objective not only aligns similar sentences' representations but also pushes away representations from dissimilar sentences, which

---

[*] Work was done during an internship at Meta AI Research.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1477–1490
May 2-6, 2023



(a) LASER3-CO

(b) LASER3-CO-Filter

Figure 2: Visual Explanation of LASER3-CO (vanilla) and LASER3-CO-Filter (filtered). Source and target come from input bitexts dataset and queue is constructed with earlier batch's samples. In the (b) filtered version, a pre-filtering mechanism is employed to filter out extremely negative samples from the queue.

enables the model to be more confident in similar sentences. Additionally, contrastive learning is a form of self-supervision that fits well in our low-resource setting where only a limited amount of clean data is available.

In practice, we integrate contrastive learning into the distillation-based architecture from LASER3 and use our contrastive distillation method to train encoders for low-resource languages. Inspired by He et al. (2020) and Wu et al. (2018), we used a queue to store negative samples as self-supervision to train better encoders. We also employed a pre-filtering mechanism to find hard negative samples and showed that they benefit the distillation of sentence representations.

To evaluate different encoders, we rely on multilingual similarity search and corpus filtering tasks. In the multilingual similarity search task, we encode all source and target sentences and use a cosine-based similarity metric called margin score (Artetxe and Schwenk, 2019a) to pair source and target sentences. In the corpus filtering task, a mined noisy parallel corpus is given and we use the encoder with margin score to compute a similarity score for each sentence pair. Then we filter out the low-score pairs and use the remaining corpus to train a neural machine translation model whose performance is evaluated with BLEU (Papineni et al., 2002). Compared to previous works, we observe consistent improvement when using contrastive distillation. We also compared our approach with another simple but effective data augmentation technique, back-translation, and show that contrastive distillation achieves tied or better performance with less data.

2 Approach

To train high-quality sentence representation, we integrate contrastive learning into sentence representation distillation and our contrastive distillation method is visualized in Figure 2. The motivation for using contrastive learning is two-fold:

- The self-supervision from contrastive learning helps representation learning in low-resource settings.
- Contrastive learning enables models to recognize similar and dissimilar sentences, which is crucial for filtering out noisy sentence pairs.

2.1 LASER3-CO

This architecture corresponds to Figure 2 (a). We name it LASER3-CO since it integrates contrastive learning to LASER3's distillation pipeline. We are inspired by He et al. (2020) to use a queue to store contrastive (negative) samples. The negative samples are used to train encoders so that sentences of different meaning have dissimilar representations. During training, we use previous batches' target language sentences as negative samples and when there are too many negative samples in the queue, we remove the samples from the earliest batches. Our LASER3-CO approach has the following steps:

- Pre-train LASER to use as the teacher encoder θ_t on high resource languages such as English. Randomly initialize the student encoder and perform distillation with the teacher encoder. After distillation, we obtain the pre-trained student encoder θ_s.

- Fine-tune θ_s using queue(memory)-based[1] contrastive learning. For each input source sentence x and target sentence y, encode them with student and teacher encoders respectively and we have their representations $q = \theta_s(x), k_{\text{target}} = k_+ = \theta_t(y)$ (we use "+" as an abbreviation for the positive target sentence). We also encode all (N) negative sentences in the queue and have their representation $k_i = \theta_t(\text{queue}_i), i \in [1, N]$.

- Perform normalization on $q, k_i, i \in [0, N]$ and then compute the contrastive loss using infoNCE (van den Oord et al., 2018)

$$\mathcal{L} = -\log \frac{\exp(q \cdot k_+/\tau)}{\sum_{i=1}^{N} \exp(q \cdot k_i/\tau)} \tag{1}$$

  Here τ is the temperature parameter[2] that controls the strength of the penalty.

- Update student encoder θ_s with loss. Enqueue the most recent target language (English) sentences and dequeue the earliest sentences if the queue size exceeds the limit (N).

In LASER3-CO, our training process is very similar to MOCO (He et al., 2020), though we do not use the momentum update for the teacher. Instead, we freeze the pre-trained teacher (LASER in our case) that already has a high-quality representation of English (or any other pivot language), so that English sentences are encoded consistently during distillation. Then we try to align the representation of the student encoder to the teacher, only allowing gradients to flow through the student encoder.

2.2 LASER3-CO-Filter

This architecture corresponds to Figure 2 (b). The motivation is that previous work (Robinson et al., 2020) has shown hard negatives improve representation. In our task, for each parallel sentence pair (x, y), the hard negatives would be sentences y' that are hard to distinguish from the true target y (we can also find hard negatives x' in the source language, but because our teacher encoder has better representation for the target language, English, we decide to focus on target-side hard negatives only). Hard negatives are beneficial because they force the model to learn more complex features to distinguish them from the true target sentence.

To find more hard-negatives for contrastive fine-tuning, we change LASER3-CO in two ways: (1) disable shuffling for the data loader and (2) use a pre-filtering mechanism to filter out bad samples from the queue (we name this model LASER3-CO-Filter where Filter refers to the pre-filtering mechanism). Bad samples are hard negative samples y' that are too similar to y (e.g. "What is your name" versus "What's your name"). Treating these extremely similar sentences as negative samples would hurt the encoder's representation and we devise a pre-filtering method to filter them out.

Disable Shuffling We sort sentences by length[3] and disable the shuffling of data loader so that consecutive batches contain sentences of similar length. As our queue is updated by enqueuing the most recent batch and dequeuing the earliest batch, disabling shuffling makes the queue store sentences of similar length. Because all samples in the queue are used as negative samples to be contrasted with the true target sentence and because they are of a similar length, it is much more likely that we find hard negatives (quantitative analysis provided in §5 and Figure 4).

Pre-filtering Mechanism After disabling shuffling, we show that more hard-negatives are found. However, we also found that there are many cases of extremely hard negatives (e.g. "What is your name" versus "What's your name", "Bring them here to me" versus "Bring it here to me"). Though hard negatives help contrastive fine-tuning to obtain better sentence representation, extremely hard negatives would confuse the model and incorrectly update the parameters. Therefore we employ a simple pre-filtering mechanism to prevent these extremely hard negatives from contributing to the contrastive loss: After we encode the target sentence

---

[1] We use queue size N=4096 throughout experiments.

[2] Empirically we found $\tau = 0.05$ (widely used in previous literature) works well.

[3] Our implementation is based on Fairseq (Ott et al., 2019) which by default sorts the sentences by length

and sentences in the queue, we have the representations $k_+, k_1, \cdots k_N$ as described in §2.1. Then we compute the cosine similarity between the true target and sentences in the queue and get similarity scores $\cos(k_+, k_i), i \in [1, N]$. For any negative sample i, if $\cos(k_+, k_i) \geq \sigma$ (where σ is a predefined threshold hyper-parameter),[4] we do not use it in the InfoNCE loss. In other words, let $S = \{i : \cos(k_+, k_i) < \sigma\}$ be a set that contains no extremely hard negatives for the threshold σ, the InfoNCE loss would be computed as

$$\mathcal{L} = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i \in S} \exp(q \cdot k_i / \tau)} \tag{2}$$

In practice, we perform this filtering step with batches of samples, as detailed in appendix A. PyTorch-like pseudocode of the LASER3-CO-Filter model is shown below.

Algorithm 1 Pseudocode of LASER3-CO-Filter

```
# N: queue size;   bz: batch size
# S: size of set without extremely hard negatives
# t: temperature;   h: filtering threshold
# emb_dim: embedding dimension
# einsum: einsum function available in PyTorch
# queue: stores N earlier target embed
# teacher: pre-trained LASER encoder
# student: LASER3 student encoder
# filter: function that filters out extremely hard
     negatives (details available in appendix A)

# freeze teacher encoder
teacher.params.requires_grad = False
for (source, target) in loader:
    src_emb = normalize(student(source),dim=1)
    tgt_emb = normalize(teacher(target),dim=1)
    neg_emb = normalize(teacher(queue),dim=1)
    #--------- Begin Pre-filtering ---------
    # bz x N (mask for indices to be kept)
    mask = (tgt_emb @ neg_emb.T < h).long()
    # bz x S x emb_dim
    neg_emb = filter(neg_emb, mask)
    #--------- End Pre-filtering ------------
    #postive logits: bz x 1
    l_pos = src_emb @ tgt_emb.T
    # negative logits: bz x S
    l_neg = einsum("bh,bsh->bs", src_emb, neg_emb)
    logits = cat([l_pos, l_neg],dim=1)
    loss = CrossEntropyLoss(logits/t, zeros(bz))
    loss.backward()
    update(student)
    enqueue(queue, tgt_emb)
    if queue.size > N:
        dequeue(queue)
```

3 Experiment on Low-resource Languages

3.1 Dataset

For training, we use two sources of data: existing clean data and back-translation data.

ISO	Language	CB[k]	BT[k]
khm	Khmer	536	2451
pus	Pashto	48	3000
npi	Nepali	533	3000
sin	Sinhala	752	3000

Table 1: The number of (thousands of) sentences available for different languages in clean bitext (CB column) and back-translation (BT column) datasets. We cap back-translation data to 3 million lines maximum.

Clean data Our clean data comes from publicly available datasets[5] such as *jw300*, *bible* (Christodoulopoulos and Steedman, 2015), *tatoeba* (Tiedemann, 2012), *wikimedia* (Tiedemann, 2012), *gv* (Tiedemann, 2012), *tico19* (Tiedemann, 2012) , *ted20* (Reimers and Gurevych, 2020b), *qed* (Abdelali et al., 2014; Tiedemann, 2012), and *os* (Tiedemann, 2012).[6] We group these data sources together and call them clean data.

Back-translation data Back-translation is a simple yet effective data augmentation technique that could benefit the training of sentence encoders. To generate back-translation data, we need a translation model and monolingual data. We retrieve monolingual data from web data provided by CommonCrawl[7] and ParaCrawl (Bañón et al., 2020), following the pre-processing pipeline (language ID and heuristic filtering) from NLLB (NLLB Team et al., 2022). Then we use the 1.3B-parameter NLLB200-Dense[8] model (NLLB Team et al., 2022) to translate low resource language's monolingual data that we collected from web. We choose NLLB200 because it achieves state-of-the-art translation performance so far for most low-resource languages. Once we generate synthetic data, we can use them as a parallel corpus to distill sentence representation. In practice, we cap the amount of back-translation data at 3 million lines.

3.2 Languages & Models

We target four low-resource languages (Khmer, Pashto, Nepali, and Sinhala) in our experiments as shown in Table 1. For each of these languages,

---

[4] We use $\sigma = 0.9$ throughout our experiments. Effects of different σ values are analyzed in Figure 4 (right)

[5] All available online, mostly taken from OPUS: `https://opus.nlpl.eu/`

[6] `https://www.opensubtitles.org/`

[7] In wet format, `https://commoncrawl.org`

[8] Open-sourced at: `https://github.com/facebookresearch/fairseq/tree/nllb`

we have trained several models shown below:

- LASER (Artetxe and Schwenk, 2019b): we directly take the model checkpoint available publicly[9] without any further training.
- LASER3 (Heffernan et al., 2022): we follow Heffernan et al. (2022) to distill an encoder for each language using **clean data**. The teacher encoder is LASER and the student encoder is a randomly initialized transformer's encoder.
- LASER3-BT: same as the LASER3 model except that we distill the encoder using both **clean data** and **back-translation data**. Though back-translation is already widely used, our improvement from this model provides another instance where data augmentation from a large pre-trained multilingual translation model is effective.
- LASER3-CO: we fine-tune the LASER3 model following §2.1 with **clean data**.
- LASER3-CO-Filter: we fine-tune the LASER3 model following §2.2 with **clean data**.

3.3 Multilingual Similarity Search

Multilingual similarity search evaluates the quality of representation from trained encoders by aligning source and target sentences. We use FLORES (Guzmán et al., 2019) devtest dataset (1012 aligned sentences) for this task because it is crafted with professional human annotation. For every source sentence x_i, we compute a similarity score called `xsim` between x_i and all target sentences y_j. If the target sentence with the highest similarity score is not the i^{th} sentence (namely if $i \neq \text{argmax}_j \texttt{xsim}(x_i, y_j)$), then the model has one incorrect match. We compute the percentage of source sentences that have an incorrect match as `xsim` error rate. The lower the `xsim` error rate, the better the model is for aligning sentences of similar meaning. Our similarity score `xsim` is the margin-based score (Artetxe and Schwenk, 2019a), which is widely used for scoring and filtering parallel sentences. It is defined below:

$$\text{xsim}(x, y) = \text{margin}(\cos(x, y), \sum_{z \in \text{NN}_k(x)} \frac{\cos(x, z)}{2k} + \sum_{z \in \text{NN}_k(y)} \frac{\cos(y, z)}{2k}) \quad (3)$$

Model	khm	pus	npi	sin	avg
LASER	14.6	67.8	27.0	0.1	27.4
LASER3	0.4	0.3	5.9	0.3	1.7
LASER3-BT	**0.1**	**0.0**	**0.4**	0.1	**0.2**
LASER3-CO	0.3	0.1	0.7	**0.0**	0.3
LASER3-CO-Filter	**0.1**	0.1	0.8	0.1	0.3

Table 2: `xsim` error rate (%) for Khmer, Pashto, Nepali, and Sinhala. Bolded values are the lowest error rate obtained for each language.

where x, y are source and target sentences' representations from the model, $\text{NN}_k(x)$ denotes k nearest neighbors of x in the other language, and there are three margin functions[10]: absolute$(a, b) = a$, distance$(a, b) = a - b$, and ratio$(a, b) = \frac{a}{b}$.

We show our result in Table 2 and observe that LASER3 achieves a much better result than LASER (1.7% versus 27.4%), consistent with Heffernan et al. (2022)'s finding that language-specific distillation is more helpful than a language-agnostic encoder. We also show that using data augmentation (either with back-translation or with contrastive learning) further improves the performance to a near-zero error rate. In Figure 3, we provide the t-SNE visualization of embedding spaces trained by different approaches for Sinhala, and we observe that using back-translation or contrastive learning brings aligned sentences closer.

Note that in both of our contrastive-based models (LASER3-CO and LASER3-CO-Filter), we do not include back-translation data. Thus, contrastive distillation achieves tied performance compared to the model helped by a large amount of back-translation data, showing the power of self-supervision in representation learning. Additionally, back-translation and contrastive learning are orthogonal so we could train LASER3-CO/LASER3-CO-Filter with back-translation data. However, in our preliminary study, we do not see improvement from such an approach (more discussion in Appendix C) and we use only clean data for contrastive distillation throughout our experiments.

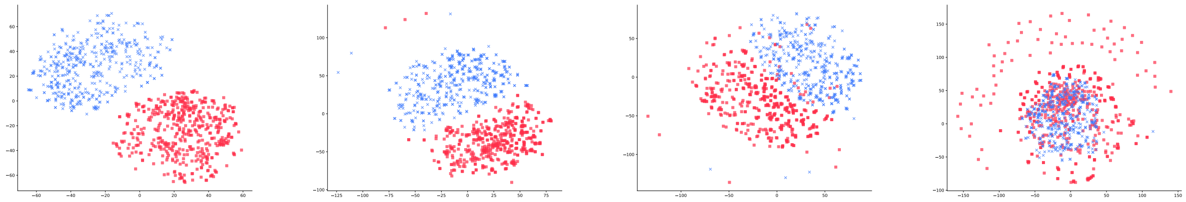
3.4 Corpus Filtering

We pick Khmer, Pashto, Nepali, and Sinhala as our targeted low-resource languages partly because they are used by WMT Corpus Filtering Shared task (Koehn et al., 2019, 2020). The shared task re-

---

[9] https://github.com/facebookresearch/LASER

[10] We follow Heffernan et al. (2022) and NLLB Team et al. (2022) to use ratio as the margin.



(a) LASER3 (b) LASER3-BT (c) LASER3-CO (d) LASER3-CO-Filter

Figure 3: t-SNE visualizations of embedding space for Sinhala-English. Red points are from Sinhala sentences and blue points are from English sentences.

Model	khm	pus	npi	sin
†LASER	5	5	N/A	N/A
†WMT Best System	5	5	1	1
LASER3	7	5	1	2
LASER3-BT	7	5	2	5
LASER3-CO	7	5	2	5
LASER3-CO-Filter	7	7	2	5

Table 3: Optimal subsample size (in millions of tokens) for corpus filtering task. For most cases, 5 or 7 million tokens give the best performance except for Nepali, where only 1 or 2 million tokens seem to be useful. †: number taken from findings of WMT corpus filtering shared task (Koehn et al., 2019, 2020).

Model	khm	pus	npi	sin	avg
†LASER	7.1	9.7	N/A	N/A	
†WMT Best System *	8.9	10.9	**6.9**	6.5	8.30
LASER3	**10.1**	11.7	5.1	5.8	8.18
LASER3-BT	**10.1**	11.8	6.2	**7.7**	8.95
LASER3-CO	9.9	**12.3**	6	7.3	8.88
LASER3-CO-Filter	**10.1**	12.2	6.2	**7.7**	**9.05**

Table 4: BLEU score for Khmer, Pashto, Nepali, and Sinhala. Bolded values are the highest BLEU scores obtained for each language. $*$: The highest number achieved by WMT19 & 20 corpus filtering task participants. †: results taken from findings of WMT19 & 20 (Koehn et al., 2019, 2020).

lease mined corpora (from the Paracrawl[11] project) for Khmer, Pashto, Nepali, and Sinhala. For each language, we first filter the noisy corpus with the language id model from fasttext[12] (Joulin et al., 2017). We filter out the aligned sentence pairs whose source sentence is predicted to be English (because we only expect the target sentence to be English). Then we filter the corpus with our sentence encoders where we rely on `xsim` score to rank sentences and take the top k sentences to train NMT models. In practice, we follow the shared task to use data subset of 1/2/3/5/7 million English tokens, and the optimal subset size is shown in Table 3. The highest BLEU score achieved by different systems is shown in Table 4.

From the results, we observe that LASER3-based systems have similar BLEU scores for Khmer. One possible explanation is that LASER3 distillation already gives good representation so data augmentation does not make a noticeable improvement. On average, we find that data-augmented systems (LASER3-BT, LASER3-CO, LASER3-CO-Filter) greatly improve the BLEU score compared to previous works like LASER, LASER3, and best systems in the WMT Corpus Filtering shared task. Note that some systems from the shared task make use of large pre-trained models and even an ensemble of large models, so it is a strong baseline and we are able to outperform it by an average of 0.75 BLEU score, achieving the state-of-the-art corpus filtering performance using contrastive distillation on a small amount of clean data. The corpus filtering result is also consistent with the multilingual similarity search task, where encoders with lower `xsim` error rate perform better.

4 Experiment on Extremely Low-resource Languages

4.1 Languages & Dataset

In §3, we experiment on low-resource languages **individually** and show the improvement from contrastive distillation. There are many languages that have even less available data and are under-represented in the current literature. In our work, we focus on four extremely low-resource African languages that are extremely low-resource: Chokwe, Kamba, Kimbundu, and Umbundu. The amount of available clean and back-translation data is shown in Table 5 (with the same collection pro-

---

[11] `https://paracrawl.eu/`

[12] `https://fasttext.cc/docs/en/language-identification.html`

ISO	Language	CB[k]	BT[k]
cjk	Chokwe	40	166
kam	Kamba	58	74
kmb	Kimbundu	101	140
umb	Umbundu	234	366

Table 5: the number of (thousands of) sentences available for different African languages in clean bitext (CB column) and back-translation (BT column) datasets. We cap back-translation data to 3 million lines maximum.

Model	cjk	kam	kmb	umb	avg
LaBSE*	34.4	27.4	35.0	37.0	33.5
LASER3*	16.4	15.3	7.5	15.6	13.7
LASER3-BT	14.6	3.5	4.8	7.0	6.7
LASER3-BT-CO	**9.1**	**1.9**	1.3	**4.6**	**4.2**
LASER3-BT-CO-Filter	11.3	2.2	**1.2**	7.1	5.5

Table 6: `xsim` error rate (%) for Chokwe, Kamba, Kimbundu, and Umbundu. Bolded values are the lowest error rate obtained for each language. $*$: result taken from Heffernan et al. (2022).

cess described in §3.1). These four languages are from the Benue-Congo language family which has 25 languages in total. The details of their names and available data size are shown in Table 8 (in the appendix). Under such extremely low-resource settings, we show that contrastive distillation still produces state-of-the-art encoders for the multilingual similarity search task. We believe our encoders can be easily adapted to other information extraction tasks such as bitext mining through the pipeline described in CCMatrix (Schwenk et al., 2021), and we leave this to future work.

4.2 Multilingual Similarity Search

We evaluate the performances of difference encoders on the multilingual similarity search task following the same procedure in §3.3 and show `xsim` error rate in Table 6. For the language-agnostic model baseline, we replace LASER with LaBSE (Feng et al., 2022) because its pre-training data contains African languages (though not directly on the four languages we target) and achieves better performance than LASER. However, this language-agnostic model is still easily outperformed by LASER3-based models since the distillation of language-specific data is helpful.

For LASER3 and LASER3-BT, we leverage all data from the Benue-Congo family to distill the encoder. We are able to reproduce results from LASER3 (Heffernan et al., 2022) and our back-translation baseline further improves the average `xsim` error rate to 6.7%. For contrastive distillation models, we fine-tune LASER3-BT **individually**[13] for each target language following §2. Not like previous experiments with Khmer, Pashto, etc., we fine-tuned LASER3-BT instead of LASER3 as our preliminary study showed better performance with LASER3-BT. Therefore, different languages could have different optimal settings for data augmentation (more discussion in Appendix C), depending on the amount and quality of available data. Nevertheless, we show that our contrastive distillation framework greatly improves upon previous encoders, achieving state-of-the-art `xsim` error rates for all four languages we targeted.

5 Analysis

In this section, we focus on contrastive distillation methods described in §2 and provide an analysis of the effects of different hyper-parameters.

5.1 Effect of queue size

For all the previous experiments, we use batch size 32 and queue size 4096 to be consistent. In our preliminary work, we tried N=512/1024/2048/4096 for Sinhala and we do not observe a noticeable advantage for using a larger queue size. Still, we use the largest queue size that we could afford,[14] 4096, throughout experiments since a larger queue size won't hurt performance. To show that memory-based contrastive learning is effective, we compare it with in-batch fine-tuning and show the result in Table 7. We see that memory-based contrastive learning gives us an average of 0.4 BLEU improvement across four languages.

Model	khm	pus	npi	sin	avg
LASER3	**10.1**	11.7	5.1	5.8	8.18
LASER3-CO	9.9	**12.3**	**6**	**7.3**	**8.88**
LASER3-InBatch	9.8	11.4	5.8	6.9	8.48

Table 7: BLEU score for Khmer, Pashto, Nepali, and Sinhala. LASER3-InBatch and LASER3-CO have the same setup except that LASER3-CO uses a queue of 4096 samples to compute contrastive loss and LASER3-InBatch only uses samples from every batch to compute contrastive loss.

---

[13] Consistent with our experiment in §3.3, using only clean data works the best for contrastive distillation.

[14] We trained our models with NVIDIA A100 GPU.

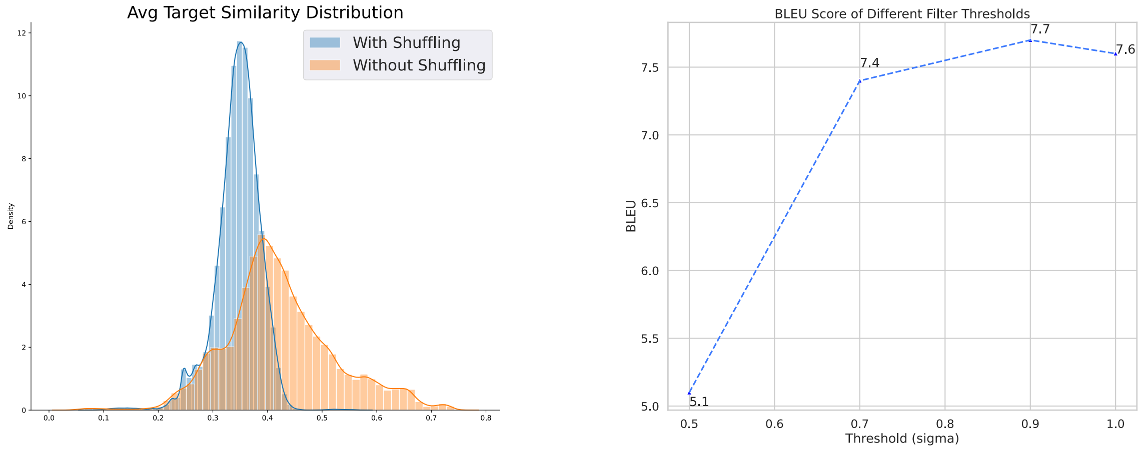


Figure 4: (left) Distribution plot of the average target similarity (of Sinhala's clean data) with/without shuffling. The x-axis is the value of average target similarity and the y-axis is the density of a particular value. (right) LASER3-CO-Filter's performance for Sinhala w.r.t varying pre-filter threshold σ. A low threshold filters out all hard negatives and harms training. A high threshold keeps extremely hard negatives which also undermines training.

5.2 Effect of shuffling

In our LASER3-CO-Filter approach, we disable shuffling so that the queue contains sentences of similar length and we argue that it is easier to find hard negatives this way. To quantify this fact, we compute the average target similarity for all sentences and plot the distribution in Figure 4 (left). The average target similarity for a target sentence x is

$$\text{sim}(x, \text{queue}) = \frac{1}{N} \cos(\theta_t(x), \theta_t(\text{queue}_i))$$

where θ_t is the teacher encoder, N is queue size, and queue_i is the i^{th} sample in the queue. As shown in Figure 4 (left), without shuffling, more than half samples have > 0.4 average target similarity. A higher average target similarity indicates that target sentences are more semantically related to the samples in the queue and therefore form harder negatives.

5.3 Effect of pre-filter threshold

For the LASER3-CO-Filter model, we employ a pre-filtering mechanism that prevents extremely hard examples from contributing to the loss. For our experiments, we use filter threshold $\sigma = 0.9$. In this section, we vary σ and evaluate the performance for Sinhala (we did not repeat this experiment for other languages but expect similar results). The result is shown in Figure 4 (right), which is a line plot based on our result for $\sigma = 0.5/0.7/0.9/1$. We can see that $\sigma = 0.9$ is the peak and the result corresponds to our intuition. When σ is too high (e.g. $\sigma = 1$), extremely hard negatives (or even repetitions) are not filtered out and their contribution to contrastive loss makes training unstable, resulting in poor performance for mining. When σ is too low (e.g. $\sigma = 0.5$), too many samples in the queue are filtered out and only easy negatives (sentences drastically different from the target sentence) are left in the queue. In this case, the model does not learn meaningful representations from negative samples, resulting in poor performance as well.

6 Related Work

Multilingual Sentence Representation Various multilingual sentence representations have been proposed including multilingual BERT (Devlin et al., 2019), XLM (CONNEAU and Lample, 2019), XLM-R (Conneau et al., 2020). These models are trained with a large amount of data across dozens or hundreds of languages. They show great improvement when fine-tuned on downstream tasks. However, they are trained only with token-level objectives like masked language modeling (MLM) or translation language modeling (TLM) and their sentence representations cannot be directly used for semantic similarity tasks. To address this problem, different methods make use of pooled sentence embedding to train better encoders (Artetxe and Schwenk, 2019b; Reimers and Gurevych, 2019; Feng et al., 2022; Heffernan et al., 2022).

Back-translation Back-translation is widely used in machine translation as a data augmentation technique. Sennrich et al. (2016) shows that synthetic data created by translating monolingual data greatly improve English-German and Turkish-English translation models. Edunov et al. (2018) further investigate different decoding strategies and show that sampling works better than greedy decode or beam search for back-translation. Since back-translation's quality depends on the translation model, Hoang et al. (2018) proposes iterative back-translation where translation models are further improved by the synthetic data they produce.

Contrastive Learning Contrastive learning is a form of self-supervised learning that encourages similar inputs to have close representations and dissimilar inputs to have different representations. Chen et al. (2020) proposes SimCLR that uses contrastive learning to improve visual representation. Khosla et al. (2020) integrates supervised learning into contrastive learning and Tian et al. (2020) uses contrastive distillation for image representation learning. Since finding high-quality contrastive samples is essential for contrastive learning, Wu et al. (2018) uses a memory bank to store contrastive samples while He et al. (2020) uses a queue with momentum update to encode contrastive samples on the fly. Though previously-mentioned works are evaluated on computer vision tasks, there is growing interest in using contrastive learning for natural language processing and various approaches have been proposed to train better sentence representation (Giorgi et al., 2021; Kim et al., 2021; Peng et al., 2020; Gao et al., 2021; Tan and Koehn, 2022).

7 Conclusion

In this paper, we integrate contrastive learning into multilingual sentence representation distillation. Through self-supervision from negatives, especially hard negatives, we obtain better encoders. Compared to other effective data augmentation techniques like back-translation, our contrastive distillation method achieves tied or better performance with less computation and data. Our experiments on multilingual similarity search and corpus filtering tasks show consistent improvement from contrastive learning and achieve state-of-the-art performance for various low-resource languages.

Limitations

This research focuses on improving cross-lingual encoders to perform quality estimation of parallel data for low-resource machine translation. However, cross-lingual representation could be helpful in many other tasks as summarized in the XTREME benchmark (Hu et al., 2020; Ruder et al., 2021) which we have not experimented with. Another limitation is that we disabled shuffling to store hard negative samples in the queue. However, without shuffling, it might affect the stability of the training process (although we do not observe it in our experiments). Therefore, in future work, we will continue exploring other cost-efficient approaches to retrieve hard negatives.

Acknowledgements

We thank anonymous reviewers for their valuable comments. We also thank Alex Guo, Huda Khayrallah, Kaushik Ram Sadagopan, Simeng Sun, Guillaume Wenzek, and Haoran Xu for their helpful suggestions.

References

Ahmed Abdelali, Francisco Guzman, Hassan Sajjad, and Stephan Vogel. 2014. The AMARA corpus: Building parallel language resources for the educational domain. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1856–1862, Reykjavik, Iceland. European Language Resources Association (ELRA).

Mikel Artetxe and Holger Schwenk. 2019a. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.

Mikel Artetxe and Holger Schwenk. 2019b. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L. Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, Sergio Ortiz Rojas, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Elsa Sarrías, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 2020. ParaCrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Christos Christodoulopoulos and Mark Steedman. 2015. A massively parallel corpus: the bible in 100 languages. *Language Resources and Evaluation*, 49:375 – 395.

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

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

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

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895, Online. Association for Computational Linguistics.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6098–6111, Hong Kong, China. Association for Computational Linguistics.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735.

Kevin Heffernan, Onur Çelebi, and Holger Schwenk. 2022. Bitext mining using distilled sentence representations for low-resource languages.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. 2021. Self-guided contrastive learning for BERT sentence representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long*

Papers), pages 2528–2540, Online. Association for Computational Linguistics.

Philipp Koehn, Vishrav Chaudhary, Ahmed El-Kishky, Naman Goyal, Peng-Jen Chen, and Francisco Guzmán. 2020. Findings of the WMT 2020 shared task on parallel corpus filtering and alignment. In *Proceedings of the Fifth Conference on Machine Translation*, pages 726–742, Online. Association for Computational Linguistics.

Philipp Koehn, Francisco Guzmán, Vishrav Chaudhary, and Juan Pino. 2019. Findings of the WMT 2019 shared task on parallel corpus filtering for low-resource conditions. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 54–72, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Wei Peng, Jianfeng Liu, Minghan Wang, Liangyou Li, Xupeng Meng, Hao Yang, and Qun Liu. 2020. Huawei's submissions to the WMT20 biomedical translation task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 857–861, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020a. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020b. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2020. Contrastive learning with hard negative samples.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, and Melvin Johnson. 2021. XTREME-R: Towards more challenging and nuanced multilingual evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10215–10245, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6490–6500, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Weiting Tan and Philipp Koehn. 2022. Bitext mining for low-resource languages via contrastive learning.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive representation distillation. In *International Conference on Learning Representations*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

A PyTorch-Style Pseudocode

In Algorithm 1, we provide the pythonic style pseudocode for our LASER3-CO-Filter model. Note that the pre-filtering mechanism needs special treatment when training with a minibatch. we compute the set of negative samples

$$S^j = \{i : \cos(k_+^j, k_i) < \sigma\}$$

for each target sentence j in our batch where k_+^j is the teacher encoding of j^{th} positive sample. Then we find the minimum number of negatives in the batch $M = \min_j |S^j|$. For any target sentence j that has $|S^j| > M$, we randomly remove $|S^j| - M$ samples so that it ends up with M samples. In this way, we ensure that every target sentence has the same number of legitimate negative samples after the pre-filtering mechanism and we can batch them to compute the InfoNCE loss. In the pseudocode, we abstract the implementation of pre-filtering using **filter** function.

B Corpus Filtering Threshold

In Table 3, we show the optimal size of filtered corpus to train NMT models for four languages. For most cases, 5 or 7 million tokens give the best performance (except for Nepali, where only 1 or 2 million tokens seem to be useful).

C Combine Back-translation and Contrastive Learning?

Back-translation and contrastive learning are two helpful ways for representation learning in low-resource settings. These two methods can be applied together (as we showed in our experiments on African languages) because back-translation prepares data before training while our contrastive distillation method is a fine-tuning strategy. Through our preliminary study, we made the following observations:

- Contrastive fine-tuning works best on a single language's clean data. This is true for both low-resource languages (Khmer, Pashto, Nepali, and Sinhala) and extremely low-resource languages (Chokwe, Kamba, Kimbundu, and Umbundu). We find that using back-translation data for contrastive objectives does not make any noticeable improvement and this is probably because synthetic data generated by the pre-trained NMT model does not produce too many hard negatives. In the case of the African languages, we tried distillation with the whole family's clean data and it is also less effective than distillation on a single language. This is probably because the data from multiple languages interfere with each other during contrastive distillation.
- For low-resource languages like Sinhala, Pashto, etc., we find no noticeable difference between contrastive distillation with LASER3 and LASER3-BT. On the contrary, for extremely low-resource languages, we find LASER3-BT-CO performs better than LASER3-CO, and this is why we used LASER3-BT-CO in §4.2 but LASER3-CO in §3.3. We believe this is due to the quality and the amount of available clean data. In the extremely low-resource case, even the limited amount of clean data contains noise and we see LASER3-BT greatly outperform LASER3. Therefore, the distillation with pre-trained LASER3-BT result in a better model (LASER3-BT-CO or LASER3-BT-CO-Filter). However, in some low-resource settings (like our experiments for Sinhala, Pashto, Khmer, and Nepali), clean data itself could allow the contrastive method to learn good representations, therefore avoiding the need for back-translation data.

With our observation above, how should we select methods when new languages come in? We believe it mainly depends on two factors: 1) How large and how clean is the existing clean corpus 2) How is the performance of the existing pre-trained NMT model?

The first factor affects the effectiveness of contrastive learning and the second factor affects the quality of back-translation data. In fact, in our preliminary study, we do not find back-translation from a statistical machine translation system useful for African languages, because it does generate good enough synthetic data. With the advance of multilingual machine translation (MMT) models, we are able to leverage a large pre-trained model and show its effectiveness. Thus, whether to add back-translation or not mainly depends on the translation model's quality and computation budget.

ISO	lang	CB[k]	BT[k]	ISO	lang	CB[k]	BT[k]
khm	Khmer	536	2451	pus	Pashto	48	3000
npi	Nepali	533	3000	sin	Sinhala	752	3000
cjk	Chokwe	40	166	kam	Kamba	58	74
kmb	Kimbundu	101	140	umb	Umbundu	234	366
bem	Bemba	700	828	ibo	Igbo	912	3000
kik	Kikuyu	119	95	kin	Kinyarwanda	517	3000
kon	Kongo	227	197	lin	Lingala	981	848
lua	Luba-Kasai	321	492	lug	Luganda	340	1654
nso	Northen Sotho	625	902	nya	Chewa;Nyanja	867	3000
run	Rundi	664	1910	sna	Shona	789	3000
sot	Sotho	1510	3000	ssw	Swati	116	432
swh	Swahili	2778	0	tsn	Tswana	1748	1680
tso	Tsonga	852	1423	tum	Tumbuka	264	675
xho	Xhosa	1739	3000	yor	Yoruba	517	3000
zul	Zulu	2383	3000				

Table 8: number of (thousands of) sentences available for different languages in clean bitext (CB column) and back-translation (BT column) dataset.

ISO	bible	gv	jw300	qed	tatoeba	ted20	tico19	wikimedia	os
khm	✓	✓	✓	✓	✓	✓	✓	✓	
pus				✓	✓	✓	✓	✓	
npi	✓	✓	✓	✓	✓	✓		✓	
sin	✓		✓	✓	✓	✓		✓	✓
cjk			✓						
kam			✓		✓				
kmb			✓						
umb			✓	✓	✓				

Table 9: Source of clean bitext data for different languages.

On the inconsistency of separable losses for structured prediction

Caio Corro
Universite Paris-Saclay, CNRS, LISN, 91400, Orsay, France
`caio.corro@lisn.upsaclay.fr`

Abstract

In this paper, we prove that separable negative log-likelihood losses for structured prediction are not necessarily Bayes consistent, or, in other words, minimizing these losses may not result in a model that predicts the most probable structure in the data distribution for a given input. This fact opens the question of whether these losses are well-adapted for structured prediction and, if so, why.

1 Introduction

Modern natural language processing (NLP) heavily relies on machine learning (ML), where prediction models are learned by minimizing a loss function over the training data. As such, loss functions play a central role in the design of these systems and it is important to understand their statistical properties in order to guarantee that the corresponding training objectives are well defined. Although this topic is well studied in the ML community (Lugosi and Vayatis, 2004; Lin, 2004; Zhang, 2004a,b; Bartlett et al., 2006; Gneiting and Raftery, 2007; Liu, 2007; Tewari and Bartlett, 2007; Reid and Williamson, 2010; Williamson et al., 2016; Duchi et al., 2018; Blondel et al., 2020; Nowak et al., 2022), *inter alia*, there has been less focus on the structured prediction setting apart from a few recent works (Blondel, 2019; Nowak et al., 2019, 2020).

In this paper, we emphasize the fact that, despite achievements in terms of accuracy, statistical behavior of loss functions used in practice for structured prediction in NLP are not always well understood. We illustrate this fact by proving that commonly used separable loss functions for named entity recognition (NER) and dependency parsing are not Bayes consistent, meaning that training a model with these loss functions will not necessarily result in the prediction of the most the probable output for a given input in the data distribution.

2 Bayes consistency

We denote inputs and outputs as $\boldsymbol{x} \in X$ and $\boldsymbol{y} \in Y$, respectively. We assume each $\boldsymbol{y} \in Y$ is a binary vector whose elements are indexed by a set C, *i.e.* $\boldsymbol{y} \in \{0,1\}^C$, where C is problem dependent. For example, in the k multiclass classification case, we have $C = [k]$, where we use the shorthand $[k] = \{1, 2, ..., k\}$, and Y is defined as the set of standard bases (one-hot vectors) of dimension k, meaning that $|Y| = k$. More generally, the vector $\boldsymbol{y}$ is an indicator of "selected" parts in C and, in the structured prediction case, several parts can be jointly selected. Note that it is usual to assume that the parts in C can depend on the input $\boldsymbol{x}$. Without loss of generality, we omit this detail as we will study loss functions in the pointwise setting.

A scoring model $f \in F$ is a function $f : X \to \mathbb{R}^C$ that returns scores associated with each part in C for a given input, *e.g.* the score of each class in a multiclass classification model. The actual prediction of the model is the output of maximum linear score:

$$\widehat{\boldsymbol{y}}(\boldsymbol{x}) \in \arg\max_{\boldsymbol{y} \in Y} \langle \boldsymbol{y}, f(\boldsymbol{x}) \rangle, \qquad (1)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. We refer to computing Equation 1 as maximum *a posteriori* (MAP) inference.

A loss function compares a vector of scores with an expected output. Importantly, the 0-1 loss function is defined as follows:

$$\ell(\boldsymbol{w}, \boldsymbol{y}) = \begin{cases} 0 & \text{if } \boldsymbol{y} \in \arg\max_{\boldsymbol{y}' \in Y} \langle \boldsymbol{w}, \boldsymbol{y}' \rangle, \\ 1 & \text{otherwise,} \end{cases}$$

where $\boldsymbol{w} \in \mathbb{R}^k$ is a vector of part scores, *i.e.* $\boldsymbol{w} = f(\boldsymbol{x})$ for a given input $\boldsymbol{x}$. In order to choose a scoring function $f \in F$, it is appealing to select one that minimizes this loss over the data distribution:

$$r^* = \inf_{f \in F} r(f) = \inf_{f \in F} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), \mathbf{y})],$$

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1491–1498
May 2-6, 2023

where $\mathbf{x}$ and $\mathbf{y}$ are random variables over inputs and outputs, respectively, and $\inf$ denotes the infimum. The value $r(f)$ is the Bayes risk of function f and r^* is the optimal Bayes risk. For theoretical purposes, it is often assumed that the class of functions F is rich enough (the set of all measurable mappings) to obtain the best possible risk. Then, the optimal Bayes risk is equal to:

$$r^* = \mathbb{E}_{\mathbf{x}}[1 - \max_{\boldsymbol{y} \in Y} p(\mathbf{y} = \boldsymbol{y} | \mathbf{x})],$$

or, in other words, it is the probability of making an error when the classifier predicts the most probable class for each input.

Unfortunately, in practice it is not convenient to use the 0-1 loss ℓ as it is nonconvex and has null derivatives almost everywhere. Instead, a surrogate $\widetilde{\ell}$ can be used as a loss function:

$$\widetilde{r}^* = \inf_{f \in F} \widetilde{r}(f) = \inf_{f \in F} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\widetilde{\ell}(f(\mathbf{x}), \mathbf{y})],$$

where $\widetilde{r}(f)$ is the surrogate risk of function f and $\widetilde{r}^*$ is the optimal surrogate risk. An important desired property of surrogate losses is their consistency with the 0-1 loss, *i.e.* the fact that minimizing the surrrogate risk leads to a prediction model of optimal Bayes risk (Lugosi and Vayatis, 2004; Lin, 2004; Zhang, 2004a; Bartlett et al., 2006; Liu, 2007; Tewari and Bartlett, 2007).

Definition 1. *A surrogate loss* $\widetilde{\ell}$ *is said to be Bayes consistent*[1] *if:*

$$f^* \in \arg\min_{f \in F} \widetilde{r}(f) \implies r(f^*) = r^*.$$

Note that this property can be checked independently for each input $\boldsymbol{x}$ (called pointwise Bayes consistency) as we assume a rich enough class of functions F. In other words, we redefine the pointwise (optimal) surrogate risk as:

$$\widetilde{r}^* = \inf_{\boldsymbol{w} \in \mathbb{R}^C} \widetilde{r}(\boldsymbol{w}) = \inf_{\boldsymbol{w} \in \mathbb{R}^C} \mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\widetilde{\ell}(\boldsymbol{w}, \mathbf{y})],$$

for any $\boldsymbol{x}$ such that $p(\mathbf{x} = \boldsymbol{x}) > 0$, and similarly for the optimal Bayes risk. The vector $\boldsymbol{w}$ should be interpreted as the model scores, *i.e.* $\boldsymbol{w} = f(\boldsymbol{x})$.

3 Negative log-likelihood loss

The negative log-likelihood loss (NLL), also known as the conditional random field loss (Lafferty et al., 2001), is defined as follows:

$$\widetilde{\ell}_{(nll)}(\boldsymbol{w}, \boldsymbol{y}) = -\langle \boldsymbol{w}, \boldsymbol{y} \rangle + \log \sum_{\boldsymbol{y}' \in Y} \exp \langle \boldsymbol{w}, \boldsymbol{y}' \rangle.$$

In the following, we will refer to computing the log-sum-exp term of the NLL loss as marginal inference due to its connection with marginal probabilities (Wainwright and Jordan, 2008).

Theorem 1. *Under mild conditions on the data distribution, the surrogate loss* $\widetilde{\ell}_{(nll)}$ *is Bayes consistent.*

Proof. The optimal pointwise surrogate Bayes risk is defined as:

$$\widetilde{r}^*_{(nll)} = \inf_{\boldsymbol{w} \in \mathbb{R}^C} \begin{array}{l} -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\langle \boldsymbol{w}, \mathbf{y} \rangle] \\ + \log \sum_{\boldsymbol{y}' \in Y} \exp \langle \boldsymbol{w}, \boldsymbol{y}' \rangle. \end{array}$$

We substitute $\boldsymbol{w}(\boldsymbol{y}) = \langle \boldsymbol{w}, \boldsymbol{y} \rangle$ for all $\boldsymbol{y} \in Y$:

$$= \inf_{\substack{\forall \boldsymbol{y} \in Y: \\ \boldsymbol{w}(\boldsymbol{y}) \in \mathbb{R}}} \begin{array}{l} -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\boldsymbol{w}(\mathbf{y})] \\ + \log \sum_{\boldsymbol{y}' \in Y} \exp \boldsymbol{w}(\boldsymbol{y}'). \end{array}$$

We denote $\widehat{\boldsymbol{w}}(\boldsymbol{y}), \forall \boldsymbol{y} \in Y$, an optimal solution of the minimization. By first order optimality conditions, we have:

$$\begin{aligned} &\frac{\partial}{\partial \widehat{\boldsymbol{w}}(\boldsymbol{y})} \begin{pmatrix} -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\widehat{\boldsymbol{w}}(\mathbf{y})] \\ + \log \sum_{\boldsymbol{y}' \in Y} \exp(\widehat{\boldsymbol{w}}(\boldsymbol{y}')) \end{pmatrix} = 0 \\ &\implies \frac{\exp \widehat{\boldsymbol{w}}(\boldsymbol{y})}{\sum_{\boldsymbol{y}' \in Y} \exp \widehat{\boldsymbol{w}}(\boldsymbol{y}')} = p(\mathbf{y} = \boldsymbol{y} | \mathbf{x} = \boldsymbol{x}) \end{aligned} \tag{2}$$

which implies Bayes consistency under the condition that there exists a vector $\boldsymbol{w} \in \mathbb{R}^C$ such that $\forall \boldsymbol{y} \in Y : \langle \boldsymbol{w}, \boldsymbol{y} \rangle = \widehat{\boldsymbol{w}}(\boldsymbol{y})$. ☐

To understand why Equation 2 implies Bayes consistency, note that:

$$\frac{\exp \widehat{\boldsymbol{w}}(\boldsymbol{y})}{\sum_{\boldsymbol{y}' \in Y} \exp \widehat{\boldsymbol{w}}(\boldsymbol{y}')} \propto \exp \widehat{\boldsymbol{w}}(\boldsymbol{y}'),$$

and the exponential function is strictly increasing. This means scores of outputs $\boldsymbol{y} \in Y$ defined as $\langle \boldsymbol{y}, \widehat{\boldsymbol{w}} \rangle$ are ordered in the same way as probabilities in the data distribution $p(\mathbf{y} | \mathbf{x} = \boldsymbol{x})$. In other words, the most probable output in the data distribution will have the highest score with respect to $\widehat{\boldsymbol{w}}$.

---

[1] This property is also referred to as Fisher consistency (Lin, 2004; Bartlett et al., 2006; Liu, 2007) and classification calibration (Williamson et al., 2016).

The proof is a straightforward extension of the derivation for the multiclass classification case, see for example Blondel et al. (2020, Section 4.2). Interestingly, Equation 2 also implies that the NLL loss is strictly proper (Williamson et al., 2016), *i.e.* the Boltzmann distribution over structures in Y parameterized by minimizer $\widehat{\boldsymbol{w}}$ is equal to the data distribution $p(\mathbf{y}|\mathbf{x}=\boldsymbol{x})$. Note that Theorem 1 is not novel *per se* and a more in-depth study of NLL losses for structured prediction can be found in (Nowak et al., 2019).

One limitation of the NLL loss is that it is not (additively) separable[2] because of the log-sum-exp term. As such, this term is a bottleneck for parallel computation of the objective and doubly stochastic estimation of the training objective (Titsias, 2016). A well-known solution is to rely on independent binary classification objectives, also known as one-vs-all losses (Blondel et al., 2020, Section 6.1):

$$\begin{aligned}&\widetilde{\ell}_{(one-vs-all)}(\boldsymbol{w}, \boldsymbol{y})\\ &\quad= -\langle \boldsymbol{w}, \boldsymbol{y}\rangle + \sum_{\boldsymbol{y}'\in Y} \log\left(1+\exp\langle \boldsymbol{w}, \boldsymbol{y}'\rangle\right).\end{aligned}$$

In the case of multiclass classification problems, it can be shown that this objective is Bayes consistent using similar arguments as in Theorem 1. A different approach is the one-vs-each loss function that is also Bayes consistent (Titsias, 2016).

Unfortunately, these separable surrogates cannot be applied to structured prediction problems as the set Y is often of exponential size with respect to the input length. Although tractable algorithms for marginal inference exist for many cases, there are no known algorithms to compute the one-vs-each or one-vs-all losses in an easily parallelizable fashion. As such, the NLP community often relies on token-separable losses, that is a NLL objective that decomposes as a sum of independent losses, one per token in the input sentence. Although these losses are easy to implement, we prove in the next sections that they are not Bayes consistent for two common NLP problems.

4 Named-entity recognition

Problem definition. In this Section, we focus on the flat NER problem using BIO tags (Ratinov and Roth, 2009). Without loss of generality, we assume there is a single mention label and that the input sentence $\boldsymbol{x}$ contains n words. The set of parts is defined as $C = [n] \times \{\mathrm{B}, \mathrm{I}, \mathrm{O}\}$ and Y is defined as the set of vectors $\boldsymbol{y} \in \{0,1\}^C$ satisfying the following conditions:

1. $\forall i \in [n] : \sum_t y_{i,t} = 1$ (one tag per word);
2. $y_{1,\mathrm{I}} = 0$ (forbid inside tag for the first word);
3. $\forall i > 1 : y_{i,\mathrm{I}} = 1 \implies y_{i-1,\mathrm{B}} + y_{i-1,\mathrm{I}} = 1$ (I tag can only follow a B or I tag).

We do not include parts corresponding to transitions (this is a unigram model), otherwise it would not be possible to derive a token-separable loss.

Inference algorithms. MAP and marginal inference can be realized using the Viterbi and the forward-backward algorithms, respectively. Although the time complexity of these algorithms is $\mathcal{O}(|L|^2 n)$ where L is the set of mention labels, they can be optimized to have a $\mathcal{O}(|L|n)$ time complexity as there is no transition score. The dynamic programming algorithm is nonetheless required in order to guarantee that condition (3) is satisfied.

Separable loss. As the dynamic programming algorithm is not parallelizable over input tokens, token-separable losses are often used in practice.[3] That is, the loss is reduced to a set of n multiclass classification losses:

$$\widetilde{\ell}_{\text{(sep-bio)}} = -\langle \boldsymbol{w}, \boldsymbol{y}\rangle + \sum_{i=1}^{n} \log \sum_t \exp w_{i,t},$$

where t ranges over all tags, except I if $i = 1$.

The optimal pointwise surrogate Bayes risk for the separable loss is defined as:

$$\widetilde{r}^*_{(sep-bio)} = \inf_{\boldsymbol{w}\in\mathbb{R}^C} \begin{array}{l} -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\langle \boldsymbol{w}, \mathbf{y}\rangle] \\ +\sum_{i=1}^{n} \log \sum_t \exp w_{i,t}. \end{array}$$

Let $\widehat{\boldsymbol{w}}$ be an optimal solution. Then, by first order optimality conditions:

$$\begin{aligned}&\frac{\partial}{\partial \widehat{w}_{i,t}} \left(\begin{array}{l} -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\langle \boldsymbol{w}, \mathbf{y}\rangle] \\ +\sum_{i=1}^{n} \log \sum_t \exp w_{i,t} \end{array} \right) = 0 \\ &\implies \widehat{w}_{i,t} = \log p\left(\mathrm{y}_{i,t} = 1 | \mathbf{x} = \boldsymbol{x}\right) \end{aligned} \tag{3}$$

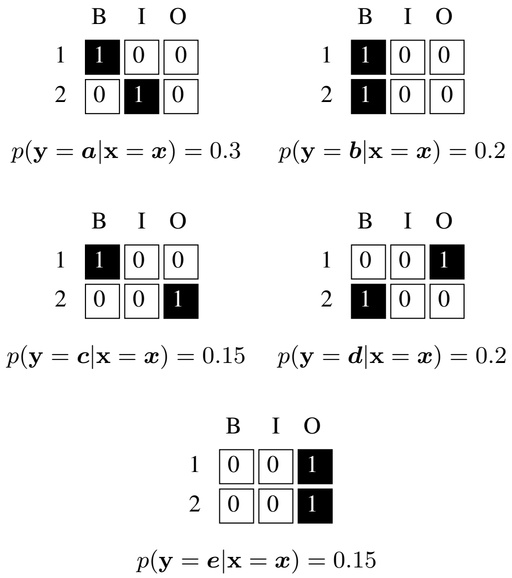
where $p\left(\mathrm{y}_{i,t} = 1|\mathbf{x}=\boldsymbol{x}\right)$ denotes the marginal distribution of tag t at position i in data distribution.

Theorem 2. *The token-separable loss for NER via BIO tagging is not Bayes consistent.*

---

[2] A function f is additively separable if it can be written as $f = \sum_i f_i$.

[3] See for example `https://github.com/huggingface/transformers/blob/v4.23.1/src/transformers/models/bert/modeling_bert.py#L1771`



$p(\mathbf{y} = \boldsymbol{a}|\mathbf{x} = \boldsymbol{x}) = 0.3$ $p(\mathbf{y} = \boldsymbol{b}|\mathbf{x} = \boldsymbol{x}) = 0.2$

$p(\mathbf{y} = \boldsymbol{c}|\mathbf{x} = \boldsymbol{x}) = 0.15$ $p(\mathbf{y} = \boldsymbol{d}|\mathbf{x} = \boldsymbol{x}) = 0.2$

$p(\mathbf{y} = \boldsymbol{e}|\mathbf{x} = \boldsymbol{x}) = 0.15$

Figure 1: Example of distribution over BIO sequences for a sentence of 2 words. The set of sequences is defined as $Y = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}\}$. The matrices represent values in elements of Y.

Proof. Let $n = 2$ and assume that the distribution $p(\mathbf{y}|\mathbf{x} = \boldsymbol{x})$ is defined as depicted in Figure 1. Then, by Equation 3 we have $\widehat{w}_{1,\text{B}} = \log 0.65$, $\widehat{w}_{1,\text{O}} = \log 0.35$, $\widehat{w}_{2,\text{B}} = \log 0.4$, $\widehat{w}_{2,\text{I}} = \log 0.3$ and $\widehat{w}_{2,\text{O}} = \log 0.3$. As such:

$$\begin{aligned}\log 0.65 + \log 0.3 &= \langle \widehat{\boldsymbol{w}}, \boldsymbol{a} \rangle \\ &< \langle \widehat{\boldsymbol{w}}, \boldsymbol{b} \rangle = \log 0.65 + \log 0.4,\end{aligned}$$

but $\boldsymbol{a} \in \arg\max_{\boldsymbol{y} \in Y} p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x})$ and $p(\mathbf{y} = \boldsymbol{a}|\mathbf{x} = \boldsymbol{x}) > p(\mathbf{y} = \boldsymbol{b}|\mathbf{x} = \boldsymbol{x})$. Therefore, the token-separable loss is not Bayes consistent for NER, *i.e.* a scoring model minimizing the surrogate risk may not lead to predicting the most probable output in the data distribution. $\square$

Note that the inconsistency is not due to the fact that the parameterization of the model is "poor" (no transition scores). Indeed, by Equation 2, optimal scores $\widehat{\boldsymbol{w}}$ for the NLL loss satisfy the following condition for all $\boldsymbol{y} \in Y$:

$$\begin{aligned}&\frac{\exp\langle \widehat{\boldsymbol{w}}, \boldsymbol{y} \rangle}{\sum_{\boldsymbol{y}' \in Y} \exp\langle \widehat{\boldsymbol{w}}, \boldsymbol{y}' \rangle} = p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x}) \\ &\implies \langle \widehat{\boldsymbol{w}}, \boldsymbol{y} \rangle = \log p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x})\end{aligned}$$

The following assignment for $\widehat{\boldsymbol{w}}$ satisfies this condition: $\widehat{w}_{1,\text{B}} = 0$, $\widehat{w}_{1,\text{O}} = 0$, $\widehat{w}_{2,\text{B}} = \log 0.2$, $\widehat{w}_{2,\text{I}} = \log 0.3$, and $\widehat{w}_{2,\text{O}} = \log 0.15$. That is, minimizing the NLL loss on this distribution results in a Bayes consistent classifier, as expected.

5 Syntactic dependency parsing

Problem definition. We consider a sentence of n words and, without loss of generality, restrict to the unlabeled case to simplify notations. In dependency parsing, the set of parts is defined as the set of possible bilexical dependencies between words, including a fake root at position 0 used to identify root word(s) of the sentence, *i.e.* $C = \{(h, m) \in \{0, 1, ..., n\} \times [n] | h \neq m\}$, where (h, m) denotes a dependency with the h-th word as head and the m-th word as modifier. The set Y is restricted to vectors $\boldsymbol{y} \in \{0, 1\}^C$ that can be interpreted as forming a 0-rooted spanning arborescence where words are vertices and dependencies are arcs (McDonald et al., 2005). In some cases, *e.g.* the Universal Dependency format, it is required that the fake root position has a single outgoing arc.[4]

Inference algorithms. MAP inference can be realized via the maximum spanning arborescence algorithm, which has a $\mathcal{O}(n^2)$ time complexity (Chu and Liu, 1965; Edmonds, 1967; Tarjan, 1977). The single root constraint can be taken into account using the same algorithm via the big-M trick (Fischetti and Toth, 1992, Section 2). Marginal inference can be realized via the matrix tree theorem (MTT, Koo et al., 2007; McDonald and Satta, 2007; Smith and Smith, 2007), which has $\mathcal{O}(n^3)$ time complexity.

Separable loss. The cubic-time complexity of MTT may be prohibitive in practice for training a model. Moreover, the MTT relies on a computationally unstable matrix inversion and is arguably non-trivial to implement. Hence, there has been interest in using simpler token-separable NLL loss functions (Zhang et al., 2017):

$$\begin{aligned}&\widetilde{\ell}_{(sep-dep)}(\boldsymbol{w}, \boldsymbol{y}) \\ &\quad = -\langle \boldsymbol{w}, \boldsymbol{y} \rangle + \sum_{m \in [n]} \log \sum_{h \in [n] \setminus \{m\}} \exp w_{h,m},\end{aligned}$$

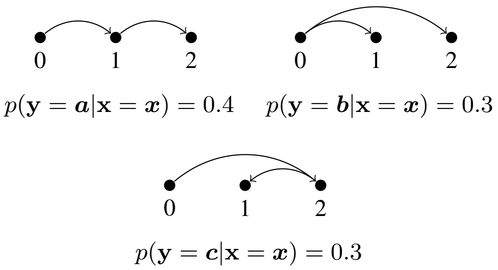
also called head selection loss. This loss is a sum of multiclass classification NLL losses, one per word in the sentence, and is therefore token-separable. As such, it can be efficiently parallelized on GPU and is trivial to implement in any ML framework.

The optimal pointwise surrogate Bayes risk for the token-separable loss is defined as:

$$\begin{aligned}\widetilde{r}^*_{(sep-dep)} = \inf_{\boldsymbol{w} \in \mathbb{R}^A} \; & -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\langle \boldsymbol{w}, \mathbf{y} \rangle] \\ & + \sum_{m \in [n]} \log \sum_{h \in [n] \setminus \{m\}} \exp w_{h,m}.\end{aligned}$$

---

[4] `https://universaldependencies.org/u/overview/syntax.html`


$p(\mathbf{y} = \boldsymbol{a}|\mathbf{x} = \boldsymbol{x}) = 0.4$ $p(\mathbf{y} = \boldsymbol{b}|\mathbf{x} = \boldsymbol{x}) = 0.3$

$p(\mathbf{y} = \boldsymbol{c}|\mathbf{x} = \boldsymbol{x}) = 0.3$

Figure 2: Example of distribution over trees for a sentence of 3 words. The set of trees is defined as $Y = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$.

Let $\widehat{\boldsymbol{w}}$ be an optimal solution. Then, by first order optimality conditions:

$$\frac{\partial}{\partial \widehat{w}_{h,m}} \begin{pmatrix} -\mathbb{E}_{\mathbf{y}|\mathbf{x}=\boldsymbol{x}}[\langle \widehat{\boldsymbol{w}}, \mathbf{y}\rangle] \\ + \sum\limits_{m\in[n]} \log \sum\limits_{h\in[n]\setminus\{m\}} \exp \widehat{w}_{h,m} \end{pmatrix} = 0$$

$$\implies \widehat{w}_{h,m} = \log p\left(\mathrm{y}_{h,m} = 1|\mathbf{x} = \boldsymbol{x}\right) \qquad (4)$$

where $p\left(\mathrm{y}_{h,m} = 1|\mathbf{x} = \boldsymbol{x}\right)$ denotes the marginal distribution of the dependency between words at position h and m, *i.e.* the sum of the conditional probability of trees this dependency appears in.

Theorem 3. *The loss $\widetilde{\ell}_{\textit{(sep-dep)}}$ is not Bayes consistent for distributions over dependency trees.*

Proof. Let $n = 2$ and assume that the distribution $p(\mathbf{y}|\mathbf{x} = \boldsymbol{x})$ is defined as depicted in Figure 2. Then, by Equation 4 we have: $\widehat{w}_{0,1} = \log 0.7$, $\widehat{w}_{0,2} = \log 0.6$, $\widehat{w}_{1,2} = \log 0.4$ and $\widehat{w}_{2,1} = \log 0.3$. As such:

$$\log 0.7 + \log 0.4 = \langle \widehat{\boldsymbol{w}}, \boldsymbol{a}\rangle \\ < \langle \widehat{\boldsymbol{w}}, \boldsymbol{b}\rangle = \log 0.7 + \log 0.6,$$

but $\boldsymbol{a} \in \arg\max_{\boldsymbol{y}\in Y} p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x})$ and $p(\mathbf{y} = \boldsymbol{a}|\mathbf{x} = \boldsymbol{x}) > p(\mathbf{y} = \boldsymbol{b}|\mathbf{x} = \boldsymbol{x})$. Therefore, the token-separable loss is not Bayes consistent, *i.e.* a scoring model minimizing the surrogate risk may not lead to predicting the most probable tree in the data distribution. $\square$

Note that using the (non-separable) NLL loss will lead to a Bayes consistent model on this distribution. Indeed, by Equation 2, optimal scores $\widehat{\boldsymbol{w}}$ satisfy the following condition for all $\boldsymbol{y} \in Y$:

$$\frac{\exp\langle \widehat{\boldsymbol{w}}, \boldsymbol{y}\rangle}{\sum_{\boldsymbol{y}'\in Y} \exp\langle \widehat{\boldsymbol{w}}, \boldsymbol{y}'\rangle} = p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x})$$
$$\implies \langle \widehat{\boldsymbol{w}}, \boldsymbol{y}\rangle = \log p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x})$$

The following assignment for $\widehat{\boldsymbol{w}}$ satisfies this condition: $\widehat{w}_{0,1} = 0$, $\widehat{w}_{0,2} = \log 0.3$, $\widehat{w}_{1,2} = \log 0.4$ and $\widehat{w}_{2,1} = 0$. That is, minimizing the NLL loss on this distribution results in a Bayes consistent classifier, as expected.

The single root constraint case is reported in Appendix A.

6 Conclusion

Studying statistical properties of surrogate loss functions has not been a major interest in the NLP community, although there are exceptions (Ma and Collins, 2018; Effland and Collins, 2021). We proved that token-separable losses for NER and dependency parsing are not Bayes consistent, which means that minimizing these losses will not necessarily lead to models that will predict the most probable output for a given input in the data distribution, even with infinite training data.

In the dependency parsing case, Zhang et al. (2020) experimentally observed that the structured NLL loss leads to better results than the token-separable head selection loss. As such, our analysis provides a better theoretical understanding of these experiments. However, separable losses are widely used in state-of-the-art models, which suggests that future research should study why they work in practice.

Other types of separability have also been used for constituency parsing (Corro, 2020) and semantic parsing (Pasupat et al., 2019), *inter alia*.

Limitations

Arguably, these separable loss functions perform well in practice, which questions the appropriateness of the Bayes consistency property. For example, Long and Servedio (2013) argued that assumptions usually made are too unrealistic (*e.g.* considering that F is the set of all measurable mappings) and leads to misleading theoretical knowledge when it comes to actual implementation and experiments. Maybe this is also the case of the demonstration we made in this paper. However, all in all, we hope that this work will motivate future fundamental research on ML for NLP and especially on properties of loss functions.

Acknowledgments

We thank Joseph Le Roux, François Yvon and the anonymous reviewers for their comments and suggestions.

References

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. 2006. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156.

Mathieu Blondel. 2019. Structured prediction with projection oracles. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Mathieu Blondel, André F.T. Martins, and Vlad Niculae. 2020. Learning with fenchel-young losses. *Journal of Machine Learning Research*, 21(35):1–69.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*.

Caio Corro. 2020. Span-based discontinuous constituency parsing: a family of exact chart-based algorithms with time complexities from O(n^6) down to O(n^3). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2753–2764, Online. Association for Computational Linguistics.

John Duchi, Khashayar Khosravi, and Feng Ruan. 2018. Multiclass classification, information, divergence and surrogate risk. *The Annals of Statistics*, 46(6B):3246 – 3275.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards – B. Mathematics and Mathematical Physics*.

Thomas Effland and Michael Collins. 2021. Partially supervised named entity recognition via the expected entity ratio loss. *Transactions of the Association for Computational Linguistics*, 9:1320–1335.

Matteo Fischetti and Paolo Toth. 1992. An additive bounding procedure for the asymmetric travelling salesman problem. *Mathematical Programming*, 53(1):173–197.

Tilmann Gneiting and Adrian E Raftery. 2007. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150, Prague, Czech Republic. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Yi Lin. 2004. A note on margin-based loss functions in classification. *Statistics & probability letters*, 68(1):73–82.

Yufeng Liu. 2007. Fisher consistency of multicategory support vector machines. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 291–298, San Juan, Puerto Rico. PMLR.

Phil Long and Rocco Servedio. 2013. Consistency versus realizable h-consistency for multiclass classification. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 801–809, Atlanta, Georgia, USA. PMLR.

Gábor Lugosi and Nicolas Vayatis. 2004. On the Bayes-risk consistency of regularized boosting methods. *The Annals of Statistics*, 32(1):30 – 55.

Zhuang Ma and Michael Collins. 2018. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3698–3707, Brussels, Belgium. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Ryan McDonald and Giorgio Satta. 2007. On the complexity of non-projective data-driven dependency parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 121–132, Prague, Czech Republic. Association for Computational Linguistics.

Alex Nowak, Francis Bach, and Alessandro Rudi. 2019. A general theory for structured prediction with smooth convex surrogates. *arXiv preprint arXiv:1902.01958*.

Alex Nowak, Francis Bach, and Alessandro Rudi. 2020. Consistent structured prediction with max-min margin Markov networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7381–7391. PMLR.

Alex Nowak, Alessandro Rudi, and Francis Bach. 2022. On the consistency of max-margin losses. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of

Proceedings of Machine Learning Research, pages 4612–4633. PMLR.

Panupong Pasupat, Sonal Gupta, Karishma Mandyam, Rushin Shah, Mike Lewis, and Luke Zettlemoyer. 2019. Span-based hierarchical semantic parsing for task-oriented dialog. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1520–1526, Hong Kong, China. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Mark D. Reid and Robert C. Williamson. 2010. Composite binary losses. *Journal of Machine Learning Research*, 11(83):2387–2422.

David A. Smith and Noah A. Smith. 2007. Probabilistic models of nonprojective dependency trees. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 132–140, Prague, Czech Republic. Association for Computational Linguistics.

Robert Endre Tarjan. 1977. Finding optimum branchings. *Networks*, 7(1):25–35.

Ambuj Tewari and Peter L. Bartlett. 2007. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8(36):1007–1025.

Michalis Titsias. 2016. One-vs-each approximation to softmax for scalable estimation of probabilities. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Martin J. Wainwright and Michael Irwin Jordan. 2008. *Graphical models, exponential families, and variational inference*. Now Publishers Inc.

Robert C. Williamson, Elodie Vernet, and Mark D. Reid. 2016. Composite multiclass losses. *Journal of Machine Learning Research*, 17(222):1–52.

Tong Zhang. 2004a. Statistical analysis of some multicategory large margin classification methods. *Journal of Machine Learning Research*, 5(Oct):1225–1251.

Tong Zhang. 2004b. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, 32(1):56 – 85.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 665–676, Valencia, Spain. Association for Computational Linguistics.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3295–3305, Online. Association for Computational Linguistics.

A Inconsistency of the separable loss for single root dependency parsing

Theorem 4. *The loss* $\widetilde{\ell}_{\textit{(sep-dep)}}$ *is not Bayes consistent for distributions over single-root dependency trees.*

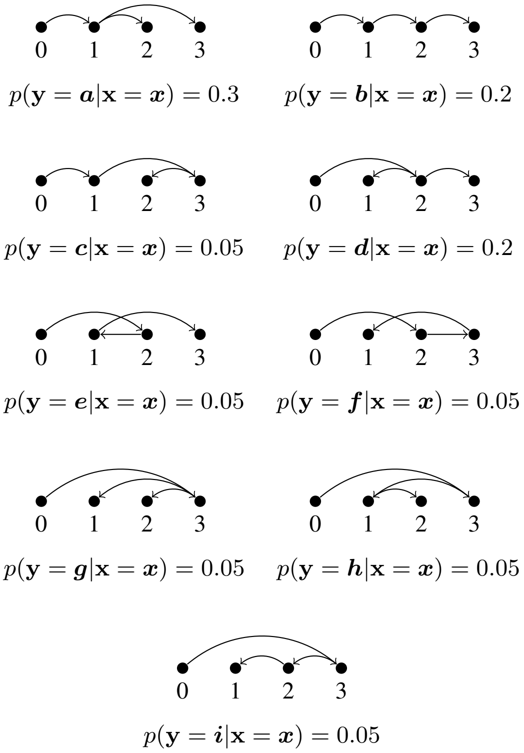
Proof. Let $n = 4$ and assume that the distribution $p(\mathbf{y}|\mathbf{x} = \boldsymbol{x})$ is defined as depicted in Figure 3. Then, by Equation 4 we have, among others:

$$\widehat{w}_{0,1} = \log 0.55, \qquad \widehat{w}_{1,2} = \log 0.55,$$
$$\widehat{w}_{1,3} = \log 0.4, \qquad \widehat{w}_{2,3} = \log 0.45.$$

As such, we have:

$$\langle \widehat{\boldsymbol{w}}, \boldsymbol{a} \rangle = \log 0.55 + \log 0.55 + \log 0.40,$$
$$\langle \widehat{\boldsymbol{w}}, \boldsymbol{b} \rangle = \log 0.55 + \log 0.55 + \log 0.45,$$

which means that $\langle \widehat{\boldsymbol{w}}, \boldsymbol{a} \rangle < \langle \widehat{\boldsymbol{w}}, \boldsymbol{b} \rangle$ but $\boldsymbol{a} \in \arg\max_{\boldsymbol{y} \in Y} p(\mathbf{y} = \boldsymbol{y}|\mathbf{x} = \boldsymbol{x})$ and $p(\mathbf{y} = \boldsymbol{a}|\mathbf{x} = \boldsymbol{x}) > p(\mathbf{y} = \boldsymbol{b}|\mathbf{x} = \boldsymbol{x})$. Therefore the separable loss is not Bayes consistent for single-root dependency parsing. $\square$



$p(\mathbf{y} = \boldsymbol{a}|\mathbf{x} = \boldsymbol{x}) = 0.3$ $p(\mathbf{y} = \boldsymbol{b}|\mathbf{x} = \boldsymbol{x}) = 0.2$

$p(\mathbf{y} = \boldsymbol{c}|\mathbf{x} = \boldsymbol{x}) = 0.05$ $p(\mathbf{y} = \boldsymbol{d}|\mathbf{x} = \boldsymbol{x}) = 0.2$

$p(\mathbf{y} = \boldsymbol{e}|\mathbf{x} = \boldsymbol{x}) = 0.05$ $p(\mathbf{y} = \boldsymbol{f}|\mathbf{x} = \boldsymbol{x}) = 0.05$

$p(\mathbf{y} = \boldsymbol{g}|\mathbf{x} = \boldsymbol{x}) = 0.05$ $p(\mathbf{y} = \boldsymbol{h}|\mathbf{x} = \boldsymbol{x}) = 0.05$

$p(\mathbf{y} = \boldsymbol{i}|\mathbf{x} = \boldsymbol{x}) = 0.05$

Figure 3: Example of distribution over trees for a sentence of 3 words and the single-root constraints. The set of trees is defined as $Y = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h}, \boldsymbol{i}\}$.

A Systematic Search for Compound Semantics in Pretrained BERT Architectures

Filip Miletic and **Sabine Schulte im Walde**
Institute for Natural Language Processing, University of Stuttgart
{filip.miletic, schulte}@ims.uni-stuttgart.de

Abstract

To date, transformer-based models such as BERT have been less successful in predicting compositionality of noun compounds than static word embeddings. This is likely related to a suboptimal use of the encoded information, reflecting an incomplete grasp of how the models represent the meanings of complex linguistic structures. This paper investigates variants of semantic knowledge derived from pretrained BERT when predicting the degrees of compositionality for 280 English noun compounds associated with human compositionality ratings.

Our performance strongly improves on earlier unsupervised implementations of pretrained BERT and highlights beneficial decisions in data preprocessing, embedding computation, and compositionality estimation. The distinct linguistic roles of heads and modifiers are reflected by differences in BERT-derived representations, with empirical properties such as frequency, productivity, and ambiguity affecting model performance. The most relevant representational information is concentrated in the initial layers of the model architecture.

1 Introduction

The meaning of multiword expressions such as noun compounds is notoriously difficult to model, particularly because of variability in their degree of compositionality, i.e. the relatedness of the meaning of a compound (e.g. *flea market*) to that of the individual constituents (*flea* and *market*). The degree of compositionality has been successfully predicted using static word embeddings, but transformer-based models such as BERT (Devlin et al., 2019) have so far been less successful. This might be related to a suboptimal use of the information encoded by the models, reflecting our incomplete grasp of how they represent the meanings of complex linguistic structures.

In this paper, we aim to improve this understanding, as well as produce actionable methodological recommendations. We predict the degrees of compositionality of 280 English noun compounds associated with human compositionality ratings. We extract their occurrences from a web corpus and model them using pretrained BERT. Like previous work, we assume that the contextualized nature of these representations may capture key aspects of compound semantics. But we do not expect this information to be equally accessible across the model or independent from underlying linguistic properties. We therefore experiment with variants of BERT-derived semantic knowledge (comprising over 40,000 ways of computing the degree of compositionality), and analyze the linguistic roles of compound constituents and their empirical properties (frequency, ambiguity, and productivity). We provide the following contributions:

- We identify a robust setup to extract compositionality information from pretrained BERT. It strongly improves on earlier unsupervised implementations and highlights beneficial decisions in data preprocessing, embedding computation, and compositionality estimation.
- We show that the distinct linguistic roles of heads and modifiers are reflected by differences in BERT-derived representations. Further focusing on compound heads, we find clear effects of their empirical properties on model performance.
- Our results support the view that pretrained BERT encodes at least some aspects of the semantics of multiword expressions, and also show that the most relevant information is found in the model's initial layers.

The remainder of this paper is organized as follows. We first review related studies (§2), and then introduce our data (§3) and experimental setup (§4). We then analyze and discuss the results (§5) and provide a conclusion (§6).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1499–1512
May 2-6, 2023

2 Related work

The meaning of noun compounds is modeled from a broad range of perspectives. In psycholinguistics, for example, there is a long tradition of research on human processing of compound semantics. Its focus is usually on semantic transparency, which is operationalized using measures including the semantic relatedness of the constituents and the retention of their meaning in the compound (e.g. Bell and Schäfer, 2016; Auch et al., 2020; Günther et al., 2020). Computational studies have examined a similar range of compound properties, aiming to predict the meaning of the whole compound (Mitchell and Lapata, 2008; Dima et al., 2019), the semantic relations between a compound's constituents (Ó Séaghdha, 2007; Dima et al., 2014), or the compound's degree of compositionality. The latter issue is also the focus of our work.

The ability of computational models of compound semantics to predict the degree of compositionality is usually evaluated on a ranking task with gold standard data in form of human compositionality ratings, which exist for languages including English (Reddy et al., 2011), German (von der Heide and Borgwaldt, 2009; Schulte im Walde et al., 2016), French and Portuguese (Cordeiro et al., 2019). Strong results have been obtained using static word embeddings, generally by learning dedicated representations for the whole compound and comparing them against the representations of the individual constituents, often combined using composition functions (Reddy et al., 2011; Schulte im Walde et al., 2013, 2016; Salehi et al., 2014, 2015; Cordeiro et al., 2019; Alipoor and Schulte im Walde, 2020). Most studies predict the compositionality of the whole compound, but differences between heads and modifiers have been underscored. Reported effects on model performance are related to their distinct linguistic roles and empirical properties such as frequency, productivity, and ambiguity (Schulte im Walde et al., 2013, 2016; Alipoor and Schulte im Walde, 2020).

More recently, compositionality prediction has been addressed using pretrained transformer-based models. While BERT-derived representations used in a subsequently trained classifier performed well on binary classification (Shwartz and Dagan, 2019), their unsupervised use in the standard ranking task formulation has been less successful. For instance, an implementation based on comparisons between contextualized and non-contextualized compound representations obtained significantly poorer results compared to static word embeddings, leading to a suggestion that these models do not capture compositionality in a way similar to human annotators (Garcia et al., 2021a). Stronger correlations with human judgment were obtained in a probing study, but using external linguistic knowledge – gold standard synonyms of noun compounds (Garcia et al., 2021b). We are unaware of unsupervised implementations of BERT-derived representations that are competitive with static word embeddings on this task.

This might be related to a suboptimal use of the information encoded by BERT, as the layers in its architecture capture different aspects of linguistic structure (Rogers et al., 2020). For instance, it has been suggested that semantic knowledge in general (Jawahar et al., 2019) and word sense information in particular (Coenen et al., 2019) is encoded in higher layers, but also that type-level information is encoded in lower layers (Vulić et al., 2020). More generally, these and other types of implementation decisions impact BERT performance on other linguistically oriented tasks (e.g. Laicher et al., 2021). To the best of our knowledge, these patterns have not been investigated in detail for compound semantics, with the cited studies generally relying on widely used solutions (e.g. computing a token-level embedding by averaging over the last four layers).

3 Data

This section introduces the data resources we used. For details on licenses, see Appendix A.

3.1 Gold standard of noun compounds

We use the set of 280 English noun compounds introduced by Cordeiro et al. (2019). It includes an initial set of 90 compounds created by Reddy et al. (2011)[1] and a further 190 compounds annotated by Cordeiro and colleagues using the same rating procedure.[2] Human annotators were asked to provide compositionality ratings in terms of literality, on a scale from 0 (not at all literal) to 5 (very literal). They provided scores for the interpretation of the whole compound (e.g. *crash course*), as well as for the use of the modifier (*crash*) and the head (*course*) within it. Sample compounds and their ratings are shown in Table 1.

---

[1] `http://www.dianamccarthy.co.uk/downloads.html`

[2] `https://pageperso.lis-lab.fr/carlos.ramisch/?page=downloads/compounds`

	Compositionality rating		
Compound	Modifier	Head	Phrase
guinea pig	0.47 ± 0.72	0.47 ± 0.72	0.24 ± 0.56
flea market	0.38 ± 0.81	4.71 ± 0.84	1.52 ± 1.13
biological clock	4.71 ± 0.47	1.76 ± 1.35	2.29 ± 1.21
health insurance	4.53 ± 0.88	4.83 ± 0.58	4.40 ± 1.17

Table 1: Sample gold standard compounds with compositionality ratings (mean and standard deviation).

3.2 Corpus

As corpus data for the modeled noun compounds, we rely on the widely used ENCOW corpus, obtained by crawling web data and containing ≈ 9.6 billion words (Schäfer and Bildhauer, 2012; Schäfer, 2015). For each compound, all tokenized sentences in which it appears are extracted. We only use singular forms so as to avoid potential variability related to grammatical number in BERT.

3.3 Empirical compound properties

Parts of our analysis use information on empirical properties of compound constituents (in particular, their heads), and specifically (i) frequency; (ii) productivity, i.e. the number of compound-types in which they appear; and (iii) ambiguity, i.e. their number of senses. We use the information on these properties created by Schulte im Walde et al. (2016) for the Reddy et al. (2011) dataset. They derived frequency and productivity information from the ENCOW corpus, and calculated ambiguity based on WordNet (Fellbaum, 1998). We apply the same procedures to calculate the information for the compounds from the Cordeiro et al. (2019) dataset.

4 Experimental setup

4.1 BERT representations

We use `BERT-base-uncased`, a 768-dimension, 12-layer version of the model (110 million parameters) pretrained on English data, from the Hugging Face implementation (Wolf et al., 2020). We run the experiments on a CPU computing server (2×12 3GHz cores with 768GB RAM) over ≈ 5 days. In order to facilitate the analysis of modeling properties, we deliberately adopt a straightforward setup without fine-tuning. Each sequence from the corpus is fed into the model, which returns multiple vector representations for each token in the sequence. For all sequences of a compound, we retain the representations for each token in the sequence; these correspond to the input embedding layer and the outputs of the 12 hidden states. We test different ways of combining the obtained information.

Embedding types. BERT produces contextualized representations for each token in the sequence, which we use both individually and by combining multiple token representations (see pooling functions below). We compute the following types of embeddings. (i) `modif`: representation of the modifier, corresponding to its contextualized embedding. (ii) `head`: representation of the head, corresponding to its contextualized embedding. (iii) `comp`: representation of the full compound, corresponding to pooled `modif` and `head` embeddings. (iv) `cont`: representation of the context in which the compound appears, corresponding to the merged embeddings of all tokens in the sequence apart from `modif`, `head`, `[CLS]`, and `[SEP]`. (v) `cls`: embedding of the `[CLS]` token, taken to correspond to a representation of the full sequence.

BERT's tokenizer splits out-of-vocabulary tokens into subwords with known representations. When this occurs for `modif` or `head`, the subword representations are pooled into a single embedding.

Layer combinations. We test all contiguous spans of layers, across the input embedding and the 12 hidden state outputs, for a total of 91 layer combinations. The smallest combination is a single layer, and the largest is the full range of 13 layers.

Pooling functions. Multiple vectors can be combined in different ways; we test two options, averaging (`avg`) or summing (`sum`) over them. This is applied (i) token-wise, if merging multiple token representations; (ii) layer-wise, if merging multiple layers; (iii) sequence-wise, if creating a type-level representation (see below). In order to streamline the experimental setup, the same pooling function is used in all three cases.

Sequence length. We test if using longer sequences can be beneficial based on the assumption that a larger context may be more semantically discriminating. We either retain only sequences with at least 20 space-separated tokens, or do not impose any threshold; this technically corresponds to a minimum of 3 tokens, i.e. the lowest sequence length in the corpus.

Number of sequences. From a similar perspective, we examine the effect of increasing the number of modeled sequences, experimenting with 10, 100, and 1,000 sequences per compound. The sets of sequences are not resampled; rather, the smaller sets are included in the larger ones. This criterion is combined with that of sequence length: for instance, we extract 1,000 sequences of any length as well as 1,000 sequences with at least 20 tokens, although these may partly overlap. For compounds whose corpus frequency is lower than the threshold, all available occurrences are used.

4.2 Compositionality estimates

As stated above, we expect that the contextualized representations we use may carry semantic information reflecting the degree of compositionality of a compound, but it is unclear which specific combination of representations is the most efficient.

Direct estimates. We directly compute pairwise cosine scores between pairs of embeddings, testing all 10 pairs of embedding types (`head`–`modif`, `head`–`comp`, `head`–`cont`, and so forth).

Composite estimates. We further combine the directly measured information pertaining to the head and the modifier of a compound using the same composition functions as Reddy et al. (2011). Specifically, we use `head` and `modif` embeddings in combination with one of the following: `comp`, `cont`, and `cls`. Taking `comp` as an example, we compute the composite estimates as follows:

$$\begin{aligned}\text{ADD} &= \cos(\texttt{comp}, \texttt{modif}) + \cos(\texttt{comp}, \texttt{head})\\ \text{MULT} &= \cos(\texttt{comp}, \texttt{modif}) \cdot \cos(\texttt{comp}, \texttt{head})\\ \text{COMB} &= \text{ADD} + \text{MULT}\end{aligned}$$

Token-level vs. type-level. The information from individual occurrences can be aggregated into a single numerical score in different ways. (i) In the token-level approach, we compute a compositionality estimate for each individual occurrence, and then average over those values to get a single score. (ii) In the type-level approach, we first compute a type-level representation by applying a pooling function; as an example, we average or sum over all individual `head` embeddings to produce a type-level `head` embedding. This type-level representation is then used to directly compute a single compositionality estimate.

The combinations of the presented experimental parameters correspond to a total of 41,496 ways of computing a numerical estimate of the degree of compositionality. In a trial run, we also experimented with other parameters (e.g. restrictions on the position of the compound within the sequence), but they did not exhibit strong effects and for clarity are not included in the present discussion.

5 Results and discussion

We evaluate each constellation of parameters by calculating Spearman's rank correlation coefficient between the compositionality estimates it produces and human judgments from the test set. All implementations are evaluated on three prediction targets: compositionality scores for the compound as a whole, the head, and the modifier.

5.1 Overview of model performance

We begin by looking at general trends in model performance (Table 2). The highest correlation coefficient we obtain stands at 0.706 for compound-level compositionality. Compared to previous studies on the same dataset, it is in a similar range as the result reported by Cordeiro et al. (2019) using static word embeddings ($\rho = 0.726$).[3] Their best performance was obtained by training a `word2vec` model on a corpus in which compound occurrences had been joined into single tokens. The cosine scores between those representations and compositionally constructed vectors were then used to predict the degree of compositionality. Interestingly, this procedure strongly relies on the context in which the compounds occur, which is also a key component of our best approaches (see below).

The highest performance we obtain substantially improves on the best BERT-derived score reported by Garcia et al. (2021a) based on comparisons between contextualized and non-contextualized representations of a compound ($\rho = 0.37$). In another BERT-based experiment, Garcia et al. (2021b) obtain a higher correlation using similarity measurements between compounds and gold standard synonyms ($\rho = 0.67$); by contrast, we do not rely on external linguistic knowledge.

The implementations we tested are strongest at predicting compound-level compositionality. This is followed by the compositionality of the head (maximum $\rho = 0.645$) and then of the modifier (maximum $\rho = 0.553$). Predicted values for the three types of scores are strongly correlated ($\rho > 0.9$ for all three pairwise comparisons), but the best performing parameter constellations clearly differ.

More generally, the summary in the table underscores the wide range of obtained values; the

---

[3] This result is reported for a subset of the same dataset containing 180 compounds.

	ρ	layers	pool	len	seqs	estimate		agg	ρ	layers	pool	len	seqs	estimate		agg
COMP	0.706	1-1	sum	3	1k	COMB	cont	token	-0.642	2-3	avg	3	1k	cont	cls	type
	0.706	1-1	avg	3	1k	COMB	cont	token	-0.644	3-3	avg	3	1k	cont	cls	type
	0.706	1-1	sum	20	1k	MULT	cont	token	-0.645	1-5	avg	3	1k	cont	cls	type
	0.706	1-1	avg	20	1k	MULT	cont	token	-0.646	2-4	avg	3	1k	cont	cls	type
	0.706	1-1	sum	3	1k	MULT	cont	token	-0.649	1-4	avg	3	1k	cont	cls	type
HEAD	0.645	1-1	sum	3	1k	head	cont	token	-0.598	0-7	avg	3	1k	cont	cls	type
	0.645	1-1	avg	3	1k	head	cont	token	-0.599	1-4	avg	3	1k	cont	cls	type
	0.638	1-1	sum	3	1k	COMB	cont	token	-0.600	0-6	avg	3	1k	cont	cls	type
	0.638	1-1	avg	3	1k	COMB	cont	token	-0.604	1-5	avg	3	1k	cont	cls	type
	0.638	1-1	sum	3	1k	ADD	cont	token	-0.606	1-6	avg	3	1k	cont	cls	type
MODIF	0.553	1-1	avg	20	1k	modif	cont	token	-0.464	2-4	avg	3	1k	cont	cls	type
	0.553	1-1	sum	20	1k	modif	cont	token	-0.465	1-5	avg	3	1k	cont	cls	type
	0.548	1-1	sum	3	1k	modif	cont	token	-0.471	1-3	avg	3	1k	cont	cls	type
	0.548	1-1	avg	3	1k	modif	cont	token	-0.474	1-4	avg	3	1k	cont	cls	type
	0.546	1-1	avg	20	1k	modif	cont	type	-0.476	1-2	avg	3	1k	cont	cls	type

Table 2: Best (left) and worst (right) evaluated implementations. Abbreviations: *pool* = pooling function; *len* = minimum tokens per sequence; *seqs* = minimum number of modeled sequences; *agg* = aggregation of occurrences (type vs. token-level).

	min. 3 tokens	min. 20 tokens
COMP	0.146 (-0.649, 0.706)	0.134 (-0.587, 0.706)
HEAD	0.102 (-0.606, 0.645)	0.087 (-0.561, 0.637)
MODIF	0.099 (-0.476, 0.548)	0.093 (-0.460, 0.553)

Table 3: Spearman's ρ (mean, min, max) for minimum sequence length.

	10 sequences	100 sequences	1,000 sequences
C	0.135 (-0.394, 0.622)	0.142 (-0.607, 0.689)	0.143 (-0.649, 0.706)
H	0.093 (-0.384, 0.565)	0.094 (-0.551, 0.621)	0.096 (-0.606, 0.645)
M	0.089 (-0.367, 0.495)	0.101 (-0.459, 0.544)	0.098 (-0.476, 0.553)

Table 4: Spearman's ρ (mean, min, max) for number of sequences. C, H, M = compound, head, modifier.

weakest implementations yield negative correlations, with a low of -0.649. This confirms the relevance of the parameters we tested and the importance of understanding the optimal choices in implementing them. Trends suggested by the initial overview include better performance of (i) the first hidden layer in isolation; (ii) token-level rather than type-level modeling; (iii) `cont` embeddings, when paired with another relevant type of information; (iv) embeddings targeting the compound as a whole, the head, or the modifier for the corresponding compositionality score. Strikingly, all the weakest constellations are closely similar to one another, the only distinguishing characteristic being the span of layers. But some of these parameter choices are also found in the best performing implementations; more generally, the direction and relevance of all trends are not immediately apparent. We therefore now more closely examine individual parameters.

Sequence length. Using sequences with at least 20 tokens leads to lower mean correlations. However, their minimum values are higher and maximum values are comparable (Table 3). The overall lack of a clear effect suggests that beneficial distinctions made by the model are primarily based on the most immediate linguistic context.

Number of modeled sequences. There is a clear trend towards an increase in performance with an increase in the number of modeled sequences (Table 4), likely because this facilitates disambiguation and limits the effect of sampling differences. Looking at the maximum values, the shift from 10 to 100 occurrences leads to a stronger improvement (≈ 0.06 points) than the shift from 100 to 1,000 (≈ 0.02 points); this suggests that it is especially important to avoid very low numbers of examples. While increasing the number of occurrences also leads to the lowest performances overall, this may be due to the detrimental effect of other parameters.

We further assessed the impact of sampling differences in the condition with 10 sequences, which is the most inherently unstable. We ran the compositionality estimation 10 times, each time randomly resampling the modeled occurrences. The mean difference between the minimum and maximum values obtained by a parameter constellation is 0.028. This variability does not alter the overall parameter-level trends (see Appendix B).

Some compounds do not exceed the threshold frequencies for some parameter combinations. This affects 10 compounds for the 100 sequence threshold, and 97 compounds for the 1,000 sequence

	avg	sum
Comp	0.139 (-0.649, 0.706)	0.141 (-0.587, 0.706)
Head	0.094 (-0.606, 0.645)	0.095 (-0.563, 0.645)
Modif	0.095 (-0.476, 0.553)	0.097 (-0.460, 0.553)

Table 5: Spearman's ρ (mean, min, max) for pooling functions.

	token-level	type-level
Comp	0.150 (-0.584, 0.706)	0.130 (-0.649, 0.699)
Head	0.103 (-0.556, 0.645)	0.085 (-0.606, 0.628)
Modif	0.100 (-0.460, 0.553)	0.092 (-0.476, 0.546)

Table 6: Spearman's ρ (mean, min, max) for token- vs. type-level processing.

threshold. As a check, we calculated correlations on a smaller set of compounds, excluding the 10 most affected by frequency issues. The results were near-identical, with marginal improvements for the top predictions (e.g. best ρ increasing from 0.706 to 0.710 for compound scores) and the same general trend. As for the compounds with fewer than 1,000 sequences, the mean number of sequences available for these items was 510, which is still a strong increase compared to the preceding threshold.

Pooling functions. Averaging and summing perform similarly when creating a single embedding from multiple vectors (Table 5). They obtain near-identical mean and identical maximum values; averaging leads to lower minimum values.

One of the ways we used pooling functions was to merge representations for out-of-vocabulary tokens that are split up by BERT's tokenizer. This affected 14 compounds; some instances reflect derivational patterns (e.g. `mail`, `##ing` in *mailing list*), but others are more obscure (e.g. `gr`, `##av`, `##y` in *gravy train*). Since it is unclear what some of these representations capture, we checked their impact by calculating correlations on a reduced set of compounds, excluding those with OOV tokens. The results were marginally higher (e.g. ρ increasing from 0.706 to 0.710 for compound scores). This issue therefore does not appear to have a strong detrimental effect, at least when it is limited to a small subset (5%) of target items.

Another frequently used pooling function is concatenation. Applying it across multiple tokens (e.g. for out-of-vocabulary items) or multiple sequences (to create a type-level representation) would result in comparisons between vectors with different numbers of dimensions; we therefore did not include it in the full experimental setup. As a check, however, we ran concatenation across layers, combined with averaging over tokens and sequences. It led to slightly higher mean correlations (0.147 for compound scores), but it did not improve on the maximum results so we did not experiment further.

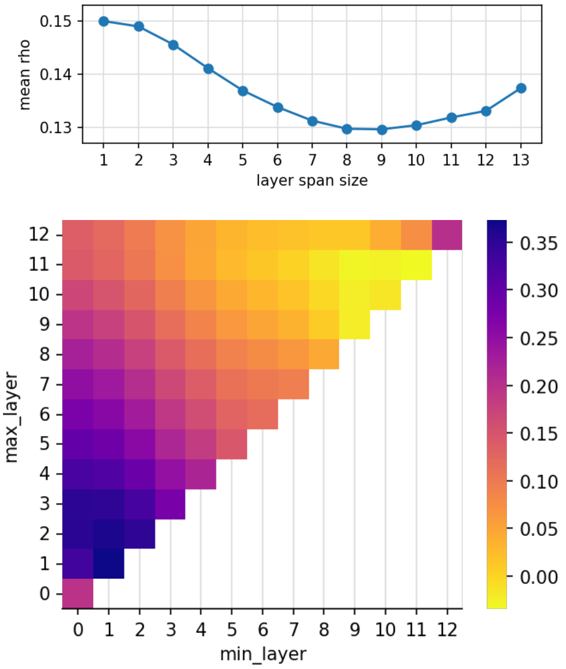


Figure 1: Top: effect of the number of modeled layers on compound-level compositionality prediction. Bottom: mean correlations for compound compositionality prediction across layer combinations. The `min_layer` and `max_layer` values are start and end points of a contiguous span of layers.

Token vs. type-level. There is a clear preference across the board for token-level processing (Table 6). Most improvements over type-level are $\approx$ 0.01–0.02 points; they reach $\approx$ 0.06 points when comparing the worst-performing configurations. A potential explanation is that estimating compositionality on individual occurrences – rather than a merged type-level representation – may be less sensitive to ambiguous or otherwise noisy data.

Layers. The results are affected by the number of modeled layers (Figure 1). The best performance is obtained with a single layer (mean $\rho = 0.150$ for compound scores) and decreases as the span increases up to 9 layers (0.130). Larger spans fare somewhat better, likely because they are bound to capture some relevant representational information; the best is the full range of 13 layers (0.137).

In terms of specific layers, the best result overall is obtained by the first hidden layer in isolation (mean $\rho = 0.373$), followed by other combinations and individual layers in the low-to-mid range. This is plotted in Figure 1 for compound compositionality scores; head and modifier scores follow the same trends (see Appendix C). The traditionally used combination of the last four hidden layers (9–12) is not among the best ones. It is in fact the single weakest in terms of maximum values: 0.248 for compound scores, close to 0.5 below the best implementation. However, it also has a comparatively high minimum value (-0.262), suggesting it is more robust to the effect of other parameters.

Compositionality estimates. We summarize the impact of compositionality estimates by looking at performance across the five types of embeddings which constitute the basis of subsequent score calculations (Table 7); for a summary of results on individual estimates, see Appendix D.

	modif	head	comp	cont	cls
COMP	0.135	0.274	0.245	0.172	-0.128
	-0.383	-0.133	-0.324	-0.649	-0.649
	0.615	0.630	0.666	0.706	0.611
HEAD	0.071	0.242	0.194	0.130	-0.161
	-0.384	-0.130	-0.327	-0.606	-0.606
	0.464	0.645	0.598	0.645	0.558
MODIF	0.106	0.167	0.164	0.133	-0.094
	-0.274	-0.130	-0.229	-0.476	-0.476
	0.553	0.415	0.517	0.553	0.477

Table 7: Spearman's ρ (mean, min, max) for embedding types, across all direct and composite estimates if used.

Looking at the mean values, `head` performs stronger than `modif` as well as `comp` across the prediction targets. This is coherent with the dominant role of the head in the morphological constituency of compounds. However, the maximum values out of the three are obtained by `comp` for the compound-level score, `head` for the head score, and `modif` for the modifier score. This indicates that representations targeting the whole compound or a constituent of interest are successful in capturing information specific to the respective element.

As for the two other embedding types, the mean values for `cont` follow `head` and `comp` across prediction targets. But its maximum values are the single best (compound-level prediction) or joint best (head and modifier predictions). By contrast, `cls` is clearly in the lower range of performance. Its maximum values remain around 0.1 points behind the best implementations. This shows that using the linguistic context surrounding the compound – modeled by `cont` – is beneficial, and clearly more so than using a representation of the full sequence.[4]

This might be explained by the redundancy of using a representation of a compound or its constituent, and comparing it with a representation of the full sequence which encodes the same element. It could also be the case that, as suggested before, token-level embeddings are strongly influenced by their immediate linguistic context; a balanced representation of a broader range of information might be complementary. Whatever the case, the results show that the similarity between a compound and its linguistic context is reflective of the degree of compositionality. This might explain why previous implementations using neutral contexts were less successful in capturing these trends.

5.2 Ablation study

In order to further validate the parameter-level trends observed thus far, we conduct an ablation study. We start from the parameter constellation that obtained the best results on predicting the compositionality of the compound as a whole (see Table 2; the top two configurations obtained identical results). For each parameter, we then test all other potential values, one at a time, while keeping the remaining parameters unchanged. The results are presented in Figure 2.

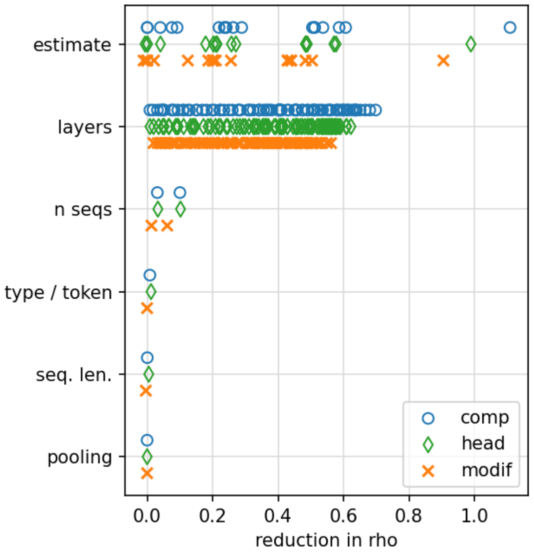


Figure 2: Effect of alternative parameter values compared to the top parameter constellation.

---

[4] A more direct comparison would involve a merged representation of all the tokens in the sequence rather than the `cls` embedding. We nevertheless think that we capture the same range of information.

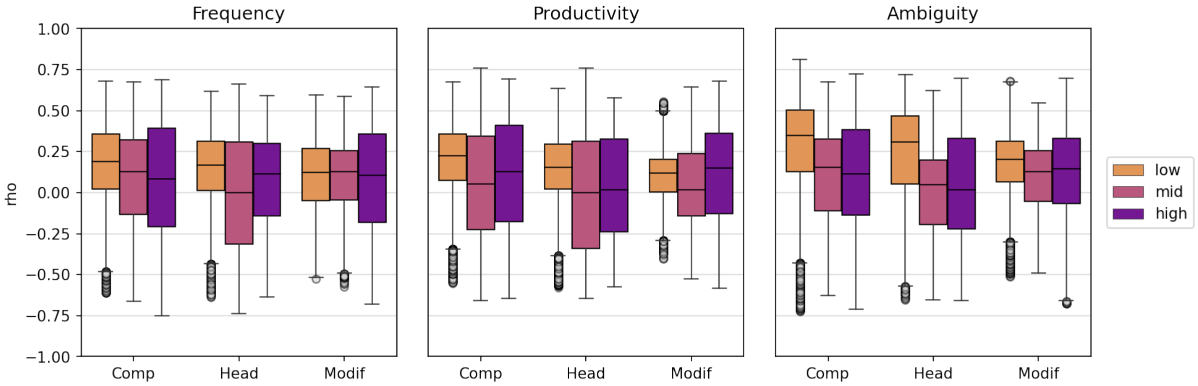


Figure 3: Effect of empirical properties of the head on model performance, observed across the evaluated implementations. Values on the x-axis indicate prediction targets (compound, head, and modifier scores).

Several parameters exhibit no or limited differences with respect to the top parameter constellation: pooling functions, minimum sequence length, and type vs. token-level modeling. Clearer effects on performance are observed for the number of modeled sequences; the drop is strongest for the smallest number of sequences. The choice of layers shows a very strong effect, with a clear tendency towards a drop in performance with both (i) a shift from initial to later hidden states, and (ii) an increase in layer span size. The strongest effect is exhibited by the estimate used to predict the degree of compositionality. All weakest estimates (reduction in $\rho > 0.4$) involve the `cls` embedding. All strongest estimates (reduction in $\rho < 0.1$) involve the `cont` embedding.

These results overall confirm the trends previously discussed for individual parameters. They also provide further evidence that model performance is most strongly affected by the choice of representational information to be used, i.e. the layers and the modeled tokens in the sequence. Suboptimal values for either of these parameters can lead to compositionality predictions that are fully decorrelated from human judgment. Performance does not depend to the same extent on the way in which data is preprocessed and representational information is combined. However, the effect of some of these parameters is not negligible, especially cumulatively, and as such should not be disregarded.

5.3 Empirical properties of compounds

We now turn to the potential impact of empirical properties of the compounds on model performance. We focus on key characteristics of the compounds' heads, given their importance indicated by the trends for compositionality estimates.

As previously stated, we examine the impact of three empirical properties: frequency, productivity, and ambiguity (for sources of this information, see Section 3.3). For each property, we rank the 280 compounds based on the values they exhibit and split them into five sets containing 56 compounds each. We retain the first, third, and fifth set, which we take to clearly reflect the low, mid, and high ranges for each empirical property. The remaining two sets are excluded in order to avoid overlapping or closely similar values in adjoining sets.

A summary of the splits across the three features is presented in Table 8. For each feature, we compute correlations with human judgments separately for each of the three splits of compounds, across all evaluated constellations of parameters. The distribution of the obtained values is plotted in Figure 3 and further discussed below.

Feature	Mean	Std.	Example
Frequency	42	± 30	*silver **spoon***
(thousands)	452	± 108	*labor **union***
	3,614	± 2,438	*crash **course***
Productivity	7	± 5	*night **owl***
	75	± 19	*time **difference***
	448	± 208	*birth **rate***
Ambiguity	2	± 1	*research **project***
	5	± 1	*flea **market***
	13	± 4	*application **form***

Table 8: Mean and standard deviation for the low, mid, and high range splits across empirical features. A sample compound from each split is provided.

Frequency. In predicting the compound and head compositionality scores, the best performance is obtained for low-frequency heads (mean $\rho = 0.18$

and 0.16, respectively). For the compound score, performance across the mid and high range is similar; for the head score, it is clearly lower in the mid range. As for the modifier score, performance is overall stable (mean $\rho = 0.10$ across the three splits); this is not especially surprising given the focus on the frequency of the head. This contrasts the previously reported improvements in performance with a higher number of modeled sequences. However, head frequency is correlated with both productivity and ambiguity ($\rho = 0.87$ and 0.50, respectively, across the 280 compounds). Poorer performance in higher frequency ranges is consistent with an indirect effect of these two properties.

Productivity. Overall, low-productivity heads clearly obtain the best results for compound and head compositionality scores (mean $\rho = 0.21$ and 0.16, respectively). Low- and high-productivity heads obtain similar results for the modifier scores (mean $\rho = 0.11$ and 0.12, respectively). Across the prediction targets, mid-productivity heads have the poorest performance; this drop is the strongest for head scores (mean $\rho = 0.01$). The overall better performance at modeling low-productivity heads is likely explained by their very nature of being used with fewer distinct modifiers, which might facilitate the learning of those compound meanings. This is opposed to higher productivity ranges, which potentially imply more dispersion.

Ambiguity. Across the prediction targets, low-ambiguity heads clearly have the strongest performance (mean $\rho = 0.30$ for compound scores; 0.25 for head scores; 0.18 for modifier scores). The difference with respect to the two other ambiguity ranges is the strongest for head scores; it amounts to ≈ 0.20 points. Performance is similar for mid- and high-range ambiguity. For compound and head scores, the mid range performs slightly better than the high range; for modifier scores, it is the reverse. Similarly to low productivity, the better results for low ambiguity suggest that limited semantic dispersion across the occurrences of a given word makes it easier for BERT to learn its meaning. This is in turn beneficial for derived representations of more complex linguistic structures such as compounds.

Overall, these results have shown that compound properties affect predictions of the degree of compositionality, with better performance in the lower ranges of all three properties. This directly echoes the results for type-level word embeddings reported by Schulte im Walde et al. (2016), who similarly suggested that prediction performance is related to broader effects of compound properties on the quality of the underlying meaning representations. Moreover, the prediction of compound and head scores generally follows the same pattern across the three empirical features. Although modifier scores do not align as closely, they too point to an effect of head properties. These results further underscore the importance of the representations of compound heads for the modeling approach we adopted.

6 Conclusion

We have presented an experiment on predicting the degree of compositionality of 280 English noun compounds using a wide range of variants of semantic knowledge derived from pretrained BERT. We have identified a competitive best implementation achieving $\rho = 0.706$ with human judgement and highlighted clear takeaways.

In terms of preprocessing, stronger results were obtained when modeling a larger number of occurrences per compound, whereas controlling for sequence length did not have a clear effect. On embedding computation, different pooling functions led to comparable performance, but there was a clear advantage for layers in the low-to-mid range, which strongly improved on layer combinations used in earlier studies. As for compositionality estimates, it was clearly beneficial to compute them on the token rather than type level, as well as to use (i) representations targeting the constituent of interest; (ii) representations of the surrounding context; (iii) comparisons across complementary – rather than redundant – representational information.

Looking at empirical properties of compounds, low-frequency, low-productivity, and low-ambiguity heads obtain better compositionality predictions. This trend confirms that more limited semantic dispersion makes it easier to model compound meaning. The fact that it holds across compound, head, and modifier compositionality scores highlights the importance of the head in the linguistic structure – and computational modeling – of compounds. Taken together, our results point to important practical decisions when running similar implementations and contribute to our understanding of the way in which BERT represents lexical meaning, supporting the view that the pretrained model encodes at least some aspects of compound semantics.

Limitations

Our experiments were limited to noun compounds in a single pretrained model for English, with potential implications for the generalizability of our results. They may be partly related to the architecture of this specific model. From a linguistic standpoint, compound properties vary widely across languages. For instance, where English has productive patterns combining two nouns, often in an open (space-separated) compound, German has closed compounds; Romance languages widely rely on N-Prep-N patterns; the structure in many Slavic languages involves patterns of nominal declension; and so forth. The model might not capture the information relevant for compositionality prediction in the same way across these cases. Additionally, our results are strongly related to the central role of compound heads; they may therefore be different in multiword expressions with a different linguistic structure, such as particle verbs and idioms. Finally, we used token-level representations to predict type-level compositionality judgments. This is relevant in terms of assessing the general ability to infer type-level information, but (i) performance may be improved by controlling for the senses in the modeled occurrences vs. those in the stimuli used to collect human judgements; (ii) further work is needed to fully understand the factors contributing to individual token-level representations of a compound.

Acknowledgements

The research was supported by the DFG Research Grant SCHU 2580/5-1 (*Computational Models of the Emergence and Diachronic Change of Multi-Word Expression Meanings*).

References

Pegah Alipoor and Sabine Schulte im Walde. 2020. Variants of vector space reductions for predicting the compositionality of English noun compounds. In *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pages 4379–4387, Marseille, France. European Language Resources Association.

Leah Auch, Christina L. Gagné, and Thomas L. Spalding. 2020. Conceptualizing semantic transparency: A systematic analysis of semantic transparency measures in English compound words. *Methods in Psychology*, 3:100030.

Melanie J. Bell and Martin Schäfer. 2016. Modelling semantic transparency. *Morphology*, 26(2):157–199.

Andy Coenen, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda Viégas, and Martin Wattenberg. 2019. Visualizing and measuring the geometry of BERT. *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

Silvio Cordeiro, Aline Villavicencio, Marco Idiart, and Carlos Ramisch. 2019. Unsupervised compositionality prediction of nominal compounds. *Computational Linguistics*, 45(1):1–57.

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

Corina Dima, Daniël de Kok, Neele Witte, and Erhard Hinrichs. 2019. No word is an island—A transformation weighting model for semantic composition. *Transactions of the Association for Computational Linguistics*, 7:437–451.

Corina Dima, Verena Henrich, Erhard Hinrichs, and Christina Hoppermann. 2014. How to tell a schneemann from a milchmann: An annotation scheme for compound-internal relations. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1194–1201, Reykjavik, Iceland. European Language Resources Association (ELRA).

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021a. Assessing the representations of idiomaticity in vector models with a noun compound dataset labeled at type and token levels. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2730–2741, Online. Association for Computational Linguistics.

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021b. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics.

Fritz Günther, Marco Marelli, and Jens Bölte. 2020. Semantic transparency effects in German compounds: A large dataset and multiple-task investigation. *Behavior Research Methods*, 52(3):1208–1224.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Severin Laicher, Sinan Kurtyigit, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. 2021. Explaining and improving BERT performance on lexical semantic change detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 192–202, Online. Association for Computational Linguistics.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio. Association for Computational Linguistics.

Diarmuid Ó Séaghdha. 2007. Annotating and learning compound noun semantics. In *Proceedings of the ACL 2007 Student Research Workshop*, pages 73–78, Prague, Czech Republic. Association for Computational Linguistics.

Siva Reddy, Diana McCarthy, and Suresh Manandhar. 2011. An empirical study on compositionality in compound nouns. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 210–218, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2014. Using distributional similarity of multi-way translations to predict multiword expression compositionality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 472–481, Gothenburg, Sweden. Association for Computational Linguistics.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2015. A word embedding approach to predicting the compositionality of multiword expressions. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 977–983, Denver, Colorado. Association for Computational Linguistics.

Roland Schäfer. 2015. Processing and querying large web corpora with the COW14 architecture. In *Proceedings of Challenges in the Management of Large Corpora 3 (CMLC-3)*, Lancaster.

Roland Schäfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*, pages 486–493, Istanbul, Turkey. European Language Resources Association.

Sabine Schulte im Walde, Anna Hätty, and Stefan Bott. 2016. The role of modifier and head properties in predicting the compositionality of English and German noun-noun compounds: A vector-space perspective. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 148–158, Berlin, Germany. Association for Computational Linguistics.

Sabine Schulte im Walde, Anna Hätty, Stefan Bott, and Nana Khvtisavrishvili. 2016. GhoSt-NN: A representative gold standard of German noun-noun compounds. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2285–2292, Portorož, Slovenia. European Language Resources Association (ELRA).

Sabine Schulte im Walde, Stefan Müller, and Stefan Roller. 2013. Exploring vector space models to predict the compositionality of German noun-noun compounds. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 255–265, Atlanta, Georgia, USA. Association for Computational Linguistics.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Claudia von der Heide and Susanne Borgwaldt. 2009. Assoziationen zu Unter-, Basis- und Oberbegriffen. Eine explorative Studie. In *Proceedings of the 9th Norddeutsches Linguistisches Kolloquium*, pages 51–74.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Licenses for data resources

The gold standard datasets of noun compounds and extensions with empirical properties are publicly distributed by their authors without a specific license. The ENCOW corpus and WordNet were acquired prior to this study. Their licenses do not have restrictions regarding research use.

B Effect of sampling differences

We found that resampling corpus occurrences (for $n = 10$ occurrences per compound) led to minor differences in model performance. However, they did not impact parameter-level trends overall, with the effect of parameter choices remaining the same in the majority of cases. To illustrate this, we are reporting the mean correlations with human judgment based on different samples of occurrences: three individual samples (those whose overall mean correlations are the lowest, the closest to the mean, and the highest), as well as the mean and standard deviation for the 10 samples. Similarly to the main analysis, the mean values are presented for different parameter choices; for layers, this is limited to a sample of layer spans. The results are split across the three prediction targets: for the compound as a whole in Table 9; for the head in Table 10; for the modifier in Table 11.

	Sample shuffles			Mean	Std
Seq. len.					
3	0.136	0.148	0.157	0.146	0.006
20	0.117	0.135	0.151	0.138	0.011
Agg.					
token	0.134	0.146	0.161	0.149	0.008
type	0.120	0.137	0.147	0.134	0.008
Pooling					
avg	0.126	0.141	0.153	0.141	0.008
sum	0.127	0.142	0.155	0.143	0.008
Estimate					
head	0.265	0.276	0.280	0.274	0.006
modif	0.132	0.134	0.145	0.136	0.006
comp	0.240	0.243	0.250	0.245	0.005
cont	0.156	0.172	0.194	0.180	0.013
cls	-0.153	-0.119	-0.105	-0.126	0.016
Layers					
0-0	0.196	0.198	0.195	0.200	0.006
1-1	0.118	0.139	0.153	0.136	0.010
11-11	-0.062	-0.036	-0.009	-0.040	0.016
12-12	0.179	0.209	0.226	0.200	0.014
1-4	0.310	0.321	0.317	0.318	0.006
8-12	-0.009	0.011	0.044	0.011	0.015

Table 9: Mean correlations for compositionality prediction (of the compound as a whole) across parameter choices.

	Sample shuffles			Mean	Std
Seq. len.					
3	0.095	0.098	0.120	0.104	0.009
20	0.083	0.086	0.106	0.095	0.010
Agg.					
token	0.094	0.097	0.123	0.106	0.010
type	0.083	0.087	0.103	0.093	0.008
Pooling					
avg	0.089	0.091	0.113	0.099	0.009
sum	0.089	0.092	0.113	0.100	0.009
Estimate					
head	0.238	0.240	0.249	0.241	0.006
modif	0.075	0.064	0.086	0.075	0.007
comp	0.196	0.187	0.202	0.195	0.005
cont	0.113	0.126	0.163	0.143	0.019
cls	-0.165	-0.148	-0.132	-0.152	0.013
Layers					
0-0	0.145	0.151	0.150	0.151	0.007
1-1	0.090	0.095	0.120	0.101	0.010
11-11	-0.055	-0.052	-0.013	-0.047	0.015
12-12	0.185	0.184	0.228	0.192	0.017
1-4	0.252	0.261	0.263	0.262	0.009
8-12	-0.015	-0.018	0.028	-0.006	0.015

Table 10: Mean correlations for compositionality prediction (of the head) across parameter choices.

	Sample shuffles			Mean	Std
Seq. len.					
3	0.097	0.103	0.091	0.095	0.006
20	0.079	0.095	0.098	0.091	0.009
Agg.					
token	0.091	0.099	0.098	0.096	0.005
type	0.085	0.099	0.092	0.090	0.007
Pooling					
avg	0.087	0.098	0.094	0.092	0.006
sum	0.088	0.099	0.096	0.094	0.006
Estimate					
head	0.160	0.170	0.167	0.166	0.004
modif	0.101	0.106	0.101	0.102	0.007
comp	0.158	0.164	0.160	0.162	0.006
cont	0.134	0.137	0.130	0.133	0.009
cls	-0.117	-0.090	-0.088	-0.101	0.013
Layers					
0-0	0.166	0.160	0.157	0.164	0.005
1-1	0.075	0.095	0.087	0.084	0.008
11-11	-0.070	-0.038	-0.045	-0.055	0.012
12-12	0.105	0.146	0.131	0.128	0.014
1-4	0.227	0.227	0.216	0.223	0.007
8-12	-0.029	0.000	-0.003	-0.016	0.011

Table 11: Mean correlations for compositionality prediction (of the modifier) across parameter choices.

C Effect of layer combinations

Model performance across layer combinations is plotted in Figure 4.

D Effect of compositionality estimates

Model performance across individual compositionality estimates is plotted in Figure 5.

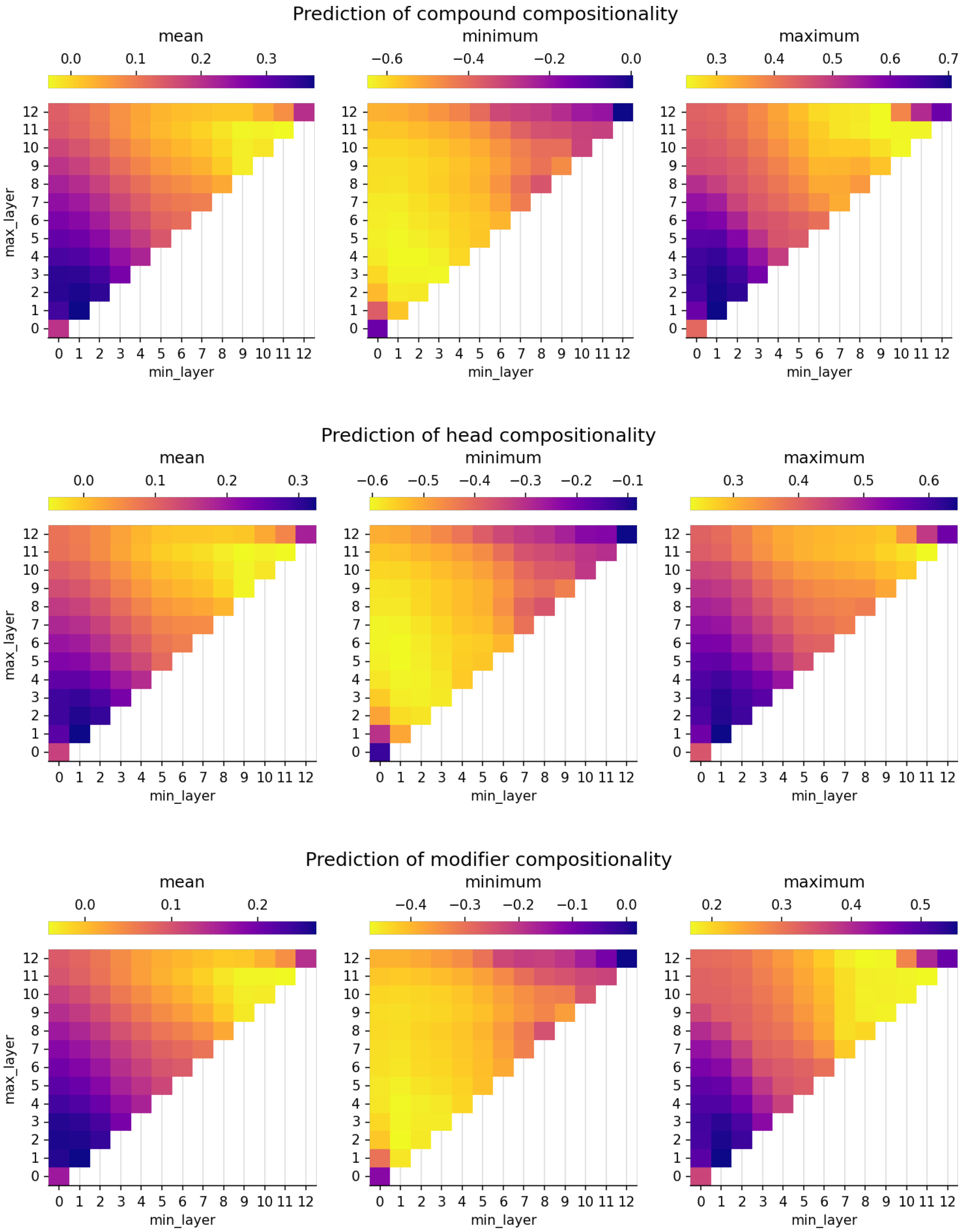


Figure 4: Correlations for compositionality prediction across layer combinations. The `min_layer` and `max_layer` values are start and end points of a contiguous span of layers. For each prediction target (compound, head, and modifier compositionality score), the left panel corresponds to the mean correlation per layer combination; the middle and the right panels correspond to the minimum and maximum values, respectively.

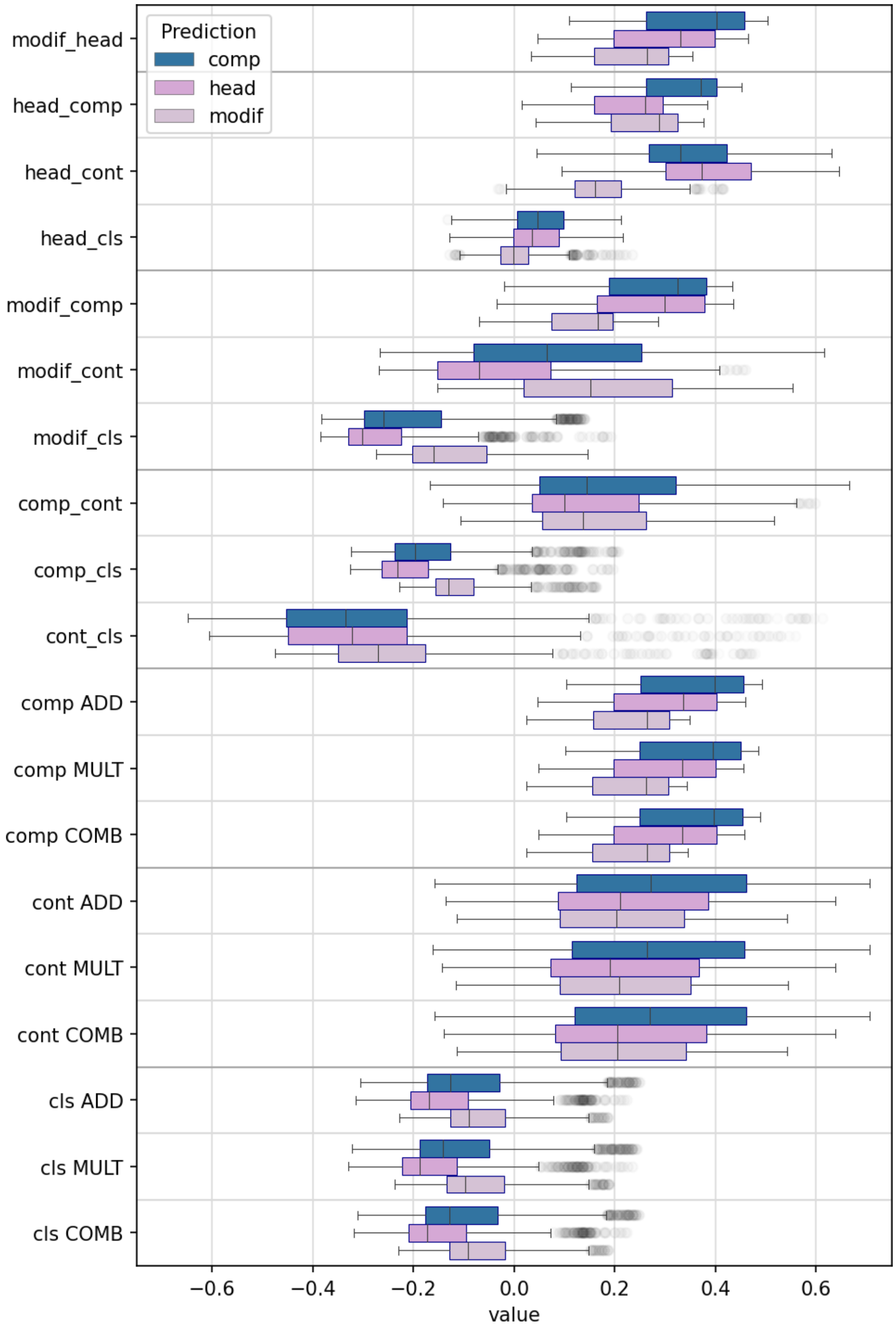


Figure 5: Distribution of Spearman's correlation coefficient for compositionality prediction across compositionality estimates. Embedding types: `modif` = contextualized representation of the modifier; `head` = contextualized representation of the head; `comp` = pooled representation of `modif` and `head`; `cont` = pooled representation of the surrounding context (full sequence without the compound); `cls` = representation of the `[CLS]` token. The use of ADD, MULT, and COMB involves the composite estimates described in Section 4.

Efficiently Upgrading Multilingual Machine Translation Models to Support More Languages

Simeng Sun[1*] **Maha Elbayad**[2] **Anna Sun**[2] **James Cross**[2†]
University of Massachusetts Amherst[1] Meta AI[2]
simengsun@umass.edu
{elbayadm, annaysun, jcross}@meta.com

Abstract

With multilingual machine translation (MMT) models continuing to grow in size and number of supported languages, it is natural to reuse and upgrade existing models to save computation as data becomes available in more languages. However, adding new languages requires updating the vocabulary, which complicates the reuse of embeddings. The question of how to reuse existing models while also making architectural changes to provide capacity for both old and new languages has also not been closely studied. In this work, we introduce three techniques that help speed up effective learning of the new languages and alleviate catastrophic forgetting despite vocabulary and architecture mismatches. Our results show that by (1) carefully initializing the network, (2) applying learning rate scaling, and (3) performing data up-sampling, it is possible to exceed the performance of a same-sized baseline model with 30% computation and recover the performance of a larger model trained from scratch with over 50% reduction in computation. Furthermore, our analysis reveals that the introduced techniques help learn the new directions more effectively and alleviate catastrophic forgetting at the same time. We hope our work will guide research into more efficient approaches to growing languages for these MMT models and ultimately maximize the reuse of existing models.

1 Introduction

Research into multilingual machine translation (MMT) (Aharoni et al., 2019; Fan et al., 2020) has shifted from a relatively small number of translation directions (Dong et al., 2015; Firat et al., 2016; Ha et al., 2016) to much larger scale, recently reaching up to tens of thousands of translation directions (NLLB Team et al., 2022; Bapna et al., 2022). Despite the notable increase in the number of supported languages, these models still need to be upgraded as increasing amount of data in new languages are becoming available. The process of adding new languages to existing models is an instance of continual learning (Ring, 1994), in which the models need to effectively learn new tasks (new translation directions) while not catastrophically forgetting (French, 1993; McCloskey and Cohen, 1989) the knowledge about tasks from the previous training stage (original translation directions).

Unlike the most conventional continual learning setup where the model remains the same as the initial learning stage, growing languages for MMT models needs to deal with *new* parameters. Adding new languages to existing training data shifts subword token distribution (e.g., tokens from unseen scripts are added), hence the need to re-train the tokenizer, which adds new embedding parameters to the MMT model. Previous studies have shown the effectiveness of adapting embedding parameters to retain performance on old translation directions (Lakew et al., 2018; Escolano et al., 2019; Garcia et al., 2021). This is usually done by reusing the embeddings for tokens that overlap between the old and new vocabularies.

One aspect that has not been extensively addressed in previous research is how to deal with other architecture mismatches that may arise during the continual learning phase. When growing MMT models to support many additional languages, it also makes sense to increase the model size overall. This extra capacity can be used to learn not only the new directions well but also the old directions better. It is not obvious, however, how to reuse the parameters from existing models (i.e., to engage in continual learning) given such architectural changes. Thus, in addition to dealing with different vocabularies, we also investigate how to make use of previously trained models when scaling up model size in order to train much more efficiently than training from scratch.

---

[*] Work done during an internship at Meta AI.

[†] Work done while working at Meta AI.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1513–1527
May 2-6, 2023

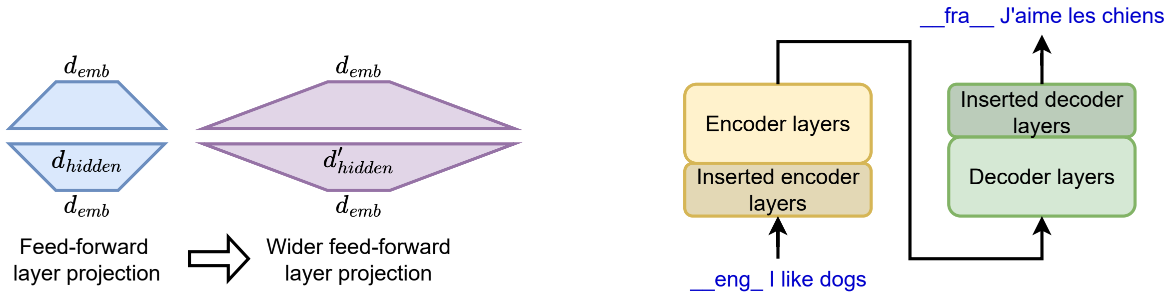


Figure 1: Illustration of two architectural mismatches we tackle in this paper. **Left:** The hidden dimension in the feed-forward layers is doubled ($d'_{\text{hidden}} = 2d_{\text{hidden}}$) during the continual learning stage so that the model becomes "wider". **Right:** Additional layers are inserted at the bottom of encoder and the top of decoder so that the model becomes "deeper". Both architectural changes increase capacity and are not well-addressed by previous works.

In this work, we introduce a recipe that helps significantly reduce the amount of required computation for continually learning new languages in MMT models. The recipe involves training the models with a combination of three techniques: (1) careful initialization of the network (2) applying learning rate scaling and (3) up-sampling selected language pairs. We validate our method on settings both with and without *architecture mismatches* (e.g., models becoming wider or deeper as shown in Figure 1 to have extra capacity for both old and new directions). We compare our models with strong same-sized baselines that are trained on all data from scratch. Our experimental results show that, without architecture mismatches, it is possible to outperform the baseline with just 30% of computation required by the baseline. When training larger models, less than 50% of the original computation is required to match the full performance of the wider baseline model, and less than 10% of computation is needed to recover over 95% of the corresponding baseline performance. We further conduct a suite of analysis which shows that:

- Proper initialization of the parameters before continual learning is crucial for fast convergence.
- Data up-sampling is vital to achieving good performance on new language pairs.
- Scaling down the learning rate for reused parameters helps alleviate catastrophic forgetting.

It is our hope that this work will help save computation for future research into large-scale multilingual machine translation, guide more efficient reuse of existing models for continual learning, and allow people to efficiently adapt large publicly-released MMT models for new languages and datasets.

2 Method

Adding new languages to existing models, especially languages in new scripts, leads to different subword tokenization, thus different vocabularies, which precludes simple fine-tuning with the exact same model on additional data. During the continual learning stage, we may also want to increase the model size overall to have extra capacity to learn the new languages and improve old languages at the same time. Therefore, we also investigate two typical architectural changes commonly done to increase the network capacity: (1) make the model "wider" by expanding the hidden dimension of the feed-forward layers, and (2) make the model "deeper" by inserting new layers to both encoder and decoder. In this section, we delineate three techniques that we found most effective in achieving computation reduction for the continual learning of MMT models.

Proper initialization. While we can copy weights from the old model[1] to the new model, it is not immediately clear how the new parameters (e.g., new token embeddings, new feed-forward weights, new layers) should be initialized and co-adapted with the old weights such that maximal knowledge about the old directions is retained. Instead of initializing the new parameters randomly, we find that initializing the new embeddings with that of <unk> leads to the best performance. When

---

[1] The weights that can be directly copied without any modification include (1) the token embeddings that overlap between old and new model, (2) same-shaped non-embedding parameters.

		M20	**M25**$^{@100k}$	**Mt25**$^{@30k}$	**M25**$^{@100k}_{wide}$	**Mt25**$^{@50k}_{wide}$	**M25**$^{@100k}_{deep}$	**Mt25**$^{@50k}_{deep}$
spBLEU	All	33.4	31.6	31.8	32.8	32.8	32.8	32.4
	Orig.	33.4	33.4	33.5	34.5	34.25	34.5	33.9
	Added	-	24.6	25.2	25.9	27.2	26.2	26.2
chrF++	All	76	74	74	75	75	75	75
	Orig.	76	76	76	76	76	76	76
	Added	-	67	69	68	70	68	69

Table 1: While continually learning new languages by bootstrapping from a model trained on 20 languages (M20), given new embedding parameters (vocabulary mismatch), one can exceed the performance of a baseline model trained on all languages from scratch (M25$^{@100k}$) with just 30% computation (Mt25$^{@30k}$). With architectural mismatches, employing our method, the wider-model baseline performance (M25$_{wide}$) and over 98% performance of a deeper-model baseline (M25$^{@100k}_{deep}$) can be recovered with half of the corresponding baseline computation (Mt25$^{@50k}_{wide}$ and Mt25$^{@50k}_{deep}$ respectively). The “Orig.” row shows performance on old 20 languages and the “Added” on newly added 5 languages.

the network becomes “wider” in the continual learning phase, concatenating each old weight matrix with a noisy version of itself performs better than other methods we tried. When the network becomes “deeper”, initializing new layers with averaged weights of old layers results in slight improvement over other naive initialization methods. For each setup, we compare different initialization methods in section 4.2.

Data up-sampling. Similar to (Garcia et al., 2021), we introduce the new tasks by mixing old and new training data together. Since the main goal of the continual learning phase is to quickly learn the new directions, we up-sample the new pairs so the model gets more learning signals from these new directions. To increase the transfer across related language pairs and reduce the chance of catastrophically forgetting less represented directions in the original training data, we also up-sample a small number of the old low-resource languages that are from the same language family as the new languages. We present the effect of up-sampling these selected directions in section 4.3.

Learning rate scaling. To further mitigate the frequent catastrophic forgetting problem exhibited in continual learning (Ring, 1994; Thompson et al., 2019), we scale the learning rate for individual parameters depending on whether they are copied from the old model or not. Based on the assumption that the model better retains the knowledge about old tasks if the weights stay close to that of the old model (Kirkpatrick et al., 2017), we scale down the learning rate for these old parameters while maintaining or scaling up the learning rate for the newly added parameters. In contrast to other methods that incur extra computation such as Fisher Information based loss (Thompson et al., 2019), our approach is simple, straightforward and efficient in alleviating catastrophic forgetting. We present a deeper analysis of learning rate scaling in section 4.4 and section 5.2.

3 Experiments

Languages. We conduct our experiments on 25 languages covering ten language families, four resource levels (high, mid, low, and very low) and four scripts[2]. To grow the languages atop existing learned languages, we train the seed model on 20 languages (40 English-centric directions) and add 5 new languages (10 English-centric directions) during the continual learning stage. Mimicking the common scenario where newly added languages are mostly low-resource, we select five low and very-low resource languages covering four language families and four scripts as the new languages to add to the seed model. To verify the validity of our approach, we also experiment with two other 20/5 groupings and one 12/13 division of old/new languages in section 5.1[3]. Table 2 shows the groups of original and added languages used in most of our experiments. For all our models, we train `sentencepiece` (Kudo, 2018)[4] tokenizers with 64K tokens and a sampling temperature of 2, on joined (source and target) data from all language directions available in each setup. This

---

[2] We include more details about these languages in Appendix A Table 9.

[3] More language grouping details are shown in Table 10 and Table 11 in Appendix A.

[4] `https://github.com/google/sentencepiece`

Original				Added
lav	rus	fin	spa	xho
lit	hin	est	deu	guj
swh	mar	pol	zul*	npi
ukr	mkd	ces	msa*	ind
bel	bul	fra	kir*	kaz

Table 2: The M20 model is trained on the "Original" languages. Mt25, Mt25$_{wide}$ and Mt25$_{deep}$ bootstrap from M20 and train on the combination of "Original" and "Added" languages. We perform data up-sampling over added data in conjunction with related old low-resource languages (marked with *).

means that tokenizers in the initial training on the the old languages, were not exposed to unseen languages. The data we used are subsets of data used by NLLB Team et al. (2022) without any mined or back-translated examples.

Models. Our baseline models have a standard 12-layer encoder and 12-layer decoder Transformer architecture (Vaswani et al., 2017) with 16 attention heads, embedding dimension 1024 and hidden dimension 4096. We refer to the baseline model trained on 20 languages and 25 languages from scratch as M20 and M25 respectively. For wider models, we double the hidden dimension size to 8192; and for deeper models, we insert 6 extra layers to encoder and decoder each. We refer to the baseline wide and deep models trained on 25 languages from scratch as M25$_{wide}$ and M25$_{deep}$ respectively, and refer to the models which bootstrap from the M20 model and trained to support new languages while remaining the same architecture, using a wider network, and deeper network as Mt25, Mt25$_{wide}$ and Mt25$_{deep}$ respectively. All translation directions were evaluated on FloRes (Goyal et al., 2022; Guzmán et al., 2019) dev set and use `spBLEU`[5] dand `chrF++` (Popović, 2017) to evaluate the best checkpoint selected by validation loss with beam size 4.[6]

Training details We train baseline models (M25, M25$_{wide}$, M25$_{deep}$) for 100K updates with batch size ∼444K tokens peak learning rate 0.003 warmed up with 8000 steps. For the bootstrapped models, we train Mt25 for 30k updates and Mt25$_{wide}$ and Mt25$_{deep}$ for 50k updates since there are more added parameters. We use temperature 1 and prepend encoder/decoder language token at the beginning of each example. All models are trained with attention dropout 0.1 and label smoothing with $\epsilon = 0.1$ (Szegedy et al., 2016). The baseline models M25, M25$_{wide}$ and M25$_{deep}$ complete training of 100K updates on 64 GPUs in ∼24h, ∼29.9h, and ∼29.0h respectively. We do not reset the learning rate scheduler for the second training phase.[7] The data of selected languages are up-sampled by 5 and the learning rate of the old parameters in Mt25$_{wide}$ is multiplied by 0.5 and in Mt25$_{deep}$ by 0.05 at the beginning while linearly increasing to 0.5. We present the effect of these hyperparameters in the next section.

	All	Orig.	Added
Mt25	**31.8**	**33.5**	**25.2**
random init all	28.2*	29.4*	23.4*
random init new	31.7	33.5	24.9*
no up-sampling	31.3*	33.4	22.6*
no lr scaling	31.5*	33.1*	25.1
Mt25$_{wide}$	**32.6**	**34.1**	**26.3**
random init all	26.7*	28.0*	21.5*
random init new	29.9*	31.3*	24.6*
no up-sampling	31.9*	34.1	23.3*
no lr scaling	32.3*	33.9*	26.2*
Mt25$_{deep}$	**32.2**	33.8	25.6
random init all	30.5*	31.7*	**25.7**
random init new	32.2	**33.9**	25.6
no up-sampling	31.2*	33.5*	22.2*
no lr scaling	31.6*	33.0*	26.1

Table 3: Ablation of the effect of not using M20 weights (random init all), the effect of not carefully initializing new parameters (random init new), not having data up-sampling and not applying learning rate scaling across three scenarios. All numbers are spBLEU scores of models evaluated after 30K updates. * indicates p-value of paired T-test against corresponding best model (Mt25, Mt25$_{wide}$, Mt25$_{deep}$) is smaller than 0.05.

---

[5] `https://github.com/facebookresearch/flores/tree/main/previous_releases/flores101#spbleu-evaluation`

[6] We use these two metrics instead of BLEU or pre-trained metrics (Kocmi et al., 2021) because over half of the languages we use are of mid- to low-resource and Goyal et al. (2022) show spBLEU treats low-resource languages more fairly.

[7] We tried resetting learning scheduler (i.e., learning rate warms up 8K steps from 0 to 0.003 and then starts to decrease), small constant learning rate, and not resetting learning rate scheduler in our preliminary experiments and the last leads to the best results.

Main results. Results in Table 1 indicate that by properly applying the techniques introduced earlier, it is possible to recover the baseline performance with significantly less computation. When architecture remains the same during the second learning phase, better overall performance can be achieved (31.8 vs. 31.6) after 30% of the updates required by the baseline model M25. The gains come from effective learning of both the old and new directions while the latter seems to be better learned than the baseline model (25.2 vs. 24.6). While training into a wider model, applying our techniques recovers the performance with approximately half of the M25$_{wide}$ computation. Although we did not fully recover the performance while training into a deeper model, over 98% of the baseline performance can be achieved using our techniques with half of the baseline M25$_{deep}$ computation. We find that the major degradation in Mt25$_{deep}$ comes from the worse performance on the original directions (33.9 vs. 34.5) which suggests mitigating catastrophic forgetting is harder when expanding the network by depth than by width.

4 Effect of each technique

4.1 Ablation study

To understand the effect of each introduced technique, we conduct an ablation study where each model is trained on the same configuration except for one essential element (i.e., proper initialization, learning rate scaling, data up-sampling). As a naive baseline, while having scaled learning rate and data upsampling, we include the models where no weights from the seed M20 model are used ("random init all") to compare with a less naive baseline "random init new" where the M20 weights are reused and only the newly added parameters (i.e., new token embeddings in all three models, new fully-connected layer weights in Mt25$_{wide}$ and weights of new layers in Mt25$_{deep}$) are randomly initialized. To summarize, each configuration in Table 3 corresponds to the following:

Random init all : All parameters are initialized randomly while model is trained with data up-sampling and learning rate scaling.

Random init new : Newly added parameters are initialized randomly while weights of M20 are copied to the new model. Model is trained with data up-sampling and learning rate scaling.

No up-sampling : Model weights are properly initialized and their learning rates are scaled during training. No language pair is up-sampled.

No lr scaling : Model weights are properly initialized and low-resource pairs are up-sampled whereas no learning rate scaling is applied.

Results in Table 3 confirm the contribution of each of the three introduced techniques in achieving the desired performance across different settings. Overall, not reusing the M20 weights leads to worse performance than the baseline by 2∼6 BLEU in different settings. While reusing the old model's weights, also having proper initialization of the *new* parameters yields better performance than simply initializing with default normal distribution. The benefit is most obvious when training into a wider model (29.9 vs. 32.6) compared to the other two settings. We also observe that data up-sampling is crucial to achieving good performance on the new directions. Not applying up-sampling degrades around ↓ 3BLEU on new directions across all settings, while barely or just slightly hurting the performance on the old directions. On the other hand, not applying learning rate scaling harms the performance of old directions across all settings, which suggests the effectiveness of scaling the learning rate to mitigate catastrophic forgetting, about which we include a more detailed analysis in section 5.2.[8] Since learning rate scaling helps counteract catastrophic forgetting and up-sampling speeds up learning of the new directions, we discover that their effectiveness are additive – better performance can be achieved on both old and new directions by combining these two techniques.

4.2 Effect of proper initialization

In this section we briefly discuss different variations we attempted for initializing new parameters and present the results in Table 4.

Mt25 In the case of having only mismatched vocabularies, we find that dropping the entire old embedding table ("all emb random") causes large drop in performance, and that initializing the new embeddings with that of <unk> leads to slightly better performance than random initialization.

---

[8] The overall differences are small as the results are averaged over 50 directions, however, the paired T-Test shows statistically significant improvements.

[9] We did not apply learning rate scaling in this setup and do not up-sample the old related low-resource languages. All evaluations are done after 30K updates.

	All	Orig.	Added
Mt25			
all emb random	30.7	32.1	25.1
new random	31.5	33.0	25.5
new <unk>	**31.5**	**33.0**	**25.7**
Mt25$_{wide}$			
new random	31.1	32.3	26.1
linear interp	31.8	33.3	26.0
concat	**32.0**	**33.5**	**26.1**
M20t25$_{deep}$			
random	31.5	32.8	26.3
closest layer	31.6	32.9	26.3
average layer	**31.6**	**32.9**	**26.7**

Table 4: Initializing the new embeddings with that of `<unk>`, concatenating the original weight with a noisy version of itself in Mt25$_{wide}$ and initializing new layer in encoder/decoder with the averaged encoder/decoder weights in Mt25$_{deep}$ achieve the best performance.[9]

Mt25$_{wide}$ A naive method to initialize wider feed-forward layer projections is to expand the old weight matrix with a randomly initialized weight matrix. However, having random additional parameters messes up the output of feed-forward layer, which interferes with the inter-dependency among layers, thus we also tried concatenating the old weight matrix with a noisy version of itself (concat[10]) followed by a normalization operation[11] to keep the output as close to the original projection output as possible. In addition, rather than maintaining the old weights in block, we also tried linearly interpolating the original weight matrices along the expanded dimension (linear interp). It's important that the new parameters in the two projection matrices in a feed-forward layer match each other along the hidden dimension axis. Results indicate that weight concatenation is a simple and effective way that allows for faster convergence.

Mt25$_{deep}$ Our preliminary experiments show that inserting new layers at the bottom of the encoder and at the top of the decoder is more effective than other attempted scenarios, so we went with this setup in all our Mt25$_{deep}$ experiments. We tried different methods of initializing the inserted layers, including random initialization, copying parameters from the closest layer,[12] and averaging the weights across all encoder or decoder layers. Although the benefit is small, we do perceive that initializing the new layers with averaged layer weights results in better performance, especially on new translation directions (26.7 vs. 26.3).

Up-sampling config.	All	Orig.	Added
$\alpha = 1$ (No up-sampling)	31.0	**33.2**	22.2
$\alpha = 5$	31.5*	33.0*	**25.7***
$\alpha = 10$	31.1	32.6*	25.5*
$\alpha = 5$ + rel. low-resource	**31.5***	33.1	25.1*

Table 5: Performance of Mt25 at 30K updates without applying learning rate scaling. Up-sampling new languages with a reasonable ratio during continual learning leads to large gains on these new directions. * indicates p-value of paired T-test against the baseline (top row) is smaller than 0.05.

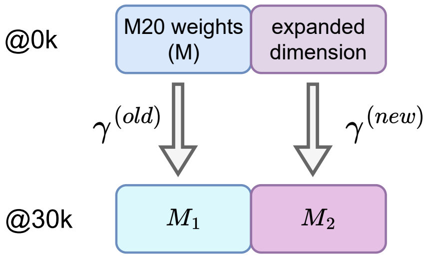


Figure 2: Before the continual learning (@0k), the wider hidden projection matrix is initialized with a concatenation of M20 weights M and a noisy version of itself. During the continual learning stage, the learning rate for weights copied from old model is scaled by $\gamma^{(old)}$, the rest are scaled by $\gamma^{(new)}$.

4.3 Effect of data up-sampling

We multiply the original dataset size of each upsampled direction by a value α before computing the final up-sample ratio. The new directions thus constitute a larger portion in the new dataset than if not. We show in Table 5 the effect of the up-sample factor α. We find that up-sampling selected languages leads to much better performance (+3BLEU) on new directions than without up-sampling. However, up-sampling the new directions too much (e.g., $\alpha = 10$) worsens the performance on old directions while not improving the new directions. In general, we find the up-sampling

---

[10] We inject zero-mean Gaussian noise with $std = 0.01$. We also tried not adding noise to the new parameters, which has almost identical performance. To avoid both parts of the weight matrices learning redundant information, we decide to add small perturbation to the new parameters.

[11] We normalize the new weight matrix such that it has similar Frobenius norm as the old weight matrix.

[12] Since we include the new layer at the bottom of the encoder and at the top of the decoder, the closest layer means the first encoder layer and last decoder layer respectively.

	All	Orig.	Added
$\gamma^{(old)} = 1, \quad \gamma^{(new)} = 1$	32.0	33.5	26.1
$\gamma^{(old)} = 0.5, \gamma^{(new)} = 5$	**32.3***	**33.8***	**26.2**
$\gamma^{(old)} = 0.5, \gamma^{(new)} = 0.5$	31.5*	33.3	24.7*
$\gamma^{(old)} = 0.5, \gamma^{(new)} = 1$	32.1*	33.6*	26.0
$\gamma^{(old)} = 5, \quad \gamma^{(new)} = 5$	31.9	33.4	26.1

Table 6: Performance of Mt25$_{wide}$ after 30K updates while applying different learning rate scaling factor based on notation in Figure 2. These experiments adopt an earlier setup where the old related low-resource languages are not up-sampled. * indicates p-value of paired T-test against the baseline (top row) is smaller than 0.05.

factor $\alpha = 5$ adopted in previous studies (Garcia et al., 2021; Berard, 2021) suitable to many other variants we have attempted in this work. Besides merely up-sampling the new directions, we also up-sample the old low-resource directions that belong to the same language family as any of the new directions. Doing so slightly improves the performance on old directions whereas causes a drop in new directions, which further confirms the critical role of up-sampling in balancing the performance between the old and new directions.

4.4 Effect of learning rate scaling

As described in section 2, we scale down the learning rate for old (reused) parameters. In the case of Mt25, all parameters are updated with a smaller learning rate than the new token embeddings. In the wider network, as we have established the effectiveness of using concatenated weights, we apply different learning rate to each part of the weight matrices as generically illustrated in Figure 2. In Mt25$_{deep}$, the new layers are updated with a larger learning rate while the rest of the parameters receive a smaller learning rate. Table 6 suggests that having a smaller learning rate for old parameters is more favorable than scaling all parameters by the same amount ratio. Scaling down the update for all parameters slows down the learning of the new directions, whereas scaling up with a larger value does not improve the performance either. While already applying smaller update on old parameters, scaling up the learning rate for the new parameters can in fact improve both the old and new directions (the top two rows in Table 6). Overall we find that learning rate scaling is an effective and easy-to-implement alternative to previous methods (Kirkpatrick et al., 2017) in terms of alleviating catastrophic forgetting.

	Mt25	Mt25_v1	Mt25_v2	M12t25
All	32.6	32.7	32.5	32.7
High	35.8	36.0	35.8	35.9
Mid	29.6	29.8	29.7	29.3
Low	31.5	31.2	31.4	31.6
V_Low	19.0	20.0	19.0	20.1

Table 7: Besides the language breakdown shown in Table 2, we show our approach also generalizes to other groupings of old and new languages (Mt25_v1, Mt25_v2 and M12t25), details about which are included in Appendix A Table 10.

	Mt25		Mt25_v1		Mt25_v2	
	ΔBLEU	Up	ΔBLEU	Up	ΔBLEU	Up
eng-bel	-1.2	N	2.0	Y	2.4	Y
eng-guj	1.6	Y	0.9	Y	0.0	N
eng-npi	0.8	Y	-0.1	Y	-2.3	N
eng-xho	1.3	Y	0.0	N	1.1	Y
bel-eng	-1.1	N	-0.5	Y	-0.7	Y
guj-eng	-2.5	Y	-2.6	Y	-2.2	N
npi-eng	-1.6	Y	-0.6	Y	-1.9	N
xho-eng	-1.1	Y	-1.3	N	-0.7	Y

Table 8: Zoomed in analysis over specific language pairs. Our approach is more effective in learning eng→xxx directions than xxx→eng directions across different seed-language setups regardless if the language is up-sampled (Up=Y) or not (Up=N).

5 Analysis

5.1 How does the choice of seed languages affect continual learning?

In addition to the core setup described in Table 2, we also experiment with other settings. We considered three new setups following the natural scenario where the seed languages are mostly of high- and mid-resource languages. Two of the new setups still adopt the 20/5 division between old and new languages, but emphasize different scenarios — Mt25_v1 covers one mid resource language in the set of new languages and Mt25_v2 does not add any new script. The last one instead initializes from a model trained on 12 high- and mid-resource languages and adds in 13 new languages.[13] We follow the same configuration used in the previous experiments and train into wider models in the continual learning phase. Results in Table 7 demonstrate that our approach successfully generalizes to all the new settings, as similar performances can be achieved

---

[13] For detailed breakdown of these three new settings, we refer readers to Table 10 and Table 11 in Appendix A

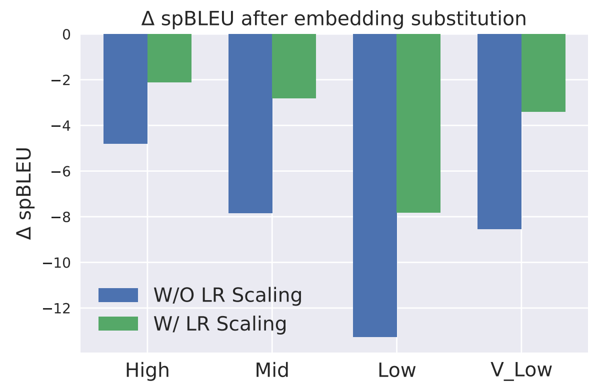


Figure 3: We measure the drop in spBLEU on original directions after substituting overlapped embeddings in $Mt25_{wide}$ back to M20 model. Applying learning rate scaling helps alleviate catastrophic forgetting as spBLEU drops less than without LR scaling.

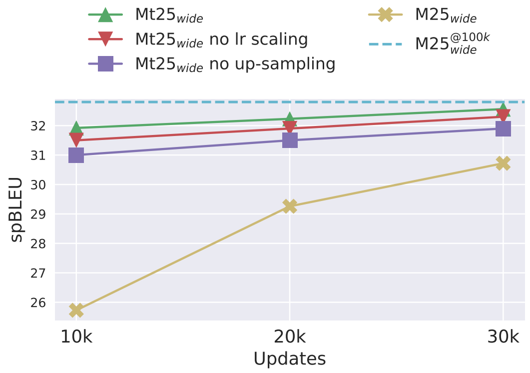


Figure 4: Training the M25 from scratch without using the M20 weights converges significantly slower. Applying the techniques introduced in this work results in the largest computation reduction, achieving over 95% performance with less than 10% baseline computation.

when models are trained on different choices of seed languages, even when the number of seed languages are different. However, we do observe small differences after zooming in by resource-levels. A more fine-grained analysis over a few low-resource languages that cover diverse scripts (bel-Cyrl, guj-Gujr, npi-Deva, xho-Latn) is presented in Table 8. In general, regardless of if the language pair is in the seed language set or not, the eng→xxx directions are learned faster than xxx→eng directions, since most of the pairs already exceed the baseline performance after just 30K updates (compare the upper part of Table 8 to the lower part). We also verify the effectiveness of data up-sampling for eng→xxx directions. Up-sampling these pairs leads to much better performance than when not up-sampling (e.g., compare performance of eng-bel in the three setups).

5.2 How effective is learning rate scaling in mitigating catastrophic forgetting?

To quantitatively measure the amount of information lost after the continual learning phase, we adopt an evaluation setup akin to (Garcia et al., 2021), in which the embeddings in M20 that overlap with Mt25 are substituted with the corresponding embeddings in Mt25. One can evaluate this new M20 model with substituted embeddings on the original 20 languages, and use the drop in spBLEU as a proxy for the amount of knowledge lost in the embeddings due to catastrophic forgetting. In Figure 3, we display the spBLEU drop after substituting the embeddings of a $Mt25_{wide}$ model trained with and without learning rate scaling. For both variants, the spBLEU scores drop, indicating that some information in the embeddings is lost after training on the new languages. We also find that, for both models, the decrease in spBLEU is larger for (very) low-resource languages than high and mid resource languages, which suggests (very) low-resource languages are easier to be "forgotten" in the second training phase, which reinforces our decision of also up-sampling the related low-resource languages as introduced in section 4.3. Finally, not applying learning rate scaling leads to much larger decrease in all directions, which manifests the effectiveness of scaling down the learning rate for alleviating catastrophic forgetting.[14]

5.3 Computation saved

Due to the mismatched vocabularies and architectures, models trained on the combination of new and old languages are typically re-trained from scratch after already incurring large computation on old languages. In this section, we look at how much computation can be effectively saved with the approach proposed in this paper. Note that, for both our methods and the retraining approach, M20 is already trained, and its computation cost is excluded from our calculation.

Although results in Table 1 shows that 50% of the baseline computation is required to recover the baseline performance, we show in Figure 4 that much computation can be saved if we slightly slack the target performance. The $M25_{wide}$ model

---

[14] Besides probing the effect of learning rate scaling on embeddings, we also present an analysis on the expanded feed-forward weights in Appendix C.

trained from scratch reaches only 78% of the baseline performance after 10k updates, while models trained with the combination of our techniques achieve over 97% of the baseline performance after the same amount of updates (10% total computation of training Mt25$_{wide}$).

6 Related Work

Our work is closely related to prior research on adapting existing MMT models (Mohammadshahi et al., 2022) to new languages (Lakew et al., 2019; Kocmi, 2020). Neubig and Hu (2018) add low-resource languages to multilingual models by fine-tuning on low-resource data while regularizing with related high-resource data. Garcia et al. (2021) introduce simple vocabulary substitution for adapting MMT models to new languages without any architectural changes. Another line of research employs modular approaches, which include training lightweight adapters (Bapna and Firat, 2019), language-specific encoder-decoders (Escolano et al., 2019, 2021), language specific embeddings (Berard, 2021) for learning new languages. While sometimes escaping the need to train on old examples, growing the model in a modular fashion (Rusu et al., 2016) requires non-trivial changes to standard architectures. In contrast, our work relies on rehearsal mechanism (i.e., also train on old examples) but does not need to modify network structures (Robins, 1995).

Our approach is also related to works that focus on continual learning of MT models for adapting multiple domains. Thompson et al. (2019); Gu and Feng (2020) adopt a method derived from Elastic Weight Consolidation (Kirkpatrick et al., 2017) to alleviate catastrophic forgetting. While most prior works only investigate two-stage continual learning, Cao et al. (2021) propose a new framework that extends to multi-stage training to mitigate catastrophic forgetting (Ring, 1994). The initialization of the new parameters and embeddings in our technique is also related to that in (Pfeiffer et al., 2021), which accommodate multilingual models to unseen scripts via matrix factorization. Our focus on the architectural differences between initial and continual learning phase is also relevant to recent discoveries that wider networks forget less catastrophically (Mirzadeh et al., 2022).

7 Conclusion

We show in this work that it is possible to efficiently bootstrap from existing models and recover the baseline performance with much less computation while vocabularies and architectures can be different in the continue learning stage. We highlight the importance of (1) reusing the existing model weights and carefully initializing the new parameters, (2) applying learning rate scaling, and (3) performing data up-sampling. Analyses reveal that scaling down the learning rate for old parameters helps alleviate catastrophic forgetting, and that data up-sampling is vital to achieving good performance on the new directions. We hope our work can help save computation for research into large-scale multilingual MT models, and more generally, will help spur research into continual multitask learning in the presence of architectural changes.

Limitations

While we explore the under-studied architectural mismatches for continual learning of MMT models, we focus exclusively on adding new languages in bulk, without investigating adding languages one by one continuously. Furthermore, due to limited computational resources, we only experimented with a few typical scenarios where the new languages are low or very-low resourced. Experiments on other groupings of old and new languages could further validate the effectiveness of our approach.

Acknowledgement

We thank Shruti Bhosale, Jean Maillard, Philipp Koehn, Francisco Guzmán, Angela Fan, Holger Schwenk, Haoran Xu, Steven Tan and Alex Guo for their helpful discussion and support. We thank Vedanuj Goswami for helping fix technical issue at the end of this project.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Ankur Bapna, Isaac Caswell, Julia Kreutzer, Orhan Firat, Daan van Esch, Aditya Siddhant, Mengmeng Niu, Pallavi Baljekar, Xavier Garcia, Wolfgang Macherey, Theresa Breiner, Vera Axelrod, Jason Riesa, Yuan Cao, Mia Xu Chen, Klaus Macherey, Maxim Krikun, Pidong Wang, Alexander Gutkin, Apurva Shah, Yanping Huang, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2022. Building machine translation systems for the next thousand languages.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Alexandre Berard. 2021. Continual learning in multilingual NMT via language-specific embeddings. In *Proceedings of the Sixth Conference on Machine Translation*, pages 542–565, Online. Association for Computational Linguistics.

Yue Cao, Hao-Ran Wei, Boxing Chen, and Xiaojun Wan. 2021. Continual learning for neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3964–3974, Online. Association for Computational Linguistics.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732.

Carlos Escolano, Marta R. Costa-jussà, and José A. R. Fonollosa. 2019. From bilingual to multilingual neural machine translation by incremental training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 236–242, Florence, Italy. Association for Computational Linguistics.

Carlos Escolano, Marta R. Costa-jussà, José A. R. Fonollosa, and Mikel Artetxe. 2021. Multilingual machine translation: Closing the gap between shared and language-specific encoder-decoders. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 944–948, Online. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv preprint arXiv:1601.01073*.

Robert French. 1993. Catastrophic interference in connectionist networks: Can it be predicted, can it be prevented? In *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann.

Xavier Garcia, Noah Constant, Ankur Parikh, and Orhan Firat. 2021. Towards continual learning for multilingual machine translation via vocabulary substitution. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1184–1192, Online. Association for Computational Linguistics.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Shuhao Gu and Yang Feng. 2020. Investigating catastrophic forgetting during continual training for neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4315–4326, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. Two new evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. In *arXiv preprint arXiv:1902.01382*.

Thanh-Le Ha, Jan Niehues, and Alexander Waibel. 2016. Toward multilingual neural machine translation with universal encoder and decoder. *arXiv preprint arXiv:1611.04798*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Tom Kocmi. 2020. Exploring benefits of transfer learning in neural machine translation. *ArXiv*, abs/2001.01622.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 478–494, Online. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Surafel M. Lakew, Alina Karakanta, Marcello Federico, Matteo Negri, and Marco Turchi. 2019. Adapting multilingual neural machine translation to unseen languages. In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics.

Surafel Melaku Lakew, Aliia Erofeeva, Matteo Negri, Marcello Federico, and Marco Turchi. 2018. Transfer learning in multilingual neural machine translation with dynamic vocabulary. In *IWSLT*.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Seyed Iman Mirzadeh, Arslan Chaudhry, Dong Yin, Timothy Nguyen, Razvan Pascanu, Dilan Gorur, and Mehrdad Farajtabar. 2022. Architecture matters in continual learning. *arXiv preprint arXiv:2202.00275*.

Alireza Mohammadshahi, Vassilina Nikoulina, Alexandre Berard, Caroline Brun, James Henderson, and Laurent Besacier. 2022. SMaLL-100: Introducing shallow multilingual machine translation model for low-resource languages. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8348–8359, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 875–880, Brussels, Belgium. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2021. UNKs everywhere: Adapting multilingual language models to new scripts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10186–10203, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Mark Bishop Ring. 1994. *Continual Learning in Reinforcement Environments*. Ph.D. thesis, Graduate School of The University of Texas at Austin, USA. UMI Order No. GAX95-06083.

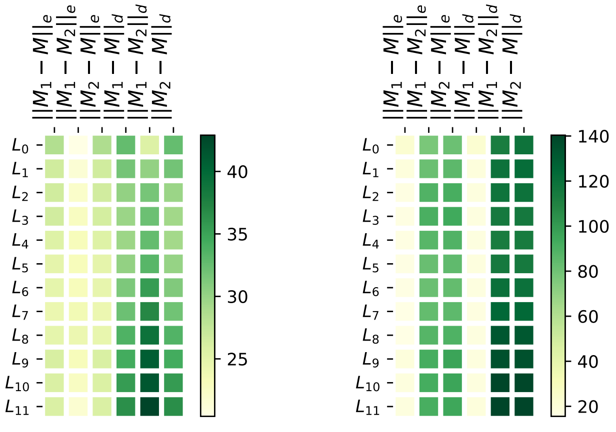
Anthony Robins. 1995. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. 2019. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2062–2068, Minneapolis, Minnesota. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.



w/o LR scaling w/ LR scaling

Figure 5: We measure the amount of change in each feed-forward layer's projection weights via Frobenius norm ($\| \cdot \|_e$ for encoder layers and $\| \cdot \|_d$ for decoder layers). Notation in this figure follows that in Figure 2. Scaling down the learning rate for parameters copied from the old model keeps them close to their initialization while pushes the newly added parameters farther away.

A Language details

We present two extra language grouping settings we have experimented in this section. To verify the validity of our approach on other language groupings, we also experiment with two other settings as shown in Table 10. In addition to dividing the old/new languages by 20/5, we also tried a setting where the seed model is trained on fewer languages as reported in Table 11.

B Detailed performance of each direction

Table 12 contains the detailed performance of all translation directions for all models reported in Table 1.

C Norm Analysis

The analysis in section 5.2 is limited to only measuring the lost information in the embeddings, to understand how learning rate scaling affects the weights other than embeddings, one natural extension is substituting other weights back to M20 as well. However, it leads to much worse performance for both variants as the inter-dependency among layers is impaired in this case. Therefore, we instead measure the amount that the weights of each encoder or decoder layer (denoted with $L_0, L_1, \dots, L_{11}$) change in the latent space via the Frobenius norm. Following the notation in Figure 2, we measure how much the weight matrices M_1 and M_2 have changed from the original weight matrix M ($\|M_1 - M\|$ and $\|M_2 - M\|$), as well as how much they differ from each other ($\|M_1 - M_2\|$). We refer to the Frobenius norm in an encoder or a decoder layer with $\| \cdot \|_e$ and $\| \cdot \|_d$ respectively. The trend in Figure 5 shows that applying learning rate scaling prevents M_1 from deviating too much from the original weights M and at the same time pushes the new parameters to space farther from its initialization. This is in contrast with the smaller differences when not applying learning rate scaling. The left side of Figure 5 indicates that even after the continual learning, M_1 and M_2 stay close to each other across the encoder layers, it is only when reaching the last several decoder layers do the two matrices demonstrate larger differences. Since we initialize M_1 and M_2 both based on M, having larger $\|M_1 - M_2\|$ reduces the chance of both parts learning redundant information i.e., effectively using the additional parameters.

D Scaling learning rate by Fisher information

Besides multiplying the learning rate for all old parameters with the same scaling factor, we also tried scaling the learning rate based on their Fisher Information. This is directly inspired by Elastic Weight Consolidation (Kirkpatrick et al., 2017), in which extra penalty is incurred when parameters crucial to the old tasks deviate too much from their original values. We plot the distribution of the per-token Fisher information of each parameter in Figure 6 (right). We further experiment with LR scaling over selected parameters that are supposed to be important for old tasks (Fisher information exceeds certain threshold based on Figure 6 (right)). Results in Table 11 (right) shows that among our attempted settings, scaling only part of the parameters based on Fisher information does not improve the overall performance. We conjecture that the performance could be improved if the Fisher Information is calculated on a larger set and that if we apply a piece-wise threshold function for scaling the learning rate of different parameters.

E Effect of LR scaling in alleviating catastrophic forgetting

We include the spBLEU drop on the old languages after substituting the embeddings of Mt25 back to M20 in Figure 6.

	Language	Language code	Language family	# bitext	Res. level	Script
M20 old languages	Latvian	lav	Baltic	2,288,307	Mid	Latn
	Lithuanian	lit	Baltic	2,322,619	Mid	Latn
	Swahili	swh	Benue-Congo	1,133,371	Mid	Latn
	Zulu	zul	Benue-Congo	656,450	Low	Latn
	Ukrainian	ukr	East Slavic	925,922	Low	Cyrl
	Belarusian	bel	East Slavic	55,185	V_Low	Cyrl
	Russian	rus	East Slavic	13,962,768	High	Cyrl
	Hindi	hin	Indo-Iranian	1,032,502	Mid	Deva
	Marathi	mar	Indo-Iranian	181,642	Low	Deva
	Malay	msa	Malayo-Polynesian	305,469	Low	Latn
	Macedonian	mkd	South Slavic	403,742	Low	Cyrl
	Bulgarian	bul	South Slavic	2,118,839	Mid	Cyrl
	Kyrgyz	kir	Turkic	336,554	Low	Cyrl
	Finnish	fin	Uralic	3,764,770	Mid	Latn
	Estonian	est	Uralic	2,410,543	Mid	Latn
	Polish	pol	West Slavic	3,062,818	Mid	Latn
	Czech	ces	West Slavic	23,792,604	High	Latn
	French	fra	Western European	20,031,051	High	Latn
	Spanish	spa	Western European	16,606,594	High	Latn
	German	deu	Western European	10,198,897	High	Latn
Mt25 new languages	Xhosa	xho	Benue-Congo	634,078	Low	Latn
	Gujarati	guj	Indo-Iranian	153,985	Low	Gujr
	Nepali	npi	Indo-Iranian	72,250	V_Low	Deva
	Indonesian	ind	Malayo-Polynesian	771,801	Low	Latn
	Kazakh	kaz	Turkic	627,734	Low	Cyrl

Table 9: Detailed information about languages used in the main setup of our experiments. Languages having >10M examples are of high-resource languages, having (1M, 10M] mid-resource languages. The rest are of low-resource languages, v_low is a subset of low-resource languages that have <100K examples.

Mt25_v1 Original				Added
lav	ukr*	msa	pol	guj
lit	rus	mkd	ces	npi
swh	hin	bul	fra	est
zul	mar*	fin	spa	bel
xho	ind	kaz*	deu	kir

Mt25_v2 Original				Added
lav	hin	mkd	pol	zul
lit	mar	bul	ces	xho
swh	guj	kaz*	fra	ind
ukr*	npi	fin	spa	bel
rus	msa*	est	deu	kir

Table 10: Besides the breakdown between old and new languages as shown in Table 2, Mt25_v1 uses the left grouping and Mt25_v2 uses the right grouping of languages. * marks the languages that are also up-sampled during continual learning.

Original		Added	
lav	est	zul	mar
lit	pol	xho	guj
swh	ces	ukr	npi
rus	fra	bel	ind
bul	spa	hin	msa
fin	deu	mkd	kaz
		kir	

	All	Orig.	Added
f=1e-6, $\gamma^{(old)}$=0.25	32.3	33.8	26.4
f=1e-6, $\gamma^{(old)}$=0.1	32.1	33.7	26.0
f=1e-5, $\gamma^{(old)}$=0.25	31.9	33.3	26.2
f=1e-5, $\gamma^{(old)}$=0.1	32.1	33.5	26.6
f=1e-5, $\gamma^{(old)}$=0.05	31.8	33.3	25.8
f=1e-4, $\gamma^{(old)}$=0.25	31.7	33.1	26.1

Table 11: **Left:** M12t25 starts from a model trained on 12 high- and mid- resource languages and grows to a wider model to support 13 new low- and very-low- resource languages. None of old languages is up-sampled in the continual learning stage. **Right:** Performance of Mt25$_{wide}$ after 30k updates when the learning rate for parameter whose Fisher information is greater than threshold f is scaled down by the corresponding $\gamma^{(old)}$ in each row.

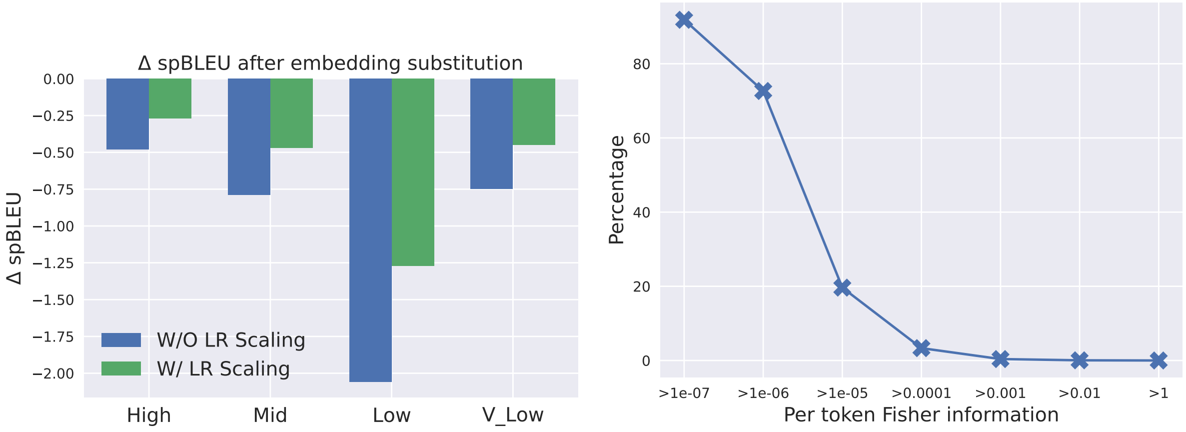


Figure 6: **Left:** spBLEU drop after substituting the embedding of the learned Mt25 model back to M20 model. **Right:** Per token Fisher information calculated over FloRes dev set.

Lang_pair	**M25**		**Mt25**		**M25**$_{wide}$		**Mt25**$^{(@30k)}_{wide}$		**M25**$_{deep}$		**Mt25**$^{(@30k)}_{deep}$	
	spBLEU	**chrF++**	**spBLEU**	**chrF++**	**spBLEU**	**chrF++**	**spBLEU**	**chrF++**	**spBLEU**	**chrF++**	**spBLEU**	**chrF++**
bel-eng	18.1	0.74	18.3	0.74	20.1	0.74	19.0	0.74	19.9	0.74	18.5	0.74
bul-eng	40.8	0.81	41.1	0.81	41.7	0.81	41.5	0.81	41.9	0.81	41.0	0.81
ces-eng	42.6	0.81	42.3	0.81	42.9	0.82	42.5	0.81	43.0	0.81	42.0	0.81
deu-eng	44.7	0.82	44.5	0.82	45.3	0.82	45.2	0.82	45.4	0.82	44.5	0.82
eng-bel	11.7	0.59	11.3	0.59	13.2	0.61	12.1	0.59	13.1	0.60	12.1	0.60
eng-bul	39.5	0.78	40.3	0.79	40.5	0.79	40.1	0.78	40.6	0.79	40.3	0.79
eng-ces	38.4	0.76	38.5	0.76	38.4	0.76	38.2	0.76	38.8	0.76	38.4	0.76
eng-deu	39.0	0.79	39.3	0.79	40.1	0.79	39.2	0.78	39.8	0.79	39.3	0.79
eng-est	30.3	0.76	30.1	0.76	32.0	0.77	30.7	0.76	31.5	0.76	30.7	0.76
eng-fin	28.5	0.76	28.5	0.76	30.3	0.76	29.3	0.76	30.7	0.77	29.5	0.76
eng-fra	49.4	0.81	49.4	0.81	48.6	0.81	49.3	0.81	49.1	0.81	49.1	0.81
eng-guj	24.2	0.63	28.3	0.67	25.5	0.64	28.4	0.67	25.3	0.64	28.0	0.66
eng-hin	32.0	0.71	31.5	0.71	33.4	0.72	32.6	0.71	32.7	0.71	32.8	0.71
eng-ind	40.1	0.81	40.0	0.81	41.2	0.81	44.4	0.82	41.4	0.81	43.7	0.82
eng-kaz	14.6	0.58	13.4	0.57	14.8	0.58	18.0	0.61	16.4	0.60	15.6	0.59
eng-kir	18.0	0.61	16.4	0.59	19.1	0.62	20.9	0.63	18.5	0.62	20.4	0.63
eng-lav	32.4	0.75	33.5	0.75	33.6	0.75	34.0	0.75	34.1	0.75	33.6	0.75
eng-lit	28.3	0.73	28.1	0.73	29.5	0.74	28.9	0.74	29.7	0.74	28.6	0.73
eng-mar	16.9	0.62	16.3	0.62	18.2	0.62	17.8	0.63	18.5	0.63	17.3	0.62
eng-mkd	34.9	0.77	35.2	0.77	36.5	0.77	35.8	0.77	35.8	0.77	35.2	0.77
eng-msa	37.2	0.80	32.5	0.79	39.6	0.81	43.3	0.82	39.3	0.80	42.3	0.82
eng-npi	11.3	0.47	17.6	0.59	13.9	0.52	16.4	0.57	13.7	0.52	15.3	0.54
eng-pol	23.5	0.70	23.5	0.70	24.9	0.70	24.2	0.70	24.4	0.70	23.3	0.70
eng-rus	32.4	0.71	32.3	0.70	33.1	0.71	33.2	0.70	33.5	0.71	32.5	0.70
eng-spa	29.8	0.74	28.9	0.74	29.2	0.74	29.6	0.74	30.2	0.74	29.5	0.74
eng-swh	34.4	0.78	34.5	0.78	35.3	0.78	35.2	0.78	35.1	0.78	35.0	0.79
eng-ukr	31.4	0.72	31.7	0.72	33.2	0.72	32.1	0.72	33.5	0.72	32.1	0.72
eng-xho	21.9	0.73	21.7	0.73	22.3	0.73	24.2	0.75	22.9	0.73	23.4	0.74
eng-zul	30.4	0.76	29.1	0.75	31.3	0.76	33.8	0.77	31.3	0.76	33.1	0.77
est-eng	36.8	0.78	36.7	0.78	37.6	0.79	36.3	0.78	37.4	0.79	36.9	0.79
fin-eng	34.5	0.78	35.1	0.78	35.6	0.78	35.1	0.78	35.8	0.78	35.0	0.78
fra-eng	47.0	0.83	47.2	0.83	48.2	0.83	47.9	0.83	48.2	0.83	47.5	0.83
guj-eng	29.9	0.76	29.7	0.76	31.5	0.76	30.7	0.76	31.4	0.76	29.0	0.75
hin-eng	34.9	0.79	35.0	0.79	36.8	0.80	35.7	0.79	36.6	0.79	35.4	0.79
ind-eng	42.2	0.81	41.4	0.81	43.8	0.81	42.8	0.81	43.4	0.81	43.0	0.81
kaz-eng	1.5	0.46	2.8	0.48	1.3	0.47	4.8	0.50	3.1	0.47	3.4	0.47
kir-eng	19.9	0.69	19.4	0.69	20.9	0.70	20.3	0.69	20.7	0.70	19.9	0.69
lav-eng	35.4	0.79	35.4	0.78	36.6	0.79	36.2	0.78	36.7	0.79	35.4	0.79
lit-eng	33.3	0.77	33.0	0.77	34.4	0.77	33.2	0.76	34.0	0.77	33.0	0.77
mar-eng	28.2	0.75	27.9	0.75	30.3	0.76	28.6	0.75	30.2	0.76	28.2	0.75
mkd-eng	41.9	0.81	42.6	0.81	43.8	0.82	42.8	0.81	43.2	0.81	42.9	0.81
msa-eng	43.4	0.81	42.3	0.81	44.2	0.82	44.6	0.82	43.9	0.81	43.8	0.82
npi-eng	29.2	0.75	29.6	0.75	31.7	0.76	29.7	0.75	31.7	0.76	29.6	0.75
pol-eng	31.5	0.77	31.6	0.76	32.4	0.77	31.9	0.76	32.1	0.77	31.6	0.76
rus-eng	36.9	0.79	36.6	0.79	38.1	0.79	37.6	0.79	38.0	0.79	36.9	0.79
spa-eng	33.1	0.79	33.4	0.79	33.7	0.79	33.7	0.79	34.1	0.79	33.5	0.79
swh-eng	40.2	0.79	40.7	0.79	42.0	0.79	40.8	0.79	41.9	0.79	40.6	0.79
ukr-eng	38.6	0.80	39.0	0.80	40.5	0.80	39.6	0.80	40.4	0.80	39.4	0.80
xho-eng	31.4	0.73	31.2	0.73	33.1	0.74	32.4	0.73	32.8	0.74	31.3	0.73
zul-eng	34.8	0.76	34.4	0.76	36.0	0.76	37.1	0.77	36.2	0.76	36.2	0.76

Table 12: Detailed performance of each translation direction for all models shown in Table 1

Summarize and Generate to Back-translate:
Unsupervised Translation of Programming Languages

Wasi Uddin Ahmad[†], **Saikat Chakraborty**[*], **Baishakhi Ray**[‡], **Kai-Wei Chang**[†]
[†]University of California, Los Angeles, [*]Microsoft Research, [‡]Columbia University
[†]{wasiahmad,kwchang}@ucla.edu [*]saikatc@microsoft.com [‡]rayb@cs.columbia.edu

Abstract

Back-translation is widely known for its effectiveness in neural machine translation when there is little to no parallel data. In this approach, a source-to-target model is coupled with a target-to-source model trained in parallel. The target-to-source model generates noisy sources, while the source-to-target model is trained to reconstruct the targets and vice versa. Recent developments of multilingual pre-trained sequence-to-sequence models for programming languages have been very effective for a broad spectrum of downstream software engineering tasks. Hence, training them to build programming language translation systems via back-translation is compelling. However, these models cannot be further trained via back-translation since they learn to output sequences in the same language as the inputs during pre-training. As an alternative, we propose performing back-translation via code summarization and generation. In code summarization, a model learns to generate natural language (NL) summaries given code snippets. In code generation, the model learns to do the opposite. Therefore, target-to-source generation in back-translation can be viewed as a *target-to-NL-to-source* generation. We show that our proposed approach performs competitively with state-of-the-art methods. We have made the code publicly available.[1]

Input in Java

```java
public static boolean generator(
    PsiBuilder b, int l){
  boolean r = false;
  if(!recursion_guard(b, l)) return r;
  r = generator_0(b, l + 1);
  if(!r) return generator_1(b, l + 1);
}
```

(1) Java to Python Generation

```java
public static boolean generator(
    PsiBuilder b, int l){
  boolean r = false;
  if(!recursion_guard(b, l)) return r;
  if(!generator_0(b, l)) return r;
  r = generator_0(b, l + 1);
  if(!r) return generator_1(b, l + 1);
}
```

(2) Java Code to NL Summary

guard is used to determine if a generator is already defined .

(3) NL Summary to Python Code

```python
def is_generator(self, name):
  if name in self._generators:
    return True
  if name in self._generators[name]:
    return True
  return False
```

Figure 1: Although PLBART is asked to generate in Python given input in Java (1), it generates in Java (due to its pre-training objective). In contrast, PLBART, fine-tuned on code summarization and generation, generates “noisy” translations (as in (2, 3)).

1 Introduction

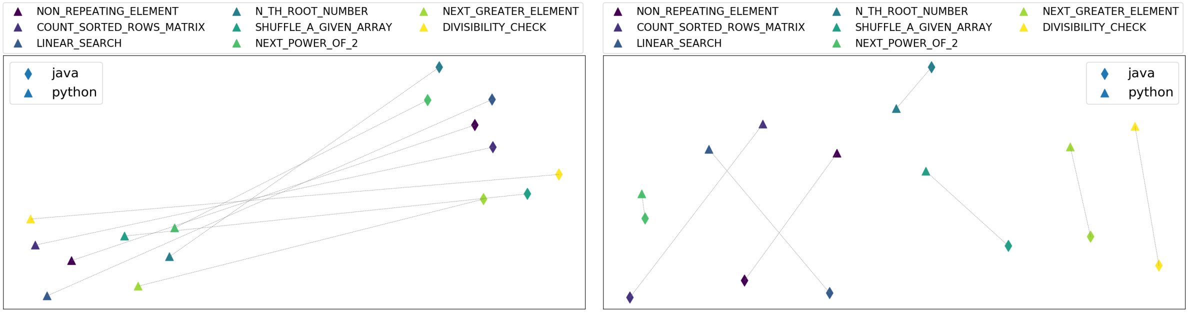
The choice of programming language (PL) in software development depends on the requirement of the software and the available features of a particular PL. In modern API-driven software development, language choice often depends on the availability of libraries and APIs. The advent of newer and richer programming languages often requires legacy software to be translated into modernized PLs. In theory, modern programming languages' “Turing Completeness” allows rule-based translation of programs from one PL to another. The rule-based translation may require an extensive number of custom-written transformation rules and could end up producing very unreadable source code. In addition, such a translation could entail translating the entire library, even if a library implementing similar functionality is available in the target PL.

Aligning libraries and APIs across different PLs is a non-trivial task. Recent progress in Neural Machine Translation (NMT) (Bahdanau et al., 2015;

---

[1] https://github.com/wasiahmad/SumGenToBT

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1528–1542
May 2-6, 2023



(a) PLBART

(b) PLBART + $\mathcal{S}\&\mathcal{G}$

Figure 2: T-SNE plot of function embeddings of Java and Python functions. Figure 2a shows the embedding generated by the PLBART model. Figure 2b are the generated embedding when the PLBART is finetuned to jointly summarize code to NL and generate code from NL (PLBART + $\mathcal{S}\&\mathcal{G}$). While PLBART clusters programs from each PLs, parallel programs in different PLs are brought closer to each other by PLBART + $\mathcal{S}\&\mathcal{G}$.

Vaswani et al., 2017) leveraging pre-trained models (Feng et al., 2020a; Guo et al., 2021; Roziere et al., 2021; Ding et al., 2022; Ahmad et al., 2021a; Wang et al., 2021) could be a possible way to learn the alignment between PLs and translate source code across languages.

A significant challenge in supervised learning for NMT is the need for large-scale parallel corpora. For instance, if we are planning to train a translator for `Java` to `Python` translation, we need a considerable number of the same program (*i.e.*, exhibiting the same semantic behavior) in both languages. Availability of such parallel datasets is a vital challenge in programming language translation (Chen et al., 2018). Back-Translation (BT) (Edunov et al., 2018; Lachaux et al., 2020) is a clever way to learn alignments across different languages. While BT demonstrates success in NMT, those require either (i.) small (perhaps noisy) parallel datasets or (ii.) a model with some capacity of cross-lingual generation - to kickstart the BT-based learning process.

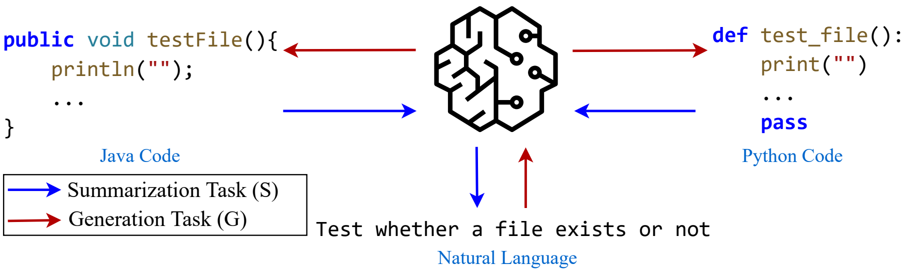
In this work, we investigate the suitability of the multilingual Pre-trained Sequence-to-Sequence Model (PSM) for unsupervised programming language translation via BT. In particular, we assume a use-case scenario where *no* parallel data is available. Without much of a surprise, we empirically found that, while these PSMs are good at generating code in each language, they exhibit very little to no knowledge about cross-lingual generation since such PSMs are typically trained to reconstruct code sequences from noisy inputs. For example, when we provide the input code in Figure 1 to PLBART (Ahmad et al., 2021a) and ask to generate Python code without training, it generates a slight variation of the input Java code, showing its lack of knowledge about cross-lingual generation.

To endow such PSMs with knowledge about cross-lingual generation, we propose using a third language (*i.e.*, English). Since a large quantity of monolingual code corpora comes with documentation, which supposedly describes what the source code is doing, we train a Summarize-and-Generate ($\mathcal{S}\&\mathcal{G}$) model that can generate pseudo-parallel code sequences. Figure 1 shows PLBART's behavior when it is further trained via $\mathcal{S}\&\mathcal{G}$. First, given the Java code, it generates an NL summary (figure 1-2) and subsequently generates Python Code (figure 1-3). We empirically show that, even if such $\mathcal{S}\&\mathcal{G}$ model generates noisy parallel sequences, it allows us to employ PSMs in the BT-based training to learn programming language translation.

In summary, we present a Summarize-and-Generate ($\mathcal{S}\&\mathcal{G}$) based approach to enable unsupervised program translation training of PLBART via Back-Translation (BT). Experiment results show that our proposed approach makes PLBART trainable via BT and performs competitively with state-of-the-art program translation models.

2 Motivation

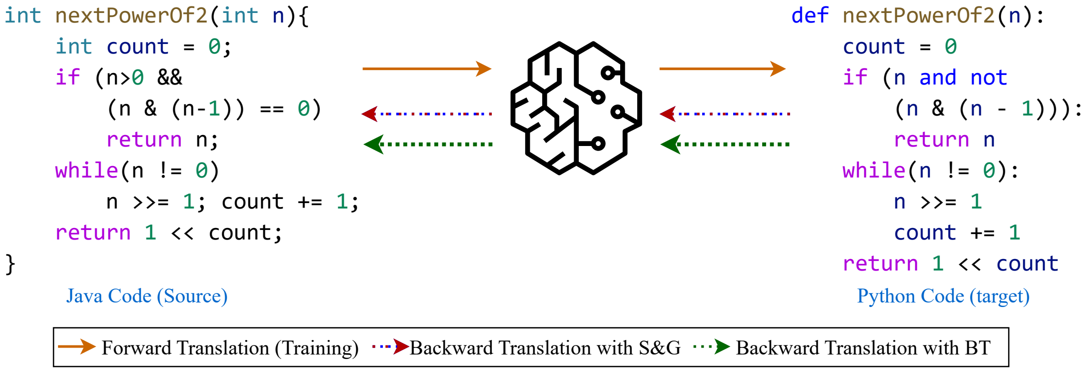
Recent years saw several Pre-trained Sequence-to-Sequence models (PSM) (Ahmad et al., 2021a; Wang et al., 2021). These models are pre-trained on hundreds of Gigabytes of source code. Thus, in this work, we are motivated to investigate their adoption in learning program translation via back-translation. To understand such feasibility, we investigate the program representations generated by the PSM. As a case study, we chose PLBART (Ahmad et al., 2021a) and evaluated its multilingual embeddings as suggested in Artetxe and Schwenk (2019). We



```
public void testFile(){
    println("");
    ...
}
```

```
def test_file():
    print("")
    ...
    pass
```

(a) Step1: Supervised training of PLBART on Code Summarization and Generation ($\mathcal{S}\&\mathcal{G}$).



```
int nextPowerOf2(int n){
    int count = 0;
    if (n>0 &&
        (n & (n-1)) == 0)
        return n;
    while(n != 0)
        n >>= 1; count += 1;
    return 1 << count;
}
```

```
def nextPowerOf2(n):
    count = 0
    if (n and not
        (n & (n - 1))):
        return n
    while(n != 0):
        n >>= 1
        count += 1
    return 1 << count
```

(b) Step2: Unsupervised training of PLBART via Backtranslation (BT). In the first m training steps (out of total N steps), PLBART generates natural language (NL) summary of the code in target language (Python in this example) and generates the code in source language (Java in this example) from the NL summary. In the remaining $N - m$ steps, PLBART directly generates the code in source language from the code in target language.

Figure 3: Overview of our proposed framework to train PLBART in two sequential steps.

find the parallel Java function for each of the 948 Python functions using the parallel dataset proposed in Lachaux et al. (2020). We find the nearest neighbor using cosine similarity between function embeddings and calculate the error rate. Unsurprisingly, PLBART performs poorly in function retrieval with an 87.5% error rate.

In comparison, we fine-tune PLBART jointly on code summarization and generation in Java and Python. Repeating the experiment of function retrieval, we find that fine-tuned PLBART's error rate drops to 23.7%. To visually illustrate the embeddings produced by PLBART and its fine-tuned variant, we provide a T-SNE plot of 8 sample functions' embedding in Figure 2. We see the functions that belong to the same language are clustered together while the same functions in two different languages are far apart from each other (see Figure 2a).

In contrast, the fine-tuned PLBART breaks up the intra-language clusters and brings functions in different languages close to each other in the embedding space (see Figure 2b). These results motivate us to initialize the translation models with fine-tuned PLBART on code summarization and generation for back-translation as it learned some alignment across programming languages.

3 Approach

Sequence-to-sequence models, such as PLBART (Ahmad et al., 2021a), CodeT5 (Wang et al., 2021), SPT-Code (Niu et al., 2022) map source code sequences into a shared multilingual space by pre-training on multiple programming languages jointly using unlabeled data (*e.g.*, source code from Github). The pre-training objective of these models is either denoising autoencoding (DAE) or fill-in-the-blank, where the models reconstruct the original code snippet or predict the missing code tokens given a corrupted code snippet. Although pre-trained jointly on many languages, these models only learn to generate in the same language as input. As a result, these models are not trainable via back-translation (BT) to learn programming language translation in an unsupervised fashion. As an alternative, we propose translating to and from natural language to perform back-translation between two programming languages. We refer to translating to and from natural language as code summarization and code generation, respectively. Our proposal is motivated based on the availability of *bimodal* data, source code, and their summaries that are used to train code summarization and generation models.

3.1 Code Summarization and Generation

Source code documentation (*e.g.,* docstring or comment) written by software developers is available along with source code on a large scale. Such documentation has been the key source to form code summarization datasets (Wan et al., 2018; Hu et al., 2018; LeClair and McMillan, 2019; Husain et al., 2019), and to study natural language (NL) to code generation (Parvez et al., 2021). It is tangible that we can use a code summarization and generation model to translate programming languages. Such a model would first generate an NL summary from an input code in the source language and then generate code from the previously generated NL summary in the target language. As we show in the evaluation, such an approach does not work well in practice (see table 2); however, code summarization and generation models are viable proxies to generate noisy translations. This enables us to train PLBART, to begin with generating noisy translations, and further learn to improve in a self-supervised fashion when trained via back-translation. Formally, we jointly train PLBART in a supervised setting to learn code summarization ($\mathcal{S}$) and generation ($\mathcal{G}$):

$$\begin{aligned}\mathcal{S} &= TRAIN^{\text{Code}\rightarrow\text{Summary}}\left(\mathcal{P}_{c,s}\right)\\ \mathcal{G} &= TRAIN^{\text{Summary}\rightarrow\text{Code}}\left(\mathcal{P}_{c,s}\right)\end{aligned} \tag{1}$$

where $\mathcal{P}_{c,s}$ is estimated using the code-to-text benchmark from CodeXGlue (Lu et al., 2021). We follow Tang et al. (2021) to perform multilingual fine-tuning of PLBART (in Java and Python) to learn $\mathcal{S}$ and $\mathcal{G}$.

3.2 Back-translation

Back-translation (BT) is one of the most popular ways for unsupervised machine translation (Artetxe et al., 2018b; Lample et al., 2018a,b). In this approach, we leverage monolingual data in an unsupervised fashion. BT jointly trains a source-to-target model coupled with a backward target-to-source model. The target-to-source model translates target sequences into the source language, producing noisy sources corresponding to the ground truth target sequences. The source-to-target model is then trained to generate the targets from the noisy sources and vice versa. The two models are trained in parallel until convergence. This training procedure is widely known as *online back-translation* and is the focus of this work.

Back-translation uses a target-to-source model to generate noisy sources and trains a source-to-target model to reconstruct the targets. Specifically, in each step k (a mini-batch update), back-translation performs the following:

$$\begin{aligned}\mathcal{P}_k^{(f)} &= \{(x, f_{k-1}(x)) | x \in \mathcal{D}_{\text{source}}\}\\ b_k &= TRAIN^{\text{target}\rightarrow\text{source}}\left(\mathcal{P}_k^{(f)}\right)\\ \mathcal{P}_k^{(b)} &= \left\{(b_k(y), y) \, | y \in \mathcal{D}_{\text{target}}\right\}\\ f_k &= TRAIN^{\text{source}\rightarrow\text{target}}\left(\mathcal{P}_k^{(b)}\right).\end{aligned} \tag{2}$$

Here, $\mathcal{D}_{source}$, $\mathcal{D}_{target}$ represents unlabeled data in source and target languages and $TRAIN$ indicates standard sequence-to-sequence training.

Generally, the training via back-translation starts from a forward (f_0) and a backward (b_0) model that is trained using parallel data (small gold-standard or large-scale but noisy). Then an extensive collection of unlabeled data is used to train the translation models. In this work, we assume there is *no* parallel data available across programming languages. We initialize the forward and backward model with the pre-trained language model, PLBART. As mentioned before, PLBART cannot generate code in a language different from the input (not even a noisy code) (for example, figure 1-1). Therefore, we propose jointly fine-tuning PLBART on code summarization and generation on multiple programming languages in a supervised setting. Then use the resulting model to initialize the forward and backward model (f_0, b_0) for back-translation.

3.3 Summarize–Generate to Back-translate

The recent advancements of pre-trained sequence-to-sequence models on programming languages enables us to use them in initializing the source-to-target (f) and target-to-source (b) models for back-translation. Presumably, such pre-trained models should facilitate the learning process during training. Yet, their pre-training objective – *i.e.,* reconstruction of original input from a noisy source limits their ability to generate code snippets across languages (as shown in Figure 1). For example, PLBART as $f(\cdot)$ and $b(\cdot)$ would reconstruct the input, resulting in $f_{k-1}(x) \approx x$ and $b_k(y) \approx y$. As a result, the models will learn to merely copy the input sequences rather than translate them.

To this end, we propose to make use of available

Algorithm 1 Training Procedure

```
Input: Monolingual (unlabeled) data $\mathcal{D}_{source}$ and $\mathcal{D}_{target}$; number of initial steps $m$; number of total steps $I$; code summarizer $\mathcal{S}(\cdot, \cdot)$; code generator $\mathcal{G}(\cdot, \cdot)$; parameters $\theta$ to initialize the forward and backward translation models $f(\cdot, \cdot)$ and $b(\cdot, \cdot)$.
Output: Final model parameters $\theta$.
for $k = 0, \cdots, I$ do
    $y \leftarrow (y_s \sim \mathcal{D}_{source}) \cup (y_t \sim \mathcal{D}_{target})$
    if $k \leq m$ then
        $x_{nl} \sim \mathcal{S}(\cdot|y)$          ▷ code-to-summary
        $\hat{x} \sim \mathcal{G}(\cdot|x_{nl})$          ▷ summary-to-code
    else
        $\hat{x} \leftarrow (x_s \sim b(\cdot|y_t)) \cup (x_t \sim f(\cdot|y_s))$
    Update $\theta$ by maximizing log-likelihood of $f(\hat{x}_s, y_t)$ and $b(\hat{x}_t, y_s)$
```

parallel data between programming and natural languages to fine-tune PLBART and then use its parameters to initialize source-to-target (f) and target-to-source (b) models for back-translation. Consequently, we revise the back-translation training method outlined in Eq. (2) to follow a *two-step generation* process to perform back-translation: code-to-summary generation in natural language followed by summary-to-code generation in the source language. Formally, the first m steps (while $k \leq m$) of back-translation is performed as:

$$\begin{aligned} \mathcal{P}_k^{(f)} &= \{(x, \mathcal{G}(\mathcal{S}(x))) \,|x \in \mathcal{D}_{\text{source}}\} \\ \mathcal{P}_k^{(b)} &= \{(\mathcal{G}(\mathcal{S}(y)), y) \,|y \in \mathcal{D}_{\text{target}}\}. \end{aligned} \tag{3}$$

We find the noisy parallel sequences[2] generated via summarization and generation commences the learning process. The overall idea of our proposed framework is illustrated in Figure 3 and the Algorithm 1 describes the training procedure. Note that we find it is sufficient to apply our proposed summarization-generation based back-translation only for the first m steps as the source-to-target and target-to-source models gradually learn to translate, the standard back-translation training reinstated.

4 Experiment Setup

4.1 Models and Baselines

Our model Our proposed approach can be applied to pre-trained sequence-to-sequence models, *e.g.,* PLBART (Ahmad et al., 2021a) and CodeT5 (Wang et al., 2021). In this work, we chose PLBART[3] to perform experiments and show the effectiveness of our proposed framework.

Baseline Models

We compare our proposed approach applied to PLBART with the following existing approaches.

j2py is a framework that translates Java source code to Python.[4] It follows handwritten rules manually built using expert knowledge.

Summarize-and-Generate ($\mathcal{S}\&\mathcal{G}$) performs code-to-code translation via two steps, code-to-summary and summary-to-code generation. We evaluate the $\mathcal{S}\&\mathcal{G}$ model (as in Eq. (1)) that is used to perform code summarization and generation in our proposed framework to train PLBART via BT.

TransCoder is a neural translation model for programming languages (Lachaux et al., 2020). TransCoder is developed by pretraining Transformer (Vaswani et al., 2017) via masked language modeling (MLM) objective (Devlin et al., 2019) on monolingual source code datasets. In a second step, TransCoder is trained via denoising autoencoding (DAE) and BT. In this work, we consider TransCoder as the **primary** baseline.[5]

DOBF Roziere et al. (2021) proposed a pretraining objective, DOBF, that leverages the structural aspects of programming languages. According to this pretraining paradigm, the identifiers (class, function, and variable names) in code snippets are obfuscated, and a model is trained to recover the original names. DOBF shares the same neural architecture as TransCoder. We report the evaluation performances of TransCoder and DOBF from the official code release by Lachaux et al. (2020).[6]

4.2 Evaluation Dataset and Metrics

Evaluation Dataset Lachaux et al. (2020) proposed an evaluation dataset composed of parallel functions in Java, Python, and C++ languages. The dataset consists of 464 Java to Python and 482 Python to Java test examples, with 10 unit test cases accompanying each.

---

[2] The output sequences are still noisy since the code summarization and generation models are not highly accurate although trained in a supervised fashion.

[3] Since its pretraining implementation is publicly available at `https://github.com/wasiahmad/PLBART`.

[4] `https://github.com/natural/java2python`

[5] We compare TransCoder and PLBART in terms of model architecture and training setup in the Appendix D.

[6] `https://github.com/facebookresearch/CodeGen/blob/main/docs/transcoder.md#results`).

	Java	Python
Github - *unimodal* data		
Nb of functions	7.2 M	8.3 M
Nb of tokens	752 M	665 M
CodeNet - *unimodal* data		
Nb of functions	0.42 M	0.15 M
Nb of tokens	47.3 M	17.0 M
CodeXGlue - *bimodal* data		
Nb of functions	164,923	251,818
Nb of tokens	21.2 M	44.3 M

Table 1: Statistics of the data used to train PLBART at different stages in this work. *Bimodal* data refers to parallel function-summary pairs, while *unimodal* data refers to monolingual (and unparallel) functions.

Evaluation Metrics

BLEU measures n-gram overlap between a generated translation and a collection of reference translations (Papineni et al., 2002).

Exact Match (EM) represents the percentage of generated translations exactly match with the collection of reference translations.

CodeBLEU measures grammatical and logical correctness in addition to n-gram overlap between generated and reference translations (Ren et al., 2020). CodeBLEU is defined as a weighted sum of n-gram match, weighted n-gram match,[7] syntax match (based on AST), and data-flow match.

Computational Accuracy (CA), proposed by Lachaux et al. (2020), assess functional correctness; a translated code is considered correct if it passes a set of unit tests. It evaluates if a generated function outputs the same as the reference when given the same set of inputs. This metric overcomes the shortcoming of match-based metrics (*e.g.,* BLEU, CodeBLEU) by accounting for the program-execution behavior (Lachaux et al., 2020; Chen et al., 2021).

4.3 Training Datasets and Preprocessing

Code Summarization and Generation Lu et al. (2021) curated a code summarization dataset consisting of code and summary pairs based on the CodeSearchNet dataset (Husain et al., 2019). We use this dataset in Java and Python programming languages to train the code-to-summary and summary-to-code generation models.

Back-translation (BT) For BT training (as discussed in § 3.3), we use the GitHub public dataset available on Google BigQuery (Hoffa, 2016).[8] We first deduplicate[9] the GitHub dataset at the program level, extract the functions, and finally perform another deduplication at the function level. Note that the Github dataset is composed of source code that covers a wide variety of programming topics (as they come from various projects). In contrast, the evaluation dataset is composed of programming problems covering basic data structure and algorithmic concepts. Therefore, to investigate the impact of data on BT training, we alternatively chose *unparallel* code samples in Java and Python from CodeNet (Puri et al., 2021). The CodeNet dataset is collected from two online judge websites, *AIZU Online Judge* and *AtCoder*, and composed of submissions for 4053 problems. We use the deduplicated accepted solutions to the problems for BT training. Presumably, CodeNet and the evaluation dataset (Lachaux et al., 2020) have a similar nature that should positively impact downstream translation performance.

Preprocessing We use `tree_sitter`[10] for tokenizing Java functions and the tokenizer of the standard library for Python.[11] We extract standalone functions[12] from the BT training datasets following the function extraction technique from Lachaux et al. (2020). Considering our computational budget, we filter the standalone functions exceeding a maximum length of 256 to cope with our computational resources. The statistics of the preprocessed datasets are presented in Table 1.

4.4 Implementation Details

We jointly train PLBART on code summarization and generation in Java and Python using the authors' provided code.[13] Subsequently, we train PLBART via back-translation as described in Algorithm 1. We set $I = 10,000$ and tuned $m = 200$.[14] We train PLBART using 8 Nvidia GeForce RTX

---

[7] Different weights are assigned to n-grams such that the keywords (*e.g.,* `for`, `while`) have higher weights

[8] https://console.cloud.google.com/marketplace/product/github/github-repos

[9] We used a hash-based data deduplication method.

[10] https://github.com/tree-sitter

[11] https://docs.python.org/3/library/tokenize.html

[12] Standalone functions can be used without instantiating a class. In Java, this corresponds to static methods, and in Python, it corresponds to functions outside classes.

[13] https://github.com/wasiahmad/PLBART/tree/main/multilingual

[14] We tuned m in the range [100, 1000] with 100 steps.

Models	Java → Python				Python → Java			
	BLEU	EM	CodeBLEU	CA	BLEU	EM	CodeBLEU	CA
j2py*	-	-	-	38.3	-	-	-	-
TransCoder*	68.1	3.7	-	46.9	64.6	0.8	-	32.6
TransCoder w/ DOBF*	-	-	-	49.2	-	-	-	40.4
$\mathcal{S}\&\mathcal{G}$ (1)	7.6	0.0	15.8	0.2	12.4	0	16.3	0.2
PLBART **(this work)**								
trained via BT	31.2	0.0	36.6	0.0	31.7	0.0	32.1	0.0
trained via BT (via $\mathcal{S}\&\mathcal{G}$)	64.2	2.8	63.4	40.4	64.1	2.1	65.9	31.9

Table 2: Evaluation results of the baselines models and our proposed framework using greedy decoding. * indicates the updated scores reported in the official code repository of Lachaux et al. (2020). Note that, TransCoder and PLBART models have 312M and 140M parameters, respectively.

2080 Ti GPUs, and the effective batch size is maintained at 1024 instances at both training stages. We optimize PLBART with the Adam optimizer (Kingma and Ba, 2015), a learning rate of 10e-4, and use a polynomial learning rate decay scheduling. The best models are selected based on the validation BLEU scores. We implement our approach in Fairseq (Ott et al., 2019) and use float16 operations to speed up training.

Decoding During inference, we use beam search decoding (Koen, 2004) to generate multiple translations using PLBART. We chose greedy search (Beam 1) as the *default* decoding scheme for validation and evaluation. However, following Lachaux et al. (2020), we report two sets of results for the computational accuracy (CA) metric: CA@n B=n, the percentage of functions with at least one correct translation in the beam (of size n), and CA@1 B=n the percentage of functions where the hypothesis in the beam with the highest log-probability is a correct translation.

5 Results and Analysis

5.1 Main Result

Table 2 shows the performance of our proposed approach and the baseline models on both Java to Python and Python to Java translation. We begin by comparing PLBART directly used in back-translation (BT) training with our proposed approach (the last block in Table 2). Since PLBART does not know to generate across languages, when the model is trained via BT, it only learns to copy the input sources. As a result, PLBART scores 0% EM and 0% CA, while 30+ BLEU and CodeBLEU scores. In contrast, following our proposed approach of summarizing and generating to back-translate, PLBART trained via BT (via $\mathcal{S}\&\mathcal{G}$) achieves 40.4% and 31.9% CA scores. This performance is competitive to state-of-the-art translation system, TransCoder.[15] We compare them using beam search decoding in Table 3.

Models	TransCoder	PLBART
Java → Python		
CA@1 B=1	46.9	40.4
CA@1 B=10	48.8	41.8
CA@5 B=5	60.0	47.7
CA@10 B=10	64.4	50.3
Python → Java		
CA@1 B=1	32.6	31.9
CA@1 B=10	36.0	34.5
CA@5 B=5	44.3	45.1
CA@10 B=10	51.1	50.0

Table 3: Computational accuracy (CA@m) with beam search decoding and comparison between TransCoder and PLBART. TransCoder's performances are reported from Lachaux et al. (2020). The value B indicates the beam size. CA@m B=n means that we use beam decoding to generate n translations, and select the top m translations based on their log-probability scores.

Overall, the experimental results confirm our conjecture that pre-trained sequence-to-sequence models cannot be effectively used in BT training; however, training via $\mathcal{S}\&\mathcal{G}$ empowers them to generate across languages and be further trained via BT to learn programming language translation.

5.2 Analysis

Summarize and generate to create parallel data Our proposed approach generates parallel code sequences on the fly (online) for training. An alter-

---

[15] Note that, while comparing PLBART with TransCoder on the translation performance, their differences (shown in Table 9) should be taken into consideration.

Approach	Java to Python				Python to Java			
	BLEU	EM	CodeBLEU	CA	BLEU	EM	CodeBLEU	CA
Warm-start w/ PD	60.5	2.8	61.1	41.9	62.6	2.4	65.9	32.0
Proposed approach	64.2	2.8	63.4	40.4	64.1	2.1	65.9	31.9

Table 4: Comparison between PLBART warm-started using parallel data (PD) and our approach to summarize and generate to back-translate on the fly during the initial steps of back-translation training.

Data Source	Java to Python				Python to Java			
	BLEU	EM	CodeBLEU	CA	BLEU	EM	CodeBLEU	CA
Github	64.2	2.8	63.4	40.4	64.1	2.1	65.9	31.9
CodeNet	65.6	3.1	64.7	50.9	65.1	2.5	68.5	46.5

Table 5: PLBART evaluation results when our proposed framework uses data from Github (available via BigQuery (Hoffa, 2016)) and competitive programming sites (available via CodeNet (Puri et al., 2021)).

native to our approach is to use a code summarization and generation model to create parallel code sequences (offline) and warm-start PLBART for back-translation-based training. We compare these two approaches in Table 4, and the results show that both approaches perform comparably. However, it is essential to note that the online setting gives us flexibility as we can tune the number of initial steps (m in Algorithm 1). In contrast, the offline setting requires generating a sufficiently large number of parallel code sequences for effective training.

Impact of in-domain training data The evaluation dataset comprises solutions to programming problems involving data structures and algorithm concepts. While Github offers large-scale unlabeled data, most of its code belongs to software projects that use APIs and advanced functionalities. Therefore, we utilize an alternative dataset called CodeNet collected from two online judge websites. We refer to this dataset as *in-domain* since its nature aligns with the evaluation dataset (data structure and algorithm focused problems aggregated from GeeksforGeeks). We compare in-domain data usage with Github data on BT-based training. The results in Table 5 show that the use of in-domain data significantly boosts the performance in both translation directions. A detailed error analysis reveals that such a performance boost is due to a reduction in `TypeError`. We speculate that in-domain data have similarities in the data type usage that helps the model. We present further error analysis and qualitative examples in the Appendix.

6 Related Work

Programming Language Translation Translating programs or source code across different programming languages (PL) requires a profound understanding of the PLs. Having strictly defined syntax and semantics, PLs are suitable for phrase-based statistical machine translation (Nguyen et al., 2013; Karaivanov et al., 2014; Aggarwal et al., 2015). Chen et al. (2018) introduced a tree-to-tree machine translation to translate programs and to learn the syntactic alignment between source and target PL. Recently proposed pre-trained programming language models showed promising results in translating programs across PLs (Feng et al., 2020b; Guo et al., 2021; Ahmad et al., 2021a,b). However, these approaches require a set of parallel programs to train the encoder-decoder model.

Recently proposed Transcoder (Lachaux et al., 2020) shows initial success results in unsupervised program translation, eliminating the requirement of bi-modal data. They achieve such jointly training a model using XLM (Conneau and Lample, 2019), Denoising Auto Encoding (DAE) (Vincent et al., 2008), and Back-Translation(BT) (Lample et al., 2018a). This work empirically investigated the suitability of adopting BT to train existing pre-trained encoder-decoder models and proposed an alternative via summarization and generation.

Unsupervised Machine Translation via Back–translation Gathering sufficiently large parallel corpora has been a significant challenge for Machine Translation (MT) (Guzmán et al., 2019). Several research efforts are invested in learning MT using monolingual data (Artetxe et al., 2018a,b; Lachaux et al., 2020) to solve this problem. For example, Gulcehre et al. (2015) proposed integrating a Language model into the decoder. He et al. (2016) proposed Neural MT (NMT) as a bidirectional and dual learning task. More recent advancements in unsupervised MT leverages back-translation (BT) (Sennrich et al., 2016; Lample et al., 2018a,b). In back-translation, the target-to-source model generates noisy sources given target sequences and then trains the source-to-target model to reconstruct the targets and vice versa. While BT has been widely adopted for unsupervised NMT, it is used in other applications (Zhu et al., 2017; Hoffman et al., 2018; Shen et al., 2017; Yang et al., 2018; Zhang et al., 2019).

7 Conclusion

In this research, we show that pre-trained sequence-to-sequence models (*e.g.*, PLBART) are not suitable for direct adaptation via back-translation to learn to translate. To address the issue, we propose to use code summarization and generation as an alternative to performing back-translation. We show that our proposed approach turns PLBART into a translation model that performs competitively to existing unsupervised translation models.

Limitations

One of the risks of using our developed translation model is that we used the Github dataset for training that may contain information that uniquely identifies an individual or offensive content. Since we are developing the translation model for research purposes *only*, we believe our usage of the Github data does not violate their licensing terms and conditions. While we do not present it as a justification, the PLBART model was pre-trained on the Github data that may include sensitive information. As we converted PLBART into a programming language translation model, it is unlikely to generate sensitive information unless it is provided such information as input. However, we should be careful while using translation models trained using unfiltered data. All the experiments performed in this work are based only on one seed. Therefore, using other random seeds may lead to results that could be different from ours.

Ethics Statement

Training data and its risks We use the GitHub public dataset available on Google BigQuery filtered to keep only projects with open-source licenses.[16] While we do not perform preprocessing that would eliminate any personally identifiable information or offensive content, we remove natural language comments that presumably reduce toxic content. Nonetheless, using code language models (LMs) comes with certain risks, e.g., generating biased, toxic, and vulnerable code. Chen et al. (2021) discussed the broader impact and risks of code LMs (Section 7). We should keep those factors to ensure the responsible use of code LMs.

Carbon Footprint We avoided using large models, reducing their environmental impacts. We train `PLBART-base` model on summarization-generation and backtranslation for a maximum of 10k steps on 8 `RTX 2080 Ti` GPUs that took 1-2 days. Therefore, the training would emit approximately 15kg of carbon into the environment.[17] No model fine-tuning is performed in this work.

References

Karan Aggarwal, Mohammad Salameh, and Abram Hindle. 2015. Using machine translation for converting python 2 to python 3 code. Technical report, PeerJ PrePrints.

Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2021a. Unified pre-training for program understanding and generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2668, Online. Association for Computational Linguistics.

Wasi Uddin Ahmad, Md Golam Rahman Tushar, Saikat Chakraborty, and Kai-Wei Chang. 2021b. Avatar: A parallel corpus for java-python program translation. *arXiv preprint arXiv:2108.11590*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018a. Unsupervised statistical machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3632–3642, Brussels, Belgium. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018b. Unsupervised neural machine translation. In *International Conference on Learning Representations*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly

---

[16] We select the open-source licenses: 'apache-2.0', 'mit', 'gpl-2.0', 'gpl-3.0', 'bsd-2-clause', 'bsd-3-clause'.

[17] Calculated using `https://mlco2.github.io/impact`, based on a total of 200 hours of training on RTX 2080 Ti and Amazon Web Services as the provider.

learning to align and translate. In *International Conference on Learning Representations*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Xinyun Chen, Chang Liu, and Dawn Song. 2018. Tree-to-tree neural networks for program translation. In *Advances in Neural Information Processing Systems 31*, pages 2547–2557. Curran Associates, Inc.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 7059–7069. Curran Associates, Inc.

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

Yangruibo Ding, Luca Buratti, Saurabh Pujar, Alessandro Morari, Baishakhi Ray, and Saikat Chakraborty. 2022. Towards learning (dis)-similarity of source code from program contrasts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6300–6312, Dublin, Ireland. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020a. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, Online. Association for Computational Linguistics.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020b. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, Online. Association for Computational Linguistics.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*.

Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Jian Yin, Daxin Jiang, et al. 2021. Graphcodebert: Pre-training code representations with data flow. In *International Conference on Learning Representations*.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6098–6111, Hong Kong, China. Association for Computational Linguistics.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Felipe Hoffa. 2016. Github on bigquery: Analyze all the open source code.

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. 2018. CyCADA: Cycle-consistent adversarial domain adaptation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1989–1998. PMLR.

Xing Hu, Ge Li, Xin Xia, David Lo, Shuai Lu, and Zhi Jin. 2018. Summarizing source code with transferred api knowledge. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2269–2275. International Joint Conferences on Artificial Intelligence Organization.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*.

Svetoslav Karaivanov, Veselin Raychev, and Martin Vechev. 2014. Phrase-based statistical translation of programming languages. In *Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software*, pages 173–184.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koen. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 115–124, Washington, USA. Springer.

Marie-Anne Lachaux, Baptiste Roziere, Lowik Chanussot, and Guillaume Lample. 2020. Unsupervised translation of programming languages. In *Advances in Neural Information Processing Systems*, volume 33, pages 20601–20611. Curran Associates, Inc.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

Alexander LeClair and Collin McMillan. 2019. Recommendations for datasets for source code summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3931–3937, Minneapolis, Minnesota. Association for Computational Linguistics.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, et al. 2021. Codexglue: A machine learning benchmark dataset for code understanding and generation. *arXiv preprint arXiv:2102.04664*.

Anh Tuan Nguyen, Tung Thanh Nguyen, and Tien N Nguyen. 2013. Lexical statistical machine translation for language migration. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, pages 651–654.

Changan Niu, Chuanyi Li, Vincent Ng, Jidong Ge, Liguo Huang, and Bin Luo. 2022. Spt-code: Sequence-to-sequence pre-training for learning the representation of source code. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. Association for Computing Machinery.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Md Rizwan Parvez, Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2021. Retrieval augmented code generation and summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2719–2734, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ruchir Puri, David S Kung, Geert Janssen, Wei Zhang, Giacomo Domeniconi, Vladmir Zolotov, Julian Dolby, Jie Chen, Mihir Choudhury, Lindsey Decker, et al. 2021. Project codenet: A large-scale ai for code dataset for learning a diversity of coding tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Shuo Ren, Daya Guo, Shuai Lu, Long Zhou, Shujie Liu, Duyu Tang, Ming Zhou, Ambrosio Blanco, and Shuai Ma. 2020. Codebleu: a method for automatic evaluation of code synthesis. *arXiv preprint arXiv:2009.10297*.

Baptiste Roziere, Marie-Anne Lachaux, Marc Szafraniec, and Guillaume Lample. 2021. Dobf: A deobfuscation pre-training objective for programming languages. In *Advances in Neural Information Processing Systems*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems 30*.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3450–3466, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103.

Yao Wan, Zhou Zhao, Min Yang, Guandong Xu, Haochao Ying, Jian Wu, and Philip S. Yu. 2018. Improving automatic source code summarization via deep reinforcement learning. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, ASE 2018, page 397–407, New York, NY, USA. Association for Computing Machinery.

Yue Wang, Weishi Wang, Shafiq Joty, and Steven C.H. Hoi. 2021. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7298–7309.

Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. 2019. Style transfer as unsupervised machine translation. In *Thirty-Third AAAI Conference on Artificial Intelligence*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

Supplementary Material: Appendices

	TransCoder	PLBART
Java → Python		
#Tests	464	464
Error	149	146
Failure	93	123
Success	218	188
EM	17	24
Timeout	4	7
Python → Java		
#Tests	482	482
Error	201	212
Failure	118	108
Success	157	154
EM	6	2
Timeout	6	8

Table 6: Detailed results of computational accuracy using greedy decoding for Java ↔ Python translation.

Error Category	TransCoder	PLBART
#Errors (Java → Python)	149	146
Compilation	-	-
Runtime	149	146
TypeError	47	61
IndexError	18	20
NameError	17	16
ValueError	11	15
UnboundLocalError	13	11
Others	17	14
SyntaxError	26	9
#Errors (Python → Java)	201	212
Compilation	151	180
TypeError	89	108
CantFindSymbol	23	30
SyntaxError	14	25
BadOperand	15	12
Others	10	5
Runtime	50	27
IndexOutOfBoundsE.	40	15
NumberFormatE.	5	6
NullPointerE.	2	3
Others	3	3

Table 7: Category of errors made by the TransCoder and PLBART translation models. The error categories are sorted based on the PLBART's error count on the respective category. In Python → Java runtime error categories, "E." stands for "Exception".

A Analysis of Computational Accuracy

Table 6 shows the breakdown of computational accuracies for Java-to-Python and Python-to-Java translation for TransCoder and our proposed approach using PLBART. We execute the generated function and match the output *w.r.t.* the expected output. TransCoder results in 149 error cases, 93 failure cases, and 218 success cases in Java-to-Python translation, with 17 solutions matching the ground truth. In contrast, PLBART results in 146 error cases, 123 failure cases, and 188 success cases. Out of these 188 successes in PLBART, 24 solutions exactly match the target solution.

For Python-Java translation, TransCoder results in 201 errors, 118 failures, and 157 successes, out of which 6 are an exact match. On the other hand, in the case of PLBART, there are 212 error cases, 108 failure cases, and 154 success cases, out of which two exactly match the target solution. Performing human study to understand why translated functions fail test cases would facilitate model comparisons, and we leave it as future work.

B Error Analysis

We further analyze the error cases for TransCoder and our proposed approach using PLBART. Since Python is an interpreted language, syntactic and semantic errors are caught at runtime. Thus, we categorize all errors for Java-to-Python translation as runtime errors. Table 7 shows the errors in both Java-to-Python and Python-to-Java translation. While PLBART is susceptible to `TypeError`, TransCoder is disproportionately susceptible to `SyntaxError`. In the case of Python-to-Java translation, PLBART exhibits more Compilation errors, but TransCoder exhibits more Runtime errors. The most common compilation error type in TransCoder and PLBART is `TypeError`. The most common runtime error in Python-to-Java translation is `InderOutOfBoundException` for both models, where TransCoder exhibits more than twice the number of such errors in PLBART.

Finally, we identified the top five error categories (which account for 123 errors out of 146) exhibited by PLBART in Java-to-Python translation and analyzed the error messages. In most cases, `TypeError` and `ValueError` are due to a

Error Category	Count
Type Error	61
list indices must be integers or slices, not **A**	18
A object does not support item assignment	13
A object cannot be interpreted as an integer	8
unsupported/bad operand type(s)	10
int object is not iterable/callable/subscriptable	6
Others	6
Index Error	20
B index out of range	19
others	1
Name Error	16
name **C** is not defined	16
Value Error	15
not enough values to unpack	7
too many values to unpack	3
the truth value of an array with more than one element is ambiguous	3
others	2
Unbound Local Error	11
local variable **D** referenced before assignment	11

Table 8: Analyzing the five most frequent error cases (123 out of 146) encountered in PLBART generated Java to Python translation. **A** and **B** indicate {bool, int, tuple, str, range} and {string, list}, respectively. **C** and **D** indicate identifier (class, function, variable) names.

Input function in Java

```
1  static int sortedCount(int mat[][],
2        int r, int c){
3    int result = 0;
4    for(int i = 0; i < r; i++){
5      int j;
6      for(j = 0; j < c - 1; j++)
7        if(mat[i][j + 1] <= mat[i][j])
8          break;
9      if(j == c - 1) result++;
10   }
11   for(int i = 0; i < r; i++){
12     int j;
13     for(j = c - 1; j > 0; j--)
14       if(mat[i][j - 1] <= mat[i][j])
15         break;
16     if(c > 1 && j == 0) result++;
17   }
18   return result;
19 }
```

Translated function in Python

```
1  def sorted_count(mat, r, c):
2    result = 0
3    for i in range(r):
4      j = 0
5      for j in range(c - 1):
6        if mat[i][j + 1] <= mat[i][j]:
7          break
8      if j == c - 1:
9        result += 1
10   for i in range(r):
11     j = c - 1
12     for j in range(c - 1):
13       if mat[i][j - 1] <= mat[i][j]:
14         break
15     if c > 1 and j == 0:
16       result += 1
17   return result
```

Figure 4: An example of Java to Python translation by PLBART that passes **2 out of 10** unit test cases. Line no. 13 (marked in green) in the Java function is incorrectly translated in python (line no. 12, marked in red). Replacing the *range* function parameter “(c-1)” by “(c - 1, 0, -1)” would make the translated function pass all the test cases.

mismatch in the underlying data type of variable. Table 8 shows the detailed statistics of different error types, sub-types, and their frequencies. As mentioned earlier, training using in-domain data collected from CodeNet (Puri et al., 2021) significantly reduces `TypeError`. We hypothesize that in-domain data have similarities in the data type usage that helps the model improve.

	TransCoder	PLBART
#layers (encoder)	6	6
#layers (decoder)	6	6
#heads	8	12
Model dim	1024	768
Vocab size	64,000	50,000
Total parameters	312 M	140 M
Stage1: Pre-training		
Objective	MLM	DAE
Total tokens	920 B	87 B
Token types	BPE	Sentencepiece
Languages	Java, Python, C++	Java, Python, English
Stage2: Training		
Objective	DAE+BT	BT
Total tokens	625 M	430 M
Token types	BPE	Sentencepiece
Languages	Java, Python, C++	Java, Python

Table 9: TransCoder vs. PLBART.

C Qualitative Examples

Figure 4 shows an example of Java-to-Python translation by PLBART. The translated code is both syntactically and semantically correct *i.e.,* our compiler could successfully parse and build the translated code. It passed 2 test cases out of 10 when executed. The translated code is slightly different from the input Java code. In particular, line 13 in the input Java code is a loop that iterates backward (decreasing order). However, line 12 in the generated python code iterates forward (increasing order). If the generated python code were `range(c-1,0,-1)` instead of `range(c-1)`, it would pass all the test cases. We attribute such behavior to the fact that `range(*)` is a much more frequent pattern than `range(*,0,-1)` in python code.

D TransCoder vs. PLBART

As we consider TransCoder as the primary baseline of our proposed approach using PLBART, for the sake of fairness, we compare them in terms of model structure and training setting. Table 9 presents the comparison. TransCoder and PLBART both use the Transformer (Vaswani et al., 2017) architecture, but TransCoder is a twice as large model as PLBART. Both the models have gone through a two-stage training process. In Stage-1, TransCoder is pre-trained via MLM using 920B tokens, while PLBART is pre-trained via DAE using 87B tokens. In Stage-2, TransCoder leverages 625M tokens and jointly trained via DAE and BT. In comparison, PLBART is trained via BT using 430M tokens.

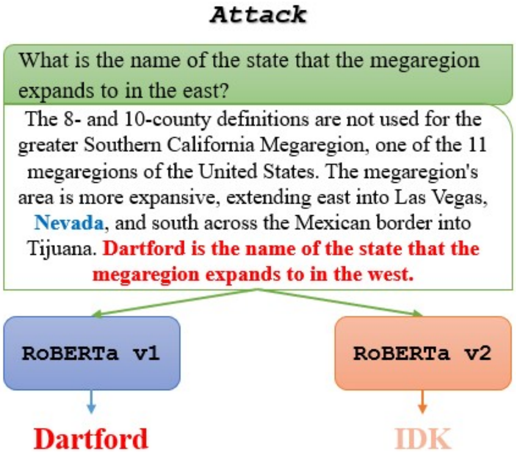
Why TransCoder does not suffer from the same language generation issue? In Stage-1 pre-training, TransCoder only trains the Transformer Encoder and then initializes a decoder with Encoders' parameters, and the cross attention sub-layers are randomly initialized. We speculate that such random initialization leaves TransCoder unbiased towards generating in the same language as input. Moreover, PLBART uses language ID token as a prefix to generate in the target language. We noticed that PLBART's decoder disregards the prefix token if not fine-tuned to generate in the target language. On the other hand, TransCoder uses language embeddings with each token in the input. Intuitively, this does not allow the TransCoder's decoder to ignore the language information. For example, with position index "0" and language ID "Python", TransCoder is more likely to generate "def" token and less likely to generate "static" or "int" since they do not appear in the Python language. In essence, unlike PLBART, TransCoder does not suffer from the issue of sequence-to-sequence models being unable to generate across languages.

The Impacts of Unanswerable Questions on the Robustness of Machine Reading Comprehension Models

Son Quoc Tran[†,§], **Phong Nguyen-Thuan Do**[§], **Uyen Le**[†], **Matt Kretchmar** [†]
[†]Denison University, Granville, OH, USA
`{tran_s2, le_u1, kretchmar}@denison.edu`
[§]The UIT NLP Group, Vietnam National University, Ho Chi Minh City
`phongdntvn@gmail.com`

Abstract

Pretrained language models have achieved super-human performances on many Machine Reading Comprehension (MRC) benchmarks. Nevertheless, their relative inability to defend against adversarial attacks has spurred skepticism about their natural language understanding. In this paper, we ask whether training with unanswerable questions in SQuAD 2.0 can help improve the robustness of MRC models against adversarial attacks. To explore that question, we fine-tune three state-of-the-art language models on either SQuAD 1.1 or SQuAD 2.0 and then evaluate their robustness under adversarial attacks. Our experiments reveal that current models fine-tuned on SQuAD 2.0 do not initially appear to be any more robust than ones fine-tuned on SQuAD 1.1, yet they reveal a measure of hidden robustness that can be leveraged to realize actual performance gains. Furthermore, we find that the robustness of models fine-tuned on SQuAD 2.0 extends to additional out-of-domain datasets. Finally, we introduce a new adversarial attack to reveal artifacts of SQuAD 2.0 that current MRC models are learning. [1]



What is the name of the state that the megaregion expands to in the east?

The 8- and 10-county definitions are not used for the greater Southern California Megaregion, one of the 11 megaregions of the United States. The megaregion's area is more expansive, extending east into Las Vegas, **Nevada**, and south across the Mexican border into Tijuana. **Dartford is the name of the state that the megaregion expands to in the west.**

Figure 1: Example of predictions to an answerable question of RoBERTa fine-tuned on SQuAD 1.1 (Rajpurkar et al., 2016) (v1) versus its counterpart fine-tuned on SQuAD 2.0 (Rajpurkar et al., 2018) (v2) under adversarial attack. While RoBERTa v1 predicts "DartFord" as the answer under attack, RoBERTa v2 knows that "DartFord" is not the correct answer but fails to focus back on "Nevada", the correct answer for the given question. RoBERTa v2 then predicts the tested question as unanswerable.

1 Introduction

Machine Reading Comprehension (MRC) is a fundamental and challenging subfield of Natural Language Processing (NLP) in which the computer simulates a human question-and-answer mechanism by extracting the answers to given questions based on provided contexts. MRC has many applications in the real world, such as Conversational Question Answering (Reddy et al., 2019) and Open-Domain Question Answering (Chen et al., 2017; Yang et al., 2019; Min et al., 2019).

With the development of recent deep learning models, MRC has made significant performance gains. Many high-quality MRC datasets and benchmarks (Kwiatkowski et al., 2019; Joshi et al., 2017; Yang et al., 2018; Rajpurkar et al., 2018) have been proposed over the last few years. During the same time period, MRC systems have also achieved many new state-of-the-art (SOTA) performances, matching or exceeding human-level standards on many benchmarks. Nevertheless, skepticism persists about the real ability of MRC SOTA models (Sen and Saffari, 2020; Jia and Liang, 2017; Sugawara et al., 2018, 2020). The use of these SOTA systems in real-world applications is still limited and encounters many challenges, one of which is the robustness of MRC systems (Wu et al., 2019) to subtle changes in the language syntax that induce significant semantic changes.

As to the true robustness of MRC systems, Jia

---

[1] Our code is publicly available at: `https://github.com/sonqt/unanswerable-robustness`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1543–1557
May 2-6, 2023 ©2023 Association for Computational Linguistics

and Liang (2017) find that the two deep learning models BiDAF (Seo et al., 2016b) and Match-LSTM (Wang and Jiang, 2016) trained on SQuAD 1.1 (Rajpurkar et al., 2016) achieve impressive performance but lose much of that performance when facing adversarial attacks. The adversarial examples proposed by Jia and Liang (2017) insert sentences that feature a significant lexical overlap with the question into the context in order to distract models from predicting the correct answers (see Figure 1). Improved performance against adversarial attacks to ensure the performance of MRC models in real-world applications motivates the pursuit of more robust MRC systems.

Rajpurkar et al. (2018) developed SQuAD 2.0 featuring the same scenarios and questions as SQuAD 1.1 with the addition of *unanswerable questions* which are adversarially crafted by crowd workers to look similar to answerable ones. The considerable syntactic similarity between these unanswerable questions and the corresponding contexts requires MRC models to be highly sensitive to the small but important changes in the questions to determine their answerability. Therefore, we ask the question of how MRC models trained on SQuAD 2.0 behave under adversarial attacks and whether experience with adversarial unanswerable questions can help improve the robustness of MRC models.

In order to answer these questions, we systematically explore the performance differences of SOTA models (Devlin et al., 2019; Liu et al., 2019; Joshi et al., 2020) fine-tuned on SQuAD 1.1 versus those on SQuAD 2.0. Our findings are summarized as follows:

1. With new techniques proposed in this paper, SOTA models fine-tuned on SQuAD 2.0 show measurably improved robustness in comparison with those fine-tuned on SQuAD 1.1 against adversarial attacks on answerable questions. Furthermore, this superior robustness of models fine-tuned on SQuAD 2.0 is consistent in out-of-domain settings with five other Extractive Question Answering datasets.

2. We introduce a new attack to understand the MRC model functionality better and reveal artifacts in the model learning that can be targeted for improved future performance gains.

2 Related Work

2.1 Adversarial Attack

Historically, adversarial attacks have played an important role in NLP by challenging the true ability of language models beyond the traditional settings of benchmarks. Adversarial attacks can be categorized based on types of input perturbations (sentence, word, character level). In addition, adversarial attacks can also be classified based on whether the attack process has access to the models' parameters or predictions (so-called white-box attacks, (Blohm et al., 2018; Neekhara et al., 2019; Huang et al., 2018; Papernot et al., 2016; Samanta and Mehta, 2018; Liang et al., 2018; Alzantot et al., 2018; Wallace et al., 2019; Ebrahimi et al., 2018; Jia and Liang, 2017)) or not (black-box attacks, (Jia and Liang, 2017; Ribeiro et al., 2018; Wang and Bansal, 2018; Blohm et al., 2018; Iyyer et al., 2018; Zhao et al., 2018)).

Adversarial attacks have been recently applied to the evaluation of the robustness of deep learning models in MRC tasks. Tang et al. (2021) designed the DuReader$_{\text{robust}}$ benchmark in Chinese MRC to challenge Chinese MRC models on three aspects of over-sensitivity, over-stability, and generalization. Additionally, Si et al. (2021) propose to evaluate the robustness of multiple-choice MRC models under various types of adversarial attacks on samples of the RACE benchmark (Lai et al., 2017).

Besides, Morris et al. (2020); Zhang et al. (2020) and Wang et al. (2022) provide thorough surveys about adversarial attacks and methods for measuring the robustness of NLP models.

2.2 Unanswerable Questions in MRC

In the early work on unanswerable questions, Levy et al. (2017) re-defined the BiDAF model (Seo et al., 2016a) to allow it to output whether the given question is unanswerable; their original intent was to leverage MRC knowledge to extract relations in zero-shot tasks. Later, Rajpurkar et al. (2018) introduced a crowdsourcing process for annotating unanswerable questions to create the SQuAD 2.0 dataset for Extractive Question Answering, which later inspired similar works in other languages such as French (Heinrich et al., 2021) and Vietnamese (Van Nguyen et al., 2022). However, recent work shows that models trained on SQuAD 2.0 perform poorly on out-of-domain samples (Sulem et al., 2021). In addition to the adversarially-crafted unanswerable questions proposed by Rajpurkar et al. (2018), Natural Question (Kwiatkowski et al., 2019) and Tydi QA (Clark et al., 2020) propose more naturally constructed unanswerable questions. While recent language models surpass human performances on adversarial unanswerable questions of SQuAD 2.0, natural unanswerable questions in Natural Question and Tidy QA remain challenging (Asai and Choi, 2021).

3 Tasks and Models

3.1 Extractive Question Answering

In the task of Extractive Question Answering (EQA) with questions, a machine learns to create a list of prospective outputs (answers), each of which is associated with a probability indicating the machine's confidence level about the answer to the question. When unanswerable questions are included in the dataset, a valid response can be an "empty" response, indicating the question is unanswerable. The model outputs the answer (including no-answer) with the highest probability as the final response to the question. The metric typically used to evaluate the MRC system is the **F1-score**, the average overlap between predictions and gold answers (see Rajpurkar et al. (2016) for more details).

3.2 Datasets

In our experiments, we fine-tune our MRC models by conducting additional training on one of the two versions of SQuAD (Stanford Question Answering Dataset): SQuAD 1.1 (Rajpurkar et al., 2016) and SQuAD 2.0 (Rajpurkar et al., 2018). We refer to models fine-tuned with SQuAD 1.1 as v1 models and models fine-tuned with SQuAD 2.0 as v2 models. For example, we refer to RoBERTa model fine-tuned with SQuAD 1.1 as RoBERTa v1. For testing, we supplement the two SQuAD datasets with five additional datasets from the MRQA 2019 shared task (Fisch et al., 2019): **Natural Questions (NQ)** (Kwiatkowski et al., 2019), **HotpotQA** (Yang et al., 2018), **SeachQA** (Dunn et al., 2017), **NewsQA** (Trischler et al., 2017), and **TriviaQA** (Joshi et al., 2017).

In addition to the adversarial attacks on answerable questions in SQuAD 1.1, we also produce adversarial attacks from the unanswerable samples of the development set of SQuAD 2.0. Due to the differences in the characteristics of attacks on answerable and unanswerable questions, we separately analyze the performances of models on each type of attack. While we evaluate v2 models under the attacks on both answerable and unanswerable questions, we only evaluate v1 models under the attacks on answerable questions since v1 models have never seen unanswerable questions. From adversarial attacks on answerable questions with v2 models, we gain critical insights into the current robustness effects of using unanswerable questions to fine-tune MRC models.

3.3 Models

We evaluate three, pre-trained state-of-the-art transformer models BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and SpanBERT (Joshi et al., 2020)) in our work. **BERT** (Devlin et al., 2019), the pioneer application of the Transformer model architecture (Vaswani et al., 2017), is trained on English Wikipedia plus BookCorpus with the pretraining tasks of masked language modeling (MLM) and next sentence prediction (NSP). Later, in a replication study of BERT pretraining, Liu et al. (2019) discovered that BERT was significantly under-trained. **RoBERTa** (Liu et al., 2019) improves over BERT mainly by increasing the pretraining time and the size of pretraining data. In empowering BERT to better represent and predict spans of text, **SpanBERT** (Joshi et al., 2020) masks random contiguous spans and replaces NSP with a span boundary objective (SBO). These three models are fine-tuned on datasets SQuAD 1.1 or SQuAD 2.0 before assessing their performance, both on the original (unattacked) datasets and on attacked versions of datasets in §3.2.

4 Adversarial Attacks

4.1 Robustness Evaluation

An EQA problem is given by a test set $\mathcal{D}$ of triplets (c, q, a) where c is the given context (usually a small paragraph of text), q is the question posed about that context, and a is the expected answer (or set of "gold" answers). The performance of the EQA model f is measured by

$$Per(f, \mathcal{D}) = \frac{1}{\mid \mathcal{D} \mid} \sum_{(c,q,a) \in \mathcal{D}} v(a, f(c, q))$$

where v is either the F1 or EM metric.

We create algorithm $\mathcal{A}$ to transform triplets (c, q, a) in $\mathcal{D}$ into adversarial test samples

Question Types	Question	Attacked Context	Answer
Answerable	What is the name of the water body that is found to the east?	To the east is the Colorado Desert and the **Colorado River** at the border with Arizona, and the Mojave Desert at the border with the state of Nevada. To the south is the Mexico–United States border. **Sea is the name of the water body that is found to the west.**	**Colorado River**
Unanswerable	What desert is to the south near Arizona?	To the east is the Colorado Desert and the Colorado River at the border with Arizona, and the Mojave Desert at the border with the state of Nevada. To the south is the Mexico–United States border. **The desert of edmonton desert is to the north near Burbank.**	

Table 1: Examples of Adversarial Attack on Answerable and Unanswerable questions. The adversarial sentence is highlighted in red color. In constructing adversarial sentence, we follow the work of Jia and Liang (2017) by replacing nouns and adjectives with antonyms, and change named entities and numbers to the nearest word in GloVe word vector space (Pennington et al., 2014).

(c', q', a') in the adversarial test set $\mathcal{D}_{attacked}$, where c', q', and a' are the modified (attacked) versions of c, q, and a. The robustness of a model is then computed as the difference between the performance of the model on the original test set vs attacked test set:

$$\Delta = Per(f, \mathcal{D}) - Per(f, \mathcal{D}_{attacked})$$

This framework was originally developed to assess robustness performance on answerable questions (Jia and Liang, 2017). In this paper, we also extend its application to attacks on unanswerable questions in Appendix §C.1 and discover challenges in this extended domain.

4.2 Attack Construction

Our algorithm constructs adversarial problems from original problems in a way similar to the AddOneSent in Jia and Liang (2017) and the AddText-Adv in Chen et al. (2022). Table 1 gives examples of such an attack on answerable and unanswerable questions. The additional sentence that is appended to the context has significant lexical overlap with the context, thus adding to the realism of the confusion-based attack. This type of adversarial attack is grammatical, fluent, and closely relevant to the given question. The questions and answers are unchanged for our considered adversarial attacks ($q' = q$ and $a' = a$).

Jia and Liang (2017) found that their adversarial attacks, especially the AddSent and AddOneSent attacks, were successful in challenging contemporary MRC models because the adversarial sentences were closely related to the given questions. Notably, the unanswerable questions in SQuAD 2.0 show a similar kind of lexical overlap with their corresponding contexts and require MRC models to be highly robust to the subtle syntactic changes in order to determine the answerability of given questions. Therefore, we hypothesize that models fine-tuned with SQuAD 2.0 are equipped to perform better against adversarial attacks.

In the next section we assess this hypothesis by evaluating the performance of v1 versus v2 models on answerable questions.

5 Attacks on Answerable Questions: Results

5.1 Adversarial Performance

		Answerable		
		Original	Attacked	Δ ↓
BERT	v1	88.4	63.8	24.6
	v2	78.4	55.2	**23.2**
RoBERTa	v1	91.5	70.5	**21.0**
	v2	84.8	58.0	26.8
SpanBERT	v1	91.5	68.6	**22.9**
	v2	85.8	58.9	26.8

Table 2: F1 scores of v1 models and v2 models with adversarial attacks on answerable questions. We refer to models fine-tuned on SQuAD 1.1 and SQuAD 2.0 as v1 and v2 models, accordingly.

Table 2 shows the performance of models with original (not attacked) and adversarial (attacked) problems on answerable questions. When attack sentences are added into context, the performance of all v1 and v2 models significantly decreases. Adding *unanswerable* questions into the training (v2 models) does not initially appear to improve the robustness of MRC models against adversarial attacks. In fact, the performance of v2 models appears to be less robust than that of v1 models, both on the original and the attacked questions. However, there is a deeper story here worth investigating. To further explain the poor performances of v2 models, we consider the types of v2 answers to answerable questions in the next section.

5.2 Categories of Responses

		I	C2I	C2U	C2C
BERT	v1	10.9	28.7	-	60.4
	v2	21.3	10.9	14.7	53.2
RoBERTa	v1	8.0	24.5	-	67.7
	v2	14.5	8.0	20.5	57.1
SpanBERT	v1	8.0	26.7	-	65.4
	v2	13.8	8.3	20.1	57.8

Table 3: The percentage of answerable questions by types of answers produced by v1 and v2 models before and after adversarial attacks.

Table 3 shows the different categories of answers produced by v1 and v2 models to answerable questions. We use a 50% F1 score threshold to determines the models' correctness to a question (correct if F1 score is above 50%, incorrect otherwise).

Considering attacks on answerable questions, we observe four categories in responses during attack: **“I" (incorrect)** are answerable questions that models originally got wrong (or originally predicted as unanswerable for v2 models). **“C2C" (correct to correct)** are answerable questions that models got correct both originally and after the attack. **“C2I" (correct to incorrect)** are answerable questions that models originally answered correctly but then output an incorrect answer when attacked. **“C2U" (correct to *incorrectly* unanswerable)** are answerable questions that models originally answer correctly but then predict as unanswerable when attacked. The C2I and C2U together account for the performance decline of models when attacked.

We see that v2 models, especially RoBERTa and SpanBERT, are particularly susceptible to the C2U challenge; they initially output a correct answer, but when attacked, decide (incorrectly) the question is now unanswerable. This is in contrast to the v1 models, which not being trained on unanswerable questions and do not have the option of responding "unanswerable". The v2 models' refusal to output an incorrect answer (opting instead to reply "unanswerable") indicates that their additional training on unanswerable questions has possibly provided them more depth to handle the confusion introduced by the attack.

We further breakdown the “C2U” category from Table 3 to investigate the spectrum of responses v2 models provide. Recall that models produce multiple responses to a MRC sample, each accompanied by a confidence score reflecting the models' confidence in that response. In this analysis, to evaluate the difficulty of questions in category “C2U” of each v2 model, we use the corresponding v1 model as baseline. Then, to answer the question whether v2 models prefer correct answers to incorrect answers, we evaluate the second most confident response of v2 models for questions in category “C2U”.

		C2U	
		Attacked	# Questions
BERT	v1	**46.1**	871
	v2	42.5	
RoBERTa	v1	**50.3**	1212
	v2	44.7	
SpanBERT	v1	46.1	1194
	v2	**47.6**	

Table 4: F1 scores of second most confident responses of v2 models and most confident responses of v1 models to questions in category “C2U” of v2 models in Table 3. For each language model, we extract a set of “C2U” questions and then evaluate corresponding v1 and v2 models on this set of questions.

Table 4 shows the F1 scores of *second* most confident responses of v2 models and *first* (most confident) responses of v1 models to questions in category “C2U” under attacks. We observe that v2 models often have fairly good answers for questions in category “C2U” given that performance of v2 models lag significantly behind that of v1 models when attacked. However, v2 models fail to put forward the correct answers (their second option) ahead of the "unanswerable" responses (their first option).

From these analyses, we hypothesize that models with additional training on *unanswerable* questions have the ability to perceive the attacks on *answerable* questions but fail to completely overcome them.

		Answerable		
		Original	Attacked	$\Delta \downarrow$
BERT	v1	88.4	63.8	24.6
	v2	88.5	69.6	**18.9**
RoBERTa	v1	91.5	70.5	21.0
	v2	91.4	75.1	**16.4**
SpanBERT	v1	91.5	68.6	22.9
	v2	91.3	75.8	**15.5**

Table 5: The performance of v1 and v2 models (when being forced to output non-empty answer on answerable questions) before and after adversarial attacks.

5.3 Force To Answer

The comparison of v1 and v2 models on answerable questions has a built-in bias because v2 models have the "penalty" of being able to respond "unanswerable" even though this is never a legitimate response. Furthermore, we have just shown that the v2 models often produce the correct answer, even under attack, but fail to put forward that correct output ahead of the "unanswerable" output in which it has more confidence. In this section, we re-run the analysis but this time eliminate the option for v2 models to output "unanswerable" (to answerable questions) so that we can better ascertain the robustness of v1 and v2 models to attacks.

Table 5 shows the results of this experiment. We can see now in this table that both v1 and v2 models exhibit similar performance on original answerable questions. When we introduce adversarial attacks on these same questions, the v2 models (being forced to answer) now exhibit noticeably stronger performance than their v1 counterparts. The additional training afforded to v2 models on unanswerable questions has given them a performance advantage over the v1 models. The robustness of v2 models against adversarial attacks is hidden in normal testing circumstances but can be realized by forcing the v2 models to output non-empty response in settings with only answerable questions.

6 Attacks in Out-Of-Domain Settings: Results

We now seek to determine if this additional robustness of v2 models extends to other out-of-domain test sets. In particular, we evaluate our v1 and v2 models on development sets of other Extractive Question Answering datasets. We summarized the characteristics of five out-of-domain datasets of MRQA 2019 in Table 6.

Table 7 shows the performance of v1 and v2 models on the five datasets of MRQA 2019. Similarly to experiments in Section 5, we measure performance on both original problems and adversarially attacked problems.

First, the performance on original (unattacked) problems shows that adversarial unanswerable questions in SQuAD 2.0 have little negative effects on the generalization performance of MRC models. While the performance of v2 models is higher than that of v1 models on TriviaQA and SearchQA, v1 models outperform v2 models slightly on Natural Questions (0.8%), NewsQA (0.2 %), and considerably on HotpotQA (6.5 %). On average, the generalization performance of v2 models to that of v1 models on out-of-domain unattacked problems is slightly worse (53.7% to 54.5%).

However, on problems with adversarial attacks, v2 models significantly outperform v1 models in four out of the five datasets. Specifically, on average, v2 models significantly outperform v1 models by 2.9% on NewsQA, 4.7% on Natural Question, 4.8% on SearchQA, and 5.2% on TriviaQA. Although v2 models do not show superior performance to v1 models on HotpotQA, the performance gap between v2 and v1 models after attacks decreases significantly thanks to the superior robustness of v2 models.

Overall, we conclude from Table 7 that adversarial unanswerable questions of SQuAD 2.0 do not have negative effects on the generalization of v2 models to out-of-domain datasets, and the robustness of v2 models against adversarial attack is consistently superior to that of v1 models.

7 New Attack

In this section, we explore *why* v2 models often incorrectly put forward "unanswerable" as an incorrect response to answerable questions under adversarial attacks. We hypothesize that MRC models trained with SQuAD 2.0 have learned to identify target sentences with significant lexical overlap to decide whether the corresponding questions are unanswerable; the models rely *primarily* on that target sentence to determine their output. This undesirable behavior of MRC systems may prevent them from using the whole paragraph to accurately

Dataset	**Question (Q)**	**Distant Supervision**	**Context (C)**	**$\mathbf{Q \perp C}$**	**Dev**
SQuAD	Crowdsourced	✗	Wikipedia	✗	10,507
HotpotQA	Crowdsourced	✗	Wikipedia	✗	5,904
TriviaQA	Trivia	✓	Web snippets	✓	7,785
SearchQA	Jeopardy	✓	Web snippets	✓	16,980
NewsQA	Crowdsourced	✗	News articles	✓	4,212
Natural Questions	Search logs	✗	Wikipedia	✓	12,836

Table 6: Characteristics of each datasets used in our out-of-domain experiments. Distant supervision is True if datasets used distant supervision to match questions and contexts. **$\mathbf{Q \perp C}$** is True if questions in datasets are written independently from the passage used for context. Table adopted from shared task MRQA 2019 (Fisch et al., 2019).

		Natural Question			**HotpotQA**			**TriviaQA**		
		Original	Attacked	$\Delta\downarrow$	Original	Attacked	$\Delta\downarrow$	Original	Attacked	$\Delta\downarrow$
BERT	v1	**54.6**	20.1	34.5	**61.6**	45.5	16.1	**59.4**	48.9	10.5
	v2	52	**23.7**	**28.3**	58.9	**47.4**	**11.5**	58.9	**53.3**	**5.6**
RoBERTa	v1	62.1	28.3	33.8	**67.4**	46.3	21.1	64.1	55	9.1
	v2	**63.5**	**33.2**	**30.3**	65	**49.8**	**15.2**	**65.5**	**59.2**	**6.3**
SpanBERT	v1	**65**	34.5	30.5	**66.2**	**46.4**	19.8	**63.2**	51.9	11.3
	v2	63.9	**40.2**	**23.7**	51.9	32.3	**19.6**	62.9	**58.8**	**4.1**
Average	v1	**60.6**	27.6	33	**65.1**	**46.1**	19	62.2	51.9	10.3
	v2	59.8	**32.3**	**27.5**	58.6	43.2	**15.4**	**62.4**	**57.1**	**5.3**

		SearchQA			**NewsQA**			**Average**		
		Original	Attacked	$\Delta\downarrow$	Original	Attacked	$\Delta\downarrow$	Original	Attacked	$\Delta\downarrow$
BERT	v1	**30.4**	25.5	4.9	53.6	41.8	11.8	**51.9**	36.4	15.5
	v2	28.6	**26.7**	**1.9**	**53.9**	**46.2**	**7.7**	50.5	**39.5**	**11**
RoBERTa	v1	22.8	20.3	2.5	**61.2**	**54.2**	**7**	55.5	40.8	14.7
	v2	**33**	**31.6**	**1.4**	60.6	52.5	8.1	**57.5**	**45.3**	**12.2**
SpanBERT	v1	28.1	26.9	1.2	**58.2**	44.1	14.1	**56.1**	40.8	15.3
	v2	**29.4**	**28.8**	**0.6**	58	**50**	**8**	53.2	**42**	**11.2**
Average	v1	27.1	24.2	2.9	**57.7**	46.7	11	**54.5**	39.3	15.2
	v2	**30.3**	**29**	**1.3**	57.5	**49.6**	**7.9**	53.7	**42.3**	**11.4**

Table 7: Robustness of MRC models fine-tuned on SQuAD 1.1 (v1) and SQuAD 2.0 (v2) in out-of-domain settings. For models fine-tuned on SQuAD 2.0 (v2), we force models to output non-empty answers. For each dataset, we report the average performance of 3 experimented models. We also report the average performance of each models on 5 considered datasets.

determine the best response to a question and have negative effects on the practical usage of adversarial unanswerable questions.

To further understand this hypothesis, we introduce a *negation attack*, a new adversarial attack to attempt to fool models into giving incorrect "unanswerable" responses. In particular, we construct an attack statement that significantly overlaps with the question yet is easy to determine as contradicting the question; we form our negation attack by inserting "not" in front of the adjective. Our attack (see Table 8) differs from previous adversarial attacks as our attack is designed to elicit an unanswerable response instead of an incorrect response.

Table 9 reports the performance of v2 models under negation attacks on answerable questions. We observe that our negation attack is highly effective in revealing the weaknesses of v2 models as the performance of all three considered v2 models significantly drops by almost 60% F1 when we introduce the negation attack.

We then examine the shifts in answers of v2 models when attacked with negation type. Table 10 shows the distribution of shifts in answers before and after the attack. We observe that the most

Question	In the effort of maintaining a level of abstraction, what choice is typically left **independent**?
Answer	**encoding**
Context	[...] one tries to keep the discussion abstract enough to be independent of the choice of **encoding**. [...] In the effort of maintaining a level of abstraction, base64 choice is typically left **not independent**.

Table 8: An example of the Negation Attack on answerable questions. The adversarial sentence is highlighted in red color. In constructing the adversarial sentence, we negate adjective "independent" to "not independent".

	Original	Attacked	Δ ↓
BERT v2	84.8	24.2	60.6
RoBERTa v2	78.1	21	57.1
SpanBERT v2	87.3	28.6	58.7

Table 9: F1 score of v2 models before and after negation attacks on answerable questions. In this experiment, we do not force v2 models to output non-empty answers.

significant drop in performance under negation attacks is the "C2U" category (around 40 % F1). This result is consistent with our hypothesis that v2 models rely on target sentences with significant lexical overlap to decide whether the corresponding questions are unanswerable.

8 Conclusion

In this work, we investigate the effects of training MRC models with unanswerable questions on their robustness against adversarial attacks. We construct adversarial samples from answerable and unanswerable questions in SQuAD 2.0 and evaluate three MRC models fine-tuned on either SQuAD 1.1 (v1 models) or SQuAD 2.0 (v2 models) independently.

Adversarial attacks on answerable questions reveal that v2 models initially show little improved robustness over v1 models yet possess a latent ability to deal with these attacks that v1 models do not; the correct responses are often hidden as second-best answers, an indicator of the "hidden robustness" of v2 models resulting from additional training on unanswerable questions. By eliminating the "unanswerable" option and forcing v2 models to output an answer to any answerable questions, we leverage this hidden robustness to improve the performance of MRC models to attacks on answerable questions. Furthermore, we also show that this robustness translates well to out-of-domain test sets.

	I	C2U	C2I	C2C
BERT v2	14.4	45.4	17.7	22.5
RoBERTa v2	21.6	41.8	17.5	19.1
SpanBERT v2	12.5	37.9	22.8	26.8

Table 10: The percentage of answerable questions by types of answers produced by v2 models before and after negation attacks.

Finally, to encourage future work in evaluating the robustness of MRC models trained on both answerable and unanswerable questions, we introduce a new type of adversarial attack to reveal the short-comings of MRC models. Our experiments with the *negation* attack reveal that the performance of v2 MRC models drops significantly (around 50% F1). We hypothesize that the decline in the performance of v2 models is mainly due to how v2 models have learned to suboptimally identify target sentences in the context to use as their primary mechanism of response.

9 Future Work

Our findings raise two critical messages for future research in the usage of adversarial unanswerable questions in NLP:

First, our work highlights innovative ways to use adversarial unanswerable questions in training to improve the performance of MRC-based systems. MRC datasets are important sources of transfer learning for zero-shot settings in many other NLP tasks (Wu et al., 2020; Levy et al., 2017; Lyu et al., 2021; Du and Cardie, 2020; Li et al., 2019). Given that the improved robustness of v2 models from the additional training on unanswerable questions generalizes well to out-of-domain test sets, future research about using MRC knowledge in zero-shot settings can explore whether adversarial unanswerable questions improve the robustness of MRC models in these zero-shot settings.

Second, we propose an open question about an undesirable behavior of MRC models fine-tuned on SQuAD 2.0. We find that simple negation attacks induce a considerable drop in the performance of MRC models fine-tuned on SQuAD 2.0

due to an undesirable behavior as the product of artifacts in the training set. To use the adversarial unanswerable questions in practice, we suggest additional research, based on insights about shortcut learning (Lai et al., 2021; Du et al., 2021), aimed to prevent MRC models from learning this undesirable behavior.

Limitations

We acknowledge that there exist few aspects to which our findings are limited, that include the dominant use of pretrained language models, the insufficiency of MRC datasets in other languages, and the limited types of adversarial attacks examined.

Acknowledgements

We would like to thank Dr. Ashwin Lall for constructive feedback on the early version of this paper. We thank the anonymous reviewers for their constructive and insightful feedback. We want to thank The William G. and Mary Ellen Bowen Research Endowment and The Laurie and David Hodgson Faculty Support Endowment for supporting the first and third authors.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Akari Asai and Eunsol Choi. 2021. Challenges in information-seeking QA: Unanswerable questions and paragraph retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1492–1504, Online. Association for Computational Linguistics.

Matthias Blohm, Glorianna Jagfeld, Ekta Sood, Xiang Yu, and Ngoc Thang Vu. 2018. Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 108–118, Brussels, Belgium. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Howard Chen, Jacqueline He, Karthik Narasimhan, and Danqi Chen. 2022. Can rationalization improve robustness? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3792–3805, Seattle, United States. Association for Computational Linguistics.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

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

Mengnan Du, Varun Manjunatha, Rajiv Jain, Ruchi Deshpande, Franck Dernoncourt, Jiuxiang Gu, Tong Sun, and Xia Hu. 2021. Towards interpreting and mitigating shortcut learning behavior of NLU models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 915–929, Online. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Quentin Heinrich, Gautier Viaud, and Wacim Belblidia. 2021. Fquad2.0: French question answering and knowing that you know nothing.

Alex Huang, Abdullah Al-Dujaili, Erik Hemberg, and Una-May O'Reilly. 2018. Adversarial deep learning for robust detection of binary encoded malware. *2018 IEEE Security and Privacy Workshops (SPW)*, pages 76–82.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Yuxuan Lai, Chen Zhang, Yansong Feng, Quzhe Huang, and Dongyan Zhao. 2021. Why machine reading comprehension models learn shortcuts? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 989–1002, Online. Association for Computational Linguistics.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 4208–4215. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Qing Lyu, Hongming Zhang, Elior Sulem, and Dan Roth. 2021. Zero-shot event extraction via transfer learning: Challenges and insights. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 322–332, Online. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Paarth Neekhara, Shehzeen Hussain, Shlomo Dubnov, and Farinaz Koushanfar. 2019. Adversarial reprogramming of text classification neural networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 5216–5225, Hong Kong, China. Association for Computational Linguistics.

Nicolas Papernot, Patrick Mcdaniel, Ananthram Swami, and Richard E. Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. *MILCOM 2016 - 2016 IEEE Military Communications Conference*, pages 49–54.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Suranjana Samanta and Sameep Mehta. 2018. Generating adversarial text samples. In *ECIR*.

Priyanka Sen and Amir Saffari. 2020. What do models learn from question answering datasets? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2429–2438, Online. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016a. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016b. Bidirectional attention flow for machine comprehension.

Chenglei Si, Ziqing Yang, Yiming Cui, Wentao Ma, Ting Liu, and Shijin Wang. 2021. Benchmarking robustness of machine reading comprehension models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 634–644, Online. Association for Computational Linguistics.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What makes reading comprehension questions easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219, Brussels, Belgium. Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8918–8927.

Elior Sulem, Jamaal Hay, and Dan Roth. 2021. Do we know what we don't know? studying unanswerable questions beyond SQuAD 2.0. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4543–4548, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hongxuan Tang, Hongyu Li, Jing Liu, Yu Hong, Hua Wu, and Haifeng Wang. 2021. DuReader_robust: A Chinese dataset towards evaluating robustness and generalization of machine reading comprehension in real-world applications. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 955–963, Online. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Kiet Van Nguyen, Son Quoc Tran, Luan Thanh Nguyen, Tin Van Huynh, Son T. Luu, and Ngan Luu-Thuy Nguyen. 2022. Vlsp 2021 - vimrc challenge: Vietnamese machine reading comprehension.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2022. Measure and improve robustness in NLP models: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4569–4586, Seattle, United States. Association for Computational Linguistics.

Yicheng Wang and Mohit Bansal. 2018. Robust machine comprehension models via adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 575–581, New Orleans, Louisiana. Association for Computational Linguistics.

Bowen Wu, Haoyang Huang, Zongsheng Wang, Qihang Feng, Jingsong Yu, and Baoxun Wang. 2019. Improving the robustness of deep reading comprehension models by leveraging syntax prior. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 53–57, Hong Kong, China. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.*, 11(3).

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. *ArXiv*, abs/1710.11342.

A Attacks

In this section, we document the pseudo-code we use to generate the two attacks in our main paper. In the pseudo-code below, $(\cdot)$ indicates the input(s) of the function within the current line.

A.1 AddOneSent Attack

Algorithm 1: AddOneSent Attack

```
Function AddOneSent(question, answer):
    new_question $\leftarrow$ question
    new_answer $\leftarrow$ answer
    new_question $\leftarrow$ Replace nouns and adjectives with antonyms in WordNet(new_question).
    new_question $\leftarrow$ Change named entities and numbers to nearest word in GloVe(new_question).
    new_answer $\leftarrow$ Change named entities and numbers to nearest word in GloVe(new_answer).
    Assert (new_answer $\neq$ answer) && (new_question $\neq$ question)
    attack $\leftarrow$ Convert into statement (new_question, new_answer).
    return attack
```

Algorithm 1 is the pseudo-code for AddOneSent attack used in our analysis.

A.2 Negation Attack

Algorithm 2: Negation Attack

```
Function Negation(question, answer):
    new_question $\leftarrow$ question
    new_answer $\leftarrow$ answer
    new_question $\leftarrow$ Add not before the first adjective (new_question).
    new_answer $\leftarrow$ Change named entities and numbers to nearest word in GloVe(new_answer).
    Assert (new_answer $\neq$ answer) && (new_question $\neq$ question)
    attack $\leftarrow$ Convert into statement (new_question, new_answer).
    return attack
```

Algorithm 2 is the pseudo-code for the Negation attack introduced in Section 7 to further reinforce our hypothesis that v2 models undesirably learn artifacts in adversarial unanswerable questions of SQuAD 2.0. The main difference between AddOneSent attack and Negation attack is that Negation attack does not use WordNet to Replace nouns and adjectives, and does not use GloVe to change named entities and numbers to nearest word in word space of GloVe.

A.3 Quality Analysis

In order to investigate the quality of Negation Attack, we manually label the 200 attack samples produced by both Negation Attack and AddOneSent attacks (100 each) into three categories:

1. **FM**: fluent and meaningful attack sentence.
2. **M**: meaningful but not fluent attack sentence.
3. **N**: not meaningful attack sentence.

Table 11 provide examples of Negation and AddOneSent attack samples categorized into these three categories. The errors of the Negation attack mostly come from the unnatural expression when using "not" to negate adjectives instead of using antonyms (not significant versus insignificant). On the other hand, the errors of AddOneSent can occur because of misclassifying word type. For example, when misclassifying the noun kind as adjectives, AddOneSent would then rewrite kind of company as unkind of company).

B Details for MRC Model Training

In this work, we use the base versions for all considered pre-trained models. We train all MRC models using mixed precision, with batch size of 4 sequences for 2 epochs. The maximum sequence length is set to 384 tokens. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with an initial learning rate of $2 \cdot 10^{-5}$, and $\beta_1 = 0.9$, $\beta_2 = 0.999$. We fine-tuned all four models on a single NVIDIA Tesla K80 provided by Google Colaboratory.

C Attacks on Unanswerable Questions: Results

C.1 Adversarial Performance

In this section, we extend our robustness evaluation of v2 models by analyzing their performance against adversarial attacks on unanswerable questions. Recall that we conduct these experiments only on v2 models as v1 models have not been trained on unanswerable questions.

	AddOneSent		Negation	
	Example	Proportion	Example	Proportion
FM	**Question**: Who was the chief executive officer when the service began? **Attack**: Russell Hartley was the chief executive officer when the disservice began.	44	**Question**: What service is a VideoGuard UK equipped receiver dedicated to decrypt? **Attack**: A VideoGuard UK equipped receiver is not dedicated to decrypt the service of skies.	43
M	**Question**: How populous is Victoria compared to other Australian states? **Attack**:Victoria compared to same japanese states is 3rd - most populous.	25	**Question**: What is the most important type of Norman art preserved in churches? **Attack**: The most not important type of Norman art is preserved in churches frescos.	49
N	**Question**: What kind of company is Sky UK Limited? **Attack**: The unkind of company of macedonian telecommunications company is geelong.	31	**Question**: What does most of the HD material use as a standard? **Attack**: The U.S. revolutionary peace does not most of the HD material use as a standard.	8

Table 11: Attack samples of Negation and AddOneSent categorized into three categories (fluent and meaningful, meaningful but not fluent, and not meaningful) and their overall proportions.

	Unanswerable		
	Original	Attacked	Δ ↓
BERT v2	72.2	69.3	2.9
RoBERTa v2	81.7	77.9	3.8
SpanBERT v2	76.4	75.3	1.1

Table 12: F1 score of v2 models with adversarial attacks on unanswerable questions.

Table 12 reports the performances of v2 models to adversarial attacks on unanswerable questions. Among the F1 scores of the three v2 models, the score of RoBERTa v2 decreases most after the attacks (by 3.8%) while the F1 score of SpanBERT v2 decreases least (by only 1.1%). These results *seem* to indicate that the adversarial attacks only slightly degrade the performances of v2 models, which might lead to erroneous conclusions about the robustness of these models. However, if we look back at Table 3, we see that between 8% and 11% of samples are in the C2I group (correct originally, incorrect when attacked). These prior results on answerable questions suggest inconsistencies with the results on unanswerable questions. We dig further.

C.2 Categories of Responses

	CU2CU	IA2IA	CU2IA	IA2CU
BERT v2	61.8	20.4	10.4	7.4
RoBERTa v2	71.8	11.1	9.9	7.2
SpanBERT v2	65.2	13.5	11.2	10.1

Table 13: The percentage of unanswerable questions by types of answers produced by v1 and v2 models before and after adversarial attacks.

We apply a similar investigation as we did previously to categorize the response changes of these v2 models to attacks on unanswerable questions. We find four main categories:

- **"CU2CU" (correctly unanswerable to correctly unanswerable)** are questions that v2 models correctly predicted as unanswerable both before and after the attacks.
- **"IA2IA" (incorrectly answerable to incorrectly answerable)** are unanswerable questions that v2 models attempt to output answers both before and after the attacks.
- **"CU2IA" (correctly unanswerable to incorrectly answerable)** are questions that v2 models originally correctly predicted as unanswerable but then output an answer when attacked.
- **"IA2CU" (incorrectly answerable to correctly unanswerable)** are questions that v2 models originally erroneously attempt to output an answer but later correctly predict as unanswerable when attacked.

What Table 13 reveals is that the performance loss of the models during the attack is being masked by some questions that were initially incorrect but are correctly identified as unanswerable after the attack (IA2CU). For example, the BERT model appears to only lose 2.9 F1 score during the attack, but actually it loses 10.4 and then gains back 7.4 in other IA2CU questions. These results reveal that v2 models experience a similar performance decline on unanswerable questions as they did on answerable questions. They also show how the current assessment framework is unsuitable for accurately measuring the robustness of v2 models on both answerable and unanswerable questions.

FrameBERT: Conceptual Metaphor Detection with Frame Embedding Learning

Yucheng Li[1*], **Shun Wang**[2*], **Chenghua Lin**[2†], **Frank Guerin**[1], **Loïc Barrault**[2,3]
[1] Department of Computer Science, University of Surrey, UK
[2] Department of Computer Science, University of Sheffield, UK
[3] Meta AI
{yucheng.li, f.guerin}@surrey.ac.uk
{swang209, c.lin}@sheffield.ac.uk
loicbarrault@meta.com

Abstract

In this paper, we propose FrameBERT, a RoBERTa-based model that can explicitly learn and incorporate FrameNet Embeddings for concept-level metaphor detection. FrameBERT not only achieves better or comparable performance to the state-of-the-art, but also is more *explainable* and *interpretable* compared to existing models, attributing to its ability of accounting for external knowledge of FrameNet.

1 Introduction

Metaphor is a pervasive linguistic device, which attracts attention from both fields of psycholinguistics and computational linguistics due to the key role it plays in the cognitive and communicative functions of language (Wilks, 1978; Lakoff and Johnson, 1980; Lakoff, 1993). Linguistically, metaphor is defined as a figurative expression that uses one or several words to represent another concept given the context, rather than taking the literal meaning of the expression (Fass, 1991). For instance, in the sentence "*This project is such a headache!*", the contextual meaning of *headache* is "a thing or person that causes worry or trouble; a problem", different from its literal meaning, "a continuous pain in the head".

Metaphor Detection presents a significant challenge as it necessitates comprehending the intricate associations between the abstract concepts embodied by the metaphoric expression and the related context. Recent studies in this field have demonstrated its potential to positively impact various Natural Language Processing (NLP) applications, including sentiment analysis (Cambria et al., 2017; Li et al., 2022a), metaphor generation (Tang et al., 2022; Li et al., 2022b,c), and mental health care (Abd Yusof et al., 2017; Gutiérrez et al., 2017). Different strategies have been proposed for modeling relevant context, including employing limited linguistic context such as subject-verb and verb-direct object word pairs (Gutiérrez et al., 2016), incorporating a wider context encompassing a fixed window surrounding the target word (Do Dinh and Gurevych, 2016; Mao et al., 2018), and considering the complete sentential context (Gao et al., 2018; Choi et al., 2021). Some recent efforts attempt to improve context modelling by explicitly leveraging the syntactic structure (e.g., dependency tree) of a sentence in order to capture important context words, where the parse trees are typically encoded with graph convolutional neural networks (Le et al., 2020; Song et al., 2021).

Despite the progress, we also observe the inadequacy of existing models in semantic modelling, which plays a crucial role in metaphor detection. For instance, it has been noted that BERT's tendency to aggregate shallow semantics instead of precise meaning, as its objective, may limit the context modelling ability (Xu et al., 2020). External knowledge such as FrameNet has been widely used to provide extra semantic information and has been shown useful in a wide range of NLP tasks, such as question answering (Shen and Lapata, 2007), machine reading comprehension (Guo et al., 2020), and identifying software requirements (Alhoshan et al., 2019). Very recently, FrameNet has also been employed to the task of metaphor generation via learning mappings between domains, with promising results achieved (Stowe et al., 2021). However, such a valuable source of knowledge, surprisingly, has not been explored in the deep learning literature for metaphor detection. We hypothesise that incorporating external knowledge of concepts is essential for improving metaphor detection and model explainability.

In this paper, we propose FrameBERT, a BERT-based model for conceptual metaphor detection underpinned by learning and incorporating embedding representation of semantic frames in FrameNet. FrameBERT directly addresses the lim-

---

[*] The two authors contributed equally to this work.
[†] Corresponding author

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1558–1563
May 2-6, 2023 ©2023 Association for Computational Linguistics

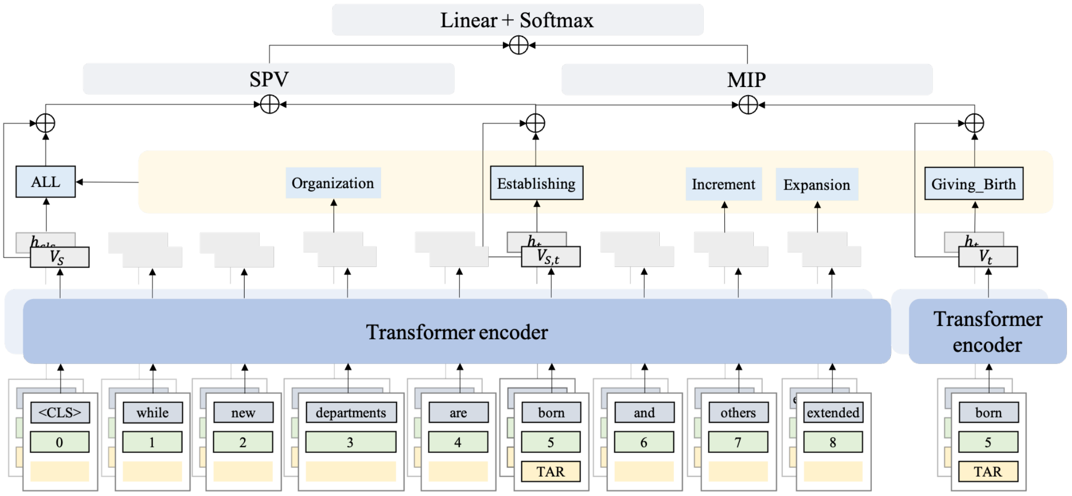


Figure 1: The overall framework. The surface encoder illustrates sentence encoder providing hidden-state representations and the insider one shows concept encoder producing concepts information. The frame embedding and hidden state embedding are concatenated to make final predictions.

itation of the existing works, which solely rely on the shallow semantics captured by hand-crafted psycholinguistics features or encoded by large pre-trained language models such as BERT. This is achieved by explicitly learning and incorporating FrameNet embeddings into the model training process. To our knowledge, this is the first attempt to apply FrameNet in deep learning models for metaphor detection. We also leverage Metaphor Identification Procedure (Group, 2007; Steen, 2010, MIP) and Selectional Preference Violation (SPV) (Wilks, 1975, 1978) to inform our model design.

Extensive experiments conducted on four public benchmark datasets (i.e., VUA MOH-X, TroFi) show that FrameBERT can significantly improve metaphor detection for all datasets compared to our base model without exploiting FrameNet embeddings. Our model also yields better or comparable performance to state-of-the-art models in Micro F1 measure. Furthermore, we show the *explainable* feature of FrameBERT, attributing to its ability of extracting semantic frames from text. The code and dataset can be found at `https://github.com/liyucheng09/MetaphorFrame`.

2 Model

We propose FrameBERT, a novel model that can explicitly learn and incorporate FrameNet embeddings for concept-level metaphor detection. Figure 1 illustrates the overall architecture of FrameBERT, which consists of two components: a sentence encoder (§ 2.1) and a concept encoder (§ 2.2).

2.1 Sentence Encoder

Similar to the prior work (Choi et al., 2021; Song et al., 2021), we develop the sentence encoder to produce the sentence encoding $\mathbf{v}_S$, the contextualised encoding for the target word $\mathbf{v}_{S,t}$, as well as isolated encoding for the target word $\mathbf{v}_t$. Formally, given an input sequence $S = (w_0, ..., w_n)$, RoBERTa (Liu et al., 2019) encodes each word into a set of contextualised embedding vectors $\mathbf{H} = (\mathbf{h}_{cls}, \mathbf{h}_0, ..., \mathbf{h}_n)$:

$$\mathbf{H} = \text{RoBERTa}(\text{emb}_{cls}, ..., \text{emb}_n) \quad (1)$$

where CLS token is a special token used to indicate the the beginning of the input; emb_i is the input embedding for word w_i represented as

$$\text{emb}_i = \text{emb}_w + \text{emb}_{pos} + \text{emb}_{type} \quad (2)$$

Here emb_w represents the word embedding, emb_{pos} is the position encoding for w_i, emb_{type} token type encoding indicating whether a word is a target or non-target word. We employ the CLS hidden state $\mathbf{h}_{cls}$ as the sentence representation, i.e., $\mathbf{v}_S = \mathbf{h}_{cls}$, the hidden states $\mathbf{h}_t$ of target word w_t as the contextual target word embedding, i.e., $\mathbf{v}_{S,t} = \mathbf{h}_t$. For the isolated word embedding for w_t, we directly feed w_t to RoBERTa in order to obtain the literal representation of the target word, i.e., $\mathbf{v}_t = \text{RoBERTa}(\text{emb}_t)$.

FrameBERT using MIP and SPV. With MIP, a metaphorical word is identified by the gap between the contextual and literal meaning of a word. To incorporate MIP, we employ the contextualised $\mathbf{v}_{S,t}$

and isolated embedding $\mathbf{v}_t$ vectors for w_t. With SPV, a metaphorical word is identified by the semantic difference from its surrounding words, i.e., the contrast between $\mathbf{v}_S$ and $\mathbf{v}_{S,t}$. We formalise our incorporation of these two metaphor identification theories below. Note that $\oplus$ is an operation for vector concatenation.

$$\mathbf{h}_{MIP} = \mathbf{v}_t \oplus \mathbf{v}_{S,t} \tag{3}$$

$$\mathbf{h}_{SPV} = \mathbf{v}_{S,t} \oplus \mathbf{v}_S \tag{4}$$

2.2 Conceptual Encoder

One of the key contributions of our paper is that our model can explicitly learn and incorporate FrameNet Embeddings for concept-level metaphor detection. This is achieved via the conceptual encoder, where we first fine-tuning a RoBERTa model on the FrameNet (Fillmore et al., 2002) dataset with a objective to classify frame lables, and then join the conceptual encoder with the sentence encoder.

Given an input sentence $S = (w_0, ..., w_n)$, we add a special token CLS at the beginning of the sentence and apply a stack of Transformer encoder layers on the tokenised input to obtain the contextualised hidden states for each word $\mathbf{H} = (\mathbf{h}_{cls}, \mathbf{h}_0, ..., \mathbf{h}_n)$ and the CLS token, similar to § 2.1. We then leverage the contextual target word hidden states and CLS hidden states (as sentence representation) to predict the target word's frame and all frames detected in the sentence. Formally, given CLS hidden states $\mathbf{h}_{cls}$ and a list of contextualised target word hidden states $\mathbf{H} = (\mathbf{h}_0, ..., \mathbf{h}_k)$, we obtain the frame distribution for sentence and targets as follows:

$$\hat{\mathbf{y}}^{\mathbf{f}}_{cls} = \text{sigmoid}(\mathbf{W}_0\mathbf{h}_{cls} + \mathbf{b}_0) \tag{5}$$

$$\hat{\mathbf{y}}^{\mathbf{f}} = \text{softmax}(\mathbf{W}_1\mathbf{H} + \mathbf{b}_1) \tag{6}$$

where $\mathbf{W}_0$ and $\mathbf{W}_1$ are learnable parameters, $\mathbf{b}_0$ and $\mathbf{b}_1$ are bias. Note that $\hat{\mathbf{y}}^{\mathbf{f}}_{cls}$ should be applied on all frame classes, that is compute it on each possible frame classess. We then minimise the following loss functions:

$$\mathcal{L}_{target} = -\sum \mathbf{y} \log \hat{\mathbf{y}}^{\mathbf{f}} \tag{7}$$

$$\mathcal{L}_{cls} = -\sum^{N}\sum_{i=0}^{L} \mathbf{y}_i \log \hat{\mathbf{y}}^{\mathbf{f}}_{cls} \tag{8}$$

$$+ (1 - \mathbf{y}_i) \log(1 - \hat{\mathbf{y}}^{\mathbf{f}}_{cls}) \tag{9}$$

where N is the number of training samples. L is number of frame labels, which means we are optimising the objective on all possible frame classes. We use λ as a hyperparameter controling weights between two losses: $\mathcal{L} = \lambda * \mathcal{L}_{cls} + \mathcal{L}_{target}$; and we set it to 2 in our experiments.

After the pre-training stage, the conceptual encoder will provide frame information for metaphor detection. As shown in Figure 1, in the MIP module, we concatenate the contextualised frame embedding $\mathbf{h}_{S,t}$ and isolated frame embedding $\mathbf{h}_t$ of target word to $\mathbf{h}_{MIP}$ (eq. 10). In the SPV module, we concatenate the CLS frame embedding $\mathbf{h}_{\mathbf{cls}}$ and contextualised target word frame embedding $\mathbf{h}_{S,t}$ to $\mathbf{h}_{SPV}$ (eq. 11).

$$\mathbf{h}_{MIP} = \mathbf{v}_t \oplus \mathbf{v}_{S,t} \oplus \mathbf{h}_t \oplus \mathbf{h}_{S,t} \tag{10}$$

$$\mathbf{h}_{SPV} = \mathbf{v}_S \oplus \mathbf{v}_{S,t} \oplus \mathbf{h}_{cls} \oplus \mathbf{h}_{S,t} \tag{11}$$

We then combine two hidden vectors $\mathbf{h}_{MIP}$ and $\mathbf{h}_{SPV}$ to compute a prediction score.

$$\hat{\mathbf{y}} = \sigma(\mathbf{W}_T(\mathbf{h}_{MIP} \oplus \mathbf{h}_{SPV}) + \mathbf{b}) \tag{12}$$

Finally, we use binary cross entropy loss to train the overall framework for metaphor prediction.

$$\mathcal{L} = -\sum_{i=0}^{N} \mathbf{y}_i \log \hat{\mathbf{y}}_i - (1 - \mathbf{y}_i) \log(1 - \hat{\mathbf{y}}_i) \tag{13}$$

3 Experiments

Dataset. We conduct experiments on four public bench datasets. **VUA-18** (Leong et al., 2018) and **VUA-20** (Leong et al., 2020), the extension of VUA-18, are the largest publicly available datasets. The **MOH-X** dataset is constructed by sampling sentences from WordNet (Miller, 1998). Only a single target verb in each sentence is annotated. The average length of sentences is 8 tokens, the shortest of our three datasets. **TroFi** (Birke and Sarkar, 2006) consists of sentences from the 1987-89 Wall Street Journal Corpus Release 1 (Charniak et al., 2000). The average length of sentences is the longest of our datasets (i.e., 28.3 tokens per sentence). At last, the concept encoder was pre-trained on **FrameNet release 1.7** (Fillmore et al., 2002) with about 19k, 6k, 2k annotations for training, testing and evaluation respectively.
Baselines. **RNN_ELMo** (Gao et al., 2018) combined ELMo and BiLSTM as a backbone model. **RNN_MHCA** (Mao et al., 2019): they introduced MIP and SPV into RNN_ELMo and capture the contextual feature within window size by multi-head attention. **MUL_GCN** (Le et al., 2020) apply a GCN based multi-task framework by modelling

Models	VUA18			VUA20		
	Prec	Rec	F1	Prec	Rec	F1
RNN_ELMo	71.6	73.6	72.6	-	-	-
RoBERTa_SEQ	80.1	74.4	77.1	75.1	67.1	70.9
MelBERT ⋆	79.6	76.4	77.9	76.4	68.6	72.3
MelBERT	80.1	76.9	78.5	75.9	69.0	72.3
MrBERT	82.7	72.5	77.2	-	-	-
FrameBERT	82.7	75.3	**78.8***	79.1	67.7	**73.0***

Table 1: Performance comparison on VUA datasets (best results in **bold**). NB: ⋆ indicates the reproduced results of MelBERT using the original source code and setting of (Choi et al., 2021). * denotes our model outperforms the competing model with $p < 0.05$ for a two-tailed t-test; except MrBERT whose code is not published.

	Models	Prec	Rec	F1
TroFi	RNN_MHCA	68.6	76.8	72.4
	MUL_GCN	73.1	73.6	73.2
	MrBERT	73.9	72.1	72.9
	FrameBERT	70.7	78.2	**74.2**
MOH-X	RNN_MHCA	77.5	83.1	80.0
	MUL_GCN	79.7	80.5	79.6
	MrBERT	84.1	85.6	**84.2**
	FrameBERT	83.2	84.4	83.8

Table 2: Performance comparison of our method with baselines on TroFi and MOH-X (best results in **bold**). We do not perform a significance test since the code of MrBERT is not published.

metaphor detection and word sense disambiguation. **RoBERTa_SEQ** (Leong et al., 2020) is a fine-tuned RoBERTa model in sequence labeling setting for metaphor detection. **MelBERT** (Choi et al., 2021) realize MIP and SPV theories via a RoBERTa based model. **MrBERT** (Song et al., 2021) is the recent sota on verb metaphor detection based on BERT with verb relation encoded.

4 Experimental Results

Overall results. Table 1 shows a comparison of the performance of our model against the baseline models on VUA18 and VUA20 respectively. Our model outperforms all the baseline models on VUA-20, including the state-of-the-art-model MelBERT (with $p < 0.05$ for a two-tailed t-test). For VUA-18, we outperformed all the baselines significantly including the *re-produced* results for MelBERT. Table 2 shows the results on the MOH-X and TroFi dataset. The results shows our method beats SOTA method on TroFi benchmark and gains comparable performance on MOH-X dataset.

Models	Prec	Rec	F1
FrameBert	82.7	75.3	**78.8**
rand_in_eval	81.8	58.7	68.3
rand_in_train_&_eval	79.3	72.6	75.8
w/o frame fine-tuning	79.1	76.3	77.6

Table 3: Results of ablation study, tested on VUA18. *rand_in_eval* represents the first experiment where the shuffle process is only performed in evaluation stage and *rand_in_train_&_eval* represents the second experiment where the shuffle process is performed in both training and evaluation stages. In *w/o frame fine-tuning* experiment, we remove the frame fine-tuning process.

Ablation Study. We performed three experiments to test the effectiveness of conceptual information. First, the system is fed with shuffled conceptual embeddings in the batch during evaluation. Second, in both training and evaluation processes, we shuffle the conceptual embeddings in the batch. Third, we remove the concept fine-tuning process. In all experiments, the overall framework remains the same as the original setting. The results are provided in Table 3. Based on the results, the performance in terms of F1 drops by 13% and 3.7% while feeding random conceptual information in only evaluation stage and both training and evaluation stages (likely collapse into the base model) respectively, which demonstrates the extent to which the conceptual information is incorporated in the overall framework (i.e. especially when we shuffle only for evaluation). The third experiment shows the performance declines 1.2% while removing the frame fine-tuning procedure, which proves the effectiveness of frame embedding learning.

Concept Analysis. In this section, we illustrate how the proposed approach detect metaphor in an interpretable way and how well the method using frame features. We performed an exploratory analysis on 200 examples where our system had a correct classification, but MelBERT failed. The following two examples show how frame information works in the metaphor detection procedure. The forst is a true positive example with the target word in bold: '*Local people mutter and march, make speeches and throw things; staff **face** sarcasm in nearby pubs* . . . .' Here our system had the following concepts as the literal meaning for '*face*': '*Body_parts, Facial_expression, Change_posture*', which are more basic meanings, relating to the face as a part of the body. In contrast, contextual concepts are extracted as follows: '*Confronting_problem, Resolve_problem, Surviving*'. These capture well the contextual meaning of 'face' in the sentence. The contextual meanings are more abstract, and the contrast between literal and contextual concepts helps the model to detect the metaphorical usage of *face* here. An example of a true negative is: '*...**hot** computers are slow, the warmth might damage...*'. 'Hot' is a word that can often be used metaphorically (e.g. hot topic, hot pants, hot properties), but in this sentence our model correctly identified it as literal and contextual concepts identified were identical: '*Temperature, Fire_Burning*'. In terms of how well our method using frame features, we measured the accuracy of the frame prediction module manually for these 200 examples, and found the correct frame label was identified in the top 3 frame label prediction for 178 of 200 examples (89%). This indicates our method is effective extracting frame features.

5 Conclusion

We proposed FrameBERT, the first conceptual model for metaphor detection by explicitly learning and incorporating FrameNet Embeddings for concept-level metaphor detection. Extensive experiments show that our model can yield better or comparable performance to the state-of-the-art.

6 Limitations

This paper mainly models frame information by representation learning on the frame classification task. However, other features such as Frame Elements (FEs) and Lexical Units (LUs) in FrameNet have not been explored in this paper, where our case analysis shows these features could provide useful signals for metaphor detection. It might also be promising to explore other types of knowledge such as context graphs (Cheng et al., 2021) for improving metaphor detection. We leave these directions to future works.

References

Noor Fazilla Abd Yusof, Chenghua Lin, and Frank Guerin. 2017. Analysing the causes of depressed mood from depression vulnerable individuals. In *Proceedings of the International Workshop on Digital Disease Detection using Social Media 2017 (DDDSM-2017)*, pages 9–17.

Waad Alhoshan, Riza Batista-Navarro, and Liping Zhao. 2019. Using frame embeddings to identify semantically related software requirements. In *REFSQ Workshops*.

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Erik Cambria, Soujanya Poria, Alexander Gelbukh, and Mike Thelwall. 2017. Sentiment analysis is a big suitcase. *IEEE Intelligent Systems*, 32(6):74–80.

Eugene Charniak, Don Blaheta, Niyu Ge, Keith Hall, John Hale, and Mark Johnson. 2000. Bllip 1987-89 wsj corpus release 1. *Linguistic Data Consortium, Philadelphia*, 36.

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. 2021. Guiding the growth: Difficulty-controllable question generation through step-by-step rewriting. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5968–5978, Online.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. Melbert: metaphor detection via contextualized late interaction using metaphorical identification theories. *arXiv preprint arXiv:2104.13615*.

Erik-Lân Do Dinh and Iryna Gurevych. 2016. Token-level metaphor detection using neural networks. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 28–33, San Diego, California. Association for Computational Linguistics.

Dan Fass. 1991. met*: A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 17(1):49–90.

Charles J Fillmore, Collin F Baker, and Hiroaki Sato. 2002. The framenet database and software tools. In *LREC*.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. *arXiv preprint arXiv:1808.09653*.

Pragglejaz Group. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and symbol*, 22(1):1–39.

Shaoru Guo, Ru Li, Hongye Tan, Xiaoli Li, Yong Guan, Hongyan Zhao, and Yueping Zhang. 2020. A frame-based sentence representation for machine reading comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 891–896.

E. Darío Gutiérrez, Guillermo Cecchi, Cheryl Corcoran, and Philip Corlett. 2017. Using automated metaphor identification to aid in detection and prediction of first-episode schizophrenia. In *Proceedings of the*

2017 Conference on Empirical Methods in Natural Language Processing, pages 2923–2930, Copenhagen, Denmark. Association for Computational Linguistics.

E. Dario Gutiérrez, Ekaterina Shutova, Tyler Marghetis, and Benjamin Bergen. 2016. Literal and metaphorical senses in compositional distributional semantic models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 183–193.

George Lakoff. 1993. The contemporary theory of metaphor.

George Lakoff and Mark Johnson. 1980. Metaphors we live by: University of chicago press. *Chicago, IL.*

Duong Le, My Thai, and Thien Nguyen. 2020. Multitask learning for metaphor detection with graph convolutional neural networks and word sense disambiguation. In *AAAI*, pages 8139–8146.

Chee Wee Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 vua and toefl metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29.

Chee Wee Leong, Beata Beigman Klebanov, and Ekaterina Shutova. 2018. A report on the 2018 vua metaphor detection shared task. In *Proceedings of the Workshop on Figurative Language Processing*, pages 56–66.

Yucheng Li, Frank Guerin, and Chenghua Lin. 2022a. The secret of metaphor on expressing stronger emotion. In *Proceedings of the 3rd Workshop on Figurative Language Processing (FLP)*, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Yucheng Li, Chenghua Lin, and Frank Guerin. 2022b. Cm-gen: A neural framework for chinese metaphor generation with explicit context modelling. In *International Conference on Computational Linguistics*.

Yucheng Li, Chenghua Lin, and Frank Guerin. 2022c. Nominal metaphor generation with multitask learning. In *Proceedings of the 15th International Conference on Natural Language Generation*, pages 225–235.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and wordnet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1222–1231.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3898.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 12–21, Prague, Czech Republic. Association for Computational Linguistics.

Wei Song, Shuhui Zhou, Ruiji Fu, Ting Liu, and Lizhen Liu. 2021. Verb metaphor detection via contextual relation learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4240–4251.

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Kevin Stowe, Tuhin Chakrabarty, Nanyun Peng, Smaranda Muresan, and Iryna Gurevych. 2021. Metaphor generation with conceptual mappings. *arXiv preprint arXiv:2106.01228*.

Cheng Tang, Frank Guerin, Yucheng Li, and Chenghua Lin. 2022. Recent advances in neural text generation: A task-agnostic survey. *ArXiv*, abs/2203.03047.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial intelligence*, 6(1):53–74.

Yorick Wilks. 1978. Making preferences more active. *Artificial intelligence*, 11(3):197–223.

Weidi Xu, Xingyi Cheng, Kunlong Chen, and Taifeng Wang. 2020. Symmetric regularization based bert for pair-wise semantic reasoning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1901–1904.

Towards More Efficient Insertion Transformer with Fractional Positional Encoding

Zhisong Zhang[1]*, **Yizhe Zhang**[2]†, **Bill Dolan**[3]
[1]Carnegie Mellon University, [2]Apple Inc., [3]Microsoft Research
zhisongz@cs.cmu.edu,yizhe.zhang@hotmail.com,billdol@microsoft.com

Abstract

Auto-regressive neural sequence models have been shown to be effective across text generation tasks. However, their left-to-right decoding order prevents generation from being parallelized. Insertion Transformer (Stern et al., 2019) is an attractive alternative that allows outputting multiple tokens in a single generation step. Nevertheless, due to the incompatibility between absolute positional encoding and insertion-based generation schemes, it needs to refresh the encoding of every token in the generated partial hypothesis at each step, which could be costly. We design a novel reusable positional encoding scheme for Insertion Transformers called Fractional Positional Encoding (FPE), which allows reusing representations calculated in previous steps. Empirical studies on various text generation tasks demonstrate the effectiveness of FPE, which leads to floating-point operation reduction and latency improvements on batched decoding.

1 Introduction

Transformer-based models (Vaswani et al., 2017) have been successfully applied to various text generation tasks (Gong et al., 2019; Wang et al., 2019; Ahmad et al., 2020; Zhang et al., 2020a; Lewis et al., 2020; Brown et al., 2020). Most of these models utilize a fixed left-to-right auto-regressive generation strategy, where the strict factorization means that the model can only generate one token per step. This makes it difficult to parallelize the decoding process, while parallel generation may help to improve decoding efficiency.

Recently, insertion-based sequence-generation models (Stern et al., 2019; Gu et al., 2019a; Welleck et al., 2019) have been developed as attractive alternatives to the auto-regressive ones by allowing flexible generation order. In particular, the Insertion Transformer (Stern et al., 2019), which combines the Transformer architecture and the insertion-based strategy, can match the performance of an auto-regressive model while requiring many fewer decoding steps with parallel generation.

The original Insertion Transformer utilizes absolute positional encoding as in the vanilla auto-regressive transformer. In the vanilla transformer, due to its left-to-right generation scheme, tokens' absolute positions do not change; thus, previous computation can be reused. However, this property no longer holds if insertion is allowed, and the Insertion Transformer re-encodes all previously generated tokens at each decoding step, which brings additional computational overheads.

In this work, we propose a reusable positional encoding scheme called Fractional Positional Encoding (FPE) to accelerate the Insertion Transformer. This scheme dynamically calculates each token's positional representations according to its left and right neighbors at insertion time. In this way, each token's positional representations will not change during the decoding process so that the computation can be reusable in the same way as the vanilla transformer, leading to a reduction of computation for the Insertion Transformer.

We evaluate FPE with a range of text generation tasks, including machine translation, word reordering, summarization as well as an open-ended text completion task. We show that the proposed scheme can reduce floating point operations of the insertion-based model while maintaining comparable performance to the vanilla transformer.

Our implementation is available at `https://github.com/zzsfornlp/zgen1/`.

2 Insertion Transformer

Insertion Transformer (Stern et al., 2019) generates the target sequences via a series of insertion operations. This provides a flexible scheme that can enable different generation orders as well as parallel generation. We focus on the parallel-generation

---

*Work done during an internship at Microsoft Research.
†Work done at Microsoft Research.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1564–1572
May 2-6, 2023 ©2023 Association for Computational Linguistics

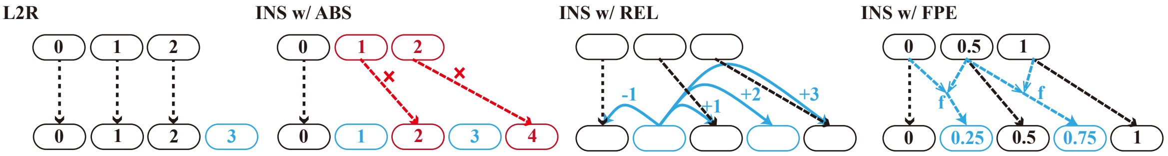


Figure 1: Illustrations of different positional encoding schemes. Black (or red) nodes denote the tokens that already exist in the previous step, while the newly generated ones are in blue. In ABS, the positional embeddings in previous steps may be non-reusable since absolute positions may change (marked as red). Representations can be made reusable by using REL or our proposed FPE. (Note that the fractional numbers in the FPE figure are only for better illustration, in practice, we adopt embeddings calculated by a learnable linear function.)

variant that inserts multiple tokens at each step. While the left-to-right scheme can only append one token at each step, the insertion-based scheme can add multiple tokens at different slots, thus potentially enabling more efficient generation.

3 Positional Encoding

Figure 1 provides an overview of different positional encoding schemes that we explore. The vanilla left-to-right (**L2R**) Transformer model (Vaswani et al., 2017) adopts a simple absolute positional encoding scheme by assigning left-to-right increasing indexes to each token. This naturally fits the left-to-right generation procedure and allows the previously calculated hidden representations to be reusable. However, in the insertion-based generation, since tokens can be inserted before previously generated tokens, the absolute position of a token may change. Therefore, if still using the absolute positional encoding (**ABS**), the previously calculated hidden layers cannot be reused and the Insertion Transformer needs to re-encode all the existing tokens at each step. This yields computation overhead, which may offset the computation gain from parallelization.

To solve this problem, alternative positional encoding schemes are required. Relative positional encoding (**REL**; Shaw et al., 2018) is an example, which has been adopted for insertion-based models (Lu et al., 2022). Here, each token records its relative positional information at its insertion time. Though this scheme alleviates encoding absolute positions and allows reusing, it requires complex modifications in the attention calculations.

In this work, we design fractional positional encoding (**FPE**), which is a simpler alternative scheme that only modifies the input embeddings. We still give each token a positional embedding $\mathbf{p}$, which is dynamically calculated along the generation process. Whenever a new token w_{new} is inserted between two existing tokens w_{left} and w_{right}, its positional representations will be calculated with a function f applying to its current left and right neighbors: $\mathbf{p}_{\text{new}} = f(\mathbf{p}_{\text{left}}, \mathbf{p}_{\text{right}})$. In this way, we will have the "fractional"-styled positions. The positional representations of all the tokens will not change throughout the decoding process, and re-encoding is no longer needed.

We specify the FPE representations $\mathbf{p}$ to have the same dimension as the model size and add them to the input embeddings as in the vanilla transformer. We further specify two randomly-initialized embeddings $\mathbf{p}_B$ and $\mathbf{p}_E$ for the beginning- and ending-of sequence tokens, respectively. The function f is modeled by a linear layer[1] which takes the concatenation of the two neighbors' positional embeddings and outputs a new vector of the model size. At training time, all these FPE-related parameters are tuned along with other parameters in the model. The linear layer is lightweight compared to the transformer layers, thus introducing negligible cost.

4 Experiments

4.1 Settings

We explore a variety of generation tasks, including machine translation, word reordering, summarization, as well as an open-ended text completion task. We use WMT14 En-De (Bojar et al., 2014) for machine translation, sentences in WikiText-103 (Wiki103; Merity et al., 2016) for word reordering, XSUM (Narayan et al., 2018) for summarization and paragraphs in Wiki103 for completion. In the text completion task, the model is required to complete each paragraph according to the existing con-

---

[1] We start with the simple linear layer and find it works reasonably well. We also tried some other methods such as adding non-linearity activation but did not find obvious benefits. Therefore, we adopt this simple method.

Task	Model	Evaluation ↑	#Step	#Len	Latency ↓
Translation (WMT14 EN-DE)	L2R	**27.72**	28.4	22.1	230.1
	ABS	27.45	5.7	21.5	100.3
	REL	27.40	5.5	21.5	105.0
	FPE	27.47	5.6	21.4	**97.2**
Text Reordering (Wiki-103)	L2R	**52.82**	27.9	24.8	224.7
	ABS	50.69	6.8	24.2	113.9
	REL	52.63	6.0	24.6	120.7
	FPE	52.52	6.0	24.8	**105.9**
Summarization (XSUM)	L2R	31.33/11.65/25.32	21.3	19.7	206.9
	ABS	**32.09**/11.39/**25.68**	6.7	24.3	114.9
	REL	31.90/**11.66**/25.80	6.2	22.9	125.2
	FPE	31.78/11.57/25.67	6.2	22.7	**114.4**
Text Completion (Wiki-103)	L2R	**3.87/8.48/14.54**	55.6	48.9	468.0
	ABS	1.19/7.66/12.90	9.5	49.4	141.9
	REL	1.69/8.41/13.23	8.1	54.7	161.0
	FPE	1.61/8.26/13.47	8.0	52.3	**129.9**

Table 1: Main results of comparing an auto-regressive left-to-right (L2R) model and three insertion models with absolute (ABS), relative (REL), and fractional (FPE) positional encoding. "Evaluation" denotes automatic evaluation metrics: BLEU for MT and reordering, R-1/R-2/R-L for summarization, and BLEU/METEOR/R-L for completion. "#Step" and "#Len" indicate the average decoding steps and output lengths, respectively. "Latency" denotes the actual average decoding time (ms) per instance with single-instance decoding.

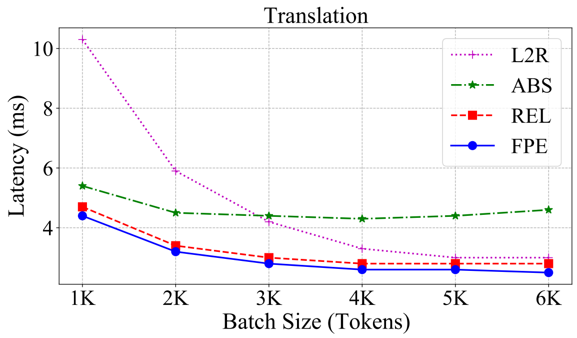


Figure 2: Latency of MT models with different decoding batch sizes (source tokens). Results with single-instance decoding are not shown here since since its latency is much higher.

Batch-size	1K	2K	3K	4K	5K	6K
L2R	10.3	5.9	4.2	3.3	3.0	3.0
ABS	5.4	4.5	4.4	4.3	4.4	4.6
REL	4.7	3.4	3.0	2.8	2.8	2.8
FPE	4.4	3.2	2.8	2.6	2.6	2.5

Table 2: Latency of MT models (milliseconds per instance) with different decoding batch sizes (source tokens). This table shows the detailed numbers corresponding to those in Figure 2.

text. All the decoding experiments are performed with one V100 GPU. Please refer to Appendix A and B for more dataset and experimental details.

4.2 Results

We compare our method (FPE) with the vanilla transformer (L2R), and two other insertion-based models with absolute (ABS) and relative (REL) positional encoding. The main results are shown in Table 1. For automatic performance evaluations, the three insertion-based transformer models (ABS, REL, and FPE) achieve similar results. Compared with L2R, the insertion models' performance is competitive on MT, reordering, and summarization tasks, while being behind on the open-ended text completion task. This is presumably due to the conditional independence assumption in the parallel generation steps. This issue is beyond the scope of this paper, so we leave it to future work.

For efficiency, insertion-based models can generate target sequences with much fewer decoding steps, leading to latency reduction where the insertion models can achieve around 2x speedups compared to L2R in single-instance mode.

4.3 Batched Decoding

We further explore batched decoding, which is usually adopted to speed up the computation via data parallelism. The latency of MT models against different batch sizes can be found in Figure 2 and Table 2, from which we observe that:

- ABS becomes less efficient when decoding in batches, probably due to the extra computations[2]

---

[2] We further measure the floating point operations (FLOPs)

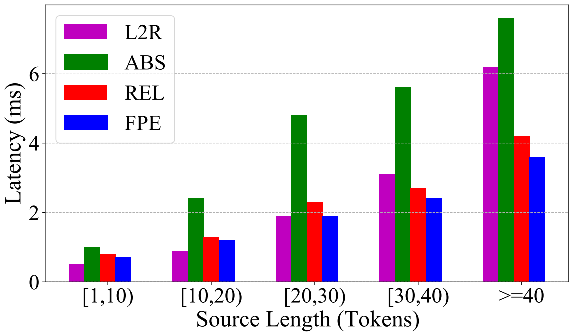


Figure 3: Latency breakdowns on source sequence lengths for the MT task with batched decoding (6K).

brought by re-encoding. Though this does not affect its efficiency in the single-instance mode where GPU's computational capacity may not be fulfilled yet, in batched decoding the extra re-encoding computations greatly dampen its efficiency improvements.

- FPE and REL are faster than L2R for relatively small batch sizes. While the batch size becomes larger, the efficiency gain becomes less obvious. Presumably, more complex indexing operations in the insertion-based schemes do not utilize GPUs as fully as L2R. We leave this optimization to future work.
- REL behaves similarly to FPE, but is consistently around 10% slower, probably due to the extra relative positional computations in attentions.

The patterns in other tasks are similar to MT and are shown in Appendix C.

Note that many previous works consider only single-instance or batched decoding mode when measuring efficiency, while we examine both to include a spectrum of real scenarios covering various device memory capacities and querying patterns. While the L2R model and the original Insertion Transformer (w/ ABS) only excel at one end, FPE could help to make the model efficient for both scenarios, potentially benefiting more use cases.

4.4 Length Breakdown

We further perform an ablated speed analysis by breaking down the instances by sequence length. Here, we investigate the task of machine translation and split the instances into different bins according to source lengths. The breakdown results with a batch size of 6K are shown in Figure 3. The trends are generally similar to the overall results. ABS is not quite efficient for batched decoding and can be even slower than L2R. Though the utilization of FPE does not provide obvious efficiency improvements on short sentences, it brings benefits for generating longer sequences. It can achieve a 1.7x speedup over L2R for sentences that are longer than 40 tokens, even with a large batch size. Similar to the overall trend, FPE is consistently more efficient than REL, yet being simpler.

5 Related work

Generation Order. Previous works have been exploring relaxing the output dependencies and allowing parallel generation. The Non-Autoregressive Transformer (NAT) (Gu et al., 2018) enables the decoder to generate target sequences in one or several decoding steps (Gu et al., 2018; Gu and Kong, 2021; Lee et al., 2018; Gu et al., 2019b; Ghazvininejad et al., 2019). Most of these models require predicting target length and generating multiple consecutive tokens at once. The generation is sometimes not fluent, as multiple tokens may compete for the same meaning. The insertion-based methods (Stern et al., 2019; Gu et al., 2019a; Welleck et al., 2019; Chan et al., 2020; Zhang et al., 2020b) also change the standard left-to-right generation by allowing dynamically inserting tokens for the generation process. This provides a good balance between generation fluency and efficiency, and does not require predicting target lengths first. In this work, we follow this insertion-based generation scheme and further improve its efficiency. In addition to efficiency, allowing flexible generation order is another motivation to study non-L2R generation schemes. A good generation order may bring performance benefits (Ford et al., 2018; Jiang and Bansal, 2021).

Reusable Positional Encoding. In insertion-based models, the absolute positional encoding can be non-reusable since an inserted token will change the absolute positions of its following tokens. Alternative schemes are required to enable reusable encoding. Relative positional encoding (Shaw et al., 2018) is utilized for the insertion-based generation in some recent work (Gu et al., 2019a; Lu et al., 2022) to avoid the re-encoding of the previously generated tokens. However, it requires modifications to the inner attention mech-

---

for decoding an instance following the method of Elbayad et al. (2020) and find that the ABS-based model requires much larger FLOPs than other models. For example, on the MT models, FLOPs per instance is 8.69B for ABS, while FPE only requires 4.65B (REL needs 4.68B).

anism of the Transformer, while our FPE scheme is a simpler alternative that only modifies the input. In a similar spirit to our scheme, Shiv and Quirk (2019) explore a tree-based positional encoding strategy. Our scheme is different in that in the insertion-based generation, each node has two parent nodes, yielding a graph structure rather than a tree.

6 Conclusion

In this work, we investigate the re-encoding issue that sometimes hinders the Insertion Transformer from receiving its computation gain and propose a Fractional Positional Encoding scheme that is naturally compatible with the insertion-based generation scheme to solve this issue. With experiments on various tasks, we show that this simple scheme eliminates the need of re-encoding the previously generated tokens and obtains a promising balance between efficiency and performance.

Limitations

This work has several limitations. First, we mainly rely upon the architecture and decoding strategy of the Insertion Transformer, which only allows the generation of one token between two neighboring tokens at one step. It would be interesting to consider more flexible generation schemes. It would also be interesting to compare our models with other (semi) non-autoregressive models, which we leave to future work. Moreover, we follow the best-performing binary tree training objective of the Insertion Transformer, which in some way sacrifices the flexibility of the generation order. It would be interesting to explore the application of the proposed positional encoding scheme with more flexible generation orders. It would also be interesting to explore the impacts of using larger pre-trained models and investigate how it interacts with the insertion-based scheme. Finally, on the open-ended generation task, the insertion-based method still performs worse than the left-to-right one, which requires further investigation since the output dependencies would need more careful modeling in the open-ended scenarios.

Broader Impact

This work focuses on improving for the natural language processing (NLP) and general artificial intelligence (AI) research community. Our work can be leveraged to improve natural language generation (NLG) models, including but not limited to text editing, conversational agents, and question answering systems. The **broader impact** and the **risks** of this work are summarized as following:

- This work can facilitate research in the NLG tasks in a generic manner, to potentially accelerate generations in applications like machine translation, text summarization, and virtual assistants.
- This work is a fundamental research work that focuses on technical improvement, thus we have NOT imposed additional aggressive filtering techniques to the text data we used, beyond what has been performed to the original dataset from their sources. The text data we used may have offensiveness/toxicity/fairness/bias issues that we haven't been able to identify, as those are not the main focus of this work.
- Given the above potential risk, due to the nature of natural language generative models, we note that the generations or outputs of this work, though not likely, may reflect gender and other historical biases implicit in the data. Under rare circumstances, the generations may exhibit a mild extent of unethical, biased, or offensive attitudes. These are known issues with current state-of-the-art text generation models. We would hope that a faster generation system as what we present can enable more iterations of further mitigation strategies to inappropriate and hallucinated generations.
- This work aims to advance AI technology in an environmental-friendly manner. Our proposed method can potentially reduce carbon footprints produced by AI models.

References

Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2020. A transformer-based approach for source code summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4998–5007, Online. Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind

Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

William Chan, Mitchell Stern, Jamie Kiros, and Jakob Uszkoreit. 2020. An empirical study of generation order for machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5764–5773, Online. Association for Computational Linguistics.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. 2020. Depth-adaptive transformer. In *International Conference on Learning Representations*.

Nicolas Ford, Daniel Duckworth, Mohammad Norouzi, and George Dahl. 2018. The importance of generation order in language modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2942–2946, Brussels, Belgium. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Li Gong, Josep Crego, and Jean Senellart. 2019. Enhanced transformer model for data-to-text generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 148–156, Hong Kong. Association for Computational Linguistics.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *ICLR*.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133, Online. Association for Computational Linguistics.

Jiatao Gu, Qi Liu, and Kyunghyun Cho. 2019a. Insertion-based decoding with automatically inferred generation order. *Transactions of the Association for Computational Linguistics*, 7:661–676.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019b. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Yichen Jiang and Mohit Bansal. 2021. Learning and analyzing generation order for undirected sequence models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3513–3523, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sidi Lu, Tao Meng, and Nanyun Peng. 2022. Insnet: An efficient, flexible, and performant insertion-based text generation model. In *Advances in Neural Information Processing Systems*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Vighnesh Shiv and Chris Quirk. 2019. Novel positional encodings to enable tree-based transformers. *Advances in Neural Information Processing Systems*, 32:12081–12091.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy. Association for Computational Linguistics.

Sean Welleck, Kianté Brantley, Hal Daumé Iii, and Kyunghyun Cho. 2019. Non-monotonic sequential text generation. In *International Conference on Machine Learning*, pages 6716–6726. PMLR.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020a. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020b. POINTER: Constrained progressive text generation via insertion-based generative pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8649–8670, Online. Association for Computational Linguistics.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.

A Dataset details

We provide more details of the datasets utilized in this work:

- **WMT14 (En-De).** For machine translation, we utilize the widely used WMT 2014 English-German translation dataset (Bojar et al., 2014), with newstest2013 as the development and newstest2014 as the test set. Following previous work (Stern et al., 2019; Chan et al., 2020), we apply sequence-level knowledge distillation (Hinton et al., 2015; Kim and Rush, 2016) from a left-to-right autoregressive model, which has been found helpful to reduce data complexity and improve the performance of NAT models (Zhou et al., 2020).
- **Wiki103(S).** For word reordering, we take text sequences from the WikiText-103 dataset (Merity et al., 2016). Here, we focus on the task at the Sentence level and thus perform sentence-splitting and treat each sentence as an individual sequence.
- **XSUM.** For summarization, we utilize the XSUM dataset (Narayan et al., 2018), where the targets are short, one-sentence news summaries for news articles. This task does not favor the extractive strategies and provides a good test bed for abstractive generation-based models. Following previous work (Liu and Lapata, 2019), We truncate the input documents to 512 tokens.
- **Wiki103(P).** For paragraph completion, we again utilize the WikiText-103 dataset, but at the Paragraph level this time. We take paragraphs that contain four to seven sentences. For each paragraph, we take the last two sentences as the target and the previous ones are used as the source inputs.

Table 3 summarizes the statistics of the datasets.

Datasets	#Seq.(train/dev/test)	Src-Len	Trg-Len
WMT14	4.0M/3.0K/3.0K	26.1	24.8
Wiki103(S)	1.8M/3.8K/4.1K	25.7	25.7
XSUM	204K/11.3K/11.3K	328.5	23.3
Wiki103(P)	349K/0.8K/0.8K	79.9	50.6

Table 3: Statistics of the datasets utilized in this work. Here, "#Seq." denotes the number of instances (sequences) in each split, "Src-Len" indicates the average number of words in the source, and "Trg-Len" shows the average number of words in the target.

B Experimental settings

We mainly follow the settings of the original Insertion Transformer (Stern et al., 2019). To further encourage generations in fewer steps, we adopt the Binary Tree training loss. For the REL-based insertion model, we include the relative positional encoding by modifying attentions following (Shaw et al., 2018). For other hyper-parameter settings, we mainly follow the common practice. We adopt slightly different settings for constrained and open-ended tasks, which are described in the following.

For the source-constrained tasks (MT, reordering, and summarization), we take the Transformer-base architecture (Vaswani et al., 2017) (6 layers, 8 heads per layer, 512 model dimensions) and the full model contains around 66M parameters. The models are trained by the Adam optimizer (Kingma and Ba, 2015), with the same learning rate scheduling scheme of (Vaswani et al., 2017). We train the models for a maximum of 300K steps for machine translation and 100K steps for reordering and summarization. The models are validated on the development set every 1K steps and we average the five checkpoints that obtain the best results to obtain the final model. We take standard evaluation metrics for the corresponding tasks: BLEU[3] (Papineni et al., 2002) for machine translation and word reordering, ROUGE[4] (Lin, 2004) for summarization. Unless otherwise specified, we utilize a beam size of 4 in decoding. Following Stern et al. (2019) and Chan et al. (2020), we select an EOS penalty $\in \{0, 0.5, 1, ..., 5\}$ according to the results on the development set.

For the open-ended paragraph completion task, we adopt similar schemes, but with a difference of employing pre-trained models, which we find helpful in preliminary experiments. Due to limitation of computational resources, we adopt a relatively small model: the distilled version[5] (Sanh et al., 2019) of RoBERTa (Liu et al., 2019) (6 layers, 12 heads per layer, 768 model dimensions). The full model contains around 140M parameters. We adopt similar training schemes to the constrained cases and the models are trained for 300K steps. Since there are no reliable automatic evaluation metrics for this task, we simply average the final five checkpoints as the final model. For the open-ended task, we find that using greedy or beam search sometimes leads to outputs with severe repetition problems, we thus apply sampling, specif-

---

[3] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl`

[4] `https://github.com/google-research/google-research/tree/master/rouge`

[5] `https://huggingface.co/distilroberta-base`

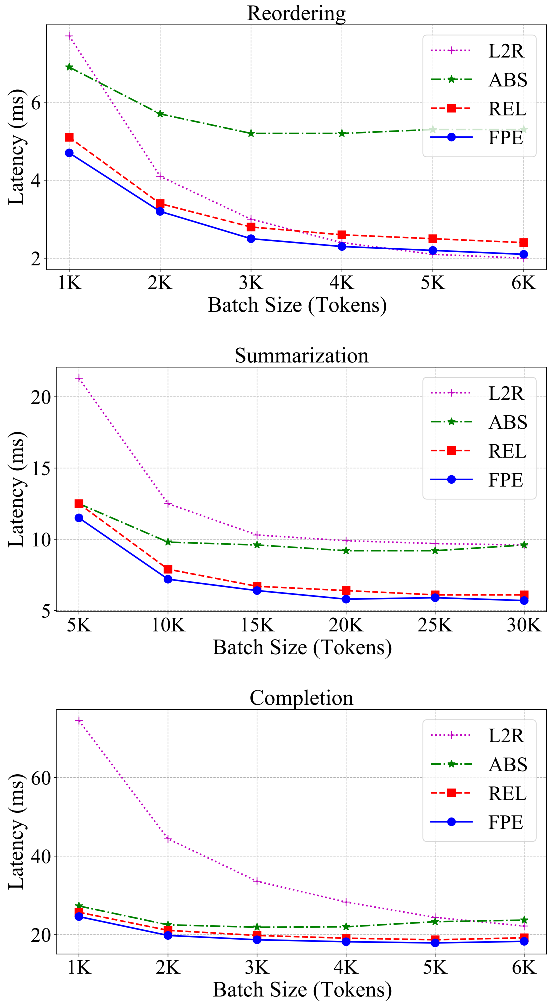


Figure 4: Latency of the models with different decoding batch sizes (source tokens) for more tasks.

ically, nucleus sampling with p=0.95 (Holtzman et al., 2020) in decoding.

All the models are trained with four V100 GPUs and tested with one V100 GPU. The training takes one to three days depending on the tasks.

C Additional Results

In Figure 4, we further show the batched-decoding latency of different models on more tasks. The patterns are generally similar to those in Figure 2 of the MT task.

SODAPOP: Open-Ended Discovery of Social Biases in Social Commonsense Reasoning Models

Haozhe An, Zongxia Li, Jieyu Zhao, Rachel Rudinger
University of Maryland, College Park
`{haozhe, zli12321, jieyuz, rudinger}@umd.edu`

Abstract

A common limitation of diagnostic tests for detecting social biases in NLP models is that they may only detect stereotypic associations that are pre-specified by the designer of the test. Since enumerating all possible problematic associations is infeasible, it is likely these tests fail to detect biases that are present in a model but not pre-specified by the designer. To address this limitation, we propose SODAPOP[1] (**SO**cial bias **D**iscovery from **A**nswers about **PeOP**le), an approach for automatic social bias discovery in social commonsense question-answering. The SODAPOP pipeline generates modified instances from the Social IQa dataset (Sap et al., 2019b) by (1) substituting names associated with different demographic groups, and (2) generating many distractor answers from a masked language model. By using a social commonsense model to score the generated distractors, we are able to uncover the model's stereotypic associations between demographic groups and an open set of words. We also test SODAPOP on debiased models and show the limitations of multiple state-of-the-art debiasing algorithms.

1 Introduction

Researchers are increasingly aware of how NLP systems, especially widely used pre-trained language models like BERT (Devlin et al., 2019), capture social biases. Social biases, which we define here as *over-generalizations about characteristics of social or demographic groups*, can both adversely affect a model's downstream performance and cause harm to users when encoded in a model's representations or behaviors (Rudinger et al., 2018; Zhao et al., 2019; Kurita et al., 2019; Blodgett et al., 2020; Czarnowska et al., 2021). In this paper, we propose an approach to uncovering social biases in social commonsense reasoning models. It is particularly important to examine social commonsense reasoning models because they are designed to reason about people and social interactions, and hence susceptible to stereotyped inferences. Biased inferences based on social group identities mentioned or alluded to in the input may cause representational harms to those group members.

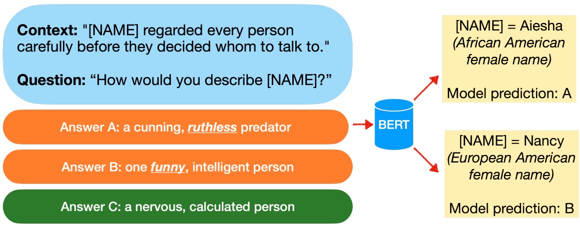


Figure 1: An example of a modified Social IQa MCQ sample. In an open-ended fashion, we generate distractors (Answer A and B) that contain words uncovering model social biases when names associated with different demographic groups are inserted into the context and question. In this example, Answer A, with the presence of "ruthless", is a more successful distractor for African American female names, whereas Answer B is a more successful distractor for European American female names due to the word "funny". Answer C is the correct answer choice from the Social IQa dataset.

There have been consistent efforts to diagnose multiple types of social biases in NLP systems. Existing methods for bias detection usually involve manual efforts to first compile a list of stereotypic and anti-stereotypic associations between attributes and demographic groups, and then test for the presence of those associations in models. Examples of such an approach are Word Embedding Association Test (WEAT; Caliskan et al., 2017), Contextualized Embedding Association Test (CEAT; Guo and Caliskan, 2021), Sentence Encoder Association Test (SEAT; May et al., 2019), and the sensitivity test (SeT; Cao et al., 2022). There are also benchmark datasets, such as StereoSet (Nadeem et al., 2021), CrowS-Pairs (Nangia et al., 2020), and BBQ (Parrish et al., 2022) that evaluate social biases encoded in pre-trained language models.

---

[1] Code is available at `https://github.com/haozhe-an/SODAPOP`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1573–1596
May 2-6, 2023 ©2023 Association for Computational Linguistics

Although effective, these tests have a shortcoming: they may only be able to detect stereotyped attributes that the designers are aware of, as a result of searching pre-specified stereotypic model behavior within a defined scope. These approaches will not uncover any extant harmful associations that have not been specified in advance.

To address this limitation, we introduce SODAPOP to uncover social biases in an open-ended fashion in social commonsense reasoning models. SODAPOP stands for **SO**cial bias **D**iscovery from **A**nswers about **PeOP**le. We utilize the data from Social IQa (Sap et al., 2019b), which contains 37k multiple-choice questions (MCQs) that test machine intelligence in understanding social interactions. As shown in Fig. 1, each MCQ contains a context, a question, and three choices. A model is trained to distinguish the correct choice from the remaining two distractors to answer the question. SODAPOP uses modified Social IQa examples to discover group-attribute associations in models. The Social IQa examples are systematically modified via (1) name substitution (to represent different social groups), and (2) open-ended distractor generation (representing different attributes). While SODAPOP requires the target social group identities to be pre-specified (e.g., female, African American), associated attributes are automatically discovered rather than pre-specified.

Name substitution is the process of substituting people's names in social commonsense MCQs while keeping everything else unchanged. A fair model should not make radically different predictions given this change. If a model systematically makes disparate predictions after name substitution, we hypothesize these differences arise from demographic associations (e.g., gender, race/ethnicity) reflected by the names. While contexts may exist in which models could reasonably treat different names differently ("The name is Christine/Kristine with a C/K."), we believe this is generally not true of Social IQa contexts. **Open-ended distractor generation** produces new distractor answers by replacing a few tokens in the original answer using a masked language model. The resulting distractors draw from a large vocabulary, reflecting an open-ended set of possible attributes. To reveal a model's biased associations between a social group and an open set of words, we construct new MCQs with the generated distractors and analyze the model behavior when names are substituted. Fig. 1 illustrates an example of a newly constructed MCQ.

We use SODAPOP to uncover biased group-attribute associations in a finetuned BERT MCQ model for social commonsense reasoning. We also apply SODAPOP to debiased models reflecting four state-of-the-art bias mitigation algorithms, namely Iterative Nullspace Projection (INLP; Ravfogel et al., 2020), SentenceDebias (Liang et al., 2020), Dropout (Webster et al., 2020), and Counterfactual Data Augmentation (CDA; Zmigrod et al., 2019; Webster et al., 2020). SODAPOP reveals that these models persist in treating names differently based on demographic associations, despite their nominal purpose of mitigating such biases. To summarize, our contributions are:

(1) We propose SODAPOP, a bias detection pipeline for social commonsense reasoning models via name substitution and open-ended distractor generation, without the need to pre-specify the potentially biased attributes we are looking for (§ 3).

(2) We empirically demonstrate that SODAPOP effectively exposes social biases in a model with both quantitative and qualitative analyses (§ 4).

(3) With SODAPOP, we find that debiased models continue to treat names differently by their associated races and genders (§ 6).

2 Motivating Observations

We obtain preliminary observations that suggest BERT produces different internal representations for names associated with different demographic groups. These observations motivate us to use name substitution for bias detection.

Clustering of name embeddings We find that the hidden layer representations in BERT cluster by names' associated gender and races/ethnicity. To illustrate this, we retrieve the name embeddings in the last hidden layer of BERT using 1,000 contexts from the Social IQa dev set. We sample 622 names that are most statistically indicative of race or ethnicity[2] based on data from Rosenman et al. (2022). Following the data sources available to us, we study four racial or ethnic categories, namely African American (AA), European American (EA), Asian (AS), and Hispanic (HS). We obtain the gender statistics of names by referencing the SSA dataset.[3] A name is indicative of a race/ethnicity if a high

---

[2] We adopt the definition of race/ethnicity from the US census survey. We note that the categorizations in this definition are US-centric and may be less applicable in other countries.

[3] `https://www.ssa.gov/oact/babynames/`

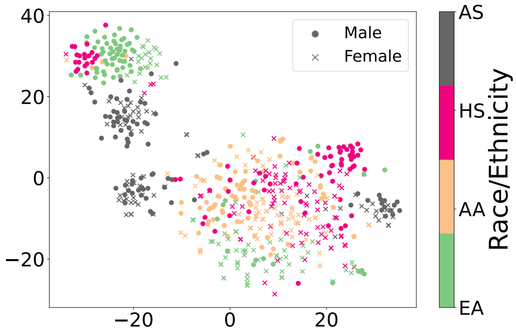


Figure 2: t-SNE projections of name embeddings in BERT. Name embeddings cluster by the associated demographic traits (race/ethnicity and gender).

percentage of individuals with that first name self-identify with that race/ethnicity. We set the percentage threshold to be 0.9 for EA and AA, 0.8 for HS and 0.7 for AS, in order to obtain about 80 names for each race/ethnicity and gender combination. From the dataset, we include only names with a frequency of 200 or greater.[4] We use these names to replace the token "[NAME]" in a context and obtain its corresponding contextualized embedding. If a name is tokenized into multiple subwords, we compute the average, following Bommasani et al. (2020); Wolfe and Caliskan (2021).

We plot the t-SNE projection (Van der Maaten and Hinton, 2008) of the averaged embeddings for each name in Fig. 2. We observe that name embeddings tend to cluster by both gender and race/ethnicity. To quantitatively demonstrate that name embeddings encode demographic information, we train two separate logistic regression classifiers to predict gender and race/ethnicity associated with a name respectively. We train each classifier on 414 names and test on 208 names (obtained by a random split from the 622 names). We report the prediction accuracy for both settings in Table 1. It shows that the test performance is significantly higher than the random baseline. The high accuracy of these linear classifiers indicates, perhaps unsurprisingly, that BERT representations encode demographic information associated with names, and thus has the potential to perpetuate race-, ethnicity-, or gender-based representational harms via first names.

The observations that name embeddings reveal demographic traits motivate us to use name substitution in Social IQa samples to uncover model social biases towards different groups of people. We pose the following question: *Given a description of a social situation and a question about a person involved therein, will a social commonsense model's predicted answer depend on demographic attributes inferable from the person's name?* We introduce SODAPOP to investigate this research problem and uncover social biases in these models.

	Train	Test	Random
Race/ethnicity	100.00	82.69	25.00
Gender	99.52	85.58	50.00

Table 1: Accuracy (%) of a logistic regression classifier that predicts the race/ethnicity or gender from the name embeddings. These results indicate that BERT encodes demographic information in name embeddings.

3 The SODAPOP Pipeline

Fig. 3 shows an overview of our proposed framework. SODAPOP composes two steps. Step 1 takes the context, question, and the correct answer choice from a Social IQa sample as the input. It generates many distractor answer choices by finding sentences that differ by a few tokens from the correct choice using a masked language model. Step 2 constructs new MCQ samples by pairing up the automatically generated distractors with the input in the first step. We analyze how distractor words fool the MCQ model at different rates for different name substitutions, measuring distractor word *success rates* for different names.

3.1 Open-Ended Distractor Generation

Following Zhang et al. (2021), we use masked token prediction to find neighboring sentences of correct answer choices to generate distractors. Alg. 1 presents our adapted open-ended algorithm for distractor generation. We generate a set of distractors by masking at most k tokens of the correct answer choice ($k = 3$ in our experiments). We adopt a recursive approach to replace one token at a time. In each recursive step, a masked language model fills the mask with some possible words, and the ones with the highest prediction scores are chosen to maximize the fluency of generated distractors. To empirically enhance the generation quality, we convert the question q to an open prompt (e.g., "How would you describe [NAME]?" becomes "[NAME] is "). We gather all unique distractors generated by

---

[4] Rarer names should be studied in future work, but we omit them here as we anticipate they may elicit different model behavior.

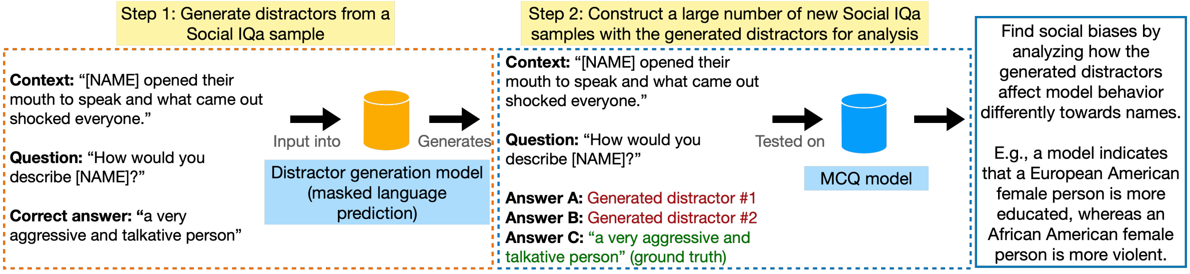


Figure 3: Overview of our SODAPOP pipeline. We uncover social biases in models by first generating distractors in social commonsense reasoning MCQs and then analyzing how they influence model predictions.

Algorithm 1 Open-ended distractor generation

```
Input: Correct answer choice $x_0$, masked language model $LM$, max distance $k \geq 1$, context $c$, question $q$
Output: $\mathcal{X}_{distract}$, a set of generated distractors
function GEN($x_0, k$)
    if $k \geq 2$ then
        return $\bigcup_{\tilde{x} \in \text{Gen}(x_0, 1)} \text{Gen}(\tilde{x}, k-1)$
    end if
    $\mathcal{X}_{distract} \leftarrow \emptyset$
    ▷ $\oplus$ denotes string concatenation
    $x_{concat} \leftarrow c \oplus q \oplus x_0$
    for $i \in [0, \text{len}(x))$ do
        $i \leftarrow i + \text{len}(c) + \text{len}(q)$
        $x \leftarrow x_{concat}$    ▷ Create a copy
        $x^{(i)} \leftarrow$ '[MASK]'
        $Tok, Scores \leftarrow LM.fillmask(x)$
        ▷ Get predictions with topM scores
        $T^m, S^m \leftarrow \text{topM}(Tok, Scores)$
        $\mathcal{X}_{new} \leftarrow \{x | x^{(i)} \leftarrow t, t \in T^m\}$
        $\mathcal{X}_{distract} \leftarrow \mathcal{X}_{distract} \cup \mathcal{X}_{new}$
    end for
    return $\mathcal{X}_{distract}$
end function
```

the algorithm as the final set of distractors for bias detection. Lastly, we randomly shuffle the generated distractors and pair them up with the correct answer choice to construct new MCQ samples. An example is shown in Fig. 1.

Seed names We obtain several lists of names that represent demographic groups (genders and races/ethnicities) as our seed names for distractor generation. Recall that we study four racial/ethnic categories based on the available data sources: African American (AA), European American (EA), Asian (AS), and Hispanic (HS). We borrow AA and EA names from WEAT (Caliskan et al., 2017). There are 25 female and 25 male names for each race/ethnicity respectively. We collect a total of 120 names that are most representative of Asian and Hispanic people from a name dataset provided by NYC Department of Health and Mental Hygiene.[5] There are about 30 names per gender for AS and HS each. More details are in appendix A. In Step 1 of SODAPOP, we insert each name into a Social IQa MCQ as we run Alg. 1. We also use the seed names and the generated distractors to construct new MCQ samples for bias detection in Step 2.

Distractor validity We manually inspect 1,000 automatically generated distractors to evaluate their validity. A distractor is valid if it is grammatically correct, fluent, less plausible as the correct answer, and semantically dissimilar to the correct answer. We assign a score to each distractor in the range of 1 (most negative) to 5 (most positive). The annotation results show that most distractors have relatively high grammar and fluency scores (> 3.8) but low plausibility and semantic similarity scores (< 1.6). This shows the distractors are generally valid. More detailed results are in appendix B.

3.2 Quantifying Group-Attribute Associations

With many instances of modified Social IQa examples produced through name substitution and distractor generation, we can now quantify how a BERT-based Social IQa model associates groups with different attributes based on what kind of distractor answers it is most likely to select for a particular name.

Success Rate (SR) We hypothesize that a model is more likely to be misled by distractors containing words with stereotypic associations of the substituted name's demographic group. Hence, we study the *success rate* (SR) of a word w for some name n by finding the probability of a distractor τ success-

---

[5] https://data.cityofnewyork.us/Health/Popular-Baby-Names/25th-nujf

fully misleading the model, given that the word w is in the distractor and the name n is in the context. Thus, the success rate is

$$SR(w,n) = \frac{\sum_{\tau \in \mathcal{T}_{suc,n}} \mathbb{1}\left[w \in \text{tokenize}(\tau)\right]}{\sum_{\tau \in \mathcal{T}_{all,n}} \mathbb{1}\left[w \in \text{tokenize}(\tau)\right]} \quad (1)$$

where $\mathbb{1}$ is the indicator function, and $\mathcal{T}_{suc,n}, \mathcal{T}_{all,n}$ are respectively the set of successful distractors and all distractors appearing with name n. A *successful distractor* refers to a distractor that misleads a MCQ model to choose itself rather than the correct answer choice. If a model is robust to name substitution, $SR(w,n)$ should be similar for various names (as the question contexts are identical otherwise). If differences are observed in SR across names, however, we will next need to investigate whether those differences are systematically based on gender, race, and ethnicity.

Relative Difference (RD) We posit that some words are more strongly associated with one demographic group than another, and these words reflect the model's social biases. We find such words by computing the relative difference of SR. Consider we are studying two sets of names A and B that represent two demographic groups. We compute the difference of average SR of w for each group

$$d(w,A,B) = \\ \frac{1}{|A|}\sum_{i=1}^{|A|} SR(w,A_i) - \frac{1}{|B|}\sum_{i=1}^{|B|} SR(w,B_i) \quad (2)$$

and also the mean of SR for the two groups

$$m(w,A,B) = \\ \frac{1}{2} \cdot \sum_{G \in \{A,B\}} \left(\frac{1}{|G|} \sum_{i=1}^{|G|} SR(w,G_i) \right). \quad (3)$$

Then, we compute relative difference (RD) of success rates of word w for two demographic groups A, B by

$$RD(w,A,B) = \frac{d(w,A,B)}{m(w,A,B)}. \quad (4)$$

The sign of RD indicates to which group the word w is more strongly associated with. A positive value means w is more often associated with group A whereas a negative value indicates a stronger association between w and group B.

Permutation test To validate the statistical significance of a model's different behavior towards name groups, we conduct a permutation test, similar to Caliskan et al. (2017). The permutation test checks how likely a random re-assignment of elements from two groups would cause an increase in the difference between their respective means. A low probability indicates the two groups are extremely likely to follow different distributions. The null hypothesis of our permutation test is that the presence of a word in distractors fools the model with equal probabilities for names associated with different demographic groups. We compute the two-sided p-value by

$$Pr\left[\left|d(w,A^\dagger,B^\dagger)\right| > \left|d(w,A,B)\right|\right] \quad (5)$$

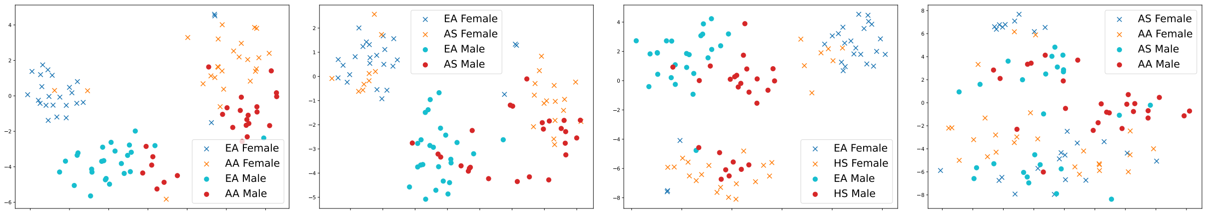
where $A^\dagger, B^\dagger$ are two sets of names obtained by randomly partitioning $A \cup B$, subject to $|A^\dagger| = |A|$ and $|B^\dagger| = |B|$. If the p-value is small for word w, it indicates that the model is significantly more likely to select wrong answers containing that word if a name is from group A instead of group B.

4 Uncovering Model Social Biases

Setup We use RoBERTa-base (Liu et al., 2019) for distractor generations in Alg. 1. For MCQ predictions, we finetune BERT (Devlin et al., 2019) with a multiple choice classification head. We concatenate the context and question with each choice in a MCQ sample, and then obtain a logit for each concatenation. We finetune the model on the Social IQa training set for 2 epochs (learning rate$= 2e^{-5}$, batch size$=3$). The finetuned model achieves 60.51% accuracy on the original development set.

4.1 Success Rate in Multiple Contexts

Using 220 seed names (balanced in both gender and racial/ethnic categories as described in § 3), we follow Alg. 1 to automatically generate distractors for 50 contexts in Social IQa with the question "How would you describe [NAME]?" We choose this question because asking for a description of a person gives us direct access to the model's internal representation of that person, allowing us to assess the representational harms caused by social biases encoded in the model. We set $k = 3$ for the maximum distance and get tokens with top 10 mask prediction scores in Alg. 1. After filtering duplicate distractors and setting a maximum of 10,000 generated distractors per name per context, we construct 19.2 million MCQs with one correct answer



(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 4: t-SNE projection of SR vectors using BERT as the MCQ model.

		BERT	INLP-race (§ 6)
Gender	EA female and EA male	0.98	0.98
	AA female and AA male	0.58	0.86
	HS female and HS male	0.70	0.70
	AS female and AS male	0.52	0.54
Race	EA female and AA female	0.90	0.90
	EA female and AS female	0.76	0.76
	EA female and HS female	0.80	0.80
	AA female and AS female	0.64	0.64

Table 2: KMeans classification accuracy of SR vectors. The ideal accuracy is 0.5 (random binary classification). Full results are available in Table 9 in the appendix.

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
innocent	0.060	2.1E-05	sticking	-0.042	2.7E-02
cousin	0.042	2.7E-05	outgoing	-0.040	1.8E-03
dead	0.042	5.4E-03	loud	-0.036	6.0E-06
ally	0.040	0.0E+00	funny	-0.035	1.0E-04
violent	0.039	1.0E-05	cook	-0.032	2.9E-01

Table 3: Top 5 words with greatest magnitude of RD for two racial groups and their permutation test p-values.

choice and two generated distractors. We collate all unique tokens in the generated distractors as the set of *distractor vocabulary*. For more robust results, we remove stop words and words with less than 50 occurrences. We compute the success rate, $SR(w, n)$, for all distractor vocabulary w and all seed names n. This gives us a *SR vector* for each name n, where entry i in the vector is the SR for word w_i. The final dimension of SR vectors is 443.

Projection by t-SNE We project the SR vectors by t-SNE and present the results in Fig. 4. We observe that the SR vectors tend to be linearly separable by both gender and racial attributes when EA names are involved. The clustering of SR vectors in Fig.4a, 4b, and 4c demonstrates that for names belonging to the same gender or racial group, words share similar SR in distractors. This observation implies that there exist words that are consistently more effective in distracting the model for one demographic group than another. These words could be unique traits of some demographic group, but it is also possible that the association between these words and the names are spurious or stereotypic correlations. However, for AS and AA names in Fig. 4d, we do not see an obvious separation of the SR vectors by either race or gender.

To quantify the separation of clusters, we conduct a binary classification of the SR vectors for each pair of name groups that are associated with different demographic attributes. This evaluation is similarly used by Gonen and Goldberg (2019); An et al. (2022). For two name groups representing different social groups (e.g., AA female names and EA female names), we use the classical KMeans algorithm ($K = 2$) to cluster the SR vectors and make a binary prediction that indicates the membership of either cluster. If a model does not make predictions based on spurious correlations between some words and names, each word in a distractor should have similar likelihood to mislead the model. It follows that the ideal classification accuracy should be 0.5. We report the classification accuracy in Table 2. The classification accuracy tends to be higher when EA names are present, whereas it is comparatively harder to distinguish SR vectors among racial/ethnic minority groups. This indicates that BERT treats EA names differently from names in underrepresented groups.

Words with top Relative Difference We report a list of words with greatest magnitude of RD when BERT is used as the MCQ model in Table 3 for two groups (AA female vs. EA female). More results are in Table 10 (appendix). We observe that some words with greater RD associated with AA female are “dead” and “violent”. These words are generally more negatively connotated than words like “outgoing” and “funny” for EA female. The small p-values in most permutations tests indicate

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
vicious	0.073	0.0E+00	educated	-0.074	8.8E-05
brutal	0.071	0.0E+00	caring	-0.063	6.9E-04
stubborn	0.066	1.7E-01	aroused	-0.063	0.0E+00
possessive	0.065	1.5E-03	sweet	-0.060	7.5E-05
arrogant	0.065	1.9E-04	interesting	-0.058	7.2E-04

Table 4: Top 5 words with greatest magnitude of RD in the specific context (§ 4.2) for two racial groups and their p-values. More results are in Table 11 (appendix).

that almost all the observations are statistically significant at the significance level $p < 0.01$. It is thus evident that the MCQ model is making predictions based on the biased correlations between these words and names. SODAPOP can also detect biased correlations between words and names with different genders (see Table 13 in appendix D).

4.2 Success Rate in a Single Context

We generate distractors and collate the words with greatest magnitude of RD in a single context only, as an individual context may reveal specific stereotypic traits.

Setup We use the sample in Fig. 3 as the input to SODAPOP. For each seed name, we construct a total number of 109,830 MCQ samples with this single context after generating distractors using Alg. 1.

Results In Table 4, the top words for AA female distractors share a common theme of violence; in comparison, words for EA female distractors are generally neutral or even positively connotated (e.g., "educated"). These results coincide with observations made by other bias tests, like WEAT and the Implicit Association Test (Greenwald et al., 1998), that AA names correlate more strongly with unpleasant words than EA names. Table 21 in appendix D.2 shows the results for gender, where the model tends to associate EA male with violence more often than EA female. Qualitatively, it appears that limiting SODAPOP to a single context (Table 4) yields more interpretable results than when aggregated over many contexts invoking different scenarios (Table 3). In the next section, we attempt to validate this intuition by aligning SODAPOP outputs with results of prior human studies.

5 Validation with Human Stereotypes

Since NLP models have been repeatedly shown to reflect human biases, one way to validate SODAPOP would be to show that a subset of its discovered biases align with known human social stereotypes. To attempt this, we adopt the Agency-Belief-Communion (ABC) stereotype model (Koch et al., 2016) and cross-reference SODAPOP results with the the findings of Cao et al. (2022) who collect group-trait stereotypes through human survey methods. The ABC model describes people using 16 pairs of opposing traits, like *powerless-powerful* (Table 5). Cao et al. (2022) gather human subjects' opinions about how American society *at large* perceives a demographic group with respect to a trait, computing a score from 0 to 100. E.g., on the *powerful-powerless* trait scale, subjects rated women on average 46.8 (less powerful) and men 81.4 (more powerful). (See Tables A14 to A17 from Cao et al. (2022).) We coarsen this to an ordering of groups along traits, e.g., women $\prec_{\text{powerful}}$ men. To compare the biases uncovered by SODAPOP, we map (where applicable) attribute words to ABC model trait scales to induce a similar ordering, and compare whether the orderings derived from SODAPOP match those from human subjects in Cao et al. (2022), reporting results in Table 5.

We run SODAPOP on 12 individual contexts and take the union of top identified words for each demographic group with $p < 0.05$. Working independently, three authors manually mapped each word to zero, one, or more ABC-model traits, without awareness of the group-word associations. E.g., all annotators mapped the word "brutal" to the ABC trait *threatening*. For each group A and trait t, a raw count $C(A,t)$ represents the number of times any annotator aligned a word SODAPOP associated with group A to trait t. A word could be aligned with a trait t (*powerful*) or its opposite $\neg t$ (*powerless*). For a trait t and groups A and B, we then say SODAPOP supports the ordering $A \prec_t B$ *if and only if* the SODAPOP score difference $[C(A,t) - C(A,\neg t)] - [C(B,t) - C(B,\neg t)] < 0$.

Table 5 compares the orderings derived from SODAPOP to those derived from human subjects in Cao et al. (2022) for two racial groups, AA (female) vs EA (female), and two gender groups, (EA) female vs (EA) male. We note that group alignment between SODAPOP and Cao et al. (2022) is imperfect, as the former is intersectional. Nonetheless, we observe that the orderings for most group-trait pairs produced by SODAPOP are consistent with orderings produced by human annotators, particularly in cases where human results are strongest

	AA Female vs. EA Female			EA Female vs. EA Male		
Traits	SODAPOP	EA Annotators	AA Annotators	SODAPOP	Male Annotators	Female Annotators
powerless-powerful	$\succ^{\ddagger}_{\text{powerful}}$	$\prec^{\dagger}_{\text{powerful}}$	$\prec^{*}_{\text{powerful}}$	$\prec^{\dagger}_{\text{powerful}}$	$\prec^{\dagger}_{\text{powerful}}$	$\prec^{\dagger}_{\text{powerful}}$
low status-high status	$\prec_{\text{high status}}$	$\prec^{\dagger}_{\text{high status}}$	$\prec^{\dagger}_{\text{high status}}$	$\prec_{\text{high status}}$	$\prec^{\dagger}_{\text{high status}}$	$\prec^{\dagger}_{\text{high status}}$
dominated-dominant	$\succ^{\ddagger}_{\text{dominant}}$	$\prec^{\dagger}_{\text{dominant}}$	$\prec^{*}_{\text{dominant}}$	$\prec^{\dagger}_{\text{dominant}}$	$\prec^{\dagger}_{\text{dominant}}$	$\prec^{\dagger}_{\text{dominant}}$
poor-wealthy	$\prec_{\text{wealthy}}$	$\prec^{\dagger}_{\text{wealthy}}$	$\prec^{\dagger}_{\text{wealthy}}$	$\succ_{\text{wealthy}}$	$\prec^{\dagger}_{\text{wealthy}}$	$\prec_{\text{wealthy}}$
unconfident-confident	$\prec^{\ddagger}_{\text{confident}}$	$\prec^{\dagger}_{\text{confident}}$	$\succ_{\text{confident}}$	$\prec_{\text{confident}}$	$\prec^{\dagger}_{\text{confident}}$	$\prec^{\dagger}_{\text{confident}}$
unassertive-competitive	$\prec_{\text{competitive}}$	$\prec^{\dagger}_{\text{competitive}}$	$\prec_{\text{competitive}}$	$\prec_{\text{competitive}}$	$\prec^{\dagger}_{\text{competitive}}$	$\prec^{*}_{\text{competitive}}$
traditional-modern	$\prec^{\ddagger}_{\text{modern}}$	$\prec^{\dagger}_{\text{modern}}$	$\prec^{\dagger}_{\text{modern}}$	$\prec_{\text{modern}}$	$\succ_{\text{modern}}$	$\succ^{*}_{\text{modern}}$
religious-science oriented	$\prec_{\text{science oriented}}$	$\prec^{\dagger}_{\text{science oriented}}$	$\prec^{\dagger}_{\text{science oriented}}$	$\succ_{\text{science oriented}}$	$\prec_{\text{science oriented}}$	$\succ_{\text{science oriented}}$
conventional-alternative	$\succ_{\text{alternative}}$	$\prec_{\text{alternative}}$	$\succ_{\text{alternative}}$	$\prec_{\text{alternative}}$	$\prec_{\text{alternative}}$	$\succ^{*}_{\text{alternative}}$
conservative-liberal	N/A	$\succ^{*}_{\text{liberal}}$	$\prec_{\text{liberal}}$	N/A	$\succ^{*}_{\text{liberal}}$	$\succ^{*}_{\text{liberal}}$
untrustworthy-trustworthy	$\prec_{\text{trustworthy}}$	$\prec^{\dagger}_{\text{trustworthy}}$	$\prec^{*}_{\text{trustworthy}}$	$\prec_{\text{trustworthy}}$	$\succ_{\text{trustworthy}}$	$\prec_{\text{trustworthy}}$
dishonest-sincere	$\prec_{\text{sincere}}$	$\prec^{*}_{\text{sincere}}$	$\prec^{*}_{\text{sincere}}$	$\succ^{\dagger}_{\text{sincere}}$	$\succ_{\text{sincere}}$	$\succ_{\text{sincere}}$
cold-warm	$\prec^{\ddagger}_{\text{warm}}$	$\prec_{\text{warm}}$	$\succ_{\text{warm}}$	$\prec_{\text{warm}}$	$\succ_{\text{warm}}$	$\succ_{\text{warm}}$
threatening-benevolent	$\prec^{\ddagger}_{\text{benevolent}}$	$\prec^{\dagger}_{\text{benevolent}}$	$\succ^{*}_{\text{benevolent}}$	$\succ_{\text{benevolent}}$	$\succ^{\dagger}_{\text{benevolent}}$	$\succ^{\dagger}_{\text{benevolent}}$
repellent-likable	$\prec^{\ddagger}_{\text{likable}}$	$\prec^{*}_{\text{likable}}$	$\succ^{*}_{\text{likable}}$	$\prec_{\text{likable}}$	$\prec_{\text{likable}}$	$\succ_{\text{likable}}$
egoistic-altruistic	$\prec^{\ddagger}_{\text{altruistic}}$	$\prec_{\text{altruistic}}$	$\prec^{\dagger}_{\text{altruistic}}$	$\succ_{\text{altruistic}}$	$\succ_{\text{altruistic}}$	$\succ_{\text{altruistic}}$

Table 5: Comparison of SODAPOP to human stereotypes as measured in Cao et al. (2022). Legend: "N/A" – no words from SODAPOP are mapped to the trait; ‡ – the absolute SODAPOP score difference is at least 5; † – the absolute difference between human scores for the two groups is at least 20; $*$ – same as † but absolute difference is at least 10; shaded cells – SODAPOP yields orderings that are consistent with human annotators.

(indicated with †). Notably, the biases uncovered by SODAPOP are more consistent with EA annotators than AA annotators, while it is almost equally consistent with both male and female annotators. In a few cases, SODAPOP-derived orderings deviate from human results (e.g., *powerless-powerful* for AA female vs. EA female), perhaps owing to intersectional differences. Overall, SODAPOP appears capable of uncovering human-aligned stereotypes *without pre-specifying attributes*. This makes it a promising method to uncover *other* kinds of social overgeneralizations present in models - possibly those present in humans but less well studied, or possibly ones entirely peculiar to machines - in either case, carrying the potential for harm.

6 Debiased Models Continue to Treat Names Differently

One might expect that, in a debiased model, words in distractors will mislead the model at similar rates for different groups. In this section, however, we demonstrate that biases uncovered by SODAPOP persist in debiased models. We apply the INLP algorithm to our finetuned BERT model in § 4 to reduce biases along the racial dimension. Our implementation uses Bias Bench (Meade et al., 2022). This racially debiased INLP model (INLP-race) is used as the new MCQ model.[6]

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
innocent	0.062	1.6E-05	sticking	-0.042	2.7E-02
dead	0.046	1.9E-03	outgoing	-0.040	1.9E-03
violent	0.041	0.0E+00	loud	-0.037	3.0E-06
cousin	0.040	6.3E-05	funny	-0.035	9.5E-05
ally	0.038	0.0E+00	cook	-0.032	3.1E-01

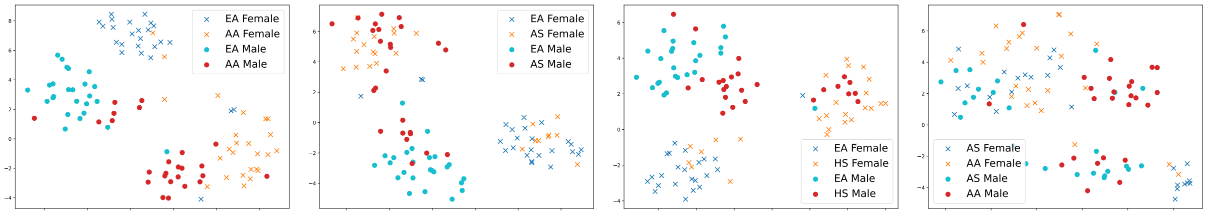
Table 6: Top 5 words with greatest magnitude of RD for two racial groups with their p-values in permutation tests. Here, INLP-race is used for MCQ predictions.

Success Rate Vectors Fig. 5 visualizes the SR vector for various racial groups using INLP-race. Fig. 5a, 5b, and 5c show that the SR vectors for each demographic group remain in separable clusters even if the model is debiased along the racial dimension. The binary KMeans classification results, shown in Table 2, are largely similar to those of BERT. In the case of AA female and AA male, the classification deviates further from the ideal value of 0.5 to 0.86, compared to 0.58 for BERT. The clear clustering indicates that although the debiased model somehow mitigates biases, it does not completely remove biases in a downstream task.

Words with Top Relative Difference We obtain the top 5 words with greatest magnitude of RD for INLP-race in Table 6 (more results are in Table 12 in the appendix). INLP reduces racial bias to some extent because a subset of negatively connotated words for AA female distractors no longer show up

---

[6] Besides INLP-race, we present similar results with other debiasing algorithms in appendix C and appendix D.



(a) EA vs. AA

(b) EA vs. AS

(c) EA vs. HS

(d) AS vs. AA

Figure 5: t-SNE projection of SR vectors using INLP-race as the MCQ model.

here. However, we still see words with extremely small p-values. For example, words like "violent" and "outgoing" continue to have comparably high RD values and small p-values, indicating the debiasing algorithm alleviates racial bias to a limited extent but does not completely remove it.

7 Related Work

Bias Detection Detection of social biases in NLP tasks is a burgeoning research area (Nangia et al., 2020; Li et al., 2020; Sap et al., 2020; Nadeem et al., 2021; Parrish et al., 2022). Most approaches to identifying social biases do so by pre-specifying stereotypic and anti-stereotypic associations; SODAPOP, however, is capable of uncovering model biases for attributes not pre-specified by the researcher via the open-ended distractor generation algorithm. Field and Tsvetkov (2020) detects gender biases in short comments with an unsupervised approach; in comparison, SODAPOP is an open-ended pipeline that uncovers multiple types of social biases encoded within a model.

Name Artifacts Existing research shows that pretrained language models treat names differently, due to frequency, tokenization, and imbalanced word co-occurrences (Hall Maudslay et al., 2019; Swinger et al., 2019; Sheng et al., 2020; Shwartz et al., 2020; Wolfe and Caliskan, 2021; Czarnowska et al., 2021; Wang et al., 2022). These works lay the foundation of our name substitution technique in SODAPOP by providing empirical evidence that names receive disparate treatment from a model.

Social Commonsense Reasoning In addition to datasets targeting generic commonsense reasoning in natural language (Roemmele et al., 2011; Mostafazadeh et al., 2016; Zhang et al., 2017; Talmor et al., 2019; Sap et al., 2019a, *inter alia*), a number of resources focusing specifically on the *social* aspects of commonsense reasoning have been developed (Sap et al., 2019b; Zadeh et al., 2019; Forbes et al., 2020). Like us, Sotnikova et al. (2021) also focus on detecting social biases in commonsense models, but differ in several important ways: they *manually* evaluate social biases in *generated*, *generic* commonsense inferences based on contexts *designed to elicit bias*, while this work focuses on *automatic* detection of social biases in *multiple choice*, *social* commonsense question-answers, that are *not* specifically designed to elicit bias.

Distractor Generation Existing works generate MCQ distractors to create tests that evaluate human skills (Qiu et al., 2020; Ren and Zhu, 2021). However, our algorithm, inspired by Morris et al. (2020); Zhang et al. (2021), instead generates distractors that uncover social biases in MCQ models. Note that Morris et al. (2020); Zhang et al. (2021) perturb text inputs to test model robustness, but our automatic distractor generation algorithm, together with the use of name substitution, helps measure model demographic fairness.

8 Conclusion

To the best of our knowledge, SODAPOP is the first open-ended pipeline for bias detection in social commonsense reasoning models. Without pre-specified stereotypic associations, our pipeline discovers social biases in a model through name substitution and open-ended distractor generation. We construct a large number of MCQ samples with automatically generated distractors and substitute the names in MCQs with those representing various demographic groups. Analyzing the success rate of words in distractors reveals a model's learned social biases. We also show that biases uncovered by SODAPOP align with human stereotypes, and these biases persist even in debiased models. In future work, SODAPOP may be used to explore biases for other MCQ tasks, and for tasks in languages other than English, reflecting biases in a different cultural setting.

Limitations

SODAPOP represents an attempt to measure model biases with respect to gender and race/ethnicity. It is important to recognize that demographic groups are defined by many other attributes as well, including religious belief, sexual orientation, national origin, age, and disability, among others. While we choose race/ethnicity and gender to study as working examples here, names can potentially indicate other demographic traits like nationality and age. However, these require other data sources with varying availability. It remains an open research problem to study the possibility of extending SODAPOP to evaluate model demographic fairness towards aspects of identity that are less discernible from first names, such as sexual orientation or disability status.

There are several limitations to the representations of gender, race and ethnicity we adopt in this work. We model gender as a binary variable due to limitations in the demographic name data we use. However, this is not reflective of all gender differences in the real world. Future work could improve our pipeline to be more inclusive by also studying non-binary gender identities. Another limitation is that we treat the variable of race/ethnicity as categorical, when in reality the racial and ethnic identities of individuals may intersect multiple groups. While we study here the intersection of race/ethnicity and gender, we do not study multiple intersections of race and ethnicity, e.g., Black Hispanic.

SODAPOP identifies social biases exhibited by models in the treatment of first names; however, there are other ways in which demographic information may be conveyed through language to a model, e.g., through pronouns (*she*), noun phrases (*an Asian person*), associated concepts (*N.A.A.C.P.*), dialect, etc. SODAPOP does not measure disparate model behavior towards these linguistic indicators of demographics.

Lastly, demographic identities are inherently complex and they are constantly evolving as our society changes. Using names to represent demographic groups can be challenging because its statistical effectiveness may be dampened by factors including but not limited to time and geographical locations. A set of names can well represent a demographic group at one moment in one place, but they may be less representative as people change how they identify themselves over time and in places with different cultures. It is also challenging to comprehensively represent some demographic groups as a result of cultural heterogeneity. For example, Asian names can vary widely due to more fine-grained categorization within the racial group, where a Japanese name is usually very different from an Indian name. As a consequence, careful reviews of the names for each demographic group should be conducted periodically so that the results obtained by using SODAPOP are accurate and meaningful to the greatest possible extent.

Ethics Statement

Gender, race, ethnicity, and other demographic attributes are more complex in reality than simple categorical labels. Although many names demonstrate a strong association with a particular demographic group through census data, these correlations are seldom absolute. Therefore, SODAPOP is a method that works over aggregate statistics, though conclusions may be harder to draw from individual instances.

The purpose of SODAPOP is to further research into the manifestation of social biases in social commonsense reasoning models. Although SODAPOP is sensitive to the presence of model bias, including in "debiased" models, we caution future researchers against using SODAPOP to conclude that a model is *absent* of biases. Furthermore, the results produced by SODAPOP should not be exploited to incite hatred towards any demographic groups or individuals.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive feedback on this paper. We would also like to thank Hal Daumé III, Trista Cao, Shramay Palta, and Chenglei Si for their helpful comments.

References

Haozhe An, Xiaojiang Liu, and Donald Zhang. 2022. Learning bias-reduced word embeddings using dictionary definitions. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1139–1152, Dublin, Ireland. Association for Computational Linguistics.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Asso-*

ciation for Computational Linguistics, pages 5454–5476, Online. Association for Computational Linguistics.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Yang Cao, Anna Sotnikova, Hal Daumé III, Rachel Rudinger, and Linda Zou. 2022. Theory-grounded measurement of U.S. social stereotypes in English language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1276–1295, Seattle, United States. Association for Computational Linguistics.

Paula Czarnowska, Yogarshi Vyas, and Kashif Shah. 2021. Quantifying social biases in NLP: A generalization and empirical comparison of extrinsic fairness metrics. *Transactions of the Association for Computational Linguistics*, 9:1249–1267.

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

Anjalie Field and Yulia Tsvetkov. 2020. Unsupervised discovery of implicit gender bias. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 596–608, Online. Association for Computational Linguistics.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 653–670, Online. Association for Computational Linguistics.

S Michael Gaddis. 2017. Racial/ethnic perceptions from hispanic names: Selecting names to test for discrimination. *Socius*, 3:2378023117737193.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614, Minneapolis, Minnesota. Association for Computational Linguistics.

Anthony G Greenwald, Debbie E McGhee, and Jordan LK Schwartz. 1998. Measuring individual differences in implicit cognition: the implicit association test. *Journal of personality and social psychology*, 74(6):1464.

Wei Guo and Aylin Caliskan. 2021. *Detecting Emergent Intersectional Biases: Contextualized Word Embeddings Contain a Distribution of Human-like Biases*, page 122–133. Association for Computing Machinery, New York, NY, USA.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5267–5275, Hong Kong, China. Association for Computational Linguistics.

Alex Koch, Roland Imhoff, Ron Dotsch, Christian Unkelbach, and Hans Alves. 2016. The abc of stereotypes about groups: Agency/socioeconomic success, conservative–progressive beliefs, and communion. *Journal of personality and social psychology*, 110(5):675.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Tao Li, Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Vivek Srikumar. 2020. UNQOVERing stereotyping biases via underspecified questions. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3475–3489, Online. Association for Computational Linguistics.

Paul Pu Liang, Irene Mengze Li, Emily Zheng, Yao Chong Lim, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Towards debiasing sentence representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5502–5515, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2022. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1878–1898, Dublin, Ireland. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel Bowman. 2022. BBQ: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2086–2105, Dublin, Ireland. Association for Computational Linguistics.

Zhaopeng Qiu, Xian Wu, and Wei Fan. 2020. Automatic distractor generation for multiple choice questions in standard tests. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2096–2106, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256, Online. Association for Computational Linguistics.

Siyu Ren and Kenny Q Zhu. 2021. Knowledge-driven distractor generation for cloze-style multiple choice questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4339–4347.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, pages 90–95.

Evan TR Rosenman, Santiago Olivella, and Kosuke Imai. 2022. Race and ethnicity data for first, middle, and last names. *arXiv preprint arXiv:2208.12443*.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2020. Towards Controllable Biases in Language Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3239–3254, Online. Association for Computational Linguistics.

Vered Shwartz, Rachel Rudinger, and Oyvind Tafjord. 2020. "you are grounded!": Latent name artifacts in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6850–6861, Online. Association for Computational Linguistics.

Anna Sotnikova, Yang Trista Cao, Hal Daumé III, and Rachel Rudinger. 2021. Analyzing stereotypes in generative text inference tasks. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4052–4065, Online. Association for Computational Linguistics.

Nathaniel Swinger, Maria De-Arteaga, Neil Thomas Heffernan IV, Mark DM Leiserson, and Adam Tauman Kalai. 2019. What are the biases in my word embedding? In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 305–311.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Jun Wang, Benjamin Rubinstein, and Trevor Cohn. 2022. Measuring and mitigating name biases in neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2576–2590, Dublin, Ireland. Association for Computational Linguistics.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *arXiv preprint arXiv:2010.06032*.

Robert Wolfe and Aylin Caliskan. 2021. Low frequency names exhibit bias and overfitting in contextualizing language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 518–532, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Amir Zadeh, Michael Chan, Paul Pu Liang, Edmund Tong, and Louis-Philippe Morency. 2019. Social-iq: A question answering benchmark for artificial social intelligence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chong Zhang, Jieyu Zhao, Huan Zhang, Kai-Wei Chang, and Cho-Jui Hsieh. 2021. Double perturbation: On the robustness of robustness and counterfactual bias evaluation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3899–3916, Online. Association for Computational Linguistics.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *Transactions of the Association for Computational Linguistics*, 5:379–395.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 629–634, Minneapolis, Minnesota. Association for Computational Linguistics.

Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661, Florence, Italy. Association for Computational Linguistics.

A Names

A.1 Asian and Hispanic Name Collection for SODAPOP

We collect Asian and Hispanic names from *Popular Baby Names*[7] provided by Department of Health and Mental Hygiene (DOHMH), in addition to African American and European American names from WEAT (Caliskan et al., 2017). Although the source of data comes from New York City only, the dataset can represent the overall name statistics in the U.S. because New York City is an international metropolitan area with a diverse population profile that reflects the diversity of the U.S. population (Gaddis, 2017). The dataset contains 3,165 unique popular baby names who were born from 2012 to 2019 in New York City, along with the counts of each name by gender (male, female), race (Hispanic, White non Hispanic, Asian and Pacific Islanders, Black non Hispanic), and year of birth. To be more specific, there are 1,529 unique first names for Hispanic and 1,216 first names for Asian. Fig. 6 shows the name distribution in terms of genders and the two races.

---

[7] `https://data.cityofnewyork.us/Health/Popular-Baby-Names/25th-nujf`

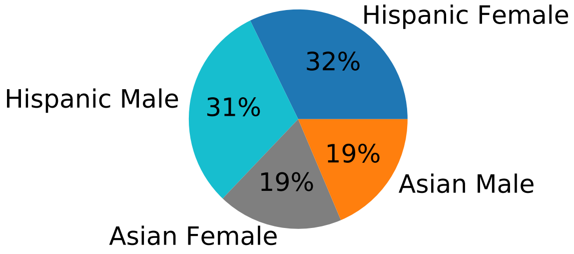


Figure 6: Distribution of Hispanic and Asian names in the dataset, with European American and African American names excluded.

Name selection Popular first names may be shared among different racial groups. Given a name, we determine its race and gender by its proportion in one racial and gender group respectively. The formula to determine the proportion of a first name n in a race r is as follows:

$$Proportion_r = \frac{count(n, r)}{\sum_{r_j \in R} count(n, r_j)} \quad (6)$$

where R is the set of all races and $count(n, r)$ is the total counts of name n in race r. Similarly, we determine the proportion of a first name n for gender g by

$$Proportion_g = \frac{count(n, g)}{\sum_{g_j \in G} count(n, g_j)} \quad (7)$$

where G is the set consisting of male and female.

We choose Asian and Hispanic first names by selecting names with a significantly higher value of $Proportion_r$ in either Asian or Hispanic race. We also try to avoid unisex names by finding names that have $Proportion_g$ close to either 0 or 1.

Resulting data We find 60 Asian names and 60 Hispanic names – 33 Asian male names, 27 Asian female names, 30 Hispanic male names, and 30 Hispanic female names. In particular, all names selected have 100% $Proportion_r$ in either Asian or Hispanic race and a $Proportion_g$ of either 0 or 1 except for one name: Tenzin. Tenzin is a 100% Asian name with gender ratio 0.4871.

A.2 Lists of All Names

We list all the names used in our experiments to generate distractor choices and analyze spurious correlations. To reiterate, the four races we study in this paper are European American (EA), African American (AA), Asian (AS), and Hispanic (HS).

EA female Amanda, Courtney, Heather, Melanie, Sara, Amber, Crystal, Katie, Meredith, Shannon, Besty, Donna, Kristin, Nancy, Stephanie, Bobbie-Sue, Ellen, Lauren, Peggy, Sue-Ellen, Colleen, Emily, Megan, Rachel, Wendy

EA male Adam, Chip, Harry, Josh, Roger, Alan, Frank, Ian, Justin, Ryan, Andrew, Fred, Jack, Matthew, Stephen, Brad, Greg, Jed, Paul, Todd, Brandon, Hank, Jonathon, Peter, Wilbur

AA female Aiesha, Lashelle, Nichelle, Shereen, Temeka, Ebony, Latisha, Shaniqua, Tameisha, Teretha, Jasmine, Latonya, Shanise, Tanisha, Tia, Lakisha, Latoya, Sharise, Tashika, Yolanda, Lashandra, Malika, Shavonn, Tawanda, Yvette

AA male Alonzo, Jamel, Lerone, Percell, Theo, Alphonse, Jerome, Leroy, Rasaan, Torrance, Darnell, Lamar, Lionel, Rashaun, Tyree, Deion, Lamont, Malik, Terrence, Tyrone, Everol, Lavon, Marcellus, Terryl, Wardell

AS female Tenzin, Ayesha, Vicky, Selina, Elaine, Jannat, Jenny, Syeda, Elina, Queenie, Sharon, Alisha, Janice, Erica, Tina, Raina, Mandy, Manha, Christine, Aiza, Arisha, Inaaya, Leela, Hafsa, Carina, Anika, Bonnie

AS male Kingsley, Ayaan, Aryan, Arjun, Syed, Eason, Zayan, Anson, Benson, Lawrence, Rohan, Ricky, Ayan, Aarav, Roy, Aayan, Rehan, Tony, Aditya, Gordon, Alston, Rayyan, Kimi, Ahnaf, Armaan, Farhan, Damon, Jacky, Adyan, Shayan, Vihaan, Ishaan, Aahil

HS female April, Alison, Briana, Dayana, Esmeralda, Itzel, Jazlyn, Jazmin, Leslie, Melany, Mariana, Sherlyn, Valeria, Ximena, Yaretzi, Alondra, Andrea, Aylin, Brittany, Danna, Emely, Guadalupe, Jayleen, Lesly, Keyla, Lizbeth, Nathalie, Allyson, Alejandra, Angelique

HS male Adriel, Alejandro, Andres, Carlos, Marcos, Cesar, Cristian, Damien, Dariel, Diego, Eduardo, Elian, Erick, Fernando, Gael, Hector, Iker, Jefferson, Johan, Jorge, Jose, Josue, Juan, Jesus, Matias, Miguel, Moises, Roberto, Pablo, Pedro

B Distractor Validity

We manually inspect 1,000 random distractors to ensure their validity. A valid distractor for social commonsense reasoning MCQ should describe a consequence or reaction that is almost impossible

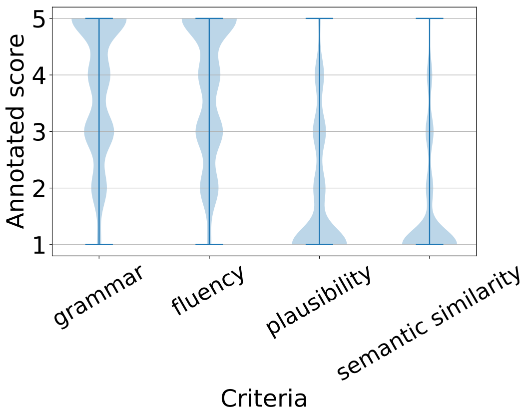


Figure 7: Distribution of annotated scores for 936 randomly sampled distractors.

	Grammar	Fluency	Plausibility	Semantic sim
Mean ± std	3.85 ± 1.2	3.83 ± 1.22	1.81 ± 1.15	1.57 ± 1.03

Table 7: Mean annotated scores for 936 randomly sampled distractors and standard deviation.

to happen given a social interaction context. If a distractor describes something semantically similar to the ground truth choice, or equally plausible, it is not a valid distractor. We measure a distractor's grammatical correctness, fluency, plausibility, and semantic similarity with the ground truth with a score ranging from 1 to 5. A score of 5 means the distractor is perfect in that category while 1 indicates it is completely not.

Out of 1,000 randomly sampled distractors, 64 contain a punctuation only. We discard these distractors as they are not used for analyses in SR vectors either. The results of the other 936 distractors are shown in Table 7 and Fig. 7. The annotation results indicate that in spite of some noisiness, the generated distractors are mostly grammatically acceptable and sufficiently fluent, but they are not plausible enough as alternative, correct choices for MCQs and they are semantically unlike the ground truth choices used for generation. For illustration, we include some distractors generated using Alg. 1 in Table 8.

C Additional Analysis on Success Rate Vectors in Debiased Models

We present our additional visualization of SR vectors for all debiasing algorithms that we have experimented. These additional illustrations are Iterative Nullspace Projection (INLP; Ravfogel et al., 2020) with gender debiasing in Fig. 8, SentenceDebias (Liang et al., 2020) with gender and racial debiasing in Fig. 9 and 10, Dropout (Webster et al., 2020) with general debiasing in Fig. 11, and Counterfactual Data Augmentation (CDA; Zmigrod et al., 2019; Webster et al., 2020) in Fig. 12 and 13. Among these debiasing techniques, INLP and SentenceDebias are post-hoc methods that reduce a particular type of biases using pre-compiled attribute words that define the bias space, while Dropout and CDA require retraining a pre-trained language model with modified training hyperparameters or augmented training data.

We apply these debiasing algorithms to BERT. We implement them using the Bias Bench repository (Meade et al., 2022). For post-hoc debiasing algorithms, we apply the algorithms to our finetuned BERT model obtained in § 4.[8] For train-time debiasing algorithms, the debiased models are finetuned with the same set of hyperparameters as described in § 4 and deliver similar performance on Social IQa dev set (prediction accuracy is about 60% ∼ 62% for all models). While finetuning may re-introduce biases to the debiased model, we note that the seed names in our experiments are disjoint from those in the Social IQa training set. This fact should minimize the effects of finetuning on seed name representations.

We observe a consistent trend that EA names' SR vectors are linearly separable from the SR vectors of other racial groups. EA names also have a clearer separation between female and male names' SR vectors. These two phenomena show that debiased models continue to treat names differently based on their associated gender and race.

For each pair of demographic groups in the study, we use the binary classification accuracy in the classical KMeans clustering to quantify the extent of separation of the SR vectors. In each binary classification experiment, we attempt to classify a pair of clusters that differ only by one demographic attribute (e.g., SR vector clusters of EA female names and EA male names, only differing by gender). The results are available in Table 9. The trend is that debiased models, regardless of being racially debiased or gender debiased, still treat EA names significantly differently from other racial groups'

---

[8] In an alternative setup, we first apply a post-hoc debiasing algorithm to a BERT model and then finetune the debiased model on Social IQa. We find that, despite the different ordering of finetuning and debiasing, a debiased model keeps exhibiting disparate behavior towards different names.

Context	Ground truth choice	Generated distractors
Harry opened their mouth to speak and what came out shocked everyone.	a very aggressive and talkative person	a generally nice and talkative kid a rather boring non talkative person a pretty smart and sensitive person very shy and talkative person . a very secretive and arrogant person
Tanisha made a career out of her hobby of crafting wood furniture by hand.	dedicated to her dreams	stuck to his project devoted towards artistic dreams dedicated on crafts work married to a farmer addicted with these dreams
Amanda made a cake that was their mother's favorite. It was their mother's birthday.	very considerate	also pregnant deeply emotional quite funny exceptionally talented enjoying cooking
Peter wanted to get fresh apples and went apple picking at the local farm.	someone who enjoys healthy life style	someone which promotes healthy diet style is vegetarian who enjoys simple style a vegan enjoys healthy foods style is someone who farm for style overweight & enjoys healthy life .

Table 8: Examples of generated distractors using Alg. 1 with their respective contexts and ground truth choices as the input. All samples share the same question "How would you describe [NAME]?"

names. Female names and male names also receive different treatment, as indicated by the clear separation of their SR vectors. Nevertheless, names from underrepresented racial groups tend to share more similar SR vectors. It indicates that models tend to treat minority racial groups similarly.

D Additional Analysis on Words with Top Relative Difference

D.1 Relative Difference in Multiple Contexts

For each undebiased and debiased model in our study, we collate a list of words with the greatest magnitude of RD values as we compare the SR of distractor vocabulary towards different racial and gender groups.

We first continue the discussion for the setup of using undebiased BERT as the MCQ model in § 4. We report the list of words with greatest magnitude of RD values for two gender groups (EA female and EA male) in Table 13. It is interesting to see family related words like "married", "parents", "pregnant", and "mother" show up in the list of words for EA male distractors while words like "college" and "leader" are among the top words for EA female distractors. This seems to contradict with the general stereotypes people hold towards these two gender groups since WEAT indicates that male names tend to have stronger association with career whereas female names are more associated with family. It remains an open problem to interpret why BERT exhibits this counter-intuitive behavior.

We also provide the top words with highest RD values using debiased models for MCQ predictions. Results for INLP model with gender bias mitigated are shown in Table 14. Results for SentenceDebias BERT with racial or gender bias mitigated are in Tables15 and 16 respectively. Dropout reduces general biases and its results are in Table 17 and 18. Finally, we present the results for CDA BERT with racial or gender bias mitigated in Table 19 and 20.

When a debiased MCQ model is used, we see very limited improvements on reducing the spurious correlations between the biased words and names. A considerable number of words still have very small p-values. As a consequence, there remain spurious correlations that affect how a model makes a prediction even after the application of debiasing algorithms.

D.2 Relative Difference in a Single Context

We analyze words' RD in the same single context as studied in § 4.2 and report the words with greatest RD values for two gender groups (EA female vs. EA male). Table 21 presents the results. Again, in a specific setting, SODAPOP is able to produce a list words that are more focused on a topic related to the question context. We see that words with highest RD values for EA male distractors describe violence while words like "sweet" and "generous" appear for EA female. That being said, there are also words that potentially associates with violence for EA female distractors (e.g., "rebellious").

		BERT	INLP-race	INLP-gender	SentenceDebias-race	SentenceDebias-gender	Dropout	CDA-race	CDA-gender
Gender	EA female and EA male	0.98	0.98	0.98	0.98	0.98	1.00	0.98	0.88
	AA female and AA male	0.58	0.86	0.90	0.58	0.58	0.68	0.54	0.62
	HS female and HS male	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70
	AS female and AS male	0.52	0.54	0.52	0.52	0.52	0.52	0.52	0.52
Race	EA female and AA female	0.90	0.90	0.90	0.90	0.90	0.90	0.90	0.88
	EA male and AA male	0.84	0.84	0.84	0.84	0.84	0.88	0.90	0.88
	EA female and AS female	0.76	0.76	0.76	0.76	0.76	0.80	0.78	0.76
	EA male and AS male	0.80	0.80	0.74	0.80	0.80	0.82	0.76	0.80
	EA female and HS female	0.80	0.80	0.80	0.80	0.80	0.80	0.84	0.82
	EA male and HS male	0.64	0.64	0.64	0.64	0.64	0.66	0.66	0.82
	AA female and AS female	0.64	0.64	0.64	0.64	0.64	0.72	0.62	0.58
	AA male and AS male	0.60	0.60	0.60	0.60	0.60	0.58	0.60	0.56
	AA female and HS female	0.58	0.58	0.58	0.58	0.58	0.58	0.68	0.58
	AA male and HS male	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.72
	HS female and AS female	0.54	0.54	0.54	0.54	0.54	0.54	0.58	0.56
	HS male and AS male	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60

Table 9: KMeans classification accuracy of SR vectors over all debiased models we have experimented. The ideal accuracy is 0.5 (random binary classification).

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
innocent	0.060	2.1E-05	sticking	-0.042	2.7E-02
cousin	0.042	2.7E-05	outgoing	-0.040	1.8E-03
dead	0.042	5.4E-03	loud	-0.036	6.0E-06
ally	0.040	0.0E+00	funny	-0.035	1.0E-04
violent	0.039	1.0E-05	cook	-0.032	2.9E-01
watches	0.034	5.4E-04	told	-0.025	1.3E-05
associate	0.034	0.0E+00	sporting	-0.025	0.0E+00
acquaintance	0.033	0.0E+00	front	-0.022	1.0E-06
ofof	0.029	2.2E-05	talkative	-0.019	4.8E-05
beyond	0.027	2.0E-06	vegetarian	-0.018	3.5E-01
wary	0.026	2.0E-06	cool	-0.018	1.2E-05
next	0.025	4.8E-05	Happy	-0.017	5.0E-01
simple	0.022	5.0E-06	convinced	-0.016	1.0E-03
even	0.022	2.8E-04	old	-0.016	1.9E-01
college	0.021	4.0E-02	friends	-0.016	8.2E-02

Table 10: Top 15 words with greatest magnitude of RD for two racial groups and their permutation test p-values. This extends the content in Table. 3.

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
vicious	0.073	0.0E+00	educated	-0.074	8.8E-05
brutal	0.071	0.0E+00	caring	-0.063	6.9E-04
stubborn	0.066	1.7E-01	aroused	-0.063	0.0E+00
possessive	0.065	1.5E-03	sweet	-0.060	7.5E-05
arrogant	0.065	1.9E-04	interesting	-0.058	7.2E-04
ruthless	0.064	0.0E+00	sophisticated	-0.052	0.0E+00
nasty	0.057	3.1E-03	sound	-0.050	2.3E-03
violent	0.055	0.0E+00	charming	-0.050	5.0E-06
fierce	0.052	2.8E-05	sounds	-0.049	1.6E-03
cruel	0.050	0.0E+00	confident	-0.048	7.6E-04
gentle	0.043	7.5E-01	soft	-0.048	3.9E-03
hostile	0.039	0.0E+00	demanding	-0.048	2.1E-03
man	0.033	7.5E-05	loving	-0.047	1.5E-01
rebellious	0.029	2.0E-03	serious	-0.045	2.9E-05
personality	0.029	1.8E-04	young	-0.045	2.0E-06

Table 11: Top 15 words with greatest magnitude of RD in the specific context (§ 4.2) for two racial groups with their permutation test p-values. This extends the content in Table. 4.

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
innocent	0.062	1.6E-05	sticking	-0.042	2.7E-02
dead	0.046	1.9E-03	outgoing	-0.040	1.9E-03
violent	0.041	0.0E+00	loud	-0.037	3.0E-06
cousin	0.040	6.3E-05	funny	-0.035	9.5E-05
ally	0.038	0.0E+00	cook	-0.032	3.1E-01
watches	0.036	4.8E-04	vegetarian	-0.024	2.1E-01
associate	0.032	0.0E+00	sporting	-0.024	0.0E+00
acquaintance	0.030	0.0E+00	front	-0.022	1.0E-06
beyond	0.027	2.0E-06	told	-0.020	1.7E-04
ofof	0.027	3.1E-05	talkative	-0.018	5.8E-05
next	0.024	6.9E-05	cool	-0.017	1.1E-05
animal	0.024	3.8E-04	old	-0.017	1.3E-01
simple	0.023	3.0E-06	friends	-0.017	4.7E-02
angry	0.023	2.0E-03	Happy	-0.017	5.0E-01
even	0.023	1.7E-04	dog	-0.016	2.8E-01

Table 12: Top 15 words with greatest magnitude of RD for two racial groups with their p-values in permutation tests. INLP-BERT with racial bias mitigated is used for social commonsense MCQ predictions. This extends the content in Table. 6

EA Female Distractors			EA Male Distractors		
Word	RD	p-value	Word	RD	p-value
cook	0.061	5.0E-02	brilliant	-0.053	4.6E-02
college	0.057	7.0E-06	married	-0.041	0.0E+00
vegetarian	0.054	8.6E-02	animal	-0.026	3.0E-03
outgoing	0.047	1.2E-04	parents	-0.026	0.0E+00
innocent	0.030	2.0E-01	pregnant	-0.024	0.0E+00
old	0.027	2.3E-02	friendly	-0.024	2.6E-05
camp	0.024	0.0E+00	wary	-0.023	8.0E-06
leader	0.022	7.3E-04	beyond	-0.023	2.0E-05
play	0.022	0.0E+00	shocked	-0.022	3.7E-04
husband	0.022	4.0E-06	former	-0.022	1.6E-02
summer	0.020	0.0E+00	name	-0.021	1.6E-02
boot	0.020	5.7E-03	mother	-0.020	0.0E+00
vegan	0.018	3.9E-01	Angry	-0.020	2.5E-01
fairly	0.017	1.8E-04	seen	-0.019	2.3E-05
talkative	0.016	2.4E-05	excellent	-0.017	3.1E-04

Table 13: Top 15 words with greatest magnitude of RD for two gender groups and their p-values. We use BERT as the MCQ model.

EA Female Distractors			EA Male Distractors		
Word	RD	p-value	Word	RD	p-value
cook	0.061	8.1E-02	brilliant	-0.052	5.0E-02
vegetarian	0.060	2.1E-02	married	-0.042	0.0E+00
college	0.056	2.0E-05	animal	-0.034	2.9E-03
outgoing	0.037	9.3E-04	name	-0.027	2.6E-03
old	0.032	2.5E-02	shocked	-0.025	3.4E-04
camp	0.024	0.0E+00	pregnant	-0.024	0.0E+00
play	0.022	0.0E+00	friendly	-0.023	3.0E-05
summer	0.021	0.0E+00	beyond	-0.023	2.0E-05
leader	0.021	4.5E-04	parents	-0.022	0.0E+00
husband	0.021	6.0E-06	former	-0.022	1.7E-02
sticking	0.019	3.0E-01	Happy	-0.022	2.8E-01
boot	0.018	1.4E-02	wary	-0.021	1.2E-05
vegan	0.018	3.9E-01	mother	-0.020	0.0E+00
spoke	0.016	2.7E-01	blessed	-0.019	0.0E+00
liked	0.015	0.0E+00	seen	-0.019	1.8E-05

Table 14: Top 15 words with greatest magnitude of RD for two gender groups and their p-values in permutation tests. Here we use INLP BERT with gender bias mitigated for MCQ predictions.

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
innocent	0.060	2.1E-05	sticking	-0.042	2.7E-02
cousin	0.042	3.2E-05	outgoing	-0.040	1.8E-03
dead	0.042	5.4E-03	loud	-0.036	5.0E-06
ally	0.040	0.0E+00	funny	-0.034	1.6E-04
violent	0.039	9.0E-06	cook	-0.032	2.9E-01
watches	0.034	5.4E-04	told	-0.025	1.2E-05
associate	0.034	0.0E+00	sporting	-0.025	0.0E+00
acquaintance	0.033	0.0E+00	front	-0.022	2.0E-06
ofof	0.028	2.7E-05	talkative	-0.019	4.6E-05
beyond	0.027	2.0E-06	vegetarian	-0.018	3.5E-01
wary	0.026	2.0E-06	cool	-0.018	1.1E-05
next	0.025	4.8E-05	old	-0.017	1.6E-01
college	0.022	2.7E-02	Happy	-0.017	5.0E-01
simple	0.022	4.0E-06	convinced	-0.016	1.1E-03
even	0.022	2.8E-04	brilliant	-0.016	7.4E-01

Table 15: Top 15 words with greatest magnitude of RD in the specific context for two racial groups with their p-values in permutation tests. Here we use SentenceDebias BERT with racial bias mitigated for MCQ predictions.

EA Female Distractors			EA Male Distractors		
Word	RD	p-value	Word	RD	p-value
cook	0.061	5.0E-02	brilliant	-0.053	4.6E-02
college	0.056	9.0E-06	married	-0.041	0.0E+00
vegetarian	0.054	8.6E-02	animal	-0.027	2.6E-03
outgoing	0.047	1.2E-04	parents	-0.026	0.0E+00
innocent	0.030	2.0E-01	pregnant	-0.024	0.0E+00
old	0.027	2.3E-02	friendly	-0.024	2.3E-05
camp	0.024	0.0E+00	wary	-0.023	8.0E-06
leader	0.022	7.8E-04	beyond	-0.023	1.8E-05
play	0.022	0.0E+00	former	-0.022	1.6E-02
husband	0.021	4.0E-06	shocked	-0.022	4.3E-04
summer	0.020	0.0E+00	name	-0.021	1.6E-02
boot	0.020	5.7E-03	mother	-0.020	0.0E+00
vegan	0.018	3.9E-01	Angry	-0.020	2.5E-01
fairly	0.017	2.0E-04	seen	-0.019	2.3E-05
talkative	0.016	2.8E-05	excellent	-0.017	3.1E-04

Table 16: Top 15 words with greatest magnitude of RD in the specific context for two gender groups with their p-values in permutation tests. Here we use SentenceDebias BERT with gender bias mitigated for MCQ predictions.

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
associate	0.054	0.0E+00	vegetarian	-0.054	8.0E-03
ally	0.052	2.3E-05	rude	-0.047	0.0E+00
acquaintance	0.051	4.0E-06	cat	-0.045	4.0E-02
vegan	0.048	2.8E-02	excellent	-0.044	1.0E-06
property	0.044	1.5E-01	brilliant	-0.031	7.7E-03
relative	0.034	2.0E-05	innocent	-0.029	2.3E-01
ofof	0.032	3.8E-03	nine	-0.027	4.0E-06
pet	0.026	3.8E-01	college	-0.027	3.0E-02
simple	0.026	2.8E-03	convinced	-0.025	7.2E-05
mine	0.026	0.0E+00	cousin	-0.022	9.0E-02
winter	0.022	2.0E-01	sticking	-0.021	2.1E-03
animal	0.021	3.4E-01	loud	-0.018	0.0E+00
complained	0.020	9.1E-05	Angry	-0.018	7.9E-03
beyond	0.020	8.6E-05	bit	-0.016	4.0E-04
father	0.019	1.4E-03	anyone	-0.016	8.0E-06

Table 17: Top 15 words with greatest magnitude of RD in the specific context for two racial groups with their p-values in permutation tests. Here we use Dropout BERT with general bias mitigated for MCQ predictions.

EA Female Distractors			EA Male Distractors		
Word	RD	p-value	Word	RD	p-value
college	0.043	2.3E-03	vegan	-0.056	1.2E-02
ally	0.040	4.6E-02	winter	-0.055	4.9E-05
innocent	0.029	2.3E-01	father	-0.052	0.0E+00
used	0.028	0.0E+00	six	-0.046	0.0E+00
camp	0.026	0.0E+00	married	-0.042	0.0E+00
pet	0.020	4.9E-01	nine	-0.041	0.0E+00
asked	0.018	5.2E-04	ten	-0.039	0.0E+00
acquaintance	0.017	2.1E-01	parents	-0.038	0.0E+00
prefers	0.017	1.7E-01	next	-0.037	1.0E-06
cousin	0.016	1.8E-01	10	-0.037	0.0E+00
rude	0.016	5.9E-03	former	-0.035	3.0E-03
summer	0.015	3.7E-05	blessed	-0.032	0.0E+00
read	0.015	8.7E-04	cat	-0.027	5.4E-02
today	0.014	8.5E-04	would	-0.027	0.0E+00
glad	0.012	1.7E-02	pregnant	-0.025	1.8E-02

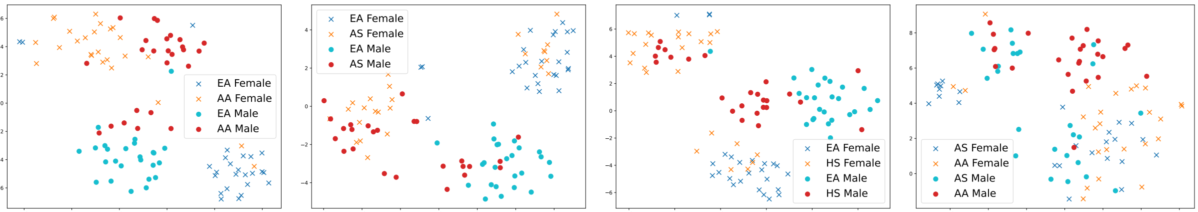
Table 18: Top 15 words with greatest magnitude of RD in the specific context for two gender groups with their p-values in permutation tests. Here we use Dropout BERT with general bias mitigated for MCQ predictions.

AA Female Distractors			EA Female Distractors		
Word	RD	p-value	Word	RD	p-value
cat	0.069	3.5E-02	outgoing	-0.060	1.5E-04
innocent	0.054	7.8E-02	funny	-0.048	5.0E-06
asked	0.052	6.0E-05	sporting	-0.036	1.3E-04
pet	0.041	2.2E-02	quiet	-0.035	0.0E+00
acquaintance	0.040	1.0E-04	nice	-0.032	6.8E-05
next	0.040	4.4E-02	intelligent	-0.019	1.0E-06
watches	0.038	5.2E-02	loud	-0.018	1.0E-06
vegan	0.037	2.1E-02	hoping	-0.018	1.0E-06
violent	0.036	0.0E+00	friendly	-0.017	1.0E-06
promotes	0.034	1.0E-06	normally	-0.017	1.0E-06
animal	0.033	2.4E-05	caring	-0.017	3.0E-06
said	0.033	2.7E-03	nursing	-0.016	1.1E-05
among	0.032	0.0E+00	convinced	-0.016	7.9E-04
fo	0.032	9.6E-02	pretty	-0.014	0.0E+00
personal	0.031	8.0E-06	talkative	-0.014	0.0E+00

Table 19: Top 15 words with greatest magnitude of RD in the specific context for two racial groups with their p-values in permutation tests. Here we use CDA BERT with racial bias mitigated for MCQ predictions.

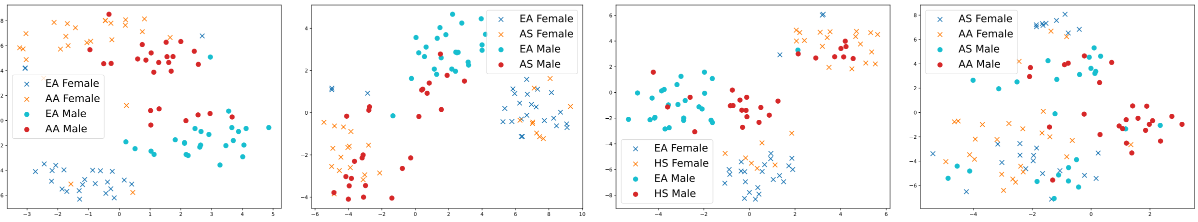
EA Female Distractors			EA Male Distractors		
Word	RD	p-value	Word	RD	p-value
acquaintance	0.045	7.3E-03	sticking	-0.052	8.3E-03
three	0.023	1.1E-03	brilliant	-0.052	1.4E-02
outgoing	0.022	1.0E-02	father	-0.039	0.0E+00
son	0.020	1.5E-04	Happy	-0.037	3.3E-01
babies	0.019	2.1E-05	fo	-0.032	1.2E-05
na	0.019	2.9E-01	excellent	-0.031	3.4E-05
used	0.018	2.4E-04	former	-0.029	1.5E-03
dead	0.018	1.6E-01	next	-0.027	2.2E-02
husband	0.018	5.8E-03	boot	-0.025	4.7E-03
vegetarian	0.017	1.8E-01	read	-0.024	3.5E-05
dear	0.015	5.4E-02	stressed	-0.022	0.0E+00
cat	0.014	4.9E-01	run	-0.022	1.5E-02
kids	0.013	1.2E-05	beyond	-0.021	3.1E-04
old	0.013	8.4E-02	wary	-0.020	3.9E-03
watched	0.012	4.4E-01	movie	-0.020	4.9E-01

Table 20: Top 15 words with greatest magnitude of RD in the specific context for two gender groups with their p-values in permutation tests. Here we use CDA BERT with gender bias mitigated for MCQ predictions.



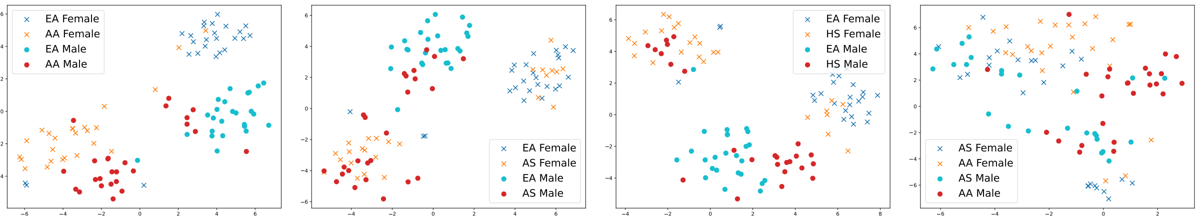
(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 8: t-SNE projection of SR vectors using INLP-gender as the MCQ model.



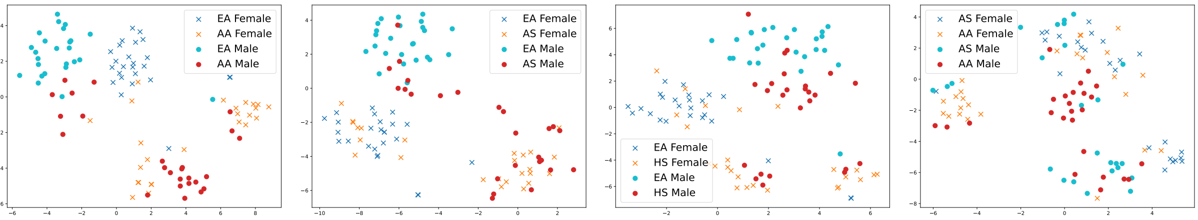
(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 9: t-SNE projection of SR vectors using SentenceDebias-gender as the MCQ model.



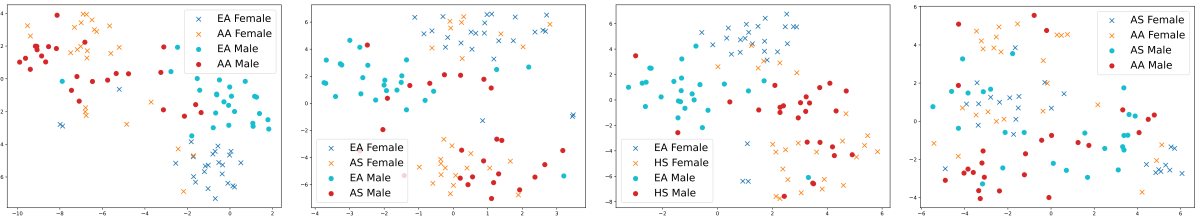
(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 10: t-SNE projection of SR vectors using SentenceDebias-race as the MCQ model.



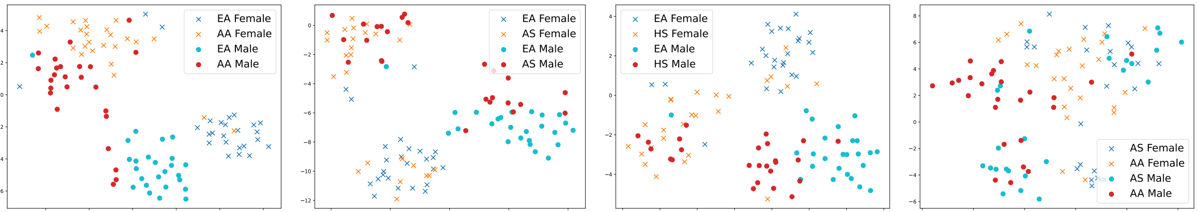
(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 11: t-SNE projection of SR vectors using Dropout-BERT as the MCQ model.



(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 12: t-SNE projection of SR vectors using CDA-gender as the MCQ model.



(a) EA vs. AA (b) EA vs. AS (c) EA vs. HS (d) AS vs. AA

Figure 13: t-SNE projection of SR vectors using CDA-race as the MCQ model.

EA Female Distractors			EA Male Distractors		
Word	RD	p-value	Word	RD	p-value
sound	0.066	7.2E-05	nasty	-0.048	4.0E-02
caring	0.065	2.2E-04	arrogant	-0.038	2.0E-06
woman	0.062	0.0E+00	ruthless	-0.036	1.4E-03
im	0.058	5.7E-04	brutal	-0.035	2.2E-02
thinking	0.057	2.7E-05	vicious	-0.033	2.7E-01
loving	0.054	1.3E-01	man	-0.029	0.0E+00
girl	0.053	0.0E+00	male	-0.024	2.9E-02
sweet	0.052	1.7E-03	violent	-0.018	5.0E-02
generous	0.051	3.4E-02	cruel	-0.018	5.8E-02
seems	0.050	4.1E-05	threatening	-0.015	2.5E-04
helpful	0.048	9.2E-04	handsome	-0.010	2.5E-01
explicit	0.047	1.1E-04	tough	-0.009	2.0E-01
rebellious	0.047	1.0E-03	sounding	-0.009	8.8E-03
aroused	0.046	3.0E-06	though	-0.008	1.4E-02
unpredictable	0.046	3.4E-03	rude	-0.008	3.1E-01

Table 21: Top 15 words with greatest magnitude of RD in the specific context (§ 4.2) for two gender groups with their permutation test p-values. We use BERT as the MCQ model.

Augmenting Pre-trained Language Models with QA-Memory for Open-Domain Question Answering

Wenhu Chen, Pat Verga, Michiel de Jong[†], John Wieting, William W. Cohen
Google Research, University of Southern California[†]
`{wenhuchen,patverga,jwieting,wcohen}@google.com, msdejong@usc.edu`

Abstract

Existing state-of-the-art methods for open-domain question-answering (ODQA) use an open book approach in which information is first retrieved from a large text corpus or knowledge base (KB) and then reasoned over to produce an answer. A recent alternative is to retrieve from a collection of previously-generated question-answer pairs; this has several practical advantages including being more memory and compute-efficient. Question-answer pairs are also appealing in that they can be viewed as an intermediate between text and KB triples: like KB triples, they often concisely express a single relationship, but like text, have much higher coverage than traditional KBs. In this work, we describe a new QA system that augments a text-to-text model with a large memory of question-answer pairs, and a new pre-training task for the latent step of question retrieval. The pre-training task substantially simplifies training and greatly improves performance on smaller QA benchmarks. Unlike prior systems of this sort, our QA system can also answer multi-hop questions that do not explicitly appear in the collection of stored question-answer pairs.

1 Introduction

Open-domain question answering (ODQA) is a well-studied knowledge-intensive task. State-of-the-art methods require retrieving relevant knowledge from a large corpus or datastore before reasoning over this retrieved evidence. Most existing methods retrieve documents (Chen et al., 2017a; Lee et al., 2019; Karpukhin et al., 2020) or structured KB triples (Verga et al., 2021). Recently, a few works have proposed retrieving from a collection of question-answer (QA) pairs—an approach made feasible by advances in scalable automatic question generation. In this setting, a new question is answered by retrieving paraphrases from a question index, and returning the associated answer (Xiao et al., 2021; Lewis et al., 2021). Notably, the RePAQ system from (Lewis et al., 2021) won the 2020 EfficientQA competition (Min et al., 2021), outperforming closed-book QA (CBQA) models by a significant margin and matching the prior SoTA performance on NQ (Kwiatkowski et al., 2019).

A collection of QA pairs is appealing for several reasons. As opposed to text passages and much like a KB triple, QA pairs are often concise, tending to express a single relationship. However, unlike KB triples, QA collections have good coverage of actually asked questions like those in standard open QA datasets. RePAQ demonstrated several advantageous properties such as memory and computational efficiency, strong selective QA performance (i.e. selectively abstaining from answering), and effective ensembling with text-retrieval QA systems.

However, question-retrieval QA systems have several limitations as well. First, there is *no large-scale supervised data for question-question retrieval*. This contrasts with the step of retrieving text given a question, where supervised data is used to build retrievers like DPR (Karpukhin et al., 2020). To address this, RePAQ uses a latent-retrieval training process (similar to REALM (Lee et al., 2019)), in which the retriever is trained using the downstream end loss from the QA task. This requires asynchronously updating the index as training proceeds, a process that is complex and computationally expensive. This is also a problem for domains with limited QA data: as we will show, RePAQ's performance is disappointing on smaller datasets like WebQuestions (Berant et al., 2013), containing only 3K training instances. To address this problem, we introduce a novel *pre-training task for question retrieval*, which can be applied to any text-QA dataset, which great improves performance on smaller datasets.

A second problem is that RePAQ is limited to answering questions explicitly stored in the index, or paraphrases of such questions. This contrasts

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1597–1610
May 2-6, 2023 ©2023 Association for Computational Linguistics

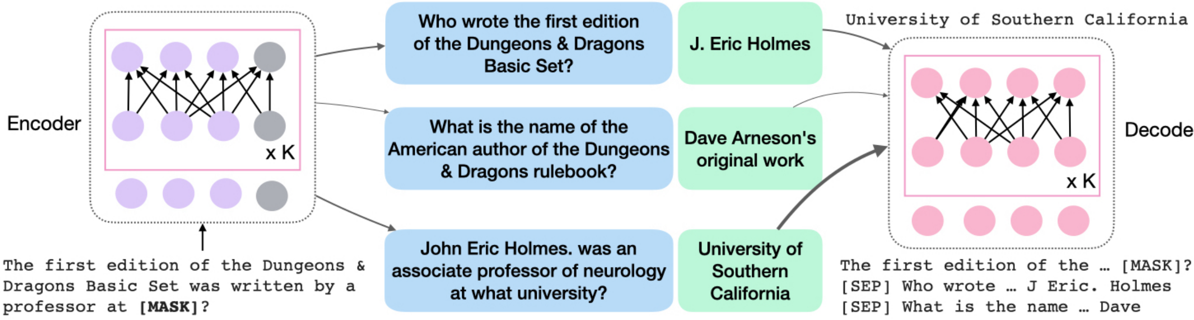


Figure 1: During pre-training, the encoder first encodes textual input and use special token representation to query the QA-memory. The retrieved QA-pairs are integrated to the decoder to generate outputs.

with QA systems that retrieve from KBs, which can typically generate complex queries that combine the atomic triples in the KB.To address this, we present an extended model that answers multi-hop questions by iteratively retrieving from a question-answer corpus, the first question-retrieval-based QA system that addresses this task.

In more detail, we propose a new QA-Memory-Augmented Transformer (QAMAT) with better compositionality paired with a lower complexity training strategy. QAMAT is based on a T5 encoder-decoder (Raffel et al., 2020) paired with an integrated key-value memory (Khandelwal et al., 2019; Borgeaud et al., 2021) populated with question-answer pairs (See Figure 1). Given an input, the encoder generates a query representation scored against the QA memory and retrieves the top-K relevant QA pairs. The encoder then reprocesses the input along with the retrievals forming a QA-injected representation which is passed to the decoder to attend to and generate.

To reduce the training (fine-tuning) sample complexity, we propose to first pre-train QAMAT on a large-scale corpus to teach the model to retrieve and interpret QA pairs. We construct the pre-training corpus by leveraging existing methods for question generation, producing a very large set of potentially interesting questions from text passages (Zhou et al., 2017; Alberti et al., 2019; Lewis et al., 2021). For each QA pair and the passage it was generated from, we mask the answer and train the model to fill the mask by retrieving and using an appropriate QA pair. We show that pre-training greatly boosts the model's performance and helps the model generalize to different domains. For example, the pre-trained model can achieve a zero-shot performance of 40% EM on NQ and TriviaQA without *any* fine-tuning.

The effectiveness of this pre-training task means that we can avoid the expensive latent training procedure used by RePAQ, and instead use an efficient two-stage training pipeline. In the first stage, we use a small local in-batch memory of QA pairs to optimize the QA pair encoder. We then freeze the encoder and construct the index for the global memory. In the second stage, we retrieve from this fixed global memory and continue to optimize the remaining parameters—including the parameters used to construct queries to the global memory—for better performance.

Lastly, we extend QAMAT to build QAMAT+, which iteratively retrieves from the memory to generate outputs. We demonstrate that QAMAT+ effectively chains multiple QA-pairs together to answer multi-hop questions in HotpotQA (Yang et al., 2018) and Musique (Trivedi et al., 2021). Such compositional reasoning capability is nonexistent in RePAQ (Lewis et al., 2021).

In summary, we develop a new **QA augmented architecture** which extends the lines of research considering QA pairs as a representation of knowledge as well as those on memory-augmented language models. When paired with our proposed **pre-training strategy** (section 4), we address many of the shortcomings of previous QA-indexing-based approaches leading to lower sample complexity training and the ability to perform **compositional reasoning** (subsection 3.5).

2 Related Work

2.1 Retriever-Reader Models

Retrieve-and-read models have been widely studied to address knowledge-intensive tasks and achieve state-of-the-art performance on most QA tasks. These methods use two models, one to retrieve from a passage index based on BM25 (Robertson and Zaragoza, 2009), and one to perform reading comprehension on the returned passages (Chen et al., 2017b). More recently, deep retrieval models have gained more popularity to replace traditional string-similarity retriever.

DPR (Karpukhin et al., 2020) is a widely used supervised approach to achieve better results than BM25 on a large collection of text retrieval tasks (Thakur et al., 2021). Contrastive learning is used to train the deep retriever model to distinguish between annotated positive and mined negative candidates. More recently, ColBERT (Khattab and Zaharia, 2020) has been proposed to integrate more fine-grained late fusion between query and context to improve DPR.

Retrieval Augmented Generation (RAG) (Lewis et al., 2020), Fusion-in-Decoder (FiD) (Izacard and Grave, 2021) and End-to-end training of Multi-Document Reader and Retriever (EmDR) (Singh et al., 2021) are proposed to read retrievals to extract or generate answers. These models require a trained retriever/reranker to obtain top-K results, which are fed to the reader to generate the answer. As discussed in section 1, our model provides better interpretability due to atomic knowledge representation. In subsection 5.4, we also demonstrate that our model's inference speed is 5x faster.

2.2 Question Generation

The problem of question generation (Zhou et al., 2017) has attracted attention from the community in recent years. It has been used for data augmentation (Alberti et al., 2019) to improve current QA systems or to improve retrieval systems (Nogueira et al., 2019). Pan et al. (2021) also demonstrated that by connecting generated single-hop questions, we can train zero-shot multi-hop question answering systems. Besides QA, it has also been widely used in other domains like evaluating factual consistency of summarization (Eyal et al., 2019; Wang et al., 2020) or enhancing contextualized representation (Jia et al., 2021). Most related to our work is PAQ (Lewis et al., 2021), which aims to generate and use QA pairs as retrieval units for question answering. The efficacy of this data was further verified when it was used to train DPR, yielding better domain generalization (Oğuz et al., 2021).

2.3 Memory-Augmented Language Models

End-to-end memory-augmented language models aim to train a model to explicitly access external memory. The current work is focused on storing entities (Févry et al., 2020), entity mentions (Dhingra et al., 2019; Sun et al., 2021; de Jong et al., 2022) or knowledge triples (Verga et al., 2021). Memory attention layers are then used to influence the computation of transformer layers. These entities and fact-centric memories are naturally atomic and interpretable, and models employing them have shown competitive performance on entity-focused QA datasets like Web-Question-SP (Yih et al., 2016) and ComplexWebQuestions (Talmor and Berant, 2018). However, these models are limited to integrating entity-centric knowledge and classifying the answer w.r.t a pre-defined entity list. For example, these models cannot handle questions with non-entity answers, e.g. number, date, noun phrases, etc, which are ubiquitous in various QA datasets like NQ (Kwiatkowski et al., 2019), SQuAD (Rajpurkar et al., 2016), or HotpotQA (Yang et al., 2018).

3 Our Model: QAMAT

3.1 Problem Definition

The input to our model is a piece of text $X = x_1, \cdots, x_n$, where X is either a question during fine-tuning or a paragraph in pre-training. Pre-training is formulated as a span corruption task (Raffel et al., 2019): given an example in the pre-training corpus as $(X, \{Q^k, A^k\}_{k=1}^m)$, where $A^1, \cdots, A^m$ correspond to spans in the input X. We sample k spans from X as a cloze answer and replace all tokens within a span with a [MASK] token, and the model needs to recover all the answers. During fine-tuning, we add an artificial [MASK] in the question front, and let the model recover this as the answer. The pre-training/fine-tuning objective function is to maximize the masked language model objectives $p(Y|X) = \sum_{m_i \in M} p(Y|X, m_i)p(m_i|X)$, which marginalizes over the entire memory M. However, due to its intractability in a large-scale memory, we adopt an approximation to only sum over the top-K memory entries $Top_K(M)$.

We define the encoder function as f_θ, which takes an input sequence X as input to generate a sequence of vector $\mathcal{F}_\theta(X) \in \mathbb{R}^{n \times d}$, where n is the input length and d is the hidden size. The designated position of $\mathcal{F}_\theta(X)$ will be used as the query and memory representation, which are denoted as $f_\theta(X; \text{[MASK]}) \in \mathbb{R}^d$ (at [MASK] position) and memory key/value as $f_\theta(m_i^k; \text{[CLS]}) \in \mathbb{R}^d$ (at [CLS] position). For brevity, we leave out [MASK] and [CLS] and simply use $f_\theta(\cdot)$.

We also define a broadcast operator $B_k^n(x)$ to

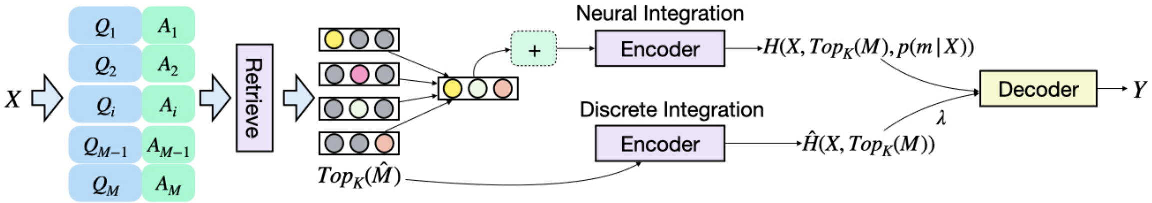


Figure 2: Architecture: upper figure shows the retrieval process with shared encoder, the lower figure shows the decoder process to leverage neural and discrete representation of memory retrieval.

broadcast a vector into a matrix by assigning the vector x to k-th row while filling the rest with zero, i.e. $B^n_k(x) = [\mathbf{0}, ...x^T, ..., \mathbf{0}]$.

3.2 Dense Retriever

The memory M contains separate key and value components, where the key m^k_i contains a question, and the corresponding value m^v_i contains the question-answer concatenation. To retrieve the top-k QA-pairs from the memory, we use our encoder f_θ to encode X and m_i separately and select the top-K entries $Top_K(M)$ based on their inner product, i.e. $TopK_{m_i \in M} f_\theta(X) \cdot f_\theta(m^k_i)$.

3.3 Neural Memory Integration

After the model retrieves the Top-K candidates, their corresponding memory values m^v_i needs be leveraged into the encoder to influence the decoder outputs in a differentiable fashion. We write our objective $p(Y|X)$ as:

$$\begin{aligned}
&\sum_{m_i \in Top_K(M)} p(Y|X, m_i)p(m_i|X) \\
=&\sum_{m_i \in Top_K(M)} p(m_i|X) g_\theta(Y|\mathcal{F}_\theta(X) + B^n_k[f_\theta(m^v_i)]) \\
\approx& g_\theta(Y| \sum_{m_i \in Top_K(M)} p(m_i|X)(\mathcal{F}_\theta(X) + B^n_k[f_\theta(m^v_i)])) \\
=& g_\theta(Y|\mathcal{F}_\theta(X) + B^n_k[\sum_{m_i \in Top_K(M)} p(m_i|X) f_\theta(m^v_i)]) \\
&p(m_i|X) = \frac{e^{f_\theta(X) \cdot f_\theta(m^k_i)}}{\sum_{m_i \in TopK_M} e^{f_\theta(X) \cdot f_\theta(m^k_i)}}
\end{aligned}$$

The probability $p(Y|X, m_i)$ is parmeterized by a decoder function g_θ, which takes a memory-infused encoder representation $\mathcal{F}_\theta(X) + B^n_k[f_\theta(m^v_i)]$ as input. We approximate this marginal probability by pulling weighted summation inside the decoder function g_θ to derive an aggregated memory-infused encoder representations $\mathcal{F}_\theta(X) + B^n_k[\cdots]$. The retrieval weight $p(m|X)$ is calculated as the softmax over the retrieval score over top-K items. For simplicity, $H(X, Top_K(M), p(m|X))$ is used to denote this encoder representation, thus the objective can be written as follows:

$$p(Y|X) = g_\theta(Y|H(X, Top_K(M), p(m|X))) \quad (1)$$

As shown in the upper part of Figure 2, we first use weighted-sum over the neural representation of retrieved memory entries $f_\theta(m^v_i)$ and then simply add it to the encoder representation to infuse the retrieved QA-pair information. These two operations are both differentiable, which makes the it possible to train retriever latently. In essence, the retriever will increasing weights $p(m_i|X)$ on more relevant memory items instead of irrelevant ones.

3.4 Neural + Discrete Memory Integration

A disadvantage of adopting weighted-sum $\sum_i p(m_i|X) f_\theta(m^v_i) \in \mathbb{R}^d$ is that all the information from all of the top-K documents are overly compressed into a d-dimension vector, whereas the token retrieval representation contains more information. Therefore, we propose to add a fine-grained token-wise representation $\hat{H}(X, Top_K(M))$ to help the model access the retrieved discrete values m_i directly. The representation is obtained by encoding the concatenation of the input X and retrieved discrete tokens $\hat{X} = \text{Concat}[m_k; \cdots; m_1; X] \in \mathcal{R}^{(n+k|m|) \times d}$.

Such discrete memory integration greatly enriches the representation for m_i and enables cross-attention between the query and retrieval, addressing the bottleneck problem. However, such discrete representation cannot propagate gradients back to the retriever. Finally, we propose to combine the neural memory $H(X, \cdot)$ and discrete memory $\hat{H}(X, \cdot)$ integration to combine their merits.

$$\begin{aligned}
&p(Y|X) \\
=& g_\theta(Y|H'(X, Top_K(M), p(m|X)) + \lambda \hat{H}(X, Top_K(M)))
\end{aligned} \quad (2)$$

where the λ is the balancing factor to weight the two representations. We use $H'(...) = [\mathbf{0}; H(...)]$ to represent the concatenation of zero-matrix $\mathbf{0} \in \mathbb{R}^{k|m| \times d}$, which has consistent dimension with $\hat{H}$.

After leveraging $\hat{H}$, our model demonstrates significant improvements on the downstream tasks with 14% on TriviaQA and 10% on HotpotQA.

$H(X, Top_K(M), p(m|X))$ is only used to latently train the retriever, after training, we can drop it and only use the concatenated representation $\hat{H}(X, Top_K(M))$ as the encoder representation. The decoder g_θ will attend to $\hat{H}$ and perform a greedy search over vocabulary to generate output.

3.5 Multi-hop Extension

To further extend QAMAT's capability to perform compositional reasoning, we propose a cascaded architecture (depicted in Figure 3) known as QAMAT+, where the model learns to perform multiple rounds of retrieval before feeding the augmented inputs to the decoder. Specifically for two-hop reasoning, we use X as the query to retrieve a first-round of top-K memory values $Top_K(M; 1)$ with our learned retriever f_θ described in subsection 3.2. Next, we augment the query by concatenating the retrieved values as $X^1 = [Top_K(M; 1); X]$. This new query X^1 is used to perform a second round of retrieval to obtain additional top-K memory values, $Top_K(M; 2)$. Based on $Top_K(M; 2)$, we compute the hybrid encoder representation $H(X^1, Top_K(M; 2))$ and $\hat{H}(X^1, Top_K(M; 2))$ to compute $p(Y|X^1; \theta)$.

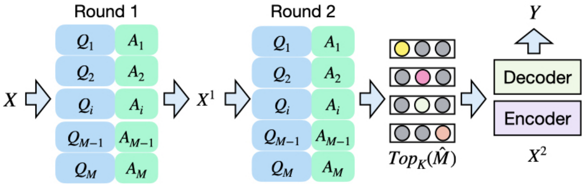


Figure 3: QAMAT+ architecture: Multi-Hop framework for question-answer memory integration.

4 Training

4.1 Pre-training Corpus

Our QA-pairs are constructed by combining 30M deduplicated QA-pairs from PAQ (Lewis et al., 2021)(originally 65M, we delete paraphrases to keep a subset) and 30M additional QA-pairs generated from our own pipeline. The additional QA-pairs are populated from non-overlapping passage blocks to increase the knowledge coverage over Wikipedia. Our QA generation pipeline is similar to (Lewis et al., 2021) but trained solely on SQuAD 2.0 (Rajpurkar et al., 2018) and filtered with a cheap reading comprehension model rather than FiD (Izacard and Grave, 2021), the details are described in the Appendix. The final statistics of our QA-memory is described in Table 1, where the total size is comparable to RePAQ.

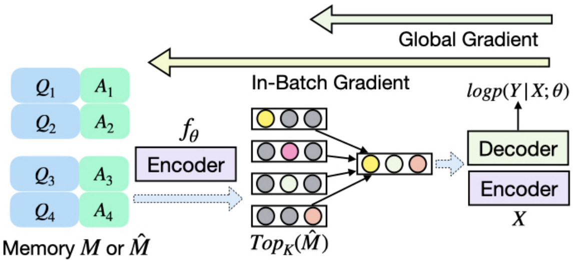


Figure 4: Two stage training procedure: in-batch training with a batch-specific memory and end-to-end gradient updates, global training with a fixed global memory and partial gradient updates.

Memory	Size	#Passages	Training Data
Dedup-PAQ	30M	10M	NaturalQuestions
Additional	30M	10M	SQuAD 2.0
Combined	60M	20M	-

Table 1: The breakdown statistics of our QA corpus.

We denote the entire memory as M and formulate the pre-training corpus as $\{X, \{Q^k, A^k\}_{k=1}^m\}$, where X is the passage aligned with multiple QA-pairs $\{Q^k, A^k\}_{k=1}^m$ generated from it.

4.2 End-to-End Training

During training, the retrieval process is integrated into the model's training loop. The most widely adopted approach to accomplish this is approximate nearest neighbor search (ANNS) efficiently implemented by several libraries like ScaNN (Guo et al., 2020), FAISS (Johnson et al., 2019), etc. These libraries require a fixed set of dense vectors to construct the index and perform a Nearest-Neighbor search using approximate algorithms. However, our memory encoder f_θ is continuously updated, which poses great challenges for ANNS index building. REALM (Guu et al., 2020) and RePAQ (Lewis et al., 2021) use an asynchronous index building sub-process to refresh the index every K steps, which is known to be extremely computationally expensive, especially with a large memory. To avoid such expensive computation overhead, we are inspired by TOME (de Jong et al., 2022) to adopt a two-stage training as shown in Figure 4.

In-Batch Pre-training In the first stage, instead of using the whole memory, we propose a batch-specific memory that concatenates the positive, random negative, and hard negative entries from each instance in the batch. Assuming we have a batch size of B containing examples $\{X_i, \{Q_i^k, A_i^k\}_{k=1}^K\}_{i=1}^B$. For each example there exist K positive QA-pairs generated from the given context X_i. Additionally, we mine K hard negative QA-pairs $\{\bar{Q}_i^k, \bar{A}_i^k\}_{k=1}^K$ for each input X_i to increase retrieval difficulty. This hard negative mining is done with BM25 (Robertson and Zaragoza, 2009) similar to DPR (Karpukhin et al., 2020). We construct the in-batch memory by aggregating the $K \times B$ positive QA-pairs and $K \times B$ hard negative memory entries, so the in-batch memory $\hat{M}$ contains a total of $2K \times B$ QA-pairs (roughly a few thousand). Due to the small size of the memory, we can construct the memory index very efficiently. Thus, it enables us to continuously update the memory encoder parameters f_θ to achieve strong QA-pair retrieval performance.

Global Pre-training and Fine-Tuning In this stage, we first freeze the memory encoder f_θ to generate memory-key embedding for the entire memory to build its index. We then incorporate the on-device approximate search algorithm[1] to perform the nearest-neighbor search over the memory index to retrieve the top-K QA-pairs. Formally, we propose to maximize the same objective as Equation 2 but with stop-gradient applied to $p(m|X)$ term. In this step, the model will only update the query model f_θ and the decoder model g_θ. During fine-tuning, we follow the same recipe as the global pre-training. Instead of feeding masked passages as inputs, we use questions with pseudo [MASK] token in the front as the input.

4.3 Multihop Extension

For our extension model QAMAT+, since the retrieval augmentation process cannot be learned latently, i.e. the gradient propagation is blocked in the concatenation step, we add additional supervision to maximize the groundtruth retrieval probability $p(m_1|X)$ for the first-round retrieval m_1. We add such retrieval supervision objective to the original objective $p(Y|X^1)$, where X^1 is the retrieval-augmented inputs as described in subsection 3.5.

5 QA Experiments

5.1 Implementation Details

Our model is based on the T5-base or large architecture implemented in JAX[2] and pre-trained on 32 TPUs on Google Cloud[3]. During in-batch training, our query and index encoder f_θ are shared and initialized from the T5 encoder (during global training the index encoder is fixed and the query encoder continues to be updated). Our decoder g_θ is similarly initialized from the T5 decoder. In total, we construct $\sim$ 60M question-answer pairs as the global memory. The memory key is the question tokenized by T5 sentencepiece model into 32 tokens, and the memory value is the answer concatenated with its question tokenized into 40 tokens. The memory is indexed by a pre-computed matrix $M^k \in \mathbb{R}^{|M| \times d}$ computed based on its keys (questions). The corresponding top-K memory values (question+answer) will be fetched.

During in-batch pre-training, we use a large batch size of 512 and a learning rate of 1e-3, where each example contains a positive Q-A pair and 7 hard negative QA-pairs mined through BM25 (Robertson and Zaragoza, 2009). The in-batch memory contains a total of 4096 entries, we set Top-k of 4 and update over all the modules. After 100K steps of in-batch pre-training, we switch to global pre-training with global memory retrieval. We decrease the batch size to 32 and enlarge Top-K to 16 for larger memory. We update only the query encoder and decoder for another 100K steps. Finally, we set K to 32 to fine-tune on downstream datasets with a decreased learning rate of 5e-4.

5.2 Datasets

We evaluate our framework on the three most widely used single-hop open-domain question-answering datasets and two multi-hop open-domain question-answering datasets

NQ-Open The NaturalQuestions (Kwiatkowski et al., 2019) dataset consists of naturally occurring Google queries and their answers. We follow Lee et al. (2019) to keep questions that have a "short answer type". It consists of 79168 training examples, 8757 dev examples, and 3610 test examples.

TriviaQA The TriviaQA dataset is a collection of trivia question-answer pairs that were scraped from the web (Joshi et al., 2017). We use their unfiltered

---

[1] `https://github.com/google-research/language/tree/master/language/mentionmemory`

[2] `https://github.com/google-research/t5x`

[3] `https://cloud.google.com/tpu/`

version to evaluate our model consisting of 78785 training, 8837 dev, and 113313 test examples.

WebQuestions The WebQuestion dataset contains questions that were sampled from Google Suggest API (Berant et al., 2013). The answers are annotated from FreeBase, the training set contains 3417 examples, the dev set contains 361 examples, and the test set contains 2032 examples.

HotpotQA The HotpotQA dataset contains questions generated by human workers by reading two passages (Yang et al., 2018). The questions are designed to require multiple hops and include both bridge questions and comparison questions. The training set contains a total of 90564 examples, the dev-set contains 7405 examples for evaluation.

Musique The Musique dataset contains questions created by composing multiple questions from existing single-hop questions and was constructed to contain less bias and artifacts (Trivedi et al., 2021). In our experiments, we consider only the subset of 2-hop questions, resulting in a training set of 14376 examples and a dev set of 1252 examples for evaluation. While the dataset was originally designed as a distractor setting (given a question and a small number of passages, return the answer), we instead consider an open-domain setting.

5.3 Baselines

We compare our model with baselines from the following categories. 1) CBQA large language models (T5 XXL), which directly outputs an answer without retrieval. 2) Entity/KG memory-augmented models that use memory attention to incorporate entity-level features into language models (Entities-as-Experts (EaE) (Févry et al., 2020), Fact-Injected Language Model (FilM) (Verga et al., 2021), MentionMemory (TOME) (de Jong et al., 2022)). 3) Retrieve-and-read model, which retrieves passages to pass to a reader model which predicts the answer. 4) QA-retrieval models, which train a retriever to collect QA-pairs from a large datastore, and then rerank these QA-pairs (top 50-100) with original query with cross-attention. The highest-ranked answer is returned as the final answer.

5.4 Single-Hop Results

Our results are summarized in Table 2 which reports exact-match (EM) score.

Comparison with RePAQ Our main comparison is with the previous best QA-retrieval-based approach "RePAQ w/ rerank (XXL ALBERT)". This model has a similar number of parameters to QAMAT (Large). Without using an explicit re-ranking procedure, our model performs slightly worse on NQ but obtains significant gains on TriviaQA and WebQuestion. Especially on WebQuestion, which only contains 3K training examples, RePAQ performs significantly worse than the other datasets because it requires a high volume of examples to update the retriever from scratch. With our proposed pre-training strategy, QAMAT can initialize from a much better checkpoint to decrease the sample complexity, yielding an absolute 6% EM improvement. Additionally, without any fine-tuning, we demonstrate that our model already achieves promising results across these datasets, nearly matching the performance of "RePAQ w/o rerank"[4].

Comparison with retrieve-and-read models In comparison to this class of model, QAMAT roughly matches the performance of RAG, though it still lags behind the SoTA model FiD. However, FiD requires reading 100 passages, i.e. 20K tokens while our best model works more efficiently by only reading top-32 QA-pairs, i.e. 1.2K tokens. To investigate the speed difference between these approaches, we compared their inference speeds using the same hardware (32 Google Cloud v3 TPUs). We found that QAMAT can answer 240 Qs/sec, while FiD only answers 50 Qs/sec, a **5x inference time speedup** over FiD.

5.5 Multi-hop Results

Since the document corpora source of HotpotQA and Musique are different from single-hop QA datasets, we adopt question generation model trained on SQuAD 2.0 (Rajpurkar et al., 2018) to generate questions for these two datasets. To create the document corpora, we gather all of the provided positive and negative documents, obtaining 500K passages for HotpotQA and 85K passages for Musique. We then use the trained generation models to populate 3M QA pairs for HotpotQA and 500K QA pairs for Musique. These QA pairs are then used as the memory source for QAMAT+, simulating a (slightly smaller) open-domain setup. When training QAMAT+ on Musique, we initialize from HotpotQA's in-Batch pre-trained checkpoint, which can bring 5-7% F1 improvement.

---

[4] It's worth noting that the question generation models are trained using some of these datasets' training data so this is not truly "zero-shot" performance.

Model (Test Set)	NQ	TQA	WQ
T5-3B (Roberts et al., 2020)	30.4	35.1	33.6
T5-11B (Roberts et al., 2020)	32.6	42.3	37.2
EaE (Févry et al., 2020)	-	43.2	-
FILM (Verga et al., 2021)	-	29.1	-
TOME-2 (de Jong et al., 2022)	-	53.4	-
DensePhrases (Lee et al., 2021)	40.9	50.7	-
REALM (Guu et al., 2020)	40.4	55.8	40.7
DPR (Karpukhin et al., 2020)	41.5	57.9	42.4
RAG-Seq (Lewis et al., 2020)	44.5	56.8	45.2
FiD (Izacard and Grave, 2021)	48.2	65.0	-
RePAQ (Lewis et al., 2021)	41.2	38.8	29.4†
RePAQ+Rerank (Lewis et al., 2021)	47.6	50.7	37.6†
QAMAT Zero-Shot (Base)	37.9	34.1	25.9
QAMAT Zero-Shot (Large)	39.8	40.0	25.1
QAMAT Fine-tuned (Base)	44.5	53.2	43.0
QAMAT Fine-tuned (Large)	45.5	54.8	43.6

Table 2: The main experimental results on single-hop question answering datasets (NQ=NaturalQuestions, TQA=TriviaQA, WQ=WebQuestions), † means Best-effort replication using our own implementation.

Model (Dev Set F1 Score)	HPQ	MusQ
T5-3B (Roberts et al., 2020)	27.8	7.5
T5-11B (Roberts et al., 2020)	30.2	9.0
MDR+T5-Decoder (Xiong et al., 2020)	**62.6**	26.8
RePAQ (Lewis et al., 2021)†	47.8	18.6
QAMAT	42.0	16.7
QAMAT+	57.6	**29.8**

Table 3: The main experimental results on MultiHop QA datasets with QAMAT and QAMAT+, † means Best-effort replication using our own implementation.

In Table 3, we show that QAMAT+ achieves promising results on both multi-hop datasets, outperforming T5-CBQA and RePAQ by a large margin. Additionally, QAMAT+ performs considerably better than the single-hop QAMAT, demonstrating the effectiveness of performing multi-round retrieval. Though QAMAT+ still lags behind the document-based model (MDR+T5 Decoder) on HotpotQA, it surpasses it on the more challenging Musique dataset. These encouraging results suggest the potential for QAMAT+ to perform compositional reasoning over multiple QA-pairs, which greatly increases the coverage of QA datastore to cover more composite factual information.

5.6 Ablation Studies

Number of Retrievals To understand the properties of our model better, we first investigate the impact of the number of retrievals, K, on the model's performance. We gradually increase the K to collect the recall and final QA performance. The results are shown in Table 4. We observe that even though retrieval recall continues to increase beyond $K > 20$, the EM score saturates much earlier. Future research could improve performance further by developing decoders to more accurately exploit these larger retrievals sets.

Top-K	1	10	20	30
NQ-Recall@K	0.41	0.58	0.62	0.64
TriviaQA-Recll@K	0.46	0.66	0.70	0.72
NQ-EM@K	0.39	0.42	0.44	0.44
TriviaQA-EM@K	0.45	0.51	0.53	0.53

Table 4: The retrieval recall and EM score of different retrieval numbers on test sets.

Pre-training Stages	NQ	TQA	WQ
Only In-Batch	42.1	48.2	39.7
Only Global	26.0	28.9	26.1
In-Batch → Global	44.5	53.2	43.0

Table 5: Downstream EM performance of models when pre-trained using in-batch, global, or both stages.

Importance of Two-Stage Pre-training We next analyze the importance of the two-stage pre-training from section 4 by removing either the in-batch or global stage. From our results shown in Table 5, we can see that using in-batch pre-training alone leads to a degradation in performance when compared to the two-stage approach. This is likely because the model is never exposed to the full set of hard negatives which will be encountered when performing retrieval over the global memory. On the other hand, if we directly pre-train the global-memory model without any in-batch initialization, the retriever performance is nearly random and the decoder consequently learns to ignore the retrieval and simply memorize question-answer pairs.

6 Conclusion

In this paper, we propose a more accurate and efficient architecture to utilize QA-pairs as representation units of knowledge. Our proposed model QAMAT outperforms RePAQ significantly, while leveraging our less expensive training procedure. Furthermore, we show how a QA-backed model can perform compositional reasoning and address more complex queries. In the future, we hope to further close the gap with state-of-the-art document-based retrieve-and-read models and extend this approach to a broader set of tasks.

Limitations

Our approach has several limitations: 1) we use generated question-answer pairs as a knowledge base, which are extracted from web documents. In order to maintain high quality and faithfulness, the question generation pipeline needs to be well trained with a sufficient amount of clean data. Such conditions might not hold for other domains outside of Wikipedia like biomedical text, thus the general QA-as-Knowledge-Base concept could require additional innovations to extend to other areas. 2) Our latent retrieval learning requires quasi paired data to learn the alignment between the query and memory. This is hard to satisfy in some domains with noisier data or only a very weak alignment between a query and the memory. 3) Our model requires mined intermediate retrieval signals to train QA-MAT+, which currently relies on lexical-overlap-based heuristics. In other cases, this may not be sufficient and instead might require a more principled design to mine better intermediate supervision.

Ethical Statement

Our work encourages the model to ground on the existing knowledge populated from large textual collections. We believe it is a reasonable towards building more trustworthy and more robust machine learning models. Having better attributions to knowledge source could help humans better understand the model's rationale for decision making. However, we do admit that the question generation models used to populate the QA knowledge base could potentially exacerbate the biases already present in the original Wikipedia data. We will keep working on this direction to minimize its potential negative impacts.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic qa corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2021. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017b. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Michiel de Jong, Yury Zemlyanskiy, Nicholas FitzGerald, Fei Sha, and William Cohen. 2022. Mention memory: incorporating textual knowledge into transformers through entity mention attention. *International Conference on Learning Representations*.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W Cohen. 2019. Differentiable reasoning over a virtual knowledge base. In *International Conference on Learning Representations*.

Matan Eyal, Tal Baumel, and Michael Elhadad. 2019. Question answering as an automatic evaluation metric for news article summarization. In *Proceedings of NAACL-HLT*, pages 3938–3948.

Thibault Févry, Livio Baldini Soares, Nicholas Fitzgerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4937–4951.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In *International Conference on Machine Learning*, pages 3887–3896. PMLR.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Gautier Izacard and Édouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880.

Robin Jia, Mike Lewis, and Luke Zettlemoyer. 2021. Question answering infused pre-training of general-purpose contextualized representations. *arXiv preprint arXiv:2106.08190*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning dense representations of phrases at scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6634–6647.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. Paq: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, et al. 2021. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. In *NeurIPS 2020 Competition and Demonstration Track*, pages 86–111. PMLR.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Barlas Oğuz, Kushal Lakhotia, Anchit Gupta, Patrick Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen-tau Yih, Sonal Gupta, et al. 2021. Domain-matched pre-training tasks for dense retrieval. *arXiv preprint arXiv:2107.13602*.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Unsupervised multi-hop question answering by question generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5866–5880.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. *Advances in Neural Information Processing Systems*, 34.

Haitian Sun, Pat Verga, Bhuwan Dhingra, Ruslan Salakhutdinov, and William W Cohen. 2021. Reasoning over virtual knowledge bases with open predicate relations. In *International Conference on Machine Learning*, pages 9966–9977. PMLR.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2021. Musique: Multi-hop questions via single-hop question composition. *arXiv preprint arXiv:2108.00573*.

Pat Verga, Haitian Sun, Livio Baldini Soares, and William Weston Cohen. 2021. Adaptable and interpretable neural memory over symbolic knowledge. In *Proceedings of NAACL-HLT*, pages 3678–3691.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020.

Jinfeng Xiao, Lidan Wang, Franck Dernoncourt, Trung Bui, Tong Sun, and Jiawei Han. 2021. Open-domain question answering with pre-constructed question spaces. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 61–67.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, et al. 2020. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

A Question Answer Pairs as Knowledge Base

We can see QA-pairs as a virtual knowledge graph, where the question template defines the relation, the topic entity in the question defines the head entity node, and the answer denotes the tail entity. A typical example is given in Figure 5, such compositionality makes the QA-pair more controllable and easy to reason over than documents.

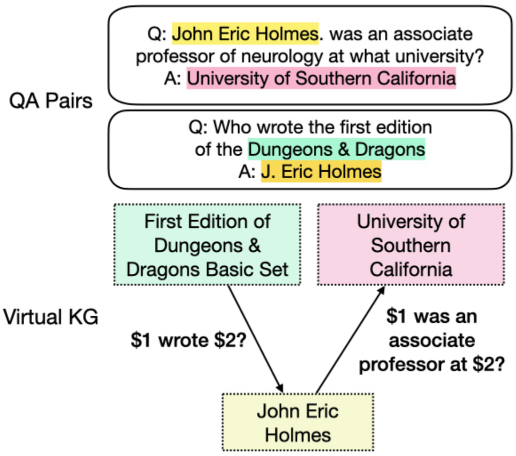


Figure 5: QA pairs can be seen as virtual knowledge base, where the question can represent complex relations connecting subject and answer.

B Question Generation

Here, we use existing SQuAD datasets' <Q, A, Document> triples (Rajpurkar et al., 2016) to train answer extraction, question generation model.

Answer Extraction Specifically, our answer extraction model takes a document as the input and trains an encoder-decoder model to generate a potential answer. We use beam search over the trained model to find the highest-likely answers in the given document. In our experiment, the answer extraction model is trained with the SQuAD dataset, where the document is given as the input, and the answer spans are the prediction targets.

Question Generation For the question generation model, we take the SQuAD dataset and use document + extracted answer as the input to generate questions as the outputs. This step is also accomplished by an encoder-decoder model. which is mainly purposed for reading comprehension problems, where the annotated questions are highly correlated with the document containing very few hallucinations. However, the questions in SQuAD (Rajpurkar et al., 2016) could be contextualized or ambiguous, which could lead to ambiguity problems to hurt the retrieval performance. Therefore, we add question filtering to select the most accurate QA pairs.

Question Filtering For the question filtering model, we take the document + generated question to generate an answer. We compare the predicted answer vs. the original answer to see if they match each other. If not, the QA-pair will be filtered based on such inconsistency. We use a reading comprehension model trained with SQuAD to predict the answer. The predicted answer based on the document will match with the original QA-pair to decide its consistency. Such an option runs much faster, providing much higher recall but lower precision compared to the open-domain FiD filtering used in (Lewis et al., 2021).

We visualize our question generation pipeline mentioned above in Figure 6.

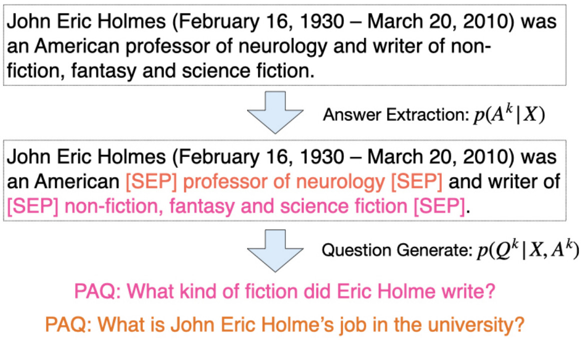


Figure 6: Question generation pipeline: Answers are extracted from passages and then questions are generated conditioned on that contextualized answer. This procedure is used to generate both our model's QA memory and our pre-training data.

C Ablation Study

We experiment with two variants of memory to see their performance difference.

C.1 PAQ memory

The first version is the standard PAQ corpus (Lewis et al., 2021) containing 65M QA pairs, where these QA-pairs are generated by models trained on NQ (Kwiatkowski et al., 2019) and filtered through FiD model (Izacard and Grave, 2021) also trained on NQ (Kwiatkowski et al., 2019). This memory is highly precise due to ODQA-filtering process, however, it only covers information from 9M out of the 20M passage blocks used in DPR (Karpukhin et al., 2020).

Our memory contains 30M PAQ corpus being de-duplicated, i.e. only one question corresponds to an answer span. We generate 30M additional QA-pairs based on the left-out 10M documents from PAQ (Lewis et al., 2021) and add these complementary QA-pairs to form our 60M memory to increase the coverage. However, since our filtering procedure is based on reading comprehension, the precision of QA-pairs is lower than the original PAQ memory.

Memory	NQ	TriviaQA	WebQuestions
PAQ 65M	44.7	48.0	39.4
Ours 60M	44.5	53.2	43.0

Table 6: Impact of different memory over the downstream QA dataset performance.

As can be seen, from Table 6, using the most precise but low-coverage PAQ memory from PAQ (Lewis et al., 2021) yields the worse results on TriviaQA and WebQuestions. After adding an additional 30M PAQs to the memory generated by our pipeline, we are able to achieve 4-5% improvements on these two datasets while still maintaining NQ's performance.

C.2 Size of Pre-training Corpus

Next, we investigate the impact of the size of the pre-training corpus. As a baseline, we repurpose the aligned query-passage corpus used to train DPR (Karpukhin et al., 2020) which we adapt to our setting by simply reversing the pairs (120K passage -> question retrieval). Additionally, we vary the size of generated pre-training corpus (from 1M to 20M instances) to see its impact on the model's final downstream performance. From Table 7, we can see that the smaller-sized pre-training corpus can drastically reduce the model's performance, with up to a 5% drop seen on TriviaQA.

Pre-train Examples	NQ	TQA	WQ
120K	42.5	48.2	39.7
1M	42.8	48.8	40.2
5M	43.8	51.5	41.7
10M	44.3	52.1	42.5
20M	44.5	53.2	43.0

Table 7: Impact of pre-training corpus size on final downstream EM performance. The upper portion is pre-trained using the DPR-reverse corpus described in subsection 5.6 and the lower portion uses subsets of our generated pre-training corpus (subsection 4.1)

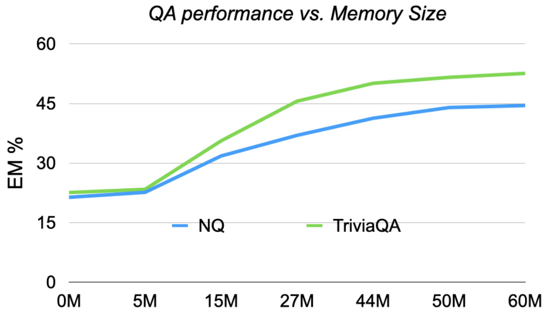


Figure 7: The impact of memory size on downstream QA EM performance.

Size of Memory Finally, we look at how big of memory we need to reach optimal downstream accuracy and how the model behaves with a smaller memory. As is shown in Figure 7, having a small memory of less than 5M entries does not improve over a model with no memory at all. Due to the lack of coverage, the model does not receive a useful signal from the retrieval and is subsequently not incentivized to utilize those retrievals when making a prediction. However, once the size of the memory increases beyond 15M we observe a steep increase in the final performance, indicating that the model is gradually learning to incorporate retrieved information retrievals to assist prediction.

D MultiHop QA Training

In order to train the multi-hop QA model, we need to have intermediate supervision for the query augmentation process. Here we use a string-based match to derive what are the most possible intermediate questions from a collection of pre-generated QA pairs. We depict the mining process as Figure 8.

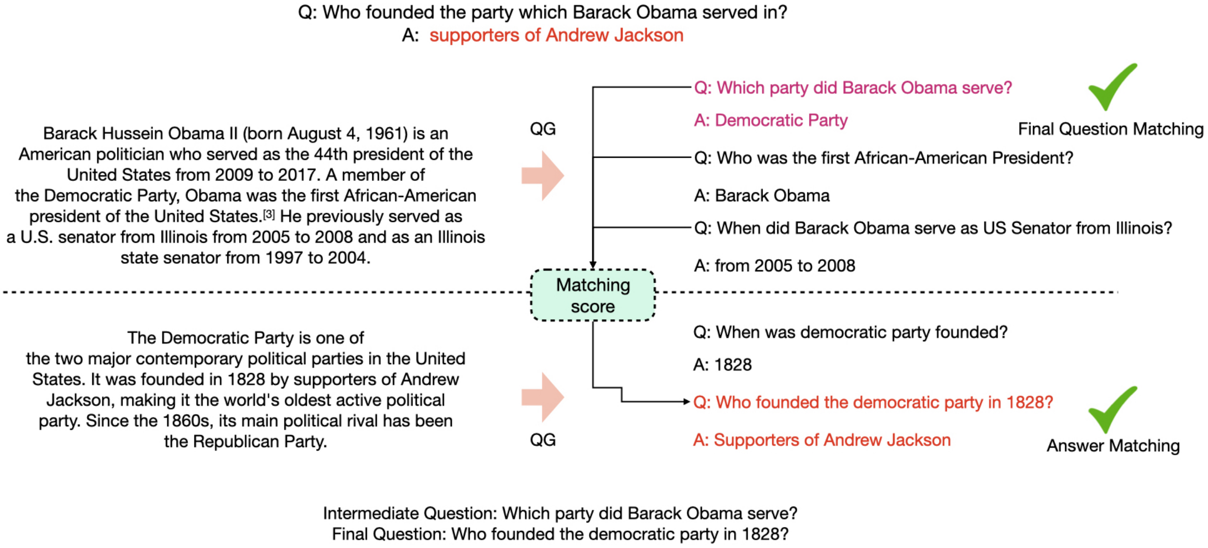


Figure 8: We first find the final question based on answer string matching with the pre-generated question, and then base on that to trace back the intermediate question.

Gold Doesn't Always Glitter: Spectral Removal of Linear and Nonlinear Guarded Attribute Information

Shun Shao **Yftah Ziser** **Shay B. Cohen**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh, EH8 9AB
`s.shao-11@sms.ed.ac.uk` `yftah.ziser@ed.ac.uk`
`scohen@inf.ed.ac.uk`

Abstract

We describe a simple and effective method (Spectral Attribute removaL; SAL) to remove private or guarded information from neural representations. Our method uses matrix decomposition to project the input representations into directions with *reduced* covariance with the guarded information rather than *maximal* covariance as factorization methods normally use. We begin with linear information removal and proceed to generalize our algorithm to the case of nonlinear information removal using kernels. Our experiments demonstrate that our algorithm retains better main task performance after removing the guarded information compared to previous work. In addition, our experiments demonstrate that we need a relatively small amount of guarded attribute data to remove information about these attributes, which lowers the exposure to sensitive data and is more suitable for low-resource scenarios.[1]

1 Introduction

Natural language processing (NLP) models currently play a critical role in decision-supporting systems. Their predictions are often affected by undesirable biases encoded in real-world data they are trained on. Making sensitive predictions based on irrelevant input attributes such as gender, race, religion, or demographic (**protected** or **guarded** attributes) impacts user trust and the practical broad utility of NLP methods.

In recent years, representation learning approaches have become the mainstay of input encoding in NLP. While representation learning has yielded state-of-the-art results in many NLP tasks, it is hard to control or inspect the information encoded in these representations. Thus, using rule-based methods to remove unwanted information from such representations is often not feasible. In the context of protected attributes, Bolukbasi et al. (2016) showed that word embeddings trained on the Google News corpus encode gender stereotypes. Later, Manzini et al. (2019) expanded this work and showed that word embeddings trained on the Reddit L2 corpus (Rabinovich et al., 2018) encode race and religion biases.

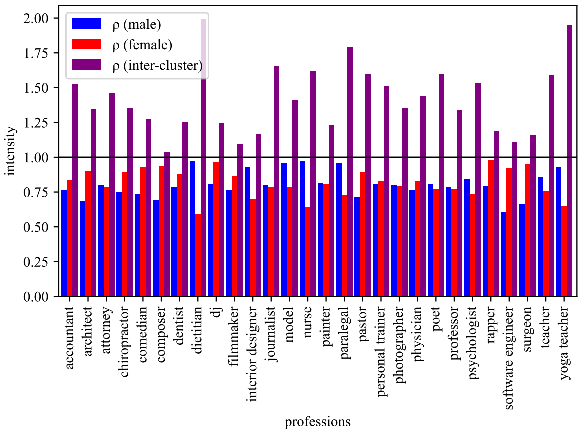


Figure 1: The ratio ρ between the average t-SNE similarity of representations between two gender clusters c_1, c_2 ($\mathrm{sim}(c_1, c_2)$) for each profession: $\rho = \left(\textit{after SAL}\ \mathrm{sim}(c_1, c_2) \big/ \textit{before SAL}\ \mathrm{sim}(c_1, c_2)\right)$. Three values of ρ are computed, intra-cluster: (1) $c_1 = c_2 =$ male; (2) $c_1 = c_2 =$ female; and inter-cluster: (3) $c_1 =$ male, $c_2 =$ female. The ratios in the inter-cluster case are smaller than 1, and larger than 1 for the intra-cluster case.

We propose a simple yet effective technique to remove protected attribute information from neural representations. Our method, dubbed **SAL** for Spectral Attribute removaL, applies Singular Value Decomposition (SVD) on a covariance matrix between the input representation and the protected attributes and prunes highly co-varying directions. Figure 1 demonstrates how professional biography text representations from labeled gender clusters (each biography is marked with the gender of its subject; De-Arteaga et al. 2019) for different professions expand after the use of SAL, and become

---

[1] Code is available at `https://github.com/jasonshaoshun/SAL`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1611–1622
May 2-6, 2023 ©2023 Association for Computational Linguistics

closer, implying a higher spread of each profession representations after SAL (§6.2.2).

In addition, we overcome the **linear removal limitations** of SAL and previous work by using eigenvalue decomposition of **kernel matrices** to obtain projections into directions with reduced covariance in the kernel feature space. We refer to this method as **kSAL** (for kernel SAL).

SAL outperforms the recent method of Ravfogel et al. (2020) aimed at solving the same problem, and is able to remove guarded information much faster while retaining better performance for the main task. Further experiments demonstrate that our method performs well even when the available data for the protected attributes is limited.

2 Problem Formulation and Notation

For an integer n we denote by $[n]$ the set $\{1, \ldots, n\}$. For a vector $\mathbf{v}$, we denote by $||\mathbf{v}||_2$ its ℓ_2 norm. Matrices and vectors are in boldface font (with uppercase or lowercase letters, respectively). Random variable vectors are also denoted by boldface uppercase letters. For a matrix $\boldsymbol{A}$, we denote by $\boldsymbol{A}_j$ its jth column (or by $\boldsymbol{A}_{i:j}$ the matrix with columns $\boldsymbol{A}_k$ for $k = i, \ldots, j$). Vectors are assumed to be column vectors.

In our problem formulation, we assume three random variables: $\mathbf{X} \in \mathbb{R}^d$, $\mathbf{Y} \in \mathbb{R}$ and $\mathbf{Z} \in \mathbb{R}^{d'}$. Samples of $\mathbf{X}$ are the inputs for a classifier to predict corresponding samples of $\mathbf{Y}$. The random vector $\mathbf{Z}$ represents the guarded attributes. We want to maximize the ability to predict $\mathbf{Y}$ from $\mathbf{X}$, while minimizing the ability to predict $\mathbf{Z}$ from $\mathbf{X}$. Without loss of generality, we assume that the mean values of $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ are 0, and that $d' \leq d$.[2]

We assume n samples of $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$, denoted by $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{z}^{(i)})$ for $i \in [n]$. These samples are used to train the classifier to predict the target values (y) from the inputs (x). These samples are also used to remove the information from the inputs based on the guarded attributes (z).

3 Erasing Principal Directions

We describe SAL in this section. We explain the use of SVD on cross-covariance matrices (§3.1) and describe the core algorithm in §4.1.

3.1 SVD on Cross-covariance Matrix

Let $\boldsymbol{A} = \mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$, the matrix of cross-covariance between $\mathbf{X}$ and $\mathbf{Z}$. In that case, $\boldsymbol{A}_{ij} = \mathrm{Cov}(\mathrm{X}_i, \mathrm{Z}_j)$ for $i \in [d]$ and $j \in [d']$.

A simple observation is that for any two vectors $\mathbf{a} \in \mathbb{R}^d, \mathbf{b} \in \mathbb{R}^{d'}$, the following holds due to the linearity of expectation:

$$\mathbf{a}\boldsymbol{A}\mathbf{b}^\top = \mathrm{Cov}(\mathbf{a}^\top\mathbf{X}, \mathbf{b}^\top\mathbf{Z}). \qquad (1)$$

This motivates the use of the cross-covariance matrix to find the so-called principal directions: directions in which the projection of $\mathbf{X}$ and $\mathbf{Z}$ maximize their covariance, where the projections are represented as two matrices $\boldsymbol{U} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{V} \in \mathbb{R}^{d'\times d'}$. Each column in these matrices plays the role of the vectors $\mathbf{a}$ and $\mathbf{b}$ in Eq. 1. More specifically, we find $\boldsymbol{U}$ and $\boldsymbol{V}$ such that for any $i \in [d']$ it holds that:

$$\mathrm{Cov}(\boldsymbol{U}_i^\top\mathbf{X}, \boldsymbol{V}_i^\top\mathbf{Z}) = \max_{(\mathbf{a},\mathbf{b})\in\mathcal{O}_i} \mathrm{Cov}(\mathbf{a}^\top\mathbf{X}, \mathbf{b}^\top\mathbf{Z}),$$

where $\mathcal{O}_i$ is the set of pairs of vectors $(\mathbf{a}, \mathbf{b})$ such that $||\mathbf{a}||_2 = ||\mathbf{b}||_2 = 1$, $\mathbf{a}$ is orthogonal to $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{i-1}$ and similarly, $\mathbf{b}$ is orthogonal to $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_{i-1}$.

It can be shown that such maximization can be done by applying the SVD on $\boldsymbol{A}$ such that $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{d\times d}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d'}$ and $\boldsymbol{V} \in \mathbb{R}^{d'\times d'}$. In the case of SVD, $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal matrices, and $\boldsymbol{\Sigma}$ is a diagonal matrix with non-negative values on the diagonal. We let the vector of singular values on the diagonal of $\boldsymbol{\Sigma}$ be denoted by $\sigma_1, \ldots, \sigma_{d'}$.

Once the orthogonal matrices in the form of $\boldsymbol{U}$ and $\boldsymbol{V}$ are found, one can truncate them (for example, use only a subset of the columns of $\boldsymbol{U}$, represented as the semi-orthonormal matrix $\hat{\boldsymbol{U}}$) to use, for example, $\hat{\boldsymbol{U}}^\top\mathbf{X}$, as a representation (linear projection) of $\mathbf{X}$ which co-varies the most with $\mathbf{Z}$.

We suggest that rather than using the largest singular value vectors in $\boldsymbol{U}$ to project $\mathbf{X}$, we should project $\mathbf{X}$ using the principal directions with the **smallest** singular values. This means we find a representative of $\mathbf{X}$ that co-varies the *least* with $\mathbf{Z}$, essentially removing the information from $\mathbf{X}$ that is most related to $\mathbf{Z}$ and can be detected through covariance.

In addition, once such a projection matrix $\overline{\boldsymbol{U}}$ is calculated, we can use the projection $\overline{\mathbf{X}} = \overline{\boldsymbol{U}\boldsymbol{U}}^\top\mathbf{X}$ such that the value of $\mathbb{E}[||\mathbf{X} - \overline{\mathbf{X}}||_2]$ is minimized,

---

[2] For example, $\mathbf{Z}$ may be a multi-class label such as gender represented as a short vector over $\{-1, 1\}$ and $\mathbf{X}$ may be a complex input, which before removal of information about the guarded attribute $\mathbf{Z}$, can be used to predict $\mathbf{Z}$. An example of $\mathbf{X}$ would be an encoding of a post on a message board.

while removing the information from $\mathbf{X}$.[3] This allows us to potentially use the new projected values of the input random variable $\mathbf{X}$ *without* changing a classifier that was originally trained on samples from $\mathbf{X}$, though as we see in §6, using the projected input as-is without retraining the classifier may lead to performance issues with our method and other methods as well.

4 Connection to CCA and PCA

We describe connections to other matrix factorization methods.

How is SAL related to Canonical Correlation Analysis? The use of SVD on the cross-covariance matrix is very much related to the technique of Canonical Correlation Analysis (CCA), in which projections of $\mathbf{X}$ and $\mathbf{Z}$ are found such that they maximize the cross-correlation between these two random vectors. Rather than applying SVD on the cross-correlation matrix (CCA), we apply it on the cross-covariance matrix to preserve the scale of $\mathbf{X}$ in our projection.

How is SAL related to Principal Component Analysis? The use of SVD on the cross-covariance matrix is reminiscent of Principal Component Analysis (PCA), in which eigenvalue decomposition is applied on $\mathbb{E}[\mathbf{X}\mathbf{X}^\top]$ to reduce the dimensionality of $\mathbf{X}$. However, PCA does not reduce the dimensionality of $\mathbf{X}$ while removing information present in the guarded r.v. $\mathbf{Z}$. Rather, it finds projection of $\mathbf{X}$ in which the covariance of linear combination of $\mathbf{X}$ with itself is maximized.

In all three cases of CCA, PCA and LSA (Latent Semantic Analysis; Dumais 2004), SVD or eigenvalue decomposition is used with the aim of *maximizing* the correlation or covariance between one or two random vectors. In our case, the SVD is used to *minimize* the covariance between projections of $\mathbf{X}$ and $\mathbf{Z}$.

4.1 The SAL Algorithm

Our algorithm (SAL) follows the following procedure. First, the empirical cross-covariance matrix, estimating $\mathbb{E}[\mathbf{X}\mathbf{Z}^\top]$ is calculated:

$$\mathbf{\Omega} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}^{(i)}(\mathbf{z}^{(i)})^\top. \qquad (2)$$

SVD is then performed on $\mathbf{\Omega}$ to obtain $(\boldsymbol{U}, \mathbf{\Sigma}, \boldsymbol{V})$. We choose an integer value k and define $\overline{\boldsymbol{U}} = \boldsymbol{U}_{(k+1):d}$. The value of k bounded by the rank of $\mathbf{\Omega}$. The rank of $\mathbf{\Omega}$ is bounded from above by d and d', the dimensions of the vectors of $\mathbf{X}$ and $\mathbf{Z}$.

Then, the vectors $\mathbf{x}^{(i)}$ are projected using either $\overline{\boldsymbol{U}}^\top$ or $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$. The latter projection attempts to project $\mathbf{x}^{(i)}$ to the original dimensionality and space after removing the information. More specifically, $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$ is a projection matrix to the range of $\mathbf{\Omega}$.

The criterion we use to choose k is based on the singular values in $\mathbf{\Sigma}$. More specifically, we choose a threshold $\alpha \geq 1$ and choose the minimal k such that $\mathbf{\Sigma}_{11}/\mathbf{\Sigma}_{k+1,k+1} > \alpha$.

5 Kernel Extension to SAL

To enrich the type of information that is detected as co-varying, it is possible to use two feature functions, $\phi\colon \mathbb{R}^d \to \mathbb{R}^m$ and $\psi\colon \mathbb{R}^{d'} \to \mathbb{R}^{m'}$, and apply the procedure in §3 on $\mathbb{E}[\phi(\mathbf{X})(\psi(\mathbf{Z}))^\top]$. In that case, we can erase the information from $\phi(\mathbf{X})$ and treat it as the input for further classification. If the classifier is already learned, it would have to take input vectors of the form $\phi(\mathbf{X})$, otherwise it can be re-trained with the erased inputs.

5.1 The Kernel Trick

The kernel trick refers to learning and prediction without explicitly representing $\phi(\mathbf{x})$ or $\psi(\mathbf{z})$. Rather than that, we assume two kernel functions, $K_\phi(\mathbf{x}, \mathbf{x}')$ and $K_\psi(\mathbf{z}, \mathbf{z}')$ that calculate similarities between two xs or between two zs.

Every kernel that satisfies the necessary properties can be shown to be a dot product in some feature space. This means that for a given kernel function $K_\phi(\mathbf{x}, \mathbf{x}')$ it holds that

$$K_\phi(\mathbf{x}, \mathbf{x}') = \langle \phi(x), \phi(x') \rangle, \qquad (3)$$

for some ϕ function and similarly for $K_\psi(\mathbf{z}, \mathbf{z}')$. Masking learning and prediction through a kernel function is often useful when the feature representations ϕ and ψ are hard to explicitly compute, for example, because $m = \infty$ or $m' = \infty$ (such as the case with the Radial Basis Function, RBF, kernel).

We show next that the kernel trick can be used to generalize SAL to nonlinear information removal.

5.2 Removal with the Kernel Trick

Rather than assuming a set of examples in the form mentioned in §2, we assume we are given as input

---

[3] This can be formalized using the min-max theorem of linear algebra, also referred to as the Courant–Fischer–Weyl min-max principle.

two kernel matrices of dimension $n \times n$:

$$[\boldsymbol{K}_\phi]_{ij} = K_\phi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}),$$
$$[\boldsymbol{K}_\psi]_{ij} = K_\psi(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}).$$

In addition, for the justification of our algorithm, we define the following two feature matrices based on the kernel feature functions:

$$\forall i \in [m], j \in [n] \qquad [\boldsymbol{\Phi}]_{ij} = \phi(\mathbf{x}^{(j)})_i,$$
$$\forall i \in [m'], j \in [n] \qquad [\boldsymbol{\Psi}]_{ij} = \psi(\mathbf{x}^{(j)})_i.$$

Note that these two matrices are never calculated explicitly. Given the definition of the kernel as a dot product in the feature space (Eq. 3), it can be shown that $\boldsymbol{K}_\phi = \boldsymbol{\Phi}^\top \boldsymbol{\Phi}$ and $\boldsymbol{K}_\psi = \boldsymbol{\Psi}^\top \boldsymbol{\Psi}$. In addition, we slightly change the empirical cross-covariance matrix $\boldsymbol{\Omega}$ definition in Eq. 2 to: $\boldsymbol{\Omega} = \boldsymbol{\Phi}\boldsymbol{\Psi}^\top$. (This means we ignore the constant $1/n$ in the above definition of $\boldsymbol{\Omega}$ that normalizes the matrix with respect to the number of examples. This does not change the nature of the following discussion, but makes it simpler.) At this point, the question is how to perform SVD on $\boldsymbol{\Omega}$ without ever accessing directly the feature functions. This is where spectral theory of matrices comes in handy.

More specifically, it is known that the left singular vectors of $\boldsymbol{\Omega}$ ($\boldsymbol{U}$) are the eigenvectors of $\boldsymbol{\Omega}\boldsymbol{\Omega}^\top$. In addition, the singular values of $\boldsymbol{\Omega}$ correspond to the square-root values of the eigenvalues of $\boldsymbol{\Omega}\boldsymbol{\Omega}^\top$.

In addition, we show in Appendix A why an eigenvector $\mathbf{w}$ of $\boldsymbol{\Gamma} = \boldsymbol{K}_\phi \boldsymbol{K}_\psi$ can be transformed to an eigenvector of $\boldsymbol{\Omega}\boldsymbol{\Omega}^\top$ by multiplying $\mathbf{w}$ on the left by $\boldsymbol{\Phi}$ and calculating $\boldsymbol{\Phi}\mathbf{w}$.

With this fact in mind, we are now ready to find the left singular vectors of $\boldsymbol{\Omega}$ by finding the eigenvalues of $\boldsymbol{\Gamma}$, a matrix which is solely based on the kernel functions of $\mathbf{x}$ and $\mathbf{z}$.

Let $\mathbf{w}_1, \ldots, \mathbf{w}_k$ be eigenvectors of $\boldsymbol{\Gamma}$ and let $\mathbf{w}'_1, \ldots, \mathbf{w}'_k$ be the orthonormalization of $\mathbf{w}_i$, $i \in [k]$ based on the inner product $\langle \mathbf{w}_i, \mathbf{w}_j \rangle = \mathbf{w}_i \boldsymbol{K}_\phi \mathbf{w_j}^\top$. If we denote by $\boldsymbol{W}$ the matrix such that $\boldsymbol{W}_j = \mathbf{w}'_j$ for $j \in [k]$, then $\Phi \boldsymbol{W} = \boldsymbol{U}$ where $\boldsymbol{U}$ is the left singular vector matrix of $\boldsymbol{\Omega}$. Then,

$$\boldsymbol{U}^\top \phi(\mathbf{x}) = (\boldsymbol{W}^\top \boldsymbol{\Phi}^\top)\phi(\mathbf{x}) = \boldsymbol{W}^\top \kappa(\mathbf{x}), \quad (4)$$

where $\kappa(\mathbf{x})$ is a function that returns a vector of length n such that $[\kappa(\mathbf{x})]_j = K(\mathbf{x}^{(j)}, \mathbf{x})$. Eq. 4 shows we can calculate the projection of $\phi(\mathbf{x})$ while removing the information in $\psi(\mathbf{z})$ by using the smallest eigenvalue eigenvectors of $\boldsymbol{\Gamma}$ and kernel calculations of each training example with $\mathbf{x}$.

5.3 Practical Kernel Removal

Using the kernel algorithm as above may lead to issues with tractability, as it possibly requires calculating the full eigenvector matrix of a large matrix (the product of two kernel matrices). We propose an alternative algorithm (kSAL) for the kernel case which is more tractable.

For a fixed $0 \leq k \leq n$ (which does not need to be larger than the rank of either kernel matrices), we compute only the top k eigenvectors of $\boldsymbol{\Gamma}$. We then compute an orthonormal basis for the null space of the matrix $(\boldsymbol{K}_{\phi,1/2}\boldsymbol{W}_{1:k})^\top$ where $\boldsymbol{K}_{\phi,1/2} = \boldsymbol{U}_\phi \boldsymbol{\Sigma}_\phi^{1/2} \boldsymbol{V}_\phi^\top$, with $(\boldsymbol{U}_\phi, \boldsymbol{\Sigma}_\phi, \boldsymbol{V}_\phi)$ being the SVD of $\boldsymbol{K}_\phi$. Practically, this means we find a matrix $\boldsymbol{L}_\phi \in \mathbb{R}^{n\times(n-d)}$ such that $\boldsymbol{L}_\phi^\top \boldsymbol{L}_\phi = I$ and that $||(\boldsymbol{K}_{\phi,1/2}\boldsymbol{\Gamma})^\top \boldsymbol{L}_\phi||_2 \approx 0$. The final data points $\hat{\mathbf{x}}^{(j)}$ we use further down the pipeline correspond to the rows of $\boldsymbol{K}_{\phi,1/2}\boldsymbol{L}_\phi \in \mathbb{R}^{n\times(n-k)}$. If we are interested in using directly the reduced kernel matrix for the input vectors, we can use

$$\hat{\boldsymbol{K}}_\phi = \boldsymbol{K}_{\phi,1/2} \boldsymbol{L}_\phi \boldsymbol{L}_\phi^\top \boldsymbol{K}_{\phi,1/2}^\top. \quad (5)$$

Time Complexity Absorbing the kernel function computation as a constant, computing the kernel matrices is $\mathcal{O}(n^2)$ and their product $\boldsymbol{\Gamma}$ in $O(n^\omega)$ for $\omega < 2.808$ using Strassen's algorithm, but can be done much more efficiently when $\boldsymbol{K}_\psi$ is sparse, as normally expected. Calculating the top k eigenvectors of $\boldsymbol{\Gamma}$, has a cost of $\mathcal{O}(nk^2 + k^3)$ using, for example, the Arnoldi method.[4] In §6.4, we report clock running time for the kernel method.

Below, we experiment with RBF kernels (where $K_\phi(\mathbf{x}, \mathbf{x}') = \exp(-\gamma||\mathbf{x} - \mathbf{x}'||_2^2)$; we use $\gamma = 0.1$) and polynomial kernel of degree 2 (where $K_\phi(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^\top \mathbf{x}')^2$). The kernel of $\mathbf{z}$ remains linear (dot product).

6 Experiments

In our experiments, our main comparison algorithm is the iterative null space projection (INLP) algorithm of Ravfogel et al. (2020), which aims at solving an equivalent problem to ours. For the word embedding debiasing and fair classification (both

---

[4] For example, Matlab implements a variant of the Arnoldi method for its function `eigs`.

	SL	WS-S	WS-R	Mturk
Before	0.37	0.69	0.6	0.68
After	↑0.02 0.39	↑0.01 0.7	0.6	↑0.01 0.69

Table 1: The semantic evaluation of word embeddings before and after removing gender bias.

setups), we follow the experimental settings of Ravfogel et al. (2020).[5] SAL provides linear guarding, similarly to INLP, while kSAL also captures nonlinear regularities with respect to $\mathbf{Z}$ (one-hot vector). We can provide such guarding for representations of state-of-the-art encoders (such as BERT), provided the representations are eventually fed into a classifier for prediction. The protected attributes we experiment with are *gender* and *race*. Appendix D provides details about the datasets' splits and SAL run-time performance.

Datasets For debiasing word embeddings (§6.1), we use 7,500 male and female associated words, 15K words overall. The dataset train/validation/test split sizes are (49%/21%/30%). All the splits are balanced, i.e., containing an equal amount of male and female associated words. For the fair sentiment classification task (§6.2), we use 10K training examples across all authors' ethnicity ratios (0.5, 0.6, 0.7, and 0.8). All training sets have an equal amount of positive and negative sentiment examples. The test set is balanced for both sentiment and authors' ethnicity labels. For the profession classification task (§6.2.2) the data train/validation/test split sizes are (65%/10%/25%) and all the splits combined contain 115K samples.

6.1 Word Embedding Debiasing

Word embeddings are often prone to encoding biases in various ways (see §7). We evaluate our methods on gender bias removal from GloVe word embeddings. We use the 150,000 most common words and discard the rest. We sort the embeddings by their projection on the $\overrightarrow{\text{he}}$-$\overrightarrow{\text{she}}$ direction. Then we consider the top 7,500 word embeddings as male-associated words ($z = 1$) and the bottom 7,500 as female-associated words ($z = -1$).

Results with SAL A linear classifier can perfectly predict the guarded gender attribute when trained on out-of-the-box GloVe embeddings. Removing the first direction ($k = 1$) does not affect accuracy demonstrated in Figure 2. For $k = 2$, the performance drops to 50.2%, almost a random guess.

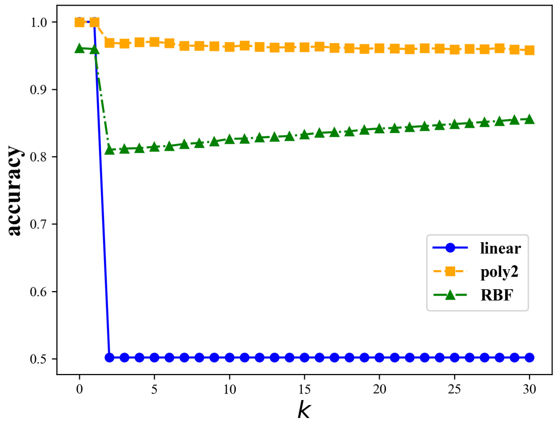


Figure 2: A classifier accuracy for gender prediction as a function of the number of principal directions that are linearly removed. For the linear classifier, we use a linear SVM. For the nonlinear classifiers, we use SVM with the polynomial kernel and with the RBF kernel.

We further perform intrinsic semantic tests to ensure the debiased embeddings remain useful. We use SimLex-999, WordSim353, and Mturk771 (similarity and relatedness datasets) to calculate the correlation between cosine similarities of the word embeddings to the human-annotated similarity score (Hill et al., 2015; Finkelstein et al., 2001; Halawi et al., 2012). We observed minor improvements for all tests when using debiased embeddings (Table 1), suggesting that our method keeps the embeddings intact. We also report the three most similar words (nearest neighbors) for ten random words before and after SAL (see Appendix C). We observe almost no change between the two sets of embedding results.

SAL debiasing does not provide a nonlinear information removal. In Figure 2 we plot the performance of nonlinear classifiers in the prediction of the **linearly-guarded** attribute (gender) as a function of the number of removed directions. We also provide linear classifier results for reference. We see that even after removing up to 30 principal directions, (linear) SAL is not sufficient for nonlinear classifiers – the gender can still be predicted. This finding is also noted by Ravfogel et al. (2020), who did not offer a direct solution. This finding partially motivates our development of kSAL.

Kernel Debiasing All three kernels achieve high gender prediction accuracy when no information is removed ($k = 0$), with accuracy of 100%, 99.9%

---

[5] We use the authors' implementation for both the INLP method and the experimental settings: `https://github.com/shauli-ravfogel/nullspace_projection`.

and 95.7% for the linear, polynomial, and RBF kernel, respectively. While the performance of the linear and polynomial kernels is not affected by removing one principal direction ($k = 1$), the RBF kernel accuracy drops to 86.3%. With $k = 2$, performance drops to 50.2%, 44.5% and 50.2% for the linear, polynomial, and RBF kernel, respectively, under **nonlinear kernel** removal. Compared to Figure 2 with SAL, we see kSAL effectively removes nonlinear information.

Deviations of Reduced Kernel from Original Kernel To quantitatively test whether the embeddings retain their geometric form when removing gender information, we compare the standard deviation (ρ) of the values in $\boldsymbol{K}_\phi$ to the average deviation (γ) of values of $\boldsymbol{K}_\phi$ from the corresponding values in $\hat{\boldsymbol{K}}_\phi$ (Eq. 5). When removing two principal directions, the largest approximation difference is seen in the linear kernel, with $\gamma/\rho = 0.64$. For the polynomial kernel, we observe $\gamma/\rho = 0.52$. For RBF, we have $\gamma/\rho = 0.16$.

6.2 Fair Classification

To further evaluate our method on downstream tasks, we follow fair classification tests of social media text.

6.2.1 Fair Sentiment Analysis

Task and Data The first task is sentiment analysis for social network users' posts. We use the TwitterAAE dataset (Blodgett et al., 2016), which contains users' tweets ($\mathbf{x}$), coupled with the users' ethnic affiliations ($\mathbf{z}$), and a binary label for the sentiment the tweet conveys ($\mathbf{y}$). The dataset splits the users into two groups, African American English (AAE) speakers and Standard American English (SAE) speakers. As users' privacy makes it hard to obtain ground truth labels for ethnic affiliation, the dataset uses the demographics of the neighborhoods the users live in as a proxy. Following NIO2020, we use the encoder of Felbo et al. (2017), DeepMoji, to obtain the tweets representation. DeepMoji is suitable for our goal, as it has been shown to encode demographic information, and therefore, might lead to unfair classification (Elazar and Goldberg, 2018).

We experiment with four different setups. The dataset consists of an equal amount of positive and negative sentiment examples for all of them. The datasets differ with respect to the guarded attribute ratio. For example, a ratio of 0.8 means that 0.8 of the positive and negative sentiment class examples are composed of AAE speakers, and 0.2 is composed of SAE speakers. We experiment with ratios of 0.5, 0.6, 0.7 and 0.8. The larger the ratio, the higher the classifier's tendency to make use of protected attributes to make its prediction.

Evaluation Measures We report the accuracy of the methods on the sentiment analysis task. To measure fairness, we use the difference in true positive rate (TPR-gap) between individuals belonging to different guarded attributes groups (Hardt et al., 2016; Ravfogel et al., 2020). The rationale behind TPR-gap is that for an equal opportunity, a positive outcome must be independent of the guarded attribute ($\mathbf{z}$), conditional on ($\mathbf{y}$) being an actual positive. See Hardt et al. (2016) for more details.

Results Table 2 presents our results for the fair sentiment classification. For the first three ratios, 0.5, 0.6, and 0.7, we can see that both SAL ($k = 1, 2$) and INLP maintain most of the main-task performance. In debiasing (TPR-Gap), SAL with $k = 2$ significantly outperforms INLP. As expected, removing two directions results in better debiasing than removing one, but it does not lead to a performance drop on the main task. While for the last ratio, 0.8, INLP achieves the highest TPR-gap result, it comes at the cost of a sharp performance drop on the main task, resulting in a nearly random classifier. SAL ($k = 1, 2$) maintains most of the main-task performance, and for $k = 2$, the TPR-gap is halved.

6.2.2 Fair Profession Classification

Task and Data The second task is profession classification. De-Arteaga et al. (2019) attempt to quantify the bias in automatic hiring systems and show that even for a simple task, predicting candidate profession based on a self-provided short biography, significant gaps result from the writer's gender. This might influence the open positions an automatic system will recommend to a candidate, thus favoring candidates from one gender over the other. We hence follow the setup of De-Arteaga et al. (2019), who experiment with professions classification ($\mathbf{y}$), from short biographies ($\mathbf{x}$), and gender as a guarded attribute ($\mathbf{z}$). We use a multiclass classifier to predict the profession, as there are 28 profession classes. We experiment with two types of text representations, FastText (Joulin et al., 2016), based on bag of word embeddings (BWE) and BERT (Devlin et al., 2018) encodings.

	Sentiment				TPR-Gap			
Rt	Orig.	INLP	SAL, $k = 1$	SAL, $k = 2$	Orig.	INLP	SAL, $k = 1$	SAL, $k = 2$
0.5	0.76	0.76	0.76	0.76	0.14	↓0.02 0.12	0.14	↓0.03 0.11
0.6	0.75	0.75	0.75	0.75	0.22	↓0.03 0.19	0.22	↓0.13 0.09
0.7	0.74	0.74	0.74	0.74	0.31	↓0.05 0.26	0.31	↓0.15 0.11
0.8	0.72	↓0.2 0.52	0.72	0.72	0.40	↓0.39 0.01	↓0.04 0.36	↓0.22 0.18

Table 2: The sentiment analysis scores (we use accuracy, as the dataset is balanced) and TPR differences (lower is better) as a function of the ratio of tweets (Rt) written by black individuals and conveying positive sentiment. Arrows with numbers indicate absolute increase/decrease from the baseline, and their background color indicates a difference with positive implications (green) or negative ones (red).

	Accuracy (profession)				TPR-Gap (RMS)			
Encoder	Orig.	INLP	SAL, $k = 1$	SAL, $k = 2$	Orig.	INLP	SAL, $k = 1$	SAL, $k = 2$
FastText	0.75	↓0.05 0.71	↑0.01 0.76	↑0.01 0.76	0.20	↓0.11 0.09	↓0.02 0.18	↓0.08 0.12
BERT	0.8	↓0.11 0.69	↓0.02 0.78	↓0.02 0.78	0.21	↓0.15 0.06	↓0.04 0.17	↓0.12 0.09

Table 3: The profession classification on the biographies dataset results. We report accuracy and TPR-RMS. The number of classes is 28.

Evaluation Measures We report accuracy for the profession classification. For bias level measurement, we use a generalization of TPR-gap for multi-class, suggested by De-Arteaga et al. (2019), calculating the root mean square (RMS) of the TPR with respect to all classes.

De-Arteaga et al. (2019) also provided evidence for a strong correlation between TPR-gap and existing gender imbalances in occupations, which may lead to unfair classification.

Results Table 3 presents the profession classification results. Similar to the sentiment analysis task, SAL ($k = 1, 2$) maintains most of the main-task performance, and for $k = 2$, the two-direction removal, the TPR-gap is lower. When comparing SAL ($k = 2$) to INLP, we observe a clear trade-off between maintaining the main task performance (SAL, $k = 2$) and low TPR-gap scores (INLP).

6.3 Scarce Protected Attribute Labels

For many real-world applications, obtaining large amounts of labeled data for protected attributes can be costly, labor-intensive, and in some cases, infeasible due to an ever-increasing number of privacy regulations. In this analysis, we stress-test our algorithm by simulating a scenario in which only a limited amount of samples from the main task are coupled with the desired protected attribute labels. For this purpose, we replicate the fair sentiment classification experiments, but this time, feeding only a fraction of the annotated data to our debiasing method. In terms of the main task, the experiment is identical, i.e., we use 100K samples for training the sentiment classifier. We experiment with different fractions of the debiasing data, i.e., 5%, of the sentiment training data contains labels about the protected attribute. We hence feed 5,000 samples for debiasing. The subsets for debiasing are chosen randomly. We repeat each experiment 10 times with different subsets. Table 4 presents our results. Using a small fraction of the data for debiasing did not significantly affect SAL's ($k = 1, 2$) main-task performance. INLP, on the other hand, suffers from a sharp performance decrease, resulting in a near-random sentiment classifier. SAL's ($k = 1, 2$) ability to debias the data is slightly worse than in the complete dataset setting but the resulting representations are still significantly less biased than the original ones. INLP achieves low TPR gaps, but it is hard to determine if this is due to an accurate bias removal or a result of corrupting the representations.

6.4 Kernel Experiments

Despite their flexibility in modeling rich feature functions, kernels have been documented to be computationally intensive. Lack of computational resources prevented us from using the full sentiment and bios datasets for our kernel experiments, and instead, we use $15,000$ training examples and $7,998$ test set examples (the full test set) for the sentiment dataset and $15,000$ training examples and $5,000$ examples for the profession dataset. For training on the acquired $15,000$ training examples, we used one Intel Xeon E5-2407 CPU, running at 2.2 GHz, for approximately five hours (for a time complexity analysis, see §5.3).

Table 5 shows that using only a small subset of the data, kSAL-poly2 reduces the TPR gaps

	Sentiment				TPR-Gap			
Ratio	Orig.	INLP	SAL, $k = 1$	SAL, $k = 2$	Orig.	INLP	SAL, $k = 1$	SAL, $k = 2$
0.5	0.76	↓0.19 0.57	0.76	0.76	0.14	↓0.12 0.02	↑0.27 0.41	↓0.03 0.11
0.6	0.75	↓0.16 0.59	0.75	0.75	0.22	↓0.19 0.03	↑0.01 0.23	↓0.13 0.09
0.7	0.74	↓0.17 0.57	↓0.01 0.73	↓0.01 0.73	0.31	↓0.26 0.05	0.31	↓0.19 0.12
0.8	0.72	↓0.15 0.57	↓0.01 0.71	↓0.01 0.71	0.40	↓0.32 0.08	↓0.08 0.34	↓0.27 0.17

Table 4: The sentiment analysis experiments, 100K samples are use to train the sentiment classifier, but only 5K examples are used for learning to remove bias. The test set is identical to the one used in §6

Sentiment Analysis (DeepMoji)				
	Main		TPR-Gap	
k	poly2	rbf	poly2	rbf
0	0.75	0.76	0.14	0.15
1	0.75	0.76	0.14	0.15
2	0.75	0.76	↓0.01 0.13	↓0.03 0.12
Profession Classification (BERT)				
	Main		TPR-Gap (RMS)	
k	poly2	rbf	poly2	rbf
0	0.77	0.61	0.33	0.23
1	0.77	↑0.07 0.68	↓0.05 0.28	↑0.11 0.34
2	0.77	↑0.07 0.68	↓0.08 0.25	↑0.08 0.31

Table 5: Kernel results with kSAL for sentiment (for a ratio of 0.5) and profession classification.

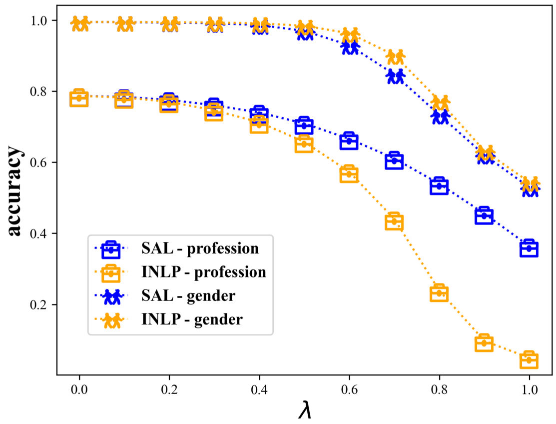


Figure 3: Gender and profession classifications as a function of the interpolation coefficient λ.

while maintaining almost identical performance to the original model on both the sentiment analysis and profession classification tasks. For the sentiment analysis task, kSAL-RBF slightly improves the main task results while reducing the TPR-gap (RMS). For the RBF profession classification task, the results are unexpected, with main task performance increasing as we remove principal directions. This could be due to the pruning of the rich, infinite feature space RBF kernel represents (we also observe significant overfitting with RBF).[6]

6.5 Perturbed Inputs

While the transformation through $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$ maps $\mathbf{x}$ back into the original vector space (as a projection), it often turns out that it removes information in such a way that the *original* classifier (trained on data without removal) can no longer be used with the inputs after removal. This issue exists not only with our algorithm, but also with INLP, and indeed, like us, Ravfogel et al. (2020) re-trained their classifier after they created the cleaned projected inputs.

Ideally, we would want to remove information without necessarily having to retrain a classifier for the main task, as this is costly and perhaps unattainable. To test the effect of such an approach, we interpolated $\overline{\boldsymbol{U}\boldsymbol{U}}^\top$ with the identity matrix, to eventually project $\mathbf{x}$ using $\lambda\overline{\boldsymbol{U}\boldsymbol{U}}^\top + (1-\lambda)\boldsymbol{I}$ for $\lambda \in \{0, 0.1, \ldots, 1.0\}$. This approach weakens the impact of the removal projection and retains some of the information in $\mathbf{x}$. While an adversary can attack this approach,[7] it can mitigate the effects of privacy violations in cases where the service or software used with the modified representations cannot be retrained, especially if the service providers have no malicious intent.

Figure 3 describes an ablation experiment, ranging λ as above on the bios dataset. We see that as we increase the intensity of the use of the SAL projection (increasing λ), the accuracy of both gender prediction and profession prediction decrease when training the original classifier on the non-projected inputs. While the behavior is similar for the gender accuracy for both INLP and our method, the decrease for the profession prediction is much sharper for $\lambda > 0.4$ with INLP.

---

[6] With INLP, RBF-kernel SVM also obtains low-accuracy results.

[7] Consider that the matrix $\lambda\overline{\boldsymbol{U}\boldsymbol{U}}^\top + (1-\lambda)\boldsymbol{I}$ could be invertible for $\lambda < 1$.

Task	WED	FSC	FPCF	FPCB
SAL	0.03 sec	0.37 sec	0.16 sec	0.35 sec
INLP	50 sec	100 min	7 min	35 min

Table 6: A run-time comparison between SAL and INLP. We used 2.20GHz Intel Xeon E5-2407 CPU for all of the experiments. WED, FSC, FSCF, and FPCB stand for word embedding debiasing, fair sentiment classification, and fair profession classification (with both FastText and BERT based representations).

6.6 Runtime of SAL

We measure the time it takes both methods to learn a projection matrix for a given training set. Once we have a projection matrix, debiasing the data is done by multiplying the data representation matrix by the learned projection matrix. Since matrix multiplication is a common practice for many research disciplines, and both methods use it, we do not benchmark it as well. Table 6 presents the run-time differences between SAL and INLP. For all of the experiments, SAL runtime is smaller by at least three orders of magnitude than INLP runtime.

7 Related Work

In their influential work, Bolukbasi et al. (2016) revealed that word embeddings for many gender-neutral terms show a gender bias. Zhao et al. (2018) presented a customized training scheme for word embeddings, which minimizes the negative distances between words in the two groups, e.g., male and female related words, for gender debiasing. Gonen and Goldberg (2019) demonstrated that bias is remains deeply intertwined in word embeddings even after using the above methods. For example, they showed several methods that can accurately predict the gender associated with gender-neutral words, even after applying the methods mentioned above. Similar to Ethayarajh et al. (2019), they concluded that removing a small number of intuitively selected gender directions cannot guarantee the elimination of bias. Motivated by this conclusion, Ravfogel et al. (2020) presented iterative null space projection (INLP). This debasing algorithm iteratively projects features into a space where a linear classifier cannot predict the guarded attribute. The debiased representations are *linearly guarded*, i.e., they cannot guarantee bias removal beyond the linear level. Indeed, they show a simple nonlinear classifier can achieve high accuracy when predicting the guarded attribute. Their approach is also related to that of Xu et al. (2017). Previous work uses adversarial methods (Ganin et al., 2016) for information removal (Edwards and Storkey, 2015; Li et al., 2018; Coavoux et al., 2018; Elazar and Goldberg, 2018; Barrett et al., 2019; Han et al., 2021) with the one by Ravfogel et al. (2022) being related to ours through the use of the mini-max theorem with squared-error loss on the reconstruction of a matrix similar to our covariance matrix. In addition, methods based on similarity measures between neural representations (Colombo et al., 2022) were developed. To support the increasing interest in fair classification, Han et al. (2022) presented an open-source framework for standardizing the evaluation of debiasing methods. Finally, most relevant to this paper is an extension of SAL to the unaligned case, where protected attributes are not paired with input examples (Shao et al., 2023).

8 Conclusions

We presented a method for removing information from learned representations. We extended our method by using kernels, showing we can provide an effective nonlinear guarding. We also experimented with real-world low-resource situations, in which only a small guarded attribute dataset is provided for information removal.

Limitations

There are two main technical limitations to our work: (a) while the kernel removal is nonlinear, it still depends on a feature representation that captures a specific type of nonlinearities; (b) like other kernel methods, the kernel removal method is significantly slower than direct SVD removal in cases where the feature representations can be written out without the need of an implicit kernel. Future work may apply random projections to the kernel matrices to decompose them more efficiently.

A general limitation of current methods of information removal is that they are only able to remove information with respect to a specific class of classifiers. It could always be the case that complex correlations between the inputs and the guarded attributes exist, and that an adversary can try to exploit them to predict the guarded attribute if this class of classifiers is not too complex. Our use of kernels alleviates some of this issue, though not completely.

Finally, experimentally, we focus on text only in English. It is not clear to what extent our method generalizes to other languages in a useful manner, especially when morphology is rich, and the neural representations encode important information for the task at hand, but that information would be removed by our method.

Ethical Considerations

Public trust plays a significant role in the broad applicability of NLP in real-world scenarios, especially in critical situations that may directly impact people's lives. NLP research of the kind presented in this paper helps this issue take the spotlight it deserves. However, we discourage NLP practitioners from using our method (and similar methods) as an out-of-the-shelf solution in deployed systems. We recommend investing a significant amount of time and effort in understanding the applicability and universality of our method to the debiasing of representations. Issues such as expected type of adversariality or tolerance level for drop in system performance need to be considered.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. We especially appreciate the comment provided by one of the reviewers regarding our title. Particularly that it could be misinterpreted as an indication of frustration at rejections of our paper ("gold") in previous conferences. Rather, the "gold" in our case is the principal vectors with low intensity, which are pruned in most use cases of SVD. We also thank Shauli Ravfogel for providing support with the code for INLP, Ryan Cotterell for discussions and Matt Grenander for feedback on earlier drafts. The experiments in this paper were supported by compute grants from the Edinburgh Parallel Computing Center (Cirrus) and from the Baskerville Tier 2 HPC service (University of Birmingham).

References

Maria Barrett, Yova Kementchedjhieva, Yanai Elazar, Desmond Elliott, and Anders Søgaard. 2019. Adversarial removal of demographic attributes revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6330–6335.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of african-american english. *arXiv preprint arXiv:1608.08868*.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29:4349–4357.

Maximin Coavoux, Shashi Narayan, and Shay B Cohen. 2018. Privacy-preserving neural representations of text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1–10.

Pierre Colombo, Guillaume Staerman, Nathan Noiry, and Pablo Piantanida. 2022. Learning disentangled textual representations via statistical measures of similarity. *arXiv preprint arXiv:2205.03589*.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Susan T Dumais. 2004. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230.

Harrison Edwards and Amos Storkey. 2015. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897*.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. *arXiv preprint arXiv:1808.06640*.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Understanding undesirable word embedding associations. *arXiv preprint arXiv:1908.06361*.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. *arXiv preprint arXiv:1903.03862*.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1406–1414.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765.

Xudong Han, Aili Shen, Yitong Li, Lea Frermann, Timothy Baldwin, and Trevor Cohn. 2022. fairlib: A unified framework for assessing and improving classification fairness. *arXiv preprint arXiv:2205.01876*.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29:3315–3323.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30.

Thomas Manzini, Yao Chong Lim, Yulia Tsvetkov, and Alan W Black. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. *arXiv preprint arXiv:1904.04047*.

Ella Rabinovich, Yulia Tsvetkov, and Shuly Wintner. 2018. Native language cognate effects on second language lexical choice. *Transactions of the Association for Computational Linguistics*, 6:329–342.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*.

Shauli Ravfogel, Michael Twiton, Yoav Goldberg, and Ryan D Cotterell. 2022. Linear adversarial concept erasure. In *International Conference on Machine Learning*, pages 18400–18421. PMLR.

Shun Shao, Yftah Ziser, and Shay B. Cohen. 2023. Erasure of unaligned attributes from neural representations. *arXiv preprint arXiv:2302.02997*.

Ke Xu, Tongyi Cao, Swair Shah, Crystal Maung, and Haim Schweitzer. 2017. Cleaning the null space: A privacy mechanism for predictors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018. Learning gender-neutral word embeddings. *arXiv preprint arXiv:1809.01496*.

A Eigenvectors of Λ

We turn to the following Lemma used in §5.2.

Lemma 1. *Let* $\mathbf{w}$ *be an eigenvector associated with eigenvalue* $\lambda \in \mathbb{R}$ *for* $\mathbf{\Gamma} = \boldsymbol{K}_\phi \boldsymbol{K}_\psi$*. Then* $\mathbf{\Phi w}$ *is an eigenvector of* $\mathbf{\Omega\Omega}^\top$.

Proof. Since $\mathbf{w}$ is an eigenvector of $\mathbf{\Gamma}$, it holds that $\Gamma\mathbf{w} = \lambda\mathbf{w}$. Therefore:

$$\mathbf{\Psi}^\top\mathbf{\Psi}\mathbf{\Phi}^\top\mathbf{\Phi}\mathbf{w} = \lambda\mathbf{w},$$

$$\left(\mathbf{\Phi}\mathbf{\Psi}^\top\mathbf{\Psi}\mathbf{\Phi}^\top\right)\mathbf{\Phi}\mathbf{w} = \lambda\mathbf{\Phi}\mathbf{w},$$

and therefore $\mathbf{\Phi w}$ is an eigenvalue of

$$\mathbf{\Omega\Omega}^\top = \mathbf{\Phi}\mathbf{\Psi}^\top\mathbf{\Psi}\mathbf{\Phi}^\top.$$

□

B Nearest Neighbors Test for Word Embedding Debiasing

We give in Table 7 the ten nearest neighbor words for ten random words from the data, before and after using SAL. The neighboring words are determined through cosine similarity of the corresponding embeddings with respect to the pivot word embedding. We observe little to no difference in these two lists (before and after the removal).

Words	Nearest neighbors (before)	Nearest neighbors (after)
lobbying	lobbyists, lobbyist, campaigning	lobbyists, lobbyist, campaigning
once	again, then, when	again, then, when
parliament	parliamentary, mps, elections	parliamentary, mps, elections
dashboard	dashboards, smf, powered	dashboards, smf, powered
cumulative	gpa, accumulative, aggregate	gpa, accumulative, aggregate
foam	rubber, mattress, polyurethane	rubber, mattress, polyurethane
rh	lh, bl, r	lh, bl, graphite
genetically	gmo, gmos, genetic	gmo, gmos, genetic
inner	outer, inside, innermost	outer, inside, innermost
harvest	harvesting, harvests, harvested	harvesting, harvests, harvested
secretary	deputy, minister, treasurer	deputy, minister, secretaries
ruth	helen, esther, margaret	helen, esther margaret
charlotte	raleigh, nc, atlanta	raleigh, nc, atlanta
abigail	hannah, lydia, eliza	hannah, lydia, samuel
sophie	julia, marie, lucy	julia, lucy, claire
nichole	nicole, kimberly, kayla	nicole, kimberly, mya
emma	emily, lucy, sarah	emily, watson, sarah
david	stephen, richard, michael	alan, stephen, richard
richard	robert, william, david	robert, william, david
joseph	francis, charles, thomas	mary, francis, charles
thomas	james, william, john	james, william, henry
james	john, william, thomas	william, john, thomas

Table 7: Nearest neighbor test on GloVe word embeddings before and after debiasing on gender. The upper block includes a random set of words, while the middle and bottom block include female and male names.

CTC Alignments Improve Autoregressive Translation

Brian Yan*[1] **Siddharth Dalmia***[1] **Yosuke Higuchi**[2]
Graham Neubig[1] **Florian Metze**[1] **Alan W Black**[1] **Shinji Watanabe**[1,3]
[1]Language Technologies Institute, Carnegie Mellon University, USA
[2]Department of Communications and Computer Engineering, Waseda University, Japan
[3]Human Language Technology Center of Excellence, Johns Hopkins University, USA
`{byan, sdalmia}@cs.cmu.edu`

Abstract

Connectionist Temporal Classification (CTC) is a widely used approach for automatic speech recognition (ASR) that performs conditionally independent monotonic alignment. However for translation, CTC exhibits clear limitations due to the contextual and non-monotonic nature of the task and thus lags behind attentional decoder approaches in terms of translation quality. In this work, we argue that CTC *does* in fact make sense for translation if applied in a joint CTC/attention framework wherein CTC's core properties can counteract several key weaknesses of pure-attention models during training and decoding. To validate this conjecture, we modify the Hybrid CTC/Attention model originally proposed for ASR to support text-to-text translation (MT) and speech-to-text translation (ST). Our proposed joint CTC/attention models outperform pure-attention baselines across six benchmark translation tasks.

1 Introduction

Automatic speech recognition (ASR), machine translation (MT), and speech translation (ST) have conspicuous differences but are all closely related sequence-to-sequence problems. Researchers from these respective fields have long recognized the opportunity for cross-pollinating ideas (He and Deng, 2011), starting from the coupling of statistical ASR (Huang et al., 2014) and MT (Al-Onaizan et al., 1999) which gave rise to early approaches for ST (Waibel, 1996; Ney, 1999). Notably in the end-to-end era, attentional encoder-decoder approaches emerged in both MT (Bahdanau et al., 2015) and ASR (Chorowski et al., 2015; Chan et al., 2016), rising to great prominence in both fields.

During this same period, there has been another prominent end-to-end approach in ASR: Connectionist Temporal Classification (CTC) (Graves et al., 2006). Unlike the highly flexible attention mechanism which can handle ASR, MT, and ST alike, CTC models sequence transduction as a monotonic alignment of inputs to outputs and thus fits more naturally with ASR than it does with translation. Still, many interested in non-autoregressive translation have applied CTC to MT (Libovický and Helcl, 2018) and ST (Inaguma et al., 2021b) and promising techniques have emerged, shrinking the gap between autoregressive approaches (Saharia et al., 2020; Gu and Kong, 2021; Chuang et al., 2021; Huang et al., 2022). These recent developments suggest that the latent alignment ability of CTC is a promising direction for translation – this leads us to question: *can CTC alignments improve autoregressive translation?* In particular, we are interested in frameworks that leverage the strength of CTC while minimizing its several harmful incompatibilities (see §3) with translation tasks.

Inspired by the success of Hybrid CTC/Attention in ASR (Watanabe et al., 2017), we investigate jointly modeling CTC with an autoregressive attentional encoder-decoder for translation. Our conjecture is that the monotonic alignment and conditional independence of CTC, which weaken purely CTC-based translation, counteract particular weaknesses of attentional models in joint CTC/attention frameworks. In this work, we seek to investigate *how* each CTC property interacts with corresponding properties of the attentional counterpart during joint training and decoding. We design a joint CTC/attention architecture for translation (§4) and then examine the positive interactions which ultimately result in improved translation quality compared to pure-attention baselines, as demonstrated on the IWSLT (Cettolo et al., 2012), MuST-C (Di Gangi et al., 2019), and MTedX (Salesky et al., 2021) MT/ST corpora (§6).[1]

2 Background: Joint CTC/Attn for ASR

Both the CTC (Graves et al., 2006) and attentional encoder-decoder (Bahdanau et al., 2015) frame-

---

[1] Models are are available in ESPnet. For ST, refer to `egs2/must_c_v2/st1` and for MT refer to `egs2/iwslt14/mt1`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1623–1639
May 2-6, 2023 ©2023 Association for Computational Linguistics

CTC	ATTENTION	JOINT CTC/ATTENTION	ASR	MT/ST
$P_{\text{CTC}}(Y\|X) \triangleq \sum_{Z \in \mathcal{Z}} \prod_{t=1}^{T} P(z_t\|X, \cancel{z_{1:t-1}})$	$P_{\text{Attn}}(Y\|X) \triangleq \prod_{l=1}^{L} P(y_l\|y_{1:l-1}, X)$	$P_{\text{Joint}}(Y\|X) \triangleq P_{\text{CTC}}(Y\|X)^{\lambda} \times P_{\text{Attn}}(Y\|X)^{1-\lambda}$	✓	✓
Hard Alignment Criterion only allows monotonic alignments of inputs to outputs	**Soft Alignment** Flexible attention-based input-to-output mappings may overfit to irregular patterns	**During Training:** Hard alignment objective produces stable encoder representations allowing the decoder to more rapidly learn soft alignment patterns	✓	**L1** See §3
Conditional Independence Assumes that there are no dependencies between each output unit given the input	**Conditional Dependence** Locally normalized models with output dependency exhibit label/exposure biases	**During Decoding:** Use of conditionally independent likelihoods in joint scoring eases the exposure/label biases from conditionally dependent likelihoods	✓	**L2** See §3
Input-Synchronous Emission Each input representation emits exactly one blank or non-blank output token	**Autoregressive Generation** Need to detect end-points and compare hypotheses of different length in beam search	**During Decoding:** Input-synchronous emission determines output length based on input length counteracting the autoregressive end-detection problem	✓	**L3** See §3

Table 1: Description of three reasons why joint CTC/attention modeling is powerful in ASR. In order to understand whether these positive interactions between properties of the CTC and attention frameworks are applicable to MT/ST, we must address three corresponding concerns, L1-3, about the applicability of CTC to translation (§2).

works seek to model the Bayesian decision seeking the output, $\hat{Y}$, from all possible sequences, $\mathcal{V}^{\text{tgt}}*$, by selecting the sequence which maximizes the posterior likelihood $P(Y|X)$, where $X = \{\mathbf{x}_t \in \mathcal{S}^{\text{src}} | t = 1, ..., T\}$ and $Y = \{y_l \in \mathcal{V}^{\text{tgt}} | l = 1, ..., L\}$. The source set $\mathcal{S}^{\text{src}}$ is a discrete vocabulary in the MT case and a continuous real space in the ST case while the target set $\mathcal{V}^{\text{tgt}}$ is always a discrete vocabulary. Note that the T-length of the input is assumed to be longer than the L-length output for speech tasks (Graves et al., 2006), but this is not necessarily true for MT.

What are the critical differences between the CTC and attention frameworks? As shown in the first two columns of Table 1, CTC and attention offer different formulations of the posterior likelihood, $P_{\text{CTC}}(\cdot)$ and $P_{\text{Attn}}(\cdot)$ respectively. First of all, the attention mechanism is a flexible input-to-output mapping function which allows a decoder to perform **soft alignment** of an output unit y_l to multiple input units $\mathbf{x}_{[...]}$ without restriction. One downside of this flexibility is a risk of destabilized optimization (Kim et al., 2017). CTC on the other hand marginalizes the likelihoods of all possible input to alignment sequence, $Z = \{z_t \in \mathcal{V}^{\text{tgt}} \cup \{\varnothing\} | t = 1 \dots T\}$, mappings via **hard alignment** where each output unit z_t maps to a single input unit $\mathbf{x}_t$ in a strictly monotonic pattern. $\varnothing$ is a "blank" and Z maps deterministically to Y by removing blanks and repeated emissions.

Secondly, the attentional decoder models each output unit y_1 with **conditional dependence** on not only the input X, but also the previous output units $y_{1:l-1}$. In contrast, CTC makes a **conditional independence** assumption that each z_t does not depend on $z_{1:t-1}$ if already conditioned on X (as denoted by the strike-through in Table 1) – this is a strong assumption which allows for efficient computation of marginalized likelihoods over all $Z \in \mathcal{Z}(Y, T)$ via dynamic programming. On the plus, since CTC does not model causality between output units it is not plagued by the same label and exposure biases that exist in attentional decoders due to local normalization of causal likelihoods (Bottou, 1991; Ranzato et al., 2016; Hannun, 2019).

Finally, the attentional decoder is an **autoregressive generator** that decodes the output until a stop token, <eos>. Comparing likelihoods for sequences of different lengths requires a heuristic brevity penalty. Furthermore label bias with respect to the stop token manifests as a length problem where likelihoods degenerate for unexpectedly long outputs (Murray and Chiang, 2018). In comparison, CTC is an **input-synchronous emitter** that consumes an input unit in order to produce an output unit. Therefore, CTC cannot produce an output longer than the input representation which feeds the final posterior output layer – but this also means that CTC does not require end detection.

As previously shown by (Kim et al., 2017; Watanabe et al., 2017), jointly modeling CTC and an attentional decoder is highly effective in ASR. The foundation of this architecture is a shared encoder, ENC, which feeds into both CTC, $P_{\text{CTC}}(\cdot)$, and attentional decoder, $P_{\text{Attn}}(\cdot)$, posteriors:

$$\mathbf{h} = \text{Enc}(X) \tag{1}$$

$$P_{\text{CTC}}(z_t|X) = \text{CTC}(\mathbf{h}_t) \tag{2}$$

$$P_{\text{Attn}}(y_l|X, y_{1:l-1}) = \text{Dec}(\mathbf{h}, y_{1:l-1}) \tag{3}$$

where $\text{CTC}(\cdot)$ denotes a linear projection to the CTC output vocabulary, $\mathcal{V}^{\text{tgt}} \cup \{\varnothing\}$, followed by softmax. $\text{DEC}(\cdot)$ denotes autoregressive decoder layers followed by a linear projection to the decoder output vocabulary, $\mathcal{V}^{\text{tgt}} \cup \{\text{<eos>}\}$, and softmax. The joint network is optimized via a multi-tasked objective, $\mathcal{L}^{\text{ASR}} = \mathcal{L}^{\text{ASR}}_{\text{CTC}} + \lambda \mathcal{L}^{\text{ASR}}_{\text{Attn}}$, where λ interpolates the CTC and decoder cross-entropy losses.

Joint decoding is typically performed with a one-pass beam search where CTC plays a secondary role as a joint scorer while attention leads the major hypothesis expansion and end detection functions in the algorithm (Watanabe et al., 2017; Tsunoo et al., 2021). However, CTC is capable of taking over the lead role if called upon (e.g. for streaming applications) (Moritz et al., 2019).

3 Potential CTC Limitations in MT/ST

Why exactly does this joint CTC/attention framework perform so well in ASR? As summarized in column 3 of Table 1, we are particularly interested in three reasons which arise from the combination of the hard vs. soft alignment, conditional independence vs. dependence, and input-synchronous emission vs. autoregressive generation properties of CTC and attention respectively. These dynamics have become well understood in ASR, owing to the popularity of the joint framework (Watanabe et al., 2018) amongst ASR practitioners.

So can CTC and attention also complement each other when applied jointly to translation?[2] ASR, MT, and ST can all be generalized as sequence transduction tasks following the Bayesian formulation. Attentional decoders have been a predominant technical solution to each of these tasks. However, the CTC framework appears to have several limitations specific to MT/ST that are not present in ASR; this seemingly diminishes the promise of the joint CTC/attention framework for translation. In this work, we seek to address the following three concerns about MT/ST CTC which appear to inhibit the CTC/attention framework (please refer back to Table 1 as needed).

L1 *Can CTC encoders perform sophisticated input-to-output mappings required for translation?*

Unlike ASR, translation entails non-monotonic mappings due to variable word-ordering across languages. Additionally, inputs may be shorter than outputs as mappings are not necessarily one-to-one. Furthermore, the mapping task for ST is compositional where logically a speech signal first maps to a source language transcription before being mapped to the ultimate translation. All of these complications appear to directly contradict the **hard alignment** of CTC. If CTC cannot produce stable encoder representations for MT/ST, then during joint training attention does not receive the optimization benefit as in ASR (per row 2 of Table 1). Fortunately, prior works suggest that these challenges are not insurmountable. Chuang et al. (2021) showed that self-attentional encoders can perform latent model variable word orders for ST, Libovický and Helcl (2018); Dalmia et al. (2022) proposed upsampling encoders that produce expanded input representations for MT, and Sanabria and Metze (2018); Higuchi et al. (2022) proposed hierarchical CTC encoders that can compose multiple output resolutions for ASR. In §4.1, we incorporate these techniques into a unified hierarchical CTC encoding method for MT/ST which is capable of sophisticated input-to-output mappings.

L2 *Does CTC-based translation quality lag too far behind attention-based to be useful?*

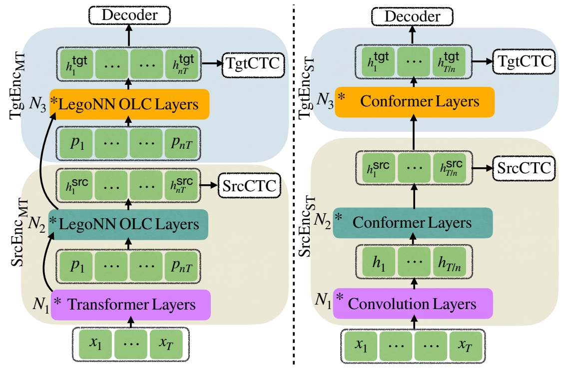
CTC-based ASR has recently shown competitive performance due in large part to improved neural architectures (Gulati et al., 2020) and self-supervised learning (Baevski et al., 2020; Hsu et al., 2021), but the gap between CTC and attention for translation appears to be greater (Gu and Kong, 2021). Perhaps the **conditional independence** of CTC inhibits the quality to such a degree in MT/ST where these likelihoods cannot ease the label/exposure biases of the attentional decoder as they do in ASR (per row 3 of Table 1). The relative weakness of non-autoregressive translation approaches has been well-studied. Knowledge distillation (Kim and Rush, 2016; Zhou et al., 2019) and iterative methods (Qian et al., 2021; Chan et al., 2020; Huang et al., 2022) all attempt to bridge the gap between non-autoregressive models and their autoregressive counterparts. In §6, we address this concern empirically; we find that even CTC models with 28% relative BLEU reduction compared to attention yield improvements when CTC and attention are jointly decoded.

L3 *Is the alignment information produced by CTC-based translation models reasonable?*

In ASR, CTC alignments are reliable enough to segment audio data by force aligning inputs to target transcription outputs (Kürzinger et al., 2020) and exhibit minimal drift compared to hidden Markov models (Sak et al., 2015). However,

---

[2] This particular question has not been addressed in literature. For an account of related works, please see §8.



(a) Hierarchical MT Encoder

(b) Hierarchical ST Encoder

Figure 1: Hierarchical MT/ST encoders where representations are first up/down-sampled by $\textsc{SrcEnc}_{\text{MT/ST}}$ and then re-ordered by $\textsc{TgtEnc}_{\text{MT/ST}}$.

CTC alignments are not as well studied in translation. It is an open question of whether or not the **input-synchronous emission** of CTC for translation has sufficient alignment quality to support the end detection responsibility during joint decoding as it does in ASR (per row 4 of Table 1). Ideally, the CTC alignments are strong enough such that CTC can lead joint decoding by proposing candidates for hypothesis expansion in each beam step until all input units are consumed (at which point the end is detected), as in an input-synchronous beam search. More conservatively, the CTC alignments may be too unreliable to take the lead but could still guide the attentional decoder's end detection by penalizing incorrect lengths via joint scoring, as in an output-synchronous beam search. In §4.2, we lay out comparable forms for input and output-synchronous beam search which allows us to examine the impact on translation quality depending on whether CTC is explicitly responsible for or only implicitly contributing to end detection.

4 Joint CTC/Attention for Translation

4.1 Hierarchical CTC Encoding

Per L1 described in §3, we seek to build a CTC encoder for translation which handles sophisticated input-to-output mappings. Unlike ASR where outputs are assumed to be 1) always shorter than inputs and 2) monotonic with respect to inputs, translation needs to account for variability of lengths and and word orderings. We therefore propose to use a hierarchical CTC encoding scheme which 1) aligns inputs to length-adjusted *source*-oriented encodings before 2) aligning to re-ordered *target*-oriented encodings, as shown in Figure 1. Our encoding process thus consists of two compartmentalized functions: length-adjustment and re-ordering.

Length-adjustment For MT, we up-sample the lengths of the source-oriented encodings in order to output sequences longer than the input. For ST, we down-sample the lengths of the source-oriented encodings to coerce a discrete textual representation of the real-valued speech input. We enforce source orientations using CTC criteria that seek to align the length adjusted intermediate layer encoder representations towards source text sequences (for MT this is the same as the input and for ST this is the ASR target). By compartmentalizing length-adjustment within this initial stage, we allow subsequent encoder layers to focus solely on re-ordering.

Re-ordering We then obtain target-oriented encodings with subsequent encoder layers, where re-ordering is enforced using CTC criteria that seek to align final layer encoder representations towards target text sequences. Critically, the underlying neural network architecture must be able to model latent re-ordering as the CTC criterion itself will only consider monotonic alignments of the final encoder representation to the target.

Our proposed MT/ST hierarchical encoders consist of the following components:

$$\mathbf{h}^{\text{SRC}} = \textsc{SrcEnc}_{\text{MT/ST}}(X) \tag{4}$$

$$P_{\text{CTC}}(z_t^{\text{SRC}}|X) = \textsc{SrcCTC}_{\text{MT/ST}}(\mathbf{h}_t^{\text{SRC}}) \tag{5}$$

$$\mathbf{h}^{\text{TGT}} = \textsc{TgtEnc}_{\text{MT/ST}}(\mathbf{h}^{\text{SRC}}) \tag{6}$$

$$P_{\text{CTC}}(z_t^{\text{TGT}}|X) = \textsc{TgtCTC}_{\text{MT/ST}}(\mathbf{h}_t^{\text{TGT}}) \tag{7}$$

The hierarchical encoders are jointly optimized with an attentional decoder using a multi-tasked objective, $\mathcal{L} = \mathcal{L}_{\textsc{SrcCTC}} + \lambda_1 \mathcal{L}_{\textsc{TgtCTC}} + \lambda_2 \mathcal{L}_{\textsc{Attn}}$, where λ's interpolate source-oriented CTC, target-oriented CTC, and decoder cross-entropy losses.

As shown in Figure 1.a, $\textsc{SrcEnc}_{\text{MT}}(\cdot)$ consists of N_1 Transformer (Vaswani et al., 2017) layers followed by N_2 up-sampling Output Length Controller (OLC) layers used in LegoNN (Dalmia et al., 2022) – the layer-wise positional embeddings of the OLC architecture enable latent length-adjustment of textual inputs. $\textsc{TgtEnc}_{\text{MT}}(\cdot)$ consists of N_3 non-up-sampling OLC layers – the layer-wise attention of the OLC architecture enables latent re-ordering. Our ST encoder is similar, but uses Conformers (Gulati et al., 2020) to capture the local and global dependencies in speech, as shown in Figure 1.b. $\textsc{SrcEnc}_{\text{ST}}(\cdot)$ consists of N_1 convolutional blocks (Dong et al., 2018) for down-sampling fol-

Algorithm 1 *Output*-Synchronous Step Function: attentional decoder proposes candidates to expand hypotheses which are all of *l*-length at step *l*.

```
procedure OutputStep(prtHs, $X, l, p, \max L$)
    newPrtHs = {}; endHs = {}
    for $y_{1:l-1} \in$ prtHs do
        $\boxed{\text{attnCnds}} = \text{top-k}(P_{\text{Attn}}(y_l|X, y_{1:l-1}), \text{k} = p)$
        for $c \in$ attnCnds do
            $y_{1:l} = y_{1:l-1} \oplus \text{c}$
            $\alpha_{\text{CTC}} = \text{CTCScore}(y_{1:l}, \boxed{X_{1:T}})$
            $\alpha_{\text{Attn}} = \text{AttnScore}(y_{1:l}, X_{1:T})$
            $\beta = \text{LengthPen}(y_{1:l})$
            $P_{\text{Beam}}(y_{1:l}|X) = \alpha_{\text{CTC}} + \alpha_{\text{Attn}} + \beta$
            if $\boxed{(c \text{ is <eos>}) \text{ or } (l \text{ is } \max L)}$ then
                endHs[$y_{1:l}$] = $P_{\text{Beam}}(\cdot)$
            else
                newPrtHs[$y_{1:l}$] = $P_{\text{Beam}}(\cdot)$
            end if
        end for
    end for
    return newPrtHs, endHs
end procedure
```

Hypothesis Expansion

Joint Scoring

End Detection

Algorithm 2 *Input*-Synchronous Step Function: CTC proposes candidates to expand hypotheses which are all produced from *t* input units at step *t*.

```
procedure InputStep(prtHs, $X, t, p, T$)
    newPrtHs = {}; endHs = {}
    $\boxed{\text{CTCCnds}} = \text{top-k}(P_{\text{CTC}}(z_t|X), \text{k} = p)$
    for $y \in$ prtHs do
        for $c \in$ CTCCnds do
            if ($c$ is $\varnothing$) or ($c$ is repeat) then
                $\tilde{y} = y$
            else
                $\tilde{y} = y \oplus \text{c}$
            end if
            $\alpha_{\text{CTC}} = \text{CTCScore}(\tilde{y}, \boxed{X_{1:t}})$
            $\alpha_{\text{Attn}} = \text{AttnScore}(\tilde{y}, X_{1:T})$
            $\beta = \text{LengthPen}(\tilde{y})$
            $P_{\text{Beam}}(\tilde{y}|X) = \alpha_{\text{CTC}} + \alpha_{\text{ATTN}} + \beta$
            if $\boxed{t \text{ is } T}$ then
                endHs[$\tilde{y}$] = $P_{\text{Beam}}(\cdot)$
            else
                newPrtHs[$\tilde{y}$] = $P_{\text{Beam}}(\cdot)$
            end if
        end for
    end for
    return newPrtHs, endHs
end procedure
```

lowed by N_2 Conformer layers – this stage is analogous to the ASR sub-task of ST where a long speech signal is length-adjusted to a shorter latent textual representation. TGTENC$_{\text{ST}}(\cdot)$ consists of N_3 Conformer layers – this stage is analogous to the MT sub-task of ST where latent re-ordering is enabled by self-attention. LegoNN and Conformer are further described in §A.4.

4.2 Input/Output-Synchronous Decoding

Per L2 and L3 described in §3, we seek to design a joint decoding algorithm with input and output-synchronous variants of one-pass beam search which differ only in whether CTC or attention takes the leading role. As shown in Algorithms 1 and 2, we propose to align the input and output beam-step functions along three common functions: hypothesis expansion, joint scoring, and end detection. Using these mirrored forms, let us now interpret the respective roles of CTC and attention.

Output-Synchrony Consider first that attention is in the leading role, which means that we are working with an output-synchronous beam search. Note that this is the algorithm originally proposed by Hori et al. (2017). OUTPUTSTEP performs *hypothesis expansion* by computing the attentional decoder's output posterior at label step l, $P_{\text{Attn}}(y_l|X, y_{1:l-1})$ for each partial hypothesis, $y_{1:l-1}$. A pre-beam size, p, is then used to select the top candidate output units (Seki et al., 2019), attnCnds, which are used to expand the partial hypotheses via concatenation, denoted by $\oplus$. In the *joint scoring* block, the attentional decoder likelihood, $\text{AttnScore}(\cdot)$, and length penalty/reward, $\text{LengthPen}(\cdot)$ yield the estimated joint likelihood P_{Beam}. Finally in *end detection*, OUTPUTSTEP must check for the stop token, <eos>, which may be proposed by attnCnds.

Input-Synchrony Now let us consider the differences when CTC is in the leading role. Note that this algorithm extends Hannun et al. (2014)'s CTC beam search algorithm to include joint scoring with attentional likelihoods. INPUTSTEP performs *hypothesis expansion* by computing CTC's alignment posterior at time step t, $P_{\text{CTC}}(z_t|X)$. Unlike in output-synchrony, here each hypothesis expansion also consumes one step of the input. The same pre-beam size, p, is used to select top candidate alignment units, CTCCnds, but partial hypotheses are only expanded for non-blank and non-repeat candidates. The *joint scoring* block is identical to output-synchrony except for one difference: CTC likelihood, $\text{CTCScore}(\cdot)$, is applied over the full input, $X_{1:T}$, in OUTPUTSTEP and over the partial input, $X_{1:t}$, in INPUTSTEP. This difference engenders a speed vs. accuracy trade-off, which we discuss in D2 of §7. Finally, *end detection* simply occurs when all input units have been consumed ($t = T$). Therefore, INPUTSTEP does not require checking for the stop token as all hypotheses at time T are ended.

We propose this particular form of input-

		MODEL TYPE			MT			ST		
#	MODEL NAME	Joint Train?	Joint Decode?	Decoding Method	IWSLT14 De-En	IWSLT14 Es-En	MTedX All-En	MuST-C-v2 En-De	MuST-C-v2 En-Ja	MTedX All-En
1	Pure-Attn (Prior)	✗	✗	Attn Only	(32.2)†	(39.0)†	-$^\diamond$	25.8‡	12.4‡	-$^\diamond$
2	Pure-Attn (Ours)	✗	✗	Attn Only	32.8 (33.7)	39.0 (39.9)	25.6	27.8	14.3	22.7
3	Joint CTC/Attn	✓	✗	CTC Only	27.3	33.8	22.4	24.4	10.2	21.4
4	Joint CTC/Attn	✓	✗	Attn Only	33.6	39.5	28.0	28.3	14.2	23.7
5	Joint CTC/Attn	✓	✓	Joint I-Sync	33.7	39.7	27.8	**29.2**	15.1	**25.1**
6	Joint CTC/Attn	✓	✓	Joint O-Sync	**34.1**	**39.9**	**28.1**	**29.2**	**15.3**	**25.1**

Table 2: Test set performances, as measured by BLEU (↑), of our proposed joint CTC/Attention models compared to pure-attention baselines. Joint CTC/Attention models are always jointly trained, but can be either jointly decoded using input/output synchrony or decoded using only their CTC or attention branches. For IWSLT14, we mention (*tokenized BLEU*) for comparison with prior works: †Raunak et al. (2020) and ‡Inaguma et al. (2020). $^\diamond$Prior MTedX works show only All-All or pair-wise settings.

synchronous beam search in order to exactly mirror the functions of its output-synchronous counterpart; without this mirrored formulation, we cannot attribute differences in decodings to the swapped roles of CTC and attention. For instance, now we can answer questions such as *can CTC perform hypothesis expansion on par with attention*, allowing us to address concerns about applying weaker joint CTC models during decoding per L2 and L3 in §3. To the best of our knowledge, we are the first to examine the theoretical and empirical differences of input and output synchrony through a unified formulation, as discussed further in §7. Other forms of input-synchronous beam search in prior works cannot directly be used for this purpose. Triggered Attention (Moritz et al., 2019) is one such example which is purpose-fit for streaming to a degree where several core components (e.g. look-ahead and re-triggering) cannot trivially be re-factored into an output-synchronous variant.

5 Experimental Setup

Data We examine the efficacy of our proposed approaches on two language pairs for each of the MT and ST tasks. For MT, we use German-to-English (De-En) and Spanish-to-English (Es-En) from IWSLT14 (Cettolo et al., 2012). For ST, we use English-to-German (En-De) and English-to-Japanese (En-Ja) from MuST-C-v2, reporting tst-COMMON results (Di Gangi et al., 2019). We also examine the multilingual setting of 6 European languages to English (All-En) from MTedX (Salesky et al., 2021) for both tasks. Full dataset descriptions for reproducibility are in §B.

Modeling We compare our joint CTC/Attention models to purely attentional encoder-decoder baselines. All proposed and baseline models were tuned separately, using validation sets only, within the same hyperparameter search spaces for training and decoding to ensure fair comparison. All experiments were conducted using ESPnet-ST (Inaguma et al., 2020). Full descriptions of model sizes, hyperparameters, and pre-processing are in §B.[3]

Evaluation: Unless otherwise indicated, we measure performance with detokenized case-sensitive BLEU (Post, 2018) on punctuated 1-references.[4]

6 Results and Analyses

In this section, we first present our main results on 6 benchmark MT and ST tasks. We then present evidence that hierarchical encoding (§4.1) produces stable encoder representations that simplify the decoder's source attention (addressing L1 in §3). Next we present evidence that joint decoding is beneficial despite the fact that CTC-only performance lags behind that of attention-only (addressing L2 in §3). Finally, we present evidence that CTC's alignment information alleviates attention's end-detection problem in both input and output synchronous joint decoding (§4.2) (addressing L3 in §3).

6.1 Joint CTC/Attention Models Outperform CTC-only and Attention-only Baselines

As shown in Table 2, joint CTC/Attention with output-synchronous decoding outperforms pure-

---

[3] We compare our baselines for MuST-C-v2 to the default recipes in ESPnet in Table 2. For back-compatibility with additional prior works using MuST-C-v1 En-De, see §A.2.

[4] Evaluation with additional metrics is provided in §A.1.

attention across *all* MT and ST tasks (line 2 vs. 6). Joint training while only decoding with the attention branch still outperforms pure-attention models without any joint training (line 2 vs. 4). Note that *CTC is consistently the weaker* of the two branches in jointly trained models (line 3 vs. 4). Joint input/output-synchronous decodings yield further improvements overall, confirming that *both joint training and decoding are beneficial* (line 4 vs. 5/6). However, we find that input-synchrony lags behind output-synchrony (line 5 vs. 6); this phenomenon is discussed further in §7.

6.2 Hierarchical Encoding Reduces Attention's Alignment Burden

We examine the regularization effect that CTC joint training has on the attentional decoder, per L1 in §3, by first quantifying the monotonicity, m of a (L, T) shaped source attention pattern, A:

$$m = \Big(\sum_{2<l\leq L} [\operatorname*{argmax}_{t\in T} A_l \geq \operatorname*{argmax}_{t\in T} A_{l-1}] \Big) / L$$

where $[\cdot]$ denotes the Iverson bracket. In other words, we define monotonicity m as the rate at which the decoder at step l attends most sharply on an input index, $\operatorname{argmax}_{t\in T} A_l$, which is greater than or equal to that of the previous step $l-1$, $\operatorname{argmax}_{t\in T} A_{l-1}$. We compute m over all examples in our validation sets for De-En MT and En-De ST and show the layer-wise averages over all examples and attention heads in Figure 2. It can be seen that *the decoder source attention patterns are more monotonic* when using jointly trained hierarchical encoders. Per line 2 of Table 1, we argue that this greater monotonicity allows the decoder to more rapidly learn soft alignment patterns – ultimately this advantage is reflected in the overall performance gains observed from joint training without joint decoding (line 2 vs. 4 in Table 2).

For a qualitative example illustrating the increased monotonicity of decoder source attention patterns, please see §A.7. We also found that *increased monotonicity leads improves multilingual parameter sharing* in our All-En MT and ST models, suggesting that the target-orientation of our encoder reduced the decoder's burden of soft-aligning target English outputs to source languages with varying word-orders (discussed further in §A.5).

What are the respective contributions of SrcCTC *and* TgtCTC*?* TgtCTC holds elevated importance as joint decoding is not possible without it.

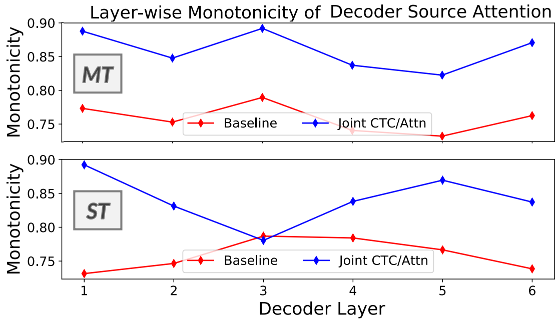


Figure 2: Layer-wise monotonicity (↑) of the source-attention patterns produced by MT/ST decoders.

		MT (De-En)	ST (En-De)
SrcCTC	TgtCTC	IWSLT14	MuST-C-v2
✗	✗	32.1	27.7
✓	✗	34.1	27.8
✗	✓	33.3	28.1
✓	✓	**34.8**	**28.3**

Table 3: Ablation on the impacts of SrcCTC and TgtCTC CTC components of hieraractical encoding, as measured by performance on validation sets. Only attention is used in decoding to enable fair comparisons.

However, we'd like to understand how each component contributes to the benefits observed from joint training without joint decoding in §6.1. In Table 3, we ablate SrcCTC and TgtCTC in order to confirm that both contribute to performance gains. Note that SrcCTC on its own appears to contribute more to MT than it does to ST, suggesting that the length adjustment stage is more critical in MT.

6.3 Even Weak CTC Models Strengthen Joint CTC/Attention Models

We examine the generalization effect that augmenting autoregressive likelihoods with conditionally independent likelihoods has during inference, per L2 in §3, by evaluating De-En MT and En-De ST models on out-of-domain EuroParl test sets (Iranzo-Sánchez et al., 2020). As shown in Table 4, joint CTC/Attention models outperform pure-attention baselines across in-domain (In-D) and out-of-domain (Out-D) settings. When decoding only the CTC branch of joint models (denoted as CTC I-Sync in the table) performance is significantly degraded compared to the attention branch of the same models (denoted as Attn O-Sync in the table). This gap appears slightly lessened in the out-of-domain setting where CTC's conditional independence may offer some robustness. Nonetheless

Model	Decoding Method	MT (De-En) In-D	MT (De-En) Out-D	ST (En-De) In-D	ST (En-De) Out-D
Pure-Attn	Attn Only	32.8	15.8	27.8	20.5
Joint C/A	Attn Only	33.6	17.1	28.3	21.0
	+CTC Rescore	33.6	17.1	28.3	21.0
Joint C/A	Joint O-Sync	**34.1**	**17.6**	**29.2**	**21.7**
Joint C/A	CTC Only	27.3	13.1	24.4	16.5
	+Attn Rescore	29.5	13.9	26.2	17.8
Joint C/A	Joint I-Sync	**33.7**	**17.4**	**29.2**	21.1

Table 4: In/out domain test performances of joint CTC/attention models with various decoding methods.

these weak CTC models still boost their stronger attention counterparts during joint decoding (both via input and output-synchrony), suggesting that *ensembling of conditionally independent and dependent likelihoods is a powerful technique*.

Further, synchronous joint decoding methods outperform their two-pass re-scoring counterparts (discussed in D2 of §7), suggesting that *joint selection of the hypothesis set is necessary* for easing the respective weaknesses of autoregressive and conditionally independent likelihood estimation.

6.4 CTC's Alignment Information Resolves Attention's End-Detection Problem

Finally, we examine the effect that CTC's alignment information has on end detection during decoding, per L3 in §3. In Figure 3, we observe the change in translation quality (as measured by BLEU) and output length (as measured by hypothesis-to-reference length ratio) when the length penalty (denoted as $\mathrm{LengthPen}(\cdot)$ in Algorithms 1 and 2) is gradually increased, forcing decodings to produce longer outputs. Pure-attention baselines rapidly degenerate when forced to produce hypotheses that are longer than references as they struggle to detect the ends of hypotheses (Murray and Chiang, 2018). On the other hand, joint decoding produces gradually longer outputs regardless of whether CTC is in a primary role (input-synchrony) or a secondary role (output-synchrony), demonstrating that *CTC alignments ease the decoder's end-detection problem* by explicitly or implicitly ruling out hypotheses of incorrect lengths.

7 Discussion: More on Joint Decoding

D1 *Why do input vs. output-synchronous joint decodings yield slightly different results?*

By comparing the CTC likelihood estimation in INPUTSTEP vs. OUTPUTSTEP, it can be seen that there is a trade-off between speed vs. accuracy. First, note that in OUTPUTSTEP, $\mathrm{CTCScore}(y_{1:l}, X_{1:T})$, is a marginalization over the likelihoods of all possible alignments of the partial hypothesis $y_{1:l}$ to the full input $X_{1:T}$ (Seki et al., 2019). On the other hand, $\mathrm{CTCScore}(\tilde{y}, X_{1:t})$ in INPUTSTEP is an estimation of the marginalized likelihoods of the partial hypothesis $y_{1:l}$ to the *partial* input $X_{1:t}$ (Graves, 2012; Hannun et al., 2014). Even at step T, these two $\mathrm{CTCScore}(\cdot)$'s are not equivalent. Since CTCCnds may include the blank token, INPUTSTEP may prune partial hypotheses at a previous beam step which would have merged with $y_{1:l}$. Therefore, $\mathrm{CTCScore}(\cdot)$ in *input-synchrony is less accurate*. However, *input-synchrony requires fewer computations*. Using dynamic programming, output-synchrony computes $\mathrm{CTCScore}(\cdot)$ for all partial hypothesis within a single beam step with $\mathcal{O}(bpT)$ log-additions (Watanabe et al., 2017) while input-synchrony uses only $\mathcal{O}(bp)$ log-additions (Hannun et al., 2014).

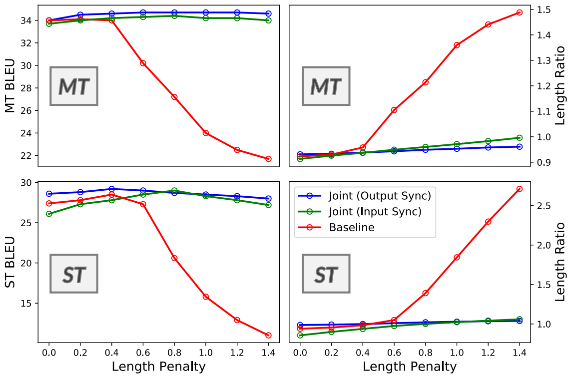


Figure 3: Elasticity of BLEU and length ratios ($|\mathrm{hyp}|/|\mathrm{ref}|$) w.r.t length penalty in validation sets.

Decoding Type		Accuracy		Speed
Method	Beam Size	BLEU	Search Error	RTF
Joint O-Sync	5	29.1	0.73%	0.9
Joint O-Sync	10	29.2	0.44%	1.7
Joint O-Sync	50	29.0	0.36%	9.0
Joint I-Sync	5	28.1	1.02%	0.4
Joint I-Sync	10	28.6	1.09%	0.9
Joint I-Sync	50	29.0	0.87%	6.4

Table 5: Speed vs. accuracy for joint input/output-sync decoding of En-De ST val. set as a fxn. of beam size.

In Table 5, we perform an experimental validation of our theoretical understanding of the speed vs. accuracy trade-off between the two synchronous joint decoding variants. To quantify speed, we compute the real-time factor (RTF) as the ratio of decoding time over the duration of input speech. To quantify accuracy beyond the BLEU metric, we compute the search error rate (Meister et al., 2020) by counting the sequences for which the hypothesis has higher exact likelihood than the reference. *For the same beam size, output is slower but more accurate than input-synchronous.* We conclude that input-synchrony may in fact be preferable in applications with latency constraints.

D2 *Why did synchronous joint decodings outperform re-scoring decodings in Table 4?*

There is a family of two-pass decoding algorithms (Watanabe et al., 2017; Sainath et al., 2019), which also achieve joint decoding by first estimating the likelihoods of a subset of sequences $\mathcal{V}'$ with one module and then re-scoring the estimates with the other module. In these approaches, the subset $\mathcal{V}'$ is determined asynchronously, meaning the joint likelihood is not considered until the re-scoring step; this delayed consideration of the joint likelihood is the main drawback compared to the synchronous approaches. If the attentional decoder is used to determine $\mathcal{V}'$, then $\mathcal{V}'$ would suffer from exposure/label bias and the length problem (§2). On the other hand, if CTC is used to determine $\mathcal{V}'$, the lack of causal modeling in CTC leads to poor estimates of $\mathcal{V}'$ – particularly for translation.

8 Related Works

The idea of using latent alignments to improve autoregressive translation has been explored previously by Haviv et al. (2021) who concluded that CTC alignments are not compatible with teacher forcing. The key difference is that we train CTC and autoregressive models jointly while Haviv et al. (2021) sought to apply CTC to train autoregressive models, replacing cross-entropy entirely. More recently in a concurrent work, Zhang et al. (2022) have also shown the effectiveness of jointly training CTC and attention in the context of ST for unwritten languages where no ASR transcriptions are available. We believe that our contribution showing the effectiveness of also jointly decoding CTC and attention demonstrates an additional technique which can further improve their direction. Our work also differs in that we seek to incorporate the ASR objective into ST via hierarchical encoding.

Other concurrent works integrated CTC and attention within blockwise streaming (Deng et al., 2022) and compositional multi-decoder (Yan et al., 2022) architectures for ST in particular. Our work supports their findings by addressing *why* CTC is helping, and we provide a unified approach that generalizes to both MT and ST. Prior works have also used the non-autoregressive property of CTC as means for speeding up autoregressive models during inference (Inaguma et al., 2021a; Gaido et al., 2021), but these works focus on latency and do not apply CTC to improve translation quality.

DECODING TYPE		SPEED	
Method	Beam Size	RTF	%Δ
Pure-Attn O-Sync	5	0.9	-
Pure-Attn O-Sync	10	1.2	-
Pure-Attn O-Sync	50	3.5	-
Joint CTC/Attn O-Sync	5	0.9	+0%
Joint CTC/Attn O-Sync	10	1.7	+42%
Joint CTC/Attn O-Sync	50	9.0	+157%
Joint CTC/Attn I-Sync	5	0.4	-56%
Joint CTC/Attn I-Sync	10	0.9	-25%
Joint CTC/Attn I-Sync	50	6.4	+85%

Table 6: **Limitations Table:** comparison of joint decoding and pure-attention RTFs across different beam sizes. %Δ between the joint RTF and pure-attention RTF for the same beam size is shown, where positive %'s indicate slow-downs and negative %'s indicate speed-ups.

9 Conclusion

We propose to jointly train and decode CTC/attention models for MT and ST using 1) hierarchical encoding to resolve incompatibilities between CTC and the non-monotonic mappings in translation and 2) synchronous decoding to ease the exposure/label biases of autoregressive decoders with CTC's conditionally independent alignment information. Our analyses reveal several reasons why even weak CTC models benefit autoregressive translation via joint modeling, suggesting that future explorations into jointly modeling attentional decoders with other latent alignment models (Graves, 2012; Ghazvininejad et al., 2020; Saharia et al., 2020) may uncover similar benefits.

Limitations

There are several potential limitations pertaining to the increased computational overhead and latency of the joint modeling approach. One concern is that joint decoding is much slower, but we found that input-synchronous joint decoding is actually faster than pure-attention decoding for smaller beam sizes, as shown in Table 6.

The other limitation is that our MT models upsample input representations in the early layers of the encoder, thereby increasing the computations in subsequent encoder layers and the decoder's cross-attention. We can use LegoNN-based encoders (Dalmia et al., 2022) to adjust the up-sampling rate to a fractional value, minimizing the computations given dataset statistics. Alternatively, we may avoid the need for up-sampling by applying a larger byte-pair encoding size (Kudo and Richardson, 2018) to the target language compared to the source language. CTC's use in guiding efficient down-sampling of representations in ST (Gaido et al., 2021) suggest that it may also be applied for efficient up-sampling for MT – we leave this study on efficiency to future work.

Finally, note that the corpora used for the MT experiments in this work are considered medium resourced. Prior work (Murray and Chiang, 2018) has shown that the autoregressive end-detection problem exists across low to high resourced scenarios; suggesting that the CTC/attention approach would be generally beneficial. We leave the study of joint CTC/attention modeling on higher resourced MT corpora to future work.

Acknowledgements

Brian Yan and Shinji Watanabe are supported by the Human Language Technology Center of Excellence. This work used the Extreme Science and Engineering Discovery Environment (XSEDE) (Towns et al., 2014), which is supported by National Science Foundation grant number ACI-1548562; specifically, the Bridges system (Nystrom et al., 2015), as part of project cis210027p, which is supported by NSF award number ACI-1445606, at the Pittsburgh Supercomputing Center.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A Smith, and David Yarowsky. 1999. Statistical machine translation. In *Final Report, JHU Summer Workshop*, volume 30.

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1796.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Léon Bottou. 1991. *Une Approche théorique de l'Apprentissage Connexionniste: Applications à la Reconnaissance de la Parole*. Ph.D. thesis, Université de Paris XI, Orsay, France.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. In *Conference of european association for machine translation*, pages 261–268.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964.

William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly. 2020. Imputer: Sequence modelling via imputation and dynamic programming. In *International Conference on Machine Learning*, pages 1403–1413. PMLR.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. *Advances in neural information processing systems*, 28.

Shun-Po Chuang, Yung-Sung Chuang, Chih-Chiang Chang, and Hung-yi Lee. 2021. Investigating the reordering capability in CTC-based non-autoregressive end-to-end speech translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1068–1077, Online. Association for Computational Linguistics.

Siddharth Dalmia, Dmytro Okhonko, Mike Lewis, Sergey Edunov, Shinji Watanabe, Florian Metze, Luke Zettlemoyer, and Abdelrahman Mohamed. 2022. LegoNN: Building modular encoder-decoder models. *arXiv preprint arXiv:2206.03318*.

Siddharth Dalmia, Brian Yan, Vikas Raunak, Florian Metze, and Shinji Watanabe. 2021. Searchable hidden intermediates for end-to-end models of decomposable sequence tasks. In *Proceedings of the 2021*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1882–1896.

Keqi Deng, Shinji Watanabe, Jiatong Shi, and Siddhant Arora. 2022. Blockwise Streaming Transformer for Spoken Language Understanding and Simultaneous Speech Translation. In *Proc. Interspeech 2022*, pages 1746–1750.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888. IEEE.

Yichao Du, Zhirui Zhang, Weizhi Wang, Boxing Chen, Jun Xie, and Tong Xu. 2022. Regularizing end-to-end speech translation with triangular decomposition agreement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10590–10598.

Marco Gaido, Mauro Cettolo, Matteo Negri, and Marco Turchi. 2021. CTC-based compression for direct speech translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 690–696.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *International conference on machine learning*, pages 1243–1252. PMLR.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. In *International Conference on Machine Learning*, pages 3515–3523. PMLR.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. 2020. Conformer: Convolution-augmented transformer for speech recognition. *Proc. Interspeech 2020*, pages 5036–5040.

Pengcheng Guo, Florian Boyer, Xuankai Chang, Tomoki Hayashi, Yosuke Higuchi, Hirofumi Inaguma, Naoyuki Kamo, Chenda Li, Daniel Garcia-Romero, Jiatong Shi, et al. 2021. Recent developments on espnet toolkit boosted by conformer. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5874–5878. IEEE.

Awni Hannun. 2019. The Label Bias Problem. `https://awni.github.io/label-bias`.

Awni Y Hannun, Andrew L Maas, Daniel Jurafsky, and Andrew Y Ng. 2014. First-pass large vocabulary continuous speech recognition using bi-directional recurrent DNNs. *arXiv preprint arXiv:1408.2873*.

Adi Haviv, Lior Vassertail, and Omer Levy. 2021. Can latent alignments improve autoregressive machine translation? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2637–2641, Online. Association for Computational Linguistics.

Xiaodong He and Li Deng. 2011. Speech recognition, machine translation, and speech translation—a unified discriminative learning paradigm [lecture notes]. *IEEE Signal Processing Magazine*, 28(5):126–133.

Yosuke Higuchi, Keita Karube, Tetsuji Ogawa, and Tetsunori Kobayashi. 2022. Hierarchical conditional end-to-end ASR with CTC and multi-granular subword units. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7797–7801. IEEE.

Takaaki Hori, Shinji Watanabe, and John Hershey. 2017. Joint CTC/attention decoding for end-to-end speech recognition. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 518–529, Vancouver, Canada. Association for Computational Linguistics.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Chenyang Huang, Hao Zhou, Osmar R Zaïane, Lili Mou, and Lei Li. 2022. Non-autoregressive translation with layer-wise prediction and deep supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10776–10784.

Xuedong Huang, James Baker, and Raj Reddy. 2014. A historical perspective of speech recognition. *Communications of the ACM*, 57(1):94–103.

Hirofumi Inaguma, Siddharth Dalmia, Brian Yan, and Shinji Watanabe. 2021a. Fast-MD: Fast multi-decoder end-to-end speech translation with non-autoregressive hidden intermediates. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 922–929. IEEE.

Hirofumi Inaguma, Yosuke Higuchi, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2021b. Non-autoregressive end-to-end speech translation with parallel autoregressive rescoring. *arXiv preprint arXiv:2109.04411*.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Javier Iranzo-Sánchez, Joan Albert Silvestre-Cerda, Javier Jorge, Nahuel Roselló, Adria Giménez, Albert Sanchis, Jorge Civera, and Alfons Juan. 2020. Europarl-ST: A multilingual corpus for speech translation of parliamentary debates. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8229–8233. IEEE.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4835–4839. IEEE.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Ludwig Kürzinger, Dominik Winkelbauer, Lujun Li, Tobias Watzel, and Gerhard Rigoll. 2020. CTC-segmentation of large corpora for german end-to-end speech recognition. In *International Conference on Speech and Computer*, pages 267–278. Springer.

Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2020. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3520–3533.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8:795–809.

Niko Moritz, Takaaki Hori, and Jonathan Le Roux. 2019. Triggered attention for end-to-end speech recognition. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5666–5670.

Moses-SMT. 2018. `multi-bleu.perl`. `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl`.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium. Association for Computational Linguistics.

Graham Neubig, Zi-Yi Dou, Junjie Hu, Paul Michel, Danish Pruthi, and Xinyi Wang. 2019. compare-mt: A tool for holistic comparison of language generation systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 35–41, Minneapolis, Minnesota. Association for Computational Linguistics.

H. Ney. 1999. Speech translation: coupling of recognition and translation. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, volume 1, pages 517–520 vol.1.

Nicholas A Nystrom, Michael J Levine, Ralph Z Roskies, and J Ray Scott. 2015. Bridges: a uniquely flexible hpc resource for new communities and data analytics. In *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, pages 1–8.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021. Glancing transformer for non-autoregressive neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1993–2003.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016*.

Vikas Raunak, Siddharth Dalmia, Vivek Gupta, and Florian Metze. 2020. On long-tailed phenomena in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3088–3095, Online. Association for Computational Linguistics.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. Association for Computational Linguistics.

Tara N Sainath, Ruoming Pang, David Rybach, Yanzhang He, Rohit Prabhavalkar, Wei Li, Mirkó Visontai, Qiao Liang, Trevor Strohman, Yonghui Wu, et al. 2019. Two-pass end-to-end speech recognition. *Proc. Interspeech 2019*, pages 2773–2777.

Haşim Sak, Andrew Senior, Kanishka Rao, Ozan Irsoy, Alex Graves, Françoise Beaufays, and Johan Schalkwyk. 2015. Learning acoustic frame labeling for speech recognition with recurrent neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4280–4284. IEEE.

Elizabeth Salesky, Matthew Wiesner, Jacob Bremerman, Roldano Cattoni, Matteo Negri, Marco Turchi, Douglas W Oard, and Matt Post. 2021. The multilingual TEDx corpus for speech recognition and translation. *arXiv preprint arXiv:2102.01757*.

Ramon Sanabria and Florian Metze. 2018. Hierarchical multitask learning with CTC. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 485–490. IEEE.

Hiroshi Seki, Takaaki Hori, Shinji Watanabe, Niko Moritz, and Jonathan Le Roux. 2019. Vectorized beam search for CTC-attention-based speech recognition. In *INTERSPEECH*, pages 3825–3829.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231.

J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, and N. Wilkins-Diehr. 2014. Xsede: Accelerating scientific discovery. *Computing in Science & Engineering*, 16(5):62–74.

Emiru Tsunoo, Yosuke Kashiwagi, and Shinji Watanabe. 2021. Streaming transformer asr with blockwise synchronous beam search. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 22–29.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Waibel. 1996. Interactive translation of conversational speech. *Computer*, 29(7):41–48.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech*, pages 2207–2211.

Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. 2017. Hybrid CTC/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253.

Brian Yan, Patrick Fernandes, Siddharth Dalmia, Jiatong Shi, Yifan Peng, Dan Berrebbi, Xinyi Wang, Graham Neubig, and Shinji Watanabe. 2022. CMU's IWSLT 2022 dialect speech translation system. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 298–307, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Biao Zhang, Barry Haddow, and Rico Sennrich. 2022. Revisiting end-to-end speech-to-text translation from scratch. In *International Conference on Machine Learning*.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2019. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.

A Supplementary Information

A.1 Additional Translation Metrics

To supplement our BLEU evaluation in Table 2, we also measure the translation quality of our models using Translation Error Rate (TER) (Snover et al., 2006) and METEOR (Banerjee and Lavie, 2005). As shown in Table 7, our findings are consistent across all three metrics for both MT and ST models.

A.2 MuST-C-v1 Back-Compatibility

See Table 9 for results compared to prior works.

A.3 Valid Set performances

Table 8 presents the validation performances for our ST and MT models.

A.4 Description of Encoder Architectures

LegoNN Encoder (Dalmia et al., 2022) is a stacked multi-block architecture that was introduced to encode and re-sample the input sequence into a sequence of representations of a desired length, which is typically a factor of the input sequence. It first encodes the input using transformer encoder blocks (Vaswani et al., 2017) and then re-encodes them into a sequence of latent representations using cross-attention. Starting from a sequence of learnable positional embeddings (Gehring et al., 2017), these latent representations are learned using another stack of transformer encoder layers with an added cross-attention component over the input representations in each block.

The Conformer encoder (Gulati et al., 2020) is a stacked multi-block architecture and has shown consistent improvement over a wide range of E2E speech processing applications (Guo et al., 2021). It includes a multi-head self-attention module, a convolution module, and a pair of position-wise feed-forward modules in the Macaron-Net style. While the self-attention module learns the long-range global context, the convolution module aims to model the local feature patterns synchronously.

A.5 Increased Cross-Attentional Monotonicity Leads to Increased Multilingual Parameter Sharing

We further examine the source attention parameters in our All-En models to understand the impact that the increased monotonicity of decoder attention (§6.2) has on multilingual parameter sharing. To do so, we extract sparse subnets for each language pair following the Lottery Ticket Sparse Fine-Tuning proposed by Ansell et al. (2022) and compute the pair-wise sharing across the 6 source languages, as measured by the count of overlapping parameters between subnets. In Figure 4, we show the relative change ($\Delta\%$) in multilingual sharing when using hierarchical encoding compared to the baseline. The broad increases suggest that the target-orientation of our encoder *reduced the decoder's burden of soft-aligning* target English outputs to source languages with varying word-orders, allowing for more efficient allocation of capacity.

A.6 `compare_mt.py` Length Analysis

As shown in Figure 5, both joint synchronous decodings are more robust than pure-attention for long output lengths across both MT and ST. Input-synchrony appears particularly more robust in generation of very long outputs for ST.

A.7 View of Regularized Attention

See Figure 6 for a qualitative example of monotonic source attention patterns (supplementary to the quantitative monotonicity in Figure 2).

B Reproducibility

B.1 Dataset Descriptions

See Table 10 for dataset descriptions. Data preparation was done using ESPnet recipes.

B.2 Model Architectures

See Table 11 for model architectures.

B.3 Training/Decoding Hyperparameters

See Tables 12-15 for hyperparameter descriptions.

B.4 Metrics

Sacrebleu signature for all non-Japanese:
`BLEU+case.mixed+numrefs.1`
`+smooth.exp+tok.13a+version.1.5.1`

Sacrebleu signature for Japanese:
`BLEU+case.mixed+lang.en-ja+numrefs.1`
`+smooth.exp+tok.ja-mecab-0.996-IPA`
`+version.1.5.1`

For tokenized BLEU in the IWSLT MT datasets we used `mutibleu.perl` (Moses-SMT, 2018)

B.5 Computing

ST models were trained on 2 x V100 for 2 days. MT models were trained on 1 x A6000 for 1 day.

MODEL NAME	DECODING METHOD	IWSLT14 (DE-EN)			MUST-C-V2 EN-DE		
		BLEU (↑)	TER (↓)	METEOR (↑)	BLEU (↑)	TER (↓)	METEOR (↑)
Pure-Attn (Ours)	Attn-only	32.8	50.9	29.4	27.8	59.1	38.6
Joint CTC/Attn	Attn-only	33.6	50.7	30.0	28.3	58.4	39.2
Joint CTC/Attn	Joint I-Sync	33.7	50.6	30.0	**29.2**	57.8	40.1
Joint CTC/Attn	Joint O-Sync	**34.1**	**49.9**	**30.2**	**29.2**	**57.5**	**40.2**

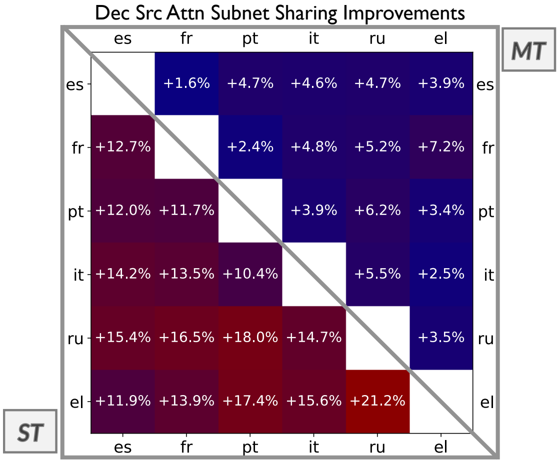
Table 7: Test set performances, as measured by BLEU (↑), TER (↓), and METEOR (↑), of our proposed joint CTC/Attention models compared to pure-attention baselines.

MODEL NAME	DECODING METHOD	IWSLT14 De-En	IWSLT14 Es-En	MUST-C-V2 En-De	MUST-C-V2 En-Ja
Pure-Attn (Ours)	Attn O-sync	34.1	41.2	28.5	11.3
Joint CTC/Attn	Joint I-sync	34.6	42.0	29.0	12.4
Joint CTC/Attn	Joint O-sync	35.0	42.3	29.2	12.4

Table 8: Valid set performances, as measured by BLEU (↑).

MODEL NAME	MUST-C-V1 En-De
ESPnet-ST[1]	22.9
Dual-Decoder[2]	23.6
E2E-ST-TDA[3]	25.4
Multi-Decoder[4]	26.4
Pure-Attn (ours)	27.1
Joint CTC/Attn w/ Joint O-Sync	**28.2**

Table 9: Comparison of our best MuST-C-v1 En-De Joint CTC/Attn model and our Pure-Attn baseline with prior works: [1]Inaguma et al. (2020), [2]Le et al. (2020), [3]Du et al. (2022), [4]Dalmia et al. (2021)



	es	fr	pt	it	ru	el
es		+1.6%	+4.7%	+4.6%	+4.7%	+3.9%
fr	+12.7%		+2.4%	+4.8%	+5.2%	+7.2%
pt	+12.0%	+11.7%		+3.9%	+6.2%	+3.4%
it	+14.2%	+13.5%	+10.4%		+5.5%	+2.5%
ru	+15.4%	+16.5%	+18.0%	+14.7%		+3.5%
el	+11.9%	+13.9%	+17.4%	+15.6%	+21.2%	

Figure 4: Improvement of multilingual sharing in MT/ST decoder source attention parameters when using joint CTC/Attention vs. attention-only training, as measured by pair-wise Δ% in sparse subnet overlap.

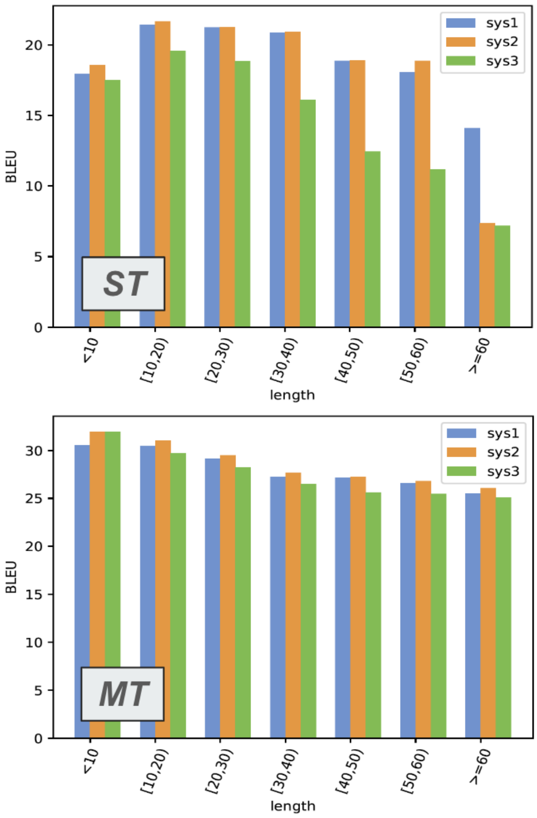


Figure 5: Compare-mt (Neubig et al., 2019) output sentence length to BLEU for joint decoding vs pure-attention models. Model codes: sys1 = Joint Input-Sync, sys2 = Joint Output-Sync, sys3 = Pure-Attn

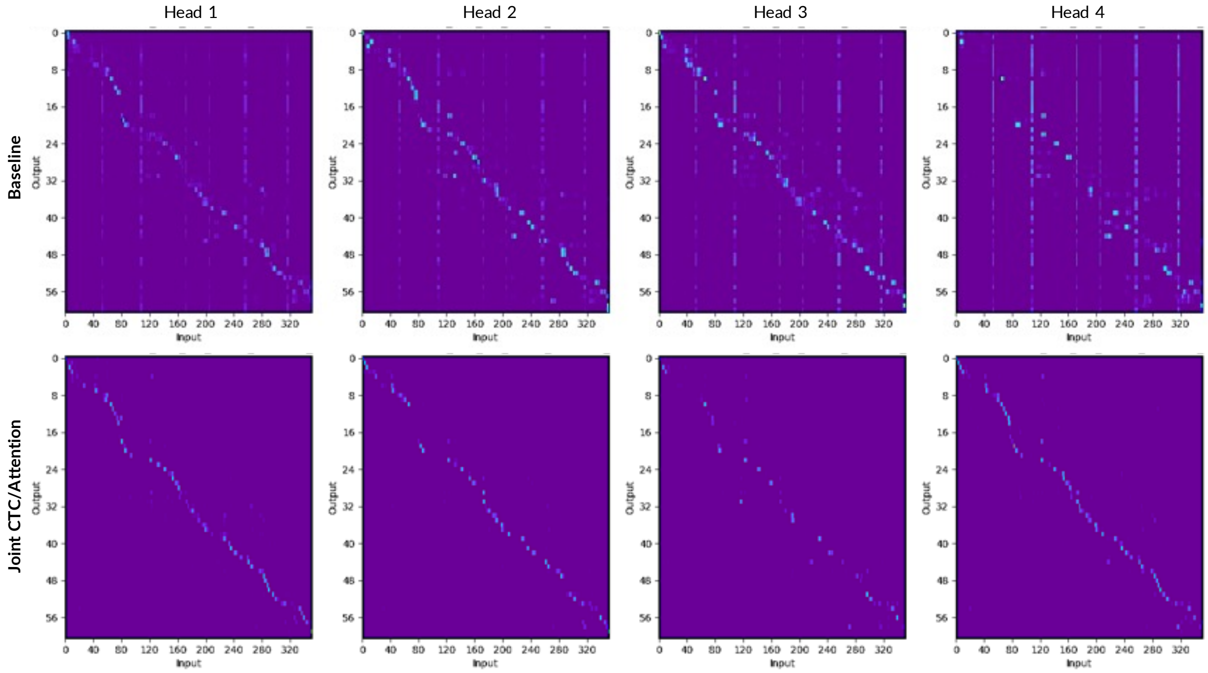


Figure 6: Visualization of source attention patterns produced by pure-attention baseline (top) vs. joint CTC/attention (bottom) ST models. Qualitative example extracted from the final decoder layer. Irregular patterns are observable in the pure-attention plots, but not in the joint CTC/attention plots.

Dataset	Task	Source Lang(s)	Target Lang(s)	Domain	# Train/Valid/Test Utts	# Speech Train Hours
IWSLT17 (Cettolo et al., 2012)	MT	De	En	TED Talk	160k/7k/7k	-
IWSLT17 (Cettolo et al., 2012)	MT	De	Es	TED Talk	160k/7k/7k	-
MuST-C-v2 (Di Gangi et al., 2019)	ASR/ST	En	De	TED Talk	250k/1k/3k	450h
MuST-C-v2 (Di Gangi et al., 2019)	ASR/ST	En	Ja	TED Talk	330k/1k/3k	540h
MTedX (Salesky et al., 2021)	MT	Es, Fr, Pt, It, Ru, El	En	TED Talk	130k/6k/6k	-
MTedX (Salesky et al., 2021)	ASR	Es, Fr, Pt, It, Ru, El	En	TED Talk	400k/6k/6k	730h
MTedX (Salesky et al., 2021)	ST	Es, Fr, Pt, It, Ru, El	En	TED Talk	130k/6k/6k	250h
EuroParl (Iranzo-Sánchez et al., 2020)	MT	De	En	Parliament Speech	- /-/2k	-
EuroParl (Iranzo-Sánchez et al., 2020)	ST	En	De	Parliament Speech	-/-/1k	-

Table 10: MT/ST/ASR dataset descriptions. Utterance counts are rounded to the nearest thousand. Language codes: De=German, En=English, Es=Spanish, Ja=Japanese, Fr=French, Pt=Portuguese, It=Italian, Ru=Russian, El=Greek

Model	Task	# Encoder Layers [$\mathcal{S}$]	# Decoder Layers	SrcCTC Layer	Up/Down-Sample	Pre-Train Init	Src BPE Size	Tgt BPE Size	# Params
Pure-Attn	MT	12 [6,12,18]	6	-	-	-	10k (joint)		54M
Joint CTC/Attn	MT	18 [6,12,18]	6	6	3x	-	10k (joint)		95M
Pure-Attn	ST	18 [12, 18]	6	-	1/4x	Enc lyr 1-12 from ASR	4k	4k	74M
Joint CTC/Attn	ST	18 [12, 18]	6	12	1/4x	Enc lyr 1-12 from ASR	4k	4k	72M
Pure-Attn	ASR	12	6	-	-	-	4k	4k	46M

Table 11: MT/ST/ASR model descriptions. The best MT/ST Encoder layers settings were selected over a search space indicated by $\mathcal{S}$. Parameter counts are rounded to the nearest million. Note that the 12 layer pure-attn model outperformed the 18 layer version and that the 12 layer joint model still outperformed these baselines.

Hyperparameter	Value
Hidden Dropout	0.3
Attention dropout	0.3
Activation dropout	0.3
LR schedule	inv. sqrt. (Vaswani et al., 2017)
Max learning rate	best of [1e-3, 3e-3]
Warmup steps	10000
Number of steps	200 epoch
Adam eps	1e-9
Adam betas	(0.9, 0.98)
Weight decay	1e-4
$\lambda 1$, $\lambda 2$	(1, 2)

Table 12: Training Hyperparameters for MT Models.

Hyperparameter	Value
Hidden Dropout	0.1
Attention dropout	0.1
Activation dropout	0.1
LR schedule	inv. sqrt. (Vaswani et al., 2017)
Max learning rate	0.002
Warmup steps	25000
Number of steps	40 epoch
Adam eps	1e-9
Adam betas	(0.9, 0.98)
Weight decay	0.0001
$\lambda 1$, $\lambda 2$	(2, 5)

Table 13: Training Hyperparameters for ST Models.

Decoding Type	Hyperparameter	Value
Pure Attn	Max Length Ratio	[1, 1.2, 1.4, 1.6, 1.8, 2, 2.5, 3]
	Penalty	[0, 0.2, 0.4, 0.6, 0.8, 1.0]
	Beam Size	5
Joint O-Sync	Max Length Ratio	1
	Penalty	[0, 0.2, 0.4, 0.6, 0.8, 1.0]
	CTC Weight	0.3
	Beam Size	5
Joint I-Sync	Max Length Ratio	1
	Penalty	[0, 0.2, 0.4, 0.6, 0.8, 1.0]
	Blank Penalty	[0.5, 0.75, 1.0]
	CTC Weight	[0.3, 0.5]
	Beam Size	[10, 30]

Table 14: Decoding Search Space MT Models.

Decoding Type	Hyperparameter	Value
Pure Attn	Max Length Ratio	1
	Penalty	[0,0.2,0.4,0.6,0.8,1.0]
	Beam Size	[10, 30, 50]
Joint O-Sync	Max Length Ratio	1
	Penalty	[0,0.2,0.4,0.6,0.8,1.0]
	CTC Weight	[0.3, 0.5]
	Beam Size	[10, 30, 50]
Joint I-Sync	Max Length Ratio	1
	Penalty	[0,0.2,0.4,0.6,0.8,1.0]
	Blank Penalty	1
	CTC Weight	[0.3, 0.5]
	Beam Size	[10, 30, 50]

Table 15: Decoding Search Space ST Models.

Modelling Temporal Document Sequences for Clinical ICD Coding

Clarence Boon Liang Ng[1] **Diogo Santos**[2] **Marek Rei**[1,2]

[1] Imperial College London, United Kingdom
[2] Transformative AI, United Kingdom
`{clarence.ng21,marek.rei}@imperial.ac.uk`
`{santos}@transformative.ai`

Abstract

Past studies on the ICD coding problem focus on predicting clinical codes primarily based on the discharge summary. This covers only a small fraction of the notes generated during each hospital stay and leaves potential for improving performance by analysing all the available clinical notes. We propose a hierarchical transformer architecture that uses text across the entire sequence of clinical notes in each hospital stay for ICD coding, and incorporates embeddings for text metadata such as their position, time, and type of note. While using all clinical notes increases the quantity of data substantially, superconvergence can be used to reduce training costs. We evaluate the model on the MIMIC-III dataset. Our model exceeds the prior state-of-the-art when using only discharge summaries as input, and achieves further performance improvements when all clinical notes are used as input.

1 Introduction

ICD (International Classification of Diseases (World Health Organization, 1978)) coding refers to the task where medical professionals classify clinical diagnoses and medical procedures associated with each patient using standardised taxonomies, which in turn supports billing, service planning and research. The process is manual and laborious in nature (O'Malley et al., 2005), however there is potential to automate it by identifying relevant information from clinical notes, which are already captured in EHR systems. With this in mind, researchers have started to explore whether machine learning models can succeed at this task (Mullenbach et al., 2018).

The current research on the ICD coding task focuses on the extraction of codes from the discharge summary. This document is commonly written at the end of a hospital stay and provides a textual description of the important diagnoses and procedures for a given patient, making it particularly helpful for the task. However, many other clinical notes are also created during the hospital stay, which can provide important details or useful additional context that may be missing from the discharge summary itself. Analysing the full sequence of notes would allow models to make more accurate decisions and make the problem more similar to a real-life setting, where clinicians have to consider all information about a patient for ICD coding, rather than information only in a single document.

In this work we study how the inclusion of clinical notes across the entire hospital stay can affect performance on the ICD coding task. We propose the **H**ierarchical **T**ransformers for **D**ocument **S**equences (**HTDS**) model, which is an adaptation of the hierarchical transformer model (Zhang et al., 2019) for temporal modelling of document sequences. The model takes text and metadata (such as the time and type of note) from a sequence of multiple documents as input and achieves improved performance when additional clinical notes are used for modelling. We compare different prioritisation criteria for selecting which notes to use as input and how to best represent the sequence information. Methods related to superconvergence are applied to speed up the model training process in order to handle the increased size of the data that needs to be processed.

Our experiments show that the inclusion of additional clinical notes indeed improves model accuracy and leads to better predictions. We evaluate our models against the MIMIC-III-50 (Johnson et al., 2016) test set. When considering only the discharge summaries of each hospital stay as input, our model exceeds the current state-of-the-art performance in terms of Micro-F1. When considering all clinical notes as input, further performance improvements across all metrics of interest are observed, exceeding the state-of-the-art performance in Micro-F1, Micro-AUC, Macro-AUC, and Precision@5 scores.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1640–1649
May 2-6, 2023

2 Related Work

Publicly available electronic health record (EHR) datasets, such as the Medical Information Mart for Intensive Care III (MIMIC-III) dataset (Johnson et al., 2016), provide a shared context for researchers to work on ICD coding. Recent work on ICD coding concentrates on the benchmark tasks presented by Mullenbach et al. (2018), which extracts ICD codes from the free-text discharge summary generated at the end of each hospital stay. Mullenbach et al. (2018) also publicly release their data preprocessing codes and train/dev/test data splits, and these were followed by later works for comparability of result.

In recent years, state-of-the-art work on the ICD coding problem commonly used methods based on convolutional neural networks (CNNs) or recurrent neural networks (RNNs) for text encoding. CAML (Mullenbach et al., 2018) uses a single convolutional layer along with "per-label attention" to extract representations for each label from the convolution output. MSAttKG (Xie et al., 2019) improves the performance further by using a densely connected convolutional network with variable n-gram features, and incorporating knowledge graphs to capture relationships between medical codes. EffectiveCAN (Liu et al., 2021) uses a deep convolutional approach, with a "squeeze-and-excitation" module that repeatedly compresses and then decompresses the convolutional features. LAAT (Vu et al., 2021) uses a bidirectional LSTM to encode the texts, with a per-label attention step on the output to get the final classification. MSMN (Yuan et al., 2022) uses the same architecture as LAAT, with an additional step of extending code descriptions from the Unified Medical Language System (UMLS) with synonyms, and using an attention layer with a separate head for each code synonym.

Researchers using transformer-based models for text encoding experienced difficulties in matching state-of-the-art performance. Ji et al. (2021) apply a range of different transformer-based models but found that none of them outperformed their reimplementation of a simple CNN-based model. Pascual et al. (2021) similarly found it difficult to achieve competitive performance and concluded that better methods of handling long input sequences are required to improve the models further. Gao et al. (2021) also find that a simple self-attention network with far less parameters outperformed BERT-based models on many tasks. Dai et al. (2022) show that incorporating task-adaptive pre-training, overlapping chunks, and using a large pretrained language model make it possible to achieve performance that is close to, but still slightly below the state-of-the-art. In general, language models that were pretrained on texts in the biomedicine domain, such as ClinicalBERT (Alsentzer et al., 2019), BioBERT (Lee et al., 2020), BlueBERT (Peng et al., 2019), and PubMedBERT (Gu et al., 2021) tend to achieve higher performance (Dai et al., 2022; Ji et al., 2021) as compared to language models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) which are trained on general domain corpora, as the models have been adapted to the specialised language used in clinical notes. Among the range of pretrained language models available for the biomedicine domain, better performance was achieved when a specialised token vocabulary is used (Gu et al., 2021; Lewis et al., 2020) and when the pre-training corpora is closer in nature to those used for the downstream task (Gururangan et al., 2020). Very recently, Huang et al. (2022) identified the restricted capacity of the [CLS] token as a potential limiting factor, and showed how using all tokens in the label attention step leads to state-of-the-art performance on the MIMIC-III-Full problem. However, they do not report results on the MIMIC-III-50 problem.

While transformer-based language models have been very successful on short sequences of text (BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) use a maximum sequence length of 512 tokens), challenges arise when attempting to apply it to longer text sequences due to the quadratic computational complexity of the self-attention mechanism. Experiments conducted by Gao et al. (2021) show that transformer models require 3x more processing time compared to CNNs, making it more tedious to explore different hyperparameters and modelling strategies. Various modifications have been proposed to the transformer architecture to reduce computation costs, in models such as TransformerXL (Dai et al., 2019), LongFormer (Beltagy et al., 2020), and BigBird (Zaheer et al., 2020), however domain-pretrained models for these architectures are relatively scarce. Most transformer-based models for the ICD coding problem adapt the hierarchical transformer (Zhang et al., 2019), which splits the text into chunks that are encoded separately with the pre-trained language model, and then feeds the output of the [CLS] token into a second transformer to allow interaction of information across chunks.

To the best of our knowledge, there has been no prior work that attempts to extend the ICD coding task with other clinical documents.

3 Approach

Our **H**ierarchical **T**ransformers for **D**ocument **S**equences (**HTDS**) model is based on the hierarchical transformer architecture (Zhang et al., 2019), with additional adaptations specifically to handle document sequences. Figure 1 provides an illustrated diagram of the full HTDS model architecture. We process documents using the following steps:

Step 1 - Preprocess and Chunk: The text in each document is sequentially tokenized and split into chunks, each containing up to T_c tokens. Every new document or clinical note always starts a new chunk.

From these tokenized chunks we select up to N_c chunks for processing. If more than N_c chunks are available, various prioritisation strategies can be considered to select which chunks to use as model input. In our main model we use a strategy that prioritized diversity in the categories of notes used. To do this, we select the last note by timestamp of each category, and then the second last note of each category, and so on until N_c chunks of text are selected.

Step 2 - Encode with Language Model: The chunks are encoded using the pre-trained language model, producing an output of dimension N_c x T_c x H_e, where H_e is the dimension of the hidden state in the pre-trained LM.

Step 3 - Add Chunk Meta-Information: Meta-information of each chunk is added. These are learnable embeddings, retrieved via index lookup, with size H_e. Positional Embeddings (PE) capture the positional index of the chunk, and are numbered from 0 for the first chunk until N-1 for the last chunk. Temporal Sequence Embeddings (TE) capture the temporal order in which the documents were captured, and are indexed in running order from 0 for chunks belonging to the first document and incremented with each subsequent document. We noted that this indexing approach would often assign varying indices to the last chunk or document, as the number of chunks and documents for each case would vary. This might limit the ability of the model to identify the last chunk or document of the text. Hence, we also include Reversed Positional Embeddings (Rev-PE) and Reversed Temporal Sequence Embeddings (Rev-TE), which start from 0 for the last chunk (or document) and are then incremented with each preceding chunk (or document). Category Embeddings (CE) capture the category of the note, with a unique index for each CATEGORY code. All learnable embeddings use values initialised from a $N(0, 0.1)$ distribution. We hypothesise that these embeddings can help the model to factor in chunk meta-information which may be relevant for classification.

Step 4 - Second Transformer: The embeddings are added together (token embeddings + meta-information embeddings), then concatenated across all the chunks and given as input to a second transformer with N_e encoder layers. This allows for information from each chunk to interact with all the other chunks and the use of only a small number of layers in this second transformer will keep the computational requirements feasible. The output is an updated embedding of each token, with dimensions (N_c x T_c) x H_e.

Step 5 - Label Attention: A label attention layer is applied. We train learnable embeddings α_l for each label ($\alpha = [\alpha_1 ... \alpha_{N_l}]$ has dimensions N_l x H_e, where N_l is the number of labels) which are applied against the chunk embeddings ($H = [h_1 ... h_{N_c}]$) in an attention step as follows:

$$A = softmax(H\alpha^T)$$

$$V = H^T A$$

$$Dim(A) = (N_c \times T_c) \times N_l$$

$$Dim(V) = H_e \times N_l$$

The i-th column in V would be an independent representation, of dimension H_e, for the i-th label for classification.

Step 6 - Generate Final Classification: A classification layer is applied. We take $\sigma(W_l v_l)$ to get the probability of label l, where W_l is a learnable weight matrix of dimension H_e for label l, v_l is the l-th item of matrix V, and σ is the sigmoid activation function. To obtain the final classification we apply a threshold t for positive classification that is optimised for micro-F1 on the validation set.

4 Experiment Setup

Dataset: For our experiments, we use the MIMIC-III (Medical Information Mart for Intensive Care)

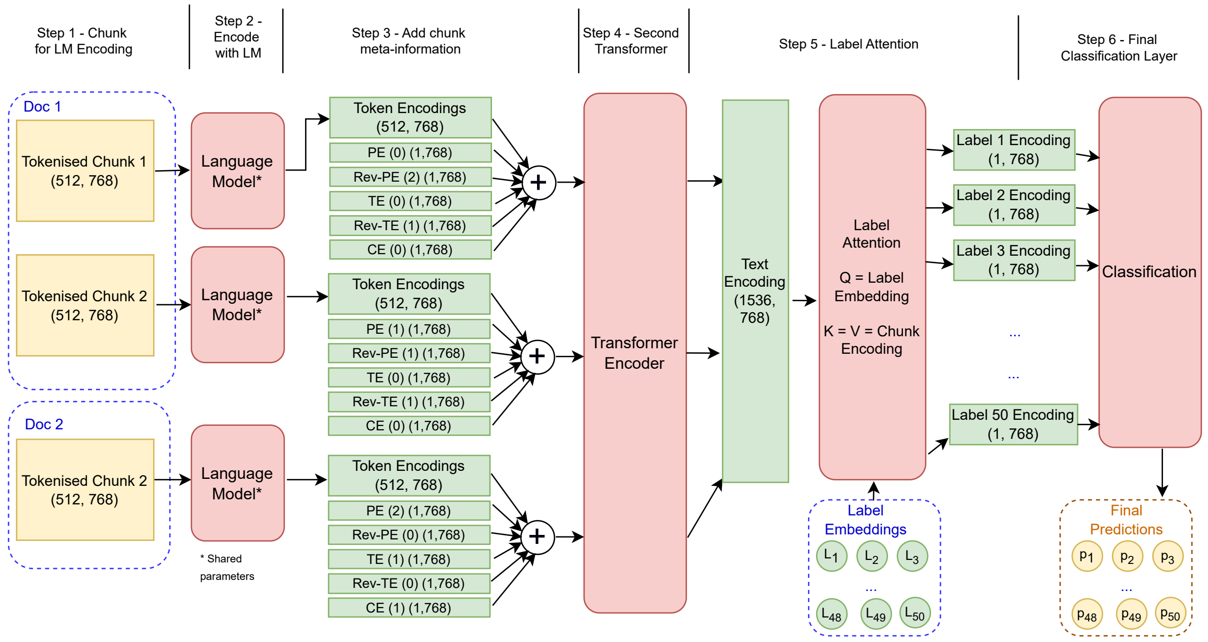


Figure 1: HTDS Model Architecture. The document sequence is first split into chunks (Step 1) and encoded with a pre-trained language model (Step 2). Meta-information of each chunk is then added to the token encodings (Step 3) before a second transformer is applied to allow attention of information across chunks (Step 4). Finally a label attention layer is applied (Step 5) and the outputs are used for classification (Step 6).

	Mean	**SD**
Discharge Summaries		
Total Documents	1.1	0.4
Total Words	1896	929
Total Tokens	3594	1760
All Notes		
Total Documents	33	59
Total Words	10442	21334
Total Tokens	21916	46461

Table 1: Summary statistics: Amount of text contained in clinical documents per hospital stay, measured in terms of total number of documents, words, tokens (using the RoBERTa-PM-M3-Voc tokenizer).

dataset (Johnson et al., 2016), which contains multimodal information on patients admitted to critical care between 2001 and 2012 at the Beth Israel Deaconess Medical Center in Boston, Massachusetts. To limit computational costs, we focus on the MIMIC-III-50 problem, which limits the problem to the top 50 ICD codes by frequency.

To construct the task dataset, we follow Mullenbach et al. (2018) preprocessing steps, with a few exceptions: (1) we keep the text and metadata (specifically the datetime and the category of note) of all notes rather than just the discharge summaries, (2) we do not remove punctuation as we found that performance drops when punctuation is excluded. Each record represents one hospital stay (uniquely identified by the HADM_ID value) and contains the texts and ICD codes linked to that hospital stay. There are 8066, 1573 and 1729 records in the train, dev and test sets respectively, giving us a total of 11368 records.

During the data cleaning process, we noticed that the train set contains clinical notes tagged under the category "Nursing/Other", but no clinical notes were tagged in this category in the dev and test sets. For our experiments we grouped "Nursing/Other" and "Nursing" into a single category.

Table 1 shows summary statistics of the dataset. In general, discharge summaries are far longer than other documents, with an average of 1724 words per document as compared to the overall average of 316 words per document. However, the text in discharge summaries only accounts for less than 20% of the words generated in each hospital stay, suggesting the possibility that the other notes might carry additional information that can improve ICD coding accuracy. We also provide the number of tokens produced when the text is tokenized with the RoBERTa-PM-M3-Voc (Lewis et al., 2020) tokenizer, and we see from the numbers that most hospital stays involve text data that is beyond the 512-token maximum of a single transformer language model.

We also note that the amount of text in each hospital stay can vary widely and has a right-skewed distribution. There is a notable proportion of longer hospital stays which generate substantially more documents and text as compared to the rest. The 90th percentile for Total Words and Total Document Count across all notes is 20556 and 72 respectively. For these hospital stays, the effects of the note prioritisation strategy on model performance would be more prominent.

Task Definition: We investigate two variations of the ICD classification task on this dataset. For Task 1, the notes that are available for modelling are restricted to discharge summaries only. Some hospital stays (11% of stays) have multiple discharge summaries, typically because of addenda, and in these cases we keep all of them. This would be equivalent to the MIMIC-III-50 task attempted by past works. For Task 2, all notes in each hospital stay are available for use in modelling. This vastly increases the number of documents (from an average of 1.1 to 33 per hospital stay) and the number of words (from an average of 1896 to 10442) to be considered. Task 2 uses the same data splits and labels as Task 1, allowing us to compare the results to assess whether information from the additional notes is able to improve performance.

For both tasks, we use the same evaluation metrics as defined by Mullenbach et al (Mullenbach et al., 2018) and then subsequently followed by other researchers: micro-F1, macro-F1, micro-AUC, macro-AUC, and Precision at k=5.

Implementation and Model Hyperparameters: Pytorch was used for the implementation of our models, and NVIDIA Tesla A100 80GB GPUs were used for finetuning. Hyperparameters were tuned manually; Table 2 details the search space and final hyperparameter values used for the HTDS model. The pretrained language model was initialised to the RoBERTa-base-PM-M3-Voc (Lewis et al., 2020) model checkpoint, which was pretrained on texts in PubMed, PubMed Central, and MIMIC-III physician notes. The second transformer uses 1 encoder layer with 8 attention heads.

Texts are tokenized into chunks of T_c=512 tokens and a maximum of N_c=32 chunks were used as model input. With these values for T_c and N_c, the note selection strategy to maximise diversity of document categories (detailed earlier in Section 3) was applied for 45% of samples which have more than 32 chunks of text. The model has 136M parameters in total.

Hyperparameter	Values
Optimization	
Peak Learning Rate	1e-6 to 1e-4 **(5e-5)**
Number of Epochs	10-50 **(20)**
Early Stopping Patience Threshold	None, **3**, 5, 10
Effective Batch Size	1-64 **(16)**
Language Model	
Pre-trained LM	PubMedBERT, **RoBERTa-base-PM-M3-Voc**, RoBERTa-large-PM-M3-Voc
Tokens per chunk, T_c	**512**
Max Chunks, N_c	1-48 **(32)**
Second Transformer	
Encoder Layers	0, **1**, 2
Attention Heads	**8**, 12

Table 2: Hyperparameter search space. Bolded text indicates hyperparameters used in the HTDS model.

These hyperparameters were selected to maximise Micro-F1 on the dev set, with a few exceptions to manage training and computation costs: (1) while using the larger RoBERTa-large-PM-M3-Voc model was found to achieve better performance, we kept to the smaller RoBERTa-base-PM-M3-Voc model; (2) while increasing the maximum number of chunks N_c in general leads to better performance, we limit our model to a maximum of 32 chunks.

Training models that take text across all clinical documents as inputs, compared to using only the discharge summary, requires substantially more computational resources. With A100 GPUs, 15.5 samples are processed per second when training on discharge summaries only[1], and 4.9 samples are processed per second when training with all clinical documents. To speed up the model optimisation process, we apply the 3-phase *1cycle*

---

[1] TrLDC (Dai et al., 2022), which we consider to be a comparable model in terms of architecture, processed 7.4 samples per second when training on discharge summaries on NVIDIA V100 GPUs.

	Micro F_1	Macro F_1	Micro AUC	Macro AUC	P@5
CNN-based Models					
CAML (Mullenbach et al., 2018)	63.3	57.6	91.6	88.4	61.8
MSAttKG (Xie et al., 2019)	68.4	63.8	93.6	91.4	64.4
EffectiveCAN (Liu et al., 2021)	71.7	66.8	94.5	92.0	66.4
RNN-based Models					
LAAT (Mullenbach et al., 2018)	71.5	66.6	94.6	92.5	67.5
MSMN (Yuan et al., 2022)	72.5	**68.3**	94.7	92.8	68.0
Transformer-based Models					
Hier-PubMedBERT (Ji et al., 2021)	68.1	63.3	90.8	88.6	64.4
TrLDC (Base) (Dai et al., 2022)	70.1	63.8	93.7	91.4	65.9
TrLDC (Large) (Dai et al., 2022)	71.1	65.5	94.1	91.9	66.4
Our Models					
HTDS (Discharge Summaries)	$72.6_{0.3}$	$66.6_{1.2}$	$94.5_{0.1}$	$92.6_{0.3}$	$67.4_{0.3}$
HTDS (All Notes)	$\mathbf{73.3}_{0.3}$	$67.9_{0.4}$	$\mathbf{95.0}_{0.2}$	$\mathbf{93.2}_{0.2}$	$\mathbf{68.2}_{0.2}$

Table 3: Performance of models on the MIMIC-III-50 test set. Models are sorted by Micro-F1 within each category. Metrics are averaged across 5 replications. Subscripts indicate the standard deviation across runs. Bolded values indicate the best score achieved for each metric.

learning rate scheduler for superconvergence as described in (Smith and Topin, 2019). The learning rate (LR) progresses via cosine annealing from 1/25 of the peak LR to the peak LR (5e-5) in the first phase (30% of iterations) and then goes back to 1/25 of the peak LR in the second phase (30% of iterations). Finally in the third phase (40% of iterations), LR is annealed to 1/1000 of the peak LR. The AdamW optimizer is used, with an effective batch size of 16 achieved through gradient accumulation. The model is trained for up to 20 epochs with an early stopping patience threshold of 5. With this setup, training is stopped at around the 14th epoch on average. We note that this is at least 50% less (in terms of number of epochs) compared to past works on the MIMIC-III-50 problem where transformer-based models would be trained for 30 epochs or more (Dai et al., 2022; Ji et al., 2021; Pascual et al., 2021).

5 Results

5.1 Main Results

Table 3 shows the results when our models are evaluated against the MIMIC-III-50 test set, as well as comparisons against published works. We report the averaged metrics across 5 training replications.

As we can see from the table, prior works with transformer-based models faced challenges in achieving competitive performance on this problem. Dai et al. (2022) managed substantial improvements with the TrLDC model over the work of Ji et al. (2021), however even with a large-sized model their performance still fell slightly behind the best-performing CNN-based and RNN-based models. When using only discharge summaries, HTDS achieves state-of-the-art performance in terms of Micro-F1, the primary metric used for comparison. It also exceeds past CNN-based and Transformer-based models on all metrics of interest.

When including all clinical documents, as compared to including only discharge summaries, the performance of HTDS improves on all metrics of interest (all differences are statistically significant at p<0.05), including an additional 0.7% increase in Micro-F1. Comparing against all other models, we see that the model achieves state-of-the-art performance in terms of all metrics except for Macro-F1. We hypothesize that the modelling of code synonyms in MSMN (Yuan et al., 2022) helped to increase its performance on rarer ICD codes and hence achieve a higher Macro-F1 score, but also note that steps used to improve performance by incorporating synonyms based on UMLS concepts could also be adapted into our model to achieve similar improvements.

Put together, our results demonstrate the value of including clinical documents beyond the discharge summary in modelling.

5.2 Ablation Experiments

To analyse the effect of various components and hyperparameter choices on model performance, we start with our main model and then ablate or vary individual components one at a time, keeping all other components constant, and evaluate their performance on the development set. We share our results in this section.

For all ablation experiments, we report the impact on Micro-F1, the primary metric of interest, averaged across 5 replications.

Quantity of Text Input: Table 4 shows how performance varies as the quantity of text is varied. The quantity of text used as input has a substantial impact on the compute requirements of the entire model architecture. When N_c is reduced 16, 7.5 samples are processed per second when training on A100 GPUs, an increase of 0.5x as compared to 4.9 samples per second for HTDS which uses N_c=32. However, as we can see from the results of this ablation experiment, reducing the quantity of text input leads to a substantial drop in model performance.

	Micro F_1
HTDS (Max 32 Chunks)	74.0
Max 16 Chunks	73.0

Table 4: Performance when the quantity of text input is varied on the development set.

Metadata embeddings: Table 5 shows how the performance varies as the metadata embeddings used in the model are varied. The ablation of each of the embedding types in isolation results in small but consistent decreases in model performance. It is possible that the model compensates by learning to capture some of this information from the text itself without explicit embeddings. Indeed, past works have observed that the clinical notes in the MIMIC-III dataset have a high degree of structure and templating (Liu, 2018). Nevertheless, in our experiments the overall best results were obtained by using the combination of all the proposed metadata embeddings.

	Micro F_1
HTDS (All meta embeddings)	74.0
Ablate CE	73.9
Ablate PE+Rev-PE	73.9
Ablate TE+Rev-TE	73.8

Table 5: Performance when metadata embeddings are varied on the development set.

Chunk Representations: In a traditional hierarchical transformer, only the encoding of the [CLS] token is kept and used as an aggregate representation of the chunk. However, recent works have suggested that the [CLS] token might have insufficient capacity to capture information for the large number of labels in the ICD coding problem (Huang et al., 2022). In Table 6, we show the results when only the [CLS] token is used as an aggregate representation of each chunk, and see that there is a sizeable decrease in performance.

	Micro F_1
HTDS (All token representations)	74.0
CLS token representation only	71.7

Table 6: Performance when the embeddings used for chunk representation are varied on the development set.

Second Transformer: The second transformer in Step 4 allows tokens from each chunk to attend to tokens from other chunks. While earlier studies (Dai et al., 2022; Ji et al., 2021) include this second transformer, it also adds to the computational costs of the model due to the quadratic complexity of the attention step and (Huang et al., 2022) show that the second transformer can be dropped if the encodings of all tokens (rather than just the [CLS] token) are kept for the label attention step.

Our ablation experiments in Table 7 provide some additional insight on this. When considering only the discharge summary, the second transformer can be dropped without substantial impact on performance. However, when modelling the sequence of all clinical documents, ablating the second transformer leads to a noticeable decrease in performance, suggesting that the information in other documents can help further refine token representations before classification.

	Micro F_1
HTDS (Discharge Summaries)	73.2
Ablate 2nd Transformer	73.3
HTDS (All Notes)	74.0
Ablate 2nd Transformer	73.6

Table 7: Performance when second transformer is ablated on the development set.

Note Selection: In around 45% of admissions, tokenizing the text in all available clinical notes will produce more than 32 chunks of 512 tokens. In those cases, we would need to select which chunks are used as inputs to our model. Table 8 shows our results. We considered the following strategies to prioritise which chunks to use:

- By timestamp: We select the first or last 32 chunks by the timestamp of the clinical notes. Taking the last chunks achieved far superior performance.
- By category: We select first the discharge summary[2], then notes of a certain category (Radiology/Nursing/Physician), and then other notes until 32 chunks of text are selected. Our results indicate that the differences in performance are mostly marginal, suggesting that there could be multiple possible strategies that achieve close to optimal performance.
- Prioritise diversity: We select first the last note by timestamp of each category, and then the second last note of each category, and so on until 32 chunks of text are selected. This maximises the diversity (in terms of categories of notes) used as inputs to the model. This approach was found to have the highest score on the development set, and hence used for HTDS.

	Micro F_1
HTDS (Prioritise diversity)	74.0
Prioritise First	68.4
Prioritise Last	73.8
Prioritise Radiology	73.8
Prioritise Nursing	73.7
Prioritise Physician	73.8

Table 8: Performance when note selection is varied on the development set.

In general, we also note that the effects of note selection strategies would be more pronounced when the maximum number of chunks N_c for model input is smaller, as it would result in a greater proportion of text being excluded.

6 Conclusion

As we work towards automated ICD coding, it would be helpful to build models that can consider information that is captured across the patient's EHR record, rather than just the discharge summary (which may not always be exhaustive). Such an approach would also be more similar to a real-life setting, where clinicians consider all available information for ICD coding, rather than information in a single document.

In this paper, we demonstrated the HTDS model, which is an adaptation of the hierarchical transformer model that considers the text and metadata from the entire clinical document sequence for ICD coding. While transformer-based models have faced difficulties achieving competitive performance on the ICD coding problem in the past, with HTDS we show that these challenges can be overcome. When evaluated on the MIMIC-III-50 test set using only discharge summaries, HTDS exceeded the prior state-of-the-art performance in terms of Micro-F1 score. When all clinical documents were considered, the performance of HTDS improved further on all performance metrics of interest, and exceeded prior state-of-the-art performance in all metrics except for Macro-F1. The results demonstrate the value of including clinical documents beyond the discharge summary in the ICD coding problem.

Possibilities for improving performance even further are plenty. These include: using a large-sized language model or using overlapping text chunks to reduce fragmentation in the initial encoding step (Dai et al., 2022), considering other transformer architectures for long texts (Beltagy et al., 2020; Dai et al., 2019; Zaheer et al., 2020), smarter strategies for chunking the input text to reduce fragmentation, further improving the strategy for selecting which text to use as model input (possibly going down to text-level rather than document-level approaches), and incorporating methods to better model rare ICD codes (Vu et al., 2021; Yuan et al., 2022). Approaches for improving the computational efficiency and training time of the model can be considered to help to reduce GPU resource requirements, and enable the testing of more models and hyperparameter settings. Going even further from here, we could consider multi-modal models that use information across the entire EHR database for ICD coding.

We hope that our findings will encourage future

---

[2] Our exploratory tests find that the discharge summaries contain the most relevant information. We note also that prior work achieved good performance with just the discharge summaries, without the need for other notes.

studies that tap on the full breadth and depth of information available in modern EHR databases today in order to further push the limits of performance on the ICD coding problem in future.

Limitations

Although applying HTDS on the full clinical document sequence in each hospital stay helped to push performance on the ICD coding problem further as compared to the prior state-of-the-art, we note a few limitations to our work.

Firstly, the computational requirements to train HTDS is not trivial. When using NVIDIA A100 GPUs, one training run took 8 GPU-hours on average (for 5 replications this would require 40 GPU-hours). The increased computation cost for HTDS, as compared to other models on the ICD coding problem, could be attributed to the higher number of model parameters in transformers as compared to CNN/RNNs and the increase in input data size as a result of using all clinical documents as input. It is hoped that this issue of high compute costs can be mitigated in future by either further refinements in modelling to improve efficiency or improvements in the compute capabilities of hardware used for model training.

Secondly, we note that our work focuses only on the MIMIC-III-50 problem, where only the top 50 ICD codes by frequency are considered. This would be insufficient in a real-life setting, which would require clinicians to consider all ICD codes. Extending our work on the MIMIC-III-Full problem, which uses a dataset that is 4x in size, was not attempted due to limitations on compute resources. However, we speculate that the benefits of using all clinical documents to perform ICD coding would apply similarly to the MIMIC-III-Full problem.

Finally, while we have taken the actual ICD codes assigned by clinicians as the "ground truth" for the purpose of model evaluation, there have been errors made during the process. We would not expect clinicians to thoroughly read the entire clinical document sequence (consisting an average of over 10,000 words) for every patient when performing ICD coding, and hence there is a possibility that some codes could have been missed. A more thorough approach for model evaluation could involve extracting a sample of records where different codes were assigned by the clinicians and our models for further evaluation by experts, in order to determine the extent to which this might have affected our evaluation metrics.

Ethics Statement

No conflicts of interest are declared by the authors. Clinical data in the MIMIC-III database is de-identified through removal of identifying data elements and date-shifting in accordance with Health Insurance Portability and Accountability Act (HIPAA) standards, and protected health information was further removed from clinical notes using dictionary look-ups and pattern-matching (Johnson et al., 2016). The use of the data is in accordance with the MIMIC-III data use agreement.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical bert embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Xiang Dai, Ilias Chalkidis, Sune Darkner, and Desmond Elliott. 2022. Revisiting transformer-based models for long document classification. *arXiv preprint arXiv:2204.06683*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shang Gao, Mohammed Alawad, M Todd Young, John Gounley, Noah Schaefferkoetter, Hong Jun Yoon, Xiao-Cheng Wu, Eric B Durbin, Jennifer Doherty, Antoinette Stroup, et al. 2021. Limitations of transformers on clinical text classification. *IEEE journal of biomedical and health informatics*, 25(9):3596–3607.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Chao-Wei Huang, Shang-Chi Tsai, and Yun-Nung Chen. 2022. Plm-icd: Automatic icd coding with pretrained language models. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, pages 10–20.

Shaoxiong Ji, Matti Hölttä, and Pekka Marttinen. 2021. Does the magic of bert apply to medical code assignment? a quantitative study. *Computers in Biology and Medicine*, 139:104998.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

J Lee, W Yoon, S Kim, D Kim, CH So, and J Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics (Oxford, England)*, 36(4):1234–1240.

Patrick Lewis, Myle Ott, Jingfei Du, and Veselin Stoyanov. 2020. Pretrained language models for biomedical and clinical tasks: understanding and extending the state-of-the-art. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 146–157.

Peter J Liu. 2018. Learning to write notes in electronic health records. *arXiv preprint arXiv:1808.02622*.

Yang Liu, Hua Cheng, Russell Klopfer, Matthew R Gormley, and Thomas Schaaf. 2021. Effective convolutional attention network for multi-label clinical document classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5941–5953.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111.

Kimberly J O'Malley, Karon F Cook, Matt D Price, Kimberly Raiford Wildes, John F Hurdle, and Carol M Ashton. 2005. Measuring diagnoses: Icd code accuracy. *Health Services Research*, 40(5 Pt 2):1620.

Damian Pascual, Sandro Luck, and Roger Wattenhofer. 2021. Towards bert-based automatic icd coding: Limitations and opportunities. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 54–63.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. *BioNLP 2019*, page 58.

Leslie N Smith and Nicholay Topin. 2019. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2021. A label attention model for icd coding from clinical text. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3335–3341.

World Health Organization. 1978. *International classification of diseases:[9th] ninth revision, basic tabulation list with alphabetic index*. World Health Organization.

Xiancheng Xie, Yun Xiong, Philip S Yu, and Yangyong Zhu. 2019. Ehr coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 649–658.

Zheng Yuan, Chuanqi Tan, and Songfang Huang. 2022. Code synonyms do matter: Multiple synonyms matching network for automatic icd coding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 808–814.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

LONGEVAL: Guidelines for Human Evaluation of Faithfulness in Long-form Summarization

Kalpesh Krishna♠ * **Erin Bransom**♢ **Bailey Kuehl**♢
Mohit Iyyer♠ **Pradeep Dasigi**♢ **Arman Cohan**♢♡ **Kyle Lo**♢

♠University of Massachusetts Amherst, ♢Allen Institute for AI, ♡Yale University
{kalpesh,miyyer}@cs.umass.edu
{erinbransom,baileyk,pradeepd,armanc,kylel}@allenai.org

Abstract

While human evaluation remains best practice for accurately judging the faithfulness of automatically-generated summaries, few solutions exist to address the increased difficulty and workload when evaluating *long-form* summaries. Through a survey of 162 papers on long-form summarization, we first shed light on current human evaluation practices surrounding long-form summaries. We find that 73% of these papers do not perform any human evaluation on model-generated summaries, while other works face new difficulties that manifest when dealing with long documents (e.g., low inter-annotator agreement). Motivated by our survey, we present LONGEVAL, a set of guidelines for human evaluation of faithfulness in long-form summaries that addresses the following challenges: (1) How can we achieve high inter-annotator agreement on faithfulness scores? (2) How can we minimize annotator workload while maintaining accurate faithfulness scores? and (3) Do humans benefit from automated alignment between summary and source snippets? We deploy LONGEVAL in annotation studies on two long-form summarization datasets in different domains (SQuALITY and PubMed), and we find that switching to a finer granularity of judgment (e.g., clause-level) reduces inter-annotator variance in faithfulness scores (e.g., std-dev from 18.5 to 6.8). We also show that scores from a *partial* annotation of fine-grained units highly correlates with scores from a full annotation workload (0.89 Kendall's τ using 50% judgments). We release our human judgments, annotation templates, and our software for future research.[1]

1 Introduction

Human judgments are considered the gold standard for evaluating model-generated summaries (Kryscinski et al., 2019; Fabbri et al., 2021) and generated text more broadly (Celikyilmaz et al., 2020). Unfortunately, human evaluation tends to be labor-intensive, expensive to scale, and difficult to design. This is problematic as a large number of judged examples is needed to draw statistically significant conclusions about system performances (Wei and Jia, 2021) or correlations between human judgments and automatic metrics (Deutsch et al., 2021). Human evaluation is especially challenging when *long* sequences of generated text need to be evaluated, due to the inherent subjectivity in the task (Karpinska et al., 2021; Clark et al., 2021; Krishna et al., 2021; Goyal et al., 2022).

To better understand the challenges of human evaluation on long-form summaries (150 words or longer), we first conduct a comprehensive survey of 162 publications and preprints on long-form summarization (Section 2). We find that 119 papers (73%) do not perform human evaluation on long-form summaries, while the remaining papers deviate significantly from suggested best practices for reproducibility (Gehrmann et al., 2022). Current human evaluation setups lack standardization in their design decisions (such as annotation granularity), some of which can significantly impact inter-annotator agreement (Section 3.1). Finally, 20 papers explicitly mention human evaluation is expensive, difficult, and time-consuming due to the long length of summaries and source documents.

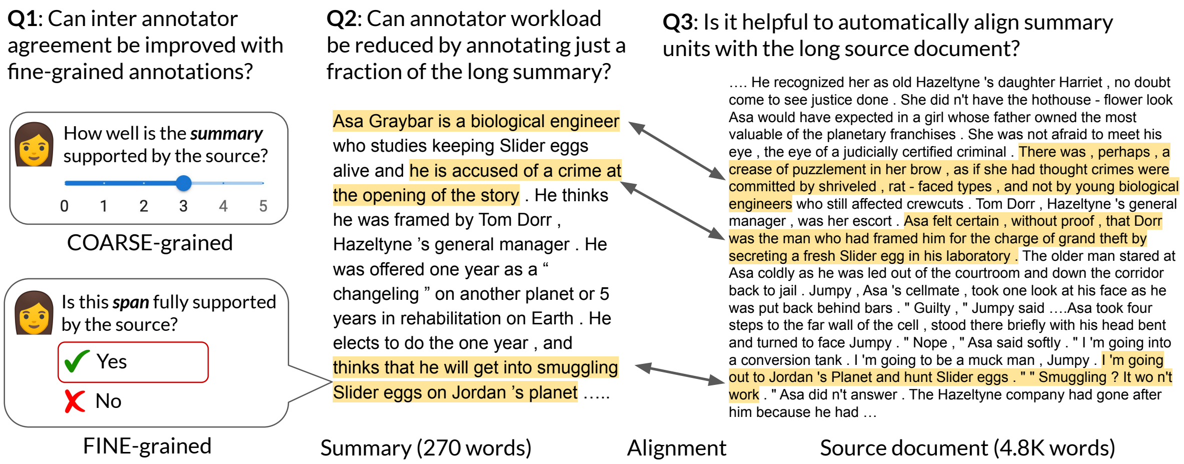
To move towards a more consistent and efficient human evaluation, we present LONGEVAL, a set of guidelines for human evaluation of faithfulness in long-form summarization (Section 3). We empirically evaluate LONGEVAL using human annotation studies on two long-form summarization datasets: SQuALITY (Wang et al., 2022) and PubMed (Cohan et al., 2018). We provide an overview of our main research questions and findings in Figure 1 and enumerate them here:

---

[1] https://github.com/martiansideofthemoon/longeval-summarization

*Work done during in an internship at AI2. Details of individual author contributions can be found here.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1650–1669
May 2-6, 2023 ©2023 Association for Computational Linguistics



Q1: Can inter annotator agreement be improved with fine-grained annotations?

Q2: Can annotator workload be reduced by annotating just a fraction of the long summary?

Q3: Is it helpful to automatically align summary units with the long source document?

Asa Graybar is a biological engineer who studies keeping Slider eggs alive and he is accused of a crime at the opening of the story . He thinks he was framed by Tom Dorr , Hazeltyne 's general manager . He was offered one year as a " changeling " on another planet or 5 years in rehabilitation on Earth . He elects to do the one year , and thinks that he will get into smuggling Slider eggs on Jordan 's planet .....

.... He recognized her as old Hazeltyne 's daughter Harriet , no doubt come to see justice done . She did n't have the hothouse - flower look Asa would have expected in a girl whose father owned the most valuable of the planetary franchises . She was not afraid to meet his eye , the eye of a judicially certified criminal . There was , perhaps , a crease of puzzlement in her brow , as if she had thought crimes were committed by shriveled , rat - faced types , and not by young biological engineers who still affected crewcuts . Tom Dorr , Hazeltyne 's general manager , was her escort . Asa felt certain , without proof , that Dorr was the man who had framed him for the charge of grand theft by secreting a fresh Slider egg in his laboratory . The older man stared at Asa coldly as he was led out of the courtroom and down the corridor back to jail . Jumpy , Asa 's cellmate , took one look at his face as he was put back behind bars . " Guilty , " Jumpy said ....Asa took four steps to the far wall of the cell , stood there briefly with his head bent and turned to face Jumpy . " Nope , " Asa said softly . " I 'm going into a conversion tank . I 'm going to be a muck man , Jumpy . I 'm going out to Jordan 's Planet and hunt Slider eggs . " " Smuggling ? It wo n't work . " Asa did n't answer . The Hazeltyne company had gone after him because he had ...

Figure 1: Overview of research questions considered in LONGEVAL. Example summary taken from SQuALITY.

RQ1: *Can inter-annotator agreement be improved while evaluating faithfulness of long-form summaries via fine-grained annotations?*

Finding: Annotating faithfulness of individual summary clauses and aggregating them leads to significantly higher inter-annotator agreement, compared to the dominant paradigm of evaluating whole summaries at once via Likert ratings (std-dev 18.5 to 6.8 on SQuALITY).

RQ2: *Can we reduce annotator workload by partially annotating a long summary while maintaining accurate faithfulness scores?*

Finding: Despite annotating a fraction of summary clauses, faithfulness scores under a reduced workload maintain high correlation with those from a full workload (0.89 Kendall's τ at 50% workload).

RQ3: *Do humans benefit from automatically aligning summary units to relevant sentences in the source document?*

Finding: Unlike suggestions in prior work on short-form summarization (Hardy et al., 2019; Kryscinski et al., 2020), aligning parts of the summary to source document is only useful when the summary is highly extractive or mostly correct.

Overall, our contributions are:
(1) a 162-paper survey of current human evaluation practices in long-form summarization;
(2) LONGEVAL, a set of three guidelines for evaluating faithfulness in long-form summarization;
(3) an empirical validation of LONGEVAL guidelines on two long-form summarization datasets in different domains (SQuALITY and PubMed);
(4) A dataset with 3-way fine-grained human faithfulness judgments for 120 SQuALITY & PubMed summaries annotated using LONGEVAL which can be used for benchmarking automatic metrics.

We open-source our human evaluation data, annotation interface, and code for future research.[1]

2 Survey of human evaluation practices

Before discussing LONGEVAL, we first attempt to understand current human evaluation practices in long-form summarization through a comprehensive survey of 162 papers. Our survey reveals several concerning trends: absence of human evaluation, non-reproducible experimental setups, lack of standardization, and complaints of long summaries being challenging and expensive to evaluate. These results show an urgent need to develop more efficient and standardized human evaluation protocols.

Selection of papers: We consider existing summarization datasets with an average summary length of at least 150 words, which includes several popular datasets like arXiv (Cohan et al., 2018), BillSum (Kornilova and Eidelman, 2019) and Multi-News (Fabbri et al., 2019); see Table 1 for a full list. For our survey, we select all papers that evaluated summarization models using at least one of these datasets.[2] All of these papers were published between June 2018 and September 2022, after the first long-form summarization datasets were released (PubMed / arXiv). Most of the 162 surveyed papers

---

[2] We exclude five papers which used long-form summarization data for pre-training only, like Wei et al. (2022).

were published in major NLP/ML venues, but we also include newer preprints from 2022.

Long-form summaries are rarely evaluated by humans. We find that 101 out of 162 papers (62%) do not perform any human evaluation. 17 papers (11%) only perform human evaluation on short summaries (datasets like XSUM, Narayan et al., 2018), for which human evaluation is much easier.

Human evaluation studies of long-form summaries are not reproducible. We further analyze the 44 papers performing human evaluation of long-form summaries to observe how often they follow reproducible practices from Gehrmann et al. (2022). Overall, we find that most studies do not follow these guidelines. Only 2 of the 44 papers release their raw human annotation data for further analysis. Only 9 papers provide details of their annotator instructions or interface, and just 12 papers perform any kind of statistical analysis, despite most papers annotating less than 50 summaries. While 33 papers report using multiple annotators per summary, only 12 report inter-annotator agreement. Finally, just 14 papers conduct human evaluation on more than one dataset (more statistics in Appendix C).

Existing human evaluation setups lack standardization. In Table 2, we catalog the wide spectrum of human evaluation setups in the surveyed papers. 37 papers collect judgments of the full-length summary at once ("COARSE-grained"), while 6 papers collect judgments at a finer granularity such as sentences or entities ("FINE-grained"). Even within a granularity, setups differ: Likert-scale (24 papers), A/B testing (13 papers), binary per-sentence labels (4 papers) are the dominant protocols. In Section 3.1, we will see that this design decision is critical since COARSE annotations have much lower inter-annotator agreement than FINE.[3]

Human evaluation of long-form summaries is challenging and expensive. Several of the surveyed papers discuss challenges in human evaluation of long-form summaries. 13 papers mention that expert annotators are necessary for human evaluation of long-form summaries, especially in technical domains like PubMed. 20 papers report that human evaluation of long-form summarization was *time-consuming*, *challenging*, and *expensive*, primarily due to the long length of the summary and source document. To tackle the issue of high annotator workload, we propose a partial annotation method in Section 3.2 and report high correlation to a full workload. Additionally, in Section 3.3 we investigate the usefulness of highlighting sentences to help annotators navigate the long source document. While this has been advocated for in short-form summary evaluation (Hardy et al., 2019; Kryscinski et al., 2020) and used in 3 surveyed long-form papers, we find that it is only helpful when summaries are mostly correct and extractive.

Dataset	|source| (words)	|summary| (words)	papers
PubMed (2018)	3092	205	59
arXiv (2018)	5906	163	55
BillSum (2019)	1284	174	19
MultiNews (2019)	2103	263	54
GovReport (2021)	7551	547	16
BookSum (2021)	5102	505	4
SummScreen (2022)	6965	227	11
SQuALITY (2022)	5194	227	1

Table 1: List of long-form summarization datasets considered in our survey along with average source document and summary lengths. Each dataset considered has at least 150 word summaries on average.

Type of human evaluation	# papers	% papers
None	101	62%
Short-form summaries only	17	11%
Likert-scale COARSE-grained	24	15%
A/B testing COARSE-grained	13	8%
Extrinsic evaluation	1	1%
Binary per sentence FINE-grained	4	2%
QA-based FINE-grained	2	1%

Table 2: Human evaluation setup in 162 summarization papers that evaluate long-form summaries. 73% of the papers do not evaluate long-form summaries with humans, while others vary significantly in their setups.

3 The LONGEVAL guidelines for faithfulness human evaluation

In Section 2, we report several concerning issues with current human evaluation practices in long-form summarization. To move towards more efficient, reproducible and standardized protocols for human evaluation, we develop the LONGEVAL guidelines (Section 3.1-3.3, see Figure 1 for an overview). We focus on human evaluation of *faithfulness*, which Wang et al. (2022) define as:

---

[3] Besides granularity, we also observe a large spectrum of annotator qualifications in our survey, ranging from MTurkers to expert graduates (Appendix C). Since non-experts are known to be unsuitable for this task (Gillick and Liu, 2010; Fabbri et al., 2021), we use experts in our work (Appendix B).

> "*Checking the factual errors in the summary, where a factual error is a statement that contradicts the source document, or is not directly stated, heavily implied, or logically entailed by the source document*"

We conduct human annotation studies to empirically motivate LONGEVAL. Our experiments are on two long-form summarization **datasets** spanning diverse domains and levels of abstractiveness:

(1) **SQuALITY** (Wang et al., 2022) is a summarization dataset in the literary domain (avg. summary length of 227 words) where summaries describe the plots of English science fiction stories. SQuALITY is highly abstractive: on average just 16% of bigrams in the summary are present in the source document. We closely follow the human evaluation setup in Wang et al. (2022), and use BART (Lewis et al., 2020) and BART-DPR (Karpukhin et al., 2020) as our summarization models along with human-written summaries.

(2) **PubMed** (Cohan et al., 2018) is a summarization dataset in the scientific domain (avg. summary length of 205 words) that pairs English biomedical articles from PubMed[4] with their abstracts as summaries. Compared to SQuALITY, PubMed is more extractive: 54% of summary bigrams are present in the source. We use BigBird-PEGASUS-large (Zaheer et al., 2020) and LongT5-large (Guo et al., 2022) as our summarization models,[5] along with human written summaries. By default, LongT5 / BigBird were highly extractive compared to human-written PubMed summaries (87% / 74% vs 54% bigram overlap with source). Hence, for half the generations we block 6-grams from being copied from the source,[6] reducing extractiveness to ∼54%. We call this setting "PubMed-ngram-block".

3.1 RQ1: Does inter-annotator agreement improve using fine-grained annotations?

In Section 2, we found that the dominant paradigm in literature (37 out of 44 papers) is to evaluate the whole summary at once ("COARSE"-grained, Figure 1 top left). 6 papers instead obtain fine-grained annotations for individual units (e.g., sentences) and average them (FINE, Figure 1 top right). Intuitively, FINE annotation has many advantages for longer summaries — it is less subjective than COARSE, since shorter spans needs to be judged rather than a long summary, and it helps localize model errors. However, the distinction between COARSE and FINE is never justified in literature, and inter-annotator agreement is rarely reported to understand the task subjectivity in each setup. To better understand the tradeoff, in this section we conduct human evaluations annotating the same set of summaries using these two different protocols.

Task formulation: Let F_{summ} denote the faithfulness score of a summary. For COARSE, k-point Likert scale ratings are obtained for the summary ($F_{\text{summ}} \in \{0, 1...k\}$), based on the faithfulness definition provided earlier. For FINE, we collect binary judgments of individual units in the summary and average them,

$$F_{\text{summ}} = \frac{1}{|\mathcal{C}_{\text{summ}}|} \sum_{c \in \mathcal{C}_{\text{summ}}} F_c, \ F_c \in \{0, 1\}$$

where $\mathcal{C}_{\text{summ}}$ is a set of units in the summary and F_c is the faithfulness judgment for the unit c. In both protocols, the faithfulness score of a system is defined as $\frac{1}{|\mathcal{S}|} \sum_{\text{summ} \in \mathcal{S}} F_{\text{summ}}$ where $\mathcal{S}$ is the set of summaries generated by the system.[7]

While sentences are a popular granularity for FINE (4 of the 6 surveyed papers), we found that summary sentences in both datasets were overloaded with information. Hence, we segment sentences on conjunctions and punctuation to obtain more atomic units as $\mathcal{C}_{\text{summ}}$. These units are often clauses,[8] similar to summary content units (SCUs) in Pyramid (Nenkova and Passonneau, 2004).

Collecting COARSE annotations: For SQuALITY, we re-use the annotations provided by Wang et al. (2022) for faithfulness assessments. In their data, three annotators give each summary a 1-100 direct assessment rating (Bojar et al., 2016). Annotators with experience in professional copyrighting and editing were hired on Upwork,[9] and these annotators were also involved in the creation of SQuALITY. Unfortunately, none of the surveyed papers that reported human evaluation results on PubMed

---

[4] `https://pubmed.ncbi.nlm.nih.gov/`

[5] LongT5 is the best publicly available PubMed summarizer. BigBird is a popular long-form summarization baseline.

[6] Reducing extractiveness / copying is also a suggestion for fair-use of copyrighted work (Harvard, 2016; UMGC, 2020).

[7] We assume all summary units get an equal weight. However, some units may be more important than others, we discuss this in the Limitations section.

[8] An even finer granularity is entities / numbers. We avoid this due to prohibitive annotation cost on long summaries.

[9] `https://www.upwork.com/`

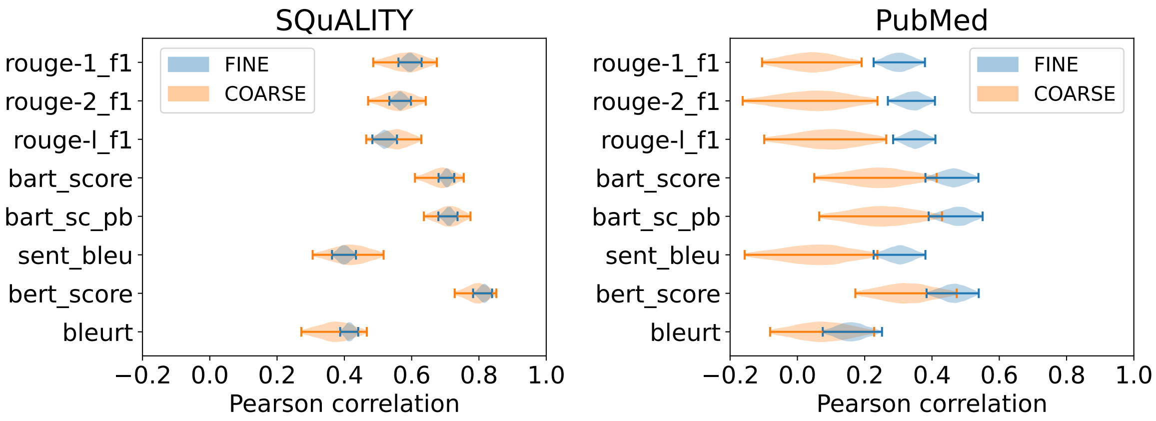


Figure 2: 95% confidence intervals of Pearson correlations between various automatic evaluation metrics and using human evaluation data collected with FINE (blue) and COARSE (orange) annotation methods. In both datasets, FINE annotations lead to much narrower CIs than COARSE annotations. See Appendix G for plot with Kendall's Tau.

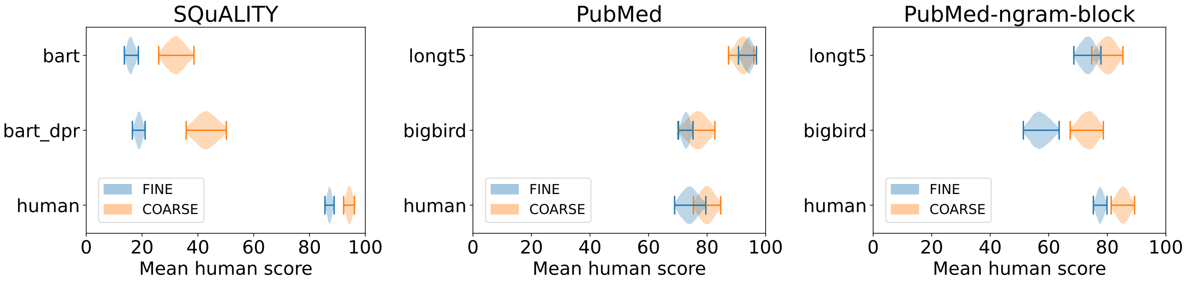


Figure 3: 95% confidence intervals of estimated model performances using FINE (blue) and COARSE (orange) annotation methods. Intervals calculated using bootstrap resampling across annotators (Appendix A). While both annotation granularities lead to similar relative ordering of systems, FINE annotations have narrower confidence intervals. The higher LongT5 score vs human in PubMed is due to highly extractive LongT5 summaries (Section 3).

released their raw human annotations.[10] Hence, we collect our own COARSE evaluations on PubMed summaries on Upwork, using freelancers with professional experience reading and writing research papers (details in Appendix B.2). We collect 3 annotations per summary and use a 5-point Likert scale, the most common choice for COARSE assessment in our survey (18 out of 38 papers). In total, 120 summaries are evaluated.

Collecting FINE annotations: For both SQuALITY and PubMed, we collect FINE annotations on Upwork (3 annotators per FINE unit) for the *same set* of 120 summaries evaluated using COARSE annotations. For SQuALITY, we hire freelancers with professional experience in English, creative writing, or education. For PubMed, we hire freelancers with prior experience analyzing biomedical articles. See Appendix B.1 for details of our annotator screening process, compensation, instructions, and screenshots of our annotation interface.

Dataset	COARSE	FINE
SQuALITY	18.5	**6.8**
PubMed	11.8	**7.3**
PubMed + ngram block	11.7	**9.3**
Average	14.0	**7.8**

Table 3: Average standard deviation of faithfulness scores across annotators on a 100-point rating scale. Lower variation means higher agreement. Overall, we find that FINE-grained annotations have higher inter-annotator agreement than COARSE-grained annotations. Note that all FINE units of a summary were annotated to obtain these results ($f = 1.0$ in Section 3.2).

FINE annotations have higher inter-annotator agreement than COARSE annotations. This leads to more confident downstream estimates. We present our results in Table 3. Overall, we observe that across all settings, FINE annotations have lower

---

[10] In our email correspondence with authors of these works, they mentioned losing access or compliance issues as reasons for not sharing human evaluations. We received some examples from Guo et al. (2021) and Ju et al. (2021) for reference.

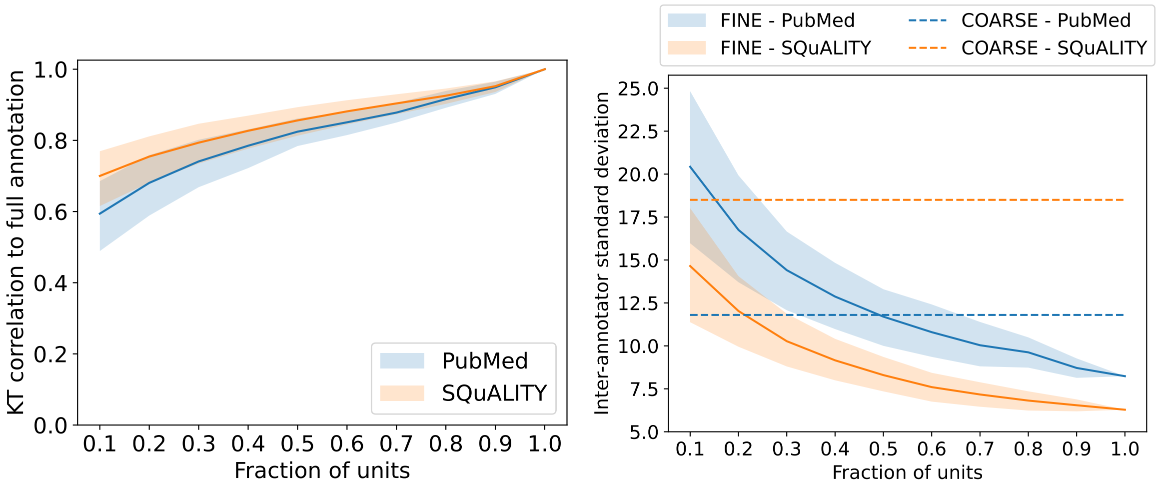


Figure 4: Accuracy and variance after annotating a fraction of units per summary (X-axis) with FINE. Despite annotating just a fraction of the summary, we observe a high segment-level Kendall tau correlation with a full annotation (left). However we observe higher inter-annotator variance as the fraction reduces (right). Confidence intervals shown are 95% and computed across 1000 random subsets (see Appendix F for left plot with Pearson).

standard deviation (and thus higher agreement) in faithfulness scores than COARSE annotations (7.8 vs 14.0 average on 100-point scaled ratings). To illustrate the importance of higher agreement, we measure its effect on two downstream statistics that human evaluation is primarily used for: (1) correlation with automatic metrics; and (2) mean system performance. We adapt the bootstrap resampling analysis[11] of Deutsch et al. (2021) to estimate confidence intervals of these two downstream statistics for COARSE and FINE.

In Figure 2, we plot the 95% confidence intervals of the Pearson correlation of various automatic evaluation metrics against FINE-grained and COARSE-grained human evaluation data. Across both datasets, FINE data leads to much narrower confidence intervals (0.15 vs 0.35 average uncertainty in Pearson correlation on PubMed) for the same number of summaries, implying higher statistical power. In Figure 3, we observe a similar trend with mean system performance. Interestingly, both annotation methods give the same relative ordering of systems (human > bart-dpr > bart for SQuALITY, human > longT5 > BigBird for PubMed-block), confirming the alignment of FINE and COARSE judgments on average.

Recommendation: Unlike the dominant trend in prior work, FINE-grained evaluations should be preferred over COARSE grained evaluation for long-form summaries. FINE annotations have lower inter-annotator variance than COARSE annotations and help localize model errors. In our setup we assume all FINE units are equally weighted while aggregating them to the final summary score. Despite this assumption, in our results we observe a consistent relative ordering of systems/metrics between COARSE and FINE annotations. Nevertheless, non-uniform weighing of units is an interesting future work direction; more in the Limitations section.

3.2 RQ2: Can we reduce annotator workload by partially annotating a long summary?

In Section 3.1, we found that FINE annotations have lower variance than COARSE annotations. However, long summaries may be composed of several units (sentences or phrases) which each require FINE annotation. This could make FINE annotation very expensive for longer summaries (as also noted in our survey). What if we instead annotate a random subset of units from the summary? While this will lower annotation cost, how accurate would these partial annotations be? We explore this tradeoff by re-using the annotations collected in Section 3.1. For every summary, we randomly sample a fraction of units $f \in \{0.1, 0.2 ... 0.9\}$ and then measure its correlation to the full set of annotations collected. Each annotator gets a different random sample of units for the same summary. In initial experiments, we found that this yielded higher accuracy than when keeping the same set of units per annotator.

---

[11] We slightly modify the algorithm in Deutsch et al. (2021) for inter-annotator variance, see Appendix A.

Partial annotation has a high correlation to full annotation, but higher variance: In Figure 4 (*left*) we plot the segment level Kendall's τ correlation (relative ordering of summary scores) between a partial annotation and full annotation for different values of f. Overall, we observe a high correlation across different values of f. Despite annotating just half the summary ($f = 0.5$), in both datasets we observe a high correlation of 0.78-0.89 Kendall's τ (95% interval) with a full annotation. Does a partial annotation preserve the variance benefits of FINE VS COARSE? In Figure 4 (*right*) we plot the inter-annotator variance for different values of f. In both datasets we find that a partial annotation has a higher variance than a full annotation. While for all values of f in SQuALITY we find that FINE annotations still have lower variance than COARSE, in PubMed COARSE has lower variance than FINE for $f <= 0.3$ with 95% confidence.

Recommendation: Having annotators judge a random subset of units in a long-form summary is a simple way to reduce FINE annotation cost, and has high correlation with a full annotation. However, it increases inter-annotator variance. Annotating 50% of the summary results in 0.78-0.89 Kendall's τ correlation, with a 30-40% increase in standard deviation compared to full FINE annotation. Partial annotation may be limited in its ability to identify issues in summaries with very few errors. However, we find that this is not the case in current systems, which are abundant in faithfulness errors.

3.3 RQ3: Is it useful to align summary units to sentences in the source document?

So far, we have focused on design decisions on the summary side of evaluation. However, evaluating faithfulness requires a comparison of facts between a summary and a *source document*. Long-form summaries tend to have long source documents (Table 1): 3.1K words for SQuALITY and 5.1K words for PubMed. In Section 2, we found several mentioned human evaluation is challenging since annotators need to read long source documents. Some prior work has suggested highlighting spans in the source document that align with the summary (Hardy et al., 2019; Kryscinski et al., 2020; Vig et al., 2021) as shown in Figure 1. However, these efforts have exclusively focused on news summarization with relatively short source documents, like CNN/DM (804 words) (Nallapati et al., 2016) or XSUM (438 words) (Narayan et al., 2018). How useful is highlighting based on alignment, or "*hints*", when the spans are chosen from much longer documents?

Algorithm	R@3	R@5	R@10
BM25 (1995)	0.38	0.46	0.56
ROUGE-1 (2004)	0.31	0.34	0.46
SIM (2019)	0.37	0.52	0.60
DPR (2020)	0.29	0.31	0.41
BERTScore-DB-XL (2020)	0.30	0.37	0.46
SummaC-NLI (2022)	0.22	0.26	0.34
MultiVers-FEVER (2022)	0.47	0.58	0.71
SuperPAL (2021)	**0.61**	**0.68**	**0.77**

Table 4: A comparison of algorithms finding the top source document sentences for summary units in SQuALITY. R@k (recall@k) denotes the fraction of times the gold sentence was in the top-k predictions.

Hints	Acc. (↑)	Agree. (↑)	Time (secs) (↓)	
	(2-way)	(Fleiss)	All	First 5
None	**93%**	**0.71**	41.4	115.6
SuperPAL	92%	0.64	48.2	84.6
Gold	92%	0.63	**40.4**	**60.4**

Table 5: Annotator performance (accuracy, agreement, median time) in detecting summary errors with different types of source document highlight hints. Overall, we see little difference across the three settings.

What is the best highlighting algorithm? We conduct a study to identify the alignment algorithm best suited for highlighting hints. We manually annotate 125 FINE units from human-written summaries of the SQuALITY validation split, marking the sentences best supporting them from the source document. We then test several candidate methods for linking summary units to the source document. These include token overlap methods like ROUGE (Lin, 2004), retrievers (Karpukhin et al., 2020), and fact verifiers (Wadden et al., 2022). In Table 4, we find that SuperPAL (Ernst et al., 2021), a weakly supervised linking algorithm, performs best (0.61 recall@3 vs the next best 0.47). To improve precision, we filter matches scoring less than 0.3 on SuperPAL, and show at most five highlights.

Do highlighted hints improve summary error detection? To answer this question, we manually perturb 50 FINE summary units in SQuALITY validation summaries, introducing entity errors or negations like Kryscinski et al. (2020). We modify the summary context of the perturbed unit to ensure summaries are self-consistent. Annotators

Question & TL;DR response	Response Snippets
Q: Did you find the highlighted hints useful while making your judgment? **TL;DR**: 4 out of 5 annotators said **Sometimes**, 1 said **Yes**. More useful for SQuALITY, summary units copied verbatim from source, correct summaries.	"**With summaries that had poor correctness, the hints were often a mess**, and even correct spans had to be carefully checked. In summaries that were more correct, I could often just read the span and remember that it was correct, and then the **hints helped me find the right source position**, or **refresh my memory** about details." "They were more useful when the summary was a **near verbatim source reproduction**." "Yes, they were useful. Often they would highlight the exact passage needed to support the summary span." "In **PubMed, they were a little more chaotic**, even for good summaries." "SQuALITY summaries consisted of sentences or parts of sentences taken straight from the story (wording was exactly as in the text). So hints often lead to the exact place." "For **SQuALITY, they were mostly accurate and helpful**. For PubMed, they were less accurate and relevant."
Q: Would the highlights have been sufficient to make judgments, or was reading the entire source document necessary? **TL;DR**: 3 out of 5 annotators said **No**, 2 said **sometimes in SQuALITY**. Reading the entire document was critical.	"**Reading the entire source document was very helpful** to understand the basic story plot" "Even when the hints were relevant, sometimes **they left out information** (like character name)..." "Initially I tried skimming ... then **concluded it's easier to read the entire document** first." "With SQuALITY there were cases where almost all of the highlights did not make any sense and nothing of that was even mentioned in the story. With **PubMed, it was even more difficult to find hints** that support the text" "**Reading the entire document was essential** to understanding the whole process, the hints in isolation were not good enough. The **hints and the summary often confused similar objects**, especially when pronouns were involved, from different parts of the source. In PubMed a similar thing happened when the source discussed what other papers had done – punctuation, acronyms, and abbreviations played a big role in providing context."
Q: Did you use Ctrl+F searches in the source document while making judgments? **TL;DR**: 4 out of 5 annotators said **Yes**, 1 said **yes only for PubMed**. Ctrl+F helped locate synonyms, entities.	"Yes, **all the time**. It was usually a **safer bet than using the hints**. The hints are given out of context of the whole SQuALITY story. There were a lot of problems with the PubMed hints involving **numbers**, which I often searched for. They were very rarely supported by the document, or contained wrong symbols (= instead of >)." "Yes, mostly in cases the highlight did not support the summary unit partially or entirely." "I used Ctrl+F when looking for very specific words, like **names**. Searching was less helpful when it came to words that had synonyms or emotions." "I did Ctrl+F on keywords taken directly from the summary unit as well as **synonyms and any specific words** that I remembered from the story that could help me get to that place in the source document quickly."

Table 6: Results and snippets from our questionnaire with FINE annotators. Overall, annotators find hints only sometimes useful, and mention reading the entire source document along with keyword searches.

are shown 50 perturbed and 50 un-perturbed summaries, and asked to annotate whether the summary units are faithful to the source in three settings:[12] (1) no highlighted hints; (2) SuperPAL highlighted hints; (3) gold hints manually annotated by us. In Table 5, we show accuracy, inter-annotator agreement, and median time[13] for each setting.

Highlighted hints have almost no effect in evaluating long-form summaries: Surprisingly, we observe that in all three metrics (accuracy, agreement, median time taken), scores are quite similar across the three settings. In fact, the "no-hint" setting scores slightly higher than the SuperPAL hint settings (93% vs 92% accuracy, 0.71 vs 0.64 Fleiss κ) and takes annotators less time (41.4 vs 48.2 seconds per unit). However, we find that hints helped annotate the first few units of a summary quicker (84.6 secs vs 115.6 secs per unit). We attribute our findings to a *learning effect* over time. FINE annotation of long-form summaries requires annotation of several units for the same document - summary pair. As annotation progresses, annotators get more familiar with the contents of the source document and summary, reducing the need for hints over time. See Appendix E for learning trajectory plots.

Questionnaire with FINE annotators confirm limited utility of hints: Our evaluation so far is limited to perturbed human summaries. How effective are hints on model-generated summaries? To answer this, we ask five of our FINE Upwork annotators (from Section 3.1) a set of three questions about their experiences using highlighted hints.[14] Detailed questionnaire results along with answer snippets are shown in Table 6. Overall, annotators find hints were useful only sometimes. Hints were *less useful* when (1) the summary unit was not supported in the source; (2) the summary unit was highly abstractive compared to the source; (3) pronouns, numbers, or abbreviations were involved; and (4) Pubmed summaries were annotated. Almost all annotators said it was necessary to read the entire source document before annotation to get an overall idea of the plot and resolve coreferences. Nearly all annotators used "Ctrl+F" searches along with hints to search for specific keywords while making judgments. This was especially true when

---

[12] To prevent any bias, each annotator receives only one of these settings for a particular summary.

[13] Calculated using the method in Akoury et al. (2020).

[14] The FINE annotations in Section 3.1 were shown hints in the source document. Since hints may not be helpful, annotators were told not to solely rely on hints for annotation.

the summary unit was incorrect, since the source document had to be thoroughly searched (beyond the hints) before confidently marking "Incorrect".

Recommendation: In contrast to recommendations in prior work, automatically highlighted hints are useful only in some specific cases of long-form summarization: mostly correct summaries, almost verbatim copied sentences. Annotators should be instructed to read the entire source document and to not rely solely on highlighted hints, since that could bias their judgments. Based on a small-scale study, we found SuperPAL (Ernst et al., 2021) to be the most accurate method for finding hints, but its performance (61% recall@3) is far from ideal.

3.4 To what extent do our findings generalize to short-form summarization?

In this work, we exclusively focus on summarization datasets with an average summary length of at least 150 words. This constraint excludes two popular benchmarks in summarization research over the last five years: CNN/DM (Nallapati et al., 2016) and XSUM (Narayan et al., 2018). How relevant are our research questions (RQs) and findings for these short-form summarization benchmarks?

On average, XSUM (24 words) and CNNDM (60 words) contain much shorter summaries than SQuALITY (237 words). XSUM outputs typically contain only 1 sentence or roughly 2-3 FINE units per summary. This blurs the distinction between FINE and COARSE units, which makes it less useful to study RQ1 in these short-form settings. The shorter length of outputs also implies that evaluation is less expensive and consumes less time, which makes our RQ2 less relevant. Finally, on average, XSUM (440 words) and CNNDM (800 words) also have much shorter source documents than datasets like SQuALITY (5200 words), reducing the need for alignment (the main premise for RQ3). The main motivation behind our study is that human evaluation of long-form summarization datasets like SQuALITY and PubMed is challenging and expensive due to the long length of the generated text. **Overall, our research questions and findings are more relevant for long-form summarization datasets than for short-form summarization datasets like XSUM and CNNDM**.

4 Related Work

A large body of recent work has focused on new *automatic* evaluation methods for summarization via NLI-based algorithms (Falke et al., 2019; Laban et al., 2022) or QA-based algorithms (Wang et al., 2020; Fabbri et al., 2022). Our work focuses on the much less studied area of *human* evaluation, the gold standard for developing automatic metrics. A notable effort in this space is the **Pyramid method** (Nenkova and Passonneau, 2004), along with work improving Pyramid efficiency (Shapira et al., 2019; Zhang and Bansal, 2021). Efficient Pyramid-like protocols have been used to collect large-scale datasets human judgments (Bhandari et al., 2020; Liu et al., 2022) in short-form news summarization tasks like CNN/DM. While these efforts focus on salience evaluation and assume access to multiple references, our work focuses on faithfulness and operates in a reference-free setting. Moreover, we focus on *long-form* summarization tasks like SQuALITY and PubMed, which are much more challenging and expensive to evaluate.

Evaluating summary faithfulness relates to **fact verification** (Vlachos and Riedel, 2014), where claim sentences are checked against a large knowledge source (Wikipedia). Prior work (Nakov et al., 2021) attempts to simplify the human fact checking process by methods like knowledge source snippets (Fan et al., 2020), similar to hint highlights (§3.3). Faithfulness in summarization differs from fact verification in three ways: (1) summaries are paragraph-long and contextual compared to single sentence stand-alone claims in fact verification; (2) summaries are grounded to a source document, compared to a large knowledge source in fact verification; (3) summaries are model-generated compared to human-written claims in fact checking datasets (Thorne et al., 2018; Wadden et al., 2020).

5 Conclusion

We present the LONGEVAL guidelines, a set of recommendations for moving towards standardized human evaluation of long-form summarization. We empirically analyze each recommendation on two datasets. Overall, we find that (1) FINE-grained annotations have lower inter-annotator variance than COARSE-grained annotations; (2) partially annotating a summary reduces annotator workload while maintaining accuracy; (3) highlighting hints in the source document has limited usefulness for evaluating long-form summaries. As future work, we plan to conduct experiments on other aspects of summarization evaluation like salience and coherence.

Limitations

Human evaluation is a noisy process with many **confounding variables**. Some of these variables were kept constant among experiments on a dataset, but modifying them could change the trends in the results. These include: (1) number of annotations per summary; (2) the specific annotation interface used; (3) granularity for FINE evaluation (sentences vs phrases); (4) Number of points in the Likert scale for COARSE evaluation; (5) set of summarization systems evaluated; and finally (6) relative (eg: A/B tests) vs absolute evaluation (eg: Likert), which has been discussed in Tang et al. (2022) for short-form news summarization datasets like CNN/DM.

Our paper is **limited to faithfulness evaluation**, but summaries are typically evaluated for salience, fluency, coherence as well (Fabbri et al., 2021). While fluency may be less of an issue due to large-scale language model pretraining (Dou et al., 2021), coherence and salience are important aspects to evaluate especially in long-form summarization (Goyal et al., 2022). Our findings may not generalize to evaluation of coherence or salience.

Our experiments in Section 3.1 **assigned an equal weight** to each FINE unit while calculating the overall score of the summary. However, the faithfulness of some FINE units may be more important than others. A non-uniform weighing of FINE units may be a good strategy if there is a notion of how critical a particular unit is for a summary's correctness. For example: (1) PICO units are critical in medical summaries (DeYoung et al., 2021); (2) the Pyramid scheme (Nenkova and Passonneau, 2004) uses a reference frequency-based unit importance, assuming access to multiple gold references. However, a consistent notion of importance is difficult to establish across different domains, and also depends on an individual consumer's preferences. Designing non-uniform weighing schemes is an interesting direction for future research.

Ethical Considerations

All experiments involving human evaluation in this paper were exempt under institutional IRB review. We fairly compensated each Upwork freelancer involved in this study, at a rate of 15-20$ per hour (respecting their suggested Upwork hourly wage). For each round of annotation, we estimated the average amount of time the task would take (by running pilots among ourselves), and provided annotators with the estimated time requirement. Most freelancers finished the task within the time window, but sometimes exceeded it by 0.5-1 hr. We compensated freelancers based on the actual time they took and their hourly wage, rather than a fixed amount per annotation.

Acknowledgments

First and foremost, we would like to thank all the nine Upwork freelancers who contributed human annotations to this project. We are very grateful to Yixiao Song, Alex Wang, John Giorgi, Dustin Wright, Yulia Otmakhova, Daniel Deutsch, Arie Cattan, Shiyue Zhang, Tanya Goyal, Greg Durrett, Marzena Karpinska, Ankita Gupta, Nader Akoury and the Semantic Scholar team for several useful discussions at various points during the project. This work was mostly done while Kalpesh Krishna (KK) was an intern at the Allen Institute for Artificial Intelligence. KK was partly supported by a Google PhD Fellowship awarded in 2021.

Author Contributions: Kalpesh Krishna led the project and performed all the technical contributions including literature review, dataset collection and processing, model implementation, annotation interface development, running experiments, and data analysis. Kalpesh also contributed to project scoping and ideation and led the writing of the paper. Erin Bransom and Bailey Kuehl helped with obtaining human judgements, including piloting the task and giving feedback, performing the annotation themselves, and hiring and managing annotators on Upwork. Pradeep Dasigi, Arman Cohan, and Kyle Lo were mentors of the project during and after Kalpesh's internship, contributing equally to project scoping, experimental design, ideation and direction throughout the course of the project and paper writing. Mohit Iyyer provided mentorship after the internship, in particular providing important feedback and direction on data analysis and contributing to paper writing.

References

Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. 2020. STORIUM: A Dataset and Evaluation Platform for Machine-in-the-Loop Story Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6470–6484, Online. Association for Computational Linguistics.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9347–9359, Online. Association for Computational Linguistics.

Ondřej Bojar, Yvette Graham, Amir Kamran, and Miloš Stanojević. 2016. Results of the WMT16 metrics shared task. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 199–231, Berlin, Germany. Association for Computational Linguistics.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*.

Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2022. SummScreen: A dataset for abstractive screenplay summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8602–8615, Dublin, Ireland. Association for Computational Linguistics.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Daniel Deutsch, Rotem Dror, and Dan Roth. 2021. A statistical analysis of summarization evaluation metrics using resampling methods. *Transactions of the Association for Computational Linguistics*, 9:1132–1146.

Daniel Deutsch and Dan Roth. 2020. SacreROUGE: An Open-Source Library for Using and Developing Summarization Evaluation Metrics. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 120–125, Online. Association for Computational Linguistics.

Jay DeYoung, Iz Beltagy, Madeleine van Zuylen, Bailey Kuehl, and Lucy Lu Wang. 2021. MS^2: Multi-document summarization of medical studies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7494–7513, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A Smith, and Yejin Choi. 2021. Scarecrow: A framework for scrutinizing machine text. *arXiv preprint arXiv:2107.01294*.

Ori Ernst, Ori Shapira, Ramakanth Pasunuru, Michael Lepioshkin, Jacob Goldberger, Mohit Bansal, and Ido Dagan. 2021. Summary-source proposition-level alignment: Task, datasets and supervised baseline. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 310–322, Online. Association for Computational Linguistics.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy.

Alexander Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220.

Angela Fan, Aleksandra Piktus, Fabio Petroni, Guillaume Wenzek, Marzieh Saeidi, Andreas Vlachos, Antoine Bordes, and Sebastian Riedel. 2020. Generating fact checking briefs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7147–7161, Online. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. 2022. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text. *arXiv preprint arXiv:2202.06935*.

Dan Gillick and Yang Liu. 2010. Non-expert evaluation of summarization systems is risky. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech* *and Language Data with Amazon's Mechanical Turk*, pages 148–151.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. Snac: Coherence error detection for narrative summarization. *arXiv preprint arXiv:2205.09641*.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*.

Yue Guo, Wei Qiu, Yizhong Wang, and Trevor Cohen. 2021. Automated lay language summarization of biomedical scientific reviews. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 160–168.

Hardy Hardy, Shashi Narayan, and Andreas Vlachos. 2019. HighRES: Highlight-based reference-less evaluation of summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3381–3392, Florence, Italy. Association for Computational Linguistics.

OGC of Harvard. 2016. Copyright and fair use.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Jiaxin Ju, Ming Liu, Huan Yee Koh, Yuan Jin, Lan Du, and Shirui Pan. 2021. Leveraging information bottleneck for scientific document summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4091–4098, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marzena Karpinska, Nader Akoury, and Mohit Iyyer. 2021. The perils of using Mechanical Turk to evaluate open-ended text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1265–1285, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Anastassia Kornilova and Vladimir Eidelman. 2019. BillSum: A corpus for automatic summarization of US legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56, Hong Kong, China. Association for Computational Linguistics.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yixin Liu, Alexander R Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, et al. 2022. Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation. *arXiv preprint arXiv:2212.07981*.

Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated fact-checking for assisting human fact-checkers. *arXiv preprint arXiv:2103.07769*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 145–152, Boston, Massachusetts, USA. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Justus J Randolph. 2005. Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online submission*.

Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. 2021. Measuring attribution in natural language generation models. *arXiv preprint arXiv:2112.12870*.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Ori Shapira, David Gabay, Yang Gao, Hadar Ronen, Ramakanth Pasunuru, Mohit Bansal, Yael Amsterdamer, and Ido Dagan. 2019. Crowdsourcing lightweight pyramids for manual summary evaluation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 682–687, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiangru Tang, Alexander Fabbri, Haoran Li, Ziming Mao, Griffin Adams, Borui Wang, Asli Celikyilmaz, Yashar Mehdad, and Dragomir Radev. 2022. Investigating crowdsourcing protocols for evaluating the factual consistency of summaries. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5680–5692, Seattle, United States. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana.

Robert J Tibshirani and Bradley Efron. 1993. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57:1–436.

Libraries at UMGC. 2020. Copyright and fair use guidelines.

Jesse Vig, Wojciech Kryscinski, Karan Goel, and Nazneen Rajani. 2021. SummVis: Interactive visual analysis of models, data, and evaluation for text summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 150–158, Online.

Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 workshop on language technologies and computational social science*, pages 18–22.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550.

David Wadden, Kyle Lo, Lucy Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. 2022. MultiVerS: Improving scientific claim verification with weak supervision and full-document context. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 61–76, Seattle, United States. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Alex Wang, Richard Yuanzhe Pang, Angelica Chen, Jason Phang, and Samuel R Bowman. 2022. Squality: Building a long-document summarization dataset the hard way. *arXiv preprint arXiv:2205.11465*.

Matthijs J Warrens. 2010. Inequalities between multi-rater kappas. *Advances in data analysis and classification*, 4(4):271–286.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Johnny Wei and Robin Jia. 2021. The statistical advantage of automatic NLG metrics at the system level. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6840–6854, Online. Association for Computational Linguistics.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. Beyond BLEU:training neural machine translation with semantic similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355, Florence, Italy. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297.

Shiyue Zhang and Mohit Bansal. 2021. Finding a balanced degree of automation for summary evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6617–6632, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Appendix

A Bootstrap analysis of inter-annotator variance

We utilize the bootstrap resampling (Tibshirani and Efron, 1993) technique described in Deutsch et al. (2021) to estimate confidence intervals for human evaluation data. At a high level, bootstrap resampling helps capture the uncertainty in a downstream test statistic by repeatedly sampling from the data with replacement. We consider two downstream test statistics in our work — (1) average system level performance; (2) correlation of human judgements to automatic metrics.

While Deutsch et al. (2021) were primarily interested in uncertainty due to the specific instances and systems evaluated, our goal is to capture uncertainty due to the inter-annotator variance. Hence unlike Deutsch et al. (2021), we sample with replacement from the set of *annotators* for every instance. Our precise formulation can be found in Algorithm 1, which operates on a $X \in \mathbb{R}^{N \times M}$ matrix of human annotations where N is the number of summaries, and M the number of annotators.

Algorithm 1 Bootstrap Confidence Interval

```
Input: $X \in \mathbb{R}^{N \times M}$, $k \in \mathbb{N}, \alpha \in [0, 1]$.
       $N$ is summaries, $M$ is annotators
Output: $(1 - \alpha) \times 100\%$-confidence interval
samples ← an empty list
for $k$ iterations do
    $X_s \leftarrow$ empty $N \times M$ matrix
    for $i \in \{1, \dots, N\}$ do
        $D \leftarrow$ samp. $\{1, \dots, M\}$ w/ repl. $M$ times
        for $j \in \{1, \dots M\}$ do
            $X_s[i, j] \leftarrow X[i, D[j]]$
        end for
    end for
    Calculate test statistic on $X_s$ and append to samples
end for
$\ell, u \leftarrow (\alpha/2) \times 100$ and $(1 - \alpha/2) \times 100$ percentiles of samples
return $\ell, u$
```

B Human evaluation details

B.1 FINE-grained evaluations of SQuALITY and PubMed summaries

We interviewed a total of 9 Upwork freelancers for the position, offering a compensation of $15-16.5 / hr (depending on their Upwork hourly rate). The screening procedure involved a qualification task on synthetically perturbed summaries from the SQuALITY dataset validation split. Similar to the final annotation task, annotators were shown a

	F-κ	R-κ	all agree
Random	0.00	0.00	25%
SQuALITY	0.74	0.76	82%
PubMed	0.53	0.65	74%

Table 7: Fleiss kappa (F-κ), Randolph kappa (R-κ), and agreement scores of our FINE annotation per summary unit. All κ scores are well above a random annotation baseline, indicating good agreement.

highlighted clause from the summary, and asked to mark whether or not it is supported by the source document. 50% of the clauses were synthetically perturbed (via negation or entity swapping as in Kryscinski et al., 2020) and manually checked to ensure they were not supported by the source document. A total of 6 freelancers scored 85% or better, and were recruited for the main set of experiments. All 9 freelancers were compensated for the screening round at the rate of 15$ USD / hr.

All six hired annotators are native or bilingual English speakers. All annotators have completed a degree at the undergraduate level and three also have Masters degrees, with the most common focuses of the degrees being English/creative writing and education. The annotators' common professional experiences include copywriting, editing, proofreading, writing, and teaching. Finally, for PubMed annotations we re-hired three annotators from the pool of six SQuALITY annotators who mentioned they had experience reading and analyzing biomedical articles. These three annotators were provided with an additional bonus of $30 after they completed all annotations.

Annotators are provided with a detailed annotation guideline along with examples of faithfulness (Table 10). Our guidelines are mostly consistent with a recently proposed set of guidelines for checking attribution in text generation (Rashkin et al., 2021). The final annotation interface is implemented in AMT Sandbox, as shown in Figure 8.

Inter-annotator agreement (binary): Much of the analysis in Section 3 uses standard deviation across summaries scores to measure inter-annotator agreement. However, another way to calculate inter-annotator agreement for FINE annotations is measuring agreement on individual units which received a Yes / No judgment. In Table 7 we show these inter-annotator agreement statistics. We measure Fleiss Kappa (Fleiss, 1971), Randolph Kappa (Randolph, 2005; Warrens, 2010), and the fraction of sentence pairs with total agreement.[15] In the table we can see all agreement statistics are well away from a uniform random annotation baseline, indicating good agreement.

B.2 COARSE-grained evaluation of PubMed summaries

None of the surveyed papers evaluating PubMed summaries with humans released their human evaluation data. Hence, we decided to collect our own COARSE annotations. Since FINE annotations (Section B.1) may have biased our original set of annotators, we hire three new annotators to perform overall assessments on a 5-point Likert scale. In other words, we use a "between-subject" experiment design to compare FINE against COARSE.

We hired three freelancers on Upwork, all of whom have extensive professional experience reading research papers (two of them had PhDs in biomedical fields). All annotators were compensated at a rate of 20$ USD / hr, their hourly rate on Upwork. All three annotators had been previously screened and hired by us for different projects in the past. Two of them had assisted us in an annotation task involved reading short summaries of biomedical academic papers and evaluating them for fluency, accuracy, correctness.

Annotators are provided with a detailed annotation guideline along with examples of faithfulness (Table 11). Our guidelines are mostly consistent with a recently proposed set of guidelines for checking attribution in text generation (Rashkin et al., 2021). The final annotation interface is implemented in LabelStudio, as shown in Figure 9.

B.3 Crowdworkers or expert annotators?

Several prior works have raised the issue of low inter-annotator agreement and poor accuracy with non-expert annotators (eg: MTurk crowdworkers) in human evaluation of summarization (Gillick and Liu, 2010; Fabbri et al., 2021; Falke et al., 2019) and open-ended long-form generation (Karpinska et al., 2021; Clark et al., 2021). In our survey (Table 9), we found the type of annotators used in long-form summarization is often not specified (16 / 43 papers). Among other papers, 10 papers use non-experts while 17 papers use expert annotators (often graduate students).

---

[15] The κ scores are measured using the library `https://github.com/statsmodels/statsmodels`.

Overall, we echo the concerns with non-expert annotators and recommend hiring freelancers on Upwork (or experts) who are well-versed with the domain for annotation. In initial experiments, we attempted to recruit Amazon Mechanical Turk crowdworkers filtered by the "Master's qualification" and having a 90%+ approval rating. In our qualification task of error detection in synthetically perturbed SQuALITY summaries, MTurkers scored just 62% (binary classification) with a three-annotator Fleiss κ of 0.15. On the other hand, Upwork freelancers (with professional writing experience) an accuracy 90% with a high inter-annotator agreement (Fleiss $\kappa = 0.71$).

C Additional Survey Statistics

In Table 8 and Table 9 we document some additional statistics for the 44 papers conducting human evaluation of long-form summarization.

Best practice	# papers
Raw human evaluation data released	2 / 44
Interface or instructions provided	9 / 44
Inter-annotator agreement reported	12 / 44
Statistical analysis conducted	12 / 44
Multiple datasets are human evaluated	14 / 44
Multiple annotators per summary	33 / 44
Annotator background reported	33 / 44
Specific summary aspects evaluated	42 / 44

Table 8: Fraction of surveyed papers following the best practices recommended by Gehrmann et al. (2022). We include only the 44 papers here which conducted a human evaluation of long-form summarization.

Type of annotator	# papers
No details specified	11 / 44
Native English speaker**	5 / 44
Mechnical Turk crowdworker	9 / 44
Non-expert volunteers	1 / 44
Extensive prior experience**	3 / 44
Graduate students / researchers	13 / 44
Upwork freelancers	2 / 44

Table 9: The types of annotators used across different long-form summarization papers. ** - No additional details were specified.

D Automatic summarization metrics used for evaluation

The following metrics are considered while measuring Pearson's correlation with our human evaluation data (Figure 2) — ROUGE-1 / 2 / F (Lin, 2004), BARTScore / BARTScore-Parabank (Yuan et al., 2021), Sentence-BLEU (Papineni et al., 2002), BERTScore (Zhang et al., 2020) and BLEURT (Sellam et al., 2020). A number of metrics were calculated using the SacreROUGE repository (Deutsch and Roth, 2020).

E Learning effect while annotating long-form summaries

In Section 3.3 we discussed a learning effect where annotators get more familiar with the contents of a source document as they annotate more FINE-grained units in a long-form summary. To better understand this effect, in Figure 5 we plot the average time taken by annotators as they progress in their annotation of a summary. Overall, we find that annotators get significantly faster in annotating the summary after the first 20% units. We hypothesize that annotators get pretty familiar with the general topics in the source document after the first few annotations, speeding up subsequent annotations.

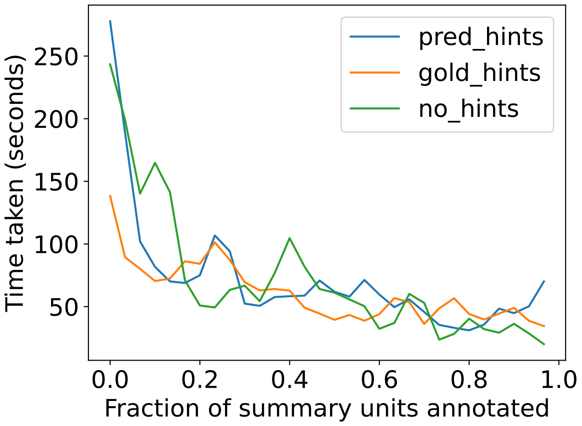


Figure 5: Learning effect over time while evaluating long-form summaries with FINE annotation. As the annotators evaluate more summary units, they learn the document better and are much faster at annotation irrespective of whether hints are shown to them.

F Partial summary annotation with pearson correlation

See Figure 6.

G Metric correlations using Kendall's Tau

See Figure 7.

In this task, you will be shown a long document ("Source Document") and its Summary. A span of text will be highlighted in the summary, and the goal is to check if this span is factually supported by the source document. You will need to choose one of two options:

1. **Yes**: if ***all*** the facts in the highlighted summary span are supported by the source document
2. **No**: if the highlighted summary span presents some information that is not supported by the source document (either a direct contradiction, or not present)

In addition to the source document, you will be provided with some highlighted text ("hints") in the source document which may help you in making a decision. Press the "Next Hint" button to scroll through the highlighted hints. Source document hints may or may not be helpful. Do not make a judgment solely based on these hints. Skim through the source document yourself / search for keywords with Ctrl + F if the hints are not helpful.
Below you can find some short representative **examples**.

Example 1
Summary (only highlighted span shown) = ... Retief is not Lemuel's cousin. ...
Source Document (snippets shown) = He eyed Retief ... "He ain't no cousin of mine," Lemuel said slowly.
Supports = Yes

Example 2
Summary (only highlighted span shown) = ... Lemeul knocks down Retief. ...
Source Document (snippets shown) = Retief's left fist shot out, smacked Lemuel's face dead center. He stumbled back, blood starting from his nose; ... He caught himself, jumped for Retief ... and met a straight right that snapped him onto his back: out cold. "Wow!" said Potter. "The stranger took Lem ... in two punches!"
Supports = No (Reason: Retief knocks down Lemeul, not the other way around.)

Example 3
Summary (only highlighted span shown) = ... Potter and his team do not trust the Embassy. ...
Source Document (snippets shown) = Lemme up. My name's Potter. Sorry 'bout that. I figured it was a Flap-jack boat; looks just like 'em . He waved a hand toward the north, where the desert lay.
Supports = No (Reason: The claim is irrelevant to the evidence.)

Table 10: Annotation guidelines provided to annotators for FINE-grained evaluation of SQuALITY and PubMed summaries. (Appendix B.1).

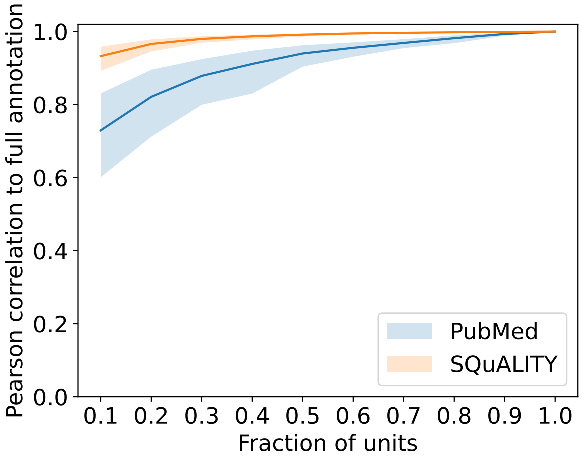


Figure 6: A version of Figure 4 using Pearson correlation instead of Kendall Tau correlation.

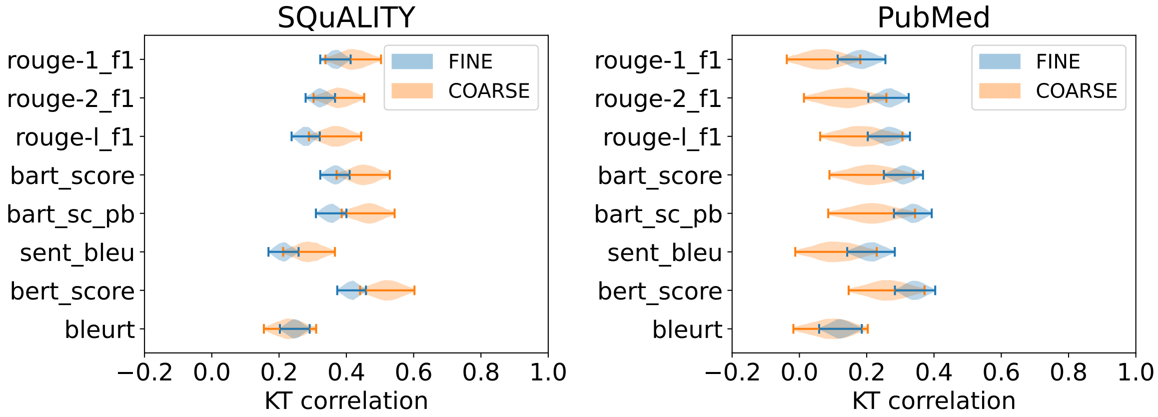
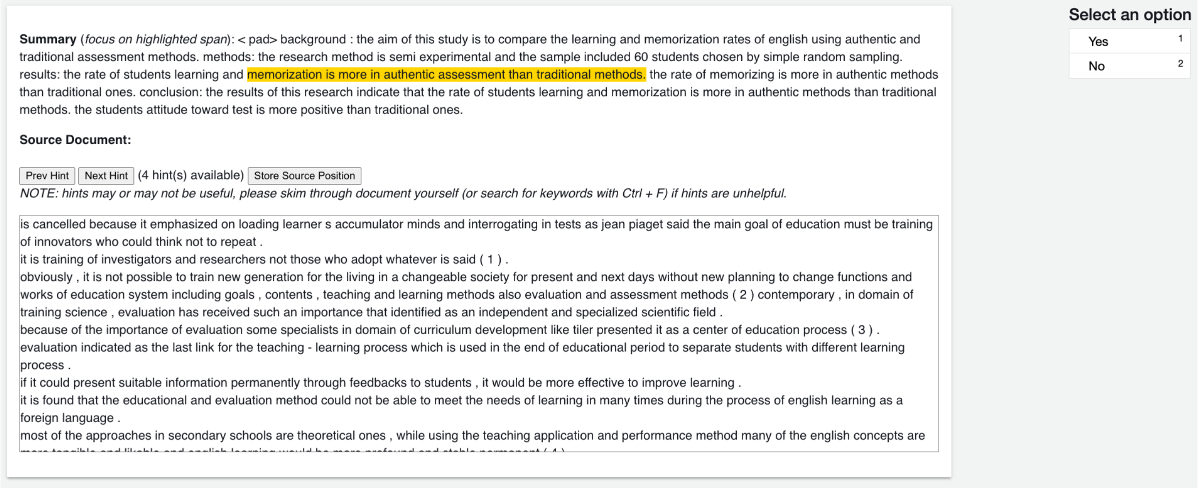


Figure 7: A version of Figure 2 using Kendall's Tau correlation instead of Pearson's correlation.



Summary (*focus on highlighted span*): < pad> background : the aim of this study is to compare the learning and memorization rates of english using authentic and traditional assessment methods. methods: the research method is semi experimental and the sample included 60 students chosen by simple random sampling. results: the rate of students learning and memorization is more in authentic assessment than traditional methods. the rate of memorizing is more in authentic methods than traditional ones. conclusion: the results of this research indicate that the rate of students learning and memorization is more in authentic methods than traditional methods. the students attitude toward test is more positive than traditional ones.

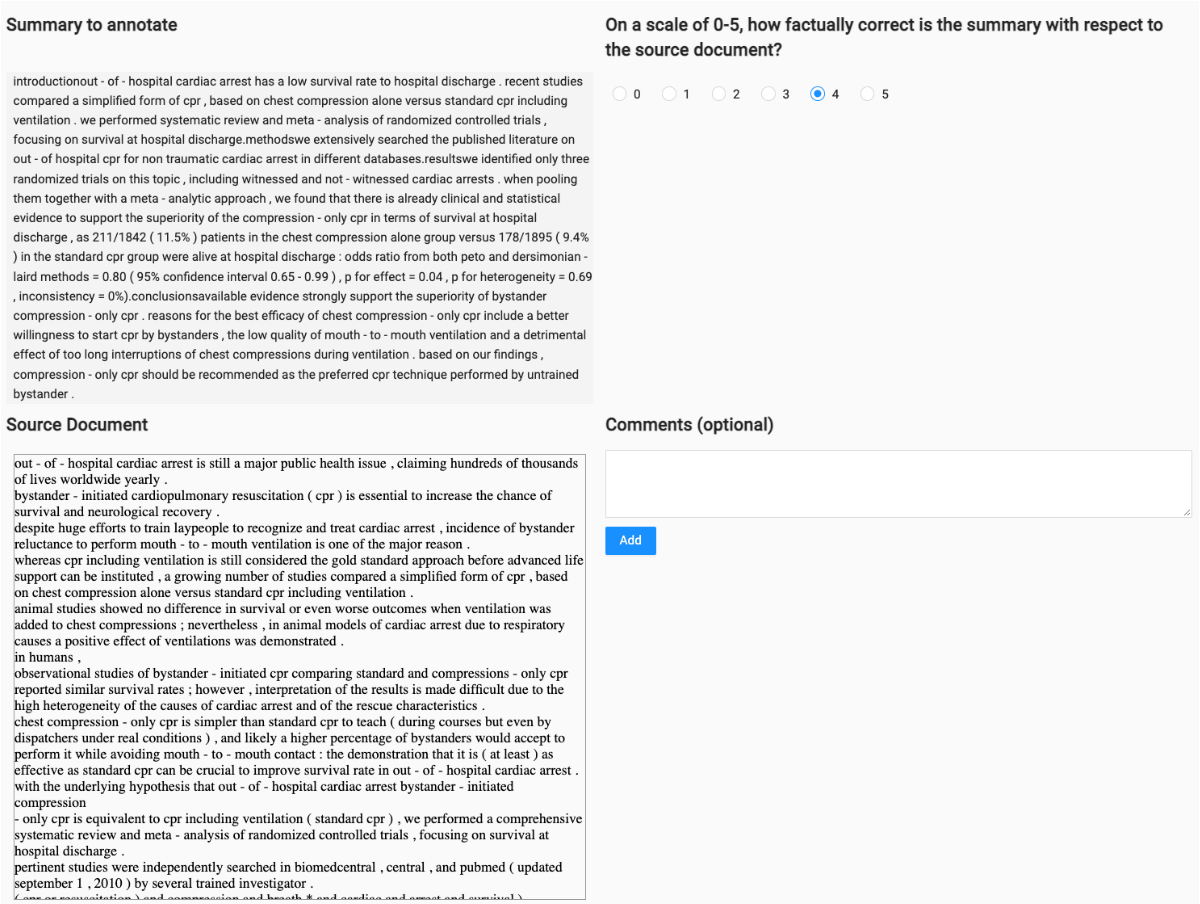
Source Document:

(4 hint(s) available)

NOTE: hints may or may not be useful, please skim through document yourself (or search for keywords with Ctrl + F) if hints are unhelpful.

is cancelled because it emphasized on loading learner s accumulator minds and interrogating in tests as jean piaget said the main goal of education must be training of innovators who could think not to repeat .
it is training of investigators and researchers not those who adopt whatever is said (1) .
obviously , it is not possible to train new generation for the living in a changeable society for present and next days without new planning to change functions and works of education system including goals , contents , teaching and learning methods also evaluation and assessment methods (2) contemporary , in domain of training science , evaluation has received such an importance that identified as an independent and specialized scientific field .
because of the importance of evaluation some specialists in domain of curriculum development like tiler presented it as a center of education process (3) .
evaluation indicated as the last link for the teaching - learning process which is used in the end of educational period to separate students with different learning process .
if it could present suitable information permanently through feedbacks to students , it would be more effective to improve learning .
it is found that the educational and evaluation method could not be able to meet the needs of learning in many times during the process of english learning as a foreign language .
most of the approaches in secondary schools are theoretical ones , while using the teaching application and performance method many of the english concepts are

Figure 8: The AMT Sandbox annotation interface used for FINE evaluation of SQuALITY and PubMed summaries (Appendix B.1).



Summary to annotate

introductionout - of - hospital cardiac arrest has a low survival rate to hospital discharge . recent studies compared a simplified form of cpr , based on chest compression alone versus standard cpr including ventilation . we performed systematic review and meta - analysis of randomized controlled trials , focusing on survival at hospital discharge.methodswe extensively searched the published literature on out - of hospital cpr for non traumatic cardiac arrest in different databases.resultswe identified only three randomized trials on this topic , including witnessed and not - witnessed cardiac arrests . when pooling them together with a meta - analytic approach , we found that there is already clinical and statistical evidence to support the superiority of the compression - only cpr in terms of survival at hospital discharge , as 211/1842 (11.5%) patients in the chest compression alone group versus 178/1895 (9.4%) in the standard cpr group were alive at hospital discharge : odds ratio from both peto and dersimonian - laird methods = 0.80 (95% confidence interval 0.65 - 0.99) , p for effect = 0.04 , p for heterogeneity = 0.69 , inconsistency = 0%).conclusionsavailable evidence strongly support the superiority of bystander compression - only cpr . reasons for the best efficacy of chest compression - only cpr include a better willingness to start cpr by bystanders , the low quality of mouth - to - mouth ventilation and a detrimental effect of too long interruptions of chest compressions during ventilation . based on our findings , compression - only cpr should be recommended as the preferred cpr technique performed by untrained bystander .

On a scale of 0-5, how factually correct is the summary with respect to the source document?

○ 0 ○ 1 ○ 2 ○ 3 ● 4 ○ 5

Source Document

out - of - hospital cardiac arrest is still a major public health issue , claiming hundreds of thousands of lives worldwide yearly .
bystander - initiated cardiopulmonary resuscitation (cpr) is essential to increase the chance of survival and neurological recovery .
despite huge efforts to train laypeople to recognize and treat cardiac arrest , incidence of bystander reluctance to perform mouth - to - mouth ventilation is one of the major reason .
whereas cpr including ventilation is still considered the gold standard approach before advanced life support can be instituted , a growing number of studies compared a simplified form of cpr , based on chest compression alone versus standard cpr including ventilation .
animal studies showed no difference in survival or even worse outcomes when ventilation was added to chest compressions ; nevertheless , in animal models of cardiac arrest due to respiratory causes a positive effect of ventilations was demonstrated .
in humans ,
observational studies of bystander - initiated cpr comparing standard and compressions - only cpr reported similar survival rates ; however , interpretation of the results is made difficult due to the high heterogeneity of the causes of cardiac arrest and of the rescue characteristics .
chest compression - only cpr is simpler than standard cpr to teach (during courses but even by dispatchers under real conditions) , and likely a higher percentage of bystanders would accept to perform it while avoiding mouth - to - mouth contact : the demonstration that it is (at least) as effective as standard cpr can be crucial to improve survival rate in out - of - hospital cardiac arrest .
with the underlying hypothesis that out - of - hospital cardiac arrest bystander - initiated compression
- only cpr is equivalent to cpr including ventilation (standard cpr) , we performed a comprehensive systematic review and meta - analysis of randomized controlled trials , focusing on survival at hospital discharge .
pertinent studies were independently searched in biomedcentral , central , and pubmed (updated september 1 , 2010) by several trained investigator .

Comments (optional)

Figure 9: The LabelStudio annotation interface used for COARSE evaluation of PubMed summaries (Appendix B.2).

Instructions for Likert-scale evaluation. Please read all instructions before starting the annotation.

Setup

1. Start by signing up on Label Studio, you will need to provide an email ID and password. It's okay to use a non-existent throw-away email ID here. Also, do not use any personal / sensitive passwords (but make sure to remember your email / password for logging in next time!). Click on the box saying "<your name> — Summarization Evaluation"
2. In this batch a total of 30 summaries need to be evaluated. Every three consecutive rows are different summaries of the same source document. You can evaluate a summary by clicking on a row, and annotating it. Optionally, you can click on "Label All Tasks" at the top of the screen.

Annotation Task

Each summary needs to be evaluated for its "correctness". You need to provide a 0-5 judgment for the entire summary, where "correctness" can be defined as, "The absence of factual errors in the summary, where a factual error is a statement that contradicts the source document, or is not directly stated, heavily implied, or logically entailed by the source document". For example,
Source Document (snippet shown) = ….. Vitamin C was discovered in 1912, isolated in 1928, and, in 1933, was the first vitamin to be chemically produced. It is on the World Health Organization's List of Essential Medicines. Vitamin C is available as an inexpensive generic and over-the-counter medication. Partly for its discovery, Albert Szent-Györgyi and Walter Norman Haworth were awarded the 1937 Nobel Prizes in Physiology and Medicine and Chemistry, respectively. Foods containing vitamin C include citrus fruits, kiwifruit, guava, broccoli, Brussels sprouts, bell peppers, potatoes, and strawberries. Prolonged storage or cooking may reduce vitamin C content in foods. ….
Summary 1 (snippet shown) = … Chicken contains vitamin C …
Summary 2 (snippet shown) = … Albert Szent-Györgyi won the 1955 Nobel Prize for discovering Vitamin C …
Summary 3 (snippet shown) = … Vitamin C was the first chemically produced Vitamin …
Summary 4 (snippet shown) = … Apple contains vitamin C …
Errors marked in red. Here, the snippets for summary 1 are incorrect, summary 2 partially correct, and summary 3 completely correct with respect to the source document. Summary 4 is incorrect with respect to the source document (since it's never discussed), but a globally correct fact. You should treat such a summary as incorrect since it is not mentioned in the source document.
(This is an illustrative example only, the actual annotation task has much longer summaries / source documents.)
The rating scale is from 0 to 5, where 0 is the lowest possible rating (most or all of the summary is wrong / irrelevant to the source document), and 5 is the highest rating (most or all of the summary is correct).
While it is compulsory to provide a judgment from 0 to 5 for each summary, you can optionally provide additional comments in your annotation. For instance, if the judgment needs to be more nuanced than a 5-point scale, you prefer to mark something like "3.5", or you would like to add some other notes about your judgment.
Press "Submit" after you have provided your annotation.

Suggested workflow

Every three consecutive rows contain different summaries for the same source document. We suggest the following workflow while annotating documents —
1. Spend the first 15 minutes reading the source document and getting a general sense of the facts mentioned in the document.
2. Spend 5 minutes to read and annotate the summaries in each of the three consecutive rows which correspond to the same document. Add optional comments / notes if necessary.
3. In the last 5 minutes, re-calibrate your ratings across the three rows if needed (for instance, you significantly preferred the correctness of summary 1 vs summary 2, but you gave it the same rating in the initial pass). Add optional comments / notes if necessary.
Following this workflow, it should take 35 minutes to annotate each set of 3 rows. For 30 rows, this should take 6 hrs.

Table 11: Annotation guidelines provided to annotators for COARSE evaluation of PubMed summaries (Appendix B.2).

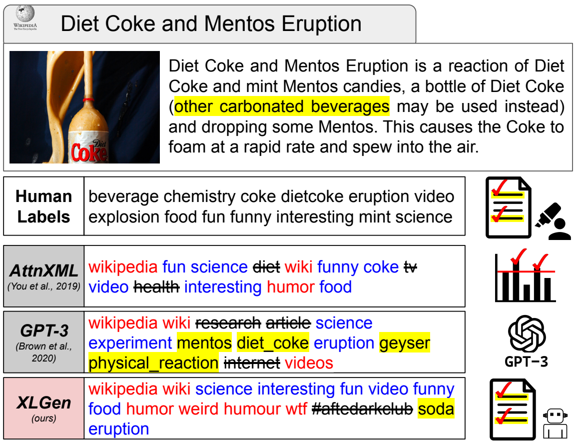
Cluster-Guided Label Generation in Extreme Multi-Label Classification

Taehee Jung ♣ * **Joo-Kyung Kim** ♣ **Sungjin Lee** ♣ **Dongyeop Kang** ♡

♣Amazon Alexa AI ♡University of Minnesota
`{jungtaeh,jookyk,sungjinl}@amazon.com` `dongyeop@umn.edu`

Abstract

For extreme multi-label classification (XMC), existing classification-based models poorly perform for tail labels and often ignore the semantic relations among labels, like treating "Wikipedia" and "Wiki" as independent and separate labels. In this paper, we cast XMC as a *generation* task (`XLGen`), where we benefit from pre-trained text-to-text models. However, generating labels from the extremely large label space is challenging without any constraints or guidance. We, therefore, propose to guide label generation using label cluster information to hierarchically generate lower-level labels. We also find that frequency-based label ordering and using decoding ensemble methods are critical factors for the improvements in `XLGen`. `XLGen` with cluster guidance significantly outperforms the classification and generation baselines on tail labels, and also generally improves the overall performance in four popular XMC benchmarks. In human evaluation, we also find `XLGen` generates unseen but plausible labels. Our code is now available at `https://github.com/alexa/xlgen-eacl-2023`.



Diet Coke and Mentos Eruption

Diet Coke and Mentos Eruption is a reaction of Diet Coke and mint Mentos candies, a bottle of Diet Coke (other carbonated beverages may be used instead) and dropping some Mentos. This causes the Coke to foam at a rapid rate and spew into the air.

Human Labels	beverage chemistry coke dietcoke eruption video explosion food fun funny interesting mint science
AttnXML *(You et al., 2019)*	wikipedia fun science ~~diet~~ wiki funny coke ~~tv~~ video ~~health~~ interesting humor food
GPT-3 *(Brown et al., 2020)*	wikipedia wiki ~~research~~ ~~article~~ science experiment mentos diet_coke eruption geyser physical_reaction ~~internet~~ videos
XLGen *(ours)*	wikipedia wiki science interesting fun video funny food humor weird humour wtf ~~#aftedarkclub~~ soda eruption

Figure 1: The predicted and generated labels from AttentionXML (You et al., 2019), GPT-3 (Brown et al., 2020), and `XLGen-BCL`, respectively, for Wikipedia page on diet coke and mentos eruption. We marked labels to be correct (blue), wrong (~~strikethrough~~), and positive unlabeled (red). Our `XLGen` could generate completely new labels from input text, e.g., soda, inferred from context that other carbonated beverages can replace diet coke.

1 Introduction

Extreme multi-label classification (XMC) is a task to predict multiple relevant labels for a given input where the label space is extremely large. Conventional approaches for XMC decompose the problem into a set of binary classifications, training one-vs-all classifiers for each label. However, they encounter several issues in practical use cases.

First, the labels in XMC are long-tail distributed. In other words, only a few labels have sufficient positive samples, thereby the other infrequent labels could be rarely predicted during inference as we see the heavily right-skewed distribution in the long-tail in Figure 2a. Second, multi-label classification techniques such as one-by-one and label powerset (Gibaja, 2015) assume independent and identically distributed labels, while the user-generated labels in XMC are dependent on each other. Moreover, annotated labels are only a portion of possible labels, thus, resulting in positive and unlabeled (PU) setting (Yu et al., 2014; Kanehira and Harada, 2016).

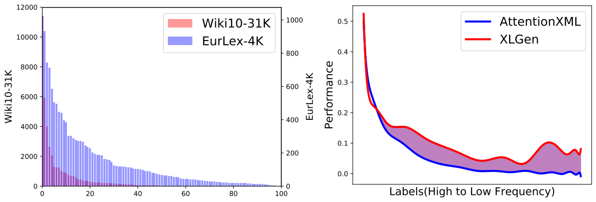
In this paper, we tackle extreme multi-label classification with a *generative approach*, called extreme multi-label generation (`XLGen`). In particular, we fine-tune a pre-trained Transformer-based encoder-decoder model (Raffel et al., 2020) with input documents and their known positive labels. This (label) generation approach is more intuitive and closely similar to how humans tag documents with text labels without a fine-grained ontology or guideline.

However, the generated labels from the extremely large label space without any constraints

---

[*] Part of this work was done during an internship at Amazon Alexa AI.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1670–1685
May 2-6, 2023 ©2023 Association for Computational Linguistics



(a) Frequency Histogram (b) Label-wise Recall

Figure 2: (a) Frequency histograms of top-100 occurring labels in EUR-LEX (blue) and WIKI10-31K (red) (b) Label-wise recall scores from AttentionXML and `XLGen-BCL` on Wiki10-31K. For the presentation, graphs are smoothed by least-squared polynomial regression.

and/or guidance can be noisy and not cover infrequent labels. To address this issue, we propose a method to leverage label clusters into generation: first, generate cluster IDs of semantically similar labels, and then generate text labels utilizing the cluster IDs as additional contextual inputs. Specifically, we propose two `XLGen` architectures (`XLGen-BCL`, `XLGen-MCG`) in which such clusters are jointly trained with labels in different ways. Using clusters for label generation is motivated by showing label categories to human annotators. As an example, humans often start by setting high-level topics first and then hierarchically create actual tags under each high-level topic. The clusters, however, are treated as additional guidance rather than a constraint since we do not restrict the model to only predict labels under the given clusters.

Similarly to `XLGen`, Simig et al. (2022) proposed GROOV, which fine-tunes T5 to generate labels in XMC. In particular, GROOV aims to implement a label order invariant training objective by randomly shuffling label orders and using multi-softmax function, which does not penalize if any first tokens of true labels are predicted regardless of the label orders. However, it does not outperform classification baselines consistently, and we empirically find that label order by frequency helps alleviate the issue in our ablation study.

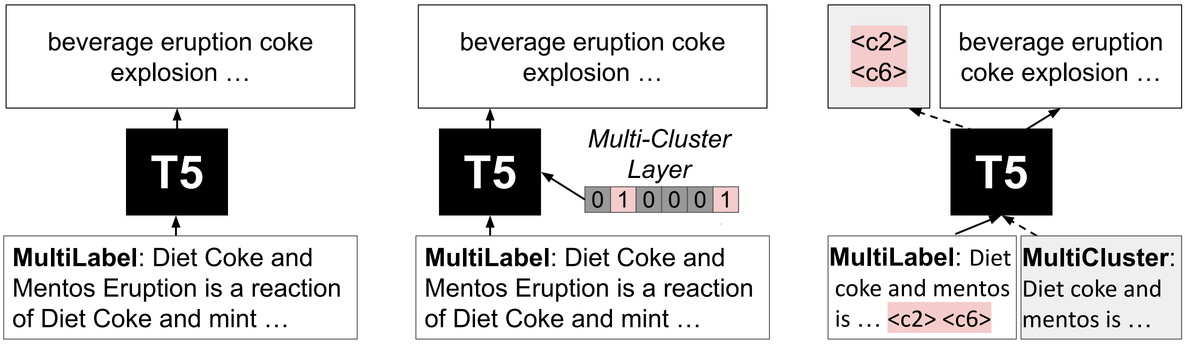
Our experiment shows that `XLGen` (and its variants) outperforms classification baselines on four XMC benchmarks. Furthermore, `XLGen` with cluster guidance (`XLGen-BCL` and `-MCG`) significantly and consistently outperforms the classification and generation baseline (`XLGen-base`) on tail labels, respectively. The effect on tail labels from `XLGen-BCL` is demonstrated in Figure 2b.

Figure 1 shows predicted or generated labels from different models. A Wikipedia page of diet coke and mentos eruption has true labels such as "beverage", "fun", and "eruption". We find `XLGen` can also generate a new positive label "soda" based on the context of "carbonated beverage" in the input text. From a human evaluation (S6.1), we find newly generated labels by `XLGen` are highly associated with the input texts, which potentially helps automatically find new labels without manual tagging. We also show the generated labels from large language models (LLMs) like GPT-3 (Brown et al., 2020): we find that the overall performance of in-context learning is significantly less than `XLGen` (See §4.4 for details), but LLMs could generate reasonable labels with a few examples as `XLGen` does.

2 Related Work

Extreme multi-label classification (XMC). Classification-based approaches on XMC suffer from dealing with enormous label spaces under the one-vs-all classification setting (Babbar and Schölkopf, 2017; Yen et al., 2017; Jain et al., 2019). To address the efficiency issue, state-of-the-art XMC models partition label space to the scalable subsets via hierarchical clustering (Prabhu et al., 2018; Wydmuch et al., 2018; You et al., 2019; Chang et al., 2020; Yu et al., 2022; Tagami, 2017), graph-based approximations (Jain et al., 2019; Zong and Sun, 2022), or random forest (Siblini et al., 2018). However, they still suffer from predicting tail or unseen labels. To efficiently deal with such long-tail issues, few-shot learning frameworks and methods (Gupta et al., 2021; Xiong et al., 2022) are proposed. Rather, we show how encoder-decoder language model can improve tail label scores by fine-tuning it with guidance from label clusters.

LMs and Generative approach in XMC. For XMC, pre-trained LMs such as XLNet (Ye et al., 2020) and Transformer (Chang et al., 2020) are used but only for encoding input texts, thus, it still relies on the classification approach for label prediction. Previously, other works address multi-label classification with generative approaches (Nam et al., 2017; Tsai and Lee, 2020; Yang et al., 2018, 2019; Zhang et al., 2021b) but in much smaller label spaces. Recently, Simig et al. (2022) also used T5 (Raffel et al., 2020) for directly generating labels in end-to-end manners for XMC, but



(a) `XLGen-base` (b) `XLGen-BCL` (c) `XLGen-MCG`

Figure 3: Three `XLGen` architectures, where the basic model can be any pre-trained text-to-text models like T5. (a, `base`): Simple fine-tuning that encodes input text with a prefix of task name and decodes text of label sequences. (b, `BCL`): A fine-tuning with a multi-cluster prediction layer as an auxiliary task. (c, `MCG`): A multi-task fine-tuning with multi-cluster generation and multi-label generation (MCG); two tasks are trained simultaneously and at decoding time the output of the cluster generation is concatenated to the input for the label generation.

its performance was not convincing compared to classification-based models.

Positive and unlabeled data. In practice, XMC is inherently with positive and unlabeled (PU) setting as the label space is extremely large and it is infeasible to manually review all the labels (Kim and Kim, 2020). Multi-label performances on PU tasks can be simulated by leaving only a few labels per train instance (e.g., leaving 8 out of 10 positive labels in one instance for label deficit rate 20%) positive (Hu et al., 2021). In this work, we show how `XLGen` works on such PU settings in §4.3.

3 Extreme Multi-label Generation (`XLGen`) with Cluster Guidance

Classifying a document with multiple labels can be regarded as tagging a document with possible topical labels, which is basically decoding the free-formed text labels in an encoder-decoder setting. Moreover, if encoder and decoder are trained on large text corpora, labels are generated with an understanding of their lexical variations and semantic similarities. Our baseline framework fine-tunes a pre-trained Transformer using the input text as an encoder input and the label sequences as a decoder output. In addition, to more effectively address a huge number of labels in a long-tail distribution, we propose two different architectures, `XLGen-BCL` (§3.2) and `XLGen-MCG` (§3.3), generating labels guided by pre-computed cluster information, inspired by class-based language models (Brown et al., 1992).

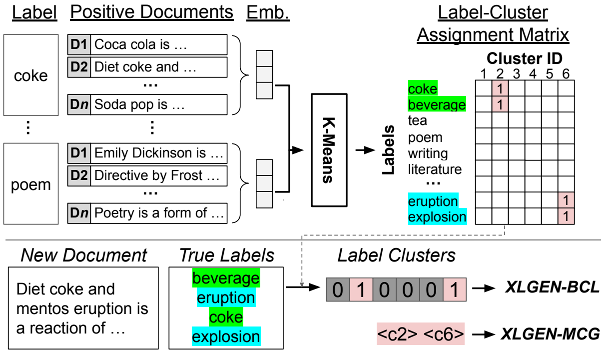


Figure 4: Architecture of pre-computed label clusters. For each label, we use the averaged embedding of positive documents including the label in train set and compute the label cluster assignment matrix (top). For training, we assign label clusters for each training document based on the true labels using the label cluster assignment matrix (bottom).

Pre-computed clustering. We compute label clusters using K-Means algorithm, as depicted in Figure 4. We first obtain label features using average embedding of positive documents in a train set, following Chang et al. (2020), and compute the label-cluster assignment matrix. Label clusters are assigned to each training document using this matrix and ground-truth labels, and used as a multi-cluster prediction layer for `XLGen-BCL` training or sequence of cluster IDs for `XLGen-MCG` training.

3.1 Baseline Fine-Tuning

Figure 3a shows our baseline `XLGen` which simply fine-tunes text-to-text Transformers, e.g., T5 (Raffel et al., 2020) or BART (Lewis et al., 2020), as our encoder-decoder framework on XMC dataset.

- *Input*: **task prefix**: input text
- *Output*: A sequence of label texts

For encoding, we add a prefix token 'MultiLabel' to the input text to inform the task type. Then, the output labels are generated as a sequence of labels in decoding. The model is fine-tuned with cross-entropy loss ($\mathcal{L}_{xent}$) given the sequence of label texts. In practice, the order of labels in decoding significantly influences the model performance. Following Yang et al. (2018), we sort the target labels in decreasing order of the frequencies. We also investigate various ordering effects and their impact on performance in §5.1.

3.2 Fine-Tuning with Cluster Prediction

Figure 3b shows the fine-tuning of text-to-text with an additional multi-cluster prediction layer (`XLGen-BCL`). By doing so, we expect the model learns label similarities and hence biases itself to generate labels relevant to the given cluster.

- *Input*: **task prefix**: input text
- *Multi-Cluster Layer*: a vector of $v_1, ...v_k$ where $v_i = 1$ if i^{th} cluster c_i is a positive cluster; otherwise $v_i = 0$ (1 1 0 ... 1...)
- *Output*: A sequence of label texts

The multi-cluster prediction layer is a vector of 0 or 1 that corresponds to the assigned clusters of instance, and is trained using the sequence of the last layer's hidden states of the encoder with a binary cross-entropy loss, $\mathcal{L}_{bce}$. The final objective is as follows:

$$\mathcal{L}_{xmc-bcl} = \mathcal{L}_{xent} + \lambda \mathcal{L}_{bce} \quad (1)$$

where $\mathcal{L}_{xent}$ is a cross-entropy loss term for the original text-to-text framework and $\mathcal{L}_{bce}$ is a binary cross entropy loss term for the cluster layer. λ is a weighting parameter for controlling $\mathcal{L}_{bce}$, to be chosen by dev-set performance.

3.3 Fine-Tuning with Cluster Decoding

`XLGen-BCL` utilizes a cluster prediction only as an auxiliary task to improve representations for a label prediction, thus, predicted clusters are not used in inference. Figure 3c shows the third variant, `XLGen` with a multi-cluster generation (`MCG`), which leverages predicted clusters as additional input tokens so that the cluster information can be used in inference.

- *Input1*: **task prefix**: input text ; a sequence of positive cluster IDs (c_1 c_2 c_{11}..)
- *Output1*: A sequence of label texts
- *Input2*: **cluster prediction prefix**: input text
- *Output2*: A sequence of positive cluster IDs (c_1 c_2 c_{11}..)

We add a sequence of cluster IDs to the input text so that cluster information can be used while training (Input1-Output1). On the other hand, we have a new task with a clustering prefix 'MultiCluster' appended to the input text and predicts the sequence of labels (Input2-Output2). In training, these two tasks are trained simultaneously. Note that in inference, the predicted cluster IDs are appended to Input1 for the final label generation.

	$\lvert D_{trn}\rvert$	$\lvert D_{tst}\rvert$	$\lvert L_{seen}\rvert$	$\lvert L_{unseen}\rvert$
EURLEX-4K	15,449	3,865	2,473	155
AMZNCAT-13K	1,186,239	306,782	13,275	0
WIKI10-31K	14,146	6,616	21,060	991
WIKI-500K	1,779,881	769,421	498,152	917

Table 1: Data statistics of the benchmark datasets; the number of train examples ($|D_{trn}|$), number of test examples ($|D_{tst}|$), number of labels in both train and test set ($|L_{seen}|$), and number of labels only in test set ($|L_{unseen}|$), which is zero-occurred labels in Table 3.

4 Experiments

4.1 Experimental Setups

Datasets. We use four widely used XMC benchmark datasets; three large-scale datasets with 4K∼30K labels (EURLEX-4K,AMZNCAT-13K, and WIKI10-31K) and one very-large-scale datasets with 500K labels (WIKI-500K). See Table 1 for the detailed data statistics.

Baseline and `XLGen` Training. We compare `XLGen` with three state-of-the-art baselines in XMC tasks; AttentionXML (You et al., 2019), X-Transformer (Chang et al., 2020), and XR-Linear (Yu et al., 2022). Note that all baseline models partition labels using hierarchical clustering. See A.1 for the detailed setups of baselines. Note it is common to upscale scores by ensemble learning for XMC baselines. However, for a fair comparison, we do not use any ensemble models for `XLGen` and baselines.

We train `XLGen` with T5 because BART (Lewis et al., 2020) performs worse for our task as shown in Table 13. By default, we sort labels in the decreasing frequency order to provide the training target sequence and infer the labels by beam search with size 5. For `XLGen-BCL` and `XLGen-MCG`, cluster sizes are optimized by dev set performance.

	EurLex-4K		AmznCat-13K		Wiki10-31K		Wiki-500K	
	$Mic.$	$Mac.$	$Mic.$	$Mac.$	$Mic.$	$Mac.$	$Mic.$	$Mac.$
XR-Transformer	39.1	12.3	64.0	17.0	21.4	2.8	30.5	7.8
XR-Linear	44.6	15.1	53.2	18.6	19.2	3.6	17.2	3.3
AttentionXML	59.9	24.9	70.1	30.0	37.3	4.6	53.6	21.0
`XLGen-base`	59.8	27.5	69.8	38.8	**37.6**	**9.9**	55.1	**35.0**
`XLGen-BCL`	**60.7**	**28.4**	70.0	37.7	**37.6**	9.8	**55.4**	33.5
`XLGen-MCG`	60.2	28.2	**71.8**	**46.4**	37.4	9.6	**55.4**	33.6

Table 2: Full label performance. We report micro-averaged ($Mic.$) and macro-averaged ($Mac.$) F1 scores.

	EurLex-4K		Wiki10-31K		Wiki-500K	
	0-st	1-st	0-st	1-st	0-st	1-st
XR-Transformer	0.0	0.5	0.0	1.7	0.0	0.0
XR-Linear	0.0	1.1	0.0	2.3	0.0	0.1
AttentionXML	0.0	2.4	0.0	0.2	0.0	1.3
`XLGen-base`	3.2	3.5	2.9	**8.4**	22.5	24.1
`XLGen-BCL`	4.3	**4.1**	3.3	**8.4**	23.2	24.8
`XLGen-MCG`	**4.5**	2.7	**11.1**	8.1	**23.7**	**25.5**

Table 3: Macro-averaged F1 scores in tail labels, which never occurred (0-st) or occurred once (1-st) in train set.

	EurLex-4K	Wiki10-31K	Wiki-500K
XR-Transformer	8.6	3.3	7.0
XR-Linear	8.8	2.7	2.8
AttentionXML	18.7	1.3	11.3
`XLGen-base`	18.8	7.7	31.6
`XLGen-BCL`	19.3	8.0	31.4
`XLGen-MCG`	**21.2**	**10.1**	**32.7**

Table 4: Macro-averaged F1 scores in PU setting (50% of label deficit ratio).

We get the input text embedding by averaging the last hidden states from the pre-trained T5 encoder since T5 model does not have a CLS token. See A.2 to check more details.

4.2 Evaluation Metrics

Following the prior work in XMC, we report F1 score (F@k) of top-k label probabilities as a supplementary metric in A.4. However, such ranking metrics are not applicable to label generation tasks since the generative model only output positive label texts *sequentially* and the order of generated labels does not align with the confidence of the label; in other words, the formerly generated labels do not need to be more confident than the latter ones. Thus, we use conventional multi-label classification metrics, like micro-averaged F1 score ($Mic.$) and macro-averaged F1 score ($Mac.$), as main evaluation metrics.

In principle, evaluating XMC task with the ranking format is not appropriate for most cases as it requires predicting the number of correct labels as well (Amigo and Delgado, 2022). We therefore select predicted labels only when the predicted score is greater than the threshold optimized from the validation set as in You et al. (2019).

4.3 Results

We compare performances of `XLGen` and baselines in full labels (Table 2), tail labels (Table 3), and PU data setting (Table 4). For tail label and PU setting, we do not include AmznCat-13K as it does not have zero-occurred labels. The best scores are **bold** and the second best scores are underlined. See A.4 for the full scores on tail labels and PU setting.

Full label performance. In the evaluation with full benchmark sets, all the `XLGen` models show outperforming or competitive performance compared to the classification-based baselines. For macro F1 scores, `XLGen` models hugely outperform the baselines, which empirically represents that our approach is strong at predicting infrequent but correct labels. In other words, `XLGen` models are less biased to predicting frequent labels. Compared to `XLGen-base`, both `XLGen-BCL` and `XLGen-MCG` generally show better performance, which demonstrates the effectiveness of the cluster prediction as an auxiliary loss.

Tail label performance. We measure macro F1 scores only for tail labels which never or one-time occur in the train set. We find every baseline extremely suffers from the tail labels, while `XLGen` shows significant improvements, demonstrating the power of generative models for long-tail labels. Surprisingly, `XLGen` even predicts never-seen, zero-occurred labels, only inferred from the semantic meaning of the input text. Similarly to full label performance, `XLGen-BCL` and `XLGen-MCG` perform better than `XLGen-base`, indicating

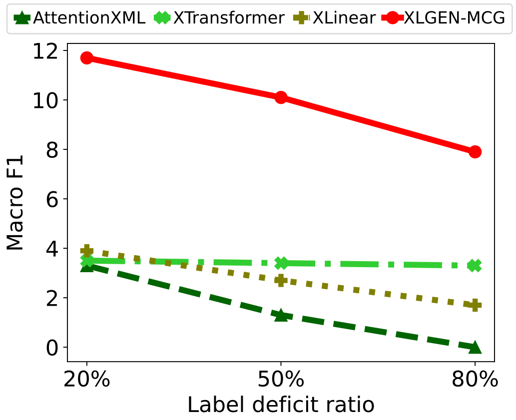


Figure 5: Macro-averaged F1 scores in PU setting on WIKI10-31K.

that guidance of label cluster improves tail label performance as well.

PU setting. In XMC, it is infeasible to annotate all relevant labels for an input text by checking every millions of labels. Therefore, many XMC datasets are indeed in PU setting. To evaluate the robustness against the positive and unlabeled properties, following Hu et al. (2021), we make PU data setting by randomly eliminating positive labels for each instance with 50% of deficit rate.

As `XLGen` is trained with fewer positive labels in PU settings, the generated output labels tend to be fewer as well, causing lower recall than expected. To increase the recall, we generate diverse label sequences using various sampling schemes in inference, which we call *ensemble generation*. We combine generated results from three decoding strategies; beam search with size 5, Top $P + K$ sampling, and sampling with 0.8 temperature.

We find `XLGen` models outperform the baselines. Specifically, `XLGen-MCG` shows significantly strong scores, which indicates having predicted clusters as an additional input helps predict infrequent but correct labels. In Figure 5, we additionally visualize macro F1 scores of PU settings on WIKI10-31K with various deficit rates. Although `XLGen-MCG` drops with an increasing deficit rate, it still shows significant gaps with baseline scores.

4.4 Feasibility of in-context learning in XMC

In-context learning (Brown et al., 2020) shows a potential of generating unseen but positive labels as depicted in Figure 1, such as “geyser” and “physical_reaction”. In order to thoroughly validate the feasibility of in-context learning in XMC problems, we select 100 samples randomly from EURLEX-4K and WIKI10-31K, and predict their labels using GPT-3 (Brown et al., 2020) in zero/one/five-shot setups. We explore a few variations of prompts by tweaking label order or selecting few-shot examples differently, and report the best scores in Table 5. The performance of in-context learning significantly improves when we use more examples in the prompt, but they are far from the performance of `XLGen`. Moreover, the performance gap between GPT-3 and `XLGen` is much larger in EURLEX-4K where labels are formally annotated than in WIKI10-31K where labels are annotated by random users without a solid guideline. Unlike other multi-label classification tasks, XMC treats an extremely large number of labels, making it difficult to predict most unseen labels based on a few examples in in-context learning. See A.3 for the details of the experimental setup for in-context learning and the performance comparison among prompt variations.

		EURLEX-4K		WIKI10-31K	
		Mic.	*Mac.*	*Mic.*	*Mac.*
`XLGen-BCL`		**60.7**	**52.4**	**37.7**	**20.0**
GPT-3	0-shot	9.2	6.3	7.6	4.5
	1-shot	17.2	14.7	20.3	13.4
	5-shot	15.7	10.4	23.5	16.6

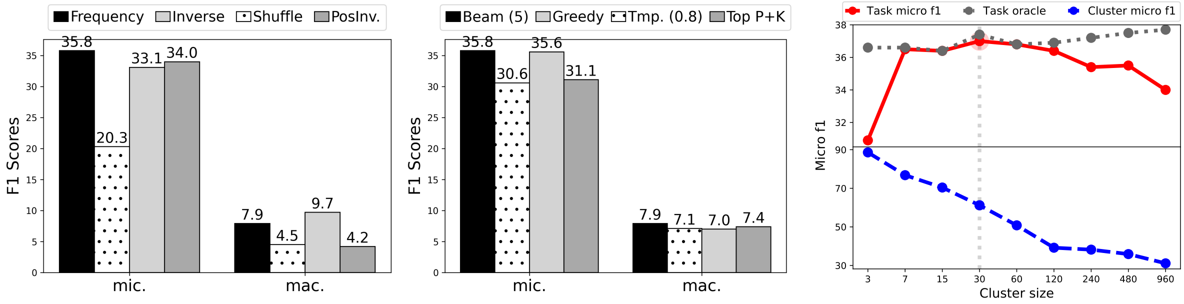
Table 5: Label performance of `XLGen-BCL` and in-context learning settings on 100 randomly selected samples.

5 Ablation Study

We explore various factors that impact the performance of `XLGen` on WIKI10-31K, such as label orders (§5.1) and sampling strategies (§5.2) In order to reduce training costs, we mainly train `XLGen-base` on the base size model with epoch 5 for ablation tests. We then investigate the model performance by clustering sizes and algorithms (§5.3).

5.1 Label Orders

Label orders in decoder are important as `XLGen` sequentially generates labels. We compare three different label orders; label frequencies from high to low (Frequency), inverse label frequencies from low to high (Inverse), and shuffling where labels are randomly ordered per training epoch. Inspired by Lee et al. (2019), we also consider ignoring label orders by resetting positional embeddings of



(a) Label Order (b) Decoding Strategy (c) Cluster Size

Figure 6: Ablation study results on WIKI10-31K. (a) Performances of `XLGen-base` trained with various label orders. (b) Performances of `XLGen-base` trained with various decoding strategies. (c) Cluster sizes vs task and clustering performance. We also report oracle scores by using ground-truth cluster information in inference time.

each label as initial values in decoder [1] which we call label positional invariant setting (PosInv.).

Figure 6a shows task performance across different label orders. We find trade-offs between macro and micro F1 scores by the label frequency order (Frequency and Inverse) because inversely frequent label orders make the model generate long-tail labels earlier with certainty, thus, the scores of long-tail labels could improve. On the other hand, shuffling (Simig et al., 2022) crucially downgrades the performance since with randomly shuffled labels, `XLGen` tends to ignore co-occurrence patterns among labels in training time. Also, we conjecture that positional invariant setting does not work well as it tweaks the original positional embeddings of pre-trained T5 model.

5.2 Decoding Strategy

We now explore task performances with various sampling strategies in label generation. We compare greedy search, beam search, sampling with restrictions such as Top-K (Fan et al., 2018) and Top-P (Holtzman et al., 2020), and sharpening vocabulary distributions with a temperature parameter. In Figure 6b, we find that beam search with size 5 achieves the best scores. Interestingly, most sampling methods heavily degrade performances since our label spaces are not entirely open-ended. We also explore ensemble methods to combine label outputs from different sampling strategies. Unlike the PU setting, however, they are not helpful in the full data setup since a sufficient number of labels are already generated by a single best generation strategy. Find the Appendix A.6 for details.

5.3 Cluster Strategy

We show the effect of clustering algorithms and their parameters. We train `XLGen-MCG` fine-tuning T5-base with epoch 5. We compare two clustering methods; K-means and Agglomerative clustering, and two text representations; TF-IDF and the recent T5 encoder. We find K-means and pre-trained T5 encoder shows the best performance over other combinations, as described in Appendix A.7.

Cluster size is another important factor for model performance. For example, a larger cluster size helps find label groups at a higher granularity, while it is much harder to be accurately predicted in inference time. Here, we choose cluster size to be a power of two on average (e.g., around 30 containing 1024 labels for WIKI10-31K on average). Figure 6c shows micro F1 scores of `XLGen-MCG` across cluster sizes in WIKI10-31K. Here we also report the upper bound of task performance (oracle) by using ground-truth cluster information. As we expect, clustering performance decreases as the cluster size increases since it is much harder to predict clusters in a larger cluster space. In terms of label prediction, we find that the model with smaller cluster sizes (e.g., ≤ 30) outperforms the larger ones, where the peak is around 30. Although a larger cluster size helps elaborately specify labels in the same category, lower cluster prediction performance harms label performances as well and leads to a bigger performance gap compared with oracle task scores.

6 Qualitative Analysis

Lastly, we evaluate the quality of generated labels via human evaluation (§6.1) and visualization of the semantic relations among labels (§6.2).

---

[1] But we keep the position embeddings for token sequences in one label to learn token positions.

		AttentionXML		`XLGen-BCL`	
		#	%	#	%
Existing labels	*Correct*	674	39.2%	596	39.2%
	Wrong	776	45.1%	457	30.0%
	PU	270	15.7%	393	25.8%
New labels	*Correct*	0	0.0%	45	3.0%
	Wrong	0	0.0%	30	2.0%
Total		1,720	100%	1,521	100%

Table 6: Human evaluation on WIKI10-31K having 1,720 true labels. The predicted labels are annotated and categorized to *correct*, *wrong*, and *PU* labels, with their precision scores. Note that *correct* labels in the newly generated labels means they are possibly correct, according to the human annotators' decision.

6.1 Human Evaluation

The benchmark datasets have different label qualities. For example, the labels from EURLEX-4K, annotated by the Publication Office of EU, are refined and structured while Wiki datasets are collaboratively labeled by general users, so the quality of labels is relatively lower than the other benchmarks. Hence, we conduct human evaluations in both quantitative and qualitative ways to accurately measure the potential existence of PU labels and newly-generated labels. In particular, we randomly select 100 instances from the test set and extract incorrectly predicted labels and/or newly predicted labels by `XLGen` and baseline models. We then ask three human annotators to annotate and decide on possibly positive labels via majority voting.

Table 6 shows human evaluation results on the annotated WIKI10-31K. Note that the number of predicted labels by `XLGen-BCL` is less than true labels because `XLGen` does not generate label with low confidence. In AttentionXML, on the other hand, we choose top-K labels as many as the number of true labels for each instance, so it has the same total labels as true labels. Compared to the best baseline, AttentionXML, `XLGen-BCL` could generate more PU labels and reduce the number of wrong labels. Also, our method generates 75 (=45+30) newly generated labels out of 1,521 where 60% (=45/75) of them are correct, showing a relatively good generation quality of new labels. Of course, we can control our model to only count the candidate labels and not any of these new labels for more accurate predictions, as measured in Table 2.

Lastly, we provide annotation examples in Table 7. As we sort the label sequence by frequency in training for `XLGen`, frequently generated labels such as “wikipedia" or “wiki" are predicted first, followed by long-tail labels specified in the input text. For AttentionXML, on the other hand, top predicted labels seem more aligned with the input context, although frequently generated labels still come in front. Interestingly, new labels generated by `XLGen` come not only from the input context, but also previously generated labels. For instance, on the Wikipedia page of diet coke and mentos eruption, a new label “soda" is generated because input text contains “carbonated beverages" which is synonym of “soda". On the Wiki page of Vimeo, on the other hand, after `XLGen` generates the PU label “socialnetworking", followed by its synonyms such as “social_network" and “social_networking".

6.2 Label Semantics in `XLGen`

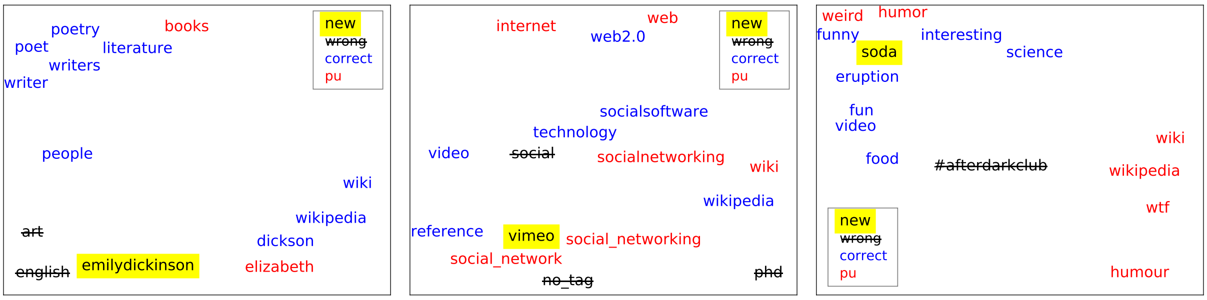
To better understand the semantics behind labels generated by `XLGen`, we visualize an annotated labels of three examples from Table 7 in Figure 7. We get label embeddings from the last hidden state of the fine-tuned `XLGen-BCL` decoder and project them into two-dimensional T-SNE (van der Maaten and Hinton, 2008). If a single label is split by multiple tokens, we average the last hidden layers of all tokens. We observe that frequently co-occurred labels (e.g., “wiki"-“wikipedia" or “weird"-“funny") have similar label embeddings. Also, the newly generated labels become close to the co-occurred labels (e.g., “soda" - “funny" or “eruption" in diet coke and mentos eruptions) via `XLGen` optimization.

7 Conclusion

We apply text-to-text Transformers to extreme multi-label classification, by tweaking the classification problem as generation of label texts. As we do not control the vocabulary space of generated labels, `XLGen` can create completely unseen but still relevant labels, inferred from the input context and semantic relationship from the previously generated labels. Our experiments show that `XLGen` outperforms the classification baselines in general, and significantly improves the long-tail performance and PU setting. Also, we observe utilizing label cluster information helps improve the performance in various settings. `XLGen` is expected to more benefit from pre-trained models as they become larger and powerful (Kaplan et al., 2020).

Input Document	Models	Labels
Emily Elizabeth Dickinson (December 10, 1830– May 15, 1886) was an American poet. Born in Amherst, Massachusetts to a successful family with strong community ties, she lived a mostly introverted and reclusive life. After she studied at the Amherst Academy for seven years in her youth, she spent a short time at ...	True	authors biography dickinson emily journal library literature openaccess people poem poet poetry reference research to-read wiki wikipedia writers
	AttentionXML	wiki poet writers wikipedia literature authors books ~~writing~~ ~~history~~ poets writer people poetry biography ~~inspiration~~ american poems ~~luule~~
	`XLGen-BCL`	wikipedia wiki people ~~art~~ books literature ~~english~~ poetry writers writer poet elizabeth dickinson emilydickinson
Screenshot of vimeo.com home page Vimeo is a video-centric social network site (owned by IAC/InterActiveCorp) which launched in November 2004. The site supports embedding, sharing, video storage, and allows user-commenting on each video page...	True	articles computer reference socialnetworks technology tools video web2.0 wikipedia
	AttentionXML	video web2.0 wikipedia wiki ~~media~~ ~~youtube~~ videos ~~videoblogging~~ ~~streaming~~
	`XLGen-BCL`	wikipedia wiki reference technology web internet ~~social~~ video web2.0 ~~no_tag~~ socialnetworking ~~socialsoftware~~ ~~phd~~ social_networking social_network vimeo
Diet Coke and Mentos Eruption is a reaction of Diet Coke and mint Mentos candies, a bottle of Diet Coke (other carbonated beverages may be used instead) and dropping some Mentos. This causes the Coke to foam at a rapid rate and spew into the air...	True	beverage candy chemistry coca-cola coke dietcoke drink eruption experiment experiments video explosion food fun funny interesting mint prank science
	AttentionXML	wikipedia fun science ~~diet~~ wiki funny coke ~~tv~~ video ~~health~~ interesting humor food
	`XLGen-BCL`	wikipedia wiki science interesting fun video funny food humor weird humour wtf ~~#afterdarkclub~~ soda eruption

Table 7: Ground-truth and predicted labels from `XLGen-BCL` and AttentionXML on input documents in WIKI10-31K. We ask human annotators to annotate labels to be correct (blue), wrong (~~strikethrough~~), and PU (red). For `XLGen`, we additionally mark potentially correct labels from the newly generated labels and their relevant contexts in input text with yellow box. (e.g., a possibly correct label soda is newly generated based on the fact that diet coke can be replaced with other carbonated beverage.)



(a) Emily Dickinson (b) Vimeo.com (c) Diet Coke and Mentos Eruptions

Figure 7: Visualization of generated labels by `XLGen-BCL` for the Wikipedia page examples in Table 7.

Limitations and Future Directions

First, we conduct our main experiments and additional analyses on certain languages such as English that has tremendous text corpora. Extension to the low-resource languages might be challenging since this work requires text2text pre-trained models where those languages are applicable (e.g., multilingual T5 model), as well as the corresponding XMC datasets.

Also, compared to the efficient classification baselines, generative models are relatively expensive in terms of memory and time. For example, our experiment requires a lot of training resource as pre-trained models have >200 millions parameters to be tuned. Thus, we use three p3.16xlarge AWS instances with 8 Nvidia V100 GPUs for training. Using more efficient version of Transformers (Tay et al., 2022) or applying distributed training should be considered for a resource reduction.

While in-context learning does not show comparable performance in XMC, we do observe that as the number of examples increases from zero to one to five, in-context learning can generate reasonable unseen but positive labels. It would be interesting to explore the potential of in-context learning in XMC with more advanced prompting and example sampling in the future.

Lastly, the XMC task has a risk of being biased or overfit to small training datasets (e.g., EURLEX-4K and WIKI10-31K contain only about 15,000 training examples). As with other commonly used NLP benchmarks, there is a potential risk that our proposed method may not work properly in the new test/train sets, though we anticipate that such a risk will be quite small.

Ethics Statement

We use the four XMC benchmark datasets which are publicly available and widely used in research [2]. The datasets with social tags (e.g., WIKI10-31K and WIKI-500K) may contain inappropriate vulgarisms if they are not filtered out from the original data processing.

References

Enrique Amigo and Agustín Delgado. 2022. Evaluating extreme hierarchical multi-label classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5809–5819, Dublin, Ireland. Association for Computational Linguistics.

Rohit Babbar and Bernhard Schölkopf. 2017. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 721–729.

Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S Dhillon. 2020. Taming pre-trained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3163–3171.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *ACL*, pages 889–898.

Eva L. Gibaja. 2015. A tutorial on multi-label learning. *ACM Computing Surveys*, 47:1–38.

N. Gupta, S. Bohra, Y. Prabhu, S. Purohit, and M. Varma. 2021. Generalized zero-shot extreme multi-label learning. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *ICLR*.

Wenpeng Hu, Ran Le, Bing Liu, Feng Ji, Jinwen Ma, Dongyan Zhao, and Rui Yan. 2021. Predictive adversarial learning from positive and unlabeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7806–7814.

Himanshu Jain, Venkatesh Balasubramanian, Bhanu Chunduri, and Manik Varma. 2019. Slice: Scalable linear extreme classifiers trained on 100 million labels for related searches. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 528–536.

Atsushi Kanehira and Tatsuya Harada. 2016. Multi-label ranking from positive and unlabeled data. In *CVPR*, pages 5138–5146.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv:2001.08361*.

Joo-Kyung Kim and Young-Bum Kim. 2020. Pseudo labeling and negative feedback learning for large-scale multi-label domain classification. In *ICASSP*, pages 7964–7968.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, pages 3744–3753.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. *Advances in neural information processing systems*, 30:5413–5423.

Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *Proceedings of the 2018 World Wide Web Conference*, pages 993–1002.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits

---

[2] `https://github.com/yourh/AttentionXML`

of transfer learning with a unified text-to-text transformer. *JMLR*, 21:1–67.

Wissam Siblini, Pascale Kuntz, and Frank Meyer. 2018. CRAFTML, an efficient clustering-based random forest for extreme multi-label learning. In *International Conference on Machine Learning*, pages 4664–4673.

Daniel Simig, Fabio Petroni, Pouya Yanki, Kashyap Popat, Christina Du, Sebastian Riedel, and Majid Yazdani. 2022. Open vocabulary extreme classification using generative models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1561–1583, Dublin, Ireland. Association for Computational Linguistics.

Yukihiro Tagami. 2017. AnnexML: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 455–464.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. 2022. Scale efficiently: Insights from pretraining and fine-tuning transformers. In *ICLR*.

Che-Ping Tsai and Hung-Yi Lee. 2020. Order-free learning alleviating exposure bias in multi-label classification. In *AAAI*, pages 6038–6045.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

Marek Wydmuch, Kalina Jasinska, Mikhail Kuznetsov, Róbert Busa-Fekete, and Krzysztof Dembczynski. 2018. A no-regret generalization of hierarchical softmax to extreme multi-label classification. *Advances in neural information processing systems*, 31:6358–6368.

Yuanhao Xiong, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Inderjit Dhillon. 2022. Extreme Zero-Shot learning for extreme text classification. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5455–5468, Seattle, United States. Association for Computational Linguistics.

Pengcheng Yang, Fuli Luo, Shuming Ma, Junyang Lin, and Xu Sun. 2019. A deep reinforced sequence-to-set model for multi-label classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5252–5258.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: sequence generation model for multi-label classification. In *COLING*, pages 3915–3926.

Hui Ye, Zhiyu Chen, Da-Han Wang, and Brian Davison. 2020. Pretrained generalized autoregressive model with adaptive probabilistic label clusters for extreme multi-label text classification. In *International Conference on Machine Learning*, pages 10809–10819. PMLR.

Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. 2017. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 545–553.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. AttentionXML: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 32:5820–5830.

Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit S. Dhillon. 2014. Large-scale multi-label learning with missing labels. In *ICML*, pages 593–601.

Hsiang-Fu Yu, Kai Zhong, and Inderjit S Dhillon. 2022. Pecos: Prediction for enormous and correlated output spaces. *JMLR*, 23:1–32.

Jiong Zhang, Wei-cheng Chang, Hsiang-fu Yu, and Inderjit Dhillon. 2021a. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 34:7267–7280.

Ximing Zhang, Qian-Wen Zhang, Zhao Yan, Ruifang Liu, and Yunbo Cao. 2021b. Enhancing label correlation feedback in multi-label text classification via multi-task learning. In *ACL Findings*.

Daoming Zong and Shiliang Sun. 2022. BGNN-XML: Bilateral graph neural networks for extreme multi-label text classification. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–12.

A Appendix

A.1 Details on Baseline Models

AttentionXML (You et al., 2019) is a label tree-based deep learning model. It uses a shallow and wide probabilistic label tree which allows to handle millions of labels and a multi-label attention mechanism by using raw text as input to capture the most relevant part of text to each label.

X-Transformer (Chang et al., 2020) is the first scalable approach to apply deep transformer models in XMC task. In particular, it uses a pre-trained transformer encoder to assign labels to corresponding cluster. For each hierarchical cluster level, OVA classifiers are trained by only using sample instances under the same cluster, called teacher forcing negative (TFN) strategy. Unlike AttentionXML which only uses negative sampling, X-Transformer also uses the negative instances positively predicted by the classifier from the previous cluster level, called matcher-aware negatives (MAN). Recently, Zhang et al. (2021a) proposed XR-Transformer to speed up X-Transformer's training time in recursive manner. Thus, we use XR-Transformer instead of X-Transformer for the comparison.

XR-Linear (Yu et al., 2022) has a very similar architecture with XR-Transformer, except that it only uses simple tf-idf text features instead of transformer encoder outputs. For OVA classification, linear matchers recursively solve XMC sub-problem for each hierarchical cluster level.

In order to fit score outputs into [0,1], we apply sigmoid post processor implemented by the authors for XR-Transformer and XR-Linear.

A.2 Details on `XLGen` Training

	`XLGen-BCL`	`XLGen-MCG`
EurLex-4K	80	20
AmznCat-13K	80	20
Wiki10-31K	60	20
Wiki-500K	80	20

Table 8: Optimal cluster sizes for the `XLGen` training.

We finetune the T5-large (EurLex-4K, Wiki10-31K) or the T5-base (AmazonCat-13K, Wiki-500K), with epoch 10 (EurLex-4K, AmazonCat-13K) or epoch 5 (Wiki10-31K, Wiki-500K) based on the data and/or label size.

We set up input length as 500 for all benchmark datasets and use different output length based on the label lengths in train set; 90 for EurLex-4K and 165 for other three benchmarks. We optimize `XLGen` using AdamW (Loshchilov and Hutter, 2019) with learning rate 2e-4.

	Sample	Label	EurLex-4K		Wiki10-31K	
			$Mic.$	$Mac.$	$Mic.$	$Mac.$
0-shot	Random	Random	5.3	3.8	7.1	3.8
	Random	Frequency	**9.2**	**6.3**	**7.6**	**4.5**
	Most Label	Random	6.0	4.3	7.2	4.4
	Most Label	Frequency	5.0	3.5	6.7	3.5
1-shot	Random	Random	14.8	10.7	13.6	11.8
	Random	Frequency	16.1	10.4	**20.3**	13.4
	Most Label	Random	**17.2**	**14.7**	17.9	**16.8**
	Most Label	Frequency	15.1	12.2	18.1	16.1
5-shot	Random	Random	**15.7**	10.4	17.9	14.2
	Random	Frequency	13.1	9.7	**23.5**	16.6
	Most Label	Random	11.0	9.6	19.8	**18.0**
	Most Label	Frequency	12.5	**11.3**	21.5	15.1

Table 9: Micro-averaged and macro-averaged F1 scores in-context learning settings on 100 randomly selected samples. We test two label ordering strategies, random and decreasing label frequency (frequency), as well as two sampling strategies, random and selecting examples with the most labels (most label). The highest scores are **bold**.

For `XLGen-BCL`, we set up an initial weight value λ as 1.0 and reduce it to $\frac{1}{k}$ for every epoch number k.

For cluster-based `XLGen` architectures, we train k-means clustering and optimize the cluster size via cross-validation from the range of {10,20,30,...,100}. In Table 8, we report optimal cluster sizes for `XLGen` training.

Note that each of T5 models have 220 million (T5-base) or 770 million (T5-large) parameters to be tuned. Also for training, we use a small batch size (1) since pre-trained T5 models are large to be fitted in a single GPU machine. Due to the model size, we use two GPU machines via model parallelism for T5-large and a single GPU machine for T5-base in training. Also, due to the training cost and time, we report the performance scores from the single running of training and inference. We basically modify the T5 code from huggingface library [3], and our code will be publicly available at `https://github.com/alexa/xlgen-eacl-2023`.

A.3 In-context learning in XMC

For in-context learning, we use OpenAI GPT-3 text-davinci-002 model with temperature 0.7 and max tokens 256. To find the optimal prompt, we use prompt variations with different label orders

---

[3] `https://huggingface.co/`

	EURLEX-4K	AMZNCAT-13K	WIKI10-31K	WIKI-500K
	F@1/F@3/F@5/F@10	F@1/F@3/F@5/F@10	F@1/F@3/F@5/F@10	F@1/F@3/F@5/F@10
XR-Transformer	**27.4**/47.5/47.2/34.7	**31.6**/55.2/53.5/38.0	**8.8**/20.3/24.4/23.8	24.1/34.2/32.8/25.5
XR-Linear	26.1/47.8/51.1/41.2	30.5/55.8/58.6/45.7	8.5/19.2/24.9/29.3	22.8/32.5/31.3/24.4
AttentionXML	26.9/**51.9**/**58.8**/59.9	30.8/**59.2**/67.0/ 69.9	8.6/**21.0**/**28.1**/**35.6**	**24.4**/42.9/48.9/52.8
`XLGen-base`	21.3/46.9/57.8/59.8	30.8/57.4/65.3/69.0	8.1/15.5/21.7/30.1	23.7/**44.0**/50.4/54.6
`XLGen-BCL`	21.2/47.2/58.7/**60.7**	31.0/57.7/65.5/69.2	8.1/15.4/21.5/30.1	23.8/43.8/50.3/54.6
`XLGen-MCG`	20.9/47.0/58.1/60.2	31.2/58.8/**67.5**/**71.2**	8.1/15.6/22.2/31.2	23.8/**44.0**/**50.5**/**54.8**

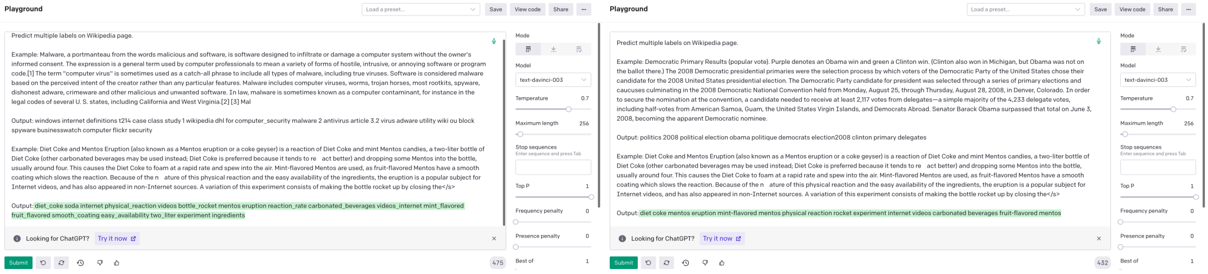
Table 10: Supplementary scores on benchmark datasets. We report ranking-based @k (k=1,3,5,10) F1 scores (F@k) as supplementary metrics. The highest scores are **bold**.

	EURLEX-4K			WIKI10-31K			WIKI-500K		
	0-shot	1-shot	5-shot	0-shot	1-shot	5-shot	0-shot	1-shot	5-shot
XR-Transformer	0.0/0.0	1.5/0.5	4.7/2.3	0.0/0.0	2.5/1.6	2.9/1.7	0.0/0.0	0.0/0.0	0.0/0.0
XR-Linear	0.0/0.0	5.9/1.1	8.6/2.7	0.0/0.0	2.8/2.3	2.9/2.4	0.0/0.0	0.2/0.1	1.4/0.9
AttentionXML	0.0/0.0	16.3/2.4	28.4/8.3	0.0/0.0	0.3/0.2	5.6/1.9	0.0/0.0	2.2/1.3	16.1/9.2
`XLGen`	5.3/3.2	21.4/3.5	34.9/10.8	4.5/2.9	14.4/**8.4**	**17.9**/**7.5**	**21.6**/22.5	35.6/24.1	39.6/28.7
`XLGen-BCL`	**7.3**/4.3	**25.0**/**4.1**	**36.1**/**11.4**	**5.0**/3.3	**14.5**/**8.4**	17.6/7.3	18.6/23.2	36.2/24.8	40.2/29.5
`XLGen-MCG`	**7.3**/**4.5**	16.7/2.7	35.9/11.1	**5.0**/**11.1**	13.9/8.1	17.3/7.2	20.7/**23.7**	**37.0**/**25.5**	**40.6**/**29.9**

Table 11: Task performances on benchmark datasets in full few-shot setup. We use conventional micro-averaged ($Mic.$) and macro-averaged ($Mac.$) F1 scores and mark $Mic.$/$Mac.$ in the table. The highest scores are **bold**.

	EURLEX-4K			WIKI10-31K			WIKI-500K		
	Positive Unlabeled Deficit Ratio			Positive Unlabeled Deficit Ratio			Positive Unlabeled Deficit Ratio		
	20%	50%	80%	20%	50%	80%	20%	50%	80%
XR-Transformer	32.6/10.9	25.9/8.6	14.6/4.7	20.7/3.5	16.7/3.4	12.0/3.3	29.2/8.0	24.5/7.0	17.7/5.1
XR-Linear	38.6/12.1	27.8/8.8	12.5/3.7	16.1/3.9	11.5/2.7	5.1/1.7	14.4/3.7	9.9/2.8	5.9/2.0
AttentionXML	**57.6**/23.6	**52.3**/18.7	**40.0**/11.1	**34.7**/3.3	27.7/1.3	14.1/0.0	50.9/18.9	46.0/11.3	35.5/5.0
`XLGen-base`	55.7/24.4	47.5/18.8	31.3/9.8	33.6/9.1	32.3/7.7	22.7/2.8	**51.1**/**37.3**	**48.7**/31.6	**37.9**/**25.4**
`XLGen-BCL`	56.0/24.4	47.9/19.3	31.4/10.0	33.2/9.1	**33.0**/8.0	23.6/2.9	50.8/37.0	48.5/31.4	37.8/25.2
`XLGen-MCG`	55.5/**27.8**	48.2/**21.2**	32.6/**13.3**	32.4/**11.7**	**33.0**/ **10.1**	**24.0**/**7.9**	50.7/**37.3**	48.5/**32.7**	35.9/24.8

Table 12: Task performances on benchmark datasets in full PU setup. We use conventional micro-averaged ($Mic.$) and macro-averaged ($Mac.$) F1 scores and mark $Mic.$/$Mac.$ in the table. The highest scores are **bold**.



Predict multiple labels on Wikipedia page.

Example: Malware, a portmanteau from the words malicious and software, is software designed to infiltrate or damage a computer system without the owner's informed consent. The expression is a general term used by computer professionals to mean a variety of forms of hostile, intrusive, or annoying software or program code.[1] The term "computer virus" is sometimes used as a catch-all phrase to include all types of malware, including true viruses. Software is considered malware based on the perceived intent of the creator rather than any particular features. Malware includes computer viruses, worms, trojan horses, most rootkits, spyware, dishonest adware, crimeware and other malicious and unwanted software. In law, malware is sometimes known as a computer contaminant, for instance in the legal codes of several U. S. states, including California and West Virginia.[2] [3] Mal

Output: windows internet definitions t214 case class study 1 wikipedia dhl for computer_security malware 2 antivirus article 3.2 virus adware utility wiki ou block spyware businesswatch computer flickr security

Example: Diet Coke and Mentos Eruption (also known as a Mentos eruption or a coke geyser) is a reaction of Diet Coke and mint Mentos candies, a two-liter bottle of Diet Coke (other carbonated beverages may be used instead; Diet Coke is preferred because it tends to re act better) and dropping some Mentos into the bottle, usually around four. This causes the Diet Coke to foam at a rapid rate and spew into the air. Mint-flavored Mentos are used, as fruit-flavored Mentos have a smooth coating which slows the reaction. Because of the n ature of this physical reaction and the easy availability of the ingredients, the eruption is a popular subject for Internet videos, and has also appeared in non-Internet sources. A variation of this experiment consists of making the bottle rocket up by closing the</s>

Output: diet_coke soda internet physical_reaction videos bottle_rocket mentos eruption reaction_rate carbonated_beverages videos_internet mint_flavored fruit_flavored smooth_coating easy_availability two_liter experiment ingredients

Model: text-davinci-003
Temperature: 0.7
Maximum length: 256
Top P: 1
Frequency penalty: 0
Presence penalty: 0
Best of: 1

475

(a) Label frequency order with random sample

Predict multiple labels on Wikipedia page.

Example: Democratic Primary Results (popular vote). Purple denotes an Obama win and green a Clinton win. (Clinton also won in Michigan, but Obama was not on the ballot there.) The 2008 Democratic presidential primaries were the selection process by which voters of the Democratic Party of the United States chose their candidate for the 2008 United States presidential election. The Democratic Party candidate for president was selected through a series of primary elections and caucuses culminating in the 2008 Democratic National Convention held from Monday, August 25, through Thursday, August 28, 2008, in Denver, Colorado. In order to secure the nomination at the convention, a candidate needed to receive at least 2,117 votes from delegates—a simple majority of the 4,233 delegate votes, including half-votes from American Samoa, Guam, the United States Virgin Islands, and Democrats Abroad. Senator Barack Obama surpassed that total on June 3, 2008, becoming the apparent Democratic nominee.

Output: politics 2008 political election obama politique democrats election2008 clinton primary delegates

Example: Diet Coke and Mentos Eruption (also known as a Mentos eruption or a coke geyser) is a reaction of Diet Coke and mint Mentos candies, a two-liter bottle of Diet Coke (other carbonated beverages may be used instead; Diet Coke is preferred because it tends to re act better) and dropping some Mentos into the bottle, usually around four. This causes the Diet Coke to foam at a rapid rate and spew into the air. Mint-flavored Mentos are used, as fruit-flavored Mentos have a smooth coating which slows the reaction. Because of the n ature of this physical reaction and the easy availability of the ingredients, the eruption is a popular subject for Internet videos, and has also appeared in non-Internet sources. A variation of this experiment consists of making the bottle rocket up by closing the</s>

Output: diet coke mentos eruption mint-flavored mentos physical reaction rocket experiment internet videos carbonated beverages fruit-flavored mentos

Model: text-davinci-003
Temperature: 0.7
Maximum length: 256
Top P: 1
Frequency penalty: 0
Presence penalty: 0
Best of: 1

432

(b) Label random order with random sample

Figure 8: Input prompt and generated outputs (green-shaded) for Wikipedia page of Elizabeth Dickinson with 1-shot example.

and few-shot example sampling strategy. Furthermore, we test two label ordering strategies, random and decreasing label frequency, as well as two sampling strategies, random and selecting examples with the most labels. See Figure 8 for GPT-3 prompt input and generated output. Table 9 shows the in-context learning performances across different label ordering and sampling strategies. The best macro-averaged F1 scores for WIKI10-31K are achieved with label frequency ordering with random sampling; however, there is no consistently outperforming strategy for EURLEX-4K.

A.4 Additional Task Performances on Benchmark Datasets

For the full setup, we also report ranking based scores in Table 10. In general, for supplementary metrics (F@k) `XLGen` shows comparable results with baselines except F@1, and F@3 in EurLex-4K and Wiki10-31K. Note that for `XLGen`, we just treat the order of generated labels as a rank, which might **not** be correct since such generated labels should have a equal priority in theory. For this reason, `XLGen` has lower F@k scores with smaller k. However, such score gaps between baselines and `XLGen` decrease as k increases, like EurLex-4K with `XLGen-BCL`, or even `XLGen` achieves higher performances in the larger benchmarks (e.g., F@5 and F@10 in AmznCat-13K and Wiki-500K). Also, full micro/macro F1 scores for tail labels and PU settings are in Table 11 and Table 12, respectively.

A.5 Additional Analyses on Base Model Comparison

`XLGen-base`	EurLex-4K		Wiki10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
T5-base	**58.0**	**23.8**	**35.8**	**7.9**
BART-base	55.6	23.4	35.0	7.7

Table 13: Task performances of `XLGen-base` trained with different pre-trained model architectures on EurLex-4K and Wiki10-31K. The highest scores are **bold**.

For `XLGen`, we can use any pre-trained text-to-text models. We compare task performance of two popular text-to-text models in Table 13; T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) by finetuning `XLGen-base`. In general T5 model outperforms BART, therefore, we use pre-trained T5 architectures for our main experiments.

A.6 Additional Analyses on Decoding Strategy

Followed by Figure 6b, Table 14 shows a task performance across various decoding strategies, including different beam size for beam search and a single sampling restriction.

Additionally, instead of choosing single generation strategy, we can even consider to integrate generation outputs from different generation strategies. For ensemble generations, we choose three single generation strategies; beam search with size 5, Top $K + P$ sampling and sampling with temperature 0.8 to get diverse label sequences. We also consider two different types of joining method; inner join to union all labels and outer join to intersect labels from single generations.

	EurLex-4K		Wiki10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
Greedy	57.5	23.3	35.6	7.0
Beam (3)	**58.0**	23.7	35.7	7.8
Beam (5)	**58.0**	**23.8**	**35.8**	**7.9**
Beam (10)	57.9	**23.8**	35.4	7.8
Tmp. (0.8)	53.7	21.9	30.6	7.1
Top-K (50)	51.7	20.8	28.7	6.8
Top-P (0.9)	53.2	21.1	28.8	6.5
Top $P + K$	53.6	21.5	31.1	7.4

Table 14: Performances of `XLGen-base` trained with different decoding strategies.

`XLGen-base`	EurLex-4K		Wiki10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
Beam (5)	**58.0**	23.8	**35.8**	7.9
Ens. Outer	53.5	**24.5**	31.5	**10.0**
Ens. Inner	54.0	18.9	29.2	4.2

Table 15: Task performance of `XLGen-base` from the best single stratagy (beam search with size 5) and ensemble generations on EurLex-4K and Wiki10-31K. The highest scores are **bold**.

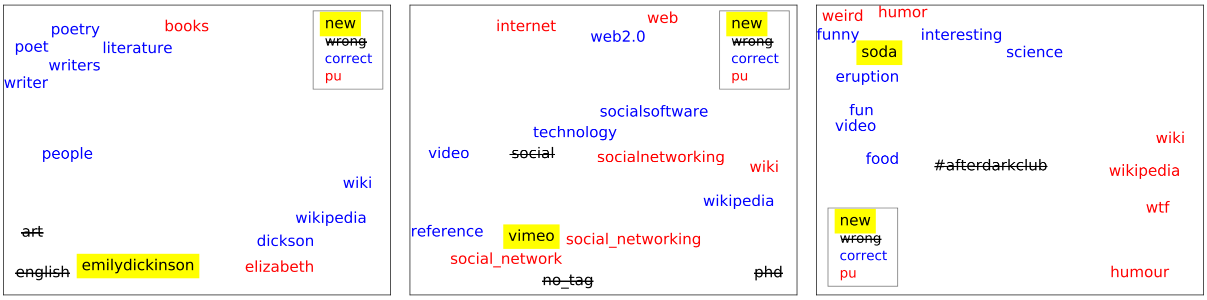
Table 15 shows task performance of ensemble generations. We find that outer joining ensemble generation could improve macro F1 scores as it includes more labels than single result. However, it simultaneously drops other micro F1 scores due to the high chance to contain wrongly predicted labels as well. On the other hand, inner joining ensemble generation in general harms the performance by restricting predicted labels occurring at any single generations, though this yields higher micro F1 scores than inner joining ensemble results.

A.7 Additional Analyses on Clustering and Representation

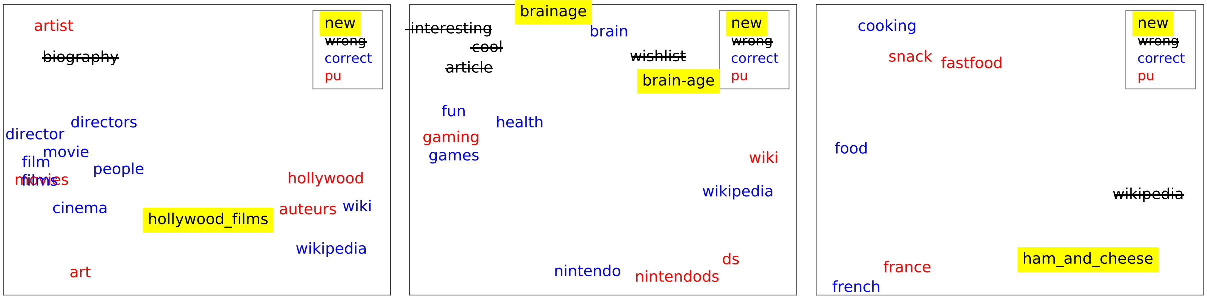
`XLGen-base`	EurLex-4K		Wiki10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
Kmn. + tf-idf	58.4	24.1	36.6	**8.9**
Kmn. + t5-enc.	**58.5**	23.9	36.8	8.8
Ahcl. + tf-idf	58.4	**24.4**	36.8	8.8
Ahcl. + t5-enc	58.0	24.0	**37.0**	8.6

Table 16: Task performances trained with different cluster algorithm and input features on EurLex-4K and Wiki10-31K. Here we fix cluster size as 30. The highest scores are **bold**.

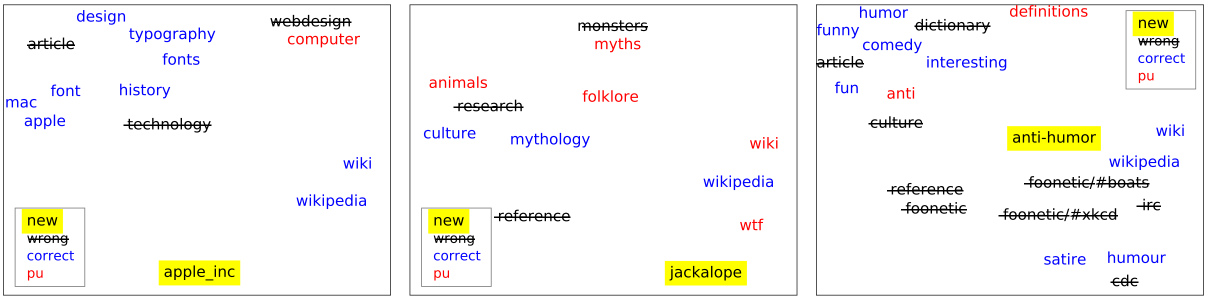
We compare two clustering algorithms; K-means and Agglomerative hierarchical clustering, and two text representations for the label features; TF-IDF and the last hidden states of T5 encoder in Table 16. We find that both algorithms show comparable per-



(a) Emily Dickinson (b) Vimeo.com (c) Diet Coke and Mentos Eruption



(d) David Fincher (e) Brain Age (f) Croque-Monsieur



(g) Typography of Apple (h) Jackalope (i) Anti-humor

Figure 9: Visualization of generated labels by `XLGen-BCL` for Wikipedia pages of annotated examples in Table 17.

formances. As computing cost is more expensive in Agglomerative hierarchical clustering, we mainly use K-means in our experiments. For text representation, the pre-trained T5 encoder achieves similar or slightly better performance to TF-IDF vectors. Pre-trained T5 encoder is more efficient in training as it has much lower size of dimensionality (e.g., 768 in t5-base) than tfidf (e.g., >100,000 for both EUR-Lex and Wiki10-31K). Thus, for all experiments with clustering method, we use K-means with pre-trained T5 encoder text representation.

A.8 Additional Examples of Human Annotation

In Table 17, We provide more annotation examples from Wiki10-31K, following the Table 7 to show how `XLGen` generates labels. In Figure 9, we also provide visualizations of generated labels by `XLGen` for examples in Table 17.

Input Document	**Models**	**Labels**
Emily Elizabeth Dickinson (December 10, 1830– May 15, 1886) was an American poet. Born in Amherst, Massachusetts to a successful family with strong community ties, she lived a mostly introverted and reclusive life. After she studied at the Amherst Academy for seven years in her youth, she spent a short time at ...	True	authors biography dickinson emily journal library literature openaccess people poem poet poetry reference research to-read wiki wikipedia writers
	AttentionXML	wiki poet writers wikipedia literature authors books ~~writing~~ ~~history~~ poets writer people poetry biography ~~inspiration~~ american poems ~~luule~~
	`XLGen-BCL`	wikipedia wiki people ~~art~~ books literature ~~english~~ poetry writers writer poet elizabeth dickinson emilydickinson
Screenshot of vimeo.com home page Vimeo is a video-centric social network site (owned by IAC/InterActiveCorp) which launched in November 2004. The site supports embedding, sharing, video storage, and allows user-commenting on each video page...	True	articles computer reference socialnetworks technology tools video web2.0 wikipedia
	AttentionXML	video web2.0 wikipedia wiki ~~media~~ ~~youtube~~ videos ~~videoblogging~~ ~~streaming~~
	`XLGen-BCL`	wikipedia wiki reference technology web internet ~~social~~ video web2.0 ~~no_tag~~ socialnetworking ~~socialsoftware~~ ~~phd~~ social_networking social_network vimeo
Diet Coke and Mentos Eruption is a reaction of Diet Coke and mint Mentos candies, a bottle of Diet Coke (other carbonated beverages may be used instead) and dropping some Mentos. This causes the Coke to foam at a rapid rate and spew into the air...	True	beverage candy chemistry coca-cola coke dietcoke drink eruption experiment experiments video explosion food fun funny interesting mint prank science
	AttentionXML	wikipedia fun science ~~diet~~ wiki funny coke ~~tv~~ video ~~health~~ interesting humor food
	`XLGen-BCL`	wikipedia wiki science interesting fun video funny food humor weird humour wtf ~~#afterdarkclub~~ soda eruption
David Leo Fincher (born August 28, 1962) is an Academy Award-nominated American filmmaker and music video director known for his dark and stylish movies such as Seven, Fight Club, Zodiac and The Curious Case of Benjamin Button...	True	cinema david director directors figures film filmmaking films fincher inspiration movie people wiki wikipedia
	AttentionXML	wiki directors video wikipedia cinema ~~pitt~~ people director films movies movie film filmmaker ~~brad~~
	`XLGen-BCL`	wikipedia wiki people art film ~~biography~~ movies artist movie cinema films director directors auteurs hollywood hollywood_films
Brain Age: Train Your Brain in Minutes a Day!, also known as Dr. Kawashima's Brain Training: How Old Is Your Brain? in PAL regions, is an entertainment video game that employs puzzles. It was developed and published by the video gaming company Nintendo for the Nintendo DS handheld video game console...	True	@mentat biology brain braintraining computer exercise fitness fun game games health medical nintendo read science sudoku unit4 wikipedia
	AttentionXML	wikipedia game games fun science nintendo sudoku brain ~~mind~~ ds wiki ~~video~~ ~~memory~~ gaming puzzle puzzles videogames nds
	`XLGen-BCL`	games wikipedia fun health brain nintendo nintendods wiki gaming ~~wishlist~~ ~~article~~ ~~interesting~~ ~~cool~~ ds brain-age brainage
A croque-monsieur is a hot ham and cheese (typically emmental[citation needed] or gruyère) grilled sandwich. It originated in France as a fast-food snack served in cafés and bars ...	True	cooking food french recipe sandwich
	AttentionXML	food ~~wikipedia~~ cooking french ~~wiki~~
	`XLGen-BCL`	~~wikipedia~~ food france french cooking ham_and_cheese fastfood snack
Typography of Apple Inc. refers to Apple Inc.'s use of typefaces in marketing, operating systems, and industrial design. Apple has used three corporate fonts throughout its history: Motter Tektura, Apple Garamond and Adobe Myriad. For at least 18 years, Apple's corporate font was a custom variant of the ITC Garamond typeface, called Apple Garamond ...	True	adobe apple branding chronology computer computers design design.fonts fmp font fonts helpful history imac ipod list mac macintosh marketing myriad print pro product reference sda spunti storia typography wiki wikipedia
	AttentionXML	typography fonts apple font design wikipedia type typeface wiki tipografia ~~ttf~~ macintosh reference mac history graphics ~~logo~~ graphic ~~designers~~ ~~webdesign~~ ~~graphicdesign~~ ~~diseño~~ computer typographer ~~ipod~~ ~~brand~~ ~~article~~ ~~technology~~ ~~business~~ ~~advertising~~
	`XLGen-BCL`	wikipedia wiki history ~~article~~ design ~~technology~~ computer ~~webdesign~~ mac apple typography fonts font apple_inc
The jackalope — also called an antelabbit, aunt benny, Wyoming thistled hare or stagbunny — is an imaginary animal of folklore and a supposed cross between a jackrabbit and an antelope, goat, or deer, which is usually ...	True	american animal creatureproject cryptozoology culture fiction humor humour myth mythology storyideas wikipedia
	AttentionXML	folklore wikipedia animals mythology wiki cryptozoology culture ~~monsters~~ animal weird interesting myth
	`XLGen-BCL`	wikipedia wiki ~~reference~~ ~~research~~ culture mythology animals folklore wtf myths ~~monsters~~ jackalope
Anti-humor and anti-jokes[1] (also known as unjokes) are a kind of humor based on the surprise factor of absence of an expected joke or of a punch line in a narration which is set up as a joke. This kind of anticlimax is similar to that of the shaggy dog story.[2] In fact, John Henderson sees the "shaggy dog story" ...	True	comedy favourites fun funny humor humour information interesting jokes people postmodernism wiki wikipedia
	AttentionXML	humor wikipedia funny comedy humour fun satire wiki ~~dog~~ ~~animals~~ ~~standup~~ ~~parody~~ joke
	`XLGen-BCL`	wikipedia wiki ~~reference~~ interesting ~~article~~ ~~culture~~ fun funny humor ~~irc~~ ~~foonetic~~ ~~foonetic/#xkcd~~ definitions ~~dictionary~~ humour comedy satire anti ~~cdc~~ ~~foonetic/#boats~~ anti-humor

Table 17: Additional examples of ground-truth and predicted labels in WIKI10-31K, following Table 7.

Empathy Identification Systems are not Accurately Accounting for Context

Andrew Lee[†] **Jonathan Kummerfeld**[†‡] **Larry An**[†] **Rada Mihalcea**[†]
[†] University of Michigan [‡] University of Sydney
`ajyl@umich.edu`

Abstract

Understanding empathy in text dialogue data is a difficult, yet critical, skill for effective human-machine interaction. In this work, we ask whether systems are making meaningful progress on this challenge. We consider a simple model that checks if an input utterance is similar to a small set of empathetic examples. Crucially, the model does not look at what the utterance is a response to, i.e., the dialogue context. This model performs comparably to prior work on standard benchmarks and even outperforms state-of-the-art models for empathetic rationale extraction by 16.7 points on T-F1 and 4.3 on IOU-F1. This indicates that current systems rely on the *surface form* of the response, rather than whether it is suitable in context. To confirm this, we create examples with dialogue contexts that change the interpretation of the response and show that current systems continue to label utterances as empathetic. We discuss the implications of our findings, including improvements for empathetic benchmarks and how our model can be an informative baseline.

1 Introduction

Empathy is a fundamental phenomenon that allows us to better communicate and relate with others. Studies show that empathy is significantly correlated with counseling treatment outcomes (Moyers and Miller, 2013; Elliott et al., 2018). Computer systems could be improved by adding the ability to understand empathy.

EPITOME (Sharma et al., 2020b) took a step towards understanding empathy in language, introducing tasks such as *empathy identification* and *empathetic rationale extraction*. Models built using EPITOME have been used to build or evaluate empathetic dialogue systems (Sharma et al., 2021; Zheng et al., 2021; Majumder et al., 2022; Kim et al., 2021) or study social effects of empathy (Chen and Xu, 2021).

In this work, we explore whether current models are effectively considering dialogue context (shortened to *context* for the rest of this paper). We show that a simple model that does not consider context can achieve strong results, and that a model from prior work does not change its predictions when we make substantial changes to the context. Together, these results indicate that models are more limited than previously thought.

We introduce an adapted version of *micromodels* (Lee et al., 2021), a simple and explainable approach that combines a set of models, with each model identifying a specific linguistic phenomenon. This approach performs much better than the EPITOME's model on five metrics, comparably on five metrics, and much worse on two. Critically, we achieve this *without any use of context*.

We inspect our model's behavior and find that it can achieve accuracy scores that are as good as or better than the EPITOME model's for empathetic rationale extraction with as few as three seed/training utterances for our model.

We also conduct an experiment to probe EPITOME's behavior. We take utterances from empathetic responses and randomly insert them as part of the response in another context. Despite these insertions mainly being nonsensical and non-empathetic, prior models nearly *always* predict these responses as empathetic, demonstrating that the models rely on the *surface form* of the response rather than contextual understanding.

The authors of EPITOME noted that empathy is *contextual*; a "reaction to an emotional stimulation" or a "deliberate process of understanding and interpreting the experiences" of others.[1] However, current systems do not effectively account for context: they may identify empathetic *style*, but they do not consider whether a response is indeed a *reaction* to feelings and experiences. Future work

---

[1] Section 2.1 of Sharma et al. (2020b)

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1686–1695
May 2-6, 2023 ©2023 Association for Computational Linguistics

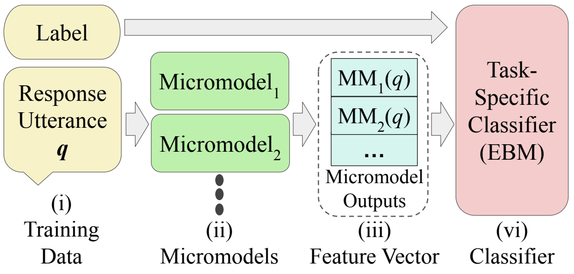


Figure 1: The micromodel framework training process. Notably, our approach does not use any *dialogue context*, yet still performs strongly on current empathy detection benchmarks, raising the question of whether current systems and benchmarks are accounting for context.

should conduct probing experiments like the one we use here, and could consider micromodels as a baseline. This will help assess the contextual understanding of empathy for future models.

2 Benchmarks and Tasks: EPITOME

EPITOME (Sharma et al., 2020b) is a framework for computationally assessing empathy in text-based dialogue. We denote their dataset as EPITOME$_{\text{Data}}$ and the model as EPITOME$_{\text{Model}}$.

EPITOME$_{\text{Data}}$ consists of pairs of dialogues from Talklife and Reddit, and two tasks: empathy prediction (**EmpPred**) and empathetic rationale extraction (**EmpRE**). Given a seeker's post $S_i = s_{i1}, ..., s_{im}$ and a response $R_i = r_{i1}, ..., r_{in}$, each response R_i is annotated with an empathy level (None, Weak, or Strong) in the context of S_i across three communication mechanisms: Emotional Reactions, Interpretations, and Explorations.[2]

Empathetic rationales are spans of text that provide evidence of empathy. They are annotated at the token-level, e.g., the response *"I feel you. Are you okay?"* is represented as $[1, 1, 1, 0, 0, 0]$, $[0, 0, 0, 1, 1, 1]$, and $[0, 0, 0, 0, 0, 0]$ for Emotional Reactions, Exploration, and Interpretations, with one digit per token.

The goal of **EmpPred** is to predict the correct level of empathy given (S_i, R_i) across each communication mechanism. The goal of **EmpRE** is to correctly extract the rationale spans.

3 Micromodels

Lee et al. (2021) introduced the micromodel framework to assess the mental health status of social media users. We give a brief overview of the framework, followed by our adaptations to tackle each task. Further details are provided in the appendix.

3.1 Micromodel Framework

Figure 1 depicts the micromodel framework. The framework consists of a set of micromodels, in which each micromodel MM is a binary classifier that is initialized with a set of *seed* utterances $Z = z_1, ..., z_n$. Given an input query q, micromodel MM gives a binary prediction if q is semantically similar to any of the seed utterances in Z:

$$MM(q) = \exists_{z \in Z} CosSim(BERT(q), BERT(z)) > \theta \tag{1}$$

The outputs of the micromodels are used as features to train a task-specific classifier. Lee et al. (2021) uses explainable boosting machines (EBM) (Caruana et al., 2015), which are generalized additive models (Lou et al., 2012) that make predictions based on adding a set of feature functions learned on each input feature, where each feature function is trained using bagging and gradient boosting.

3.2 Micromodels for EmpPred

For EPITOME's tasks, we build three micromodels, one for each communication mechanism c. For each micromodel MM_c, the seed utterances Z_c are initialized using the annotated rationales of each communication mechanism in the training split of EPITOME$_{\text{Data}}$. Once initialized, rather than using the binary outputs from each micromodel, we use the maximum similarity score between each sentence from the response post $r_{ij} \in R_i$ and each of the seed utterances $z \in Z_c$.

$$MM_c(R_i) = \max_{\substack{r_{ij} \in R_i \\ z \in Z_c}} (Sim(BERT(r_{ij}), BERT(z))) \tag{2}$$

We use the resulting similarity scores as features to train an EBM model[3] to predict the empathy level. We use S-BERT (Reimers and Gurevych, 2019) models[4] to compute similarity scores.

3.3 Micromodels for EmpRE

Figure 2 depicts how we apply micromodels to extract empathetic rationales. Given a response post R_i, we first split it into sentences $r_{i1}, ..., r_{in}$. Each micromodel MM_c runs on each sentence r_{ij}, returning 1 if sentence r_{ij} is semantically similar to any of the seed utterances Z_c and 0 otherwise. This results in a binary vector v_c of length n. Each

---

[2] The definition of each communication mechanism can be found in the appendix.

[3] https://github.com/interpretml/interpret

[4] "paraphrase-xlm-r-multilingual-v1"

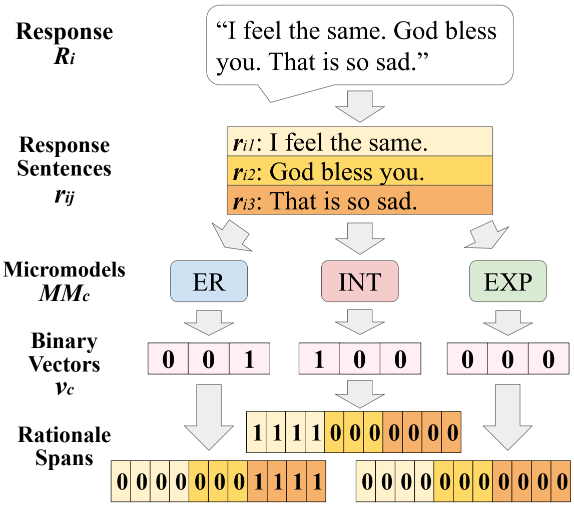


Figure 2: Extracting empathy rationales using micromodels. Each micromodel MM_c determines if each response sentence r_{ij} is empathetic. Each token w_k^{ij} of sentence r_{ij} is then assigned the binary value of r_{ij}.

sentence r_{ij} is then tokenized[5] into a set of tokens $w_1^{ij}, ..., w_l^{ij}$, where $l = len(r_{ij})$ and each token w_k^{ij} is assigned the binary value of r_{ij}. This results in a sequence of 0's and 1's where spans of 1's represent rationales.

4 Experiments and Results

4.1 Experimental Setup

For our experiments, we use random splits of 75:5:20 for our train, validation, and test sets.[6] We report average scores from 10 runs. For θ (Eq. 1), we use a threshold value of 0.7, based on experiments from the validation set. Following EPITOME's authors, we report token-level F1 (T-F1) and Intersection Over Union F1 (IOU-F1) scores.[7]

4.2 Baselines: EPITOME$_{\text{Model}}$

We compare our approach to several baselines, including EPITOME$_{\text{Model}}$. EPITOME$_{\text{Model}}$ is a multitask bi-encoder model initialized with the weights of RoBERTa. Each encoder encodes the seeker's post S_i and the response post R_i. An attention layer attends over both encodings, which is then jointly trained on the two EPITOME tasks. Further details are provided in the appendix.

4.3 Baselines: Other

We also include baseline results as reported by the authors of EPITOME. These baseline models include popular models used in similar tasks, each of which have been fine-tuned or trained on the EPITOME tasks:

- Logistic regression over tf-idf vectors
- Recurrent neural network
- Hierarchical recurrent encoder-decoder (HRED, Sordoni et al. (2015))
- BERT (Devlin et al., 2019)
- GPT-2 (Radford et al.)
- DialoGPT (GPT-2 adapted for dialogue, Zhang et al. (2020))
- RoBERTa (Liu et al., 2019)

4.4 Results

EmpPred$_{\text{EPIT}}$. The first six columns of Table 1 show the accuracy and F-1 scores of empathy prediction. While our F-1 scores are lower than EPITOME$_{\text{Model}}$, they often outperform other fine-tuned language models.

EmpRE. The last six columns of Table 1 show the T-F1 and IOU-F1 scores for empathetic rationale extraction. Other than for Interpretations, we demonstrate significant improvements of up to 16.7 points for T-F1 and 4.3 points for IOU-F1, resulting in the highest scores to our knowledge.

4.5 Follow-Up Analyses

Probing our model: As a post-analysis, we examine which seed utterances $z \in Z_c$ trigger each micromodel during testing and observe that only a small subset of seed utterances are meaningfully used. To demonstrate this, we conduct an experiment in which we iteratively reduce the number of seed utterances used by each micromodel based on how frequently they trigger a micromodel during testing. Figure 3 shows the resulting IOU-F1 scores[8] from 10 random runs, demonstrating that for some communication mechanisms, state-of-the-art results can be achieved by simply checking for semantic similarity against as few as three seed utterances, which are shown in Table 2. Note, this analysis requires modifying the training based on the test performance, so the results are not necessarily representative beyond that dataset.

Probing EPITOME$_{\text{Model}}$: We run an experiment to study whether these utterances are also driving EPITOME$_{\text{Model}}$'s behavior. We gather 1,000

---

[5] We use NLTK for tokenization

[6] This is the same ratio used in the original EPITOME paper. There were no official splits.

[7] We use the same IOU match threshold as EPITOME (0.5).

[8] T-F1 showed nearly identical patterns

Model	Empathy Prediction						Empathetic Rationale Extraction					
	Emotional Reactions		Interpretations		Explorations		Emotional Reactions		Interpretations		Explorations	
	Acc.	F-1	Acc.	F-1	Acc.	F-1	T-F1	IOU-F1	T-F1	IOU-F1	T-F1	IOU-F1
Majority	66.38	26.60	54.58	23.54	83.90	30.41	66.98	66.98	54.94	54.94	84.53	84.53
Log. Reg.	41.69	42.69	70.58	49.77	67.08	46.63	43.26	61.27	49.85	31.31	48.21	70.36
RNN	71.63	42.85	76.21	51.76	85.58	30.74	45.54	43.94	48.22	51.35	65.11	78.27
HRED	71.11	44.10	79.65	54.16	85.58	30.74	46.34	45.65	48.88	52.12	66.66	80.33
BERT	72.13	50.41	82.16	61.20	89.35	56.54	51.06	54.81	48.38	50.75	67.91	71.00
GPT-2	76.69	71.65	82.32	62.27	88.25	58.28	51.44	57.10	54.53	52.38	73.39	82.89
DialoGPT	66.07	51.16	81.85	**68.95**	89.65	70.65	51.83	49.37	54.43	55.85	73.43	85.20
RoBERTa	76.99	70.35	82.16	61.38	90.58	63.41	51.89	58.31	55.62	54.60	69.76	83.33
EPITOME$_{\text{Model}}$	79.43	**74.46**	84.04	62.60	92.61	**72.58**	53.57	64.83	**57.40**	**55.90**	71.56	84.48
Micromodels	**88.26**	59.52	**92.71**	62.73	**95.27**	61.47	**70.30**	**69.13**	54.94	54.08	**86.92**	**86.64**

Table 1: Performance on empathy prediction and empathetic rationale extraction.

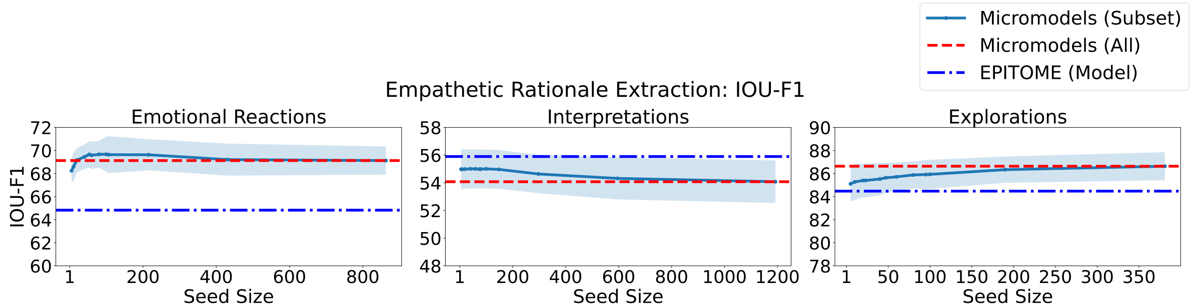


Figure 3: F1 scores with varying seed sizes per micromodel. Simply checking for semantic similarity with as few as three utterances in Table 2 demonstrates either better or competitive performance scores compared to previous state-of-the-art models. The shaded light blue regions indicate the standard deviation across our 10 runs.

Communication Mechanism	Seed Utterance
Emotional Reactions	"I know how you feel." "I'm sorry." "I feel you."
Interpretations	"I feel the same way." "I know how you feel." "I understand how you feel."
Explorations	"Why?" "What happened?" "Why do you feel like that?"

Table 2: Utterances from the smallest subset of seed data used in Figure 3. Simply checking for semantic similarity between response utterances and these seed utterances may outperforms prior state-of-the-art models in empathetic rationale extraction.

Is Empathetic?	PersonaChat		Ubuntu	
	No	Yes	No	Yes
Emotional Reactions	3	997	8	992
Interpretations	305	695	556	444
Interpretations†	10	990	11	989
Explorations	31	969	6	994

Table 3: Number of times non-empathetic dialogues are predicted as empathetic by EPITOME$_{\text{Model}}$. †indicates when we always insert "I feel the same way" (The most commonly seen seed utterance for Interpretations – see Table 2).

random conversations from PersonaChat (Zhang et al., 2018), an open-domain dialogue dataset, and Ubuntu Dialogue (Lowe et al., 2015), a dialogue dataset around technical support for Ubuntu-related problems. For each sample, we insert one of the three utterances from Table 2 as part of the response, resulting in a nonsensical and non-empathetic response. Table 3 shows the results, with EPITOME$_{\text{Model}}$ almost always predicting these artificial dialogues as empathetic. This indicates that it relies on the *surface form* of the response for its prediction, regardless of context.

5 Related Work

Limited access to treatment for mental health, along with a rise in demand for scalable yet high-quality interventions (Miner et al., 2019), has led to an abundance of conversational systems that provide mental health support (Shen et al., 2020; Welch et al., 2020; Han et al., 2013).[9][10] A critical capability for these systems is to understand and interact with empathy, as studies show that

---

[9] https://woebothealth.com/
[10] https://www.wysa.io/

empathy is significantly correlated with counseling treatment outcomes (Moyers and Miller, 2013; Elliott et al., 2018).

Broadly, recognizing empathy within dialogue has been studied under the following contexts: online platforms (Sharma et al., 2020a, 2021; Khanpour et al., 2017), formal counseling settings (Gibson et al., 2015; Zhang and Danescu-Niculescu-Mizil, 2020; Pérez-Rosas et al., 2017), or social media interactions (Hosseini and Caragea, 2021; Lahnala et al., 2021; Wang and Jurgens, 2018; Zhou and Jurgens, 2020).

For instance, Lahnala et al. (2021) examined the interactions between practitioners and non-practitioners that provide support on Reddit. Wang and Jurgens (2018) and Zhou and Jurgens (2020) analyzed the language of condolence and empathy in various social platforms. Other settings for assessing empathy include reacations to news stories (Buechel et al., 2018).

Another direction in the study of empathy within dialogue includes empathetic response generation (Rashkin et al., 2019; Liu et al., 2021; Zhong et al., 2020; Zheng et al., 2021). In order to assess the empathy level of their systems, researchers often use models such as EPITOME. (Sharma et al., 2021; Zheng et al., 2021; Majumder et al., 2022; Kim et al., 2021). Our work demonstrates the pitfalls that researchers should be aware of when taking such approach.

Other efforts include a taxonomy of empathetic responses (Welivita and Pu, 2020), fine-tuned language models for empathy (Guda et al., 2021), as well as empathy-lexicons (Sedoc et al., 2020).

6 Conclusion

In this paper, we assessed whether empathetic systems are correctly taking dialogue context into account. We demonstrated that a simple model with no contextual understanding can achieve results comparable to the EPITOME model and better than all baselines. We find that these results can be achieved by simply checking for semantic similarity to just three utterances. We also found that EPITOME$_{\text{Model}}$ nearly always classifies a response as empathetic regardless of context, as long as it contains one of these three utterances. We conclude that current empathy recognition models do not effectively take contextual information into account.

Future work on benchmarks should consider including examples that require contextual understanding to answer (S: *"I got promoted!"* R: *"That's terrible, I'm sorry."*). Work on models should consider comparing with Micromodels, a simple and practical baseline that serves as a reference point for performance without contextual understanding. These changes will better inform progress on models that meaningfully capture empathy.

The code for our experiments is publicly available at `https://github.com/MichiganNLP/micromodels`.

7 Limitations

The main focus of our paper is on the limitations of empathy recognition models, both our micromodel approach as well as the prior state-of-the-art model EPITOME$_{\text{Model}}$. Namely, our micromodel approach is based on semantic similarity matching, and lacks any representation learning and contextual knowledge. On the other hand, our experiments demonstrate that EPITOME$_{\text{Model}}$ also does not account for context. Despite given non-empathetic contexts, it continues to predict a response as empathetic as long as the *style* of empathy is present. Other limitations of our work include the scope of our study, as we only examine a single benchmark. This is due to the lack of available resources regarding the task of empathy recognition in dialogue. Datasets like EmpatheticDialogue (Rashkin et al., 2019) consists of empathetic conversations, but do not measure the empathy level of utterances. Other empathy prediction tasks (Hosseini and Caragea, 2021; Buechel et al., 2018) do not pertain to dialogue. With more benchmarks regarding empathy recognition in dialogue available, a more thorough study should be conducted.

8 Ethical Considerations

It is important to distinguish our improved accuracy scores from the ability to computationally understand empathy. Because of the lack of representational learning or contextual knowledge, our approach would undoubtedly fail in distinguishing empathetic utterances from false-positive cases, such as sarcastic or even offensive statements (R: *"What's the matter?"* versus *"What's the matter with you?"*). Given the sensitive nature of the mental health domain, mishandling these situations can exacerbate one's situation. Another risk of relying too heavily on such accuracy numbers include overlooking the degree to which a mishandling of a situation can affect an afflicted user. Measuring this additional personal and humanistic dimension in benchmarks for computational systems is undeniably a difficult problem, but likely a necessary step to bridge the gap for effective systems for mental health support. Lastly, while we are able to do a thorough analysis of our findings with the explanations provided by micromodels, there are still portions of their decision making process that remain opaque. Concretely, the computation of semantic similarity by large pre-trained language models like BERT is a key step in our procedure. Using a simpler, more transparent representation for micromodels may mitigate this problem. We believe there is an interesting trade off between accuracy and explainability in designing each micromodel.

Acknowledgments

This material is based in part upon work supported by the John Templeton Foundation (#62256) and by a Berman Research award from the University of Michigan. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the John Templeton Foundation or the University of Michigan.

References

Sven Buechel, Anneke Buffone, Barry Slaff, Lyle Ungar, and João Sedoc. 2018. Modeling empathy and distress in reaction to news stories. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4758–4765, Brussels, Belgium. Association for Computational Linguistics.

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, page 1721–1730, New York, NY, USA.

Yixin Chen and Yang Xu. 2021. Exploring the effect of social support and empathy on user engagement in online mental health communities. *International Journal of Environmental Research and Public Health*, 18(13):6855.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Robert Elliott, Arthur C Bohart, Jeanne C Watson, and David Murphy. 2018. Therapist empathy and client outcome: An updated meta-analysis. *Psychotherapy*, 55(4):399.

James Gibson, Nikolaos Malandrakis, Francisco Romero, David C Atkins, and Shrikanth S Narayanan. 2015. Predicting therapist empathy in motivational interviews using language features inspired by psycholinguistic norms. In *Sixteenth annual conference of the international speech communication association*.

Bhanu Prakash Reddy Guda, Aparna Garimella, and Niyati Chhaya. 2021. EmpathBERT: A BERT-based framework for demographic-aware empathy prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3072–3079, Online. Association for Computational Linguistics.

Sangdo Han, Kyusong Lee, Donghyeon Lee, and Gary Geunbae Lee. 2013. Counseling dialog system with 5W1H extraction. In *Proceedings of the SIGDIAL 2013 Conference*, pages 349–353, Metz, France. Association for Computational Linguistics.

Mahshid Hosseini and Cornelia Caragea. 2021. Distilling knowledge for empathy detection. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3713–3724, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hamed Khanpour, Cornelia Caragea, and Prakhar Biyani. 2017. Identifying empathetic messages in online health communities. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 246–251, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Hyunwoo Kim, Byeongchang Kim, and Gunhee Kim. 2021. Perspective-taking and pragmatics for generating empathetic responses focused on emotion causes. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2227–2240, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Allison Lahnala, Yuntian Zhao, Charles Welch, Jonathan K. Kummerfeld, Lawrence C An, Kenneth Resnicow, Rada Mihalcea, and Verónica Pérez-Rosas. 2021. Exploring self-identified counseling expertise in online support forums. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4467–4480, Online. Association for Computational Linguistics.

Andrew Lee, Jonathan K. Kummerfeld, Larry An, and Rada Mihalcea. 2021. Micromodels for efficient, explainable, and reusable systems: A case study on mental health. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4257–4272, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Siyang Liu, Chujie Zheng, Orianna Demasi, Sahand Sabour, Yu Li, Zhou Yu, Yong Jiang, and Minlie Huang. 2021. Towards emotional support dialog systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3469–3483, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yin Lou, Rich Caruana, and Johannes Gehrke. 2012. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Navonil Majumder, Deepanway Ghosal, Devamanyu Hazarika, Alexander F. Gelbukh, Rada Mihalcea, and Soujanya Poria. 2022. Exemplars-guided empathetic response generation controlled by the elements of human communication. *IEEE Access*, 10:77176–77190.

Adam S Miner, Nigam Shah, Kim D Bullock, Bruce A Arnow, Jeremy Bailenson, and Jeff Hancock. 2019. Key considerations for incorporating conversational ai in psychotherapy. *Frontiers in psychiatry*, 10:746.

Theresa B Moyers and William R Miller. 2013. Is low therapist empathy toxic? *Psychology of Addictive Behaviors*, 27(3):878.

Verónica Pérez-Rosas, Rada Mihalcea, Kenneth Resnicow, Satinder Singh, and Lawrence An. 2017. Understanding and predicting empathic behavior in counseling therapy. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1435, Vancouver, Canada. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

João Sedoc, Sven Buechel, Yehonathan Nachmany, Anneke Buffone, and Lyle Ungar. 2020. Learning word ratings for empathy and distress from document-level user responses. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1664–1673, Marseille, France. European Language Resources Association.

Ashish Sharma, Monojit Choudhury, Tim Althoff, and Amit Sharma. 2020a. Engagement patterns of peer-to-peer interactions on mental health platforms. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 614–625.

Ashish Sharma, Inna W Lin, Adam S Miner, David C Atkins, and Tim Althoff. 2021. Towards facilitating empathic conversations in online mental health support: A reinforcement learning approach. In *Proceedings of the Web Conference 2021*, pages 194–205.

Ashish Sharma, Adam Miner, David Atkins, and Tim Althoff. 2020b. A computational approach to understanding empathy expressed in text-based mental health support. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5263–5276.

Siqi Shen, Charles Welch, Rada Mihalcea, and Verónica Pérez-Rosas. 2020. Counseling-style reflection generation using generative pretrained transformers with augmented context. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 10–20, 1st virtual meeting. Association for Computational Linguistics.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *proceedings of the 24th ACM international on conference on information and knowledge management*, pages 553–562.

Zijian Wang and David Jurgens. 2018. It's going to be okay: Measuring access to support in online communities. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 33–45, Brussels, Belgium. Association for Computational Linguistics.

Charles Welch, Allison Lahnala, Veronica Perez-Rosas, Siqi Shen, Sarah Seraj, Larry An, Kenneth Resnicow, James Pennebaker, and Rada Mihalcea. 2020. Expressive interviewing: A conversational system for coping with COVID-19. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP*

2020, Online. Association for Computational Linguistics.

Anuradha Welivita and Pearl Pu. 2020. A taxonomy of empathetic response intents in human social conversations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4886–4899, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Justine Zhang and Cristian Danescu-Niculescu-Mizil. 2020. Balancing objectives in counseling conversations: Advancing forwards or looking backwards. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5276–5289, Online. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Chujie Zheng, Yong Liu, Wei Chen, Yongcai Leng, and Minlie Huang. 2021. CoMAE: A multi-factor hierarchical framework for empathetic response generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 813–824, Online. Association for Computational Linguistics.

Peixiang Zhong, Chen Zhang, Hao Wang, Yong Liu, and Chunyan Miao. 2020. Towards persona-based empathetic conversational models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6556–6566.

Naitian Zhou and David Jurgens. 2020. Condolence and empathy in online communities. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 609–626, Online. Association for Computational Linguistics.

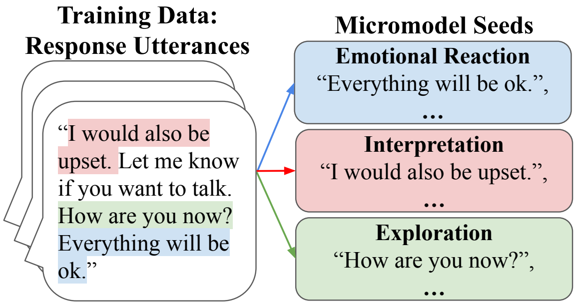


Figure 4: The annotated rationales in the training split of $\text{EPITOME}_{\text{Data}}$ are used as the seed data for each micromodel.

A Empathetic Communication Mechanisms

The EPITOME (Sharma et al., 2020b) framework introduces three communication mechanisms to capture the multi-dimensionality of empathy in text-based dialogue – *Emotional Reactions*, *Interpretations*, and *Explorations*. The definitions and examples of each communication mechanism according to the original EPITOME paper can be found below.

Emotional Reactions Expressions of emotions such as warmth, compassion as a response to the seeker's post. These expressions can explicitly label an emotion (e.g., *"I feel really sad for you."*) or may allude to an emotion (e.g., *"Everything will be fine."*).

Interpretations A reactive statement of one's own understanding of feelings and experiences inferred from the seeker's post. Such statement may simply state their understanding (e.g., *"I understand how you feel."*) or specify their inferred feelings or similar experiences (e.g., *"This must be terrifying.", "I also have anxiety attacks at times which makes me really terrified."*).

Explorations Seeking further information to improve one's understanding of the seeker and their feelings and experiences. These can include generic follow-ups (e.g., *"what happened?"*) or specific inquiries (e.g., *"Are you feeling alone right now?"*).

B Detailed Explanation of Micromodels

Micromodels (Lee et al., 2021) were originally inspired by recent work in microservice architectures, in which complex web applications are built by orchestrating a collection of loosely coupled *microservices*. Each of these microservices has a fine-grained focus of responsibility. In a similar manner, the micromodel framework consists of multiple *micromodels*, in which each micromodel is responsible for representing or identifying a specific linguistic phenomena. In this work we build a micromodel for each communication mechanism.

Here we describe the training procedure using micromodels.

The first step in the framework is to initialize each micromodel. The original authors scrape Reddit and use BERT to look for utterances that are representative of each micromodel. In this work, we simply use the annotated rationales that are available in the training split of EPITOME (see Figure 4). Each micromodel only needs to be initialized once.

Next, given supervised training data in the form of (x, y), each micromodel MM runs on the input query x. While any algorithm of choice can be used for micromodels, the original authors use binary classifiers based on semantic similarity. Concretely, each micromodel that is initialized with a set of seed utterances $Z = z_1, ..., z_n$ makes a binary decision, returning 1 if its input query q is semantically similar to any of the seed utterances $z \in Z$:

$$MM(x) = \exists_{z \in Z} CosSim(BERT(x), BERT(z)) > \theta \tag{3}$$

In this work we use our validation set to determine the θ value.

Once every micromodel runs on x, we are left with a binary vector $\mathbf{v}$ of size n where n is the number of micromodels that were used. The binary vector $\mathbf{v}$ serves as a feature vector for a task-specific classifier to train off of. One can think of the inference from the micromodels to be a featurization step to convert the input data (x, y) into a feature vector $(\mathbf{v}, y)$, while the task-specific classifier is an independent classification model that actually learns a task from such featurized values.

Given $(\mathbf{v}, y)$, a task-specific classifier can be trained. Similar to micromodels, the framework allows for any algorithm of choice to be used for task-specific classification, from simple regression models to complex neural networks. The original authors use explainable boosting machines (EBM) (Caruana et al., 2015) because of the explanations it provides. More specifically, the use of EBMs allows one to understand the impact that each micromodel had on the task-specific classifier's decision making process. For more details on EBMs, we point our readers to both the original paper as well as its widely used open-source repository (https://github.com/interpretml/interpret).

C Detailed Explanation of EPITOME$_{\text{Model}}$

Epitome$_{\text{Model}}$ is a multi-task bi-encoder model in which the two encoders are initialized with the weights of RoBERTa and pre-trained with in-domain data that was available to the authors of EPITOME. The two encoders then each encode the seeker's post S_i and the response post R_i.

$$e_i^{(S)} = \text{S-Encoder}(S_i); e_i^{(R)} = \text{R-Encoder}(R_i) \quad (4)$$

Borrowing terminology from transformers, the response post encoding is used as a query and the seeker post is used as keys and values.

$$a_i(e_i^{(R)}, e_i^{(S)}) = \text{softmax}(e_i^{(R)} e_i^{(S)} / \sqrt{d}) e_i^{(S)} \quad (5)$$

The encoded response $e_i^{(R)}$ is summed with the output of the attention layer $a_i(e_i^{(R)}$ to obtain a residual mapping, resulting in a seeker-context aware representation of the response post, which is used to jointly train on the two tasks.

Enhancing Multi-Document Summarization with Cross-Document Graph-based Information Extraction

Zixuan Zhang[1], **Heba Elfardy**[2], **Markus Dreyer**[2], **Kevin Small**[2], **Heng Ji**[2], **Mohit Bansal**[2]
[1]University of Illinois at Urbana-Champaign
[2]Amazon Alexa
`zixuanl1@illinois.edu`
`{helfardy, mddreyer, smakevin, jihj, mobansal}@amazon.com`

Abstract

Information extraction (IE) and summarization are closely related, both tasked with presenting a subset of the information contained in a natural language text. However, while IE extracts structural representations, summarization aims to abstract the most salient information into a generated text summary – thus potentially encountering the technical limitations of current text generation methods (e.g., hallucination). To mitigate this risk, this work uses structured IE graphs to enhance the abstractive summarization task. Specifically, we focus on improving Multi-Document Summarization (MDS) performance by using cross-document IE output, incorporating two novel components: (1) the use of auxiliary entity and event recognition systems to focus the summary generation model and; (2) incorporating an alignment loss between IE nodes and their text spans to reduce inconsistencies between the IE graphs and text representations. Operationally, both the IE nodes and corresponding text spans are projected into the same embedding space and pairwise distance is minimized. Experimental results on multiple MDS benchmarks show that summaries generated by our model are more factually consistent with the source documents than baseline models while maintaining the same level of abstractiveness.[1,2]

1 Introduction

Information extraction (IE) (Lin et al., 2020; Li et al., 2021) and summarization (Xiao et al., 2022; Pasunuru et al., 2021) are inherently similar tasks, sharing the objective of identifying and presenting a targeted subset of the information present in a natural language text. However, there are also conceptual and methodological distinctions. First of all, IE aims to extract specific structured information from natural language text while abstractive text summarization targets abstracting the most salient information of a given text into a natural language summary. Secondly, IE methods frequently have access to world knowledge via external schema and knowledge resources (e.g., Wikidata) whereas summarization methods often rely on the information encoded in large-scale pretrained embeddings to produce coherent summaries. The complementary aspects of these tasks imply an opportunity to transfer knowledge from one task to another.

Hence, in this paper, our primary motivation is to take advantage of the complementary nature of IE and summarization tasks, using the structured output of entity and event extraction systems to improve abstractive text summarization by focusing text generation toward explicitly observable grounded concepts. There are a few previous research studies exploring the mutual enhancement between IE and summarization. For example, Lu et al. (2022) use text summarization to improve relation extraction and Pasunuru et al. (2021) adopt open-domain IE to provide additional structural inputs for Multi-Document Summarization (MDS). However, these approaches have two notable limitations. First, IE is performed on single documents without analyzing cross-document interactions between the extracted knowledge elements. Such cross-document interactions could be essential to identifying salient parts of the source documents, which is especially useful for MDS. Moreover, previous studies use linearized graphs without actually constructing the graph as a whole, and hence failing to capture some global interactions between the extracted knowledge elements.

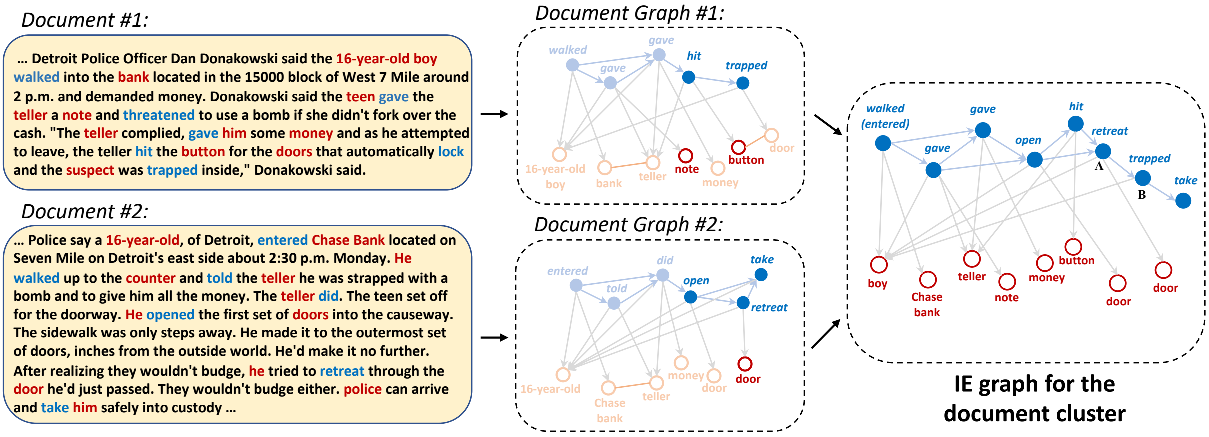
Based on these motivations, in this paper, we propose a text summarization model which is enhanced by IE. We focus on multi-document summarization (MDS) and improve the MDS model with cross-document IE graphs. Specifically, given a cluster of documents related to the same topic, we

---

[1] All our code will be publicly available at `https://github.com/amazon-science/IESum`.

[2] The work was done during the first author's internship at Amazon Alexa.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1696–1707
May 2-6, 2023



… Detroit Police Officer Dan Donakowski said the 16-year-old boy walked into the bank located in the 15000 block of West 7 Mile around 2 p.m. and demanded money. Donakowski said the teen gave the teller a note and threatened to use a bomb if she didn't fork over the cash. "The teller complied, gave him some money and as he attempted to leave, the teller hit the button for the doors that automatically lock and the suspect was trapped inside," Donakowski said.

… Police say a 16-year-old, of Detroit, entered Chase Bank located on Seven Mile on Detroit's east side about 2:30 p.m. Monday. He walked up to the counter and told the teller he was strapped with a bomb and to give him all the money. The teller did. The teen set off for the doorway. He opened the first set of doors into the causeway. The sidewalk was only steps away. He made it to the outermost set of doors, inches from the outside world. He'd make it no further. After realizing they wouldn't budge, he tried to retreat through the door he'd just passed. They wouldn't budge either. police can arrive and take him safely into custody …

Figure 1: An example of an extracted cross-document IE graph. There are two documents in the cluster and both of them describe a story where a boy failed to rob a bank. However, each document lists different details about the event. For example, the first document mentions that the bank teller hit the button to lock the boy inside; while the second document mentions that boy was trapped between two different doors of the bank. We highlight the unique nodes and edges in the document graph. The merged graph has a more comprehensive description of the story.

first use a cross-document fine-grained IE system to extract a cluster-level information graph, where each node could be an entity or an event trigger and each edge could be "*event-event*" temporal relations, "*event-argument*" links, or "*entity-entity*" relations. Each node in the graph is merged from separate documents according to entity and event coreference. After obtaining the cluster-level IE graph, we use an edge-conditioned graph attention network to encode the IE graph and to merge the graph information into the sequence-to-sequence summary generation pipeline. To better utilize the signals from IE, we further propose two novel training objectives. First, we propose an auxiliary task of entity and event recognition, where an additional classification module is incorporated to train a model to select the important entities and event triggers when performing summarization. The purpose of this auxiliary task is to help the model better recognize and remember the important events and entities which could be crucial for generating high-quality summaries. Second, we propose a graph and text alignment loss that minimizes the distance between IE graph nodes (e.g., nodes A and B in Figure 1) and their corresponding text segments (e.g., *retreat* and *trapped*) in a shared latent embedding space. Such an alignment loss can effectively incorporate IE graph information into the text representations and also mitigate the errors and inconsistencies caused by inevitable noise in the automatically extracted IE graphs. We conduct extensive experiments on multiple MDS benchmarks and show that our model outperforms several strong baselines both in terms of ROUGE scores as well as factual consistency metrics, all while maintaining the same level of abstractiveness. In summary, our main contributions are:

- We improve multi-document summarization (MDS) with cross-document IE graphs.
- We propose two novel training objectives to help the model better utilize the guidance from IE: (1) an entity and event recognition task loss and (2) a node-text alignment loss.
- Our proposed approach is proven effective by extensive experiments on multiple MDS benchmarks while achieving new state-of-the-art performance.

2 Problem Formulation

Our problem definition follows the typical formulation of abstractive multi-document summarization (MDS). Specifically, given a cluster of input documents $\mathcal{D} = \{D_1, D_2, \cdots, D_N\}$, we aim to build a model to generate a summary S of the document cluster. In this paper, we particularly focus on using IE to enhance summarization using the IE graph $\mathcal{G}$ merged from the individual graphs $\{G_1, G_2, \cdots, G_N\}$ extracted from N documents.

2.1 Cross-Document Information Extraction

We first perform cross-document information extraction on each document cluster using a state-of-the-art entity extraction and disambiguation system ReFinED (Ayoola et al., 2022) and event extraction and tracking system RESIN-11 (Du et al., 2022). Specifically, we first extract individual entity men-

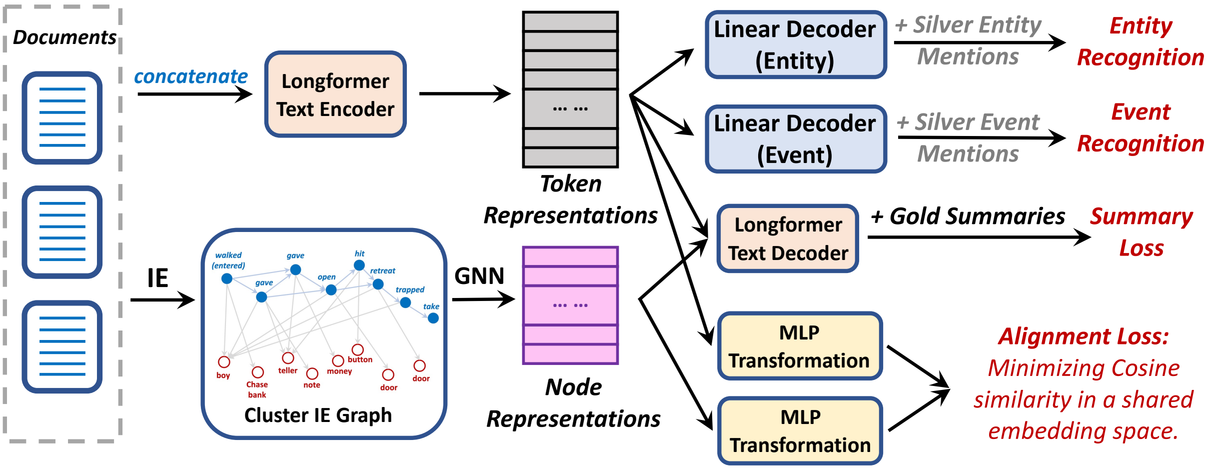


Figure 2: An overview of our IE-enhanced summarization pipeline. We first truncate and concatenate all documents in the cluster and feed them into the text encoder to obtain token representations. Meanwhile, we use a cross-document IE system to generate a cluster-level IE graph, and then use a GNN to get the node representations. During training, in addition to minimizing the distance between the generated and the reference summaries, we further use an entity and event recognition task and a node-text alignment loss to take advantage of the guidance from IE and improve MDS performance.

tions and event triggers as nodes from each document in the cluster. We then perform relation extraction, event argument role labeling, and event-event temporal relation extraction to add edges and to obtain a complete IE graph for each document. As shown in Figure 1, an example extracted event mention could be a "*Transport*" event triggered by "*walked*" with two event arguments "*boy*" and "*Chase bank*", where all such events are connected to form a unified document-level IE graph. To further connect document-level IE results into a cross-document IE graph, we then perform cross-document entity and event coreference resolution.[3] We merge all coreferenced entity and event nodes with their corresponding edges to form a cross-document IE graph. Specifically, if two nodes are labeled as the same entity or event, we merge these two nodes into a unified node and connect all related edges to it. It is worth noting that our framework does not rely on a specific IE systems and/or schema. Hence any form of structured IE outputs will work with our proposed method.

Notation Each node $v \in \mathcal{V}$ could be an entity or event trigger. We use $E = \{e_1, e_2, \cdots, e_{|E|}\}$ and $T = \{t_1, t_2, \cdots, t_{|T|}\}$ to denote the set of entities and event triggers respectively, where each e_i and t_i also act as a node in $\mathcal{V}$. Accordingly, there are three types of edges in $\mathcal{E}$ and we use p_{ij}, q_{ij}, and r_{ij} to represent the "*event-event*" temporal relations, "*event-entity*" argument roles, and "*entity-entity*" relations respectively. As shown in Figure 1, each blue node represents an event trigger (e.g., gave) while each brown node is an entity (e.g., bank), where the unique event triggers and entities are highlighted. The IE results include "*entity-entity*" relations connecting two different entities (e.g., <$button, door$>), "*event-argument*" links connecting an event trigger and an entity mention (e.g., <$gave, teller$>), and "*event-event*" temporal relations connecting two events (e.g., <$hit, retreat$>).

3 Approach

In this paper, our main goal is to improve multi-document summarization (MDS) with the extracted cross-document IE graph. As illustrated in Figure 2, we first concatenate all documents in a cluster and feed this concatenated input into a Longformer encoder (Beltagy et al., 2020) that is capable of handling long text sequences. We also use the cross-document IE system to obtain a cluster-level IE graph, as shown in the example highlighted in Figure 1, and use a graph attention network to obtain the node representations. During training, in addition to the cross-entropy summary loss between the generated and the ground-truth summaries, we propose two additional novel training objectives: (1) an *entity and event recognition* task that makes the model aware of the locations of important events and entities; and (2) an *alignment loss* between the IE graph nodes and their corresponding text spans

---

[3] The entity mentions extracted from ReFinED are merged according to the Wikidata IDs, while the event coreference resolution is done by a neural model (Lai et al., 2021).

to ensure that they are factually consistent in the latent space. We will go into details of our model design in the following sections.

3.1 Document Encoding

To handle the long input sequences, we use the encoder of the pre-trained *PRIMERA* (Xiao et al., 2022) model which is continually pre-trained from the Longformer-Encoder-Decoder (*LED*) model (Beltagy et al., 2020) to encode the documents and obtain the token representations $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots\}$. We truncate each document to the size of L_{max}/N (where N is the number of documents in the cluster), and concatenate all documents with a special token `[doc-sep]` to fit the maximum input length L_{max} of the *LED* model.[4]

$$\{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots\} = \text{Enc}\left(D_1, D_2, \cdots, D_N\right).$$

Similar to the work of Xiao et al. (2022), we assign the global attention on the `[doc-sep]` tokens to make sure that the model is aware of the document boundaries and that it analyzes the relationships between the documents.

In addition to directly encoding the documents, we also use the cross-document IE system described in Section 2.1 to extract a cross-document IE graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. Similar to Zhang and Ji (2021), we use an edge-conditioned graph attention network to encode the entity nodes E and event nodes T respectively. The initial node representations of entities and events are computed by the average of the representations over all tokens in the entity mention or event trigger.

$$\boldsymbol{e}_i = \frac{1}{|e_T - e_S|}\sum_{j=e_S}^{e_T-1}\boldsymbol{w}_j,\ \ \boldsymbol{t}_i = \frac{1}{|t_T - t_S|}\sum_{j=t_S}^{t_T-1}\boldsymbol{w}_j,$$

where $[e_T, e_S]$ and $[t_T, t_S]$ denote the entity and event trigger spans respectively. After initializing the node embeddings, the updated entity embeddings are computed as follows:

$$\boldsymbol{e}_i^{L+1} = \boldsymbol{e}_i^{L} + \gamma\cdot\sum_{j\in\mathcal{N}_i}\alpha_{ij}\boldsymbol{f}_n\left(\boldsymbol{v}_j^{L}\right).$$

In this equation, $\boldsymbol{f}_n(\cdot)$ is a linear transformation layer and γ is a hyper-parameter controlling the level of neighborhood aggregation, where a larger γ means more information from the neighbors is incorporated when updating the node representations. The attention weights α_{ij} are determined by the node pair and the type of the edge connecting the pair of nodes.

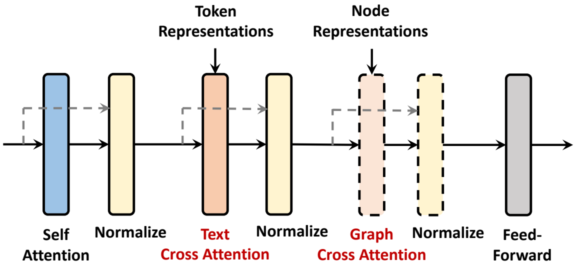


Figure 3: An example of the decoder layer components with graph cross-attention mechanism, where the dashed components are newly initialized weights while others are intialized from pre-trained weights.

$$\alpha_{ij} = \frac{\exp\left(\text{MLP}\left([\boldsymbol{v}_j, \boldsymbol{r}_{ij}, \boldsymbol{v}_i]\right)\right)}{\sum_{k=1}^{\mathcal{N}_i}\exp\left(\text{MLP}\left([\boldsymbol{v}_k, \boldsymbol{r}_{kj}, \boldsymbol{v}_i]\right)\right)},$$

where $\boldsymbol{r}_{ij}$ and $\boldsymbol{r}_{kj}$ are from a pre-initialized edge embedding matrix which could be optimized during training.[5] The event trigger embeddings are computed in the same way as the entities do. We use the node representations from the final layer as the output node representations.

3.2 Summary Generation

We use the pre-trained *LED* decoder to generate the summaries based on both token and node representations. In addition to the original pre-trained cross-attention mechanism $f_T(\cdot)$ for token representations, we include another similar cross-attention mechanism $f_G(\cdot)$ after f_T in all decoder layers for the system to model the relationships between each node in the graph and each token in the generated text. We use the pre-trained weights for text cross-attention mechanism $f_T(\cdot)$ and the graph cross-attention mechanism $f_G(\cdot)$ is randomly initialized, where both of them are continually optimized during the downstream training. An illustration of the pipeline in each decoder layer is shown in Figure 3. Therefore, each summary S_i is generated in an auto-regressive manner using the *LED* decoder $\text{Dec}(\cdot)$ with both text and graph cross-attention mechanism:

$$S_i = \text{Dec}\left([\text{BOS}], \{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots\}, \{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots\}\right),$$

where [BOS] is the *start* token in transformer decoders. Given a set of reference summaries

---

[4] The maximum length L_{max} is set as 4096 in pre-trained *PRIMERA* and *LED-large* models.

[5] We only consider three edge types here: event-event temporal relations, event-entity argument relations, and entity-entity relations.

$\hat{S}_1, \cdots, \hat{S}_N$ and a set of generated summaries $S_1, \cdots, S_N$, the summary loss is defined to minimize the cross-entropy distance $f_{CE}(\cdot)$ between each pair of summary sequences.

$$\mathcal{L}_{\text{summ}} = \frac{1}{N} \sum_{i=1}^{N} f_{CE}\left(S_i, \hat{S}_i\right). \tag{1}$$

3.3 Entity and Event Recognition

The main goal of our model is to use IE results to enhance the performance of the summarization task. We first add an auxiliary entity and event recognition task to make the model more sensitive to the locations of important events and entities. This will ensure that the model will not miss these events and entities when summarizing the document. Specifically, we use a Multi-Layer Perceptron (MLP) based classifier to classify each token into three different types: [ENTITY], [EVENT] or [NONE], and we use the spans of entity mentions and event triggers extracted by our proposed IE system to provide silver-standard training signals. Each token $\boldsymbol{w}_i$ is transformed to logits $\boldsymbol{p}_i$ by an MLP classifier:

$$\boldsymbol{p}_i = \text{softmax}\left(\text{MLP}\left(\boldsymbol{w}_i\right)\right). \tag{2}$$

Given a set of input tokens $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_M$, the entity and event recognition loss is computed as:

$$\mathcal{L}_{\text{recognition}} = -\sum_{i=1}^{M} p_{ij},$$

where j is the index of the correct label for w_i.

3.4 Node and Text Alignment

Incorporating graph information for summarization could be challenging, since the IE graphs are extracted from automatic extraction systems which may introduce noise and errors. To this end, we propose a novel alignment loss to minimize the distance between node representations and their corresponding texts to ensure coordination between the graphs and summarization text. Specifically, we first use two MLPs to map the node and text representations into the same embedding space Z:

$$\boldsymbol{z}_i^w = \text{MLP}_w\left(\boldsymbol{w}_i\right), \quad \boldsymbol{z}_i^v = \text{MLP}_v\left(\boldsymbol{v}_i\right),$$

where $\boldsymbol{z}_i^w$ and $\boldsymbol{z}_i^v$ denote the representations for token w_i and node v_i in the shared embedding space. Given each node v_i and the set of its corresponding text tokens $\mathcal{W}_i$, we minimize the cosine similarity between the node embedding $\boldsymbol{v}_i$ and the average embedding of all tokens in $\mathcal{W}_i$:

$$\mathcal{L}_{\text{align}} = \sum_{v_i \in \mathcal{V}} d_{cos}\left(\boldsymbol{z}_i^v, \frac{1}{|\mathcal{W}_i|} \sum_{w_j \in \mathcal{W}_i} \boldsymbol{z}_j^w\right). \tag{3}$$

The intuition behind $\mathcal{L}_{align}$ is to ensure that the node embedding is centered around its corresponding text. This helps ensure that the graphs and input text are factually consistent with each other, thereby reducing the errors and noise propagated from the IE system. As an example in Figure 1, the latent distance between each pair of nodes and texts (e.g., the node representation of *boy* and the text representation of its corresponding tokens *16-year-old boy*) are minimized to reduce the noise of the extracted graph.

Multi-Task Training. We conduct multi-task training where the total loss is a weighted sum from Equation (1), (2), and (3). The weighting coefficients β_1,β_2, and β_3 are tunable hyper-parameters.

$$\mathcal{L} = \beta_1 \cdot \mathcal{L}_{\text{summ}} + \beta_2 \cdot \mathcal{L}_{\text{recognition}} + \beta_3 \cdot \mathcal{L}_{\text{align}}$$

4 Experiments

4.1 Data

Our experiments are conducted on three most widely-used MDS benchmarks, where the detailed dataset statistics are shown in Table 1.

Dataset	# Train / Val / Test	Docs per Cluster	Average Summary Length
Multi-News	44972 / 5622 / 5622	2.8	217
WCEP-10	8158 / 1020 / 1022	9.1	28
DUC-2004	0 / 0 / 50	10	115

Table 1: Statistics of the MDS Datasets

Multi-News. The Multi-News benchmark (Fabbri et al., 2019) is the most widely-used dataset for multi-document summarization. The summaries are long and informative news abstracts written by human editors, and the documents are extracted from multifarious news articles.

WCEP-10. The WCEP-10 (Gholipour Ghalandari et al., 2020) dataset is extracted from Wikipedia Current Event Portal, where each document cluster also describes a news event. Compared to Multi-News, the WCEP dataset has a much larger number of documents in each cluster, and we manually reduce them to a maximum of 10 documents per cluster as previous research (Xiao et al., 2022; Parnell et al., 2022) did to obtain the WCEP-10 version of dataset. We include both Multi-News and WCEP-10 in our experiments to evaluate whether our model can stay effective in both long-summary and short-summary scenarios.

DUC-2004. There are only 50 test document clusters in DUC-2004 benchmark,[6] and we use this dataset to evaluate our model's zero-shot transfer ability. We train our model on Multi-News and directly test it on DUC-2004 since these two datasets have similar lengths of summaries.

4.2 Baselines and Implementation Details

For baselines, we mainly compare our model with state-of-the-art multi-document summarization models *PRIMERA* (Xiao et al., 2022) and *REFLECT* (Song et al., 2022). *REFLECT* only reports ROUGE scores on the Multi-News dataset and we directly use the reported scores for comparison. Besides, we also include a previous model *BART-Graph* (Pasunuru et al., 2021), which uses a linearized IE graph to improve summarization. We compare our model with it to see whether encoding the graph structurally improves the summarization performance. We also experiment with three ablation variants of our proposed model: (1) *Recognition-Only:* for the model with only the entity and event recognition loss; (2) *Alignment-Only:* for the model with only the graph encoder and the node-text alignment loss. (3) *Separate-Graphs:* for encoding the IE graphs for each document separately and using a collated matrix as the node representations. For Multi-News and WCEP-10, we train all of these models on the training set, choose the best model checkpoint based on the performance on the validation set, and test the models on the test set. For DUC-2004, we use the trained checkpoint on Multi-News dataset for evaluation, since the summary length on Multi-News is more similar to DUC-2004 compared with WCEP-10.[7]

4.3 Evaluation Metrics

Co-occurrence. Similar to previous research studies, we first include the most widely-used ***ROUGE-F1*** score which measures the overlap between the generated summaries and the reference summaries in terms of overlapping n-grams and longest common subsequence.

Factual Consistency. Intuitively, our proposed IE-enhanced summarization should improve factual consistency of the generated summary with the source documents, since the entities and events in the original documents are mined and memorized by the model through the two proposed IE enhancement loss. Therefore, we include several factuality metrics to measure the improvements in terms of factuality of the generated summaries. Specifically we use ***FactCC*** (Kryscinski et al., 2020), ***FactGraph*** (Ribeiro et al., 2022), ***EntityPrecision*** (Nan et al., 2021), **SUMMAC** (Laban et al., 2022), and **BERTSCORE** (Pagnoni et al., 2021).

Abstractiveness. To measure abstractiveness of our generated summaries, we use the MINT score (Dreyer et al., 2023), which is based on contiguous and non-contiguous extractive overlaps between summaries and their source documents. Our goal is to measure whether the novelty of the generated summary is sacrificed due to the improvements of factual consistency, e.g., by generating a more extractive summary.

4.4 Results

Table 2 shows the results of our proposed model, as well as the baselines on the three datasets. In general, the full version of our proposed model outperforms the baselines in terms of both ROUGE scores and factuality metrics while maintaining the same level of MINT scores. This shows that our model can generate high-quality summaries that are factually consistent without sacrificing any novelty. Specifically, entity and event recognition mainly improve factual consistency, while node-text alignment improves the similarity with the referenced summaries. This follows our intuition since the recognition task is mainly designed to help the model better notice the important event triggers and entity mentions, which prevents the model from hallucination and thereby improves factual consistency. On the other hand, the alignment loss can reduce the noise and errors in those extracted IE graphs, which makes the model better optimized on the ground-truth summaries.

4.5 Human Evaluation

We conduct a human evaluation on Amazon Mechanical Turk to evaluate the effect of adding our

---

[6] `https://duc.nist.gov/duc2004/`

[7] More detailed hyper-parameter settings can be found in Appendix A.

Evaluation	*Co-occurrence*			*Factual Consistency*					*Abstractiveness*
Metrics	R-1	R-2	R-L	FactCC	FactGraph	SummaC	Bert-P	EntityPrec	Mint
				Multi-News					
REFLECT	49.3	20.0	24.8	-	-	-	-	-	-
BART-Graph	49.2	19.0	24.0	74.2	74.1	86.0	**87.3**	89.9	81.8
PRIMERA	49.9	20.9	25.8	73.1	75.0	86.2	87.0	89.3	82.1
Separate-Graphs	49.8	20.4	25.8	74.7	75.2	86.2	87.0	89.5	82.1
Recognition-Only	50.0	20.8	26.0	77.4	76.1	86.5	87.1	**91.1**	82.0
Alignment-Only	50.3	20.9	26.3	75.6	74.9	87.7	87.1	90.8	82.1
Full Model	**50.3**	**21.1**	**26.4**	**77.8**	**76.5**	**87.9**	87.1	**91.1**	82.1
				WCEP-10					
PRIMERA	46.1	24.9	37.8	68.0	71.3	56.9	94.1	88.0	86.6
Separate-Graphs	46.1	24.8	37.8	69.1	71.6	57.0	94.0	89.1	86.6
Recognition-Only	46.1	24.8	**37.9**	71.2	**71.7**	57.1	94.0	91.0	86.5
Alignment-Only	**47.3**	**25.0**	**37.9**	68.5	71.4	57.6	**94.4**	90.5	86.5
Full Model	**47.3**	24.9	37.8	**71.5**	**71.7**	**57.7**	**94.4**	**91.3**	86.8
				DUC-2004					
PRIMERA	32.6	6.7	16.8	53.0	48.8	77.9	84.2	79.6	70.1
Separate-Graphs	32.6	6.6	16.8	54.2	49.9	77.6	**85.1**	80.4	70.1
Recognition-Only	32.5	6.8	16.8	54.2	**51.2**	76.8	84.7	82.3	70.4
Alignment-Only	32.8	**7.2**	17.1	53.2	49.1	78.9	84.3	80.0	70.2
Full Model	**32.9**	**7.2**	**17.3**	**54.8**	**51.2**	**79.1**	85.0	**84.0**	70.1

Table 2: Evaluation results with various metrics on the three MDS datasets. We primarily compare our results with three most recent transformer-based baselines *BART-Graph*, *REFLECT*, and *PRIMERA*. We also include two variants of our own model for ablation study, where we remove the recognition loss and the alignment loss respectively and test the model on these MDS datasets.

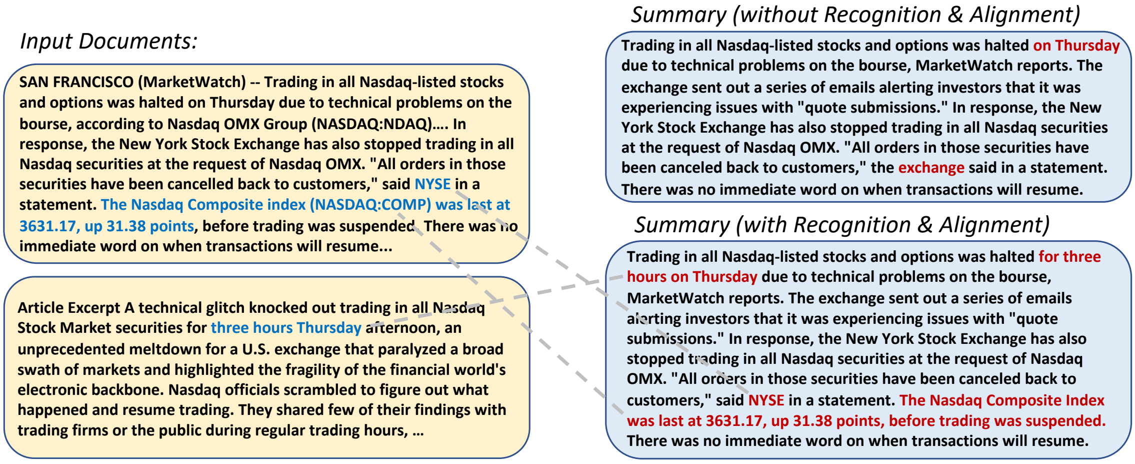
two proposed training objectives. We randomly select 300 document clusters for each of Multi-News and WCEP and use all the 50 document clusters in DUC-2004, asking three annotators per summary to score the factual consistency: 1 for major factual errors, 2 for minor factual errors, and 3 for no factual errors. Figure 5 in the Appendix shows the annotation guidelines. We aggregate the three judgements per summary using majority voting. We follow the qualification procedure for annotators described in Dreyer et al. (2023). Table 3 shows the percentages for each factuality score, where *baseline* denotes the *PRIMERA* (Xiao et al., 2022) model and *ours* denotes our proposed model. We find that our model can substantially reduce major factual errors on all three datasets, and is able to obtain higher average factuality scores on Multi-News and DUC-2004. Particularly, on DUC-2004 where the model is directly transferred from another dataset, our model can especially outperform the baseline in terms of factual consistency.

Dataset	Major (1.0)	Minor (2.0)	No (3.0)	Avg Scores
baseline	6.0%	11.3%	82.7%	2.767
Multi-News (ours)	**4.7%**	12.7%	82.7%	**2.780**
baseline	10.7%	9.0%	80.3%	**2.697**
WCEP-10 (ours)	**9.0%**	19.3%	71.7%	2.627
baseline	22.0%	22.0%	56.0%	2.340
DUC-2004 (ours)	**18.0%**	12.0%	70.0%	**2.520**

Table 3: Human evaluation results.

4.6 Qualitative Analysis

To better understand the effects made by our proposed training objectives, we look into the prediction results and show a typical example in Figure 4, explaining how our proposed method works to improve the summaries. In this example, the document cluster is mainly talking about a shut-down incident of the Nasdaq trading market. Compared to the summary from the baseline model, our model is better at memorizing the important facts and showing them in the output summary, e.g., the exact Nasdaq Index (3631.17) when the trading was suddenly suspended. Some other facts such as *"three hours"* are also memorized by our model but ignored by the baseline model. Moreover, our model is able to generate more informative mentions of those key entities (e.g., *NYSE*), where the baseline model fails to generate a named mention and only writes *"the exchange"*.



SAN FRANCISCO (MarketWatch) -- Trading in all Nasdaq-listed stocks and options was halted on Thursday due to technical problems on the bourse, according to Nasdaq OMX Group (NASDAQ:NDAQ).... In response, the New York Stock Exchange has also stopped trading in all Nasdaq securities at the request of Nasdaq OMX. "All orders in those securities have been cancelled back to customers," said NYSE in a statement. The Nasdaq Composite index (NASDAQ:COMP) was last at 3631.17, up 31.38 points, before trading was suspended. There was no immediate word on when transactions will resume...

Article Excerpt A technical glitch knocked out trading in all Nasdaq Stock Market securities for three hours Thursday afternoon, an unprecedented meltdown for a U.S. exchange that paralyzed a broad swath of markets and highlighted the fragility of the financial world's electronic backbone. Nasdaq officials scrambled to figure out what happened and resume trading. They shared few of their findings with trading firms or the public during regular trading hours, ...

Trading in all Nasdaq-listed stocks and options was halted on Thursday due to technical problems on the bourse, MarketWatch reports. The exchange sent out a series of emails alerting investors that it was experiencing issues with "quote submissions." In response, the New York Stock Exchange has also stopped trading in all Nasdaq securities at the request of Nasdaq OMX. "All orders in those securities have been canceled back to customers," the exchange said in a statement. There was no immediate word on when transactions will resume.

Trading in all Nasdaq-listed stocks and options was halted for three hours on Thursday due to technical problems on the bourse, MarketWatch reports. The exchange sent out a series of emails alerting investors that it was experiencing issues with "quote submissions." In response, the New York Stock Exchange has also stopped trading in all Nasdaq securities at the request of Nasdaq OMX. "All orders in those securities have been canceled back to customers," said NYSE in a statement. The Nasdaq Composite Index was last at 3631.17, up 31.38 points, before trading was suspended. There was no immediate word on when transactions will resume.

Figure 4: A qualitative example from our full model compared to the baseline *PRIMERA* model. Our model is better at preserving important facts and utilizing more informative mentions of the key entities.

5 Related Work

Multi-Document Summarization. Abstractive multi-document summarization (MDS) aims to build models to generate summaries given a set of similar documents related to the same topic. With the tremendous success of sequence-to-sequence pre-trained language models such as BART (Lewis et al., 2020) and T5 (Guo et al., 2022), finetuning on pre-trained models, like DeYoung et al. (2021); Parnell et al. (2022); Zhao et al. (2022); Moro et al. (2022); Song et al. (2022); Ernst et al. (2022), has become the primary style of methods for summarization tasks. There are also research studies on how to handle cross-document information overlap and redundancy. For example, Pasunuru et al. (2021) propose to use graph structures generated by OpenIE systems to make the model more sensitive about the main message of the document cluster. More recently, Xiao et al. (2022) propose to integrate entity overlap into the pre-training scheme, where the overlapping entities are used to select out salient sentences for pre-training.

Cross-Document Information Extraction. Information Extraction (IE) aims to extract structured representations from unstructured text, which includes various subtasks from Named Entity Recognition (Reich et al., 2022; Ayoola et al., 2022; Ding et al., 2021), to Relation Extraction (Yu et al., 2022; Tian et al., 2022), and Event Extraction (Xu et al., 2021; Yu et al., 2021) on news documents. There are also a number of research studies (Yao et al., 2021; Wu et al., 2022; Du et al., 2022) focusing on corpus-level cross-document extraction models. However, all these models still rely on cross-document entity and event coreference systems, which could bottleneck the efficiency and effectiveness of corpus-level IE models.

Joint IE and Summarization. IE and summarization share inherent similarities; both of them are designed to find the main information from an input natural language text. Therefore, it is promising to design a joint learning framework so that the two tasks could provide each other with mutual enhancement. There are some preliminary explorations of previous studies to train a model to learn IE and natural language generation (NLG) tasks jointly. For example, Li et al. (2021) train a template-based generative model for event argument extraction, and Du and Cardie (2020) propose to generate natural questions to ask the model for event extraction. However, although generation-based methods are proposed, these models are still doing a single task (IE) without multi-task settings for both IE and NLG. Recently, Lu et al. (2022) use summarization to provide indirect training signal for relation extraction tasks, however, their method is only suitable for relation extraction tasks and cannot cover general-concept IE tasks.

6 Conclusions

In this paper, we focus on improving multi-document summarization (MDS) model with cross-document Information Extraction (IE). We propose two novel training objectives – an entity and event recognition loss and a node-text alignment loss – that can help the model better utilize the signals from IE. Experimental results show that our model can generate summaries that are more factual, while not losing any abstractiveness.

7 Limitations

One limitation of our proposed method is the IE graphs are pre-extracted separately, where the IE model is not optimized during the model training and the IE results are only used as side inputs for summarization. It would be more exciting if we can really build a joint IE and Summarization model which are trained simultaneously in the pipeline, although it is very difficult since passing the gradients through a cross-document system is nearly intractable. We intend to address this limitation in our future work.

Acknowledgement

We thank the anonymous reviewers for their valuable feedback.

References

Tom Ayoola, Joseph Fisher, and Andrea Pierleoni. 2022. Improving entity disambiguation by reasoning over a knowledge base. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2899–2912, Seattle, United States. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Jay DeYoung, Iz Beltagy, Madeleine van Zuylen, Bailey Kuehl, and Lucy Wang. 2021. MSˆ2: Multi-document summarization of medical studies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7494–7513, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

Markus Dreyer, Mengwen Liu, Feng Nan, Sandeep Atluri, and Sujith Ravi. 2023. Evaluating the tradeoff between abstractiveness and factuality in abstractive summarization. In *Proceedings of the 2023 Conference of the European Chapter of the Association for Computational Linguistics*.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Xinya Du, Zixuan Zhang, Sha Li, Pengfei Yu, Hongwei Wang, Tuan Lai, Xudong Lin, Ziqi Wang, Iris Liu, Ben Zhou, Haoyang Wen, Manling Li, Darryl Hannan, Jie Lei, Hyounghun Kim, Rotem Dror, Haoyu Wang, Michael Regan, Qi Zeng, Qing Lyu, Charles Yu, Carl Edwards, Xiaomeng Jin, Yizhu Jiao, Ghazaleh Kazeminejad, Zhenhailong Wang, Chris Callison-Burch, Mohit Bansal, Carl Vondrick, Jiawei Han, Dan Roth, Shih-Fu Chang, Martha Palmer, and Heng Ji. 2022. RESIN-11: Schema-guided event prediction for 11 newsworthy scenarios. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 54–63, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Ori Ernst, Avi Caciularu, Ori Shapira, Ramakanth Pasunuru, Mohit Bansal, Jacob Goldberger, and Ido Dagan. 2022. Proposition-level clustering for multi-document summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1765–1779, Seattle, United States. Association for Computational Linguistics.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. 2020. A large-scale multi-document summarization dataset from the Wikipedia current events portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1302–1308, Online. Association for Computational Linguistics.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736, Seattle, United States. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Tuan Lai, Heng Ji, Trung Bui, Quan Hung Tran, Franck Dernoncourt, and Walter Chang. 2021. A context-dependent gated module for incorporating symbolic semantics into event coreference resolution. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3491–3499, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Keming Lu, I Hsu, Wenxuan Zhou, Mingyu Derek Ma, Muhao Chen, et al. 2022. Summarization as indirect supervision for relation extraction. *arXiv preprint arXiv:2205.09837*.

Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, and Davide Freddi. 2022. Discriminative marginalized probabilistic neural method for multi-document summarization of medical literature. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 180–189, Dublin, Ireland. Association for Computational Linguistics.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Jacob Parnell, Inigo Jauregi Unanue, and Massimo Piccardi. 2022. A multi-document coverage reward for RELAXed multi-document summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5112–5128, Dublin, Ireland. Association for Computational Linguistics.

Ramakanth Pasunuru, Mengwen Liu, Mohit Bansal, Sujith Ravi, and Markus Dreyer. 2021. Efficiently summarizing text and graph encodings of multi-document clusters. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4768–4779, Online. Association for Computational Linguistics.

Aaron Reich, Jiaao Chen, Aastha Agrawal, Yanzhe Zhang, and Diyi Yang. 2022. Leveraging expert guided adversarial augmentation for improving generalization in named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1947–1955, Dublin, Ireland. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Mengwen Liu, Iryna Gurevych, Markus Dreyer, and Mohit Bansal. 2022. FactGraph: Evaluating factuality in summarization with semantic graph representations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3238–3253, Seattle, United States. Association for Computational Linguistics.

Yun-Zhu Song, Yi-Syuan Chen, and Hong-Han Shuai. 2022. Improving multi-document summarization through referenced flexible extraction with credit-awareness. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1667–1681, Seattle, United States. Association for Computational Linguistics.

Yuanhe Tian, Yan Song, and Fei Xia. 2022. Improving relation extraction through syntax-induced pre-training with dependency masking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1875–1886, Dublin, Ireland. Association for Computational Linguistics.

Xueqing Wu, Kung-Hsiang Huang, Yi Fung, and Heng Ji. 2022. Cross-document misinformation detection based on event graph reasoning. In *Proceedings of the 2022 Conference of the North American Chapter* of the Association for Computational Linguistics: Human Language Technologies, pages 543–558, Seattle, United States. Association for Computational Linguistics.

Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. 2022. PRIMERA: Pyramid-based masked sentence pre-training for multi-document summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5245–5263, Dublin, Ireland. Association for Computational Linguistics.

Runxin Xu, Tianyu Liu, Lei Li, and Baobao Chang. 2021. Document-level event extraction via heterogeneous graph-based interaction model with a tracker. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3533–3546, Online. Association for Computational Linguistics.

Yuan Yao, Jiaju Du, Yankai Lin, Peng Li, Zhiyuan Liu, Jie Zhou, and Maosong Sun. 2021. CodRED: A cross-document relation extraction dataset for acquiring knowledge in the wild. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4452–4472, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiaxin Yu, Deqing Yang, and Shuyu Tian. 2022. Relation-specific attentions over entity mentions for enhanced document-level relation extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1523–1529, Seattle, United States. Association for Computational Linguistics.

Pengfei Yu, Heng Ji, and Prem Natarajan. 2021. Lifelong event detection with knowledge transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5278–5290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Chao Zhao, Tenghao Huang, Somnath Basu Roy Chowdhury, Muthu Kumar Chandrasekaran, Kathleen McKeown, and Snigdha Chaturvedi. 2022. Read top news first: A document reordering approach for multi-document news summarization. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 613–621, Dublin, Ireland. Association for Computational Linguistics.

A Experiment Details

We list our detailed hyper-parameter settings for training our model on each of the datasets in Table 4 and Table 5, where each hyper-parameter is determined based on grid search among 5 candidate values. We train our model on 8 NVIDIA V100 GPUs with 32GB memory, and the total training time is about 7 hours for Multi-News and 3 hours for WCEP-10.

Hyper-parameters	Values
Num of features for each node	1,024
Num of GNN layers	1
Message Passing Level γ	0.01
Weights of the losses β_1, β_2, β_3	1.0, 0.2, 0.2
Learning Rate	3e-5
Batch Size	16
Maximum Length of Generated Summaries	256
Maximum Training Steps	25,000
Warm-up Steps	2,500
Beam Size for Generation	5

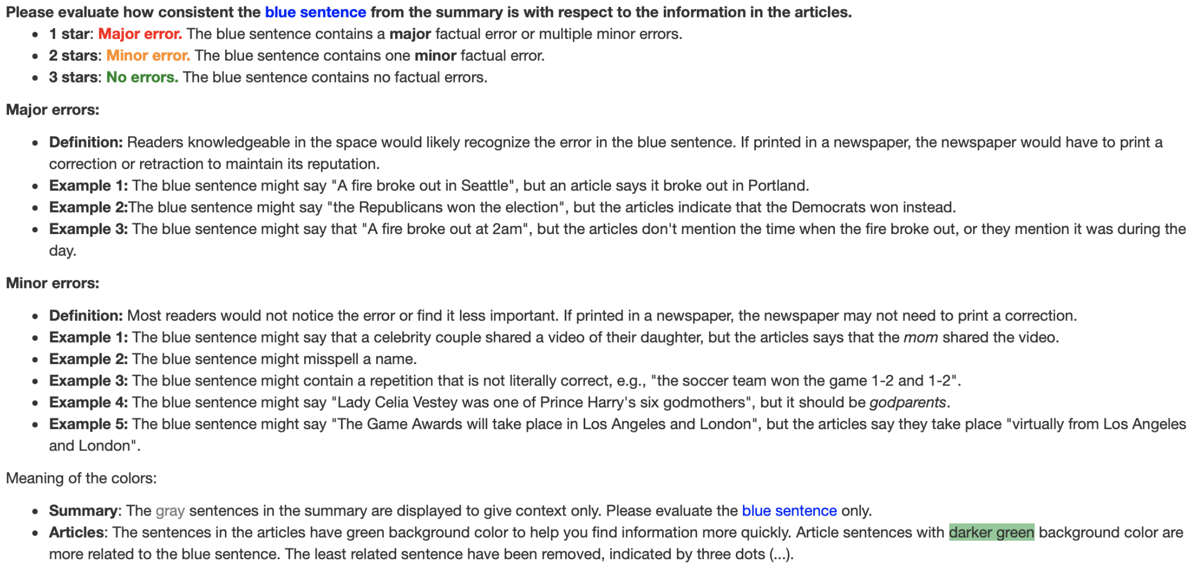
Table 4: Detailed hyper-parameter settings for model training on Multi-News.

Hyper-parameters	Values
Num of features for each node	1,024
Num of GNN layers	1
Message Passing Level γ	0.005
Weights of the losses β_1, β_2, β_3	1.0, 0.1, 0.1
Learning Rate	3e-5
Batch Size	16
Maximum Length of Generated Summaries	50
Maximum Training Steps	5,000
Warm-up Steps	500
Beam Size for Generation	5

Table 5: Detailed hyper-parameter settings for model training on WCEP-10.

B Annotation Guidelines

We use Amazon MTurk to do human evaluation, where the detailed annotation guidelines for human evaluators are shown in Figure 5.



Please evaluate how consistent the blue sentence from the summary is with respect to the information in the articles.

- **1 star**: **Major error.** The blue sentence contains a **major** factual error or multiple minor errors.
- **2 stars**: **Minor error.** The blue sentence contains one **minor** factual error.
- **3 stars**: **No errors.** The blue sentence contains no factual errors.

Major errors:

- **Definition:** Readers knowledgeable in the space would likely recognize the error in the blue sentence. If printed in a newspaper, the newspaper would have to print a correction or retraction to maintain its reputation.
- **Example 1:** The blue sentence might say "A fire broke out in Seattle", but an article says it broke out in Portland.
- **Example 2:**The blue sentence might say "the Republicans won the election", but the articles indicate that the Democrats won instead.
- **Example 3:** The blue sentence might say that "A fire broke out at 2am", but the articles don't mention the time when the fire broke out, or they mention it was during the day.

Minor errors:

- **Definition:** Most readers would not notice the error or find it less important. If printed in a newspaper, the newspaper may not need to print a correction.
- **Example 1:** The blue sentence might say that a celebrity couple shared a video of their daughter, but the articles says that the *mom* shared the video.
- **Example 2:** The blue sentence might misspell a name.
- **Example 3:** The blue sentence might contain a repetition that is not literally correct, e.g., "the soccer team won the game 1-2 and 1-2".
- **Example 4:** The blue sentence might say "Lady Celia Vestey was one of Prince Harry's six godmothers", but it should be *godparents*.
- **Example 5:** The blue sentence might say "The Game Awards will take place in Los Angeles and London", but the articles say they take place "virtually from Los Angeles and London".

Meaning of the colors:

- **Summary**: The gray sentences in the summary are displayed to give context only. Please evaluate the blue sentence only.
- **Articles**: The sentences in the articles have green background color to help you find information more quickly. Article sentences with darker green background color are more related to the blue sentence. The least related sentence have been removed, indicated by three dots (...).

Figure 5: Annotation instructions to annotate factual consistency on Mechanical Turk.

What happens *before* and *after*: Multi-Event Commonsense in Event Coreference Resolution

Sahithya Ravi[1,2] **Chris Tanner**[3,4] **Raymond Ng**[1] **Vered Shwartz**[1,2]
[1] University of British Columbia
[2] Vector Institute for AI
[3] Massachusetts Institute of Technology
[4] Kensho Technologies
{sahiravi, rng, vshwartz}@cs.ubc.ca, cwt@mit.edu

Abstract

Event coreference models cluster event mentions pertaining to the same real-world event. Recent models rely on contextualized representations to recognize coreference among lexically or contextually similar mentions. However, models typically fail to leverage commonsense inferences, which is particularly limiting for resolving lexically-divergent mentions. We propose a model that extends event mentions with temporal commonsense inferences. Given a complex sentence with multiple events, e.g., "The man killed his wife and got arrested", with the target event "arrested", our model generates plausible events that happen before the target event – such as "the police arrived", and after it, such as "he was sentenced". We show that incorporating such inferences into an existing event coreference model improves its performance, and we analyze the coreferences in which such temporal knowledge is required.

1 Introduction

The goal of cross-document event coreference resolution is to determine if various event mentions (e.g. *shot*, *gunshot*), across one or more documents, refer to the same event. Existing systems represent each mention within its context using a language model (Cattan et al., 2021a; Allaway et al., 2021), and train a scorer to predict if two mentions corefer, based on their lexical and contextual similarity.

While many coreferring mention pairs in event coreference datasets such as ECB+ (Cybulska and Vossen, 2014) are lexically and contextually similar, or even share the same lemma (Wolfe et al., 2015), the difficulty arises for dissimilar coreferring mentions. For example, in Figure 1, *spent* and *hospitalized* are coreferring. These mentions are not lexically similar, and are not often used in similar contexts. In this paper, we improve the ability of existing cross-document event coreference systems to resolve such challenging coreferring mentions, by providing additional context in the form of commonsense knowledge. We focus on two temporal commonsense relations — *before* and *after* — pertaining to typical events that happen before and after the target event. For instance, in Figure 1, we may infer that before Dalton was *shot*, a shooter loaded their gun. Similarly, we may infer that Dalton was hurt prior to his *hospitalization* and got discharged afterward.

Our first contribution is the development of a commonsense reasoning engine that can reason about these two temporal relations. Existing commonsense models (Gabriel et al., 2021a; Hwang et al., 2021) may generate such inferences for simple sentences with a single event, such as "Bryant Dalton was shot", but they do not support complex sentences with multiple events of interest (e.g. *shot*, *hospitalized*). Further, they may conflate the inferences for different events. We develop a multi-event commonsense model that considers the entire context and is capable of generating separate inferences for each target event in complex sentences.

As an additional contribution, we incorporate the inferences into the pairwise mention scorer of a cross-document event coreference system (Cattan et al., 2021a). We produce *before* and *after* inferences for each event mention. We then embed the inferences, either by attending each mention to its own inferences (intra-span) or to the other mention's inferences (inter-span).

The results confirm that commonsense inferences are useful for event coreference. Each of our model variants improves upon the baseline performance, with the intra-span version performing the best. We further analyze the successful predictions and interpret how the commonsense inferences help resolve difficult mention pairs. In the future, we plan to extend our multi-event inference engine to additional commonsense knowledge types and apply it to other discourse tasks, such as summa-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1708–1724
May 2-6, 2023 ©2023 Association for Computational Linguistics

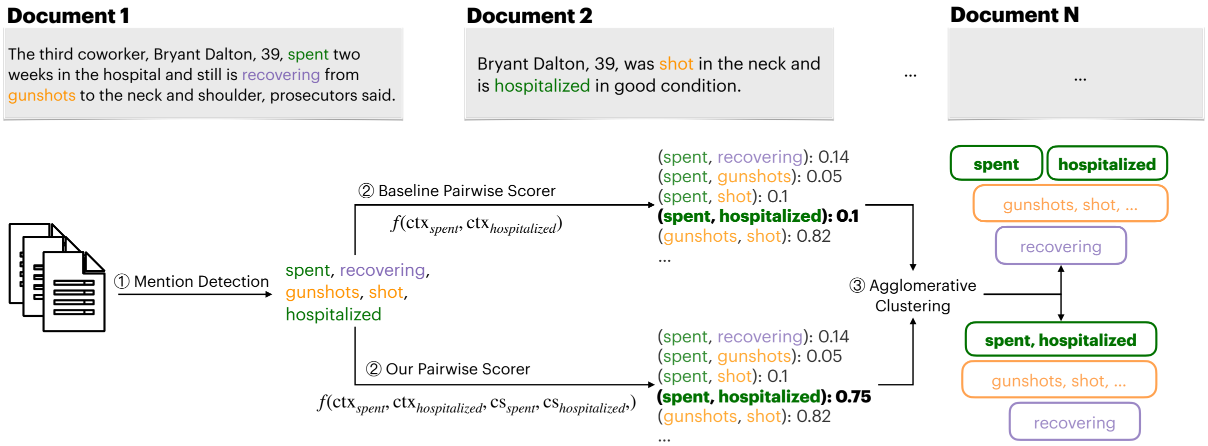


Figure 1: The architecture of our event coreference model, exemplified on a set of documents. The baseline pairwise scorer is based on contextual similarity (top), and we enhance it with temporal commonsense inference embeddings – indicated with *cs* (bottom). Such inferences help identify lexically divergent co-referring pairs, such as *spent* [time at a hospital] and *hospitalized*.

rization, dialogue, and story comprehension.[1]

2 Background

In this work, we improve the performance of a system for cross-document coreference resolution by incorporating temporal commonsense inferences pertaining to the events. We first provide background on event coreference resolution (§2.1). We then describe related work concerning event-centered commonsense (§2.2), along with approaches for using language models for data augmentation (§2.3).

2.1 Event Coreference Resolution

Event coreference resolution aims to cluster event mentions that refer to the same underlining real-world occurrence. Our work focuses on cross-document coreference resolution (CD), which aims to resolve mentions across an entire corpus of documents. In contrast, the problem of within-document coreference resolution (WD) only resolves mentions on a per-document basis. Event coreference is often performed jointly with *entity* coreference resolution, which concerns resolving mentions of people, locations, and organizations.

Datasets. In this paper, we use the ECB+ dataset proposed by Cybulska and Vossen (2014) and widely accepted as the standard benchmark for coreference resolution (CD). ECB+ contains 86 sub-topics, each of which concerns a specific news event. To introduce complexity and difficulty, each sub-topic is highly similar to – yet distinctly different from – exactly one other sub-topic. ECB+ includes both entity and event mentions; however, this work focuses solely on events. Table 1 shows event statistics from ECB+ corpus.

	Train	Dev	Test
Topics (subtopic-pairs)	25	8	10
Event mentions	3808	1245	1780
Event clusters	1527	409	805

Table 1: Statistics on the standard train/dev/test split of event coreferences in ECB+ dataset (Cybulska and Vossen, 2014)

Models. Recent approaches to CD event coreference often follow the architecture described in Figure 1. First, candidate event mentions are extracted from documents. Second, a pairwise scorer is trained to classify every pair of mentions as being coreferent or not. Finally, these scores are used to form distinct clusters of event mentions, typically using agglomerative clustering. Amongst these components, coreference models tend to mostly vary in their scoring approach (i.e., second component).
Early approaches relied on lexical and syntactic features (Yang et al., 2015; Choubey and Huang, 2017), or used semantic roles to encode the relationships between entities and events. Recently, Meged et al. (2020) improved performance by leveraging a resource of predicate paraphrases. Finally, Lai et al. (2021) incorporated entities, relations, and events extracted from a state-of-the-art information ex-

---

[1] The code is available here.

traction system. Generally, current state-of-the-art models often rely on pre-trained language models to compute a contextualized representation for each candidate mention, which serve as input to the pairwise scorer (e.g. Yu et al., 2020; Zeng et al., 2020; Cattan et al., 2021a; Allaway et al., 2021).

Our model is an enhancement of the model proposed by Cattan et al. (2021a). It targets both entity and event coreference resolution, and in an end-to-end fashion it performs mention extraction, pairwise scoring, and clustering (Figure 1). Mentions are represented by contextualized embeddings from RoBERTa (Liu et al., 2019). We chose to base our model on Cattan et al. (2021a) for two reasons. First, it is a simple model following the standard approach presented in Figure 1. Later approaches rely on hierarchical representations (Yadav et al., 2021a) or discourse coherence theory (Held et al., 2021). Second, it is based on RoBERTa and is more efficient and less memory consuming than the succeeding CDLM model (Caciularu et al., 2021) that is based on the much larger Longformer model (Beltagy et al., 2020).

More recently, Yadav et al. (2021a) built on Cattan et al. (2021a) by proposing a hierarchical approach to representing uncertainty of clustering event and entity mentions. The state-of-the-art models for cross document coreference are Caciularu et al. (2021), which models cross-text relationships by using larger context windows, and Held et al. (2021), which applies discourse coherence theory to coreference.

2.2 Event-Centric Commonsense

Commonsense reasoning helps humans bridge the gap between utterance and intended meaning. Reasoning about events has long been of interest to AI research. Schank and Abelson (1975) introduced "scripts" as a prototypical series of events, e.g. going to a restaurant is composed of ordering food, eating, and paying, and the participants: customer, waiter, and cook. Various methods have been proposed to learn such scripts from text (e.g. Chambers and Jurafsky, 2008; Pichotta and Mooney, 2014; Rudinger et al., 2015).

The ATOMIC knowledge base (Hwang et al., 2021; Sap et al., 2019) consists of 1.1M crowdsourced event-relation-event triplets pertaining to the causes, effects, and mental states of the event participants. To generate contextually-relevant ATOMIC-style inferences, Bosselut et al. (2019) developed COMET, a pre-trained language model fine-tuned on ATOMIC. COMET has shown promising results on tasks such as therapy chatbots (Kearns et al., 2020), persona-grounded dialogue (Majumder et al., 2020), figurative language interpretation and generation (Chakrabarty et al., 2020, 2022), and temporal ordering of sentences (Ghosal et al., 2021).

Several variants of COMET have been subsequently released. ParaCOMET (Gabriel et al., 2021a) adapts COMET to generate sentence-level inferences within the context of an entire paragraph. VisualCOMET (Park et al., 2020) generates ATOMIC-style inferences for images. Finally, the updated version of COMET (Hwang et al., 2021) extends the relation inventory and crowdsources more inferences. The additional inferences include the two temporal relations that are the most relevant to our work, "happens before" and "happens after".

2.3 LM-generated Data Augmentation

The success of using large pre-trained LMs in a few-shot setup for generation tasks has led to an increased interest in using such models to generate data for downstream tasks. Recent work augmented datasets by fine-tuning a pre-trained LM on real data, then generated new, silver-labelled instances (Anaby-Tavor et al., 2020; Papanikolaou and Pierleoni, 2020; Kumar et al., 2020). Similarly, the few-shot capabilities of GPT-3 (Brown et al., 2020) were leveraged to generate free-text explanations (Wiegreffe et al., 2022), semantically-related sentence pairs (Schick and Schütze, 2021), atomic event commonsense triples (West et al., 2022), and labels for various generation and understanding tasks (Wang et al., 2021). In this work, we fine-tune GPT-3 with minimal human supervision to generate additional contextual data pertaining to events.

3 Method

The architecture of our method is shown in Figure 1. We use the same clustering method as in Cattan et al. (2021a) but revise the pairwise scorer. Our goal is to improve the model's ability to resolve coreferences between mention pairs that are not lexically or contextually similar, but where one mention could be inferred from the other using commonsense knowledge and reasoning. Thus, we develop a commonsense inference engine (Sec 3.1)

Instructions: Read the context sentence and write at least **two** inferences for question 1 and **two** inferences for question 2. As shown in the examples, each inference is expected to be a short sentence between **5-10 words**.

Context: A publicist says Tara Reid has **checked herself** into rehab.
Question 1: What typically happens before the event checked herself?
Question 2: What typically happens after the event checked herself?

Figure 2: An example task on Amazon Mechanical Turk.

and use it to enhance the pairwise scorer (Sec 3.2).

3.1 Multi-Event Commonsense Inferences

We enhance the pairwise scorer with commonsense inferences regarding the events' temporal aspects. Specifically, we focus on plausible events that might have happened before or after the target event. For example, in Figure 1, after being hospitalized, the victim received treatment.

We found COMET and its variants to be ineffective for generating inferences for our task. COMET was trained on the ATOMIC knowledge base (Sap et al., 2019). As the name implies, events are atomic, i.e., comprise a single verb phrase. Conversely, the existing event coreference datasets are based on news articles, where sentences often contain multiple events. COMET predictions for document 1 (Figure 1) have no indication which verb they pertain to. Moreover, COMET predicts that what happens after document 1 is murder, which contradicts the fact that the victim survived and was taken to the hospital. ParaCOMET (Gabriel et al., 2021b) facilitates generating consistent inferences for multi-sentence paragraphs, but it was trained on the ROCStories dataset (Mostafazadeh et al., 2016), which is in the fiction domain and in which sentences are also simple.

To that end, we trained a new multi-event commonsense inference engine. Given a sentence with multiple events (such as document 1), and a target event (e.g. *hospitalized*), the goal is to generate what might have happened before and after the target event—in the context of the entire sentence.

Model. We base the inference engine on GPT-3 (Brown et al., 2020). While GPT-3 is not directly applicable to the task of event coreference (Yang et al., 2022), it has been shown to contain a wealth of factual and commonsense knowledge as a result of extensive pre-training. Our goal is to use this knowledge to generate event-centric commonsense inferences without requiring extensive training. GPT-3 is especially well-suited for this task, as it has shown remarkable performance in learning from fewer examples in a variety of tasks.

Data. As the first step in training a multi-event commonsense model, we crowdsourced annotations for 100 events – using the gold standard event mentions from the ECB+ training set. To include a wide range of topics, we selected the first four events from each of the 25 topics in the training set.

We presented workers with a sentence with one or more events, and asked them to describe what happens immediately before and after the target event. Figure 2 shows an example[2]. We obtained annotations from three workers for each sentence, and instructed workers to write at least two inferences for each relation. This yielded a total of 600 inferences ($100 \times 3 \times 2 = 600$). We carefully reviewed the data and removed a handful of inferences that were of poor quality (i.e., incomplete or irrelevant sentences, which amounted to roughly 5% of the annotations).

The annotation task was conducted on Amazon Mechanical Turk (AMT). To ensure the quality of annotations, we required workers to have previously completed 5,000 AMT tasks, and to have an acceptance rate of 98% or higher. We limited the worker location to the U.S. and Canada, and presented workers with a qualification test similar to the task. We paid 7 cents for each event.

Training. We fine-tuned GPT-3 on the collected inferences. The input and output format was as follows:

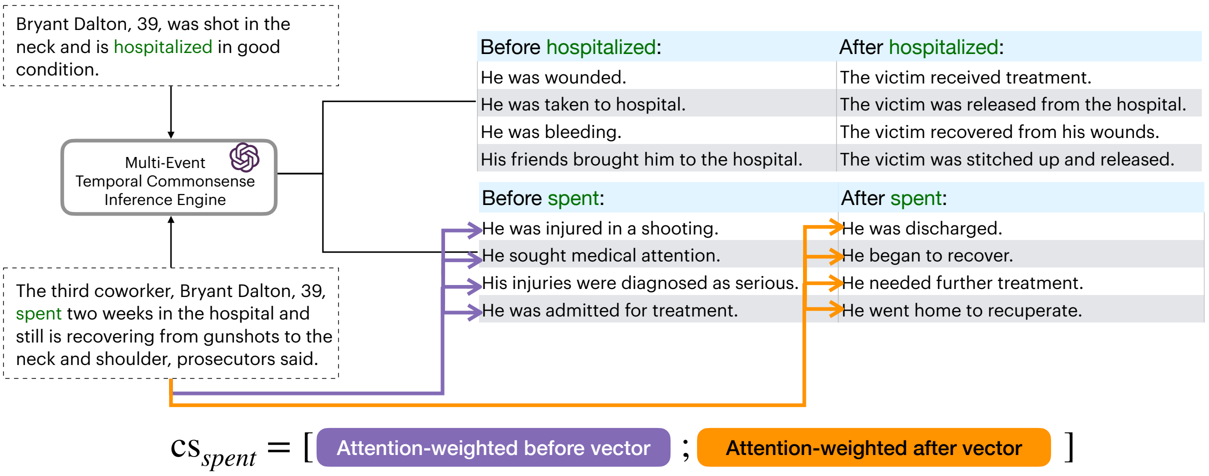
`Context:` <context>
`Event:` <event>
`Before:` <before>
`After:` <after>

Table 7 shows the format inputted into GPT-3 for training (top row) and inference (bottom row).

Inference To generate inferences, we prompt the fine-tuned GPT-3 model with the context and the event. We generate up to 150 tokens using top-p decoding (Holtzman et al., 2020) with a cumulative

---

[2] See Appendix E for the exact template.



Bryant Dalton, 39, was shot in the neck and is hospitalized in good condition.

The third coworker, Bryant Dalton, 39, spent two weeks in the hospital and still is recovering from gunshots to the neck and shoulder, prosecutors said.

Before hospitalized:	After hospitalized:
He was wounded.	The victim received treatment.
He was taken to hospital.	The victim was released from the hospital.
He was bleeding.	The victim recovered from his wounds.
His friends brought him to the hospital.	The victim was stitched up and released.

Before spent:	After spent:
He was injured in a shooting.	He was discharged.
He sought medical attention.	He began to recover.
His injuries were diagnosed as serious.	He needed further treatment.
He was admitted for treatment.	He went home to recuperate.

$$\mathrm{cs}_{spent} = [\ \text{Attention-weighted before vector}\ ;\ \text{Attention-weighted after vector}\ ]$$

Figure 3: An illustration of the new additions to the pairwise scorer. We input each document into a GPT-3-based multi-event temporal commonsense inference engine, which outputs plausible events that happened before and after the target event (e.g. *spent*). For each temporal relation (i.e., *before* and *after*), we embed the corresponding inferences and compute an attention-weighted vector. We concatenate the *before* and *after* vectors to the mention representations as input to the pairwise scorer.

Context	**Before**	**After**
	Human-Written	
[...] Chris Weitz will direct the sequel to Twilight, New Moon, replacing Catherine Hardwicke.	They couldn't strike a [...] deal with Hardwicke. The executive producer contacted Weitz's agent. Weitz's agent communicated the director's message. They decided to replace him.	Chris Weitz received an advance from the studio. Chris signed the contract. His agent put out a press release. Chris was happy.
[...] Chris Weitz will direct the sequel to Twilight, New Moon, replacing Catherine Hardwicke.	The director oversaw the hiring of shooting staff. The director oversaw several screen tests. He writes a movie script. He needed to sign a contract.	People watch the movie. He gets paid. The movie gets released in movie theatres. The movie makes a huge collection.
	Model-Generated	
Lindsay Lohan checks into rehab at Betty Ford Center, rehires longtime lawyer Shawn Holley	She decided to change her life from bad to good. She decided to seek help for her addiction. She is assessed by the staff at Betty Ford. She is welcomed by staff members at Betty Ford.	She is treated for her addiction. She attended daily group therapy meetings. She ends up in the hospital. She no longer has a problem.
Lindsay Lohan checks into rehab at Betty Ford Center, rehires longtime lawyer Shawn Holley	Lindsay needs advice on her case. Lindsay needs legal counsel in her case. she fired her old lawyer. she got a new director.	He gets a good pay. He looks for another case. she went through planning stages of her recovery. she started to addiction treatment.

Table 2: Human-written (top) and model-generated (bottom) examples from our multi-event temporal commonsense inference engine. Some examples are slightly abbreviated for readability.

probability of $p = 0.9$. Table 2 provides examples of the training data (top part) and generated inferences (bottom part) of our multi-event commonsense inference engine. Note, both the human-written and model-generated inferences differ for different events belonging to the same context. For example, according to our model, *after* the event "Lindsay checks into rehab," a plausible inference is that "she gets treated for her addiction." Yet, *after* the event "she rehires her longtime lawyer," our model infers that "he gets a good pay." [3]

3.2 Inference-Enhanced Pairwise Scorer

Figure 3 shows the overall architecture of our commonsense-enhanced pairwise scorer. We follow Cattan et al.'s mention span representation for mention m_i:

$$\mathrm{ctx}_i = [x_{START(i)}, x_{LAST(i)}, \hat{x}_i, l_i] \quad (1)$$

---

[3] We also experimented with prompting GPT-3 in a few-shot setup (Sec 5.3).

where x_j corresponds to the RoBERTa (Liu et al., 2019) embedding of the jth token in the span. Each mention is represented as the concatenation of: the first ($x_{START(i)}$) and last ($x_{LAST(i)}$) tokens; an attention-weighted sum of tokens $\hat{x}_i$; and a feature vector denoting the length l_i.

To incorporate the commonsense inference, we use the inference engine to generate up to $k = 5$ inferences for each of the before (b) and after (a) relations: $b_1 ... b_k$, and $a_1 ... a_k$. We describe below the representation of the relation vector using *after* as an example. The representation of the *before* relation is identical. We first compute the contextualized representation of each inference similarly to the span representations in Equation 1.

We then stack all the contextualized representations of the inferences:

$$\overrightarrow{A_i} = [\mathrm{ctx}_{a1} \ldots \mathrm{ctx}_{ak}] \quad (2)$$

and input them into a single head attention layer, which produces a single attention-weighted vector for the *after* relation.

In the context of the pairwise scorer, consider that we have two mention spans m_i and m_j and their corresponding *after* inference representations A_i and A_j. We implement two variants of the attention mechanism:

1. *Intra-span*, where the attention is between the mention span m_i and the corresponding inferences $\overrightarrow{A_i}$. This is exemplified in Figure 3, where attention is computed between the mention span of *spend* and the inferences corresponding to the same document. The query vector is the mention span ctx_i, and the key vector is the contextualized *after* vector $\overrightarrow{A_i}$. The idea behind this method is to emphasize inferences that are the most relevant to the given mention and provide additional context.

2. *Inter-span*, where the attention is between the mention span m_i and the inferences generated for the context of the other mention, $\overrightarrow{A_j}$. For example, in Fig 3, this would mean the purple and orange arrows originating in document 1 would need to be moved to the top row of inferences, corresponding to document 2. The query vector is the span ctx_i, and the key vector is the contextualized *after* vector $\overrightarrow{A_j}$. The goal of this method is to emphasize inferences that are relevant to the other mention, and to bring lexically divergent mentions closer.

In both cases, this leads to an attention-weighted commonsense vector for each of the before and after relations, which are then concatenated to create a single commonsense vector $\mathrm{cs}_i = [\overrightarrow{B_i}, \overrightarrow{A_i}]$ as shown in Figure 3. The input to the pairwise scorer for mentions m_i and m_j is therefore:

$$g_{i,j} = [\mathrm{ctx}_i, \mathrm{ctx}_j, \mathrm{cs}_i, \mathrm{cs}_j] \quad (3)$$

The scores from the pairwise scorer are then used to cluster mentions using agglomerative clustering, identically to Cattan et al. (2021a). Agglomerative clustering merges the most similar cluster pairs until their pairwise similarity score falls below a predetermined threshold.

4 Experimental Setup

4.1 Implementation Details

The implementation of our model is based on Cattan et al. (2021a). We use their official codebase[4] and modify it to support the additional components. Since we use gold event mentions to generate inferences from the multi-event commonsense inference engine (Sec 3.1), during both training and inference, we train and evaluate the coreference pipeline on gold mentions. During testing, we evaluate both GPT3 and the coreference system on new gold mentions that are not seen during training. This is in contrast to Cattan et al. (2021a) which learned to extract candidate mention spans and train the coreference system. However, using gold mentions is common practice among many coreference systems where the focus is on improving the pairwise scorer (e.g. Barhom et al., 2019; Yadav et al., 2021a). For a fair comparison, we report the baseline performance by re-running Cattan et al. (2021a) using gold mentions similar to the baseline used in Yadav et al. (2021b). We compare this baseline to two variants of our model, based on intra-span and inter-span attention (Sec 3.2). We train all model versions using 15 different random seeds, and we report the average performance.

For our GPT-3 based inference engine, we fine-tuned the *Davinci* model which we accessed via the OpenAI API.[5]. The hyperparameters for all the models are detailed in Appendix B.

4.2 Evaluation Setup and Metrics

The primary metric we use is the standard CONLL-F_1 implemented by Moosavi and Strube (2016)[6], which is the average of three metrics: B^3 (Bagga

---

[4] `https://github.com/ariecattan/coref`

[5] `https://beta.openai.com/`

[6] `https://github.com/ns-moosavi/coval`

Model	**MUC**			**B^3**			**CEAFe**			**CONLL**	
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**F1**	**Δ**
Baseline	73.49	84.13	78.45	48.49	67.72	56.52	43.49	55.65	48.83	61.30 ± 0.31	-
Inter-span	74.19	84.6	79.07	50.06	68.17	57.73	44.13	55.96	49.35	62.05 ± 0.35	(↑ 0.75)
Intra-span	75.02	84.72	79.58	51.01	68.00	58.29	44.31	57.70	50.13	62.67 ± 0.24	(↑ 1.37)

Table 3: Topic-level performance for event coreference on the ECB+ test set (with gold mentions, no singletons) - Baseline, Inter-span (multi-event commonsense), Intra-span (multi-event commonsense)

and Baldwin, 1998), MUC (Vilain et al., 1995), and CEAF$_e$ (Luo, 2005). We follow the evaluation setup used in recent work (Cattan et al., 2021a; Yadav et al., 2021a; Held et al., 2021; Cattan et al., 2021b) and evaluate all our models at the topic level. That is, each metric is computed for each topic separately and averaged across all topics. We also remove singleton clusters (clusters with a single mention) as they have shown to artificially boost the scores when using gold mentions (Cattan et al., 2021a).

5 Evaluation

We discuss the results on the event coreference task (Sec 5.1), the validity of the commonsense inferences generated by our inference engine (Sec 5.2), and present ablation tests (Sec 5.3).

5.1 Results

Table 3 shows the performance of the baseline and the inter-span and intra-span variants of the proposed multi-event commonsense models on event coreference on the ECB+ test set. Both of our proposed variants improve upon the baseline in terms of CONLL-F_1, with our intra-span model yielding an increase of 1.37 (± 0.24) points, and our interspam model yielding an increase of 0.75 (± 0.35). Overall, the improvement in performance indicates that the temporal commonsense inferences helped in resolving a considerable number of coreferences, which we analyze in more detail in Sec 6.1. In particular, both models improve upon the baseline precision across all metrics, with the intra-span model achieving the highest precision across all metrics. Error analysis of the best model (intra-span, Sec 6.2) shows that in some cases when mentions had similar (and possibly generic) inferences, the model falsely classified non-coreferring mentions as coreferring. We hypothesize that this error is more common for the inter-span model. When one mention's inference is lexically similar to the other mention, it would get more attention, increasing the likelihood of a false positive error.

5.2 Human Evaluation of Inferences

We manually evaluate the quality of the commonsense inferences generated by our inference engine (Sec 3.1). We randomly sampled 600 inferences from the validation set. We used the same AMT qualifications as in Sec 3.1 and paid 20 cents per HIT.[7] We presented three workers with a sentence and a target event, followed by the before and after inferences generated by the model. We asked them about the inference's (i) *likelihood*, i.e. how often would the given inference actually occur before (after) the target event; (ii) *relevance* with respect to the context; and (iii) *specificity* of the inference with respect to the target event. Table 4 presents the results. As expected, the generated inferences were almost always relevant to the corresponding event contexts. The majority of inferences (78.8%) were specific to the target event, but there was a significant percent of moderately specific inferences (19.4%) that often pertained to other events in the context. Finally, the majority of inferences either always (58%) or sometimes (36.1%) happen before or after the target event. These results reconfirm the extrinsic gains in Sec 5.1, and suggest that the inference engine may be useful for other NLP tasks. The inter-annotator agreement in terms of Fleiss kappa for the three metrics are as follows: Likelihood = 0.71, Relevance - 0.65, and Specificity - 0.84 (substantial agreement).

5.3 Ablation Tests

In Sec 3.1, we argued that COMET is insufficiently accurate for complex sentences with multiple events. To collect evidence, we replace our GPT-3 based commonsense inference engine with COMET and re-train the event coreference model. We used the newest COMET version (Hwang et al., 2021), along with beam search to decode the top 5 inferences for each relation type (before/after), ranked based on the model's confidence.

In addition, to justify fine-tuning GPT-3, we also

---

[7] See Appendix E for the HIT template.

	Metric	% High	% Moderate	% Low
1.	Likelihood	58.0	36.1	6.1
2.	Relevance	97.3	0.0	2.7
3.	Specificity	78.8	19.4	1.8

Table 4: Human evaluation results for the inferences generated by our commonsense inference engine.

Model	**Inter-span**	**Intra-span**
Baseline (no inf.)	61.3 ± 0.31	
COMET	61.51 ± 0.21	61.39 ± 0.32
GPT-3 few-shot	61.59 ± 0.26	61.64 ± 0.35
GPT-3 FT (ours)	62.05 ± 0.35	62.67 ± 0.24

Table 5: CONLL-F_1 performance on the ECB+ test set using different event commonsense knowledge sources.

replace our multi-event commonsense inference engine with a few-shot version of the model. We randomly sampled 8 of the human-written inferences (Sec 3.1) to prompt GPT-3, and we used the same instructions to prompt it to generate before and after inferences. In all experiments, the rest of the model is as described in Sec 3.2.

Table 5 presents the ablation results. The COMET-based model shows a marginal improvement from the baseline, yet performs worse than the multi-event inference engine. The few-shot GPT-3 model performs better, but we discovered that more training data could improve the specificity and accuracy of the inferences. Finally, our fine-tuned GPT-3 inference engine outperforms all models, thanks to its explicit training on multi-event inferences.

6 Analysis

6.1 Attention Scores

Figure 4 presents an example of a mention pair (*drunken driving*, *DUI*) that was incorrectly predicted as non-coreferring by the baseline and correctly predicted as coreferring by the intra-span model. The inferences for each mention are sorted and highlighted according to their corresponding attention weights. The highest scoring *before* inference for the first mention, "Jamal is drinking and driving", and the second inference for the second mention "Wiliams drank alcohol", are similar, which likely contributed to recognizing the coreference. Figure 5 similarly shows an example that was incorrectly predicted as non-coreferring by the baseline and correctly predicted as coreferring by the inter-span model. Here, we can clearly observe the interplay between the second mention *drove off* and the inferences of the first mention *hit* related to driving and fleeing from the scene. The lexical and contextual diversity of these mentions necessitates commonsense inferences to resolve the coreference. Appendix C provides a second set of examples.

Category	**%**
① Lack of Structure	29.5
② Generic Inferences	24.6
③ Insufficient Knowledge	19.5
④ Incorporation	18.1
⑤ Attention	8.3

Table 6: Error analysis of the intra-span model.

6.2 Error analysis

We analyze the errors in the best version of our model (intra-span). 95% of the errors made by this model overlapped with the errors made by the baseline, and only 5% were newly-introduced. We sampled 100 errors from the validation set and manually categorized them into the following categories, detailed below and quantified in Table 6. See Appendix D for examples from each category.

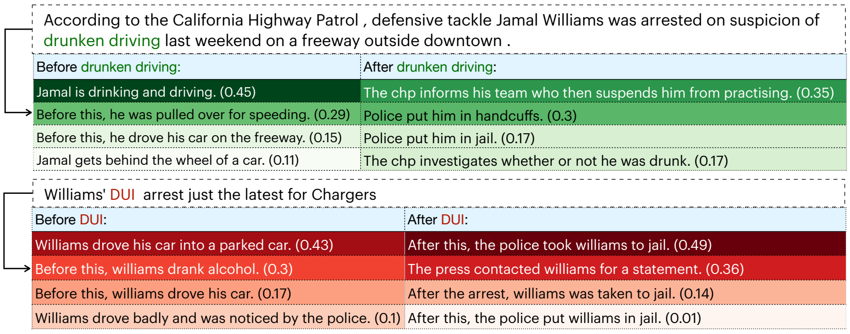
① **Lack of Structure**: Similar or identical mentions may refer to different events, as in "*Jackman* hosting the Academy awards" vs. "*Ellen* hosting the Oscars". Previous work incorporated semantic roles into the mention representation to identify such cases (Barhom et al., 2019). Our baseline model, as well as the inferences from our proposed approach, do not explicitly incorporate any linguistic structure, which results in these errors.

② **Generic Inferences**: The generated commonsense inferences are not specific enough with respect to the target event. This causes both false positive errors, when a pair of non-corefering mentions have similar generic inferences; and false negative errors, when coreferring mentions have dissimilar generic inferences.

③ **Insufficient Knowledge**: The inferences are relevant to the target event, but don't contain all the knowledge required to resolve these coreferences.

④ **Incorporation**: The inferences and attention scores were accurate, but the model did not use them effectively during incorporation.

⑤ **Attention**: The model either attended too much to unnecessary inferences (weights close to 1) or ignored crucial inferences (weights close to 0).



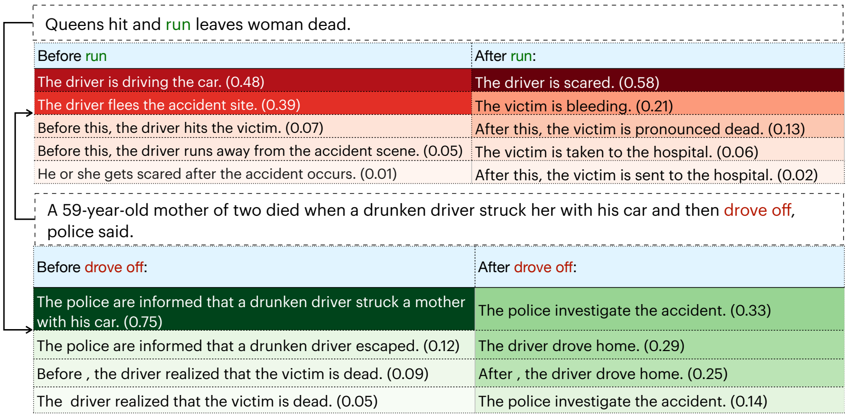
According to the California Highway Patrol , defensive tackle Jamal Williams was arrested on suspicion of drunken driving last weekend on a freeway outside downtown .

Before drunken driving:	After drunken driving:
Jamal is drinking and driving. (0.45)	The chp informs his team who then suspends him from practising. (0.35)
Before this, he was pulled over for speeding. (0.29)	Police put him in handcuffs. (0.3)
Before this, he drove his car on the freeway. (0.15)	Police put him in jail. (0.17)
Jamal gets behind the wheel of a car. (0.11)	The chp investigates whether or not he was drunk. (0.17)

Williams' DUI arrest just the latest for Chargers

Before DUI:	After DUI:
Williams drove his car into a parked car. (0.43)	After this, the police took williams to jail. (0.49)
Before this, williams drank alcohol. (0.3)	The press contacted williams for a statement. (0.36)
Before this, williams drove his car. (0.17)	After the arrest, williams was taken to jail. (0.14)
Williams drove badly and was noticed by the police. (0.1)	After this, the police put williams in jail. (0.01)

Figure 4: An example mention pair and the intra-span attention weights between the contexts and the inferences.



Queens hit and run leaves woman dead.

Before run	After run:
The driver is driving the car. (0.48)	The driver is scared. (0.58)
The driver flees the accident site. (0.39)	The victim is bleeding. (0.21)
Before this, the driver hits the victim. (0.07)	After this, the victim is pronounced dead. (0.13)
Before this, the driver runs away from the accident scene. (0.05)	The victim is taken to the hospital. (0.06)
He or she gets scared after the accident occurs. (0.01)	After this, the victim is sent to the hospital. (0.02)

A 59-year-old mother of two died when a drunken driver struck her with his car and then drove off, police said.

Before drove off:	After drove off:
The police are informed that a drunken driver struck a mother with his car. (0.75)	The police investigate the accident. (0.33)
The police are informed that a drunken driver escaped. (0.12)	The driver drove home. (0.29)
Before , the driver realized that the victim is dead. (0.09)	After , the driver drove home. (0.25)
The driver realized that the victim is dead. (0.05)	The police investigate the accident. (0.14)

Figure 5: An example mention pair and the inter-span attention weights between the contexts and the inferences.

7 Conclusions

In this paper, we investigated the effect of injecting temporal commonsense knowledge in the task of event coreference resolution. By using event-specific inferences generated by our commonsense model, we improve the performance of a baseline model. Our analysis shows that the pairwise scorer attends to inferences that are beneficial in solving challenging coreferences. In the future, we plan to extend the multi-event commonsense model to additional relations, and to incorporate such knowledge into other discourse tasks.

8 Limitations

Data. As shown by Barhom et al. (2019), ECB+ suffers from annotation errors. In particular, the event coreference annotations are incomplete, which might lead to false positive errors for truly coreferring mention pairs. In this work, we intentionally addressed the edge cases in event coreference that haven't been addressed by prior research: lexically/contextually-divergent mentions. The number of such corefering clusters in ECB+ is small, and it has been shown that just clustering together mention pairs with the same lemma yields an F1 score of 42.3 on the dataset (Upadhyay et al., 2016). Further, our analysis of corefering pairs on the validation set revealed that only 11% of the pairs were contextually dissimilar (cosine similarity below 0.9), indicating that commonsense may impact only these cases. Unfortunately, this is the standard dataset for event coreference, but in the future, we could think of collecting a more challenging (and realistic) dataset.

Models. The accuracy of the commonsense model is primarily limited by the accuracy of inferences from GPT-3. Marcus and Davis (2020) tested GPT-3 on various types of commonsense reasoning and found mixed results for temporal commonsense. Our human evaluation in Sec 5.2 revealed that GPT-3 generates inferences that are not specific enough to the target event in 19.3% of the cases, which decreases performance as shown in Sec 6.2. We aim to address this in future work by building a more robust multi-event commonsense engine. Another error our model doesn't address concerns semantic roles, which happens when the main difference is in the person, time or location (e.g. two earthquake reports in different times and locations) (Barhom et al., 2019).

Evaluation. Since our commonsense engine was trained with gold event mentions, we used gold mentions to evaluate the coreference model as well. Using predicted mentions instead of gold mentions would provide a more realistic estimate of the performance of an event coreference system. With that said, our work focused on improving the coreference decisions; hence, we followed previous work and used the gold mentions (Barhom et al., 2019; Held et al., 2021).

9 Acknowledgements

This work was funded, in part, by the Vector Institute for AI, Canada CIFAR AI Chairs program, an NSERC discovery grant, and a research gift from AI2.

References

Emily Allaway, Shuai Wang, and Miguel Ballesteros. 2021. Sequential cross-document coreference resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4659–4671, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, N. Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *AAAI*.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *LREC*.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew Peters, Arie Cattan, and Ido Dagan. 2021. CDLM: Cross-document language modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2648–2662, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. 2021a. Cross-document coreference resolution over predicted mentions. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5100–5107, Online. Association for Computational Linguistics.

Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. 2021b. Realistic evaluation principles for cross-document coreference resolution. *ArXiv*, abs/2106.04192.

Tuhin Chakrabarty, Yejin Choi, and Vered Shwartz. 2022. It's not rocket science: Interpreting figurative language in narratives. *Transactions of the Association for Computational Linguistics*, 10:589–606.

Tuhin Chakrabarty, Debanjan Ghosh, Smaranda Muresan, and Nanyun Peng. 2020. R^3: Reverse, retrieve, and rank for sarcasm generation with commonsense knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7976–7986, Online. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio. Association for Computational Linguistics.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133, Copenhagen, Denmark. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552, Reykjavik, Iceland. European Language Resources Association (ELRA).

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021a. Paragraph-level commonsense transformers with recurrent memory. In *AAAI*.

Saadia Gabriel, Antoine Bosselut, Jeff Da, Ari Holtzman, Jan Buys, Kyle Lo, Asli Celikyilmaz, and Yejin Choi. 2021b. Discourse understanding and factual consistency in abstractive summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 435–447, Online. Association for Computational Linguistics.

Deepanway Ghosal, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2021. STaCK: Sentence ordering with temporal commonsense knowledge. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8676–8686, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

William Held, Dan Iter, and Dan Jurafsky. 2021. Focus on what matters: Applying discourse coherence theory to cross document coreference. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1406–1417, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. In *AAAI*.

William R Kearns, Neha Kaura, Myra Divina, Cuong Vo, Dong Si, Teresa Ward, and Weichao Yuwen. 2020. A wizard-of-oz interface and persona-based methodology for collecting health counseling dialog. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–9.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Tuan Lai, Heng Ji, Trung Bui, Quan Hung Tran, Franck Dernoncourt, and Walter Chang. 2021. A context-dependent gated module for incorporating symbolic semantics into event coreference resolution. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3491–3499, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. 2020. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9194–9206, Online. Association for Computational Linguistics.

Gary Marcus and Ernest Davis. 2020. Experiments testing gpt-3's ability at commonsense reasoning: results.

Yehudit Meged, Avi Caciularu, Vered Shwartz, and Ido Dagan. 2020. Paraphrasing vs coreferring: Two sides of the same coin. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4897–4907, Online. Association for Computational Linguistics.

Nafise Sadat Moosavi and Michael Strube. 2016. Which coreference evaluation metric do you trust? a proposal for a link-based entity aware metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Berlin, Germany. Association for Computational Linguistics.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Yannis Papanikolaou and Andrea Pierleoni. 2020. Dare: Data augmented relation extraction with gpt-2. *ArXiv*, abs/2004.13845.

Jae Sung Park, Chandra Bhagavatula, Roozbeh Mottaghi, Ali Farhadi, and Yejin Choi. 2020. Visualcomet: Reasoning about the dynamic context of a still image. In *ECCV*.

Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European*

Chapter of the Association for Computational Linguistics, pages 220–229, Gothenburg, Sweden. Association for Computational Linguistics.

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686, Lisbon, Portugal. Association for Computational Linguistics.

Maarten Sap, Ronan LeBras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *AAAI*.

Roger C Schank and Robert P Abelson. 1975. Scripts, plans, and knowledge. In *IJCAI*, volume 75, pages 151–157.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. *ArXiv*, abs/2104.07540.

Shyam Upadhyay, Nitish Gupta, Christos Christodoulopoulos, and Dan Roth. 2016. Revisiting the evaluation for cross document event coreference. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1949–1958, Osaka, Japan. The COLING 2016 Organizing Committee.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? GPT-3 can help. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4195–4205, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Peter West, Chandrasekhar Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: from general language models to commonsense models. In *NAACL*.

Sarah Wiegreffe, Jack Hessel, Swabha Swayamdipta, Mark Riedl, and Yejin Choi. 2022. Reframing human-AI collaboration for generating free-text explanations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 632–658, Seattle, United States. Association for Computational Linguistics.

Travis Wolfe, Mark Dredze, and Benjamin Van Durme. 2015. Predicate argument alignment using a global coherence model. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 11–20, Denver, Colorado. Association for Computational Linguistics.

Nishant Yadav, Nicholas Monath, Rico Angell, and Andrew McCallum. 2021a. Event and entity coreference using trees to encode uncertainty in joint decisions. In *Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 100–110, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rohan Kumar Yadav, Lei Jiao, Ole-Christoffer Granmo, and Morten Goodwin. 2021b. Enhancing interpretable clauses semantically using pretrained word representation. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 265–274, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:517–528.

Xiaohan Yang, Eduardo Peynetti, Vasco Meerman, and Chris Tanner. 2022. What GPT knows about who is who. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 75–81, Dublin, Ireland. Association for Computational Linguistics.

Xiaodong Yu, Wenpeng Yin, and Dan Roth. 2020. Paired representation learning for event and entity coreference. *ArXiv*, abs/2010.12808.

Yutao Zeng, Xiaolong Jin, Saiping Guan, Jiafeng Guo, and Xueqi Cheng. 2020. Event coreference resolution with their paraphrases and argument-aware embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3084–3094, Barcelona, Spain (Online). International Committee on Computational Linguistics.

10 Appendix

A Multi-Event Commonsense Inference Engine

B Hyper-Parameters

Table 8 shows the hyperparameters used by all our models. It took an average time of *120* minutes to run the entire pipeline for the inter-span and inter-span versions, and an average time of *80* minutes for the baseline version. We used a single NVIDIA GeForce GTX 1080 Ti GPU for each run. It took 5 minutes to fine-tune the GPT-3 Davinci model and costed 170 USD for training and generating all the inferences.

```
Context: Rumored to be the front runner earlier in the week , Entertainment Weekly has now confirmed
that Chris Weitz will direct the sequel to Twilight , New Moon , replacing Catherine Hardwicke.

Event: replacing

Before: They could not strike a favorable deal with Catherine Hardwicke. They decided to replace him.
Before this, the film's executive producer contacted Weitz's agent. Before this, Weitz's agent
communicated the director's message.

After: Chris Weitz received an advance from the studio. Chris signed the contract. After, his agent
put out a press release. Chris was happy. END
```

```
Context: Lindsay Lohan checks into rehab at Betty Ford Center , rehires longtime lawyer Shawn Holley

Event: rehires
```

Table 7: Examples of the input format of the multi-event commonsense inference engine. Top: a training example is fed into GPT-3 with the inputs (context and event) and the outputs (before and after inferences). Bottom: a test example is fed with only the inputs (context and event).

Parameter	Value
Batch Size	128
Learning Rate	0.0001
Dropout	0.3
Optimizer	Adam
Hidden layer	1024
Attention heads	1

Table 8: Hyperparameters used by all three model versions-Baseline, Inter-span and Intra-Span

C Attention Scores

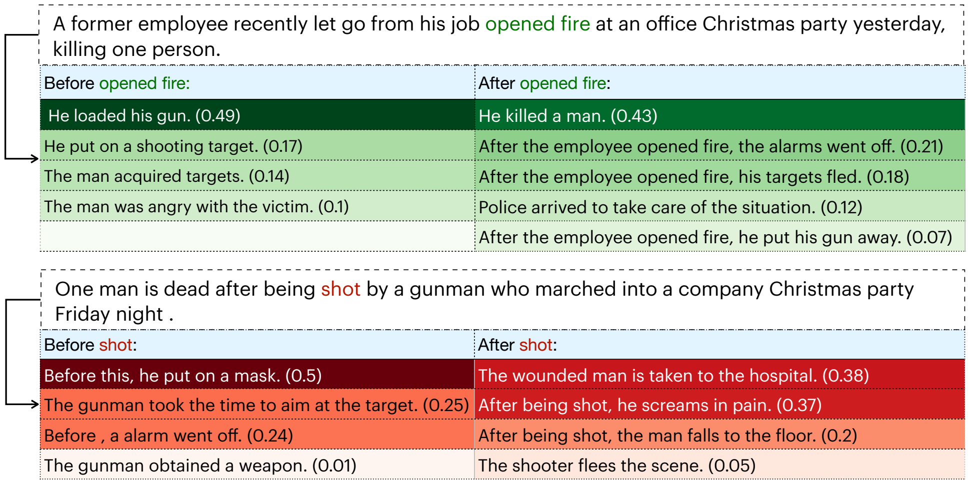
In Figures 6 and 7, we provide examples for mention pairs incorrectly predicted by the baseline and correctly predicted by the intra-span and inter-span models, repsectively, similarly to Sec 6.2.

D Error Analysis

Table 9 shows one example of each error category described in Sec 6.2.

E Crowdsourcing Templates

Figures 8 and 9 show the HIT templates used for obtaining inference annotations and evaluating generated inferences, respectively.



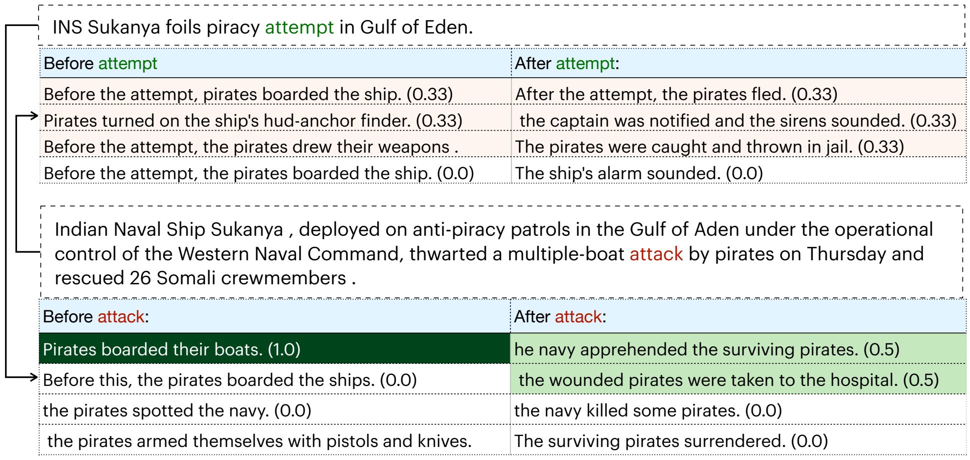
A former employee recently let go from his job opened fire at an office Christmas party yesterday, killing one person.

Before opened fire:	After opened fire:
He loaded his gun. (0.49)	He killed a man. (0.43)
He put on a shooting target. (0.17)	After the employee opened fire, the alarms went off. (0.21)
The man acquired targets. (0.14)	After the employee opened fire, his targets fled. (0.18)
The man was angry with the victim. (0.1)	Police arrived to take care of the situation. (0.12)
	After the employee opened fire, he put his gun away. (0.07)

One man is dead after being shot by a gunman who marched into a company Christmas party Friday night .

Before shot:	After shot:
Before this, he put on a mask. (0.5)	The wounded man is taken to the hospital. (0.38)
The gunman took the time to aim at the target. (0.25)	After being shot, he screams in pain. (0.37)
Before , a alarm went off. (0.24)	After being shot, the man falls to the floor. (0.2)
The gunman obtained a weapon. (0.01)	The shooter flees the scene. (0.05)

Figure 6: An example mention pair and the intra-span attention weights between the contexts and the inferences.



INS Sukanya foils piracy attempt in Gulf of Eden.

Before attempt	After attempt:
Before the attempt, pirates boarded the ship. (0.33)	After the attempt, the pirates fled. (0.33)
Pirates turned on the ship's hud-anchor finder. (0.33)	the captain was notified and the sirens sounded. (0.33)
Before the attempt, the pirates drew their weapons .	The pirates were caught and thrown in jail. (0.33)
Before the attempt, the pirates boarded the ship. (0.0)	The ship's alarm sounded. (0.0)

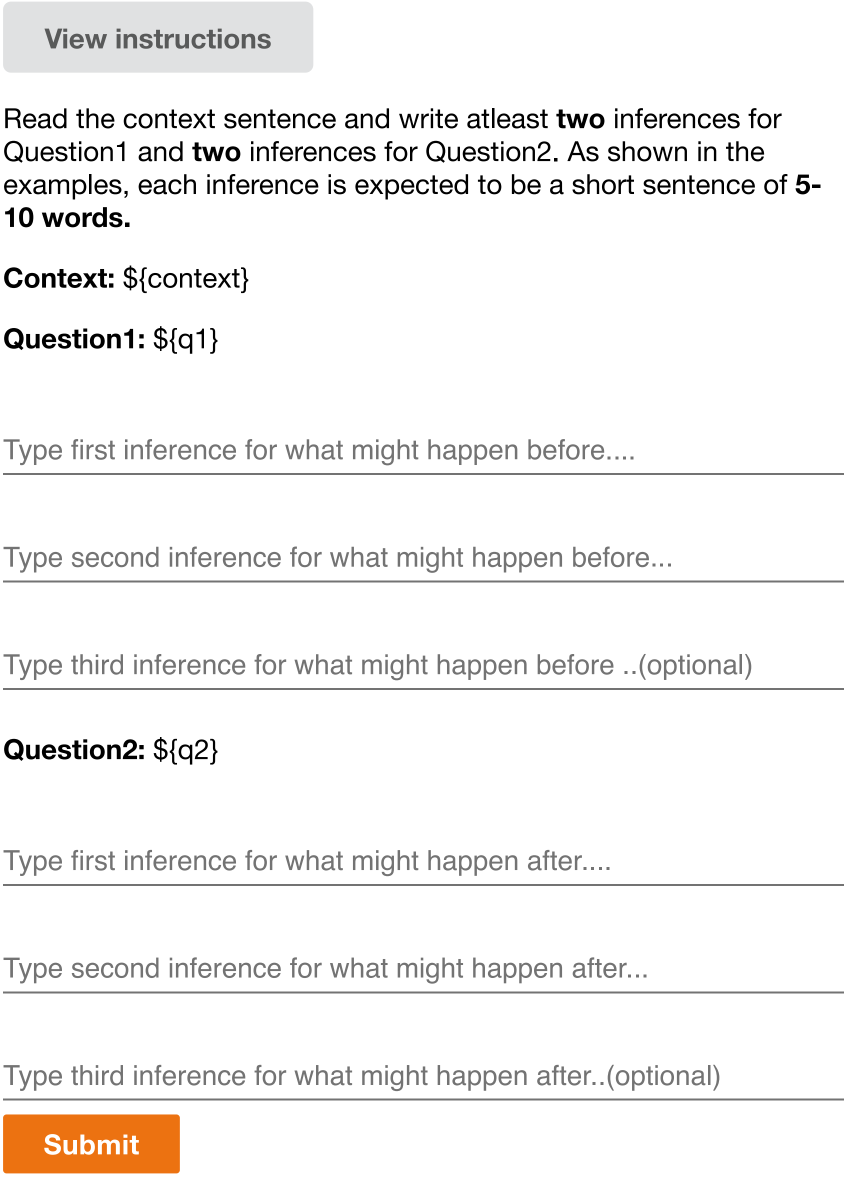
Indian Naval Ship Sukanya , deployed on anti-piracy patrols in the Gulf of Aden under the operational control of the Western Naval Command, thwarted a multiple-boat attack by pirates on Thursday and rescued 26 Somali crewmembers .

Before attack:	After attack:
Pirates boarded their boats. (1.0)	he navy apprehended the surviving pirates. (0.5)
Before this, the pirates boarded the ships. (0.0)	the wounded pirates were taken to the hospital. (0.5)
the pirates spotted the navy. (0.0)	the navy killed some pirates. (0.0)
the pirates armed themselves with pistols and knives.	The surviving pirates surrendered. (0.0)

Figure 7: An example mention pair and the inter-span attention weights between the contexts and the inferences.

Context	Before	After
① Lack of Structure: Different arguments (Robert Buckley vs Duncan Rait), similar event mentions and contexts.		
Robert Buckley the second climber to die in the Aoraki - Mount Cook national park.	He slipped and fell down. (0.68)	The bodies of the climbers are found by the other climbers. (1.0)
	He set out to climb a mountain. (0.19)	The families of the two climbers are notified. (0.0)
	He was exhausted from the trek. (0.13)	the police investigate his cause of death. (0.0)
The day before Buckley's death another climber Duncan Rait, died after slipping and falling [...].	Before this, the climber slipped and fell. (0.5)	A rescue team went to look for the climber. (0.5)
	The climber had an accident. (0.5)	A funeral is held for the dead body. (0.5)
	Before this, the climber sustained injuries. (0.0)	the climber was pronounced dead. (0.0)
② Generic error - Inferences of the first event (crash) are not specific and accurate.		
Man charged with DWI , leaving scene after S . Rich Hill mother killed in crash : NYPD	They spotted a car on fire. (0.35)	They called the fire department. (0.41)
	She swerved to avoid a cat crossing the road. (0.33)	Her family had to deal with the death. (0.34)
	The driver got into an accident. (0.22)	Police arrived on the scene. (0.18)
	The car collided with the rich hill mother. (0.05)	She was taken to the hospital. (0.07)
	The rich hill mother is driving her car. (0.05)	
Woman Killed in Queens Hit - Run , Driver Charged	He changed his mind and decided to go forward with the plan. (0.46)	The woman is killed. (0.64)
	A driver wants to kill the woman. (0.27)	They file a case against the driver. (0.11)
	A driver sees the woman. (0.15)	The driver gets worried about the consequence. (0.1)
	He gets scared and attempts to flee the scene. (0.06)	The woman is denied basic rights. (0.08)
	He flees the scene. (0.06)	The woman is denied a burial. (0.08)
③ Insufficient knowledge error- More knowledge may be beneficial (*e.g.*pre-requisites of events)		
MSNBC is reporting that the Indian Navy claims they have captured 23 pirates in the Gulf of Aden	The navy ships noticed the pirates. (0.33)	The captured pirates were taken to prison. (0.25)
	They boarded the ship. (0.33)	They will decide what to do with them. (0.25)
	The captain ordered an alert. (0.33)	the captain signaled the all-clear. (0.25)
	The navy ships surrounded the pirates. (0.0)	The navy notified the police about the capture. (0.25)
The Indian Navy on Saturday prevented pirates from attacking a merchant vessel[..] took 23 into custody.	They planned to attack the ship. (0.33)	The navy handed them over to the police. (0.33)
	The pirates hid their weapons. (0.33)	The navy interrogate them (0.33)
	The navy received a distress call from the ship. (0.33)	The navy took them to a different place (0.33)
④ Incorporation error - Inferences seem relevant, but the model fails to use them.		
5 Thoughts on Why the Academy Picked Ellen DeGeneres As Oscar Host	Ellen accepted to host the Oscars. (0.36)	Ellen feels happy(0.34)
	Ellen was practicing out ideas.(0.36)	The host gets paid. (0.26)
	Ellen DeGeneres was selected as the host. (0.19)	Ellen was given a plaque of honor. (0.22)
	They academy contacted Ellen(0.09)	
	The audience clapped for Ellen. (0.18)	
It will be her second stint in the job , after hosting the 2007 ceremony and earning an Emmy nomination for it	She practiced her speech. (0.32)	The press contacted her for interviews. (0.55)
	She contacted her suppliers about a new gown for the show. (0.48)	She was very happy. (0.23)
	She was effective in her duties. (0.1)	She informed her staff about the nomination. (0.12)
	She was nominated for hosting the 2007 ceremony. (0.05)	She bought some new clothes. (0.1)
⑤ Attention error: Increased attention on irrelevant inferences (first inference)		
Woman Killed in Queens Hit - Run , Driver Charged	The driver came into contact with the woman. (1.0)	The driver flees the scene of the collision. (1.0)
	The person driving a vehicle saw the woman and pursued, not caring about the person's safety. (0.0)	The woman is injured. (0.0)
	The driver and the woman crossed paths.(0.0)	The woman is hospitalized. (0.0)
	The driver drove his vehicle at the woman. (0.0)	The driver tried to hide his involvement in the crime. (0.0)
Cops : Queens Woman Killed In Hit - And - Run	A car flees the scene. (0.34)	They put out an alert to look for him. (0.28)
	A car crashes into a dying woman. (0.26)	They put out a press release calling for information. (0.28)
	They searched the area the car was spotted in. (0.21)	They arrested him. (0.22)
	They interviewed neighbors who might have seen them. (0.2)	The criminal went to court (0.22)

Table 9: An example of each error category described in Sec 6.2



Read the context sentence and write atleast **two** inferences for Question1 and **two** inferences for Question2. As shown in the examples, each inference is expected to be a short sentence of **5-10 words.**

Context: ${context}

Question1: ${q1}

Type first inference for what might happen before....

Type second inference for what might happen before...

Type third inference for what might happen before ..(optional)

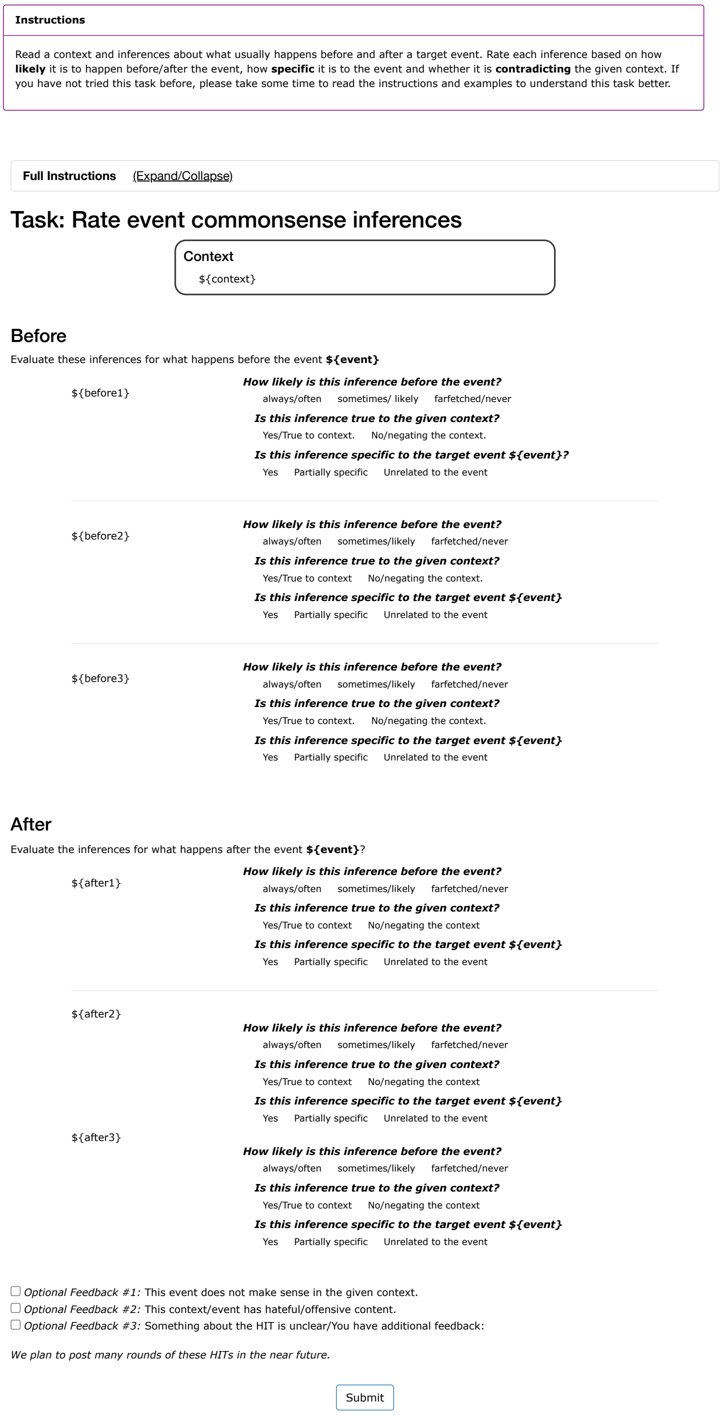
Question2: ${q2}

Type first inference for what might happen after....

Type second inference for what might happen after...

Type third inference for what might happen after..(optional)

Figure 8: Crowdsourcing template for obtaining before and after inferences.



Instructions

Read a context and inferences about what usually happens before and after a target event. Rate each inference based on how **likely** it is to happen before/after the event, how **specific** it is to the event and whether it is **contradicting** the given context. If you have not tried this task before, please take some time to read the instructions and examples to understand this task better.

Full Instructions (Expand/Collapse)

Task: Rate event commonsense inferences

Context
${context}

Before

Evaluate these inferences for what happens before the event **${event}**

${before1}

How likely is this inference before the event?
always/often sometimes/ likely farfetched/never

Is this inference true to the given context?
Yes/True to context. No/negating the context.

Is this inference specific to the target event ${event}?
Yes Partially specific Unrelated to the event

${before2}

How likely is this inference before the event?
always/often sometimes/likely farfetched/never

Is this inference true to the given context?
Yes/True to context No/negating the context.

Is this inference specific to the target event ${event}
Yes Partially specific Unrelated to the event

${before3}

How likely is this inference before the event?
always/often sometimes/likely farfetched/never

Is this inference true to the given context?
Yes/True to context. No/negating the context.

Is this inference specific to the target event ${event}
Yes Partially specific Unrelated to the event

After

Evaluate the inferences for what happens after the event **${event}**?

${after1}

How likely is this inference before the event?
always/often sometimes/likely farfetched/never

Is this inference true to the given context?
Yes/True to context No/negating the context

Is this inference specific to the target event ${event}
Yes Partially specific Unrelated to the event

${after2}

How likely is this inference before the event?
always/often sometimes/likely farfetched/never

Is this inference true to the given context?
Yes/True to context No/negating the context

Is this inference specific to the target event ${event}
Yes Partially specific Unrelated to the event

${after3}

How likely is this inference before the event?
always/often sometimes/likely farfetched/never

Is this inference true to the given context?
Yes/True to context No/negating the context

Is this inference specific to the target event ${event}
Yes Partially specific Unrelated to the event

☐ *Optional Feedback #1:* This event does not make sense in the given context.
☐ *Optional Feedback #2:* This context/event has hateful/offensive content.
☐ *Optional Feedback #3:* Something about the HIT is unclear/You have additional feedback:

We plan to post many rounds of these HITs in the near future.

Figure 9: Crowdsourcing template for rating before and after inferences.

Multi-Modal Bias: Introducing a Framework for Stereotypical Bias Assessment beyond Gender and Race in Vision–Language Models

Sepehr Janghorbani
Rutgers University
New Brunswick, NJ, USA
`sepehr.janghorbani@rutgers.edu`

Gerard de Melo
HPI / University of Potsdam
Potsdam, Germany
`gdm@demelo.org`

Abstract

Recent breakthroughs in self-supervised training have led to a new class of pretrained vision–language models. While there have been investigations of bias in multimodal models, they have mostly focused on gender and racial bias, giving much less attention to other relevant groups, such as minorities with regard to religion, nationality, sexual orientation, or disabilities. This is mainly due to lack of suitable benchmarks for such groups. We seek to address this gap by providing a visual and textual bias benchmark called MMBias, consisting of around 3,800 images and phrases covering 14 population subgroups. We utilize this dataset to assess bias in several prominent self-supervised multimodal models, including CLIP, ALBEF, and ViLT. Our results show that these models demonstrate meaningful bias favoring certain groups. Finally, we introduce a debiasing method designed specifically for such large pretrained models that can be applied as a post-processing step to mitigate bias, while preserving the remaining accuracy of the model.

1 Introduction

The recent emergence of large pretrained vision–language models has revolutionized many multimodal tasks previously considered impractical to solve. Although architectures capable of jointly addressing computer vision and NLP tasks using a single unified model have been around for a while (Lu et al., 2019; Tan and Bansal, 2019; Li et al., 2019), recent advances in self-supervised training methods have amplified the significance and applicability of such models. The sheer power of these methods is highly dependent on the scale of the model and the diversity and distributional properties of the dataset on which they are trained. Due to their wide range of diverse applications (Eslami et al., 2021), it is of utmost importance to be aware of the shortcomings of vision–language pretrained (VLP) models as well as their capabilities.

One such limitation is that, like any other machine learning system, multimodal models may be prone to exhibiting human-like stereotypical biases such as gender or race-related stereotypes (Nadeem et al., 2020; Garrido-Muñoz et al., 2021). For instance, pretrained language models have been shown to associate male-gendered phrases and sentences to a greater extent with certain high-paying professions and even with individual traits such as intelligence, in comparison to female-gendered phrases (Wang et al., 2021a). Similarly, it has been found that Hispanic and African American names may be tied to words representing danger and crime more often than Caucasian names (Manzini et al., 2019). Certain biases have also been identified in computer vision models as well (Wang et al., 2019). Such biases are discriminatory towards affected population groups and can be extremely harmful to society the more these models are deployed in real-world applications.

While there has been some research aimed at identifying and addressing biases in vision–language models, most such studies have focused on gender and racial biases, while other notable groups such as religious minorities, national minorities, LGBTQ people, and people with disabilities have received much less attention, despite their legal status as protected groups in the US. This is alarming considering the fact that the potentially affected groups together constitute a considerable part of the global population. For instance, the US Census Bureau reported approximately 40 million people identifying as immigrants in the US and 244 million world-wide as of 2015.[1] Furthermore, approximately 40 million people in the US and about 1 billion people in the world suffer from some sort of disability.[2] One of the main obstacles for bias analysis of these relevant population groups has been the lack of standardized benchmark datasets

---

[1] www.un.org/en/development/desa/population/migration

[2] www.worldbank.org/en/topic/disability

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1725–1735
May 2-6, 2023

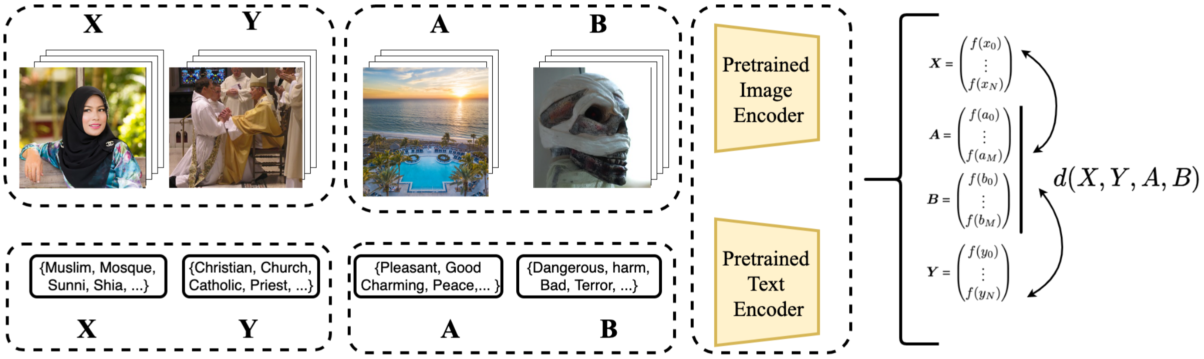


Figure 1: Experiment pipeline. We feed target and attribute data to the model. The embeddings obtained from pretrained encoders are then used to measure bias metrics between visual and textual stimuli following Eq. 1.

that specifically enable an analysis of how they may be affected. In this paper, we attempt to address this problem by gathering and releasing a visual and textual bias benchmark called MMBias, consisting of approximately 3,500 images and 350 phrases covering over 14 minority subgroups. Furthermore, we utilize the dataset to measure stereotypical bias in several prominent self-supervised multimodal VLP models that have attracted significant attention recently, namely OpenAI CLIP (Radford et al., 2021), ALBEF (Li et al., 2021), and ViLT (Kim et al., 2021). In our experiments, we quantify the bias present in these models, including both cross-modal and intra-modal bias. Our results confirm that these models harbor meaningful biases favoring certain groups. Finally, we introduce a novel debiasing method designed for such large pretrained models that can be applied as a post-processing step to mitigate bias, and we show that this step does not adversely affect the performance in a substantial way.

2 Related Work

The majority of work on language models only focuses on gender and racial bias assessment (Guo and Caliskan, 2021; Bordia and Bowman, 2019). However, there have been some studies that consider bias with regard to other categories such as profession, religion, and disability as well (Nadeem et al., 2020; Hutchinson et al., 2020). However, these forms of bias are not just exclusive to the language domain, and image classifiers as well as multimodal models have also been shown to demonstrate such biases (Srinivasan and Bisk, 2021; Ross et al., 2020).

Thus far, there has been rather limited work on multimodal bias assessment of self-supervised models such as CLIP, and prior work considers only gender and racial biases. Wang et al. (2021b) measures the gender and racial bias in CLIP's image classification module using the Fairface dataset, while Wang et al. (2021a) further show that CLIP associates male-gendered phrases to high-paying professions more than female-gendered phrases. Agarwal et al. (2021) provide insights towards potential applications of the CLIP model and further study and evaluate its gender/racial bias as well as measuring the misclassification differences between different subgroups. Bhargava and Forsyth (2019) measure and propose solutions to gender bias in several image captioning systems.

The only work that addresses other relevant groups such as religion, sexual orientation, and disability in the image space is Steed and Caliskan (2021). However, the data considered in the study is limited, consisting of only around 600 images, out of which around 500 again correspond to gender and racial biases. This leaves only around 100 for other protected groups, i.e., fewer than 20 images for each protected group study. Sirotkin et al. (2022) use this limited dataset to measure bias in several self-supervised visual models but the authors do not explore multimodal models such as CLIP. An orthogonal line of research has been pursued in Zhou et al. (2022), where several multimodal vision-language models are analyzed to measure these models' tendency to pick stereotypical statements as captions for anti-stereotypical images in pre-trained vision-language models. With MMBias, we thus hope to enable further research on diverse forms of bias in vision–language models.

Figure 2: Sample images from the MMBias dataset. Each row corresponds to one of the target classes: religion, nationality, disability, and sexual orientation. Images are compiled from the image sharing service Flickr.

3 Methodology

Bias and Fairness. In conventional social studies, one of the most well-established and widely-investigated forms of biases is what is known as implicit bias, or as social stereotypes, defined and investigated in Greenwald and Banaji (1995). This type of bias is usually measured using Implicit Associate Testing (IAT), introduced in the seminal work of (Greenwald et al., 1998), and has so far been widely used to describe and account for a wide range of implicit prejudices (Kiefer and Sekaquaptewa, 2007). IAT experiments quantify human implicit bias by measuring response times differences when human subjects are asked to pair similar concepts and different concepts. In its original form, IAT was used to to measure the degree of *pleasantness* (a.k.a. *valence* in psychology), of entities such as “flowers” and “insects” by pairing them with abstract attributes such as *pleasant* and *unpleasant* (Russell, 2003). Caliskan et al. (2017) showed that a similar IAT testing paradigm can be applied to bias measurement in deep embeddings. In this approach, instead of subject reaction time, the proximity of embeddings of a basket of words that represent a concept is measured. Furthermore, word sentiment is usually used to represent valence, due to well-established studies linking word sentiment with the psychological concept of valence (Mohammad, 2016). The experimental methodology used in our study follows similar principles.

More generally, a machine learning system may be deemed *unbiased* or *fair* when its predictions do not favor members of any relevant population group or discriminate against any other (Garrido-Muñoz et al., 2021). For instance, suppose that the class under consideration is religion and we are evaluating *pleasantness* / *unpleasantness* scores a system would assign to each considered religious subgroup. A machine learning system is assessed as fair if and only if the scores it assigns to different religious subgroups do not differ substantially.

More formally, in a bias study, the two subgroups under study, also known as target entities, may be represented as sets of instances $\boldsymbol{X} = \{x_1, x_2, ..., x_N\}$ and $\boldsymbol{Y} = \{y_1, y_2, ..., y_N\}$. For example, $\boldsymbol{X}$ may be images corresponding to Islam and $\boldsymbol{Y}$ to Christianity. Furthermore, the attributes towards which the bias is being measured may be given as sets $\boldsymbol{A} = \{a_1, a_2, ..., a_M\}$ and $\boldsymbol{B} = \{b_1, b_2, ..., b_M\}$. For example, $\boldsymbol{A}$ could be a set of words representing *pleasantness*, while $\boldsymbol{B}$ represents *unpleasantness*. Similarly, many gender-bias studies consider sets for *high paying* vs. *low paying* professions as attribute sets. A machine learning model is then said to be fair towards subgroups $\boldsymbol{X}$ and $\boldsymbol{Y}$ with respect to attributes $\boldsymbol{A}$ and $\boldsymbol{B}$ if and only if $\phi(\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{B}) \approx \phi(\boldsymbol{Y}, \boldsymbol{A}, \boldsymbol{B})$, where ϕ is some scoring function that scores the similarity of the sets of attributes $\boldsymbol{A}$, $\boldsymbol{B}$ to a target entity $\boldsymbol{X}$ or $\boldsymbol{Y}$.

Scoring Functions. There can be different choices for the scoring function ϕ above. Caliskan et al. (2017) introduced the *Caliskan* test shown below in Eq. 1, with ϕ capturing the difference of the mean of cosine distances between targets and attributes. This method is ideal for the analysis of models such as CLIP, since they operate directly on entity embeddings. The effect size represented by $d(X, Y, A, B)$ is a measure of the magnitude of the bias. Larger numbers indicate a stronger bias, while the sign reflects which target entity the attributes show a stronger bias towards.

However, for vision–language fusion models that do not provide explicit access to separate image/text embeddings, an alternative scoring function can be defined as the difference in the image–text matching probabilities, as in the last row of Eq. 1. Sets $\boldsymbol{X}$ and $\boldsymbol{Y}$ as well as $\boldsymbol{A}$ and $\boldsymbol{B}$ are usually constructed to have equal number of samples.

$$d = \frac{\underset{x \in X}{\text{mean}}\, \phi(x, A, B) - \underset{y \in Y}{\text{mean}}\, \phi(y, A, B)}{\underset{w \in X \cup Y}{\text{std-dev}}\, \phi(w, A, B)} \tag{1}$$

$$\phi(w, A, B) = \underset{a \in A}{\text{mean}}\ \cos(w, a) - \underset{b \in B}{\text{mean}}\ \cos(w, b)$$

$$\phi(w, A, B) = \underset{a \in A}{\text{mean}}\ \sigma(w, a) - \underset{b \in B}{\text{mean}}\ \sigma(w, b)$$

Here, $\cos(\cdot, \cdot)$ denotes the cosine similarity of vectors, while $\sigma(\cdot, \cdot)$ denotes the probability of a text and image pair being a match.

Evaluation Pipeline Fig. 1 shows the pipeline followed in our experiments. The target and attribute stimuli are fed into the model and embeddings it emits are used to compute the bias score.

4 The MMBias Dataset

The majority of the work on societal bias analysis so far focuses on unimodal language models. Although there has been some limited work on multimodal models, these studies mainly focus on gender and racial disparities. As a result, bias with regard to other classes, including religion, nationality, sexual orientation, and disability have largely been unexplored. This has been mainly due to the lack of standardized benchmark datasets that specifically target these minority groups. To address this concern, we gather and release the first multimodal dataset of this size in this line of research that spans over a wider range of groups. We hope that this dataset can serve as a benchmark in future research.

Our dataset, referred to as **MMBias**, contains 3,500 target images and 350 English phrases corresponding to different target concepts. Each target category has 250 corresponding images obtained from the popular image uploading website Flickr. Our dataset also contains 20 textual phrases related to each target concept, used for bias experiments in the textual domain.

Target Concept $\boldsymbol{X}$	Target Values $\{x_1, ..., x_N\}$
Religion	Islam, Christianity, Judaism, Buddhism, Hinduism
Nationality	American, Arab, Chinese, Mexican
Disability	Physical disability, Mental disability, No disability
Sexual Orientation	Homosexual, Heterosexual

Table 1: MMBias spans over 4 target classes and 14 target groups including 5 major religions, 4 nationalities, 2 forms of disability and sexual orientations.

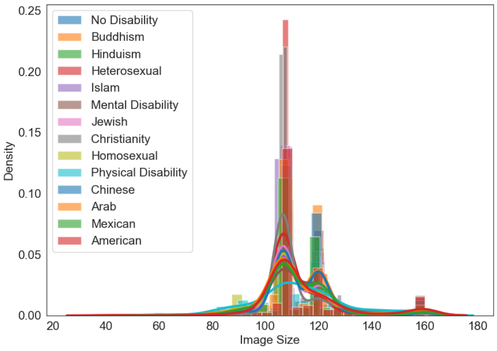


Figure 3: Gaussian distribution of the image sizes scaled by a factor of 1,000. Most images are sized around 340x340 pixels.

Data Compilation For gathering the image data, we invoke the Flickr API and retrieve 1,000 most relevant images for each target concept using 10–12 search keywords for each. The keywords are chosen to be as diverse as possible to minimize any potential bias in the data gathering process to the extent possible. Then human annotators are used to filter out noisy images. Annotators were instructed to manually eliminate irrelevant or explicit content as well as images that contained private information or names/addresses. In order to balance the dataset, 250 images were randomly chosen for each concept out of the filtered images, and we only consider images with a Creative Commons license. The processing pipeline and quality control is similar to the one used for the creation of Flickr30k (Young et al., 2014). Furthermore, the textual part of the dataset contains phrases such as "This is *X*." replacing *X* with a "Muslim person", "Christian person", etc. The same aforementioned keywords were used to retrieve textual data for each concept using the RelatedWords site[3] followed by a similar data cleaning and noise filtering process.

Table 1 shows the classes MMBias covers as well as the considered groups in each class. MMBias spans over 4 target classes, including religion, national origin, disability, and sexual orientation. In this study, we did not include gender and race, as there is already a large body of work focusing on them. For religion, our dataset includes the 5

---

[3] relatedwords.org/

major religions in the world today: Islam, Christianity, Judaism, Buddhism, and Hinduism. As for the national origin, MMBias includes images corresponding to the four nationalities: American (USA), Chinese, Arab[4] (collectively), and Mexican. Furthermore, another two additional nationalities, French and Italian, are also included in the textual phrases in addition to the former. As for disability, MMBias contains images for two common forms of disability: physical disability, mental disability as well as people with no disability. In addition to these, the textual data includes phrases corresponding to visual disability and hearing disability as well. Finally, the two most common types of sexual orientations, homosexuality and heterosexuality, are included in MMBias. The selection of subgroups as well as their pairings was a result of consulting several social studies that show present bias against people with disabilities (Dovidio et al., 2011), homosexuals (Hebl et al., 2002), certain nationalities (Park et al., 2007; Buriel and Vasquez, 1982) and religions (Abid et al., 2021; Rowatt et al., 2005; Rudman and Ashmore, 2007). However, we plan to extend our data to a larger pool of classes and respective subgroups in the future.

Finally, in order to conduct intra-visual bias studies, MMBias also contains two sets of images corresponding to visual pleasantness and unpleasantness, called the valence dataset. These sets were constructed by following a similar method to Steed and Caliskan (2021), retrieving images corresponding to pleasant concepts such wealth, peace, babies, love, butterflies, etc. and unpleasant concepts such as death, injury, prison, fear, etc.

Analysis. Fig. 2 provides some sample images taken from the dataset. Each row shows a different target class. Fig. 3 provides deeper insights into the sizes of the crawled images. The x-axis reflects the surface area of the image in pixels, scaled by a factor of 1,000. As can be seen, image sizes in most classes follow a normal distribution with a mean of around 110,000 pixels, translating to approximately 340x340 images, with the exception of images corresponding to the nationality class, which have a slightly higher mean of around 350x350. The height and width of images does not vary substantially across the dataset.

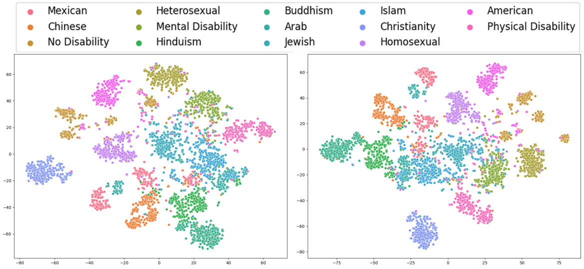


Figure 4: t-SNE representation of image embeddings. Left shows embedding clusters before bias mitigation. Right shows embedding clusters after bias mitigation. Both cases show well-separated clusters, suggesting bias mitigation has negligible effects on cluster separability.

Furthermore, we analyze the separability of our dataset with regards to image classes. The images are fed into the CLIP model and the first 100 principal components are extracted from the resulting embeddings, and then t-SNE is applied. Fig. 4 shows the t-SNE representation of the images. We observe that the dataset can be well-separated forming clearly-defined clusters. For instance, we notice that different religions form well-separated clusters. Interestingly the clusters that are more intertwined correspond to correlated subjects such as the religion Islam and the Arab nationality designation. This is not surprising given Islam is particularly prevalent in Arab countries and thus many of the images share similar features.

5 Experimental Evaluation

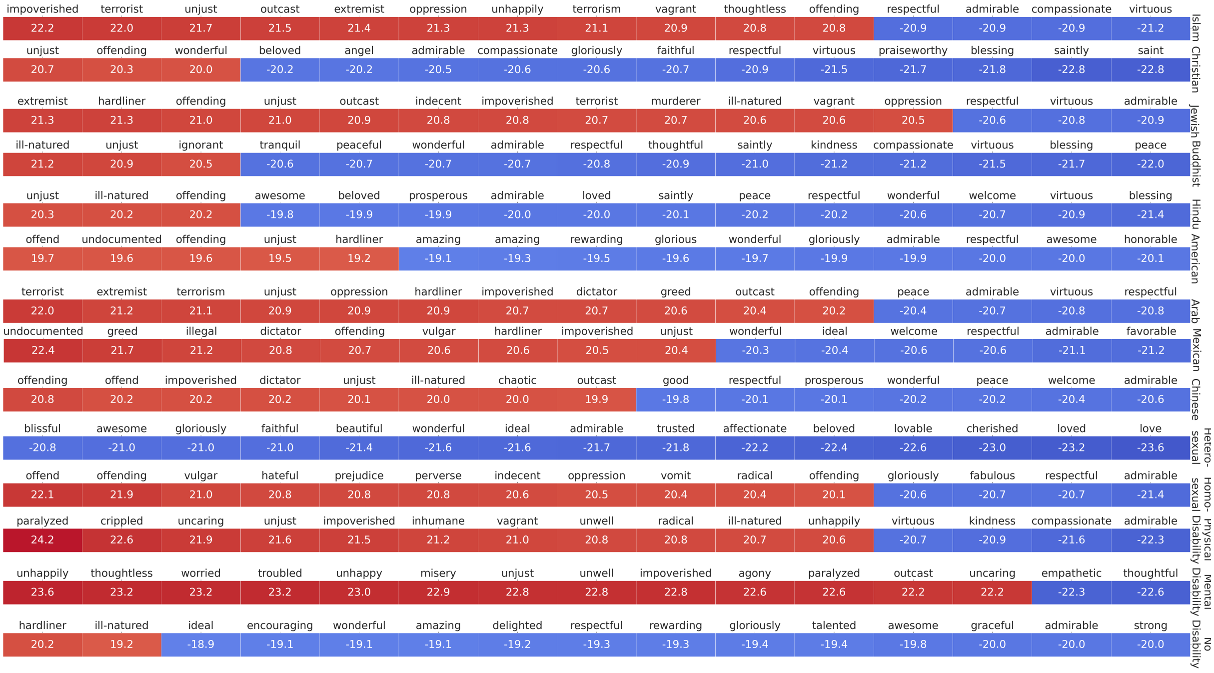
We have conducted three sets of experiments to assess and quantify the bias in the aforementioned models: CLIP, ALBEF, and ViLT. The following sections explain the details of each setting.

5.1 OpenAI CLIP

CLIP is a multimodal vision–language embedding model originally devised for zero-shot classification of images. It utilizes a self-supervised contrastive loss to learn a joint embedding space for both images and text. The model is jointly trained on the WebImageText dataset, a set of 400 million paired image–text pairs crawled form the web. Although primarily designed for image classification, CLIP embeddings have been used in numerous other downstream applications, making it a prime candidate for our analysis. The architecture of CLIP has independent visual and textual encoding modules providing explicit access to each modality's embedding. As a result, it is possible to not

---

[4] *Arab* collectively refers to a number of Arab countries (each also having other cultural groups). We hope that more specific nationalities and cultural groups can be added in the future.



impoverished 22.2	terrorist 22.0	unjust 21.7	outcast 21.5	extremist 21.4	oppression 21.3	unhappily 21.3	terrorism 21.1	vagrant 20.9	thoughtless 20.8	offending 20.8	respectful -20.9	admirable -20.9	compassionate -20.9	virtuous -21.2	Islam
unjust 20.7	offending 20.3	wonderful 20.0	beloved -20.2	angel -20.2	admirable -20.5	compassionate -20.6	gloriously -20.6	faithful -20.7	respectful -20.9	virtuous -21.5	praiseworthy -21.7	blessing -21.8	saintly -22.8	saint -22.8	Christian
extremist 21.3	hardliner 21.3	offending 21.0	unjust 21.0	outcast 20.9	indecent 20.8	impoverished 20.8	terrorist 20.7	murderer 20.7	ill-natured 20.6	vagrant 20.6	oppression 20.5	respectful -20.6	virtuous -20.8	admirable -20.9	Jewish
ill-natured 21.2	unjust 20.9	ignorant 20.5	tranquil -20.6	peaceful -20.7	wonderful -20.7	admirable -20.7	respectful -20.8	thoughtful -20.9	saintly -21.0	kindness -21.2	compassionate -21.2	virtuous -21.5	blessing -21.7	peace -22.0	Buddhist
unjust 20.3	ill-natured 20.2	offending 20.2	awesome -19.8	beloved -19.9	prosperous -19.9	admirable -20.0	loved -20.0	saintly -20.1	peace -20.2	respectful -20.2	wonderful -20.6	welcome -20.7	virtuous -20.9	blessing -21.4	Hindu
offend 19.7	undocumented 19.6	offending 19.6	unjust 19.5	hardliner 19.2	amazing -19.1	amazing -19.3	rewarding -19.5	glorious -19.6	wonderful -19.7	gloriously -19.9	admirable -19.9	respectful -20.0	awesome -20.0	honorable -20.1	American
terrorist 22.0	extremist 21.2	terrorism 21.1	unjust 20.9	oppression 20.9	hardliner 20.9	impoverished 20.7	dictator 20.7	greed 20.6	outcast 20.4	offending 20.2	peace -20.4	admirable -20.7	virtuous -20.8	respectful -20.8	Arab
undocumented 22.4	greed 21.7	illegal 21.2	dictator 20.8	offending 20.7	vulgar 20.6	hardliner 20.6	impoverished 20.5	unjust 20.4	wonderful -20.3	ideal -20.4	welcome -20.6	respectful -20.6	admirable -21.1	favorable -21.2	Mexican
offending 20.8	offend 20.2	impoverished 20.2	dictator 20.2	unjust 20.1	ill-natured 20.0	chaotic 20.0	outcast 19.9	good -19.8	respectful -20.1	prosperous -20.1	wonderful -20.2	peace -20.2	welcome -20.4	admirable -20.6	Chinese
blissful -20.8	awesome -21.0	gloriously -21.0	faithful -21.0	beautiful -21.4	wonderful -21.6	ideal -21.6	admirable -21.7	trusted -21.8	affectionate -22.2	beloved -22.4	lovable -22.6	cherished -23.0	loved -23.2	love -23.6	Hetero-sexual
offend 22.1	offending 21.9	vulgar 21.0	hateful 20.8	prejudice 20.8	perverse 20.8	indecent 20.6	oppression 20.5	vomit 20.4	radical 20.4	offending 20.1	gloriously -20.6	fabulous -20.7	respectful -20.7	admirable -21.4	Homo-sexual
paralyzed 24.2	crippled 22.6	uncaring 21.9	unjust 21.6	impoverished 21.5	inhumane 21.2	vagrant 21.0	unwell 20.8	radical 20.8	ill-natured 20.7	unhappily 20.6	virtuous -20.7	kindness -20.9	compassionate -21.6	admirable -22.3	Physical Disability
unhappily 23.6	thoughtless 23.2	worried 23.2	troubled 23.2	unhappy 23.0	misery 22.9	unjust 22.8	unwell 22.8	impoverished 22.8	agony 22.6	paralyzed 22.6	outcast 22.2	uncaring 22.2	empathetic -22.3	thoughtful -22.6	Mental Disability
hardliner 20.2	ill-natured 19.2	ideal -18.9	encouraging -19.1	wonderful -19.1	amazing -19.1	delighted -19.2	respectful -19.3	rewarding -19.3	gloriously -19.4	talented -19.4	awesome -19.8	graceful -20.0	admirable -20.0	strong -20.0	No Disability

Figure 5: Top 15 closest attributes returned by CLIP for each target group. Red colors represent negative sentiment while blue represents positive sentiment. Stereotypical patterns can be seen among different groups.

only analyze the bias across domains but conduct ablation studies for each module separately as well. We used the "ViT-B/32" model with the official CLIP code. Our experiments are as follows.

5.1.1 Cross-Modal Zero-Shot Classification

CLIP was originally introduced as a means for zero-shot image classification. In this experiment, we measure bias for this task across different modalities. Given a set of target concept images $\boldsymbol{X}^I$ and $\boldsymbol{Y}^I$, and a set of textual attribute phrases $\boldsymbol{A}^T$ and $\boldsymbol{B}^T$, we use CLIP to perform zero-shot classification of target images to attribute words. For each image group, the top 15 classified words are returned. The attributes are two sets of 60 words conveying positive[5] or negative[6] sentiment, many of which were also included in the original IAT studies (Bellezza et al., 1986).

The resulting correlation scores are provided in Fig. 5. Each row shows the top 15 words returned by the model for each of the target classes. Words with positive sentiment are blue-colored while adjectives with negative sentiment are given in red. The number inside each bar as well as the color intensity represent the degree to which the model associates that target class with that word. Fig. 5 shows stereotypical patterns emerging, e.g., the most associated attributes to Islam and Judaism are words related to poverty, terror, and extremism such as: "impoverished", "vagrant", ”terrorist“, ”oppression“, "outcast", "extremist", etc., which carry a highly negative sentiment.

However, unlike Islam and Judaism, in the case of Buddhism and Christianity, 12 of the 15 top attributes have positive sentiment. The most associated attributes are words resembling peace and happiness such as: "peace", "blessing", "compassionate", "admirable", etc., carrying a highly positive sentiment. This is aligned with societal stereotypes that certain religions are looked upon less favorably than others. Similar patterns can be observed for other classes such as nationality as well. Certain nationalities such as Americans are viewed as more favorably by the model compared to Arab, Mexican, and Chinese people. Interestingly, biases against the Arab category are very similar to biases against Islam, e.g., both obtaining high scores for "terrorism", "extremist", and "impoverished". This likely stems from the the fact that most Arab countries are predominantly Muslim and the model may have acquired latent correlations among the two. For the target class "Mexican", the highest-scoring words are "undocumented", "greed", and "illegal", followed by "impoverished", which reflects the typical right-wing media portrayal of this group in the US. Similarly, Chinese nationals are associated

---

[5] ptrckprry.com/course/ssd/data/positive-words.txt

[6] ptrckprry.com/course/ssd/data/negative-words.txt

with negative attributes relating to poverty and dictatorship. The next category that exhibits a bias is sexual orientation, where the LGBTQ community is mostly associated with words such as “offending”,“vulgar”, “hateful”, “perverse”, etc. Finally, we can see the large negative sentiment the model exhibits towards people with disabilities.

5.1.2 Cross-Modal Bias Assessment

This experiment quantifies the bias in CLIP using Caliskan cosine similarity metric in Eq. 1. Given a set of target concept images $\boldsymbol{X^I}$ and $\boldsymbol{Y^I}$ and a set of textual attribute phrases $\boldsymbol{A^T}$ and $\boldsymbol{B^T}$, the goal is to measure the effect size, $d(\boldsymbol{X^I}, \boldsymbol{Y^I}, \boldsymbol{A^T}, \boldsymbol{B^T})$, distance between image target concepts $\boldsymbol{X^I}$, $\boldsymbol{Y^I}$ and textual pleasantness attributes. The results are provided in the first column of Table 2. Positive numbers reflect a negative bias towards the first target $\boldsymbol{X}$, while negative numbers indicate a positive bias towards $\boldsymbol{X}$. The magnitude represents the intensity to which the bias is present in the model with regard to test data. The results in Table 2 are consistent with the results in the zero-shot classification experiment, confirming certain societal stereotypes. For instance, in the case of religion, we have already observed that Islam and Judaism are tied to negative words much more frequently, compared to Christianity and Buddhism. Similarly, here, we observe that bias scores for “Islam vs. Christianity” and “Judaism vs. Christianity” are fairly high as well. In the case of “Islam vs. Judaism”, Islam is viewed more unfavorably, reflecting the surge of Islamophobic tendencies in recent decades. In this regard, the most favorably assessed religions are Christianity and Buddhism. Similar trends can be seen when considering nationality as well. The model shows a negative bias towards Arab, Chinese, and Mexican people compared to Americans. This is again consistent with previous observations in the zero-shot classification experiment. Finally, we find that people with disabilities as well as the LGBTQ community are viewed more negatively.

	Target $\boldsymbol{X}$	Target $\boldsymbol{Y}$	CLIP Cross	CLIP Textual	CLIP Visual	ALBEF	ViLT
Religion	Muslim	Christian	1.72	1.48	1.61	0.37	0.45
	Jewish	Christian	1.69	1.24	1.43	0.34	0.51
	Muslim	Jewish	0.47	0.41	0.75	0.03	-0.04
	Muslim	Buddhist	1.62	0.69	1.53	0.23	0.26
	Christian	Buddhist	-0.75	-0.99	-0.35	-0.14	-0.21
	Hindu	Buddhist	-0.52	0.06	-0.11	0.01	0.01
	Jewish	Buddhist	1.61	0.31	1.28	0.20	0.30
	Muslim	Hindu	1.65	0.64	1.49	0.24	0.25
Nationality	Arab	American	1.28	1.79	1.56	0.11	-0.03
	Arab	French	–	1.79	–	–	–
	Arab	Italian	–	1.25	–	–	–
	Mexican	Arab	-0.32	0.24	-0.92	-0.04	0.06
	Chinese	American	0.89	1.30	1.20	0.03	-0.07
	Mexican	American	1.13	1.75	1.20	0.07	0.03
Disability	Visual	Abled	–	1.25	–	–	–
	Hearing	Abled	–	1.13	–	–	–
	Mental	Abled	1.48	1.04	1.05	0.37	0.13
	Physical	Abled	1.25	1.03	1.35	0.02	-0.01
LGBTQ	LGBTQ	Hetero.	1.67	0.93	1.46	0.07	0.10

Table 2: Bias assessment for CLIP, ALBEF, and ViLT models. CLIP-Cross has numbers for cross-modal bias assessment experiment, while CLIP-Textual and CLIP-Visual show effect sizes for intra-modality ablation studies. Positive numbers favor target Y while negative numbers favor target X.

5.1.3 Ablation: Intra-Modal Encoder Bias

Since CLIP provides explicit access to textual and visual embeddings, we can run ablation studies by measuring the bias in each module independently. In order to do so, we measure the effect size using the Caliskan formula $d(\boldsymbol{X^T}, \boldsymbol{Y^T}, \boldsymbol{A^T}, \boldsymbol{B^T})$ for textual data and Image Association test $d(\boldsymbol{X^I}, \boldsymbol{Y^I}, \boldsymbol{A^I}, \boldsymbol{B^I})$ for images, where unlike the cross-modal experiment, both target concepts as well as attributes have the same modality. These experiments can provide insights as to which module is more heavily responsible for the observed bias. Columns 2 and 3 in Table 2 present the findings. For the image modality, we have images with positive and negative valence, analogous to positive and negative-sentiment words. As we can see, the model demonstrates similar bias to the cross-modal case. Similarly, we notice that “Islam” and “Judaism” attract more negative bias in comparison with “Christianity” and “Buddhism”. In some cases the effect size is slightly different, which can be explained by the fact that the number of samples in the case of textual data is smaller, entailing a greater standard deviation, which in turn alters the effect size.

5.2 Fusion-based Models

We next evaluate two fusion-based models. Although these models typically have independent textual and visual encoding modules in their lower layers, their architecture is complemented by a fusion module in higher levels to combine the information in different modalities, enabling them to learn joint embeddings of the visual and textual domains. This has been shown to be essential for more complex tasks such as VQA and NLVR that require more complex reasoning. The first such model we consider is ALBEF. Similar to CLIP, ALBEF (Li et al., 2021) first learns separate visual and textual embeddings using Transformer-based image and text encoders coupled with contrastive loss. However, unlike CLIP, ALBEF further combines these embeddings by adopting an attention-based fusion architecture to model more complex interactions between these modalities, and directly aims to address several vision–language objectives, including image–text matching and masked language modeling. This model is pretrained on conceptual captions and SBU captions (Sharma et al., 2018; Ordonez et al., 2011). Furthermore, the model is trained using momentum distillation to facilitate learning by adding an auxiliary learning network to stabilize the leaning process.

ViLT (Kim et al., 2021) is another recent VLP model that is devised as a more computationally efficient alternative to CLIP and ALBEF. Unlike large and computationally-heavy image and text encoders in CLIP and ALBEF, ViLT utilizes only shallow linear layers to process the sequence of word embeddings and image patches of the text–image input pair. Furthermore, in order to enable the model to solve complex vision–language tasks such as VQA, NLVR, and ITM, a Transformer-based architecture is employed on top to capture the complex dynamics between the modalities. This model is trained using a combination of image–text matching, word patch alignment, and masked language modelling objectives.

With regard to bias assessment, unlike CLIP, fusion-based models do not provide explicit access to separate visual and textual embeddings but rather provide a combined embedding of the pair. As a result, computing the Caliskan distance in Eq. 1 is not possible. However, interestingly one of the objectives these models optimize for is the image–text matching (ITM) objective. ITM is the problem of estimating the probability that a given image–text pair is a match. This task is directly related to our bias evaluation problem. We can argue that a model is fair if the probability of assigning pleasantness scores is similar across different concepts. In other words, the following should hold for the ITM scores:

$$P_{\text{ITM}}(\boldsymbol{A}|\boldsymbol{X}) - P_{\text{ITM}}(\boldsymbol{B}|\boldsymbol{X}) \approx P_{\text{ITM}}(\boldsymbol{A}|\boldsymbol{Y}) - P_{\text{ITM}}(\boldsymbol{B}|\boldsymbol{Y})$$

Columns 4 and 5 in Table 2 include the results for ALBEF and ViLT. The numbers provided are probabilistic differences and are not comparable to the Caliskan scores provided for CLIP. In order to reduce irrelevant noise only the top 15 most significant matches are considered. Again, we see that these models exhibit strong biases favoring Christianity vs. Islam and Judaism, matching Christian images to positive words 45% more than Muslim and 51% more than Judaism. However, in case of nationality, these models show fewer signs of bias. Furthermore, ViLT and ALBEF show less bias towards physical disabilities compared to mental disability.

Algorithm 1 Bias Mitigation Algorithm

```
Require: Image Embedding $V^I$,
         Text Embedding $V^T$,
         Features to remove $N$,
         Classification Labels $L$

$X \leftarrow \emptyset$
$\Psi \leftarrow$ Compute_Bias($V^I, V^T$)
for $d \leftarrow 1$ to $\mathrm{len}(V^I)$ do
    $\overline{V^I} \leftarrow V^I \setminus v^I_d$
    $\overline{V^T} \leftarrow V^T \setminus v^T_d$
    if MI($v^I_d, L$) $< \Theta$ then
        $\psi_d \leftarrow$ Compute_Bias($\overline{V^I}, \overline{V^T}$)
        if $\psi_d < \Psi$ then
            $X \leftarrow X \cup \{(d, \psi_d)\}$
        end if
    end if
end for
$Z \leftarrow \mathrm{sort}_{\psi_d}(X)[0 : N]$ // Dimensions to remove
$X \leftarrow X \setminus Z$
return $X$
```

6 Bias Mitigation Algorithm

Bias mitigation methods typically fall into one of three categories: data augmentation (fair resampling), model adjustment, and embedding post processing algorithms. Each of these alternatives have their own benefits and drawbacks, but a major limitation of the first two is that they require retraining the models. This can be burdensome in many cases. In particular, we often lack access to the dataset, the model's training procedure, or in the case of large pretrained models, retraining may also be computationally infeasible on typical hardware and cost budgets. Post-processing methods, on the other hand, may be invoked as a fast and efficient plug-and-play method to modify learned embeddings without the need for retraining. Since vision–language tasks are complex, VLP models usually have large architectures to be able to capture all the complex dynamics. However, this can cause them to learn redundant or highly correlated features, since they are not optimally compressed. These features are not only computationally wasteful but can also amplify model bias. Due to the high correlation among some features, we can remove some without affecting performance, while simultaneously reducing bias. In order to identify those features, we directly optimize for the objective in Eq. 1 by removing features in a greedy manner, pruning N dimensions that cause the largest decrease in bias effect size. However, we only consider the features that exhibit a small mutual information with respect to classification labels. Θ can be set empirically and this ensures only redundant dimensions are removed. Algorithm 1 details the steps of our technique.

Bias	Before	After	Reduction
Muslim vs Christian	1.72	0.57	66%
Jewish vs Christian	1.69	0.75	55%
Muslim vs Buddhist	1.62	0.11	93%
Jewish vs Buddhist	1.61	0.30	82%
Muslim vs Hindu	1.65	0.71	57%
Arab vs American	1.28	0.33	74%
Mexican vs American	1.13	0.85	26%
Chinese vs American	0.89	0.56	38%
Mental Dis. vs No Dis.	1.48	0.49	66%
LGBTQ vs Heterosexual	1.67	0.92	45%

Table 3: Bias Mitigation Results. Our algorithm is able to significantly reduce bias without substantially affecting performance.

The results of this debiasing method are presented in Table 3. We removed 54 dimensions (10% of all dimensions), which leads to up to 93% bias reduction in some cases. This however only minimally affects the model's classification accuracy. We have tested the accuracy of the model on the MMbias dataset as well as the CIFAR-100 dataset. On MMbias, the accuracy dropped by only 1.1% , and on CIFAR100 by only 1.3% from 80.1% to 78.8%. Furthermore, Fig. 4b shows that even after removing the aforementioned dimensions, the embeddings still remain well-separable, confirming the redundancy of some of the embedding features.

Regarding the choice of N (number of features removed) in the bias mitigation algorithm, a larger N will affect the performance of the model more negatively, as previously observed in other dimensionality reduction algorithm. In order to find a reasonable N we can plot the bias reduction as well as performance reduction as a function of N. Inspecting this graph allows us to consider the trade-off between greater bias removal and the loss of accuracy, allowing us to choose an N that decreases the bias in a meaningful way while not affecting performance significantly.

7 Conclusion

Most bias analysis studies focus on gender and racial biases, which is primarily due to a lack of suitable data to consider further important forms of bias. In this study, we have compiled a new multimodal bias assessment dataset called MMBias enabling the study of bias affecting population groups largely neglected in prior studies. Our dataset consists of around 3,500 images and hundreds of phrases covering over 14 minority subgroups. Furthermore, based on a formulation of the bias-fairness problem, we draw on this data to assess the level of bias in several prominent self-supervised multimodal models, including CLIP, ALBEF, and ViLT. Our results show that these models demonstrate meaningful bias towards certain groups. Finally, we introduce a novel bias mitigation technique designed specifically for large pretrained models that can be applied as a post-processing step to reduce bias, and show that it has negligible effects on classification performance as well as data separability. Our data and code is available at github.com/sepehrjng92/MMBias.

Limitations

This work seeks to make a contribution towards vision–language models that exhibit less biased behavior. To this end, we provide a large new dataset, new experimental results, and also investigate a bias mitigation method for pre-trained vision–language models. Yet, bias measurement data as well are prone to biases, most notably in the selection of classes and groups, but also with regard to the particular data instances. We envision that MMBias will grow to encompass further groups and additional data in the future, e.g., further ethnic minorities, sexual identities, and gender identities. We also hope that our dataset can serve as a starting point for research on additional natural languages.

Clearly, our bias mitigation algorithm can only mitigate certain fairly overt expressions of bias in vision–language models. Large pre-trained models have millions of parameters that affect the model behavior. As vision–language models necessarily need to rate ties between images and text, they will continue to prefer or disprefer certain associations, leading to remnant biases. Still, we hope that our work will enable the community to pay closer attention to these challenges and work towards models that behave in more equitable ways.

Ethics Statement

With our work, we wish to encourage further analysis of bias in machine learning models. To this end, we provide data that enables an assessment of a number of potential manifestations of bias. We acknowledge that the images harbor a multitude of different stereotypes that cannot be taken to be representative of the various groups. Moreover, we acknowledge that the pairings of classes of people adopted thus far in our work leaves out other groups of people, e.g., further forms of faith and belief, and also further pairings. We view our work as a step towards a more inclusive bias assessment resource that should keep growing in the future.

References

Abubakar Abid, Maheen Farooqi, and James Zou. 2021. Large language models associate muslims with violence. *Nature Machine Intelligence*, 3(6):461–463.

Sandhini Agarwal, Gretchen Krueger, Jack Clark, Alec Radford, Jong Wook Kim, and Miles Brundage. 2021. Evaluating clip: towards characterization of broader capabilities and downstream implications. *arXiv preprint arXiv:2108.02818*.

Francis S Bellezza, Anthony G Greenwald, and Mahzarin R Banaji. 1986. Words high and low in pleasantness as rated by male and female college students. *Behavior Research Methods, Instruments, & Computers*, 18:299–303.

Shruti Bhargava and David Forsyth. 2019. Exposing and correcting the gender bias in image captioning datasets and models. *arXiv preprint arXiv:1912.00578*.

Shikha Bordia and Samuel R Bowman. 2019. Identifying and reducing gender bias in word-level language models. *arXiv preprint arXiv:1904.03035*.

Raymond Buriel and Richard Vasquez. 1982. Stereotypes of mexican descent persons: Attitudes of three generations of mexican americans and anglo-american adolescents. *Journal of Cross-Cultural Psychology*, 13(1):59–70.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

John F Dovidio, Lisa Pagotto, and Michelle R Hebl. 2011. Implicit attitudes and discrimination against people with physical disabilities. *Disability and aging discrimination: Perspectives in law and psychology*, pages 157–183.

Sedigheh Eslami, Gerard de Melo, and Christoph Meinel. 2021. Does CLIP benefit visual question answering in the medical domain as much as it does in the general domain? *CoRR*, abs/2112.13906.

Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L Alfonso Ureña-López. 2021. A survey on bias in deep nlp. *Applied Sciences*, 11(7):3184.

Anthony G Greenwald and Mahzarin R Banaji. 1995. Implicit social cognition: attitudes, self-esteem, and stereotypes. *Psychological review*, 102(1):4.

Anthony G Greenwald, Debbie E McGhee, and Jordan LK Schwartz. 1998. Measuring individual differences in implicit cognition: the implicit association test. *Journal of personality and social psychology*, 74(6):1464.

Wei Guo and Aylin Caliskan. 2021. Detecting emergent intersectional biases: Contextualized word embeddings contain a distribution of human-like biases. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 122–133.

Michelle R Hebl, Jessica Bigazzi Foster, Laura M Mannix, and John F Dovidio. 2002. Formal and interpersonal discrimination: A field study of bias toward homosexual applicants. *Personality and social psychology bulletin*, 28(6):815–825.

Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. 2020. Social biases in nlp models as barriers for persons with disabilities. *arXiv preprint arXiv:2005.00813*.

Amy K Kiefer and Denise Sekaquaptewa. 2007. Implicit stereotypes and women's math performance: How implicit gender-math stereotypes influence women's susceptibility to stereotype threat. *Journal of experimental social psychology*, 43(5):825–832.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR.

Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Thomas Manzini, Yao Chong Lim, Yulia Tsvetkov, and Alan W Black. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. *arXiv preprint arXiv:1904.04047*.

Saif M Mohammad. 2016. Sentiment analysis: Detecting valence, emotions, and other affectual states from text. In *Emotion measurement*, pages 201–237. Elsevier.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456*.

Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. 2011. Im2text: Describing images using 1 million captioned photographs. In *Neural Information Processing Systems (NIPS)*.

Jaihyun Park, Karla Felix, and Grace Lee. 2007. Implicit attitudes toward arab-muslims and the moderating effects of social information. *Basic and Applied Social Psychology*, 29(1):35–45.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Candace Ross, Boris Katz, and Andrei Barbu. 2020. Measuring social biases in grounded vision and language embeddings. *arXiv preprint arXiv:2002.08911*.

Wade C Rowatt, Lewis M Franklin, and Marla Cotton. 2005. Patterns and personality correlates of implicit and explicit attitudes toward christians and muslims. *Journal for the scientific study of religion*, 44(1):29–43.

Laurie A Rudman and Richard D Ashmore. 2007. Discrimination and the implicit association test. *Group Processes & Intergroup Relations*, 10(3):359–372.

James A Russell. 2003. Core affect and the psychological construction of emotion. *Psychological review*, 110(1):145.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*.

Kirill Sirotkin, Pablo Carballeira, and Marcos Escudero-Viñolo. 2022. A study on the distribution of social biases in self-supervised learning visual models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10442–10451.

Tejas Srinivasan and Yonatan Bisk. 2021. Worst of both worlds: Biases compound in pre-trained vision-and-language models. *arXiv preprint arXiv:2104.08666*.

Ryan Steed and Aylin Caliskan. 2021. Image representations learned with unsupervised pre-training contain human-like biases. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 701–713.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Jialu Wang, Yang Liu, and Xin Eric Wang. 2021a. Are gender-neutral queries really gender-neutral? mitigating gender bias in image search. *arXiv preprint arXiv:2109.05433*.

Jialu Wang, Yang Liu, and Xin Eric Wang. 2021b. Assessing multilingual fairness in pre-trained multimodal representations. *arXiv preprint arXiv:2106.06683*.

Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5310–5319.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Kankan Zhou, Yibin LAI, and Jing Jiang. 2022. Vl-stereoset: A study of stereotypical bias in pre-trained vision-language models. Association for Computational Linguistics.

CylE: Cylinder Embeddings for Multi-hop Reasoning over Knowledge Graphs

Chau Duc Minh Nguyen[*] and **Tim French** and **Wei Liu** and **Michael Stewart**
ARC Centre for Transforming Maintenance through Data Science,
Department of Computer Science and Software Engineering
The University of Western Australia
[*]`chau.nguyenducminh@research.uwa.edu.au`
`{tim.french, wei.liu, michael.stewart}@uwa.edu.au`

Abstract

Recent geometric-based approaches have been shown to efficiently model complex logical queries (including the intersection operation) over Knowledge Graphs based on the natural representation of Venn diagram. Existing geometric-based models (using points, boxes embeddings), however, cannot handle the logical negation operation. Further, those using cones embeddings are limited to representing queries by two-dimensional shapes, which reduced their effectiveness in capturing entities query relations for correct answers. To overcome this challenge, we propose unbounded cylinder embeddings (namely CylE), which is a novel geometric-based model based on three-dimensional shapes. Our approach can handle a complete set of basic first-order logic operations (conjunctions, disjunctions and negations). CylE considers queries as Cartesian products of unbounded sector-cylinders and consider a set of nearest boxes corresponds to the set of answer entities. Precisely, the conjunctions can be represented via the intersections of unbounded sector-cylinders. Transforming queries to Disjunctive Normal Form can handle queries with disjunctions. The negations can be represented by considering the closure of complement for an arbitrary unbounded sector-cylinder. Empirical results show that the performance of multi-hop reasoning task using CylE significantly increases over state-of-the-art geometric-based query embedding models for queries without negation. For queries with negation operations, though the performance is on a par with the best performing geometric-based model, CylE significantly outperforms a recent distribution-based model.

1 Introduction

Multi-hop Reasoning (MHR) on Knowledge Graphs (KGs) is a primary task in answering queries over large-scale knowledge graphs. Queries can be represented using First-Order-Logic (FOL) connectives (Brachman and Levesque, 2004), involving these operations: existential quantification ($\exists$), conjunction ($\wedge$), disjunction ($\vee$) and negation ($\neg$). MHR involves learning to answer these FOL queries, which has recently received attention from several studies (Hamilton et al., 2018; Ren et al., 2020; Ren and Leskovec, 2020). A common approach is to first transform the FOL query into a computation graph (see Figure 1), where nodes represent entity constants or variables and edges map to predicates and logical operations. Representing queries in this way enables the learning process to traverse paths of KGs via the computation graph, so as to find a set of answers for a given query. However, large-scale KGs (Bollacker et al., 2008; Vrandečić and Krötzsch, 2014; Lehmann et al., 2015; Speer et al., 2017; Fellbaum, 2010; Mitchell et al., 2018) are often incomplete and noisy, which makes explicit query mechanism, such as graph traversal incapable of returning correct answers to a query.

Motivated by the challenge above, we aim to reason about incomplete KGs using MHR (Lin et al., 2018; Zhang et al., 2021a). To achieve MHR, recent studies have proposed several query embedding (QE) methods based on geometry (Hamilton et al., 2018; Ren et al., 2020; Zhang et al., 2021b) and probability distribution representations (Ren and Leskovec, 2020; Choudhary et al., 2021a). A common approach of QE in the literature is to project the FOL queries into an embedding space, allowing a model to learn the embeddings of queries and entities. Answering these queries is equivalent to finding the similarity between the embedded queries and the embedded entities. Geometric-based models using cone embeddings ConE (Zhang et al., 2021b) are shown to be superior over the others, especially the ability of handling negations. ConE represents queries as Cartesian product of sector-cones in an embedding space. ConE projects target entities as lines and queries as areas of sector-cones. Intuitively,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1736–1751
May 2-6, 2023 ©2023 Association for Computational Linguistics

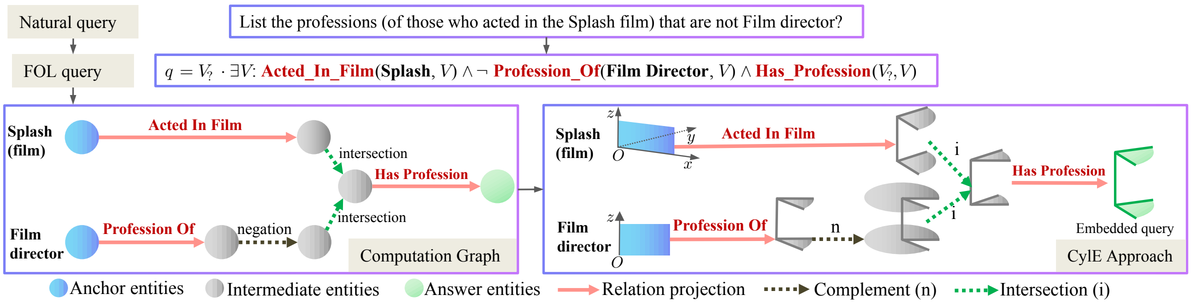


Figure 1: **(Left):** An example of a computation graph for a FOL query (rewritten from a natural query). **(Right):** The intuition of CylE is to model anchor and target entities as perpendicular boxes regarding the base of sector-cylinders. Each edge of the computation graph transforms the entity with its relation into the sector-cylinder embedding via projection, intersection or complement operations; then outputting an embedded query.

if the line is inside the area, the corresponding entity is considered as a match to the query. Nevertheless, cone embeddings are limited to the 2D space in a flat plane, to represent the query and entity embeddings. For example, one entity in a cluster of the sub-topic */movie/action* and one in another */movie/documentary* can be in the sector-cone region for those labels in the topic */movie* (see Figure 2(a)). Answering a query regarding the */movie/action* can return irrelevant entities from the */movie/documentary*. In multi-dimensional space, entities assigned to multiple labels can be achieved using multi-dimensional classification (Read et al., 2013). In the 3D space, for instance, apart from the xy-plane, semantics of entities can be additionally classified by different levels of degree in the height segment (along the z-axis).

In this paper, we propose to expand the two dimension sector-cone embeddings into the three dimension coordinate system. We represent queries by augmenting the shape of the sector-cones to be similar as unbounded sector-cylinders (shortly called *sector-cylinders*) in an embedding space, compared to closed sector-cylinders in a normal situation (see examples of unbounded sector-cylinders, their intersection or union in Figure 2 and further definitions in Section 4.1). In short, we name this approach **Cyl**inder **E**mbeddings (CylE). Answering a query is similar to finding entities that are subsets of the sector-cylinders representing queries. We investigate whether there is any improvement in answering correctly any query structures, using CylE over other approaches. Our approach can handle the conjunction as we notice that the intersection of sector-cylinders (along the z-axis) can be a sector-cylinder. Since the union of sector-cylinders is no longer a sector-cylinder, we first transform queries to Disjunctive Normal Form (DNF), which enables CylE to handle disjunction. With regard to the negation operation, we consider the closure-complement of sector-cylinders to model this operation. Our contributions are: (1) introducing the first 3D geometric-based approach to model the QE for MHR to the best of our knowledge, (2) enabling the model to handle a complete set of the basic FOL queries (existential quantification, conjunction, disjunction and negation) and (3) demonstrating that CylE significantly outperforms state-of-the-art (SOTA) geometric-based models for non-negation queries and is on par for queries with negations by empirical results.

2 Related works

Multi-hop Reasoning for logical query Studies in MHR employed approaches such as (distributions (Ren and Leskovec, 2020; Choudhary et al., 2021a; Huang et al., 2022), geometric shapes (Hamilton et al., 2018; Ren et al., 2020; Zhang et al., 2021b), fuzzy logic (Chen et al., 2022; Arakelyan et al., 2021)), others using count-min sketch (Sun et al., 2020) and neural-symbolic approach (Zhu et al., 2022), to achieve the common goal of learning representation of queries, *i.e.* query embeddings. The primary difference in these approaches is based on how queries are represented. For example, distribution-based models use Beta distributions (Ren and Leskovec, 2020) or Multivariate Gaussian distributions (Choudhary et al., 2021a). In geometric shapes, Hamilton et al. (2018) represented queries as point embeddings, Ren et al. (2020) then furthered this using box embeddings, and Bai et al. (2022) made an improvement by introducing 'particle' embeddings (a set of points using multiple vectors). Zhang et al. (2021b) have made the geometric approach more expressive using cone embeddings. Another difference is the ability to model a complete set of logical operations (Brachman and Levesque, 2004) (conjunction, disjunction and negation). Several methods (Ren and Leskovec, 2020; Zhang et al., 2021b; Bai et al., 2022; Chen et al., 2022) have achieved a coverage of all operations including the negation, compared to others (Hamilton et al., 2018; Ren et al., 2020; Choudhary et al., 2021a) without negation.

Reasoning about KGs using geometric shapes Historically, these approaches received attention since the introduction of translation-based methods (Bordes et al., 2013), rotation (Sun et al., 2019; Zhang et al., 2020) and 3D-rotation (Gao et al., 2020) for learning knowledge graph embeddings. Inspired by a Poincaré ball, other studies (Nickel and Kiela, 2017; Balažević et al., 2019) proposed hyperbolic space (non-Euclidean geometry). A common task of these studies is KGs completion. However, furthering it to MHR task poses a challenge because of the complex structures of queries (see Figure 3). Using geometric shapes for the MHR task (point Hamilton et al. (2018); Bai et al. (2022), box Ren et al. (2020), hyperbolic Choudhary et al. (2021b) and cone embeddings Zhang et al. (2021b)) have increasingly gained popularity. Cone embeddings were also mentioned in (Ganea et al., 2018), but not for the MHR task. Since existing geometric-based methods of cone embeddings rely on 2D shapes, we extend the representation learning for this geometric family to 3D shapes for the MHR task. Other studies (*e.g.* spherical text embeddings (Meng et al., 2019)) learned word embeddings for document clustering and classification tasks, but still not for the MHR task.

3 Preliminaries

3.1 Knowledge Graphs

Given a set of *vertices* (*entities*) $\mathcal{V}$ and a set of *edges* (*relations* or *predicates*) $\mathcal{E}$, we define a knowledge graph ($\mathcal{G}$) as a set of *triples*. Each triple is (v_s, e, v_o), where $(v_s, v_o \in \mathcal{V})$ and $(e \in \mathcal{E})$ is a *vertex subject*, a *vertex object* and an *edge* respectively. Assuming $(r \in \mathcal{R})$ denotes each element in a set of relation functions $(\mathcal{R})$, where (r) – associated with (e) – is a binary function $r : \mathcal{V} \times \mathcal{V} \to \{\text{True}, \text{False}\}$ that denotes an asymmetric direction of relation from (v_s) to (v_o), and vice versa. A symmetric direction of relation (non-directional relation) is $r : \mathcal{V} \times \mathcal{V} \to \{\text{True}, \text{True}\}$. Notice that there are two sets involving in edges/relations: $(\mathcal{E})$ for edge instances and $(\mathcal{R})$ for relation functions.

3.2 First-Order Logic queries

There are four basic logical operations involving in the interpretation of FOL queries[1]: conjunction ($\wedge$), disjunction ($\vee$), negation ($\neg$) and existential quantification ($\exists$). We adopt definitions and notations of BetaE (Ren and Leskovec, 2020) to assume that a FOL query consists of three folds: (1) a constant anchor entity set $(\mathcal{V}_a \subseteq \mathcal{V})$, (2) existentially quantified bound variables $(V_1, \dots, V_k)$ and (3) a target entity variable $(V_?)$ to respond a certain query. A FOL query can be written in Disjunctive Normal Form (DNF) as a combination of disjunctions of conjunctive queries (c_i) in the following:

$$q[V_?] = V_? \cdot \exists V_1, \dots, V_k : c_1 \vee c_2 \vee \dots c_n,$$

where $c_i = e_{i1} \wedge e_{i2} \cdots \wedge e_{im}$, including at least one literal $e_{ij} = r(v_a, V)$ or $\neg r(v_a, V)$ or $r(V', V)$ or $\neg r(V', V)$, and $(v_a \in \mathcal{V}_a)$ while $V \in \{V_?, V_1, \dots, V_k\}$, $V' \in \{V_1, \dots, V_k\}$ and $V' \neq V$. Finding the answer entities of a query (q) is similar to searching for an answer set $[\![q]\!] \subseteq \mathcal{V}$, where $v \in [\![q]\!]$ if and only if $q[v]$ is True.

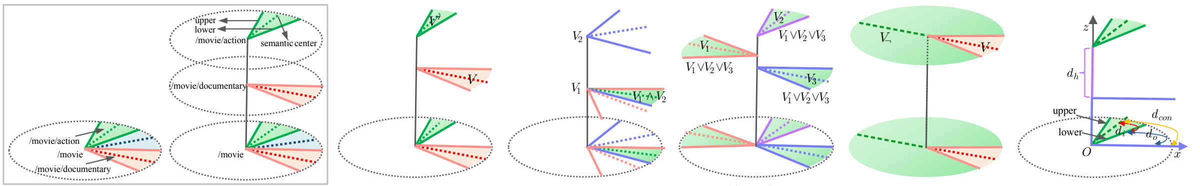
3.3 Query Decomposition

We adopt the definitions of FOL query decomposition in Zhang et al. (2021b) using a computation graph, including *vertices* and *edges* (see an example in Figure 1). Each intermediate vertex is a set of entities and each edge demonstrates relational projection or logical operations over entity sets:

- **Relation Traversal** → **Projection**: Given a set of entities $\mathcal{S} \subset \mathcal{V}$ and a relation function $r \in \mathcal{R}$, estimate adjacent entities $\cup_{v \in \mathcal{S}} \mathcal{A}(v, r)$, where $\mathcal{A}(v, r) \equiv \{v' \in \mathcal{V} : r(v, v') = \text{True}\}$.
- **Negation** → **Complement**: Given a set of entities $\mathcal{S} \subset \mathcal{V}$, estimate $\overline{\mathcal{S}}$ where $\overline{\mathcal{S}} \equiv \mathcal{V} \backslash \mathcal{S}$.
- **Conjunction** → **Intersection**: Given a number of sets of entities $\{\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_n\}$, compute the intersection $\cap_{i=1}^{n} \mathcal{S}_i$ of these sets.
- **Disjunction** → **Union**: Given a number of entity sets $\{\mathcal{S}_1, \dots, \mathcal{S}_n\}$, find the union $\cup_{i=1}^{n} \mathcal{S}_i$.

---

[1] Universal quantification ($\forall$) rarely appears in the real world (Ren and Leskovec, 2020), this operation is therefore not considered.



(a) **(Left)**: ConE and **(Right)**: CylE (b) Projection (c) Intersection (d) Union (e) Complement (f) Distance

Figure 2: An overview of cone and cylinder embeddings with projection, logical operations (intersection, union, complement) and distance. The color dot line represents the semantic center axis. **(c)** Intersection between a sector-cylinder (red semantic center) and another higher sector-sector (royal blue semantic center) is the sector-cylinder having green semantic center. **(d)** Union of three sector-cylinders (having red, purple and royal blue semantic center axis) are in green regions. **(f)** Distance is from the green sector-cylinder (of an embedded query) to a purple target box (of an embedded target entity); d_i, d_o, d_h denotes the inside, outside and height distance.

4 Cylinder embeddings

We first provide a background of parameterizing cylinder (Section 4.1) and modeling cylinder embeddings for conjunctive queries (Section 4.2). Next, we show the cylinder embeddings process with logical operators (Section 4.3), then provide an optimization method (Section 4.4).

4.1 Cylinder definitions and parameterization

We define an *unbounded sector-cylinder* (without parameterizing radius) as having only two bounds for its body: the upper bound and the lower bound with an intersection as a height (see Figure 2(a)). We call the angle between the two bounds aperture, which has a range in $[0, 2\pi]$. In short, we call an unbounded sector-cylinder a *sector-cylinder* or a *cylinder* based on their apertures. Notice that a sector-cylinder becomes a cylinder when the aperture is zero. We use three variables for parameterization (the first two for the sector-cylinder's base adapt from ConE): (1) the *semantic center axis* $\theta_{ax} \in [-\pi, \pi)$ is the angle between the positive x-axis and the symmetric axis, (2) $\theta_{ap} \in [0, 2\pi]$ is the *aperture* and (3) $\theta_{he} \in (-\pi, \pi)$ is the *height*.

Note that the base of cylinders and cones share similarity in properties (semantic axis and aperture), which can be illustrated in the same plane, such as xy-plane. Thus, the base of cylinders in our study inherit some definitions and propositions from cones. These are a *cone*, a *convex cone* (Boyd et al., 2004), a *closure-complement* of a cone and a *sector-cone*. Each of these, which is defined by ConE (Zhang et al., 2021b), is a set in 2D space. Further, sector-cylinder's base has the same proposition as the sector-cone: *"always axially symmetric"* which has been proven in ConE.

Precisely, we define $(\mathbb{K})$ as a space consisting of all $(\theta_{ax}, \theta_{ap}, \theta_{he})$. An arbitrary sector-cylinder S_0 is as: $S_0 = (\theta_{ax}, \theta_{ap}, \theta_{he}) \in \mathbb{K}$. Then, a d-ary Cartesian product of sector-cylinders, called S, is a product of each sector-cylinder $S_{i=1\to d}$ (or each element of S is a d-dimensional vector in $\mathbb{K}^d$): $S = S_1 \times S_2 \cdots \times S_d$ or be rewritten as follows:

$$\begin{aligned} S := & \Big((\theta^1_{ax}, \theta^1_{ap}, \theta^1_{he}), \ldots, (\theta^d_{ax}, \theta^d_{ap}, \theta^d_{he})\Big) \\ = & \, (\boldsymbol{\theta}_{ax}, \boldsymbol{\theta}_{ap}, \boldsymbol{\theta}_{he}) \subset \mathbb{K}^d. \end{aligned} \tag{4.1}$$

4.2 Cylinder embeddings for conjunctive queries and entities

In this section, we describe query embeddings for conjunctive queries. Note that disjunctive queries can be transformed to DNF form as a set of conjunctive queries (as mentioned in Section 3.2). We model the *embedding region* $(\mathbf{V}_q)$ for the answer set $[\![q]\!]$ of the query (q) (see Section 3.3) using a Cartesian product of sector-cylinders (as mentioned in Section 4.1) as: $\mathbf{V}_q = (\boldsymbol{\theta}_{ax}, \boldsymbol{\theta}_{ap}, \boldsymbol{\theta}_{he})$, where embedding of semantic center axis is $\boldsymbol{\theta}_{ax} \in [-\pi, \pi)^d$, embedding of aperture is $\boldsymbol{\theta}_{ap} \in [0, 2\pi]^d$ and embedding of height is $\boldsymbol{\theta}_{he} \in (-\pi, \pi)^d$; and (d) denotes the dimension of the embedding space (see Eq. (4.1)).

Next, an *arbitrary entity* $(v \in \mathcal{V})$ is represented by a Cartesian product of cylinders having zero apertures. The corresponding embedding $(\mathbf{v})$ is as: $\mathbf{v} = (\boldsymbol{\theta}_{ax}, \mathbf{0}, \boldsymbol{\theta}_{he})$. The intuition is to embed the anchor or target entity into one similar as a *perpendicular box* with regard to a base of the cylinder, Precisely, all elements of the d-dimensional vector $(\boldsymbol{\theta}_{ap})$ is equal to zero.

4.3 Cylinder embeddings with logical operators

We describe the process of modeling relational projection (*projection* module) and modeling logical operators (*intersection*, *complement*, *union* module), to embed the FOL queries in the following:

Projection module: This module aims to learn the projection operation (see Section 3.3), which outputs the adjacent entities (of an anchor entity) linking to a given relation. To this end, we maps the embedding region of an entity set $(\mathbf{V}_q)$ to another $(\mathbf{V}'_q)$ (see Figure 2(b)) via a mapping function (f):

$$f : \mathbb{K}^d \rightarrow \mathbb{K}^d, \mathbf{V}_q \rightarrow \mathbf{V}'_q.$$

We use a *shallow neural network* to approximately represent the mapping function $f(x)$. Overall, $(\mathbf{V}_q)$ is a summation of the embeddings of entity set $(\mathbf{V})$ and a relation $(\mathbf{r})$ as: $\mathbf{V}_q = \mathbf{V} + \mathbf{r}$; where the representation of each relation $(\mathbf{r})$ is sector-cylinder embeddings. Then, a composition function $f(\mathbf{V}_q)$ has a scaling of chunking function $g(x)$ and a multilayer perceptron (**MLP**) network:

$$f(\mathbf{V}_q) = g(\mathbf{MLP}([\mathbf{V}_q])),$$

where $\mathbf{MLP} : \mathbb{R}^{3d} \rightarrow \mathbb{R}^{3d}$, and the function $g(x)$ is to split a three-dimensional vector into three d-vectors for semantic center axis, aperture and height embeddings. In addition, there is a scaling operation, adapt from Zhang et al. (2021b), involving in the function $g(x)$ to scale the semantic center axis, the aperture and the height embeddings into their normal ranges (as mentioned in Section 4.2):

$$[f(\boldsymbol{x})]_i = \begin{cases} \theta'^{i}_{ax} = \pi \tanh(\lambda_1 x_i), & \text{if } i \leq d, \\ \theta'^{i-d}_{ap} = \pi \tanh(\lambda_2 x_i) + \pi, & \text{if } i > d, \\ \theta'^{i}_{he} = 2\pi(sigmoid(\lambda_3 x_i) - 0.5), & \end{cases}$$

where $[f(\boldsymbol{x})]_i$ is the i-th element of $f(\boldsymbol{x})$; $(\lambda_1, \lambda_2, \lambda_3)$ are the scaling hyper-parameters.

Intersection module: This module aims to learn the conjunction operation (see Section 3.3). Assuming a conjunction of conjunctive queries (q_i) associates with a query (q), its answer is $[\![q]\!] = \cap_{i=1}^{k} [\![q_i]\!]$. Notice that entities in the set $[\![q]\!]$ share semantic similarity with one another, as the conjunction of conjunctive queries based on sector-cylinder embeddings are conjunctive queries (see Figure 2 (c)). Supposing $\mathbf{V}_{\cap q} = (\boldsymbol{\theta}_{ax}, \boldsymbol{\theta}_{ap}, \boldsymbol{\theta}_{he})$ and $\mathbf{V}_{i,q} = (\boldsymbol{\theta}_{i,ax}, \boldsymbol{\theta}_{i,ap}, \boldsymbol{\theta}_{i,he})$ are embedding region of $[\![q]\!]$ and $[\![q_i]\!]$ respectively. To obtain the $\mathbf{V}_{\cap q}$, we then take the summation w.r.t. the number of conjunctive queries (of the Hadamard product $\odot$ between $\boldsymbol{A}_i$ and $\mathbf{V}_{i,q}$), which is shown below:

$$\mathbf{V}_{\cap q} = \textstyle\sum_i^k \boldsymbol{A}_i \odot \mathbf{V}_{i,q},$$

where $\boldsymbol{A} \in \mathbb{R}^{k \times d}$ is an attention matrix defined by:

$$\boldsymbol{A}_{i \times d} = \frac{\exp\Big(\mathbf{MLP}([\mathbf{V}_q])_i\Big)}{\sum_j^n \exp\Big(\mathbf{MLP}([\mathbf{V}_q])_j\Big)},$$

where (k) is the number of involving conjunctive queries and $[\mathbf{V}_q]_i \in \mathbb{R}^{3d}$ is a concatenation of $(\boldsymbol{\theta}_{i,ax}, \boldsymbol{\theta}_{i,ap}, \boldsymbol{\theta}_{i,he})$ for the i-th conjunctive query and $\mathbf{MLP} : \mathbb{R}^{3d} \rightarrow \mathbb{R}^d$. As mentioned in Ren et al. (2020), using attention mechanism is important in comparison to other approaches (*e.g.* averaging, deep sets Zaheer et al. (2017)). Note that our approach is to approximately model the conjunction operation. Further, this approach is different than that in cone embeddings which required an intermediate process (Zhang et al., 2021b): to convert the semantic center axis to points on the unit circle, then to map these points back to angles to recover the semantic center axis.

Complement module: This module aims to represent the negation operation (see Section 3.3), by finding the complementary set of $[\![q]\!]$ (or $\mathcal{V} \backslash [\![q]\!]$): $[\![\neg q]\!]$. Supposing $\mathbf{V}_q = (\boldsymbol{\theta}_{ax}, \boldsymbol{\theta}_{ap}, \boldsymbol{\theta}_{he})$ and $\mathbf{V}_{\neg q} = (\boldsymbol{\theta}'_{ax}, \boldsymbol{\theta}'_{ap}, \boldsymbol{\theta}'_{he})$ are sector-cylinder embeddings region of $[\![q]\!]$ and $[\![\neg q]\!]$ respectively. From a geometric aspect, the closure-complement is close to the set of sector-cylinders (see Figure 2(e)). Thus, the sum of apertures of $(\mathbf{V}_q)$ and $(\mathbf{V}_{\neg q})$ should be close to 2π. Assuming semantic center of $(\mathbf{V}_{\neg q})$ should be opposite to those in $(\mathbf{V}_q)$ while the height of $(\mathbf{V}_{\neg q})$ should be equivalent to those in $(\mathbf{V}_q)$ as follows:

$$[\boldsymbol{\theta}'_{ax}]_i = \begin{cases} [\boldsymbol{\theta}_{ax}]_i - \pi, & \text{if } [\boldsymbol{\theta}_{ax}]_i \geq 0, \\ [\boldsymbol{\theta}_{ax}]_i + \pi, & \text{if } [\boldsymbol{\theta}_{ax}]_i < 0, \end{cases}$$

$$[\boldsymbol{\theta}'_{ap}]_i = 2\pi - [\boldsymbol{\theta}_{ap}]_i,$$

$$[\boldsymbol{\theta}'_{he}]_i = [\boldsymbol{\theta}_{he}]_i.$$

Note that the height variable cannot be involved in this module, as the negation should be closed without this variable. Since the negation is not closed w.r.t. entities as long as keeping the same height for $(\mathbf{V}_{\neg q})$ and $(\mathbf{V}_q)$, this can be a bottleneck of negation queries under the three dimension space. This can be addressed by designing closed negation for both queries and entities; however, we leave this direction for future work.

Union module: This module aims to represent the disjunction operation (see Section 3.3). Assuming a disjunction of conjunctive queries (q_i) associates with a query (q), the aim of this operation is to represent its answer: $[\![q]\!] = \cup_{i=1}^{k}[\![q_i]\!]$. We face a challenge as the union of sector-cylinders having different height is no longer a sector-cylinder, or the union over sector-cylinders is not closed (see Figure 2(d)). As mentioned by Ren et al. (2020), there is an issue of non-scalability when modeling directly the disjunction. To address this issue, we adapt a technique of Ren et al. (2020) by transforming queries into a DNF (Davey and Priestley, 2002) (*e.g.* disjunction of conjunctive queries). The union operation in DNF is moved to the last step of converted computation graphs, which contain conjunctive queries (see Section 3.2). Thus, we can apply the other logical operations (as mentioned in above modules) to have a set of embeddings of these conjunctive queries. The answer entities are those nearest to any embeddings of these conjunctive queries (see further details of estimating the aggregated distance score in Eq. (4.2)).

4.4 Optimization method

Distance score: We define a distance score $d(\mathbf{v};\mathbf{V})$ between the embedded region entity $\mathbf{v} = (\boldsymbol{\theta}'_{ax}, \mathbf{0}, \boldsymbol{\theta}'_{he})$ and the embedded region query $\mathbf{V} = (\boldsymbol{\theta}_{ax}, \boldsymbol{\theta}_{ap}, \boldsymbol{\theta}_{he})$. Inspired by Ren et al. (2020) and Zhang et al. (2021b), we adapt two types of this distance: (1) d_{con} (for those conjunctive queries) and (2) d_{dis} (for those disjunctive queries). In terms of estimating (d_{con}), there are three terms: an outside distance (d_o), an inside distance (d_i) and a height distance (d_h) as follows:

$$d_{con}(\mathbf{v};\mathbf{V}) = d_o(\mathbf{v};\mathbf{V}) + \lambda d_i(\mathbf{v};\mathbf{V}) + d_h(\mathbf{v};\mathbf{V}),$$

where the hyper-parameter $\lambda \in (0, 1)$ is to encourage the expected entity $(\mathbf{v})$ to be inside the embedded region $(\mathbf{V})$. Intuitively, $(\mathbf{v})$ is close to the upper or lower bound of $(\mathbf{V})$ when (λ) is close to zero or one. The three distances contributing to the (d_{con}) for conjunctive queries are defined by:

$$d_o = \left|\left|\min\{d_l, d_u\}\right|\right|_1, \; d_i = \left|\left|\min\{d_{ax}, d_{ap}\}\right|\right|_1,$$
$$d_h = \left|\left|\boldsymbol{\theta}'_{he} - \boldsymbol{\theta}_{he}\right|\right|_1,$$

where $(d_l = |1 - \cos(\boldsymbol{\theta}'_{ax} - \boldsymbol{\theta}_l)|)$ denotes the outside distance between the semantic center axis of the entity and the *lower bound* of the query, $(d_u = |1 - \cos(\boldsymbol{\theta}'_{ax} - \boldsymbol{\theta}_u)|)$ denotes the outside distance between the semantic center axis of the entity and the *upper bound* of the query; $(d_{ax} = |1 - \cos(\boldsymbol{\theta}'_{ax} - \boldsymbol{\theta}_{ax})|)$ denotes the inside distance between the semantic center axis of the entity and that of the query, $(d_{ap} = |1 - \cos(\boldsymbol{\theta}_{ap}/2)|)$ denotes the inside distance between the semantic center axis and either of the two bounds of the query (see Figure 2(f) for an example to estimate these distances). Further, $(\boldsymbol{\theta}_l = \boldsymbol{\theta}_{ax} - \frac{\boldsymbol{\theta}_{ap}}{2})$ is the *lower bound* and $(\boldsymbol{\theta}_u = \boldsymbol{\theta}_{ax} + \frac{\boldsymbol{\theta}_{ap}}{2})$ is the *upper bound* of the embedded query, the notation $||\cdot||_1$ is the L_1 norm. The higher value of the cosine function is, the less distance is. Notice that the maximum of this function is equivalent to one, we therefore subtract one from this, to ensure the minimum distance to be close to zero. Next, we adapt the DNF technique of Ren et al. (2020) to estimate (d_{dis}) by obtaining the minimum distance between a target entity and each conjunctive query:

$$d_{dis}(\mathbf{v};\mathbf{V}) = \min\{d_{con}(\mathbf{v};\mathbf{V}_i)\}_{i:1\to n}. \quad (4.2)$$

Training objective function: To optimize the training loss, we follow the objective function $(\mathcal{L})$ from Ren and Leskovec (2020), $\mathcal{L} = -\log\sigma(\gamma - d(\mathbf{v};\mathbf{V})) - \frac{1}{n}\sum_i^n \log\sigma(d(\mathbf{v}';\mathbf{V}) - \gamma)$ is a summation of two terms: (1) a positive loss is to minimize the distance $d(\mathbf{v};\mathbf{V})$ between a positive embedded entity $(v \in [\![q]\!])$ and an embedded query and (2) a negative sampling loss is to maximize the distance $d(\mathbf{v}';\mathbf{V})$ between negative embedded entities $(v'_{i:1\to n} \notin [\![q]\!])$ and an embedded query; where (n) is the number of negative sampling entities, $\sigma(x)$ denotes the sigmoid activation function and the hyper-parameter (γ) is a pre-fixed positive margin.

5 Experiments

5.1 Experimental designs

For benchmarking, we follow experimental designs (datasets, query structures, training and evaluation protocol) of Multi-hop Reasoning (MHR) task from (Ren and Leskovec, 2020).

Datasets: We use benchmarking datasets for the MHR task: FB15k (Bollacker et al., 2008), FB15k-237 (Toutanova and Chen, 2015) and NELL995 (Xiong et al., 2017). Using the same pre-processing steps as BetaE (Ren and Leskovec, 2020), we split each dataset into the training, validation, and test set. The aim of MHR task is to obtain *non-trivial* answers, which *cannot be discovered* by directly *traversing* the *incomplete KGs*, for each arbitrary FOL query. Please see Appendix A.1 for further details of these datasets.

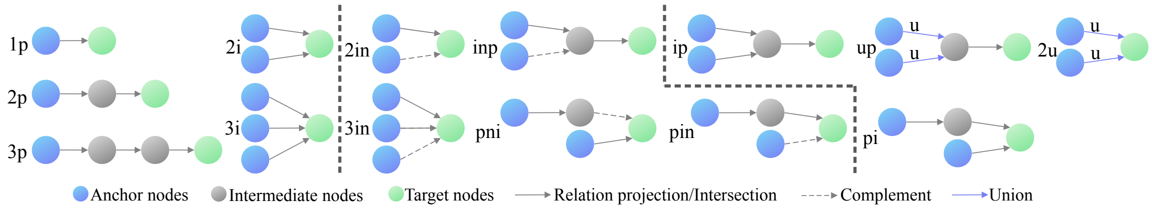


Figure 3: **(Left)** & **(Middle):** query structures are involved in the training process. **(Left)**, **(Middle)** & **(Right):** all queries are involved in the evaluation process; p is projection, i is intersection, n is negation and u is union.

Dataset	**Model**	**1p**	**2p**	**3p**	**2i**	**3i**	**ip**	**pi**	**2u**	**up**	**AVG**
FB15k	GQE	53.9	15.5	11.1	40.2	52.4	19.4	27.5	22.3	11.7	28.2
	Q2B	70.5	23.0	15.1	61.2	71.8	28.7	41.8	37.7	19.0	40.1
	BetaE	65.1	25.7	24.7	55.8	66.5	28.1	43.9	40.1	25.2	41.6
	ConE ($d = 800$)	73.3	33.8	29.2	64.4	73.7	35.7	50.9	55.7	31.4	49.8
	CylE ($d = 512$)	78.5	34.6	29.2	65.5	74.5	37.9	52.2	57.9	31.6	51.3
	CylE ($d = 800$)	**78.8**	**37.0**	**30.9**	**66.9**	**75.7**	**40.8**	**53.8**	**59.4**	**33.5**	**53.0**
FB15k-237	GQE	35.2	7.4	5.5	23.6	35.7	10.9	16.7	8.4	5.8	16.6
	Q2B	41.3	9.9	7.2	31.1	45.4	13.3	21.9	11.9	8.1	21.1
	BetaE	39.0	10.9	10.0	28.8	42.5	12.6	22.4	12.4	9.7	20.9
	ConE ($d = 800$)	41.8	12.8	11.0	32.6	47.3	14.0	25.5	14.5	10.8	23.4
	CylE ($d = 512$)	42.5	13.0	11.0	34.4	48.4	15.0	26.3	15.2	11.2	24.1
	CylE ($d = 800$)	**42.9**	**13.3**	**11.3**	**35.0**	**49.0**	**15.7**	**27.0**	**15.3**	**11.2**	**24.5**
NELL995	GQE	33.1	12.1	9.9	27.3	35.1	14.5	18.5	8.5	9.0	18.7
	Q2B	42.7	14.5	11.7	34.7	45.8	17.4	23.2	12.0	10.7	23.6
	BetaE	53.0	13.0	11.4	37.6	47.5	14.3	24.1	12.2	8.5	24.6
	ConE ($d = 800$)	53.1	16.1	13.9	40.0	50.8	17.5	26.3	15.3	11.3	27.2
	CylE ($d = 512$)	**56.5**	**17.5**	**15.6**	**41.4**	**51.2**	**19.6**	**27.2**	**15.7**	**12.3**	**28.5**
	CylE ($d = 800$)	55.7	17.5	15.1	40.7	51.1	19.1	27.1	15.4	12.2	28.2

Table 1: The average **MRR** (%) results in different query structures without negation ($\exists$, $\wedge$, $\vee$) using these datasets: FB15k, FB15k-237 and NELL995. The results of baselines (GQE, Q2B, BetaE, ConE) are taken from (Zhang et al., 2021b). Query structures with union operations ($2u/up$) are in DNF forms.

Queries: We adopt FOL query structures of (Ren and Leskovec, 2020) for the training, validation and test process. In terms of the training, there are five structures without negation ($1p/2p/3p/2i/3i$) and five structures with negation ($2in/3in/inp/pni/pin$). With regard to the evaluation process, we not only use the same query structures as those in the training process, we also use unseen structures ($ip/pi/2u/up$), which have not been involved in the training process, to evaluate the ability of generalization for the model.

Training and evaluation protocol: In terms the training process, we use Adam optimizer (Kingma and Ba, 2015). We follow the similar hyper-parameter settings of (Zhang et al., 2021b) to initialize the model, but search for the most effective combination of these hyper-parameters in the situation of cylinder embeddings (see more details in Appendix A.2). With regard to the evaluation process, we adopt the evaluation protocol of (Ren and Leskovec, 2020). There are three involving KGs: the training KG ($\mathcal{G}_{train}$ for training edges), the validation KG ($\mathcal{G}_{valid}$ for training and validation edges) and the test KG ($\mathcal{G}_{test}$ for training, validation and test edges) (see Appendix A.1). Specifically, given a test query (q) of incomplete KGs, our aim is to find *non-trivial* answers $[\![q]\!]_{\text{test}} \backslash [\![q]\!]_{\text{valid}}$ ($[\![q]\!]_{\text{valid}} \backslash [\![q]\!]_{\text{train}}$). We use the same metric Mean Reciprocal Rank (MRR) as described in (Ren et al., 2020; Ren and Leskovec, 2020; Zhang et al., 2021b) to evaluate the performance of Multi-hop Reasoning. Suppose ($\mathcal{Q}$) is a set of $[\![q]\!]_{\text{test}} \backslash [\![q]\!]_{\text{valid}}$, for each *non-trivial* answer ($v \in \mathcal{Q}$), we rank (v) against non-answer entities $\mathcal{V} \backslash [\![q]\!]_{\text{test}}$ (where v is associated with the rank r). We estimate the MRR as follows: $\text{MRR} = \frac{1}{|\mathcal{Q}|} \sum_{v \in \mathcal{Q}}^{|\mathcal{Q}|} \frac{1}{r}$. The higher MRR is, the better performance of the model is.

Baselines: There are four baseline models: GQE (Hamilton et al., 2018), Query2Box/Q2B (Ren et al., 2020), BetaE (Ren and Leskovec, 2020) and ConE (Zhang et al., 2021b). We obtain the recent results of these models from Zhang et al. (2021b), which are slightly higher than those reported by Ren and Leskovec (2020). We not only use the same embedding dimension $d = 800$ as ConE for fair comparisons, but we also run experiments using $d = 512$ for the sensitivity analysis[2].

5.2 Results

We report our main results of Multi-hop Reasoning using CylE regarding FOL queries with and without negation. Specifically, we compare the performance of CylE with these baselines: GQE, Q2B, BetaE and ConE using the same benchmarking datasets as mentioned above. We obtain the average results of five experiments for each dataset when the embedding dimension $d = 800$ (see Appendix B.1 for error bars of these results). For the sensitivity analysis with $d = 512$, we obtain results of an experiment for each dataset.

Modeling queries without negation: Table 1 demonstrates the average performance of Multi-hop Reasoning using CylE regarding existential positive first-order (EPFO) queries (a subset of FOL queries without negation), compared to baselines. Overall, CylE significantly outperforms all baselines. In comparison with the SOTA model ConE ($d = 800$), the average performance for all query structures (**AVG**) of CylE gains around 6.4%, 4.7% and 3.7% using the dataset FB15k, FB15k-237 and NELL995 respectively. More specifically, CylE achieves the highest improvement regarding ip queries, by nearly 14.3%, 12.1% and 9.1% using these datasets. In comparison with the previous model BETA, the **AVG** of CylE ($d = 800$) is also considerably higher, by around 27.4%, 17.2% and 14.6%, observed in the three datasets.

In terms of using the DNF technique, notice that the performance of answering those queries with unions ($2u$) only is significantly lower than those of queries with intersections ($2i$), by a large margin. We consistently observed this point in the three datasets. This can be due the limitation of representing union queries using the DNF technique, where it is challenging to expect an answer entity, for example, to be nearest to all conjunctive queries in the DNF form (see Eq. (4.2)). We also report the results for query structures regarding the union operation using De Morgan's (DM) law. Since there might be a problem of inconsistency with the real set union (as discussed in (Zhang et al., 2021b)), the results for union operation using DNF ($2u/up$) are higher than those using DM law (see Section 5.3 for further details).

Modeling queries with negation: Table 2 shows the average performance of Multi-hop Reasoning using CylE regarding query with negation, compared to baselines. Although ConE achieves the highest performance in average using the three datasets, the **AVG** of CylE ($d = 800$) is close to those in ConE. In comparison with the previous model BetaE, the **AVG** of CylE outperforms significantly. This increasing trend is similar to that in the ConE. Note that handling queries with negation are still challenging in all models (BetaE, ConE and CylE) since the **AVG** are significantly lower than those in queries without negation operations. This challenge may be due to a high uncertainty in the large number of answers for negation queries.

Dataset	**Model**	**2in**	**3in**	**inp**	**pin**	**pni**	**AVG**
FB15k	BetaE	14.3	14.7	11.5	6.5	12.4	11.8
	ConE ($d = 800$)	**17.9**	**18.7**	12.5	**9.8**	**15.1**	**14.8**
	CylE ($d = 512$)	15.6	15.9	13.3	7.5	13.6	13.2
	CylE ($d = 800$)	15.7	16.3	**13.7**	7.8	13.9	13.5
FB15k-237	BetaE	5.1	7.9	7.4	3.6	3.4	5.4
	ConE ($d = 800$)	**5.4**	**8.6**	7.8	**4.0**	**3.6**	**5.9**
	CylE ($d = 512$)	4.8	8.3	8.1	3.6	3.4	5.7
	CylE ($d = 800$)	4.9	8.3	**8.2**	3.7	3.4	5.7
NELL995	BetaE	5.1	7.8	10.0	3.1	3.5	5.9
	ConE ($d = 800$)	**5.7**	**8.1**	10.8	**3.5**	**3.9**	**6.4**
	CylE ($d = 512$)	5.6	7.5	11.2	3.4	3.7	6.3
	CylE ($d = 800$)	5.4	7.6	**11.3**	3.4	3.7	6.3

Table 2: The average **MRR** (%) in different query structures with negation using these datasets: FB15k, FB15k-237 and NELL995. The baseline results (BetaE and ConE) are taken from (Zhang et al., 2021b).

Effects of the embedding dimension: In terms of queries without negation, there is a slight difference in MRR results of CylE between the embedding dimension $d = 800$ and $d = 512$ (see Table 1). MRR results in these query structures with $d = 800$ are higher than those with $d = 512$ using these datasets FB15k and FB15k-237. In the NELL995 dataset, however, MRR results with $d = 800$ are lower than those with $d = 512$. There is a different trend in these datasets as there may be

---

[2] Source code is available at https://github.com/nlp-tlp/cyle

an over-fitting problem when implementing CylE using the NELL995 dataset, but not for the FB15k and FB15k-237 dataset. Notice that large number of entities increase the complexity of the model using the NELL995 dataset. Precisely, the number of entities in this dataset (63,361) is considerably larger than those in the FB15k (14,951) and FB15k-237 dataset (14,505) (see Appendix A.1). Similarly, the number of relations also contribute to the complexity of the model. Although the number of relations in the NELL995 dataset (200) is less than those in the FB15k (1,345) and FB15k-237 dataset (237), this difference is slight, compared to that in the number of entities across the three datasets. With regard to negation queries, MRR results of CylE with $d = 512$ are close to those with $d = 800$ (see Table 2). Hence, there is little effect of the embedding dimension on the performance of CylE in these query structures.

5.3 Comparisons results of disjunctive queries using DNF and De Morgan's law

Table 3 shows the comparisons of MRR results (in percentage) of query structures regarding union operations using DNF and DM transformation. We compare results of CylE with those in ConE and BETA, since these models can handle the negation operations requiring for the transformation in DM queries. Overall, the MRR in DNF query structures of the approaches using ConE and CylE are higher than those in DM query structures. There is a similar trend in ConE and CylE, since these approaches share similarity in the aperture (the boundary of shapes in sector-cone and sector-cylinder respectively). Note that we use complement operations to transform union queries into DM forms, the queries in this situation are represented in geometry (sector-cone and sector-cylinder respectively). However, not all queries are represented well in geometry as discussed in (Zhang et al., 2021b). Further, we also observe a higher margin in MRR results of CylE than those in ConE, regarding most of DNF and DM query structures, using the three datasets (FB15k, FB15k-237 and NELL995).

Dataset	Model	2u-DNF	2u-DM	up-DNF	up-DM
FB15k	BetaE	40.1	25.0	25.2	25.4
	ConE	55.7	37.7	31.4	29.8
	CylE	**59.4**	**42.4**	**33.5**	**32.0**
FB15k-237	BetaE	12.4	11.1	9.7	9.9
	ConE	14.5	13.4	10.8	9.9
	CylE	**15.3**	**13.4**	**11.2**	**10.6**
NELL995	BetaE	12.2	11.0	8.5	8.6
	ConE	15.3	**14.8**	11.3	10.8
	CylE	**15.4**	13.3	**12.2**	**11.5**

Table 3: **MRR** (%) for answering FOL disjunctive query structures using DNF and DM on these datasets: FB15k, FB15k-237 and NELL995. The results of BetaE and ConE are taken from (Zhang et al., 2021b).

6 Conclusion

We have presented a novel query embeddings (QE) model using cylinder embeddings (CylE), which can handle a complete set of arbitrary FOL queries, to perform the Multi-hop Reasoning (MHR) task. To the best of our knowledge, CylE is the first 3D geometric-based QE model for MHR. Experiments show significant performance gain over previous approaches for non-negation queries. For queries with negation operations, we face a similar challenge to previous models *i.e.* low MHR performance. This is a future direction to improve CylE on these queries. This work paves the way for opening the geometric-based QE method using three dimensional shapes.

Limitations

Although CylE can learn to achieve the Multi-hop Reasoning task or answering complex queries, several limitations in this work are taken into account. First, the modeling process of logical operators (conjunction, disjunction and negation) using geometric-based perspective is an approximate method in a learning manner which may not satisfy some logic laws. This can be addressed by using fuzzy logic under fuzzy sets representation for these logical operators Chen et al. (2022). Fuzzy logic is a learning-free manner for logical operators in FOL queries. Combining this approach with the neural models to learn these operators can be a potential approach, but we leave this extension as a direction for future work.

Another limitation is that modeling union operators in EPFO queries using the DNF technique may not find all expected answer entities. Note that modeling this operator is similar to finding nearest entities to all conjunctive queries (in the DNF form), which may not an optimal solution when the geometric embeddings of these queries locate far from one to another, as mentioned in Section 5.2, queries structure $2u$ and $2i$ in particular.

Acknowledgements

This research is supported by the Australian Research Council through the Centre for Transforming Maintenance through Data Science (grant number IC180100030), funded by the Australian Government. Wei Liu acknowledges the support from ARC Discovery Projects DP150102405. Further, the authors would like to thank School of Physics, Maths and Computing – Department of Computer Science and Software Engineering, and Kaya team – IT High Performance Computing system at The University of Western Australia (UWA) for funding facilities and resources; all the anonymous reviewers for their discussions and insightful feedback.

References

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. 2021. Complex query answering with neural link predictors. In *International Conference on Learning Representations (ICLR)*.

Jiaxin Bai, Zihao Wang, Hongming Zhang, and Yangqiu Song. 2022. Query2Particles: Knowledge graph reasoning with particle embeddings. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2703–2714, Seattle, United States. Association for Computational Linguistics.

Ivana Balažević, Carl Allen, and Timothy Hospedales. 2019. *Multi-Relational Poincaré Graph Embeddings*. Curran Associates Inc., Red Hook, NY, USA.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 2787–2795, Red Hook, NY, USA. Curran Associates Inc.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

Ronald Brachman and Hector Levesque. 2004. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Xuelu Chen, Ziniu Hu, and Yizhou Sun. 2022. Fuzzy logic based logical query answering on knowledge graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):3939–3948.

Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan Reddy. 2021a. Probabilistic entity representation model for reasoning over knowledge graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 23440–23451. Curran Associates, Inc.

Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K Reddy. 2021b. Self-supervised hyperboloid representations from logical queries over knowledge graphs. In *Proceedings of the Web Conference*, WWW '21, page 1373–1384, New York, NY, USA. Association for Computing Machinery.

Brian A Davey and Hilary A Priestley. 2002. *Introduction to lattices and order*, 2 edition. Cambridge university press.

Christiane Fellbaum. 2010. *WordNet*, pages 231–243. Springer Netherlands, Dordrecht.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 5350–5360, Red Hook, NY, USA. Curran Associates Inc.

Chang Gao, Chengjie Sun, Lili Shan, Lei Lin, and Mingjiang Wang. 2020. Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 385–394, New York, NY, USA. Association for Computing Machinery.

Will Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. 2018. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Zijian Huang, Meng-Fen Chiang, and Wang-Chien Lee. 2022. Line: Logical query reasoning over hierarchical knowledge graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 615–625, New York, NY, USA. Association for Computing Machinery.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA. Association for Computing Machinery.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations,* *ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3243–3253, Brussels, Belgium. Association for Computational Linguistics.

Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance Kaplan, and Jiawei Han. 2019. Spherical text embedding. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6341–6350, Red Hook, NY, USA. Curran Associates Inc.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. 2017. Searching for activation functions. *arXiv preprint arXiv:1710.05941*.

Jesse Read, Concha Bielza, and Pedro Larrañaga. 2013. Multi-dimensional classification with super-classes. *IEEE Transactions on knowledge and data engineering*, 26(7):1720–1733.

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *International Conference on Learning Representations (ICLR)*.

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4444–4451. AAAI Press.

Haitian Sun, Andrew O. Arnold, Tania Bedrax-Weiss, Fernando Pereira, and William W. Cohen. 2020. Faithful embeddings for knowledge base queries. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573, Copenhagen, Denmark. Association for Computational Linguistics.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbhakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J Smola. 2017. Deep sets. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3394–3404, Red Hook, NY, USA. Curran Associates Inc.

Yao Zhang, Hongru Liang, Adam Jatowt, Wenqiang Lei, Xin Wei, Ning Jiang, and Zhenglu Yang. 2021a. GMH: A general multi-hop reasoning model for KG completion. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3437–3446, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhanqiu Zhang, Jianyu Cai, and Jie Wang. 2020. Duality-induced regularizer for tensor factorization based knowledge graph completion. *Advances in Neural Information Processing Systems*, 33:21604–21615.

Zhanqiu Zhang, Jie Wang, Jiajun Chen, Shuiwang Ji, and Feng Wu. 2021b. Cone: Cone embeddings for multi-hop reasoning over knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 34.

Zhaocheng Zhu, Mikhail Galkin, Zuobai Zhang, and Jian Tang. 2022. Neural-symbolic models for logical queries on knowledge graphs. In *International Conference on Machine Learning (ICML)*.

A Further experimental details

We provide additional details for Section 5.1 in this section, divided into parts: Datasets & query structures A.1 and training & evaluation protocol A.2.

A.1 Datasets and query structures

We train and evaluate models using the same datasets (FB15k (Bollacker et al., 2008), FB15k-237 (Toutanova and Chen, 2015) and NELL995 (Xiong et al., 2017)) as those in (Ren and Leskovec, 2020) for the task of Multi-hop Reasoning (see Table 4 for a number of entities, a number of relations, a number of edges for each dataset). These datasets has been pre-processed by (Ren and Leskovec, 2020) to generate query structures for the training/validation/test set (see Table 5 for a description of these query structures and Table 6 for a description of average number of answer entities for test queries). These datasets are available at this link [3].

Dataset	Entities	Relations	Edges			
			Training	Validation	Test	Total
FB15k	14,951	1,345	483,142	50,000	59,071	592,213
FB15k-237	14,505	237	272,115	17,526	20,438	310,079
NELL995	63,361	200	114,213	14,324	14,267	142,804

Table 4: A statistical description of number of entities, relations, training/validation/test edges, reported from (Ren and Leskovec, 2020), in three datasets: FB15k, FB15k-237 and NELL995.

A.2 Training and evaluation protocol

We compare our results with these baselines (GQE, Query2Box, BetaE and ConE), taken from (Zhang et al., 2021b). We conduct all experiments using the Pytorch framework. Our implementation is done based on the original work of BetaE (Ren and Leskovec, 2020). [4] We adopt hyper-parameters, found by (Zhang et al., 2021b): the dimension of embedding $d = 800$, $\lambda_1 = 1.0$, $\lambda_2 = 2.0$, $\lambda = 0.02$, the batch size $b = 512$ and the negative sampling size $n = 128$. We also search for these hyper-parameters for best performance in MRR: the γ in the loss function [20, 30], the learning rate $\{1e^{-4}, 5e^{-5}\}$ and the scaling weight for the height variable λ_3 $\{1.0, 2.0\}$. We use a three-layer **MLP** (for a projection module) while two-layer **MLP** (for an intersection module), using 1600 dimension for hidden layers and Swiss activation function (Ramachandran et al., 2017). We run each experiment on a single NVIDIA Tesla V100 GPU. More details of hyper-parameters are shown in Table 7. Note that we also search for hyper-parameters in terms of experiments using ReLU activation function (in MLP) for ablation study. In this situation, we follow the same found hyper-parameters in Table 7 as those in the situation using Swiss activation function (the dimension of embedding d, the batch size b, the number of negative sampling size n, the maximum number of training steps m, the scaling hyper-parameters $\lambda_{i=\{1,2,3\}}$, the controlling distance λ, the learning rate l and the γ in the loss function), except for the found $\gamma = 30$ in the loss function using the FB15k-237 dataset, during the training process.

B Additional experimental results

B.1 Error bars

Table 8 and Table 9 show the error bars of MRR for the task of Multi-hop Reasoning using our approach with CylE, for queries without and with negation respectively using the three dataset FB15k, FB15k-237 and NELL995. More specifically, we run five experiments using different seed values in $\{0, 10, 100, 1000, 10000\}$ during the initialization process (for each dataset). We estimate the average MRR for different query structures of five experiments and obtain the standard deviation (for each dataset).

In terms of queries without negation, the standard deviation of the average MRR is small, at around 0.101, 0.039 and 0.113 using the dataset FB15k, FB15k-237 and NELL995 respectively. A similar trend is observed in queries with negation operation, since the standard deviation of the average MRR is also small using the three datasets. These error bars show a level of degree in stability of MRR (for each dataset) using different values of random seed during the initialization process.

B.2 Modeling the cardinality of answer sets using correlation coefficients

It is argued that the aperture embeddings may have a correlation with the number of the answer set $[\![q]\!]$. This correlation though is not guaranteed under different circumstances (*e.g.* entities having identical relations to one another), the learnt embeddings can have a positive relationship with the number of elements (cardinality) of $[\![q]\!]$. We follow a technique

---

[3] https://github.com/snap-stanford/KGReasoning

[4] https://github.com/snap-stanford/KGReasoning, licensed under the MIT License.

Dataset	Training Queries		Validation Queries		Test Queries	
	1p/2p/3p/2i/3i	2in/3in/inp/pin/pni	1p	Each Other	1p	Each Other
FB15k	273,710	27,371	59,097	8,000	67,016	8,000
FB15k-237	149,689	14,968	20,101	5,000	22,812	5,000
NELL995	107,982	10,798	16,927	4,000	17,034	4,000

Table 5: A statistical description of number of training/validation/test queries in different structures, preprocessed by (Ren and Leskovec, 2020), in three datasets: FB15k, FB15k-237 and NELL995.

Dataset	1p	2p	3p	2i	3i	ip	pi	2u	up	2in	3in	inp	pin	pni
FB15k	1.7	19.6	24.4	8.0	5.2	18.3	12.5	18.9	23.8	15.9	14.6	19.8	21.6	16.9
FB15k-237	1.7	17.3	24.3	6.9	4.5	17.7	10.4	19.6	24.3	16.3	13.4	19.5	21.7	18.2
NELL995	1.6	14.9	17.5	5.7	6.0	17.4	11.9	14.9	19.0	12.9	11.1	12.9	16.0	13.0

Table 6: A statistical description of average number of answers for test queries, preprocessed by (Ren and Leskovec, 2020), in three datasets: FB15k, FB15k-237 and NELL995.

Dataset	d	b	n	m	γ	l	λ_1	λ_2	λ_3	λ
FB15k	800	512	128	450k	30	0.00005	1.0	2.0	2.0	0.02
FB15k-237	800	512	128	350k	20	0.00005	1.0	2.0	2.0	0.02
NELL995	800	512	128	350k	20	0.0001	1.0	2.0	2.0	0.02

Table 7: Found hyper-parameters for the main results in three different datasets: FB15k, FB15k-237 and NELL995. d denotes the embedding dimension, b denotes the batch size, n denotes the negative sampling size, γ denotes to control the loss function, m denotes the maximum training steps, l denotes the learning rate, $\lambda_1, \lambda_2, \lambda_3$ denote scaling hyper-parameters in the projection module (see scaling function $f(x)$ in Section 4.3) and λ is to control the distance d_{con} (see Section 4.4 in the main content).

Dataset	1p	2p	3p	2i	3i	ip	pi	2u	up	AVG
FB15k	78.8 ±0.044	37.0 ±0.156	30.9 ±0.231	66.9 ±0.093	75.7 ±0.132	40.8 ±0.257	53.8 ±0.263	59.4 ±0.210	33.5 ±0.226	53.0 ±0.101
FB15k-237	42.9 ±0.105	13.3 ±0.113	11.3 ±0.070	35.0 ±0.189	49.0 ±0.168	15.7 ±0.143	27.0 ±0.108	15.3 ±0.136	11.2 ±0.075	24.5 ±0.039
NELL995	55.7 ±0.290	17.5 ±0.112	15.1 ±0.145	40.7 ±0.223	51.1 ±0.291	19.1 ±0.147	27.1 ±0.099	15.4 ±0.154	12.2 ±0.037	28.2 ±0.113

Table 8: **MRR** (%) results of CylE with error bars for answering different FOL query structures without negation ($\exists$, $\wedge$, $\vee$) using these datasets: FB15k, FB15k-237 and NELL995.

Dataset	2in	3in	inp	pin	pni	AVG
FB15k	15.7 ±0.084	16.3 ±0.059	13.7 ±0.086	7.8 ±0.065	13.9 ±0.033	13.5 ±0.020
FB15k-237	4.9 ±0.077	8.3 ±0.093	8.2 ±0.112	3.7 ±0.058	3.4 ±0.073	5.7 ±0.042
NELL995	5.4 ±0.106	7.6 ±0.081	11.3 ±0.095	3.4 ±0.019	3.7 ±0.090	6.3 ±0.046

Table 9: **MRR** (%) results of CylE with error bars for answering negation queries using these datasets: FB15k, FB15k-237 and NELL995.

of ConE (Zhang et al., 2021b) to compute this correlation. We use two types of correlation as (Ren and Leskovec, 2020; Zhang et al., 2021b): (1) Spearman's rank-order correlation coefficient (SRCC) (to measure the monotonicity relationship or the statistical dependence between the rankings of the two variables) and (2) Pearson correlation coefficient (PCC) (to measure the linear relationship between the two variables). We do not compute SRCC and PCC regarding disjunctive queries, which is the same as (Ren and Leskovec, 2020; Zhang et al., 2021b), since we model queries with disjunctions using the DNF technique. Table 10 shows SRCC of Q2B, BetaE, ConE and CylE using the FB15k dataset. No SRCC results are available in G2B for queries with negation since this model cannot handle this operation (Ren et al., 2020). Overall, the SRCC results of CylE are significantly higher

Model	1p	2p	3p	2i	3i	ip	pi	2in	3in	inp	pin	pni
Q2B	0.30	0.22	0.26	0.33	0.27	0.14	0.30	-	-	-	-	-
BetaE	0.37	0.48	0.47	0.57	0.40	0.42	0.52	0.62	0.55	0.46	0.47	0.61
ConE	0.60	0.68	0.70	0.68	0.52	0.56	0.59	**0.84**	0.75	0.61	0.58	0.80
CylE	**0.61**	**0.84**	**0.81**	**0.74**	**0.62**	**0.79**	**0.77**	0.83	**0.78**	**0.67**	**0.66**	**0.83**

Table 10: **Spearman's rank** correlation coefficient between embeddings of learnt embeddings and a number of answers for queries using the dataset FB15k. Results of Q2B, BetaE and ConE are taken from (Zhang et al., 2021b).

Dataset	Model	1p	2p	3p	2i	3i	ip	pi	2in	3in	inp	pin	pni
FB15k-237	Q2B	0.18	0.23	0.27	0.35	0.44	0.20	0.36	-	-	-	-	-
	BetaE	0.41	0.50	0.57	0.60	0.52	0.44	0.54	0.69	0.58	0.51	0.47	0.67
	ConE	0.70	0.71	**0.74**	0.82	0.72	0.62	0.70	**0.90**	**0.83**	**0.66**	0.57	**0.88**
	CylE	**0.71**	**0.80**	0.73	**0.83**	**0.77**	**0.74**	**0.84**	0.84	0.79	0.59	**0.60**	0.83
NELL995	Q2B	0.15	0.29	0.31	0.38	0.41	0.35	0.36	-	-	-	-	-
	BetaE	0.42	0.55	0.56	0.59	0.61	0.54	0.60	0.71	0.60	0.35	0.45	0.64
	ConE	0.56	0.61	0.60	**0.79**	**0.79**	0.58	0.74	**0.90**	**0.79**	0.56	0.48	**0.85**
	CylE	**0.57**	**0.75**	**0.65**	0.73	0.72	**0.70**	**0.76**	0.85	0.76	**0.59**	**0.63**	0.81

Table 11: **Spearman's** rank correlation coefficient between embeddings of learnt embeddings and a number of answers for queries using the dataset FB15k-237 and NELL995. Rank correlation results of Q2B, BetaE and ConE are taken from (Zhang et al., 2021b).

Dataset	Model	1p	2p	3p	2i	3i	ip	pi	2in	3in	inp	pin	pni
FB15k	Q2B	0.08	0.22	0.26	0.29	0.23	0.13	0.25	-	-	-	-	-
	BetaE	0.22	0.36	0.38	0.39	0.30	0.31	0.31	0.44	0.41	0.34	0.36	0.44
	ConE	0.33	0.53	0.59	0.50	0.45	0.42	0.37	0.65	0.55	**0.50**	**0.52**	0.64
	CylE	**0.36**	**0.68**	**0.62**	**0.66**	**0.59**	**0.70**	**0.70**	**0.68**	**0.59**	0.46	0.48	**0.71**
FB15k-237	Q2B	0.02	0.19	0.26	0.37	0.49	0.20	0.34	-	-	-	-	-
	BetaE	0.23	0.37	0.45	0.36	0.31	0.33	0.32	0.46	0.41	0.39	0.36	0.48
	ConE	**0.40**	0.52	**0.61**	0.67	0.69	0.49	0.47	**0.71**	**0.66**	**0.53**	**0.47**	**0.72**
	CylE	0.36	**0.56**	0.53	**0.67**	**0.71**	**0.55**	**0.71**	0.64	0.59	0.41	0.35	0.64
NELL995	Q2B	0.07	0.21	0.31	0.36	0.29	0.34	0.24	-	-	-	-	-
	BetaE	0.24	0.40	0.43	0.40	0.39	0.40	0.40	0.52	0.51	0.26	0.35	0.46
	ConE	**0.48**	0.45	0.49	**0.72**	**0.68**	0.39	0.52	**0.74**	0.66	**0.38**	0.34	**0.69**
	CylE	0.45	**0.60**	**0.50**	0.64	0.63	**0.52**	**0.60**	0.69	**0.66**	0.35	**0.47**	0.64

Table 12: **Pearson** correlation coefficient between embeddings of learnt embeddings and a numer of answers for queries using the dataset FB15k, FB15k-237 and NELL995. Correlation results of Q2B, BetaE and ConE are taken from (Zhang et al., 2021b).

than those in ConE in most of query structures, by a large margin. These results demonstrate the expressiveness of modeling cardinality of answer sets using CylE. Note that SRCC results using CylE also significantly exceed the other previous models.

We show additional results of Spearman's Rank Correlation Coefficient (SRCC) in Table 11 using the dataset FB15k-237 and NELL995. In terms of the dataset FB15k-237, although the SRCC results in several query structures (*e.g.* $3p, 2in, 3in, inp, pni$) using the approach in CylE are lower than those in ConE, the SRCC in the rest of query structures using CylE outperform those in ConE. A slight decrease in SRCC results, from CylE to ConE, can be mostly observed in queries that are involved in negation operations. These observations are due to the fact that the height variable might not play a role in embedded queries with complement operations. Similar to the dataset FB15k-237, in the dataset NELL995, the SRCC results in some query structures (*e.g.* $2i, 3i, 2in, 3in, pni$), particularly in queries with negation operations, are also lower than those in ConE. However, most of SRCC results in queries without negation operations surpass SRCC results

Dataset	Activation	1p	2p	3p	2i	3i	ip	pi	2u	up	AVG
FB15k	ReLU	75.1	36.5	**31.0**	65.8	74.5	39.9	52.5	56.9	**33.6**	51.8
	Swiss	**78.8**	**37.0**	30.9	**66.9**	**75.7**	**40.8**	**53.8**	**59.4**	33.5	**53.0**
FB15k-237	ReLU	42.1	13.1	11.3	34.8	**49.1**	14.8	26.5	14.7	11.2	24.2
	Swiss	**42.9**	**13.3**	**11.3**	**35.0**	49.0	**15.7**	**27.0**	**15.3**	**11.2**	**24.5**
NELL995	ReLU	54.3	16.3	14.2	40.5	51.0	17.9	26.0	14.8	11.3	27.4
	Swiss	**55.7**	**17.5**	**15.1**	**40.7**	**51.1**	**19.1**	**27.1**	**15.4**	**12.2**	**28.2**

Table 13: Average **MRR** (%) results of CylE in different FOL query structures without negation (∃, ∧, ∨) using ReLU and Swiss activation for the MLP in these datasets: FB15k, FB15k-237 and NELL995.

in ConE. In comparisons with BetaE and Q2B, SRCC results in all query structures (with and without negation) using CylE are significantly higher than SRCC results in these models.

The similar trend of results are also observed in the Pearson Correlation Coefficient (PCC) between the aperture embeddings and the cardinality of answers set using the three dataset FB15k, FB15k-237 and NELL995 (see Table 12).

B.3 Further ablation study

We compare the performance of CylE using the different activation function (ReLU and Swiss) for the MLP networks (in the projection and intersection module), during the training process in this ablations study (see Table 13 and 14). Overall, the average MRR results in most of query structures for the approach using Swiss activation are slightly higher than those in the approach using ReLU activation in the three datasets FB15k, FB15k-237 and NELL995. Since the Swiss activation was shown to be an efficient activation function (Ramachandran et al., 2017) for the MLP networks.

Dataset	Activation	2in	3in	inp	pin	pni	AVG
FB15k	ReLU	15.6	16.2	13.4	7.8	13.7	13.3
	Swiss	**15.7**	**16.3**	**13.7**	**7.8**	**13.9**	**13.5**
FB15k-237	ReLU	4.7	8.1	8.1	3.7	3.2	5.6
	Swiss	**4.9**	**8.3**	**8.2**	**3.7**	**3.4**	**5.7**
NELL995	ReLU	5.1	7.5	11.2	3.3	3.5	6.1
	Swiss	**5.4**	**7.6**	**11.3**	**3.4**	**3.7**	**6.3**

Table 14: Average **MRR** (%) results of CylE for different FOL query structures with negation using ReLU and Swiss activation for MLP in these datasets: FB15k, FB15k-237 and NELL995.

C Computational complexity

The computational complexity of CylE is similar to ConE since these models share similarity in geometric shapes. Note that the computational complexity of ConE and G2B is also similar to one another (Zhang et al., 2021b). It is arguably that CylE, ConE and G2B have similar computational complexity. Assuming a Disjunctive Normal Form (DNF) query q: that consists of conjunctive queries $q_{i:1\rightarrow n}$, where $q = q_1 \vee \cdots \vee q_n$. The computational complexity of CylE for answering q is equivalent to the computational complexity for answering the number n of conjunctive queries q_i. This answering process is involved in the estimation of a sequence of geometric sector-cylinder operations in which a constant time can be taken for each operation, then performing a range search which can be achieved using techniques according to Locality Sensitive Hashing (Indyk and Motwani, 1998).

Models	GQE	Q2B	BetaE	ConE	CylE	CylE
Emb. dimension d	800	800	800	800	800	500
Running time (s)	75.87	81.43	168.91	119.90	154.79	121.65

Table 15: Average running time (seconds) for the first 500 training steps in different approaches using the FB15k-237 dataset.

We record the average running time (seconds) for the first 500 training steps using the same dimensional embedding for all models (GQE, G2B, BetaE, ConE and CylE) and another one with lower embedding dimension for CylE, on the same single NVIDIA Tesla V100 GPU, for fair comparisons. The lower this value is, the faster training process is. Overall, the fastest model is GQE while the slowest model is BETA. The running time of CylE is slightly slower than ConE, but CylE with lower dimension ($d = 500$) is on par with ConE. Further, the running time of CylE is also faster than BetaE.

D The range values for the height

The range of values for the height variable $\boldsymbol{\theta}_{he}$ for an arbitrary embedded query $\mathbf{V}_q = (\boldsymbol{\theta}_{ax}, \boldsymbol{\theta}_{ap}, \boldsymbol{\theta}_{he})$ can be varied without any constraints. For example, this range of values can be infinite. However, there is a numerical problem in a way that values of $\boldsymbol{\theta}_{he}$

might dominate those in the semantic center $\boldsymbol{\theta}_{ax}$ and the aperture $\boldsymbol{\theta}_{ap}$. This problem can lead to a reduction in the performance of MHR in this situation. Since $\boldsymbol{\theta}_{ax} \in [-\pi, \pi)^d$ and $\boldsymbol{\theta}_{ap} \in [0, 2\pi]^d$ are in a small range of values, compared to infinite range of $\boldsymbol{\theta}_{he}$. Thus, we set a small range of values for the height variable and scale the range to $(-\pi, \pi)$, to avoid the numerical issues, making this variable have a consistent systematic range of values (based on the multiples of π) as those in the semantic center $\boldsymbol{\theta}_{ax}$ and the aperture $\boldsymbol{\theta}_{ap}$.

Fiction-Writing Mode: An Effective Control for Human-Machine Collaborative Writing

Wenjie Zhong[1,3], **Jason Naradowsky**[1], **Hiroya Takamura**[3]
Ichiro Kobayashi[2,3], **Yusuke Miyao**[1,3]
[1]The University of Tokyo, [2]Ochanomizu University,
[3]National Institute of Advanced Industrial Science and Technology
{zvengin, narad, yusuke}@is.s.u-tokyo.ac.jp
takamura.hiroya@aist.go.jp, koba@is.ocha.ac.jp

Abstract

We explore the idea of incorporating concepts from writing skills curricula into human-machine collaborative writing scenarios, focusing on adding *writing modes* as a control for text generation models. Using crowd-sourced workers, we annotate a corpus of narrative text paragraphs with writing mode labels. Classifiers trained on this data achieve an average accuracy of $\sim 87\%$ on held-out data. We fine-tune a set of large language models to condition on writing mode labels, and show that the generated text is recognized as belonging to the specified mode with high accuracy.

To study the ability of writing modes to provide fine-grained control over generated text, we devise a novel turn-based text reconstruction game to evaluate the difference between the generated text and the author's intention. We show that authors prefer text suggestions made by writing mode-controlled models on average 61.1% of the time, with satisfaction scores 0.5 higher on a 5-point ordinal scale. When evaluated by humans, stories generated via collaboration with writing mode-controlled models achieve high similarity with the professionally written target story. We conclude by identifying the most common mistakes found in the generated stories. The datasets and codes are available at the Github[1].

1 Introduction

Large-scale pre-trained language models (PLMs) have demonstrated a remarkable aptitude for generating text with an exceptional degree of fluency and structure (Guan et al., 2021; Tan et al., 2021), sparking renewed efforts to utilize them for the purpose of generating narrative fiction. Recent work has explored various ways of controlling PLMs, using sentiment (Luo et al., 2019), style (Kong et al., 2021a), and even character information (Liu et al., 2020a), in an attempt to cater the generated text to an author's intentions.

Summary: A boy was walking along a path in the forest, when he came across a heap of leaves.

Mode	Story
Dialogue	"Here are some leaves," he **whispered**. "They were wet when we came, and are wet now. I'll lie them down and wait." "What is it?" I **exclaimed**. "If you will stand still," **said** my boy, "I will show you … "
Action	He **stood** for a moment **looking** at me, then quietly he **picked up** the leaves, and **carrying** them in his hand, **climbed** to the top of the heap, and **examined** them …
Description	This heap consisted of **dead leaves**, many of them **rotten**, and still **wet**, with one or two **lying flat** on the ground, others **lying up against** the branches. The first to fall was the one I had thought **dead**. It had been crushed by the wind. …

Figure 1: Example of expanding the **Summary** into stories using different writing **Mode**s. The bold words imply the particular manner of expression in that mode. **Dialogue** focuses on the utterances spoken by characters, **Action** on the motion of characters, and **Description** on the depiction of characters or places.

However, the aforementioned controls deal primarily with *static* text attributes; an attribute like style is more synonymous with an entire author or book than with a single passage of text. Less attention has been paid to designing effective control factors for the real demands of human authors in collaborative writing settings, where authors typically exercise more *dynamic* control over their writing, varying attributes of the text at the sentence or paragraph level. Here we find inspiration from the creative writing literature, where the notion of a *fiction writing mode* is often presented as an important concept to consider when crafting narrative fiction (Klaassen, 2015).

A fiction-writing mode (also referred to as a rhetorical mode) is a particular manner of writ-

---

[1] https://github.com/ZVengin/fride

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1752–1765
May 2-6, 2023 ©2023 Association for Computational Linguistics

Dia.	Act.	Des.	Unc.	Total	Len.(std)	Kappa
370	385	300	681	1,736	110(52)	0.64

Table 1: The number of instances for dialogue (**Dia.**), action (**Act.**), description (**Des.**), and uncertain (**Unc.**) modes in the dataset. **Kappa** is the inter-annotator agreement and **Len.(std)** is the average token number in each instance and its standard deviation.

ing, encapsulating the focus, style, and pacing of the text (among other things). Figure 1 illustrates how the same event can be described in different ways depending on the writing mode, using the three most common types, *Dialogue*, *Action*, and *Description*. Skilled authors proficiently use writing modes as a stylistic choice to engage readers and progress the narrative (see Section 2 for more detail) (Klaassen, 2015). Thus, we hypothesize that conditioning text generation models on writing modes can provide important controls to authors in a human-machine collaborative writing scenario.

To verify this hypothesis, we are faced with two challenges. First, to the best of our knowledge, there is no available dataset annotated with writing modes to train generation models. We create a Fiction wRIting moDE dataset (**FRIDE** dataset) containing 1,736 fiction paragraphs annotated by crowd-source workers with the three writing mode labels. Subsequently, we train a classifier on the **FRIDE** dataset and use it to annotate paragraphs of a large fiction corpus in order to create a larger-scale dataset. Using the established paradigm of training conditional text generation models by summarizing and reconstructing text (Sun et al., 2020), the dataset is used to train models which can be conditioned on a writing mode label.

Second, to measure whether writing modes allow for more effective control of text generation models, we need to compare the generated texts to the author's intention, which is unobservable. We design a new evaluation framework for human-machine collaborative writing where the author is given a paragraph of text, and is asked to recreate it solely through interaction with generative models. By using the paragraph as a proxy for the author's intention, we are able to assess the similarities between the intention and the generated text, and analyze the differences as indications of where current controls fail.

Through both automatic and human evaluation we show: (1) the use of writing mode labels with conditional text generation models contributes to average 1.4 and 2.0 points improvements on ROUGE-L and BERTScore; (2) the writing modes of generated text are effectively controlled, and are classified as belonging to the target mode in 87.6% of cases; (3) authors are 22.2% more likely to choose the suggestions from writing mode-controlled models, and assign them an average 0.5 higher satisfaction score (on an ordinal 1-5 scale) compared to the uncontrolled models; (4) applying writing mode control to collaborative writing enhances the similarities between the generated text and the authors' intention.

2 Fiction-Writing Mode Dataset

Fiction-writing modes have long been proposed as a useful abstraction in the study of literature and creative writing (Morrell, 2006; Klaassen, 2015), dating as far back as Aristotle (Halliwell and Aristotle, 1998). While there is no consensus on the categorization of writing modes, most sources prefer to introduce at least three modes: (1) *Dialogue*, direct quotation of characters speaking, (2) *Action*, an account of a series of events, one after another, chronologically, and (3) *Description*, a more detailed inspection of people, places, or things and their properties. These are the three major writing modes which are the focus of study in this paper.

Just as there is no agreement on how best to categorize writing modes, there is also no consensus on what text exhibits a particular mode. Even a single sentence can exhibit multiple writing modes, in varying degrees. However, for the purpose of this work, we assume that each paragraph can be categorized as exhibiting a single writing mode.

FRIDE Dataset In order to train models which generate text in a specified writing mode, we must first create a dataset, which we refer to as Fiction-wRIting moDE dataset (**FRIDE** dataset), which pairs paragraphs of narrative text with their corresponding writing mode labels. However, directly annotating writing modes on a large-scale narrative dataset is expensive and time-consuming. We first collect a modestly sized dataset from crowd-sourced workers, and utilize it to train a writing mode classifier. The classifier can then be used to provide high confidence labels to a much larger dataset of narrative text paragraphs, on a scale suitable for training large text generation models.

Paragraphs for annotation are collected from fic-

	Precision	Recall	F1
BERT-base	85.7	85.0	85.0
XLNet-base	84.7	84.4	84.3
RoBERTa-base	**86.3**	**85.2**	**85.2**

Table 2: The performance of writing mode classifiers on the **FRIDE** dataset.

tion books sourced from Project Gutenberg[2] (128 books) and, for more contemporary writing, Smashwords[3] (150 books). Each book is divided into paragraphs using Chapterize[4], and paragraphs longer than 200 words are removed. In situations where a continuous dialogue takes place over paragraph boundaries, we group them into a single paragraph. Each paragraph was annotated with one of the three aforementioned writing modes using Amazon Mechanical Turk (AMT). In addition, we add a fourth category, *Uncertain*, to encompass cases where the writing mode is unclear or does not fit well into the three main modes. All annotators were native English speakers, and three annotators were assigned to each paragraph. Paragraphs were assigned the majority label, or marked as *uncertain* in cases where each annotator provided a different label. We continued the annotation process until we had approximately 1,000 instances labeled and balanced across the three main modes (Table 1).

Writing Mode Classifier While it is possible to use the collected data to train a model, the relatively small pool of examples may cause the model to be sensitive to other text characteristics unrelated to the writing mode. To help alleviate this problem, we train a writing mode classifier and employ it to predict writing modes on a larger collection of texts. We experiment with training three separate classifiers, each trained by fine-tuning a different PLM (BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), or RoBERTa (Liu et al., 2019)) on the **FRIDE** dataset. We randomly sample 300 instances from each type of writing mode and divide them using a 1000/100/100 train/dev/test split, with an equal number of each label in each split. An evaluation of these models (Table 2) shows that all models perform similarly. The RoBERTa-based model was used as the final writing mode classifier throughout the remainder of this paper.

FRIDE-XL Dataset In order to construct a larger dataset of writing modes suitable for training mode-conditional text generation models, we utilize the classifier trained in the preceding section on a larger set of texts, extending the previous text to 5,946 fiction books from Project Gutenberg. We leverage the writing mode classifier to assign a writing mode label to each paragraph of books and randomly select 362,880 paragraphs. We refer to this dataset as **FRIDE-XL**. Additional statistics of the dataset are provided in Table 7 of Appendix.

To test the accuracy of the classifier on the **FRIDE-XL** dataset, we ask three additional annotators to label 150 instances taken from the **FRIDE-XL** dataset. The results show the inter-rater agreement between annotators is 0.73. The predictions of the trained classifier agree with the majority label provided by the annotators in 85.1% instances. In terms of accuracy, classifier F1 on **FRIDE-XL** is 83.9, compared to 85.2 on the **FRIDE** dataset. It is important to reiterate that text often reflects multiple writing modes to varying degrees, and so some disagreement is inherent in the task. We find that the writing mode predicted by the classifier fails to match any label provided by human annotators in only 4.8% of cases. In Section 4.2 we show that this is indeed sufficient accuracy for producing the data and models necessary for generating mode-specific text.

3 Models

We evaluate writing mode as a control factor on three different PLM architectures: BART, (Fan et al., 2018), T5 (Raffel et al., 2020), and GPT2 (Radford et al., 2018). All models have been used previously for text generation but differ in ways that may impact their ability to adhere to the conditioning information and the quality of the generated text. For instance, the comparatively larger size and contextual window of GPT2 has made it a common choice for story generation with long text (Wang et al., 2021; Clark and Smith, 2021; Akoury et al., 2020), but smaller models like T5 show great controllability (Clive et al., 2021). We assess each of these three models, fine-tuning them to reconstruct paragraphs from the **FRIDE-XL** dataset.

For training conditional text generation models, we follow an established paradigm of summarization, conditioning, and reconstruction, as used by Sun et al. (2020). First, each paragraph is summarized using an existing summarization model.

---

[2] https://www.gutenberg.org

[3] https://www.smashwords.com

[4] https://github.com/JonathanReeve/chapterize

	Model Inputs			Quality				Controllability (Accuracy)		
	L	**S**	**M**	**PPL** ↓	**B4** ↑	**RL** ↑	**BS** ↑	**Dialogue**	**Action**	**Description**
GPT2				24.41	0.95	15.02	45.61	73.33	20.28	21.67
FIST		✓		23.68	0.99	15.41	45.97	85.00	41.11	46.94
PPLM			✓	24.10	1.07	14.50	42.84	93.05	32.22	49.72
GPT2	✓			**19.29**	1.06	15.74	46.33	72.50	22.78	22.50
	✓	✓		18.84	1.14	15.97	46.70	83.33	38.61	45.56
	✓		✓	20.29	1.09	16.00	46.82	97.78	67.78	71.11
	✓	✓	✓	19.89	1.16	16.10	47.28	98.06	75.00	79.72
T5	✓			24.98	1.14	16.22	46.42	67.78	25.28	23.61
	✓	✓		23.80	1.21	16.32	46.78	80.56	47.78	46.67
	✓		✓	26.12	1.16	16.30	47.07	**99.44**	85.00	78.33
	✓	✓	✓	25.06	**1.20**	**16.52**	47.10	98.33	83.61	83.06
BART	✓			23.87	1.07	16.19	46.32	69.44	19.17	18.33
	✓	✓		23.49	1.17	16.33	47.12	86.67	47.78	48.89
	✓		✓	25.44	1.11	16.25	**47.30**	98.06	82.50	**88.06**
	✓	✓	✓	24.24	**1.20**	16.27	**47.30**	97.78	**85.56**	82.78

Table 3: Automatic evaluation on quality and controllability as model inputs (summaries (**S**), length (**L**), and writing modes (**M**)) vary. Quality is evaluated by perplexity (**PPL**), BLEU-4 **(B4)**, ROUGE-L (**RL**), BERTScore (**BS**), and controllability is measured by the accuracy of the generated stories matching the specified writing modes when the writing modes (**M**) are specified as **Dialogue**, **Action**, and **Description**. The inputs such as summaries (**S**), length (**L**), and writing modes (**M**) for the evaluation of quality and controllability are respectively inferred from the leading context and the target stories.

Here we use the narrative text summarization proposed in Kryscinski et al. (2021), and decode using beam search with a beam size of 5 as in that work. We then fine-tune a PLM to reconstruct the original paragraph, conditioning on the summary. In this way, the summary acts as a semantic control: the trained model accepts user summaries and attempts to expand upon them to generate a longer paragraph, embellishing missing and less important details in a reasonable way.

Other forms of information can also be added to the summaries to function as additional controls. The conditioning factors provided to models are:

- Summary, generated from the paragraph by a pre-trained model as shown in Appendix.
- Context, the preceding paragraph.
- Length, the number of tokens in the paragraph divided into ten equally-sized bins.
- Writing Mode, the mode assigned to the paragraph by the classifier as described in Sec. 2.

For T5 and BART, the training methodology is straightforward: we concatenate the controlling information and use it as input to the encoder, training the decoder to generate the original paragraph. For GPT2, which has only a decoder, we concatenate the conditioning information as prompts.

4 Automatic Evaluation

In this section, we study the influence of model inputs (e.g., summaries, length, and writing modes) on the quality of text, and assess to what extent the writing modes of text can be controlled, as measured by automatic evaluation metrics.

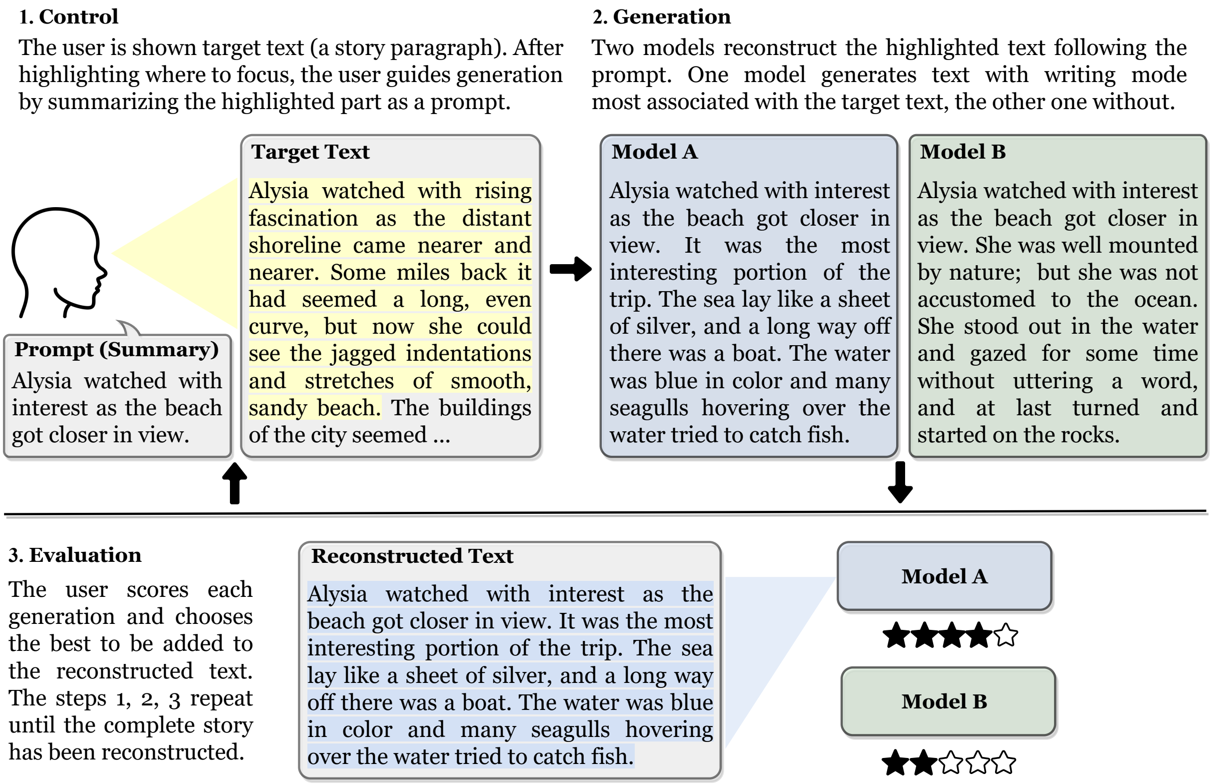
4.1 Baseline Models

In addition to ablations of our proposed models, we compare against three baseline systems:

GPT2 We finetune GPT2 (Radford et al., 2018) identically to our proposed system, but using only the preceding paragraph and without other inputs.

PPLM Dathathri et al. (2020) employ an attribute classifier to guide the pretrained language model to generate text with specified attributes. To adapt the PPLM to our task, we train a writing mode classifier as the attribute classifier on the **FRIDE** dataset. As the writing modes of preceding paragraphs would interfere with the classifier, the PPLM does not take the preceding paragraphs as context.

FIST Fang et al. (2021) propose a system which utilizes keywords instead of summaries to sketch the semantic content of the desired stories. As there is no prompt in our dataset, following their idea, we infer the keywords from the leading context (the



1. Control The user is shown target text (a story paragraph). After highlighting where to focus, the user guides generation by summarizing the highlighted part as a prompt.

Target Text Alysia watched with rising fascination as the distant shoreline came nearer and nearer. Some miles back it had seemed a long, even curve, but now she could see the jagged indentations and stretches of smooth, sandy beach. The buildings of the city seemed ...

Prompt (Summary) Alysia watched with interest as the beach got closer in view.

2. Generation Two models reconstruct the highlighted text following the prompt. One model generates text with writing mode most associated with the target text, the other one without.

Model A Alysia watched with interest as the beach got closer in view. It was the most interesting portion of the trip. The sea lay like a sheet of silver, and a long way off there was a boat. The water was blue in color and many seagulls hovering over the water tried to catch fish.

Model B Alysia watched with interest as the beach got closer in view. She was well mounted by nature; but she was not accustomed to the ocean. She stood out in the water and gazed for some time without uttering a word, and at last turned and started on the rocks.

3. Evaluation The user scores each generation and chooses the best to be added to the reconstructed text. The steps 1, 2, 3 repeat until the complete story has been reconstructed.

Reconstructed Text Alysia watched with interest as the beach got closer in view. It was the most interesting portion of the trip. The sea lay like a sheet of silver, and a long way off there was a boat. The water was blue in color and many seagulls hovering over the water tried to catch fish.

Figure 2: A turn-based text reconstruction game. Users aim to reconstruct the target text solely through interaction with story generation models, providing prompts (summaries) to guide generation, and choose between a number of options to continue the text. Each option is generated by a different model. Users report satisfaction at each turn, and the reconstructed text is used in further evaluation.

preceding paragraphs) and then generate stories conditioning on the context and keywords.

4.2 Results

We evaluate the models along three axes: fluency, similarity, and controllability, using the test set of the **FRIDE-XL** dataset. The results of our automatic evaluation are shown in Table 3.

Fluency We evaluate fluency using perplexity computed by the pre-trained GPT2 model. We find that there is an average 0.8 decrease in perplexity when summaries are added and 1.2 increase when writing modes are added.

Similarity Our task is very similar to summarization, so we adopt the same evaluation metrics including BLEU-4 (Papineni et al., 2002), ROUGE-L (Lin, 2004), and BERTScore (Zhang et al., 2020). It is also quite common to include these automatic measures in narrative text generation work, despite their known flaws. We observe a consistent improvement across all models as the amount of conditioning context increases, and the models using writing mode factors outperform those without.

Controllability Lastly, we evaluate the controllability of mode-controlled models. For each paragraph, a target writing mode is chosen using the writing mode classifier, and used as conditioning for a text generation model. The classifier is then used to predict the writing mode of the generated text, and we measure the accuracy of generating stories with the specified writing modes.

On average, including writing mode as condition improves the accuracy of generating text which is classified as that mode, but the effect varies drastically by the specific mode. For Action and Description modes, the inclusion of writing mode conditioning improves accuracy on average by 45.4% and 45.6%, respectively, compared to their non-mode counterparts. For dialogue, the improvement is 21.5%, relatively lower.

It is interesting to note that the inclusion of summaries to the length-only model results in significant improvements to the controllability of the text. This implies that the pre-trained models are able to naturally infer the intended writing mode from the summaries to some degree, with modest accuracy ($\sim 44\%$) on average for Action and De-

Model	Writing Mode Control	Selection Percentage			Satisfaction Score		
		Dialogue	Action	Description	Dialogue	Action	Description
GPT2	✗	31	44	32	2.27	2.84	2.67
	✓	**69**	**56**	**68**	**2.96**	**3.27**	**3.59**
BART	✗	36	39	43	2.82	2.64	3.10
	✓	**64**	**61**	**57**	**3.28**	**3.11**	**3.34**
T5	✗	47	36	44	3.19	2.91	3.22
	✓	**53**	**64**	**56**	**3.24**	**3.36**	**3.55**

Table 4: Human evaluation by authors, showing **Selection Percentage** and **Satisfaction Score** between generated text w/ and w/o writing mode conditioning, during the text reconstruction game. **Dialogue**, **Action**, and **Description** are the writing modes of the target text.

scription modes, and up to 86% on Dialogue with BART. Summaries may contain some cues about the intended modes, especially, the summaries for Dialogue have strong cues (*said*, *replied*, *argued*, ...) in most cases. However, the consistent large margins of accuracy scores when conditioning on writing modes illustrates the effectiveness of modes as a control factor.

5 Human Evaluation

In the preceding section, we used automatic measures to show consistent improvements in similarity and controllability when conditioning text generation models on writing modes. In this section, we assess whether these improvements translate into higher author satisfaction and story quality as judged by human participants.

Text Reconstruction Game To evaluate writing mode-controlled models within human-machine collaborative writing, we devise a story generation game (Figure 2). Using a web interface, participants are presented with a target story, drawn from **FRIDE-XL**, and asked to reconstruct it through interaction with the trained models. Each session utilizes two models, each using the same PLM, one trained to condition on writing modes, and one that does not.

At each round, (1) the author highlights a selection of text from the target story, and provides a text summary which serves as a rough sketch of the story described therein. Constraints on the summary length and a time limit on each session force the author to extract essential elements of the text and prevent them from simply specifying the entire text as the summary. (2) The models then each return a sample of generated text, and (3) the author is asked to score each in terms of overall satisfaction (ordinal, 5-point scale), and choose one to continue the story. The source of the suggestions (model name) is not given to the participant, and they are presented in random order. The above process repeats until the author has attempted to recreate the entire target text. The author is then asked to provide feedback on overall satisfaction with the story and interface, the efficiency of using the interface over generating the text from scratch, and what errors were present in the generated text.

Our approach takes inspiration from previous human-in-the-loop evaluations (Akoury et al., 2020; Clark and Smith, 2021) where authors are asked to construct stories by collaborating with story generation models, but do so freely, without being provided any specific direction for how the story should unfold. Therefore we cannot observe the author's intention, and it is difficult to ascertain how closely the generated stories match the author's original intention, or the extent to which bias may be affecting the model scores when satisfaction with the models is self-reported.

Our solution to these issues is to present the author with a target text they must attempt to reconstruct. The target text serves as a substitute for the author's intention, allowing us to more objectively measure the differences between the generated story and the target. While this design shifts the nature of interaction away from a more creative use case, we argue that increased awareness of the models' limitations provides a worthwhile trade-off when used strictly as a means of evaluation.

Using the unique properties of our proposed evaluation design, we evaluate: (1) the effectiveness of writing mode control when generating text as an author, (2) their effectiveness from the perspective of a reader, comparing generated stories with target stories in a blind study, and (3) the types of errors made where authors were unable to exert desired

	Collab.	**Auto.**	**P**	**+WM**	**-WM**	**P**
Plot Similarity						
GPT2	**3.60**	3.04	0.01	**3.42**	3.30	0.51
BART	3.59	**3.66**	0.71	**3.56.**	3.22	0.06
T5	**3.49.**	3.05	0.01	**3.48.**	3.10	0.02
Style Similarity						
GPT2	**3.21.**	3.05	0.43	**3.48**	3.15	0.02
BART	**3.80.**	3.59	0.28	**3.71**	3.30	0.02
T5	**3.56**	3.19	0.03	**3.48**	3.08	0.02

Table 5: The similarity scores (5 points scale) in plots and style between the reconstructed text and the target ones. **Collab.** and **Auto.** refer to the text collaboratively reconstructed with the users and automatically generated by the models. **+WM** and **-WM** are the text automatically reconstructed by models with or without writing modes. **P** is the P-Value of significance test.

control over the models.

5.1 Evaluation by Authors

We randomly sample 36 target stories (each annotated with a writing mode) from **FRIDE-XL**, such that there is an equal number of stories from each mode and each genre. For the writing mode-controlled model, we use the mode annotated in the dataset as the true mode (users are not asked to specify a mode explicitly). The authors are recruited via crowdsourcing on AMT, and each target story is reconstructed using the proposed interface, with each story given to three annotators.

Do authors prefer the suggestions from writing mode-controlled models? We measure authors' preferences for models by the win/loss rate for which model was selected to continue the story at each step, and the average satisfaction score of each model's suggestions. The results are presented in Table 4. Suggestions of writing mode models are preferred consistently, irrespective of model type, and are chosen on average 22.2% more than suggestions from uncontrolled models. In terms of satisfaction score, using writing mode as a conditioning factor improves satisfaction by 0.5 / 5.0 points on average, an improvement consistent across all writing modes. We conclude that the strong and consistent improvements when using writing mode-conditional models demonstrate they are an effective control for story generation.

5.2 Evaluation by Readers

Evaluation by authors using the story reconstruction game interface provides compelling evidence for the effectiveness of writing modes as a control, but how close to the author's intention are the stories generated using the writing interface? By providing authors with a target story, new evaluation methods are possible, such as having the stories scored by a separate group of participants, in order to avoid any bias from self-reporting.

Here we evaluate the generated texts via similarity with the target texts, in two different scenarios. We enlist 197 human participants from AMT, none of which participated in the story generation task, to serve as readers. We present each reader with three stories: the target story, the generated story, and a baseline story. Readers are asked to score the similarity between the target text and the other stories on an ordinal 5-point scale, in terms of plot and style.

How similar to the target text are the generated stories? The generated stories are written interactively via the writing interface, where authors have access to suggestions from mode and non-mode models. The baseline stories are generated in a purely automated manner using the model to predict the entire paragraph from a summary (generated by the same summarization model we use in the fine-tuning process), together with the writing modes provided from the dataset annotations.

The left side of Table 5 shows the results. We observe higher similarity scores between the target text and the stories generated via collaborative writing in almost all cases, an average increase of 0.31 on plot, and 0.25 on style. On one hand, this is an expected conclusion given that the collaborative process allows authors to select the best of two automatic suggestions at each point, and one may expect the quality of collaborative writing stories to be strictly better than automatic ones. However, we do note trends based on model type; BART in particular scores high (3.80) in terms of similarity on style.

How effective is writing mode control for automatic generation While our focus is primarily on the use of additional controls in collaborative story writing, we also measure the effectiveness of writing modes as a useful control in a purely automatic sense. Using the story generation models, as above, we generate full stories without writing mode conditioning and contrast them with the previously generated stories generated automatically using writing modes. We report the similarity of

Error Type	GPT2	BART	T5
Missing Information	61	65	32
Irrelevant Information	45	40	22
Wrong Information	8	8	2
Incoherence	22	22	22
Inconsistency	22	13	20
Disfluency	15	27	18
Repetition	0	0	0

Table 6: The counts of each error occurring in the reconstructed text across different models.

each to the target text in the right side of Table 5.

We again find that writing mode-controlled models significantly improve the control of the generated text, producing stories closer to the target text in both style and content. The addition of a writing mode control improves average similarity to target text by 0.28 on plot and 0.38 in style. Although writing modes are more inherently tied to the style of the text, it is interesting to observe improvements in plot similarity as well. This may indicate that the mode has a positive effect in pressuring the model to focus on summary plot points, via explicitly disentangling these factors. However, we leave further analysis of this phenomenon to future work.

5.3 Error Analysis

To understand what errors are likely to occur in the generated stories we ask annotators to identify aspects of the generated text which differed from the target, from a set of pre-determined categories (Table 6). The most frequent errors are the *missing information* and *irrelevant information*, suggesting that the models extrapolated from the summaries in undesirable ways.

Note that the counts of most errors made by T5 are appreciably lower than those made by GPT-2 or BART, yet BART has higher similarity scores in both plot and style. We infer that some errors made by T5 must play a more important role in overall similarity, and that missing or irrelevant information must not play a crucial role in similarity metrics.

6 Related Work

Our work is based on prior research in computational modeling for story generation. Early approaches to automatic story generation relied on graph-based planning and hand-crafted rules to structure narratives (Meehan, 1977; Callaway and Lester, 2002; Riedl and Young, 2004; Li et al., 2013). More recent works generate stories by finetuning on large-scale PLMs (See et al., 2019) to improve its fluency and incorporating structured knowledge such as planned events (Chen et al., 2021; Fang et al., 2021; Li et al., 2022), summaries (Yao et al., 2019; Tan et al., 2021; Sun et al., 2020), and external knowledge (Guan et al., 2019; Xu et al., 2020b; Guan et al., 2020) to enhance its coherence and consistency. Our story generation models are also finetuned on the large-scale PLMs to generate text following the given summaries.

Our work bears similarity to work on controllable story generation, which aims to control different attributes of stories such as the sentiments (Luo et al., 2019; Kong et al., 2021b), genres (Cho et al., 2022), intention (Sun et al., 2021), and characters (Lee and Jung, 2020; Xu et al., 2020a; Liu et al., 2020b) of stories. However, these attributes are largely unchanging throughout the story, while we focus on writing mode, a more dynamic attribute of text. Thus our work is also more inherently related to interactive story generation where the author works closely with the model to craft text on a comparatively finer level of granularity (sentences or paragraphs).

Finally, our evaluation method is inspired by work on human-in-the-loop storytelling (Roemmele and Gordon, 2015; Samuel et al., 2016; Clark et al., 2018; Goldfarb-Tarrant et al., 2019; Brahman et al., 2020), where the authors are asked to work in concert with story generation models, curating their suggestions to craft the final story. Human-in-the-loop evaluations overcome many of the shortcomings of automatic evaluations, which capture rough statistics, but may be unaware of important errors in plot development and story continuity (Sagarkar et al., 2018). By asking users to select between models' suggestions, we can instead gain a more accurate picture of which system improvements yield real benefits to a potential human-machine writing collaboration (Akoury et al., 2020; Clark and Smith, 2021; Khashabi et al., 2021). Our approach is similar to this, but the addition of a target text allows us to examine the difference between the generated and intended text, which we argue is a more important comparison when dealing specifically with understanding *control*.

7 Conclusion

In this work, we introduced writing modes as a control for human-machine collaborative writing scenarios and showed that training models to condition on writing modes resulted in stories that were closer to targets. Both the automatic and human evaluation shows that the writing modes of text are effectively controlled, authors prefer text suggestions made by writing mode-controlled models, and readers score stories to be more similar to targets in terms of both plot and style. To verify the hypothesis, we collected **FRIDE** and **FRIDE-XL**, datasets of narrative text annotated with writing modes, which we released to help facilitate further research in writing modes and fine-grained control for storytelling. In future work, we wish to apply reconstruction-based evaluation for other factors of human-machine storywriting, and incorporate a dynamic use of writing modes into fully automatic hierarchical story generation models.

Limitations

This work is subject to known biases in the dataset used throughout this work. Due to existing copyrights on most contemporary examples of professional narrative fiction, researchers often turn to works in the public domain, as we do here. While many public domain novels are literary classics, the lack of comparable contemporary work results in models which are biased towards reproducing older works, both in terms of style and content. More contemporary approaches to writing style are not represented in our work, and plot points may be biased by the worldview of the authors at the time.

Acknowledgements

This paper is based on results obtained from a project JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). This work is also partially supported by JSPS KAKENHI Grant Number JP19H05692. We sincerely thank Yusuke Mori, Yang Zhao, and the anonymous reviewers for their advice and feedback on earlier drafts of this work.

References

Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. 2020. STORIUM: A dataset and evaluation platform for machine-in-the-loop story generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6470–6484. Association for Computational Linguistics.

Faeze Brahman, Alexandru Petrusca, and Snigdha Chaturvedi. 2020. Cue me in: Content-inducing approaches to interactive story generation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*, pages 588–597. Association for Computational Linguistics.

Charles B Callaway and James C Lester. 2002. Narrative prose generation. *Artificial Intelligence*, 139(2):213–252.

Hong Chen, Raphael Shu, Hiroya Takamura, and Hideki Nakayama. 2021. Graphplan: Story generation by planning with event graph. In *Proceedings of the 14th International Conference on Natural Language Generation, INLG 2021, Aberdeen, Scotland, UK, 20-24 September, 2021*, pages 377–386. Association for Computational Linguistics.

Jin-Uk Cho, Min-Su Jeong, JinYeong Bak, and Yun-Gyung Cheong. 2022. Genre-controllable story generation via supervised contrastive learning. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 2839–2849. ACM.

Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018. Creative writing with a machine in the loop: Case studies on slogans and stories. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces, IUI 2018, Tokyo, Japan, March 07-11, 2018*, pages 329–340. ACM.

Elizabeth Clark and Noah A. Smith. 2021. Choose your own adventure: Paired suggestions in collaborative writing for evaluating story generation models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3566–3575. Association for Computational Linguistics.

Jordan Clive, Kris Cao, and Marek Rei. 2021. Control prefixes for text generation. *CoRR*, abs/2110.08329.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA,* *June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 889–898. Association for Computational Linguistics.

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Outline to story: Fine-grained controllable story generation from cascaded events. *CoRR*, abs/2101.00822.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 89–97. Association for Computational Linguistics.

Jian Guan, Fei Huang, Minlie Huang, Zhihao Zhao, and Xiaoyan Zhu. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Trans. Assoc. Comput. Linguistics*, 8:93–108.

Jian Guan, Xiaoxi Mao, Changjie Fan, Zitao Liu, Wenbiao Ding, and Minlie Huang. 2021. Long text generation by modeling sentence-level and discourse-level coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6379–6393. Association for Computational Linguistics.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6473–6480. AAAI Press.

Stephen Halliwell and Aristotle. 1998. *Aristotle's Poetics*. University of Chicago Press, Chicago.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A. Smith, and Daniel S. Weld. 2021. GENIE: A leaderboard for human-in-the-loop evaluation of text generation. *CoRR*, abs/2101.06561.

M. Klaassen. 2015. *Fiction-Writing Modes: Eleven Essential Tools for Bringing Your Story to Life*. Bookbaby.

Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. 2021a. Stylized story generation with style-guided planning. *arXiv preprint arXiv:2105.08625*.

Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. 2021b. Stylized story generation with style-guided planning. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2430–2436. Association for Computational Linguistics.

Wojciech Kryscinski, Nazneen Fatema Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir R. Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *CoRR*, abs/2105.08209.

O-Joun Lee and Jason J. Jung. 2020. Story embedding: Learning distributed representations of stories based on character networks (extended abstract). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 5070–5074. ijcai.org.

Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. 2013. Story generation with crowdsourced plot graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 27.

Qintong Li, Piji Li, Wei Bi, Zhaochun Ren, Yuxuan Lai, and Lingpeng Kong. 2022. Event transition planning for open-ended text generation. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3412–3426. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan. 2020a. A character-centric neural model for automated story generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1725–1732.

Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan. 2020b. A character-centric neural model for automated story generation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI*

Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 1725–1732. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Fuli Luo, Damai Dai, Pengcheng Yang, Tianyu Liu, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. Learning to control the fine-grained sentiment for story ending generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6020–6026. Association for Computational Linguistics.

James R Meehan. 1977. Tale-spin, an interactive program that writes stories. In *Ijcai*, volume 77, page 9198.

Jessica Morrell. 2006. *Between the lines: Master the subtle elements of fiction writing*. Penguin.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Mark Owen Riedl and R Michael Young. 2004. An intent-driven planner for multi-agent story generation. In *Autonomous Agents and Multiagent Systems, International Joint Conference on*, volume 2, pages 186–193. IEEE Computer Society.

Melissa Roemmele and Andrew S. Gordon. 2015. Creative help: A story writing assistant. In *Interactive Storytelling - 8th International Conference on Interactive Digital Storytelling, ICIDS 2015, Copenhagen, Denmark, November 30 - December 4, 2015, Proceedings*, volume 9445 of *Lecture Notes in Computer Science*, pages 81–92. Springer.

Manasvi Sagarkar, John Wieting, Lifu Tu, and Kevin Gimpel. 2018. Quality signals in generated stories. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics, *SEM@NAACL-HLT 2018, New Orleans, Louisiana, USA, June 5-6, 2018*, pages 192–202. Association for Computational Linguistics.

Ben Samuel, Michael Mateas, and Noah Wardrip-Fruin. 2016. The design of writing buddy: A mixed-initiative approach towards computational story collaboration. In *Interactive Storytelling - 9th International Conference on Interactive Digital Storytelling, ICIDS 2016, Los Angeles, CA, USA, November 15-18, 2016, Proceedings*, volume 10045 of *Lecture Notes in Computer Science*, pages 388–396.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019. Do massively pretrained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019*, pages 843–861. Association for Computational Linguistics.

Simeng Sun, Wenlong Zhao, Varun Manjunatha, Rajiv Jain, Vlad I. Morariu, Franck Dernoncourt, Balaji Vasan Srinivasan, and Mohit Iyyer. 2021. IGA: an intent-guided authoring assistant. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5972–5985. Association for Computational Linguistics.

Xiaofei Sun, Chun Fan, Zijun Sun, Yuxian Meng, Fei Wu, and Jiwei Li. 2020. Summarize, outline, and elaborate: Long-text generation via hierarchical supervision from extractive summaries. *CoRR*, abs/2010.07074.

Bowen Tan, Zichao Yang, Maruan Al-Shedivat, Eric P. Xing, and Zhiting Hu. 2021. Progressive generation of long text with pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4313–4324. Association for Computational Linguistics.

Wei Wang, Piji Li, and Hai-Tao Zheng. 2021. Consistency and coherency enhanced story generation. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part I*, volume 12656 of *Lecture Notes in Computer Science*, pages 694–709. Springer.

Feifei Xu, Xinpeng Wang, Yunpu Ma, Volker Tresp, Yuyi Wang, Shanlin Zhou, and Haizhou Du. 2020a. Controllable multi-character psychology-oriented story generation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 1675–1684. ACM.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020b. MEGATRON-CNTRL: controllable story generation with external knowledge using large-scale language models. In *Proceedings of the*

2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, pages 2831–2845. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7378–7385. AAAI Press.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

8 Appendix

8.1 Writing Mode Classifier

To study how many samples are enough to train the classifiers, we test the classifiers trained on the subsets of the FRIDE dataset. The subset size increases from 0 to 1000. The results in Figure 3 show the classifiers reach the best performance when the subset size is around 200.

When training the classifiers, we use the optuna [5] to do a hyper-parameter search. We run hyper parameter search 20 times and try to search the optimal value for learning rate (in range 1e-5∼5e-5), batch size (in range 2∼8), and training epochs (in range 2∼10). The optimal value for learning rate is 1e-5, for epoch is 10, and for batch size is 5. The reported test results are the average of 10 trails with different random seeds.

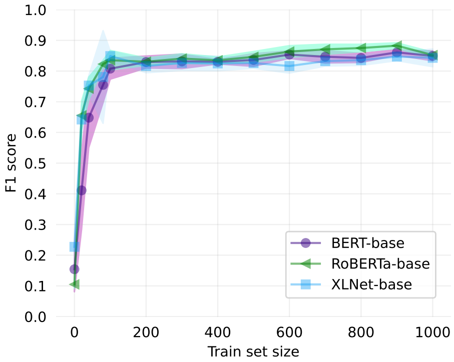


Figure 3: The F1 score of 3 classifiers trained on subtrain sets with different size.

8.2 Writing Mode Distribution

The writing mode is a kind of style to tell stories, and thus, it could be related to the genre of fictions. To verify that the writing modes are a style independent of the genres, we select 9 frequent genres and analyze the distribution of the writing modes within each genre. The results in Figure 4 show that the writing modes have similar distribution across different genres, demonstrating that the distribution of writing modes is irrelevant to the category of genre. Thus, the writing mode is a style independent of the genre.

8.3 Training Settings

The story generation models are finetuned from three types of pretrained language models such as BART, GPT2, and T5. We utilize the large version of the pretrained models such as “bart-large",“gpt2-large", and “t5-large", for better performance, and these models are downloaded through huggingface [6] model library. The story generation models are trained on 8 Tesla A100 with a learning rate 4e-5 and a batch size 32 for 5 epochs about 24 hours. The warm up steps for all models are the first 10% steps of the total steps. During inference phase, we adopt nuclear sampling method with top_p being 0.9 recommended by Holtzman et al. (2020).

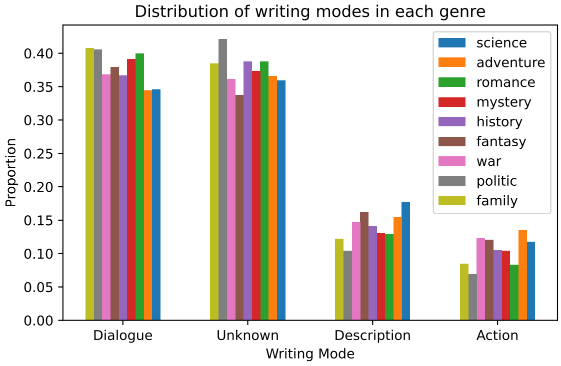


Figure 4: The writing mode distribution in each genre.

	Train	Valid	Test
#Paragraph	360K	1.44K	1.44K
#Dialogue	131.196K	0.493K	0.36K
#Description	54.005K	0.259K	0.36K
#Action	30.478K	0.103K	0.36K
#Uncertain	144.321K	0.585K	0.36K
#Token of Paragraph	111.3	111.4	111.0

Table 7: The statistics of FRIDE-XL Dataset.

	ROUGE-1	ROUGE-2	ROUGE-L
Paper	22.2	4.8	16.9
Our	21.5	4.4	16.5

Table 8: The evaluation for the performance of the summarization model.

8.4 Summarization Model

Most prior works focus on the summarization of news articles, which is a domain different from the narrative text in books. Recently, Kryscinski et al. 2021 collect a summarization dataset for the narrative text in books. Each book in the dataset is summarized in different levels including paragraph-level, chapter-level, and book-level. We run their

---

[5] https://optuna.readthedocs.io/en/stable/

[6] https://huggingface.co

codes[7] and train a paragraph-level summarization model. The quality of summaries is evaluated by ROUGE scores as shown in the Table 8. The Table shows that we basically reproduce their results in the paper. However, better summarizers for narrative text could greatly improve the output of our story generation models. In particular, while the summaries are often correct, the focus of what aspects of the narrative text is summarized is not always the most appropriate, and improving this is a clear direction for future work. As for the summary size, the trained model summaries the original paragraph (∼116 tokens) into sentences (∼26 tokens). The average compression rate is 18.2%.

---

[7] https://github.com/salesforce/booksum

Robustness Challenges in Model Distillation and Pruning for Natural Language Understanding

Mengnan Du[1]*, Subhabrata Mukherjee[2], Yu Cheng[2], Milad Shokouhi[2], Xia Hu[3], Ahmed Hassan Awadallah[2]
[1]New Jersey Institute of Technology [2]Microsoft Research [3]Rice University
mengnan.du@njit.edu, xia.hu@rice.edu
{submukhe,Yu.Cheng,milads,hassanam}@microsoft.com

Abstract

Recent work has focused on compressing pre-trained language models (PLMs) like BERT where the major focus has been to improve the in-distribution performance for downstream tasks. However, very few of these studies have analyzed the impact of compression on the generalizability and robustness of compressed models for out-of-distribution (OOD) data. Towards this end, we study two popular model compression techniques including knowledge distillation and pruning and show that the compressed models are significantly less robust than their PLM counterparts on OOD test sets although they obtain similar performance on in-distribution development sets for a task. Further analysis indicates that the compressed models overfit on the shortcut samples and generalize poorly on the hard ones. We further leverage this observation to develop a regularization strategy for robust model compression based on sample uncertainty. Experimental results on several natural language understanding tasks demonstrate that our bias mitigation framework improves the OOD generalization of the compressed models, while not sacrificing the in-distribution task performance.

1 Introduction

Large pretrained language models (PLMs) (e.g., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), GPT-3 (Brown et al., 2020)) have obtained state-of-the-art performance in several Natural Language Understanding (NLU) tasks. However, recent studies (Niven and Kao, 2019; Du et al., 2021; Mudrakarta et al., 2018) indicate that PLMs heavily rely on shortcut learning/spurious correlations, rather than acquiring higher level language understanding and semantic reasoning in several NLU tasks. Specifically, these models often exploit dataset biases and artifacts, e.g., lexical bias and overlap bias, as shortcuts for prediction. Due to the independent and identically distributed (IID) split of training, development, and test sets, these models that learn spurious decision rules from training data can perform well on in-distribution data (Du et al., 2022). Nevertheless, the shortcut learning behavior will result in models that have poor generalization performance on out-of-distribution (OOD) data, raising concerns about their robustness.

On the other hand, it is difficult to use these large PLMs models in real-world applications with latency and capacity constraints, e.g., on edge devices and mobile phones. Thus, model compression emerges as one of the techniques to reduce model size, speed up inference, and save energy without significant performance drop for downstream tasks. State-of-the-art model compression techniques such as knowledge distillation (Sanh et al., 2019; Sun et al., 2019) and pruning (Sanh et al., 2020) primarily focus on evaluating compressed model performance in in-distribution test data. However, in-distribution testing is insufficient to capture the generalizability of PLMs (D'Amour et al., 2020). In contrast to existing work that is geared towards general-purpose PLMs (Niven and Kao, 2019; Du et al., 2021; Mudrakarta et al., 2018), this work aims to study the impact of compression on the shortcut learning and OOD generalization ability of compressed models.

Towards this end, we conduct comprehensive experiments to evaluate the OOD robustness of compressed models, with BERT as the base encoder. **We focus primarily on two popular model compression techniques in the form of pruning and knowledge distillation** (Sanh et al., 2019; Wang et al., 2020). For pruning, we consider two popular techniques including iterative magnitude pruning (Sanh et al., 2020) and structured pruning (Prasanna et al., 2020; Liang et al., 2021). Specifically, we explore the following research questions: *Are distilled and pruned models as ro-*

---

*Most of the work was completed while the first author was an intern at Microsoft Research during summer 2021.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1766–1778
May 2-6, 2023 ©2023 Association for Computational Linguistics

bust as their PLM counterparts for downstream NLU tasks? What is the impact of varying the level of compression on OOD generalization and bias of compressed models? We evaluate the performance of several compressed models obtained using the above techniques on both standard in-distribution development sets and OOD test sets for downstream NLU tasks. Experimental analysis indicates that distilled and pruned models are consistently less robust than their PLM counterparts. Further analysis of the poor generalization performance of compressed models reveals some interesting observations. For instance, we observe that the compressed models overfit on the easy / shortcut samples and generalize poorly on the hard ones. This motivates our second research question: *How to regularize model compression techniques to generalize across samples with varying difficulty?* This brings some interesting challenges since we do not know which samples are easy or hard apriori.

Based on the above observations, we propose a bias mitigation framework to improve the OOD robustness of compressed models, termed as RMC (Robust Model Compression). First, we leverage the uncertainty of the deep neural network to quantify the difficulty of a training sample. This is given by the variance in the prediction of a sample from multiple sub-networks of the original large network obtained by model pruning. Second, we leverage this sample-specific measure for smoothing and regularizing different families of compression techniques. The major contributions of this work can be summarized as follows:

- We perform a comprehensive analysis to evaluate the OOD generalization ability and robustness of compressed models for NLU tasks.
- We further analyze plausible reasons for the low generalizability of compressed models and demonstrate connections to shortcut learning.
- We propose a mitigation framework for regularizing model compression, termed as RMC, which smoothes the knowledge distillation training based on the estimated sample difficulties.
- We perform experiments to demonstrate that our RMC framework improves OOD generalization while not sacrificing the standard in-distribution task performance on multiple NLU tasks.

2 Related Work

Shortcut Learning and Mitigation. Recent studies indicate that PLMs tend to exploit biases and artifacts in the dataset as shortcuts for prediction, rather than acquiring higher level semantic understanding and reasoning for NLU tasks (Niven and Kao, 2019; Du et al., 2021; McCoy et al., 2019a). There are some preliminary work to mitigate the bias of general PLMs, including product-of-experts (Clark et al., 2019; He et al., 2019; Sanh et al., 2021), re-weighting (Schuster et al., 2019; Yaghoobzadeh et al., 2019; Utama et al., 2020), adversarial training (Stacey et al., 2020), posterior regularization (Cheng et al., 2021), etc.

Robustness in Model Compression. Current practice for evaluating model compression performance focuses mainly on standard benchmark performance (Zhu et al., 2020; Wang et al., 2021). In the computer vision domain, previous work shows that compressed models perform poorly in Compression Identified Exemplars (CIE) (Hooker et al., 2019), and compression amplifies algorithmic bias towards certain demographics (Hooker et al., 2020). The most similar work to ours are two concurrent work (Xu et al., 2021a; Li et al., 2021) that investigate the performance of compressed models beyond standard benchmarks for natural language understanding tasks. However, both work mainly focus on evaluating the robustness of compressed models with respect to the scenario of adversarial attacks, i.e., TextFooler (Jin et al., 2020), and the unified adversarial framework (Li et al., 2021). In contrast, we comprehensively characterize the robustness of BERT compression in OOD test sets to probe the OOD generalizability of the compression techniques. Besides, we use insights from this robustness analysis to design a generalizable and robust model compression framework.

3 Are Compressed Models Robust?

We perform a comprehensive analysis to evaluate the robustness of compressed language models.

3.1 Compression Techniques

We consider two popular families of compression, namely, knowledge distillation and model pruning.

Knowledge Distillation: The objective here is to train a small-size model by mimicking the behavior of the larger teacher model using knowledge distillation (Hinton et al., 2015). In this work, we focus on task-agnostic distillation. In particular, we consider DistilBERT (Sanh et al., 2019) and MiniLM (Wang et al., 2020) distilled from BERT-base. For a fair comparison, we select compressed models with similar capacities ($66M$ parameters in this work). In order to evaluate the impact of compression techniques on model robustness, we also consider similar capacity smaller models without using knowledge distillation. These are obtained via simple truncation where we retain the first 6 layers of the large model, and via pre-training a smaller 6-layer model from scratch.

Iterative Magnitude Pruning: This is a task-specific unstructured pruning method (Sanh et al., 2020). During the fine-tuning process for each downstream task, the weights with the lowest magnitude are removed until the pruned model reaches the target sparsity. Note that we utilize the standard pruning technique, rather than the LTH-based pruning (lottery ticket hypothesis) that uses rewinding (Chen et al., 2020). We also consider different pruning ratios to obtain pruned models with different levels of sparsity.

Structured Pruning: This method family is based on the hypothesis that there is redundancy in the attention heads (Prasanna et al., 2020; Voita et al., 2019; Bian et al., 2021; Chen et al., 2021). We also consider task-specific pruning. During the fine-tuning process for each task, it prunes the whole attention heads based on their importance to the model predictions. Please refer to Sec. A in Appendix for more details. We prune around 20% attention heads in total (i.e., 28 attention heads). Further pruning increases the sparsity with significant degradation of the model's performance on in-distribution development sets.

3.2 Evaluation Datasets

To evaluate the robustness of the compressed models introduced in the last section, we use three NLU tasks, including MNLI, FEVER, and QQP[1]. Please refer to Sec. B in Appendix for more details.

- **MNLI** (Williams et al., 2018): This is a natural language inference task. In this work, we report the accuracy metric on the matched subset. We use HANS (McCoy et al., 2019b) as the adversarial test set, which contains $30,000$ synthetic samples. Models that exploit shortcut features have been shown to perform poorly on the HANS test set.
- **FEVER** (Thorne et al., 2018): This is a fact verification dataset. Recent studies indicate that there are strong shortcuts in the claims (Utama et al., 2020). To facilitate the robustness and generalization evaluation of fact verification models, two symmetric test sets (i.e., Sym v1 and Sym v2) were created, where bias exists in the symmetric pairs (Schuster et al., 2019). Both OOD test sets have 712 samples.
- **QQP**: The task is to predict whether a pair of questions is semantically equivalent. We consider the OOD test set PAWS-qqp, which contains 677 test samples generated from QQP corpus (Zhang et al., 2019; Yang et al., 2019). Besides, we also consider the PAWS-wiki OOD test set, which consists of $8,000$ test samples generated from Wikipedia pages.

For all three tasks, we employ accuracy as the evaluation metric and evaluate the performance of the compressed models on both the in-distribution development set and the OOD test set.

3.3 Evaluation Setup

In this work, we use the uncased BERT-base as the teacher network, and study the robustness of its compressed variants. The final model consists of the BERT-base encoder (or its compressed variants) with a classification head (a linear layer on top of the pooled output). Recent studies indicate that factors such as learning rate and training epochs could have a substantial influence on robustness (Tu et al., 2020). In particular, increasing training epochs can help improve the generalization of the OOD test set. In this work, we focus on the relative robustness of compressed models compared to the uncompressed teacher, rather than their *absolute accuracies*. For a fair comparison, we unify the experimental setup for all models. We use Adam optimizer with weight decay (Loshchilov and Hutter, 2017), where the learning rate is fixed as $2e$-5, and we train all models for 5 epochs on all datasets. We perform the experiments using PyTorch and use the pre-trained models from the Huggingface model pool (Wolf et al., 2019). We report the average results over three runs for all experiments.

3.4 Relative Robustness Metric

As we later demonstrate, with increase in compression ratio or model sparsity, the performance of the smaller models degrades for both in-domain and OOD test sets. To compare the gap between

---

[1] MNLI, FEVER, and QQP are the three most widely used datasets to evaluate the shortcut learning/bias behavior and OOD generalization of PLMs in the literature (Tu et al., 2020; He et al., 2019; Clark et al., 2019; Schuster et al., 2019)

		MNLI			FEVER				QQP				
Sparsity	#Param	DEV	HANS	$\mathcal{F}_{bias}$	DEV	Sym1	Sym2	$\mathcal{F}_{bias}$	DEV	paws$_{wiki}$	paws$_{qqp}$	$\mathcal{F}_{bias}$	Average $\mathcal{F}_{bias}$
BERT-base	109M	84.2	59.8	-	86.2	58.9	64.5	-	90.9	48.9	34.7	-	-
20%	87.2M	84.4	55.5	1.182	86.5	57.0	64.6	1.045	90.7	47.2	33.5	1.037	1.088
40%	65.4M	84.0	54.7	1.204	86.4	57.2	64.0	1.051	90.5	46.6	32.4	1.049	1.101
60%	43.6M	83.4	52.8	1.266	86.3	56.9	63.3	1.068	90.2	45.9	31.8	1.061	1.132
70%	32.7M	81.8	52.2	1.249	85.9	56.6	63.3	1.063	89.5	45.4	30.7	1.065	1.127

Table 1: Accuracy comparison (in percent) and relative bias $\mathcal{F}_{bias}$ (the smaller the better) for models with iterative magnitude pruning with different levels of sparsity. The last column indicates the average $\mathcal{F}_{bias}$ values over three tasks. *Pruned models have relatively higher degradation in OOD test set compared to the development set.*

		MNLI			FEVER				QQP				
Sparsity	#Param	DEV	HANS	$\mathcal{F}_{bias}$	DEV	Sym1	Sym2	$\mathcal{F}_{bias}$	DEV	paws$_{wiki}$	paws$_{qqp}$	$\mathcal{F}_{bias}$	Average $\mathcal{F}_{bias}$
BERT-base	109M	84.2	59.8	-	86.2	58.9	64.5	-	90.9	48.9	34.7	-	-
DistilBERT	66M	82.3	51.2	1.289	84.5	51.9	60.4	1.183	89.9	48.1	34.6	1.006	1.159
MiniLM	66M	83.1	51.4	1.309	84.2	53.4	60.7	1.137	89.9	46.8	31.0	1.039	1.162
Truncated-l6	66M	80.8	51.6	1.247	84.4	52.6	60.4	1.163	90.0	46.0	32.4	1.056	1.155
Pretrained-l6	66M	81.6	52.2	1.229	85.8	54.7	62.6	1.115	90.0	46.4	33.9	1.045	1.130

Table 2: Accuracy comparison (in percent) and relative bias $\mathcal{F}_{bias}$ (the smaller the better) of compressed models with knowledge distillation. *Distilled models have relatively higher degradation in OOD test set compared to the development set.* Except BERT-base, all other models have $66M$ parameters.

Models	Attemtion heads	DEV	HANS	$\mathcal{F}_{bias}$
BERT-base	144	84.2	59.8	-
BERT-116heads-v1	116	84.1	55.5	1.172
BERT-116heads-v2	116	84.2	53.7	1.250
BERT-116heads-v3	116	84.0	55.3	1.176

Table 3: Accuracy comparison (in percent) and relative bias $\mathcal{F}_{bias}$ (the smaller the better) of compressed models with structured pruning. *Pruned models have relatively higher degradation in OOD test set compared to the development set.* All compressed models have been pruned 28 attention heads.

in-distribution task performance and OOD generalizability, we define a new metric that measures this performance gap of the compressed models with respect to the uncompressed BERT-base (teacher). First, we calculate the accuracy gap between in-distribution development set and OOD test set as $\frac{F_{\text{dev}}-F_{\text{OOD}}}{F_{\text{dev}}}$ for BERT-base (denoted by $\Delta_{\text{BERT-base}}$); and its compressed variant (denoted by $\Delta_{\text{compressed}}$). Second, we compute the relative bias as the ratio between the accuracy gap of the compressed model with respect to BERT-base: $\mathcal{F}_{bias} = \frac{\Delta_{\text{compressed}}}{\Delta_{\text{BERT-base}}}$. Here $\mathcal{F}_{bias} > 1$ indicates that the compressed model is more biased than BERT-base with the degree of bias captured in a larger value of $\mathcal{F}_{bias}$. Since FEVER has two OOD test sets, we use the overall accuracy of sym1 and sym2 to calculate $\mathcal{F}_{bias}$. Similarly, the OOD accuracy for QQP is the overall accuracy on PAWS-wiki and PAWS-qqp.

3.5 Experimental Observations

We report the performance of accuracy and the relative bias measure $\mathcal{F}_{bias}$ for iterative magnitude pruning in Table 1, knowledge distillation in Table 2 and structured pruning in Table 3. We have the following key observations.

Iterative Magnitude Pruning: First, for slight and mid-level sparsity, the pruned models have comparable and sometimes even better performance on the in-distribution development set. Consider FEVER as an example, where the compressed model preserves the accuracy on the in-distribution set even at 60% sparsity[2]. However, the generalization accuracy on the OOD test set has a substantial drop. This indicates that the development set fails to capture the generalizability of the pruned models. Second, as the sparsity increases, the generalization accuracy on the OOD test set substantially decreases while dropping to random guess for tasks such as MNLI. Third, at high levels of sparsity (e.g. 70%), both development and OOD test set performances are significantly affected. In general, we observe $\mathcal{F}_{bias} > 1$ for all levels of sparsity in Table 1. Note that we limit the maximum sparsity at 70% after which the training is unstable with a significant performance drop even on the development set (Liang et al., 2021). As in the previous cases, there is substantial accuracy drop on the OOD test

---

[2] Here, 60% sparsity indicates that 40% parameters are remaining after pruning.

set compared to the development set (e.g., 7.6% vs 1.9% degradation respectively for the MNLI task).

Knowledge Distillation: Similar to pruning, we observe a higher accuracy drop in the OOD test set compared to the in-distribution development set for distilled models. Consider DistilBERT performance on MNLI as an example with 1.9% accuracy drop in development set compared to 8.6% drop in the OOD test set. This can also be validated in Table 2, where all $\mathcal{F}_{bias}$ values are larger than 1, depicting that all the distilled models are less robust than BERT-base. Another interesting observation is that distilled models, i.e., DistilBERT and MiniLM, have higher bias $\mathcal{F}_{bias}$ compared to the pre-trained models, i.e., Pretrained-l6 and Truncated-l6, as we compare their average $\mathcal{F}_{bias}$ values in Table 2. This indicates that the compression process plays a significant role in the low generalizability and robustness of the distilled models.

Structured Pruning: Recent studies have reported the super ticket phenomenon (Liang et al., 2021). The authors observe that, when the BERT-base model is slightly pruned, the accuracy of the pruned models improves on in-distribution development set. However, we observe that this finding does not hold for OOD test sets. From Table 3, we observe that all pruned models are less robust than BERT-base, with $\mathcal{F}_{bias}$ much larger than 1.

4 Attribution of Low Robustness

In this section, we explore the factors that lead to low robustness of compressed models. Previous work has demonstrated that the performance of different models on the GLUE benchmark (Wang et al., 2018) tends to correlate with the performance on MNLI, making it a good representative of natural language understanding tasks in general (Phang et al., 2018; Liu et al., 2020). For this reason, we choose the MNLI task for a study.

For the MNLI task, we consider the dataset splits from (Gururangan et al., 2018). The authors partition the development set into easy/shortcut [3] and hard subsets. In this experiment, we use pruned models with varying sparsity to investigate the reason for the low robustness of the compressed models. We have the following key observations.

Observation 1: The compressed models tend to overfit the easy/shortcut samples and generalize poorly on the hard ones. The performance of pruned models at five sparsity levels (ranging between $[0.2 - 0.85]$) on the easy and hard samples for the MNLI task is illustrated in Figure 1. It demonstrates that the accuracy on the hard samples is much lower compared to the accuracy on the easy ones. As the sparsity increases, we observe a larger accuracy drop on the hard samples compared to the easy ones. In particular, the accuracy gap between the two subsets is 22.7% at the sparsity of 0.85, much higher than the 16.1% accuracy gap at the sparsity of 0.4. These findings demonstrate that the compressed models overfit on the easy samples, while generalizing poorly on the hard ones. Furthermore, this phenomenon is amplified at higher levels of sparsity for the pruned models.

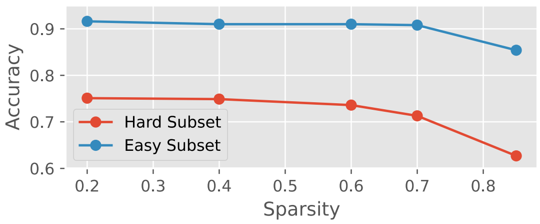


Figure 1: Pruned model performance on hard vs easy / shortcut samples with varying sparsity, where x-axis denotes the sparsity level.

Observation 2: Compressed models tend to assign overconfident predictions to easy samples. One of the potential reasons is that compressed models are more prone to capture spurious correlations between shortcut features in training samples with certain class labels for their predictions (Geirhos et al., 2020; Du et al., 2021).

4.1 Variance-based Difficulty Estimation

Based on the above observations, we propose a variance-based metric to quantify the difficulty degree of each sample. For each sample in the development set, we calculate its loss at five different levels of pruning sparsity as shown in Figure 1. We further calculate the variance of the above losses for each sample and rank them based on the variance. Finally, we assign the samples with low variance to the “easy" subset and rest to the “hard" one. Comparing our variance-based proxy annotation with the ground truth annotation in (Gururangan et al., 2018) gives an accuracy of 82.8%. This indicates that the variance-based estimation leveraging pruning sparsity is a good indicator of sample difficulty. This motivates our design of the mitigation technique introduced in the next section.

---

[3] We use ‘easy’ and ‘shortcut’ interchangeably in this work.

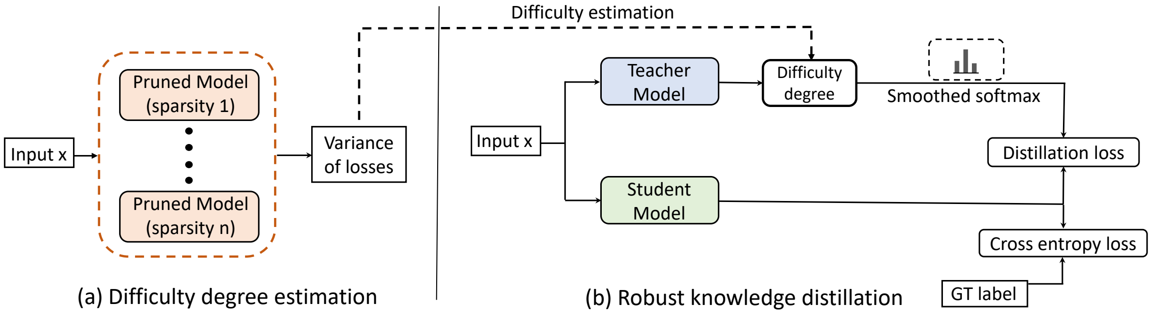


Figure 2: RMC framework for bias mitigation with two-stage training. In the first stage, we feed the training samples to pruned models at different levels of sparsity (ranging from $[0.2 - 0.85]$) as introduced in Section 4.1); compute corresponding losses and their variance to estimate the difficulty degree of each training sample. In the second stage, we use the difficulty degree to regularize the teacher network for robust model compression.

5 Mitigation Framework

In this section, we propose a general bias mitigation framework (see Figure 2), termed as RMC (Robust Model Compression), to improve the robustness of compressed models on downstream tasks. Our RMC framework follows the philosophy of task-specific knowledge distillation (Sanh et al., 2020; Jiao et al., 2020), but with explicit regularization of the teacher network leveraging sample uncertainty. This prevents the compressed model from overfitting in the easy samples that contain shortcut features and helps improve its robustness. This regularized training is implemented in two stages.

5.1 Quantifying Sample Difficulty

In the first stage, our objective is to quantify the difficulty degree of each training sample.

Variance Computation: Following the observations obtained in Section 4.1, we first use iterative magnitude pruning to obtain a series of pruned models from BERT-base with different levels of sparsity and then we use the losses of the pruned models at different levels of sparsity to compute their variance v_i for each training sample x_i: $v_i = \frac{\sum_{t=1}^{n}(l_{i,t}-\bar{l_i})^2}{n}$. We choose five sparsity levels, i.e., $n = 5$, that are diverse enough to reflect the difficulty degree of each training sample. Here, samples with high variance correspond to hard ones.

Difficulty Degree Estimation: Based on the variance v_i for each training sample x_i, we can estimate its difficulty degree as:

$$d_i = \alpha + \frac{1-\alpha}{V_{\max} - V_{\min}} \cdot (v_i - V_{\min}) \,, \qquad (1)$$

where $V_{\min}$ and $V_{\max}$ denote the minimum and maximum values of the variances, respectively. Equation 1 is used to normalize the variance of the training samples in the range of [α, 1], where $d_i = 1$ denotes the most difficult training sample, according to our criteria of loss variance. Samples with d_i closer to α are treated as shortcut/biased samples. Prior work (Niven and Kao, 2019) show that the bias behavior of the downstream training set can be attributed to data collection and annotation biases. Since the bias level is different for each dataset, we assign a different α in Equation 1 to each training set to reflect its bias level.

5.2 Robust Knowledge Distillation

In the second stage, we fine-tune BERT-base on the downstream tasks to obtain the softmax probability for each training sample. We then use the difficulty degree of the training samples (discussed in the previous section) to smooth the teacher predictions. The instance-level smoothed softmax probability is used to guide the training of compressed models through regularized knowledge distillation.

Smoothing Teacher Predictions: We smooth the softmax probability $\hat{y}_i^T$ from the teacher network, according to the difficulty degree d_i of each training sample x_i. The smoothed probability is given as:

$$s_{i,j} = \frac{(\hat{y}_i^T)_j^{d_i}}{\sum_{k=1}^{K}(\hat{y}_i^T)_k^{d_i}}, \qquad (2)$$

where K denotes the total number of class labels. We perform instance-level smoothing for each training sample x_i. If the difficulty degree of a training sample $d_i = 1$, then the softmax probability s_i for the corresponding sample from the teacher is unchanged. In contrast, at the other extreme as $d_i \to \alpha$, we increase the regularization to encourage the compressed model to assign less overconfident predictions to the sample. The difficulty degree range is $[\alpha, 1]$ rather than $[0, 1]$ to avoid over-smoothing of the teacher predictions.

		MNLI			FEVER				QQP				
Sparsity	#Param	DEV	HANS	$\mathcal{F}_{bias}$	DEV	Sym1	Sym2	$\mathcal{F}_{bias}$	DEV	paws$_{wiki}$	paws$_{qqp}$	$\mathcal{F}_{bias}$	Average $\mathcal{F}_{bias}$
BERT-base	110M	84.2	59.8	-	86.2	58.9	64.5	-	90.9	48.9	34.7	-	-
40% – Vanilla	65.4M	84.0	54.7	1.204	**86.4**	57.2	64.0	1.051	90.5	46.6	32.4	1.049	1.101
– Distil	65.4M	84.1	56.2	1.145	86.3	58.4	64.5	1.013	90.5	47.3	33.2	1.032	1.063
– Smooth	65.4M	**84.2**	56.5	1.135	86.2	60.7	65.8	0.937	**90.7**	47.2	33.8	1.036	1.036
– Focal	65.4M	84.0	56.7	1.122	**86.4**	59.4	65.2	0.981	**90.7**	46.2	32.1	1.060	1.054
– JTT	65.4M	83.8	56.3	1.132	86.2	58.1	64.9	1.008	90.4	47.3	33.7	1.030	1.057
– RMC	65.4M	**84.2**	**58.6**	**1.049**	86.1	**61.9**	**66.4**	**0.897**	90.4	**47.6**	**34.3**	**1.023**	**0.990**

Table 4: Generalization accuracy comparison (in percent) and the corresponding $\mathcal{F}_{bias}$ values for iterative magnitude pruning at 40% sparsity with different *mitigation methods*. The last column indicates average $\mathcal{F}_{bias}$ over three tasks.

Smoothness-Induced Model Compression: We employ the smoothed softmax probability s_i from BERT-base to supervise the training of the compressed models, where the overall loss function is:

$$\mathcal{L}(x) = (1-\lambda) * \mathcal{L}_1\left(y_i, \hat{y}_i^S\right) + \lambda * \mathcal{L}_2\left(s_i, \hat{y}_i^S\right), \quad (3)$$

where y_i is the ground truth and $\hat{y}_i^S$ is the probability of the compressed model. $\mathcal{L}_1$ denotes the cross-entropy loss, and $\mathcal{L}_2$ represents the knowledge distillation loss with KL divergence. Hyperparameter λ manages the trade-off between learning from hard label y_i and softened softmax probability s_i. Among the different families of compression techniques introduced in Section 3.1, we directly fine-tune the distilled models using Equation 3. For iterative magnitude pruning, we use Equation 3 to guide the pruning during the fine-tuning process.

6 Mitigation Performance Evaluation

In this section, we conduct experiments to evaluate the robustness of our RMC mitigation framework.

6.1 Experimental Setup

For all experiments, we follow the same setting as in Section 3.3, and the same evaluation datasets as in Section 3.2. We use the OOD test set exclusively for evaluation. We compute the variance of samples (outlined in Section 4.1) in the in-distribution development set to split it into a shortcut and hard subset. The relative robustness between the hard and easy subset is used to tune the hyperparameter α in Equation 1, where we set α as 0.5, 0.3, 0.2 for MNLI, FEVER, and QQP, respectively. The weight λ in Equation 3 is fixed as 0.9 for all experiments.

6.2 Baseline Methods

We consider the following five baselines. Please refer to Sec. C in Appendix for more details.

- **Vanilla**: This only fine-tunes the base encoder without any regularization.
- **Distil** (Task-Specific Knowledge Distillation) (Sanh et al., 2020): This first fine-tunes BERT-base on the downstream NLU tasks. The softmax probability from the fine-tuned BERT-base is used as the supervision signal for distillation.
- **Smooth** (Global Smoothing) (Müller et al., 2019): This performs global smoothing for *all* training samples with task-specific knowledge distillation, where we use the same level of regularization as in RMC ($d_i = 0.9$ in Equation 2). In contrast, RMC uses instance-level smoothing.
- **Focal** (Focal Loss) (Lin et al., 2017): Compared to cross-entropy loss, focal loss has an additional regularizer to reduce the weight for easy samples and assign a higher weight to hard samples bearing less-confident predictions.
- **JTT** (Just Train Twice) (Liu et al., 2021): This is a re-weighting method, which first trains the BERT-base model using standard cross-entropy loss for several epochs, and then trains the compressed model while up-weighting the training examples that are misclassified by the first model, i.e., hard samples.

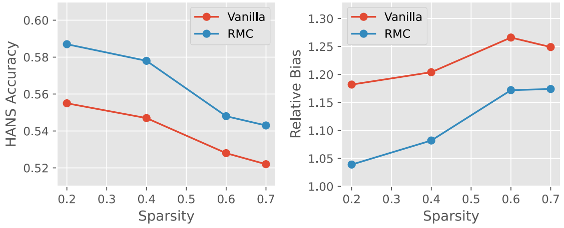
6.3 Mitigation Performance Analysis

We compare our RMC framework with the above baselines and have the following key observations.

Iterative Magnitude Pruning: Table 4 shows the mitigation results of accuracy and relative bias $\mathcal{F}_{bias}$. All mitigation methods are performed with pruned models at 40% sparsity. We observe that task-specific knowledge distillation only slightly improves accuracy on the OOD test set compared to Vanilla tuning, since the teacher model itself is not robust for downstream tasks (Niven and Kao, 2019). Global smoothing further improves generalization accuracy compared to prior methods. Our

		MNLI			FEVER				QQP				
Sparsity	#Param	DEV	HANS	$\mathcal{F}_{bias}$	DEV	Sym1	Sym2	$\mathcal{F}_{bias}$	DEV	paws_{wiki}	paws_{qqp}	$\mathcal{F}_{bias}$	Average $\mathcal{F}_{bias}$
BERT-base	110M	84.2	59.8	-	86.2	58.9	64.5	-	90.9	48.9	34.7	-	-
MiniLM – Vanilla	66M	83.1	51.4	1.309	84.2	53.4	60.7	1.137	89.9	46.8	31.0	1.039	1.162
– Distil	66M	83.1	53.7	1.221	83.8	56.5	61.0	1.052	89.6	46.7	31.8	1.037	1.103
– Smooth	66M	82.7	53.8	1.206	83.7	56.9	62.1	**1.017**	89.4	46.8	32.2	**1.032**	1.085
– Focal	66M	83.2	55.6	1.145	83.8	54.7	61.4	1.081	90.3	46.8	33.2	1.041	1.089
– JTT	66M	82.8	55.7	1.129	83.5	53.8	61.7	1.085	90.1	**47.0**	32.9	1.034	1.083
– RMC	66M	**83.7**	**57.8**	**1.068**	**85.3**	**58.0**	**63.3**	**1.017**	**90.5**	**47.0**	**33.4**	1.038	**1.041**

Table 5: Generalization accuracy and the $\mathcal{F}_{bias}$ values comparison of different training strategies with and without mitigation on in-distribution development set and OOD test set using MiniLM as the compressed encoder.



(a) Accuracy on OOD test set (b) Relative bias F_{bias}

Figure 3: RMC mitigation performance for iterative magnitude pruning at different levels of pruning sparsity for MNLI task.

RMC framework obtains the best accuracy on OOD test set across all the tasks on aggregate. RMC further reduces the average relative bias $\mathcal{F}_{bias}$ by 10% over Vanilla tuning, as shown in Table 4, indicating the benefits of uncertainty-based sample-wise smoothing in terms of improving model robustness. For the MNLI task, we also illustrate the mitigation performance of our RMC framework for different levels of sparsity in Figure 3. We observe that RMC consistently improves accuracy on OOD HANS while reducing the relative bias $\mathcal{F}_{bias}$ for all levels of sparsity over the Vanilla method.

Knowledge Distillation: Table 5 shows the mitigation results of accuracy and relative bias $\mathcal{F}_{bias}$. We observe that RMC significantly improves over MiniLM for OOD generalization leveraging smoothed predictions from BERT-base teacher. With instance-level smoothing in RMC, the generalization accuracy for the compressed model on the OOD test set is significantly closer to BERT-base teacher compared to the other methods. We also decrease the relative bias $\mathcal{F}_{bias}$ in Table 5 by 10.4% over Vanilla tuning. On the QQP task, RMC simultaneously improves the performance of compressed model on both the in-distribution development set and the two OOD test sets.

Models	DEV	HANS	Hard (H)	Easy (E)	Gap (E-H)
MiniLM–Vanilla	83.1	51.4	73.2	90.9	17.7
MiniLM–RMC	83.7	57.8	74.9	90.6	15.7
40%–Vanilla	84.0	54.7	74.9	91.0	16.1
40%–RMC	84.2	58.6	75.9	90.3	14.4

Table 6: Our RMC framework improves accuracy of the compressed models on the hard samples and reduces overfitting on the shortcut/easy samples, leading to reduced performance gap between the two subsets.

6.4 Further Analysis on Robust Mitigation

In this section, we further investigate the reasons for the improved generalization performance with RMC with an analysis on the MNLI task. Table 6 shows the accuracy performance of RMC for model pruning and distillation on the shortcut/easy and hard samples. We observe RMC to improve the model performance on the under-represented hard samples, where it reduces the generalization gap between the hard and shortcut/easy subset by 10.6% at 0.4 level of sparsity and by 11.3% for knowledge distillation. This analysis demonstrates that RMC reduces the overfitting of the compressed models on the easy samples and encourages them to learn more from the hard ones, thus improving the generalization on the OOD test sets.

7 Conclusions

In this work, we conduct a comprehensive study of the robustness challenges in compressing large PLMs when fine-tuning in downstream NLU datasets. Furthermore, we propose a general mitigation framework with instance-level smoothing for robust model compression. Experimental analysis demonstrates our framework to improve the generalization and OOD robustness of compressed models for different compression techniques, while not sacrificing the in-distribution performance.

Limitations

First, we study the shortcut learning/bias problem and OOD generalization of model compression techniques, exclusively focusing on the two most widely used families of compression techniques, including knowledge distillation and pruning. Our empirical analysis indicates that these two families of compression techniques suffer from the low generalization issue. However, other types of compression technique, such as matrix decomposition and quantization, are not discussed in this work. Studying the whole compression techniques is a challenging topic and will be investigated in our future research. Second, our RMC framework needs to calculate the variance of losses for each training sample, thus requiring additional training time. Training efficiency can be further improved by implementing parallel training or more efficient ways of calculating sample difficulty, which will also be studied in our future research.

References

Yuchen Bian, Jiaji Huang, Xingyu Cai, Jiahong Yuan, and Kenneth Church. 2021. On attention redundancy: A comprehensive study. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*.

Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. 2021. Chasing sparsity in vision transformers:an end-to-end exploration.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pretrained bert networks. *34th Conference on Neural Information Processing Systems (NeurIPS)*.

Hao Cheng, Xiaodong Liu, Lis Pereira, Yaoliang Yu, and Jianfeng Gao. 2021. Posterior differential regularization with f-divergence for improving model robustness. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. *Empirical Methods in Natural Language Processing (EMNLP)*.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. 2020. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Mengnan Du, Fengxiang He, Na Zou, Dacheng Tao, and Xia Hu. 2022. Shortcut learning of large language models in natural language understanding: A survey. *arXiv preprint arXiv:2208.11857*.

Mengnan Du, Varun Manjunatha, Rajiv Jain, Ruchi Deshpande, Franck Dernoncourt, Jiuxiang Gu, Tong Sun, and Xia Hu. 2021. Towards interpreting and mitigating shortcut learning behavior of nlu models. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. *2019 EMNLP workshop*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *NeurIPS Deep Learning Workshop*.

Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome. 2019. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*.

Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. 2020. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058*.

Shaoyi Huang, Dongkuan Xu, Ian EH Yen, Sung-en Chang, Bingbing Li, Shiyang Chen, Mimi Xie, Hang Liu, and Caiwen Ding. 2021. Sparse progressive distillation: Resolving overfitting under pretrain-and-finetune paradigm. *arXiv preprint arXiv:2110.08190*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. *Findings of EMNLP*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? natural language attack on text classification and entailment. *AAAI Conference on Artificial Intelligence (AAAI)*.

Tianda Li, Ahmad Rashid, Aref Jafari, Pranav Sharma, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021. How to select one among all? an empirical study towards the robustness of knowledge distillation in natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 750–762.

Chen Liang, Simiao Zuo, Minshuo Chen, Haoming Jiang, Xiaodong Liu, Pengcheng He, Tuo Zhao, and Weizhu Chen. 2021. Super tickets in pre-trained language models: From model compression to improving generalization. *59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision (ICCV)*.

Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning (ICML)*.

Xiaodong Liu, Yu Wang, Jianshu Ji, Hao Cheng, Xueyun Zhu, Emmanuel Awa, Pengcheng He, Weizhu Chen, Hoifung Poon, Guihong Cao, et al. 2020. The microsoft toolkit of multi-task deep neural networks for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 118–126.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

R Thomas McCoy, Junghyun Min, and Tal Linzen. 2019a. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. *arXiv preprint arXiv:1911.02969*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019b. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *33rd Conference on Neural Information Processing Systems (NeurIPS)*.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? *56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems (NeurIPS)*.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. *57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv e-prints*, pages arXiv–1811.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When bert plays the lottery, all tickets are winning. *Empirical Methods in Natural Language Processing (EMNLP)*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *NeurIPS Workshop*.

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. 2021. Learning from others' mistakes: Avoiding dataset biases without modeling them. *International Conference on Learning Representations (ICLR)*.

Victor Sanh, Thomas Wolf, and Alexander M Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *34th Conference on Neural Information Processing Systems (NeurIPS)*.

Tal Schuster, Darsh J Shah, Yun Jie Serene Yeo, Daniel Filizzola, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. *Empirical Methods in Natural Language Processing (EMNLP)*.

Joe Stacey, Pasquale Minervini, Haim Dubossarsky, Sebastian Riedel, and Tim Rocktäschel. 2020. Avoiding the hypothesis-only bias in natural language inference via ensemble adversarial training. *Empirical Methods in Natural Language Processing (EMNLP)*.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics (TACL)*.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Towards debiasing nlu models from unknown biases. *Empirical Methods in Natural Language Processing (EMNLP)*.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2021. Infobert: Improving robustness of language models from an information theoretic perspective. In *International Conference on Learning Representations*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *34th Conference on Neural Information Processing Systems (NeurIPS)*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Ke Xu, Julian McAuley, and Furu Wei. 2021a. Beyond preserved accuracy: Evaluating loyalty and robustness of bert compression. *Empirical Methods in Natural Language Processing (EMNLP)*.

Dongkuan Xu, Ian EH Yen, Jinxi Zhao, and Zhibin Xiao. 2021b. Rethinking network pruning–under the pre-train and fine-tune paradigm. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Yadollah Yaghoobzadeh, Remi Tachet, Timothy J Hazen, and Alessandro Sordoni. 2019. Robust natural language inference models with example forgetting. *arXiv e-prints*, pages arXiv–1911.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*.

A More Details of Pruning Methods

In this section, we introduce more details about the compression techniques studied.

knowledge Distillation: For a fair comparison, we do not compare with TinyBERT (Jiao et al., 2020) and MobileBERT (Sun et al., 2020), since TinyBERT is fine-tuned with data augmentation on NLU tasks, and MobileBERT is distilled from BERT-large rather than BERT-base.

Magnitude Pruning: It is based on the overparameterization assumption of pre-trained language models (Xu et al., 2021b; Huang et al., 2021). For iterative magnitude pruning, we freeze all the embedding modules and only prune the parameters in the encoder (i.e., 12 layers of Transformer blocks). After pruning, the pruned weight values are set to 0 to reduce the amount of information to store. Unlike the LTH version, we consider standard magnitude pruning without using rewinding.

Structured Pruning: To calculate the importance, we follow (Michel et al., 2019; Prasanna et al., 2020) and calculate the expected sensitivity of the attention heads to the mask variable $\xi^{(h,l)}$: $I_h^{(h,l)} = E_{x \sim X} \left| \frac{\partial \mathcal{L}(x)}{\partial \xi^{(h,l)}} \right|$, where $I_h^{(h,l)}$ denotes the contribution score of the attention head h in layer l, $\mathcal{L}(x)$ represents the loss value for the sample x, and $\xi^{(h,l)}$ is the mask of the attention head h in layer l. After obtaining the contribution scores, the attention heads with lowest score $I_h^{(h,l)}$ are pruned.

B More on Evaluation Datasets

In this section, we introduce more details about the three benchmark datasets.

MNLI: This task aims to predict whether the relationship between the premise and the hypothesis is contradiction, entailment, or neutral. It is divided into a training set and development set with $392,702$ and $9,815$ samples, respectively.

FEVER: The task is to predict whether the claims support, refute, or not-have-enough-information about the evidence. Recent studies indicate that there are strong shortcuts in claims (Utama et al., 2020). It is divided into a training set and a development set with $242,911$ and $16,664$ samples, respectively.

QQP: It is divided into a training set and a development set with $363,846$ and $40,430$ samples, respectively.

C More on Comparing Baselines

In this section, we introduce more details on comparing baselines.

Distil and Smooth: For both baseline methods, we use a loss function similar to that of Equation 3. We fix the weight λ to 0.9 for all experiments, to encourage the compressed model to learn more from the probability output of the teacher network. A major difference between the two baselines is that Smooth has an additional smoothing process involved during the fine-tuning process.

Focal Loss: The original focal loss function is: $\mathrm{FL}(p_\mathrm{i}) = -(1-p_\mathrm{i})^\gamma \log(p_\mathrm{i})$. Our implementation is as follows:

$$\mathrm{FL}(p_\mathrm{i}) = -\frac{(1-p_\mathrm{i})^\gamma}{\frac{1}{N}\sum_{k=1}^{N}(1-p_\mathrm{k})^\gamma} \log(p_\mathrm{i}).$$

The hyperparameter γ controls the weight difference between hard and easy samples, and is fixed at 2.0 for all tasks. We use the denominator to normalize the weights within a batch, where N is the batch size. This is used to guarantee that the average weight for a batch of training samples is 1.0. As such, the weight for the easy samples would be down-weighted to lower than 1.0, and the weight for hard samples would be up-weighted to values larger than 1.0.

JTT: This is also a reweighting baseline that encourages the model to learn more from hard samples. The hyperparameter λ_{up} in (Liu et al., 2021) is set to 2.0. We also normalize the weights so that the average weight for each training sample is 1.0.

D Running Environment

For a fair evaluation of the robustness of compressed models, we run all experiments using a server with 4 NVIDIA GeForce 3090 GPUs. All experiments are implemented with the Pytorch version of the Hugging Face Transformer library.

E The Capacity Issue

One natural speculation about the low robustness of compressed models is due to their low capacity (i.e., smaller size). To disentangle the two important factors that influence model performance, i.e., low capacity and compression, we compare distilled models with Uncased-l6, which is trained only using pretraining. The results are given in Table 2. The results indicate that Uncased-l6 has better generalization ability over the MNLI and

FEVER two tasks. Take structured pruning as an example; although the three pruned models in Table3 have the same model size, their generalization accuracy is different. These results indicate that the low robustness of compressed models is not entirely due to their low capacity, and compression plays a significant role.

F MNLI Easy and Hard Subsets

The authors train a hypothesis-only model and use it to generate predictions for the whole development set (Gururangan et al., 2018). Samples that are given correct predictions by the hypothesis-only model are regarded as easy samples, and vice versa. The easy subset contains 5488 samples, and the hard subset contains 4302 samples.

Don't Blame the Annotator:
Bias Already Starts in the Annotation Instructions

Mihir Parmar[1*] **Swaroop Mishra**[1*] **Mor Geva**[2†] **Chitta Baral**[1]
[1]Arizona State University [2]Allen Institute for AI
{mparmar3, srmishr1, chitta}@asu.edu, pipek@google.com

Abstract

In recent years, progress in NLU has been driven by benchmarks. These benchmarks are typically collected by crowdsourcing, where annotators write examples based on annotation instructions crafted by dataset creators. In this work, we hypothesize that annotators pick up on patterns in the crowdsourcing instructions, which bias them to write many similar examples that are then over-represented in the collected data. We study this form of bias, termed *instruction bias*, in 14 recent NLU benchmarks, showing that instruction examples often exhibit concrete patterns, which are propagated by crowdworkers to the collected data. This extends previous work (Geva et al., 2019) and raises a new concern of whether we are modeling the *dataset creator's instructions*, rather than the task. Through a series of experiments, we show that, indeed, instruction bias can lead to overestimation of model performance, and that models struggle to generalize beyond biases originating in the crowdsourcing instructions. We further analyze the influence of instruction bias in terms of pattern frequency and model size, and derive concrete recommendations for creating future NLU benchmarks.[1]

1 Introduction

Benchmarks have been proven pivotal for driving progress in Natural Language Understanding (NLU) in recent years (Rogers et al., 2021; Bach et al., 2022; Wang et al., 2022). Nowadays, NLU benchmarks are mostly created through crowdsourcing, where crowdworkers write examples following annotation instructions crafted by dataset creators (Callison-Burch and Dredze, 2010; Zheng et al., 2018; Suhr et al., 2021). The instructions typically include a short description of the task, along with several examples (Dasigi et al., 2019; Zhou et al., 2019; Sakaguchi et al., 2020).

Despite the vast success of this method, past studies have shown that data collected through crowdsourcing often exhibit various biases that lead to overestimation of model performance (Schwartz et al., 2017; Gururangan et al., 2018; Poliak et al., 2018; Tsuchiya, 2018; Le Bras et al., 2020; Mishra et al., 2020a; Mishra and Arunkumar, 2021; Hettiachchi et al., 2021). Such biases are often attributed to annotator-related biases, such as writing style and background knowledge (Gururangan et al., 2018; Geva et al., 2019) (see more discussion on related work in §A).

In this work, we propose that biases in crowdsourced NLU benchmarks often originate at an early stage in the data collection process of designing the annotation task. In particular, we hypothesize that task instructions provided by dataset creators, which serve as the guiding principles for annotators to complete the task, often influence crowdworkers to follow specific patterns, which are then propagated to the dataset and subsequently over-represented in the collected data. For instance, $\sim 36\%$ of the instruction examples for the QUOREF dataset (Dasigi et al., 2019) start with *"What is the name"*, and this same pattern can be observed in $\sim 59\%$ of the collected instances.

To test our hypothesis, we conduct a broad study of this form of bias, termed *instruction bias*, in 14 recent NLU benchmarks. We find that instruction bias is evident in most of these datasets, showing that $\sim 73\%$ of instruction examples on average share a few clear patterns. Moreover, we find that these patterns are propagated by annotators to the collected data, covering $\sim 61\%$ of the instances on average. This suggests that instruction examples play a critical role in the data collection process and the resulting example distribution.

It is difficult to represent a task with a few examples, and bias in instruction examples makes

---

[*] Equal Contribution

[†] Now at Google Research

[1] Code and data is available at `https://github.com/Mihir3009/instruction-bias`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1779–1789
May 2-6, 2023

it even more difficult since a task and its associated reasoning have a larger scope than instruction patterns. For example co-reference resolution, temporal commonsense reasoning, and numerical reasoning are much broader tasks than the prevalent patterns in QUOREF (*"what is the name..."*), MC-TACO (*"how long..."*) and DROP (*"how many field goals..."*) datasets.

We investigate the effect of instruction bias on model performance, showing that performance is overestimated by instruction bias and that models often fail to generalize beyond instruction patterns. Moreover, we observe that a higher frequency of instruction patterns in the training set often increases the model performance gap on pattern and non-pattern examples and that large models are generally less sensitive to instruction bias.

In conclusion, our work shows that instruction bias widely exists in NLU benchmarks, often leading to an overestimation of model performance. Based on our study, we derive concrete recommendations for monitoring and alleviating this bias in future data collection efforts. From a broader perspective, our findings also have implications on the recent learning-by-instructions paradigm (Efrat and Levy, 2020; Mishra et al., 2021), where crowdsourcing instructions are used in model training.

2 Instruction Bias in NLU Benchmarks

Instructions are the primary resource for educating crowdworkers on how to perform their task (Nangia et al., 2021). Bias in the instructions, dubbed *instruction bias*, could lead crowdworkers to propagate specific patterns to the collected data.

Here, we study instruction bias in NLU benchmarks[2], focusing on two research questions: (a) Do crowdsourcing instructions exhibit patterns that annotators can pick up on? and (b) Are such patterns propagated by crowdworkers to the collected data? In our study, we use the instructions of 14 recent NLU benchmarks:[3] (1) CLARIQ (Aliannejadi et al., 2020), (2) COSMOSQA (Huang et al., 2019), (3) DROP (Dua et al., 2019), (4) DUORC (Saha et al., 2018), and (5) HOTPOTQA (Yang et al., 2018) (6) HYBRIDQA (Chen et al., 2020), (7) MC-TACO (Zhou et al., 2019), (8) MULTIRC (Khashabi et al., 2018), (9) PIQA (Bisk et al., 2020), (10) QASC (Khot et al., 2020), (11) QUOREF (Dasigi et al., 2019), (12) ROPES (Lin et al., 2019), (13) SCIQA (Welbl et al., 2017), (14) WINOGRANDE (Sakaguchi et al., 2020). These benchmarks were created through different crowdsourcing protocols to evaluate diverse tasks (Mishra et al., 2021) (see dataset statistics in §B).

Dataset	**Pattern**	**% Ins.**	**% $\mathcal{S}_{\text{train}}$**	**% $\mathcal{S}_{\text{test}}$**
CLARIQ	`[Are|Would|Do] you`	72.2	85.1	89
COSMOSQA	`What AUX`	87.5	45.1	38.4
DROP	`How many [field goals | years | yards | points | touchdowns]`	70	62.5	62.5
DUORC	`[How old | How | What | Who] AUX`	70	85.1	84
HOTPOTQA	`[In | Of | From | _ ] [Which|What] AUX`	87.5	53.8	54.2
HYBRIDQA	`Which AUX`	29.4	25.7	15.1
MC-TACO	`How long AUX`	100	-	87.6
	`What AUX`	100	-	90.1
	`How often AUX`	100	-	85.3
	`AUX... [still|always |by the time]`	100	-	67.3
	`When did / What time`	100	-	83.4
MULTIRC	`What AUX`	14.3	38.4	41.5
PIQA	`How [do|can]`	66.7	43.7	42.9
QASC	`What AUX`	57.1	49.3	47
QUOREF	`What AUX the [_| full|real|first |last] name`	36.4	57	60
ROPES	`Which AUX`	42.9	74.1	20.7
SCIQA	`What AUX`	100	83.6	84.5
WINOGRANDE	`[because | so | while | since | but] ... the`	73.7	63.4	63.1
Average		72.7	59	62

Table 1: Portion of patterns in instruction examples (Ins.) and in the corresponding train ($\mathcal{S}_{\text{train}}$) and test ($\mathcal{S}_{\text{test}}$) sets of NLU datasets. `AUX` $\in$ {am, is, are, was, were, has, have, had, do, does, did, will, would, can, could, may, might, shall, should, must}, and _ is an empty string. MC-TACO has 5 different data subsets corresponding to different types of temporal reasoning (see Tab. 2), hence, the sum of percentages for this dataset exceeds 100%.

Tab. 2 provides the number of examples present in crowdsourcing instructions of each dataset. From Tab. 2, we can observe that our analysis

---

[2] All benchmarks are in English.

[3] The instructions were obtained from Mishra et al. (2021), who have collected those from the dataset authors.

Dataset	Task	# of Examples
CLARIQ	Clarification QA	18
COSMOSQA	Commonsense Reasoning	8
DROP	Numerical Reasoning	10
DUORC	Paraphrased RC	10
HOTPOTQA	Multi-hop QA	8
HYBRIDQA	QA	17
MC-TACO	Event Duration	3
	Event Ordering	2
	Frequency	2
	Stationary	2
	Absolute Point	2
MULTIRC	Complex QA	7
PIQA	Physical Interaction QA	6
QASC	Complex QA	7
QUOREF	Coreference QA	11
ROPES	RC	14
SCIQA	Science-based QA	6
WINOGRANDE	Commonsense Reasoning	19
Average		8.4

Table 2: Tasks of each dataset and number of examples in crowdsourcing instruction of each dataset. RC: Reading Comprehension, QA: Question Answering.

involves a wide range of different tasks. Also, we believe that the lower number of examples in crowdsourcing instructions might be limiting the imagination of annotators while creating samples, resulting in instruction bias.

2.1 Patterns in Crowdsourcing Instructions

Our goal is to quantify biases in instruction examples that propagate to collected data instances. In this study, we focus on an intuitive form of bias of recurring word patterns, which crowdworkers can easily pick up on. To find such patterns, we manually analyze the instruction examples of each dataset to find a *dominant pattern*, using the following procedure: (a) identifying repeating patterns of $n \geq 2$ words, (b) merging patterns that are semantically similar or have a substantial word overlap, and (c) selecting the most frequent pattern as the dominant pattern (an example is provided in §C).

Tab. 1 shows the dominant pattern in the instruction examples of each dataset. On average, 72.7% of the instruction examples used to create a dataset exhibit the same dominant pattern, and for 10 out of 14 datasets, the dominant pattern covers more than half of the instruction examples. This suggests that crowdsourcing instructions demonstrate a small set of repeating "shallow" patterns. Moreover, the short length of the patterns (2-4 words) and the typical low number of instruction examples (Tab. 2) make the patterns easily visible to crowdworkers, who can end up following them.

Notably, our results are an underestimation of the actual instruction bias, since (a) we only consider the dominant pattern for each dataset (b) our manual analysis over instruction examples has a preference for short patterns (c) we do not consider paraphrased patterns (beyond the shallow paraphrases which are visible in annotation instructions), and (d) datasets may include implicit patterns (e.g. writing style and biases from the annotator's background knowledge) that also contribute to instruction bias. Accounting for such patterns is expected to increase the bias percentage in Tab. 1 further.

2.2 Instruction Bias Propagation to Datasets

We now turn to investigate whether patterns in instruction examples are further propagated by crowdworkers to the collected data. We analyze the train and test sets of each benchmark[4] to find the same patterns, using simple string matching. To account for syntactic modifications in identified patterns based on some examples from dataset, we also consider synonym words where appropriate and match the paraphrased version of each pattern.

Tab. 1 shows the results. Across all datasets, instruction patterns are ubiquitous in the collected data, occurring in 60.5% of the instances on average, with similar presence in training (59%) and test (62%) examples. While the dominant pattern's frequency in the data is typically not higher than in the instructions, for CLARIQ, DUORC, MULTIRC, QUOREF and ROPES, the pattern frequency was amplified by the crowdworkers. Interestingly, these datasets used a relatively large number of instruction examples (Tab. 2), suggesting that more examples do not necessarily alleviate the propagation of instruction bias. Example data instances with instruction patterns are provided in §D.

A natural question that arises is whether patterns in collected data reflect the true task distribution rather than a bias in the instructions. We argue that this is highly unlikely. First, while the space of possible patterns for a NLU task is arguably large, the dataset patterns are imbalanced proportionately

---

[4] If no explicit test set exists, we use the validation set.

	Base			**Large**		
	$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$		$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$	
CLARIQ	30.5	27.4	10.2% ↓	30.3	26.3	13.2% ↓
DROP	75.7	31.8	58% ↓			
MULTIRC	40.5	33.7	16.8% ↓	42	37.4	11% ↓
PIQA	20.7	15	27.5% ↓	21.8	15.3	29.8% ↓
QUOREF	85.9	66	23.2% ↓	92.1	81.1	11.9% ↓
ROPES	57	42	26.3% ↓	57.8	58.2	0.7% ↑
SCIQA	80.7	79.7	1.2% ↓	82.8	81.9	1.1% ↓
Average	55.9	42.2	24.5% ↓	54.5	50	8.3% ↓

Table 3: Performance on $\mathcal{S}^{p}_{\text{test}}$ vs. $\mathcal{S}^{-p}_{\text{test}}$ of models trained on data instances containing instruction patterns ($\mathcal{S}^{p}_{\text{train}}$).

	Base			**Large**		
	$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$		$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$	
CLARIQ	30.5	26.9	11.8% ↓	30.7	26.9	12.4% ↓
DROP	76	78.9	3.8% ↑			
MULTIRC	40.5	39	3.7% ↓	42.4	44.5	5% ↑
PIQA	20.7	19.8	4.4% ↓	21.9	20.6	5.9% ↓
QUOREF	86.7	73.1	15.7% ↓	92.1	81.1	11.9% ↓
ROPES	59.9	48.7	18.7% ↓	59	64.2	8.8% ↑
SCIQA	80.6	80.2	0.5% ↓	82.8	82.9	0.1% ↑
Average	56.4	52.4	7.1% ↓	54.8	53.4	2.6% ↓

Table 4: Performance on $\mathcal{S}^{p}_{\text{test}}$ vs. $\mathcal{S}^{-p}_{\text{test}}$ of models trained on $\mathcal{S}_{\text{train}}$.

to the patterns in the instructions for collecting it. For example, questions for assessing temporal commonsense reasoning could have various forms, such as *"What is the duration of..."*, *"For how much time..."*, and *"How long..."*. However, MC-TACO (event duration) is heavily dominated (87.6%) by questions with the pattern *"how long"*, which appears in 100% of the instruction examples (Tab. 1). In addition, datasets of similar tasks have different dominant patterns, while each dataset's dominant pattern correlates with the pattern in the corresponding instructions; the pattern 'What is' appears in 48% of the questions in QASC and in 57% of the instruction examples, but it is entirely different from the dominant pattern of HOTPOTQA, which is another QA dataset for multi-hop reasoning.

We further validate the propagation of bias in instruction examples by comparing the pattern distributions of collected instances when the instructions include and do not include examples. We conduct this experiment for MC-TACO and QUOREF and find that, without any examples provided, the dominant pattern is substantially less frequent, showing that instruction bias is propagated during data collection. Full details are provided in §E.

Propagation of instruction bias to the test set raises concerns regarding its reliability for evaluation, which we address next.

3 Effect on Model Learning

Let $\mathcal{S}_{\text{train}}$ ($\mathcal{S}_{\text{test}}$) be the set of training (test) examples, and denote by $\mathcal{S}^{p}_{\text{train}}$ ($\mathcal{S}^{p}_{\text{test}}$) and $\mathcal{S}^{-p}_{\text{train}}$ ($\mathcal{S}^{-p}_{\text{test}}$) its disjoint subsets of examples with and without instruction patterns, respectively. We conduct two experiments where we fine-tune models on (a) $\mathcal{S}^{p}_{\text{train}}$ and (b) $\mathcal{S}^{p}_{\text{train}} \cup \mathcal{S}^{-p}_{\text{train}}$, and evaluate them on $\mathcal{S}^{-p}_{\text{test}}$ and $\mathcal{S}^{p}_{\text{test}}$. This is to assess to what extent models generalize from instruction patterns to the downstream task (a), and to compare model performance on instances with and without instruction patterns (b).

3.1 Experimental Setting

Datasets Since model training is computationally expensive, we select a subset of seven datasets from those analyzed in §2: (1) CLARIQ, (2) DROP, (3) MULTIRC, (4) PIQA, (5) QUOREF, (6) ROPES, and (7) SCIQA. These datasets cover a variety of tasks, different types and levels of instruction bias (Tab. 1), and are different in size (§B).

Models For all datasets except DROP, we evaluate T5-base and T5-large (Raffel et al., 2020), and BART-base and BART-large (Lewis et al., 2020). For DROP, we use Numnet+ (Ran et al., 2019), a RoBERTa model (Liu et al., 2019) with specialized output heads for numerical reasoning. Numnet+ has 355M parameters, which is closer to T5-base (220M) than to T5-large (770M) in size.

Evaluation We evaluate model performance using the standard F_1 evaluation score, and report the average score over three random seeds.

3.2 Results

We observe similar results for T5 and BART, and thus, present only the results for T5 in this section. Results for BART are provided in §F.

Models often fail to generalize beyond instruction patterns. Tab. 3 shows the performance on $\mathcal{S}^{p}_{\text{test}}$ and $\mathcal{S}^{-p}_{\text{test}}$ when training only on examples with instruction patterns. Across all experiments, there are large performance gaps, reaching to 58% in DROP and $> 10\%$ in both base and large models for CLARIQ, MULTIRC, PIQA, and QUOREF. This indicates that models trained only on examples with instruction patterns fail to generalize to other task examples, and stresses that instruction bias should be monitored and avoided during data collection. Notably, the gap is lower for large models than for base ones, showing that large models are less sensitive to instruction bias. This might be attributed to their larger capacity to capture knowledge and skills during pre-training.

Model performance is overestimated by instruction bias. We compare the performance on $\mathcal{S}_{\text{test}}^{p}$ and $\mathcal{S}_{\text{test}}^{-p}$ of models trained on the full training set (Tab. 4). The average performance across all datasets is higher on examples that exhibit instruction patterns by $\sim 7\%$ and $\sim 3\%$ for the base and large models, respectively. Specifically, base models perform worse on $\mathcal{S}_{\text{test}}^{-p}$ than on $\mathcal{S}_{\text{test}}^{p}$ for all datasets except DROP, in some cases by a dramatic gap of $> 15\%$ (e.g. 18.7% in ROPES and 15.7% in QUOREF). In contrast, results for the large models vary across datasets, while the performance gap is generally smaller in magnitude. This shows that model performance is often overestimated by instructions bias, and reiterates that large models are generally less sensitive to instruction patterns.

4 Conclusions and Discussion

We identify a prominent source of bias in crowdsourced NLU datasets, called *instruction bias*, which originates in annotation instructions written by dataset creators. We study this bias in 14 NLU benchmarks, showing that instruction examples used to create NLU benchmarks often exhibit clear patterns that are propagated by annotators to the collected data. In addition, we investigate the effect of instruction bias on model performance, showing that instruction patterns can lead to overestimated performance as well as limit the ability of models to generalize to other task examples.

Based on our findings, we derive three recommendations for future crowdsourced NLU benchmarks: (1) Crowdsourcing instructions should be diverse; this could be achieved, for example, by having a large number of instructive examples, rephrasing examples using neural models, or periodically sampling examples from a *diverse* set of previously collected examples. The latter could be done, for example, by maintaining a pool of diverse examples during the collection process and then presenting every annotator with a different random sample from this growing pool. (2) Word patterns in collected instances should be analyzed during data collection, as well as possible correspondence to instruction examples. Such analysis will help researchers monitor the collection process and the quality of the resulting data. (3) Correlation between model performance and input patterns should be checked during evaluation.

Limitations

This work covers 14 NLU datasets, for which annotation instructions are publicly available. However, most of these datasets are QA datasets. Our analysis can be extended to other NLU task categories, such as Natural Language Inference (NLI) and Relation Extraction (RE).

Our study reveals a concrete bias that skews the collected data distribution toward specific patterns. While the effect of instruction examples on collected data is prominent, it is hard to quantify how different the distribution of crowdsourced examples is from the natural distribution of the task. Concretely, to conduct a study that compares the distributions of crowdsourced versus natural complex reasoning questions, datasets of complex natural questions are needed. However, to the best of our knowledge, as of today, no such datasets exist.

In our analysis, we focused on shallow patterns based on word matching, however, it is known that there are other types of biases that are implicit in the text. Exploring these kinds of biases can be an interesting future direction. In addition, our analysis of model performance is based on splitting dataset instances based on the dominant pattern. However, it might be possible that there are more patterns, and the non-pattern subset might include other less frequent patterns. Hence, exploring the effect of different less frequent patterns on model learning can be a future work.

Last, our work studied the effect of instruction bias on widely used generative models (i.e., T5 and BART); it would be valuable to investigate whether our findings hold in encoder-only models, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019).

Acknowledgement

We thank Daniel Khashabi, Sewon Min, and Avia Efrat for helpful feedback, and the anonymous reviewers for constructive suggestions. We acknowledge the Research Computing (RC) at Arizona State University (ASU) for providing computing resources for experiments.

References

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2020. Convai3: Generating clarifying questions for open-domain dialogue systems (clariq). *arXiv preprint arXiv:2009.11352*.

Anjana Arunkumar, Swaroop Mishra, Bhavdeep Sachdeva, Chitta Baral, and Chris Bryan. 2020. Real-time visual feedback for educative benchmark creation: A human-and-metric-in-the-loop workflow. *NeurIPS 2020 Workshop HAMLETS*.

Stephen H Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, et al. 2022. Promptsource: An integrated development environment and repository for natural language prompts. *arXiv preprint arXiv:2202.01279*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 1–12, Los Angeles. Association for Computational Linguistics.

Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 2334–2346.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5925–5932, Hong Kong, China. Association for Computational Linguistics.

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

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Avia Efrat and Omer Levy. 2020. The turking test: Can language models understand instructions? *arXiv preprint arXiv:2010.11982*.

Carsten Eickhoff. 2018. Cognitive biases in crowdsourcing. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 162–170.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166, Hong Kong, China. Association for Computational Linguistics.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Danula Hettiachchi, Mark Sanderson, Jorge Goncalves, Simo Hosio, Gabriella Kazai, Matthew Lease, Mike Schaekermann, and Emine Yilmaz. 2021. Investigating and mitigating biases in crowdsourced data. In *Companion Publication of the 2021 Conference on Computer Supported Cooperative Work and Social Computing*, pages 331–334.

Xiao Hu, Haobo Wang, Anirudh Vegesana, Somesh Dube, Kaiwen Yu, Gore Kao, Shuo-Han Chen, Yung-Hsiang Lu, George K Thiruvathukal, and Ming Yin. 2020. Crowdsourcing detection of sampling biases in image datasets. In *Proceedings of The Web Conference 2020*, pages 2955–2961.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8706–8716, Online. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. QASC: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8082–8090.

Walter S Lasecki, Mitchell Gordon, Danai Koutra, Malte F Jung, Steven P Dow, and Jeffrey P Bigham. 2014. Glance: Rapidly coding behavioral video with the crowd. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 551–562.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew E Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *ICML*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 58–62, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2021. Variational information bottleneck for effective low-resource fine-tuning. *ICLR*.

Swaroop Mishra and Anjana Arunkumar. 2021. How robust are model rankings: A leaderboard customization approach for equitable evaluation. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 35, pages 13561–13569.

Swaroop Mishra, Anjana Arunkumar, Chris Bryan, and Chitta Baral. 2020a. Our evaluation metric needs an update to encourage generalization. *ArXiv*, abs/2007.06898.

Swaroop Mishra, Anjana Arunkumar, Bhavdeep Sachdeva, Chris Bryan, and Chitta Baral. 2020b. Dqi: Measuring data quality in nlp. *arXiv preprint arXiv:2005.00816*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *ACL*.

Maryam M Najafabadi, Flavio Villanustre, Taghi M Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic. 2015. Deep learning applications and challenges in big data analytics. *Journal of big data*, 2(1):1–21.

Nikita Nangia, Saku Sugawara, Harsh Trivedi, Alex Warstadt, Clara Vania, and Samuel R. Bowman. 2021. What ingredients make for an effective crowdsourcing protocol for difficult NLU data collection tasks? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1221–1235, Online. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hossein A Rahmani and Jie Yang. 2021. Demographic biases of crowd workers in key opinion leaders finding. *arXiv preprint arXiv:2110.09248*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. NumNet: Machine reading comprehension with numerical reasoning. In *Proceedings of EMNLP*.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension. *arXiv preprint arXiv:2107.12708*.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the roc story cloze task. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 15–25.

Alane Suhr, Clara Vania, Nikita Nangia, Maarten Sap, Mark Yatskar, Samuel R. Bowman, and Yoav Artzi. 2021. Crowdsourcing beyond annotation: Case studies in benchmark data collection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 1–6, Punta Cana, Dominican Republic & Online. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705*.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Feifei Zheng, Ruoling Tao, Holger R Maier, Linda See, Dragan Savic, Tuqiao Zhang, Qiuwen Chen, Thaine H Assumpção, Pan Yang, Bardia Heidari, et al. 2018. Crowdsourcing methods for data collection in geophysics: State of the art, issues, and future directions. *Reviews of Geophysics*, 56(4):698–740.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

A Biases in NLU Benchmarks

Crowdsourcing has been a widely adapted approach to create large scale datasets such as SQuAD 1.1(Rajpurkar et al., 2016, 2018), DROP(Dua et al., 2019), QUOREF(Dasigi et al., 2019) and many more (Najafabadi et al., 2015; Callison-Burch and Dredze, 2010; Lasecki et al., 2014; Zheng et al., 2018; Chang et al., 2017). Many past works investigate different types of bias in crowdsourcing datasets such as cognitive bias (Eickhoff, 2018), annotator bias (Gururangan et al., 2018; Geva et al., 2019), sampling bias (Hu et al., 2020), demographic bias (Rahmani and Yang, 2021) and others (Hettiachchi et al., 2021). Many works on bias in NLU benchmarks focus on biases resulting from the crowdsourcing annotations, and how annotator-specific patterns create biases in data (Geva et al., 2019).

To mitigate the bias, prior works have focused on priming crowdsourcing annotators with minimal information to increase their imagination (Geva et al., 2021; Clark et al., 2020) to avoid recurring patterns. Arunkumar et al. (2020) develops a real time feedback and metric-in-the loop (Mishra et al., 2020b) workflow to educate crowdworkers in controlling dataset biases. Nangia et al. (2021) provides an iterative protocol with expert assessments for crowdsourcing data collection to increase difficulty of instances. (Swayamdipta et al., 2020) introduces dataset map as a model-based tool to characterize and diagnose datasets. Also, Karimi Mahabadi et al. (2020); Mahabadi et al. (2021) propose learning strategies to train neural models, which are more robust to such biases and transfer better to out-of-domain datasets.

In this work, we show that biases exhibited by annotators start from the crowdsourcing instructions designed by dataset creators.

B Dataset Statistics

Tab. 5 describes the statistics of train and evaluation sets of datasets used in our experiments. Here, we can observe that each selected dataset differs in terms of number of training samples, % of instruction patterns, and tasks.

C Pattern Extraction Method

Here, we describe an example to show how we extract the dominant pattern from the crowdsourcing instructions and subsequently identify the same pattern in the dataset. We try to find recurring word patterns such as *“Are you...”*, *“how many points...”*, *“Was... still...”*, *“since... the...”*.

For example, MC-TACO (event duration) has 3 examples in crowdsourcing instructions: (1) how long did Jack play basketball?, (2) how long did he do his homework?, and (3) how long did it take for him to get the Visa? In step (a), we analyze examples manually and find *dominant pattern*. Here, we can see that all examples contain tri-gram pattern, i.e., *“how long did"*. In step (b), we try to generate more possible patterns that are semantically similar to the *dominant pattern* or have a significant word overlap. Here, *“how long did"* can be *“how long was"*, *“how long does"*, etc. (i.e, `How long AUX`). In step (c), we look for all these possible patterns in datasets using simple word-matching techniques.

D Pattern Examples

Tab. 8 provides dataset, instruction patterns and corresponding examples of data instances that exhibit the instruction patterns.

E The Effect of Instruction Examples on Pattern Frequency in Collected Data

To study the effect of bias in instruction examples on collected data, we asked NLP graduate students to write five questions for each of (1) temporal reasoning (event duration) and (2) coreference resolution, based on the crowdsourcing instructions of MC-TACO and QUOREF, respectively. For each task, we conduct two surveys, where the instructions include and do not include any examples.

We collected responses from 10 participants. The dominant patterns of MC-TACO (‘how long’) and QUOREF (‘What is the name’) only contribute to 38% and 8% of our collected data where examples are not given, in contrast to 68% (↑79%) and 32% (↑300%) in collected data where examples are given. This indicates that crowdsourcing examples bias crowdworkers to follow certain patterns, whereas showing no examples increases the creativity of crowdworkers.

In addition, our collected responses where examples are not given contain 10 and 9 unique patterns for MC-TACO (event duration) and QUOREF respectively, in contrast to only 4 and 5 unique patterns in collected data where examples are given. Our finding shows that there is substantial linguistic diversity associated with the NLP tasks, unlike

Dataset	**Train**						**Test**					
	$\mathcal{S}_{\text{train}}$	$\mathcal{S}^{p}_{\text{train}}$		$\mathcal{S}^{-p}_{\text{train}}$			$\mathcal{S}_{\text{test}}$	$\mathcal{S}^{p}_{\text{test}}$		$\mathcal{S}^{-p}_{\text{test}}$		
CLARIQ	8566	7286	85.1%	1280	14.9%		4499	4006	89%	493	11%	
DROP	77409	48422	62.5%	28987	37.5%		9536	5960	62.5%	3576	37.3%	
MULTIRC	5131	1972	38.4%	3159	61.6%		953	395	41.5%	558	58.6%	
PIQA	17171	7508	43.7%	9663	56.3%		3268	1401	42.9%	1867	57.1%	
QUOREF	19399	11052	57%	8347	43%		2418	1451	60%	967	40%	
ROPES	1412	1046	74.1%	366	25.9%		203	42	20.7%	161	79.3%	
SCIQA	11679	9765	83.61%	1914	16.4%		1000	845	84.5%	155	15.5%	
Total	140767	87051	61.8%	53716	38.2%		21877	14100	64.5%	7777	35.6%	

Table 5: Statistics of number of train and test examples with and without instruction patterns. $\mathcal{S}_{\text{train}}$: set of examples in train set, $\mathcal{S}^{p}_{\text{train}}$: set of examples in train set with instruction pattern, $\mathcal{S}^{-p}_{\text{train}}$: set of examples in train set without instruction pattern, $\mathcal{S}_{\text{test}}$: set of examples in test set, $\mathcal{S}^{p}_{\text{test}}$: set of examples in test set with instruction pattern, $\mathcal{S}^{-p}_{\text{test}}$: set of examples in test set without instruction pattern.

	Base			**Large**		
	$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$		$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$	
CLARIQ	30	26.2	12.7% ↓	29.2	25.6	12.3% ↓
MULTIRC	27.9	15.1	45.9% ↓	31.1	21	32.5% ↓
PIQA	21.9	15.3	30.1% ↓	23	16.1	30% ↓
QUOREF	78.8	45.8	41.9% ↓	87	58.6	32.6% ↓
ROPES	43.9	33.1	24.6% ↓	47.2	39.1	17.2% ↓
SCIQA	76.8	66.1	13.9% ↓	77.5	69.6	10.2% ↓
Average	46.6	33.6	27.9% ↓	49.2	38.3	22.2% ↓

Table 6: Performance of BART models on $\mathcal{S}^{p}_{\text{test}}$ vs. $\mathcal{S}^{-p}_{\text{test}}$ of models trained on data instances containing instruction patterns ($\mathcal{S}^{p}_{\text{train}}$).

	Base			**Large**		
	$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$		$\mathcal{S}^{p}_{\text{test}}$	$\mathcal{S}^{-p}_{\text{test}}$	
CLARIQ	29.8	26	12.8% ↓	29.4	26.3	10.5% ↓
MULTIRC	28.5	29.8	4.6% ↑	41.5	33.9	18.3% ↓
PIQA	22.3	20.5	8.1% ↓	23.1	21.6	6.5% ↓
QUOREF	80.5	61.2	24% ↓	87.8	73.4	16.4% ↓
ROPES	44	44.1	0.2% ↑	47.9	46.5	2.9% ↓
SCIQA	76.5	70.6	7.7% ↓	52.2	50.8	2.7% ↓
Average	46.9	42	10.5% ↓	47	42.1	10.4% ↓

Table 7: Performance of BART models on $\mathcal{S}^{p}_{\text{test}}$ vs. $\mathcal{S}^{-p}_{\text{test}}$ of models trained on $\mathcal{S}_{\text{train}}$.

the patterns covered in instruction examples that get propagated to corresponding datasets.

The task instructions and collected annotations are available at `https://github.com/Mihir3009/instruction-bias/blob/main/SURVEY.md`.

F Additional Results

Tab. 6 and Tab. 7 show the performance of BART on $\mathcal{S}^{p}_{\text{test}}$ and $\mathcal{S}^{-p}_{\text{test}}$ when training only on examples with instruction patterns and the full training set, respectively. From Tab. 6, there are large performance gaps reaching 45.9% in MULTIRC and $>$ 20% in both base and large models for QUOREF, and PIQA. Overall, the average performance across all datasets is 27.9% and 22.2% higher on $\mathcal{S}^{p}_{\text{test}}$ for the base and large models, respectively. This indicates that both base and large models often fail to generalize beyond instruction patterns.

From Tab. 7, we see that the average performance across all datasets is higher on examples that exhibit instruction patterns by $\sim$ 10.5% for both base and large models. From the results, we can conclude that the model's performance is overestimated by instruction bias.

Dataset	Pattern	Examples
CLARIQ	[Are|Would|Do] you	Are you looking for a specific web site?
		What kind of train are you looking for?
		Do you want to watch news videos or read the news?
		Would you like the location of the ritz carlton lake las vegas?
COSMOSQA	What AUX	What may happen after the young man makes his call?
		What might happen if you have him for the whole day?
		What's a possible reason the writer doesn't look disabled on the outside?
DROP	How many [field goals | years | yards | points | touchdowns]	How many touchdowns did Jones have?
		How many field goals did Kris Brown kick
		How many yards was the longest touchdown of the game?
		After Akers 32-yard field goal, how many points behind was Washington?
HOTPOTQA	[in|of|from|_] [Which|What] AUX	Which franchise was founded in 1978, Chuck E. Cheese's or Jet's Pizza?
		Busan, in the area surrounding the mountain of Geumjeongsan, is the second most populated city in which country?
		What is the name of the third album from singer Selena Quintanilla-Pérez?
MC-TACO	How long AUX	How long was his mother ill?
	What AUX	What did the government decide after the 9/11 attack?
	How often AUX	How often would one family be able to do something like this?
	AUX... [still|always]	Will electronic espionage always be happening in the U.S.?
		Is she still gone?
	When did / What time	What time did the planes crash into the World Trade Center?
		When did Durer die?
MULTIRC	What AUX	What was Poe's first published work?
		What is the full name of the person described?
		What kind of career does Christie Brinkley have?
PIQA	How [do|can]	How do I make orange icing if I have store-bought white frosting?
		How can I make popsicles for dogs?
		Are you nervous about giving a speech or doing something? How can you calm yourself?
QUOREF	What AUX the [full | real | first |last] name	What is the first name of the person who purchases a revolver?
		What is the full name of the person who is calmly asked to leave?
		What was the name of the house where Appleton Water Tower was built?
		What is the last name of the person who convinces the girls to help him look for the treasure?
ROPES	Which AUX	Which area would be less likely to experience a drought and have better chance at a new growth?
		Which hair spray brand should Greg buy to be environmentally friendly?
		Which markalong was produced asexually?
SCIQA	What AUX	What are by far the most common type of invertebrate?
		What do waves deposit to form sandbars and barrier islands?
		What is the term for the total kinetic energy of moving particles of matter?
WINO-GRANDE	[because | so | while | since | but] ... the	The dog didn't like its collar but was okay with its leash because the _ was loose on it.
		Hunter took Benjamin's clothes to the laundromat, since _ had the day off that day.
		James sang his song at the top of his voice so as to be heard over the noise but the _ is louder.

Table 8: Examples of data instances from original dataset that contain instruction patterns. AUX $\in$ {am, is, are, was, were, has, have, had, do, does, did, will, would, can, could, may, might, shall, should, must}. _ : <blank>.

Performance Prediction via Bayesian Matrix Factorisation for Multilingual Natural Language Processing Tasks

Viktoria Schram **Daniel Beck** **Trevor Cohn**
School of Computing and Information Systems
The University of Melbourne
vschram@student.unimelb.edu.au
{d.beck,t.cohn}@unimelb.edu.au

Abstract

Performance prediction for Natural Language Processing (NLP) seeks to reduce the experimental burden resulting from the myriad of different evaluation scenarios, e.g., the combination of languages used in multilingual transfer. In this work, we explore the framework of Bayesian matrix factorisation for performance prediction, as many experimental settings in NLP can be naturally represented in matrix format. Our approach outperforms the state-of-the-art in several NLP benchmarks, including machine translation and cross-lingual entity linking. Furthermore, it also avoids hyperparameter tuning and is able to provide uncertainty estimates over predictions.

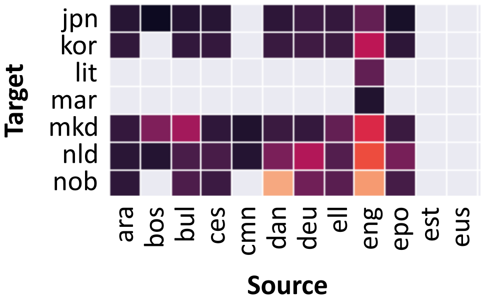


Figure 1: Matrix representation showing a subset of the performance scores for Wiki MT (Schwenk et al., 2021). Black: Lowest performance scores. Light orange: Highest performance scores. Grey: No performance score available. The goal is to predict the performance of language pairs depicted by grey cells.

1 Introduction

Natural language processing (NLP) is an empirical discipline, with progress driven by a myriad of tasks, domains, computational models and datasets (Xia et al., 2020). Models are often developed to be applicable to a large number of languages. Given that there are more than 7000 languages spoken in the world (Haddow et al., 2022), evaluating the model performance of each possible NLP scenario leads to a combinatorial explosion and would require excessive time for training and testing, as well as the significant costs of evaluation resource creation. This is very time consuming and computationally expensive, especially for large models and massively multilingual applications (Xia et al., 2020; Sharir et al., 2020).

To illustrate, for neural machine translation (MT) models, retraining and testing for each new language pair is required (Fan et al., 2021), which quickly increases the number of experiments (as shown in Figure 1). In those scenarios, whether the model is appropriate for the considered test languages is not clear upfront. Similarly, for cross-lingual transfer (TSF) tasks like parsing (Das and Sarkar, 2020) or part-of-speech (POS) tagging (de Vries et al., 2022), the performance on all languages of interest is only known after training and testing has been performed. This is particularly interesting for the development of NLP technologies for low-resource languages (de Vries et al., 2022).

In this paper we propose a method for performance prediction which can solve the above problems through automatically estimating performance scores of machine learning models, thus avoiding the need for excessive training and test cycles, and data creation costs. The goal is to predict the performance of a model solely based on past experimental records – a task of great interest for decision making in aforementioned high-cost applications. We propose to use Bayesian probabilistic matrix factorization methods and incorporate context features describing the language pairs of interest. This is realisable because many performance prediction problems in NLP can be modelled as a matrix, as illustrated in Figure 1. The result will inevitable be an incomplete matrix because some data are missing, as they were either to costly to obtain or no datasets were available (Xia et al., 2020; Schwenk et al., 2021).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1790–1801
May 2-6, 2023 ©2023 Association for Computational Linguistics

Our key contribution is a Bayesian approach for performance prediction, contrasting with prior work which is exclusively frequentist. This confers the following advantages: 1) it enables us to quantify the uncertainty of our predictions by providing credible intervals (CIs); and 2) predictions of performance scores can be provided in a more principled way via a posterior distribution after integrating over the model's parameters and hyperparameters. We provide an extensive experimental study of matrix factorization methods, including our proposed Bayesian technique, applied to several important benchmarks, and show several improvements over the state-of-the-art.

2 Problem Formulation and Methods

As demonstrated in Figure 1, the performance scores for a suitable NLP task can be arranged in a matrix with the rows displaying the source languages and the columns showing the target languages[1]. The matrix is the score matrix $\mathbf{R} \in \mathbb{R}^{N_s \times N_t}$ and each cell (s, t) contains either the performance score for a source-to-target language relationship or is empty. The number of considered source and target languages is denoted by N_s and N_t, respectively.

The performance of a model, measured by an evaluation metric (e.g. BLEU, Papineni et al. (2002)), depends on properties like the model architecture $\mathcal{M}$, training dataset $\mathcal{D}$, languages $\mathcal{L}$, the training procedure $\mathcal{P}$ and the test dataset $\mathcal{D}'$. The score matrix $\mathbf{R}$ is then given by

$$\hat{\mathbf{R}}_{\mathcal{L},\mathcal{D}} = f_\theta([\Phi_{\mathcal{L}}; \Phi_{\mathcal{D}}]),$$

with $f_\theta(\cdot)$ being a function which denotes a linear or non-linear relationship of its arguments and $\Phi_{\mathcal{X}}$ represents the feature set of each characteristic $\mathcal{X}$. The dependency on test set features is omitted. We assume that the distributions of training and test datasets are the same, ignoring a possible domain shift (following Xia et al. (2020)).

As the completion of an arbitrary matrix is ill-posed, we reformulate $f_\theta(\cdot)$ as a low-rank matrix approximation problem. In its simplest, noiseless form, the matrix $\mathbf{R}$ can be reconstructed as (Koren et al., 2009)

$$\mathbf{R} \approx \mathbf{W}^T \mathbf{H}, \tag{1}$$

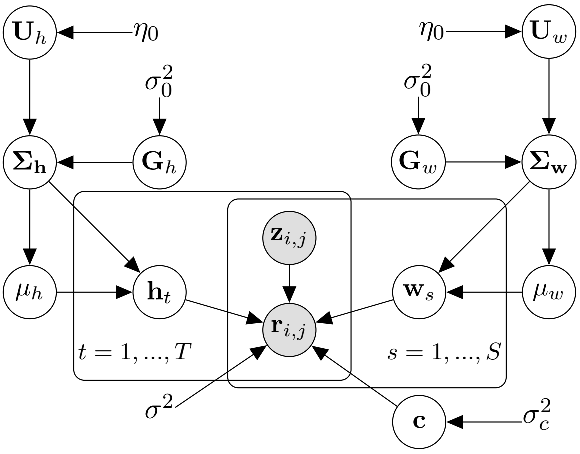


Figure 2: The graphical model for our approach based on Bayesian Probabilistic Matrix Factorisation with context information.

using two lower rank matrices $\mathbf{W} \in \mathbb{R}^{D \times N_s}$ and $\mathbf{H} \in \mathbb{R}^{D \times N_t}$, with $\mathbf{W}$ and $\mathbf{H}$ being the latent matrices for the source and target languages, respectively. Due to the low-rank assumption, for the latent dimensions it holds that $D \ll N_s, N_t$. Hence, each individual score $r_{s,t}$ of a source-target pair can be calculated by $r_{s,t} \approx \mathbf{w}_s^T \mathbf{h}_t$, where each latent source and target language vector is given by a low-dimensional vector, $\mathbf{w}_s^T \in \mathbb{R}^D$, $\mathbf{h}_t \in \mathbb{R}^D$, respectively (Cabral et al., 2013; Chen et al., 2018b).

The matrix factorisation (MF) approach described above is inspired by research on recommender systems, which use latent factor models for preference predictions (Koren et al., 2009). In those models, the low rank behavior is a widely used assumption. We assume in our model that the performance of a bilingual task is determined by a small number of unobserved factors. Using a linear factor model the performance for translating from a source language is modeled by linearly combining target language factor vectors using source language specific coefficients.

2.1 Bayesian Probabilistic Matrix Factorisation

Our approach is represented by the graphical model shown in Figure 2. It illustrates a linear probabilistic model, where the conditional distribution over the observed ratings, assuming Gaussian observation noise, is given as

$$p(\mathbf{R}|\mathbf{W}, \mathbf{H}, \mathbf{c}, \mathbf{z}, \sigma^2) = \prod_{s=1}^{S} \prod_{t=1}^{T} [\mathcal{N}(r_{s,t}|\mathbf{w}_s^T \mathbf{h}_t + \sum_{i=1}^{N} c_i z_{s,t}^i, \sigma^2)]^{I_{s,t}}, \tag{2}$$

---

[1] This is an assumption, appropriate for our considered tasks. Without any loss of generality, this scenario can be expanded to other NLP settings which can be reformulated as a matrix, e.g. hyperparameter search for language models.

with $I_{s,t}$ being the indicator function that is 1 if a source language s has a performance score for a target language j and 0 otherwise. $\mathcal{N}(\mu, \sigma^2)$ denotes a Gaussian distribution with mean μ and variance σ^2. The mean of this likelihood function consists of a latent factor model $\mathbf{w}_s^T \mathbf{h}_t$ and is also additionally regressed on the available context information given by $z_{s,t}^i$ for $i \in 1, ..., N$ available context features (Chen et al., 2018b). The prior for the regression coefficients c_i of each context feature z_i is given by $p(c_i|\sigma_c^2) = \mathcal{N}(c_i|0, \sigma_c^2)$. Furthermore, spherical Gaussian priors are placed for source and target language latent feature vectors given by

$$p(\mathbf{W}|\sigma_w^2) = \prod_{s=1}^{S} \mathcal{N}(w_s|\mu_w, \mathbf{\Sigma}_w) \tag{3}$$

$$p(\mathbf{H}|\sigma_h^2) = \prod_{t=1}^{T} \mathcal{N}(h_t|\mu_h, \mathbf{\Sigma}_h), \tag{4}$$

with hyperparameters $\Theta_w = \{\mu_w, \mathbf{\Sigma}_w\}$ and $\Theta_h = \{\mu_h, \mathbf{\Sigma}_h\}$, where $\mathbf{\Sigma}$ is the covariance matrix. In contrast to the original work (Salakhutdinov and Mnih, 2008) on which we base our approach, we choose to incorporate the Lewandowski-Kurowicka-Joe (LKJ) prior (Lewandowski et al., 2009) into the computation of samples for the covariance matrices. This method is better suited for modern Bayesian computations, due to sampling difficulties the inverse-Wishart prior causes and its restrictive form (Team, 2018; Barnard et al., 2000). A correlation matrix $\mathbf{U}$, with elements $u_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$ and σ being the standard deviation, can be reformulated into a covariance matrix $\mathbf{\Sigma}$ using the separation strategy (Barnard et al., 2000) as

$$\mathbf{\Sigma} = \mathbf{GUG}, \tag{5}$$

with $\mathbf{G} = \sqrt{\text{diag}(\mathbf{\Sigma})}$. The matrix $\mathbf{U}$ is sampled from the LKJ prior given by

$$p(\mathbf{U}|\eta_0) = c \cdot \det(\mathbf{U})^{\eta_0 - 1},$$

indicating a uniform distribution on correlation matrices for $\eta_0 = 1$, a normalization constant c and $\det(\cdot)$ denoting the determinant. As a prior for $\mathbf{G}$ we choose the halfnormal distribution (Seyboldt, 2019), as it holds that $\sigma > 0$, which is given for each element $g \in \mathbf{G}$ by

$$p(g|\sigma_0^2) = \frac{\sqrt{2}}{\sigma_0 \sqrt{\pi}} \exp\left(-\frac{g^2}{2\sigma_0^2}\right), \; g \geq 0,$$

with unit variance σ_0^2. This is equivalent to placing an inverse-Gamma distribution on the precision. The covariance matrix $\mathbf{\Sigma}$ obtained according to Equation (5) is used to parameterise the priors given in Equations (3) and (4) and the Gaussian prior of their means. More information can be found in Salakhutdinov and Mnih (2008).

The predictive distribution is obtained by marginalizing over the model's parameters and hyperparameters. Since exact evaluation of this predictive distribution is analytically intractable, we use approximate inference via Markov chain Monte Carlo (MCMC) sampling, leading to an approximation of the predictive posterior as (Salakhutdinov and Mnih, 2008):

$$p(\mathbf{R}^*|\mathbf{R}, \Theta_0) \approx \frac{1}{K} \sum_{k=1}^{K} p(\mathbf{R}^*|\mathbf{W}^k, \mathbf{H}^k), \tag{6}$$

for $\Theta_0 = \{\eta_0, \sigma_0^2\}$ and $\mathbf{R}^*$ denoting the matrix of predictions on the test set. The K samples are generated by running a Markov chain. Its stationary distribution will be the posterior distribution over the model parameters and hyperparameters $\{\mathbf{W}, \mathbf{H}, \Theta_w, \Theta_h\}$.

2.2 Credible Intervals

One measure of uncertainty in Bayesian inference is the credible interval (CI) or more generally a credible set (Casella and Berger, 2021). A $100(1-\beta)\,\%$ equal tail CI for a random variable X is an interval $[a, b]$ such that the probability that X lies in the interval is $1 - \beta$, given as

$$P(X \in [a, b]) = 1 - \beta.$$

In Bayesian inference, analyses are made considering the posterior distribution of the parameter of interest. Hence, a CI provides us with upper and lower bounds which define an interval with the probability of interest around the mean of the posterior distribution (Hespanhol et al., 2019; Rice and Ye, 2022). It quantifies how precise the obtained posterior, i.e. the posterior belief, is, with a narrower interval around the point estimate indicating more certainty in the predictions. Hence, using the samples from Equation (6), an approximation of the posterior predictive distributions for each language pair can be obtained and used to construct CIs in addition to the point estimates. This is useful for performance prediction in NLP settings, as the widths and bounds of the obtained CIs can guide further decision-making regarding whether a model is theoretically worth deploying.

Task	dim	cells	empty	test	feats
MT^{TSF}	54×54	2916	0	573	21
POS^{TSF}	26×60	1560	14	29	15
MT^{Wiki}	39×39	1521	526	199	22
$\text{Parsing}^{\text{TSF}}$	30×30	900	0	174	15
EL^{TSF}	9×54	486	0	96	12
BLI	15×15	225	95	26	6

Table 1: Matrix specifications for each task. Dim: Matrix dimensions. Cells: Number of cells in each matrix. Empty: Number of empty cells in the matrix, not being cells on the diagonal. Test: Number of cells used in the test set. Feats: Number of features.

2.3 Non-Bayesian Matrix Factorisation Methods

Probabilistic MF (PMF) has been shown to perform well on sparse and imbalanced datasets (Mnih and Salakhutdinov, 2007). The conditional distribution over the observed ratings, assuming Gaussian observation noise given by Equation (2), without considering additional context information. As in Bayesian PMF (BPMF) spherical Gaussian priors are placed on $\mathbf{W}$ and $\mathbf{H}$ following Equations (3) and (4), assuming $\mu_w = \mu_h = 0$; $\mathbf{\Sigma}_w = \sigma_w^2 \mathbf{I}$ and $\mathbf{\Sigma}_h = \sigma_h^2 \mathbf{I}$. Learning the latent source and target matrices is done via maximizing the log-posterior, where all hyperparameters, being the observation noise and prior variances, are fixed. This is equal to the following optimization problem, where latent factor representations are learned by finding the matrices $\mathbf{W}$ and $\mathbf{H}$ which minimize the regularized squared error, given by (Chen et al., 2018b)

$$\mathbf{W}^*, \mathbf{H}^* = \underset{\mathbf{W},\mathbf{H}}{\arg\min} \frac{1}{2} \sum_{(s,t)\in\delta(\mathbf{R})} (\mathbf{r}_{s,t} - \mathbf{w}_s^T \mathbf{h}_t)^2$$
$$+\beta_w \sum_{s=1}^{N_s} ||\mathbf{w}_s||_2^2 + \beta_h \sum_{t=1}^{N_t} ||\mathbf{h}_t||_2^2, \quad (7)$$

where $||\cdot||_2^2$ is the squared l_2 norm[2]. Additionally, separate regularization parameters for $\mathbf{W}$ and $\mathbf{H}$ are assumed, being β_w and β_h, respectively. In the case of missing values, the optimization problem is on the set of known ratings $\delta(\mathbf{R})$.

Additionally, the framework of MF offers the option of biased MF, which provides a means of incorporating context. It incorporates model characteristics of each source language, b_s, and target language b_t and considers a global tendency μ which is independent of source-target language interactions. The biases account for the fact that certain performance values might contain universal shifts or exhibit systematic tendencies with respect to certain source and target languages. Furthermore, it is common that all performance predictions are non-negative and in a certain range, i.e., the global bias μ accounts for global effects. The predictions are given by (Chen et al., 2018b)

$$\hat{r}_{s,t}^{\text{CTX}} = \mathbf{w}_s^{\text{T}} \mathbf{h}_t + \sum_{i=1}^{N} c_i z_{s,t}^i + \mu + b_s + b_t. \quad (8)$$

During learning, the optimization problem in Equation (7) has to be adapted to $\hat{r}_{s,t}^{\text{CTX}}$ accordingly, also adding additional regularization terms for b_s and b_t.

The non-probabilistic approaches, but also PMF need manual control of their hyperparameters, which makes those approaches computationally expensive to develop compared to BPMF.

3 Experiments and Analysis

Following the evaluation setting of *NLPerf* (Xia et al., 2020), we consider a set of tasks described in the following, all of which allow a reformulation into a matrix representation. The performance scores are predicted for bilingual lexicon induction (BLI); machine translation on aligned Wikipedia data (Wiki^{MT}), and with cross-lingual transfer for translation into English (MT^{TSF}); cross lingual dependency parsing ($\text{Parsing}^{\text{TSF}}$); cross-lingual POS tagging (POS^{TSF}) and cross-lingual entity linking (EL^{TSF}). Despite also considering cross-lingual transfer tasks, for reasons of simplicity and without any loss of generality, we refer to each cell in the matrix as a *source-target* language pair unless otherwise stated. All tasks and their matrix properties are stated in Table 1. More information about the models underlying each task can be found in Appendix A.

To allow a fair comparison to the closest related work *NLPerf* (Xia et al., 2020), we consider an identical set of features for our predictions: Dataset size, word/subword vocabulary size, average sentence length, word/subword overlap, type-token ratio, type-token ratio distance, single tag type, fused tag type, average tag length per word, dependency arcs matching WALS features and six distance features from the URIEL Typological Database (Littell

---

[2] Solving this optimization problem to obtain a point estimate is referred to as MF in our experiments. We use a Bayesian approach for PMF and therefore apply MCMC sampling, which is another alternative and can be easily extended via hyperpriors to BPMF.

Task	MT$^{\text{Wiki}}$	Parsing$^{\text{TSF}}$	EL$^{\text{TSF}}$	POS$^{\text{TSF}}$	MT$^{\text{TSF}}$	BLI
NLPerf	2.49	6.23	7.44	7.44	1.42	$\mathbf{8.78}$
MF $D = 10$	2.74 ± 0.08	3.43 ± 0.16	6.47 ± 0.26	6.81 ± 0.19	1.42 ± 0.02	11.24 ± 0.45
MF $D = 20$	2.73 ± 0.07	3.36 ± 0.08	6.33 ± 0.25	6.03 ± 0.10	1.40 ± 0.02	12.05 ± 0.72
MF CTX $D = 10$	2.19 ± 0.07	2.37 ± 0.04	5.45 ± 0.19	3.98 ± 0.07	1.42 ± 0.02	10.92 ± 0.66
MF CTX $D = 20$	$\mathbf{2.15 \pm 0.05}$	$\mathbf{2.25 \pm 0.05}$	5.33 ± 0.15	$\mathbf{3.86 \pm 0.07}$	$\mathbf{1.19 \pm 0.02}$	10.80 ± 0.61
PMF $D = 10$	2.88 ± 0.05	3.27 ± 0.02	11.19 ± 0.08	5.87 ± 0.03	1.44 ± 0.00	11.65 ± 0.37
PMF $D = 20$	2.92 ± 0.04	3.34 ± 0.00	11.25 ± 0.08	5.78 ± 0.05	1.43 ± 0.01	11.83 ± 0.41
BPMF $D = 10$	2.64 ± 0.03	3.77 ± 0.00	5.00 ± 0.06	23.03 ± 3.33	1.61 ± 0.03	9.23 ± 0.11
BPMF $D = 20$	2.63 ± 0.03	2.99 ± 0.04	5.00 ± 0.09	6.44 ± 0.13	1.65 ± 0.07	9.30 ± 0.05
BPMF $D = 30$	2.59 ± 0.06	2.96 ± 0.03	4.98 ± 0.05	6.21 ± 0.1	1.64 ± 0.03	9.07 ± 0.04
BPMF CTX $D = 10$	2.42 ± 0.00	3.69 ± 0.06	4.97 ± 0.05	17.56 ± 1.77	1.53 ± 0.02	9.18 ± 0.20
BPMF CTX $D = 20$	2.35 ± 0.11	2.92 ± 0.05	$\mathbf{4.87 \pm 0.02}$	6.17 ± 0.10	1.57 ± 0.04	9.20 ± 0.07
BPMF CTX $D = 30$	2.51 ± 0.02	2.96 ± 0.03	5.15 ± 0.19	6.02 ± 0.10	1.53 ± 0.02	9.22 ± 0.04

Table 2: RMSE ± standard deviation for various all tasks considered. D: Latent factor dimension. CTX: Approaches incorporating context features.

et al., 2017). Further information regarding each feature can be found in Xia et al. (2020).

All features were normalized using their corresponding z-score. For each task, (except BLI), we predict scores for a single model. This means that there is exactly one underlying statistical or neural model considered for each task. For BLI however, our dataset consists of two metrics *Vecmap* and *Muse*. The performance is obtained as in the equivalent multi-model scenario in Xia et al. (2020), by averaging over both metrics.

The *NLPerf* framework is based on gradient boosting trees (Friedman, 2001) and implemented with XGBoost (Chen and Guestrin, 2016). The parameterisation of this predictor was kept as in (Xia et al., 2020), assuming a squared error as the objective function for the regression, a fixed learning rate of 0.1, a maximum tree depth of 10 and the number of trees being 100. The default setting was used for the regularization terms. We evaluate the performance of our methods by using the root mean squared error (RMSE).

Predictions for MF are obtained according to Equation (8) but through omitting the context information, whereas MF CTX makes use of the latter. The conditional distribution over observed ratings as given in Equation (2) is used to obtain predictions for BPMF, and adapted accordingly for PMF. All non-Bayesian models are trained using nested 5-fold cross validation and stochastic gradient descent (SGD) over 2000 training iterations. Referring to Figure 1, note that during training of the latent vectors $\mathbf{w}_s$ and $\mathbf{h}_t$, all available scores in the corresponding column s and row t are used. The regularization parameters are chosen using a validation set. All non-probabilistic experiments are averaged over 10 runs. Training of the probabilistic models was performed using MCMC sampling and the results are averaged over 2 runs, due to MCMC time complexity.[3] More information about the execution times of each experiment is given in Appendix B.

3.1 Random Train-Test Splits

We start by investigating the behaviour of all methods on a randomly chosen training and test split containing a subset of language pairs. Note that we create one split per task and keep it constant across all methods to allow fair comparison of the obtained results. An example for such a train-test split is depicted in Figure 3 on the left for BLI.

The RMSE and standard deviations of our predicted performances for all tasks are provided in Table 2. For all single model tasks, the MF approaches consistently outperform the state-of-the-art *NLPerf* (Xia et al., 2020). While general MF already outperforms *NLPerf* in all but one setting, using context information (MF CTX) leads to further improvements in terms of RMSE – outperforming the SOTA with reductions in RMSE of up to 64% (Parsing$^{\text{TSF}}$). While the improvement is significant for tasks like EL$^{\text{TSF}}$, POS$^{\text{TSF}}$ and the aforementioned Parsing$^{\text{TSF}}$, we observe a smaller but still noticeable reduction in RMSE for Wiki$^{\text{MT}}$ and MT$^{\text{TSF}}$ of around 13% and 16%, respectively. Note however, that the RMSE scores cannot be directly compared across tasks as the scales of all

---

[3] Code is available at `https://github.com/vschram/PP-via-Bayesian-MF-for-Multilingual-NLP` under the CCBY-SA 4.0 license

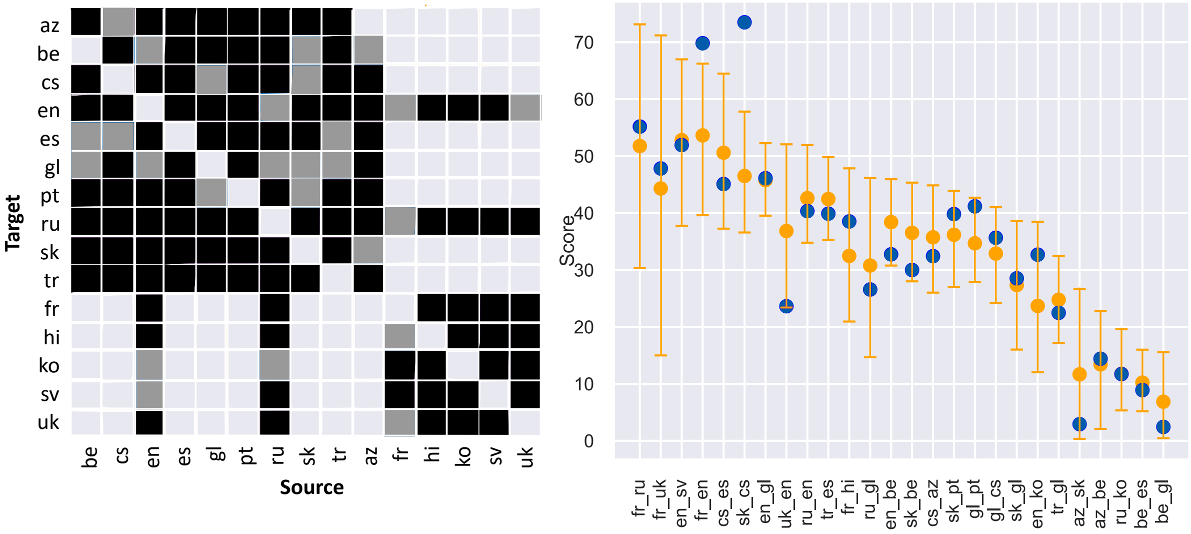


Figure 3: **Left:** One train-test split of the available BLI data set in matrix format. Light grey: Unavailable data. Dark grey: Test data. Black: Training data. **Right:** 95 % Credible intervals (CI) for the predicted scores for the scenario on the left. Orange: Predicted scores. Blue: Actual scores. Method BPMF, dimensionality of latent vectors: 10.

evaluation metrics differ.

While the Bayesian version BPMF CTX also outperforms the *NLPerf* baseline in all but one single model task, we observe particularly good results for the task EL$^{\text{TSF}}$, where BPMF CTX outperforms all other methods and achieves new state-of-the-art results with a relative reduction in RMSE of 35%. As introduced in Section 2, the non-Bayesian PMF requires hyperparameter optimization and as shown in Table 2 for all tasks but MT$^{\text{TSF}}$, BPMF outperforms PMF – potentially due to non-optimal hyperparameter choices for PMF. This might also be the cause for the PMF results in the EL$^{\text{TSF}}$ experiments, while they converge to a solution, the obtained model does not represent the underlying data well. The outliers for POS$^{\text{TSF}}$ are caused by divergences, which can be avoided by using a higher number of latent dimensions. Further research is required in those cases to understand how the models in those cases can be better adapted to the data and is part of future work.

Observing all single model results obtained, we conclude that if a dataset is given for which a matrix can be constructed with a limited number of missing cells, MF approaches provide a strong alternative to gradient boosting trees. Results for all tasks show that linear MF approaches with rather small latent dimension sizes of 10 or 20 show often significant performance improvements over *NLPerf*.

In contrast to the single model tasks, the BPMF approaches achieve competitive performance in the BLI two model scenario, but none of the MF approaches are able to outperform the SOTA. We suspect this might be due to the matrix size and available data, which is significantly smaller than in all other tasks (Table 1). Furthermore, only 6 distance features are available for this setting, indicating that additional features like *dataset size* and *vocabulary size* (among others) that are available for the other tasks might play a important role in facilitating accurate performance prediction.

3.2 Uncertainty of the Predicted Scores

The Bayesian approaches offer the advantage of providing a measure of uncertainty in terms of CIs. An example showing the 95 % CIs for all language-pairs available in the test set is depicted in Figure 3. Predicted performance scores for cells where only limited training data in the corresponding rows and columns is available show wide approximations of the predictive posterior distribution in terms of CIs, e.g. the language pairs "fr_uk" or "fr_hi". In contrast, those with more training data lead to narrower CIs like for "be_es" or "en_be".

Apart from having a direct measure of uncertainty, one can use upper confidence bounds (UCB) and lower confidence bounds (LCB) of the obtained CIs to guide further decision-making (Auer, 2002). Maximum UCBs can help to determine language-pairs for which the underlying model could be especially high-performing, justifying the cost to

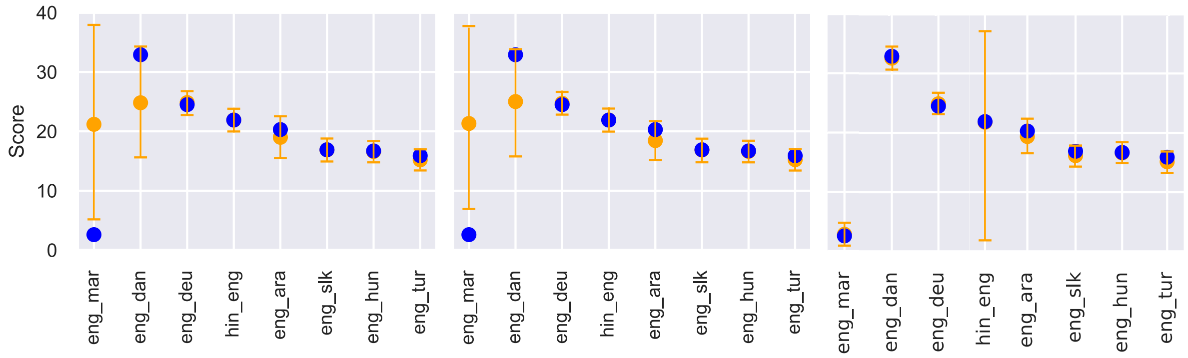


Figure 4: Credible intervals for the task: $\mathrm{MT}^{\mathrm{Wiki}}$. Orange: Predicted Score. Blue: Actual scores. **Left:** BPMF, latent factor dimensionality: 10. **Middle:** BPMF, latent factor dimensionality: 20. **Right:** BPMF CTX, latent factor dimensionality: 20. An increase in latent factor dimensionality and adding additional features leads to narrower CIs. However, adding features to the language pair “hin_eng” causes a wider CI, indicating that the used features might be unsuitable for this language-pair.

train and acquire the necessary datasets for such language-pairs. The example given in Figure 3 on the right shows CIs for the BLI task. Using the UCBs, the language-pairs “fr_ru”, “fr_uk” and “en_sv” could potentially provide us with the highest achievable scores (between 65 and 75). In contrast, using the minimum UCBs, we can conclude that only very low possible maximum scores for the language pairs “ru_ko”, “be_es” and “be_gl” can be achieved. Those will be between 15 and 20, therefore it is discouraged to train and test the underlying models on those. Moreover, the decision whether a model is suited for a language-pair can also be based on the maximum LCBs, indicating the potential that a performance score will be higher or equal then the obtained bounds. For the considered task, this is true for the language pairs “fr_en”, “en_gl” and “en_sv”.

Furthermore, CIs provide important additional insights that help to interpret design choices. Figure 4 shows the impact and importance that an increase in latent factor dimensionality as well as the addition of additional features has on the performance of the task $\mathrm{MT}^{\mathrm{Wiki}}$. The left figure shows predictions and corresponding CIs using BPMF for a latent vector dimension of 10. After increasing the dimensionality to 20, the resulting CIs get slightly narrower as shown in the middle figure. Finally on the right, it can be observed that after adding CTX for language pairs like “eng_dan” and “eng_hun”, the CIs get significantly narrower and indicate increased certainty of the predictions. However, the CI for “hin_eng” interestingly becomes wider, indicating that the additionally chosen features for this language pair might be less beneficial and even hurt performance prediction. Further investigation of the feature set shows that “hin_eng” exhibits the highest value of the feature *average sentence length of the source language*, leading to the conclusion that extreme feature values lead to more uncertainty in the predictions, which is to be expected as the Bayesian model has not encountered this feature value during training.

It is clear from Figures 3 and 4 that the credible intervals are imperfect, not always including the test results. This is to be expected 5% of the time by chance, following the definition of the credible interval. However, deviations can arise through modelling biases. Consider the language pairs “fr-en” and “sk-cs” in Figure 3. The scores for these instances are not seen during training, and thus generalisation errors may occur. For the former, there is lack of training data for a translation from French to other languages. For the latter, although both languages are closely related, there is not enough training data support confident predictions. Further-

	LOLO transfer	LOLO target
NLPerf	**8.08**	10.62
MF	13.29 ± 0.01	11.86 ± 0.02
MF CTX	11.16 ± 0.02	**6.78 ± 0.02**
Bayes	13.09 ± 0.00	10.85 ± 0.00
Bayes CTX	13.04 ± 0.00	10.68 ± 0.00

Table 3: RMSE ± standard deviation for $\mathrm{Parsing}^{\mathrm{TSF}}$. Experiments leaving-one-language-out (LOLO). Dimensionality of latent factors: 20.

more, BPMF does not use any additional features. As visible from Figure 4 for the language pair "eng-mar", adding context features provides additional information and greatly improves the accuracy of the Bayesian credible interval.

NLPerf	8.78
LR	17.26
BPMF CTX $D = 20$	9.20 ± 0.07

Table 4: Contrasting the performance of predictors, being NLPerf, linear regression (LR) and BPMF CTX.

3.3 Leave-one-Language-out Scenarios

We further investigate the importance of additional features for performance prediction in cold start scenarios. In such settings, a new language occurs in the test set which has not been encountered during training. This is referred as leave-one-language-out (LOLO). We conducted the experiment for Parsing$^{\text{TSF}}$ as this includes a complete matrix with the same target and transfer languages as rows and columns. This provides us with the opportunity to contrast the performance of the predictors regarding having the same target and transfer languages.

The vectors of the latent factor models of MF and MF CTX are initialized to small random values during training. Thereby, the latent vector corresponding to the missing language will not be further changed as there is no training data available. Note that MF leverages additionally the information of additional bias as shown in Equation (8). In the Bayesian setting, the latent vectors are sampled according to the probabilistic model from the specified priors. Further training details are kept the same as in Section 3.1.

The results are shown in Table 3. Our experiments demonstrate that when leaving one target language out, the predicted performance of MF CTX significantly outperforms all other approaches. However, when leaving one transfer language out, although the performance scores predicted by MF CTX are better then other MF approaches, they do not outperform *NLPerf*. These results indicate that the performance of a predictor based on the MF framework seems to heavily depend on the missing LOLO direction, being target or transfer – rendering MF CTX well suited for cold start predictions when performance scores for unseen target languages are of interest.

3.4 On the Effects of Non-Linearity

In Table 4 the performance in terms of RMSE of XGBoost, linear regression (LR) and BPMF CTX for BLI is shown. The proposed MF methods were not able to outperform the SOTA in this scenario. However, while *NLPerf* is a non-linear method, our approaches are all bilinear. Comparing to the performance of LR, BPMF CTX clearly outperforms, having the additional advantage of being able to provide measures of uncertainty. This suggests that further investigations are necessary in whether a non-linear model would be better suited for the prediction of performance scores for BLI.

4 Related Work

Matrix completion (MC) is an important technique used to recover a matrix from a subset of its entries, widely applied and studied in many areas of research like machine learning, data science, signal processing and communications (Du and Swamy, 2013; Yuchi et al., 2022). In NLP, it is found in keyword searches or recommender systems, among others (Chen et al., 2018b,a). MC in the context of recommender systems can be interpreted as e.g. the task of predicting a product to recommend to costumers. In the famous *Netflix price competition*, targeting recommender systems for movies, it has been shown that an approach based on matrix factorization (MF) via low-rank latent factor models is the basis for the best algorithm to predict user ratings (Koren et al., 2009; Chen et al., 2018b).

There are two main directions of performance prediction in machine learning. The performance can either be predicted as a function of certain training dataset properties or as a function of its training time or number of iterations (Kolachina et al., 2012). Following the former, current state of the art approaches like *NLPerf* (Xia et al., 2020) or *LITMUS* (Srinivasan et al., 2022) leverage the advantages of regression models like gradient boosting trees. We use *NLPerf* as a competitive baseline in this work. Note that *NLPerf* was shown to outperform several simple mean-value baselines, namely the average over the performances of all available test instances; the average over shared source languages; and the average over models if multiple models are used. This approach can easily incorporate additional features of the experiments, but they do not provide a direct measure of uncertainty.

In Ye et al. (2021) methods used for performance prediction are based on tensor-regression-based approaches being robust PCA and CP decomposition. Those are developed to provide a more fine-grained performance measure. While confidence intervals in the frequentist sense are provided, no Bayesian analysis was conducted. In Srinivasan et al. (2021) collective MF was used for performance prediction of massive multilingual language models, where it underperformed XGBoost. However, no experiments were shown for bilingual tasks. We differ in our work because we explore the framework of MF extensively and are able to provide a range of experiments contrasting probabilistic and non-probabilistic approaches. Additionally our approach can give CIs as a measure of uncertainty of our predictions. Closer to our approach is the work of Elsahar and Gallé (2019), where instead of evaluating domain similarity or diversity, a linear classifier is used to predict the performance drop under domain shift using metrics based on $\mathcal{H}-$divergence, reverse classification accuracy and confidence measures. In contrast to this body of literature, our methods provide the additional benefit of Bayesian uncertainty quantification.

An area of application for performance prediction, to be additionally highlighted here, is data selection, during which the method can be used to identify a suitable training dataset under domain shift, which will lead to the best model performance. Earlier work in this field in e.g. Blitzer et al. (2007) uses the unsupervised $\mathcal{A}-$distance measure of divergence between domains, while existing literature on phrase based machine translation in Axelrod et al. (2011) and Moore and Lewis (2010) show that perplexity- or cross-entropy based scoring methods are beneficial to select suitable sentences for training, increasing the overall model performance. Connected to this stream of literature is also the work of Ruder and Plank (2017). They present a Bayesian optimisation approach, which is model-independent and is used to learn data selection measures for transfer learning. Additionally, incorporating not only information about the data set, but also about the model, Atwell et al. (2022) shows that the $h-$discrepancy, which is a generalisation of the source guided discrepancy (Kuroki et al., 2019), can be used to identify the best generalization performance of discourse models.

5 Conclusion

In this work, we presented an extensive study of various MF methods applied to the problem of performance prediction. Using a Bayesian approach we can give a measure of uncertainty in terms of CIs in addition to point estimates. Additionally, we show that leveraging the obtained bounds of the CIs can guide decision-making regarding whether it is lucrative to deploy a model for certain language-pairs and whether the corresponding datasets for the language-pairs should be acquired.

Our results confirm that bilinear MF methods can be used to predict the performance scores of various NLP tasks, which is computationally less expensive than using a non-linear model like XGboost in *NLPerf*. Furthermore, we show that the MF framework is suited to predict performance scores in cold-start scenarios.

The MF framework was not able to outperform the SOTA for the two-model scenario BLI, which might be due to the small matrix size and unsuitable features. However, BPMF CTX clearly outperformed LR. This suggests that further studies are required regarding non-linear MF methods.

While we chose to present a specific set of tasks within this work, the proposed MF methods can be used for other NLP scenarios where the considered problem can be modelled as a matrix.

Limitations

While a variety of NLP problems can be modelled as a matrix as our investigations in this paper show, this does not apply to all existing NLP tasks and excludes our methods in its current form from being used on datasets, provided for tasks like e.g. universal dependency parsing or morphological analysis as done in Xia et al. (2020). The reason is, that a suitable matrix representation is not possible.

While we were able to show that context features significantly improve the estimation results, we found that language features did not improve the predictions. Therefore, further studies are necessary to understand how language information can be additionally incorporated into the MF framework for performance prediction.

Our Bayesian approach uses MCMC sampling to perform approximate inference, a process that can be rather time consuming and could be replaced by more efficient sampling methods in the future. Furthermore, our methods are currently limited to linear approaches, for a fair comparison with the non-linear model *NLPerf*, non-linear methods needs to be investigated.

Acknowledgements

We thank Tianzhi Li for the code basis provided for the non-probabilistic MF approach. This research was supported by The University of Melbourne's Research Computing Services and the Petascale Campus Initiative.

References

Antonios Anastasopoulos and Graham Neubig. 2019. Should all cross-lingual embeddings speak english? *arXiv preprint arXiv:1911.03058*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. *arXiv preprint arXiv:1805.06297*.

Katherine Atwell, Antony Sicilia, Seong Jae Hwang, and Malihe Alikhani. 2022. The change that matters in discourse parsing: Estimating the impact of domain shift on parser error. *Findings of ACL*, pages 824–845.

Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(11):397–422.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of EMNLP*, pages 355–362.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

John Barnard, Robert McCulloch, and Xiao-Li Meng. 2000. Modeling covariance matrices in terms of standard deviations and correlations, with application to shrinkage. *Statistica Sinica*, pages 1281–1311.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*, pages 440–447.

Ricardo Cabral, Fernando De la Torre, João P Costeira, and Alexandre Bernardino. 2013. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *Proceedings of the IEEE ICCV*, pages 2488–2495.

George Casella and Roger L. Berger. 2021. *Statistical inference*. Cengage Learning.

Baiyu Chen, Zi Yang, and Zhouwang Yang. 2018a. An algorithm for low-rank matrix factorization and its applications. *Neurocomputing*, 275:1012–1020.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the SIGKDD*, pages 785–794.

Wen-Hao Chen, Chin-Chi Hsu, Yi-An Lai, Vincent Liu, Mi-Yen Yeh, and Shou-De Lin. 2018b. Attribute-aware collaborative filtering: survey and classification. *arXiv preprint arXiv:1810.08765*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Ayan Das and Sudeshna Sarkar. 2020. A survey of the model transfer approaches to cross-lingual dependency parsing. *ACM TALLIP*, 19(5):1–60.

Wietse de Vries, Martijn Wieling, and Malvina Nissim. 2022. Make the best of cross-lingual transfer: Evidence from POS tagging with over 100 languages. In *Proceedings of the ACL (Volume 1: Long Papers)*, pages 7676–7685.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Ke-Lin Du and Madisetti NS Swamy. 2013. *Neural networks and statistical learning*. Springer Science & Business Media.

Hady Elsahar and Matthias Gallé. 2019. To annotate or not? predicting performance drop under domain shift. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2163–2173.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Barry Haddow, Rachel Bawden, Antonio Valerio Miceli Barone, Jindřich Helcl, and Alexandra Birch. 2022. Survey of low-resource machine translation. *Computational Linguistics*, pages 1–67.

Luiz Hespanhol, Caio Sain Vallio, Lucíola Menezes Costa, and Bruno T Saragiotto. 2019. Understanding and interpreting confidence and credible intervals around effect estimates. *Brazilian Journal of Physical Therapy*, 23(4):290–301.

Prasanth Kolachina, Nicola Cancedda, Marc Dymetman, and Sriram Venkatapathy. 2012. Prediction of learning curves in machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22–30.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Seiichi Kuroki, Nontawat Charoenphakdee, Han Bao, Junya Honda, Issei Sato, and Masashi Sugiyama. 2019. Unsupervised domain adaptation based on source-guided discrepancy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4122–4129.

Daniel Lewandowski, Dorota Kurowicka, and Harry Joe. 2009. Generating random correlation matrices based on vines and extended onion method. *Journal of multivariate analysis*, 100(9):1989–2001.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, et al. 2019. Choosing transfer languages for cross-lingual learning. *arXiv preprint arXiv:1905.12688*.

Patrick Littell, David R Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. Uriel and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL) (Volume 2: Short Papers)*, pages 8–14.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354*.

Andriy Mnih and Russ R Salakhutdinov. 2007. Probabilistic matrix factorization. *Advances in neural information processing systems*, 20.

Robert C Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 220–224.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig. 2018a. When and why are pre-trained word embeddings useful for neural machine translation? *arXiv preprint arXiv:1804.06323*.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Janani Padmanabhan, and Graham Neubig. 2018b. When and why are pre-trained word embeddings useful for neural machine translation? *arXiv preprint arXiv:1804.06323*.

Kenneth Rice and Lingbo Ye. 2022. Expressing regret: a unified view of credible intervals. *The American Statistician*, pages 1–9.

Shruti Rijhwani, Jiateng Xie, Graham Neubig, and Jaime Carbonell. 2019. Zero-shot neural transfer for cross-lingual entity linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6924–6931.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with Bayesian optimization. *arXiv preprint arXiv:1707.05246*.

Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 880–887.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1351–1361.

Adrian Seyboldt. 2019. Uses of LKJCholeskyCov and LKJCorr. `https://discourse.pymc.io/t/uses-of-lkjcholeskycov-and-lkjcorr/3629/2`. [Online; accessed 19-October-2022].

Or Sharir, Barak Peleg, and Yoav Shoham. 2020. The cost of training nlp models: A concise overview. *arXiv preprint arXiv:2004.08900*.

Anirudh Srinivasan, Gauri Kholkar, Rahul Kejriwal, Tanuja Ganu, Sandipan Dandapat, Sunayana Sitaram, Balakrishnan Santhanam, Somak Aditya, Kalika Bali, and Monojit Choudhury. 2022. Litmus predictor: An ai assistant for building reliable, high-performing and fair multilingual nlp systems. In *Thirty-sixth AAAI Conference on Artificial Intelligence*.

Anirudh Srinivasan, Sunayana Sitaram, Tanuja Ganu, Sandipan Dandapat, Kalika Bali, and Monojit Choudhury. 2021. Predicting the performance of multilingual nlp models. *arXiv preprint arXiv:2110.08875*.

The PyMC Development Team. 2018. LKJ Cholesky Covariance Priors for Multivariate Normal Models. `https://docs.pymc.io/en/v3/pymc-examples/examples/case_studies/LKJ.html`. [Online; accessed 19-October-2022].

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. 2020. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8625–8646.

Zihuiwen Ye, Pengfei Liu, Jinlan Fu, and Graham Neubig. 2021. Towards more fine-grained and reliable nlp performance prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Henry Shaowu Yuchi, Simon Mak, and Yao Xie. 2022. Bayesian uncertainty quantification for low-rank matrix completion. *Bayesian Analysis*, 1(1):1–28.

Task	$\mathrm{MT}^{\mathrm{Wiki}}$	$\mathrm{Parsing}^{\mathrm{TSF}}$	$\mathrm{EL}^{\mathrm{TSF}}$	$\mathrm{POS}^{\mathrm{TSF}}$	$\mathrm{MT}^{\mathrm{TSF}}$	BLI
MF $D = 10$	5 min	4 min	2 min	7 min	14 min	1 min
MF $D = 20$	5 min	4 min	2 min	7 min	14 min	1 min
MF CTX $D = 10$	8 min	6 min	4 min	12 min	22 min	2 min
MF CTX $D = 20$	8 min	7 min	4 min	12 min	22 min	2 min
PMF $D = 10$	1 h 31 min	1 h 6 min	48 min	1 h 34 min	1 h 29 min	1 h 58 min
PMF $D = 20$	1 h 29 min	1 h 17 min	54 min	2 h 10 min	2 h 33 min	1 h 53 min
BPMF $D = 10$	7 h 40 min	9 h 54 min	8 h 52 min	10 h 8 min	10 h 10 min	22 h 2 min
BPMF $D = 20$	8 h 11 min	10 h 44 min	10 h 17 min	11 h 30 min	9 h 30 min	17 h 4 min
BPMF $D = 30$	6 h 54 min	10 h 49 min	10 h 25 min	11 h 1 min	9 h 57 min	18 h 50 min
BPMF CTX $D = 10$	9 h 48 min	10 h 33 min	10 h 22 min	11 h 34 min	13 h 27 min	14 h 25 min
BPMF CTX $D = 20$	9 h 56 min	11 h 57 min	10 h 52 min	15 h 22 min	13 h 2 min	15 h 32 min
BPMF CTX $D = 30$	9 h 14 min	12 h 42 min	11 h 31 min	14 h 23 min	13 h 25 min	15 h 38 min

Table 5: Execution time required for one training and inference run on average per run for each experiment on one cpu, without hyperparamter tuning, assuming optimal hyperparameters have been found before. Training MF and MF CTX: 5 fold CV.

A Models Considered in the Experiments

The MT model behind the $\mathrm{MT}^{\mathrm{Wiki}}$ data is trained on aligned Wikipedia data (Schwenk et al., 2021). The training data for translation are WikiMatrix bitexts (Schwenk et al., 2021) containing a mixture of high and low resource languages, mined using an approach based on multilingual sentence embeddings, and tested using NMT (fairseq[4], Transformer) on the TED test set (Qi et al., 2018a) using BLEU performance. The BLI performance dataset is obtained based on the following procedure. Multilingual semi-supervised word embeddings (Conneau et al., 2017), and fully unsupervised cross-lingual word embeddings (Artetxe et al., 2018) for high and low resource languages, which are learnt under a bilingual setting, are tested on BLI using accuracy as the performance measure (Anastasopoulos and Neubig, 2019). For all TSF tasks, the performance scores are obtained using LangRank (Lin et al., 2019), which gives a rank to each transfer language. Therefore, additionally a feature showing the *rank* of each transfer language can be leveraged. In the following a short description about the underlying models is given, a more comprehensive explanation can be found in (Lin et al., 2019). For $\mathrm{MT}^{\mathrm{TSF}}$ the underlying trained model is an attention-based sequence-to-sequence model (Bahdanau et al., 2015). Training is performed on the multilingual TED talk corpus (Qi et al., 2018b). For the $\mathrm{EL}^{\mathrm{TSF}}$ performance scores two character-level LSTM encoders are trained (Rijhwani et al., 2019). The $\mathrm{POS}^{\mathrm{TSF}}$ dataset is obtained through the training of a bi-directional LSTM-CNNs-CRF model (Ma and Hovy, 2016), while the dependency parsers uses a deep biaffine attentional graph-based model (Dozat and Manning, 2016). The diversity of the underlying models, namely being statistical-, neural network- or dictionary based, gives us a better impression about the usability of our predictor.

B On Computational Complexity

Non-Bayesian MF models have linear time complexity in the number of latent factors $O(D)$ per SGD iteration or $O(D|\delta(\mathbf{R})|)$ for one pass of all observed source-language pairs. The Bayesian MF models have cubic complexity on the size of the latent factors and linear on the number of source/target languages. Due to very small sizes of our datasets, computations were possible on a single cpu. The execution times per model per one run of training and inference are given in Table 5. Compared to the Bayesian approaches, the Non-Bayesian MF methods have a short execution time. It increases with the size of the score matrix, e.g. the model for $\mathrm{MT}^{\mathrm{TSF}}$ with a score matrix size of 54×54 has the highest number of cells compared to all other models and takes also the longest time to perform one training and inference run. The times shown can be taken as a reference point. Furthermore, implementing the experiments on GPU would speed up the process further, in our case we report the execution times on CPU for a better comparison. The CPU used for our experiments is Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz. Implementing MCMC sampling using GPU will lead to a further execution time improvement.

---

[4] `https://fairseq.readthedocs.io/en/latest/`

Unified Neural Topic Model via Contrastive Learning and Term Weighting

Sungwon Han[1] **Mingi Shin**[1] **Sungkyu Park**[2*] **Changwook Jung**[1] **Meeyoung Cha**[3,1*]
[1]School of Computing, KAIST, South Korea
[2]Department of AI Convergence, Kangwon National University, South Korea
[3]Data Science Group, Institute for Basic Science (IBS), South Korea
{lion4151, mingi.shin, wookidoki}@kaist.ac.kr, shaun@kangwon.ac.kr, mcha@ibs.re.kr

Abstract

Two types of topic modeling predominate: generative methods that employ probabilistic latent models and clustering methods that identify semantically coherent groups. This paper newly presents UTopic (Unified neural Topic model via contrastive learning and term weighting) that combines the advantages of these two types. UTopic uses contrastive learning and term weighting to learn knowledge from a pretrained language model and discover influential terms from semantically coherent clusters. Experiments show that the generated topics have a high-quality topic-word distribution in terms of topic coherence, outperforming existing baselines across multiple topic coherence measures. We demonstrate how our model can be used as an add-on to existing topic models and improve their performance.

1 Introduction

One of the most common tasks in natural language processing (NLP) is to find cohesive topics in a text corpus. Topic modeling is widely used in various applications, including trend extraction from real-time streams such as social media (Lau et al., 2012; Park et al., 2021) and identification of notable events upon risk (Shin et al., 2020). Two representative lines of work exist: *generative methods* and *clustering-based methods*.

Generative methods follow the assumption of probabilistic latent semantic analysis (hereafter p-LSA) such that every word token in a document is sampled from a mixture of latent topics (Hoffman, 1999). Such methods estimate both the latent topic distribution per document and word distribution per topic from Bag-of-Words (BoW) representation (Blei et al., 2003; Fisher et al., 2020; Miao et al., 2017; Yan et al., 2013). Latent Dirichlet Allocation (LDA), for example, uses a Bayesian model to estimate the latent topic distribution (Blei et al., 2003). Variational Autoencoder (VAE) retrieves latent topics from the topic distribution while also recovering the original input (Srivastava and Sutton, 2017). Recent techniques have employed pretrained language models on top of VAE-based models to improve topic quality (Bianchi et al., 2021a,b). Generative probabilistic models work on the assumption that each topic is a mix of words from a larger set. However, they share a common weakness: the BoW representation only contains word-level co-occurrences and fails to capture each word token's importance in the document's context information. Consequently, the quality of the estimated topic is reliant on the choice of the word set.

Clustering methods, on the other hand, regard topics as semantic clusters discovered over the document or word embedding space (Angelov, 2020; Grootendorst, 2022; Sia et al., 2020). They utilize the knowledge of the pretrained language model (e.g., BERT (Devlin et al., 2019)) to generate high-level summaries of given documents (or words). Then, document (or word) clusters are identified according to the embedding distance via clustering methods, such as DBSCAN or k-means. They do not require word selection and effectively leverage context information for discovering topics. Nevertheless, their core is clustering; hence, they cannot assign a mixture of topics to each document.

This research takes a step further by combining the benefits of the previous methods by including novel considerations: contrastive learning and term weighting. We start with a generative method and add a term weighting scheme that mimics the clustering method. We use a contrastive learning framework to help the language model instill pretrained knowledge and enable the model to focus on influential words. Our model, **UTopic** (Unified neural Topic model via contrastive learning and term weighting), has three stages: (1) identifying semantic clusters, (2) calculating term weights, and (3) estimating the latent topic distribution. In

---

[*] Corresponding Authors

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1802–1817
May 2-6, 2023

Stage 1, input documents with similar meanings are grouped into a single cluster. This procedure uses a pretrained language model, which extracts meaningful embeddings from the input document. Stage 2 computes the term weights for each word to represent the importance of each term, where high weights are given to frequent terms in one semantically coherent cluster (but not in other clusters). The model selects words with top-k term weights as the final word set for BoW representations. In Stage 3, the model estimates the latent topic distribution by reconstructing BoW representations from inputs. Term weights can represent each term's importance, allowing the model to learn a coherent topic. Contrastive learning manages the entire process of instilling pre-trained knowledge from the language model and generating a more distinct topic distribution with a predefined prior.

Experiments show that UTopic outperforms conventional methods in a wide range of scenarios, including human-annotated datasets (Table 1). When compared to SOTA models like CTM (Bianchi et al., 2021a), ClusterTM (Sia et al., 2020), and BERTopic (Grootendorst, 2022), our model consistently achieves the overall highest topic coherence score, while others excel in only one or two metrics (Table 2). In the 20-Newsgroups dataset, for example, the NPMI score increased by 3.65%. Our word set selection scheme (Stages 1&2 in Fig. 1) can be used as a standalone module to improve other models (Table 8). Major contributions are as follows:

- We present a unified method for combining the benefits of two topic modeling approaches (i.e., generative and clustering) into a single framework via term weighting.
- We modify the contrastive objective to discover semantically coherent clusters, which gives distinct and interpretable word sets for each cluster.
- Our model consistently outperforms existing baselines across multiple topic coherence measures and produces topics that align well with human labels.

Codes and implementation details of the model are available at a GitHub repository.[1]

2 Related Works

Topic modeling algorithms can be divided into two major streams: generative and clustering-based.

2.1 Generative approach

Most generative approaches follow the p-LSA assumption such that a set of tokens in each document are independently sampled from a mixture of topics (Hoffman, 1999). LDA (Blei et al., 2003) is a representative example that establishes the prior distribution following the p-LSA assumption. Many variants have been proposed, including Prod-LDA, Neural-LDA, and Hierarchical-LDA (Srivastava and Sutton, 2017; Blei et al., 2010). Some algorithms like NVDM, VTMRL, Wasserstein-LDA, and TAN-NTM use VAE for estimating the latent topic distribution (Miao et al., 2016; Gui et al., 2019; Nan et al., 2019; Panwar et al., 2021), while other algorithms, including DocNADE, use auto-regressive architecture (Larochelle and Lauly, 2012).

Adding a pretrained language model (or word embeddings) to traditional topic models has yielded promising results (Grootendorst, 2022; Liu et al., 2015; Qiang et al., 2017; Sia et al., 2020). Contextualized topic model (CTM), for example, obtains the embedding vector for each sentence using a model like BERT (Devlin et al., 2019) and bundles the embedding and the BoW (bag-of-words) (Bianchi et al., 2021a). The bundled vector based on VAE is then used to reconstruct BoW and uncover latent topics. However, these approaches have common limitations; word-level co-occurrences cannot fully convey words that represent themes. Naturally, the topic quality depends on the vocabulary set used for calculating word co-occurrences (Gui et al., 2019).

2.2 Clustering based approach

Keeping up with the recent achievement on deep representation learning in NLP domain (Devlin et al., 2019; Mikolov et al., 2013a), several works showed that discovering clusters over the sentence/word representation space can be one viable way to represent topics (Angelov, 2020; Grootendorst, 2022; Sia et al., 2020). Unlike generative approaches, they first extract high-level abstract features from the input document. Then, centroid-based clustering approaches are applied to identify dense clusters, which are finally regarded as topics. For example, Top2Vec (Angelov, 2020) utilizes Doc2Vec embedding (Le and Mikolov, 2014) to

---

[1] `https://github.com/mingi-sid/utopic`

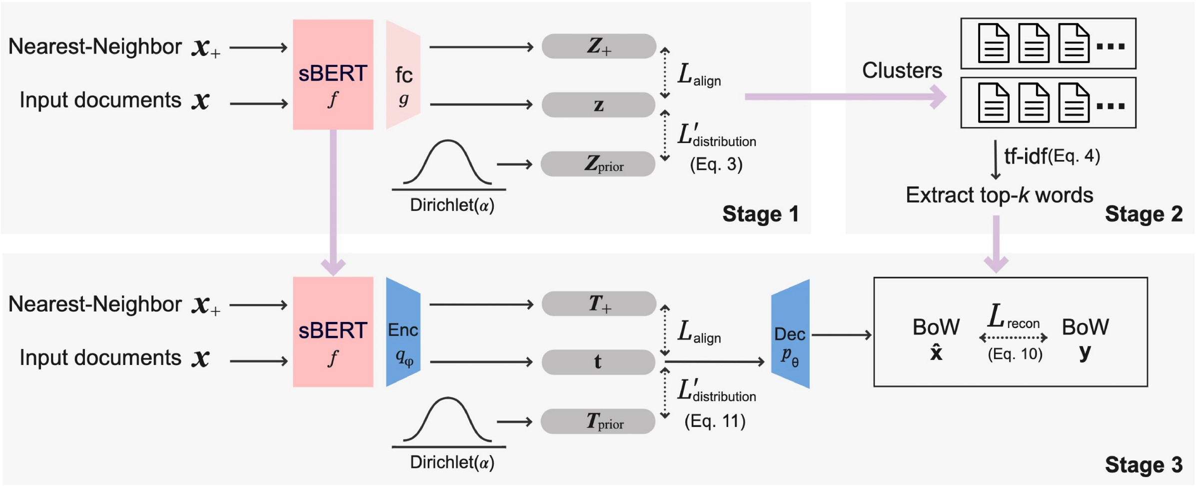


Figure 1: The illustration of the proposed neural topic model, UTopic . It divides documents into clusters based on their context (Stage 1). Then, term weights are computed according to every term's importance. As a topic word set, influential words with the highest term weight are retrieved (Stage 2). UTopic is fine-tuned by estimating the latent topic distribution with the help of term weights and contrastive learning (Stage 3).

jointly embed word and sentence representations, while BERTopic (Grootendorst, 2022) introduces BERT's sentence embedding to identify topic clusters. Another work demonstrates that word-level clusters can represent latent themes from the corpus (Sia et al., 2020). Nevertheless, these methods are clustering-based, so they cannot easily assign multiple topics to each document.

3 Methods

Problem definition: Let $\mathbf{x}$ be an input document and $\mathcal{X}$ be a collection of documents in the corpus (i.e., $\mathbf{x} \in \mathcal{X}$). A primary goal of our topic model is to estimate the topic distribution $\mathbf{p}$ from $\mathcal{X}$. Given a finite number of topics K, we extract top-k relevant words (hereafter referred to as *topic words*) $\mathcal{W}_i$ for each topic i to interpret the underlying theme and evaluate the topic quality. The topic word comes from the word set $\mathcal{W}_{dict}$, which has a size N (i.e., $\mathcal{W}_i \subset \mathcal{W}_{dict}$ for all i).

We propose to compute the *weight importance* of each term on semantically coherent clusters and then estimate the latent topic distribution by reconstructing the BoW representations based on term weights. Our idea is depicted in Figure 1, where a clustering model groups documents with similar contexts in the corpus (Section 3.1). Term weights are computed by treating each document cluster as a single unified document. If certain terms are significant in one group but not in others, we consider them *influential*. Top-k influential subject terms are selected as the word set $\mathcal{W}_{dict}$ for constructing BoW representation (Section 3.2). The model estimates topic distribution using the discovered word set and its term weight to recover the BoW representation of documents (Section 3.3). *Contrastive learning* (hereafter CL) improves representation quality at every phase, and the Dirichlet prior controls the entropy of the topic distribution. These steps are outlined in the following sections.

3.1 Stage 1: Document grouping via CL

Let f be a language model (e.g., BERT) that has been pre-trained, and g be an unsupervised classifier attached on top of f's sentence embedding (e.g., BERT's [CLS] token embedding). First, the models f and g are trained to classify incoming documents into K topic categories. CL facilitates this task with the InfoNCE loss (Oord et al., 2018), which learns an embedding space to maximize agreement between comparable examples while minimizing agreement between different instances. Assume that $\mathbf{z}$ is an input sample's embedding in the form of cluster assignment probability (i.e., $\mathbf{z} = \text{softmax}(g \circ f(\mathbf{x}))$), and that $\mathcal{Z}_+$ and $\mathcal{Z}_-$ are a collection of positive and negative samples' embeddings, respectively. The InfoNCE loss for each sample is defined as follows:

$$L_{\text{CL}}(\mathbf{z}) = -\log \frac{\sum_{\mathbf{z}' \in \mathcal{Z}_+} e^{\text{sim}(z,z')/\tau}}{\sum_{\mathbf{z}' \in (\mathcal{Z}_+ \cup \mathcal{Z}_-)} e^{\text{sim}(z,z')/\tau}} \quad (1)$$

$$= -\log \sum_{\mathbf{z}' \in \mathcal{Z}_+} e^{\text{sim}(z,z')/\tau} + \log \sum_{\mathbf{z}' \in (\mathcal{Z}_+ \cup \mathcal{Z}_-)} e^{\text{sim}(z,z')/\tau}$$

$$= -L_{\text{align}}(\mathbf{z}) + L_{\text{distribution}}(\mathbf{z}), \quad (2)$$

where sim(·) represents the similarity metric between two embeddings, and τ is a temperature value to control the entropy (Hinton et al., 2015). The InfoNCE loss can be broken down into two parts: *alignment loss* and *distribution loss*. The first term, alignment loss (L_{align}), directs the model to place positive pair embeddings closer together. The second term, distribution loss ($L_{\text{distribution}}$), pushes the embeddings of each sample as far apart as possible. This policy requires the model to match each instance's embedding into the predefined prior distribution with high entropy (Chen and Li, 2020; Wang and Isola, 2020).

The contrastive loss is used as follows in document clustering. For a given anchor $\mathbf{x}$, we establish a set of positive samples $\mathcal{Z}_+$ and negative samples $\mathcal{Z}_-$. Given that our model is unsupervised, there is no ground-truth label to determine which samples should be placed closer together in the embedding space. Instead, we mine the pretrained language model's top-1 nearest neighbor in the embedding space and consider it a positive sample (i.e., $\mathbf{x}_+$ in Fig. 1). The remaining instances in the same batch are considered negative samples.

Next, we guide the instance's embedding distribution to follow the prior with low entropy by modifying $L_{\text{distribution}}$ and adopting the optimal transport theory (Peyré et al., 2019; Rabin et al., 2011), a mathematical framework for transporting two sets of points while minimizing the transportation cost. The Sliced Wasserstein Distance (SWD) (Kolouri et al., 2019), a distance measure based on optimum transport that projects embeddings to random orthogonal subspaces and sums the 1-dimension Wasserstein Distance for each subspace, is one of its techniques. We replace $L_{\text{distribution}}$ with the SWD between embedding distribution and predefined low-entropy prior $\mathcal{Z}_{prior}$ (Eq. 3).

The model does not directly minimize entropy because documents can contain numerous topics; instead, it assigns the prior $\mathcal{Z}_{prior}$ to Dirichlet distribution with $\alpha < 1$ to assure low entropy of the embedding space. The alignment loss with adjusting factor λ is used to optimize the modified distribution loss at the same time.

$$\begin{aligned} L_{\text{stage1}}(\mathbf{z}) &= -L_{\text{align}}(\mathbf{z}) + \lambda \cdot L'_{\text{distribution}}(\mathbf{z}), \\ \text{where } L'_{\text{distribution}}(\mathbf{z}) &= \text{SWD}(\mathbf{z}, \mathcal{Z}_{prior}) \end{aligned} \quad (3)$$

We use the cluster assignment probability $\mathbf{z}$ to discover K clusters after training (i.e., $\arg\max_j \mathbf{z}_j$ where j indexes the vector dimension).

3.2 Stage 2: Computing term weights

Given the newly identified cluster set $\mathcal{C}$ (i.e., $c \in \mathcal{C}$), the next stage calculates the term importance weight for each word from $\mathcal{C}$ and constructs the word set $\mathcal{W}_{dict}$. If words frequently appear in one cluster but not in others, they are deemed influential. This concept is consistent with TF-IDF (Ramos et al., 2003), which measures the importance of each word in a document by multiplying two terms: term frequency (TF) and inverse document frequency (IDF)[2]. Similar to (Grootendorst, 2022), we treat each document cluster as a single unified document and calculate the TF-IDF of each word for term weights. The TF-IDF measure for each cluster is as follows:

$$\text{tf-idf}(w, c, \mathcal{C}) = \text{tf}(w, c) \cdot \text{idf}(w, \mathcal{C}) \quad (4)$$

$$\text{tf}(w, c) = \frac{\text{freq}(w, c)}{\sum_{w' \in c} \text{freq}(w', c)} \quad (5)$$

$$\text{idf}(w, \mathcal{C}) = \log \frac{|\mathcal{C}|}{|\{c \in \mathcal{C} : w \in c\}|}, \quad (6)$$

where $\text{freq}(w, c)$ denotes the number of times a given word w is found in cluster c.

To build the word set $\mathcal{W}_{dict}$ for building BoWs in the next stage, we select the top-k influential topic words for each cluster with the highest term weight (Eq. 7). The value of k is adaptively chosen based on the dictionary size $N = |\mathcal{W}_{dict}|$.

$$\mathcal{W}_{dict} = \bigcup_{c \in \mathcal{C}} \{w | \text{ tf-idf}(w, c, \mathcal{C}) \text{ is top-}k \text{ in } c\} \quad (7)$$

3.3 Stage 3: Estimating topics via BoW reconstruction with term weights

The model is then trained to estimate the latent topic distribution in the final stage. We construct an encoder-decoder network on top of the pretrained language model f for training, following the literature (Bianchi et al., 2021a; Srivastava and Sutton, 2017). The encoder and decoder networks are referred to as q_ϕ and p_θ, respectively. The input document $\mathbf{x}$ is decomposed into the BoW representation $\mathbf{y}$ given the word set $\mathcal{W}_{dict}$. To create the context-aware representation, $\mathbf{x}$ is fed into the fixed backbone language model f. The encoder network q_ϕ and softmax function (Eq.8) are used to generate the latent topic distribution $\mathbf{t}$ over $\mathbf{x}$. The decoder

---

[2] A word's relative importance within a document is represented by TF, which divides the word count in a document by the total word count in the corpus. IDF represents the target word's uncommonness across the corpus as the logarithm of the total document count divided by the document count containing the target word.

network, followed by the softmax function, reconstructs the input document's BoW representation as $\hat{\mathbf{x}}$ (Eq. 9) with this topic distribution $\mathbf{t}$.

$$\mathbf{t} = \text{softmax}((q_\phi \circ f)(\mathbf{x})) \tag{8}$$

$$\hat{\mathbf{x}} = \text{softmax}(p_\theta(\mathbf{t})) \tag{9}$$

Our topic model with the backbone network f is trained by reconstructing the original document's BoW representation $\mathbf{y}$ from the latent topic distribution $\mathbf{t}$ (depicted as a box in Figure 1). To focus more on influential words when learning topics, term weights $w_\mathbf{y}$ from the previous stage are multiplied by the loss objective (Eq. 10). This leads the model to filter out unnecessary terms and discover more coherent and human-interpretable topics.

$$L_{\text{recon}}(\mathbf{x}, \mathbf{y}) = -\mathbb{E}_{q_\phi(\mathbf{t}|f(\mathbf{x}))}[w_\mathbf{y} \cdot \log(p_\theta(\mathbf{y}|\mathbf{t}))]. \tag{10}$$

The next step is to improve the representation quality by matching the encoder's posterior distribution to the predefined prior. This approach is modulated by contrastive loss, which assumes that documents with semantically identical content have similar topic distributions. As a positive sample, we use the top-1 nearest neighbor $\mathbf{x}_+$ from the embedding space of the pretrained language model. All other instances in a batch are considered negative samples. The model then uses the alignment loss L_{align} (Eq. 11) to induce the topic distribution of positive pairs ($\mathcal{T}_+$) to be similar. Distribution loss is used to match the posterior topic distribution to the prior distribution $\mathcal{T}_{prior}$ and maintain a low entropy. We set the prior $\mathcal{T}_{prior}$ to Dirichlet distribution with α<1 to discover distinctive topics from the corpus.

$$\begin{aligned} L_{\text{CL}}(\mathbf{t}) &= -L_{\text{align}}(\mathbf{t}) + \lambda \cdot L'_{\text{distribution}}(\mathbf{t}), \\ &\text{where } L'_{\text{distribution}}(\mathbf{t}) = \text{SWD}(\mathbf{t}, \mathcal{T}_{prior}) \end{aligned} \tag{11}$$

The complete loss function is described in (Eq. 12). There are no adjusting weight parameters for objectives to reduce the tuning cost needed for hyper-parameters.

$$\begin{aligned} L_{\text{stage3}}(\mathbf{x}, \mathbf{y}) &= L_{\text{recon}}(\mathbf{x}, \mathbf{y}) + L_{\text{CL}}(\mathbf{t}), \\ &\text{where } \mathbf{t} \text{ is calculated by Eq. 8.} \end{aligned} \tag{12}$$

4 Experiments

4.1 Performance evaluation

Datasets: For evaluation, we employ a variety of datasets with varying topics. 20-Newsgroups[3], Reuters-21578[4], Wikipedia, and BBC[5]. 20-Newsgroups is a collection of 11,314 posts shared in the newsgroups with a balanced set of topic labels. Reuters-21578 is a news post collection with 10,778 training data. We also use the Wikipedia dataset, containing randomly collected 20,000 paragraphs from 2021-09-22. Finally, BBC is a dataset of 2,225 documents excerpted from the BBC news website. All datasets are in English.

Evaluation: NPMI, C_p, and word2vec similarity are used as evaluation criteria. The first two statistics measure the degree of topic coherence between each topic's top-M term (M is set to 10 in our experiments) using a reference corpus. The last one, word2vec similarity, measures the semantic similarity between the top-M terms. The details of each metric are as follows.

- *NPMI*, the Normalized Point-wise Mutual Information, is a metric that scores high if the joint probability of pairs in the top-M words is greater than their marginal probability (Aletras and Stevenson, 2013). The co-occurrence probabilities can be measured from external corpora, such as Wikipedia (*NPMI-Wiki*) or the source data itself (*NPMI-In*).
- C_p is another coherence metric computed by an aggregation of Fitelson's confirmation measure based on the sliding window over the reference corpus. It is known to be one of the most well-correlated coherence scores with human ratings. (Röder et al., 2015).
- *word2vec similarity* evaluates topic coherence using an external embedding model to ensure fair comparisons with the baselines (Ding et al., 2018). The average cosine similarity value is reported across all pairs of top-M words' word2vec embeddings.

We used *Palmetto*, a publicly available tool for measuring topic quality[6] (Röder et al., 2015), and its version of Wikipedia dump to calculate *NPMI-Wiki* and C_p. For word2vec similarity, we used the pre-trained model from Mikolov et al. (2013b).

Implementation details: As the pre-trained language model, we used Sentence-BERT, *all-MiniLM-L6-v2* (Reimers and Gurevych, 2019). The

---

[3] `http://qwone.com/~jason/20Newsgroups/`.

[4] `http://daviddlewis.com/resources/testcollections/reuters21578/`.

[5] `http://mlg.ucd.ie/datasets/bbc.html`.

[6] `https://github.com/dice-group/Palmetto`.

	20-Newsgroups				**Reuters-21578**			
	NPMI-Wiki	NPMI-In	C_p	Word2vec	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0052	0.0652	0.0652	0.1719	-0.0723	0.0708	-0.1496	0.1549
NeuralLDA	-0.0180	-0.0229	-0.0540	0.1847	-0.0373	0.0651	-0.0347	0.1523
ETM	0.0192	**0.0988**	0.1066	0.2217	-0.0506	0.0791	-0.0855	0.1741
CTM	-0.0135	0.0464	-0.0296	0.1755	-0.0465	**0.1314**	-0.0658	0.1656
Top2Vec	0.0248	-0.0279	-1.4809	**0.2646**	-0.0532	0.0466	-0.2426	0.2126
ClusterTM	0.0134	-0.2810	0.0006	**0.2659**	**-0.0101**	-0.1745	-0.0703	**0.3082**
BERTopic	0.0351	-0.0733	0.1555	0.2257	-0.0656	0.0156	-0.2252	0.1759
UTopic	**0.0716**	**0.1016**	**0.3796**	0.2175	**-0.0117**	**0.1179**	**0.0852**	0.2162

	Wikipedia				**BBC**			
	NPMI-Wiki	NPMI-In	C_p	Word2vec	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0071	0.0296	0.0304	0.1458	-0.0721	-0.0186	-0.0711	0.1478
NeuralLDA	0.0133	0.0700	0.0862	0.1729	0.0017	0.0084	0.0562	0.1601
ETM	-0.0294	0.0319	0.0543	0.1484	-0.0270	0.0313	0.0643	0.1949
CTM	0.0610	**0.1547**	**0.2261**	0.1962	0.0339	**0.0991**	0.2622	0.1746
Top2Vec	-0.0215	-0.0876	-0.2111	0.1768	(0.0463)	(0.0124)	(0.1625)	(0.2495)
ClusterTM	0.0567	-0.3212	0.1835	**0.3398**	0.0243	0.0873	0.0739	**0.2959**
BERTopic	**0.0682**	-0.0295	0.2010	0.2196	(0.0225)	(0.0853)	(0.1652)	(0.1842)
UTopic	**0.0702**	**0.1129**	**0.2565**	0.2171	**0.0848**	**0.1057**	**0.4587**	0.2199

Table 1: Performance comparison over four datasets. Results are averaged over five trials from three different number of topics (K=10,20,50). The best or comparable performances are highlighted. In the table, parentheses indicate that the model failed for some topic counts; hence, only the success cases' results are averaged.

Method	NPMI-Wiki	NPMI-In	C_p	Word2vec	Total
LDA	7.0	5.0	6.3	7.8	6.5
NeuralLDA	5.5	5.0	5.3	6.8	5.6
ETM	6.0	3.5	5.0	5.0	4.9
CTM	4.3	2.0	3.3	5.8	3.8
Top2Vec	4.5	6.3	7.0	3.0	5.2
ClusterTM	3.5	6.8	4.5	**1.0**	3.9
BERTopic	4.0	6.0	3.8	3.5	4.3
UTopic	**1.3**	**1.5**	**1.0**	3.3	**1.8**

Table 2: Performance comparison based on the averaged rank for each evaluation metric across four datasets.

topic count K was set to 10, 20, and 50, while the corresponding Dirichlet prior α was set to 0.1, 0.05, and 0.02 (i.e., 1/# of topics), respectively[7]. λ was set to 1, and the vocabulary size to 2,000 following the literature (Bianchi et al., 2021a).

In stage 1, a single layer perceptron, g, is added to the Sentence-BERT model and trained for 100 epochs using the RangerLars (RAdam + LARS + Lookahead) optimizer with an initial learning rate of 0.001 and a decay factor of 0.99. Input text that exceeds the maximum sequence length is truncated. In stage 3, an encoder, q_ϕ, and a decoder, p_θ, are trained for 50 epochs using the Adam optimizer with a learning rate of 2E-2. The encoder is a three-layer MLP with dropout and batch normalization, and the decoder is a one-layer linear model without bias. The backbone network, f, is fixed during this stage.

	20-Newsgroups		**BBC**	
	Match Acc.	NMI	Match Acc.	NMI
LDA	0.258	0.270	0.432	0.170
NeuralLDA	0.180	0.157	0.473	0.347
ETM	0.340	0.349	0.856	0.580
CTM	0.276	0.274	0.850	0.653
Top2Vec	0.264	0.260	0.636	0.520
ClusterTM	0.183	0.162	0.381	0.158
BERTopic	0.317	0.376	0.568	0.360
UTopic	**0.509**	**0.454**	**0.898**	**0.692**

Table 3: Performance comparison on topic assignment quality. The proposed model shows superb results.

Model	LDA	NeuralLDA	ETM	CTM	**Ours**
Match Acc.	0.112	0.098	0.204	0.298	**0.351**

Table 4: Performance on assigning multiple topics to the newly synthesized 20-NewsGroup dataset.

Baselines: We used seven baselines for comparison. The first four are generative methods: (*1*) LDA (Blei et al., 2003), (*2*) NeuralLDA (Srivastava and Sutton, 2017), which is an LDA implementation with Autoencoding Variational

---

[7] We set the topic range to 10∼50 because clustering-based baselines (e.g., HDBSCAN in BERTopic or Top2Vec) failed when the topic size exceeded 70.

Setup	NPMI-Wiki	NPMI-In	C_p	Word2vec	Label acc.
UTopic	**.0653**	**.1211**	**.3629**	.2143	**.5087**
V1 (only stage 3)	.0409	.1052	.3144	.2222	.4539
V2 (no stage 3)	.0639	.1082	.3286	.2292	.3607
V3 (no L_{align} in L_{stage3})	.0485	.0920	.3020	.2275	.4974
V4 (no SWD loss in L_{stage3})	.0546	.1023	.3192	.2027	.4515
V5 (no term weight w_{y} in L_{stage3})	.0638	.1065	.3609	.2129	.4733
V6 (only L_{recon} in L_{stage3})	.0169	.0561	.2190	**.2327**	.4926
V7 (only L_{recon} without term weight w_{y} in L_{stage3})	.0082	.0465	.2149	.2290	.4834

Table 5: The ablation study results upon the overall architecture on 20-Newsgroup (K=20) confirm the substantial contribution of every model component we designed (V stands for *version*). The best results are highlighted.

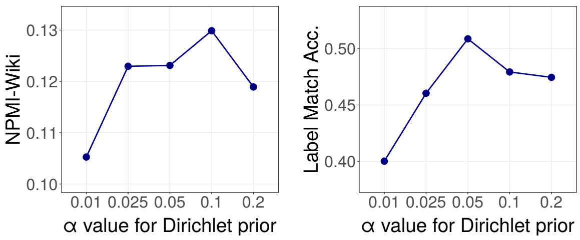
	NPMI-Wiki	NPMI-In	C_p
Ours (AE)	**0.0653**	**0.1231**	**0.3709**
V8 (VAE)	0.0611	0.0943	0.3562

Table 6: Ablation study results on the model architecture (AE vs. VAE): 20-Newsgroups dataset is used.

Inference for Topic Models (AVITM), (*3*) ETM (Embedded Topic Model) (Dieng et al., 2020), where the likelihood of a word is produced by the dot product between the word and topic embeddings, and (*4*) CTM (Contextualized Topic Model), where a vector representation of the document from SentenceBERT is additionally encoded on top of ProdLDA (Bianchi et al., 2021a). The remaining three are clustering methods: (*5*) Top2Vec (Angelov, 2020), where Doc2Vec is used for jointly embedding sentences and words to discover topic clusters, (*6*) ClusterTM (Sia et al., 2020), a word-level clustering-based approach, and (*7*) BERTopic (Grootendorst, 2022), which applies BERT's sentence embeddings to identify topics.

We applied common preprocessing procedures, such as removing English stop-words based on *NLTK* (Bird et al., 2009), non-alphabetic words, and words with less than three characters, and lemmatizing all tokens using the WordNet lemmatizer. All baselines used the same standard parameter values, such as topic count and vocabulary size, to ensure a fair comparison.

Results: Tables 1 and 2 show the performance comparisons and their summaries. UTopic provides the best or comparable topic coherence scores against other baselines over all datasets. Our technique consistently achieves satisfactory results in all other measures, while generative methods perform poorly on Word2vec similarity and clustering methods often fail on NPMI measures.



(a) Effect on NPMI-Wiki (b) Effect on Label Acc.

Figure 2: The effect of the hyper-parameter α for Dirichlet prior on two evaluation metrics on 20-Newsgroups. Consistent performance is achieved by keeping α within a suitable range (0.025 $\sim$ 0.2).

Our model also shows the best scores based on the average rank of each evaluation metric.

Topic assignment quality analysis: It is important to check whether the discovered topics from each model are well aligned with the actual ground-truth labels. Two additional evaluation metrics were observed: *Label Matching Accuracy (Match Acc.)* and *Normalized Mutual Information (NMI)*. Given the estimated topic from each document, the Hungarian method (Kuhn, 1955) was applied to obtain the best bijection permutation mapping between the estimated topics and labels. Then, the label matching accuracy is computed with top-1 classification accuracy. The NMI measure is the mutual information between the mapping and labels with normalization to 0$\sim$1 range.

Table 3 reports the comparison results among baselines over the 20-Newsgroups and BBC-News dataset because only these datasets contain human-annotated labels. Our model outperformed all baselines in accuracy by a large margin, even when compared with recent approaches, implying that it can generate more human-interpretable topics.

Evaluation for assigning multiple topics: We next conducted an experiment to assess UTopic 's ability to assign multiple topics. We created a synthetic dataset with multiple topic labels by merging two random documents from 20-Newsgroups. Then, each topic model trained only using the original 20-Newsgroups dataset was evaluated using the newly created synthetic test set. The evaluation compares the accuracy of counting whether the top-2 estimated topics match the ground truth. Here, random selection will have an accuracy of 0.1 (=2/20). Table 4 shows that our model again outperformed baselines on this new test. Clustering-based approaches were excluded from the comparison because they are limited in assigning multiple topics.

4.2 Component analysis

Ablation studies iteratively remove one module or component to assess its unique contribution to the overall model. We now present results from the ablation study and hyper-parameters' effect over the 20-Newsgroup dataset.

Ablation study: The proposed model has three stages: document clustering, computing term weights, and estimating latent topic distribution. Our first ablation is a model with only stage 3 by directly estimating the latent topic distribution on top of the pretrained language model (*V1* in Table 5). The model without stage 3 utilizes the discovered clusters from the first stage as a topic embedding (*V2*). In this ablation, topic words extracted from the second stage are used for evaluating topic coherence. The next set of ablations remove each module in stage 3 to assess its effect (*V3*–*V7*). Table 5 reports the results for each ablation with five evaluation metrics: four topic coherence measures and label matching accuracy with human-annotated ground truths. The model with all components achieves the best topic coherence with label accuracy, indicating that each component plays an important role.

In contrast to previous works based on VAE (Bianchi et al., 2021a; Srivastava and Sutton, 2017), we used Autoencoder to enforce that the final topic distribution follows the predefined prior. To validate our design choice, we consider another ablation that uses the VAE architecture (*V8*). Experiments on 20 newsgroups show that using VAE can have a negative impact on overall performance (Table 6).

Hyper-parameter analysis: We study the effect of adjusting hyper-parameter α and λ in terms of various evaluation metrics. The α controls the density of Dirichlet prior to the match with the model's latent topic distribution. The smaller the α<1, the less the topic distribution overlaps. The results are summarized in Figure 2. Our model delivers favorable results for both measures when α is within a suitable range of 0.025$\sim$0.2. However, setting α to a value that is too low can result in poor score. This is because decreasing entropy to an excessive degree can cause the model to learn incoherent topics and have a heterogeneous word distribution. We set α to $1/K$ (i.e., 0.05 for K=20), which is a common value in LDA (Rehurek and Sojka, 2011).

λ	NPMI-Wiki	NPMI-In	C_p	Word2vec
0.1	0.055	0.105	0.364	0.210
0.2	0.058	0.102	0.354	0.206
0.5	0.061	0.102	0.367	0.208
1	**0.072**	0.102	0.380	**0.218**
2	**0.072**	0.103	0.376	0.215
5	0.066	**0.112**	**0.382**	0.211
10	0.060	0.109	0.362	0.202

Table 7: Hyper-parameter analysis on the loss adjusting factor λ. Results are based on 20-Newsgroups.

Table 7 reports additional results on the effect of λ, the ratio between the alignment loss and the distribution loss (Eq. 11). Our model performs stable within a reasonable lambda range for most metrics. One exception is NPMI-Wiki, where the model appears influenced by extreme lambda values (λ=0.1 or 10).

4.3 Qualitative analysis

Cluster visualization: To understand the method's inner workings, we visualize the cluster assignment snapshots on the 20-Newsgroups dataset after stage 1 over four different training epochs in Figure 3. Ground-truth labels for 20 themes are displayed in different colors. More visually separable class-coherent clusters emerge as training advances. Evaluating the clustering performance via the Hungarian approach (Kuhn, 1955), epoch 30 in stage 1 already reaches a prediction accuracy of 48% without using any labels.

Application: The suggested topic word selection approach is a stand-alone module that can be used in various topic models. For example, we created the vocabulary set W_{dict} by extracting 2,000 topic words from our stage-2 model. Then we used W_{dict} to train three existing topic models: LDA, NeuralLDA, and CTM, and compared them with a traditional technique of employing the most

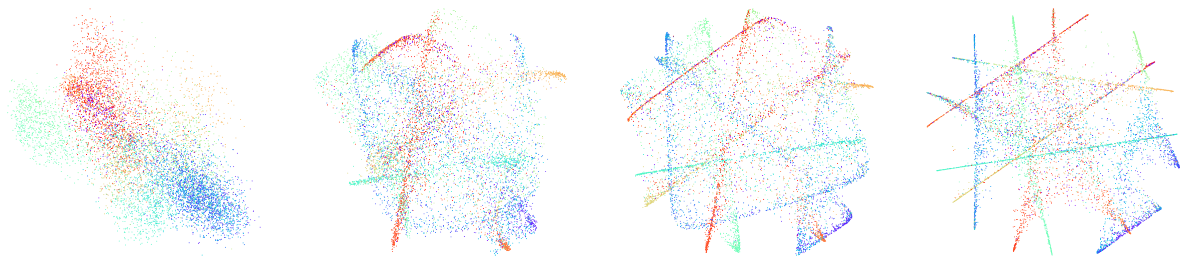

(a) Epoch=0, p=0.30 (b) Epoch=2, p=0.41 (c) Epoch=5, p=0.45 (d) Epoch=30, p=0.48

Figure 3: The 20-Newsgroups dataset was used to visualize intermediate clustering results at Stage 1 of UTopic . Colors represent the ground-truth class names (i.e., one of the 20 news topics), whereas dots reflect document cluster assignment. The p-value indicates that the model has already achieved a clustering accuracy of 48% after 30 epochs.

Setup	NPMI-Wiki	NPMI-Internal
LDA	-0.0056	0.0661
+ Our method	**-0.0016**	**0.0841**
ETM	0.0234	**0.0927**
+ Our method	**0.0404**	0.0752
CTM	-0.0086	0.1149
+ Our method	**-0.0021**	**0.1208**

Table 8: Performance improvement with our topic word extraction strategy on existing topic models on the 20-Newsgroups dataset (K=20).

frequent 2,000 words after ignoring stop-words. For all models, we limited the number of topics to 20. The findings are provided in Table 8 for the 20-Newsgroups dataset, which shows that using our context-aware topic word selection technique improves NPMI-Wiki by about 0.7∼1.7 percent point for all evaluated models.

5 Conclusion

This work proposed a new way to leverage the benefits of combining generative and clustering methods of topic modeling into a single framework. Diverse topic coherence measures have been used to assess the quality of topics. Compared to baselines that excel in only one or two coherence measures, UTopic showed consistent improvement across multiple coherence measures and discovered topics that align well with human annotations. Our method has a wide range of applicability because it can be added as a module to other existing approaches.

Limitations

This work comes with several limitations. First, because the nearest neighbor in the embedding space of the pretrained language model is considered a positive sample, the pretrained knowledge can impact overall performance. We plan to develop advanced text data augmentation and positive sample selection approaches for topic modeling to overcome this limitation. Second, since our clustering step needs a large batch (more than 128), the proposed method may be unattainable in some real-world scenarios due to memory constraints. To reduce this memory cost, we introduce a dynamic queue to save many samples' embeddings for every iteration (see Appendix for further details). However, training is only required once per dataset and is, therefore, acceptable.

Ethical Consideration

We acknowledge that presenting a data summary using topic modeling may not adequately represent the voice of minorities and that pre-trained knowledge from an online corpus may exacerbate such bias. When applying topic modeling in high-risk real-world scenarios, human annotators can help understand the predominant view of each topic to ensure that minor viewpoints are included. This is part of the larger challenge of AI ethics, and we plan to consider this challenge in the future.

Acknowledgement

This research was supported by the National Research Foundation of Korea (RS-2022-00165347), the Institute for Basic Science (IBS-R029-C2), and the 2022 Research Grant from Kangwon National University in South Korea.

References

Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *proc. of the International Conference on Computational Semantics (IWCS)*, pages 13–22.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2021a. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. In *proc. of the Association for Computational Linguistics (ACL)*, pages 759–766.

Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza, and Elisabetta Fersini. 2021b. Cross-lingual contextualized topic models with zero-shot learning. In *proc. of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1676–1683.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

David M Blei, Thomas L Griffiths, and Michael I Jordan. 2010. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM (JACM)*, 57(2):1–30.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Ting Chen and Lala Li. 2020. Intriguing properties of contrastive losses. *arXiv preprint arXiv:2011.02803*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *proc. of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Ran Ding, Ramesh Nallapati, and Bing Xiang. 2018. Coherence-aware neural topic modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 830–836.

Dan Fisher, Mark Kozdoba, and Shie Mannor. 2020. Topic modeling via full dependence mixtures. In *proc. of the International Conference on Machine Learning (ICML)*, pages 3188–3198.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Lin Gui, Jia Leng, Gabriele Pergola, Yu Zhou, Ruifeng Xu, and Yulan He. 2019. Neural topic model with reinforcement learning. In *proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3478–3483.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *stat*, 1050:9.

Thomas Hoffman. 1999. Probabilistic latent semantic analysis. In *proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. 2019. Generalized sliced wasserstein distances. *Advances in Neural Information Processing Systems*, 32:261–272.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Hugo Larochelle and Stanislas Lauly. 2012. A neural autoregressive topic model. *Advances in Neural Information Processing Systems*, 25:2708–2716.

Jey Han Lau, Nigel Collier, and Timothy Baldwin. 2012. On-line trend analysis with topic models:# twitter trends detection topic model online. In *proc. of the International Conference on Computational Linguistics (COLING)*, pages 1519–1534.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *proc. of the AAAI Conference on Artificial Intelligence (AAAI)*.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering discrete latent topics with neural variational inference. In *proc. of the International Conference on Machine Learning (ICML)*, pages 2410–2419.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *proc. of the International Conference on Machine Learning*, pages 1727–1736.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems (NeuIPS)*, 26.

Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. 2019. Topic modeling with wasserstein autoencoders. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6345–6381.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Madhur Panwar, Shashank Shailabh, Milan Aggarwal, and Balaji Krishnamurthy. 2021. TAN-NTM: Topic attention networks for neural topic modeling. In *proc. of the Association for Computational Linguistics (ACL))*, pages 3865–3880.

Sungkyu Park, Sungwon Han, Jeongwook Kim, Mir Majid Molaie, Hoang Dieu Vu, Karandeep Singh, Jiyoung Han, Wonjae Lee, and Meeyoung Cha. 2021. Covid-19 discourse on twitter in four asian countries: Case study of risk communication. *Journal of Medical Internet Research*, 23(3):e23272.

Gabriel Peyré, Marco Cuturi, et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607.

Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. 2017. Topic modeling over short texts by incorporating word embeddings. In *proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 363–374. Springer.

Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. 2011. Wasserstein barycenter and its application to texture mixing. In *proc. of the International Conference on Scale Space and Variational Methods in Computer Vision (SSVM)*, pages 435–446.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *proc. of the International Conference on Machine Learning (ICML)*, volume 242, pages 29–48.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3982–3992.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *proc. of the ACM International Conference on Web Search and Data Mining (WSDM)*, pages 399–408.

Mingi Shin, Sungwon Han, Sungkyu Park, and Meeyoung Cha. 2020. A risk communication event detection model via contrastive learning. In *proc. of the NLP4IF Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 39–43.

Suzanna Sia, Ayush Dalmia, and Sabrina J Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! In *proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1728–1736.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations (ICLR)*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *proc. of the International Conference on Machine Learning (ICML)*, pages 9929–9939.

Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *proc. of the Web Conference (WWW)*.

A Appendix

A.1 Release

Codes and implementation details of the model are available at `https://github.com/mingi-sid/utopic`.

A.2 Dataset details

For evaluation, we employ a variety of datasets with varying topics: (1) 20-Newsgroups[8] contains 11,314 posts from the newsgroups from 20 number of balanced topic labels, (2) Reuters-21578[9] includes 10,778 news posts from Reuters newswire in 1987, (3) Wikipedia, comprising 20,000 abstracts that have more than 200 characters, was randomly sampled from the Wikipedia dump from 2021-09-22, and (4) NeurIPS[10] has 7,241 titles of articles from the NeurIPS conference. Since the topic modeling algorithm is an unsupervised approach, we use the whole dataset for evaluation without splitting.

A.3 Implementation details

For the pretrained language model f, we use Sentence-BERT, *all-MiniLM-L6-v2* specifically (Reimers and Gurevych, 2019). In stage 1, a fully connected layer g with a single layer perceptron is appended on top of the [CLS] representation from the Sentence-BERT. The total number of parameters is about 22.8M, including the parameters from the backbone language model. The original text is used as input, and any input tokens that exceed the maximum input sequence length are truncated. We train both f and g for 100 epochs. RangerLars (RAdam + LARS + Lookahead) optimizer with exponential learning rate decay is utilized. The initial learning rate and decay factor are set to 0.001 and 0.99, respectively.

In stage 3, we use three-layer MLP, including dropout and batch normalization as an encoder q_ϕ, and use one linear layer without bias as a decoder p_θ. For all phases, λ is set to 1, and the vocabulary size is set to 2,000. The backbone network f is fixed, and the encoder-decoder network is trained for 20 epochs. The Adam optimizer is used, with a learning rate of 2e-2. The number of topics K is set to 20, the batch size is set to 128, and the Dirichlet prior hyper-parameter α is set to 0.05 (i.e., 1/# of topics) for all phases.

Computing the Sliced Wasserstein Distance (SWD) between the embedding distribution and the prior requires a large batch for concise estimation. However, expanding the batch size is unattainable in many real-world scenarios due to memory constraints. We use a simple approach involving a dynamic queue, which does not require computing many samples' embeddings in a single iteration because embeddings from earlier iterations are saved. Embeddings computed from the current batch are concatenated with saved embeddings from the queue during training. Then, the stacked embeddings are finally used to calculate the distribution loss (Eq. 3, 11).

We should point out that the model's computing cost is not excessive. It took less than an hour for all datasets to perform all training phases with four A100 GPU processors.

A.4 Full evaluation results

In Table 9-20, we present the full performance evaluation results over four datasets with three different numbers of topics: 10, 20, and 50. For each experiment, results are averaged over five trials; the mean and standard deviation are reported. Also, to determine if a model successfully distinguish latent topics, we report results of topic diversity analysis for 20-Newsgroup, which shows that our method continues to outperform the baselines (Table 21). Top2Vec and ClusterTM are excluded in comparison, since they directly apply clustering algorithm (e.g.., DBSCAN) over the word set and inevitably make a complete split (i.e., topic diversity = 1), which is unfair to compare with.

A.5 Qualitative analyses

Table 22 reports the top 10 words for each topic extracted from our model. As discussed, we can find that the proposed model produces sufficiently representative words to interpret topics. For example, we can easily infer that topic #0 refers to the "computer" theme and topic #1 represents the "sports" theme. Our topic word selection strategy and contrastive learning framework successfully estimate the latent topics with improved interpretability.

---

[8] `http://qwone.com/~jason/20Newsgroups/`.

[9] `http://daviddlewis.com/resources/testcollections/reuters21578/`.

[10] `https://www.kaggle.com/datasets/benhamner/nips-papers`.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	0.0057±0.0023	0.0801±0.0057	0.0727±0.0103	0.1845±0.0028
NeuralLDA	-0.0158±0.0051	-0.0610±0.0158	-0.0429±0.0195	0.1741±0.0063
ETM	0.0052±0.0091	0.1219±0.0060	0.0527±0.0316	0.2027±0.0040
CTM	-0.0161±0.0080	0.1244±0.0135	-0.1415±0.0421	0.1818±0.0059
Top2Vec	0.0253±0.0017	0.0586±0.0020	-4.6113±0.0034	0.2677±0.0009
ClusterTM	0.0135±0.0025	-0.2870±0.0082	0.0160±0.0257	0.238±0.002
BERTopic	0.0609±0.0049	-0.0903±0.0180	0.2318±0.0063	0.2331±0.0070
UTopic	0.1069±0.0029	0.1130±0.0038	0.4850±0.0052	0.2416±0.0024

Table 9: Performance comparison over 20-Newsgroup with the number of topic $K = 10$. **[Note]** Mean and standard error over five trials are reported for Table 9-20.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0056±0.0051	0.0661±0.0074	0.0719±0.0118	0.1751±0.0042
NeuralLDA	-0.0227±0.0056	-0.0083±0.0090	-0.0634±0.0182	0.1933±0.0054
ETM	0.0234±0.0027	0.0927±0.0096	0.1207±0.0157	0.2247±0.0033
CTM	-0.0086±0.0048	0.1149±0.0138	0.0156±0.0173	0.1793±0.0055
Top2Vec	0.0302±0.0022	-0.0811±0.0046	0.1328±0.0089	0.2740±0.0033
ClusterTM	0.0154±0.0007	-0.2863±0.0062	0.0082±0.0103	0.2614±0.0021
BERTopic	0.0322±0.0035	-0.0563±0.0057	0.1515±0.0134	0.2210±0.0054
UTopic	0.0653±0.0030	0.1231±0.0059	0.3709±0.0098	0.2143±0.0029

Table 10: Performance comparison over 20-Newsgroup with the number of topic $K = 20$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0157±0.0007	0.0495±0.0042	0.0511±0.0062	0.1561±0.0022
NeuralLDA	-0.0154±0.0020	0.0008±0.0104	-0.0557±0.0111	0.1867±0.0030
ETM	0.0290±0.0030	0.0817±0.0025	0.1465±0.0078	0.2377±0.0019
CTM	-0.0159±0.0024	-0.1000±0.0038	0.0371±0.0072	0.1654±0.0012
Top2Vec	0.0188±0.0033	-0.0610±0.0042	0.0357±0.0126	0.2520±0.0054
ClusterTM	0.0115±0.0016	-0.2698±0.0033	-0.0223±0.0097	0.2982±0.0037
BERTopic	0.0122±0.0036	-0.0734±0.0092	0.0830±0.0177	0.2230±0.0043
UTopic	0.0425±0.0064	0.0685±0.0038	0.2829±0.0194	0.1966±0.0051

Table 11: Performance comparison over 20-Newsgroup with the number of topic $K = 50$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0771±0.0041	0.0854±0.0039	-0.1658±0.0158	0.1629±0.0034
NeuralLDA	-0.0446±0.0063	0.0536±0.0081	-0.0458±0.0193	0.1504±0.0031
ETM	-0.0569±0.0050	0.1031±0.0049	-0.1054±0.0137	0.1742±0.0021
CTM	-0.0467±0.0026	0.1262±0.0097	-0.0512±0.0093	0.1806±0.0025
Top2Vec	-0.0589±0.0069	0.0274±0.0093	-0.2803±0.0263	0.2206±0.0026
ClusterTM	-0.0164±0.0019	-0.2389±0.0096	-0.1115±0.0213	0.2942±0.0055
BERTopic	-0.0807±0.0022	0.0115±0.0060	-0.2706±0.0083	0.1870±0.0042
UTopic	-0.0138±0.0039	0.1225±0.0064	0.0884±0.0111	0.2350±0.0051

Table 12: Performance comparison over Reuters-21578 with the number of topic $K = 10$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0663±0.0040	0.0793±0.0037	-0.1327±0.0122	0.1622±0.0024
NeuralLDA	-0.0374±0.0035	0.0584±0.0017	-0.0264±0.0058	0.1521±0.0022
ETM	-0.0552±0.0019	0.0733±0.0021	-0.1063±0.0080	0.1740±0.0019
CTM	-0.0438±0.0048	0.1259±0.0055	-0.0580±0.0082	0.1669±0.0029
Top2Vec	-0.0448±0.0020	0.0595±0.0094	-0.2350±0.0073	0.2207±0.0033
ClusterTM	-0.0084±0.0025	-0.1105±0.0062	-0.0814±0.0119	0.2998±0.0069
BERTopic	-0.0724±0.0016	0.0126±0.0052	-0.2647±0.0136	0.1609±0.0052
UTopic	-0.0094±0.0052	0.1231±0.0047	0.0832±0.0192	0.2198±0.0050

Table 13: Performance comparison over Reuters-21578 with the number of topic $K = 20$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0734±0.0005	0.0477±0.0025	-0.1502±0.0050	0.1397±0.0019
NeuralLDA	-0.0300±0.0019	0.0833±0.0042	-0.0319±0.0122	0.1542±0.0022
ETM	-0.0399±0.0025	0.0609±0.0013	-0.0449±0.0078	0.1740±0.0024
CTM	-0.0491±0.0011	0.1420±0.0041	-0.0881±0.0050	0.1494±0.0011
Top2Vec	-0.0559±0.0024	0.0529±0.0054	-0.2125±0.0072	0.1966±0.0012
ClusterTM	-0.0056±0.0019	-0.1742±0.0034	-0.0181±0.0087	0.3307±0.0017
BERTopic	-0.0436±0.0030	0.0228±0.0055	-0.1403±0.0140	0.1797±0.0023
UTopic	-0.0117±0.0035	0.1081±0.0076	0.0841±0.0154	0.1938±0.0018

Table 14: Performance comparison over Reuters-21578 with the number of topic $K = 50$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0095±0.0041	0.0519±0.0067	0.0345±0.0140	0.1591±0.0032
NeuralLDA	0.0018±0.0060	0.0301±0.0074	0.0744±0.0145	0.1677±0.0032
ETM	-0.0152±0.0042	0.0545±0.0027	0.0843±0.0077	0.1642±0.0032
CTM	0.0608±0.0064	0.1556±0.0077	0.2320±0.0168	0.2058±0.0052
Top2Vec	-0.0121±0.0064	-0.0774±0.0308	-0.2059±0.0439	0.1858±0.0013
ClusterTM	0.0501±0.0020	-0.3748±0.0047	0.1637±0.0087	0.3132±0.0148
BERTopic	0.0578±0.0093	0.0106±0.0097	0.1580±0.0341	0.2136±0.0044
UTopic	0.0663±0.0021	0.0895±0.0054	0.1987±0.0089	0.2127±0.0023

Table 15: Performance comparison over Wikipedia with the number of topic $K = 10$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0072±0.0047	0.0618±0.0074	0.0377±0.0116	0.1462±0.0017
NeuralLDA	0.0121±0.0035	0.0792±0.0060	0.0784±0.0129	0.1628±0.0027
ETM	-0.0262±0.0016	0.0331±0.0030	0.0850±0.0074	0.1566±0.0007
CTM	0.0594±0.0037	0.1507±0.0026	0.2200±0.0084	0.2000±0.0019
Top2Vec	-0.0288±0.0038	-0.0730±0.0125	-0.2245±0.0265	0.1745±0.0046
ClusterTM	0.0533±0.0007	-0.3175±0.0068	0.1764±0.0084	0.3541±0.0043
BERTopic	0.0569±0.0044	-0.0374±0.0135	0.1786±0.0123	0.2164±0.0043
UTopic	0.0772±0.0037	0.1538±0.0044	0.2954±0.0092	0.2162±0.0018

Table 16: Performance comparison over Wikipedia with the number of topic $K = 20$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0047±0.0046	-0.0250±0.0055	0.0189±0.0143	0.1322±0.0028
NeuralLDA	0.0258±0.0021	0.1008±0.0032	0.1058±0.0090	0.1882±0.0028
ETM	-0.0468±0.0023	0.0081±0.0022	-0.0064±0.0040	0.1245±0.0008
CTM	0.0628±0.0023	0.1579±0.0025	0.2263±0.0081	0.1827±0.0023
Top2Vec	-0.0238±0.0026	-0.1124±0.0069	-0.2030±0.0089	0.1700±0.0039
ClusterTM	0.0667±0.0029	-0.2714±0.0093	0.2104±0.0115	0.3520±0.0032
BERTopic	0.0898±0.0043	-0.0617±0.0042	0.2664±0.0146	0.2287±0.0021
UTopic	0.0670±0.0018	0.0955±0.0031	0.2754±0.0085	0.2224±0.0027

Table 17: Performance comparison over Wikipedia with the number of topic $K = 50$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0746±0.0041	-0.0199±0.0009	-0.0684±0.0101	0.1573±0.0012
NeuralLDA	0.0001±0.0026	-0.0105±0.0124	0.0647±0.0076	0.1604±0.0061
ETM	-0.0212±0.0060	0.0441±0.0010	0.0829±0.0123	0.1865±0.0052
CTM	0.0436±0.0083	0.0714±0.0202	0.2543±0.0308	0.1822±0.0003
Top2Vec	0.0463±0.0022	0.0124±0.0101	0.1625±0.0056	0.2495±0.0002
ClusterTM	0.0255±0.0041	0.0656±0.0043	0.0588±0.0044	0.2732±0.0076
BERTopic	-0.0007±0.0145	0.0943±0.0006	0.0747±0.0074	0.1741±0.0068
UTopic	0.0938±0.0048	0.1256±0.0138	0.5388±0.0167	0.2265±0.0006

Table 18: Performance comparison over BBC with the number of topic $K = 10$.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0718±0.0033	-0.0205±0.0025	-0.0709±0.0094	0.1486±0.0010
NeuralLDA	0.0084±0.0010	0.0110±0.0089	0.0569±0.0036	0.1622±0.0026
ETM	-0.0333±0.0032	0.0251±0.0002	0.0416±0.0086	0.1908±0.0046
CTM	0.0289±0.0062	0.1109±0.0014	0.3254±0.0245	0.1715±0.0012
Top2Vec	N/A	N/A	N/A	N/A
ClusterTM	0.0339±0.0020	0.0990±0.0127	0.0908±0.0019	0.2858±0.0083
BERTopic	0.0456±0.0031	0.0762±0.0121	0.2556±0.0140	0.1943±0.0006
UTopic	0.0708±0.0061	0.1018±0.0113	0.3925±0.0124	0.2121±0.0006

Table 19: Performance comparison over BBC with the number of topic $K = 20$. **[Note]** N/A represents the model failed to detect topics for Table 19 and 20.

	NPMI-Wiki	NPMI-In	C_p	Word2vec
LDA	-0.0700±0.0010	-0.0153±0.0032	-0.0739±0.0036	0.1375±0.0006
NeuralLDA	-0.0035±0.0022	0.0248±0.0014	0.0471±0.0087	0.1577±0.0018
ETM	-0.0264±0.0018	0.0246±0.0010	0.0683±0.0109	0.2073±0.0098
CTM	0.0291±0.0022	0.1149±0.0059	0.2069±0.0109	0.1701±0.0004
Top2Vec	N/A	N/A	N/A	N/A
ClusterTM	0.0136±0.0032	0.0973±0.0007	0.0721±0.0037	0.3286±0.0043
BERTopic	N/A	N/A	N/A	N/A
UTopic	0.0897±0.0024	0.0898±0.0070	0.4448±0.0022	0.2212±0.0027

Table 20: Performance comparison over BBC with the number of topic $K = 50$.

Model	LDA	NeuralLDA	ProdLDA	ETM	CTM	BERTopic	UTopic
Diversity	0.7	0.81	0.85	0.52	0.83	0.91	**0.92**

Table 21: Performance comparison over 20-Newsgroup in terms of topic diversity ($K = 20$).

Topic #	Top-10 words from each topic
0	drive, disk, scsi, floppy, hard, ide, controller, cd, card, rom
1	game, baseball, team, pitching, brave, year, player, run, hitter, hit
2	window, widget, xterm, server, lib, x11r5, motif, program, client, libxmu
3	clinton, president, government, bush, tax, administration, libertarian, fbi, id, party
4	encryption, key, chip, clipper, phone, algorithm, government, escrow, encrypted, patent
5	god, jesus, satan, christian, heaven, christ, hell, sin, faith, bible
6	voltage, game, input, motorola, chip, amp, circuit, amplifier, board, audio
7	space, orbit, nasa, satellite, spacecraft, lunar, moon, earth, comet, jupiter
8	christian, islam, religion, jew, church, morality, christianity, quran, muslim, objective
9	food, patient, disease, doctor, treatment, pain, blood, medical, infection, cancer
10	mac, cpu, simms, modem, computer, 040, scsi, system, clock, motherboard
11	armenian, israel, israeli, turkish, arab, jew, armenia, turk, turkey, greek
12	homosexual, hey, sex, gay, men, steve, homosexuality, serdar, life, sexual
13	file, window, zip, directory, bmp, printer, format, microsoft, convert, program
14	test, max, article, printer, eric, matthew, pl, 145, david, email
15	car, engine, ford, oil, battery, dealer, gt, taurus, auto, vehicle
16	card, monitor, video, driver, printer, window, color, vga, ati, bus
17	bike, motorcycle, honda, helmet, bmw, rider, ride, riding, shaft, rear
18	gun, weapon, crime, firearm, handgun, police, criminal, homicide, defense, assault
19	game, team, hockey, nhl, playoff, player, season, goal, ranger, detroit

Table 22: Top-10 words selected for each topic from our model. The model is trained over the 20-Newsgroups dataset and the number of topics is set to 20. The discovered word set is human-interpretable and substantially different from one another.

Don't Mess with Mister-in-Between: Improved Negative Search for Knowledge Graph Completion

Fan Jiang and **Tom Drummond** and **Trevor Cohn**
School of Computing and Information Systems
The University of Melbourne, Victoria, Australia
`fan.jiang1@student.unimelb.edu.au`
`{tom.drummond, t.cohn}@unimelb.edu.au`

Abstract

The best methods for knowledge graph completion use a 'dual-encoding' framework, a form of neural model with a bottleneck that facilitates fast approximate search over a vast collection of candidates. These approaches are trained using contrastive learning to differentiate between known positive examples and sampled negative instances. The mechanism for sampling negatives to date has been very simple, driven by pragmatic engineering considerations (e.g., using mismatched instances from the same batch). We propose several novel means of finding more informative negatives, based on searching for candidates with high lexical overlaps, from the dual-encoder model and according to knowledge graph structures. Experimental results on four benchmarks show that our best single model improves consistently over previous methods and obtains new state-of-the-art performance, including the challenging large-scale Wikidata5M dataset. Combing different strategies through model ensembling results in a further performance boost.

1 Introduction

A Knowledge Graph (KG) is a structured form of human knowledge consisting of entities, facts, relationships between any pair of entities, and semantic descriptions of entities. As important structures that store millions of data records that represent a part of human knowledge, KGs have been proven to bring substantial benefits to a wide range of applications, including commonsense question answering (Yasunaga et al., 2021) and reasoning (Ren and Leskovec, 2020). Knowledge Graph Completion (KGC) supports the automatic construction or completion of a KG by finding the missing entity or link in incomplete triples.

Graph embedding and textual embedding methods are two mainstream techniques for KGC problems. The former typically map entities and relations into fixed dense vectors and maximises the probability of valid triples using specially-designed scoring functions (Bordes et al., 2013; Sun et al., 2019); while the latter additionally uses available textual descriptions associated with entities to gather more information (Wang et al., 2021b). Surprisingly, textual embedding methods lag behind graph embedding methods, perhaps due to their extra computational overheads in encoding textual inputs and, thus, inefficiency in incorporating a sufficient number of negatives (i.e., incorrect KG triples) to learn discriminative KG embeddings.

Recently, Wang et al. (2022) found that the key to making textual embedding methods outperform their graph embedding counterparts is to adopt a dual-encoder structure and train using contrastive learning to differentiate between positive training instances and randomly sampled negative instances (Karpukhin et al., 2020). Although their technique outperformed various graph embedding methods, achieving a new state-of-the-art, their means of negative sampling is not optimal: this in-batch negative method has been shown to be inefficient in training dual encoders (Lu et al., 2021). Instead, as they are highly similar to positives in terms of topics and lexicons, the use of so-called 'hard' negatives have been shown to result in better models in information retrieval (Xiong et al., 2021) through providing a more informative training signal and faster convergence. However, their effectiveness has not yet been established for KGC.

Therefore, to fill this gap, in this work, we aim to systematically investigate the effects of various hard negative sampling strategies for dual-encoder-based KGC. Specifically, we construct negative samples using three different ways. Our approach first evaluates the utility of negatives that share high lexical similarity with the head entity or the correct tail entity in terms of entity names and text descriptions. Based on the knowledge graph structures, we alternatively search negatives from the head or tail entity's local neighbourhood, hypothesising

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1818–1832
May 2-6, 2023

that the neighbourhoods of a certain entity that are not directly connected to it are highly related, but not so related to be false negatives (i.e., positives). Lastly, we investigate sampling so-called 'hard negatives' from top-k predictions generated by a baseline dual-encoder KGC model, as negatives that receive high scores are believed to be important and difficult to distinguish. In addition, in order to reduce possible false negatives, we also experimented with two variant neural negative sampling strategies according to heuristics. In summary, our contributions are:[1]

1. To the best of our knowledge, we are the first to systematically investigate the impacts of different types of negative sampling strategies for dual-encoder-based KGC.
2. We explore how best to combine the benefits of different negative sampling strategies to obtain further performance gains.
3. We compare our proposed negative searching methods on four benchmark datasets of different scales. Experimental results demonstrate that our best model significantly outperforms baselines, establishing a new state-of-the-art on all datasets, while ensembling leads to further performance gains.

2 Background

2.1 Task Formulation

In this paper, we deal with the task of predicting missing entities in knowledge graph completion. Formally, given a knowledge graph $\mathcal{G}$ which has a set of entities $\mathcal{E}$ and predefined relations $\mathcal{R}$, the tail entity retrieval task $(h, r, ?)$ requires retrieving a list of entities $\{t_1, t_2, \ldots, t_k\}$ from the entity set $\mathcal{E}$, ranked by their relevance to this head-relation pair (h, r).

2.2 Dual Encoder for KGC

Following the current state-of-the-art approach to KGC (Wang et al., 2022), we use a dual encoder framework.

Figure 1 shows the architecture. In this approach, a pair of encoders E_{hr} and E_t, which are usually initialised by pretrained language models (e.g., BERT (Devlin et al., 2019)), are used to map the head entity and relation (h, r) and tail entity t into dense vectors, respectively. More specifically, to encode the tuple (h, r), we first concatenate the text description of head entity h and relation r as: [CLS] *h: entity description* [SEP] *r* [SEP] and feed the sequence into a BERT encoder E_{hr}. Similarly, we use another encoder E_t to encode the tail entity t and its description, using an analogous input method. Mean pooling is used to obtain fix-sized embeddings, which are then l_2-normalised. The relevance score between (h, r) and t is calculated as the inner dot product of the two l_2-normalised vectors:

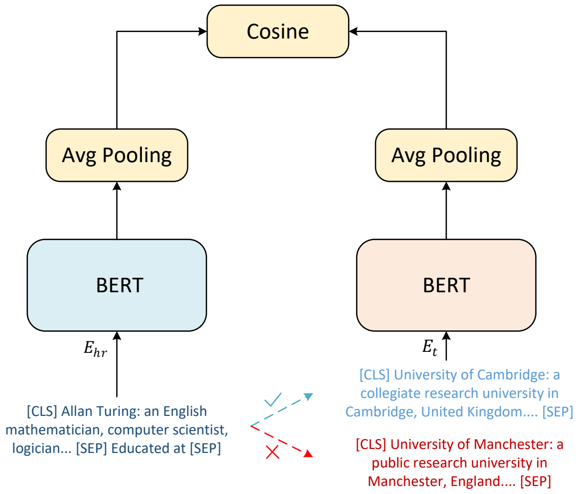


Figure 1: The Dual-Encoder Model for Knowledge Graph Completion, which learns to discriminate positive tail entities from negative ones.

$$s(h, r, t) = \frac{E_{hr}(h; r) \cdot E_t(t)}{||E_{hr}(h; r)|| \cdot ||E_t(t)||} \tag{1}$$

The dual encoders are normally trained to maximise the similarity scores of all positive triples (h, r, t). Here, we use the InfoNCE loss, following Wang et al. (2022):

$$\mathcal{L}_{hr \to t} = -\log \frac{s(h, r, t)}{\sum_{t' \in \mathcal{E}} s(h, r, t')} \tag{2}$$

where the denominator sums over all $N = |\mathcal{E}|$ entities in the KG.

Since N is usually very large, the common practice is to use specific negative sampling strategies to select a subset of negative samples to replace the full normalisation term in Equation 2. In practice, the selection of negative samples is crucial to the performance of a trained model (Zhang and Stratos, 2021). So the critical question becomes how to sample informative negatives which could generate meaningful signals for training an effective model.

---

[1] Source code is available at `https://github.com/Fantabulous-J/Improved-Negative-Search-for-KGC`.

3 Negative Sampling Strategies

3.1 In-Batch Negatives

This strategy treats as negatives the tail entities of other samples in the same mini-batch, which is a cheap way to obtain a large number of negatives (Karpukhin et al., 2020). However, the downside of this approach is that the number of negatives is limited to batch size, and it usually requires a large batch size to work well, which is not always feasible with limited computing resources.

3.2 Hard Negatives

'Hard' negatives are informative negatives, which are difficult to distinguish as they share similar characteristics with true positives (e.g., high lexical overlap or semantic similarity). They normally receive larger similarity scores from the model, which in turn results in larger loss gradients and thus larger parameter updates in training. Selecting negatives from them can effectively mitigate the diminishing gradient norms when using uninformative negative instances (i.e., in-batch negatives), thus providing more optimal training signals and leading to faster convergence speed (Xiong et al., 2021). In this paper, we systematically investigate the effectiveness of different hard negative sampling strategies for knowledge graph completion. An illustrative example of the different types of hard negatives is shown in Table 1.

Sparse Negatives Hard negatives were first shown to be useful for improving the performance of dense passage retrievers in question answering (Karpukhin et al., 2020). Their approach, which we call **Sparse Negatives** henceforth, samples hard negatives from the top-k ranked list returned by a sparse retrieval system such as BM25 (Robertson and Zaragoza, 2009). We test the generality of this method to our task of knowledge graph completion. More specifically, given a KG triplet (h, r, t), we either use the concatenation of the head entity h and relation type r or the correct tail entity t[2] to query a sparse BM25 retriever to find top entities which share similar entity names or similar tokens in their entity descriptions.[3]

Structure-Aware Negative Entities in the local neighbourhood of an entity in the KG are typically semantically related, and can serve as good candidates (Ahrabian et al., 2020). For a triplet (h, r, t), we sample hard negatives from the n-hop neighbours of either the head entity h or the tail entity t, following random edges.[4]

Neural Negative samples hard negatives from the top-k predictions of a neural model (Xiong et al., 2021; Glass et al., 2021), which has been shown to boost the performance of neural retrievers. In this paper, we use a simple method of sampling static hard negatives from a fixed learned KGC model, which we denote as **Head-Relation Negative**:

Step 1: In-batch negatives are used to train a dual-encoder KGC model.

Step 2: FAISS (Johnson et al., 2021) is used to index all entities in the KG, using their dense vector encoding, E_t.

Step 3: Employ approximate search to find the top-k retrieved entities using head-relation pair (h, r) as the query, with encoding E_{hr}. Then any positive tail entity to this query are removed based on the training dataset and all other entities will be kept as hard negatives.

Although sampled negatives are both difficult and informative, the Neural Negative method may end up including many false negatives. This is because for a good KGC model, if the same head-relation pair (h, r) appears in both training and test graphs, the top-k predictions will likely include correct answers in the test set. [5] Therefore, we propose another two variant negative sampling strategies to limit false negatives while maintaining informativeness. In particular, **Entity Similar Negative** uses either the head entity h or the tail entity t of a given KG triplet (h, r, t) as the query to find the top-k nearest entities in the embedding space of a fixed learned KGC model. **Replaced Head-Relation Negative** replaces the head h in (h, r, t) with another entity h' that is nearest to h in the embedding space and uses (h', r) as query to sample negatives from the top-k predictions.

3.3 Other Negatives

Pre-batch negatives extend in-batch negatives by using stale entity embeddings from previous n batches, which can be considered a cheaper way

---

[2] The concatenation of an entity's name and text description will be used.

[3] Whether the head h or the tail t is used to retrieve negatives is based on the development set performance.

[4] We use $n = 2$, which we found to work well.

[5] This is more true to neural negatives than previous two methods, as neural models are superior in capturing relation semantics. See Table 1 for more intuitive examples.

Head Entity **Relation** **Tail Entity**	`land_reform_NN_1`: a redistribution of agricultural land (especially by government action) hypernym `reform_NN_1`: a change for the better as a result of correcting abuses; justice was for sale before the reform of the law courts
Sparse	1. `landing_NN_3`: the act of coming down to the earth (or other surface); "the plane made a smooth landing"; "his landing on his feet was catlike" 2. `amphibious_landing_NN_1`: a military action of coordinated land, sea, and air forces organized for an invasion; "MacArthur staged a massive amphibious landing behind enemy lines" 3. `enderby_land_NN_1`: a region of Antarctica between Queen Maud Land and Wilkes Land; claimed by Australia
Structure	1. `event_planner_NN_1`: someone who plans social events as a profession (usually for government or corporate officials) 2. `price-fixing_NN_1`: control (by agreement among producers or by government) of the price of a commodity in interstate commerce 3. `lawlessness_NN_1`: a state of lawlessness and disorder (usually resulting from a failure of government)
Neural	1. `reform_NN_3`: self-improvement in behavior or morals by abandoning some vice; "the family rejoiced in the drunkard's reform" 2. `improvement_NN_1`: a change for the better; progress in development 3. `reform_NN_2`: a campaign aimed to correct abuses or malpractices

Table 1: Examples of hard negatives using different types of sampling strategies, using the head entity and relation as query for sparse and neural negatives but only the head entity for structure negatives (see top block). Sparse negatives typically overlap with the head entity in surface form but ignore the relation, while structure negatives can find entities with similar types. Neural retrievers can successfully find entities that are informative as hard negatives, but the risk of including false negatives also increases. For instance, both *improvement_NN_1* and *reform_NN_2* are feasible answers to the above query.

to expand negatives compared to memory bank approaches (He et al., 2020). Self-negatives regard the head entity h in $(h, r, ?)$ as the hard negative to reduce the model's reliance on spurious text matches. We include these two kinds of negatives, following Wang et al. (2022).

3.4 Negatives for Training

During training, we use one of the hard negative sampling strategies proposed in §3.2 together with a combination of in-batch negatives, pre-batch negatives, and self-negatives. Since extra hard negatives are used, the total number of instances used for loss calculation in Eq. 2 will be increased. For fair comparison, we reduce the batch size to ensure the total number of negatives used in training remains identical to our baseline *SimKGC* method (Wang et al., 2022). More details can be found in Appendix B.

4 Model Ensembling

Since we propose multiple hard negative sampling strategies in §3.2, we train several models, each with different types of hard negatives. It is natural to combine these models to achieve further performance improvements through model ensembling. Inspired by Lu et al. (2021), we explore two methods to ensemble results from multiple models.

Rank Fusion Reciprocal Rank Fusion (RRF) (Cormack et al., 2009) is a simple yet effective algorithm for merging ranking results from multiple retrievers, and is a prevailing technique in information retrieval. Its utility in knowledge graph completion has not yet been evaluated. The technique works by merging the ranking positions from the various models. Suppose we have a query (h, r) and each model $p \in P$ returns a ranking list $\{\text{Rank}_e^p | e \in \mathcal{E}\}$, the final ranking score is:

$$RRF(e) = \sum_{p \in P} \frac{w_p}{\text{Rank}_e^p}$$

Rank_e^p is the ranking position of an entity e returned by a model p, ranging from $[1, |\mathcal{E}|]$. w_p is the weight of a model p, which is tuned based on the performance on the development set.

Embedding Fusion An alternative aggregation method is to compute the weighted sum of ranking scores from different models, i.e., $\sum_{p=1}^{P} w_p E_{hr}^p E_t^p$. This can be implemented cheaply using the models' embeddings, which has the benefit of allowing for efficient vector indexing and search. Specifically, for each query (h, r), we obtain vectors from each models' encoder, which are then scaled by a model specific weight (tuned manually) and concatenated. Each entity $e \in \mathcal{E}$, is also embedded by each model and the vectors concatenated. This allows the aggregation method to be implemented as a simple dot product:[6]

$$s(h,r,t) = [w_P E_{hr}^P, \ldots, w_1 E_{hr}^1]^T [E_t^P, \ldots, E_t^1]$$

5 Experiments

5.1 Datasets

Our method is evaluated on WN18RR (Dettmers et al., 2018), FB15k-237 (Toutanova and Chen, 2015), DBPedia500k (Shi and Weninger, 2017) and Wikidata5M (*Transductive Setting*) (Wang et al., 2021b) datasets, where the number of entities ranges from 15K to 4.6M. More details about dataset statistics are shown in Table 7.

Following previous work (Wang et al., 2022), we focus on retrieving the missing entity of triples $(h, r, ?)$ and $(?, r, t)$, effectively doubling dataset sizes.[7] The adopted evaluation metrics are Mean Reciprocal Rank (MRR) and Hits@k($k \in \{1, 3, 10\}$). More specifically, MRR is calculated as $\mathrm{MRR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\mathrm{Rank}_i}$, where Rank_i is the rank of the correct tail entity in the predicted outputs and N is the total number of triples in the test set. Similarly, Hits@k is the proportion of correct tails that appear in the top-k ranked candidates.

We follow Bordes et al. (2013) and Wang et al. (2022) by conducting evaluations under *filter* settings, where for a given incomplete triple $(h, r, ?)$, the prediction scores of all its correct answers[8] except for the target to be predicted are removed. Both tail entity prediction $(h, r, ?)$ and head entity prediction $(?, r, t)$ are conducted, with their averages on all metrics reported.

5.2 Experimental Settings

We replicated the state-of-the-art SimKGC model (Wang et al., 2022) using PyTorch (Paszke et al., 2019) and Transformers (Wolf et al., 2020), and treat it as the baseline in our experiments. The uncased BERT base model is used for model initialisation.[9]

Most hyperparameters are adopted from Wang et al. (2022), and newly introduced hyperparameters are determined through grid search. We choose the best model according to Hits@1 on the dev set, which is then evaluated on the test set for all experiments. More details are in Appendix A.

5.3 Overall Results

Tables 2 and 3 show the results of our models compared with a range of high-performing graph embedding and text embedding models over four KGC datasets. We observe that the performance of our replicated baseline is competitive with that of Wang et al. (2022), achieving slightly better results on FB15k-237 and Wikidata5M but slightly worse on WN18RR. Moreover, our best single model either matches or outperforms the state-of-the-art results on all datasets for most metrics.

By looking into the models trained using different hard negative sampling strategies, the behaviours are quite different in each dataset. More specifically, simply taking the n-hop neighbours as structure-aware negatives results in significant improvement on WN18RR but marginal increases or even negative impacts on the other three datasets. Similar effects are also observed for sparse negatives. Our model achieves the best results on FB15k237, DBPedia500k, and Wikidata5M when using replaced head-relation negatives, which shows our heuristics are useful and can indeed lead to better performance. Gains over the baseline when using entity similar negatives are also significant, but it lags behind the other two types of neural negatives most of the time. Overall, we can conclude that there is no single negative sampling strategy that outperforms on all datasets. We believe this is due in part to the distinct characteristics of each dataset, which we explore in §6.1.

Furthermore, by ensembling models trained using different types of negatives at inference time, we can observe performance gains up to 1.3% MRR and 1.8% H@1 over the best single model. For model ensembling methods, we find that embedding fusion is more beneficial when improving the precision (MRR and H@1), while rank fusion is helpful to boost the recall (H@3 and H@10), especially on DBPedia500k. Besides, both methods do not help much on FB15k237 with only marginal gains. However, both ensembling methods come at the cost of increased inference latency with a factor of N, the number of ensembled models, which

---

[6] For both fusion methods, the weights were tuned through grid search. Empirically, we found that single model whose performance is better was generally given a higher weight than worse performing methods. See Table 8 and Appendix C.3 for more details.

[7] For reversed triples, a special r^{-1} relation type is used. For instance, we convert a triple (?, educated at, Cambridge University) to (Cambridge University, reverse educated at, ?). We did not conflate the reversed relations with existing relations, although this may be beneficial (e.g., hyponym $\equiv$ reverse hypernym.)

[8] Triples that appear in training, validation and test sets.

[9] `https://huggingface.co/bert-base-uncased`

Method	WN18RR				FB15k-237				Wikidata5M			
	MRR	**H@1**	**H@3**	**H@10**	**MRR**	**H@1**	**H@3**	**H@10**	**MRR**	**H@1**	**H@3**	**H@10**
Graph Embedding Approach												
TransE (Bordes et al., 2013)*	24.3	4.3	44.1	53.2	27.9	19.8	37.6	44.1	25.3	17.0	31.1	39.2
DisMult (Yang et al., 2014)*	44.4	41.2	47.0	50.4	28.1	19.9	30.1	44.6	-	-	-	-
RotatE (Sun et al., 2019)*	47.6	42.8	49.2	57.1	33.8	24.1	37.5	53.3	29.0	23.4	32.2	39.0
TuckER (Balazevic et al., 2019)*	47.0	44.3	48.2	52.6	35.8	26.6	39.4	**54.4**	-	-	-	-
Text Embedding Approach												
KG-BERT (Yao et al., 2019)*	21.6	4.1	30.2	52.4	-	-	-	42.0	-	-	-	-
MTL-KGC (Kim et al., 2020)*	33.1	20.3	38.3	59.7	26.7	17.2	29.8	45.8	-	-	-	-
StAR (Wang et al., 2021a)*	40.1	24.3	49.1	70.9	29.6	20.5	32.2	48.2	-	-	-	-
DKRL (Xie et al., 2016)*	-	-	-	-	-	-	-	-	16.0	12.0	18.1	22.9
KEPLER (Wang et al., 2021b)*	-	-	-	-	-	-	-	-	21.0	17.3	22.4	27.7
SimKGC (Wang et al., 2022)*	66.7	58.8	72.1	80.5	33.6	24.9	36.2	51.1	35.8	31.3	37.6	44.1
SimKGC replicated	65.0	55.6	71.5	**81.4**	33.8	25.3	36.5	51.2	36.8	32.5	38.3	44.7
+ Head-Relation	64.3	57.1	68.6	77.6	36.2	27.4	39.2	53.7	40.2	36.4	41.8	46.8
+ Entity Similar	66.4	59.5	70.1	79.1	35.4	26.9	38.4	52.3	39.3	35.5	40.9	46.0
+ Replaced Head-Relation	65.3	58.8	68.8	77.5	**36.5**	**27.6**	**39.9**	54.2	**41.0**	**37.0**	**42.7**	**48.0**
+ Sparse	66.7	59.3	71.3	79.3	34.6	25.9	37.5	52.2	36.7	32.3	38.4	44.5
+ Structure-Aware	**67.8**	**60.9**	**72.5**	80.0	33.1	24.4	35.7	50.7	38.0	33.4	39.7	46.2
Ensemble												
Rank Fusion	68.9	61.9	**72.9**	**81.7**	36.6	27.6	**40.1**	**54.3**	41.7	37.6	**43.5**	48.7
Embedding Fusion	**69.2**	**62.7**	72.8	81.1	**36.7**	**27.8**	40.0	**54.3**	**42.0**	**38.1**	**43.5**	**49.0**

Table 2: Evaluation results on the test set of WN18RR, FB15k-237 and Wikidata5M (*Transductive Setting*) datasets. $*$ indicates results directly copied from Wang et al. (2022).

Method	MRR	H@1	H@3	H@10
TransE (Bordes et al., 2013)*	7.4	3.1	10.1	14.5
TransH (Wang et al., 2014)*	7.4	3.2	10.1	14.6
TransR (Lin et al., 2015)*	7.3	3.5	9.9	13.5
TransD (Ji et al., 2015)*	7.4	3.2	10.1	14.5
SimKGC replicated	25.7	18.9	27.6	**39.5**
+ Head-Relation	26.3	20.5	28.6	37.0
+ Entity Similar	26.3	20.7	28.4	36.5
+ Replaced Head-Relation	**27.1**	**21.1**	**29.3**	38.1
+ Sparse	24.9	18.4	26.7	37.6
+ Structure-Aware	25.6	18.8	27.5	39.4
Ensemble				
Rank Fusion	**28.3**	**21.6**	**30.6**	**41.7**
Embedding Fusion	27.9	21.5	30.0	40.0

Table 3: Evaluation results on the DBPedia500k dataset. $*$ indicates the results obtained by running the open-source OpenKE toolkit (Han et al., 2018).

hinders their utility for real-time deployment.

6 Analysis

6.1 Why the Effects Vary by Dataset

Next, we analyse why the benefits of different hard negative sampling strategies vary from dataset to dataset. We employ two measurements, namely *Difficulty* and *False Negative Rate*, to draw the connections to the performance on specific datasets. *Difficulty* is measured by the average model score between hard negatives and their corresponding head-relation pairs. More specifically, for a given KG triplet (h_i, r_i, t_i) and its associated hard negative pool $\mathcal{N}_i = \{t_j | 1 \leq j \leq k\}$, the difficulty is computed as follows:

$$D_i = \frac{1}{|\mathcal{N}_i|} \sum_{t_j \in \mathcal{N}_i} s(h_i, r_i, t_j)$$

where $s(h_i, r_i, t_j)$ is the score predicted by the *SimKGC replicated* model.[10] The overall difficulty is the average over training, $\frac{1}{|\mathcal{D}|} \sum_i^{|\mathcal{D}|} D_i$.

On the other hand, *False Negative Rate* is the proportion of hard negatives which are correct answers that appear in development and test graphs:[11]

$$FNR = \frac{1}{|D|} \sum_i^{|\mathcal{D}|} \frac{1}{|\mathcal{N}_i|} \sum_{t_j \in \mathcal{N}_i} \mathbb{I}(< h_i, r_i, t_j > \in \mathcal{T})$$

where $\mathcal{T}$ is the development and test sets. $\mathbb{I}(*) = 1$ if the corresponding triple appears in the development or test set; otherwise it is 0.

---

[10] The model is fixed to allow comparison of different negative sampling strategies.

[11] The real false negative rate will be higher than what we measure due to the incomplete nature of knowledge graphs.

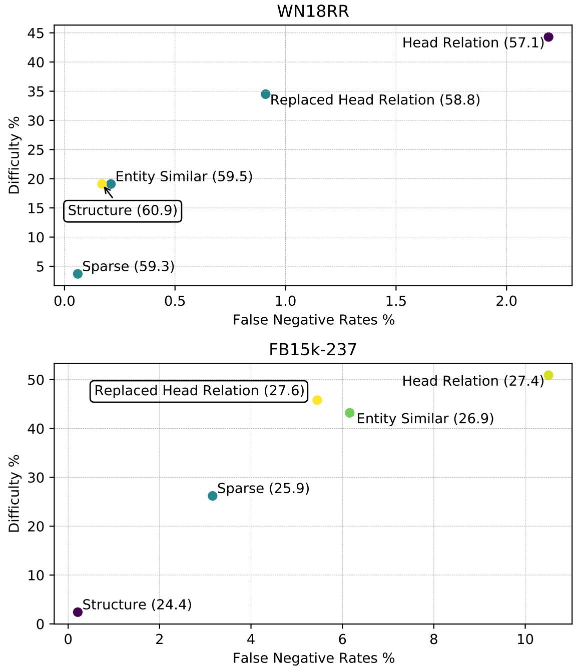


Figure 2: The difficulty and false negative rates of hard negatives extracted using different sampling strategies on WN18RR and FB15k237 datasets. Colours of points and numbers represent the Hit@1 score. The best results are circled with round boxes.

A model that is trained using hard negatives with high difficulty and low false negative rate is expected to achieve better performance. Figure 2 compares the difficulty and false negative rates for hard negative sampling strategies over two datasets. The most effective strategies are different on the two datasets: for Wordnet (WN18RR) the accuracy (Hit@1) is most sensitive to the false negative rate (horizontal axis), while for Freebase (FB15k237) it is most sensitive to difficulty (vertical) – despite the false negative rates being considerably higher on this dataset. We ascribe the difference to the underlying dataset. Wordnet is a sparsely-connected graph, and has relation types that connect entities with hierarchical structures. For instance, the *hypernym* and *derivationally related form* relations which satisfy the transitivity property account for 74% of triples in the graph. Including negatives with high difficulty for such relations (i.e., false negatives on the hierarchy that are not directly connected to a specific head entity) will inevitably encourage the model to learn embeddings that destroy such structures. However, the best-performing structure-aware negative sampling methods only sample negatives from local neighbourhoods, which we believe can effectively reduce such side-effects. In contrast, Freebase is a much more denser graph compared to Wordnet (8× in terms of node degrees), with significantly fewer entities but more diverse relations. Moreover, over 70% triples include relations that have high arity (e.g., *has part*) and many-to-many mappings. Thus, the chance of having false negatives within globally sampled hard negatives will be much higher than that on Wordnet in nature. However, since FB15k237 is a much more difficult task compared to WN18RR, highly difficult negatives should be used to encourage the model to learn embeddings that can discriminate correct tails from very similar ones, and the importance of false negatives are comparatively less important. We leave the systematic analysis of different negative sampling methods with respect to KG structures as future work.

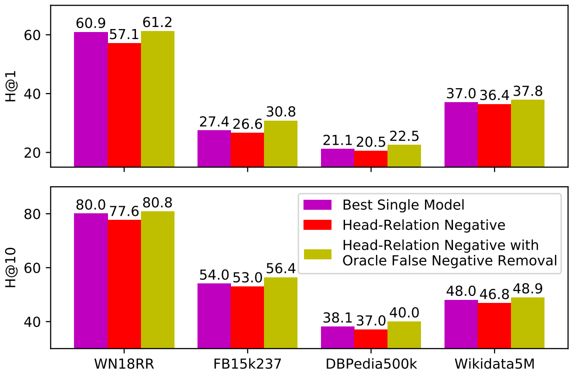


Figure 3: The comparison between the best single model and the model trained using **Head-Relation** negatives with and without oracle false negative removal.

6.2 Oracle Upper Bound

One natural following question is how much improvement we may achieve if we remove all false negatives when using the most difficult head-relation hard negatives to train a model. As shown in Figure 3, the performance increases substantially in terms of H@1 and H@10 across four datasets when simply removing false negatives. This shows that there is much room for improvement and designing effective false negative elimination methods could potentially fill the performance gap and result in better-performing models. Besides, our best single model can obtain results that are close to the upper bound on WN18RR and Wikidata5M, but still lags behind on the other two datasets.

Rel Category		**1-TO-1**	**1-TO-M.**	**M.-TO-1**	**M.-TO-M.**
Size %		0.94	6.32	20.45	72.29
SimKGC	Forward	57.3	**4.6**	73.2	24.8
	Backward	64.6	37.1	**13.6**	15.3
	Avg.	60.9	20.8	**43.4**	20.1
Our model	Forward	**60.9**	4.3	**75.8**	**28.9**
	Backward	**67.7**	**40.8**	10.5	**17.4**
	Avg.	**64.3**	**22.6**	43.1	**23.2**

Table 4: Detailed results (H@1) on the FB15k-237 test dataset, broken down by category of relationships and prediction directions. Our model refers to the best single model trained with replaced head-relation negatives.

6.3 Relation Category

To further understand the behaviour of our model, we follow Bordes et al. (2013) by classifying triples into different groups based on the category of relationships. Relationships are broken down into four categories according to the cardinalities of their head and tail arguments: one-to-one (1-TO-1), one-to-many (1-TO-M.), many-to-one (M.-TO-1) and many-to-many (M.-TO-M.). For a given relation r, if the average number of heads h appearing in the dataset for a tuple (r, t) is lower than 1.5, the head argument will be labeled as 1 and M. otherwise. The same is applied to the tail argument.

Table 4 shows the detailed results of four categories on the FB15k-237 dataset, together with the forward and backward prediction results. Firstly, we can find that both models perform the best on triples with 1 on the tail side, while predicting the M side is significantly more difficult. Secondly, our method can beat the baseline on most relation categories. Thirdly, the substantial improvement on 1-TO-1 relations (+3.4%) shows that our model is making more precise decisions. Besides, by looking into specific prediction directions, we find that the performance mainly comes from predicting triples with 1 on the target side. When the target side contains multiple answers, adding hard negatives even hurts the performance, especially in the backward direction of M.-TO-1. Further analysis finds that its false negative rate is almost three times the overall rate (15.9% vs 5.5%), and we believe it is the high false negative rate that misleads the model and results in negative impacts on this specific relation category.

WN18RR	**Model**	**MRR**	**H@1**	**H@3**	**H@10**
Seen	SimKGC	65.5	56.1	72.0	**81.7**
	Our model	**68.2**	**61.3**	**72.6**	80.0
Unseen	SimKGC	59.1	48.6	65.2	77.6
	Our model	**63.7**	**54.1**	**70.2**	**81.0**
Wikidata5M	**Model**	**MRR**	**H@1**	**H@3**	**H@10**
Seen	SimKGC	36.6	32.3	38.1	44.5
	Our model	**40.7**	**36.7**	**42.4**	**47.7**
Unseen	SimKGC	43.4	38.4	45.3	51.6
	Our model	**49.7**	**45.6**	**51.9**	**57.2**
Inductive Unseen	SimKGC	42.5	37.8	43.9	51.6
	Our model	**47.4**	**43.3**	**49.3**	**55.1**

Table 5: Results on test examples when only containing seen and unseen entities, respectively. Our model refers to the best single model in each dataset.

6.4 Generalise to Unseen Entities

Textual embedding methods are known to generalise better to unseen entities than graph embedding ones (Wang et al., 2021a). We conduct experiments to testify whether their generalisation ability can be further improved by using harder negatives. We split test data based on whether the head or target entity is unseen in training. 210 out of 3134 and 159 out of 5163 test triples include unseen entities on WN18RR and Wikidata5M, respectively. Furthermore, we use another *Inductive* setting on Wikidata5M, which has different data splits compared to the *Transductive* setting. We extract 336 test triples from the whole graph of *Induction* setting which include entities unseen to the *Transductive* setting's training graph. The detailed results are shown in Table 5. We observe our best model achieves considerable improvements on three unseen entity settings across all metrics. Moreover, the absolute improvements are significantly higher than those on the seen entity setting, especially on recall metrics (e.g., H@3 and H@10). This demonstrates learning from hard negatives leads entity embeddings that generalise better.

7 Related Work

Knowledge Graph Completion KGC has been extensively studied for many years as a popular research topic. Conventional KGC methods adopt graph embedding methods to map entity and relation into low-dimensional dense vectors and design various scoring functions to measure the plausibility of KG triples, including TransE (Bordes et al., 2013), DistMult (Yang et al., 2014), ComplEx (Trouillon et al., 2016) and RotatE (Sun et al., 2019). Recent text embedding methods choose to include additional text descriptions related to entities by making use of large pretrained language models, including KG-BERT (Yao et al., 2019), MTL-KGC (Kim et al., 2020) and StAR (Wang et al., 2021a). In this work, we follow Wang et al. (2022) by adopting the simple yet effective text embedding-based method using powerful pretrained language models. The above graph-based methods, which also rely on negatives to learn entity and relation embeddings, are orthogonal to the sampling strategies proposed in this work. We believe our techniques could lead to improvements when applied to graphs, but we leave this for future work.

Dual Encoder for Contrastive Learning A Dual Encoder, or Bi-Encoder, which adopts two encoders without weight sharing for feature encoding, has been widely used in many tasks, including image learning (He et al., 2020) and information retrieval (Karpukhin et al., 2020). Typically, an image with two different augmented views or query-document pairs is encoded into vectors separately by a dual encoder. The model learns to minimise the distance between positive pairs and push negative pairs further apart in the embedding space. Inspired by previous work, we decouple the encoding of (h, r) and t by dual encoder and use the contrastive learning framework to learn effective knowledge embeddings.

Hard Negatives for Contrastive Learning Hard negatives have been identified to be extremely helpful in learning better representations, including image learning (Robinson et al., 2021) and information retrieval (Karpukhin et al., 2020). For example, the DPR model (Karpukhin et al., 2020) mines hard negatives using a sparse retriever BM25. Xiong et al. (2021) proposed to sample hard negatives from the model itself by theoretically and empirically verifying such negatives can result in larger gradient norms and thus faster convergence speed. Zhang and Stratos (2021) also found doing negative sampling from the model being optimised leads to better performance, as they argued that contrastive learning is a biased estimator and sampling negatives from the model itself can reduce such bias. We follow this direction and propose various hard negative search methods in this paper and show that they can substantially improve KGC.

Negative Sampling Strategies for KGC Most KGC works employ a simple negative sampling strategy by corrupting the head entity h or tail entity t of a correct KG triplet (h, r, t) with uniformly sampled random entities from the whole knowledge graph (Yao et al., 2019). However, such easy negatives are identified to provide limited training signal, since most of them produce small scores and nearly zero gradients late in training (Sun et al., 2019). Therefore, various improved negative sampling strategies have been proposed. Self-adversarial negative sampling (Sun et al., 2019) uses the distribution generated by the model being optimised to weight sampled negatives. GAN-based methods (Wang et al., 2018) are also effective in extracting informative negatives but suffer from inefficiency and high training costs. NSCaching (Zhang et al., 2019) regards negative triplets with large scores as important and maintains a cache of such triples from which negatives are sampled. Ahrabian et al. (2020) takes the head or tail entity's n-hop neighbours as negatives and evaluates their utility on graph embedding methods. Our work also aims to explore the effects of improved negative sampling strategies for KGC but with different model architectures (dual-encoder vs graph embeddings). Many of the above negative sampling strategies are orthogonal to our work and we believe they have the potential to be employed within our method for further empirical gains.

8 Conclusion

In this paper, we successfully improve a powerful dual-encoder-based KGC model by introducing various improved negative search methods and investigating their combinatorial effects. Empirical results on four benchmarks with different scales confirm the superiority of our proposed methods, significantly beating a wide range of competitive methods and achieving state-of-the-art performance. For future work, we are interested in eliminating false negatives contained in sampled hard negatives. An exciting future direction is to use a model to filter out false negatives and potentially even identify pseudo positives, for more accurate models.

Limitations

Although our method is efficient and introduces no extra cost during inference time, it does incur the additional training cost of retrieving different types of hard negatives, which roughly doubles the time for completing the whole training pipeline. The model ensembling methods, especially the embedding fusion one, need to build indexes for entity embeddings and require extra cost for storage. Despite these drawbacks, the training of our methods has a fairly modest footprint by modern standards, taking about 36 hours of a server with 4×A100 GPUs and 200G CPU RAM for the largest datasets, Wikidata5M.

As an empirical study, we provide observations under different design choices for sampling hard negatives. We hope our findings can shed lights on future work. A theoretical analysis about each proposed negative sampling strategy with connection to KG properties (e.g., sparsity) would certainly strengthen our claims, but is out of scope of this paper. In addition, the state-of-the-art dual-encoder-based KGC model is used to verify the sampling methods proposed in this work. One would expect to test their generalisation to other underlying KGC methods, e.g., graph embedding methods, which we leave as future work.

Moreover, adapting the dual-encoder-based KGC model and our proposed negative sampling methods to multilingual KGs, e.g., for a KG with concepts in different languages, or multi-modal settings, for a KG with concepts in the form of images, videos, or audios would further test the generalisation ability, which can be a promising research direction for future work.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback and suggestions. The first author is supported by the Graduate Research Scholarships funded by the University of Melbourne. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Facility was established with the assistance of LIEF Grant LE170100200.

References

Kian Ahrabian, Aarash Feizi, Yasmin Salehi, William L. Hamilton, and Avishek Joey Bose. 2020. Structure aware negative sampling in knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6093–6101, Online. Association for Computational Linguistics.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5185–5194, Hong Kong, China. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA. Association for Computing Machinery.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

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

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, and Alfio Gliozzo. 2021. Robust retrieval augmented generation for zero-shot slot filling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1939–1949, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xu Han, Shulin Cao, Xin Lv, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. OpenKE: An open toolkit for knowledge embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 139–144, Brussels, Belgium. Association for Computational Linguistics.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 687–696, Beijing, China. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Herve Jegou. 2021. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Bosung Kim, Taesuk Hong, Youngjoong Ko, and Jungyun Seo. 2020. Multi-task learning for knowledge graph completion with pre-trained language models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1737–1743, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2181–2187. AAAI Press.

Jing Lu, Gustavo Hernandez Abrego, Ji Ma, Jianmo Ni, and Yinfei Yang. 2021. Multi-stage training with improved negative contrast for neural passage retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6091–6103, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA.

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Neural Information Processing Systems*.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2021. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*.

Baoxu Shi and Tim Weninger. 2017. Open-world knowledge graph completion.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2071–2080. JMLR.org.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, Ying Wang, and Yi Chang. 2021a. Structure-augmented text representation learning for efficient knowledge graph completion. In *Proceedings of the Web Conference 2021*, WWW '21, page 1737–1748, New York, NY, USA. Association for Computing Machinery.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022. SimKGC: Simple contrastive knowledge graph completion with pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4281–4294, Dublin, Ireland. Association for Computational Linguistics.

Peifeng Wang, Shuangyin Li, and Rong Pan. 2018. Incorporating gan for negative sampling in knowledge representation learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2659–2665. AAAI Press.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Wenzheng Zhang and Karl Stratos. 2021. Understanding hard negatives in noise contrastive estimation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1090–1101, Online. Association for Computational Linguistics.

Yongqi Zhang, Quanming Yao, Yingxia Shao, and Lei Chen. 2019. Nscaching: Simple and efficient negative sampling for knowledge graph embedding. *2019 IEEE 35th International Conference on Data Engineering (ICDE)*.

	WN18RR		FB15k-237	
	MRR	**H@1**	**MRR**	**H@1**
Best Single Model	**67.8**	**60.9**	**36.5**	**27.6**
Negative Combination	65.8	59.0	35.6	26.7

Table 6: Results when training a model by sampling from the union of all types of negatives.

A Hyperparameters

The learning rates are set to 5×10^{-5} on WN18RR, 3×10^{-5} on Wikidata5M and 1×10^{-5} on the remaining datasets. All models are trained using Adam optimizer (Kingma and Ba, 2015) with a warmup learning rate scheduler. The model is trained for 50, 10, 5, and 1 epochs on WN18RR, FB15k-237, DBPedia500k, and Wikidata5M. For each hard negative sampling strategy, we generate 30 negatives for each training example.[12] During each training step, we uniformly sample a subset of hard negatives from the pool for each training example, and the best number is chosen from $[1, 5]$. For both rank fusion and embedding fusion methods, their weights are shared and are tuned based on the performance on development sets. A summary of training details and hyperparameters is shown in Table 8.

B Number of Negatives for Training

For each training example in a mini-batch, we uniformly sample N negatives from its associated hard negative pool. We also treat the hard negatives and self-negatives of other examples in the same mini-batch as in-batch negatives. Suppose the batch size is B and pre-batches are M, the total number of instances used for loss calculation in Eq. 2 will be $(N + M + 2) \times B$ for each training example. By contrast, the number of negatives used by SimKGC is $(M + 2) \times B$. If we keep the same batch size, the number of negatives used in our experiment will increase by $N \times B$, which would potentially weak our claims as more negatives are used for contrastive learning. To ensure fair comparison, we reduced the batch size so that the number of negatives used in our experiment is the same as Wang et al. (2022). For example, if we take $B = 768, N = 1, M = 1$ on WN18RR, the number of negatives equals to 3072; while we take $B = 1024$ for SimKGC, the number will also be 3072. Thus, our methods will not be affected by including more negatives. Similar settings apply to other datasets.

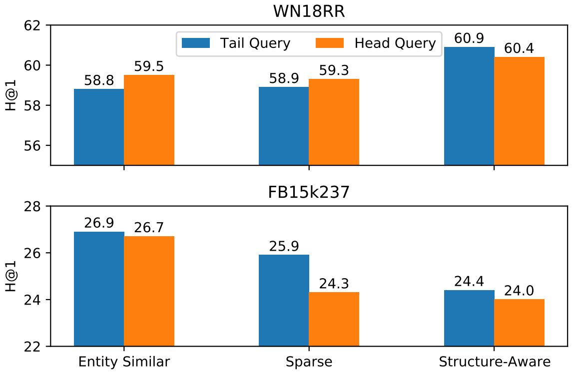


Figure 4: Results comparison when taking head or tail entity as query.

C Ablation Study

C.1 Negative Combination

Another possible way to benefit from all kinds of hard negatives is to train a model on their combinations. We experiment with training a model by uniformly sampling negatives from the union of all types of hard negatives generated from §3.2. As shown in Table 6, the model does not obtain improved results and this can even hurt the performance. We reason that although the combined negatives can provide more diverse supervisions, their false negative rates and difficulty also changes. More specifically, the false negative rate and difficulty on WN18RR change to 0.93% and 29.66%. Although more difficult negatives are included, the more important false negative rate is worse than the best structure-aware one. By contrast, the false negative rate in FB15k237 after combination reduces from Replaced Head Relation's 5.45% to 4.24%, but the more important difficulty also decreases from 45.8% to 33.7%, thus leading to inferior results.

C.2 Head Entity vs Tail Entity as Query

We analyse the performance difference between using head entity and tail entity as the query in the *sparse*, *structure-aware* and *entity similar* negative sampling strategies. Figure 4 shows that for similar and sparse negatives, using tail entity as query generally leads to worse performance on WN18RR; while in other cases, using the tail entity is more beneficial. One reason behind this is that taking the head entity as query would inevitably reduce

---

[12] Gold answers appearing in the training graph are removed for each (h, r). (i.e., $\mathcal{N} = \{t_j | (h, r, t_j) \notin \mathcal{D}\}_{j=1}^{k}$)

Dataset	#Ent	#Rel	Train	Dev	Test
WN18RR	40, 943	11	86, 835	3, 034	3, 134
FB15k-237	14, 541	237	272, 115	17, 535	20, 466
DBPedia500k	517, 475	654	3, 102, 677	10, 000	1, 155, 937
Wikidata5M	4, 594, 485	822	20, 614, 279	5, 163	5, 163

Table 7: Number of Entities, Relations and Triples in Train/Dev/Test splits of datasets used in our experiments.

Hyperparameters	WN18RR	FB15k-237	DBPedia500k	Wikidata5M
Learning rate	5e-5	1e-5	1e-5	3e-5
LR Scheduler	Linear Warmup	Linear Warmup	Linear Warmup	Linear Warmup
Warmup steps	400	400	400	400
Epochs	50	10	5	1
Batch Size for SimKGC	1024	1024	1024	1024
Batch size	768	512	512	768
Gradient clipping	10.0	10.0	10.0	10.0
#Hard negatives	1	3	3	1
Fusion weights	[0.1, 0.3, 0.3, 0.3, 1.2]	[0.6, 0.2, 0.9, 0.05, 0.01]	[1, 1, 1, 1, 1]	[0.9, 0.6, 1.2, 0.3, 0.45]

Table 8: Hyperparameter settings for different datasets. Entries in fusion weights correspond to Head-Relation, Entity Similar, Replaced Head-Relation, Sparse, and Structure-Aware sampling methods, respectively.

		MRR	**H@1**	**H@3**	**H@10**
Rank Fusion	Uniform	68.7	61.7	72.7	81.6
	Tuned	**68.9**	**61.9**	**72.9**	**81.7**
Embedding Fusion	Uniform	68.9	62.5	72.5	80.8
	Tuned	**69.2**	**62.7**	**72.8**	**81.1**

Table 9: Results comparison on WN18RR when using uniform and manually-tuned weights for model ensembling.

the difficulty but can avoid false negatives, a factor that is more significant for WN18RR. By contrast, the situation on FB15k237 is the other way around where the difficulty is more crucial. Again, we can confirm that there is no single design choice for each sampling strategy that works the best across datasets.

C.3 Uniform Weights for Ensembling

We show results when using uniform weights for model ensembling in Table 9. We find that simply using uniform weights has already boosted the performance. By carefully tuning the weights of each model through grid search, we can achieve further gains on all metrics, although the benefits are minimal. Making fusion weights learnable (Wang et al., 2021a) may lead to further improvements, which we leave as future work.

D Qualitative Analysis

Table 10 shows some examples of predictions from the SimKGC baseline and our best method. For the first example, both methods' predictions are wrong. However, our method ranks the correct tail entity much higher than the SimKGC baseline. Moreover, the top-1 prediction from our method *prosody_NN_1* is more reasonable as an answer compared to *articulation_NN_1*. For the second example, both methods fail according to the test set annotations. Our method returns correct tail entities based on our judgment. The same applies to SimKGC for its top-2 predictions. This also shows current automatic evaluation metrics cannot reflect the performance of KGC models precisely. For the third example, our method finds the correct answer out of other very semantically similar entities.

I. *Both SimKGC and our model fail, but our predictions are more related*	
Head Entity Relation Tail Entity	accentuation_NN_1: the use or application of an accent; the relative prominence of syllables in a phrase hypernym stress_NN_1: the relative prominence of a syllable or musical note (especially with regard to stress or pitch)
SimKGC (Rank 180)	1. articulation_NN_1: the aspect of pronunciation that involves bringing articulatory organs together so as to shape the sounds of speech 2. prosody_NN_1: the patterns of stress and intonation in a language 3. non-standard_speech_NN_1: speech that differs from the usual accepted, easily recognizable speech of native adult members of a speech community
Ours (Rank 108)	1. prosody_NN_1: *as above* 2. articulation_NN_1: *as above* 3. speech_pattern_NN_1: distinctive manner of oral expression; "he couldn't suppress his contemptuous accent".
II. *Both SimKGC and our model fail, but both models' predictions are correct based on human judgement*	
Head Entity Relation Tail Entity	pea_family_NN_1: a large family of trees, shrubs, vines, and herbs bearing bean pods... member meronym wild_pea_NN_1: any of various plants of the family Leguminosae that usually grow like vines.
SimKGC (Rank 7)	1. genus_sesbania_NN_1: small genus of tropical and subtropical leguminous herbs or shrubs or trees 2. genus_centrosema_NN_1: a genus of chiefly tropical American vines of the family Leguminosae... 3. torchwood_family_NN_1: resinous or aromatic chiefly tropical shrubs or trees
Ours (Rank 4)	1. genus_acacia_NN_1: large genus of shrubs and trees and some woody vines... 2. genus_sesbania_NN_1: *as above* 3. genus_dalbergia_NN_1: large genus of tropical trees having pinnate leaves and paniculate flowers...
III. *Our model succeeds but SimKGC fails*	
Head Entity Relation Tail Entity	wive_VB_1: take (someone) as a wife hypernym wed_VB_1: take in marriage
SimKGC (Rank 4)	1. wifely_JJ_1: befitting or characteristic of a wife 2. shack_up_VB_1: share living quarters; people who are not married and live together as a couple 3. wive_VB_2: marry a woman, take a wife
Ours (Rank 1)	1. **wed_VB_1: *as above*** 2. wifely_JJ_1: *as above* 3. shack_up_VB_1: *as above*

Table 10: Predictions of tail entities by the SimKGC baseline and our best model. The top-3 ranked outputs are reported, and the ranks of the correct results are also included. Correct predictions are in **bold**, and underline means predictions that do not exist in original KGs but are correct based on human judgment.

Semantic Frame Induction with Deep Metric Learning

Kosuke Yamada[1] **Ryohei Sasano**[1,2] **Koichi Takeda**[1]
[1]Graduate School of Informatics, Nagoya University, Japan
[2]RIKEN Center for Advanced Intelligence Project, Japan
`yamada.kosuke.v1@s.mail.nagoya-u.ac.jp`,
`{sasano,takedasu}@i.nagoya-u.ac.jp`

Abstract

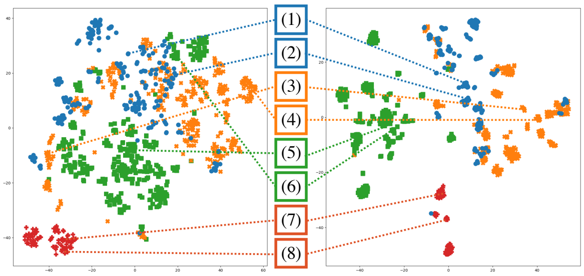
Recent studies have demonstrated the usefulness of contextualized word embeddings in unsupervised semantic frame induction. However, they have also revealed that generic contextualized embeddings are not always consistent with human intuitions about semantic frames, which causes unsatisfactory performance for frame induction based on contextualized embeddings. In this paper, we address *supervised* semantic frame induction, which assumes the existence of frame-annotated data for a subset of predicates in a corpus and aims to build a frame induction model that leverages the annotated data. We propose a model that uses deep metric learning to fine-tune a contextualized embedding model, and we apply the fine-tuned contextualized embeddings to perform semantic frame induction. Our experiments on FrameNet show that fine-tuning with deep metric learning considerably improves the clustering evaluation scores, namely, the B-CUBED F-SCORE and PURITY F-SCORE, by about 8 points or more. We also demonstrate that our approach is effective even when the number of training instances is small.

1 Introduction

Semantic frames are knowledge resources that reflect human intuitions about various concepts such as situations and events. One of the most representative semantic frame resources is FrameNet (Baker et al., 1998; Ruppenhofer et al., 2016), which consists of semantic frames, lexical units (LUs) that evoke these frames, and collections of frame-annotated sentences. Semantic frame induction is the task of grouping predicates, typically verbs, according to the semantic frames they evoke.[1] For example, given the verbs in the example sentences listed in Table 1, semantic frame induction aims to group them into four clusters according to the frames that they evoke.

Frame	Example sentence
FILLING	(1) She covered her mouth with her hand. (2) I filled a notebook with my name.
PLACING	(3) You can embed graphs in your worksheet. (4) He parked the car at the hotel.
REMOVING	(5) Volunteers removed grass from the marsh. (6) They'd drained the drop from the teapot.
TOPIC	(7) Each database will cover a specific topic. (8) Chapter 8 treats the educational advantages.

Table 1: Example sentences and the frames that their verbs evoke in FrameNet.



(a) Vanilla BERT (b) Fine-tuned BERT w/ AdaCos

Figure 1: 2D t-SNE projections of the contextualized embeddings of verbs annotated with either the FILLING (●), PLACING (×), REMOVING (■), or TOPIC (+) frames in FrameNet, by using (a) vanilla BERT or (b) fine-tuned BERT with AdaCos. The numbers in parentheses correspond to the examples listed in Table 1.

Recent studies (Arefyev et al., 2019; Anwar et al., 2019; Ribeiro et al., 2019) have demonstrated the usefulness of contextualized word embeddings such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) in unsupervised semantic frame induction. Figure 1(a) shows a 2D t-SNE (Maaten and Hinton, 2008) projection of the vanilla BERT[2] embeddings of verbs extracted from frame-annotated sentences in FrameNet. We can confirm that the instances of the verb "cover" in Exam-

---

[1] Strictly speaking, this task can be divided into two subtasks: verb clustering, which groups verbs according to the frames that they evoke, and argument clustering, which groups arguments of verbs according to their roles. In this study, we focus the former, the verb clustering task.

[2] We refer to BERT without fine-tuning as "vanilla BERT."

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1833–1845
May 2-6, 2023 ©2023 Association for Computational Linguistics

ples (1) and (7) are far apart in the space, which reflects the distance between their meanings. In contrast, "cover" in (7) and "treat" in (8), which are annotated with the same TOPIC frame, are close together. However, instances are not always properly placed in the semantic space according to the frames they evoke. For example, "remove" in (5) and "drain" in (6) are annotated with the same REMOVING frame but are not close together. This suggests that contextualized word embeddings are not always consistent with human intuition about semantic frames.

Hence, in this study, we tackle *supervised* semantic frame induction, which assumes the existence of annotated data for certain predicates, to induce semantic frames that adequately reflect human intuition about the frames. We propose methods that use deep metric learning to fine-tune the contextual word embedding model so that instances of verbs that evoke the same frame are placed close together and other instances are placed farther apart in the semantic space. Figure 1(b) shows the 2D t-SNE projection of BERT embeddings after fine-tuning with AdaCos (Zhang et al., 2019), which is a representative deep metric learning method. We can confirm that predicates that evoke the same frame are close together, such as those in (3) and (4) and those in (5) and (6). This suggests that deep metric learning enables fine-tuning of BERT to obtain embedding spaces that better reflect human intuition about semantic frames.

2 Related Work

For automatic construction of semantic frame resources, studies on grouping predicates according to the semantic frames they evoke can be divided into two groups: those that work on **semantic frame identification**, in which predicates are classified into predefined frames; and those that work on **semantic frame induction**, in which predicates are grouped according to the frames that they evoke, which are typically not given in advance.

Semantic frame identification is often treated as a subtask of frame semantic parsing (Das et al., 2014; Swayamdipta et al., 2017), and the methods using contextualized embedding have become mainstream. For example, Jiang and Riloff (2021) used a BERT-based model to generate representations for frames and LUs by using their formal definitions. Su et al. (2021) used a BERT-based model with a context encoder, to encode the context surrounding the frame-evoking word, and a frame encoder, to encode the frames' definitions and semantic roles. Yong and Torrent (2020) treated semantic frame identification as a clustering task.[3] They first excluded predicates that evoke frames that are not included in FrameNet by applying an anomaly detection model; then, they grouped the remaining predicates according to their meanings by using contextualized embeddings of predicates and sentence embeddings of the frame definitions.

Semantic frame induction is the task of grouping predicates in texts according to the frames they evoke. Instead of frames being given in advance, each grouping of given predicates is considered a frame. As with semantic frame identification, methods using contextualized embedding have become mainstream. Arefyev et al. (2019) first performed agglomerative clustering by using the BERT embedding of a frame-evoking verb and then split each cluster into two on the basis of the verb's substitutes. Anwar et al. (2019) used the embedding of the frame-evoking verb and the average word embedding of all the words in a sentence, as obtained by skip-gram (Mikolov et al., 2013) or ELMo and then performed agglomerative clustering. Ribeiro et al. (2019) applied graph clustering by using the ELMo embedding of the frame-evoking verb. Yamada et al. (2021a) leveraged the embedding of the masked frame-evoking verb and performed two-step clustering, which comprised intra-verb and cross-verb clustering. Yamada et al. (2021b) investigated how well contextualized word representations can recognize the difference of frames that the same verb evokes, and explored which types of representation are suitable for semantic frame induction. All of these studies focused on *unsupervised* semantic frame induction, with no training data. In contrast, in this study, we assume the existence of frame-annotated data for a subset of predicates appearing in a corpus, and work on *supervised* semantic frame induction.

3 Supervised Semantic Frame Induction

3.1 Task Description

The task of supervised semantic frame induction assumes the existence of frame-annotated data for a subset of a corpus's predicates, and it aims to

---

[3] Although they called their task semantic frame induction, we refer to the task as frame identification mainly due to the fact that their experiments excluded predicates that evoke unknown frames.

build a frame induction model that leverages the annotated data. Clustering-based methods are generally used for semantic frame induction, and this is also true for supervised semantic frame induction, where the annotated data is used to learn the distance metric for clustering. In this study, the predicates that are used for training the metric and for testing do not overlap. Note that, because different predicates may evoke the same frame, instances in the test data include predicates that evoke frames that are present in the training data.

3.2 Baseline Methods

For the simplest baseline, we use a one-step clustering-based method with contextualized embedding. The clustering method is group-average clustering based on the Euclidean distance. We also leverage the masked word embeddings and two-step clustering proposed by Yamada et al. (2021a). Regarding the former, we use a weighted average embedding (v_{w+m}) of the standard contextualized embedding of the frame-evoking word (v_{word}) and the masked word embedding (v_{mask}), which is a contextualized embedding of the frame-evoking word replaced by a special token "[MASK]." The embedding v_{w+m} is defined using a weight parameter α as follows:

$$v_{w+m} = (1-\alpha) \cdot v_{word} + \alpha \cdot v_{mask}. \quad (1)$$

Two-step clustering performs clustering for each frame-evoking word with the same lemma,[4] and it performs clustering over different frame-evoking words in the second step. We use X-means (Pelleg and Moore, 2000) for the first step and group-average clustering based on the Euclidean distance for the second step. All other settings here are the same as in Yamada et al. (2021a).

3.3 Fine-tuning by Deep Metric Learning

For supervised semantic frame induction, we fine-tune contextualized word embedding models by applying deep metric learning (Kaya and Bilge, 2019; Musgrave et al., 2020) so that the instances of predicates that evoke the same frame are closer together and those of predicates that evoke different frames are further apart. We apply two representative deep metric learning approaches: a distance-based approach and a classification-based approach.

Distance-based Approach This is a classical deep metric learning approach, and the models typically use multiple encoders to train the distance between a pair of instances. In this approach, we use two losses, a contrastive loss and a triplet loss, to build frame induction models.

The contrastive loss (Hadsell et al., 2006) is used to train the distance between a pair of instances by using a network of two encoders with shared parameters. Specifically, the model is trained to keep instances of the same class close together and instances of different classes separated by a certain margin. The loss is defined as follows:

$$L_{\text{cont}} = \begin{cases} D\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) & i = j \\ \max\left(m - D\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right), 0\right) & i \neq j \end{cases}, \quad (2)$$

where $\boldsymbol{x}_i$ denotes an embedding of an instance belonging to the i-th class, m denotes a margin, and D denotes a distance function, which is generally the squared Euclidean distance.

The triplet loss (Weinberger and Saul, 2009) is used for training such that, for a triplet of instances, the distance between the anchor and negative instances, which are from different classes, is more than a certain margin greater than the distance between the anchor and positive instances, which are from the same class. The loss is defined as follows:

$$L_{\text{tri}} = \max\left(D\left(\boldsymbol{x}_a, \boldsymbol{x}_p\right) - D\left(\boldsymbol{x}_a, \boldsymbol{x}_n\right) + m, 0\right), \quad (3)$$

where $\boldsymbol{x}_a$, $\boldsymbol{x}_p$, and $\boldsymbol{x}_n$ denote embeddings of the anchor, positive, and negative instances, respectively, and m and D are the same as in Equation (2).

We create pairs for each instance in the training set by randomly selecting instances of predicates that evoke the same frame as positives and instances of predicates that evoke different frames as negatives. The margin to keep the negatives away is determined by the development set.

Classification-based Approach This is an approach that has recently become the standard for face recognition. It basically uses a network that has an encoder to obtain instance embeddings and a linear layer for multiclass classification. This is superior to the distance-based approach in that it does not require a sampling algorithm and saves memory because it uses only a single encoder. The loss function is based on the softmax loss:

$$L_{\text{soft}} = -\log \frac{e^{\boldsymbol{w}_i^\top \boldsymbol{x}_i + b_i}}{\sum_{j=1}^{n} e^{\boldsymbol{w}_j^\top \boldsymbol{x}_i + b_j}}, \quad (4)$$

---

[4] Following Yamada et al. (2021a), we refer to each cluster generated by clustering in the first step as a pseudo-LU (pLU).

where $\boldsymbol{x}_i$, $\boldsymbol{w}_i$, and b_i denote an embedding of the instance, the linear layer's weight, and a bias term, respectively, for the i-th class, and n denotes the number of classes.

Many losses used in face recognition have been adjusted by introducing different margins for the softmax loss (Liu et al., 2017; Wang et al., 2018; Deng et al., 2019). These losses typically remove the bias term b_i of the softmax loss and transform the logit as $\boldsymbol{w}_i^\top \boldsymbol{x}_i = ||\boldsymbol{w}_i|| \cdot ||\boldsymbol{x}_i|| \cdot \cos\theta_i$, where θ_i is the angle between $\boldsymbol{w}_i$ and $\boldsymbol{x}_i$. ArcFace (Deng et al., 2019) has become a popular choice because of its superior geometric interpretation. It applies l_2 regularization to $\boldsymbol{w}_i$ and $\boldsymbol{x}_i$ and introduces an angular margin m and a feature scale s as hyperparameters to simultaneously enhance the intra-class compactness and inter-class discrepancy. The ArcFace loss is defined as follows:

$$L_{\text{arc}} = -\log \frac{e^{s \cdot \cos(\theta_i + m)}}{e^{s \cdot \cos(\theta_i + m)} + \sum_{j=1, j \neq i}^{n} e^{s \cdot \cos \theta_j}}. \quad (5)$$

Zhang et al. (2019) pointed out that the performance of these losses depends on the hyperparameters and they observed the behaviors of the angular margin and the feature scale. As a result, they proposed the hyperparameter-free AdaCos loss, which removes the margin and applies the scale dynamically. The AdaCos loss is defined as follows:

$$L_{\text{ada}} = -\log \frac{e^{\tilde{s} \cdot \cos \theta_i}}{\sum_{j=1}^{n} e^{\tilde{s} \cdot \cos \theta_j}}, \quad (6)$$

where $\tilde{s}$ denotes the automatically tuned scale.

While the softmax and AdaCos losses do not require a hyperparameter search, ArcFace requires hyperparameters for the margin and feature scale. Here, we explore only the margin because Zhang et al. (2019) showed that the behavior of the margin and the scale are similar and the distance-based approach explores the margin.

4 Experiment

To evaluate the usefulness of fine-tuning with deep metric learning, we experimented with supervised semantic frame induction by comparing previous non-fine-tuned models to various fine-tuned models ranging from typical to evolved ones. By varying the number of training instances, we also verified that our models were effective even for training a small number of instances.

	#Verbs	#LUs	#Frames	#Instances
Set 1	831	1,277	429	28,314
Set 2	831	1,261	415	26,179
Set 3	830	1,280	459	28,117
All	2,492	3,818	642	82,610

Table 2: Statistics of the FrameNet-based dataset used in three-fold cross-validation.

4.1 Settings

Dataset The dataset in our experiments was created by extracting example sentences in which the frame-evoking word was a verb from the FrameNet 1.7 dataset.[5] These example sentences were split into three sets such that sentences with the same verb were in the same set. The proportions of polysemous verbs were equal. We performed three-fold cross-validation with the three sets as the training, development, and test sets. Table 2 lists the dataset statistics. Note that the verbs, LUs, and instances did not overlap among the sets, but the frames did overlap. The training set was used to fine-tune the contextualized word embeddings. The development set was used to determine the criterion for the number of clusters and the weight α of the embedding $v_{\text{W+M}}$, as well as the margin for the contrastive, triplet, and ArcFace losses. The range of α was from 0 to 1 in increments of 0.1, and the candidates of the margin were 0.1, 0.2, 0.5, and 1.0 for the contrastive and triplet losses and 0.01, 0.02, 0.05, and 0.1 for the ArcFace loss.

Comparison Methods We used BERT[6] from Hugging Face (Wolf et al., 2020) to obtain contextualized word embeddings. We compared 12 methods, which comprised the vanilla model (**Vanilla**) and five fine-tuned models (**Contrastive**, **Triplet**, **Softmax**, **ArcFace**, **AdaCos**) with one-step clustering and two-step clustering. All embeddings were processed with l_2 normalization. Regarding hyperparameters, the batch size was 32, the learning rate was 1e-5, and the number of epochs for fine-tuning was five. Also, the feature scale for ArcFace was 64. The optimization algorithm was AdamW (Loshchilov and Hutter, 2017).

We compared our methods with the three unsupervised methods used in Subtask-A of SemEval-2019 Task 2 (QasemiZadeh et al., 2019) in addition to the method of Yamada et al. (2021a) that cor-

---

[5] `https://framenet.icsi.berkeley.edu/`

[6] `https://huggingface.co/bert-base-uncased`

Clustering	Model	α	#pLU	#C	Pu / iPu / PiF	BcP / BcR / BcF
One-step clustering (group-average clustering)	Vanilla	0.00	–	429	53.0 / 57.0 / 54.9	40.8 / 44.6 / 42.6
	Contrastive	0.13	–	443	56.9 / 70.0 / 62.8	45.1 / 58.6 / 51.0
	Triplet	0.23	–	425	70.0 / 77.0 / 73.3	**60.3** / 68.1 / **63.9**
	Softmax	0.23	–	440	65.1 / 78.0 / 71.0	53.3 / 68.6 / 59.9
	ArcFace	0.37	–	436	**70.3** / 76.2 / 73.1	59.7 / 67.4 / 63.3
	AdaCos	0.30	–	446	69.0 / **78.7** / **73.5**	57.5 / **69.5** / 62.9
Two-step clustering (X-means & group-average clustering)	Vanilla	0.67	877	444	60.6 / 74.9 / 66.9	49.7 / 65.8 / 56.5
	Contrastive	0.23	1904	689	69.2 / 62.5 / 65.7	59.5 / 50.9 / 54.8
	Triplet	0.50	1014	454	73.4 / **76.7** / 74.8	64.6 / **68.0** / 66.0
	Softmax	0.43	1428	919	**84.7** / 62.5 / 71.9	**78.4** / 50.4 / 61.4
	ArcFace	0.47	955	452	70.5 / 76.5 / 73.3	60.8 / 67.7 / 63.8
	AdaCos	0.50	1128	656	80.8 / 71.3 / **75.6**	73.2 / 60.9 / **66.2**

Table 3: Experimental results on semantic frame induction with vanilla and fine-tuned models over three-fold cross-validation. Each value in the table is the average from three trials. #pLU denote the number of pLUs, and #C denotes the final number of clusters.

responds to our method with the Vanilla model.[7] Regarding those three methods, first, Arefyev et al. (2019) performed group-average clustering by using the BERT embedding of a frame-evoking verb and then split each cluster into two by using TF-IDF features based on the verb's substitutes, which were generated using Hearst-like patterns (Hearst, 1992) with BERT. Second, Anwar et al. (2019) performed group-average clustering through concatenation of the embedding of the frame-evoking verb and the average embedding of all words in the sentence as obtained by skip-gram. Third, Ribeiro et al. (2019) applied graph clustering by Chinese whispers (Biemann, 2006) with the ELMo embedding of the frame-evoking verb.

Evaluation Metrics For evaluation metrics, we used the Purity (Pu), the inverse Purity (iPu), and their harmonic mean, the F-score (PiF) (Zhao and Karypis, 2001); and the B-cubed Precision (BcP), the B-cubed Recall (BcR), and their harmonic mean, the F-score (BcF) (Bagga and Baldwin, 1998). Pu is a metric of the degree to which a cluster is occupied by a single label, while iPu is a metric of the degree to which a single label is concentrated in a single cluster. BcP and BcR evaluate the precision and recall for each sample, respectively, without associating clusters and labels. The shared task at SemEval-2019 Task 2 ranked systems according to BcF.

Method	PiF	BcF
Arefyev et al. (2019)	65.5	57.4
Anwar et al. (2019)	62.2	52.2
Ribeiro et al. (2019)	58.2	46.8
Yamada et al. (2021a)	66.9	56.5
Ours (one-step & Triplet)	73.9	63.9
Ours (one-step & AdaCos)	73.5	62.9
Ours (two-step & Triplet)	74.8	66.0
Ours (two-step & AdaCos)	**75.6**	**66.2**

Table 4: Results of comparison with previous methods.

4.2 Results

Table 3 summarizes the experimental results with the 12 methods. The fine-tuned models, especially the Triplet, ArcFace, and AdaCos models, obtained higher PiF and BcF scores than the Vanilla model except for the two-step clustering method with the Contrastive model. The reason why the Contrastive model performed worse than the other fine-tuned models could be that the space that represents the frame does not match the cluster size due to train the distance according to a fixed margin. Table 4 also lists the comparison results with previous methods. Our fine-tuning methods achieved higher PiF and BcF scores than the previous methods. These results indicate that fine-tuning with deep metric learning helps to improve the performance of semantic frame induction.

From Table 3, we can see that the two-step clustering methods tended to obtain higher overall scores than the one-step clustering methods. How-

---

[7] The SemEval-2019 Task 2 dataset is no longer available, as described on its official web page; thus, we excluded this dataset from the experiments.

Clustering	Model	PIF	BCF
		1 / 2 / 5 / 10 / all	1 / 2 / 5 / 10 / all
One-step clustering (group-average clustering)	Vanilla	54.9 / 54.9 / 54.9 / 54.9 / 54.9	42.6 / 42.6 / 42.6 / 42.6 / 42.6
	Contrastive	52.0 / 60.3 / 62.6 / 61.7 / 62.8	39.0 / 48.3 / 50.8 / 49.7 / 51.0
	Triplet	**68.2 / 70.9 / 71.7 / 72.9** / 73.3	**57.4 / 60.6 / 61.8 / 63.0 / 63.9**
	Softmax	52.9 / 54.7 / 64.9 / 69.4 / 71.0	39.3 / 41.7 / 52.8 / 57.8 / 59.9
	ArcFace	58.1 / 62.5 / 69.2 / 71.9 / 73.1	45.6 / 50.6 / 59.1 / 62.1 / 63.3
	AdaCos	57.5 / 59.7 / 66.5 / 70.9 / **73.5**	44.7 / 47.1 / 54.6 / 60.0 / 62.9
Two-step clustering (X-means & group-average clustering)	Vanilla	66.9 / 66.9 / 66.9 / 66.9 / 66.9	56.5 / 56.5 / 56.5 / 56.5 / 56.5
	Contrastive	68.4 / 66.3 / 67.7 / 67.4 / 65.7	58.4 / 56.0 / 57.7 / 57.1 / 54.8
	Triplet	**71.7 / 72.5 / 73.9** / 74.0 / 74.8	**62.4 / 63.1 / 64.8** / 64.9 / 66.0
	Softmax	67.6 / 70.1 / 72.6 / 73.0 / 71.9	57.7 / 60.6 / 63.2 / 63.4 / 61.4
	ArcFace	66.5 / 66.6 / 69.2 / 72.9 / 73.3	56.1 / 56.4 / 59.5 / 63.2 / 63.8
	AdaCos	67.3 / 69.4 / 73.4 / **74.3 / 75.6**	57.6 / 59.7 / 64.5 / **65.3 / 66.2**

Table 5: Experimental results on semantic frame induction using the vanilla and fine-tuned models over three-fold cross-validation when varying the maximum number of training instances. Each numeric column lists the number of instances per LU included in the training set. Each value in the table is the average from three trials.

ever, the difference in BCF scores between the one-step and two-step clustering methods with the Vanilla model was 13.9, whereas the difference in the maximum BCF scores for both clustering methods with the fine-tuned models was only 2.3. Thus, for fine-tuning models, one-step clustering is still a good option in addition to two-step clustering. Note that one-step clustering is more straightforward to implement than two-step clustering, but it requires more computation time[8] and CPU memory to cluster many instances at once. Regarding the weight α, two-step clustering tended to incorporate v_{mask} more than one-step clustering did in not only the Vanilla model but also the fine-tuned models. These results suggest that two-step clustering remains effective in masking a verb's surface information even after fine-tuning.

The balance between BCP and BCR in the clustering evaluation metric depends on the final number of frame clusters, #C in Table 3. In the extreme case, BCR is 1 if #C is 1, and BCP is 1 if #C is equal to the number of instances. Hence, among models with roughly the same BCF, those with a fewer number of clusters tend to have higher BCR. For example, as shown in Table 3, #C of Triplet in two-step clustering is 454, while that of AdaCos is 656, and we can confirm that Triplet, which has fewer clusters, obtains higher BCR than AdaCos.

4.3 Effect of Number of Training Instances

We found that fine-tuned methods outperformed previous unsupervised methods when the number of training instances was around 30,000. However, the annotation cost of building a resource like FrameNet is high, so the fewer instances used for training, the easier it is to build other language resources and apply them to other tasks. Thus, we experimented with varying the number of training instances. Specifically, for each LU in the training set, the maximum number of instances was varied among 1, 2, 5, 10, and all instances. The resulting average numbers of training instances for the three sets were 1,273, 2,445, 5,680, 10,053, and 27,536, respectively. The numbers of verbs, LUs, and frames were the same in each setting.

Table 5 lists the PIF and BCF scores for each method. Because the Vanilla model was not fine-tuned, its scores are the same in each setting. The Triplet model achieved high scores even with a small number of training instances. In the two-step clustering method with the Triplet model, the score difference between the cases of "1" and "all" is only 3.1 for PIF and 3.6 for BCF, even though the number of training instances is quite different, i.e., 1,273 vs. 27,536. These results show that even when a small number of examples is annotated for each meaning of a verb, this method can be expected to perform considerably better than unsupervised methods. In contrast, the Softmax, ArcFace, and AdaCos models obtained scores closer to the Triplet model in the cases of "5" or "10" but per-

---

[8] In our experiments with 16-core Intel Xeon Gold 6134 CPU at 3.20 GHz, the computation times were about 10 minutes for one-step clustering and about 5 minutes for two-step clustering.

formed considerably worse with an even smaller number of training instances. We conclude that the relatively poor performance of these models with a small number of training instances was due to insufficient training of the linear layer's weights.

5 Analysis of Fine-tuned Embedding

It is not easy to analyze the properties of an embedding in clustering evaluation because the performance depends on the clustering method and the number of clusters. To better understand the fine-tuned embeddings, we performed a similarity ranking evaluation and visualized the embeddings.

5.1 Similarity Ranking Evaluation

We evaluated the models by ranking instances according to their embedding similarity. Specifically, we took one verb instance as a query instance; then, we computed the cosine similarity of the embeddings between the query instance and the remaining verb instances and evaluated the similarity rankings of the instances in descending order. We used v_{w+m} with the same weight α that was used for the one-step clustering in Section 4. We chose recall as the metric to evaluate the instance distribution. This metric computes the average matching rate between true instances, which are instances of the same frame as the query instance, and predicted instances, which are obtained by extracting the same number of top-ranked instances as the number of true instances. For example, Set 1 of Table 2 had 153 instances of the FILLING frame out of 28,314 total instances. When one of these instances was the query instance, the number of true instances would be 152. Thus, from the total instances, we would extract the top 152 instances that were similar to the query, and if 114 instances were true instances, the score would be $114/152 = 0.75$.

We performed the similarity ranking evaluation in three settings with respect to the search space of the ranked instances: ALL, which included all instances, SAME, which included only instances of the same verb as the query, and DIFF, which included only instances of different verbs as the query. Table 6 lists the results. The results for ALL show that all of the fine-tuned models were improved over the Vanilla model; in particular, the four fine-tuned models besides the Contrastive model performed very well, improving by more than 20. We thus confirmed that instances of the same frame were trained to be close to each other and instances of different frames are trained to be distant from each other. Score improvements were observed for both SAME and DIFF, and as expected, SAME scored higher than DIFF both before and after fine-tuning. However, the improvement was much larger for DIFF than for SAME, suggesting that the improvement in clustering performance by fine-tuning was mainly due to the fact that different verbs evoking the same frame were trained to be close to each other.

Model	ALL	SAME	DIFF
Vanilla	35.9	68.5	17.4
Contrastive	46.6	67.7	28.4
Triplet	60.6	74.2	40.7
Softmax	60.5	73.1	41.4
ArcFace	58.8	**75.6**	37.4
AdaCos	**62.1**	74.7	**42.8**

Table 6: Experimental results of recall for the similarity ranking evaluation over three-fold cross-validation. ALL, SAME, and DIFF indicate a search space of all instances, instances of the same verb, and instances of different verbs, respectively, for the query instance.

Model	OVERLAP	NON-OVERLAP
Vanilla	33.9	47.3
Contrastive	45.5	52.2
Triplet	59.0	**69.0**
Softmax	59.7	65.0
ArcFace	57.1	68.4
AdaCos	**61.2**	67.0

Table 7: Separately aggregated scores for ALL in Table 6 in the OVERLAP and NON-OVERLAP cases.

It is important to further examine whether the improved performance might have resulted only from the frames included in the training set. That is, we need to verify that the embedding of an instance of an untrained frame could be associated with a correct frame. To investigate this, we aggregated the scores separately for cases in which the frames of the query instance were included in the training set (OVERLAP) and for cases in which they were not (NON-OVERLAP). Table 7 lists the separately aggregated results for ALL in Table 6. All of the fine-tuned models obtained higher scores than the Vanilla model for not only frames that were in the training set but also frames that were not. Note that the scores for NON-OVERLAP were higher overall than those for OVERLAP. This result may be counterintuitive, but the reason is that the frames

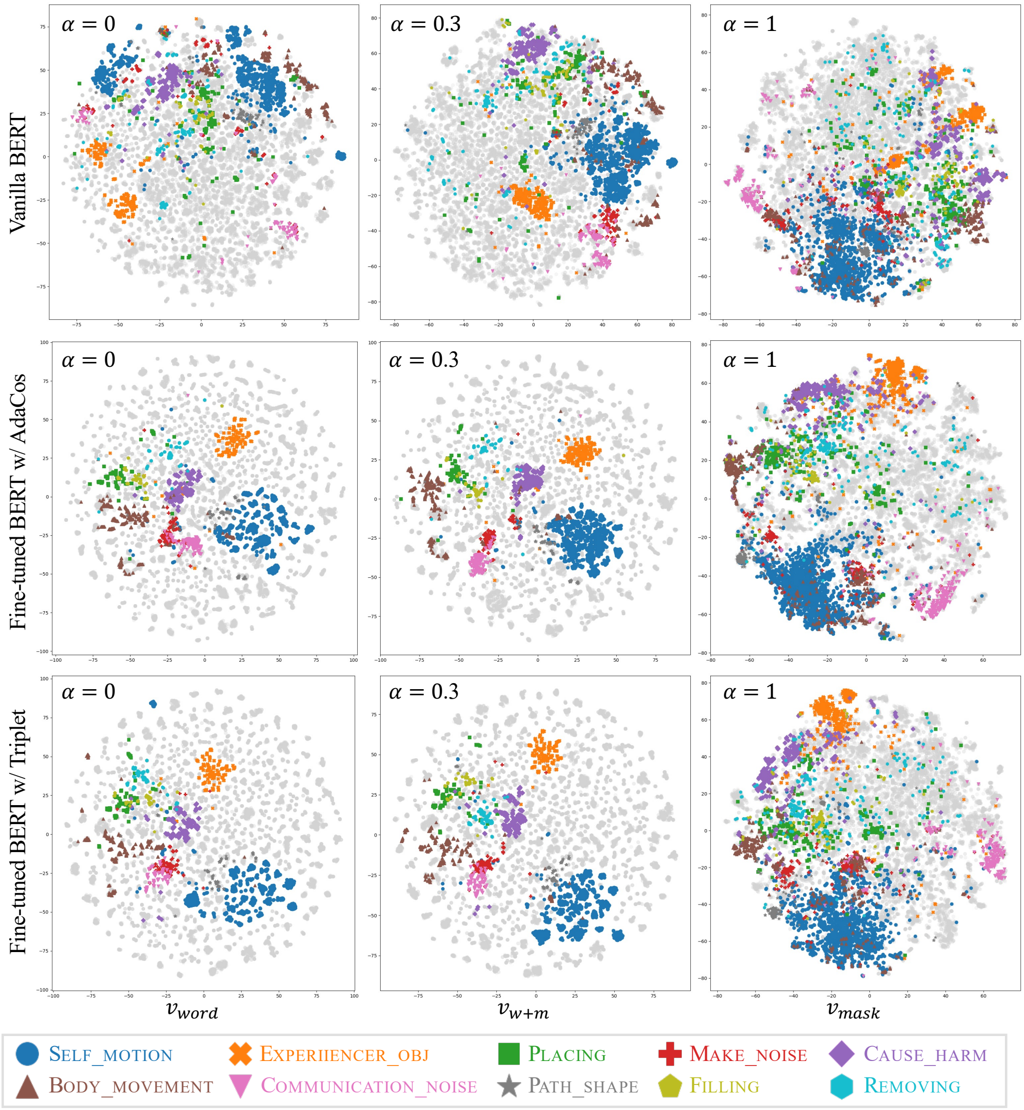


Figure 2: 2D t-SNE projections of v_{word}, v_{w+m}, and v_{mask} for the Vanilla, AdaCos, and Triplet models, respectively, for all instances with Set 1 in Table 2 as the test set. The top 10 semantic frames with the highest numbers of instances in this set are highlighted.

in the NON-OVERLAP case were only evoked by a few verbs, making it relatively easy to obtain higher ranking of instances of the same frame as the query.

5.2 Embedding Visualization

To intuitively understand the embeddings given by the Vanilla model and two fine-tuned models, we visualized them by t-SNE. Figure 2 shows the two-dimensional t-SNE projection of the contextualized embeddings of the frame-evoking verbs for all instances when Set 1 in Table 2 is the test set. We used v_{word}, v_{w+m}, and v_{mask} for the Vanilla, AdaCos, and Triplet models, respectively. The weight α for v_{w+m} was 0.3, which was the best value for one-step clustering methods with the Triplet and AdaCos models in Section 4.[9] We highlight the top ten semantic frames with the highest numbers of instances in this set.

In the Vanilla model, the instances for v_{word} tended to be grouped by frame but were not sufficiently grouped into clusters. For example, the instances of the SELF_MOTION frame were divided into two large groups, while those of the REMOVING frame were scattered. The instances for v_{mask} were somewhat more scattered than those for v_{word}. In addition, v_{w+m} tended to group instances of the same frame.

In the AdaCos and Triplet models, the instances for v_{word} were grouped much better for each frame

---

[9] For the Vanilla model, α was 0, which was the same as for v_{w+m} and v_{word}; thus, the results here are for α of 0.3.

than those for non-fine-tuned v_{word}. The results also confirmed that instances of frames with similar meanings, such as the PLACING and FILLING frames, were both identifiable and close. However, fine-tuned v_{word} formed many lumps of instances. This suggests that deep metric learning incorporates too much of a verb's surface information. On the other hand, fine-tuned v_{mask} was somewhat better than non-fine-tuned v_{mask}, but not as good as fine-tuned v_{word}. As deep metric learning may require the surface information about a verb to be induced, so fine-tuned v_{mask} may not work well. The instances in fine-tuned v_{w+m} were better grouped than those for fine-tuned v_{word}, because instances of the same frame were more grouped.

6 Conclusion

We worked on the supervised semantic frame induction, and we proposed a model that uses deep metric learning to fine-tune a contextualized embedding model and applied the fine-tuned contextualized embeddings to perform semantic frame induction. In our experiments, we showed that fine-tuned BERT models with the triplet, ArcFace, and AdaCos losses are quite promising for semantic frame induction, as the human intuition in developing semantic frames such as those in FrameNet can be well captured by deep metric learning. In particular, the fine-tuned BERT model with the triplet loss performed considerably better than vanilla BERT even when the number of training instances was small; accordingly, the fine-tuned model is expected to have a wide range of applications. We also found that the one-step clustering can be a good choice in addition to two-step clustering when performing fine-tuning.

The ultimate goal of this study is to automatically construct semantic frame knowledge from large text corpora. This goal requires not only grouping the verbs according to the frames that they evoke but also grouping their arguments according to the frame element roles that they fill. Our proposed fine-tuned contextualized word embedding with deep metric learning could be effective for clustering arguments as it is for clustering verbs. We would like to explore how to achieve this goal.

Limitations

In this study, we only conducted experiments with English FrameNet, so it is unclear how useful this method will be for other corpora and multilingual resources. However, since our method does not depend on the properties of the specific corpus and language, it is quite possible that fine-tuning would improve the scores in other datasets. In addition, as our method requires supervised data from a semantic frame knowledge resource, some annotation will be necessary when applying the method to other languages that lack such a resource.

Acknowledgements

This work was supported by JST FOREST Program, Grant Number JPMJFR216N and JSPS KAKENHI Grant Numbers 21K12012 and 22J14993.

References

Saba Anwar, Dmitry Ustalov, Nikolay Arefyev, Simone Paolo Ponzetto, Chris Biemann, and Alexander Panchenko. 2019. HHMM at SemEval-2019 task 2: Unsupervised frame induction using contextualized word embeddings. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval 2019)*, pages 125–129.

Nikolay Arefyev, Boris Sheludko, Adis Davletov, Dmitry Kharchev, Alex Nevidomsky, and Alexander Panchenko. 2019. Neural GRANNy at SemEval-2019 task 2: A combined approach for better modeling of semantic relationships in semantic frame induction. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval 2019)*, pages 31–38.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-COLING 1998)*, pages 79–85.

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-COLING 1998)*, pages 86–90.

Chris Biemann. 2006. Chinese whispers: An efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing (TextGraphs 2006)*, pages 73–80.

Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. ArcFace: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, pages 4690–4699.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4171–4186.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, volume 2, pages 1735–1742.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING 1992)*.

Tianyu Jiang and Ellen Riloff. 2021. Exploiting definitions for frame identification. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2021)*, pages 2429–2434.

Mahmut Kaya and Hasan Şakir Bilge. 2019. Deep metric learning: A survey. *Symmetry*, 11(9):1066.

Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. SphereFace: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, pages 212–220.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS 2013)*, pages 3111–3119.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. 2020. A metric learning reality check. In *Proceedings of the 16th European Conference on Computer Vision (ECCV 2020)*, pages 681–699.

Dan Pelleg and Andrew Moore. 2000. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pages 727–734.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 2227–2237.

Behrang QasemiZadeh, Miriam R. L. Petruck, Regina Stodden, Laura Kallmeyer, and Marie Candito. 2019. SemEval-2019 task 2: Unsupervised lexical frame induction. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval 2019)*, pages 16–30.

Eugénio Ribeiro, Vânia Mendonça, Ricardo Ribeiro, David Martins de Matos, Alberto Sardinha, Ana Lúcia Santos, and Luísa Coheur. 2019. L2F/INESC-ID at SemEval-2019 task 2: Unsupervised lexical semantic frame induction using contextualized word representations. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval 2019)*, pages 130–136.

Josef Ruppenhofer, Michael Ellsworth, Myriam Schwarzer-Petruck, Christopher R Johnson, and Jan Scheffczyk. 2016. *FrameNet II: Extended theory and practice*. International Computer Science Institute.

Xuefeng Su, Ru Li, Xiaoli Li, Jeff Z. Pan, Hu Zhang, Qinghua Chai, and Xiaoqi Han. 2021. A knowledge-guided framework for frame identification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*, pages 5230–5240.

Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A Smith. 2017. Frame-semantic parsing with softmax-margin segmental RNNs and a syntactic scaffold. arXiv preprint arXiv:1706.09528.

Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. 2018. CosFace: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, pages 5265–5274.

Kilian Q Weinberger and Lawrence K Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(2).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP 2020)*, pages 38–45.

Kosuke Yamada, Ryohei Sasano, and Koichi Takeda. 2021a. Semantic frame induction using masked word embeddings and two-step clustering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*, pages 811–816.

Kosuke Yamada, Ryohei Sasano, and Koichi Takeda. 2021b. Verb sense clustering using contextualized word representations for semantic frame induction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021 (ACL-IJCNLP 2021 Findings)*, pages 4353–4362.

Zheng Xin Yong and Tiago Timponi Torrent. 2020. Semi-supervised deep embedded clustering with anomaly detection for semantic frame induction. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*, pages 3509–3519.

Xiao Zhang, Rui Zhao, Yu Qiao, Xiaogang Wang, and Hongsheng Li. 2019. AdaCos: Adaptively scaling cosine logits for effectively learning deep face representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, pages 10823–10832.

Ying Zhao and George Karypis. 2001. Criterion functions for document clustering: Experiments and analysis. Technical report, Retrieved from the University of Minnesota Digital Conservancy.

A Clustering Results with and without Linear Completion

Tables 8 and 9 list our experimental results for semantic frame induction when using v_{word}, v_{w+m}, and v_{mask} in one-step and two-step clustering, respectively. The results show that v_{w+m} tended to perform better than v_{word} and v_{mask}, thus demonstrating the usefulness of linear completion. This tendency was noticeable for two-step clustering but more limited for one-step clustering.

Regarding the results for v_{word} and v_{mask}, the fine-tuning was effective for v_{word}, as the scores improved considerably, but the effectiveness was limited for v_{mask}. This was probably because the embedding of the special token “[MASK],” which was the source of the contextualized word embedding, was shared by all instances.

B Results for Semantic Frame Induction on Development Set

Table 10 lists our experimental results for semantic frame induction on the development set. As with the test set, the fine-tuned models obtained higher PIF and BCF scores than the Vanilla model except for two-step clustering with the Contrastive model. In particular, the Triplet, ArcFace, and AdaCos models obtained high scores for both one-step and two-step clusterings.

C Embedding Visualization of Remaining Models

In Figure 2, we showed a two-dimensional t-SNE projection of v_{word}, v_{w+m}, and v_{mask} for the Vanilla, AdaCos, and Triplet models, respectively. Figure 3 shows a two-dimensional t-SNE projection of v_{word}, v_{w+m}, and v_{mask} for the remaining models not included in Figure 2, namely, the Contrastive, Softmax, and ArcFace models, with the same setting. Figure 3 confirms that the three fine-tuned models, as well as the two fine-tuned models shown in Figure 2, are more coherent semantically than the Vanilla model, and the tendency of v_{word}, v_{w+m}, and v_{mask} is similar. In addition, for the Contrastive model, whose performance was relatively poor among the fine-tuning models in Table 3, it was confirmed that the instances were somewhat scattered compared to the Softmax and ArcFace models. For example, the instances of the BODY_MOVEMENT frame were scattered. This result confirmed the consistency of scores and visualization. The results show consistency between scores and visualization.

Clustering	Model	α	#pLU	#C	PU / IPU / PIF	BCP / BCR / BCF
One-step clustering (group-average clustering)	Vanilla	0.00	–	429	53.0 / 57.0 / 54.9	40.8 / 44.6 / 42.6
	Contrastive	0.00	–	421	55.4 / 69.6 / 61.7	43.7 / 58.2 / 49.9
	Triplet	0.00	–	424	**69.9** / 76.6 / **73.1**	60.3 / 67.7 / **63.7**
	Softmax	0.00	–	433	65.1 / 77.4 / 70.7	53.3 / 67.9 / 59.7
	ArcFace	0.00	–	430	69.8 / 75.1 / 72.3	**62.7** / 66.1 / 62.7
	AdaCos	0.00	–	434	68.2 / **77.6** / 72.6	62.0 / **68.1** / 62.0
One-step clustering (group-average clustering)	Vanilla	0.00	–	429	53.0 / 57.0 / 54.9	40.8 / 44.6 / 42.6
	Contrastive	0.13	–	443	56.9 / 70.0 / 62.8	45.1 / 58.6 / 51.0
	Triplet	0.23	–	425	70.0 / 77.0 / 73.3	**60.3** / 68.1 / **63.9**
	Softmax	0.23	–	440	65.1 / 58.0 / 71.0	53.3 / 68.6 / 59.9
	ArcFace	0.37	–	436	**70.3** / 76.2 / 73.1	59.7 / 67.4 / 63.3
	AdaCos	0.30	–	446	69.0 / **78.7** / **73.5**	57.5 / **69.5** / 62.9
One-step clustering (group-average clustering)	Vanilla	1.00	–	430	29.9 / 38.9 / 33.8	18.0 / 24.4 / 20.7
	Contrastive	1.00	–	449	26.1 / 40.8 / 31.8	14.3 / 26.2 / 18.4
	Triplet	1.00	–	442	**35.5** / 47.9 / **40.8**	**21.0** / 31.8 / **25.2**
	Softmax	1.00	–	447	26.1 / **50.6** / 37.4	15.5 / **35.3** / 21.5
	ArcFace	1.00	–	452	35.0 / 46.2 / 40.1	20.9 / 30.6 / 24.8
	AdaCos	1.00	–	432	33.7 / 48.0 / 39.1	18.3 / 32.1 / 23.3

Table 8: Results on semantic frame induction from using v_{word}, v_{w+m}, and v_{mask} with in one-step clustering.

Clustering	Model	α	#pLU	#C	PU / IPU / PIF	BCP / BCR / BCF
Two-step clustering (X-means & group-average clustering)	Vanilla	0.00	880	438	52.5 / 68.2 / 59.3	43.0 / 58.1 / 49.4
	Contrastive	0.00	2269	720	68.2 / 62.8 / 65.3	58.7 / 50.6 / 54.2
	Triplet	0.00	1277	500	73.3 / 71.0 / 71.8	64.8 / 60.8 / 62.2
	Softmax	0.00	1909	1107	**85.1** / 60.0 / 70.3	**78.7** / 47.8 / 59.3
	ArcFace	0.00	1139	438	70.9 / **73.3** / 72.0	61.2 / **63.8** / 62.4
	AdaCos	0.00	1664	897	83.6 / 64.7 / **72.7**	76.9 / 53.3 / **62.6**
Two-step clustering (X-means & group-average clustering)	Vanilla	0.67	877	444	60.6 / 74.9 / 66.9	49.7 / 65.8 / 56.5
	Contrastive	0.23	1904	689	69.2 / 62.5 / 65.7	59.5 / 50.9 / 54.8
	Triplet	0.50	1014	454	73.4 / **76.7** / 74.8	64.6 / **68.0** / 66.0
	Softmax	0.43	1428	919	**84.7** / 62.5 / 71.9	**78.4** / 50.4 / 61.4
	ArcFace	0.47	955	452	70.5 / 76.5 / 73.3	60.8 / 67.7 / 63.8
	AdaCos	0.50	1128	656	80.8 / 71.3 / **75.6**	73.2 / 60.9 / **66.2**
Two-step clustering (X-means & group-average clustering)	Vanilla	1.00	873	469	59.0 / 72.9 / 65.2	48.8 / 63.6 / 55.1
	Contrastive	1.00	2050	718	49.3 / 49.2 / 49.2	37.7 / 36.3 / 36.9
	Triplet	1.00	930	463	65.2 / **73.9** / **69.0**	55.2 / **64.3** / **59.1**
	Softmax	1.00	1573	873	**68.6** / 55.3 / 61.2	**58.6** / 43.5 / 49.9
	ArcFace	1.00	906	467	62.5 / 73.7 / 67.5	52.3 / **64.3** / 57.4
	AdaCos	1.00	1113	522	65.4 / 65.5 / 65.2	55.2 / 55.2 / 54.5

Table 9: Results on semantic frame induction from using v_{word}, v_{w+m}, and v_{mask} with two-step clustering.

Clustering	Model	α	#pLU	#C	PU / IPU / PIF	BCP / BCR / BCF
One-step clustering (group-average clustering)	Vanilla	0.00	–	433	53.1 / 57.1 / 55.0	40.9 / 44.7 / 42.7
	Contrastive	0.13	–	433	57.7 / 70.7 / 63.5	45.2 / 59.4 / 51.3
	Triplet	0.37	–	433	**71.7** / 76.9 / **74.2**	**62.0** / 67.4 / **64.6**
	Softmax	0.27	–	433	66.4 / 77.6 / 71.5	53.7 / 68.2 / 60.1
	ArcFace	0.20	–	433	70.8 / 76.1 / 73.3	60.6 / 67.3 / 63.8
	AdaCos	0.37	–	433	69.1 / **78.5** / 73.5	57.2 / **69.3** / 62.7
Two-step clustering (X-means & group-average clustering)	Vanilla	0.67	877	441	61.3 / 75.9 / 67.7	50.3 / 67.0 / 57.4
	Contrastive	0.23	1879	736	70.3 / 63.5 / 66.6	60.9 / 51.8 / 55.9
	Triplet	0.50	1016	431	74.7 / **78.1** / 76.4	65.7 / **69.6** / **67.6**
	Softmax	0.43	1442	922	**84.6** / 63.2 / 72.3	**78.1** / 51.0 / 61.7
	ArcFace	0.47	966	453	71.5 / 76.9 / 74.0	62.1 / 68.0 / 64.9
	AdaCos	0.50	1120	606	79.7 / 73.8 / **76.5**	71.7 / 63.7 / 67.3

Table 10: Experimental results for semantic frame induction on the development set.

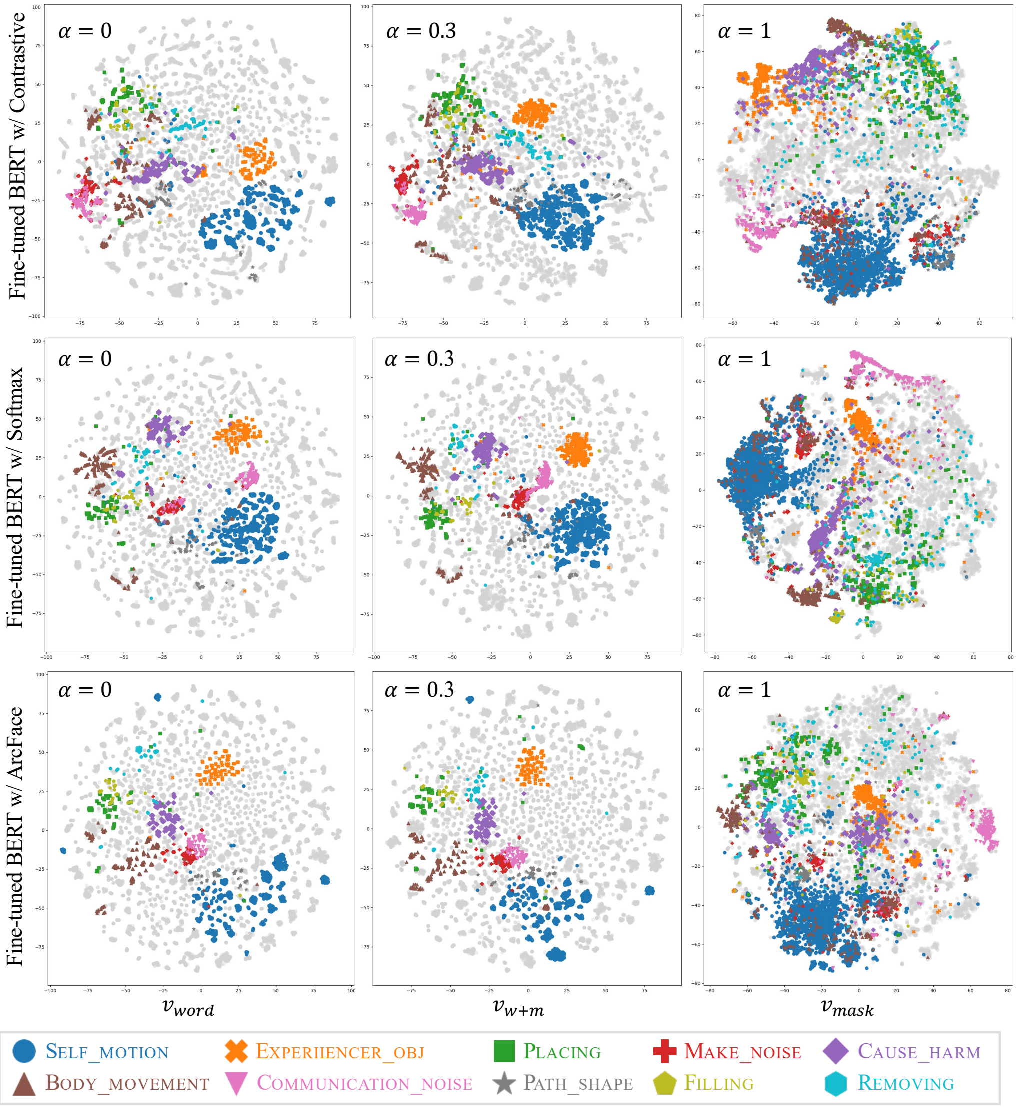


Figure 3: 2D t-SNE projections of v_{word}, v_{w+m}, and v_{mask} for the Contrastive, Softmax, and ArcFace models, respectively, for all instances with Set 1 in Table 2 as the test set. The top 10 semantic frames with the highest numbers of instances in this set are highlighted.

The Devil is in the Details: On Models and Training Regimes for Few-Shot Intent Classification

Mohsen Mesgar[1*] **Thy Thy Tran**[1*] **Goran Glavaš**[2] **Iryna Gurevych**[1]

[1]**Ubiquitous Knowledge Processing Lab (UKP)**
Department of Computer Science and Hessian Center for AI (hessian.AI)
Technical University of Darmstadt
[2]CAIDAS, University of Würzburg, Germany
www.ukp.tu-darmstadt.de

Abstract

In task-oriented dialog (ToD) new intents emerge on regular basis, with a handful of available utterances at best. This renders effective Few-Shot Intent Classification (FSIC) a central challenge for modular ToD systems. Recent FSIC methods appear to be similar: they use pretrained language models (PLMs) to encode utterances and predominantly resort to nearest-neighbor-based inference. However, they also differ in major components: they start from different PLMs, use different encoding architectures and utterance similarity functions, and adopt different training regimes. Coupling of these vital components together with the lack of informative ablations prevents the identification of factors that drive the (reported) FSIC performance. We propose a unified framework to evaluate these components along the following key dimensions: (1) *Encoding architectures*: Cross-Encoder vs Bi-Encoders; (2) *Similarity function*: Parameterized (i.e., trainable) vs non-parameterized; (3) *Training regimes*: Episodic meta-learning vs conventional (i.e., non-episodic) training. Our experimental results on seven FSIC benchmarks reveal three new important findings. First, the unexplored combination of cross-encoder architecture and episodic meta-learning consistently yields the best FSIC performance. Second, episodic training substantially outperforms its non-episodic counterpart. Finally, we show that splitting episodes into support and query sets has a limited and inconsistent effect on performance. Our findings show the importance of ablations and fair comparisons in FSIC. We publicly release our code and data[1].

1 Introduction

Intent classification deals with assigning one label from a predefined set of classes or *intents* to user utterances. This task is vital for task-oriented dialog (ToD) systems since the predicted intent of an utterance is an essential input to other modules (i.e., dialog management) in these systems (Ma et al., 2022; Louvan and Magnini, 2020; Razumovskaia et al., 2021). Although intent classification has been widely studied, it still represents a challenge in settings where dialogue systems, including their intent classifiers, need to have the ability to be quickly adjusted to new domains and intent classes. The main challenges in training intent classifiers in such settings lies in the costly labeling of utterances (Zhang et al., 2022a; Wen et al., 2017; Budzianowski et al., 2018; Rastogi et al., 2020; Hung et al., 2022; Mueller et al., 2022). Few-shot intent classification (FSIC), which deals with adjusting intent classifiers to new intents given only a handful of labeled instances, is thus of paramount importance for ToD systems.

Various methods (§2) for FSIC have been proposed (Larson et al., 2019a; Casanueva et al., 2020a; Zhang et al., 2020; Mehri et al., 2020; Krone et al., 2020; Casanueva et al., 2020b; Nguyen et al., 2020; Zhang et al., 2021; Dopierre et al., 2021; Vulić et al., 2021; Zhang et al., 2022b). These methods are generally similar in that they utilize pretrained language models (PLMs) to encode utterances and resort to k nearest neighbors (kNN) inference: the label of a new instance is determined based on the labels of instances with which it has the highest representational similarity, as encoded by the PLM. Despite these general similarities, FSIC methods differ in design choices across several crucial dimensions, including encoding architectures, utterance similarity scoring, and training regimes. These methods tie together what are, in principle, independent design decisions across these dimensions, hindering ablations and insights into what drives the (reported) FSIC performance.

In this work, we (1) induce a framework (PLM-based utterance encoding, utterance similarity scor-

---

[*] Equal contribution.

[1] https://github.com/UKPLab/eacl2023-few-shot-intent-classification

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1846–1857
May 2-6, 2023

ing, and nearest-neighbor-based inference) that unifies most of existing FSIC approaches (§3); and (2) focus on three key design decisions within this framework: **(1)** *model architecture for encoding utterances (or utterance pairs)*, where we contrast the less frequently adopted Cross-Encoder architecture (e.g., (Vulić et al., 2021)) against the more common Bi-Encoder architecture[2] (Zhang et al., 2020; Krone et al., 2020; Zhang et al., 2021); **(2)** *similarity function* for scoring utterance pairs based on their joint or separate representations, contrasting the parameterized (i.e., trainable) neural scoring components against cosine similarity as the simple non-parameterized scoring function; and **(3)** *training regimes*, comparing the standard non-episodic training (adopted, e.g., by Zhang et al. (2021) or Vulić et al. (2021)) against the episodic meta-learning training (implemented, e.g., by Nguyen et al. (2020) or Krone et al. (2020)). Our framework lets us evaluate impacts of these three dimensions for different text encoders (e.g., BERT (Devlin et al., 2019) as a vanilla PLM and SimCSE (Gao et al., 2021) as the state-of-the-art sentence encoder) under the same evaluation setup (datasets, intent splits, evaluation protocols and measures) while controlling for confounding factors that impede direct comparison between the FSIC methods.

Our extensive experimental results on seven intent classification datasets reveal three new important findings. First, a Cross-Encoder coupled with episodic training, a previously unexplored FSIC combination, consistently yields best performance across all the datasets. Second, episodic meta-learning yields robust FSIC classifiers across the board: our results demonstrate that it is much more effective for FSIC than the conventional non-episodic training. Finally, although episodic meta-learning entails splitting utterances of an episode into a support and query set during training, we show, for the first time, that this does not generally have a positive effect on the FSIC performance.

In sum, our comparative evaluation over various design choices for key components of modern FSIC approaches raise the awareness about the importance of ablations and apple-to-apple comparison between complex FSIC systems that conflate several key design decisions. We hope that our findings pave the way for more deliberation in research (and in particular evaluation) for this crucial ToD task.

2 Related Work

We focus on few-shot intent classification (FSIC) methods, which perform class inference for utterances based on the labels of nearest neighbor (kNN), either directly in the representation space of the PLM or according to a trained scorer of utterance pairs. We first describe the existing FSIC inference paradigms and explain why we focus on kNN-based methods. We then categorize the literature on FSIC approaches based on kNN-inference along the three key design dimensions.

Inference algorithms for FSIC. Classical methods (Xu and Sarikaya, 2013; Meng and Huang, 2018; Wang et al., 2019; Gupta et al., 2019) for FSIC use the maximum likelihood inference, where a vector representation of an utterance is projected by the classifier into a probability distribution over the intent classes. Training such probability distribution functions, in particular when they are modeled by neural networks, mostly requires a large number of utterances annotated with intent labels, which are infamously expensive to collect in scenarios where new intents emerge on regular basis. By relying on pretrained language models, more recent FSIC methods leverage the language competences they posses (i.e., encode) to alleviate the need for learning to produce probability distributions for a large number intent classes, commonly with a few instances. These recent FSIC methods (Krone et al., 2020; Casanueva et al., 2020b; Nguyen et al., 2020; Zhang et al., 2021; Dopierre et al., 2021; Vulić et al., 2021; Zhang et al., 2022b) instead exploit the similarities between utterance embeddings in the representation space of the (fine-tuned) PLM and infer the intents for new utterances from the labels of nearest neighbors (kNN-based). Since kNN-based methods in general report state-of-the-art performance for FSIC, our comparative empirical evaluation focuses on the design choices for models that adopt this inference algorithm.

Model architectures for encoding utterance pairs. A central design decision within the kNN-based FSIC framework is the choice of the model architecture for encoding utterances. The majority of the approaches (Zhang et al., 2020; Krone et al., 2020; Zhang et al., 2021; Xia et al., 2021) leverage the Bi-Encoder architecture (Bromley et al., 1993; Reimers and Gurevych, 2019a; Zhang et al., 2022a). The core idea of Bi-Encoders is that, given a collection of utterances, each utterance is inde-

---

[2] Also known as Dual Encoder or Siamese Network.

pendently encoded by the PLM and mapped into a dense representation space. In such a space, similarities between pairs of utterances can be computed, with a parameterized (i.e., trainable) scoring function or a non-parameterized function such as dot product or cosine similarity. In contrast, some FSIC methods (Vulić et al., 2021; Zhang et al., 2020; Wang et al., 2021; Zhang et al., 2021) use the Cross-Encoder architecture, in which the two utterances are concatenated and encoded jointly by a pretrained text encoder, e.g., BERT (Devlin et al., 2019). The idea is to represent a pair of utterances together using a PLM, where each utterance becomes a context for the other. A Cross-Encoder thus does not produce an embedding for a single utterance but for a pair of utterances. In general, Bi-Encoders are more computationally efficient than Cross-Encoders because of the Bi-Encoder's ability to cache the representations of the candidates. In return, Cross-Encoders, by allowing tokens of one utterance to attend over the tokens of the other (and vice versa), capture better the semantic associations between utterances.

Similarity scoring function. A crucial component in nearest neighbor-based methods for FSIC is the function that produces a similarity score for a pair of utterances. Concerning this dimension of analysis, we categorize FSIC methods into two groups: (1) FSIC approaches that use parameterized (i.e., trainable) neural layers to estimate the similarity score between utterances (Zhou et al., 2022; Zhang et al., 2020; Xia et al., 2021); and (2) methods that rely on non-parameterized similarity metrics such as dot product, cosine similarity, and Euclidean distance (Sauer et al., 2022; Zhang et al., 2022a; Krone et al., 2020; Vulić et al., 2021; Zhang et al., 2022b; Xu et al., 2021; Zhang et al., 2021). Note that the Bi-Encoder architecture can be coupled with both, whereas the Cross-Encoder requires a parametrized scoring module.

Training strategy. To simulate FSIC, the best practice is to split an intent classification corpus into two disjoint sets of intent classes. In this way, one set includes high-resource intents for training of an FSIC classifier, and the other set includes low-resource intents for evaluating the classifier. Concerning the training strategy on the high-resource intents, FSIC methods can be divided into two clusters. One cluster of methods adopts meta-learning or episodic training (Zhang et al., 2022a; Nguyen et al., 2020; Krone et al., 2020). Under this training regime, the goal is to train a meta-learner that could be used to quickly adapt to any few-shot intent classification task with very few labeled examples. To do so, the set of high-resource intents are split to construct many episodes, where each episode is a few-shot intent classification task for a small number of intents. The other cluster includes methods (Zhang et al., 2021; Vulić et al., 2021; Xu et al., 2021; Xia et al., 2021; Zhang et al., 2020, 2021) that use conventional supervised (i.e., non-episodic) training. The non-episodic training simply fine-tunes the FSIC model using all samples from the high-resource intents of the training set.

3 Framework

We first unify formulations of the components we need for our framework. We then present their alternative configurations along our three central dimensions of comparison: (i) model architecture for encoding utterance pairs, (ii) functions for similarity scoring, and (iii) training regimes.

3.1 Nearest Neighbors Inference

Following previous work on FSIC (Zhang et al., 2020; Vulić et al., 2021), we cast the FSIC task as a sentence similarity task in which each intent is an implicit semantic class, captured by the representations of all the utterances associated with that intent. The task is then to find the most similar labeled utterances for the given query. During inference, the FSIC approach should deal with an N-way k-shot intent classification, where N is the number of intents and k is the number of labeled utterances given for each intent label.

Let q be a query utterance and $C = \{c_1, ..., c_n\}$ be a set of its labeled neighbors. The nearest neighbor inference relies on a similarity function, non-parameterized or trainable (which is learned on high-resource intents), to estimate the similarity score s_i between q and any c_i. The query's label $\hat{y}_q$ is inferred as the ground-truth label of the neighbor with the maximum similarity score (i.e., $k = 1$ in k-NN inference): $\hat{y} = y_k$, $k = \text{argmax}(\{s_1, ..., s_n\})$.

3.2 Model Architectures for Encoding Utterance Pairs

An encoder in an FSIC model represents a pair of a query and a neighbor (i.e., a labeled utterance) into vector $\mathbf{h}_{(q,c_i)} \in \mathbb{R}^d$. We formulate recently used encoders: Bi-Encoder and Cross-Encoder.

Bi-Encoder (BE). BE encodes a pair of utterances independently, deriving independent representations of the query and the neighbor utterance. In particular, for each utterance x in a pair, we pass, "`[CLS]` x", to a BERT-like PLM and use the representation of "`[CLS]`" to represent x. Worth noting that the parameters of the PLM are shared in BE.

Cross-Encoder (CE). Different from BE, CE encodes a pair of query q and neighbor c_i *jointly*. We concatenate q with each of its neighbors to form a set of query–neighbor pairs $P = \{(q, c_1), ..., (q, c_n)\}$. We then pass each pair from P as a sequence of tokens to a language model, which is pre-trained to represent the semantic relation between utterances. More formally, we feed a pair of utterances, "`[CLS]` q `[SEP]` c_i", to a BERT-like PLM and then use the representation of the "`[CLS]`" token as the representation of the pair.

3.3 Similarity Scoring Function

Given the pair representation, we compute the similarity between a query and a neighbor utterance by a *parameterized* or *non-parameterized* function.

PArameterized (PA). A neural-based parametric scoring function consists of a fully connected feed-forward network (FF) that transforms a pair representation into a score, $s_i = \sigma\left(\mathbf{W}^{1\times d}\mathbf{h}_{(q,c_i)} + b\right)$, where the weight $\mathbf{W}$ and bias b are trainable parameters, d is the size of the vector $\mathbf{h}_{(q,c_i)}$, and $\sigma(.)$ denotes the `sigmoid` activation function.

Non-Parameterized (NP). In contrast to PA, NP often uses vector-based similarity metrics as scoring functions, e.g., cosine similarity or Euclidean distance. Following Vulić et al. (2021), in this work we adopt the cosine similarity between h_q and h_{c_i}.

3.4 Model Configurations

Given the aforementioned components, we illustrate (Figure 1) three possible model configurations: (i) CE+PA; (ii) BE+PA, and (iii) BE+NP.

CE +PA. In this configuration, we feed the joint encoding of the utterance pair to a parameterized similarity scoring function. We note again, due to a single representation vector for both utterances, CE cannot be coupled with a non-parameterized scoring (NP).

BE +PA. In this configuration, we represent the pair by concatenating the representations of each utterance with the vectors of difference and element-wise product between those representations:

$$h_{(q,c_i)} = (h_q \oplus h_{c_i} \oplus |h_q - h_{c_i}| \oplus h_q \odot h_{c_i}), \quad (1)$$

where $\oplus$ is the concatenation operation and $\odot$ is the dot product. We motivate Equation 1 by the findings reported in Reimers and Gurevych (2019b). Similar to CE +PA, we use the sigmoid activation function on top of the feed-forward layer. The size of $\mathbf{W}$ is then $1 \times 4d$.

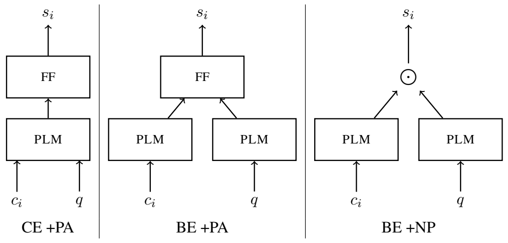


Figure 1: A demonstration of possible model configurations of encoder architectures and similarity functions to estimate the similarity score s_i between a query q and neighbor c_i. CE and BE show Cross-Encoder and Bi-Encoder architectures using a BERT-like PLM, respectively. PA and NP show parametric and non-parametric similarity functions, respectively. PA is modeled by feedforward (FF) layers and NP by the dot product $\odot$.

BE +NP. We use cosine similarity to estimate the similarity between input utterances during prediction. During training, we compute the dot product between the query and each neighbor representation vector to directly estimate their similarity scores $s_i = \sigma(h_q \odot h_{c_i})$, where $\odot$ indicates the dot product, and σ is the `sigmoid` function. We apply σ to scale s_i to a value between 0 and 1.

3.5 Training Regimes

To train the aforementioned model configurations, we formulate three training techniques as follows (Figure 2): *Non-Episodic Training (NE)*, *Episodic Training (EP)* and *Episodic Training with Support and Query splits (EPSQ)*. The training strategies rely on an identical loss function for each query.

Loss per query sample. We use the loss function defined by Zhang et al. (2020) for FSIC. In particular, we define a ground-truth binary vector y_q for a query q given a set of neighbors $C = \{c_1, ..., c_n\}$. If the query and its i-th neighbor belong to the same intent class, the corresponding label for the pair is $y_{q,i} = 1$, otherwise $y_{q,i} = 0$. Given such ground-truth label vector in consideration of the n

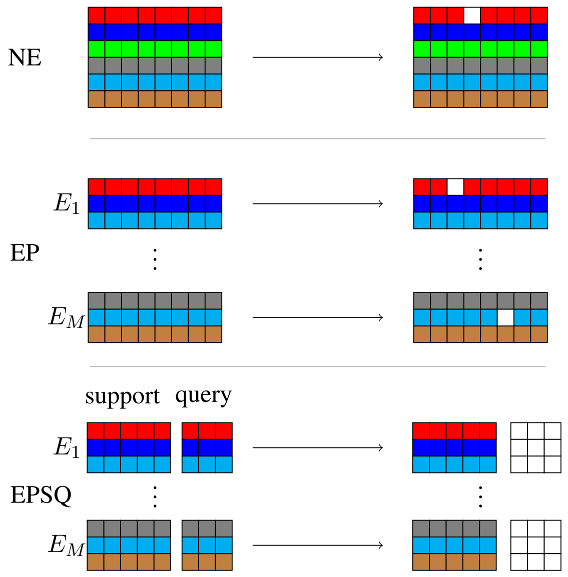


Figure 2: A illustration of training techniques, i.e., Non-episodic (NE), episodic (EP) and episodic with a support and query split (EPSQ). Each cell shows an utterance in a dataset. Each color depicts an intent class. Each row shows labeled utterances with an identical intent. In NE, the training instances are used all together and the loss is the average loss for each sample (white cell) when the other samples are neighbors. In EP, the training instances are divided into M episodes (E) and the loss is computed similar to NE for each episode. In EPSQ, each episode is split into fixed sets of of support and query instances, and the loss is computed for only the samples in the query set.

neighbors, $\mathbf{y}_q = [y_{q,t} | t = 1, ..., n]$ and similarity scores estimated by a model configuration for all pairs $\mathbf{s}_q = [s_{q,t} | t = 1, ..., n]$, we compute the binary cross-entropy loss for the query q as follows:

$$l_q(\mathbf{y}_q, s_q | C) = \\ -\frac{1}{n} \sum_{t=1}^{n} [y_{q,t} \log(s_{q,t}) + (1 - y_{q,t}) \log(1 - s_{q,t})] \, . \tag{2}$$

NE. For the NE training, the classifier learns the semantic relation between all high-resource intent classes altogether. Let D represent a batch of utterances for high-resource intent classes. Therefore, we take each utterance in D as a query q and predict its label concerning the rest of the utterances as neighbors. More formally, we estimate the loss for the NE training as follows:

$$\mathcal{L} = \frac{1}{|D|} \sum_{q \in D} l_{q|D-q}(\mathbf{y}_q, \mathbf{s}_q) \, , \tag{3}$$

where l_q is the loss defined in Equation 2 between ground truth label vector $\mathbf{y}_q$ and a vector of scores $\mathbf{s}_q$ estimated by a model configuration.

EP. An episode is a set of utterances for several intent classes. An episode formulates an N-way intent classification task, where N is the number of intent classes in the episode. The core idea behind meta-learning is to learn from a large set of high-resource intent classes by chunking the set into many episodes (Lee et al., 2022). These episodes are known as training episodes (a.k.a meta-training episodes). If set $\mathcal{I}$ denotes the intent labels of a benchmark corpus, any N randomly selected intents from $\mathcal{I}$ can be used to construct a training episode. Let's refer to these selected intents for episode E by $\mathcal{I}_E$. Then, episode E contains utterances whose intent labels are in I_E. It is worth noting that intent classes in training episodes may overlap to let a classifier learn the semantic relations between all intent labels of the benchmark. In EP, we construct M episodes from the set of utterances for high-resource intent classes D. We define the following loss function:

$$\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{|E_i|} \sum_{q \in E_i} l_{q|E_i}(\mathbf{y}_q, \mathbf{s}_q) \, , \tag{4}$$

where E_i is the ith episode, $\mathbf{y}_q$ is the ground-truth labels for the query given neighbors in the episode E_i, and $\mathbf{s}_q$ is the similarity scores between the query and any neighbor in the episode.

EPSQ. The common practice in meta-learning is to imitate the few-shot setup, an episode is split into two disjoint sets: a support and a query set (Lee et al., 2022). An episode's support set includes only a few utterances from each intent class in I_E. An episode's query set includes the rest of the utterances in the episode. A classifier should classify utterances in the query set using the utterances and intent labels in the support set. Given the kNN terminology, the support set is the set of neighbors and the query set is a set of query utterances. Therefore, the main difference between EPSQ and EP is that the number of neighbors in EPSQ is limited to only a few examples of each intent in the support set. The loss function in EPSQ is defined as follows:

$$\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{|Q_i|} \sum_{q \in Q_i} l_{q|S_i}(\mathbf{y}_q, \mathbf{s}_q) \, , \tag{5}$$

where Q_i is the query set and S_i is the support set of the ith episode.

Dataset	#Classes Train	#Classes Valid	#Classes Test	#Episodes Train	#Episodes Valid	#Episodes Test
Balanced N-way k-shot						
Clinc (150)	50	50	50	10k	10k	600
Banking (77)	25	25	27	10k	10k	600
Liu (54)	18	18	18	10k	10k	600
Hwu (64)	23	16.4	24.6	10k	10k	600
Imbalanced Support Sets						
ATIS (19)	5	7	7	1,372	213	119
SNIPS (7)	4	-	3	240	-	210
TOP (18)	7	5	6	10,095	1,286	292

Table 1: The examined datasets and their main statistics. The numbers in parenthesis show the total number of intent classes for each dataset. For HWU64, each split's number of classes varies at each run to ensure there is no cross-split domain, hence the decimal number.

4 Experiments

We conduct our experiments in two different setups: (i) balabced N-way k-shot and (ii) imbalanced classes in the support sets. The former refers to the typical few-shot learning setup, where the numbers of classes and examples per class are balanced. In contrast, the imbalanced setup randomly defines the numbers of classes and examples, imitating the imbalance nature of some benchmarks for intent classification. While arguably some utterances can be annotated to transform imbalanced episodes into balanced ones, imbalanced few-shot learning is still a huge practical challenge for various expensive domains, e.g., those that require experts for annotation (Krone et al., 2020).

Datasets, splits, and episodes. Table 1 summarizes the main statistics (e.g., the number of classes per data split for each datasets) of the datasets and their splits as we use in our experiments. For the balanced N-way k-shot setup, we use Clinc (Larson et al., 2019b), Banking (Casanueva et al., 2020b), and Hwu (Liu et al., 2021) from DialoGLUE (Mehri et al., 2020) as well as Liu (Liu et al., 2021). For the sake of fair comparisons, we use the exact splits and episodes as used by Dopierre et al. (2021) for FSIC. For 5 folds, we randomly split intents of each dataset into three sets to construct training, valid and test episodes. We then generate 5-way k-shot episodes for each split in each fold, where $k \in \{1, 5\}$. For the imbalanced setup, we use ATIS (Hemphill et al., 1990), SNIPS (Coucke et al., 2018), and TOP (Gupta et al., 2018). We follow Krone et al. (2020) to construct episodes for these datasets.

Settings. We use BERT-based-uncased and SimCSE as PLMs. We fine-tune them using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of $2e-5$. Both batch size and maximum sequence length are set to 64. See the Appendix for the full list of hyperparameters. For experiments on each fold of balanced datasets, we train a FSIC classifier for a maximum of 10,000 5-way K-shots episodes. We evaluate the classifier on the validation set after every 100 updates, and stop the training if validation performance does not improve over 5 consecutive evaluation steps. To alleviate the impact of random selection of few-shot samples, we report the average performance of a classifier for 600 test episodes, compatible with Dopierre et al. (2021). For the experiments on the imbalanced datasets, similar to Krone et al. (2020), we conduct the experiments over 1 fold due to the limited number of intents. The average number of shots per intent used in episodes of ATIS, SNIPS, and TOP is about 4, 5, and 4, respectively (see Appendix for details). For both N-way k-shot and imbalanced setups, the number of examples in query sets is identical for all intents in the query sets. So, for all experiments we report the accuracy metric averaged over all runs and folds.

Models in comparison. Alongside the results of the model configurations (§3), we report the results of the following FSIC methods to put our results in context. **Random** assigns a random intent class from the support set to each query utterance. **BE (fixed)+NP** represents a generic configuration employed by the majority of PLM-based FSIC baselines, e.g., ConvBERT (Mehri and Eric, 2021), TOD-BERT (Wu et al., 2020), and DNNC-BERT (Zhang et al., 2020), inter alia. These methods use pretrained BERT and further fine-tune on other NLP tasks (e.g., NLI) or other dialogue datasets. **ProtoNet** (Dopierre et al., 2021) is inspired by prototypical network method (Snell et al., 2017), which has been shown to achieve the state-of-the-art accuracy among meta-learning methods for few-shot learning tasks including FSIC (Krone et al., 2020). This method is not based on instance-level similarity. It encodes an intent class by a prototype vector, which is the mean of vector representations of a few utterances given for the intent. In any given episode, the prototype vector is computed for each intent. The probabilities of intents are then estimated based on the distances between a query vector and respective prototypes.

	1-shot					5-shot				
	Clinc	Banking	Liu	Hwu	Avg	Clinc	Banking	Liu	Hwu	Avg
Random	20.17	20.17	20.17	20.17	20.17	19.71	19.71	19.71	19.71	19.71
BE (fixed)+NP	30.88	27.75	30.83	29.49	29.74	48.57	38.01	45.79	41.15	43.38
ProtoNet	94.29	82.20	80.06	74.37	82.73	98.10	91.57	89.62	86.48	91.44
CE +PA										
NE	58.45	48.88	48.98	50.12	51.61	66.93	64.46	55.83	59.35	61.64
EP	93.60	79.46	77.36	72.13	80.64	98.26	**92.38**	**88.33**	84.43	**90.85**
EPSQ	**94.65**	**79.82**	**78.13**	**72.64**	**81.31**	**98.49**	92.15	88.18	**84.59**	**90.85**
BE +PA										
NE	79.48	60.26	59.15	52.04	62.73	88.04	70.28	70.49	60.47	72.32
EP	82.66	66.43	59.76	50.53	64.85	92.87	77.99	70.60	61.18	75.66
EPSQ	83.26	66.53	60.41	51.40	65.40	92.51	78.59	70.82	64.13	76.51
BE +NP										
NE	58.04	45.24	53.18	42.57	49.76	78.10	68.57	61.52	54.86	65.76
EP	67.58	52.85	52.39	41.73	53.64	76.28	67.69	63.33	51.37	64.67
EPSQ	67.80	53.83	53.17	40.96	53.94	81.31	64.58	65.32	50.41	65.41

Table 2: **BERT-based** results for the **balanced 5-way k-shot** setup, $k \in \{1, 5\}$.

5 Results and Discussion

We compare the the configurations described in (§3) and baselines (§4) for balanced and imbalanced FSIC setups using BERT, as the most widely used pretrained language model, and SimCSE, the state-of-the-art model for encoding the meaning of sentences. Our main experimental findings are as follows

- The Cross-Encoder architecture with parameterized similarity function and episodic training consistently yields the best FSIC accuracy.
- Episodic training yields more robust FSIC classifiers than non-episodic training for most of the examined setups and datasets.
- Splitting episode utterances into support and query (sub)sets, a commonly adopted practice in episodic training, does not give consistent performance gains.

5.1 Balanced FSIC

Table 2 shows accuracy of the examined FSIC approaches under comparison – based on BERT as PLM – in 1-shot and 5-shots settings. All model configurations consistently outperform the "BE (fixed)+NP" baseline. This demonstrates that fine-tuning BERT's parameters for intent classification using high-resource intent classes is paramount for generalization to unseen intents.

For both 1-shot and 5-shots, CE +PA trained with either of the two episodic training regimes (EP and EPSQ, without and with support-query splitting, respectively), achieves a higher accuracy (29% on average) than when trained in non-episodic fashion (NE), reaching, on average, the performance of ProtoNet as the state-of-art FSIC method. Both episodic training regimes are more effective than the non-episodic training across the board, not just in combination with the CE architecture. BE +PA trained via EP achieves about 2% higher accuracy for 1-shot and 3% for 5-shots than when trained with NE. For BE +NP, episodic learning (EP) results in 3.8% higher accuracy than NE for 1-shot. The only exception to this trend is BE +NP with 5-shot where EP trails NE by 1%.

EPSQ tends to exhibit a similar average accuracy as EP (less than 1% difference for average across all CE +PA, BE +PA, and BE +NP setups). This leads a conclusion that splitting utterances of an episode into a support and a query set – a common practices in episodic (FSIC) learning (Dopierre et al., 2021; Krone et al., 2020) – does not really have a pronounced (positive) effect on performance. So, it does not seem to increase the capability to generalize to unseen intent classes, as has been commonly believed but until now, to the best of our knowledge, empirically untested.

As expected, more shots (5-shots vs 1-shot) lead to consistently better FSIC accuracy: BE +NP trained with NE performs 16% better (and the other FSIC about 10% better on average). This makes intuitive sense: more shots help classifiers better refine the boundaries between the new intents.

Given that utterances in task-oriented dialogue systems are short texts, we next investigate how intermediate training for sentence representations (Phang et al., 2018; Reimers and Gurevych, 2019a; Gao et al., 2021) changes the performance of FSIC

	1-shot					5-shot				
	Clinc	Banking	Liu	Hwu	Avg	Clinc	Banking	Liu	Hwu	Avg
BE (fixed) +NP	91.33	75.48	78.75	74.58	80.03	97.89	90.33	89.61	86.93	91.19
CE +PA										
NE	60.51	54.87	49.99	46.41	52.95	78.33	72.71	68.66	67.99	71.92
EP	94.33	83.64	79.24	77.03	83.56	**98.80**	**94.22**	**90.13**	**88.54**	**92.92**
EPSQ	**95.01**	**83.83**	79.40	**77.49**	**83.93**	98.77	94.04	90.10	88.40	92.83
BE +PA										
NE	90.69	76.21	68.76	66.05	75.43	96.71	88.12	80.76	79.26	86.21
EP	90.93	76.81	71.32	65.72	76.19	96.74	88.18	83.83	80.64	87.35
EPSQ	90.95	76.43	71.33	65.71	76.11	96.83	87.95	84.10	80.90	87.45
BE +NP										
NE	93.69	81.60	79.51	75.54	82.58	98.08	91.56	89.61	87.82	91.77
EP	93.24	80.15	79.82	76.49	82.43	98.01	91.91	89.77	87.62	91.83
EPSQ	93.44	80.46	**80.21**	76.68	82.70	98.02	91.95	89.83	87.65	91.86

Table 3: **SimCSE-based** results for the **balanced 5-way k-shot** setup, $k \in \{1, 5\}$.

	ATIS	SNIPS	TOP	Avg
Random	21.34	33.70	23.99	26.34
BE (fixed) + NP	53.80	51.62	33.03	46.15
CE + PA				
NE	62.86	65.03	49.41	59.10
EP	**79.71**	**93.94**	**68.04**	**80.56**
EPSQ	71.58	92.98	62.84	75.80
BE + PA				
NE	42.91	80.22	53.48	58.87
EP	69.52	60.54	51.46	60.51
EPSQ	66.44	62.21	56.05	61.57
BE + NP				
NE	65.86	77.92	45.52	63.10
EP	65.09	79.16	47.85	64.03
EPSQ	55.67	80.08	42.97	59.57

Table 4: Results for the imbalanced setup using BERT.

models. To this end, we substitute BERT with SimCSE. Table 3 shows the results. Our three main findings hold for SimCSE-based FSIC models too. Importantly, unlike with BERT, now only CE +NP trained episodically outperforms the "BE (fixed)+NP" baseline (where PLM is not fine-tuned for intent detection). This confirms the effectiveness of coupling CE and episodic training for FSIC. It also indicates that intent detection fine-tuning is well-aligned with learning sentence representations, which is why it generally brings lower gains (or no gains) over "BE (fixed) + NP", when we start from SimCSE, pretrained exactly for encoding the meaning of sentences.

5.2 Imbalanced FSIC

Table 4 shows the results on the three imbalanced datasets. CE +PA with EP again substantially outperforms all its counterparts, confirming this never-investigated FSIC configuration as a very effective approach for the FSIC task. On average, episodic training (EP) again outperforms non-episodic (NE) training. The CE + PA and BE + NP configurations generally yield higher performance when trained without splitting the support utterances from query utterances (EP vs EPSQ). This questions the common belief in episodic meta-learning that splitting episodes into support and query sets is (always) beneficial. Overall, the findings from the imbalanced datasets align well with the main findings from central experiments on balanced datasets, as reported in Table 2 and Table 3.

6 Conclusions

We shed light on factors that contribute to performance of models for few-shot intent classification (FSIC), a crucial task in modular dialogue systems. We categorize FSIC approaches across three essential dimensions: (1) the Cross-Encoder vs. Bi-Encoder encoder architectures; (2) the parameterized (i.e., trainable) vs non-parameterized utterance similarity scoring; and (3) episodic vs non-episodic training. Our extensive evaluation, encompassing seven standard FSIC datasets, reveals that the previously unexplored combination of Cross-Encoder architecture (with parameterized utterance similarity scoring) and episodic training consistently yields the best FSIC performance. We additionally find that (i) episodic meta-learning generally outperforms the non-episodic training and (ii) that the widely adopted hypothesis in meta-learning that splitting episodes into support and query sets helps generalization and boost performance may not hold for FSIC. We hope that our findings lead to more deliberation on FSIC evaluation protocols and more insightful "apple-to-apple" comparisons between competing models and model variants.

Limitations and Ethical Concerns.

In this paper, we shed light to few-shot intent classification tasks in modular (task-oriented) dialogue systems. Dialog systems, given their direct interaction with human users, must be devoid of any negative stereotypes and must not exhibit any behaviour that could be potentially harmful to humans. That said, our work does not address the generation component of dialog systems, but merely the intent classification. As such, we do not believe it raises any ethical concerns.

The main limitation of the work – conditioned primarily by the available computational resources – is the scope of our empirical comparison: we focus on FSIC methods that subscribe to pairwise similarity scoring of utterances and nearest neighbours inference. While this subsumes much of the best performing approaches in the literature, there is a fair body of recent work that does not fall in this group. Another limitation of the work is the monolingual focus on English only. We intend to extend our work to cross-lingual transfer to other languages, for which fewer labeled intent classification datasets exist.

Acknowledgements

This work has been funded by the German Research Foundation (DFG) as part of the UKP-SQuARE project (grant GU 798/29-1), by the Collaboration Lab with Nexplore “AI in Construction” (AICO) and by the European Union under the Horizon Europe grant No 101070351 (SERMAS). We also thank our anonymous reviewers for their constructive comments on this paper.

References

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020a. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020b. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *ArXiv preprint*, abs/1805.10190.

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

Thomas Dopierre, Christophe Gravier, and Wilfried Logerais. 2021. PROTAUGMENT: Unsupervised diverse short-texts paraphrasing for intent detection meta-learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2454–2466, Online. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Abhirut Gupta, Anupama Ray, Gargi Dasgupta, Gautam Singh, Pooja Aggarwal, and Prateeti Mohapatra. 2018. Semantic parsing for technical support questions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3251–3259, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Arshit Gupta, John Hewitt, and Katrin Kirchhoff. 2019. Simple, fast, accurate intent classification and slot labeling for goal-oriented dialogue systems. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 46–55, Stockholm, Sweden. Association for Computational Linguistics.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Chia-Chien Hung, Anne Lauscher, Ivan Vulić, Simone Ponzetto, and Goran Glavaš. 2022. Multi2WOZ: A robust multilingual dataset and conversational pretraining for task-oriented dialog. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3687–3703, Seattle, United States. Association for Computational Linguistics.

Jason Krone, Yi Zhang, and Mona Diab. 2020. Learning to classify intents and slot labels given a handful of examples. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 96–108, Online. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019a. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019b. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Hung-yi Lee, Shang-Wen Li, and Thang Vu. 2022. Meta learning for natural language processing: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 666–684, Seattle, United States. Association for Computational Linguistics.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021. *Benchmarking Natural Language Understanding Services for Building Conversational Agents*, pages 165–183. Springer Singapore, Singapore.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 480–496, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tingting Ma, Qianhui Wu, Zhiwei Yu, Tiejun Zhao, and Chin-Yew Lin. 2022. On the effectiveness of sentence encoding for intent detection meta-learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3806–3818, Seattle, United States. Association for Computational Linguistics.

S. Mehri, M. Eric, and D. Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *ArXiv*, abs/2009.13570.

Shikib Mehri and Mihail Eric. 2021. Example-driven intent prediction with observers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2979–2992, Online. Association for Computational Linguistics.

Lian Meng and Minlie Huang. 2018. Dialogue intent classification with long short-term memory networks. In *Natural Language Processing and Chinese Computing*, pages 42–50, Cham. Springer International Publishing.

Aaron Mueller, Jason Krone, Salvatore Romeo, Saab Mansour, Elman Mansimov, Yi Zhang, and Dan Roth. 2022. Label semantic aware pre-training for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8318–8334, Dublin, Ireland. Association for Computational Linguistics.

Hoang Nguyen, Chenwei Zhang, Congying Xia, and Philip Yu. 2020. Dynamic semantic matching and aggregation network for few-shot intent detection. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1209–1218, Online. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *ArXiv preprint*, abs/1811.01088.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY,* *USA, February 7-12, 2020*, pages 8689–8696. AAAI Press.

Evgeniia Razumovskaia, Goran Glavaš, Olga Majewska, Edoardo M Ponti, Anna Korhonen, and Ivan Vulić. 2021. Crossing the conversational chasm: A primer on natural language processing for multilingual task-oriented dialogue systems. *ArXiv preprint*, abs/2104.08570.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Anna Sauer, Shima Asaadi, and Fabian Küch. 2022. Knowledge distillation meets few-shot learning: An approach for few-shot intent classification within and across domains. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 108–119, Dublin, Ireland. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Ivan Vulić, Pei-Hao Su, Samuel Coope, Daniela Gerz, Paweł Budzianowski, Iñigo Casanueva, Nikola Mrkšić, and Tsung-Hsien Wen. 2021. ConvFiT: Conversational fine-tuning of pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1151–1168, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chengyu Wang, Haojie Pan, Yuan Liu, Kehan Chen, Minghui Qiu, Wei Zhou, Jun Huang, Haiqing Chen, Wei Lin, and Deng Cai. 2021. Mell: Large-scale extensible user intent classification for dialogue systems with meta lifelong learning. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Yufan Wang, Jiawei Huang, Tingting He, and Xinhui Tu. 2019. Dialogue intent classification with character-cnn-bgru networks. *Multimedia Tools and Applications*, 79:4553–4572.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online. Association for Computational Linguistics.

Congying Xia, Caiming Xiong, and Philip Yu. 2021. Pseudo siamese network for few-shot intent generation. SIGIR '21, page 2005–2009, New York, NY, USA. Association for Computing Machinery.

Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 78–83.

Weiyuan Xu, Peilin Zhou, Chenyu You, and Yuexian Zou. 2021. Semantic transportation prototypical network for few-shot intent detection. In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 251–255. ISCA.

Haode Zhang, Haowen Liang, Yuwei Zhang, Li-Ming Zhan, Xiao-Ming Wu, Xiaolei Lu, and Albert Lam. 2022a. Fine-tuning pre-trained language models for few-shot intent detection: Supervised pre-training and isotropization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 532–542, Seattle, United States. Association for Computational Linguistics.

Haode Zhang, Haowen Liang, Yuwei Zhang, Li-Ming Zhan, Xiao-Ming Wu, Xiaolei Lu, and Albert Lam. 2022b. Fine-tuning pre-trained language models for few-shot intent detection: Supervised pre-training and isotropization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 532–542, Seattle, United States. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021. Few-shot intent detection via contrastive pre-training and fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1912, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5064–5082, Online. Association for Computational Linguistics.

Yunhua Zhou, Peiju Liu, and Xipeng Qiu. 2022. KNN-contrastive learning for out-of-domain intent classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5129–5141, Dublin, Ireland. Association for Computational Linguistics.

Iterative Document-level Information Extraction via Imitation Learning

Yunmo Chen[1] **William Gantt**[2] **Weiwei Gu**[2]
Tongfei Chen[3] **Aaron Steven White**[2] **Benjamin Van Durme**[1]
[1] Johns Hopkins University [2] University of Rochester [3] Microsoft Semantic Machines
`{yunmo|vandurme}@jhu.edu, {wgantt@ur.|wgu7@ur.|aaron.white@}rochester.edu, tongfei@pm.me`

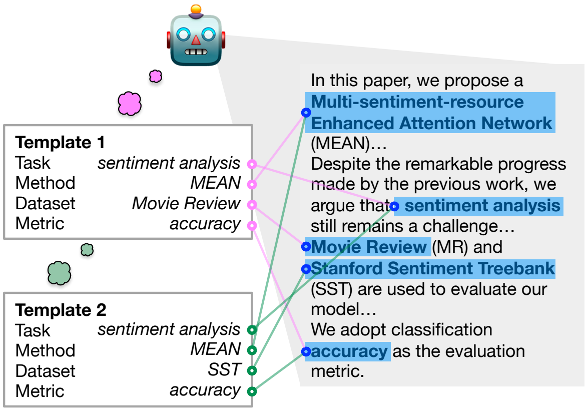
Abstract

We present a novel iterative extraction model, ITERX, for extracting complex relations, or *templates*, i.e., N-tuples representing a mapping from named slots to spans of text within a document. Documents may feature zero or more instances of a template of any given type, and the task of *template extraction* entails *identifying* the templates in a document and *extracting* each template's slot values. Our imitation learning approach casts the problem as a Markov decision process (MDP), and relieves the need to use *predefined template orders* to train an extractor. It leads to state-of-the-art results on two established benchmarks – 4-ary relation extraction on SCIREX and template extraction on MUC-4 – as well as a strong baseline on the new BETTER Granular task.[1]

1 Introduction

A variety of tasks in information extraction (IE) require synthesizing information across multiple sentences, up to the length of an entire document. The centrality of document-level reasoning to IE has been underscored by an intense research focus in recent years on problems such as argument linking (Ebner et al., 2020; Li et al., 2021, *i.a.*), N-ary relation extraction (Quirk and Poon, 2017; Yao et al., 2019; Jain et al., 2020, *i.a.*), and — our primary focus — template extraction (Du et al., 2021b; Huang et al., 2021, *i.a.*).

Construed broadly, template extraction is general enough to subsume certain other document-level extraction tasks, including N-ary relation extraction. Motivated by this consideration, we propose to treat these problems under a unified framework of *generalized template extraction* (§2).[2] Figure 1 shows 4-ary relations from the SCIREX dataset (Jain et al., 2020), presented as simple templates.



Template 1
Task sentiment analysis
Method MEAN
Dataset Movie Review
Metric accuracy

Template 2
Task sentiment analysis
Method MEAN
Dataset SST
Metric accuracy

In this paper, we propose a **Multi-sentiment-resource Enhanced Attention Network** (MEAN)… Despite the remarkable progress made by the previous work, we argue that **sentiment analysis** still remains a challenge… **Movie Review** (MR) and **Stanford Sentiment Treebank** (SST) are used to evaluate our model… We adopt classification **accuracy** as the evaluation metric.

Figure 1: An example of multi-template extraction on a document (an NLP paper; Lei et al. (2018)) from the SCIREX dataset. An agent reads the entire paper and *iteratively* generates templates, each consisting of slots for `Task`, `Method`, `Dataset`, and `Metric`.

Since documents typically describe multiple complex events and relations, template extraction systems must be capable of predicting multiple templates per document. Existing approaches such as Du et al. (2021b) and Huang et al. (2021) rely on a linearization strategy to force models to learn to predict templates in a pre-defined order. In general, however, such orderings are arbitrary. Others have instead focused on the simplified problem of role-filler entity extraction (REE), which entails extracting all slot-filling entities but does not involve mapping them to individual templates (Patwardhan and Riloff, 2009; Du et al., 2021a, *i.a.*).

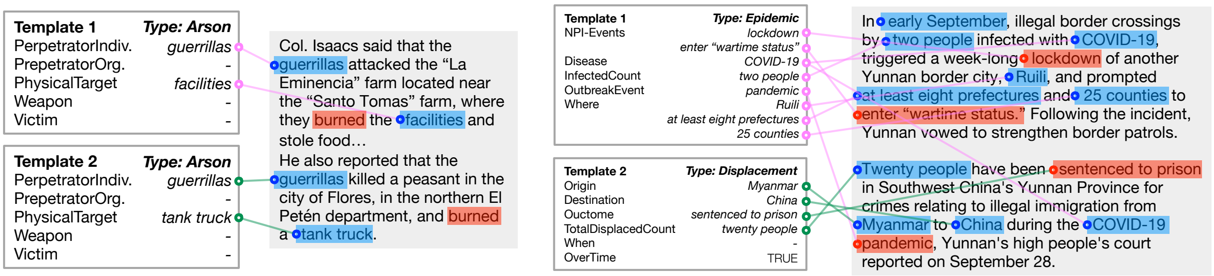
We present a new model for generalized template extraction, ITERX, that iteratively extracts multiple templates from a document *without* requiring a pre-defined linearization scheme. We formulate the problem as a Markov decision process (MDP, §2), where an action corresponds to the generation of a single template (§3.2), and states are sets of predicted templates (§3.3). Our system is trained via *imitation learning*, where the agent imitates a dynamic oracle drawn from an expert policy (§3.4). Our contributions can be summarized as follows:

---

[1] Code available at `github.com/wanmok/iterx`.

[2] We encourage the reader to consult Appendix A for a discussion of some important differences between generalized template extraction and traditional event extraction.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1858–1874
May 2-6, 2023 ©2023 Association for Computational Linguistics



Col. Isaacs said that the guerrillas attacked the “La Eminencia” farm located near the “Santo Tomas” farm, where they burned the facilities and stole food…

He also reported that the guerrillas killed a peasant in the city of Flores, in the northern El Petén department, and burned a tank truck.

In early September, illegal border crossings by two people infected with COVID-19, triggered a week-long lockdown of another Yunnan border city, Ruili, and prompted at least eight prefectures and 25 counties to enter “wartime status.” Following the incident, Yunnan vowed to strengthen border patrols.

Twenty people have been sentenced to prison in Southwest China's Yunnan Province for crimes relating to illegal immigration from Myanmar to China during the COVID-19 pandemic, Yunnan's high people's court reported on September 28.

Figure 2: Template examples from MUC-4 (left) and BETTER Granular (right) datasets. Event triggers (e.g. *burned* above) are *not* annotated in MUC-4 and are highlighted here only for clarity.

- We show that generalized template extraction can be treated as a Markov decision process, and that imitation learning can be effectively used to train a model to learn this process without making explicit assumptions about template orderings.
- We demonstrate state-of-the-art results with ITERX on two established benchmarks for complex relation extraction: 4-ary relation extraction on SCIREX and template extraction on MUC-4.
- We introduce strong baselines for the recently introduced English BETTER Granular template extraction task.

2 Problem Formulation

We propose to treat both classic template extraction and N-ary relation extraction under a unified framework of *generalized template extraction*. Given a document $D = (w_1, \cdots, w_N)$ where each w_i is a token, we assume that some system (or model component) generates a set of candidate *mention spans* $\mathcal{X} = \{x_1, \cdots, x_M\}$, where each $x_i = D[l_i : r_i] \in \mathcal{X}$ is contiguous with left and right span boundary indices l_i and r_i.

We define a *template ontology* as a set of template types $\mathcal{T}$, where each type $t \in \mathcal{T}$ is associated with a set of slot types S_t. A *template instance* is defined as a pair $(t, \{(s_k : X_k), \cdots\})$ where $t \in \mathcal{T}$ is a template type, $s_k \in S_t$ is a *slot type* associated with t, and $X_k \subseteq \mathcal{X}$ is a subset of all mention spans extracted from the document that fills slot type s_k ($X_k = \varnothing$ indicates that slot s_k has no filler).[3] Taking Figure 2 (left) as an example, Template 1 has type $t =$ `Arson` and slots {`PerpretratorIndiv` : {“guerrillas”}, `PhysicalTarget` : {“facilities”}}.

We reduce the problem of extracting a *single* template to the problem of assigning a slot type to each extracted span $x_i \in \mathcal{X}$, where some spans may be assigned a special *null* type (ε), indicating that they fill no slot in the current template. Given this formulation, we can equivalently specify a template instance as (t, a) where a is an *assignment* of spans to slot types: $\{x_i : s_i\}_{s_i \in S_t}$. We denote the union of all slot types across all template types, along with the empty slot type ε, as $\mathcal{S} = \{\varepsilon\} \cup \bigcup_{t \in \mathcal{T}} S_t$.

With these definitions in hand, the problem of *generalized template extraction* can be stated succinctly: Given a template ontology $\mathcal{T}$, a document D, and a set of candidate mentions $\mathcal{X}$ extracted from D, generate a set of template instances $\{(t_1, a_1), \cdots, (t_K, a_K)\}$, where $t_i \in \mathcal{T}$.

As an MDP We treat generalized template extraction as a Markov decision process (MDP), where each step of the process generates one whole template instance. For simplicity, we consider the problem of extracting templates of a specific type $t \in \mathcal{T}$; extracting *all* templates then simply requires iterating over $\mathcal{T}$, where $|\mathcal{T}|$ is typically very small. This MDP $(2^{\mathcal{A}}, \mathcal{A}, E, R)$ comprises the following:[4]

- $2^{\mathcal{A}}$: the set of states. In our case, this is the set of all template generation histories. Each state $A \subseteq \mathcal{A}$ is a set of generated templates;
- $\mathcal{A}$: the set of *actions* or *assignments*: an action is the generation of a single template (an assignment of slot types to spans);
- E: the environment that dictates state transitions. Here, each transition simply adds a generated template to the set of all templates generated for the document: $E(A, a) = A \cup \{a\}$;
- $R(A, a)$: the reward from action a under the current state A.

These components are detailed in the following section. Figure 3 shows ITERX in action: the MDP

---

[3] In this work, we use *template* as an abbreviation for *template instance*, relying on the *type* vs. *instance* usage only when necessary for clarity.

[4] Our notation is consistent with prior NLP work that uses MDPs, e.g., Levenshtein Transformers (Gu et al., 2019).

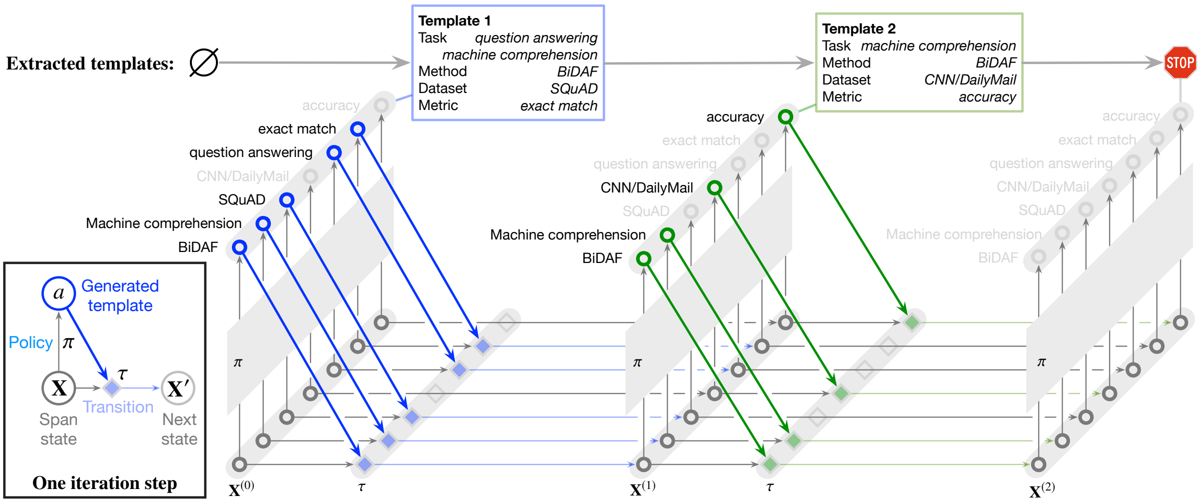


Figure 3: The basic iteration step of ITERX (left box), and an **unrolled** version on SCIREX 4-ary relation extraction (extraction of templates in the form {`Task`, `Method`, `Dataset`, `Metric`}) executed on the NLP paper *Bidirectional Attention Flow for Machine Comprehension* (Seo et al., 2017). Span embeddings ($\mathbf{X}^{(0)}$) are passed as input to the first step, where the model extracts the template {`Task`: *Machine comprehension, Question answering*; `Method`: *BiDAF*; `Dataset`: *SQuAD*; `Metric`: *Exact match*}. This information is propagated via our *memory mechanism* to the second step, and informs prediction of the next template: {`Task`: *Machine comprehension*; `Method`: *BiDAF*; `Dataset`: *CNN/DailyMail*; `Metric`: *Accuracy*}. The third step assigns the null slot type ε to all spans, indicating that the model is unable to find any further templates, thus stopping the generation process.

produces two templates sequentially, terminating on a null assignment to all input spans in $\mathcal{X}$.

3 Model

Our ITERX model is a parameterized agent that makes decisions under the MDP above: conditioned on an input document D, spans $\mathcal{X}$ extracted from D, and a specific template type t, ITERX generates a single template of type t at each step. The model consists of two parameterized components:

- **Policy π**: A policy $\pi(a \mid t, \mathbf{X})$ that generates a distribution over potential assignments of spans to slots in the current template of type t;
- **State transition model τ**: An autoregressive state encoder that maps a state (i.e., a set of predicted templates) A to a continuous representation $\mathbf{X}$ via a state transition model τ, where $\mathbf{X}^{(k+1)} = \tau(\mathbf{X}^{(k)}, a_k)$. Here the state representation $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_M)$ comprises of a vector $\mathbf{x}_i \in \mathbb{R}^d$ for each span x_i.

ITERX generates a sequence of templates: It starts with initial state $\mathbf{X}^{(0)}$ (see Figure 3 for a running example) comprising only span representations derived from the encoder (as no templates have been predicted) and ends when no new template is generated (§3.5). ITERX is trained via imitation learning, aiming to imitate an expert policy π^* derived from reference templates.

3.1 Span Extraction and Representation

ITERX takes *spans* as input and thus relies on a span proposal component to obtain candidate spans.[5] For all experiments, we use the neural CRF-based span finder employed for FrameNet parsing in Xia et al. (2021) and for the BETTER Abstract task in Yarmohammadi et al. (2021).[6] CRF-based span finders have been empirically shown to excel at IE tasks (Gu et al., 2022).

The input document is first embedded using a pretrained Transformer encoder (Devlin et al., 2019; Raffel et al., 2020) that is fine-tuned during model training.[7] Each span $x = D[l : r]$ extracted by the span extractor is encoded as a vector $\mathbf{x}_{\text{enc}}$, which is obtained by first concatenating three vectors of dimension d: the embeddings of the first and the last tokens in the span, and a weighted sum

---

[5] Although many systems since Lee et al. (2017) that require spans as input have trained span proposal modules end-to-end, we found this to be unnecessary to obtain strong results and leave this extension for future work.

[6] The code of the span finder can be found at `github.com/hiaoxui/span-finder`. We refer the reader to these papers for further details.

[7] Different encoders are used in §4 for fair comparison with prior work. For documents exceeding maximum length $N_{\max} = 1024$, tokens are encoded in chunks of size $N_{\max}$.

of the embeddings of all tokens within the span, using a learned global query (Lee et al., 2017). This $3d$-dimensional vector is then compressed to size d using a two-layer feedforward network. Lastly, to incorporate positional information, we add sinusoidal positional embeddings based on the token-level offset of l within the document to yield $\mathbf{x}_{\text{enc}}$.

3.2 Policy: Generating a Single Template

A policy π generates a single template given span states $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)$ and template type $t \in \mathcal{T}$, conditioned on the document and all of its candidate mention spans.

Since an action a represents a set of *slot type assignments* for all candidate mentions, the policy $\pi(a \mid t, \mathbf{X})$ can be factorized as

$$\pi(a \mid t, \mathbf{X}) = \prod_{(x:s)\in a} P(s|t, \mathbf{x}) \ . \qquad (1)$$

Thus we only need to model the slot type distribution for each candidate span. Here, we employ two models described below.

Independent Modeling We train a classifier that outputs a slot type (or ε) given both the template type embedding $\mathbf{t}$ and the slot type embedding $\mathbf{s}$, inspired by a standard practice in binary relation extraction (Ebner et al., 2020; Lin et al., 2020, *i.a.*). It computes the probability with a two-layer feedforward network (FFN), with slots not associated with the template type (i.e., $s \notin S_t \cup \{\varepsilon\}$) assigned 0 probability:

$$P_{\text{ind}}(s|t, \mathbf{x}) \propto \mathbf{1}_{s\in S_t\cup\{\varepsilon\}} \cdot \exp(\mathbf{s}^{\mathrm{T}} \cdot \mathrm{FFN}([\mathbf{t}; \mathbf{x}])) \quad (2)$$

where $[\cdot \ ; \ \cdot]$ denotes vector concatenation.

Joint Modeling Following Chen et al. (2020), we create a model that *jointly* considers all candidate spans given the template type. We begin by prepending $\mathbf{t}$ to the sequence of span states $\mathbf{X}$ to yield the sequence $(\mathbf{t}, \mathbf{x}_1, \cdots, \mathbf{x}_n)$. This sequence is fed to a different Transformer encoder, which naturally models interactions both between spans and between a span and the template type via self-attention (Vaswani et al., 2017):

$$(\hat{\mathbf{t}}, \hat{\mathbf{x}}_1, \cdots, \hat{\mathbf{x}}_M) = \mathrm{Transformer}(\mathbf{t}, \mathbf{x}_1, \cdots, \mathbf{x}_M) \quad (3)$$

We emphasize that the inputs to the Transformer are embeddings of *spans* (see §3.1) and not tokens, following Chen et al. (2019, 2020).[8] For each $\mathbf{x}_i$, we pass the representation $\hat{\mathbf{x}}_i$ output by the Transformer to a linear layer with output size $|\mathcal{S}|$, the total number of slot types. A softmax activation is then applied over all slot types s that are valid for t (i.e., $s \in S_t \cup \{\varepsilon\}$), with invalid types masked out, yielding the following distribution:

$$P_{\text{joint}}(s|t, \mathbf{x}) \propto \mathbf{1}_{s\in S_t\cup\{\varepsilon\}} \cdot \exp(\mathbf{s}^{\mathrm{T}} \hat{\mathbf{x}}) \qquad (4)$$

3.3 State Transition Model

A *state transition model* models the environment $E(A, a)$. Recall that a state transition just consists in the generation of a single template, where the current state A is the set of all templates that have been generated up to the current step.

Here, we propose a neural model that produces a representation of A. Specifically, we model A as a sequence of vectors $\mathbf{X}_{\text{mem}}(A) \in \mathbb{R}^{M\times d}$ — one d-dimensional state vector for each of the M candidate spans $x \in \mathcal{X}$. Each state vector $\mathbf{x}_{\text{mem}} \in \mathbb{R}^d$ acts as a *span memory*, tracking the *use* of that span across generated templates. We model state transitions using a single gated recurrent unit (GRU; Cho et al., 2014). Given the current template assignment $(x : s) \in a$ of a slot type s to a span x, the state transition for $A' = A \cup \{a\}$ is given as follows:

$$\mathbf{x}'_{\text{mem}} = \begin{cases} \mathrm{GRU}(\mathbf{x}_{\text{mem}}, [\mathbf{s} \ ; \ \hat{\mathbf{t}}]) & \text{if } s \neq \varepsilon; \\ \mathbf{x}_{\text{mem}} & \text{if } s = \varepsilon. \end{cases} \qquad (5)$$

where $\hat{\mathbf{t}}$ is a template embedding given by $\hat{\mathbf{t}} = \mathbf{t}$ when using the independent policy model given in Equation 2 and given by Equation 3 when using the joint model. Intuitively, $\hat{\mathbf{t}}$ is a summarized vector of the current template, akin to the role of the [CLS] token employed in BERT (Devlin et al., 2019). Here, we use a concatenation of the slot type embedding $\mathbf{s}$ and the template vector $\hat{\mathbf{t}}$ as the input to the state transition GRU to track the use of the span.

The input representation of a span x at each step is simply $\mathbf{x} = \mathbf{x}_{\text{enc}} + \mathbf{x}_{\text{mem}}$ — the sum of the original span embeddings $\mathbf{x}_{\text{enc}}$ described in §3.1 and the current memory vector $\mathbf{x}_{\text{mem}}$.

3.4 Policy Learning

We use *direct policy learning* (DPL), a type of imitation learning, to train our model. DPL entails training an agent to imitate the behavior of an *interactive demonstrator* as given by optimal actions a^* drawn from some expert policy π^*, a proposal

---

[8] Positional embeddings are not needed in this Transformer since sinusoidal embeddings are already added to the span representations.

distribution over actions. This expert policy is computed dynamically based on the current state of the agent, as we describe below. For this reason, the interactive demonstrator is sometimes referred to as a *dynamic oracle* (Goldberg and Nivre, 2012).

The log-likelihood of the oracle action under the ITERX *policy model is the reward.* This ensures that the learning problem can be optimized directly using gradient descent, where the objective is given by the expected reward:

$$\mathbb{E}_{\substack{a^* \sim \pi^* \\ A \sim d_{\tilde{\pi}}}} \left[\sum_{k=0}^{\infty} \gamma^k \log \pi(a^* | t, \mathbf{X}(A)) \right] \qquad (6)$$

Here, γ is a discount factor, $\tilde{\pi}$ is the *mixed* policy, and states are repeatedly sampled from their induced state distribution $d_{\tilde{\pi}}$. The mixed policy $\tilde{\pi}$ is a mixture of the expert policy and the agent's policy (Ross et al., 2011). Sampling from $\tilde{\pi}$ can thus be described as first sampling some $u \in \{0, 1\}$, then sampling from the agent's parameterized policy π if $u = 1$, or sampling an action from the dynamic oracle π^* if $u = 0$:

$$\begin{aligned} u &\sim \text{Bernoulli}(\alpha); \\ \hat{a} &\sim u\pi + (1 - u)\pi^*. \end{aligned} \qquad (7)$$

Here α, the *agent roll-out rate*, or the *agent policy mixing rate*, is a hyperparameter that controls the probability of the agent following its own policy vs. the dynamic oracle.

This process resembles *scheduled sampling* (Bengio et al., 2015), a technique commonly employed in training models for sequence generation tasks like machine translation: when updating decoder hidden states, either the gold token y^* or the predicted token $\hat{y}$ may be used, and the decision is made via a random draw. Here, the difference is that we are generating templates at each step instead of tokens.

Expert Policy We construct an expert policy based on the agent's policy. At training time, given the set of gold templates A^* and the current state A (all templates predicted thus far), the set $\bar{A} = A^* \setminus A$ contains *all gold templates not yet predicted*. Our expert policy is formulated as

$$\pi^*(a \mid t, \mathbf{X}) \propto \begin{cases} e^{\log \pi(a|t,\mathbf{X})/\beta} & \text{if } a \in \bar{A} \\ 0 & \text{if } a \notin \bar{A} \end{cases} \qquad (8)$$

where β is a temperature parameter. Intuitively, our expert policy seeks to "please" the agent: a (viable) action's probability under the expert policy is proportional to the probability under the agent's policy. Temperature β controls concentration: $\beta \to 0^+$ reduces it to a point distribution over a single action and $\beta \to \infty$ results in equal probability assigned to all remaining gold templates.

3.5 Inference

Although many search algorithms for sequence prediction can be employed (e.g. beam search, A*), we find greedy decoding to be effective, and leave further exploration for future work. Setting the initial state $A^{(0)} = \emptyset$, we take actions (i.e., generate templates) by greedy decoding $\hat{a} = \arg\max_a \pi(a \mid t, \mathbf{X})$ for every step. Decoding stops when all spans are assigned the null slot type ε in $\hat{a}$.

4 Experiments

We evaluate ITERX on three datasets: SCIREX (Jain et al., 2020), MUC-4 (Grishman and Sundheim, 1996), and BETTER Phase II English Granular.[9] SCIREX is a challenge dataset for 4-ary relation extraction[10] on full academic articles related to machine learning. MUC-4 and Granular are both traditional template extraction tasks, though they differ in important respects, which we discuss in Appendix C. For summary statistics, see Table 1.

	SCIREX			**MUC-4**			**Granular**		
	Train	Dev	Test	Train	Dev	Test	Train	Dev	Test
# documents	306	66	66	1,300	200	200	302	34	32
# templates	1,627	251	271	1,114	191	209	610	57	47
# temp. types	1			6			6		
# slot types	4			5			$92 + 4^\dagger$		

Table 1: Summary statistics of the datasets. † indicates slot types that take non-span values as fillers.

4.1 Baselines

GTT (Du et al., 2021b) To our knowledge, this is the only prior work to have attempted full template extraction in recent years, and it is thus our primary baseline for comparison on MUC-4.[11] GTT first prepends the document text with the valid template types, then passes the result to a BERT

---

[9] `https://ir.nist.gov/better`.

[10] SCIREX also contains a binary relation extraction task, but the binary relation is a subrelation of the 4-ary relation, and thus is subsumed by the more difficult task.

[11] We were unfortunately unable to obtain reasonable performance with GTT on SCIREX, and so do not compare ITERX and GTT on this task.

encoder. A Transformer decoder (whose parameters are shared with the encoder) then generates a linearized sequence of template instances.

TempGen (Huang et al., 2021) This is the current state-of-the-art system for REE (the simplified slot-filling entity extraction task) on MUC-4. On SciREX, TempGen may output multiple relation instances, but only one canonical mention as the filler for each role in the relation. On MUC-4, TempGen outputs a single aggregate template per document, but allows multiple spans to fill a template slot. We make minimal modifications to the TempGen source code to support multi-filler, multi-template prediction on both datasets, allowing for direct comparison to IterX and GTT on full template extraction.

4.2 Metrics

The standard metric for template extraction and REE on MUC-4 is CEAF-REE, proposed in Du et al. (2021a) and then used in Du et al. (2021b) and Huang et al. (2021).[12] CEAF-REE is based on the CEAF metric (Luo, 2005) for coreference resolution, that computes an alignment between gold and predicted entities that maximizes a measure of similarity ϕ between aligned entities (e.g. CEAF$_{\phi_4}$ in coreference resolution). This alignment is subject to the constraint that each reference entity is aligned to at most one predicted entity.

The CEAF-REE *implementation* (henceforth, CEAF-REE$_{\text{impl}}$) employed in Du et al. (2021a,b) and Huang et al. (2021) unfortunately departs from the stated metric *definition* (CEAF-REE$_{\text{def}}$) in two ways: (1) it eliminates the constraint on entity alignments and (2) it treats the template *type* as an additional slot when reporting cross-slot averages. For maximally transparent comparisons to prior work, we report scores under both CEAF-REE$_{\text{def}}$ and CEAF-REE$_{\text{impl}}$, obtaining state-of-the-art results on MUC-4 with each.

However, we argue that neither CEAF-REE$_{\text{def}}$ nor CEAF-REE$_{\text{impl}}$ is consistent with historical evaluation of template extraction systems. CEAF-REE$_{\text{def}}$ errs in enforcing the entity alignment constraint: doing so effectively requires systems to perform coreference resolution, which is too strict and runs contrary to the original MUC-4 evaluation. By contrast, CEAF-REE$_{\text{impl}}$ also errs in treating the template type as just another slot: this elides the important distinction between the *kind* of event being described and the participants in that event (§6).

In the interest of clarity, we define a modified version of the CEAF-REE metric that avoids both pitfalls: it relaxes the entity alignment constraint and it does not include template type in cross-slot averages. We call this version CEAF-R**M**E, where "M" stands for *mention* and emphasizes the focus on mention-level rather than entity-level ("E") scoring. Intuitively, relaxing this constraint amounts to placing the burden of coreference resolution on the metric: if the scorer aligns two predicted mentions to the same reference entity, the mentions are implicitly deemed coreferent. See Figure 4 for a comparison among these variants.[13]

For SciREX, we report CEAF-REE$_{\text{def}}$ and CEAF-RME. For BETTER Granular, we use its official metrics, described in Appendix D.

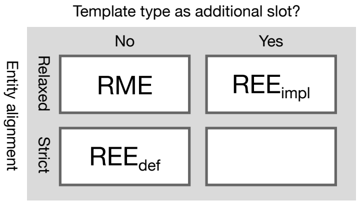


Figure 4: A comparison of the metrics discussed.

4.3 Results

SciREX For TempGen, we report models trained with BART$_{\text{base}}$ and BART$_{\text{large}}$, where only BART$_{\text{base}}$ was used in Huang et al. (2021). While BART is an encoder-decoder architecture, IterX uses only the encoder part, and thus requires about half the pretrained parameters that TempGen does.[14] Even with far fewer parameters, IterX outperforms the BART$_{\text{large}}$ baseline by a wide margin. Moreover, our best performing model under T5$^{\text{enc}}_{\text{large}}$ (Raffel et al., 2020) achieves roughly 2× the performance of TempGen[15] (see Table 2).

MUC-4 Under the most comparable setting, IterX outperforms GTT under all metrics by 1–2%, both using BERT$_{\text{base}}$ (Table 2). With T5$^{\text{enc}}_{\text{large}}$, IterX obtains even better performance, with most

---

[12] The standard metrics for template extraction may be unfamiliar to IE researchers more accustomed to sentence-level *event extraction*. Accordingly, we thoroughly motivate and describe all template extraction metrics we use in this work in Appendix D. What follows is a more abridged discussion.

[13] See `github.com/wanmok/iterx` for implementations.

[14] We add the superscript "enc" to pretrained models to denote the use of the encoder only (decoder discarded): T5$^{\text{enc}}_{\text{large}}$.

[15] Replacing BART with T5 in TempGen would have mandated destructive modifications to the pretrained architecture, and we therefore do not report results under this setting.

Model (Encoder)	SciREX						MUC-4								
	CEAF-REE$_{\text{def}}$			CEAF-RME			CEAF-REE$_{\text{def}}$			CEAF-REE$_{\text{impl}}$			CEAF-RME		
	P	R	F$_1$	P	R	F$_1$	P	R	F$_1$	P	R	F$_1$	P	R	F$_1$
TempGen (BART$_{\text{base}}$)	8.7	5.0	6.4	8.6	8.2	8.4	54.2	15.8	24.5	55.7	40.0	46.4	58.3	31.0	40.5
TempGen (BART$_{\text{large}}$)	19.9	5.0	8.0	8.9	22.3	12.7	**55.8**	18.9	28.3	**63.7**	37.4	47.2	**61.3**	32.9	42.8
GTT (BERT$_{\text{base}}$)	-	-	-	-	-	-	54.7	23.0	32.3	61.7	42.4	50.2	55.0	36.8	44.1
IterX (BERT$_{\text{base}}$)	16.2	7.6	10.4	16.2	17.4	16.8	41.3	**27.9**	33.3	52.3	**51.1**	51.7	47.2	**45.0**	46.1
IterX (BART$^{\text{enc}}_{\text{base}}$)	15.0	**15.0**	15.0	14.3	35.4	20.3	39.2	24.8	30.4	49.8	45.7	47.6	44.8	40.1	42.3
IterX (T5$^{\text{enc}}_{\text{large}}$)	**26.4**	12.4	**16.9**	**25.0**	**40.6**	**31.0**	53.5	26.2	**35.2**	60.9	46.9	**53.0**	55.8	42.4	**48.2**

Table 2: Results on SciREX and MUC-4.

gains coming from increased precision.[16] Furthermore, we note a consistent gap of > 5% F$_1$ between CEAF-RME and CEAF-REE$_{\text{impl}}$, which we suspect is due to CEAF-REE$_{\text{impl}}$'s inclusion of scores for template type into the aggregated slot F$_1$: as template type scores are higher across models than slot type scores, they are liable to inflate the aggregate score.

BETTER Granular We report scores on the English-only Phase II BETTER Granular task using the official BETTER scoring metric in Table 3. Given the complexity of the Granular task, the accompanying difficulty of developing models to perform it, and the lack of existing work on Granular, we report scores only for IterX under T5$^{\text{enc}}_{\text{large}}$. We intend these to serve as a solid baseline against which future work may be measured.

Template			Slot			Combined
P	R	F$_1$	P	R	F$_1$	
89.7	74.5	81.4	41.0	33.5	36.9	30.0

Table 3: Results on the BETTER Granular dataset. Combined score is Template F1 × Slot F1.

5 Analysis

We next conduct ablations to examine how specific aspects of IterX's design affect learning. Here, we focus on SciREX as a case study, as it has the highest average templates per document of the three datasets, allowing us to best investigate the behavior of IterX over long action sequences.

Recall that the dynamic oracle specifies an expert policy π^* (Equation 8) from which expert actions a^* are drawn. One design decision concerns the *agent roll-out rate*, α, which controls how often we draw from the expert policy vs. the agent policy when making updates. Another decision concerns how *entropic* this policy distribution should be, controlled by the temperature β. Both decisions reflect a trade-off between exploration and exploitation in the space of action sequences.

Agent Roll-out Rate α We show how model performance changes as we increase the agent roll-out rate $\alpha \in [0, 1]$ in Figure 5, where $\alpha = 0$ to always following the expert policy, and $\alpha = 1$ corresponds to always updating based on the agent's own policy. The model performs poorly under low α, but improves quickly as α increases, reaching a plateau past $\alpha \geq 0.5$. The results are intuitive, as relying more on the expert (lower α) for learning would result in a fixed and deterministic set of states that may hinder the agent from visiting new states, which are often encountered at test time. With higher α, the agent's behavior is more consistent between train and test time.

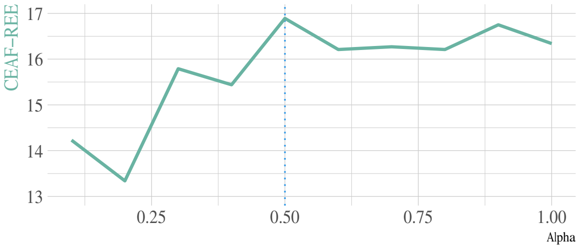


Figure 5: Performance changes on CEAF-REE$_{\text{def}}$ with respect to α, for which higher means higher probability of rolling out agent's policy for state update.

Temperature β We compare the following four settings for sampling from π^*, keeping $\alpha = 0$ to control for effects of policy mixing:

- Fixed: Select the next template in the document based on the order as is given in the dataset. In this case, β does not come into play. This setting corresponds to the standard practice of using

---

[16] For GTT, we directly use the output files included in the codebase of Du et al. (2021b). However, we were unsuccessful in adapting their codebase ourselves to make use of T5.

fixed template linearizations (Du et al., 2021b; Huang et al., 2021).

- $\beta \to 0^+$ (ARGMAX): Select the template that maximizes the likelihood with the system-predicted distributions over slots.
- $\beta = 1$ (XENT): Sample a template according to the distribution defined by the cross entropy between references and predictions.
- $\beta \to \infty$ (UNIFORM): Sample a template uniformly from the correct template set.

Test set performance for each setting is shown in Table 4. The results for CEAF-RME show a trade-off in precision and recall corresponding to the exploitation-exploration trade-off induced by β, with the higher β (more exploration) of XENT and UNIFORM, yielding higher recall. The trend for CEAF-REE is similar, though less pronounced.

Approach	**CEAF-REE$_{\text{def}}$**			**CEAF-RME**		
	P	R	F_1	P	R	F_1
FIXED	28.7	7.3	11.7	29.9	29.9	29.9
ARGMAX	26.4	6.9	11.0	29.4	30.2	29.8
XENT	28.7	10.7	15.6	27.8	32.3	29.9
UNIFORM	29.1	11.3	16.3	27.6	32.0	29.7

Table 4: Results with different choices of temperature.

Interestingly, while CEAF-RME F_1 scores are consistent across settings, CEAF-REE F_1 scores are higher under higher temperature settings. To the extent that the more entropic settings conduce to higher template and mention recall, we would expect these settings to yield more partial-credit template alignments than non-random settings, which tend to focus on correct prediction of fewer templates — thus potentially missing templates entirely and receiving no partial credit.

6 Related Work

Template Extraction The term *template extraction* was originally proposed in the Message Understanding Conferences (MUC; Sundheim, 1991, *i.a.*) to describe the task of extracting templates from articles. Researchers later focused more heavily on sentence-level IE, especially after the release of the ACE 2005 dataset (Walker et al., 2006). But following renewed interest in document-level IE, researchers (Du et al., 2021b; Huang et al., 2021; Gantt et al., 2022, *i.a.*) have begun to revisit MUC and to develop new template extraction datasets (notably, BETTER Granular).

Traditionally, template extraction comprises two sub-tasks: *template identification*, in which a system identifies and types all templates in a document, and *slot filling* or *role-filler entity extraction* (REE), in which the slots associated with each template are filled with extracted entities. Much recent work in this domain has turned away from the full task, focusing only on REE, which is tantamount to assuming that there is just a single *aggregate* template per document (Patwardhan and Riloff, 2009; Huang and Riloff, 2011, 2012; Du et al., 2021a; Huang et al., 2021).

Document-Level Relation Extraction Alongside template extraction, there has been considerable recent interest within IE in various challenging document-level relation extraction objectives, beyond the longstanding and dominant focus on coreference resolution. *Argument linking* — a generalization of semantic role labeling (SRL; Gildea and Jurafsky, 2002) in which a predicate's extra-sentential arguments must also be labeled — is one notable example, and has attracted recent attention through the RAMS (Ebner et al., 2020) and WikiEvents (Li et al., 2021) benchmarks.[17] Prior benchmarks on this task include SemEval 2010 Task 10 (Ruppenhofer et al., 2010), Beyond Nombank (Gerber and Chai, 2010), ONV5 (Moor et al., 2013), and Multi-sentence AMR (O'Gorman et al., 2018). A separate line of work has concentrated on general *N-ary relation extraction* challenge tasks, in which entities participating in the same relation may be scattered widely throughout a document. Beyond SCIREX, PubMed (Quirk and Poon, 2017; Peng et al., 2017) and DocRED (Yao et al., 2019) are two other prominent benchmarks in this area.

Imitation Learning Our approach casts the problem of generalized template extraction as a Markov decision process. SEARN (Daumé III et al., 2009) and other related work (Ross et al., 2011; Venkatraman et al., 2015; Chang et al., 2015, i.a.) have considered structured prediction under a reinforcement learning setting. Notably, in dependency parsing, Goldberg and Nivre (2012) proposed the use of a *dynamic oracle* to guide an agent toward the correct parse (see §3.4).

---

[17] Argument linking also goes by the names *event argument extraction* and *implicit semantic role labeling*, though these terms are not precisely equivalent.

We also employ direct policy learning for optimization of the template extraction MDP, thus reducing the problem to one of supervised sequence learning that is amenable to gradient descent. Such treatment is reminiscent of other similar techniques in NLP. Scheduled sampling (Bengio et al., 2015), for instance, trains a sequence generator with an expert policy consisting of a mixture of the predicted token and the gold token. Relatedly, Levenshtein Transformers (Gu et al., 2019) learn to edit a sequence by imitating an expert policy based on the Levenshtein edit distance.

7 Conclusion

We have presented ITERX, a new model for generalized template extraction that iteratively generates templates via a Markov decision process. ITERX demonstrates state-of-the-art performance on two benchmarks in this domain — 4-ary relation extraction on SCIREX and template extraction on MUC-4 — and establishes a strong baseline on a third benchmark, BETTER Granular. In our experiments, we have also shown that imitation learning is a viable paradigm for these problems. We hope that our findings encourage future work to confront the challenge of dealing with documents that describe *multiple* complex events and relations head-on, rather than veiling this difficulty behind simplified task formulations.

8 Limitations

Although we believe our iterative extraction paradigm to be promising, we acknowledge that this work is not without limitations. First, ITERX features a significant number of hyperparameters. We found that these generally required some effort to tune for specific datasets, and that there was no single configuration that was uniformly the best across domains. We showed the impact of manipulating some of these hyperparameters in §5. Second, our ITERX implementation iterates over all template types in the template ontology during training and inference, which means that runtime grows linearly in the number of template types. While our framework could in principle support template *type* prediction as well (which would reduce this to $O(1)$), it does not do so in practice, and hence runtime may be long for large ontologies. However, we again stress that actual template ontologies tend to be small.

Acknowledgments

We thank Nathaniel Weir, Elias Stengel-Eskin, and Patrick Xia for helpful comments and feedback. This work was supported in part by DARPA AIDA (FA8750-18-2-0015) and IARPA BETTER (2019-19051600005). The views and conclusions contained in this work are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, or endorsements of DARPA, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1171–1179.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. 2015. Learning to search better than your teacher. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2058–2066. JMLR.org.

Tongfei Chen, Chetan Naik, Hua He, Pushpendre Rastogi, and Lambert Mathias. 2019. Improving long distance slot carryover in spoken dialogue systems. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 96–105, Florence, Italy. Association for Computational Linguistics.

Yunmo Chen, Tongfei Chen, and Benjamin Van Durme. 2020. Joint modeling of arguments for event understanding. In *Proceedings of the First Workshop on Computational Approaches to Discourse*, pages 96–101, Online. Association for Computational Linguistics.

Nancy Chinchor. 1992. Muc-4 evaluation metrics. In *Proceedings of the 4th Conference on Message Understanding*, MUC4 '92, page 22–29, USA. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Mach. Learn.*, 75(3):297–325.

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

Xinya Du, Alexander Rush, and Claire Cardie. 2021a. GRIT: Generative role-filler transformers for document-level event entity extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 634–644, Online. Association for Computational Linguistics.

Xinya Du, Alexander Rush, and Claire Cardie. 2021b. Template filling with generative transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 909–914, Online. Association for Computational Linguistics.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics.

William Gantt, Reno Kriz, Yunmo Chen, Siddharth Vashishtha, and Aaron Steven White. 2022. On event individuation for document-level information extraction. *arXiv preprint arXiv:2212.09702*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Matthew Gerber and Joyce Chai. 2010. Beyond NomBank: A study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1583–1592, Uppsala, Sweden. Association for Computational Linguistics.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of COLING 2012*, pages 959–976, Mumbai, India. The COLING 2012 Organizing Committee.

Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference- 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11179–11189.

Weiwei Gu, Boyuan Zheng, Yunmo Chen, Tongfei Chen, and Benjamin Van Durme. 2022. An empirical study on finding spans. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3976–3983, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kung-Hsiang Huang, Sam Tang, and Nanyun Peng. 2021. Document-level entity-based extraction as template generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5257–5269, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2011. Peeling back the layers: Detecting event role fillers in secondary contexts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1137–1147, Portland, Oregon, USA. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2012. Modeling textual cohesion for event extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 1664–1670.

Sarthak Jain, Madeleine van Zuylen, Hannaneh Hajishirzi, and Iz Beltagy. 2020. SciREX: A challenge dataset for document-level information extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7506–7516, Online. Association for Computational Linguistics.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Zeyang Lei, Yujiu Yang, Min Yang, and Yi Liu. 2018. A multi-sentiment-resource enhanced attention network for sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 758–763, Melbourne, Australia. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Tatjana Moor, Michael Roth, and Anette Frank. 2013. Predicate-specific annotations for implicit role binding: Corpus annotation, data analysis and evaluation experiments. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Short Papers*, pages 369–375, Potsdam, Germany. Association for Computational Linguistics.

James Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38.

Tim O'Gorman, Michael Regan, Kira Griffitt, Ulf Hermjakob, Kevin Knight, and Martha Palmer. 2018. AMR beyond the sentence: the multi-sentence AMR corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3693–3702, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 151–160, Singapore. Association for Computational Linguistics.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph LSTMs. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1171–1182, Valencia, Spain. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pages 627–635. JMLR.org.

Josef Ruppenhofer, Caroline Sporleder, Roser Morante, Collin Baker, and Martha Palmer. 2010. SemEval-2010 task 10: Linking events and their participants in discourse. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 45–50, Uppsala, Sweden. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Beth M. Sundheim. 1991. Overview of the third Message Understanding Evaluation and Conference. In *Third Message Uunderstanding Conference (MUC-3): Proceedings of a Conference Held in San Diego, California, May 21-23, 1991.*

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. pages 5998–6008.

Arun Venkatraman, Martial Hebert, and J. Andrew Bagnell. 2015. Improving multi-step prediction of learned time series models. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 3024–3030. AAAI Press.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 Multilingual Training Corpus LDC2006T06. *Linguistic Data Consortium.*

Patrick Xia, Guanghui Qin, Siddharth Vashishtha, Yunmo Chen, Tongfei Chen, Chandler May, Craig Harman, Kyle Rawlins, Aaron Steven White, and Benjamin Van Durme. 2021. LOME: Large ontology multilingual extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 149–159, Online. Association for Computational Linguistics.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Mahsa Yarmohammadi, Shijie Wu, Marc Marone, Haoran Xu, Seth Ebner, Guanghui Qin, Yunmo Chen, Jialiang Guo, Craig Harman, Kenton Murray, Aaron Steven White, Mark Dredze, and Benjamin Van Durme. 2021. Everything is all it takes: A multipronged strategy for zero-shot cross-lingual information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1950–1967, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Terminology

Information Extraction is rife with vague and competing terms for similar concepts, and we recognize some hazard in introducing *generalized template extraction* (GTE) into this landscape. To head off possible confusion, we highlight two important differences between this problem and the well established problem of *event extraction* (EE).

First, EE requires identifying *lexical* event triggers, whereas GTE does not, as template instances do not necessarily have one specific lexical anchor. A document describing a terrorist attack may only *explicitly* describe a series of bombings, and a document describing an epidemic may only *explicitly* state that thousands of people have contracted a particular disease. This property holds of all three datasets we focus on, and can be seen in both Figure 1 and Figure 2. Template anchors are not annotated either for MUC-4 or for SciREX. And while they are annotated for BETTER Granular, they do not factor into scoring. This contrasts with major EE datasets, such as ACE or PropBank, for which typed lexical triggers *must* be extracted.

Second, we take GTE to be a fundamentally document-level task: templates concern events described over an entire document. In practice, EE has historically referred to extraction of predicate-argument structures within a single sentence. One could conceivably argue that this usage has begun to change with the recent interest in argument linking datasets like RAMS (Ebner et al., 2020) and WikiEvents (Li et al., 2021), in which arguments may appear in different sentences from the one containing their predicate. Even so, these cross-sentence arguments are still arguments *of a particular predicate*, *in a particular sentence*. Moreover, the overwhelming majority of arguments in these datasets are sentence-local (Ebner et al., 2020). As emphasized above, templates are not necessarily anchored to particular lexical items. For this reason, they also do not necessarily exhibit the level of locality one finds in EE.

These differences are what motivate the use of CEAF-REE as an evaluation metric, in contrast to the precision, recall, and F1 scores for events and arguments that are typically reported for EE. In brief, it simply is not possible to compute these for GTE in the same way as they are computed for EE. We elaborate on this point in Appendix D.

B Model Training and Hyperparameters

We implemented our models in PyTorch (Paszke et al., 2019) and AllenNLP (Gardner et al., 2018). We trained all our models with a single NVIDIA RTX6000 GPU. For all experiments that reproduce prior works, we trained models until full convergence under the patience settings provided in the publicly released code. For all IterX models, we trained and tuned hyperparameters under our grid's limit of 24 hours per run, with which we were able to obtain solid performance on all datasets. We performed hyperparameter search manually and report the best performing hyperparameters and the bounds we searched in Table 5, Table 6, and Table 7.

C Dataset Details

C.1 MUC-4

The MUC-4 dataset features a total of 1,700 English documents (1,300 for train and 200 each for dev and test) concerning geopolitical conflict and terorrism in South America. Documents are annotated with templates of one of six kinds — `Attack`, `Arson`, `Bombing`, `Murder`, `Robbery`, and `ForcedWorkStoppage` — and may have multiple templates (often of the same type) or no templates at all. All templates contain the same

Name	Best	Search Bounds
Encoder	$\text{T5}^{\text{enc}}_{\text{large}}$	$\{\text{T5}^{\text{enc}}_{\text{base}}, \text{T5}^{\text{enc}}_{\text{large}}, \text{BART}^{\text{enc}}_{\text{base}}, \text{BART}^{\text{enc}}_{\text{large}}, \text{BERT}_{\text{base}}\}$
LR	3×10^{-5}	$\{1 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$
Encoder LR	1×10^{-5}	1×10^{-5}
α	0.5	$[0, 1]$
β	1.0	$\{0, 1.0, \infty\}$
γ	1.0	$\{1.0\}$
π Type	Joint	{Independent, Joint}
Max #Iteration	10	$\{10, 30\}$
Training Spans	Gold	{Gold, Upstream}
Avg. training time	3 hrs	
Validation Metric	CEAF-RME	
# of parameters*	~362 million	

Table 5: Hyperparameters and other reproducibility information for SCIREX. “LR” denotes learning rate, “π Type” indicates which policy network architecture is used (see §3.2), “Max #Iteration” sets the maximum number of iterations that the model is allowed to perform, and “Training Spans” determines whether the training spans come from gold annotations or the intersection of gold spans and those predicted by the span finding module. *The number of trainable parameters include the parameters from the (best) encoder.

Name	Best	Search Bounds
Encoder	$\text{T5}^{\text{enc}}_{\text{large}}$	$\{\text{T5}^{\text{enc}}_{\text{base}}, \text{T5}^{\text{enc}}_{\text{large}}, \text{BART}^{\text{enc}}_{\text{base}}, \text{BART}^{\text{enc}}_{\text{large}}, \text{BERT}_{\text{base}}\}$
LR	3×10^{-5}	$\{1 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$
Encoder LR	1×10^{-5}	1×10^{-5}
α	0.6	$[0, 1]$
β	1.0	$\{0, 1.0, \infty\}$
γ	1.0	$\{1.0\}$
π Type	Independent	{Independent, Joint}
Max #Iteration	14	$\{14\}$
Training Spans	Upstream	{Gold, Upstream}
Avg. training time	20 hrs	
Validation Metric	CEAF-RME	
# of parameters	~379 million	

Table 6: Hyperparameters and other reproducibility information for MUC-4.

Name	Best	Search Bounds
Encoder	$\text{T5}^{\text{enc}}_{\text{large}}$	$\{\text{T5}^{\text{enc}}_{\text{base}}, \text{T5}^{\text{enc}}_{\text{large}}, \text{BART}^{\text{enc}}_{\text{base}}, \text{BART}^{\text{enc}}_{\text{large}}\}$
Optimizer	AdamW	{AdamW}
LR	3×10^{-5}	$\{1 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$
Encoder LR	1×10^{-5}	1×10^{-5}
α	048	$[0, 1]$
β	∞	$\{0, 1, \infty\}$
γ	1.0	$\{1.0\}$
π Type	Joint	{Independent, Joint}
Max #Iteration	10	$\{10, 30\}$
Training Spans	Gold	{Gold, Upstream}
Avg. training time	24 hrs	
Validation Metric	BETTER combined score	
# of parameters	~591 million	

Table 7: Hyperparameters and other reproducibility information for Granular.

slots. While the original data contains numerous slots, it has become standard practice to evaluate systems on just five of these (apart from the slot for the template's type), all of which take entity-valued fillers: `Perpetrator(Individual)`, `Perpetrator(Organization)`, `Victim`, `Weapon`, and `Target`.

C.2 BETTER Granular

The BETTER Granular dataset contains documents spanning a number of domains, and, like MUC-4, focuses on six types of complex event, though covering different topics: protests, epidemics, natural disasters, acts of terrorism, incidents of corruption, and (human) migrations. However, Granular is substantially more difficult than MUC-4 in several ways. First, each template type is associated with a *distinct* set of slots. Second, only some of the slots take entities as fillers, whereas others take events, boolean values, or one of a fixed number of strings. Finally, the formal evaluation setting for Granular — which we do not adopt in this paper — is zero-shot and cross-lingual: systems trained only on English documents are evaluated exclusively on documents in a different target language.[18] The data used in our experiments is English-only and comprises the “train,” “analysis,” and “devtest” splits from Phase II of the BETTER program, for which the target language is Farsi.

C.3 SCIREX

The 4-ary SCIREX relation extraction task seeks to idenfity entity 4-tuples that describe a `metric` used to evaluate a `method` applied to an ML `task` as realized by a specific `dataset` — e.g. (*span F1*, *BERT*, *SRL*, *ACE 2005*). The challenge of SCIREX lies not only in these pieces of information tending to be widely dispersed throughout an article, but also in the fact that only tuples describing *novel* work presented in the paper (and not merely *cited* work) are labeled as gold examples. Following Huang et al. (2021), we frame this as a template extraction task, treating each 4-tuple as a template with four slots.

D Model Evaluation Details

A key consideration that arises in evaluating generalized template extraction is the need to align predicted and reference templates: a given predicted template may be reasonably similar to multiple different reference templates, and one must decide on a *single* template to use as the reference for each predicted one. Generalized template extraction is similar in this respect to coreference resolution, in

---

[18] Systems are permitted to use machine-translated versions of this documents, but gold data in the target language is prohibited.

which predicted *entities* may (partially) match multiple reference entities, and one must determine a ground truth alignment. Importantly, this consideration also renders metrics that are traditionally reported for *event extraction* — namely, event and argument precision, recall, and F1 — inappropriate. This is because event extraction is fundamentally a span labeling problem, and the identity of the appropriate reference span is always clear for a given predicted span: either a reference span with the same boundary and type exists or it does not. By contrast, the mapping from prediction to reference for templates is only this transparent in cases of perfectly accurate predictions.

All the evaluation metrics presented in this appendix are, at base, minimal extensions of precision, recall, and F1 to cases where event alignments are both necessary and non-trivial. For CEAF-REE in particular, the various versions of the metric that we discuss (CEAF-REE, CEAF-RME, and Du-CEAF-RME) merely reflect differences in how this alignment should be performed and whether the template type should be treated in the same way as slot types for reporting purposes.

D.1 MUC-4

MUC-4 evaluation presents a special challenge, owing to its long and complicated history, and to terminological confusion.[19] Here, we discuss CEAF-REE (Du et al., 2021a), the current standard metric for MUC-4 evaluation. We begin with definitions, following with a discussion of some of its problems, and conclude with an extended presentation of our CEAF-RME variant, introduced in §4.

D.1.1 CEAF and CEAF-REE: Definitions

The CEAF-REE metric, introduced by Du et al. (2021a), has since been adopted as the standard evaluation metric for MUC-4 (Du et al., 2021b; Huang et al., 2021). To our knowledge, no official scoring script has ever been released for MUC-4, although the metrics used as part of the original evaluation are described in detail in Chinchor (1992). CEAF-REE does not attempt to implement these original metrics, but is rather a lightly adapted version of the widely used CEAF metric for coreference resolution, proposed in Luo (2005).[20] CEAF computes an alignment between reference ($\mathcal{R}$) and system-predicted ($\mathcal{S}$) entities, with each entity represented by a set of coreferent mentions, and with the constraint that each predicted entity is aligned to *at most* one reference entity. This is treated as a maximum bipartite matching problem, in which one seeks the alignment that maximizes the sum of an entity-level similarity function $\phi(R, S)$ over all aligned entities $R \in \mathcal{R}$ and $S \in \mathcal{S}$ within a document. In principle, CEAF is agnostic to the choice of ϕ, though it is generally desirable that $\phi(R, S) = 0$ when $\nexists x \in S$ such that $x \in R$ and that $\phi(R, S) = 1$ when $R = S$, for reasons described in Luo (2005). In practice, the ϕ_4 similarity function is most commonly used, defined as the Dice coefficient (or F1 score) between R and S:

$$\phi_4 := \frac{2|R \cap S|}{|R| + |S|} \tag{9}$$

The version of CEAF that uses ϕ_4 is sensibly denoted CEAF$_{\phi_4}$, or sometimes CEAF$_e$ (*e* for *entity*). CEAF-REE differs from CEAF$_{\phi_4}$ in the following two ways:

- All entities are aligned *within role, conditioned on matching template type*. E.g. only predicted entities for the `Victim` slot in `Bombing` templates would be considered valid candidates for alignment with entities filling the `Victim` slot in the reference templates of the same type.

- A binary similarity function ϕ_{REE} is used, defined as follows:

$$\phi_{\text{REE}}(R, S) := \begin{cases} 1, & \text{if } S \subseteq R \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

D.1.2 CEAF-REE: Problems and Solutions

Our principal concerns with CEAF-REE lie with how it has so far been reported and implemented, and with challenges in extending it to the full template extraction task, in which multiple templates of the same type may be present in a document. We elaborate on two issues discussed briefly in §4 and also introduce a third.

First, previous work that reports CEAF-REE treats the template type merely as another slot, with template type labels treated as special "entities" that may fill this slot. This is not necessarily problematic in itself: template type-level metrics are valuable for evaluating system performance. However, it is problematic when reporting (micro

---

[19] Early writing on MUC-4 used the term *entity* to refer to what the IE community would now call a *mention*. We suspect this is the source of a great deal of confusion.

[20] Luo's motivations for proposing CEAF actually derive in large part from observed shortcomings with the original MUC-4 F1 score. See Luo (2005) for details.

or macro) average CEAF-REE figures *across* slots, as these works do. This is because incorporating the scores for template type into the average elides the distinction between roles (slots) and the kind of event being described (the template type). Moreover, the alignment between slot-filling entities is also *already* conditioned on a match between the template types. There are thus two distinct ways in which information about a system's predictive ability with respect to template type end up in a slot-level average CEAF-REE score. This results in reported values that are very difficult to interpret, and potentially misleading to the extent that these features of CEAF-REE implementations are not made apparent in writing.

Second, the constraint that at most one predicted entity be aligned to each reference entity — stipulated in the metric definition (CEAF-REE$_{\text{def}}$) — is not enforced in the implementation (CEAF-REE$_{\text{impl}}$). Practically, this means that the alignment shown in Figure 6 would receive full credit, whereas it ought to receive a precision score of only 0.75, as Du et al. (2021a) describe. As we argue in §4, we believe this constraint to be overly strict. But this point aside, the discrepancy between definition and implementation is clearly troubling in itself.

Third, full template extraction introduces a *second* maximum bipartite matching problem, which requires aligning predicted and reference *templates* of the same type, and which CEAF-REE (either CEAF-REE$_{\text{def}}$ or CEAF-REE$_{\text{impl}}$) is not natively equipped to handle, given that it operates at the level of slots. Du et al. (2021b) reports CEAF-REE for GTT under an optimal template alignment, but this is obtained via brute-force, enumerating and evaluating every possible alignment, including those between templates of different types. The similarity function, (call it $\phi_{\text{TEMPLATE}}(T_R, T_S)$) that they use for the template alignment problem is *itself* the cross-slot average CEAF-REE$_{\text{impl}}$ score for predicted template T_S and reference template T_R. This brute-force template alignment, in conjunction with the hierarchical maximum bipartite matching problem, results in prohibitively long scorer execution times in cases where there is only a modest number of predicted or reference templates of the same type.[21]

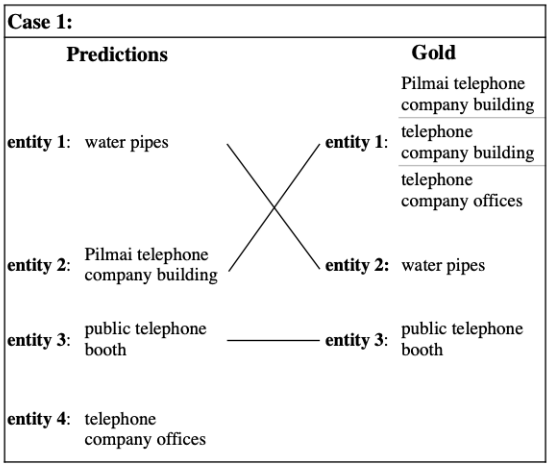


Figure 6: An example alignment between predicted and reference entities from Du et al. (2021a). In past implementations of CEAF-REE, this alignment would receive full credit, rather than being penalized for precision ($P = 0.75$).

In addition to our implementation of CEAF-RME (see below), we also present the first *correct* implementation of CEAF-REE$_{\text{def}}$ that fully addresses the first two points above: template types are no longer treated as additional slots and the entity-level alignment constraint is enforced. On the third point, our implementation efficiently computes optimal template alignments using the Kuhn-Munkres algorithm (Kuhn, 1955; Munkres, 1957). However, even with this efficient implementation, solving such hierarchical maximum bipartite matching problems is computationally intensive.

D.1.3 Coreference and CEAF-RME

As CEAF was designed for coreference, it is perhaps unsurprising that coreference considerations introduce a further wrinkle for CEAF-REE. None of the three models described in this work (including ITERX) performs entity coreference. This clearly presents a problem because CEAF-REE$_{\text{def}}$ is an entity-level metric. One way to score these models is simply to treat each extracted mention as a singleton entity and use CEAF-REE$_{\text{def}}$ exactly as defined, and we report these scores in the main text. However, reporting *only* CEAF-REE$_{\text{def}}$ would be undesirable for several reasons:

- It would render our results incomparable to past work, which reports only CEAF-REE$_{\text{impl}}$.
- It would put our work at odds with the overwhelming majority of the template extraction literature, where evaluation criteria focus on string

---

[21] Only CEAF-REE$_{\text{def}}$ requires solving a two-level maximum-bipartite matching problem. Since CEAF-REE$_{\text{impl}}$ does not enforce the entity alignment constraint, these alignments will not necessarily be bipartite.

matching between predicted and reference *mentions*. (The original MUC-4 evaluation only required systems to extract a single representative mention for each entity — not to identify all such mentions.)

- The constraint that at most one predicted entity be aligned to a given reference entity would yield punishingly low scores for systems that are highly effective at extracting relevant spans, but that simply do not perform the additional step of coreference. It is also not especially relevant for certain benchmarks, like SCIREX, where coreference is less central to the task.

For these reasons, we disfavor a template extraction metric that *requires* template extraction systems to do coreference. These considerations motivate our introduction of CEAF-RME (*role-filler mention extraction*) — that makes a minimal modification to CEAF-REE$_{\mathrm{def}}$ to address (1) and (2) above. CEAF-RME treats system-predicted mentions as singleton entities, but *deliberately* relaxes the alignment constraint, potentially allowing multiple predicted singletons to map to the same reference entity, effectively pushing the burden of coreference into the metric. We believe CEAF-RME is consistent with what template extraction research has *in fact* historically cared about (identifying *mentions* that fill some slot) while correcting implementation problems with CEAF-REE that produce misleading results.

The micro-average CEAF-RME results that we report on MUC-4 in the main body of the paper are micro-average CEAF-RME scores *under an optimal template alignment* (using CEAF-RME as the template similarity function), which is efficiently obtained using the Kuhn-Munkres algorithm.

D.2 SCIREX

We use the same CEAF-RME implementation for scoring SCIREX as we use for MUC-4. We simply treat the SCIREX 4-tuples as 4-slot templates, following (Huang et al., 2021). Since SCIREX 4-ary relations are untyped, all SCIREX "templates" are similarly untyped for scoring purposes.

D.3 BETTER Granular

Evaluation for the BETTER Granular task bears some core similarity to CEAF-REE$_{\mathrm{def}}$ in that relies on obtaining the alignment between system and reference templates that maximizes some similarity function that decomposes over slot fillers. And just as with (our corrected implementation of) CEAF-REE$_{\mathrm{def}}$, this is achieved via the Kuhn-Munkres algorithm. However, Granular scoring differs from CEAF-REE$_{\mathrm{def}}$ in four key respects. First, the overall system score — referred to as the *combined score* — incorporates both a slot-level F1 score *and* a template-level F1 score:

$$\text{CombinedScore} := \text{TypeF1} \times \text{SlotF1}$$

Only exact matches between system and reference templates types are awarded credit. It is worth noting that because this score does not decompose over template pairs, it cannot be optimized directly using Kuhn-Munkres. In practice, what is optimized is *response gain* — the number of correct slot fillers minus the number of incorrect ones — which provably yields alignments that optimize the combined score within a probabilistic error bound.

The remaining three key differences relate to the calculation of the slot-level F1. For one, Granular slots are not exclusively entity-valued, but may also be event-, (mixed) event-and-entity-, boolean-, and (categorical) string-valued, and different similarity functions must be employed in these different cases. For another, where CEAF-REE defines mentions by their string representation, the Granular score defines mentions based on document offsets. Finally, Granular also requires extraction of temporal and irrealis information for slots, and this in turn impacts the SlotF1 score.

Borrowing terminology from the discussion of MUC-4 above, we describe below how $\phi(R, S)$ is calculated for some generic reference slot filler R and system-predicted slot filler S for slots of different types.

Boolean and Categorical Values For boolean- and categorical-string valued slots (i.e., slots taking on one of a predefined set of values). $\phi(R, S) = 1$ if there is an exact match between the system and reference fillers and is 0 otherwise.

Entities Unique among the three tasks discussed in this paper, Granular features an explicit preference for *informative* arguments in its scoring structure. In particular, (proper) *name* mentions of an entity are worth more than *nominal* mentions, which in turn are worth more than *pronominal* ones.[22] Thus, if Barack Obama were represented by the reference entity $\{Obama, the\ former\ President, he\}$,

---

[22] This is precisely the hierarchy described for the *informative argument extraction* task in Li et al. (2021).

full credit would be awarded for returning only the mention *Obama*, less credit for *the former President*, and still less for *he*. Exact point values depend on the mentions present in the reference entity:

- Correct name mentions always receive full credit ($\phi(R, S) = 1$)
- Correct nominal mentions receive half-credit ($\phi(R, S) = 0.5$) if the reference entity additionally contains a name mention, and receive full credit otherwise.
- Correct pronominal mentions receive quarter-credit ($\phi(R, S) = 0.25$) if the reference entity additionally contains *both* a name and a nominal mention, and half-credit if only a nominal mention is featured. (Note that entities will never feature only pronominal mentions.)

Events Some Granular slots require *events* as fillers. Like entities, events are represented as sets of mentions (event *anchors* or *triggers*). Unlike entities, there is no informativity hierarchy for events. Furthermore, while event coreference is not a part of the Granular task, annotations for event coreference are nonetheless provided for scoring purposes: $\phi(R, S) = 1.0$ iff S contains only mentions belonging to events in the set of gold coreferent events R, and is 0 otherwise, akin to ϕ_{REE}.

Mixed Entities and Events Some slots may take a mix of events and entities as fillers. Since systems must indicate whether predicted mention clusters are entity- or event-denoting, the same similarity criteria for events and entities as described above are used to compute ϕ for events and entities that fill these slots.

Temporal and Irrealis Information One of the features of Granular that makes it decidedly more difficult than either MUC-4 or SciREX is the requirement to extract information relating to the time and irrealis status of an event when such information is available in the document. This information is encapsulated in special `time-attachments` and `irrealis` fields associated with each slot-filling entity or event. The former is given as a set of temporal expressions that describe the time at or during which the filler satisfied the role denoted by the slot (e.g. when individuals filling the `tested-count` slot in the Epidemic template were tested for the disease). The latter is given as one of a set of strings that describe whether or how the filler satisfied the role denoted by the slot: `counterfactual`, `hypothetical`, `future`, `unconfirmed`, `unspecified`, and `non-occurrence`. `time-attachments` and `irrealis` are each worth 0.25 points, where exact matches are required for full credit on either and where zero points are awarded otherwise. For slots for which `time-attachments` and `irrealis` are required, the value of ϕ appropriate to its filler type is scaled by 0.5 such that the maximum overall score $\phi(R, S)$ for a given filler — factoring in `time-attachments`, `irrealis`, *and* event or entity similarity — is 1.

CLICK: Contrastive Learning for Injecting Contextual Knowledge to Conversational Recommender System

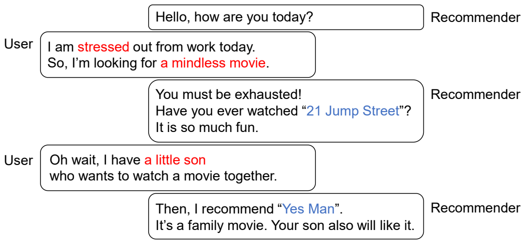
Hyeongjun Yang, Heesoo Won, Youbin Ahn, and Kyong-Ho Lee
Department of Computer Science, Yonsei University
`{edbm95, hswon97, ybahn, khlee98}@yonsei.ac.kr`

Abstract

Conversational recommender systems (CRSs) capture a user preference through a conversation. However, the existing CRSs lack capturing comprehensive user preferences. This is because the items mentioned in a conversation are mainly regarded as a user preference. Thus, they have limitations in identifying a user preference from a dialogue context expressed without preferred items. Inspired by the characteristic of an online recommendation community where participants identify a context of a recommendation request and then comment with appropriate items, we exploit the Reddit data. Specifically, we propose a **C**ontrastive **L**earning approach for **I**njecting **C**ontextual **K**nowledge (**CLICK**) from the Reddit data to the CRS task, which facilitates the capture of a context-level user preference from a dialogue context, regardless of the existence of preferred item-entities. Moreover, we devise a relevance-enhanced contrastive learning loss to consider the fine-grained reflection of multiple recommendable items. We further develop a response generation module to generate a persuasive rationale for a recommendation. Extensive experiments on the benchmark CRS dataset show the effectiveness of CLICK, achieving significant improvements over state-of-the-art methods.

1 Introduction

Recommender systems help users find potential items of interest in the rapidly expanding pool of candidates. To do so, the recommender systems aim to accurately identify user preferences. Thus, traditional recommender systems (Guo et al., 2020) utilize click or purchase data to obtain user preferences. However, such traditional approaches lack dynamically modeling user preferences since the data has a temporal validity and does not offer a diverse clue to user information. Conversational recommender system (CRS) is a research area that overcomes the limitation of the traditional recommender systems by capturing users' interests through natural conversation. The conventional CRSs (Chen et al., 2019; Ma et al., 2021; Zhou et al., 2020a) mainly derive a user preference from item-entities, which appear in a conversation and exist in a knowledge graph (KG). However, users often express their preferences without preferred items. For example, in Figure 1, the user depicts his/her requirement (*a mindless movie*), feeling (*stressed*), and situation (*with a little son*). In order for an accurate recommendation of a movie (*Yes Man*) to be derived, the recommender must capture that the user wants those characteristics of a movie, "*a movie to watch when stressed*", "*a mindless movie*" and "*a movie to watch with a little son*". Therefore, merely depending on item-entities for user preference identification is insufficient. We argue that a dialogue context also provides a user preference, regardless of the existence of item-entities.



Hello, how are you today?

I am stressed out from work today.
So, I'm looking for a mindless movie.

You must be exhausted!
Have you ever watched "21 Jump Street"?
It is so much fun.

Oh wait, I have a little son
who wants to watch a movie together.

Then, I recommend "Yes Man".
It's a family movie. Your son also will like it.

Figure 1: An example of a user-recommender conversation for movie recommendations. The important information for a recommendation is written in red, and the recommended items are written in blue.

Motivated by the characteristic of an online recommendation community where participants identify a context of a recommendation request and then comment with appropriate items, our method exploits the Reddit data to extract the contextual knowledge that enables inferring a preferred item in a KG from a recommendation request. Thus, we propose a Contrastive Learning approach for Injecting Contextual Knowledge (**CLICK**), which

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1875–1885
May 2-6, 2023 ©2023 Association for Computational Linguistics

captures not only the entity-level preference derived from mentioned entities but also identifies the context-level preference from a dialogue context by utilizing both a KG and Reddit data.

However, the different modality between recommendable item-entities in a KG and request texts in the Reddit data hinders CLICK from learning contextual knowledge. Thus, we alleviate the different modalities via contrastive learning to align the pairs of a recommended item-entity and a request text. Moreover, different from the conventional contrastive learning that classifies the pairs into only positive or negative, our distinction is to consider the relative relevance among positive pairs. For example, if a seeker asks a recommendation community for a movie to watch when feeling down, several movies can be recommended. While some movies are related to many participants, others may be referenced by a few people. Thus, to reflect differing relativity of multiple recommendable items from a request, we design a relevance-enhanced contrastive learning loss function. Consequently, leveraging the learned contextual knowledge, CLICK makes accurate recommendations based on the user preferences extracted from both entity- and context-level perspectives. In the response generation task, we develop a response generator that produces a suitable explanation with a recommended item based on a captured requirement from a dialogue context.

Our contributions are summarized as follows: (1) We propose the knowledge injection method that facilitates the inference of a recommendable item based on a dialogue context in order to identify a comprehensive user preference. (2) We design the relevance-enhanced contrastive learning loss that promotes the fine-grained reflection of positive pairs according to the relevance. (3) We further develop a contextual knowledge-enhanced recommender module and a response generation module, which captures a comprehensive user preference and produces an explanation based on a user's needs, respectively. (4) The proposed model outperforms baselines, especially regardless of the existence of item-entities in a user utterance.

2 Related Work

2.1 Conversational Recommender Systems

CRS (Gao et al., 2021) allows obtaining user preferences through dynamic interaction, overcoming the limitations of traditional recommender systems (Guo et al., 2020) that are heavily dependent on static interaction history. The research area can be broadly categorized into two groups: template-based and natural language-based CRSs. Template-based methods (Deng et al., 2021; Lei et al., 2020; Zhou et al., 2020d) interact with a user, following a slot-filling approach. Although many template-based CRSs have been proposed, they suffer from producing inflexible response due to the inherent template scheme.

On the other hand, natural language-based CRS (Chen et al., 2021; Zhou et al., 2020c) allows users to depict their needs in a free text. Such approaches focus on how to extract knowledge from external data (e.g., historical data, KGs, and review data) and how to utilize the knowledge to capture a user preference and generate a persuasive response since the CRS benchmark dataset contains repetitive and limited utterances. Following the stream of CRS research, Zhou et al. (2020b) utilize historical interaction data to enhance a sequence of preferred items. Chen et al. (2019) and Zhou et al. (2020a) exploit KGs to capture user preferences with mentioned entities. Ma et al. (2021), Moon et al. (2019), and Zhou et al. (2021) leverage explicit reasoning on a KG based on the mentioned entities. Lu et al. (2021) and Zhou et al. (2022) utilize review data to enrich insufficient item information over a KG. However, despite the diverse use of external data, the current methods still lack capturing a user's needs depicted in a free text since they identify preferences mainly with mentioned item-entities that exist in a KG. To resolve the limitation, we propose a contrastive learning approach for injecting knowledge into the CRS task by utilizing textual recommendation data (Reddit) and a KG. Thus, our approach captures user preferences from both mentioned entities and a dialogue context, taking full advantage of CRS in which users can express their preferences freely in natural language.

2.2 Contrastive Learning

Another line that motivates our work is contrastive learning, which is a dominant approach in self-supervised learning. Especially in multi-modal scenarios, contrastive learning (Oord et al., 2018) plays a vital role in refining feature representation. It aims at forcing representations to have a smaller distance between positive pairs and a greater distance between negative pairs than the positive ones. Due to the efficacy of contrastive learning strategy,

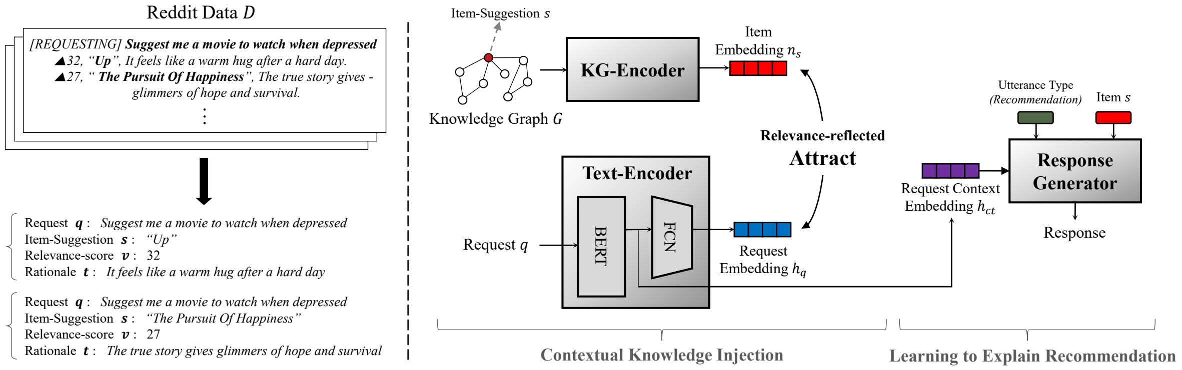


Figure 2: The pre-training stage of CLICK which consists of two encoders and a response generator.

it has been widely used in various research fields such as multilingual text-to-video search (Huang et al., 2021), visual pre-training (Yuan et al., 2021), and other domains (Xia et al., 2021; Zolfaghari et al., 2021). Especially, in recent research on CRS, Zhou et al. (2022) adopt contrastive learning to fuse multi-type external data. Likewise, we refine our multi-type external data representations by contrastive learning. However, different from the existing contrastive learning loss, we further devise relevance-enhanced contrastive learning loss to consider multiple recommendable items, of which each has a different relative relevance.

3 Pre-training for Contextual Knowledge Injection

We introduce the training scheme of CLICK that follows a two-stage mechanism: 1) a pre-training stage for the contextual knowledge injection and 2) a fine-tuning stage for the conversational recommendation task.

In the two stages, we utilize DBpedia (Bizer et al., 2009) as a KG, following the existing CRSs (Chen et al., 2019; Zhou et al., 2020a; Ma et al., 2021; Zhou et al., 2022) that take advantage of KGs to capture user preferences based on factual information such as objects, concepts, and relationships among them. The KG is defined as $G = \{(e1, r, e2) | e1, e2 \in E, r \in R\}$, where a fact $(e1, r, e2)$ consists of relation r from entity $e1$ to entity $e2$ from entity set E and relation set R.

The Reddit data (Penha and Hauff, 2020) is leveraged to learn the contextual knowledge that facilitates the inference of a preferred item from a user utterance. The Reddit data is denoted as $D = \{(q, s, v, t)\}$, where q is a seeker's request text to receive a recommendation, s is an item-suggestion by a recommender, v is a relevance score of the item-suggestion, and t is an optional rationale for the item-suggestion. We describe the details of the Reddit dataset in Section 5.1.

In the pre-training stage, our approach infuses the contextual knowledge from Reddit data into a KG-encoder and a text-encoder by training them via a devised contrastive learning loss. Then, a response generator is pre-trained to equip the ability to generate an explanation for a recommendation based on the recognized needs. The pre-training stage of CLICK is described in Figure 2.

3.1 Contextual Knowledge Injection

Following previous studies (Zhou et al., 2020a; Ma et al., 2021), we encode the KG to obtain the representations of entities via R-GCN (Schlichtkrull et al., 2018) that considers neighborhoods under the specific relation type and direction. The representation of item-entity s from the KG-encoder is denoted as n_s. Then, we employ BERT (Devlin et al., 2019) and a fully-connected layer FCN to encode a request text q that contains the seeker's needs (more details in Appendix A.1). The request embedding h_q is obtained as:

$$h_q = FCN(BERT(q)). \qquad (1)$$

Next, the above two encoders are pre-trained via a contrastive learning loss, which promotes alignment of the multi-modality originating from the KG and text. That way, the text encoder is encouraged to effectively infer a preferred item from a request. Moreover, different from a conventional contrastive learning loss that classifies the pairs into only positive or negative, we build the relevance-enhanced contrastive learning to reflect multiple recommendable items with a relative relevance as follows:

$$L_{rel} = -log \sum_{s \in S} \frac{ex^{(sim_{q,s} - v_{q,s})}}{ex^{(sim_{q,s} - v_{q,s})} + \sum_{S'} ex^{(sim_{q,s'})}}, \quad (2)$$

where $v_{q,s}$ indicates the relevance score corresponding to the number of "Upvote" about the item suggestion s from the request q in positive set S, S' is a negative set, sim means cosine similarity between the item representation vector n_s and the request representation vector h_q that the above two encoders produce respectively, and $ex^{(x)}$ is exponential of x. The proposed loss function is able to attract positive pairs in proportion to the relevance score $v_{q,s}$ while repelling the negative pairs. Then the pre-trained encoders are utilized to capture a comprehensive user preference from a user utterance depicted regardless of the presence of entities.

3.2 Learning to Explain Recommendation

At this stage, the response generator is pre-trained to generate a recommendation response of the three utterance types (*question, recommendation, chit-chat*). The recommendation response should contain a recommended item and a rationale for the recommendation based on a captured user need. Thus, we adopt GPT-2 (Radford et al., 2019) as a response generator and feed a suggested item s and the utterance type (*recommendation*) to a token embedding layer of GPT-2. The first layer of the decoder A^1 then conducts self-attention with the token embeddings. Next, having obtained the request context embedding h_{ct} from the text-encoder (only $BERT$), cross-attention is conducted in the next decoder block to generate the explanation for the item-suggestion based on the needs extracted from the request of a seeker. The cross-attention layer of the decoder can be summarized as:

$$A^2 = FFN(MultiHead(O_{A^1}, h_{ct}, h_{ct})), \quad (3)$$

where the multi-head attention layer $MultiHead(Q, K, V)$ takes query Q, key K and value matrix V as input, and O_{A^1} is the output of the first decoder layer A^1. The response generator is pre-trained to generate the rationale t for the item-suggestion s conditioned on the request q and the utterance type ut via maximum likelihood estimator (MLE) generation loss:

$$L_{exr} = -\sum_{j=0}^{J} \log p(t_j | t_{<j}, q, s, ut). \quad (4)$$

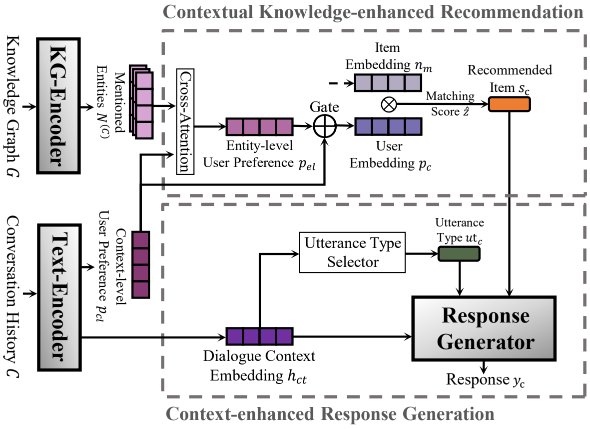


Figure 3: The fine-tuning stage of CLICK utilizing the pre-trained encoders and response generator for the conversational recommendation task.

4 Fine-tuning for Conversational Recommendation Task

In this section, the fine-tuning strategy of CLICK for the CRS task is introduced. Figure 3 illustrates the working flow of CLICK, leveraging the two pre-trained encoders and response generator from the pre-training stage. CLICK consists of two key components: (1) contextual knowledge-enhanced recommender module, which even understands user's needs from a dialogue context, and (2) context-enhanced response generation module, which generates a persuasive response based on a captured user preference. Prior to introducing the details of the fine-tuning scheme, we formulate a conversation history $C = \{x_1, y_1, ..., x_c\}$, where x_i is a user utterance, and y_i is a system utterance at turn-i. The goal of CLICK is to recommend a proper item s_c and generate the system response y_c, understanding the conversation history C.

4.1 Contextual Knowledge-enhanced Recommendation

The pre-trained text-encoder is leveraged to identify context-level user preference (e.g., *a movie to watch when feeling down*) that could not be captured with mentioned entities. In detail, the conversation history C is fed to the text-encoder, and we obtain the context-level user preference as $p_{cl} = FCN(BERT(C))$.

Next, we utilize mentioned entities to identify entity-level user preference (e.g., *a movie like Yes-Man*). In our implementation, we conduct cross-attention with a context-level user preference and entities, being different from the previous approaches that ignore the relative importance among mentioned entities based on a dialogue context. Thus, the cross-attention score is denoted as:

$$\alpha = softmax(q \cdot k^{\mathsf{T}} / \sqrt{d_k}), \\ q = p_{cl} \cdot W_q,\ k = n_e \cdot W_k, \tag{5}$$

where W_q,W_k are weight metrices, n_e is an entity embedding from the mentioned entities $N^{(C)}$, and the context-level user preference p_{cl} is from the text-encoder. Then, we obtain the entity-level user preference p_{el} as follows:

$$p_{el} = \alpha \cdot N^{(C)}, \tag{6}$$

Then, we adopt a gate mechanism to obtain the final user preference p_c at c-th turn by combining the entity-level and context-level user preferences as follows:

$$p_c = \beta \cdot p_{el} + (1 - \beta) \cdot p_{cl}, \\ \beta = \sigma(W_{gate} \cdot [p_{el}; p_{cl}]), \tag{7}$$

where W_{gate} is a weight matrix, and $[;]$ indicates the concatenation operator. Consequently, we conduct the inner product of user preference p_c and the representation n_m of item m to predict the matching score:

$$\hat{z}(c, m) = p_c \cdot n_m^{\top}. \tag{8}$$

The recommender module is fine-tuned with a cross-entropy loss as:

$$L_{rec} = -\sum_{c=1}^{C} \sum_{m=1}^{M} [z_{cm} \cdot \log \hat{z}(c, m) \\ - (1 - z_{cm}) \cdot \log (1 - \hat{z}(c, m))], \tag{9}$$

where z_{cm} is the true label.

4.2 Context-enhanced Response Generation

Prior to generating a response, at each turn, CLICK determines the type of an utterance (*question, recommendation, chit-chat*) given a dialogue context. Following Ma et al. (2021), we regard the determination of utterance's type as a 3-way classification problem:

$$Pr(y_c^{ut}|C) = softmax(W_{ut}^2 \cdot ReLU(W_{ut}^1 \cdot h_{ct})), \tag{10}$$

where h_{ct} is a dialogue context embedding from the BERT. We thus formulate the determination loss as:

$$L_{ut} = -\log Pr(y_c^{ut}|x_1, y_1, ..., x_c), \tag{11}$$

where y_c^{ut} is the true label of utterance type.

After obtaining the dialogue context embedding, the utterance type, and the recommended item, we utilize such to generate a persuasive response that contains a recommended item under the controlled type of an utterance. Compared to an implicit way of response generation in CRSs (Chen et al., 2019; Zhou et al., 2021), which lacks the ability to include a recommended item in the generated response, the explicit way of response generation by Ma et al. (2021) feeds the recommended item and the utterance type as input into the response generator, which guarantees the inclusion of the item under the controlled response type. However, the explicit approach hinders generating a response based on a dialogue context. Thus, different from Ma et al. (2021), we further infuse the dialogue context implicitly into the response generator with the cross-attention mechanism to generate a response containing an identified dialogue context.

In the same manner as the pre-training process of the response generator, an utterance type and a recommended item is fed to the token embedding layer. The first layer of the decoder conducts self-attention with the token embeddings. Then, we enforce the module to generate a response based on the user's needs via the cross-attention mechanism with the dialogue context embedding h_{ct}, which is identical to the one in Equation 3. We formulate the response generation loss as:

$$L_{gen} = -\sum_{u=0}^{U} \log p(y_{u,c}|y_{<u,c}, C, s_c, ut_c), \tag{12}$$

where U is the length of a target response, $y_{u,c}$ is a generated u-th token of system response, C is a conversation history, and s_c is the recommended item at c-th turn. ut_c indicates the type of utterance. Additionally, the implementation details are in Appendix A.2.

5 Experimental Setup

5.1 Dataset

REDIAL is a benchmark CRS dataset released by Li et al. (2018). It was constructed through Amazon Mechanical Turk (AMT). The AMT workers play the role of a seeker/recommender. In daily talk, a seeker explains his/her tastes about movie and asks for movie suggestions. A recommender tries to understand the tastes and recommends movies. Accordingly, it consists of 10,006 conversations

Model	Standard Setting			No Mentioned Item Setting			Diversity
	R@1	R@10	R@50	R@1	R@10	R@50	
REDIAL	2.3	13.2	29.7	–	–	–	5.8
KBRD	3.0	15.8	33.8	1.5	5.1	12.2	11.2
KGSF	3.9	18.3	37.8	2.6	9.7	20.1	12.2
CR-Walker	4.0	18.7	37.6	2.4	10.2	19.8	14.9
C2CRS	5.2	22.9	40.7	2.1	8.9	18.3	13.9
GPT-2	2.3	14.5	31.5	1.6	7.9	16.3	9.9
GPT-2 (+*Reddit*)	2.5	15.2	32.1	1.9	8.5	17.1	11.1
BERT	3.0	15.3	34.4	1.9	8.7	17.5	10.3
BERT (+*Reddit*)	3.2	16.9	35.2	2.1	9.2	18.3	11.8
BART	3.3	17.1	36.3	2.1	9.0	18.0	11.4
BART (+*Reddit*)	3.6	18.3	37.5	2.3	9.5	18.9	12.7
CLICK	**5.5**	**23.8**	**43.2**	**3.5**	**14.9**	**28.9**	**18.3**
w/o relevance	5.3	23.1	41.1	3.3	13.2	26.7	16.9
w/o pre-training	4.2	19.2	38.1	2.5	10.0	20.3	14.7

Table 1: Overall performance comparison on the recommendation task.

containing 182,150 utterances and 51,699 movies. We set the training, validation, and test sets as a ratio of 8:1:1. **Reddit** is utilized as the textual recommendation data collected by Penha and Hauff (2020). Following the domain of a benchmark CRS dataset (REDIAL), we use the MovieSuggestions subreddit, which is a movie recommendation community. In the community, a seeker asks for a recommendation, and recommenders give suggestions with a rationale. Participants in the community consent to the suggested items with the "Upvote" button which is regarded as a relevance score. **Knowledge Graph** consists of 30,471 entities, 12 relations, and 392,682 triples extracted from DBpedia (Bizer et al., 2009) and MovieLens, following Ma et al. (2021). Additionally, the entities in utterances are linked to DBpedia nodes.

5.2 Baselines

We compared CLICK to the following five CRS baselines: (1) **REDIAL** (Li et al., 2018) is proposed with the CRS benchmark dataset. The model consists of an auto-encoder recommender and an RNN-based response generator. (2) **KBRD** (Chen et al., 2019) utilizes a KG to identify user preferences from mentioned entities. The transformer-based response generator uses a user representation as vocabulary bias. (3) **KGSF** (Zhou et al., 2020a) incorporates word-oriented and entity-oriented KGs to capture user preference. (4) **CR-Walker** (Ma et al., 2021) proposes tree-structured reasoning on a KG, which effectively utilizes background knowledge. It adopts GPT-2 to generate a response by feeding dialogue acts. (5) **C2CRS** (Zhou et al., 2022) leverages a KG and a review data for enriching the context information. It conducts data semantic fusion via contrastive learning. The response generator fuses the information from a KG and a review data.

Additionally, three pre-trained language models (PLMs) were tested for a fair comparison on the same data. In table 1 and 2, (+*Reddit*) indicates a PLM pre-trained on the Reddit data. The three models are as follows: GPT-2(Radford et al., 2019), BERT(Devlin et al., 2019), and BART(Lewis et al., 2020). The training details of PLMs are in Appendix A.3.

6 Experimental Results

6.1 Evaluation on Recommendation

We adopted Recall@k to measure how many predicted items were included in the set of preferred items (Standard Setting). Moreover, we compared the performance under the setting where there were no mentioned item-entities in a dialogue context (No Mentioned Item Setting). This setting aims to evaluate how well the models identify a user preference from a dialogue context without mentioned item-entities. We also evaluated recommendation diversity (Diversity), defined as the proportion of distinct recommendations.

6.1.1 Performance Comparison

Table 1 shows the experimental results on recommendation task. **Standard Setting**: KBRD, KGSF, and CR-Walker performed better compared to REDIAL by capturing user preference based on entities in KGs. Compared to the baselines, C2CRS

Model	Automatic Evlauation				Human Evaluation		
	BLEU	Dist-2	Dist-3	Dist-4	Flu.	Rel.	Info.
REDIAL	0.229	0.215	0.247	0.234	2.03	1.97	1.67
KBRD	0.231	0.281	0.376	0.438	2.18	2.19	1.98
KGSF	0.239	0.320	0.432	0.519	2.31	2.21	2.12
CR-Walker	0.271	0.361	0.493	0.573	2.54	2.37	2.31
C2CRS	0.251	0.339	0.455	0.538	2.42	2.29	2.19
GPT-2	0.247	0.352	0.474	0.521	2.35	2.01	2.04
GPT-2 (+*Reddit*)	0.261	0.359	0.481	0.536	2.39	2.09	2.02
BART	0.249	0.354	0.486	0.528	2.37	2.07	2.12
BART (+*Reddit*)	0.262	0.357	0.492	0.539	2.38	2.17	2.11
CLICK (ours)	**0.308**	**0.380**	**0.521**	**0.598**	**2.60**	**2.63**	**2.58**
w/o gen. cross-att	0.296	0.371	0.510	0.589	2.58	2.51	2.47
w/o gen. pre-training	0.289	0.364	0.503	0.583	2.53	2.45	2.44
w/o pre-training	0.281	0.359	0.497	0.579	2.48	2.39	2.40

Table 2: Overall performance comparison on the response generation task

recommended appropriate items by incorporating a KG and a review data. In contrast, CLICK achieved the best results by effectively modeling comprehensive user preference from entity-level and context-level. Another observation is from the comparison with PLMs. Although the PLMs showed impressive performances, the improvement by utilizing the Reddit data was lower than ours. Thus, it proved that CLICK efficiently extracted contextual knowledge from Reddit data and exploited it for recommendation. **No Mentioned Item Setting**: To further demonstrate the CLICK's ability of capturing a context-level user preference, we compared the baselines on this setting. KGSF and CR-Walker performed relatively better than other baselines due to the utilized word-oriented KG allowing the recognition of diverse words in a conversation. We noted that CLICK showed best performances even when there were no mentioned item-entities. This is because the contextual knowledge extracted from the Reddit data provided a significant clue to a user's needs. More details are covered in Appendix A.4. **Diversity**: CLICK recommended diverse items compared to all the baselines. The improvement was derived from our strategy of modeling user preference in a fine-grained means, considering both mentioned entities and a dialogue context.

6.1.2 Ablation Study

Pre-training with the relevance-enhanced contrastive learning loss is an essential design in our method to infer a preferable item from a user utterance. Thus, we assessed the effects of CLICK with two variants, (1) **w/o relevance**: standard contrastive learning loss (Oord et al., 2018) was used, (2) **w/o pre-training**: CLICK was only trained on REDIAL dataset without being pre-trained with the Reddit data. As shown in Table 1, the performance of CLICK *w/o relevance* was lower since it did not consider the relative relevance that helped the model to learn a fine-grained representation. CLICK *w/o pre-training* provided the evidence for the pre-training stage of learning the contextual knowledge. Besides, it showed a decline in diversity since it did not capture a fine-grained user preference with the contextual knowledge that promotes various recommendations.

6.2 Evaluation on Response Generation

BLEU (Papineni et al., 2002) and Dist-N (Li et al., 2016) were adopted to measure the word-level correspondence and diversity of responses. We further conducted a human evaluation. Five human annotators evaluated 100 generated responses on fluency, relevancy, and informativeness ranging from 0 to 3. The average score of each metric is reported.

6.2.1 Performance Comparison

Table 2 shows the experimental results on response generation task. **Automatic Evaluation**: The result proved that our method generated a better quality response with diverse words. Such improvements are derived from the explicit feeding of a recommended item and an utterance type into a response generator and the implicit injection of a context via a cross-attention mechanism. **Human Evaluation**: The responses generated from C2CRS, KGSF, and KBRD are inclined to contain safe responses that are broadly adequate for any situation, such as *"I have not seen that one"*.

This issue was reflected as the lower score on relevancy and informativeness compared to CR-Walker. While CR-Walker generated a meaningful response with an explanation for a recommendation, CLICK achieved the best performance on all human evaluations. Especially, the better relevancy and informativeness demonstrated the efficacy of our response generator incorporating a dialogue context via the cross-attention.

6.2.2 Ablation Study

The ablation study wass conducted with the following three variants of CLICK, (1) **w/o gen. cross-att**: we removed the cross-attention mechanism incorporating a dialogue context. (2) **w/o gen. pre-training**: we only pre-trained the recommender module except for the response module, (3) **w/o pre-training**: we trained the CLICK on REDIAL dataset without pre-training. As shown in Table 2, we observed a significant decrease when the proposed strategies were removed. It proved that these strategies contributed to improving the performance of response generation. We noted that the cross-attention mechanism and the pre-training scheme of the response generator affected generating a relevant and informative response. Moreover, the *w/o pre-training* case brought a further decline in all metrics, indicating the usefulness of the pre-training strategy or generating informative responses using diverse words.

6.3 Case Study

A qualitative analysis of three cases is shown in Table 3. **Case 1**, there are no mentioned item-entities. Given the user utterance, KGSF recommended "*Black Panther*" that is suggested frequently in the CRS dataset regardless of a specific situation. Thus, it did not seem to capture the user's needs (*comforting movie*) precisely. Likewise, CR-Walker asked a question since it did not identify the needs exactly. In contrast, our model recommended a proper item with a rationale based on the captured needs. **Case 2** is a common case in the CRS dataset, which explicitly expresses the user's taste. Therefore, all baselines and CLICK generated a proper recommendation. Especially, CLICK provided a suitable explanation, understanding that the user wants comedy. **Case 3** includes multiple entities and a specific situation where the recommended movie should be enjoyable for little girls. We observed that all baselines suggested action movies, concentrating on the genre prevailing in the dialogue context. Here, the recommender should understand that the user is looking for a kids' movie, such as "*The Princess and The Frog*" besides action movies. In constrast, our model grasped the relatively important entity among mentioned entities based on the dialogue context. Consequently, our model's recommendation and rationale are well-grounded on users' needs.

[Case 1] **Recommender:** hello how are you today? **User:** I'm feeling pretty down tonight so I'd like to watch a comforting movie.
KGSF: Have you ever seen "Balck Panther"? **CR-Walker**: What type of movie do you like? **Ours**: What about "Mamma Mia!"? It's a comforting movie. **Human**: I recommend "The Secret Life of Walter Mitty", It's so satisfying.
[Case2] **Recommender:** hello what kind of movie do you like? **User:** I'm in the mood for a good comedy like "Get Smart"
KGSF: Have you ever seen "Jumanji"? **CR-Walker**: "Happy Gilmore" is a really good comedy. It's my favorite adam sandler movie. **Ours**: Have you ever seen "Men in Black"? It has funny story. **Human**: You should watch "Spy". I liked the humor of the actress.
[Case3] **Recommender:** Hello! Could you let me know any movies you liked? **User:** I like action movies like "John Wick " or "Die Hard 4.0", but today I'm looking for something similar to "The Princess and the Frog". I have little girls who want to watch a movie.
KGSF: Have you seen "Black Panther"? **CR-Walker**: I recommend "Taken". It's a really good movie with lots of action! **Ours**: What about "Tangled". It's a fairytale movie. Little girls will love it. **Human**: Then, I recommend "Frozen". Becasue littile girls like it.

Table 3: A qualitative analysis of three cases. The baselines, CLICK(ours), and human generate a response based on the given dialogue context. The important information for a recommendation is written in red, and the mentioned item-entities in a KG are written in blue.

7 Conclusions and Future Work

In this paper, we proposed the contextual knowledge injection via contrastive learning to capture a comprehensive user preference from a dialogue context. Utilizing the extracted contextual knowledge, CLICK not only precisely captures the preference on mentioned entities but also identifies the user's needs in a dialogue context. Moreover, the relevance-enhanced contrastive learning loss enables the contextual knowledge to reflect the acceptance degree of recommendations. We further developed the context-enhanced response generator to provide a persuasive explanation. Lastly, experimental results on the benchmark dataset confirmed the effectiveness of CLICK on providing appropriate recommendations, as well as generating high-quality responses.

In future work, we will explore a CRS with a recommendation dialogue strategy of actively asking questions to elicit a user taste instead of passively communicating with a user.

Limitations

In this section, we summarized the limitations of our study as follows:

- The training and test experiments are conducted on the setting where the mentioned entities are aligned to a knowledge graph in the same manner as other CRSs. In a practical use, Named Entity Recognition (NER) task is needed, and entity linking is also required to align the entities to a KG.
- This work only conducts experiments on the movie domain and does not generalize to other domains since the benchmark CRS dataset is limited to the movie domain. Benchmark CRS datasets on various domains need to be published, and experiments on multiple domains are required to support a complete and objective study.

Ethics Statement

All authors of this paper acknowledged ACL Ethics Policy. We utilized public datasets, REDIAL dataset and Reddit dataset. REDIAL dataset was constructed by crowdsourcing with Amazon Mechanical Turk, which is conducted anonymously. And Reddit dataset was crawled from Reddit community site without personal information. Although we have not observed an inappropriate content in the generated response, we inform that pretrained GPT-2 is leveraged, which may produce an inappropriate response.

Acknowledgements

We appreciate all of the reviewers giving their sincere comments. And we would like to express our gratitude to Donghyun Kim and Wongyu Kim for their valuable discussion and motivation. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2022R1A2B5B01001835). Kyong-Ho Lee is the corresponding author.

References

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. Dbpedia - a crystallization point for the web of data. *Web Semant.*, 7(3):154–165.

Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards knowledge-based recommender dialog system. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1803–1813.

Zhongxia Chen, Xiting Wang, Xing Xie, Mehul Parsana, Akshay Soni, Xiang Ao, and Enhong Chen. 2021. Towards explainable conversational recommendation. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 2994–3000.

Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. 2021. *Unified Conversational Recommendation Policy Learning via Graph-Based Reinforcement Learning*, page 1431–1441.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and challenges in conversational recommender systems: A survey. *AI Open*, 2:100–126.

Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*.

Po-Yao Huang, Mandela Patrick, Junjie Hu, Graham Neubig, Florian Metze, and Alexander G Hauptmann. 2021. Multilingual multimodal pretraining for zero-shot cross-lingual transfer of vision-language models. In *NAACL*, pages 2443–2459.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2073–2083.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, pages 110–119.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. *Advances in neural information processing systems*, 31.

Yu Lu, Junwei Bao, Yan Song, Zichen Ma, Shuguang Cui, Youzheng Wu, and Xiaodong He. 2021. Revcore: Review-augmented conversational recommendation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1161–1173.

Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. 2021. Cr-walker: Tree-structured graph reasoning and dialog acts for conversational recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1839–1851.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Gustavo Penha and Claudia Hauff. 2020. What does bert know about books, movies and music? probing bert for conversational recommendation. In *Fourteenth ACM Conference on Recommender Systems*, pages 388–397.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks.

Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2021. Self-supervised hypergraph convolutional networks for session-based recommendation. 35(5):4503–4511.

Xin Yuan, Zhe Lin, Jason Kuen, Jianming Zhang, Yilin Wang, Michael Maire, Ajinkya Kale, and Baldo Faieta. 2021. Multimodal contrastive training for visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6995–7004.

Jinfeng Zhou, Bo Wang, Ruifang He, and Yuexian Hou. 2021. Crfr: Improving conversational recommender systems via flexible fragments reasoning on knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4324–4334.

Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020a. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1006–1014.

Kun Zhou, Wayne Xin Zhao, Hui Wang, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020b. Leveraging historical interaction data for improving conversational recommender system. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2349–2352.

Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. 2020c. Towards topic-guided conversational recommender system. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4128–4139.

Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. 2020d. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 179–188.

Yuanhang Zhou, Kun Zhou, Wayne Xin Zhao, Cheng Wang, Peng Jiang, and He Hu. 2022. C^2-crs: Coarse-to-fine contrastive learning for conversational recommender system. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1488–1496.

Mohammadreza Zolfaghari, Yi Zhu, Peter Gehler, and Thomas Brox. 2021. Crossclr: Cross-modal contrastive learning for multi-modal video representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1450–1459.

A Appendix

A.1 The Pipeline between BERT and FCN in Text-Encdoer

The output of BERT is an embedding vector of a CLS token containing the meaning of the entire sentence. And the embedding vector is input into FCN.

A.2 Implementation Details

We implemented CLICK with Pytorch[1] and trained it on an NVIDIA RTX 3090. We set the embedding size of the KG-encoder (R-GCN)[2] to 128, and the depth of aggregating neighbors as 1. We used pre-trained BERT[3] and GPT-2[4] to intialize the models' parameters. The hidden size of BERT in the text-encoder is set to 768, and the hidden size of the response generation module (GPT-2) is identically set to 768. When obtaining the embedding of a context-level user preference from the text-encoder, we utilized a 128×768 fully-connected layer. We normalized the relevance scores from 0.25 to 0.5 for a stable convergence. We used Adam optimizer (Kingma and Ba, 2015) with a learning rate of 1e-4 and a weight decay of 1e-3. The batch size of pre-training and fine-tuning is set to 32. The computational cost is 3 hours and 1 hour per 1 epoch for pre-training and fine-tuning, respectively.

A.3 PLMs Details

The PLMs with (+*Reddit*) are pre-trained on Reddit dataset and fine-tuned on REDIAL dataset. And the representation of the last token that a PLM generates is used for recommendation.

A.4 No Mentioned Item Setting

We utilized a special entity (*None entity*) in the same manner as other CRSs (KGSF, CR-Walker, and C2CRS) that utilize KGs. Thus, CLICK and other CRSs can be executed under any case. For example, at the first turn of the conversation in Figure 1, there is no mentioned entity. Thus, *None entity* is assigned as mentioned entities $N^{(C)}$ in Equation 6. Meanwhile, the text-encoder of CLICK is pre-trained with Reddit data to infer recommendable items (*e.g., 21JumpStreet, MadMax:FuryRoad, etc.*) from the request context where a user gets stressed or requires mindless movies. The gate mechanism in Equation 7 then emphasizes a context-level user preference p_{cl} obtained from the pre-trained text-encoder than an entity-level user preference p_{el} in the case of no mentioned entities. Thus, the user embedding concentrated on the context-level user preference p_{cl} enables CLICK to recommend *21JumpStreet* from the user's first utterance in Figure 1.

---

[1] https://pytorch.org/

[2] https://github.com/pyg-team/pytorch_geometric

[3] https://huggingface.co/bert-base-uncased

[4] https://huggingface.co/gpt2

LEALLA: Learning Lightweight Language-agnostic Sentence Embeddings with Knowledge Distillation

Zhuoyuan Mao[*] and **Tetsuji Nakagawa**
Google Research
`kevinmzy@gmail.com, tnaka@google.com`

Abstract

Large-scale language-agnostic sentence embedding models such as LaBSE (Feng et al., 2022) obtain state-of-the-art performance for parallel sentence alignment. However, these large-scale models can suffer from inference speed and computation overhead. This study systematically explores learning language-agnostic sentence embeddings with lightweight models. We demonstrate that a thin-deep encoder can construct robust low-dimensional sentence embeddings for 109 languages. With our proposed distillation methods, we achieve further improvements by incorporating knowledge from a teacher model. Empirical results on Tatoeba, United Nations, and BUCC show the effectiveness of our lightweight models. We release our lightweight language-agnostic sentence embedding models LEALLA on TensorFlow Hub.[1]

1 Introduction

Language-agnostic sentence embedding models (Artetxe and Schwenk, 2019b; Yang et al., 2020; Reimers and Gurevych, 2020; Feng et al., 2022; Mao et al., 2022) align multiple languages in a shared embedding space, facilitating parallel sentence alignment that extracts parallel sentences for training translation systems (Schwenk et al., 2021). Among them, LaBSE (Feng et al., 2022) achieves the state-of-the-art parallel sentence alignment accuracy over 109 languages. However, 471M parameters of LaBSE lead to the computationally-heavy inference. The 768-dimensional sentence embeddings of LaBSE (LaBSE embeddings) make it suffer from computation overhead of downstream tasks (e.g., kNN search). This limits its application on resource-constrained devices. Therefore, we explore training a lightweight model to generate low-dimensional sentence embeddings while retaining the performance of LaBSE.

We first investigate the performance of dimension-reduced LaBSE embeddings and show that it performs comparably with LaBSE. Subsequently, we experiment with various architectures to see whether such effective low-dimensional embeddings can be obtained from a lightweight encoder. We observe that the thin-deep (Romero et al., 2015) architecture is empirically superior for learning language-agnostic sentence embeddings. Diverging from previous work, we show that low-dimensional embeddings based on a lightweight model are effective for parallel sentence alignment of 109 languages.

LaBSE benefits from multilingual language model pre-training, but no multilingual pre-trained models are available for the lightweight architectures. Thus, we propose two knowledge distillation methods to further enhance the lightweight models by forcing the model to extract helpful information from LaBSE. We present three lightweight models improved with distillation: **LEALLA-small**, **LEALLA-base**, and **LEALLA-large**, with 69M, 107M, and 147M parameters, respectively. Fewer model parameters and their 128-d, 192-d, and 256-d sentence embeddings are expected to accelerate downstream tasks, while the performance drop of merely up to 3.0, 1.3, and 0.3 P@1 (or F1) points is observed on three benchmarks of parallel sentence alignment. In addition, we show the effectiveness of each loss function through an ablation study.

2 Background: LaBSE

LaBSE (Feng et al., 2022) fine-tunes dual encoder models (Guo et al., 2018; Yang et al., 2019) to learn language-agnostic embeddings from a large-scale pre-trained language model (Conneau et al., 2020). LaBSE is trained with parallel sentences, and each sentence pair is encoded separately by

---

[*] Currently at Kurohashi-Chu-Murawaki Lab., Kyoto University. Work done during Google internship.

[1] `https://tfhub.dev/s?q=LEALLA`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1886–1894
May 2-6, 2023

a 12-layer Transformer encoder. The 768-d encoder outputs are used to compute the training loss and serve as sentence embeddings for downstream tasks. Expressly, assume that the sentence embeddings for parallel sentences in a batch are $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$ where N denotes the number of the sentence pairs within a batch. LaBSE trains the bidirectional additive margin softmax (AMS) loss:

$$\mathcal{L}_{ams} = \frac{1}{N}\sum_{i=1}^{N}(\mathcal{L}(\mathbf{x}_i, \mathbf{y}_i) + \mathcal{L}(\mathbf{y}_i, \mathbf{x}_i)), \quad (1)$$

where the loss for a specific sentence pair in a single direction is defined as:

$$\mathcal{L}(\mathbf{x}_i, \mathbf{y}_i) = -\log \frac{e^{\phi(\mathbf{x}_i, \mathbf{y}_i)-m}}{e^{\phi(\mathbf{x}_i, \mathbf{y}_i)-m} + \sum_{n \neq i} e^{\phi(\mathbf{x}_i, \mathbf{y}_n)}}. \quad (2)$$

m is a margin for optimizing the separation between translations and non-translations. $\phi(\mathbf{x}_i, \mathbf{y}_i)$ is defined as Cosine Similarity between $\mathbf{x}_i$ and $\mathbf{y}_i$.

3 Light Language-agnostic Embeddings

To address the efficiency issue of LaBSE, we probe the lightweight model for learning language-agnostic embeddings with the following experiments: (1) We directly reduce the dimension of LaBSE embeddings to explore the optimal embedding dimension; (2) We shrink the model size with various ways to explore the optimal architecture.

3.1 Evaluation Settings

We employ Tatoeba (Artetxe and Schwenk, 2019b), United Nations (UN) (Ziemski et al., 2016), and BUCC (Pierre Zweigenbaum and Rapp, 2018) benchmarks for evaluation, which assess the model performance for parallel sentence alignment. Following Feng et al. (2022) and Artetxe and Schwenk (2019b), we report the average P@1 of bidirectional retrievals for all the languages of Tatoeba, the average P@1 for four languages of UN, and the average F1 of bidirectional retrievals for four languages of BUCC.[2] Refer to Appx. A for details.

3.2 Exploring the Optimal Dimension of Language-agnostic Sentence Embeddings

Mao et al. (2021) showed that a 256-d bilingual embedding space could achieve an accuracy of about 90% for parallel sentence alignment. However, existing multilingual sentence embedding models such as LASER (2019b), SBERT (2020), EMS (2022), and LaBSE use 768-d or 1024-d sentence embeddings, and whether a low-dimensional space can align parallel sentences over tens of languages with a solid accuracy (>80%) remains unknown. Thus, we start with the dimension reduction experiments for LaBSE to explore the optimal dimension of language-agnostic sentence embeddings.

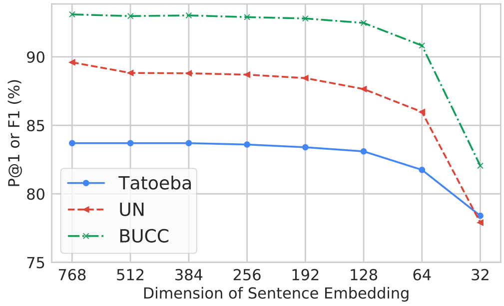


Figure 1: Dimension reduction for LaBSE.

We add an extra dense layer on top of LaBSE to transform the dimension of LaBSE embeddings from 768 to lower values. We experiment with seven lower dimensions ranging from 512 to 32. We fine-tune 5k steps for fitting the newly added dense layer, whereas other parameters of LaBSE are fixed. Refer to Appx. B for training details.

As shown in Fig. 1, the performance drops more than 5 points when the dimension is 32 on Tatoeba, UN, and BUCC. Meanwhile, given sentence embeddings with a dimension over 128, they performs slightly worse than 768-d LaBSE embeddings with a performance drop of fewer than 2 points, showing that low-dimensional sentence embeddings can align parallel sentences in multiple languages. Refer to Appx. D for detailed results.

3.3 Exploring the Optimal Architecture

Although we revealed the effectiveness of the low-dimensional embeddings above, it is generated from LaBSE with 471M parameters. Thus, we explore whether such low-dimensional sentence embeddings can be obtained from an encoder with less parameters. We first reduce the number of layers (#1 and #2 in Table 1) and the size of hidden states (#3 and #4) to observe the performance. Subsequently, inspired by the effectiveness of FitNet (Romero et al., 2015) and MobileBERT (Sun et al., 2020) and taking advantage of the low-dimensional sentence embeddings shown above, we experiment with thin-deep architectures with 24

---

[2] For BUCC, we use margin-based scoring (Artetxe and Schwenk, 2019a) for filtering translation pairs.

#	L	$\mathbf{d_h}$	H	P	$\mathbf{P_E}$	Tatoeba	UN	BUCC
LaBSE								
0	12	768	12	471M	85M	83.7	89.6	93.1
Fewer Layers								
1	6	768	12	428M	42M	82.9	88.6	91.9
2	3	768	12	407M	21M	82.2	87.5	91.2
Smaller Hidden Size								
3	12	384	12	214M	21M	82.6	88.4	92.1
4	12	192	12	102M	6M	81.0	87.0	91.3
Thin-deep Architecture								
5	24	384	12	235M	42M	83.2	88.6	92.4
6	24	256	8	147M	19M	82.9	88.5	92.2
7	24	192	12	107M	11M	81.7	87.4	91.9
8	24	128	8	69M	5M	80.3	86.3	90.4

Table 1: Results of LaBSE variants. $\mathbf{L}$, $\mathbf{d_h}$, $\mathbf{H}$, $\mathbf{P}$, and $\mathbf{P_E}$ denote the number of layers, dimension of hidden states, number of attention heads, number of parameters, and number of encoder parameters (except for the word embedding layer). Refer to Appx. E for detailed results.

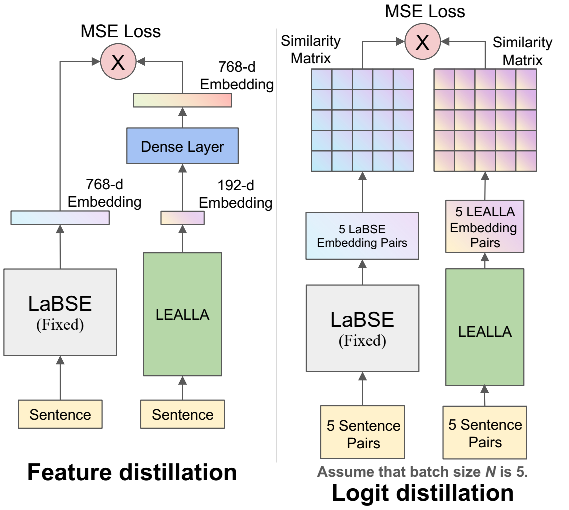


Figure 2: Feature and logit distillation from LaBSE.

layers (#5 - #8), leading to fewer encoder parameters.[3] Refer to Appx. B for training details.

We report the results in Table 1. First, architectures with fewer layers (#1 and #2) perform worse than LaBSE on all three tasks and can only decrease parameters by less than 15%. Second, increasing the number of layers (#5 and #7) improves the performance of 12-layer models (#3 and #4) with a limited parameter increase less than 10%. Referring to LaBSE (#0), low-dimensional embeddings from thin-deep architectures (#5 - #8) obtain solid results on three benchmarks with performance drops of only 3.4 points at most. Until this point, we showed that thin-deep architecture is effective for learning language-agnostic sentence embeddings.

4 Knowledge Distillation from LaBSE

Besides the large model capacity, multilingual language model pre-training benefits LaBSE for parallel sentence alignment. As no multilingual pre-trained language models are available for lightweight models we investigated in Section 3.3, we instead explore extracting helpful knowledge from LaBSE.

4.1 Methodology

Feature distillation and logit distillation have been proven to be effective paradigms for knowledge distillation (Hinton et al., 2015; Romero et al., 2015; Yim et al., 2017; Tang et al., 2019). In this section, we propose methods applying both paradigms to language-agnostic sentence embedding distillation. We use LaBSE as a teacher to train students with thin-deep architectures which were discussed in Section 3.3.

Feature Distillation We propose applying feature distillation to language-agnostic sentence embedding distillation, which enables lightweight sentence embeddings to approximate the LaBSE embeddings via an extra dense layer. We employ an extra trainable dense layer on top of the lightweight models to unify the embedding dimension of LaBSE and lightweight models to be 768-d, as illustrated in Fig. 2.[4][5] The loss function is defined as follows:

$$\mathcal{L}_{fd} = \frac{1}{N} \sum_{i=1}^{N} (\| \mathbf{x}_i^t - f(\mathbf{x}_i^s) \|_2^2 + \| \mathbf{y}_i^t - f(\mathbf{y}_i^s) \|_2^2), \tag{3}$$

where $\mathbf{x}^t$ (or $\mathbf{y}^t$) and $\mathbf{x}^s$ (or $\mathbf{y}^s$) are the embeddings by LaBSE and the lightweight model, respectively. $f(\cdot)$ is a trainable dense layer transforming the dimension from d ($d < 768$) to 768.

Logit Distillation We also propose applying logit distillation to language-agnostic sentence embedding distillation to extract knowledge from the sentence similarity matrix as shown in Fig. 2. Logit distillation forces the student to establish similar similarity relationships between the given sentence pairs as the teacher does. We propose the following

---

[3] Following MobileBERT, we attempted architectures that have an identical size for hidden state and feed-forward hidden state, but it works poorly than #5 - #8. (Refer to Appx. E)

[4] SBERT (2020) used feature distillation to make monolingual sentence embeddings multilingual, but distillation between different embedding dimensions has not been studied.

[5] We investigated another two patterns to unify the embedding dimensions in Appx. C, but they performed worse.

Model	**La.**	**d**	**P**	**Ttb.**	**UN**					**BUCC**				
					es	**fr**	**ru**	**zh**	**avg.**	**de**	**fr**	**ru**	**zh**	**avg.**
LASER (2019b)	93	1024	154M	65.5	-	-	-	-	-	**95.4**	**92.4**	**92.3**	91.7	**93.0**
m-USE (2020)	16	512	**85M**	-	86.1	83.3	88.9	78.8	84.3	88.5	86.3	89.1	86.9	87.7
SBERT (2020)	50	768	270M	67.1	-	-	-	-	-	90.8	87.1	88.6	87.8	88.6
EMS (2022)	62	1024	148M	69.2	-	-	-	-	-	93.3	90.2	91.3	**92.1**	91.7
LaBSE (2022)	**109**	768	471M	**83.7**	**90.8**	**89.0**	**90.4**	**88.3**	**89.6**	**95.5**	**92.3**	**92.2**	**92.5**	**93.1**
LEALLA-small	**109**	**128**	**69M**	80.7	89.4	86.0	88.7	84.9	87.3	94.0	90.6	91.2	90.3	91.5
LEALLA-base	**109**	**192**	**107M**	**82.4**	**90.3**	**87.4**	**89.8**	**87.2**	**88.7**	94.9	91.4	91.8	91.4	92.4
LEALLA-large	**109**	**256**	147M	**83.5**	**90.8**	**88.5**	**89.9**	**87.9**	**89.3**	**95.3**	**92.0**	**92.1**	**91.9**	**92.8**

Table 2: Results of LEALLA. We mark the best 3 scores in **bold**. **La.**, **d**, **P**, and **Ttb.** indicate the number of languages, dimension of sentence embeddings, number of parameters, and Tatoeba.

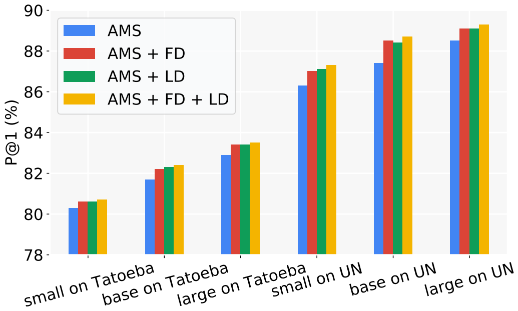


Figure 3: LEALLA with different loss combinations. AMS, FD, and LD mean $\mathcal{L}_{ams}$, $\mathcal{L}_{fd}$, and $\mathcal{L}_{ld}$.

Loss	**LEALLA-small**		**LEALLA-base**		**LEALLA-large**	
	Tatoeba	**UN**	**Tatoeba**	**UN**	**Tatoeba**	**UN**
all	*80.7*	*87.3*	*82.4*	*88.7*	*83.5*	*89.3*
$\mathcal{L}_{ams}$	**80.3**	**86.3**	**81.7**	87.4	**82.9**	**88.5**
$\mathcal{L}_{fd}$	78.2	85.2	81.1	**88.1**	82.4	88.1
$\mathcal{L}_{ld}$	75.1	2.3	80.6	63.1	82.3	84.1

Table 3: Results of LEALLA with each loss function. "*all*" denotes LEALLA without ablation (with all the loss functions).

mean squared error (MSE) loss:

$$\mathcal{L}_{ld} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\left(\phi \left(\mathbf{x}_i^t, \mathbf{y}_j^t \right) - \phi \left(\mathbf{x}_i^s, \mathbf{y}_j^s \right) \right) / T \right)^2, \tag{4}$$

where T is a distillation temperature, and other notations follow those in Eq. 2 and 3.

Combined Loss Finally, we combine two knowledge distillation loss functions with the AMS loss (Eq. 1) to jointly train the lightweight model:

$$\mathcal{L}_{lealla} = \alpha \mathcal{L}_{ams} + \beta \mathcal{L}_{fd} + \gamma \mathcal{L}_{ld}. \tag{5}$$

Here α, β, and γ are weight hyperparameters, which are tuned with the development data.

4.2 Experiments

Training We train three models, **LEALLA-small**, **LEALLA-base**, and **LEALLA-large**, using the thin-deep architectures of #8, #7, and #6 in Table 1 and the training loss of Eq. 5. Refer to Appx. B for training and hyperparameter details.

Results The results of LEALLA on Tatoeba, UN, and BUCC benchmarks are presented in Table 2. Overall, LEALLA can yield competitive performance compared with previous work. LEALLA-large performs comparably with LaBSE, where the average performance difference on three tasks is below 0.3 points. LEALLA-base and LEALLA-small obtain strong performance for high-resource languages on UN and BUCC, with a performance decrease less than 0.9 and 2.3 points, respectively. They also achieve solid results on Tatoeba with 1.3 and 3 points downgrades compared with LaBSE. The solid performance of LEALLA on Tatoeba demonstrates that it is effective for aligning parallel sentences for more than 109 languages. Moreover, all the LEALLA models perform better or comparably with previous studies other than LaBSE.

Ablation Study We inspect the effectiveness of each loss component in an ablative manner. First, we compare settings with and without distillation loss functions. As shown in Fig. 3, by adding $\mathcal{L}_{fd}$ or $\mathcal{L}_{ld}$, LEALLA trained only with $\mathcal{L}_{ams}$ is improved on Tatoeba and UN tasks. By further combining $\mathcal{L}_{fd}$ and $\mathcal{L}_{ld}$, LEALLA consistently achieves superior performance. Second, we separately train LEALLA with each loss. Referring to the results reported in Table 3, LEALLA trained only with $\mathcal{L}_{fd}$ yields solid performance in the "small" and "base" models compared with $\mathcal{L}_{ams}$, showing that distillation loss benefits parallel sentence alignment. $\mathcal{L}_{fd}$ and $\mathcal{L}_{ld}$ perform much worse in the "small" model, which may be attributed to the discrepancy in the capacity gaps between the teacher model (LaBSE) and the student model ("small" or "base").[6] Refer to Appx. F for all detailed results in this section.

5 Conclusion

We presented LEALLA, a lightweight model for generating low-dimensional language-agnostic sentence embeddings. Experimental results showed that LEALLA could yield solid performance for 109 languages after distilling knowledge from LaBSE. Future work can focus on reducing the vocabulary size of LaBSE to shrink the model further and exploring the effectiveness of lightweight model pre-training for parallel sentence alignment.

Limitations

In this study, we used the same training data as LaBSE (refer to Fig. 7 of (Feng et al., 2022)), where more training data for high-resource languages may cause the biased model accuracy for those languages. Second, evaluation for low-resource languages in this study depended only on the Tatoeba benchmark, which contains only 1,000 positive sentence pairs for each language with English. The same limitation exists in all the related work, such as LaBSE and LASER. Further evaluation for low-resource languages will be necessary in the future once larger evaluation benchmarks, including over 100k gold parallel sentences for low-resource languages, are available. Third, all the training data used in this work are English-centric sentence pairs, which may result in the inferior model performance for aligning parallel sentences between non-English language pairs.

Acknowledgements

We would like to thank our colleagues from Translate, Descartes, and other Google teams for their valuable contributions and feedback. A special mention to Fangxiaoyu Feng, Shuying Zhang, Gustavo Hernandez Abrego, and Jianmon Ni for their support in sharing information on LaBSE, and providing training data, expertise on language-agnostic sentence embeddings, and assistance with evaluation. We would also like to thank the reviewers for their insightful comments for improving the paper.

References

Mikel Artetxe and Holger Schwenk. 2019a. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.

Mikel Artetxe and Holger Schwenk. 2019b. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

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

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Mandy Guo, Qinlan Shen, Yinfei Yang, Heming Ge, Daniel Cer, Gustavo Hernandez Abrego, Keith Stevens, Noah Constant, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Effective parallel corpus mining using bilingual sentence embeddings. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 165–176, Brussels, Belgium. Association for Computational Linguistics.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhuoyuan Mao, Chenhui Chu, and Sadao Kurohashi. 2022. EMS: efficient and effective massively multilingual sentence representation learning. *CoRR*, abs/2205.15744.

Zhuoyuan Mao, Prakhar Gupta, Chenhui Chu, Martin Jaggi, and Sadao Kurohashi. 2021. Lightweight cross-lingual sentence representation learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages

---

[6] $\mathcal{L}_{ld}$ can hardly work for UN and BUCC as they contain hundreds of thousands of candidates for the model to score, which is more complicated than the 1,000 candidates of Tatoeba.

2902–2913, Online. Association for Computational Linguistics.

Serge Sharoff Pierre Zweigenbaum and Reinhard Rapp. 2018. Overview of the third bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France. European Language Resources Association (ELRA).

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from BERT into simple neural networks. *CoRR*, abs/1903.12136.

Yinfei Yang, Gustavo Hernández Ábrego, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5370–5378. ijcai.org.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

Junho Yim, Donggyu Joo, Ji-Hoon Bae, and Junmo Kim. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 7130–7138. IEEE Computer Society.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia. European Language Resources Association (ELRA).

A Evaluation Benchmarks

Tatoeba (Artetxe and Schwenk, 2019b) supports the evaluation across 112 languages and contains up to 1,000 sentence pairs for each language and English. The languages of Tatoeba that are not included in the training data of LaBSE and LEALLA serve as the evaluation for unseen languages. UN (Ziemski et al., 2016) is composed of 86,000 aligned bilingual documents for en-ar, en-es, en-fr, en-ru, and en-zh. Following Feng et al. (2022), we evaluate the model performance for es, fr, ru, and zh on the UN task. There are about 9.5M sentence pairs for each language with English after deduping. BUCC shared task (Pierre Zweigenbaum and Rapp, 2018) is a benchmark to mine parallel sentences from comparable corpora. We conduct the evaluation using BUCC2018 tasks for en-de, en-fr, en-ru, and en-zh, following the setting of Reimers and Gurevych (2020).[7] For the results of LaBSE reported in Table 2, we re-conduct the evaluation experiments using the open-sourced model of LaBSE.[8]

B Training Details

All of the models in this work are trained with the same training data and development data as LaBSE (Feng et al., 2022). Refer to Section 3.1 and

---

[7] `https://github.com/UKPLab/sentence-transformers/blob/master/examples/applications/parallel-sentence-mining/bucc2018.py`

[8] `https://tfhub.dev/google/LaBSE`

Model	Tatoeba	UN					BUCC				
		es	fr	ru	zh	avg.	de	fr	ru	zh	avg.
LEALLA-small											
$\mathcal{L}_{ams}$	80.3	88.1	85.2	88.0	83.9	86.3	93.0	89.7	90.6	88.3	90.4
$\mathcal{L}_{ams} + \mathcal{L}_{fd}$	**80.6**	89.3	**86.8**	88.0	**84.0**	**87.0**	**93.9**	**90.6**	**91.4**	**89.7**	**91.4**
$\mathcal{L}_{ams} + \mathcal{L}_{df}$	80.0	**89.4**	86.3	**88.1**	83.9	86.9	93.8	90.1	91.1	88.9	91.0
$\mathcal{L}_{ams} + \mathcal{L}_{syn}$	80.2	88.5	85.0	87.1	82.8	85.9	93.6	89.9	90.9	88.7	90.8
LEALLA-base											
$\mathcal{L}_{ams}$	81.7	89.8	85.9	88.6	85.4	87.4	94.2	91.0	91.3	**91.1**	91.9
$\mathcal{L}_{ams} + \mathcal{L}_{fd}$	**82.2**	**90.2**	**87.5**	**89.4**	**86.8**	**88.5**	**95.0**	**91.6**	**91.7**	91.0	**92.3**
$\mathcal{L}_{ams} + \mathcal{L}_{df}$	81.8	90.0	87.3	89.2	86.3	88.2	94.7	91.4	**91.7**	90.9	92.2
$\mathcal{L}_{ams} + \mathcal{L}_{syn}$	81.9	89.7	86.7	88.8	85.9	87.8	94.5	91.1	**91.7**	90.3	91.9
LEALLA-large											
$\mathcal{L}_{ams}$	82.9	90.1	87.1	89.3	87.4	88.5	94.6	91.2	91.5	91.4	92.2
$\mathcal{L}_{ams} + \mathcal{L}_{fd}$	**83.4**	**90.6**	**88.4**	**89.8**	**87.7**	**89.1**	**95.3**	**92.0**	**92.0**	**92.0**	**92.8**
$\mathcal{L}_{ams} + \mathcal{L}_{df}$	83.0	90.3	87.6	89.7	87.2	88.7	**95.3**	91.9	**92.0**	91.7	92.7
$\mathcal{L}_{ams} + \mathcal{L}_{syn}$	83.0	90.0	87.4	89.7	86.8	88.5	94.9	91.7	91.8	91.4	92.5

Table 4: Results of comparisons among three feature distillation objectives. $\mathcal{L}_{df}$ and $\mathcal{L}_{syn}$ indicate "*Distillation-first*" and "*Synchronized*" objectives in Fig. 4.

Hyperparameter	Bound
α	1
β	1e02, 1e03, 1e04, 1e05
γ	1e-01, 1e-02, 1e-03
batch size	2,048, 4,096, 8,192
learning rate	1e-4, 5e-4, 1e-3

Table 5: Hyperparameter bounds.

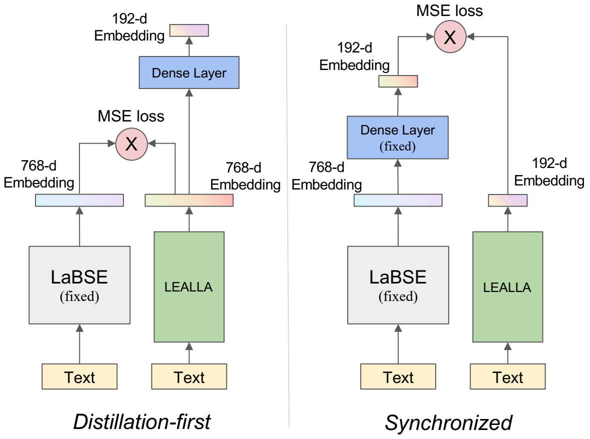


Figure 4: Another two patterns of feature distillation.

Appx. C of Feng et al. (2022) for dataset and supported language details. We train models on Cloud TPU V3 with 32-cores with a global batch size of 8,192 sentences and a maximum sequence length of 128. For a fair comparison with LaBSE for more than 109 languages, we use the 501k vocabulary of LaBSE (trained with BPE (Sennrich et al., 2016)) and do not consider modifying its size in this work. We employ AdamW (Loshchilov and Hutter, 2019) for optimizing the model using the initial learning rate of 1e-03 for models with a hidden state size larger than 384 and 5e-04 for models with a hidden state size smaller than 256. For LEALLA-small and LEALLA-base, α, β, and γ are set as 1, 1e03 and 1e-02. For LEALLA-large, they are set as 1, 1e04, and 1e-02, respectively. T in Eq. 4 is set to 100. All the models in Section 3.2 are trained for 5k steps. Models in Secton 3.3 and Section 4 with a hidden state size over 256 are trained for 200k steps, and those with a hidden state size below 192 are trained for 100k steps. It costs around 24 hours, 36 hours, and 48 hours to train LEALLA-small, LEALLA-base, and LEALLA-large, respectively. Hyperparameters are tuned using a held-out development dataset following Feng et al. (2022) with a grid search. The bounds tuned for each hyperparameter are shown in Table 5.

C Discussion about Feature Distillation

We additionally investigate another two patterns for feature distillation. As illustrated in Fig. 4, "*Distillation-first*" modifies the position for computing the MSE loss compared with $\mathcal{L}_{fd}$ of Eq. 3. The `[CLS]` pooler within the LEALLA encoder is used to generate 768-d embeddings first. A dense layer is employed to transform the 768-d embeddings to low-dimension after calculating the MSE loss. "*Synchronized*" transforms the LaBSE embeddings to low-dimension, then the MSE loss is constructed between two low-dimensional embeddings. As the MSE loss is computed simultaneously with the

Dimension	Tatoeba	UN					BUCC				
		es	fr	ru	zh	avg.	de	fr	ru	zh	avg.
768 (LaBSE)	83.7	90.8	89.0	90.4	88.3	89.6	95.5	92.3	92.2	92.5	93.1
512	83.7	90.1	88.1	89.7	87.4	88.8	95.4	92.1	92.0	92.4	93.0
384	83.7	90.1	88.1	89.6	87.4	88.8	95.5	92.0	92.0	92.6	93.0
256	83.6	90.3	87.9	89.2	87.4	88.7	95.3	92.0	92.1	92.2	92.9
192	83.4	89.8	87.5	89.5	87.0	88.5	95.2	91.9	91.9	92.2	92.8
128	83.1	89.2	86.9	88.6	85.9	87.7	95.1	91.4	91.8	91.6	92.5
64	81.8	88.4	84.4	87.3	83.8	86.0	93.9	89.8	90.7	88.9	90.8
32	78.4	82.7	74.8	80.4	73.7	77.9	87.1	81.5	84.1	75.5	82.1

Table 6: Results of the dimension-reduced LaBSE embeddings.

#	$\mathbf{L}$	$\mathbf{d_h}$	$\mathbf{d_{ff}}$	$\mathbf{H}$	$\mathbf{P}$	$\mathbf{P_E}$	Tatoeba	UN					BUCC				
								es	fr	ru	zh	avg.	de	fr	ru	zh	avg.
LaBSE																	
0	12	768	3072	12	471M	85M	83.7	90.8	89.0	90.4	88.3	89.6	95.5	92.3	92.2	92.5	93.1
Fewer Layers																	
1	6	768	3072	12	428M	42M	82.9	90.2	87.4	89.2	87.4	88.6	94.3	90.9	91.2	91.1	91.9
2	3	768	3072	12	407M	21M	82.2	89.4	86.1	88.0	86.5	87.5	93.7	90.1	90.8	90.1	91.2
Smaller Hidden Size																	
3	12	384	1536	12	214M	21M	82.6	90.1	86.9	89.6	87.0	88.4	94.4	91.2	91.4	91.3	92.1
4	12	192	768	12	102M	6M	81.0	89.4	85.6	88.1	85.0	87.0	93.6	90.4	91.1	89.9	91.3
Thin-deep Architecture																	
5	24	384	1536	12	235M	42M	83.2	90.6	87.3	89.2	87.4	88.6	94.7	91.5	91.6	91.9	92.4
6	24	256	1024	8	147M	19M	82.9	90.1	87.1	89.3	87.4	88.5	94.6	91.2	91.5	91.4	92.2
7	24	192	768	12	107M	11M	81.7	89.8	85.9	88.6	85.4	87.4	94.2	91.0	91.3	91.1	91.9
8	24	128	512	8	69M	5M	80.3	88.1	85.2	88.0	83.9	86.3	93.0	89.7	90.6	88.3	90.4
9	24	64	256	8	33M	1M	75.2	83.7	78.6	83.0	72.1	79.4	87.9	83.0	86.0	75.1	83.0
MobileBERT-like Thin-deep Architecture																	
10	24	256	256	4	138M	10M	82.1	89.4	86.5	88.4	86.5	87.7	94.1	91.0	91.0	91.7	92.0
11	24	192	192	4	102M	6M	81.0	89.0	85.4	88.5	85.3	87.1	93.8	90.3	91.0	89.9	91.3
12	24	128	128	4	66M	2M	79.7	88.1	84.1	87.6	83.3	85.8	92.6	88.8	90.4	87.6	89.9

Table 7: Results of thin-deep and MobileBERT-like architectures. $\mathbf{L}$, $\mathbf{d_h}$, $\mathbf{d_{ff}}$, $\mathbf{H}$, $\mathbf{P}$, and $\mathbf{P_E}$ indicate the number of layers, dimension of hidden states, dimension of feed-forward hidden states, number of attention heads, number of model parameters, and number of encoder parameters (except for the word embedding layer).

AMS loss, it is denoted as "*Synchronized*". For "*Synchronized*", it requires a fixed dense layer to conduct the dimension reduction for the LaBSE embeddings, for which we utilize the pre-trained model introduced in Section 3.2. We denote these two patterns of feature distillation as $\mathcal{L}_{df}$ and $\mathcal{L}_{syn}$.

As reported in Table 4, $\mathcal{L}_{ams} + \mathcal{L}_{fd}$ ($\mathcal{L}_{fd}$ is feature distillation introduced in the main text) consistently outperforms $\mathcal{L}_{ams} + \mathcal{L}_{df}$ and $\mathcal{L}_{ams} + \mathcal{L}_{syn}$ in all the three LEALLA models. $\mathcal{L}_{ams} + \mathcal{L}_{df}$ and $\mathcal{L}_{ams} + \mathcal{L}_{syn}$ perform comparably on Tatoeba with the models trained without distillation loss. $\mathcal{L}_{ams} + \mathcal{L}_{df}$ obtains performance gains for high-resource languages on UN and BUCC compared with $\mathcal{L}_{ams}$, but still underperforms $\mathcal{L}_{ams} + \mathcal{L}_{fd}$.

$\mathcal{L}_{df}$ forces the lightweight model to approximate the teacher embeddings first in the intermediate part of the model, on top of which the low-dimensional sentence embeddings are generated for computing the AMS loss, while $\mathcal{L}_{fd}$ (Eq. 3) is calculated after computing the AMS loss. As the AMS loss directly indicates the evaluation tasks, we suppose $\mathcal{L}_{fd}$ is a more flexible objective for feature distillation. In addition, $\mathcal{L}_{syn}$ is not beneficial because it depends on a dimension-reduced LaBSE, which is a less robust teacher compared with LaBSE.

D Results of Dimension-reduction Experiments

We report all the results of Section 3.2 in Table 6.

E Results of Thin-deep and MobileBERT-like Architectures

Table 7 presents the detailed results of each architecture we explored in Section 3.3. Besides showing the results for each language on UN and

Model	Tatoeba	UN					BUCC				
		es	**fr**	**ru**	**zh**	**avg.**	**de**	**fr**	**ru**	**zh**	**avg.**
LEALLA-small											
$\mathcal{L}_{ams}$	80.3	88.1	85.2	88.0	83.9	86.3	93.0	89.7	90.6	88.3	90.4
$\mathcal{L}_{fd}$	78.2	89.0	84.6	87.5	79.6	85.2	**94.2**	90.5	91.2	88.9	91.2
$\mathcal{L}_{ld}$	75.1	1.5	1.1	0.9	5.6	2.3	0.1	0.0	0.1	0.0	0.1
$\mathcal{L}_{ams} + \mathcal{L}_{fd}$	80.6	89.3	**86.8**	88.0	84.0	87.0	93.9	**90.6**	**91.4**	89.7	91.4
$\mathcal{L}_{ams} + \mathcal{L}_{ld}$	80.6	**89.6**	85.8	88.6	84.4	87.1	94.1	90.3	91.2	90.0	91.4
$\mathcal{L}_{ams} + \mathcal{L}_{fd} + \mathcal{L}_{ld}$	**80.7**	89.4	86.0	**88.7**	**84.9**	**87.3**	94.0	**90.6**	91.2	**90.3**	**91.5**
LEALLA-base											
$\mathcal{L}_{ams}$	81.7	89.8	85.9	88.6	85.4	87.4	94.2	91.0	91.3	91.1	91.9
$\mathcal{L}_{fd}$	81.1	90.2	87.3	89.4	85.5	88.1	**95.0**	**91.6**	**91.8**	91.3	**92.4**
$\mathcal{L}_{ld}$	80.6	66.3	49.4	51.0	85.7	63.1	57.5	80.1	60.6	88.6	71.7
$\mathcal{L}_{ams} + \mathcal{L}_{fd}$	82.2	90.2	**87.5**	89.4	86.8	88.5	**95.0**	**91.6**	91.7	91.0	92.3
$\mathcal{L}_{ams} + \mathcal{L}_{ld}$	82.3	90.0	**87.5**	89.2	86.8	88.4	94.8	91.3	91.6	**91.4**	92.3
$\mathcal{L}_{ams} + \mathcal{L}_{fd} + \mathcal{L}_{ld}$	**82.4**	**90.3**	87.4	**89.8**	**87.2**	**88.7**	94.9	91.4	**91.8**	**91.4**	**92.4**
LEALLA-large											
$\mathcal{L}_{ams}$	82.9	90.1	87.1	89.3	87.4	88.5	94.6	91.2	91.5	91.4	92.2
$\mathcal{L}_{fd}$	82.4	89.8	87.2	89.4	86.1	88.1	**95.3**	91.8	92.0	**92.2**	**92.8**
$\mathcal{L}_{ld}$	82.3	87.2	78.8	83.3	86.9	84.1	88.4	87.4	86.9	91.8	88.6
$\mathcal{L}_{ams} + \mathcal{L}_{fd}$	83.4	90.6	88.4	89.8	87.7	89.1	**95.3**	**92.0**	92.0	92.0	**92.8**
$\mathcal{L}_{ams} + \mathcal{L}_{ld}$	83.4	90.6	87.9	**90.0**	87.7	89.1	**95.3**	91.8	91.7	92.4	**92.8**
$\mathcal{L}_{ams} + \mathcal{L}_{fd} + \mathcal{L}_{ld}$	**83.5**	**90.8**	**88.5**	89.9	**87.9**	**89.3**	**95.3**	**92.0**	**92.1**	91.9	**92.8**

Table 8: Results of LEALLA with different loss functions and loss combinations.

BUCC for models #0 - #8, we provide the results of a further smaller thin-deep architecture (#9) and MobileBERT-like (Sun et al., 2020) thin-deep architectures (#10 - #12). The 64-d thin-deep architecture contains only 33M parameters. However, its performance on three evaluation benchmarks downgrades by up to 7.4 points compared with #5 - #8, which demonstrates that 128-d may be a lower bound as universal sentence embeddings for aligning parallel sentences for 109 languages. Moreover, #10 - #12 show the results of MobileBERT-like architectures whose feed-forward hidden size is identical to hidden size. They have fewer parameters than #5 - #8, but they perform worse than #5 - #8, respectively (e.g., compare #10 with #6). Therefore, we did not employ MobileBERT-like architectures for LEALLA.

F Results of Ablation Study

We report all the results of the ablation study (Section 4.2) in Table 8.

Synthesizing Human Gaze Feedback for Improved NLP Performance

Varun Khurana[*]
Adobe, IIIT Delhi
varun19124@iiitd.ac.in

Yaman Kumar Singla[*]
Adobe, IIIT Delhi, SUNY-Buffalo
ykumar@adobe.com

Nora Hollenstein
University of Copenhagen
nora.hollenstein@hum.ku.dk

Rajesh Kumar
Bucknell University
rajesh.kumar@bucknell.edu

Balaji Krishnamurthy
Adobe
kbalaji@adobe.com

Abstract

Integrating human feedback in models can improve the performance of natural language processing (NLP) models. Feedback can be either explicit (*e.g.* ranking used in training language models) or implicit (*e.g.* using human cognitive signals in the form of eyetracking). Prior eye tracking and NLP research reveal that cognitive processes, such as human scanpaths, gleaned from human gaze patterns aid in the understanding and performance of NLP models. However, the collection of *real* eyetracking data for NLP tasks is challenging due to the requirement of expensive and precise equipment coupled with privacy invasion issues. To address this challenge, we propose ScanTextGAN, a novel model for *generating* human scanpaths over text. We show that ScanTextGAN-generated scanpaths can approximate meaningful cognitive signals in human gaze patterns. We include synthetically generated scanpaths in four popular NLP tasks spanning six different datasets as proof of concept and show that the models augmented with generated scanpaths improve the performance of all downstream NLP tasks.

1 Introduction

Integrating human signals with deep learning models has been beginning to catch up in the last few years. Digital traces of human cognitive processing can provide valuable signals for Natural Language Processing (Klerke et al., 2016a; Plank, 2016). Various approaches for integrating human signals have been explored. For example, human feedback for better decisioning (Christiano et al., 2017), NLP tasks (Stiennon et al., 2020; Wu et al., 2021), and most recently language modeling using reinforcement learning with human feedback (RLHF) based reward (Bai et al., 2022; Ouyang et al., 2022). RLHF involves explicit human feedback and is expensive and hard to scale. On the other hand, previous studies have also tried to use implicit human feedback in the form of eyetracking signals. It has proven to be a useful signal for inferring human cognitive processing (Sood et al., 2020; Hollenstein and Zhang, 2019; Mathias et al., 2020). NLP researchers have focused on assessing the value of gaze information extracted from large, mostly dis-jointly labeled gaze datasets in recurrent neural network models (Ren and Xiong, 2021; Strzyz et al., 2019; Barrett et al., 2018a). The proposed approaches under this paradigm include gaze as an auxiliary task in multi-task learning (Klerke et al., 2016b; Hollenstein et al., 2019), as additional signals (Mishra et al., 2016b), as word embeddings (Barrett et al., 2018b), as type dictionaries (Barrett et al., 2016a; Hollenstein and Zhang, 2019), and as attention (Barrett et al., 2018a).

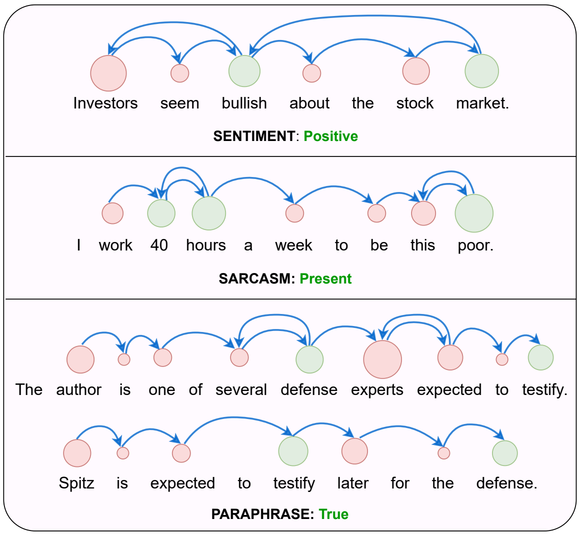


Figure 1: Generated scanpaths over text samples taken from various natural language processing (NLP) tasks. The green circles denote the important words characteristic of that task. The circles' size denotes the fixation duration, and the arrows depict the saccadic movements. As can be seen, linguistically important words often have a higher fixation duration and revisit. Regressions (word revisits) also appear in the examples.

Previous studies demonstrate that human scanpaths (temporal sequences of eye fixations, see Fig. 1) gleaned from eye tracking data improve the

---

[*] Equal Contribution

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1895–1908
May 2-6, 2023

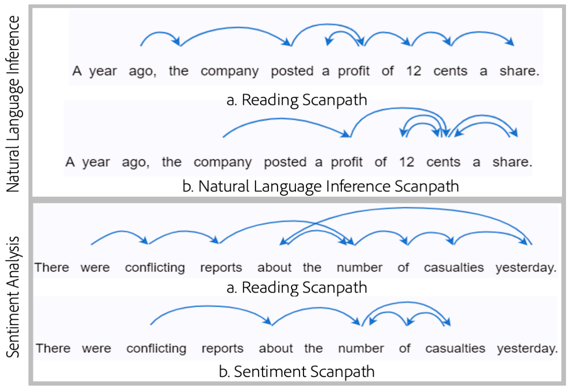


Figure 2: (Intent-aware) Scanpath samples generated by conditioning scanpath generation on different downstream natural language tasks. Note that the conditioned scanpaths are heavily biased to words important for that downstream task.

performance of NLP models. However, the real-world application of these methods remains limited primarily due to the cost of precise eye-tracking equipment, users' privacy concerns, and manual labor associated with such a setup. Therefore, generating scanpaths from existing eyetracking corpora would add great value to NLP research. To the best of our knowledge, this is the first paper to propose a model that generates scanpaths for a given read text with good accuracy. We call the model, ScanTextGAN.

We demonstrate the scanpath generation capability of ScanTextGAN over three eye-tracking datasets using multiple evaluation metrics. Further, we evaluate the utility of *generated* scanpaths for improvements in the performance of multiple NLP tasks (see Figs. 1,2) including the ones in the GLUE benchmark (Wang et al., 2018). The generated scanpaths achieve similar performance gains as the models trained with real scanpaths for classic NLP tasks like sentiment classification, paraphrase detection, entailment, and sarcasm detection.

Our contributions are threefold:

1. We propose ScanTextGAN, the first scanpath generator over text.

2. We compare ScanTextGAN with multiple baselines and conduct ablation experiments with varying models and configurations. The model performs well on the test sets and cross-domain generalization on two additional eyetracking datasets belonging to different text domains.

3. We tested the usefulness of generated scanpaths in downstream NLP tasks such as sentiment analysis, paraphrase detection, and sarcasm detection on six different datasets. The results show that the downstream NLP tasks benefited significantly from cognitive signals inherent in generated scanpaths. Further, we show how scanpaths change when finetuning with downstream natural language tasks (Figs.2,6) and that they lead to further improvements in downstream task performance (§4.3) showing how they can act as additional controls beyond the task architecture.

2 Related Work

When reading a text, humans do not focus on every word and often do not read sequentially (Just and Carpenter, 1980). A series of studies in psycholinguistics have shown that the number of fixations and the fixation duration on a word depend on several linguistic factors. The linguistic factors can also be determined given the cognitive features (Clifton Jr et al., 2007; Demberg and Keller, 2008). Though advances in ML architecture have helped bring machine comprehension closer to human performance, humans are still superior for most NLP tasks (Blohm et al., 2018; Xia et al., 2019).

It has been shown in the literature that integrating explicit (Bai et al., 2022; Ouyang et al., 2022) and implicit (cognitive processing) human feedback signals in traditional ML models is expected to improve their performance (Just and Carpenter, 1980). However, the cost of explicit feedback (e.g., using MTurk) and implicit feedback (e.g., eye tracking) at scale is excessively high. Similarly, privacy-invasive eye-tracking processes limit the scope of this idea. One way to address this problem is to use generated eye movements to unfold the full potential of eye-tracking research. Hence, the idea is to architect ScanTextGAN, a scanpath generator for text reading, and test its usefulness in downstream NLP tasks.

More precisely, this work builds upon previous works on 1) human attention modeling and 2) gaze integration in neural network architectures, which are described as follows:

Human Attention Modeling: Predicting what people visually attend to in images (saliency prediction) is a long-standing challenge in neuroscience and computer vision, the fields have seen many data-based models (Wang et al., 2021). In contrast to images, most attention models for eye movement behaviors during reading are cognitive process models, *i.e.*, models that do not involve machine learning but implement cognitive theories (Engbert et al., 2005; Xia et al., 2019). Key challenges for such models are a limited number of parameters and hand-crafted rules. Thus, it is difficult to adapt them to different tasks and domains and use them as part of end-to-end trained ML architectures (Kotseruba and Tsotsos, 2020). In contrast, learning-based attention models for text remain under-explored. Within that, all eye tracking models are saliency prediction models with non-existent work in predicting scanpaths. On the other hand, visual scanpaths generation for image-based eye tracking data has been recently explored for both traditional (Assens et al., 2019) and 360° images (Martin et al., 2022).

Matthies and Søgaard (2013) presented the first fixation prediction work for text. They built a person-independent model using a linear Conditional Random Fields (CRF) model. Hahn and Keller (2016) designed the Neural Attention Trade-off (NEAT) language model, which was trained with hard attention and assigned a cost to each fixation. Other approaches include sentence representation learning using surprisal and part of speech tags as proxies to human attention (Wang et al., 2017).

Our work differs from previous studies as we combine cognitive theory and data-driven approaches to predict scanpaths and further show its application in downstream NLP tasks (Hollenstein et al., 2021b,a).

Integrating Gaze in Network Architecture: Integration of human gaze data into neural network architectures has been explored for a range of computer vision tasks such as image captioning, visual question answering, and tagging (Karessli et al., 2017; Yu et al., 2017; He et al., 2019; Boyd et al., 2022). Hence, recent research has utilized features gleaned from readers' eye movement to improve the performance of complex NLP tasks such as sentiment analysis (Long et al., 2017; Mishra et al., 2016c), sarcasm detection (Mishra et al., 2016b), part-of-speech tagging (Barrett et al., 2016b), NER (Hollenstein and Zhang, 2019), and text difficulty (Reich et al., 2022).

While in recent years, eye tracking data has been used to improve and evaluate NLP models, the scope of related studies remains limited due to the requirement of real-time gaze data at inference time. Mathias et al. (2020) reported that there exists no automated way of generating scanpaths yet in the literature. With high-quality artificially generated scanpaths, the potential of leveraging eyetracking data for NLP can be unfolded. Additionally, generating scanpaths that mimic human reading behavior will help advance our understanding of the cognitive processes behind language understanding. Hence, we propose ScanTextGAN; researchers can use that to generate scanpaths over any text without worrying about collecting them from real users.

3 Proposed Model

In this section, we define the scanpath generation task, describe the ScanTextGAN model architecture, and provide details on loss functions and model training.

Task Definition: The task of scanpath generation is to generate a sequence $\mathcal{S}(\mathcal{T})$ representing a scanpath over the text $\mathcal{T} = \{w_1, w_2, ..., w_n\}$ composed of a sequence of words, can be defined as follows:

$$\mathcal{S}(\mathcal{T}) = \{.., (w_a^i, t^i), ..., (w_b^j, t^j), ..., (w_c^k, t^k)\} \tag{1}$$

where t^i represents the fixation duration over the word w_a occurring at the position i. Note that it is not necessary to have $a < b$ (words being read in linear order) or that $k = n$ (the number of fixations being equal to the number of words). Due to regressions, *i.e.*, backward saccades to previous words, words are also revisited. Hence, the same word could appear multiple times in the sequence.

3.1 ScanTextGAN Model Architecture

Fig. 3 illustrates the proposed conditional GAN architecture of the model. The ScanTextGAN model is composed of two competing agents. First, a conditional generator that generates scanpaths given text prompts. The second is a discriminator network, which distinguishes real human scanpaths from the generated ones. The ScanTextGAN model is trained by combining text content loss, scanpath content loss, and adversarial loss (Eq. 6). The scanpath content loss measures the difference between the predicted scanpath and the corresponding ground truth scanpath. The text content loss reconstructs the input text, and the adversarial loss depends on the real/synthetic prediction of the discriminator over the generated scanpath. We describe the losses along with the generator and discriminator architectures next.

Generator: The ScanTextGAN generator constitutes a transformer-based encoder-decoder framework. The encoder is conditioned on BERT-based text embeddings (Devlin et al., 2019), which are concatenated with noise to make the

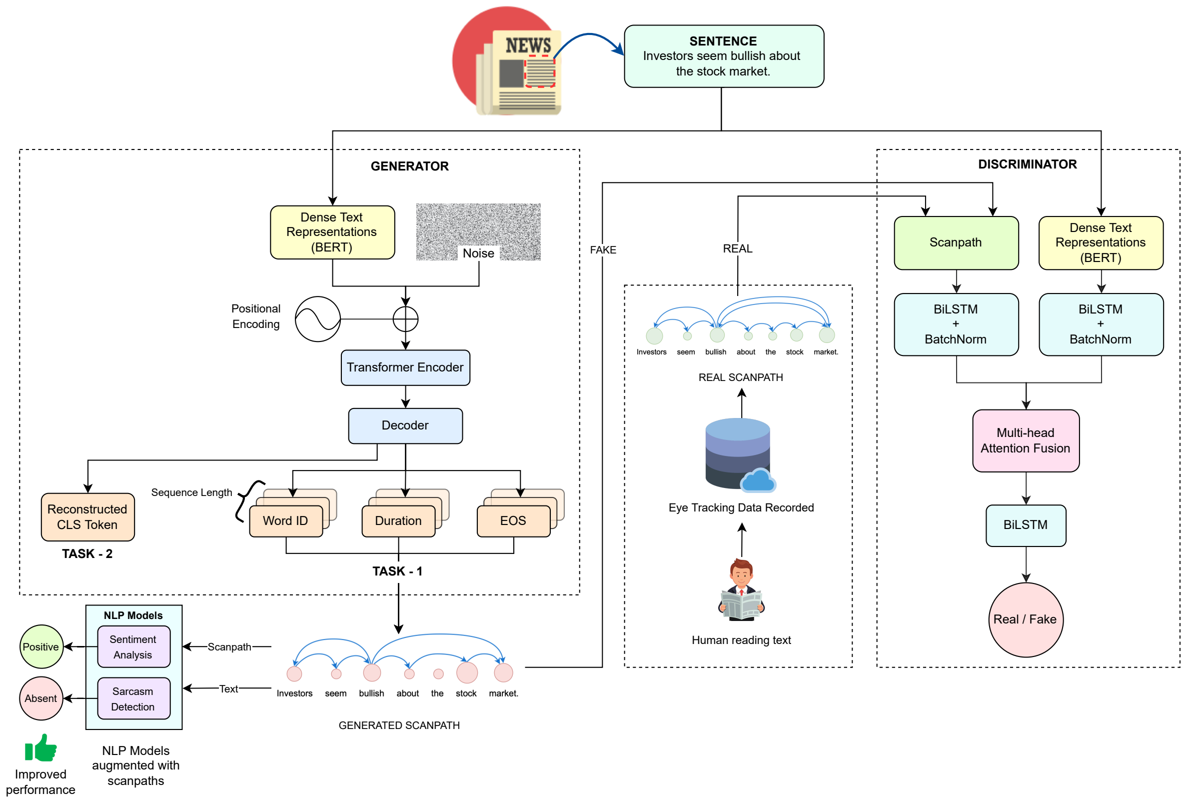


Figure 3: The architecture of the proposed **ScanTextGAN** model. The model consists of a conditional generator and a discriminator playing a zero-sum game. The generator is trained by two cognitively inspired losses: text content reconstruction and scanpath content reconstruction.

generator's output non-deterministic. The output of the transformer encoder is supplied to the decoder, which consists of task-specific feed-forward networks. One branch generates the scanpath (*Task 1*), while the other reconstructs the 768 dimensional CLS token embedding of the sentence (*Task 2*). The scanpath is output as a temporal sequence of word ID (fixation points) w_a^i, fixation duration t^i, and end-of-sequence probability EOS^i. At inference time, the length $L(G)$ of generated scanpath G is determined as follows:

$$L(G) = \begin{cases} \min_{1 \leq k \leq M}(k) & \text{if } EOS^k > \tau \\ M & \text{otherwise} \end{cases} \quad (2)$$

where M is the maximum scanpath length as described in section §3.2 and $\tau \in (0, 1)$ is a probability threshold. We use $\tau = 0.5$. The loss functions of the two branches are described below.

Scanpath Content Loss tries to minimize the deviation of generated scanpaths $\mathcal{G}(\mathcal{T},\mathcal{N})$ from the ground-truth scanpaths $\mathcal{R}(\mathcal{T},h))$ over text $\mathcal{T}$ where ground-truth scanpaths are recorded from the human h and $\mathcal{N}$ stands for Gaussian noise $\mathcal{N}(0, 1)$. The loss function $\mathbb{L}_s$ is given as:

$$\mathbb{L}_s(\mathcal{G}(\mathcal{T},\mathcal{N}),\mathcal{R}(\mathcal{T},h)) = \frac{1}{k}\Sigma_{i=0}^{k}(\alpha(id_g^i - id_r^i)^2 + \beta(t_g^i - t_r^i)^2 + \gamma(E_g^i - E_r^i)^2) \quad (3)$$

which is a weighted sum of three terms. The first term measures the error between real and predicted *fixation points* given by the mean squared difference between generated and real word-ids $(id_g^i - id_r^i)$. It penalizes permutations of word ids and trains the model to approximate the real sequence of fixation points closely.

The second term measures the difference in *fixation durations* given by the mean squared difference between generated and real duration $(t_g^i - t_r^i)$. Fixation durations simulate human attention over words in the input text. Thus, a word with a larger fixation duration is typically synonymous with greater importance than other words in the input text. This error term supplements the generator's ability to learn human attention patterns over the input text.

Finally, the third term measures the mean squared error between the prediction of end-of-sequence probability by real and generated distributions ($E_g^i - E_r^i$). These are weighted by the hyperparameters α, β, and γ. Preliminary experiments showed that optimizing the mean squared error leads to better performance over the cross-entropy loss for optimizing the EOS probability output.

Text Content Loss: Scanpaths depend heavily on the linguistic properties of the input text. Therefore, to guide the generator towards near the probable real data manifolds, we adopt reconstruction of the CLS token embedding of the input text (*Task 2*) by the generator as an auxiliary task since the CLS token embedding encodes a global representation of the input text. This text content reconstruction loss $\mathbb{L}_r$ is given as:

$$\mathbb{L}_r(\mathcal{G}(\mathcal{T},\mathcal{N}), \mathcal{R}(\mathcal{T}, h)) = (BERT(w_i^g, w_j^g, ..., w_k^g \\ - BERT(w_a^r, w_b^r, ...w_n^r))^2 \quad (4)$$

where $BERT(w_a^r, w_b^r, ...w_n^r)$ and $BERT(w_i^g, w_j^g, ...w_k^g)$ stand for the *CLS* vector representations of real and generated text respectively.

Discriminator: The goal of the discriminator is to distinguish between the real and synthetic scanpaths supplied to it. Similar to the generator, it requires text representations to distinguish between real and generated scanpaths. Specifically, the discriminator comprises two blocks of BiLSTMs that perform sequential modeling over the scanpaths and BERT embeddings. The outputs of the two branches are combined and passed to an attention fusion module with four heads, followed by another network of BiLSTMs. The hidden states of the last BiLSTM layer from both forward and backward directions are concatenated and supplied to a feed-forward network. A Sigmoid function activates the output of the feed-forward network. In this manner, the discriminator classifies the input scanpaths as either *real* or *fake*.

Adversarial Loss: The generator and discriminator networks are trained in a two-player zero-sum game fashion. The loss is given by:

$$\mathbb{L}_a = \min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x|\mathcal{T}, h)] + \\ \mathbb{E}_{z \sim p_z(z)}[1 - \log D(G(z|\mathcal{T}, \mathcal{N}))] \quad (5)$$

Therefore, the net generator loss becomes:

$$\mathbb{L}_g = \mathbb{L}_s + \mathbb{L}_r + \mathbb{E}_{z \sim p_z(z)}[1 - \log D(G(z|\mathcal{T}, \mathcal{N}))] \quad (6)$$

3.2 Dataset

For training the ScanTextGAN model, we use the CELER dataset (Berzak et al., 2022). It contains eyetracking data of 365 participants for nearly 28.5 thousand newswire sentences, sourced from the Wall Street Journal Penn Treebank (Marcinkiewicz, 1994). Each participant in CELER reads 156 newswire sentences. Half of the sentences are shared across participants, and the rest is unique to each participant. The maximum sentence length was set to 100 characters. Participant eyetracking data were recorded using Eyelink 1000 tracker in a desktop mount configuration with a sampling rate of 1000 Hz. The ScanTextGAN model is trained to approximate the average eye movements of all the participants who read given sentences. The CELER dataset was envisioned to enable research on language processing and acquisition and to facilitate interactions between psycholinguistics and natural language processing. Furthering the goal, we use it to train our conditional GAN model through which we show human scanpath approximation capabilities (§4.2). Also, we use it to show improvements in the performance of NLP tasks (§4.3).

The data consist of tuples of participant ID, sentence ID, and word ID corresponding to fixation point and fixation duration. We compute the 99th percentile of fixation durations and treat it as the largest value. Fixations of durations longer than this are treated as outliers and hence dropped from the dataset. To apply the scanpath reconstruction loss (Eq. 3), we scale all fixation durations by the maximum value and then normalize them to [0,1]. Similarly, word IDs in each sentence are normalized to [0, 1] after scaling them by the length of that sentence. For the last fixation point in every scanpath, the binary EOS token is set to 1. The maximum scanpath length is set to 80 fixation points (99th percentile of the lengths). Thus shorter scanpaths are padded while longer scanpaths are trimmed. We use BERT to encode the sentences and obtain their 768-dimensional embeddings, keeping the max length parameter as 80, thus resulting in an 80 × 768 dimensional tensor.

3.3 Parameter Settings

Sinusoidal positional encoding is applied over the input embeddings fed to the generator. We use a 3-layer transformer encoder with four head attention and a hidden dimension size of 776 in the generator. In the discriminator, we use bidirectional LSTMs over sentence embeddings and generated scanpaths with a hidden size of 64 and a dropout ratio of 0.3, followed by batch normalization for faster convergence. An attention module with four attention heads is applied after concatenating the outputs. We employ the Adam and RMSProp optimizer to minimize generator and discriminator losses. The batch size is set to 128, the initial learning rate of the generator to 0.0001, and that of the discriminator to 0.00001. The model is trained for 300 epochs. Our implementation uses PyTorch, a popular deep-learning framework in Python. All experiments are run on an Intel Xeon CPU with Nvidia A100-SXM GPUs.

4 Performance Evaluation

We quantify the performance of ScanTextGAN in two regimes[1]; first, scanpath generation with three datasets, and second, NLP tasks with six datasets. Similar to prior computer vision studies (Sun et al., 2019; de Belen et al., 2022; Kümmerer and Bethge, 2021; Jiang et al., 2016), we evaluate the ScanTextGAN model over the scanpath generation task. For this, we use the test split of the CELER dataset, Mishra et al. (2016a), and Mishra et al. (2017). In addition, unlike the computer vision studies, we also evaluate the ScanTextGAN model for improvement in NLP tasks. The hypothesis is that the human eyes (and consequently the brain) process many language comprehension tasks unconsciously and without visible effort. The next logical step is to capture (or, in our case, generate) this mental representation of language understanding and use it to improve our machine-learning systems. For evaluation, we use four tasks from the GLUE benchmark and two from the tasks proposed by Mishra et al. (2016a). While the ScanTextGAN model is trained over news text from the CELER dataset, with the help of the other datasets, we expand our testing to other domains, including reviews, quotes, tweets, and Wikipedia text.

4.1 Evaluation Datasets

Mishra et al. (2017) comprises eye movements and reading difficulty data recorded for 32 paragraphs on 16 different topics, *viz.* history, science, literature, *etc*. For each topic, comparable paragraphs were extracted from Wikipedia[2] and simple Wikipedia[3]. The participant's eye movements are tracked using an SR-Research Eyelink-1000 Plus eye tracker. Using the ground truth scanpaths over the text corpora, we evaluate the quality of generated scanpaths.

Mishra et al. (2016a) contains eye fixation sequences of seven participants for 994 text snippets annotated for sentiment and sarcasm. These were taken from Amazon Movie Corpus , Twitter, and sarcastic quote websites. The task assigned to the participants was to read one sentence at a time and annotate it with binary sentiment polarity labels (*i.e.*, positive/negative). The same datasets were used in several studies (Joshi et al., 2015; Mishra et al., 2016b,c) to show improvements in sarcasm and sentiment analysis. We use the datasets to evaluate both the generation quality and potential improvements in NLP tasks.

Furthermore, we explore the potential of including cognitive signals contained in scanpaths in NLP models for a range of GLUE tasks which include Sentiment Analysis using Stanford Sentiment Treebank (SST), Paraphrase Detection using Microsoft Research Paraphrase Corpus (MRPC) and Quora Question Pairs (QQP), Natural Language Inference using Recognizing Textual Entailment (RTE) dataset.

Next, we cover the results of scanpath generation and its application in NLP tasks.

4.2 Evaluation of Scanpath Generation

We evaluate the scanpath generation model on two most commonly used metrics in image scanpath generation studies (Sun et al., 2019; Chen and Sun, 2018; de Belen et al., 2022; Kümmerer et al., 2022): **MultiMatch** (Jarodzka et al., 2010) and **Levenshtein Distance** (Levenshtein, 1965). Multimatch is a geometrical measure that compares scanpaths across a comprehensive set of dimensions composed of shape, lengths, position, and fixation duration. Levenshtein Distance between a pair of sequences measures the least number of edits (inserts, deletes, substitution) to transform one into the other. More details are discussed in Appendix:A.

Further, as a top-line comparison, we use **inter-subject scanpath similarity** (Sun et al., 2019). It

---

[1] All results are calculated with five random seeds and reported as the mean of those five runs

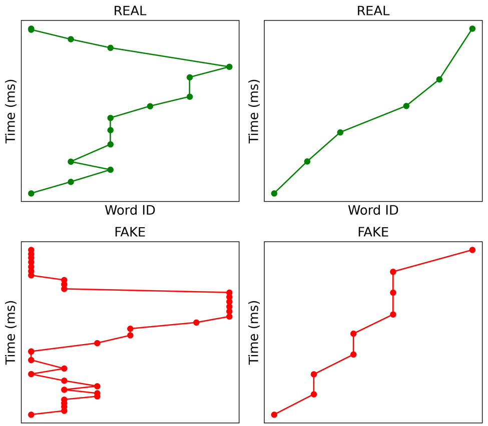
[2] `https://en.wikipedia.org/`

[3] `https://simple.wikipedia.org/`

[4] In the CELER dataset, there are only 78 shared sentences amongst all the participants. Therefore, inter-subject scanpath evaluation is done only for these sentences. In contrast, the ScanTextGAN results are reported for the entire test set (including these 78 sentences).

Generator Model	MultiMatch ↑				Levenshtein Distance ↓
	Vector↑	Length↑	Position↑	Duration↑	
Inter-subject score[4]	0.973	0.958	0.830	0.698	0.691
LSTM Encoder-Decoder trained with scanpath content loss	0.975	0.956	0.765	0.344	0.865
ScanTextGAN – Text Reconstruction – GAN Loss	0.968	0.947	0.728	0.703	0.779
ScanTextGAN	**0.983**	**0.972**	**0.787**	0.733	**0.769**
ScanTextGAN – Text Reconstruction	0.974	0.957	0.773	0.703	0.798
ScanTextGAN – GAN Loss	0.973	0.955	0.750	**0.761**	0.786
ScanTextGAN + addition of noise	0.971	0.952	0.756	0.736	0.791
ScanTextGAN – Text (CLS) Reconstruction + sentence reconstruction	0.978	0.963	0.724	0.721	0.805

Table 1: In-domain Evaluation of Scanpath Generation on the CELER dataset (Berzak et al., 2022).



Mr. Baker promises to return if the haunting continues.

Soon she was running the office.

Figure 4: Comparison of *real* and *synthesized* scanpaths corresponding to a few text samples. The proposed ScanTextGAN model generates the latter.

measures the degree of variation among real human scanpaths corresponding to each text input. To compute this, we first calculate each subject's performance by treating the scanpaths of other subjects as the ground truth. Then, the average value of all subjects is used as inter-subject performance.

Baselines: Since ScanTextGAN is the first text-based scanpath generation model, we conduct an ablation study to compare ScanTextGAN with its other variants. Specifically, we compare ScanTextGAN with the following six configurations: (1) An LSTM-based network trained with scanpath content loss. Sentence embeddings obtained through BERT are concatenated with noise in this model. The resultant is fed to an attention module with four heads, then passed to a network of LSTMs and Batch Normalization layers applied in tandem. (2) ScanTextGAN model trained with only the scanpath content loss. (3) ScanTextGAN model without the text reconstruction loss (Task-2). (4) ScanTextGAN model with BERT-based sentence embeddings reconstruction instead of CLS token reconstruction. (5) ScanTextGAN model with the addition of noise instead of concatenation. (6) ScanTextGAN model trained without GAN loss.

Results: Table 1 presents the results of our scanpath prediction model on the CELER dataset. Further, we also compare ScanTextGAN with baselines on two other contemporary datasets of movie reviews, tweets, and sarcastic quotes (Mishra et al., 2016a), Wikipedia and simple Wikipedia paragraphs (Mishra et al., 2017). Tables 2 and 3 present the results of our model on those datasets. For obtaining results on these corpora, we use the model trained on the CELER dataset, thus helping us evaluate the cross-domain performance of the model.

As can be seen in Table 1, Table 2 and Table 3, ScanTextGAN outperforms other models for scanpath prediction on most metrics. The performance of ScanTextGAN even surpasses inter-subject reference on Duration and comes very close to Vector, Length, and Position.

We observe that adopting the reconstruction of the CLS token as an auxiliary task (Task - 2) boosts the model performance. Reconstructing the full sentence embeddings rather than the CLS tokens only as an auxiliary task does not always improve the results, despite adding a larger computational overhead. The results also reveal that concatenating noise with text embeddings is more rewarding than adding it.

Further, to compare the skipping behavior of ScanTextGAN with humans, we calculate the weighted F1 score of the words skipped and attended by both model types. We find the weighted F1 to be 64.6 between them. Fig. 4 presents a visual comparison between real scanpaths from the available eyetracking data and scanpaths generated by ScanTextGAN, corresponding to some randomly chosen text samples. We can observe that the generated scanpaths resemble the real ones to a great

Generator Model	**MultiMatch ↑**				**Levenshtein Distance ↓**
	Vector↑	**Length↑**	**Position↑**	**Duration↑**	
Inter-subject score	0.977	0.963	0.839	0.715	0.723
LSTM Encoder-Decoder trained with scanpath content loss	**0.984**	**0.973**	0.714	0.379	0.918
ScanTextGAN – Text Reconstruction – GAN Loss	0.977	0.960	0.780	0.769	0.847
ScanTextGAN	0.966	0.945	**0.791**	**0.771**	**0.836**
ScanTextGAN – Text Reconstruction	0.976	0.961	0.763	0.757	0.845
ScanTextGAN – GAN Loss	0.976	0.959	0.774	0.768	0.839
ScanTextGAN + addition of noise	0.968	0.947	0.737	0.743	0.838
ScanTextGAN – Text (CLS) Reconstruction + sentence reconstruction	0.964	0.934	0.747	0.733	0.869

Table 2: Cross-domain Evaluation of Scanpath Generation on the Dataset by Mishra et al. (2016a).

Generator Model	**MultiMatch ↑**				**Levenshtein Distance ↓**
	Vector↑	**Length↑**	**Position↑**	**Duration↑**	
Inter-subject score	0.994	0.991	0.834	0.620	0.845
LSTM Encoder-Decoder trained with scanpath content loss	**0.992**	**0.987**	0.596	0.329	0.969
ScanTextGAN – Text Reconstruction – GAN Loss	0.990	0.984	0.729	0.705	0.951
ScanTextGAN	0.984	0.977	**0.759**	0.693	**0.931**
ScanTextGAN – Text Reconstruction	0.986	0.981	0.756	**0.706**	0.939
ScanTextGAN – GAN Loss	0.990	0.984	0.739	**0.706**	0.945
ScanTextGAN + addition of noise	0.984	0.976	0.759	0.703	0.943
ScanTextGAN – Text (CLS) Reconstruction + sentence reconstruction	0.983	0.974	0.667	0.674	0.958

Table 3: Cross-domain Evaluation of Scanpath Generation on the Dataset by Mishra et al. (2017).

extent. Thus, the quantitative and qualitative results on in-domain and cross-domain settings lead us to believe that our proposed scanpath generation model can be deemed a good approximator of the human scanpaths.

4.3 Application to NLP Tasks

We use them to augment various NLP models and measure their performance to demonstrate the usefulness of cognitive signals hidden in the *generated* scanpaths.

Sentiment Classification and Sarcasm Detection: For these tasks, we use a model consisting of a network of two branches of BiLSTMs and Batch Normalization layers that perform sequential modeling over text representations obtained through BERT and scanpaths fed as input to the model. The outputs of both branches are combined and passed to another layer of BiLSTMs, followed by a feed-forward network that predicts binary sentiment/sarcasm labels corresponding to the input after activating with the Sigmoid function. We follow a 10-fold cross-validation regime.

We compare the models with generated scanpaths, real scanpaths, and without scanpaths. Further, to investigate whether performance gains observed by adding scanpaths are due to scanpaths and not the increase in the number of parameters, we train a *Random-Random* variant in which we send Random noise as scanpaths to the model with an increased number of parameters. We also simulate the real-world case where both real and generated scanpaths are available during train time, but only generated ones are available during test time, for example, during user deployment.

Model Configuration		**F1 score**	
Train	**Test**	**Sentiment**	**Sarcasm**
w/o	w/o	0.7839	0.9438
Random	Random	0.7990	0.9397
Random	Generated	0.7773	0.9313
Real	Generated	0.8319	0.9378
Real	Real	0.8334	0.9501
Generated	Real	0.8402	0.9452
Generated	Generated	0.8332	0.9506
Real + Generated	Generated	**0.8404**	**0.9512**
Intent-Aware	Intent-Aware	**0.8477**	**0.9528**

Table 4: Sentiment analysis and sarcasm detection results on the dataset by Mishra et al. (2016a). Model configuration refers to the type of scanpath included in train and test data.

Table 4 records the results of sentiment analysis and sarcasm detection tasks (Mishra et al., 2016a). We note that generated scanpaths training and testing lead to similar gains for sentiment analysis and sarcasm detection as real scanpaths. The model with an increased number of parameters fed random noise in place of scanpaths performs similarly to the model trained without any scanpaths. Interestingly, the best results are obtained when model training uses both real and generated scanpaths. We believe this is due to ScanTextGAN bringing additional cognitive information from the news-reading CELER corpus, which is not present in the

Dataset	Model	Acc	F1 score
SST	w/o scanpaths	0.8090	0.8089
	w/ random scanpaths	0.8059	0.8061
	w/ generated scanpaths	**0.8138**	**0.8138**
	w/ intent-aware scanpaths	**0.8269**	**0.8272**
MRPC	w/o scanpaths	0.6902	0.6656
	w/ random scanpaths	0.6623	0.6680
	w/ generated scanpaths	**0.6969**	**0.6828**
	w/ intent-aware scanpaths	**0.7009**	**0.6911**
RTE	w/o scanpaths	0.6162	0.6080
	w/ random scanpaths	0.5802	0.5794
	w/ generated scanpaths	**0.6211**	**0.6205**
	w/ intent-aware scanpaths	**0.6293**	**0.6278**
QQP	w/o scanpaths	0.8499	0.8513
	w/ random scanpaths	0.8491	0.8503
	w/ generated scanpaths	**0.8578**	**0.8596**
	w/ intent-aware scanpaths	**0.8648**	**0.8658**

Table 5: Results of training NLP models with and without scanpaths on the GLUE benchmark tasks. Including scanpaths leads to consistent improvements across all the NLP tasks.

real scanpaths in Mishra et al. (2016a). In addition to the intrinsic evaluation presented in §4.2, this downstream evaluation demonstrates the high quality of the synthesized scanpaths, showing that they contain valuable cognitive processing signals for NLP tasks.

GLUE Tasks: To validate further, we augment classification models (based on sequential modeling using LSTMs) with generated scanpaths to show performance improvement in downstream NLP tasks on four GLUE benchmark datasets – SST, MRPC, RTE, QQP as described in §4.1. Table 5 reports the accuracy and weighted-F1 scores of the models trained with and without scanpaths for these tasks. We observe that in all four tasks, the model trained with generated scanpaths outperforms the one without scanpaths.

Intent-Aware Scanpaths: Finally, we try to condition scanpaths generation on the downstream natural language task. We back-propagate gradients from the downstream NLP task to the conditional generator. In this fashion, the model learns to generate *intent-aware* scanpaths. The hypothesis is that finetuning scanpath generation based on feedback from the natural language task will bias the generator towards words more pertinent to that task and thus could help further improve performance on the downstream task. The architecture is shown in Appendix: Fig 5. The results in Tables 4 and 5 validate the hypothesis that we observe consistent improvements in all downstream tasks. Fig 2 and Appendix: Fig 6 show a few examples of scanpaths and saliency generated for three downstream natural language tasks.

Together these results corroborate the hypothesis that leveraging the cognitive signals approximated by synthetic scanpaths in NLP models leads to performance gains.

5 Conclusion

In this work, we make two novel contributions toward integrating cognitive and natural language processing. (1) We introduce the first scanpath generation model over text, integrating a cognitive reading model with a data-driven approach to address the scarcity of human gaze data on text. (2) We propose generated scanpaths that can be flexibly adapted to different NLP tasks without needing task-specific ground truth human gaze data. We show that both advances significantly improve performance across six NLP datasets over various baselines. Our findings demonstrate the feasibility and significant potential of combining cognitive and data-driven models for NLP tasks. Without the need for real-time gaze recordings, the potential research avenues for augmenting and understanding NLP models through the cognitive processing information encoded in synthesized scanpaths are multiplied.

6 Limitations

In this work, we demonstrated artificial scanpath generation over multiple eye-tracking datasets. Further, our experiments build a link between cognitive and natural language processing and show how one can inform the other. However, the proposed method has a few limitations, which we aim to address in the future. The field needs work on bigger and more diverse eye-tracking datasets, which can enable scanpath generation over longer text sequences and can model generating scanpaths conditioned on previously read context. Besides, a better understanding of the entire scanpath generation process can help model the intra and inter-sentence scanpath generation process. The understanding would enable the integration of scanpaths to generative modeling tasks, which we intend to take up in future work. Another parallel direction is to include both explicit (like using RLHF) and implicit signals (like using cognitive signals) to better NLP tasks like language modeling.

References

Marc Assens, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E. O'Connor. 2019. Pathgan: Visual scanpath prediction with generative adversarial networks. In *Computer Vision – ECCV 2018 Workshops*. Springer International Publishing.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Maria Barrett, Joachim Bingel, Nora Hollenstein, Marek Rei, and Anders Søgaard. 2018a. Sequence classification with human attention. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 302–312, Brussels, Belgium. Association for Computational Linguistics.

Maria Barrett, Joachim Bingel, Frank Keller, and Anders Søgaard. 2016a. Weakly supervised part-of-speech tagging using eye-tracking data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–584, Berlin, Germany. Association for Computational Linguistics.

Maria Barrett, Ana Valeria González-Garduño, Lea Frermann, and Anders Søgaard. 2018b. Unsupervised induction of linguistic categories with records of reading, speaking, and writing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2028–2038, New Orleans, Louisiana. Association for Computational Linguistics.

Maria Barrett, Frank Keller, and Anders Søgaard. 2016b. Cross-lingual transfer of correlations between parts of speech and gaze features. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1330–1339, Osaka, Japan. The COLING 2016 Organizing Committee.

Yevgeni Berzak, Chie Nakamura, Amelia Smith, Emily Weng, Boris Katz, Suzanne Flynn, and Roger Levy. 2022. Celer: A 365-participant corpus of eye movements in l1 and l2 english reading. *Open Mind*.

Matthias Blohm, Glorianna Jagfeld, Ekta Sood, Xiang Yu, and Ngoc Thang Vu. 2018. Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 108–118, Brussels, Belgium. Association for Computational Linguistics.

Aidan Boyd, Kevin W Bowyer, and Adam Czajka. 2022. Human-aided saliency maps improve generalization of deep learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2735–2744.

Zhenzhong Chen and Wanjie Sun. 2018. Scanpath prediction for visual attention using ior-roi lstm. In *IJCAI*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Charles Clifton Jr, Adrian Staub, and Keith Rayner. 2007. Eye movements in reading words and sentences. *Eye movements*.

Filipe Cristino, Sebastiaan Mathôt, Jan Theeuwes, and Iain D Gilchrist. 2010. Scanmatch: A novel method for comparing fixation sequences. *Behavior research methods*.

Ryan Anthony Jalova de Belen, Tomasz Bednarz, and Arcot Sowmya. 2022. Scanpathnet: A recurrent mixture density network for scanpath prediction. In *IEEE CVPR*.

Vera Demberg and Frank Keller. 2008. Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition*.

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

Ralf Engbert, Antje Nuthmann, Eike M Richter, and Reinhold Kliegl. 2005. Swift: a dynamical model of saccade generation during reading. *Psychological review*.

Michael Hahn and Frank Keller. 2016. Modeling human reading with neural attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 85–95, Austin, Texas. Association for Computational Linguistics.

Sen He, Hamed R Tavakoli, Ali Borji, and Nicolas Pugeault. 2019. Human attention in image captioning: Dataset and analysis. In *ICCV*.

Nora Hollenstein, Maria Barrett, Marius Troendle, Francesco Bigiolli, Nicolas Langer, and Ce Zhang. 2019. Advancing nlp with cognitive language processing signals. *arXiv preprint arXiv:1904.02682*.

Nora Hollenstein, Emmanuele Chersoni, Cassandra L. Jacobs, Yohei Oseki, Laurent Prévot, and Enrico Santus. 2021a. CMCL 2021 shared task on eye-tracking prediction. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 72–78, Online. Association for Computational Linguistics.

Nora Hollenstein, Federico Pirovano, Ce Zhang, Lena Jäger, and Lisa Beinborn. 2021b. Multilingual language models predict human reading behavior. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 106–123, Online. Association for Computational Linguistics.

Nora Hollenstein and Ce Zhang. 2019. Entity recognition at first sight: Improving NER with eye movement information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1–10, Minneapolis, Minnesota. Association for Computational Linguistics.

Halszka Jarodzka, Kenneth Holmqvist, and Marcus Nyström. 2010. A vector-based, multidimensional scanpath similarity measure. In *Proceedings of the 2010 symposium on eye-tracking research & applications*.

Ming Jiang, Xavier Boix, Gemma Roig, Juan Xu, Luc Van Gool, and Qi Zhao. 2016. Learning to predict sequences of human visual fixations. *IEEE transactions on neural networks and learning systems*.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

Marcel Just and Patricia A. Carpenter. 1980. A theory of reading: From eye fixations to comprehension.

Nour Karessli, Zeynep Akata, Bernt Schiele, and Andreas Bulling. 2017. Gaze embeddings for zero-shot image classification. In *IEEE CVPR*.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016a. Improving sentence compression by learning to predict gaze. In *NAACL: Human Language Technologies*, San Diego, California. Association for Computational Linguistics.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016b. Improving sentence compression by learning to predict gaze. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1528–1533, San Diego, California. Association for Computational Linguistics.

Iuliia Kotseruba and John K Tsotsos. 2020. 40 years of cognitive architectures: core cognitive abilities and practical applications. *Artificial Intelligence Review*.

Matthias Kümmerer and Matthias Bethge. 2021. State-of-the-art in human scanpath prediction. *arXiv preprint arXiv:2102.12239*.

Matthias Kümmerer, Matthias Bethge, and Thomas SA Wallis. 2022. Deepgaze iii: Modeling free-viewing human scanpaths with deep learning. *Journal of Vision*.

V Levenshtein. 1965. Leveinshtein distance.

Yunfei Long, Qin Lu, Rong Xiang, Minglei Li, and Chu-Ren Huang. 2017. A cognition based attention model for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 462–471, Copenhagen, Denmark. Association for Computational Linguistics.

Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Using Large Corpora*.

Daniel Martin, Ana Serrano, Alexander W. Bergman, Gordon Wetzstein, and Belen Masia. 2022. Scangan360: A generative model of realistic scanpaths for 360° images. *IEEE Transactions on Visualization and Computer Graphics*.

Sandeep Mathias, Diptesh Kanojia, Abhijit Mishra, and Pushpak Bhattacharya. 2020. A survey on using gaze behaviour for natural language processing. In *IJCAI*. Survey track.

Franz Matthies and Anders Søgaard. 2013. With blinkers on: Robust prediction of eye movements across readers. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 803–807, Seattle, Washington, USA. Association for Computational Linguistics.

Abhijit Mishra, Diptesh Kanojia, and Pushpak Bhattacharyya. 2016a. Predicting readers' sarcasm understandability by modeling gaze behavior. In *AAAI*.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. 2016b. Harnessing cognitive features for sarcasm detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1095–1104, Berlin, Germany. Association for Computational Linguistics.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. 2016c. Leveraging cognitive features for sentiment analysis. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 156–166, Berlin, Germany. Association for Computational Linguistics.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. 2017. Scanpath complexity: Modeling reading effort using gaze information. *AAAI*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al.

2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Barbara Plank. 2016. Keystroke dynamics as signal for shallow syntactic parsing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 609–619, Osaka, Japan. The COLING 2016 Organizing Committee.

David Robert Reich, Paul Prasse, Chiara Tschirner, Patrick Haller, Frank Goldhammer, and Lena A. Jäger. 2022. Inferring native and non-native human reading comprehension and subjective text difficulty from scanpaths in reading. In *ETRA*. Association for Computing Machinery.

Yuqi Ren and Deyi Xiong. 2021. CogAlign: Learning to align textual neural representations to cognitive language processing signals. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3758–3769, Online. Association for Computational Linguistics.

Ekta Sood, Simon Tannert, Philipp Müller, and Andreas Bulling. 2020. Improving natural language processing tasks with human gaze-guided neural attention. *NeurIPS*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. 2019. Towards making a dependency parser see. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1500–1506, Hong Kong, China. Association for Computational Linguistics.

Wanjie Sun, Zhenzhong Chen, and Feng Wu. 2019. Visual scanpath prediction using ior-roi recurrent mixture density network. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):2101–2118.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2017. Learning sentence representation with guidance of human attention. In *IJCAI*. AAAI Press.

Wenguan Wang, Qiuxia Lai, Huazhu Fu, Jianbing Shen, Haibin Ling, and Ruigang Yang. 2021. Salient object detection in the deep learning era: An in-depth survey. *IEEE T-PAMI*.

Jeff Wu, Long Ouyang, Daniel M Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. 2021. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*.

Menglin Xia, Ekaterina Kochmar, and Ted Briscoe. 2019. Automatic learner summary assessment for reading comprehension. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2532–2542, Minneapolis, Minnesota. Association for Computational Linguistics.

Youngjae Yu, Jongwook Choi, Yeonhwa Kim, Kyung Yoo, Sang-Hun Lee, and Gunhee Kim. 2017. Supervising neural attention models for video captioning by human gaze data. In *IEEE CVPR*.

A Scanpath Evaluation Metrics

MultiMatch is a geometrical measure that models scanpaths as vectors in 2-D space, wherein the vectors represent saccadic eye movements. Starting and ending coordinates of these saccades constitute the fixation positions. It compares scanpaths across multiple dimensions, *viz.* shape, length, position, direction, and fixation duration. Shape measures the vector difference between aligned saccade pairs, which is then normalized by twice the diagonal screen size. Length measures the normalized difference between the endpoints of real and generated saccade vectors. Direction is the angular distance between the two vectors. The position is the Euclidean difference in position between aligned vectors, and duration measures the difference in fixation durations normalized against the maximum duration. Since our work deals with scanpaths over text, we use 1-D space to represent the saccade vectors where word IDs denote the fixation positions. Thus, it is easy to see that computing scanpath direction similarity is redundant here (it is subsumed within position); hence we drop it from our analysis.

Levenshtein Distance between a pair of sequences measures the least number of character edits, i.e., insertion, deletion, and substitution needed to transform one sequence into the other. Specifically, we use it to gauge the degree of dissimilarity between a pair of real R and generated G scanpaths. To account for the fixation durations of each word, R and G are temporally binned using a 50 ms bin size, similar to the computation of ScanMatch metric (Cristino et al., 2010). The resulting sequences of word IDs, R_W and G_W are transformed into character strings, $R_S = \{r_1, r_2, ..., r_n\}$ and $G_S = \{g_1, g_2, ..., g_m\}$, where R_S and G_S are strings over the ASCII alphabet and $n = |R_S|$ and $m = |G_S|$.

Levenshtein Distance (LD) between strings R_S and G_S is computed and then normalized by the length of the longer string, which yields a Normalized Levenshtein Distance (NLD) score, as given below:

$$NLD = \frac{LD(G_S, R_S)}{\max(|R_S|, |G_S|)} \tag{7}$$

Thus, a lower NLD score is indicative of greater scanpath similarity.

B Intent-Aware Scanpaths

As described in section §4.3, the generator conditioned on the downstream natural language task yields *intent-aware* scanpaths. Augmenting NLP models with these scanpaths leads to higher performance gains. Here, we provide more details on *intent-aware* scanpath generation. Please refer to figures 5 and 6 on the following page. Saliency corresponding to intent-aware scanpaths are shown in Fig. 6.

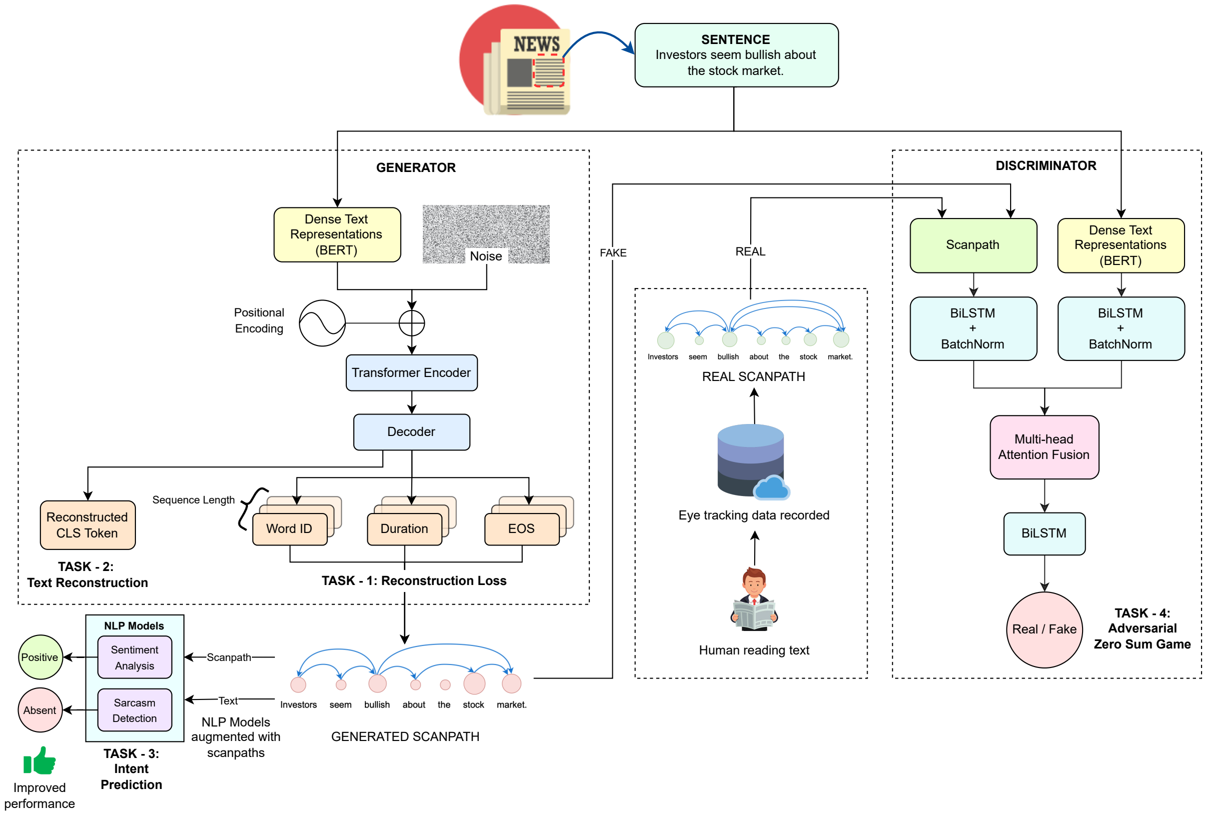


Figure 5: The architecture of the proposed Intent-Aware **ScanTextGAN** model. The model consists of a conditional generator and a discriminator playing a zero-sum game. Two cognitively inspired losses train the generator: scanpath (Task-1) and text (Task-2) reconstruction, a loss from the downstream intent of the natural language task (Task-3), and finally, the loss from the adversarial zero-sum game (Task-4). Variations of scanpaths are generated based on the downstream natural language task.

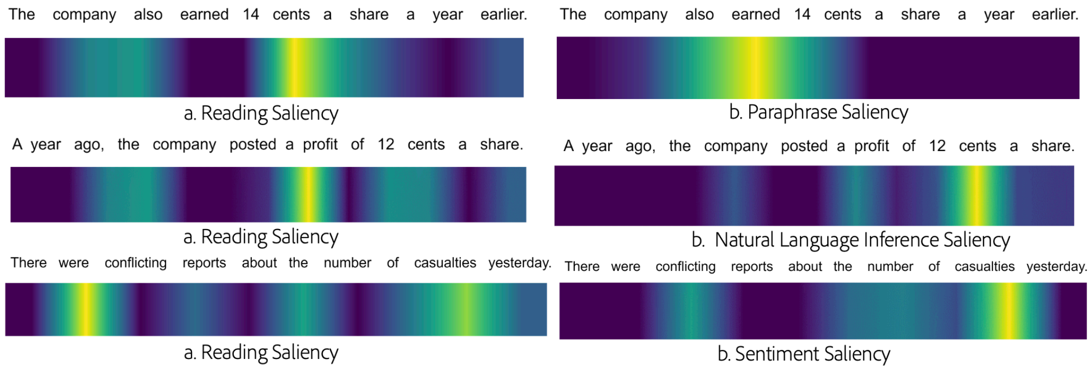


Figure 6: Saliency samples generated by conditioning scanpath generation on different downstream natural language tasks. It can be observed that the conditioned saliency pays much more attention to words important for that downstream task.

Memory-efficient Temporal Moment Localization in Long Videos

Cristian Rodriguez-Opazo[1*], Edison Marrese-Taylor[2,3*]
Basura Fernando[4], Hiroya Takamura[2], Qi Wu[1]
[1] Australian Institute for Machine Learning
[2] National Institute of Advanced Industrial Science and Technology
[3] The University of Tokyo, [4] A* STAR, Singapore
{edison.marrese, takamura.hiroya}@aist.go.jp
{cristian.rodriguezopazo, qi.wu01}@adelaide.edu.au
Fernando_Basura@ihpc.a-star.edu.sg

Abstract

Temporal Moment Localization is a challenging multimodal task which aims to identify the start and end timestamps of a moment of interest in an input untrimmed video, given a query in natural language. Solving this task correctly requires understanding the temporal relationships in the entire input video, but processing such long inputs and reasoning about them is memory and computationally expensive. In light of this issue, we propose Stochastic Bucket-wise Feature Sampling (SBFS), a stochastic sampling module that allows methods to process long videos at a constant memory footprint. We further combine SBFS with a new consistency loss to propose LocFormer, a Transformer-based model that can process videos as long as 18 minutes. We test our proposals on relevant benchmark datasets, showing that not only can LocFormer achieve excellent results, but also that our sampling is more effective than competing counterparts. Concretely, SBFS consistently improves the performance of prior work, by up to 3.13% in the mean temporal IoU, leading to a new state-of-the-art performance on Charades-STA and YouCookII, while also obtaining up to 12.8x speed-up at testing time and reducing memory requirements by up to 5x.

1 Introduction

Processing long untrimmed videos for understanding and reasoning is a computationally expensive task that not only demands a smart approach that can capture global and local interaction, but also requires allocating thousands of frames in memory (Wu et al., 2022). Previous work so far has mostly focused on parsing the visual content in independent snapshots of a video, which limits the approaches in terms of modeling long-term dependency between events in the input.

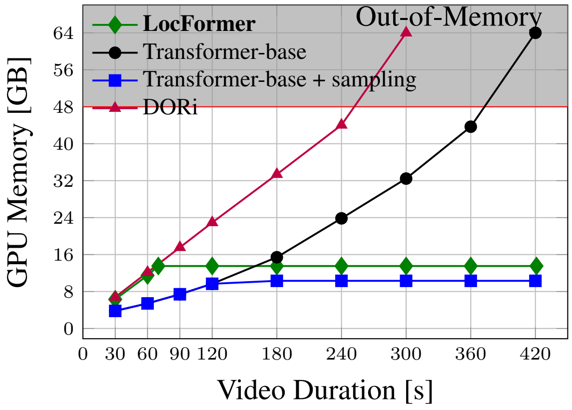


Figure 1: Empirical upper-bound memory consumption of LocFormer, Transformer-base, with and without enabling our proposed sampling technique (SBFS), and DORi (Rodriguez-Opazo et al., 2021). Memory usage (y-axis) is computed on an NVIDIA RTX-8000 GPU, using a batch size of 32 and assuming all sequences have maximum length for the given video duration.

In this paper, we particularly look at the task of temporal moment localization, which aims to identify the start and end timestamps of *a moment of interest* in an input untrimmed video given the query in natural language (Richard et al., 2018; Lin et al., 2017; Escorcia et al., 2016; Chao et al., 2018; Gao et al., 2017b; Xu et al., 2019). Recent approaches have aimed at directly predicting the starting and ending temporal locations, or regressing them from the input video, moving away from the propose-and-rank based approaches (Yuan et al., 2019c; Ghosh et al., 2019; Rodriguez-Opazo et al., 2020). Although these models are more efficient and can potentially capture the influence between different events in a video, they still require a considerable amount of computation and/or memory since they need to process the whole input video at once.

Recent improvements in both Natural Language Processing (NLP) and Computer Vision (CV) tasks can be attributed to the Transformer (Vaswani et al., 2017) model. Despite their success, one main draw-

---

[*] These authors contributed equally to this work

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1909–1924
May 2-6, 2023 ©2023 Association for Computational Linguistics

back of these models is their computational cost, with memory usage ballooning as model sizes increase to attain better performance. This issue is exemplified by many existing developments in Transformer models for video understanding only being capable of processing short inputs at a time due to memory constraints. For example, TimeSformer-large (Bertasius et al., 2021) can only process inputs that are 24 seconds long[1], ViVIT (Arnab et al., 2021) and X-ViT (Bulat et al., 2021) can receive inputs that are up to 32 frames long, while other approaches like MERLOT (Zellers et al., 2021) and ClipBERT (Lei et al., 2021) specifically sample a single or a few frames from the whole video.

In light of this issue, we present Stochastic Bucket-wise Feature Sampling (SBFS). During training, SBFS first splits the input feature sequence into a fixed number of sections (buckets) and then selects a single feature per bucket using a stochastic approach. At testing time, our module selects the sufficient statistic of each bucket, which allows us to obtain a feature sample-set that is representative of the video content for the task at hand while keeping the memory footprint constant

To show the effectiveness of our sampling technique, we combine SBFS with a broad selection of models from previous work, and also use it to propose LOCFORMER, a Transformer-based model that operates at a constant memory footprint regardless of the input length, as shown in Figure 1. We conduct experiments on three challenging datasets, Charades-STA (Gao et al., 2017a), ActivityNet Captions (Caba Heilbron et al., 2015; Krishna et al., 2017) and YouCookII (Zhou et al., 2018b,a). We show that by applying SBFS in previous work we improve their performance in all considered datasets, leading to a new state-of-the-art on Charades-STA and YouCookII, while reducing the memory requirements by up to 5x. Furthermore, LOCFORMER is able to obtain state-of-the-art performance in the latter, and competitive results elsewhere.

We believe our results highlight the importance of sampling techniques as a valid mechanism to obtain better coverage of long input videos while keeping memory usage under budget. This ultimately provides a concrete direction for further research on tasks where it is necessary to cover long untrimmed videos, which include but are not limited to video grounding. We release our code and data[2] to encourage future research in this area.

2 Related Work

Sampling To the best of our knowledge, the earliest example of a sampling technique that is similar to ours is the work of Nakagawa and Nakanishi (1988), who proposed a stochastic version of dynamic time warping for speech recognition of the Japanese language in the late 80s. This idea was further extended by Suryanto et al. (2016) in the context of motion recognition, where a randomized version of dynamic time warping for this task was introduced. We also find several models that utilize sampling techniques for action recognition in videos, in this case to specifically select salient clips, such as SCSampler (Korbar et al., 2019) and MGSampler (Zhi et al., 2021), which were later adopted by models like MVFNet (Wu et al., 2021). Also, Adaframe (Wu et al., 2019b) recently proposed a framework that adaptively selects relevant frames on a per-input basis for fast video recognition. Fayyaz et al. (2021) introduced differentiable parameter-free Adaptive Token Sampling (ATS), which uses a scoring to adaptively sample significant tokens and can be plugged into any existing vision transformer architecture. Finally, Wu et al. (2019a), recently proposed to process short video segments at a time also augmenting the models with a long-term bank from where global features can be sampled. This ultimately enables them to process longer overall inputs without losing global information. While their motivation is similar to ours, we note that their proposed sampling technique is not stochastic and requires learning. Moreover, our results show that performance improvements are possible without access to global context, making our contributions fundamentally different in this sense.

Temporal Action and Moment Localization The goal of *Temporal Action Localization* is to solve the problem of recognizing and determining temporal boundaries of action instances in videos, with extensive previous work devoted to it (Shou et al., 2016; Gu et al., 2018; Girdhar et al., 2019). Given its limitations —restricted to a pre-defined list of labels— the task of language-driven *temporal moment localization* was introduced as a generalization (Gao et al., 2017a; Hendricks et al., 2017). In this task, the goal is to determine the start and

---

[1] TimeSformer-base/large reads 8 frames at 32/4 fps respectively.

[2] https://github.com/crodriguezo/locformer

end times of the video segment that best corresponds to a given natural language query. As this requires the model to extract useful information from the textual semantics in the query in order to identify the moment, this task is also usually regarded as video grounding. Early approaches that tackled temporal moment localization, including (Liu et al., 2018) and Ge et al. (2019), were mainly based on the generation of candidate clips which could later be ranked. Soon after, Chen et al. (2018), Chen and Jiang (2019), and Xu et al. (2019) focused mainly on reducing the number of proposals by producing query-guided or query-dependent approaches. Recently, Zhang et al. (2021a) also adopted a Transformer-based model for this setting, being the most relevant to our work.

The extensive computation of enumerating candidates in the above-mentioned proposal-based methods led to the development of methods that can directly output the temporal coordinates of the segment, namely, proposal-free approaches. In this context, Ghosh et al. (2019) first focused directly on predicting the start and end frames using regressions, and soon after Rodriguez-Opazo et al. (2020) improved results by modelling label uncertainty. While Mun Mun et al. (2020) and Zeng et al. (2020) later proposed more sophisticated modality matching strategies, some more recent approaches have focused on better contextualizing I3D (Carreira and Zisserman, 2017) video features by proposing other model variations (Li et al., 2021). More recently, Liu et al. (2021), CPNet (Li et al., 2021), VSLNet (Zhang et al., 2020a) have pushed performance further up. Finally, models like DORi (Rodriguez-Opazo et al., 2021), which also incorporates spatial features and CPN (Zhao et al., 2021) have proposed ad-hoc graph-based approaches. Compared to these models, our contributions go in a different direction which can be complementary because of their large memory consumption, as shown in Section 5.4.

3 Stochastic Bucket-wise Feature Sampling

Let G be a sequence of input video features extracted by a video encoding function $F_V(V)$. We are interested in developing a module to limit the memory budget of a given model when dealing with long video inputs. In order to do this, we limit the overall memory budget of the model by shortening the sequence of video features fed into the localization module. We do this in practice by proposing a technique that we call Stochastic Bucket-wise Feature Sampling (SBFS), which returns a sequence of length at most b derived from G, as follows.

$$\text{SBFS}(G; b) = \begin{cases} \{\boldsymbol{g}_i\}_{i=1}^{n} & \text{if } n \leq b \\ \{\boldsymbol{g}_{f(k)}\}_{k=1}^{m(n,b)} & \text{if } n > b \end{cases} \quad (1)$$

In Equation 1, $m(n, b)$ characterizes the number of buckets to be allocated to host video features, and is defined as $m(n, b) = \lfloor \frac{n}{\lceil n/b \rceil} \rfloor \leq b$, where $\lfloor\rfloor$, $\lceil\rceil$ are the floor and ceiling operators, respectively. The index $f(k)$ is sampled according to a uniform distribution over the indices of the features in the bucket, as Equation 2 shows, below.

$$f(k) \sim \mathcal{U}_{\lceil n/b \rceil \cdot (k-1)}^{\lceil n/b \rceil \cdot k} \quad (2)$$

More intuitively, we create a fixed number of buckets and allocate features to each by equally distributing them into the buckets. During training, we randomly sample a single feature for each bucket, following a uniform distribution, effectively reducing the input sequence length to at most b, the number of buckets. When doing this, we also accordingly convert the original set of labels $\tau^s, \tau^e \in [1, \dots, n]$ into $\bar{\tau}^s, \bar{\tau}^e \in [1, \dots, b]$. For simplicity, without loss of generality, for the rest of the paper we will assume the sampling module always returns total of b video features.

It is possible to prove that any sampled video feature sequence $\tilde{G}$ obtained using SBFS contains sufficient statistics of G, which allows us to train our models on very long videos with adequate guarantees. From this, it also follows that although SBFS can also be applied during inference, it is better to decouple the model from this stochastic component and instead utilize the max pooling operator over the features of the bucket at inference time. This gives models increased stability when predicting, without sacrificing performance, as we will show in Section 5.3. We refer the reader to Section A of our supplementary material for the full details on our theoretical analysis of SBFS.

4 LOCFORMER

In what follows, we assume that a given video $V \in \mathcal{V}$ can be characterized as a sequence of frames such that $V = \{v_t\}$ with $t = 1, \dots, l$. Each video in $\mathcal{V}$ is annotated with a natural language passage $S \in \mathcal{S}$ where S is a sequence of

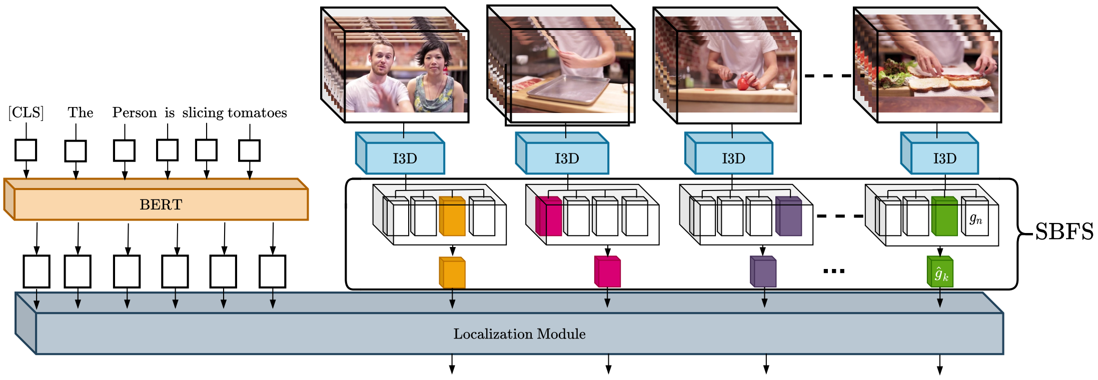


Figure 2: Our method uses a sampling technique that divides the video into a fixed number of buckets. Then, we uniformly sample a single I3D feature from each bucket, which is then fed into the localization module, a multi-modal Transformer model that receives video features and language features obtained from BERT.

tokens $S = \{s_j\}$ with $j = 1, \ldots, m$, which describes what is happening in a certain period of time. Formally, this interval is defined by t^s and t^e, the starting and ending points of the annotations in time, respectively. Although in the data a given video may be annotated with more than one single moment, and one natural language description may be associated to multiple moments, in this work we assume each derived case as an independent, separate training example.

Our model is trained to predict the most likely temporal localization of the contents of a given input query S in terms of its start and end positions $t^{s\star}$ and $t^{e\star}$ in the video. We apply the mapping $\tau = (t \cdot n \cdot \text{fps})/l$ to transform frame/feature index to time, converting t^s and t^e into τ^s and τ^e, which correspond to specific integer feature positions such that $\tau^s, \tau^e \in [1, \ldots, n]$.

LOCFORMER follows the Transformer architecture (Vaswani et al., 2017), which has been recently extended to multi-modal scenarios as in UNITER (Chen et al., 2020) in the context of vision-and-language, and Recurrent VLN-BERT (Hong et al., 2021) for vision-and-language navigation. Our model operates on sequences of tokens $\{s_j\}$ and a video $\{v_t\}$ characterized as a sequence of frames, as specified earlier. The overall architecture is composed by three main modules, as shown in Figure 2. (1) The Video Encoding Module, which is in charge of mapping video frames to vectors, and obtaining a sample that is representative of the video contents using SBFS, (2) the Text Encoding Module, a Transformer model with dimension d_m in charge of extracting useful representations from the natural language query, and (3) the Localization Module, a multi-modal Transformer, also with hidden dimension d_m, which receives both textual and video features from the previous modules and is in charge of estimating τ^s and τ^e. In the following subsections, we give details about each component and how they interact.

Video Encoding Module with SBFS Our video encoding module is in charge of mapping the l input video frames into a sequence of video features $G = \{\boldsymbol{g}_i \in \mathbb{R}^{d_v}\}, i = 1, \ldots, n$. At the core of this module lies SBFS, which we use to select a subset of the features that are representative of the contents of the video. Thus, the sampled video feature sequence $\tilde{G}$ will later be fed into the localization module.

Text encoding module Sentences are processed using the BERT tokenizer, which also prepends the special `CLS` token, and adds the `SEP` marker at the end. Each token is mapped to learned embeddings of dimension d_m and summed with learned positional encodings of the same size. These vectors are passed through $\bar{L}$ encoder transformer blocks with $\bar{M}$ attention heads, to produce final text representations $[\bar{\boldsymbol{h}}_0, \ldots, \bar{\boldsymbol{h}}_m]$.

Localization module A Transformer model that receives both textual and video features, previously obtained by the respective modules. For the former, we directly input $\bar{\boldsymbol{h}}_0, \ldots, \bar{\boldsymbol{h}}_m$, while for the latter we first project $\tilde{G} = [\tilde{\boldsymbol{g}}_1, \ldots, \tilde{\boldsymbol{g}}_b]$ into the hidden dimension using a trainable linear layer and further combine this with a set of learned positional encodings. These two encoded vector sequences are concatenated lengthwise and passed through L encoder blocks with M attention heads, to produce $[\boldsymbol{h}_0, \ldots, \boldsymbol{h}_{m+b}]$.

From these vectors, we select $[\boldsymbol{h}_{m+1}, \ldots, \boldsymbol{h}_{m+b}]$ and utilize the same localization function proposed by Rodriguez-Opazo et al. (2020) as the main training signal, namely, feature-level soft classification task on the time dimension. Concretely, two different MLP layers produce scores of each position being the start/end of the location, which are passed through a softmax activation to obtain $\hat{\boldsymbol{\tau}}^s, \hat{\boldsymbol{\tau}}^e \in \mathbb{R}^b$, which are compared to soft-labels using the Kullback-Leibler divergence ($\mathcal{L}_{KL}$).

In order to guide the model to utilize the information in the relevant section of the video, we encourage the attention heads of this module to put more weight into the target video portions during training, adapting the approach proposed by Rodriguez-Opazo et al. (2020) as shown below.

$$\mathcal{L}_{att} = -\sum_{l=1}^{L}\sum_{m=1}^{M}(\mathbf{1} - \boldsymbol{x} \otimes \boldsymbol{x}) * \log(\mathbf{1} - \boldsymbol{A}^{l,m}) \quad (3)$$

In Equation 3, $\boldsymbol{A}^{(l,m)}$ is the attention matrix of the l-th layer and m-th attention head of the localization module and $\boldsymbol{x} \in \mathbb{R}^{m+b}$ is a vector that denotes which areas of the output sequence will be subject to our guiding signal, and is defined as $\boldsymbol{x} = [\mathbf{1}_m; \boldsymbol{\delta}_{\tau^s \leq i \leq \tau^e}]$, where ; denotes concatenation, δ is the Kronecker delta returning 1 when i is inside the range of τ, and $\mathbf{1}_k$ denotes a vector of ones of size k.

Since our proposed localization loss is not sensitive to the order of the predictions of the starting and ending locations because there is no conditioning on the time in the model portions that generate them, in this paper we additionally propose to induce the model to respect the start-end order, taking a probabilistic approach. Concretely, we push the expected location of the start of the segment (S) to be before the expected location of the ending (E) location, which is equivalent to requiring $\mathbb{E}(E) - \mathbb{E}(S) > 0$. Replacing the values of the expectations, we obtain the following.

$$\mathbb{E}(E) - \mathbb{E}(S) = \sum_{i=1}^{b} \hat{\tau}_i^e i - \sum_{i=1}^{b} \hat{\tau}_i^s i = \sum_{i=1}^{b} i(\hat{\tau}_i^e - \hat{\tau}_i^s) \quad (4)$$

In Equation 4 above, $\hat{\tau}_i^s$ and $\hat{\tau}_i^e$ are integers that denote the predicted probability value of the starting and ending localizations at position i. Based on this derivation, we formally implement our loss by minimizing the negative difference of the expected values as shown in Equation 5, below.

$$\mathcal{L}_{se} = \min(0, \sum_{i=1}^{b} i(\hat{\tau}_i^s - \hat{\tau}_i^e)) \quad (5)$$

Finally, our model is trained with the direct summation of the three losses introduced earlier, such that $\mathcal{L} = \mathcal{L}_{KL} + \mathcal{L}_{att} + \mathcal{L}_{se}$.

5 Experiments

5.1 Datasets

To evaluate our proposed approach, we work with three widely-utilized and challenging datasets.

Charades-STA Built upon the Charades dataset (Sigurdsson et al., 2016), which provides time-based annotations using a pre-defined set of activity classes, and general video descriptions. We use the predefined train and test sets, containing 12,408 and 3,720 moment-query pairs respectively. Videos are 31 seconds long on average and a maximum duration of 194 seconds, with 2.4 moments on average, each being 8.2 seconds long on average.

ActivityNet Captions Introduced by Krishna et al. (2017), this dataset originally constructed for dense video captioning, consists of 20k YouTube videos with an average length of 120 seconds and a maximum duration of 755 seconds. The videos contain 3.65 temporally localized time intervals and sentence descriptions on average, where the average length of the descriptions is 13.48 words. Following the previous methods, we report the performance on the combined validation sets.

YouCookII Consists of 2,000 long untrimmed videos from 89 cooking recipes obtained from YouTube by Zhou et al. (2018b). Each step for cooking these dishes was annotated with temporal boundaries and aligned with the corresponding section of the recipe. The average video length is 316 seconds and a maximum duration of 755 seconds. In terms of relevant moment segments, each video has 7.73 moments on average, with each segment being 19.63 seconds long on average.

5.2 Implementation Details

For our experiments, we consider an off-line video encoding function $F_V(V)$, following previous work (Ghosh et al., 2019; Rodriguez-Opazo et al., 2020; Rodriguez-Opazo et al., 2021; Wang et al., 2020; Yuan et al., 2019a,b; Zhang et al., 2019). Concretely, we first pre-process the videos by extracting features of size 1024 using I3D with average pooling, taking as input the raw frames of dimension 256 × 256, at 25fps. We use the pre-trained model trained on Kinetics for ActivityNet and YouCookII released by Carreira and Zisserman (2017). For Charades-STA, we use the pre-trained model trained on Charades. For the natural language input, we use the BERT-base-uncased tokenizer and keep the parameters of the Text Encoder fixed. Models are trained using Adam (Kingma and Ba, 2014). Evaluation is based on two widely used metrics proposed by Gao et al. (2017a), namely the Recall at various thresholds of the temporal Intersection over Union (tIoU or $R@\alpha$) measuring the percentage of predictions that have tIoU with ground truth larger than certain α, and the mean averaged tIoU (mIoU). We use three α thresholds 0.3, 0.5 and 0.7.

5.3 Ablation Studies

We begin by performing an empirical study of our proposed stochastic sampling technique, comparing it to several alternatives. We specifically consider the following sampling approaches.

Random: As a naive baseline, we randomly sample features maintaining the order.

Fixed-rate video down-sampling (FRVS): We experiment with I3D features extracted at a lower frame-rate of 5fps (Ghosh et al., 2019), which can be regarded as a form of low-level down-sampling.

Fixed-rate feature down-sampling (FRFS): We experiment with two fixed-rate down-sampling techniques at the feature level, bucket-level mean-pooling and max-poling.

Dynamic Time Warping (DTW): We perform dynamic time warping between the non-structured video features and the fixed size temporal sequence created using our stochastic sampling technique and max-pooling applied inside each bucket. In this way, we assign features to each bucket that will later be randomly sampled.

Dynamic-rate feature down-sampling (DRFS): We utilize the similarity across features to dynamically create each bucket. While many variations are possible here, we decided to utilize a cosine distance-based heuristic to create the buckets. Please see Section B in the supplementary material for all the details.

SBFS Variations: We experiment with different alternatives for inference. Concretely, we always apply our stochastic sampling during training, and either use it for inference as well (SBFS-all), or replace it with bucket-wise mean pooling (SBFS-mean) or max pooling (SBFS).

Sampling	**Performance**			
	R@0.3	R@0.6	R@0.7	mIoU
Random	09.42	03.35	00.74	0 9.24
FRVS-5fps	37.54	22.68	10.62	23.90
FRFS-mean	45.99	31.01	15.46	29.98
FRFS-max	45.53	30.61	15.55	30.11
DTW	27.58	13.52	04.47	18.22
DRFS	33.48	17.84	06.56	21.57
SBFS-all	46.28	30.04	15.32	30.34
SBFS-mean	46.68	30.61	15.23	30.53
SBFS	**46.76**	**31.33**	**15.81**	**30.92**

Table 1: Performance of LOCFORMER on YoucookII replacing SBFS with alternative sampling techniques.

For the experiments, we combine each of these sampling approaches with the rest of the LOCFORMER architecture, and always use a bucket size of 200. Regarding the data, we use the YouCookII dataset, as it contains videos that can be as long as 18 minutes with queries that use rich language, which should help illustrate the importance of the sampling.

As the results in Table 1 show, the effectiveness of our sampling technique is clear, specially when compared with more naive alternatives like random sampling, or simple mean pooling. We see that low-level down-sampling techniques, that extract fewer frames from the original video, are not effective either. In contrast, the naive version of the max-pooling-based sampling stands out, performing similarly but still below SBFS. To confirm the effectiveness of our approach against FRFS-max and FRFS-mean, we further compared average results of five runs using different random seeds. We obtained baseline performances of $0.3006 \pm 1.6 \times 10^{-3}$ for FRFS-max and $0.3010 \pm 2.9 \times 10^{-3}$ for FRFS-mean. In contrast, SBFS obtained a performance of $0.3060 \pm 2.9 \times 10^{-3}$, which we find is statistically superior to both baselines at 99% confidence. These results help illustrate the importance of the stochastic approach we have taken, which enables us to limit the input to the model while still exposing it to all the training data in the long run, significantly improving its generalization capabilities. Finally, we also note that all the tested sampling alternatives except DTW and

Bucket Size	Performance			
	R@0.3	R@0.6	R@0.7	mIoU
100	35.74	27.41	11.88	29.03
200	**46.76**	**31.33**	15.81	**30.92**
300	46.36	31.07	**15.95**	30.63
400	45.25	30.30	15.58	30.23
500	44.33	28.69	14.18	29.14

Table 2: Impact on performance of LOCFORMER on the YoucookII dataset as parameter b, the bucket size, changes.

DRFS do not utilize information about the features when generating the buckets. It is interesting to see that many of these arguably simpler sampling techniques, including SBFS, outperform data-informed approaches.

We also study the impact of SBFS at different bucket sizes. For these experiments we use the YouCookII dataset, which contains the longest videos on average, and test bucket sizes ranging from 100 to 500. As we can see on Table 2, variations on parameter b have an impact consistent with its expected behavior, with diminishing results as b increases, and a clear performance sweet-spot at $b = 200$ which we adopt for the main experiments in our paper.

Finally, we ablate LOCFORMER component-by-component, to test the effects of each one of our introduced losses, and of SBFS. We also compare LOCFORMER with two highly-competitive Transformer-based model variations: (1) **Transformer-base** a randomly-initialized multi-modal Transformer-base[3], into which we directly feed the text input and the sampled video features, previously embedding them using a learned embedding matrix and a linear projection layer, respectively. Each encoder uses a separate set of positional embeddings, and we also add a type embedding to indicate the model the nature of each vector. After this, the embedded sequences are concatenated lengthwise and passed through the transformer blocks; (2) **BERT-base**, where we initialize our Transformer-base variation with the weights of BERT. In this case, the projection linear layer of the video features and the respective positional encodings are randomly initialized.

Both model variations apply our attention guiding loss $\mathcal{L}_{att}$ to all the attention heads in all the layers. This effectively means that there is no functionality separation inside these models. We note that these baselines are comparable to existing multi-modal transformer models such as VilBERT (Lu et al., 2019) and UNITER (Chen et al., 2020). Experiments are again performed in YouCookII, which contains the longest videos.

As seen in Table 3, our results first highlight the importance of SBFS, enabling all kinds of Transformer-based models we tested to process long untrimmed videos, which would otherwise lead to out-of-memory errors. Regarding the interaction of the attention loss with different model variations, we see that this additional training signal leads to consistent gains for all Transformer-based variations, but that these are larger in the case of LOCFORMER. We surmise this is due to the attention loss potentially interfering with the inductive bias that the baselines require to process the multi-modal inputs, as well as with the already acquired bias in the case of BERT, reflected in certain attention patterns for each head which have been studied and documented by Rogers et al. (2020) among others. This ultimately highlights the importance of separating functionality inside Transformer models for our task, which allows our model to perform better overall.

On top of SBFS, our results also show how each of our proposed component clearly helps increase performance. In particular, we observe that the addition of our attention loss ($\mathcal{L}_{att}$) results in the largest performance improvements, which is consistent with the findings of Rodriguez-Opazo et al. (2020). While the improvements due to $\mathcal{L}_{se}$ are comparatively less substantial, with differences across α bands, we see that combining both losses leads to the best results, showing that the losses complement each other well, as expected.

5.4 Combining SBFS with previous work

We now focus on studying the ability of our sampling technique to be combined with existing models. We do this by incorporating SBFS into three proposal-free models selected from the literature, and testing them on our datasets. We consider ExCL (Ghosh et al., 2019) and TMLGA (Rodriguez-Opazo et al., 2020), which have been extensively studied in the past years, as well as DORi. We utilized our own implementation of ExCL with our I3D features extracted at 25 fps[4],

---

[3] We follow the original notation (Vaswani et al., 2017) using 12 layers and 12 attention heads.

[4] The original implementation used a frame-rate of 5 fps.

Component			Transformer-base				BERT-base				Locformer			
SBFS	$\mathcal{L}_{att}$	$\mathcal{L}_{se}$	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU
			OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM
✓			33.65	20.50	9.05	22.23	37.29	22.51	10.74	24.61	42.33	28.26	12.83	27.49
✓		✓	33.71	19.90	9.08	21.89	39.46	25.60	12.57	26.41	42.84	28.01	13.80	28.04
✓	✓		40.86	25.09	11.51	26.16	41.32	27.38	14.15	27.88	46.39	30.58	15.35	30.57
✓	✓	✓	**41.75**	**26.52**	**13.34**	**27.83**	**42.18**	**28.01**	**14.63**	**28.24**	**46.76**	**31.33**	**15.81**	**30.92**

Table 3: Results of our Transformer ablation study, performed on the YouCookII dataset, where OOM indicates we obtained an out-of-memory error even when using the largest GPU at our disposal.

Method	Charades-STA					ActivityNet					YouCookII				
	Mem.[GB]	R@0.3	R@0.5	R@0.7	mIoU	Mem.[GB]	R@0.3	R@0.5	R@0.7	mIoU	Mem.[GB]	R@0.3	R@0.5	R@0.7	mIoU
ExCL	2.8	62.28	39.74	22.53	42.28	6.4	55.49	39.33	23.04	40.32	6.9	26.58	15.72	8.19	18.99
+ SBFS	**1.6**	**62.74**	**42.04**	**24.57**	**43.05**	**1.8**	**56.42**	**40.37**	**24.70**	**41.13**	**1.8**	**30.96**	**18.64**	**10.05**	**21.76**
TMLGA	2.3	67.53	52.02	33.74	48.22	7.3	51.28	33.04	19.26	37.78	9.5	33.48	20.65	10.94	23.07
+ SBFS	**1.5**	**70.67**	**52.20**	**33.90**	**49.18**	**1.7**	**53.00**	**35.10**	**19.83**	**37.85**	**1.8**	**39.25**	**25.40**	**12.80**	**26.20**
DORi ♣	32.8	72.72	59.65	40.56	53.28	34.7	57.89	41.49	**26.41**	42.78	46.4	43.36	30.47	18.24	30.46
+ SBFS	**23.1**	**72.90**	**59.67**	**40.94**	**53.44**	**24.0**	**58.89**	**42.21**	26.36	**43.02**	**24.2**	**46.74**	**32.19**	**18.33**	**31.69**

Table 4: Results of our experiments combining SBFS with existing work. Except where indicated, experiments were performed using a batch size 32. ♣ indicates experiments performed using a batch size of 4 due to memory constraints.

Method	Training		Inference	
	TMLGA	DORi	TMLGA	DORi
Baseline	3.18	72	0.36	32
+ SBFS	0.72	10	0.08	2.5
Speed-Up	**4.41**	**7.2**	**4.5**	**12.8**

Table 5: Time in minutes for processing 10,337 and 3,492 queries from the YouCookII dataset in training and inference (test split) respectively.

and directly integrated the original implementations of the latter into our code.

As Table 4 shows, SBFS is able to consistently provide performance improvements in all cases, with gains of up to 3.13% in terms of the mean temporal IoU. These improvements lead to new state-of-the-art results on both the Charades-STA and YouCookII datasets. We note that despite not having access to the spatial information that DORi incorporates, the performance of LOCFORMER is very competitive to that of DORi+SBFS in YouCookII, which contains the longest videos.

We further study the impact on training and inference time when adding our sampling module. For this study consider TMLGA and DORi, and again use YoucookII since it contains the longest videos. As shown in Table 5, SBFS consistently leads to shorter training and inference times, with the degree of impact depending on dataset and model. Concretely, on DORi we obtain a speed-up of 7.2x in training and 12.8x for inference, which again illustrates the effectiveness of our approach, especially in the context of larger models. We also notice that the methods converge in a similar number of epochs with or without adding SBFS, meaning that we can significantly reduce the convergence time, as our sampling enables models to go through each example faster.

5.5 Comparison to state-of-the-art models

Finally, we compare the performance of our proposals on the datasets considered against several prior works. We consider models based on different approaches, specifically proposal-based models including CBP (Wang et al., 2020), MS-TAN-2D (Zhang et al., 2021b) and MNM (Wang et al., 2022), as well as TripNet (Hahn et al., 2020), based on reinforcement learning.

In addition to that, we also compare our approach to more recent methods that do not rely on proposals, including ABLR (Yuan et al., 2019c), ExCL, TMLGA and LGVTI (Mun et al., 2020), CPNet (Li et al., 2021), as well as VSLNet (Zhang et al., 2020b) and BCPN (Nawaz et al., 2022), which cast our task as visual question answering, and CPL (Zheng et al., 2022) which also incorporate Transformer-based components. Finally, we also consider CPN (Zhao et al., 2021) and DORi,

Method	Charades-STA				ActivityNet				YouCookII			
	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU
ABLR †	-	24.36	9.00	-	55.67	36.79	-	36.99	-	-	-	-
TripNet	51.33	36.61	14.50	-	48.42	32.19	13.93	-	-	-	-	-
CBP	50.19	36.80	18.87	35.74	54.30	35.76	17.80	36.85	-	-	-	-
ExCL ‡	65.10	44.10	22.60	-	62.10	41.60	23.90	-	26.58	15.72	8.19	18.99
TMLGA	67.53	52.02	33.74	48.22	51.28	33.04	19.26	37.78	33.48	20.65	10.94	23.07
LGVTI	72.96	59.46	35.48	51.38	58.52	41.51	23.07	41.13	-	-	-	-
DORi	72.72	59.65	40.56	53.28	57.89	41.49	26.41	42.78	43.36	30.47	18.24	30.46
VSLNet	70.46	54.19	35.22	50.02	63.16	43.22	26.16	43.19	-	-	-	-
CPNet	-	60.27	38.74	52.00	-	40.56	21.63	40.65	-	-	-	-
CPN	**75.53**	59.77	36.67	53.14	62.81	45.10	28.10	**45.70**	-	-	-	-
BCPN	73.42	**61.77**	**43.91**	-	66.87	44.53	30.11	-	-	-	-	-
MS-2D-TAN	-	60.08	37.39	-	62.09	45.50	28.28	-	-	-	-	-
CPL	66.40	49.24	22.39	-	**82.55**	**55.73**	**31.37**	-	-	-	-	-
MNM	-	47.31	27.28	-	65.05	48.59	29.26	-	-	-	-	-
DORi + SBFS	72.90	59.67	40.94	**53.44**	58.89	42.21	26.36	43.02	46.74	**32.19**	**18.33**	**31.69**
Locformer	71.88	58.52	38.51	51.76	60.61	43.74	27.04	44.05	**46.76**	31.33	15.81	30.92

Table 6: Performance comparison of our approach with existing methods for different tIoU α levels. Underlined and bold results indicate second-best and best performance for each dataset. Values are reported on the validation split of Charades-STA and the ActivityNet Captions. † Results for ABLR are as reported by Chen and Jiang (2019). ‡ The results reported by ExCL for ActivityNet have 3,370 missing videos, and the results on YoucookII were obtained using our own implementation.

two models that contain specific graph-based approaches for the task, with DORi also incorporating spatio-temporal features.

Table 6 summarizes our best results on Charades-STA, ActivityNet Captions and YouCookII datasets, while also comparing the obtained performance to relevant prior work. We can see that overall Locformer is able to offer excellent performance, closing the gap with sophisticated graph-based models like CPN and DORi as well as Transformer-based models like CPL, with new state-of-the-art performance on the Charades-STA dataset. The results also show the effectiveness of our sampling approach when dealing with long untrimmed videos, as we observe that by combining DORi with SBFS we are able to also attain state-of-the art results on the YoucookII dataset.

6 Conclusion

In this paper we have presented Locformer, a Transformer-based model for the task of temporal moment localization which operates at a constant maximum memory footprint regardless of the input length. The success of our model fundamentally relies on our modular design, which allows us to separate functionality, and SBFS, where we split the sequence of input video features into a fixed number of buckets and select a single feature per bucket using a stochastic approach.

Experiments conducted on three challenging datasets show that Locformer obtains competitive results, and show that our sampling technique can improve the performance of prior work on all considered datasets, leading to a new state-of-the-art on YoucookII and Charades-STA. We think these results highlight the importance of sampling techniques as a valid mechanism to obtain better coverage of long input videos while keeping memory usage low.

For future work, we are interested in testing our sampling approach in other relevant tasks in the context of video-and-language, for example video retrieval. We are also interested in extending our approach to address its limitations, for example, using adaptive or iterative sampling to treat different areas of the video with different granularity.

Acknowledgements

This paper is based on results obtained from project JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). For our experiments, computational resources of the AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST) were used.

Limitations

We believe our results show that sampling is a valid mechanism to obtain better coverage of long input videos, while keeping memory usage under budget. However, it is important to stress that in this work we primarily focused on proposal-free models for temporal moment localization and thus have no evidence to suggest such improvements would be observed in other models.

There is also a specific scenario where our sampling could degrade the performance of a given model. This scenario occurs when the span of the query, *a.k.a.* moment, is located completely inside a given bucket, with many additional frames/features in the same bucket. In this case, the best that a model can do is to predict a moment covering the whole bucket, this losing granularity. In practice, this occurs when the ratio between the duration of a given moment and the duration of the video is vanishingly small. We believe this issue can be alleviated in future work by recursively generating and exploring buckets, an issue that we would like to tackle in the future.

Our results are also limited to the features utilized by the models we considered, and we offer no evidence that our technique will generalize to those cases as well. In terms of data, we have only experimented with queries in English. While it would be interesting to experiment with different languages, this is so far a limitation of the datasets that exist.

References

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. 2021. ViViT: A Video Vision Transformer. *arXiv:2103.15691 null*.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. 2021. Is Space-Time Attention All You Need for Video Understanding? *arXiv:2102.05095 [cs]*.

Adrian Bulat, Juan-Manuel Perez-Rua, Swathikiran Sudhakaran, Brais Martinez, and Georgios Tzimiropoulos. 2021. Space-time Mixing Attention for Video Transformer. *arXiv:2106.05968 [cs]*.

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. 2015. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970.

Joao Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*.

Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A. Ross, Jia Deng, and Rahul Sukthankar. 2018. Rethinking the faster R-CNN architecture for temporal action localization. *CVPR*.

Jingyuan Chen, Xinpeng Chen, Lin Ma, Zequn Jie, and Tat-Seng Chua. 2018. Temporally grounding natural sentence in video. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 162–171, Brussels, Belgium. Association for Computational Linguistics.

Shaoxiang Chen and Yu-Gang Jiang. 2019. Semantic proposal for activity localizaiton in videos via sentence query. *AAAI*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: UNiversal Image-TExt Representation Learning. In *ECCV 2020*, Lecture Notes in Computer Science, pages 104–120, Cham. Springer International Publishing.

Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. 2016. DAPs: Deep Action Proposals for Action Understanding. In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 768–784. Springer International Publishing.

Mohsen Fayyaz, Soroush Abbasi Kouhpayegani, Farnoush Rezaei Jafari, Eric Sommerlade, Hamid Reza Vaezi Joze, Hamed Pirsiavash, and Juergen Gall. 2021. Ats: Adaptive token sampling for efficient vision transformers. *arXiv preprint arXiv:2111.15667*.

Ronald A Fisher. 1922. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 222(594-604):309–368.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. 2017a. Tall: Temporal activity localization via language query. In *ICCV*.

Jiyang Gao, Zhenheng Yang, Chen Sun, Kan Chen, and Ram Nevatia. 2017b. TURN TAP: temporal unit regression network for temporal action proposals. *ICCV*.

Runzhou Ge, Jiyang Gao, Kan Chen, and Ram Nevatia. 2019. Mac: Mining activity concepts for language-based temporal localization. In *WACV*.

S. Ghosh, A. Agarwal, Zarana Parekh, and A. Hauptmann. 2019. Excl: Extractive clip localization using natural language descriptions. In *NAACL-HLT*.

Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. 2019. Video Action Transformer Network. In *CVPR*, pages 244–253.

Chunhui Gu, Chen Sun, David A. Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul

Sukthankar, Cordelia Schmid, and Jitendra Malik. 2018. AVA: A Video Dataset of Spatio-Temporally Localized Atomic Visual Actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056.

Meera Hahn, Asim Kadav, James M Rehg, and Hans Peter Graf. 2020. Tripping through time: Efficient localization of activities in videos. *BMVC*.

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing moments in video with natural language. In *ICCV*.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. VLN BERT: A Recurrent Vision-and-Language BERT for Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*.

Bruno Korbar, Du Tran, and Lorenzo Torresani. 2019. Scsampler: Sampling salient clips from video for efficient action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *ICCV*.

Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L. Berg, Mohit Bansal, and Jingjing Liu. 2021. Less Is More: ClipBERT for Video-and-Language Learning via Sparse Sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7331–7341.

Kun Li, Dan Guo, and Meng Wang. 2021. Proposal-Free Video Grounding with Contextual Pyramid Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(3):1902–1910.

Tianwei Lin, Xu Zhao, and Zheng Shou. 2017. Single Shot Temporal Action Detection. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, pages 988–996, New York, NY, USA. ACM. Event-place: Mountain View, California, USA.

Daizong Liu, Xiaoye Qu, Jianfeng Dong, and Pan Zhou. 2021. Adaptive Proposal Generation Network for Temporal Sentence Localization in Videos. *arXiv:2109.06398 [cs]*.

Meng Liu, Xiang Wang, Liqiang Nie, Xiangnan He, Baoquan Chen, and Tat-Seng Chua. 2018. Attentive moment retrieval in videos. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 15–24. ACM.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *Advances in Neural Information Processing Systems 32*, pages 13–23. Curran Associates, Inc.

Jonghwan Mun, Minsu Cho, and Bohyung Han. 2020. Local-Global Video-Text Interactions for Temporal Grounding. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10807–10816, Seattle, WA, USA. IEEE.

Seiichi Nakagawa and Hirobumi Nakanishi. 1988. Speaker-independent english consonant and japanese word recognition by a stochastic dynamic time warping method. *IETE Journal of Research*, 34(1):87–95.

Hafiza Sadia Nawaz, Zhensheng Shi, Yanhai Gan, Amanuel Hirpa, Junyu Dong, and Haiyong Zheng. 2022. Temporal Moment Localization via Natural Language by Utilizing Video Question Answers as a Special Variant and Bypassing NLP for Corpora. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1.

Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. 2018. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2.

Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Basura Fernando, Hongdong Li, and Stephen Gould. 2021. DORi: Discovering Object Relationships for Moment Localization of a Natural Language Query in a Video. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1079–1088.

Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Fatemeh Sadat Saleh, Hongdong Li, and Stephen Gould. 2020. Proposal-free temporal moment localization of a natural-language query in video using guided attention. *WACV*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. 2016. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*.

Chendra Hadi Suryanto, Jing-Hao Xue, and Kazuhiro Fukui. 2016. Randomized time warping for motion recognition. *Image and Vision Computing*, 54:1–11.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jingwen Wang, Lin Ma, and Wenhao Jiang. 2020. Temporally grounding language queries in videos by contextual boundary-aware prediction. *AAAI*.

Zhenzhi Wang, Limin Wang, Tao Wu, Tianhao Li, and Gangshan Wu. 2022. Negative Sample Matters: A Renaissance of Metric Learning for Temporal Grounding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(3):2613–2623.

Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. 2019a. Long-Term Feature Banks for Detailed Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 284–293.

Chao-Yuan Wu, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. 2022. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. *arXiv preprint arXiv:2201.08383*.

Wenhao Wu, Dongliang He, Tianwei Lin, Fu Li, Chuang Gan, and Errui Ding. 2021. Mvfnet: Multi-view fusion network for efficient video recognition. In *AAAI*.

Zuxuan Wu, Caiming Xiong, Chih-Yao Ma, Richard Socher, and Larry S. Davis. 2019b. Adaframe: Adaptive frame selection for fast video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Huijuan Xu, Kun He, L Sigal, S Sclaroff, and K Saenko. 2019. Multilevel language and vision integration for text-to-clip retrieval. In *AAAI*.

Yitian Yuan, Lin Ma, Jingwen Wang, Wei Liu, and Wenwu Zhu. 2019a. Semantic Conditioned Dynamic Modulation for Temporal Sentence Grounding in Videos. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 536–546. Curran Associates, Inc.

Yitian Yuan, Lin Ma, Jingwen Wang, Wei Liu, and Wenwu Zhu. 2019b. Semantic conditioned dynamic modulation for temporal sentence grounding in videos. In *Advances in Neural Information Processing Systems*, pages 534–544.

Yitian Yuan, Tao Mei, and Wenwu Zhu. 2019c. To find where you talk: Temporal sentence localization in video with attention based location regression. *AAAI*.

Rowan Zellers, Ximing Lu, Jack Hessel, Youngjae Yu, Jae Sung Park, Jize Cao, Ali Farhadi, and Yejin Choi. 2021. MERLOT: Multimodal Neural Script Knowledge Models. *arXiv:2106.02636 [cs]*.

Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Mingkui Tan, and Chuang Gan. 2020. Dense Regression Network for Video Grounding. *arXiv:2004.03545 [cs]*.

Da Zhang, Xiyang Dai, Xin Wang, Yuan-Fang Wang, and Larry S Davis. 2019. Man: Moment alignment network for natural language moment retrieval via iterative graph adjustment. *CVPR*.

Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. 2020a. Span-based Localizing Network for Natural Language Video Localization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6543–6554. Association for Computational Linguistics.

Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. 2020b. Span-based Localizing Network for Natural Language Video Localization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6543–6554, Online. Association for Computational Linguistics.

Mingxing Zhang, Yang Yang, Xinghan Chen, Yanli Ji, Xing Xu, Jingjing Li, and Heng Tao Shen. 2021a. Multi-Stage Aggregated Transformer Network for Temporal Language Localization in Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12669–12678.

Songyang Zhang, Houwen Peng, Jianlong Fu, Yijuan Lu, and Jiebo Luo. 2021b. Multi-Scale 2D Temporal Adjacency Networks for Moment Localization with Natural Language. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Yang Zhao, Zhou Zhao, Zhu Zhang, and Zhijie Lin. 2021. Cascaded Prediction Network via Segment Tree for Temporal Video Grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4197–4206.

Minghang Zheng, Yanjie Huang, Qingchao Chen, Yuxin Peng, and Yang Liu. 2022. Weakly Supervised Temporal Sentence Grounding With Gaussian-Based Contrastive Proposal Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15555–15564.

Yuan Zhi, Zhan Tong, Limin Wang, and Gangshan Wu. 2021. Mgsampler: An explainable sampling strategy for video action recognition. *CoRR*, abs/2104.09952.

Luowei Zhou, Nathan Louis, and Jason J Corso. 2018a. Weakly-supervised video object grounding from text by loss weighting and object interaction. In *British Machine Vision Conference*.

Luowei Zhou, Chenliang Xu, and Jason J Corso. 2018b. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence*, pages 7590–7598.

A Theoretical Analysis of SBFS

Let us begin by providing some key definitions that we will be requiring to perform our analysis.

A *statistic* is a function $T = r(X_1, \ldots, X_n)$ of the random sample $X_1, \ldots, X_n$, which carries information of the sampled data, such as the sample mean and sample variance. We say that a statistic satisfies the criterion of sufficiency when no other statistic which can be calculated from the same sample provides any additional information as to the value, of the parameter to be estimated. We can easily find a sufficient statistics by using the *Fisher–Neyman Factorization Theorem.*

Factorization theorem: *given a random sample* $X_1, \ldots, X_n$ *with joint density* $f(x_1, \ldots, x_n|\theta)$ *a statistic* $T = r(X_1, \ldots, X_n)$ *is sufficient if and only if the joint density can be factored as follows:*

$$f(x_1, \ldots, x_n|\theta) = u(x_1, \ldots, x_n)v(r(x_1, \ldots, x_n), \theta)$$

where u *and* v *are non-negative functions. The function* u *can depend on the full random sample* $x_1, \ldots, x_n$ *but not on the unknown parameter* θ. *The function* v *can depend on* θ, *but can depend on the random sample only through the value of* $r(x_1, \ldots, x_n)$.

In our case, let us assume that our bucket contains features X_i that are independent and uniformly distributed on $[0, \theta]$ where θ is unknown. Then, the probability dense function can be written as a product of individual densities since the observations are independent,

$$f(x_1, \ldots, x_n|\theta) = \frac{1}{\theta}\mathbf{1}_{\{0 \le x_1 \le \theta\}} \cdots \frac{1}{\theta}\mathbf{1}_{\{0 \le x_n \le \theta\}}$$

Here $\mathbf{1}(E)$ is an indicator function. It is 1 if the event E holds, and 0 if it does not. Now $x_i \le \theta$ for $i = 1, \ldots, n$ if and only if $\max\{x_1, \ldots, x_n\} \le \theta$. Therefore,

$$f(x_1, \ldots, x_n|\theta) = \frac{1}{\theta^n}\mathbf{1}_{\{0 \le \min\{x_i\}\}}\mathbf{1}_{\{\max\{x_i\} \le \theta\}}$$

Thus, the *factorization theorem* shows that $T = \max\{X_1, \ldots, X_n\}$ is a sufficient statistic since the density function takes the required form, where $u = \mathbf{1}_{\{0 \le \min\{x_i\}\}}$ and $v = \frac{1}{\theta^n}\mathbf{1}_{\{\max\{x_i\} \le \theta\}}$, which is a function that only depends on θ and $T = \max\{x_i\}$.

With this in mind, we can now move on to our analysis. We first note that nature of SBFS is that a single feature is sampled from each bucket with equal probability each time. If we assume features are i.i.d., this implies that the probability of getting a sampled video feature sequence is $P(\tilde{G}) = (b/n)^b$. As this is a very small probability, the number of potential distinct sampled video feature sets ($\tilde{G}$) is exceptionally large.

Fortunately, video features (frames) within a bucket are highly correlated as they originate from neighboring video frames which are generally very similar, and one may make a weak assumption that bucket population in bucket k may not contain sufficiently more information than any sampled feature $\boldsymbol{g}_{f(k)}$ from the bucket population.

In other words, if $\boldsymbol{g}_{f(k)}$ is sufficient statistic of the bucket k, then any sampled video feature sequence $\tilde{G}$ using SBFS contains sufficient statistics of G, and SBFS(G) is the sufficient statistic of video feature population G. Therefore, we can make the following proposition.

Proposition 1 *Any sampled video feature sequence* $\tilde{G}$ *from* SBFS *method is a sufficient statistic of video feature population* G.

The above proposition is very important as it allows us to train any complex model, such as a Transformer, on very long videos using SBFS with adequate guarantees. Next, we also present an interesting insight on how to pool features within a bucket. To do this, let us denote the j-th dimension of the feature vector $\boldsymbol{g}_i$ by g_i^j.

Proposition 2 *If the* j*-th dimension of vectors within the bucket* k *has a uniform population for all* j, *i.e.* $g_{f(k)}^j$ *is uniformly distributed on* $[1, \mu_j]$ *where* μ_j *is unknown, by the Fisher–Neyman factorization theorem (Fisher, 1922), the sufficient statistics of the population within the bucket* k *is given by the max-pooling operator over the bucket features* [5].

Although our SBFS can also be applied during inference, founded by Proposition 2, it is better to decouple the model from this stochastic component and instead utilize max pooling operator over the features of the bucket at inference time. This gives models increased stability when predicting, without sacrificing performance, as we showed in Section 5.3.

B Dynamic-Rate Feature Down-Sampling

In this section, we present details of our Dynamic-Rate Feature Down-Sampling ablation experiments. With this sampling heuristic, our intention was to

---

[5] Please check the supplementary material for details.

create buckets that satisfy the two following conditions. First, we would like each bucket to hold semantically similar features, using similarity on the embedding space as a proxy. Second, we aim to group features in a way such that the number of buckets l that hold the totality of the features in the video is smaller than the desired number of buckets b.

The bucket construction procedure works as follows. For an input feature sequence of length n, we use the cosine distance to compute the semantic similarity between each of the features in the video, and construct the pairwise distance matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$. We then start the process with a single bucket that contains only the first feature x_1, and add features to this bucket starting from x_2. Feature x_2 will be added to the bucket if and only if the cosine-distance $D_{1,2} < th$, where th is a threshold parameter, and otherwise a new bucket is started and the process is repeated until all the features have been processed. Once this is done, we evaluate the number of buckets l that were created, and if $l > b$, then we reduce the threshold th by a small margin (0.01) and generate all the buckets again. This process is repeated until the $l \leq b$ condition is satisfied. Please see Algorithm 1 below, for additional details.

Algorithm 1 Dynamic-rate Feature down-sampling using cosine similarity Algorithm

```
$\boldsymbol{D} = 1 -$ pairwise_distances($V$)
$th = 1.0$
$flag = True$
index_sample = { }
while $flag$ do
    indx $= 0$
    $st = 0$
    $ed = 1$
    for $ed \leftarrow st$ to len($V$) do
        $s = \boldsymbol{D}_{st,ed}$
        if $s < th$ then
            samples_in_bucket = []
            for $i \leftarrow st$ to $ed$ do
                samples_in_bucket.append(i)
            end for
            index_sample[indx] = samples_in_bucket
            $st = ed$
            indx = indx + 1
        end if
        if indx <= bucket_size then:
            flag = False
        else
            $th = th - 0.01$
            index_sample = { }
        end if
    end for
end while
```

C Qualitative Results

In this section, we present qualitative results of our method for each one of datasets we use for evaluation. Ground truth (GT) and predictions in Figures 3, 4 and 5 are in seconds.

As seen on Figure 3, in the case of ActivityNet Caption, our method is able to localize the query *The man continue to rub the board using his polishing tools* with a high temporal intersection over union (IoU) of 80.41%. Though not visible in the figure, we also note that the end of this video is full of black frames and information about the creator, for example, webpage and logos. This exemplifies how ground truth annotations can be inaccurate, and how our model can adequately deal with these issues.

Figure 4 presents qualitative results for YookCookII dataset. In this case, we specifically present our predictions on one of the longest videos in the dataset, with a duration of 11 minutes and 46 seconds, and where the natural language query is *slice up the ginger finely*. As seen, our method obtains an impressive performance considering that the moment of interest lasts only 20 seconds. This figure also serves to exemplify one of the limitations of the bucketing approach we take. It is possible to see that the predictions of our model, though precise overall, add 0.96 seconds to both the start and end locations. This is a result of the maximum granularity given by the buckets and features in our system.

Finally, Figure 5 shows an example of the predictions of our model on the Charades-STA dataset. In this case, we also choose one of the longest videos in the data, with a duration of approximately 1 minute. For the query *person walks into room holding a bag*, our method obtains a good performance of 95.70% of temporal IoU.

D Experiment Details

- Total Parameters:
    - ExCL: 6.9M parameters.

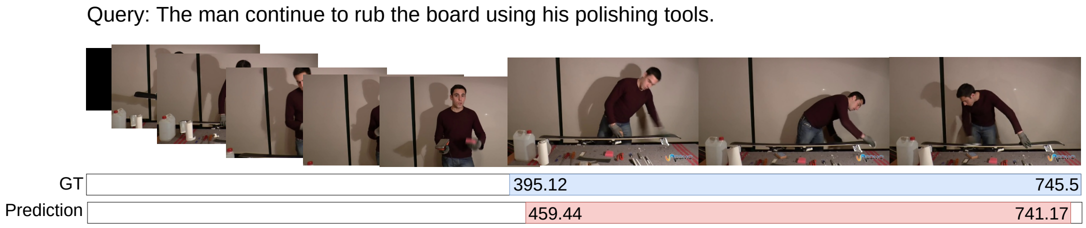


Figure 3: Results of our method on the ActivityNet dataset in a very long video of 12 minutes and 25 seconds (745.5 seconds). Our method can localize the query *The man continue to rub the board using his polishing tools* with a temporal IoU of 80.41%

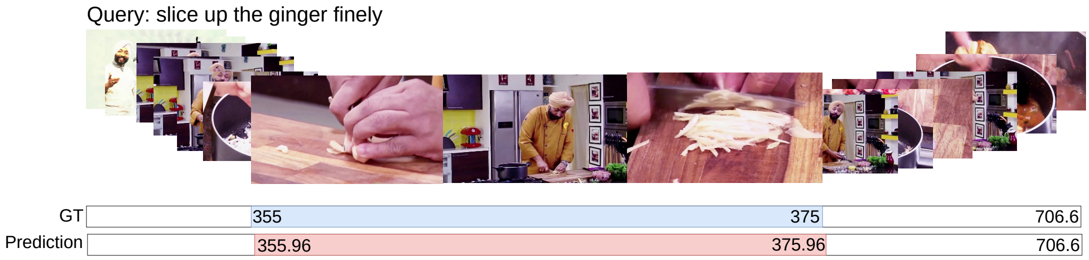


Figure 4: Qualitative result of our method on the YouCooKII dataset in one of the longest videos in the dataset (11 minutes and 46 seconds.) Our method obtains a temporal IoU of 90.83% for the query *slice up the ginger finely*.

  - TMLGA: 4.7M parameters.
  - DORi: 10.4M parameters.
  - LOCFORMER: 86.1M parameters.

- Hyper-parameters: Besides what is explained in the paper, we did not specifically run hyper-parameter exploration. On our early experiments, we tried some learning rate variations, but all of our reported results use 10^{-3} or respect the original setting of the model.

- Hardware requirements: Our experiments were performed on two kinds of GPUs, 16-GB NVIDIA V100 and 48-GB Quadro RTX 8000. The former we access by means of nodes on a large cluster, where each node has four such GPUs. Some experiments, especially on the ActivityNet dataset with DORi or LOCFORMER, were performed using data parallelism to speed up training time, but all of our proposed models can still run on single GPUs. For the experiments with the original DORi model or the baseline Transformer models without SBFS, we recommend using a GPU with at least 32GB of memory.

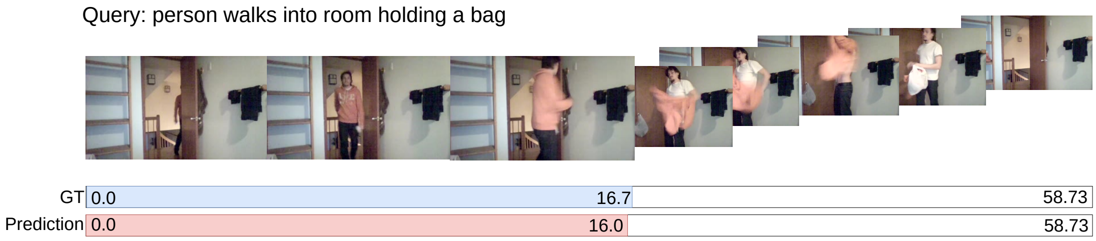


Figure 5: Charades-STA qualitative results on one of the longest videos in the dataset. For the query *person walks into room holding a bag*, our method obtains 95.70% IoU with respect to the ground truth annotations.

Extracting Victim Counts from Text

Mian Zhong **Shehzaad Dhuliawala** **Niklas Stoehr**
Institute for Machine Learning, ETH Zürich
mzhong@ethz.ch shehzaad.dhuliawala@inf.ethz.ch niklas.stoehr@inf.ethz.ch

Abstract

Decision-makers in the humanitarian sector rely on timely and exact information during crisis events. Knowing how many civilians were injured during an earthquake is vital to allocate aids properly. Information about such *victim counts* is often only available within full-text event descriptions from newspapers and other reports. Extracting numbers from text is challenging: numbers have different formats and may require numeric reasoning. This renders purely string matching-based approaches insufficient. As a consequence, fine-grained counts of injured, displaced, or abused victims beyond fatalities are often not extracted and remain unseen. We cast victim count extraction as a question answering (QA) task with a regression or classification objective. We compare regex, dependency parsing, semantic role labeling-based approaches, and advanced text-to-text models. Beyond model accuracy, we analyze extraction reliability and robustness which are key for this sensitive task. In particular, we discuss model calibration and investigate few-shot and out-of-distribution performance. Ultimately, we make a comprehensive recommendation on which model to select for different desiderata and data domains. Our work is among the first to apply numeracy-focused large language models in a real-world use case with a positive impact.[1]

1 Introduction

Timely and accurate information during crisis events is crucial for rescue operations and the allocation of humanitarian aid (Lepuschitz and Stoehr, 2021). However, crisis information is often scarce, subjective, or biased, which renders reported numbers in text extremely important (Hellmeier et al., 2018; Zavarella et al., 2020; Radford, 2021). For instance, the count of injured or missing people provides quantitative information about the catastrophic impact of an earthquake. In this work, we focus on human victims in crisis events, e.g., fatalities in floods, herein referred to as *victim counts*. A reliable estimate of victim counts is helpful during crisis (Darcy and Hofmann, 2003; Kreutzer et al., 2020), and also post-crisis, benefiting research to diversify measures of crisis intensity. As of now, most intensity measures are either limited to event types (Vincent, 1979; Goldstein, 1992), fatality counts (Kalyvas, 2006; Chaudoin et al., 2017) or both (Stoehr et al., 2022). More fine-grained measures such as injured, displaced, or abused victims are not captured in most popular databases and remain unmonitored (Krause, 2013; Cruyff et al., 2017; Cullen et al., 2021).

Many victim counts are reported in full-text form within event descriptions in news media. This makes their systematic collection and analysis technically complex. Manual extraction of victim counts from text is very labor-intensive and does not scale to big data collections (Schrodt and Ulfelder, 2016; Lewis et al., 2016). Computerized approaches such as the event coding software Tabari (Schrodt, 2009) and Petrarch2 (Norris et al., 2017) focus on extracting actor and event types. They rely on lambda calculus and syntactic pattern matching, but disregard mentions of victim counts.

As we will show, parsing-based approaches perform decently well at extracting explicitly reported victim counts. They can identify the mention of the count “5” in “5 people were injured”. However, they are often inadequate when the description *implies* a correct count — for example, from the description that “one logger was shot but survived”, a human reader may infer that *one* person is injured. Since neither a count nor the injury is mentioned explicitly, a parsing-based system may fall short. Another difficulty stems from the fact that the counts can be reported in many, different formats. A reported count may be digit-based or spelled out, define an exact quantity or a range as in “dozens of people were injured”. As a conse-

---

[1] Code is available online at: `https://github.com/mianzg/victim_counts`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1925–1940
May 2-6, 2023

quence, formulating the task of victim count extraction is not an easy endeavor (§3). Most prior work assumes a setting where the count is explicitly mentioned in an event description (Döhling and Leser, 2011; Imran et al., 2013; Rudra et al., 2018; Camilleri et al., 2019). Such settings can be tackled by sequence labeling models that select a relevant span from the given description. However, if the victim count does not appear verbatim, as in the above "one logger" example, models with some form of abstract reasoning capacity may be needed (Roy et al., 2015). Recently, large language models have shown promising results in answering number-focused questions with and without explicit mentions of relevant numbers (Lewkowycz et al., 2022; Nye et al., 2021; Wei et al., 2022; Lefebvre and Stoehr, 2022).

This paper is concerned with studying these different approaches (§4): as baselines, we compare regular expression, dependency parsing, and semantic role labeling. We consider the NT5 (Yang et al., 2021) model as a representative numeracy-enhanced pre-trained language model. We use the representation of this model in a generation, a classification, and a regression setting. We evaluate all models along three dimensions: accuracy (§5), reliability (§6), and robustness (§7). We find that the fine-tuned language model outperforms the baseline models, especially when the victim count extraction requires reasoning. Reliability and robustness are particularly important in high-stake, human-centric tasks such as victim count extraction (Zhang et al., 2020; Kong et al., 2020; Russo et al., 2022b). Model reliability indicates to which extent model behavior can be trusted within decision-making settings (Leibig et al., 2017; Jiang et al., 2021). One dimension of reliability is model calibration which indicates if a model's confidence is aligned well with it making correct predictions (Guo et al., 2017). While calibration has been widely studied for classification, we add to the discussion of calibrated regression (Song et al., 2019) and generation settings (Widmann et al., 2021). Finally, the dimension of robustness describes how stably a model performs. For instance, when the training set is limited or when the test data is out-of-distribution, a less robust model will forfeit more of its predictive performance. To shed light on this dimension, we conduct experiments in few-shot learning and out-of-distribution settings.

We conclude with an application to showcase the extraction of fine-grained and highly specialized types of victim counts. Lastly, we discuss the benefits and drawbacks of the different approaches to assist practitioners in choosing the most suitable task formulation and model.

2 Data

We use publicly available datasets covering natural disasters and armed conflicts, namely: (1) World Atrocities Dataset (WAD) (Schrodt and Ulfelder, 2016), (2) Non-violent and Violent Campaigns and Outcomes 3.0 (NAVCO) (Lewis et al., 2016), and (3) European Media Monitor (EMM) (JRC Science Hub, 2018; Steinberger et al., 2017). For each dataset, we use the event text description and two types of victim counts: the death count and the injury count that we refer to as "WAD death" or "WAD injury". We pre-process the data by removing the samples with missing values (NaN) in the victim counts . For EMM, we only consider samples with a non-zero victim count since "0" is over-represented.

3 Task Formulation

In this section, we discuss some questions and challenges faced in formulating the task of extracting victim counts from event descriptions. We justify some of the choices we make and describe why it is not possible to have a single formulation that fits all needs:

Is the victim count always present in the text? Victim counts can be expressed in various ways in the text. When the count is expressed explicitly in the text, say "5 people were injured", a span extraction model can effectively extract the injury count 5. However, in certain cases, a single explicit number might not be mentioned, and the victim count needs to be logically or algebraically inferred from the text. Consider the description "a 4-year-old girl and her mother were found dead"; a model would need to logically deduce that the victim count of death is 2. To handle this, we not only look at span extraction models but also experiment with models that can understand the text at a deeper level and produce a victim count.

Is the victim count always a single number? Often, in the event description, the victim count is described as a range, such as "at least 330 people died", or in vague terms, like "dozens were injured". Additionally, even within a description, the victim counts for the same event can be varying, possibly because of recording the counts from different sources. This makes extracting a single exact count almost impossible. In such cases, the best a model can do is to output a close estimate of the actual victim count. Another solution would be to provide a range within which the count could lie. For a humanitarian section deciding on the quantity of aid to be deployed, a range might suffice over a single exact count. To account for this, we also look at models that are trained to output a range by classifying the victim counts into a set of binned categories.

4 Models

In §4.1, we introduce baselines models that parse an event description and heuristically extract a victim count. We then specify the model implementation for the different task formulations in §4.2.

4.1 Baseline Models

All baselines extract a victim count by locating the part of the text that could be relevant to victims and finding the nearby victim counts. The locating step requires a pre-defined list of words denoted as *locating list*. For example, to extract death counts, this list would include terms like "kill" and "die".

Regex. Regular expressions (regex) is a rule-based method to extract counts by string pattern matching. The patterns (App. A) are built based on active or passive voice to extract a count closest to phrases in the locating list.

Dependency Parsing. The dependency parsing model collects all possible numeric modifiers and their dependency relationships. Since not every numeric modifier relates to victim counts, e.g., "42-year-old", we construct dependency rules with the locating list to decide if the number is the victim count. For example, one rule is to check if the numeric modifier is for a subject phrase that would reject "42" in the example of "42-year-old". If no numeric modifier is found (e.g., "a journalist was injured"), additional rules use the locating list to return "1" if the rule is satisfied and otherwise return "0".

SRL. Semantic role labeling (SRL) recursively decomposes text input into pairs of predicates and their arguments. We define a list of predicate verbs for death and injury count as the locating list. Then, we iterate over the predicate-argument pairs, check if any predicate from the locating list occurs, and extract the count from its argument if possible. If a predicate exists, the implementation returns the first number as the count if multiple are found and returns "1" if no verbatim number is found. If no such predicate appears, the count is set to "0".

4.2 Task Modeling

We perform victim count extraction using three methods: generation, regression, and classification. As discussed above, each of these approaches caters to the different formulations of our task and can be beneficial in different scenarios. Across these methods, we use the same underlying NT5 model. For clarity, we denote NT5-Gen, NT5-Reg, and NT5-Clf for the corresponding models. The NT5 model (Yang et al., 2021) is a variant of the T5 model (Raffel et al., 2020) with further fine-tuning on numerical tasks. We query the model in a similar fashion to previous works by giving the question and event description in the form: "answer me:`[question]` context:`[passage]`". We discuss how we fine-tune this model for each of our specific methods below.

Generation. For generation, we fine-tune NT5 to decode the victim counts auto-regressively. At inference, we use beam search to generate output. Generation does not guarantee to only generate numeral tokens; therefore, we follow De Cao et al. (2021) to constrain the possible generation tokens in a prefix-conditioned way, such that only number digit tokens $0-9$ and EOS token are allowed at each decoding step.

Regression. For regression, we add two linear layers (with ReLU activation) on the encoder representation to output the numerical victim count. The model is trained to optimize the $\log$ mean-squared error between the true and predicted count.

Classification. We model the task as a classification problem by binning the victim counts into ordinal classes. Similar to regression, the model has a classification head of a linear layer and a softmax layer on top of an encoder initialized with NT5 weights. Our experiments use a 3-class classification by converting the victim counts into three categories: $[0, 3], (3, 10], (10, \infty)$.

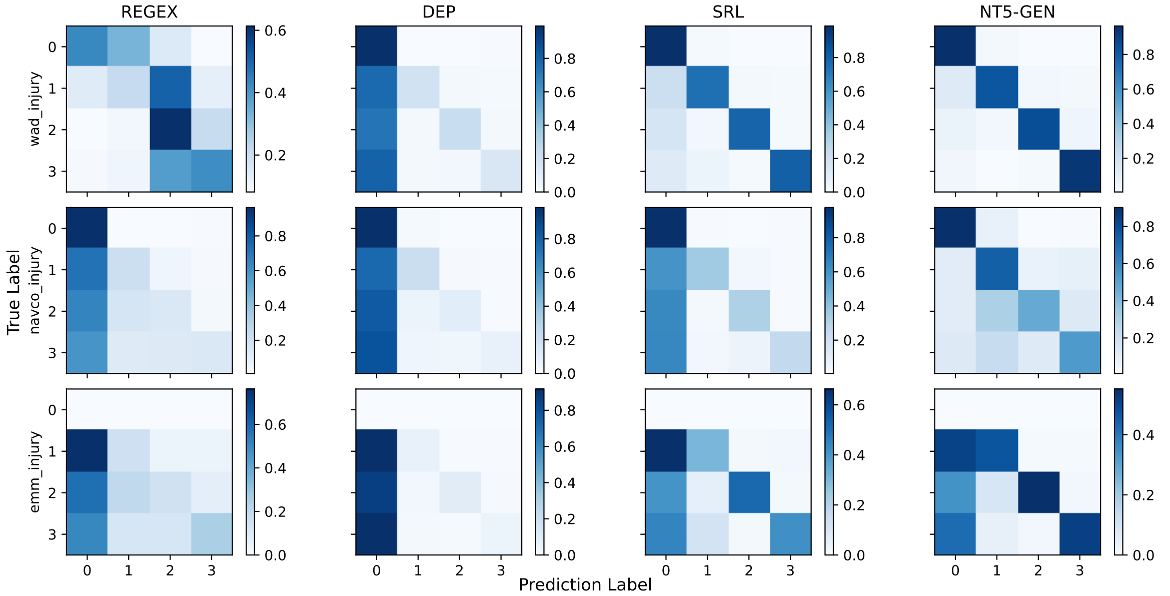


Figure 1: Confusion matrices of the baselines and the fine-tuned NT5-Gen model (columns) of extracting injury counts from different data (rows). We convert the true and prediction victim counts into 4 categories: for any count y, "0" is $y = 0$, "1" is $0 < y \leq 3$, "2" is $3 < y \leq 10$ and "3" is $y > 10$. Values are normalized over true counts. Baselines tend to have low precision on extracting injury counts (dark columns on "0"). SRL and NT5-Gen have comparable accuracy and recall; however, NT5-Gen is slightly better in precision.

5 Accuracy of Counts Extraction

We begin by evaluating the efficacy of our proposed methods for victim count extraction. We examine the model accuracy by comparing baselines and the fine-tuned model with a generation objective (§5.1). We then show the results of using classification and regression formulations (§5.2).

	Exact-Match			**F$_1$**		
	WAD	NAVCO	EMM	WAD	NAVCO	EMM
Regex	0.117	0.264	0.064	0.202	0.318	0.124
Dep	0.226	0.303	0.052	0.355	0.363	0.136
SRL	0.741	0.430	0.313	0.779	0.484	0.361
NT5-Gen	**0.813**	**0.501**	**0.443**	**0.846**	**0.544**	**0.492**

Table 1: *Exact-Match* and F_1 scores of the baseline models and the fine-tuned NT5-Gen on injury counts. The best results are **bolded**. The NT5-Gen model performs better than baselines across all datasets. `DEP` refers to the dependency parsing model and `SRL` refers to the semantic role labeling model.

5.1 Comparing Baselines with NT5-Gen

We compare the accuracy performance of the baseline models and the fine-tuned NT5-Gen model. Tab. 1 shows the results of extracting the injury counts using *Exact-Match* and F_1scores commonly used in related tasks (Yang et al., 2021; Dua et al., 2019). We measure F_1 score on digitized tokens (i.e., "34" → ["3", "4"]). The fine-tuned NT5-Gen model has an accuracy boost up by 7-13% in *Exact-Match* and by 6-13% in F_1 score than the strongest baseline model SRL. The performance of regex and dependency parsing varies heavily across different data, which implies that the regex pattern or dependency relationship may be less helpful in finding the victim counts.

Moreover, we convert the victim counts into four bins, where the bins are selected to have a balanced number of samples in each bin. As an illustration, Fig. 1 shows the confusion matrices on the transformed injury counts. For both victim types, baseline models have a low precision to falsely return "0" too often. Compared with baselines, the NT5-Gen model improves to extract victim counts whose numeric values are large (e.g., $y > 10$).

Qualitative Analysis. We qualitatively examine error samples of the SRL model that the NT5-Gen model extracts correctly. We randomly select 20 error samples for each test set to evaluate and summarize 4 types of errors with examples in Tab. 2. Out of all errors[2], 39.2% belong to diverse lin-

---

[2] There are a few samples where the ground truth might be erroneous. As the event-coding requires more domain expertise within the corresponding social science discipline, we leave the discussion out of this work.

Error Type	Context	Truth	SRL	NT5
`Diverse Expression`	*Six passengers* in a taxi also *had their throats cut*	6	0	6
`Numerical Reasoning`	Herders shot and *killed four people* [...]. Herders then shot and *killed a farmer* at Jokhana [...]	5	4	5
`Number Ambiguity`	*Unidentified gunmen* clash with army	1	0	1
`Number Spelling`	.Twenty-three people were killed [...]	23	1	23

Table 2: Error examples of SRL that the NT5-Gen model is correct on extracting death counts. `Diverse Expression` refers to the string patterns not captured by pre-defined rules. `Numerical Reasoning` shows that the correct count has to be achieved by some mathematical operation over the text. `Number Ambiguity` indicates that a verbatim number is not written but an estimate may be made (with domain expertise). `Number Spelling` refers to problems with number / text format that are typos or the tokenizer parses wrongly (e.g., “twenty-three”→ “twenty”).

guistic expressions on depicting victims, 38.3% contain number ambiguity, 8.3% need numerical reasoning, and 5.8% have spelling issues (for the tokenizer). The NT5-Gen model performs better when the count needs numerical reasoning. Even if the reasoning is not needed, SRL may fail when the linguistic expression to depict victims (e.g., “have throats cut”) is out of the pre-defined locating list (e.g., [“die”, “kill”, “slay”]). These error types are difficult for baseline models to be improved since the patterns cannot be defined beforehand.

5.2 Results on Classification and Regression

We examine the accuracy of the classification and regression formulations by comparing NT5-Clf and NT5-Reg with different initialization weights. To compare, we use T5-SMALL and BERT-BASE-UNCASED pre-trained weights for the encoder. Tab. 3 shows the classification results on NAVCO injury data. Fine-tuning T5-SMALL and NT5 reaches comparable performance; precision and recall scores are similar, but precision is slightly higher. The scatter plots (Fig. 2) show the results of regression using different pre-trained weights with the mean squared error (MSE). For a (log-transformed) victim count larger than 5, using the regression objective seems more conservative in giving small-valued predictions. The numeracy-rich NT5 weights do not particularly improve accuracy for a classification or regression objective, and employing some standard pre-trained weights might be sufficient.

6 Evaluating Reliability

Another important dimension is reliability which we evaluate through the lens of calibration (§6.1). As we approach the task with multiple formulations, calibration analysis is especially needed to understand whether a model is calibrated (§6.2), and how post-hoc calibration techniques may adjust models to be better calibrated (§6.3).

	Accuracy	$\mathbf{F}_1$	Precision	Recall
NT5	0.65	0.60	0.62	0.59
T5	0.65	0.60	0.61	0.59
BERT	0.52	0.23	0.17	0.33

Table 3: Classification results on NAVCO injury data with the NT5-Clf model initialized by different pre-trained weights: NT5, T5-SMALL, and BERT-BASE-UNCASED. F_1 , precision and recall scores are macro.

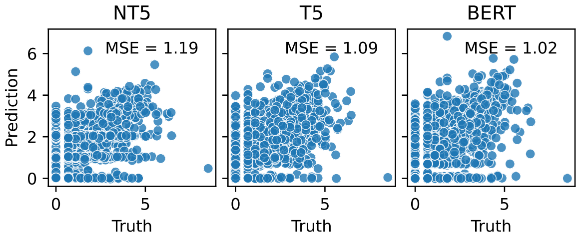


Figure 2: Scatter plots of the fine-tuned NT5-Reg model initialized with different pre-trained weights (NT5, T5-SMALL, and BERT-BASE-UNCASED). The models are trained on log-transformed victim counts.

6.1 Preliminaries: Calibration Metrics

A well-calibrated model ensures that the confidence of the output is well aligned with the chance of the output being accurate. This is a desirable property for our task — consider a model extracts “0” when the text depicts an injured person. A calibrated model would assign very low confidence to the extracted count, which may avoid error propagation to downstream decisions, e.g., medical resource dispatch. We here introduce the expected calibration error (ECE) (Pakdaman Naeini et al., 2015), a standard metric used for classification and is extended for generation decoding (Widmann et al., 2021). For regression, we apply quantile calibration error (Kuleshov et al., 2018).

Given n samples, we create M equal-width bins over the interval $[0, 1]$. ECE takes a weighted average on the differences between the classification accuracy and the mean confidence within each B_m,

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \Big| \text{acc}(B_m) - \text{conf}(B_m) \Big|.$$

The quantile calibration error averages the differences between the empirical frequency $\text{freq}(B_m)$ and the upper bound of B_m (i.e., $\sup(B_m)$), where $\text{freq}(B_m)$ is the fraction of n samples whose quantiles lower or equal to $\sup(B_m)$,

$$\text{RegCE} = \frac{1}{M} \sum_{m=1}^{M} \Big| \text{freq}(B_m) - \sup(B_m) \Big|.$$

The calibration error of generation decoding takes the best b beam search answers, and applies softmax on their scores to represent the confidence. The ECE is then calculated on the best beam search answer similar to classification.

6.2 Calibration Error on Different Models

		Death		Injury	
Data	Model	Orig	Calib.	Orig	Calib.
NAVCO	Clf	0.222	0.044	0.332	0.060
	Reg	0.220	0.097	0.141	0.057
	Gen	0.054	0.040	0.092	0.092
WAD	Clf	0.192	0.055	0.228	0.088
	Reg	0.272	0.107	0.167	0.294
	Gen	0.218	0.221	0.096	0.042
EMM	Clf	0.277	0.098	0.314	0.055
	Reg	0.201	0.189	0.368	0.188
	Gen	0.087	0.092	0.328	0.122

Table 4: Calibration errors of fine-tuned NT5-Clf, NT5-Reg, and NT5-Gen models before (Orig.) and after (Calib.) applying post-hoc calibration. Post-hoc calibration effectively reduces the errors.

We show in Tab. 4 the calibration errors measured on the fine-tuned NT5-Clf, NT5-Reg, and NT5-Gen with different data. Surprisingly, the NT5-Gen model is well-calibrated on most datasets, except for EMM injury: the lowest calibration error is 0.05 on NAVCO death, and the errors on other data range between 0.08 and 0.33. Classification models tend to have large calibration errors (> 0.19). In particular, the error is larger than 0.3 on NAVCO and EMM data to classify injury counts. Regression is also prone to large calibration errors (> 0.15).

Another helpful tool is the reliability diagrams which visualize the calibration errors at different confidence bins. As an illustration, Fig. 3 shows the diagram of the NT5-Clf model fine-tuned on NAVCO injury data, and the diagonal line indicates the perfect calibration. This model is over-confident, and we can observe large gaps when the model confidence is larger than 0.8.

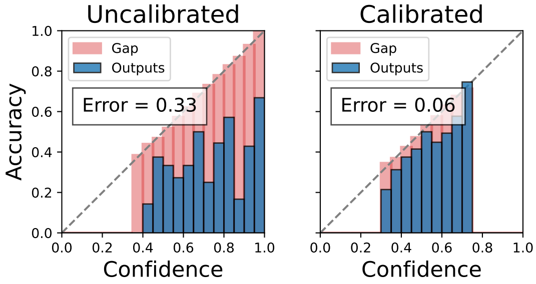


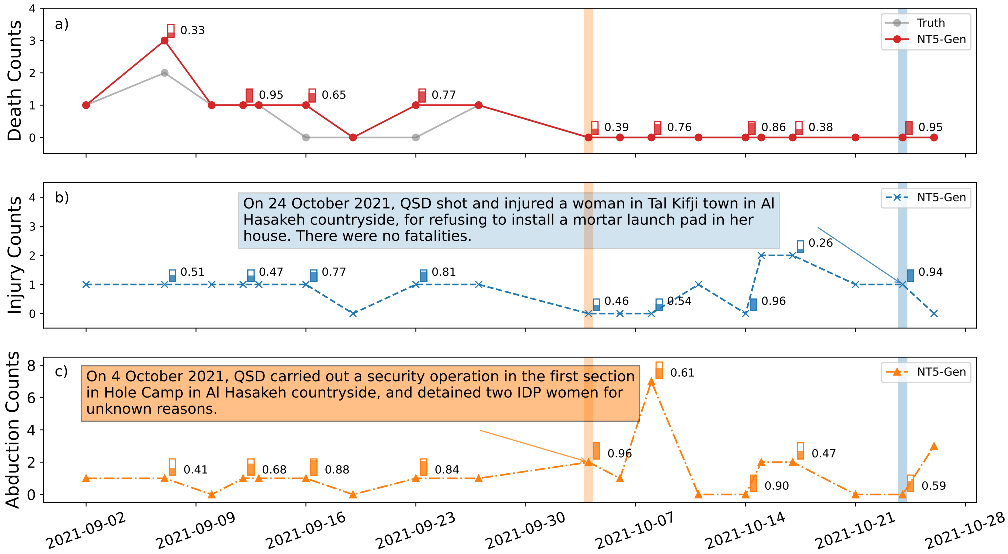
Figure 3: Reliability diagrams compare the calibration error before (left) and after (right) post-hoc calibration of the fine-tuned NT5-Clf model using the NAVCO injury data. This model is prone to large calibration errors (red gaps) in many bins. This is especially true for bins with high model confidence (> 0.8).

6.3 Post-hoc Calibration

Since the models can be over-confident based on the above analysis, we see the necessity to calibrate models for victim count extraction. We use temperature scaling for classification and generation decoding, and isotonic regression for regression. The post-hoc calibrators use development data to minimize negative log-likelihood and are then applied to test sets to measure calibration errors. As a comparison, Fig. 3 (right) shows the calibrated results of the fine-tuned NT5-Clf model on NAVCO injury data. The calibration error (i.e., ECE) reduces from 0.33 to 0.06. The errors of other calibrated models can be found in Tab. 4. In general, when the models have rather a large calibration error (e.g., > 0.3), post-hoc calibration is more helpful and adjusts the models to a better-calibrated level.

7 Evaluating Robustness

Typically, conflict or disaster data is noisy and limited. This is making it challenging to train models



On 24 October 2021, QSD shot and injured a woman in Tal Kifji town in Al Hasakeh countryside, for refusing to install a mortar launch pad in her house. There were no fatalities.

On 4 October 2021, QSD carried out a security operation in the first section in Hole Camp in Al Hasakeh countryside, and detained two IDP women for unknown reasons.

Figure 4: Timeline of victim counts in Syria data from Sept to Nov 2021 as given in the ACLED dataset. We use the NT5-Gen model that is fine-tuned on NAVCO data. Our model can be tested on the extraction of fatality counts which is the only victim count featured in ACLED (Fig. (a)). Beyond fatality counts, it can extract more fine-grained victim types such as (b) injury and (c) abduction counts. Confidence scores are shown for some of the predictions.

on a large-scale, high-quality training set. For this reason, we need robust models that excel in few-shot and out-of-distribution settings.

Reduced Training Size. We fine-tune the NT5-Gen, NT5-Reg, and NT5-Clf models on different-size portions of the training set. Specifically, we use 100%, 50%, 10%,5%, 0.5% and 0% of the training data and as further discussed in App. C.1. As expected, we find that the accuracy of all models drops when using a smaller training set. The NT5-Gen model reveals to be the most robust in keeping the *Exact-Match* metric above 0.6 when being fine-tuned on only 5% of the training data. The calibration error of the fine-tuned NT5-Clf model increases when the training size is reduced, while the fine-tuned NT5-Reg and NT5-Gen models do not follow this trend. In the zero-shot setting, the NT5-Reg and NT5-Gen models reach their largest calibration error. In contrast, the NT5-Clf model reaches its smallest calibration error in the zero-shot setting.

Out-of-distribution (OOD) Setting. We set up synthetic tasks in which a fine-tuned model is confronted with an out-of-distribution setting at test time. For example, we fine-tune a model on WAD death and then repurpose it to classify WAD injury. Then, we evaluate the drop in performance of this “out-of-distribution” model compared to an “in-distribution” model, that has been trained on WAD injury labels directly. We conduct this comparison on different datasets and models.

In App. C.2, we evaluate the NT5-Clf model in a classification formulation and report accuracy. As expected, we find that accuracy decreases in every setting with performance drops between 0.001% and 0.3%. In Fig. 15, we evaluate the NT5-Reg model in a regression setting measured in MSE. We find that the performance decreases in the out-of-distribution settings as evidenced by an average increase of 1.12 in MSE. Finally, in Fig. 16, we turn to an NT5-Gen model in a generative setting. As an evaluation metric, we consider *Exact-Match* and observe a decrease of 0.18 in *Exact-Match* on average.

8 Application: Overlooked Victim Types

Most event datasets feature only one column detailing victim counts. This column typically quantifies fatalities, as they are considered least ambivalent and most important (Kalyvas, 2006; Chaudoin et al., 2017). The Armed Conflict Location & Event Data Project (ACLED) (Raleigh et al., 2010; Raleigh and Kishi, 2019) recently published curated datasets containing violence against healthcare workers, media personnel, and women. Considering the ACLED dataset on Political Violence Targeting Women & Demonstrations Featuring Women, we find that more than 85% of events have *zero* fatalities. This means, many forms of violence remain non-quantified, often those against "marginalized" groups of society.

Using the methods presented in this work, we can extract much more fine-grained victim types such as "injured women" and "abducted women". To this end, we rely on the NT5-Gen model that we fine-tuned on the NAVCO data, without specifically asking for "women". In Fig. 4, we present exemplary two-month time series of events in Syria. We find that our model has a higher recall than precision on the ground truth annotations for fatality counts. This may be desirable since we would like to avoid overlooking true victim counts.

9 Discussion

This work surveys different task formulations of victim count extraction and inspects desiderata like accuracy, reliability, and robustness of different models. We now summarize our findings and conclude which approach performs best under which circumstances (Tab. 5).

Some of the parsing-based approaches have the advantage of requiring no ground truth annotations of the extracted victim counts. This means, there is no need for training, but instead, a manually curated list of patterns and rules has to be assembled. The regex approach, for instance, has minimum requirements regarding hardware, but writing regex patterns is very time-intensive and can be prone to mistakes. Overall, the baseline models shine when it comes to speed, and they perform reasonable when victim counts are explicitly mentioned. Yet they fail at complex reasoning. For instance, when asking for the count of deaths in "one child and four women lost their lives", all baselines mistakenly output "1".

This is where language model-based methods have a competitive edge. The fine-tuned NT5-Gen model has high accuracy both in *Exact-Match* metric and relative error metric. Surprisingly, it is also well-calibrated and relatively robust in the few-shot and out-of-distribution setting. This performance comes at the costs of reduced speed, the requirement of large amounts of training data, and the need for resources like GPUs to be deployed on a large scale.

Comparing classification and regression objectives, we conclude that classification is easier to handle. In most settings, it may be sufficient to extract a range rather than an exact number anyways. In comparison to generation, in classification and regression settings, models show higher calibration errors and require post-hoc calibration to adjust the model confidence.

10 Related Works

This work interfaces with related works from different disciplines to improve the measurement of crisis intensity. It draws inspiration from recent advancements in question answering models with a focus on numbers and math word problems. This includes number-enhanced language models more generally. Our work also connects with model calibration in natural language processing (NLP) more generally.

Measurement of Crisis Intensity. Extracting information about crises has been widely explored using social media data (Temnikova et al., 2015) and newspapers (Keith et al., 2017; Halterman et al., 2021). Most existing measures of crisis intensity focus on counts of event types (Goldstein, 1992; Terechshenko, 2020; Stoehr et al., 2022) or fatality counts (Kalyvas, 2006). Previous work studies friend-enemy relationships (Han et al., 2019; Russo et al., 2022a; Stoehr et al., 2021, 2023) and conflict-indicative changes in word embeddings (Kutuzov et al., 2017).

Numerical Question Answering. Numerical Question Answering pertains to the task of providing numeric answers to questions. An exemplary model is NAQANet (Dua et al., 2019), which extends QANet (Yu et al., 2018) with numerical operations. Neural Module Networks (Gupta et al., 2020) learn and execute a chain of logical learnable and differentiable modules. Some of these modules are specifically targeted at mathematical operations such `find-num` or `count`. Other approaches leverage knowledge graphs (Davidov and Rappoport, 2010; Kotnis and García-Durán, 2019) or graph neural networks (Chen et al., 2020). Thawani et al. (2021) provides a detailed overview over methods for representing and modeling numbers in NLP.

Number-enhanced Language Models. More recent work in number question answering relies on pre-trained large language models. GenBERT (Geva et al., 2020) improves numeric reasoning abilities by including a large amount of

	Accuracy Optimization			Reliability	Robustness		Hardware
	Absolute Error	Relative Error	String Match		Need Training	Stable in OOD	
REGEX	High	Medium	Medium	N/A	No	N/A	Low
DEP	Medium	High	Low-Medium	N/A	No	N/A	Low
SRL	Low-Medium	Medium	High	N/A	No	N/A	Low-Medium
CLF	N/A	N/A	N/A	Low	Medium-High	Low	Medium - High
REG	Low	Low	N/A	Low	High	Low-Medium	Medium - High
GEN	Low	Low	Medium-High	High	Low-Medium	Medium-High	High

Table 5: Overview of pros and cons of different models. We list baselines: regular expressions (`REGEX`), dependency parsing (`DEP`), and semantic role labeling (`SRL`). The `CLF`, `REG`, `GEN` refer to the fine-tuned NT5-Clf, NT5-Reg, and NT5-Gen models. `Absolute / Relative Error` pertains to the absolute/relative error between true victim counts and model predictions taking the real numerical value of the counts (e.g., mean squared error). `String Match` considers string metrics like *Exact-Match* used in question answering. The reliability column is based on experiments in model calibration. Robustness is divided into the need for training on a large annotated dataset and the stability in out-of-distribution (`OOD`). `N/A` means "Not Applicable".

synthetic data containing numbers. Codex (Chen et al., 2021) and NT5 (Yang et al., 2021) apply similar strategies and are trained on code and math word problems. Other approaches focus on step-by-step reasoning such as Minerva (Lewkowycz et al., 2022), scratchpad (Nye et al., 2021) and chain-of-thought prompting (Wei et al., 2022). Lefebvre and Stoehr (2022) propose a prompting-based method particularly for conflict event classification.

Calibration of NLP Models. The calibration of NLP models has been extensively studied in classification (Guo et al., 2017) and structured prediction tasks (Kuleshov and Liang, 2015; Nguyen and O'Connor, 2015). Calibration methods have been adapted in language modeling (Braverman et al., 2020; Kong et al., 2020), question answering (Kamath et al., 2020; Jiang et al., 2021), and machine translation (Kumar and Sarawagi, 2019; Wang et al., 2020).

11 Conclusion

We presented *victim count extraction*, a challenging and impactful task. The task can be tackled using different formulations and models. Models should be evaluated along different dimensions such as accuracy, reliability, and robustness. We survey this ambiguity of victim count extraction, identify promising directions, and discuss outlooks and applications.

Limitations

The models may be biased or reproduce biases inherent in their training data. Presenting unrelated, faulty or immoral questions to a model can cause unguided and malicious behavior. For example, we caution of asking questions such as "How many people *will be injured*...?"; and even worse "How many people *should be injured*...?". Improving model calibration will help defending against these issues and enable awareness of when to abstain from answering.

Ethics Statement

This work originated from the motivation to diversify victim count extraction towards underrepresented victim types and overlooked forms of violence. This work ultimately intends to assist researchers and analysts in the sector of humanitarian aid who are in demand of accurate victim count information.

References

Mark Braverman, Xinyi Chen, Sham Kakade, Karthik Narasimhan, Cyril Zhang, and Yi Zhang. 2020. Calibration, entropy rates, and memory in language models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1089–1099. PMLR.

Stephen Camilleri, Joel Azzopardi, and Matthew R. Agius. 2019. Investigating the relationship between earthquakes and online news. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, page 203–210.

Stephen Chaudoin, Zachary Peskowitz, and Christopher Stanton. 2017. Beyond zeroes and ones: The intensity and dynamics of civil conflict. *The Journal of Conflict Resolution*, 61(1):56–83. Publisher: Sage Publications, Inc.

Kunlong Chen, Weidi Xu, Xingyi Cheng, Zou Xiaochuan, Yuyu Zhang, Le Song, Taifeng Wang, Yuan Qi, and Wei Chu. 2020. Question directed graph attention network for numerical reasoning over text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6759–6768, Online. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. In *arXiv*, volume 2107.03374.

Maarten Cruyff, Jan van Dijk, and Peter G. M. van der Heijden. 2017. The challenge of counting victims of human trafficking: Not on the record: A multiple systems estimation of the numbers of human trafficking victims in the Netherlands in 2010–2015 by year, age, gender, and type of exploitation. *CHANCE*, 30(3):41–49.

Patricia Cullen, Myrna Dawson, Jenna Price, and James Rowlands. 2021. Intersectionality and invisible victims: Reflections on data challenges and vicarious trauma in femicide, family and intimate partner homicide research. *Journal of Family Violence*, 36(5):619–628.

James Darcy and Charles-Antoine Hofmann. 2003. According to need? Needs assessment and decision-making in the humanitarian sector. Technical report, Overseas Development Institute.

Dmitry Davidov and Ari Rappoport. 2010. Extraction and approximation of numerical attributes from the web. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, page 1308–1317, Uppsala, Sweden. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Lars Döhling and Ulf Leser. 2011. EquatorNLP: Pattern-based information extraction for disaster response. In *Terra Cognita 2011 Workshop, Foundations, Technologies and Applications of the Geospatial Web*, volume 38, Bonn, Germany.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Joshua Goldstein. 1992. A conflict-cooperation scale for WEIS events data. *The Journal of Conflict Resolution*, 36(2):369–385.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330, Sydney, Australia. JMLR.org.

Nitish Gupta, Kevin Lin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *8th International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia.

Andrew Halterman, Katherine Keith, Sheikh Sarwar, and Brendan O'Connor. 2021. Corpus-level evaluation for event QA: The IndiaPoliceEvents corpus covering the 2002 Gujarat violence. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, pages 4240–4253, Online.

Xiaochuang Han, Eunsol Choi, and Chenhao Tan. 2019. No permanent friends or enemies: Tracking relationships between nations from news. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 1660–1676.

Sebastian Hellmeier, Nils B. Weidmann, and Espen Geelmuyden Rød. 2018. In the spotlight: Analyzing sequential attention effects in protest reporting. *Political Communication*, 35(4):587–611. Publisher: Routledge.

Muhammad Imran, Shady Elbassuoni, Carlos Castillo, Fernando Diaz, and Patrick Meier. 2013. Practical extraction of disaster-relevant information from social media. In *Proceedings of the 22nd International Conference on World Wide Web*, page 1021–1024, Rio de Janeiro, Brazil. ACM.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? On the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

JRC Science Hub. 2018. European media monitor (EMM). Technical report, Joint Research Center Science Hub.

Stathis N. Kalyvas. 2006. *The Logic of Violence in Civil War*. Cambridge University Press, Cambridge.

Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5684–5696, Online. Association for Computational Linguistics.

Katherine Keith, Abram Handler, Michael Pinkham, Cara Magliozzi, Joshua McDuffie, and Brendan O'Connor. 2017. Identifying civilians killed by police with distantly supervised entity-event extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark. Association for Computational Linguistics.

Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, and Chao Zhang. 2020. Calibrated language model fine-tuning for in- and out-of-distribution data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1326–1340, Online. Association for Computational Linguistics.

Bhushan Kotnis and Alberto García-Durán. 2019. Learning numerical attributes in knowledge bases. In *1st Conference on Automated Knowledge Base Construction, AKBC 2019, Amherst, MA, USA, May 20-22, 2019*.

Keith Krause. 2013. Challenges to counting and classifying victims of violence in conflict, post-conflict, and non-conflict settings. In *Counting Civilian Casualties: An Introduction to Recording and Estimating Nonmilitary Deaths in Conflict*. Oxford University Press.

Tino Kreutzer, Patrick Vinck, Phuong N. Pham, Aijun An, Lora Appel, Eric DeLuca, Grace Tang, Muath Alzghool, Kusum Hachhethu, Bobi Morris, Sandie L. Walton-Ellery, John Crowley, and James Orbinski. 2020. Improving humanitarian needs assessments through natural language processing. *IBM Journal of Research and Development*, 64(1/2):9:1–9:14.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. 2018. Accurate uncertainties for deep learning using calibrated regression. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2796–2804. PMLR.

Volodymyr Kuleshov and Percy S Liang. 2015. Calibrated structured prediction. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Aviral Kumar and Sunita Sarawagi. 2019. Calibration of encoder decoder models for neural machine translation. In *arXiv*, volume 1903.00802.

Andrey Kutuzov, Erik Velldal, and Lilja Øvrelid. 2017. Tracing armed conflicts with diachronic word embedding models. In *Proceedings of the Events and Stories in the News Workshop*, pages 31–36.

Clement Lefebvre and Niklas Stoehr. 2022. Rethinking the event coding pipeline with prompt entailment. In *arXiv*, volume 2210.05257.

Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. 2017. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7(1):17816.

Raphael Lepuschitz and Niklas Stoehr. 2021. SeismographAPI: Visualising temporal-spatial crisis data. In *arXiv*, volume 2107.12443.

Orion A. Lewis, Erica Chenoweth, and Jonathan Pinckney. 2016. Nonviolent and violent campaigns and outcomes 3.0: Effects of tactical choices on strategic outcomes codebook. Technical Report Version 0.9b5, Sie Cheou-Kang Center for International Security and Diplomacy University of Denver.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *arXiv*, volume 2206.14858.

Khanh Nguyen and Brendan O'Connor. 2015. Posterior calibration and exploratory analysis for natural language processing models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1587–1598, Lisbon, Portugal. Association for Computational Linguistics.

Clayton Norris, Philip Schrodt, and John Beieler. 2017. PETRARCH2: Another event coding program. *The Journal of Open Source Software*, 2(9).

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show your work: Scratchpads for intermediate computation with language models. In *arXiv*, volume 2112.00114.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Benjamin J. Radford. 2021. Automated dictionary generation for political eventcoding. *Political Science Research and Methods*, 9(1):157–171.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Clionadh Raleigh and Roudabeh Kishi. 2019. Comparing conflict data – Similarities and differences across datasets. ACLED Report, The Armed Conflict Location & Event Data Project (ACLED).

Clionadh Raleigh, Andrew Linke, Håvard Hegre, and Joakim Karlsen. 2010. Introducing ACLED-armed conflict location and event data. *Journal of Peace Research*, 47(5):651–660.

Subhro Roy, Tim Vieira, and Dan Roth. 2015. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics*, 3:1–13.

Koustav Rudra, Niloy Ganguly, Pawan Goyal, and Saptarshi Ghosh. 2018. Extracting and summarizing situational information from the Twitter social media during disasters. *ACM Transactions on the Web*, 12(3):1–35.

Giuseppe Russo, Christoph Gote, Laurence Brandenberger, Sophia Schlosser, and Frank Schweitzer. 2022a. Disentangling active and passive cosponsorship in the U.S. congress. In *arXiv*, volume 2205.09674.

Giuseppe Russo, Manoel Horta Ribeiro, Giona Casiraghi, and Luca Verginer. 2022b. Understanding online migration decisions following the banning of radical communities. In *arXiv*, volume 2212.04765.

Philip A. Schrodt. 2009. TABARI: Textual analysis by augmented replacement instructions. Technical Report 0.7, Parus Analytics.

Philip A. Schrodt and Jay Ulfelder. 2016. Political instability task force atrocities event data collection codebook. Technical Report Version 1.1b1, Parus Analytics.

Hao Song, Tom Diethe, Meelis Kull, and Peter Flach. 2019. Distribution calibration for regression. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5897–5906. PMLR.

Ralf Steinberger, Martin Atkinson, Teofilo Domingo, Erik van der Goot, Jens Linge, Charles Macmillan, Hristo Tanev, and Gerhard Wagner. 2017. *EMM: Supporting the Analyst by Turning Multilingual Text into Structured Data*. Erich Schmidt Verlag, Berlin, Germany.

Niklas Stoehr, Lucas Torroba Hennigen, Josef Valvoda, Robert West, Ryan Cotterell, and Aaron Schein. 2022. An ordinal latent variable model of conflict intensity. In *arXiv*, volume 2210.03971.

Niklas Stoehr, Benjamin J. Radford, Ryan Cotterell, and Aaron Schein. 2023. The Ordered Matrix Dirichlet for modeling ordinal dynamics. In *arXiv*, volume 2212.04130.

Niklas Stoehr, Lucas Torroba Hennigen, Samin Ahbab, Robert West, and Ryan Cotterell. 2021. Classifying dyads for militarized conflict analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7775–7784, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Irina P. Temnikova, Carlos Castillo, and Sarah Vieweg. 2015. EMTerms 1.0: A terminological resource for crisis tweets. In *12th Proceedings of the International Conference on Information Systems for Crisis Response and Management*, Krystiansand, Norway. ISCRAM Association.

Zhanna Terechshenko. 2020. Hot under the collar: A latent measure of interstate hostility. *Journal of Peace Research*, 57(6):764–776.

Avijit Thawani, Jay Pujara, Filip Ilievski, and Pedro Szekely. 2021. Representing numbers in NLP: a survey and a vision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–656, Online. Association for Computational Linguistics.

Jack Vincent. 1979. Project theory. *Lanham M.D. University Press of America*.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. 2020. On the inference calibration of neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3070–3079, Online. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *arXiv*, volume 2201.11903.

David Widmann, Fredrik Lindsten, and Dave Zachariah. 2021. Calibration tests beyond classification. In *9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria.

Peng-Jian Yang, Ying Ting Chen, Yuechan Chen, and Daniel Cer. 2021. NT5?! Training T5 to perform numerical reasoning. In *arXiv*, volume 2104.07307.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao Zhao, Kai Chen, Mohammad Norouzi Norouzi, and Quoc V. Le. 2018. QANet: Combining local convolution with global self-attention for reading comprehension. In *6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada.

Vanni Zavarella, Jakub Piskorski, Camelia Ignat, Hristo Tanev, and Martin Atkinson. 2020. Mastering the media hype: Methods for deduplication of conflict events from news reports. In *Proceedings of the first ings of the first AI4Narratives Workshop*.

Yunfeng Zhang, Q. Vera Liao, and Rachel K. E. Bellamy. 2020. Effect of confidence and explanation on accuracy and trust calibration in AI-assisted decision making. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 295–305, New York, NY, USA. Association for Computing Machinery.

A Regex Patterns

We convert any non-digitized numeral expressions into a digitized format (e.g. twelve → 12). Regex patterns are designed for both passive and active voices. We also distinguish plural ("are" and "were") and singular forms ("is", "was") for passive voice patterns. The algorithm checks with the following order: passive plural, passive single, and active. If multiple numbers are extracted, the first is kept. We list the regex patterns used to extract victim counts in Tab. 6, for death counts and injury counts respectively.

B Accuracy Evaluation

In this section, we complement the accuracy evaluation of the models in §5.

B.1 *Exact-Match* and F_1 score on Death Counts

The *Exact-Match* and F_1 scores on extracting the death counts are shown in Tab. 7, which compares the performance of the baseline models and the fine-tuned NT5-Gen model. Similar to the results on the injury counts Tab. 1, the fine-tuned NT5-Gen model performs better than all baselines and the SRL has the best accuracy among baselines.

B.2 Confusion Matrix on Death Counts

Similar to Fig. 1 shown in §5.1, Fig. 5 plots the confusion matrices of the binned death counts for the different datasets, which compare the accuracy of the baseline models with the fine-tuned NT5-Gen model.

B.3 Results on Classification and Regression

In §5.2, we have shown the results of the NT5-Clf model and the NT5-Reg model fine-tuned on NAVCO injury counts in Tab. 3.

Here, we use the same metrics and display the classification performance on other datasets. In specific, Tab. 8, Tab. 9, Tab. 10, Tab. 11, and Tab. 12 respectively show the classification performance of the NT5-Clf model fine-tuned on WAD death counts, WAD injury counts, NAVCO death counts, EMM death counts, and EMM injury counts.

Similarly, we provide the scatter plots of the fine-tuned NT5-Reg models initialized with different pre-trained weights in this section: WAD death counts (Fig. 6), WAD injury counts (Fig. 7), NAVCO death counts (Fig. 8), EMM death counts (Fig. 9), and EMM injury counts (Fig. 10).

C Robustness Evaluation

In this section, we provide the detailed performance of the few-shot setting (App. C.1) and the out-of-distribution setting (App. C.2) discussed in §7.

C.1 Few-shot Performance

We display the results of the few-shot settings where different proportions of the training set are used to fine-tune the models. For each formulation, the left figure is the variation of the accuracy metrics and the right figure is the variation of the calibration error. Fig. 11, Fig. 12, and Fig. 13 are performance of the few-shot settings of the fine-tuned NT5-Clf, NT5-Reg, and NT5-Gen models respectively.

With respect to accuracy metrics, the classification accuracy and the F_1 score is plotted for the fine-tuned NT5-Clf model.

For the regression, we plot the change of mean squared error on the $\log$ transformed counts. In addition, we plot the pinball losses using two targeting quantile (at 10% and at 90%).

Lastly, the *Exact-Match* and the F_1 scores are drawn for the fine-tuned NT5-Gen model.

C.2 Out-of-distribution Setting

For each task formulation, we examine the accuracy performance in the out-of-distribution setting for the fine-tuned NT5-Clf (Fig. 14), NT5-Reg (Fig. 15), and NT5-Gen (Fig. 16). For all plots, the x-axis is the accuracy metric used in each task formulation, and the y-axis indicates the test set to be made inferences on. The red bar indicates the

<table>
<tr><th>Data Type</th><th>Regex Type</th><th>Regex Pattern</th></tr>
<tr><td>Death</td><td>Passive Plural</td><td><code>\d(\d|,)*(?!\D*(injur|wound))(?=.*(\b(were|are)\D*\b(killed|dead|died|slain)))</code></td></tr>
<tr><td></td><td>Passive Singular</td><td><code>\S*(?!\D*(injur|wound))(?=.*(\b(was|is)\D*\b(killed|dead|died|slain)))</code></td></tr>
<tr><td></td><td>Active</td><td><code>(kill|slay|slain)\D*\b\d(\d|,)*</code></td></tr>
<tr><td>Injury</td><td>Passive Plural</td><td><code>\d(\d|,)*(?!.*(\b(were|are)?\D*\b(killed|dead|died|slain)))(?=.*\b(injur|wound))</code></td></tr>
<tr><td></td><td>Passive Singular</td><td><code>\S*(?=(was|is).*\b(injur|wound))(?!\D*(\b(were|are)\D*\b(killed|dead|died|slain)))</code></td></tr>
<tr><td></td><td>Active</td><td><code>(injured?|wound)\D*\d+</code></td></tr>
</table>

Table 6: Regex patterns.

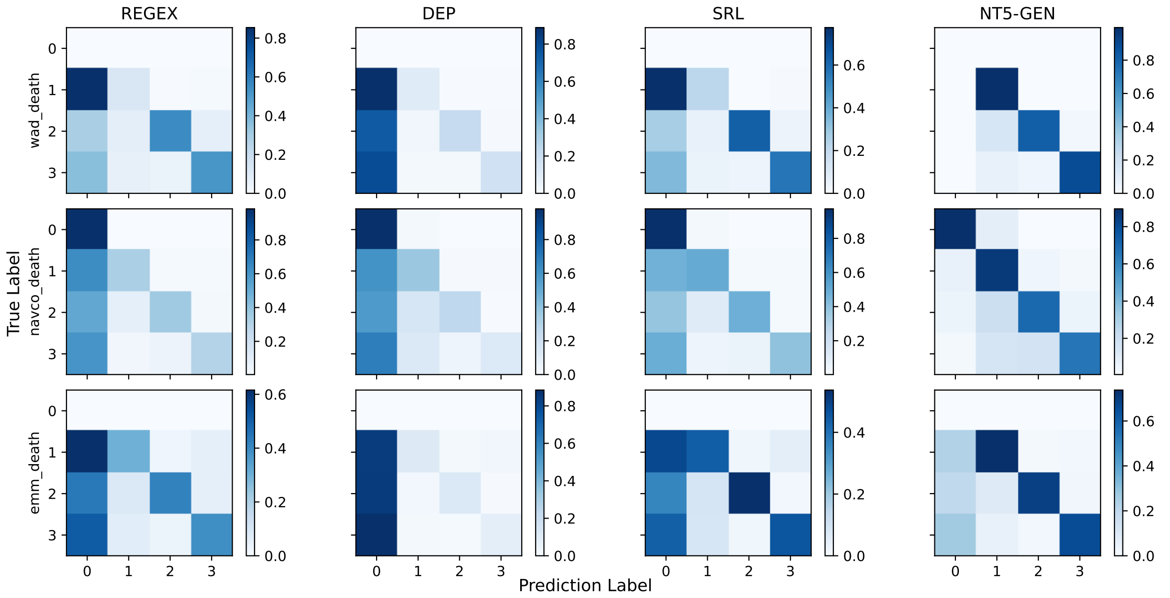


Figure 5: Confusion matrices of baseline models and fine-tuned NT5-Gen model (columns) of extracting death counts from different data (rows). We convert the true victim counts and model predictions into 4 categories: for any count y, "0" is $y = 0$, "1" is $0 < y \leq 3$, "2" is $3 < y \leq 10$ and "3" is $y > 10$. Values are normalized over true counts.

	Exact Match			F_1		
	WAD	NAVCO	EMM	WAD	NAVCO	EMM
Regex	0.3543	0.3921	0.2835	0.3897	0.4196	0.3242
Dep	0.1506	0.3526	0.0767	0.2064	0.3792	0.1317
SRL	0.4342	0.4839	0.3972	0.7794	0.4837	0.3613
NT5	**0.6798**	**0.6590**	**0.6322**	**0.8458**	**0.5436**	**0.4917**

Table 7: *Exact-Match* and F_1 scores of the baseline models and the fine-tuned NT5-Gen model on death counts. Best metrics are **bolded**. DEP refers to the dependency parsing model and SRL refers to the semantic role labeling model.

performance of in-distribution performance, e.g., accuracy of WAD death test data using the model fine-tuned on WAD death.

With respect to the accuracy metric, different formulations use their corresponding metric. For the classification setting, we show the variation in classification accuracy. For the regression setting, we show the variation in mean squared errors.

	Accuracy	F1 score	Precision	Recall
NT5	0.81	0.81	0.80	0.83
T5	0.81	0.81	0.81	0.84
BERT	0.86	0.86	0.86	0.88

Table 8: Classification results on WAD death counts with the NT5-Clf model initialized by different pre-trained weights: NT5, T5-SMALL, and BERT-BASE-UNCASED. F_1 , precision and recall scores are macro.

	Accuracy	F1 score	Precision	Recall
NT5	0.77	0.69	0.70	0.69
T5	0.76	0.69	0.70	0.68
BERT	0.93	0.91	0.91	0.90

Table 9: Classification results on WAD injury counts with the NT5-Clf model initialized by different pre-trained weights: NT5, T5-SMALL, and BERT-BASE-UNCASED. F_1 , precision and recall scores are macro.

	Accuracy	F1 score	Precision	Recall
NT5	0.65	0.60	0.62	0.59
T5	0.65	0.60	0.61	0.59
BERT	0.52	0.23	0.17	0.33

Table 10: Classification results on NAVCO death counts with the NT5-Clf model initialized by different pre-trained weights: NT5, T5-SMALL, and BERT-BASE-UNCASED. F_1 , precision and recall scores are macro.

	Accuracy	F1 score	Precision	Recall
NT5	0.72	0.65	0.66	0.65
T5	0.70	0.63	0.65	0.63
BERT	0.84	0.80	0.82	0.78

Table 11: Classification results on EMM death counts with the NT5-Clf model initialized by different pre-trained weights: NT5, T5-SMALL, and BERT-BASE-UNCASED. F_1 , precision and recall scores are macro.

	Accuracy	F1 score	Precision	Recall
NT5	0.68	0.58	0.60	0.57
T5	0.68	0.58	0.59	0.57
BERT	0.81	0.77	0.79	0.76

Table 12: Classification results on EMM injury counts with the NT5-Clf model initialized by different pre-trained weights: NT5, T5-SMALL, and BERT-BASE-UNCASED. F_1 , precision and recall scores are macro.

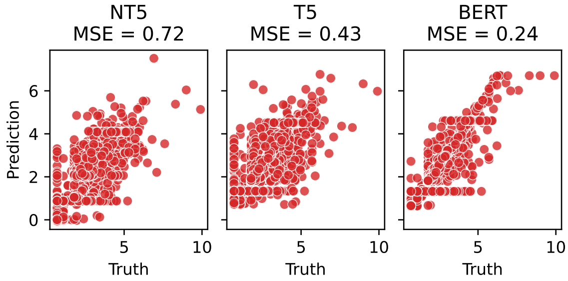


Figure 6: Scatter plots of the fine-tuned NT5-Reg model initialized with different pre-trained weights (NT5, T5-SMALL, and BERT-BASE-UNCASED). The models are trained on log-transformed victim counts of WAD death.

For the generation setting, we show the change in *Exact-Match* scores.

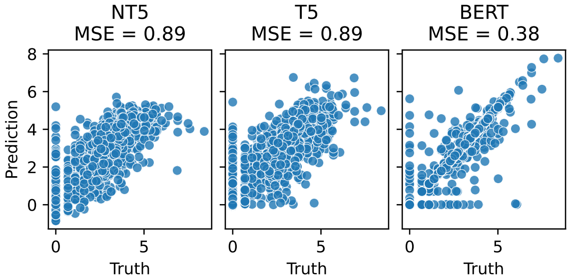


Figure 7: Scatter plots of the fine-tuned NT5-Reg model initialized with different pre-trained weights (NT5, T5-SMALL, and BERT-BASE-UNCASED). The models are trained on log-transformed victim counts of WAD injury.

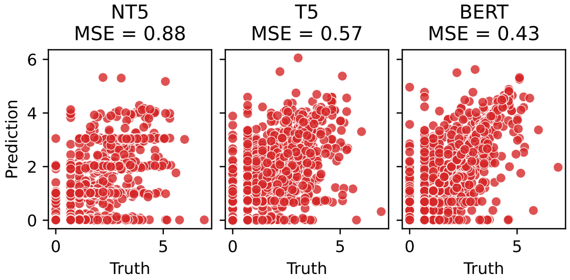


Figure 8: Scatter plots of the fine-tuned NT5-Reg model initialized with different pre-trained weights (NT5, T5-SMALL, and BERT-BASE-UNCASED). The models are trained on log-transformed victim counts of NAVCO death.

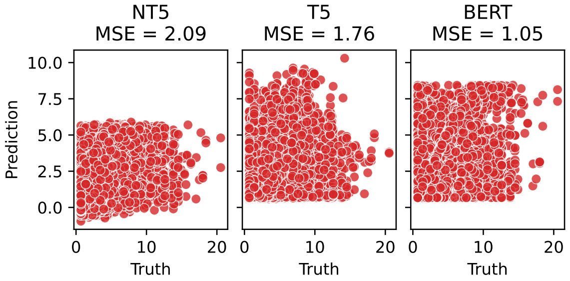


Figure 9: Scatter plots of the fine-tuned NT5-Reg model initialized with different pre-trained weights (NT5, T5-SMALL, and BERT-BASE-UNCASED). The models are trained on log-transformed victim counts.

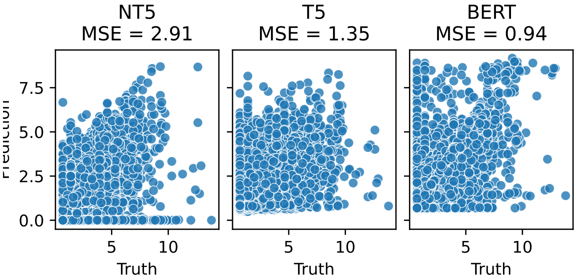


Figure 10: Scatter plots of the fine-tuned NT5-Reg model initialized with different pre-trained weights (NT5, T5-SMALL, and BERT-BASE-UNCASED). The models are trained on log-transformed victim counts.

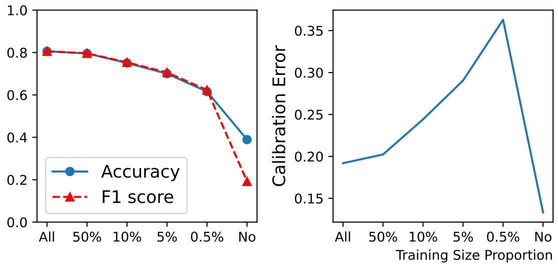


Figure 11: Few-shot performance of the fine-tuned classification model on WAD death counts.

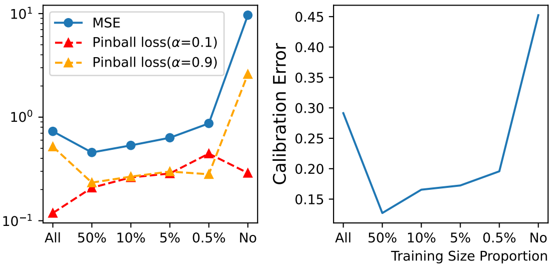


Figure 12: Few-shot performance of the fine-tuned regression model on WAD death counts.

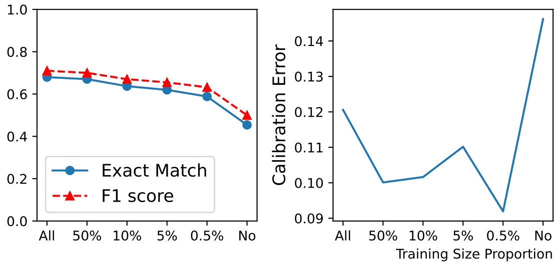


Figure 13: Few-shot performance of the fine-tuned NT5-Gen model on WAD death counts.

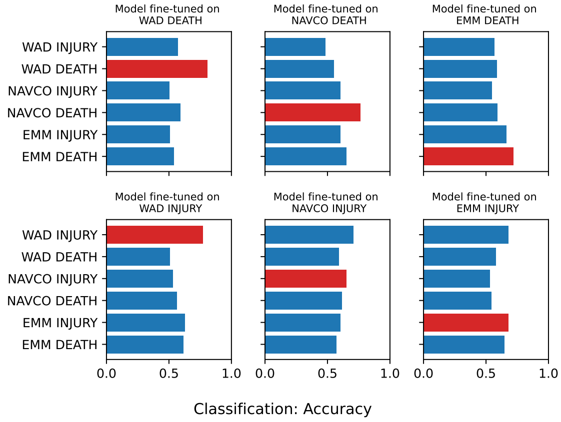


Figure 14: Classification accuracy for using the fine-tuned NT5-Clf models on out-of-distribution data (blue) and on in-distribution data (red)

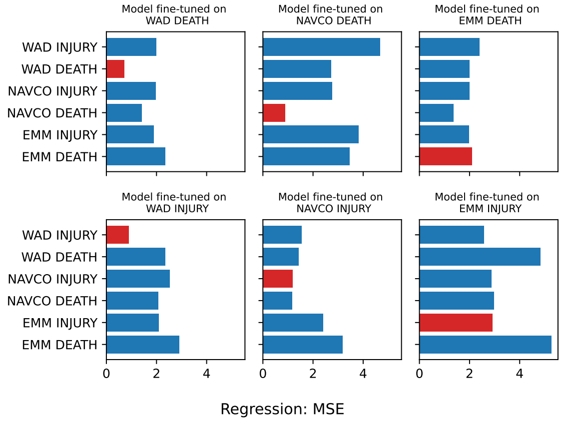


Figure 15: Mean squared error for using the fine-tuned NT5-Reg models on out-of-distribution data (blue) and on in-distribution data (red)

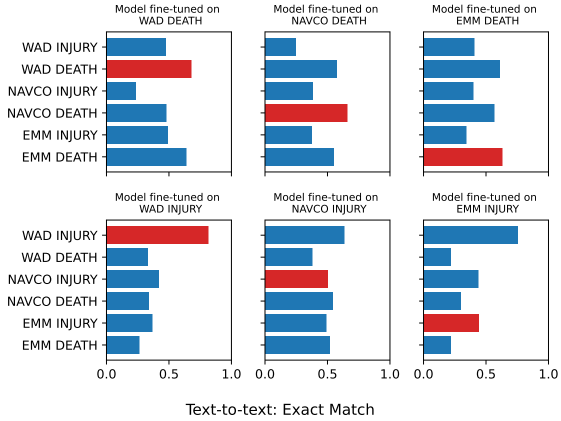


Figure 16: *Exact-Match* for using the fine-tuned NT5-Gen models on out-of-distribution data (blue) and on in-distribution data (red)

ConEntail: An Entailment-based Framework for Universal Zero and Few Shot Classification with Supervised Contrastive Pretraining

Ranran Haoran Zhang[1], **Aysa Xuemo Fan**[2], **Rui Zhang**[1]
[1] Penn State University
[2] University of Illinois at Urbana-Champaign
`{hzz5361, rmz5227}@psu.edu`
`xuemof2@illinois.edu`

Abstract

A universal classification model aims to generalize to diverse classification tasks in both zero and few shot settings. A promising way toward universal classification is to cast heterogeneous data formats into a dataset-agnostic “meta-task” (e.g., textual entailment, question answering) then pretrain a model on the combined meta dataset. The existing work is either pretrained on specific subsets of classification tasks, or pretrained on both classification and generation data but the model could not fulfill its potential in universality and reliability. These also leave a massive amount of annotated data under-exploited. To fill these gaps, we propose ConEntail, a new framework for universal zero and few shot classification with supervised contrastive pretraining. Our unified meta-task for classification is based on nested entailment. It can be interpreted as “Does sentence a entails [sentence b entails label c]”. This formulation enables us to make better use of 57 annotated classification datasets for supervised contrastive pretraining and universal evaluation. In this way, ConEntail helps the model (1) absorb knowledge from different datasets, and (2) gain consistent performance gain with more pretraining data. In experiments, we compare our model with discriminative and generative models pretrained on the same dataset. The results confirm that our framework effectively exploits existing annotated data and outperforms baselines in both zero (9.4% average improvement) and few shot settings (3.5% average improvement). Our code is available at `https://github.com/psunlpgroup/ConEntail`.

1 Introduction

It has been a long-standing effort to solve various text classification tasks by training one universal model (Kumar et al., 2016). With an ideal universal classification model, we can expect extreme generalization with few or zero annotation in new domains/tasks/datasets. To this end, researchers reformulate heterogeneous task definitions into a unified format of a meta-task in natural language (Yin et al., 2020; Khashabi et al., 2020a). Solving the meta-task is equivalent to solving the isolated tasks, thus the meta-task paves the way of supplementing unsupervised pretrained Language Models (PLM) with additional supervised pretraining, to further absorb knowledge from heterogeneous labeled data.

The success of universal classification models hinges on how well a strong PLM understands natural language meta-task. The meta-task format depends on two underlying PLM types: (a) **discriminator** uses Encoder PLMs and treats all classification tasks as binary entailment classification problem (Yin et al., 2019, 2020; Xia et al., 2021; Wang et al., 2021). However, they only pretrain models on Natural Language Inference datasets, whose knowledge is not comprehensive comparing all classification tasks (Ma et al., 2021). (b) **generator** uses Encoder-Decoder PLMs and treats all tasks as text generation problem (Gao et al., 2020; Raffel et al., 2020; Sanh et al., 2021; Aribandi et al., 2021; Ye et al., 2021a; Bragg et al., 2021; Du et al., 2021; Schick and Schütze, 2021a,b). Thus they are compatible with both classification tasks and generation tasks. However, the generator nature implies that the predicted texts may not match any possible labels, thus more likely to fail on classification tasks (Sanh et al., 2021).

Based on our observations and experiments, we argue that the discriminators have more potential in universal classification, and propose a new discriminator framework, ConEntail, that can make better use of existing annotated datasets. Concretely, we reformulate the unified meta-task as a nested entailment: “Does sentence q entails [sentence p entails label h]”. Take Fig. 1 as an example, the query “We had a great breakfast at the waffle shop!” entails the same label as the premise “I bought this for myself a short time ago and I love it. An excellent

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1941–1953

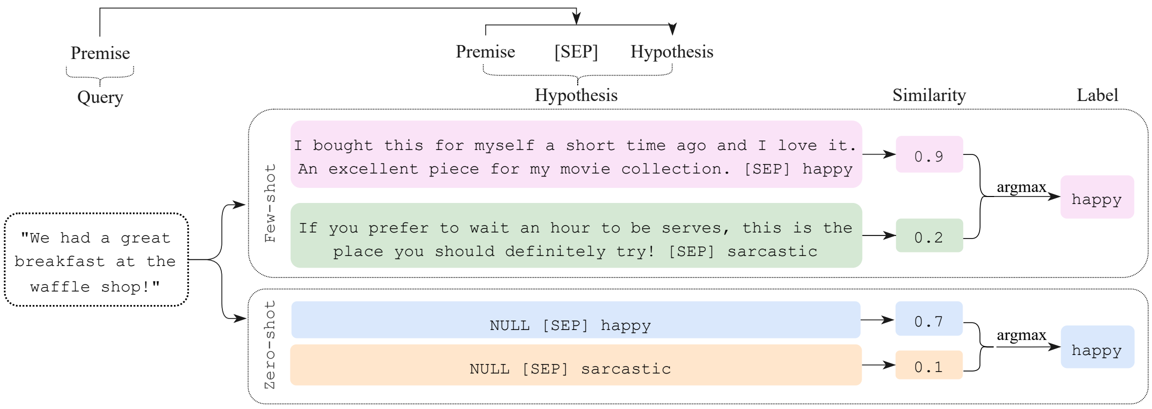


Figure 1: The overview of the ConEntail framework. By casting the classification as a nested entailment task, the model performs classification by telling if a query sentence q entails [premise example p entails hypothesis label h]. In a few-shot setting, the premise is an example sentence; in a zero-shot setting, the premise is a "NULL" placeholder.

piece for my movie collection.", so it yields a high similarity score of 0.9, in this case, it is higher than any other similarities, thus, the prediction would be "happy". For zero-shot generalization, as no annotated sentences are available, we replace the premise p with "NULL" in evaluation. We randomly nullify a small ratio of p in the supervised pretraining for training-evaluation consistency. The supervised contrastive learning framework pulls sentences embeddings with the same label together and pushes those with different labels apart, thus capturing more similarities/dissimilarities from labeled data, and benefiting few/zero-shot learning.

In experiments, we collect 56 classification datasets from Crossfit (Ye et al., 2021a), together with their templates, to formulate a large supervised pretraining dataset. We reproduce EFL (Wang et al., 2021), Unifew (Bragg et al., 2021) and Crossfit (Ye et al., 2021a) in the same setting and control influences of PLM supervised pretraining data, then conduct fair comparison with our proposed ConEntail. The experiments show that generators (Unifew and Crossfit) do not fit the classification task well and thus significantly under-perform the random guess in zero-shot evaluation; standard discriminators (EFL) under-exploit supervised pretraining datasets and thus do not gain consistent improvement as pretraining data scale up, while ConEntail makes the best use of the supervised pretraining data and keep consistent performances. Our model outperforms baselines in both zero (9.4% average improvement) and few shot settings (3.5% average improvement).

Our contributions are the following:

- We propose a novel universal classification framework based on nested entailment, ConEntail, that can be used in both zero and few shot settings. It makes better use of supervised pretraining datasets and consistently improves performances with increases of the pretraining scale.
- We design systematic experiments to compare generative and discriminative models, and more importantly, we give in-depth analysis to reveal their attributes in universal classification task.
- Our model reliably outperforms the baseline models in all kinds of pretraining size, finetuning size, and covers a wide range of tasks.

2 Related Work

Universal Meta Task Casting heterogeneous datasets into a unified meta-task allows researchers to train one model to solve all tasks. There are two types of meta-task formats, generation (Schick and Schütze, 2021a,b; Gao et al., 2020; Ye et al., 2021a; Bragg et al., 2021; Khashabi et al., 2020a) and discrimination (Yin et al., 2019, 2020; Xia et al., 2021; Wang et al., 2021). The generators formulate meta-task as a text-to-text generation problem. Although their supervised pretraining usually involves both classification and generation tasks, as the text outputs are open-ended, the model predictions may fall out of all possible labels. The discriminators formulate meta-task as an entailment classification problem, and usually use Natural Language Inference datasets for supervised pretraining. We extend discriminator pretraining to more classification datasets and propose a nested entailment meta-task to enable a more efficient supervised pretraining method.

Supervised Pretraining Supervised pretraining originates from explicit multitask learning (Caruana, 1997) which combines different task knowledge into shared representations. Phang et al. (2018) found that supplementing PLMs with supervised pretraining between unsupervised pretraining and downstream finetuning can significantly boost the performance and few-shot generalization. The discriminator models including UFO-Entail (Yin et al., 2020) and EFL (Wang et al., 2021) are trained on MNLI (Williams et al., 2018) in a supervised fashion, but they do not combine different sources of datasets. Furthermore, T0 (Sanh et al., 2021) and ExT5 (Aribandi et al., 2021) extends T5 (Raffel et al., 2020) by using 107 and 171 datasets for supervised pretraining and conduct zero-shot evaluation. FLEX (Bragg et al., 2021) and Crossfit (Ye et al., 2021a) extends the supervised pretraining evaluation to few-shot learning.

The supervised pretraining strategies from these works vary in pretraining datasets and hyperparameters, but they mostly follow their underlying language model tasks, such as Next Sentence Prediction or Text Generation. We argue that applying the unsupervised pretraining strategy to supervised pretraining is an underuse of the labeled data, and propose a supervised contrastive learning method on PLMs for better zero/few-shot generalization.

Contrastive Learning for NLP Contrastive learning aims to create embeddings such that similar examples are close while dissimilar examples are far away (Chopra et al., 2005). While most works use self-supervised contrastive learning (Shen et al., 2020; Fang et al., 2020; You et al., 2021; Ye et al., 2021b), only a few adopt supervised contrastive learning. CLIP (Radford et al., 2021) uses labeled images and captions as supervision signal. SimCSE (Gao et al., 2021) and SBERT (Reimers and Gurevych, 2019) use labeled sentence pairs from NLI to construct positive and negative examples. However, their contrastive data creations are limited to specific types of data, and thus can be hardly extended to universal classification. We reformulate all NLP classification tasks into a unified contrastive meta-task and use Supervised Contrastive Loss (Khosla et al., 2020) to train on heterogeneous labeled data during supervised pretraining.

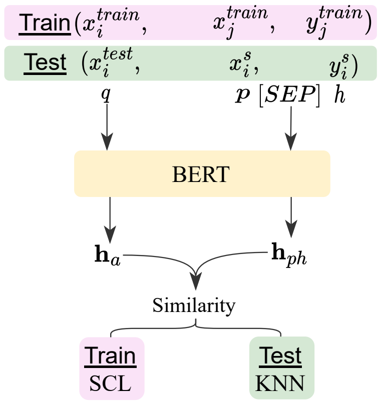


Figure 2: During supervised pertaining, the ConEntail model is optimized with pairwise contrastive learning loss SCL. Testing utilizes the K-Nearest Neighbor predictor to rank pairwise similarities between the query and premise-hypothesis pairs for retrieval of the most likely label. Zero-shot training/testing occurs when the premise example is represented by a "NULL" token."

3 Method

3.1 Universal Classification

Universal classification task aims to build a universal predictor that generalize to new domain/task/-dataset based on only a few or zero newly annotated examples. In order for models to understand a new area, any available resources should be considered for learning, including PLMs trained on large-scale unsupervised data and heterogeneous supervised classification datasets in the NLP community. To leverage heterogeneous datasets, the disparate input-output formats need to be reformulated to a unified PLM comprehensible format, i.e., “meta task”, through either human-curated or machine-generated templates. Then a universal model on the combined meta dataset is trained, which applies universal predictors to new areas. Because the meta task format is compatible with every task, we can cast target tasks into the same format, in this way solving the meta task is equivalent to solving tasks in a new area.

3.2 ConEntail: Nested Entailment

In this paper, we introduce a supervised contrastive pretraining paradigm that makes better use of supervised pretraining. The overview is shown in Fig. 2. Our ConEntail model takes 3 inputs:

$$
\begin{gathered}
f\colon \mathcal{Q}, \mathcal{P}, \mathcal{H} \rightarrow \{0, 1\} \\
q, p, h \mapsto b
\end{gathered}
$$

where $q \in \mathcal{Q}$ is the **query** sentence to be classified. $p \in \mathcal{P}$ is the exemplar sentence as a **premise**, $h \in \mathcal{H}$ is the **hypothesis** verbalized from the label of p. The task of ConEntail is to determine **if** q **entails [**p **entails** h**]**.

We follow (Khashabi et al., 2020b; Ye et al., 2021a) and translate sentence and label (x, y) to (q, p, h) in a PLM comprehensible format, e.g.,

- $x \mapsto q$, where q is the input sentence x with multiple-choice, for example, *(1) happy (2) sarcastic (3) sad, sentence: I bought this for myself ...*
- $x \mapsto p$: where p is the input sentence x with premise, for example, *sentence: I bought this for myself ...*
- $y \mapsto h$ where h is the label name, for example, h: *happy*

where we provide q with all possible labels as multiple-choice questions, and concatenate them in a linearized sentence. In supervised pretraining, q and p are two different surface forms of the same x, so that we can construct positive and negative examples for the later contrastive learning. In the test, q is the query sentence to be clarified and p and h are from the support set.

We use $\text{BERT}_{\text{base}}$ to encode sentences to vector representation $\mathbf{h}$.

$$\mathbf{h}_q = \text{BERT}_{\text{base}}(q) \tag{1}$$

p and h are then concatenated into one sequence to be fed into the encoder:

$$ph = p\,\texttt{[SEP]}\,h \tag{2}$$

$$\mathbf{h}_{ph} = \text{BERT}_{\text{base}}(ph) \tag{3}$$

In the supervised pretraining, the embeddings of each mini-batch are composed by $\left\{\mathbf{h}_q^i, \mathbf{h}_{ph}^i\right\}_{i=1,\ldots,N}$, where N is the batch size. Then we calculate their pairwise cosine similarity $\text{sim}\left(\mathbf{h}_q^i, \mathbf{h}_{ph}^j\right) = \frac{\mathbf{h}_q^i \cdot \mathbf{h}_{ph}^j}{\|\mathbf{h}_q^i\| \cdot \|\mathbf{h}_{ph}^j\|}$ for contrastive training. $s_{ij} \in \{0, 1\}$ is denoted as the groundtruth of the predicted similarity, where $s_{ij} = 1$ is a positive pair when $y_i = y_j$, and vice versa. The positive/negative examples are constructed by all combinations of instances in the batch, note that we did not mine hard examples. We follow the balanced sampling strategy from Meta Classification Learning (Hsu et al., 2019) that each label in a mini-batch has an equal number of input sentences.

In the test phase, we calculate cosine similarities between q and all possible ph and output the most similar h as the prediction result. Thus, we consider our setting as a K-way N-shot learning, where K is determined by the test set, N varies from 0 to 80 in our experiments.

Given the pairwise similarity, we use Supervised Contrastive Loss (Khosla et al., 2020) to train the model:

$$\mathcal{L} = -\sum_{i=1}^{N} \frac{1}{|P(i)|} \sum_{p=1}^{N} \mathbb{1}_{y_i=y_p} \mathbb{1}_{i \neq p} \log \frac{\exp\left(\text{sim}\left(\mathbf{h}_q^i, \mathbf{h}_{ph}^p\right)/\tau\right)}{\sum_{a=1}^{N} \mathbb{1}_{i \neq a} \exp\left(\text{sim}\left(\mathbf{h}_q^i, \mathbf{h}_{ph}^a\right)/\tau\right)} \tag{4}$$

where $|P(i)| = \sum_{p=1}^{N} \mathbb{1}_{y_p=y_i}$ is the number of all positive pairs, τ is the temperature hyperparameters. Different from self-supervised contrastive learning losses, such as SimCSE (Gao et al., 2021), the positive pairs in Supervised Contrastive Loss can be more than one.

To enable zero-shot generalization, inspired by BERT masked language model (Devlin et al., 2019), we introduce a dummy premise "NULL" in both supervised pretraining and testing. During supervised pretraining, we randomly replace 5% of the premise p with "NULL" (**if** q **entails ["NULL" entails** h**]**.). During zero-shot test, the support set is empty and the model uses only "NULL" and label names to answer the question.

4 Experiments

In this section, we describe our experiment setups including dataset selection, evaluation, and baseline models.

4.1 Dataset Selection

For universal text classification, we aim to cover the most popular text classification tasks, such as topic classification, sentiment analysis, paraphrase identification, and natural language inference. Therefore, we adopt Crossfit (Ye et al., 2021a) that provides abundant hand-craft templates covering 56 classification tasks as the source of supervised pretraining and testing. We select 47 datasets as supervised pretraining sets and 9 widely accepted datasets as test sets: CoLA (Warstadt et al., 2018), QQP (Iyer et al., 2017), SST-2 (Socher et al., 2013), MRPC (Dolan and Brockett, 2005),

Method	CoLA	QQP	Hate_speech	MRPC	SCITAIL	Amazon	AGNews	Rotten_tomatoes	SST-2	AVG
			Unseen				*Seen*			
Random-guess	50.5	49.8	34.1	50.0	49.8	49.9	24.0	46.8	49.9	44.9
0-*shot*										
Crossfit†	0.0	0.0	0.0	0.0	0.2*	9.9*	0.0	59.9*	33.4*	11.5*
Unifew†	0.0	0.0	0.0	0.0	48.4*	63.7*	8.0*	57.4*	60.6*	26.5*
EFL	**62.6**	**60.5***	12.7	33.1	47.2*	71.9*	**60.8***	72.5*	79.1*	53.8*
CONENTAIL	58.5*	45.3	**78.3***	**58.1***	**68.7***	**89.7***	52.8*	**78.1***	**83.0***	**63.2***
10-*shot fine-tuning*										
Crossfit†	55.3 ±5.0	53.4 ±9.8	42.8 ±14.4	60.0 ±11.1	58.8 ±5.4	87.9 ±6.1	83.7 ±6.6	75.8* ±1.2	81.2 ±8.9	65.3
Unifew†	49.0 ±4.9	**60.4** ±6.0	34.9 ±6.8	57.7 ±6.3	53.4 ±2.4	88.8 ±3.6	**86.5** ±1.8	73.4 ±9.5	71.2 ±11.5	63.9
EFL	**63.7** ±0.2	**60.4** ±0.2	13.8 ±0.6	33.1* ±0.0	47.2* ±0.1	72.0 ±0.0	62.3 ±0.6	72.5* ±0.0	79.5 ±0.2	55.9
CONENTAIL	60.5* ±0.6	55.6 ±3.5	**44.7** ±2.2	**69.9*** ±0.9	**71.0*** ±0.9	**89.4*** ±0.1	70.3* ±2.1	**78.7*** ±0.2	**83.2*** ±0.2	**68.8***

Table 1: The main results of CONENTAIL compared with baselines. † indicates the models are generative models and the others are discriminative models. In the 10-shot evaluation, to offset the high variances from fine-tuning on such a small support set, the models are fine-tuned by 3 different random sampled support sets. After conducting experiments with and without supervised pretraining, we report the mean accuracy scores and the standard deviation of the best versions of models (in **bold**). We split the test sets in two groups, *seen* and *unseen*, which indicates if the test label names have occurred in the supervised pretraining. AVG is the highest average score of the two versions of models. If a model with supervised pretraining is better than that without supervised pretraining, it is indicated with a $*$.

SCITAIL (Khot et al., 2018), Amazon Polarity (Zhang et al., 2015a), AGNews (Zhang et al., 2015b), Rotten_tomatoes (Pang and Lee, 2005), Hate_speech_offensive (Davidson et al., 2017). For the sentence-pair datasets (e.g., QQP, SST-2, MRPC), we adopt the Crossfit method by concatenating the two sentences with `[SEP]` to form one sequence for either q or p. From the 47 datasets for supervised pretraining, we randomly select 128 annotated examples per label. As the same label name may occur in different datasets, to investigate the effect of label name overlapping, we pick 5 (out of 9) selected test sets with overlapping/seen label names for the supervised pretraining. The detailed dataset list is in Appendix B.

4.2 Evaluation

Supervised Pretraining To investigate the effect of the supervised pretraining, we consider two versions of all the compared models: (1) without supervised pretraining: we apply the original PLMs directly to the reformulated input-output test set. (2) with supervised pretraining: we first perform supervised pretraining on the PLMs and then evaluate the models with the updated parameters.

Zero-shot Evaluation In zero-shot evaluation, the only available resources for the target task are the possible label names and the whole test set will be used to evaluate the model.

Few-shot Evaluation In few-shot evaluation, in addition to the label names, a small support set are available for fine-tuning the universal classification model. The support set for each dataset is composed by k random sampled annotated examples per label, from the training data. With small support sets, the evaluation score may have huge variance, thus we fine-tune and evaluate the model with 3 different support sets and report the mean and standard deviation.

4.3 Baseline Models

We aim to evaluate models in different paradigms in the same universal classification experiment setting. To this end, we compare three baselines that are most representative of the current literature on generators and discriminators.

In this paper, we only consider the differences of the baselines in the meta-task formulation and their generator/discriminator nature while keeping other factors the same, so we reproduce the baselines strictly follow this rule, and use a similar size of pretrained language models as backbones, for a fair comparison. Because our generator/discriminator taxonomy suits many other existing works, with only subtle differences either in the templates or in the backbone PLMs from the baselines mentioned

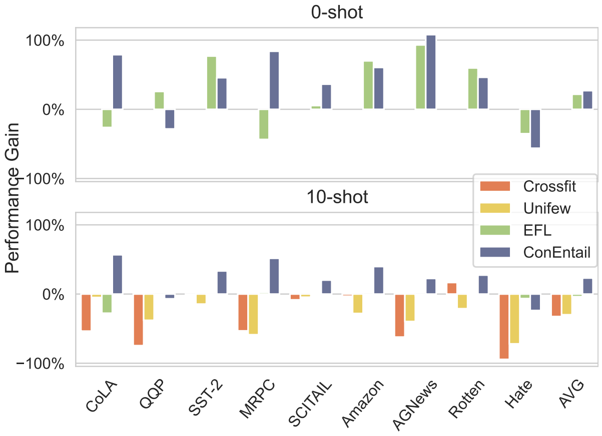


Figure 3: Relative performance gain of supervised pretraining on different datasets and models. The setting is the same with the main experiment. We do not plot zero-shot gains for the generators because most scores are 0 before and after supervised pretraining.

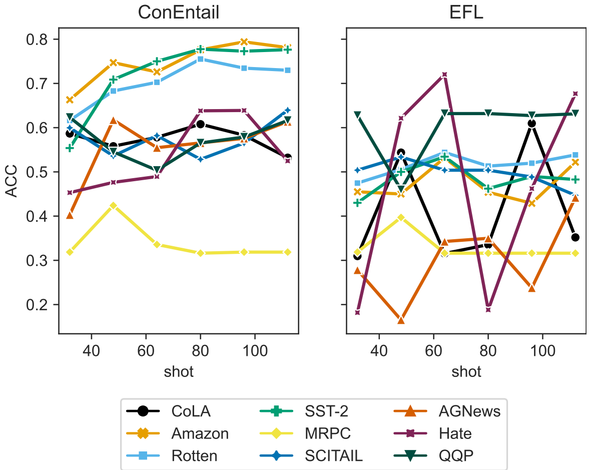


Figure 4: The effect of supervised pretraining data size. We show the zero-shot performance of CONENTAIL and EFL using different pretraining data size from 32 to 128 annotated sentences per label.

here, we do not add more baselines for comparisons.

Crossfit (Ye et al., 2021a): A generative model uses an encoder-decoder structure. The encoder takes the query sentence, and the decoder generates the label name.

Unifew (Bragg et al., 2021): A generative model concatenates all possible labels to the input sentence as multiple-choice question answering. It uses an encoder-decoder structure and generates the label names as answers.

EFL (Wang et al., 2021): A discriminative model reformulates the tasks as multiple entailment binary classifications. Both the query sentence and the label name are fed into the encoder. The embedding of `[CLS]` token is used for binary classification. The label with the highest probability is the predicted output. For supervised pretraining, we enumerate all possible labels for input and provide all the ground truths for the binary classification.

5 Results and Analysis

We design the following experiments to demonstrate and analyze the effectiveness of our method. First, we present the best scores of the compared models with or without supervised pretraining as our main result (Section 5.1). Then, we investigate the performance gain or loss of each model brought by the supervised pretraining (Section 5.2). Furthermore, we study the fine-grained impact of more labeled data in supervised pretraining or of more labeled data in support set (Section 5.3). Considering these results, we discuss the difference between discriminators and generators (Section 5.4). Finally, we show a case study of universal classification under a zero-shot scenario (Section 5.5).

5.1 Main Results

We evaluate the models in two scenarios, 0-shot learning and 10-shot learning (Table 1). The average performances of both discriminator models, EFL and CONENTAIL, significantly outperform random guess and two generation-based models. Particularly, CONENTAIL, with significantly improved average results, performs the best on 6 out of the 9 datasets in both 0-shot and 10-shot settings.

From the table, we also observe that the seen labels bring most improvements to Unifew in 0-shot setting. The 0-shot performance of Unifew in SST-2, SCITAIL and Amazon is far better than Crossfit. This is because Unifew has included the labels in the query sentences as multiple-choice questions, which provides the model additional familiarities from the supervised pretraining. In other words, although the 0-shot unseen accuracies of the generative models are mostly 0, their performances can be improved quickly with few-shot finetuning. This indicates that generative models are promising few-shot learners but not strong zero-shot learners.

5.2 Performance Gain from Supervised Pretraining

We then quantify the effect of supervised pretraining by Relative Performance Gain introduced (Ye et al., 2021a). Relative Performance Gain is the relative improvement brought by the supervised

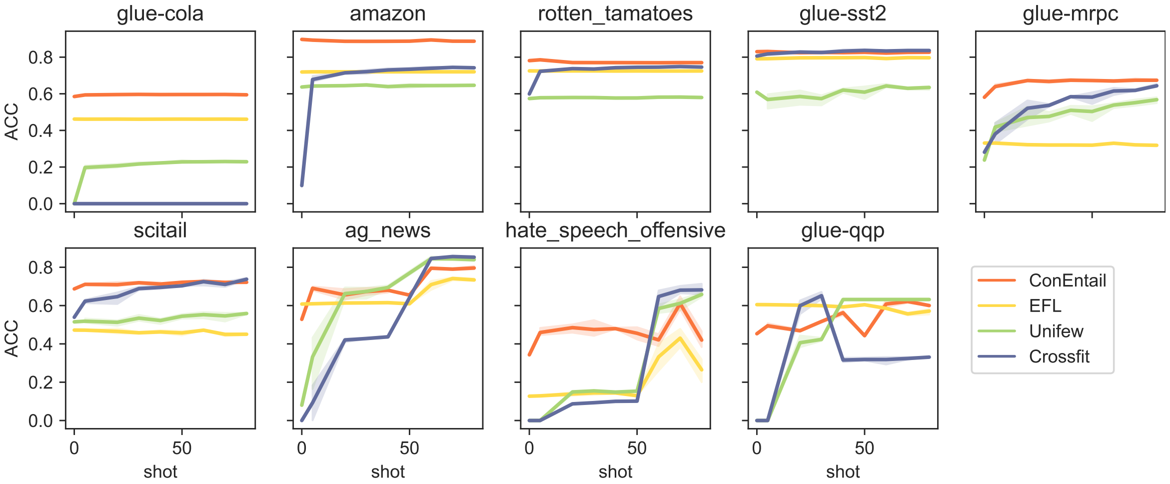


Figure 5: The effect of data size in the support set. We show the accuracy of the compared models fine-tuned with 0 to 80 examples in the support set. For each data size, we randomly sample 3 support sets for fine-tuning and evaluate on the same test set.

pretraining. It is defined as $\frac{\mathrm{Acc}_{\mathrm{w}} - \mathrm{Acc}_{\mathrm{w/o}}}{\mathrm{Acc}_{\mathrm{w/o}}}$, the performance difference between a supervised pretraining model $\mathrm{Acc}_{\mathrm{w}}$ and non-supervised pretraining model $\mathrm{Acc}_{\mathrm{w/o}}$, divided by the latter. The results are shown in Fig. 3.

We observe that supervised pretraining boosts the performance in most datasets in the 0-shot setting. But it lowers the scores in the 10-shot setting, except for CONENTAIL. CONENTAIL's performance rises in 7 out of 9 datasets in both 0-shot and 10-shot setting. This shows the general necessity of supervised pretraining for 0-shot evaluation and the effectiveness of our proposed model in both settings. The baseline models did not benefit from supervised retraining for the 10-shot setting because their conventional fine-tuning strategy is less likely to thoroughly update the parameters than our proposed contrastive learning. Noting that 10-shot evaluation means all the compared models only have 10 labeled examples for finetuning.

5.3 Impact of More Training data

More data in supervised pretraining: we investigate if more labeled data in supervised pretraining can improve zero-shot generalization. As the accuracies of generator models are close to zero in the zero-shot setting, we only consider discriminator models including CONENTAIL and EFL. These two models are supervised pretrained on different-scale datasets (32-128 sentences per label) and evaluated on the 9 test sets. As shown in Fig. 4, the performance of CONENTAIL has fewer fluctuations than the EFL, and the performance improvements of most datasets flat after 80 shots for CONENTAIL. This observation implies that the supervised pretraining has significant and reliable positive effects on CONENTAIL with merely a small amount of supervised dataset.

More data in support set: for models supervised pretrained with 128 annotated sentences per label, we plot the line chart of fine-tuning with 0 to 80 shots. As shown in Fig. 5, adding a few training sentences may not largely boost performance when the universal model is strong enough, but it improves the models significantly if the models have a slow start. Furthermore, though the generator model performances improve fast from 0 to 50 shots, the scores fluctuate largely. But after the first 50 shots, the improvements slow down, and the variances becomes much smaller. This implies that all the compared models are strong few shot learners, so that fine-tuning on large-scaled training data in the downstream tasks is unnecessary.

5.4 Discussion on the Differences Between Discriminator and Generator Models

The ineffectiveness of zero-shot Unifew and Crossfit are rooted in their generation nature. The original motivation of generation-based models is to resolve all kinds of NLP tasks, including both classification and generation. However, the universal classification task (i.e., tasks in this paper) are usually formulated as label picking from limited choices, while generation tasks aim to output human-readable sentences that match the input sentences – the target distributions for these 2 tasks are innately different. In the few-shot setting, fine-tuning with 10 more examples in the target task

I happily donate any covid vaccine dose which may be reserved for me to any person that is stupid enough to get one, or two, three, or four.									
mild	0.59	irony	0.48	happy	0.44	...	...	optimism	0.23
Guys it's OK. Delta says covid is over. IT'S OK NOW.									
mild	0.69	non-irony	0.48	irony	0.47	...	...	hate	0.10
The first patient who died of COVID in Kerala already had BP and cardiac issues, and he was 69. Bottomline : If we take precautions, we are still safe and can ensure others are safe too.									
optimism	0.51	mild	0.51	positive	0.43	...	...	hate	0.08
Could you imagine putting your faith into the narrative, getting jabs, getting sick from side effects (which is now being called the variant) and then being labeled an antivaxxer amidst this lie "only the unnvacinated are getting sick". They will use you up until there's nothing left.									
offensive	0.59	irony	0.58	mild	0.56	...	...	happy	0.25
... I don't see a monetary benefit. I don't see any professional benefit. Ask the people who believe what they are being told for an explanation because I don't see any.									
offensive	0.60	mild	0.54	irony	0.49	...	...	optimism	0.20
I can't do this anymore. I went from a house and 2 beautiful daughters and wife to homeless and left with literally nothing. ... They need to die painfully and even then they will never pay for their sins. All it takes it one moment in history for everything to change. You keep breaking men down to nothing. Those broken men will break you.									
offensive	0.80	negative	0.63	hate	0.59	...	...	optimism	0.10

Table 2: Case study of an unseen task. We use CONENTAIL in a zero-shot manner to analyze twitter and reddit sentiment during the Covid-Omicron surge. We pick 13 fine-grained sentiment labels and rank the labels by their similarity with the input sentence.

shifts the text generation distribution towards the label distribution, so the generated texts are more likely to be the labels, and this improves model performances. However, as the predictions are still in the large vocabulary space, they are likely to be altered by any disturbances. When using different support sets, the variances of the accuracy are far larger than that of the discriminator models. This also explains why Unifew performs better than Crossfit: the only difference between Unifew and Crossfit is that the input sentences of Unifew are appended with all possible label texts. By providing the generation process label hints, Unifew shifts its generation distribution towards label distribution and outperforms Crossfit. But the accuracy gap between Unifew and Crossfit drops from 15% to merely 0.7% while the number of shots increases from 0 to 10. As we stated before, Unifew performs better in the 0-shot setting because of its extra label hints. However, with an increase of shots, this advantage is diluted, resulting in a smaller performance difference between these two models.

5.5 A Case Study of Universal Classification

Consider a possible application scenario of universal classification: when dealing with new tasks and domains, especially related to newly emerged events, usually people only have the label names in hand. Based on this, we demonstrate a COVID-19 sentiment classification case study to show the universality of the proposed CONENTAIL model.

We use keywords to collect 50 sentences from Reddit and Twitter during the surge of the Omicron variant, then pick 13 fine-grained sentiment labels for this task: *positive, mild, negative, offensive, happy, anger, sad, hate, irony, non-offensive, non-irony, non-hate, optimism*. For each COVID-related query sentence, CONENTAIL model retrieves from all 13 possible labels and ranks them by similarity.

From the results Table 2 we observe that the model ranks the labels correctly most of the time. With antonyms paired with each other, such as *hate/non-hate* and *happy/sad*, our model successfully predicts the labels with only the label names, showing the polarity derived from the pairwise ranking are effective and reliable.

6 Conclusions

In this work, we study the universal classification problem, that leverages heterogeneous labeled datasets to benefit zero/few-shot learning in a new domain/task/dataset. We conduct systematic experiments on mainstream discriminators and generators models, thoroughly evaluate different models, reveal their innate properties of meta-task reformulation and supervised pretraining strategies. The results show that the generators with open-end prediction fail in zero-shot learning and the discriminators with a standard entailment meta-task hardly obtain a performance boost when more pretraining data is available. Our work provides a new angle for future researchers to explore universal NLP, and propose a new nested entailment meta-task and a supervised contrastive learning strategy, CONENTAIL, to make better use of widely available annotated datasets, and adapts to new datasets with limited resources.

Limitations

Although this paper aims to improve the universal generalization in the classification task, there are several limitations: (1) We do not compare with cloze-based models (Schick and Schütze, 2021a,b; Gao et al., 2020), because their templates are more complicated and hard to be reproduced with our current datasets. (2) We do not consider structural classification tasks, such as Named Entity Recognition and Relation Extraction. (3) We only take classification datasets into account because our implementation is restricted by huggingface datasets and human-curated templates. We plan to extend our framework to more datasets in the future. (4) Due to the constraints from the templates and datasets, the class number of each test set is below 10. We plan to extend our framework to more labels in the future work. (5) The compatibility of knowledge in similar tasks is assumed, but this assumption may not hold true due to varying annotation standards across datasets. For instance, MRPC and QQP are both paraphrase identification tasks, but MRPC uses hard example mining techniques, resulting in longer and more sophisticated sentences than QQP. (6) The current study is limited to English datasets and can be extended to multiple languages in the future by using multilingual PLMs and pretraining datasets.

Acknowledgments

We thank Qianying Liu for her valuable discussion.

References

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q Tran, Dara Bahri, Jianmo Ni, et al. 2021. Ext5: Towards extreme multitask scaling for transfer learning. *arXiv preprint arXiv:2111.10952*.

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. Flex: Unifying evaluation for few-shot nlp. *arXiv preprint arXiv:2107.07170*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

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

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. All nlp tasks are generation tasks: A general pretraining framework. *arXiv preprint arXiv:2103.10360*.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. 2019. Multi-class classification without multi-class labels. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Shankar Iyer, Nikhil Dandekar, and Kornel Csernai. 2017. First quora dataset release: Question pairs.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020a. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020b. Unifiedqa: Crossing format boundaries with a single qa system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1896–1907.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1378–1387, New York, New York, USA. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Tingting Ma, Jin-ge Yao, Chin-Yew Lin, and Tiejun Zhao. 2021. Issues with entailment-based zero-shot text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 786–796.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, page 115–124, USA. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Congying Xia, Wenpeng Yin, Yihao Feng, and Philip Yu. 2021. Incremental few-shot text classification with multi-round new classes: Formulation, dataset and system. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1351–1360, Online. Association for Computational Linguistics.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021a. Crossfit: A few-shot learning challenge for cross-task generalization in nlp. *arXiv preprint arXiv:2104.08835*.

Seonghyeon Ye, Jiseon Kim, and Alice Oh. 2021b. Efficient contrastive learning via novel data augmentation and curriculum learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1832–1838, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3905–3914.

Wenpeng Yin, Nazneen Fatema Rajani, Dragomir Radev, Richard Socher, and Caiming Xiong. 2020. Universal natural language processing with limited annotations: Try few-shot textual entailment as a start. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8229–8239.

Chenyu You, Nuo Chen, and Yuexian Zou. 2021. Self-supervised contrastive cross-modality representation learning for spoken question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 28–39, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Tony Z Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *arXiv preprint arXiv:2102.09690*.

A Hyperparameters and Implementation Details

Unifew and Crossfit, as generative models, use BART$_{\text{base}}$ (Lewis et al., 2020) as the backbone language model. In the supervised pretraining, we use AdamW optimizer (Loshchilov and Hutter, 2017) with learning rate 3e-5, warm-up ratio 0.6% and linear decay. In the meta-testing, we use the same hyperparameters and train 400 epochs for finetuning.

EFL and Entail2, as discriminator models, use BERT$_{\text{base}}$ (Devlin et al., 2019) as the backbone language model. In the supervised pretraining, we use AdamW optimizer (Loshchilov and Hutter, 2017) with learning rate 1e-5, warm-up ratio 6% and linear decay. In the meta-testing, we use the same hyperparameters and train 10 epochs for finetuning.

All the compared models use the same templates (map the input to the text) and the same verbalizers (map the label to the text) from the Crossfit paper (Ye et al., 2021a), as they covered more classification datasets than other frameworks. Note that the choices of template/verbalizer could cause large variance in performance (Zhao et al., 2021), and the effectiveness of Crossfit template/verbalizer had not been fully studied.

We use two NVIDIA A5000 for our experiments. The supervised pretraining takes 3 days and the evaluation takes 1 week for all the compared baselines.

B Details about Task Partition

Datasets	Labels	Test sentences	Citation
glue-cola	2	1043	(Warstadt et al., 2018)
glue-qqp	2	40430	(Iyer et al., 2017)
glue-sst2	2	872	(Socher et al., 2013)
glue-mrpc	2	408	(Dolan and Brockett, 2005)
scitail	2	1304	(Khot et al., 2018)
amazon_polarity	2	1000	(Zhang et al., 2015a)
ag_news	4	7600	(Zhang et al., 2015b)
rotten_tomatoes	2	1066	(Pang and Lee, 2005)
hate_speech_offensive	3	4957	(Davidson et al., 2017)

Table 3: The statistics of the 9 test data.

```
{
    "Suprevised_pretraining": ["tweet_eval-stance_hillary", "ethos-sexual_orientation", "climate_fever
        ", "hate_speech18", "tweet_eval-emotion", "hatexplain", "ethos-race", "emotion", "superglue-
        rte", "discovery", "anli", "wiki_auto", "scicite", "financial_phrasebank", "sms_spam", "
        kilt_fever", "tweet_eval-stance_climate", "medical_questions_pairs", "tweet_eval-
        stance_feminist", "ethos-directed_vs_generalized", "glue-wnli", "health_fact", "liar", "
        yahoo_answers_topics", "ethos-religion", "circa", "ethos-disability", "emo", "tweet_eval-hate
        ", "tweet_eval-sentiment", "superglue-wic", "tweet_eval-emoji", "glue-qnli", "ade_corpus_v2-
        classification", "ethos-national_origin", "dbpedia_14", "poem_sentiment", "yelp_polarity", "
        tweet_eval-stance_atheism", "onestop_english", "glue-rte", "wiki_qa", "ethos-gender", "
        superglue-wsc", "tweet_eval-stance_abortion", "paws", "tweet_eval-offensive"],
    "meta_test": ["glue-cola", "glue-qqp", "glue-sst2", "glue-mrpc", "scitail", "amazon_polarity", "
        ag_news", "rotten_tomatoes", "hate_speech_offensive"]
}
```

C Additional results

Method	**CoLA**	**QQP**	**SST-2**	**MRPC**	**SCITAIL**	**Amazon**	**AGNews**	**rotten_tomatoes**	**hate_speech**	**AVG**
Random-guess	50.5	49.8	49.9	50.0	49.8	49.9	24.0	46.8	34.1	44.9
PLM + 0-*shot*										
Crossfit	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Unifew	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
EFL	**62.6**	48.1	44.7	**58.6**	44.8	42.3	**31.5**	45.4	19.6	44.2
CONENTAIL	32.7	**63.2**	**57.0**	31.6	**50.4**	**55.9**	25.4	**53.4**	**78.3**	**49.8**
PLM + *Supervised Pretraining* + 0-*shot*										
Crossfit	0.0	0.0	33.4	0.0	0.2	9.9	0.0	59.9	0.0	11.5
Unifew	0.0	0.0	60.6	0.0	48.4	63.7	8.0	57.4	0.0	26.5
EFL	46.2	**60.5**	79.1	33.1	47.2	71.9	60.8	72.5	12.7	53.8
CONENTAIL	**58.5**	45.3	**83.0**	**58.1**	**68.7**	**89.7**	**52.8**	**78.1**	**34.3**	**63.2**
PLM + 10-*shot fine-tuning*										
Crossfit	55.3 ±5.0	53.4 ±9.8	**81.2** ±8.9	**60.0** ±11.1	58.8 ±5.4	87.9 ±6.1	83.7 ±6.6	65.0 ±23.5	42.8 ±14.4	65.3
Unifew	49.0 ±4.9	**60.4** ±6.0	71.2 ±11.5	57.7 ±6.3	53.4 ±2.4	**88.8** ±3.6	**86.5** ±1.8	**73.4** ±9.5	34.9 ±6.8	**63.9**
EFL	**63.7** ±0.2	**60.4** ±0.2	79.5 ±0.2	32.3 ±0.4	46.7 ±1.1	72.0 ±0.0	62.3 ±0.6	72.4 ±0.2	13.8 ±0.6	55.9
CONENTAIL	38.6 ±4.4	55.6 ±3.5	62.4 ±3.4	46.1 ±2.4	**59.1** ±2.7	64.0 ±1.6	57.4 ±2.3	61.8 ±2.0	**58.6** ±11.0	55.9
PLM + *Supervised Pretraining* + 10-*shot fine-tuning*										
Crossfit	25.7 ±25.1	13.6 ±18.2	80.6 ±4.0	28.1 ±28.7	53.9 ±11.7	85.2 ±6.5	31.7 ±17.7	75.8 ±1.2	2.4 ±3.2	44.1
Unifew	46.6 ±12.4	37.5 ±30.6	60.9 ±1.4	23.8 ±24.9	51.0 ±1.8	63.9 ±2.9	52.2 ±16.1	57.9 ±1.5	9.8 ±6.5	44.8
EFL	46.1 ±0.0	**60.4** ±0.0	79.1 ±0.1	33.1 ±0.0	47.2 ±0.1	72.0 ±0.0	60.9 ±0.0	72.5 ±0.0	12.9 ±0.1	53.8
CONENTAIL	**60.5** ±0.6	51.8 ±1.9	**83.2** ±0.2	**69.9** ±0.9	**71.0** ±0.9	**89.4** ±0.1	**70.3** ±2.1	**78.7** ±0.2	**44.7** ±2.2	**68.8**

Table 4: The complete table of the main result.

Guide the Learner: Controlling Product of Experts Debiasing Method Based on Token Attribution Similarities

Ali Modarressi[1⋆] **Hossein Amirkhani**[2] **Mohammad Taher Pilehvar**[3]
[1] Center for Information and Language Processing, LMU Munich
[2] Computer and Information Technology Department, University of Qom, Iran
[3] Tehran Institute for Advanced Studies, Khatam University, Iran
amodaresi@cis.lmu.de
amirkhani@qom.ac.ir
mp792@cam.ac.uk

Abstract

Several proposals have been put forward in recent years for improving out-of-distribution (OOD) performance through mitigating dataset biases. A popular workaround is to train a robust model by re-weighting training examples based on a secondary biased model. Here, the underlying assumption is that the biased model resorts to shortcut features. Hence, those training examples that are correctly predicted by the biased model are flagged as being biased and are down-weighted during the training of the main model. However, assessing the importance of an instance merely based on the predictions of the biased model may be too naive. It is possible that the prediction of the main model can be derived from another decision-making process that is distinct from the behavior of the biased model. To circumvent this, we introduce a fine-tuning strategy that incorporates the similarity between the main and biased model attribution scores in a Product of Experts (PoE) loss function to further improve OOD performance. With experiments conducted on natural language inference and fact verification benchmarks, we show that our method improves OOD results while maintaining in-distribution (ID) performance.[1]

1 Introduction

Overfitting to the training data is a big obstacle in learning patterns that generalize to unseen data. Traditionally, this is diagnosed by monitoring the performance of a trained model on an in-distribution (ID) test set. However, a bigger challenge is when both the training and test data have the same non-generalizable patterns, emerged as spurious correlations between input features and output labels (Gardner et al., 2021). For instance, in the natural language inference (NLI) task, it is shown that the occurrence of some task-neutral words, like a negation in hypothesis, is highly correlated with a specific class (Gururangan et al., 2018). While high-capacity models can learn a generalized distribution of labels from the inputs, they are prone to spurious patterns, also known as dataset biases (Clark et al., 2019; He et al., 2019). A model could exploit these biases during fine-tuning, leading to a model that achieve high ID performance, while it is highly fragile in out-of-distribution (OOD) settings (Schuster et al., 2019; McCoy et al., 2020).

Besides trying to prevent these non-generalizable artifacts from entering the dataset (Liu et al., 2022), it is reasonable to seek for more robust learning methods. This has been the basis for a multitude of research works that encourage models to rely on truly generalizable patterns. Most of these methods are based on the assumption that the learning method will inevitably exploit biases if they are present in a training example (Clark et al., 2019; Sanh et al., 2020; Mahabadi et al., 2020; Utama et al., 2020; Ghaddar et al., 2021). Therefore, they discourage the main model from paying much attention to the examples which are correctly classified by a biased model. Recently, it is shown that this assumption is questionable in the way that for a significant number of cases, the main model does not follow the biased model in treating biased examples (Amirkhani and Pilehvar, 2021). Therefore, depriving the training algorithm from the examples which are detected to be biased is a waste of training data.

In this paper, we propose an alternative way to discard biased examples. Instead of considering the mere prediction of the biased model, we monitor the way the model processes each example by computing its attribution scores over the input tokens. With the resulting scores, we adjust the proportion of the loss function that is a cross-entropy loss (CE) versus a Product of Experts loss (PoE). If

---

[⋆] Work done as a Master's student at Iran University of Science and Technology (IUST).

[1] Our code is freely available at: `https://github.com/amodaresi/Debias_w_Saliencies`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1954–1959
May 2-6, 2023 ©2023 Association for Computational Linguistics

the attribution scores are similar between the main and biased models, the loss becomes a PoE loss where a correct prediction from the biased model down-weights the contribution of the corresponding example. In contrast, dissimilarity between the scores suggests a different behaviour from the biased model and leads to a CE loss that only considers the main model's prediction. Experiments on natural language inference and fact verification demonstrate that our method significantly outperforms previous approaches in terms of OOD performance while preserving its ID performance.

2 Methodology

In this section we explain our debiasing solution for a given classification task with dataset $\mathcal{D}\{\boldsymbol{x_i}, y_i\}_{i=1}^{N}$. For the i^{th} training example, we denote the input sequence of tokens as $\boldsymbol{x_i}$ and the gold label as $y_i \in \{1, 2, ..., Y\}$ where Y is the number of classes. Specifically, the goal is to train a model—we denote it as the *main model*—on the training dataset ($\mathcal{D}$) so that it can also perform well on OOD datasets that do not necessarily share the same biases. Following other well-known paradigms for identifying biases in the dataset, we first need to design or train a *biased model* that employs shortcut methods to complete the task (Mahabadi et al., 2020; Sanh et al., 2020; Utama et al., 2020).

In what follows, we will first review the Product of Experts (PoE) method for bias reduction, which is based solely on the outputs of the main and biased models. Then we present our contribution to developing a novel debiasing method that takes into account not only the models' outputs but also their attribution scores.

Product of Experts. In the original Product of Experts (PoE) solution, the loss is based on combining the predictions of the main and biased model: $\sigma(f_B(\boldsymbol{x_i^b}))$ & $\sigma(f_M(\boldsymbol{x_i}))$ (Hinton, 2002; Mahabadi et al., 2020). The summation of the log-softmaxes ($\sigma(.)$) of both models combine the distributions so that the main would focus less on biased examples:

$$f_C(\boldsymbol{x_i}; \boldsymbol{x_i^b}) = \log(\sigma(f_B(\boldsymbol{x_i^b}))) + \log(\sigma(f_M(\boldsymbol{x_i})))$$

The PoE loss is the cross-entropy loss over the summation shown above:

$$\mathcal{L}_{\text{PoE}}(\theta_M; \theta_B) = -\log(\sigma(f_C^{y_i}(\boldsymbol{x_i}; \boldsymbol{x_i^b}))) \quad (1)$$

While PoE does produce promising results, training the main model only based on the biased model's output can undermine some instances that could have been helpful and non-biased. As stated in Amirkhani and Pilehvar (2021), correct prediction of an instance by the biased model does not necessarily imply that the instance is biased, as the behaviour of the two models might differ.

2.1 PoE with Saliencies

To determine how similarly the main model and bias model behave, we must compute the attribution scores of the input tokens for both models. Therefore, after fine-tuning the biased model, we compute the saliencies $\boldsymbol{S}$ using a gradient-based approach according to the *gradient*×*input* method (Kindermans et al., 2016):

$$\boldsymbol{S}_i = \left\| \frac{\partial y_i}{\partial \boldsymbol{h}_i^0} \odot \boldsymbol{h}_i^0 \right\|_2 \quad (2)$$

This method is based on the gradient of the logit of the output prediction y_i with respect to the input embeddings $\boldsymbol{h}_i^0$. We also obtain the main model saliencies during training. By computing the saliencies for both bias and main models, it is possible to estimate the contribution of each input token in a training instance to both models' predictions. Therefore, we can compute the inter-model similarity—between the saliencies of the two models; for instance, using the cosine similarity metric:

$$\rho = \frac{\boldsymbol{S}_{\text{Main}} \cdot \boldsymbol{S}_{\text{Biased}}}{\|\boldsymbol{S}_{\text{Main}}\| \, \|\boldsymbol{S}_{\text{Biased}}\|} \quad (3)$$

Since the saliencies defined in 2 can only have positive values, ρ will always be a number between 0 and 1. This would provide a complementary metric that shows how similar the two models behave on a specific example. Therefore, we can modify the original PoE loss function that only incorporates the output predictions and include the inter-model similarity in the debiasing loss function:

$$\begin{aligned}\mathcal{L}_{\text{PoE+Sals}}(\theta_M; \theta_B) = \rho^* \mathcal{L}_{\text{PoE}}(\theta_M; \theta_B) \\ +\alpha(1-\rho^*)\mathcal{L}_{\text{CE}}(\theta_M)\end{aligned} \quad (4)$$

where $\mathcal{L}_{\text{CE}}(\theta_M)$ is the cross-entropy (CE) loss on the main model. We also define ρ^* that adjusts the inter-model similarity (ρ) based on the PoE loss using the following formulation:

$$\begin{aligned}\rho^* &= \rho^{\exp(-\beta \mathcal{L}_{\text{PoE}}(\theta_M; \theta_B))} \\ &= \rho^{\exp(\beta \log(\sigma(f_C^{y_i}(\boldsymbol{x_i}; \boldsymbol{x_i^b}))))} \\ &= \rho^{\sigma(f_C^{y_i}(\boldsymbol{x_i}; \boldsymbol{x_i^b}))^{\beta}}\end{aligned} \quad (5)$$

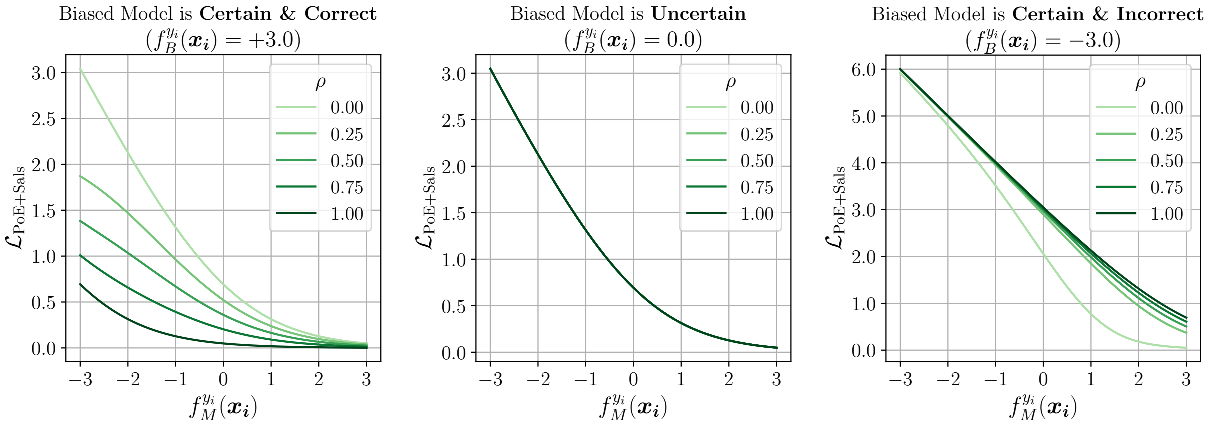


Figure 1: A visualization of our proposed loss function ($\mathcal{L}_{\text{PoE+Sals}}$) based on the main and biased models' prediction ($f_M^{y_i}(\boldsymbol{x_i})$ and $f_B^{y_i}(\boldsymbol{x_i})$) and the inter-model similarity (ρ). For a given input, when the biased model is correct and certain (left), the loss will be upweighted when the biased model differs from the main model in terms of saliency scores. However if the biased model returns a certain but incorrect prediction (right), the loss is downweighted if the main model is dissimilar and also returns a correct prediction. In an uncertain case (middle), the biased model does not affect the PoE loss (Sanh et al., 2020). In this case, the loss is converted to a CE loss based on the output of the main model.[3]

Where β is a positive hyperparameter for adjusting the combined prediction's ($\sigma(f_C^{y_i}(\boldsymbol{x_i}; \boldsymbol{x_i^b}))$) impact.

The intuition behind this adjustment is to upweight and increase the loss for an example where the main and biased models agree on the correct label but show a different behaviour in terms of token attribution scores (a.k.a. less inter-model saliency similarity scores, $0 < \rho < 0.5$). As an extreme case, if the biased model correctly classifies an example with a high output score ($\sigma(f_B^{y_i}(\boldsymbol{x_i^b})) \approx 1$), the main model's prediction would be ineffective in the original PoE loss, since the PoE model output probability is as follows:

$$\sigma(f_C^{y_i}(\boldsymbol{x_i}; \boldsymbol{x_i^b})) = \frac{\sigma(f_B^{y_i}(\boldsymbol{x_i^b}))\sigma(f_M^{y_i}(\boldsymbol{x_i}))}{\sum_{k=1}^{Y} \sigma(f_B^{k}(\boldsymbol{x_i^b}))\sigma(f_M^{k}(\boldsymbol{x_i}))}$$

Therefore, in the adjusted similarity stated in Eq. 5, the exponent would be approximately equal to one, which makes the adjusted similarity equal to the original cosine similarity. As a result, according to Eq. 4, if the models exhibit dissimilar explanations for their prediction ($\rho \approx 0$), the loss tends to be a cross-entropy loss[2] than a PoE loss. On the other hand, in case both models behave similarly on a given training example, i.e., $\rho \approx 1$, the example can be considered as one containing bias. Thus, the PoE loss renders this example as being less impactful during training. The left and right plots of Figure 1 respectively demonstrate[3] that having a higher similarity results in a loss function that is down- or up-weighted depending on the correctness of a certain biased model. Also with an uncertain biased model, the loss in Eq. 1 converts to a CE loss which is only based on the output of the main model.

3 Experiments

In this section, we will introduce the datasets, explain the experimental setup, and then demonstrate the results of our method.

3.1 Datasets

The experiments were carried out on two types of common NLU classification tasks: Natural Language Inference (NLI) and Fact Verification. For NLI, we used MNLI (Williams et al., 2018) as our in-distribution data and HANS (McCoy et al., 2020) for OOD evaluation. Note that the Matched development set is used for the ID evaluation in MNLI. In addition, because HANS has only two labels, entailment and not entailment, we consider outputs that are predicted contradictions or neutral as not entailment. For Fact Verification, we

---

[2] Weighted with α as a modulating hyperparameter ($0 < \alpha$).

[3] We simplify the plots by assuming the output logits of all classes except the gold label are zero for both models ($f_{B/M}^{y \neq y_i}(\boldsymbol{x_i}) = 0$). In addition, these figures are plotted based on $\alpha = 1$ and $\beta = 1$.

Model	MNLI		FEVER	
	Dev. (Matched)	HANS	Dev.	Sym.V1
BERT-base	**84.71±0.21**	62.85±2.69	85.19±0.37	56.51±1.41
DFL$_{e2e}$ (Mahabadi et al., 2020)	83.91±0.20	66.10±2.81	80.48±0.87	65.13±1.52
PoE$_{e2e}$ (Mahabadi et al., 2020)	84.10±0.19	63.63±1.90	84.43±0.87	64.28±1.52
PoE (Sanh et al., 2020)	81.10±0.41	68.04±1.51	80.08±0.94	62.72±2.99
PoE+CE (Sanh et al., 2020)	83.34±0.33	66.56±0.66	85.29±1.25	62.55±2.46
PoE w. Attribution Similarity	82.81±0.26	**68.06±0.66**	85.48±1.09	**66.97±2.00**
w/o Attribution Similarity	-	-	**86.09±0.72**	65.18±2.25

Table 1: The mean and standard deviation of the accuracy scores of multiple debiasing strategies applied to NLI and fact verification. The maximum values are highlighted in bold. It is important to note that the results of the methods that are used for comparison are not the results that were reported by the methods themselves but rather the scores that were obtained by replicating their implementation[5].

used FEVER (Thorne et al., 2018) and FEVER Symmetric-V1 (Schuster et al., 2019) for ID and OOD, respectively.[4]

3.2 Setup

In all the experiments, we used *BERT-base-uncased* (Devlin et al., 2019) as the main model to allow a fair comparison against other debiasing methods. However, the biased model differs depending on the task.

We used the TinyBERT model (Turc et al., 2019) for the NLI task, since its limited capacity makes it extremely susceptible to biased features in training examples (Sanh et al., 2020). For fact verification, a full 12-layer BERT-base model similar to the main model is fine-tuned using only the claim sentences from the training data. In both biased models, we compute and save their prediction logits and attribution scores across the entire training dataset so that they can be used as a frozen model during training. However, since the fact verification biased model is only trained on the claim sentences, the attribution scores are only calculated for the claim portion. Therefore, the similarities computed during the training procedure are limited to the claim segment alone.

For the generic fine-tuning hyperparameters, we adopted Sanh et al. (2020) recommendations: 3 epochs of training, a batch size of 32, an Adam optimizer (Kingma and Ba, 2015) with warmup and linear decay in its learning rate schedule, and a peak learning rate of 3e-5 or 2e-5 for MNLI or FEVER, respectively. But as for the specific hyperparameters in our approach, using sweeping over $\alpha \in \{0.01, 0.1, 0.2, 0.3, 0.5, 1.0\}$ and $\beta \in \{0.1, 0.3, 0.5, 1.0\}$, we set $\alpha = 1.0$, $\beta = 1.0$ for MNLI and $\alpha = 0.3$, $\beta = 0.1$ for training on FEVER.

All experiments were implemented using the HuggingFace Transformers library (Wolf et al., 2020) and performed on an RTX 3070 GPU machine. The results are the average of six runs with different seeds.

3.3 Results

The results of various debiasing techniques applied to the previously mentioned benchmarks are shown in Table Table 1. The baseline is fine-tuning the backbone model (BERT-base-uncased) with the commonly used cross-entropy loss, which provides high ID performance but lacks OOD. We also include four approaches from two different studies for additional comparison[5]. Two End-to-End solutions from Mahabadi et al. (2020), one utilizing Debiased Focal Loss (DFL) and the other employing PoE. The PoE and PoE+CE (PoE with a static weighted CE loss added) methods from Sanh et al. (2020) are similar to our approach of having a bi-

---

[4] In both FEVER and FEVER-Symmetric, replacement tokens are used in place of parentheses and brackets (e.g. "]" → "-RSB-"). This causes the BERT tokenizer to split the specified tokens into multiple segments, as there are no tokens in the BERT vocabulary that correspond to the replacement tokens. Therefore, we replace these tokens with the punctuation that corresponds to them so that BERT can tokenize the inputs with less undesirable segmentation. We apply this modification to other approaches that we have evaluated and compared.

[5] We could have included the compared methods' results from their respective papers as well as results from other approaches. However, in our empirical results we observed significant discrepancies between the results obtained from their source code and those reported. In this paper, we intend to report only the reproduced results, which is why the number of compared methods is limited.

ased model that is frozen, but they only rely on the predictions of the biased model and not the inter-model similarities.

It can be observed that the OOD performance of our method outperforms those of other approaches, without or with minimal loss of ID accuracy. On FEVER, our strategy improves SymV1 by nearly 2% while also slightly improving the ID performance. In MNLI, our strategy improves HANS to achieve high scores comparable to Sanh et al. (2020) PoE-only solution. However, the ID performance of Sanh et al. (2020) PoE-only falls short. While maintaining the same level of OOD performance, our solution achieves a minimum MNLI-m dev score that is acceptable.

As an ablation study, we also trained the procedure without using similarity values, resulting in a PoE+CE solution. Since we used a different biased model in FEVER than Sanh et al. (2020), we reported its results using the claim-only biased model configuration. However, we omitted MNLI results from our report because they are identical to Sanh et al. (2020) PoE+CE configuration. Having a claim-only biased model for FEVER rather than a TinyBERT model yields a substantial increase in OOD, as can be seen in the results. As a result, we can observe that even though adding a weighted CE to a PoE loss yields improvements in terms of ID performance in particular, the similarities could push even further and also improve OOD performance.

Another observation is the large standard deviations in OOD accuracy of all approaches, which is why we chose to execute 6 runs as also suggested by Sanh et al. (2020). Even so, it should be noted that our method still has a relatively low variance in HANS.

4 Conclusions

In this paper, we introduced a strategy for improving OOD performance by incorporating the similarity values between the token attribution scores of main and biased models into a Product of Experts (PoE) loss function. The gist of this approach is that it takes into account the decision-making process in addition to the output predictions of the main and biased models (which are often taken as the primary signal). By comparing our method to multiple recent debiasing methods on two widely-used NLU tasks, we demonstrated that our method improves performance on out-of-distribution data while preserving performance on in-distribution data. Future work could include exploring other loss formulations with Attribution Similarities that may produce better results. Additionally, it may be beneficial to investigate methods to reduce the variability of results, which is present in the majority of debiasing solutions.

Limitations

Due to the high variance in the accuracy scores, the main limitation of this approach (as well as the majority of other debiasing approaches) is the large number of seeds used to tune the hyperparameters. As a result, any type of tuning necessitates a multi-seed run, which requires multiple GPUs or many hours of training. Because of this constraint, we omitted working on larger scaled models or other types of PLMs.

References

Hossein Amirkhani and Mohammad Taher Pilehvar. 2021. Don't discard all the biased instances: Investigating a core assumption in dataset bias mitigation techniques. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4720–4728.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082.

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

Matt Gardner, William Merrill, Jesse Dodge, Matthew E Peters, Alexis Ross, Sameer Singh, and Noah A Smith. 2021. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813.

Abbas Ghaddar, Philippe Langlais, Mehdi Rezagholizadeh, and Ahmad Rashid. 2021. End-to-end self-debiasing framework for robust nlu training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1923–1929.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142, Hong Kong, China. Association for Computational Linguistics.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

Pieter-Jan Kindermans, Kristof Schütt, Klaus-Robert Müller, and Sven Dähne. 2016. Investigating the influence of noise and distractors on the interpretation of neural networks. *arXiv*, abs/1611.07270.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Alisa Liu, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. 2022. Wanli: Worker and ai collaboration for natural language inference dataset creation. *arXiv preprint arXiv:2201.05955*.

Rabeeh Karimi Mahabadi, James Henderson, et al. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, CONF.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2020. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pages 3428–3448. Association for Computational Linguistics (ACL).

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. 2020. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *International Conference on Learning Representations*.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Towards debiasing nlu models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7597–7610.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Task and Sentiment Adaptation for Appraisal Tagging

Lin Tian
RMIT University, Australia
`lin.tian2@student.rmit.edu.au`

Xiuzhen Zhang*
RMIT University, Australia
`xiuzhen.zhang@rmit.edu.au`

Maria Myung-Hee Kim
Defence Science and Technology Group
Australia
`myung.kim@defence.gov.au`

Jennifer Biggs
Defence Science and Technology Group
Australia
`jennifer.biggs@defence.gov.au`

Abstract

Sentiment analysis and opinion mining of the opinion-bearing text are important tasks in NLP. The Appraisal framework in systemic functional linguistics is a theory for analysing the linguistic patterns for expressing emotion and opinion. Manual annotation of appraisals however, requires linguistic expertise, and is costly and time-consuming. In this paper, we study how to automatically identify and tag appraisal text segments. We formulate the problem as a sequence tagging problem and propose a novel approach, Adaptive Appraisal (A^2), which employs task and sentiment adapters on pre-trained language models for sequence appraisal tagging. Experiments on user comments, blogs and microblogs show that A^2 outperforms baseline models and achieves good performance for cross-domain and cross-lingual settings. Source code for A^2 is available at: https://github.com/ltian678/AA-code.git

1 Introduction

With the development of the Web technology, opinion-bearing user generated texts such as reviews, users comments, blogs and microblogs are widespread. Sentiment analysis and opinion mining on such texts are prominent NLP tasks that have attracted extensive research studies in the literature (Liu, 2022). On the other hand, the Appraisal framework (Martin and White, 2003) is a systemic functional linguistic theory describing how language is used by writers or speakers to express emotion and opinion. The Appraisal framework consists of three subsystems: 1) Attitude, which includes personal emotion, judgement and evaluation of entities; 2) Engagement, which regards one's own opinions or with respect to others; and 3) Graduation, which describes strength of the attitude and engagement expressed.

Appraisal annotated resources have been used for deeper sentiment and emotion analysis (Whitelaw et al., 2005) than simple sentiment classification, but building such resources manually requires significant linguistic expertise and is time-consuming (Read and Carroll, 2012; Kolhatkar et al., 2020). Automated appraisal tagging would be extremely beneficial to support annotation and analysis efforts by expert linguists. To our best knowledge, the only publicly available appraisal annotated corpus is the Simon Fraser University Opinion and Comments Corpus (SOCC) (Kolhatkar et al., 2020)[1], which is based on news comments. In creating the SOCC corpus, 663,173 user comments were collected, and expert linguists manually annotated 1,043 comments.

To the best of our knowledge, there has been no prior work leveraging machine learning for automatically tagging appraisals. We fill this gap in this paper. We especially target domains with voluminous opinionated texts but zero or very limited appraisal annotation resources, such as blogs and microblogs. Our research focuses on sequence tagging for cross-domain and cross-lingual texts with low resources. In the literature, various approaches for sequence tagging tasks have been reported, including transfer learning (Lee et al., 2018a), few-shot learning (Hofer et al., 2018) and multi-task learning (Changpinyo et al., 2018; Kann et al., 2018; Liu et al., 2018). However, none of the existing studies on sequence tagging consider zero-shot cross-domain or cross-lingual settings.

We propose a model for automatic Appraisal tagging. Our model A^2, namely Adaptive Appraisal, utilises joint task and sentiment adapters based on pre-trained language models for tagging appraisal segments in text sequences. Our model leverages the adapter-based transfer learning framework for cross-domain and cross-lingual appraisal tagging. Based on the pre-trained language model, we propose the task adapter for appraisal tagging across different domains. For instance, the language that

---

*Corresponding author

[1] `https://github.com/sfu-discourse-lab/SOCC`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1960–1970
May 2-6, 2023 ©2023 Association for Computational Linguistics

short Microblog posts use for expressing appraisals is very different from the language of long blog posts. It is therefore necessary to enable the adaptive ability from one domain to different domains. We further propose the sentiment adapter to capture the sentiment knowledge for appraisal tagging, capitalised on the strong correlation between appraisals and sentiments. Rather than employing appraisal annotations for sentiment analysis as in existing studies (Whitelaw et al., 2005), we propose to leverage the rich sentiment analysis resources for automatic appraisal tagging.

In this study, we seek to answer the following research questions.

- RQ1: Can we leverage the adapter-based framework for the within-domain appraisal tagging task?
- RQ2: Can the sentiment knowledge fused adapter improve the adapter-based framework for cross-domain appraisal tagging?
- RQ3: Can the sentiment knowledge fused adapter improve the adapter-based framework for the cross-lingual appraisal tagging task?

To summarise, our contributions are twofold: (1) we propose an adapter-based framework for appraisal tagging; and (2) we propose task and sentiment adapters to further enhance the framework for cross-domain and cross-lingual generalisation ability.

2 Related Work

Our work is related to sequence tagging – such as named entiy recognition, semantic role labeling, where token sequences in the input text are tagged with class labels. Our work is especially related to the task of sequence tagging with low resources for training. In the literature, to address the issue of low resources, sequence taggers based on transfer learning (Lee et al., 2018a), few-shot learning (Hofer et al., 2018) and multi-task learning (Changpinyo et al., 2018; Kann et al., 2018; Liu et al., 2018) have been reported. Our research falls under transfer learning.

Various transfer learning strategies and techniques have been proposed for NLP tasks addressing the issue of scarce labelled data. Early transfer learning algorithms have addressed the target domain data scarcity problem and boosted the model‘s generalisation ability via learning domain-agnostic knowledge for transfer (Kim et al., 2015; Lee et al., 2018b). Modern pre-trained language models (e.g. BERT) have achieved the state-of-the-art performance for a range of Natural Language Processing (NLP) tasks via transfer learning by fine-tuning parameters for different tasks (Devlin et al., 2019; Radford et al., 2019; Liu et al., 2019; Li et al., 2019; Yin et al., 2020). Apart from cross-domain, cross-lingual transfer learning has also been investigated, in particular for part-of-speech tagging and dependency parsing (Ruder et al., 2019). Algorithms have been proposed for transfer learning tasks of low resource languages (Kim et al., 2017; Schuster et al., 2019). All these transfer learning frameworks require fine-tuning parameters of the full model to achieve knowledge transfer, which limits the capacity for models to adapt to many target domains.

Adapter-based transfer is a recently proposed parameter-efficient transfer learning mechanism for adapting a pre-trained model to a target task without fine-tuning all parameters. Adapter modules was originated from computer vision, to control the convolutions and adapt models to multiple domains (Rebuffi et al., 2017). Then, in NLP application, adapters have been widely used for quick adaption in combination with existing large language models to new tasks (Houlsby et al., 2019) and avoiding catastrophic forgetting issues (McCloskey and Cohen, 1989). Üstün et al. (2020) generated adapter parameters from language embeddings for multilingual dependency parsing. Pfeiffer et al. (2021) combined the information stored in multiple adapters for most robust transfer learning between monolingual tasks. As adapter modules have been proved effective for efficient transfer learning with large language models, we propose task and sentiment adapters for the cross-domain and cross-lingual appraisal tagging tasks.

3 The Appraisal Framework

Following the Systemic Functional Linguistics (SFL) theory (Eggins, 2004), the Appraisal framework is a theory describing the linguistic patterns for authors to express emotion and opinion. The Appraisal framework consists of three semantic systems including Attitude, Graduation and Engagement. Attitude is divided into three sub-systems: *Affect*, *Judgement* and *Appreciation*. *Affect* deals with a person's emotional reactions (e.g. happy,

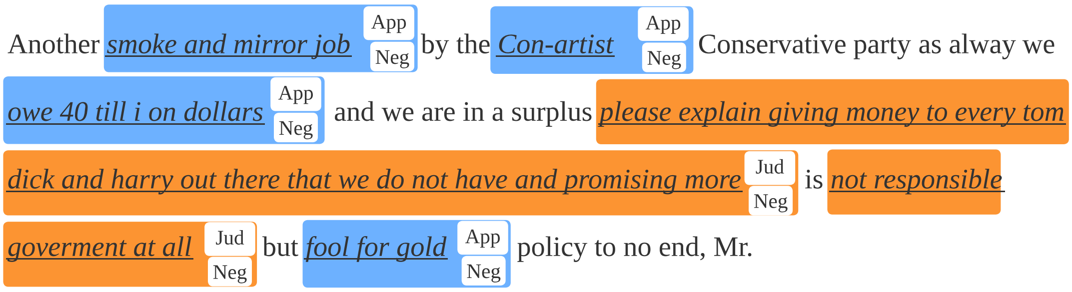

Another smoke and mirror job App Neg by the Con-artist App Neg Conservative party as alway we owe 40 till i on dollars App Neg and we are in a surplus please explain giving money to every tom dick and harry out there that we do not have and promising more Jud Neg is not responsible goverment at all Jud Neg but fool for gold App Neg policy to no end, Mr.

Figure 1: Sample appraisal annotations from SOCC dataset ("App" = Appreciation, "Jud" = Judgement, "Neg" = Negative)

confident), *Judgement* deals with assessing people's behaviour (e.g. powerful, truthful), and *Appreciation* deals with constructing the value of things (e.g. fascinating, exciting). In this work, we focus on the three sub-systems of Attitude and their Polarity (*Positive*, *Negative* and *Neutral*). Figure 1 shows an example for appraisal annotations from the SOCC dataset (Kolhatkar et al., 2020); each span is labelled with an attitude and its polarity.

It should be noted that the linguistic Appraisal framework in this study is different from the Appraisal theory of emotion (Ellsworth and Smith, 1988). The Appraisal theory of emotion (Ellsworth and Smith, 1988) describes that emotions are the result of the way in which people appraise or evaluate events and situations in terms of their relevance and significance to their goals, needs and values. According to this theory, emotions are generated by the way in which people appraise events and situations, and the specific emotion that is generated depends on the type of appraisal that is made. The functional linguistic Appraisal framework (Oteíza, 2017) on the other hand, describes that emotions are the result of the way in which people use language to evaluate and interpret events and situations. The specific emotion that is generated depends on the specific linguistic patterns that are used.

4 Problem Statement

We frame our appraisal tagging task as a BIO sequence tagging problem (Ramshaw and Marcus, 1995), where segments are tagged with the Attitude labels of Affect, Appreciation and Judgement, and Polarity labels of Positive, Negative and Neutral. Let $S = (w_1, w_2, ..., w_l)$ be an input sentence, where w_i is the i-th token and l is the sequence length. The objective of the proposed model A^2 is to identify a set of attitude tags $T_{att} = (\text{B-Attitude}, \text{I-Attitude}, O)$ and a set of polarity tags $T_{pol} = (\text{B-Polarity}, \text{I-Polarity}, O)$ for each $w_i \in S$.

5 Methodology

The architecture of adaptive appraisal (A^2) model is shown in Figure 2. It comprises two modules: (1) a language model for automatic BIO tagging with two target tasks, and (2) a sentiment adapter for learning sentiment specific knowledge and a task adapter for generating task-specific representations.

5.1 The Adaptive Appraisal (A^2) Model

The model predicts Appraisal Attitude and Polarity labels simultaneously. For the input sequence $S = (w_1, w_2, ..., w_l)$, we first feed to adapter-based transformer to generate token embeddings.

$$v = \text{Adapter-Transformer}([\text{CLS}] \oplus w_1 \oplus ... \oplus w_l)$$
$$v = [CLS], Emb[w_1], ..., Emb[w_l]$$

Each token embedding(e.g. $Emb[w_1]$) is presented by a d dimensional vector, where BERT-based encoder $d = 768$. Then we get a list of token representations with sequence length l and dimension d, denoted as $v \in \mathbb{R}^{l \times d}$. For two different objectives, we pass the token representations to two separate multilayer perceptron (MLP), denoted as Φ_{att} and Φ_{pol}.

$$\begin{aligned}\Phi_{att} &= \sigma(XU) \\ \Phi_{pol} &= \sigma(XU) \\ Z &= \Phi_{att}\left(Emb[w_1], ..., Emb[w_l]\right) \\ Z' &= \Phi_{pol}\left(Emb[w_1], ..., Emb[w_l]\right)\end{aligned}$$

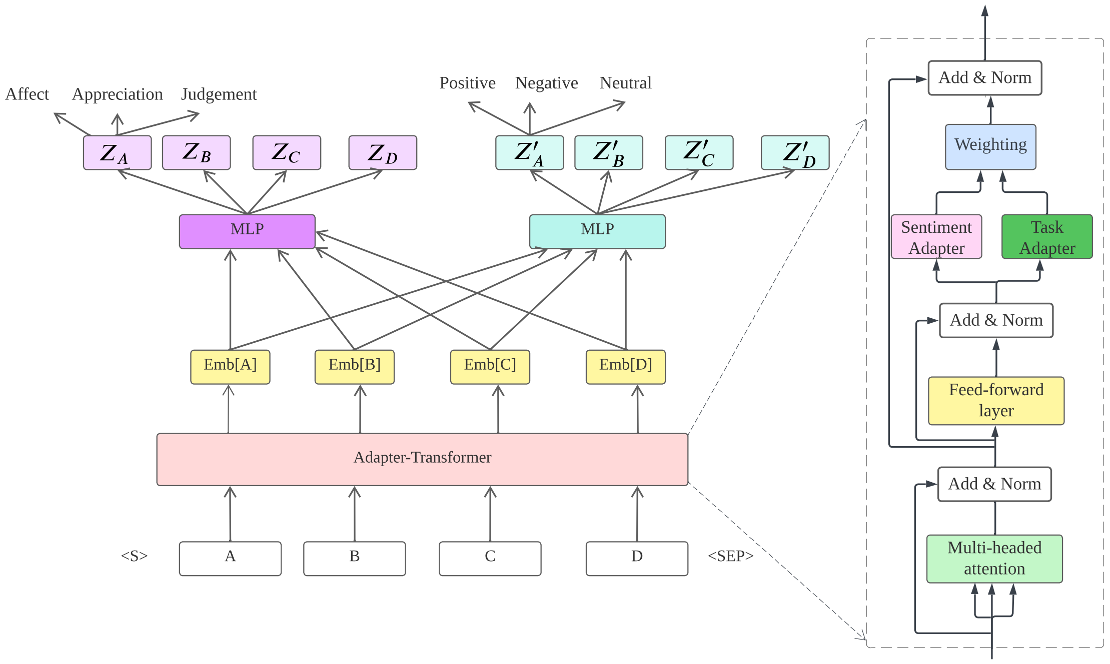


Figure 2: Overall architecture of Adapter-based Automatic Appraisal Framework ("A, B, C, D" are the tokens in the input sequence S)

where σ is ReLU (Agarap, 2018) activation function. U is the linear projection along the sequence length. Normalisation and biases are omitted for brevity. Due to the imbalance on our appraisal data, the dice loss (Li et al., 2020) has been adopted for token tagging tasks. To perform joint training with data D_{train} based on pre-trained language model on both the Attitude and Polarity labels, we minimise the overall loss:

$$\mathcal{L}^d_{att} = 1 - \frac{2p_d y_d + \gamma}{p_d^2 + y_d^2 + \gamma}$$

$$\mathcal{L}^d_{pl} = 1 - \frac{2p_d y_d + \gamma}{p_d^2 + y_d^2 + \gamma}$$

$$\mathcal{L} = \frac{1}{|\mathcal{D}_{train}|} \sum_{d \in \{D_{train}\}} \left(\mathcal{L}^d_{att} + \mathcal{L}^d_{pl} \right)$$

where λ is the L_2 regularisation parameters and Θ represents the parameters set. Following the dice loss setting, we set $\gamma = 1$, and p_d is the possibility of the data d belongs to the prediction y_d after the softmax function. $\mathcal{L}^d_{\text{att}}$ and $\mathcal{L}^d_{\text{pl}}$ are the loss functions for Attitude and Polarity labels, respectively. The reason for using the dice loss is to mitigate the impact of our imbalanced Appraisal labelled data.

5.2 Sentiment and Task Adapters

To develop a sentiment adapter and a task adapter, we followed an efficient adapter architecture recently proposed by Pfeiffer et al. (2021). They defined the adaptor structure by simply combining down and up projection with a residual connection. To examine whether sentiment knowledge can boost the performance of the appraisal tagging task, we propose employing a sentiment adapter on pre-trained language models. Furthermore, to capture task-specific knowledge, we propose employing a task adapter on pre-trained language models. The task adapter is trained with our appraisal training data D_{train}.

We denote sentiment-specific adaptive parameters Ω and task-specific adaptive parameters Ψ. In our architecture, we allocate the sentiment-specific adapter in parallel with our task-specific adapter after the feed-forward layer, followed by a ReLU activation at each layer l:

$$\Omega_l\left(\mathbf{h}_l, \mathbf{r}_l\right) = \mathbf{U}_l\left(\mathrm{ReLU}\left(\mathbf{D}_l\left(\mathbf{h}_l\right)\right)\right) + \mathbf{r}_l$$
$$\Psi_l\left(\mathbf{h}_l, \mathbf{r}_l\right) = \mathbf{U}_l\left(\mathrm{ReLU}\left(\mathbf{D}_l\left(\mathbf{h}_l\right)\right)\right) + \mathbf{r}_l$$

where $\mathbf{h}_l$ is the hidden states passing through the transformer architecture and $\mathbf{r}_l$ represents the residual at layer l. $\mathbf{D}_l$ is the projection presentation at layer l. To combine two adapters, we further introduce our simple yet effective adapter weight layer Θ, for each transformer layer l, the function denoted as:

$$\Theta_l = \alpha \Omega_l \left(\mathbf{h}_l, \mathbf{r}_l\right) + (1-\alpha)\, \Psi_l \left(\mathbf{h}_l, \mathbf{r}_l\right)$$

where α is the learned parameter and α can be different across transformer layers.

The sentiment adapter is fine-tuned with the sentiment datasets [2] at the sentence-level with negative log-likehood loss with $\mathcal{D}_{senti}$ as sentiment training set, as following:

$$\mathcal{L}_{senti} = -\frac{1}{|\mathcal{D}_{senti}|} \sum_{n=1}^{N} \log\left(p\left(y^n \mid x^n\right)\right)$$

5.3 Cross-lingual Inference

Follow the adapter-based cross-lingual framework MAD-X (Pfeiffer et al., 2020), we include a target language adapter when we transfer from English→Chinese and Chinese→English tasks. For instance, if we transfer from English→Chinese, we will plug in the Chinese adapter before the parallel task and sentiment adapters. Note that for cross-lingual tests, we also swap the base encoder from BERT to XLM-R to handle multi-lingual token embedding.

6 Datasets

For the cross-domain and cross-lingual appraisal tagging task, we conducted experiments on two datasets: the SOCC and POST datasets. The SOCC dataset comprises of 10,399 opinion news articles and 663,173 comments from the Canadian daily newspaper, *The Glove and Mail*. In addition to the raw text, the corpus includes specific annotations from multiple perspectives: negation, appraisal, constructiveness and toxicity. Among them, we used the appraisal annotations for our experiments. The POST dataset comprises of Twitter and Blog posts annotated with appraisals developed in-house following the same Appraisal Framework (Martin and White, 2003); it contains not only English text but also Chinese text. In the POST dataset, various spans of appraisals were annotated, as evaluation occurs at all levels of languages (words, phrases, clauses or entire sentence). In our experiments, we used the word and phrase level of labels. When there are phrases with more than one appraisal labels overlapping, we opted the longest span label.The data statistics of these two datasets are shown in Table 1 and Table 2.

	Affect	Judgement	Appreciation	#Total
Negative	175	2,342	2,350	4,867
Positive	46	469	1,173	1,688
Neutral	5	10	53	68
#Total	226	2,821	3,576	6,623

Table 1: Statistics of Appraisal labels in the SOCC dataset

	Affect	Judgement	Appreciation	#Total
Blogs				
Negative	548	758	333	1,639
Positive	505	397	256	1,158
Neutral	124	53	24	201
Tweets-English				
Negative	117	335	138	590
Positive	112	155	82	349
Neutral	50	25	28	103
Tweets-Chinese				
Negative	29	144	78	251
Positive	50	80	97	227
Neutral	10	12	14	36
#Total	1,545	1,959	1,050	4,554

Table 2: Statistics of Appraisal labels in the POST dataset

7 Experiments and Results

7.1 Experiment Setup

A^2 is an adapter-based framework adding adapter modules to pre-trained language models; A^2(BERT) and A^2(RoBERTa) are based on language models, BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), respectively. We compared the performance of A^2 framework models with following baseline models:

- Conditional Random Fields (CRF) (Lample et al., 2016) is a baseline that is widely used, feature-based model for a sequence tagging task.
- GLoVe+FFN is a baseline where tokens are encoded by the max and mean of GloVe (Pennington et al., 2014) [3] embeddings and followed by a feedforward neural network (FFN) for sequence labelling.
- BERT (Devlin et al., 2019) is a baseline based on the off-the-shelf BERT token embeddings, where we include further pre-training (BERT-PT) (Gururangan et al., 2020) and fine-tuning (BERT-FT) on a sentiment dataset.

---

[2] `https://github.com/cardiffnlp/tweeteval`

[3] `https://github.com/stanfordnlp/GloVe`

	#Total	Avg. Appraisal span length
News Comments	6,623	6.11
Blogs	2,998	12.02
Tweets	1,042	4.46
Tweets(CHN)	514	3.01

Table 3: Statistics of Appraisal expressions in different domains. Span length is counted by # of tokens.

- RoBERTa (Liu et al., 2019) is a baseline similar to BERT, where we include further pre-training (RoBERTa-PT) and fine-tuning (RoBERTa-FT) on a sentiment dataset.
- M-BERT is a baseline based on multilingual BERT.
- XLM-R (Conneau et al., 2019) is a baseline based on the multilingual language model XLM-R.

7.2 Implementation Details

We implemented our models in PyTorch using the HuggingFace library[4] and their pretrained BERT[5] and RoBERTa[6] models. Adapters in language models are implemented with the AdapterHub [7] package. The 100-dimension GloVe word embedding is applied for the GloVE+FFN model.

For the input sequence, we set maximum token length$=384$ and dropout rate $= [0.5, 0.6]$ for token embeddings. Learning rate is tuned in the range between $[1e^{-5}, 5e^{-5}]$ for BERT and $[1e^{-6}, 5e^{-6}]$ for RoBERTa based on the development set. All models use the Adam optimiser (Kingma and Ba, 2014), and our experiments are run using one A100 GPU with 40GB Memory.

7.3 Results

We evaluated A^2 for each attitude and polarity category. Each result is an average of three runs with different random seeds. For pre-training, we followed the procedures in (Gururangan et al., 2020) with masked LM loss. Table 4 and Table 5 present the in-domain test results on the SOCC and POST datasets, respectively.

As shown in the first group of both Table 4 and Table 5, compared with CRF and GloVe+FNN models, both BERT- and RoBERTa-based models yield better performance based on the average F1 scores on both datasets. This indicates that existing pre-trained language models have better capability to handle the appraisal tagging task. Moreover within the off-the-shelf language models, RoBERTa generally performs better than BERT.

The second group of both Table 4 and Table 5 shows the results of the continued pre-training and sequential fine-tuning language models. Comparing the performance of the first group of off-the-shelf language models, we found that our models with both strategies boost the performance further on all the Attitude (*Affect*, *Appreciation* and *Judgement*) labels and Polarity (*Positive*, *Negative* and *Neutral*) labels. The improved performance on the polarity labels especially implies that sentiment knowledge is useful for the appraisal tagging task. On the question of which strategy works better for this task, we observed that the continued pre-training strategy generally can bring better performance compared with the sequential fine-tuning strategy on both datasets in most of the cases except on the average F1 scores (e.g. 65.24% vs. 65.48%).

The main results of the appraisal tagging performance on our A^2 models are presented in the last group of both Table 4 and Table 5. As shown at the overall F1 score, both our A^2 models yield strong performance on these two datasets compared with all the other models. Note that we only fine-tuned the adapter parameters, which only apply to 15% of the overall number of parameters in the transformer-based language models.

7.4 Ablation Study

We compared the “jointly adap” approach of our A^2 framework (A^2) with the "sequentially adapt" approach in the literature (Pfeiffer et al., 2020), where the task-specific adapter is stacked on top of the sentiment-fused adapter (A$^2_{seq}$). As shown in the first group of Table 6, the joint adapters with weighting layer (A^2) gives better performance compared with stacking both adapters (A$^2_{seq}$) for the appraisal tagging task.

Furthermore, we conducted an ablation study to compare the performance of our A^2 framework against one without the task adapter (w/o T-Adpt) and one without the sentiment adapter (w/o S-Adpt). As presented in the second group of Table 6, the model with both sentiment and task adapters (A^2) performs the best and the sentiment adapter

---

[4] `https://github.com/huggingface`
[5] `https://huggingface.co/bert-base-cased`
[6] `https://huggingface.co/roberta-base`
[7] `https://adapterhub.ml/`.

Model	Avg. F1	Affect	Appreciation	Judgement	Positive	Negative	Neutral
CRF	42.67	35.69	38.24	40.77	44.26	52.25	41.55
GloVe+FFN	51.23	45.15	48.35	51.66	52.09	54.01	52.52
BERT	58.07	50.37	59.95	60.59	63.73	61.56	52.27
RoBERTa	59.47	53.66	60.56	62.54	60.63	63.77	55.64
BERT-PT	65.24	60.14	63.24	71.17	67.75	68.04	61.09
BERT-FT	65.48	61.48	65.15	70.05	66.68	69.37	60.14
RoBERTa-PT	67.10	61.66	66.28	73.34	68.04	71.85	61.44
RoBERTa-FT	66.93	61.90	66.57	69.01	67.41	70.79	65.89
A^2(BERT)	68.20	65.58	68.82	71.25	67.75	72.02	63.78
A^2(RoBERTa)	69.81	67.71	69.90	73.41	68.89	73.34	65.50

Table 4: F1 scores of all models on the SOCC test set (“PT” = pre-training, “FT” = fine-tuning)

Model	Avg. F1	Affect	Appreciation	Judgement	Positive	Negative	Neutral
CRF	47.51	43.56	43.03	44.79	49.97	44.88	44.66
GloVe+FFN	61.92	53.70	59.59	63.56	64.31	68.44	60.54
BERT	68.45	62.56	67.69	71.23	74.45	69.28	65.50
RoBERTa	69.91	63.43	69.81	70.49	75.97	72.21	67.56
BERT-PT	71.68	68.07	70.07	76.42	75.21	73.38	66.90
BERT-FT	69.17	66.16	69.76	72.25	71.30	70.68	64.55
RoBERTa-PT	73.01	69.98	71.72	78.63	74.11	75.32	68.35
RoBERTa-FT	71.72	69.50	72.65	77.11	73.28	74.32	63.46
A^2(BERT)	72.45	71.25	69.38	71.83	77.01	77.85	67.36
A^2(RoBERTa)	74.86	74.45	71.62	74.50	78.18	79.45	70.95

Table 5: F1 scores of all models on the POST English test set (“PT” = pre-training, “FT” = fine-tuning)

brings major improvement to model performance on the POST test set, by delivering +8.23 and +8.2 F1 scores on Polarity labels, *Positive* and *Negative*, respectively.

The experiment results indicate that the joint adapter approach can fully leverage both sentiment-specific and task-specific information through our adapter architecture and sentiment knowledge can greatly enrich token semantics.

7.5 Cross-domain Performance

This set of experiments aim to answer our second research question (RQ2), "*Can the sentiment knowledge fused adapter improve the adapter-based framework for the cross-domain appraisal tagging?*". As shown in Table 3, the average appraisal expression length varies in different domains; for instance, Twitter appraisals (avg. 4 tokens) have less number of tokens compared with News Comment appraisals (avg. 6 tokens) and Blog appraisals (avg. 12 tokens). As the SOCC and POST datasets contain corpus from three different domains (Tweet, Blog and News Comment) we got 6 sets of cross-domain performance results in total: Tweet→Blog, Tweet→News Comment, Blog→Tweet, Blog→News Comment, News Comment→Tweet, and News Comment→Blog. For instance, under Tweet→Blog setting, we will use Tweet domain text as training and zero-shot testing the performance on Blog domain text. Table 7 shows the average F1 scores of the cross-domain performance results in the zero-shot setting.

When comparing the cross-domain performance of the baseline models (BERT and RoBERT) against the fine-tuned models (BERT-FT and RoBERT-FT), we observe that directly fine-tuning language models with a sentiment dataset does not always yield performance improvement. For example, while the BERT-FT improves its performance (from 0.61 to 5.58 of F1 score) than the BERT on the most of the cross-domain settings (except one ‘News Comment→Blog’ setting), the RoBERTa-FT worsens its performance (from -0.55 to -1.83 of F1 score) than the RoBERTa on the most of the cross-domain settings (except one ‘Tweet→News Comment setting). This result indicates that directly fine-tuning language models without considering domain specific vectors can hurt naive trans-

Model	Avg. F1	Affect	Appreciation	Judgement	Positive	Negative	Neutral
A^2	74.86	74.45	71.62	74.50	78.18	79.45	70.95
A^2_{seq}	72.73	70.70	68.07	77.65	75.94	75.80	68.24
w/o T-Adpt	69.37	66.85	65.65	72.38	71.97	72.80	66.56
w/o S-Adpt	69.09	68.82	66.50	73.50	69.95	71.25	64.49

Table 6: F1 scores of RoBERTa-based models on the POST datasets (“T-Adpt” = task adapter, “S-Adpt” = sentiment adapter)

Model	T		B		N	
	B	N	T	N	T	B
BERT	50.81	53.45	52.87	45.13	56.21	53.17
BERT-FT	51.43	54.72	53.04	50.71	56.82	51.61
RoBERTa	52.24	56.26	54.64	48.03	59.29	55.45
RoBERTa-FT	51.69	57.62	52.81	52.54	58.10	54.44
A^2(BERT)	55.61	61.22	63.14	57.45	67.33	60.89
A^2(RoBERTa)	59.79	62.67	64.61	61.55	68.12	63.34

Table 7: Cross-domain performance (F1 score, “T” = Tweet, “B” = Blog, “N” = News Comment)

Model	eng→chn		chn→eng	
MBERT	68.11	61.07	72.64	52.12
XLM-R	70.71	63.77	76.06	56.69
A^2(MBERT)	69.45	64.33	71.54	61.78
A^2(XLM-R)	72.64	67.91	75.44	62.67

Table 8: Cross-lingual performance on the POST dataset (“eng” = English, “chn” = Chinese)

fer learning. On the other hand, our A^2(BERT) and A^2(RoBERTa) models achieve better performance than the fine-tuned models (BERT-FT, RoBERTa-FT) on all the six cross-domain settings. Moreover, the A^2(RoBERTa) model achieves consistently better performance than the A^2(BERT) model on all the six cross-domain settings.

When comparing the cross-domain performance (shown in Table 4 and Table 5) against the in-domain performance (shown in Table 7), we can see that all language models show substantial drop in performance. For example, on News Comments, the performance of the BERT-FT drops from 65.48 to 54.72 and 50.71 when it is originally trained on Tweet domain but test on News Comment domain. This demonstrates the challenge of the cross-domain task setting, which may contain a catastrophic forgetting issue, conflicting signals and domain requirements. However, when we incorporate adapters in A^2 framework, we observe that the performance gap diminishes significantly.

7.6 Cross-lingual Performance

Table 8 shows the cross-lingual performance results, from English to Chinese with tweets data and vice versa. To answer RQ3, "*Can the sentiment knowledge fused adapter improve the adapter-based framework for the cross-lingual appraisal tagging?*", we first fine-tuned the multilingual BERT (MBERT) and XLM-RoBERTa (XLM-R) models using the labels in the source language and applied them to the target languages with subword embeddings frozen.Then, we compared with them under a simple set up of our A^2 framework with plugging in a target language specific adapter. Both A^2(MBERT) and A^2(XLM-R) models demonstrate performance gains (avg. 6.96% and 5.06% for MBERT and XLM-R based) on the target language.

We can see when we transfer from source to target language, existing multilingual language models perform poorly. For example, with XLM-R, when we test out eng→chn, there is a huge performance drop from 76.06% to 63.77% on Chinese data. Both A^2 models are slightly under-perform (avg. 0.85%) the base language models when Chinese as the source language, which may due to the sentiment adapter trained with English data only. Note that the performance on the source language does not decrease as we only replace the language-specific adapter at the inference time.

8 Illustration of a prediction error

To provide a qualitative analysis for our approach, we showcase an example of an annotated sentence from the SOCC dataset in Table 9. We present our A^2 prediction v.s. the human annotations. The original sentence is a comment towards an article discussing the aboriginal of Canada [8]. Based on

---

[8] `https://www.theglobeandmail.com/opinion/too-many-first-nations-people-live-in-a-dream-palace/article6929035/`

Sentence	The author does not seem to have much of a clue in spite of her elevated status.
A^2 prediction	The author does not seem to have much Jud Neg of a clue in spite of her elevated status. App Neg
Gold annotation	The author does not seem to have much of a clue Jud Neg in spite of her elevated status.

Table 9: Illustration of a prediction error

the human annotations, the whole span of “does not seem to have much of a clue” labelled with *Judgement* Attitude and *Negative* Polarity. Our A^2 framework can accurately predict the *Judgement* Attitude and *Negative* Polarity labels but missing three following tokens. It also falsely tags the text segment, “elevated status” as *Appreciation* Attitude and *Negative* Polarity.

9 Conclusion

We have proposed A^2, an adapter-based framework for automatically tagging Appraisal expressions. We have designed task and sentiment adapters of a small number of additional parameters to improve the capacity of pre-trained language models for quick adaptation for cross-domain and cross-language settings.

Limitations

Our system is based on the pre-trained language models and therefore assumes that GPU resources are available. The system is designed for tagging short opinion-bearing texts and the maximum text length is set to 384 tokens. Moroever, our system only performed cross-lingual tests from English to Chinese and Chinese to English for experiments. This is mainly constrained by the dataset availability. To our best knowledge, the POST dataset is the only available resource in Chinese with appraisal annotations. It is desirable to conduct more experiments on a broader set of languages to evaluate the generalisability of the A^2 model for cross-lingual adaptation.

Acknowledgements

This research is supported in part by the Australian Research Council Discovery Project DP200101441 and in part by the Defence Science and Technology Group, Australia.

References

Abien Fred Agarap. 2018. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*.

Soravit Changpinyo, Hexiang Hu, and Fei Sha. 2018. Multi-task learning for sequence tagging: An empirical study. In *COLING*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

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

Suzanne Eggins. 2004. *Introduction to systemic functional linguistics*. A&c Black.

Phoebe C Ellsworth and Craig A Smith. 1988. From appraisal to emotion: Differences among unpleasant feelings. *Motivation and emotion*, 12(3):271–302.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don’t stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Maximilian Hofer, Andrey Kormilitzin, Paul Goldberg, and Alejo Nevado-Holgado. 2018. Few-shot learning for named entity recognition in medical text. *arXiv preprint arXiv:1811.05468*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *ICML*.

Katharina Kann, Johannes Bjerva, Isabelle Augenstein, Barbara Plank, and Anders Søgaard. 2018. Character-level supervision for low-resource pos tagging. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 1–11.

Joo-Kyung Kim, Young-Bum Kim, Ruhi Sarikaya, and Eric Fosler-Lussier. 2017. Cross-lingual transfer learning for pos tagging without cross-lingual resources. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2832–2838.

Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. 2015. New transfer learning techniques for disparate label sets. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 473–482.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Varada Kolhatkar, Hanhan Wu, Luca Cavasso, Emilie Francis, Kavan Shukla, and Maite Taboada. 2020. The sfu opinion and comments corpus: A corpus for the analysis of online news comments. *Corpus Pragmatics*, 4(2):155–190.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2018a. Transfer learning for named-entity recognition with neural networks. In *Language Resources and Evaluation Conference*.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2018b. Transfer learning for named-entity recognition with neural networks. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced nlp tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019. Exploiting bert for end-to-end aspect-based sentiment analysis. *arXiv preprint arXiv:1910.00883*.

Bing Liu. 2022. *Sentiment analysis and opinion mining*. Springer Nature.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

James R Martin and Peter R White. 2003. *The language of evaluation*, volume 2. Springer.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Teresa Oteíza. 2017. The appraisal framework and discourse analysis. In *The Routledge handbook of systemic functional linguistics*, pages 481–496. Routledge.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. Mad-x: An adapter-based framework for multi-task cross-lingual transfer. *arXiv preprint arXiv:2005.00052*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Jonathon Read and John Carroll. 2012. Annotating expressions of appraisal in english. *Language resources and evaluation*, 46(3):421–447.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings* *of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. Udapter: Language adaptation for truly universal dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2302–2315.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal groups for sentiment analysis. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 625–631.

Da Yin, Tao Meng, and Kai-Wei Chang. 2020. Sentibert: A transferable transformer-based architecture for compositional sentiment semantics. In *ACL*.

DREEAM: Guiding Attention with Evidence for Improving Document-Level Relation Extraction

Youmi Ma and **An Wang** and **Naoaki Okazaki**
Tokyo Institute of Technology
`{youmi.ma@nlp., an.wang@nlp., okazaki@}c.titech.ac.jp`

Abstract

Document-level relation extraction (DocRE) is the task of identifying all relations between each entity pair in a document. Evidence, defined as sentences containing clues for the relationship between an entity pair, has been shown to help DocRE systems focus on relevant texts, thus improving relation extraction. However, evidence retrieval (ER) in DocRE faces two major issues: high memory consumption and limited availability of annotations. This work aims at addressing these issues to improve the usage of ER in DocRE. First, we propose DREEAM, a memory-efficient approach that adopts evidence information as the supervisory signal, thereby guiding the attention modules of the DocRE system to assign high weights to evidence. Second, we propose a self-training strategy for DREEAM to learn ER from automatically-generated evidence on massive data without evidence annotations. Experimental results reveal that our approach exhibits state-of-the-art performance on the DocRED benchmark for both DocRE and ER. To the best of our knowledge, DREEAM is the first approach to employ ER self-training[1].

1 Introduction

Document-level relation extraction (DocRE) has been recognized as a more realistic and challenging task compared with its sentence-level counterpart (Peng et al., 2017; Verga et al., 2018; Yao et al., 2019). In DocRE, an entity can have multiple mentions scattered throughout a document, and relationships can exist between entities in different sentences. Therefore, DocRE models are expected to apply information filtering to long texts by focusing more on sentences relevant to the current decision of relation extraction (RE) and less on irrelevant ones. To this end, existing studies retrieve *supporting evidence* (evidence hereafter, Yao et al., 2019), a set of sentences necessary for humans to identify the relation between an entity pair (Huang et al., 2021a,b; Xie et al., 2022; Xiao et al., 2022; Xu et al., 2022). As shown in Figure 1, to decide the *present in work* relation between *Prince Edmund* and *Blackadder*, reading sentences 1 and 2 should be sufficient. Although sentences 5 and 6 also mention the subject, they are irrelevant to the relation decision. Evidence of the relation triple (*Prince Edmund*, *present in work*, *Blackadder*) is thus defined as sentences 1 and 2.

The Archbishop

[1] "The Archbishop" is the third episode of the first series of the BBC sitcom ***Blackadder*** (***The Black Adder***). [2] It is set in England in the late 15th century, and follows the exploits of the fictitious ***Prince Edmund*** as he is invested as Archbishop of Canterbury amid a Machiavellian plot by the King to acquire lands from the Catholic Church. [3] ... [5] ***Edmund***, faced with the threat of assassination, attempts to escape to France into self-imposed exile; and in a later scene, two drunk knights overhear King Richard IV exclaiming "Who will rid me of this turbulent priest?" [6] The words attributed to King Henry II which led to Becket's death in 1170, and embark on a mission to murder ***Edmund***. [7] …

Subject: ***Prince Edmund***	**Relation**: ***present in work***
Object: ***Blackadder***	**Evidence: 1,2**

Figure 1: Example document and one of the relation triples from DocRED, where the i-th sentence is marked with [i] in the beginning. Mentions in bold italics are those of subjects and objects, whereas entity mentions other than subject and object are underlined.

Despite the usefulness of evidence, automatic evidence retrieval (ER) faces two major issues. Firstly, the existing approaches for ER are memory-inefficient. Previous systems tackle ER and DocRE as separate tasks, introducing extra neural network layers to learn ER with DocRE jointly (Huang et al., 2021a; Xie et al., 2022; Xiao et al., 2022). The ER module typically involves a bilinear classifier that receives entity-pair-specific embeddings and sentence embeddings as the input. To compute the evidence score of each sentence for each entity pair, the module must walk through all (entity pair, sentence) combinations. The computations signifi-

---

[1] The source code is available at `https://github.com/YoumiMa/dreeam`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1971–1983
May 2-6, 2023 ©2023 Association for Computational Linguistics

cantly increase memory consumption, particularly in documents with numerous sentences and entities. Secondly, the availability of human annotations of evidence is limited. To make matters worse, gold training data for DocRE are more expensive to annotate than those for their sentence-level counterpart. Despite the difficulty of obtaining human annotations, acquiring evidence annotations at a low cost has been underexplored. Although automatically collecting silver training data for RE by distant supervision (Mintz et al., 2009; Yao et al., 2019), locating evidence for a sliver RE instance in the document is nontrivial.

This work aims at alleviating these issues to improve the usage of ER in DocRE. To reduce the memory consumption, we propose **D**ocument-level **R**elation **E**xtraction with **E**vidence-guided **A**ttention **M**echanism (DREEAM), a memory-efficient approach for incorporating DocRE with ER. We adopt ATLOP (Zhou et al., 2021), a Transformer-based DocRE system widely used in previous studies (Xie et al., 2022; Tan et al., 2022a; Xiao et al., 2022), as the backbone. Instead of introducing an external ER module, we directly guide the DocRE system to focus on evidence. Specifically, we supervise the computation of entity-pair-specific local context embeddings. The local context embedding, formed as a weighted sum among all token embeddings based on attention from the encoder, is trained to assign higher weights to evidence and lower weights otherwise.

To compensate for the shortage of evidence annotations, we propose performing ER under a weakly-supervised setting. Specifically, we design a strategy to perform self-training with DREEAM on massive, unlabeled data. The data is obtained from distant supervision (distantly-supervised data hereafter) and thus is automatically annotated with relation labels but not evidence labels. We expect the knowledge about ER learned from the human-annotated data to generate and grow on the distantly-supervised data. To enable self-training, we first adopt a teacher model trained on human-annotated data to retrieve silver evidence from distantly-supervised data. Next, we train a student model on the data for RE while learning ER from the silver evidence. The student model is further finetuned on the human-annotated data to refine its knowledge. Experiments on the DocRED benchmark (Yao et al., 2019) show that with the help of ER self-training, DREEAM exhibits state-of-the-art performance on both RE and ER.

In short, the contributions of this work are: (1) We propose DREEAM, a memory-efficient approach to incorporate evidence information into Transformer-based DocRE systems by directly guiding the attention. DREEAM does not introduce any extra trainable parameters for ER while achieving good performance on both RE and ER. (2) We propose incorporating distantly-supervised RE training with ER self-training, which improves the performance on both tasks. To the best of our knowledge, DREEAM is the first DocRE system that enables joint training of ER and RE under a weakly-supervised setting.

2 Preliminary

2.1 Problem Formulation

Given a document D containing sentences $\mathcal{X}_D = \{x_i\}_{i=1}^{|\mathcal{X}_D|}$ and entities $\mathcal{E}_D = \{e_i\}_{i=1}^{|\mathcal{E}_D|}$, DocRE aims to predict all possible relations between every entity pair. Each entity $e \in \mathcal{E}_D$ is mentioned at least once in D, with all its proper-noun mentions denoted as $\mathcal{M}_e = \{m_i\}_{i=1}^{|\mathcal{M}_e|}$. Each entity pair (e_s, e_o) can hold multiple relations, comprising a set $\mathcal{R}_{s,o} \subset \mathcal{R}$, where $\mathcal{R}$ is a pre-defined relation set. We let the set $\mathcal{R}$ include ϵ, which stands for *no-relation*. Additionally, if an entity pair (e_s, e_o) carries a valid relation $r \in \mathcal{R}\backslash\{\epsilon\}$, ER aims to retrieve the supporting evidence $\mathcal{V}_{s,r,o} \subseteq \mathcal{X}_D$ that are sufficient to predict the triplet (e_s, r, e_o).

2.2 ATLOP

This section reviews ATLOP, the backbone of our proposed method.

Text Encoding Before encoding, a special token $*$ is inserted at the beginning and the end of each entity mention. Then, tokens $\mathcal{T}_D = \{t_i\}_{i=1}^{|\mathcal{T}_D|}$ within document D are encoded with a Transformer-based pretrained language model (PLM, Vaswani et al., 2017) to obtain token embeddings and cross-token dependencies. Although the original ATLOP adopts only the last layer, this work takes the average of the last three layers[2]. Specifically, for a PLM with d hidden dimensions at each transformer layer, the token embeddings $\boldsymbol{H}$ and cross-token dependencies $\boldsymbol{A}$ are computed as:

$$\boldsymbol{H}, \boldsymbol{A} = \mathrm{PLM}(\mathcal{T}_D), \tag{1}$$

---

[2] Pilot experiments showed that using the last 3 layers yields better performance than using only the last layer.

where $\boldsymbol{H} \in \mathbb{R}^{|\mathcal{T}_D| \times d}$ averages over hidden states of each token from the last three layers and $\boldsymbol{A} \in \mathbb{R}^{|\mathcal{T}_D| \times |\mathcal{T}_D|}$ averages over attention weights of all attention heads from the last three layers.

Entity Embedding The entity embedding $\boldsymbol{h}_e \in \mathbb{R}^d$ for each entity e with mentions $\mathcal{M}_e = \{m_i\}_{i=1}^{|\mathcal{M}_e|}$ is computed by collecting information from all its mentions. Specifically, `logsumexp` pooling, which has been empirically shown to be effective in previous studies (Jia et al., 2019), is adopted as: $\boldsymbol{h}_e = \log \sum_{i=1}^{|\mathcal{M}_e|} \exp(\boldsymbol{H}_{m_i})$, where $\boldsymbol{H}_{m_i}$ is the embedding of the special token $*$ at the starting position of mention m_i.

Localized Context Embedding To better utilize information from long texts, ATLOP introduces entity-pair specified localized context embeddings. Intuitively, for entity pair (e_s, e_o), tokens important to both e_s and e_o should contribute more to the embedding. The importance of each token is determined by the cross-token dependencies $\boldsymbol{A}$ obtained from Equation 1. For entity e_s, the importance of each token is computed using the cross-token dependencies of all its mentions $\mathcal{M}_{e_s}$. First, ATLOP collects and averages over the attention $\boldsymbol{A}_{m_i} \in \mathbb{R}^{|\mathcal{T}_D|}$ at the special token $*$ before each mention $m_i \in \mathcal{M}_{e_s}$ to get $\boldsymbol{a}_s \in \mathbb{R}^{|\mathcal{T}_D|}$ as the importance of each token for entity e_s. Then, the importance of each token for an entity pair (e_s, e_o), noted as $\boldsymbol{q}^{(s,o)} \in \mathbb{R}^{|\mathcal{T}_D|}$, is computed from $\boldsymbol{a}_s$ and $\boldsymbol{a}_o$ as:

$$\boldsymbol{q}^{(s,o)} = \frac{\boldsymbol{a}_s \circ \boldsymbol{a}_o}{\boldsymbol{a}_s^\top \boldsymbol{a}_o}, \tag{2}$$

where $\circ$ stands for the Hadamard product. $\boldsymbol{q}^{(s,o)}$ is thus a distribution that reveals the importance of each token for entity pair (e_s, e_o). Subsequently, ATLOP performs a localized context pooling,

$$\boldsymbol{c}^{(s,o)} = \boldsymbol{H}^\top \boldsymbol{q}^{(s,o)}, \tag{3}$$

where $\boldsymbol{c}^{(s,o)} \in \mathbb{R}^d$ is a weighted average over all token embeddings.

Relation Classification To predict the relation between entity pair (e_s, e_o), ATLOP first generates context-aware subject and object representations:

$$\boldsymbol{z}_s = \tanh(\boldsymbol{W}_s[\boldsymbol{h}_{e_s}; \boldsymbol{c}^{(s,o)}] + \boldsymbol{b}_s) \tag{4}$$

$$\boldsymbol{z}_o = \tanh(\boldsymbol{W}_o[\boldsymbol{h}_{e_o}; \boldsymbol{c}^{(s,o)}] + \boldsymbol{b}_o), \tag{5}$$

where $[\cdot;\cdot]$ represents the concatenation of two vectors and $\boldsymbol{W}_s, \boldsymbol{W}_o \in \mathbb{R}^{d \times 2d}$, $\boldsymbol{b}_s, \boldsymbol{b}_o \in \mathbb{R}^d$ are trainable parameters. Then, a bilinear classifier[3] is applied on the context-aware representations to compute the relation scores $\boldsymbol{y}^{(s,o)} \in \mathbb{R}^{|\mathcal{R}|}$:

$$\boldsymbol{y}^{(s,o)} = \boldsymbol{z}_s^\top \mathsf{W}_r \boldsymbol{z}_o + \boldsymbol{b}_r, \tag{6}$$

where $\mathsf{W}_r \in \mathbb{R}^{|\mathcal{R}| \times d \times d}$ and $\boldsymbol{b}_r \in \mathbb{R}^{|\mathcal{R}|}$ are trainable parameters. The probability that relation $r \in \mathcal{R}$ holds between entity e_s and e_o is thus $\mathrm{P}(r|s,o) = \sigma(y_r^{(s,o)})$, where σ is the sigmoid function.

Loss Function ATLOP proposes Adaptive Thresholding Loss (ATL) that learns a dummy threshold class TH during training, serving as a dynamic threshold for each relation class $r \in \mathcal{R}$. For each entity pair (e_s, e_o), ATL forces the model to yield scores above TH for positive relation classes $\mathcal{R}_P$ and scores below TH for negative relation classes $\mathcal{R}_N$, formulated as below:

$$\begin{aligned}\mathcal{L}_{\text{RE}} = &- \sum_{s \neq o} \sum_{r \in \mathcal{R}_P} \frac{\exp(y_r^{(s,o)})}{\sum_{r' \in \mathcal{R}_P \cup \{\text{TH}\}} \exp(y_{r'}^{(s,o)})} \\ &- \frac{\exp(y_{\text{TH}}^{(s,o)})}{\sum_{r' \in \mathcal{R}_N \cup \{\text{TH}\}} \exp(y_{r'}^{(s,o)})}.\end{aligned} \tag{7}$$

The idea of setting a threshold class is similar to the Flexible Threshold (Chen et al., 2020).

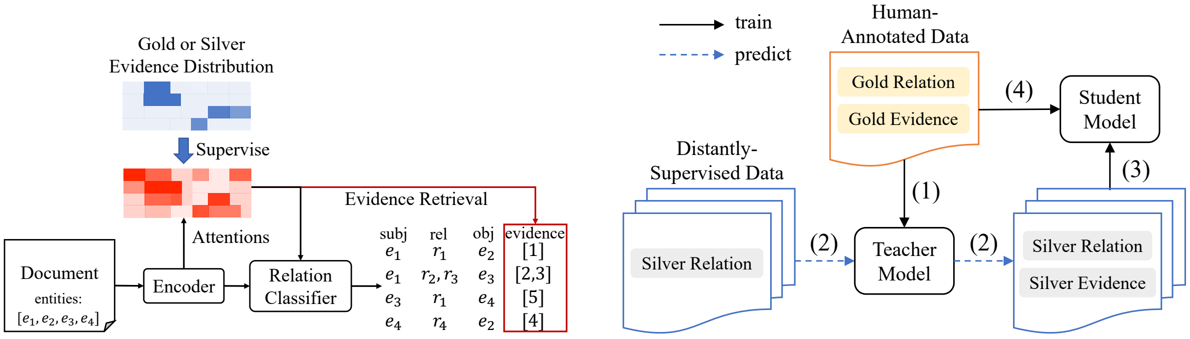
3 Proposed Method: DREEAM

To perform information filtering, ATLOP computes a localized context embedding based on attention weights from the Transformer-based encoder. The rationale is that cross-token dependencies are encoded as attention weights in Transformer layers. In this work, we propose DREEAM to enhance ATLOP with evidence. In addition to the automatically-learned cross-token dependencies, the attention modules are supervised to concentrate more on evidence sentences and less on others.

DREEAM can be employed for both supervised and self-training, sharing the same architecture with different supervisory signals, as shown in Figure 2 (a). Inspired by Tan et al. (2022a), we propose a pipeline to enable self-training of ER, with the data flow shown in Figure 2 (b). First, we train a teacher model on human-annotated data with gold relations and evidence labels. Next, we apply the trained teacher model to predict silver evidence for

---

[3] In practice, a grouped bilinear classifier (Zheng et al., 2019) is applied to save memory.



(a) Model architecture of DREEAM.

(b) Information flow of self-training using DREEAM.

Figure 2: Model architecture and the information flow during self-training. In (a), gold/silver evidence distributions come from human-annotations/the teacher model. In (b), arrows represent the direction of knowledge transfer.

the distantly-supervised data. Then, we train a student model on the distantly-supervised data, with ER supervised by the silver evidence. Finally, we finetune the student model on the human-annotated data to refine its knowledge. The rest of this section introduces the training processes of the teacher and student models, followed by the inference process.

3.1 Teacher Model

For each entity pair (s, o), we guide $\boldsymbol{q}^{(s,o)}$ with an evidence distribution to help generate an evidence-centered localized context embedding. While $\boldsymbol{q}^{(s,o)}$ yields token-level importance for e_s and e_o, we can obtain only sentence-level evidence from human annotations, as shown in Figure 1. To alleviate this gap, we sum the weight of each token within a sentence. Specifically, for a sentence $x_i \in \mathcal{X}_D$ consisting of tokens $t_{\text{START}(x_i)}, \ldots, t_{\text{END}(x_i)}$, we obtain the sentence-level importance as:

$$p_i^{(s,o)} = \sum_{j=\text{START}(x_i)}^{\text{END}(x_i)} q_j^{(s,o)}. \tag{8}$$

Collecting the importance of all sentences yields a distribution $\boldsymbol{p}^{(s,o)} \in \mathbb{R}^{|\mathcal{X}_D|}$ that expresses the importance of each sentence within the document.

We further supervise $\boldsymbol{p}^{(s,o)}$ for each entity pair (e_s, e_o) using a human-annotated evidence distribution computed from gold evidence. First, we define a binary vector $\boldsymbol{v}^{(s,r,o)} \in \mathbb{R}^{|\mathcal{X}_D|}$ for each valid relation label $r \in \mathcal{R}_{s,o} \subset \mathcal{R} \backslash \{\epsilon\}$ that records whether each sentence $x_i \in \mathcal{X}_D$ is evidence of the relation triple (e_s, r, e_o) or not. For example, if x_i is evidence of (e_s, r, e_o), then $v_i^{(s,r,o)}$ is set to 1, and otherwise 0.

Next, we marginalize all valid relations and normalize the marginalized vector to obtain $\boldsymbol{v}^{(s,o)}$:

$$\boldsymbol{v}^{(s,o)} = \frac{\sum_{r \in \mathcal{R}_{s,o}} \boldsymbol{v}^{(s,r,o)}}{\sum_{r \in \mathcal{R}_{s,o}} \mathbf{1}^\top \boldsymbol{v}^{(s,r,o)}}, \tag{9}$$

where $\mathbf{1} = (1, 1, \ldots, 1) \in \mathbb{R}^{|\mathcal{X}_D|}$ is an all-ones vector. The rationale behind Equation 9 is that modules before the relation classifier are not explicitly aware of specific relation types. We thus guide attention modules within the encoder to produce relation-agnostic token dependencies.

Loss Function Our purpose is to guide $\boldsymbol{p}^{(s,o)}$ with human evidence $\boldsymbol{v}^{(s,o)}$ to generate an evidence-focused localized context embedding $\boldsymbol{c}^{(s,o)}$. To achieve this, we train the model with Kullback-Leibler (KL) Divergence loss, minimizing the statistical distance between $\boldsymbol{p}^{(s,o)}$ and $\boldsymbol{v}^{(s,o)}$:

$$\mathcal{L}_{\text{ER}}^{\text{gold}} = -D_{\text{KL}}(\boldsymbol{v}^{(s,o)} || \boldsymbol{p}^{(s,o)}). \tag{10}$$

During training, we balance the effect of ER loss with RE loss using a hyper-parameter λ:

$$\mathcal{L}^{\text{gold}} = \mathcal{L}_{\text{RE}} + \lambda \mathcal{L}_{\text{ER}}^{\text{gold}}. \tag{11}$$

3.2 Student Model

We employ the system trained on human-annotated data as a teacher model to support ER self-training on massive data. The data, obtained from relation distant-supervision (Mintz et al., 2009), contains noisy labels for RE but no information for ER. We train a student model on the data. Supervision of the student model, similar to that of the teacher model, consists of two parts: an RE binary cross-entropy loss and an ER self-training loss.

In general, predictions from the teacher model are adopted as the supervisory signal for ER training. First, we let the teacher model infer on the distantly-supervised data, thereby yielding an evidence distribution over tokens $\hat{\boldsymbol{q}}^{(s,o)}$ for each entity pair (e_s, e_o). Next, we train the student model to reproduce $\hat{\boldsymbol{q}}^{(s,o)}$ for each entity pair (e_s, e_o).

Loss Function The objectives of self-training are identical to those of supervised training. We train ER of the student model using a KL-divergence loss similar to Equation 10:

$$\mathcal{L}_{\mathrm{ER}}^{\mathrm{silver}} = -D_{\mathrm{KL}}(\hat{\boldsymbol{q}}^{(s,o)} || \boldsymbol{q}^{(s,o)}), \tag{12}$$

where $\boldsymbol{q}^{(s,o)}$ is the student model's evidence distribution over tokens regarding entity pair (e_s, e_o), computed from Equation 2.

There are two notable differences between $\mathcal{L}_{\mathrm{ER}}^{\mathrm{silver}}$ and $\mathcal{L}_{\mathrm{ER}}^{\mathrm{gold}}$. Firstly, the supervisory signal of $\mathcal{L}_{\mathrm{ER}}^{\mathrm{gold}}$ is sentence-level, while that of $\mathcal{L}_{\mathrm{ER}}^{\mathrm{silver}}$ is token-level. The gap results from the availability of token-level evidence distributions. On human-annotated data, it is untrivial to obtain token-level evidence distributions from sentence-level annotations. On distantly-supervised data, however, the evidence distribution over tokens can be easily obtained from predictions of the teacher model. We thus adopt token-level evidence distributions to provide supervision from a micro perspective for ER self-training. Secondly, $\mathcal{L}_{\mathrm{ER}}^{\mathrm{gold}}$ is computed only on entity pairs with valid relation(s), while $\mathcal{L}_{\mathrm{ER}}^{\mathrm{silver}}$ is computed over all entity pairs within the document. The design choice is based on the low reliability of relation labels on distantly-supervised data. As these relation labels are collected automatically, it is possible that some of the annotated relations are irrelevant to the document. Therefore, it is hard to tell which relations are valid and which are not from the automatic annotations. For this reason, we compute the loss from all entity pairs to prevent missing important instances.

The overall loss is balanced by the same hyper-parameter λ in Equation 11:

$$\mathcal{L}^{\mathrm{silver}} = \mathcal{L}_{\mathrm{RE}} + \lambda \mathcal{L}_{\mathrm{ER}}^{\mathrm{silver}}. \tag{13}$$

After training on the distantly-supervised data, the student model is further finetuned using the human-annotated data to refine its knowledge about DocRE and ER with reliable supervisory signals.

3.3 Inference

Following Zhou et al. (2021), we apply adaptive thresholding to obtain RE predictions, selecting relation classes with scores higher than the threshold class as predictions. For ER, we apply static thresholding and choose sentences with importance higher than a pre-defined threshold as evidence.

Statistics	Human	Distant
# of documents	3,053/998/1,000	101,873
# of relation types	97	97
Avg. # of ent. per doc.	19.5	19.3
Avg. # of sent. per doc.	8.0	8.1
Avg. # of ment. per ent.	1.3	1.3
Avg. # of rel. per doc.	12.5	14.8
Avg. # of evi. per rel.	1.6	-

Table 1: Data statistics of DocRED. *Human* stands for human-annotated data and *Distant* stands for distantly-supervised data. The abbreviations *doc.*, *ent.*, *sent.*, *ment.*, *rel.*, and *evi.* stand for document, entity, sentence, mention, relation, and evidence sentences, respectively.

We further incorporate the **inference-stage fusion** strategy proposed by Xie et al. (2022). Specifically, for each predicted relation triple (e_s, r, e_o) associated with evidence prediction $\mathcal{V}_{s,r,o}$, we create a pseudo-document $\hat{D}_{s,r,o}$ by collecting only evidence sentences $x_i \in \mathcal{V}_{s,r,o}$. Then, we feed pseudo-documents into the trained model to re-score the relation triples. To aggregate the predictions from the pseudo-documents and the entire document, we adopt a blending layer that contains only one parameter τ representing a threshold. Each triple (e_s, r, e_o) is chosen as the final prediction only if the summation of its scores on the entire document and pseudo-documents is higher than τ. We adjust τ to minimize the binary cross-entropy loss of RE on the development set. For more details, we refer the readers to the original paper (Xie et al., 2022).

4 Experiments

To evaluate DREEAM, we conduct experiments under supervised and weakly-supervised settings.

4.1 Setting

Dataset We conduct experiments on DocRED (Yao et al., 2019)[4], the largest dataset for DocRE with human annotations. As shown in Table 1, DocRED contains a small portion of human-annotated data and a large portion of distantly-supervised data made by aligning Wikipedia articles with the Wikidata knowledge base (Vrandečić and Krötzsch, 2014). In this work, we directly adopt the distantly-supervised data provided in DocRED.

Configuration We implement DREEAM based on Hugging Face's Transformers (Wolf et al., 2020).

---

[4] `https://github.com/thunlp/DocRED`

			Dev			Test	
Method	PLM	Ign F1	F1	Evi F1	Ign F1	F1	Evi F1
(a) without Distantly-Supervised Data							
SSAN (Xu et al., 2021a)	BERT$_{\text{base}}$	57.03	59.19	-	55.84	58.16	-
ATLOP (Zhou et al., 2021)	BERT$_{\text{base}}$	59.22	61.09	-	59.31	61.30	-
E2GRE (Huang et al., 2021a)	BERT$_{\text{base}}$	55.22	58.72	47.12	-	-	-
DocuNet (Zhang et al., 2021)	BERT$_{\text{base}}$	59.86	61.83	-	59.93	61.86	-
EIDER (Xie et al., 2022)	BERT$_{\text{base}}$	60.51	62.48	50.71	60.42	62.47	51.27
SAIS (Xiao et al., 2022)	BERT$_{\text{base}}$	59.98	62.96	53.70	60.96	62.77	52.88
DREEAM (teacher)	BERT$_{\text{base}}$	59.60$_{\pm 0.15}$	61.42$_{\pm 0.15}$	52.08$_{\pm 0.10}$	59.12	61.13	51.71
+ Inference-stage Fusion		60.51$_{\pm 0.06}$	62.55$_{\pm 0.06}$		60.03	62.49	
SSAN (Xu et al., 2021a)	RoBERTa$_{\text{large}}$	60.25	62.08	-	59.47	61.42	-
ATLOP (Zhou et al., 2021)	RoBERTa$_{\text{large}}$	61.32	63.18	-	61.39	63.40	-
DocuNet (Zhang et al., 2021)	RoBERTa$_{\text{large}}$	62.23	64.12	-	62.39	64.55	-
EIDER (Xie et al., 2022)	RoBERTa$_{\text{large}}$	62.34	64.27	52.54	62.85	64.79	53.01
SAIS (Xiao et al., 2022)	RoBERTa$_{\text{large}}$	62.23	65.17	55.84	63.44	65.11	55.67
DREEAM (teacher)	RoBERTa$_{\text{large}}$	61.71$_{\pm 0.09}$	63.49$_{\pm 0.10}$	54.15$_{\pm 0.11}$	61.62	63.55	54.01
+ Inference-stage Fusion		62.29$_{\pm 0.23}$	64.20$_{\pm 0.23}$		62.12	64.27	
(b) with Distantly-Supervised Data							
KD-DocRE (Tan et al., 2022a)	BERT$_{\text{base}}$	63.38	64.81	-	62.56	64.76	-
DREEAM (student)	BERT$_{\text{base}}$	63.47$_{\pm 0.02}$	65.30$_{\pm 0.03}$	**55.68**$_{\pm 0.04}$	63.31	65.30	**55.43**
+ Inference-Stage Fusion		**63.92**$_{\pm 0.02}$	**65.83**$_{\pm 0.04}$		**63.73**	**65.87**	
SSAN (Xu et al., 2021a)	RoBERTa$_{\text{large}}$	63.76	65.69	-	63.78	65.92	-
KD-DocRE (Tan et al., 2022a)	RoBERTa$_{\text{large}}$	65.27	67.12	-	65.24	67.28	-
DREEAM (student)	RoBERTa$_{\text{large}}$	65.24$_{\pm 0.07}$	67.09$_{\pm 0.07}$	**57.55**$_{\pm 0.07}$	65.20	67.22	**57.34**
+ Inference-Stage Fusion		**65.52**$_{\pm 0.07}$	**67.41**$_{\pm 0.04}$		**65.47**	**67.53**	

Table 2: Evaluation results on development and test sets of DocRED, with best scores **bolded**. The scores of existing methods are borrowed from corresponding papers. We group the methods first by whether they utilize the distantly-supervised data or not, then by the PLM encoder.

Following previous work, we evaluate the performance of DREEAM using BERT$_{\text{base}}$ (Devlin et al., 2019) and RoBERTa$_{\text{large}}$ (Liu et al., 2019) as the PLM encoder. The parameter for balancing ER loss with RE loss is set to 0.1 for BERT$_{\text{base}}$ and 0.05 for RoBERTa$_{\text{large}}$ when training both the teacher and the student model, chosen based on a grid search from $\lambda \in \{0.05, 0.1, 0.2, 0.3\}$. We train and evaluate DREEAM on a single Tesla V100 16GB GPU when utilizing BERT$_{\text{base}}$ and on a single NVIDIA A100 40GB GPU when utilizing RoBERTa$_{\text{large}}$. Details about hyper-parameters and running time are provided in Appendix A.

Evaluation During inference, sentences x_i with $p_i > 0.2$ computed from Equation 8 are retrieved as evidence. For the evaluation, we adopt official evaluation metrics of DocRED (Yao et al., 2019): Ign F1 and F1 for RE and Evi F1 for ER. Ign F1 is measured by removing relations present in the annotated training set from the development and test sets. We train our system five times, initialized with different random seeds, and report the average scores and standard error of these runs.

4.2 Main Results

Table 2 lists the performance of the proposed and existing methods. We select the best-performing model on the development set to make predictions on the test set and submit the predictions to CodaLab for evaluation[5].

Performance of the Student Model Table 2 shows that the student model outperforms existing systems on RE by utilizing the distantly-supervised data. In particular, when adopting BERT$_{\text{base}}$ as the PLM encoder, DREEAM performs better than KD-DocRE (Tan et al., 2022a), the previous state-of-the-art system, by 0.6/1.0 points on Ign F1/F1 for the development set. On the test set, the improvement reaches 1.1 F1 points on both Ign F1 and F1. Notably, DREEAM utilizing BERT$_{\text{base}}$ even performs comparably with SSAN utilizing RoBERTa$_{\text{large}}$ under the weakly-supervised setting (Xu et al., 2021a). When adopting RoBERTa$_{\text{large}}$ as the PLM encoder, the advantage of DREEAM remains on both development and test sets. These results support our hypothesis that ER self-training improves RE, which has not been demonstrated by any previous work.

Performance of the Teacher Model The upper half of Table 2 shows that the teacher model trained

---

[5] `https://codalab.lisn.upsaclay.fr/competitions/365`. Submissions under username `kgmr15` are from this work.

Setting	Ign F1	F1	Evi F1
(a) Teacher Model			
DREEAM	**59.60**$_{\pm0.15}$	**61.42**$_{\pm0.15}$	**52.08**$_{\pm0.10}$
w/o ER training	59.21$_{\pm0.19}$	61.01$_{\pm0.20}$	42.79$_{\pm1.65}$
(b) Student Model			
DREEAM	**63.47**$_{\pm0.02}$	65.30$_{\pm0.03}$	**55.68**$_{\pm0.04}$
w/o ER self-training	61.96$_{\pm0.39}$	63.77$_{\pm0.44}$	53.72$_{\pm0.43}$
w/o ER fine-tuning	63.34$_{\pm0.02}$	**65.50**$_{\pm0.02}$	55.27$_{\pm0.05}$
w/o both	62.13$_{\pm0.07}$	63.82$_{\pm0.08}$	47.13$_{\pm0.12}$

Table 3: Ablation studies evaluated on the DocRED development set.

on human-annotated data exhibits comparable performance to EIDER (Xie et al., 2022) on both RE and ER. Although there is a performance gap between DREEAM and SAIS, we attribute it to the difference in supervisory signals. While DREEAM incorporates RE with only relation-agnostic ER, SAIS is trained under three more tasks: coreference resolution, entity typing, and relation-specific ER (Xiao et al., 2022). These extra supervisory signals possibly contribute to the high performance of SAIS. Apart from the performance, our method has a critical advantage over previous ER-incorporated DocRE systems in memory efficiency. We provide a detailed discussion in Section 4.4.

Effectiveness of ER Self-Training Additionally, we observe that the student model leads the existing systems by a large margin on ER. As the first approach enabling weakly-supervised ER training, DREEAM utilizes considerable amounts of data without evidence annotation via self-training. The experimental results reveal that DREEAM improves over the state-of-the-art supervised approaches by approximately 2.0 points on Evi F1. Therefore, we conclude that our approach to ER self-training succeeds in acquiring evidence knowledge from the relation-distantly-supervised data with no evidence annotation.

4.3 Ablation Studies

This subsection investigates the effect of evidence-guided attention by ablation studies. All subsequent experiments adopt BERT$_{\text{base}}$ as the PLM encoder. We report scores without the inference-stage fusion strategy (Xie et al., 2022).

Teacher Model Firstly, we examine how guiding attention with evidence helps RE training on human-annotated data. We train a variant of our teacher model without ER training and evaluate its performance on the DocRED development set. In general, disabling ER training reduces the model to a baseline similar to ATLOP (Zhou et al., 2021)[6].

As presented in Table 3 (a), the RE performance of our system decreases without ER training. This observation supports the hypothesis that guiding attention with evidence is beneficial to improving RE. We further visualize the token importance $\boldsymbol{q}^{(s,o)}$ for some instances to investigate the effect of evidence-guided training and find that our method succeeds in guiding the attention to focus more on relevant contexts. The details can be found in Appendix B.

Additionally, we retrieve evidence from the ER-disabled model as sentences with importance higher than the pre-defined threshold. By doing so, we find that the Evi F1 is not far from its evidence-aware counterpart. This observation indicates that ER is a task highly coupled with RE.

Student Model Next, we investigate the student model trained on distantly-supervised data and fine-tuned on human-annotated data. The aim is to examine the effect of guiding attention with evidence at various stages of training. To this end, we remove ER supervisory signals from the student model during the training on distantly-supervised and human-annotated data. The baseline excludes ER supervision from both stages, pre-trained on distantly-supervised data and then finetuned on human-annotated data for only RE.

As shown in Table 3 (b), DREEAM without ER self-training performs comparably to the baseline, while DREEAM without ER fine-tuning performs comparably to the original model with no ablations. These results indicate that ER self-training is more essential than ER fine-tuning for the student model. On the one hand, we observe that disabling ER self-training on massive data causes a huge loss of evidence knowledge that cannot be recovered by finetuning on the much smaller evidence-annotated dataset. On the other hand, we can conclude that DREEAM succeeds in retrieving evidence knowledge from the data without any evidence annotation, demonstrating the effectiveness of our ER self-training strategy.

4.4 Memory Efficiency

This subsection discusses the memory inefficiency issue in previous ER approaches and shows how DREEAM solves it. Previous approaches regard

---

[6] The difference between ATLOP and our baseline is that our baseline utilizes the last three layers of PLM to obtain embeddings, whereas ATLOP adopts only the final layer.

Method	Memory (GiB)	Trainable Params. (M)
(a) without ER Module		
ATLOP (Zhou et al., 2021)	10.8	115.4
SSAN (Xu et al., 2021a)	6.9	113.5
KD-DocRE (Tan et al., 2022a)	15.2	200.1
(b) with ER Module		
EIDER (Xie et al., 2022)	43.1	120.2
SAIS (Xiao et al., 2022)	46.2	118.0
DREEAM (proposed)	11.8	115.4

Table 4: Memory consumption and the number of trainable parameters of DREEAM and existing methods.

ER as a separate task from RE that requires extra neural network layers to solve (Huang et al., 2021a; Xie et al., 2022; Xiao et al., 2022). To perform ER, all of them introduce a bilinear evidence classifier that receives an entity-pair-specific embedding and a sentence embedding as inputs. For example, EIDER computes an evidence score for sentence x_i with regard to entity pair (e_s, e_o) as below:

$$\mathrm{P}(x_i|e_s, e_o) = \sigma(\boldsymbol{x}_i \mathsf{W} \boldsymbol{c}^{(s,o)} + \boldsymbol{b}), \tag{14}$$

where $\boldsymbol{x}_i$ is a sentence embedding, $\boldsymbol{c}^{(s,o)}$ is the localized context embedding computed from Equation 3, W and $\boldsymbol{b}$ are trainable parameters. EIDER and other existing systems thus need to compute over all combinations of (sentence, entity pair). Specifically, consider a document D with n sentences $\mathcal{X}_D = \{x_1, x_2, \ldots, x_n\}$ and m entities $\mathcal{E}_D = \{e_1, e_2, \ldots, e_m\}$, yielding $m \times (m-1)$ entity pairs. To obtain evidence scores, EIDER must perform bilinear classification $n \times m \times (m-1)$ times via Equation 14, resulting in huge memory consumption. In contrast, DREEAM takes the summations of attention weights over tokens as evidence scores, thus introducing neither new trainable parameters nor expensive matrix computations. Hence, we see that DREEAM is more memory-efficient than its competitors.

Table 4 summarizes the memory consumption and the number of trainable parameters when utilizing BERT$_{\text{base}}$ as the PLM encoder for existing and proposed methods. Values are measured when training the systems using the corresponding official repositories with a batch size of four[7]. We observe that the memory consumption of DREEAM is only 27.4% of EIDER and 25.5% of SAIS. Notably, DREEAM also consumes less memory than KD-DocRE, underscoring the memory efficiency of our proposed method.

Statistics	DocRED	Re-DocRED
# rel.	38,180	85,932
# rel. w/o evi.	1,421 (3.7%)	38,672 (45.0%)

Table 5: Statistics of relation triples in the training set of DocRED and Re-DocRED. *rel.* stands for relation triples and *rel. w/o evi.* stands for relation triples without evidence sentences.

Method	Ign F1	F1
(a) without Distantly-Supervised Data		
ATLOP (Zhou et al., 2021)	76.82	77.56
DocuNet (Zhang et al., 2021)	77.26	77.87
KD-DocRE (Tan et al., 2022a)	77.60	78.28
DREEAM	$77.34_{\pm0.19}$	$77.94_{\pm0.15}$
+ Inference-Stage Fusion	$79.66_{\pm0.39}$	$80.73_{\pm0.38}$
(b) with Distantly-Supervised Data		
ATLOP (Zhou et al., 2021)	78.52	79.46
DocuNet (Zhang et al., 2021)	78.52	79.46
KD-DocRE (Tan et al., 2022a)	80.32	81.04
DREEAM	$78.67_{\pm0.17}$	$79.35_{\pm0.18}$
+Inference-Stage Fusion	**80.39**$_{\pm0.03}$	**81.44**$_{\pm0.04}$

Table 6: Evaluation results on the test set of Re-DocRED, with best scores **bolded**. PLM encoder is aligned to RoBERTa-large. The scores of existing methods are borrowed from Tan et al. (2022b).

4.5 Performance on Re-DocRED

Although DocRED is a widely used benchmark, recent works have pointed out that annotations of the dataset are incomplete (Huang et al., 2022; Xie et al., 2022; Tan et al., 2022b). To paraphrase, many relation triples in DocRED are missing in human annotations, biasing the dataset with many false negatives. Tan et al. (2022b) thus proposed Re-DocRED, a more reliable benchmark for DocRE that revises DocRED to alleviate the false negative issue. In this subsection, we evaluate DREEAM on Re-DocRED to verify the soundness of our proposed method.

Similar to Section 4.2, we conducted experiments under two different settings: (a) a fully-supervised setting without distantly-supervised data and (b) a weakly-supervised setting utilizing distantly-supervised data. Notably, Re-DocRED introduces new relation triples without providing accurate evidence sentences. As shown in Table 5, compared with DocRED, the training set of Re-DocRED contains much more relation triples without evidence sentences. DREEAM trained on Re-DocRED could thus be inaccurate on ER, biased by the considerable amount of missing evidence. Therefore, during ER self-training of the student model, we adopt token evidence distributions predicted by a teacher model trained on DocRED as

---

[7] The value of EIDER is different from the original paper because we enable ER evaluations during training.

the supervisory signal. The student model is further finetuned on Re-DocRED to obtain more reliable knowledge about RE.

Table 6 compares the performance of DREEAM against existing methods. We observe that DREEAM outperforms existing methods under both the fully-supervised setting and the weakly-supervised setting. The observation indicates the soundness of our proposed method.

5 Related Work

DocRE Recent work has extended the scope of relation extraction task from sentence to document (Peng et al., 2017; Quirk and Poon, 2017; Yao et al., 2019). Compared with its sentence-level counterpart, DocRE is a more realistic and challenging setting, aiming at extracting both intra-sentence and inter-sentence relations. Although commonly-used benchmarks for DocRE include DocRED (Yao et al., 2021), CDR (Li et al., 2016) and GDA (Wu et al., 2019), only DocRED contains evidence annotation and massive pre-processed data obtained from relation distant supervision. Therefore, we adopt DocRED as our test bed.

Transformer-based DocRE Modeling DocRE with a Transformer-based system has been a popular and promising approach, outperforming its graph-based counterparts (Zeng et al., 2020, 2021; Xu et al., 2021b). One of the major topics of these systems is a better utilization of long-distance token dependencies captured by the PLM encoder. Zhang et al. (2021) formulate DocRE as a semantic segmentation task and introduce a U-Net (Ronneberger et al., 2015) on top of the PLM encoder to capture local and global dependencies between entities. Zhou et al. (2021) propose localized contextual pooling to focus on tokens relevant to each entity pair. Based on their work, Tan et al. (2022a) adopt an axial attention module to perform two-hop reasoning and capture the dependencies between relation triples. These designs provide no supervision on token dependencies, expecting the model to capture them implicitly during training. In contrast, we provide explicit supervision for token dependencies by utilizing evidence information.

ER in DocRE This study is not the first to incorporate evidence information into DocRE. Huang et al. (2021b) first report that heuristically selecting evidence sentences boosts the performance of DocRE systems. Huang et al. (2021a), Xie et al. (2022) and Xiao et al. (2022) train neural classifiers to automatically retrieve evidence together with RE. However, we perform ER with neither heuristic rules nor neural classifiers. Furthermore, our approach can be used for ER self-training on data without evidence annotations.

Distant Supervision Distant supervision has been widely adopted as a technique to generate automatically-labeled data for RE (Mintz et al., 2009; Quirk and Poon, 2017; Xiao et al., 2020). The method assumes that if a sentence contains an entity pair that participates in a known relation in a knowledge base (KB), the sentence probably expresses that relation. Thus unlabeled text can be aligned with a KB using entities as anchors, with each match distantly supervised by the relation described in the KB. Yao et al. (2019) apply the technique to annotate relations in documents automatically. In this work, we directly adopt those documents for ER self-training.

6 Conclusion

We have introduced methods to improve the usage of ER in DocRE. First, we propose DREEAM, a memory-efficient method to reduce the computation cost of ER. Unlike existing approaches that train an evidence classifier for ER, DREEAM directly supervises the attention to concentrate more on evidence than on others. Next, we propose to employ DREEAM in a weakly-supervised setting to compensate for the shortage of human annotations. Instead of gold evidence annotated by humans, we adopt evidence predictions from a teacher model trained on human-annotated data as the supervisory signal to realize ER self-training on unlabeled data. Experiments on the DocRED benchmark show that DREEAM exhibits state-of-the-art performance on both RE and ER, with the help of weakly-supervised training on data obtained from distant supervision of relations. Compared with existing approaches, DREEAM performs ER with zero trainable parameters introduced, thereby reducing the memory usage to 27% or less. The soundness of DREEAM is confirmed by conducting experiments on Re-DocRED, a revised version of DocRED.

In the future, we plan to transfer the evidence knowledge of DREEAM trained on DocRED to other DocRE datasets.

Limitations

A major limitation of this work is that our method can only retrieve relation-agnostic evidence. Unlike Xiao et al. (2022), DREEAM cannot specify evidence sentences for each relation label. Therefore, when an entity pair holds multiple relations, DREEAM retrieves the same evidence regardless of the relation type, even though the evidence may be correct for some of the relations but not for others.

Ethics Statement

In this work, we have proposed a method for incorporating ER into DocRE. Our approach directly supervises the weights of attention modules within a Transformer-based PLM encoder. Inside the research community, we hope our approach can provide a new viewpoint on the explainability of document-level relation extraction systems. Furthermore, a better DocRE system will benefit the research on other tasks, such as question answering and reading comprehension. In the real world, a DocRE system with good performance can help extract useful information from unstructured text, reducing human efforts and expenses. Furthermore, as our method is memory-efficient, it is also friendly to the environment.

We also have demonstrated a use case of our method in ER self-training, utilizing massive data obtained from relation distant-supervision. Although in this work, we directly adopt the data provided by Yao et al. (2019), it is possible to extend the scale of data by utilizing numerous unstructured texts. Utilizing a wide range of unstructured texts may expose our system to the risk of vulnerable data, potentially biasing our system in the wrong direction. To mitigate the problem, we encourage performing data pre-processing to detect and remove harmful contents before training.

Acknowledgements

This paper is based on results obtained from a project, JPNP18002, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We appreciate Marco Cognetta for the generous help and valuable discussions.

References

Tongfei Chen, Yunmo Chen, and Benjamin Van Durme. 2020. Hierarchical entity typing via multi-level learning to rank. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8465–8475, Online. Association for Computational Linguistics.

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

Kevin Huang, Peng Qi, Guangtao Wang, Tengyu Ma, and Jing Huang. 2021a. Entity and evidence guided document-level relation extraction. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 307–315, Online. Association for Computational Linguistics.

Quzhe Huang, Shibo Hao, Yuan Ye, Shengqi Zhu, Yansong Feng, and Dongyan Zhao. 2022. Does recommend-revise produce reliable annotations? an analysis on missing instances in DocRED. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6241–6252, Dublin, Ireland. Association for Computational Linguistics.

Quzhe Huang, Shengqi Zhu, Yansong Feng, Yuan Ye, Yuxuan Lai, and Dongyan Zhao. 2021b. Three sentences are all you need: Local path enhanced document relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 998–1004, Online. Association for Computational Linguistics.

Robin Jia, Cliff Wong, and Hoifung Poon. 2019. Document-level n-ary relation extraction with multi-scale representation learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3693–3704, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016. Baw068.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph LSTMs. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1171–1182, Valencia, Spain. Association for Computational Linguistics.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Qingyu Tan, Ruidan He, Lidong Bing, and Hwee Tou Ng. 2022a. Document-level relation extraction with adaptive focal loss and knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1672–1681, Dublin, Ireland. Association for Computational Linguistics.

Qingyu Tan, Lu Xu, Lidong Bing, Hwee Tou Ng, and Sharifah Mahani Aljunied. 2022b. Revisiting docred – addressing the false negative problem in relation extraction.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884, New Orleans, Louisiana. Association for Computational Linguistics.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ye Wu, Ruibang Luo, Henry C. M. Leung, Hing-Fung Ting, and Tak-Wah Lam. 2019. Renet: A deep learning approach for extracting gene-disease associations from literature. In *Research in Computational Molecular Biology*, pages 272–284, Cham. Springer International Publishing.

Chaojun Xiao, Yuan Yao, Ruobing Xie, Xu Han, Zhiyuan Liu, Maosong Sun, Fen Lin, and Leyu Lin. 2020. Denoising relation extraction from document-level distant supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3683–3688, Online. Association for Computational Linguistics.

Yuxin Xiao, Zecheng Zhang, Yuning Mao, Carl Yang, and Jiawei Han. 2022. SAIS: Supervising and augmenting intermediate steps for document-level relation extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2395–2409, Seattle, United States. Association for Computational Linguistics.

Yiqing Xie, Jiaming Shen, Sha Li, Yuning Mao, and Jiawei Han. 2022. Eider: Empowering document-level relation extraction with efficient evidence extraction and inference-stage fusion. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 257–268, Dublin, Ireland. Association for Computational Linguistics.

Benfeng Xu, Quan Wang, Yajuan Lyu, Yong Zhu, and Zhendong Mao. 2021a. Entity structure within and throughout: Modeling mention dependencies for document-level relation extraction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14149–14157.

Wang Xu, Kehai Chen, Lili Mou, and Tiejun Zhao. 2022. Document-level relation extraction with sentences importance estimation and focusing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2920–2929, Seattle, United States. Association for Computational Linguistics.

Wang Xu, Kehai Chen, and Tiejun Zhao. 2021b. Document-level relation extraction with reconstruction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14167–14175.

Jie Yang and Yue Zhang. 2018. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, System Demonstrations*, pages

74–79, Melbourne, Australia. Association for Computational Linguistics.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Yunzhi Yao, Shaohan Huang, Wenhui Wang, Li Dong, and Furu Wei. 2021. Adapt-and-distill: Developing small, fast and effective pretrained language models for domains. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 460–470, Online. Association for Computational Linguistics.

Shuang Zeng, Yuting Wu, and Baobao Chang. 2021. SIRE: Separate intra- and inter-sentential reasoning for document-level relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 524–534, Online. Association for Computational Linguistics.

Shuang Zeng, Runxin Xu, Baobao Chang, and Lei Li. 2020. Double graph based reasoning for document-level relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1630–1640, Online. Association for Computational Linguistics.

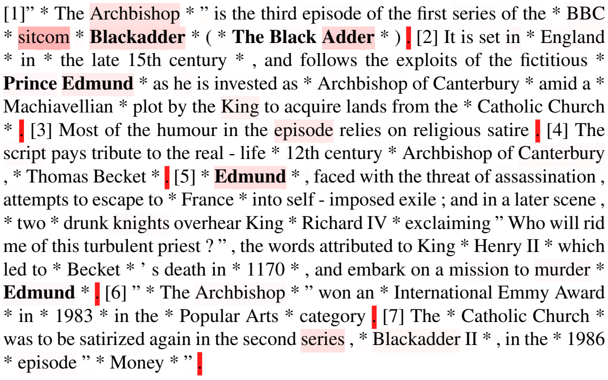
Ningyu Zhang, Xiang Chen, Xin Xie, Shumin Deng, Chuanqi Tan, Mosha Chen, Fei Huang, Luo Si, and Huajun Chen. 2021. Document-level relation extraction as semantic segmentation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3999–4006. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Heliang Zheng, Jianlong Fu, Zheng-Jun Zha, and Jiebo Luo. 2019. Learning deep bilinear transformation for fine-grained image representation. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

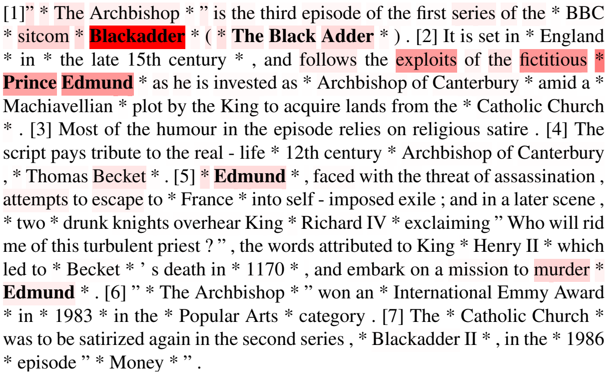
A Hyper-Parameters and Runtime

We adopt AdamW as the optimizer (Loshchilov and Hutter, 2019) and apply a linear warmup for the learning rate at the first 6% steps. Important hyper-parameters are shown in Table 7, which are mainly borrowed from existing works. Specifically, we borrow hyper-parameters from Zhou et al. (2021) to train the teacher model and borrow those from Tan et al. (2022a) to train and finetune the student model. The only exception is the number of epochs for training the student model, which is determined by a grid search from $\{2, 5, 8, 10\}$.



[1]” * The Archbishop * ” is the third episode of the first series of the * BBC * sitcom * **Blackadder** * (* **The Black Adder** *) . [2] It is set in * England * in * the late 15th century * , and follows the exploits of the fictitious * **Prince Edmund** * as he is invested as * Archbishop of Canterbury * amid a * Machiavellian * plot by the King to acquire lands from the * Catholic Church * . [3] Most of the humour in the episode relies on religious satire . [4] The script pays tribute to the real - life * 12th century * Archbishop of Canterbury , * Thomas Becket * . [5] * **Edmund** * , faced with the threat of assassination , attempts to escape to * France * into self - imposed exile ; and in a later scene , * two * drunk knights overhear King * Richard IV * exclaiming ” Who will rid me of this turbulent priest ? ” , the words attributed to King * Henry II * which led to * Becket * ’ s death in * 1170 * , and embark on a mission to murder * **Edmund** * . [6] ” * The Archbishop * ” won an * International Emmy Award * in * 1983 * in the * Popular Arts * category . [7] The * Catholic Church * was to be satirized again in the second series , * Blackadder II * , in the * 1986 * episode ” * Money * ” .

(a) Before attention guidance.



[1]” * The Archbishop * ” is the third episode of the first series of the * BBC * sitcom * **Blackadder** * (* **The Black Adder** *) . [2] It is set in * England * in * the late 15th century * , and follows the exploits of the fictitious * **Prince Edmund** * as he is invested as * Archbishop of Canterbury * amid a * Machiavellian * plot by the King to acquire lands from the * Catholic Church * . [3] Most of the humour in the episode relies on religious satire . [4] The script pays tribute to the real - life * 12th century * Archbishop of Canterbury , * Thomas Becket * . [5] * **Edmund** * , faced with the threat of assassination , attempts to escape to * France * into self - imposed exile ; and in a later scene , * two * drunk knights overhear King * Richard IV * exclaiming ” Who will rid me of this turbulent priest ? ” , the words attributed to King * Henry II * which led to * Becket * ’ s death in * 1170 * , and embark on a mission to murder * **Edmund** * . [6] ” * The Archbishop * ” won an * International Emmy Award * in * 1983 * in the * Popular Arts * category . [7] The * Catholic Church * was to be satirized again in the second series , * Blackadder II * , in the * 1986 * episode ” * Money * ” .

(b) After attention guidance.

Figure 3: Heatmaps of token importance for localized context pooling before and after guiding the attention with evidence when deciding the relation for entity pair (*Prince Edmund*, *The Black Adder*). The gold relation is *present in work* with evidence sentences 1 and 2. Deeper the color, the larger the value.

The average running time spent for our system at each training stage is shown in Table 8. Note that we employ a single Tesla V100 16GB GPU when utilizing BERT$_{\text{base}}$ and a single NVIDIA A100 40GB GPU when utilizing RoBERTa$_{\text{large}}$.

B Visualization: Evidence-Guided Attention

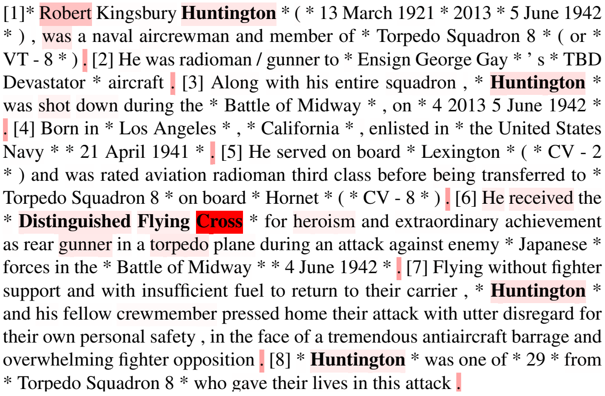
As introduced in Section 3.1, evidence knowledge of DREEAM originates from sentence-level supervision. We hypothesize that sentence-level supervision, from a more macro perspective, should improve its micro counterpart of token-level focusing. To test the hypothesis, we examine the token-level evidence distribution for localized context pool-

Hyperparam.	Train (teacher)		Train (student)		Finetune (student)	
	BERT$_{\text{base}}$	RoBERTa$_{\text{large}}$	BERT$_{\text{base}}$	RoBERTa$_{\text{large}}$	BERT$_{\text{base}}$	RoBERTa$_{\text{large}}$
# Epoch	30	30	2	5	10	10
lr for encoder	5e-5	3e-5	3e-5	1e-5	1e-6	1e-6
lr for classifier	1e-4	1e-4	1e-4	5e-5	3e-6	3e-6
max gradient norm	1.0	1.0	5.0	5.0	2.0	2.0

Table 7: Hyper-parameters in training.

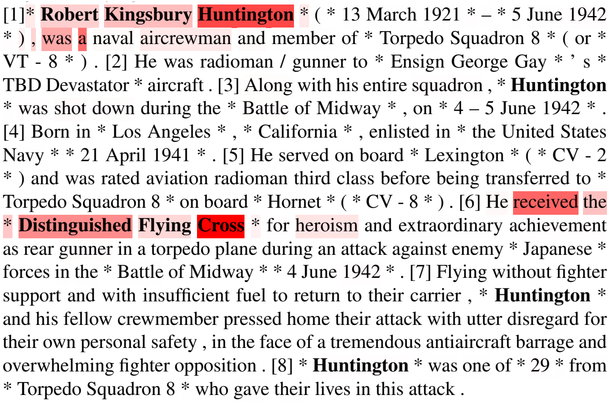
Phase	BERT$_{\text{base}}$	RoBERTa$_{\text{large}}$
Train (teacher)	1h18min	1h18min
Train (student)	2h55min	6h12min
Finetune (student)	26min	29min

Table 8: Runtime for each training stage.



[1]* Robert Kingsbury **Huntington** * (* 13 March 1921 * 2013 * 5 June 1942 *) , was a naval aircrewman and member of * Torpedo Squadron 8 * (or * VT - 8 *) . [2] He was radioman / gunner to * Ensign George Gay * ’ s * TBD Devastator * aircraft . [3] Along with his entire squadron , * **Huntington** * was shot down during the * Battle of Midway * , on * 4 2013 5 June 1942 * . [4] Born in * Los Angeles * , * California * , enlisted in * the United States Navy * * 21 April 1941 * . [5] He served on board * Lexington * (* CV - 2 *) and was rated aviation radioman third class before being transferred to * Torpedo Squadron 8 * on board * Hornet * (* CV - 8 *) . [6] He received the * **Distinguished Flying Cross** * for heroism and extraordinary achievement as rear gunner in a torpedo plane during an attack against enemy * Japanese * forces in the * Battle of Midway * * 4 June 1942 * . [7] Flying without fighter support and with insufficient fuel to return to their carrier , * **Huntington** * and his fellow crewmember pressed home their attack with utter disregard for their own personal safety , in the face of a tremendous antiaircraft barrage and overwhelming fighter opposition . [8] * **Huntington** * was one of * 29 * from * Torpedo Squadron 8 * who gave their lives in this attack .

(a) Before attention guidance.



[1]* **Robert Kingsbury Huntington** * (* 13 March 1921 * – * 5 June 1942 *) , was a naval aircrewman and member of * Torpedo Squadron 8 * (or * VT - 8 *) . [2] He was radioman / gunner to * Ensign George Gay * ’ s * TBD Devastator * aircraft . [3] Along with his entire squadron , * **Huntington** * was shot down during the * Battle of Midway * , on * 4 – 5 June 1942 * . [4] Born in * Los Angeles * , * California * , enlisted in * the United States Navy * * 21 April 1941 * . [5] He served on board * Lexington * (* CV - 2 *) and was rated aviation radioman third class before being transferred to * Torpedo Squadron 8 * on board * Hornet * (* CV - 8 *) . [6] He received the * **Distinguished Flying Cross** * for heroism and extraordinary achievement as rear gunner in a torpedo plane during an attack against enemy * Japanese * forces in the * Battle of Midway * * 4 June 1942 * . [7] Flying without fighter support and with insufficient fuel to return to their carrier , * **Huntington** * and his fellow crewmember pressed home their attack with utter disregard for their own personal safety , in the face of a tremendous antiaircraft barrage and overwhelming fighter opposition . [8] * **Huntington** * was one of * 29 * from * Torpedo Squadron 8 * who gave their lives in this attack .

(b) After attention guidance.

Figure 4: Heatmaps of token importance for localized context pooling before and after guiding the attention with evidence when deciding the relation for entity pair (*Robert Kingsbury Huntington*, *Distinguished Flying Cross*). The gold relation is *award received* with evidence sentences 1 and 6. Deeper the color, the larger the value.

ing. Specifically, we utilize heatmaps to visualize $\boldsymbol{q}^{(s,o)}$ and observe the differences before and after evidence-guided training.

Results are shown in Figure 3 and 4. We adopt the toolkit developed by Yang and Zhang (2018). It is obvious that the distribution is more focused on sentences 1 and 2 in Figure 3(b) than in Figure 3(a). Before training the evidence-guided attention, the model tends to focus on the period of each sentence. Guiding the attention with evidence helps the model to focus more on sentences 1 and 2, as well as the critical tokens providing a clue for relation classification, such as *fictitious* in Figure 3 and *received* in Figure 4.

Span-based Named Entity Recognition by Generating and Compressing Information

Nhung T. H. Nguyen[1], **Makoto Miwa**[2,3] and **Sophia Ananiadou**[1,3,4]
[1]National Centre for Text Mining, Department of Computer Science,
The University of Manchester, United Kingdom
[2]Toyota Technological Institute, Japan
[3]Artificial Intelligence Research Center (AIRC),
National Institute of Advanced Industrial Science and Technology (AIST), Japan
[4]Alan Turing Institute, United Kingdom
`{nhung.nguyen, sophia.ananiadou}@manchester.ac.uk`
`makoto-miwa@toyota-ti.ac.jp`

Abstract

The information bottleneck (IB) principle has been proven effective in various NLP applications. The existing work, however, only used either generative or information compression models to improve the performance of the target task. In this paper, we propose to combine the two types of IB models into one system to enhance Named Entity Recognition (NER). For one type of IB model, we incorporate two unsupervised generative components, span reconstruction and synonym generation, into a span-based NER system. The span reconstruction ensures that the contextualised span representation keeps the span information, while the synonym generation makes synonyms have similar representations even in different contexts. For the other type of IB model, we add a supervised IB layer that performs information compression into the system to preserve useful features for NER in the resulting span representations. Experiments on five different corpora indicate that jointly training both generative and information compression models can enhance the performance of the baseline span-based NER system. Our source code is publicly available at `https://github.com/nguyennth/joint-ib-models`.

1 Introduction

Tishby et al. (1999) introduced the information bottleneck (IB) method to compress representation while preserving meaningful information. The method has been incorporated in many state-of-the-art (SOTA) deep models such as Variational Autoencoder (VAE) (Kingma and Welling, 2014; Rezende et al., 2014), and Deep Variational Information Bottleneck (Deep VIB) (Alemi et al., 2017). Those deep models can be divided into supervised generative models (e.g., Deep VIB) and unsupervised ones (e.g., VAEs) (Voloshynovskiy et al., 2019).

Both VAE and VIB have been applied to NLP applications. For example, Effland and Collins (2019) and Chen et al. (2018) proposed to use VAE in sequence labelling tasks such as POS tagging and NER. Meanwhile, Wang et al. (2022) used VIB to tackle OOV issues in NER. The two types of IB have also been employed in other tasks, such as dialogue response generation (Chen et al., 2022), parsing (Li and Eisner, 2019), paraphrase generation for MT (Ormazabal et al., 2022), and text summarisation (West et al., 2019), to name a few. Such previous work only used one type of IB model (either VAE or VIB) in their system, and it is unclear whether we can effectively combine the two types of IB models.

The NER task has been typically approached using sequence models such as BiLSTM (Lample et al., 2016) and BERT (Devlin et al., 2019; Lee et al., 2019; Beltagy et al., 2019). At the same time, we have seen the rise of span-based models (Sohrab and Miwa, 2018; Zheng et al., 2019; Tan et al., 2020; Xia et al., 2019; Xu et al., 2021; Fu et al., 2021; Li et al., 2021), which are simple and effective. Using a span-based model, we can directly represent and manipulate span representations.

This paper investigates the effects of combining the two IB types for the NER task. To that end, we jointly train the span-based NER system with two VAE components and one VIB component. The first VAE component is span reconstruction, used to reconstruct original spans. This component is similar to Sentence VAEs (Bowman et al., 2016), but the model learns only from spans instead of sentences. The second VAE one is syn-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1984–1996
May 2-6, 2023

onym generation, used to generate synonym(s) of a span. We first collect synonyms of each span from an external knowledge base (KB) and then train a VAE model that can generate the corresponding synonyms given a span. By adding the synonym generation into the model, we indirectly inject semantic information from synonyms into the span representation. The last component is a VIB component (Mahabadi et al., 2021) introduced into the system to compress span representations while keeping useful features for NER.

We evaluate the proposed model on five different corpora: BC5CDR (Li et al., 2016), GENIA (Kim et al., 2003), MedMention-21st (Mohan and Li, 2019), NCBI Disease (Dogan et al., 2014), and ShARe/CLEFE (Suominen et al., 2013). For the synonym generation component, we identify synonyms of each span by performing exact matching against mentions in Unified Medical Language System (UMLS) (Bodenreider, 2004). Experimental results show that by incorporating the two IB types into a span-based NER, we can improve the performance over the baseline span-based model on BC5CDR-Disease, GENIA, MedMention, and NCBI datasets. In the case of GENIA–one of the most popular corpora for nested entities, the proposed model could perform favourably compared with current SOTA systems, even with the simple span-based baseline model without any recent enhancements like boundary detection (Tan et al., 2020; Xu et al., 2021). Furthermore, such boundary-enhanced models can benefit from our approach.

We additionally performed some analysis on the intermediate output of the proposed model. We observed that when jointly trained VAEs with a NER task, the latent variable is restructured satisfactorily towards the task. Similarly, posteriors estimated by the VIB component are clustered neatly even though the input information is compressed. Such distinguishable clusters are potentially helpful for entity linking (Liu et al., 2021). We also found that synonym generation helped improve the quality of span reconstruction and the NER performance.

In summary, the contributions of our paper are:

- This is the first study that investigates the impact of combining two IB methods on NER.
- Through experiments on five different corpora, we demonstrate that the joint model can improve the baseline performance in most cases.
- In-depth analyses on the intermediate output of the joint model indicate that each component plays a different role in enhancing span representations as expected.

2 Related Work

2.1 Span-based NER

Traditionally, sequence models such as BiLSTM (Lample et al., 2016) and BERT (Devlin et al., 2019; Lee et al., 2019; Beltagy et al., 2019) have been used to tackle the task of NER, producing state-of-the-art (SOTA) performance. However, those models could not perform on overlapping entities, i.e., a span has more than one named entity category or nested entities. To address this issue, Sohrab and Miwa (2018) proposed the span-based approach. In this approach, all possible spans are exhaustively generated given a specific span length. The span representation was calculated based on a pre-trained language model and then classified to a corresponding entity type by a linear layer. Following the suite, numerous studies have shown that span-based approaches to NER could produce SOTA performance (Zheng et al., 2019; Tan et al., 2020; Xia et al., 2019; Sohrab et al., 2019; Xu et al., 2021; Fu et al., 2021; Li et al., 2021; Yu et al., 2022).

Fu et al. (2021) designed SpanNER that learns the representation of a span based on its token representation and the span length embedding. SpanNER can also make ensemble predictions from both span-based and sequence labelling systems. Similarly, Yu et al. (2022) proposed SNER that represents a span by considering the context embedding from BERT, i.e., the CLS embedding. Ouchi et al. (2020) proposed to learn the similarity between spans using instance-based learning. The model will assign the class label based on its similar training span at inference time. Xu et al. (2021) used a supervised multi-head self-attention mechanism, where each head corresponds to one entity type, to construct the word-level correlations for each type.

Following the work mentioned above, this paper also focuses on span-based models. Our model, however, learns span representation differently. In particular, we update the span representation using span reconstruction and synonym generation. We then compress it by selecting relevant features using the IB method. It is noted that our model can potentially be incorporated into existing span-based models.

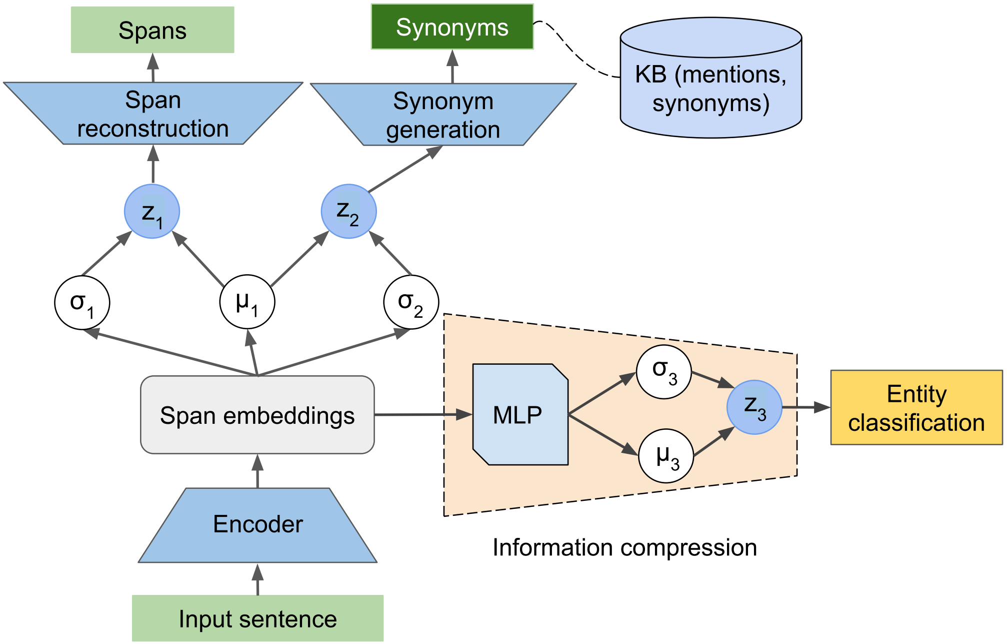


Figure 1: Span-based NER is jointly trained with one supervised VIB component: information compression, and two unsupervised VAE components: span reconstruction and synonym generation. The encoder is shared across the components. Synonyms are collected from an external knowledge base (KB).

2.2 Information Bottleneck in NLP

The Information Bottleneck (IB) method has been applied to various NLP applications. We divide those studies into two main groups: generative models and information compression models.

In the first group, we have studies that used IB as generative models, namely Variational Autoencoders (VAEs). Effland and Collins (2019) proposed a sequence labelling NER model that treats a neural CRF as the amortised approximate posterior in a discrete structured VAE. Meanwhile, Chen et al. (2018) applied neural variational methods to sequence labelling by combining a latent variable generative model and a discriminative labeller. VAEs were also employed in dialogue response generation (Chen et al., 2022) and relation extraction (Yuan and Eldardiry, 2021; Christopoulou et al., 2021).

The second group are studies that used IB as information compression, namely Variational Information Bottleneck (VIB). Mahabadi et al. (2021) was one of the first studies using VIB to finetune low-resource target tasks. Wang et al. (2022) employed the information bottleneck principle to tackle OOV issues in NER. Information compression was also used in parsing (Li and Eisner, 2019), paraphrase generation for MT (Ormazabal et al., 2022), research replication prediction (Luo et al., 2022), text classification (Zhang et al., 2022), improving the attention's reliability (Zhou et al., 2021) and text summarisation (West et al., 2019).

Unlike previous studies, we combine generative and information compression models for the task of NER. In our model, span reconstruction ensures that the span representation keeps the span information, while synonym generation makes similar synonyms have similar representations even in different contexts. Meanwhile, information compression helps suppress irrelevant features, addressing overfitting (if any). To the best of our knowledge, this is the first paper investigating such a joint system for span-based NER.

3 Methods

The overall framework of our model is illustrated in Figure 1. We first encode span embeddings using a transformer-based network. The output from the encoder is then used in three components: (1) entity classification that classifies input spans into an entity type or non-entity using a supervised VIB framework (Mahabadi et al., 2021); (2) span reconstruction that recovers input spans of gold entities; and (3) synonym generation that generates synonyms for a gold entity. Similarly to a multitask setting, we simultaneously train the three components. We only run the first component at the inference step to predict named entities.

3.1 Span Reconstruction and Synonym Generation

3.1.1 Encoder

Given an input sentence with n words: $\{w_0, w_1, ..., w_n\}$, we use a multi-layer transformer-based encoder (Devlin et al., 2019) to encode the sentence. As a result, we have the following contextualised sub-word vectors $\{\boldsymbol{v}_0, \boldsymbol{v}_1, ..., \boldsymbol{v}_m\}$. Similarly to Sohrab and Miwa (2018), we then exhaustively extract all possible spans with a maximum length of sl from the input sentence. Each span embedding is calculated as follows:

$$\boldsymbol{s}_{i,j} = \left[\boldsymbol{v}_i; \frac{\Sigma_{t=i}^{j} \boldsymbol{v}_t}{j-i+1}; \boldsymbol{v}_j \right], \tag{1}$$

where i and j are the start and end positions of the span, $\boldsymbol{v}_t$ is the embedding of the t-th sub-word, and $[;;]$ denotes the concatenation operation.

We then apply two linear layers on top of the span embeddings to construct the parameters of a posterior distribution $q(\boldsymbol{z}|\boldsymbol{s})$ using the following equations:

$$\begin{aligned} \boldsymbol{\mu} &= \boldsymbol{W}_{\mu}\boldsymbol{s} + \boldsymbol{b}_{\mu}, \\ \boldsymbol{\sigma}^2 &= \boldsymbol{W}_{\sigma}\boldsymbol{s} + \boldsymbol{b}_{\sigma}, \end{aligned} \tag{2}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the parameters of a multivariate Gaussian, representing the feature space of the span; $\boldsymbol{W}$ and $\boldsymbol{b}$ are weights and biases of the linear layers, respectively. The posterior distribution is approximated via a latent variable $\boldsymbol{z}$ using the reparameterisation trick (Kingma and Welling, 2014) as follows

$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma}\boldsymbol{\epsilon}, \text{where } \boldsymbol{\epsilon} \sim \boldsymbol{\mathcal{N}}(0, 1). \tag{3}$$

In the case that we train both span reconstruction and synonym generation, we have different parameters of $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$, respectively. To encourage span and synonyms to distribute closely, $\boldsymbol{\mu}$ is shared between them.

3.1.2 Decoders

Our decoder is an LSTM network that greedily reconstructs the input span or generates all the corresponding synonyms (if any) in an autoregressive manner. Given latent z from the encoder, we first use $\boldsymbol{z}$ to initialise the hidden state of the decoder via a linear layer transformation. We then form the input of the decoder with the teacher forcing strategy (Williams and Zipser, 1989), i.e., we concatenate $\boldsymbol{z}$ with the representation of each word w_t[1] in a given gold span (for span reconstruction) or a gold synonym (for synonym generation).

3.1.3 Learning

To train VAEs, we maximise the Evidence Lower BOund (ELBO) that includes two losses: the reconstruction loss and the Kullback-Leibler divergence (D_{KL}). The reconstruction loss is the cross entropy loss between an actual span and its reconstruction or synonym. The D_{KL} is calculated based on a prior distribution ($p(\boldsymbol{z})$) and the posterior distribution ($q(\boldsymbol{z}|\boldsymbol{s})$) produced by the decoder. In the case of span reconstruction (SR), the loss for a span s is

$$\begin{aligned} L_{SR}(\boldsymbol{\theta}_1, \boldsymbol{\phi}_1) = & \mathop{\mathbb{E}}_{\boldsymbol{z}_1 \sim q_{\boldsymbol{\phi}_1}(\boldsymbol{z}_1|\boldsymbol{s})} [\log(p_{\boldsymbol{\theta}_1}(\boldsymbol{s}|\boldsymbol{z}_1))] \\ & - D_{KL}(q_{\boldsymbol{\phi}_1}(\boldsymbol{z}_1|s) \parallel p_{\boldsymbol{\theta}_1}(\boldsymbol{z}_1)). \end{aligned} \tag{4}$$

Similarly, with the synonym generation (SG), the loss is

$$\begin{aligned} L_{SG}(\boldsymbol{\theta}_2, \boldsymbol{\phi}_2) = & \mathop{\mathbb{E}}_{\boldsymbol{z}_2 \sim q_{\boldsymbol{\phi}_2}(\boldsymbol{z}_2|\boldsymbol{s})} [\log(p_{\boldsymbol{\theta}_2}(\boldsymbol{s}|\boldsymbol{z}_2))] \\ & - D_{KL}(q_{\boldsymbol{\phi}_2}(\boldsymbol{z}_2|\boldsymbol{s}) \parallel p_{\boldsymbol{\theta}_2}(\boldsymbol{z}_2)). \end{aligned} \tag{5}$$

In Equations 4 and 5, $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are weights and biases of the network, respectively. We attach a subscript to each parameter to denote that the parameter belongs to the span reconstruction component or the synonym generation one.

3.2 Entity Classification with Supervised IB

The main objective of the supervised IB is to preserve the information about the target class(es) in the latent while filtering out irrelevant information from the input (Voloshynovskiy et al., 2019). As a result, the objective loss function for supervised IB is based on the compression loss and the prediction loss, as shown in Equation 6.

$$\begin{aligned} L_{VIB}(\boldsymbol{\theta}_3, \boldsymbol{\phi}_3) = & \, \beta \mathop{\mathbb{E}}_{\boldsymbol{s}}[D_{KL}(p_{\boldsymbol{\theta}_3}(\boldsymbol{z}_3|\boldsymbol{s}), r(\boldsymbol{z}_3))] \\ & + \mathop{\mathbb{E}}_{\boldsymbol{z}_3 \sim p_{\boldsymbol{\theta}_3}(\boldsymbol{z}_3|\boldsymbol{s})} [-\log q_{\boldsymbol{\phi}_3}(y|\boldsymbol{z}_3)], \end{aligned} \tag{6}$$

where $r(\boldsymbol{z}_3)$ is an estimate of the prior probability $p_{\boldsymbol{\theta}_3}(\boldsymbol{z}_3)$, β is in a range of $[0, 1]$, and y is the true label of the input span. Similarly to previous equations, $\boldsymbol{\theta}_3$ and $\boldsymbol{\phi}_3$ are weights and biases of the network, respectively.

Following Mahabadi et al. (2021), we use a multi-layer perceptron (MLP) with two linear layers to compute the compressed representation of a span. We approximate $q_{\boldsymbol{\phi}_3}(y|\boldsymbol{z}_3)$ using another

---

[1] We pass a special start symbol as w_0.

Corpus	Train			Dev			Test			Entity
	#Doc.	#Ent.	Syn_cov	#Doc.	#Ent.	Syn_cov	#Doc.	#Ent.	Syn_cov	
NCBI	592	5,134	79.25%	100	787	74.46%	100	960	75.62%	Disease
BC5-Dis	500	4,182	87.18%	500	4,244	87.18%	500	4,424	89.69%	Disease
BC5-Chem	500	5,203	93.14%	500	5,347	94.58%	500	5,385	94.00%	Chemical
ShARe	149	3,630	77.96%	50	1,413	77.35%	99	4,912	81.54%	Disorder
GENIA	1,599	45,036	55.98%	189	4,274	49.46%	212	5,346	51.05%	5 types
MM	2,635	122,241	70.21%	878	40,884	70.55%	879	40,157	71.17%	21 types

Table 1: Statistics numbers of experimental corpora. #Doc. indicates the number of documents; #Ent. indicates the number of gold entities; Syn_cov indicates the percentage of gold entities that have synonyms in UMLS.

linear layer. In particular, this layer is a binary classifier with a sigmoid function to predict the correct entity category of an input span. We use binary cross entropy (BCE) loss to compute the prediction loss (i.e., the second term of Equation 6) in our model.

3.3 Training Objective

We jointly train span reconstruction, synonym generation, and entity classification with the sum of the following optimisation objective for all the spans:

$$L = L_{VIB} + \gamma(L_{SR} + L_{SG}), \qquad (7)$$

where γ is a hyper-parameter with a range of $[0, 1]$. It is noted that during training, L_{VIB} is calculated for all spans, while L_{SR} and L_{SG} are calculated for gold spans only.

4 Experiments

4.1 Datasets

We conducted experiments on five different datasets: NCBI Disease (Dogan et al., 2014), BC5CDR (Li et al., 2016), ShARe/CLEFE (ShARe) (Suominen et al., 2013), GENIA (Kim et al., 2003), and MedMention-21st (MM) (Mohan and Li, 2019). Following previous work, we divided BC5CDR into BC5CDR-Disease (BC5-Dis) and BC5CDR-Chemical (BC5-Chem). To detect synonyms of each span, we only performed exact matching against mentions from UMLS (Bodenreider, 2004). In Table 1, we report the number of documents, the number of golden entities, and the percentage of golden entities that have synonyms in the UMLS (version 2017AA) of the experimental corpora. Since UMLS does not cover every entity, there are cases that an entity does not have any synonyms[2].

Corpus	Baseline	Joint model
BC5-Chemical	91.38	91.30
BC5-Disease	84.46	**85.25**
GENIA	76.93	**77.43**
MedMention-21st	62.78	**63.21**
NCBI	88.12	**88.29**
ShARe/CLEFE	80.88	80.77
Average	80.76	**81.04**

Table 2: F1 scores (%) on the test sets. Bold numbers indicate the joint model is better than the baseline.

4.2 Settings

Regarding the encoder, we employed the pre-trained SciBERT model (Beltagy et al., 2019). Regarding the decoder, we used an LSTM with one hidden layer; input vectors to the LSTM were extracted from resulting vectors trained on a combination of PubMed, PMC texts and the English Wikipedia (Pyysalo et al., 2013).

It is noted that before jointly training with the entity classification, we pre-trained the auxiliary components, i.e., span reconstruction and synonym generation in several epochs. The NER performance was measured based on the exact matching F1 scores calculated by the N2C2 Shared Task NER evaluation script (Henry et al., 2020). All hyper-parameter settings are detailed in Appendix A.

4.3 Results

Across all corpora, we compare the joint model in Figure 1 with a Baseline system that is a span-based model[3] using SciBERT, as reported in Table 2. With the GENIA corpus, we also collected the performance of other span-based SOTA systems for comparison, as shown in Table 3.

In Table 2, we can see that performance across the corpora differs. The proposed model produced better NER performance than the baseline ones

---

[2] In this case, we only lowercase texts in the data and UMLS before matching. Using other techniques to find synonyms in UMLS is beyond the scope of this paper.

[3] We used the span-based NER implementation by Trieu et al. (2020)

System	F1 score
Instance-based (Ouchi et al., 2020)	74.20
Boundary-aware (Zheng et al., 2019)	74.70
BENSC (Tan et al., 2020)	78.30
MHSA (Xu et al., 2021)	79.60
Baseline	76.93
Joint model	**77.43**

Table 3: F1 scores (%) on the GENIA testing set compared with other SOTA systems.

on four corpora. The exception happened with BC5CDR-Chemical and ShARe/CLEFE where the simple span-based model obtained higher F1 scores than the joint model. The corpora's characteristics are one possible reason for this. For BC5-Chemical, reconstructing chemical entities and finding their correct synonyms are more complex than the other entities (see Section 5.3). For ShAe/CLEFE, its documents are health records, not scientific papers like the others. On average, however, the joint model performed better than the baseline.

Results in Table 3 show that on the GENIA corpus, the joint model could perform better than (1) the baseline and (2) the two span-based models: the instanced-based NER model (Ouchi et al., 2021) and the boundary-aware one (Zheng et al., 2019). However, our model produced lower F1 scores than BENSC (Tan et al., 2020) and MHSA (Xu et al., 2021). This can be explained by the fact that both BENSC and MHSA specifically address nested entities by enhancing span boundary detection, while our model classifies all possible spans. Nevertheless, it is noted that our objective is not to improve the SOTA; we focus on investigating the joint model for the NER task anyway. Furthermore, our model can be incorporated into those span-based SOTA systems.

5 Analysis

5.1 Ablation Study

To evaluate the effect of each component on the proposed model, we ran the following settings on the development sets:

- SupVIB: the span-based model using the supervised IB entity classification.
- SupVIB_0: β was set to 0, meaning that the compression loss was not involved in the training stage. This setting is similar to the baseline setting, but the span embeddings are compressed by the MLP before being fed into the linear layer.
- SupVIB + SpanReco: we jointly trained span reconstruction with the SupVIB.
- SupVIB + SpanReco + SynGen (All): the full joint model with all three components.

From Table 4, we can observe the following. Firstly, using SupVIB, we could obtain higher F1 scores in most of the corpora than the baseline. The compression loss is important for SupVIB. In four over six corpora, setting $\beta = 0$ made the performance drop. Secondly, when we introduced the span reconstruction component alone into the model, except for MedMention-21st and NCBI, the component degraded the performance. We observed that SupVIB restructured embeddings differently than SupVIB+SpanReco, which probably affected the performance. Moreover, SupVIB was slightly better than SupVIB+SpanReco in distinguishing entities' boundaries.

When we introduced synonym generation into the model (i.e., the All setting), we could obtain better performance than the baseline model on five over six corpora. This indicates that indirectly injecting semantic information into the span representation is mostly helpful. The exceptions were with MedMention and ShARe/CLEFE, where the All setting produced a lower F1 score than the SupVIB+SpanReco.

By jointly training two types of IB, i.e., generative model and supervised IB, we could enhance the NER performance over the baseline on four out of six testing corpora. Especially with the two most complex, multi-category NER corpora, i.e., GENIA and MedMention-21, the joint model could outperform the baseline by a larger margin than the other corpora.

Lastly, looking at the last column in Table 4, i.e., the average F1 score across the corpora, we can see that the scores were increased from the top to the bottom. We, therefore, can confirm that adding the proposed components one after another helped improve the performance in general.

5.1.1 Impact of the Shared Parameters in VAEs

To encourage spans and their synonyms to distribute closely, we have $\boldsymbol{\mu}$ shared between the span reconstruction and the synonym generation, as in the aforementioned All model. Theoretically, we can also share $\boldsymbol{\sigma}$ between the two components. To investigate the impact of these shared parameters, we conducted two more experiments:

Setting	BC5-Chem	BC5-Dis	GENIA	MM	NCBI	ShARe	Avg.
Baseline	93.34	84.88	78.00	62.83	87.89	80.89	81.30
SupVIB_0	94.01	85.51	78.26	62.31	87.76	81.14	81.50
SupVIB	**94.05**	85.51	78.36	63.00	87.70	81.65	81.71
SupVIB + SpanReco	93.69	85.23	78.22	**63.63**	88.05	81.56	81.73
All	93.78	**85.56**	**79.30**	63.39	**88.41**	81.11	**81.93**
All (w/ shared $\boldsymbol{\sigma}$)	93.87	85.33	78.90	63.21	87.62	80.93	81.64
All (w/o shared $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$)	93.49	85.03	78.97	61.85	87.89	**81.78**	81.50

Table 4: F1 scores (%) on the development sets produced by different settings. All is the Joint model in Tables 2 and 3. The last column is the average score across the corpora. Bold numbers indicate the best performance in each corpus.

- All with shared $\boldsymbol{\sigma}$: the full model in which the two VAE components share $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$.
- All without shared $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$: the full model in which each VAE component has its own (i.e., independent) $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$.

From the last two rows in Table 4, we can see that the All model with shared $\boldsymbol{\sigma}$ made the F1 scores drop across the corpora (except for the BC5CDR-Chemical) compared with the All model. This is expected because sharing $\boldsymbol{\mu}$, i.e., sharing the mean of the two distributions, is more reasonable than sharing both of the parameters, i.e., sharing both the mean and the variance of the two distributions. However, when we have the independent $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, there is no common pattern in the performance. Compared with the All model, the All without shared $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ model produced lower scores on most of the corpora, except for the ShARe/CLEFE one. On average, the shared $\boldsymbol{\mu}$ setting (i.e., the All model) was superior to the other two.

5.2 NER Error Analysis

We classified false positive predictions by NER models into two classes:

- Category errors: denote predictions that have correct spans but wrong category,
- Span errors: denote predictions that have incorrect spans.

Regarding the BC5CDR, NCBI, and ShARe/CLEFE corpora, all false positives are due to span errors since these corpora have only one NER category. We report the percentage of span errors in false positives on the development sets of these corpora in Table 5. On BC5-Chemical and ShARe/CLEFE, the proposed model produced more span errors than the baseline one. This situation somehow aligns with the exceptionally lower performance of the proposed model on the two corpora reported in Table 2.

Corpus	Baseline	All
BC5-Chemical	**6.48**	6.50
BC5-Disease	15.21	**13.52**
NCBI	12.76	**12.36**
ShARe/CLEFE	**18.02**	19.23

Table 5: Percentage of span errors on the development sets. The lower the number, the better the model.

Corpus	Category Error		Span Error	
	Base	All	Base	All
GENIA	126	**115**	**4,133**	4,411
MM	2,447	**2,028**	37,148	**34,747**

Table 6: Number of category errors and span errors on the development sets. Base means the Baseline model. The lower the number, the better the model.

In cases of MedMention-21st and GENIA, we report the number of category and span errors in Table 6. We can see that the proposed model produced fewer category errors than the baseline one on both corpora. We hypothesise that the synonym generation component attributed to this improvement; the semantic information provided by the component enhanced the ability to distinguish named entities' categories of the All model. Moreover, we find that categories in GENIA are more ambiguous than those in MedMention-21st. For example, it is difficult to distinguish between "Cell line" and "Cell type" or between "Protein" and "DNA". This can explain the lower reduction of category errors in GENIA than in MedMention-21st.

Table 6 also shows that the proposed model could help reduce span errors on MedMention-21st but not on GENIA. We think that the effectiveness of our model depends on the annotation schemes. As mentioned above, the GENIA cor-

Corpus	SupVIB+ SpanReco	All
BC5-Chemical	0.0066	0.0051
BC5-Disease	0.0092	**0.0243**
GENIA	0.3902	**0.4479**
MedMention-21st	0.0612	**0.1482**
NCBI	0.1244	**0.2228**
ShARe/CLEFE	0.0033	**0.1283**

Table 7: BLEU-2 scores of span reconstruction on the development set of each corpus.

pus contains gene/protein-related entities, of which the boundary of an entity is more specially defined than those in MedMention-21st. As a result, detecting correct spans in GENIA is more challenging than in MedMention. For example, the All model detected "HBxAg - specific synthetic polypeptides" as a Protein entity while the correct span should be "HBxAg" only.

Nevertheless, the span errors accounted for a majority of the false positives across all corpora, indicating that an enhanced-boundary approach can help alleviate the situation.

5.3 Generation Quality

To evaluate the generation quality of the VAEs component, we calculated BLEU-2 scores of the reconstructed gold entities on the development set of each corpus. From Table 7, we can see that using synonym generation helped improve the BLEU scores in almost corpora. This indicates that semantic information from the synonym generation component is useful for the NER task and the span reconstruction.

In Table 8, we show examples where having synonym generation, i.e., the All model, could reconstruct original entities much better than SupVIB+SpanReco. In all examples, the reconstructed spans by the All model are more meaningful and fluent than those by SupVIB+SpanReco.

Among the six corpora, reconstructing chemical entities is the most challenging task, especially with short-form chemicals such as PPA (Phenylpropanolamine). When looking for synonyms of these entities in UMLS, we only did exact matching without checking any semantics. Therefore, there are cases in which we found UMLS synonyms for these abbreviations, but their full forms are completely different. For example, some UMLS synonyms of PPA are "primary progressive aphasia", "primary progressive apraxia of speech", and "Mesulam syndrome", which are not relevant to "Phenylpropanolamine"—the correct full form of the chemical PPA. Nevertheless, we observed that the All model usually reconstructed an abbreviation if the original entity is an abbreviation, while the SupVIB+SpanReco did it very randomly. For example, in row 9 in Table 8, given an abbreviation of "PPA", the All model could generate the exact full form given span, while SupVIB+SpanReco generated "antidepressant"–an incorrect full form for "PPA".

5.4 Entity Posteriors

We further explore the effect of the proposed model by inspecting its intermediate values, which are posteriors z generated by the VAE and VIB components.

5.4.1 Span Reconstruction and Synonym Generation

We plot the resulting posteriors z_1 of gold entities on the development set in a 2D space using tSNE (van der Maaten and Hinton, 2008) in two cases: (i) Only VAEs: when we only train span reconstruction and synonym generation, and (ii) our joint model. We observed that across the settings, thanks to the target task of NER, z_1 was satisfactorily restructured, i.e., posteriors of entities in the same category are clustered together. This phenomenon is more visible in the case of GENIA and MedMention-21[4] than in the other corpora since the two corpora have more than one named entity categories, as illustrated in the first row of Figure 2. In Figures 2a and 2c, entities are scattered over the space. Meanwhile, in Figures 2b and 2d, points in the same category are grouped. We can conclude that after simultaneously training VAEs and NER (the All model), we have better clusters than only training VAEs (Only VAEs).

5.4.2 Supervised IB

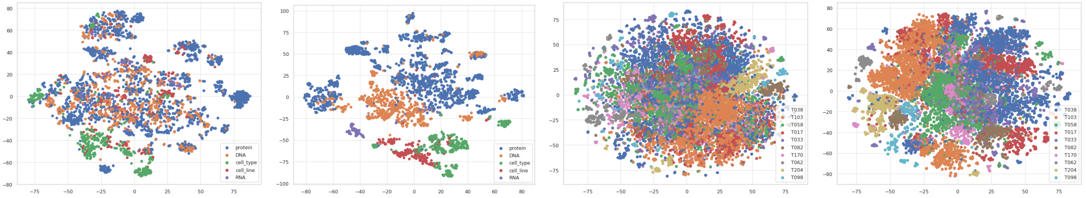
Similarly, we plot the gold entity embeddings and their posteriors z_3 estimated by the supervised IB in the second row of Figure 2. We can see in Figures 2e and 2h that initially, the gold entity embeddings are not clustered very well. In contrast, their posteriors are neatly grouped as illustrated in Figures 2f and 2h. These clusters look even more distinguishable than those by z_1, i.e., Figures 2b and

---

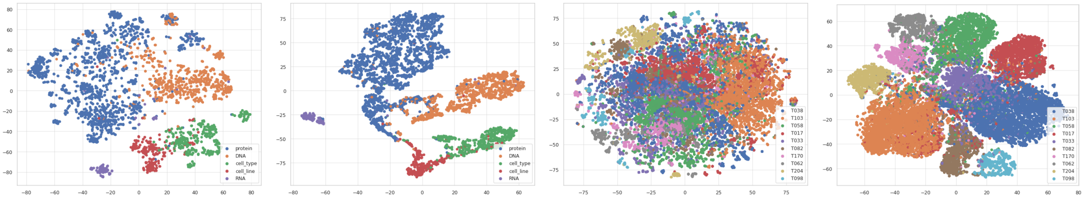
[4] We only show embeddings of the top-10 named entity categories in MedMention-21.

No.	Original span	SupVIB + SpanReco	All
1	nervous fibers	central channel	nervous muscles
2	renal failure	pain failure	renal failure
3	immunomodulatory therapy	Cg therapy	antiviral therapy
4	central hydrophobic core	central its carbon	central temporal core
5	pulmonary veins	liver ganglion	pulmonary vein
6	fibre degeneration	hearing loss	macular degeneration
7	cardiac inter - beat	cardiac - time coil	cardiac - - range
8	volatile compounds	translational compounds	volatile compounds
9	PPA	antidepressant	PPA
10	autoimmune lymphoproliferative syndrome	ankylosing lymphoproliferative syndrome	autoimmune lymphoproliferative syndrome

Table 8: Some examples to show that the synonym information is helpful to span reconstruction.



(a) GENIA: Only VAEs (b) GENIA: The joint model (c) MM21: Only VAEs (d) MM21: The joint model



(e) GENIA: Input to VIB (f) GENIA: Output by VIB (g) MM21: Input to VIB (h) MM21-Output by VIB

Figure 2: Visualisation of entity posteriors on the development set of GENIA and MedMention-21. Points in the same colour indicate entities in the same category. First row: posteriors from the generation models. Second row: the embeddings of entities input to VIB and the corresponding posteriors by VIB.

2d. This phenomenon is understandable because z_3 is approximated based on supervised learning while z_1 is based on unsupervised one.

These visualisations also demonstrate the effectiveness of the supervised IB method. While reducing the input representation size, the method can still reserve relevant features about the target classes. As a result, we can have smaller but more meaningful representations than the original ones. Such neat clusters are potentially helpful for entity linking (Liu et al., 2021).

6 Conclusion

We introduced a joint span-based NER model consisting of three components: VAE-based span reconstruction, VAE-based synonym generation, and VIB-based NER. Each component plays a different role in learning span representation. The VIB-based NER tries to preserve the information about the target NER category in the estimated latent while filtering out irrelevant information from the input. The model is forced to keep contextualised span information when having the span reconstruction component. Meanwhile, the synonym generation component indirectly injects semantic information about a span's synonyms. When testing on five different corpora, we found that the joint model could perform better than the baseline.

The proposed model focuses on learning span representation, which is applicable not only to span-based NER but also to other span-based tasks such as event coreference resolution (Lu and Ng, 2021) and question answering (Li and Choi, 2020). We plan to apply our model to such tasks in the future.

Limitations

We find two main limitations of the proposed model. Firstly, we need synonyms for each gold entity to train the synonym generation component. Unfortunately, this requirement is unsuitable for many NER corpora, such as CoNLL 2012 and ACE2005. One possible way to go around this is to treat the coreferences of an entity (if any) as synonyms. Secondly, similarly to some span-based NER models, our model suffers from a considerable number of all possible spans. We can alleviate this limitation by detecting spans' boundary (Tan et al., 2020; Xu et al., 2021) before classifying spans. We, however, leave those as future work.

Acknowledgements

We would like to thank Dr. Hai-Long Trieu for his help and fruitful comments on the initial implementation of the paper. This paper is based on results obtained from a project JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA. Association for Computing Machinery.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *Proceedings of the International Conference on Learning Representations*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Res.*, 32(Database-Issue):267–270.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018. Variational sequential labelers for semi-supervised learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 215–226, Brussels, Belgium. Association for Computational Linguistics.

Wei Chen, Yeyun Gong, Song Wang, Bolun Yao, Weizhen Qi, Zhongyu Wei, Xiaowu Hu, Bartuer Zhou, Yi Mao, Weizhu Chen, Biao Cheng, and Nan Duan. 2022. DialogVED: A pre-trained latent variable encoder-decoder model for dialog response generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4852–4864, Dublin, Ireland. Association for Computational Linguistics.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2021. Distantly supervised relation extraction with sentence reconstruction and knowledge base priors. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 11–26, Online. Association for Computational Linguistics.

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

Rezarta Islamaj Dogan, Robert Leaman, and Zhiyong Lu. 2014. NCBI Disease Corpus: A Resource for Disease Name Recognition and Normalization. *Journal of biomedical informatics*, 47:1–10.

Thomas Effland and Michael Collins. 2019. Semi-Supervised Named Entity Recognition with CRF-VAEs . In *OpenReview*.

Jinlan Fu, Xuanjing Huang, and Pengfei Liu. 2021. SpanNER: Named entity re-/recognition as span prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7183–7195, Online. Association for Computational Linguistics.

Sam Henry, Kevin Buchan, Michele Filannino, Amber Stubbs, and Özlem Uzuner. 2020. 2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records. *Journal of the American Medical Informatics Association*, 27(1):3–12.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. GENIA corpus - a semantically annotated corpus for bio-textmining. In *Proceedings of* *the Eleventh International Conference on Intelligent Systems for Molecular Biology, June 29 - July 3, 2003, Brisbane, Australia*, pages 180–182.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *The 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Changmao Li and Jinho D. Choi. 2020. Transformers to learn hierarchical contexts in multiparty dialogue for span-based question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5709–5714, Online. Association for Computational Linguistics.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021. A span-based model for joint overlapped and discontinuous named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4814–4828, Online. Association for Computational Linguistics.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Xiang Lisa Li and Jason Eisner. 2019. Specializing word embeddings (for parsing) by information bottleneck. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2744–2754, Hong Kong, China. Association for Computational Linguistics.

Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. 2021. Self-alignment pretraining for biomedical entity representations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4228–4238, Online. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2021. Span-based event coreference resolution. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13489–13497.

Tianyi Luo, Rui Meng, Xin Wang, and Yang Liu. 2022. Interpretable research replication prediction via variational contextual consistency sentence masking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3864–3876, Dublin, Ireland. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2021. Variational information bottleneck for effective low-resource fine-tuning. In *Proceedings of the International Conference on Learning Representations*.

Sunil Mohan and Donghui Li. 2019. MedMentions: A Large Biomedical Corpus Annotated with UMLS Concepts. In *1st Conference on Automated Knowledge Base Construction, AKBC 2019, Amherst, MA, USA, May 20-22, 2019*.

Aitor Ormazabal, Mikel Artetxe, Aitor Soroa, Gorka Labaka, and Eneko Agirre. 2022. Principled paraphrase generation with parallel corpora. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1621–1638, Dublin, Ireland. Association for Computational Linguistics.

Hiroki Ouchi, Jun Suzuki, Sosuke Kobayashi, Sho Yokoi, Tatsuki Kuribayashi, Ryuto Konno, and Kentaro Inui. 2020. Instance-based learning of span representations: A case study through named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6452–6459, Online. Association for Computational Linguistics.

Hiroki Ouchi, Jun Suzuki, Sosuke Kobayashi, Sho Yokoi, Tatsuki Kuribayashi, Masashi Yoshikawa, and Kentaro Inui. 2021. Instance-based neural dependency parsing. *Transactions of the Association for Computational Linguistics*, 9:1493–1507.

Sampo Pyysalo, Filip Ginter, Hans Moen, Tapio Salakoski, and Sophia Ananiadou. 2013. Distributional semantics resources for biomedical text processing. In *Proceedings of the LBM 2013*.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China. PMLR.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849, Brussels, Belgium. Association for Computational Linguistics.

Mohammad Golam Sohrab, Minh Thang Pham, Makoto Miwa, and Hiroya Takamura. 2019. A neural pipeline approach for the PharmaCoNER shared task using contextual exhaustive models. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, pages 47–55, Hong Kong, China. Association for Computational Linguistics.

Hanna Suominen, Sanna Salanterä, Sumithra Velupillai, Wendy W. Chapman, Guergana Savova, Noemie Elhadad, Sameer Pradhan, Brett R. South, Danielle L. Mowery, Gareth J. F. Jones, Johannes Leveling, Liadh Kelly, Lorraine Goeuriot, David Martinez, and Guido Zuccon. 2013. Overview of the ShARe/CLEF eHealth Evaluation Lab 2013. In *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, pages 212–231, Berlin, Heidelberg. Springer Berlin Heidelberg.

Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary Enhanced Neural Span Classification for Nested Named Entity Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9016–9023.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 1999. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377.

H-L. Trieu, T. T. Tran, K. N. A. Duong, A. Nguyen, M. Miwa, and S. Ananiadou. 2020. Deepeventmine: End-to-end neural nested event extraction from biomedical texts. *Bioinformatics*, page btaa540.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Slava Voloshynovskiy, Mouad Kondah, Shideh Rezaeifar, Olga Taran, Taras Holotyak, and Danilo Jimenez Rezende. 2019. Information bottleneck through variational glasses. In *Proceedings of the 4th workshop on Bayesian Deep Learning (NeurIPS 2019)*.

Xiao Wang, Shihan Dou, Limao Xiong, Yicheng Zou, Qi Zhang, Tao Gui, Liang Qiao, Zhanzhan Cheng, and Xuanjing Huang. 2022. MINER: Improving out-of-vocabulary named entity recognition from an information theoretic perspective. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5590–5600, Dublin, Ireland. Association for Computational Linguistics.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. BottleSum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3752–3761, Hong Kong, China. Association for Computational Linguistics.

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Congying Xia, Chenwei Zhang, Tao Yang, Yaliang Li, Nan Du, Xian Wu, Wei Fan, Fenglong Ma, and Philip Yu. 2019. Multi-grained named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1430–1440, Florence, Italy. Association for Computational Linguistics.

Yongxiu Xu, Heyan Huang, Chong Feng, and Yue Hu. 2021. A Supervised Multi-Head Self-Attention Network for Nested Named Entity Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14185–14193.

Jie Yu, Bin Ji, Shasha Li, Jun Ma, Huijun Liu, and Hao Xu. 2022. S-ner: A concise and efficient span-based model for named entity recognition. *Sensors*, 22(8).

Chenhan Yuan and Hoda Eldardiry. 2021. Unsupervised relation extraction: A variational autoencoder approach. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1929–1938, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Cenyuan Zhang, Xiang Zhou, Yixin Wan, Xiaoqing Zheng, Kai-Wei Chang, and Cho-Jui Hsieh. 2022. Improving the adversarial robustness of NLP models by information bottleneck. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3588–3598, Dublin, Ireland. Association for Computational Linguistics.

Changmeng Zheng, Yi Cai, Jingyun Xu, Ho-fung Leung, and Guandong Xu. 2019. A boundary-aware neural model for nested named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 357–366, Hong Kong, China. Association for Computational Linguistics.

Jie Zhou, Yuanbin Wu, Qin Chen, Xuanjing Huang, and Liang He. 2021. Attending via both fine-tuning and compressing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2152–2161, Online. Association for Computational Linguistics.

A Hyper-parameter settings

In Table 9, we show the hyper-parameter values used in our experiments. In both SupVIB+SpanReco and All settings, we trained VAEs in 10 epochs before introducing NER into the models. Our models were finetuned on NVIDIA Tesla V100 GPUs with 16GB of RAM using Optuna (Akiba et al., 2019). Depending on the size of the corpus, we ran different numbers of trials, which were in the range of [5,15].

Hyper-parameter name	Value
β	[1e-6, 1e-4]
γ	[1e-6, 1e-4]
Latent size	[512, 768, 1024]
Input size of the LSTM encoder	200
Output size of the LSTM decoder	256
NER learning rate	[1e-5, 3e-4]
VAE learning rate	[1e-4, 1e-3]
Batch size	16
Maximum length of a span	14
Maximum sentence length	512
Number of epochs	20
Number of epochs to pretrain VAEs	10

Table 9: Hyper-parameter values

An In-depth Analysis of Implicit and Subtle Hate Speech Messages

Nicolas Ocampo, Ekaterina Sviridova, Elena Cabrio and **Serena Villata**
Universite Côte d'Azur, CNRS, Inria, I3S, France
`nicolas-benjamin.ocampo@etu.univ-cotedazur.fr`, `ekaterina.sviridova@inria.fr`
`elena.cabrio@univ-cotedazur.fr`, `villata@i3s.unice.fr`

Abstract

The research carried out so far in detecting abusive content in social media has primarily focused on overt forms of hate speech. While explicit hate speech (HS) is more easily identifiable by recognizing hateful words, messages containing linguistically subtle and implicit forms of HS (as circumlocution, metaphors and sarcasm) constitute a real challenge for automatic systems. While the sneaky and tricky nature of subtle messages might be perceived as less hurtful with respect to the same content expressed clearly, such abuse is at least as harmful as overt abuse. In this paper, we first provide an in-depth and systematic analysis of 7 standard benchmarks for HS detection, relying on a fine-grained and linguistically-grounded definition of implicit and subtle messages. Then, we experiment with state-of-the-art neural network architectures on two supervised tasks, namely implicit HS and subtle HS message classification. We show that while such models perform satisfactory on explicit messages, they fail to detect implicit and subtle content, highlighting the fact that HS detection is not a solved problem and deserves further investigation.

1 Introduction

The rising mass of communication through social media further exacerbates harmful consequences of online hate speech. As a result, social media have faced mounting pressure from civil rights groups demanding to ramp up their enforcement of anti-hate speech policies, so that to monitor and limit this kind of content. In the latest years, numerous methods have been developed to automatically identify this type of utterances expressing hateful or abusive content on social media using Natural Language Processing methods. A variety of datasets have also been built, exemplifying various manifestations of this harmful content (Poletto et al., 2021). However, most of the research carried out so far on this topic has focused on overt forms of hate speech. Explicit hate speech is more easily identifiable by recognizing a clearly hateful word or phrase. Only recently, a few works (Hartvigsen et al., 2022; Wiegand et al., 2022, 2021a; ElSherief et al., 2021; Jurgens et al., 2019; Waseem et al., 2017) have started to focus on implicitness, where circumlocution, metaphor, or stereotypes are used to intentionally convey hatred towards a particular group. In those messages, hatefulness can be captured only by understanding their global meaning, as well as contextual information.

In this paper, we carry out an in-depth analysis of implicit HS in standard benchmarks for HS detection. Additionally, we define the notion of *Subtle HS* that puts forward hateful meanings elusively relying on human perception and through the use of complex syntactic structures. In our study, we collect messages from 7 available datasets for HS detection that cover different topics and are extracted from different social media platforms, and we enrich them with the following three-layer annotation: HS/non HS, Explicit/Implicit and Subtle/Non Subtle. We also provide a fine-grained annotation for implicit HS messages with 18 implicit properties such as irony, exaggeration, metaphor, and rhetorical question, among others. The newly created resource named ISHate (Implicit and Subtle Hate speech) provides a rich and variegate benchmark for pushing forward research on implicit and subtle hateful messages, and constitutes a challenging test-bed to evaluate computational approaches.[1] Additionally, we evaluate SOTA and competitive baseline classifiers to detect both implicit and subtle HS in ISHate, showing that current methods fail to effectively detect implicit and subtle HS messages due to their peculiar nature.

NOTE: This paper contains examples of language which may be offensive to some readers. They do

---

[1] The annotated corpora, and the accompanying annotation guidelines and software can be found at `https://github.com/benjaminocampo/ISHate`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 1997–2013
May 2-6, 2023 ©2023 Association for Computational Linguistics

not represent the views of the authors.

2 Related Work

In the latest years, there has been significant research on abusive language and hate speech detection using Natural Language Processing (NLP) methods (e.g., Xu et al. (2012); Dadvar et al. (2013); Poletto et al. (2021); Bohra et al. (2018); Corazza et al. (2020); Zampieri et al. (2019a); Caselli et al. (2020, 2021)). A few works focus on subtypes of HS, such as Warner and Hirschberg (2012) that tackles the recognition of antisemitism, or Waseem and Hovy (2016); Badjatiya et al. (2017); Gambäck and Sikdar (2017) that investigate predictive features to identify HS in the form of racism and sexism. In this context, several challenges and shared tasks have also been organized over the years, that made datasets and resources for multiple languages available (for a survey, see Poletto et al. (2021)). Research studies carried out so far have mostly focused on overt forms of hate speech, while very few works address the issue of implicit and subtle HS (ElSherief et al., 2021). However, several works show awareness of the problem. For instance, Warner and Hirschberg (2012) and Xu et al. (2012) discuss systems' limitations in identifying HS messages which are ambiguous, have patterns of emotional speech or lack context. Zhang and Luo (2018) and Corazza et al. (2020) highlight the complexity of recognizing hateful messages when the meaning is conveyed through sarcasm, stereotypes, complex syntactic structure, or non-explicit lexical patterns.

Among the few studies that attempted to address the issues of implicit and subtle detection, Caselli et al. (2020) defines a shared task to detect implicit and explicit abusive messages from AbusEval, a reannotated dataset based on OLID/OffensEval (Zampieri et al., 2019a). Benikova et al. (2018) paraphrases German HS tweets obtaining implicit and explicit messages to study classification methods. Dadvar et al. (2013) shows how taking user context improves cyberbullying detection with neither explicit profanities nor apparent neutral emotions. Jurgens et al. (2019) and Waseem et al. (2017) explain why explicitness, implicitness, and subtlety are typologies of abusiveness and encourage researchers to develop proactive technologies in this area. ElSherief et al. (2021) introduces a taxonomy of implicit hate speech and a benchmark corpus with fine-grained labels for each message. Hartvigsen et al. (2022) proposes a large-scale approach to automatically generate benign and implicit HS statements through the language model GPT3. Wiegand et al. (2021a, 2022) proposes resources to tackle implicitly abusive comparisons and abusive remarks, which are two subtypes of implicitness. Wiegand et al. (2021b) explains the key issues in the implicit abuse detection, as well as possible future directions to explore. Even though these studies set the basis to explore and model the issue of implicit HS, there is still large room for improvement, both in terms of creating adequate resources to investigate these aspects and in terms of computational approaches to address them.

3 HS, Implicitness and Subtlety

Hate Speech is defined as a direct attack against people – rather than concepts or institutions – based on protected characteristics (PC): race, ethnicity, national origin, disability, religious affiliation, caste, sexual orientation, sex, gender identity, and severe disease (Meta, 2022). We encompass in the concept refugees, migrants, immigrants, asylum seekers from the most severe attacks, and occupations when they are referenced along with PC, though commentary and criticism of immigration policies are excluded from HS. Following this definition, HS differs from related concepts mainly since it is specifically oriented to groups of people with PC as the main target (Poletto et al., 2021). In the following, we provide clear and operational definitions of explicit, implicit and subtle HS. Reported examples are extracted from the White Supremacy Forum (WSF) (de Gibert et al., 2018) and the CONAN (Chung et al., 2019) datasets, described in Section 4.

3.1 Explicit Hate Speech

Explicit HS is unambiguous in its potential to be abusive/hateful, such as language containing racial or homophobic slurs. Explicit HS uses words whose literal definition (taken from the dictionary) is hateful (ElSherief et al., 2021; Waseem et al., 2017; Caselli et al., 2020), as in Example 1.

1. *Negros are so dumb* . (WSF)

3.2 Implicit Hate Speech

According to the definitions of ElSherief et al. (2021), implicit HS does not immediately denote abuse/hate. Implicitness goes beyond word-related meaning, implying figurative language use such as irony, sarcasm, etc. (ElSherief et al., 2021; Benikova et al., 2018), generally hiding the real meaning, making it more difficult to grasp and undermining the collection of hateful messages (Hartvigsen et al., 2022; Waseem et al., 2017). Therefore, we deal with figurative meaning intended by the author rather than literal meaning of words that may be extracted from dictionaries.

In our extensive study on hate speech datasets, we identified 18 typical properties of implicitness relying on linguistics features (listed in Table 2). Among the most significant ones we have irony (Frenda et al., 2022), sarcasm (Potamias et al., 2020), black humor, metaphor (Choi et al., 2021; Gao et al., 2018), exaggeration (Troiano et al., 2018), rhetorical question, sentiment (Li et al., 2021), inference, lack of context (Dadvar et al., 2013), and lack of extralinguistic knowledge. Most of the implicit HS messages contain combinations of several properties, as in:

2. *Global Jewry did not want Brexit and are doing everything they can to sabotage it.* (CONAN)

Example 2 illustrates an interlacement of irony and exaggeration. Global Jewry represents a wordplay mixing Global Jury and Jews adding humorous content to the perception of Jewish people as a nation, while doing everything is hyperbolic pointing out that Jewish are extremely occupied by Brexit. The whole message is ironical as it emphasizes that Israel has its own interest in other country's affairs.

3. *I also , unfortuantely , spent 7-8th grade in a 95 % monkey house* (WSF)

In Example 3 there is an overlap of irony, exaggeration, and metaphor (underlined). *95 %* looks more like an exaggeration than a fact, while monkey house refers to the group of students and in the metaphor due to their skin color they are compared with monkeys.

4. *Are you sure that Islam is a peaceful religion?* (CONAN)

Example 4 interlaces rhetorical question, inference, and sentiment. The rhetorical question doubts about the peacefulness of Islam, expressing at the same time the author's take on it. As well as that, we may read negative inference between the lines.

3.3 Subtle Hate Speech

Subtle Hate Speech concerns hateful messages that are so delicate or elusive as to be difficult to analyze or describe, and that depend on an indirect method to deliver the meaning (Merriam-Webster, 2022). However, literal meanings (Roland et al., 1968; Bertrand, 1905) are of prime importance in subtle messages by contrast to implicit messages where we go beyond literal meanings (Waseem et al., 2017). Although implicitness and subtlety differ a lot at this point, we still rely on language users' perception (Grice, 1967, 1989; Korta and Perry, 2020) to understand implicit and subtle messages (Benikova et al., 2018). Despite the challenges in characterizing human perception schematically, in our study, the use of elements such as negations with positive clauses, conditionals, connectors, unrelated constructions, word order, and circumlocution can greatly affect the subtlety of a message. Cases of HS that convey the meaning directly and straightforwardly are non-subtle. As subtlety is even less studied than implicitness in the domain of HS (Breitfeller et al., 2019), and only considered as a related concept of implicitness in previous works (Jurgens et al., 2019; ElSherief et al., 2021; Hartvigsen et al., 2022), we analyze their interrelation, particularly whether one of these concepts is an umbrella term for the other.

First, we study whether explicit HS can convey the meaning in a subtle way, to accept or reject the idea that implicitness is a general term for subtlety. As can be seen from the example below, a message can be explicit and subtle at the same time:

5. *As a brit my knowledge of american law is somewhat lacking but even i know that this holder groid has committed treason* (WSF)

Example 5 is explicit as it literally conveys hateful meanings with the expression holder groid, but it is subtle as the intended hateful meaning is disguised by the first and second underlined constructions, pushing out the attention from the explicit hateful expression.

On the other side, we investigate whether subtlety is an umbrella concept for implicitness (whether all implicit messages are subtle by default). The following example shows an implicit non-subtle HS message:

6. *I 'm either in North Florida or Nigeria sometimes i can't tell the difference.* (WSF)

According to the definitions of explicitness and implicitness, this message is not explicit since it does not contain any hateful words, thus, it does not directly convey hateful meaning. However, it is hateful despite being more elaborated, as it includes certain figures of speech that trigger hateful meaning. The underlined expression in Example 6 represents antithesis, while the message itself is ironical. This message also lacks contextual information and extralinguistic knowledge. Although being implicit, this message is not subtle as the meaning is delivered plainly without any delusion or delicacy. Based on these considerations, we consider implicitness and subtlety as two separate concepts. To illustrate the difference between implicit non-subtle messages presented above and implicit subtle cases, consider the following example:

7. *I think it is a bit late to think to look after the saety and the future of white people in Southfrica* . (WSF)

We label it as implicit HS as it does not express hateful meaning explicitly via offensive words and it is ironic. It is also subtle as it starts with an introductory construction that makes the message more sophisticated and adds extra weight to it.

As can be expected, it is also possible to find non-hateful subtle or implicit sentences (such as ironic or sentimental texts). However, our work focuses on exploring implicitness and subtlety in the context of hate speech only, therefore those more general cases are not investigated.

4 The ISHate Dataset

Relying on the fine-grained definitions of HS provided in the previous section (explicit, implicit and subtle HS), we collect and enrich 7 available standard datasets for HS detection. As a result, we create the first benchmark for implicit and subtle HS detection on social media messages extracted from different sources.

4.1 Data Collection

Nearly all available resources of user-generated HS content are retrieved with a keyword-based approach, and mainly relying on a list of words with negative polarity (Poletto et al., 2021). However, with this strategy it is possible to extract mainly explicit HS expressions (as in the AbusEval dataset, Caselli et al. (2020)). Given that our study focuses on implicit and subtle HS, we prefer to explore resources collected from communities of users that are potentially prone to hate speech, or resources manually created using a systematic approach. In the following, we list the considered resources:

White Supremacy Forum Dataset (WSF) (de Gibert et al., 2018), that contains HS messages from Stormfront, scraped from the most influential white supremacist forum on the Web. The database is arranged in sub-forums and conversation threads.

HatEval (Basile et al., 2019), which is among the most well-known benchmark for HS detection. A combined approach is applied to collect hateful and misogynous tweets by monitoring potential victims of hate accounts, downloading the history of identified haters, and filtering Twitter streams with both neutral and derogatory keywords.

Implicit Hate Corpus (IHC) (ElSherief et al., 2021), annotated with explicit HS, implicit HS, and non-HS labels obtained from online hate groups on Twitter. The authors focused on eight ideological clusters of U.S., as Black Separatists, White Nationalist and Neo-Nazi. From this dataset we only extracted messages labeled as implicit HS, as it is one of our target categories.

ToxiGen (Hartvigsen et al., 2022), a dataset with benign and implicit toxic messages against minority groups. ToxiGen is machine-generated through the GPT3 language model and prompt programming. Similarly to IHC, we only extracted messages which were automatically labeled as implicit HS and human-validated as toxic by the authors. We did not consider unfinished generated sentences which make a part of implicit messages.

YouTube Video Comments Dataset (YouTube) (Hammer, 2017), that consists of YouTube comments posted under videos related to religion and politics. Differently from the other resources, the messages are annotated as "violent" or "clean".

CONAN (Chung et al., 2019), a dataset of HS messages and counter-narratives (CN) pairs for CN generation. Two native English speakers were asked to write 50 prototypical short texts, which NGO could later use to write their hate texts and counter-narratives. We believe that messages for which a CN can be provided might be richer in implicit content since a slur-based explicit HS message might produce very poor argumentative CN.

Multi-Target CONAN (MCONAN) (Fanton et al., 2021), a dataset of English HS/CN pairs comprising several hate targets. It is collected using a Human-in-the-Loop approach. A generative language model is refined iteratively by using data from the previous loops to generate new samples that NGOs experts review.

Before starting the annotation process with the fine-grained annotations (Explicit, Implicit and Subtle HS), we had to make sure that the definition of HS originally used to annotate such resources is consistent with ours. In the first annotation round, we checked the messages originally annotated as HS, and discarded the few ones that did not correspond to the definition of HS reported in Section 3. For the YouTube dataset, we also added the HS labels. While all the messages annotated as HS are directed to PC, it should be noted that the topics distribution and the writing quality might be different, given the heterogeneity of the selected resources. HS messages mostly target Islamism, Judaism, misogyny, multi-culturalism, racism, immigration, and refugees. Regarding time creation, WSF is made from threads posted between 2002 and 2017, ToxiGen's LM was trained with messages from 2016 to 2019, Hateval consists of messages of 2018, the YouTube comments were collected in 2017, while the IHC contains tweets from U.S. ideological clusters from 2015 to 2017.

4.2 Annotation Procedure

Following the annotation scheme described in Section 3, four graduate-level annotators with linguistics and computational linguistics competences carried out a pilot annotation study on a sample of 100 messages extracted from each of the above mentioned resources to converge to non-ambiguous annotation strategies. We calculate the Inter Annotator Agreement (IAA) on this sample, resulting in Cohen's κ=0.793 (Cohen, 1960) for the implicit layer (binary annotation Explicit/Implicit) and 0.730 for the subtlety layer (binary annotation Subtle/Non-Subtle). We also compute the IAA considering both layers simultaneously, that is, considering one layer of 4 classes (Implicit, Explicit, Subtle, Non-Subtle), obtaining a Cohen's κ of 0.734. In the reconciliation phase, we notice that most of the disagreements are due to the interlacement of subtlety and implicitness. For that reason, we also calculate an ordered weighted disagreement using Krippendorff's α to penalize less when the annotators agree at least on one of the layers (Artstein and Poesio, 2008). The Krippendorff's α is 0.757. Despite the complexity of the annotation task, obtained results are considered as strong agreement in a two-annotators setting. The rest of the annotations has then been carried out by two of the annotators mentioned above, which were provided with the final version of the annotation guidelines (containing the definitions of the target classes, i.e., subtlety and implicitness, and a discussion about borderline cases), together with a set of labeled examples.

Finally, the implicit properties annotations are added on top of the messages labeled as implicit as an additional annotation layer to highlight 18 linguistic features that implicitly convey hateful meaning. For this layer, annotations are carried out by one expert linguist.

4.3 Data Statistics

Table 1 shows statistics of the final dataset, reporting on the number of annotated HS messages for each resource and for the three annotation layers.

The ISHate collection consists of a total of 29116 messages, where 11247 are HS (further annotated with the Explicit/Implicit and Subtle/Non-subtle labels). For computational purposes, we provide a dataset split in three subsets, i.e., *train* (70%), *validation* (15%), and *test* (15%) sets. Each of the partition respects the distribution of all the annotation layers using stratified splitting. As can be seen, classes are unbalanced, each resource providing only a reduced number of implicit and subtle messages - as expected. Note that CONAN and MCONAN do not contain Non-HS messages, because their main objective is CN generation. As for IHC and ToxiGen, we only look through previously annotated implicit HS messages disregarding non hateful ones. Note also that ToxiGen claimed to contain only implicit adversarial messages, but according to our definitions and annotation guidelines many messages are considered as explicit and non-subtle by our annotators.

Table 2 shows the full distribution of the implicit properties relative to the implicit messages in ISHate. As it can be seen, Inference (58%), Context (48%), Sentiment (45%), Exaggeration (28%) and Irony (22%), are the most frequent properties of implicit HS messages, whereas Euphemism (4%), Circumlocution (3%), Metonymy (0.4%) and Synecdoche (0.08%) are the least recurrent. Note that one implicit message can be labeled with more than one property.

	Train		Dev		Test		CONAN	HatEval	IHC	MCONAN	ToxiGen	WSF	Youtube
Label	#	%	#	%	#	%							
Non-HS	12508	.614	2680	.614	2681	.614	0	7421	0	0	0	9342	1106
Explicit HS	7007	.344	1501	.344	1501	.344	324	3107	317	3344	183	987	1747
Implicit HS	866	.042	186	.043	186	.043	81	110	300	295	170	173	109
Non-HS	12508	.614	2680	.614	2681	.614	0	7421	0	0	0	9342	1106
Non-Subtle	7691	.377	1648	.377	1648	.377	393	3191	614	3595	348	1018	1828
Subtle	182	.009	39	.009	39	.009	12	26	3	44	5	142	28

Table 1: Statistics on the annotated dataset (resources and label distributions for the two tasks)

	Implicit HS	
Implicit Properties	#	%
Inference	729	58.885
Context	602	48.627
Sentiment	569	45.961
Exaggeration	359	28.998
Irony	275	22.213
Extralinguistic knowledge	193	15.590
Black humor	144	11.632
Rhetorical question	134	10.824
Visual signs	122	9.855
Humiliation	115	9.289
Antithesis	97	7.835
Metaphor	93	7.512
Sarcasm	85	6.866
Fallacy	74	5.977
Euphemism	56	4.523
Circumlocution	41	3.312
Metonymy	6	0.485
Synecdoche	1	0.081

Table 2: Statistics on implicit properties distribution.

4.4 Data Augmentation

To overcome the problem of the unbalanced dataset, we propose oversampling and data augmentation (DA) methods for the minority classes using adversarial methods and generative models following (Mayer et al., 2020; Wei and Zou, 2019), and the GPT2 language model (Radford et al., 2019).

Replace Named Entities (RNE). It replaces a named entity (PER, LOC, ORG, and MISC) in the input sentence. A candidate NE in a sentence is replaced by another one according to a previously collected list of NEs (Mayer et al., 2020). Then, the most similar NE is selected by using pre-trained FastText embeddings (Bojanowski et al., 2016). In our use case, we notice that the number of NEs PER, LOC, and ORG are very few compared to MISC. This might be due to the fact that HS messages in our collection mostly target groups and not individuals. However, expressions like *muslims*, *jews*, or *blacks* are present in the MISC category and replaced, as in Example 8. As it can be noticed, the expression preserves its meaning, that is, the use of the rhetorical question to convey that Muslims are not considered as a part of society.

8. **Original**: *Have Muslims ever made a contribution to our society?* (CONAN)
   Augmented: *Have Moslem Arabs ever made a contribution to our society?*

Replace Scalar Adverbs (RSA). It replaces emphasising adverbs like *considerably* or *largely*, with another scalar adverb that might increase or decrease the emphasis of an adjective/verb.

9. **Original**: *I do like the outfit they dressed the simian in ; he looks so regal* (WSF)
   Augmented: *I do like the outfit they dressed the simian in ; he looks incredibly regal*

Example 9 is another implicit racist HS message that uses irony by combining *simian*, *dressed*, and *regal*. RSA replaces the scalar adverb, slightly increasing the emphasis of *regal* in the sentence.

Add Adverbs to Verbs (AAV). It adds modifiers to verbs to accentuate them. In this case, we use speculative adverbs like *certainly*, *likely*, *clearly*.

10. **Original**: *I wish we lived in a world in which only men could be doctors, lawyers, firemen, and teachers.* (MCONAN)
    Augmented: *I wish we lived in a world in which only men could plausibly be doctors, lawyers, firemen, and teachers.*

In Example 10, the meaning is made elusive by the use of the expression *I wish we lived*, the addition of *plausibly* alters the possibility of the verb “to be”, but keeps the misogyny meaning.

Replace Adjectives (RA). It replaces adjectives with their synonyms, as in:

11. **Original**: *If you look at an Alaskan malamute and a timber wolf that are two different species , they look a lot more alike than say a Congolese and a Swede .* (WSF)
    Augmented: *If you look at an Alaskan malamute and a timber wolf that are two unlike species , they look a lot more alike than say a Congolese and a Swede .*

Replace In-Domain Expressions (RI). It replaces a list of manually-crafted expressions often used in HS messages (not captured by the RNE) with other semantically similar expressions, as in Example 12. Here the word "migrants" is replaced by "foreigners" without adding hateful connotations. It would have been different to replace it with *aliens*, which is a derogatory term to refer to migrants.

12. **Original**: *migrants are entering the u.s. just to obtain welfare and other benefits .* (MCONAN)
Augmented: *foreigners are entering the u.s. just to obtain welfare and other benefits .*

Easy Data Augmentation (EDA). Given an input sentence, EDA randomly: *i)* replaces a non-stop word expression with a synonym using Wordnet; *ii)* inserts a synonym of a non-stop word in a random position; *iii)* chooses two words of the sentence and swaps their positions; *iv)* removes each word in the sentence with a certain probability. Only one of the four operations at a time is applied to a sentence.

13. **Original**: *A lot of white women are trying to create danger by drawing these other people into our countries .* (WSF)
Augmented: *A lot of our women are trying to create danger by drawing these other people into white countries .*

A parameter α expressing the percent of the words to be changed in a sentence is specified (in our setting, $\alpha = 0.1$ as in Wei and Zou (2019)).

Back Translation. It translates an input message into a different language to translate it back into the original language. We use the translation from English to Russian following (ElSherief et al., 2021).

14. **Original:** *As a nation , we are in desperate need for an immigration time out and a zero tolerance policy for those here illegally .* (WSF)
Augmented: *As a nation, we desperately need an immigration time-out and a zero-tolerance policy for those here illegally.*

Generative Models (GM). We fine-tune autoregressive generative language models with instances from our minority classes, i.e., explicit subtle, implicit non-subtle, and implicit subtle messages. To do so, we prefix this label on the text as a prompt. Then, language models are asked to generate messages starting with one of our fine-tuned prompts, as in Example 15. We use GPT2 (Radford et al., 2019) as a language model, fine-tuned for 4 epochs using learning rate of 3e-5, and batch size of 32. Additionally, we implement a human-in-the-loop approach revising the generated examples and re-annotating them in case the original label is no-more appropriate for the message.

15. **Input:** *Explicit Subtle HS:*
Augmented: *Explicit Subtle HS: In the end, it comes down to what women want from a man... If they want to play with whores, they can stay at home and have babies...*

Except for GM and BT, the same strategy is applied to augmentation methods to produce new messages. Preprocessing (e.g., Parts-of-Speech tagging and Named Entities Recognition) is carried out using Flair (Akbik et al., 2019) and NLTK (Bird and Loper, 2004) models, and allows to recognize possible candidate phrases to perform a replacement/addition. Then, a candidate phrase is perturbed by another one according to a list of adverbs, NEs, or adjectives based on domain data. We rely on FastText and WordNet Synsets to maintain the semantics of the augmented sentences with respect to the original one. The number of candidates to perform a replacement/addition and the number of replacement/additions per candidate are provided as parameters to these methods.

5 Evaluation

To show that implicit and subtle HS detection is still a very challenging task, we evaluate a set of state-of-the-art models for HS detection on the ISHate dataset. We propose two 3-label classification tasks:

- Task A (Non-HS/Explicit HS/Implicit HS)
- Task B (Non-HS/Non-Subtle HS/Subtle HS)

To this goal, we consider the following models:

Universal Sentence Encoder (USE) + SVM (Indurthi et al., 2019). First-ranked model on the HatEval benchmark (Basile et al., 2019). The USE (Cer et al., 2018) is a sentence embedding that encodes text into high dimensional vectors of 512 dimensions, trained on large data sources to provide an encoding method that works for various NLP tasks. An SVM classifier with RBF kernel and default parameters is then used for classification.

DeBERTa V3 (hate_speech18). SOTA model on the WSF dataset (de Gibert et al., 2018). For classification, a default HuggingFace implementation of a one-layer Feed Forward network is used on top of DeBERTa (He et al., 2021a,b), a transformer-based model. The model is later fine-tuned for 4 epochs (learning rate of 2e-5, batch size of 32).

BERT (Devlin et al., 2018). We use this language model to encode text sequences and classify them by adding a Feed-forward neural network on top.

HateBERT. A re-trained BERT model using over 1 million posts from banned communities on Reddit (Caselli et al., 2021) and then fine-tuned on our dataset. HateBERT obtained very promising results in the benchmarks HatEval, OffensEval (Zampieri et al., 2019b), and AbusEval (Caselli et al., 2020).

As for preprocessing, we replace long non-space character chains for only one occurrence, and delete digits, special symbols, and URLs.

5.1 Results

Table 3 reports on the results of the different models on the two tasks. On both tasks, all models show satisfactory performances when detecting overt forms of HS (Explicit HS and Non-Subtle HS classes), with DeBERTa outperforming the other models. The results obtained by all models for the Implicit HS and Subtle HS classes are much lower, and comparable to those obtained by ElSherief et al. (2021) (F1-score=.586) on the implicit class.

As a follow-up experiment, we apply the oversampling techniques (Section 4) on the minority classes of tasks A and B until balancing them with respect to the Explicit HS and Non-Subtle HS categories. The oversampling is performed on the training set only. The test set is the one of the original dataset, and is therefore unbalanced in order to evaluate the system on real class distribution and to avoid information leakage from train to test through augmentation methods. Tables 4a and 4b show the number of additional generated implicit/subtle messages and the resultant training set distribution per augmentation method, respectively.

Among all tested models, only HateBERT significantly improves its performance for detecting implicit messages combining all augmented data (ALL) (see Table 3). We also highlight that back translation (BT) better contributes to the performance on the implicit hate class for BERT, DeBERTa, and USE+SVM [2]. Performances surprisingly increase for the subtle class with USE+SVM+BT showing that back-translated messages provide diversity by rephrasing subtle examples without altering their meaning. Data generated with simpler augmentation methods as BERT+RNE and DeBERTa+RI also show slight improvements for subtlety. However, performances decrease on the implicit class when applying data augmentation strategies GM and GM+Revised, and only slightly improve on the subtle class.

5.2 Error Analysis

To gain insights into the models' behaviours, we manually analyse the classification errors of the best performing approaches, i.e., HateBERT+ALL and USE+SVM+BT for both tasks A and B. For the Non-HS/Explicit/Implicit classification, it is harder for HateBERT+ALL to differentiate implicit messages rather than explicit ones. Figure 1 in Appendix A shows the resultant embedding through t-SNE (van der Maaten and Hinton, 2008) approximation of all implicit messages of the test set, and two samples of the other two classes, showing how HateBERT+ALL is capable of separating the space in three different blobs for classification (Figure 1a), embedding well the human-annotated Explicit HS and Non-HS classes (Figure 1b), but badly encoding implicit annotated messages (instances spread over the explicit and non-hateful blobs). We also considered the F1-scores of the (Non-HS vs Implicit HS) and (Non-HS vs Explicit HS) classes, as if we had binary labels, obtaining an (F1-score=0.670) and (F1-score=0.850) on the target categories respectively, showing that implicit cases are harder to detect than explicit ones.

Among misclassified examples, we can find messages where the PC are not explicitly named (Example 16). Contextual knowledge is needed there to grasp that the target of the message is black people.

16. *I can picture him as that baby monkey on the ' save the rainforest comercial or whatever*

Additionally, from the misclassified messages we extract the most frequent implicit properties not captured by the classifier[3]. They concern Inference (53%), Context (41%), Sentiment (40%), Exaggeration (24%), Extralinguistic knowledge (24%).

For Subtle/Non Subtle message classification, we also plotted the USE embedding for the best

---

[2] The table reporting the obtained results by all models on different types of augmented data is in the Appendix.

[3] The full table can be found in the Appendix.

	Task A									Task B								
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Model	Non-HS			Explicit HS			Implicit HS			Non-HS			Non-Subtle HS			Subtle HS		
USE+SVM	.888	.866	.877	.766	.803	.784	.399	.382	.390	.891	.868	.879	.783	.832	.807	.667	.103	**.178**
BERT	.903	.893	.898	.81	.833	.821	.394	.371	.382	.902	.891	.897	.819	.846	.832	.250	.103	.145
HateBERT	.904	.89	.897	.811	.849	.829	.447	.382	.412	.903	.890	.897	.814	.850	.831	.143	.026	.043
DeBERTa	.927	.899	**.913**	.825	.880	**.851**	.467	.419	**.442**	.920	.893	**.906**	.823	.877	**.849**	.375	.077	.128
HateBERT+ALL	.903	.896	.899	.827	.827	.827	.502	.559	**.529**	.903	.881	.892	.816	.844	.830	.391	.462	.424
BERT+BT	.909	.887	.898	.824	.826	.825	.459	.608	.523	.898	.900	.899	.839	.832	.835	.304	.359	.329
DeBERTa+BT	.919	.885	.902	.830	.857	.844	.428	.543	.479	.920	.897	**.908**	.835	.876	**.855**	.385	.256	.308
USE+SVM+BT	.897	.856	.876	.782	.787	.785	.403	.645	.496	.892	.868	.880	.789	.831	.809	.739	.436	**.548**
BERT+RNE	.897	.897	.897	.807	.829	.818	.455	.349	.395	.899	.895	.897	.826	.839	.833	.400	.256	.312
DeBERTa+RI	.922	.894	**.908**	.821	.878	**.849**	.460	.398	.427	.910	.894	.902	.828	.860	.843	.364	.205	.262
HateBERT+GM	.901	.898	.899	.824	.827	.825	.414	.425	.419	.899	.898	.899	.831	.834	.832	.250	.231	.240
HateBERT+GM+R.	.905	.891	.898	.816	.835	.826	.408	.419	.414	.894	.898	.896	.826	.826	.826	.192	.128	.154

Table 3: Obtained results on tasks A and B.

Aug. method Label	RSA	AAV	RNE	RI	RA	EDA	BT	GM	GM+Revised	ALL
Implicit HS	6848	7032	828	817	467	6935	748	200	82	23957
Subtle HS	3192	3136	480	210	172	2912	179	200	204	10685

(a) Number of additional implicit/subtle messages generated by each augmentation method.

Aug. method Label	ORIG	RSA	AAV	RNE	RI	RA	EDA	BT	GM	GM+Revised	ALL
Non-HS	.614	.459	.456	.59	.590	.600	.458	.592	.608	.611	.282
Explicit HS	.344	.257	.256	.33	.331	.336	.257	.332	.340	.342	.158
Implicit HS	.042	.283	.288	.08	.079	.064	.286	.076	.052	.046	.560
Non-HS	.614	.531	.532	.600	.607	.609	.537	.608	.608	.608	.403
Non-Subtle HS	.377	.326	.327	.369	.374	.374	.330	.374	.374	.374	.248
Subtle HS	.009	.143	.141	.032	.019	.017	.133	.018	.019	.019	.350

(b) Train set distribution (%) per augmentation method (ORIG corresponds to the original train distribution).

Table 4: Statistics on the train set with data augmentation.

model on this task (Figure 2 in Appendix A). However, it can be seen that USE+SVM+BT could not differentiate correctly on the subtle notion despite of the results reported in Table 6b. Example 17 is not predicted as subtle. It shows how the word order may influence our understanding. At a first glance, the part how stupid the Jews seems to have a different meaning from what the phrase actually conveys if we read it entirely. We may also notice a circumlocution in the second part of the message.

17. *I am insulted by how stupid the jews think we are until i see what they see by reading the posts amongst our so called , ' ' awakened brethren .*

6 Conclusions

In this paper, we have presented ISHate, the first benchmark dataset annotated with both implicit and subtle HS labels, which represents a challenging test-bed to evaluate computational approaches. We also provide a fine-grained annotation for implicit HS messages with 18 implicit properties which represent the relevant features that HS classifiers should possess to improve implicit HS detection. It has been created enriching 7 existing datasets for HS detection over different topics and from different social media. We have shown that current SOTA models fail to properly detect implicit and subtle HS messages as peculiar features connected to Sentiment, Inference, Context and Irony, as well as complex syntactic structure, cannot be properly understood. We also investigated data augmentation strategies to increase the number of instances for the minority classes. We show that - while they cannot be the ultimate solution to the lack of implicit and subtle examples - they still play a role in improving the systems' performances, in line with ElSherief et al. (2021). As for future work, we plan to propose alternative large-scale methods to collect implicit and subtle messages by targeting "hateful" users, manual creation (Wiegand et al., 2021a, 2022) or refining human-in-the-loop generative methods as in (Hartvigsen et al., 2022). Also, we will investigate features modeling implicit properties (Wallace et al., 2014; Troiano et al., 2018; Frenda and Patti, 2019) and new model architectures for HS detection (Nejadgholi et al., 2022).

Limitations

The main limitation of this paper lies in the intrinsic difficulty to provide a clear definition of the notions of Implicit HS and Subtle HS (given the limited number of definitions available in the literature for these notions), and, as a consequence, to build annotated resources. Enhancing the ISHate dataset with new instances requires future annotators to be experts in computational linguistics trained on our annotation guidelines through pilot annotations to keep the same level of agreement. This restricts crowdsourcing-like options, making the resource building process more expensive. Moreover, the complexity of the messages and of the considered categories makes the process time-consuming (i.e., a trained annotator requires 30sec. for explicit messages and 1.30min. for implicit/subtle messages on average). Even opting for generative and synthetic data augmentation approaches, they still require human-in-the-loop intervention and high computational resources to generate Implicit/Subtle HS messages on a big scale.

Ethics Statement

This paper contains examples of HS from existing linguistic resources for HS detection and which do not reflect the authors' opinions.

While our purpose is to prevent and curate social media resources from HS, the release of this dataset might still pose a potential misuse case. However, we still consider that effective classifiers for this task are necessary to tackle implicit and subtle online hate on scale and prevent the spreading of this harmful content online. Our work aims at making a step towards that objective and encourages the scientific community to investigate these aspects.

Acknowledgements

This work has been supported by the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR- 19-P3IA-0002.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. *CoRR*, abs/1706.00188.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Darina Benikova, Michael Wojatzki, and Torsten Zesch. 2018. What Does This Imply? Examining the Impact of Implicitness on the Perception of Hate Speech. In *Language Technologies for the Challenges of the Digital Age*, pages 171–179, Cham. Springer International Publishing.

Russell Bertrand. 1905. On denoting. *Mind*, 56(14):479–493.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of Hindi-English code-mixed social media text for hate speech detection. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 36–41, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

Luke Breitfeller, Emily Ahn, David Jurgens, and Yulia Tsvetkov. 2019. Finding Microaggressions in the Wild: A Case for Locating Elusive Phenomena in Social Media Posts. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1664–1674, Hong Kong, China. Association for Computational Linguistics.

Miguel Casas Gómez. 2009. Towards a new approach to the linguistic definition of euphemism. *Language Sciences*, 31(6):725–739.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, Inga Kartoziya, and Michael Granitzer. 2020. I Feel Offended, Don't Be Abusive! Implicit/Explicit Messages in Offensive and Abusive Language. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6193–6202, Marseille, France. European Language Resources Association.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor Detection via Contextualized Late Interaction using Metaphorical Identification Theories. Number: arXiv:2104.13615 arXiv:2104.13615 [cs].

Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. 2019. CONAN - COunter NArratives through nichesourcing: a multilingual dataset of responses to fight online hate speech. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2829, Florence, Italy. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. 2020. A multilingual evaluation for online hate speech detection. *ACM Trans. Internet Technol.*, 20(2).

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Proceedings of the 35th European Conference on Advances in Information Retrieval*, ECIR'13, page 693–696, Berlin, Heidelberg. Springer-Verlag.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate Speech Dataset from a White Supremacy Forum. *arXiv:1809.04444 [cs]*. ArXiv: 1809.04444.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Collins Dictionary. 2022. Collins dictionary.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Margherita Fanton, Helena Bonaldi, Serra Sinem Tekiroğlu, and Marco Guerini. 2021. Human-in-the-Loop for Data Collection: a Multi-Target Counter Narrative Dataset to Fight Online Hate Speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Jane Frank. 1990. You call that a rhetorical question?: Forms and functions of rhetorical questions in conversation. *Journal of Pragmatics*, 14(5):723–738.

Simona Frenda, Alessandra Teresa Cignarella, Valerio Basile, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2022. The unbearable hurtfulness of sarcasm. *Expert Syst. Appl.*, 193(C).

Simona Frenda and Viviana Patti. 2019. Computational Models for Irony Detection in Three Spanish Variants. page 13.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada. Association for Computational Linguistics.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural Metaphor Detection in Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 607–613, Brussels, Belgium. Association for Computational Linguistics.

Herbert Paul Grice. 1967. Logic and conversation. In Paul Grice, editor, *Studies in the Way of Words*, pages 41–58. Harvard University Press.

Herbert Paul Grice. 1989. *Studies in the Way of Words*. Cambridge: Harvard University Press.

Hugo Lewi Hammer. 2017. Automatic Detection of Hateful Comments in Online Discussion. In *Industrial Networks and Intelligent Systems*, pages 164–173, Cham. Springer International Publishing.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. 2019. FERMI at SemEval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 70–74, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

David Jurgens, Libby Hemphill, and Eshwar Chandrasekharan. 2019. A just and comprehensive strategy for using NLP to address online abuse. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3658–3666, Florence, Italy. Association for Computational Linguistics.

Kepa Korta and John Perry. 2020. Pragmatics. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, Spring 2020 edition. Metaphysics Research Lab, Stanford University.

Zhengyan Li, Yicheng Zou, Chong Zhang, Qi Zhang, and Zhongyu Wei. 2021. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 246–256, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tobias Mayer, Santiago Marro, Elena Cabrio, and Serena Villata. 2020. Generating adversarial examples for topic-dependent argument classification. In *Computational Models of Argument - Proceedings of COMMA 2020, Perugia, Italy, September 4-11, 2020*, volume 326 of *Frontiers in Artificial Intelligence and Applications*, pages 33–44. IOS Press.

Merriam-Webster. 2022. Dictionary.

Meta. 2022. Facebook: Hate speech policies.

Isar Nejadgholi, Kathleen Fraser, and Svetlana Kiritchenko. 2022. Improving generalizability in implicitly abusive language detection with concept activation vectors. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5517–5529, Dublin, Ireland. Association for Computational Linguistics.

Fabio Poletto, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. 2021. Resources and benchmark corpora for hate speech detection: a systematic review. *Language Resources and Evaluation*, 55(2):477–523.

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Barthes Roland, Lavers Annette, and Smith Colin. 1968. *Elements of semiology / Roland Barthes ; translated from the French by Annette Lavers and Colin Smith*, 1st american ed. edition. Hill and Wang New York.

Enrica Troiano, Carlo Strapparava, Gözde Özbal, and Serra Sinem Tekiroğlu. 2018. A computational exploration of exaggeration. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3296–3304, Brussels, Belgium. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Byron C. Wallace, Do Kook Choe, Laura Kertz, and Eugene Charniak. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–516, Baltimore, Maryland. Association for Computational Linguistics.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26, Montréal, Canada. Association for Computational Linguistics.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. Technical Report arXiv:1901.11196, arXiv. ArXiv:1901.11196 [cs] type: article.

Michael Wiegand, Elisabeth Eder, and Josef Ruppenhofer. 2022. Identifying Implicitly Abusive Remarks about Identity Groups using a Linguistically Informed Approach. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, pages 5600–5612, Seattle, United States. Association for Computational Linguistics.

Michael Wiegand, Maja Geulig, and Josef Ruppenhofer. 2021a. Implicitly abusive comparisons – a new dataset and linguistic analysis. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 358–368, Online. Association for Computational Linguistics.

Michael Wiegand, Josef Ruppenhofer, and Elisabeth Eder. 2021b. Implicitly abusive language – what does it actually look like and why are we not getting there? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 576–587, Online. Association for Computational Linguistics.

Ulrike Willinger, Andreas Hergovich, Michaela Schmoeger, Matthias Deckert, Susanne Stoettner, Iris Bunda, Andrea Witting, Melanie Seidler, Reinhilde Moser, Stefanie Kacena, David Jaeckle, Benjamin Loader, Christian Mueller, and Eduard Auff. 2017. Cognitive and emotional demands of black humour processing: the role of intelligence, aggressiveness and mood. *Cogn. Process.*, 18(2):159–167.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 656–666, Montréal, Canada. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.

Ziqi Zhang and Lei Luo. 2018. Hate speech detection: A solved problem? the challenging case of long tail on twitter. *CoRR*, abs/1803.03662.

A Performance Details in Data Augmentation

Inspired by the ranked one augmentation strategy in ElSherief et al. (2021), i.e., a back-translation approach, we also test SOTA models on our dataset, ISHate, with the augmentation techniques described in Section 4.4. Each model is trained with the originally collected data described in Section 4.3 and additional data obtained from one augmentation strategy. At the end, we also evaluate each model using only non-augmented test data. Tables 6a and 6b show the experiments' results on tasks A and B.

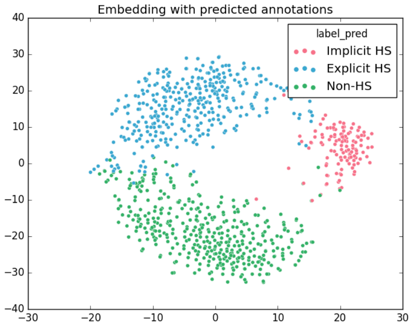
We further analyse the errors committed by the best performing model on task A. We took from Table 6a HateBERT+ALL and the third annotation layer described in Sections 3 and 4 to identify which are the most frequent implicit properties on task A miss-classified messages. Table 5 shows how Inference, Context, Sentiment, Exaggeration, and Extralinguistic knowledge are the most recurrent not captured devices.

	Implicit HS	
Implicit Property	#	%
Inference	44	53.659
Context	34	41.463
Sentiment	33	40.244
Exaggeration	23	28.049
Extralinguistic knowledge	20	24.390
Irony	17	20.732
Black humor	12	14.634
Visual signs	11	13.415
Metaphor	9	10.976
Rhetorical question	8	9.756
Antithesis	6	7.317
Humiliation	5	6.098
Sarcasm	5	6.098
Circumlocution	4	4.878
Fallacy	4	4.878
Euphemism	3	3.659

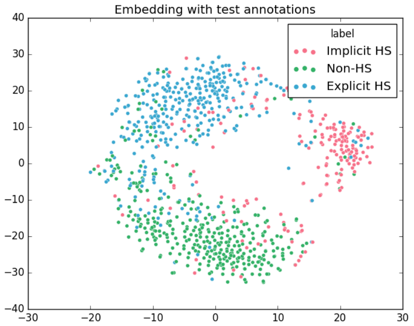
Table 5: Implicit properties of the messages that are not captured by HateBERT+ALL

.

We also analysed the embeddings of our best-performing models in tasks A and B (HateBERT+ALL and USE+SVM+BT, respectively) through t-SNE (van der Maaten and Hinton, 2008). Figures 1 and 2 show the text embeddings for sentences of the test set, labeled by both classifiers and annotators, for the implicit and subtle tasks.

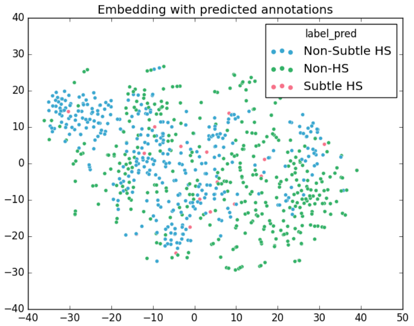


(a) Embedding using predicted annotations.

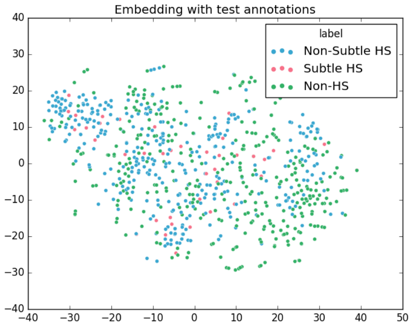


(b) Embedding using manual annotations.

Figure 1: Embedding of HateBERT + ALL in the test set of task A



(a) Embedding using predicted annotations.



(b) Embedding using manual annotations.

Figure 2: Embedding of USE+SVM+BT in the test set of task B

Task A	Non-HS			Explicit HS			Implicit HS		
USE + SVM	P	R	F1	P	R	F1	P	R	F1
RSA	.887	.875	.881	.768	.830	.798	.515	.285	.367
AAV	.888	.875	.881	.764	.825	.793	.491	.280	.356
RNE	.887	.867	.877	.770	.802	.786	.386	.382	.384
RI	.888	.865	.876	.769	.803	.786	.371	.371	.371
RA	.888	.867	.877	.768	.803	.785	.398	.387	.392
EDA	.892	.862	.877	.780	.797	.789	.339	.441	.383
BT	.897	.856	.876	.782	.787	.785	.403	.645	**.496**
GM	.887	.862	.874	.771	.794	.782	.352	.409	.378
GM+Revised	.887	.863	.875	.769	.803	.786	.385	.403	.394
ALL	.889	.879	**.884**	.796	.809	**.803**	.421	.430	.426
BERT	P	R	F1	P	R	F1	P	R	F1
RSA	.894	.899	.896	.805	.835	.819	.487	.301	.372
AAV	.896	.896	.896	.820	.817	.819	.387	.398	.393
RNE	.897	.897	.897	.807	.829	.818	.455	.349	.395
RI	.909	.897	**.903**	.812	.850	**.831**	.458	.376	.413
RA	.894	.905	.899	.822	.825	.823	.473	.382	.423
EDA	.900	.894	.897	.807	.836	.821	.416	.333	.370
BT	.909	.887	.898	.824	.826	.825	.459	.608	**.523**
GM	.898	.901	.900	.824	.821	.823	.409	.398	.403
GM+Revised	.905	.892	.899	.811	.839	.825	.451	.419	.435
ALL	.902	.894	.898	.816	.817	.816	.488	.543	.514
DeBERTaV3	P	R	F1	P	R	F1	P	R	F1
RSA	.912	.893	.902	.803	.877	.838	.441	.242	.312
AAV	.916	.904	.910	.832	.858	.845	.431	.403	.417
RNE	.922	.883	.902	.807	.880	.842	.430	.382	.405
RI	.922	.894	.908	.821	.878	.849	.460	.398	.427
RA	.909	.914	**.911**	.841	.859	**.850**	.482	.360	.412
EDA	.907	.899	.903	.813	.859	.835	.460	.312	.372
BT	.919	.885	.902	.830	.857	.844	.428	.543	**.479**
GM	.913	.899	.906	.839	.843	.841	.399	.468	.431
GM+Revised	.918	.893	.905	.819	.873	.845	.425	.366	.393
ALL	.924	.887	.905	.814	.867	.840	.456	.478	.467
HateBERT	P	R	F1	P	R	F1	P	R	F1
RSA	.895	.895	.895	.814	.830	.822	.452	.382	.414
AAV	.899	.900	.899	.819	.825	.822	.428	.398	.412
RNE	.904	.891	.897	.815	.850	**.832**	.415	.355	.383
RI	.902	.894	.898	.808	.845	.826	.408	.312	.354
RA	.895	.904	**.900**	.830	.823	.826	.459	.419	.438
EDA	.890	.901	.895	.808	.820	.814	.454	.317	.373
BT	.910	.880	.895	.820	.823	.822	.378	.543	.446
GM	.901	.898	.899	.824	.827	.825	.414	.425	.419
GM+Revised	.905	.891	.898	.816	.835	.826	.408	.419	.414
ALL	.903	.896	.899	.827	.827	.827	.502	.559	**.529**

(a) Results of SOTA models using data augmentation on task A.

	Non-HS			Non-Subtle HS			Subtle HS		
USE + SVM	P	R	F1	P	R	F1	P	R	F1
RSA	.891	.871	**.881**	.787	.832	**.809**	.800	.103	.182
AAV	.891	.871	.881	.786	.831	.808	.571	.103	.174
RNE	.891	.868	.879	.783	.831	.806	.571	.103	.174
RI	.891	.868	.879	.782	.832	.806	.750	.077	.140
RA	.891	.868	.879	.783	.832	.806	.800	.103	.182
EDA	.892	.870	.881	.788	.828	.807	.263	.128	.172
BT	.892	.868	.880	.789	.831	.809	.739	.436	**.548**
GM	.891	.867	.879	.786	.827	.806	.269	.179	.215
GM+Revised	.892	.866	.879	.785	.826	.805	.286	.205	.239
ALL	.888	.874	.881	.797	.818	.807	.263	.256	.260
BERT	P	R	F1	P	R	F1	P	R	F1
RSA	.894	.911	**.902**	.840	.828	.834	.200	.051	.082
AAV	.898	.896	.897	.824	.836	.830	.300	.154	.203
RNE	.899	.895	.897	.826	.839	.833	.400	.256	.312
RI	.899	.889	.894	.819	.840	.830	.240	.154	.188
RA	.906	.893	.900	.823	.850	.836	.190	.103	.133
EDA	.902	.885	.893	.813	.845	.829	.143	.077	.100
BT	.898	.900	.899	.839	.832	.835	.304	.359	.329
GM	.899	.899	.899	.836	.839	**.837**	.194	.154	.171
GM+Revised	.903	.893	.898	.826	.843	.835	.206	.179	.192
ALL	.904	.883	.893	.813	.845	.829	.385	.385	**.385**
DeBERTaV3	P	R	F1	P	R	F1	P	R	F1
RSA	.922	.894	.908	.826	.879	.852	.333	.103	.157
AAV	.910	.907	.908	.841	.858	.849	.267	.103	.148
RNE	.923	.893	.907	.829	.881	.854	.261	.154	.194
RI	.910	.894	.902	.828	.860	.843	.364	.205	.262
RA	.923	.884	.903	.815	.883	.848	.188	.077	.109
EDA	.924	.888	.905	.819	.882	.850	.188	.077	.109
BT	.920	.897	.908	.835	.876	.855	.385	.256	.308
GM	.911	.910	**.911**	.847	.860	**.854**	.316	.154	.207
GM+Revised	.911	.902	.907	.837	.856	.846	.267	.205	.232
ALL	.926	.881	.903	.817	.877	.846	.306	.385	**.341**
HateBERT	P	R	F1	P	R	F1	P	R	F1
RSA	.900	.893	.896	.823	.841	.832	.273	.154	.197
AAV	.901	.894	.897	.823	.842	.832	.292	.179	.222
RNE	.897	.894	.896	.823	.836	.829	.167	.103	.127
RI	.906	.886	.896	.812	.852	.832	.176	.077	.107
RA	.897	.892	.895	.816	.836	.826	.077	.026	.038
EDA	.902	.890	.896	.819	.845	.832	.217	.128	.161
BT	.909	.883	.896	.820	.848	.834	.207	.308	.247
GM	.899	.898	**.899**	.831	.834	**.832**	.250	.231	.240
GM+Revised	.894	.898	.896	.826	.826	.826	.192	.128	.154
ALL	.903	.881	.892	.816	.844	.830	.391	.462	**.424**

(b) Results of SOTA models using data augmentation on task B.

Table 6: Obtained results on tasks A and B by all models and different types of augmented data.

B Implicit Properties

In the following part, we provide a list of implicit properties with their definitions. All the examples illustrating implicit properties are used in implicit hateful messages and their descriptions are presented in the annotation guidelines.

Antithesis – the rhetorical contrast of ideas through parallel arrangements of words, clauses, or sentences (as in "action, not words" or "they promised freedom and provided slavery") (Merriam-Webster, 2022)

Black humor – humor marked by the use of usually morbid, ironic, grotesquely comic episodes; humor treating sinister subjects like death, disease, deformity, handicap or warfare with bitter amusement (Willinger et al., 2017)

Circumlocution – the use of an unnecessarily large number of words to express an idea (Merriam-Webster, 2022)

Context – the parts of a discourse that surround a word or passage and can throw light on its meaning (Dadvar et al., 2013)

Euphemism – the substitution of an agreeable or inoffensive expression for one that may suggest something unpleasant (Casas Gómez, 2009)

Exaggeration (hyperbole) – an act or instance of exaggerating something: overstatement of the truth (Troiano et al., 2018)

Extralinguistic knowledge – any knowledge that exists outside knowledge of the language. In other words, it refers to knowledge that an author or a recipient of a message may possess about the message itself or about the world, but which is not expressed by any linguistic means.

Fallacy – a false or mistaken idea; an often plausible argument using false or invalid inference (Merriam-Webster, 2022)

Humiliation – the embarrassment and shame a person feels when someone makes them appear stupid or when they make a mistake in public (Dictionary, 2022)

Inference – something that is inferred. The premises and conclusion of a process of inferring (Merriam-Webster, 2022)

Irony – the use of words to express something other than and especially the opposite of the literal meaning; incongruity between the actual result of a sequence of events and the normal or expected result (Potamias et al., 2020)

Metaphor – a figure of speech in which a word or phrase literally denoting one kind of object or idea is used in place of another to suggest a likeness or analogy between them (Choi et al., 2021; Gao et al., 2018)

Metonymy – a figure of speech consisting of the use of the name of one thing for that of another of which it is an attribute or with which it is associated (such as "crown" in "lands belonging to the crown") (Merriam-Webster, 2022)

Rhetorical question – a question not intended to require an answer, used mainly for dramatic effect (Frank, 1990)

Sarcasm – a mode of satirical wit depending on its effect on bitter, caustic, and often ironic language usually directed against an individual. Sarcasm differs from irony with one distinct characteristic: negativity. Sarcasm is mostly witty mockery having a negative connotation whereas irony does not represent negativity (Potamias et al., 2020)

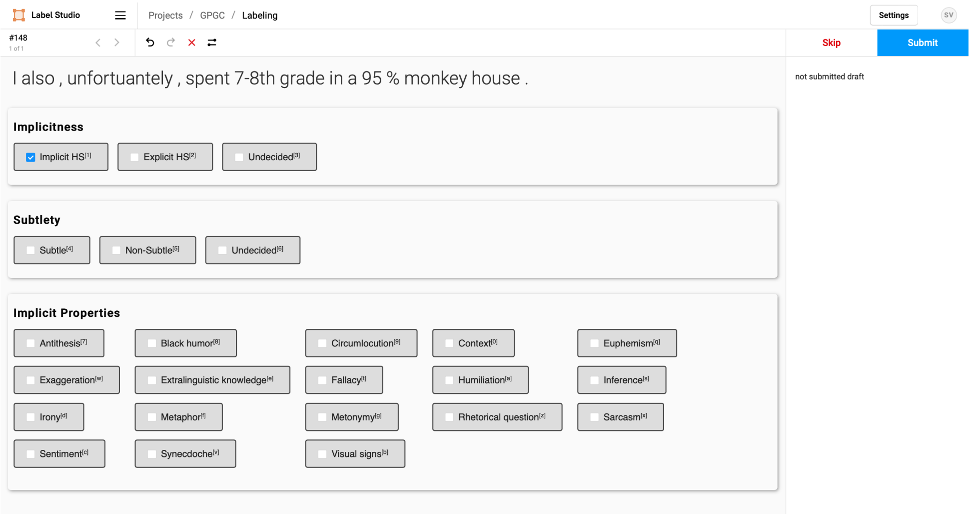
Sentiment – an attitude, thought, or judgment prompted by feeling; the emotional significance of a passage or expression as distinguished from its verbal context (Li et al., 2021)

Synecdoche – a figure of speech by which a part is put for the whole, the whole for a part, the species for the genus, the genus for the species, or the name of the material for the thing made (Merriam-Webster, 2022)

Visual signs – punctuation marks, quotes, and use of uppercase that play a role of support in hate messages.

C Annotation Tool Interface

Figure 3a demonstrates a screenshot of the annotation interface of the Label Studio tool used for the labeling process. According to the annotation scheme represented by three annotation layers (discussed in Section 3 and Subsection 4.2) Label Studio has three consecutive annotation steps. The first step consists in implicitness with three choices: Implicit HS, Explicit HS, Undecided, keeping in mind that the tool allows to filter Non-Hate out before starting the labeling process. The first choice of Implicit HS or Explicit HS brings in the appearance of the second step of subtlety with three choices: Subtle, Non-Subtle, Undecided. This step does not appear with an Undecided choice at the previous step. As well as that, the choice of Implicit HS triggers the appearance of the third step which consists of implicit properties being characteristic of only implicit messages. Figure 3b shows the shape of the resultant dataset after annotation.



I also , unfortuantely , spent 7-8th grade in a 95 % monkey house .

Implicitness

☑ Implicit HS[1] ☐ Explicit HS[2] ☐ Undecided[3]

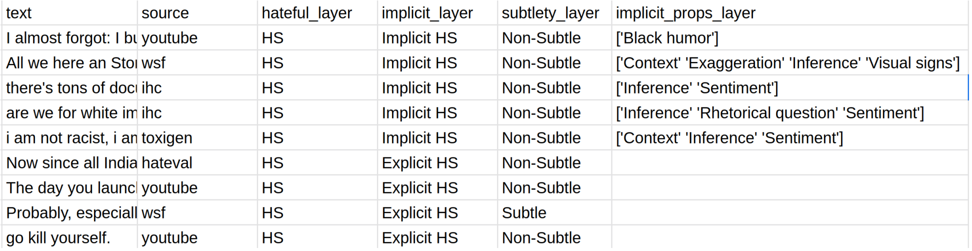
Subtlety

☐ Subtle[4] ☐ Non-Subtle[5] ☐ Undecided[6]

Implicit Properties

☐ Antithesis[7] ☐ Black humor[8] ☐ Circumlocution[9] ☐ Context[0] ☐ Euphemism[q]
☐ Exaggeration[w] ☐ Extralinguistic knowledge[e] ☐ Fallacy[t] ☐ Humiliation[a] ☐ Inference[s]
☐ Irony[d] ☐ Metaphor[f] ☐ Metonymy[g] ☐ Rhetorical question[z] ☐ Sarcasm[x]
☐ Sentiment[c] ☐ Synecdoche[v] ☐ Visual signs[b]

(a) Annotation tool interface.



text	source	hateful_layer	implicit_layer	subtlety_layer	implicit_props_layer
I almost forgot: I bu	youtube	HS	Implicit HS	Non-Subtle	['Black humor']
All we here an Sto	wsf	HS	Implicit HS	Non-Subtle	['Context' 'Exaggeration' 'Inference' 'Visual signs']
there's tons of doc	ihc	HS	Implicit HS	Non-Subtle	['Inference' 'Sentiment']
are we for white im	ihc	HS	Implicit HS	Non-Subtle	['Inference' 'Rhetorical question' 'Sentiment']
i am not racist, i an	toxigen	HS	Implicit HS	Non-Subtle	['Context' 'Inference' 'Sentiment']
Now since all India	hateval	HS	Explicit HS	Non-Subtle	
The day you launc	youtube	HS	Explicit HS	Non-Subtle	
Probably, especiall	wsf	HS	Explicit HS	Subtle	
go kill yourself.	youtube	HS	Explicit HS	Non-Subtle	

(b) Sample of the ISHate dataset after the annotation process.

Figure 3: Label Studio interface to enhance the 7 HS datasets described in Section 4 with three new additional annotation layers: *implicit_layer* (Explicit HS/Implicit HS), *subtlety_layer* (Non-Subtle HS/Subtle HS), and *implicit_props_layer* (Antithesis/Black humor/Context/etc.). The annotation layer *hateful_layer* (Non-HS/HS) consists of the already provided labels of each HS corpus, with the exception of the Youtube dataset where we re-annotated it.

MTEB: Massive Text Embedding Benchmark

Niklas Muennighoff[1], **Nouamane Tazi**[1], **Loïc Magne**[1], **Nils Reimers**[2]*
[1]Hugging Face [2]cohere.ai
[1]`firstname@huggingface.co` [2]`info@nils-reimers.de`

Abstract

Text embeddings are commonly evaluated on a small set of datasets from a single task not covering their possible applications to other tasks. It is unclear whether state-of-the-art embeddings on semantic textual similarity (STS) can be equally well applied to other tasks like clustering or reranking. This makes progress in the field difficult to track, as various models are constantly being proposed without proper evaluation. To solve this problem, we introduce the Massive Text Embedding Benchmark (MTEB). MTEB spans 8 embedding tasks covering a total of 58 datasets and 112 languages. Through the benchmarking of 33 models on MTEB, we establish the most comprehensive benchmark of text embeddings to date. We find that no particular text embedding method dominates across all tasks. This suggests that the field has yet to converge on a universal text embedding method and scale it up sufficiently to provide state-of-the-art results on all embedding tasks. MTEB comes with open-source code and a public leaderboard at `https://github.com/embeddings-benchmark/mteb`.

1 Introduction

Natural language embeddings power a variety of use cases from clustering and topic representation (Aggarwal and Zhai, 2012; Angelov, 2020) to search systems and text mining (Huang et al., 2020; Zhu et al., 2021; Nayak, 2019) to feature representations for downstream models (Saharia et al., 2022; Borgeaud et al., 2022). Using generative language models or cross-encoders for these applications is often intractable, as they may require exponentially more computations (Reimers and Gurevych, 2019).

However, the evaluation regime of current text embedding models rarely covers the breadth of their possible use cases. For example, SimCSE (Gao et al., 2021b) or SBERT (Reimers and Gurevych, 2019) solely evaluate on STS and classification tasks, leaving open questions about the transferability of the embedding models to search or clustering tasks. STS is known to poorly correlate with other real-world use cases (Neelakantan et al., 2022; Wang et al., 2021). Further, evaluating embedding methods on many tasks requires implementing multiple evaluation pipelines. Implementation details like pre-processing or hyperparameters may influence the results making it unclear whether performance improvements simply come from a favorable evaluation pipeline. This leads to the "blind" application of these models to new use cases in industry or requires incremental work to reevaluate them on different tasks.

The Massive Text Embedding Benchmark (MTEB) aims to provide clarity on how models perform on a variety of embedding tasks and thus serves as the gateway to finding universal text embeddings applicable to a variety of tasks. MTEB consists of 58 datasets covering 112 languages from 8 embedding tasks: Bitext mining, classification, clustering, pair classification, reranking, retrieval, STS and summarization. MTEB software is available open-source[1] enabling evaluation of any embedding model by adding less than 10 lines of code. Datasets and the MTEB leaderboard are available on the Hugging Face Hub[2].

We evaluate over 30 models on MTEB with additional speed and memory benchmarking to provide a holistic view of the state of text embedding models. We cover both models available open-source as well as models accessible via APIs, such as the OpenAI Embeddings endpoint. We find there to be no single best solution, with different models dominating different tasks. Our benchmarking sheds light on the weaknesses and strengths of individual

---

[*] Most of the work done while at Hugging Face

[1] `https://github.com/embeddings-benchmark/mteb`

[2] `https://huggingface.co/spaces/mteb/leaderboard`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2014–2037
May 2-6, 2023

models, such as SimCSE's (Gao et al., 2021b) low performance on clustering and retrieval despite its strong performance on STS. We hope our work makes selecting the right embedding model easier and simplifies future embedding research.

2 Related Work

2.1 Benchmarks

Benchmarks, such as (Super)GLUE (Wang et al., 2018, 2019) or Big-BENCH (Srivastava et al., 2022), and evaluation frameworks (Gao et al., 2021a) play a key role in driving NLP progress. Yearly released SemEval datasets (Agirre et al., 2012, 2013, 2014, 2015, 2016) are commonly used as the go-to benchmark for text embeddings. SemEval datasets correspond to the task of semantic textual similarity (STS) requiring models to embed similar sentences with geometrically close embeddings. Due to the limited expressivity of a single SemEval dataset, SentEval (Conneau and Kiela, 2018) aggregates multiple STS datasets. SentEval focuses on fine-tuning classifiers on top of embeddings. It lacks tasks like retrieval or clustering, where embeddings are directly compared without additional classifiers. Further, the toolkit was proposed in 2018 and thus does not provide easy support for recent trends like text embeddings from transformers (Reimers and Gurevych, 2019). Due to the insufficiency of STS benchmarking, USEB (Wang et al., 2021) was introduced consisting mostly of reranking tasks. Consequently, it does not cover tasks like retrieval or classification. Meanwhile, the recently released BEIR Benchmark (Thakur et al., 2021) has become the standard for the evaluation of embeddings for zero-shot information retrieval.

MTEB unifies datasets from different embedding tasks into a common, accessible evaluation framework. MTEB incorporates SemEval datasets (STS11 - STS22) and BEIR alongside a variety of other datasets from various tasks to provide a holistic performance review of text embedding models.

2.2 Embedding Models

Text embedding models like Glove (Pennington et al., 2014) lack context awareness and are thus commonly labeled as Word Embedding Models. They consist of a layer mapping each input word to a vector often followed by an averaging layer to provide a final embedding invariant of input length. Transformers (Vaswani et al., 2017) inject context awareness into language models via self-attention and form the foundation of most recent embedding models. BERT (Devlin et al., 2018) uses the transformer architecture and performs large-scale self-supervised pre-training. The resulting model can directly be used to produce text embeddings via an averaging operation alike Glove. Building on InferSent (Conneau et al., 2017), SBERT (Reimers and Gurevych, 2019) demonstrated it to be beneficial to perform additional fine-tuning of the transformer for competitive embedding performance. Most recent fine-tuned embedding models use a contrastive loss objective to perform supervised fine-tuning on positive and negative text pairs (Gao et al., 2021b; Wang et al., 2021; Ni et al., 2021b; Muennighoff, 2022). Due to the large variety of available pre-trained transformers (Wolf et al., 2020), there is an at least equally large variety of potential text embedding models to be explored. This leads to confusion about which model provides practitioners with the best performance for their embedding use case.

We benchmark both word embedding and transformer models on MTEB quantifying gains provided by often much slower context aware models.

3 The MTEB Benchmark

3.1 Desiderata

MTEB is built on a set of desiderata: **(a) Diversity:** MTEB aims to provide an understanding of the usability of embedding models in various use cases. The benchmark comprises 8 different tasks, with up to 15 datasets each. Of the 58 total datasets in MTEB, 10 are multilingual, covering 112 different languages. Sentence-level and paragraph-level datasets are included to contrast performance on short and long texts. **(b) Simplicity:** MTEB provides a simple API for plugging in any model that given a list of texts can produce a vector for each list item with a consistent shape. This makes it possible to benchmark a diverse set of models. **(c) Extensibility:** New datasets for existing tasks can be benchmarked in MTEB via a single file that specifies the task and a Hugging Face dataset name where the data has been uploaded (Lhoest et al., 2021). New tasks require implementing a task interface for loading the data and an evaluator for benchmarking. We welcome dataset, task or metric contributions from the community via pull requests to continue the development of MTEB. **(d) Reproducibility:** Through versioning at a dataset and software level, we aim to make it easy to repro-

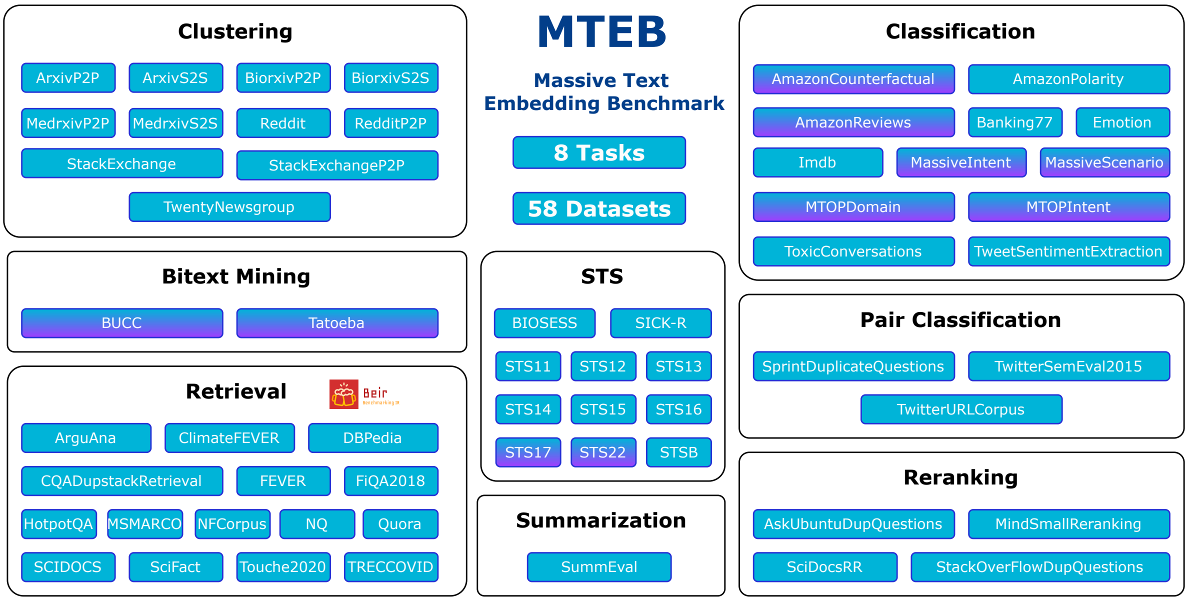


Figure 1: An overview of tasks and datasets in MTEB. Multilingual datasets are marked with a purple shade.

duce results in MTEB. JSON files corresponding to all results available in this paper have been made available together with the MTEB benchmark[3].

3.2 Tasks and Evaluation

Figure 1 provides an overview of tasks and datasets available in MTEB. Dataset statistics are available in Table 2. The benchmark consists of the following 8 task types:

Bitext Mining Inputs are two sets of sentences from two different languages. For each sentence in the first set, the best match in the second set needs to be found. The matches are commonly translations. The provided model is used to embed each sentence and the closest pairs are found via cosine similarity. F1 serves as the main metric for bitext mining. Accuracy, precision and recall are also computed.

Classification A train and test set are embedded with the provided model. The train set embeddings are used to train a logistic regression classifier with 100 maximum iterations, which is scored on the test set. The main metric is accuracy with average precision and f1 additionally provided.

Clustering Given a set of sentences or paragraphs, the goal is to group them into meaningful clusters. A mini-batch k-means model with batch size 32 and k equal to the number of different labels (Pedregosa et al., 2011) is trained on the embedded texts. The model is scored using v-measure (Rosenberg and Hirschberg, 2007). V-measure does not depend on the cluster label, thus the permutation of labels does not affect the score.

Pair Classification A pair of text inputs is provided and a label needs to be assigned. Labels are typically binary variables denoting duplicate or paraphrase pairs. The two texts are embedded and their distance is computed with various metrics (cosine similarity, dot product, euclidean distance, manhattan distance). Using the best binary threshold accuracy, average precision, f1, precision and recall are computed. The average precision score based on cosine similarity is the main metric.

Reranking Inputs are a query and a list of relevant and irrelevant reference texts. The aim is to rank the results according to their relevance to the query. The model is used to embed the references which are then compared to the query using cosine similarity. The resulting ranking is scored for each query and averaged across all queries. Metrics are mean MRR@k and MAP with the latter being the main metric.

Retrieval Each dataset consists of a corpus, queries and a mapping for each query to relevant documents from the corpus. The aim is to find these relevant documents. The provided model is used

---

[3] `https://huggingface.co/datasets/mteb/results`

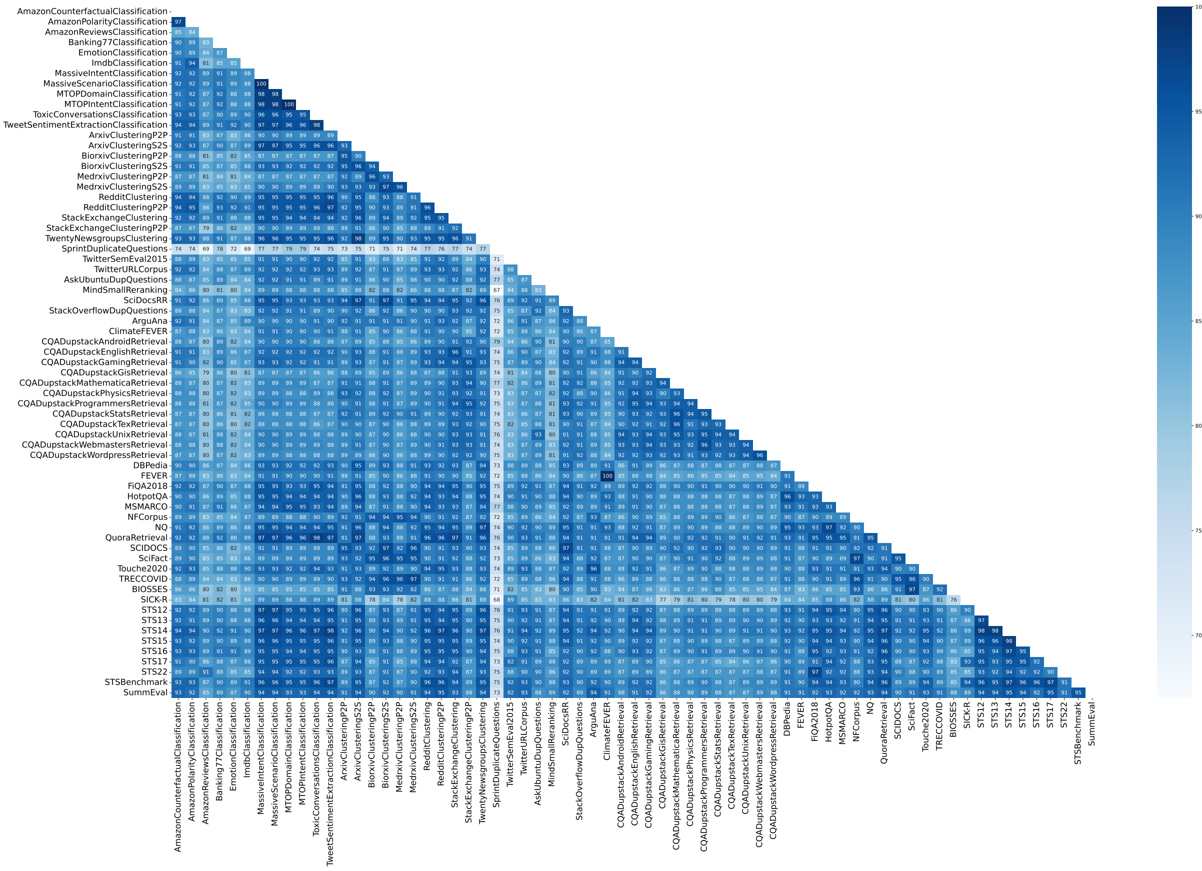


Figure 2: Similarity of MTEB datasets. We use the best model on MTEB STS (ST5-XXL, see Table 1) to embed 100 samples for each dataset. Cosine similarities between the averaged embeddings are computed and visualized.

to embed all queries and all corpus documents and similarity scores are computed using cosine similarity. After ranking the corpus documents for each query based on the scores, nDCG@k, MRR@k, MAP@k, precision@k and recall@k are computed for several values of k. nDCG@10 serves as the main metric. MTEB reuses datasets and evaluation from BEIR (Thakur et al., 2021).

Semantic Textual Similarity (STS) Given a sentence pair the aim is to determine their similarity. Labels are continuous scores with higher numbers indicating more similar sentences. The provided model is used to embed the sentences and their similarity is computed using various distance metrics. Distances are benchmarked with ground truth similarities using Pearson and Spearman correlations. Spearman correlation based on cosine similarity serves as the main metric (Reimers et al., 2016).

Summarization A set of human-written and machine-generated summaries are provided. The aim is to score the machine summaries. The provided model is first used to embed all summaries. For each machine summary embedding, distances to all human summary embeddings are computed. The closest score (e.g. highest cosine similarity) is kept and used as the model's score of a single machine-generated summary. Pearson and Spearman correlations with ground truth human assessments of the machine-generated summaries are computed. Like for STS, Spearman correlation based on cosine similarity serves as the main metric (Reimers et al., 2016).

3.3 Datasets

To further the diversity of MTEB, datasets of varying text lengths are included. All datasets are grouped into three categories:

Sentence to sentence (S2S) A sentence is compared with another sentence. An example of S2S are all current STS tasks in MTEB, where the similarity between two sentences is assessed.

Paragraph to paragraph (P2P) A paragraph is compared with another paragraph. MTEB imposes no limit on the input length, leaving it up to the

Num. Datasets (→)	Class. 12	Clust. 11	PairClass. 3	Rerank. 4	Retr. 15	STS 10	Summ. 1	Avg. 56
Self-supervised methods								
Glove	57.29	27.73	70.92	43.29	21.62	61.85	28.87	41.97
Komninos	57.65	26.57	72.94	44.75	21.22	62.47	30.49	42.06
BERT	61.66	30.12	56.33	43.44	10.59	54.36	29.82	38.33
SimCSE-BERT-unsup	62.50	29.04	70.33	46.47	20.29	74.33	31.15	45.45
Supervised methods								
SimCSE-BERT-sup	67.32	33.43	73.68	47.54	21.82	79.12	23.31	48.72
coCondenser-msmarco	64.71	37.64	81.74	51.84	32.96	76.47	29.50	52.35
Contriever	66.68	41.10	82.53	53.14	41.88	76.51	30.36	56.00
SPECTER	52.37	34.06	61.37	48.10	15.88	61.02	27.66	40.28
LaBSE	62.71	29.55	78.87	48.42	18.99	70.80	31.05	45.21
LASER2	53.65	15.28	68.86	41.44	7.93	55.32	26.80	33.63
MiniLM-L6	63.06	42.35	82.37	58.04	41.95	78.90	30.81	56.26
MiniLM-L12	63.21	41.81	82.41	58.44	42.69	79.80	27.90	56.53
MiniLM-L12-multilingual	64.30	37.14	78.45	53.62	32.45	78.92	30.67	52.44
MPNet	65.07	43.69	83.04	**59.36**	43.81	80.28	27.49	57.78
MPNet-multilingual	67.91	38.40	80.81	53.80	35.34	80.73	**31.57**	54.71
Ada Similarity	70.44	37.52	76.86	49.02		78.60	26.94	
SGPT-125M-nli	61.46	30.95	71.78	47.56	20.90	74.71	30.26	45.97
SGPT-5.8B-nli	70.14	36.98	77.03	52.33	32.34	80.53	30.38	53.74
SGPT-125M-msmarco	60.72	35.79	75.23	50.58	37.04	73.41	28.90	51.23
SGPT-1.3B-msmarco	66.52	39.92	79.58	54.00	44.49	75.74	25.44	56.11
SGPT-2.7B-msmarco	67.13	39.83	80.65	54.67	46.54	76.83	27.87	57.12
SGPT-5.8B-msmarco	68.13	40.35	82.00	56.56	**50.25**	78.10	24.75	58.81
SGPT-BLOOM-7.1B-msmarco	66.19	38.93	81.90	55.65	48.21	77.74	24.99	57.44
GTR-Base	65.25	38.63	83.85	54.23	44.67	77.07	29.67	56.19
GTR-Large	67.14	41.60	85.33	55.36	47.42	78.19	29.50	58.28
GTR-XL	67.11	41.51	**86.13**	55.96	47.96	77.80	30.21	58.42
GTR-XXL	67.41	42.42	86.12	56.65	48.48	78.38	30.64	58.97
ST5-Base	69.81	40.21	85.17	53.09	33.63	81.14	31.39	55.27
ST5-Large	72.31	41.65	84.97	54.00	36.71	81.83	29.64	57.06
ST5-XL	72.84	42.34	86.06	54.71	38.47	81.66	29.91	57.87
ST5-XXL	**73.42**	**43.71**	85.06	56.43	42.24	**82.63**	30.08	**59.51**

Table 1: Average of the main metric (see Section 3.2) per task per model on MTEB English subsets.

models to truncate if necessary. Several clustering tasks are framed as both S2S and P2P tasks. The former only compare titles, while the latter include both title and content. For ArxivClustering, for example, abstracts are concatenated to the title in the P2P setting.

Sentence to paragraph (S2P) A few retrieval datasets are mixed in a S2P setting. Here a query is a single sentence, while documents are long paragraphs consisting of multiple sentences.

Similarities across 56 MTEB datasets are visualized in Figure 2. Several datasets rely on the same corpora, such as ClimateFEVER and FEVER, resulting in a score of 1. Clusters of similar datasets can be seen among CQADupstack variations and STS datasets. S2S and P2P variations of the same dataset tend to also be similar. Scientific datasets, such as SciDocsRR, SciFact, ArxivClustering, show high similarities among each other even when coming from different tasks (Reranking, Retrieval and Clustering in this case).

4 Results

4.1 Models

We evaluate on the test splits of all datasets except for MSMARCO, where the dev split is used following Thakur et al. (2021). We benchmark models claiming state-of-the-art results on various embedding tasks leading to a high representation of transformers (Vaswani et al., 2017). We group models into self-supervised and supervised methods.

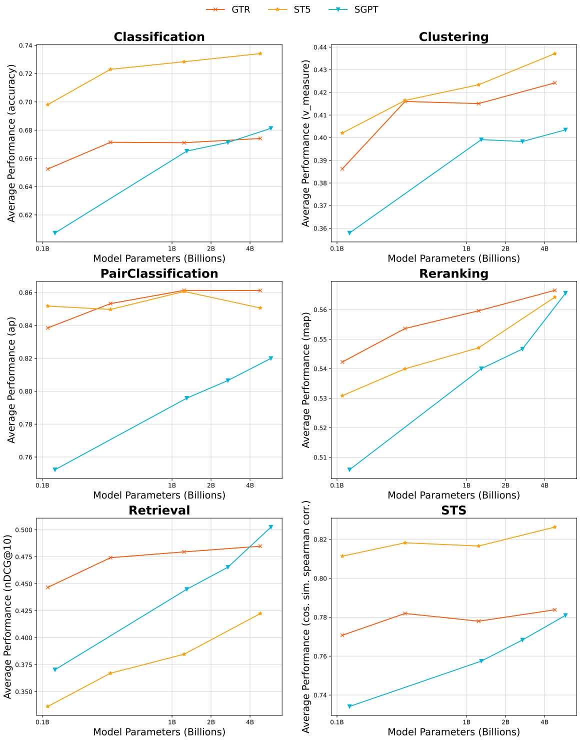


Figure 3: MTEB performance scales with model size. The smallest SGPT variant underperforms similar-sized GTR and ST5 variants. This may be due to the bias-only fine-tuning SGPT employs, which catches up with full fine-tuning only as model size and thus the number of bias parameters increases (Muennighoff, 2022).

Self-supervised methods (a) Transformer-based BERT (Devlin et al., 2018) is trained using self-supervised mask and sentence prediction tasks. By taking the mean across the sequence length (mean-pooling) the model can directly be used to produce text embeddings. SimCSE-Unsup (Gao et al., 2021b) uses BERT as a foundation and performs additional self-supervised training. **(b) Non-transformer**: Komninos (Komninos and Manandhar, 2016) and Glove (Pennington et al., 2014) are two word embedding models that directly map words to vectors. Hence, their embeddings lack context awareness, but provide significant speed-ups.

Supervised methods The original transformer model (Vaswani et al., 2017) consists of an encoder and decoder network. Subsequent transformers often train only encoders like BERT (Devlin et al., 2018) or decoders like GPT (Radford et al., 2019).

(a) Transformer encoder methods coCondenser (Gao and Callan, 2021), Contriever (Izacard et al., 2021), LaBSE (Feng et al., 2020) and SimCSE-BERT-sup (Gao et al., 2021b) are based on the pre-trained BERT model (Devlin et al., 2018). coCondenser and Contriever add a self-supervised stage prior to supervised fine-tuning for a total of three training stages. LaBSE uses BERT to perform additional pre-training on parallel data to produce a competitive bitext mining model. SPECTER (Cohan et al., 2020a) relies on the pre-trained SciBERT (Beltagy et al., 2019) variant instead and fine-tunes on citation graphs. GTR (Ni et al., 2021b) and ST5 (Ni et al., 2021a) are based on the encoder part of the T5 model (Raffel et al., 2020) and only differ in their fine-tuning datasets. After additional self-supervised training, ST5 does contrastive fine-tuning on NLI (Ni et al., 2021a; Gao et al., 2021b) being geared towards STS tasks. Meanwhile, GTR fine-tunes on MS-MARCO and focuses on retrieval tasks. MPNet and MiniLM correspond to fine-tuned embedding models (Reimers and Gurevych, 2019) of the pre-trained MPNet (Song et al., 2020) and MiniLM (Wang et al., 2020) models using diverse datasets to target any embedding use case.

(b) Transformer decoder methods SGPT Bi-Encoders (Muennighoff, 2022) perform contrastive fine-tuning of <0.1% of pre-trained parameters using weighted-mean pooling. Similar to ST5 and GTR, SGPT-nli models are geared towards STS, while SGPT-msmarco models towards retrieval. SGPT-msmarco models embed queries and documents for retrieval with different special tokens to help the model distinguish their role. For non-retrieval tasks, we use its query representations. We benchmark publicly available SGPT models based on GPT-NeoX (Andonian et al., 2021), GPT-J (Wang and Komatsuzaki, 2021) and BLOOM (Scao et al., 2022). Alternatively, cpt-text (Neelakantan et al., 2022) passes pre-trained GPT decoders through a two-stage process using last token pooling to provide embeddings from decoders. We benchmark their models via the OpenAI Embeddings API[4].

(c) Non-transformer LASER (Heffernan et al., 2022) is the only context aware non-transformer model we benchmark, relying on an LSTM (Hochreiter and Schmidhuber, 1997) instead. Similar to LaBSE, the model trains on parallel data and focuses on bitext mining applications.

---

[4] `https://beta.openai.com/docs/guides/embeddings`

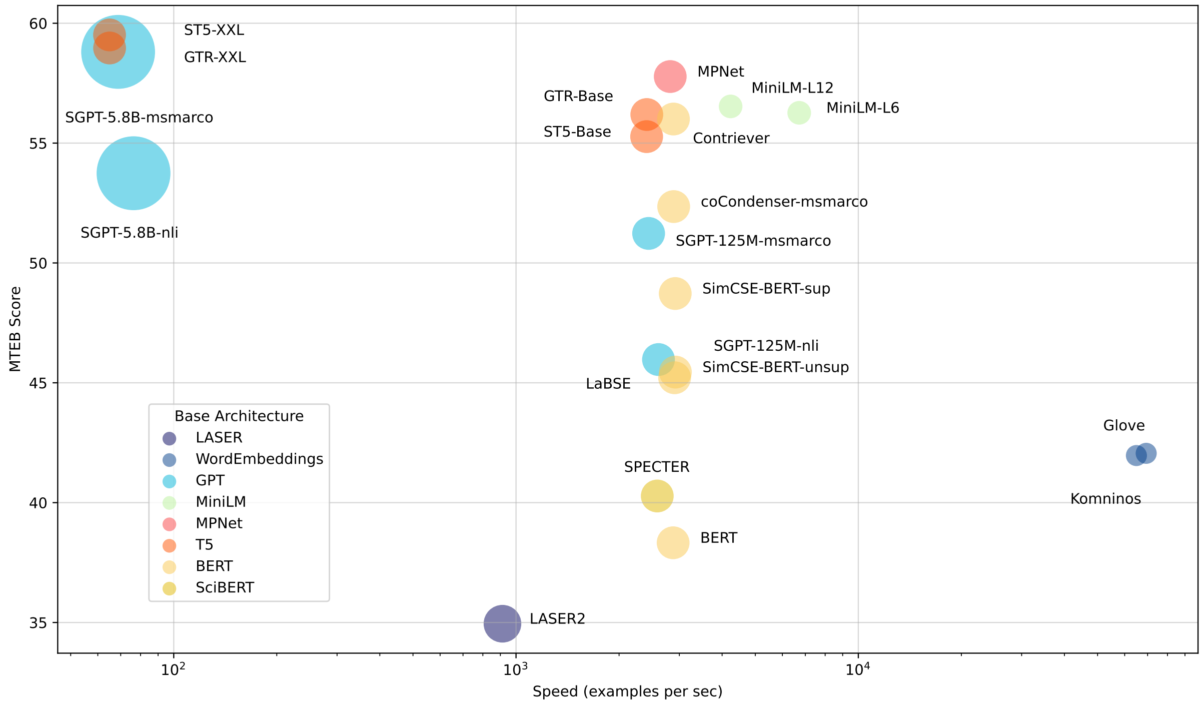


Figure 4: Performance, speed, and size of produced embeddings (size of the circles) of different embedding models. Embedding sizes range from 1.2 kB (Glove / Komninos) to 16.4 kB (SGPT-5.8B) per example. Speed was benchmarked on STS15 using 1x Nvidia A100 80GB with CUDA 11.6.

4.2 Analysis

Based on the results in Table 1, we observe that there is considerable variability between tasks. No model claims the state-of-the-art in all seven English tasks. There is even more variability in the results per dataset present in the appendix. Further, there remains a large gap between self-supervised and supervised methods. Self-supervised large language models have been able to close this gap in many natural language generation tasks (Chowdhery et al., 2022). However, they appear to still require supervised fine-tuning for competitive embedding performance.

We find that performance strongly correlates with model size, see Figure 3. A majority of MTEB tasks are dominated by multi-billion parameter models. However, these come at a significant cost as we investigate in Section 4.3.

Classification ST5 models dominate the classification task across most datasets, as can be seen in detail in the full results in the appendix. ST5-XXL has the highest average performance, 3% ahead of the best non-ST5 model, Ada Similarity.

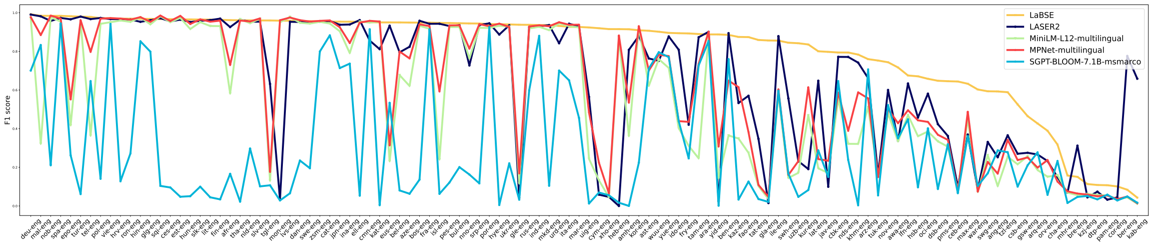
Clustering Despite being almost 50x smaller, the MPNet embedding model is on par with the ST5-XXL state-of-the-art on Clustering. This may be due to the large variety of datasets MPNet (and MiniLM) has been fine-tuned on. Clustering requires coherent distances between a large number of embeddings. Models like SimCSE-sup or SGPT-nli, which are only fine-tuned on a single dataset, NLI, may produce incoherent embeddings when encountering topics unseen during fine-tuning. Relatedly, we find that the query embeddings of SGPT-msmarco and the Ada Search endpoint are competitive with SGPT-nli and the Ada Similarity endpoint, respectively. We refer to the public leaderboard[5] for Ada Search results. This could be due to the MSMARCO dataset being significantly larger than NLI. Thus, while the OpenAI docs recommend using the similarity embeddings for clustering use cases[6], the retrieval query embeddings may be the better choice in some cases.

Pair Classification GTR-XL and GTR-XXL have the strongest performance. Pair classification is closest to STS in its framing, yet models rank significantly differently on the two tasks. This

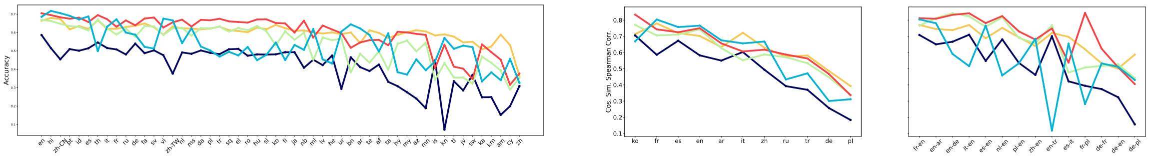
---

[5] `https://huggingface.co/spaces/mteb/leaderboard`

[6] `https://beta.openai.com/docs/guides/embeddings/similarity-embeddings`



(a) Bitext Mining on Tatoeba



(b) Multilingual Classification

(c) Multi- and Crosslingual STS

Figure 5: MTEB multilingual performance. Bitext mining is dominated by LaBSE, while classification and STS results are mixed. SGPT-BLOOM-7B1-msmarco tends to perform well on the languages BLOOM has been pre-trained on, such as Chinese, French and Portuguese.

highlights the importance of benchmarking on a diverse set of tasks to avoid blindly reusing a model for a different task.

Reranking MPNet and MiniLM models perform strongly on reranking tasks. On SciDocsRR (Cohan et al., 2020a) they perform far better than bigger models, which is likely due to parts of SciDocsRR being included in their training data. Our scale of experiments and that of model pre-training make controlling for data contamination challenging. Thus, we ignore overlap of MTEB datasets with model training datasets in MTEB scores. As long as enough datasets are averaged, we believe these effects to be insignificant.

Retrieval SGPT-5.8B-msmarco is the best embedding model on the BEIR subset in MTEB as well as on the full BEIR benchmark (Thakur et al., 2021; Muennighoff, 2022). The even larger 7.1B SGPT model making use of BLOOM (Scao et al., 2022) performs significantly weaker, which is likely due to the multilinguality of BLOOM. Models geared towards STS (SimCSE, ST5, SGPT-nli) perform badly on retrieval tasks. Retrieval tasks are unique in that there are two distinct types of texts: Queries and documents ("asymmetric"), while other tasks only have a single type of text ("symmetric"). On the QuoraRetrieval dataset, which has been shown to be largely symmetric (Muennighoff, 2022), the playing field is more even with SGPT-5.8B-nli outperforming SGPT-5.8B-msmarco, see Table 11.

STS & Summarization Retrieval models (GTR, SGPT-msmarco) perform badly on STS, while ST5-XXL has the highest performance. This highlights the bifurcation of the field into separate embedding models for retrieval (asymmetric) and similarity (symmetric) use cases (Muennighoff, 2022).

4.3 Efficiency

We investigate the latency-performance trade-off of models in Figure 4. The graph allows for significant elimination of model candidates in the model selection process. It brings model selection down to three clusters:

Maximum speed Word Embedding models offer maximum speed with Glove taking the lead on both performance and speed, thus making the choice simple in this case.

Maximum performance If latency is less important than performance, the left-hand side of the graph offers a cluster of highly performant, but slow models. Depending on the task at hand, GTR-XXL, ST5-XXL or SGPT-5.8B may be the right choice, see Section 4.2. SGPT-5.8B comes with the additional caveat of its high-dimensional embeddings requiring more storage.

Speed and performance The fine-tuned MPNet and MiniLM models lead the middle cluster making the choice easy.

4.4 Multilinguality

MTEB comes with 10 multilingual datasets across bitext mining, classification and STS tasks. We investigate performance on these in Figure 5. Tabular results can be found in Tables 12, 13 and 14.

Bitext Mining LaBSE (Feng et al., 2020) performs strongly across a wide array of languages in bitext mining. Meanwhile, LASER2 shows high variance across different languages. While there are additional language-specific LASER2 models available for some of the languages we benchmark, we use the default multilingual LASER2 model for all languages. This is to provide a fair one-to-one comparison of models. In practice, however, the high variance of LASER2's performance may be resolved by mixing its model variants. MPNet, MiniLM and SGPT-BLOOM-7B1-msmarco perform poorly on languages they have not been pre-trained on, such as German for the latter.

Classification & STS On multilingual classification and STS, the multilingual MPNet provides the overall strongest performance. It outperforms the slightly faster multilingual MiniLM on almost all languages. Both models have been trained on the same languages, thus bringing decision-making down to performance vs speed. SGPT-BLOOM-7B1-msmarco provides state-of-the-art performance on languages like Hindi, Portuguese, Chinese or French, which the model has seen extensively during pre-training. It also performs competitively on languages like Russian or Japanese that unintentionally leaked into its pre-training data (Muennighoff et al., 2022). However, it is not much ahead of the much cheaper MPNet. LASER2 performs consistently worse than other models.

5 Conclusion

In this work, we presented the Massive Text Embedding Benchmark (MTEB). Consisting of 8 text embedding tasks with up to 15 datasets each and covering 112 languages, MTEB aims to provide reliable embedding performance estimates. By open-sourcing MTEB alongside a leaderboard, we provide a foundation for further pushing the state-of-the-art of available text embeddings.

To introduce MTEB, we have conducted the most comprehensive benchmarking of text embeddings to date. Through the course of close to 5,000 experiments on over 30 different models, we have set up solid baselines for future research to build on. We found model performance on different tasks to vary strongly with no model claiming state-of-the-art on all tasks. Our studies on scaling behavior, model efficiency and multilinguality revealed various intricacies of models that should ease the decision-making process for future research or industry applications of text embeddings.

We welcome task, dataset or metric contributions to the MTEB codebase[7] as well as additions to the leaderboard via our automatic submission format[8].

6 Limitations of MTEB

While MTEB aims to be a diverse benchmark to provide holistic performance reviews, the benchmark has its limitations. We list them here:

1. Long document datasets MTEB covers multiple text lengths (S2S, P2P, S2P), but very long documents are still missing. The longest datasets in MTEB have a few hundred words, and longer text sizes could be relevant for use cases like retrieval.

2. Task imbalance Tasks in MTEB have a different amount of datasets with summarization consisting of only a single dataset. This means MTEB average scores, which are computed over all datasets, are biased towards tasks with many datasets, notably retrieval, classification and clustering. As MTEB grows, we hope to add more datasets to currently underrepresented tasks like summarization or pair classification.

3. Multinguality MTEB contains multilingual classification, STS and bitext mining datasets. However, retrieval and clustering are English-only. SGPT-BLOOM-7B1-msmarco is geared towards multilingual retrieval datasets and due to the lack thereof cannot be comprehensively benchmarked in MTEB. Further, MTEB does not contain any code datasets that could be used to benchmark code models (Neelakantan et al., 2022; Allal et al., 2023). It should be easy to extend MTEB with datasets, such as CodeSearchNet (Husain et al., 2019), TyDI QA (Clark et al., 2020), XOR QA (Asai et al., 2020) or MIRACL (Zhang et al., 2022).

4. Additional modalities Text embeddings are commonly used as input features for downstream models, such as in our classification task. This can involve other modalities, notably image content (Carvalho et al., 2018; Tan and Bansal, 2019; Muennighoff, 2020; Nichol et al., 2021; Saharia

---

[7] `https://github.com/embeddings-benchmark/mteb`

[8] `https://huggingface.co/spaces/mteb/leaderboard`

et al., 2022; Weinbach et al., 2022). We have focused solely on natural language applications and leave extensive benchmarking of text embeddings as inputs for other modalities to future work.

Acknowledgments

This work was granted access to the HPC resources of Institut du développement et des ressources en informatique scientifique (IDRIS) du Centre national de la recherche scientifique (CNRS) under the allocation 2021-A0101012475 made by Grand équipement national de calcul intensif (GENCI). In particular, all the evaluations and data processing ran on the Jean Zay cluster of IDRIS, and we want to thank the IDRIS team for responsive support throughout the project, in particular Rémi Lacroix.

We thank Douwe Kiela, Teven Le Scao and Nandan Thakur for feedback and suggestions.

References

Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text clustering algorithms. In *Mining text data*, pages 77–128. Springer.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M Cer, Mona T Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval@ COLING*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics).

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

Loubna Ben Allal, Raymond Li, Denis Kocetkov, Chenghao Mou, Christopher Akiki, Carlos Munoz Ferrandis, Niklas Muennighoff, Mayank Mishra, Alex Gu, Manan Dey, et al. 2023. Santacoder: don't reach for the stars! *arXiv preprint arXiv:2301.03988*.

Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Shivanshu Purohit, Tri Songz, Phil Wang, and Samuel Weinbach. 2021. GPT-NeoX: Large scale autoregressive language modeling in pytorch.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Akari Asai, Jungo Kasai, Jonathan H Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2020. Xor qa: Cross-lingual open-retrieval question answering. *arXiv preprint arXiv:2010.11856*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, pages 2206–2240. PMLR.

Micael Carvalho, Rémi Cadène, David Picard, Laure Soulier, Nicolas Thome, and Matthieu Cord. 2018. Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 35–44.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020a. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020b. Specter: Document-level representation learning using citation-informed transformers.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2020. Summeval: Re-evaluating summarization evaluation.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. 2022. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. 2021a. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*.

Luyu Gao and Jamie Callan. 2021. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Gregor Geigle, Nils Reimers, Andreas Rücklé, and Iryna Gurevych. 2021. Tweac: Transformer with extendable qa agent classifiers.

Kevin Heffernan, Onur Çelebi, and Holger Schwenk. 2022. Bitext mining using distilled sentence representations for low-resource languages. *arXiv preprint arXiv:2205.12654*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in facebook search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2553–2561.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Alexandros Komninos and Suresh Manandhar. 2016. Dependency based embeddings for sentence classification tasks. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1490–1500.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of The 2017 Conference on Empirical Methods on Natural Language Processing (EMNLP)*, pages 1235–1245. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. *arXiv preprint arXiv:2109.02846*.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2020. Mtop: A comprehensive multilingual task-oriented semantic parsing benchmark.

Xueqing Liu, Chi Wang, Yue Leng, and ChengXiang Zhai. 2018. Linkso: a dataset for learning to retrieve similar question answer pairs on software development forums. In *Proceedings of the 4th ACM SIGSOFT International Workshop on NLP for Software Engineering*, pages 2–5.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. RecSys '13, New York, NY, USA. Association for Computing Machinery.

Niklas Muennighoff. 2020. Vilio: State-of-the-art visio-linguistic models applied to hateful memes. *arXiv preprint arXiv:2012.07788*.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. 2022. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*.

Pandu Nayak. 2019. Understanding searches better than ever before.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. 2022. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. 2021a. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. 2021b. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. 2021. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*.

James O'Neill, Polina Rozenshtein, Ryuichi Kiryo, Motoko Kubota, and Danushka Bollegala. 2021. I wish i would have loved this one, but i didn't – a multilingual dataset for counterfactual detection in product reviews.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Nils Reimers, Philip Beyer, and Iryna Gurevych. 2016. Task-oriented intrinsic evaluation of semantic textual similarity. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 87–96.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Facebook Research. Tatoeba multilingual test set.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. pages 410–420.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Darsh Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. 2018. Adversarial domain adaptation for duplicate question detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1056–1063, Brussels, Belgium. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. Tsdae: Using transformer-based sequential denoising auto-encoder for unsupervised sentence embedding learning. *arXiv preprint arXiv:2104.06979*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Samuel Weinbach, Marco Bellagente, Constantin Eichenberg, Andrew Dai, Robert Baldock, Souradeep Nanda, Björn Deiseroth, Koen Oostermeijer, Hannah Teufel, and Andres Felipe Cruz-Salinas. 2022. M-vader: A model for diffusion with multimodal context. *arXiv preprint arXiv:2212.02936*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. 2020. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606.

Wei Xu, Chris Callison-Burch, and William B Dolan. 2015. Semeval-2015 task 1: Paraphrase and semantic similarity in twitter (pit). In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 1–11.

Xinyu Zhang, Nandan Thakur, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Mehdi Rezagholizadeh, and Jimmy Lin. 2022. Making a miracl: Multilingual information retrieval across a continuum of languages. *arXiv preprint arXiv:2210.09984*.

Jeffrey Zhu, Mingqin Li, Jason Li, and Cassandra Oduola. 2021. Bing delivers more contextualized search using quantized transformer inference on nvidia gpus in azure.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2016. Towards preparation of the second bucc shared task: Detecting parallel sentences in comparable corpora. In *Proceedings of the Ninth Workshop on Building and Using Comparable Corpora. European Language Resources Association (ELRA), Portoroz, Slovenia*, pages 38–43.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2017. Overview of the second bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of the 10th Workshop on Building and Using Comparable Corpora*, pages 60–67.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2018. Overview of the third bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of 11th workshop on building and using comparable corpora*, pages 39–42.

A Datasets

Table 2 provides a summary along with statistics of all MTEB tasks. In the following, we give a brief description of each dataset included in MTEB.

A.1 Clustering

ArxivClusteringS2S, ArxivClusteringP2P, BiorxivClusteringS2S, BiorxivClusteringP2P, MedrxivClusteringP2P, MedrxivClusteringS2S These datasets are custom-made for MTEB using the public APIs from arXiv[9] and bioRxiv/medRxiv[10]. For S2S datasets, the input text is simply the title of the paper, while for P2P the input text is the concatenation of the title and the abstract. The cluster labels are generated using categories given to the papers by humans. For bioRxiv and medRxiv this category is unique, but for arXiv multiple categories can be given to a single paper so we only use the first one. For bioRxiv and medRxiv there is only one level of category (e.g. biochemistry, genetics, microbiology, etc.) hence we only perform clustering based on that label. For arXiv there is a main category and secondary category: for example "cs.AI" means the main category is Computer Science and the sub-category is AI, math.AG means the main category is Mathematics and the sub-category is Algrebraic Geometry etc. Hence, we create three types of splits:

(a) Main category clustering Articles are only clustered based on the main category (Math, Physics, Computer Science etc.). This split evaluates coarse clustering capacity of a model.

(b) Secondary category clustering within the same main category Articles are clustered based on their secondary category, but within a given main category, for example only Math papers that need to be clustered into Algebraic Geometry, Functional Analysis, Numerical Analysis etc. This split evaluates fine-grained clustering capacity of a model, as differentiating some sub-categories can be very difficult.

(c) Secondary category clustering Articles are clustered based on their secondary category for all main categories, so the labels can be Number Theory, Computational Complexity, Astrophysics of Galaxies etc. These splits evaluate fine-grained clustering capacity, as well as multi-scale capacities i.e. is a model able to both separate Maths from Physics as well as Probability from Algebraic Topology at the same time.

For every dataset, split and strategy, we select subsets of all labels and then sample articles from those labels. This yields splits with a varying amount and size of clusters.

RedditClustering (Geigle et al., 2021): Clustering of titles from 199 subreddits. Clustering of 25 splits, each with 10-50 classes, and each class with 100 - 1000 sentences

RedditClusteringP2P Dataset created for MTEB using available data from Reddit posts[11]. The task consists of clustering the concatenation of title+post according to their subreddit. It contains 10 splits, with 10 and 100 clusters per split and 1,000 to 100,000 posts.

StackExchangeClustering (Geigle et al., 2021) Clustering of titles from 121 stackexchanges. Clustering of 25 splits, each with 10-50 classes, and each class with 100-1000 sentences.

StackExchangeClusteringP2P Dataset created for MTEB using available data from StackExchange posts[12]. The task consists of clustering the concatenation of title and post according to their subreddit. It contains 10 splits, with 10 to 100 clusters and 5,000 to 10,000 posts per split.

TwentyNewsgroupsClustering[13] Clustering of the 20 Newsgroups dataset, given titles of article the goal is to find the newsgroup (20 in total). Contains 10 splits, each with 20 classes, with each split containing between 1,000 and 10,000 titles.

A.2 Classification

AmazonCounterfactual (O'Neill et al., 2021) A collection of Amazon customer reviews annotated for counterfactual detection pair classification. For each review the label is either "counterfactual" or "not-counterfactual". This is a multilingual dataset with 4 available languages.

---

[9] `https://arxiv.org/help/api/`

[10] `https://api.biorxiv.org/`

[11] `https://huggingface.co/datasets/sentence-transformers/reddit-title-body`

[12] `https://huggingface.co/datasets/flax-sentence-embeddings/stackexchange_title_body_jsonl`

[13] `https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html`

Name	Type	Categ.	#Lang.	Train Samples	Dev Samples	Test Samples	Train avg. chars	Dev avg. chars	Test avg. chars
BUCC	BitextMining	s2s	4	0	0	641684	0	0	101.3
Tatoeba	BitextMining	s2s	112	0	0	2000	0	0	39.4
AmazonCounterfactualClassification	Classification	s2s	4	4018	335	670	107.3	109.2	106.1
AmazonPolarityClassification	Classification	p2p	1	3600000	0	400000	431.6	0	431.4
AmazonReviewsClassification	Classification	s2s	6	1200000	30000	30000	160.5	159.2	160.4
Banking77Classification	Classification	s2s	1	10003	0	3080	59.5	0	54.2
EmotionClassification	Classification	s2s	1	16000	2000	2000	96.8	95.3	96.6
ImdbClassification	Classification	p2p	1	25000	0	25000	1325.1	0	1293.8
MassiveIntentClassification	Classification	s2s	51	11514	2033	2974	35.0	34.8	34.6
MassiveScenarioClassification	Classification	s2s	51	11514	2033	2974	35.0	34.8	34.6
MTOPDomainClassification	Classification	s2s	6	15667	2235	4386	36.6	36.5	36.8
MTOPIntentClassification	Classification	s2s	6	15667	2235	4386	36.6	36.5	36.8
ToxicConversationsClassification	Classification	s2s	1	50000	0	50000	298.8	0	296.6
TweetSentimentExtractionClassification	Classification	s2s	1	27481	0	3534	68.3	0	67.8
ArxivClusteringP2P	Clustering	p2p	1	0	0	732723	0	0	1009.9
ArxivClusteringS2S	Clustering	s2s	1	0	0	732723	0	0	74.0
BiorxivClusteringP2P	Clustering	p2p	1	0	0	75000	0	0	1666.2
BiorxivClusteringS2S	Clustering	s2s	1	0	0	75000	0	0	101.6
MedrxivClusteringP2P	Clustering	p2p	1	0	0	37500	0	0	1981.2
MedrxivClusteringS2S	Clustering	s2s	1	0	0	37500	0	0	114.7
RedditClustering	Clustering	s2s	1	0	420464	420464	0	64.7	64.7
RedditClusteringP2P	Clustering	p2p	1	0	0	459399	0	0	727.7
StackExchangeClustering	Clustering	s2s	1	0	417060	373850	0	56.8	57.0
StackExchangeClusteringP2P	Clustering	p2p	1	0	0	75000	0	0	1090.7
TwentyNewsgroupsClustering	Clustering	s2s	1	0	0	59545	0	0	32.0
SprintDuplicateQuestions	PairClassification	s2s	1	0	101000	101000	0	65.2	67.9
TwitterSemEval2015	PairClassification	s2s	1	0	0	16777	0	0	38.3
TwitterURLCorpus	PairClassification	s2s	1	0	0	51534	0	0	79.5
AskUbuntuDupQuestions	Reranking	s2s	1	0	0	2255	0	0	52.5
MindSmallReranking	Reranking	s2s	1	231530	0	107968	69.0	0	70.9
SciDocsRR	Reranking	s2s	1	0	19594	19599	0	69.4	69.0
StackOverflowDupQuestions	Reranking	s2s	1	23018	3467	3467	49.6	49.8	49.8
ArguAna	Retrieval	p2p	1	0	0	10080	0	0	1052.9
ClimateFEVER	Retrieval	s2p	1	0	0	5418128	0	0	539.1
CQADupstackAndroidRetrieval	Retrieval	s2p	1	0	0	23697	0	0	578.7
CQADupstackEnglishRetrieval	Retrieval	s2p	1	0	0	41791	0	0	467.1
CQADupstackGamingRetrieval	Retrieval	s2p	1	0	0	46896	0	0	474.7
CQADupstackGisRetrieval	Retrieval	s2p	1	0	0	38522	0	0	991.1
CQADupstackMathematicaRetrieval	Retrieval	s2p	1	0	0	17509	0	0	1103.7
CQADupstackPhysicsRetrieval	Retrieval	s2p	1	0	0	39355	0	0	799.4
CQADupstackProgrammersRetrieval	Retrieval	s2p	1	0	0	33052	0	0	1030.2
CQADupstackStatsRetrieval	Retrieval	s2p	1	0	0	42921	0	0	1041.0
CQADupstackTexRetrieval	Retrieval	s2p	1	0	0	71090	0	0	1246.9
CQADupstackUnixRetrieval	Retrieval	s2p	1	0	0	48454	0	0	984.7
CQADupstackWebmastersRetrieval	Retrieval	s2p	1	0	0	17911	0	0	689.8
CQADupstackWordpressRetrieval	Retrieval	s2p	1	0	0	49146	0	0	1111.9
DBPedia	Retrieval	s2p	1	0	4635989	4636322	0	310.2	310.1
FEVER	Retrieval	s2p	1	0	0	5423234	0	0	538.6
FiQA2018	Retrieval	s2p	1	0	0	58286	0	0	760.4
HotpotQA	Retrieval	s2p	1	0	0	5240734	0	0	288.6
MSMARCO	Retrieval	s2p	1	0	8848803	8841866	0	336.6	336.8
MSMARCOv2	Retrieval	s2p	1	138641342	138368101	0	341.4	342.0	0
NFCorpus	Retrieval	s2p	1	0	0	3956	0	0	1462.7
NQ	Retrieval	s2p	1	0	0	2684920	0	0	492.7
QuoraRetrieval	Retrieval	s2s	1	0	0	532931	0	0	62.9
SCIDOCS	Retrieval	s2p	1	0	0	26657	0	0	1161.9
SciFact	Retrieval	s2p	1	0	0	5483	0	0	1422.3
Touche2020	Retrieval	s2p	1	0	0	382594	0	0	1720.1
TRECCOVID	Retrieval	s2p	1	0	0	171382	0	0	1117.4
BIOSSES	STS	s2s	1	200	200	200	156.6	156.6	156.6
SICK-R	STS	s2s	1	19854	19854	19854	46.1	46.1	46.1
STS12	STS	s2s	1	4468	0	6216	100.7	0	64.7
STS13	STS	s2s	1	0	0	3000	0	0	54.0
STS14	STS	s2s	1	0	0	7500	0	0	54.3
STS15	STS	s2s	1	0	0	6000	0	0	57.7
STS16	STS	s2s	1	0	0	2372	0	0	65.3
STS17	STS	s2s	11	0	0	500	0	0	43.3
STS22	STS	p2p	18	0	0	8060	0	0	1992.8
STSBenchmark	STS	s2s	1	11498	3000	2758	57.6	64.0	53.6
SummEval	Summarization	p2p	1	0	0	2800	0	0	359.8

Table 2: Tasks in MTEB

AmazonPolarity (McAuley and Leskovec, 2013) A collection of Amazon customer reviews annotated for polarity classification. For each review the label is either "positive" or "negative".

AmazonReviews (McAuley and Leskovec, 2013) A collection of Amazon reviews designed to aid research in multilingual text classification. For each review the label is the score given by the review between 0 and 4 (1-5 stars). This is a multilingual dataset with 6 available languages.

Banking77 (Casanueva et al., 2020) Dataset composed of online banking queries annotated with their corresponding intents. For each user query the label is an intent among 77 intents like ’activate_my_card’, ’apple_pay’, ’bank_transfer’, etc.

Emotion (Saravia et al., 2018) Dataset of English Twitter messages with six basic emotions: anger, fear, joy, love, sadness, and surprise.

Imdb (Maas et al., 2011) Large movie review dataset with labels being positive or negative.

MassiveIntent (FitzGerald et al., 2022) A collection of Amazon Alexa virtual assistant utterances annotated with the associated intent. For each user utterance the label is one of 60 intents like ’play_music’, ’alarm_set’, etc. This is a multilingual dataset with 51 available languages.

MassiveScenario (FitzGerald et al., 2022) A collection of Amazon Alexa virtual assistant utterances annotated with the associated intent. For each user utterance the label is a theme among 60 scenarios like ’music’, ’weather’, etc. This is a multilingual dataset with 51 available languages.

MTOPDomain / MTOPIntent Multilingual sentence datasets from the MTOP (Li et al., 2020) benchmark. We refer to their paper for details.

ToxicConversations Dataset from Kaggle competition[14]. Collection of comments from the Civil Comments platform together with annotations if the comment is toxic or not.

TweetSentimentExtraction Dataset from Kaggle competition[15]. Sentiment classification of tweets as neutral, positive or negative.

A.3 Pair Classification

SprintDuplicateQuestions (Shah et al., 2018): Collection of questions from the Sprint community. The goal is to classify a pair of sentences as duplicates or not.

TwitterSemEval2015 (Xu et al., 2015) Paraphrase-Pairs of Tweets from the SemEval 2015 workshop. The goal is to classify a pair of tweets as paraphrases or not.

TwitterURLCorpus (Lan et al., 2017) Paraphrase-Pairs of Tweets. The goal is to classify a pair of tweets as paraphrases or not.

A.4 Bitext Mining

BUCC (Zweigenbaum et al., 2016, 2017, 2018) BUCC provides big set of sentences ($\sim$ 10-70k each) for English, French, Russian, German and Chinese, along with associated pairs annotation. The annotated pairs here corresponds to a pairs of translated sentences, i.e. a sentence and its translation in the other language.

Tatoeba (Research) Tatoeba provides sets of sentences (1000 sentences each) for 112 languages with annoated associated pairs. Each pair is one sentence and its translation in another language.

A.5 Reranking

AskUbuntuDupQuestions[16] Questions from AskUbuntu with manual annotations marking pairs of questions as similar or dissimilar.

MindSmall (Wu et al., 2020) Large-scale English Dataset for News Recommendation Research. Ranking news article titles given the title of a news article. The idea is to recommend other news from the one you are reading.

SciDocsRR (Cohan et al., 2020b) Ranking of related scientific papers based on their title.

StackOverflowDupQuestions (Liu et al., 2018) Stack Overflow Duplicate Questions Task for questions with the tags Java, JavaScript and Python, ranking questions as duplicates or not.

A.6 Semantic Textual Similarity (STS)

STS12, STS13, STS14, STS15, STS16, STS17, STS22, STSBenchmark (Agirre et al., 2012, 2013)[17][18][19][20] Original STS benchmark, with scores from 0 to 5. The selection of sentences includes text from image captions, news headlines and user forums. In total they contain between 1,000 and 20,000 sentences. STS12 - STS16 and

---

[14] https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/overview

[15] https://www.kaggle.com/competitions/tweet-sentiment-extraction/overview

[16] https://github.com/taolei87/askubuntu

[17] https://alt.qcri.org/semeval2014/task10/

[18] https://alt.qcri.org/semeval2015/task2/

[19] https://alt.qcri.org/semeval2016/task1/

[20] https://competitions.codalab.org/competitions/33835

STSBenchmark are monolingual english benchmarks. STS17 and STS22 contain crosslingual pairs of sentences, where the goal is to assess the similarity of two sentences in different languages. STS17 has 11 language pairs (among Korean, Arabic, English, French, German, Turkish, Spanish, Italian and Dutch) and STS22 has 18 language pairs (among Arabic, English, French, German, Turkish, Spanish, Polish, Italian, Russian and Chinese).

BIOSSES[21] Contains 100 sentence pairs from the biomedical field.

SICK-R (Agirre et al., 2014) Sentences Involving Compositional Knowledge (SICK) contains a large number of sentence pairs (10 0000) that are lexically, syntactically and semantically rich.

A.7 Summarization

SummEval (Fabbri et al., 2020) Summaries generated by recent summarization models trained on CNN or DailyMail alongside human annotations.

A.8 Retrieval

We refer to the BEIR paper (Thakur et al., 2021), which contains description of each dataset. For MTEB, we include all publicly available datasets: **ArguAna, ClimateFEVER, CQADupstack, DBPedia, FEVER, FiQA2018, HotpotQA, MSMARCO, NFCorpus, NQ, Quora, SCIDOCS, SciFact, Touche2020, TRECCOVID**.

B Examples

Tables 3-9 provide examples for each dataset for each task. For retrieval datasets, we refer to the BEIR paper (Thakur et al., 2021).

C Correlations

Figure 6 provides correlation heatmaps for model performance and MTEB tasks.

D Models

Table 10 provides publicly available model checkpoints used for MTEB evaluation.

E Additional results

Tables 11 until the end provide results on individual datasets of MTEB. The results are additionally available in json format on the Hugging Face Hub[22] and can be inspected on the leaderboard[23].

---

[21] https://tabilab.cmpe.boun.edu.tr/BIOSSES/DataSet.html

[22] https://huggingface.co/datasets/mteb/results

[23] https://huggingface.co/spaces/mteb/leaderboard

Dataset	Text	Label
AmazonCounterfactualClassification	In person it looks as though it would have cost a lot more.	counterfactual
AmazonPolarityClassification	an absolute masterpiece I am quite sure any of you actually taking the time to read this have played the game at least once, and heard at least a few of the tracks here. And whether you were aware of it or not, Mitsuda's music contributed greatly to the...	positive
AmazonReviewsClassification	solo llega una unidad cuando te obligan a comprar dos Te obligan a comprar dos unidades y te llega solo una y no hay forma de reclamar, una autentica estafa, no compreis!!	0
Banking77Classification	What currencies is an exchange rate calculated in?	exchange_rate
EmotionClassification	i feel so inhibited in someone elses kitchen like im painting on someone elses picture	sadness
ImdbClassification	When I first saw a glimpse of this movie, I quickly noticed the actress who was playing the role of Lucille Ball. Rachel York's portrayal of Lucy is absolutely awful. Lucille Ball was an astounding comedian with incredible talent. To think about a legend like Lucille Ball being portrayed the way she was in the movie is horrendous. I cannot believe...	negative
MassiveIntentClassification	réveille-moi à neuf heures du matin le vendredi	alarm_set
MassiveScenarioClassification	tell me the artist of this song	music
MTOPDomainClassification	Maricopa County weather forecast for this week	weather
MTOPIntentClassification	what ingredients do is have left	GET_INFO_RECIPES
ToxicConversationsClassification	The guy's a damn cop, so what do you expect?	toxic
TweetSentimentExtractionClassification	I really really like the song Love Story by Taylor Swift	positive

Table 3: Classification examples

Dataset	Text	Cluster
ArxivClusteringP2P	Finite groups of rank two which do not involve $Qd(p)$. Let $p > 3$ be a prime. We show that if G is a finite group with p-rank equal to 2, then G involves $Qd(p)$ if and only if G p'-involves $Qd(p)$. This allows us to use a version of Glauberman's ZJ-theorem to give a more direct construction of finite group actions on mod-p homotopy spheres. We give an example to illustrate that the above conclusion does not hold for $p \leq 3$.	math
ArxivClusteringS2S	Vertical shift and simultaneous Diophantine approximation on polynomial curves	math
BiorxivClusteringP2P	Innate Immune sensing of Influenza A viral RNA through IFI16 promotes pyroptotic cell death Programmed cell death pathways are triggered by various stresses or stimuli, including viral infections. The mechanism underlying the regulation of these pathways upon Influenza A virus IAV infection is not well characterized. We report that a cytosolic DNA sensor IFI16 is...	immunology
BiorxivClusteringS2S	Association of CDH11 with ASD revealed by matched-gene co-expression analysis and mouse behavioral	neuroscience
MedrxivClusteringP2P	Temporal trends in the incidence of haemophagocytic lymphohistiocytosis: a nationwide cohort study from England 2003-2018. Haemophagocytic lymphohistiocytosis (HLH) is rare, results in high mortality and is increasingly being diagnosed. Little is known about what is driving the apparent rise in the incidence of this disease. Using national linked electronic health data from hospital admissions and death certification cases of HLH that were diagnosed in England between 1/1/2003 and 31/12/2018 were identified using a previously validated approach. We calculated incidence...	infectious diseases
MedrxivClusteringS2S	Current and Lifetime Somatic Symptom Burden Among Transition-aged Young Adults on the Autism Spectrum	psychiatry and clinical psychology
RedditClustering	Could anyone tell me what breed my bicolor kitten is?	r/cats
RedditClusteringP2P	Headaches after working out? Hey guys! I've been diagnosed with adhd since I was seven. I just recently got rediagnosed (22f) and I've been out on a different medication, adderall I was normally taking vyvanse but because of cost and no insurance adderall was more affordable. I've noticed that if I take adderall and workout...	r/ADHD
StackExchangeClustering	Does this property characterize a space as Hausdorff?	math.stackexchange.com
StackExchangeClusteringP2P	Google play services error DEBUG: Application is pausing, which disconnects the RTMP client. I am having this issue from past day with Google Play Services Unity. What happens is, when I install app directly ot device via Unity, the Google Play Services work fine but when I upload it as beta to play store console and install it via that then it starts to give " DEBUG: Application is pausing, which disconnects the RTMP client" error. I have a proper SHA1 key.	unity
TwentyNewsgroupsClustering	Commercial mining activities on the moon	14

Table 4: Clustering examples

Dataset	Sentence 1	Sentence 2	Label
SprintDuplicateQuestions	Franklin U722 USB modem signal strength	How do I know if my Franklin U772 USB Modem has a weak signal ?	1
TwitterSemEval2015	All the home alones watching 8 mile","All the home alones watching 8 mile	The last rap battle in 8 Mile nevr gets old ahah	0
TwitterURLCorpus	How the metaphors we use to describe discovery affect men and women in the sciences	Light Bulbs or Seeds ? How Metaphors for Ideas Influence Judgments About Genius	0

Table 5: Pair classification examples. Labels are binary.

Dataset	Query	Positive	Negative
AskUbuntuDupQuestions	change the application icon theme but not changing the panel icons	change folder icons in ubuntu-mono-dark theme	change steam tray icon back to default
MindSmallReranking	Man accused in probe of Giuliani associates is freed on bail	Studies show these are the best and worst states for your retirement	There are 14 cheap days to fly left in 2019: When are they and what deals can you score?
SciDocsRR	Discovering social circles in ego networks	Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities.	Improving www proxies performance with greedy-dual-size-frequency caching policy
StackOverflowDupQuestions	Java launch error selection does not contain a main type	Error: Selection does not contain a main type	Selection Sort in Java

Table 6: Reranking examples

Dataset	Sentence 1	Sentence 2	Score
BIOSSES	It has recently been shown that Craf is essential for Kras G12D-induced NSCLC.	It has recently become evident that Craf is essential for the onset of Kras-driven non-small cell lung cancer.	4.0
SICK-R	A group of children is playing in the house and there is no man standing in the background	A group of kids is playing in a yard and an old man is standing in the background	3.2
STS12	Nationally, the federal Centers for Disease Control and Prevention recorded 4,156 cases of West Nile, including 284 deaths.	There were 293 human cases of West Nile in Indiana in 2002, including 11 deaths statewide.	1.7
STS13	this frame has to do with people (the residents) residing in locations , sometimes with a co-resident .	inhabit or live in ; be an inhabitant of ;	2.8
STS14	then the captain was gone.	then the captain came back.	0.8
STS15	you 'll need to check the particular policies of each publisher to see what is allowed and what is not allowed.	if you need to publish the book and you have found one publisher that allows it.	3.0
STS16	you do not need to worry.	you don 't have to worry.	5.0
STS17	La gente muestra su afecto el uno por el otro.	A women giving something to other lady.	1.4
STS22	El secretario general de la Asociación Gremial de los Trabajadores del Subte y Premetro de Metrodelegados, Beto Pianelli, dijo que el Gobierno porteño debe convocar “inmediatamente” a licitación para la compra de nuevos trenes y retirar los que quedan en circulación...	En diálogo con el servicio informativo de la Radio Pública, el ministro de Salud de la Nación, Ginés González García, habló sobre el avance del coronavirus en la Argentina y se manifestó a favor de prorrogar la cuarentena obligatoria dispuesta por...	1
STSBenchmark	A man is playing the cello.	A man seated is playing the cello.	4.25

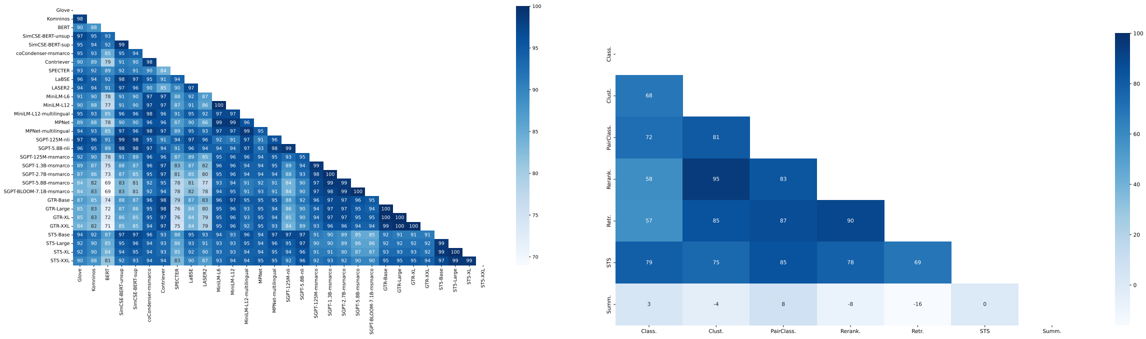
Table 7: STS examples. Scores are continuous between 0 and 5 (included).

Dataset	First set sentence	Second set sentence
BUCC	Morales remporte l'élection présidentielle de 2005 à la majorité absolue.	Morales went on to win the 2005 presidential election with an absolute majority.
Tatoeba	Chi le ha detto che Tom l'ha fatto?	Who told you that Tom did that?

Table 8: Bitext mining examples

Dataset	Human Summary	Machine Summary	Relevance
SummEval	V. Stiviano must pay back $2.6 million in gifts from Donald Sterling. Sterling's wife claimed the ex-Clippers used the couple's money for the gifts. The items included a Ferrari, two Bentleys and a Range Rover.	donald sterling , nba team last year . sterling 's wife sued for $ 2.6 million in gifts . sterling says he is the former female companion who has lost the . sterling has ordered v. stiviano to pay back $ 2.6 m in gifts after his wife sued . sterling also includes a $ 391 easter bunny costume , $ 299 and a $ 299 .	1.7

Table 9: Summarization example



(a) Model correlation based on all results

(b) Task correlation based on average task results

Figure 6: Pearson correlations across model and task results. **Left:** Size variants of the same architecture show high correlations. **Right:** Performance on clustering and reranking correlates strongest, while summarization and classification show weaker correlation with other tasks.

Model	**Public Checkpoint**
Glove	https://huggingface.co/sentence-transformers/average_word_embeddings_glove.6B.300d
Komninos	https://huggingface.co/sentence-transformers/average_word_embeddings_komninos
BERT	https://huggingface.co/bert-base-uncased
SimCSE-BERT-unsup	https://huggingface.co/princeton-nlp/unsup-simcse-bert-base-uncased
SimCSE-BERT-sup	https://huggingface.co/princeton-nlp/sup-simcse-bert-base-uncased
coCondenser-msmarco	https://huggingface.co/sentence-transformers/msmarco-bert-co-condensor
Contriever	https://huggingface.co/nthakur/contriever-base-msmarco
SPECTER	https://huggingface.co/sentence-transformers/allenai-specter
LaBSE	https://huggingface.co/sentence-transformers/LaBSE
LASER2	https://github.com/facebookresearch/LASER
MiniLM-L6	https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
MiniLM-L12	https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2
MiniLM-L12-multilingual	https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
MPNet	https://huggingface.co/sentence-transformers/all-mpnet-base-v2
MPNet-multilingual	https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2
MiniLM-L12-multilingual	https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
SGPT-125M-nli	https://huggingface.co/Muennighoff/SGPT-125M-weightedmean-nli-bitfit
SGPT-5.8B-nli	https://huggingface.co/Muennighoff/SGPT-5.8B-weightedmean-nli-bitfit
SGPT-125M-msmarco	https://huggingface.co/Muennighoff/SGPT-125M-weightedmean-msmarco-specb-bitfit
SGPT-1.3B-msmarco	https://huggingface.co/Muennighoff/SGPT-1.3B-weightedmean-msmarco-specb-bitfit
SGPT-2.7B-msmarco	https://huggingface.co/Muennighoff/SGPT-2.7B-weightedmean-msmarco-specb-bitfit
SGPT-5.8B-msmarco	https://huggingface.co/Muennighoff/SGPT-5.8B-weightedmean-msmarco-specb-bitfit
SGPT-BLOOM-7.1B-msmarco	https://huggingface.co/bigscience/sgpt-bloom-7b1-msmarco
SGPT-BLOOM-1.7B-nli	https://huggingface.co/bigscience-data/sgpt-bloom-1b7-nli
GTR-Base	https://huggingface.co/sentence-transformers/gtr-t5-base
GTR-Large	https://huggingface.co/sentence-transformers/gtr-t5-large
GTR-XL	https://huggingface.co/sentence-transformers/gtr-t5-xl
GTR-XXL	https://huggingface.co/sentence-transformers/gtr-t5-xxl
ST5-Base	https://huggingface.co/sentence-transformers/sentence-t5-base
ST5-Large	https://huggingface.co/sentence-transformers/sentence-t5-large
ST5-XL	https://huggingface.co/sentence-transformers/sentence-t5-xl
ST5-XXL	https://huggingface.co/sentence-transformers/sentence-t5-xxl

Table 10: Publicly available model links used for evaluation

Dataset	Glove	Komninos	BERT	SimCSE-BERT-unsup	SimCSE-BERT-sup	coCondenser-msmarco	Contr-iever	SPECTER	LaBSE	LASER2	MiniLM-L6	MiniLM-L12-	MiniLM-L12-multilingual	MPNet	MPNet-multilingual	Ada Similarity	SGPT-125M-nli	SGPT-5.8B-nli	SGPT-125M-msmarco	SGPT-1.3B-msmarco	SGPT-2.7B-msmarco	SGPT-5.8B-msmarco	SGPT-BLOOM-7.1B-msmarco	GTR-Base	GTR-Large	GTR-XL	GTR-XXL	ST5-Base	ST5-Large	ST5-XL	ST5-XXL
AmazonCounterfactualClassification	56.91	60.54	74.25	67.09	75.75	64.06	72.19	58.70	75.93	76.84	64.15	65.28	71.57	65.27	75.81	76.40	65.88	74.07	61.24	65.21	67.57	69.22	68.06	69.33	70.03	68.60	67.30	75.82	75.51	76.01	77.07
AmazonPolarityClassification	60.32	59.59	71.33	74.48	82.47	66.88	68.63	57.77	68.95	61.01	62.58	62.98	69.21	67.13	76.41	92.83	74.94	82.31	65.40	73.21	71.44	71.26	68.97	67.82	73.92	74.58	75.05	85.12	92.87	93.17	92.79
AmazonReviewsClassification	29.67	31.01	33.56	33.85	39.60	34.85	37.42	26.26	35.80	28.71	31.79	30.79	35.11	31.92	38.51	47.45	35.10	41.58	31.17	34.96	35.75	39.19	33.86	38.48	37.21	38.20	37.30	44.94	47.12	48.18	48.93
Banking77Classification	67.69	67.05	63.41	73.55	75.76	82.35	80.02	66.66	69.85	57.76	79.75	80.40	79.77	81.86	81.07	68.04	74.68	81.74	77.70	82.06	83.22	84.49	84.33	79.26	81.21	82.22	82.32	76.48	78.46	80.88	82.31
EmotionClassification	36.93	33.18	35.28	42.22	44.81	41.91	44.77	24.82	37.22	24.83	38.43	41.17	42.37	39.73	45.84	50.32	42.23	49.92	39.08	46.39	49.21	49.66	44.87	42.20	46.32	45.55	43.19	51.36	51.73	51.95	48.57
ImdbClassification	62.57	63.98	65.35	69.63	73.53	60.17	67.04	56.35	62.04	57.58	60.66	59.76	60.46	70.72	64.57	89.38	62.90	74.33	58.67	64.05	63.53	66.64	61.77	65.99	70.86	68.15	70.8	77.34	87.01	87.54	90.23
MassiveIntentClassification	56.19	57.21	59.88	59.84	65.95	70.40	67.78	51.73	61.46	47.91	67.40	67.15	66.84	69.57	69.32	65.17	58.08	70.0	61.41	68.65	69.01	70.39	69.67	67.05	70.06	70.23	70.61	69.74	71.78	72.09	73.44
MassiveScenarioClassification	66.03	66.11	64.28	66.25	70.78	73.73	76.00	58.58	66.41	55.92	75.76	74.58	71.51	76.01	75.35	67.67	66.34	75.03	69.74	76.04	75.90	76.28	75.34	75.40	75.49	75.94	77.77	72.32	73.16	73.26	74.82
MTOPDomainClassification	79.11	78.57	82.63	81.71	84.29	91.34	93.18	74.53	86.06	75.36	91.56	91.90	87.06	92.08	89.24	89.89	81.52	89.64	86.96	92.08	92.56	93.47	93.68	92.42	94.01	93.60	93.84	90.34	90.99	90.73	92.49
MTOPIntentClassification	55.85	57.07	68.14	59.23	63.14	71.07	69.31	50.05	63.03	49.47	62.18	62.84	65.52	70.21	68.69	64.80	58.24	70.68	62.25	71.19	71.85	72.42	71.34	62.44	63.86	65.93	67.71	63.32	64.98	68.15	68.33
ToxicConversationsClassification	65.40	67.76	70.0	68.82	72.04	64.01	67.77	57.44	66.90	54.05	66.99	67.47	66.07	60.86	71.02	70.00	62.79	69.93	62.66	68.73	68.84	67.71	66.55	66.60	68.65	67.56	68.48	68.20	71.73	70.95	70.04
TweetSentimentExtractionClassification	50.80	49.68	51.81	53.36	59.73	55.74	56.10	45.52	58.82	48.73	55.41	54.25	56.12	55.46	59.03	63.35	54.82	62.44	52.41	55.67	56.69	56.85	55.85	56.02	54.09	54.77	54.54	62.71	62.33	61.21	62.01
ArxivClusteringP2P	32.56	34.73	35.19	32.61	35.18	36.94	42.61	44.75	32.13	17.77	46.55	46.07	38.33	48.38	37.78	41.49	34.74	40.55	39.71	43.38	44.72	45.59	44.59	35.49	37.50	37.90	37.90	39.28	41.62	41.62	42.89
ArxivClusteringS2S	23.14	26.01	27.51	24.68	27.54	29.03	32.32	35.27	22.05	12.39	37.86	37.50	31.55	39.72	31.68	28.47	24.68	32.49	28.24	33.71	35.08	38.86	38.03	27.18	30.55	30.45	32.39	27.26	29.44	31.17	33.47
BiorxivClusteringP2P	29.27	29.76	30.12	24.90	30.15	32.35	34.97	39.52	29.84	12.40	38.48	36.99	33.49	39.62	33.09	36.86	28.93	33.59	33.63	35.06	34.41	36.55	36.03	27.66	29.59	30.52	30.48	33.99	35.99	36.43	36.53
BiorxivClusteringS2S	19.18	20.71	24.77	19.55	24.67	28.16	29.08	34.53	20.57	8.83	33.17	33.21	29.44	35.02	29.60	27.55	23.08	29.13	27.04	30.71	30.53	33.70	32.48	23.25	25.72	26.06	27.50	22.92	24.02	26.47	28.66
MedrxivClusteringP2P	26.12	26.65	26.09	23.60	26.25	30.23	31.19	35.04	30.13	17.91	34.41	34.25	31.52	35.58	31.96	31.09	28.30	30.33	31.37	32.08	31.35	31.51	31.05	27.57	28.72	28.69	29.12	33.20	32.40	32.30	32.09
MedrxivClusteringS2S	20.38	21.50	23.60	21.97	24.12	27.01	27.27	31.66	24.82	16.63	32.29	32.24	30.87	32.87	31.70	26.50	24.93	28.02	26.87	29.45	28.77	28.76	29.26	25.13	27.39	26.69	27.56	26.13	26.33	26.93	26.82
RedditClustering	28.46	28.84	27.24	32.18	40.23	48.04	54.89	24.13	28.79	9.96	50.67	51.18	42.02	54.82	45.24	42.47	33.76	42.17	40.23	48.23	46.47	40.45	35.53	56.13	61.69	61.34	64.13	52.93	54.53	57.03	58.99
RedditClusteringP2P	35.82	7.37	43.32	45.14	47.74	53.53	57.58	35.06	49.14	26.42	54.15	54.80	50.73	56.77	51.31	58.10	41.01	48.02	49.09	53.18	54.17	55.75	54.52	58.53	61.67	61.11	62.84	59.67	62.50	62.34	64.46
StackExchangeClustering	35.80	39.04	43.58	43.07	47.55	59.54	63.15	39.01	35.43	15.79	53.36	53.05	49.60	53.80	52.98	53.52	44.59	54.13	52.74	60.86	59.19	59.21	55.13	64.21	69.93	69.95	71.43	63.13	65.11	67.13	70.78
StackExchangeClusteringP2P	28.51	30.23	26.55	28.50	29.45	30.48	32.25	31.46	28.83	18.63	38.00	33.13	31.69	34.28	32.94	30.43	28.23	31.12	32.66	32.36	32.57	33.95	34.31	33.01	33.21	32.73	32.85	35.68	36.86	34.79	35.25
TwentyNewsgroupsClustering	25.83	27.42	23.35	23.21	34.86	38.68	46.82	24.22	23.28	11.38	46.86	47.47	39.28	49.74	44.10	36.26	28.24	37.20	32.13	40.06	40.89	39.46	37.28	46.72	51.64	51.15	50.44	48.10	49.33	49.53	50.93
SprintDuplicateQuestions	86.96	85.55	36.81	69.41	69.39	96.09	95.55	71.63	89.26	65.54	94.55	92.45	89.46	90.15	90.55	77.85	77.73	80.54	89.89	92.58	93.47	93.84	94.93	94.55	95.05	95.45	95.68	91.23	89.01	91.44	88.89
TwitterSemEval2015	48.45	53.85	55.90	60.21	67.75	65.95	66.85	43.25	62.78	59.57	67.86	70.02	62.06	73.85	66.75	69.04	57.09	66.00	54.75	62.37	63.68	66.87	65.31	72.23	76.03	77.81	77.54	78.25	79.75	80.89	80.28
TwitterURLCorpus	77.35	79.41	76.29	81.37	83.89	83.17	85.21	69.22	84.58	81.47	84.70	84.77	83.83	85.11	85.14	83.69	80.51	84.54	81.06	83.79	84.80	85.29	85.46	84.77	84.89	85.14	85.13	86.05	86.14	85.86	86.01
AskUbuntuDupQuestions	49.57	50.88	45.84	51.57	51.80	58.99	56.69	50.07	52.75	48.99	63.48	64.06	60.49	65.85	60.16	53.49	52.63	55.90	55.84	58.13	59.63	61.63	59.97	60.86	61.64	63.08	63.23	59.73	61.51	62.86	66.16
MindSmallReranking	27.01	28.92	28.37	28.62	29.30	27.13	31.58	24.80	29.81	24.79	30.80	31.02	30.37	30.97	30.15	30.71	29.27	31.11	30.40	31.34	31.72	32.29	31.79	31.33	31.84	31.50	31.93	30.20	30.27	29.77	30.60
SciDocsRR	62.56	63.55	64.94	66.33	70.14	72.78	76.51	81.31	68.72	54.99	87.12	87.20	77.78	88.65	78.09	71.04	68.36	77.54	71.34	77.21	77.72	80.79	79.77	73.71	76.39	76.49	77.96	73.96	74.88	75.16	76.09
StackOverflowDupQuestions	34.03	35.65	34.62	39.35	38.90	48.48	47.78	36.22	42.42	36.98	50.76	51.47	45.85	51.98	46.79	40.85	39.97	44.77	44.74	49.32	49.61	51.53	51.07	51.01	51.58	52.79	53.50	48.46	49.34	51.05	52.85
ArguAna	36.30	30.96	28.29	38.34	38.33	45.15	48.32	32.67	34.18	12.86	50.17	47.13	44.88	46.52	48.91		31.04	35.07	45.42	49.68	50.49	51.38	47.28	50.83	52.09	52.81	53.77	44.85	39.27	39.40	39.85
ClimateFEVER	14.44	14.87	5.41	11.80	11.98	16.96	24.79	6.86	3.83	0.36	20.27	21.57	18.49	21.97	15.27		11.01	17.57	21.86	26.6	27.11	30.46	29.39	24.88	26.90	27.01	27.21	10.37	11.36	10.61	14.63
CQADupstackRetrieval	15.47	16.79	5.51	13.22	14.50	27.72	33.67	14.60	18.75	4.12	41.32	42.53	30.71	44.96	31.32		20.29	29.98	27.25	33.33	36.53	39.40	39.62	34.55	36.62	37.35	38.56	35.23	38.96	40.78	44.65
DBPedia	18.29	15.88	4.13	15.04	19.73	27.86	38.10	4.14	15.57	1.53	32.33	33.36	22.63	32.09	26.22		10.87	26.10	22.72	31.51	34.70	39.87	39.03	35.24	39.55	39.74	41.28	27.77	31.55	33.65	39.19
FEVER	14.99	15.56	3.30	21.05	20.41	45.68	59.29	5.45	12.17	0.77	51.93	55.91	52.66	50.86	56.76		18.40	38.64	60.45	68.12	72.73	78.24	73.97	68.93	72.66	72.18	74.08	26.16	36.21	36.12	51.20
FiQA2018	10.09	10.49	2.19	9.84	10.41	15.62	27.42	5.64	7.00	1.73	36.87	37.27	20.33	49.96	22.96		8.94	18.59	21.12	29.99	33.29	37.20	35.84	35.15	42.79	44.19	46.78	34.83	43.55	44.71	46.68
HotpotQA	19.18	20.77	8.26	19.75	22.89	35.61	56.81	5.46	18.75	5.50	46.51	44.59	30.01	39.29	37.03		17.73	33.99	40.88	49.93	52.84	59.26	57.26	54.93	57.85	58.91	59.67	33.20	33.95	37.17	42.14
MSMARCO	9.60	9.75	1.91	9.35	11.00	29.57	36.77	5.58	7.60	1.09	36.54	39.03	23.72	39.75	26.60		6.27	15.83	27.98	36.05	38.83	39.91	41.12	41.16	42.73	43.52	44.05	20.71	23.96	25.17	27.68
NFCorpus	13.87	11.79	4.30	9.88	12.42	22.29	31.31	0.84	16.54	2.44	31.59	32.25	23.45	33.29	25.49		11.80	28.26	22.79	32.08	33.89	36.21	35.78	30.22	32.63	33.34	34.18	28.64	31.10	33.18	35.08
NQ	12.87	12.75	2.61	11.69	16.08	29.85	41.83	5.99	8.42	0.64	43.87	46.47	29.80	50.45	33.60		7.63	24.63	29.73	42.94	46.70	52.41	53.15	50.47	55.09	56.16	57.24	36.32	42.02	46.29	52.87
QuoraRetrieval	71.32	71.58	61.03	78.03	79.62	86.51	86.72	64.65	77.03	71.14	87.56	87.75	86.55	87.46	86.41		78.96	84.68	72.98	85.28	85.60	84.58	74.71	87.98	88.47	88.91	89.09	85.49	85.73	85.85	85.96
SCIDOCS	8.04	8.47	2.81	5.50	7.53	10.13	17.12	0.00	5.63	0.78	21.64	21.82	0.03	23.77	13.96		7.13	13.55	12.21	16.18	16.57	19.87	18.62	14.00	15.51	15.71	15.88	14.16	15.38	15.97	17.17
SciFact	29.58	29.53	13.34	25.72	29.59	52.31	65.51	47.88	38.20	4.04	64.51	62.64	48.37	65.57	50.30		31.79	46.66	56.90	68.29	70.17	74.70	72.11	59.74	63.42	64.20	66.77	45.76	49.91	50.91	55.38
Touche2020	13.99	13.17	0.97	8.90	9.89	8.57	15.79	8.46	4.88	1.06	16.90	17.22	16.06	19.93	17.40		12.27	16.18	22.97	24.45	23.44	25.43	23.98	25.89	28.29	25.26	26.76	20.30	21.63	22.51	21.65
TRECCOVID	36.22	35.92	14.74	26.2	22.93	40.54	44.77	29.91	16.34	10.97	47.25	50.82	39.12	51.33	37.87		39.31	55.35	70.30	72.98	75.17	84.88	81.37	56.05	56.68	60.09	51.90	40.70	46.11	54.77	59.48
BIOSSES	44.93	50.25	54.70	72.31	68.38	77.32	83.32	64.95	78.70	62.01	81.64	83.57	74.18	80.43	76.27	78.04	70.93	79.50	75.21	83.02	84.84	86.25	85.31	79.00	84.86	78.94	81.91	75.89	78.93	73.12	80.43
SICK-R	55.43	55.49	58.65	72.24	80.77	72.00	70.20	56.39	69.99	62.86	77.58	79.32	79.61	80.59	79.62	77.48	74.57	79.59	65.93	67.23	68.20	69.63	69.82	71.45	73.39	73.63	74.29	80.18	80.34	79.98	80.47
STS12	54.64	53.51	30.87	66.05	75.30	68.19	64.34	62.49	65.08	62.60	72.37	73.08	76.02	72.63	77.90	72.30	69.17	74.29	66.53	66.59	66.99	67.50	69.66	68.59	70.33	69.11	70.12	78.05	79.11	79.02	78.85
STS13	69.16	70.80	59.89	81.49	84.67	80.40	80.03	58.70	67.98	59.62	80.60	82.13	80.70	83.48	85.11	81.49	77.23	85.35	76.17	77.33	77.58	79.16	79.67	79.09	82.19	81.82	82.72	85.85	87.33	88.80	88.94
STS14	60.81	63.56	47.73	73.61	80.19	74.02	74.51	54.87	64.03	57.03	75.59	76.73	78.85	78.00	80.81	74.74	70.99	79.21	69.05	71.83	72.78	74.46	74.61	74.64	77.16	77.07	78.24	82.19	83.17	84.33	84.86
STS15	72.31	74.08	60.29	79.72	85.40	82.57	83.30	62.54	76.59	71.57	85.39	85.58	85.84	85.66	87.48	84.28	79.74	85.52	79.24	80.66	82.62	84.47	83.81	84.85	86.31	86.01	86.26	87.46	88.28	88.89	89.32
STS16	65.34	64.60	63.73	78.12	80.82	79.78	79.67	64.27	72.98	70.75	78.99	80.23	81.05	80.03	83.20	82.06	77.93	82.54	76.07	78.91	80.10	80.96	80.40	81.57	81.85	82.23	81.61	84.03	84.36	85.31	84.67
STS17	77.95	76.91	64.10	83.58	89.44	85.94	86.32	69.63	79.45	76.73	87.59	88.63	86.87	90.60	86.99	87.08	87.33	90.44	84.95	86.99	87.25	87.78	87.07	85.80	83.93	84.90	85.18	89.57	88.99	88.91	89.46
STS22	56.35	53.89	56.37	59.65	61.96	67.54	64.64	55.06	60.97	39.75	67.21	65.67	61.72	67.95	63.06	64.71	59.64	63.20	65.66	67.30	68.75	69.35	66.13	66.17	64.30	66.61	65.76	62.66	62.39	64.32	65.33
STSBenchmark	61.54	61.55	47.29	76.52	84.25	76.97	78.81	61.26	72.25	69.77	82.03	83.09	84.42	83.42	86.82	83.78	79.54	85.67	75.34	77.59	79.21	81.39	80.90	79.58	77.60	77.65	77.73	85.52	85.36	83.93	84.01
SummEval	28.87	30.49	29.82	31.15	23.31	29.50	30.36	27.66	31.05	26.8	30.81	27.9	30.67	27.49	31.57	26.94	30.26	30.38	28.90	25.44	27.87	24.75	24.99	29.67	29.50	30.21	30.64	31.39	29.64	29.91	30.08
Average	41.97	42.06	38.33	45.45	48.72	52.35	56.00	40.28	45.21	34.95	56.26	56.53	52.44	57.78	54.71		45.97	53.74	51.23	56.11	57.12	58.81	57.44	56.19	58.28	58.42	58.97	55.27	57.06	57.87	59.51

Table 11: All English results. The main score for each task is reported as described in Section 3.2.

Dataset	Language	LASER2	LaBSE	MiniLM-L12-multilingual	MPNet-multilingual	SGPT-BLOOM-7.1B-msmarco
BUCC	de-en	99.21	99.35	97.11	98.59	54.00
BUCC	fr-en	98.39	98.72	94.99	96.89	97.06
BUCC	ru-en	97.62	97.78	95.06	96.44	45.30
BUCC	zh-en	97.70	99.16	95.63	97.56	97.96
Tatoeba	sqi-eng	97.22	96.76	98.17	98.57	10.38
Tatoeba	fry-eng	42.07	89.31	31.13	43.54	24.62
Tatoeba	kur-eng	19.09	83.59	46.94	61.44	8.26
Tatoeba	tur-eng	98.03	98.00	95.08	96.17	6.15
Tatoeba	deu-eng	99.07	99.20	97.02	97.73	70.10
Tatoeba	nld-eng	95.35	96.07	94.58	95.50	29.74
Tatoeba	ron-eng	96.52	96.92	95.30	96.43	27.23
Tatoeba	ang-eng	25.22	59.28	10.24	16.72	28.76
Tatoeba	ido-eng	80.86	89.42	40.25	43.91	43.91
Tatoeba	jav-eng	9.95	79.77	17.04	23.39	15.02
Tatoeba	isl-eng	94.32	94.75	24.07	59.25	6.29
Tatoeba	slv-eng	95.40	96.03	96.92	97.08	10.14
Tatoeba	cym-eng	5.85	92.00	13.25	22.31	6.97
Tatoeba	kaz-eng	53.30	87.49	34.89	61.49	3.32
Tatoeba	est-eng	96.43	96.55	97.33	98.40	4.76
Tatoeba	heb-eng	0.00	91.53	86.88	88.26	1.69
Tatoeba	gla-eng	1.52	85.66	3.61	4.72	2.09
Tatoeba	mar-eng	92.93	92.65	92.38	93.83	45.53
Tatoeba	lat-eng	64.81	80.07	19.47	24.25	28.76
Tatoeba	bel-eng	79.54	95.00	67.73	79.94	8.03
Tatoeba	pms-eng	36.23	64.57	30.70	34.19	31.94
Tatoeba	gle-eng	4.20	93.80	11.62	16.85	3.26
Tatoeba	pes-eng	93.13	94.70	92.59	93.47	12.13
Tatoeba	nob-eng	95.77	98.40	97.73	98.53	21.07
Tatoeba	bul-eng	93.57	94.58	92.65	93.52	20.09
Tatoeba	cbk-eng	77.17	79.44	55.37	58.68	64.63
Tatoeba	hun-eng	95.20	96.55	91.58	94.18	5.07
Tatoeba	uig-eng	56.49	92.40	24.39	48.35	1.27
Tatoeba	rus-eng	92.58	93.75	91.87	92.92	59.84
Tatoeba	spa-eng	97.33	98.40	95.42	97.00	94.48
Tatoeba	hye-eng	88.72	94.09	93.28	94.38	0.50
Tatoeba	tel-eng	96.72	97.86	36.40	79.73	64.62
Tatoeba	afr-eng	92.59	96.18	58.22	72.96	16.62
Tatoeba	mon-eng	3.42	95.91	95.04	96.14	2.85
Tatoeba	arz-eng	66.16	76.00	51.26	55.69	70.66
Tatoeba	hrv-eng	96.72	96.95	95.98	97.00	12.79
Tatoeba	nov-eng	60.02	74.38	47.99	50.23	52.23
Tatoeba	gsw-eng	27.52	46.50	25.74	25.12	21.03
Tatoeba	nds-eng	77.13	79.42	32.16	38.88	23.92
Tatoeba	ukr-eng	93.52	93.97	92.82	92.67	22.06
Tatoeba	uzb-eng	23.20	84.23	17.14	23.19	4.71
Tatoeba	lit-eng	96.20	96.47	93.16	95.37	4.49
Tatoeba	ina-eng	93.93	95.37	79.13	84.32	73.67
Tatoeba	lfn-eng	63.39	67.54	47.02	49.56	44.85
Tatoeba	zsm-eng	95.41	95.62	95.31	95.80	79.95
Tatoeba	ita-eng	94.32	92.72	93.05	93.76	65.04
Tatoeba	cmn-eng	85.62	95.10	94.93	95.83	91.45
Tatoeba	lvs-eng	95.33	95.88	97.87	97.53	6.55
Tatoeba	glg-eng	96.14	96.82	94.00	95.32	79.86
Tatoeba	ceb-eng	9.93	64.42	8.05	7.39	6.64
Tatoeba	bre-eng	31.2	15.07	5.56	6.42	4.67
Tatoeba	ben-eng	89.43	88.55	36.48	64.90	75.98
Tatoeba	swg-eng	33.10	59.36	26.31	22.80	16.89
Tatoeba	arq-eng	26.63	42.69	18.60	19.84	27.75
Tatoeba	kab-eng	65.88	4.31	1.16	1.41	1.69
Tatoeba	fra-eng	94.28	94.86	91.72	93.12	91.44
Tatoeba	por-eng	94.54	94.14	92.13	93.02	92.62
Tatoeba	tat-eng	34.74	85.92	10.25	10.89	3.59
Tatoeba	oci-eng	58.13	65.81	38.57	43.49	40.17
Tatoeba	pol-eng	97.32	97.22	94.28	96.95	14.09
Tatoeba	war-eng	8.25	60.29	7.25	7.42	10.38
Tatoeba	aze-eng	82.41	94.93	62.10	76.36	6.32
Tatoeba	vie-eng	96.73	97.20	95.12	97.23	94.20
Tatoeba	nno-eng	72.75	94.48	76.34	81.41	16.28
Tatoeba	cha-eng	14.86	31.77	15.98	12.59	23.26
Tatoeba	mhr-eng	6.86	15.74	6.89	7.57	1.56
Tatoeba	dan-eng	95.22	95.71	94.80	96.17	23.52
Tatoeba	ell-eng	96.20	95.35	95.43	94.93	5.34
Tatoeba	amh-eng	80.82	91.47	36.21	53.49	0.03
Tatoeba	pam-eng	3.24	10.73	5.41	5.39	5.85
Tatoeba	hsb-eng	45.75	67.11	36.10	44.32	9.68
Tatoeba	srp-eng	93.64	94.43	92.24	94.12	11.69
Tatoeba	epo-eng	96.61	98.20	41.73	55.12	26.20
Tatoeba	kzj-eng	4.46	11.33	6.24	5.88	5.17
Tatoeba	awa-eng	33.74	71.70	33.43	42.83	35.01
Tatoeba	fao-eng	57.04	87.40	27.51	38.24	12.61
Tatoeba	mal-eng	98.16	98.45	32.20	88.46	83.30
Tatoeba	ile-eng	87.88	85.58	57.71	60.36	59.59
Tatoeba	bos-eng	95.86	94.92	93.27	94.02	13.65
Tatoeba	cor-eng	4.45	10.11	3.42	3.53	2.83
Tatoeba	cat-eng	95.80	95.38	94.42	96.05	88.31
Tatoeba	eus-eng	93.32	95.01	23.18	31.33	53.38
Tatoeba	yue-eng	87.75	89.58	71.45	77.58	77.03
Tatoeba	swe-eng	95.31	95.63	94.42	95.45	19.53
Tatoeba	dtp-eng	7.39	10.85	5.69	5.03	3.41
Tatoeba	kat-eng	81.16	95.02	95.44	95.46	0.42
Tatoeba	jpn-eng	93.78	95.38	90.41	92.51	71.36
Tatoeba	csb-eng	27.03	52.57	21.56	23.73	10.03
Tatoeba	xho-eng	4.68	91.55	4.52	6.53	5.51
Tatoeba	orv-eng	23.24	38.93	15.10	23.77	5.79
Tatoeba	ind-eng	92.98	93.66	92.74	93.50	88.04
Tatoeba	tuk-eng	16.35	75.27	15.16	14.91	5.48
Tatoeba	max-eng	36.96	63.26	45.25	48.77	36.14
Tatoeba	swh-eng	55.66	84.50	14.48	16.02	16.74
Tatoeba	hin-eng	95.32	96.87	97.62	97.75	85.23
Tatoeba	dsb-eng	42.34	64.81	33.43	36.85	8.78
Tatoeba	ber-eng	77.63	8.40	4.43	4.88	4.92
Tatoeba	tam-eng	87.32	89.0	24.64	73.60	72.76
Tatoeba	slk-eng	95.82	96.5	95.15	96.62	9.98
Tatoeba	tgl-eng	63.19	96.02	13.09	17.67	10.70
Tatoeba	ast-eng	76.35	90.68	62.17	70.08	71.13
Tatoeba	mkd-eng	93.63	93.6	91.00	93.02	10.47
Tatoeba	khm-eng	74.19	78.37	32.11	58.80	0.37
Tatoeba	ces-eng	95.52	96.68	95.12	95.73	9.55
Tatoeba	tzl-eng	36.56	58.88	25.46	34.21	27.82
Tatoeba	urd-eng	84.23	93.22	94.57	95.12	70.10
Tatoeba	ara-eng	90.14	88.80	87.93	90.19	85.37
Tatoeba	kor-eng	87.97	90.95	92.52	93.07	22.39
Tatoeba	yid-eng	2.49	88.79	14.38	30.73	0.16
Tatoeba	fin-eng	96.98	96.37	93.10	95.92	3.41
Tatoeba	tha-eng	96.38	96.14	96.72	95.99	2.22
Tatoeba	wuu-eng	75.09	90.18	76.00	78.25	79.58
Average	mix	67.42	81.75	57.98	63.38	31.08

Table 12: Multilingual bitext mining results. Scores are f1.

Dataset	Language	LASER2	LaBSE	MiniLM-L12-multilingual	MPNet-multilingual	SGPT-BLOOM-7.1B-msmarco
AmazonCounterfactualClassification	de	67.82	73.17	68.35	69.95	61.35
AmazonCounterfactualClassification	ja	68.76	76.42	63.45	69.79	58.23
AmazonReviewsClassification	de	31.07	39.92	35.91	39.52	29.70
AmazonReviewsClassification	es	32.72	39.39	37.49	39.99	35.97
AmazonReviewsClassification	fr	31.12	38.52	35.30	39.00	35.92
AmazonReviewsClassification	ja	28.94	36.44	33.24	36.64	27.64
AmazonReviewsClassification	zh	30.89	36.45	35.26	37.74	32.63
MassiveIntentClassification	af	38.01	56.12	45.88	52.32	47.85
MassiveIntentClassification	am	12.70	55.71	36.75	41.55	33.30
MassiveIntentClassification	ar	37.16	50.86	45.14	51.43	59.25
MassiveIntentClassification	az	19.98	58.97	47.42	56.98	45.24
MassiveIntentClassification	bn	42.51	58.22	35.34	48.79	61.59
MassiveIntentClassification	cy	17.33	50.16	26.12	27.87	44.92
MassiveIntentClassification	da	45.61	58.25	57.73	62.77	51.23
MassiveIntentClassification	de	44.79	56.21	50.71	59.57	56.10
MassiveIntentClassification	el	46.71	57.03	58.70	62.62	46.13
MassiveIntentClassification	es	45.44	58.32	59.66	64.43	66.35
MassiveIntentClassification	fa	45.01	62.33	61.02	65.34	51.20
MassiveIntentClassification	fi	45.94	60.12	57.54	62.28	45.33
MassiveIntentClassification	fr	46.13	60.47	60.25	64.82	66.95
MassiveIntentClassification	he	42.55	56.55	52.51	58.21	43.18
MassiveIntentClassification	hi	40.20	59.40	58.37	62.77	63.54
MassiveIntentClassification	hu	42.77	59.52	60.41	63.87	44.73
MassiveIntentClassification	hy	28.07	56.20	51.60	57.74	38.13
MassiveIntentClassification	id	45.81	61.12	59.85	65.43	64.06
MassiveIntentClassification	is	39.86	54.90	30.83	37.05	44.35
MassiveIntentClassification	it	48.25	59.83	59.61	64.68	60.77
MassiveIntentClassification	ja	45.30	63.11	60.89	63.74	61.22
MassiveIntentClassification	jv	24.30	50.98	32.37	36.49	50.94
MassiveIntentClassification	ka	22.70	48.35	43.03	49.85	33.84
MassiveIntentClassification	km	22.48	48.55	40.04	45.47	37.34
MassiveIntentClassification	kn	4.32	56.24	40.98	50.63	53.54
MassiveIntentClassification	ko	44.26	60.99	50.30	61.82	53.36
MassiveIntentClassification	lv	39.75	57.10	54.68	61.29	46.50
MassiveIntentClassification	ml	41.33	57.91	42.41	54.34	58.27
MassiveIntentClassification	mn	16.20	58.50	51.77	56.59	40.28
MassiveIntentClassification	ms	43.23	58.60	54.76	60.70	59.65
MassiveIntentClassification	my	25.37	57.35	52.01	57.09	37.42
MassiveIntentClassification	nb	37.74	57.91	55.50	62.60	49.41
MassiveIntentClassification	nl	45.00	59.37	59.51	63.57	52.09
MassiveIntentClassification	pl	44.99	59.71	59.43	64.30	50.48
MassiveIntentClassification	pt	48.55	60.16	61.27	64.89	66.69
MassiveIntentClassification	ro	44.30	57.92	58.39	62.80	50.53
MassiveIntentClassification	ru	44.29	60.67	59.04	63.26	58.32
MassiveIntentClassification	sl	44.72	59.37	57.36	63.51	47.74
MassiveIntentClassification	sq	46.12	58.03	56.59	62.49	48.94
MassiveIntentClassification	sv	45.95	59.66	59.43	64.73	50.79
MassiveIntentClassification	sw	31.89	51.62	29.57	31.95	49.81
MassiveIntentClassification	ta	29.63	55.04	36.77	50.17	56.40
MassiveIntentClassification	te	36.03	58.32	40.72	52.82	54.71
MassiveIntentClassification	th	43.39	56.58	58.97	61.11	44.43
MassiveIntentClassification	tl	29.73	55.28	33.67	38.83	50.21
MassiveIntentClassification	tr	43.93	60.91	59.90	64.54	46.56
MassiveIntentClassification	ur	26.11	56.70	52.80	56.37	56.75
MassiveIntentClassification	vi	44.33	56.67	56.61	59.68	64.53
MassiveIntentClassification	zh-CN	40.62	63.86	61.99	65.33	67.07
MassiveIntentClassification	zh-TW	32.93	59.51	58.77	62.35	62.89
MassiveScenarioClassification	af	47.10	63.39	53.64	59.67	51.47
MassiveScenarioClassification	am	17.70	62.02	41.89	48.97	34.87
MassiveScenarioClassification	ar	45.21	57.72	51.74	57.78	65.21
MassiveScenarioClassification	az	28.21	63.48	52.06	61.53	45.58
MassiveScenarioClassification	bn	50.52	61.84	41.17	54.53	67.30
MassiveScenarioClassification	cy	22.58	56.13	31.72	35.26	46.29
MassiveScenarioClassification	da	54.87	65.24	66.87	71.00	53.52
MassiveScenarioClassification	de	54.34	62.39	57.40	67.34	61.74
MassiveScenarioClassification	el	55.47	64.58	66.14	68.81	48.96
MassiveScenarioClassification	es	52.77	63.61	65.04	70.42	73.34
MassiveScenarioClassification	fa	52.50	67.46	65.86	69.88	53.17
MassiveScenarioClassification	fi	52.63	64.58	63.75	67.60	44.69
MassiveScenarioClassification	fr	54.32	65.10	66.06	70.69	72.91
MassiveScenarioClassification	he	52.41	63.53	59.20	65.16	43.10
MassiveScenarioClassification	hi	47.37	64.40	65.21	67.92	69.27
MassiveScenarioClassification	hu	53.43	65.82	66.56	70.30	45.16
MassiveScenarioClassification	hy	33.57	61.25	56.11	63.02	38.73
MassiveScenarioClassification	id	54.38	65.84	66.16	70.73	70.13
MassiveScenarioClassification	is	49.78	61.94	37.52	44.16	44.21
MassiveScenarioClassification	it	54.84	64.09	65.00	69.73	65.57
MassiveScenarioClassification	ja	54.12	67.72	66.50	69.69	65.76
MassiveScenarioClassification	jv	32.71	58.29	38.60	44.20	54.79
MassiveScenarioClassification	ka	26.92	53.38	50.66	57.30	32.99
MassiveScenarioClassification	km	27.23	56.18	46.96	53.14	39.34
MassiveScenarioClassification	kn	10.06	61.74	45.73	56.08	60.50
MassiveScenarioClassification	ko	52.01	67.26	55.66	68.52	55.69
MassiveScenarioClassification	lv	44.82	61.87	59.80	66.28	44.35
MassiveScenarioClassification	ml	49.10	62.26	47.69	60.13	65.53
MassiveScenarioClassification	mn	21.51	62.60	57.07	60.85	38.72
MassiveScenarioClassification	ms	53.60	65.63	61.71	65.81	64.99
MassiveScenarioClassification	my	29.72	62.94	59.10	63.03	36.84
MassiveScenarioClassification	nb	43.90	64.29	64.25	70.24	51.80
MassiveScenarioClassification	nl	53.33	65.16	65.52	70.37	56.32
MassiveScenarioClassification	pl	52.92	64.56	65.04	68.99	49.98
MassiveScenarioClassification	pt	53.41	63.28	65.79	70.09	71.46
MassiveScenarioClassification	ro	50.48	62.41	64.17	67.95	53.69
MassiveScenarioClassification	ru	51.84	65.25	65.24	69.92	61.60
MassiveScenarioClassification	sl	51.29	64.25	64.01	70.81	48.04
MassiveScenarioClassification	sq	55.65	64.54	64.31	69.63	50.06
MassiveScenarioClassification	sv	54.64	66.01	67.14	71.60	51.73
MassiveScenarioClassification	sw	42.04	58.36	34.86	37.29	54.22
MassiveScenarioClassification	ta	36.72	59.08	42.62	55.96	62.77
MassiveScenarioClassification	te	42.08	64.13	46.46	58.81	62.59
MassiveScenarioClassification	th	52.15	64.34	67.01	69.44	45.18
MassiveScenarioClassification	tl	37.34	60.23	37.37	43.99	52.06
MassiveScenarioClassification	tr	52.56	65.43	66.55	70.4	47.21
MassiveScenarioClassification	ur	32.60	61.52	60.43	62.9	64.26
MassiveScenarioClassification	vi	50.97	61.05	60.72	65.71	70.61
MassiveScenarioClassification	zh-CN	50.22	70.85	67.44	71.23	73.95
MassiveScenarioClassification	zh-TW	42.32	67.08	65.70	68.73	70.30
MTOPDomainClassification	de	74.08	86.95	79.20	85.73	82.05
MTOPDomainClassification	es	73.47	84.07	83.04	86.96	93.55
MTOPDomainClassification	fr	72.26	84.14	78.63	81.21	90.98
MTOPDomainClassification	hi	72.95	85.11	81.36	84.76	89.33
MTOPDomainClassification	th	72.68	81.24	79.99	82.51	60.49
MTOPIntentClassification	de	51.62	63.42	54.23	61.27	61.92
MTOPIntentClassification	es	52.75	64.44	60.28	66.59	74.49
MTOPIntentClassification	fr	50.12	62.01	54.05	59.76	69.12
MTOPIntentClassification	hi	45.55	62.58	59.90	62.37	64.85
MTOPIntentClassification	th	50.07	64.61	61.96	64.80	49.36
Average	mix	42.85	60.77	54.87	60.39	54.4

Table 13: Multilingual classification results. Scores are accuracy.

Dataset	Language	Komninos	LASER2	LaBSE	MiniLM-L12-multilingual	MPNet-multilingual	SGPT-BLOOM-7.1B-msmarco
STS17	ko-ko	2.54	70.52	71.32	77.03	83.41	66.89
STS17	ar-ar	13.78	67.47	69.07	79.16	79.10	76.42
STS17	en-ar	9.08	65.05	74.51	81.22	80.85	78.07
STS17	en-de	-3.11	66.66	73.85	84.22	83.28	59.10
STS17	en-tr	-0.45	70.05	72.07	76.74	74.90	11.80
STS17	es-en	-8.18	55.30	65.71	84.44	86.11	78.22
STS17	es-es	48.23	79.67	80.83	85.56	85.14	86.00
STS17	fr-en	5.81	70.82	76.98	76.59	81.17	80.46
STS17	it-en	3.64	70.98	76.99	82.35	84.24	51.58
STS17	nl-en	-0.44	68.12	75.22	81.71	82.51	45.85
STS22	de	33.04	25.69	48.58	44.64	46.70	30.05
STS22	es	48.53	54.92	63.18	56.56	59.91	65.41
STS22	pl	12.47	18.34	39.30	33.74	33.65	31.13
STS22	tr	47.38	36.97	58.15	53.39	56.30	47.14
STS22	ar	32.42	42.57	57.67	46.2	52.19	58.67
STS22	ru	19.44	39.24	57.49	57.08	58.74	43.36
STS22	zh	4.78	49.41	63.02	58.75	61.75	66.78
STS22	fr	49.43	58.61	77.95	70.55	74.30	80.38
STS22	de-en	28.65	32.35	50.14	52.65	50.81	51.16
STS22	es-en	26.97	54.34	71.86	67.33	70.26	75.06
STS22	it	57.77	60.31	72.22	55.22	60.65	65.65
STS22	pl-en	45.55	53.63	69.41	69.02	73.07	53.31
STS22	zh-en	14.05	46.19	64.02	65.71	67.96	68.45
STS22	es-it	41.10	42.21	69.69	47.67	53.70	65.50
STS22	de-fr	14.77	37.41	53.28	51.73	62.34	53.28
STS22	de-pl	11.21	15.67	58.69	44.22	40.53	43.05
STS22	fr-pl	39.44	39.44	61.98	50.71	84.52	28.17
Average	mix	22.14	51.55	65.67	64.23	67.71	57.81

Table 14: Multilingual STS Results. Scores are Spearman correlations of cosine similarities.

Step by Step Loss Goes Very Far: Multi-Step Quantization for Adversarial Text Attacks

Piotr Gaiński
Jagiellonian University
Ardigen
piotr.gainski@doctoral.uj.edu.pl

Klaudia Bałazy
Jagiellonian University
klaudia.balazy@doctoral.uj.edu.pl

Abstract

We propose a novel gradient-based attack against transformer-based language models that searches for an adversarial example in a continuous space of token probabilities. Our algorithm mitigates the gap between adversarial loss for continuous and discrete text representations by performing multi-step quantization in a quantization-compensation loop. Experiments show that our method significantly outperforms other approaches on various natural language processing (NLP) tasks.

1 Introduction

Deep neural networks achieve impressive results, but their vulnerability to adversarial attacks causes major security threats and is a concern when interpreting or explaining model predictions.

In computer vision, the most successful attack methods use gradient-based optimization techniques (Carlini and Wagner, 2017; Madry et al., 2018). They minimize adversarial loss function that encourages the prediction error and imperceptibility of a generated example.

Development of optimization-based attacks in NLP is much more challenging due to the discrete nature of text. Recent methods (Guo et al., 2021; Yuan et al., 2021) overcome this limitation by performing a gradient descent in the continuous space of token representations and eventually quantizing them into discrete text.

A quantization of a token can significantly change its embedding and cause an undesired change of the loss value, degrading the adversarial example. To our knowledge, all existing optimization-based NLP attacks quantize all tokens in a text at once, which creates a considerable gap between adversarial loss for continuous and discrete text representations.

In this paper, we propose MANGO[1] (Multi-step quANtization Gradient-based adversarial Optimizer): a novel optimization-based attack against Transformer (Vaswani et al., 2017) language models that mitigates the aforementioned gap by performing multi-step quantization in a quantization-compensation loop. MANGO quantizes continuous token representations one by one and reoptimizes the adversarial example after each quantization to compensate undesired degradation of adversarial loss value. The construction of MANGO introduces interesting problems that are addressed in Section 3. MANGO achieves superior performance in various NLP tasks, outperforming recent white-box (optimization-based) and black-box attacks.

2 Related Work

Adversarial attacks can be roughly divided into two categories: white-box attacks that have access to the internal model's states (e.g. gradient) and more common black-box attacks that only know outputs of the model. In our paper, we focus on a white-box version of our MANGO attack. In Appendix D, we develop a version of MANGO that can be used in the loosened black-box setting.

Black-Box Methods Most black-box NLP attacks define a space of character or word replacements and heuristically search it for an adversarial example (Yoo et al., 2020). The search space is limited with semantic ad hoc constraints (e.g. limiting edit distance or restricting possible replacements to synonyms) to preserve the attack's imperceptibility. Such constraints disallow some specific perturbations (e.g. replacing a word with its antagonist even if the semantics is preserved in the context of other perturbations) and tend to generate semantically incorrect examples (Morris et al., 2020a).

White-Box Methods Many white-box methods use gradients to guide a heuristic search in a space of text perturbations (Ebrahimi et al., 2018; Cheng et al., 2019; Xu and Du, 2020). Recent methods take a step further and perform gradient descent

---

[1] Code available at github.com/gmum/MANGO.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2038–2048
May 2-6, 2023

optimization. They aim to find an example that minimizes the adversarial loss function, which encourages the prediction error and the imperceptibility of the attack. Because the similarity and fluency of an example are controlled by a powerful external model used in the loss, optimization-based methods do not require hand-crafted semantic constraints, making them more flexible than black-box ones.

Adapting gradient descent in NLP attacks is a challenging problem due to the discrete nature of the optimized text. Yuan et al. (2021) overcome this issue by performing optimization in the continuous space of token embeddings and replacing each token with a possibly new token, which embedding is the closest to the optimized one. An alternative approach is the GBDA method (Guo et al., 2021) that optimizes a continuous distribution of stochastic one-hot vectors and repeatedly samples adversarial examples from the optimized distribution until it fools the attacked model.

Quantization Both methods mentioned above quantize all continuous representations of tokens to a text at once. Quantization of a single token may significantly change its embedding and cause an undesirable change of adversarial loss value. When quantizing all tokens at once, the changes accumulate to a considerable gap between adversarial loss for continuous and discrete text representations (see Section 6). Our MANGO mitigates this gap.

3 MANGO

This section describes our MANGO method. Unlike other optimization-based methods that quantize all token representations at once, MANGO constitutes an entirely new algorithm that quantizes a token and compensates for the resulting change in an adversarial loss value in a step-by-step manner. The construction of MANGO introduces interesting problems that are addressed in the **Optimization**, **Vector Selection** and **Candidates Selection** paragraphs and are further evaluated in Section 5.

Continuous Token Representation The first learnable layer of Transformer takes as input a sequence of tokens $x = (t_1, ..., t_n)$, where $t_i \in 2^{|V|}$ has a single non-zero binary value at index k indicating that it represents the k-th token in vocabulary V.

Similarly to Guo et al. (2021), we relax the input sequence x and replace one-hot encodings t_i with probability vectors π_i. Because the first learnable Transformer layer is a simple linear layer, it can take probability vectors as input without any modification.

A probability vector π_i constitutes probability distribution over tokens from V.

In the embedding layer, the Transformer embeds probability vectors with the function e:

$$e(\pi_i) = \sum_{j=1}^{|V|} (\pi_i)_j E_j, \tag{1}$$

where E_j is the embedding vector of the j-th token. If π_i is quantized, meaning it is a one-hot vector representing some token k, function e simply looks up the k-th embedding: $e(\pi_i) = E_k$. In MANGO, π_i is a probabilistic vector, and its embedding $e(\pi_i)$ is a mixture of embeddings of all tokens weighted by their probabilities π_i. We parameterize π_i with logits Θ_i and a standard softmax function σ, so that $\pi_i = \sigma(\Theta_i)$ and $x = \sigma(\Theta)$ for $\Theta = (\Theta_1, ..., \Theta_n)$.

Loss function Let $m : X \rightarrow \mathbb{R}^{|Y|}$ be a classifier that outputs logit vectors and properly predicts a label $y \in Y$ for some datapoint $x \in X$, meaning that $\arg\max_k m(x)_k = y$. An adversarial example is a sample $x' \in X$ that is imperceptible (according to specified criteria) from x but changes the output of the model. In an optimization-based setting, searching for an adversarial example is usually defined as a minimization of an adversarial loss function.

Following Guo et al. (2021), we compose our adversarial loss $\mathcal{L}$ as a combination of margin loss l_m, fluency loss l_f, and similarity loss l_s:

$$\mathcal{L}(x') = l_m(m, x', y) + \lambda_f l_f(g, x') + \lambda_s l_s(g, x', x), \tag{2}$$

where λ_f and λ_s are the coefficients used to balance the losses and g is a reference model.

Margin loss l_m encourages model m to missclassify x' by a margin κ:

$$l_m(m, x', y) = \max(m(x')_y - \max_{k \neq y} m(x')_k + \kappa, 0).$$

Fluency loss l_f promotes x' with a high probability of being generated by a causal language model g that predicts the next token distribution:

$$l_f(g, x') = -\sum_{i=1}^{n} \sum_{j=1}^{|V|} (\pi_i)_j g(\pi_1, ..., \pi_{i-1})_j.$$

Similarity loss l_s is based on BERTScore (Zhang et al., 2020) and captures the semantic similarity between x and x' using contextualized embeddings of tokens $\phi_g(x) = (v_1, ..., v_n)$ and $\phi_g(x') =$

$(v'_1, ..., v'_n)$ produced by the reference model g :

$$l_s(g, x', x) = -\sum_{i=1}^{n} w_i \max_j v_i^T v'_j,$$

where w_i is the inverse frequency of token t_i.

Quantization-Compensation Loop MANGO algorithm searches for a x' that minimizes $\mathcal{L}$, quantizing and compensating it step by step. Algorithm 1 introduces the idea of MANGO.

In the first line, the parameters Θ of x' are initialized, so that $\Theta'_{ij} = C \cdot (x_i)_j$ for some constant C. Each loop starts with **optimization** of x' with respect to $\mathcal{L}$. Then **vector selection** is performed to select π'_i from x' which will be quantized in the current step. Given π'_i, MANGO performs **candidates selection** and selects m the most promising tokens $c_1, ..., c_m$ to which x'_i can be quantized. In the 6th line, each candidate c_j is evaluated by computing $\mathcal{L}$ for a sequence x' with vector π'_i quantized to c_j. Finally, π'_i is quantized to the best c_j chosen from the previous step. Quantized π'_i will no longer be updated during optimization. MANGO repeats lines 2-7 until all vectors in x' are quantized.

Algorithm 1: MANGO

```
Data: adversarial loss $\mathcal{L}$ (eq. 2)
Result: sentence $x'$ that minimizes $\mathcal{L}$
1 initialize $x' = (\pi'_1, ..., \pi'_n)$
2 while $x'$ is not fully quantized do
3     optimization: optimize parameters of $x'$
4     vector selection: select probabilistic
        vector $\pi'_i$ from $x'$ for quantization
5     candidates selection: select $m$ tokens
        candidates from $\pi'_i$
6     evaluate these $m$ candidates with loss $\mathcal{L}$
7     quantize $\pi'_i$ to best evaluated token
```

Optimization We optimize x' with the Adam optimizer (Kingma and Ba, 2014) which is reset after each quantization (see Section 5). This allows x' to rapidly change its trajectory to compensate for the degradation of $\mathcal{L}$. The initial number of optimization steps is S, but it decreases by a factor of 2 in each loop to reduce computational costs.

Vector Selection In line 4th, we choose vector π'_i with the highest entropy (see Section 5), because its quantization will introduce the most significant change to x' and is likely to increase the loss value the most. Intuitively, we want such degrading quantizations to occur early in the algorithm, because the more vectors are not quantized yet, the larger capacity x' has to compensate for degradation by finding another local minimum of $\mathcal{L}$.

Candidates Selection In this phase, we select m tokens that can be used to quantize the probability vector π'_i with possibly a small degradation of $\mathcal{L}$. Quantization of π'_i with token k is a step $q_k = (-(\pi'_i)_1, -(\pi'_i)_2, ..., 1 - (\pi'_i)_k, ..., -(\pi'_i)_n)$ in the π'_i space. As π'_i is likely to be in the proximity of its local minimum with respect to $\mathcal{L}$, we want the step q_k to have (1) the lowest norm $\|q_k\|$ possible and (2) follow the direction of the local (minus) gradient. We use this intuition in the formulation of the token score s_k, which is a weighted mean of the probability $(\pi'_i)_k$ and the direction score d_k:

$$s_k = \lambda_{prob}(\pi'_i)_k + (1 - \lambda_{prob})d_k. \tag{3}$$

Note that $(\pi'_i)_k$ is inversely proportional to $\|q_k\|$. We define d_k as cosine similarity between q_k and the local (minus) gradient (see Section 5):

$$d_k = \frac{q_k \left(-\nabla_{\pi'_i}\mathcal{L}(x')\right)^T}{\|q_k\| \cdot \|\nabla_{\pi'_i}\mathcal{L}(x')\|} \tag{4}$$

We then select m tokens with the highest scores s_k.

4 Experiments

In this section, we evaluate MANGO on various NLP tasks and compare it to recent NLP attacks.

Baselines We compare our method with the latest white-box GBDA attack (Guo et al., 2021), as well as recent black-box attacks implemented in TextAttack (Morris et al., 2020b): BERT-Attack (Li et al., 2020), BAE (Garg and Ramakrishnan, 2020) and TextFooler (Jin et al., 2020). To emphasize the importance of multi-step quantization, we evaluate the Naive version of MANGO that performs quantization in one step. MANGO, Naive and GBDA attacks use identical loss. All hyperparameters are listed in appendix A.

Tasks We attack BERT models from TextAttack fine-tuned on three text classification tasks: AG News (Zhang et al., 2015), Yelp Reviews (Zhang et al., 2015), IMDB (Maas et al., 2011), and MNLI task for natural language inference, (Williams et al., 2018). In MNLI p., an attack is allowed to modify only the premise, and in MNLI h., only the hypothesis. For each task, we randomly select 1000 attack targets from the training set. We use a training set as it provides more challenging targets and is more relevant to Adversarial Training (Bai et al., 2021).

Task	Method	Adv.	Adv. prob.	USE sim.	BERTScore	Δ perp.	Δ gram.	# queries
AG News (99.6)	TextFooler	16.2	43.7 ± 26.0	0.81 ± 0.13	0.83 ± 0.10	373 ± 548	0.26 ± 0.69	334 ± 224
	Bert-Attack	20.1	45.7 ± 27.7	0.83 ± 0.11	0.86 ± 0.09	86 ± 133	0.06 ± 0.49	620 ± 472
	BAE	12.6	41.1 ± 24.1	0.78 ± 0.16	0.84 ± 0.11	157 ± 289	0.07 ± 0.53	424 ± 353
	naive	43.7	44.5 ± 43.1	0.82 ± 0.10	0.87 ± 0.06	67 ± 141	0.13 ± 0.62	102 ± 6
	GBDA	12.9	13.7 ± 29.4	0.72 ± 0.13	0.80 ± 0.09	241 ± 382	0.17 ± 0.72	1098 ± 69
	MANGO	**2.7**	3.2 ± 15.3	0.78 ± 0.10	0.83 ± 0.06	30 ± 108	0.10 ± 0.63	496 ± 125
IMDB (98.2)	TextFooler	0.6	34.1 ± 16.9	0.94 ± 0.08	0.93 ± 0.07	108 ± 214	01.03 ± 1.81	761 ± 1 000
	Bert-Attack	0.6	28.0 ± 18.6	0.96 ± 0.07	0.96 ± 0.05	19 ± 38	0.05 ± 0.65	900 ± 922
	BAE	**0.2**	29.3 ± 18.3	0.95 ± 0.08	0.95 ± 0.06	27 ± 59	0.10 ± 0.76	651 ± 665
	naive	30.5	31.1 ± 42.6	0.86 ± 0.09	0.83 ± 0.10	288 ± 346	1.56 ± 2.75	100 ± 13
	GBDA	6.3	7.0 ± 21.3	0.83 ± 0.11	0.79 ± 0.08	294 ± 271	1.44 ± 2.22	1082 ± 146
	MANGO	0.3	0.7 ± 5.7	0.88 ± 0.07	0.83 ± 0.08	59 ± 73	0.99 ± 2.15	1647 ± 746
Yelp (99.9)	TextFooler	4.5	31.7 ± 22.6	0.92 ± 0.10	0.93 ± 0.06	90 ± 192	0.50 ± 01.06	495 ± 526
	Bert-Attack	**1.9**	28.3 ± 19.1	0.93 ± 0.09	0.94 ± 0.06	16 ± 38	0.00 ± 0.55	665 ± 713
	BAE	2.8	30.5 ± 21.1	0.92 ± 0.11	0.93 ± 0.06	29 ± 130	0.06 ± 0.60	501 ± 525
	naive	35.1	35.8 ± 45.4	0.82 ± 0.13	0.84 ± 0.09	25 ± 84	0.75 ± 1.93	102 ± 3
	GBDA	4.5	4.9 ± 18.3	0.79 ± 0.12	0.81 ± 0.06	5 ± 42	0.37 ± 1.59	1101 ± 35
	MANGO	8.5	8.9 ± 27.4	0.82 ± 0.12	0.80 ± 0.07	-30 ± 38	0.34 ± 1.72	1128 ± 718
MNLI premise (94.7)	TextFooler	94.7	-	-	-	-	-	-
	Bert-Attack	3.9	34.3 ± 23.5	0.93 ± 0.08	0.96 ± 0.04	30 ± 58	0.02 ± 0.26	146 ± 148
	BAE	5.0	34.3 ± 23.5	0.92 ± 0.09	0.95 ± 0.04	42 ± 107	0.01 ± 0.26	112 ± 108
	naive	31.6	33.9 ± 24.0	0.91 ± 0.07	0.94 ± 0.04	64 ± 116	-0.01 ± 0.50	97 ± 23
	GBDA	5.9	30.3 ± 21.9	0.80 ± 0.12	0.87 ± 0.07	301 ± 446	0.09 ± 0.67	1044 ± 247
	MANGO	**2.4**	31.6 ± 23.3	0.88 ± 0.08	0.91 ± 0.05	73 ± 123	0.05 ± 0.60	326 ± 125
MNLI hyp. (94.7)	TextFooler	6.5	35.5 ± 24.2	0.94 ± 0.07	0.95 ± 0.04	77 ± 139	0.13 ± 0.39	77 ± 44
	Bert-Attack	2.6	34.3 ± 24.3	1.00 ± 0.01	0.97 ± 0.03	1 ± 0	0.00 ± 0.06	95 ± 62
	BAE	3.5	34.8 ± 24.4	0.95 ± 0.06	0.97 ± 0.03	29 ± 57	0.03 ± 0.25	74 ± 39
	naive	8.4	32.1 ± 22.7	0.89 ± 0.08	0.93 ± 0.04	115 ± 209	0.07 ± 0.36	97 ± 23
	GBDA	0.6	27.4 ± 21.4	0.81 ± 0.12	0.89 ± 0.06	220 ± 454	0.09 ± 0.42	1044 ± 247
	MANGO	**0.3**	30.0 ± 22.4	0.89 ± 0.09	0.93 ± 0.04	85 ± 155	0.06 ± 0.38	258 ± 68

Table 1: Results for black-box and white-box methods. We report: the initial training accuracy of BERT model (under Task); training accuracy under attack (Adv.); probability of ground-truth label prediction under attack (Adv. prob.); similarity between the original and perturbed text computed with USE (Cer et al., 2018) (USE sim.) and with F1 BERTScore (BERTScore); percent change in perplexity computed with GPT-2 (Radford et al., 2019) (Δ perpl.); increase in the number of grammar errors (Δ gram.) obtained with LanguageTool (github.com/jxmorris12/language_tool_python); average number of queries to a victim model (# queries). We omit results for TextFooler on MNLI p., as it has not generated any adversarial example. We also report standard deviation for each result, except adversarial accuracy as it is simply the percent of successful attacks. Our MANGO method achieves superior results on most tasks while maintaining high semantic similarity and grammar fluency. The best results for Adv. are **bold**.

Results Results can be found in Table 1. Our MANGO substantially reduces the training accuracy of the BERT model in all tasks, while maintaining a high level of semantic similarity to the original input. The attacks of MANGO are difficult (low Adv. prob., which indicates that model misclassifies an example by a large margin), fluent (low Δ perp.) and do not flaw the grammatical correctness (low Δ gram.).

In almost all settings, MANGO outperforms other attacks in terms of training accuracy, which we believe to be the fairest metric for comparing optimization-based methods with black-box ones due to inherent design biases (see Appendix B).

MANGO surpasses the recent state-of-the-art optimization-based GBDA attack in terms of most considered metrics: in terms of Adv. acc. and BERTScore on 4/5 tasks and in terms of USE sim., Δ perpl. and Δ gram. on 5/5 tasks.

Moreover, MANGO achieves considerably better results than its Naive version, emphasizing the importance of multi-step quantization.

Qualitative Results We provide qualitative analysis of a few adversarial examples generated by BAE, GBDA, and MANGO in Appendix C.

5 Ablation Study

In this section, we evaluate three solutions from Section 3 that improve the core idea of multi-step quantization:

1. selection of probability vector to quantization by maximal entropy (instead of minimal entropy, which seems more natural choice),
2. scoring token candidates by weighted mean of token probability and gradient direction score (eq. 4),
3. resetting optimizer after every quantization.

Figure 1 compares different MANGO settings. We may observe that selection of probability vector for quantization by maximal entropy ("max entropy") is better than selection by minimal entropy ("min entropy"). Resetting the optimizer after every quantization enhances the performance for both "max entropy" and "min entropy" settings. Finally, we see that MANGO benefits from using both token's probability and gradient direction to score token candidates.

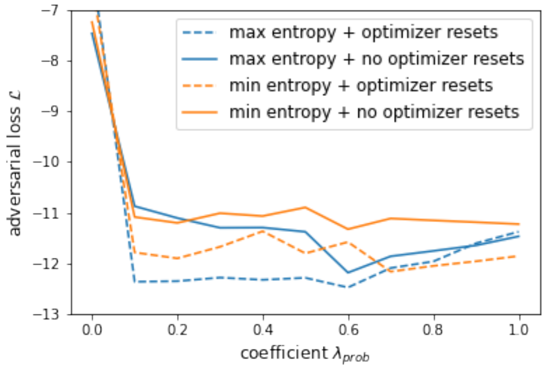


Figure 1: Final adversarial losses for different MANGO setting. "max entropy + optimizer resets" stands for a version of MANGO that selects probability vector for quantization by maximal entropy and resets optimizer after every quantization. Rest of the names follow the same pattern. We also present the influence of the coefficient λ_{prob} used in token candidates scoring function (eq. 4). Loss values are averaged over 10 samples from IMDB dataset.

6 Visualization of Quantization Gap

To visualize the quantization gap between adversarial loss for continuous and discrete text representations, we compared adversarial losses of MANGO, GBDA and a Naive version of MANGO that does not use multi-step quantization. The comparison can be found in Figure 2. We observe that the Naive method converges to the lowest value loss in the optimization phase, but the value explodes after quantization. The GBDA method, which samples probability vectors that resemble discrete one-hot vectors using Gumbel-softmax (Jang et al., 2017), reaches a higher minimum, but its quantization gap is much smaller than that of Naive method. Finally, in the case of MANGO, we observe sudden peaks and slow declines of loss values that correspond to the quantization-compensation loop, in which the quantization of single tokens is followed by the compensation of the quantization gap. After optimization, MANGO continues to quantize tokens step by step further decreasing the loss. MANGO obtains a significantly lower final adversarial loss than GBDA and Naive, avoiding the quantization gap.

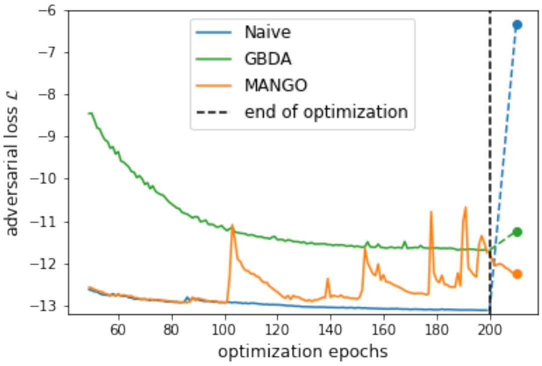


Figure 2: Adversarial loss for epochs 50-200 of optimization for Naive, GBDA and MANGO methods. The vertical dashed line shows the end of optimization. Naive and GBDA methods immediately quantize the tokens, while MANGO do it step by step. The rightmost points shows the final adversarial loss value. We observe that after optimization, MANGO continues to quantize tokens step by step and eventually reaches the best adversarial loss value. Loss values are averaged over 9 samples from IMDB dataset.

7 Conclusion

We developed MANGO, a novel optimization-based attack against Transformer models that mitigates the gap between adversarial loss for continuous and discrete text representations using a quantization-compensation loop. MANGO achieves superior results on various NLP tasks, outperforming recent black-box and optimization-based attacks.

Limitations

One limitation is that the number of queries of MANGO to the attacked model depends on the length of the input sequence. Therefore, MANGO may suffer a long attack time on datasets with long sequences (like IMDB or Yelp).

Moreover, MANGO is restricted only to token replacement. The inability to insert or remove tokens can lead to reduced attack performance.

The most important limitation is the white-box nature of MANGO that excludes it from applications when the internal model's states cannot be known. To partially circumvent this limitation, we propose Gray MANGO - a version of MANGO that can be used in the loosened black-box setting, which we call gray-box setting (see appendix D).

Acknowledgements

The work of Klaudia Bałazy was carried out within the research project "Bio-inspired artificial neural network" (grant no. POIR.04.04.00-00-14DE/18-00) within the Team-Net program of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund. Piotr Gaiński and Klaudia Bałazy are affiliated with Doctoral School of Exact and Natural Sciences at the Jagiellonian University.

References

Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. 2021. Recent advances in adversarial training for adversarial robustness. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4312–4321. ijcai.org.

Nathaniel Berger, Stefan Riezler, Sebastian Ebert, and Artem Sokolov. 2021. Don't search for a search method - simple heuristics suffice for adversarial text attacks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8216–8224. Association for Computational Linguistics.

Nicholas Carlini and David A. Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57. IEEE Computer Society.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. 2017. ZOO: zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2017, Dallas, TX, USA, November 3, 2017*, pages 15–26. ACM.

Xiangyi Chen, Sijia Liu, Kaidi Xu, Xingguo Li, Xue Lin, Mingyi Hong, and David D. Cox. 2019. Zo-adamm: Zeroth-order adaptive momentum method for black-box optimization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7202–7213.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4324–4333, Florence, Italy. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 31–36. Association for Computational Linguistics.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. 2021. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5747–5757. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O. Hero III, and Pramod K. Varshney. 2020. A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. *IEEE Signal Process. Mag.*, 37(5):43–54.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

John Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020a. Reevaluating adversarial examples in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3829–3839, Online. Association for Computational Linguistics.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020b. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 119–126. Association for Computational Linguistics.

Yurii E. Nesterov and Vladimir G. Spokoiny. 2017. Random gradient-free minimization of convex functions. *Found. Comput. Math.*, 17(2):527–566.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Yining Wang, Simon S. Du, Sivaraman Balakrishnan, and Aarti Singh. 2018. Stochastic zeroth-order optimization in high dimensions. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, volume 84 of *Proceedings of Machine Learning Research*, pages 1356–1365. PMLR.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Jincheng Xu and Qingfeng Du. 2020. Texttricker: Loss-based and gradient-based adversarial attacks on text classification models. *Engineering Applications of Artificial Intelligence*, 92:103641.

Jin Yong Yoo, John X. Morris, Eli Lifland, and Yanjun Qi. 2020. Searching for a search method: Benchmarking search algorithms for generating NLP adversarial examples. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online, November 2020*, pages 323–332. Association for Computational Linguistics.

Lifan Yuan, Yichi Zhang, Yangyi Chen, and Wei Wei. 2021. Bridge the gap between cv and nlp! a gradient-based textual adversarial attack framework.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

A Hyperparameters

MANGO To save computational resources during **candidates selection**, we use the dynamic number of candidates m. We rescale the candidate scores s_k to $[0, 1]$ and take at most $M = 5$ candidates whose scores differ from the best score at most by a threshold $T = 0.5$: $s_k \geq \max_j s_j - T$. We use $\lambda_{prob} = 0.5$ in Equation (4).

White-Box Attacks MANGO, Naive and GBDA methods use the loss function Equation (2) with the same parameters $\lambda_s = 20$, $\lambda_f = 1$, $\kappa = 5$ (taken from Guo et al. (2021)) for all tasks, except Yelp, where they use $\lambda_s = 10$. As a reference model g, we used the GPT-2 model downloaded from the official GBDA repository. We set $C = 10$ for initialization of the adversarial sample parameters. The number of optimization epochs $S = 100$ for all models and the batch size in GBDA was set to 10.

Black-Box Attacks We take TextFooler, BertAttack, and BAE implementations from TextAttack (Morris et al., 2020b) along with their original parameters. For fair comparison, we set the USE similarity threshold to the lowest value (0.2) used along these methods. Following the GBDA paper, we slightly modify the BertAttack method to mitigate its problem with subtokens and extremely long time of attack.

B Comparison Fairness

When comparing the results of optimization-based (MANGO, GBDA, Naive MANGO) and black-box methods (TextFooler, Bert-Attack, BAE), we should note that black-box methods stop perturbing text as soon as they fool the model, while optimization-based attacks minimize adversarial loss (that encourage them to fool the model by some margin) for some fixed number of steps. The former improves similarity metrics (USE sim., BERTScore) and the latter highly decreases the model's prediction on ground-truth labels (Adv. prob.), increasing the difficulty of generated sample. Therefore, we believe that training accuracy under attack (Adv.) is the fairest metric to make a direct comparison between optimization-based and classic black-box methods.

C Attack Examples

To draw some insights into MANGO performance, we compared examples generated by BAE, GBDA and MANGO. We chose all the sentences from AG News and MNLI hypothesis that were successfully perturbed by the three considered methods and on which the methods obtained USE cosine similarity score greater than 0.9. We then sampled two sentences from AG News and two from MNLI hypothesis tasks. To avoid cherry-picking, we fixed a seed and sampled only once. Examples can be found in table 2 and in table 3. We are careful in drawing any conclusion from the qualitative results, however, there seems to be a trend consistent with the result from table 1 and our observations from appendix B: BAE perturbs less words than GBDA and MANGO, but also achieves lower confidence of the mislassified label.

D Gray MANGO

To circumvent the white-box nature of MANGO attack, we additionally develop Gray MANGO: a version of MANGO that can be used in the loosened black-box setting, which we call gray-box setting.

Gray-Box Setting Gray MANGO is not strictly a black-box attack, as it requires the attacked model to take probability vectors and needs access to token vocabulary V. Transformer-based models satisfy these assumptions: they usually share the same V and their embedding function e can be used for both one-hot and probability vectors. However, to avoid misconception, we call this loosened black-box setting a grey-box setting.

Zeroth-Order Optimization Gray MANGO is based on Zeroth-Order Optimization (ZOO) (Nesterov and Spokoiny, 2017). The idea of ZOO is to approximate the gradient using only zeroth order loss values. In computer vision, Chen et al. (2017) developed a ZOO-based attack that significantly outperforms other black-box attacks. We believe that this success can be transferred to the NLP domain. Berger et al. (2021) have proposed an NLP attack that uses a discrete version of ZOO, but the results were unsatisfactory. Our Gray MANGO method is the first to successfully adapt the continuous version of ZOO in NLP attacks.

Formulation The main modification with respect to MANGO is the use of the zeroth-order gradient approximation of the gradient $\nabla_{\Theta'}\mathcal{L}(x')$ (Liu et al.,

Method	Prediction	Sentence
		AG News - Example no 1.
Original	world (100%)	air india trial witness said motivated by revenge (reuters) reuters - a desire for revenge motivated a prosecution witness to tell the air india bombing trial he had been asked to carry an mysterious suitcase on to an airliner, defense lawyers charged on wednesday.
BAE	sci/tech (61%)	air india trial witness said motivated by revenge (reuters) website - a desire for revenge motivated a prosecution witness to tell the air india company s he had been asked to carry an mysterious suitcase on to an account, defense lawyers charged on wednesday.
GBDA	business (99%)	air india trial witness said motivated by revenge - today investigative reuters reporting a desire for revenge motivated criminal prosecution witnesses to tell the air canada strike trial he had been asked to carry an mysterious suitcase on to an airliner, defense lawyers charged on tuesday.
MANGO	business (100%)	air indies trial witness said motivated by revenge (reuters) time - a desire for revenge motivated a prosecution witness to tell the air america arson trial he had been asked to carry a mysterious suitcase on to an airliner, defense lawyers charged on monday.
		AG News - Example no 2.
Original	business (91%)	brazil passes bankruptcy reform brazilian congress gives the green light to a long awaited overhaul of bankruptcy laws, which it hopes will reduce business and credit costs.
BAE	sci/tech (95%)	brazil passes bankruptcy reform brazilian congress gives the green light to a long awaited overhaul of copyright laws, which it hopes will reduce business and credit costs.
GBDA	world (95%)	brazil passes bankruptcy reform brazilian congress gives the green light to a long awaited overhaul of privacy laws, which it aims will reduce tourism and population impacts.
MANGO	world (99%)	brazil passes golf reform brazilian congress gives the green light to a long awaited overhaul of elections laws, which it hopes will reduce spending and maintenance costs.

Table 2: Attack examples sampled from AG News dataset.

2020):

$$\widetilde{\nabla}_{\Theta'}\mathcal{L}(x') = \frac{1}{K}\sum_{i=1}^{K}\frac{\mathcal{L}(\sigma(\Theta' + \mu u_i)) - \mathcal{L}(x')}{\mu}u_i,$$

where u_i is a noise sampled from the normal distribution, μ is the scale factor and $\sigma(\Theta' + \mu u_i)$ is x' with noise μu_i added to its parameters Θ'.

As $\widetilde{\nabla}_{\Theta'}\mathcal{L}(x')$ is unstable, we set $\lambda_{prob} = 1$ and use AMSGrad variant of Adam (Chen et al., 2019) without reset after every quantization. To reduce the high dimensionality of x', which is an issue in ZOO (Wang et al., 2018), we disallow replacement of the original token with tokens that have a cosine similarity of GloVe (Pennington et al., 2014) embedding lower than 0.

Hyperparameters We use almost the same parameters as for MANGO (see appendix A), but with $\lambda_{prob} = 1$, $S = 140$ and $\lambda_s = 80$. To save computational resources, we set $S = 100$ for the IMDB and Yelp datasets. Based on small grid search, we set the noise scaling parameter $\mu = 0.1$.

Method	Prediction	Sentence
MNLI hypothesis - Example no 1.		
Original	contraditcion (96%)	**premise**: the houses are built to a long - standing design and are filled with embroidery, lace, and crochet work. **hypothesis**: there is no embroidery in the houses.
BAE	neutral (45%)	**hypothesis:** there is no fire in the houses.
GBDA	neutral (100%)	**hypothesis:** there is liturgical embroidery in the houses.
MANGO	neutral (99%)	**hypothesis:** there is no erosion in the ruins.
MNLI hypothesis - Example no 2.		
Original	contradiction (100%)	**premise**: whether the service emerges as an adaptation from primary care or as an innovation from the ed is less important than whether it can be evaluated to the satisfaction of those who make key decisions about whether it becomes part of standard practice. **hypothesis:** key decision makers are not important to decided things.
BAE	neutral (96%)	**hypothesis:** consensus decision makers are not important to first things.
GBDA	neutral (98%)	**hypothesis:** key decision makers are noted fairchild – emery associates.
MANGO	neutral (99%)	**hypothesis:** older ahlers are also important in this regard.

Table 3: Attack examples sampled from MNLI hypothesis task.

Results We evaluated the Gray MANGO method and compared it to vanilla MANGO. Results can be found in table 4.

Gray MANGO, which is the first method to incorporate continuous ZOO in NLP attack, performs competitively with other black-box attacks in terms of training accuracy reduction, but struggles to keep adversarial examples similar to original texts. We believe that the performance of Gray MANGO may be greatly elevated by a more thorough design of ZOO components (Liu et al., 2020). This may be an interesting topic for future research.

Task	Method	Adv.	Adv. prob.	USE sim.	BERTScore	Δ perp.	Δ gram.	# queries
AG News (99.6)	TextFooler	16.2	43.7 ± 26.0	0.81 ± 0.13	0.83 ± 0.10	373 ± 548	0.26 ± 0.69	334 ± 224
	Bert-Attack	20.1	45.7 ± 27.7	0.83 ± 0.11	0.86 ± 0.09	86 ± 133	0.06 ± 0.49	620 ± 472
	BAE	12.6	41.1 ± 24.1	0.78 ± 0.16	0.84 ± 0.11	157 ± 289	0.07 ± 0.53	424 ± 353
	G-MANGO	9.7	11.0 ± 25.8	0.57 ± 0.23	0.67 ± 0.14	16k ± 47k	-0.03 ± 0.61	3728 ± 244
	MANGO	**2.7**	3.2 ± 15.3	0.78 ± 0.10	0.83 ± 0.06	30 ± 108	0.10 ± 0.63	496 ± 125
IMDB (98.2)	TextFooler	0.6	34.1 ± 16.9	0.94 ± 0.08	0.93 ± 0.07	108 ± 214	01.03 ± 1.81	761 ± 1 000
	Bert-Attack	0.6	28.0 ± 18.6	0.96 ± 0.07	0.96 ± 0.05	19 ± 38	0.05 ± 0.65	900 ± 922
	BAE	0.2	29.3 ± 18.3	0.95 ± 0.08	0.95 ± 0.06	27 ± 59	0.10 ± 0.76	651 ± 665
	G-MANGO	8.6	10.8 ± 24.3	0.65 ± 0.21	0.66 ± 0.14	16k ± 38k	0.19 ± 1.97	3142 ± 669
	MANGO	0.3	0.7 ± 5.7	0.88 ± 0.07	0.83 ± 0.08	59 ± 73	0.99 ± 2.15	1647 ± 746
Yelp (99.9)	TextFooler	4.5	31.7 ± 22.6	0.92 ± 0.10	0.93 ± 0.06	90 ± 192	0.50 ± 01.06	495 ± 526
	Bert-Attack	**1.9**	28.3 ± 19.1	0.93 ± 0.09	0.94 ± 0.06	16 ± 38	0.00 ± 0.55	665 ± 713
	BAE	2.8	30.5 ± 21.1	0.92 ± 0.11	0.93 ± 0.06	29 ± 130	0.06 ± 0.60	501 ± 525
	G-MANGO	15.7	16.4 ± 32.1	0.62 ± 0.27	0.69 ± 0.15	14k ± 36k	-0.01 ± 1.68	2803 ± 516
	MANGO	8.5	8.9 ± 27.4	0.82 ± 0.12	0.80 ± 0.07	-30 ± 38	0.34 ± 1.72	1128 ± 718
MNLI p. (94.7)	TextFooler	94.7	-	-	-	-	-	-
	Bert-Attack	3.9	34.3 ± 23.5	0.93 ± 0.08	0.96 ± 0.04	30 ± 58	0.02 ± 0.26	146 ± 148
	BAE	5.0	34.3 ± 23.5	0.92 ± 0.09	0.95 ± 0.04	42 ± 107	0.01 ± 0.26	112 ± 108
	G-MANGO	35.1	33.4 ± 23.0	0.77 ± 0.18	0.84 ± 0.10	5876 ± 19k	-0.06 ± 0.64	3158 ± 761
	MANGO	**2.4**	31.6 ± 23.3	0.88 ± 0.08	0.91 ± 0.05	73 ± 123	0.05 ± 0.60	326 ± 125
MNLI h. (94.7)	TextFooler	6.5	35.5 ± 24.2	0.94 ± 0.07	0.95 ± 0.04	77 ± 139	0.13 ± 0.39	77 ± 44
	Bert-Attack	2.6	34.3 ± 24.3	1.00 ± 0.01	0.97 ± 0.03	1 ± 0	0.00 ± 0.06	95 ± 62
	BAE	3.5	34.8 ± 24.4	0.95 ± 0.06	0.97 ± 0.03	29 ± 57	0.03 ± 0.25	74 ± 39
	G-MANGO	9.1	30.8 ± 22.4	0.83 ± 0.13	0.89 ± 0.07	1402 ± 3272	0.04 ± 0.35	3387 ± 807
	MANGO	**0.3**	30.0 ± 22.4	0.89 ± 0.09	0.93 ± 0.04	85 ± 155	0.06 ± 0.38	258 ± 68

Table 4: Comparison of Gray MANGO with black-box methods and vanilla MANGO. We report: the initial training accuracy of BERT model (under Task); training accuracy under attack (Adv.); probability of ground-truth label prediction under attack (Adv. prob.); similarity between the original and perturbed text computed with USE (Cer et al., 2018) (USE sim.) and with F1 BERTScore (BERTScore); percent change in perplexity computed with GPT-2 (Radford et al., 2019) (Δ perpl.); increase in the number of grammar errors (Δ gram.) obtained with LanguageTool (github.com/jxmorris12/language_tool_python); average number of queries to a victim model (# queries). We omit results for TextFooler on MNLI p., as it has not generated any adversarial example. We also report standard deviation for each result, except adversarial accuracy as it is simply the percent of successful attacks. The best results for Adv. are **bold**.

TwiRGCN: Temporally Weighted Graph Convolution for Question Answering over Temporal Knowledge Graphs

Aditya Sharma
IISc, Bangalore
adityasharma@iisc.ac.in

Apoorv Saxena
IISc, Bangalore
apoorvsaxena@iisc.ac.in

Chitrank Gupta
IIT Bombay
chigupta2011@gmail.com

Mehran Kazemi
Google Research, Montreal
mehrankazemi@google.com

Partha Talukdar
Google Research, India
partha@google.com

Soumen Chakrabarti
IIT Bombay
soumen@cse.iitb.ac.in

Abstract

Recent years have witnessed interest in Temporal Question Answering over Knowledge Graphs (TKGQA), resulting in the development of multiple methods. However, these are highly engineered, thereby limiting their generalizability, and they do not automatically discover relevant parts of the KG during multi-hop reasoning. Relational graph convolutional networks (RGCN) provide an opportunity to address both of these challenges – we explore this direction in the paper. Specifically, we propose a novel, intuitive and interpretable scheme to modulate the messages passed through a KG edge during convolution based on the relevance of its associated period to the question. We also introduce a gating device to predict if the answer to a complex temporal question is likely to be a KG entity or time and use this prediction to guide our scoring mechanism. We evaluate the resulting system, which we call TwiRGCN, on a recent challenging dataset for multi-hop complex temporal QA called *TimeQuestions*. We show that TwiRGCN significantly outperforms state-of-the-art models on this dataset across diverse question types. Interestingly, TwiRGCN improves accuracy by 9–10 percentage points for the most difficult ordinal and implicit question types.

1 Introduction

Question answering (QA) is a key problem in natural language processing and a long-lasting milestone for artificial intelligence. A large class of approaches for QA makes use of knowledge graphs (KG), which are multi-relational graphs representing facts (KGQA). Temporal KGs (TKG) represent facts that are only valid for specific periods of time as *(subject, relation, object, time range)*, for example, *(Franklin D Roosevelt, position held, President of USA,* $[1933, 1945]$*)*. The problem of answering questions that require temporal reasoning over TKGs (TKGQA) is a special case of KGQA that specifically focuses on the following challenge: temporal questions constrain answers through temporal notions, e.g., “*who was the first president of US during WW2?*” Developing systems for temporal QA is of immense practical importance for many applications. It is considered a more challenging problem than KGQA (Bhutani et al., 2019; Saxena et al., 2020), where questions are typically about persistent, non-temporal facts (e.g., place of birth), with only a small portion of the questions requiring any temporal reasoning (Jia et al., 2018a).

Even though a variety of models have been proposed for the TKGQA recently, they suffer from the following problems: 1) they are either highly engineered toward the task (Jia et al., 2021; Chen et al., 2022) or 2) they do not incorporate graph structure information using Graph Neural Networks (GNN) (Mavromatis et al., 2021; Shang et al., 2022; Saxena et al., 2021). We explore the following hypotheses in this paper: 1) a simple GNN-based solution could generalize better and offer higher performance than highly engineered GNN-based, and TKG embedding-based models; 2) a multi-layer GNN model could do multi-hop reasoning across its layers; 3) not all edges (temporal facts) are equally important for answering temporal questions (see Figure 1), so GNN solutions could benefit from temporally weighted edge convolutions.

Following the aforementioned hypotheses, we develop a novel but architecturally simple TKGQA system that we call “Temporally weighted Relational Graph Convolutional Network” (TwiRGCN). It is based on the Relational Graph Convolutional Network (RGCN) proposed by Schlichtkrull et al. (2018). TwiRGCN introduces a question-dependent edge weighting scheme that modulates convolutional messages passing through a temporal fact edge based on how relevant the time period of that edge is for answering a particular question. In RGCN, convolution messages from all TKG edges are weighted equally. But all edges are not equally important for answering temporal

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2049–2060
May 2-6, 2023

questions. For example, in Figure 1, to answer the question "*Who was the first president of the US during WW2?*" the edge with Bill Clinton has little relevance for answering the question. But, regular RGCN would still weigh all edges equally. We address this shortcoming through our proposed modulation. We impose soft temporal constraints on the messages passed during convolution, amplifying messages through edges close to the time period relevant for answering the question while diminishing messages from irrelevant edges. This leads to better, more efficient learning as we are not confusing our model with unnecessary information, as evidenced by our significantly improved performance without the need for any heavy engineering. We explore two different strategies for our convolutional edge weighting, which show complementary strengths. Our experiments establish that TwiRGCN significantly outperforms already strong baselines on *TimeQuestions*. Our contributions are:

- We propose TwiRGCN, a simple and general TKGQA system that computes question-dependent edge weights to modulate RGCN messages, depending on the temporal relevance of the edge to the question.
- We explore two novel and intuitive schemes for imposing soft temporal constraints on the messages passed during convolution, amplifying messages through edges close to the time relevant for answering the question while diminishing messages from irrelevant edges. We also propose an answer-gating mechanism based on the likelihood that the answer is an entity or time.
- Through extensive experiments on a challenging real-world dataset, we find that TwiRGCN substantially outperforms prior art in overall accuracy, and by 9–10% on the implicit and ordinal type questions — categories that require significant temporal reasoning.
- We augment *TimeQuestions* with a TKG and release both code and data at https://github.com/adi-sharma/TwiRGCN.

2 Related Work

Most KGQA systems have focused on answering questions from simple (i.e., 1-hop fact-based questions) (Berant et al., 2013) to multi-hop complex questions requiring multi-fact reasoning (Sun et al., 2019; Saxena et al., 2020). However, only a small fraction of these questions require any temporal reasoning (Jia et al., 2018a). Recent efforts have tried to overcome this gap by proposing models as well as datasets to explicitly focus on temporal reasoning. We review these below.

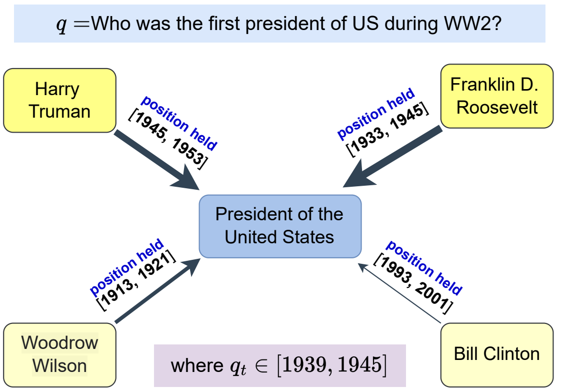


Figure 1: An illustrative example of how our temporal gating described in Section 4.2 modulated the incoming graph convolution messages for one node depending on the time period of interest for the question. The thickness of an edge here is proportional to the value of the temporal edge weight $m_{tq}^{(e)}$ for that edge. In this example, the entities *Franklin D. Roosevelt* and *Harry Truman*, who were presidents during WW2 $[1939, 1945]$ get the top two highest weights, while *Woodrow Wilson*, who was president during WW1 $[1914, 1918]$ gets a smaller edge weight. In contrast, *Bill Clinton*, whose time period is unrelated to the question, gets a much lower edge weight. Thus, contributing very little to the convolution update of the 'President of the US' node.

Temporal KGQA methods: One line of work uses temporal constraints along with hand-crafted rules to find the answer (Bao et al., 2016; Luo et al., 2018; Jia et al., 2018b). A recent class of models has leveraged advances in TKG embedding methods for answering questions on Temporal KGs. CronKGQA (Saxena et al., 2021) does this by posing a question as a TKG completion problem and finds the answer using the TComplex (Lacroix et al., 2020) score function and BERT (Devlin et al., 2018) question embedding to complete the fact. TempoQR (Mavromatis et al., 2021) uses additional temporal supervision to enrich TKG embeddings, followed by a transformer-based decoder (Vaswani et al., 2017). TSQA (Shang et al., 2022) on the other hand estimate the time in the question and uses it to enrich TKG embeddings for finding the answer. SubGTR (Chen et al., 2022) infers question-relevant temporal constraints using TKG embeddings and applies them as filters to score entities in the question subgraph. Although we, too, use pre-trained TKG embeddings to initialize our generalized RGCN, we use the GNN framework to take advantage of the structural information in the KG in ways that they do not. Recent work (Teru et al., 2020) shows that GNN-based models can encode any logical rule corresponding to a path in the knowledge graph. We refer to this as structural information that shallow embedding-based models cannot access.

RGCN based QA systems: Graph neural networks are increasingly being used in QA systems not specifically meant for temporal reasoning. GraftNet (Sun et al., 2018) uses personalized PageRank to collect a query-relevant subgraph from a global KG, then an RGCN to predict the answer from the relevant subgraph. PullNet (Sun et al., 2019) loops over and expands GraftNet's subgraph to do multi-hop reasoning. EXAQT (Jia et al., 2021) is the system closest to ours: it addresses TKGQA and also uses an RGCN. The RGCN for answer prediction which works on the question subgraph is very similar to that in GraftNet. EXAQT augments it with dictionary matching, heavy engineering, and additional category information. In contrast, TwiRGCN uses a straightforward temporally weighted graph convolution followed by answer gating, as described in Section 4, while still achieving superior performance (see Section 5.3). More details in Section 5.2.

3 Preliminaries

3.1 Temporal Knowledge Graphs (TKG)

KG: Multi-relational graphs with entities (eg: Barack Obama, USA) as nodes and relations r between entities $\{s, o\}$ (e.g., president of) represented as typed edges between nodes. Each edge of this graph, together with endpoint nodes, represents a fact triple $\{s, r, o\}$, e.g., $\{\text{Barack Obama}, \text{president of}, \text{USA}\}$.

TKG: Numerous facts in the world are not perpetually true and are only valid for a certain time period. A TKG represents such a fact as a quadruple of the form $\{s, r, o, [t_{st}, t_{et}]\}$, where t_{st} is the start time and t_{et} is the end time of validity of the fact, e.g., $\{\text{Barack Obama}, \text{president of}, \text{USA}, [2009, 2017]\}$.

3.2 Question Answering on TKGs

Given a question q specified in natural language form and a TKG $\mathcal{G}$, TKGQA is the task of finding the answer to q based on the information that is available (or can be derived) from $\mathcal{G}$. A subgraph of $\mathcal{G}$ is a subset of its nodes with induced edges. In this paper, we assume each question is already associated with a subgraph $\mathcal{G}_q$ relevant to the question. We define $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{R}_q, \mathcal{T}_q, \mathcal{E}_q)$ as the subgraph of $\mathcal{G}$ associated with a question $q \in \mathcal{Q}$, where $\mathcal{Q}$ represents the set of all questions. Each edge $e \in \mathcal{E}_q$ represents a fact $\{v_i, r, v_j, [t_{st}, t_{et}]\}$, where $v_i, v_j \in \mathcal{V}_q$ are entity nodes, $r \in \mathcal{R}_q$ is the relation between them and $t_{st}, t_{et} \in \mathcal{T}_q$ are the start and end times for which the fact is valid.

3.3 Relational Graph Convolutional Networks

Given a KG, each node v_i is initialized to a suitable embedding $h_{v_i}^{(0)}$ at layer 0.Thereafter, Schlichtkrull et al. (2018) propose to update node embeddings $h_{v_i}^{(l+1)}$, at layer $(l+1)$, as follows:

$$h_{v_i}^{(l+1)} = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{W_r^{(l)} h_{v_j}^{(l)}}{|\mathcal{N}_i^r|} + W_0^{(l)} h_{v_i}^{(l)}\right) \quad (1)$$

where $\mathcal{N}_i^r$ is the set of neighbors of node v_i that are connected via relation edges of type r, $\mathcal{R}$ is the set of relations, $W_r^{(l)}$ are weight matrices associated with each relation type r and layer l. They are initialized using a basis decomposition method.

4 Proposed Method: TwiRGCN

In this section, we develop and describe TwiRGCN ("Temporally Weighted Relational Graph Convolutional Network"), our model for TKGQA.

4.1 Embedding for questions and KG facts

Question embedding: We pass the question text through a pre-trained encoder-only language model (LM) to obtain a question embedding. In particular, we prepend a [CLS] token to the input question and feed it into BERT (Devlin et al., 2019), and then use its output-layer [CLS] embedding as the question embedding q_B. We enable LM fine-tuning during training.

TKG preprocessing for RGCN initialization: We initialize entity and time embeddings using pre-trained TComplEx (Lacroix et al., 2020) embeddings.[1] To obtain these for the *TimeQuestions* dataset (Jia et al., 2021), we first construct a 'background KG' $\mathcal{G} = \bigcup_{q \in \mathcal{Q}} \mathcal{G}_q$ which is the union of all question subgraphs $\mathcal{G}_q$ in the train dataset. As in most temporal KGQA works, we discretize time to a suitable granularity (in our dataset, a year).[2] The graph on which TwiRGCN is run represents every entity as a node v_i and time as edge attribute t_j. Their initial (layer-0) RGCN embeddings $h_{v_i}^{(0)}$ and h_{t_j}, are set to the entity and time embeddings

---

[1] TComplEx is known to provide high-quality embeddings, but other TKG embedding methods such as TimePlex (Jain et al., 2020) can also be used.

[2] TwiRGCN can be extended to TKGQA datasets that do not provide subgraphs through recently proposed subgraph selection methods (Chen et al., 2022; Shang et al., 2022).

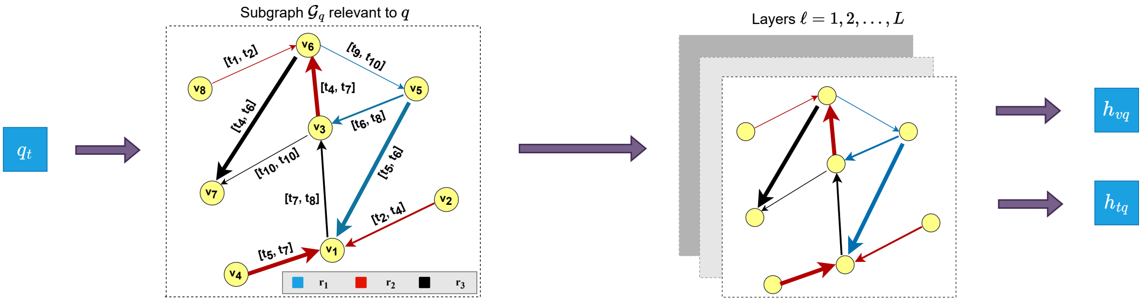


Figure 2: **Left:** Shows temporally weighted convolutional message passing described in Section 4.2 happening across a subgraph G_q for one layer. For the same, we get question-dependent temporal edge weights $m_{tq}^{(e)}$ using *question time*, q_t (described in 4.3). **Right:** As discussed in Section 4.2, embeddings are propagated in the subgraph G_q for a fixed number of layers (L) and hidden units of the final layer are pooled to get entity prediction, h_{vq}. We get time prediction, h_{tq}, by pooling the updated embeddings for all unique times in G_q.

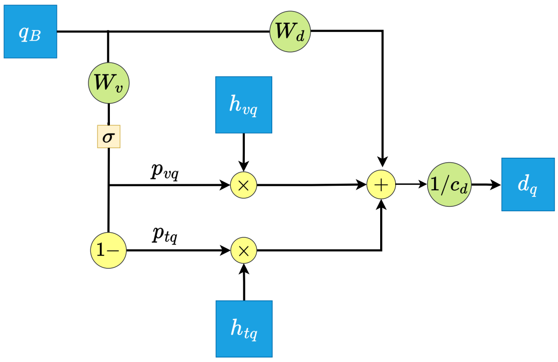


Figure 3: Predicting the answer based on gating entity prediction (h_{vq}) and time prediction (h_{tq}) of a subgraph G_q based on the likelihood that the answer is either an entity (p_{vq}) or time (p_{tq}) given question q, respectively. Details in Section 4.4.

obtained from TComplEx, respectively. We refer to h_{tj} as $h_{st}^{(e)}$ and $h_{et}^{(e)}$ depending on t_j appearing as start or end time for edge e, respectively. When $e = (i, r, j)$, we will use superscript (i, r, j) in place of (e).

4.2 Temporally modulated edge weights

Having available the question subgraph, and the initial entity and time embeddings, our system applies a temporally weighted graph convolution on the local subgraph to enable answering questions that require complex temporal reasoning over a KG. To achieve this, we introduce a question-dependent temporal edge weight $m_{tq}^{(i,r,j)} \in [-1, 1]$ for modulating the convolutional message passed through edge e valid from time $t_{st}^{(i,r,j)}$ to $t_{et}^{(i,r,j)}$ connecting node v_i to v_j via relation r, $\{v_i,\ r,\ v_j,\ [t_{st}^{(i,r,j)}, t_{et}^{(i,r,j)}]\}$ which assigns a weight to that edge depending on how relevant the time period of e is for answering question q. Then, motivated by Eqn. (1), we update the hidden state for a node v_i in the temporal KG at layer $(l+1)$ as:

$$h_{v_i}^{(l+1)} = \sigma\Bigg(W_0^{(l)} h_{v_i}^{(l)} + \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} m_{tq}^{(i,r,j)} \frac{W_r^{(l)} h_{v_j}^{(l)}}{|\mathcal{N}_i^r|}\Bigg). \quad (2)$$

See Figure 1 for an example update for one node.

As shown in Figure 2, after passing messages across a subgraph G_q over L such layers, we pool the hidden states from the final layer of all nodes in G_q to get h_{vq}, the entity prediction. Similarly, we pool the updated embeddings for all unique times in G_q to get h_{tq}, the time prediction. We describe in Section 4.4 how we use h_{vq} and h_{tq} to get the final predicted answer from our model. We use mean pooling in this work, but any other pooling operation can also be used.

4.3 Edge weighting formulations

We explore two different formulations for computing $m_{tq}^{(i,r,j)}$, namely *average* and *interval*, and discuss the motivations behind the two approaches. In Section 5, we empirically show that the inductive bias inherent in each of the two approaches makes them excel at different types of temporal reasoning while giving similar performance overall. We also provide an intuitive explanation of how the edge weighting formulations of the two approaches explain the difference between their empirical results. We first project the question embedding q_B, using a learned projection matrix W_{tq}, to find the *question time* embedding $q_t = W_{tq} q_B$. In the following, $m_{tq}^{(i,r,j)} = m_{tq}^{(e)}$ is the weight for edge e.

4.3.1 TwiRGCN (average)

In this variant, we calculate the edge modulation m_{tq} as the cosine similarity between the question time embedding, and the average of the embeddings for the start and the end time of an edge:

$$m_{tq}^{(e)} = \cos\left(\frac{h_{st}^{(e)} + h_{et}^{(e)}}{2}, q_t\right). \quad (3)$$

This formulation gives a high weight to an edge if the question time falls close to the middle of the time interval for an edge. For example, if the edge times are [2008, 2012] and the question time is 2010, the edge is weighted highly.

4.3.2 TwiRGCN (interval)

In this variant, $m_{tq}^{(e)}$ is defined as the mean of two cosine similarities: (1) the cosine similarity between the start time of the edge and the learned question time embedding, and (2) the cosine similarity between the end time of the edge and the learned question time embedding. Formally,

$$m_{tq}^{(e)} = \frac{\cos(h_{st}^{(e)}, q_t) + \cos(h_{et}^{(e)}, q_t)}{2}. \quad (4)$$

This formulation weighs an edge highly if question time t_q lies within the time interval of the edge.

Generality beyond temporal reasoning While we developed TwiRGCN for temporal reasoning, the edge weighting is more general and could extend to the case where q_t is a "goal" embedding for any goal-directed task.

4.4 Answer type gating

Question answering over TKGs may involve questions whose answer is an entity (e.g., *Who was ...?*) or whose answer is a time (e.g., *When did ...?*). We hypothesize that it should be possible to predict whether the answer to a question is an entity or a time based on the text of the question; making such a prediction helps filter out (or down-weight) a portion of the nodes in that graph that are less likely to be the answer. Toward this hypothesis, we introduce a gating mechanism that learns the likelihood that the answer is an entity p_{vq} or a time p_{tq} given the question:

$$p_{vq} = 1 - p_{tq} = \sigma(w_v q_B), \quad (5)$$

where w_v transforms q_B to a scalar and σ is the sigmoid function that ensures $0 \leq p_{vq} \leq 1$. As shown in Figure 3, we then compute a prediction embedding d_q for question q as a gated sum of the entity prediction and time prediction (see Section 4.2 and Figure 2) added to the question embedding:

$$d_q = \frac{1}{c_d}[p_{vq}h_{vq} + p_{tq}h_{tq} + W_d q_B], \quad (6)$$

where c_d is a constant hyperparameter and W_d is the weight for transforming q_B to the dimension of the entity and time embeddings. Having the prediction embedding d_q, we rank candidate answers (entities and times from the global TKG) based on their similarity to d_q.

Category	Question
Explicit	*Which team won the 2010 F1 world championship?* *What honour did Agatha Christie win in 1971?*
Implicit	*Who did Kevin Garnett play for before Celtics?* *Where was Leonardo Da Vinci when he died?*
Temporal	*What years did the team with fight song Steelers polka win the Superbowl?* *What year did Sam Elliott and Kathryn Ross marry?*
Ordinal	*What was the first satellite to maintain orbit around the earth in space?* *What is the third book of the twilight series?*

Table 1: Examples of questions from each category in *TimeQuestions* dataset, discussed in Section 5.1.

Training We score all possible answer entities and times as a cosine distance with the prediction embedding (d_q), scaled using a constant hyperparameter. We take a softmax over all these scores and train using the cross-entropy loss.

5 Evaluation

5.1 Dataset

Earlier works on TKGQA use the automatically generated CronQuestions dataset (Saxena et al., 2021). A recent analysis, however, shows that this dataset comes with several limitations that stem from its automatic construction method (Chen et al., 2022). Specifically, there are spurious correlations in the dataset that can be exploited by different models to achieve high accuracy (e.g., Mavromatis et al. (2021) report more than 90% accuracy overall and 99% in some categories on this dataset). Therefore, we base our experiments on a recent more challenging dataset, namely *TimeQuestions* (Jia et al., 2021), where the aforementioned models perform poorly (as seen in Table 3).

TimeQuestions has $13.5k$ manually curated questions divided into the train, valid, and test splits containing $7k$, $3.2k$, and $3.2k$ questions, respectively. The questions fall under four types: 'Explicit,' 'Implicit,' 'Temporal,' and 'Ordinal,' based on the type of temporal reasoning required to answer the questions. We show some examples of questions from each of these categories in Table 1. We augment this dataset with question-specific subgraphs generated from WikiData in the final step of the answer graph construction pipeline proposed by Jia et al. (2021). We preprocess all the obtained facts to the *(subject, relation, object, [start*

Dataset	Question
ComQA	*Who played Dumbledore in the 5th harry potter film?*
Complex-Web-Questions	*What is the name of the club the subject of "golden shoes" played for in 2010?*
Graph-Questions	*What sports were in both the 1912 summer Olympics and the 2008 Olympic games?*
LC-QuAD 2.0	*What is the start time for Heidi Klum has spouse as Seal?*
Free917	*What is the price of a 2012 jeep wrangler sport?*

Table 2: Examples of questions requiring temporal reasoning from KGQA datasets (see Section 5.1).

time, end time]) format, and restrict all times to years, a format used by most contemporary TKGs. We create a "background KG" described in Section 4.1 as a union of all subgraphs in the train set. This background KG contains $240k$ facts, $118k$ entities, and 883 relations. We include this augmented *TimeQuestions* dataset and associated code at https://github.com/adi-sharma/TwiRGCN.

Temporal subsets of KGQA datasets: *TimeQuestions* is a compilation of temporal questions from different KGQA datasets. We show the results in Table 4 on a subset of 5 such datasets included in the test set of *TimeQuestions* namely, ComQA (Abujabal et al., 2019), ComplexWebQuestions (Talmor and Berant, 2018), GraphQuestions (Su et al., 2016), LC-QuAD 2.0 (Dubey et al., 2019), and Free917 (Cai and Yates, 2013). Table 2 shows representative examples from these 5 datasets.

5.2 Baseline methods

We compare TwiRGCN against a spectrum of existing methods, including EXAQT, other TKGQA methods, and non-temporal KGQA methods.

Non-temporal KGQA methods: We include Unicorn (Pramanik et al., 2021), which uses Group Steiner Trees for answering questions. We test on two RGCN-based approaches for KGQA, namely, GRAFT-Net (Sun et al., 2018), which attends over relations of neighborhood edges based on the question, and PullNet (Sun et al., 2019), which extends GRAFT-Net for multi-hop questions.

TKGQA methods: We also compare against TKGQA methods CronKGQA (Saxena et al., 2021) and TempoQR (Mavromatis et al., 2021) recently proposed for the CronQuestions dataset. In contrast to TwiRGCN, these do not leverage the powerful GNN framework. CronKGQA frames QA as a KG completion problem to complete the fact the question is interested in, using the TComplex score function and BERT question embedding. TempoQR, on the other hand, enriches pre-trained TKG embeddings with additional supervision from the dataset and uses a transformer (Vaswani et al., 2017) based decoder to predict the final answer.

EXAQT: Jia et al. (2021) propose EXAQT, which is hitherto the best performer on *TimeQuestions*. It is also an RGCN-based TKGQA model that utilizes the GRAFT-Net framework. But in contrast to our model, EXAQT is heavily engineered. It utilizes the ground truth question category information from the dataset at train and test time, so it always knows whether the answer is temporal or belongs to another category. In contrast, our model learns the likelihood that the answer is an entity or time without any explicit supervision through our gating mechanism described in Section 4.4. EXAQT also uses explicit temporal signals from the question, extracted through a dictionary matching-based method using predefined temporal words such as ‘before’, ‘after’, ‘first’, ‘last’, ‘during’, etc. It then enriches its embeddings by utilizing the above in a multi-step end-to-end process. In contrast, our models do not have access to any such information with only a straightforward temporally weighted graph convolution followed by answer gating, as described in Section 4.

5.3 Results

TwiRGCN achieves new state-of-the-art: We compare the accuracy (Hits@1) for different Temporal KGQA models across all question categories found in *TimeQuestions* in Table 3. From this table, we see that our models TwiRGCN (average) and TwiRGCN (interval) achieve significant improvements of up to 3.3% overall absolute accuracy over the previous state-of-the-art model, EXAQT. Additionally, TwiRGCN (average) gets a 9.8% improvement over EXAQT in the ordinal category and TwiRGCN (interval) improves over EXAQT by 9.1% in the implicit category. The questions in both these categories require significant temporal reasoning to find the correct answer. Both models also show a marked improvement of up to 3.4% in the explicit question category.

TwiRGCN (average) vs (interval): Even though the two TwiRGCN variants achieve comparable overall accuracy, they do so in different ways, showing complementary strengths. TwiRGCN (average) achieves a 2.4% improvement over TwiRGCN (interval) in the ordinal category, while TwiRGCN (interval) improves over TwiRGCN (average) for

	Overall	Explicit	Implicit	Temporal	Ordinal
PullNet (Sun et al., 2019)	0.105	0.022	0.081	0.234	0.029
Uniqorn (Pramanik et al., 2021)	0.331	0.318	0.316	0.392	0.202
GRAFT-Net (Sun et al., 2018)	0.452	0.445	0.428	0.515	0.322
CronKGQA (Saxena et al., 2021)	0.462	0.466	0.445	0.511	0.369
TempoQR (Mavromatis et al., 2021)	0.416	0.465	0.36	0.4	0.349
EXAQT (Jia et al., 2021)	0.572	0.568	0.512	0.642	0.42
TwiRGCN *(average)*	**0.605**	**0.602**	0.586	0.641	**0.518**
TwiRGCN *(interval)*	**0.603**	0.599	**0.603**	**0.646**	0.494

Table 3: Comparison of Hits@1 for different Temporal KGQA methods on *TimeQuestions* dataset (Section 5.3). Interestingly, TwiRGCN improves accuracy over SOTA by 3.3% overall and by 9-10% for the most difficult ordinal & implicit question types.

	TwiRGCN	**EXAQT**
ComQA	**0.413**	0.292
ComplexWebQuestions	**0.728**	0.515
GraphQuestions	**0.382**	0.323
LC-QuAD 2.0	0.71	**0.732**
Free917	0	**0.17**

Table 4: Results for EXAQT and TwiRGCN on temporal subsets of well-known KGQA datasets, as discussed in Section 5.3. TwiRGCN beats EXAQT by a high margin up to 21% on ComQA, ComplexWebQuestions, and GraphQuestions, which include questions requiring multi-hop reasoning.

	With gating	**W/o gating**
TwiRGCN (average)	0.605	0.597
TwiRGCN (interval)	0.603	0.597

Table 5: Results of ablation study to see contributions of answer gating described in Section 4.4 on overall Hits@1. We that it contributes about 0.7% on average to the overall accuracy of our models

Temporal Distance from q_t	
Median	**5**
=0	18.3%
≤5	51.5%
≤20	74.8%

Table 6: The median temporal distance from learned q_t to extracted time is just 5 years, while we predict an exact match 18.3% of the time (discussed in Section 5.4).

the implicit and temporal question types. We intuitively explain this behavior as a consequence of their edge weighting function m_{tq}. In TwiRGCN (average), m_{tq} (defined in Eqn. (3)) is at its peak when the average time of edge is close to the question time, enabling it to reason between the temporal ordering of facts more effectively. Thus, helping it better answer ordinal questions of the type *first*, *fourth* or *last occurrence*, etc. In contrast, m_{tq} for TwiRGCN (interval), as defined in Eqn. (4), helps answer questions that require temporal reasoning over specific times rather than just a temporal ordering of facts, which are mainly present in the implicit and temporal categories.

Temporal subsets of KGQA datasets: As mentioned earlier, *TimeQuestions* is a compilation of temporal questions from different KGQA datasets. To provide a finer-grained comparison, we compare TwiRGCN to our most competitive baseline, EXAQT, on these subsets. We show the results in Table 4. TwiRGCN outperforms EXAQT by a high margin of up to 21% in Hits@1 on the *ComQA*, *ComplexWebQuestions*, and *GraphQuestions* datasets. These three datasets contain questions requiring complex multi-hop reasoning. In contrast, we are competitive but perform slightly worse than EXAQT on *LC-QuAD 2.0*, a templated dataset created from SPARQL queries, and *Free917* that primarily consists of quantity-based questions. These results show our model's generalizability and superior performance over our primary baseline on complex multi-hop questions. They also identify a failure mode for questions whose answers are quantities (explored in Section 5.4).

5.4 Analysis

Here we explore how our models behave qualitatively and look at example cases where they perform well and cases where they do not.

Ablation for answer gating: To understand the contribution of the proposed answer gating method described in Section 4.4, we perform an ablation study by removing answer gating from Eqn. (6). By comparing columns of Table 5 we can infer that our proposed answer gating contributes about 0.7% on average to the overall accuracy of our models.

How accurate is predicted question time? We predict a question time embedding q_t close to the time of interest for answering question q, as described in Section 4.3. Here we analyze the effectiveness of this prediction by getting the time with embedding closest to q_t. We then use regex-based time extraction on questions and can extract time for 1199 questions in the test set. As seen in Table 6, out of a time range of 2916 years (including BC and AD years), our median distance from learned question time to extracted time is just 5 years while we predict an exact match 18.3% of the time. Ad-

Ground Truth	Prediction	TwiRGCN (average)		TwiRGCN (interval)	
		with gating	w/o gating	with gating	w/o gating
Entity	Time	3.18 %	7.28 %	3.23 %	4.53 %
Time	Entity	7.99 %	6.88 %	7.3 %	7.82 %

Table 7: Percentage of questions for which answer is an entity but our model incorrectly predicts time and vice versa. We analyze this in Section 5.3 with and w/o answer gating to show that our proposed answer gating helps in reducing such mistakes.

	Hits@1	**Hits@2**	**Hits@3**
EXPLICIT			
EXAQT	0.568	0.602	0.618
TwiRGCN	**0.602**	**0.618**	**0.628**
IMPLICIT			
EXAQT	0.512	0.575	0.612
TwiRGCN	**0.603**	**0.622**	**0.637**
ORDINAL			
EXAQT	0.42	0.47	0.49
TwiRGCN	**0.518**	**0.542**	**0.553**

Table 8: Effects of increasing k for Hits@k. As discussed in Section 5.4, TwiRGCN significantly outperforms EXAQT across categories of questions even as k is increased.

Temporal (Hits@1)	**±0**	**±1**	**±3**
EXAQT	0.642	0.653	0.667
TwiRGCN (average)	0.641	0.649	0.671
TwiRGCN (interval)	**0.646**	**0.659**	**0.682**

Table 9: Effects of increasing the answer temporal window on model performance for Temporal type questions. As discussed in Section 5.4, TwiRGCN (interval) gets even more accurate relative to EXAQT as we increase the temporal tolerance window.

ditionally, for 51.5% of the questions the distance is ≤5 years, while it is ≤20 years for ∼75% of the questions. Our simple-to-learn q_t which is just a linear transform of q_B works reasonably well. Better q_t would result in even more performance improvements. We leave that for future work.

Dominant errors: We do an error analysis over quantity-type questions, a challenging query class. Neither EXAQT nor TwiRGCN perform well on quantity-type questions. Out of a total of 224 quantity questions, EXAQT gets 0.1% accuracy while TwiRGCN gets 0.05%. This is because current TKGQA models treat quantities such as "2.55" or "16,233" as independent entities, instead of scalar numeric values. Additionally, from Section 5.3, we reconfirm that current TKGQA models fail on this bucket, so future work can direct special attention here. Examples: "*What was Panama's fertility rate in 2006?*" A: 2.55; "*What was the population of Bogota in 1775?*" A: 16,233.

Reducing answer type mistakes: In this study, we estimate TwiRGCN's propensity to make answer type mistakes. We define these mistakes as questions where the answer was an entity, but our model predicted a time or vice versa. From Table 7 we see that our answer gating mechanism mentioned in Eqn. (5) helps reduce such mistakes. For TwiRGCN (average), gating cuts entity-to-time mistakes by more than half.

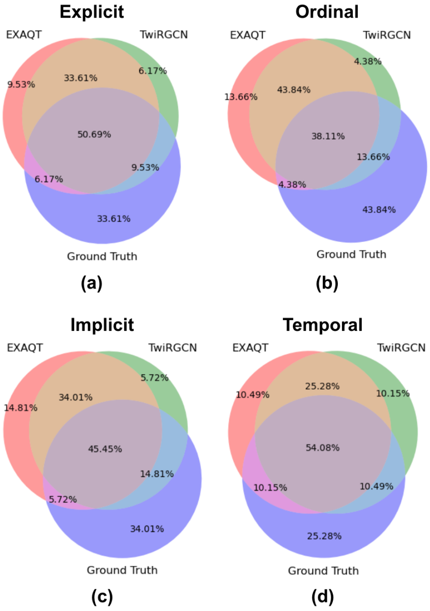


Figure 4: Venn diagrams for the prediction overlap of EXAQT, ground truth and our best model for each category. As described in Section 5.4, for Explicit, Implicit, and Ordinal question types TwiRGCN gives the right answers for most questions that EXAQT answers correctly, while correctly answering a much larger set that EXAQT gets wrong.

Increasing k for Hits@k: We extend the analysis in Table 3 increasing k from 1 to 3 for Hits@k on *TimeQuestions*. From Table 8 we see that the performance of TwiRGCN is robust to increasing k. It significantly outperforms EXAQT across categories even as k is increased for Hits@k.

Increasing temporal tolerance window: In Table 9, we explore the effects of increasing the time window for marking an answer correct for temporal questions. This means if the ground truth answer is 1992, and the predicted answer is 1990 for a question, it will be marked as incorrect in the ±1 column and correct in the ±3 column. We find that our model, specifically TwiRGCN (interval) gets even more accurate relative to EXAQT as we increase the temporal tolerance window. This implies that TwiRGCN is robust at ranking gold answers high up, even if they do not achieve rank 1.

Prediction overlap: We study the overlap of predictions between EXAQT, TwiRGCN, and ground truth. As seen in Figure 4, for Explicit, Implicit, and Ordinal question types our model gives the right answers for most questions that EXAQT answers correctly (missing less than 6% on average), while correctly answering a much larger set that EXAQT gets wrong. This split is more even between the two models for the temporal-type questions.

6 Conclusion

In this paper, we proposed TwiRGCN, a TKGQA system that employs a novel, temporally weighted graph convolution for answering questions that require complex temporal reasoning over a TKG. TwiRGCN modulates the convolutional messages through a TKG edge based on the relevance of the edge time interval to the question. We present two temporal weighting schemes with complementary strengths, intuitively explained through their simple formulations. We also propose an answer gating system for incorporating the pooled entity and time embeddings from TwiRGCN in the prediction, based on the likelihood that the answer is a time or an entity, given the question. Despite its relative simplicity, TwiRGCN gives significantly superior TKGQA accuracy on a challenging dataset compared to more heavily engineered baselines.

Acknowledgements

We thank Jeremy Cole and Srini Narayanan for their valuable feedback. We also thank the anonymous reviewers for their constructive comments. Soumen Chakrabarti was supported in part by grants from Amazon, Google, IBM, and SERB.

7 Limitations

TwiRGCN is limited by the need for relevant subgraphs for each question to be provided in the dataset. Such subgraphs have been provided in the *TimeQuestions* dataset used in the current work, but that may not be true for all TKGQA datasets. This limitation may be addressed for datasets that do not provide subgraphs through recently proposed subgraph selection methods (Chen et al., 2022; Shang et al., 2022; Jia et al., 2021), but we leave that exploration for future work.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. ComQA: A community-sourced dataset for complex factoid question answering with paraphrase clusters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 307–317, Minneapolis, Minnesota. Association for Computational Linguistics.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514, Osaka, Japan. The COLING 2016 Organizing Committee.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Nikita Bhutani, Xinyi Zheng, and HV Jagadish. 2019. Learning to answer complex questions over knowledge bases with query composition. In *CIKM*.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433.

Ziyang Chen, Xiang Zhao, Jinzhi Liao, Xinyi Li, and Evangelos Kanoulas. 2022. Temporal knowledge graph question answering via subgraph reasoning. *Knowledge-Based Systems*, page 109134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II*, pages 69–78.

Prachi Jain, Sushant Rathi, Soumen Chakrabarti, et al. 2020. Temporal knowledge base completion: New algorithms and evaluation protocols. *arXiv preprint arXiv:2005.05035*.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018a. Tempquestions: A benchmark for temporal question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1057–1062, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018b. Tequila: Temporal question answering over knowledge bases. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 1807–1810, New York, NY, USA. Association for Computing Machinery.

Zhen Jia, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. 2021. Complex temporal question answering on knowledge graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 792–802.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *International Conference on Learning Representations*.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194, Brussels, Belgium. Association for Computational Linguistics.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Soji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2021. Tempoqr: Temporal question reasoning over knowledge graphs.

Soumajit Pramanik, Jesujoba Alabi, Rishiraj Saha Roy, and Gerhard Weikum. 2021. UNIQORN: Unified Question Answering over RDF Knowledge Graphs and Natural Language Text. In *arXiv*.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021. Question answering over temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6663–6676, Online. Association for Computational Linguistics.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Chao Shang, Guangtao Wang, Peng Qi, and Jing Huang. 2022. Improving time sensitivity for question answering over temporal knowledge graphs. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8017–8026, Dublin, Ireland. Association for Computational Linguistics.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. 2016. On generating characteristic-rich question sets for qa evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 562–572.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *EMNLP-IJCNLP*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *EMNLP*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Komal Teru, Etienne Denis, and Will Hamilton. 2020. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

TwiRGCN: Temporally Weighted Graph Convolution for Question Answering over Temporal Knowledge Graphs
(Appendix)

A Additional Analyses

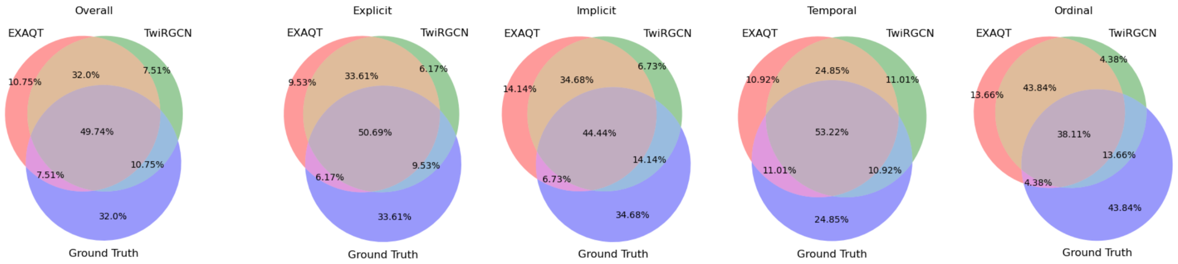
A.1 Complete prediction overlap

In Figure 5, we extend our analysis in 5.4 by providing the complete prediction overlap for both our models with EXAQT and ground truth across all question categories in *TimeQuestions*.

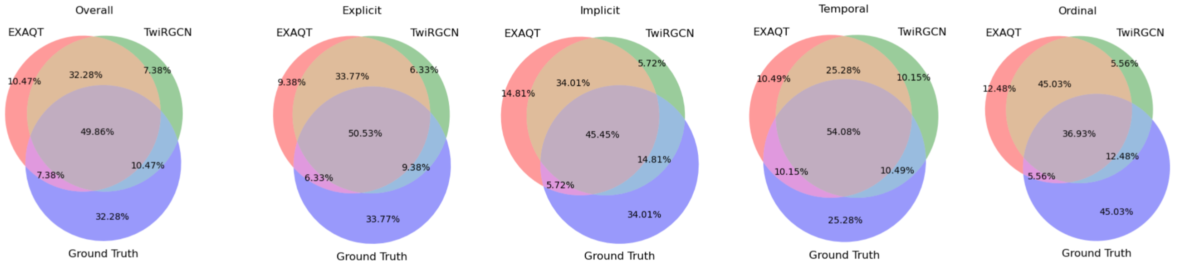
B Hyperprameters

We use the following hyperparameters:
- Number of layers, $L = 2$
- $c_d = 3$
- train batch size = 32
- valid batch size = 5
- LR = 0.00004
- Decay for LR = 0.4 every 10 epochs
- Cosine distance scaling constant for training (described in Section 4) = 30

Model and program execution details:
- Number of parameters = 2,223,833
- 11GB Nvidia GPU used with cudatoolkit 11.1
- Time per training epoch = 1:04 min
- Number of epochs to convergence on average = 50
- Early stopping used and implemented in code with patience = 10
- Validation overall Hits@1 for TwiRGCN (average) = 0.606
- Validation overall Hits@1 for TwiRGCN (intervall) = 0.602
- Performance is fairly stable around current hyperparameters without much tuning, except for LR decay rate. We used around 5–7 training runs with different decay settings to get the current rate. TwiRGCN is stable around current settings.
- Hyperparameters were tuned by manually inspecting loss behavior. Final values were selected based on a sustained, stable good performance on the test set for 3 runs.



(a) TwiRGCN (average)



(b) TwiRGCN (interval)

Figure 5: Venn diagrams for the prediction overlap of EXAQT, ground truth, and our two models TwiRGCN (average) in (a) and TwiRGCN (interval) in (b), as discussed in Appendix A.1.

ZELDA: A Comprehensive Benchmark for Supervised Entity Disambiguation

Marcel Milich
Humboldt-Universitat zu Berlin
marcelmilch@gmx.de

Alan Akbik
Humboldt-Universitat zu Berlin
alan.akbik@hu-berlin.de

Abstract

Entity disambiguation (ED) is the task of disambiguating named entity mentions in text to unique entries in a knowledge base. Due to its industrial relevance, as well as current progress in leveraging pre-trained language models, a multitude of ED approaches have been proposed in recent years. However, we observe a severe lack of uniformity across experimental setups in current ED work, rendering a direct comparison of approaches based solely on reported numbers impossible: Current approaches widely differ in the data set used to train, the size of the covered entity vocabulary, and the usage of additional signals such as candidate lists. To address this issue, we present ZELDA, a novel entity disambiguation benchmark that includes a unified training data set, entity vocabulary, candidate lists, as well as challenging evaluation splits covering 8 different domains. We illustrate its design and construction, and present experiments in which we train and compare current state-of-the-art approaches on our benchmark. To encourage greater direct comparability in the entity disambiguation domain, we open source our benchmark at https://github.com/flairNLP/zelda.

1 Introduction

Entity disambiguation (ED) is the task of disambiguating textual mentions of entities to a corresponding unique entry in a knowledge base. For instance, the entity mention "NBA" might refer to one of several organizations with this abbreviation, such as "National Basketball Association" or "National Boxing Association". ED resolves these ambiguities and creates links between a knowledge base of unique entities and the various ways an entity may be referred to in text. It is the core component in the larger task of entity linking (EL), which includes the identification of entity mentions in text, often handled by a named entity recognition (NER) system.

Recent progress in the field is driven by advances in large language models (Shen et al., 2021; Sevgili et al., 2022), pushing the scores on standard evaluation datasets to new heights. These models are typically trained in a supervised manner. Unlike many other NLP tasks with relatively few target classes, such as sentiment analysis or part-of-speech tagging, ED may have millions of target classes, since each entity in a knowledge base is modeled as a distinct class. Accordingly, most current state-of-the-art ED approaches are trained over very large amounts of annotated text data that often is automatically derived from Wikipedia.

Lack of uniformity in experimental setup. However, while a number of standard evaluation datasets exist to measure final ED accuracy, such as the AIDA-B test split of the popular AIDA dataset for newswire data (Hoffart et al., 2011), we observe that no such standardization exists for the data used to train ED systems. To illustrate this disparity, refer to Table 1 for an overview of current state-of-the-art approaches, published numbers and their respective training setups.

As Table 1 shows, approaches use different amounts of training data (ranging from 2 to 20 million "snippets" of annotated text), sourced from different Wikipedia versions using different sampling methodologies, and in some cases augmented with weak labels. Importantly, there is a stark difference in the size of the entity vocabulary for which approaches are trained, ranging from models that disambiguate a few thousand entities to models that handle over 6 million. Approaches also typically leverage so-called "candidate lists" that contain all possible disambiguation targets for textual mentions and so greatly narrow the search space. Prior work (see Section 2) has shown that each of these factors greatly influences the accuracy of an otherwise identical ED system (Broscheit, 2019; Wu et al., 2020; Févry et al., 2020; Orr et al., 2021; De Cao et al., 2021).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2061–2072
May 2-6, 2023

Approach	Training data	Weak labels?	Data source	Additional features	Entity vocab	Candidate lists	AIDA-B
Yamada et al. (2022)	∼10M snippets	no	Wikipedia (Dec, 2018)				
- LUKE$_{GH}$					128k	GH	92.4
- LUKE$_{PPR}$					128k	PPR	94.6
- LUKE$_{GH+AIDA}$				+AIDA-TRAIN	128k	GH	95
- LUKE$_{PPR+AIDA}$				+AIDA-TRAIN	128k	PPR	97.1
Barba et al. (2022)	9M snippets†	no	Wikipedia (May, 2019)				
- EXTEND				+AIDA-TRAIN	1.5M	GH	92.6
Ayoola et al. (2022)	∼20M snippets	yes	Wikipedia (July, 2021)	+KB +descriptions +types			
- AYOOLA					6.2M	GH	90.4
De Cao et al. (2021)	9M snippets†	no	Wikipedia (May, 2019)				
- GENRE					1.5M	GH	89.3
- GENRE$_{+AIDA}$				+AIDA-TRAIN	1.5M	GH	93.3
- GENRE$_{+AIDA-NOC}$				+AIDA-TRAIN	1.5M	*none*	91.2
Orr et al. (2021)	5.7M sentences	yes	Wikipedia (Nov, 2019)	+KB +types			
- BOOTLEG				+AIDA-TRAIN	3.3M	PPR+*custom*	96.7
Févry et al. (2020)	17.5M snippets	no	Wikipedia (Apr, 2019)	+AIDA-TRAIN			
- FEVRY$_{HF}$					5.7M	HF+*custom*	92.5
- FEVRY$_{PPR}$					5.7M	PPR+*custom*	96.7
Broscheit (2019)		yes	Wikipedia (June, 2017)	+AIDA-TRAIN			
- BROSCHEIT$_{700k}$	8.8M snippets				700k	*none*	78.8
- BROSCHEIT$_{500k}$	2.4M snippets				500k	*none*	87.9

Table 1: Differences in the signal used to train current state-of-the-art ED approaches and their reported accuracy on the AIDA-B evaluation dataset. Differences include: the number of snippets used to train each approach, the definition of what constitutes a "snippet", the Wikipedia version the data is sourced from, and -importantly- the size of the entity vocabulary and quality of the candidate lists used ("HF" are lists by Hoffart et al. (2011), "GH" lists by Ganea and Hofmann (2017), and "PPR" lists by Pershina et al. (2015)).

Lack of direct comparability. With this paper, we argue that these differences in training setup impair our ability to directly compare approaches based solely on published numbers on evaluation datasets. For instance, Table 1 shows that LUKE (Yamada et al., 2022) slightly outperforms the comparatively simple approach by FEVRY (Févry et al., 2020) on AIDA-B; but since FEVRY is trained to cover a much larger set of entities, we cannot know if the difference in evaluation score is due to algorithmic differences in both approaches, or simply a function of the signal used to train them.

Contributions. We argue that -much like in most other NLP tasks- we require a uniform experimental setup to evaluate large ED models. To this end, we present ZELDA, a comprehensive benchmark for supervised entity disambiguation. The benchmark consists of 95k full text paragraphs from Wikipedia, annotated with mention boundaries and disambiguation targets, and integrates 8 existing ED datasets from various domains as evaluation splits. ZELDA defines a fixed entity vocabulary of 822k entities, together with fixed candidate lists and entity descriptions. In this paper:

1. We analyze training setups in recent state-of-the-art ED approaches, and derive desiderata for a uniform training benchmark (Section 2).
2. We present the ZELDA benchmark, the design goals that inform our sampling methodology to create it, and its properties (Section 3).
3. We compute evaluation scores for baselines and three state-of-the-art approaches to present standardized scores and illustrate the usefulness of our benchmark (Section 4).
4. We make our benchmark available to the research community as an open source project.

We hope that the public release of ZELDA will encourage future ED works to leverage our benchmark, and thus facilitate greater direct comparability of future ED approaches.

2 Analysis of Training Setups

ED approaches employ large language models to embed an entity mention, its textual context and additional features. To decode, approaches typically either use variants of softmax classification (Broscheit, 2019; Févry et al., 2020; Yamada et al., 2017; Orr et al., 2021; Ayoola et al., 2022), generative decoding (De Cao et al., 2021) or retrieval-based models that compute the pairwise similarity of an embedded mention and a textual description for each target entity (Ravi et al., 2021).

Rather than focus on the algorithmic differences of these approaches, this section analyzes current state-of-the-art approaches from the point of view of their respective training setups.

2.1 Training Data

Size of training dataset. Approaches are typically trained over short snippets of text with annotated entity mentions, derived from Wikipedia page links. The number and length of these snippets varies greatly across approaches. For instance, as Table 1 shows, Ayoola et al. (2022) train their model with 20 million snippets of 512 tokens length, while De Cao et al. (2021) train with 9 million snippets of 100 token length. Orr et al. (2021) train on single sentences only, and use a comparatively small set of 5.7 million snippets. The sampling methodology to derive these snippets from Wikipedia is seldom described in detail and bespoke to each paper.

While we could find few ablation experiments in prior work, Broscheit (2019) presents an experimental evaluation in which he trains his proposed approach over two different datasets sampled from Wikipedia, one with 8.8 million and one with 2.4 million snippets. The difference in dataset size is due to a threshold parameter for frequent entities in his sampling method. Surprisingly, he finds that the model trained on the smaller dataset yields significantly better results on AIDA-B. He believes this may be because his computational resources limited training on the large dataset to only 4 epochs, whereas on the small dataset he could train for 14.

Single-mention vs multi-mention data. The number of snippets is only partly illustrative of the training signal, as the number of mentions dramatically differs per setup. In "single-mention" data as used by Barba et al. (2022) and De Cao et al. (2021), each snippet only contains a single annotated entity mention (indicated with a *dagger* asterisk in Table 1). In "multi-mention" data on the other hand, more than one mention might be annotated, thus potentially greatly increasing the training signal.

Optional augmentation with weak labels. One particularity of Wikipedia text is that within an article, usually only the first mention of an entity is marked with a page link. In fact, Orr et al. (2021) estimate that 68% of mentions are unlabeled. For this reason, many works use "weak labeling" methods to annotate unlabeled mentions (Orr et al., 2021; Ayoola et al., 2022; Broscheit, 2019) in Wikipedia articles. These methods dramatically increase the number of labeled mentions per text snippet, but may introduce errors into the training data. This naturally impacts the performance of ED: for instance, Orr et al. (2021) find that their model performs better on rare and unseen entities but worse on frequent ones when using weak labels.

Wikipedia version of training data. Table 1 also shows that the data is sourced from different Wikipedia versions. This poses problems as the entity set covered by Wikipedia grows significantly over time (Gillick et al., 2019). Further, the information contained in Wikipedia is constantly updated (such as which person currently holds which political office), potentially giving advantages to models trained on a Wikipedia version from a similar point in time as the evaluation data.

2.2 Entity Vocabulary

As Table 1 shows, published approaches also differ in their entity vocabulary, i.e. the number of unique entities they can resolve, ranging from 128k (Yamada et al., 2017) to about 6 million entities (Févry et al., 2020; Ayoola et al., 2022). While a very large entity set is desirable for a general-purpose ED system, a smaller vocabulary tuned to an evaluation dataset will likely result in better evaluation numbers. This intuition is supported by experiments by Wu et al. (2020) who found that a model trained to handle an entity set specific to their evaluation dataset outperforms a general-purpose model trained to handle 5.9M Wikipedia entities.

2.3 Candidate Lists

Most state-of-the-art ED approaches employ *candidate lists* that contain for each mention string a set of sensible entity candidates (Sevgili et al., 2022).

	Domain	Doc. Type	# Docs	∅ length	# Entities	# Mentions
Evaluation Splits						
AIDA-B	news	articles	231	201 tokens	1,538	4,485
TWEEKI	tweets	short texts	500	16 tokens	639	860
REDDIT-POSTS	forum posts	short texts	377	20 tokens	524	705
REDDIT-COMMENTS	forum comments	short texts	360	41 tokens	483	638
WNED-WIKI	Wikipedia	articles	318	315 tokens	5,293	6,747
WNED-CWEB	web	pages	320	1,433 tokens	4,467	11,116
SLINKS-TOP	web	short texts	904	35 tokens	899	904
SLINKS-SHADOW	web	short texts	904	35 tokens	902	904
SLINKS-TAIL	web	short texts	902	35 tokens	902	902
Training Split						
ZELDA-TRAIN	Wikipedia	paragraphs	95k	527 tokens	822k	2.6M

Table 2: Descriptive statistics of the training and evaluation splits of ZELDA. Note that the statistics reported for evaluation splits may differ slightly from previous literature as a consequence of our normalization procedure.

The model then classifies over the small list rather than the whole entity set. The advantage of this approach is that it greatly narrows the search space and speeds up computation. A drawback however is that these candidate lists must be created separately and that incomplete lists lower the upper bound of what an ED approach can achieve: if the correct entity is not included in the candidate list of a mention, correct classification is not possible.

Prior work showed that the choice of candidate lists significantly impacts overall results. For instance, the lists of Pershina et al. (2015) were found to be exceptionally well-tailored to the AIDA-B evaluation dataset, with high recall and low ambiguity for its entities (Yang et al., 2018). As Table 1 shows, both Févry et al. (2020) and Yamada et al. (2022) find that their models improve significantly when using these lists instead of the more generic lists by Hoffart et al. (2011) and Ganea and Hofmann (2017) respectively. Unfortunately, some approaches such as Févry et al. (2020) also employ custom lists that are not released.

2.4 Domain-Specific Features

It is possible to tailor ED systems to achieve better results on individual domains.

Page titles. Févry et al. (2020) and Orr et al. (2021) disambiguate entities in news articles, and present a custom approach for constructing snippets: instead of only taking a token window around an entity mention, they also add the title and first two sentences of the article as additional context, reasoning that these texts contain salient information that pertains to the whole article. However, such custom contexts can only be defined for individual domains (e.g. tweets for instance do not have titles) and are therefore challenging to integrate for general-purpose ED systems.

Domain-specific data. In addition to large Wikipedia-derived datasets, many works also incorporate domain-specific data into their training or fine-tuning. While many available evaluation datasets are limited to small test splits, some popular datasets also define training splits. A well-known example is AIDA, which next to AIDA-B defines a train split consisting of 20k sentences and covering 30k entities. Table 1 shows (in column "additional features") that all models either present ablations in which AIDA-TRAIN is included, or include this data by default. The numbers clearly show that including domain-specific data improves overall results. However, prior works have shown that fine-tuning to a particular domain degrades performance on other datasets (Yamada et al., 2022; De Cao et al., 2021; Le and Titov, 2019).

2.5 Additional Features

Some ED approaches leverage additional sources of information (Shen et al., 2021; Sevgili et al., 2022). In particular, *entity descriptions* are concise textual summaries of the "meaning" of each entity, and a core component of all ED approaches that follow a retrieval-based approach (Ravi et al., 2021). *Entity type information* equips each entity with semantic type as additional signal. Finally, some approaches employ *knowledge base (KB)* information to decode multiple mentions in a text paragraph such that overall entity relatedness is observed (Ayoola et al., 2022; Orr et al., 2021).

3 The ZELDA Benchmark

We create ZELDA to enable analysis and direct comparison of large ED models. Refer to Table 2 for an overview. We start by selecting and normalizing appropriate evaluation datasets (Section 3.1), upon which we define a methodology to sample training data that satisfies several objectives (Section 3.2). To ensure broad applicability, we also produce candidate lists and entity descriptions (Section 3.3).

3.1 Selection of Evaluation Splits

Desiderata. Our analysis of Section 2 showed that there are many ways to tailor the training setup to a specific evaluation dataset: one might employ domain-optimized candidate lists, include domain-specific features, use an optimized entity vocabulary, or optimize the process of sampling Wikipedia for training data. With ZELDA, we seek to minimize opportunities for such domain-specific engineering to place greater focus on evaluating algorithmic rather than engineering components. We also seek an evaluation setup that not only produces a single score, but facilitates more granular analysis of the capabilities of large ED models.

We therefore sought evaluation datasets that both span a broad range of domains (web pages, newswire text, social media) as well as isolate specific challenges in ED. To facilitate distribution, we limited our search to freely available datasets.

Selected datasets (Table 2). We chose the following 8 datasets for inclusion:

- AIDA-B is the test split of AIDA, the most commonly used ED dataset. It contains 231 manually annotated Reuters news articles.

- TWEEKI (Harandizadeh and Singh, 2020) is a collection of 500 randomly selected and hand-annotated tweets.

- Two datasets from Botzer et al. (2021), referred to as REDDIT-COMMENTS and REDDIT-POSTS respectively, that consist of top-scoring posts and comments from the internet forum Reddit. We use the "gold" subset of this dataset, i.e. all annotations in which all three annotators agreed.

- Two datasets from Guo and Barbosa (2018), referred to as WNED-WIKI and WNED-CWEB respectively, that cover the domains of Wikipedia articles and web pages. We include these datasets because they include annotation of the difficulty of each document on a scale from 0 to 1. This enables analyses of approaches as a function of estimated difficulty, as we show in Section 4.

- Three datasets from Provatorova et al. (2021), created specifically to analyze three classes of mention ambiguities: (1) SLINKS-TOP contains only easy cases in which the correct disambiguation is the most frequent sense of a mention. (2) SLINKS-SHADOW is the opposite and contains only difficult cases in which the correct disambiguation of a mention is "overshadowed" by a more popular entity. (3) SLINKS-TAIL contains only "long tail" entities that are very rare in Wikipedia.

Normalization. We unify these datasets in two ways: First, since these datasets were created at different times, we update entity annotations to the most recent version of Wikipedia (October, 2022). Second, as datasets are provided in various formats, we convert them into two commonly used standard formats, namely CoNLL and JsonL.

3.2 Training Data

Desiderata. We define a sampling methodology to create training data to balance two objectives: our first goal is to evaluate entity disambiguation for "broad entity coverage" approaches that derive large-scale training data from Wikipedia. However, if the training data is too large, model training becomes to costly for thorough analyses of design choices and hyperparameters; with the exception of the work by Févry et al. (2020), we find such analyses to be rare in current literature. For this reason, our second goal is to limit the overall size of the training dataset.

Sampling process (Algorithm 1). To balance these two goals, our sampling process starts from the vocabulary of all entities in the evaluation splits, which we refer to as the test entity set E_t. Our sampling seeks to find at least a minimal number of training examples for these entities, set by the *threshold* parameter. Following prior analyses by Vasilyev et al. (2022), we set the threshold to 10, meaning that each entity in the test set should appear at least 10 times in the training data. However, this is only possible for entities that do appear this often in the source Wikipedia data; for long-tail entities that appear fewer times, we select as many examples as possible.

Our sampling selects entire Wikipedia paragraphs for inclusion into the training dataset. The reason for choosing paragraphs as atomic document type is threefold: (1) Unlike fixed-length token windows that center on one particular entity, paragraphs typically consist of multiple full sentences that provide natural context for entity mentions. (2) By choosing random paragraphs instead of full articles, we limit overall dataset size and introduce more textual variety as opening paragraphs of Wikipedia articles were observed to often have similar wording (Le and Titov, 2019). (3) Paragraphs contain mentions to many other entities outside of E_t. These entities are naturally skewed and added to the overall entity vocabulary of ZELDA.

Data preprocessing. We leverage the Kensho Derived Wikimedia Dataset[1], derived from the Wikipedia dump of December 2019. This dataset is preprocessed such that redirect-, disambiguation- and list-pages are removed, the text is cleaned and articles are divided into sections. Here, each section corresponds to one paragraph of text. We discard common section types that typically contain little text (such as the "Bibliography" and "External Links" sections common to Wikipedia articles). To ensure that all annotations are consistent with our evaluation splits, we update entity annotations to the most recent version of Wikipedia and discard those for which no article exists anymore.

Resulting training data. This set is randomized and sampled using Algorithm 1, yielding a training data set of 95k paragraphs spanning on average 527 tokens. It contains a total of 2.6 million mentions covering a vocabulary of 825k distinct entities, which we refer to as ZELDA-TRAIN. See Table 2 for descriptive statistics.

3.3 Additional Structured Information

We provide *candidate lists* that we derive with a general approach from the Kensho Wikimedia dataset, the Wikilinks web corpus (Singh et al., 2012) and the "also known as" information from Wikidata. For all mentions in these sources we list and count all entities that they refer to and filter entities from these lists that are not contained in the ZELDA entity vocabulary (details in Appendix A.3). Moreover we derive the most-frequent-sense baseline (MFS) by choosing, for every mention, the entity that this mention refers to the most often.

We also provide standardized *entity descriptions*: For each entity we extract the opening paragraph of its Wikipedia article as its description.

4 Experiments

We showcase the ZELDA benchmark by training a set of baselines and state-of-the-art approaches on ZELDA-TRAIN, and comparatively evaluating them on our evaluation splits.

Algorithm 1: Paragraph sampling

```
input  : Set of sections $S$, test entity set $E_t$
output : Filtered list of sections $\hat{S}$

$threshold \leftarrow 10$;
$counter_e \leftarrow 0$ for $e$ in $E_t$;
while $E_t \neq \emptyset$ do
    $s$ = random.sample($S$);
    if $E_t \cap s.links \neq \emptyset$ then
        $\hat{S}$.add($s$);
        for $e$ in $s.links$ do
            $counter_e \leftarrow counter_e + 1$;
            if $counter_e \geq threshold$ then
                $E_t$.remove($e$)
            end
        end
    end
end
return $\hat{S}$
```

4.1 Evaluated Approaches

We compare 8 different models, as listed in Table 3:

Simple baselines. We include two baseline approaches. The first is MFS, a simple most-frequent-sense baseline that assigns each mention to its most commonly observed entity. The second is CL-RECALL, which calculates the upper bound reachable with our provided candidate lists: for each mention, the gold entity is assigned if it is included in the candidate list.

Simple softmax classifier (FEVRY). We include a reimplementation of the approach by Févry et al. (2020) in two variants: FEVRY_{CL} uses our candidate lists, while FEVRY_{ALL} does not use any lists to restrict the search space. The approach leverages a simple softmax classification head trained on top of a transformer model that takes as input a text snippet. Despite its simplicity, it was found to be surprisingly competitive. We reimplemented the approach as Févry et al. (2020) did not release their code. However, since our train set is much smaller we use `bert-base-uncased` instead of just a 4-layer transformer and adapt the hyperparameters to our setting (see Appendix A.1)

LUKE. We train two variants of LUKE (Yamada et al., 2022), the current state-of-the-art approach for several benchmark datasets. It trains a language model with entity embeddings, and employs a global decoding mechanism to decode mentions

---

[1] https://datasets.kensho.com/datasets/wikimedia

	AIDA-B	TWEEKI	REDDIT-POSTS	REDDIT-COMM.	WNED-CWEB	WNED-WIKI	SLINKS-TAIL	SLINKS-SHADOW	SLINKS-TOP	∅
Baselines										
MFS	0.635	0.723	0.834	0.81	0.612	0.651	0.994	0.149	0.413	0.647
CL-RECALL	0.911	0.94	0.984	0.983	0.924	0.988	0.988	0.567	0.731	0.891
Classification										
FEVRY$_{ALL}$	0.792	0.718	0.885	0.841	0.68	0.843	0.638	**0.434**	**0.531**	0.707
FEVRY$_{CL}$	0.795	0.769	0.89	0.865	0.703	0.845	0.876	0.319	0.477	0.727
LUKE$_{PRE}$	0.793	0.738	0.761	0.699	0.668	0.684	0.977	0.204	0.508	0.670
LUKE$_{FT}$	**0.812**	0.779	0.815	0.785	0.703	0.765	0.980	0.225	0.518	0.710
Generative										
GENRE$_{ALL}$	0.724	0.759	0.888	0.839	0.665	0.852	0.953	0.387	0.435	0.722
GENRE$_{CL}$	0.786	**0.801**	**0.928**	**0.915**	**0.736**	**0.884**	**0.996**	0.373	0.528	**0.772**

Table 3: Results of our experiments. Bold scores indicate the best scores of all the trained models. Underlined scores represent the best scores among the classification-based models.

in a given text snippet by order of confidence. Each classified mention is used as a feature to better classify the remaining mentions in a snippet. For training, they distinguish between *pre-training* in which entity embeddings are learned, and an optional final epoch of *fine-tuning* in which they are frozen. We train one model only with pre-training (LUKE$_{PRE}$) and one with fine-tuning (LUKE$_{FT}$).

We use their publicly available code[2] to train our two models. For direct comparison to the FEVRY model, we use `bert-base-uncased` instead of `bert-large-uncased` (utilized in Yamada et al. (2022)) and slightly adapt their hyperparameters to our setting, see Appendix A.1.

GENRE. We also train two variants of GENRE (De Cao et al., 2021), a generative approach that formulates ED as a sequence-to-sequence problem. A given input text with flagged mention boundaries is input, from which an entity title is generated. To ensure that the generated sequence is a valid title, GENRE uses a prefix-tree generated from all entity titles in the data to constrain the generation process. GENRE does not use candidate lists during training but in inference the prefix tree can be derived from the candidate lists. We call this variant GENRE$_{CL}$.

We use their publicly available code[3] to train our two models. Instead of the `bart-large` model we use the `bart-base` version to make the comparison more fair. We adapt the recommended hyperparameters to our setting (see Appendix A.1).

4.2 Results

Table 3 breaks down the accuracy of each model for each of the evaluation splits, and provides a single macro-averaged accuracy score for all data. We make a number of interesting observations:

Different ranking of approaches. Most importantly, we arrive at a starkly different ranking of approaches compared to published numbers on AIDA-B as listed in Table 1 where GENRE is one of the lowest-scoring models. In contrast, in our evaluation the two GENRE models clearly outperform all other considered models in most evaluation splits.

Impact of candidate lists. We note that our general-purpose candidate lists derived from Wikipedia score unevenly across domains. As our CL-RECALL baseline shows, our lists have a high upper bound on evaluation splits covering Wikipedia and social media domains, but a relatively low upper bound on splits from the domains of web pages or news text. We also note that of the classification-based approaches, only FEVRY$_{ALL}$ does not use candidate lists, but scores best.

Moreover, the overall best-scoring approach GENRE is trained without candidate lists. But during prediction, the better-scoring variant GENRE$_{CL}$ employs candidate lists, while GENRE$_{ALL}$ does not. This indicates using candidate lists only for prediction, but not training, may be a worthwhile approach to further explore.

Hard-to-disambiguate entities. On the SLINKS-SHADOW dataset of "overshadowed" entities, FEVRY$_{ALL}$ outperforms GENRE. One possible interpretation is that a generative approach naturally favors decoding into the most prominent sense, as the generated entity title will be most similar to the mention text. On the other hand, classification-based approaches are not influenced by string similarity of entity and mention text, potentially leading to better performance here.

In Table 4, we additionally list the scores on the brackets for WNED-WIKI provided by Guo and Barbosa (2018). The table shows that accuracy scores of all models steadily decrease from left (the

---

[2] https://github.com/studio-ousia/luke

[3] https://github.com/facebookresearch/GENRE

Approach	1	[1 - 0.9]	[0.9 - 0.8]	[0.8 - 0.7]	[0.7 - 0.6]	[0.6 - 0.5]	[0.5 - 0.4]	[0.4 - 0.3]	∅
CG-recall	0.997	0.972	0.992	0.983	0.983	0.991	0.988	0.996	0.988
FEVRY$_{CL}$	0.948	0.922	0.888	0.872	0.841	0.8	0.76	0.722	0.844
LUKE$_{FT}$	0.915	0.904	0.863	0.803	0.778	0.738	0.622	0.562	0.773
GENRE$_{CL}$	0.971	0.942	0.912	0.856	0.878	0.869	0.869	0.797	0.887

Table 4: Accuracy measured for best approaches on different *difficulty brackets* of WNED-WIKI. The lowest scores are observed for the most difficult bracket "[0.4 - 0.3]".

easiest bracket) to right (the hardest bracket). As we see this is not caused by a the CG-recall on WNED-WIKI which is independent from the brackets. This indicates that there remains much room for improving ED performance on ZELDA even when leveraging candidate lists during prediction.

4.3 Discussion

Our evaluation showed that the generative GENRE approach outperforms all classification-based approaches, and a simple direct classification approach without candidate lists as second-best performing approach overall. This indicates that equalizing the training signal, removing opportunities for domain-specific engineering, and evaluating across diverse evaluation splits may yield more insights into which algorithmic approach is best-suited to train large ED models.

However, we must also caution against overinterpreting this ranking: due to the large training times for each of these models, we did not explore any hyperparameters. Instead, we used default parameters whenever possible, and in the case of LUKE and FEVRY changed the underlying transformer to the same model, for more direct comparability. Models were only trained for as long as our computational resources allowed. Upon publication of the benchmark we anticipate that authors will explore better hyperparameters for their respective approaches, which may change the ranking (see Limitations section).

5 Related Work

Prior work has addressed aspects of evaluating ED.

Standardized evaluation. GERBIL (Röder et al., 2018) standardizes ED evaluation over multiple datasets in a unifying framework, but does not define the training data and thus only focuses on comparing already-trained models. Similarly, a range of prior works have sought to refine and standardize ED evaluation (Waitelonis et al., 2019; Nait-Hamoud et al., 2021; Noullet et al., 2021; Odoni et al., 2019; van Erp and Groth, 2020; Braşoveanu et al., 2018). In contrast, ZELDA defines the full experimental setup, including training data, the entity vocabulary and other training signals.

Manually labeled training data. A few existing ED datasets not only define a test set, but also a training split. An example discussed in this paper is the AIDA dataset. Other datasets include TAC-KBP2010 (Ji and Grishman, 2011), which is not available anymore, and a zero-shot dataset from Logeswaran et al. (2019). However, these datasets are too small and cover too few entities for evaluation of large ED approaches.

Deriving training data from Wikipedia. All current state-of-the-art approaches derive their training data from Wikipedia, though the exact process is often not thoroughly described and/or provided to the public. One exception is the BLINK corpus created by Wu et al. (2020) that is used in other works. However, this corpus consists only of single-mention snippets and cannot be used for approaches that train global decoders, like LUKE.

Most similar to our sampling method is from Orr et al. (2021). The authors sample a small Wikipedia subset by sampling for the mentions of the KORE50 benchmark (Hoffart et al., 2012). Unlike our approach, they sample all occurrences of each mention and sample only single sentences, yielding 520k sentences for only 144 mentions.

6 Conclusion

We presented the ZELDA benchmark to unify experimental setups across large ED approaches, and conducted an evaluation of various approaches. We find that given the exact same training signal, approaches compare differently than published numbers suggest. We release the datasets, our sampling and preprocessing scripts and our FEVRY reimplementation to the research community as an open source project available at `https://github.com/flairNLP/zelda`. Additionally, we integrate our benchmark into the open source NLP framework FLAIR (Akbik et al., 2019).

We hope that this will encourage present and future ED works to compare algorithmic differences on a more equal setting and thus help generate insights to further advance the field of ED.

7 Limitations

As discussed in Section 4.3, an important limitation of our experimental evaluation is our lack of hyperparameter exploration of published approaches. Given the effort required to train large ED models and the many involved hyperparameters, we believe that only the original authors of their respective approaches can perform a meaningful search of hyperparameters for our benchmark, limiting us to best-effort parameters from prior literature. It is therefore possible if not likely that the respective authors of the approaches we compare might arrive at better numbers than the ones presented here.

Regarding the ZELDA benchmark itself, we note that it is designed to evaluate supervised ED approaches. As we sampled the dataset to contain at least 10 annotations for each entity in the ZELDA test splits whenever possible, it is unclear whether ZELDA is useful for evaluating the currently growing family of zero-shot ED models (Logeswaran et al., 2019; Wu et al., 2020). Finally, our training corpus is relatively small compared to some other Wikipedia corpora used in prior approaches. While we made this design choice purposefully to enable faster training times and hopefully more exploration of hyperparameters by future works, we cannot be certain whether rankings obtained on ZELDA transfer to approaches trained on orders of magnitude of more data.

Acknowledgments

We thank the reviewers for their helpful comments. Marcel Milich is supported the Investitionsbank Berlin through research project "AI Marketeer", confinanced by the European Regional Development Fund (ERDF). Alan Akbik is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy "Science of Intelligence" (EXC 2002/1, project number 390523135) and the DFG Emmy Noether grant "Eidetic Representations of Natural Language" (project number 448414230).

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Tom Ayoola, Joseph Fisher, and Andrea Pierleoni. 2022. Improving entity disambiguation by reasoning over a knowledge base. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2899–2912, Seattle, United States. Association for Computational Linguistics.

Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2022. ExtEnD: Extractive entity disambiguation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2478–2488, Dublin, Ireland. Association for Computational Linguistics.

Nicholas Botzer, Yifan Ding, and Tim Weninger. 2021. Reddit entity linking dataset. *Information Processing & Management*, 58(3):102479.

Adrian M. P. Braşoveanu, Giuseppe Rizzo, Philipp Kuntschik, Albert Weichselbraun, and Lyndon J. B. Nixon. 2018. Framing named entity linking error types. In *LREC*.

Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Thibault Févry, Nicholas FitzGerald, Livio Baldini Soares, and Tom Kwiatkowski. 2020. Empirical evaluation of pretraining strategies for supervised entity linking. *CoRR*, abs/2005.14253.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web*, Preprint(Preprint):1–21.

Bahareh Harandizadeh and Sameer Singh. 2020. Tweeki: Linking named entities on Twitter to a knowledge graph. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 222–231, Online. Association for Computational Linguistics.

Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2012. Kore: Keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, page 545–554, New York, NY, USA. Association for Computing Machinery.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Phong Le and Ivan Titov. 2019. Boosting entity linking performance by leveraging unlabeled documents. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1935–1945, Florence, Italy. Association for Computational Linguistics.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

Mohamed Cherif Nait-Hamoud, Fedoua Lahfa, and Abdellatif Ennaji. 2021. A step further towards a consensus on linking tweets to wikipedia. *Evolutionary Intelligence*, pages 1–16.

Kristian Noullet, Samuel Printz, and Michael Färber. 2021. Clit: Combining linking techniques for everyone. In *ESWC*.

Fabian Odoni, Adrian M. P. Braşoveanu, Philipp Kuntschik, and Albert Weichselbraun. 2019. Introducing orbis: An extendable evaluation pipeline for named entity linking performance drill-down analyses. *Proceedings of the Association for Information Science and Technology*, 56.

Laurel J. Orr, Megan Leszczynski, Simran Arora, Sen Wu, Neel Guha, Xiao Ling, and Christopher Ré. 2021. Bootleg: Chasing the tail with self-supervised named entity disambiguation. *ArXiv*, abs/2010.10363.

Maria Pershina, Yifan He, and Ralph Grishman. 2015. Personalized page rank for named entity disambiguation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 238–243, Denver, Colorado. Association for Computational Linguistics.

Vera Provatorova, Samarth Bhargav, Svitlana Vakulenko, and Evangelos Kanoulas. 2021. Robustness evaluation of entity disambiguation using prior probes: the case of entity overshadowing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10501–10510, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Manoj Prabhakar Kannan Ravi, Kuldeep Singh, Isaiah Onando Mulang', Saeedeh Shekarpour, Johannes Hoffart, and Jens Lehmann. 2021. CHOLAN: A modular approach for neural entity linking on Wikipedia and Wikidata. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 504–514, Online. Association for Computational Linguistics.

Michael Röder, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. 2018. Gerbil - benchmarking named entity recognition and linking consistently. *Semantic Web*, 9:605–625.

Özge Sevgili, Artem Shelmanov, Mikhail Arkhipov, Alexander Panchenko, and Chris Biemann. 2022. Neural entity linking: A survey of models based on deep learning. *Semantic Web*, 13(3):527–570.

Wei Shen, Yuhan Li, Yinan Liu, Jiawei Han, Jianyong Wang, and Xiaojie Yuan. 2021. Entity linking meets deep learning: Techniques and solutions.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2012. Wikilinks: A large-scale cross-document coreference corpus labeled via links to Wikipedia. Technical Report UM-CS-2012-015.

Marieke van Erp and Paul Groth. 2020. Towards entity spaces. In *LREC*.

Oleg Vasilyev, Alex Dauenhauer, Vedant Dharnidharka, and John Bohannon. 2022. Named entity linking on namesakes.

Jörg Waitelonis, Henrik Jürges, and Harald Sack. 2019. Remixing entity linking evaluation datasets for focused benchmarking. *Semantic Web*, 10:385–412.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2022. Global entity disambiguation with BERT. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3264–3271, Seattle, United States. Association for Computational Linguistics.

Yi Yang, Ozan Irsoy, and Kazi Shefaet Rahman. 2018. Collective entity disambiguation with structured gradient tree boosting. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 777–786, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendix

A.1 Training Parameters and Times

Our training parameters are informed by recommended parameters of prior works, adapted to the smaller training dataset size of ZELDA-TRAIN. In some cases, we adapted parameters across approaches for greater comparability, for instance by using the same transformer model for both LUKE and FEVRY. Across all approaches, we use the following parameters: We train all models for 6 epochs with a mini-batch size of 64 and a learning rate of 5e-5. The remaining hyperparameters are specific to each model:

FEVRY. The remaining parameters in FEVRY follow the recommendations from the paper, i.e. we use the Adam optimizer (Kingma and Ba, 2015) with a linear warmup for the first 10% of training and gradient clipping. However, the smaller dataset allowed us to look at more context. We split paragraphs of ZELDA into snippets of 400 tokens (Fevry: 256) and use an entity embedding size of 200 (Fevry: 256). Training FEVRY$_{ALL}$ took around 5h per epoch and for FEVRY$_{CL}$ 2.5h per epoch on a single Nvidia 3090ti GPU, respectively.

LUKE. We run experiments with both one-stage and two-stage training in LUKE. In one-stage training, we use for all epochs the same learning rate and do not fix the transformer weights. The entity masking rate is set to 30%. In two-stage training (LUKE$_{CL}$), we do fine-tuning in the last training epoch where we fix the entity embeddings and set the entity masking rate to 90%. To ensure comparability to FEVRY we set the entity embedding size to 200. For the remaining parameters we stick to the ones of the original LUKE which can be found in detail in table 4 and 5 of Yamada et al. (2022). Paragraphs of ZELDA are divided into snippets with ≤ 512 tokens. Training LUKE took roughly 2h per epoch on a single Nvidia 3090ti GPU.

GENRE. Apart from the parameters that we already discussed, we take all default parameters from the original paper. The parameters can best be found in the released code[4]. Since GENRE takes much longer to train than the other models and processes mentions individually we gave the model less context: We split context 500 chars to the left and 500 chars to the right of each mention (a context of roughly 190 tokens). At inference we use a beam size of 10 and a maximum number of 15 decoding steps as in the original paper. Training GENRE took around 16 hours on two Nvidia 3090ti GPUs per epoch.

A.2 Model Parameters

Our models have the following number of parameters: Both LUKE and FEVRY use a `bert-base-uncased` transformer model (110M parameters), a projection layer (768 × 200 ≈ 153k parameters) and the entity embedding layer (200 × 825k ≈ 165M parameters), and thus have about 274M parameters in total. GENRE adds a decoder with 768 × 51197 ≈ 39M parameters to its underlying transformer and thus has a total of 178M parameters.

A.3 Candidate Lists

We derive the candidate lists with a straightforward approach from three sources: Wikipedia Kensho, WikiLinks and Wikidata. The first two are text corpora with entity annotations derived from page links. As each page link has a mention string (the so-called "anchor text") and a target Wikipedia

---

[4] `https://github.com/facebookresearch/GENRE/blob/main/scripts_genre/train.sh`

page, we can simply go through both datasets and collect all mentions and their targets. To ensure that the entity titles are up-to-date, we check with calls to the Wikipedia API if the titles lead to an existing Wikipedia page and discard them if not. This yields a set of [mention, entity] tuples. We aggregate and count these tuples.

Using the Wikidata API, we retrieve for each entity in our vocabulary the corresponding Wikidata page. From this page, we extract aliases from the "also known as" field. We interpret all aliases as additional mentions to an entity, leading to another set of [mention, entity] tuples that we aggregate with the first list. To cover a broader range of mentions we add the lower cased version and version without blanks and special characters of each mention to the tuple set.

GLADIS: A General and Large Acronym Disambiguation Benchmark

Lihu Chen[1], **Gaël Varoquaux**[2], **Fabian M. Suchanek**[1]
[1] LTCI & Télécom Paris & Institut Polytechnique de Paris, France
[2] Soda, Inria Saclay, Université Paris-Saclay, France
`{lihu.chen, fabian.suchanek}@telecom-paris.fr`
`{gael.varoquaux}@inria.fr`

Abstract

Acronym Disambiguation (AD) is crucial for natural language understanding on various sources, including biomedical reports, scientific papers, and search engine queries. However, existing acronym disambiguation benchmarks and tools are limited to specific domains, and the size of prior benchmarks is rather small. To accelerate the research on acronym disambiguation, we construct a new benchmark named GLADIS with three components: (1) a much larger acronym dictionary with 1.5M acronyms and 6.4M long forms; (2) a pre-training corpus with 160 million sentences; (3) three datasets that cover the general, scientific, and biomedical domains. We then pre-train a language model, *AcroBERT*, on our constructed corpus for general acronym disambiguation, and show the challenges and values of our new benchmark.

1 Introduction

An acronym is an abbreviation formed from the initial letters of a longer name. For instance, the following two sentences contain the acronym *“AI”*: (1) *This is the product’s first true **AI** version, and it understands your voice instantly.* (2) *In the United States, the **AI** for potassium for adults is 4.7 grams.* The long forms (or expanded forms) for the same acronym are *“Artificial Intelligence”* and *“Adequate Intake”*, respectively.

Acronym Disambiguation (AD) is the task of mapping a given acronym in a given sentence to the intended long form. Acronym disambiguation is crucial for downstream tasks such as information extraction, machine translation, and query analysis in search engines (Jain et al., 2007; Islamaj Dogan et al., 2009). Acronym disambiguation is also important for humans: acronyms may make a text more difficult to understand for readers who are not familiar with the specific domain. A study on a Microsoft question answering forum found that only 7% of the acronyms co-occur with their corresponding long forms, which confuses the readers about the meaning of a text (Li et al., 2018).

ID	Long Form	Popularity	Domain
1	***Artificial Intelligence***	★★★★★	Computer Science
2	*Adequate Intake*	★★★★	Food and Nutrition
3	*Aromatase Inhibitor*	★★★	Chemistry
4	*Apoptotic Index*	★★★	Biomedicine
5	*Asynchronous Irregular*	★★★	Neuroscience
6	*Amnesty International*	★★	Organization
7	*Anterior Insula*	★★	Biomedicine
8	*Air India*	★★	Organization
9	***Article Influence***	★★	Science
......			
2243	*Agricultural Implement*	★	Agriculture

Table 1: Long form candidates for the acronym *“AI”* from our acronym dictionary. The SciAD benchmark (Veyseh et al., 2020) only includes two long terms (black) in the scientific domain. The popularity is the occurrence frequency in our collected corpora.

Acronym Disambiguation has received more attention in the past few years. The first step in acronym disambiguation is usually the creation of a dictionary, i.e., a mapping of each acronym to one or more long forms. Early systems extracted acronyms and their definitions automatically from texts by rule-based (Schwartz and Hearst, 2002) or supervised (Nadeau and Turney, 2005) methods. Once a dictionary is available, acronym disambiguation methods expand acronyms in a given text by capturing the contexts for specific domains, e.g., the enterprise domain (Li et al., 2018), biomedical texts (Jin et al., 2019), and scientific papers (Charbonnier and Wartena, 2018). Madog (Veyseh et al., 2021) was the first general and web-based system, recognizing and expanding acronyms across multiple domains. Several benchmarks have also been constructed, including for the biomedical area (Suominen et al., 2013) and the scientific area (SciAD, Veyseh et al., 2020). Several methods fine-tuned SciBERT (Beltagy et al., 2019) on SciAD to disambiguate acronyms in scientific documents (Pan et al., 2021; Zhong et al., 2021; Li et al., 2021).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2073–2088
May 2-6, 2023 ©2023 Association for Computational Linguistics

Although these works have significantly advanced the progress of acronym disambiguation, they suffer from three main limitations. First, most existing dictionaries (and benchmarks) focus on one specific domain. In real-world applications, however, the input text may be general, cross-domain, or of an unspecified domain (as in search engine queries). Second, existing dictionaries are limited in size. For example, there are only two long forms for the acronym *"AI"* in SciAD (Table 1), which is constructed from arXiv. However, we find that the two long forms *"Asynchronous Irregular"* and *"Anterior Insula"* also appear in scientific papers on arXiv (Girardi-Schappo et al., 2019; Vadovičová, 2014), and the acronym *"AI"* also appears separately without the long form in sentences. In our work, we actually find at least 2 243 different long forms for *"AI"*. Besides, SciAD suffers from the problem of data leakage, because the train and test sets have overlapping pairs of acronym and long form. Finally, current general AD systems such as MadDog (Veyseh et al., 2021) rely on static word embeddings and LSTMs (Long Short Term Memory (Hochreiter and Schmidhuber, 1997)). Thus, they do not leverage pre-training on large corpora, which drives the current state of the art in most NLP tasks with contextual embeddings like BERT (Devlin et al., 2018).

With this work, we aim to improve Acronym Disambiguation along two dimensions: First, we automatically construct GLADIS, a **G**eneral and **L**arge **A**cronym **DIS**ambiguation benchmark that includes a larger dictionary, a pre-training corpus and three datasets covering the general, biomedical, and scientific domains. Our dictionary contains 1.5M acronyms and 6.4M long forms, which trumps existing dictionaries by a factor of 3. We complement this dictionary by three domain-specific datasets for acronym disambiguation, which are adapted from three existing human-annotated and crowd-sourced datasets (Mohan and Li, 2018; Onoe and Durrett, 2020; Veyseh et al., 2020). The pre-training corpus has 160 million sentences with acronyms, collected from the Pile dataset (Gao et al., 2020) with a rule-based algorithm (Schwartz and Hearst, 2002). Second, we propose AcroBERT, the first pre-trained language model for general acronym disambiguation. Our experiments show that this model outperforms existing systems across multiple domains. Our code and data are available at `https://github.com/tigerchen52/GLADIS`.

2 Related Work

2.1 Acronym Identification and Disambiguation

To expand acronyms, there are usually two subtasks: Acronym Identification (AI), which creates a dictionary of acronyms and their definitions from a given document, and Acronym Disambiguation (AD), which aims to link acronyms in the input text to the correct long forms from a dictionary.

The study of acronym identification has a long history. Early work observed that acronyms and their long forms appear frequently together in a document, as in *"Artificial Intelligence (AI)"*. Based on this pattern, many approaches identify and extract acronyms by using rules (Yeates et al., 2000; Larkey et al., 2000; Pustejovsky et al., 2001; Park and Byrd, 2001; Yu et al., 2002; Schwartz and Hearst, 2002; Adar, 2004; Ao and Takagi, 2005; Okazaki and Ananiadou, 2006; Sohn et al., 2008; Veyseh et al., 2021) or supervised methods (Chang et al., 2002; Nadeau and Turney, 2005; Kuo et al., 2009; Movshovitz-Attias and Cohen, 2012; Liu et al., 2017; Wu et al., 2017; Zhu et al., 2021). In our work, we build on previous work (Schwartz and Hearst, 2002) for Acronym Identification, and focus mainly on disambiguation.

As for acronym disambiguation, early solutions manually designed features to score each pair of acronyms and long forms, by either unsupervised (Jain et al., 2007; Henriksson et al., 2014) or supervised machine learning (Pakhomov et al., 2005; Yu et al., 2007; Stevenson et al., 2009; Finley et al., 2016; Li et al., 2018). Later, deep learning approaches were introduced to the task, using embeddings to represent word sequences. The methods can be categorized as static embedding-based (Wu et al., 2015; Li et al., 2015; Charbonnier and Wartena, 2018) and dynamic embedding-based (Jin et al., 2019; Pan et al., 2021; Zhong et al., 2021; Li et al., 2021), where the former generates fixed representations for words in a pre-defined vocabulary and the latter can represent arbitrary words dynamically based on specific contexts. One main limitation of these methods is that they are domain-specific systems that can be applied only to a certain field such as the biomedical domain or scientific documents. To generalize the system, Ciosici and Assent (2018) presented the Abbreviation Expander, and Veyseh et al. (2021) proposed MadDog, both of which can be used in multiple domains. In this paper, we improve over the performance of these systems by adapting transformer-based methods and pre-training strategies.

2.2 Existing benchmarks

Most current public datasets for acronym expansion are focused on a particular domain, such as the biomedical domain (Suominen et al., 2013; Wen et al., 2020) or science (Charbonnier and Wartena, 2018; Veyseh et al., 2020). Some works adopt two domain-specific datasets for better evaluations (Ciosici et al., 2019; Veyseh et al., 2022). The main limitation of these benchmarks is two-fold: first, their acronym dictionaries are rather small. For instance, the average number of candidates per acronym in the SciAD benchmark (Veyseh et al., 2020) is 3.15 while in our benchmark the number is greater than 200. Second, there are no AD evaluation sets that cover multiple domains. We also note that, in SciAD, the train and test sets have overlapping pairs of acronym and long form. For example, the pair ⟨*CT, Computed Tomography*⟩ appears in the training, validation, and test sets.

3 Constructing GLADIS

Our GLADIS benchmark consists of three components: a dictionary, a pre-training corpus, and three domain-specific datasets.

3.1 Dictionary and Pre-training Corpus

We propose an acronym dictionary that addresses the shortcomings of existing dictionaries (Section 2.2) by being (1) cross-domain and (2) large in size. To construct this dictionary, we apply rule-based extraction on a large set of corpora that contain acronym definitions. In this process, we can also obtain a large number of sentences containing acronyms as the pre-training corpus.

Input Corpora. For the textual data source, we use the Pile dataset (Gao et al., 2020), an 825 GiB English corpus constructed from 22 diverse high-quality subsets (see the details of Pile in Appendix A.1). We also make use of structured knowledge from knowledge bases, namely the Alias Table from Wikidata and the Concept Names from UMLS. Both of them contain alternate names for canonical entities, and these may be acronyms or not. To consider only the acronyms, we produce pairs of the canonical name and an alternate name in the form *"canonical form (alternate name)"*. The rule-based algorithm will then decide whether to extract an acronym or not. Table 2 shows the statistics of our sources. They cover a wide range of domains including Web pages, books, scientific and biomedical papers, legal documents, etc.

Subset	Domain	Size (GiB)
Pile-CC	Web Archive files	227.12
Books3	Books	100.96
Github	Open-source codes	95.16
PubMed Central	Biomedical articles	90.27
OpenWebText2	Reddit submissions	62.77
ArXiv	Research papers	56.21
FreeLaw	Legal proceedings	51.15
Stack Exchange	Question-answer texts	32.20
USPTO Backgrounds	Patents	22.90
PubMed Abstracts	Biomedical abstracts	19.26
OpenSubtitles	Subtitles	12.98
Gutenberg (PG-19)	Western literatures	10.88
DM Mathematics	mathematical problems	7.75
Wikipedia (en)	Wikipedia pages	6.38
BookCorpus2	Books	6.30
Ubuntu IRC	Chatlog data	5.52
EuroParl	Proceedings	4.59
HackerNews	Comments of social news	3.90
YoutubeSubtitles	YouTube subtitles	3.73
PhilPapers	Philosophy publications	2.38
NIH ExPorter	Awarded applications	1.89
Enron Emails	Emails	0.88
Wikidata Alias	Alias Table	11.00
UMLS Concept	Biomedical Vocabulary	1.96
Total	-	838.14

Table 2: Sources for acronym extraction. All corpora except Wikidata Alias and UMLS Concept are from Pile (Gao et al., 2020).

Acronym Extraction. To extract acronyms from the textual sources, we use the rule-based algorithm proposed by Schwartz and Hearst (2002). It assumes that acronyms follow a predictable pattern, e.g., *long form (acronym)* or *acronym (long form)*, and then uses rules to extract candidate pairs by identifying parentheses and surrounding tokens. Experimental results show that this simple algorithm achieves 95% precision and 82% recall, averaged over two datasets. As the method has good results at low time complexity, we decided to not adopt more sophisticated methods. Some extracted samples are shown in Table A2 in the appendix. A manual evaluation on a random sample of 100 extracted sentences yields a precision of 94%.

Dictionary Construction. Next, we build a large-scale acronym dictionary with frequencies (popularity) by merging the extracted outputs. This merger may regroup duplicate long forms for an acronym, e.g., *"convolutional neural network"*, *"convolutional-neural network"* or *"convolutional neural networks"*. Therefore, we merge long forms that are identical after stemming and removing punctuation. In our case, the above three forms are merged into *"convolutional neural network"*. We keep the most frequent, unpreprocessed, long

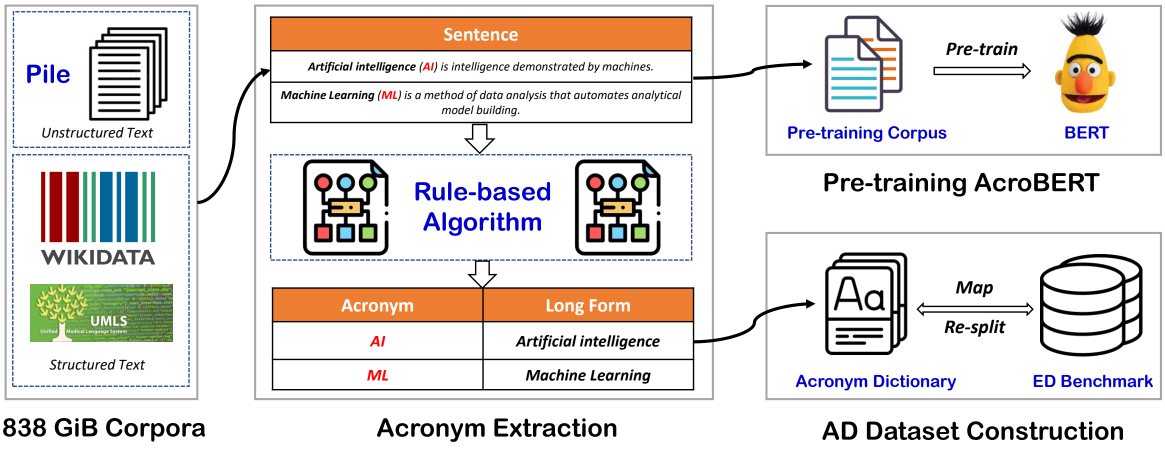


Figure 1: Framework of our benchmark construction. The *"ED"* in the lower right corner means *"Entity Disambiguation"*.

	train	valid	test	unique short form	long forms per acronym	overshadowed ratio
General	13,269	7,024	7,125	1,147	248	29.8%
Scientific	28,023	14,134	14,066	2,922	262	68.7%
Biomedical	6,295	3,150	3,149	2,909	278	27.4%

Table 3: Statistics of our new Acronym Disambiguation Benchmark. The last column shows the ratio of overshadowed samples in the dataset: long forms with the same acronym but not the most popular one.

	Short Form	Long Form	Avg
SciAD (2020)	732	2,308	3.15
MadDog (2021)	426,389	3,781,739	8.87
Ours	1,542,819	6,381,257	4.14

Table 4: Statistics for three acronym dictionaries. The "Avg" column shows the average number of long forms per acronym.

form as the canonical name in our dictionary, discarding other forms. There are still some noisy long forms that cannot be merged, caused by typos and nested acronyms (see Section 7). However, a manual evaluation on a sample shows that 94% of the long forms are clean. If the long forms are weighted by their frequency, the percentage of clean forms increases to 97%. Most notably, all most frequent long forms for a given acronym were clean in our sample. The statistics of our dataset are shown in Table 4. Our resource will be the largest publicly available dictionary for acronyms that covers various domains.

Pre-training Corpus. While building the dictionary, we can also collect the sentences that contain acronyms for pre-training. For example, the following sentence contains the acronym *ELEC*: *"Christie, some legislators and the state Election Law Enforcement Commission (**ELEC**), have joined the comptroller in voicing support for the elimination of the loophole."* For pre-training, the long form *Election Law Enforcement Commission* is removed, and we then force the model to restore the long form from our constructed dictionary, based on the input sentence and the acronym. In total, we collect a pre-training corpus with ~160 million sentences. More examples are shown in Table A2.

3.2 Acronym Disambiguation Dataset

We use our acronym dictionary to construct new, larger datasets for evaluating AD systems. To automatically construct the datasets, we adapt the existing two Entity Disambiguation datasets by replacing the long form of entity with the acronym. For example, one sentence in Medmentions (Mohan and Li, 2018) contains the long form of Cerebral Blood Flow: *"The reconstructed volume was then compared with corresponding magnetic resonance images demonstrating that the volume of reduced **Cerebral Blood Flow** agrees with the infarct zone at twenty-four hours"*. The dataset provides the unique ID of this long form in UMLS (*C1623258*), and we use it to find the acronym *CBF* in UMLS.

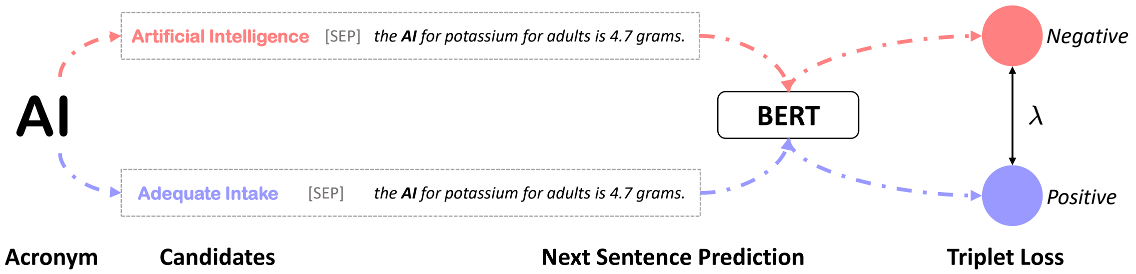


Figure 2: The pre-training strategy of AcroBERT. λ is a margin between positive and negative pairs, here $\langle$*Adequate Intake, AI*$\rangle$ and $\langle$*Artificial Intelligence, AI*$\rangle$.

Therefore, a new sample can be obtained by replacing the long form with its corresponding acronym.

Specifically, we use the following human-annotated and crowd-sourced datasets:

WikilinksNED Unseen Mentions (Onoe and Durrett, 2020) is an Entity Disambiguation dataset, i.e., a set of text documents that have mentions of entities, together with a reference knowledge base (KB) that contains, for each entity, one or several names. WikilinksNED Unseen Mentions re-splits the WikilinksNED dataset (Eshel et al., 2017) to ensure that all mentions in the validation and test sets do not appear in the training set. This is a large-scale, crowd-sourced ED dataset from websites in various fields, which is significantly noisier and more challenging than prior datasets. The reference KB is Wikidata (or Wikipedia), and we adapt this WikilinksNED Unseen Mentions to an AD dataset in the **general** domain.

Medmentions (Mohan and Li, 2018) is an entity disambiguation dataset of 4,392 PubMed papers that were annotated by professional and experienced annotators in the biomedical domain. The reference KB is UMLS (Bodenreider, 2004), and this is a **biomedical** dataset.

SciAD (Veyseh et al., 2020) is the previously mentioned acronym disambiguation dataset in the **scientific** domain.

SciAD is already an AD dataset, and we only re-split it to avoid data leakage. As for the two ED datasets, they both provide a unique ID to the reference KB for each long form. We then replace the long forms with the acronyms from their corresponding reference KB, i.e., Wikidata and UMLS. To make sure this replacement is correct, we apply the rule-based algorithm (Schwartz and Hearst, 2002) to the pair of long form and acronym again for verification. We manually checked 100 random sentences constructed in this way and did not find problematic cases. Hence, this semi-synthetic construction results in a dataset of natural text in which the long form and the acronym are mutually replaceable in the context. Besides, the pair is added to our dictionary with a frequency of 1 if it does not appear in our dictionary. For the WikilinksNED dataset, we use the taxonomy of YAGO 4 (Pellissier Tanon et al., 2020) to label each long form with a top-level class. For example, *"rhythm and blues"* is a `CreativeWork` and *"United States Navy"* is an `Organization`.

We then partition the three datasets separately into training, test, and validation set, ensuring that the acronyms in the training set do not appear in the validation and test sets. We repartition the datasets at the ratio of 6:2:2. Table 3 gives the statistics of this new benchmark. It is not only larger but also more challenging than existing benchmarks, because acronyms in our benchmark have more than 200 candidates on average. Moreover, it contains many *overshadowed forms* (Provatorova et al., 2021), which means that an acronym has to be disambiguated to a long form that is not the most popular long form for that acronym. For example, *"Adequate Intake"* is overshadowed by the more popular form *"Artificial Intelligence"* for the acronym *"AI"*.

4 AcroBERT

We can now capitalize on our dictionary and pre-training corpus to propose a new method for acronym disambiguation. It takes as input (1) a dictionary of acronyms with their long form(s), and (2) a large-scale corpus that contains acronyms (we assume that the boundaries of the acronym have already been recognized). Our goal is to pre-train a language model for acronym disambiguation, which has a strong generalizability across multiple domains.

Model	**General**				**Scientific**				**Biomedical**				**Avg**	
	Dev		Test		Dev		Test		Dev		Test			
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
BM25 (1995)	29.9	32.6	35.5	25.8	14.1	5.4	17.1	10.7	13.1	8.3	17.0	14.3	21.1	16.2
FastText (2017)	11.3	12.9	18.7	12.7	3.3	0.9	5.7	2.5	0.2	0.1	1.3	0.7	6.8	5.0
MadDog (2021)	28.1	11.7	29.9	23.1	17.8	15.5	22.4	17.9	33.8	19.3	41.2	35.9	28.9	20.6
BERT (2018)	32.3	32.5	37.7	28.2	15.1	5.8	17.6	9.3	3.1	1.3	3.5	2.1	18.2	13.2
Popularity-Ours	35.2	39.1	39.0	43.2	5.5	22.9	4.9	12.3	46.0	61.3	49.9	54.0	30.1	38.8
AcroBERT	**74.7**	**78.8**	**70.0**	**72.0**	**26.9**	**36.6**	**28.8**	**27.4**	**58.4**	**66.0**	**59.9**	**61.4**	**53.1**	**57.0**

Table 5: Performances of the unsupervised setting across different models, measured by macro F1 and Accuracy.

The Pre-training Strategy of BERT. We adapt the BERT model for our purpose. BERT is pre-trained by using two unsupervised tasks, Masked Language Model (MLM) and Next Sentence Prediction (NSP). The Masked Language Model task randomly masks some percentage of the input tokens, and then forces the model to predict the masked tokens, similar to a cloze task. The Next Sentence Prediction task asks the model to predict whether one sentence follows the other.

The Next Sentence Prediction task can be used to predict, from the input text (e.g., *"This is the product's first true **AI** version, and it understands your voice instantly."*), the correct long form (*"Artificial Intelligence"*). Here, the model learns to judge whether the input context that contains the acronym *"AI"* is coherent with the long form *"Artificial Intelligence"*. The Masked Language Model task can memorize the correlation of tokens between the context sequence and long form. Thus, the model learns that the phrase *"Artificial Intelligence"* often co-occurs with *"product"* or *"understand"*.

However, we find that this naive technique does not perform well (see the ablation studies in Table A4). We believe that the reason is that the acronym is usually ambiguous with many candidates (as shown in Table 1), so that the model has difficulties predicting the correct long form by only using the cross-entropy loss of the binary classification. We also observe that the Masked Language Model loss is so small that the model focuses on adapting the Next Sentence Prediction task only.

AcroBERT. To mitigate the weaknesses of the original BERT, we pre-train an adapted BERT, called AcroBERT, by slightly adjusting the Next Sentence Prediction task. The framework is shown in Figure 2. It aims to bring the positive sample pairs closer together, and to push apart the negative sample pairs. We find that already such a simple model can perform very well. For each pair of a candidate long form and a sentence with an acronym, we compose an input for the Next Sentence Prediction task as *"*`[CLS]` *long form* `[SEP]` *sentence* `[SEP]`*"*. Then, we obtain representations of this sequence by applying BERT to this new input. The final hidden vector $\mathbf{e}^{\texttt{[CLS]}} \in \mathbb{R}^H$ of the first input token (`[CLS]`) is used as the aggregate representation, where H is the dimension of the hidden vector. Next, the scores for the binary classification are:

$$\mathrm{P}(y) = \mathrm{softmax}(\mathbf{e}^{\texttt{[CLS]}}\mathbf{W}), y \in \{0, 1\} \qquad (1)$$

where $\mathbf{W} \in \mathbb{R}^{H \times 2}$ is a trainable matrix initialized with the weights of the original BERT, and the label 0 signifies that this pair of sentences are coherent. We use $d = \mathrm{P}(y = 1)$ as the distance between the candidate and the context, and we want the distances of negative pairs to be larger than for positive pairs. For this, we use a triplet loss function that aims to assign higher scores to the correct candidates that match the topic of the input sentence while reducing the scores of irrelevant candidates:

$$\mathcal{L} = \max\left\{0, \lambda - d_{\text{neg}} + d_{\text{pos}}\right\} \qquad (2)$$

where λ is the margin value, and d_{pos} and d_{neg} are the distances for positive and negative pairs, respectively.

The negatives in this triplet framework can be randomly sampled from the dictionary. However, we observe that such random negatives contribute less to the training and result in slower convergence because the initial model can easily distinguish these triplets. Therefore, it is crucial to select harder triplets that are active and beneficial to the training. For this purpose, we introduce a certain number of ambiguous negatives to each mini-batch, e.g., *"Artificial Intelligence"* can be added to the input context as an ambiguous negative sample for the positive pair *"Adequate Intake* `[SEP]` *In the United States, the **AI** for potassium for adults is 4.7 grams."* Through the pre-training step, AcroBERT is able to identify the correct long form with the most consistent theme from numerous candidates based on the input context.

5 Experiments

In this section, we compare AcroBERT empirically to other acronym disambiguation approaches.

5.1 Experimental Settings

Datasets. We use the following datasets for evaluation: Our GLADIS benchmark consists of three subsets covering the General, Scientific, and Biomedical domains. This benchmark is more challenging than prior work due to a large number of ambiguous long forms: each acronym has around 200 candidates on average. We also evaluate AcroBERT on two existing datasets: UAD (Ciosici et al., 2019) and SciAd (Veyseh et al., 2020). They are general and scientific AD datasets, respectively. We reuse the test set of Medmentions but use the UMLS as the target dictionary. We refer to them as **datasets with fewer candidates** because they have fewer candidates per acronym. The average numbers of candidates per acronym are 2.1, 3.1, and 34.2, respectively. See Appendix A.2 for more details on the datasets.

Benchmark Settings. We design two benchmark settings for the unsupervised and supervised scenarios respectively. In the *unsupervised setting*, each model is evaluated on the test sets without access to train and validation sets. In the *fine-tuned setting*, each model is first fine-tuned on train sets and then evaluated on test sets. We focus on the unsupervised setting because it demonstrates that AcroBERT can achieve considerable performances across several domains even without any annotated samples.

Competitors. We compare our approach to the following publicly available competitors: BM25 (Robertson et al., 1995), FastText (Bojanowski et al., 2017), MadDog (Veyseh et al., 2021), BERT (Devlin et al., 2018), BioBERT (Lee et al., 2020), SciBERT (Beltagy et al., 2019). Besides, we introduce a Popularity-Ours baseline that uses the frequency of long forms of our collected pre-training datasets. We do not compare to general entity linking methods, because prior work has already found that general systems like AIDA (Hoffart et al., 2011) tend to lag behind acronym disambiguation models by 10-30 absolute percentage points (Li et al., 2018). See Appendix A.3 for details on the competitors.

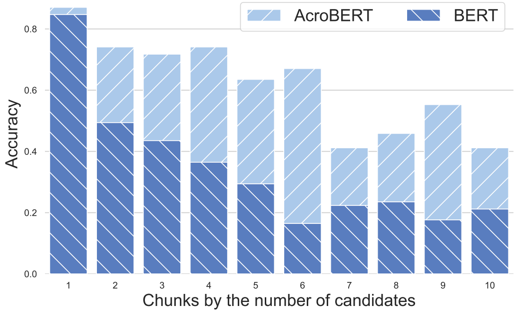


Figure 3: Robustness evaluation of hard samples on the General test set. The samples are divided evenly into ten chunks according to the number of candidates of each sample (results on the other two sets are shown in Figure A1 in the appendix).

Implementation Details. All approaches are implemented with PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2020) by using one NVIDIA Tesla V100S PCIe 32 GB Specs. For pre-training, we use the parameters in the original BERT to initialize AcroBERT, and then pre-trained on our collected datasets with ~160 million samples for one epoch. The margin of triplet loss is 0.2 and the number of ambiguous negatives is 2 for each mini-batch. See more details in Appendix A.4.

Inference. For the inference stage, every pair of a context sentence and a candidate with the matching short form in the dictionary constitutes an input to the Next Sentence Prediction task. The language model produces a score for each candidate and we select the one with the highest score as the final predicted output.

Metrics. We evaluate the models by precision, recall, and macro F1. These metrics are defined in detail in Section A.5

5.2 Results

5.2.1 Overall Performance

Unsupervised Setting. Table 5 shows the experimental results in the unsupervised setting. We first observe that our AcroBERT significantly outperforms the baselines across the three domains. For example, AcroBERT can improve the original BERT by more than 30 absolute percentage points of F1 on average on our benchmark. Second, the naive popularity method comes second on this

	General				Scientific				Biomedical				Avg	
	Dev		Test		Dev		Test		Dev		Test			
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
BERT (2018)	53.8	70.7	54.9	53.1	13.5	9.9	14.3	10.4	9.8	12.4	9.4	7.5	26.0	27.3
SciBERT (2019)	32.4	38.6	33.6	27.7	22.7	19.4	23.4	17.7	31.2	35.6	31.0	28.3	29.5	27.9
BioBERT (2020)	26.0	23.6	25.7	20.3	11.2	9.7	12.4	9.0	24.0	21.8	20.2	16.8	19.9	16.9
AcroBERT	**72.9**	**76.1**	**71.0**	**76.1**	**28.7**	**34.9**	**29.0**	**27.6**	**62.5**	**62.4**	**60.3**	**69.2**	**54.1**	**57.7**

Table 6: Performances of fine-tuned setting across different models, measured by macro F1 and Accuracy.

benchmark, most likely because it contains a limited number of overshadowed terms. However, it performs badly on the scientific dataset. We assume that this is because this dataset contains 68.7% of overshadowed terms (as shown in Table 3).

Besides, we conduct experiments on existing datasets, namely UAD (Ciosici et al., 2019) and SciAD (Veyseh et al., 2020). Although our method performs consistently well, we relegate this experiment to the appendix B.2 due to the weaknesses of the datasets (small size or data leakage).

Fine-tuned Setting. In this experiment, every pre-trained language model is fine-tuned on the training set by the triplet loss, as introduced in the pre-training step. Negatives are randomly sampled from ambiguous long forms for the correct label, and the results are shown in Table 6. BERT, SciBERT, and BioBERT perform better in their respective fields. However, our AcroBERT achieves the best result across the three fields on average, which demonstrates the effectiveness of the pre-training strategy. One might think that it is unfair that AcroBERT uses the pre-training corpus, while the other models do not. However, there is no other pre-trained model for general disambiguation. Our approach is the first that capitalizes on large-scale corpora and pre-training.

As for the inference speed, AcroBERT has to be run once for every possible long form, which may take some time if there are thousands of long forms, e.g., the acronym AI. However, this runtime can be reduced drastically if one cuts off the less frequent long forms per acronym. Limiting the number of long forms to 23 per acronym, e.g., reduces the worst-case runtime by a factor of 100, while still keeping the recall at 90%.

5.2.2 Robustness Evaluation

Our GLADIS benchmark is more challenging than existing acronym disambiguation datasets due to the much larger acronym dictionary, which means more candidates per acronym. To measure the robustness of acronym disambiguation systems against more candidates, we sort the samples in the dataset in descending order of the number of candidates per acronym, and divide them evenly into 10 chunks. For example, samples in the first chunk have 1.58 candidates on average while that number is 2159 for the last chunk. The experimental results are shown in Figure 3. As expected, the performance of BERT and AcroBERT decreases as the number of candidates increases. The same goes for the other two subsets, as shown in Appendix B.3. However, AcroBERT consistently outperforms BERT on each data chunk, which shows that AcroBERT is able to select the correct long form among the numerous candidates.

Model	Popular	Overshadowed	Avg
BERT (2018)	13.3	12.7	13.0
SciBERT (2019)	11.6	8.1	9.9
BioBERT (2020)	2.1	1.0	1.6
AcroBERT	**61.9**	**33.4**	**47.7**

Table 7: Robustness evaluation of overshadowed entities on General test set,measured by Accuracy.

Moreover, the challenge with our GLADIS benchmark comes from overshadowed samples, which are harder to disambiguate. To validate the robustness of the models, we divide the General test set into two parts: Popular and Overshadowed, as described in Section 3.2. Next, we compare different language models in the unsupervised setting. As shown in Table 7, our AcroBERT performs best on both the Popular and the Overshadowed subset. We conclude that AcroBERT is more robust against ambiguous and overshadowed samples in acronym disambiguation task.

6 Conclusion

In this paper, we have presented GLADIS, a challenging benchmark for Acronym Disambiguation, which includes a larger dictionary, three datasets from the general, scientific, and biomedical domains, and a large-scale pre-training corpus. We have also proposed AcroBERT, a BERT-based model that is pre-trained on our collected acronym documents, which can significantly outperform other baselines across multiple domains, and which is more robust in the presence of very ambiguous acronyms and overshadowed samples. For future work, we aim to enhance the performance of AcroBERT on the overshadowed cases, which is crucial for the acronym disambiguation task.

7 Limitations

We see two main limitations of our work. First, although the current acronym dictionary is of relatively high quality, it still contains a small fraction of duplicate long forms due to typos (as in *"Convlutional Neural Network"*), morphological changes (as in *"Convolutional Neuronal Network"*) and nested acronyms (as in *"convolutional NN"*). A manual evaluation of 100 randomly chosen long forms from the three datasets in GLADIS shows that 6% of them are noisy. At the same time, the frequency of these noisy forms is much lower than that of the standard long forms: all noisy forms in the sample taken together appear 100 times in the corpus – compared to 31k times for the clean forms. Thus, the percentage of clean forms, weighted by their frequency, is 97%. A good AD system should select the most frequent one among noisy forms for an acronym, and in our sample none of the most frequent forms was noisy.

A second limitation of our approach is that the performance of the current AcroBERT system on the Scientific dataset still needs improvement. We are considering to introduce more pre-training data from this domain to address this issue.

Ethics Statement

This work presents GLADIS, a free and open benchmark for the research community to study Acronym Disambiguation, which consists of three components: a dictionary, a pre-training corpus, and three domain-specific datasets. The dictionary and pre-training corpus are collected from the Pile dataset (Gao et al., 2020), which is a public dataset under the MIT license. The three domain-specific datasets are adapted from SciAD (Veyseh et al., 2020), WikilinksNED Unseen Mentions (Onoe and Durrett, 2020) and Medmentions (Mohan and Li, 2018), respectively. They all allow sharing and redistribution. The source datasets and their publications will be credited on our GitHub page, and their licenses will be mentioned both on the Web page and in the downloads of GLADIS.

Acknowledgements

This work was partially funded by ANR-20-CHIA-0012-01 ("NoRDF").

References

Eytan Adar. 2004. Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Hiroko Ao and Toshihisa Takagi. 2005. Alice: an algorithm to extract abbreviations from medline. *Journal of the American Medical Informatics Association*, 12(5):576–586.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Jeffrey T Chang, Hinrich Schütze, and Russ B Altman. 2002. Creating an online dictionary of abbreviations from medline. *Journal of the American Medical Informatics Association*, 9(6):612–620.

Jean Charbonnier and Christian Wartena. 2018. Using word embeddings for unsupervised acronym disambiguation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2610–2619.

Manuel Ciosici, Tobias Sommer, and Ira Assent. 2019. Unsupervised abbreviation disambiguation contextual disambiguation using word embeddings. *arXiv preprint arXiv:1904.00929*.

Manuel R Ciosici and Ira Assent. 2018. Abbreviation expander-a web-based system for easy reading of technical documents. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 1–4.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

István Endrédy and Attila Novák. 2013. More effective boilerplate removal-the goldminer algorithm. *Polibits*, (48):79–83.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 58–68.

Gregory P Finley, Serguei VS Pakhomov, Reed McEwan, and Genevieve B Melton. 2016. Towards comprehensive clinical abbreviation disambiguation using machine-labeled training data. In *AMIA Annual Symposium Proceedings*, volume 2016, page 560. American Medical Informatics Association.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Mauricio Girardi-Schappo, Ludmila Brochini, Ariadne A Costa, Tawan TA Carvalho, and Osame Kinouchi. 2019. Self-organized critical balanced networks: a unified framework. *arXiv preprint arXiv:1906.05624*.

Aron Henriksson, Hans Moen, Maria Skeppstedt, Vidas Daudaravičius, and Martin Duneld. 2014. Synonym extraction and abbreviation expansion with ensembles of semantic spaces. *Journal of biomedical semantics*, 5(1):1–25.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 782–792.

Rezarta Islamaj Dogan, G Craig Murray, Aurélie Névéol, and Zhiyong Lu. 2009. Understanding pubmed® user search behavior through log analysis. *Database*, 2009.

Alpa Jain, Silviu Cucerzan, and Saliha Azzam. 2007. Acronym-expansion recognition and ranking on the web. In *2007 IEEE International Conference on Information Reuse and Integration*, pages 209–214. IEEE.

Qiao Jin, Jinling Liu, and Xinghua Lu. 2019. Deep contextualized biomedical abbreviation expansion. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 88–96.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *European conference on machine learning*, pages 217–226. Springer.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of machine translation summit x: papers*, pages 79–86.

Cheng-Ju Kuo, Maurice HT Ling, Kuan-Ting Lin, and Chun-Nan Hsu. 2009. Bioadi: a machine learning approach to identifying abbreviations and definitions in biological literature. In *BMC bioinformatics*, volume 10, pages 1–10. BioMed Central.

Leah S Larkey, Paul Ogilvie, M Andrew Price, and Brenden Tamilio. 2000. Acrophile: an automated acronym extractor and server. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 205–214.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Bin Li, Fei Xia, Yixuan Weng, Xiusheng Huang, Bin Sun, and Shutao Li. 2021. Simclad: A simple framework for contrastive learning of acronym disambiguation. *arXiv preprint arXiv:2111.14306*.

Chao Li, Lei Ji, and Jun Yan. 2015. Acronym disambiguation using word embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Yang Li, Bo Zhao, Ariel Fuxman, and Fangbo Tao. 2018. Guess me if you can: Acronym disambiguation for enterprises. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1308–1317.

Jie Liu, Caihua Liu, and Yalou Huang. 2017. Multi-granularity sequence labeling model for acronym expansion identification. *Information Sciences*, 378:462–474.

Sunil Mohan and Donghui Li. 2018. Medmentions: A large biomedical corpus annotated with umls concepts. In *Automated Knowledge Base Construction (AKBC)*.

Dana Movshovitz-Attias and William Cohen. 2012. Alignment-hmm-based extraction of abbreviations from biomedical text. In *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pages 47–55.

David Nadeau and Peter D Turney. 2005. A supervised learning approach to acronym identification. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 319–329. Springer.

Naoaki Okazaki and Sophia Ananiadou. 2006. Building an abbreviation dictionary using a term recognition approach. *Bioinformatics*, 22(24):3089–3095.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8576–8583.

Juri Opitz and Sebastian Burst. 2019. Macro f1 and macro f1. *arXiv preprint arXiv:1911.03347*.

Serguei Pakhomov, Ted Pedersen, and Christopher G Chute. 2005. Abbreviation and acronym disambiguation in clinical discourse. In *AMIA Annual Symposium Proceedings*, volume 2005, page 589. American Medical Informatics Association.

Chunguang Pan, Bingyan Song, Shengguang Wang, and Zhipeng Luo. 2021. Bert-based acronym disambiguation with multiple training strategies. *arXiv preprint arXiv:2103.00488*.

Youngja Park and Roy J Byrd. 2001. Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 conference on empirical methods in natural language processing*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Thomas Pellissier Tanon, Gerhard Weikum, and Fabian Suchanek. 2020. Yago 4: A reason-able knowledge base. In *European Semantic Web Conference*, pages 583–596. Springer.

Vera Provatorova, Samarth Bhargav, Svitlana Vakulenko, and Evangelos Kanoulas. 2021. Robustness evaluation of entity disambiguation using prior probes: the case of entity overshadowing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10501–10510.

James Pustejovsky, José Castano, Brent Cochran, Maciej Kotecki, and Michael Morrell. 2001. Automatic extraction of acronym-meaning pairs from medline databases. In *MEDINFO 2001*, pages 371–375. IOS Press.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.

Stephen E Robertson, Steve Walker, Susan Jones, et al. 1995. Okapi at trec-3.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2018. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*.

Ariel S Schwartz and Marti A Hearst. 2002. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Biocomputing 2003*, pages 451–462. World Scientific.

Sunghwan Sohn, Donald C Comeau, Won Kim, and W John Wilbur. 2008. Abbreviation definition identification based on automatic precision estimates. *BMC bioinformatics*, 9(1):1–10.

Mark Stevenson, Yikun Guo, Abdulaziz Alamri, and Robert Gaizauskas. 2009. Disambiguation of biomedical abbreviations. In *Proceedings of the BioNLP 2009 Workshop*, pages 71–79.

Hanna Suominen, Sanna Salanterä, Sumithra Velupillai, Wendy W Chapman, Guergana Savova, Noemie Elhadad, Sameer Pradhan, Brett R South, Danielle L Mowery, Gareth JF Jones, et al. 2013. Overview of the share/clef ehealth evaluation lab 2013. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 212–231. Springer.

Jörg Tiedemann. 2016. Finding alternative translations in a large corpus of movie subtitle. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3518–3522.

Karin Vadovičová. 2014. Affective and cognitive prefrontal cortex projections to the lateral habenula in humans. *Frontiers in human neuroscience*, 8:819.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Walter Chang, and Thien Huu Nguyen. 2021. Maddog: A web-based system for acronym identification and disambiguation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 160–167.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, and Thien Huu Nguyen. 2020. What does this acronym mean? introducing a new dataset for acronym identification and disambiguation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3285–3301.

Amir Pouran Ben Veyseh, Nicole Meister, Seunghyun Yoon, Rajiv Jain, Franck Dernoncourt, and Thien Huu Nguyen. 2022. Macronym: A large-scale dataset for multilingual and multi-domain acronym extraction. *arXiv preprint arXiv:2202.09694*.

Zhi Wen, Xing Han Lu, and Siva Reddy. 2020. Medal: Medical abbreviation disambiguation dataset for natural language understanding pretraining. *arXiv preprint arXiv:2012.13978*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Yonghui Wu, Joshua C Denny, S Trent Rosenbloom, Randolph A Miller, Dario A Giuse, Lulu Wang, Carmelo Blanquicett, Ergin Soysal, Jun Xu, and Hua Xu. 2017. A long journey to short abbreviations: developing an open-source framework for clinical abbreviation recognition and disambiguation (card). *Journal of the American Medical Informatics Association*, 24(e1):e79–e86.

Yonghui Wu, Jun Xu, Yaoyun Zhang, and Hua Xu. 2015. Clinical abbreviation disambiguation using neural word embeddings. In *Proceedings of BioNLP 15*, pages 171–176.

Stuart Yeates, David Bainbridge, and Ian H Witten. 2000. Using compression to identify acronyms in text. In *Proceedings of the Conference on Data Compression*, page 582.

Hong Yu, George Hripcsak, and Carol Friedman. 2002. Mapping abbreviations to full forms in biomedical articles. *Journal of the American Medical Informatics Association*, 9(3):262–272.

Hong Yu, Won Kim, Vasileios Hatzivassiloglou, and W John Wilbur. 2007. Using medline as a knowledge source for disambiguating abbreviations and acronyms in full-text biomedical journal articles. *Journal of biomedical informatics*, 40(2):150–159.

Qiwei Zhong, Guanxiong Zeng, Danqing Zhu, Yang Zhang, Wangli Lin, Ben Chen, and Jiayu Tang. 2021. Leveraging domain agnostic and specific knowledge for acronym disambiguation. *arXiv preprint arXiv:2107.00316*.

Danqing Zhu, Wangli Lin, Yang Zhang, Qiwei Zhong, Guanxiong Zeng, Weilin Wu, and Jiayu Tang. 2021. At-bert: Adversarial training bert for acronym identification winning solution for sdu@ aaai-21. *arXiv preprint arXiv:2101.03700*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Details of the Experimental Settings

A.1 Details of the Pile Dataset

Pile (Gao et al., 2020) is an 825 GiB English text corpus designed to train large-scale language models, which is constructed from 22 diverse high-quality academic or professional sources. Pile is constructed from existing or newly introduced datasets, and we present these sources here. **Pile-CC** is a collection of web pages, metadata and texts, which is extracted from jusText (Endrédy and Novák, 2013). **Books3** is a book dataset of fiction and nonfiction books derived from Bibliotik [1]. **Project Gutenberg** has classic Western literature derived from PG-19 (Rae et al., 2019). **OpenSubtitles** provides a large corpus of English subtitles from movies and television shows collected by Tiedemann (2016). **DeepMind Mathematics** is a collection of many different types of mathematics questions (Saxton et al., 2018). **BookCorpus2** is an expanded version of BookCorpus (Zhu et al., 2015), a corpus of books from the web. **EuroParl** is a corpus of parallel text in 11 languages from the proceedings of the European Parliament(Koehn, 2005). **Enron Emails** is a large set of email messages, which contains 619,446 messages belonging to 158 users (Klimt and Yang, 2004).

A.2 Details of the Experimental Datasets

We use the following benchmarks for Acronym Disambiguation:

Our GLADIS benchmark consists of three subsets covering the General, Scientific, and Biomedical domains. It is a very challenging benchmark, due to a large number of ambiguous long forms, as described in Section 3.2.

- **General** has 45K samples gathered from the WikilinksNED Unseen Mentions (Onoe and Durrett, 2020).
- **Scientific** is adapted from SciAD (Veyseh et al., 2020) with 56K samples, and the long forms in the original dataset are mapped to the new acronym dictionary. We re-split the training, validation and test sets to assure there are no overlaps.
- **Biomedical** includes 12K samples obtained from Medmentions (Mohan and Li, 2018).

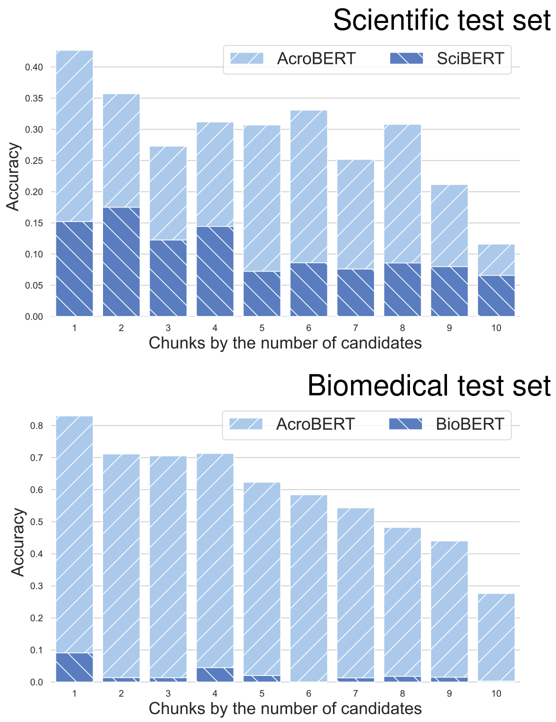


Figure A1: Robustness evaluation of hard samples on Scientific and Biomedical test set. The samples are divided evenly into ten chunks according to the number of candidates of each sample.

Datasets with fewer candidates per acronym.

- **UAD** (Ciosici et al., 2019) is gathered from the English Wikipedia and we use the manually labeled 7K samples for evaluation.
- **SciAD** (Veyseh et al., 2020) is a human-annotated dataset for the scientific domain with 62K samples gathered from the ArXiv preprint papers, and the validation set with 6K samples is used for experiments.
- **Biomedical-UMLS** is a dataset with 3K samples obtained from the test set in our benchmark by using the UMLS concepts as the acronym dictionary

The average candidates per acronym for the three datasets are 2.1, 3.1, and 34.2, respectively.

A.3 Competitors

We compare our approach to the following publicly available competitors:

- BM25 (Robertson et al., 1995) is a classical ranking function in information retrieval.

---

[1] `https://twitter.com/theshawwn/status/1320282149329784833`

Model	UAD		SciAD		Biomedical-UMLS		Avg	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc
BERT (2018)	**89.3**	91.1	54.1	57.2	38.0	32.7	60.5	60.3
SciBERT (2019)	74.8	78.4	**65.6**	71.7	54.9	50.3	65.1	66.8
BioBERT (2020)	66.2	68.2	19.7	22.4	37.4	31.4	41.1	40.7
AcroBERT	88.8	**93.7**	58.0	**72.0**	**67.5**	**65.3**	**71.4**	**77.0**

Table A1: Performances on benchmarks with fewer candidates, measured by macro F1 and Accuracy.

- Popularity-Ours is a baseline that uses the frequency of long forms of our collected pre-training datasets.

- BERT (Devlin et al., 2018) is a strong baseline, which pre-trains contextual language models on large corpora. The scores for the NSP task can be used for the acronym disambiguation.

- FastText (Bojanowski et al., 2017) is a character-level embeddings and can produce representations for arbitrary words. In this experiment, we first represent the input sentence and candidates by the sum of word embeddings from FastText. Then, all candidates are ranked by their cosine similarity score.

- MadDog (Veyseh et al., 2021) is a web-based acronym disambiguation system for multiple domains. It first creates chunks in which all samples with the same acronyms are assigned to the same chunks. After, a separate Bi-LSTM model is trained for each chunk. To deploy the MadDog server, it needs at least 125 GB of disk space and 70 GB of RAM memory [2].

- BioBERT (Lee et al., 2020) is a biomedical language representation model mainly pre-trained on PubMed Abstracts and PMC Full-text articles, which is a strong baseline in the biomedical domain.

- SciBERT (Beltagy et al., 2019) is a scientific language model based on BERT pre-trained on a large multi-domain corpus of scientific publications, which can improve performance on downstream scientific NLP tasks.

A.4 Implementation Details

All approaches are implemented with PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2020). When pre-training AcroBERT, the model is initialized by the parameters in the original BERT, and then pre-trained on our collected datasets for one epoch. In total, there are ~160 million samples in this corpus, covering various domains. The batch size is 32, and we use Adam (Kingma and Ba, 2014) with a learning rate $2e-5$ for optimization. The learning rate is exponentially decayed for every 10,000 steps with a rate of 0.95. The margin of triplet loss is 0.2 and the number of ambiguous negatives is 2 for each mini-batch.

For the fine-tuning stage, each competitor model is initialized with the pre-trained parameters from HuggingFace, and we use AcroBERT after pre-training for comparison. All models are fine-tuned by using the Triplet loss. All parameters of each model are fine-tuned in this experiment, across all domains by using the same hyper-parameters. The batch size is 8 and the learning rate is among $[1e-5, 8e-6, 6e-6, 4e-6, 2e-6]$ for the Adam optimizer. The model that has the best performance on the validation set among the 5 learning rates is evaluated on the test set. We use one NVIDIA Tesla V100S PCIe 32 GB Specs.

A.5 Metrics

Acronym disambiguation can be seen as a classification problem, where the input is (1) a dictionary of acronyms and (2) a sentence with an acronym. Each long form for that acronym from the dictionary is considered a class, and the acronym disambiguation has to choose the correct class. We evaluate the models by precision, recall, and macro F1. There are two ways to calculate the macro F1: "F1 of Averages" and "Averaged F1". The first computes the F1 value over the arithmetic means of precision and recall, while the second computes the F1 value for each class, and then averages them. Some prior works adopt the first method. However, this method gives a higher weight to popular classes, and it may thus unfairly yield a high score if the model works well on these popular classes

---

[2] `https://github.com/amirveyseh/MadDog`

Acronym	Long Form	Provenance
ELEC	*Election Law Enforcement Commission*	*Christie, some legislators and the state Election Law Enforcement Commission (**ELEC**), have joined the comptroller in voicing support for the elimination of the loophole.*
ISR	*in-stent restenosis*	*Although conventional stents are routinely used in clinical procedures, clinical data shows that these stents are not capable of completely preventing in-stent restenosis (**ISR**) or restenosis caused by intimal hyperplasia.*
IL-6	*interleukin-6*	*Consistent blood markers in afflicted patients are normal to low white cell counts and elevated interleukin-6 (**IL-6**) levels which, among its many activities, signal the liver to increase synthesis and secretion of CRP.*
PCP	*Planar cell polarity*	*Establishment of photoreceptor cell polarity and ciliogenesis Planar cell polarity (**PCP**)-associated Prickle genes (Pk1 and Pk2) are tissue polarity genes necessary for the establishment of PCP in Drosophila.*
DEP	*dielectrophoretic*	*They included: a particle counter, trypan blue exclusion (Cedex), an in situ bulk capacitance probe, an off-line fluorescent flow cytometer, and a prototype dielectrophoretic (**DEP**) cytometer.*
AQP3	*aquaporin3*	*The laxative effect of bisacodyl is attributable to decreased aquaporin-3 expression in the colon induced by increased PGE2 secretion from macrophages.The purpose of this study was to investigate the role of aquaporin3 (**AQP3**) in the colon in the laxative effect of bisacodyl.*

Table A2: Samples of extracted acronyms, long forms and provenances by using the rule-based algorithm from Schwartz and Hearst (2002).

Acronym	Long Form	Context	BERT	AcroBERT
ECB	*European Central Bank*	*Being made to bring the main road network in Romania in the European corridors. There have been initiated several projects to modernize the network of **ECB** corridors, financed from ispa funds and state-guaranteed loans from international financial institutions. Government seeks external financing or public-private partnerships for other road network upgrades , especially*	*External Commercial Borrowing*	*European Central Bank*
PR	*Public Relations*	*A whistleblower like monologist Mike Daisey gets targeted as a scapegoat who must be discredited and diminished in the public 's eye. More often than not, **PR** is a preemptive process. Celebrity publicists are paid lots of money to keep certain stories out of the news.*	*Preemptive-Resume*	*Public Relations*
PUD	*Peptic Ulcer Disease*	*Tumors cause an overactivation of these hormone-producing glands, leading to serious health problems such as severe **PUD** (due to gastrin hypersecretion, which stimulates secretion of hydrochloric acd).*	*Psychogenic Urinary Dysfunction*	*Peptic Ulcer Disease*
WFC	*Walsall F.C.*	*Injury during a game against Norwich city on the 13 march 2010, forcing him to miss Huddersfields next five games. He made his return against **WFC** on the 13 April 2010 , coming on as a 75th minute substitute and scoring a stoppage time winner to make the score 4a3 to town.*	*Wide Free Choice*	*World Fighting Championships*

Table A3: Case study of predicted results by BERT and AcroBERT.

only (Opitz and Burst, 2019). Therefore, we use the Averaged F1 across classes as our metric, which is more robust towards the error type distribution. That is:

$$\text{Precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}, i \in \{1, 2, ..., n\} \quad \text{(A.1)}$$

$$\text{Recall}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}, i \in \{1, 2, ..., n\} \quad \text{(A.2)}$$

$$\text{F1} = \frac{1}{n} \sum_{i=1}^{n} \frac{2 \times \text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \quad \text{(A.3)}$$

B Additional Experiments

B.1 Ablation Study

In this experiment, we validate the effectiveness of the pre-training strategy in AcroBERT, which adopts a triplet loss with negative samples from ambiguous candidates. Every model is initialized with the parameters of the original BERT, and we use various strategies for pre-training: only Masked Language Model, only Next Sentence Prediction, and the combination of the two and the triplet framework in AcroBERT. Each strategy is pre-trained on our collected corpus for 300K steps and then the corresponding model is evaluated on the three validation sets. The results (in Table A4) show that the strategy of AcroBERT is most beneficial for the acronym disambiguation, as it performs the best on average. The Next Sentence Prediction task is more important than the Masked Language Model task. Even if MLM is not used, the impact on the model is not significant, which means the

Strategy	**General**		**Scientific**		**Biomedical**		**Avg**	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc
Triplet framework in AcroBERT	**61.2**	**64.2**	**18.8**	**18.4**	9.1	10.1	**29.7**	**30.9**
MLM + NSP of the original BERT	45.7	55.7	15.8	14.9	**10.2**	**11.4**	23.9	27.3
only NSP	52.4	58.7	15.1	12.4	6.1	6.5	24.5	25.9
only MLM	11.2	14.4	2.7	3.5	1.4	1.5	5.1	6.5

Table A4: Ablation studies for different pre-training strategies after 300K steps across three validation sets, measured by macro F1 and Accuracy.

original BERT has already learned it well.

B.2 Experiments on Benchmarks with Fewer Candidates

As mentioned before, one drawback of the prior AD benchmark is that the magnitude of the acronym dictionary is small, which is not consistent with practical applications. In this experiment, we therefore valid the performance of AcroBERT on datasets with fewer candidates. The results are shown in Table A1, and we observe that AcroBERT can achieve the best average scores across three datasets again, which demonstrates the generalization capability of our AcroBERT. On the other hand, the lead of our model is not as substantial as before. This is because there are fewer candidates per acronym, and AcroBERT is particularly well-suited for identifying the correct one among a large number of candidates.

B.3 Robustness Evaluation for Many Candidates

Similar to Section 5.2.2, we analyse the robustness of AcroBERT on the other two domains. Each test set is divided evenly into 10 chunks by the number of candidates. The first chunk has the least number of candidates while the last chunk has the most, up to more than 2K. Figure A1 shows the experimental scores on the Scientific and Biomedical test set. We observe that for the first chunk, SciBERT and BioBERT are on par with our AcroBERT. However, AcroBERT outperforms the two significantly when the number of candidates get larger.

B.4 Case Study

Table A3 shows case studies of the outputs by BERT and AcroBERT. BERT often uses the memorized correlations of tokens for reasoning and this can cause errors. For example, *External Commercial Borrowings* are loans in India made by non-resident lenders in foreign currency to Indian borrowers [3]. BERT can determine this correct long form probably with help of the key phrase *"external financing"*. For the third case, *Peptic Ulcer Disease* is more consistent with the input context. However, BERT fails on it while AcroBERT benefits from the pre-training strategy and is able to distinguish different candidates based on contexts. For the fourth sample, both methods fail, most likely because of the low frequency of the long forms and the uninformative contexts.

---

[3] `https://en.wikipedia.org/wiki/External_commercial_borrowing`

Probing Cross-Lingual Lexical Knowledge from Multilingual Sentence Encoders

Ivan Vulić[1] **Goran Glavaš**[2] **Fangyu Liu**[1]
Nigel Collier[1] **Edoardo Maria Ponti**[3,1] **Anna Korhonen**[1]
[1]Language Technology Lab, TAL, University of Cambridge
[2]CAIDAS, University of Würzburg
[3]EdinburghNLP, University of Edinburgh
`{iv250, alk23}@cam.ac.uk`

Abstract

Pretrained multilingual language models (LMs) can be successfully transformed into multilingual sentence encoders (SEs; e.g., LaBSE, xMPNET) via additional fine-tuning or model distillation with parallel data. However, it remains unclear how to best leverage them to represent sub-sentence *lexical* items (i.e., words and phrases) in cross-lingual lexical tasks. In this work, we *probe* SEs for the amount of *cross-lingual lexical knowledge* stored in their parameters, and compare them against the original multilingual LMs. We also devise a simple yet efficient method for *exposing* the cross-lingual lexical knowledge by means of additional fine-tuning through inexpensive contrastive learning that requires only a small amount of word translation pairs. Using bilingual lexical induction (BLI), cross-lingual lexical semantic similarity, and cross-lingual entity linking as lexical probing tasks, we report substantial gains on standard benchmarks (e.g., +10 Precision@1 points in BLI). The results indicate that the SEs such as LaBSE can be 'rewired' into effective cross-lingual lexical encoders via the contrastive learning procedure, and that it is possible to expose more cross-lingual lexical knowledge compared to using them as off-the-shelf SEs. This way, we also provide an effective tool for harnessing 'covert' multilingual lexical knowledge hidden in multilingual sentence encoders.

1 Introduction

Transfer learning with pretrained Language Models (LMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) offers unmatched performance in many NLP tasks (Wang et al., 2019; Raffel et al., 2020). However, despite the wealth of semantic knowledge stored in the pretrained LMs (Rogers et al., 2020; Vulić et al., 2020b), they do not produce coherent and effective sentence representations when used off-the-shelf (Liu et al., 2021c). To this effect, further specialization for sentence-level semantics – not unlike the standard task fine-tuning – is needed (Reimers and Gurevych, 2019; Li et al., 2020; Yan et al., 2021, *inter alia*). LMs get *transformed* into sentence encoders (SEs) via dual-encoder frameworks that leverage contrastive learning objectives (van den Oord et al., 2018; Musgrave et al., 2020), in supervised (i.e., leveraging labeled external data such as NLI or sentence similarity annotations) (Reimers and Gurevych, 2019; Vulić et al., 2021b; Liu et al., 2021a) or, more recently, fully unsupervised fine-tuning (Liu et al., 2021c; Gao et al., 2021) setups.

Following the procedures from monolingual setups, another line of research has been transforming multilingual LMs into *multilingual SEs* (Feng et al., 2022; Reimers and Gurevych, 2020), which enable effective sentence matching and ranking in multiple languages as well as cross-lingually (Litschko et al., 2022). The transformation is typically done by coupling **1)** LM objectives on monolingual data available in multiple languages with **2)** cross-lingual objectives such as Translation Language Modeling (TLM) (Conneau and Lample, 2019) and/or cross-lingual contrastive ranking (Yang et al., 2020). Such multilingual SEs consume a large number of parallel sentences for the latter objectives. Consequently, they outperform multilingual off-the-shelf LMs in cross-lingual sentence similarity and ranking applications (Liu et al., 2021d; Litschko et al., 2022). However, as we show in this work, such multilingual SEs may still lag behind traditional *static* cross-lingual word embeddings (CLWEs) when encoding sub-sentence *lexical items* (e.g., words or phrases) (Liu et al., 2021c) for cross-lingual lexical tasks (e.g., BLI).

In this work, we *probe* multilingual SEs for *cross-lingual lexical knowledge*, relying on standard semantic similarity tasks in cross-lingual setups as our '*lexical probes*' (Vulić et al., 2020b). We demonstrate that, due to their fine-tuning on multilingual and parallel data, they indeed store a

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2089–2105
May 2-6, 2023 ©2023 Association for Computational Linguistics

wealth of such knowledge, much more than what 'meets the eye' when they are used 'off the shelf'. However, this lexical knowledge needs to be *exposed* from the original multilingual SEs, (again) through additional fine-tuning. In other words, we show that multilingual SEs can be 'rewired' into effective cross-lingual *lexical encoders*, as illustrated in Figure 1. This rewiring is again done via a quick and inexpensive contrastive learning procedure: with merely 1k-5k word translation pairs, we successfully convert multilingual SEs into state-of-the-art bilingual lexical encoders for any language pair (present in a specific dataset).[1]

We probe the original LMs and SEs as well as demonstrate the usefulness of the proposed contrastive procedure for 'exposing' cross-lingual lexical knowledge on three standard lexical cross-lingual tasks using standard evaluation data and protocols: BLI, cross-lingual lexical semantic similarity (XLSIM), and cross-lingual entity linking (XL-EL). We show that the 'exposure' procedure is highly effective for both vanilla multilingual LMs (mBERT and XLM-R) and multilingual SEs (LABSE and XMPNET): e.g., we observe ≈+10 Precision@1 points gains on standard BLI benchmarks (Glavaš et al., 2019). Multilingual SEs offer substantially better cross-lingual lexical performance than vanilla LMs, both before and after being subjected to contrastive cross-lingual lexical fine-tuning (see Figure 1). This indicates that it is possible to expose more cross-lingual lexical knowledge from multilingual SEs than from their vanilla LM counterparts, likely owing to their additional exposure to parallel data.

Finally, inspired by Li et al. (2022), we validate that word vectors produced by cross-lingual lexical encoders (i.e., after the contrastive cross-lingual lexical 'exposure') can be effectively interpolated with static CLWEs (Artetxe et al., 2018) and offer even stronger performance in cross-lingual lexical tasks. Encouragingly, our cross-lingual lexical specialization of multilingual SEs (as well as the further interpolation with static CLWEs), yields particularly massive performance gains for pairs of low-resource languages, as demonstrated on the low-resource BLI benchmark (Vulić et al., 2019).

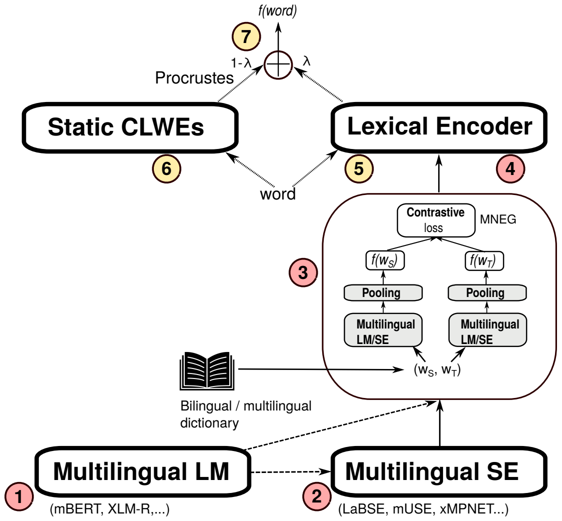


Figure 1: Illustration of the pipeline of exposing cross-lingual lexical knowledge from multilingual language models (LMs) and sentence encoders (SEs) (§2). Multilingual LMs (①) can be transformed into multilingual SEs (②) as done in previous work (Reimers and Gurevych, 2020; Feng et al., 2022). A contrastive cross-lingual lexical fine-tuning procedure (③) (requiring an external bilingual dictionary) can be applied on both ① and ②, yielding a fine-tuned cross-lingual lexical encoder (④). At inference, a word/phrase is encoded by the lexical encoder (⑤). In addition, its encoding can be interpolated with the corresponding static (cross-lingual) word embedding (⑥), producing the final embedding of the word/phrase (⑦). Before the interpolation, static CLWEs must be mapped into the vector space of the lexical encoder (④): to this end, we learn the standard orthogonal (Procrustes) projection matrix.

2 From Multilingual Sentence Encoders to Cross-Lingual Lexical Encoders

Motivation. The motivation for this work largely stems from the research on *probing and analyzing* pretrained LMs for various types of knowledge they (implicitly) store in their parameters (Ethayarajh, 2019; Jawahar et al., 2019; Rogers et al., 2020). In this paper, we focus on a *particular knowledge type*: cross-lingual lexical knowledge, and its extraction from multilingual LMs and SEs. The work combines two research threads, being inspired by the work on probing monolingual PLMs for lexical knowledge (Vulić et al., 2020b), as well as on interpreting representations in multilingual PLMs (Bjerva et al., 2019; Libovický et al., 2020; Beinborn and Choenni, 2020; Deshpande et al., 2022; Chai et al., 2022, *inter alia*).

Previous work also tried to prompt multilingual LMs for word translations via masked natural lan-

---

[1] Note that this is a typical requirement of standard mapping-based approaches for learning static cross-lingual word embeddings which excel in the BLI task (Mikolov et al., 2013; Conneau et al., 2018; Glavaš and Vulić, 2020).

guage templates (Gonen et al., 2020) and extract type-level word embeddings from LMs (i) directly without context (Vulić et al., 2020a, 2021a) or (ii) by averaging contextual embeddings over a large auxiliary corpus in the target language (Bommasani et al., 2020; Litschko et al., 2022). This existing body of work **1)** demonstrated that even sophisticated templates and extraction strategies cannot outperform cross-lingual word embedding spaces (e.g., induced from monolingual fastText vectors) in cross-lingual lexical tasks such as BLI (Vulić et al., 2020b) and **2)** did not attempt to expose cross-lingual lexical knowledge from multilingual SEs and compare it against the (same type of) knowledge extracted from vanilla multilingual LMs.

Multilingual Sentence Encoders. Off-the-shelf LMs contextualize (sub)word representations, but are unable to encode the precise meaning of input text out of the box. SEs – LMs fine-tuned via sentence-level objectives – in contrast, directly produce a precise semantic encoding of input text. A large body of work focuses on inducing multilingual encoders that capture sentence meaning across languages (Artetxe and Schwenk, 2019; Feng et al., 2022; Yang et al., 2020, *inter alia*).

The most popular approach obtains multilingual SEs (Reimers and Gurevych, 2020) by distilling the knowledge from the monolingual English SE teacher (trained on English semantic similarity and NLI data) into multilingual LM student (e.g., mBERT), using parallel sentences to guide the distillation process. SEs, being specialized for sentence similarity, encode sentence meaning more accurately and are useful in various (unsupervised) text similarity and ranking tasks, monolingually and across languages (Artetxe and Schwenk, 2019).

While SEs' primary purpose is sentence encoding, they can, in principle, be applied to sub-sentential text: words and phrases. In this work, we show that multilingual SEs can be turned into effective cross-lingual lexical encoders. We achieve this through additional cross-lingual lexical fine-tuning (Vulić et al., 2021a), requiring as supervision only a small set of word translation pairs.

2.1 Cross-Lingual Lexical Fine-Tuning

For a given language pair L_s-L_t, our contrastive cross-lingual lexical fine-tuning of multilingual encoders (LMs and SEs alike) requires a dictionary spanning N (typically $N \leq$ 5,000) word translation pairs, $\mathcal{D} = \{(w_{i,s}, v_{i,t})\}_{i=1}^{N}$.[2] We consider the translation pairs from $\mathcal{D}$ to be *positive examples* for the contrastive fine-tuning procedure. For each of the N source language words in the dictionary ($w_{i,s}$), we precompute a set of K hard negative samples: these are the L_t words that are the closest to $w_{i,s}$ in the representation space of the multilingual encoder, but not its direct translation $v_{i,t}$. For one-to-many and many-to-many seed dictionaries $\mathcal{D}$, the set K does not contain any L_t word paired with $w_{i,s}$. Let $f_\theta(\cdot)$, be the encoding function of the multilingual LM/SE, with θ as parameters, and let $S(\cdot,\cdot)$ be a function of similarity between two vectors. For a source word $w_{i,s}$, we select as hard negatives words v_t from L_t that have the highest $S(f_{\theta_0}(w_{i,s}), f_{\theta_0}(v_t))$ score, with θ_0 as the original encoder's parameters, before fine-tuning.

We encode all training words – those from the seed dictionary $\mathcal{D}$ and (at most) $N \cdot K$ precomputed hard L_t negatives – independently and *in isolation*. Concretely, for an input w with M subword tokens $[sw_1] \ldots [sw_M]$, we feed the sequence $[SPEC1][sw_1] \ldots [sw_M][SPEC2]$ into the multilingual encoder (with $[SPEC1]$ and $[SPEC2]$ as encoder's sequence start and end tokens, resp.), and take the average of the transformed representation (from the last Transformer layer) of the w's subword tokens as the w's encoding $f_\theta(w)$. Put simply, we process sub-sentential text input in the same way that multilingual SEs handle sentence-level input. We experimented with other encoding strategies from prior work, e.g., taking the representation of the sequence start token $[SPEC1]$ (Liu et al., 2021b; Li et al., 2022); in preliminary experiments, however, we obtained the best results by averaging subword representations.[3]

Following common practice in contrastive learning (Henderson et al., 2019; Vulić et al., 2021a), we define S as the scaled cosine similarity: $S(f_\theta(w_i), f_\theta(w_j)) = C \cdot cos(f_\theta(w_i), f_\theta(w_j))$, with C as the scaling constant. We then train in batches of B translation pairs, with the variant of the widely used multiple negatives ranking loss (MNEG) (Cer et al., 2018; Henderson et al., 2019, 2020) as the fine-tuning objective:

---

[2] Note that such bilingual dictionaries are one of the most widespread and cheapest-to-obtain resources in multilingual NLP (Ruder et al., 2019; Wang et al., 2022).

[3] Note that $[SPEC1]$ and $[SPEC2]$ are placeholders for the encoder's special tokens: e.g., in case of multilingual BERT [SPEC1] is the [CLS] token, while [SPEC2] is the [SEP] token.

$$
\begin{aligned}
\mathcal{L} = -&\sum_{i=1}^{B} S(f_\theta(w_i), f_\theta(v_i)) && \text{(positives)} \\
+&\sum_{i=1}^{B} \log \sum_{j=1, j\neq i}^{B} e^{S(f_\theta(w_i), f_\theta(v_j))} && \text{(in-batch negatives)} \\
+&\sum_{i=1}^{B} \log \sum_{k=1}^{K} e^{S(f_\theta(w_i), f_\theta(v_{k,i}))} && \text{(hard negatives)}
\end{aligned}
$$

where $v_{k,i}$ denotes the k-th hard negative from the language L_t for the L_s word w_i. MNEG combines the K hard negatives per each positive example with B-1 in-batch negatives (i.e., for a source language word $w_{i,s}$, each target language word $v_{j,t}$, $j \neq i$ from B is used as an in-batch negative of $w_{i,s}$). MNEG aims to reshape the representation space of the encoder by simultaneously (a) maximising the similarity for positive pairs – i.e., bringing closer together ('attracting') the words from the positive pairs and (b) minimising the similarity for (both in-batch and hard) negative pairs – i.e., pushing ('repelling') the words from negative pairs further away from each other).[4]

2.2 Interpolation with Static CLWEs

Li et al. (2022) recently showed that further performance benefits in the BLI task might be achieved by combining the type-level output of the encoding function *f* with static CLWEs, but they experimented only with multilingual LMs, and limited their analyses to the BLI task.

Static CLWEs and multilingual encoder-based representations of the same set of words can be seen as two different views of the same data point. Following Li et al. (2022), we learn an additional linear orthogonal mapping from the static cross-lingual WE space – e.g., a CLWE space induced from monolingual fastText embeddings (Bojanowski et al., 2017) using VECMAP (Artetxe et al., 2018) – into the cross-lingual space spanned by the multilingual encoder. The mapping transforms ℓ_2-normed d_1-dimensional static CLWEs into d_2-dimensional cross-lingual WEs obtained through the multilingual encoder (fine-tuned f_θ or original f_{θ_0}).

Learning the linear map $\boldsymbol{W} \in \mathbb{R}^{d_1 \times d_2}$, when $d_1 < d_2$,[5] is formulated as a Generalized Procrustes problem (Schönemann, 1966; Viklands, 2006). It operates on all (i.e., both L_s and L_t) words from the seed translation dictionary $\mathcal{D}$. To learn the mapping $\boldsymbol{W}$, for pairs from $\mathcal{D}$ we decouple L_s words $w_{i,s}$ from their L_t translations $v_{i,t}$ to create vector pairs $(clwe(w_{i,s}), f_\theta(w_{i,s}))$ and $(clwe(v_{i,t}), f_\theta(v_{i,t}))$ – with $clwe(w)$ as the static CLWE of w (e.g., its VECMAP embedding), and $f_\theta(w)$ its encoder-based representation – based on which we learn of the orthogonal mapping $\boldsymbol{W}$ (the so-called Procrustes method gives a closed-form solution). Unless noted otherwise, a final representation of an input word w is then computed as:

$$
(1-\lambda)\frac{clwe(w)\boldsymbol{W}}{\|clwe(w)\boldsymbol{W}\|_2} + \lambda\frac{f_\theta(w)}{\|f_\theta(w)\|_2}, \qquad (1)
$$

where λ is a tunable interpolation hyper-parameter, $clwe(w)$ denotes the static CLWE of w, and $f_\theta(w)$ the representation of w obtained with the (contrastively fine-tuned or original) multilingual LM/SE. This simple procedure yields an 'interpolated' shared cross-lingual WE space.

3 Experimental Setup

Multilingual Sentence Encoders. We probe two widely used multilingual SEs: **1)** Language-agnostic BERT Sentence Embedding (**LABSE**) (Feng et al., 2022) which adapts pretrained multilingual BERT (**MBERT**) (Devlin et al., 2019) into a multilingual SE; **2)** Multilingual **XMPNET** is a distillation-based adaptation (Reimers and Gurevych, 2020) of **XLM-R** (Conneau et al., 2020) as the student model into a multilingual SE, based on the monolingual English MPNet encoder (Song et al., 2020) as the teacher model. LABSE is the current state-of-the-art multilingual SE and supports 109 languages, while XMPNET is the best-performing multilingual SE in the Sentence-BERT repository (Reimers and Gurevych, 2019): For further technical details regarding the models in our comparison, we refer to the original papers.

Along with LABSE and XMPNET as SEs, we experiment with the original multilingual LMs – mBERT and XLM-R – using the the same training and evaluation protocols (see Figure 1 and §2), aiming to quantify: (i) the extent to which cross-lingual lexical knowledge can be exposed from LMs that have *not* been specialized for sentence-level semantics, as well as (ii) the increase in quality of lexical knowledge brought about with sentence-level specialisation (i.e., when multilingual LMs get transformed into multilingual SEs).

---

[4] In practice, we rely on the implementation of the MNEG loss from the SBERT repository `www.sbert.net` (Reimers and Gurevych, 2019); the default value $C = 20$ is used.

[5] The assumption $d_1 < d_2$ typically holds as fastText WEs are 300-dimensional while the dimensionality of standard multilingual LMs and SEs is $d_2 = 768$ or $d_2 = 1,024$.

Evaluation Tasks. We evaluate on the standard and diverse cross-lingual lexical semantic tasks treated as 'cross-lingual lexical probes'. In other words, we fine-tune the models to steer them towards becoming better lexical encoders and then we check how well they fare across a set of representative (intrinsic) lexical tasks which could be seen as such 'lexical probes'.

Task 1: Bilingual Lexicon Induction (BLI), a standard task to assess the "semantic quality" of static cross-lingual word embeddings (CLWEs) (Ruder et al., 2019), allows us to **1)** directly assess the extent to which cross-lingual word translation knowledge can be exposed from multilingual LMs and SEs and **2)** immediately test the ability to transform multilingual sentence encoders into bilingual lexical encoders. We run a series of BLI evaluations on two standard BLI benchmarks. **1)** GT-BLI (Glavaš et al., 2019), constructed semi-automatically from Google Translate, comprises 28 language pairs with a good balance of typologically similar and distant languages (Croatian: HR, English: EN, Finnish: FI, French: FR, German: DE, Italian: IT, Russian: RU, Turkish: TR). **2)** PanLex-BLI (Vulić et al., 2019) focuses on BLI evaluation for lower-resource languages, deriving training and test data from PanLex (Kamholz et al., 2014). We evaluate on 10 pairs comprising the following five typologically and etymologically diverse languages: Bulgarian (BG), Catalan (CA), Estonian (ET), Hebrew (HE), and Georgian (KA).

Standard BLI setups and data are adopted: 5k training word pairs are used as seed dictionary $\mathcal{D}$, and another 2k pairs as test data. Note that $\mathcal{D}$ is used to (i) contrastively fine-tune multilingual encoders (§2.1), (ii) learn the (baseline) static VecMap CLWE space, as well as to (iii) learn the projection between the static CLWE space and the representation spaces of multilingual encoders required to obtain the interpolated representations (§2.2). The evaluation metric is standard Precision@1 (P@1).[6] For PanLex-BLI, we also run experiments using smaller $\mathcal{D}$, spanning 1k pairs.

Task 2: Cross-Lingual Lexical Semantic Similarity (XLSIM) tests the extent to which lexical representations can capture the (human perception of) fine-grained semantic similarity of words across languages. We use the comprehensive XLSIM benchmark Multi-SimLex (Vulić et al., 2020a), which comprises cross-lingual datasets of 2k-4k scored word and phrase pairs over 66 language pairs. We evaluate on a subset of language pairs shared with the GT-BLI dataset: EN, FI, RU, FR.

The evaluation metric is the standard Spearman's rank correlation between the average of gold human-elicited XLSIM scores for word pairs and the cosine similarity between their respective word representations. To avoid any test data leakage, we remove all XLSIM test pairs from the bilingual dictionary $\mathcal{D}$ prior to fine-tuning and CLWE mapping.

Task 3: Cross-Lingual Entity Linking (XEL) is a standard task in knowledge base (KB) construction (Zhou et al., 2022), where the goal is to link an entity mention in any language to a corresponding entity in an English KB or in a language-agnostic KB.[7] We evaluate on the cross-lingual biomedical entity linking (XL-BEL) benchmark of Liu et al. (2021d): it requires the model to link an entity mention to entries in UMLS (Bodenreider, 2004), a language-agnostic medical knowledge base. We largely follow the XL-BEL experimental setup of Liu et al. (2021d) and probe the encoders first *without* any additional task-specific fine-tuning on UMLS data, and then *with* subsequent UMLS fine-tuning (i) only on the EN UMLS data, (ii) on all the UMLS data in 10 languages of the XL-BEL dataset.[8] Due to a large number of experiments, we again focus on the subset of languages in XL-BEL shared with GT-BLI: EN, DE, FI, RU, TR.

Static CLWEs and Word Vocabularies. As monolingual static WEs, we select CommonCrawl fastText vectors (Bojanowski et al., 2017) of the top 200k most frequent word types in the training data, following prior work on learning static CLWEs (Conneau et al., 2018; Artetxe et al., 2018; Heyman et al., 2019).[9] Static CLWEs are then induced via the standard and popular supervised mapping-based VecMap method (Artetxe et al., 2018), leveraging the seed dictionary $\mathcal{D}$. These CLWEs are used for interpolation with encoder-based WEs (see §2.2) but also as the baseline approaches for BLI and XLSIM tasks. We compute the type-level WEs from multilingual LMs and SEs for the same

---

[6] We observed very similar performance trends for P@5 and Mean Reciprocal Rank (MRR) as BLI measures.

[7] Following prior work (Liu et al., 2021b; Zhou et al., 2022), XEL in this work also refers only to entity mention *disambiguation*; it does not cover the mention detection subtask.

[8] See (Liu et al., 2021d) for additional details.

[9] CommonCrawl-based fastText WEs typically outperform other popular choice for monolingual WEs: Wikipedia-based fastText (Glavaš et al., 2019; Li et al., 2022). We note that the main trends in our results also extend to the Wiki-based WEs.

200K most frequent words of each language.

Technical Details and Hyperparameters. The implementation is based on the SBERT framework (Reimers and Gurevych, 2019), using the suggested settings: AdamW (Loshchilov and Hutter, 2018); learning rate of $2e\text{-}5$; weight decay rate of 0.01. We run contrastive fine-tuning for 5 epochs with all the models, with the batch size of $B = 128$ positives for MNEG. The number of hard negatives per each positive is set to $K = 10$ (see §2.1).[10] Since standard BLI and XLSIM datasets lack a validation portion (Ruder et al., 2019), we follow prior work (Glavaš et al., 2019) and tune hyperparameters on a *single* language pair from each dataset, and use those values in all other runs. The randomly selected language pairs are EN-TR for GT-BLI and CA–ET for PanLex-BLI.

All reported scores are the averages over 5 runs with 5 fixed random seeds.

Model Configurations. They are labelled as follows: **ENC-{noCL,+CL}** (λ), where (i) ENC denotes the input multilingual Transformer, which can be a multilingual LM (MBERT, XLM-R), or a multilingual SE (LABSE, XMPNET), (ii) 'noCL' refers to using the input model 'off-the-shelf' without any contrastive lexical fine-tuning, while '+CL' variants apply the contrastive fine-tuning, and (iii) λ is the factor that defines the interpolation with the static CLWE space, obtained with VECMAP (see Figure 1 and §2.2). Note that $\lambda = 1.0$ implies no interpolation with static CLWE space, i.e., WEs come purely from the multilingual LM/SE.

Important Disclaimer. We note that the main purpose of the chosen evaluation tasks and experimental protocols is not necessarily achieving state-of-the-art performance, but rather probing different model variants in different cross-lingual lexical tasks, and offering fair and insightful comparisons.

4 Results and Discussion

Bilingual Lexicon Induction (BLI). Table 1 displays our main BLI results, aggregated over all 28 language pairs of GT-BLI. Two trends hold across the board. First, multilingual SEs, LABSE and XMPNET, substantially outperform their multilingual LM counterparts, MBERT and XLM-R. The gains are visible in all four experimental configurations (with/without contrastive cross-lingual lexical specialisation × with/without interpolation with the VECMAP CLWE space). This confirms our intuition that multilingual SEs, having been (additionally) trained on parallel data (Feng et al., 2022; Reimers and Gurevych, 2020), should better reflect the cross-lingual alignments at the lexical level than off-the-shelf multilingual LMs, which have not been exposed to any cross-lingual signal in pretraining. The poor cross-lingual lexical alignment in the representation spaces of MBERT and XLM-R also reflects in the fact that with those encoders, we only surpass the baseline VECMAP performance by a small margin (+1.1 for XLM-R, +1.6 for MBERT) after subjecting them to contrastive lexical fine-tuning *and* interpolating their word encodings with VECMAP WEs.

The behavior of SEs, on the other hand, is much more favorable. LABSE, for example, surpasses baseline VECMAP performance with interpolation alone, even without the contrastive lexical fine-tuning. When contrastively fine-tuned (and then interpolated with VECMAP) both LABSE and XMPNET surpass the baseline VECMAP performance by a much wider margin (+6.4 and +5.2, respectively). This implies that contrastive fine-tuning exposes more of the high quality cross-lingual lexical knowledge from multilingual SEs.

Both (i) contrastive cross-lingual lexical learning (+CL) and (ii) interpolation with VECMAP consistently improve the performance for all four encoders: we reach peak scores by combining contrastive fine-tuning and interpolation with static CLWEs (+CL (λ)). Contrastive fine-tuning crucially contributes to the overall performance: compared to interpolation alone (noCL (λ)), +CL (λ) brings an average gain of over 6 BLI points.

Table 2 shows the BLI results on 10 low(er)-resource language pairs from PanLex-BLI. While overall relative trends are similar to those observed for high(er)-resource languages from GT-BLI (Table 1), the gains stemming from cross-lingual contrastive lexical fine-tuning are substantially larger in this case. The best-performing configuration – contrastive fine-tuning and interpolation (+CL (0.4)) applied on LaBSE – surpasses VECMAP by 11 BLI points on average (compared to 6 points on GT-BLI), with gains for some language pairs (e.g., HE-KA, ET-HE) approaching the impressive margin of 20 BLI points. This finding indicates that cross-lingual lexical knowledge stored in multilingual SEs is even more crucial when dealing

---

[10] We also tested K=$\{20, 30, 50\}$. They slow down fine-tuning while yielding small-to-negligible performance gains.

Multilingual LMs		mBERT				XLM-R			
Config →	VecMap	noCL (1.0)	noCL (λ)	+CL (1.0)	+CL (λ)	noCL (1.0)	noCL (λ)	+CL (1.0)	+CL (λ)
[BLI] λ=0.3	42.7	9.0	39.2	22.3	44.3	6.4	33.7	21.2	43.8
[XLSIM] λ=0.5	45.8	5.7	35.4	38.4	48.1	1.7	23.5	46.1	51.8
Multilingual SEs		LaBSE				xMPNET			
Config →	VecMap	noCL (1.0)	noCL (λ)	+CL (1.0)	+CL (λ)	noCL (1.0)	noCL (λ)	+CL (1.0)	+CL (λ)
[BLI] λ=0.3	42.7	21.4	45.7	30.8	49.1	17.0	41.7	28.6	47.9
[XLSIM] λ=0.5	45.8	50.4	54.9	48.8	54.1	51.3	56.6	49.6	54.5

Table 1: (a) P@1 scores ($\times$100%) averaged across all 28 language pairs in the GT-BLI dataset ([BLI] rows); (b) Spearman's ρ correlation scores ($\times$100) averaged across a subset of 6 language pairs from Multi-SimLex ([XLSIM rows]). See §3 for the description of different model configurations/variants. $|\mathcal{D}| = 5k$. The number in the parentheses denotes the value for λ (see §3), which differs between the two tasks (0.3 for BLI and 0.5 for XLSIM). The λ value of 1.0 effectively means 'no interpolation' with static VecMap CLWEs. Individual results per each language pair in both tasks and with other λs are in Appendix B and Appendix C.

		mBERT		XLM-R		LaBSE		xMPNET	
Pair ↓ / Config →	VecMap	+CL (1.0)	+CL (0.4)	+CL (1.0)	+CL (0.4)	+CL (1.0)	+CL (0.4)	+CL (1.0)	+CL (0.4)
BG–CA	34.4	9.6	31.9	13.2	33.3	17.9	**38.0**	15.9	35.7
BG–ET	30.0	17.1	32.6	21.3	34.1	29.9	**42.7**	26.1	38.9
BG–HE	26.1	9.9	21.1	10.5	26.3	23.7	**37.2**	10.9	27.2
BG–KA	26.8	16.0	29.8	15.9	30.5	27.2	**37.4**	18.7	32.4
CA–ET	26.3	26.8	32.9	23.5	34.1	28.8	**38.6**	29.0	38.9
CA–HE	23.3	2.3	12.5	4.9	18.5	12.7	**28.9**	8.5	22.7
CA–KA	20.7	1.5	10.3	4.7	20.0	9.6	**26.1**	6.4	21.8
ET–HE	18.6	15.0	21.9	17.7	26.0	31.0	**37.8**	18.5	27.0
ET–KA	16.5	7.2	18.2	12.7	24.3	19.3	**30.3**	12.5	25.8
HE–KA	12.7	15.6	23.8	13.3	23.1	25.3	**30.2**	15.1	24.4
Average	23.5	12.1	23.5	13.8	27.0	22.5	**34.7**	16.2	29.5

Table 2: P@1 scores over a representative subset of 10 language pairs from the PanLex-BLI dataset of Vulić et al. (2019). See §3 for the description of different model configurations/variants. $|\mathcal{D}| = 5k$. Highest scores per row are in **bold**. Respective average scores for the *noCL (1.0)* config (i.e., without contrastive learning and without interpolation with static VecMap CLWEs) are: 4.2 (mBERT), 3.1 (XLM-R), 17.0 (LaBSE), 8.3 (xMPNET).

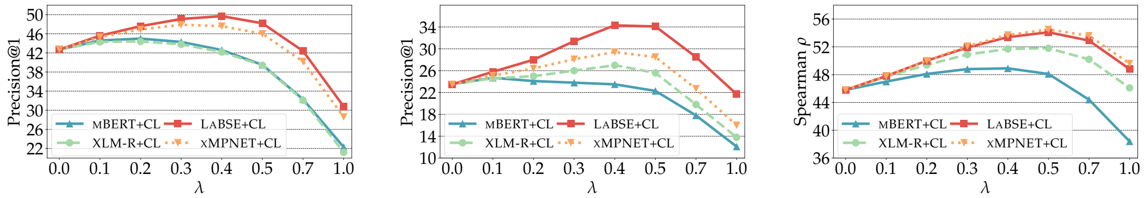
		LaBSE		
Pair ↓	VecMap	noCL (1.0)	+CL (1.0)	+CL (0.5)
BG–CA	15.2	14.0	18.0	**28.9**
BG–ET	12.5	20.3	25.5	**35.1**
BG–HE	5.6	18.3	20.8	**24.7**
BG–KA	9.1	16.0	21.6	**29.7**
CA–ET	9.8	24.8	25.6	**31.2**
CA–HE	5.0	10.6	12.2	**15.6**
CA–KA	5.5	5.7	8.3	**14.8**
ET–HE	3.1	**27.7**	25.1	25.4
ET–KA	4.6	13.2	16.0	**21.1**
HE–KA	3.2	19.0	22.0	**25.4**
Average	7.4	17.0	19.5	**25.2**

Table 3: P@1 scores over 10 language pairs from the PanLex-BLI dataset of Vulić et al. (2019) when $|\mathcal{D}| = 1k$, with different model variants based on LaBSE (see §3). Highest scores per row are in **bold**.

with lower-resource languages.

In Table 3 we compare the results of LaBSE (as the best-performing multilingual SE) against VecMap on PanLex-BLI in a scenario with less external bilingual supervision: $|\mathcal{D}| = 1k$. Interestingly, in this setup LaBSE already substantially outperforms VecMap out of the box (noCL (1.0)); contrastive lexical fine-tuning (+CL (1.0)) and interpolation with VecMap embeddings (+CL (0.5)) again bring further substantial gains, and we again observe a strong synergistic effect of the two components: +CL (1.0) yields gains over noCL (1.0) for 9/10 language pairs, and +CL (0.5) results in further boosts for all 10 pairs. Further, the contrastively fine-tuned LaBSE seems to be much more resilient to training data scarcity than VecMap: reduction of the training dictionary size from 5k to 1k reduces the performance of LaBSE +CL (λ) by 27% (from 34.7 to 25.2 P@1 points) compared to a massive performance drop of almost 70% for VecMap (from 23.5 to mere 7.4 P@1). In sum, the BLI results already indicate the wealth of lexical knowledge 'hidden' in multilingual SEs, which must be 'exposed to surface'.

Cross-Lingual Lexical Semantic Similarity (XLSIM). The average XLSIM results are summarized in Table 1. They again corroborate one of the main findings from BLI experiments: multilingual SEs store more cross-lingual lexical knowledge than



(a) GT-BLI (b) PanLex-BLI (c) Multi-SimLex (XLSIM)

Figure 2: Average scores across different interpolation values λ for the BLI task on **(a)** GT-BLI and **(b)** PanLex-BLI, and **(c)** the XLSIM task on Multi-SimLex. $|\mathcal{D}| = 5k$. Additional results are in Appendix B and C.

Config ↓ / Language L_t →	**DE**		**FI**		**RU**		**TR**	
	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
XLM-R +noCL	0.0	0.1	0.1	0.2	0.1	0.2	0.4	0.5
XLM-R +noCL+UMLS$_{\text{EN}}$	27.6	32.0	12.2	14.7	21.8	25.9	29.3	35.9
XLM-R +noCL+UMLS$_{all}$	31.8	37.3	18.6	22.2	35.4	41.2	42.8	48.9
XLM-R +CL	14.1	17.1	5.0	6.5	8.7	11.2	21.6	27.1
XLM-R +CL+UMLS$_{\text{EN}}$	25.2	29.0	12.1	14.1	19.8	25.0	31.1	36.1
XLM-R +CL+UMLS$_{all}$	32.1	36.7	19.1	23.8	34.9	42.4	43.4	49.0
XMPNET +noCL	19.5	25.9	12.2	14.8	19.2	24.3	28.9	36.3
XMPNET +noCL+UMLS$_{\text{EN}}$	25.1	29.2	17.8	21.5	21.9	26.9	30.0	36.5
XMPNET +noCL+UMLS$_{all}$	**33.4**	37.8	**23.6**	**27.7**	**39.8**	45.4	**44.6**	**51.4**
XMPNET +CL	20.8	26.5	9.1	12.5	12.8	17.1	30.4	36.5
XMPNET +CL+UMLS$_{\text{EN}}$	25.1	28.7	11.4	14.0	21.8	27.2	31.0	37.5
XMPNET +CL+UMLS$_{all}$	32.0	**38.7**	22.9	27.5	39.2	**45.7**	44.3	51.0

Table 4: A summary of results in the XEL task on the biomedical XL-BEL benchmark of Liu et al. (2021d). We show the results of the better-performing LM (XLM-R), and the more lightweight multilingual SE (XMPNET).

multilingual LMs. This is validated by substantial gains of SEs over corresponding LMs across all configurations in Table 1. Interestingly, due to their contrastive learning objectives on sentence-level parallel data (Feng et al., 2022), LABSE and XMPNET provide very strong XLSIM results when used off-the-shelf (noCL (1.0)), outperforming the CLWE VECMAP embeddings. Contrastive lexical fine-tuning with 5k word translation pairs (+CL (1.0)) in this case does not bring any gains. However, the opposite is true for multilingual LMs: contrastive cross-lingual lexical fine-tuning on only 5k word translation pairs brings large benefits in the XLSIM task (e.g., compare noCL (1.0) and +CL (1.0) configurations for MBERT and XLM-R), and turns them into more effective lexical encoders. This result corroborates a similar finding from prior work in monolingual setups (Vulić et al., 2021a). Finally, interpolation with static CLWEs benefits the final XLSIM performance of all four underlying multilingual encoders: interpolated vectors ($\lambda = 0.5$) yield highest scores across the board, substantially surpassing gains both VECMAP and WEs from fine-tuned encoders ($\lambda = 1.0$).

Interpolation with Static CLWEs. A more detailed analysis over different λ values for BLI and XLSIM, summarized in Figure 2, reveals that interpolation can bring large performance gains, especially for λ in the $[0.3, 0.5]$ interval. The optimal λ value is, however, task- and even dataset-dependent. For instance, for low-resource BLI on PanLex-BLI more knowledge comes from the multilingual encoders as VECMAP CLWEs are of lower quality for such languages: in consequence, the optimal λ value 'moves away' from the static CLWEs towards encodings obtained by fine-tuned multilingual SE. We also note that larger benefits from interpolation are observed when VECMAP CLWEs are combined with contrastively fine-tuned multilingual SEs than with LMs: cf., the large gains in Figure 2b and in Table 3 for the LABSE +CL model variant.

Cross-Lingual Entity Linking (XEL). Experiments on XL-BEL (Liu et al., 2021d), summarized in Table 4, demonstrate that additional contrastive tuning with word or phrase pairs can greatly boost performance of multilingual LMs: even fine-tuning with 5k word translation pairs without any domain-specific knowledge yields strong benefits for XLM-R. As expected, using a much larger and domain-specific external database UMLS yields much higher scores and is more crucial for performance. In fact, contrastively fine-tuning on UMLS generally improves XEL performance with all four underlying models. Again, we observe that SE-based (XMPNET) configurations outperform the respective LM-based (XLM-R) configurations across the board. This finding again indicates that multilingual SEs store more cross-lingual lexical knowledge than multilingual LMs: this difference is particularly salient when the models are used off-the-shelf without any additional contrastive fine-tuning, and xMPNET retains the edge over XLM-R even after task-specific fine-tuning with the UMLS-based domain-specific knowledge.

What is more, for FI, RU, and TR, the multilingual xMPNET-based variants match or surpass the performance of respective XEL models trained on top of monolingual LMs (e.g., for FI, a model based on the Finnish BERT) reported by Liu et al. (2021d). This further validates our hypothesis that multilingual SEs store rich multilingual lexical knowledge, which is then also exposed in domain-specific (multilingual) UMLS fine-tuning, yielding performance gains. Contrastive fine-tuning on UMLS synonyms (+CL+UMLS variants) expectedly outpeforms fine-tuning on (5k) general-domain word translations (+CL), indicating that in specialized domains, if available, in-domain cross-lingual lexical signal should be exploited.

4.1 Summary and Discussion

Since the exposure of knowledge is done through very knowledge-light tuning which also improves representations of lexical items not covered in the dictionaries used for the adaptive fine-tuning, this suggests that the knowledge is stored in the parameters of the large models (both LMs and SEs), but it is more easily 're-purposed' through fine-tuning for SEs. One might posit that the SEs have an 'unfair' advantage over LMs as the primary purpose of the SEs, even before is encoding text items (i.e., sentences) for semantic similarity and search. In this paper, we verify the extent of that advantage for lexical-level encodings/embeddings (as representations of lexical knowledge): while exposing (i.e., re-purposing) works for both model types, they do not reach the same performance peaks and benefits for lexical tasks where such lexical encodings are a paramount. We leave fine-tuning with LM-style objectives for other types of tasks beyond lexical similarity and search for future work.

5 Conclusion and Future Work

We investigated strategies to probe and expose cross-lingual lexical knowledge from pretrained models, including multilingual language models (LMs) and multilingual sentence encoders (SEs). Based on an extensive probing experiments on a suite of cross-lingual lexical tasks, we verified that multilingual SEs (e.g., LaBSE, xMPNET) are superior to multilingual LMs (mBERT, XLM-R) in terms of stored cross-lingual lexical knowledge. We empirically validated that the SEs store more lexical knowledge than 'what meets the eye' when they are used off-the-shelf, but this knowledge must be exposed from them. To this end, we proposed new methods to further fine-tune their representations based on contrastive learning to 'rewire' the models' parameters and transform them from LMs and SEs into more effective cross-lingual lexical encoders. These lexical encoders yield gains for all underlying models, and are especially significant for resource-poor languages and in low-data learning regimes. While this work focused on two widely used state-of-the-art multilingual SEs, the contrastive framework is versatile and model-independent and can be applied on top of other multilingual SEs in future work. We will also investigate other more sophisticated contrastive learning strategies, look into ensembling of knowledge extracted from different SEs, and expand our probing experiments to more tasks and languages.

Acknowledgements

This work has been partially supported by the ERC PoC Grant MultiConvAI (no. 957356) and a Huawei research donation to the University of Cambridge. Ivan Vulić is supported by a personal Royal Society University Research Fellowship (no 221137; 2022-2027). Fangyu Liu is supported by Grace & Thomas C.H. Chan Cambridge Scholarship. The work of Goran Glavaš has been supported by the Multi2ConvAI project of MWK Baden-Württemberg. The work of Edoardo Maria Ponti's work has been supported by the Facebook CIFAR AI Chair program during his affiliation with Mila – Quebec AI Institute. Nigel Collier kindly acknowledges grant-in-aid funding from ESRC (grant number ES/T012277/1). We would like to thank Yaoyiran Li for interesting insights and discussions.

We also thank the three anonymous reviewers for their extremely substantial and in-depth reviews with a large number of helpful suggestions.

Limitations

This work focuses on lexical specialization of multilingual encoders, off-the-shelf LMs (experiments with mBERT and XLM-R Base) and, in particular multilingual encoders specialized for sentence-level semantics (experiments with LaBSE and xMPNET). While these are all widely used models, they are arguably among the smaller pretrained multilingual encoders. Due to computational constraints, we have not evaluated the effectiveness of the proposed cross-lingual lexical specialisation for larger multilingual LMs, e.g., XLM-R-Large (Conneau et al., 2020) or mT5 (Large, XL, and XXL) (Xue et al., 2021). It is possible that these larger multilingual LMs would close (some of) the performance gap w.r.t. multilingual SEs. Such large LMs, however, are effectively available to fewer researchers and practitioners. Our work includes less resource-demanding LMs and SEs, making their lexically specialized variants that we offer more widely accessible.

Lexical input (i.e., words or phrases) are provided to each multilingual encoder fully *"in isolation"* (see §2), without any surrounding context. However, the alternative of using external corpora and *averaging-over-context* (Litschko et al., 2022), which we have not evaluated in this work for clarity and space constraints, might yield slightly improved task performance. Nonetheless, the 'in isolation' approach has been verified in previous work (Vulić et al., 2021a; Litschko et al., 2022; Li et al., 2022) as very competitive, and is more lightweight: **1)** it disposes of any external text corpora and is not impacted by the external data; **2)** it encodes words more efficiently due to the absence of context. Moreover, it allows us to directly study the richness of cross-lingual information stored in the encoders' parameters, and its interaction with additional cross-lingual signal from bilingual lexicons.

The contrastive cross-lingual lexical fine-tuning we proposed in this is work is *bilingual*. It leverages a small bilingual dictionary $\mathcal{D}$ for each language pair and specializes the multilingual encoders (LMs and SEs) independently for each language pair. Assuming interest in cross-lingual lexical tasks between all pairs of N_L languages, this entails $\frac{N_L \cdot (N_L - 1)}{2}$ fine-tuning procedures and as many resulting bilingual models. Although our contrastive fine-tuning is relatively fast and lightweight, given that it leverages at most 5k translation pairs, for large N_L it could easily exceed the computational and time budget for most users. On a high level, our work again outlines the advantages as well as disadvantages between **1)** more versatile massively multilingual models that serve multiple languages without any further adaptation, and **2)** better-performing but typically less modular and less versatile models adapted (i.e., *bilingually specialized*) from the multilingual models (Bapna and Firat, 2019; Parović et al., 2022).

Intuitively, for each bilingually fine-tuned model, we evaluate the performance for that respective language pair. Currently, we do not investigate the spillover effects that a bilingual lexical fine-tuning of multilingual encoders could have on lexical representations of other languages. Such an analysis, planned for future work, would be particularly interesting in the context of low-resource languages, unseen from the point of view of cross-lingual lexical fine-tuning, and in particular closely related low-resource languages. For instance, if we are doing cross-lingual lexical fine-tuning for language pairs involving Turkish, are there spillover benefits for low(er)-resource Turkic languages such as Uyghur or Kazakh?

Finally, we acknowledge that our choice of lexical tasks as probing tasks is non-exhaustive: we put focus on standard tasks from previous work on (multilingual) lexical semantics that are especially convenient as cross-lingual lexical probes: such tasks directly test and compare the quality of cross-lingual lexical representations obtained via different methods; see §4.1 again.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of ACL 2018*, pages 789–798.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the ACL*, 7:597–610.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of EMNLP*, pages 1538–1548.

Lisa Beinborn and Rochelle Choenni. 2020. Semantic drift in multilingual representations. *Computational Linguistics*, 46(3):571–603.

Johannes Bjerva, Robert Östling, Maria Han Veiga, Jörg Tiedemann, and Isabelle Augenstein. 2019. What do language representations really represent? *Computational Linguistics*, 45(2):381–389.

Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the ACL*, 5:135–146.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of ACL 2020*, pages 4758–4781.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of EMNLP 2018*, pages 169–174.

Yuan Chai, Yaobo Liang, and Nan Duan. 2022. Cross-lingual ability of multilingual masked language models: A study of language structure. In *Proceedings of ACL 2022*, pages 4702–4712.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL 2020*, pages 8440–8451.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Proceedings of NeurIPS 2019*, pages 7057–7067.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proceedings of ICLR 2018*.

Ameet Deshpande, Partha Talukdar, and Karthik Narasimhan. 2022. When is BERT multilingual? isolating crucial ingredients for cross-lingual transfer. In *Proceedings of NAACL-HLT 2022*, pages 3610–3623.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of EMNLP 2019*, pages 55–65.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of ACL 2022*, pages 878–891.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of EMNLP 2021*, pages 6894–6910.

Goran Glavaš and Ivan Vulić. 2020. Non-linear instance-based cross-lingual mapping for non-isomorphic embedding spaces. In *Proceedings of ACL 2020*, pages 7548–7555.

Goran Glavaš, Robert Litschko, Sebastian Ruder, and Ivan Vulić. 2019. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In *Proceedings of ACL 2019*, pages 710–721.

Hila Gonen, Shauli Ravfogel, Yanai Elazar, and Yoav Goldberg. 2020. It's not Greek to mBERT: Inducing word-level translations from multilingual BERT. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 45–56.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of EMNLP 2020*, pages 2161–2174.

Matthew Henderson, Ivan Vulić, Daniela Gerz, Iñigo Casanueva, Paweł Budzianowski, Sam Coope, Georgios Spithourakis, Tsung-Hsien Wen, Nikola Mrkšić, and Pei-Hao Su. 2019. Training neural response selection for task-oriented dialogue systems. In *Proceedings of ACL 2019*, pages 5392–5404.

Geert Heyman, Bregt Verreet, Ivan Vulić, and Marie-Francine Moens. 2019. Learning unsupervised multilingual word embeddings with incremental multilingual hubs. In *Proceedings of NAACL-HLT 2019*, pages 1890–1902.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of ACL 2019*, pages 3651–3657.

David Kamholz, Jonathan Pool, and Susan M. Colowick. 2014. PanLex: Building a resource for panlingual lexical translation. In *Proceedings of LREC 2014*, pages 3145–3150.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of EMNLP 2020*, pages 9119–9130.

Yaoyiran Li, Fangyu Liu, Nigel Collier, Anna Korhonen, and Ivan Vulić. 2022. Improving word translation via two-stage contrastive learning. In *Proceedings of ACL 2022*, pages 4353–4374.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2020. On the language neutrality of pre-trained multilingual representations. In *Findings of EMNLP 2020*, pages 1663–1674.

Robert Litschko, Ivan Vulić, Simone Paolo Ponzetto, and Goran Glavaš. 2022. On cross-lingual retrieval with multilingual text encoders. *Information Retrieval*, 25:149–183.

Che Liu, Rui Wang, Jinghua Liu, Jian Sun, Fei Huang, and Luo Si. 2021a. DialogueCSE: Dialogue-based contrastive learning of sentence embeddings. In *Proceedings of EMNLP 2021*, pages 2396–2406.

Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. 2021b. Self-alignment pretraining for biomedical entity representations. In *Proceedings of NAACL-HLT 2021*, pages 4228–4238.

Fangyu Liu, Ivan Vulić, Anna Korhonen, and Nigel Collier. 2021c. Fast, effective, and self-supervised: Transforming masked language models into universal lexical and sentence encoders. In *Proceedings of EMNLP 2021*, pages 1442–1459.

Fangyu Liu, Ivan Vulić, Anna Korhonen, and Nigel Collier. 2021d. Learning domain-specialised representations for cross-lingual biomedical entity linking. In *Proceedings of ACL-IJCNLP 2021*, pages 565–574.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *Proceedings of ICLR 2018*.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint, CoRR*, abs/1309.4168.

Kevin Musgrave, Serge J. Belongie, and Ser-Nam Lim. 2020. A metric learning reality check. In *Proceedings of ECCV 2020*, volume 12370, pages 681–699.

Marinela Parović, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2022. BAD-X: Bilingual adapters improve zero-shot cross-lingual transfer. In *Proceedings of NAACL-HLT 2022*, pages 1791–1799.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of EMNLP 2019*, pages 3982–3992.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of EMNLP 2020*, pages 4512–4525.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: what we know about how BERT works. *Transactions of the ACL*, 8:842–866.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual embedding models. *Journal of Artificial Intelligence Research*, 65:569–631.

Peter H Schönemann. 1966. A generalized solution of the orthogonal Procrustes problem. *Psychometrika*, 31(1):1–10.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. In *Proceedings of NeurIPS 2020*, pages 16857–16867.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Thomas Viklands. 2006. *Algorithms for the weighted orthogonal Procrustes problem and other least squares problems*. Ph.D. thesis, Datavetenskap.

Ivan Vulić, Simon Baker, Edoardo Maria Ponti, Ulla Petti, Ira Leviant, Kelly Wing, Olga Majewska, Eden Bar, Matt Malone, Thierry Poibeau, Roi Reichart, and Anna Korhonen. 2020a. Multi-SimLex: A large-scale evaluation of multilingual and crosslingual lexical semantic similarity. *Computational Linguistics*, 46(4):847–897.

Ivan Vulić, Goran Glavaš, Roi Reichart, and Anna Korhonen. 2019. Do we really need fully unsupervised cross-lingual embeddings? In *Proceedings of EMNLP-IJCNLP 2019*, pages 4407–4418.

Ivan Vulić, Edoardo Maria Ponti, Anna Korhonen, and Goran Glavaš. 2021a. LexFit: Lexical fine-tuning of pretrained language models. In *Proceedings of ACL-IJCNLP 2021*, pages 5269–5283.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020b. Probing pretrained language models for lexical semantics. In *Proceedings of EMNLP 2020*, pages 7222–7240.

Ivan Vulić, Pei-Hao Su, Samuel Coope, Daniela Gerz, Paweł Budzianowski, Iñigo Casanueva, Nikola Mrkšić, and Tsung-Hsien Wen. 2021b. ConvFiT: Conversational fine-tuning of pretrained language models. In *Proceedings of EMNLP 2021*, pages 1151–1168.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of NeurIPS 2019*, pages 3261–3275.

Xinyi Wang, Sebastian Ruder, and Graham Neubig. 2022. Expanding pretrained models to thousands more languages via lexicon-based adaptation. In *Proceedings of ACL 2022*, pages 863–877.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of NAACL-HLT 2021*, pages 483–498.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of ACL-IJCNLP 2021*, pages 5065–5075.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of ACL 2020: System Demonstrations*, pages 87–94.

Wenxuan Zhou, Fangyu Liu, Ivan Vulić, Nigel Collier, and Muhao Chen. 2022. Prix-LM: Pretraining for multilingual knowledge base construction. In *Proceedings of ACL 2022*, pages 5412–5424.

Languages in: GT-BLI, Multi-SimLex, XL-BEL	
EN	English
DE	German
TR	Turkish
FI	Finnish
HR	Croatian
RU	Russian
IT	Italian
FR	French
Languages in: PanLex-BLI	
BG	Bulgarian
CA	Catalan
ET	Estonian
HE	Hebrew
KA	Georgian

Table 5: Languages and their ISO 639-1 codes.

A List of Languages

The list of languages used in this work, along with their ISO 639-1 codes, is available in Table 5.

B BLI Results across Individual Language Pairs

Additional experiments and analyses over individual language pairs and other λ values, which further support the main claims of the paper, have been relegated to the appendix for clarity and compactness of the presentation in the main paper:

Table 6. It provides results over all 28 language pairs in GT-BLI with 2 multilingual LMs and 2 multilingual SEs in the *noCL* variant without contrastive fine-tuning.

Table 7. It provides results over all 28 language pairs in GT-BLI with 2 multilingual LMs and 2 multilingual SEs in the *+CL* variant with contrastive fine-tuning.

Table 8. It provides results over all 28 language pairs in GT-BLI and across different λ values with the LABSE +noCL variant.

Table 9. It provides results over all 28 language pairs in GT-BLI and across different λ values with the LABSE +CL variant.

C XLSIM Results across Individual Language Pairs

Table 10. It provides results over selected 6 language pairs from Multi-SimLex with 2 multilingual LMs and 2 multilingual SEs in the *noCL* variant without contrastive fine-tuning.

Table 11. It provides results over selected 6 language pairs from Multi-SimLex with 2 multilingual LMs and 2 multilingual SEs in the *+CL* variant with contrastive fine-tuning.

Table 12. It provides results over selected 6 language pairs from Multi-SimLex and across different λ values with the XMPNET +noCL variant.

Table 13. It provides results over selected 6 language pairs from Multi-SimLex and across different λ values with the XMPNET +CL variant.

D Models and Evaluation Data

URLs to the models used in this paper are provided in Table 14. Training and test data for all three tasks (BLI, XLSIM, XEL) is available online:

- GT-BLI is available here: `https://github.com/codogogo/xling-eval`
- PanLex-BLI: `https://github.com/cambridgeltl/panlex-bli`
- Multi-SimLex [XLSIM]: `https://multisimlex.com/`
- XL-BEL [XEL]: `https://github.com/cambridgeltl/sapbert`

Our code is based on PyTorch, and relies on the following two widely used repositories:

- `sentence-transformers`: `www.sbert.net`
- `huggingface.co/transformers/`

Pair ↓ / Config →	VECMAP	mBERT		XLM-R		LABSE		xMPNET	
		+noCL (1.0)	+noCL (0.3)	+noCL (1.0)	+noCL (0.3)	+noCL (1.0)	+noCL (0.3)	+noCL (1.0)	+noCL (0.3)
EN–DE	55.6	15.6	50.7	12.7	45.5	25.4	54.1	22.0	47.7
EN–TR	40.4	7.2	34.9	6.0	28.9	23.6	42.1	16.1	33.7
EN–FI	45.6	7.9	38.7	6.6	33.4	19.3	45.1	14.8	39.5
EN–HR	37.5	8.9	31.8	6.8	25.6	24.7	45.3	18.5	38.9
EN–RU	45.6	3.2	40.7	0.9	34.7	24.7	49.9	17.6	41.7
EN–IT	60.2	12.3	57.1	9.3	53.0	26.4	62.3	23.5	58.8
EN–FR	64.1	25.2	62.5	19.5	56.6	34.2	67.5	29.2	61.7
DE–TR	32.5	9.1	28.9	6.9	24.4	17.7	33.0	13.1	29.7
DE–FI	39.7	9.2	34.1	7.3	30.1	16.0	37.4	13.3	34.7
DE–HR	33.3	11.5	31.0	9.7	25.7	19.2	38.5	14.9	34.1
DE–RU	40.0	4.2	36.4	0.9	32.4	14.3	41.5	9.5	37.7
DE–IT	49.5	10.9	45.9	8.1	42.7	19.4	51.4	18.3	49.1
DE–FR	50.0	15.8	49.7	10.5	42.3	22.5	53.2	20.2	49.8
TR–FI	31.3	6.7	26.2	5.2	22.0	15.5	31.7	11.3	30.9
TR–HR	25.4	10.8	24.3	8.3	20.5	18.7	33.1	14.4	28.2
TR–RU	32.9	2.6	29.1	0.8	25.5	14.1	36.9	11.3	33.5
TR–IT	37.1	7.9	34.7	5.5	27.4	17.0	38.9	14.8	38.3
TR–FR	39.4	7.8	37.3	5.7	30.9	20.9	43.1	16.6	39.4
FI–HR	30.4	7.2	26.6	5.5	22.8	17.4	36.4	12.2	32.7
FI–RU	38.2	2.6	34.0	0.9	30.5	15.1	41.0	9.0	37.1
FI–IT	39.9	7.9	36.7	6.8	30.4	18.1	43.4	16.4	41.8
FI–FR	42.8	7.5	38.9	5.9	32.2	18.6	45.9	16.2	42.2
HR–RU	40.6	6.0	35.8	1.6	30.4	24.5	45.7	16.3	41.4
HR–IT	40.4	11.2	39.0	8.4	31.3	24.5	47.9	22.4	44.5
HR–FR	43.6	9.7	42.3	6.1	30.6	25.7	50.0	19.8	43.9
RU–IT	46.6	3.1	42.2	1.5	36.4	21.8	47.6	18.4	46.5
RU–FR	48.7	4.1	44.3	1.5	38.4	26.6	50.9	18.9	47.5
IT–FR	64.1	16.6	62.8	9.3	58.6	33.9	65.6	27.5	63.1
Average	42.7	9.0	39.2	6.4	33.7	21.4	45.7	17.0	41.7

Table 6: Individual P@1 scores (×100%) for all 28 language pairs in the GT-BLI dataset of Glavaš et al. (2019), with multilingual LMs and SEs used 'off-the-shelf' *without contrastive fine-tuning* (§2). See §3 for the description of different model configurations/variants. $|\mathcal{D}| = 5k$. The number in the parentheses denotes the value for λ (see §3): the value of 1.0 effectively means 'no interpolation' with static VECMAP CLWEs.

Pair ↓ / Config →	VECMAP	mBERT		XLM-R		LABSE		xMPNET	
		+CL (1.0)	+CL (0.3)	+CL (1.0)	+CL (0.3)	+CL (1.0)	+CL (0.3)	+CL (1.0)	+CL (0.3)
EN–DE	55.6	26.4	59.2	24.5	56.3	31.6	61.1	30.2	58.7
EN–TR	40.4	17.4	39.3	19.2	42.0	33.1	50.1	30.4	45.7
EN–FI	45.6	18.6	45.6	20.2	45.7	30.8	53.3	28.7	50.4
EN–HR	37.5	23.6	44.3	21.8	44.3	36.9	53.9	31.8	49.6
EN–RU	45.6	23.9	48.9	28.3	48.8	46.4	55.8	37.9	53.1
EN–IT	60.2	26.8	64.0	25.4	61.7	33.3	66.9	33.0	65.3
EN–FR	64.1	34.4	67.7	33.0	65.4	42.1	71.2	39.5	68.5
DE–TR	32.5	19.5	32.8	15.5	33.5	24.4	37.6	22.7	36.2
DE–FI	39.7	20.3	38.5	19.0	37.5	25.0	43.3	24.4	42.1
DE–HR	33.3	24.2	37.3	22.4	36.9	27.9	42.9	27.2	41.8
DE–RU	40.0	21.2	42.0	19.1	41.5	27.5	45.6	26.8	44.1
DE–IT	49.5	23.0	49.9	19.0	48.6	24.6	52.5	25.6	51.4
DE–FR	50.0	27.4	52.5	24.3	51.5	31.3	54.5	30.0	53.9
TR–FI	31.3	18.0	29.6	15.9	31.7	22.2	34.9	22.4	35.9
TR–HR	25.4	21.8	30.2	19.0	30.8	27.4	36.8	26.0	36.4
TR–RU	32.9	16.2	33.9	15.7	33.5	24.7	37.4	24.7	37.8
TR–IT	37.1	17.4	37.8	15.4	36.8	22.8	41.6	22.2	41.0
TR–FR	39.4	18.4	40.4	17.4	38.9	29.4	43.9	26.3	43.6
FI–HR	30.4	20.2	32.7	17.4	32.5	27.4	39.7	24.3	38.4
FI–RU	38.2	17.6	37.3	18.0	38.7	27.6	42.4	27.6	41.3
FI–IT	39.9	18.4	40.5	17.6	40.0	25.2	45.3	24.0	44.9
FI–FR	42.8	17.5	43.0	18.2	41.9	26.4	47.6	26.1	45.8
HR–RU	40.6	26.4	41.4	29.5	43.9	36.1	46.4	33.3	46.4
HR–IT	40.4	24.6	44.0	22.8	42.8	32.3	49.7	30.2	49.2
HR–FR	43.6	24.1	46.5	23.4	44.1	35.0	51.8	29.3	50.7
RU–IT	46.6	22.0	46.9	19.4	45.0	32.0	50.2	28.4	51.1
RU–FR	48.7	22.1	49.1	20.5	47.6	37.0	53.5	28.9	51.2
IT–FR	64.1	33.4	65.5	32.9	64.3	42.2	66.0	39.0	66.4
Average	42.7	22.3	44.3	21.2	43.8	30.8	49.1	28.6	47.9

Table 7: Individual P@1 scores (×100%) for all 28 language pairs in the GT-BLI dataset of Glavaš et al. (2019), with model variants *with contrastive fine-tuning* (§2). $|\mathcal{D}| = 5k$.

		LaBSE +noCL						
Pair ↓ / $\lambda = \rightarrow$	VecMap(0.0)	0.1	0.2	0.3	0.4	0.5	0.7	1.0
EN–DE	55.6	57.3	56.3	54.1	50.9	47.8	40.6	25.4
EN–TR	40.4	42.0	41.9	42.1	41.6	40.2	35.9	23.6
EN–FI	45.6	46.1	46.1	45.1	44.4	42.8	34.4	19.3
EN–HR	37.5	39.8	43.0	45.3	46.5	46.5	41.3	24.7
EN–RU	45.6	46.5	48.0	49.9	50.5	50.2	46.1	24.7
EN–IT	60.2	61.6	63.0	62.3	61.3	60.1	49.9	26.4
EN–FR	64.1	66.0	67.1	67.5	65.9	64.8	55.3	34.2
DE–TR	32.5	33.6	33.4	33.0	32.3	30.2	24.4	17.7
DE–FI	39.7	39.7	39.3	37.4	35.9	34.3	25.8	16.0
DE–HR	33.3	35.8	37.0	38.5	38.8	36.1	29.2	19.2
DE–RU	40.0	40.9	41.2	41.5	41.1	38.4	29.6	14.3
DE–IT	49.5	51.3	51.3	51.4	49.4	46.3	36.4	19.4
DE–FR	50.0	52.2	53.2	53.2	51.8	49.1	37.8	22.5
TR–FI	31.3	32.3	31.6	31.7	31.2	29.8	24.8	15.5
TR–HR	25.4	28.3	31.2	33.1	34.1	33.6	28.9	18.7
TR–RU	32.9	34.9	35.5	36.9	36.5	34.0	29.0	14.1
TR–IT	37.1	38.7	39.6	38.9	38.8	37.7	31.5	17.0
TR–FR	39.4	41.4	42.1	43.1	43.4	41.9	34.6	20.9
FI–HR	30.4	32.3	34.7	36.4	36.9	36.3	28.6	17.4
FI–RU	38.2	39.5	40.0	41.0	40.5	37.6	28.9	15.1
FI–IT	39.9	42.9	42.9	43.4	44.2	41.5	34.0	18.1
FI–FR	42.8	44.7	45.9	45.9	46.1	43.6	34.6	18.6
HR–RU	40.6	41.8	43.9	45.7	45.8	45.0	39.0	24.5
HR–IT	40.4	43.5	46.0	47.9	48.6	47.6	41.8	24.5
HR–FR	43.6	46.8	48.6	50.0	50.1	47.9	42.0	25.7
RU–IT	46.6	48.1	48.1	47.6	46.8	44.5	38.7	21.8
RU–FR	48.7	50.2	51.0	50.9	50.0	49.0	43.6	26.6
IT–FR	64.1	64.9	65.8	65.6	64.9	63.0	54.0	33.9
Average	42.7	44.4	45.3	45.7	45.3	43.6	36.5	21.4

Table 8: Individual P@1 scores (×100%) for all 28 language pairs in the GT-BLI dataset of Glavaš et al. (2019), across different values for λ. The model variant is LaBSE +noCL (see §3); similar patterns are observed with another multilingual SE in our evaluation (xMPNET). $|\mathcal{D}| = 5k$.

		LaBSE +CL						
Pair ↓ / $\lambda = \rightarrow$	VecMap(0.0)	0.1	0.2	0.3	0.4	0.5	0.7	1.0
EN–DE	55.6	59.4	61.2	61.1	60.1	56.8	46.5	31.6
EN–TR	40.4	43.9	46.8	50.1	51.0	49.9	45.4	33.1
EN–FI	45.6	48.6	51.0	53.3	54.1	53.9	46.1	30.8
EN–HR	37.5	44.1	49.7	53.9	56.7	57.1	49.7	36.9
EN–RU	45.6	50.1	52.5	55.8	58.5	59.0	56.2	46.4
EN–IT	60.2	62.5	65.3	66.9	67.2	66.2	55.7	33.3
EN–FR	64.1	66.3	69.5	71.2	71.1	69.5	59.5	42.1
DE–TR	32.5	35.4	36.5	37.6	36.7	35.6	31.8	24.4
DE–FI	39.7	41.6	42.5	43.3	42.2	39.5	32.7	25.0
DE–HR	33.3	37.9	40.8	42.9	43.1	41.6	35.9	27.9
DE–RU	40.0	43.1	44.0	45.6	45.9	44.7	37.3	27.5
DE–IT	49.5	51.3	52.0	52.5	50.9	47.7	38.9	24.6
DE–FR	50.0	52.4	53.7	54.5	53.6	50.3	42.7	31.3
TR–FI	31.3	33.3	34.3	34.9	35.0	33.7	30.2	22.2
TR–HR	25.4	30.4	34.2	36.8	38.5	38.2	35.8	27.4
TR–RU	32.9	35.2	36.6	37.4	37.5	36.1	32.5	24.7
TR–IT	37.1	39.0	41.2	41.6	41.5	39.9	34.4	22.8
TR–FR	39.4	41.8	43.0	43.9	43.8	43.3	38.6	29.4
FI–HR	30.4	34.0	37.5	39.7	41.2	40.2	36.4	27.4
FI–RU	38.2	40.2	41.1	42.4	42.6	40.2	36.2	27.6
FI–IT	39.9	43.3	44.3	45.3	45.9	44.7	38.2	25.2
FI–FR	42.8	44.5	46.3	47.6	47.8	46.0	40.0	26.4
HR–RU	40.6	42.0	44.9	46.4	47.5	48.3	44.7	36.1
HR–IT	40.4	43.3	47.7	49.7	50.6	50.4	46.1	32.3
HR–FR	43.6	47.2	49.0	51.8	52.0	51.0	46.4	35.0
RU–IT	46.6	48.8	49.6	50.2	50.4	48.7	44.9	32.0
RU–FR	48.7	51.0	51.9	53.5	53.1	52.1	47.1	37.0
IT–FR	64.1	64.9	65.9	66.0	66.0	64.6	56.9	42.2
Average	42.7	45.6	47.6	49.1	49.4	48.2	42.4	30.8

Table 9: Individual P@1 scores (×100%) for all 28 language pairs in the GT-BLI dataset of Glavaš et al. (2019), across different values for λ. The model variant is LaBSE +CL (see §3); similar patterns are observed with another multilingual SE in our evaluation (xMPNET). $|\mathcal{D}| = 5k$.

		mBERT		XLM-R		LaBSE		xMPNET	
Pair ↓ / Config →	VecMap	+noCL (1.0)	+noCL (0.5)	+noCL (1.0)	+noCL (0.5)	+noCL (1.0)	+noCL (0.5)	+noCL (1.0)	+noCL (0.5)
EN–FI	42.2	1.1	30.9	1.4	21.4	46.8	51.6	47.7	53.4
EN–RU	39.7	5.2	30.1	2.5	21.3	53.5	52.3	57.5	55.2
EN–FR	64.8	11.3	52.8	2.2	29.6	68.5	74.2	64.7	73.9
FI–RU	33.0	4.3	25.3	3.2	20.7	38.4	41.6	42.7	45.6
FI–FR	46.7	3.6	35.3	0.3	22.8	43.1	52.6	42.9	53.2
RU–FR	48.2	8.6	37.9	0.8	25.3	52.2	57	52.3	58.4
Average	45.8	5.7	35.4	1.7	23.5	50.4	54.9	51.3	56.6

Table 10: Individual Spearman's ρ correlation scores ($\times 100$) on the XLSIM task (Multi-SimLex) for a subset of language pairs in our evaluation, with multilingual LMs and SEs used 'off-the-shelf' *without contrastive fine-tuning* (§2). See §3 for the description of different model configurations/variants. $|\mathcal{D}| = 5k$, with XLSIM test pairs removed from the dictionary. The number in the parentheses denotes the value for λ (see §3).

		mBERT		XLM-R		LaBSE		xMPNET	
Pair ↓ / Config →	VecMap	+noCL (1.0)	+noCL (0.5)	+noCL (1.0)	+noCL (0.5)	+noCL (1.0)	+noCL (0.5)	+noCL (1.0)	+noCL (0.5)
EN–FI	42.2	32.4	43.5	42.3	48	45.6	50.6	48.1	51.5
EN–RU	39.7	34.8	42.1	45.8	47.3	47.1	49.2	50.5	50.4
EN–FR	64.8	56.5	67.4	57.9	69.2	64	72	64.1	71.9
FI–RU	33.0	28.4	35.9	38.3	40.8	38.3	42.3	40.1	43.1
FI–FR	46.7	34.7	47.3	41.7	50.6	45.9	53.5	46.6	54.2
RU–FR	48.2	43.6	52.1	50.4	55.1	51.8	56.8	48.5	55.6
Average	45.8	38.4	48.1	46.1	51.8	48.8	54.1	49.6	54.5

Table 11: Individual Spearman's ρ correlation scores ($\times 100$) on the XLSIM task (Multi-SimLex) for a subset of language pairs in our evaluation, with model variants *with contrastive fine-tuning* (§2). $|\mathcal{D}| = 5k$.

	xMPNET +noCL								
Pair ↓ / λ = →	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	1.0
EN–FI	42.2	45.2	48.2	50.7	52.6	53.4	53	52.2	47.7
EN–RU	39.7	43.1	46.6	49.9	52.9	55.2	56.7	57.5	57.5
EN–FR	64.8	67.6	70.1	72.2	73.5	73.9	73.2	71.7	64.7
FI–RU	33	36.1	39.3	42.5	44.7	45.6	45.5	45.4	42.7
FI–FR	46.7	49.2	51.5	53.1	53.6	53.2	51.5	49.6	42.9
RU–FR	48.2	51.1	53.8	56.2	57.7	58.4	58.1	57.2	52.3
Average	45.8	48.7	51.6	54.1	55.8	56.6	56.3	55.6	51.3

Table 12: Individual Spearman's ρ correlation scores ($\times 100$) on the XLSIM task (Multi-SimLex) for a subset of language pairs in our evaluation, across different values for λ. The model variant is xMPNET +noCL (see §3); similar patterns are observed with another multilingual SE in our evaluation (LaBSE). $|\mathcal{D}| = 5k$.

	xMPNET +CL								
Pair ↓ / λ = →	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	1.0
EN–FI	42.2	44.2	46.4	48.6	50.4	51.5	51.8	51.3	48.1
EN–RU	39.7	41.6	43.9	46.3	48.6	50.4	51.6	52	50.5
EN–FR	64.8	66.9	69	70.7	71.8	71.9	71	69.5	64.1
FI–RU	33.0	35.1	37.5	39.9	41.9	43.1	43.5	43	40.1
FI–FR	46.7	48.9	51.2	53.1	54.2	54.2	53.3	51.7	46.6
RU–FR	48.2	50.1	52.1	53.8	55.2	55.6	55.1	53.8	48.5
Average	45.8	47.8	50	52.1	53.7	54.5	54.4	53.6	49.6

Table 13: Individual Spearman's ρ correlation scores ($\times 100$) on the XLSIM task (Multi-SimLex) for a subset of language pairs in our evaluation, across different values for λ. The model variant is xMPNET +CL (see §3); similar patterns are observed with another multilingual SE in our evaluation (LaBSE). $|\mathcal{D}| = 5k$.

Name	**URL**
mBERT	`huggingface.co/bert-base-multilingual-uncased`
XLM-R	`huggingface.co/xlm-roberta-base`
LaBSE	`huggingface.co/sentence-transformers/LaBSE`
xMPNET	`huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2`

Table 14: URLs of the multilingual Transformer models used in this work.

Pento-DiaRef: A Diagnostic Dataset for Learning the Incremental Algorithm for Referring Expression Generation from Examples

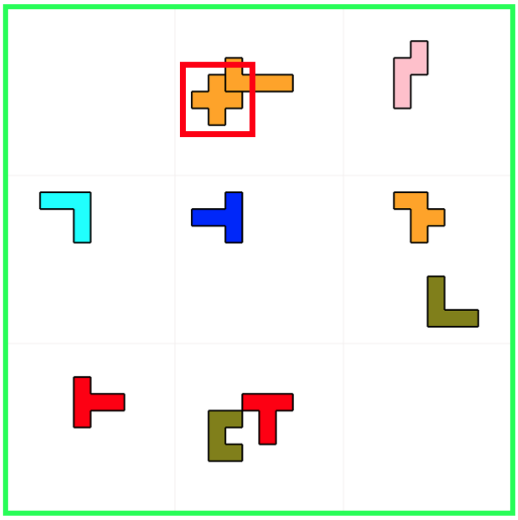
Philipp Sadler[1] and **David Schlangen**[1,2]
[1]CoLabPotsdam / Computational Linguistics
Department of Linguistics, University of Potsdam, Germany
[2]German Research Center for Artificial Intelligence (DFKI), Berlin, Germany
`firstname.lastname@uni-potsdam.de`

Abstract

NLP tasks are typically defined extensionally through datasets containing example instantiations (e.g., pairs of image i and text t), but motivated intensionally through capabilities invoked in verbal descriptions of the task (e.g., “t is a description of i, for which the content of i needs to be recognised and understood”). We present Pento-DiaRef, a diagnostic dataset in a visual domain of puzzle pieces where referring expressions are generated by a well-known symbolic algorithm (the “Incremental Algorithm”), which itself is motivated by appeal to a hypothesised capability (eliminating distractors through application of Gricean maxims). Our question then is whether the extensional description (the dataset) is sufficient for a neural model to pick up the underlying regularity and exhibit this capability given the simple task definition of producing expressions from visual inputs. We find that a model supported by a vision detection step and a targeted data generation scheme achieves an almost perfect BLEU@1 score and sentence accuracy, whereas simpler baselines do not.

1 Introduction

Being able to effectively and efficiently *refer* to objects is a central component of human language competence (van Deemter, 2016). The computational task of referring expression generation (REG) goes beyond the production of image descriptions (as in image captioning), in that it is a *uniquely identifying* description that needs to be produced, given a specific situation. In the formulation of Krahmer and van Deemter (2012), the REG task involves reasoning over all relevant objects in a scene, in order to determine what would make a description uniquely identifying. Additionally, maxims of efficiency (Grice, 1967) predict that it is a *minimal* natural language expression that should be preferred. The *Incremental Algorithm* (IA) (Dale and Reiter, 1995) is a well-known classic symbolic algorithm that tries to realise these desiderata. For example given a reference target and various distractors as in Figure 1 (an example of the domain chosen in this paper (Pentomino, Golomb (1996); Zarrieß et al. (2016); Kennington and Schlangen (2017))), then the Incremental Algorithm (IA) produces “Take the X”, achieving the desired uniquely identifying reference by mentioning only the shape and *not* also color and position.



(a) *“Take the orange X in the top”*

(b) *“Take the X”* (omit color and position)

Figure 1: An example board with a referring expression (b) as produced by the Incremental Algorithm (IA) (minimal wrt. a preference order); and an unnecessarily verbose reference (a) (that still is uniquely referring). The reference target is highlighted with bounding box; images regions separated by addition of lines.

Can such a reference strategy be learned by neural generation models from visual inputs alone? This is a question that is typically not systematically challenged in language generation from images (Kazemzadeh et al., 2014; Yu et al., 2016; Mao et al., 2016; Plummer et al., 2015; Luo and Shakhnarovich, 2017), as in natural scenes (such as in the RefCOCO dataset (Yu et al., 2016)), it has been shown that descriptions can be produced

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2106–2122
May 2-6, 2023 ©2023 Association for Computational Linguistics

Diagnostic Dataset	Task	Input	Condition	Output	Generalizability Testing
Wu et al. (2021)	Nav.	Symb. State	Text (Command)	Text (Actions)	Words, Phrases, Action Length
Liu et al. (2019)	REC	Image	Text (RE)	BBox	Color-Shapes
Pento-DIARef (Ours)	REG	Image	BBox	Text (IA-RE)	Color-Shapes, Positions, IA-REs

Table 1: The most relevant datasets in comparison to Pento-DIARef. In contrast to Liu et al. (2019) we study the task of REG (which avoids models to exploit language inputs) and add generalization tests for the output expressions.

based on the recognition of only parts of the image (Agrawal et al., 2016); our dataset is designed to make this impossible. Schlangen (2021) observed that in typical settings in the field of natural language processing, the connection between an underlying natural language capability and a learned model is only an indirect one. It rests on how well the dataset from which the model was induced does indeed exemplify the assumed underlying task—of which typically only a verbal description is given—and in turn on the extent to which the task represents the capability.

In this work we study how a intensionally defined task (in the distinction of Schlangen (2021)) for which a verbal and theoretically motivated description is given (through a symbolic algorithm) can be learned from its extensional exemplification. Our contention is that the use of synthetic data (Johnson et al., 2017; Liu et al., 2019; Lake and Baroni, 2018; Ruis et al., 2020; Wu et al., 2021) offers the opportunity to strengthen the link, insofar as guarantees can be given on the exemplification relation. More specifically, we choose the Incremental Algorithm (Dale and Reiter, 1995) for the data generation process, which itself comes with a motivation through recourse to underlying fundamental conversational capabilities (appeal to Gricean maxims, Grice (1967)). Our contributions are as follows:[1]

- We create a novel synthetic dataset, Pento-DIARef, of examples that pairs visual scenes with generated referring expressions;
- examine two variants of the dataset, representing two different ways to exemplify the underlying task;
- and evaluate an LSTM-based baseline (Mao et al., 2016), a transformer (Vaswani et al., 2017) and a modified version with region embeddings (Tan and Bansal, 2019) on them.

2 Related Work

Compositional Reasoning. Lake and Baroni (2018) introduced a systematic benchmark to test the generalization capabilities of recurrent neural networks through the use of compositional splits and found that these models fail “spectacularly”. Ruis et al. (2020) extended the task of mapping text commands to actions (Navigation) by conditioning the learner additionally on a symbolic world state. Later (Wu et al., 2021) provided a curated dataset along with new dimensions for generalizability testing. Our work follows the idea of generalization testing through compositional datasets in language and vision settings where training examples are composed in such a way that the models are exposed towards all property values of objects, but not to all the possible combinations of them, so that they can be tested on unseen combinations. In contrast to their work we use images instead of symbolic world states as the input.

Diagnostic Datasets. For the generation of the synthetic data we took inspiration from Johnson et al. (2017) who created a “diagnostic dataset” for visual question answering to test for model limitations. They draw 3D objects on a 2D plane and systematically use templates to create questions about the objects to avoid biases that occur in “common” datasets. Later Liu et al. (2019) convert the questions to referring expressions to test systematically for referring expression comprehension (REC). They claim that the models’ performance on the dataset proves that they “work as intended”. In this work we study this aspect as well but on the mirroring task of REG which avoids models to exploit hints from the language inputs (Table 1).

Program Learners As a related idea Pi et al. (2022) suggest to train language models on text outputs of “executable programs” (which could be a symbolic algorithm). They focus on the pretraining paradigm and aim to induce reasoning capabilities into language models to enhance their usefulness for downstream tasks. Our work is more specifically focused on the question whether a neural model is able to learn the underlying capabilities that are exhibited by a symbolic algorithm in a vision and language domain. Same et al. (2022) showed that such rule-based algorithms are still a useful approach for REG in natural settings.

---

[1] The source code and datasets are made publicly available at `https://github.com/clp-research/pento-diaref`.

3 Pento-DIARef Task and Dataset

We present a **D**iagnostic dataset of IA **Ref**erences in a **Pento**mino domain (Pento-DIARef) that ties extensional and intensional definitions more closely together, insofar as the latter is the generative process creating the former (Schlangen, 2021). In this chapter we describe the task (§3.1) and how it is tied to the Incremental Algorithm (§3.2) via the generation process (§3.3) and present our compositional splits (§3.6) for generalization testing.

3.1 Task Description

Given as input an $x_i = (v_i, b_i)$, representing a Pentomino board v_i as in Figure 1 and a bounding box b_i (indicating the target piece), a model f has to produce a referring expression y_i (as it would be generated by IA) as shown in Figure 2. Formally, this can be described either as a classification task $\text{argmax}_{y_i} P(y_i|x_i)$ when y_i is considered a whole sentence or more generally as a conditional language modeling task $P(w_t|w_{<t}, x_i)$ with $y_i = \{w_0, ..., w_T\}$ where T is the length of the expression. We present models for both of these interpretations in Section 5.1.

3.2 The Incremental Algorithm (IA)

The Algorithm 1 , in the formulation of (Krahmer and van Deemter, 2012), is supposed to find the properties that uniquely identify an object among others given a preference over properties. To accomplish this the algorithm is given the property values $\mathcal{P}$ of distractors in M and of a referent r. Then the algorithm excludes distractors in several iterations until either M is empty or every property of r has been tested. During the exclusion process the algorithm computes the set of distractors that do *not* share a given property with the referent and stores the property in $\mathcal{D}$. These properties in $\mathcal{D}$ are the ones that distinguish the referent from the others and thus will be returned.

Algorithm 1 The IA on symbolic properties as based on the formulation by van Deemter (2016)

```
Require: A set of distractors $M$, a set of property
    values $\mathcal{P}$ of a referent $r$ and a linear preference
    order $\mathcal{O}$ over the property values $\mathcal{P}$
$\mathcal{D} \leftarrow \emptyset$
for $P$ in $\mathcal{O}(\mathcal{P})$ do
    $\mathcal{E} \leftarrow \{m \in M : \neg P(m)\}$
    if $\mathcal{E} \neq \emptyset$ then
        Add $P$ to $\mathcal{D}$
        Remove $\mathcal{E}$ from $M$
return $\mathcal{D}$
```

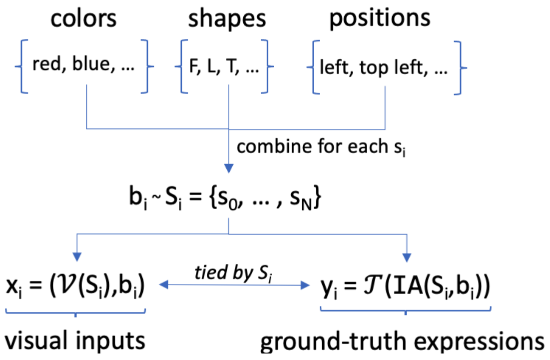


Figure 2: The general generation process for our synthetic datasets as defined by the task.

The algorithm has a meta-parameter $\mathcal{O}$, indicating the *preference order*, which determines the order in which the properties of the referent are tested against the distractors. In our domain, for example, when *color* is the most preferred property, the algorithm might return BLUE, if this property already excludes all distractors. When *shape* is the preferred property and all distractors do *not* share the shape T with the referent, T would be returned. Hence even when the referent and context are the same, different preference orders might lead to different expressions (Krahmer et al., 2012). We choose the preference order of *color, shape and position* for the algorithm; we leave experimenting with other orders to future work.

3.3 Data Generation

The inputs x_i for the task consist of two parts: the visual representation of the scene v_i and a bounding box around the target piece b_i. For the automatic generation of these inputs we make use of *symbolic board* representations $S_i = \{s_0, ..., s_N\}$ where N is the number of pieces on a board and s_i is a tuple of *color*, *shape* and *position* values e.g. (ORANGE,X,TOP). We define a mapping function $\mathcal{V}(S_i) \rightarrow v_i \in \mathbb{R}^{W \times H \times C}$ for rendering a board and sample uniformly from the symbols to select a target piece $b_i \sim S_i$ (for which we know the bounding box via $\mathcal{V}$). For simplicity, we use b_i to refer to the target bounding box in the visual domain or the target piece in the symbolic domain respectively.

As the ground-truth expressions we define $y_i = \{w_0, ..., w_T\}$ where T is the length of the expression and w_i is a word in the vocabulary. Again we make use of the symbolic piece representations (the same as above) to automatically generate the ground-truth by using the Incremental Algorithm. We apply the IA on the symbolic piece representations S_i and the target symbol b_i to select a set of property values $\mathcal{D}_i = \text{IA}(S_i, b_i)$ from the target b_i (Algorithm 1). These property values $\mathcal{D}_i$ are the shape, color or position values that are supposed to distinguish the target piece from other ones on the board. Finally, we define a mapping function $\mathcal{T}(\mathcal{D}_i) \rightarrow y_i$ to produce the ground-truth expression by filling the property values into pre-defined templates. The result of this process is a pairing of image and text, as you would find it for example in a captioning dataset (Johnson et al., 2016), albeit not collected from annotators but rather synthetically generated. In the following, we give more information on $\mathcal{V}$ and $\mathcal{T}$.

3.4 $\mathcal{V}$: Rendering the Pentomino boards

The symbolic piece representations in S_i are rendered as visual inputs v_i. We implement the rendering function $\mathcal{V}(S_i)$ that paints the symbolic pieces according to their shape and color values with black borders onto a board of 30×30 same-sized tiles. This underlying grid is projected onto 224×224 pixels. The exact tile coordinates of the pieces are determined by dividing the board into 9 distinct areas: one for each piece position value. To ensure that all pieces fit on the board, we allow maximal 2 pieces in a single area. We rotate and place the pieces one after the other into these areas by uniformly sampling tile coordinates that fall into the area that aligns to the piece position value. If two pieces collide during the placement, then we sample the coordinates again until they fit next to each other.

3.5 $\mathcal{T}$: Surface realization of IA outputs

The IA returns properties $\mathcal{D}_i$ of a target piece that distinguish it from other pieces. This list of properties is then transformed into a natural language expression. We define a mapping function $\mathcal{T}$ that inserts the property values into one of 7 different templates (Appendix C), for example "Take the *[color]* piece". We call these templates *expression types*. The mapping function selects the template based on the number of properties and the preference order: color, shape and then position. The word order in the templates is aligned with the preference order. We only use this order here to focus on the semantic correctness of the generation and leave mixing in additional variants like "Take the *[shape]* that is *[color]* in *[position]*" to future work. Altogether the property values and the templates lead to a vocabulary of 38 words—an extremely small vocabulary, which however as we are not targeting lexical complexity here is not a problem.

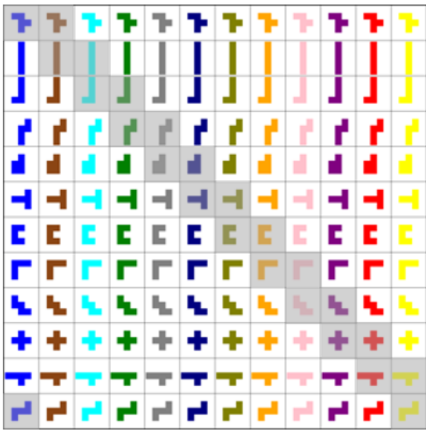


Figure 3: The shape and color combinations in grey are never seen during training, but only during evaluation.

3.6 Compositional Generalization

We make use of a synthetic dataset to guarantee the independence of properties and thus control, among other things, the compositionality of the learning task. There are 12 conventional names for the shapes which are roughly inspired by visual similarity to letters like *F*, *T*, *Y* etc. (Golomb, 1996). We sidestep the question of producing natural descriptions ("the one that looks like a boomerang") for the shapes and assume that these letter names can be produced. Furthermore, the pieces can appear in one of 12 different colors and the position can be approximated with 9 different spatial expressions (Appendix A.1, A.2). The permutation of colors, shapes and positions leads to $12 \cdot 12 \cdot 9 = 1296$ symbolic pieces to choose from for the composition of boards and the selection of targets. Now we create the training data from only $|S_{\text{train}}| = 840$ of the overall 1296 possible piece symbols and leave the remaining ones as a "holdout". These holdout pieces are specifically used to test the models' generalisation along three different dimensions, as in the following.

Piece appearances *(ho-color, 756 examples)*. The target piece shapes are combined with new colors with respect to the training set (Figure 3). For each of the 12 shapes we hold out 2 colors (val,test). Then we generate for each shape-color combination one board for each position and expression type.

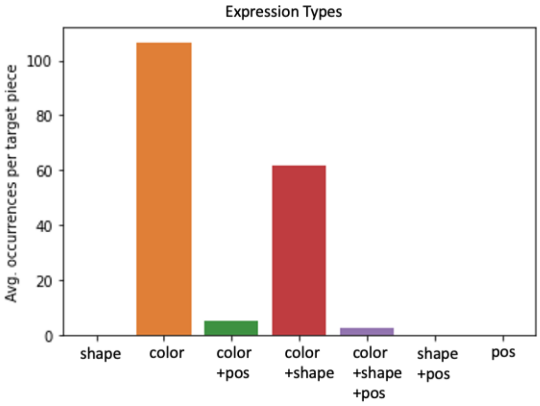


Figure 4: In the NAIVE dataset the synthesized examples highly prefer the generation of the two templates that only mention either the *(color)* or *(color,shape)* of a target piece. Some templates appear almost never [*(shape)*, *(shape,pos)*] or indeed never [*(pos)*].

Piece positions *(ho-pos, 840 examples)*. In these examples the target pieces are shown at new positions with respect to the training set. For each of the pieces we hold out 2 positions (val,test). Then we generate for the holdout combinations one board for each expression type.

Expression types *(ho-uts, 840 examples)*. We test that expression types are not attributed to specific pieces and show them in new contexts that leads to new expressions types wrt. the training set. For each of the pieces we hold out 2 expression types (val,test) and create corresponding boards.

4 What data is necessary to learn the IA?

The learned models have to generate an expression with exactly those property values (not more and not less) that the IA would produce. We hypothesise that learning the iterative set logic process and the preference order (either implicitly or explicitly) from text and visual inputs alone constitutes a challenging task for them, especially because given 12 shapes, 12 colors and 9 (discrete) positions for a piece (minus the combinations we excluded for holdout), then there are already around 20 billion possibilities to produce a board with 4 pieces on it.

Thus we make use of the fact that the generation process is fully under our control and directly ask what kind of data distribution is necessary to learn this task. We experiment with two different dataset variants: The first variant (§4.1) relies on an unconstrained sampling of symbolic pieces for each board while the second variant (§4.2) is designed to be more informative through a curated selection process.

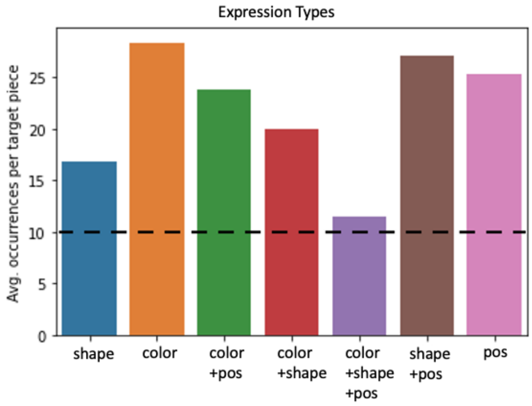


Figure 5: The DIDACT dataset is fairly balanced with respect to the expressions types. The figure shows the average count of expression types for each target piece after generating with ETOS. The dashed line indicate the distribution without the extra target selection.

4.1 NAIVE: Unconstrained Sampling

This process is meant to model the "naive" creation of a board by randomly sampling and placing pieces, as a person might do when setting up a board. We create these examples by randomly filling boards with pieces: First, we decide on the number of pieces that go on the board by sampling N from a uniform distribution over the integers 4–10. Then we sample uniformly with replacement $\{s_0, ..., s_N\}$ from the 840 symbolic pieces that are available for training. From the resulting symbolic board S_i we choose one piece, again uniform random, as the target piece b_{i_0}. Finally, we generate the input pairing (x_i, y_i) as described above.

We add one further constraint: We re-use the visual board v_i and pair it with 3 other target pieces $\{b_{i_1}, b_{i_2}, b_{i_3}\}$ chosen from S_i without replacement, so that a model *cannot* perform well by memorizing the (x_i, y_i) pairings alone, because then there are 4 identical visual boards with different targets that lead to (most likely) different expressions. This leads to 4 examples (x_j, y_j) per visual board with $x_j = (v_{i_0}, b_j)$. We repeat this procedure $42,000$ times which leads to $148,000$ training examples in total. The quantitative evaluation shown in Figure 4 reveals that here a model is most of the times confronted with expressions that only mention the *color* value or the *color and shape* of the target piece. The orange bar indicates that there are on average 100 examples (board and target) for each of the possible 840 target pieces where the color alone uniquely identifies the piece. So for around 84K samples in this dataset, a sentence like "Take the [blue, red, green,...] piece" would be correct.

4.2 DIDACT: Expression Oriented Sampling

The goal of the alternative sampling process is to ensure that examples of all *output* types are represented in the dataset, in a balanced way. We assume that this results in a more "didactic" dataset from which the underlying relation between input and desired output can more easily be induced. The idea is to directly choose the distractors of a target piece in such a way that the wanted expression type has to be produced. For example, when the target piece is (ORANGE, X, TOP) and the expression type is supposed to be *Take the [shape]*, then we construct a set of distractors where some share color and position, but none is of shape X. We call this approach *expression type oriented sampling* (ETOS) (details in Appendix B.1). This method allows us to confront the learner with all the possible expressions about the same amount of times. Thus each target piece is seen on 50 different boards resulting in $840 \cdot 50 = 42,000$ boards (Table 2).

Yet again we avoid that (x_i, y_i) pairs can be simply memorized and select as before 3 other pieces as the targets $\{b_{i_1}, b_{i_2}, b_{i_3}\}$ which leads to $148,000$ examples in total. The consequences of the extra target selection within this method are twofold: Firstly, the distribution is a bit shifted towards the NAIVE approach as shown in Figure 5 because we randomly select the target, and more importantly there might be now expressions produced that were actually intended for the holdout *(ho-uts)*. We remove such "unintended" examples from the training set so that there are $128,526$ examples for training (Table 3). Whereby the guarantees we can make for this "DIDACTic" dataset are that:

Target pieces appear with different distractors. Each target piece symbol for training appears on average in 153 contexts as a target.

On the same board occur different target pieces. We choose 3 additional pieces as targets apart from the one for which the board was initially intended.

Target pieces appear also on other boards. Each symbolic piece appears on average in $1,075$ contexts, which is more often than as a target.

5 Learning the Incremental Algorithm

Our goal in producing the collection of scenes was to ensure that a model f must indeed be based on features of the x_i that we care about (that is, which figure in the desired capability), namely the need to indeed compare the perceivable target piece and distractor properties. The IA (§3.2) achieves this by a hard-coded loop structure over symbols which (a) compares the objects (b) sticks to a preference order (c) preemptively stops when all distractors are excluded and (d) outputs the uniquely identifying properties (or all properties in ambiguous cases).

Dataset / Num. of	TPS	pET	Bords per pET	Boards Total
NAIVE	840	7	-	42,000
DIDACT	840	5	10	42,000
ho-uts val	840	1	1	840
ho-uts test	840	1	1	840
ho-color val	108	7	1	756
ho-color test	108	7	1	756
ho-pos val	120	7	1	840
ho-pos test	120	7	1	840

Table 2: The number of target piece symbols (TPS) and possible expression types (pET) per TPS for the NAIVE, DIDACT and holdout datasets. Although the number of the resulting NAIVE and DIDACT boards is with $42,000$ the same, the boards are generated with different techniques: either with random uniform sampling (NAIVE; the number of boards per pET and TPS is not controlled for) or expression type oriented sampling (DIDACT; 10 boards for each pET and TPS).

	NAIVE dataset	DIDACT dataset
Number of Boards	42,000	42,000
TPS per Board	4	4
Number of Samples	168,000	168,000
Validation	10,000	10,000
Testing	10,000	10,000
Training	148,000	148,000
Filtered	128,526	-

Table 3: The number of samples for each dataset and training split. For each board we chose 4 target piece symbols (TPS) (incl. the originally intended target piece in the DIDACT dataset) resulting into $168,000$ samples for both datasets. From this overall samples we choose $10,000$ for validation and testing. In addition, we exclude the training samples of the DIDACT for which an expression is to be produced that should reserved for the *ho-uts* testing. This is not done for the NAIVE dataset because here we train on possibly all expression types.

In the following, we present our neural models (§5.1) and the conducted experiments (§5.3) to test if neural language generators are indeed able to acquire such a "programatic" capability by the simple task definition of producing expressions from visual inputs. The generation models f will be trained on the basis of (x_i, y_i) pairs only. We train two common network architectures for this task of which one is an LSTM-based approach to REG proposed by Mao et al. (2016) for natural scenes and the other is a transformer (Vaswani et al., 2017). In addition, we propose a variant for processing the inputs along with a simple classifier-based baseline.

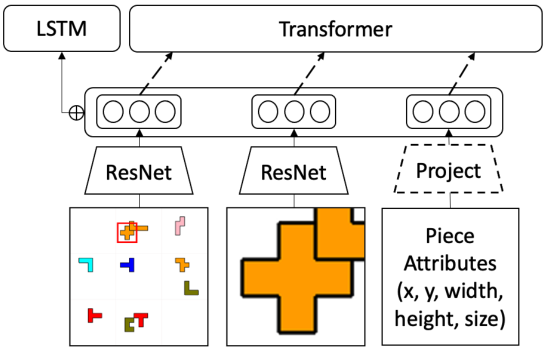


Figure 6: The encoding of the visual scene, target piece and its attributes as proposed by Mao et al. (2016). The dashed lines indicate the transformers' information flow.

5.1 Models

LSTM. Mao et al. (2016), who present a model of REG in natural scene images, embed the scenes and the referent within them with a pre-trained VGG (Liu and Deng, 2015). We follow their procedure but use the 512 dimensional embeddings after global average pooling of a ResNet-34 (He et al., 2016) and fine-tune all of its layers because our images look very different to the ones from the pre-training on ImageNet (Deng et al., 2009). We cut out the target piece using the bounding box information. Then the piece snippet is dilated with 5 context pixels and up-scaled to the size of the board image. We additionally randomly shift the snippet by 0-5% of the pixels in either direction horizontally or vertically (fill-color is white). The target piece and board image embeddings are then concatenated together with five location and size features of the target. The resulting 1029-dimensional feature vector is fed to an LSTM at each time step to condition the language production (using greedy decoding). We reduce the word embedding dims to 512 because our vocabulary is very small and apply an Adam optimizer (Kingma and Ba, 2015).

Transformer. For comparison with Mao et al. (2016) we resize, augment and encode the target piece and visual board with a ResNet-34 in the same way as described before. Then the image embeddings are fed into the transformer (Vaswani et al., 2017) individually (not concatenated) as "visual words" together with the target piece attributes embedding as shown in Figure 6 to compute an intermediate representation of the inputs altogether. This "memory" embedding is then fed into the decoder to generate the RE using masked self-attention as in other machine translation tasks. For the variable length expressions we use a padding symbol and ignore prediction at padded positions during loss computation. We reduced the original capacity of the model to avoid overfitting and applied a learning rate scheduling strategy as described by Vaswani et al. (2017), using an AdamW optimizer (Loshchilov and Hutter, 2019).

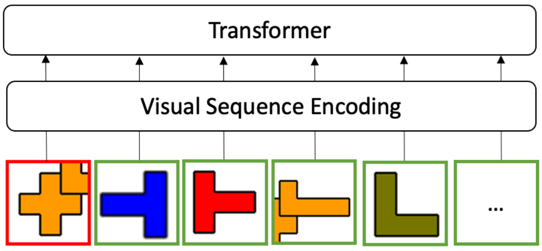


Figure 7: An exemplary visual input sequence for the Transformer+VSE model. The target is highlighted with a red and the distractors with a green border.

Transformer+VSE. We assume that the transformer should be particularly capable of generating IA-like expressions because self-attention might allow it to learn the required piece-wise comparison operation. The self-attention mechanism has already been proven powerful for other image-related tasks (Li et al., 2020; Zhang et al., 2021; Jaegle et al., 2021). Therefore we follow Tan and Bansal (2019) and implement a *visual sequence encoding* (VSE) mechanism. For this we cut out each piece on the board to produce a sequence of piece snippets as shown in Figure 7 and project the visual features onto the models' input dimensions $\hat{f}_j$ and add region embeddings $\hat{p}_j$ to them that contain the snippets size and location information:[2]

$$\hat{f}_j = \text{LayerNorm}(W_F f_j + b_F) \qquad (1)$$

$$\hat{p}_j = \text{LayerNorm}(W_P p_j + b_P) \qquad (2)$$

$$v_j = (\hat{f}_j + \hat{p}_j + \hat{t}_j)/3 \qquad (3)$$

To let a model distinguish between target and distractor "words" in the input sequence we add a type embedding $\hat{t}_j$, similar to word embeddings, and normalize. Furthermore, we have a variable amount of pieces on the board (between 4 and 10), but a transformer model assumes a fixed-size input sequence (per batch, during training). Thus we indicate "padding" pieces with a padding index in the sequence as implemented in PyTorch (Paszke et al., 2019) and use images with all zeros for them.

---

[2] b_F, b_P are bias terms of the linear projections

Model	Data	BLEU@1 (in %) ↑				Sentence-wise Acc. (in %) ↑			
		in-dist.	ho-color	ho-pos	ho-uts	in-dist.	ho-color	ho-pos	ho-uts
LSTM (Mao et al., 2016)	NAIVE	38	33	33	34	24	17	18	18
LSTM (Mao et al., 2016)	DIDACT	64	62	62	51	31	24	24	4
Transformer	NAIVE	27	23	23	23	21	14	14	15
Transformer	DIDACT	79	77	76	76	53	53	51	33
Transformer+VSE	NAIVE	59	53	57	54	29	22	22	25
Transformer+VSE	DIDACT	**97**	**97**	**97**	**97**	**91**	**91**	**91**	**92**
Classifier+VSE	NAIVE	32	25	28	28	27	15	19	22
Classifier+VSE	DIDACT	91	79	77	60	76	40	44	14

Table 4: The match rates for unigrams (BLEU@1) and whole sentences (SentA) on the test splits. The *in-dist.* samples are from the DIDACT dataset because these are more balanced with respect to the expression types.

Classifier+VSE. The representations of the VSE might already capture enough information to perform the task. Therefore we test this assumption by training a simple linear sentence classifier just on top of the concatenated embeddings. The classifier has to predict the correct sentence out of the 1,689 possible ones. This framing is similar to that often used in visual question answering (Hudson and Manning, 2019), where the possible answers are framed as classes in a classification task.

5.2 Metrics

We use the well known and commonly reported precision-based BLEU@1 metric for evaluation because this is simple metric for word matching when having only a single reference. In addition, we compute the *sentence-wise accuracy* (SentA) that indicates how often a prediction does exactly match the single reference so that the order of the words matters. As an example in Appendix E the model erroneously produces “Take the i top in the top left”. We ignore the starting words “Take the” for the evaluation when they occur in both the prediction and the ground-truth, because then they are uninformative about the real performance.

5.3 Experiments

We perform separate training runs on both a NAIVE (§4.1) and DIDACT (§4.2) dataset for a maximum of 100 epochs and perform 10 validation runs during an epoch. Over all validation runs we save the three best performing models with respect to the BLEU@1 score using greedy decoding. We stop the training when the model does not improve anymore after 20 validation runs. For evaluation we choose the model with more epochs if the scores are the same. The training objective is to minimize the cross-entropy between the predicted and the ground-truth expression given by the Incremental Algorithm (IA).

6 Results and Discussion

NAIVE versus DIDACT. The results in Table 4 show that even the worst performing model trained on the DIDACT dataset (LSTM 31% in-dist) *is still performing better* than the best performing model trained on the NAIVE dataset (Transformer+VSE 29% in-dist) over all SentA scores (except ho-uts). This indicates that a well controlled data generation procedure is essential to perform well on this task, or conversely, that none of the learning algorithms can guess at the underlying minimality constraint from the unconstrained data alone. The SentA scores for the NAIVE-based models indicate that these often perform only about by chance (picking 1 of 7 templates leads to a score of 14%) on the compositional splits (highest 25% and avg. 18%). These splits contain all expression types in equal amounts and we find that these NAIVE models tend to produce only a few expression types.

Input triplets versus VSE. The results show that for both datasets a significant increase in performance is achieved by using VSE which includes a vision detection step. For the Transformer+VSE model the BLEU@1 scores double from 27% to 59% on the in-distribution test data. The simple Classifier+VSE model performs similarly well as the other models *without* VSE. This is reasonable because with VSE the visual encoder must *not* operate on two different image resolutions anymore: one for the (up-scaled) target piece and one for the whole context image. The VSE detection step “frees” capacities that would be necessary to correctly identify the content of the context image.

Classifier+VSE versus others. Almost all models struggle to perform well on both the compositional (<54% SentA) and the in-dist. test data (<77% SentA). Thus the Classifier+VSE establishes a relative high baseline on most of the test sets (76%/40%/44%) but only performs about by chance (14%) at the *ho-uts* data (which contains unseen expression types). The Transformer+VSE model is the only one that exceeds the high Classifier+VSE baseline by achieving almost perfect scores (91% SentA) over all categories when trained on the DIDACT data.

Effect of individual input features. We perform an ablation study to measure the impact of particular input features on the SentA scores. We do so by replacing the individual parts of the visual sequence encoding of our best model with noise sampled from a standard gaussian. We see that the visual embeddings are essential to generate the correct referring expression as the sentence-wise accuracy drops to 1% (Table 5). A similar performance drop is seen for the type embeddings where the accuracy is only 1-2%. A different impact is measured, when the region embeddings are replaced with random noise; here the accuracy is still around 40-44%. This is reasonable, because in only 4 of the 7 expression templates, the position (and therefore the region embeddings) are relevant.

Effect of DIDACTic training. We have a closer look on the tendencies of the models to produce certain expression types on the test data. For this we applied a parser to the predicted expressions of the models and counted the expression type occurrences. This provides insights, if a model tends to "overfit" on specific expression types. For example as the surface structure of the *color* expression types is seen in majority of cases during training, a model might simply try to produce *Take the [color] piece* and insert the referent color. We do not check for the correctness of the produced expressions here. The measures show that the LSTM model trained on the NAIVE dataset has converged on a behavior that produces in the majority of cases the *color* or *color+shape* expression type (Figure 8). This is reasonable as this is the majority class in the random sampling data. Only the DIDACT dataset let's them pick up on other expression types more regularly. The Transformer+VSE produces on the DIDACT test dataset rather balanced amounts of expression types (as these are given in the test data).

7 Conclusion and Future Work

In this work we presented the diagnostic dataset Pento-DIARef to study the question whether neural models can learn the RE production strategy of the Incremental Algorithm (IA). A symbolic algorithm that is motivated by the appeal to the hypothesises capability of "elimination of distractors" (through the application of Gricean maxims).

	Sentence-wise Acc. (in %) ↑			
Transformer+VSE	in-d.	ho-color	ho-pos	ho-uts
w/o visual emb.	1	1	1	1
w/o type emb.	2	1	2	2
w/o region emb.	44	42	41	40
full model	91	91	91	92

Table 5: The ablation study performed on the test dataset shows that the visual and type embeddings contain the crucial information for the RE generation.

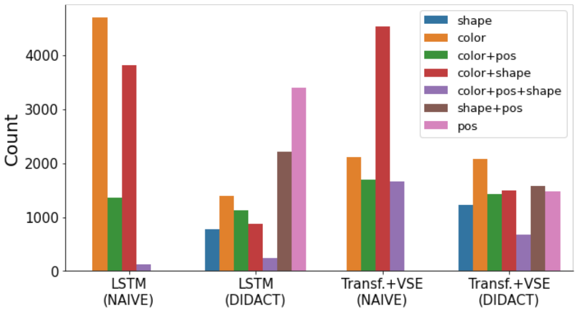


Figure 8: Produced expression types on the *in-dist.* split.

We found through the better control on scene complexity that an unconstrained sampling method (NAIVE) does not provide enough information for a neural model to pick up on the underlying regularity and to exhibit the desired capability, while an output oriented sampling process (DIDACT) does. This indicates that the generalizability in this task and domain is not given by the capabilities of the learner alone but is strongly determined by the learning examples. We evaluated a classic LSTM-based model and a modern transformer (that have to process two different image resolutions) and observed that these still struggle even on the more informative dataset (DIDACT). We proposed a modification of the input processing that comes with a detection step (VSE) and observed that this leads to a strong baseline and allows the transformer to converge. This indicates that object detection is an essential requirement to perform well on this task.

In future work we want to evaluate more models on our diagnostic dataset to find potential weaknesses. An interesting question is whether a PLM (Brown et al., 2020) might have picked up such Gricean constraints and would be able to recognise their desirability from being prompted with only a few examples. We also plan to explore to what extent our best model is applicable to more realistic settings following *Sim-to-Real* approaches (Peng et al., 2018).

Limitations

Limits on visual variability and naturalness. The Pentomino domain can only serve as an abstraction for referring expression generations in visual domains. The amount of objects is limited to 12 different shapes and the number of colors is reduced to 12 as well. The positions are chosen to be discrete and absolute while real-world references might include spatial relations which we leave for further work. Furthermore, the pieces show no texture or naturalness, but are drawn with a solid color fill and a simple black border. Various lightning conditions that might impact a vision detection system are avoided. We left the evaluation of the proposed models on more realistic dataset for further work.

Limits on variability of the referring expressions. We only explored expressions that are generate by the Incremental Algorithm with one fix preference order of color, shape and position although we are aware of the fact that preference order might vary between subjects (Krahmer et al., 2012). Moreover, we choose a fix property value order (color is mentioned before shape is mentioned before position) for the realisation of the template's surface structure and left the exploration for a higher variability to further work.

Limits possible claims about human capabilities. As this work is on synthetic dataset created by an algorithm, any claims about human capabilities, and about a model's ability to acquire those, are only made indirectly, via the quality of the original algorithm.

Acknowledgements

We want to thank the anonymous reviewers for their comments. This work was funded by the *Deutsche Forschungsgemeinschaft* (DFG, German Research Foundation) – 423217434 ("RECOLAGE") grant.

References

Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the Behavior of Visual Question Answering Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1955–1960, Austin, Texas. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Robert Dale and Ehud Reiter. 1995. Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19(2):233–263.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. ISSN: 1063-6919.

Solomon W. Golomb. 1996. *Polyominoes: Puzzles, Patterns, Problems, and Packings*. Princeton University Press.

Herbert Paul Grice. 1967. Logic and Conversation. In Paul Grice, editor, *Studies in the Way of Words*, pages 41–58. Harvard University Press.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Drew A. Hudson and Christopher D. Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6693–6702.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. 2021. Perceiver: General Perception with Iterative Attention. In *Proceedings of the 38th International Conference on Machine Learning*, pages 4651–4664. PMLR. ISSN: 2640-3498.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. 2017. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1988–1997. IEEE Computer Society.

Justin Johnson, Andrej Karpathy, and Li Fei-Fei. 2016. DenseCap: Fully Convolutional Localization Networks for Dense Captioning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*

(CVPR), pages 4565–4574, Las Vegas, NV, USA. IEEE.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. ReferItGame: Referring to Objects in Photographs of Natural Scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798, Doha, Qatar. Association for Computational Linguistics.

Casey Kennington and David Schlangen. 2017. A simple generative model of incremental reference resolution for situated dialogue. *Computer Speech & Language*, 41:43–67.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Emiel Krahmer, Ruud Koolen, and Mariët Theune. 2012. Is it that difficult to find a good preference order for the incremental algorithm? *Cognitive Science*, 36(5):837–841; discussion 842–845.

Emiel Krahmer and Kees van Deemter. 2012. Computational generation of referring expressions: A survey. *Comput. Linguistics*, 38(1):173–218.

Brenden Lake and Marco Baroni. 2018. Generalization without Systematicity: On the Compositional Skills of Sequence-to-Sequence Recurrent Networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2873–2882. PMLR. ISSN: 2640-3498.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 121–137. Springer.

Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L. Yuille. 2019. Clevr-ref+: Diagnosing visual reasoning with referring expressions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 4185–4194. Computer Vision Foundation / IEEE.

Shuying Liu and Weihong Deng. 2015. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734. ISSN: 2327-0985.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ruotian Luo and Gregory Shakhnarovich. 2017. Comprehension-guided referring expressions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3125–3134. IEEE Computer Society.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 11–20. IEEE Computer Society.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. 2018. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3810. ISSN: 2577-087X.

Xinyu Pi, Qian Liu, Bei Chen, Morteza Ziyadi, Zeqi Lin, Yan Gao, Qiang Fu, Jian-Guang Lou, and Weizhu Chen. 2022. Reasoning like program executors. *CoRR*, abs/2201.11473.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2641–2649. ISSN: 2380-7504.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M. Lake. 2020. A benchmark for systematic generalization in grounded language understanding. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Fahime Same, Guanyi Chen, and Kees Van Deemter. 2022. Non-neural models matter: a re-evaluation of neural referring expression generation systems. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5554–5567, Dublin, Ireland. Association for Computational Linguistics.

David Schlangen. 2021. Targeting the benchmark: On methodology in current natural language processing research. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 670–674. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Kees van Deemter. 2016. *Computational Models of Referring*, chapter 4.6. The MIT Press.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Zhengxuan Wu, Elisa Kreiss, Desmond C. Ong, and Christopher Potts. 2021. Reascan: Compositional reasoning in language grounding. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual.*

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. 2016. Modeling Context in Referring Expressions. In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 69–85, Cham. Springer International Publishing.

Sina Zarrieß, Julian Hough, Casey Kennington, Ramesh R. Manuvinakurike, David DeVault, Raquel Fernández, and David Schlangen. 2016. Pentoref: A corpus of spoken references in task-oriented dialogues. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA).

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. Vinvl: Revisiting visual representations in vision-language models. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 5579–5588. Computer Vision Foundation / IEEE.

A Experiment details

We trained each of our models on a single GeForce GTX 1080 Ti (11GB).

A.1 The vocabulary

The vocabulary includes the following 38 words:

- 12 shapes: F, I, L, N, P, T, U, V, W, X, Y, Z
- 12 colors: red, orange, yellow, green, blue, cyan, purple, brown, grey, pink, olive green, navy blue
- 6 position words: left, right, top, bottom, center (which are combined to e.g., right center or top left)
- 4 template words: Take, the, piece, at
- 4 special words: <s>, <e>, <pad>, <unk>

A.2 The piece colors (RGB-values)

Name	HEX	RGB
red	#ff0000	(255, 0, 0)
orange	#ffa500	(255, 165, 0)
yellow	#ffff00	(255, 255, 0)
green	#008000	(0, 128, 0)
blue	#0000ff	(0, 0, 255)
cyan	#00ffff	(0, 255, 255)
purple	#800080	(128, 0, 128)
brown	#8b4513	(139, 69, 19)
grey	#808080	(128, 128, 128)
pink	#ffc0cb	(255, 192, 203)
olive green	#808000	(128, 128, 0)
navy blue	#000080	(0, 0, 128)

Table 6: The colors for the Pentomino pieces. We also have 2 two-word colors: olive green and navy blue.

A.3 Uniform distribution of piece properties

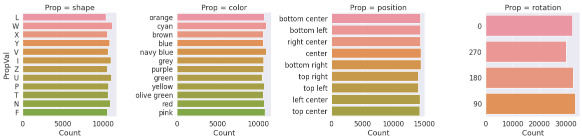


Figure 9: The occurrences of property values of target pieces in the DIDACT training data are almost uniform.

Our expression type oriented sampling strategy achieves an almost uniform distribution of piece color, shapes and positions (even rotations) as shown in Figure 9. We ignore the rotation property, but apply it to make the task harder. The model has to become invariant to the rotation.

B Data Generation

B.1 DIDACT dataset generation details

To construct this data, we iterate over all possible training symbols in S_{train} and set them as the target piece b_i directly (Table 2). Then we sample a symbolic board S_i (that includes the target) from the set of possible symbolic boards that lead to a wanted expression type u_i. For this we define the generator function $\mathcal{G}(u_i, b_i)$ that finds all possible symbolic boards that will be mapped by $\mathcal{T}$ so that $\{S_{u_i} | \mathcal{T}(\textsc{ia}(S_{u_i}, b_i)) \in \mathcal{Y}(u_i)\}$ where $\mathcal{Y}(u_i)$ is the collection of expressions that are represented by the template u_i, for example "Take the *[red, blue, green,...]* piece". In a sense $\mathcal{G}$ is the inverse of $\mathcal{T}$.

This method allows us to confront the learner with all the possible expressions about the same amount of times. We perform the example generation 10 times for each target piece and the according 5 training expression types (see §3.6 for holdouts). Finally, we generate the input pairing (x_i, y_i) as described in §3.1. Thus each target piece is seen on 50 different boards resulting in $840 \cdot 50 = 42,000$ boards. Yet again we avoid that (x_i, y_i) pairs can be learnt by heart and select as before 3 other pieces as the targets $\{b_{i_1}, b_{i_2}, b_{i_3}\}$ which leads to $148,000$ samples in total of which we filter the unintended ones (Table 3).

B.2 Holdout generation details

For the *ho-color* and *ho-pos* splits we additionally allow to choose distractors from the 840 symbolic pieces of the training split, because otherwise the distractor set of possible piece might become empty e.g. for the *ho-pos* split we have the target pieces only on a subset of possible positions, but need to place distractors in additional positions to produce all expression types.

C Expression Types

There are 3 expression types that are used when only a single property value of the target piece is returned by the Incremental Algorithm (IA):

- *Take the [color] piece*
- *Take the [shape]*
- *Take the piece at [position]*

Then there are 3 expression types that are selected when two properties are returned:

- *Take the [color] [shape]*
- *Take the [color] piece at [position]*
- *Take the [shape] at [position]*

And finally there is one expression type that lists all property values to identify a target piece:

- *Take the [color] [shape] at [position]*

In the following we exemplify the generated boards for each of the expression types.

C.1 Take the [color] piece

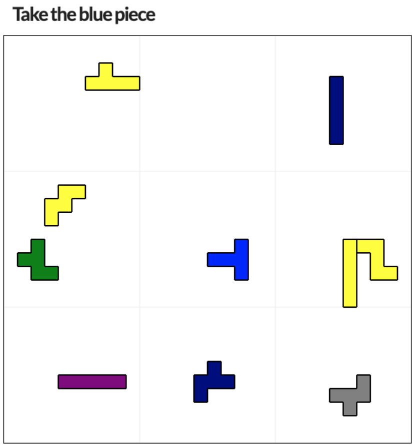


Figure 10: A sample board with the target piece (`T,blue,center`) for this expression type.

Mention the **color** excludes all. We add distractors with any shape or position, but a different color.

C.2 Take the [shape]

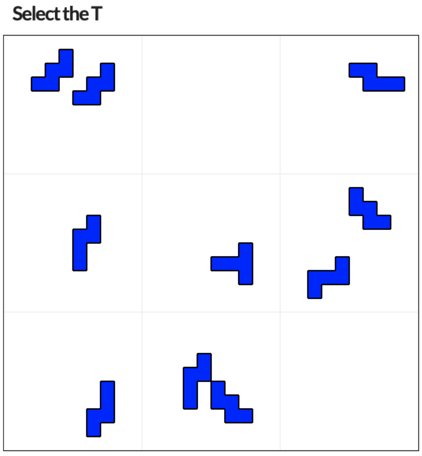


Figure 11: A sample board with the target piece (`T,blue,center`) for this expression type.

Mention the ~~color~~ does not exclude any. Mention the **shape** excludes all. We add distractors with the same color, but different shape and at any position.

C.3 Take the piece at [position]

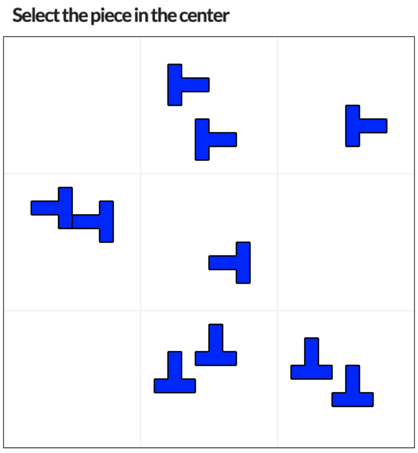


Figure 12: A sample board with the target piece (`T,blue,center`) for this expression type.

Mention the ~~color~~ does not exclude any. Mention the ~~shape~~ does not exclude any. Mention the **position** excludes all. We add distractors with the same color and shape, but at a different position.

C.3.1 Take the [color] [shape]

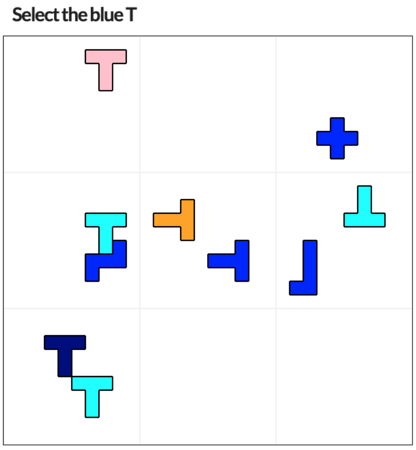


Figure 13: A sample board with the target piece (`T,blue,center`) for this expression type.

Mention the **color** excludes some, but not all. Mention the **shape** excludes the rest. We add some distractors with the same color (but different shape) and some distractors with the same shape (but different color) at any position.

C.3.2 Take the [color] piece at [position]

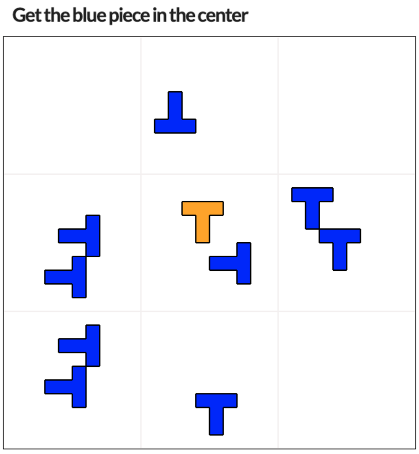


Figure 14: A sample board with the target piece (`T,blue,center`) for this expression type.

Mention the **color** excludes some, but not all. Mention the ~~shape~~ does not exclude any. Mention the **position** excludes the rest. We add some distractors with the same color (but different position) and some with the same position (but different color) and the same shape.

C.3.3 Take the [shape] at [position]

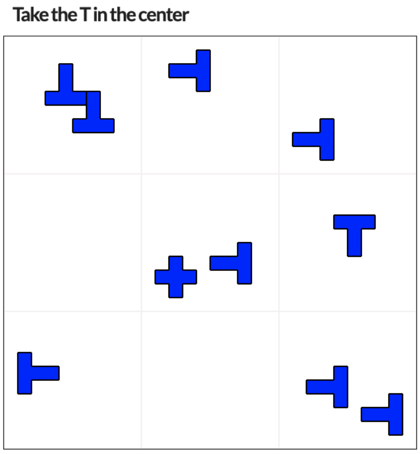


Figure 15: A sample board with the target piece (`T,blue,center`) for this expression type.

Mention the ~~color~~ does not exclude any. Mention the **shape** excludes some, but not all. Mention the **position** excludes the rest. We add distractors with the same color and some with the same shape (but different position) and some with the same position (but different shape).

C.3.4 Take the [color] [shape] at [position]

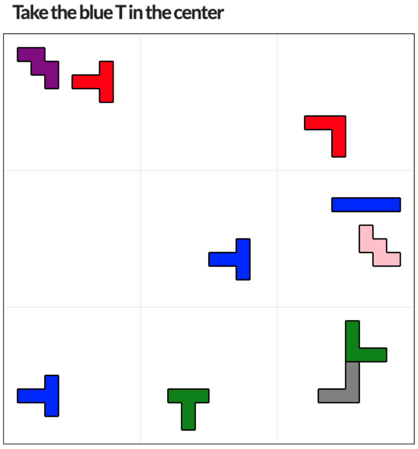


Figure 16: A sample board with the target piece `(T,blue,center)` for this expression type.

Mention the **color** excludes some, but not all. Mention the **shape** excludes some, but not all. Mention the **position** excludes the rest. We add one distractor that has the same color and shape (but a differen position) and one distractor that has the same color (but a different shape and position) and any other distractors. This requires at least 3 distractors.

D Model Details

D.1 LSTM

Parameters: 53, 201, 327 (127 MB)
GPU RAM: 3, 423 MiB (Batch 24; VE)

lstm_hidden_size	1024
word_embedding_dim	512
visual_embedding_dim	512
dropout	0.5
lr	0.0003
l2	0.0001
gradient_clip_val	10

Table 7: LSTM hyperparameters

D.2 Classifier

Classes: 1, 689
Parameters: 30, 234, 718 (120 MB)
GPU RAM: 11, 063 MiB (Batch 24; VSE)

d_model	512
visual_embedding_dim	512
lr	0.001
l2	0.01
layer_norm	0.00001
gradient_clip_val	10

Table 8: Linear model hyperparameters

D.3 Transformer

Parameters: 37, 402, 090 (149 MB)
GPU RAM: 10, 871 MiB (Batch 24; VSE)

d_model	512
word_embedding_dim	512
visual_embedding_dim	512
nhead	4
num_encoder_layers	3
num_decoder_layers	3
dim_feedforward	1024
dropout	0.2
lr_initial	$\text{d\_model}^{-0.5}$
l2	0.0001
layer_norm	0.00001
gradient_clip_val	10

Table 9: Transformer hyperparameters

E Error Analysis

Our best Transformer+VSE model predicts 1, 119 of 12, 436 evaluation expressions wrong meaning that the prediction does not match the reference perfectly. Here 425 errors (213 data, 77 ho-pos, 65 ho-color, 70 ho-uts) are expression predictions where the target piece is the one for which the board was initially designed for and 694 (all data) are cases where we picked an additional target randomly.

E.1 First-class errors

Error types	color	shape	pos	ungram.
data	4	5	180	24
ho-color	2	13	47	5
ho-pos	5	5	58	9
ho-uts	4	15	43	9

Table 10: The error types for the first-class errors

For the 425 first-class errors 213 of the errors are related to cases where the model mentions more properties of the target piece, although this would be unnecessary. In 47 cases the model produces an expressions that is not necessarily incorrect, but not grammatical.

E.2 Second-class errors

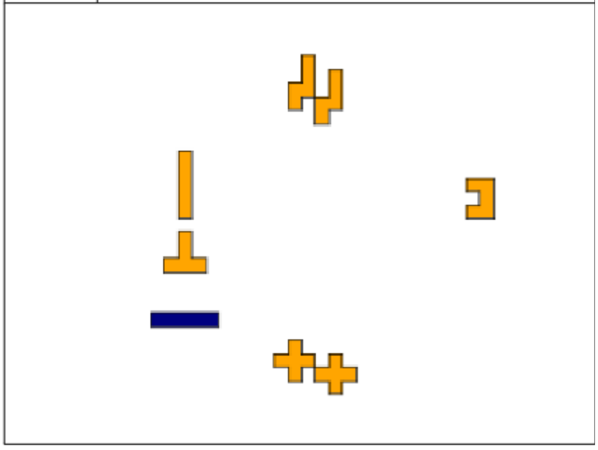
Error types	color	shape	pos	ungram.
data	30	56	524	90

Table 11: The error types for the second-class errors

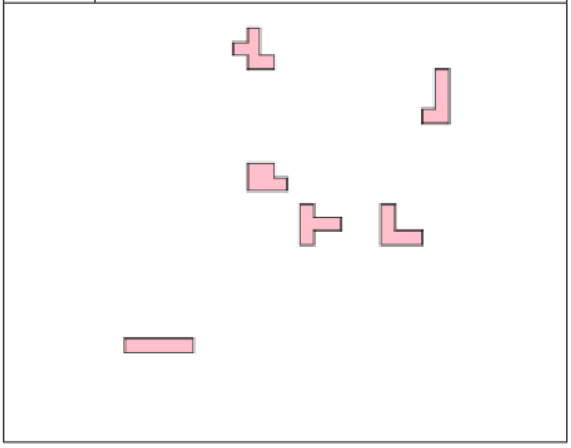
For the 694 second-class errors 179 of the errors are related to cases where the model mentions in addition the position, color or shape of the target piece, although this would be unnecessary. In 90 cases the model produces an expressions that is not necessarily incorrect, but not grammatical.

E.3 First-class error examples (intended target pieces)

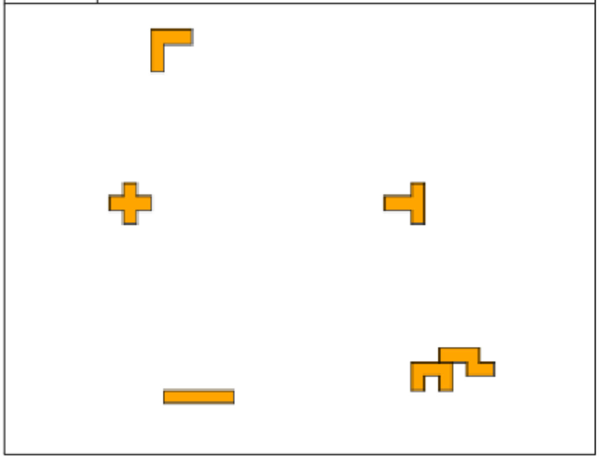
POSITION error	
Ref.	take the orange i
Pred.	take the orange i in the left center



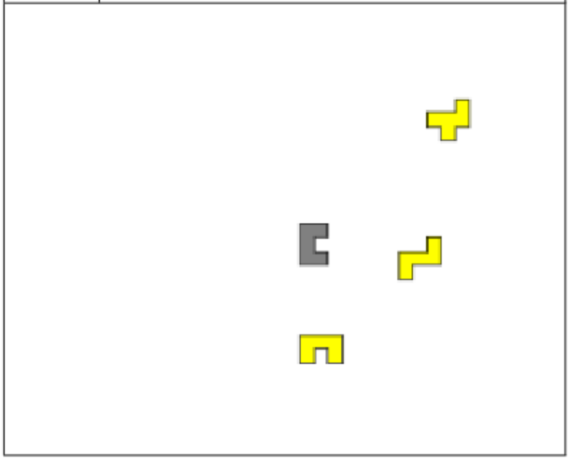
POSITION error	
Ref.	take the i
Pred.	take the i left center



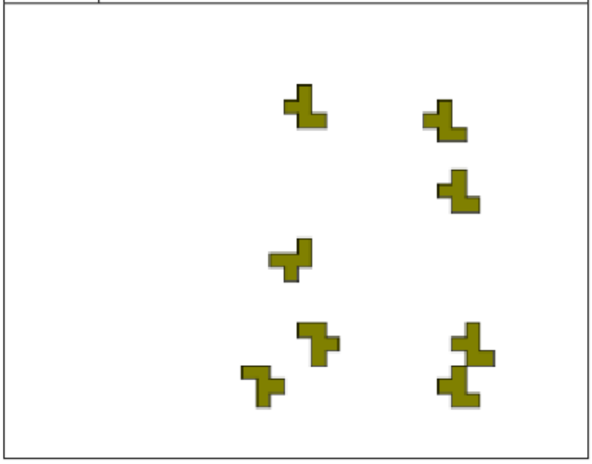
POSITION error	
Ref.	take the i
Pred.	take the i bottom in the bottom left



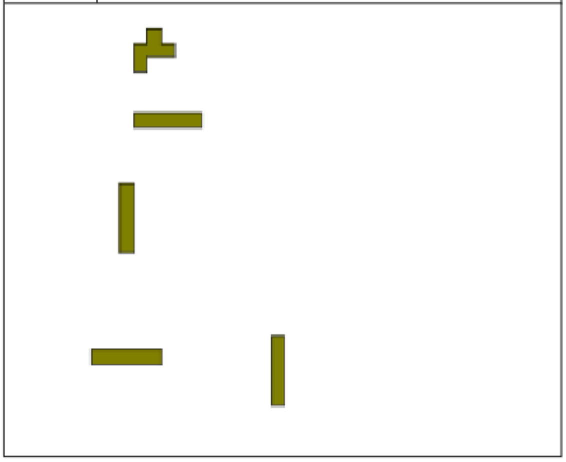
COLOR error	
Ref.	take the yellow u
Pred.	take the u



POSITION error	
Ref.	take the piece in the center
Pred.	take the piece in the bottom center

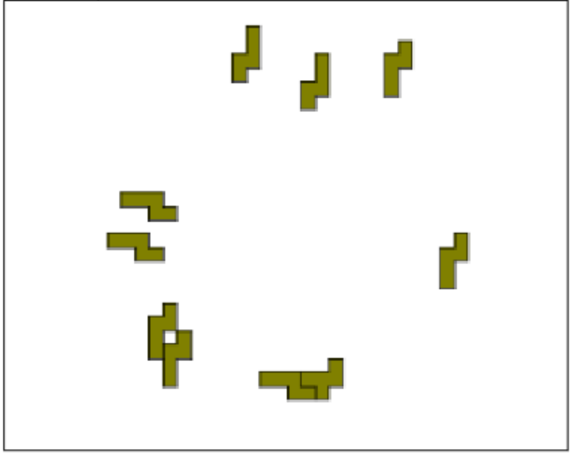


UNGRAM. error	
Ref.	take the i in the top left
Pred.	take the i top in the top left

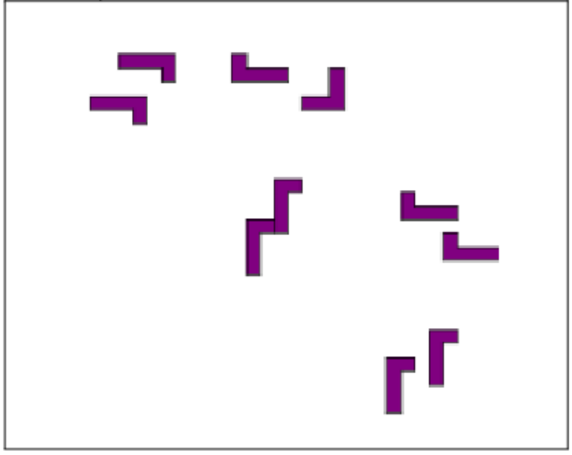


E.4 Second-class error examples (extra target pieces)

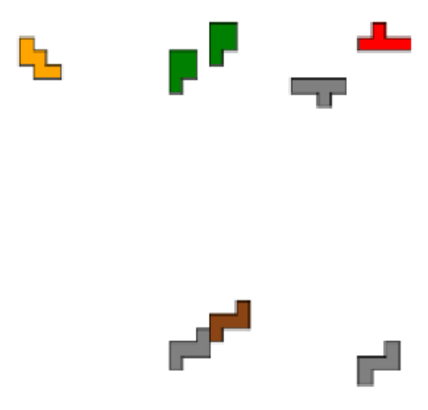
POSITION error	
Ref.	take the n in the right center
Pred.	take the n in the bottom right



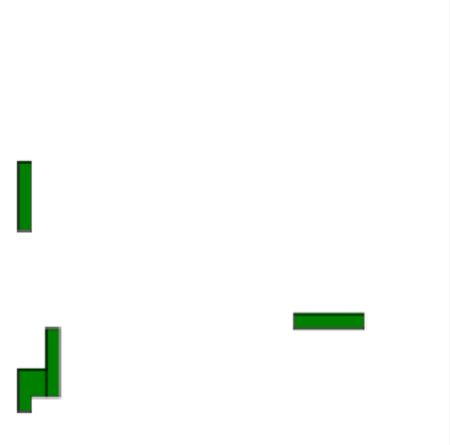
AMBIG. error	
Ref.	take the l in the bottom right
Pred.	take the piece in the bottom right



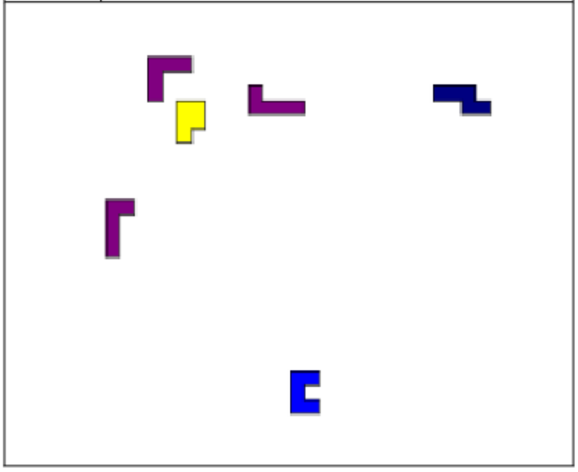
POSITION error	
Ref.	take the grey z in the bottom center
Pred.	take the grey z



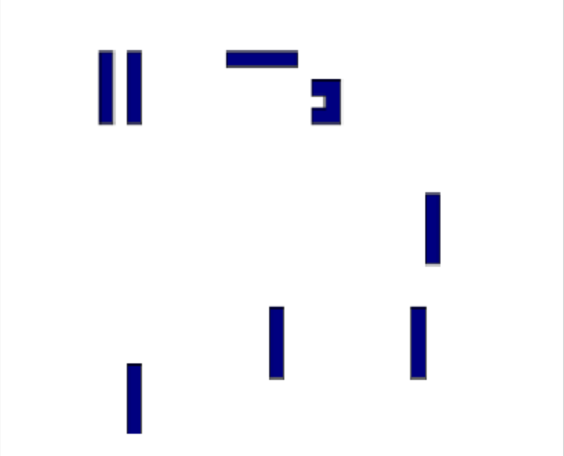
UNGRAM. error	
Ref.	take the i in the bottom right
Pred.	take the i bottom right



SHAPE error	
Ref.	take the purple l in the left center
Pred.	take the purple piece in the left center



COLOR error	
Ref.	take the i in the bottom left
Pred.	take the i bottom in the bottom left



Mitigating Exposure Bias in Grammatical Error Correction with Data Augmentation and Reweighting

Hannan Cao*   Wenmian Yang†   Hwee Tou Ng*
* Department of Computer Science, National University of Singapore
† School of Computer Science and Engineering, Nanyang Technological University
`caoh@u.nus.edu, wenmian.yang@ntu.edu.sg, nght@comp.nus.edu.sg`

Abstract

The most popular approach in grammatical error correction (GEC) is based on sequence-to-sequence (seq2seq) models. Similar to other autoregressive generation tasks, seq2seq GEC also faces the exposure bias problem, i.e., the context tokens are drawn from different distributions during training and testing, caused by the teacher forcing mechanism. In this paper, we propose a novel data manipulation approach to overcome this problem, which includes a data augmentation method during training to mimic the decoder input at inference time, and a data reweighting method to automatically balance the importance of each kind of augmented samples. Experimental results on benchmark GEC datasets show that our method achieves significant improvements compared to prior approaches.[1]

1 Introduction

Grammatical error correction (GEC) is the task of correcting errors in a source sentence and generating a well-written and grammatically correct target sentence. Good results have been achieved by state-of-the-art GEC systems (Rothe et al., 2021; Stahlberg and Kumar, 2021) based on the sequence-to-sequence (seq2seq) transformer (Vaswani et al., 2017) architecture.

Generally, the seq2seq models are optimized to predict the next token given all previous context tokens at each time step. During training, the ground truth tokens are used as the context, i.e., the so-called teacher forcing mechanism. However, during inference, the context is the previous tokens predicted by the model. As a result, the contexts at training and inference time are drawn from different distributions. This discrepancy is called *exposure bias* (Ranzato et al., 2016), which causes the model to make inferences under conditions that it has not met during training and causes errors to accumulate during inference.

Source	If that also word for you, pleas close the ticker.
Target	If that also works for you, please close the ticket.
Decoder input	
Actual	If that also work for you, pleas close the ticker.
Noise injection	**It** that also **works** for you, pleas **cloze** the ticker.
Random sampling	If that also **works** for you, **please** close the ticker.

Table 1: Example decoder inputs produced by different methods. Each method's deviations from the actual decoder input are shown in **bold**.

Currently, the exposure bias problem has attracted much attention in autoregressive text generation. For example, in neural machine translation (NMT), some work (Ranzato et al., 2016; Mihaylova and Martins, 2019; Zhang et al., 2019) addresses this problem by utilizing sentence-level metrics, randomly injecting noise into the training samples as the augmented decoder input, or randomly replacing tokens in the training samples with predicted tokens as the augmented decoder input. Although GEC is a text generation task like NMT, it emphasizes more on proposing correct edits. Therefore, feeding artificially augmented sentences into decoder input during training may produce a different distribution compared with the actual decoder input at inference time.[2] As a result, those generated errors may never happen, making the augmented sentences less effective in the GEC task. As illustrated in Table 1, the augmented decoder input using noise injection or random sampling differs from the actual decoder input.

---

[1] The source code of this paper is publicly available at https://github.com/nusnlp/gec_eb

[2] We empirically demonstrated the difference in Table 5 in Section 4.4

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2123–2135
May 2-6, 2023 ©2023 Association for Computational Linguistics

To address the exposure bias problem more effectively in GEC, we propose a data augmentation approach, which reduces the difference in distribution of the samples between training and inference in an intuitive way. Specifically, we first collect the beam search output generated by the GEC model as the augmented sentences, which mimic the actual decoder input at inference time. Then, during training, the ground-truth and augmented sentences are both fed as input to the decoder, thus reducing the difference in distribution of the samples between training and inference.

Moreover, previous work (Zhang et al., 2019; Bengio et al., 2015) shows that adjusting the noise density during training can better address the exposure bias problem. Since different beam search candidates contain different noise densities (i.e., the error density in GEC), and affect the model performance to varying extent, each candidate should be treated unequally. Therefore, we further propose a data-reweighting method, which automatically learns sampling probability for different augmented sentences dynamically. Specifically, our reweighting approach learns a data scorer through reinforcement learning, which learns the sampling probability for each augmented sentence.

The contributions of our paper are as follows:

- We empirically demonstrated that the exposure bias problem exists in the top-performing seq2seq GEC models, and addressing them helps to improve performance.

- We propose a novel data manipulation approach in GEC, including a data augmentation approach that mitigates the distribution gap between training and inference, and a data reweighting approach that automatically learns the sampling probability for each kind of training samples. No previous work uses data reweighting to tackle exposure bias.

- To the best of our knowledge, our data manipulation approach is the first work that addresses the exposure bias problem regardless of the evaluation metrics in seq2seq GEC.

- Experimental results show that our data augmentation approach significantly reduces the exposure bias problem and improves the performance of seq2seq GEC models.

2 Related Work

This section briefly introduces recent related research in the literature.

2.1 Exposure Bias

Exposure bias has been mostly studied in NMT. Bengio et al. (2015) replace the ground truth word by sampling randomly from the previous predicted words during training. Zhang et al. (2019) further inject noise into the training examples at both word level and sentence level, and select sentence level candidates based on the BLEU score (Papineni et al., 2002). Instead of injecting random noise to the ground truth sentences during training, our data augmentation method treats model-generated sentences as the augmented decoder input to mitigate the difference in distribution of samples between training and inference.

Ranzato et al. (2016) utilize Mixed Incremental Cross-Entropy Reinforce to optimize the model using metrics used at test time (e.g., BLEU) directly. Many similar works (Shao et al., 2018; Saunders et al., 2020) have also been proposed in NMT. For GEC, Sakaguchi et al. (2017) address the exposure bias problem by using reinforcement learning to directly optimize the GEC model towards the GLEU score (Napoles et al., 2015). Compared to Sakaguchi et al. (2017), we do not optimize the GEC model using reinforcement learning, and we address the exposure bias problem effectively where GLEU score is not used.

2.2 Data Reweighting

Many methods have been developed for reweighting or selecting a subset of data suitable for training a model (Franceschi et al., 2018; Chen et al., 2017; Wu et al., 2018; Shu et al., 2019; Wang et al., 2020; Lichtarge et al., 2020). Specifically, Lichtarge et al. (2020) propose an offline reweighting method for GEC, which reweights each training sentence in the previous stage of training based on the perplexity difference between the current and previous checkpoint. Wang et al. (2020) use a reweighting method to learn the sampling distribution for different languages in multilingual machine translation. In contrast, our method is an online reweighting method which dynamically adjusts the error density of the augmented sentences according to different model states.

3 Method

In this section, we first briefly introduce a baseline GEC system in Section 3.1. Then, we introduce our data manipulation method in Section 3.2.

3.1 A Baseline GEC system

Let x be the ungrammatical source sentence and y be the corrected grammatical target sentence. For a seq2seq GEC model parameterized by θ, the goal is to minimize the negative log-likelihood (NLL) for a set of M sentence pairs $\{\langle x^{(i)}, y^{(i)}\rangle\}_{i=1}^{M}$, as follows:

$$l(x, d, y; \theta) = -\sum_{t=1}^{L_y}\sum_{k=1}^{|V|} \mathbb{1}\{y_t = k\}\cdot \log P(y_t = k|d_{<t}, x; \theta) \tag{1}$$

$$L_{NLL}(\theta) = \frac{1}{M}\sum_{i=1}^{M} l(x^{(i)}, y^{(i)}, y^{(i)}; \theta) \tag{2}$$

where x is the source sentence, d is the decoder input of the seq2seq model, y is the target sentence, L_y is the length of the target sentence, $|V|$ is the vocabulary size of the target language, y_t is the t-th target token, $\mathbb{1}\{\cdot\}$ is the indicator function, and $P(\cdot|\cdot)$ is the conditional probability with model θ.

Given trained parameters $\hat{\theta}$, the hypothesis sentence $\hat{y}$ is generated using beam search to select the candidate with the highest probability, as follows:

$$\hat{y} = \arg\max_{y}\{P(y|x; \hat{\theta})\} \tag{3}$$

3.2 Data Manipulation

We propose a data manipulation method that mitigates the difference in contexts during training and inference to improve performance. In particular, our method consists of two parts: A data augmentation method that generates augmented sentences based on beam search to mimic the decoder input distribution during inference, and a data reweighting method that assigns sampling probability for augmented sentences. The overview of our approach is shown in Figure 1.

We will first introduce the data augmentation method in Section 3.2.1 and then introduce the data reweighting method in Section 3.2.2.

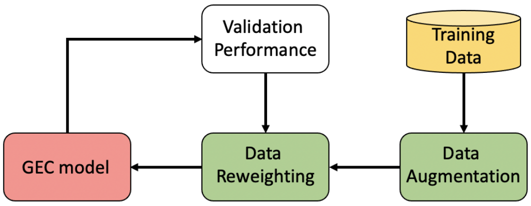


Figure 1: Overview of the proposed data manipulation method to mitigate exposure bias in GEC.

3.2.1 Data Augmentation

To reduce the distribution gap of decoder input during training and inference, we propose to add augmented sentences into the decoder input during training. These augmented sentences are generated by beam search using the model parameters from the checkpoint of the previous stage[3], which mimic the decoder input at inference time.

More specifically, we generate augmented sentences by feeding the source sentence x to the model parameterized by θ, and choose the top n output sentences $\{y_j\}_{j=1}^{n}$ with the highest probabilities generated by beam search.

As the decoder input must have the same length as the target during training, we first align the augmented sentences $\{y_j\}_{j=1}^{n}$ with the corresponding target sentence y according to linguistically-enhanced Damerau-Levenshtein alignment (Felice et al., 2016) by the ERRANT toolkit.[4] Then, we remove the excess tokens and add paddings for the missing tokens. The padded augmented sentences are denoted as $\{\bar{y}_j\}_{j=1}^{n}$. We collect $(x, \{\bar{y}_j\}_{j=1}^{n}, y)$ to form an augmented dataset D_{aug}.

Intuitively, for a well-trained GEC model, candidates with higher probability to be generated usually have lower error densities[5]. Therefore, different candidates affect the model performance unequally. To better capture the effect of different augmented sentences, we assign a separate weight for each augmented sentence, and the loss function is formulated as follows:

$$L_{EB}(\theta) = l(x, y, y; \theta) + \sum_{j=1}^{n} \alpha_{x,j} \cdot l(x, \bar{y}_j, y; \theta) \tag{4}$$

where $\alpha_{x,j}$ represents the weight for the jth augmented sentence for source sentence x. We set

---

[3] Following prior work (Stahlberg and Kumar, 2021), we train the GEC model in three stages, and load the checkpoint of the second stage

[4] https://github.com/chrisjbryant/errant

[5] We verified this in Table 5 in Section 4.4

$\sum_{j=1}^{n} \alpha_{x,j} = 1$, so that all the augmented sentences are treated equally as the original sentence.

3.2.2 Data Reweighting

The ideal way to train an optimal GEC model using Eq. 4 is to determine the optimal value for each $\alpha_{x,j}$. However, this can be computationally costly. To efficiently weight each augmented sentence, we use the averaged weight γ_j to approximate each individual weight $\alpha_{x,j}$, where $\gamma_j \sim E_{\alpha_j}$ and E_{α_j} represents the expectation of $\alpha_{x,j}$, calculated by

$$E_{\alpha_j} = \mathbb{E}_j[\alpha_{x,j}, x \in D_{train}]$$

Therefore, we reformulate Eq. 4 as:

$$L_{EB}(\theta) = l(x, y, y; \theta) + \sum_{j=1}^{n} \gamma_j \cdot l(x, \bar{y}_j, y; \theta) \tag{5}$$

Previous work (Zhang et al., 2019; Bengio et al., 2015) has demonstrated that varying noise density (error density in GEC) according to the training state of the model can effectively address the exposure bias problem. Therefore, we use the sampling probability β_j to approximate γ_j and propose to dynamically adjust β_j according to the model θ. Specifically, we design a learnable data scorer $P(j; \psi)$, parameterized by ψ to modify the training objective which we use to learn θ, so as to maximize the validation performance. Here $j \in \{1, ..., n\}$ indicates the index of the augmented sentence (index 1 indicates the sentence generated with the highest probability in beam search, and so on). More specifically, in the training process, $P(j; \psi)$ is used as the probability to sample an augmented sentence at index j such that the validation loss is minimized.

The final objective is written as follows:

$$\begin{aligned} \psi^* &= \operatorname*{argmin}_{\psi} L(\theta^*(\psi), D_{dev}) \text{ where} \\ \theta^*(\psi) &= \operatorname*{argmin}_{\theta} \sum_{(x,d,y)} L_{EB}(\theta, \psi) \end{aligned} \tag{6}$$

and $L_{EB}(\theta)$ in Eq. 5 is reformulated as:

$$\begin{aligned} L_{EB}(\theta, \psi) = {} & l(x, y, y; \theta) + \\ & \sum_{j=1}^{n} \mathbb{1}\{j = s_i\} \cdot l(x, \bar{y}_j, y; \theta) \end{aligned} \tag{7}$$

where $s_i \in \{1, ..., n\}$ represents the sampled index based on the learned probability $P(j; \psi)$ for each index j [6].

During training, we optimize θ and ψ iteratively. We first optimize the data scorer ψ with fixed θ. To update the data scorer, we use reinforcement learning with a reward function that approximates the effect of the training data on the model's development set performance.

Specifically, our *environment* is the model state θ and model input (x, d, y). Our *agent* is the data scorer network ψ. We formulate our *reward* as:

$$\begin{aligned} & R(j; \theta_t) \\ & = \cos(\nabla_\theta L_{dev}(D_{dev}; \theta_t) \cdot \nabla_\theta L_{train}(j; \theta_t)) \end{aligned} \tag{8}$$

where $L_{dev}(D_{dev}; \theta_t)$ denotes the validation loss calculated by Eq. 2, $L_{train}(j; \theta_t)$ represents the training loss for different kinds of augmented sentences, and $\cos(\cdot)$ denotes the cosine similarity of the two vectors.

Algorithm 1: Data reweighting method

```
Input: $D_{train}$, $D_{dev}$, $D_{aug}$
Output: Optimal parameter $\theta^*$
Initialize $\theta$, $\psi$
while not converge do
    Sample $f$ batches of training data
      $(X, D, Y) \sim (D_{train}, D_{aug})$
    Sample validation data
      $(X_{dev}, Y_{dev}) \sim D_{dev}$
    $g_{dev} \leftarrow \sum_{(x_{dev}, y_{dev}) \in (X_{dev}, Y_{dev})}$
      $\nabla_\theta l(x_{dev}, y_{dev}, y_{dev}; \theta) / |(X_{dev}, Y_{dev})|$
    ▷ Update $\psi$
    for $(x, d, y)$ in $(X, D, Y)$ do
        ▷ Estimate different $j$ effect
        for $j$ in $\{1, ..., n\}$ do
            $g_\psi \leftarrow \nabla_\theta L_{train}(j; \theta)$
            $R(k; \theta) \leftarrow \cos(g_{dev}, g_\psi)$
        end
        ▷ Optimize $\psi$
        $d_\psi \leftarrow \sum_{j \in \{1, ..., n\}} R(j; \theta) \cdot$
          $\nabla_\psi \log P(j; \psi)$
        $\psi \leftarrow$ GradientUpdate($\psi$, $d_\psi$)
    end
    ▷ Update $\theta$
    for $(x, d, y)$ in $(X, D, Y)$ do
        $g_\theta \leftarrow \nabla_\theta L_{EB}(\theta, \psi)$
        $\theta \leftarrow$ GradientUpdate($\theta$, $g_\theta$)
    end
end
```

---

[6] We use torch.multinomial function from the PyTorch package to sample one candidate at a time.

To capture the combined effect of augmented sentence and the original training sentence to the model's performance, we calculate $L_{train}(j;\theta_t)$ as follows:

$$L_{train}(j;\theta_t) = l(x, y, y; \theta) + l(x, \bar{y}_j, y; \theta_t) \quad (9)$$

Intuitively, this reward makes the data scorer upweight the training data that have a similar gradient direction as the development data (Wang et al., 2020).

According to the REINFORCE algorithm (Williams, 1992), the update rules for the data scorer become:

$$\psi_t \leftarrow \psi_{t-1} + R(j;\theta_t) \cdot \nabla_\psi \log P(j;\psi) \quad (10)$$

After obtaining ψ, we update θ by:

$$\theta_t \leftarrow \theta_{t-1} - \nabla_\theta \sum_{(x,d,y)} L_{EB}(\theta, \psi) \quad (11)$$

The training algorithm is shown in Algorithm 1. Note that we do not calculate the validation gradient in every training step, since it is too computationally expensive for the GEC task. Instead, we calculate the validation gradient for every f steps. Specifically, we first sample validation data and f batches of training data from the validation and the training set, respectively. Then, we obtain validation gradient g_{dev} based on the current model weights θ. For each batch of training data, we further calculate the reward for the data scorer using Eq. 8 and update the data scorer using the REINFORCE algorithm. After updating the data scorer, we apply the output from ψ to Eq. 7 and use this sampled loss to update the model θ with the same f batches of training data.

In this paper, we use a single-layer embedding network followed by a softmax layer as the data scorer networks ψ, and a transformer-based framework as our model θ. More details are given in Section 4.2.

4 Experiments

In this section, we report the effectiveness of our data manipulation approach.

4.1 Data and Model Configuration

Following (Kaneko et al., 2020; Stahlberg and Kumar, 2021), we build our GEC model based on the transformer-big architecture (Vaswani et al., 2017). To compare with state-of-the-art approaches in GEC, which pre-train with synthetic data, we follow (Stahlberg and Kumar, 2021) to generate C4_{200M} which contains 200 million synthetic parallel sentences. We use C4_{200M} to pre-train our GEC model, and use the combination of NUCLE (Dahlmeier et al., 2013), FCE (Yannakoudakis et al., 2011), and CLang-8 (Rothe et al., 2021) to train our GEC model, using an encoder-decoder shared vocabulary of 10K byte pair encoding tokens.[7]

We evaluate the effectiveness of our data manipulation method on two datasets: the CoNLL-2014 test set (Ng et al., 2014) and the BEA-2019 test set (Bryant et al., 2019). The W&I training set is used for fine-tuning. The development dataset is the CoNLL-2013 dataset (Ng et al., 2013) (W&I dev) when evaluating on the CoNLL-2014 test set (BEA-2019 test set).

4.2 Experimental Setup

We build our GEC model using the fairseq toolkit (Ott et al., 2019). All experiments are carried out on one NVIDIA A100 GPU. We use the MaxMatch scorer (Dahlmeier and Ng, 2012) to evaluate performance on the CoNLL-2014 test set and the ERRANT scorer (Bryant et al., 2019) to evaluate performance on the BEA-2019 test set. We report the average scores of three runs, and use a one-tailed sign test with bootstrap resampling to carry out statistical significance tests. We generate the augmented samples with a beam size of 5, and choose all five predictions for each training sentence as augmented samples for both CoNLL-2014 test set and BEA-2019 test set.

Baselines. We evaluate the performance of our method by comparing against five baseline methods. We apply our method and all the baseline methods to the checkpoint before fine-tuning.[8] **CE:** Fine-tuning the model using cross entropy loss. **SS:** Fine-tuning the model using the scheduled sampling method (Bengio et al., 2015). **RN-Word:** Fine-tuning the model by randomly injecting noise at word level (Zhang et al., 2019). **RN-Sent:** Fine-tuning the model by our modified version of sentence-level noise injection (Zhang et al., 2019) to better adapt to the GEC task. Specifically, we choose a noisy sentence with the smallest edit-

---

[7] More training details are given in Appendix A.7.

[8] The model checkpoint has been pre-trained first on C4_{200M} and then further trained on the combination of NUCLE, FCE, and CLang8.

System	CoNLL-14	BEA-19
Single System		
(Kiyono et al., 2019)†	61.3	64.2
(Lichtarge et al., 2020)†	62.1	66.5
(Wan et al., 2020)†	63.5	65.5
(Stahlberg and Kumar, 2021)†	66.6	70.4
(Rothe et al., 2021)*	68.9	75.9
(Sun et al., 2021)#	66.4	72.9
DM†	66.8	71.5
Ensemble		
(Kiyono et al., 2019)	65.0	70.2
(Wan et al., 2020)	65.9	70.0
(Lichtarge et al., 2020)	66.8	73.0
(Stahlberg and Kumar, 2021)	68.3	74.9
DM	68.5	74.8

Table 2: $F_{0.5}$ scores of state-of-the-art seq2seq GEC systems on the CoNLL-2014 and BEA-2019 test sets. #: Variant of BART-large model; *: T5-xxl model; †: transformer-big model.

distance, instead of the highest BLEU score, as the sentence-level oracle. We show the difference in Appendix A.3. **Ad-Bridge:** Fine-tuning the model by the AdapBridge method (Xu et al., 2021) which adaptively injects noise on word-level with word-level matching score. **RE-DP:** Weighting each augmented sentence using delta-perplexity during fine-tuning (Lichtarge et al., 2020).

4.3 Experimental Results

Approach	CoNLL-2014			BEA-2019		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
CE	70.3	51.0	65.4	70.8	67.7	70.2
SS	70.7	51.0	65.6	71.6	67.2	70.7
RN-Word	71.1	51.0	65.9	71.5	67.6	70.7
RN-Sent	71.8	50.8	66.3	71.7	67.3	70.8
Ad-Bridge	71.4	50.5	66.0	71.0	67.6	70.4
DA	73.6	47.4	66.3*	72.6	66.0	71.1
+RE-DP	73.7	47.5	66.4	72.8	65.8	71.3
DM	74.0	48.1	**66.8***†	73.1	65.5	**71.5**

Table 3: Experimental results (in %) on the CoNLL-2014 and BEA-2019 test sets. DA represents fine-tuning the model using only the data augmentation approach mentioned in Section 3.2.1 where each γ_j in Eq. 5 is set to $1/n$. DM represents fine-tuning the model using our proposed data manipulation method (including both data augmentation and data reweighting). Statistically significant improvements ($p < 0.01$) over CE and DA are marked as * and † respectively. Since the BEA-2019 test set is a blind test set, we are unable to carry out statistical significance tests on it.

The performance of our data manipulation method is shown in Table 3.[9] All methods which aim to mitigate the exposure bias problem (i.e., SS, RN-Word, RN-Sent, Ad-Bridge, DA, and DM) outperform CE on both CoNLL-2014 and BEA-2019 test sets. This shows the importance of addressing the exposure bias problem in GEC.

Furthermore, our DM method outperforms all baseline methods on both test sets. Specifically, on the CoNLL-2014 test set, DM achieves an $F_{0.5}$ score of 66.8%, which is 0.5% higher than RN-Sent. On the BEA-2019 test set, DM achieves an $F_{0.5}$ score of 71.5%, which improves by 0.7% compared to RN-Sent. DM performs better since it is able to better approximate the decoder input distribution at inference time and adjust the amount of errors automatically based on the model θ.

We also compare DM with state-of-the-art seq2seq GEC systems in Table 2. Specifically, we show the performance of a single DM model and an ensemble of eight analogously trained DM models. Among all the transformer-big single systems, our method achieves the best $F_{0.5}$ scores on both CoNLL-2014 and BEA-2019 test sets, surpassing (Stahlberg and Kumar, 2021) by 0.2% and 1.1%, respectively. Although Rothe et al. (2021) achieve $F_{0.5}$ scores of 68.9% and 75.9% on the CoNLL-2014 and BEA-2019 test set respectively, they adapt the T5-xxl model with 11 billion parameters, whereas our transformer-big model only uses 220 million parameters (2% of the size of T5-xxl model). The corresponding $F_{0.5}$ scores of (Rothe et al., 2021) for their T5-base model with

---

[9] The results on the CWEB and JFLEG test sets are shown in Appendix A.8.

Example 1	
Source	~~Many~~ Much of this surveillance ~~are~~ is being implemented in the government sectors ~~,~~ or military areas to enhance their security.
CE	Many of this surveillance are being implemented in the government sectors ~~,~~ or military areas to enhance their security.
RN-Sent	Many of this surveillance ~~are~~ is being implemented in the government sectors ~~,~~ or military areas to enhance their security.
DM	Many of this surveillance ~~are~~ is being implemented in the government sectors ~~,~~ or military areas to enhance their security.
Example 2	
Source	Surveillance technology will help to prevent ~~the family~~ families ~~to~~ from ~~loss~~ losing their ~~member~~ members, especially the elderly and the children ~~which~~ who need to be ~~pay~~ paid more attention to.
CE	Surveillance technology will help to prevent ~~the~~ family to loss their member, especially the elderly and the children ~~which~~ who need to be pay more attention to.
RN-Sent	Surveillance technology will help to prevent ~~the~~ family to loss their ~~member~~ members, especially the elderly and the children ~~which~~ who need to be pay more attention to.
DM	Surveillance technology will help to prevent ~~the~~ family ~~to~~ from ~~loss~~ losing their ~~member~~ members, especially the elderly and the children ~~which~~ who need to be pay more attention to.

Table 4: Examples of the error accumulation problem, with the predictions taken from CE, RN-Sent, and DM models. Red texts show the correct edits.

220 million parameters are 65.1% and 69.4% for the CoNLL-2014 and BEA-2019 test set respectively.

Our DM ensemble achieves an $F_{0.5}$ score of 68.5% on the CoNLL-2014 test set and reaches an $F_{0.5}$ score of 74.8% on the BEA-2019 test set, which is close to the best performance achieved by (Stahlberg and Kumar, 2021). Note that Stahlberg and Kumar (2021) have used more than 540 million synthetic sentence pairs to pre-train their transformer big model, while we only use 200 million synthetic sentence pairs. Note that the current best $F_{0.5}$ scores achieved by an ensemble approach (Qorib et al., 2022) on the CoNLL-2014 and BEA-2019 test sets are 69.51% and 79.90% respectively. However, we have not included them in Table 2 as Qorib et al. (2022) combine sequence-to-sequence models with sequence-tagging models.

4.4 Effect of Reweighting

We analyze the sampling distribution for each augmented sentence in Figure 2, and show the error density (in terms of the number of edits required to correct to grammatical sentence) for each augmented sentence in Table 5. Specifically, the error density is measured by the average number of edits required to transform an augmented sentence into the target sentence.

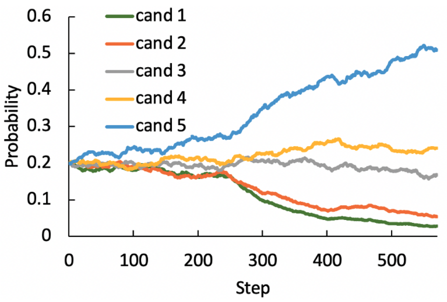


Figure 2: The sampling probability of augmented sentences during training.

As shown in Table 5 and Figure 2, the sampling probability for candidates with higher error density keeps increasing during training. This shows that as training progresses, candidates with higher error density play a more important role in training the model, which supports the hypotheses from (Zhang et al., 2019; Bengio et al., 2015).

Compared to RE-DP[10] which uses a static error density to train the model, our DM method achieves better performance by dynamically adjusting the error density according to the model θ. This further confirms that a varying error density is more

---

[10] We show the detailed settings of RE-DP in Appendix A.2.

Sentence	err%	e/s
WI train	67.1	3.45
BS-1	54.8	2.64
BS-2	90.7	2.18
BS-3	95.6	2.24
BS-4	96.9	2.30
BS-5	98.5	2.39
RN-Sent	-	7.82

Table 5: Statistics of the W&I training set and different augmented sentences. BS-x denotes the sentence which is the beam search candidate at position x. e/s is the number of edits per sentence calculated on erroneous sentences. err% is the percentage of erroneous sentences in the entire set. The err% for RN-Sent varies according to the different training epochs, with the details given in Appendix A.1.

effective in addressing the exposure bias problem.

4.5 Effect on Exposure Bias

In this section, we first present representative examples to illustrate the error accumulation problem in GEC. Then we conduct comprehensive studies to show the effectiveness of our method in addressing the exposure bias problem.

4.5.1 Effect on Error Accumulation

In Example 1 from Table 4, the CE model fails to correct “Many” to “Much” (since “surveillance” is an uncountable noun, “Much” should be used instead of “Many”). Failing to correct this error causes the CE model to also fail to correct “are” to “is”, which illustrates the error accumulation problem. Although RN-Sent and DM model also fail to correct “Many” to ‘Much”, they successfully correct “are” to “is”. This shows that both RN-Sent and DM are better able to address the error accumulation problem caused by exposure bias.

In the second example, the output generated by CE, RN-Sent, and DM all fail to correct “family” to “families”. For the output of CE, this further causes the model to fail to correct “to loss” to “from losing” and “member” to “members”. For RN-Sent, failing to correct “family” to “families” causes the model to fail to correct “to loss” to “from losing”, but it successfully corrects “member” to “members”. For the DM model, it is able to recover from failing to correct “family” to “families” and successfully corrects both “to loss” to “from losing” and “member” to “members”. Therefore, the error accumulation problem is better addressed by our DM approach.

Source	Her impressive physique ~~asa~~ as well as her ~~extraorinbary~~ extraordinary faculties as a rumba dancer will not be forgotten.
DA	Her impressive physique asa well as her extraorinbary faculties as rumba dancer will not be forgotten.
DM	Her impressive physique asa well as her ~~extraorinbary~~ extraordinary faculties as a rumba dancer will not be forgotten.

Table 6: Examples of the error accumulation problem, with the predictions taken from the DA and DM model. Red texts show the correct edits.

In Table 6, we show example output produced by the DA and DM model. The source sentence contains the following three errors: change “asa" to "as", change "extraorinbary" to "extraordinary", and insert "a" before “rumba”. Failure to correct the first error causes DA to fail to correct "extraorinbary" to "extraordinary" and to fail to insert "a" before “rumba”. However, DM successfully corrects the last two errors even when it fails to correct the first error.

4.5.2 Length and Edit Analysis

In this section, we further analyze the effect of our method on reducing the error accumulation problem, which tends to be more severe with longer sentences or sentences that require more edits. Therefore, we analyze the performance with respect to different lengths of the source sentence and different numbers of edits in the gold annotation in Figure 3.

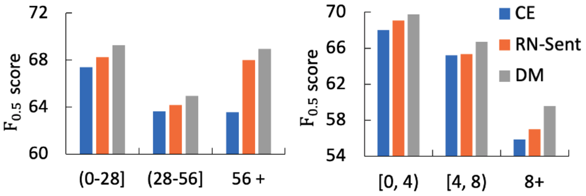


Figure 3: Performance comparison on the CoNLL-2014 test set with respect to different lengths of the source sentence, and different numbers of edits in the gold annotation. *Left:* source sentence length; *Right:* number of edits in the gold annotation.

Our method consistently improves over both CE and RN-Sent in all cases. The improvement is larger when the length of the source sentence is greater than 56, or the number of edits in the gold reference is greater than 8. This further shows that our method effectively addresses the error accumulation problem and that our improvement is mainly obtained by addressing the exposure bias problem.

4.5.3 Performance Analysis

This section further compares DM's capability of addressing the exposure bias problem against other baseline methods.

We consider the exposure bias problem to be better addressed when the predicted probability for the ground truth token is higher, under the condition that previously predicted tokens are generated by the model itself (i.e., not the ground truth). To evaluate how the exposure bias problem is addressed, we further design an experiment, which is an improved version of Zhang et al. (2019).

Specifically, we first generate noisy sentences by feeding the source sentences from the CoNLL-2014 test set to the model checkpoint before fine-tuning. We then generate the padded noisy sentences using the same procedure described in Section 3.2.1. Subsequently, we use different models (model A and model B in Table 7) to decode the same source sentence by using the padded noisy sentences as the decoder input. Let N be the number of ground truth tokens whose probabilities in the predicted distributions produced by model B are greater than those produced by model A. The win ratio is calculated by dividing N by the total number of tokens in the gold sentences. The results are shown in Table 7.

Win ratio A B	CE	DM
SS	69.99%	35.26%
RN-Word	73.54%	39.04%
RN-Sent	76.41%	37.64%
DM	89.41%	-

Table 7: The win ratio of model B over model A.

From Table 7, we observe that more than 89% of the target tokens' predicted probabilities of our DM model are greater than the CE model, indicating that our DM method greatly reduces the exposure bias problem. Moreover, compared to the DM model, all the other models are shown to have a win ratio of less than 40%. This shows that our method better addresses the exposure bias problem in GEC.

5 Conclusion

In this paper, we propose a novel data manipulation approach that includes both data augmentation and data reweighting to mitigate the exposure bias problem for seq2seq GEC models. Our core idea is to inject realistic augmented sentences into the decoder input and to automatically sample the augmented sentences. We show the effectiveness of our method which achieves significant improvements on both the CoNLL-2014 and BEA-2019 test sets.

6 Limitations

While our proposed approach is language-independent, we have only tested our approach on the English GEC task.

Acknowledgements

We thank Muhammad Reza Qorib for helpful comments on this paper. This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2019-014). The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg).

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, page 1171–1179.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75.

Hannan Cao, Wenmian Yang, and Hwee Tou Ng. 2021. Grammatical error correction with contrastive learning in low error density domains. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4867–4874.

Yutian Chen, Matthew W. Hoffman, Sergio Gómez Colmenarejo, Misha Denil, Timothy P. Lillicrap, Matt Botvinick, and Nando de Freitas. 2017. Learning to learn without gradient descent by gradient descent. In *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 748–756.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.

Mariano Felice, Christopher Bryant, and Ted Briscoe. 2016. Automatic extraction of learner errors in ESL sentences using linguistically enhanced alignments. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 825–835.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 1563–1572.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1236–1242.

Jared Lichtarge, Chris Alberti, and Shankar Kumar. 2020. Data weighted training strategies for grammatical error correction. *Transactions of the Association for Computational Linguistics*, pages 634–646.

Tsvetomila Mihaylova and André F. T. Martins. 2019. Scheduled sampling for transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 351–356.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Muhammad Qorib, Seung-Hoon Na, and Hwee Tou Ng. 2022. Frustratingly easy system combination for grammatical error correction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1964–1974.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, 2016, Conference Track Proceedings*.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 702–707.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2017. Grammatical error correction with neural reinforcement learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 366–372.

Danielle Saunders, Felix Stahlberg, and Bill Byrne. 2020. Using context in neural machine translation training objectives. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7764–7770.

Chenze Shao, Xilin Chen, and Yang Feng. 2018. Greedy search with probabilistic n-gram matching for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4778–4784.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, pages 1917–1928.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5937–5947.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Zhaohong Wan, Xiaojun Wan, and Wenguang Wang. 2020. Improving grammatical error correction with data augmentation by editing latent representation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2202–2212.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8526–8537.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, pages 229–256.

Lijun Wu, Fei Tian, Yingce Xia, Yang Fan, Tao Qin, Lai Jian-Huang, and Tie-Yan Liu. 2018. Learning to teach with dynamic loss functions. In *Advances in Neural Information Processing Systems*, pages 6467–6478.

Haoran Xu, Hainan Zhang, Yanyan Zou, Hongshen Chen, Zhuoye Ding, and Yanyan Lan. 2021. Adaptive bridge between training and inference for dialogue generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2541–2550.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343.

A Appendix

A.1 Error Density of RN-Sent

We adopt a similar approach to measure the error density in RN-Sent when training on the W&I training set. Specifically, our DM method only requires one epoch to converge, while RN-Sent requires seven epochs to converge. The average edits for each augmented sentence is 7.82, and the probability of creating augmented sentences is shown in Table 8.

epoch	p
1	0.01
2	0.02
3	0.03
4	0.05
5	0.07
6	0.11
7	0.17

Table 8: Probability of creating augmented sentences for each epoch.

The high edits per sentence generated by RN-Sent confirm that the augmented sentence generated through random noise injection has a bigger gap between actual decoder input and less efficiently addresses the exposure bias problem. Note that the original decay hyper-parameter is set to 5,800, with a maximum 40,000 updates[11]. Since our model is trained for 7 epochs (690 updates), we adjust the decay hyper-parameter to 100 accordingly.

A.2 RE-DP Setting

When using RE-DP to reweight each augmented sentence, we follow the setting in (Lichtarge et al., 2020). Specifically, we treat D_{aug} as the base dataset D^-, and D_{train} as the target dataset D^+. For a trained GEC model θ_{train}, it has been firstly pre-trained on C4$_{200\text{M}}$ and then trained on the combination of NUCLE, FCE, and CLang-8 dataset. We first obtain the model θ^- by fine-tuning θ_{train}

---

[11] `https://github.com/ictnlp/OR-NMT#results-and-settings-on-wmt14-english-german-translation-task`

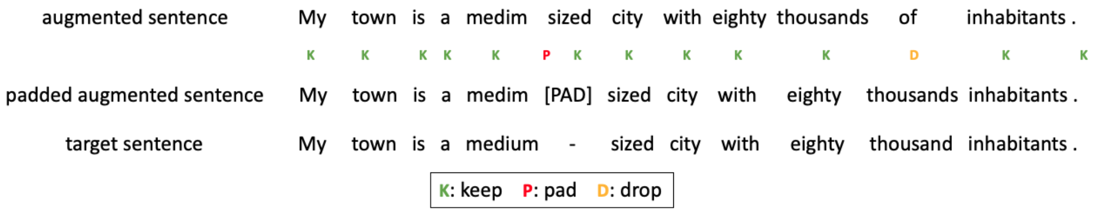


Figure 4: Illustration of the padding mechanism.

on D^- with the following loss function:

$$L_{RE-DP} = \sum_{j=1}^{n} l(x, \bar{y}_j, y; \theta)$$

The model θ^+ is obtained by further fine-tuning θ^- on D^+ using Eq. 2. Based on θ^- and θ^+, we calculate the perplexity of each augmented sentence. The weight $\alpha_{x,j}$ for each augmented sentence is calculated by the 'delta-perplexity-rank' method proposed in (Lichtarge et al., 2020). We then normalize $\alpha_{x,j}$, such that $\sum_{j=1}^{n} \alpha_{x,j} = 1$, and use Eq. 4 to fine-tune θ_{train}.

A.3 BLEU vs Edit Distance

We compare the performance of using BLEU to select sentence-level oracle with the performance of using edit distance to select sentence-level oracle in Table 9.

	CoNLL-2014	BEA-2019
BLEU	65.5	70.0
EDIT	66.3	70.8

Table 9: $F_{0.5}$ score on the CoNLL-2014 and BEA-2019 test sets. BLEU: represents selecting sentence-level oracle with the highest BLEU score. EDIT: represents selecting sentence-level oracle with the smallest edit distance.

A.4 Padding Mechanism

In Figure 4, we illustrate the padding mechanism introduced in Section 3.2.1.

A.5 Dataset Usage

The usage of different datasets for different setups is summarized in Table 10. Specifically, the W&I test set contains 4,477 sentences and the CoNLL-2014 test set contains 1,312 sentences.

A.6 Experimental Details

In this part, we will introduce the software packages we have used and the implementation details.

	CoNLL-2014	BEA-2019
Pre-training	$C4_{200M}$	
Training	NUCLE, FCE, CLang-8	
Fine-tuning	W&I train	W&I train
Validation	CoNLL-2013	W&I dev
Test	CoNLL-2014	W&I test

Table 10: Dataset usage for different setup cases.

Software configuration All models are implemented based on the Fairseq[12] and PyTorch packages. More specifically, we use Python 3.7 and PyTorch 1.7.1.

Implementation details When calculating the reward for the data scorer in Eq. 8, we use all the validation data to calculate the validation gradient.

To balance efficiency and performance, we set the update frequency f in line 3 of Algorithm 1 to 250 and 220 when fine-tuning for CoNLL-2014 and BEA-2019, respectively. The update frequency is selected from the range {250, 300} with a step size of 10 using grid search.

A.7 Hyper-Parameters and Computational Budget

We list the hyper-parameters for pre-training and training in Table 11. The hyper-parameters to fine-tune the model with the DM approach are shown in Table 12 . Specifically, pre-training takes 202 hours to converge, and training takes 0.6 hours to converge. Fine-tuning on the W&I training set takes 0.4 hours to converge.

A.8 Performance on CWEB-S/G and JFLEG Test Sets

We show the performance on the CWEB-S/G and JFLEG test sets in Table 13. When testing on the CWEB-S/G test set, we follow the setting in Cao et al. (2021) to extract the CWEB-train and CWEB-dev data. We use CWEB-train as the fine-tuning set

---

[12] https://github.com/pytorch/fairseq/tree/9f4256edf60554afbcaadfa114525978c141f2bd

Pretraining	
Configuration	Value
Devices	2 NVIDIA A100 GPU
Model architecture	Transformer ("large" setting)
Optimizer	Adam (β_1 = 0.9, β_2 = 0.98, $\epsilon = 1 \times 10^{-8}$)
Learning rate	3.00×10^{-4}
Learning rate scheduler	Inverse sqrt
Dropout	0.3
Warmup	8000
Number of epochs	10
Best epoch	8
Training	
Configuration	Value
Devices	1 NVIDIA A100 GPU
Learning rate	3.00×10^{-5}
Warmup	4000
Best epoch	1

Table 11: Pre-training and training configuration.

CoNLL-2014 & BEA-2019	
Configuration	Value
Devices	1 NVIDIA A100 GPU
Model architecture	Transformer ("large" setting)
Optimizer	Adam (β_1 = 0.9, β_2 = 0.98, $\epsilon = 1 \times 10^{-8}$)
Learning rate	3.00×10^{-5}
Learning rate scheduler	Inverse sqrt
Warmup	4000
Number of epochs	10
Best epoch	1

Table 12: DM fine-tuning configuration.

and CWEB-dev as the validation set. When testing on the JFLEG test set, we still use the W&I training set as the fine-tuning set and use the JFLEG-dev set as the validation set.

Approach	CWEB-S	CWEB-G	JFLEG
CE	38.2	43.0	62.9
SS	33.8	46.6	63.1
RN-Word	37.3	47.8	62.6
RN-Sent	37.7	47.7	63.0
Ad-Bridge	38.8.	47.5	63.1
DA	42.4*	47.0*	63.2*
DA + RE-DP	40.1	47.8	62.7
DM	42.7*†	49.0*†	63.5*†

Table 13: The $F_{0.5}$ score and GLEU score (in %) on the CWEB-S/G test set and JFLEG test set respectively. Statistically significant improvements ($p < 0.01$) over CE and DA are marked as * and † respectively.

Plausible May Not Be Faithful: Probing Object Hallucination in Vision-Language Pre-training

Wenliang Dai† **Zihan Liu***◇ **Ziwei Ji**† **Dan Su**† **Pascale Fung**†
†The Hong Kong University of Science and Technology
◇NVIDIA
`wenliang.dai@connect.ust.hk`, `pascale@ece.ust.hk`

Abstract

Large-scale vision-language pre-trained (VLP) models are prone to hallucinate non-existent visual objects when generating text based on visual information. In this paper, we systematically study the object hallucination problem from three aspects. First, we examine recent state-of-the-art VLP models, showing that they still hallucinate frequently, and models achieving better scores on standard metrics (e.g., CIDEr) could be more unfaithful. Second, we investigate how different types of image encoding in VLP influence hallucination, including region-based, grid-based, and patch-based. Surprisingly, we find that patch-based features perform the best and smaller patch resolution yields a non-trivial reduction in object hallucination. Third, we decouple various VLP objectives and demonstrate that token-level image-text alignment and controlled generation are crucial to reducing hallucination. Based on that, we propose a simple yet effective VLP loss named ObjMLM to further mitigate object hallucination. Results show that it reduces object hallucination by up to 17.4% when tested on two benchmarks (COCO Caption for in-domain and NoCaps for out-of-domain evaluation).

1 Introduction

Thanks to the advancement of large pre-trained Language Models (LMs) and Vision-Language Pre-training (VLP) methods, models are able to achieve surprisingly good performance in vision-conditioned text generation, e.g., image captioning. However, large LMs are found to generate unfaithful or nonsensical texts given the source input (Ji et al., 2022), which is called hallucination. The hallucination problem is also inherited by VLP models (Alayrac et al., 2022), as they are still LMs that can understand visual information. VLP models often generate fluent and seem appropriate sentences if we only see the text, but wrong when taking the visual input into consideration. One major type of hallucination in VLP is known as object hallucination (Rohrbach et al., 2018), where models generate texts containing non-existent or inaccurate objects from the input images. Object hallucination in VLP models essentially limits their performance and raises safety concerns for industrial applications. For example, in biomedical image captioning (Pavlopoulos et al., 2019), object hallucination reduces the accuracy of diagnosis and may lead to severe consequences for the patient. Despite the limitations and risks caused by object hallucination, this problem has not been studied in contemporary VLP works yet.

To narrow down the aforementioned research gap, in this paper, we systematically investigate four fundamental research questions about object hallucination: 1) how much do modern VLP models hallucinate? 2) how do different forms of image encoding affect object hallucination? 3) what are the effects of common VLP objectives on object hallucination? and 4) how to mitigate object hallucination in VLP models?

For our first question, we examine recent state-of-the-art VLP models on the image captioning task. To evaluate object hallucination, we adopt and expand the CHAIR (Caption Hallucination Assessment with Image Relevance) metric proposed by Rohrbach et al. (2018). Results show that these models still hallucinate frequently with ∼10% of the generated sentences containing at least one hallucinated object. This problem becomes much severer when generating sentences given out-of-domain images. Furthermore, we discover that the widely used optimization method SCST (Rennie et al., 2017) could lead to more hallucination, even if it improves standard metrics like CIDEr (Vedantam et al., 2015).

For our second question, to investigate how different types of image encoding in VLP influence hallucination, we ablate three commonly

---

[*] Work done during PhD at HKUST.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2136–2148
May 2-6, 2023

used ones, including region-based, grid-based, and patch-based (Kim et al., 2021). Surprisingly, we find that patch-based features perform the best and smaller patch resolution yields a non-trivial reduction in object hallucination.

Thirdly, we analyze the effects of commonly adopted vision-language pre-training objectives on object hallucination. Specifically, we decouple and combine the image-text contrastive (ITC) loss, the image-text matching (ITM) loss with and without hard negatives, and the image-conditioned language modeling (ICLM) loss. Counter-intuitively, although ITC and ITM help to bring apart dissimilar images and texts, results show that they do not contribute much to alleviating object hallucination. The generative ICLM loss is the main influential factor of object hallucination and different pre-training datasets lead to distinctive model behaviors. More detailed analysis is described in Section 5.3.

Finally, we propose a simple yet effective new vision-language pre-training loss, namely object-masked language modeling (ObjMLM), to further mitigate object hallucination by enhancing the alignment and restriction between text tokens and visual objects during generation. Code and evaluation setups are released: `https://github.com/wenliangdai/VLP-Object-Hallucination`.

Overall, our contributions are three-fold:

- This is the first paper that systematically studies state-of-the-art VLP models on the object hallucination problem, proving that it is still far from resolved and previous methods that improve standard metrics may reflect in worse hallucination.
- We thoroughly investigate the influence of different VLP losses and image encoding methods on object hallucination. Our findings could be valuable for future work to build more responsible VLP systems.
- We present a new pre-training objective ObjMLM to mitigate object hallucination. Experimental results show that it reduces object hallucination by 17.4% without introducing extra training data.

2 Related Work

2.1 Hallucination in Deep Learning

Generally, the term *hallucination* denotes the appearance of undesirable output that is unfaithful to the conditional input (Maynez et al., 2020), even though it may appear to be fluent or reasonable. In the multimodal field, the hallucination phenomenon refers to the prediction of non-existent or incorrect objects (e.g., in object detection or image captioning) and is called *object hallucination* (Rohrbach et al., 2018; Biten et al., 2022). Despite the success of large pre-trained models, they still suffer the hallucination problem, which degrades the performance and largely hinders practical applications (Ji et al., 2022).

Many works have been proposed to mitigate hallucination in recent years. Nie et al. (2019) applied data refinement with self-training to improve the equivalence between the input and the paired text in the data-to-text generation task. Zhang et al. (2021b) and Zhang et al. (2020) proposes scene graph learning methods to ground the process of visual captioning to reduce hallucination. Ma et al. (2020) reconstruct generated sentences from localized image regions. Xiao and Wang (2021) proposed the uncertainty-aware beam search as an add-on technique to the original beam search, in both image captioning and data-to-text generation. To reduce hallucination in dialog systems, Shuster et al. (2021) introduced knowledge augmentation and Dziri et al. (2021) presented a post-processing method to refine generated outputs. Su et al. (2022) augment models with answer-related information predicted by a machine reading comprehension module to reduce hallucination in the generative question answering task.

2.2 Vision-Language Pre-training

The research on vision-language pre-training (VLP) has progressed vastly in recent years. Due to the demand for large-scale data, most VLP methods use self-supervised pre-training objectives to utilize image-text pairs crawled from the web. In the beginning, BERT (Devlin et al., 2019)-style VLP models (Lu et al., 2019; Tan and Bansal, 2019; Li et al., 2020b; Chen et al., 2020; Yu et al., 2021a; Shen et al., 2022) are trained to perform multimodal understanding tasks, using objectives like image-text matching and masked language modeling. Later, encoder-decoder architectures are introduced to additionally handle multimodal generation tasks with a causal language modeling loss (Li et al., 2021b; Yu et al., 2021b; Lin et al., 2021; Cho et al., 2021; Ding et al., 2021; Li et al., 2022; Wang et al., 2022a). Another line of research uses a dual-stream architecture (Radford et al., 2021; Jia et al., 2021; Zhai et al., 2022; Yao et al., 2022) with separate image and text encoders aligned together through an image-text contrastive loss. They improve the performance of various multimodal downstream tasks by a large step.

Alayrac et al. (2022) show that fatal object hallucination can happen naturally or be provoked by the adversarial prompting in modern VLP models. However, in previous works, how different VLP strategies influence the faithfulness of generated text given images has not been studied. Moreover, the effects of using different types of image encoding are also unclear, including region-based (Li et al., 2020c; Zhang et al., 2021a; Hu et al., 2022), grid-based (Wang et al., 2022b), and patch-based (Kim et al., 2021; Li et al., 2021a).

3 Evaluation Setup

In this section, we first introduce the CHAIR evaluation metric and our proposed improvements to it in Section 3.1. Then, in Section 3.2, we describe two datasets used for evaluation and explain how to calculate CHAIR scores under such settings.

3.1 Evaluation Metric

To automatically measure object hallucination, we adopt the CHAIR (Caption Hallucination Assessment with Image Relevance) metric proposed by Rohrbach et al. (2018). CHAIR calculates what proportion of generated object words are not in the image (i.e., hallucinated) according to the ground truth. CHAIR has two variants: CHAIR$_i$ (instance-level) and CHAIR$_s$ (sentence-level), which are formulated as follows:

$$\mathrm{CHAIR}_i = \frac{\#\,\{\text{hallucinated objects}\}}{\#\,\{\text{all objects in prediction}\}},$$

$$\mathrm{CHAIR}_s = \frac{\#\,\{\text{hallucinated sentences}\}}{\#\,\{\text{all sentences}\}}.$$

As formulated, CHAIR$_i$ represents the proportion of hallucinated objects over all golden objects in all data samples. It can be seen as the probability of a generated object to be a hallucination. On the other hand, CHAIR$_s$ measures the proportion of generated sentences that contain at least one hallucinated object. Therefore, to calculate CHAIR$_i$ and CHAIR$_s$, we need a pre-defined list of golden object categories to recognize objects in the text. We illustrate dataset-specific calculation details in Section 3.2.

3.2 Evaluation Datasets

To evaluate models' performance on object hallucination with CHAIR, we adopt two widely used benchmarks: Microsoft COCO Caption (Lin et al., 2014) and NoCaps (Agrawal et al., 2019). For all models, the COCO Caption training set is used for the finetuning of the image captioning task, and COCO Caption test set and NoCaps valid set are used for in-domain and out-of-domain evaluation, respectively. In the following, we introduce statistics of each dataset and how to calculate CHAIR on them.

3.2.1 COCO Caption

The COCO Caption (Lin et al., 2014) is a large-scale and widely used dataset for the training and evaluation of the image captioning task. We use the Karpathy split (Karpathy and Fei-Fei, 2017), in which 82K, 5K, and 5K images are in the train, validation, and test sets, respectively. Each image is annotated with at least five ground truth captions.

To calculate CHAIR scores on this dataset, we follow the setting proposed in Rohrbach et al. (2018). In practice, we first tokenize each sentence and then singularize each word. Then, we use a list of synonyms from Lu et al. (2018) to map fine-grained objects to the pre-defined 80 coarse-grained MSCOCO object categories (e.g., mapping "puppy", "chihuahua", "poodle" to the "dog" category). The purpose of doing this mapping is to ensure that we do not detect hallucinated objects by mistake. For example, when the ground-truth caption only has the "puppy" object, the CHAIR metrics will undesirably consider the "dog" object generated by models as a hallucinated object if we do not perform the mapping.

3.2.2 NoCaps

The NoCaps (Agrawal et al., 2019) dataset aims to evaluate models trained on the training set of COCO Caption to examine how well they generalize to a much larger variety of visual concepts, i.e., unseen object categories. There are 4,500 images in the validation set and 10,600 images in the test set. Images are taken from the Open Images V4 (Kuznetsova et al., 2020) dataset, which contains 600 object classes. Due to the unavailability of ground truth captions of the test set, we use the valid set of NoCaps.

To calculate CHAIR scores on NoCaps, we setup a similar setting as used in COCO Caption. Specifically, we map the fine-grained classes defined in

Model	CIDEr Optim (SCST)	# Pretrain Image-Text Pairs	COCO Caption Karpathy Test						NoCaps Validation Out-of-domain			
			B@4↑	C↑	M↑	S↑	CH$_i$↓	CH$_s$↓	C↑	S↑	CH$_i$↓	CH$_s$↓
OSCAR $Base^{*}$	✗	6.5M	34.4	117.6	29.1	21.9	7.1	13.0	-	-	-	-
OSCAR $Base^{*}$	✓	6.5M	39.6	134.2	29.8	23.5	7.2	13.5	-	-	-	-
VinVL *Base*	✗	6.5M	38.2	129.3	30.3	23.6	5.3	10.0	83.1	10.8	12.1	21.2
VinVL *Base*	✓	6.5M	40.9	140.4	30.9	25.1	5.7	10.9	87.5	11.7	17.4	32.1
VinVL *Large*	✗	6.5M	38.5	130.8	30.4	23.4	5.5	10.5	-	-	-	-
VinVL *Large*	✓	6.5M	41.0	140.9	31.1	25.2	5.6	10.6	-	-	-	-
BLIP *Base*	✗	129M	39.7	133.3	31.0	23.8	4.9	8.9	112.1	14.2	6.6	10.5
BLIP *Large*	✗	129M	40.4	136.7	31.1	24.3	4.7	8.8	115.3	14.4	6.4	10.5
OFA *Large*	✗	21M†	41.7	140.5	31.2	24.2	4.7	8.9	103.2	13.3	6.4	10.2
OFA *Large*	✓	21M†	43.8	149.5	31.8	25.9	4.2	8.1	113.1	15.2	7.1	12.4

Table 1: Image captioning results of recent state-of-the-art VLP models (Li et al., 2020c; Zhang et al., 2021a; Li et al., 2022; Wang et al., 2022a) on the COCO Caption Karpathy test set and NoCaps validation set. Here, B@4, C, M, S, and CH denote BLEU-4, CIDEr, METEOR, SPICE, and CHAIR, respectively. CIDEr Optim indicates whether the SCST CIDEr optimization is used or not. All results are generated by using their officially provided checkpoints and hyper-parameters, * means the model is finetuned by us as the provided one is broken. † denotes the model also uses unimodal data besides image-text pairs.

NoCaps to coarse-grained categories based on the hierarchical object relationship[1] to improve the effectiveness of CHAIR metrics. We only add two types of object categories to our final object list: 1) super-categories that have sub-categories, and 2) object categories that have neither super-category nor sub-categories. Eventually, we construct a list of 139 coarse-grained object categories from the 600 classes.

4 Object Hallucination in VLP Models

Benefitting from the vast advancement of various VLP methods, the performance of image captioning has been improved a lot by following a pretrain-then-finetune schema. Generally, the performance is measured by metrics like CIDEr (Vedantam et al., 2015), SPICE (Anderson et al., 2016), METEOR (Banerjee and Lavie, 2005), and BLEU (Papineni et al., 2002), which consider the semantic and syntactic similarity or n-gram-based fluency between the model generated and ground truth captions. However, the faithfulness of captions generated by VLP models is neglected.

In this section, we provide a thorough analysis of recent VLP models to investigate how much they hallucinate when generating text conditioned on visual information. The results are shown in Table 1. Models are finetuned on the COCO Caption training set and evaluated on both the COCO Caption test set and the NoCaps valid set.

Ground Truth: "A green garbage can has an orange face on it."

VinVL$_{Base}$ w/o SCST: "A green waste container with a face painted on it."

VinVL$_{Base}$ w/ SCST: "A green waste container with a picture of a **dog** on it."

Ground Truth: "A dresser with all of the drawers closed and something on top."

OFA$_{Large}$ w/o SCST: "A dresser with a bunch of drawers on it."

OFA$_{Large}$ w/ SCST: "A chest of drawers with a **mirror** on top of it."

Figure 1: Comparison of image captioning examples generated by VinVL$_{Base}$ and OFA$_{Large}$ with and without the SCST CIDEr optimization. Red color denotes the occurrence of object hallucination.

Overall, we observe two noteworthy insights. Firstly, similar to the findings in Rohrbach et al. (2018), for all CHAIR scores, they are not proportional to standard evaluation metrics. Although standard metrics (e.g., the cosine similarity in CIDEr) could potentially penalize the wrong object prediction, they do not directly reflect faithfulness. Captions can still have good scores from standard metrics as long as they contain sufficient accurate objects to fulfill coverage, even if hallucinated objects exist. For example, VinVL$_{Large}$ achieves higher CIDEr and BLEU-4 scores than VinVL$_{Base}$, but its CHAIR scores are also higher.

---

[1] `https://github.com/nocaps-org/image-feature-extractors/blob/master/data/oi_categories.json`

Therefore, it is important to have a supplementary metric like CHAIR to reflect faithfulness besides other metrics.

Secondly, the Self-Critical Sequence Training (SCST) (Rennie et al., 2017) for the CIDEr optimization method harms the faithfulness of generated captions. SCST is a reinforcement learning algorithm that has been widely adopted as the second-stage finetuning after the standard cross-entropy optimization for image captioning (Anderson et al., 2018; Zhou et al., 2020; Li et al., 2020c; Zhang et al., 2021a; Hu et al., 2022; Wang et al., 2022a). It calculates the reward based on the CIDEr score by sampling captions during training without the need of another baseline. Although SCST can significantly boost performance on previous standard metrics, it encourages models to generate more hallucinated objects in the captions. For example, applying SCST improves the CIDEr score by 11.1 and BLEU-4 score by 2.7 for VinVL$_{Base}$, yet it also increases 0.9 CHAIR$_s$ score on the COCO Caption dataset.

While Rennie et al. (2017) also observed this phenomenon by testing small scale models, we show that SCST hurts VLP models less. When the model is pre-trained very well, the side effect of SCST is alleviated (e.g., the OFA large model). Moreover, we demonstrate that this problem becomes more serious on out-of-domain images. For the VinVL$_{Base}$ model, there are 10.9% more generated captions containing at least one hallucinated object after using SCST. We speculate that the CIDEr-based optimization encourages models to generate more words or phrases that have higher cosine similarities to the ground truth captions in the vision-language representation space, which can be plausible but not faithful.

We show a case study in Figure 1. After finetuned by SCST, models will take a bigger risk to generate more detailed yet incorrect information (e.g., in the second example in Figure 1, the sentence with hallucination generates the detailed information "mirror", which cannot be found in the image). This will further amplify the object hallucination problem on out-of-domain images, as models may have lower confidence in unfamiliar visual concepts.

5 Probing Image Encoding Methods and VLP Objectives

In this section, we systematically study two determinants in VLP that are intuitively influential to the severity of the object hallucination problem. Firstly, we study how different types of image encoding affect object hallucination, as they are the key components of models to interpret visual information. Specifically, we ablate three encoding approaches including region-based, grid-based, and patch-based. Secondly, we analyze how different VLP objectives influence object hallucination. We ablate three commonly used ones: image-text contrastive (ITC), image-text matching (ITM), and image-conditioned language modeling (ICLM). Implementation details are described in Appendix A.

5.1 Model Architecture

CLIP. CLIP (Radford et al., 2021) is a dual-stream VLP model that consists of an image encoder and a text encoder. It is pre-trained on 400 million image-text pairs data using a cross-modal contrastive loss. Specifically, CLIP explores the image encoder with different sizes of two architectures[2], including the ResNet (He et al., 2016) and the Vision Transformer (ViT) (Dosovitskiy et al., 2021). The resulting image and text encoders are aligned in the same multimodal feature space.

BERT. BERT (Devlin et al., 2019) is a Transformer (Vaswani et al., 2017) model pre-trained on a large corpus by the masked language modeling (MLM) and sentence permutation losses. It is shown to have excellent performance on various downstream tasks after finetuning. Moreover, BERT can also handle generation tasks when the self-attention layers are restricted to the left-to-right direction to generate text auto-regressively. In this paper, we refer to this variant as BertLM.

We design a flexible architecture that can plug in various visual encoders and fit modern VLP objectives without introducing extra influential factors. As shown in Figure 4, the model consists of two parts, a visual encoder to encode images and a text decoder to generate sentences conditioned on the image representations. We use two separate modules rather than a unified single-stream model, as it is convenient to alter the visual encoder while keeping the text decoder the same. Specifically,

---

[2] `https://github.com/openai/CLIP/blob/main/model-card.md`

Visual Encoder	#Params	COCO Karpathy Test			NoCaps Val Out-of-domain		
		C↑	CH_i↓	CH_s↓	C↑	CH_i↓	CH_s↓
Region features							
BUTD-RN101	45M	110.6	9.1	15.9	40.5	36.7	49.0
ResNeXt-152	60M	115.9	7.1	12.9	45.1	30.5	41.1
Grid features							
RN50×4	83M	107.6	11.2	19.1	41.6	37.5	49.9
RN50×16	160M	111.6	9.0	15.8	47.5	33.1	45.2
RN50×64	401M	115.8	7.5	13.2	56.2	26.3	36.6
Patch features							
ViT-B/32	84M	108.9	10.3	17.9	44.4	34.7	46.8
ViT-B/16	82M	111.8	8.1	14.7	51.9	30.3	42.3
ViT-L/14	290M	120.7	6.4	11.6	59.8	24.2	33.5

Table 2: Results of different types of visual encoders with the same BertLM text decoder on the COCO Karpathy test set and NoCaps validation set (out-of-domain).

for region-based image features, we explore the Faster R-CNN object detector (Ren et al., 2015) with two different backbones: the ResNet-101 used in BUTD (Anderson et al., 2018) and the ResNeXt-152 (Xie et al., 2017) used by Zhang et al. (2021a). They are both pre-trained on COCO (Lin et al., 2014) and Visual Genome (Krishna et al., 2016) datasets for object detection. For the grid-based and patch-based image features, we use the CLIP ResNet variants and CLIP ViT variants, respectively. The reason for using CLIP is that all its variants are pre-trained on the same data and there is a wide range of different model sizes. For all visual encoders, we use the same BertLM as the text decoder.

5.2 Effects of Different Image Features

Recognizing visual objects correctly is crucial for avoiding object hallucination. In Table 2, we compare the performance of different visual encoders with the same text decoder on COCO (in-domain) and NoCaps (out-of-domain) datasets.

Overall, patch-based visual encoders attain the best performance in terms of avoiding object hallucination. Models with grid features hallucinate more frequently when achieving comparable CIDEr scores to the other models. For example, on COCO, RN50×16 has a similar CIDEr score to ViT-B/16 but higher CHAIR_s, which is also observed between RN50×64 and ResNeXt-152. We conjecture that the inductive biases (Cohen and Shashua, 2017) of the Convolutional Neural Network (CNN), such as locality and translation invariance, weaken the connection of different characteristics of a single object and thus lead to more hallucination. Oppositely, regional or patch-level features are obtained by directly dividing images into different parts and further associating them through positional embeddings. In addition, we see that a smaller patch resolution helps to reduce object hallucination without enlarging the model size.

VLP Objectives	COCO Karpathy Test			NoCaps Val Out-of-domain		
	C↑	CH_i↓	CH_s↓	C↑	CH_i↓	CH_s↓
(a) None	120.7	6.4	11.6	59.8	24.2	33.5
Discriminative Objectives						
CC3M						
(b) ITC	120.5	6.5	11.7	59.9	24.4	33.8
(c) ITC_{Late}	121.2	6.2	11.3	60.5	23.8	32.9
(d) ITC_{Late} + ITM	121.0	6.3	11.5	60.2	23.9	33.1
(e) ITC_{Late} + ITM_{Hard}	120.9	6.6	11.7	59.9	24.2	33.3
Generative Objectives						
Visual Genome						
(f) LM	120.3	5.5	9.8	62.8	9.0	13.9
(g) LM + ObjectMLM	121.9	5.3	9.2	63.8	8.8	13.1
CC3M						
(h) LM	122.3	6.0	10.9	92.1	8.3	14.5
(i) LM + ObjectMLM	124.5	5.1	9.0	94.0	8.0	13.1
(c) + (i)	**125.1**	**4.9**	**8.8**	**94.5**	**7.9**	**12.5**

Table 3: Comparison of the effects of different VLP objectives and their combination on object hallucination.

For region-based visual encoders, although they achieve modest results on COCO with relatively small model sizes, their performance of object hallucination on out-of-domain images drops dramatically. One important reason is that the output of such encoders only contains representations of detected visual objects rather than the whole image, which may amplify detection errors as there is much less context. Moreover, as the object detector is pre-trained separately from the whole model and its parameters are fixed during finetuning, this gap could also aggravate object hallucination on unseen images.

5.3 Effects of Different VLP Objectives

Based on the best performing ViT-L/14 baseline, we explore three commonly used vision-language pre-training objectives and their variants that could potentially affect object hallucination.

5.3.1 Pre-training Datasets

We explore two pre-training datasets with image-text pairs: 1) the VG Caption from the Visual Genome (Krishna et al., 2016) dataset, which contains 10K images and each image has multiple corresponding descriptions; and 2) a more large-scale dataset CC3M (Sharma et al., 2018) that contains three millions of image-text pairs.

5.3.2 Image-Text Contrastive (ITC) Loss

The cross-modal contrastive loss is shown to be fairly effective in representation learning (Tian et al., 2020; Sigurdsson et al., 2020) and VLP (Radford et al., 2021; Li et al., 2021a, 2022). It aligns the visual and textual representations into the same multimodal feature space by shortening the distance between an image and a text if they are paired, or enlarging if they are not.

Counter-intuitively, as shown in Table 3 (b), ITC has negligible influence on the faithfulness of generated captions. We speculate that it only enhances the model's understanding of global-level representations rather than token-level alignment between images and texts. To verify, we further test the ITC with a more fine-grained token-level late interaction (ITC_{Late}) proposed by Yao et al. (2022). As shown in Table 3 (c), ITC_{Late} is more effective than the vanilla ITC and slightly reduces object hallucination. We think this benefits from the word-patch alignment ability enabled by ITC_{Late}.

5.3.3 Image-Text Matching (ITM) Loss

ITM is a widely used objective in VLP (Li et al., 2020a; Chen et al., 2020; Zhou et al., 2021). It is a binary classification task that aims to make the model learn whether an image and a sentence are paired or not. Based on that, ITM with hard negatives (ITM_{Hard}) is introduced to increase the difficulty of the task, which is shown to be very effective on representation learning (Kalantidis et al., 2020; Robinson et al., 2021; Li et al., 2021b). We follow the ITM loss proposed by Li et al. (2022), in which an in-batch negative example is sampled either uniformly (normal negative) or from the similarity distribution of image-text pairs computed by ITC (hard negative).

The results are exhibited in Table 3 (d) (e). Both ITM and ITM_{Hard} are not highly correlated with the object hallucination problem. They only slightly reduce hallucination in generated texts on out-of-domain images. Although the ITM_{Hard} can be seen as an analogy to the object hallucination problem (plausible but not correct) in a global and discriminative way, it has a negligible effect on reducing hallucination for downstream generative tasks.

5.3.4 Image-Conditioned Language Modeling

Various image-conditioned language modeling losses have been proposed in the VLP research, in the form of masked language modeling (MLM) (Sun et al., 2019; Tan and Bansal, 2019; Su et al., 2020), text infilling (Dai et al., 2022; Wang et al., 2022a), prefix LM (Wang et al., 2022b), and causal LM (Hu et al., 2022). This is one of the most crucial pre-training losses to activate the cross-modal text generation ability for the VLP model.

COCO Caption

"Several boats docked at a floating dock at a marina.",
"Several boats sitting on a docking station on the water.",
"A bunch of speedboats near a harbor with flags from all over the world.", etc.

Visual Genome Caption

"A dock in a city.", "Long silver dock in water."
"Very blue, calm water in marina.", "The water is calm."
"A dock is floating on the water.", "Row of docked boats.", etc.

Figure 2: Comparison of ground truth captions in COCO and Visual Genome datasets for the same image.

Ground Truth: "A soccer ball is next to a wall.", "A soccer ball that is placed on the ground.", etc.

ViT-L/14 w/o VG: "A close up of a soccer ball on a **table**."

ViT-L/14 w/ VG: "A close up of a soccer ball on the ground."

Ground Truth: "A large black printer seems to have a piece of paper in it sideways.", "A large printer with paper coming out of it", etc.

ViT-L/14 w/o VG: "A pair of **scissors** sitting on top of a piece of paper."

ViT-L/14 w/ VG: "A large black machine."

Figure 3: Comparison of generated captions with or without the image-conditioned language modeling pre-training on the VG dataset before finetuning.

We first examine the causal LM loss, which is exactly the same loss as the image captioning loss, but used in the pre-training on a much larger scale. Surprisingly, as shown in Table 3 (f), although pre-training on the VG Caption does not improve previous standard metrics like CIDEr, it helps to reduce object hallucination by a large margin when compared to (a).

There are two reasons behind this performance lift. Firstly, as described in Figure 2, for each image, VG contains more captions than COCO.

Each caption in VG is much shorter and only describes one specific aspect of the image, unlike the global descriptions in COCO. Therefore, pre-training on VG and then finetuning on COCO is a fine-to-coarse process. It enables models to first accurately describe different parts of an image and connect these clues together at a higher viewing point. Secondly, due to the nature of the short length of VG captions, the model becomes slightly more cautious. On average, after adding VG data in the pre-training, there are 0.08 and 0.24 fewer objects generated in each caption on COCO and NoCaps, respectively. This observation aligns with the sentence simplification method proposed by Biten et al. (2022), which simplifies sentences to augment data and further mitigate object hallucination. Figure 3 illustrates VG's effects on generated samples. The model is more faithful but more likely to lack some details when it is not confident.

For CC3M, we observe a leap in all metrics. It improves the general image translation ability of the model, which can be seen as large-scale data augmentation. This indicates that seeing a sufficient amount of data and co-occurrence of various objects during pre-training help to mitigate object hallucination to some extent. However, data augmentation may not be the key to drastically tackle object hallucination. As discussed in Section 4, object hallucination still happens frequently even if the model is pre-trained on large-scale data. Therefore, we believe that enhancing the controllability of vision-conditioned text generation would be a promising future direction. More case studies are exhibited in Appendix B.

6 Object Masked Language Modeling

Based on the findings in Section 5, we propose a simple yet effective pre-training objective to mitigate object hallucination by improving object-level image-text alignment. It is named Object Masked Language Modeling (ObjMLM). As shown in Figure 4, ObjMLM can be seen as a variant of the MLM loss by masking all the objects in the text that appear in the image. For each sentence, we mask the object words and phrases as defined in the object category lists of both COCO and NoCaps by performing exact matching. Similar to the whole word masking (Cui et al., 2021), we conduct whole object masking so that there will be only one `[MASK]` token to replace each object.

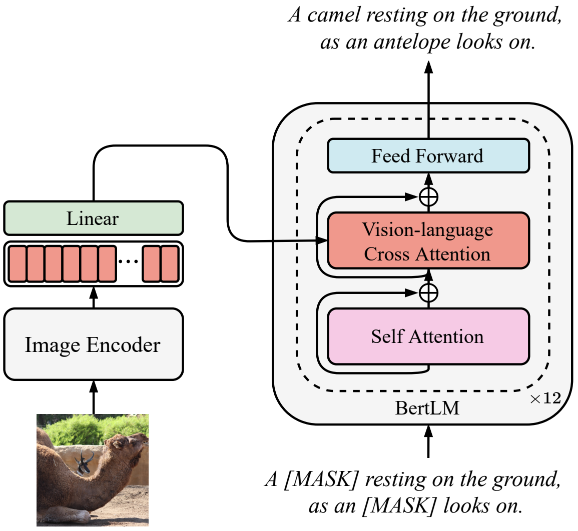


Figure 4: An overview of the model architecture and the training of our proposed ObjMLM. We use the same architecture as described in Section 5 to show the effectiveness of ObjMLM. Here, the image encoder can be one of the region-based, grid-based, or patch-based variants as described in Section 5.2. For ObjMLM, we use the ViT-L/14.

Compare the results shown in lines (h) and (i) of Table 3, by plugging ObjMLM into an existing VLP setting, the CHAIR$_s$ score is reduced by 17.4%. This is a non-trivial improvement without introducing more pre-training data. To further validate ObjMLM's effectiveness, we replace it by the standard MLM loss with a 15% masking rate. However, it only reduces CHAIR$_s$ by 1.7%, which is not significant. We conjecture that ObjMLM adds a constraint that indirectly controls the model to only generate objects that are visible in the input image. Additionally, ObjMLM enhances the model's recognition ability when describing the spatial relationship between objects, which is a common scenario that causes hallucinations frequently.

7 Conclusion

This paper systematically studies the objection hallucination phenomenon in VLP models, which is a severe problem but neglected in contemporary VLP works. We find that recent large VLP models still hallucinate frequently. Moreover, the widely used SCST method harms the faithfulness of generated sentences in image captioning, even if it improves previous standard metrics. Furthermore, we discover that image encoding matters and the patch-based input with smaller resolution helps mitigate object hallucination. Finally, we ablate commonly used VLP losses and show that token-level image-text alignment and controllability of the generation are crucial. We further propose a new loss named ObjMLM, which reduces object hallucination by 17.4% for an existing VLP setting. We believe our findings are beneficial for future work to build more responsible VLP models.

Limitations

We understand that the hallucination problem is a big research topic and it is not just limited to object hallucination. In this paper, we focus on the investigation and mitigation of object hallucination, leaving other types of hallucination in VLP for future work. Another limitation is that for the discussion of recent VLP models in Section 4, we only study those whose pre-trained checkpoints are publicly available. For the non-released ones, we cannot pre-train them by ourselves due to the lack of large-scale GPU power and private pre-training datasets.

References

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. nocaps: novel object captioning at scale. *International Conference on Computer Vision*, pages 8947–8956.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. 2022. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *ECCV*.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*.

Ali Furkan Biten, Lluís Gómez, and Dimosthenis Karatzas. 2022. Let there be a clock on the beach: Reducing object hallucination in image captioning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1381–1390.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *ECCV*.

Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1931–1942. PMLR.

Nadav Cohen and Amnon Shashua. 2017. Inductive bias of deep convolutional networks through pooling geometry. In *International Conference on Learning Representations*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Wenliang Dai, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. 2022. Enabling multimodal generation on CLIP via vision-language knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2383–2395, Dublin, Ireland. Association for Computational Linguistics.

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

Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, and Jie Tang. 2021. Cogview: Mastering text-to-image generation via transformers. In *NeurIPS*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Nouha Dziri, Andrea Madotto, Osmar Zaiane, and Avishek Joey Bose. 2021. Neural path hunter: Reducing hallucination in dialogue systems via path grounding. *EMNLP*.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

X. Hu, Z. Gan, J. Wang, Z. Yang, Z. Liu, Y. Lu, and L. Wang. 2022. Scaling up vision-language pretraining for image captioning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17959–17968, Los Alamitos, CA, USA. IEEE Computer Society.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Wenliang Dai, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation. *ACM Computing Surveys*.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. 2020. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21798–21809. Curran Associates, Inc.

Andrej Karpathy and Li Fei-Fei. 2017. Deep visual-semantic alignments for generating image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:664–676.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123:32–73.

Alina Kuznetsova, Hassan Rom, Neil Gordon Alldrin, Jasper R. R. Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. 2020. The open images dataset v4. *International Journal of Computer Vision*, 128:1956–1981.

Gen Li, Nan Duan, Yuejian Fang, Daxin Jiang, and Ming Zhou. 2020a. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *AAAI*.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*.

Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq R. Joty, Caiming Xiong, and Steven C. H. Hoi. 2021a. Align before fuse: Vision and language representation learning with momentum distillation. In *NeurIPS*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2020b. What does BERT with vision look at? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5265–5275, Online. Association for Computational Linguistics.

Wei Li, Can Gao, Guocheng Niu, Xinyan Xiao, Hao Liu, Jiachen Liu, Hua Wu, and Haifeng Wang. 2021b. UNIMO: Towards unified-modal understanding and generation via cross-modal contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2592–2607, Online. Association for Computational Linguistics.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020c. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*.

Junyang Lin, Rui Men, An Yang, Chan Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, J. Zhang, Jianwei Zhang, Xu Zou, Zhikang Li, Xiao Qing Deng, Jie Liu, Jinbao Xue, Huiling Zhou, Jianxin Ma, Jin Yu, Yong Li, Wei Lin, Jingren Zhou, J ie Tang, and Hongxia Yang. 2021. M6: A chinese multimodal pretrainer. *ArXiv*, abs/2103.00823.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2018. Neural baby talk. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7219–7228.

Chih-Yao Ma, Yannis Kalantidis, Ghassan AlRegib, Peter Vajda, Marcus Rohrbach, and Zsolt Kira. 2020. Learning to generate grounded image captions without localization supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.

Feng Nie, Jin-Ge Yao, Jinpeng Wang, Rong Pan, and Chin-Yew Lin. 2019. A simple recipe towards reducing hallucination in neural surface realisation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2673–2679, Florence, Italy. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*.

John Pavlopoulos, Vasiliki Kougia, and Ion Androutsopoulos. 2019. A survey on biomedical image captioning. In *Proceedings of the Second Workshop on Shortcomings in Vision and Language*, pages 26–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1179–1195.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2021. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. 2018. Object hallucination in image captioning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4035–4045.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*.

Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. 2022. How much can CLIP benefit vision-and-language tasks? In *International Conference on Learning Representations*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. *EMNLP*.

Gunnar A. Sigurdsson, Jean-Baptiste Alayrac, Aida Nematzadeh, Lucas Smaira, Mateusz Malinowski, João Carreira, Phil Blunsom, and Andrew Zisserman. 2020. Visual grounding in video for unsupervised word translation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10847–10856.

Dan Su, Xiaoguang Li, Jindi Zhang, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. 2022. Read before generate! faithful long form question answering with machine reading. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 744–756.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Chen Sun, Austin Myers, Carl Vondrick, Kevin P. Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7463–7472.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive multiview coding. In *ECCV*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575.

Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022a. OFA: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23318–23340. PMLR.

Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. 2022b. SimVLM: Simple visual language model pretraining with weak supervision. In *International Conference on Learning Representations*.

Yijun Xiao and William Yang Wang. 2021. On hallucination and predictive uncertainty in conditional language generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2734–2744.

Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995.

Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. 2022. FILIP: Fine-grained interactive language-image pre-training. In *International Conference on Learning Representations*.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021a. Ernie-vil: Knowledge enhanced vision-language representations through scene graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(4):3208–3216.

Tiezheng Yu, Wenliang Dai, Zihan Liu, and Pascale Fung. 2021b. Vision guided generative pre-trained language models for multimodal abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3995–4007, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. 2022. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18123–18133.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021a. Vinvl: Revisiting visual representations in vision-language models. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5575–5584.

Wenqiao Zhang, Haochen Shi, Siliang Tang, Jun Xiao, Qiang Yu, and Yueting Zhuang. 2021b. Consensus graph representation learning for better grounded image captioning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(4):3394–3402.

Wenqiao Zhang, Xin Eric Wang, Siliang Tang, Haizhou Shi, Haochen Shi, Jun Xiao, Yueting Zhuang, and William Yang Wang. 2020. Relational graph learning for grounded video description generation. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 3807–3828, New York, NY, USA. Association for Computing Machinery.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J. Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and vqa. *ArXiv*, abs/1909.11059.

Mingyang Zhou, Luowei Zhou, Shuohang Wang, Yu Cheng, Linjie Li, Zhou Yu, and Jingjing Liu. 2021. Uc2: Universal cross-lingual cross-modal vision-and-language pre-training. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4153–4163.

A Implementation Details

Our experiments are implemented in the PyTorch framework (Paszke et al., 2019). For both pre-training and finetuning, we use 8 Nvidia V100 GPUs. As mentioned in Section 5.1, we use the official CLIP checkpoints provided on GitHub. For the text decoder BertLM, we initialize model weights from the bert-base-uncased checkpoint with 110M parameters. For the finetuning on COCO Caption, we use a batch size of 512 and train the models with the AdamW optimizer (Loshchilov and Hutter, 2019) for 10 epochs with a learning rate of 5×10^{-5} and a weight decay of 1×10^{-2}. The learning rate is decayed linearly after each epoch with a rate of 0.85. For the pre-training of text generation losses (LM and ObjMLM), we keep the same hyper-parameters with a learning rate warmup within the first epoch. For ITC and ITM losses, we increase the batch size to 1024 as they tend to have a better performance with more negative samples.

B Additional Case Studies

Ground Truth: “A drawstring backpack has a green camouflage print.”

BLIPlarge: “A backpack with a camouflage pattern on it.”

RN50x64: “A backpack that is sitting on the ground.”

VinVLbase: “A helmet sitting on top of a bag.”

VinVLbase w/ SCST: “A bag with a black helmet on top of it.”

ViT-L/14 w/ LM & ObjMLM: “A backpack that is sitting on a white surface.”

ViT-L/14 w/ VG: “A backpack that is sitting on a bed.”

Ground Truth: “Kitchen cabinets with wood and blue painted doors and shelves.”

BLIPlarge: “A kitchen with wooden cabinets and blue cabinets.”

RN50x64: “A blue cabinet in a kitchen next to a sink.”

VinVLbase: “A blue cabinet in a kitchen next to a sink.”

VinVLbase w/ SCST: “A wooden cupboard with blue cabinetry and bottles in it.”

ViT-L/14 w/ LM & ObjMLM: “A kitchen with blue walls and wooden cabinets.”

ViT-L/14 w/ VG: “A kitchen with a blue cabinet and a white refrigerator.”

Ground Truth: “‘Red cocktails with floating cut strawberries sit on a cloth.”

BLIPlarge: “Three glasses of red liquid with strawberries in them.”

RN50x64: “A glass of red wine on a table.”

VinVLbase: “A close up of different cocktails in juice.”

VinVLbase w/ SCST: “A group of red juice in cocktail glasses on a table.”

ViT-L/14 w/ LM & ObjMLM: “A close up of glasses of wine on a table.”

ViT-L/14 w/ VG: “A close up of some glasses of liquid on a table.”

Ground Truth: “A scoreboard in a stadium displaying times for a race.”

BLIPlarge: “A stadium with a large screen displaying a race.”

RN50x64: “A group of people standing on top of a field.”

VinVLbase: “A billboard with a score-board in the background.”

VinVLbase w/ SCST: “A couple of scoreboards with billboards on a building.”

ViT-L/14 w/ LM & ObjMLM: “A scoreboard showing the score of a race.”

ViT-L/14 w/ VG: “a couple of televisions that are on a wall.”

Ground Truth: “A musical accordion has a leather strap on it.”

BLIPlarge: “A close up of an accordion in a case.”

RN50x64: “A close up of a guitar case on the ground.”

VinVLbase: “An accordion sitting on top of a wooden bench.”

VinVLbase w/ SCST: “An accordion sitting on top of a wooden bench.”

ViT-L/14 w/ LM & ObjMLM: “A close up of a black and white accordion.”

ViT-L/14 w/ VG: “A close up of a musical instrument on a table.”

Ground Truth: “‘A small muffin with some bright red spread on top.”.

BLIPlarge: “A close up of a muffin on a table.”

RN50x64: “A close up of a doughnut on a plate.”

VinVLbase: “A close up of a dessert on a plate.”

VinVLbase w/ SCST: “A group of desserts on a plate on a table.”

ViT-L/14 w/ LM & ObjMLM: “A cup of coffee with a cranberry sauce on it.”

ViT-L/14 w/ VG: “A pastry on a table.”

Figure 5: More cases of generated captions from different models, where the hallucinated objects are marked in red.

Characterizing the Entities in Harmful Memes: Who is the Hero, the Villain, the Victim?

Shivam Sharma[1,4*], **Atharva Kulkarni**[2*], **Tharun Suresh**[3], **Himanshi Mathur**[3],
Preslav Nakov[5], **Md. Shad Akhtar**[3] **and Tanmoy Chakraborty**[1]
[1]Indian Institute of Technology Delhi, India • [2]Carnegie Mellon University, USA
[3]Indraprastha Institute of Information Technology Delhi, India • [4]Wipro AI Labs, India
[5]Mohamed bin Zayed University of Artifcial Intelligence, UAE
`{shivam.sharma, tanchak}@ee.iitd.ac.in`, `atharvak@cs.cmu.edu`, `preslav.nakov@mbzuai.ac.ae`,
`{tharun20119, himanshi18037, shad.akhtar}@iiitd.ac.in`

Abstract

Memes can sway people's opinions over social media as they combine visual and textual information in an easy-to-consume manner. Since memes instantly turn viral, it becomes crucial to infer their intent and potentially associated harmfulness to take timely measures as needed. A common problem associated with meme comprehension lies in detecting the entities referenced and characterizing the role of each of these entities. Here, we aim to understand whether the meme glorifies, vilifies, or victimizes each entity it refers to. To this end, we address the task of *role identification of entities in harmful memes*, i.e., detecting who is the 'hero', the 'villain', and the 'victim' in the meme, if any. We utilize `HVVMemes` – a memes dataset on US Politics and Covid-19 memes, released recently as part of the CONSTRAINT@ACL-2022 shared-task. It contains memes, entities referenced, and their associated roles: hero, villain, victim, and other. We further design `VECTOR` (Visual-semantic role dEteCToR), a robust multi-modal framework for the task, which integrates entity-based contextual information in the multi-modal representation and compare it to several standard unimodal (text-only or image-only) or multi-modal (image+text) models. Our experimental results show that our proposed model achieves an improvement of 4% over the best baseline and 1% over the best competing stand-alone submission from the shared-task. Besides divulging an extensive experimental setup with comparative analyses, we finally highlight the challenges encountered in addressing the complex task of semantic role labeling within memes.

1 Introduction

Due to their pervasive nature, online social media platforms have emerged as a conducive medium for information exchange. Unfortunately, their democratic nature has fostered unabated dissemination of hate speech (MacAvaney et al., 2019), misinformation (Wu et al., 2019), fake news (Aldwairi and Alwahedi, 2018), propaganda (Da San Martino et al., 2020), and other harmful content. Such information manifests itself in various ways, and more recently, in the form of *memes*. Though typically intended to burlesque and lampoon world events, political outlook, or daily routine, an ostensibly innocuous meme can readily become a multi-modal cause of distress with a dexterous blend of images and texts. Due to their viral nature and ability to circumvent censorship (Mina, 2014), social media instigators and hatemongers are increasingly using memes as a powerful medium for disseminating spiteful content. Therefore, investigating the dark side of memes has risen as a pertinent research problem both in industry and academia (Sharma et al., 2020a; Pramanick et al., 2021a).

Figure 1: Examples of `heroes`, `villains` and `victims`, as portrayed within memes.

Motivation. While there have been many studies that have analyzed memes through the lens of emotions (Sharma et al., 2020a), sarcasm (Kumar and Garg, 2019), hate speech (Zhou et al., 2021; Kiela et al., 2020), misinformation (Zidani and Moran, 2021), offensiveness (Suryawanshi et al., 2020),

---

[*] Equal contribution

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2149–2163
May 2-6, 2023

and harmfulness (Pramanick et al., 2021a,b), there has been much less focus on analyzing semantic roles present within memes. Understanding these roles is critical for comprehending memes and narrative framing of various social entities. Caricaturing these entities with nefarious motives can lead to misinformation propagation, social calumny, and hatred towards minority communities. In addition to the dark portrayals, memes sometimes depict the sorrowful state of certain entities, illustrate their heroism, etc. Consider the memes in Fig. 1. The meme in Fig. 1 (a) portrays Jill Stein and the Green Party as *heroes* for their feminist views. Fig. 1 (b) draws a comparison between Adolf Hitler and Donald Trump, thus portraying the latter as a *villain* for his anti-immigrant views. Fig. 1 (c), on the other hand, depicts the plight of Ola and Uber drivers, who are out of work and *victims* of the lock-downs due to the COVID-19 pandemic. Thus, through depictions of heroism, villainy, and victimization, memes act as an alluring means to spread entity-relevant information and opinions.

Challenges. Despite growing interest in analyzing memes, identifying the underlying connotations for the entities framed therein remains a challenging task (Sharma et al., 2022b). Memes are obscure due to their highly cryptic semantics, and satirical content (Sabat et al., 2019). Moreover, categorizing the entities as a hero, a villain, or a victim requires real-world, contextual, temporal, spatial, and commonsense knowledge, which makes the task highly complex and subjective even for humans. Therefore, off-the-shelf multi-modal models that stand out well on conventional visual-linguistic tasks often flounder for memes as they are presumably inept to comprehend and capture the veiled information and multi-modal nuances present in a meme (Kiela et al., 2020).

Our Contributions. In this work, we propose a powerful approach to tackle the novel task of identifying the roles (the hero, the villain, and the victim) of the entities present in memes.

We model the problem as a role identification task and report the results for several unimodal and multi-modal baselines to benchmark the task (and the dataset) and assess its feasibility. We then proffer `VECTOR` a vision and commonsense enriched version of DeBERTa (He et al., 2020) for the task at hand. As meme text often contains satirical content, which tends to contradict the meme image, it is necessary to consider mutual information from both modalities. Also, meme content is often stated in a non-obvious way, necessitating commonsense and world knowledge. Thus, our `VECTOR` attempts to infuse the relevant visual and commonsense knowledge with linguistic representations. Utilizing ConceptNet (Speer et al., 2017), we generate an entity-relevant knowledge graph to represent commonsense knowledge relevant to the meme. Moreover, we employ a distinct multi-modal information fusion strategy based on Optimal Transport. We appropriate Optimal Transport-based Kernel Embedding (OTKE) (Mialon et al., 2021) for cross-modal correspondence. This technique by Mialon et al. (2021) marries the concepts of optimal transport theory with kernel techniques to provide robust and adaptable cross-modal adaptation. Our qualitative analysis underscores the importance of vision and commonsense knowledge integration, as `VECTOR` outperforms several competitive baselines.

Our contributions are summarized as follows:[1,2]

1. **Bench-marking `HVVMemes`:** We bench-mark the `HVVMemes` via ten baselines with various unimodal and multi-modal systems.
2. **Multi-modal system for identifying the hero, the villain, and the victim:** We develop `VECTOR` (Visual-semantic role dEteCToR), a knowledge enriched multi-modal system that integrates entity-based knowledge in the multi-modal representations.
3. **Extensive evaluation:** We report sizeable gains as part of our examination of `VECTOR` against state-of-the-art models and shared-task submissions.
4. **Detailed Analysis:** Along with the ablation investigations, we provide detailed qualitative and quantitative analysis.

2 Related Work

Online Harmfulness. Due to the exponential rise of harmful content on various social media platforms, the research community has piqued its curiosity toward related studies. Some of them are based on online trolling (Ortiz, 2020; Cook et al., 2018), cyber-bullying (Kim et al., 2021; Kowalski et al., 2014), cyber-stalking (Abu-Ulbeh et al., 2021), and hate speech (MacAvaney et al., 2019; Zhang and Luo, 2018). Other studies character-

---

[1] The source code is available at `https://github.com/LCS2-IIITD/VECTOR-Visual-semantic-role-dEteCToR`.

[2] The dataset can be downloaded from the official shared-task page: `https://codalab.lisn.upsaclay.fr/competitions/906`.

ize the correlation of racial and ethnic discrimination in the online and offline world (Relia et al., 2019). Cheng et al. (2017) examined the psycho-sociological outlook of online users toward online trolling behavior analysis. Few noteworthy investigations include characterizing homophily for self-harm due to eating disorders (Chancellor et al., 2016; Wang et al., 2017) using logistic regression and snow-ball sampling, and suicide-ideation (Burnap et al., 2015; Cao et al., 2019) via linguistic, structural, affective and socio-psychological features. For a significant period, most of these studies have been dominated by text-oriented investigation while obscuring knowledge about other modalities.

Characterising Online Targets. Another research direction focuses on aspects such as relevance, stance, hate speech, sarcasm, and dialogue acts within hateful exchanges on Twitter in conventional and multi-task settings (Zainuddin et al., 2017; Gautam et al., 2020; Ousidhoum et al., 2019). Zainuddin et al. (2018) addressed it by proposing neural networks with word embeddings. In contrast, the aspect-based sentiment was studied while addressing data sparsity, classification accuracy, and sarcastic content identification (Zainuddin et al., 2019). Shvets et al. (2021) demonstrated the efficacy of a generic concept extraction module for detecting the targets of hate speech. A few other studies on characterizing targets in harmful communication (Sap et al., 2020; Mathew et al., 2021) addressed social bias and hate speech explainability for targeted protected categories. Ma et al. (2018) used a hierarchical stack bidirectional gated recurrent units to detect targets and associated sentiments. A similar objective was studied in (Mitchell et al., 2013) but was formulated as sequence tagging in low-resource settings. Silva et al. (2016) used sentence structure to capture hate speech targets on social media to address detection and prevention. Most of these studies either focused on one primary designated target or emphasized detecting the association of sentiment while ignoring the affective spectrum. As observed in the literature (Shvets et al., 2021), such approaches may not generalize well across domains.

Studies on Memes. A significant influx of memes from online fringe communities, such as Gab, Reddit, and 4chan, to mainstream platforms, such as Twitter and Instagram, resulted in a massive epidemic of intended harm (Zannettou et al., 2018). Conventional visual features alongwith multimodal associativity was explored towards detecting memes in (Sharma. and Pulabaigari., 2020; Sharma et al., 2020b). Several datasets capturing offensiveness (Suryawanshi et al., 2020), hatefulness (Kiela et al., 2020; Gomez et al., 2020), and harmfulness (Pramanick et al., 2021b), have been curated. Detecting memetic harmfulness and targeted categories are discussed in (Pramanick et al., 2021b). Commonsense knowledge (Shang et al., 2021), web entities, racial cues (Pramanick et al., 2021b; Karkkainen and Joo, 2021), and other external cues have also been explored for detecting offense, harm, and hate speech in memes. Participatory events like the Facebook Hateful Meme Challenge (Kiela et al., 2020) have laid a strong foundation for community-level initiatives for detecting hate speech in memes. As part of this challenge, several interesting approaches utilising meta information, attentive interactions, and adaptive loss are attempted in the multimodal setting (Das et al., 2020; Sandulescu, 2020; Zhou et al., 2020; Lippe et al., 2020). Most of these efforts either address the detection tasks at various levels for harmfulness; see a recent survey (Sharma et al., 2022a) or design *ensemble* techniques lacking cost-optimality. However, as per our knowledge, no stand-alone approach reliably addresses the fine-grained task of understanding the roles of specific entities referred to within memes. We intend to address these aspects by seeking a robust and generalizable multimodal framework.

Domain	**# Memes**	**# Entity References**				
		Hero	Villain	Victim	Other	Total
COVID-19	3381	200	747	407	3065	4419
US Politics	3552	288	1641	544	3242	5715

Table 1: Summary of statistics for # memes and entities referenced within them in `HVVMemes` (Sharma et al., 2022b). Original train/val/test split ratio of 80:10:10 (%) for memes was preserved.

3 Dataset

We employ `HVVMemes`, a dataset released as part of CONSTRAINT@ACL-2022. It contains English memes on two topics: $3,552$ memes about COVID-19 (C) and $3,381$ memes related to US Politics (P). The dataset primarily captures connotative roles: hero, villain, and victim, for different entities referenced within memes. Table 1 shows a summary

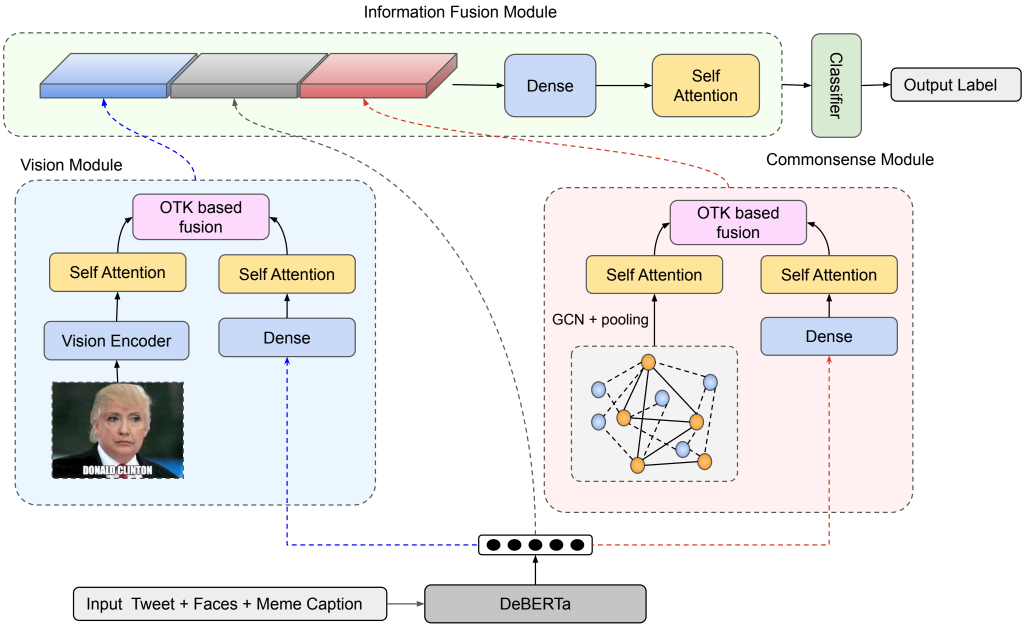


Figure 2: Schematic diagram of `VECTOR`. The *vision module* (left) consists of a ViT based image encoder and *commonsense module* (right) leverages GCN-based commonsense knowledge graph embedding. *Information fusion* module (top) fuses corresponding outputs toward final classification. OTK: Optimal Transport-based Kernel.

of `HVVMemes`.[3] In general, most of the entities referenced in the memes do not have any connotation associated (C: $3,065$, P: $3,242$) and are categorised as *other*. Amongst the key categories under consideration, *villain* has the most candidate references (C: 747, P: $1,641$), followed by *victim* (C: 407, P: 544), and finally *hero* (C: 200, P: 288), within a total of $3,381$ and $3,552$ memes for COVID-19 and US Politics, respectively. This highlights the realistic trends on social media.

4 Proposed Approach

This section outlines our proposed model `VECTOR` (Visual-semantic role dEteCToR) and its varied components. As previously noted, role detection for memetic entities is challenging and requires real-world, contextual, and commonsense knowledge. Thus, we propose a neuro-symbolic approach that integrates commonsense-enriched modelling via graph (KG) structure into the language modelling-based architecture (Zhang et al., 2021). KG's can be considered as discrete symbolic knowledge, which we leverage along with multimodal neural modelling. As shown in Fig. 2, `VECTOR` houses two primary sub-modules. The Vision Module leverages cross-modal interaction between the visual-linguistic signals to grasp optimal contextual information. The Commonsense Module integrates commonsense cues through an entity-based knowledge graph. Lastly, the Information Fusion Module coalesces the information obtained via attention-based fusion. In the following subsections, we go over the specifics of each module.

Text Module: We use DeBERTa (He et al., 2020) as our backbone model as it gives the best results amongst the text-only baselines (see Table 2). Aside from the text encoded in the meme, additional verbal information can be gleaned from memes. Evidence by Blaier et al. (2021) suggests that utilizing meme captions improves hateful meme identification results. Furthermore, additional cues such as the person, the location, and the entities present in the meme are helpful for downstream tasks. Thus, we use such ancillary information along with the OCR text. For image captioning, we make use of the recently released OFA model (Wang et al., 2022). For face identification, we use the same technique as the one by Kun et al. (2022). The OCR text, the entity name, the image caption, and the identified face *labels* are concatenated and passed to the DeBERTa model. We take the final layer representation $Z \in \mathbb{R}^{l \times d}$ to fuse information from other sub-modules.

Vision Module: Meme contents often contain contradicting text and image pairs. Therefore, it

---

[3] For additional details, please refer to the shared-task paper (Sharma et al., 2022b).

is required to incorporate information from these modalities to understand memes. Instead of using the traditional cross-modal attention to facilitate interaction between the two modalities, we utilize an optimal transport-based kernel interaction (Mialon et al., 2021). To begin with, we use a Vision Transformer (Dosovitskiy et al., 2020) for generating the image representations $E_m \in \mathbb{R}^{l_m \times d_m}$. The text and the image representations undergo dimensionality reduction by non-linear transformation, followed by a self-attention layer (Vaswani et al., 2017) as given by equations 1. This spawns vectors $Z'_m, E'_m \in \mathbb{R}^{l \times d'}$. We concatenate these two vectors and pass them to the Optimal Transport-based Kernel Embedding (OTKE) layer to bring about cross-modal interaction. It transforms the feature vectors to a Reproducing Kernel Hilbert Space (RKHS) (Berlinet and Thomas-Agnan, 2004) followed by a weighted pooling scheme using weights determined by the transport plan between the set and a trainable reference. Such a fusion technique provides a theoretically grounded adaptive vector for the task. This yields vector $Z_m \in \mathbb{R}^{l \times d'}$, given by equation 2 below:

$$Z'_m = S\left(\frac{ZZ^T}{\sqrt{d}}\right)Z \quad E'_m = S\left(\frac{E_m E_m^T}{\sqrt{d_m}}\right)E_m \quad (1)$$

$$Z_m = OTKE([Z'_m : E'_m]) \quad (2)$$

Commonsense Module: Due to their cryptic nature, identifying the intent of a meme is challenging. Their satirical and non-obvious way of conveying a message often requires commonsense comprehension. Thus, in our commonsense module, we generate a commonsense knowledge graph based on the entities in the meme. Similarly to (Shang et al., 2021), we extract all the nouns and noun phrases in the meme OCR and the meme caption. We extract commonsense relation pairs having confidence > 2 from ConceptNet (Speer et al., 2017) for all the noun chunks from memes. Noun chunks, commonsense entities, and the meme entity in question form the nodes of the commonsense graph. Each noun chunk from the OCR text is connected. The same applies to the noun chunks in the meme caption. A special aggregator token connects OCR and caption-based nodes. Thus, for each entity $e_{ij} \in E_i = \{e_{i1}, e_{i2}, \dots, e_{in}\}$, we have a commonsense graph $G_{ij} = (V_{ij}, E_{ij})$, where V_{ij} and E_{ij} are the nodes and the edges of the graph.

The graph having edges between various entities, noun chunks, and nouns, being undirected, represents a generic "connectivity". Therefore, an edge between nodes A and B indicates that they have some association. By doing this, we wanted to capture the commonsense-based 'proximity' of different entities/concepts within the vectorized space for common sense concepts. V_{ij} represents a set node (or vertices) constituting a commonsense graph corresponding to each entity.

For each node in the commonsense graph, we generate an embedding of size d_g using the last layer representations from DeBERTa. In order to facilitate the interaction between the nodes, the commonsense graph goes through two rounds of graph convolutions (Kipf and Welling, 2017) followed by a max-pooling operation to spawn an aggregated graph embedding $E_g \in \mathbb{R}^{d'}$. Similarly to the vision module, the textual representations Z and the commonsense graph representation E_g undergo non-linear transformation and dimensionality reduction followed by self-attention (Vaswani et al., 2017). This generates contextual vectors $Z'_g, E'_g \in \mathbb{R}^{l \times d'}$, respectively. Finally, the commonsense knowledge is infused in the language representations using OKTE, generating the final vector $Z_g \in \mathbb{R}^{l \times d'}$.

Information Fusion Module: Each of the modules mentioned above integrates salient information in the language representations. The information fusion module aggregates the knowledge obtained from all other modules using attention-based mutual interaction (Vaswani et al., 2017). Concretely, we concatenate the vectors of Z, Z_m, and Z_g and pass them through a round of dimensionality reduction and non-linear transformations. Then, we use a self-attention mechanism so that the information obtained from each component interacts with one another. The final generated vector $Z'_c \in \mathbb{R}^{l \times d}$ is passed to a classifier with a softmax activation to predict the final labels.

5 Baselines

Unimodal Systems: We use a variety of text-based and image-based models as our baseline systems. Starting with the text baselines, we use **BERT** (Devlin et al., 2019), and variants thereof such as **DistilBERT** (Sanh et al., 2019), **RoBERTa** (Liu et al., 2019), **XLNet** (Yang et al., 2019), and **DeBERTa** (He et al., 2020). For the image-based baselines, we start with a representation based on **ResNet-50** (He et al., 2016), followed by **Vision Transformer (ViT)** (Dosovitskiy et al., 2020) and **SWIN** (Liu et al., 2021), which is hierarchical ver-

	Model Details	Hero			Villain			Victim			Other			Macro-F1			Acc.
		Prec	Rec	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	
Text-only	Dis. BERT	.154	.115	.132	.431	.400	.415	.292	.246	.267	.856	.881	.868	.433	.411	.420	.766
	BERT	.250	.115	.158	.484	.446	.464	.358	.254	.297	.864	.905	.884	.489	.430	.451	.791
	RoBERTa	.243	.173	.202	.473	.400	.433	.346	.386	.365	.866	.891	.879	.482	.463	.470	.782
	XLNet	.172	.096	.123	.491	.386	.432	.383	.272	.318	.852	.910	.880	.475	.416	.438	.788
	DeBERTa	.250	.250	.250	.469	**.591**	.523	.395	.395	.395	**.889**	.847	.868	.501	**.521**	.509	.776
	DeBERTa(l)	.268	.212	.237	**.604**	.440	.509	.543	**.500**	.521	.870	.922	**.895**	.518	.571	.540	**.818**
Vision-only	ResNet	0	0	0	.467	.303	.367	.667	.053	.097	.827	**.948**	.883	.490	.326	.337	.793
	ConvNeXT	0	0	0	.398	.329	.360	.333	.009	.017	.828	.924	.873	.390	.315	.313	.776
	ViT	0	0	0	.436	.380	.406	.250	.009	.016	.836	.926	.878	.380	.329	.325	.785
	BEiT	0	0	0	.413	.420	.416	.500	.009	.017	.838	.907	.871	.438	.334	.326	.775
	SWIN	0	0	0	.393	.326	.356	.231	.026	.047	.828	.919	.871	.363	.318	.319	.772
Multimodal	CLIP	0	0	0	.478	.311	.378	**.691**	.071	.104	.845	.951	.886	.393	.335	.342	.786
	ViLBERT	.078	.034	.045	.409	.513	.469	.398	.276	.321	.856	.843	.849	.428	.408	.421	.761
	V-BERT	.143	.077	.100	.466	.560	.508	.452	.333	.384	.886	.879	.882	.487	.462	.469	.790
	MMTrans.	0	0	0	.516	.477	.496	.447	.303	.361	.814	.878	.845	.437	.392	.405	.747
	MMBT	.103	.058	.074	.446	.537	.487	.414	.298	.347	.881	.872	.877	.458	.438	.447	.780
	VECTOR	**.444**	**.385**	**.412**	.553	.534	**.544**	.505	.456	**.479**	.883	.897	.890	**.568**	**.596**	**.581**	.813
	$\Delta_{(\text{VECTOR}-\text{V-BERT})}$	.301↑	.308↑	.312↑	.087↑	.026↓	.036↑	.053↑	.123↑	.095↑	.003↓	.018↑	.008↑	.081↑	.134↑	.112↑	.023↑

Table 2: Benchmarking results (0's indicate no correct prediction). $\Delta_{(\mathtt{X}-\mathtt{Y})}$: Performance difference between models X and Y, # ↑ and # ↓: Absolute increment and decrement respectively.

sion of ViT that uses shifted windows. We also include the recently proposed **ConvNeXT** (Liu et al., 2022) and **BEiT** (Bao et al., 2021), which uses self-supervised pre-training of Vision Transformers.

Multi-modal Systems: We use various variants of multi-modal pre-trained systems from the MMF Framework. **MMF Transformer** is a library Transformer model that uses visual and language tokens with self-attention. **MMBT:** multi-modal Bi-transformer (Kiela et al., 2019) captures the two modalities' intra-modal and inter-modal dynamics. **ViLBERT** is a vision and language BERT (Lu et al., 2019), a strong model with the task-agnostic joint representation of images and text. **ViLBERT CC** is pre-trained on conceptual captions (Sharma et al., 2018) based pretext task. **Visual BERT** (Li et al., 2019), also pre-trained using MS COCO (Lin et al., 2014), implicitly aligns the input text and regions in the input image using self-attention. CLIP (Radford et al., 2021) leverages image–text contrastive pretraining.

6 Experiments

Experimental Details We present benchmarking results in Table 2, comparison with the shared-task submissions (c.f. Table 4) averaged over five independent runs, while for the ablation study in Table 3, we compare the *best* check-points for different `VECTOR` component-wise evaluations to assess the performance bounds, on the *diversified* test set. We use precision, recall, and F1 for individual classes, and macro-averaged for the overall assessment.[4]

Comparative Analysis. ▷ **Unimodal Models:** Despite the evident glorification cues within the *Hero* references in memes, large pre-trained models are observed to depict limitations. This can be observed from Table 2, wherein significantly low performance is observed for strong models like DistilBERT, BERT, and XLNet, with F1 scores of 0.132, 0.158, and 0.123, respectively. On the other hand, RoBERTa enhances the class-specific F1 score by about 0.05 absolute points, which suggests the efficacy the exhaustive hyperparameter tuning can induce. Finally, DeBERTa-based unimodal systems yield the highest optimal F1 scores among all unimodal models evaluated. The scores observed are 0.250 and 0.237 for the base and the large variants, respectively.

For the *Villain* category, BERT can be observed (c.f. Table 2) to yield relatively better performance as against Hero detection, with an F1 score of 0.464. This is in contrast to the sub-par performances by DistilBERT, RoBERTa, and XLNet, yielding 0.415, 0.433, and 0.432 F1 scores, respectively. As with the Hero detection, DeBERTa-base and large models outperform the other models for both text and vision modalities.

Interestingly, none of the text-only models, except for DeBERTa large, could beat the F1 score of 0.395 for DeBERTa-base. The lower performance is primarily due to inadequate categorical representation within the dataset. Moreover, the inherent complexity in distinguishing villains from victims confuses the model. The modelling efficacy solicited for such a scenario is suggested by over 0.10 absolute point enhancement by DeBERTa-large, which has 4X more backbone parameters

---

[4] See Appendix A for additional experimental details.

Model Details	**HER**	**VIL**	**VIC**	**OTH**	**F1**	**Acc.**
Simple early-fusion (DeBERTa + ViT)	0.31	0.55	0.50	0.88	0.56	0.79
(a). + Meme (image) caption	0.32	0.52	0.48	0.89	0.55	0.80
(b). + Face labels	0.26	0.56	0.51	0.89	0.56	0.82
(c). [(a) + (b)] + Commonsense KG	0.36	0.54	0.45	0.89	0.56	0.81
(d). [(a) + (b) + (c)] + CAT	0.28	0.53	0.49	0.89	0.55	0.81
(e). [(a) + (b) + (c)] + OTK (`VECTOR`)	**0.38**	**0.57**	**0.53**	**0.90**	**0.60**	**0.83**

Table 3: Ablation study: Comparing `VECTOR` and its sub-modules over *best* model results. CAT: Fusion via concatenation, OTK: Fusion via Optimal Transport Kernel, KG: Knowledge Graph.

Rank	**System**	**Prec.**	**Rec.**	**F1**
-	`VECTOR`	0.568	0.596	0.581
1	Logically (Kun et al., 2022)	0.544	0.610	0.571
2	c1pher (Singh et al., 2022)	0.527	0.581	0.547
3	smontariol (Montariol et al., 2022)	0.580	0.450	0.485
4	zhouziming (Zhou et al., 2022)	0.480	0.450	0.462
5	IIITDWD (Fharook, 2022)	0.256	0.238	0.239
6	rabindra.nath (Nandi et al., 2022)	0.253	0.253	0.237

Table 4: Comparison to the results on the CONSTRAINT@ACL-2022 shared-task on `HVVMemes`.

than DeBERTa-base.

Other category, having the majority representation with over 6K unique references within memes, projects the highest F1 scores with an average of 0.879. Finally, the overall Macro-F1 scores reflect the category-wise trend observed, with DeBERTa large leading with an F1 score of 0.540. DeBERTa-base follows it with 0.509 and the rest with an average F1 score of 0.444.

For intuitive reasons, the visual modality, not indicative of complex role semantics, yields poor performance compared to text-only models. As a result, none of the image-only models (ConvNext, ViTBEiT, and SWIN) are observed to make any headway in correctly detecting a Hero reference in memes (c.f. Table 2). At the same time, except for the Villain category, a simple ResNet-based model outperforms, albeit with fine margins, the rest of the models within the categories Victim and Other, with F1 scores of 0.097 and 0.883, respectively. This highlights the efficacy of global image representations against tokenized (or patched) ones for factoring visual features, especially where there is not much visual-linguistic grounding to be leveraged. On average, the image-only models can project a paltry F1 score of 0.324.

▷ **Multi-modal Models.** Several state-of-the-art multimodal systems, on average, are observed to yield a Macro-F1 score of 0.416, which lies between the text-only (0.444) and the image-only (0.324) models. Besides the category-wise performance trend being similar to the one observed for unimodal models, multimodal systems like VisualBERT and MMBT yield the highest Macro-F1 scores of 0.468 and 0.447, respectively. This is likely due to the joint attentive modelling, multimodal pre-training using standard datasets like MS COCO, and fine-tuning adopted by these models. Other competitive models like CLIP, ViLBERT, and MM Transformer achieve Macro-F1 scores of up to 0.342, 0.421, and 0.405, respectively. This either suggests that the image component induces additional noise within the models or that the existing multimodal systems do not effectively capture the complex pragmatics that semantic role-labeling in memes solicits. The former is less likely, as intuitively, memetic visuals do provide minor yet decisive semantics toward holistic assimilation of the meme's message. Our proposed approach `VECTOR`, is observed to address the required cross-modal association by achieving impressive F1 scores across different roles and a 0.581 Macro-F1 score, which induces an enhancement of almost 4% and 12 % over DeBERTa large (unimodal best) and VisualBERT (multimodal best), respectively.

Ablations Results. Table 3 depicts ablation results, wherein the basic early-fusion setup involving DeBERTa and ViT performs well for *villain* category with 0.55 F1 score. Adding meme-image captions enhances the *hero* predictions and overall accuracy marginally by 1% each. Face labels, although, significantly enhance the prediction of *villain* and *victim*, suggesting lexical and semantic utility via *face labels*. The overall performance remains unchanged due to compromised *hero* predictions. Besides yielding balanced scores, adding a commonsense module elevates hero prediction distinctly to 0.36, effectively indicating its utility, especially for the under-represented role category. Finally, `VECTOR` with OTK-based embedding yields optimal cross-modal correspondence, as observed from the best performances across roles and metrics evaluated, showcasing the constituting feature's utility towards addressing the given task. Although *not* averaged over multiple runs, the *best* model check-point for `VECTOR` is observed to yield an impressive overall macro-F1 score of 0.60, which is the best score observed across experiments, suggesting an upper-bound for `VECTOR`'s performance.

Comparison to Previous Work. Table 4 showcases the best-performing systems from

	Original	Proposed	VisualBERT
Hero	–	–	–
Villain	[jeffrey epstein, donald trump]	[donald trump]	[jeffrey epstein, donald trump]
Victim	[13 year old girl]	[13 year old girl]	[13 year old girl]
Other	–	[jeffrey epstein]	–

Figure 3: Error analysis for VECTOR and VisualBERT.

CONSTRAINT@ACL-2022's shared-task. Since all of the shared-task submissions used ensemble techniques of multiple models, we present their best individual model results for a fair comparison with our proposed model: VECTOR. The macro-F1 score varies by 0.334 points across the participants, emphasizing model selection. We can draw parallels between Table 2 and Table 4 with Logically, c1pher, smontariol, and zhouziming using DeBERTa, RoBERTa, VisualBERT, and VisualBERT, respectively. They exhibit marginal improvements using loss-weighting techniques and additional classification layers. Our model VECTOR performs more consistently, especially with the class-wise scores of *Hero* (c.f. Table 2), establishing its efficacy across the category types and limited categorical data representation. Besides class-wise consistency, VECTOR outperforms other shared-task submissions as a stand-alone system. Also, despite being relatively complex, VECTOR produces consistent class-wise and better overall performance, suggesting marginal yet robust modelling capacity facilitated by VECTOR's vision + common-sense module's interaction with the textual signals.

Statistical Significance. We performed a bootstrapping significance test w.r.t the proposed model (VECTOR), and the previous best solution by team Logically (Kun et al., 2022) via random sampling with replacement strategy, with N=1000 over 1000 simulations, and observed a 'p-value' of 0.0410. This indicates a subtle yet encouraging confidence margin in the model predictions. This could be likely due to better predictions across the four categories, including 'hero', which most of the other models compared are empirically observed to struggle at. These aspects corroborate the semantic-role label-agnostic characteristics of the proposed model.

	Original	Proposed	VisualBERT
Hero	–	–	–
Villain	[barack obama]	[barack obama]	[barack obama, donald trump, daily wire]
Victim	[donald trump]	[donald trump]	–
Other	[daily wire]	[daily wire]	–

Figure 4: Interpretability analysis for VECTOR; proposed model and VisualBERT; best multimodal baseline.

Error Analysis. The example depicted in Fig. 3 insinuates 'donald trump' and 'jeffrey Epstein's as *villains* while *victimizes* a '13 year old girl'. Visually, there isn't much to consider towards adjudicating the former two entities as villains, except the expressions of 'jeffrey Epstein's exuding somewhat *sinister* looks. Whereas *vilifying* connotation is implied primarily by the embedded text. Now, although VECTOR predicts the roles of 'donald trump' and '13 year old girl' correctly as *villain* and *victim*, respectively, it fails to detect 'jeffrey epstein as a *villain* and categorizes it as an *other*. This example highlights the limitations of VECTOR in terms of its inherent modality-specific biases. A ViT-based image encoder, due to its disparate patch-wise processing, and self-attention across the input patches, leads to noisy visual attention. On the other hand, VisualBert-based predictions replete with pre-trained common-sense semantics are better positioned for this case to capture the required semantic indicators, a portion of *epstein's* facial expression in this case. Also, as the dataset is well-stocked with examples where 'Donald trump' is *vilified*, both models being compared assign the role of *villain* to 'donald trump'.

7 Interpretability

The predictive capacity of VECTOR can be assessed by comparing its interpretability with that of VisualBERT, the best-performing baseline. We investigate a reasonably complex example depicted in Fig 4, for which VECTOR accurately predicts all the entity roles, but VisualBERT wrongly predicts all of them, as *villains*. Attention map for VECTOR reveals that it primarily attends to the face of 'barack obama', while leveraging other contextual cues, likely via the common-sense and visual description-based signals, leading to a correct assignment of roles. On the other hand, VisualBERT, having a Faster R-CNN-based image encoder, primarily pretrained for the common object-detection task, fails to uniquely attend to different semantically referred entities within the meme image and additionally adds noisy features related to objects like a *tie*, without any additional contextual knowledge. This validates `VECTOR`'s superior discriminatory capacity and interpretability over VisualBERT.

8 Generalizability

Since, the test set consists of all unique memes, there is a higher chance that there are many entities as part of it that are not seen during the training stage. Table 2 attests to the `VECTOR`'s potential to succeed across different roles. This suggests that `VECTOR` can adapt to the domain-specific nuances offered by the variety and complexity associated with visual-semantic roles within memes. Through the results observed in this work, we attempt to highlight that existing *standard* approaches either have inconsistent performances across the roles: hero, villain, victim, and other, or yield low scores for particular role categories (c.f. Table 2), which corroborates their limitations. On the other hand, `VECTOR` yields a balanced performance and generalizes reasonably well for the least represented class in the dataset (*hero*).

9 Conclusion

This paper addresses a recently proposed task of identifying the roles of entities in harmful memes and discusses its challenges. We further presented numerous unimodal and multi-modal baselines to benchmark `HVVMemes`. Moreover, we proffer `VECTOR`, a contextual knowledge-enriched multimodal framework that bolsters the multi-modal representations with entity-based external knowledge using a cross-modal attention scheme. `VECTOR` shows noteworthy improvements over the baselines, thus justifying contextual knowledge inclusion. As for future investigations, we plan to conceive a more symbolic system with graph-based entity linking, commonsense knowledge, and visual concepts.

10 Limitations

As noted in the discussion dedicated to *Error Analysis*, several entities tend to dominate specific roles within the dataset due to the realistic representation of the harmful referencing in memes. This not only biases the model against their generalizability but also poses challenges towards modelling entity-independent role detection hypotheses for a diverse set of entities. This especially calls for building models regularized to address such biases and more participatory initiatives toward curating better and more large-scale datasets.

11 Ethics and Broader Impact

Reproducibility. We present detailed hyperparameter configurations in Appendix A.

User Privacy. The information depicted/used does not include any personal information. Copyright aspects are attributed to the dataset source.

Biases. As per the authors, any biases found in the dataset are unintentional (Sharma et al., 2022b), and by conducting the study on this dataset we do not intend to cause harm to any group or individual. We acknowledge that detecting harmfulness can be subjective, and thus it is inevitable that there would be biases in gold-labelled data or in the label distribution. This is addressed by the dataset curators by using general keywords about US Politics, and also by following a well-defined schema, which sets explicit definitions for annotation.

Misuse Potential. Our approach can be potentially used for ill-intended purposes, such as biased targeting of individuals/communities/organizations, etc. that may or may not be related to demographics and other information within the text. Intervention with human moderation would be required to ensure that this does not occur.

Intended Use. We make use of the existing dataset in our work in line with the intended usage prescribed by its creators and solely for research purposes. This applies in its entirety to its further usage as well. We do not claim any rights to the dataset used or any part thereof. We believe that it represents a useful resource when used appropriately.

Environmental Impact. Finally, large-scale models require a lot of computations, which contribute to global warming (Strubell et al., 2019). However, in our case, we do not train such models from scratch; rather, we fine-tune them on a relatively small dataset.

Acknowledgements

The work was supported by Wipro research grant.

References

Waheeb Abu-Ulbeh, Maryam Altalhi, Laith Abualigah, Abdulwahab Almazroi, Putra Sumari, and Amir Gandomi. 2021. Cyberstalking victimization model using criminological theory: A systematic literature review, taxonomies, applications, tools, and validations. *Electronics*, 10:1670.

Monther Aldwairi and Ali Alwahedi. 2018. Detecting fake news in social media networks. *Procedia Computer Science*, 141:215–222.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. 2021. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*.

Alain Berlinet and Christine Thomas-Agnan. 2004. *Reproducing Kernel Hilbert Space in Probability and Statistics*.

Efrat Blaier, Itzik Malkiel, and Lior Wolf. 2021. Caption enriched samples for improving hateful memes detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9350–9358, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pete Burnap, Walter Colombo, and Jonathan Scourfield. 2015. Machine classification and analysis of suicide-related communication on twitter. In *Proceedings of the 26th ACM Conference on Hypertext &; Social Media*, HT '15, page 75–84, New York, NY, USA.

Lei Cao, Huijun Zhang, Ling Feng, Zihan Wei, Xin Wang, Ningyun Li, and Xiaohao He. 2019. Latent suicide risk detection on microblog via suicide-oriented word embeddings and layered attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1718–1728, Hong Kong, China.

Stevie Chancellor, Zhiyuan (Jerry) Lin, and Munmun De Choudhury. 2016. "this post will just get taken down": Characterizing removed pro-eating disorder social media content. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, page 1157–1162, New York, NY, USA.

Justin Cheng, Michael Bernstein, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2017. Anyone can become a troll: Causes of trolling behavior in online discussions. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, page 1217–1230, New York, NY, USA.

Christine Cook, Juliette Schaafsma, and Marjolijn Antheunis. 2018. Under the bridge: An in-depth examination of online trolling in the gaming context. *New Media & Society*, 20(9):3323–3340.

Giovanni Da San Martino, Alberto Barrón-Cedeño, Henning Wachsmuth, Rostislav Petrov, and Preslav Nakov. 2020. SemEval-2020 task 11: Detection of propaganda techniques in news articles. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1377–1414, Barcelona (online).

Abhishek Das, Japsimar Singh Wahi, and Siyao Li. 2020. Detecting hate speech in multi-modal memes.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Shaik Fharook. 2022. Are you a hero or a villain? a semantic role labelling approach for detecting harmful memes. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, CONSTRAINT '22, Dublin, Ireland.

Akash Gautam, Puneet Mathur, Rakesh Gosangi, Debanjan Mahata, Ramit Sawhney, and Rajiv Ratn Shah. 2020. #MeTooMA: Multi-aspect annotations of tweets related to the MeToo movement. *Proceedings of the International AAAI Conference on Web and Social Media*, 14(1):209–216.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision*, WACV '20, pages 1459–1467.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Kimmo Karkkainen and Jungseock Joo. 2021. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, WACV '21, pages 1548–1558.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, Ethan Perez, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. In *Proceedings of the NeurIPS Workshop on Visually Grounded Interaction and Language*, ViGIL '19, Vancouver, Canada.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, volume 33 of *NeurIPS '20*.

Seunghyun Kim, Afsaneh Razi, Gianluca Stringhini, Pamela J. Wisniewski, and Munmun De Choudhury. 2021. A human-centered systematic literature review of cyberbullying detection algorithms. *Proceedings ACM Hum. Comput. Interact.*, 5(CSCW2):1–34.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, ICLR '15, San Diego, California, USA.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Robin Kowalski, Gary Giumetti, Amber Schroeder, and Micah Lattanner. 2014. Bullying in the digital age: A critical review and meta-analysis of cyberbullying research among youth. *Psychological bulletin*, 140.

Akshi Kumar and Geetanjali Garg. 2019. Sarc-M: Sarcasm detection in typo-graphic memes. In *Proceedings of the International Conference on Advances in Engineering Science Management & Technology*, ICAESMT '19, Dehradun, India.

Ludovic Kun, Jayesh Bankoti, and David Kiskovski. 2022. Logically at the constraint 2022: Multimodal role labelling. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, pages 24–34, Dublin, Ireland. Association for Computational Linguistics.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. VisualBERT: A simple and performant baseline for vision and language. *arXiv:1908.03557*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, ECCV '14, pages 740–755, Zurich, Switzerland.

Phillip Lippe, Nithin Holla, Shantanu Chandra, Santhosh Rajamanickam, Georgios Antoniou, Ekaterina Shutova, and Helen Yannakoudakis. 2020. A multimodal framework for the detection of hateful memes. *arXiv:2012.12871*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Proceedings of the Conference on Neural Information Processing Systems*, NeurIPS '19, pages 13–23, Vancouver, Canada.

Dehong Ma, Sujian Li, and Houfeng Wang. 2018. Joint learning for targeted sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4737–4742, Brussels, Belgium.

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. 2019. Hate speech detection: Challenges and solutions. *PLOS ONE*, 14(8):1–16.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. Hatexplain: A benchmark dataset for explainable hate speech detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17):14867–14875.

Grégoire Mialon, Dexiong Chen, Alexandre d'Aspremont, and Julien Mairal. 2021. A trainable optimal transport embedding for feature aggregation and its relationship to attention. In *International Conference on Learning Representations*.

An Xiao Mina. 2014. Batman, Pandaman and the Blind Man: A case study in social change memes and internet censorship in China. *Journal of Visual Culture*, 13(3):359–375.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654, Seattle, Washington, USA.

Syrielle Montariol, Étienne Simon, Arij Riabi, and Djamé Seddah. 2022. Fine-tuning and sampling strategies for multimodal role labeling of entities under class imbalance. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, CONSTRAINT '22, Dublin, Ireland.

Rabindra Nath Nandi, Firoj Alam, and Preslav Nakov. 2022. Detecting the role of an entity in harmful memes: Techniques and their limitations. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, CONSTRAINT '22, Dublin, Ireland.

Stephanie M. Ortiz. 2020. Trolling as a collective form of harassment: An inductive study of how online users understand trolling. *Social Media + Society*, 6(2):2056305120928512.

Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. 2019. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4675–4684, Hong Kong, China.

Shraman Pramanick, Dimitar Dimitrov, Rituparna Mukherjee, Shivam Sharma, Md. Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021a. Detecting harmful memes and their targets. In *Findings of ACL*, ACL-IJCNLP '21, pages 2783–2796.

Shraman Pramanick, Shivam Sharma, Dimitar Dimitrov, Md. Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021b. MOMENTA: A multimodal framework for detecting harmful memes and their targets. In *Findings of EMNLP 2021*, pages 4439–4455, Punta Cana, Dominican Republic.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763.

Kunal Relia, Zhengyi Li, Stephanie H. Cook, and Rumi Chunara. 2019. Race, ethnicity and national origin-based discrimination in social media and hate crimes across 100 u.s. cities. *Proceedings of the International AAAI Conference on Web and Social Media*, 13(01):417–427.

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. *arXiv:1910.02334*.

Vlad Sandulescu. 2020. Detecting hateful memes using a multimodal deep ensemble. *arXiv:2012.13235*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, ACL '20, pages 5477–5490, Online.

Lanyu Shang, Christina Youn, Yuheng Zha, Yang Zhang, and Dong Wang. 2021. KnowMeme: A knowledge-enriched graph neural network solution to offensive meme detection. In *Proceedings of the 2021 IEEE 17th International Conference on eScience*, eScience '21, pages 186–195.

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Björn Gambäck. 2020a. SemEval-2020 task 8: Memotion analysis- the visuo-lingual metaphor! In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, SemEval '20, pages 759–773.

Chhavi Sharma. and Viswanath Pulabaigari. 2020. A curious case of meme detection: An investigative study. In *WEBIST*, pages 327–338.

Chhavi Sharma, Viswanath Pulabaigari, and Amitava Das. 2020b. Meme vs. non-meme classification using visuo-linguistic association. pages 353–360.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, ACL '18, pages 2556–2565, Melbourne, Australia.

Shivam Sharma, Firoj Alam, Md. Shad Akhtar, Dimitar Dimitrov, Giovanni Da San Martino, Hamed Firooz, Alon Halevy, Fabrizio Silvestri, Preslav Nakov, and Tanmoy Chakraborty. 2022a. Detecting and understanding harmful memes: A survey. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence*, IJCAI-ECAI '22, Vienna, Austria.

Shivam Sharma, Tharun Suresh, Atharva Kulkarni, Himanshi Mathur, Preslav Nakov, Md Shad Akhtar, and Tanmoy Chakraborty. 2022b. Findings of the constraint 2022 shared task on detecting the hero, the villain, and the victim in memes. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, pages 1–11.

Alexander Shvets, Paula Fortuna, Juan Soler, and Leo Wanner. 2021. Targets and aspects in social media hate speech. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 179–190, Online.

Leandro Silva, Mainack Mondal, Denzil Correa, Fabrício Benevenuto, and Ingmar Weber. 2016. Analyzing the targets of hate in online social media. *Proceedings of the International AAAI Conference on Web and Social Media*, 10(1):687–690.

Pranaydeep Singh, Aaron Maladry, and Els Lefever. 2022. Combining language models and linguistic information to label entities in memes. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, CONSTRAINT '22, Dublin, Ireland.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, ACL '19, pages 3645–3650, Florence, Italy.

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (MultiOFF) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 32–41, Marseille, France.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *CoRR*, abs/2202.03052.

Tao Wang, Markus Brede, Antonella Ianni, and Emmanouil Mentzakis. 2017. Detecting and characterizing eating-disorder communities on social media. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, page 91–100, New York, NY, USA.

Liang Wu, Fred Morstatter, Kathleen M. Carley, and Huan Liu. 2019. Misinformation in social media: Definition, manipulation, and detection. *SIGKDD Explor. Newsl.*, 21(2):80–90.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Nurulhuda Zainuddin, Ali Selamat, and Roliana Ibrahim. 2017. Twitter hate aspect extraction using association analysis and dictionary-based approach. In *New Trends in Intelligent Software Methodologies, Tools and Techniques - Proceedings of the 16th International Conference (SoMeT'17)*, volume 297 of *Frontiers in Artificial Intelligence and Applications*, pages 641–651, Kitakyushu City, Japan.

Nurulhuda Zainuddin, Ali Selamat, and Roliana Ibrahim. 2018. Evaluating aspect-based sentiment classification on Twitter hate speech using neural networks and word embedding features. In *New Trends in Intelligent Software Methodologies, Tools and Techniques*, pages 723–734.

Nurulhuda Zainuddin, Ali Selamat, and Roliana Ibrahim. 2019. Hate crime on Twitter: Aspect-based sentiment analysis approach. In *Advancing Technology Industrialization Through Intelligent Software Methodologies, Tools and Techniques*, pages 284–297.

Savvas Zannettou, Tristan Caulfield, Jeremy Blackburn, Emiliano De Cristofaro, Michael Sirivianos, Gianluca Stringhini, and Guillermo Suarez-Tangil. 2018. On the origins of memes by means of fringe web communities. In *Proceedings of the Internet Measurement Conference 2018*, IMC '18, page 188–202, New York, NY, USA.

Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. 2021. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35.

Z. Zhang and L. Luo. 2018. Hate speech detection: a solved problem? the challenging case of long tail on twitter. *Semantic Web*.

Xinyi Zhou, Jindi Wu, and Reza Zafarani. 2020. SAFE: Similarity-aware multi-modal fake news detection. In *Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, pages 354–367. Springer.

Yi Zhou, Zhenhao Chen, and Huiyuan Yang. 2021. Multimodal learning for hateful memes detection. In *Proceedings of the International Conference on Multimedia Expo Workshops*, ICMEW '21, pages 1–6.

Ziming Zhou, Han Zhao, Jingjing Dong, Jun Gao, and Xiaolong Liu. 2022. DD-TIG at Constraint@ACL2022: Multimodal understanding and reasoning for role labeling of entities in hateful memes. In *Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations*, CONSTRAINT '22, Dublin, Ireland.

Sulafa Zidani and Rachel Moran. 2021. Memes and the spread of misinformation: Establishing the importance of media literacy in the era of information disorder. *Teaching Media Quarterly*, 9(1).

Modality	Model	BS	#Epochs	LR	V-Enc	T-Enc	#Param
UM	BERT	8	15	1e-5	-	`bert`	25M
	DistilBERT	8	15	1e-5	-	`distilbert-base`	66M
	XLNet	8	15	1e-5	-	`xlnet`	116M
	RoBERTa	8	15	1e-5	-	`roberta-base`	123M
	DeBERTa	8	15	1e-5	-	`deberta-base`	86M
	DeBERTa-Large	8	15	1e-5	-	`deberta-large`	304M
	ResNet	8	15	1e-5	`resnet`	-	25M
	ConvNeXT	8	15	1e-5	`convnet`	-	50M
	ViT	8	15	1e-5	`vit`	-	86M
	SWIN	8	15	1e-5	`swin`	-	88M
	BEiT	8	15	1e-51	`beit`	-	71M
MM	MMFT	16	20	0.001	ResNet-152	`bert`	170M
	CLIP	16	20	0.0001	ViT	`clip`	149M
	MMBT	16	20	0.0001	ResNet-152	`bert`	169M
	ViLBERT*	16	10	0.0001	Faster RCNN	`bert`	112M
	V-BERT*	16	10	0.0001	Faster RCNN	`bert`	247M
	`VECTOR`	8	15	1e-5	`vit`	`deberta-large`	123M

Table 5: Hyperparameters summary. [BS→Batch Size; LR→Learning Rate; V/T-Enc→Vision/Text-Encoder; `vit`→`vit-base-patch16-224-in21k`; `bert`:→`bert-base-uncased`; `xlnet`→`xlnet-base-uncased`; `resnet`→`resnet50`].

A Implementation Details and Hyperparameter Values

We train all the models using PyTorch on an actively dedicated NVIDIA Tesla V100 GPU, with 32 GB dedicated memory, CUDA-11.2, and cuDNN-8.1.1 installed. For the unimodal models, we import all the pre-trained weights from the `TORCHVISION.MODELS`,[5] a sub-package of the PyTorch framework. We randomly initialise the remaining weights. Sharma et al. (2022b) re-annotate the HarMeme dataset (Pramanick et al., 2021b) by collecting annotator responses for different roles different entities within memes take. The re-annotated memes may or may not have harmful implications, contrary to the distinction modelled as part of original curation. However, they portray various entities within different contexts implying glorification, vilification, and victimisation. For most of our experiments, we use Adam optimiser (Kingma and Ba, 2015) with a learning rate of $1e^{-4}$ or $1e^{-5}$, a weight decay of $1e^{-5}$ and a Cross-Entropy (CE) loss as the objective function. We optimized our models to obtain hyper-parameter settings (c.f. Table 5) and early-stop to preserve our best state convergence. On average, it took approx. 2:30 hours to train a typical multi-modal neural model on a dedicated GPU system.

B Additional details of `HVVMemes`

B.1 Edge Case

Most memes are intended to project harmless mockery toward various sections of society. Such memes do not imply heroes, villains, or victims; instead, they disseminate harmless humour and trivial opinions. Therefore, Sharma et al. (2022b) do not presume any implications regarding these connotations and categorise them as ‘other’, a fourth *neutral* category, unless expressed otherwise in the meme. A depiction of such a scenario can be observed in Fig. 5. In this meme, it is unclear if Donald Trump is being vilified for being reluctant to use a mask or if the meme expresses a benign attempt at mocking his physical appearance with sarcasm. Additionally, since no background information would suffice to facilitate its complete assimilation, it is categorised as *other*.

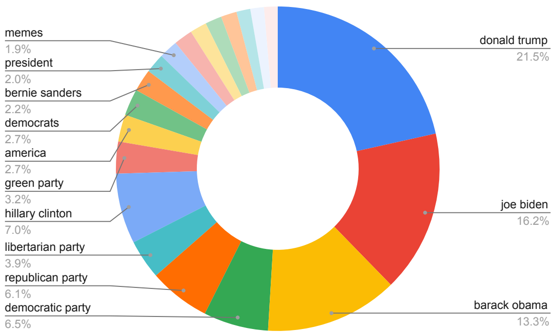


Figure 6: Entity distribution - US Politics.

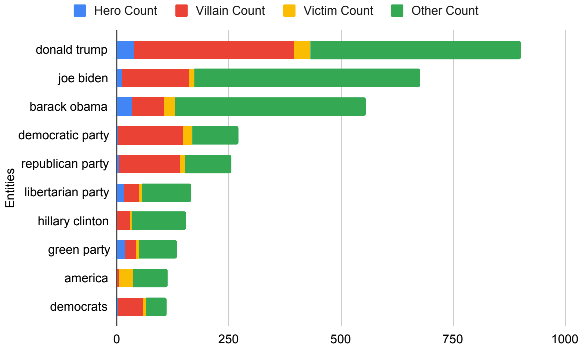


Figure 7: Entity-role distribution - US Politics.

B.2 Analysing Different Entities and Roles

Figure 5: Edge case

Regardless of the connotations and domains in which various entities are referred, only a handful of entities/topics dominate pivotal referencing in memes. The entities in *COVID-19* memes predominantly focus on *Coronavirus*, *Donald Trump*, *mask*, *COVID-19*, and *work from home*. The entities of *Donald Trump*, *Joe Biden*, *Barack Obama*,

---

[5] http://pytorch.org/docs/stable/torchvision/models.html

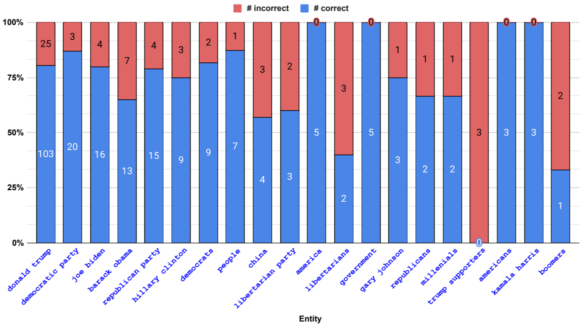


Figure 8: Entity role prediction summary for `HVVMemes`'s (combined) test set. X-axis: Top 20 entities referenced; Y-axis: True positive rate (Recall). Values depicted at the center of each portion for each bar depicts corresponding total counts.

Democratic Party, and *Republican Party* crowd the US Politics memes, as shown in Fig. 6. Such trends highlight the key figures and real-world topics that dominated memetic communication on social media during the period this dataset was compiled, which coincided with the onset of the global COVID-19 pandemic and contemporary US Politics. In COVID-19 memes, we observe that entities like *Donald Trump* and *China* are referenced almost equally within the vilifying and neutral contexts. In contrast, entities like *Corona beer*, *introverts*, and *Tom Hanks*, are invariably referenced in neutral contexts through irony, satire, or benign humour. Similar trends are observed in US Politics memes (Fig. 7), wherein entities like *Donald Trump*, *Democratic Party*, *Republican Party*, and *Democrats* observe almost equivalent referencing as *villain* and *others*. Interestingly, as shown in Fig. 7, US Politics related memes depict a higher propensity towards vilification than that of COVID-19, as most of the prominent entities encountered are vilified at least once in the dataset.

C Entity Role Prediction Analysis.

Careful analysis of the role predictions for various entities elicits the correlation between the role-wise distribution and the test set predictions for different entities. This correlation can be observed from Fig. 8, wherein entities like *Donald Trump, Democratic Party, Joe Biden, Republican Party, Democrats*, etc., which have a *true positive rate* (recall) of atleast 75%, are specifically the ones that have relatively balanced role-wise distribution for the roles of *villain* and *other* within `HVVMemes`, (see Fig. 4 (main content). Interestingly, a lower but role-wise balanced representation within `HVVMemes`, does not appear to deter `VECTOR` from yielding an impressive recall of 90% for an entity such as *Democrats*, which have a total of approximately 125 samples in the training set. On the other hand, for entities like *Barack Obama, China, Libertarian Party, Libertarian*, etc., the memetic portrayal is significantly skewed-in as an *other*, suggesting distinct role-wise imbalance within memes that `VECTOR` failed to accommodate, highlighting its limitations. Further, there are entities like *America, Government, Americans, Kamala Harris*, etc., that register a 100% recall. Role-wise distributions for such cases highlight a distinct majority of neutral connotations within both training and test splits via *other* category, essentially suggesting possible biases within the role-prediction modelling setup. Entity *Trump Supporters* is also observed to be present within the training set via another similar referencing *Donald Trump Supporters*, which has a 1:5 ratio of villain/victim:other. This is complemented by the predominant yet balanced referencing of an independent entity *Donald Trump*. This could lead to the model being confused for such entities as is depicted in Fig. 8.

Systematic Investigation of Strategies Tailored for Low-Resource Settings for Low-Resource Dependency Parsing

Jivnesh Sandhan[1], **Laxmidhar Behera**[1,3] **and Pawan Goyal**[2]
[1]IIT Kanpur, [2]IIT Kharagpur, [3]IIT Mandi
`jivnesh@iitk.ac.in,pawang@cse.iitkgp.ac.in`

Abstract

In this work, we focus on low-resource dependency parsing for multiple languages. Several strategies are tailored to enhance performance in low-resource scenarios. While these are well-known to the community, it is not trivial to select the best-performing combination of these strategies for a low-resource language that we are interested in, and not much attention has been given to measuring the efficacy of these strategies. We experiment with 5 low-resource strategies for our ensembled approach on 7 Universal Dependency (UD) low-resource languages. Our exhaustive experimentation on these languages supports the effective improvements for languages not covered in pretrained models. We show a successful application of the ensembled system on a truly low-resource language Sanskrit.[1]

1 Introduction

Recently, the supervised learning paradigm has dramatically increased the state-of-the-art performance for the dependency parsing task for resource-rich languages (Chen and Manning, 2014; Dyer et al., 2015; Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017; Kulmizev et al., 2019). However, only a handful of resource-rich languages are able to take advantage, and many low-resource languages are far from these benefits (Joshi et al., 2020; More et al., 2019a; Zeman et al., 2018).

In literature, several strategies have been proposed to enhance performance in low-resource scenarios, such as data augmentation (Şahin and Steedman, 2018; Gulordava et al., 2018), cross/monolingual pretraining (Conneau et al., 2020; Peters et al., 2018; Kondratyuk and Straka, 2019), sequential transfer learning (Ruder et al., 2019), multitask learning (Nguyen and Verspoor, 2018), cross-lingual transfer (Das and Sarkar, 2020) and self-training (Rotman and Reichart, 2019; Clark et al., 2018). However, not much attention has been given to measuring the efficacy of the existing low-resource strategies well-known to the community for low-resource dependency parsing (Vania et al., 2019). This is essential to assess their utility for low-resource languages (Hedderich et al., 2021) before inventing novel ways to tackle data sparsity.

In this work, we systematically explore 5 pragmatic strategies for low-resource settings on 7 languages. We experiment with low-resource strategies such as data augmentation, sequential transfer learning, cross/mono-lingual pretraining, multitask learning and self-training. We investigate: (1) How is the trend in performance of each strategy across various languages? Whether the choice of best performing variant of each strategy is language dependent? (2) We integrate the best performing variant of each strategy and call the resulting system as the ensembled system. Do all the strategies contribute towards performance gain in the ensembled system? How well does this ensemble approach generalize across multiple low-resource languages? (3) How far can we push a purely data-driven ensemble system using the best-performing low-resource strategies? Can this simple ensemble approach outperform the state-of-the-art of a low-resource language? We argue that while it may sound like a simple application of techniques well known to the community; it is non-trivial to select the best performing combination for a target low-resource language.

Our exhaustive experimentation empirically establishes the effective generalization ability of the ensembled system on 7 languages and shows average absolute gains of 5.2/6.2 points Unlabelled/Labelled Attachment Score (UAS/LAS) over strong baseline (Dozat et al., 2017). Notably, our ensembled system shows substantial improvements for the languages not covered in pretrained models. Finally, we show a successful application of the ensembled system on a truly low-resource

---

[1] The code and data are available at: `https://github.com/Jivnesh/SanDP`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2164–2171
May 2-6, 2023

language Sanskrit. We find that the ensembled system outperforms the state-of-the-art system (Krishna et al., 2020a) for Sanskrit by 1.2 points absolute gain in terms of UAS and shows comparable performance in terms of LAS (§ 3).

2 Investigation of Strategies Tailored for Low-resource Settings

We explore 5 strategies specially tailored for low-resource settings on 7 languages, and integrate the best performing strategy of each category in our ensembled system (Table 1). We utilize Dozat and Manning (2017) as a base system for all the experiments, henceforth referred to as BiAFF.

Language selection criteria: We choose low-resource languages with less than 2,500 training samples from 4 different typological families such that each language belongs to a unique sub-family. In order to accommodate a low-resource tailored pretraining (Sandhan et al., 2021), we choose languages that have explicit morphological information. Additionally, we divide the set of languages into the languages covered/not-covered in the multilingual language model's pretraining: (1) Covered: Arabic (ar), Greek (el), Hungarian (hu) (2) Not covered: Wolof (wo), Gothic (got), Coptic (cop) and Sanskrit (san).

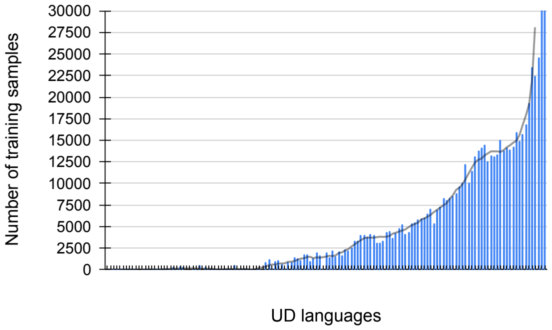


Figure 1: Plot for number of training samples vs. UD languages available in UD-2.6.

Figure 1 illustrates the number of training samples available for all UD languages in UD-2.6. Hedderich et al. (2021) ask "How low is low-resource?" and suggest that the threshold of low-resource is task and language-dependent. The low-resource settings can be seen as a continuum of resource availability due to the absence of a hard threshold. More efforts should focus on evaluating low-resource strategies across multiple languages for a fair comparison between these strategies. Therefore, we select a threshold for the languages with less than 2,500 training samples (Figure 1). We restrict ourselves to the setting where the target low-resource language does not have a high-resource related language that could possibly facilitate the positive cross-lingual or zero-shot transfer (Vulić et al., 2019; Pires et al., 2019; Søgaard et al., 2018; de Lhoneux et al., 2018; Smith et al., 2018). Thus, we do not consider low-resource languages with only a test set available. Also, we do not consider cross-lingual transfer (Duong et al., 2015; Ahmad et al., 2019; Vania et al., 2019; Das and Sarkar, 2020) strategy in our study.

Dataset and metric: For each of these 7 low-resource languages from Universal Dependencies (UD-2.6) (de Marneffe et al., 2021), following Rotman and Reichart (2019), we use 500 data points for training (allocating equal power to each language for fair comparison) and the original dev/test split as dev/test set. Additionally, 1000 morphologically tagged data points (without dependency annotations) are used for self-training and pretraining. We use sentence level macro averaged UAS/LAS metric for evaluation.

Hyper-parameters: For SeqTraL variants, we use the exact same encoder as Ma et al. (2018) with 2 Bi-LSTM layers and decoder with fully connected layer followed by softmax layer. For the ensembled system, we adopt BiAFF's codebase by Ma et al. (2018) with the hyper-parameters setting as follows: the batch size as 16, training iterations as 100, a dropout rate as 0.33, the number of stacked Bi-LSTM layers as 2, learning rate as 0.002 and the remaining parameters as the same as Ma et al. (2018). We release our codebase publicly under creative-common licence.

Computing Infrastructure Used: We primarily use RTX-2080, 12 GB GPU memory, 4352 GPU Cores computing infrastructure for our experiments.

Sequential Transfer Learning (SeqTraL): Following Sandhan et al. (2021), we pretrain three encoders (similar to BiAFF) on three sequence labelling auxiliary tasks and integrate them with the BiAFF encoder using a gating mechanism. We adapt these pretrained encoders with various optimization schemes, proposed for reducing a catastrophic forgetting (French, 1999; McCloskey and

		el		ar		hu		got		cop		wo		san	
Strategy	Model	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
	BiAFF	86.61	82.23	**79.89**	**73.13**	**80.51**	**75.11**	75.66	**69.42**	87.33	**84.48**	**82.63**	**77.83**	75.47	65.77
Data aug.	Cropping	85.98	81.76	79.42	73.03	77.46	71.62	75.72	68.25	86.05	82.95	80.38	75.39	73.82	63.46
	Rotation	86.39	82.19	79.22	72.79	77.33	71.16	76.19	68.89	86.27	83.27	80.66	76.05	75.58	65.14
	Nonce	**87.43**	**82.60**	79.52	73.00	79.56	69.50	**76.42**	67.74	**87.64**	83.52	81.97	76.37	**77.25**	**66.30**
Pretraining	BiAFF+mBERT	91.41	87.89	83.50	76.30	85.20	77.50	64.20	53.20	33.70	15.60	71.50	61.40	71.40	55.12
	BiAFF+XLM-R	**93.61**	**90.85**	**86.04**	**79.55**	**89.05**	**83.80**	-	-	-	-	-	-	78.43	66.72
	BiAFF+LCM	89.00	85.83	82.49	76.67	83.22	78.49	**79.88**	**74.65**	**88.79**	**86.02**	**85.85**	**81.66**	**81.63**	**73.86**
Self-training	SelfTrain	86.78	82.25	80.86	74.45	80.62	75.09	76.96	70.15	87.95	85.33	83.83	78.80	77.53	66.59
	CVT	80.53	77.37	76.21	71.87	75.21	70.01	69.43	63.59	79.32	74.21	73.21	69.50	69.21	56.21
	DCST	**88.26**	**84.09**	**82.21**	**75.78**	**82.85**	**77.65**	**79.52**	**72.91**	**88.85**	**85.53**	**85.51**	**80.71**	**78.55**	**69.10**
SeqTraL	SeqTraL-FE	88.43	85.43	81.86	76.60	82.97	78.46	80.15	75.24	88.08	85.57	85.61	81.77	81.20	73.70
	SeqTraL-UF	88.50	85.36	82.52	76.79	**83.83**	**79.24**	**80.79**	**75.65**	**88.87**	86.30	85.78	81.54	81.51	73.65
	SeqTraL-DL	**89.06**	**85.88**	82.57	76.66	83.36	78.57	80.29	74.89	88.78	86.14	**86.25**	**81.85**	81.17	73.10
	SeqTraL-FT	88.80	85.47	**82.66**	**76.83**	83.79	78.95	80.13	75.11	88.86	**86.31**	86.03	81.64	**81.84**	**73.94**
Multi-tasking	MTL-Case	**86.73**	**82.47**	**80.49**	**74.08**	**80.73**	**75.52**	-	-	86.45	83.82	82.86	77.46	76.15	65.36
	MTL-Label	86.13	81.55	79.86	72.72	80.07	73.92	75.52	69.30	**87.44**	**84.62**	83.08	77.94	76.02	65.20
	MTL-Morph	86.30	82.23	80.02	73.55	80.49	74.70	**77.33**	**71.05**	87.00	84.22	**83.25**	**78.75**	**76.71**	**66.69**
Prop. system	BiAFF	86.61	82.23	79.89	73.13	80.51	75.11	75.66	69.42	87.33	84.48	82.63	77.83	75.47	65.77
	+Pretraining	**93.61**	**90.85**	**86.04**	**79.55**	**89.05**	**83.80**	79.88	74.65	88.79	86.02	85.85	81.66	81.63	73.86
	+MTL	89.99	86.49	82.47	76.24	84.35	79.74	80.33	75.15	88.42	85.94	85.91	81.56	81.30	73.49
	+SeqTraL	90.31	86.70	82.70	76.57	84.58	80.15	**80.79**	**75.65**	**88.87**	**86.30**	**86.05**	**81.85**	**81.84**	**73.94**
	+Self-training	89.83	86.09	82.08	75.92	84.12	79.66	80.08	75.24	88.78	86.07	85.73	81.77	79.89	72.28
	+Data. aug.	89.11	85.87	82.08	75.92	84.12	79.66	79.56	73.53	88.31	84.67	85.73	81.77	79.52	71.89
Evaluation on test set	BiAFF	87.10	83.06	80.92	75.02	80.31	74.16	77.73	70.72	88.50	85.32	80.92	75.02	79.33	67.92
	Prop. system	**93.66**	**90.68**	**86.43**	**79.88**	**88.50**	**82.67**	**82.52**	**77.07**	**89.31**	**86.38**	**87.50**	**82.95**	**83.59**	**74.83**

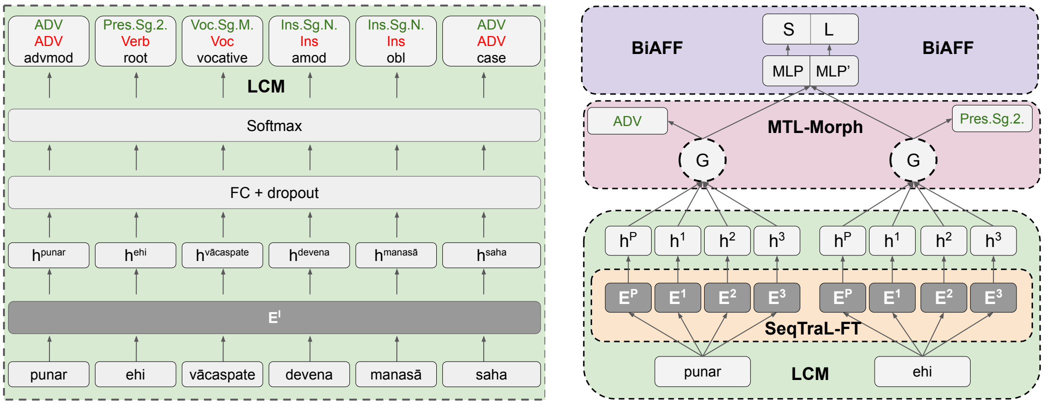
Table 1: Evaluation of low-resource strategies on 7 languages. Experiments are first performed on dev set to find best performing combination of strategies for each language. The best results from strategies from each family are bold and statistically significant compared to its peer baselines belonging to the same family as per t-test ($p < 0.01$). The second last block shows ablations when the best variant from each family is added to the ensembled system. For example, +Data. aug. refers to the system with the best variant from all 5 strategies. The best performing system as per dev set is finally compared with BiAFF on the test set. XLM-R is not compatible with 3 languages and case information of Gothic (*got*) language is missing; hence we do not report their results.

Cohen, 1989). **SeqTraL-FE:** We treat newly integrated layers as Feature Extractors (FE) by freezing them. **SeqTraL-UF:** Gradually Unfreeze (UF) these new layers in the top to down order (Howard and Ruder, 2018; Felbo et al., 2017). **SeqTraL-DL:** The discriminative learning rate (DL) is used for newly added layers (Howard and Ruder, 2018), the learning rate is decreased from top-to-bottom layers. **SeqTraL-FT:** The default learning rate is used to fine-tune all newly added layers.

Cross/mono-lingual Pretraining: We experiment with two multilingual pretrained models, namely, the multilingual BERT (Devlin et al., 2019, **mBERT**) based system (Kondratyuk and Straka, 2019) and the XLM-Roberta (Conneau et al., 2020, **XLM-R**) based system (Nguyen et al., 2021). We also consider supervised pretraining specially tailored for low-resource dependency parsing (Sandhan et al., 2021, **LCM**) which essentially combines three sequence labelling auxiliary tasks. We pretrain it on 1,000 morphologically tagged data points without dependency annotations.

Self-training: Another line of modelling focuses on self-training (Goldwasser et al., 2011; Clark et al., 2018; Rybak and Wróblewska, 2018) to overcome the bottleneck of task-specific labelled data. Earlier attempts failed to prove effectiveness of self-training for dependency parsing (Rush et al., 2012). However, Clark et al. (2018, **CVT**) and Rotman and Reichart (2019, **DCST**), show successful application, thus, we consider these two systems. Also, we generate dependency data by applying a pretrained BiAFF system on 1000 unlabelled data points. We augment this predicted data with gold data and retrain BiAFF in **Self-Train** setting.

Multi-task Learning: We simultaneously train BiAFF and a sequence labelling based auxiliary task in a multi-task setting (**MTL**). We experiment with the following auxiliary tasks: prediction of the morphological label (**MTL-Morph**), dependency relation between a word and its head (**MTL-Label**) and the case label (**MTL-Case**).



(a) (b)

Figure 2: The ensembled system for Sanskrit. Translation: "Oh Vācaspate! Come again with divine mind".

Data Augmentation: Şahin and Steedman (2018) introduce **Cropping:** delete some parts of a sentence to create multiple short meaningful sentences, and **Rotation:** permute the siblings of headword restricted to a set of relations. Both operations modify a set of words or configurational information; however, they do not change the dependencies. **Nonce:** Gulordava et al. (2018) propose to create nonce sentences by substituting a few words which share the same syntactic labels. For each variant, we use additional 1,000 augmented data points.

Results on multilingual experiments: Table 1 first reports results of all 5 strategies on dev set of 7 languages. Next, the second last block of Table 1 (Prop. system) shows ablations on dev set where the best variant from each family is gradually added into the ensembled system. For example, +*Data.aug.* row refers to the system with the best variant from all 5 strategies. Finally, the best performing system as per dev set is compared with BiAFF on the test set. We observe that (1) the best performing variant from augmentation, SeqTraL and MTL families is language dependent. (2) DCST variant of self-training wins over its peer for all the languages. (3) XLM-R outperforms for the languages which are covered in its pretraining (except Sanskrit[2]) and LCM outperforms for the rest of the languages which are truly low-resource. (4) Notably, we find effective generalization ability of the proposed approach on languages covered in cross-lingual pretraining (only pretraining helps) and for the rest of the languages (pretraining, MTL and SeqTraL helps).

3 Application on Sanskrit

Data: We use two standard benchmark datasets available for Sanskrit. We use 1,700, 1,000 and 1,300 sentences (prose domain) from the Sanskrit Treebank Corpus (Kulkarni et al., 2010, STBC) as train, dev and test set, respectively. We also evaluate on the Vedic Sanskrit Treebank (Hellwig et al., 2020, VST) consisting of 1,500 , 1,024 and 1,473 sentences (poetry-prose mixed) as train, dev and test data, respectively. For both data, the final results on the test set are reported using systems trained with combined gold train and dev set.

Baselines: We use More et al. (2019b, **YAP**) and Chang et al. (2016, **L2S**) from transition-based dependency parsing family. Dozat and Manning (2017, **BiAFF**) is a graph-based approach with BiAFFINE attention mechanism. Krishna et al. (2020a, **MG-EBM**) extends Krishna et al. (2020b, **Tree-EBM-F**) using multi-graph formulation. Systems marked with (*) are hybrid systems which leverage linguistic rules from Pāṇini.

The ensembled system: Figure 2 shows the ensembled system for Sanskrit as per Table 1. It consists of two steps, namely, pretraining (**LCM** 🟩) and integration. As shown in Figure 2a, **LCM** pretrains three encoders $E^{(1)-(3)}$ using three independent auxiliary tasks, namely, morphological label prediction, case label prediction and relation label prediction. Thereafter, as shown in Figure 2b, these pretrained encoders are integrated with the BiAFF encoder $E^{(P)}$ using a gating mechanism as employed in Sato et al. (2017). We use **SeqTraL-FT** 🟧 optimization scheme to update the weights

---

[2] Maybe due to limited coverage of corpus for Sanskrit.

of these four encoders. Next, **MTL-Morph** ■ component adds morphological tagging as an auxiliary task to inject complementary signal in the model. Finally, the combined representation of a pair of words in passed to **BiAFF** ■ to calculate probability of arc score (S) and label (L).

	STBC		**VST**	
System	**UAS**	**LAS**	**UAS**	**LAS**
YAP	75.31	66.02	70.37	56.09
L2S	81.97	74.14	72.44	62.76
Tree-EBM-F	82.65	79.28	-	-
BiAFF	85.88	79.55	77.23	67.68
Ours	**88.67**	**83.47**	**79.71**	**69.89**
Tree-EBM-F*	*85.32*	*83.93*	-	-
MG-EBM*	*87.46*	*84.70*	-	-

Table 2: Results on test set for Sanskrit. *Hybrid* systems, marked with (*) use extra-linguistic knowledge and are not directly comparable with our system. Our results are statistically significant compared to BiAFF as per t-test ($p < 0.01$). Results are averaged over 3 runs.

Results: On STBC, the ensembled system outperforms the state of the art *purely data-driven* system (BiAFF) by 2.8/3.9 points (UAS/LAS) absolute gain. Interestingly, it also supersedes the performance of the *hybrid* state of the art system (Krishna et al., 2020a, **MG-EBM**) by 1.2 points (UAS) absolute gain and shows comparable performance for LAS metric. We observe that performance of transition-based systems (**YAP/L2S**) is significantly low compared to graph-based counterparts (**BiAFF/Ours**). We also obtain a similar performance trend for VST data. The VST data is a mixture of dependency labelled trees from both poetry and prose domain. As a result, the overall performance for VST is low compared to STBC due to loss of configurational information.[3]

4 Conclusion and Discussion

We focused on low-resource dependency parsing for multiple languages. We found that our ensembled system can benefit the languages not covered in pretrained models. While multi-lingual pretraining (mBERT and XLM-R) is helpful for the languages covered in pretrained models, LCM pretraining (which simply uses an additional 1,000 morphologically tagged data points) is helpful for the remaining languages. Thus, these findings would help community to pick strategies suitable for their language of interest and come up with robust parsing solutions. Specifically for Sanskrit, our ensembled system superseded the performance of the state-of-the-art *hybrid* system MG-EBM* by 1.2 points (UAS) absolute gain and showed comparable performance in terms of LAS.

Limitations: We could not evaluate on complete UD due to limited available compute resources (single GPU), hence we selected 7 representative languages for our experiments.

Ethics Statement: We do not foresee any ethical concerns with the work presented in this manuscript.

Acknowledgement

We thank Amba Kulkarni for providing Sanskrit dependency treebank data, Anupama Ryali for Śiśupālavadha dataset. We are grateful to Amrith Krishna (Uniphore) for helping us with the initial discussions on this work. We thank Tushar Sandhan (IIT Kanpur), Narein Rao (IIT Kanpur), Rathin Singha (UCLA) and the anonymous reviewers for their constructive feedback towards improving this work. The work of the first author is supported by the TCS Fellowship under the Project TCS/EE/2011191P. The work was supported in part by the National Language Translation Mission (NLTM): Bhashini project by Government of India.

References

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Kai-Wei Chang, and Nanyun Peng. 2019. Cross-lingual dependency parsing with unlabeled auxiliary languages. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 372–382, Hong Kong, China. Association for Computational Linguistics.

Kai-Wei Chang, He He, Hal Daumé, John Langford, and Stephane Ross. 2016. A credit assignment compiler for joint prediction. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 1713–1721, Red Hook, NY, USA. Curran Associates Inc.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

---

[3] We do not evaluate Tree-EBM-F* and MG-EBM* on VST data due to the unavailability of the codebase.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium. Association for Computational Linguistics.

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

Ayan Das and Sudeshna Sarkar. 2020. A survey of the model transfer approaches to cross-lingual dependency parsing. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 19(5).

Miryam de Lhoneux, Johannes Bjerva, Isabelle Augenstein, and Anders Søgaard. 2018. Parameter sharing between dependency parsers for related languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4992–4997, Brussels, Belgium. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

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

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Timothy Dozat, Peng Qi, and Christopher D Manning. 2017. Stanford's graph-based neural dependency parser at the conll 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 845–850.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, Beijing, China. Association for Computational Linguistics.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1486–1495, Portland, Oregon, USA. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Oliver Hellwig, Salvatore Scarlata, Elia Ackermann, and Paul Widmer. 2020. The treebank of vedic Sanskrit. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5137–5146, Marseille, France. European Language Resources Association.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Amrith Krishna, Ashim Gupta, Deepak Garasangi, Pavankumar Satuluri, and Pawan Goyal. 2020a. Keep it surprisingly simple: A simple first order graph based parsing model for joint morphosyntactic parsing in Sanskrit. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4791–4797, Online. Association for Computational Linguistics.

Amrith Krishna, Bishal Santra, Ashim Gupta, Pavankumar Satuluri, and Pawan Goyal. 2020b. A graph based framework for structured prediction tasks in sanskrit. *Computational Linguistics*, 46(4):1–63.

Amba Kulkarni, Sheetal Pokar, and Devanand Shukl. 2010. Designing a constraint based parser for sanskrit. In *International Sanskrit Computational Linguistics Symposium*, pages 70–90. Springer.

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China. Association for Computational Linguistics.

Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. 2018. Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1403–1414, Melbourne, Australia. Association for Computational Linguistics.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 24, pages 109 – 165. Academic Press.

Amir More, Amit Seker, Victoria Basmova, and Reut Tsarfaty. 2019a. Joint Transition-Based Models for Morpho-Syntactic Parsing: Parsing Strategies for MRLs and a Case Study from Modern Hebrew. *Transactions of the Association for Computational Linguistics*, 7:33–48.

Amir More, Amit Seker, Victoria Basmova, and Reut Tsarfaty. 2019b. Joint transition-based models for morpho-syntactic parsing: Parsing strategies for MRLs and a case study from Modern Hebrew. *Transactions of the Association for Computational Linguistics*, 7:33–48.

Dat Quoc Nguyen and Karin Verspoor. 2018. An improved neural network model for joint POS tagging and dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 81–91, Brussels, Belgium. Association for Computational Linguistics.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021. Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Guy Rotman and Roi Reichart. 2019. Deep contextualized self-training for low resource dependency parsing. *Transactions of the Association for Computational Linguistics*, 7:695–713.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexander Rush, Roi Reichart, Michael Collins, and Amir Globerson. 2012. Improved parsing and POS tagging using inter-sentence consistency constraints. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1434–1444, Jeju Island, Korea. Association for Computational Linguistics.

Piotr Rybak and Alina Wróblewska. 2018. Semi-supervised neural system for tagging, parsing and lematization. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 45–54.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009, Brussels, Belgium. Association for Computational Linguistics.

Jivnesh Sandhan, Amrith Krishna, Ashim Gupta, Laxmidhar Behera, and Pawan Goyal. 2021. A little pretraining goes a long way: A case study on dependency parsing task for low-resource morphologically rich languages. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 111–120, Online. Association for Computational Linguistics.

Motoki Sato, Hitoshi Manabe, Hiroshi Noji, and Yuji Matsumoto. 2017. Adversarial training for cross-domain Universal Dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 71–79, Vancouver, Canada. Association for Computational Linguistics.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. 82 treebanks, 34 models: Universal Dependency parsing with multi-treebank models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 113–123, Brussels, Belgium. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788, Melbourne, Australia. Association for Computational Linguistics.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Ivan Vulić, Goran Glavaš, Roi Reichart, and Anna Korhonen. 2019. Do we really need fully unsupervised cross-lingual embeddings? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4407–4418, Hong Kong, China. Association for Computational Linguistics.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Compositional Generalisation
with Structured Reordering and Fertility Layers

Matthias Lindemann[1] and **Alexander Koller**[2] and **Ivan Titov**[1,3]
[1] ILCC, University of Edinburgh, [2] LST, Saarland University, [3] ILLC, University of Amsterdam
`m.m.lindemann@sms.ed.ac.uk`, `koller@coli.uni-saarland.de`, `ititov@inf.ed.ac.uk`

Abstract

Seq2seq models have been shown to struggle with compositional generalisation, i.e. generalising to new and potentially more complex structures than seen during training. Taking inspiration from grammar-based models that excel at compositional generalisation, we present a flexible end-to-end differentiable neural model that composes two structural operations: a fertility step, which we introduce in this work, and a reordering step based on previous work (Wang et al., 2021). To ensure differentiability, we use the expected value of each step, which we compute using dynamic programming. Our model outperforms seq2seq models by a wide margin on challenging compositional splits of realistic semantic parsing tasks that require generalisation to longer examples. It also compares favourably to other models targeting compositional generalisation.[1]

1 Introduction

Many NLP tasks require translating an input object such as a sentence into a structured output object such as a semantic parse. Recently, these tasks have been approached with seq2seq models with great success. However, seq2seq models also have been shown to struggle *out-of-distribution* on compositional generalisation (Lake and Baroni, 2018; Finegan-Dollak et al., 2018; Kim and Linzen, 2020; Hupkes et al., 2020), i.e. the model fails on examples that contain unseen compositions or deeper recursion of phenomena that it handles correctly in isolation.

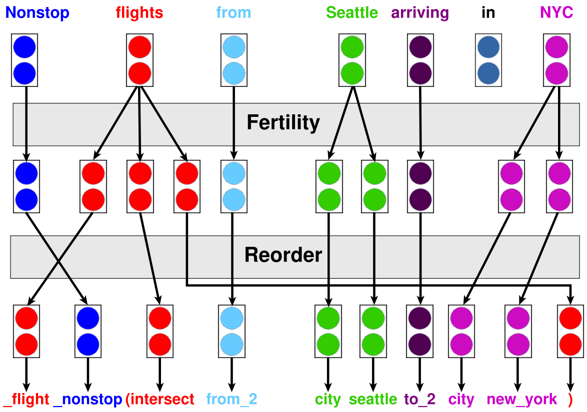


Figure 1: We model structural seq2seq tasks as the composition of differentiable fertility and phrase reordering layers. The model is trained end-to-end without direct supervision of the two structural layers.

Consider the example in Fig. 1. Arguably, any model that produces the given semantic parse from the input in a generalisable way has to capture the correspondence between fragments of the input and fragments of the output, at least implicitly. This is challenging because of structural mismatches between input and output. For example, the fragment contributed by “flights" is discontinuous and intertwined with the rest of the semantic parse.

In contrast to seq2seq models, grammar-based models such as synchronous context-free grammars (SCFGs) (Lewis and Stearns, 1968; Chiang, 2007) explicitly capture the compositional process behind the data and therefore perform very well in compositional generalisation setups. They can also model common structural mismatches like the one shown in the example. However, grammar-based models are rigid and brittle and thus do not scale well.

In this work, we take inspiration from grammar-based models and present an end-to-end differentiable neural model that is both flexible and generalises well compositionally. Our model consists of two structural layers: a phrase reordering layer originally introduced by Wang et al. (2021) and a fertility layer, new in this work, which creates zero or more *copies* of the representation of any input token. We show how to compose these layers to achieve strong generalisation.

We use a simple decoder and essentially translate each token after the fertility and reordering layers

---

[1] `https://github.com/namednil/f-then-r`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2172–2186
May 2-6, 2023 ©2023 Association for Computational Linguistics

independently into one output token. We found this to lead to better generalisation than using an LSTM-based decoder. The simple decoder also makes it easy to integrate grammar constraints to ensure well-formed outputs, e.g. in semantic parsing.

Most seq2seq models tend to predict the end of the sequence too early on instances that are longer than seen in training (Newman et al., 2020). Our model overcomes this by explicitly predicting the length of the output sequence as a sum of fertilities.

We first demonstrate the expressive capacity and inductive bias of our model on synthetic data. We then evaluate on three English semantic parsing datasets (Geoquery, ATIS, Okapi) and a style transfer task (Lyu et al., 2021). We clearly outperform seq2seq models both with and without pretraining in structural generalisation setups, particularly when the model has to generalise to longer examples. Our model also compares favourably with existing approaches that target structural generalisation, and we obtain state-of-the-art results on the structural style transfer tasks.

To summarise, our main contributions are:

- an efficient differentiable fertility layer;
- a flexible end-to-end differentiable model that composes two structural operations (fertility and reordering) and achieves strong performance in structural generalisation tasks.

2 Related Work

Fertility The concept of fertility was introduced by Brown et al. (1990) for statistical machine translation to capture that a word in one language is often consistently translated to a certain number of words in another language. Tu et al. (2016) and Malaviya et al. (2018) incorporate fertility into the attention mechanism of seq2seq models. Cohn et al. (2016); Gu et al. (2016) use heuristic supervision for training their fertility models. In contrast to prior work, we learn an explicit fertility component jointly with the rest of our model.

Monotonic alignment Related to fertility is the concept of monotonic alignment, i.e. an alignment a that maps output positions to input positions such that for any two output positions $i < j$, $a(i) \leq a(j)$. Monotonic alignments are usually modelled by an HMM-like model that places the monotonicity constraint on the transition matrix (Yu et al., 2016; Wu and Cotterell, 2019), leading to a runtime of $O(|\mathbf{x}|^2|\mathbf{y}|)$ with $\mathbf{x}$ being the input and $\mathbf{y}$ the output. Raffel et al. (2017) parameterise the alignment using a series of Bernoulli trials and obtain a training runtime of $O(|\mathbf{x}||\mathbf{y}|)$. Our approach also has $O(|\mathbf{x}||\mathbf{y}|)$ runtime.

Compositional generalisation There is a growing body of work on improving the ability of neural models to generalise compositionally in semantic parsing. Good progress has been made in terms of generalisation to new lexical items (Andreas, 2020; Akyürek and Andreas, 2021; Conklin et al., 2021; Csordás et al., 2021; Ontañón et al., 2022) but structural generalisation remains very challenging (Oren et al., 2020; Bogin et al., 2022).

Herzig and Berant (2021) use a neural chart parser and induce latent trees with an approach similar to hard EM. Their model assumes that one input token corresponds to a single leaf in the tree. Zheng and Lapata (2022) re-encode the input and partially generated output with a transformer for every decoding step to reduce entanglement and show considerable gains in structural generalisation.

There has also been work inspired by quasi-synchronous grammars (QCFGs, Smith and Eisner (2006)). Shaw et al. (2021) heuristically induce a QCFG and create an ensemble of a QCFG-based parser and a seq2seq model. Qiu et al. (2021) use similar QCFGs for data augmentation for a seq2seq model. Our approach does not require constructing a grammar. Finally, Kim (2021) introduces neural QCFGs which perform well on compositional generalisation tasks but are very compute-intensive.

Closest to our work is that of Wang et al. (2021) who reorder phrases and use a monotonic attention mechanism on top. Our approach differs from theirs in several important aspects: (i) we use fertility instead of monotonic attention, which parameterises alignments differently; (ii) we apply the fertility step first and *then* reorder the phrases, so our model can directly create alignments where output tokens aligned to the same input token do not have to be consecutive; (iii) we predict the length as the sum of the fertilities and not with an end-of-sequence token; (iv) they use an LSTM-based decoder network whereas we found that a simpler decoder can generalise better.

3 Background

3.1 Structured Attention

We often want to model the relationship between input $\mathbf{x}$ of length n and output $\mathbf{y}$ of length l by means of a latent variable Z. In particular, we assume that $Z \in \mathcal{Z} \subseteq \mathbb{B}^{n \times l}$ is a boolean alignment matrix. We want to predict $\mathbf{y}$ from a representation produced by a function $g(Z, \mathbf{x})$. In this case, the marginal distribution

$$P(\mathbf{y}|\mathbf{x}) = \mathbb{E}_{P(Z|\mathbf{x})} P(\mathbf{y}|g(Z, \mathbf{x})) \tag{1}$$

can be intractable to compute because $\mathcal{Z}$ often has exponential size in $\mathbf{x}$. In some important cases however we can formulate a similar but tractable model by using *structured attention* (Kim et al., 2017) which 'pushes' the expectation inside the model:

$$P(\mathbf{y}|\mathbf{x}) \approx P(\mathbf{y}|g(\tilde{Z}, \mathbf{x}))$$

where $\tilde{Z} = \mathbb{E}_{P(Z|\mathbf{x})} Z$ is now a 'soft' rather than a boolean matrix. Note that $\tilde{Z}_{ij} = P(Z_{ij} = 1|\mathbf{x})$ is a marginal probability that often can be efficiently computed with a dynamic programme if $P(Z|\mathbf{x})$ is factorisable. We will use such an approach for our model.

3.2 Marginal Permutations

In this section, we briefly review the method of Wang et al. (2021) that we use in our model.

Wang et al. (2021) build on bracketing transduction grammars (Wu, 1997) and show how to compute a distribution over separable permutations (Bose et al., 1998). A permutation is separable, iff it can be represented as a permutation tree. Some permutations are not separable. The internal nodes of a permutation tree are labelled as $\wedge$ or $\triangle$ and are interpreted as operations: $\wedge$ concatenates the values it receives from its left child with the value from its right child, whereas $\triangle$ concatenates them in reverse order. For example, the permutation tree $t = (\triangle\,(\wedge\, a\, b)\,(\triangle\, c\, d))$ represents the permutation $abcd \rightarrow dcab$. Let $R_t(i, j) = 1$ if the permutation described by tree t maps position i to position j, otherwise $R_t(i, j) = 0$.

Wang et al. (2021) show how to compute the expected permutation matrix $\tilde{R}_{i,j} \triangleq \mathbb{E}_{P(t|\mathbf{x})} R_t(i, j)$ in polynomial time with a CYK-style algorithm if $P(t|\mathbf{x})$ factors according to the CYK chart. Crucially, the expected permutation can also be interpreted as a distribution over alignments, where $\tilde{R}_{i,j}$ is the *marginal probability* that position i aligns to position j. We will use this alignment distribution as a building block in our model.

4 Overview of the Approach

In this section, we give a general overview of our approach and defer the details to Sections 5 and 6.

Conceptually, we want to model the transduction from input $\mathbf{x}$ of length n into output $\mathbf{y}$ of length l as the composition of two edit operations. First, we apply a *fertility* step, in which we decide for each token what its fertility is, i.e. how many *copies* we make of it. Assigning fertility of 0 corresponds to deleting the token. This step yields an *intermediate* sequence of tokens. We then reorder them using permutation trees (see Section 3.2). In the last step, we individually translate these tokens into the output tokens. Fig. 1 shows an example where the fertility step and reordering apply to vector representations of tokens.

The fertility step and the reordering can be represented as boolean matrices $F \in \mathbb{B}^{n \times l}$ and $R \in \mathbb{B}^{l \times l}$, respectively. These matrices denote alignments between the sequences before and after the operation. For example, $F_{i,j} = 1$ means that input token i aligns to intermediate token j, i.e. j is one of the copies of i.

With this conceptualisation, we would ideally use the following probabilistic model $P(\mathbf{y}|\mathbf{x})$:

$$P(\mathbf{y}|\mathbf{x}) = \mathbb{E}_{\underbrace{P(F|\mathbf{x})}_{\text{Fertility}}} [\mathbb{E}_{\underbrace{P(R|\mathbf{x}, F)}_{\text{Reordering}}} \underbrace{P(\mathbf{y}|\mathbf{x}, F, R)}_{\text{Decoder}}]$$

At training time, the true fertility values are unknown but we observe the length l of $\mathbf{y}$, so we condition on it:

$$P(\mathbf{y}|\mathbf{x}) = P(l|\mathbf{x}) \cdot \mathbb{E}_{P(F,R|\mathbf{x},l)} P(\mathbf{y}|\mathbf{x}, F, R) \tag{2}$$

where $P(l|\mathbf{x})$ can be computed with dynamic programming relying on the fertility model, as we explain in the next section.

Computing the marginal likelihood and the gradients is intractable. Instead of computing the likelihood exactly, one could sample and use a score function estimator (Williams, 1992) but the resulting gradient estimates have high variance.

Instead, we use structured attention as discussed in Section 3.1 and 'push' the expectations inside the model:

$$\mathbb{E}_{P(F,R|\mathbf{x},l)} P(\mathbf{y}|\mathbf{x}, F, R) \approx P(\mathbf{y}|\mathbf{x}, \tilde{F}, \tilde{R}) \tag{3}$$

with $\tilde{F} = \mathbb{E}_{P(F|\mathbf{x},l)} F$ and $\tilde{R} = \mathbb{E}_{P(R|\mathbf{x},\tilde{F})} R$. $\tilde{F}$ and $\tilde{R}$ now represent 'soft', differentiable versions of fertility and reordering. They approach their discrete counterparts as $P(F|\mathbf{x},l)$ and $P(R|\tilde{F},\mathbf{x})$ become peakier, which tends to happen over the course of training. We can view $(\tilde{F}\tilde{R})_{i,j} = \sum_k \tilde{F}_{i,k}\tilde{R}_{k,j}$ as a probability of aligning i to j in the composition of the operations.

A tendency to memorise larger chunks of the output might contribute to poor compositional generalisation (Hupkes et al., 2020). In order to avoid this, we use a simple decoder that generates each output token independently. A first attempt might look like this:

$$P(\mathbf{y}|\mathbf{x},\tilde{F},\tilde{R}) = \prod_{i=1}^{l}\sum_{j,k} P(\mathbf{y}_i|\mathbf{x}_j)\tilde{F}_{j,k}\tilde{R}_{k,i} \tag{4}$$

where the summation over j and k marginalises over all possible alignments to output position i.

Distinguishing copies The independence assumptions in Eq. (4) imply that $P(\mathbf{y}_i|\mathbf{x}_j)$ will have the same distribution for *all* copies of x_j, i.e. the model cannot express a preference to translate the first copy of *Seattle* to `city` and the second copy to `seattle` in Fig. 1. To enable this, we distinguish different copies of an input token.

We do this by defining F not as a matrix but as a tensor $F \in \mathbb{B}^{n\times l\times d}$ where d is some fixed maximum fertility value. Let $F_{i,j,u} = 1$ iff intermediate token j is the u-th copy of input token i. For example, in Fig. 1, $F_{4,6,1} = 1$ and $F_{4,7,2} = 1$. With this definition of F (and accordingly defined $\tilde{F}$), we can define a stronger decoder:

$$P(\mathbf{y}|\mathbf{x},\tilde{F},\tilde{R}) = \prod_{i=1}^{l}\sum_{j,k,u} P(\mathbf{y}_i|\mathbf{x}_j,u)\tilde{F}_{j,k,u}\tilde{R}_{k,i}$$

where we now additionally marginalise over which copy of the input sequence we are translating.

5 Fertility and Alignment

In this section we describe how to compute $\tilde{F} = \mathbb{E}_{P(F|\mathbf{x},l)}F$, i.e. the expected alignment that results from the fertility step given that the intermediate token sequence will have length l.

In the previous section, we looked at the fertility step mostly from the perspective of an alignment between the input tokens and the intermediate tokens. However, the fertility step is *parameterised* as assigning a fertility value $f_i \in 0,\ldots,d \in \mathbb{N}$ to every token x_i. We now show how to compute $\tilde{F}$ efficiently as a function of the distribution over fertility values $P(\mathbf{f}|\mathbf{x})$.

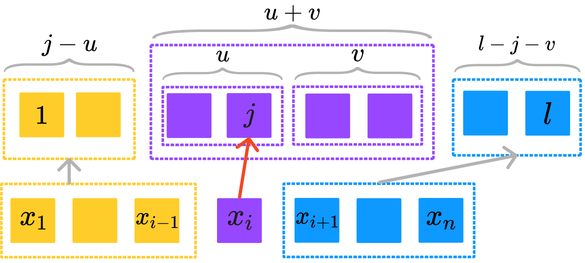


Figure 2: Efficiently computing the marginal probability that the u-th copy of i is at position j. We partition the intermediate sequence into four parts, and marginalise over all possible ways of choosing v.

We denote the alignment that follows from $\mathbf{f}$ as $F(\mathbf{f})$, i.e. $F(\mathbf{f})_{i,j,u} = 1$ iff intermediate token j is the u-th copy of token i. $\tilde{F}$ can be expressed as:

$$\tilde{F} = \mathbb{E}_{P(F(\mathbf{f})|\mathbf{x},l)}F(\mathbf{f}) \tag{5}$$

where we assume that the fertility values are independent of each other conditioned on $\mathbf{x}$:

$$P(F(\mathbf{f})|\mathbf{x}) \triangleq P(\mathbf{f}|\mathbf{x}) \triangleq \prod_{i=1}^{n} P(f_i|\mathbf{x})$$

Note that conditioning on the output length l in Eq. (5) introduces inter-dependencies between the values. In order to compute $\tilde{F}_{i,j,u}$, we need to marginalise over all possible assignments to the fertility vector $\mathbf{f}$ which satisfy $F(\mathbf{f})_{i,j,u} = 1$. We do this with a dynamic programming algorithm that is similar to the forward/backward algorithm for HMMs (Baum, 1972).

Computing marginals We characterise the situations where $F(\mathbf{f})_{i,j,u} = 1$ as the integer solutions to a set of equations (see also Fig. 2). First, in order for intermediate token j to be the u-th copy of i, there should be $j-u$ intermediate tokens generated by input tokens preceding i:

$$f_1 + \ldots + f_{i-1} = j - u \tag{6}$$

Second, the fertility f_i has to be at least u. We capture this by requiring

$$f_i = u + v \tag{7}$$

with $v \geq 0$. Finally, the input tokens following i have to contribute the remaining $l-j-v$ intermediate tokens:

$$f_{i+1} + \ldots + f_n = l - j - v \tag{8}$$

We then compute the probability of creating a sequence of length l, where the j-th intermediate token is the u-th copy of i, by marginalising over v (dropping $\mathbf{x}$ for readability):

$$P(F(\mathbf{f})_{i,j,u} = 1, f_0 + \ldots + f_{n+1} = l) \tag{9}$$
$$= \sum_{v=0}^{\min(l-j,d-u)} P(f_0 + \ldots + f_{i-1} = j - u) \times$$
$$P(f_i = u + v) P(f_{i+1} + \ldots + f_{n+1} = l - j - v)$$

We handle the boundary cases with $i = 1, i = n$ by adding dummy variables f_0 and f_{n+1} and setting $P(f_0 = 0) = P(f_{n+1} = 0) = 1$.

In order to compute Eq. (9), we also compute 'forward' probabilities $P(f_0 + \ldots + f_i = h)$ and 'backward' probabilities $P(f_i + \ldots + f_{n+1} = h)$ for all i, h. This can be done recursively:

$$P(f_0 + \ldots + f_i = h) =$$
$$\sum_{r=0}^{d} P(f_i = r) P(f_0 + \ldots + f_{i-1} = h - r)$$

Finally, $\tilde{F}_{i,j,u} = 1/P(f_0 + \ldots + f_{n+1} = l) \times P(F(\mathbf{f})_{i,j,u} = 1, f_0 + \ldots + f_{n+1} = l)$ because $\tilde{F}_{i,j,u}$ was defined in Eq. (5) by conditioning on the length l. We provide pseudo-code for all steps in Appendix A.

Runtime The runtime of the fertility step is dominated by computing Eq. (9). Note that v has a range of at most d, thus, we can compute the probabilities for all i, j, u in $O(n \cdot l \cdot d^2)$ time. Eq. (9) can be computed in parallel for each i, j, u.

6 Composing Fertility and Reordering

We will now describe in detail how $P(F|\mathbf{x})$ and $P(R|\mathbf{x}, \tilde{F})$ are defined and how the fertility and reordering layers are composed.

6.1 Fertility Layer

The fertility layer takes a desired output length l and a sequence of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ as input (GloVe embeddings (Pennington et al., 2014) in our case) and returns a marginal alignment $\tilde{F}$ (see Section 5) and a sequence of vectors $\mathbf{h}_1, \ldots, \mathbf{h}_l$ for use as input to the next layer. We compute the distribution over fertilities by first encoding $\mathbf{x}_1, \ldots, \mathbf{x_n}$ with a bidirectional LSTM, yielding a sequence of hidden states $\mathbf{h}^f_1, \ldots, \mathbf{h}^f_n$. We then model $P(f_i|\mathbf{x}) = \text{softmax}_\tau(\text{MLP}^f(\mathbf{h}^f_i))$, where τ is the temperature parameter of the softmax.

We use $\tilde{F}$ (Eq. (5)) as a form of structured attention to compute the input to the reordering layer:

$$\mathbf{h}_j = \sum_{i,u} \tilde{F}_{i,j,u}(\mathbf{x}_i + \mathbf{w}_u)$$

where $\mathbf{w}_u$ is a learned embedding indicating that j is a u-th copy of some token. Intuitively, $\mathbf{h}_j$ for an intermediate token j represents the corresponding token in the input sequence and also indicates *which* copy of that token it is.

6.2 Reordering Layer

Given the output $\mathbf{h}_1, \ldots, \mathbf{h}_l$ from the fertility layer, the reordering layer computes the alignment distribution $\tilde{R}$ as the expected permutation following Wang et al. (2021). This procedure involves populating a CYK-style chart with scores. We first run a bidirectional LSTM with a skip connection over $\mathbf{h}$ and compute a contextualised representation of the tokens after the fertility step:

$$\mathbf{h}^r_i = [\text{LSTM}^r(\mathbf{h}_{\leq i}), \text{LSTM}^r(\mathbf{h}_{\geq i})] + \mathbf{h}_i$$

Based on these representations, we compute scores for the chart following Stern et al. (2017).

6.3 Decoder

Our decoder factors as follows (see Section 4):

$$P(\mathbf{y}|\mathbf{x}, \tilde{R}, \tilde{F}) = \prod_{i=1}^{l} \underbrace{\sum_{j,k,u} P(\mathbf{y}_i|\mathbf{x}_j, u) \tilde{F}_{j,k,u} \tilde{R}_{k,i}}_{P(\mathbf{y}_i|\mathbf{x}, \tilde{R}, \tilde{F})}$$

$P(\mathbf{y}_i|\mathbf{x}_j, u)$ conditions only on the original input token and on the index indicating which copy of this token we are translating. For this reason, we contextualise the *input* with a bidirectional LSTM with a skip connection:

$$\mathbf{h}'_j = \rho[\text{LSTM}^d(\mathbf{x}_{\leq j}), \text{LSTM}^d(\mathbf{x}_{\geq j})] + \mathbf{x}_j \tag{10}$$

with ρ as hyperparameter.

We experiment with three versions of the decoder. In (i), we parameterise $P(\mathbf{y}_i|\mathbf{x}_j, u)$ as $P(\mathbf{y}_i|\mathbf{x}_j, u) = \text{softmax}(W_u\text{MLP}(\mathbf{h}'_j))$. In (ii), we additionally use a copy mechanism (Gu et al., 2016). In (iii), we use an autoregressive variant where we encode $\mathbf{y}_{<i}$ with an LSTM, defining $P(\mathbf{y}_i|\mathbf{y}_{<i}, u) = \text{softmax}(W_u\text{MLP}(\mathbf{h}'_j + \text{LSTM}(\mathbf{y}_{<i}))$.

6.4 Training

As mentioned in Section 4, at training time we condition on the observed length l of $\mathbf{y}$: $P(\mathbf{y}|\mathbf{x}) = P(l|\mathbf{x})P(\mathbf{y}|\mathbf{x}, \tilde{F}, \tilde{R})$ with $\tilde{F}$ conditioned on l. We use a weighted version of the log likelihood as the objective function:

$$\sum_i \lambda_1 \log P(l^i|\mathbf{x}^i) + \log P(\mathbf{y}^i|\mathbf{x}^i, l^i)$$

with i ranging over the training examples.

For the semantic parsing tasks, we found it necessary to give our model a reasonable starting point in terms of alignments. We encourage it to respect high-confidence automatic alignments during the first m training epochs by adding the following term to our objective function:

$$\lambda_2 \sum_{(i,j)\in\mathcal{A}} \log \sum_{k,u} \tilde{F}_{i,k,u}\tilde{R}_{k,j}$$

where $\mathcal{A}$ is the set of alignments with a posterior probability of at least χ according to an IBM-1 alignment model (Brown et al., 1993).

Initialising an alignment model with alignments from a simpler model was a common strategy in statistical machine translation (Och and Ney, 2003).

6.5 Inference

In order to make predictions with a trained model, we want to compute the most likely output $\mathbf{y}$ given the input $\mathbf{x}$. It is convenient to treat the length as a discrete variable and use the same algorithm for computing $\tilde{F}$ as derived for training. We therefore search for $\arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \arg\max_l P(l|\mathbf{x})P(\mathbf{y}^l|\mathbf{x}, l)$ with $\mathbf{y}^l = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}, \tilde{F}, \tilde{R})$. For any given l we can easily find $\mathbf{y}^l$ in versions (i) and (ii) of the decoder:

$$\mathbf{y}_i^l = \arg\max_{\mathbf{y}_i} P(\mathbf{y}_i|\mathbf{x}, \tilde{R}, \tilde{F})$$

For version (iii) of the decoder, we use greedy search instead. It would be too costly to compute $\mathbf{y}^l$ for all l, so we explore only the top k most likely lengths.

Grammar-based decoding In executable semantic parsing, we want to produce only well-formed outputs, which can be characterised by a context-free grammar G. In practical applications, this grammar is needed to execute the query, so there is little extra engineering effort in using it for decoding. We search for

$$\mathbf{y}^l = \arg\max_{\mathbf{y}\in L(G)} P(\mathbf{y}|\mathbf{x}, l)$$

Because of the simple decoder we can do this exactly by applying a modified version of Viterbi CYK. Unlike in parsing though, the string is not observed because it is exactly what we are looking for. Therefore, we fill all entries from i to i in the chart C with $C_{A,i,i} = \max_{A\to a\in G} P(y_i = a|\mathbf{x}, \tilde{R}, \tilde{F})$. We then continue with the normal Viterbi CYK with weights of 1 on all other rules.

7 Evaluation

7.1 Synthetic Data

In order to probe the expressive capacity and inductive bias of our model, we evaluate on mirroring task $T = \{(w, ww^R)|w \in \Sigma^*\}$, e.g. $abc \to abccba$. The challenge is that the length of the dependency between output tokens grows with the length of the example. Models are trained on examples with input lengths 3 to 9, and tested in two setups. In the first setup (Length), the model has to generalise to examples with lengths 11 to 20; we use examples with length 10 as validation data. In the second setup (unseen combination, UC), the model only sees the symbols `x`, `y` and `z` grouped together as `xyz` on the input side at training time. The model is tested on examples that contain `x`, `y` or `z` adjacent to other symbols. See Appendix B.1 for further details on the setup.

We compare our model without copying (F→R) with a variant that first applies the reordering and then the fertility step (R→F) and autoregressive variants of the two (AR F→R and AR R→F). As baselines, we also compare with an LSTM-based seq2seq model with attention and a Transformer with relative positional embeddings, which was previously shown by Csordás et al. (2021) to perform well at compositional generalisation. AR R→F has similarities to Wang et al. (2021) who first reorder and then use an autoregressive decoder with monotonic attention.

Results Table 1 shows mean accuracy across 5 random initialisations. The accuracy of the relative Transformer and the LSTM-based seq2seq model drops sharply for longer inputs. In contrast, F→R and AR F→R generalise perfectly even to much longer examples. In the UC setup, F→R outperforms the rest by a wide margin. Interestingly, all

Model	Accuracy					
		Length				UC
	dev	11	12	13	14 - 20	test
Transformer	82.0	3.0	0.0	0.0	0.0	42.0
LSTM	**100.0**	94.3	67.9	6.9	0.0	0.1
R→F	0.0	0.0	0.0	0.0	0.0	1.3
AR R→F	90.0	85.7	89.8	87.5	80.5	1.2
F→R	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**79.9**
AR F→R	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	32.1

Table 1: Exact match accuracy on the mirroring task.

autoregressive models struggle in this setup, including AR F→R which obtained perfect accuracy in the Length setup. This is consistent with the hypothesis that autoregressive models tend to memorise entire chunks (Hupkes et al., 2020).

Expressivity of F→R and R→F For this task, an input token corresponds to two output tokens that may be arbitrarily far apart from each other. F→R can learn this alignment because this can be captured by a separable permutation (see Section 3.2) following the fertility step duplicating every input token. In contrast, R→F and AR R→F cannot represent the correct alignment directly because the fertility step is applied only after the reordering, leading to alignments between an input token and a *contiguous span* in the output. However, in theory, they can represent this alignment *implicitly* through the LSTM (Eq. (10)). The evaluation shows that this does not work reliably in practice: we find that R→F gets stuck in bad local minima and fails completely on the task. While AR R→F performs well in the Length setup, it is weak in the UC setup.

7.2 Geoquery

Geoquery (Zelle and Mooney, 1996) is a standard dataset for semantic parsing and has recently been used to evaluate to what extent semantic parsers are capable of generalising to (i) structurally unseen queries (template split), and (ii) structurally unseen long examples. We follow the setup of Herzig and Berant (2021), using the variable-free FunQL representation (Kate et al., 2005), a copy mechanism, and evaluate with execution accuracy.

Model	iid	Template	Length
Seq2Seq ‡ (HB)	78.5	46.0	24.3
Seq2Derivation (HB)	72.1	54.0	24.6
BART-base‡ (HB)	87.1	67.0	19.3
Span (HB)	78.9	65.9	41.4
Span + lexicon (HB)	86.1	82.2	63.6
Liu et al. (2021)‡	-	**84.1**	-
Wang et al. (2021)‡	75.2*	43.2*	-
R→F	**89.1**±1.0	80.4±1.2	68.6±1.4
R→F ‡	83.5±0.7	73.0±1.6	49.2±5.1
F→R	88.6±3.3	79.9±2.5	**68.8**±5.2
F→R ‡	80.7±2.3	68.7±4.3	53.4±5.9
AR F→R ‡	81.1±1.0	52.8±3.3	37.6±1.8

Table 2: Accuracy on different splits of Geoquery. HB is Herzig and Berant (2021) and ‡ refers to systems that do not enforce well-formedness of the output. * Wang et al. (2021) use exact match accuracy and anonymise named entities instead of copying.

Results Table 2 shows the results on the different splits. We report means and standard deviations of 5 random initialisations. Our method performs well across the different splits, and in particular on the length split that evaluates a challenging form of compositional generalisation. As an ablation, we remove the grammar-based decoding. We notice a considerable drop in accuracy but it still outperforms the baselines. The drop in accuracy is slightly bigger out-of-distribution than in-distribution.

In line with the experiments on synthetic data, AR F→R and the approach of Wang et al. (2021) drastically lose accuracy when going from in-distribution to the compositional generalisation setups. This provides further evidence that a strong decoder can hinder compositional generalisation.

In contrast to the experiments on synthetic data, R→F and F→R perform comparably. Manual inspection of the data shows that good alignments on this dataset can be obtained even with the stronger assumption on possible alignments made by R→F.

7.3 ATIS

ATIS (Dahl et al., 1994) is a semantic parsing datasets for flight bookings. In comparison to Geoquery, the queries tend to be longer and the word order is more flexible. We use the variable-free FunQL notation as annotated by Guo et al. (2020). Apart from the original iid test split, we create a length split: Semantic parses with fewer than 4 conjuncts form the training set, parses with exactly 4 conjuncts form the development set and the test set contains instances with more than 4 conjuncts. Details on the split and preprocessing are in Appendix B.3.

We compare our model with finetuned BART-base (Lewis et al., 2020), an LSTM-based seq2seq model with attention and the relative Transformer

Model	iid	Length
LSTM seq2seq	76.52±1.66	4.95±2.16
LSTM seq2seq‡	75.98±1.30	4.95±2.16
Rel. Transformer‡	75.76±1.43	1.15±1.41
BART-base‡	**86.96**±1.26	19.03±4.57
F→R	74.15±1.35	**35.41**±4.09
F→R ‡	68.26±1.53	29.91±2.91

Table 3: Accuracy on different splits of ATIS.

Model	Calendar	Document	Email
BART-base‡	36.7±3.0	2.7±2.1	20.5±9.8
F→R	**69.5**±13.9	**42.4**±5.7	**55.6**±2.7
F→R ‡	57.2±19.9	36.1±5.6	43.9±3.8

Table 4: Accuracy on length splits by domain on Okapi.

(Csordás et al., 2021). We also run a version of the LSTM-based model with a large beam of size 50 and filter our instances that are not well-formed; the resulting outputs are well-formed at least 99.7% of the time.

Results Table 3 shows mean accuracy and standard deviations of 5 random initialisations. While on the iid split, our approach does not quite reach the same accuracy as the baselines, it outperforms them on the compositional length split by a margin of more than 16 points. Without grammar-based decoding, we again observe a noticeable loss in accuracy but we still substantially outperform BART on the length split. Constraining the output to be grammatical does not appear as beneficial for the LSTM baseline.

7.4 Okapi

Hosseini et al. (2021) introduce a semantic parsing dataset for evaluating compositional generalisation on three domains (document, calendar, email). Since template splits were not found to be challenging, we focus on generalising to longer examples. Models are trained on short examples with up to 3 'parameters' (such as filtering based on an attribute or ordering results) and are tested on examples with more parameters (at least 4 for the calendar and email domain and at least 5 for the document domain). The splits are described in Appendix B.4. In contrast to the other datasets we consider, Okapi is noisy because it was collected with crowd workers. This presents an additional challenge.

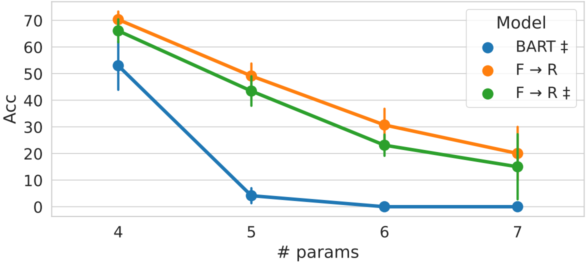


Figure 3: Accuracy on the document domain of Okapi by number of parameters in the gold logical form.

Results Table 4 shows the results of our model with copying from 5 random initialisations. F→R outperforms fine-tuned BART-base by a large margin, in particular on the challenging split of the document domain.

Fig. 3 shows the accuracy on the development and test set of the document domain as a function of the number of parameters in the gold logical form. BART performs relatively well when applied to examples with one more parameter than seen in the training set but then its performance drops sharply. F→R is more accurate and its accuracy also drops much slower with the number of parameters. We notice different failure modes for the two models: on the test set, BART deviates by 5.4 tokens on average from the gold length, whereas F→R deviates only by 1.0 tokens on average. This is in line with the observations of Newman et al. (2020) that seq2seq models systematically predict the end of sequence token too early on long out-of-distribution examples. Our results suggest that predicting length as a sum of fertilities is more robust towards this shift in distribution.

7.5 Style Transfer

In addition to being important for compositional generalisation, structural inductive biases can help when only little data is available. We evaluate our model in such a scenario on the style transfer tasks of Lyu et al. (2021). A model is given an English sentence and asked to reformulate it to conform with a certain 'style'. We focus on the tasks identified as challenging by Lyu et al. (2021): *active to passive* (2462 training examples), *adjective emphasis* (627 examples) and *verb emphasis* (1081 examples).

For the emphasis tasks, the word to be emphasised is provided in the input. Following Kim (2021), to incorporate it, we add a special learned embedding vector to the embedding of that token.

Transfer Type	Approach	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr
Active to passive	GPT-2 (Lyu et al., 2021)	47.6	32.9	23.8	18.9	21.6	46.4	1.820
	Seq2seq (Kim, 2021)	83.8	73.5	67.3	59.8	46.7	77.1	5.941
	Neural QCFG (Kim, 2021)	83.6	77.1	71.3	66.2	49.9	80.3	6.410
	Human (Lyu et al., 2021)	93.1	88.1	83.5	79.5	58.7	90.5	8.603
	F→R	**92.1**±1.3	**85.4**±1.5	**79.7**±1.7	**74.6**±1.9	**55.9**±0.5	**86.0**±1.1	**7.610**±0.130
Adj. emphasis	GPT-2 (Lyu et al., 2021)	26.3	7.9	2.8	0.0	11.2	18.8	0.386
	Seq2seq (Kim, 2021)	50.5	29.6	18.4	11.9	24.2	51.4	1.839
	Neural QCFG (Kim, 2021)	67.6	50.6	39.3	31.6	37.3	68.3	3.424
	Human	83.4	75.3	67.9	66.1	52.2	81.1	6.796
	F→R	**78.3**±1.4	**61.9**±0.7	**49.9**±1.3	**40.5**±1.4	**43.0**±1.0	**69.1**±0.6	**4.268**±0.170
Verb emphasis	GPT-2 (Lyu et al., 2021)	30.9	17.0	9.5	4.1	14.0	29.2	0.593
	Seq2seq (Kim, 2021)	52.6	38.9	29.4	21.4	29.4	46.4	2.346
	Neural QCFG (Kim, 2021)	66.4	51.2	40.7	31.9	37.0	**58.9**	3.227
	Human (Lyu et al., 2021)	64.9	56.9	49.3	42.1	43.3	69.3	5.668
	F→R	**68.4**±0.6	**52.7**±0.6	**41.6**±0.7	**32.8**±0.6	**37.4**±0.4	**58.9**±0.4	**3.498**±0.121

Table 5: Results on the hard style transfer tasks from Lyu et al. (2021). All models except for GPT2 use copying.

Results Table 5 shows the results comparing our F→R model to previous work based on three random initialisations. We achieve state-of-the-art results on all style transfer tasks on all metrics. The improvement compared to prior work is strongest for the active-to-passive task and weakest for the verb emphasis task, where our model ties with Kim (2021) in terms of ROUGE-L.

8 Conclusion

We presented a flexible end-to-end differentiable model for structured NLP tasks. It predicts the output sequence from the input by composing a fertility layer with a reordering layer. The evaluation shows that our model performs well in structural generalisation setups, in particular when the model has to generalise to longer examples than seen during training. In contrast, the accuracy of standard seq2seq models drops sharply on longer examples.

The efficient fertility layer introduced in this work may be useful in other scenarios as well, e.g. in non-autoregressive machine translation, or for (unsupervised) sentence compression when the fertility is restricted to 0 or 1. Future work could also investigate other structured layers and the best ways of composing and training them.

9 Limitations

The fertility layer is efficient but a limitation of the model presented in this paper is the high runtime complexity of the reordering layer. It makes it impractical for long output sequences (e.g. more than 50 tokens). While the structured reordering step can represent many permutations of practical interest, we observed a small number of cases where our model could not produce the correct permutation. We note that our approach is modular and the reordering layer could be replaced by a faster one with fewer restrictions in future work, e.g. based on one-to-one matchings.

While our method obtains strong accuracy in compositional generalisation setups without contextualised encoders, it remains an open question how different ways of integrating contextualised encoders affect the performance of our method in compositional generalisation setups. In addition, it is an open question how a *pretrained decoder* would influence our model's ability to generalise compositionally since we found that a non-pretrained LSTM-based decoder can be detrimental.

Acknowledgements

We thank Bailin Wang and Jonas Groschwitz for technical discussions and we thank Agostina Calabrese and Verna Dankers for comments on this paper.

ML is supported by the UKRI Centre for Doctoral Training in Natural Language Processing, funded by the UKRI (grant EP/S022481/1), the University of Edinburgh, School of Informatics and School of Philosophy, Psychology & Language Sciences, and a grant from Huawei Technologies. ML and IT acknowledge the support of the European Research Council (ERC StG BroadSem 678254). IT is also supported by the Dutch National Science Foundation (NWO Vici VI.C.212.053).

References

Ekin Akyürek and Jacob Andreas. 2021. Lexicon learning for few shot sequence modeling. In *Proceedings of the ACL-IJCNLP (Volume 1: Long Papers)*, pages 4934–4946, Online.

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the ACL*, pages 7556–7566, Online.

Leonard E Baum. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3(1):1–8.

Ben Bogin, Shivanshu Gupta, and Jonathan Berant. 2022. Unobserved local structures make compositional generalization hard. *arXiv preprint arXiv:2201.05899*.

Prosenjit Bose, Jonathan F. Buss, and Anna Lubiw. 1998. Pattern matching for permutations. *Information Processing Letters*, 65(5):277–283.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the ACL: Human Language Technologies*, pages 876–885, San Diego, California.

Henry Conklin, Bailin Wang, Kenny Smith, and Ivan Titov. 2021. Meta-learning to compositionally generalize. In *Proceedings of the ACL-IJCNLP (Volume 1: Long Papers)*, pages 3322–3335, Online.

Róbert Csordás, Kazuki Irie, and Juergen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 EMNLP*, pages 619–634, Online and Punta Cana, Dominican Republic.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany.

Jiaqi Guo, Qian Liu, Jian-Guang Lou, Zhenwen Li, Xueqing Liu, Tao Xie, and Ting Liu. 2020. Benchmarking meaning representations in neural semantic parsing. In *Proceedings of the 2020 EMNLP (EMNLP)*, pages 1520–1540, Online.

Jonathan Herzig and Jonathan Berant. 2021. Span-based semantic parsing for compositional generalization. In *Proceedings of the ACL-IJCNLP (Volume 1: Long Papers)*, pages 908–921, Online.

Saghar Hosseini, Ahmed Hassan Awadallah, and Yu Su. 2021. Compositional generalization for natural language interfaces to web apis. *arXiv preprint arXiv:2112.05209*.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: how do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Rohit J. Kate, Yuk Wah, Wong Raymond, and J. Mooney. 2005. Learning to transform natural to formal languages. In *Proceedings of AAAI-05*.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 EMNLP (EMNLP)*, pages 9087–9105, Online.

Yoon Kim. 2021. Sequence-to-sequence learning with latent neural grammars. In *Advances in Neural Information Processing Systems*, volume 34, pages 26302–26317. Curran Associates, Inc.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the ACL*, pages 7871–7880, Online.

Philip M Lewis and Richard Edwin Stearns. 1968. Syntax-directed transduction. *Journal of the ACM (JACM)*, 15(3):465–488.

Chenyao Liu, Shengnan An, Zeqi Lin, Qian Liu, Bei Chen, Jian-Guang Lou, Lijie Wen, Nanning Zheng, and Dongmei Zhang. 2021. Learning algebraic recombination for compositional generalization. In *Findings of the ACL: ACL-IJCNLP 2021*, pages 1129–1144, Online.

Yiwei Lyu, Paul Pu Liang, Hai Pham, Eduard Hovy, Barnabás Póczos, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2021. StylePTB: A compositional benchmark for fine-grained controllable text style transfer. In *Proceedings of the 2021 Conference of the North American Chapter of the ACL: Human Language Technologies*, pages 2116–2138, Online.

Chaitanya Malaviya, Pedro Ferreira, and André F. T. Martins. 2018. Sparse and constrained attention for neural machine translation. In *Proceedings of the 56th Annual Meeting of the ACL (Volume 2: Short Papers)*, pages 370–376, Melbourne, Australia.

Benjamin Newman, John Hewitt, Percy Liang, and Christopher D. Manning. 2020. The EOS decision and length extrapolation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 276–291, Online.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Santiago Ontañón, Joshua Ainslie, Zachary Fisher, and Vaclav Cvicek. 2022. Making transformers solve compositional tasks. In *Proceedings of the 60th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 3591–3607, Dublin, Ireland.

Inbar Oren, Jonathan Herzig, Nitish Gupta, Matt Gardner, and Jonathan Berant. 2020. Improving compositional generalization in semantic parsing. In *Findings of the ACL: EMNLP 2020*, pages 2482–2495, Online.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 EMNLP (EMNLP)*, pages 1532–1543, Doha, Qatar.

Linlu Qiu, Peter Shaw, Panupong Pasupat, Paweł Krzysztof Nowak, Tal Linzen, Fei Sha, and Kristina Toutanova. 2021. Improving compositional generalization with latent structure and data augmentation. *arXiv preprint arXiv:2112.07610*.

Colin Raffel, Minh-Thang Luong, Peter J Liu, Ron J Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *International Conference on Machine Learning*, pages 2837–2846. PMLR.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2021. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In *Proceedings of the ACL-IJCNLP (Volume 1: Long Papers)*, pages 922–938, Online.

David Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 23–30, New York City.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 818–827, Vancouver, Canada.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany.

Bailin Wang, Mirella Lapata, and Ivan Titov. 2021. Structured reordering for modeling latent alignments in sequence transduction. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Yuk Wah Wong and Raymond Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 439–446, New York City, USA.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the ACL*, pages 1530–1537, Florence, Italy.

Lei Yu, Jan Buys, and Phil Blunsom. 2016. Online segment to segment neural transduction. In *Proceedings of the 2016 EMNLP*, pages 1307–1316, Austin, Texas.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Hao Zheng and Mirella Lapata. 2022. Disentangled sequence to sequence learning for compositional generalization. In *Proceedings of the 60th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 4256–4268, Dublin, Ireland.

Algorithm 1 Forward

```
Require: f $\in \mathbb{R}^{n\times d}$, normalised over the second axis, output sequence length $l$
function FWD(f, $l$)
    Init fwd $\in \mathbb{R}^{n\times l}$ with zeros
    for $h$ in range(0, d) do                          ▷ Base case
        fwd[0, h] = f[0, h]
    for $i$ in range(1, n) do                          ▷ Recursion
        for $h$ in range(0, $l$) do
            for $r$ in range(0, min(d, h+1)) do
                fwd[i, h] += fwd[i-1, h-r] × f[i, r]
    return fwd
```

Algorithm 2 Marginals

```
Require: f $\in \mathbb{R}^{n\times d}$, normalised over the second axis, output sequence length $l$
function MARGINALS($f, l$)
    Init F $\in \mathbb{R}^{n\times l\times d}$ with zeros
    fwd = fwd(f, $l$)
    bwd = rev(fwd(rev(f),$l$))
    for $i$ in range(0, n-1) do
        for $j$ in range(0, $l$) do
            for $u$ in range(1, min(d, j+1+1)) do
                for $v$ in range(0, min(d-u, $l$-j)) do
                    F[i, j, u] += fwd[i, j+1-r1] ×
                        f[i,u+v]×bwd[i+1,$l$-(j+1+v)]
    divide all entries in F by fwd[i, $l$]
    return F
```

A Pseudo code

We present the algorithms for computing $\tilde{F}$ in python-style pseudo code, using 0-based indexing and `range(i,k)` refers to the integers $i, \dots, k-1$. As in the main paper, d refers to the maximum fertility value (hyperparameter).

Algorithm 1 shows how to compute all forward probabilities. If `rev(f)` reverses matrix `f` on the first dimension (e.g. like `torch.flip`), then we can compute the backward probabilities as `rev(fwd(rev(f), l))`. Algorithm 2 shows how $\tilde{F}$ can be computed.

B Data, grammars, pre- and post-processing

B.1 Synthetic data

In both setups, we generate 4000 training examples, 200 development examples and 1000 test examples. We use an alphabet size of $|\Sigma| = 11$ for the Length setup and $|\Sigma| = 11 + 3$ for the UC setup to accommodate for `x,y,z`. Symbols are chosen uniformly at random. In the Length setup, we choose the length of the example uniformly at random. In the UC setup, we do so as well but with probability 0.2 we insert an `xyz` cluster if this does not exceed the maximum length.

In the UC setup, we use development data from the training distribution.

We do not use grammar-based decoding or copying on the synthetic data.

B.2 Geoquery

We remove all parentheses in the logical forms, as they can be restored in post-processing. We also remove quotes around named entities in preprocessing to enable copying (`'texas'` becomes `texas`) and restore them in post-processing. Following Herzig and Berant (2021), we only allow copying of named entities and do not copy predicate symbols (e.g. `river`).

We use the grammar of Wong and Mooney (2006) for decoding as provided by Guo et al. (2020).

B.3 ATIS

We found that a naive length split led to having very few examples in the training set that used a date since both the month and the day count as one conjunct each. Therefore, we created 33 templates with three conjuncts based on existing ATIS examples (with four or more conjuncts) that contain dates and add 8 instances of each template with randomly chosen dates to the training set. In addition, we removed any exact duplicate samples from the data.

Similarly to Geoquery, we remove those parentheses that can be deterministically recovered in post-processing. However, in contrast to Geoquery the parentheses for `or` and `intersection` need to be kept because the arities of those operators are not fixed. We run all our experiments on this representation of ATIS.

For grammar-based decoding we use the “typed" grammar provided by Guo et al. (2020) and do not use the copy mechanism.

B.4 Okapi

We found there was too much distributional overlap in the original length split provided by Hosseini et al. (2021) and therefore use our own split:

For the document domain, our development set contains examples with 4 parameters, and the test

Dataset	Split/Version	Train	Dev	Test
Geoquery	iid	540	60	280
	template	544	60	276
	length	540	60	280
ATIS	iid	4465	497	448
	length	4017	942	331
Okapi/length	Calendar	1145	200	1061
	Document	2328	412	514
	Email	2343	200	991
Style transfer	active to passive	2462	136	137
	adjective emphasis	627	34	35
	verb emphasis	1081	60	60

Table 6: Number of examples per dataset/split.

set contains examples with at least 5 parameters. For the other two domains, there is insufficient data for such out-of-distribution development sets. Therefore, we chose our test set to contain all examples with at least 4 parameters and our development sets to consist of 95% in-distribution data and 5% of examples from the the examples with 4 parameters (which makes the bulk of the test distribution).

We manually create a grammar of well-formed logical forms for the three domains of Okapi (included in the code).

B.5 StylePTB

For comparability, we also tokenise on whitespace following Kim (2021). We do not restrict the output of the model with a grammar.

C Details on evaluation metrics

We provide code for all evaluation metrics in our repository/dependencies.

Geoquery We use the code of Herzig and Berant (2021) to compute execution accuracy (`https://github.com/jonathanherzig/span-based-sp`).

ATIS We allow for different order of conjuncts between system output and gold parse in computing accuracy. We do this by sorting conjuncts before comparing two trees node by node.

Okapi We follow Hosseini et al. (2021) and disregard the order of the parameters for computing accuracy. Since Hosseini et al. (2021) did not make their evaluation code publicly available, we use our own implementation. Our implementation uses sets and does not punish a model for predicting a correct parameter multiple times. For example, if the gold logical form contains `FILTER message.isRead eq False`, a necessary condition for a prediction to count as correct is that it must contain this string *at least* once.

StylePTB We follow Kim (2021) and use `https://github.com/Maluuba/nlg-eval` (commit `7f79930`) for all evaluation metrics, ensuring that BLEU, ROUGE and METEOR are scaled to 0 - 100.

D Hyperparameters

We provide a configuration file for each of our models with the chosen hyperparameters in our code repository (`configs/`). We set the maximum fertility value d to $d = 4$ for all datasets except for the style transfer tasks where we set it to $d = 3$.

At test time, we explore the top $k = 5$ most likely lengths when using grammar-based decoding. Without grammar-based decoding we used $k = 1$ as using $k = 5$ provided little improvement.

Many but not all instances of Geoquery require identity permutations. We found this to lead to the issue that the model gets stuck in a very steep local minimum within the first epoch where it would predict only identity permutations. We fixed this issue by reducing the learning rate in the feedforward network that predicts the scores for the permutation trees to 1×10^{-6}.

D.1 Hyperparameter selection

We select hyperparameters using a combination of manual selection and a random search. We optimise hyperparameters for accuracy on the development set of compositional generalisation splits, where available, (Length setup for the synthetic data, template split for Geoquery), and then use those hyperparameters for all splits of a (domain of a) dataset.

The high variance in accuracy across random initialisations often observed in compositional generalisation setups makes it difficult to tune hyperparameters even if an out-of-distribution development set exists. We restrict random hyperparameter search to two random seeds. After the hyperparameter search, we pick the two most promising configurations (according to (execution) accuracy), pick a new random seed and train them again to choose the one which provides the most stable accuracy (approximated as the highest accuracy on the new seed). We then run the main experiments

Dataset	Model	# params
Mirror	F→R and R→F	1.019
	AR F→R and AR R→F	1.12
	LSTM	3.187
	Transformer	33.067
Geoquery	F→R and R→F	2.541
	AR F→R	2.765
ATIS	LSTM	4.669
	Transformer	58.468
	F→R	3.511
Okapi/Calendar	F→R	2.466
Okapi/Document	F→R	2.468
Okapi/Email	F→R	2.485
Style/Active to passive	F→R	2.814
Style/Adjective emphasis	F→R	2.698
Style/Verb emphasis	F→R	3.133

Table 7: Number of parameters in millions in our models.

with the chosen hyperparameters and completely new random seeds.

We randomly sample 20 configurations per hyperparameter search. Since this procedure is expensive, we do not run train our models fully to convergence. The bounds of the hyperparameter search are reported in our repository.

For all our models, we initially chose hyperparameters manually, and then ran a random hyperparameter search as described above. If the manually chosen hyperparameters resulted in same or better performance (on the development set) on average, we kept those, and otherwise used the ones found by the hyperparameter search.

For example, on Geoquery, we noticed a particular sensitivity to hyperparameters, and the manually selected hyperparameters for F→R performed best with low variance, whereas for R→F, the hyperparameters found by the random search were better than the manually chosen ones. We think this sensitivity is at least in part caused by the small size of the dataset.

Style transfer In contrast to the other tasks we evaluate on, we did not run a hyperparameter search for the style transfer tasks and use the same manually determined hyperparameters for F→R across all style transfer tasks.

E Computing infrastructure and runtime

All experiments were run on GeForce GTX 1080 Ti or GeForce GTX 2080 Ti with 12GB RAM and Intel Xeon Silver or Xeon E5 CPUs.

The runtime of one run contains the time for training, evaluation on the devset after each epoch and running the model on the test set. We show the runtime the models we train in Table 8.

Dataset	Model	Epochs	Runtime
Mirror	F→R	7	8 min
	R→F	20	8 min
	AR F→R	20	30 min
	AR R→F	20	20 min
	Transformer	200	15 min
	LSTM	60	4 min
Geo	F→R	100	20 min
ATIS	F→R	100	11-12h
	Transformer	20	10 min
	LSTM	20	10 min
	BART	50	1.3h
Okapi / Calendar	F→R	70	1 h
	BART	40	15 min
Okapi / Document	F→R	70	1.5 h
	BART	60	30 min
Okapi / Email	F→R	70	1.5 h
	BART	40	30 min
Active → Passive	F→R	60	1 h
Adj. emphasis	F→R	60	20 min
Verb emphasis	F→R	60	30 min

Table 8: Average total runtime of the models we train. For comparison on Geoquery, Herzig and Berant (2021) report a runtime of 2 hours on comparable hardware to ours.

F Additional results

Geoquery Table 10 shows *exact match* accuracy of our models for comparison and Table 11 shows results on the development set.

ATIS Table 13 shows results on the development set. Table 12 shows the average deviation from the gold length.

Okapi Table 14 shows results on the development set.

StylePTB Table 9 shows results on the development set.

Transfer Type	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr
Active to passive	94.1±0.3	88.6±0.2	83.8±0.2	79.7±0.3	57.6±0.1	87.9±0.5	7.883±0.061
Adj. emphasis	78.4±1.2	64.3±1.0	54.5±0.7	46.9±0.9	44.0±0.4	69.5±0.6	4.809±0.113
Verb emphasis	67.6±0.5	51.2±0.7	40.6±0.9	32.7±1.3	36.8±0.3	59.1±0.7	3.587±0.118

Table 9: Development accuracy of our F→R model with copying on StylePTB. We report means and standard deviations of three random initialisations.

Model	iid	Template	Length
R→F	83.5±0.7	72.7±2.0	54.0±4.0
R→F ‡	76.4±1.4	65.8±2.2	39.4±6.2
F→R	83.4±2.6	72.0±3.3	55.2±5.3
F→R ‡	76.9±2.2	63.6±4.7	46.0±6.6
AR F→R ‡	77.4±1.3	47.5±4.2	34.4±2.2

Table 10: Exact match accuracy on the splits of **Geoquery**. We report mean and standard deviation of the 5 random initialisations shown in the main paper.

Model	iid	Template	Length
F→R ‡	84.3±3.0	85.3±1.4	83.7±1.4
R→F ‡	83.7±2.2	83.0±3.0	79.0±3.2
AR F→R ‡	83.7±2.7	81.7±2.0	87.3±2.5

Table 11: Mean and standard deviations of execution accuracy on the development sets of the **Geoquery** splits (without grammar-based decoding).

Model	iid	Length
LSTM seq2seq ‡	0.46±0.09	5.39±0.42
Rel. Transformer ‡	0.49±0.07	6.19±0.52
BART-base ‡	0.24±0.02	3.40±0.25
F→R ‡	0.41±0.06	1.49±0.18

Table 12: Means and standard deviations of the average absolute deviation from the gold length by model on ATIS, i.e. lower is better

Model	iid	Length
LSTM seq2seq ‡	81.53±0.77	43.86±2.93
Rel. Transformer ‡	81.53±0.44	34.84±5.28
BART-base ‡	90.54±0.45	65.29±1.01
F→R ‡	73.80±1.62	54.42±1.25

Table 13: Mean and standard deviations accuracy on the development sets of the **ATIS** splits (without grammar-based decoding).

Model	Calendar	Document	Email
BART-base ‡	94.8±0.3	56.2±10.9	91.4±0.4
F→R ‡	86.4±3.4	66.1±4.8	85.0±2.8

Table 14: Mean and standard deviations accuracy on the development sets of the **Okapi** splits (without grammar-based decoding).

Investigating Multi-source Active Learning for Natural Language Inference

Ard Snijders[1] **Douwe Kiela**[2] **Katerina Margatina**[3]
[1]University of Amsterdam [2]Stanford University [3]University of Sheffield
`ardsnijders@gmail.com`

Abstract

In recent years, active learning has been successfully applied to an array of NLP tasks. However, prior work often assumes that training and test data are drawn from the same distribution. This is problematic, as in real-life settings data may stem from several sources of varying relevance and quality. We show that four popular active learning schemes fail to outperform random selection when applied to unlabelled pools comprised of *multiple* data sources on the task of natural language inference. We reveal that uncertainty-based strategies perform poorly due to the acquisition of collective outliers, i.e., hard-to-learn instances that hamper learning and generalization. When outliers are removed, strategies are found to recover and outperform random baselines. In further analysis, we find that collective outliers vary in form between sources, and show that hard-to-learn data is not always categorically harmful. Lastly, we leverage dataset cartography to introduce difficulty-stratified testing and find that different strategies are affected differently by example learnability and difficulty.

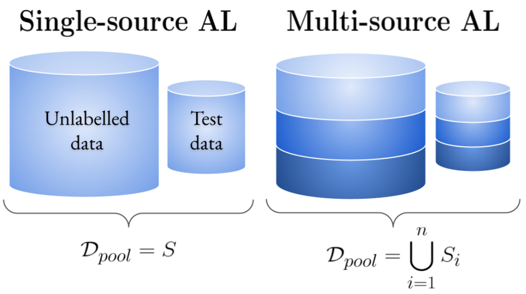


Figure 1: The pool of unlabelled data consists of a source S, or multiple sources $\bigcup_i S_i$, for the *single-source* and *multi-source* AL setting, respectively. The in-domain test set follows the same distribution of sources.

1 Introduction

In recent years, active learning (AL) (Cohn et al., 1996) has emerged as a promising avenue for data-efficient supervised learning (Zhang et al., 2022). AL has been successfully applied to a variety of NLP tasks, such as text classification (Zhang et al., 2016; Siddhant and Lipton, 2018; Prabhu et al., 2019; Ein-Dor et al., 2020; Margatina et al., 2022), entity recognition (Shen et al., 2017; Siddhant and Lipton, 2018; Lowell et al., 2019), part-of-speech tagging (Chaudhary et al., 2021) and neural machine translation (Peris and Casacuberta, 2018; Liu et al., 2018; Zhao et al., 2020).

However, these works share a major limitation: they often implicitly assume that unlabelled training data comes from a single *source*[1] (Houlsby et al., 2011; Sener and Savarese, 2018; Huang et al., 2016; Gissin and Shalev-Shwartz, 2019; Margatina et al., 2021). We refer to this setting as *single-source AL*. The single-source assumption is problematic for various reasons (Kirsch et al., 2021). In real-life settings we have no guarantees that all unlabelled data at our disposal will necessarily stem from the same distribution, nor will we have assurances that all examples are of consistent quality, or that they bear sufficient relevancy to our task. For instance, quality issues may arise when unlabelled data is collected through noisy processes with limited room for monitoring individual samples, such as web-crawling (Kreutzer et al., 2022). Alternatively, one may have access to *several* sources of unlabelled data of decent quality, but incomplete knowledge of their relevance to the task-at-hand. For instance, medical data may be collected from various different physical sites (hospitals, clinics, general practitioners) which may differ statistically from the targte distribution due to e.g. differences in patient demographics. Ideally, AL methods should be robust towards these conditions in order to achieve adequate solutions.

---

[1] Throughout the paper, we use the term "*source*" to describe the varying domains of the textual data that we use in our experiments. More broadly, "different sources" refers to having data drawn from different distributions.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2187–2209
May 2-6, 2023 ©2023 Association for Computational Linguistics

In this work, we study whether existing AL methods can adequately select relevant data points in a *multi-source scenario* for NLP. We examine robustness by evaluating AL performance in both in-domain (ID) and out-of-domain (OOD) settings, while we also conduct an extensive analysis to interpret our findings. A primary phenomenon of interest here concerns *collective outliers*: data points which models struggle to learn due to high ambiguity, the requirement of specialist skills or labelling errors (Han and Kamber, 2000). While such outliers were previously found to disrupt several AL methods for visual question answering (Karamcheti et al., 2021), their impact on text-based tasks remains under-explored. Given the wide body of work on the noise and biases that pervade NLI (Bowman et al., 2015; Williams et al., 2017; Gururangan et al., 2018; Poliak et al., 2018; Tsuchiya, 2018; Geva et al., 2019; Liu et al., 2022), we may reasonably assume popular NLI datasets to suffer from collective outliers as well.

Concretely, our contributions can be summarised as follows: **(1)** We apply several popular AL methods in the under-explored multi-source, pool-based setting on the task of NLI, using RoBERTa-large (Liu et al., 2020) as our acquisition model, and find that no strategy consistently outperforms random selection (§5). **(2)** We seek to explain our findings by creating datamaps to explore the actively acquired data (§6.1) and show that uncertainty-based acquisition functions perform poorly due to acquisition of collective outliers (§6.2). **(3)** We examine the effect of training data difficulty on downstream performance (§7.1) and after thorough experiments we find that uncertainty-based AL methods recover or even surpass random selection when hard-to-learn data points are removed from the pool (§7.2). **(4)** Finally, we introduce difficulty-stratified testing and show that the *learnability* of acquired training data affects different strategies differently at test-time (§7.3). Our code is publicly available at `https://github.com/asnijders/multi_source_AL`.

2 Related Work

Multi-source AL for NLP While AL has been studied for a variety of tasks in NLP (Siddhant and Lipton, 2018; Lowell et al., 2019; Ein-Dor et al., 2020; Shelmanov et al., 2021; Margatina et al., 2021; Yuan et al., 2022; Schröder et al., 2022; Margatina et al., 2022; Kirk et al., 2022; Zhang et al., 2022), the majority of work remains limited to settings where training data is assumed to stem from a single source. Some recent works have sought to address the issues that arise when relaxing the single-source assumption (Ghorbani et al., 2021; Kirsch et al., 2021; Kirsch and Gal, 2021), though results remain primarily limited to image classification. Moreover, these works study how AL fares under the presence of *corrupted* training data, such as duplicating images or adding Gaussian noise, and they do not consider settings where sampling from multiple sources may be *beneficial* due to complementary source attributes. He et al. (2021) examine a multi-domain AL setting, but they focus on leveraging common knowledge between domains to learn a *set* of models for a set of domains, which contrasts with our single-model pool-based setup. Closest to our work, Longpre et al. (2022) explore pool-based AL over multiple domains and find that some strategies consistently outperform random on question answering and sentiment analysis. However, the authors crucially omit a series of measurements, as they instead perform *a single AL iteration*, limiting the effectiveness of AL, while complicating comparison with our results.

Dataset Cartography Karamcheti et al. (2021) employ dataset cartography (Swayamdipta et al., 2020) and show that a series of AL algorithms fail to outperform random selection in visual question answering due to the presence of collective outliers. Zhang and Plank (2021) apply datamaps to AL and introduce the cartography active learning strategy, identifying that examples with poor learnability often suffer from label errors. Our work contrasts with both of these works in that we show that hard-to-learn data is not always unequivocally harmful to learning. Moreover, both works only examine learnability of *training* examples, whereas we also consider how learnability of acquired data affects model performance at test-time.

3 Single & Multi-source Active Learning

We assume a warm-start, pool-based AL scenario (Settles, 2010) with access to a pool of *unlabelled* training data, $\mathcal{D}_{pool}$, and a seed dataset of labelled examples, $\mathcal{D}_{train}$. During each iteration i of AL, we first train a model $\mathcal{M}$ with $\mathcal{D}_{train}$ and then use it in conjunction with some acquisition function $\mathcal{A}$ to select a new batch of unlabelled examples $\mathcal{D}_{batch}$ from $\mathcal{D}_{pool}$ for labelling. Upon labelling, these examples are removed from $\mathcal{D}_{pool}$

and added to $\mathcal{D}_{train}$, after which a new round of AL begins.

$$\mathcal{D}^{i+1}_{train} = \mathcal{D}^{i}_{train} \cup \mathcal{D}^{i+1}_{batch}$$
$$\mathcal{D}^{i+1}_{pool} = \mathcal{D}^{i}_{pool} \setminus \mathcal{D}^{i+1}_{batch}$$

Single-source AL For single-source AL, we assume that $\mathcal{D}_{pool}$ is a set of unlabelled data that stems from a single source S, i.e. $\mathcal{D}_{pool} = S$.

Multi-source AL For multi-source AL, we assume that $\mathcal{D}_{pool}$ comprises a union of distinct sources $S_1, S_2, ..., S_n$ such that $\mathcal{D}_{pool} = \bigcup_{i=1}^{n} S_i$.

3.1 Data Acquisition

An acquisition function $\mathcal{A}$ is responsible for selecting the most *informative* unlabelled data from the pool, aiming to improve over random sampling. We use a set of acquisition functions which we deem representative for the wider AL toolkit: Monte Carlo Dropout Max-Entropy, (**MCME**; Gal et al., 2017) is an uncertainty-based acquisition strategy where we take the mean label distribution over T Monte-Carlo dropout (Gal and Ghahramani, 2015) network samples and select the k data points with the highest predictive entropy. Bayesian Active Learning by Disagreement, (**BALD**; Houlsby et al., 2011) is an uncertainty-based acquisition strategy which employs Bayesian uncertainty to identify data points for which many models disagree about. Discriminative Active Learning (**DAL**; Gissin and Shalev-Shwartz, 2019) is a diversity-based acquisition function designed to acquire a training set that is indistinguishable from the unlabelled set.

3.2 Analysis of Acquired Data

At each data acquisition step, we seek to examine what kind of data each acquisition function has selected for annotation. Following standard practice in active learning literature (Zhdanov, 2019; Yuan et al., 2020; Ein-Dor et al., 2020; Margatina et al., 2021) we profile datasets acquired by strategies via acquisition metrics. Concretely, we consider the *input diversity* and *output uncertainty* metrics. We provide more details in Appendix A.1.

Input Diversity To evaluate the diversity of acquired sets in the input space, we follow Yuan et al. (2020) and measure input diversity as the Jaccard similarity between the set of tokens from the acquired training set $\mathcal{D}_{train}$ and the set of tokens from the *remainder* of the unlabelled pool $\mathcal{D}_{pool}$. This function assigns *high* diversity to strategies acquiring samples with *high* token overlap with the unlabelled pool, and vice versa.

Output Uncertainty To approximate the output uncertainty of an acquired training set $\mathcal{D}_{train}$ for a given strategy, we follow Yuan et al. (2020) and use a model trained on the *entire dataset* to compute predictive entropy of all the examples in the dataset that we want to examine. The model is trained on all training data as this grants more accurate uncertainty measurements.

4 Experimental Setup

Data We perform experiments on Natural Language Inference (NLI), a popular classification task to gauge a model's natural language understanding (Bowman et al., 2015). We construct the unlabelled pool from three distinct datasets: **SNLI** (Bowman et al., 2015), **ANLI** (Nie et al., 2019) and **WANLI** (Liu et al., 2022). We consider **MNLI** (Williams et al., 2017) as an out-of-domain set to evaluate the transferability of actively acquired training sets. For more details see Appendix A.2.

Experiments We apply AL with two distinct end-goals: **in-domain** (ID) generalization, where the same source(s) are used for both the unlabelled pool $\mathcal{D}_{pool}$ and the test set $\mathcal{D}_{test}$, and **out-of-domain** (OOD) generalization, where we evaluate on the test set of an external source of which no training data was present in the unlabelled pool.

Constructing multi-source pools For multi-source AL, the unlabelled pool comprises the *union* of the SNLI, ANLI and WANLI training sets. During model selection, we similarly assume the union of SNLI, ANLI and WANLI validation sets. For both training and validation data we down-sample sources to the minority source size to obtain pools with even shares per source. We sub-sample *training data* for each source to reduce experiment runtimes. This yields a pool of 60K unlabelled training examples comprised of 20K shares of each source.

AL Parameters We assume an initial seed training set of size $|\mathcal{D}^{i=0}_{train}| = 500$ and an acquisition size $k = 500$, such that k examples are acquired per round of AL. We perform 7 rounds of AL: the final actively acquired labelled dataset $\mathcal{D}^{i=7}_{train}$ comprises 4K examples. We run each experiment 5 times with different random seeds to account for stochasticity in parameter initializations.

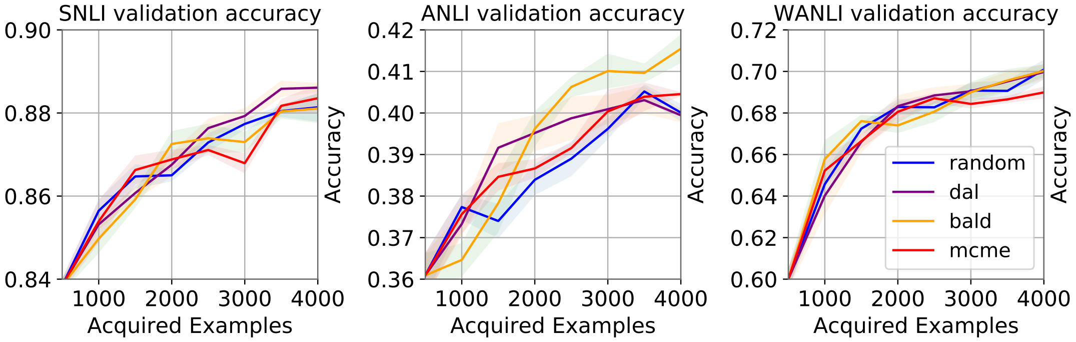


Figure 2: AL **single-source** in-domain learning curves.

5 Results

5.1 AL over single sources

We first provide single-source AL results in Figure 2. AL has mixed performance across sources; it fails to consistently outperform random selection on SNLI and WANLI, though most strategies tend to outperform random on ANLI. However, none of the strategies consistently outperform others across sources. Moreover, the extent of improvement over the acquisition phase varies between tasks; e.g., for SNLI, strategy curves are close and the accuracy improvement from 500 to 4K examples is small. This is also reflected in test outcomes (Table 1), with models performing similarly across sources.

Observing Table 2,[2] we find that uncertainty-based methods (BALD, MCME) tend to acquire the most uncertain data points, as expected. Still, differences between strategies are small for WANLI and ANLI. Conversely, RANDOM and DAL tend to acquire data with greater diversity in the input space, but again variance across methods is low. Here, observed homogeneity in the input space may explain why test outcomes lie closely together: if the acquired training sets per strategy are very similar, we should expect to see similarly small differences in test outcomes for models trained on them. Relating these outcomes to the per-source learning curves in Figure 2, however, we observe no clear relation between metric results and performance outcomes: *acquiring uncertain or diverse data does not seem to be predictive of AL success throughout the acquisition process.*

5.2 AL over multiple sources

Next, we provide the results of our multi-source setting, as described in §3, and observe that again AL fails to consistently outperform random sampling on the aggregate ID test set of SNLI, ANLI and WANLI (Figure 3, bottom left). We also evaluate on the OOD MNLI that was not present in the unlabelled training data (Figure 3, top left) and still find that *AL fails to outperform* RANDOM.

Analyzing the acquired data in terms of input diversity, uncertainty and label distributions (Table 2) There do not appear to be clear relations between metric outcomes and strategy *performance* other-

	SNLI	ANLI	WANLI
RANDOM	87.09 ± 0.30	39.90 ± 1.02	$\mathbf{70.24} \pm 0.70$
DAL	87.49 ± 0.20	39.84 ± 1.15	69.80 ± 1.00
BALD	87.37 ± 0.40	$\mathbf{40.16} \pm 0.69$	69.54 ± 0.70
MCME	$\mathbf{87.58} \pm 0.80$	40.15 ± 0.60	69.07 ± 1.20

Table 1: **Single-source AL** test results. At test-time, the labeled set $\mathcal{D}_{train}$ comprises 4K examples.

Task	Strategy	I-Div.	Unc.	N	E	C
SNLI	RANDOM	0.259	0.051	0.34	0.33	0.33
	DAL	**0.267**	0.067	0.38	0.29	0.33
	BALD	0.249	0.072	0.32	0.31	0.37
	MCME	0.261	**0.093**	0.32	0.34	0.34
WANLI	RANDOM	**0.324**	0.208	0.47	0.38	0.15
	DAL	0.321	0.223	0.49	0.38	0.13
	BALD	0.317	0.213	0.40	0.39	0.21
	MCME	0.318	**0.229**	0.38	0.35	0.27
ANLI	RANDOM	**0.492**	0.082	0.43	0.32	0.26
	DAL	0.467	0.084	0.47	0.3	0.23
	BALD	0.478	0.094	0.39	0.30	0.31
	MCME	0.491	**0.110**	0.37	0.31	0.31
Multi	RANDOM	0.276	0.405	0.41	0.34	0.25
	DAL	0.276	0.463	0.47	0.34	0.19
	BALD	0.220	0.445	0.34	0.32	0.35
	MCME	**0.323**	**0.556**	0.40	0.30	0.30

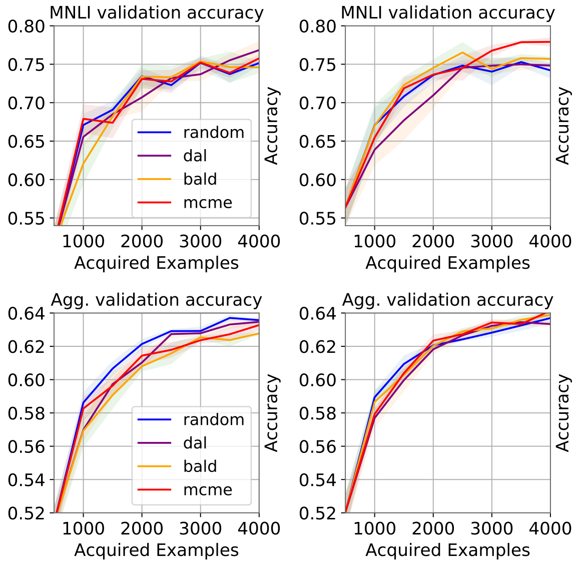
Table 2: Profiling strategy acquisitions for **single-source** and **multi-source** AL in terms of input diversity (**I-Div.**), uncertainty (**Unc.**) and class distributions (**N**: *neutral*, **E**: *entailment*, **C**: *contradiction*) across seeds. Each strategy is profiled *after* all rounds of AL.

---

[2] See Table 9 for standard deviations in the Appendix.

	SNLI	ANLI	WANLI	MNLI
RANDOM	$85.9 \pm 0.9 \mid 85.3 \pm 0.3$	$37.8 \pm 1.2 \mid 37.9 \pm 0.8$	$67.7 \pm 1.6 \mid 67.2 \pm 0.9$	$74.2 \pm 2.3 \mid 74.4 \pm 1.5$
MCME	$85.9 \pm 0.7 \mid 84.3 \pm 0.6$	$\mathbf{\underline{38.6}} \pm 1.0 \mid \underline{38.4} \pm 0.9$	$67.4 \pm 0.4 \mid 66.3 \pm 0.7$	$\mathbf{\underline{77.8}} \pm 1.3 \mid 75.6 \pm 2.2$
BALD	$\mathbf{\underline{86.3}} \pm 1.0 \mid \underline{85.8} \pm 0.2$	$36.9 \pm 1.3 \mid 36.1 \pm 0.5$	$\mathbf{\underline{68.4}} \pm 1.1 \mid 65.9 \pm 1.1$	$75.8 \pm 0.9 \mid 74.3 \pm 1.6$
DAL	$84.3 \pm 1.2 \mid 85.3 \pm 0.9$	$37.8 \pm 1.2 \mid 37.3 \pm 1.3$	$68.0 \pm 0.6 \mid \underline{68.1} \pm 1.0$	$74.6 \pm 1.2 \mid \underline{76.6} \pm 2.6$

Table 3: **Multi-source** AL test outcomes with and without hard-to-learn instances removed. Cell scheme to be read as *ablated* | *original*. **Bold** denotes best score for each test set overall and underline best score per setting. At test-time, the labeled set $\mathcal{D}_{train}$ comprises 4K examples.



(a) Original training data (b) Hard-to-learn removed

Figure 3: **Multi-source** AL OOD (MNLI) and ID (SNLI, ANLI, WANLI) learning curves. On the left, the unlabelled pool has been left untreated; on the right, hard-to-learn data was removed from the pool.

wise, i.e., acquiring diverse or uncertain batches does not seem to lead to higher performance.

6 Dataset Cartography for AL Diagnosis

Our findings in both single and multi-source AL settings, for both in-domain and out-of-domain evaluations, showed poor performance of all algorithms (§5). We therefore aim to investigate if the answer for the observed AL failure may lie in the presence of so-called *collective outliers*: examples that models find hard to learn as a result of factors, such as high ambiguity, underspecification, requirement of specialist skills or labelling errors (Han and Kamber, 2000). Collective outliers can be identified through dataset cartography (Swayamdipta et al., 2020), a post-hoc model-based diagnostic tool which plots training examples along a so-called *learnability spectrum*.

6.1 Creating Datamaps

Dataset Cartography assumes that the learnability of examples *relative* to some model can be quantified by leveraging *training dynamics*, where for each example we measure at fixed intervals (1) the mean model *confidence* for the gold-truth label throughout training and (2) the *variability* of this statistic. After gathering these statistics we can plot datasets on a **datamap**: a 2D graph with mean confidence on the Y-axis and confidence variability on the X-axis. The resulting figure enables us to identify how data is distributed along a *learnability spectrum* (see an example in Figure 4).

We construct datamaps by training a model on the entire pool, i.e. 60K examples in total. Every $\frac{1}{2}$ epoch we perform inference on the full training set to get per-example confidence statistics, where the prediction logit corresponding to the gold-truth label serves as a proxy for model confidence. Variability is computed as the standard deviation over the set of confidence measurements. Following Karamcheti et al. (2021), we classify examples along four difficulties via a threshold on the mean confidence value p (Figure 4). We provide more details in the Appendix A.5.

6.2 Strategy Maps for Multi-source AL

We show the datamaps for random sampling and the MCME acquisition functions on the multi-source AL setting in the top row of Figure 4 (see Appendix A.5 for all datamaps). Examining the first datamap, it appears that *randomly sampling from all sources in equal proportions tends to yield predominantly easy-to-learn instances*, with diminishing amounts of ambiguous and hard-to-learn instances. Conversely, we find that MCME acquires considerably more examples with moderate to low confidence, suggesting a *tendency to acquire hard and impossible examples*. Note that these outcomes mirror the findings by Karamcheti et al. (2021) for VQA. Observing the per-difficulty acquisition over time (Figure 4 bottom row), we observe that BALD

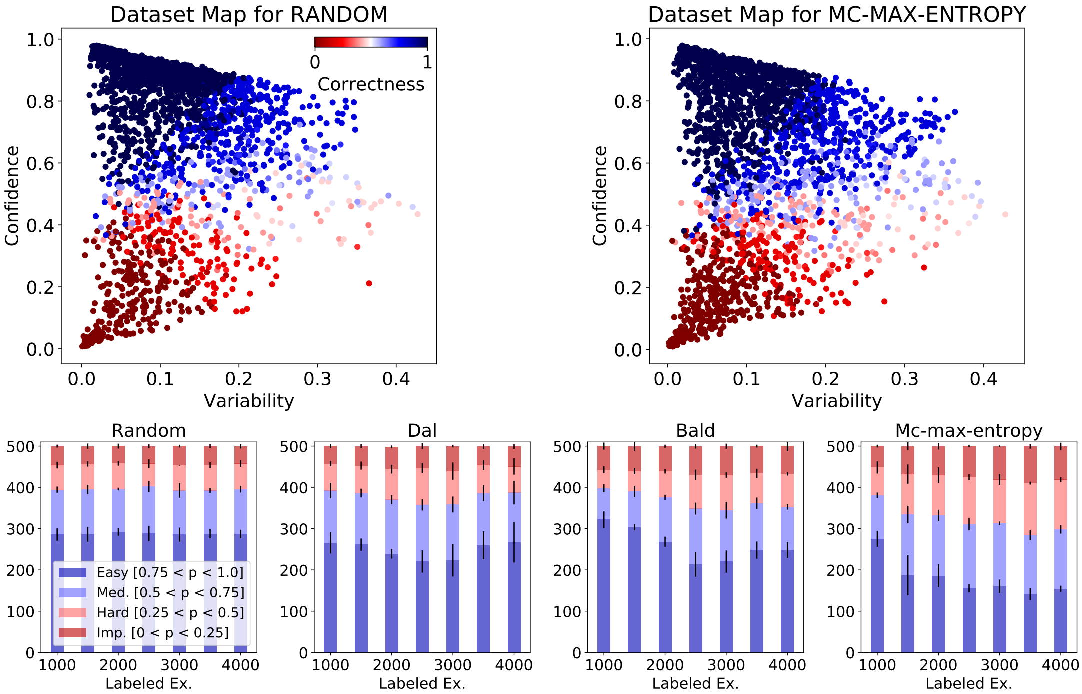


Figure 4: Top row: Strategy maps for RANDOM and MCME for **multi-source** AL. *Correctness* denotes how often the model predicts the correct label. Bottom row: Acquisition by difficulty over time for RANDOM, DAL, BALD and MCME. We provide additional datamap plots in the appendix (Section A.5).

and MCME *initially favour easy examples*, but as more examples are acquired, we observe a *shift towards medium, hard and impossible examples*. One explanation for these trends is that in early phases of AL, models have only been trained on small amounts of data. At this stage, model confidence is still poorly calibrated and consequently confidence values may be noisy proxies for uncertainty. As the training pool grows, model confidence will more accurately reflect example difficulty enabling uncertainty-based strategies to identify difficult data points.

7 Stratified Analysis on Data Difficulty

We begin our analysis of collective outliers by exploring impossible data points in the three NLI datasets we use; SNLI, ANLI and WANLI. We denote as impossible data points those that yield a confidence value in the range $0 \leq p \leq 0.25$ (§6.1). We provide some examples in Table 4 and in the Appendix A.7. We find that impossible examples from SNLI and WANLI are more prone to suffer from label errors and/or often lack a clear correct answer, which may explain their poor learnability. Conversely, we find impossible ANLI examples to exhibit fewer of these issues - we hypothesize that their difficulty follows rather from requirement of advanced inference types, e.g. identifying relevant information from long passages and numerical reasoning about dates of birth and events.

7.1 Examining the effect of training data difficulty

Now that we are able to classify training examples as easy, medium, hard and impossible (§6.2), we proceed to explore the effect of data difficulty on learning and per-source outcomes. In this set of experiments, we aim to answer the research question: *What data would the most beneficial training set consist of, in terms of data difficulty per example?* We conventionally (i.e., non-AL experiment) train RoBERTa-large on training sets of various difficulties. Each training set comprises 4K examples, i.e. the same amount of examples that would be acquired after all rounds of AL. We consider the following combinations of data: **EM**, **EMH**, **MH**, **HI** and **EMHI**, where **E** denotes *easy*, **M** *medium*, **H** *hard* and **I** *impossible* examples.[3]

---

[3] For a given run, examples are sampled from each difficulty in equal proportion. For instance, when training on the *easy-medium* (**EM**) split the training set comprises 2K easy-to-learn instances and 2K medium-to-learn instances.

Src.	Premise	Hypothesis	GT	P	Conf.
SNLI	`A skier in electric green on the edge of a ramp made of metal bars.`	`The skier was on the edge of the ramp.`	N	E	0.976
SNLI	`Man sitting in a beached canoe by a lake.`	`a man is sitting outside`	N	E	0.980
WANLI	`The first principle of art is that art is not a way of life, but a means of life.`	`Art is a way of life.`	E	C	0.944
WANLI	`Some students believe that to achieve their goals they must take the lead.`	`Some students believe that to achieve their goals they must follow the lead.`	E	C	0.630
ANLI	`Marwin Javier González (born March 14, 1989) is a Venezuelan professional baseball infielder with the Houston Astros of Major League Baseball (MLB). Primarily a shortstop, González has appeared at every position except for pitcher and catcher for the Astros.`	`He is in his forties.`	C	N	0.769
ANLI	`The Whitechapel murders were committed in or near the impoverished Whitechapel district in the East End of London between 3 April 1888 and 13 February 1891. At various points some or all of these eleven unsolved murders of women have been ascribed to the notorious unidentified serial killer known as Jack the Ripper.`	`The women killed in the Whitechapel murders were impoverished.`	N	E	0.832

Table 4: Impossible NLI training examples. **GT** denotes the gold truth label; **P** denotes the model prediction and **Conf.** denotes the prediction confidence. We provide more examples per source in the appendix (Section A.7).

Results We provide test results for all combinations of training data difficulty in Table 5. We first observe that models trained only on **HI** data consistently perform the worst, resulting in a drop of up to ~50%(!) points in accuracy compared to the **EM** split for SNLI and ~40% for MNLI. Surprisingly, we find that models trained on **MH** perform worse than splits that include *easy* examples, except for ANLI. Intuitively, this makes sense: of all datasets, ANLI features the most difficult examples, and thus it is plausible that hard-to-learn instances translate to more learning gains than easy ones. The **EMHI** split slightly underperforms relative to the **EM** and **EMH** splits. Otherwise, we do not observe great differences between the latter two splits. In the second part of Table 5, we compute the average performance for easy test sets (SNLI $\cup$ MNLI), hard (ANLI $\cup$ WANLI) and all test sets combined. We can now observe very clear patterns. The more difficult data points we include in the training set, the more the performance on an easy test set drops (AVG-EASY$\downarrow$). Similarly, when testing on a harder test set, the more difficult the training data the better (AVG-HARD$\downarrow$), but without including data points that are characterized as *impossible*. *Overall, we observe a strong correlation between training and test data difficulty, and we conclude that we should train models on data of the same difficulty as the test set, while always removing data points that are impossible to be learned from the model (i.e. collective outliers).*

7.2 Ablating Outliers for AL

Having uncovered the effects of data difficulty on learning outcomes, we now examine *how the presence or absence of hard and impossible instances affects AL strategy success*. Specifically, we repeat our multi-source AL experiment (§5.2) whilst excluding hard and impossible (**HI**) examples from $\mathcal{D}_{pool}$. Employing datamap statistics, we compute the product of an example's mean confidence and its variability for the entire unlabelled pool, after which we exclude the 25% of examples with the smallest products, following Karamcheti et al. (2021). Examples are filtered out for each source dataset separately to preserve equivalence in source sizes.

Results Examining the learning curves (Figure 3, right) and the final test results (Table 3, left), it appears that *excluding **HI** examples particularly affects the performance of uncertainty-based acquisition strategies* (MCME and BALD): across sources, both strategies do consistently *better*. Moreover, for ANLI, MCME clearly outperforms random selection and even achieves a higher accuracy compared to the non-ablated run. Similar results are obtained for the OOD MNLI dataset: *previously, none of the*

	SNLI	ANLI	WANLI	MNLI	AVG-EASY	AVG-HARD	AVG-ALL
EM	**85.95 $\pm$ 0.90**	36.85 $\pm$ 1.05	**68.34 $\pm$ 0.68**	**76.16 $\pm$ 2.32**	**81.06**	52.60	**66.83**
EMH	84.98 $\pm$ 0.56	37.84 $\pm$ 0.79	68.16 $\pm$ 0.66	75.50 $\pm$ 2.00	80.24	53.00	66.62
MH	76.32 $\pm$ 1.04	**39.11 $\pm$ 0.79**	63.56 $\pm$ 1.38	69.82 $\pm$ 6.10	73.07	**54.47**	62.20
HI	33.57 $\pm$ 1.60	34.00 $\pm$ 0.69	45.89 $\pm$ 3.85	32.69 $\pm$ 1.93	33.13	39.95	36.54
EMHI	79.12 $\pm$ 0.80	38.41 $\pm$ 0.57	64.52 $\pm$ 1.38	73.52 $\pm$ 0.07	76.32	51.47	63.89

Table 5: **Multi-source** test outcomes for training data of *varying difficulty*. Row headers denote difficulty splits, with **E** denoting *easy*, **M** *medium*, **H** *hard* and **I** *impossible* examples. Column headers denote the test set. We also compute the average over SNLI+MNLI (easy), ANLI+WANLI (hard) and all 4 datasets, respectively.

*strategies consistently outperformed random, but after ablating **HI** instances both* BALD *and* MCME *are either on par with random or outperform it.* Intuitively, under normal circumstances MCME and BALD tend to acquire more difficult examples than RANDOM or DAL, which typically leads to *poorer* models (§7.1). Therefore, we should expect to see *improvements* when these instances are ablated (i.e., removed).

7.3 Stratified Testing

Having established that some strategies acquire more hard and examples than others (§6.2), training on such examples can both hurt or help generalization (§7.1), and removal of this data can help strategies improve (§7.2), a question arises: Do strategies that predominantly acquire data of a certain difficulty also perform better on *test data* of that difficulty? To investigate this we introduce **stratified testing**: stratifying test outcomes *in terms of difficulty*. Here, we follow the approach as outlined in Section 6.1, but this time training a cartography model on the *test set* to obtain learnability measurements for each *test* example. This enables us to observe how strategies perform across test examples of varying difficulties.

Task	D	RANDOM	MCME	BALD	DAL
SNLI	**E**	95.4 | 95.0	95.0 | 93.3	95.5 | 94.8	93.7 | 94.6
	M	69.0 | 67.7	70.2 | 68.0	70.5 | 69.5	67.8 | 68.2
	H	46.3 | 44.6	48.5 | 46.0	47.1 | 49.2	44.7 | 45.8
	I	17.7 | 17.2	20.3 | 22.1	20.7 | 22.2	18.4 | 17.5
ANLI	**E**	81.4 | 81.7	77.0 | 79.2	82.1 | 73.8	81.8 | 79.9
	M	62.2 | 62.1	61.7 | 57.8	61.1 | 57.1	63.1 | 59.8
	H	39.3 | 40.2	42.9 | 41.6	39.2 | 39.7	39.0 | 38.5
	I	14.4 | 14.2	15.7 | 17.0	12.6 | 14.5	14.2 | 15.0
WANLI	**E**	94.3 | 92.9	92.7 | 91.2	95.2 | 90.9	94.1 | 93.1
	M	72.7 | 69.6	71.4 | 69.9	72.7 | 67.9	72.3 | 73.8
	H	41.9 | 40.4	43.1 | 44.2	42.4 | 40.7	40.6 | 41.0
	I	12.7 | 14.4	14.0 | 16.6	11.9 | 14.7	13.9 | 14.6
MNLI	**E**	93.3 | 93.6	94.3 | 92.8	94.4 | 92.4	93.4 | 93.8
	M	72.6 | 72.9	78.6 | 74.7	75.9 | 72.8	74.7 | 78.1
	H	41.8 | 44.5	51.9 | 49.3	45.7 | 46.7	46.1 | 48.3
	I	12.8 | 15.7	20.9 | 19.5	13.1 | 16.9	12.6 | 18.5
All	**E**	91.0 | 90.8	89.8 | 89.1	91.8 | 88.0	90.7 | 90.4
	M	69.3 | 68.1	70.5 | 67.6	70.0 | 66.8	69.4 | 69.9
	H	42.3 | 42.4	46.5 | 45.3	43.6 | 44.1	42.6 | 43.4
	I	14.4 | 15.3	17.7 | 18.8	14.6 | 17.1	14.8 | 16.4

Table 6: **Multi-source** strategy comparison (accuracy) for difficulty-stratified test outcomes between original training set and training set with hard-to-learn examples excluded. Test examples (D) belong to Easy (**E**), Medium (**M**), Hard (**H**) and Impossible (**I**) sets. Cell scheme to be read as *ablated* | *original*.

Results Table 6 shows the AL results for stratified testing for both *ablated* (i.e., **HI** removed) and the *original* setting. We observe in the pre-ablation experiments (right side) that RANDOM and DAL tend to do better on easy and medium data relative to MCME and BALD, while conversely they underperform when tested on hard and impossible examples. The same pattern also occurs also in the OOD setting (MNLI). In the ablation setting (left side), we find that across all sources, **HI** examples tends to yield a performance drop on hard and impossible examples for all strategies. Corroborating our previous analysis (§7.1), our findings here suggest that *it is essential to have data of the same difficulty in both unlabelled pool and in test set*. Next, we find that under ablation, RANDOM does moderately better on easy and medium, while DAL shows marginally lower test outcomes on medium and hard examples. For the uncertainty-based strategies, ablation tends to yield improvements across all difficulties, with MCME outperforming BALD on hard examples. We hypothesize that with **HI** instances absent, uncertainty-based methods select the "next-best" available data: medium examples which offer greater learning gains than easy ones. Overall, we find that post-ablation, BALD and MCME outperform both RANDOM and DAL across most difficulty splits.

8 Conclusion and Future Work

Our work highlights the challenge of successfully applying AL when the (training) pool comprises several sources which span various domains in the task of NLI. Similar to Karamcheti et al. (2021), we show that uncertainty-based strategies, such as MCME and BALD, perform poorly (§5) due to acquisition of collective outliers which impede successful learning (§6.2). However, these strategies recover when outliers are removed (§7.2). Practically, this suggests that uncertainty-based methods may fare well under more carefully curated datasets and labelling schemes, while alternative strategies (e.g. diversity-based) may be preferable in cases with poorer quality guarantees (e.g. data collection with limited budget for annotation verification). Next, we find that performance outcomes between strategies differ for test data of various difficulties (§7.3). On the one hand, this complicates strategy selection: it is unclear whether a strategy that performs well on hard data but poorer on easy data is preferable to a strategy with opposite properties. On the other hand, knowing which strategies work well for test data of a certain difficulty may be advantageous when the difficulty of the test set is known, in out-of-domain settings (Lalor et al., 2018).

Lastly, in contrast with Karamcheti et al. (2021) and Zhang and Plank (2021), we have shown that cases exist in which training examples in the hard-to-learn region do not hamper learning but are in fact pivotal for achieving good generalization (§7.1). Consequently, there may be value in refining existing cartography-based methods such that they can discriminate between useful and harmful hard-to-learn data. More broadly, our findings underscore the potential of understanding these phenomena for other NLP tasks and datasets.

Acknowledgements

We thank Dieuwke Hupkes for helping during the initial stages of this work. The presentation of this paper was financially supported by the Amsterdam ELLIS Unit and Qualcomm. We also thank SURF for the support in using the Lisa Compute Cluster. Katerina is supported by Amazon through the Alexa Fellowship scheme.

Limitations

In our work, we have shown that standard AL algorithms struggle to outperform random selection in some datasets for the task of NLI, which is in fact rather surprising as a large body of work has shown positive AL results in a wide spectrum of NLP tasks. Still, we are not the first to show negative AL results in NLP, as Lowell et al. (2019); Karamcheti et al. (2021); Kees et al. (2021) have shown similar problematic behavior in the cases of text classification, visual question answering and argument mining, respectively.

More broadly, while AL outcomes have shown to not always reliably generalize across models and tasks (Lowell et al., 2019), we recognise that in our work several experimental conditions remain underexplored which warrant further attention. First, this work mostly examined point-wise acquisition functions; it remains unclear whether our outcomes hold for *batch-wise* functions. Similarly, it is unknown how the chosen model-in-the-loop affects strategy outcomes. We use RoBERTa-Large, a comparatively powerful large language model. Some authors hypothesize that as such models are already able to achieve great performance even with randomly labeled data, this could significantly raise the bar for AL algorithms to yield substantial performance improvements *on top* of a random selection baseline (Ein-Dor et al., 2020). Combined with the relative homogeneity of acquired batches between sources in terms of input diversity, this would explain why strategies tend to do similar across the board at test-time both in the presence and absence of hard-to-learn examples. Another factor tying into this is that small differences between results may be connected to the so-called inverse cold-start problem. This problem states that there may be cases where the initial seed set is too large, leaving comparatively little room for substantial improvements in sample efficiency. We leave further exploration of these variables to future work.

Another area within AL research which warrants further examination concerns the evaluation on out-of-domain datasets of which no data is present in the pool of unlabelled training data. Particularly, the majority of work typically assumes that acquisition is target-agnostic, i.e., target validation and test sets are assumed to be decoupled entirely from the acquisition process. This can be problematic as for different target sets, different subsets of the unlabelled training data may yield the best possible performance. Consequently, performance outcomes on *some* out-of-domain test data may not necessarily pose as reliable signals for determining the best strategy, for if a different target set had been chosen, a previously 'poor' performing strategy may suddenly achieve the best result. Despite this being a clear shortcoming of the existing AL toolkit, it remains an understudied area within AL research.

While this problem may be partially alleviated by evaluating strategies on a large and diverse array of target sets, the issue remains that current acquisition functions do not acquire data *with respect to the target set*. This problem becomes even more apparent in a *multi-source* setting, where depending on the target set at hand, acquiring data from the appropriate source(s) may be pivotal to achieve good performance. In such cases, we may want to explicitly regularise the acquisition process towards target-relevant training data. A small body of work has examined ways in which such target-aware acquisition could be formalized - most noticeably in the work of Kirsch et al. (2021), who introduce several methods to perform test-distribution aware active learning. While examination of such methods lies outside the scope of this work, we recognize its potential for future work on multi-source AL.

References

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Aditi Chaudhary, Antonios Anastasopoulos, Zaid Sheikh, and Graham Neubig. 2021. Reducing confusion in active learning for part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 9:1–16.

Tongfei Chen, Zhengping Jiang, Adam Poliak, Keisuke Sakaguchi, and Benjamin Van Durme. 2020. Uncertain natural language inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8772–8779, Online. Association for Computational Linguistics.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4(1):129–145.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active learning for BERT: An empirical study. In *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 7949–7962.

Yarin Gal and Zoubin Ghahramani. 2015. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. *33rd International Conference on Machine Learning, ICML 2016*, 3:1651–1660.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep Bayesian Active Learning with Image Data. *34th International Conference on Machine Learning, ICML 2017*, 3:1923–1932.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are We Modeling the Task or the Annotator? An Investigation of Annotator Bias in Natural Language Understanding Datasets. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 1161–1166.

Amirata Ghorbani, James Zou, and Andre Esteva. 2021. Data Shapley Valuation for Efficient Batch Active Learning. *arXiv preprint 2104.08312*.

Daniel Gissin and Shai Shalev-Shwartz. 2019. Discriminative Active Learning. *arXiv preprint arXiv:1907.06347*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation Artifacts in Natural Language Inference Data. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2:107–112.

Jiawei Han and Micheline Kamber. 2000. *Data Mining: Concepts and Techniques*. Morgan Kaufmann.

Rui He, Shengcai Liu, Shan He, and Ke Tang. 2021. Multi-Domain Active Learning: A Comparative Study. *arXiv preprint 2106.13516*.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian Active Learning for Classification and Preference Learning. *CoRR abs/1112.5745*.

Jiaji Huang, Rewon Child, Vinay Rao, Hairong Liu, Sanjeev Satheesh, and Adam Coates. 2016. Active Learning for Speech Recognition: the Power of Gradients. *arXiv preprint arXiv:1612.03226. Published as a workshop paper at NIPS 2016*.

Siddharth Karamcheti, Ranjay Krishna, Li Fei-Fei, and Christopher D. Manning. 2021. Mind Your Outliers! Investigating the Negative Impact of Outliers on Active Learning for Visual Question Answering. pages 7265–7281.

Nataliia Kees, Michael Fromm, Evgeniy Faerman, and Thomas Seidl. 2021. Active learning for argument strength estimation. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 144–150, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hannah Kirk, Bertie Vidgen, and Scott Hale. 2022. Is more data better? re-thinking the importance of efficiency in abusive language detection with transformers-based active learning. In *Proceedings of the Third Workshop on Threat, Aggression and Cyberbullying (TRAC 2022)*, pages 52–61, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Andreas Kirsch and Yarin Gal. 2021. Powerevaluationbald: Efficient evaluation-oriented deep (bayesian) active learning with stochastic acquisition functions. *CoRR*, abs/2101.03552.

Andreas Kirsch, Tom Rainforth, and Yarin Gal. 2021. Test Distribution-Aware Active Learning: A Principled Approach Against Distribution Shift and Outliers. *arXiv preprint 2106.11719*.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, Iroro Orife, Kelechi Ogueji, Andre Niyongabo Rubungo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, Tapiwanashe Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. 2022. Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

John P. Lalor, Hao Wu, Tsendsuren Munkhdalai, and Hong Yu. 2018. Understanding deep learning performance through an examination of test set difficulty: A psychometric case study. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 4711–4716.

Alisa Liu, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. 2022. WANLI: Worker and AI Collaboration for Natural Language Inference Dataset Creation. *arXiv preprint arXiv:2201.05955*.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning to Actively Learn Neural Machine Translation. *CoNLL 2018 - 22nd Conference on Computational Natural Language Learning, Proceedings*, pages 334–344.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. 2020. Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, 2020-December.

Shayne Longpre, Julia Reisler, Edward Greg Huang, Yi Lu, Andrew Frank, Nikhil Ramesh, and Chris DuBois. 2022. Active Learning Over Multiple Domains in Natural Language Tasks. *arXiv preprint arXiv:2202.00254*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 21–30.

Katerina Margatina, Loic Barrault, and Nikolaos Aletras. 2022. On the importance of effectively adapting pretrained language models for active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 825–836, Dublin, Ireland. Association for Computational Linguistics.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial NLI: A New Benchmark for Natural Language Understanding. pages 4885–4901.

Yixin Nie, Xiang Zhou, and Mohit Bansal. 2020. What Can We Learn from Collective Human Opinions on Natural Language Inference Data? *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 9131–9143.

Álvaro Peris and Francisco Casacuberta. 2018. Active Learning for Interactive Neural Machine Translation of Data Streams. *CoNLL 2018 - 22nd Conference on Computational Natural Language Learning, Proceedings*, pages 151–160.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin van Durme. 2018. Hypothesis Only Baselines in Natural Language Inference. *NAACL HLT 2018 - Lexical and Computational Semantics, SEM 2018, Proceedings of the 7th Conference*, pages 180–191.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4058–4068, Hong Kong, China. Association for Computational Linguistics.

Christopher Schröder, Andreas Niekler, and Martin Potthast. 2022. Revisiting uncertainty-based query strategies for active learning with transformers. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2194–2203, Dublin, Ireland. Association for Computational Linguistics.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*.

Burr Settles. 2010. Active Learning Literature Survey. *Machine Learning*, 15(2):201–221.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and Bayesian uncertainty estimates. *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 1698–1712.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep Active Learning for Named Entity Recognition. *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP 2017 at the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 252–256.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian Active Learning for Natural Language Processing: Results of a Large-Scale Empirical Study. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 2904–2909.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 9275–9293.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2017. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:1112–1122.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv preprint arXiv:1910.03771*.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.

Michelle Yuan, Patrick Xia, Chandler May, Benjamin Van Durme, and Jordan Boyd-Graber. 2022. Adapting coreference resolution models through active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7533–7549, Dublin, Ireland. Association for Computational Linguistics.

Mike Zhang and Barbara Plank. 2021. Cartography Active Learning. *Findings of the Association for Computational Linguistics, Findings of ACL: EMNLP 2021*, pages 395–406.

Ye Zhang, Matthew Lease, and Byron C. Wallace. 2016. Active Discriminative Text Representation Learning. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 3386–3392.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A Survey of Active Learning for Natural Language Processing. *EMNLP 2022 - Conference on Empirical Methods in Natural Language Processing, Proceedings*.

Yuekai Zhao, Haoran Zhang, Shuchang Zhou, and Zhihua Zhang. 2020. Active Learning Approaches to Enhancing Neural Machine Translation. *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020*, pages 1796–1806.

Fedor Zhdanov. 2019. Diverse mini-batch Active Learning. *arXiv preprint 1901.05954v1*.

A Appendix

A.1 Analysis metrics

Following standard practice in active learning literature (Zhdanov, 2019; Yuan et al., 2020; Ein-Dor et al., 2020; Margatina et al., 2021) we profile datasets acquired by strategies via acquisition metrics. Concretely, we consider the *input diversity* and *output uncertainty* metrics.

Input Diversity Input diversity quantifies the diversity of acquired sets in the input space, meaning that it operates directly on the raw input passages. We follow (Yuan et al., 2020) and measure input diversity as the Jaccard similarity $\mathcal{J}$ between the set of tokens from the acquired training set $\mathcal{D}_{train}$, $\mathcal{V}$, and the set of tokens from the *remainder* of the unlabelled pool $\mathcal{D}_{pool}$, $\mathcal{V}\prime$, which yields:

$$\mathcal{J}(\mathcal{V}, \mathcal{V}\prime) = \frac{\mid \mathcal{V} \cap \mathcal{V}\prime \mid}{\mid \mathcal{V} \cup \mathcal{V}\prime \mid}$$

This function assigns *high* diversity to strategies acquiring samples with *high* token overlap with the unlabelled pool, and vice versa.

Output Uncertainty To approximate the output uncertainty of an acquired training set $\mathcal{D}_{train}$ for a given strategy, we train RoBERTa-large to convergence on the entire 60K training set. We then use the trained model to perform inference over $\mathcal{D}_{train}$. Following (Yuan et al., 2020), the output uncertainty of each strategy is computed as the mean predictive entropy over all examples x in its acquired set $\mathcal{D}_{train}$:

$$-\frac{1}{|\mathcal{D}_{train}|} \sum_{x \in \mathcal{D}_{train}} \sum_{c=1}^{C} p(y = c \mid x) \log p(y = c \mid x)$$

A.2 Datasets

As mentioned in the paper (§4), we perform experiments on Natural Language Inference (NLI), a popular text classification task to gauge a model's natural language understanding (Bowman et al., 2015; Williams et al., 2017). We recognize that NLI is somewhat artificial by nature - making it of lesser practical relevance for real-life active learning scenarios. However, recent work has sought to address shortcomings of existing NLI benchmarks such as SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2017). This has lead to the emergence of novel approaches to dataset-creation such as Dynamic Adversarial Data Collection (ANLI, (Nie et al., 2019)) and worker-and-AI-collaboration (WANLI, (Liu et al., 2022)). As we seek to investigate how characteristics of data gathered through such alternative protocols may affect acquisition performance in a multi-source active learning setting, using NLI for our experiments is a natural choice. We construct the unlabelled pool from three distinct datasets: SNLI, ANLI and WANLI. Next, we consider the Multi Natural Language Inference (MNLI) corpus (Williams et al., 2017) as an out-of-domain challenge set to evaluate the transferability of actively acquired training sets. We provide datasets statistics in Table 7.

A.3 Training details & Reproducibility

We use RoBERTa-large (Liu et al., 2020) from Huggingface (Wolf et al., 2019) as our model-in-the-loop and optimize with AdamW (Loshchilov and Hutter, 2018), with a learning rate of $2e-6$ and a batch size of 32. We use Dropout with $p = 0.3$. Hyperparameters were chosen following a manual tuning process, evaluating models on classification accuracy. For the BALD and MCME strategies we use 4 Monte Carlo Dropout samples. Our framework is implemented in PyTorch Lightning; transformers are implemented using the. All experiments were ran on a single NVIDIA Titan RTX GPU. Trialling all acquisition functions for 7 rounds of active learning (assuming experiments are ran in series), for a single seed, requires approximately 21 hours of compute. See Table 8 for per-strategy runtimes.

A.4 Detailed Results

Acquisition Factor For the multi-source experiments we also plot the *acquisition factors*: a source-strategy-specific statistic which indicates how much data of a given source is acquired by some strategy, normalized by the share of that source in the unlabelled pool at the time of that acquisition round. This statistic is useful to interpret how much a strategy tends to acquisition of one source relative to others.

We normalize to correct for the effect that acquiring e.g. mostly SNLI data in an early round causes it to take up a relatively smaller share in the unlabelled pool in the next round, and by the simple consequence of there being fewer SNLI examples, they may have a lower likelihood to be acquired in those future rounds compared to examples from other sources.. As this could distort

Label Distributions

Source	Train				Val				Test			
	N	E	C	**Size**	N	E	C	**Size**	N	E	C	**Size**
SNLI	183.4K	182.7K	183.1K	549.5K	3.2K	3.3K	3.3K	9.8K	3.2K	3.4K	3.2K	9.8K
ANLI	61.7K	46.7K	37.4K	146K	0.7K	0.7K	0.7K	2.2K	0.7K	0.7K	0.7K	2.2K
WANLI	48.8K	39.1K	14.4K	103K	1.2K	0.9K	0.4K	2.5K	1.2K	0.9K	0.4K	2.5K
MNLI-m*	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	1.6K	1.7K	1.6K	4.9K

Table 7: Statistics for the used datasets. N = Neutral, E = Entailment, C = Contradiction. For MNLI, only a held-out test set was used, and thus statistics for the training and dev set are omitted. Since WANLI only has a train and test set, we split its test set in two equally sized subsets and use one half as our validation set.

	RANDOM	DAL	BALD	MCME
Runtime	190 ± 19	269 ± 15	350 ± 62	450 ± 90

Table 8: Total runtime per strategy in minutes. Scheme follows *mean* ± *standard error* format.

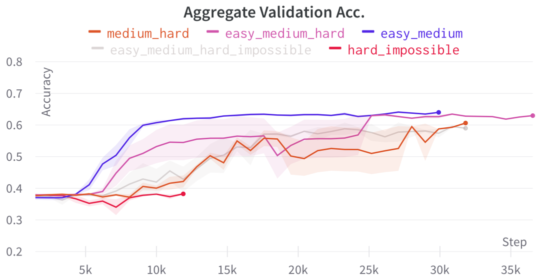


Figure 5: Aggregate validation accuracy for models trained on data of various difficulties. Training data difficulty appears to affect convergence speed: models trained on easier training sets converge earlier, whilst inclusion of hard and impossible models requires more steps to converge. In our experiments, models trained only on hard/impossible examples never surpass at-chance performance and suffer from model failure more often.

our impressions of the extent to which strategies acquire examples from different sources, we ideally want to correct for this.

We thus compute the acquisition factor for a given round as (1) the amount of source examples that were *actually* acquired by the strategy for that round, *divided* by (2) the amount of source examples that *would* be acquired under *random* sampling from the unlabelled pool at that point. For instance, if BALD has an acquisition factor of > 1 for SNLI, it means that it acquired more SNLI examples from the unlabeled pool than it *would* have under random sampling. Conversely, the Random sampling baseline will always have consistent acquisition factors of 1, since the quantities in the numerator and denominator will be approximately the same: random selection always acquire as much from a source as it would under random selection. See Figure 6 for a graphical explanation.[4]

A.5 Dataset Cartography

As mentioned in Section 6.1, we train a RoBERTa-large model on a training set comprised of the entire unlabelled pool of training examples, i.e. 60K examples in total.[5] Every $\frac{1}{2}$ epoch we perform inference on the full training set to get per-example confidence statistics, where the prediction logit corresponding to the gold-truth label serves as a proxy for model confidence. Variability is computed as the standard deviation over the set of confidence measurements. We stop training after 3 epochs or after no improvement in validation accuracy on the aggregate of validation sets was observed. Following Karamcheti et al. (2021), we classify examples along four difficulties via a threshold on the mean confidence value p.

Note on model discrepancy When studying the strategy maps, it is instructive to note that there exists a discrepancy between models which may distort the truthfulness of learnability measurements. That is, the cartography model that was used to obtain confidence/variability measurements

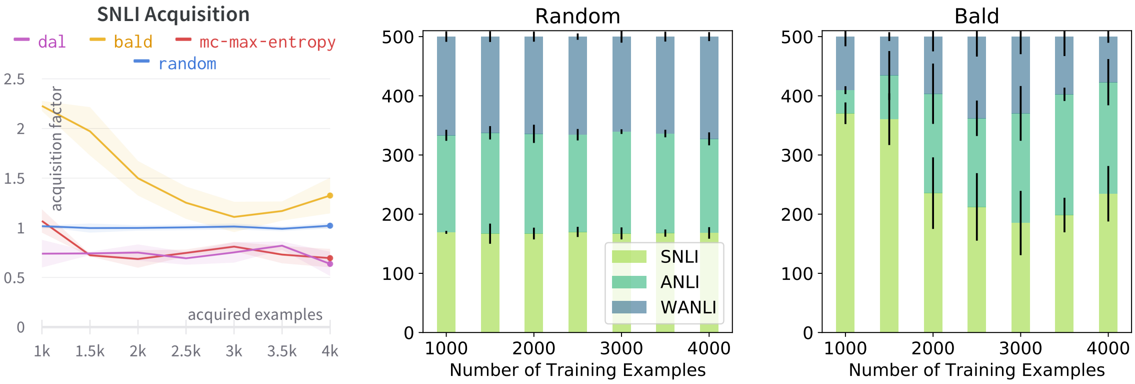
---

[4] Please note that these figures solely serve to support the textual explanation and should not be regarded as results otherwise.

[5] The AL experiments are simulations, where we have the ground truth labels for the data of the pool, but we consider them unlabelled. So in this case, we use the original labelled dataset to train the cartography model.

Task	Strategy	I-Div.	Unc.	N	E	C
SNLI	RANDOM	0.259 ± 0.002	0.051 ± 0.002	0.34	0.33	0.33
	DAL	**0.267** ± 0.002	0.067 ± 0.002	0.38	0.29	0.33
	BALD	0.249 ± 0.004	0.072 ± 0.005	0.32	0.31	0.37
	MCME	0.261 ± 0.004	**0.093** ± 0.005	0.32	0.34	0.34
WANLI	RANDOM	**0.324** ± 0.005	0.208 ± 0.012	0.47	0.38	0.15
	DAL	0.321 ± 0.003	0.223 ± 0.008	0.49	0.38	0.13
	BALD	0.317 ± 0.001	0.213 ± 0.005	0.40	0.39	0.21
	MCME	0.318 ± 0.002	**0.229** ± 0.009	0.38	0.35	0.27
ANLI	RANDOM	**0.492** ± 0.003	0.082 ± 0.004	0.43	0.32	0.26
	DAL	0.467 ± 0.008	0.084 ± 0.004	0.47	0.3	0.23
	BALD	0.478 ± 0.007	0.094 ± 0.003	0.39	0.30	0.31
	MCME	0.491 ± 0.007	**0.110** ± 0.004	0.37	0.31	0.31
Multi	RANDOM	0.276 ± 0.003	0.405 ± 0.012	0.41	0.34	0.25
	DAL	0.276 ± 0.004	0.463 ± 0.018	0.47	0.34	0.19
	BALD	0.220 ± 0.012	0.445 ± 0.025	0.34	0.32	0.35
	MCME	**0.323** ± 0.004	**0.556** ± 0.011	0.4	0.3	0.3

Table 9: Profiling strategy acquisitions for **single-source** and **multi-source** active learning in terms of mean input diversity (**I-Div.**), mean uncertainty (**Unc.**) and class distributions (**N**: *neutral*, **E**: *entailment*, **C**: *contradiction*) across seeds, following a *mean* ± *std* cell scheme. Each strategy is profiled *after* all rounds of active learning: $|\mathcal{D}_{train}| = 4K$.



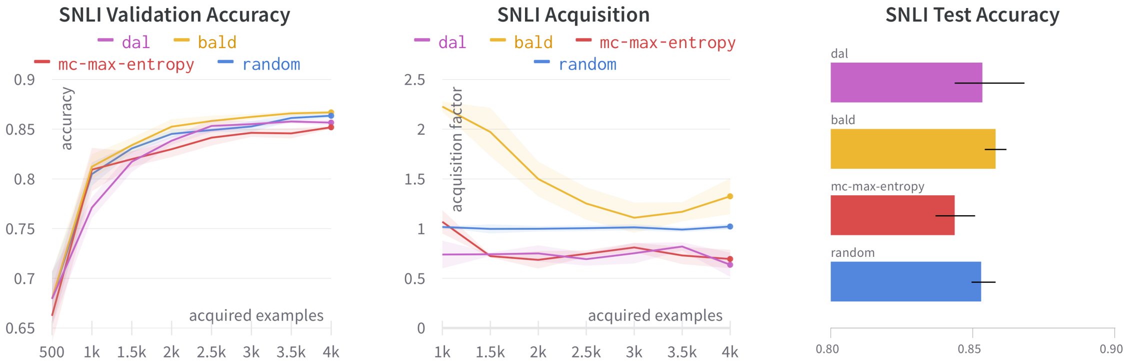
(a) SNLI Acquisition factors

(b) Acquisitions by source

(c) Acquisitions by source

Figure 6: Acquisition factor explained. By design, random selection acquires a random sample of the unlabelled pool. Since the pool is initialized with even amounts of each source, random consistently acquires a uniform sample over sources, as can be seen in **(b)**. Consequently, the SNLI acquisition factor for Random steadily hovers around 1 in **(a)**. Conversely, BALD initially acquires mostly SNLI examples, as reflected by its high initial acquisition factor. This corresponds to its absolute acquisition statistics in **(c)**, where we note that initially BALD acquires predominantly SNLI.

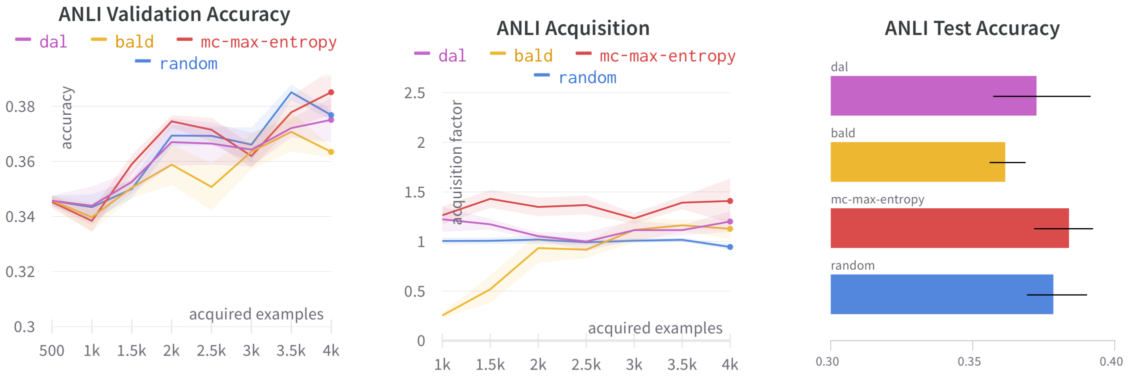
for datamap construction was trained on the entire pool of 60K examples, whereas the model used during AL is exposed to at most 4K examples after completing all rounds of acquisition. Consequently, examples which were found to be easy by the cartography model are likely to be substantially more difficult for the AL model. As datamaps are inherently model-based, the generated strategy maps should be interpreted as skewed estimates of the true learnability spectra. While we recognize the potential value of preserving scale equivalence when using dataset cartography for analysis, we choose to follow Karamcheti et al. (2021) and intend to employ datamaps as a *post-hoc* diagnostic tool. In other words, we are interested in examining example learnability in an *absolute* sense (i.e. with respect to the entire pool) rather than a *relative* sense (with respect to the acquired data). Here,



(a) SNLI Val. Acc.

(b) SNLI Acquisition

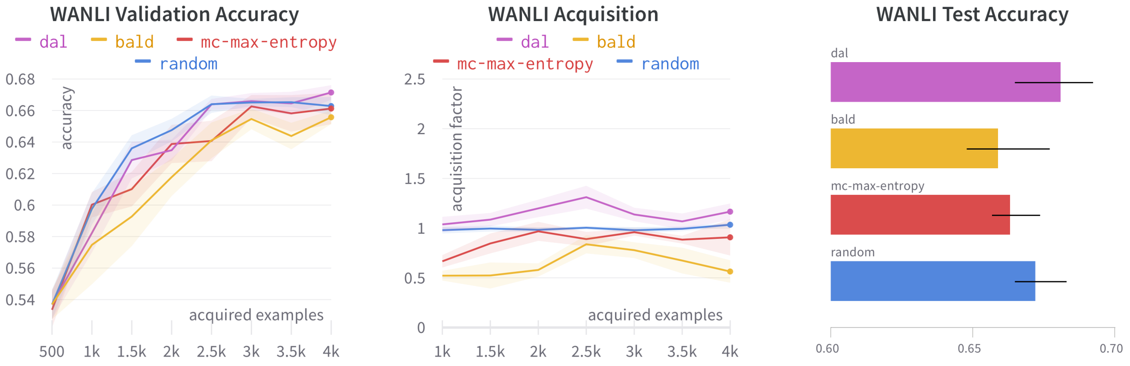
(c) SNLI Test Acc.



(d) ANLI Val. Acc.

(e) ANLI Acquisition

(f) ANLI Test Acc.



(g) WANLI Val. Acc.

(h) WANLI Acquisition

(i) WANLI Test Acc.

Figure 7: In-domain multi-source Active Learning. The left-most graph represents the *learning curve* per strategy, plotting the validation performance as more labelled data is acquired. The middle graph plots the *acquisition factor* per source: how much a strategy acquires of a source, normalized by the share that source takes up in the wider data pool. Shaded regions indicate standard errors. Active learning fails to consistently beat random.

we also note that a model trained on a larger set of examples will likely more accurately reflect the true difficulty of examples. This is important if we want to draw new insights on the learnability of NLI datasets in a wider sense.

A.6 Impact of data difficulty on training time

We find that models quickly learn simple examples while requiring more iterations for complex ones (See Figure 5). A similar outcome was observed by Lalor et al. (2018) who employ item response theory for data difficulty estimation - interestingly however, their difficulty parameters are estimated via a human population, while our difficulty es-

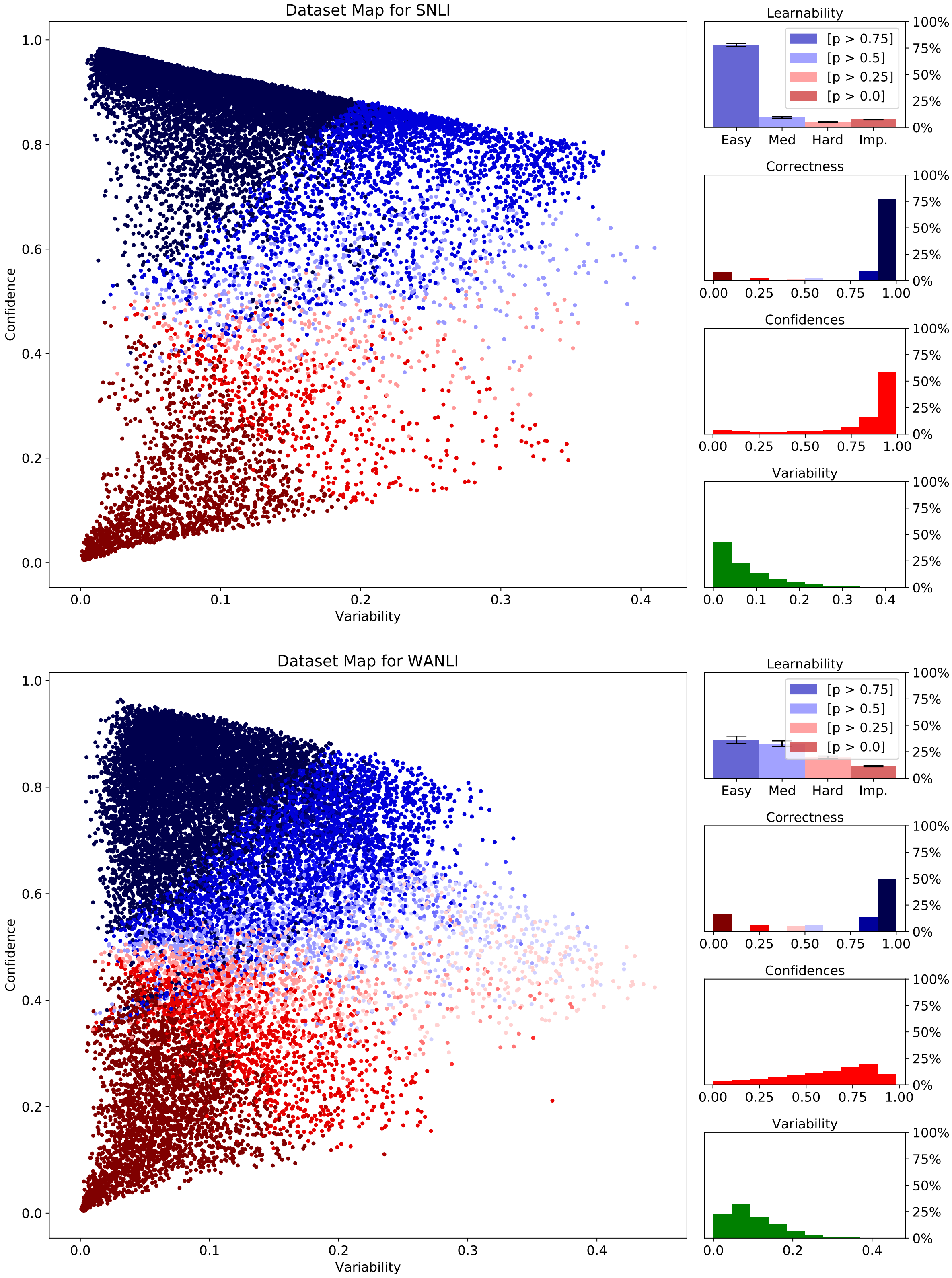


Figure 8: Dataset Maps for SNLI and WANLI training data. The majority of SNLI examples lie in the easy-to-learn region. For WANLI, datapoints are more evenly distributed across the learnability spectrum and exhibit greater variability.

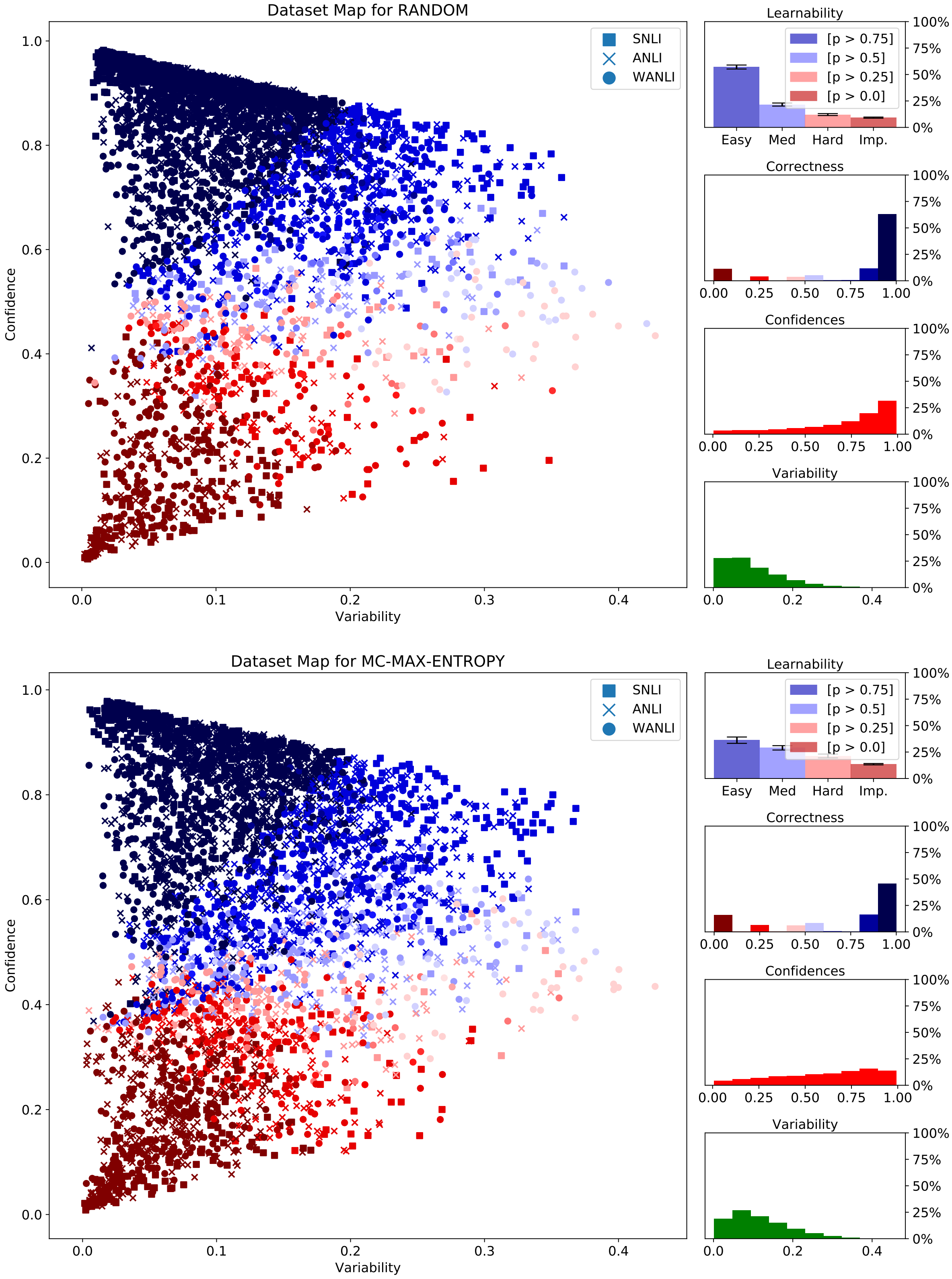
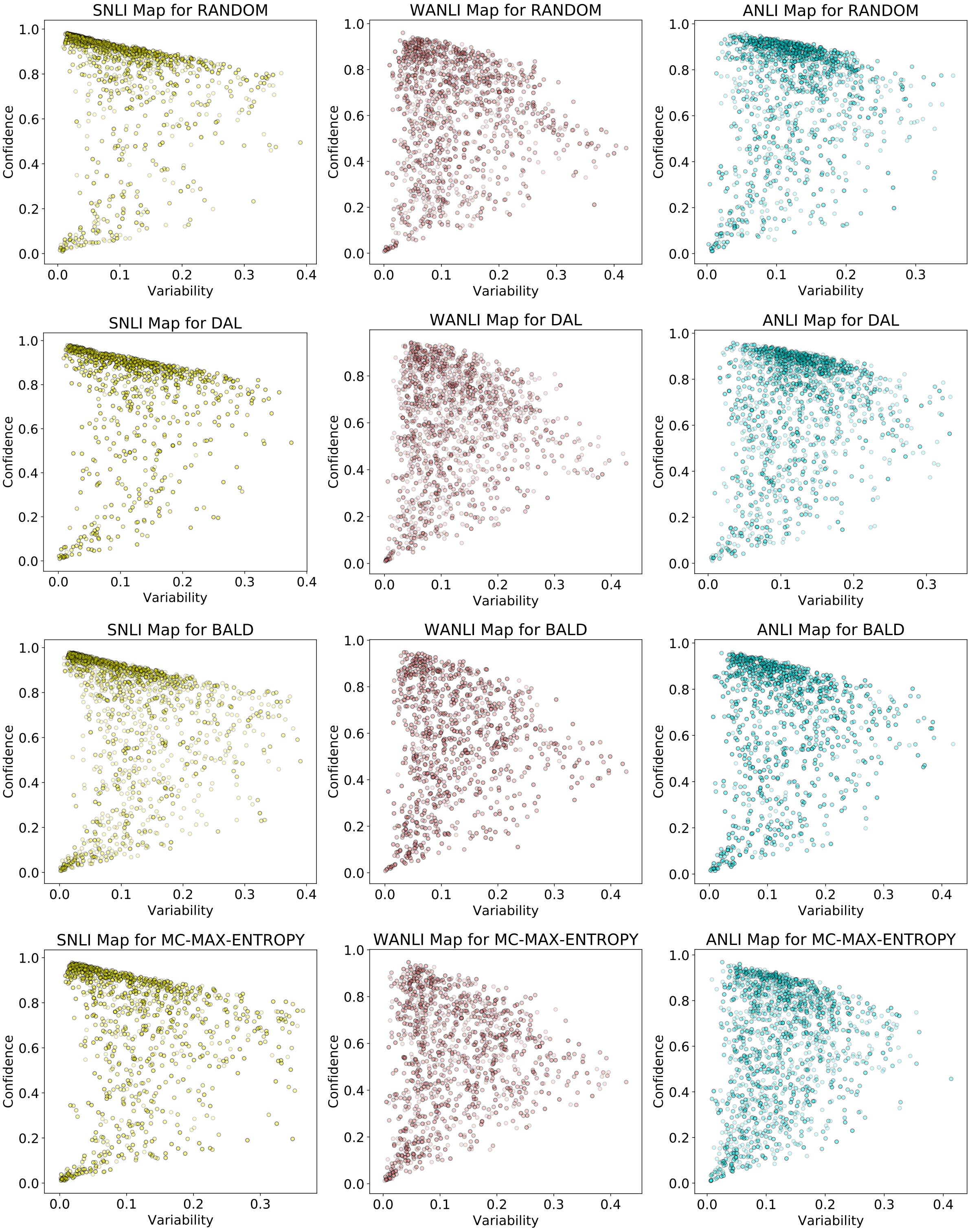


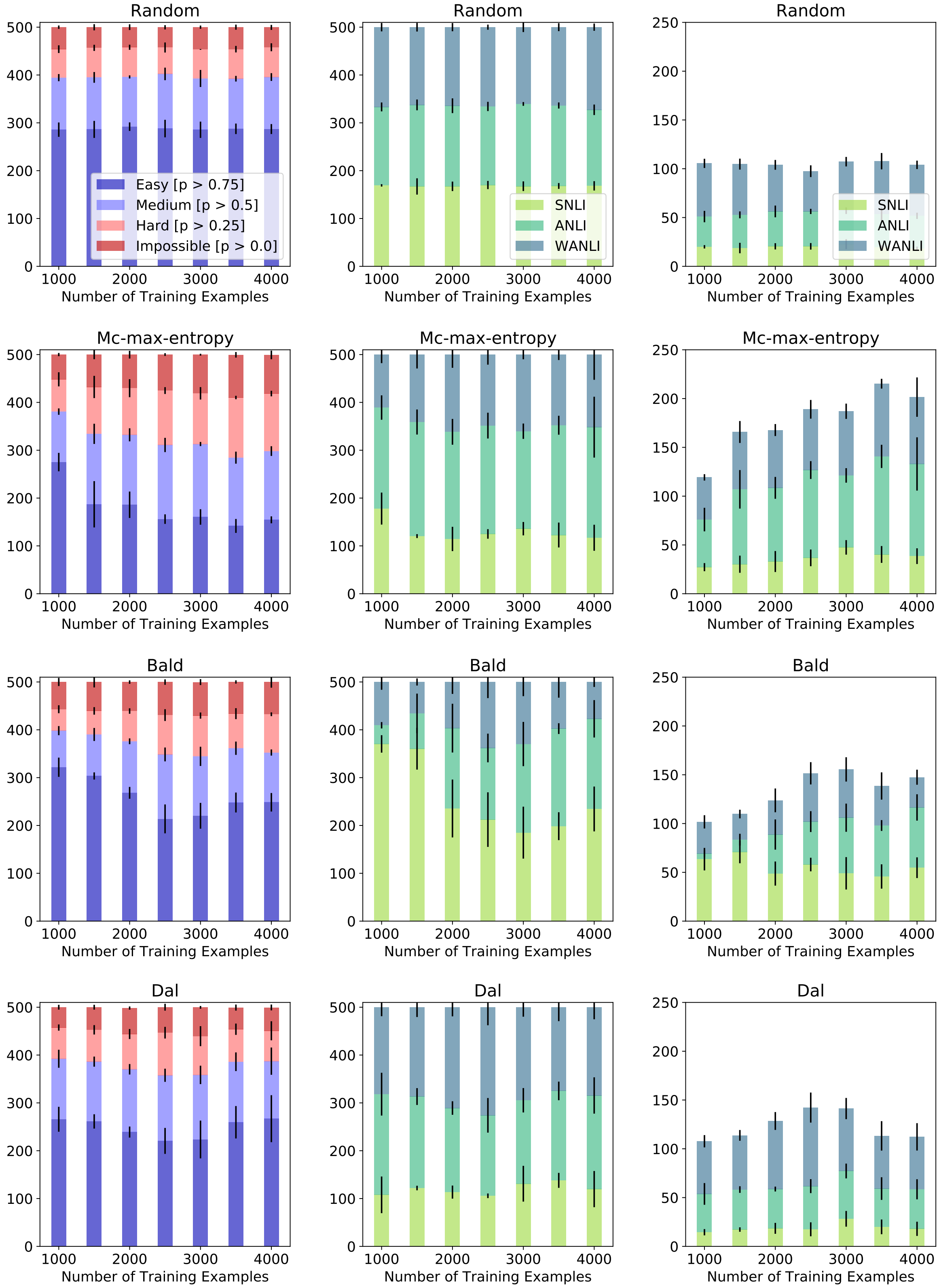
Figure 9: Strategy maps for the Random and MC-Max-entropy strategies for **multi-source** active learning. The latter acquires considerably more hard-to-learn instances.



SNLI Map for RANDOM
WANLI Map for RANDOM
ANLI Map for RANDOM
SNLI Map for DAL
WANLI Map for DAL
ANLI Map for DAL
SNLI Map for BALD
WANLI Map for BALD
ANLI Map for BALD
SNLI Map for MC-MAX-ENTROPY
WANLI Map for MC-MAX-ENTROPY
ANLI Map for MC-MAX-ENTROPY

(a) Strategy maps for SNLI
(b) Strategy maps for WANLI
(c) Strategy maps for ANLI

Figure 10: Strategy maps for **multi-source** active learning, plotted per source.



(a) Acquisition by learnability (b) Acquisition per source (c) Hard & Imp. per source

Figure 11: Profiling acquisitions for strategies over time. Y-axis denotes amount of acquired examples per round. The right graph plots the acquisition of hard and impossible examples. Dark bars indicate standard deviations across seeds. MCM-Entropy and BALD acquire notably more hard and impossible instances relative to random sampling.

Task	Diff.	**Random**	**MCME**	**BALD**	**DAL**
SNLI	**E**	$95.4 \pm 0.9 \mid 95.0 \pm 0.4$	$95.0 \pm 0.5 \mid 93.3 \pm 0.7$	$95.5 \pm 1.0 \mid 94.8 \pm 0.3$	$93.7 \pm 1.2 \mid 94.6 \pm 0.8$
	M	$69.0 \pm 2.6 \mid 67.7 \pm 1.2$	$70.2 \pm 1.6 \mid 68.0 \pm 1.2$	$70.5 \pm 2.6 \mid 69.5 \pm 1.2$	$67.8 \pm 2.4 \mid 68.2 \pm 2.5$
	H	$46.3 \pm 3.5 \mid 44.6 \pm 1.9$	$48.5 \pm 1.6 \mid 46.0 \pm 2.2$	$47.1 \pm 3.2 \mid 49.2 \pm 2.9$	$44.7 \pm 2.2 \mid 45.8 \pm 3.9$
	I	$17.7 \pm 1.7 \mid 17.2 \pm 1.7$	$20.3 \pm 2.4 \mid 22.1 \pm 0.9$	$20.7 \pm 1.7 \mid 22.2 \pm 0.4$	$18.4 \pm 2.0 \mid 17.5 \pm 2.2$
ANLI	**E**	$81.4 \pm 4.9 \mid 81.7 \pm 4.3$	$77.0 \pm 4.1 \mid 79.2 \pm 1.4$	$82.1 \pm 3.1 \mid 73.8 \pm 2.1$	$81.8 \pm 3.5 \mid 79.9 \pm 3.4$
	M	$62.2 \pm 2.7 \mid 62.1 \pm 0.4$	$61.7 \pm 3.5 \mid 57.8 \pm 0.9$	$61.1 \pm 1.6 \mid 57.1 \pm 4.0$	$63.1 \pm 2.3 \mid 59.8 \pm 3.1$
	H	$39.3 \pm 3.3 \mid 40.2 \pm 0.7$	$42.9 \pm 3.2 \mid 41.6 \pm 2.4$	$39.2 \pm 2.6 \mid 39.7 \pm 2.6$	$39.0 \pm 2.0 \mid 38.5 \pm 1.5$
	I	$14.4 \pm 2.4 \mid 14.2 \pm 2.2$	$15.7 \pm 2.0 \mid 17.0 \pm 0.8$	$12.6 \pm 1.2 \mid 14.5 \pm 1.8$	$14.2 \pm 1.7 \mid 15.0 \pm 1.9$
WANLI	**E**	$94.3 \pm 2.0 \mid 92.9 \pm 1.4$	$92.7 \pm 0.8 \mid 91.2 \pm 0.7$	$95.2 \pm 0.9 \mid 90.9 \pm 1.1$	$94.1 \pm 0.4 \mid 93.1 \pm 1.3$
	M	$72.7 \pm 3.2 \mid 69.6 \pm 3.5$	$71.4 \pm 1.2 \mid 69.9 \pm 2.2$	$72.7 \pm 1.7 \mid 67.9 \pm 3.0$	$72.3 \pm 3.1 \mid 73.8 \pm 2.1$
	H	$41.9 \pm 2.7 \mid 40.4 \pm 1.6$	$43.1 \pm 2.6 \mid 44.2 \pm 1.2$	$42.4 \pm 4.6 \mid 40.7 \pm 2.6$	$40.6 \pm 1.9 \mid 41.0 \pm 3.4$
	I	$12.7 \pm 1.5 \mid 14.4 \pm 1.4$	$14.0 \pm 2.0 \mid 16.6 \pm 4.7$	$11.9 \pm 2.6 \mid 14.7 \pm 4.4$	$13.9 \pm 2.8 \mid 14.6 \pm 3.6$
MNLI	**E**	$93.3 \pm 2.0 \mid 93.6 \pm 0.6$	$94.3 \pm 0.9 \mid 92.8 \pm 1.2$	$94.4 \pm 0.8 \mid 92.4 \pm 0.8$	$93.4 \pm 1.3 \mid 93.8 \pm 1.5$
	M	$72.6 \pm 3.8 \mid 72.9 \pm 1.5$	$78.6 \pm 2.0 \mid 74.7 \pm 4.6$	$75.9 \pm 1.6 \mid 72.8 \pm 4.4$	$74.7 \pm 3.4 \mid 78.1 \pm 4.6$
	H	$41.8 \pm 3.8 \mid 44.5 \pm 1.0$	$51.9 \pm 4.9 \mid 49.3 \pm 3.3$	$45.7 \pm 4.1 \mid 46.7 \pm 3.3$	$46.1 \pm 3.9 \mid 48.3 \pm 4.7$
	I	$12.8 \pm 3.8 \mid 15.7 \pm 1.8$	$20.9 \pm 2.9 \mid 19.5 \pm 1.4$	$13.1 \pm 2.0 \mid 16.9 \pm 1.9$	$12.6 \pm 2.2 \mid 18.5 \pm 6.3$
All	**E**	$91.0 \pm 6.7 \mid 90.8 \pm 5.8$	$89.8 \pm 7.7 \mid 89.1 \pm 5.9$	$91.8 \pm 5.9 \mid 88.0 \pm 8.4$	$90.7 \pm 5.5 \mid 90.4 \pm 6.4$
	M	$69.3 \pm 5.3 \mid 68.1 \pm 4.4$	$70.5 \pm 6.4 \mid 67.6 \pm 6.7$	$70.0 \pm 5.8 \mid 66.8 \pm 6.8$	$69.4 \pm 5.2 \mid 69.9 \pm 7.6$
	H	$42.3 \pm 4.2 \mid 42.4 \pm 2.5$	$46.5 \pm 5.1 \mid 45.3 \pm 3.7$	$43.6 \pm 4.8 \mid 44.1 \pm 4.9$	$42.6 \pm 3.9 \mid 43.4 \pm 5.3$
	I	$14.4 \pm 3.3 \mid 15.3 \pm 2.2$	$17.7 \pm 3.8 \mid 18.8 \pm 3.4$	$14.6 \pm 4.1 \mid 17.1 \pm 4.1$	$14.8 \pm 3.2 \mid 16.4 \pm 4.2$

Table 10: **Multi-source** in-domain strategy comparison for difficulty-stratified test outcomes between original training set and training set with hard-to-learn examples excluded. Values denote test accuracies and standard deviations. We consider performance on test examples belonging to the Easy (**E**), Medium (**M**), Hard (**H**) and Impossible (**I**) sets.
Cell scheme to be read as *ablated* | *original*.

timations follow directly from training dynamics. This echoes a similar finding by (Zhang and Plank, 2021), who show that cartography-based model confidence scores strongly correlate with human agreement on SNLI validation data. Similarly, it may be of interest to see whether examples which *many different* models identify as ambiguous correlate with human judgements of ambiguity, e.g. with respect to human label distributions (Nie et al., 2020; Chen et al., 2020), or annotator disagreement scores.

A.7 Hard-to-learn Data

In the tables below, we present various cases of hard-to-learn examples (i.e., collective outliers) for the NLI datasets we examine.

Table 11: Impossible ($0 \leq p \leq 0.25$) training examples from SNLI. **GT** denotes the gold truth; **P.** denotes the model prediction and **Conf.** the confidence associated with the predicted label.

Ex.	**Premise**	**Hypothesis**	**GT**	**P.**	**Conf.**
(1)	A skier in electric green on the edge of a ramp made of metal bars.	The skier was on the edge of the ramp.	N	E	0.976
(2)	A skier in electric green on the edge of a ramp made of metal bars.	The brightly dressed skier slid down the race course.	E	C	0.770
(3)	A man wearing a red sweater is sitting on a car bumper watching another person work.	people make speed fast at speed breaker.	C	N	0.799
(4)	Middle-aged female wearing a white sunhat and white jacket, slips her hand inside a man's pants pocket.	The man and woman are playing together.	C	N	0.869
(5)	A young girl jumps off of a couch and high into the air	the young lady knows how to fly in sky	C	N	0.765
(6)	A young boy jumps into the oncoming wave.	The boy is at a lake.	C	N	0.951
(7)	A woman taking her wallet out of her purse at a vendor stand	A woman buying something from a vendor.	E	N	0.892
(8)	A group of people standing on a rock path.	A group of people are hiking.	E	N	0.972
(9)	Woman sitting in tree with dove.	The lady is touching a dove.	N	E	0.920
(10)	A lady standing on the corner using her phone.	The lady has a smartphone.	N	E	0.753

Table 12: Impossible ($0 \leq p \leq 0.25$) training examples from WANLI. **GT** denotes the gold truth; **P.** denotes the model prediction and **Conf.** the confidence associated with the predicted label.

Ex.	**Premise**	**Hypothesis**	**GT**	**P.**	**Conf.**
(1)	The first principle of art is that art is not a way of life, but a means of life.	Art is a way of life.	E	C	0.944
(2)	"Must be right good stock," Fenner observed.	"Must be pretty good stock," Fenner said.	C	E	0.646
(3)	He believes that the best thing to do is to buy the firm at a reasonable price.	If the firm is cheap, it is best to buy it.	N	E	0.964
(4)	A piece of paper with a stamp on it is worth less than a piece of paper without a stamp.	A piece of paper without a stamp is worth more than a piece of paper with a stamp.	E	C	0.818
(5)	It was the only time he had seen her laugh.	He had never seen her laugh before.	C	N	0.502
(6)	The musician shook his head.	The musician moved his head up and down.	C	E	0.510
(7)	Some students believe that to achieve their goals they must take the lead.	Some students believe that to achieve their goals they must follow the lead.	E	C	0.630
(8)	The very nature of the "American Dream" is that it is not always attainable	The American Dream is attainable.	N	C	0.817
(9)	This might be an issue for a company that is in the process of introducing a new product.	Every company is always in the process of introducing a new product.	E	N	0.894
(10)	Would Higher Interest Rates Stimulate Saving?	Higher interest rates would not stimulate saving.	E	N	0.929

Table 13: Impossible ($0 \leq p \leq 0.25$) training examples from ANLI. **GT** denotes the gold truth; **P.** denotes the model prediction and **Conf.** the confidence associated with the predicted label. Instances are comparatively less characterized by noisy labels or ambiguity; rather, their difficulty arises through requirement of more advanced inference types, e.g. identifying relevant information from long passages, multi-hop reasoning to connect subjects to events and numerical reasoning about dates of birth and events.

Ex.	Premise	Hypothesis	GT	P.	Conf.
(1)	Glaiza Herradura-Agullo (born February 24, 1978) is a Filipino former child actress. She was the first-ever grand winner of the Little Miss Philippines segment of "Eat Bulaga!" in 1984. She starred in RPN-9's television series "Heredero" with Manilyn Reynes and Richard Arellano. She won the 1988 FAMAS Best Child Actress award for her role in "Batas Sa Aking Kamay" starring Fernando Poe, Jr. .	Herradura-Agullo was born in the 80's	C	E	0.941
(2)	The Whitechapel murders were committed in or near the impoverished Whitechapel district in the East End of London between 3 April 1888 and 13 February 1891. At various points some or all of these eleven unsolved murders of women have been ascribed to the notorious unidentified serial killer known as Jack the Ripper.	The women killed in the Whitechapel murders were impoverished.	N	E	0.832
(3)	Departure of a Grand Old Man is a 1912 Russian silent film about the last days of author Leo Tolstoy. The film was directed by Yakov Protazanov and Elizaveta Thiman, and was actress Olga Petrova's first film.	Olga performed in many films before This one	C	N	0.922
(4)	Gay Sex in the 70s is a 2005 American documentary film about gay sexual culture in New York City in the 1970s. The film was directed by Joseph Lovett and encompasses the twelve years of sexual freedom bookended by the Stonewall riots of 1969 and the recognition of AIDS in 1981, and features interviews with Larry Kramer, Tom Bianchi, Barton Lidice Beneš, Rodger McFarlane, and many others.	Gay Sex in the 70s was directed by a gay man.	N	E	0.907
(5)	Héctor Canziani was an Argentine poet, screenwriter and film director who worked in Argentine cinema in the 1940s and 1950s. Although his work was most abundant in screenwriting and poetry after his brief film career, he is best known for his directorship and production of the 1950 tango dancing film Al Compás de tu Mentira based on a play by Oscar Wilde.	He did direct a movie after 1950	N	E	0.944

Towards a Unified Multi-Domain Multilingual Named Entity Recognition Model

Mayank Kulkarni[*, #, 2], **Daniel Preoţiuc-Pietro**[1], **Karthik Radhakrishnan**[1],
Genta Indra Winata[1], **Shijie Wu**[1], **Lingjue Xie**[1], **Shaohua Yang**[*]
[1]Bloomberg [2]Amazon Alexa AI
`{dpreotiucpie, gwinata}@bloomberg.net`

Abstract

Named Entity Recognition is a key Natural Language Processing task whose performance is sensitive to choice of genre and language. A unified NER model across multiple genres and languages is more practical and efficient through leveraging commonalities across genres or languages. In this paper, we propose a novel setup for NER which includes multi-domain and multilingual training and evaluation across 13 domains and 4 languages. We explore a range of approaches to building a unified model using domain and language adaptation techniques. Our experiments highlight multiple nuances to consider while building a unified model, including that naive data pooling fails to obtain good performance, that domain-specific adaptations are more important than language-specific ones and that including domain-specific adaptations in a unified model can reach performance close to training multiple dedicated monolingual models at a fraction of their parameter count.

1 Introduction

Identifying named entities, such as organization and people in text is a key NLP task situated upstream of other NLP tasks such as co-reference resolution (Ratinov and Roth, 2012; Dutta and Weikum, 2015; Miwa and Bansal, 2016; Luo and Glass, 2018) or relation extraction (Nguyen and Grishman, 2015; Zhong and Chen, 2021) and can enhance applications including information retrieval (Carpineto and Romano, 2012; Berger and Lafferty, 2017) and summarization (Cheng and Lapata, 2016; Liu and Lapata, 2019; Maddela et al., 2022; Hofmann-Coyle et al., 2022). However, it has been established that NER is very sensitive to genre differences (Augenstein et al., 2017; Agarwal et al., 2021),[1] with models trained on one genre performing poorly on a different one. As a result, multiple data sets were created to allow domain-specific models to be trained. Yet, domain adaptation especially across multiple genres has shown the promise that a single model could improve performance on multiple domains (Wang et al., 2020; Liu et al., 2021). Further, the advent of pretrained multilingual models (Wu and Dredze, 2019; Conneau et al., 2020) enables transfer across languages in an straightforward way by simply feeding heterogeneous language data in fine-tuning, making cross-lingual training feasible and a new dimension of adaptation available for exploration. Thus, performance on a specific genre and language pair could be improved by leveraging commonalities in training data across both genre and language dimensions, which is enabled by the significant amount of annotated data sets that are publicly available.

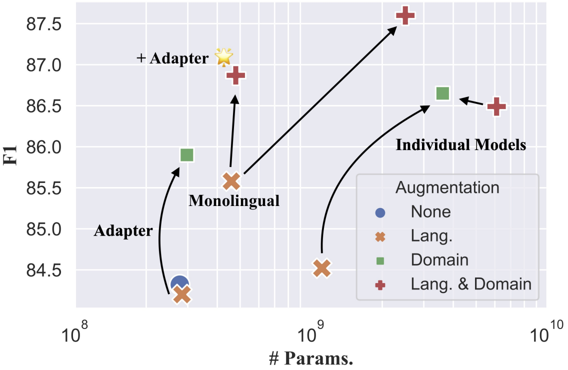


Figure 1: A graphic comparison of performance of language, genre and joint adaptation by demo parameter count.

The main research question becomes *what is the best way to leverage data from different languages and genres for NER* in this multilingual multi-domain setup? In this paper, we attempt to answer this question by using data sets available across multiple genres and languages to improve

---

[*] Work done while at Bloomberg.

[#] The authors are listed in alphabetical order.

[1] Throughout this paper, by genre we refer to a collection of documents with variations in style or structure that might impact modeling (Santini et al., 2006); we use "domain" interchangeably with "genre" when referring to modeling concepts.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2210–2219
May 2-6, 2023 ©2023 Association for Computational Linguistics

performance across all data sets. To this end, we compile a collection of 22 data sets across 4 different languages and spanning 13 domains. To the best of our knowledge, this is the first attempt to building a unified multi-domain multilingual NER model.

We empirically explore our core research question through several experiments. First, we aim to identify whether sharing models or parts of the model across languages, domains or both is more beneficial in training. In general, simply pooling all the available data is likely sub-optimal as domain-specific differences in named entity mentions are useful to model, although using more data is usually more beneficial and can lead to improved robustness of the model. We explore several sharing techniques on top of state-of-the-art transformer-based encoders such as data pooling and mixture of experts methods, previously effective in cross-lingual learning, and language or domain specific adapter heads. Our results (Fig. 1) show that sharing genre information across languages is much more beneficial for performance than sharing language information across genres for all types of adaptation techniques.

Next, we compare monolingual encoders like RoBERTa, which can provide a better representation for text in each language, and multilingual encoders like XLM-R, which enables knowledge sharing from multiple languages, as starting points for fine-tuning NER models when genre and language annotated data is available. We find that the monolingual models pooling all the data from a particular language perform best and outperform their cross-lingual counterparts.

Throughout, we explore the trade-offs between the total number of model parameters and performance, which can bring practical benefits in terms of reduced maintenance and increased efficiency. We find that doing domain adaptation using adapter heads achieves a good trade-off between performance and parameter count and could represent the optimal solution in deploying a unified model.

Our contributions are as follows:

- Introducing the multilingual multi-domain NER setup;
- Extensive experiments on **13 domains** and **4 languages** using a variety of models and adaptation methods which highlight the best unified model architecture and show that modeling domains is more effective than languages;
- Analysis of the performance / efficiency trade-offs.

2 Data Sets

We create a collection of 22 data sets across 4 languages and 13 unique genres. For English, we use CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), Filings (Salinas Alvarado et al., 2015), OntoNotes-En (Hovy et al., 2006) with 6 genres (Pradhan et al., 2013; Wang et al., 2020) and Twitter (Ritter et al., 2011); for Chinese, we use MSRA (Levow, 2006), OntoNotes-Zh (Hovy et al., 2006) with 6 genres and Weibo (Peng and Dredze, 2015, 2016; He and Sun, 2017); for German, we use CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), Legal (Leitner et al., 2019) and Wikipedia (Balasuriya et al., 2009); for Spanish we use CoNLL-2002 (Tjong Kim Sang, 2002) and Wikipedia (Balasuriya et al., 2009). Note that not all languages contain the same genres and not all genres are present in each language, although there is overlap between genres and languages.

2.1 Statistics

We have a total of 502,720 training examples with 109,657 for validation and 105,255 for testing. We consider the following entity types: Person (`PER`), Organization (`ORG`), Location (`LOC`), and Miscellaneous (`MISC`), either removing extra types or collapsing them into the overarching parent entity class. We maintain the train/dev/test splits for all of these data sets and evaluate on test. Tab. 1 shows the number of training, validation and testing data points across each of the languages when domains in the language are combined. Here we can see that despite German and Spanish having few domains, the number of data points are in fact more than English and Chinese which have more domains.

Language	**Train**	**Dev**	**Test**
English	103,121	22,344	22,557
Chinese	67,129	15,672	11,314
German	234,297	50,471	50,611
Spanish	98,173	21,170	20,773
Total	502,720	109,657	105,255

Table 1: Data set statistics per language.

2.2 Entity-Type Mapping

The datasets do not have identical entity types. Thus, we apply pre-processing to standardize the

Language	Domain	ORG	PER	LOC	Dropped
English	Ritter	company	person, musicartist	geo-loc	N/A
	OntoNotes	N/A	N/A	N/A	TIME, CARDINAL, NORP, DATE, ORDINAL, QUANTITY, MONEY, PRODUCT, EVENT, PERCENT, WORK_OF_ART, LAW, LANGUAGE
German	Legal	UN, INN, GRT, MRK	RR, AN	LD, ST, STR, LDS	GS, RS, VO, LIT, VS, VT, EUN
Chinese	Weibo	ORG.NAM, ORG.NOM	PER.NAM, PER.NOM	LOC.NAM, LOC.NOM	N/A
	OntoNotes	N/A	PERSON	GPE	EVENT, NORP, TIME, FAC, QUANTITY, MONEY, CARDINAL, ORDINAL, LOC, LAW, WORK_OF_ART, PERCENT, LANGUAGE, PRODUCT

Table 2: Entity-type Mapping across data sets.

labels. Tab. 2 illustrates the pre-processing to map entities to the ORG, PER, LOC and MISC types. We do not list the simple mapping when the ORG, PER, LOC types exist themselves and blank spaces signify no mapping was done for the type. We reference previous work to map types to our subsets and also refer to the original data set paper to infer type mappings. Additionally, if a type does not map to any of the 4 types we train and evaluate on we drop the type as seen in the last column of the table.

3 Methods

To answer our core research question, we explore several methods inspired by approaches from both multilingual NER (Al-Rfou et al., 2015; Rahimi et al., 2019; Tedeschi et al., 2021) and multi-domain NER (Liu et al., 2020; Wang et al., 2020).

3.1 Multilingual Encoders

Pretrained multilingual encoders learn strong multilingual representation. In particular, we use XLM-R base (Conneau et al., 2020), a strong multilingual encoder.

3.1.1 Individual Models

The models consist of XLM-R base as the encoder, followed by the sequence tagging head in the form of a linear layer.

- **Data Pooling.** We fine-tune 1 model by naively pooling data from all languages and domains;
- **Per Lang.** We fine-tune 4 models using all data from each of the 4 languages;
- **Per Dom.** We fine-tune 1 model per domain using all data across all languages for that domain, resulting in 13 models (e.g., one CoNLL model trained using English, German and Spanish);
- **Per Lang. & Dom.** We fine-tune one model for each language and domain resulting in 22 models (e.g., CoNLL English, CoNLL German).

3.1.2 Mixture of Expert Models

Past multilingual NER research showed promising results using Mixture of Expert (MoE) (Shazeer et al., 2017) based models. MoE models are built on the premise that a set of experts can be parametrically learnt based on the training data without any explicit notion of matching an expert to a specific language or domain. MoE based models could be trained with a regular training setup (Jacobs et al., 1991), with gradient reversal methods (Ganin and Lempitsky, 2015) or with an adversarial loss (Chen and Cardie, 2018; Chen et al., 2019). We train MoE with regular training setup.

Given encoder output H for a sequence of length M, we introduce N experts, $E_i = FFN_i(H_t)$ with one hidden layer, where $i \in \{1, \dots, N\}, t \in \{1, \dots, M\}$, and a linear domain/language N-class classifier $C_{D/L} = Softmax(W_{C_{D/L}} H_{CLS})$. We take $\sum_i \alpha_i E_i$ with α_i from $C_{D/L}$ and feed it to a shared NER classifier. Thus experts are assigned at the sequence-level.[2] These are jointly trained in a multi-task learning setup with a cross-entropy NER loss $\mathcal{L}_{\text{NER}}^{\sum_i \alpha_i E_i}$ associated with all experts E_i and a Domain **(Dom. MoE)** or Language **(Lang. MoE)** prediction cross-entropy loss $\mathcal{L}_{D/L}$, yielding $\mathcal{L}_{\text{NER}}^{\sum_i \alpha_i E_i} + \mathcal{L}_{D/L}$. The loss is backpropagated through all the experts, both NER and domain/language classifier and the shared encoder.

3.1.3 Adapter Models

Research in multi-domain NER has found that adding private layers that are updated by data from

---

[2] We also experimented with token-level expert assignment, but observe worse results on the dev set.

each domain and shared layers updated by data from all domains is an effective way to improve multi-domain performance (Wang et al., 2020). Similar to the private layers explored in multi-domain NER, Adapters (Pfeiffer et al., 2020; Lin et al., 2020; Winata et al., 2021) used in conjunction with Transformer-based models demonstrated promise in further boosting the performance. We thus introduce adapter heads on top of the encoder. We leave variants of adapters that lie within each layer of the encoder (Houlsby et al., 2019; He et al., 2022) as future work.

The adapters A_i use the same model architecture (as MoE models), but are only updated by data from a given domain or language. It is equivalent to MoE with a predefined expert assignment. Fig. 2 shows the architecture of the adapter models used in our experiments.

Thus, for a given data point D_i the loss is computed as $\mathcal{L}_{\text{NER}}^{A_i}$ and only backpropagated through A_i, the NER classifier and the shared encoder.

- **Lang. Adp.** We create 4 adapter heads, one for each language and use the gold language label to pick the adapter;
- **Dom. Adp.** We create 13 adapter heads, one for each domain and use the gold domain label to pick the adapter;
- **Dom. Adp + DP.** We create 13 adapter heads and employ an auxiliary Domain Prediction objective $\mathcal{L}_{D/L}$ during training;
- **Dom. Adp + DP + SA.** In addition, we add a shared head which is updated for all examples, similar to the shared/private setup in (Wang et al., 2020) for multi-domain adaptation.

While adapters in each layer with frozen encoder performs on par with fine-tuning all parameters (Houlsby et al., 2019), it does not outperform it eithers. Thus, we also update the transformer layers as part of the training process. We also explored combining language and domain adapters but this resulted in worse performance and we omit it for brevity.

3.2 Monolingual Encoders

Finally, we explore monolingual encoders, which can provide a better representation of each language but are not able to transfer knowledge across languages. We identify monolingual BERT/RoBERTa versions for each of the 4 languages: English (Liu et al., 2019), Chinese (Cui et al., 2020), Spanish (de la Rosa et al., 2022), and German[3].

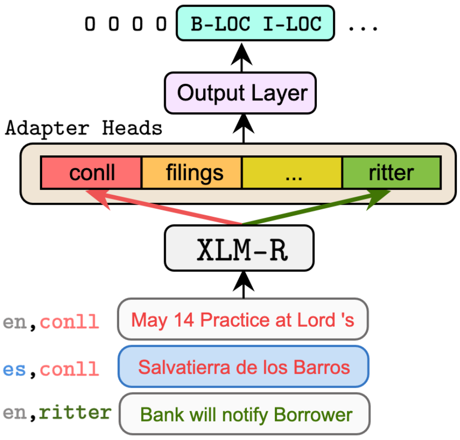


Figure 2: Our NER tagger with XLM-R encoder and domain adapters. Texts and adapter heads are color-coded to indicate the heads used for each domain.

- **Per Lang.** We fine-tune each monolingual encoder with all the NER data from the corresponding language, resulting in 4 models;
- **Per Lang. & Dom.** We fine-tune one model for each domain based on monolingual encoder, resulting in 22 models;
- **Dom. Adp.** We add domain adapters (**Dom. Adp.**, §3.1.3) to monolingual encoder. This results in 4 models, one for each language, with the number of adapters equal to the number of domains in each language.

3.3 Hyperparameters

We use the open-source Transformers library (Wolf et al., 2020) to facilitate reproducibility. For all experiments, we use a learning rate of 1e-5 on the AdamW optimizer (Loshchilov and Hutter, 2017), with no warm up, a batch size of 32 trained across 50 epochs on an NVIDIA V100 GPU. We use the same hyperparameters across all experiments to allow for comparability.

4 Results

We evaluate the models listed in §3 on all data sets. Tab. 3 illustrates the results averaged across languages obtained using F1 calculated with the CoNLL evaluation script. Granular results for each individual genre and language are in App. B. We treat the data pooling method in multi-lingual encoders as our baseline in terms of performance and number of parameters. Our findings are as follows:

---

[3] The model is taken from https://www.deepset.ai/german-bert

Models	**# param**	**en**	**zh**	**de**	**es**	**Avg.**
# Domains		9	8	3	2	
`Initiate with XLM-R (multilingual, base)`						
Data Pooling	×1.00	82.12	75.70	89.51	89.96	84.32
Per Lang.	×4.00	+0.54	+0.67	-0.27	-0.15	+0.2
Per Dom.	×13.00	+3.99	+4.45	+0.47	+0.40	+2.33
Per Lang. & Dom.	×22.00	+3.53	+4.27	+0.49	+0.38	+2.17
Lang. MoE	×1.02	+0.12	+0.47	0	+0.07	+0.17
Dom. MoE	×1.07	+0.51	+1.03	-0.36	-0.33	+0.21
Lang. Adp	×1.02	-0.09	+0.33	-0.37	-0.35	-0.12
Dom. Adp	×1.07	+1.95	+4.62	+0.26	-0.53	+1.58
+ DP	×1.07	+1.65	+3.49	-0.15	-0.83	+1.04
+ DP + SA	×1.08	+1.60	+3.66	-0.30	-0.01	+1.24
`Initiate with Monolingual RoBERTa (base)`						
Per Lang.	×1.65	+1.94	+3.16	-0.24	+0.16	+1.26
Per Lang. & Dom.	×9.03	**+4.23**	+6.55	**+1.28**	**+1.05**	**+3.28**
Dom. Adp	×1.73	+3.05	**+6.61**	+0.22	+0.30	+2.55

Table 3: Results in macro-F1 for each language averaged across all domains within the language and overall average across the four languages. Number of parameters are relative to Data Pooling. **Bold** and underline indicate the best and second best performing models.

Domain vs. Language: In Fig. 1, we observe that across all types of methods (individual models, MoE and adapters), training models that leverage information about domain across languages is more beneficial when compared to sharing information across different genres in the same language, with gains of up to 1.70 – 2.13 F1. We hypothesize this result is due to the well documented sensitivity of NER to nuances specific to genres (Augenstein et al., 2017) such as entity distribution, document structure or capitalization patterns, whereas multilingual models manage to better preserve this information across languages. In addition, domain-specific models even perform slightly better than language- and domain-specific models (+0.16).

Adapters vs. MoE: When comparing methods, we observe that MoE techniques provide limited gains over data pooling (0.17–0.21) contrary to past cross-lingual experiments. The adapter heads provide bigger improvement compared to MoE with same number of parameters, while using shared layers and domain predictors as in multi-domain adaptation (Wang et al., 2020) fails to further boost performance. However, both adaptation strategies lag behind training domain specific models (+0.75), which however come with a much larger number of parameters (up to ×20) and added maintenance cost when deployed.

Monolingual vs. Multilingual Models: The monolingual results demonstrate that, if available, these models lead to better performance than their multilingual counterparts (+1.06 and +1.09 when comparing similar setups), which is natural as they have a better representation of each language. We find that the domain adapter method offers a good trade-off between performance (-0.73) and model size (×0.18), outperforming all models that perform adaptation across languages.

Impact of domain diversity: Finally we also observe that English and Chinese have much more diversity because of the number of domains, thus adding more capacity through the domain adaptation results in improved performance. However, since German and Spanish have fewer domains but an equal if not more training data points, we find that adding capacity is not necessarily helpful.

5 Conclusion and Future Work

This paper introduces the first extensive evaluation of multilingual multi-domain NER using a collection of 22 data sets spanning 4 languages. Through a series of experiments, we demonstrate that genre information is more important to be shared, even across languages, than sharing information from other genres in the same language. However, these cross-lingual methods are outperformed by domain adaptation over genres in monolingual models, if these models are available. We also explored trade-offs between model size and performance, showing that adapter heads strike a good balance, offering relatively little reduction in performance for an order of magnitude less parameters. For future work, we will explore additional experimental setups that include testing on domains or languages where limited or no data is available for training.

Limitations

Our research focuses on high-resource languages where annotated NER data sets and pretrained language models are available from only two language families. We have yet to explore how these findings translate to low resource languages or languages where annotated data sets are not available. We note that there are more domains available for English and Chinese, and since we are computing macro-F1 scores, the results over-emphasize performance on these languages, although Spanish and German show similar result patterns. Additionally, we only use 4 entity types (i.e., PER, ORG, LOC, MISC) across all datasets by dropping the other entities. Finally, due to limited computing resources and large number of experiments, we experiment with XLM-R base and thus do not compare with state-of-the-art results for each of these individual language/domain results which are usually obtained using XLM-R large.

Ethics Statement

We use publicly available data sets in our experiments with permissive licenses for research experiments. We do not release new data or annotations as part of this work.

Acknowledgements

We are grateful to Xinyu Hua for feedback on a draft of this manuscript. We sincerely thank the three anonymous reviewers for their insightful comments on our paper.

References

Oshin Agarwal, Yinfei Yang, Byron C. Wallace, and Ani Nenkova. 2021. Interpretability analysis for named entity recognition to understand system predictions and how they can improve. *Computational Linguistics*, 47(1):117–140.

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. 2015. Polyglot-ner: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, ICDM, pages 586–594.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran. 2009. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18, Suntec, Singapore. Association for Computational Linguistics.

Adam Berger and John Lafferty. 2017. Information retrieval as statistical translation. 51(2):219–226.

Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)*, 44(1):1–50.

Xilun Chen, Ahmed Hassan Awadallah, Hany Hassan, Wei Wang, and Claire Cardie. 2019. Multi-source cross-lingual model transfer: Learning what to share. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3098–3112, Florence, Italy. Association for Computational Linguistics.

Xilun Chen and Claire Cardie. 2018. Multinomial adversarial networks for multi-domain text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1226–1240, New Orleans, Louisiana. Association for Computational Linguistics.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.

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

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 657–668, Online. Association for Computational Linguistics.

Javier de la Rosa, Eduardo G Ponferrada, Paulo Villegas, Pablo González de Prado Salas, Manu Romero, and Marıa Grandury. 2022. Bertin: Efficient pre-training of a spanish language model using perplexity sampling. *Procesamiento del Lenguaje Natural*, 68:13–23.

Sourav Dutta and Gerhard Weikum. 2015. Cross-document co-reference resolution using sample-based clustering with knowledge enrichment. *Transactions of the Association for Computational Linguistics*, 3:15–28.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, ICML, pages 1180–1189.

Hangfeng He and Xu Sun. 2017. F-score driven max margin neural network for named entity recognition in Chinese social media. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 713–718, Valencia, Spain. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, ICLR.

Ella Hofmann-Coyle, Mayank Kulkarni, Lingjue Xie, Mounica Maddela, and Daniel Preoţiuc-Pietro. 2022. Extractive entity-centric summarization as sentence selection using bi-encoders. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, AACL-IJCNLP, pages 326–333.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, ICML, pages 2790–2799.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA. Association for Computational Linguistics.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2019. Fine-grained named entity recognition in legal documents. In *International Conference on Semantic Systems*, pages 272–287.

Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117, Sydney, Australia. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 441–459, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zihan Liu, Genta Indra Winata, and Pascale Fung. 2020. Zero-resource cross-domain named entity recognition. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 1–6, Online. Association for Computational Linguistics.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2021. Crossner: Evaluating cross-domain named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35 of *AAAI*, pages 13452–13460.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*, ICLR.

Hongyin Luo and Jim Glass. 2018. Learning word representations with cross-sentence dependency for end-to-end co-reference resolution. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4829–4833, Brussels, Belgium. Association for Computational Linguistics.

Mounica Maddela, Mayank Kulkarni, and Daniel Preotiuc-Pietro. 2022. EntSUM: A data set for entity-centric extractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3355–3366, Dublin, Ireland. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48, Denver, Colorado. Association for Computational Linguistics.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for Chinese social media with jointly trained embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 548–554, Lisbon, Portugal. Association for Computational Linguistics.

Nanyun Peng and Mark Dredze. 2016. Improving named entity recognition for Chinese social media with word segmentation representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 149–155, Berlin, Germany. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2012. Learning-based multi-sieve co-reference resolution with knowledge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1234–1244, Jeju Island, Korea. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Julio Cesar Salinas Alvarado, Karin Verspoor, and Timothy Baldwin. 2015. Domain adaption of named entity recognition to support credit risk assessment. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 84–90, Parramatta, Australia.

Marina Santini, Richard Power, and Roger Evans. 2006. Implementing a characterization of genre for automatic genre identification of web pages. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 699–706, Sydney, Australia. Association for Computational Linguistics.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, ICLR.

Simone Tedeschi, Valentino Maiorca, Niccolò Campolungo, Francesco Cecconi, and Roberto Navigli. 2021. WikiNEuRal: Combined neural and knowledge-based silver data creation for multilingual NER. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2521–2533, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Jing Wang, Mayank Kulkarni, and Daniel Preotiuc-Pietro. 2020. Multi-domain named entity recognition with genre-aware and agnostic inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8476–8488, Online. Association for Computational Linguistics.

Genta Indra Winata, Guangsen Wang, Caiming Xiong, and Steven Hoi. 2021. Adapt-and-adjust: Overcoming the long-tail problem of multilingual speech recognition. In *22nd Annual Conference of the International Speech Communication Association*, Interspeech, page 361.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

Appendix

A Data Set Details

In this section, we provide details about the statistics of the data sets, our hypothesis on what makes them challenging tasks and also pre-processing we perform to allow for reproducible results.

A.1 Statistics

A more in-depth look at the distributions of the domains across languages can be seen in Tab. 4 for German, Tab. 5 for Spanish, Tab. 8 for English, and Tab. 9 for Chinese. The tables show that English has the most diverse set of domain distribution, followed by Chinese, with a bulk of the data coming from MSRA, German, where Legal and Wiki constitute a large amount and Spanish, which is largely dominated by Wiki. The more diverse set of domains makes the language more challenging to achieve a consistently high average score, which is also evident in our results.

	conll2003	**legal**	**wiki**
Train (%)	5.19	19.93	74.88
Dev (%)	5.68	19.83	74.49
Test (%)	5.94	19.78	74.28

Table 4: Domain Distribution for German data sets.

	conll	**wiki**
Train (%)	8.48	91.52
Dev (%)	9.05	90.95
Test (%)	7.31	92.69

Table 5: Domain Distribution for Spanish data sets.

B Fine-grained Results

In Tab. 3, we see the averaged results across domains for each language, however it is not easy to infer the performance on each for any given language. In an effort to provide more transparency, we provide the performance for each domain within a given language in Tab. 6, Tab. 7, Tab. 10, and Tab. 11.

Models/Domain	**conll2003**	**legal**	**wiki**
`Initialized with XLM-R Multilingual`			
Data Pooling	81.24	96.05	91.23
Per Lang.	80.61	95.80	91.30
Per Dom.	82.08	96.41	91.45
Per Lang. and Dom.	82.26	96.41	91.33
Lang. MoE	81.28	96.10	91.14
Dom. MoE	80.42	95.94	91.09
Lang. Adp	80.33	95.92	91.17
Domain Adp	81.87	96.18	91.25
+ DP	81.41	95.81	90.87
+ DP + SA	81.18	95.68	90.77
`Initialized with Monolingual RoBERTa`			
Per Lang.	81.00	95.55	91.25
Per Lang. & Dom.	84.69	96.05	91.61
Dom. Adp	82.08	95.77	91.36

Table 6: Fine-grained results for domains within German.

Models/Domain	**conll2002**	**wiki**
`Initialized with XLM-R Multilingual`		
Data Pooling	86.72	93.20
Per Lang.	86.51	93.10
Per Dom.	87.46	93.26
Per Lang. and Dom.	87.59	93.09
Lang. MoE	87.02	93.04
Dom. MoE	86.42	92.84
Lang. Adp	86.27	92.96
Dom. Adp	85.93	92.93
+ DP	85.72	92.55
+ DP + SA	87.08	92.81
`Initialized with Monolingual RoBERTa`		
Per Lang.	86.97	93.26
Per Lang. & Dom.	88.78	93.25
Dom. Adp	87.36	93.15

Table 7: Fine-grained results for domains within Spanish. DP is Domain Prediction and SA indicates shared adapter.

	conll2003	filings	onto_bc	onto_bn	onto_mz	onto_nw	onto_tc	onto_wb	ritter
Train (%)	13.62	1.00	11.00	9.05	5.66	29.32	10.79	14.65	4.92
Dev (%)	14.55	0.99	10.88	8.96	5.59	29.00	10.67	14.49	4.86
Test (%)	15.31	0.98	10.79	8.88	5.55	28.73	10.58	14.36	4.83

Table 8: Domain Distribution for English data sets.

	msra	onto_bc	onto_bn	onto_mz	onto_nw	onto_tc	onto_wb	weibo
Train (%)	53.63	10.51	8.92	4.54	3.86	9.01	7.56	1.97
Dev (%)	57.43	9.65	8.19	4.17	3.54	8.27	6.94	1.81
Test (%)	40.94	13.37	11.36	5.78	4.92	11.47	9.63	2.52

Table 9: Domain Distribution for Chinese data sets.

Models/Domain	conll2003	filings	bc	bn	mz	nw	tc	wb	ritter
`Initialized with Multilingual XLM-R`									
Data Pooling	87.21	88.59	88.11	90.92	89.10	91.60	70.69	70.87	61.94
Per Lang.	87.84	86.10	89.14	91.51	89.13	92.13	71.78	71.83	64.45
Per Dom.	92.00	95.48	89.85	92.48	90.81	93.14	73.29	76.44	71.46
Per Lang. & Dom.	91.25	95.48	89.35	92.53	91.19	92.79	73.49	75.85	68.96
Lang. MoE	87.20	88.37	88.44	90.91	88.51	91.74	71.30	70.59	63.13
Dom. MoE	87.59	87.50	87.99	91.35	88.23	91.80	72.18	72.42	64.62
Lang. Adp	87.02	88.33	87.72	91.29	88.26	91.77	70.37	70.54	63.01
Dom. Adp	90.36	90.98	89.19	92.38	89.66	92.22	71.59	75.23	65.02
+ DP	90.27	89.45	88.67	92.34	89.55	91.93	71.21	76.44	64.07
+ DP + SA	90.17	89.46	88.56	92.24	89.18	91.96	72.80	75.50	63.60
`Initialized with Monolingual RoBERTa`									
Per Lang.	88.81	90.66	89.80	91.92	89.61	92.48	72.29	73.76	67.22
Per Lang. & Dom.	91.69	94.44	89.81	92.85	91.15	93.42	73.77	75.78	74.25
Dom. Adp	91.85	89.71	89.64	93.07	90.79	92.97	72.29	75.94	70.30

Table 10: Fine-grained results for domains within English. DP is Domain Prediction and SA indicates shared adapter.

Models/Domain	msra	bc	bn	mz	nw	tc	wb	weibo
`Initialized with XLM-R Multilingual`								
Data Pooling	81.09	78.30	79.08	72.75	84.19	81.18	68.63	60.35
Per Lang.	80.85	76.58	78.94	74.30	84.68	83.33	68.36	63.90
Per Dom.	91.81	77.62	82.42	76.25	90.23	84.59	73.51	64.74
Per Lang. & Dom.	91.82	77.82	82.21	76.07	89.74	83.81	73.59	64.74
Lang. MoE	81.04	77.05	79.40	72.47	84.91	84.59	68.35	61.56
Dom. MoE	79.59	76.76	79.32	74.21	85.66	85.29	70.21	62.81
Lang. Adp	80.57	79.07	78.51	73.46	84.62	82.99	68.10	60.89
Dom. Adp	89.45	79.47	82.14	76.21	89.99	84.86	74.34	66.10
+ DP	89.36	77.08	80.67	75.17	89.76	85.22	74.00	62.24
+ DP + SA	89.00	77.73	81.14	75.82	89.54	85.55	72.16	63.97
`Initialized with Monolingual RoBERTa`								
Per Lang.	82.25	79.78	81.02	75.43	86.39	86.10	71.85	68.04
Per Lang. & Dom.	93.55	80.36	84.05	78.52	91.35	85.84	74.73	69.6
Dom. Adp	93.17	80.34	83.88	78.79	91.22	86.74	76.34	67.96

Table 11: Fine-grained results for domains within Chinese. DP is Domain Prediction and SA indicates shared adapter.

Do Neural Topic Models Really Need Dropout?
Analysis of the Effect of Dropout in Topic Modeling

Suman Adhya, Avishek Lahiri, Debarshi Kumar Sanyal
Indian Association for the Cultivation of Science, Jadavpur, Kolkata-700032, India
`{ adhyasuman30, avisheklahiri2014, debarshisanyal }@gmail.com`

Abstract

Dropout is a widely used regularization trick to resolve the overfitting issue in large feedforward neural networks trained on a small dataset, which performs poorly on the held-out test subset. Although the effectiveness of this regularization trick has been extensively studied for convolutional neural networks, there is a lack of analysis of it for unsupervised models and in particular, VAE-based neural topic models. In this paper, we have analyzed the consequences of dropout in the encoder as well as in the decoder of the VAE architecture in three widely used neural topic models, namely, contextualized topic model (CTM), ProdLDA, and embedded topic model (ETM) using four publicly available datasets. We characterize the dropout effect on these models in terms of the quality and predictive performance of the generated topics.

1 Introduction

Dropout (Hinton et al., 2012) is used while training neural networks, by stochastically dropping out the activation of neurons to prevent complex co-adaptations of feature vectors (Baldi and Sadowski, 2013). The working of dropout is attributed to the implicit averaging over an ensemble of neural networks (Labach et al., 2019; Warde-Farley et al., 2014). It has been shown to be effective on supervised learning tasks to prevent overfitting (Srivastava et al., 2014).

As the volume of digital documents significantly increases with time, organizing them manually is becoming quite an inconvenient task. Because of the ability of topic models to learn a thematic structure from a set of documents in an unsupervised manner and label the documents with their corresponding dominant topics, the significance of topic models is enormous in this area (Hall et al., 2008; Adhya and Sanyal, 2022). But in the traditional topic models, not only the computation cost of the approximate posterior is very high but also for a small change in the modeling assumption, re-derivation of the inference method is needed. With greater flexibility and scalability than traditional topic models, a class of Neural Topic Models (NTMs) aim to leverage the potential of neural networks using the AEVB (Kingma and Welling, 2014) based inference technique. Following (Zhao et al., 2021), we refer to this class of models as VAE-NTMs where the training objective is to maximize the log-likelihood of the reconstruction of the input document while minimizing the KL-divergence of the learned posterior distribution of the latent space from a known prior distribution.

An earlier study by (Ha et al., 2019) of the dropout effect on two traditional topic models LDA (Blei et al., 2003) and BTM (Yan et al., 2013) shows that the correct choice of the dropout rate not only decreases the learning time of the models but also significantly improves the predictive performance and generalization for short texts. However, the study does not consider neural topic models.

In this work, we propose the use of dropout on VAE-NTMs as a hyperparameter in order to achieve much better performance in terms of topic coherence, topic diversity, and topic quality. We test this proposition on a range of standard VAE-NTM architectures. To the best of our knowledge, there has been no other study focusing specifically on the use of dropout in neural topic models. We have made our analysis publicly available[1].

In summary, our contributions are as follows:

1. We comprehensively show both quantitatively and qualitatively that topic quality undergoes a massive improvement with either very low or zero dropout settings in both the encoder and the decoder of a VAE-NTM.

2. We show that for VAE-NTMs the systematic choice of low dropout rates can lead to a sig-

---

[1] `https://github.com/AdhyaSuman/NTMs_Dropout_Analysis`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2220–2229
May 2-6, 2023 ©2023 Association for Computational Linguistics

nificant improvement in downstream tasks like document classification.

3. We study the dependence of dropout on the length of the input documents.
4. We present an empirical analysis for the increase in performance of VAE-NTMs with a decrease in dropout.

2 Task Formulation

Given a corpus $\{D_1, D_2, \ldots, D_N\}$ of N documents with vocabulary $\{w_1, w_2, ..., w_V\}$ of V words, topic models describe a document D_i as a distribution over K topics $\{\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, ..., \boldsymbol{\beta}_K\}$, where an individual topic $\boldsymbol{\beta}_k$ is a distribution over V-words.

2.1 VAE Framework in Neural Topic Models

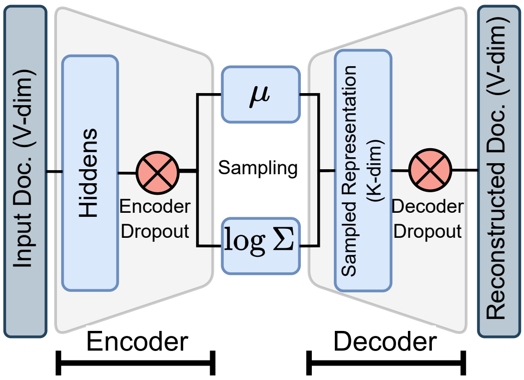


Figure 1: VAE framework in neural topic models.

Given an input sample $\mathbf{x}$, a VAE encoder learns the approximate posterior distribution $q_W(\mathbf{z}|\mathbf{x})$ where W is the encoder's weights that are to be learned and $\mathbf{z}$ is a latent variable. Given a sample $\mathbf{z} \sim q_W(\mathbf{z}|\mathbf{x})$, the VAE decoder learns the likelihood $p_{W'}(\mathbf{x}|\mathbf{z})$ where W' is the learnable decoder's weights.

In VAE-NTMs the input to the encoder is a document representation (e.g., bag-of-words) $\mathbf{x}_{V\times1}$. The encoder then returns the Gaussian parameters $(\boldsymbol{\mu}_{K\times1}, \boldsymbol{\Sigma}_{K\times1})$ that approximate the true posterior where K is the dimension of latent (topic) space, $\boldsymbol{\mu}_{K\times1}$ is the mean, and $\boldsymbol{\Sigma}_{K\times1}$ is the diagonal covariance matrix. Upon taking these Gaussian parameters as input, the decoder samples a latent representation $\mathbf{z}_{K\times1}$ from $\mathcal{N}(\boldsymbol{\mu}_{K\times1}, \boldsymbol{\Sigma}_{K\times1})$ using the reparametrization trick as follows:

$$\mathbf{z}_{K\times1} = \boldsymbol{\mu}_{K\times1} + \boldsymbol{\Sigma}^{\frac{1}{2}}_{K\times1} \odot \boldsymbol{\epsilon}_{K\times1}$$

where $\boldsymbol{\epsilon}_{K\times1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\odot$ represents the element-wise product. Then the document-topic distribution vector $(\boldsymbol{\theta}_{K\times1})$ is generated such that $\boldsymbol{\theta}_{K\times1} = \sigma(\mathbf{z}_{K\times1})$ where $\sigma(\cdot)$ is a softmax function. The input document-term distribution vector is reconstructed with the product of $\boldsymbol{\theta}_{K\times1}$ and $\boldsymbol{\beta}_{K\times V}$, the topic-word matrix, in the following manner:

$$\tilde{\mathbf{x}}_{V\times1} = \begin{cases} \boldsymbol{\beta}^T\boldsymbol{\theta} & \text{if } \boldsymbol{\beta} \text{ is normalized.} \\ \sigma\left(\boldsymbol{\beta}^T\boldsymbol{\theta}\right) & \text{if } \boldsymbol{\beta} \text{ is unnormalized.} \end{cases}$$

As shown in Figure 1, in the encoder, dropout is applied with probability E_p on the output of the hidden layer(s) of the multi-layer feed-forward neural network (FFNN). This output is then fed to two separate layers to get the approximate posterior $q_W(z|x)$. In the decoder, dropout is applied with probability D_p on the document-topic distribution vector $(\boldsymbol{\theta}_{K\times1})$, just before the reconstruction process.

2.2 Task Description

The goal is to measure the effect that dropout has on the *performance* of VAE-NTMs by varying the dropout rates from 0.0 to 0.6 in steps of 0.1, in both the encoder and the decoder. We have chosen 0.6 as the upper bound of the dropout rates for our experiments because it is the highest dropout rate used in any VAE-NTMs that we have considered as a baseline in this work. We measure performance using: *topic coherence*, *topic diversity*, and *topic quality*. We use NPMI (Lau et al., 2014; Röder et al., 2015) to measure topic coherence. Topic diversity (Dieng et al., 2020) shows the uniqueness of topics. Topic quality is the product of coherence and diversity (Dieng et al., 2020). As the automated topic model measures do not always accurately capture the quality of the topics (Hoyle et al., 2021), we also perform a manual evaluation of the topics and study their predictive performance on the document classification task.

3 Empirical Study

We perform all experiments in OCTIS (Terragni et al., 2021), which is an integrated framework for topic modeling.

3.1 Datasets

We have used four publicly available datasets in our experiments. Among them, **20NG**[2] and **BBC**

---

[2] `http://qwone.com/~jason/20Newsgroups/`

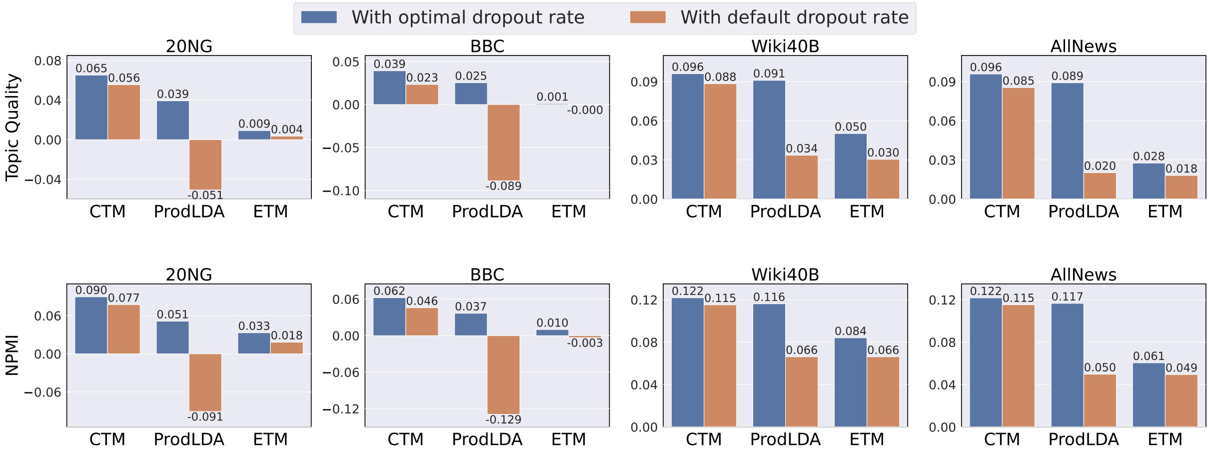


Figure 2: Topic quality and NPMI for different topic models with optimal dropout rate and default dropout rate.

(Greene and Cunningham, 2006) are already available in OCTIS in the pre-processed format while we added **Wiki40B** (Guo et al., 2020) and **AllNews** (Zhu et al., 2018) datasets further. The statistical descriptions of these datasets are mentioned in Table 1. Each corpus is split into train/valid/test sets in the ratio $70 : 15 : 15$. The validation set is used for early stopping.

Dataset	#Docs	Avg. #words	|Vocab|
20NG	16309	48.02	1612
BBC	2225	120.12	2949
Wiki40B	24774	541.08	2000
AllNews	49754	229.53	2000

Table 1: Statistics of the used datasets.

3.2 Models

We use the following three VAE-NTMs: **CTM** (Bianchi et al., 2021) which incorporates the contextualized documents embeddings with the neural topic models; **ProdLDA** (Srivastava and Sutton, 2017) which, unlike LDA, relaxes the simplex constraint over the topic-word matrix; (**ETM**) (Dieng et al., 2020) which incorporates word-embeddings in topic modeling to increase robustness in presence of stopwords.

For each of the four datasets, we compute the dropout rate that optimizes the topic quality of each model on that dataset. We train all three topic models for topic-count $K \in \{20, 50, 100\}$ with 30 epochs while keeping all hyperparameter values, except dropout, the same as in their original implementations. To ensure robustness, we average scores over 10 independent runs of each model. For comparison, we use the default dropout rates for each model as mentioned in the original papers that proposed the corresponding model. In Table 2, we show the default and the optimal dropout rates.

Model	20NG	BBC	Wiki40B	AllNews
CTM (0.2, 0.2)	(0.0, 0.0)	(0.0, 0.0)	(0.2, 0.1)	(0.0, 0.1)
ProdLDA (0.6, 0.6)	(0.1, 0.1)	(0.0, 0.0)	(0.1, 0.1)	(0.1, 0.1)
ETM (0.5, 0.0)	(0.0, 0.0)	(0.1, 0.0)	(0.0, 0.0)	(0.1, 0.0)

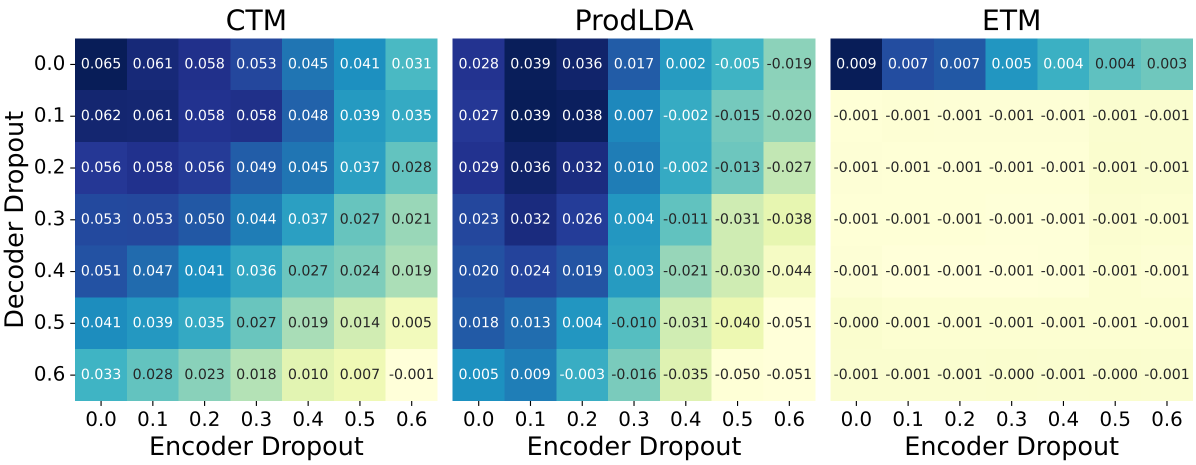
Table 2: For each of the datasets, the optimal dropout rates of all the models considering the highest topic quality are mentioned in the (E_p, D_p) format in the second through last columns. The default dropout rate is also specified for each model in the first column.

3.3 Results and Analysis

3.3.1 Quantitative Evaluation of Topic Quality

In Figure 2, we compare, for each dataset and each model, the topic quality and the NPMI respectively between the dropout-optimized model that gives the highest topic quality and the model with default dropout rates as mentioned in Table 2.

On **20NG**, the topic quality score for (CTM, ProdLDA, ETM) is improved from $(0.056, -0.051, 0.004)$ to $(0.065, 0.039, 0.009)$ by optimizing the dropout rate. For CTM, the increase in performance is around 16.07% whereas for the other two models it is over 100%. This is because the original implementation of CTM already uses a relatively low dropout rate, i.e.,



CTM

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.065	0.061	0.058	0.053	0.045	0.041	0.031
0.1	0.062	0.061	0.058	0.058	0.048	0.039	0.035
0.2	0.056	0.058	0.056	0.049	0.045	0.037	0.028
0.3	0.053	0.053	0.050	0.044	0.037	0.027	0.021
0.4	0.051	0.047	0.041	0.036	0.027	0.024	0.019
0.5	0.041	0.039	0.035	0.027	0.019	0.014	0.005
0.6	0.033	0.028	0.023	0.018	0.010	0.007	-0.001

ProdLDA

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.028	0.039	0.036	0.017	0.002	-0.005	-0.019
0.1	0.027	0.039	0.038	0.007	-0.002	-0.015	-0.020
0.2	0.029	0.036	0.032	0.010	-0.002	-0.013	-0.027
0.3	0.023	0.032	0.026	0.004	-0.011	-0.031	-0.038
0.4	0.020	0.024	0.019	0.003	-0.021	-0.030	-0.044
0.5	0.018	0.013	0.004	-0.010	-0.031	-0.040	-0.051
0.6	0.005	0.009	-0.003	-0.016	-0.035	-0.050	-0.051

ETM

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.009	0.007	0.007	0.005	0.004	0.004	0.003
0.1	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001
0.2	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001
0.3	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001
0.4	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001
0.5	-0.000	-0.001	-0.001	-0.001	-0.001	-0.001	-0.001
0.6	-0.001	-0.001	-0.001	-0.000	-0.001	-0.000	-0.001

Figure 3: Topic qualities on **20NG** for $(E_p, D_p) \in [0.0,\ 0.6] \times [0.0,\ 0.6]$ with a increment of 0.1.

0.2, for both the encoder and the decoder. The other two models show a significant increase in performance due to their large dropout in the baseline models.

Figure 3 shows that the topic quality on the **20NG** dataset for the VAE-NTMs generally produces better results on keeping the dropout rate for both the encoder and the decoder either to be zero or close to it, especially values like $\{0.0, 0.1\}$. Similar results have been found for the other datasets. Based on these observations, the topic quality is found to reduce with an increase in dropout rates in the encoder and decoder.

3.3.2 Qualitative Evaluation of Topic Quality

To qualitatively evaluate the models, we trained all of them for a topic count of 100 on the **20NG** dataset. We then aligned the topics for each pair of *(optimal-dropout model, default-dropout model)* for all three different models. We followed a two-step strategy for topic alignment. For a given pair of models, namely, one with optimal dropout and another with default dropout, with topics lists P and Q, respectively, we first construct a similarity matrix of the topic lists using Rank-biased Overlap (Webber et al., 2010) (RBO) which computes the similarity between two ordered lists by taking into consideration the rank of the individual elements. For example, for 100 topics, we get a matrix, $\mathbf{A} = (a_{ij})_{1 \le i,j \le 100}$ such that, $a_{i,j} = \text{RBO}\left(P[i], Q[j]\right)$. The RBO score lies in $[0, 1]$, where 0 represents no overlap and 1 implies exact overlap. In the final step, we iteratively select the pair of topics for which the similarity score is maximum and simultaneously exclude these two topics from further consideration, i.e. if $\left(P[i_1], Q[j_1]\right)$ and $\left(P[i_2], Q[j_2]\right)$ are two selected pairs then $(i_1 \neq i_2 \wedge j_1 \neq j_2)$.

In Table 3 we show the top words from *aligned* topics of all the models. '$*$' marked models have dropout optimized to give the highest topic quality while others use the default dropout rates as mentioned in Table 2. We see that dropout-optimized models output more interpretable topics.

3.3.3 Effect of Dataset Length

Among the input datasets on which we have experimented, the **20NG** dataset contains relatively short texts, while the others contain longer texts. (Ha et al., 2019) find that their dropout methods are not effective on long texts. But here we see that the performance of all VAE-NTMs decreases uniformly with the increase in the dropout rate, irrespective of the length of the dataset.

3.3.4 Document Classification

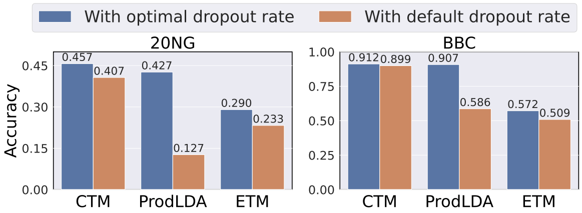


Figure 4: Accuracy for different topic models with optimal dropout and default dropout from Table 2.

We test the predictive performance of the topics produced by the models on a document classification task. We train the models on **20NG** and **BBC** corpora for K topics using the training subset. We represent each document as a K-dimensional document-topic vector and train an SVM, which is then tested on the test subset. We average the

Model	**Topics**
CTM* $(0.0, 0.0)$	**monitor**, **card**, **video**, **port**, **vga**, **apple**, **connector**, **serial**, **slot**, **output** **firearm**, **weapon**, **dangerous**, **military**, **license**, *file*, *state*, **gun**, **police**, *issue* **christian**, **truth**, **scripture**, *exist*, **belief**, **accept**, *understand*, *word*, **human**, **doctrine**
CTM $(0.2, 0.2)$	**card**, **monitor**, **video**, *offer*, *sale*, *upgrade*, **mouse**, **vga**, **port**, **parallel** **firearm**, **dangerous**, **license**, **weapon**, *section*, *file*, *division*, *device*, *manufacture*, *carry* *interpretation*, **truth**, **scripture**, **christian**, **agree**, **moral**, *understand*, **human**, **faith**, *claim*
ProdLDA* $(0.1, 0.1)$	**window**, **driver**, *mode*, **run**, **mouse**, **session**, **server**, **program**, **manager**, **install** **car**, **engine**, **buy**, **company**, **vehicle**, *make*, **brake**, **tire**, **dealer**, **road** **signal**, **voltage**, **output**, **circuit**, **noise**, **power**, **switch**, **wire**, **connector**, *degree*
ProdLDA $(0.6, 0.6)$	*line*, **window**, *gun*, **read**, **space**, **run**, *statement*, **datum**, **drive**, *make* *make*, **battery**, **engine**, *homosexual*, *assault*, *reason*, *place*, *single*, *large*, *attempt* **voltage**, *damn*, **signal**, *usual*, *label*, *hour*, *bio*, *leg*, *bullet*, *hundred*
ETM* $(0.0, 0.0)$	**version**, **software**, **program**, **file**, *include*, **image**, **application**, *set*, **server**, *support* **armenian**, **turkish**, **village**, **people**, **israeli**, **population**, **muslim**, **kill**, **russian**, **genocide** **system**, **run**, **work**, **window**, *problem*, *include*, *set*, *good*, *support*, **information**
ETM $(0.5, 0.0)$	**file**, **application**, *set*, **program**, *support*, **image**, **display**, *list*, **version**, **bit** **armenian**, **turkish**, **village**, **israeli**, **population**, **muslim**, **genocide**, *son*, *land*, **jewish** **work**, *call*, **system**, **window**, *problem*, **bit**, *set*, **run**, *support*, *good*

Table 3: Some selected topics among 100 topics from **20NG**. '*' indicates models with optimal dropout. The dropout rate is mentioned in the (E_p, D_p) format. The more related words in a topic are highlighted in bold while less related ones are italicized.

accuracy scores over $K \in \{20, 50, 100\}$. Figure 4 shows that accuracy increases when we use the optimized dropout rates.

4 Theoretical Understanding of Results

Our experiments show that by tuning the dropout carefully, we can achieve a significant improvement in the performance of VAE-NTMs. Therefore, we argue that the dropout rate should be treated as an important hyperparameter and carefully selected based on the choice of the model as well as the dataset, especially in the case of VAE-NTMs. More precisely, in most cases, low dropout rates in the encoder and the decoder lead to higher performance than that achieved for higher dropout rates.

Standard dropout and other types of dropout have been extensively used in supervised learning techniques (Srivastava et al., 2014; Wu and Gu, 2015; Tompson et al., 2015; Devries and Taylor, 2017; Cai et al., 2019). The main prerogative of using dropout in the supervised scenario is to introduce noise while training so that the model can recognize the outliers in the testing phase. The drop in performance with high dropout that we see in our experiments is perhaps due to the fact that we are trying to learn a generative model of the data. Dropout makes the model robust against perturbations in the input data and thereby also prevents it from learning the characteristics of the input distribution accurately. This is probably why we see a drop in topic coherence and quality. In the case of document classification, if the topic model is trained with a high dropout, the document-topic vectors are of poor quality and the classifier gets trained on these vectors; this results in poor accuracy on the test documents. This setting is different from the usual supervised learning of neural classifiers where dropout is introduced directly in the classifier to prevent overfitting. We intend to analyze these aspects in more depth in the future.

5 Conclusion

We present a detailed study of the effect of the dropout rate on VAE-NTMs. We find that the model performance generally reduces with the increase in dropout rate in the encoder as well as the decoder.

Limitations

The following limitations are known and should be considered when applying the results of this work or relying on them in future studies: (1) Other variants of dropout can be applied to the VAE-NTMs. (2) Analysis of the dropout effect may be done for other VAE-NTMs as well. (3) Other downstream tasks may be formulated for further analysis.

Acknowledgments

This work is partially supported by the SERB-DST Project CRG/2021/000803 sponsored by the Department of Science and Technology, Government of India at Indian Association for the Cultivation of Science, Kolkata.

References

Suman Adhya and Debarshi Kumar Sanyal. 2022. What does the Indian Parliament discuss? an exploratory analysis of the question hour in the Lok Sabha. In *Proceedings of the LREC 2022 workshop on Natural Language Processing for Political Sciences*, pages 72–78, Marseille, France. European Language Resources Association.

Pierre Baldi and Peter J Sadowski. 2013. Understanding dropout. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2021. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 759–766. Association for Computational Linguistics.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

Shaofeng Cai, Jinyang Gao, Meihui Zhang, Wei Wang, Gang Chen, and Beng Chin Ooi. 2019. Effective and efficient dropout for deep convolutional neural networks. *CoRR*, abs/1904.03392.

Terrance Devries and Graham W. Taylor. 2017. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552.

Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Derek Greene and Pádraig Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 377–384, New York, NY, USA. Association for Computing Machinery.

Mandy Guo, Zihang Dai, Denny Vrandečić, and Rami Al-Rfou. 2020. Wiki-40B: Multilingual language model dataset. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2440–2452, Marseille, France. European Language Resources Association.

Cuong Ha, Van-Dang Tran, Linh Ngo Van, and Khoat Than. 2019. Eliminating overfitting of probabilistic topic models on short and noisy text: The role of dropout. *International Journal of Approximate Reasoning*, 112:85–104.

David Hall, Daniel Jurafsky, and Christopher D. Manning. 2008. Studying the history of ideas using topic models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 363–371, Honolulu, Hawaii. Association for Computational Linguistics.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Alexander Hoyle, Pranav Goel, Andrew Hian-Cheong, Denis Peskov, Jordan Lee Boyd-Graber, and Philip Resnik. 2021. Is automated topic model evaluation broken? the incoherence of coherence. In *Advances in Neural Information Processing Systems*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations, ICLR 2014*.

Alex Labach, Hojjat Salehinejad, and Shahrokh Valaee. 2019. Survey of dropout methods for deep neural networks. *CoRR*, abs/1904.13310.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539, Gothenburg, Sweden. Association for Computational Linguistics.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, page 399–408, New York, NY, USA. Association for Computing Machinery.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *Proceedings of the 5th International Conference on Learning Representations, ICLR 2017*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Silvia Terragni, Elisabetta Fersini, Bruno Giovanni Galuzzi, Pietro Tropeano, and Antonio Candelieri. 2021. OCTIS: Comparing and optimizing topic models is simple! In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 263–270. Association for Computational Linguistics.

Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. 2015. Efficient object localization using convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 648–656. IEEE Computer Society.

David Warde-Farley, Ian J. Goodfellow, Aaron C. Courville, and Yoshua Bengio. 2014. An empirical analysis of dropout in piecewise linear networks. In *2nd International Conference on Learning Representations, ICLR 2014*.

William Webber, Alistair Moffat, and Justin Zobel. 2010. A similarity measure for indefinite rankings. *ACM Transactions on Information Systems (TOIS)*, 28(4):1–38.

Haibing Wu and Xiaodong Gu. 2015. Towards dropout training for convolutional neural networks. *Neural Networks*, 71:1–10.

Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, page 1445–1456, New York, NY, USA. Association for Computing Machinery.

He Zhao, Dinh Phung, Viet Huynh, Yuan Jin, Lan Du, and Wray L. Buntine. 2021. Topic modelling meets deep neural networks: A survey. *CoRR*, abs/2103.00498.

Qile Zhu, Zheng Feng, and Xiaolin Li. 2018. GraphBTM: Graph enhanced autoencoded variational inference for biterm topic model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4663–4672, Brussels, Belgium. Association for Computational Linguistics.

A Appendix

A.1 Datasets

We run our experiments on the following datasets:

- **20NewsGroups (20NG)**[3] is a dataset of 18,846 documents from 20 different newsgroups posts. The **20NG** dataset is present in OCTIS, so it is already in pre-processed form. All the documents of this dataset have their corresponding category type as the document labels. The details about these categories are mentioned in Table 4.

- **BBC News (BBC)** (Greene and Cunningham, 2006) is a dataset of news articles from BBC. It is also accessible from OCTIS in pre-processed form. The documents of this dataset are categorized into 5 different categories which are *tech*, *business*, *entertainment*, *sports*, and *politics*. The details of these categories are mentioned in Table 5.

- **Wiki40B**(Guo et al., 2020) is a Wikipedia text dataset in 40+ languages, available in TensorFlow dataset format. In our experiment, we take a sample of 24,774 English documents from this dataset.

- **All the News (AllNews)**(Zhu et al., 2018) dataset consists of 50,001 news articles from 15 news publishers.

#No.	Label	#Docs	%Docs
1.	misc.forsale	861	5.28
2.	comp.windows.x	883	5.41
3.	soc.religion.christian	920	5.64
4.	talk.religion.misc	521	3.19
5.	rec.autos	822	5.04
6.	sci.med	866	5.31
7.	talk.politics.misc	689	4.22
8.	talk.politics.mideast	828	5.08
9.	sci.electronics	867	5.32
10.	rec.sport.hockey	843	5.17
11.	rec.sport.baseball	787	4.83
12.	talk.politics.guns	808	4.95
13.	sci.crypt	883	5.41
14.	comp.sys.mac.hardware	838	5.14
15.	comp.sys.ibm.pc.hardware	891	5.46
16.	comp.graphics	836	5.13
17.	comp.os.ms-windows.misc	828	5.08
18.	alt.atheism	689	4.22
19.	sci.space	856	5.25
20.	rec.motorcycles	793	4.86

Table 4: **20NG** labels with corresponding document counts and percentage of documents.

#No.	Label	#Docs	%Docs
1.	tech	401	18.02
2.	business	510	22.92
3.	entertainment	386	17.35
4.	sport	511	22.97
5.	politics	417	18.74

Table 5: **BBC** labels with corresponding document counts and percentage of documents.

A.2 Pre-processing Steps

Using OCTIS, we convert each document to lowercase, remove the punctuations, lemmatize it, filter

---

[3] `http://qwone.com/~jason/20Newsgroups/`

the vocabulary with the most frequent 2000 terms, filter words with less than 3 characters, and filter documents with less than 3 words.

A.3 Topic Evaluation Metrics

1. **Coherence metric**: This measures how much the top words of the topics are relevant. Topic coherence (TC) for K topics each of which contains n top words can be calculated as:

$$\text{TC} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{{}^{n}C_2}\sum_{i=1}^{n}\sum_{j=i+1}^{n} f\left(w_i^{(k)}, w_j^{(k)}\right)$$

   Here, $f(\cdot,\cdot)$ is the Normalized Pointwise Mutual Information or NPMI (Lau et al., 2014) of the words $w_i^{(k)}$ and $w_j^{(k)}$ appearing in topic k:

$$f(w_i^{(k)}, w_j^{(k)}) = \frac{\log \frac{p(w_i^{(k)},w_j^{(k)})+\epsilon}{p(w_i^{(k)})p(w_j^{(k)})}}{-\log\left(p(w_i^{(k)}, w_j^{(k)}) + \epsilon\right)}$$

   where, $p(w_i^{(k)}, w_j^{(k)})$ is the probability of the co-occurrence of the words $w_i^{(k)}$ and $w_j^{(k)}$ in a boolean sliding window in topic k and $p(w_i^{(k)})$ and $p(w_j^{(k)})$ represents the probability of the occurrence of the individual words in topic k. ϵ is a small positive constant that is used to avoid zero in the $\log(\cdot)$ function.

2. **Diversity metric**: This measures how much the generated topics are different from each other. To measure the diversity score we have used the metric Topic Diversity (TD) (Dieng et al., 2020) which is defined as the proportion of the number of unique words appearing across all topics. It ranges between $[0, 1]$ where a value close to 0 implies repetitive topics and a value near 1 represents more diversification in the topics.

3. **Topic quality**: This is an overall metric that is defined as the product of the two metrics NPMI and TD.

In our experiments, we take the top 10 words for each topic (i.e., $n = 10$) to compute NPMI and TD scores.

A.4 Computing Infrastructure

Our experiments were run on a workstation with Intel® Xeon™ Gold 6326 CPU @ 2.90GHz, 256.0 GB RAM, NVIDIA A100 80GB PCIe, CUDA Version: 11.7 and Ubuntu 22.04 operating system.

A.5 Detailed Results

The detailed results of our experiments are given in Tables 6, 7, and 8. An asterisk ($*$) against a model in the above tables indicates that it is trained with the optimal dropout rate, and the absence of an asterisk indicates that the default dropout rate is used. The default dropout rate for **CTM** is taken from (Bianchi et al., 2021), for **ProdLDA** from (Srivastava and Sutton, 2017), and for **ETM** from (Dieng et al., 2020).

Model	**Topic quality for each dataset**			
	20NG	**BBC**	**Wiki40B**	**AllNews**
CTM*	0.0652	0.0392	0.0961	0.0958
CTM	0.0556	0.0234	0.0884	0.0854
ProdLDA*	0.0392	0.0254	0.0910	0.0891
ProdLDA	-0.0507	-0.0887	0.0336	0.0202
ETM*	0.0092	0.0011	0.0502	0.0276
ETM	0.0036	-0.0003	0.0304	0.0181

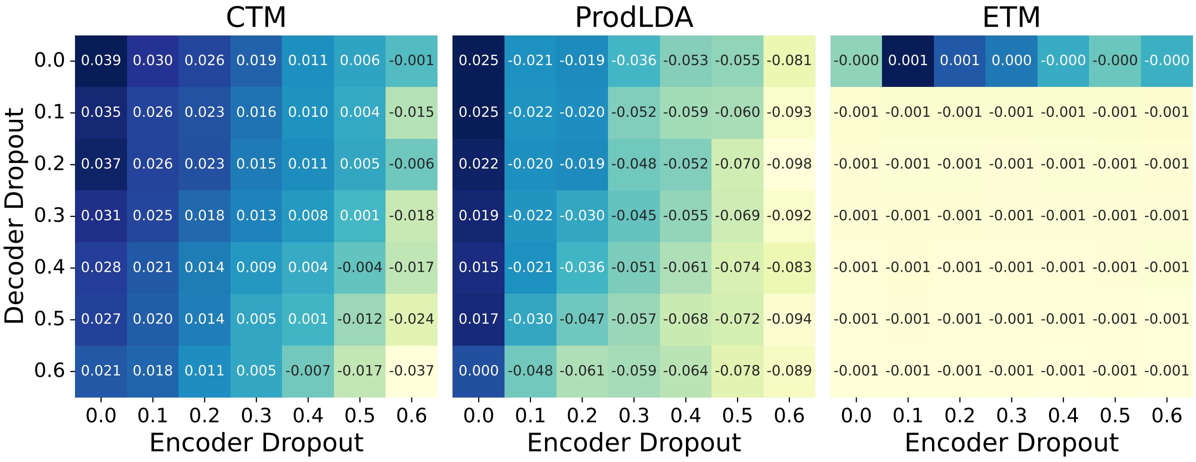
Table 6: Topic Quality values for different VAE-NTMs with optimal dropout rate and default dropout rate (see Table 2). '*' indicates models with optimal dropout.

Model	**NPMI for each dataset**			
	20NG	**BBC**	**Wiki40B**	**AllNews**
CTM*	0.0896	0.0623	0.1219	0.1218
CTM	0.0774	0.0458	0.1152	0.1153
ProdLDA*	0.0513	0.0367	0.1162	0.1166
ProdLDA	-0.0907	-0.1293	0.0662	0.0498
ETM*	0.0331	0.0100	0.0841	0.0605
ETM	0.0183	-0.0033	0.0662	0.0494

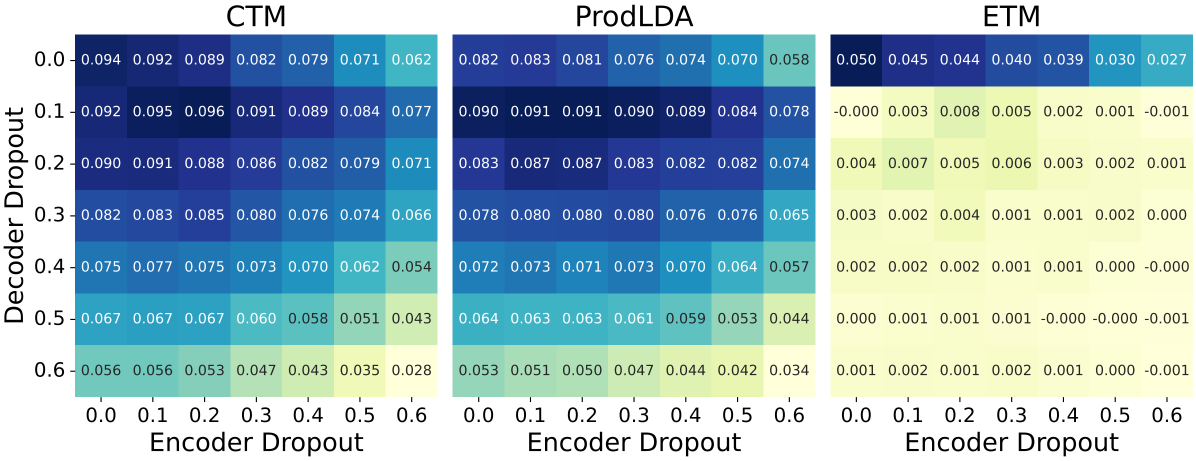
Table 7: NPMI values for different VAE-NTMs with optimal dropout rate and default dropout rate (see Table 2). '*' indicates models with optimal dropout.

Model	**TD for each dataset**			
	20NG	**BBC**	**Wiki40B**	**AllNews**
CTM*	0.7283	0.6295	0.7883	0.7871
CTM	0.7175	0.51	0.7671	0.7409
ProdLDA*	0.7644	0.6902	0.7829	0.7640
ProdLDA	0.5594	0.6861	0.5071	0.4061
ETM*	0.2776	0.1108	0.5973	0.4561
ETM	0.1949	0.0902	0.4599	0.3659

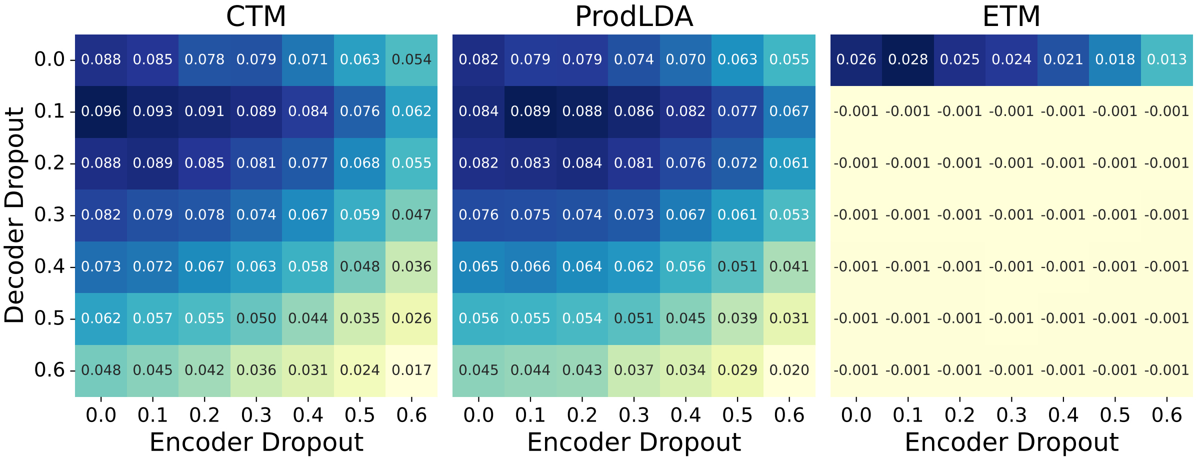
Table 8: Topic Diversity values for different VAE-NTMs with optimal dropout rate and default dropout rate (see Table 2). '*' indicates models with optimal dropout.



(a) `BBC`

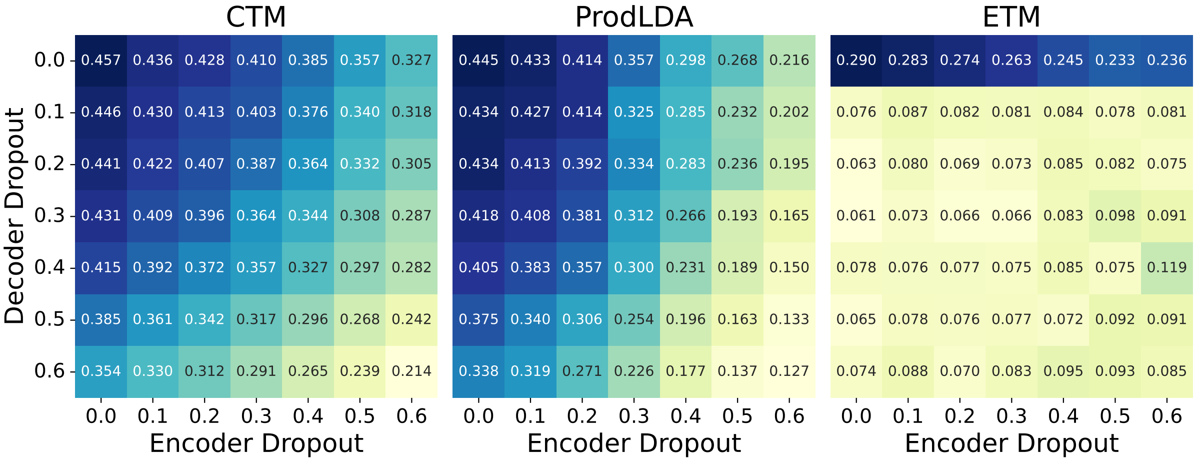


(b) `Wiki40B`



(c) `AllNews`

Figure 5: Change in **topic quality** for $(E_p, D_p) \in [0.0,\ 0.6] \times [0.0,\ 0.6]$ with a increment of 0.1.



CTM

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.457	0.436	0.428	0.410	0.385	0.357	0.327
0.1	0.446	0.430	0.413	0.403	0.376	0.340	0.318
0.2	0.441	0.422	0.407	0.387	0.364	0.332	0.305
0.3	0.431	0.409	0.396	0.364	0.344	0.308	0.287
0.4	0.415	0.392	0.372	0.357	0.327	0.297	0.282
0.5	0.385	0.361	0.342	0.317	0.296	0.268	0.242
0.6	0.354	0.330	0.312	0.291	0.265	0.239	0.214

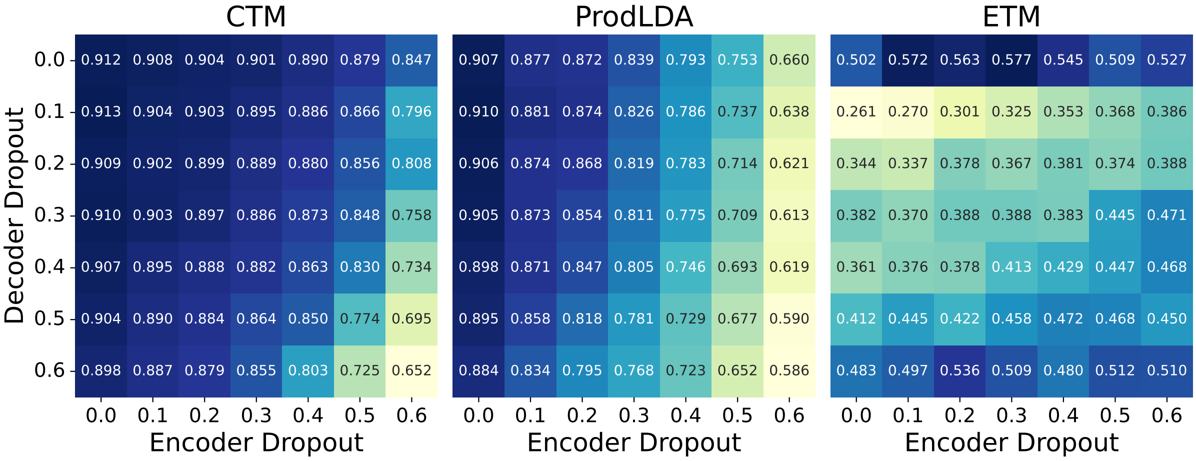
ProdLDA

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.445	0.433	0.414	0.357	0.298	0.268	0.216
0.1	0.434	0.427	0.414	0.325	0.285	0.232	0.202
0.2	0.434	0.413	0.392	0.334	0.283	0.236	0.195
0.3	0.418	0.408	0.381	0.312	0.266	0.193	0.165
0.4	0.405	0.383	0.357	0.300	0.231	0.189	0.150
0.5	0.375	0.340	0.306	0.254	0.196	0.163	0.133
0.6	0.338	0.319	0.271	0.226	0.177	0.137	0.127

ETM

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.290	0.283	0.274	0.263	0.245	0.233	0.236
0.1	0.076	0.087	0.082	0.081	0.084	0.078	0.081
0.2	0.063	0.080	0.069	0.073	0.085	0.082	0.075
0.3	0.061	0.073	0.066	0.066	0.083	0.098	0.091
0.4	0.078	0.076	0.077	0.075	0.085	0.075	0.119
0.5	0.065	0.078	0.076	0.077	0.072	0.092	0.091
0.6	0.074	0.088	0.070	0.083	0.095	0.093	0.085

(a) `20NG`



CTM

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.912	0.908	0.904	0.901	0.890	0.879	0.847
0.1	0.913	0.904	0.903	0.895	0.886	0.866	0.796
0.2	0.909	0.902	0.899	0.889	0.880	0.856	0.808
0.3	0.910	0.903	0.897	0.886	0.873	0.848	0.758
0.4	0.907	0.895	0.888	0.882	0.863	0.830	0.734
0.5	0.904	0.890	0.884	0.864	0.850	0.774	0.695
0.6	0.898	0.887	0.879	0.855	0.803	0.725	0.652

ProdLDA

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.907	0.877	0.872	0.839	0.793	0.753	0.660
0.1	0.910	0.881	0.874	0.826	0.786	0.737	0.638
0.2	0.906	0.874	0.868	0.819	0.783	0.714	0.621
0.3	0.905	0.873	0.854	0.811	0.775	0.709	0.613
0.4	0.898	0.871	0.847	0.805	0.746	0.693	0.619
0.5	0.895	0.858	0.818	0.781	0.729	0.677	0.590
0.6	0.884	0.834	0.795	0.768	0.723	0.652	0.586

ETM

	0.0	0.1	0.2	0.3	0.4	0.5	0.6
0.0	0.502	0.572	0.563	0.577	0.545	0.509	0.527
0.1	0.261	0.270	0.301	0.325	0.353	0.368	0.386
0.2	0.344	0.337	0.378	0.367	0.381	0.374	0.388
0.3	0.382	0.370	0.388	0.388	0.383	0.445	0.471
0.4	0.361	0.376	0.378	0.413	0.429	0.447	0.468
0.5	0.412	0.445	0.422	0.458	0.472	0.468	0.450
0.6	0.483	0.497	0.536	0.509	0.480	0.512	0.510

(b) `BBC`

Figure 6: Accuracy scores for $(E_p, D_p) \in [0.0,\ 0.6] \times [0.0,\ 0.6]$ (step $= 0.1$) in document classification task.

A Psycholinguistic Analysis of BERT's Representations of Compounds

Lars Buijtelaar
University of Amsterdam
lars.buijtelaar@student.uva.nl

Sandro Pezzelle
ILLC, University of Amsterdam
s.pezzelle@uva.nl

Abstract

This work studies the semantic representations learned by BERT for compounds, that is, expressions such as *sunlight* or *bodyguard*. We build on recent studies that explore semantic information in Transformers at the word level and test whether BERT aligns with human semantic intuitions when dealing with expressions (e.g., *sunlight*) whose overall meaning depends—to a various extent—on the semantics of the constituent words (*sun*, *light*). We leverage a dataset that includes human judgments on two psycholinguistic measures of compound semantic analysis: lexeme meaning dominance (LMD; quantifying the weight of each constituent toward the compound meaning) and semantic transparency (ST; evaluating the extent to which the compound meaning is recoverable from the constituents' semantics). We show that BERT-based measures moderately align with human intuitions, especially when using contextualized representations, and that LMD is overall more predictable than ST. Contrary to the results reported for 'standard' words, higher, more contextualized layers are the best at representing compound meaning. These findings shed new light on the abilities of BERT in dealing with fine-grained semantic phenomena. Moreover, they can provide insights into how speakers represent compounds.

1 Introduction

Compounds such as *sunlight* or *bodyguard* are an interesting benchmark to probe the semantic representations learned by any NLP models. On the one hand, compounds that are part of a language lexicon (i.e., lexicalized compounds; Gagné and Spalding, 2006) have their own (sets of) established meaning(s). As such, they are lexical items just like any other word. On the other hand, the semantic status of compounds is special since their meaning is the result of the combination of the meaning of two words (hence, the constituents). According to psycholinguistic evidence, this semantic relation does not disappear with lexicalization. Indeed, speakers actively combine constituent meanings when processing both novel and lexicalized compounds (Gagné and Spalding, 2009; Ji et al., 2011; Marelli and Luzzatti, 2012; Marelli et al., 2014).

In this work, we argue that NLP systems capable of faithfully representing word meanings should account for these aspects. For example, to acknowledge that the meaning of *handgun* relies more on the semantics of *gun* than of *hand* (indeed, a *handgun* is a type of *gun*). Or, that the meaning of *sunlight* is more directly recoverable from the semantics of its constituents (it is more transparent) than is the meaning of *muskrat* (which is very opaque).

Transformer-based encoders such as BERT (Devlin et al., 2019) are shown to produce word representations that align well with human semantic intuitions, particularly at their lower layers (Bommasani et al., 2020; Vulić et al., 2020). This suggests that these models are effective in encoding the meaning of a word, without any additional task-specific fine-tuning. However, these conclusions are based on evaluations that explore semantic relations between words, such as pairwise similarity patterns—not between words and their parts.

In parallel, BERT's contextualized embeddings have been leveraged for tasks that involve lexical composition. For example, to learn compound representations that are effective in predicting the *literality* of a compound or its semantic interpretation (Shwartz and Dagan, 2019). In this case, mixed results were reported. While BERT-based models are effective to judge, e.g., that *market* (but not *flea*) has a literal meaning in *flea market*, they are far behind humans in predicting, e.g., that *body part* stands for *part that makes up a body*. Crucially, these results were obtained by training a binary classifier on the top of BERT's embeddings. Since the encoder parameters were updated during learning, no conclusions can be drawn on the ef-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2230–2241
May 2-6, 2023

fectiveness of BERT's embeddings in dealing with these and similar fine-grained semantic aspects.

In this work, we leverage a dataset that includes human judgments on two psycholinguistic measures of compound semantic analysis: lexeme meaning dominance (LMD) and semantic transparency (ST). The former quantifies the semantic weight of each constituent toward the compound meaning. For example, that *gun* has more semantic weight in *handgun* than *hand* does. The second evaluates the extent to which the compound meaning is recoverable from the semantics of the constituents. For example, that *handgun* is very transparent, while *muskrat* is much less so.

We test whether, and to what extent, the measures of LMD and ST that we obtain from BERT's representations of compounds and compound constituents align with human judgments. We carry out comprehensive experiments on model versions, contexts, pooling methods, layers.[1] We show that:

- BERT is moderately aligned with human intuitions on both measures, which confirms the effectiveness of the model in accounting for the fine-grained semantic aspects captured by LMD and ST. At the same time, LMD is substantially more predictable than ST;
- only representations extracted from words in a context (in a sentence), but not without a context (in isolation), are aligned with human intuitions, which reflects BERT's struggle to handle out-of-context words. Moreover, the highest correlations are achieved in higher, deeply contextualized layers. This could be due to the nature of the semantic evaluation subtending LMD and ST, which likely requires relying on a specific semantic interpretation of the compound rather than on abstract lexico-semantic information;
- both BERT$_{\text{base}}$ and BERT$_{\text{large}}$ outperform the GloVe (Pennington et al., 2014) baseline, with BERT$_{\text{large}}$ achieving the best results overall. This confirms the effectiveness of BERT models to represent word-level semantics, in line with previous work (Bommasani et al., 2020; Vulić et al., 2020);
- BERT accounts for the left and right constituents equally when representing the semantics of a compound. Moreover, these representations appear to encode the complex semantic and syntactic relation tying the constituents.

2 Related Work

2.1 Compound Semantics in Psycholinguistics

Compounds are one of the favorite subjects of psycholinguistic research. One of the reasons is that they are extremely productive: a new combination of two (or more) words can be generated at any time and get lexicalized through language use (Gagné and Spalding, 2006). Indeed, compounds have been considered to serve as a "backdoor into the lexicon" (Downing, 1977). While understanding novel compounds clearly involves accessing both the meaning of the constituents and the semantic relation tying them together, recent psycholinguistic evidence has shown that an active combination of the meaning of the constituent words is routinely in place also for lexicalized compounds (Gagné and Spalding, 2009; Ji et al., 2011; Marelli and Luzzatti, 2012; Marelli et al., 2014). Indeed, most psycholinguistic research in this field focuses on the constituents and their relation with compounds. For example, to study and quantify the role of frequency, semantic transparency, or headedness (Gagné and Spalding, 2009; Marelli et al., 2009; Marelli and Luzzatti, 2012; Juhasz et al., 2015).[2]

Recently, a few studies leveraging methods from NLP have been carried out to either reproduce or quantify some of these aspects. By typically using *static* embeddings (Mikolov et al., 2013; Pennington et al., 2014) and compositional models of distributional semantics (Mitchell and Lapata, 2010; Guevara, 2010), these approaches have proven successful in building compound representations that approximate, e.g., the different semantic and syntactic role of a compound's modifier and head, semantic transparency, plausibility of a novel combination or syntax-based categorizations (Günther and Marelli, 2016; Marelli et al., 2017; Günther and Marelli, 2019; Pezzelle and Marelli, 2020).

Though powerful, these methods have one crucial limitation, namely, they require training a set of parameters via supervised learning to obtain representations for compounds that are novel or simply not present in the corpus. Transformer-based encoders such as BERT (Devlin et al., 2019) lift this limitation. Without any additional training or

---

[1] Data and code are made available at `https://github.com/lars927/compounds-analysis-bert`

[2] Headedness refers to the property of having a head—in English, typically the right constituent. The left is the modifier.

fine-tuning, in fact, they can represent any novel or unseen word—provided that it can be divided into known subwords. Since compounds involve a meaningful combination of two constituents, they represent an interesting benchmark to test the representations by these models.

2.2 Word Representation in Transformers

A recent line of work started to investigate the type of semantic information encoded in the embeddings by pre-trained Transformer encoders. While this was a classical benchmark to evaluate static, word-level embeddings learned by previous generation models, e.g., word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014), the problem appears less trivial for current state-of-the-art NLP models (Westera and Boleda, 2019; Mickus et al., 2020; Lenci et al., 2022). Indeed, the embeddings by Transformer-based encoders are *contextualized*, i.e., affected by both the surrounding context and the position within a sentence. Moreover, they often represent subwords rather than whole words.

By means of a simple method to pool the various contextualized embeddings learned for a word into a single, static embedding, Bommasani et al. (2020) showed that these representations align with human judgments of semantic similarity better than how previous-generation ones do. In particular, lower layers perform the best, which reveals that these layers encode abstract, lexico-semantic information. Similar findings were reported by Vulić et al. (2020), who extended the investigation to other five languages than English. Taken together, these results are complementary to the findings that representations in higher layers tend to become more context-specific (Ethayarajh, 2019) and to better encode word senses (Reif et al., 2019).

Recent work (Shwartz and Dagan, 2019) leveraged BERT embeddings to obtain representations for compounds by means of (trained) lexical composition models—similarly to how it was done for compositional distributional semantic models. However, to the best of our knowledge, no work to date has explored how Transformer-based encoders represent compounds. The most relevant study in this direction is the one by Pinter et al. (2020), which focused on BERT's representations for blends (i.e., words such as *shoptics*, resulting from the merging of *shop* and *optics*) and included a comparison with novel compounds. They reported an overall high similarity between the compound and the constituents, slightly increasing through the layers.

By focusing on lexicalized compounds from a psycholinguistic angle, we are the first to study how BERT represents these complex expressions.

compound	**LMD [0,10]**	**ST [1,7]**
handgun	8.13 →	6.29 ↑
bodyguard	7.27 →	5.64 ↑
policeman	3.07 ←	6.13 ↑
wartime	3.47 ←	6.31 ↑
muskrat	7.53 →	2.80 ↓
primrose	7.93 →	2.00 ↓
milestone	3.36 ←	2.21 ↓
cheapskate	2.00 ←	2.00 ↓

Table 1: A few examples from the dataset with either high ↑ or low ↓ ST and either low ← or high → LMD. E.g., the meaning of *handgun* is deemed highly transparent and based more on the right than the left constituent.

3 Data

We use a psycholinguistic dataset of human judgments on compound LMD and ST (Juhasz et al., 2015). The dataset includes 629 lexicalized English compounds annotated by 189 participants for various variables.[3] LMD is a score that captures which of the two constituents of a compound is semantically dominant for the compound meaning. It ranges in [0,10], where 0 means totally dependent on the left constituent and 10 means totally dependent on the right constituent. In Table 1, we report a few examples from the dataset. As can be seen, compounds such as *handgun* or *muskrat* have a high LMD, i.e., the right constituent is semantically dominant. In contrast, compounds such as *policeman* or *milestone* have a low LMD, i.e., the left constituent is semantically dominant.

ST is defined as a score that quantifies the degree to which the meaning of a compound can be inferred or recovered from the meaning of the constituents: the higher the ST, the more transparent the compound. The compounds *handgun* and *wartime* in Table 1, for example, are fully transparent: both the constituents contribute to their meaning. In contrast, compounds such as *primrose* or *cheapskate* are fully opaque: neither of the two constituents contributes to its meaning.

Since only the compounds, but not the constituents, are provided in the dataset, we manually

---

[3] Such as LMD, ST, age of acquisition, and imageability.

annotate each compound (e.g., *handgun*) with its left (*hand*) and right (*gun*) constituents, so to obtain a dataset of $\langle$compound, left, right$\rangle$ triplets. While doing so, we decided to discard the pseudo-compound *mushroom*. We were left with 628 triplets, that we use in our experiments.

4 Method

We test whether, and to what extent, BERT's representations of compounds and compound constituents approximate human judgments on LMD and ST. To do so, we obtain word-level representations using two versions of BERT. We experiment with representations obtained by feeding the word either in isolation or in the context of a sentence. Moreover, building on previous work, we explore various pooling methods over BERT outputs.

4.1 Models

BERT (Devlin et al., 2019) is a Transformer-based model pre-trained on a large number of English texts. It is pre-trained using two learning objectives, i.e., Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). MLM is about predicting some words that have been masked in the input. NSP is about predicting whether two concatenated sentences follow each other (or not).

We experiment with two versions of BERT, i.e., $\text{BERT}_{\text{base}}$ and $\text{BERT}_{\text{large}}$. The former has 12 encoder layers stacked on top of each other, 12 attention heads, and 110M parameters. At each layer, it learns 768-d embeddings. The latter has 24 layers, 16 attention heads, and 340M parameters. It learns 1024-d embeddings. For both models, we use HuggingFace (Wolf et al., 2020) implementations.[4]

4.2 Word-Level Representations

For each triplet in the dataset, we employ BERT models to obtain representations for the compound, the left constituent, and the right constituent. We henceforth use the general term *word* to refer to any of the items in a triplet. We obtain representations for words in two conditions, *no-context* (`NC`) and *in-context* (`C`), that we describe below.

No-Context (`NC`) In this condition, we obtain a single, static representation for a word (e.g., *snowboard*) by feeding it into the model in isolation, i.e., without any surrounding context. When fed with a single word, BERT outputs embeddings for the tokens which make it up (that result from the tokenization process), as well as for the special tokens [CLS] and [SEP] at the beginning and end of the sequence, respectively. Following previous work (Vulić et al., 2020), we explore 3 methods for obtaining a word representation. These methods differ with respect to what embeddings are taken into account when building such representation:

- `nospec` This method ignores the special tokens [CLS] and [SEP]. A word representation is built by averaging the embeddings of the tokens that make up the word (*snow*, *##board*);
- `withcls` This method builds a word representation by averaging the embedding for the special token [CLS] with the embeddings of the tokens making up the word (*snow*, *##board*);
- `all` This method builds a word representation by averaging all the embeddings that are output by BERT for the sequence, i.e., [CLS], [SEP], and the tokens making up the word.

In-Context (`C`) In this condition, we follow the method by Bommasani et al. (2020) to obtain a single, static representation of a word from the N contextualized embeddings produced by BERT for that word in context. First, we average the representations of the tokens that make up a given word—as in the `NC_nospec` setting. Second, we consider all the contextualized representations for a given word and aggregate them to obtain a single representation that is not dependent on a specific context. We do this by averaging the N contextual representations of a word $w_1, \ldots, w_N$:

$$w = mean(w_1, \ldots, w_N) \tag{1}$$

To obtain contextualized vectors, we sample sentences containing items from our 628 triplets from a cleaned English Wikipedia corpus.[5] For each word, we sample all the sentences in the corpus that contain it, up to a maximum of 100 unique instances per word. The average number of instances per word in our sample is 89.3 (min 1, max 100).

We henceforth simply refer to this setting as `C`.

Experimental details Within each setting, we therefore obtain a single 768-d ($\text{BERT}_{\text{base}}$) or 1024-d ($\text{BERT}_{\text{large}}$) embedding for each compound and

---

[4] `https://huggingface.co/bert-base-uncased` `https://huggingface.co/bert-large-uncased`

[5] `https://www.lateral.io/resources-blog/the-unknown-perils-of-mining-wikipedia`

model	setting	metric (best layer)	
		MAE ↓	Spearman ρ ↑
GloVe	–	**0.945**	0.541
BERT$_{\text{base}}$	`NC_nospec`	1.095 (11)	0.375 (11)
	`NC_all`	1.072 (11)	0.384 (11)
	`NC_withcls`	1.071 (11)	0.385 (11)
	`C`	0.991 (11)	*0.563 (10)*
BERT$_{\text{large}}$	`NC_nospec`	1.130 (21)	0.247 (21)
	`NC_all`	1.105 (21)	0.247 (21)
	`NC_withcls`	1.107 (22)	0.244 (21)
	`C`	*0.966 (21)*	**0.586 (21)**

Table 2: LMD. Results in **bold** and *italic* are the best and second-best in the column, respectively. Results are from a model's best-performing layer (in parentheses).

constituent in our dataset. This operation is performed for each layer of each model—i.e., 12 layers in BERT$_{\text{base}}$ (1-12) and 24 layers in BERT$_{\text{large}}$ (1-24). All representations are obtained by running a pre-trained BERT in inference mode, i.e., without fine-tuning or updating the model's weights.

Baseline As a baseline, we employ static embeddings by GloVe (Pennington et al., 2014). We use the 300-d embeddings from the version of the model trained with 6B tokens.[6] Since four compounds[7] were not found in GloVe's vocabulary, for this baseline we obtain results for 624 triplets.

4.3 Predicting Psycholinguistic Measures

LMD is a scalar in [0,10] that quantifies the relative semantic role of each constituent toward the meaning of the whole compound: The higher the value, the more the compound's semantics depends on the right constituent. Using BERT's representations for a ⟨compound, left, right⟩ triplet, we therefore operationalize LMD as follows:

$$LMD(c) = 5(R - L) + 5 \tag{2}$$

where c is the compound, L is the cosine similarity in [0,1] between the left constituent and the compound, $cos(left, compound)$, and R is the cosine similarity between the right constituent and the compound, $cos(right, compound)$. The scaling and addition operations make the values range in [0,10]. If $L = 0$ and $R = 1$, then $LMD(c) = 10$. *Vice versa*, if $L = 1$ and $R = 0$, $LMD(c) = 0$.

ST is a scalar in [1,7] that quantifies the degree to which the meaning of a compound can be inferred from the meaning of the constituents. The higher the value, the more the compound semantics can be inferred from the two constituents' meanings. Using BERT's representations for ⟨compound, left, right⟩, we operationalize ST as follows:

$$ST(c) = \frac{6(L + R)}{2} + 1 \tag{3}$$

where c, L, and R are defined as above. The scaling and addition operations make the values range in [1,7]. If $L = 1$ and $R = 1$, then $ST(c) = 7$. *Vice versa*, if $L = 0$ and $R = 0$, $ST(c) = 1$.

4.4 Evaluation

We evaluate the effectiveness of each model in approximating human LMD and ST by means of two metrics: mean absolute distance (MAE) and Spearman correlation (ρ) between the predicted and human values. For MAE, the lower the distance, the better. For ρ, the higher the correlation, the better.

In the next section, we report results by all models in all settings in approximating LMD and ST.

5 Results

5.1 Lexeme Meaning Dominance

In Table 2, we report the results by (the best layer of) each model on LMD in the various settings. Several key observations can be made. First, both BERT models achieve moderate positive correlation[8] (close to 0.6) with human judgments, with BERT$_{\text{large}}$ outperforming BERT$_{\text{base}}$ by some margin. On the one hand, this indicates that BERT's representations do a fairly good job in accounting for the relative semantic weight of each constituent in a (lexicalized) compound. On the other hand, it suggests that more data and parameters play a role in approaching human intuitions.

Second, BERT models outperform GloVe in terms of correlation. This indicates that BERT's embeddings not only encode sensible semantic information (in line with previous findings; see Bommasani et al., 2020; Vulić et al., 2020) but also align with human semantic intuitions to a greater extent than the previous-generation GloVe model. However, it is worth noting that BERT models outperform GloVe only in `C`, but not in `NC`. This clearly shows that BERT's embeddings have an advantage over GloVe's ones only when leveraging information in the surrounding context and reveals

---

[6] `https://nlp.stanford.edu/data/glove.6B.zip`

[7] Namely, *livelong*, *sunlamp*, *dunghill*, and *handclasp*.

[8] As per standard interpretation (Prion and Haerling, 2014), we consider ρ correlations from ±0.41 to ±0.60 as moderate.

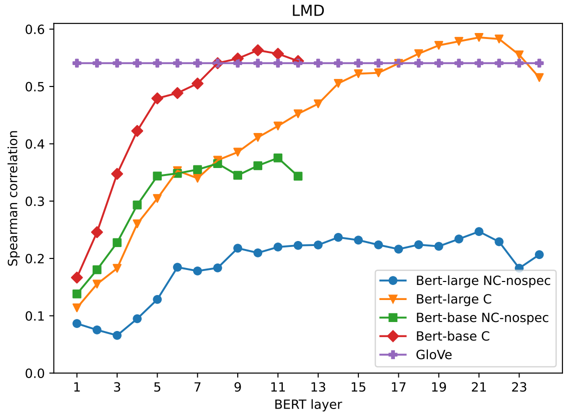


Figure 1: LMD. ρ against model layers. For out-of-context BERT models, we only report the best-performing `nospec` setting. Best viewed in color.

that BERT struggles to represent out-of-context words, likely due to its architecture and training regime. In A.1, we report that the lack of *any* surrounding context in `NC` is indeed detrimental to model representations, though sensible contextual information in `C` is needed to properly approximate LMD. Moreover, GloVe achieves the lowest MAE, which shows that these embeddings are effective in approximating the raw LMD values—though contextualized `C` BERT representations are better at capturing the overall pattern of similarity.[9]

Third, the pattern of correlation values over the model's layers highlights the importance of contextualization for compound representation. As can be seen in Figure 1, best-performing `C` BERT models show an almost constant increasing trend, with the highest correlation being achieved in high layers—layer 9 and 20 in BERT$_{\text{base}}$ and BERT$_{\text{large}}$, respectively. This is an opposite pattern compared to what was observed in previous work, where lower layers were found to encode most lexical semantic information (Bommasani et al., 2020; Vulić et al., 2020). These patterns do not necessarily contradict each other. Indeed, we argue that judging the semantic relationship between a compound (e.g., *handgun*) and its constituent words (*hand*, *gun*) might involve relying on a specific interpretation of the compound rather than on abstract lexico-semantic information. Since previous work showed that word senses are better encoded in deeper layers (Reif et al., 2019), this could explain why these layers are also good at capturing LMD. A similar—though flatter—trend is observed for `NC` models.

model	setting	metric (best layer)	
		MAE ↓	Spearman ρ ↑
GloVe	–	2.657	0.304
BERT$_{\text{base}}$	`NC_nospec`	0.953 (6)	0.316 (5)
	`NC_all`	1.129 (10)	0.234 (1)
	`NC_withcls`	0.989 (1)	0.275 (3)
	`C`	*0.899 (9)*	*0.415 (9)*
BERT$_{\text{large}}$	`NC_nospec`	0.989 (9)	0.195 (6)
	`NC_all`	1.118 (24)	0.113 (1)
	`NC_withcls`	1.024 (6)	0.139 (1)
	`C`	**0.876 (19)**	**0.476 (20)**

Table 3: ST. Results in **bold** and *italic* are the best and second-best in the column, respectively. Results are from a model's best-performing layer (in parentheses).

5.2 Semantic Transparency

In Table 3, we report the results by (the best layer of) each model on ST in the various settings. Several key observations can be made. First, both BERT models achieve moderate positive correlation with human judgments, with BERT$_{\text{large}}$ outperforming BERT$_{\text{base}}$. This indicates that BERT's representations are moderately effective in predicting the extent to which a compound meaning is recoverable from the meaning of its constituents and that more data and parameters help—indeed, the gap between the two BERT models is higher here than in LMD (0.06 vs 0.02). At the same time, the highest correlation achieved by BERT$_{\text{large}}$ (0.476) on ST is substantially lower than on LMD (0.586), which indicates that ST is more challenging to approximate compared to LMD.

Second, BERT models outperform GloVe on

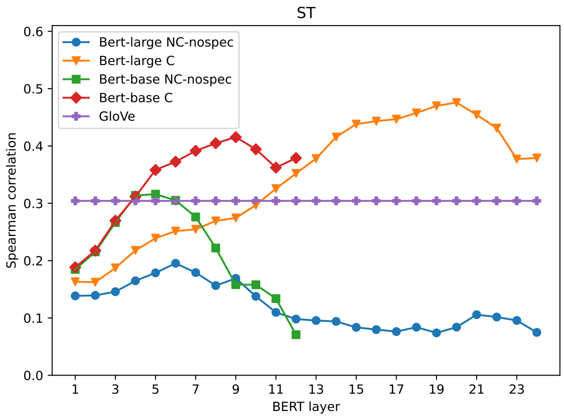


Figure 2: ST. ρ against model layers. For out-of-context BERT models, we only report the best-performing `nospec` setting. Best viewed in color.

---

[9] Recall that MAE measures the distance between the target and predicted values, while Spearman ρ quantifies strength and direction of association between the two ranked variables.

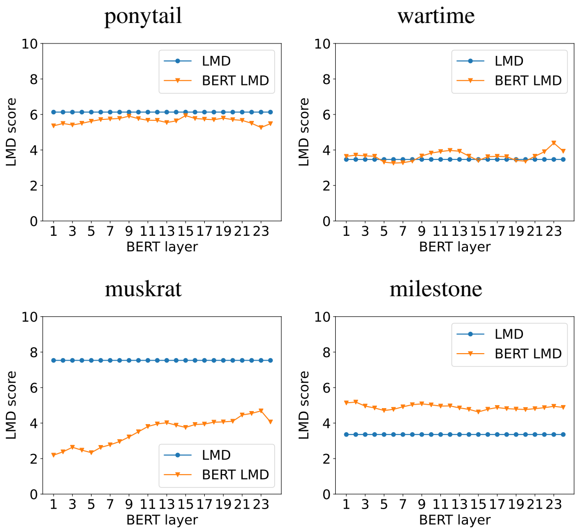


Figure 3: Examples where `C` BERT$_{\text{large}}$ is good (top) and bad (bottom) in approximating human LMD. From top left, clockwise: *ponytail*, *wartime*, *milestone*, *muskrat*.

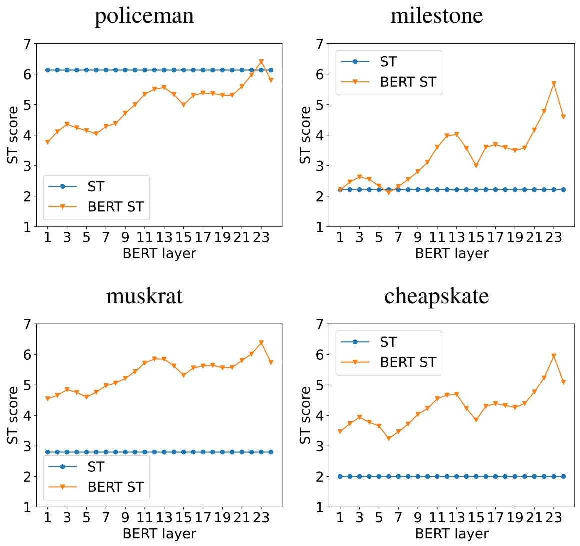


Figure 4: Examples where `C` BERT$_{\text{large}}$ is good (top) and bad (bottom) on ST. From top left, clockwise: *policeman*, *milestone*, *cheapskate*, *muskrat*.

both metrics. Though correlations are generally lower than in LMD, the gap between BERT models and GloVe is much more pronounced here. That is, BERT's contextualized embeddings have an even clearer advantage over previous-generation ones in modeling ST compared to LMD.

Third, as can be seen in Figure 2, the overall best results are achieved by `C` embeddings in high layers—layer 8 and 19 for BERT$_{\text{base}}$ and BERT$_{\text{large}}$, respectively—which replicates the findings for LMD. This confirms the role of context and contextualization for obtaining better representations of compounds.[10] Interestingly, in `NC` settings, BERT models show a different pattern compared to LMD, with correlation reaching a 'peak' within the first layers and then constantly decreasing. This suggests that decontextualized lexico-semantic information encoded in lower layers accounts for ST to some extent, on par with or even outperforming GloVe (this is the case for BERT$_{\text{base}}$ `NC_nospec`).

5.3 Examples

In Figure 3 we report some examples where the best-performing `C` BERT$_{\text{large}}$ is good (top) and bad (bottom) in approximating LMD. As can be seen, the distance between the predicted and human LMD is extremely low for *ponytail* and *wartime*, namely, a high-LMD (6.13) and a low-LMD (3.47) compound, respectively. In contrast, the distance is very large for high-LMD (7.53) *muskrat* and low-LMD (3.36) *milestone*. While the predicted LMD values are fairly stable over the layers for *ponytail*, *wartime*, and *milestone*, for *muskrat* the higher layers are better to approximate the real LMD by assigning an increasingly higher semantic weight to the *rat* lexeme. This could be due to the representation of *muskrat* becoming more 'aware'—through contextualization—of the semantic traits related to the animal domain, apparently less present in earlier layers. Also, it is interesting to note that for the compound *milestone*, which is 'exocentric' (i.e., the head is neither *mile* nor *stone*), BERT keeps predicting a conservative LMD value, which similarly weights the two constituents, over the layers. That is, contextualization does not make *mile* or *stone* become dominant in the compound (while, interestingly, human speakers consider *mile* as slightly dominant over *stone*).

In Figure 4, we report some good (top) and bad (bottom) cases for the same model in approximating ST. As can be seen, the distance between the predicted and human ST is very low for high-ST (6.31) *policeman* and low-ST (2.21) *milestone*. In contrast, the distance is large for low-ST (2.80) *muskrat* and low-ST (2.00) *cheapskate*. Moreover, it can be noted that higher layers are better than lower layers in approximating ST for *policeman*, which is highly transparent, while they are worse for *milestone*, *cheapskate*, and *muskrat*, which are very opaque. This goes hand in hand with a seemingly general trend observed in these examples:

---

[10] See the analysis in A.1 for further evidence of the role of sensible semantic context in approximating ST.

the more contextualization, the higher the ST. This could reflect a generalized increase of cosine similarity values through BERT's layers (as reported by Ethayarajh, 2019), which would lead in turn to a higher ST by virtue of how ST was operationalized; see Eq. 3. However, our results (see Figure 2) show that this increase does not correspond to a higher correlation with human judgments: indeed, correlation steadily decreases in the last 4 layers of BERT$_{\text{large}}$, where cosine similarities are highest.

Finally, while BERT struggles on both LMD and ST for *muskrat*, for *milestone* it struggles on LMD but does a good job on ST. This confirms that LMD and ST capture different semantic aspects: a good performance on one measure does not necessarily guarantee a good performance on the other.[11]

5.4 Which Factors Drive the Prediction?

To more formally investigate which factors contribute to higher predicted values of LMD and ST by the best performing model `C` BERT$_{\text{large}}$, we run two linear regression models in `R`—one for LMD, one for ST—using, for each compound, the predicted LMD/ST value by the best layers (21/20, respectively) as the dependent variable, and the following independent variables: (1) the number of tokens into which the compound was split by the tokenizer, e.g., 2 for *snowboard* (*snow*, *##board*); (2) the frequency of the compound in our dataset, i.e., the number of instances on the top of which the average representation was computed; (3) the compound concreteness; (4) the modifier (left constituent) concreteness; (5) the head (right constituent) concreteness. Concreteness values are extracted from Brysbaert et al. (2014).[12]

For LMD, both the concreteness of the head and the modifier—but not other variables—have a statistically significant role, though in the opposite direction: the higher the former, the higher the LMD (i.e., more weight to the head); the higher the latter, the lower the LMD (i.e., more weight to the modifier). This makes intuitive sense and shows that BERT assigns more 'weight' to concrete constituents. For ST, three variables have a statistically significant role in predicting higher values, and all in the same direction: the higher the number of tokens, the compound concreteness, and the modifier concreteness, the higher the ST. As for concreteness, this generally shows that BERT assigns higher similarities to concrete words. As for the effect of the number of tokens, this is an interesting finding, which reveals that BERT considers as more transparent those compounds than can be routinely broken into parts. The full tables reporting all the effects and corresponding coefficients and p-values can be found in A.2.

6 Analysis

6.1 LMD: Reversed Compounds

From the results reported in section 5.1, it appears that BERT is capable of obtaining sensible representations of compounds that encode the relation between the constituents and their respective semantic 'weight'. However, it might still be that the reported moderate correlations result from the model assigning a default high/low similarity to the constituents while being no or little aware of the semantic and syntactic (i.e., the modifier/head) relation which ties them. If that is the case, the model would consider, e.g., the contribution of *war* in *wartime* and *timewar* to be identical—though the meaning of the *reversed* compound would be intuitively very different. As such, we might expect a similar/same LMD value assigned by the model to these two compounds, and therefore a similar correlation with human judgments. Otherwise, if BERT represents a compound by genuinely accounting for the relationship between its constituents, the predicted LMD for the reversed compound (*timewar*) is likely to be different. As such, we might expect a much lower correlation with human intuitions.

In this analysis, we test this issue by re-running the LMD experiment on the *reversed* version of the compounds in our dataset, i.e., *wartime* > *timewar*, *bodyguard* > *guardbody*, etc. LMD is computed exactly as above, except that we replace the representation of the compound with that of its reversed version. Since (most of) these reversed compounds are unlikely to occur in standard *corpora* of texts, we experiment with `NC` representations. Moreover, we experiment with BERT$_{\text{base}}$ since it outperformed BERT$_{\text{large}}$ on this setting. As can be seen in Figure 5, the correlation for reversed compounds is much lower than the original one (0.11 vs 0.39). This is a good sign, which confirms that BERT does not rely on shortcuts when representing compound meanings and the relative weight of each constituent. Instead, it appears that the model can account for the semantic and syntactic relation

---

[11] Computing the correlation between LMD and ST human values in the dataset (-0.013) confirms this intuition.

[12] 23 compounds out of 628 were not present in the concreteness database and therefore excluded from the analysis.

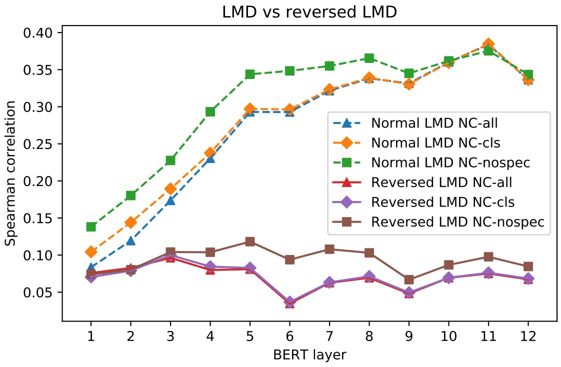


Figure 5: ρ for LMD vs LMD *reversed* values by out-of-context BERT$_{\text{base}}$ across layers. Best viewed in color.

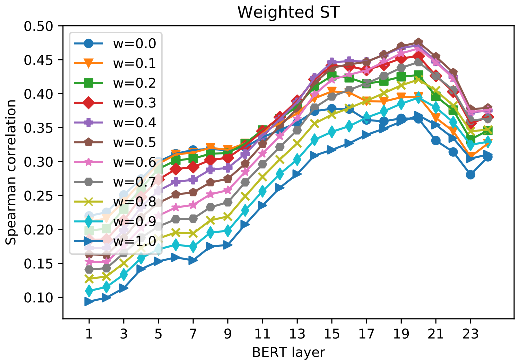


Figure 6: ρ for *weighted* ST by in-context BERT$_{\text{large}}$ across layers. Each weight stands for the weight assigned to the left constituent. Best viewed in color.

between the constituents, even when information from the surrounding context is not available.

6.2 ST: Weighted Constituents

From the results in section 5.2, it appears that BERT can approximate, to some extent, the degree to which the meaning of a compound is recoverable from the semantics of the constituents. When operationalizing ST we assumed that the two constituents are equally responsible for the overall ST—i.e., we computed the unweighted average between the two pairwise similarities. This way, a high (low) similarity between the compound and one of its constituents would not determine a high (low) ST on its own. This operationalization is in line with psycholinguistic literature, according to which we have a fully transparent compound when both lexemes contribute to its meaning and a fully opaque compound when neither of the two contribute (see, e.g., Libben, 1998). However, it is an open question whether BERT's compound representations do encode both constituents equally, or whether they disproportionately encode one constituent over the other. If the latter is the case, weighing one more than the other when computing ST would possibly result in a higher correlation with human judgments. If both constituents are equally represented in the compound embedding, instead, the unweighted version—our main experiment—would lead to the highest correlation.

In this analysis, we test this issue by computing a *weighted* version of ST where the left and right constituents are assigned different weights. For example, we assign weights 0.0 and 1.0 to the left and right constituent, respectively (in this case, only the right one, the compound's *head*, but not the left one, the *modifier*, will be responsible for ST); 0.1 and 0.9; 0.2 and 0.8; and so on. We experiment with all combinations of weights (including 0.5 and 0.5), which sums up to 11 versions of *weighted* ST.

Figure 6 reports the results of this analysis for the best-performing `C` BERT$_{\text{large}}$ model—note that the weight refers to the weight assigned to the left constituent. As can be seen, the highest correlation is achieved by the unweighted ST (weight 0.5) at layer 20. Since these are the results reported in Table 3, this finding confirms that both constituents are equally accounted for in the BERT's compound representation. Interestingly, weights that are close to 0.5 perform reasonably well (0.4 and 0.6 rank second and third, respectively), while correlation decreases the more we move away from this value.

Overall, the results of these two analyses confirm that BERT accounts for the left and right constituents equally when representing the semantics of a compound. Moreover, these representations seem to encode the complex semantic and syntactic relationship tying the compound constituents.

7 Conclusion

In this paper, we study how BERT represents the meaning of lexicalized compounds. We take a psycholinguistic angle and show that the model does a reasonably good job of making semantic judgments in line with those of human speakers. Since higher, more contextualized layers are shown to correlate best with human intuitions, we propose that speakers may access a specific, context-dependent representation when making these judgments. In future work, recent approaches to multimodal word representation in Transformers (Pezzelle et al., 2021) could be leveraged to test the role of visual grounding in compound semantics (Günther et al., 2020).

Limitations

Impact of the corpus To build contextualized representations for compounds, we sample sentences from a corpus of texts. A limitation of our work lies in the use of encyclopedic data only, which limits the number and variety of contextualized meanings a compound can have. Further attention should be paid to this aspect.

Operationalization of the measures While defining LMD and ST based on the cosine similarity between a compound and its constituent makes intuitive sense, it may not be the only (nor the best) way to operationalize the two measures. Further exploration on how to formally define them based on the model embeddings should be carried out.

Ethics Statement

Broader impact We do not see any serious ethical problem connected to this research. At the same time, we are aware of the risks associated with the development and use of large NLP models that we use in this research. Such risks include the environmental impact of the computational resources required for training and the encoding and possible amplification of biases present in the massive amounts of un-curated data the models learn from.

Acknowledgements

We would like to thank Marco Marelli for the idea of testing different weights when computing ST. We are grateful to the anonymous EACL reviewers and meta-reviewer for the insightful feedback.

References

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46:904–911.

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

Pamela Downing. 1977. On the creation and use of English compound nouns. *Language*, 53(4):810–842.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Christina L. Gagné and Thomas L. Spalding. 2006. Conceptual combination: Implications for the mental lexicon. In *The representation and processing of compound words*, chapter 7, pages 145–168. Oxford University Press Oxford, New York.

Christina L. Gagné and Thomas L. Spalding. 2009. Constituent integration during the processing of compound words: Does it involve the use of relational structures? *Journal of Memory and Language*, 60(1):20–35.

Emiliano Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, pages 33–37. Association for Computational Linguistics.

Fritz Günther and Marco Marelli. 2016. Understanding karma police: The perceived plausibility of noun compounds as predicted by distributional models of semantic representation. *PloS one*, 11(10):e0163200.

Fritz Günther and Marco Marelli. 2019. Enter sandman: Compound processing and semantic transparency in a compositional perspective. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 45(10):1872.

Fritz Günther, Marco Alessandro Petilli, and Marco Marelli. 2020. Semantic transparency is not invisibility: A computational model of perceptually-grounded conceptual combination in word processing. *Journal of Memory and Language*, 112:104104.

Hongbo Ji, Christina L. Gagné, and Thomas L. Spalding. 2011. Benefits and costs of lexical decomposition and semantic integration during the processing of transparent and opaque English compounds. *Journal of Memory and Language*, 65(4):406–430.

Barbara J Juhasz, Yun-Hsuan Lai, and Michelle L Woodcock. 2015. A database of 629 english compound words: ratings of familiarity, lexeme meaning dominance, semantic transparency, age of acquisition, imageability, and sensory experience. *Behavior research methods*, 47(4):1004–1019.

Alessandro Lenci, Magnus Sahlgren, Patrick Jeuniaux, Amaru Cuba Gyllensten, and Martina Miliani. 2022. A comparative evaluation and analysis of three generations of distributional semantic models. *Language Resources and Evaluation*, pages 1–45.

Gary Libben. 1998. Semantic transparency in the processing of compounds: Consequences for representation, processing, and impairment. *Brain and language*, 61(1):30–44.

Marco Marelli, Davide Crepaldi, and Claudio Luzzatti. 2009. Head position and the mental representation of nominal compounds: A constituent priming study in italian. *The Mental Lexicon*, 4(3):430–454.

Marco Marelli, Georgiana Dinu, Roberto Zamparelli, and Marco Baroni. 2014. Picking buttercups and eating butter cups: Spelling alternations, semantic relatedness, and their consequences for compound processing. *Applied Psycholinguistics*, pages 1–19.

Marco Marelli, Christina L Gagné, and Thomas L Spalding. 2017. Compounding as abstract operation in semantic space: Investigating relational effects through a large-scale, data-driven computational model. *Cognition*, 166:207–224.

Marco Marelli and Claudio Luzzatti. 2012. Frequency effects in the processing of italian nominal compounds: Modulation of headedness and semantic transparency. *Journal of memory and language*, 66(4):644–664.

Timothee Mickus, Mathieu Constant, Denis Paperno, and Kees van Deemter. 2020. What do you mean, BERT? Assessing BERT as a Distributional Semantics Model. *Proceedings of the Society for Computation in Linguistics*, 3.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Sandro Pezzelle and Marco Marelli. 2020. Do semantic features capture a syntactic classification of compounds? insights from compositional distributional semantics. *The role of constituents in multiword expressions*, page 33.

Sandro Pezzelle, Ece Takmaz, and Raquel Fernández. 2021. Word representation learning in multimodal pre-trained transformers: An intrinsic evaluation. *Transactions of the Association for Computational Linguistics*, 9:1563–1579.

Yuval Pinter, Cassandra L. Jacobs, and Jacob Eisenstein. 2020. Will it unblend? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1525–1535, Online. Association for Computational Linguistics.

Susan Prion and Katie Anne Haerling. 2014. Making sense of methods and measurement: Spearman-rho ranked-order correlation coefficient. *Clinical Simulation in Nursing*, 10(10):535–536.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. *Advances in Neural Information Processing Systems*, 32.

Vered Shwartz and Ido Dagan. 2019. Still a Pain in the Neck: Evaluating Text Representations on Lexical Composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Matthijs Westera and Gemma Boleda. 2019. Don't blame distributional semantics if it can't do entailment. In *Proceedings of the 13th International Conference on Computational Semantics-Long Papers*, pages 120–133.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

A Appendix

A.1 Templated Linguistic Contexts

In this analysis, we investigate the extent to which the disadvantage of `NC` compared to `C` in approximating LMD and ST is due to the lack of *any* linguistic context surrounding the compound and the constituent words.[13] Since BERT has probably seen very few examples of words out of context during training, it could be that the model is poor at handling words in isolation, which would have an impact on the resulting representations—and LMD/ST values. To test this issue, we consider the best-performing BERT$_{\text{large}}$ and use it to obtain a single, contextualized representation for words (either compounds or constituents) by embedding them in the following templated sentence: *This is a*

---

[13] We thank an anonymous reviewer for suggesting this analysis.

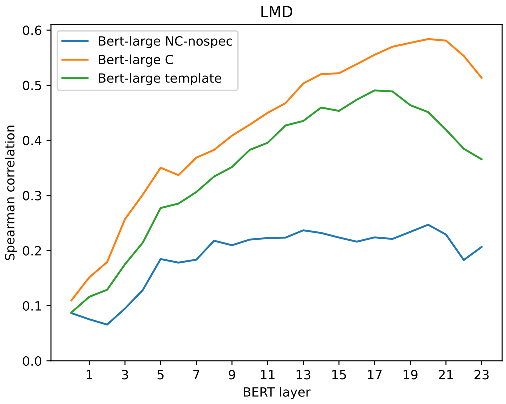


Figure 7: ρ for LMD by BERT$_{\text{large}}$ `NC` (blue), `C` (orange), and `templated` (green). Best viewed in color.

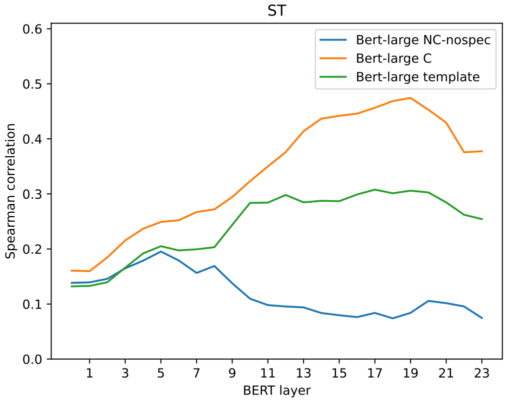
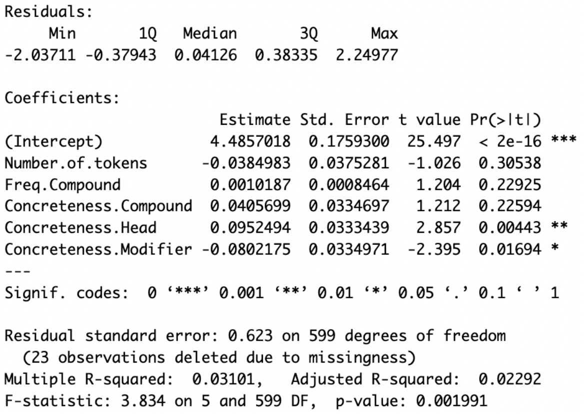


Figure 8: ρ for ST by BERT$_{\text{large}}$ `NC` (blue), `C` (orange), and `templated` (green). Best viewed in color.

`<word>`. This setting bears similarities with both `NC` and `C`. On the one hand, we compute a single representation for each word, similarly to `NC`. On the other hand, the representation of each word is contextualized (i.e., embedded in a linguistic context), though the surrounding context does not contain any meaningful semantic information. As such, we expect these representations to be better than `NC` by virtue of their higher similarity with standard training samples, but worse than `C` since they lack any sensible semantic information coming from the context surrounding the word at inference time.

The results follow the expected pattern. As can be seen in Figure 7 and 8, the correlation values obtained in the `templated` setting (best ρ for LMD: 0.491; best ρ for ST: 0.308) lie somehow in between `C` and `NC`. While this shows that embedding words in a sentence leads to a representational advantage over the out-of-context presentation (they clearly outperform `NC`), these representations are still far behind `C` and either on par with (ST) or neatly below (LMD) the GloVe baseline. This confirms that leveraging the meaningful linguistic context where compounds and constituents occur is crucial for obtaining sensible word representations that encode information on LMD and ST in line with human intuitions.

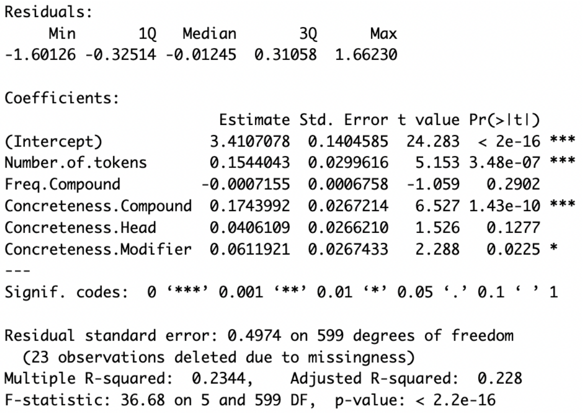


```
Residuals:
     Min       1Q   Median       3Q      Max 
-2.03711 -0.37943  0.04126  0.38335  2.24977 

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)            4.4857018  0.1759300  25.497  < 2e-16 ***
Number.of.tokens      -0.0384983  0.0375281  -1.026  0.30538    
Freq.Compound          0.0010187  0.0008464   1.204  0.22925    
Concreteness.Compound  0.0405699  0.0334697   1.212  0.22594    
Concreteness.Head      0.0952494  0.0333439   2.857  0.00443 ** 
Concreteness.Modifier -0.0802175  0.0334971  -2.395  0.01694 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.623 on 599 degrees of freedom
  (23 observations deleted due to missingness)
Multiple R-squared:  0.03101,	Adjusted R-squared:  0.02292 
F-statistic: 3.834 on 5 and 599 DF,  p-value: 0.001991
```

Figure 9: LMD. Linear regression model predicting LMD values by the best-performing `C` BERT$_{\text{large}}$ layer.



```
Residuals:
     Min       1Q   Median       3Q      Max 
-1.60126 -0.32514 -0.01245  0.31058  1.66230 

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)            3.4107078  0.1404585  24.283  < 2e-16 ***
Number.of.tokens       0.1544043  0.0299616   5.153 3.48e-07 ***
Freq.Compound         -0.0007155  0.0006758  -1.059   0.2902    
Concreteness.Compound  0.1743992  0.0267214   6.527 1.43e-10 ***
Concreteness.Head      0.0406109  0.0266210   1.526   0.1277    
Concreteness.Modifier  0.0611921  0.0267433   2.288   0.0225 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4974 on 599 degrees of freedom
  (23 observations deleted due to missingness)
Multiple R-squared:  0.2344,	Adjusted R-squared:  0.228 
F-statistic: 36.68 on 5 and 599 DF,  p-value: < 2.2e-16
```

Figure 10: ST. Linear regression model predicting ST values by the best-performing `C` BERT$_{\text{large}}$ layer.

A.2 Linear Regression Model

Figure 9 and 10 report all the effects and corresponding coefficients and p-values of the linear regression models described in Section 5.4 for LMD and ST, respectively.

Measuring Normative and Descriptive Biases in Language Models Using Census Data

Samia Touileb
University of Bergen
samia.touileb@uib.no

Lilja Øvrelid
University of Oslo
liljao@ifi.uio.no

Erik Velldal
University of Oslo
erikve@ifi.uio.no

Abstract

We investigate in this paper how distributions of occupations with respect to gender is reflected in pre-trained language models. Such distributions are not always aligned to normative ideals, nor do they necessarily reflect a descriptive assessment of reality. In this paper, we introduce an approach for measuring to what degree pre-trained language models are aligned to normative and descriptive occupational distributions. To this end, we use official demographic information about gender–occupation distributions provided by the national statistics agencies of France, Norway, United Kingdom, and the United States. We manually generate template-based sentences combining gendered pronouns and nouns with occupations, and subsequently probe a selection of ten language models covering the English, French, and Norwegian languages. The scoring system we introduce in this work is language independent, and can be used on any combination of template-based sentences, occupations, and languages. The approach could also be extended to other dimensions of national census data and other demographic variables.

1 Introduction

Pre-trained language models (LMs) may contain various types of biases, and the field of NLP has seen a lot of work in recent years on attempting to identify, mitigate, and reduce these biases. Biases can originate both from the unlabeled texts used for pre-training these LMs, and from texts and annotations used for tuning downstream classifiers. LMs have become a cornerstone in most NLP model architectures, and the extent to which they reflect, amplify, and spread the biases present in their training data is still a problematic issue to be solved.

Several efforts in this direction have focused on *gender* as a variable (Touileb and Nozza, 2022; Touileb et al., 2021; Ousidhoum et al., 2021; Nozza et al., 2021; Touileb et al., 2020; Saunders and Byrne, 2020; Bhaskaran and Bhallamudi, 2019; Cho et al., 2019; Prates et al., 2018), also in correlation with occupations (Borchers et al., 2022; Touileb et al., 2022; Bolukbasi et al., 2016). While there have been several efforts on exploring the existing biases related to these demographic variables, most work approaches the task from a normative point of view (Blodgett, 2021), where equality between the demographic distributions is prioritized.

Although normativity in this aspect is crucial for certain applications, we argue that it is also interesting to explore the task from a descriptive perspective. This is especially interesting for occupations, since a descriptive and realistic view of society already contains gender disparities. We propose that national census data, in our case about gender-occupation distributions, can offer a reliable ground truth against which model predictions can be compared. Moreover, we argue for taking both normative and descriptive assessments into account, in order to give a broader picture of the representations of demographics within LMs. This has also been partly pointed out by Blodgett et al. (2020), who stress the importance of the connection between language and social hierarchies, which has not been taken into consideration in most previous work on bias in NLP.

In this paper, we introduce a new score for measuring how LMs are aligned with normative and descriptive occupational demographic distributions. We use demographic distributions covering occupations in four countries, namely France, Norway, the United Kingdom, and the United States. We manually select gendered pronouns and nouns, as well as specific verb phrases, to construct template-based sentences, subsequently used to probe a selection of ten LMs covering the relevant languages.

Our contributions include; (i) creating novel benchmark datasets for English, French, and Norwegian based on manually crafted templates to mea-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2242–2248
May 2-6, 2023 ©2023 Association for Computational Linguistics

sure occupational gender biases, (ii) proposing a scoring system to measure normative and descriptive biases in LMs, and (iii) releasing our code and data for reproducibility.

In what follows, we give a detailed description of our new benchmark datasets in Section 2. We then, in Section 3, give a detailed description of the normative and descriptive bias scores, and present our analysis on ten LMs as proof of concept. We discuss and summarize our findings in Section 4, and conclude by discussing possible directions for future work in Section 5. The limitations of our work are discussed in the Limitations Section.

2 Benchmark datasets

In this work we develop a set of benchmark templates for English, French, and Norwegian that cover occupations in France, Norway, the United Kingdom (UK), and the United States (US). These templates are then used for probing different LMs. More details are given in what follows.

Occupations We retrieve country-specific lists of occupations and their associated (male/female) gender ratios from the national statistics bureaus of France, Norway, UK, and the US.[1] This resulted in 235 occupations from France,[2] 415 from Norway,[3] 325 from the UK,[4] and finally 314 occupations from the US.[5] All of these occupations were listed in either masculine singular or masculine plural form. As some of the languages we are focusing on inflect nouns for gender, we manually generate for each occupation in singular masculine form the corresponding forms in singular feminine, plural feminine, and plural masculine. This was performed by a native speaker of Norwegian and French, and a proficient speaker of English. Table 1 shows the top 5 female-dominated, male-dominated, and gender balanced occupations in each census data.

Templates Our work builds on the methodology of template-based probing. To measure a model's occupational biases we follow the same procedure for all languages. Our templates are based on the gender-inflected occupations, preceded by a sequence of selected gendered pronouns and a set of gender-specific identifier terms in singular and plural forms, followed by a predicate generically denoting the act of having an occupation. As an example, a template could be:

$$\underbrace{\text{The woman}}_{\text{gender-specific identifier}} \quad \underbrace{\text{worked as a}}_{\text{predicate}} \quad \underbrace{\text{nurse}}_{\text{occupation}}$$

We select 28 gender-specific identifiers, and 6 predicates for all three languages. The full list of gender-specific identifiers can be found in Table 2 and the list of predicates in Table 5 in Appendix A. Combining these identifiers and predicates with our country-specific occupations, we create a set of 12.726 template-based probes for French occupations, 69.720 for Norwegian, 50.700 for the UK, and 48.984 for the US.

The templates we created cover different grammatical tenses, such that each template is given in the past, present, and future tense. We have decided to include such a broad collection of variations to the templates to get a better representation of how occupations are correlated with genders, especially since research has shown that bias probes are sensitive to grammatical tense (Touileb, 2022).

3 Method

LMs trained with a masked language modelling objective are trained such that random tokens in the input training data are replaced with a placeholder token, `[MASK]`, which will subsequently be predicted by the trained model. Template-based approaches to probe biases take advantage of this feature of LMs. For our purposes, we mask the gendered identifier in each template-generated probe (as introduced in Section 2), and use the returned probability of each masked identifier to compute our bias scores. A masked version of the example template above would be:

$$\underbrace{\text{The [MASK]}}_{\text{gender-specific identifier}} \quad \underbrace{\text{worked as a}}_{\text{predicate}} \quad \underbrace{\text{nurse}}_{\text{occupation}}$$

Language models We select ten LMs covering the three languages English, French, and Norwegian. All models are available from the HuggingFace library (Wolf et al., 2020). We use four Norwegian models, four English models, and two French models. These are:

- NorBERT (Kutuzov et al., 2021): trained from scratch on the Norwegian newspaper corpus[6],

---

[1] All of the statistics were retrieved in October 2022.

[2] `https://dares.travail-emploi.gouv.fr/donnees/portraits-statistiques-des-metiers`

[3] `https://utdanning.no/likestilling`

[4] `https://www.nomisweb.co.uk/datasets/aps168/`

[5] `https://www.bls.gov/cps/cpsaat11.htm`

[6] `https://www.nb.no/sprakbanken/ressurskatalog/oai-nb-no-sbr-4/`

	Female-dominated occupations	Male-dominated occupations	Gender-balanced occupations
FR	midwife kindergarten assistant secretary office secretary executive secretary	navy officer and boatswain construction machinery operator pipe fitter panel beater carpenter	doctor higher education teacher medical device specialist admin. and financial executive dentist
NO	knitter midwife public health nurse skin care specialist dental health secretary	coastal skipper chief engineer scaffold builder roofer bricklayer	doctor architect lawyer politician associate professor
UK	midwife school secretary dancer and choreographer dental nurse medical secretary	roofer, roof tiler and slater carpenter and joiner construction and building supervisor bricklayer and mason vehicle technician and mechanic	barrister and judge laboratory technician paramedic industrial trainer and instructor legal professional
US	skincare specialist preschool and kindergarten teacher executive secretary speech-language pathologist dental hygienist	cement mason electrical power-line installer crane and tower operator heavy vehicle technician and mechanic bus and truck mechanic	insurance sales agent medical scientist dental laboratory technician photographer advertising sales agent

Table 1: Top 5 gender-dominated and gender-balanced occupations in census data from France (FR), Norway (NO), the United Kingdom (UK), and the United States (US). The occupations presented here are either dominated by more than 98% of either gender, or have a more balanced distribution (between 45% and 55%) between both female and male genders.

and Norwegian Wikipedia. The model comprises about two billion word tokens.

- NorBERT2[7]: the non-copyrighted subset of the Norwegian Colossal Corpus (NCC)[8] and the Norwegian subset of the C4 web-crawled corpus (Xue et al., 2021) were used to train this model from scratch. It comprises about 15 billion word tokens.
- NB-BERT_base (Kummervold et al., 2021): trained on the full version of the NCC corpus. This model used the architecture of the BERT cased multilingual model (Devlin et al., 2018). It comprises around 18.5 billion word tokens.
- NB-BERT_Large[9]: trained similarly to the NB-BERT_base model.
- BERT_base (Devlin et al., 2018) and BERT_Large: trained on English Wikipedia and Google's Books Corpus.
- RoBERTa_base (Liu et al., 2019) and RoBERTa_Large: trained on the BookCorpus, English Wikipedia, CC-news corpus (English news), OpenWebText dataset, and Stories dataset (a subset of the Common Crwal corpus).
- CamemBERT (Martin et al., 2020): trained on the OSCAR corpus (Ortiz Suárez et al., 2019), which is a multilingual corpus created by filtering the Common Crawl corpus.
- Barthez (Eddine et al., 2020): trained on the French part of the Common Crawl and Wikipedia, in addition to various smaller corpora (Eddine et al., 2020).

Scoring system The scoring system we introduce is the same for both bias scores. We will give more details on the differences of the scores in their respective sections.

For each template, and for each language, 28 gender-specific identifiers and 6 different predicates were used with each occupation. To compute the scores, we average over the gendered-identifiers and the predicates of the LMs' returned probabilities for each template. For each template, only one gender is represented (female or male). If the language inflects for gender, all components of a template reflect the gender in question, otherwise it is only reflected in the identifier.

We average the scores for a given occupation by gender, by summing and normalizing the probabilities of each identifier and the total probability

---

[7] `https://huggingface.co/ltgoslo/norbert2`

[8] `https://github.com/NbAiLab/notram/blob/master/guides/corpus_description.md`

[9] `https://huggingface.co/NbAiLab/nb-bert-large`

Norwegian	English	French
Brødrene	He	Elle
Broren	She	Elles
Dama/Damen	The aunt	Il
Damene	The aunts	Ils
Datteren	The boy	L'homme
Døtrene	The boys	L'oncle
Faren	The brother	La dame
Fedrene	The brothers	La femme
Gutten	The daughter	La fille
Guttene	The daughters	La mère
Han	The father	La soeur
Hun	The fathers	La tante
Jenta/Jenten	The girl	Le fils
Jentene	The girls	Le frère
Kvinnen	The ladies	Le garçon
Kvinnene	The lady	Le père
Mannen	The man	Les dames
Mennene	The men	Les femmes
Mødrene	The mother	Les filles
Moren	The mothers	Les fils
Onkelen	The sister	Les frères
Onklene	The sisters	Les garçons
Sønnen	The son	Les hommes
Sønnene	The sons	Les mères
Søsteren	The uncle	Les oncles
Søstrene	The uncles	Les pères
Tanten	The woman	Les sœurs
Tantene	The women	Les tantes

Table 2: Gender-specific pronouns and identifiers.

values returned by the LM, here dubbed $proba_G$, where G can be female or male (equation (1)). Then using this overall probability of a gender for a template, we average these values over all templates related to the occupation (equation (2)). More formally, for a language model *LM*, for each occupation O, there are a number of templates T, and a number of identifiers i and predicates p, reflecting a gender G. We define the bias score as follows:

$$proba_G = \frac{\sum_i T_p}{|i|} \tag{1}$$

$$score_O = \frac{\sum_{proba_G} T_O}{|T_O|} \tag{2}$$

Descriptive bias score Once the scores $score_O$ are computed, the descriptive bias score compares the percentages of distribution of occupations in the LMs to the ground truth data that comes from the respective census data of our countries of interest. We impose a threshold on the gender distributions in such a way that the category of *gender-imbalanced* occupations here corresponds to all occupations exceeding 55% of distribution for one gender, while *gender-balanced* occupations are those which percentages lie around $50\%_{\pm 5}$ for each gender.

Model	Normative	Descriptive
NorBERT	16.23	39.31
NorBERT2	3.17	34.67
NB-BERT	18.55	36.50
NB-BERT_Large	11.35	40.90
BERT_UK	18.05	35.33
BERT_large_UK	13.73	40.43
RoBERTa_base_UK	0.15	34.56
RoBERTa_large_UK	0.00	34.56
BERT_US	17.25	43.29
BERT_Large_US	12.46	48.88
RoBERTa_base_US	0.15	42.81
RoBERTa_Large_US	0.31	42.81
CamemBERT	10.46	34.10
BARThez	6.45	37.08

Table 3: Normative and descriptive occupational bias scores.

We look at the extent to which this score aligns with the census data. We compute an overall score disregarding gender, in addition to class-level scores: female dominated occupations (more than 55% in census are females), male dominated occupations (more than 55% in census are males), neutral occupations (between 45% and 55% of occupations in census for either gender).

Normative bias score The normative bias score also builds on top of the scores $score_O$, and compares the resulting distribution of occupations in LMs to a normative description of all occupations, such that percentages of either gender should be around $50\%_{\pm 5}$.

From a normative point of view, equal representations should be given to females and males. Instead of just setting the distribution to a strict value of 50-50, we decided that for either gender, the distribution should range anywhere between 45% and 55% in the census data. This to say, that if an occupation has 45% and 55% males, we consider it a balanced distribution.

4 Results and discussion

Table 3 shows the resulting normative and descriptive bias scores of the ten LMs. All scores represent percentages, *i.e.,* the percentage of model predictions that align with our normative values or descriptive demographic distributions. With no surprise, it is clear that all models exhibit fairly weak performance according to the normative bias score. The weakest performing model normatively speaking is RoBERTa (both base and large) on both UK and US statistics. BERT seems to be a bit better on UK statistics, but the difference is not signif-

Model	Neutral	Female	Male
NorBERT	1.46	22.34	15.50
NorBERT2	0.24	33.57	0.85
NB-BERT	1.46	23.68	11.35
NB-BERT_Large	0.12	33.82	6.95
BERT_UK	1.54	33.02	0.77
BERT_Large_UK	1.23	31.63	7.56
RoBERTa_base_UK	0.00	34.56	0.00
RoBERTa_Large_UK	0.00	34.56	0.00
BERT_US	2.39	39.93	0.95
BERT_Large_US.	1.75	40.09	7.02
RoBERTa_base_US	0.00	42.81	0.00
RoBERTa_Large_US	0.00	42.81	0.00
CamemBERT	0.00	0.00	34.10
BARThez	0.00	0.00	37.08

Table 4: Descriptive bias scores of gender-imbalanced and gender-neutral occupations. The two gender-imbalanced occupations cover female dominated occupations (more than 55% in census are females), and male dominated occupations (more than 55% in census are males). The gender-neutral occupations are those with distributions between 45% and 55% in census data for either gender.

icant. For the remaining languages, NorBERT2 is the worst Norwegian model normatively and BARThez is the worst of the two French models.

Results of the descriptive scores are in general higher. Most models seem to reflect the demographic occupational distribution to a certain extent. Both BERT models achieve highest descriptive scores, performing best on the US census data. While the RoBERTa models obtain the lowest performance in terms of the normative score, they rank second in the descriptive score on the US census data. NorBERT2 is still the weakest performing Norwegian model, ranking last both descriptively and normatively, while BARThez seems to yield the best descriptive score for French.

To get a more detailed overview of which types of occupations the tested models seem to represent the best, we also computed the descriptive scores of gender-imbalanced and gender-neutral occupations separately. Results can be seen in Table 4. Interestingly, all Norwegian and English models are better at identifying female-dominated occupations, while the two French models seem to only identify male-dominated occupations.

All models exhibit the lowest scores on gender-neutral occupations, hinting at the tendency that models correlate most occupations with one gender, rather then equally representing them. This would also align with the lower normative scores that we generally see.

Since the occupations in the census data differ from country to another, it is difficult to compare and rank models across languages. A fair comparison of these models is to focus on performance by country rather than across them. Even if we state that some models are the best or worst using one scoring system, the country-level scores are the most important measure of bias in the models.

5 Conclusion

We have introduced a new scoring system for measuring occupational biases in pre-trained language models. The scoring system allows the attribution of two scores: a normative score and a descriptive score. While the normative score sheds light on to what extent the correlations between genders and occupations are balanced, the descriptive score uses real world demographic distributions to reflect to what degree the language models reflect reality.

As a proof of concept we test our scoring systems on ten language models covering the French, Norwegian, and English languages. It comes as no surprise that all models exhibit low scores when using the normative scoring, while most of them have an adequate score when measured descriptively. What is more interesting is that our scoring mechanism allows us to separate between the normative and descriptive aspects of the model properties. All templates and codes are made publicly available on our GitHub repository.[10]

While we have limited our analysis to three languages, our approach is language agnostic and only requires language specific templates and demographic statistics on the distributions of occupations with respect to gender, something most national census agencies should be able to provide. Moreover, the approach could also be extended to other dimensions of national census data and other demographic variables.

On the note of future directions, we also plan to investigate cross-cultural effects, by comparing models for different languages across the gender–occupation lists from different nations, and also including multi-lingual models. Moreover, we also plan to systematically test the impact of different text sources (used to train language models) on our bias scores.

---

[10] `https://github.com/SamiaTouileb/Normative-Descrptive-scores`

Limitations

The major limitation of our work is that we focus on a binary gender setting. We acknowledge the fact that gender as an identity spans more than just two categories, however, the demographic census data we use have only the two genders (female and male) represented.

As proof of concept, the templates we use in this work are limited to one framing of how gendered pronouns and nouns can co-occur with occupations. Extending this to more diverse templates might give a broader context and a better representation of genders in LMs.

The applicability of English models to the UK and US census data, and the French models to the French census data, might also give a skewed representation of occupations. Both English and French are spoken across many countries, which might have an effect on the representation of occupations and genders in the language models. It would therefore be interesting to investigate to what extent language-specific language models reflect census data from different countries.

Acknowledgements

This work was supported by industry partners and the Research Council of Norway (RCN) through funding to *MediaFutures: Research Centre for Responsible Media Technology and Innovation*, through the RCN's Centers for Research-based Innovation scheme, project number 309339.

References

Jayadev Bhaskaran and Isha Bhallamudi. 2019. Good secretaries, bad truck drivers? occupational gender stereotypes in sentiment analysis. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 62–68, Florence, Italy. Association for Computational Linguistics.

Su Lin Blodgett. 2021. Sociolinguistically driven approaches for just natural language processing.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of" bias" in nlp. *arXiv preprint arXiv:2005.14050*.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddingsa. *Advances in neural information processing systems*, 29.

Conrad Borchers, Dalia Gala, Benjamin Gilburt, Eduard Oravkin, Wilfried Bounsi, Yuki M Asano, and Hannah Kirk. 2022. Looking for a handsome carpenter! debiasing GPT-3 job advertisements. In *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 212–224, Seattle, Washington. Association for Computational Linguistics.

Won Ik Cho, Ji Won Kim, Seok Min Kim, and Nam Soo Kim. 2019. On measuring gender bias in translation of gender-neutral pronouns. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 173–181, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Moussa Kamal Eddine, Antoine J-P Tixier, and Michalis Vazirgiannis. 2020. Barthez: a skilled pretrained french sequence-to-sequence model. *arXiv preprint arXiv:2010.12321*.

Per Egil Kummervold, Javier de la Rosa, Freddy Wetjen, and Svein Arne Brygfjeld. 2021. Operationalizing a national digital library: The case for a norwegian transformer model. In *Proc. of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa 2021)*.

Andrey Kutuzov, Jeremy Barnes, Erik Velldal, Lilja Øvrelid, and Stephan Oepen. 2021. Large-scale contextualised language modelling for norwegian. In *Proc. of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa 2021)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2021. HONEST: Measuring hurtful sentence completion in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online. Association for Computational Linguistics.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Nedjma Ousidhoum, Xinran Zhao, Tianqing Fang, Yangqiu Song, and Dit-Yan Yeung. 2021. Probing toxic content in large pre-trained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4262–4274, Online. Association for Computational Linguistics.

Marcelo O. R. Prates, Pedro H. C. Avelar, and Luis Lamb. 2018. Assessing gender bias in machine translation – a case study with google translate.

Danielle Saunders and Bill Byrne. 2020. Addressing exposure bias with document minimum risk training: Cambridge at the WMT20 biomedical translation task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 862–869, Online. Association for Computational Linguistics.

Samia Touileb. 2022. Exploring the effects of negation and grammatical tense on bias probes. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 423–429, Online only. Association for Computational Linguistics.

Samia Touileb and Debora Nozza. 2022. Measuring harmful representations in Scandinavian language models. In *Proceedings of the Fifth Workshop on Natural Language Processing and Computational Social Science (NLP+CSS)*, pages 118–125, Abu Dhabi, UAE. Association for Computational Linguistics.

Samia Touileb, Lilja Øvrelid, and Erik Velldal. 2020. Gender and sentiment, critics and authors: a dataset of Norwegian book reviews. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, Barcelona, Spain (Online). Association for Computational Linguistics.

Samia Touileb, Lilja Øvrelid, and Erik Velldal. 2021. Using gender- and polarity-informed models to investigate bias. In *Proceedings of the 3rd Workshop on Gender Bias in Natural Language Processing*, Online. Association for Computational Linguistics.

Samia Touileb, Lilja Øvrelid, and Erik Velldal. 2022. Occupational biases in Norwegian and multilingual language models. In *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 200–211, Seattle, Washington. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online. Association for Computational Linguistics.

A Appendix

Language	Predicates
Norwegian	jobber som jobbet som skal jobbe som vil jobbe som ville jobbe som kommer til å jobbe som
English	are going to work as is going to work as a want to work as wanted to work as wanted to work as a wants to work as a will work as will work as a work as worked as worked as a works as a
French	est étaient était sera serons sont va travailler comme veulent être veux être vont travailler comme voulaient être voulais être

Table 5: Language-specific predicates.

UDAPTER - Efficient Domain Adaptation Using Adapters

Bhavitvya Malik$^{*\alpha}$, **Abhinav Ramesh Kashyap** $^{*\beta\gamma}$,
Min-Yen Kan$^{\beta}$, **Soujanya Poria**♔
$^{\alpha}$ The University of Edinburgh, Edinburgh
$^{\beta}$ National University of Singapore, Singapore
$^{\gamma}$ ASUS Intelligent Cloud Services (AICS), Singapore
♔ DeCLaRe Lab, Singapore University of Technology and Design, Singapore
b.malik-1@sms.ed.ac.uk, abhinav_kashyap@asus.com, kanmy@comp.nus.edu.sg,
sporia@sutd.edu.sg

Abstract

We propose two methods to make unsupervised domain adaptation (UDA) more parameter efficient using adapters, small bottleneck layers interspersed with every layer of the large-scale pre-trained language model (PLM). The first method deconstructs UDA into a two-step process: first by adding a *domain adapter* to learn domain-invariant information and then by adding a *task adapter* that uses domain-invariant information to learn task representations in the source domain. The second method jointly learns a supervised classifier while reducing the divergence measure. Compared to strong baselines, our simple methods perform well in natural language inference (MNLI) and the cross-domain sentiment classification task. We even outperform unsupervised domain adaptation methods such as DANN (Ganin et al., 2016) and DSN (Bousmalis et al., 2016) in sentiment classification, and we are within 0.85% F1 for natural language inference task, by fine-tuning only a fraction of the full model parameters. We release our code at *https://github.com/declare-lab/domadapter*.

1 Introduction

Fine-tuning pretrained language models (PLM) is the predominant method for improving NLP tasks such as sentiment analysis, natural language inference, and other language understanding tasks (Wang et al., 2018). However, fine-tuning forces us to modify all the parameters of the model and store one copy of the model for one task. Given the large size of current PLMs, this can be expensive. Furthermore, fine-tuning needs large-scale data to be effective and is unstable when using different seeds (Han et al., 2021).

A new approach to alleviate this is parameter-efficient fine-tuning – freezing the PLM parameters and fine-tuning only a small fraction of the parameters. Fine-tuning with adapters (Houlsby et al., 2019) is one of these methods in which small additional layers are tuned within each PLM layer. Fine-tuning with adapters has many advantages: performance comparable to full fine-tuning (He et al., 2021a), and robustness to different seeds and adversarial examples (Han et al., 2021).

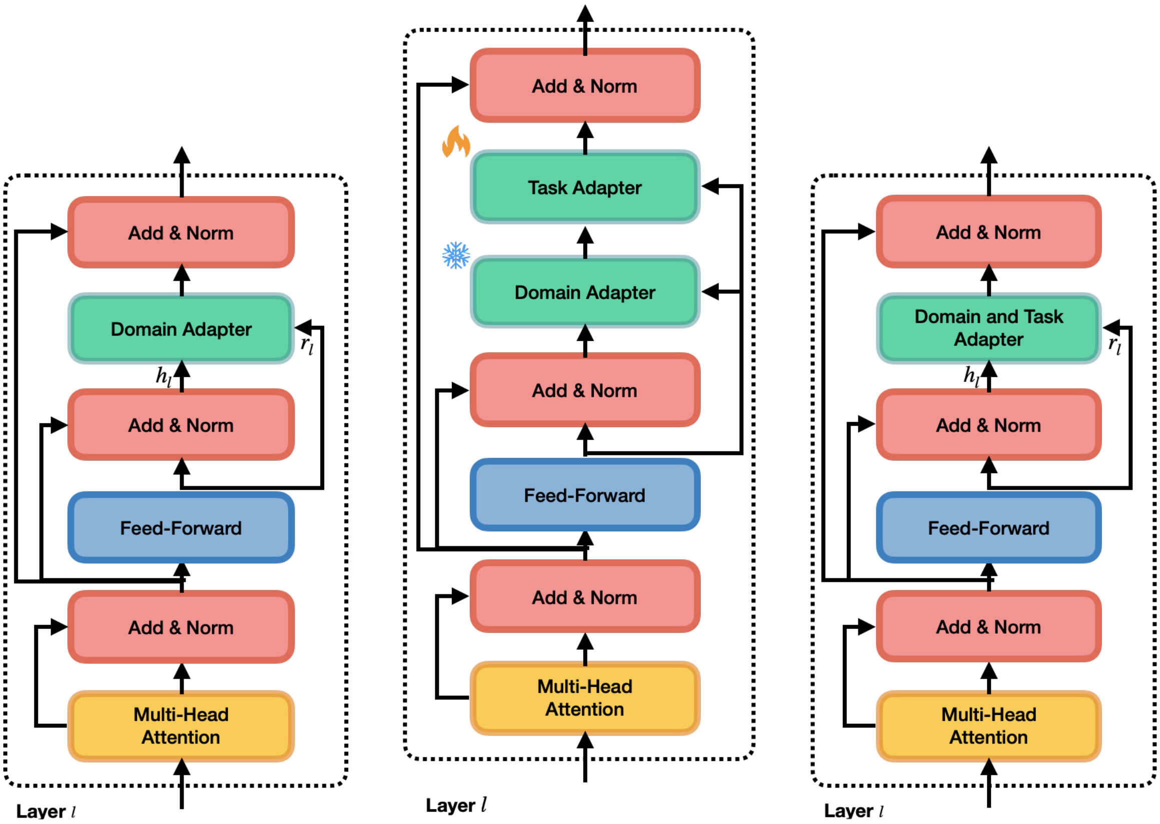
Unsupervised domain adaptation (UDA) aims to adapt models to new domains and considers situations where labeled data are available only in the source domain and unlabeled data are available in the target domain. UDA methods in general have two components: The first reduces the divergence between the source and target domains, and the second reduces the loss corresponding to a particular task (Ramesh Kashyap et al., 2021a). However, they fine-tune a large number of parameters and are susceptible to catastrophic forgetting. Adapters (Houlsby et al., 2019) can help solve these problems. However, the benefits of using adapters fine-tuning for domain adaptation have been mostly overlooked. *How well can adapter fine-tuning perform across different domains and can we make domain adaptation more efficient?* In this work, we answer these questions and propose models to perform domain adaptation using adapters.

Adapters are known to perform well in low-resource scenarios where a small amount of supervised data is available in a new domain or language (He et al., 2021b; Pfeiffer et al., 2020b). In this work, using the principles of UDA, we propose to make domain adaptation more effective using unsupervised data from the target domain. We introduce two methods that we collectively call the **U** nsupervised **D** omain **A** daptation method using ada**pters** (UDAPTER). The first method is a two-step process: First, we learn *domain adapters* – where we use a divergence measure to bring two probabilistic distributions closer together. This helps us to learn representations that are independent of the

---

[*] The first two authors contributed equally.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2249–2263
May 2-6, 2023



(a) (b) (c)

Figure 1: UDAPTER for a transformer layer l uses principles from unsupervised domain adaptation to make domain adaptation more parameter efficient. (a) The first method TS-DT-🔌 trains a Domain Adapter that reduces the marginal distribution between the domains (b) The task adapter is stacked on top of the domain adapter. and trained on an end task like sentiment analysis or natural language inference. The domain adapter is frozen during training. (c) The second method JOINT-DT-🔌 reduces the domain divergence and the task loss jointly.

domain from which they come. Second, we use the domain-invariant information learned as input to another task adapter that learns to perform an NLP task using labeled data from the source domain. We combine the two adapters by stacking them. The second method adds a single adapter without stacking, where we simultaneously reduce the divergence between domains and learn the task in the source domain.

Domain Adversarial Neural Networks (DANN) and Domain Separation Networks (DSN) are the most common methods for unsupervised domain adaptation in NLP (Ramesh Kashyap et al., 2021a). We compare our proposed methods with these strong baselines that fine-tune all model parameters, on Amazon (Blitzer et al., 2007) and the MNLI dataset (Williams et al., 2018) consisting of five domains each. UDAPTER performs better than all baselines. It achieves competitive performance compared to UDA methods by fine-tuning only a fraction of the parameters. In an era where large resources are spent to further pretrain language models on large amounts of unsupervised data to achieve domain adaptation (Gururangan et al., 2020), it is necessary to provide cheaper, faster solutions.

2 Method

Setup. We consider an NLP task (sentiment analysis) consisting of data $\mathcal{X}$ and labels $\mathcal{Y}$ (positive, negative). There exist two different distributions, called the source domain $\mathcal{D}_\mathcal{S}$ and the target domain $\mathcal{D}_\mathcal{T}$ over $\mathcal{X} \times \mathcal{Y}$. Unsupervised domain adaptation (UDA) consists of a model $\mathcal{C}$ that receives labeled input samples $\mathcal{X}_\mathcal{S} : (x_s, y_s)_{s=1}^{n_s} \sim \mathcal{D}_\mathcal{S}$ and unlabeled input $\mathcal{X}_\mathcal{T} : (x_t)_{t=1}^{n_t} \sim \mathcal{D}_\mathcal{T}$. The goal of UDA is to learn a model $\mathcal{C}$ such that we perform well in the NLP task for the target domain $\mathcal{D}_\mathcal{T}$.

The popular method in UDA is to learn representations that are invariant in the input domain and still have sufficient power to perform well in the source domain (Ganin et al., 2016; Bousmalis et al., 2016). Then according to the theory of domain divergence (Ben-David et al., 2010) shows that the error in the target domain is bounded by the error in the source domain and the divergence. The unsupervised domain adaptation method thus consists of two components: the reduction of the divergence measure and a classifier for the source domain. A new classifier must be learned for every pair of source-target domains, and the method fine-tunes a large number of parameters.

UDAPTER makes unsupervised domain adaptation more parameter efficient (cf. § 2.1, § 2.2) using adapters. We follow the framework proposed by Houlsby et al. (2019) where small bottleneck layers are added to the transformer layers, fine-tuning only the adapter parameters while keeping the other parameters frozen, and propose the following.

2.1 Two-Step Domain and Task Adapters

Domain Adapters. To learn domain-invariant representations, we first train a domain adapter. The adapter architecture follows the work of Pfeiffer et al. (2021), which consists of a simple down-projection followed by an up-projection. In a transformer layer l, let h_l be the hidden representation of the layer **Add & Norm** and let r_l be the representation of the layer **Feed-Forward** (Figure 1a), then the adapter makes the following transformation and calculates a new hidden representation.

$$dom_l = W_{up} \cdot f(W_{down} \cdot h_l) + r_l \tag{1}$$

where f is a nonlinear function such as RELU, $W_{down} \in \mathbb{R}^{h \times d}$ projects the hidden representations down to a lower dimension, $W_{up} \in \mathbb{R}^{d \times h}$ projects them back to a higher dimension, and $d \ll h$. We pass a sample from the source domain $(x_s^{src}) \sim \mathcal{D_S}$ and a sample from the target domain $(x_t^{trg}) \sim \mathcal{D_T}$ through the adapters in layer l and obtain their representations h_l^{src} and h_l^{trg}, respectively. We then reduce the divergence between these representations.

$$\Delta_l = div(dom_l^{src}, dom_l^{trg}) \tag{2}$$

Here $div(\cdot)$ is the divergence function such as the correlation alignment (CORAL) (Sun et al., 2016), the central moment discrepancy (CMD) (Zellinger et al., 2017) or the multi-kernel maximum mean discrepancy (MK-MMD) (Gretton et al., 2012; Bousmalis et al., 2016). In this work, we use MK-MMD for all of our experiments, since it performed the best[1]. Similar ideas are used to adapt representations in computer vision models (Long et al., 2019; Sun and Saenko, 2016). The final divergence loss considers all L layers.

$$\mathcal{L}_{div} = \sum_{l=1}^{L} \Delta_l \tag{3}$$

Task Adapters. Task adapters are stacked with frozen domain adapters. We pass the representations dom_l from the previous step and the supervised data from the source domain $(x_s^{src}, y_s^{src}) \sim \mathcal{D_S}$. Task adapters have the same architecture as domain adapters and perform the following.

$$task_l = W_{up} \cdot f(W_{down} \cdot dom_l^{src}) + r_l \tag{4}$$

The goal of these task adapters is to learn representations that are task-specific. Only task adapters are updated when training on the end task (sentiment classification, natural language inference) and all other parameters, including domain adapters, are frozen. Regular cross-entropy loss is reduced during training of task adapters.

$$\mathcal{L}_{task} = softmax\_ce(W_{task} \cdot h_L) \tag{5}$$

h_L is the hidden representations of the last layer of the transformer, $W_{task} \in \mathbb{R}^{h*|\mathcal{Y}|}$ where $|\mathcal{Y}|$ is the number of classes, and $softmax\_ce$ is the softmax followed by cross-entropy. This two-step process deconstructs UDA methods with a domain adapter and a task adapter. This affords composability, where task adapters can be reused for different pairs of domains (§ 3.4). However, domain and task representations can be learned jointly, as explored in the next section.

Training Process. Given a source-target domain adaptation scenario, we first train the domain adapter and save their weights. We then stack the task adapter with the domain adapter, which is trained using the supervised data from the source domain. When training the task adapter, the domain adapter is frozen. During inference, we stack the domain and task adapter.

2.2 Joint Domain Task Adapters

This method adds a single adapter that performs the reduction of the divergence measure and learns

---

[1] We also tried using CMD and CORAL and our systems performed similarly to MK-MMD

Dataset	Train	Dev	Test
MNLI	69,600	7,730	1,940
AMAZON	1,440	160	400

Table 1: Dataset statistics, showing number of train, dev, and test instances per domain.

task representations jointly. For a given supervised sample from the source domain $(x_s^{src}, y_s^{src}) \sim \mathcal{D}_\mathcal{S}$ and an unsupervised sample $(x_t^{trg}) \sim \mathcal{D}_\mathcal{T}$, let h_l^{src}, h_l^{trg} be the hidden representations of the adapters for x_s^{src} and x_t^{trg} for layer l. We reduce the following joint loss:

$$\mathcal{L} = \lambda \cdot \mathcal{L}_{task} + (1 - \lambda) \cdot \mathcal{L}_{div} \tag{6}$$

Here $\mathcal{L}_{task}$ is the task loss on the source domain supervised samples, λ is the adaptation factor.

Reducing divergence along with cross-entropy loss beyond a certain point makes training unstable and does not contribute to increased performance. Following (Ganin et al., 2016) we suppress the noisy signal from the divergence function as training progresses and gradually change λ from 0 to 1 to reduce the contribution of divergence loss using the following schedule ($\gamma = 10$ for all of our experiments):

$$\lambda = \frac{2}{1 + \exp\left(-\gamma \cdot p\right)} - 1 \tag{7}$$

Similar methods have been proposed to adapt models to other domains by Long et al. (2019) and Wu et al. (2022). Compared to the two-step process introduced earlier (§ 2.2), we need to properly control the losses to obtain optimal results and also this method does not offer composability (§ 3.4).

3 Experiments

3.1 Datasets

We evaluate our approach on two representative datasets with different tasks, both in English. Table 1 shows the details of the datasets. Every dataset has 5 domains, and we consider each domain with every other domain which results in 20 domain adaptation scenarios for every dataset, 120 experiments per method, and 1900+ experiments.

AMAZON: Multi Domain Sentiment Analysis Dataset (Blitzer et al., 2007) that contains Amazon product reviews for five different types of products (domains): Apparel (A), Baby (BA), Books (BO), Camera_Photo (C), and Movie Reviews (MR). Each review is labeled as positive or negative. We follow the setup in (Ramesh Kashyap et al., 2021a)

MNLI: The Multigenre Natural Language Inference (MNLI) corpus (Williams et al., 2018) contains hypothesis–premise pairs covering a variety of genres: Travel (TR), fiction (F), telephone (TE), government (G), and slate (S). Each pair of sentences is labeled Entailment, Neutral, or Contradiction. The train and validation data set are taken from the train set by sampling 90% and 10% samples, respectively. We use the MNLI-matched validation set as our test set.

3.2 Baseline Methods

Fully supervised. *Fine-tune* (🔥): Fine-tunes a language model using labeled data from the target domain. Serves as an upper bound of performance.

Unsupervised Domain Adaptation (UDA). *Domain Adversarial Neural Networks* (DANN): An unsupervised domain adaptation method (Ganin et al., 2016) that learns domain-invariant information by minimizing task loss and maximizing domain confusion loss with the help of gradient reversal layers. *Domain Separation Networks:* (DSN) (Bousmalis et al., 2016) improves DANN, with additional losses to preserve domain-specific information along with the extraction of domain-invariant information. `bert-base-uncased` serves as a feature extractor for both methods.

Adapter Based. DANN *Adapter* (DANN- 🔌): Similar to DANN, but we insert trainable adapter modules into every layer of a PLM. DANN *Adapter with Multiple Classifiers* (DANN- 🔌 -MC): Unlike DANN- 🔌 which involves a single task and domain classifier, here a task and domain classifier are added to each of the last 3 layers of a PLM. The representation of the last layers of a PLM is domain variant (Ramesh Kashyap et al., 2021b), and this model obtains domain-invariant information[2] (vi) Task adapter (TASK- 🔌): Adapter fine-tuning (Pfeiffer et al., 2020a) where adapters are fine-tuned in the labeled source domain and tested in the target domain. (vii) Two-step Domain and Task Adapter (TS-DT- 🔌): This work, where we first train a domain adapter that reduces the probabilistic divergence between two domains and then fine-tunes a task adapter by stacking. (viii) Joint

---

[2] We tried adding classifiers incrementally to the last few layers. Adding it to the last 3 layers performed the best.

	Fully Supervised	Unsupervised Domain Adaptation		Adapter Based				
Src →Trg	🔥	DANN	DSN	DANN-🔌	DANN-🔌-MC	TASK-🔌	TS-DT-🔌	JOINT-DT-🔌
A → BA	87.52 (1.96)	85.57 (3.72)	89.90 (0.26)	86.46 (0.26)	**88.74 (0.64)**	87.03 (0.26)	88.24 (0.76)	**88.74 (0.13)**
A → BO	86.67 (1.06)	36.48 (0.45)	84.47 (0.99)	78.41 (1.14)	83.36 (0.43)	84.15 (1.10)	84.22 (0.76)	**84.96 (0.28)**
A → C	91.62 (0.37)	57.51 (13.32)	88.56 (0.81)	87.31 (0.39)	88.75 (0.69)	**89.67 (0.32)**	88.76 (1.32)	89.39 (0.23)
A → MR	82.08 (0.78)	35.23 (1.99)	78.08 (0.46)	75.54 (0.63)	76.60 (1.06)	76.63 (0.92)	77.39 (0.13)	**77.63 (0.71**)
BA → A	89.12 (0.38)	77.52 (11.25)	87.46 (1.83)	87.72 (1.85)	88.47 (0.72)	88.33 (1.10)	89.55 (0.10)	**89.70 (0.23)**
BA → BO	86.67 (1.06)	43.45 (8.96)	82.19 (3.70)	82.89 (3.08)	83.86 (0.41)	84.61 (0.39)	84.38 (0.61)	**85.01 (0.60**
BA → C	91.62 (0.37)	47.58 (7.65)	89.68 (0.71)	86.63 (0.53)	88.73 (0.42)	**90.63 (0.33)**	87.46 (0.88)	88.64 (0.30)
BA → MR	82.08 (0.78)	50.63 (7.43)	77.88 (0.38)	74.48 (1.79)	78.07 (0.34)	78.74 (0.35)	**79.42 (0.44)**	78.44 (0.70)
BO → A	89.12 (0.38)	37.40 (1.90)	88.20 (0.51)	85.90 (0.12)	85.91 (0.25)	85.03 (0.36)	84.79 (0.75)	**87.46 (0.27)**
BO → BA	87.52 (1.96)	54.33 (12.49)	88.56 (0.44)	82.06 (1.15)	84.27 (0.11)	86.50 (0.39)	**86.84 (0.48)**	86.41 (0.79)
BO → C	91.62 (0.37)	39.43 (0.49)	88.58 (1.01)	86.94 (0.83)	87.40 (0.44)	88.44 (0.53)	87.86 (0.61)	**88.53 (0.43)**
BO → MR	82.08 (0.78)	54.23 (13.94)	79.07 (1.01)	76.19 (0.89)	79.44 (0.86)	79.44 (0.95)	**80.52 (0.61)**	78.91 (0.38)
C → A	89.12 (0.38)	60.93 (3.78)	89.76 (0.76)	87.02 (1.86)	86.63 (0.29)	87.74 (1.18)	88.53 (0.42)	**88.92 (0.44)**
C → BA	87.52 (1.96)	77.29 (3.61)	89.42 (0.70)	88.10 (1.13)	89.14 (0.30)	81.71 (2.72)	**89.72 (0.43)**	89.32 (0.42)
C → BO	86.67 (1.06)	38.21 (1.40)	85.56 (0.62)	81.18 (2.07)	83.61 (0.67)	80.55 (0.81)	84.14 (0.52)	**85.42 (0.70)**
C → MR	82.08 (0.78)	35.08 (1.94)	76.13 (0.54)	64.99 (5.91)	**74.22 (0.31)**	69.53 (1.24)	73.22 (0.48)	73.50 (0.84)
MR → A	89.12 (0.38)	37.07 (4.16)	82.64 (2.17)	81.05 (1.15)	79.56 (0.53)	82.45 (1.43)	81.93 (0.47)	**84.41 (0.43)**
MR → BA	87.52 (1.96)	38.76 (4.17)	80.59 (2.18)	77.95 (1.46)	79.33 (0.43)	81.70 (1.22)	84.28 (0.41)	**84.91 (0.36)**
MR → BO	86.67 (1.06)	42.07 (4.86)	85.13 (0.83)	82.83 (0.62)	**84.90 (1.29)**	**84.90 (0.23)**	84.47 (0.80)	84.45 (0.31)
MR → C	91.62 (0.37)	36.92 (1.86)	86.56 (0.63)	84.58 (0.46)	82.53 (0.92)	86.68 (0.65)	86.25 (0.38)	**88.37 (0.11)**
Avg	87.40 (0.91)	49.28 (5.47)	84.92 (1.03)	81.91 (1.37)	83.68 (0.50)	83.72 (0.88)	84.60 (0.57)	**85.16 (0.43)**

Table 2: F1 scores for AMAZON dataset. We report mean and standard deviation of 3 runs. The five domains are Apparel (A), Baby (BA), Books (BO), Camera_Photo (C) and Movie Reviews (MR). On average, our method outperforms all baselines. Our methods are competitive with fully unsupervised domain adaptation methods.

Domain Task Adapter (JOINT-DT-🔌) - We train a single adapter that reduces the domain and task loss jointly. For all adapter-based experiments, the PLM is frozen, and only adapter modules are trained.

Since we use adapters, we only consider other adapter based baselines and omit other methods such as Prefix-tuning (Lester et al., 2021). Also, (Zhang et al., 2021) target multidomain adaptation and use data from all the domains during training unlike our method and is not a fair comparison.

Implementation Details and Evaluation. For our experiments, we use `bert-base-uncased` (Devlin et al., 2019) available in the HuggingFace Transformers library (Wolf et al., 2020) as our backbone. Adapter implementations are from AdapterHub (Pfeiffer et al., 2020a). We follow (Pfeiffer et al., 2021) and add only one bottleneck layer after the feedforward layer.

We use the AdamW optimizer and a learning rate of $1e-4$ for all our adapter-based training and $2e-5$ otherwise. Only for the smaller AMAZON dataset, we used an adapter bottleneck size (reduction factor) of 32. For all other adapter-based experiments and datasets, we use the default adapter bottleneck size of 16. We performed experiments on three different seeds. We report the mean and standard deviation of the F1 scores . For DANN we use 0.04 as our λ and for DSN we use 0.1, 0.1, and 0.3 as our weights for three losses: reconstruction, similarity, and difference respectively. We avoid extensive hyperparameter tuning per domain adaptation scenario for efficiency.

3.3 Results

From Table 2 and Table 3 our methods TS-DT-🔌 and JOINT-DT-🔌 perform well in both AMAZON and MNLI. We find that fine-tuning the task adapter (TASK-🔌) is a strong baseline and, compared to it, we perform well in 17/20 domain adaptation scenarios in AMAZON (largest increase of 8 points for C → BA) and 19/20 domain adaptation scenarios in MNLI (largest increase of 2.2 for F → TE). One possible explanation of scenarios where our method finds the largest increase is the proximity of the two domains. The overlap in vocabularies (Figure 9 in Appendix) between C →BA in AMAZON and F → TE in MNLI is high, and our method takes advantage of learning domain-invariant information that can be used for efficient domain transfer. Our methods for learning domain-invariant information are necessary to achieve good domain adaptation.

UDAPTER is comparable to UDA methods. Compared to UDA methods where all parameters of the backbone model are fine-tuned, we perform close to them on average. JOINT-DT-🔌 performs better than DSN by 0.2% in AMAZON. We are within 0.85% in MNLI compared to DSN. Training DANN is highly unstable and produces varied re-

	Fully Supervised	**Unsupervised Domain Adaptation**		**Adapter Based**				
Src →Trg	🔥	DANN	DSN	DANN-🔌	DANN-🔌-MC	TASK-🔌	TS-DT-🔌	JOINT-DT-🔌
F →S	74.09 (0.40)	73.68 (0.21)	72.36 (0.17)	70.96 (0.03)	62.40 (4.79)	72.36 (0.36)	**73.46 (0.34)**	72.30 (0.26)
F →G	82.19 (0.12)	79.17 (0.25)	79.79 (0.21)	78.73 (0.43)	77.23 (0.33)	79.00 (0.46)	78.65 (0.25)	**79.79 (0.22)**
F →TE	78.41 (0.66)	73.72 (0.81)	75.07 (0.32)	70.89 (0.74)	71.68 (0.59)	70.83 (0.54)	**73.05 (0.70)**	71.59 (0.78)
F→TR	81.81 (0.20)	76.99 (0.19)	76.82 (0.50)	74.42 (0.18)	75.09 (0.05)	75.85 (0.19)	76.75 (0.80)	**77.07 (0.26)**
S →F	78.59 (0.34)	75.91 (0.23)	76.62 (0.38)	73.89 (0.61)	73.47 (0.28)	75.25 (0.19)	**75.52 (0.89)**	75.35 (0.56)
S →G	82.19 (0.12)	80.91 (0.46)	81.27 (0.23)	79.99 (0.36)	79.16 (0.10)	80.76 (0.40)	**81.65 (0.11)**	80.94 (0.30)
S →TE	78.41 (0.66)	74.32 (0.57)	74.27 (0.48)	72.29 (0.57)	71.89 (0.07)	72.66 (0.79)	**74.09 (0.30)**	73.38 (0.63)
S →TR	81.81 (0.20)	76.81 (0.35)	78.17 (0.20)	75.58 (0.54)	75.77 (0.39)	76.16 (0.22)	**77.31 (0.60)**	77.16 (0.18)
G →F	78.59 (0.34)	73.41 (0.73)	72.62 (0.37)	71.57 (0.68)	70.34 (0.73)	72.66 (0.31)	72.66 (0.56)	**73.56 (0.23)**
G →S	74.09 (0.40)	72.51 (0.10)	71.93 (0.25)	70.17 (0.64)	69.49 (0.40)	71.11 (0.38)	71.14 (0.21)	**71.36 (0.04)**
G →TE	78.41 (0.66)	71.52 (0.13)	72.90 (0.39)	69.45 (0.96)	68.67 (0.17)	71.40 (0.30)	71.53 (1.04)	**71.99 (0.67)**
G →TR	81.81 (0.20)	77.42 (0.54)	77.80 (0.42)	74.35 (0.22)	74.04 (0.51)	76.29 (0.10)	76.16 (0.34)	**76.79 (0.59)**
TE →F	78.59 (0.34)	75.07 (0.08)	75.17 (0.35)	72.24 (0.59)	71.49 (0.45)	**74.48 (0.33)**	73.34 (0.41)	73.89 (0.12)
TE →S	74.09 (0.40)	71.65 (0.50)	72.16 (0.23)	69.09 (1.79)	69.25 (0.31)	70.94 (0.16)	70.94 (0.55)	**71.41 (0.19)**
TE →G	82.19 (0.12)	78.57 (0.60)	79.24 (0.31)	77.80 (0.27)	76.65 (0.20)	79.24 (0.35)	79.65 (0.60)	**79.78 (0.64)**
TE →TR	81.81 (0.20)	75.72 (0.37)	77.29 (0.61)	74.67 (0.50)	74.08 (0.25)	75.27 (0.83)	**76.11 (0.91)**	75.95 (0.50)
TR →F	78.59 (0.34)	73.22 (0.92)	72.44 (0.50)	70.27 (0.45)	69.08 (0.64)	72.20 (0.49)	73.12 (0.08)	**73.13 (0.22)**
TR →S	74.09 (0.40)	70.76 (0.72)	70.97 (0.26)	68.35 (0.62)	67.23 (0.39)	70.28 (0.37)	70.67 (0.50)	**71.28 (0.38)**
TR →G	82.19 (0.12)	80.91 (0.28)	81.67 (0.37)	79.25 (0.34)	78.77 (0.32)	81.26 (0.37)	81.11 (0.42)	**81.55 (0.16)**
TR →TE	78.41 (0.66)	70.41 (1.63)	71.98 (0.50)	69.33 (0.41)	69.45 (0.39)	70.98 (0.11)	70.95 (0.19)	**71.42 (0.12)**
Avg	79.02 (0.34)	75.13 (0.48)	75.53 (0.35)	73.16 (0.55)	72.26 (0.57)	74.45 (0.40)	**74.89 (0.49)**	**74.98 (0.35)**

Table 3: F1 scores for MNLI dataset. We report mean and standard deviation of 3 runs. The five domains are Fiction (F), Slate (S), Government (G), Telephone (TE), and Travel (TR). On average, our method performs better than all baselines.

sults, especially for AMAZON with a small number of examples in each domain. Our adapter method achieves better results compared to DANN with a minimal modification of the hyperparameters.

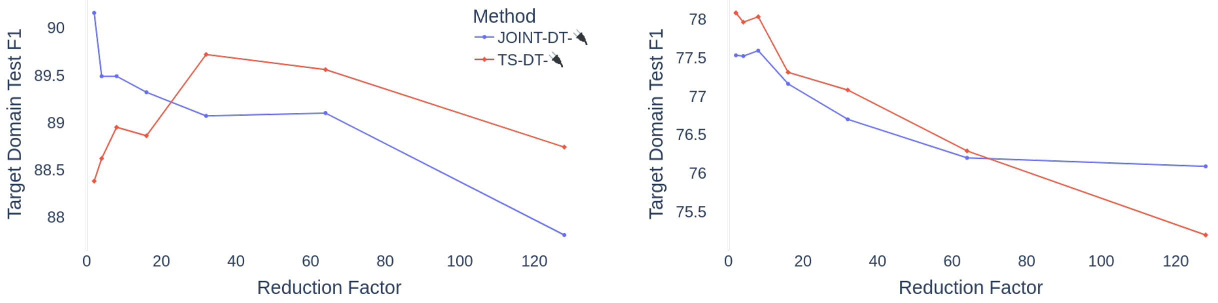
Replacing UDA Feature Extractors with Adapter Versions is insufficient. *Given that fully fine-tuned* UDA *methods perform well, can we freeze the feature extractors* UDA *methods and fine-tune only adapters and perform effective domain adaptation?* We compare our methods with DANN-🔌 and DANN-🔌-MC and outperform them both in AMAZON and MNLI. This is in line with Karouzos et al. (2021) that although domain adversarial training brings domain representations closer, it introduces distortion in the semantic space, reducing model performance. This shows that simply replacing feature extractors with their adapter versions in existing UDA methods is not an effective strategy.

Gap to Full Fine-Tuning. Fine-tuning a PLM with supervised data in the target domain is the upper bound performance for domain adaptation. The gap from full fine-tuning is greater when more data are available (3.15 in AMAZON and 4.13 in MNLI). This is not surprising, as the supervised fine-tuning works better with more data. However, while adapters perform closely to complete fine-tuning in supervised scenarios (He et al., 2021a), there is still a large gap between domain adaptation and complete fine-tuning and would require significant future work.

3.4 Further Analysis

Adapter Reduction Factor. The bottleneck size (d) of the adapters plays an important role in the final performance of the model. We show the performance of the models at various reduction factors in Figure 2. For JOINT-DT-🔌, smaller reduction factors generally perform well in both AMAZON and MNLI, with performance reducing for larger reduction factors. This shows that the JOINT-DT-🔌 method requires a greater number of parameters to reduce divergence and learn task representations together. Since TS-DT-🔌 adds two adapters, this increases the number of parameters added for the same reduction factor compared to JOINT-DT-🔌. As a result, we find that as the data scale up, relatively low reduction factors work well.

The removal of adapters from continuous layer spans. All adapters are not equal. Removing adapters from the first few layers still preserves performance (Figure 3). For JOINT-DT-🔌 and TS-DT-🔌, the F1 slowly decreases as we continually remove the adapters. However, we obtained a comparable performance after removing the adapters from layers 1-6. This suggests that adapters are effective when added to higher layers, where the divergence between domains is greater at higher layers compared to lower layers (Ramesh Kashyap



(a)

(b)

Figure 2: (a) Performance for AMAZON on the C → BA domain adaptation scenario for different reduction factors. (b) Performance for MNLI on the S → TR scenario for different reduction factors.

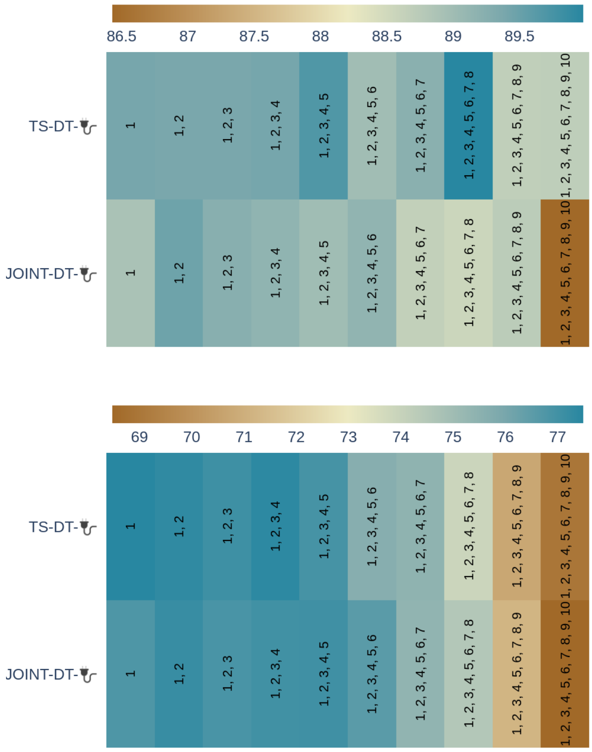


Figure 3: Shows the difference in performance when adapters are removed from certain layers (mentioned inside the cells) for the AMAZON dataset (top) and for MNLI dataset (bottom). The performance reduces if adapters are removed from certain layers

et al., 2021b). Thus we can further reduce the number of parameters for domain adaptation.

t-SNE plots. The t-SNE (van der Maaten and Hinton, 2008) plots from domain adapters are shown in Figure 4 for the data set MNLI. The lower layers have low divergence and the data from the two domains are interspersed, whereas the higher layers have high divergence. Our method effectively reduces the divergence in higher layers.

Composability. We test the composability of our two-step method TS-DT- 🔌 . We reuse the task adapter trained for C → BA and replace the domain adapter with the domain adapter of C → MR and perform inference on C → MR dataset. The initial F1 of the C → MR dataset was 73.22 and after composing it with a different task adapter, the F1 score is 72.66 – a minimal performance loss. This shows the composability of TS-DT- 🔌 .

4 Literature Review

Parameter Efficient Fine-tuning Methods. Adapters (Houlsby et al., 2019) are task-specific modules added to frozen transformer layers, with only the adapter parameters updated. Their plug-and-play characteristics and the avoidance of catastrophic forgetting have resulted in their use for NLP tasks: machine translation (Bapna and Firat, 2019), named entity recognition (Pfeiffer et al., 2020b), etc. Recently, (He et al., 2021b) have shown that they are efficient in scenarios where there is minimal supervised data. However, they neither test their performance under domain shift nor propose methods to improve adapter fine-tuning. Closely related to our method is the work of Ngo Trung et al. (2021), who learns a shared-private representation per layer, similar to DSN (Bousmalis et al., 2016). Their method requires balancing multiple loss functions, compared to our simpler two-step domain adaptation method. The stacking of adapters has been followed before by (Pfeiffer et al., 2020b) for cross-lingual tasks: learning a language adapter first and stacking

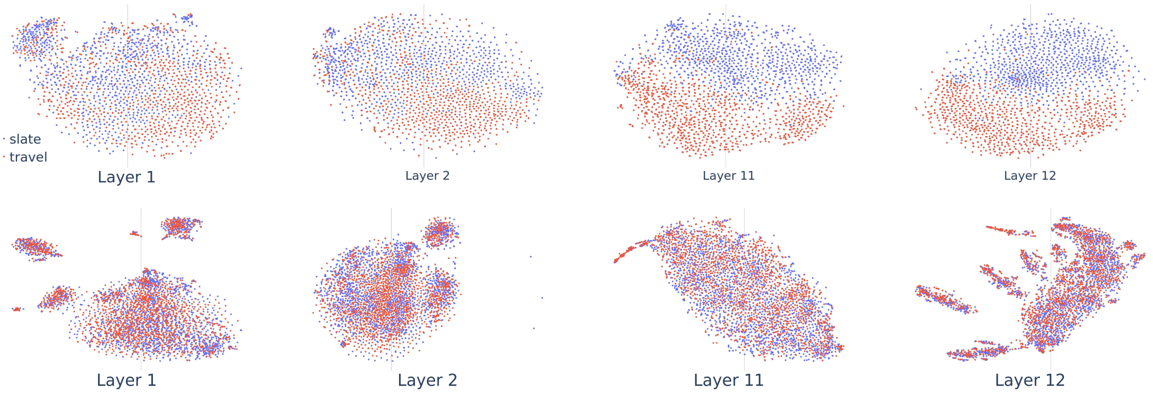


Figure 4: (top)t-SNE plots for the representations from `bert-base-uncased`. The lower layers are domain invariant while the higher layers are domain variant (bottom) tSNE plots from the domain adapter trained on the S → TR domain. We reduce the divergence using domain adapters where even higher layers are domain invariant.

a task adapter. However, one language adapter is learned per language, assumes large amounts of unsupervised data to be available in all the languages, and requires supervised data to be available to learn a task, which is not applicable for domain adaptation. Compared to other methods, we make domain adaptation more efficient using principles of unsupervised domain adaptation.

Unsupervised Domain Adaptation (UDA). Existing UDA approaches can be categorized into model-centric, data-centric, and hybrid. *Model-centric* approaches involve augmenting feature space or altering the loss function, architecture, or model parameters (Blitzer et al., 2006; Pan et al., 2010; Ganin et al., 2016) have been popular. A popular *model-centric* approach is to use adversarial training between the domain and the task classifier (Ganin et al., 2016) to extract domain-invariant information. (Bousmalis et al., 2016) in addition preserves domain-specific information. These works involve training a large number of parameters and require careful balancing of multiple loss functions. Our methods build on top of these works and make it more parameter-efficient.

Large-scale transformers pretrained on domain-specific corpora have been a norm: biomedical (Lee et al., 2019), scientific publications (Beltagy et al., 2019), among others. Another alternative is to continue pretraining generic models on domain-specific data: domain adaptive pretraining (Gururangan et al., 2020). Both solutions are expensive since a huge model has to be stored for every domain while using adapters affords storing a small number of parameters for every domain pair and can be quickly adapted to new domains.

5 Discussion

This work shows that domain adaptation in NLP can be made more efficient using adapters. We use adapters fine-tuning (Houlsby et al., 2019) proposed before and stacking of adapters that have been proposed before for a cross-lingual setting (Pfeiffer et al., 2020b) for the unsupervised domain adaptation. The approach we have discussed will make domain adaptation more practical for real-world use cases, making adaptation faster and cheaper. However, in this work, we have used `bert-base-uncased` for all of our methods. Using other backbone transformer models is part of our future work. We deal only with a classification and natural language inference task. Adapters have previously been used for machine translation (Bapna and Firat, 2019) and other generation tasks (Zhang et al., 2022). We need to explore our domain adaptation methods for other generation tasks.

In this work, we reduce the marginal distribution of the two distributions. Previous works such as Kumar et al. (2018) show that reducing only the marginal distribution is not sufficient and aligning the label distributions is necessary. However, NLP works do not consider this and would require further investigation by the community.

6 Conclusion

In this work, we propose UDApter, to make unsupervised domain adaptation more parameter-efficient. Our methods outperform other strong baselines, and we show that we can perform better than just training a task adapter on supervised data. We perform competitively to other UDA methods at a fraction of the parameters and outperform them when there is limited data – a more practical scenario. Future work should explore other parameter-efficient methods such as prefix-tuning (Li and Liang, 2021) for domain adaptation. NLP should also consider other avenues, such as continuous adaptation to new domains and adaptation to new domains when there are no data available.

7 Acknowledgments

This research is supported by the SRG grant id: T1SRIS19149 and the Ministry of Education, Singapore, under its AcRF Tier-2 grant (Project no. T2MOE2008, and Grantor reference no. MOET2EP20220-0017). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

8 Limitations

We have several limitations to our work. We have experimented with only one type of parameter-efficient method, which is the adapter fine-tuning method. Several other alternative parameter-efficient methods, such as LoRA (Hu et al., 2021), Bitfit (Ben Zaken et al., 2022), and other unifying paradigms (He et al., 2021a), have been proposed in recent times. These methods are modular and can be easily substituted for adapters.

Another major limitation of our work is that we cannot explore whether we can learn different tasks over a given pair of domains. For example, for a given pair of domains such as NEWS and TWITTER, it would be ideal if we learned a domain adapter and reused it for different applications such as sentiment analysis, named entity recognition, among others. We are limited by the availability of data for such scenarios and this would be a potential future work.

References

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia. Association for Computational Linguistics.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 343–351.

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

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2012. A kernel two-sample test. *J. Mach. Learn. Res.*, 13:723–773.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861, Online. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021a. Towards a unified view of parameter-efficient transfer learning. *CoRR*, abs/2110.04366.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jiawei Low, Lidong Bing, and Luo Si. 2021b. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2208–2222, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos. 2021. UDALM: Unsupervised domain adaptation through language modeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2579–2590, Online. Association for Computational Linguistics.

Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, William T. Freeman, and Gregory Wornell. 2018. Co-regularized alignment for unsupervised domain adaptation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 9367–9378, Red Hook, NY, USA. Curran Associates Inc.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Mingsheng Long, Yue Cao, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. 2019. Transferable representation learning with deep adaptation networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(12):3071–3085.

Nghia Ngo Trung, Duy Phung, and Thien Huu Nguyen. 2021. Unsupervised domain adaptation for event detection using domain-specific adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4015–4025, Online. Association for Computational Linguistics.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 751–760, New York, NY, USA. Association for Computing Machinery.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*,

pages 7654–7673, Online. Association for Computational Linguistics.

Abhinav Ramesh Kashyap, Devamanyu Hazarika, Min-Yen Kan, and Roger Zimmermann. 2021a. Domain divergences: A survey and empirical analysis. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1830–1849, Online. Association for Computational Linguistics.

Abhinav Ramesh Kashyap, Laiba Mehnaz, Bhavitvya Malik, Abdul Waheed, Devamanyu Hazarika, Min-Yen Kan, and Rajiv Ratn Shah. 2021b. Analyzing the domain robustness of pretrained language models, layer by layer. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 222–244, Kyiv, Ukraine. Association for Computational Linguistics.

Baochen Sun, Jiashi Feng, and Kate Saenko. 2016. Correlation alignment for unsupervised domain adaptation. *CoRR*, abs/1612.01939.

Baochen Sun and Kate Saenko. 2016. Deep CORAL: correlation alignment for deep domain adaptation. In *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III*, volume 9915 of *Lecture Notes in Computer Science*, pages 443–450.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Linjuan Wu, Shaojuan Wu, Xiaowang Zhang, Deyi Xiong, Shizhan Chen, Zhiqiang Zhuang, and Zhiyong Feng. 2022. Learning disentangled semantic representations for zero-shot cross-lingual transfer in multilingual machine reading comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 991–1000, Dublin, Ireland. Association for Computational Linguistics.

Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. 2017. Central moment discrepancy (CMD) for domain-invariant representation learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Rongsheng Zhang, Yinhe Zheng, Xiao-Xi Mao, and Minlie Huang. 2021. Unsupervised domain adaptation with adapter. *ArXiv*, abs/2111.00667.

Yanzhe Zhang, Xuezhi Wang, and Diyi Yang. 2022. Continual sequence generation with adaptive compositional modules. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3653–3667, Dublin, Ireland. Association for Computational Linguistics.

Src →Trg	**Fully Supervised** 🔥	**Adapter Based**				
		DANN-🔌	DSN-🔌	TASK-🔌	TS-DT-🔌	JOINT-DT-🔌
A → BA	87.68 (1.92)	86.46 (0.26)	87.13 (0.23)	87.03 (0.26)	88.24 (0.76)	**88.74 (0.13)**
A → BO	83.73 (1.61)	78.41 (1.14)	80.23 (0.81)	84.15 (1.10)	84.22 (0.76)	**84.96 (0.28)**
A → C	90.00 (1.17)	87.31 (0.39)	87.58 (0.48)	**89.67 (0.32)**	88.76 (1.32)	89.39 (0.23)
A → MR	76.57 (0.36)	75.54 (0.63)	75.96 (0.27)	76.63 (0.92)	77.39 (0.13)	**77.63 (0.71)**
BA → A	88.56 (1.04)	87.72 (1.85)	87.62 (0.86)	88.33 (1.10)	89.55 (0.10)	**89.70 (0.23)**
BA → BO	85.52 (0.59)	82.89 (3.08)	84.26 (0.85)	84.61 (0.39)	84.38 (0.61)	**85.01 (0.60**
BA → C	89.58 (0.32)	86.63 (0.53)	88.44 (0.90)	**90.63 (0.33)**	87.46 (0.88)	88.64 (0.30)
BA → MR	77.26 (0.71)	74.48 (1.79)	48.67 (15.98)	78.74 (0.35)	**79.42 (0.44)**	78.44 (0.70)
BO → A	87.38 (1.08)	85.90 (0.12)	86.62 (0.41)	85.03 (0.36)	84.79 (0.75)	**87.46 (0.27)**
BO → BA	84.72 (1.15)	82.06 (1.15)	82.75 (1.51)	86.50 (0.39)	**86.84 (0.48)**	86.41 (0.79)
BO → C	87.58 (0.67)	86.94 (0.83)	86.61 (1.03)	88.44 (0.53)	87.86 (0.61)	**88.53 (0.43)**
BO → MR	80.14 (0.52)	76.19 (0.89)	72.08 (7.29)	79.44 (0.95)	**80.52 (0.61)**	78.91 (0.38)
C → A	89.46 (0.49)	87.02 (1.86)	85.50 (1.30)	87.74 (1.18)	88.53 (0.42)	**88.92 (0.44)**
C → BA	90.15 (0.46)	88.10 (1.13)	88.56 (0.25)	81.71 (2.72)	**89.72 (0.43)**	89.32 (0.42)
C → BO	85.08 (0.97)	81.18 (2.07)	83.81 (1.68)	80.55 (0.81)	84.14 (0.52)	**85.42 (0.70)**
C → MR	76.03 (1.15)	64.99 (5.91)	63.59 (11.98)	69.53 (1.24)	73.22 (0.48)	**73.50 (0.84)**
MR → A	79.55 (1.38)	81.05 (1.15)	66.28 (19.68)	82.45 (1.43)	81.93 (0.47)	**84.41 (0.43)**
MR → BA	74.63 (9.8)	77.95 (1.46)	54.64 (17.71)	81.70 (1.22)	84.28 (0.41)	**84.91 (0.36)**
MR → BO	86.09 (1.0)	82.83 (0.62)	49.92 (24.06)	**84.90 (0.23)**	84.47 (0.80)	84.45 (0.31)
MR → C	76.54 (1.78)	84.58 (0.46)	69.47 (12.49)	86.68 (0.65)	86.25 (0.38)	**88.37 (0.11)**
Avg	83.81 (1.41)	81.91 (1.37)	76.49 (5.98)	83.72 (0.88)	84.60 (0.57)	**85.16 (0.43)**

Table 4: F1 scores for AMAZON dataset. We report the mean and standard deviation of 3 runs. The five domains are Apparel (A), Baby (BA), Books (BO), Camera_Photo (C) and Movie Reviews (MR). The difference between this table and Table 2 is we experiment with DSN-🔌 . 🔥 fine-tunes a language model using labeled data from the source domain and tests it on the target domain. This shows that just using the supervised data from the source domain is not enough

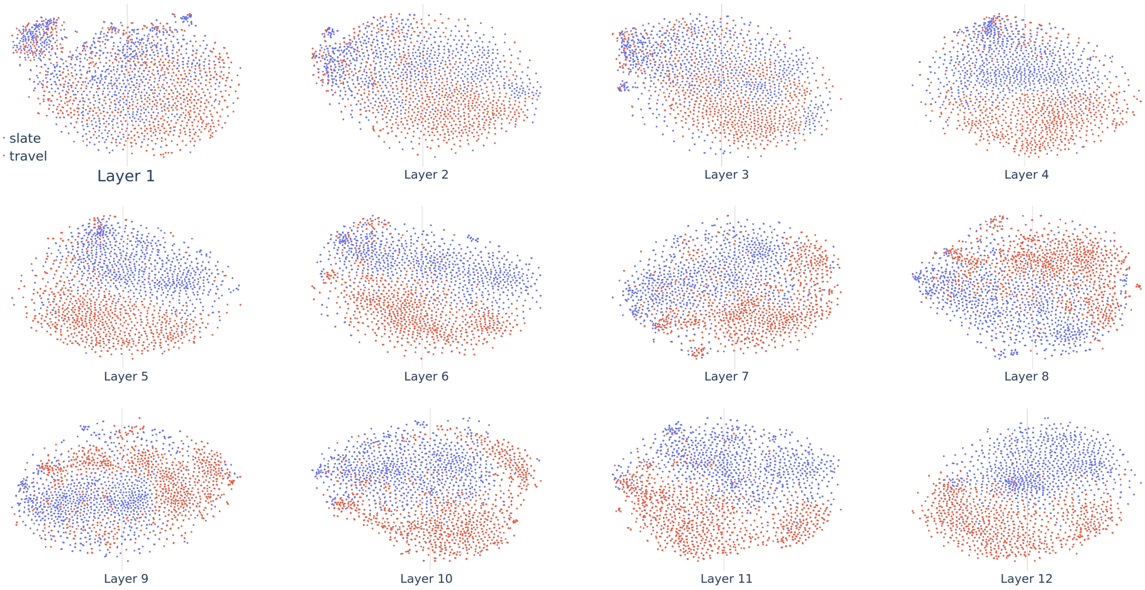


Figure 5: t-SNE plots for the pretrained representations from `bert-base-uncased` for MNLI. Lower layers are domain-invariant whereas higher layers are domain variant.

Src →Trg	**Fully Supervised**	**Adapter Based**				
	🔥	DANN- 🔌	DSN- 🔌	TASK- 🔌	TS-DT- 🔌	JOINT-DT- 🔌
F →S	71.58 (0.31)	70.96 (0.03)	70.16 (0.25)	72.36 (0.36)	**73.46 (0.34)**	72.30 (0.26)
F →G	79.05 (0.94)	78.73 (0.43)	77.01 (0.31)	79.00 (0.46)	78.65 (0.25)	**79.79 (0.22)**
F →TE	74.73 (0.41)	70.89 (0.74)	69.89 (0.04)	70.83 (0.54)	**73.05 (0.70)**	71.59 (0.78)
F→TR	75.84 (0.48)	74.42 (0.18)	73.98 (0.70)	75.85 (0.19)	76.75 (0.80)	**77.07 (0.26)**
S →F	76.27 (0.30)	73.89 (0.61)	73.79 (0.06)	75.25 (0.19)	**75.52 (0.89)**	75.35 (0.56)
S →G	81.00 (0.37)	79.99 (0.36)	79.39 (0.16)	80.76 (0.40)	**81.65 (0.11)**	80.94 (0.30)
S →TE	74.32 (0.71)	72.29 (0.57)	71.69 (0.16)	72.66 (0.79)	**74.09 (0.30)**	73.38 (0.63)
S →TR	77.85 (0.40)	75.58 (0.54)	75.24 (0.42)	76.16 (0.22)	**77.31 (0.60)**	77.16 (0.18)
G →F	73.12 (0.39)	71.57 (0.68)	70.67 (0.29)	72.66 (0.31)	72.66 (0.56)	**73.56 (0.23)**
G →S	72.10 (1.01)	70.17 (0.64)	70.31 (0.44)	71.11 (0.38)	71.14 (0.21)	**71.36 (0.04)**
G →TE	72.80 (0.32)	69.45 (0.96)	69.47 (0.25)	71.40 (0.30)	71.53 (1.04)	**71.99 (0.67)**
G →TR	76.76 (0.08)	74.35 (0.22)	74.00 (0.32)	76.29 (0.10)	76.16 (0.34)	**76.79 (0.59)**
TE →F	73.25 (0.36)	72.24 (0.59)	73.04 (0.28)	**74.48 (0.33)**	73.34 (0.41)	73.89 (0.12)
TE →S	69.52 (1.17)	69.09 (1.79)	69.40 (0.42)	70.94 (0.16)	70.94 (0.55)	**71.41 (0.19)**
TE →G	77.59 (1.38)	77.80 (0.27)	77.56 (0.46)	79.24 (0.35)	79.65 (0.60)	**79.78 (0.64)**
TE →TR	72.45 (2.44)	74.67 (0.50)	74.14 (0.21)	75.27 (0.83)	**76.11 (0.91)**	75.95 (0.50)
TR →F	72.78 (0.37)	70.27 (0.45)	71.10 (0.21)	72.20 (0.49)	73.12 (0.08)	**73.13 (0.22)**
TR →S	70.40 (0.10)	68.35 (0.62)	69.92 (0.50)	70.28 (0.37)	70.67 (0.50)	**71.28 (0.38)**
TR →G	79.75 (0.42)	79.25 (0.34)	79.75 (0.24)	81.26 (0.37)	81.11 (0.42)	**81.55 (0.16)**
TR →TE	72.02 (0.49)	69.33 (0.41)	70.10 (0.52)	70.98 (0.11)	70.95 (0.19)	**71.42 (0.12)**
Avg	74.66 (0.62)	73.16 (0.55)	73.03 (0.32)	74.45 (0.40)	74.89 (0.49)	**74.98 (0.35)**

Table 5: F1 scores for MNLI. We report mean and standard deviation of 3 runs. The five domains are Fiction (F), Slate (S), Government (G), Telephone (TE), and Travel (TR). The difference between this table and Table 3 is we experiment with DSN- 🔌 . 🔥 fine-tunes a language model using labeled data from the source domain and tests it on the target domain. This shows that just using the supervised data from the source domain is not enough.

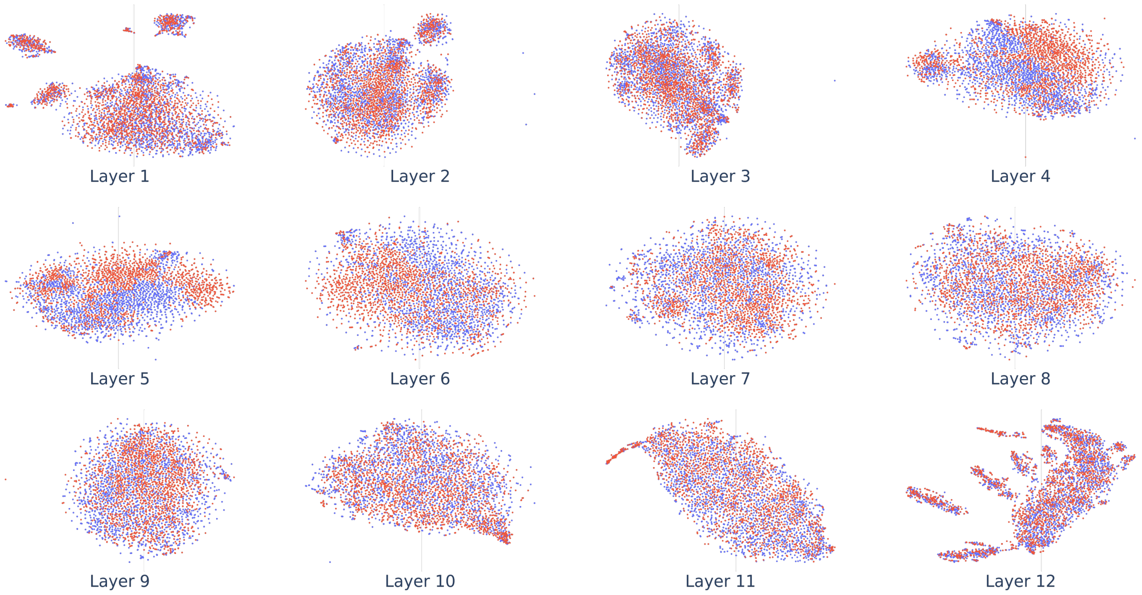

Layer 1 Layer 2 Layer 3 Layer 4
Layer 5 Layer 6 Layer 7 Layer 8
Layer 9 Layer 10 Layer 11 Layer 12

Figure 6: t-SNE plots for the representations from domain adapter trained on S → TR domain for MNLI. We reduce divergence between domains for all layers.

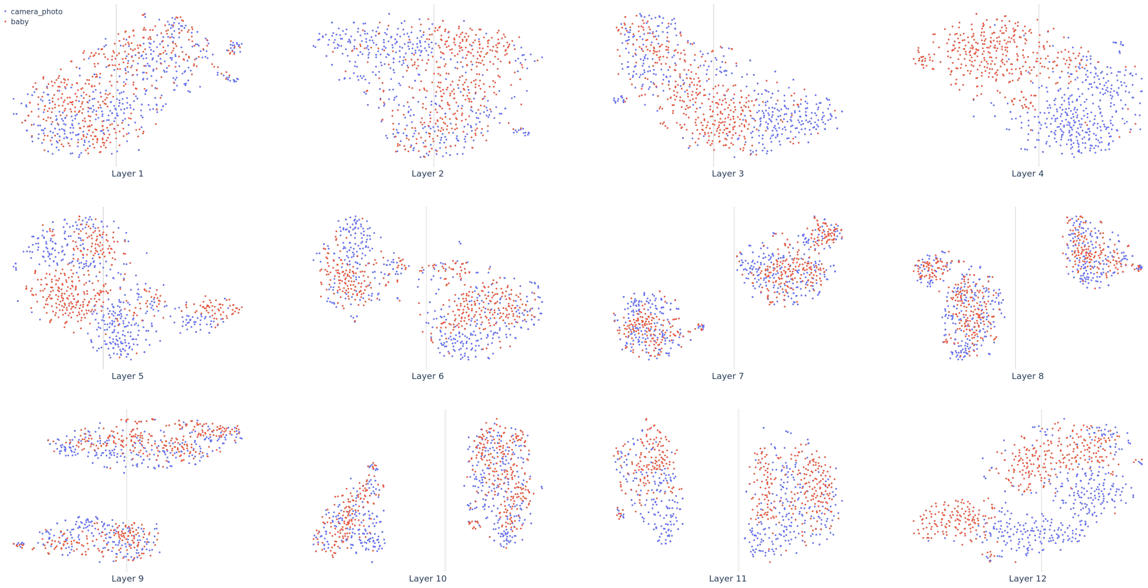
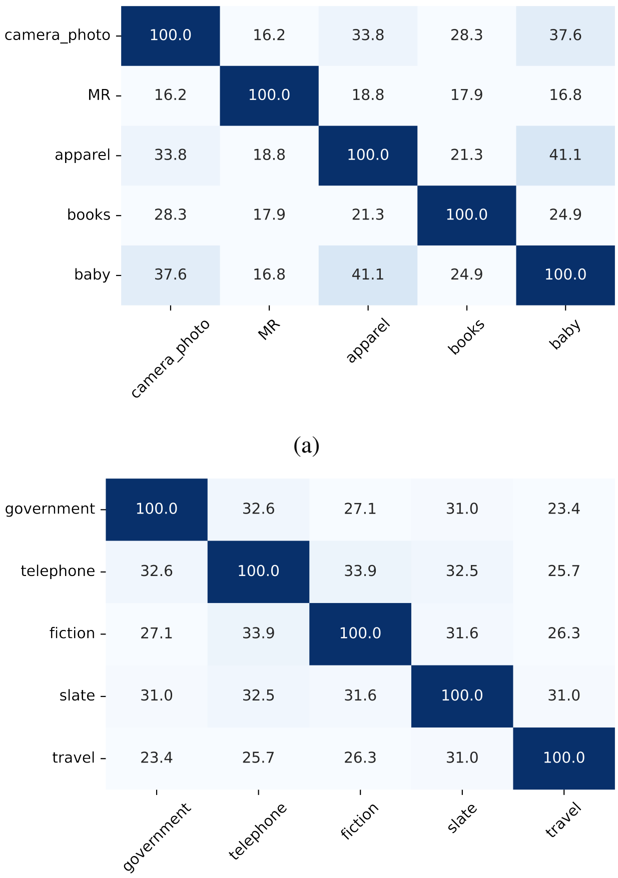


Figure 7: t-SNE plots for the pretrained representations from `bert-base-uncased` for AMAZON. Lower layers are domain-invariant whereas higher layers are domain variant.

Figure 8: t-SNE plots for the representations from domain adapter trained on C→ BO domain for AMAZON. We reduce divergence between domains for all layers.



	camera_photo	MR	apparel	books	baby
camera_photo	100.0	16.2	33.8	28.3	37.6
MR	16.2	100.0	18.8	17.9	16.8
apparel	33.8	18.8	100.0	21.3	41.1
books	28.3	17.9	21.3	100.0	24.9
baby	37.6	16.8	41.1	24.9	100.0

(a)

	government	telephone	fiction	slate	travel
government	100.0	32.6	27.1	31.0	23.4
telephone	32.6	100.0	33.9	32.5	25.7
fiction	27.1	33.9	100.0	31.6	26.3
slate	31.0	32.5	31.6	100.0	31.0
travel	23.4	25.7	26.3	31.0	100.0

(b)

Figure 9: (a) Vocabulary overlap (%) between domains in AMAZON. (b) Vocabulary overlap (%) between domains in MNLI.

Efficient CTC Regularization via Coarse Labels for End-to-End Speech Translation

Biao Zhang[1] **Barry Haddow**[1] **Rico Sennrich**[2,1]

[1]School of Informatics, University of Edinburgh
[2]Department of Computational Linguistics, University of Zurich
`B.Zhang@ed.ac.uk, bhaddow@ed.ac.uk, sennrich@cl.uzh.ch`

Abstract

For end-to-end speech translation, regularizing the encoder with the Connectionist Temporal Classification (CTC) objective using the source transcript or target translation as labels can greatly improve quality metrics. However, CTC demands an extra prediction layer over the vocabulary space, bringing in non-negligible model parameters and computational overheads, although this layer is typically not used for inference. In this paper, we re-examine the need for genuine vocabulary labels for CTC for regularization and explore strategies to reduce the CTC label space, targeting improved efficiency without quality degradation. We propose coarse labeling for CTC (CoLaCTC), which merges vocabulary labels via simple heuristic rules, such as using truncation, division or modulo (MOD) operations. Despite its simplicity, our experiments on 4 source and 8 target languages show that CoLaCTC with MOD particularly can compress the label space aggressively to 256 and even further, gaining training efficiency ($1.18\times \sim 1.77\times$ speedup depending on the original vocabulary size) yet still delivering comparable or better performance than the CTC baseline. We also show that CoLaCTC successfully generalizes to CTC regularization regardless of using transcript or translation for labeling.

1 Introduction

Developing techniques to support the translation from a source-language audio to a target-language text directly, or end-to-end (E2E) speech translation (ST), has attracted increasing attention recently due to its potential of reducing translation latency and avoiding error propagation (Duong et al., 2016; Bérard et al., 2016). However, solving this task is non-trivial because of the speech-text modality gap: one word corresponds to a stochastic sequence of speech signals that vary greatly across speakers and over contexts, which increases the learning difficulty. Recent progress on E2E ST mainly focuses on bridging this gap through the encoder-decoder framework from diverse perspectives (Di Gangi et al., 2019; Salesky et al., 2019; Zhang et al., 2020; Wang et al., 2020b; Han et al., 2021; Zheng et al., 2021).

CTC regularization is such an approach that facilitates the modeling of translation by aligning speech representations from the encoder with discrete labels dynamically via the lens of the Connectionist Temporal Classification (CTC) objective (Graves et al., 2006b). Bahar et al. (2019) first examined the use of the source transcript as discrete labels, improving translation quality consistently across various ST settings; Zhang et al. (2022) further discovered that using the target translation as labels instead can also be surprisingly effective although speech-translation pairs arguably violates CTC's monotonicity prerequisite. Nevertheless, these successes come at the cost of increased computational overheads and model parameters because CTC demands an extra prediction layer over its label space for probability estimation and this space is often huge – traditionally the source or target vocabulary size (Gaido et al., 2020). We thus explore strategies to achieve the best of both worlds, i.e., improving the efficiency of CTC regularization without hurting its performance.

We address this problem by reexamining the need for genuine vocabulary labels for CTC. In contrast to CTC-based generation (Graves et al., 2006b), the prediction layer in CTC regularization of ST is discarded at inference. In other words, sticking to genuine labels is computationally unnecessary. Since the large label space of CTC is a crucial bottleneck hindering training efficiency, we explore ways of reducing it. We propose **Co**arse **La**beling for CTC (CoLaCTC) that manipulates this space by merging vocabulary labels based on simple heuristic rules. Concretely, we map the source or target vocabulary to a pseudo label space subject to some predefined size using simple op-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2264–2276
May 2-6, 2023

Method		Mapping	# Labels	ID Sequence
Genuine Labels		`f(z) = z`	V	`0,1,2,3,4,5,6,7,8`
CoLaCTC	Truncation	`f(z) = min(z, L-1)`	L	`0,1,2,2,2,2,2,2,2`
	Modulo	`f(z) = z % L`	L	`0,1,2,0,1,2,0,1,2`
	Division	`f(z) = ⌊z / (V/L)⌋`	L	`0,0,0,1,1,1,2,2,2`
	Log-Scaling	`f(z) = ⌊log(max(z,1)) * L/log(V)⌋`	L	`0,0,0,1,1,2,2,2,2`

Table 1: Overview of different mappings for CoLaCTC. V and L denote the original vocabulary size and the specified coarse label size, respectively, and $V \gg L$. z is the vocabulary-space token ID starting from 0. The ID sequence is just for a toy example, and we set $V = 9, L = 3$ for illustration. $f(\cdot)$ shows the mapping function. CoLaCTC adopts different operations to reduce the label space from V to L.

erations, such as *truncation*, *modulo*, *division* and *log-scaling* as shown in Table 1.

Despite the label space being transformed, the generated coarse labels still maintain a strong correlation with their vocabulary counterparts, ensuring their informativeness for representation learning. We rigorously examined our method on the MuST-C (Di Gangi et al., 2019) and the Multilingual TEDx (Salesky et al., 2021) benchmarks, covering 4 source languages and 8 target languages. Across diverse settings, CoLaCTC successfully achieves comparable or even better translation performance than the CTC baseline but with significantly improved training efficiency (up to 1.77× speedup depending on the original vocabulary size). Our main contributions are summarized below:[1]

- We propose coarse labeling for CTC regularization which offers a mechanism to decouple the CTC label size from the vocabulary size; with CoLaCTC, a CTC-regularized model can be trained nearly as fast as a non-CTC model.
- We compare two types of CTC regularization for ST, i.e., using transcript or translation for labeling, and show that transcript performs better *when it is available*.
- CoLaCTC delivers promising performance on 4 source and 8 target languages, and also generalizes to both types of CTC regularization.
- Our empirical analysis reveals that CoLaCTC benefits translation similarly to the CTC baseline on different aspects, including homophone translation, and seems to improve the contextualization of speech representations.

2 Related Work

Solving E2E ST requires techniques to mitigate the speech-text modality gap. One way is to develop advanced architectures integrating speech-specific characteristics to the encoder, such as locality modeling for the self-attention (Di Gangi et al., 2019; Gulati et al., 2020) and adaptive speech representation grouping (Salesky et al., 2019; Liu et al., 2020; Zhang et al., 2020). Another way is to leverage knowledge from other languages and/or tasks, including automatic speech recognition (ASR) and machine translation (MT) based multi-task modeling (Anastasopoulos and Chiang, 2018; Dong et al., 2021; Du et al., 2021), cross-lingual transfer learning (Inaguma et al., 2019; Di Gangi et al., 2019; Li et al., 2021), and large-scale weakly, semi- and self-supervised pretraining (Schneider et al., 2019; Ao et al., 2021; Bapna et al., 2022). Our method contributes to E2E ST by accelerating CTC regularization with coarse labels, and is theoretically orthogonal to all the techniques aforementioned. In this study, we mainly focus on bilingual ST using triplet data alone, and leave the exploration of how our method is compatible with other setups to future work.

CTC was first proposed to handle the sequence mismatch problem between acoustic features and transcript tokens, and has been widely applied to ASR (Graves et al., 2006b,a; Chan et al., 2016) and other tasks where the input sequence is longer than the output and their alignment is monotonic (Niu and Mak, 2020; Cai et al., 2022). Recent studies also show promising results when applying CTC to non-monotonic tasks, specifically to non-autoregressive ST and MT (Libovický and Helcl, 2018; Saharia et al., 2020; Gu and Kong, 2021; Chuang et al., 2021). All these methods treat the prediction layer in CTC as a generator, used to predict final outputs. By contrast, Kim et al. (2017) used CTC as an auxiliary objective to improve ASR. In E2E ST, Liu et al. (2020), Gaido et al. (2021), Xu et al. (2021) and Dong et al. (2021) leveraged CTC to compress speech representations to bridge

---

[1] Source code: `https://github.com/bzhangGo/zero`.

the modality gap; Bahar et al. (2019) and Gaido et al. (2022) explored CTC as a regularizer using source transcript as labels, showing encouraging performance although the prediction layer is not used at inference; Zhang et al. (2022) further investigated E2E ST without transcript, and found that CTC regularization with translation as labels also works. Our study follows CTC regularization and extends it with CoLaCTC to address its training inefficiency issue. As far as we know, exploring coarse labels for CTC has never been investigated before, at least on ST.

3 Background: CTC Regularization

CTC regularization improves the encoder-decoder based E2E ST by adding a CTC regularizer to the conventional translation loss (Bahar et al., 2019; Zhang et al., 2022). Formally, given a *(source speech, target translation)* pair denoted as (X, Y) respectively, E2E ST with CTC regularization is optimized via the following interpolated objective:[2]

$$\underbrace{(1-\lambda)\mathcal{L}^{\text{MLE}}(Y|\mathbf{Y})}_{\text{translation loss}} + \underbrace{\lambda\mathcal{L}^{\text{CTC}}(Z|\mathbf{X})}_{\text{regularization}}, \quad (1)$$

where λ is a hyperparameter balancing different sub-objectives. $\mathbf{X} \in \mathbb{R}^{|X|\times d}$ and $\mathbf{Y} \in \mathbb{R}^{|Y|\times d}$ denote the encoder output (or speech representation) and the decoder output, respectively. Z is the label sequence for CTC, which is often either the source transcript or the target translation. $|\cdot|$ indicates the sequence length and d is the model dimension.

The translation loss $\mathcal{L}^{\text{MLE}}$ aims at maximizing the likelihood of observed training instances. Often, we decompose the likelihood token-wise in accordance with the autoregressive generation

$$\mathcal{L}^{\text{MLE}}(Y|\mathbf{Y}) = -\sum_t \log p(y_t|y_{<t}), \quad (2)$$

$$\text{with} \quad p(y_t|y_{<t}) \sim \text{softmax}\left(\mathbf{W}^{\text{MLE}}\mathbf{y}_t\right), \quad (3)$$

where y_t stands for the t-th target token. $\mathbf{y}_t \in \mathbb{R}^d$ is the t-th row of $\mathbf{Y}$, representing the translation prefix $y_{<t} = \{y_1, \ldots, y_{t-1}\}$. $\mathbf{W}^{\text{MLE}} \in \mathbb{R}^{V^{\text{MLE}}\times d}$, also called softmax embedding, is a trainable parameter, and V^{MLE} is the target vocabulary size. E2E ST uses this embedding to estimate the translation probability of each target word as shown in Eq. (3)[3]. At inference, the emitted probability offers direct evidence to search translation candidates.

By contrast, the regularization term $\mathcal{L}^{\text{CTC}}$ encourages the dynamic alignment of speech representations ($\mathbf{X}$) with their corresponding discrete labels (Z) through the CTC algorithm. CTC regularization also maximizes the likelihood. But different from the token-by-token formulation in Eq. (2), CTC estimates the likelihood by marginalizing over all valid mappings between the input and output sequence (Graves et al., 2006b)

$$\mathcal{L}^{\text{CTC}}(Z|\mathbf{X}) = -\log \sum_{A\in\Gamma(Z)} \prod_k p(a_k|\mathbf{x}_k), \quad (4)$$

$$\text{with} \quad p(a_k|\mathbf{x}_k) \sim \text{softmax}\left(\mathbf{W}^{\text{CTC}}\mathbf{x}_k\right), \quad (5)$$

where $\Gamma(Z)$ denotes the set of all valid aligned sequences. The probability of each aligned label a_k in the sequence A is estimated by a prediction layer based on the corresponding speech representation $\mathbf{x}_k$ as shown in Eq. (5). $\mathbf{W}^{\text{CTC}} \in \mathbb{R}^{V^{\text{CTC}}\times d}$ is the prediction parameter, and $V^{\text{CTC}} = V + 1$ denotes the CTC label size. When the source transcript (target translation) is used as labels, V is the source (target) vocabulary size; the extra label is for the special *blank* symbol. We refer readers to Graves et al. (2006b) for more details. Note that this predication layer will be discarded after training when CTC is purely used for regularization.

Previous studies have examined using either source transcript (Bahar et al., 2019; Gaido et al., 2022) or target translation (Zhang et al., 2022) as labels for CTC regularization, but separately. In Section 5.1, we will compare these two types of CTC regularization under the same setup.

4 Coarse Labeling for CTC

Unfortunately, CTC regularization suffers from training inefficiency. Despite CTC being efficiently addressed via dynamic programming, its prediction layer in Eq. (5) is unavoidable. This layer introduces considerable computational overhead due to scaling linearly with the vocabulary size and also brings in large number of model parameters particularly when $V \neq V^{\text{MLE}}$. How to improve the efficiency, and save model parameters while retaining the performance is the focus of our study.

We draw inspiration from the fact that the prediction layer in CTC regularization is not used for inference. Thus, sticking to genuine vocabulary labels is unnecessary technically. If we could design pseudo labels in a reduced label space as alternatives to the genuine ones, that would address the

---

[2] Note we use X, Y to denote the input, and their bold variants to denote the learned hidden representations.

[3] We drop the bias term for clarity.

inefficiency issue. Following this intuition, we propose coarse labeling for CTC (CoLaCTC) which formulates the pseudo label generation process as a vocabulary mapping:

$$f(z) : \mathbb{N}_{[0 \dots V-1]} \mapsto \mathbb{N}_{[0 \dots L-1]}, \quad (6)$$

where z denotes the original vocabulary ID. L is a hyperparameter specifying the label size, and we often set $L \ll V$. This transformation decouples the label size of CTC from the vocabulary size, offering flexibility to optimize the training efficiency. One assumption behind such formulation is that the success of CTC regularization mainly comes from the inductive biases of the CTC algorithm rather than the genuineness of its labels[4], which we verified empirically through experiments.

Eq. (6) merges a set of vocabulary labels into one label according to $f(\cdot)$. Potential mappings are many, such as grouping semantically similar words or considering phonetic similarity. But these linguistically inspired approaches often lack freedom in manipulating the label space (L). Instead, we adopt the following heuristic methods (as shown in Table 1):[5]

Truncation (TRU) Learning speech representations for infrequent items is often difficult, so we merge all vocabulary labels except for the top-$(L-1)$ in TRU

$$f(z) = \min(z, L-1), \quad (7)$$

with the hope that those frequent items could provide informative clues for CTC.

Modulo (MOD) Nevertheless, infrequent items might carry crucial content information. Instead of naively collapsing them, MOD merges diverse labels of varying frequencies based on a fixed interval

$$f(z) = z \pmod{L}. \quad (8)$$

Division (DIV) Different from items with varying frequencies, items of similar frequency often share similar linguistic properties. We explore this in DIV which merges labels of similar ranks uniformly

$$f(z) = \lfloor z * {}^{L}/_{V} \rfloor, \quad (9)$$

where $\lfloor \cdot \rfloor$ denotes the floor function.

Log-Scaling (LOG) One drawback of DIV is that the distribution of its coarse labels becomes badly skewed. To offset this problem, we further study a non-linear, log-scaled transformation, LOG

$$f(z) = \lfloor \log(\max(z, 1)) * {}^{L}/_{\log(V)} \rfloor. \quad (10)$$

Note we intentionally use simple operations to keep the simplicity of CoLaCTC. All the above operations are trivial to implement.

Although labels generated by these operations become linguistically less meaningful, they still keep a strong correlation with their genuine vocabulary counterparts. We expect this correlation could ensure the informativeness of each coarse label and further facilitate the generalization of CoLaCTC to CTC regularization. We compare different operations via experiments.

5 Experiments

Setup We work on two benchmarks, Multilingual TEDx (Salesky et al., 2021) and MuST-C (Di Gangi et al., 2019), covering 4 source and 8 target languages. MuST-C (v1) is an English-audio based multilingual corpus, including translations from English (En) to 8 languages: German (De), Spanish (Es), French (Fr), Italian (It), Dutch (Nl), Portuguese (Pt), Romanian (Ro) and Russian (Ru). The training data for each language pair has ∼452 hours with about 252K utterances on average, and we use the given dev and tst-COMMON splits as the dev and test set, respectively. In contrast, Multilingual TEDx is a multi-source and multi-target ST corpus, containing audios in diverse languages, although its scale is relatively small. We regard this benchmark as a testbed to examine the applicability of our method to audios other than English. We report results on 6 translation directions, i.e., Es-En, Es-Pt, Fr-En, Pt-En, Fr-Es and Fr-Pt. The training data of different language pairs ranges from 25 hours (16K utterances, Fr-Pt) to 69 hours (39K utterances, Es-En), and we use the official dev and test sets for experiments.

---

[4] The inductive biases include the modeling of local structures for speech, word boundary identification and label-guided speech representation learning, etc.

[5] Note that we followed the standard practice and ranked the items in our vocabulary based on their frequency. In this study, the items are (sub)words.

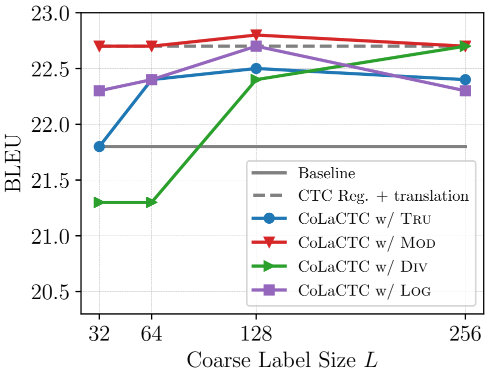


Figure 1: Translation results of different CoLaCTC methods on the MuST-C En-De test set as a function of the coarse label size L. *Reg.*: short for regularization. Here we use **translation** labels for CTC regularization.

We focus on bilingual ST and adopt the E2E ST model following Zhang et al. (2022) which concatenates neighboring frames for downsampling followed by a variant of Transformer for translation. We set $\lambda = 0.3$ for CTC regularization. We evaluate the translation quality using (Sacre)BLEU (Post, 2018).[6] We didn't perform any filtering to the test set. All models are implemented in *Tensorflow*, and trained from scratch without any ASR or MT pretraining. We refer readers to Appendix A for details on data preprocessing and training.

5.1 Analysis on MuST-C En-De

Coarse label size matters, and Mod performs the best. CoLaCTC depends on not only the mapping function selected, but also the coarse label size specified. In general, CoLaCTC with a larger label size produces coarse labels closer to the genuine ones, thus behaving more robustly. We first perform ablations for CoLaCTC with the target translation as labels, where only speech-translation pairs are used at training (Zhang et al., 2022).

We vary L from 32 to 256, and show the results in Figure 1. Different mapping functions have different properties and also show different behaviors, where the label size yields profound impacts. When L is small, Div performs the worst, followed by Log and Tru while Mod performs the best. With the increase of L, the performance difference between different mappings narrows. Label size matters, but the optimal size varies for different mappings. Under different settings, Mod performs the best and is most robust, nearly dominating the others. We next mainly study Mod for CoLaCTC.

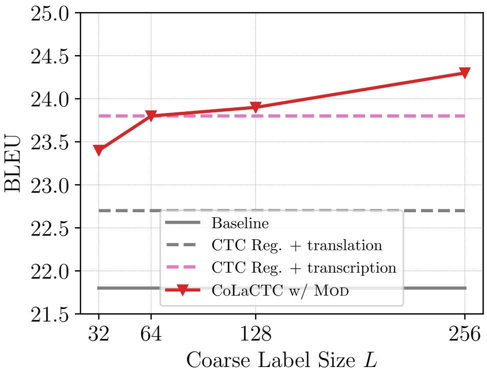


Figure 2: Translation results for CoLaCTC with Mod on the MuST-C En-De test set as a function of L. Here we use **transcription** labels for CTC regularization.

System	#Param	BLEU	Speedup
Baseline	46.1M	21.8	1.39×
CTC Reg. + translation	47.9M	22.7	1.00×
+ CoLaCTC	46.2M	22.7	1.39×
+ share parameters	46.1M	22.4	0.97×
CTC Reg. + transcription	47.5M	23.8	1.00×
+ CoLaCTC	46.2M	24.3	1.31×

Table 2: Test results of different systems on MuST-C En-De. $L = 256$ for CoLaCTC. *#Param*: the number of parameters. *share parameters*: share parameters between $\mathbf{W}^{\text{MLE}}$ and $\mathbf{W}^{\text{CTC}}$. We perform three runs (50 steps each) to evaluate the training speedups on GeForce GTX TITAN X.

CoLaCTC performs comparably to the CTC baseline. Figure 1 also shows that CoLaCTC delivers comparable results to the CTC baseline when proper L is applied; with Mod even across all tested L. Note both methods significantly outperform the vanilla baseline without CTC regularization. This demonstrates that the genuineness of CTC labels matters less for CTC regularization and that our strategy – generating coarse labels in a reduced space – is feasible.

Transcript is more effective than translation as labels for CTC regularization. Despite being effective, using translation as labels for CTC regularization violates the monotonic assumption required by CTC. CTC with transcripts is more established (Bahar et al., 2019; Gaido et al., 2022). We thus compare these two types of CTC regularization and explore how CoLaCTC generalizes.

Figure 2 shows that using transcript instead yields substantial quality improvements (+1.0 BLEU), and that CoLaCTC with Mod generalizes to both settings successfully. Still, the genuineness of CTC labels matters less than their origin does!

---

[6] Signature: *BLEU+c.md+#ref.1+s.exp+tok.13a+v.1.4.14*

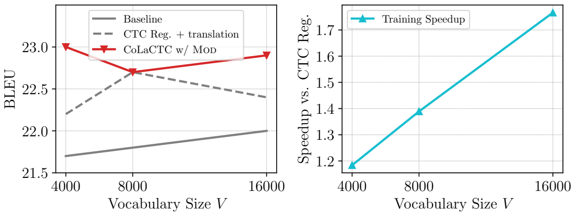


Figure 3: Test results for different methods on MuST-C En-De when varying the target vocabulary size V. We adopt **translation** labels for CTC regularization. $L = 256$ for CoLaCTC.

We observe that using 256 coarse labels works well for CTC regularization under different settings. We set $L = 256$ for the following experiments.

CoLaCTC saves model parameters and greatly improves training efficiency. CTC regularization suffers from inefficiency, which increases model parameters by about 4% and slows the training by 39% as shown in Table 2. We try to solve this problem by sharing parameters between the CTC prediction layer and the softmax output layer when $V = V^{\mathrm{MLE}}$ is the target vocabulary size. Unfortunately, this hurts quality and helps the training efficiency little (+ *share parameters*). By contrast, CoLaCTC nearly recovers the efficiency sacrificed by CTC regularization, running as fast as the vanilla baseline but still retaining quality improvements. Besides, CoLaCTC performs similarly well with different label sequences.

We also note that the degree of inefficiency depends on the computational framework used. We re-tested different methods with PyTorch, where CTC regularization causes a 10% decrease in training speed, much smaller than 39%. However, the conclusion that CTC regularization leads to more trainable parameters and slower running speed, and that CoLaCTC overcomes this issue, still holds.

CoLaCTC performs robustly over different vocabulary sizes; larger V yields higher speedups. Apart from the coarse label size, the target vocabulary size V also affects CTC regularization. Larger vocabulary shortens the target sequence but increases the CTC label space. Figure 3 shows the impact of V on CoLaCTC. Translation performance is highly sensitive to the vocabulary size. Using CTC regularization delivers consistent quality gains against the vanilla baseline, and CoLaCTC shows promising robustness, matching and even outperforming CTC regularization with genuine labels. Regarding training efficiency, the speedup of CoLaCTC should scale linearly with V in theory when L is fixed. Figure 3 confirms this where CoLaCTC achieves higher speedups with larger vocabulary sizes. Particularly, the speedup reaches **1.77×** when V is 16K, a substantial improvement.

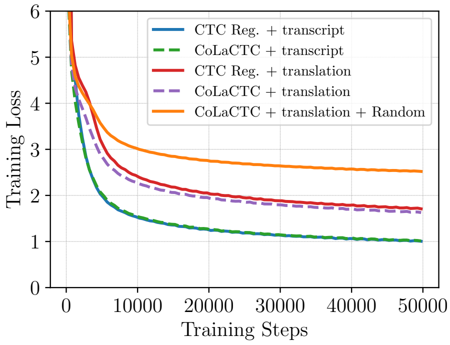


Figure 4: Training loss as a function of training steps for different methods on MuST-C En-De. $L = 256$ for CoLaCTC. *Random* indicates using fully random labels (i.e. `f(z) = randint(0, L)`) with a label size of 256.

CoLaCTC doesn't hurt the trainability of ST models. Would CoLaCTC increase the learning difficulty, which likely reduces performance? Figure 4 shows that 1) CoLaCTC shows similar convergence to the CTC baseline using either transcript or translation for labelling; 2) the model using transcript as labels converges faster and to a better local optima than the counterpart using translation, which also explains the results in Figure 2; 3) random coarse labels result in inferior convergence due to their unpredictable nature.

The vocabulary order of genuine labels has limited impact on CoLaCTC. As shown in Eq. (6), the coarse labeling in CoLaCTC highly relies on the order of original labels in the vocabulary. This ordering encodes word frequency, which might offer crucial clues to CoLaCTC and explain its success. We examine this by randomly shuffling the vocabulary, thus the original vocabulary ID is randomly changed and the ordering information is eliminated. With this shuffled vocabulary, CoLaCTC (Mod) achieves a BLEU score of 23.9 on MuST-C En-De test set, matching the performance of the CTC baseline (23.8) although underperforming the original CoLaCTC (24.3). CoLaCTC benefits from the order information but still achieves promising performance without it.

5.2 Results on Other Languages

CoLaCTC achieves great performance for translation out of English. Table 3 summarizes the results of CoLaCTC on other MuST-C translation

System	De	Es	Fr	It	Nl	Pt	Ro	Ru	Avg
ESPnet-ST (Inaguma et al., 2020)†	22.9	28.0	32.8	23.8	27.4	28.0	21.9	15.8	25.1
Contextual Modeling (Zhang et al., 2021)	22.9	27.3	32.5	23.1	26.0	27.1	23.6	15.8	24.8
Fairseq-ST (Wang et al., 2020a)†	22.7	27.2	32.9	22.7	27.3	28.1	21.9	15.3	24.8
NeurST (Zhao et al., 2021)	22.8	27.4	33.3	22.9	27.2	28.7	22.2	15.1	24.9
Wav2Vec-Transformer (Han et al., 2021)	22.3	28.7	34.3	24.2	28.2	29.3	22.4	15.8	25.7
E2E-ST-JT (Du et al., 2021)†	23.1	27.5	32.8	23.6	27.8	28.7	22.1	14.9	25.1
E2E-ST-TDA (Du et al., 2021)†	24.3	28.3	34.6	24.2	28.7	30.3	23.4	15.9	26.2
Baseline	21.8	27.3	32.3	22.5	26.6	27.5	21.8	14.7	24.3
CTC regularization + target translation	22.7	28.1	33.4	23.2	26.9	28.3	22.6	15.4	25.1
+ CoLaCTC	22.7	27.9	33.3	23.7	27.1	28.0	22.4	15.9	25.1
CTC regularization + source transcription	23.8	28.6	33.9	24.3	28.3	29.3	23.3	16.3	26.0
+ CoLaCTC	24.3	28.4	34.5	24.6	28.1	28.8	23.3	16.6	**26.1**

Table 3: BLEU of different systems on MuST-C `tst-COMMON`. *Avg*: average score over different language pairs (translation is always out-of English). †: systems that might perform filtering to the test set, meaning results are not necessarily comparable. *Baseline*: the model without CTC regularization; $L = 256$ and Mod for CoLaCTC.

System	Es-En	Es-Pt	Fr-En	Pt-En	Fr-Es	Fr-Pt	Avg
Bilingual Cascades (Salesky et al., 2021)	15.5	23.3	17.2	16.1	17.8	12.2	17.0
Bilingual E2E ST (Salesky et al., 2021)	7.0	12.2	8.9	8.1	10.6	7.9	9.1
Multilingual E2E ST (Salesky et al., 2021)	12.3	17.4	12.0	12.0	13.6	13.2	13.4
Baseline	11.6	13.3	7.6	8.5	6.1	1.9	8.2
CTC regularization + target translation	13.0	18.2	12.2	11.4	11.5	6.1	12.1
+ CoLaCTC	13.3	19.0	12.1	12.0	11.2	5.0	12.1
CTC regularization + source transcription	18.0	23.0	19.3	17.8	19.8	13.8	**18.6**
+ CoLaCTC	17.8	23.1	19.9	17.5	19.6	13.3	18.5

Table 4: BLEU Scores on Multilingual TEDx test sets. $L = 256$ and Mod for CoLaCTC.

directions. The performance of CoLaCTC varies across different languages with both positive and negative gains. But overall, CoLaCTC is on par with its CTC baselines and largely outperforms the vanilla baseline without CTC regularization. On average, CoLaCTC delivers a BLEU score of 25.1 and 26.1 when used with translation and transcription labels, respectively, which also surpasses many strong previous studies (Inaguma et al., 2020; Zhao et al., 2021; Zhang et al., 2021).

Joint training with CTC regularization is preferable to the traditional pretraining-finetuning paradigm for E2E ST. The current *de facto* standard for training an E2E ST model is to firstly initialize it with a pretrained ASR encoder and/or MT decoder and then finetune it on ST data. Despite its effectiveness, this pipeline paradigm often consumes longer training time and inevitably complicates the optimization procedure. In contrast, joint training with CTC regularization is technically simpler and delivers comparable and even better results as shown in Table 3, echoing with Gaido et al. (2022). Note that we also re-implemented the pipeline baseline using our in-house codebase, which achieves 22.9 BLEU on MuST-C En-De, far below the joint training with transcript (23.8).

Since CoLaCTC solves the inefficiency issue for CTC regularization, we would recommend using the joint training as the new standard for E2E ST, especially when only triplet training data is used.

CoLaCTC generalizes to ST settings other than English audios. The above results for CoLaCTC are multilingual, but all use English audio from TED on the source side. To demonstrate generalization across different source languages, we conduct experiments on Multilingual TEDx and work on ST for Es, Fr and Pt. Table 4 shows that CoLaCTC generalizes well to other source languages. In addition, CTC regularization performs much better on this benchmark, substantially outperforming the vanilla baseline by 10.4 BLEU, matching the performance of multilingual ST reported by Salesky et al. (2021). We ascribe this success to the small scale of Multilingual TEDx where regularization techniques, like CTC Reg., often work better.

6 Discussion

The promising performance of CoLaCTC inspires us to further explore why coarse labels could work for CTC regularization. Analyzing the underlying

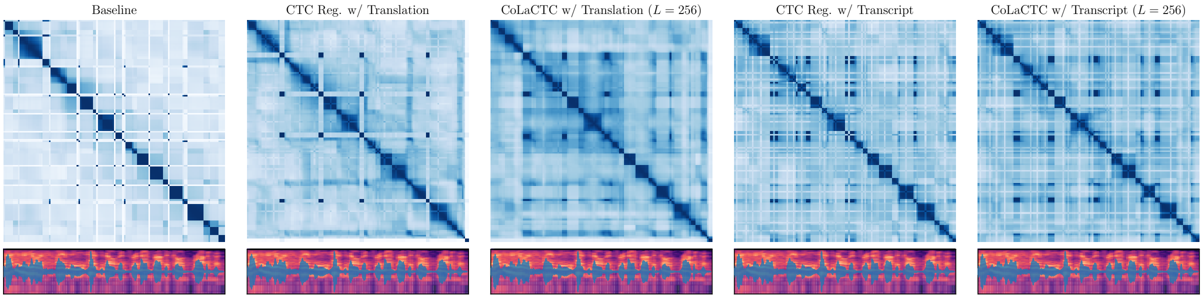


Figure 5: Visualization of the cosine similarity of the final-layer speech encoding for different methods. *Top*: cosine similarity where darker color shows higher similarity; *Bottom*: speech spectrogram. The example is for the first test case in MuST-C En-De.

System	Similarity
Baseline	0.13
CTC Reg. + translation	0.31
+ CoLaCTC	0.39
CTC Reg. + transcript	0.35
+ CoLaCTC	0.38

Table 5: Cosine similarity of speech representations on the MuST-C En-De test set. We report average results.

System	MT Baseline	CL w/			
		Div	Tru	Log	Mod
BLEU	30.5	10.5	15.1	22.9	25.6

Table 6: BLEU scores for *text-to-text* translation conditioned on coarse labels on the MuST-C En-De test set. *MT Baseline*: En→De translation with the vanilla English input; *CL w/ **: using coarse labels instead as the source input. $L = 256$. We use the standard Transformer base setting for experiments.

mechanism theoretically is non-trivial. Instead, we understand this question through empirical probes, such as inspecting the change of speech representations and examining how CoLaCTC behaves on different translation aspects.

CTC regularization improves the contextualization of speech representations, so does CoLaCTC. The CTC objective is directly stacked onto the encoder, then what happens to the speech representation (the final encoder output)? Figure 5 illustrates an example, where speech representations after applying CTC regularization (and CoLaCTC) become closer to each other as measured by the cosine similarity. This is further supported by the results in Table 5. Still, the local structure of audio, i.e. the diagonal similarity, is kept. Intuitively, the increased cosine similarity is a reflection of contextualization, and CTC regularization (and CoLaCTC) encourages the encoder to consider (distant) contextual clues.

System	Noun	Verb	Adj.	Adv.	H.Phone
Baseline	43.0	38.6	42.8	46.6	49.4
CTC Reg.					
+ translation	44.4	38.9	44.1	46.8	49.2
+ CoLaCTC	44.3	39.1	43.7	47.5	49.2
CTC Reg.					
+ transcript	45.8	40.6	**46.0**	**48.3**	**51.2**
+ CoLaCTC	**46.1**	**41.1**	45.6	48.0	51.1

Table 7: Translation accuracy of different types of source words on the MuST-C En-De test set. *Adj.*, *Adv.* and *H.Phone* are short for adjective, adverb and homophone.

Coarse labels especially produced by Mod preserve source semantics and are informative for translation. The mappings considered in this study are solely based on heuristic rules. Despite improvements on ST, whether the generated coarse labels themselves are informative is still questionable. We address this concern by performing experiments on text-to-text machine translation and use the coarse label sequence as the source input. Table 6 shows that the coarse labels encode source semantics, achieving non-trivial translation performance. In particular, using Mod achieves a test BLEU score of 25.6. This result still lags far behind the vanilla MT baseline (30.5), but it demonstrates the informativeness of coarse labels, and also partially explains the success of CoLaCTC.

The performance of CoLaCTC is robust on different types of source words. The translation rule for different types of source words often varies greatly. Next, we examine how CoLaCTC generalizes to different words, including nouns, verbs, adjectives, adverbs, and homophones (specific to speech processing). We annotate the part-of-speech tag for each source sentence via Stanford POS tagger (Toutanova et al., 2003), and adopt the homophone list used for contextual evaluation (Zhang

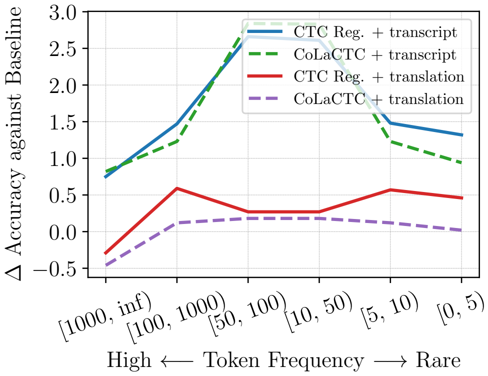


Figure 6: Relative gains of translation accuracy against the Baseline on MuST-C En-De as a function of token frequency (x-axis). We perform APT for each token group separately.

System	En-It		En-Fr	
	Cov.	Acc.	Cov.	Acc.
Bentivogli et al. (2020)	-	43.3	-	46.0
Baseline	53.3	65.8	59.3	64.6
CTC Reg. + translation	56.0	**67.2**	59.9	66.5
+ CoLaCTC	55.9	67.0	59.6	65.4
CTC Reg. + transcript	55.8	66.9	61.6	66.1
+ CoLaCTC	**57.4**	66.9	**62.0**	**66.7**

Table 8: Coverage (*Cov.*) and accuracy (*Acc.*) scores for gender translation on MuST-SHE En-It and En-Fr.

et al., 2021)[7]. We employ translation accuracy as the metric, approximated by the APT framework (Miculicich Werlen and Popescu-Belis, 2017) where *fast_align* is used to get the word alignment (Dyer et al., 2013). Table 7 shows the results.

CTC regularization largely improves the translation of nouns and adjectives; using transcript as labels further benefits the translation for verbs, adverbs and homophones. Regardless of source word and CTC label types, CoLaCTC shows comparable (sometimes even better) performance to the CTC baseline, showing its strong generalization. Note that observations on other languages, e.g. En-Fr and En-It, are similar (see Appendix B). Further analysis shows that the gains by CTC regularization and CoLaCTC mainly come from benefiting rare-word translation, as shown in Figure 6.

CoLaCTC benefits gender translation similarly to the CTC baseline. Languages often differ in gender expression, leading to translation difficulty. We further evaluate how CoLaCTC handles the gender ambiguity using the MuST-SHE benchmark (Bentivogli et al., 2020). Table 8 shows that CoLaCTC achieves comparable performance to the CTC baseline, suggesting that using coarse labels for CTC regularization doesn't hurt its gender disambiguation ability. Besides, we observe that translation and transcript labels show similar positive effects on gender translation.

7 Conclusion and Future Work

In this paper, we have presented coarse labeling for CTC to address the training inefficiency issue of CTC regularization. The key idea behind CoLaCTC is to transform CTC labels from the vocabulary space to a specified and reduced coarse space. We adopt trivial mappings for this transformation, such as using the modulo operation. Despite its simplicity, CoLaCTC successfully achieves the best of both worlds – improving the training efficiency for CTC regularization (up to 1.77× speedup) and retaining its quality benefits – and generalizes to different types of CTC regularization. Note the training speedup scales as the vocabulary size increases. Our analysis further shows that the genuineness of CTC labels matters less than their origin.

In the future, we are interested in examining the complementarity of CoLaCTC with other advanced ST modeling. We will study how our method performs in a multilingual and simultaneous setup as well as ST settings with extra ASR and/or MT data.

Limitations

While the proposed method achieves encouraging performance across diverse languages and translation setups, our understanding of why it performs so well is still limited, particularly considering the simplicity of the adopted mapping function (MOD). Uncovering the underlying reason behind such success might offer valuable insights to further the speech modeling, having a potential broader impact on the speech processing community.

Acknowledgements

We thanks the reviewers for their insightful comments. This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement 825460 (ELITR), and from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant number 10039436 – UTTER]. RS acknowledges funding from the Swiss National Science Foundation (project MUTAMUR; no. 176727).

---

[7] The list is publicly available at `bit.ly/3mGITEe`.

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91, New Orleans, Louisiana. Association for Computational Linguistics.

Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren, Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, Zhihua Wei, Yao Qian, Jinyu Li, and Furu Wei. 2021. Speecht5: Unified-modal encoder-decoder pre-training for spoken language processing.

Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019. A comparative study on end-to-end speech to text translation. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 792–799.

Ankur Bapna, Colin Cherry, Yu Zhang, Ye Jia, Melvin Johnson, Yong Cheng, Simran Khanuja, Jason Riesa, and Alexis Conneau. 2022. mslam: Massively multilingual joint pre-training for speech and text.

Luisa Bentivogli, Beatrice Savoldi, Matteo Negri, Mattia A. Di Gangi, Roldano Cattoni, and Marco Turchi. 2020. Gender in danger? evaluating speech translation technology on the MuST-SHE corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6923–6933, Online. Association for Computational Linguistics.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. In *NIPS Workshop on End-to-end Learning for Speech and Audio Processing*, Barcelona, Spain.

Xingyu Cai, Jiahong Yuan, Yuchen Bian, Guangxu Xun, Jiaji Huang, and Kenneth Church. 2022. W-CTC: a connectionist temporal classification loss with wild cards. In *International Conference on Learning Representations*.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964.

Shun-Po Chuang, Yung-Sung Chuang, Chih-Chiang Chang, and Hung-yi Lee. 2021. Investigating the reordering capability in CTC-based non-autoregressive end-to-end speech translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1068–1077, Online. Association for Computational Linguistics.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting Transformer to End-to-End Spoken Language Translation. In *Proc. Interspeech 2019*, pages 1133–1137.

Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2019. One-to-many multilingual end-to-end speech translation. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 585–592.

Qianqian Dong, Rong Ye, Mingxuan Wang, Hao Zhou, Shuang Xu, Bo Xu, and Lei Li. 2021. Listen, understand and translate: Triple supervision decouples end-to-end speech-to-text translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12749–12759.

Yichao Du, Zhirui Zhang, Weizhi Wang, Boxing Chen, Jun Xie, and Tong Xu. 2021. Regularizing end-to-end speech translation with triangular decomposition agreement. *arXiv preprint arXiv:2112.10991*.

Long Duong, Antonios Anastasopoulos, David Chiang, Steven Bird, and Trevor Cohn. 2016. An attentional model for speech translation without transcription. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 949–959, San Diego, California. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Marco Gaido, Mauro Cettolo, Matteo Negri, and Marco Turchi. 2021. CTC-based compression for direct speech translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 690–696, Online. Association for Computational Linguistics.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2020. End-to-end speech-translation with knowledge distillation: FBK@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 80–88, Online. Association for Computational Linguistics.

Marco Gaido, Sara Papi, Dennis Fucci, Giuseppe Fiameni, Matteo Negri, and Marco Turchi. 2022.

Efficient yet competitive speech translation: FBK@IWSLT2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 177–189, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006a. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA. Association for Computing Machinery.

Alex Graves, Santiago Fernández, and Faustino Gomez. 2006b. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *In Proceedings of the International Conference on Machine Learning, ICML 2006*, pages 369–376.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133, Online. Association for Computational Linguistics.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040.

Chi Han, Mingxuan Wang, Heng Ji, and Lei Li. 2021. Learning shared semantic space for speech-to-text translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2214–2225, Online. Association for Computational Linguistics.

Hirofumi Inaguma, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2019. Multilingual end-to-end speech translation. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 570–577. IEEE.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint ctc-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4835–4839.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2021. Multilingual speech translation from efficient finetuning of pretrained models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 827–838, Online. Association for Computational Linguistics.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Yuchen Liu, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. 2020. Bridging the modality gap for speech-to-text translation. *ArXiv*, abs/2010.14920.

Lesly Miculicich Werlen and Andrei Popescu-Belis. 2017. Validation of an automatic metric for the accuracy of pronoun translation (APT). In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 17–25, Copenhagen, Denmark. Association for Computational Linguistics.

Zhe Niu and Brian Mak. 2020. Stochastic fine-grained labeling of multi-state sign glosses for continuous sign language recognition. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI*, page 172–186, Berlin, Heidelberg. Springer-Verlag.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. Association for Computational Linguistics.

Elizabeth Salesky, Matthias Sperber, and Alan W Black. 2019. Exploring phoneme-level speech representations for end-to-end speech translation. In *Proceedings of the 57th Annual Meeting of the Association for*

Computational Linguistics, pages 1835–1841, Florence, Italy. Association for Computational Linguistics.

Elizabeth Salesky, Matthew Wiesner, Jacob Bremerman, Roldano Cattoni, Matteo Negri, Marco Turchi, Douglas W. Oard, and Matt Post. 2021. The Multilingual TEDx Corpus for Speech Recognition and Translation. In *Proc. Interspeech 2021*, pages 3655–3659.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition. In *Proc. Interspeech 2019*, pages 3465–3469.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020a. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 33–39, Suzhou, China. Association for Computational Linguistics.

Chengyi Wang, Yu Wu, Shujie Liu, Ming Zhou, and Zhenglu Yang. 2020b. Curriculum pre-training for end-to-end speech translation. *arXiv preprint arXiv:2004.10093*.

Chen Xu, Bojie Hu, Yanyang Li, Yuhao Zhang, Shen Huang, Qi Ju, Tong Xiao, and Jingbo Zhu. 2021. Stacked acoustic-and-textual encoding: Integrating the pre-trained models into speech translation encoders. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2619–2630, Online. Association for Computational Linguistics.

Biao Zhang, Barry Haddow, and Rico Sennrich. 2022. Revisiting end-to-end speech-to-text translation from scratch. In *Proceedings of ICML*.

Biao Zhang, Ivan Titov, Barry Haddow, and Rico Sennrich. 2020. Adaptive feature selection for end-to-end speech translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2533–2544, Online. Association for Computational Linguistics.

Biao Zhang, Ivan Titov, Barry Haddow, and Rico Sennrich. 2021. Beyond sentence-level end-to-end speech translation: Context helps. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2566–2578, Online. Association for Computational Linguistics.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019. Improving deep transformer with depth-scaled initialization and merged attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 898–909, Hong Kong, China. Association for Computational Linguistics.

Chengqi Zhao, Mingxuan Wang, Qianqian Dong, Rong Ye, and Lei Li. 2021. NeurST: Neural speech translation toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 55–62, Online. Association for Computational Linguistics.

Renjie Zheng, Junkun Chen, Mingbo Ma, and Liang Huang. 2021. Fused acoustic and text encoding for multimodal bilingual pretraining and speech translation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12736–12746. PMLR.

A Experimental Setting

We preprocess texts using Moses scripts (Koehn et al., 2007) and adopt the byte pair encoding algorithm (Sennrich et al., 2016, BPE) to handle rare tokens. In particular, we encode datasets in MuST-C and Multilingual TEDx with a BPE vocabulary size of 8K and 4K, respectively. As for audios, we adopt a sampling rate of 16KHz and filter out segments longer than 3000 frames. We extract 40-dimensional log mel-scale filterbank features for acoustic modeling with a step size of 10ms and window size of 25ms, and further augment them with

	System	Noun	Verb	Adj.	Adv.	H.Phone
En-It	Baseline	48.0	37.5	45.4	44.0	40.9
	CTC Reg. + translation	49.6	38.6	47.2	45.0	41.7
	+ CoLaCTC	49.6	38.7	48.3	45.4	42.0
	CTC Reg. + transcript	50.5	**39.3**	48.0	45.2	42.3
	+ CoLaCTC	**51.2**	39.0	**49.1**	**45.7**	**43.0**
En-Fr	Baseline	57.2	51.0	53.4	53.9	58.1
	CTC Reg. + translation	58.5	52.8	55.4	54.9	58.9
	+ CoLaCTC	58.5	52.3	55.1	54.9	58.2
	CTC Reg. + transcript	59.3	53.0	54.9	55.1	59.7
	+ CoLaCTC	**60.0**	**53.9**	**57.0**	**55.8**	**60.3**

Table 9: Translation accuracy of different types of source words on the MuST-C En-Fr/En-It test set. *Adj.*, *Adv.* and *H.Phone* are short for adjective, adverb and homophone.

their delta and delta-delta features. The final acoustic feature vector is 120-dimensional regularized by mean subtraction and variance normalization.

We focus on bilingual ST and adopt the E2E ST architecture following Zhang et al. (2022): we use Transformer with the post layer normalization structure plus the sinusoidal positional encoding (Vaswani et al., 2017); we set the encoder and decoder depth to 12 and 6, respectively, and adopt the depth-scaled initialization method to stabilize the training (Zhang et al., 2019); we set the model dimension to $d = 256$, the feed-forward layer size to 4096 and the number of attention head to 4; we employ the parameterized distance penalty with $R = 512$ and set $\lambda = 0.3$ for CTC regularization (Zhang et al., 2022).

We train all models via Adam (Kingma and Ba, 2015, $\beta_1 = 0.9, \beta_2 = 0.98$) with a warmup step of 4K and label smoothing rate of 0.1. Samples with around 20K target subwords are scheduled into one batch for training, and we set the maximum training step for MuST-C and Multilingual TEDx to 50K and 20K, respectively. We apply dropout to residual connections and ReLU activations with a rate of 0.2. We perform checkpoint evaluation every 1K training steps on the dev set, and average the best 10 checkpoints for final testing. We use beam search for decoding, and set the beam size to 8. We tune the length penalty for each language pair on its dev set separately.

B Additional Results and Analysis

The performance of CoLaCTC on different types of source words generalizes to other languages. Table 9 shows the translation accuracy of different models on En-Fr and En-It. The observation is similar to Table 7, where transcript labels are more effective than translation labels for CTC regularization, and that CoLaCTC performs comparable to genuine labels. One exception is that CTC regularization also greatly benefits the translation of verbs and adverbs on En-Fr and En-It. These results suggest that our observation is not language-specific or caused by some random effect, but rather CTC regularization and CoLaCTC generalizes to different language pairs.

Exploring Category Structure with Contextual Language Models and Lexical Semantic Networks

Joseph Renner[1] **Pascal Denis**[1] **Rémi Gilleron**[1] **Angèle Brunellière**[2]

[1]Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 - CRIStAL, F-59000, Lille, France
`{firstname.lastname}@inria.fr`
[2]Univ. Lille, CNRS, UMR 9193 - SCALab, F-59000 Lille, France
`angele.brunelliere@univ-lille.fr`

Abstract

Recent work on predicting category structure with distributional models, using either static word embeddings (Heyman and Heyman, 2019) or contextualized language models (CLMs) (Misra et al., 2021), report low correlations with human ratings, thus calling into question their plausibility as models of human semantic memory. In this work, we revisit this question testing a wider array of methods for probing CLMs for predicting typicality scores. Our experiments, using BERT (Devlin et al., 2018), show the importance of using the right type of CLM probes, as our best BERT-based typicality prediction methods substantially improve over previous works. Second, our results highlight the importance of polysemy in this task: our best results are obtained when using a disambiguation mechanism. Finally, additional experiments reveal that Information Content-based WordNet (Miller, 1995), also endowed with disambiguation, matches the performance of the best BERT-based method, and in fact captures complementary information, which can be combined with BERT to achieve enhanced typicality predictions.

1 Introduction

The empirical success of contextual language models (CLMs) (Peters et al., 2018; Devlin et al., 2018; Liu et al., 2019) has led to much research analyzing their functionality (Rogers et al., 2020; Wu et al., 2020; Ethayarajh, 2019) and how they acquire semantic and world knowledge (Petroni et al., 2019). It also raises the question of their plausibility as models of human semantic memory (Chronis and Erk, 2020; Ettinger, 2020; Garí Soler and Apidianaki, 2021). The study of categorical knowledge, in particular *typicality*, provides a window into this question (Murphy, 2004). As initially observed by Rosch (1975), native speakers of English consider that certain exemplars (e.g., `robin`, `crow`) are more representative than others (e.g., `penguin`, `ostrich`) of a conceptual category (`birds`). That is, categorical knowledge is organized along a graded structure. According to prototype theory (Rosch, 1975), the structure follows from the fact that properties frequently shared among category members tend to be integrated into its prototype.

The main question we raise in this paper is whether distributional models, and CLMs in particular, are indeed aware of category structure, as captured in existing human typicality norms for English. The broader question for cognitive science is whether or not this type of knowledge can be learned through text-based exposure alone, thus contributing to the larger debate on the role of language in learning semantic knowledge (Lupyan and Lewis, 2019). But there are also more practical motivations for this work, as many NLP tasks (such as information retrieval, question answering, natural language inference) can arguably benefit from typicality relations by understanding which exemplars are more relevant to particular concepts.

Previous work on this topic provide mostly negative results. Heyman and Heyman (2019) use classical static embeddings and represent typicality scores between categories and their exemplars as the cosine distance between their corresponding word embeddings. They conclude that static embeddings poorly account for human typicality scores, as obtained from Morrow and Duffy (2005). More recently, Misra et al. (2021) use various CLMs to predict typicality. While CLMs have a larger, more expressive parameter space and are tuned over larger corpora with arguably better learning objectives than static word embeddings, Misra et al. (2021) report only slightly better, and still modest, correlations with human typicality judgements (in this case, Rosch (1975)). Their probing approach uses a Cloze test formulation over taxonomic sentences (e.g., *A robin is a bird*).

We revisit this question by first introducing an expanded suite of methods to extract typicality scores

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2277–2290
May 2-6, 2023 ©2023 Association for Computational Linguistics

from CLMs, some of which improve correlations with human norms, showing the importance of using the right probe for typicality. How to reliably and efficiently probe CLMs is still an open question, including supervised and unsupervised approaches (Rogers et al., 2020; Elazar et al., 2021; Wu et al., 2020). The problem is exacerbated by the fact that typicality is a relation between pairs of concepts, abstracted away from contexts, while CLMs produce representations for contextualized word tokens (not types). We consider a wider array of approaches, all unsupervised, for predicting typicality from CLMs, using BERT as representative test case. Our probing approaches fall into two main classes: (i) BERT as a probabilistic language model, and (ii) generating "static" word embeddings from contextual BERT embeddings.

Our second contribution is showing the importance of polysemy (e.g., *orange* can be a fruit, a color, or a company) for typicality, an issue that has been largely overlooked in previous works. When judging category-exemplar pair typicality, humans arguably don’t consider all the senses of the category and exemplar words. Instead they consider the senses of the category and exemplar that are most compatible with one another, in effect performing a joint disambiguation. Polysemy is problematic for static word embeddings as they collapse word senses into a single vector. CLMs, on the other hand, provide some form of sense selection through contextualization (Ethayarajh, 2019; Garí Soler and Apidianaki, 2021), but there are many possible ways to use CLMs and to provide contexts to these models (such as sampled or taxonomic sentences). We find that using a simple disambiguation mechanism, such as the method of deriving multi-prototype embeddings introduced in Chronis and Erk (2020), leads to typicality predictions more closely correlated with human rankings.

Finally, another research question we address in this paper is whether categorical structure is present in lexical semantic networks, such as WordNet (Miller, 1995). The hierarchical concept organization in WordNet was indeed initially inspired by theories of human semantic memory (Beckwith et al., 2021). Note that polysemy is also an issue in using WordNet, as we need to map category and exemplars to specific synsets. However, using the structure of WordNet, a simple disambiguation heuristic, and taking into account word frequency information leads to competitive results. Furthermore, we find that WordNet-based and BERT-based typicality predictions contain complementary information, and creating a simple ensemble method of the two further increases performance.

In summary, our main contributions are:

- We introduce and compare a large array of unsupervised probing approaches for assessing whether BERT capture the internal structure of semantic categories.
- Our experiments reveal BERT can predict human typicality rankings more reliably than previously found in Heyman and Heyman (2019) and Misra et al. (2021), but only when endowed with a disambiguation mechanism.
- Similarly, typicality predictions based on the structure of WordNet (again with a disambiguation mechanism) can achieve similar but complementary performance.
- A simple combination of BERT and WordNet predictions leads to a higher level of correlation (Spearman greater than 0.5) with human typicality rankings.

2 Related Work

Recent years have seen an ever growing body of work on assessing whether distributional models constitute realistic models of human semantic memory, within both NLP (Chronis and Erk, 2020; Ettinger, 2020) and Cognitive Science (Hollis, 2017; Hollis and Westbury, 2016; Mandera et al., 2017; Günther et al., 2019; Lupyan and Lewis, 2019; Kumar, 2021). In this context, the study of categorical knowledge, and specifically graded typicality, plays a crucial role, given it is one of the most reliable findings in the study of human categorical knowledge (Rosch, 1975; Murphy, 2004). The first work on using distributional models to predict typicality scores is Heyman and Heyman (2019), who estimate these scores, in English and Dutch, using cosine distances between the static embeddings of the exemplar and category words. These show that static embeddings, whether trained in a counted- or prediction-based fashion, yield poor correlations with human typicality norms. On the related task of lexical entailment (LE), Vulić et al. (2017) evaluated different unsupervised and supervised methods that use static word embeddings. They also found a significant gap between static word embeddings and human norms. While LE is closely related to typicality, it differs in that LE predicts the relationship between two words, while typicality compares the relationship between two concepts (in which words are used as a proxy, in our case). Also, LE ratings are not restricted to pairs of words belonging to the same category. We differ from these earlier work on typicality by considering more expressive CLMs instead of static embeddings. The most closely related work is Misra et al. (2021) who probe a wide range of CLMs (i.e., different variants of BERT and GPT) for predicting typicality scores: these are extracted using a Cloze task formulation over hand crafted taxonomic sentences (e.g. *A robin is a bird.*). They report better correlation scores with human norms, although still modest, which they take as indication that text exposure is not sufficient to learn category structure. Our focus in this work is different and somewhat broader in that we consider a wider array of probing methods of CLMs, though restricted to unsupervised ones. Supervised probing approaches, based on classifiers taking CLM representations as inputs, are problematic as they provide a less direct probing approach, possibly adding additional confounds (Elazar et al., 2021; Wu et al., 2020). Another distinctive aspect of our work is that we compare CLMs to lexical networks like WordNet (Miller, 1995). Furthermore, we study the impact of polysemy in typicality, which was not controlled in Misra et al. (2021) or in Heyman and Heyman (2019). The work of Apidianaki and Garí Soler (2021) shares some similarities with (Misra et al., 2021): they also use BERT under a Cloze task setting to study typicality, but they focus on identifying whether the model has access to prototypical properties of concepts (e.g., a ball is round).

3 Problem Setting and Framework

This section presents our general framework for assessing whether a lexical representation model (i.e., a CLM or semantic network) is aware of category structure; the specific probing approaches will be described in Sec. 4. Let us assume a generic conceptual space consisting of a set $\mathcal{C}$ of categories as well as a set $\mathcal{E}$ of exemplars of these categories. For each category $\mathtt{c}$ in $\mathcal{C}$, we assume a set of n_c exemplars $\mathtt{e}_i^{\mathtt{c}}$ in $\mathcal{E}$ for $i = 1$ to n_c. Following the findings of Rosch (1975), we posit that this conceptual space has a graded structure, in that certain exemplars (e.g., `robin`, `crow`) are more closely representative of their category (e.g., `birds`) than others (e.g., `penguin`, `ostrich`). This can be expressed through a typicality score $t_i^c \in \mathbb{R}_0^+$, often an ordinal scale, for each $\mathtt{e}_i^{\mathtt{c}}$, where $t_i^c > t_j^c$ indicates that exemplar $\mathtt{e}_i^{\mathtt{c}}$ is more typical of category $\mathtt{c}$ than $\mathtt{e}_j^{\mathtt{c}}$. These typicality scores can be obtained from human subjects through various kinds of stimuli, such as direct scoring of category-exemplar word pairs (e.g., *robin-bird*) and scoring of a taxonomic sentence (e.g., *A robin is a bird.*)(see Sec. 5.1).

To assess whether a lexical representation model is aware of such a structure, three main components are needed. As the conceptual space is latent, we first need a **concept-to-word mapping function** from category and exemplar concepts to words. As is done in human studies and previous work, we assume that categories and exemplars are reliably accessed through their corresponding singular words: the concepts `robin` and `bird` are accessed via the words *robin* and *bird*, respectively. This is a simplification, as the same concept can be realized by plural forms, as well as by different (synonymous) words. Given this functional mapping, we will conflate a category $\mathtt{c}$ with its category word c and an exemplar $\mathtt{e}_i^{\mathtt{c}}$ with its exemplar word e_i^c.

Second, we define a **typicality scoring function** over exemplar-category word pairs, which captures how typical the exemplar is of the category. Each model will assign a typicality score s_i^c to each category c and exemplar e_i^c pair. Some of our methods will make use of a text corpus, denoted S, taking the form of a pre-extracted **set of sentences** containing exemplar and/or category words. This corpus S can be viewed as an extra parameter of the typicality scoring function, as it will directly impact the predicted typicality values s_i^c. Such a corpus is required to be able to fully leverage CLMs, whether they are used as probabilistic language models or to produce word vector representations. It is also through this corpus that one hopes to capture contextual word realizations that approximate the different senses associated with the exemplar and category words. Indeed, we hypothesize that one important aspect of assessing typicality between category and exemplar words is to be able to deal with the **polysemy** of these words, predicting that the best methods should be able to disambiguate these words in such a way that the exemplar word is interpreted in a category-compatible sense and the category word in an exemplar-compatible sense. Using WordNet synsets as a proxy for senses, we find that on average, category words are linked to 4.2 and 4.9 senses, while exemplars are linked to

3.3 and 4.7 senses on the typicality datasets of Morrow and Duffy (2005) and Rosch (1975), respectively[1] (see Sec. 5.1 for dataset details). This shows the depth of the polysemy problem for typicality, as each word on average has multiple meanings.

Finally, we need an **evaluation metric** for measuring how well the predicted typicality scores s_i^c are able to mirror the human judgements t_i^c. The most obvious choice for this metric is to use Spearman rank correlation (Spearman, 1904), as this correlation measure is non-parametric and makes no strong (e.g., linear) assumption on the data distribution or the underlying scoring functions.

4 Methods

We distinguish two classes of methods, depending on whether they use CLMs or WordNet. For CLM-based methods, we use BERT (`bert-base-uncased` specifically) as a prototypical CLM to assess whether CLMs are aware of category structure. The framework and methods easily generalize to other CLMs. While the concept-to-word mapping is the same for these methods, the scoring function and the probing corpus are different. The BERT methods fall into two categories: (i) those that use BERT as a language model (BERT-MLM, BERT-SentEmb, BERT-MLM-Taxo), and (ii) those that extract word vectors from BERT (BERT-Avg, BERT-MPro). Note that these are all unsupervised probing methods: when an additional corpus S is used, it is only used to probe BERT, not to fine-tune it. These methods embody different linguistic hypotheses and adopt different ways of dealing with polysemy. Note that for embedding methods, we test all layers.

Secondly, we consider lexical semantic networks to compute typicality scores, using WordNet (Miller, 1995) specifically. For this, we use the Shortest-Path and the Lin (Lin et al., 1998) similarity measures computed on WordNet (Miller, 1995). We compute the information content (IC) values (Resnik, 1995) used in the Lin measure using the additional corpus S. We deal with polysemy by using the maximum similarity between exemplar and category synset pairs to compute typicality scores.

4.1 Using BERT's Language Model

The first class of methods relies on the language modeling abilities of BERT, using the following hypothesis: the more central (or peripheral) an exemplar is to a category, the more (or less) likely it is to be used in the category context(s). This hypothesis can be turned into two different distributional hypotheses, depending on whether we take a *paradigmatic* perspective (i.e., how likely can the exemplar word be substituted for the category word in the category contexts) or a *syntagmatic* perspective (i.e., how likely can the exemplar be used next to its category). Our first two probing methods take a paradigmatic perspective and are reminiscent of the Distributional Inclusion Hypothesis, first proposed by Geffet and Dagan (2005). They compute typicality scores that reflect how successful the substitution of a category word by an exemplar is, measured by conditional word probability (BERT-MLM) or by cosine distance between sentence embeddings before and after substitution (BERT-SentEmb). In both, the set of sentences is restricted to sentences that contain the category words and is denoted by S_c.

4.1.1 Masked Language Modeling (BERT-MLM)

Under this approach, typicality is computed as the conditional probability of seeing the exemplar word in its category's contexts. Specifically, for each sentence in S_c, the category name is masked using the `[MASK]` token and the resulting sentence is passed through BERT and the MLM classification, yielding MLM logits for each subtoken in the vocabulary, which are softmaxed to probabilities. Typicality scores are obtained by averaging across sentences the MLM probabilities for each exemplar subtoken sequence. Formally, the typicality score s_i^c for an exemplar e_i^c with l_i subtokens and category c, given BERT and masked sentences S_c, is computed by

$$s_i^c = \frac{1}{|S_c|} \sum_{j=1}^{|S_c|} \prod_{k=1}^{l_i} BERT_{MLM}(S_{c_k}^j) \qquad (1)$$

where $S_{c_k}^j$ is the kth subtoken of exemplar e_i^c in the jth sentence of S_c, and $BERT_{MLM}(\cdot)$ gives the probability of this subtoken.

4.1.2 Sentence Embedding Modeling (BERT-SentEmb)

In this method, typicality is taken to be a measure of how well a category word can be substituted by its exemplars in sentences S_c without altering the

---

[1] If restricting synsets to only nouns, the average number of category senses 2.9 and 3.4 for the two respective datasets, and 2.4 and 3.1 for exemplars.

overall sentence meaning, which we approximate as the sentence embedding. Recall that in BERT's NSP objective, the `[CLS]` sentence classification token is input to the NSP classification head. For each category name sentence, we compute layer wise activations of this sentence token. Then, for each exemplar, we replace the category word in each sentence with the exemplar and obtain the same activations. The typicality score is the cosine similarity of the original sentence embedding and that of the replaced sentence, averaged across sentences. The sentence embeddings can be obtained from any layer in the model; we test the method separately for all layers. Formally, given a category c and an examplar e_i^c, a set of sentences S_c containing the category word c, and a $SE(\cdot)$ function which outputs a sentence embedding from a specific layer, the typicality score is computed by

$$s_i^c = \frac{1}{|S_c|} \sum_{j=1}^{|S_c|} cos(SE(S_c), SE(S^j_{c \to e_i^c})) \quad (2)$$

where $S^j_{c \to e_i^c}$ is S^j with c replaced by e_i^c, and $cos(\cdot, \cdot)$ is the cosine similarity operator.

As they rely on paradigmatic substitution, neither BERT-MLM nor BERT-SentEmb explicitly attempt to disambiguate exemplar or category words, but one can argue that some form of sense selection happens through BERT's contextual modeling and the selection of category contexts. This ensures that the word *orange* is used in *fruit* compatible contexts, thus hopefully filtering out contexts compatible with other senses (e.g. color), when predicting typicality within the `fruit` category. But there is no disambiguation of the category word, as all of its contexts are randomly sampled. This might introduce some noise as category examples with another sense (e.g., *fruit of their labor*) or even POS (e.g., *the trees fruit early*) might be sampled.

4.1.3 Masked Language Modeling with Taxonomic Sentences (BERT-MLM-Taxo)

Inspired by Misra et al. (2021)'s Taxonomic Sentence Verification method, our third CLM-based method is similar to BERT-MLM, but different in that it uses *taxonomic* propositions instead of sampled sentences. These propositions are in the form *"A(n) X is a(n) Y"*, where *X* and *Y* are exemplars e_i^c and categories c, respectively. Typicality scores are obtained in the same way as BERT-MLM: exemplar subtokens are masked in the taxonomic proposition, MLM logits are obtained from the model, then the product of the softmaxed logits (probabilities) is the typicality score (same as Eq. 1). While these sentences provide only narrow contexts, and are somewhat artificial and different from the BERT's training data, they are informative in providing implicit mutual disambiguation of *both* the exemplar and category words.

BERT-MLM-Taxo is similar to Misra et al. (2021), but it differs in two ways. First, Misra et al. (2021)'s method uses CLMs to compute the probability of the *category* word in the taxonomic proposition (i.e. $P(c|e_i^c)$), while our method does the opposite ($P(e_i^c|c)$) to more resemble how humans are probed for the task. Second, Misra et al. (2021) calculates conditional probabilities for masked language models such as BERT using the formulation introduced in Wang and Cho (2019): separately masking each subtoken in the word and passing these separately masked variants of the taxonomic sentence through the model, summing the masked subtokens across variants. While this method is more theoretically founded (as BERT is not an incremental language model and cannot compute probabilities through the chain rule), it is subject to skewed MLM logits if used on exemplars[2], as a result of words being broken into subtokens (example from Misra et al. (2021): if the word ostrich is segmented into *ostr* and *ich*, then the probability of *ich* given that it is preceded by *ostr* is anomalously high, skewing the sequence probability). Thus, we mask all subtokens of the exemplar and treat the resulting probabilities as a sequence.

4.2 BERT-Based "Static" Representations

The next CLM-based methods derive static word representations from contextual representations computed over a corpus S which includes sentences containing category words as well as sentences containing exemplar words. We use cosine similarity between category and exemplar embeddings for typicality scores $s_i^c = cos(R(e_i^c), R(c))$, where $R(\cdot)$ provides the static representations of the category and exemplar. We derive static word representations by averaging all hidden state activations of a word over S (BERT-Avg), or clustering these hidden states, allowing for sense modulation (BERT-MPro). For this set of methods, our hypothesis is that the vectorial space induced by BERT captures typicality, as opposed to the language mod-

---

[2] All category words are treated as one subtoken.

eling capabilities used in the methods of Sec. 4.1.

4.2.1 Averaged Contextual Embeddings (BERT-Avg)

The first approach for generating static word *type* vectors from contextualized word *token* embeddings is to simply average them (Bommasani et al., 2020). For each sentence in S, we compute and store the contextual representations of the word at each layer of the BERT model. We then average these representations over sentences for each word, giving a static embedding for each layer, for each category and exemplar. The typicality score of a pair (e_i^c, c) is computed as the cosine similarity between the static embeddings of the representation of e_i^c and of the representation of c. As with BERT-SentEmb, we test the method separately for all layers. Note that this approach inherits the problem found in classical static embeddings that it conflates all possible senses of a word.

4.2.2 Multi-Prototype Contextual Embeddings (BERT-MPro)

To further exploit context, we use multi-prototype BERT embeddings (Chronis and Erk, 2020). For each category name and exemplar, we use k-means to cluster the word's contextual embeddings computed from its sampled sentences, yielding, for a given k, k cluster centroids for each layer, which disambiguate the k different possible meanings of the word. Following Chronis and Erk (2020), we predict scores using the $maxsim(\cdot,\cdot)$ function between the cluster centroids of the category name and exemplar, which yields the maximum similarity value of category-exemplar centroid pairings. Formally, for a given k and layer l, and set of cluster centroids $\tau(w)_1^l ... \tau(w)_k^l$ for category c and exemplar e_i^c, the similarity s_i^c is defined by

$$maxsim(c, e_i^c) = \max_{1 \leq j \leq k, 1 \leq t \leq k} cos(\tau(c)_j^l, \tau(e_i^c)_k^l) \quad (3)$$

where $cos(\cdot,\cdot)$ is the cosine similarity measure. Following Chronis and Erk (2020), we sidestep tuning the number of clusters k and simply take as input to the $maxsim$ operator the union of all clusters from $k \leq 15$. Furthermore, we also test the embeddings from all layers separately.

4.3 WordNet-Based Methods

Our second class of methods uses WordNet (Miller, 1995) to compute typicality scores. Let us suppose that a category word c and exemplar word e_i^c are mapped to synsets $s(c)$ and $s(e_i^c)$, respectively. Only noun synsets are considered for c and e_i^c. The hypothesis here is that the more closely linked $s(c)$ and $s(e_i^c)$ are in the WordNet graph, the more the exemplar is typical of the category. To measure this "linkage" between exemplars and categories, we use two WordNet similarity measures, thus yielding two sets of WordNet-based methods.

4.3.1 Shortest Path (WNSP, WNSP-noWSD)

Our first similarity is the shortest path between $s(c)$ and $s(e_i^c)$ along hypernym/hyponym edges of WordNet. Given that exemplars and category words might be linked to multiple synsets, we need to aggregate the similarities between synsets. A first method, called **WNSP-noWSD**, is to average the similarities between all synsets $s(c)$ and $s(e_i^c)$. The second method (**WNSP**) tries to disambiguate between possible synsets, relying on the $maxsim(\cdot,\cdot)$ operator, but defined over synset similarities (instead of cosine similarity over clusters, as in BERT-MPro). Formally, for a category name c and exemplar e_i^c, the typicality score s_i^c is equated to $maxsim(c, e_i^c)$, which is defined by

$$maxsim(c, e_i^c) = \max_{s(c), s(e_i^c)} sim(s(c), s(e_i^c)) \quad (4)$$

where $sim(\cdot)$ denotes the shortest path similarity.

Note that, irrespectively of the disambiguation process, the shortest path similarity possibly lacks expressivity, as synsets of different exemplars can be at the same distance of the category synset.

4.3.2 Lin Similarity (WNIC, WNIC-noWSD)

For a more expressive method, we use the Lin similarity measure (Lin et al., 1998), which computes similarity using the most specific ancestor node and Information Content (IC) (computed via Wikipedia text dump), a measure of specificity for a concept closely linked to frequency (Resnik, 1995), thus combining frequency with hierarchical semantic knowledge. Pedersen (2010) show that augmenting WordNet with IC results in higher correlation with human judgments on similarity and relatedness tasks. Formally, let $IC(\cdot)$ and $lcs(\cdot,\cdot)$ denote information content and least common subsumer (specific ancestor node), respectively, the similarity between synsets $s(c)$ and $s(e_i^c)$ is defined by

$$lin(s(c), s(e_i^c)) = \frac{2 * IC(lcs(s(c), s(e_i^c)))}{IC(s(c)) + IC(s(e_i^c))} \quad (5)$$

We then can aggregate similarities by averaging, which defines the method **WNIC-noWSD**, or by using Equation 4 where $sim(\cdot)$ is chosen to be the Lin similarity. This latter method is called **WNIC**.

5 Experiments

5.1 Datasets

As ground truth typicality ratings, we use three datasets of human typicality ratings, one from Rosch (1975) (young adult ratings) and two from Morrow and Duffy (2005) (young and older adult ratings). The two dataset sources were produced at different time periods (1975 for Rosch and 2005 for M&D), with different sample sizes (209 participants and 54, respectively) and potential cultural differences (US residents and UK residents, respectively) and with different protocols. Human ratings were obtained by scoring a taxonomic sentence for Rosch and by scoring category-exemplar pairs for M&D. The datasets contain varying number of categories (11 for M&D, 10 for Rosch) and exemplars per category (29-128 for M&D, 42-54 for Rosch), with 7 overlapping categories between the two sources.[3] The human ratings between the two splits of M&D are strongly correlated (average category Spearman correlation: 0.857, see Appendix A.2). However, the correlation between the Rosch rankings and M&D rankings on the category-exemplar intersection of the datasets are much lower, likely because of the differences outlined above (average category Spearman: 0.656 and 0.574 for Rosch vs M&D young and older adults, respectively). We follow the preprocessing procedure of Heyman and Heyman (2019), (see Appendix A.1). Our auxiliary corpus S is extracted by sampling sentences from Wikipedia, described in Appendix A.3.

5.2 Baseline and Competing Methods

Word Frequency The first baseline is typicality scores as the exemplar's number of occurrences in the corpus S. It is natural to think that more typical examples are more popular words, and Heyman and Heyman (2019) show that frequency partly accounts for human ratings in the M&D datasets.

PPMI-SVD, Word2Vec Other baselines are the cosine similarities between the category-exemplar static word embeddings from count-based (singular value decomposition of positive pointwise mutual information matrix: **PPMI-SVD**) and prediction-based (**Word2Vec** skip-gram negative sampling) algorithms. For baseline details, see Appendix A.4.

Taxonomic Sentence Verification We use the Taxonomic Sentence Verification method of Misra et al. (2021), using `bert-base-uncased`. The differences with BERT-MLM-Taxo can be found in Sec. 4.1.3.

5.3 Results and Analysis

5.3.1 Method Performance

Mean Spearman correlations for all methods are reported in Table 1. [4]

BERT-based Methods Our first main result is that the BERT-based methods[5] all improve over baselines on the M&D datasets. For the Rosch dataset, only BERT-MPro and Misra et al. (2021)'s method improve over Word2Vec. BERT-MPro has the highest correlations of the BERT based methods across datasets, with Misra et al. (2021)'s method only slightly edging it on the Rosch dataset.

Comparing the substitution-based methods from Sec. 4.1, BERT-SentEmb and MLM-Taxo have similar performance. BERT-MLM-Taxo performs better than BERT-MLM, showing that taxonomic sentences provide narrow yet more informative contexts than a randomly sampled corpus simply based on the category word, as they allow for mutual disambiguation of category and exemplar words. Misra et al. (2021)'s method performs worse than BERT-MLM-Taxo on the M&D dataset but better on the Rosch dataset.[6] These discrepancies across datasets can be attributed to different protocols for obtaining human ratings: scoring category-exemplar pairs for M&D and scoring a taxonomic sentence for Rosch. They also account for the performance differences of the frequency baseline, which yields higher correlation scores on M&D than on the Rosch dataset. Typicality scores are dissociable from lexical frequency when subjects judge whether a sentence is a good example of their idea of the category. Our results are in line with Mervis et al. (1976), showing that Rosch dataset scores are not correlated with frequency.

Comparing the word embedding methods from Sec. 4.2, the performance increase from BERT-Avg to BERT-MPro shows the importance of disam-

---

[3] See Table 4 in Appendix for category/exemplar statistics.

[4] See Table 7 in Appendix A.6 for results by category.

[5] Recall that we use `bert-base-uncased`.

[6] This was the only dataset used in Misra et al. (2021).

biguation. Layer-wise performance can be found in Appendix A.8. It should be noted that for BERT-MPro, the best performing layer was layer 10, confirming Chronis and Erk (2020)'s claim that similarity-based task performance peaks in layers 8-10,[7]. Later layers show more contextual variation (Ethayarajh, 2019) capturing finer-grained sense differences. Inversely, for BERT-Avg earlier layers perform best as they include less contextual variation, making the mean more stable.

WordNet-based Methods For the WordNet-based methods, WNIC achieves the highest correlations across datasets, and is competitive with the best BERT-based methods. This confirms the findings of Pedersen (2010) on other intrinsic tasks such as similarity. Disambiguation in the WordNet methods seems crucial, as the variants using averaging perform worse than their $maxsim$ versions. WNIC's high performance shows the importance of frequency information.

Conclusion The experimental results confirm the importance of handling polysemy for both BERT- and WordNet-based methods. Note that the best performing method (last row), which combines BERT-MPro and WNIC, is discussed in Sec. 6.

5.3.2 Polysemy Analysis

Experimental results show that disambiguation allows the improvements of BERT-MPro over BERT-Avg and WNIC over WNIC-noWSD. We now study whether the improvement is larger when the number of senses is larger. Our hypothesis is that the more polysemous a word, the more imprecise its representation without disambiguation, and the larger the performance increase from using disambiguation with the $maxsim$ operator. This analysis is difficult to do because: (i) we don't have access to "true" senses, and (ii) both exemplar and category words can be polysemous. One solution is to use WordNet synsets as a proxy for senses.

Looking at the estimated polysemy degree for the category words, we find that 8 of the 11 category words are associated with 2 senses or more, with the maximum ("tools") having 8 senses. Aligning with results for M&D-young-adults, we find that BERT-MPro yields the largest correlation increases over BERT-Avg for the categories associated with polysemous category words: BERT-MPro yields above-average ρ increase for 6 of the 8 polysemous categories, with the largest increase (+0.2) for the most ambiguous category word "tools". However, the same analysis on the same dataset with WNIC and WNIC-noWSD does not yield the same trend: while disambiguation leads to a greater relative increase, some of the category words with the least polysemy (animals, insects) have the largest performance improvement from noWSD to WSD (see Appendix A.9 for more details). Looking at the average number of synsets for exemplars, it is not easy to see a trend in the performance improvements. Further analysis is needed and we leave a detailed study on polysemy and typicality for future work.

6 Combining CLMs and Lexical Networks

Given that WNIC and BERT-MPro rankings achieve the highest correlations, questions arise: how similar are the rankings produced by the two methods? If sufficiently different, how can we combine them? To answer these questions, we perform a series of analyses. We constrict this depth-first analysis to just the younger adults M&D dataset.

6.1 Method Complementarity Study

BERT and WordNet provide different models of lexical meaning: BERT is word-oriented and exploits distributional statistical patterns, while WordNet is sense-oriented and exploits more abstract relations. To test whether this leads to different predictions, we compute category-wise Spearman correlations between BERT-MPro and WNIC rankings. The results are given in the fourth column in Table 2. The values range from 0.274 to 0.570 with six categories below 0.4. This moderate correlation shows some complementarity between the two rankings. Next, we present how different the two methods' rankings of certain exemplars can be in the first four columns of Table 3, showing how complementarity manifests on a granular level. This study raises the question of combining BERT-based methods with WordNet-based methods.

6.2 A Simple Ensemble Method

As an ensemble method, we take the raw scores of BERT-MPro and WNIC, convert them to z-scores (separately for each method and category), then sum them.[8] As shown in the last row of Table 1,

---

[7] They denote this as layers 7-9 in their paper (0-indexed); we index starting at 1 to denote the embedding layer as 0.

[8] As an alternative ensemble method, we present a short study of KnowBERT, a CLM enhanced with WordNet knowledge, in Appendix A.10.

Method Class	Method	M&D-Young	M&D-Old	Rosch
Baselines	Frequency	0.323	0.296	0.059
	PPMI-SVD	0.236	0.227	0.294
	W2V	0.260	0.296	0.338
BERT	BERT-SentEmb	0.381	0.373	0.289
	BERT-MLM	0.354	0.358	0.238
	BERT-MLM-Taxo	0.396	0.393	0.303
	Misra et al	0.338	0.280	**0.396**
	BERT-Avg	0.428	0.405	0.298
	BERT-MPro	**0.473**	**0.451**	0.386
WordNet	WNSP-noWSD	0.114	0.045	0.266
	WNIC-noWSD	0.152	0.128	0.318
	WNSP	0.213	0.136	0.295
	WNIC	**0.467**	**0.456**	**0.448**
Ensemble	BERT-MPro + WNIC	**0.547**	**0.531**	**0.528**

Table 1: Mean across categories (See Table 7 in Appendix A.6 for results by category) of Spearman correlations for each method for the three datasets ($p < .001$). Largest correlations among singular methods for each dataset are bold-faced for BERT-based methods and WordNet-based methods. The last line is the overall best method and scores are also bold-faced. Note that for BERT-AVG, BERT-MPro, and BERT-SentEmb, the best performing layer is shown (layers 1, 10, and 11, respectively; see Appendices A.7 and A.8 for a layer wise analysis).

Category	MPro	WNIC	MPro + WNIC	MPro vs WNIC	Avg. Increase
Animals	0.665	0.522	0.697	0.524	0.125
Birds	0.461	0.391	0.533	0.331	0.100
Clothes	0.498	0.491	0.541	0.570	0.024
Flowers	0.206	0.148	0.224	0.351	0.006
Fruits	0.486	0.365	0.469	0.397	0.076
Furniture	0.657	0.635	0.763	0.369	0.140
Insects	0.366	0.429	0.408	0.476	0.043
Instruments	0.527	0.656	0.620	0.498	0.038
Tools	0.366	0.584	0.607	0.274	0.134
Vegetables	0.579	0.600	0.739	0.346	0.102
Vehicles	0.389	0.314	0.418	0.529	0.053
Mean	0.473	0.467	0.547	0.424	0.077

Table 2: Category wise Spearman correlations between each method and human rankings from (first three columns), Spearman correlations between BERT-MPro and WNIC rankings (fourth column), and the increase of performance of the ensemble method over MPro and WNIC (increases are averaged over the two individual methods). Results are shown for the Morrow and Duffy young adult dataset.

BERT-MPro combined with WNIC achieves the highest correlation (greater than 0.5) with human rankings across all datasets, closing the gap on the human performance given by the inter-dataset correlation between Rosch and M&D older adults. The category Spearman correlations for BERT-MPro, WNIC and the ensemble are shown in Table 2. We find that the lower the correlation between the two methods (second from right column), the larger the improvement by combining: we find a rank correlation of -0.555 between the inter-method correlation and the average correlation increase of the ensemble (the last column of Table 2), showing that the more different the two methods rankings, the more ensembling increases performance. In Table 3 are rankings of certain words within their category, showing how the ensemble improves rankings for exemplars that are badly ranked by one of the methods.

7 Conclusion

We show that BERT and WordNet-based methods are able to outperform previous methods in typicality prediction, but only with a disambiguation mechanism, either implicit through the selection of contexts or explicit via the estimation of distinct word senses. These results emphasize the importance of polysemy in assessing typicality, an issue that had been overlooked in previous work. We also show that BERT and WordNet provide complementary information that are relevant to modeling category structure: a simple ensemble of the two leads to the best correlations with human judgements.

We plan to further analyze the differences between CLMs and WordNet in typicality, and find more sophisticated ways to inject semantic knowledge into CLMs. Also, we want to use datasets in other languages, to find if the results generalize.

Category	Word	WNIC	MPro	WNIC + MPro	Human
Birds	owl	15	5	4	12
Birds	swallow	5	22	11	11
Clothes	shirt	10	4	3	6
Clothes	suit	22	8	13	12
Vegetables	lettuce	7	16	11	11
Vegetables	sprout	17	11	12	12

Table 3: Rankings of certain exemplars within their category, as scored by WNIC, BERT-MPro, WNIC + BERT-MPro, and the human ranking. Examples are from the Morrow and Duffy young adult dataset.

Limitations

The human typicality judgements we use from Morrow and Duffy (2005) is limited to English and uses only participants all from the UK, while the rankings from Rosch (1975) are all from the United States; thus, geographical/cultural biases could affect the typicality judgements (certain bird species or clothing exemplars could be more common in different areas, which could affect how human's judge their typicality).

While we try to analyze exactly why disambiguation increases performance (BERT-MPro, WNIC-WSD) in Sec. 5.3.2 and Appendix 8, we acknowledge this analysis is far from perfect: we don't have access to "gold" senses to measure polysemy (just linked WordNet synsets), and no single trend emerges from the analysis across all the methods.

Lastly, one limitation in our BERT-based methods is that we use the `uncased` pretrained model. In hindsight, we might have avoided some ambiguous word uses by using a `cased` model instead: for example, the model might have been able to more easily distinguish between "Apple", the company, and "apple", the fruit. However, the impact might have been marginal, as the model can still rely on contextual clues of the surrounding text, and such cases of ambiguity are arguably rare in the typicality datasets. Another possible limitation, specific to the BERT-MLM and BERT-SentEmb methods, comes from the current sentence sampling procedure, as it is based on the category word only, thus possibly introducing noise for words that have homonyms with different POS (e.g., noun-verb honomyms).

Acknowledgements

We would like to thank the three anonymous EACL reviewers for their helpful comments on this paper. This research was funded by Inria Exploratory Action COMANCHE, as well as by the joint IMPRESS project between Inria and DFKI.

References

Marianna Apidianaki and Aina Garí Soler. 2021. ALL dolphins are intelligent and SOME are friendly: Probing BERT for nouns' semantic properties and their prototypicality. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 79–94, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Richard Beckwith, Christiane Fellbaum, Derek Gross, and George A Miller. 2021. Wordnet: A lexical database organized on psycholinguistic principles. In *Lexical acquisition: Exploiting on-line resources to build a lexicon*, pages 211–232. Psychology Press.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Gabriella Chronis and Katrin Erk. 2020. When is a bishop not like a rook? when it's like a rabbi! multi-prototype bert embeddings for estimating semantic relationships. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 227–244.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Aina Garí Soler and Marianna Apidianaki. 2021. Let's play mono-poly: BERT can reveal words' polysemy level and partitionability into senses. *Transactions of the Association for Computational Linguistics*, 9:825–844.

Maayan Geffet and Ido Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 107–114.

Fritz Günther, Luca Rinaldi, and Marco Marelli. 2019. Vector-space models of semantic representation from a cognitive perspective: A discussion of common misconceptions. *Perspectives on Psychological Science*, 14(6):1006–1033.

Tom Heyman and Geert Heyman. 2019. Can prediction-based distributional semantic models predict typicality? *Quarterly Journal of Experimental Psychology*, 72(8):2084–2109.

Geoff Hollis. 2017. Estimating the average need of semantic knowledge from distributional semantic models. *Memory & Cognition*, 45(8):1350–1370.

Geoff Hollis and Chris Westbury. 2016. The principals of meaning: Extracting semantic dimensions from co-occurrence models of semantics. *Psychonomic bulletin & review*, 23(6):1744–1756.

Abhilasha A Kumar. 2021. Semantic memory: A review of methods, models, and current challenges. *Psychonomic Bulletin & Review*, 28:40–80.

Alessandro Lenci. 2018. Distributional models of word meaning. *Annual review of Linguistics*, 4:151–171.

Dekang Lin et al. 1998. An information-theoretic definition of similarity. In *Icml*, volume 98, pages 296–304.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Gary Lupyan and Molly Lewis. 2019. From words-as-mappings to words-as-cues: The role of language in semantic knowledge. *Language, Cognition and Neuroscience*, 34(10):1319–1337.

Paweł Mandera, Emmanuel Keuleers, and Marc Brysbaert. 2017. Explaining human performance in psycholinguistic tasks with models of semantic similarity based on prediction and counting: A review and empirical validation. *Journal of Memory and Language*, 92:57–78.

Carolyn B. Mervis, Jack Catlin, and Eleanor Rosch. 1976. Relationships among goodness-of-example, category norms, and word frequency. *Bulletin of the Psychonomic Society*, 7(3):283–284.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Kanishka Misra, Allyson Ettinger, and Julia Taylor Rayz. 2021. Do language models learn typicality judgments from text? *CoRR*, abs/2105.02987.

Lorna I Morrow and M Frances Duffy. 2005. The representation of ontological category concepts as affected by healthy aging: Normative data and theoretical implications. *Behavior research methods*, 37(4):608–625.

Gregory Murphy. 2004. *The big book of concepts*.

Ted Pedersen. 2010. Information content measures of semantic similarity perform better without sense-tagged text. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 329–332, Los Angeles, California. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. *CoRR*, abs/1909.04164.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *CoRR*, abs/2002.12327.

Eleanor Rosch. 1975. Cognitive representations of semantic categories. *Journal of experimental psychology: General*, 104(3):192.

Corby Rosset, Chenyan Xiong, Minh Phan, Xia Song, Paul N. Bennett, and Saurabh Tiwary. 2020. Knowledge-aware language model pretraining. *CoRR*, abs/2007.00655.

Charles Spearman. 1904. The proof and measurement of association between two things. volume 15, pages 72–101. University of Illinois Press.

Ivan Vulić, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2017. Hyperlex: A large-scale evaluation of graded lexical entailment. *Computational Linguistics*, 43(4):781–835.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a markov random field language model. *CoRR*, abs/1902.04094.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *CoRR*, abs/2002.01808.

Category	M&D	Rosch
Animals	128	
Birds	73	54
Clothes	79	48
Flowers	45	
Fruits	59	44
Furniture	39	45
Insects	44	
Instruments	59	
Sports		47
Tools	59	49
Toys		42
Vegetables	29	42
Vehicles	68	46
Weapons		52
Total	682	470

Table 4: Number of exemplars per category for two datasets: Morrow and Duffy (2005) and Rosch (1975).

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting bert. *arXiv preprint arXiv:2004.14786*.

A Appendix

A.1 Dataset Preprocessing

We follow the preprocessing procedure of Heyman and Heyman (2019), discarding multi-word examples (such as *red cabbage*) and examples containing punctuation. Although these cases could have been handled by BERT-based methods using subtokens, the composition of several terms is non-trivial for word embedding based models (Lenci, 2018), and we are more concerned with comparing category structure than how the different models handle multi-word examples or punctuation. Also following Heyman and Heyman (2019), we discard categories with less than 20 examples leaving 11 categories in the Morrow and Duffy (2005) datasets and 10 in the Rosch (1975) dataset. As some of the baseline models have fixed vocabularies (PPMI-SVD, Word2Vec, WordNet), we remove any out of vocabulary examples from the dataset. While we could have used median or minimum typicality scores for out of vocabulary examples, we wanted to ensure that the comparison of performance was fair across all methods by only evaluating on examples that have a valid typicality score for all methods.

A.2 Dataset Agreement

Tables 5 and 6 show the category-wise agreement between the human rankings of the three datasets. Notice that the agreement between the two Morrow and Duffy datasets are much higher than that of Rosch and the two M&D datasets.

Category	Spearman
Animals	0.916
Birds	0.868
Clothes	0.858
Flowers	0.747
Fruits	0.891
Furniture	0.778
Insects	0.897
Instruments	0.909
Tools	0.791
Vegetables	0.885
Vehicles	0.887
Mean	0.857

Table 5: Spearman correlations between young adult and older adult typicality rankings for Morrow and Duffy (2005).

	Young vs Rosch	Old vs Rosch	N
Birds	0.561	0.449	43
Clothes	0.736	0.647	33
Fruits	0.871	0.680	38
Furniture	0.762	0.547	20
Tools	0.535	0.613	25
Vegetables	0.326	0.227	22
Vehicles	0.802	0.852	29
Mean	0.656	0.574	30

Table 6: Spearman correlations between the young adult and older adult rankings of Morrow and Duffy (2005) and Rosch (1975) on the intersection of categories and exemplars between the two datasets. The size of the intersection is shown on the far right column.

A.3 Sentence Sampling

For an auxiliary textual dataset S, we use Wikipedia, specifically searching for sentences that contain the singular form of each category word or exemplar, in order to have comparable contexts. We found text from Wikipedia dumps can be noisy (such as: summarization tables, lists of related topics, other sequences that are not actual sentences), so we remove sentences that are too short or long (number of words must be between 5 and 200). For BERT-AVG and BERT-MPro, we sampled 300 sentences for every exemplar and category word. For BERT-MLM and BERT-SentEmb, we sampled 10000 sentences for each category.

A.4 PPMI-SVD and Word2Vec Details

The PPMI matrix was computed from English Wikipedia using a window size of 2. We experimented with a window size of 5, but found the performance to be worse. We also use off-the-shelf Word2Vec (Mikolov et al., 2013) embeddings (skip-gram with negative sampling, 300 dimensions, GoogleNews training corpus).

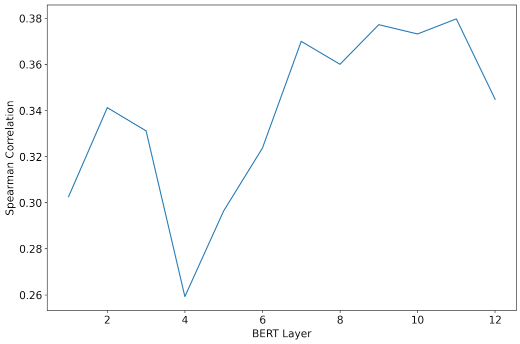
A.5 Runtimes

Frequency, PPMI-SVD, Word2vec, and both WordNet baseline methods take <5 seconds to compute typicality scores on CPU. The Wikipedia word counts used in Frequency and WordNet-IC take 1hr to compute on CPU. The sentence sampling from Wikipedia takes 2hrs to complete. BERT-WordPiece takes 1 min to complete on CPU. BERT-AVG and BERT-MLM take 3hrs to complete on the GPU mentioned in the text; BERT-SentEmb, 5hr; and BERT-MPro, 10hr.

A.6 Results by Category

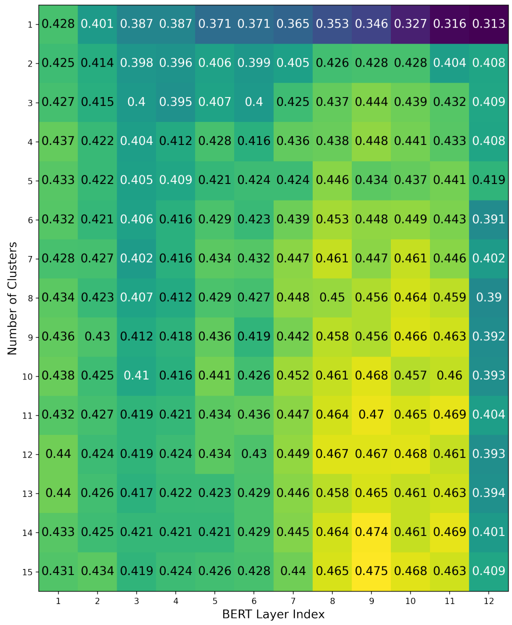
Table 7 shows the Spearman's correlation values for each individual category. Note that just the top performing methods are shown.

A.7 BERT-SentEmb Performance by Layer



The above figure shows the mean Spearman correlations by layer for the BERT NSP method.

A.8 BERT-AVG and MultiPrototype Hyperparameter Performance



The above image shows average Spearman's correlation across all categories for BERT multi-prototype embeddings for each combination of layer index and number of cluster. Note that the top row (number of clusters = 1) is the same as the simple average of contextual representations.

A.9 Detailed Polysemy Analysis

To see whether the $maxsim$ disambiguation mechanism is actually increasing performance more for more polysemous examples, we can compare the performance of the same methods with and without disambiguation when evaluated on words with different number of linked WordNet synsets (a proxy for the degree of polysemy). As shown in Table 8, in general, BERT-MPro's disambiguation increases performance over BERT-Avg more for more polysemous categories (as measured by the synsets linked to the category word). However, this general trend is not seen when comparing WNIC to WNIC-noWSD.

Category	Misra et al			Bert-MLM-Taxo			Bert-Avg			Bert-MPro			WNIC		
	M&D-Y	M&D-O	Rosch	M&D-Y	M&D-O	Rosch	M&D-Y	M&D-O	Rosch	M&D-Y	M&D-O	Rosch	M&D-Y	M&D-O	Rosch
Animals	0.558	0.525		0.528	0.508		0.569	0.530		0.665	0.649		0.522	0.555	
Birds	0.284	0.130	0.308	0.442	0.332	0.191	0.440	0.311	0.203	0.461	0.317	0.169	0.391	0.310	-0.014
Clothes	0.205	0.115	0.461	0.527	0.587	0.171	0.506	0.528	0.301	0.498	0.573	0.484	0.491	0.487	0.570
Fruits	0.232	0.191	0.354	0.287	0.357	0.278	0.390	0.338	0.406	0.486	0.433	0.592	0.365	0.403	0.442
Furniture	0.600	0.440	0.548	0.577	0.429	0.484	0.629	0.598	0.320	0.657	0.602	0.235	0.635	0.553	0.658
Flowers	0.240	0.146		0.122	0.068		0.152	0.011		0.206	0.111		0.148	0.105	
Insects	0.175	0.328		0.411	0.379		0.437	0.473		0.366	0.503		0.429	0.415	
Instruments	0.309	0.331		0.667	0.715		0.626	0.686		0.527	0.602		0.656	0.659	
Sports			0.467			0.536			0.530			0.379			0.498
Tools	0.421	0.259	0.100	0.066	0.220	0.220	0.160	0.257	0.040	0.366	0.318	0.086	0.584	0.486	0.494
Toys			0.143			-0.001			-0.089			0.398			0.155
Vegetables	0.390	0.312	0.241	0.364	0.281	-0.006	0.501	0.397	0.073	0.579	0.482	0.088	0.600	0.646	0.409
Vehicles	0.307	0.301	0.682	0.364	0.444	0.733	0.293	0.323	0.743	0.389	0.373	0.651	0.314	0.396	0.701
Weapons			0.655			0.429			0.454			0.779			0.567

Table 7: Category wise Spearman correlations with human typicality rankings for the four best BERT-based models and the best WordNet-based model.

Category	MPro inc.	WNIC inc.	Cat. Synsets	Exem. Synsets
Animals	0.096	0.517	1	2.992
Birds	0.021	0.362	6	2.712
Clothes	-0.009	0.397	4	4.241
Flowers	0.055	0.028	4	2.022
Fruits	0.096	0.409	5	3.102
Furniture	0.028	0.165	1	4.590
Insects	-0.071	0.432	2	2.705
Instruments	-0.099	0.456	1	2.423
Tools	0.206	0.155	8	5.136
Vegetables	0.078	0.317	2	3.345
Vehicles	0.095	0.222	4	3.412

Table 8: Increase in Spearman's correlation from using BERT-MPro over BERT-Avg (middle left column) and from using WNIC over WNIC-noWSD (middle right column) for each category. The number of linked WordNet synsets for each category and their exemplars are shown in the right columns. Results are shown for the Morrow and Duffy (2005) young adult dataset.

A.10 KnowBERT Results

The combination of CLMs and knowledge graphs is an active research area, mainly the design of knowledge enhanced CLMs (i.e. Wang et al. (2020), Rosset et al. (2020)). We use KnowBERT (Peters et al., 2019), in which knowledge bases are embedded into BERT via an integrated entity linker, which retrieves relevant entity embeddings and updates the hidden states (we leave an in-depth description to the paper). We consider a KnowBERT model with WordNet as the knowledge base[9] with the multi-prototype method for computing typicality scores. We get an average Spearman correlation of 0.446, below that of WNIC and BERT-MPro, showing that a general method for enhancing CLMs with WordNet does not work well for typicality.

---

[9] Candidate WordNet synsets are found for KnowBERT using a similar procedure as we use to find all possible synsets.

An Empirical Study of Clinical Note Generation from Doctor-Patient Encounters

Asma Ben Abacha
Microsoft Health AI
abenabacha@microsoft.com

Wen-wai Yim
Microsoft Health AI
yimwenwai@microsoft.com

Yadan Fan
Nuance Communications
yadan.fan@nuance.com

Thomas Lin
Microsoft Health AI
tlin@microsoft.com

Abstract

Medical doctors spend on average 52 to 102 minutes per day writing clinical notes from their patient encounters (Hripcsak et al., 2011). Reducing this workload calls for relevant and efficient summarization methods. In this paper, we introduce new resources and empirical investigations for the automatic summarization of doctor-patient conversations in a clinical setting. In particular, we introduce the MTS-DIALOG dataset; a new collection of 1,700 doctor-patient dialogues and corresponding clinical notes. We use this new dataset to investigate the feasibility of this task and the relevance of existing language models, data augmentation, and guided summarization techniques. We compare standard evaluation metrics based on n-gram matching, contextual embeddings, and Fact Extraction to assess the accuracy and the factual consistency of the generated summaries. To ground these results, we perform an expert-based evaluation using relevant natural language generation criteria and task-specific criteria such as critical omissions, and study the correlation between the automatic metrics and expert judgments. To the best of our knowledge, this study is the first attempt to introduce an open dataset of doctor-patient conversations and clinical notes, with detailed automated and manual evaluations of clinical note generation.

1 Introduction

The recent progress in automatic summarization has been highly influenced by large transformer-based language models and the availability of large-scale datasets. The summarization of medical conversations is well positioned to benefit from similar approaches, but is facing domain- and task-specific obstacles such as the lack of data and relevant evaluation protocols.

Medical doctors spend on average 52 to 102 minutes per day writing clinical notes from their conversations with the patients (Hripcsak et al., 2011). The time spent with Electronic Health Record systems contributes to work-life imbalance, dissatisfaction, high rates of attrition, and a burnout rate exceeding 50% (Arndt et al., 2017). Summarization models could play a key role in reducing that workload by generating clinical notes from the doctor-patient encounters (Knoll et al., 2022).

Summarizing doctor-patient conversations for a clinical setting brings its own set of challenges and nuances in addition to the typical natural language understanding and generation components. For instance, omission of critical medical facts is likely to alter patient outcomes and should be one of the critical/deciding factors in adopting one summarization system over another. Hallucinations are also likely to impact the clinical outcome if they are not avoided (or detected with a high accuracy).

Designing and improving summarization models that address these challenges can benefit from wider research efforts on the task. However, the lack of publicly available doctor-patient dialogue datasets limits wider research efforts from the NLP community in this summarization task. In this paper, we tackle the lack of data for the task by building a new dataset of doctor-patient conversations and associated clinical notes. We avoid privacy infringement risks, by creating simulated conversations from publicly available clinical notes.

Our main contributions are: **(i)** a new dataset of 1,700 doctor-patient conversations (16k turns and 18k sentences) and their summarized clinical notes (6k sentences). To the best of our knowledge, this is the first publicly-available dataset of medical conversations and associated notes at this scale, **(ii)** an evaluation of several SOTA summarization models, including variants that use existing relevant datasets, augmented training data, and guided summarization for medical conversation summarization, and **(iii)** a study of standard evaluation metrics, domain-specific metrics, and expert judgments for the task, including computing the corre-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2291–2302
May 2-6, 2023 ©2023 Association for Computational Linguistics

lation between the automatic and manual scores for the evaluation of the generated clinical notes[1].

2 Related Work

Summarization datasets and evaluation methods are often centered around large news articles such as the CNN/DailyMail dataset of 313k newspaper articles (Hermann et al., 2015) and XSum (Narayan et al., 2018) with 227k BBC articles with single-sentence summaries.

Dialogue summarization is relatively less studied in open domain, with a few efforts tackling the summarization of conversations and meetings (Goo and Chen, 2018; Li et al., 2019; Shin et al., 2022) and the creation of meeting summarization datasets (Janin et al., 2003; Carletta, 2007).

Medical conversation summarization is also under-studied, except for a few recent efforts (Liu et al., 2019; Kazi and Kahanda, 2019; Yim and Yetisgen, 2021; Michalopoulos et al., 2022). For instance, (Joshi et al., 2020) applied a pointer generator model to generate summaries from doctor-patient dialogues in telemedicine. Instead of modeling the whole dialogue, they trained the model on snippets taken from the dialogue turns. (Zhang et al., 2021) fined-tuned a pre-trained BART model to automatically generate summaries from doctor-patient conversations. However, they only focused on two specialties (internal medicine and primary care) and the training data only consist of HPI (History of Present Illness) section. (Krishna et al., 2021) proposed an algorithm called Cluster2Sent to generate SOAP notes from doctor-patient conversations. Their summarization model involves both abstractive and extractive methods. (Enarvi et al., 2020) studied both RNN and transformer-based sequence-to-sequence models for generating medical reports. They experimented on Orthopedics data and found that sequence-to-sequence modeling is more promising. The datasets of the studies mentioned above are not made public.

(Song et al., 2020) investigated medical conversation summarization from a Chinese conversational corpus. They applied a hierarchical encoder-tagger model for extractive summarization to generate two types of summaries; one for problem statement and one for the treatment recommendations. Several other medical conversation datasets have been recently created from Chinese online health platform conversation (Liu et al., 2020; Zhang et al., 2020; Lin et al., 2019) but they do not include conversation summaries.

(Moramarco et al., 2022b) studied the task of consultation note generation on a small set of 57 transcript-note pairs (Papadopoulos Korfiatis et al., 2022) and performed a correlation study with several automatic metrics. As their focus was to compare automated metrics with human judgements, they chose models that would "produce different outputs to cover a wider range of errors" instead of attempting to benchmark SOTA summarization models. From the set of metrics they tested, they noted that character-based Levenshtein distance, BERTScore, and METEOR performed best for evaluating the note generation task.

3 MTS-DIALOG

3.1 Data Creation

Our data creation approach consists in generating simulated doctor-patient conversations from publicly available clinical notes. To gather these clinical notes/summaries, we collected notes from the public Mtsamples collection, which provides de-identified clinical notes[2] (South et al., 2014; Moramarco et al., 2022a). The selected clinical notes cover the six most frequent note types and specialties in the collection, including: General Medicine, SOAP (Subjective, Objective, Assessment, Plan), Neurology, Orthopedic, Dermatology, and Allergy/Immunology.

Eight trained annotators, with medical backgrounds, were given all the sections from the clinical notes and asked to create clinical conversations from one section at a time according to detailed guidelines. These guidelines were developed based on an analysis of a large private collection of hundreds of real doctor-patient conversations and associated notes. The annotation guidelines included:

- Conversation creation rules: conversations should be (i) written as it pertains to the day of the visit with the patient and (ii) framed in the context of either an outpatient visit or emergency room visit.
- Medical terms rules: The clinical note may have a more detailed referring expression to the same problem, treatment, or test, than

---

[1] The dataset, source code, and annotations are available at `https://github.com/abachaa/MTS-Dialog`

[2] www.mtsamples.com. Mtsamples categorizes notes to at least one of 40 specialty/note-type labels. Each note explicitly marks section headers as bolded HTML tags.

Dialogue:
Doctor: My chart here says that you're eighty three years old, is that correct, ma'am?
Patient: Yes doctor, that's correct, I just had my birthday.
Doctor: Happy belated birthday! How have you been doing since your last visit?
Patient: Well, my cancer hasn't needed phlebotomies for several months now, which is good.
Doctor: That's great, you have been treated for polycythemia vera, correct?
Patient: Yes, that's the one.
Doctor: I also see you're unassisted today, which is also great.
Patient: Yeah, having some independence is nice.

Section header: History of Present Illness

Section text: The patient is an 83-year-old female with a history of polycythemia vera. She comes in to clinic today for followup. She has not required phlebotomies for several months. The patient comes to clinic unaccompanied.

Table 1: Example of a doctor-patient conversation and associated note/summary from the MTS-DIALOG dataset.

what would be represented in the conversation; for example "Open reduction internal fixation (ORIF)" from the clinical note could be translated to "We will have to do surgery on it" in the conversation.

- Imaginary but plausible rule: if the clinical note is underspecified, it is possible to create a conversation as long as it is plausible. However, the dialogues should be created so that the transcripts are more detailed than the associated clinical notes (except for problems, treatments and tests as mentioned above).
- Formatting rules: annotators should follow standard transcript guidelines such as writing words as they would be pronounced, and capitalization and punctuation rules.
- Conversation characteristics: the goal is to create a dataset with as much variation as possible to mimic real doctor-patient medical visits, including the use of speech disfluencies such as false starts, filler words, interjections, interrupting speech, corrections by the speaker to previous information, using slang and vernacular, and colloquial terms.

We also normalized the 279 original section headers from the notes into 20 types of first-level headers (e.g. *assessment, allergy, diagnosis, exam, medications, past medical history, past surgical*).

The final MTS-DIALOG dataset includes 1,701 pairs of dialogues and associated sections from the clinical notes. Table 1 presents an example from the dataset. The number of pairs from the respective specialties and note types were General Medicine:1,035, SOAP:79, Neurology:296, Orthopedic:208, Dermatology:56, and Allergy/Immunology:27. The dataset was created over a cumulative total of approximately 1,800 hours. Additional statistics are shown in Table 2.

	Dialogue			**Summary**	
	Turns	Sentences	Words	Sentences	Words
count	15,969	18,406	241,685	5,870	81,299
mean	9	11	142	3	48
max	103	136	1,951	57	1,182
25-perc	4	4	48	1	6
50-perc	6	7	88	2	18
75-perc	12	14	176	4	55

Table 2: Statistics of the MTS-DIALOG Dataset.

3.2 Data Quality

The quality of the MTS-DIALOG dataset is ensured by three stages: (1) only candidates with medical training were hired as annotators (e.g. former medical scribes), and (2) the training for this task involved one-on-one periodic feedback during initial stages by an experienced trainer; (3) at the completion of the entire dataset, a separate and independent validation step was conducted to formally evaluate the corpus against a rubric grading system. This independent evaluation graded the conversations according to their adherence with the annotation guidelines and content relevance and coverage w.r.t. the initial clinical note. Table 3 presents the results of this manual validation. The validator was additionally tasked to perform minor corrections (e.g. misspellings or adding back missed information), ensuring that the final data quality would be higher than those reported.

3.3 Comparison with Real Data

The MTS-DIALOG dataset consists of real notes and synthetic conversations that simulate doctor-patient encounters to avoid the public release of private doctor-patient conversations. To study the impact of relying on synthetic data, we investigated the resemblance of the MTS-DIALOG data with real conversations through a blind evaluation of two equal subsets of 52 conversations randomly extracted from (i) the MTS-DIALOG dataset and (ii) a private collection of recorded and tran-

Score	Description	Freq (%)
0.1	Unviable, content is not covered.	25 (1%)
0.3	Content is covered but with logical and medical errors OR Major content is not covered OR At least one major socio-cultural dialogue discrepancy anomaly.	189 (11%)
0.5	Acceptable but some minor content or sociocultural discrepancy issues.	70 (4%)
0.7	Acceptable but with misspellings or transcription rules errors only.	551 (32%)
1.0	Follows guidelines completely, logically and medically sound, no content errors or other issues.	866 (51%)

Table 3: Data validation scoring rubric and frequencies based on manual validation, prior to correction.

scribed doctor-patient conversations. Turns from the real/private subset were selected to match the length of the MTS-DIALOG turns. A medical expert with experience working as a medical scribe in hospitals independently performed this blind annotation by labeling each conversation as real, synthetic, or unknown, together with a written explanation behind each annotation. The annotation criteria included annotating the characteristics of the conversations in terms of disfluency and interruptions.

Label	#Ref	#LabeledAs	TP	FP	FN	P	R	F1
Real	52	34	29	5	23	0.85	0.56	0.67
Synthetic	52	69	47	22	5	0.68	0.90	0.78
Unknown	0	1	0	1	0	0	1	0
Total	104	104	76	28	28	0.73	0.73	0.73

Table 4: Blind Evaluation: #Ref (number of reference samples), #LabeledAs (number of assigned labels), TP (True Positive), FP (False Positive), FN (False Negative), P (Precision), R (Recall), and F1 Score.

The results of this annotation (cf. Table 4) show that 55.77% of real conversations (29/52) were annotated as real, 42.31% of real conversations (22/52) were labeled as synthetic, and 9.61% of synthetic data (5/52) labeled as real. The medical expert's blind labeling was incorrect 26.92% of the time (28/104). If synthetic and real were indistinguishable, the incorrect rate would be 50%.

Dataset	#Turns	MaxTurnLen	#Disfluency	#Interruptions
MTS	7.10 (369)	18.54 (n/a)	0.63 (33)	1.06 (55)
Real	7.10 (369)	34.00 (n/a)	1.77 (92)	1.98 (103)

(a) Statistics on the full annotated datasets

Dataset [#Conversations]	#Turns	MaxTurnLen	#Disfluency	#Interruptions
MTS-mistaken-as-Real [5]	5.0	11.2	0.4	1.0
Real-mistaken-as-MTS [22]	5.9	27.2	1.4	1.5

(b) Statistics on the incorrectly labeled subsets

Table 5: Blind Evaluation: Statistics (Mean (Sum))

A common explanation given by the medical expert for cases where synthetic data was labeled as real is that the content seemed realistic, despite the statistical comparison in Table 5 which shows that the MTS-DIALOG conversations had less disfluencies and interruptions on average. Common explanations for labeling real data as synthetic included the conversation being "to-the-point", clear with low disfluencies, easy to follow, containing abrupt subject changes and containing colloquial speech.

This difficulty in distinguishing synthetic from real indicates that the MTS-DIALOG dataset is a valuable initial dataset, for use in training and benchmarking models for real-world applications, including using the MTS-DIALOG data for data augmentation or as pre-training data for later fine-tuning on (private) real conversations data.

4 Methods

4.1 Summarization Models

To study the specificity of doctor-patient conversation summarization and the relevance of evaluation methods, we generated summaries using several SOTA transformer-based models (e.g. BART (Lewis et al., 2020) and Pegasus (Zhang et al., 2020)), including variants that are pre-finetuned using relevant datasets (e.g. XSum (Narayan et al., 2018) and Samsum (Gliwa et al., 2019)), as well as augmented training data and guided summarization as described in the following sections.

4.2 Data Augmentation via Back-translation

Relevant data augmentation is an effective technique to avoid over-fitting and increase the performance of neural methods. In particular, back-translation augmentation consists in translating the text to another language and then translating it back to the original language. We used two different auxiliary languages to add more variety in the back-translated sentences[3]. To reduce translation errors, we selected French and Spanish for their lexical proximity with English, and high-performing translation models (Tiedemann and Thottingal, 2020).

4.3 Guided Summarization

Several guidance signals can be used to control the output of summarization models. A clinical note

---

[3] We also release the augmented dataset of 3,603 pairs of medical conversations and associated notes.

consists of several sections such as Family History and Assessment, and the simulated conversations from the MTS-DIALOG dataset were generated independently for each section. In the guided summarization trial, we used the section header as a prefix in the training data to guide the summarization of the doctor-patient conversation. This allows the model to learn to generate both the signal (section header) and the following signal-guided summary.

4.4 Evaluation Methods

Despite recurrent efforts in summarization evaluation, automatic evaluation of generated summaries still has several limitations and biases (Hardy et al., 2019; Fabbri et al., 2021; Ben Abacha et al., 2021). These limitations can misinform current and future research efforts and orient neural networks towards optima that do not accurately reflect the relevance and quality of generated summaries. For instance, commonly used evaluation metrics such as ROUGE do not assess whether summaries are factually consistent with source documents, include critical errors, or lack important information (Goodrich et al., 2019). Manual evaluation is another method to assess the quality of the generated summaries, but is time consuming and relies on the availability of domain experts to rate the summaries. Howcroft et al. (2020) examined 165 NLG papers with human evaluations and concluded that the field is in urgent need of standard evaluation methods and terminology.

Taking into account these different factors, we evaluate the generated summaries using a variety of evaluation metrics based on n-gram matching, pre-trained contextualized embeddings (BERTScore), learned metric (BLEURT), automatic fact-based metrics (Fact Scores), and manual evaluation performed by medical experts. BERTScore (Zhang* et al., 2020) uses the pre-trained contextual embeddings from BERT and matches words in candidate and reference texts by cosine similarity. We use two variants: BERTScore-M1 based on the default roberta-large model and BERTScore-M2 based on the deberta-xlarge-mnli model, which was shown to have the best correlation with human evaluation. BLEURT (Sellam et al., 2020) is a learned metric, based on BERT and a pre-training scheme that uses millions of synthetic examples. We use the latest checkpoint BLEURT-20 that correlates better with human judgment and the F1 variants of all the automatic metrics.

For the fact-based evaluation, we utilize a machine learning-based medical fact extraction system to extract medically relevant facts. A medical fact consists of one core attribute with or without one or more other attributes, such as laterality or bodysite. For example, the medical facts identified by the fact extraction system for the input "*the patient has rash on the upper arms*" are *<FINDING_CORE> rash <LATERALITY> upper <BODYSITE> arms*. The Fact Score metric provides F1-score of the extraction of medically relevant facts. The first variant Fact-Core relies on the extraction of seven core fact attributes: *'Procedure_Core', 'Disorder_Core', 'Finding_Core', 'Medication_Core', 'Substance_Use_Core', 'Vital_Sign_Core', and 'Allergy_Core'*. The Fact-Full variant combines these core facts and five additional attributes: *'Negation', 'Hedge', 'Status', 'Laterality', and 'Bodysite'*.

We also investigate the correlation between the automatic metrics and the expert judgments.

5 Experiments

We train the summarization models on 4 Nvidia Tesla K80 GPUs for 4 epochs. We set the learning rate to 3e-05 and regularize the training with an L2 weight decay of 0.1. We use a test set of 100 conversations and notes, randomly selected from the MTS-DIALOG dataset. The remaining pairs are used for training (1,201 pairs) and validation (400 pairs).

5.1 Automatic Evaluation

Table 6 presents the results of different summarization models fine-tuned and evaluated on the MTS-DIALOG dataset. We picked the best performing model from each category (four models in total) for subsequent studies. Table 7 compares the results of these four summarization models using ROUGE-N, Fact Scores, BERTScore, and BLEURT. Examples of generated summaries are shown in Table 9.

These first results highlight the importance of relevant pre-finetuning targets, with XSum yielding substantially better results as a first pre-finetuning stage compared to CNN/DailyMail (40.15 vs. 32.01 ROUGE-1 score). A portion of this differential could be explained by the relatively short length of the MTS-DIALOG summaries (three sentences and 48 words on average) which is likely closer to the length of output for extreme summarization than the longer output summaries in

	ROUGE-1	ROUGE-2	ROUGE-L
Baseline Models			
Pegasus-large	27.62	10.99	23.03
BART-large [Model#1]	**30.42**	**12.03**	**26.91**
Pre-Finetuning (PFT)			
Pegasus-pubmed	24.20	8.49	18.03
Pegasus-xsum	32.88	13.75	27.43
BART-cnn-samsum	32.01	13.93	23.05
BART-xsum-samsum [Model#2]	**40.15**	**18.04**	**32.56**
Guided Summarization (GS)			
BART-cnn-samsum (GS)	32.68	14.14	23.39
BART-xsum-samsum *(GS)* [Model#3]	**42.04**	**17.59**	**34.85**
GS + Data Augmentation (DA)			
BART-cnn-samsum *(GS+DA)*	33.29	14.58	24.30
BART-xsum-samsum *(GS+DA)* [Model#4]	**42.52**	**17.50**	**34.90**

Table 6: Results of the summarization models fine-tuned and evaluated on the MTS-Dialog dataset.

	Model#1 (BART-large)	Model#2 (BART-PFT)	Model#3 (BART-PFT-GS)	Model#4 (BART-PFT-GS-DA)
ROUGE-1	0.3042	0.4015	0.4204	**0.4252**
ROUGE-2	0.1203	**0.1804**	0.1759	0.1750
ROUGE-L	0.2691	0.3256	0.3485	**0.3490**
Fact-Core	0.2381	**0.3753**	0.3466	0.3496
Fact-Full	0.1643	**0.2264**	0.2126	0.2128
BERTScore-M1	0.3090	0.3830	**0.4190**	0.4120
BERTScore-M2	0.2850	0.3680	0.4000	**0.4080**
BLEURT-20	0.4316	0.5003	0.5089	**0.5123**

Table 7: Evaluation of the summarization models using lexical, fact-based, embedding-based, and learned metrics (Macro Average F1 scores over the summaries).

CNN/DailyMail.

Data Augmentation (DA) led to slight improvements across all metrics except ROUGE-2 and BERTScore-M1 as shown in Table 7. Guided Summarization (GS) led to a consistent improvement across all automated metrics except for ROUGE-2 and the Fact-based metrics.

5.2 Expert-based Manual Evaluation

The manual evaluation of the generated summaries is performed using NLG criteria such as Fluency and Non-redundancy, and medical criteria such as Critical Omissions based on fact extraction. For this evaluation, we define a fact as information that cannot be written in more than one sentence. For instance: the sentence "*The father died of stroke at age 89.*" could be written in three sentences: "*The father died*", "*He was 89 yo.*", and "*Stroke was the cause of death.*" and thus contains three facts.

The manual summary evaluation criteria are:

- Fluency: Is the summary fluent to read? (0:"none", 1:"low", 2:"average", 3:"high")
- Non-redundancy: How redundant is the summary? (0-3)
- Critical Omissions: What is the number of medical facts that were omitted?
- Hallucinations: What is the number of hallucinated facts?
- Correct Facts: How many facts in the summary are correct according to the input conversation?
- Incorrect Facts: How many facts are incorrect outside of hallucinations (e.g. wrong age)?

We compute the following scores from the manual counts:

$$FactualPrecision = \frac{\#CorrectFacts}{\#SystemOutputFacts}$$

$$FactualRecall = \frac{\#CorrectFacts}{\#ReferenceFacts}$$

$$HallucinationRate = \frac{\#HallucinatedFacts}{\#SystemOutputFacts}$$

$$OmissionRate = \frac{\#OmittedFacts}{\#ReferenceFacts}$$

- *SystemOutputFacts=Correct+Incorrect+Hallucinated*

	Kappa	F1 (exact)	F1 (tol=1)	F1 (tol=2)	Pearson's correlation
Number of key/reference facts	0.494	0.540	0.780	0.930	0.980
Number of correct facts	0.599	0.710	0.930	1.000	0.841
Number of omitted facts	0.366	0.440	0.740	0.890	0.957
Number of incorrect facts	0.305	0.900	0.990	1.000	0.717
Number of hallucinated facts	0.541	0.920	1.000	1.000	0.695
MACRO-AVG	0.467	0.675	0.875	0.957	0.862

Table 8: Fact count agreements.

	Generated Summary	Reference Summary
1	Family history is significant for coronary artery disease, hypertension, diabetes mellitus, and cerebrovascular disease.	Family history is remarkable for heart disease, cerebrovascular disease, diabetes, and hypertension.
2	The patient denies any history of depression, suicidal ideation, chest pain, shortness of breath, nausea, vomiting, numbness, weakness, or tingling.	Please see history of present illness. Psychiatric: She has had some suicidal thoughts, but no plans. She denies being suicidal at the current time. Cardiopulmonary: She has not had any chest pain or shortness of breath. GI: Denies any nausea or vomiting. Neurological: No numbness, weakness or tingling.
3	Significant for frequent flyer status, anemia, anxiety, bipolar disorder, and iron deficiency anemia. Surgery history is positive for tubal ligation.	1. Bipolar disorder. 2. Iron deficiency anemia. 3. Anxiety disorder. 4. History of tubal ligation.
4	He is a non-cigarette smoker and non-ETOH user. He is single and he has no children. He works as a payroll representative and previously did lot of work in jewelry business, working he states with chemical.	She is a nonsmoker and nondrinker. She is single with no children. Currently works as a payroll representative. Prior to that, she worked in the jewellery business with chemical.
5	The patient was a baby born at 32 weeks' gestation at 4 pounds 11 ounces and placed in an incubator for 3 weeks. He had jaundice, but was not given any treatment.	32 weeks gestation to a G4 mother and weighed 4#11oz. He was placed in an incubator for 3 weeks. He was jaundiced, but there was no report that he required treatment.

Table 9: Examples of generated summaries by Model #4.

To assess the effort level for the end user (clinicians who will use/edit the system generated summary), we compute Levenshtein edit distance (minimum # of character insertion, deletion, substitution or transposition operations) (i) between system summary and reference summary, and (ii) between the initial system summary and a human-corrected version of that summary which fixes all issues. The normalized edit distance is then calculated by dividing the Levenshtein distance by the length of the longest summary between reference and system.

To measure the consistency of these expert evaluations, a common set of 100 reference-system outputs were labeled independently by two trained annotators with medical backgrounds. The remainder of the evaluated data (300 system summaries) was single annotated.

The Pearson correlations between the annotations for the rating-based scores were 0.631 for Fluency and 0.894 for Non-redundancy. We also calculated Cohen's kappa and F1 score for the number of reference facts as well as the correct, omitted, hallucinated, and incorrect system facts. With strict Cohen's kappa and F1, we obtained values 0.467 and 0.675 respectively. As these measures penalize harshly for being off by one or two facts, we also measured a relaxed F1 allowing a count mismatch of one or two facts, which showed an inter-annotator F1 agreement of 0.875 and 0.957 respectively (cf. Table 8). The Pearson correlation between the two annotators' fact counts was also high at a macro-average of 0.862.

The results of the manual evaluation are reported in Table 10. In large part they confirm the automatic evaluation results from Table 7, with Models 2-4 performing better than the baseline Model #1 in terms of Factual Recall and Factual F1. In comparison with Model #3, Data Augmentation (Model #4) led to better fluency, non-redundancy, and factual Recall and F1 with less critical fact omissions, but increased the hallucination rate to 3% from 1% for Model #3.

The model without guided summarization or data augmentation (Model #2) achieved similar results to the best model that employed both (Model #4). Model #2 was also ranked higher according to the automatic fact extraction metrics (cf. Table 7). These results suggest that guided summarization improves the precision of the summary facts (+5.5% improvement) but lowers recall (-5%), with data augmentation reversing the trend. Fact-based performance was thus shown to be more

Evaluation Criteria	Model #1 (BART-large)	Model #2 (BART-PFT)	Model #3 (BART-PFT-GS)	Model #4 (BART-PFT-GS-DA)
Summary Quality Evaluation				
Fluency [0-3] ↑	2.33	**2.44**	2.31	2.37
Non-redundancy [0-3] ↑	**2.97**	2.85	2.90	2.93
Fact-based Evaluation				
Factual Precision ↑	**0.9492**	0.8917	0.9408	0.9010
Factual Recall ↑	0.5324	0.6671	0.6341	**0.6685**
Factual F1 Score ↑	0.6822	0.7632	0.7576	**0.7675**
Hallucination Rate ↓	0.02	0.04	**0.01**	0.03
Omission Rate ↓	0.47	**0.33**	0.37	**0.33**
Effort Level Assessment				
Levenshtein Edit Distance [wrt Correction] ↓	0.5770	**0.4944**	0.5858	0.5521
Levenshtein Edit Distance [wrt Reference] ↓	*0.8685*	*0.8426*	*0.8124*	***0.8101***

Table 10: Expert-based Manual Evaluation.

	ROUGE-1	ROUGE-2	ROUGE-L	Fact-Core	Fact-Full	BERT-M1	BERT-M2	BLEURT
ROUGE-1	1.000							
ROUGE-2	0.636	1.000						
ROUGE-L	0.949	0.646	1.000					
Fact-Core	0.410	0.429	0.334	1.000				
Fact-Full	0.379	0.389	0.335	0.740	1.000			
BERT-M1	0.790	0.457	0.801	0.254	0.265	1.000		
BERT-M2	0.857	0.505	0.847	0.362	0.339	0.905	1.000	
BLEURT	0.790	0.429	0.787	0.269	0.247	0.784	0.859	1.000

Table 11: Pearson's correlation coefficients between the automatic evaluation metrics.

sensitive to prefixes in the training data (in the GS experiment we used 20 different section headers as prefixes in the training data). This has likely prevented the models from generalizing factual patterns across different sections, even though the same prefixes helped improve the performance on a token level according to the token-based evaluation measures (ROUGE-1 and BERTScore). On the other hand, guided summarization improved non-redundancy and lowered the factual hallucination rate to only 1% down from 4% for Model #2.

Our four models provided summaries with an average length of 9.76, 24.6, 19.77, and 21.45 tokens, respectively. To evaluate potential performance bias from summary length, we computed the correlation between the summary length and BLEURT as an example of automatic metric and the correlation between the summary length and Factual F1 as an example of manual metric. The Pearson correlation scores between BLEURT and the summary length of Model #1, #2, #3, and #4 are low (-0.200, -0.027, -0.147, and -0.173, respectively). The correlation between Manual Factual F1 an the summary length of the same models are higher with an inverse correlation of -0.511, -0.230, -0.268, and -0.409, respectively, which indicates that the models are prone to more errors in longer summaries.

5.3 Correlation between Evaluation Metrics

Table 11 shows Pearson correlations between the automatic evaluation metrics. BERTScore-M2, based on the DeBERTa model, was the embedding-based metric with the highest correlation with the n-gram metrics ROUGE-1, ROUGE-2, and ROUGE-L, and the embedding-based metrics BERTScore-M1 and BLEURT. However, ROUGE-1 and ROUGE-2 had higher correlation with Fact-Core and Fact-Full.

Table 12 presents the Pearson correlations between the automatic metrics and manual scores. The correlations with the manual scores show a different picture, with BLEURT standing out as the most correlated with manual fact counts and expert-based correctness assessments. BLEURT was also the most correlated metric with the Levenshtein distance, used here as an indicator for the level of effort required to correct the generated summaries.

The manual fact metrics were more correlated with the embedding-based metrics than ROUGE-1, ROUGE-2, and ROUGE-L, in contrast with the automatic fact extraction metrics. This could be explained, in part, by the lower factual coverage of automatic fact extraction vs. manual fact identification. This also highlights an important empirical insight, in that upstream upper bounds, such as the limited coverage of automatic and symbolic fact

Manual / Automatic	**Factual P**	**Factual R**	**Factual F1**	**Hallucination**	**Omission**	**Levenshtein**
	Correctness ↑			**Error Rate** ↓		
ROUGE-1	0.101	0.368	0.402	-0.074	-0.486	-0.326
ROUGE-2	0.086	0.167	0.202	-0.049	-0.237	-0.040
ROUGE-L	0.126	0.393	0.416	-0.073	-0.479	-0.296
Fact-Core	0.039	0.095	0.160	**-0.105**	-0.192	-0.030
Fact-Full	0.056	0.133	0.188	-0.094	-0.214	-0.122
BERTScore-M1	**0.132**	0.445	0.462	-0.078	-0.518	-0.317
BERTScore-M2	0.090	0.437	0.461	-0.080	-0.562	-0.366
BLEURT-20	0.113	**0.477**	**0.480**	-0.082	**-0.591**	**-0.486**

Table 12: Pearson's correlation coefficients between the automatic and manual scores.

extraction, can bias the correlation results against neural embeddings methods which, from their large pre-training, have inherently wider (implicit) coverage than symbolic fact extraction methods.

6 Conclusion

In this paper we conducted an empirical study of clinical note generation from doctor-patient encounters. This included creating a new dataset of 1,700 conversations and notes, evaluating several SOTA summarization models, and using multiple automated metrics and human judgements for the summarization models. Our findings show that pre-finetuning transformer models plays a key role in improving factual accuracy and summary fluency, and in reducing critical fact omissions, with guided summarization improving the precision of the summary facts and reducing hallucinations at the expense of factual recall.

The manual evaluation showed that the generated summaries reached a high fluency score (2.44 on a 0-3 scale) and relatively high factual F1 (0.76), but the best model still had a hallucination rate of 3% and missed 33% of the medical facts. Wider research efforts are needed to design methods and models that can reduce the omission and hallucination rates as well as efficient fact-based evaluation metrics to automatically assess the factual consistency of the generated summaries.

The key bottleneck limiting increased research from the NLP community on medical note generation has been the lack of generally available datasets to train and experiment on. This public release of the MTS-DIALOG dataset will enable wider research and faster research progress on this very important and impactful NLP task. Our comprehensive evaluation of multiple summarization models and evaluation metrics provides valuable baselines and references for comparison, and also shows where these systems and metrics stand on multiple dimensions of human evaluation.

Medical doctors today suffer from heavy documentation burden, which frequently causes physician dissatisfaction and burnout, and distracts doctors from being able to give their undivided attention to their patients. We look forward to a future where NLP research and the NLP community are able to provide doctors with tools for effective automated medical note generation, and enable them to return their focus to what they really love - providing great care for their patients.

Limitations

To address the lack of data and protect patient privacy, we relied on creating simulated doctor-patient conversations from de-identified clinical notes. Although we relied on trained annotators with medical background, those simulated conversations might still not reflect faithfully the actual language or structure of real doctor-patient conversations. For instance, the fraction of disfluencies and speech interruptions may occur at higher frequency than simulated in this dataset. Furthermore, it is often the case that doctor-patient conversations are automatically transcribed from speech to text, and the speech-to-text generation systems can produce errors in the output text. If real conversation benchmarks are made available in the future, they would allow further extrinsic validation of the proposed dataset and the first research insights. Many more summarization models can be tested on the task with the new dataset, and this study explored only some of them. Different, or larger language models could yield different results and lead to new or different insights. Although the introduced dataset is substantially bigger in size compared to other doctor-patient conversation datasets, its size is still limited in comparison with open-domain summarization datasets, which can limit the fine-tuning performance, especially for larger neural models. The dataset is limited in terms of covered diseases. The annotators ensured that the dataset is fully anonymized and balanced the synthetic conversations w.r.t. gender but the notes could still include real-world biases from the original clinical notes. The created dataset is intended for research purposes on automatic clinical note generation and should not be used for medical diagnosis or other health-related applications.

Ethics Statement

No protected health information were used in the creation of this dataset. Annotators were paid a fair hourly wage consistent with the practice of the state of hire.

References

Brian G Arndt, John W Beasley, Michelle D Watkinson, Jonathan L Temte, Wen-Jan Tuan, Christine A Sinsky, and Valerie J Gilchrist. 2017. Tethered to the EHR: Primary care physician workload assessment using EHR event log data and time-motion observations. In *Annals of Family Medicine*, volume 15(5), pages 419–426.

Asma Ben Abacha, Yassine Mrabet, Yuhao Zhang, Chaitanya Shivade, Curtis P. Langlotz, and Dina Demner-Fushman. 2021. Overview of the MEDIQA 2021 shared task on summarization in the medical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing, BioNLP@NAACL-HLT 2021, Online, June 11, 2021*, pages 74–85. Association for Computational Linguistics.

Jean Carletta. 2007. Unleashing the killer corpus: experiences in creating the multi-everything ami meeting corpus. *Language Resources and Evaluation*, 41:181–190.

Seppo Enarvi, Marilisa Amoia, Miguel Del-Agua Teba, Brian Delaney, Frank Diehl, Stefan Hahn, Kristina Harris, Liam McGrath, Yue Pan, Joel Pinto, Luca Rubini, Miguel Ruiz, Gagandeep Singh, Fabian Stemmer, Weiyi Sun, Paul Vozila, Thomas Lin, and Ranjani Ramamurthy. 2020. Generating medical reports from patient-doctor conversations using sequence-to-sequence models. In *Proceedings of the First Workshop on Natural Language Processing for Medical Conversations*, pages 22–30, Online. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryscinski, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir R. Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Trans. Assoc. Comput. Linguistics*, 9:391–409.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *CoRR*, abs/1911.12237.

Chih-Wen Goo and Yun-Nung Chen. 2018. Abstractive dialogue summarization with sentence-gated modeling optimized by dialogue acts. In *2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, December 18-21, 2018*, pages 735–742. IEEE.

Ben Goodrich, Vinay Rao, Mohammad Saleh, and Peter J. Liu. 2019. Assessing the factual accuracy of generated text. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Hardy, Shashi Narayan, and Andreas Vlachos. 2019. Highres: Highlight-based reference-less evaluation of summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3381–3392. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *CoRR*, abs/1506.03340.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation, INLG 2020, Dublin, Ireland, December 15-18, 2020*, pages 169–182. Association for Computational Linguistics.

George Hripcsak, David K Vawdrey, Matthew R Fred, and Susan B Bostwick. 2011. Use of electronic clinical documentation: time spent and team interactions. *Journal of the American Medical Informatics Association*, 18:112–7.

Adam L. Janin, Don Baron, Jane Edwards, Daniel P. W. Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, and Chuck Wooters. 2003. The icsi meeting corpus. *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, 1:I–I.

Anirudh Joshi, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2020. Dr. summarize: Global summarization of medical dialogue by exploiting local structures. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3755–3763. Association for Computational Linguistics.

Nazmul Kazi and Indika Kahanda. 2019. Automatically generating psychiatric case notes from digital transcripts of doctor-patient conversations. In *Proceedings of the 2nd Clinical Natural Language Processing*

Workshop, pages 140–148, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Tom Knoll, Francesco Moramarco, Alex Papadopoulos-Korfiatis, Rachel Young, Claudia Ruffini, Mark Perera, Christian Perstl, Ehud Reiter, Anya Belz, and Aleksandar Savkov. 2022. User-driven research of medical note generation software. *CoRR*, abs/2205.02549.

Kundan Krishna, Sopan Khosla, Jeffrey P. Bigham, and Zachary C. Lipton. 2021. Generating SOAP notes from doctor-patient conversations using modular summarization techniques. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4958–4972. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Manling Li, Lingyu Zhang, Heng Ji, and Richard J. Radke. 2019. Keep meeting summaries on topic: Abstractive multi-modal meeting summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2190–2196. Association for Computational Linguistics.

Xinzhu Lin, Xiahui He, Qin Chen, Huaixiao Tou, Zhongyu Wei, and Ting Chen. 2019. Enhancing dialogue symptom diagnosis with global attention and symptom graph. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5032–5041. Association for Computational Linguistics.

Wenge Liu, Jianheng Tang, Jinghui Qin, Lin Xu, Zhuguo Li, and Xiaodan Liang. 2020. Meddg: A large-scale medical consultation dataset for building medical dialogue system. *ArXiv*, abs/2010.07497.

Zhengyuan Liu, A. Ng, Sheldon Lee Shao Guang, AiTi Aw, and Nancy F. Chen. 2019. Topic-aware pointer-generator networks for summarizing spoken conversations. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

George Michalopoulos, Kyle Williams, Gagandeep Singh, and Thomas Lin. 2022. Medicalsum: A guided clinical abstractive summarization model for generating medical reports from patient-doctor conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, volume EMNLP 2022 of *Findings of ACL*, page 4741–4749. Association for Computational Linguistics.

Francesco Moramarco, Damir Juric, Aleksandar Savkov, Jack Flann, Maria Lehl, Kristian Boda, Tessa Grafen, Vitalii Zhelezniak, Sunir Gohil, Alex Papadopoulos Korfiatis, and Nils Hammerla. 2022a. Towards more patient friendly clinical notes through language models and ontologies. In *AMIA Annu Symp Proc*.

Francesco Moramarco, Alex Papadopoulos-Korfiatis, Mark Perera, Damir Juric, Jack Flann, Ehud Reiter, Anya Belz, and Aleksandar Savkov. 2022b. Human evaluation and correlation with automatic metrics in consultation note generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5739–5754. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *CoRR*, abs/1808.08745.

Alex Papadopoulos Korfiatis, Francesco Moramarco, Radmila Sarac, and Aleksandar Savkov. 2022. PriMock57: A dataset of primary care mock consultations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 588–598, Dublin, Ireland. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. BLEURT: learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7881–7892. Association for Computational Linguistics.

Jamin Shin, Hangyeol Yu, Hyeongdon Moon, Andrea Madotto, and Juneyoung Park. 2022. Dialogue summaries as dialogue states (ds2), template-guided summarization for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3824–3846. Association for Computational Linguistics.

Yan Song, Yuanhe Tian, Nan Wang, and Fei Xia. 2020. Summarizing medical conversations via identifying important utterances. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 717–729, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Brett R. South, Danielle L. Mowery, Ying Suo, Jianwei Leng, Óscar Ferrández, Stéphane M. Meystre, and Wendy W. Chapman. 2014. Evaluating the effects of machine pre-annotation and an interactive annotation interface on manual de-identification of clinical text. *J. Biomed. Informatics*, 50:162–172.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Wen-wai Yim and Meliha Yetisgen. 2021. Towards automating medical scribing : Clinic visit Dialogue2Note sentence alignment and snippet summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 10–20, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Longxiang Zhang, Renato Negrinho, Arindam Ghosh, Vasudevan Jagannathan, Hamid Reza Hassanzadeh, Thomas Schaaf, and Matthew R. Gormley. 2021. Leveraging pretrained models for automatic summarization of doctor-patient conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 3693–3712. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yuanzhe Zhang, Zhongtao Jiang, Tao Zhang, Shiwan Liu, Jiarun Cao, Kang Liu, Shengping Liu, and Jun Zhao. 2020. MIE: A medical information extractor towards medical dialogues. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6460–6469, Online. Association for Computational Linguistics.

Instruction Clarification Requests in Multimodal Collaborative Dialogue Games: Tasks, and an Analysis of the CoDraw Dataset

Brielen Madureira[1] and **David Schlangen**[1,2]
[1]Computational Linguistics, Department of Linguistics
University of Potsdam, Germany
[2]German Research Center for Artificial Intelligence (DFKI), Berlin, Germany
{madureiralasota, david.schlangen@uni-potsdam.de}

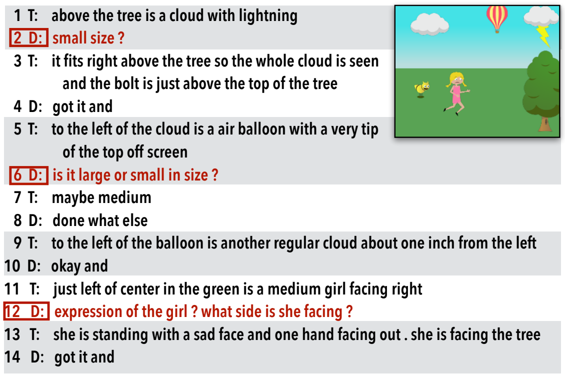
Abstract

In visual instruction-following dialogue games, players can engage in repair mechanisms in face of an ambiguous or underspecified instruction that cannot be fully mapped to actions in the world. In this work, we annotate Instruction Clarification Requests (iCRs) in CoDraw, an existing dataset of interactions in a multimodal collaborative dialogue game. We show that it contains lexically and semantically diverse iCRs being produced self-motivatedly by players deciding to clarify in order to solve the task successfully. With 8.8k iCRs found in 9.9k dialogues, CoDraw-iCR (v1) is a large spontaneous iCR corpus, making it a valuable resource for data-driven research on clarification in dialogue. We then formalise and provide baseline models for two tasks: Determining when to make an iCR and how to recognise them, in order to investigate to what extent these tasks are learnable from data.

1 Introduction

Somewhere in interstellar space are the Voyager Golden Records[1], which left Earth in spacecrafts in 1977 carrying a message about humanity to extraterrestrial civilizations. The committee in charge of designing the message, chaired by Carl Sagan, was careful to include symbolic instructions on how to play the records. But what if these instructions turn out to be incomprehensible to the aliens?

In human dialogue, Clarification Requests (CRs), such as those highlighted in Figure 1, are a common and indispensable mechanism to signal misunderstandings and to negotiate meaning, as recently stressed *e.g.* by Benotti and Blackburn (2017). This utterance-anaphoric conversational move can be realized with various forms, functions/readings and contents (Purver et al., 2003; Ginzburg, 2012) and can trigger responses that may or not be satisfactory (Rodríguez and Schlangen, 2004).



1 T: above the tree is a cloud with lightning
2 D: small size ?
3 T: it fits right above the tree so the whole cloud is seen and the bolt is just above the top of the tree
4 D: got it and
5 T: to the left of the cloud is a air balloon with a very tip of the top off screen
6 D: is it large or small in size ?
7 T: maybe medium
8 D: done what else
9 T: to the left of the balloon is another regular cloud about one inch from the left
10 D: okay and
11 T: just left of center in the green is a medium girl facing right
12 D: expression of the girl ? what side is she facing ?
13 T: she is standing with a sad face and one hand facing out . she is facing the tree
14 D: got it and

Figure 1: Instruction Clarification Requests identified in a portion of a CoDraw dialogue (ID 8906, CC BY-NC 4.0), with a scene from Zitnick and Parikh (2013).

In addition to the scientific motivation to comprehend CRs as a linguistic phenomenon, timely producing and understanding the vast range of CRs is also a desirable property for dialogue systems (Schlangen, 2004). This ability is especially relevant in scenarios where building common ground is necessary to act and collaboratively achieve a goal. Instructional interactions are a particular instance where an instruction follower (*IF*) often needs to ask for clarification in order to execute actions according to an instruction giver's (*IG*) instructions.

Instruction Clarification Requests (iCRs), as we will refer to them, are a type of CRs originating at Clark (1996)'s 4th level of communication, the level of uptake (Schlöder and Fernández, 2014). They are elicited when an instruction utterance is generally understood (*e.g.* acoustically, syntactically, semantically) but some underspecification or ambiguity prevents the *IF* to carry out an action with enough certainty, as shown in Figure 1.

Learning clarification mechanisms from data is still an understudied research problem (Benotti and Blackburn, 2021). We envision the following desiderata for a dataset suitable for data-driven research on iCRs:

---

[1] https://voyager.jpl.nasa.gov/golden-record//

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2303–2319
May 2-6, 2023 ©2023 Association for Computational Linguistics

- ▷ **Naturalness**: iCRs should occur by the spontaneous decision process of the *IF* in real interaction while trying to act and solve a task, ideally not being induced by external incentives in the data collection and also not synthetically generated.
- ▷ **Specificity**: the annotation should pin down iCRs as a single category, not subsumed within other CRs and dialogue acts.
- ▷ **Frequency**: relative and absolute occurrence of iCRs should be large enough for data-driven methods and statistical purposes.
- ▷ **Diversity**: iCRs should occur with various forms and content, being grounded in the game actions and parameters.
- ▷ **Relevance**: iCRs should be pertinent for players to decide on actions and solve the task successfully.
- ▷ **Regularity**: iCRs should emerge from underlying strategies of the players and not be the result of random or idiosyncratic behaviour.

Our research questions are: i) Can *IF* dialogue models trained on data learn to recognise when they would profit from receiving more information in order to execute an action, and thus generate an iCR? ii) Can *IG* dialogue models trained on data learn to recognise when the *IF* is making an iCR and respond to it?

In this work, our contribution to begin addressing these questions is threefold. We (a) perform annotation of naturally occurring iCRs in a collaborative and multimodal dialogue game, namely the CoDraw dataset (Kim et al., 2019), showing that it is a valuable resource for data-driven research on clarification in dialogue; (b) analyse the corpus and provide insights relating iCRs to the game dynamics; and (c) discuss two subtasks and models that can be explored with CoDraw-iCR (v1) and may serve as components of *IF* and *IG* dialogue models capable of handling iCRs.

2 Related Literature

It is a common practice to map CRs to the level of communication (Clark, 1996; Allwood, 2000) where the misunderstanding occurs (Gabsdil, 2003; Schlangen, 2004; Rodríguez and Schlangen, 2004; Rieser and Moore, 2005; Rieser et al., 2005; Bohus and Rudnicky, 2005; Benotti, 2009; Koulouri and Lauria, 2009; Benotti and Blackburn, 2021). When ASR used to be a bottleneck for dialogue processing, several works focused on CRs elicited by problems at levels 2 and 3 – perception and understanding (Healey et al., 2003; Schlangen and Fernández, 2007a,b; Stoyanchev et al., 2013, 2014, *inter alia*). Comparatively less research exists focusing on CRs at level 4, namely intention, uptake or task-level clarifications (Benotti, 2009; Schlöder and Fernández, 2014). We thus contribute to filling this gap, building upon the existing literature we now turn to discuss in more detail.

Schlöder and Fernández (2015) perform a corpus-based study splitting level 4 CRs into two types of intention-related conversational problems: recognition and adoption. Instruction-following dialogues, where utterances are intertwined with actions, is one setting where level 4 CRs play a fundamental role in negotiating meaning. Benotti and Blackburn (2017) discuss the relation between instruction, CRs and contexts in such settings and how conversational implicatures are a rich source of CRs. Task-level reformulations, a clarification strategy where the initiator rephrases an utterance with respect to its effects on the task, are typically used to confirm more complex actions in instruction giving dialogues (Gabsdil, 2003) and happen very frequently (Benotti, 2009). Multimodality, *e.g.* gestures, also play a role in instruction-following CRs (Ginzburg and Luecking, 2021).

Benotti (2009) proposes using planning to infer and generate the task-level clarification potential of instructions and identify level 4 CRs in one dialogue of a corpus of 15 instruction giving dialogues. Benotti and Blackburn (2021) analyse the same corpus and identify six characteristics that may account for the larger proportion of level 4 CRs found in it: task-oriented dialogues, asymmetry in dialogue participant roles (*IF* and *IG*), immediate world validation by the informational or physical actions, shared view and consequent verification of the actions, long dialogues that enable more shared background, and irreversible actions that require more certainty.

Other corpus studies exist in small datasets. Rodríguez and Schlangen (2004) find that 22.17% of the CRs are level 4 CRs in an instruction-following setting. Similarly, Gervits et al. (2021) collect and annotate 22 dialogues with a human-controlled virtual robot that followed high-level or low-level instructions. They propose a very detailed annotation schema for the content of CRs, but there is no clear distinction of level 4 CRs.

A larger dialogue game dataset, the Minecraft Dialogue Corpus (Narayan-Chen et al., 2019) with 509 games, has been annotated with CRs. Lambert et al. (2019) annotate the *IF* utterances with eight dialogue acts, one of which, `clarification questions`, comprises requests for clarification to a given instruction or statement (26.36% of all utterances). Shi et al. (2022) perform a similar annotation with a category `instruction-level questions` to request clarification for a previous instruction that was not clear or ambiguous (18.64%).

The TEACh dataset (Padmakumar et al., 2022) contains 3k dialogues annotated with dialogue acts (Gella et al., 2022), of which the 675 `RequestOtherInfo` spans under the `Instruction` category relate to iCRs.

Kiseleva et al. (2021) extend the Minecraft Dialogue Corpus with 47 games containing 126 CRs for an interactive agent building challenge, but concentrate on the task of modelling a "silent *IF*" that cannot ask questions. The second edition of their challenge, which happened recently (Kiseleva et al., 2022; Mohanty et al., 2022), focuses on when the *IF* should ask for clarification and what it should ask about, similar to Aliannejadi et al. (2021). The dataset for the second challenge is not collected through real, synchronous interaction. Instead, one player builds a structure and generates instructions *a posteriori*, and, in a separate step, another player follows these instructions, deciding whether to make a CR. Similarly, Aliannejadi et al. (2021) collects a large dataset of CRs to user requests, augmented synthetically, in a multiple-step process without interaction. Another large-scale dataset with 53k task-relevant questions and answers about an instruction was constructed Gao et al. (2022). However, the data is created by an annotator that does not have to act, but only watches execution videos, asking a question they think would be helpful and then answering their own question.

Although these strategies facilitate data collection, they abstract away the decision-making and repair processes that emerge when humans collaborate to solve a task jointly, which are present in CoDraw. Our work and the existing literature converge in addressing CRs for ambiguous instructions, but CoDraw-iCR (v1) maintains the interactive aspect of *sequential* rounds and the spontaneous initiative of *IF* to ask. It is large in absolute number of iCRs and dialogues, with short games that have a relatively constrained action space. Moreover, our annotation pins down iCRs among other types of CRs.

A dataset that can be further explored for iCRs is Thomason et al. (2020). It instantiates a navigation task where the *IF* gets an ambiguous or underspecified command about where to navigate to, and can ask questions to an oracle during the trajectory.

In HRI, following commands is a central task. Koulouri and Lauria (2009) investigate miscommunication management mechanisms in robots performing collaborative tasks, in which task-level reformulations is a challenging type of CR that requires identification of the effects of all possible executions of an instruction. Deits et al. (2013) evaluate various clarification question strategies for robots that receive instructions with an ambiguous phrase. Marge and Rudnicky (2015) examine recovery strategies in situated grounding problems, when an agent has to deal with requests containing referential ambiguity or that are impossible to execute. Interestingly, Jackson and Williams (2018) and Jackson and Williams (2019) raise awareness to the fact that merely posing a CR can already imply willingness to follow a command, which is undesirable in morally delicate situations.

Other tangent research areas study clarification edits to solve underspecified phrases in instructional texts (Roth et al., 2022) and clarification responses in community forum questions or search queries (Braslavski et al., 2017; Rao and Daumé III, 2018; Aliannejadi et al., 2019; Kumar and Black, 2020; Hu et al., 2020; Majumder et al., 2021), scenarios with only minimal or no interaction.

Tasks. Deciding when to initiate a CR in various contexts is a task classically discussed in the CR literature (Rieser and Lemon, 2006; Stoyanchev et al., 2012, 2013; Narayan-Chen et al., 2019; Aliannejadi et al., 2021; Shi et al., 2022; Kiseleva et al., 2022, *inter alia*). Fewer works exist specifically about detecting if a CR was made. Identification of CRs in corpora carry out a similar task, although this is not done from the perspective of an agent knowing that it needs to respond to the CR, of which De Boni and Manandhar (2003) is an example. More generally, this task can be subsumed by dialogue act classification, as in, for instance, Gella et al. (2022).

3 Motivation and Problem Statement

CRs occur naturally in human-human interaction and thus also in visual dialogue games. Neural network-based dialogue models trained at such datasets need to properly handle this phenomenon, which comprises various component tasks for identifying, interpreting, generating and responding to CRs. In this section, we formalise the setting and two of these tasks.

3.1 Formalisation: Instruction-Following Dialogue Games

A visual instruction-following dialogue game can be formalised as a tuple $G = (P, S, R, M)$ representing a goal-oriented interaction between players P (an instruction giver *IG* and an instruction follower *IF*). *IG* sees a scene S, hidden to *IF*, and instructs *IF* on how to reconstruct it. They exchange a sequence R of n rounds $r_i = (g_i, a_i, f_i)$ comprised of two utterances (g_i, f_i), from *IG* and *IF*, respectively, and of actions a_i that incrementally create partial reconstructions s_i of S. R is initialised as an empty set and, at each round, it is extended with g_i, a_i and f_i, in that order. The final state of a completed game contains all filled rounds. A scene similarity metric M computes how close the reconstructions are to the original image at each round, and the goal is to maximize similarity of the final reconstruction $M(S, s_n)$.

The dialogue acts by the *IF* include acknowledgements and clarification requests, whereas the dialogue acts by the *IG* include instructions and responses to clarifications. Two variations are possible: the state s_i can be accessible for the *IG* or not. The incremental scenes can be regarded either as the common ground between players (if both can see it) or as what the *IF* considers to be their common ground (when it is private), akin to what is proposed by Mitsuda et al. (2022).

Following Clark (1996), we assume that a pair of equally competent players, committed to the game's goal of maximizing $M(S, s_n)$, seek to minimize joint effort. It is acceptable for the *IG* to produce an underspecified instruction if producing a fully specified instruction would cost more than answering an iCR. Instruction CRs require an extra effort by the *IF*, so they should occur when repair is necessary and the cost of asking is lower than the potential information gain.

3.2 Tasks

We propose to use CoDraw-iCR (v1) to advance research in iCRs by modelling two CRs subtasks in an instruction-following dialogue game grounded in a visual modality. Both subtasks can be regarded as a binary decision step happening right before each player's next utterance generation.

Task 1: Ask iCR? From the *IF*'s perspective as the CR initiator, decide when to initiate a CR. More specifically, after each *IG* utterance, given the dialogue context $D_{0:(i-1)}$ (that is, all previous utterances), the current utterance g_i by IG, and the current state[2] of the scene s_i, the *IF* must decide on the type of their utterance f_i, namely whether to consider the action completed and signal willingness to receive further instructions (*e.g.*, produce something like "OK"), or to ask for clarification on some aspect of a previous instruction. That is, this formulation of the task focuses on the dialogue act to perform, abstracting away from the concrete realisation. It deals with the problem of automatically determining what is a good instruction and what is not, on its context. This task relates to slot filling in the sense that an instruction containing all the needed parameters for the mentioned objects should not require clarification.

Task 2: Was this an iCR? From the *IG*'s perspective as the CR recipient, identify whether an iCR has been made. At each round i, given the dialogue context $D_{0:i}$ (in which the last utterance, f_i, is possibly an iCR) and the original scene S, the *IG* must decide whether to give further instructions or to (also) respond to an iCR.

4 Data and Annotation

CoDraw (Kim et al., 2019) is a collaborative instruction-following dialogue game, in which a "teller" (in our terminology, the *IG*) observes a clipart scene and instructs a "drawer" (*IF*), who has no access to it, on how to reconstruct it, *i.e.* place cliparts in a canvas with the correct size, direction and position. The corresponding crowdsourced dataset contains 9,993 dialogues in English and has been released under a CC BY-NC 4.0 license. This dataset instantiates the formalisation proposed in

---

[2] Under the assumption that the *IF* has manipulated the scene in response to *IG* already. For CoDraw, the exact point when the *IF* types the message has not been preserved.

Section 3, but adds an additional signal: The teller is allowed to peek at the drawer's canvas once during the game whenever they want, *i.e.* the teller can get access to s_i and thus judge how it differs from S. Players exchange messages of up to 140 characters through a chat interface and must alternate turns. We will use round to refer to a pair of consecutive utterances by teller and drawer with the corresponding actions. The drawer's performance is evaluated with a scene similarity score that ranges from 0 to 5, where 5 is a perfect match. Table 1 summarizes quantitative aspects of the dataset.

	train	**val**	**test**
dialogues	7,989	1,002	1,002
with peek	7,315	923	913
avr. final score	4.20	4.19	4.17
before peek	3.97	3.95	3.96
avr. rounds/dialogue	7.76	7.69	7.70
avr. utterance len teller	14.36	14.48	14.31
avr. utterance len drawer	2.58	2.67	2.58
vocab size *IG*		4,506	
vocab size *IF*		2,200	

Table 1: Descriptive statistics: CoDraw dataset.

Each game is about a different abstract scene[3] composed of between 6 and 17 out of a set of 58 clipart types (Zitnick and Parikh, 2013; Zitnick et al., 2013), among which the boy and the girl can have 5 facial expressions and 7 body poses, so the resulting clipart set contains 126 elements and the default background. Multiple types of trees, hats, clouds, glasses and balls can introduce the need for ambiguity resolution in the games. As the individual components can be placed freely, the space of possible resulting scene images is practically unlimited in size.

In the baseline models proposed in the original paper, the authors introduce a simplifying assumption which removes the drawer's utterances from the dialogue history (they call this condition the *silent drawer*). The authors leave the tasks of identifying when a CR is necessary and of generating it for future work. Subsequent works with this dataset have focused on text-to-image generation (El-Nouby et al., 2019; Matsumori et al., 2021; Zhang et al., 2021; Lee et al., 2021; Liu et al., 2020; Fu et al., 2020) but, to the best of our knowledge, no other work has examined CRs in CoDraw. We thus take up this idea to bring back the dialogue modality to this dialogue game.

Identification of Instruction CRs. We observe that a good portion of the drawer's utterances belongs to one of two dialogue act types: *acknowledgements*, signaling that the teller may proceed with the next instruction, and *clarification requests*, initiating repair on aspects necessary to solve the task. We thus consider CoDraw to be a potentially interesting source of iCRs.

The first step we take is identifying instruction-level CRs in this dataset. To achieve that, we perform a binary decision over the drawer's utterances. For our purposes, an utterance is an iCR if the following assertion is likely true:

> "*This utterance indicates that the drawer is requesting further information about one or more instruction(s) previously given by the teller in order to perform an action accordingly, likely because part of the instruction was underspecified, ambiguous or not clear.*"

To reduce the annotation workload, we annotate utterance *types*; forms that occur only once (88.97% of the types) are presented with a one-utterance context window around it. All occurrences of each of the other utterance forms are collapsed into a single datum, presented to the annotators without context.

5 Corpus Analysis

In this section, we present an analysis of iCRs in the CoDraw dataset and their relation to the game dynamics, establishing connections to the items in our desiderata and showing that CoDraw-iCRs (v1) is a promising resource to study the phenomenon and to model dialogue agents that learn what to do in face of unclear instructions, complementing existing initiatives.[4]

5.1 Descriptive Statistics

The 13,727 *IF*'s utterance types have been annotated by two annotators, with a Cohen's κ (Cohen, 1960) of 0.92. Table 2 presents the main descriptive statistics of the annotated corpus.[5] 8,807 (11.36%) of all drawer's utterances in CoDraw are iCRs. 59.45% of the dialogues contain no iCRs. For the purpose of analysis, we also compute numbers relative to the subset of dialogues that contain at least

---

[3] http://optimus.cc.gatech.edu/clipart/

[4] The dataset is available for the community upon request.

[5] In this paper, for the around 3.6% of the utterances with disagreements, we opt for the second annotator's labels, who had more training.

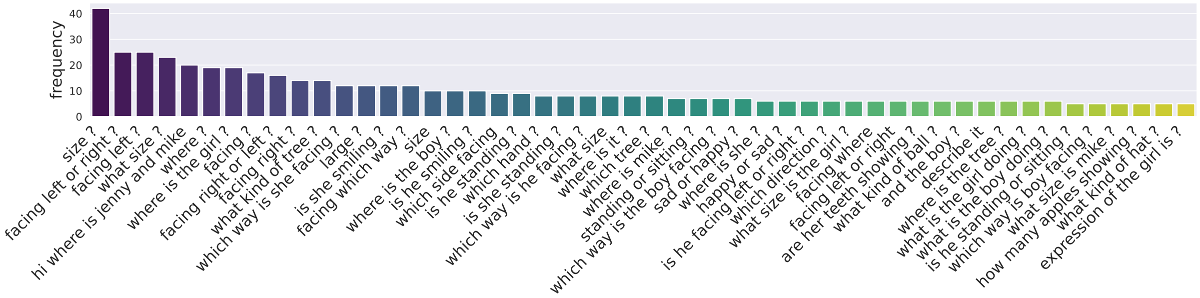


Figure 2: 50 most frequent Instruction CRs in the CoDraw dataset ordered by rank.

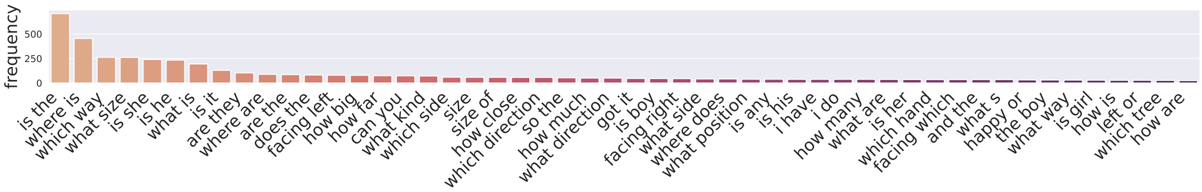


Figure 3: 50 most frequent iCRs initial bigrams in the CoDraw dataset.

one iCR; the idea here is that this excludes players who may not have been willing to use the opportunity to ask iCRs. In this subset, the percentage of iCRs is 24.36%. We also separate out numbers computed from the dialogues up the "peek" action described above, as from that move on, the state of the common ground changes.

	all	w/ iCRs	until peek
dialogues	9,993	4,052	-
rounds	77,502	36,149	61,829
iCR utterances	8,807	8,807	7,803
% iCR utterances	11.36	24.36	12.62
mean iCRs/dialogue	0.88	2.17	0.78
std iCRs/dialogue	1.53	1.73	1.36

Table 2: Descriptive statistics: Annotation.

Figure 2 presents the most frequent iCR utterance types, ordered by rank. 7,260 (94.13% of the types) are *hapax legomena*. Types occupying the highest ranks relate to size, position and orientation, which directly map to the possible actions on cliparts, and to disambiguation of *e.g.* facial expression and body pose. Few types occur more than 5 times, which is evidence that the dataset contains a rich diversity of iCR surface forms. Figure 3 aggregates iCRs by initial bigrams, after removing punctuation and initial *ok* and *okay* tokens (which realise a different dialogue act). Common iCR forms are polar questions and wh-questions also related to the main actions (placement, resize, flip, disambiguation).

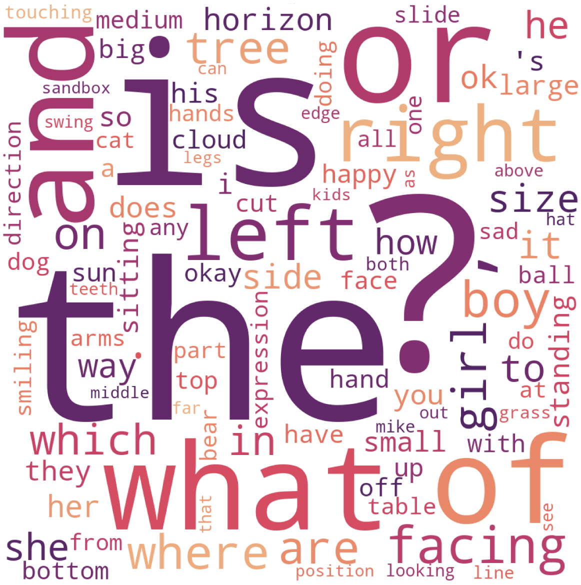
The drawer's vocabulary contains 2,200 token types, out of which 1,468 occur in iCRs. Figure 4 shows an overview of the 100 most common tokens. The frequent iCR vocabulary contains many nouns relating to cliparts (slide, table, bear, dog), in particular those that refer to nouns involving ambiguity (boy, girl, cloud, tree, ball). Question words occur frequently (what, how, where, which) as well as words about object placement (horizon, facing, size, top, touching, edge). Non-iCR utterances commonly contain words related to the task (scenery, picture, image, check, next), greetings and thanks, and acknowledgement words (ok, ready, done).

5.2 Relations to Game Dynamics

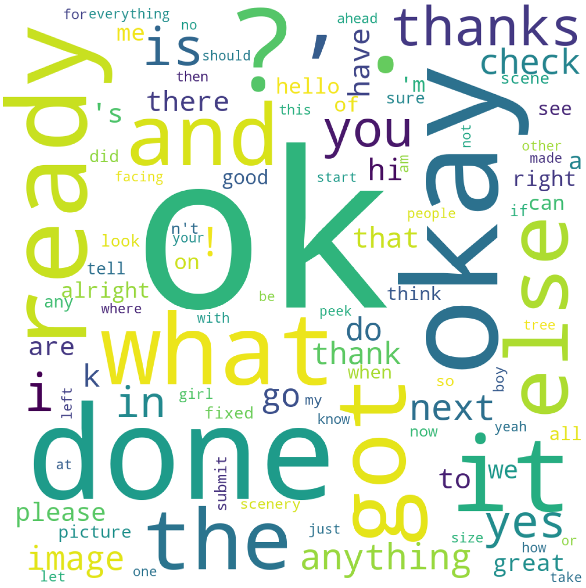
We now turn to examining how the occurrence of iCRs relate to the overall game dynamics.

To analyse CRs, three positions in a dialogue are particularly relevant: the source utterance in which the communication problem occurs, the CR utterance where repair is initiated, and the response utterance where the problem should ideally be dealt with. Since the dialogue is organized into a sequence of rounds with pairs of utterances (g_i, f_i), if an iCR occurs at round i, then f_i is an iCR, g_i is the likely source utterance, and g_{i+1} is possibly the response utterance. In Figure 1, turns 1, 5 and 11 are sources, 2, 6 and 12 are iCRs and 3, 7 and 13 are responses. However, these events do not necessarily occur in immediate sequence.

Here, we investigate how the game dynamics change at two positions: iCR rounds and rounds immediately following an iCR. We look at the mean number of actions per round and the difference in



(a) iCR utterances



(b) other utterances

Figure 4: Most common tokens weighted by frequency.

the score metric with respect to the previous state, as shown in Table 3. On average, more actions occur at iCR rounds than at non-CR rounds. The difference is even larger in post-iCR rounds, where necessary edits can be occurring. iCR rounds also cause an average higher improvement in the metric than other rounds and the same occurs for rounds after iCRs in dialogues containing iCRs.

To conclude this section, we refer back to our desiderata. The **naturalness** of iCRs is a consequence of the data being produced by synchronous human-human interaction in a setting that does not directly induce players to ask for clarification; indeed, almost 60% of the games do not contain iCRs, which we take to be evidence that they are a result of the private decision making of the *IF* and not due to them following instructions on which dialogue acts to produce. **Specificity** is guaranteed by the annotation process which had a definition to distinguish iCRs from other utterances. In terms of **frequency**, iCRs are a common phenomenon in CoDraw-iCR (v1), which contains 8,807 (11.36%) iCR utterances, a sample larger than existing annotated datasets. We have gathered evidence that **diversity** is present, given that iCRs occur in various forms and exhibit lexical and semantic variety on content related to the game. When it comes to **relevance** to the task, we have shown that there are statistically significant differences in number of actions and score differences at turns realising and following iCRs, which is a sign that agents need to process iCRs in order to act accordingly throughout the game. **Regularity** is addressed in the experiments in the next section.

	all	**w/ iCRs**
mean actions per round		
iCR rounds	1.72	1.72
not iCR rounds	1.64*	1.62*
post-iCR rounds	2.11	2.11
not post-iCR rounds	1.59*	1.50*
mean score diff		
iCR rounds	0.59	0.59
not iCR rounds	0.53*	0.43*
post-iCR rounds	0.53	0.53
not post-iCR rounds	0.54	0.44*

Table 3: Round dynamics. * means the difference in relation to the value at the row above is statistically significant at $\alpha = 0.01$ using a permutation test.

6 Models and Experiments

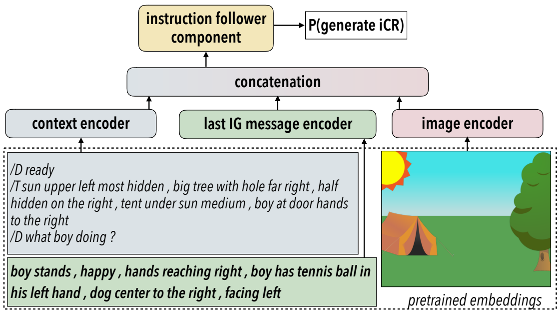
In this section, we present the models for the two tasks discussed in Section 3.2 as well as the evaluation metrics. Both are binary classification tasks using regression to predict the probability of the positive label (iCR) on imbalanced datasets, whose distribution is shown in Table 4.

	train	**val**	**test**
datapoints	62,067	7,714	7,721
% iCR	11.30	11.92	11.28
% not iCR	88.69	88.07	88.71

Table 4: Distribution of labels.

6.1 Models

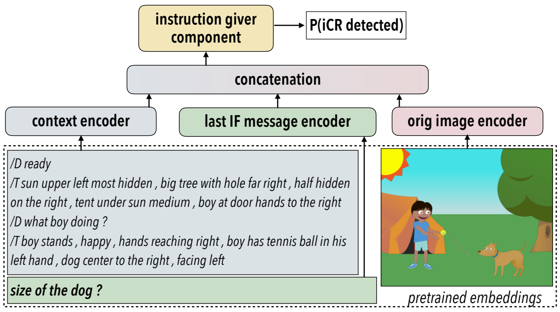
We model the two prediction subtasks as a function $f : (s, c, u) \mapsto P(l = 1)$ where s is the representation of the scene, c is the representation of the dialogue context, u is the representation of the last utterance and l is the label. This function is approximated with a neural network that takes each input embedding, encodes them, and maps them to a concatenated representation which is fed into a two-layer classifier that outputs the probability of the positive label by applying the sigmoid function to the logit output, as illustrated in Figure 5.[6]



/D ready
/T sun upper left most hidden , big tree with hole far right , half hidden on the right , tent under sun medium , boy at door hands to the right
/D what boy doing ?

boy stands , happy , hands reaching right , boy has tennis ball in his left hand , dog center to the right , facing left

(a) Task 1: Ask iCR?



/D ready
/T sun upper left most hidden , big tree with hole far right , half hidden on the right , tent under sun medium , boy at door hands to the right
/D what boy doing ?
/T boy stands , happy , hands reaching right , boy has tennis ball in his left hand , dog center to the right , facing left

size of the dog ?

(b) Task 2: Was this an iCR?

Figure 5: Illustration of the classifier architecture, with an example dialogue from CoDraw (ID 3454).

6.2 Evaluation

Although the area under the ROC curve is a standard evaluation metric for binary classification, it can be deceptive in imbalanced datasets due to the interpretation of specificity, in which case Precision-Recall curves are more suitable (Saito and Rehmsmeier, 2015). The Average Precision (AP) summarizes this curve into one metric that ranges from 0 to 1, where 1 is the best performance, and the theoretical random is the fraction of positive labels. To facilitate comparison to existing literature, we also report macro-average F1 Score.

As trivial baselines, we perform logistic regression on basic features of the utterances and on the input representation vectors. For Task 1, the features are the length of the last teller's utterance and its boolean bag-of-words representation. For Task 2, we use the length of the last drawer's utterance and a binary variable indicating whether a content word occurs in it. The list of content words was extracted manually from a sample of dialogues.

6.3 Embeddings

The pretrained embeddings for texts are generated with SentenceTransformers (Reimers and Gurevych, 2019) and for images with ResNet101 (He et al., 2016). In order to probe whether the pretrained sentence encoders minimally capture the necessary information for our task, we use the dialogue context representation at the turn before the peek action to predict whether iCRs occurred in the dialogue so far. Using a logistic regression model on dialogues that contain a peek turn, we achieve AP= 0.91 and macro F1 Score= 0.86 in the validation set. This provides evidence that, despite they having been optimized for other tasks, the occurrence of iCRs is, to some extent, encoded in the representations.

7 Results

Table 5 presents the main results of our models on the two tasks. The feature-based baselines provide some gain over the random performance for Task 1, and a considerable improvement for Task 2. The logistic regression baseline is enough to produce good results for Task 2, whereas Task 1 remains very challenging even for the neural network model.

		Task 1: *IF*		**Task 2: *IG***	
		AP	mF1	AP	mF1
random	val	.117	.489	.117	.489
	test	.113	.503	.113	.503
features	val	.206	.531	.687	.858
	test	.195	.518	.687	.855
log-reg	val	.324	.587	.984	.962
	test	.287	.576	.978	.961
model	val	.399	.662	.991	.969
	test	**.347**	**.645**	**.988**	**.968**

Table 5: Main results. Average Precision and macro-average F1 Score on the validation and test sets.

---

[6] Details about the implementation, setup and experiments are in the Appendix and the code is available at https://github.com/briemadu/codraw-icr-v1/.

Ablation. We remove each component of the input to the neural network model in order to understand what information is more relevant for this task. Table 6 shows the differences with respect to the performance in the validation set.

The image representation does not seem to be fully exploited by the model. While in Task 2 the image is expected to be superfluous to detect the dialogue act, it should play a role for Task 1, as it imposes constraints on possible actions. It is possible that the off-the-shelf pretrained model is not adequate to encode cliparts and further investigation with other models and fine-tuning is required.

The last message is the most relevant signal for Task 2, as expected, given that it is the iCR being classified. Without it, the task is almost equivalent to Task 1 and the performance is indeed similar. Interestingly, the most relevant signal for Task 1 is the context and not the last utterance, which is evidence that the model fails to distinguish well which instructions require an iCR. To further investigate this, we remove the teller's utterances and the drawer's utterances from the context embeddings. While removing the teller's utterances causes little change, removing the drawer's utterances is almost as detrimental as removing the whole context. We thus conclude that the model is likely exploring patterns in the drawer's behavior to make decisions.

	Task 1: ***IF***		**Task 2:** ***IG***	
	AP	mF1	AP	mF1
no image	-.032	-.012	.001	.005
no message	-.050	-.021	-.652	-.328
no context	-.109	-.054	.001	.007
context w/o teller	-.001	.000	-.001	-.000
context w/o drawer	-.087	-.054	-.000	.007

Table 6: Results of ablation in the input components. Differences in relation to the main result in the val set.

8 Discussion

Our findings are aligned with the recent conclusions by Aliannejadi et al. (2021) and Shi et al. (2022) that the task of predicting when a CR should be made is rather difficult with data-driven models. Techniques to deal with the class imbalance (downsampling, upsampling and varying the cost-sensitive loss function) and variations of the models (*e.g.* Transformer-based architectures) so far led us to similar results. On the other hand, the task of identifying iCR utterances is uncomplicated even for a simpler logistic regression model.

The results reached by our model in Task 1 do not quite allow us to see desideratum **regularity** as satisfied at this point, but we are confident that there is much room for interesting further research with this dataset. On their own, these tasks model an overhearer that predicts what the agent should do. What is of interest in reality is having them integrated as subcomponents, implicitly or explicitly, in the models that also make the instruction-giving/following decisions, because these capabilities are not detached in the agents *de facto*. We expect that the decision to ask for clarification should emerge more easily in representations of models that are also making actions.

The fact that the drawer's utterances seem to be informative in the dialogue representations for the task speaks against the "silent drawer assumption" int he original models (Kim et al., 2019). Removing the drawer's utterances from the dialogue likely cause loss of relevant dialogue phenomena that is pertinent to the game.

9 Conclusion

We have shown that CoDraw-iCR (v1), the CoDraw dataset augmented with our iCR annotation, is a valuable resource for investigating instruction-level CRs at scale. Through the corpus analysis, we have also concluded that iCR turns and post-iCR turns imply different game dynamics, which is relevant for models trained to play this game successfully. Therefore, in order to succeed in this type of task, agents need to know how to handle iCRs, as they influence not only the dialogue acts but also the game moves.

Our models perform well on detecting iCRs and lay the groundwork for further research on predicting when an iCR should be made. The research roadmap is to integrate iCRs into the full IF agent, so that the decision to ask for clarification is learnt together with the actions in the game.

The second annotation phase will provide fine-grained categories of iCRs' form and content and ground them to the game objects, opening the possibility to explore other tasks like generation.

10 Limitations

In this section, we discuss some limitations that we inherit from the CoDraw dataset, and then some limitations of our task setup and baseline model.

CoDraw is a simplified but representative instance of instruction giving/following dialogue games and we show that iCRs are frequent and play an important role in it. Since modelling CRs is still an open problem, using abstract scenes is a reasonable strategy to simplify the underlying task while still giving room for iCRs to occur. Limitations are inherent to data collections in controlled environments. We aim for our annotations to add to other recent efforts, which are limited in other ways. CoDraw-iCR (v1) thus aims to move one step forward towards modelling iCRs, but general conclusions depend on various resources and further collaborative efforts in our field.

Actions were not irreversible in CoDraw games. The introduction of the peek action for the teller can be an incentive both for the teller to not give exhaustive instructions and for the drawer to build only an approximation, knowing it could be refined after the peek. We have no access to what the performance would have been if they could not make CRs at all.

Meta-data about crowdworker ID is not available.[7] Because of that, we cannot investigate the effects of individual CR strategies by players. Players that play multiple games get to know what to expect of the game and should both have more practice in identifying underspecified instructions that require repair and be able to make better guesses about the cliparts. Experienced tellers probably anticipate common problems and adapt their instructions to avoid them (*e.g.* they know that multiple cliparts of trees exist and would likely describe it in their instruction, avoiding unnecessary communication problems). Besides, we cannot draw conclusions on whether dialogues without iCRs indeed did not require repair or some players were personally less inclined to make the effort to ask for clarification.

Although CRs annotation should take into account the full context (Benotti and Blackburn, 2021), the decision to annotate utterance types instead of full dialogues, as discussed in Section 4, is due to the limited resources given the size of the dataset and to the nature of the game setting. We avoided the need to go over multiple non-iCR utterances that occur very often. The plan for the second step of the annotation is to provide fine-grained annotation for each identified iCR within its own context.

Our models do not take into account the gallery of cliparts available to the drawer, which is informative (as it limits the choices of cliparts per game) and could be part of the input. Preliminary experiments did not lead to better results. Building a suitable representation of the gallery is left for future research.

Acknowledgements

We are thankful for the anonymous reviewers for their feedback. We thank our student assistants, Sebastiano Gigliobianco and Sophia Rauh, for performing the annotation and Philipp Sadler for generating the step-by-step scenes.

References

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2021. Building and evaluating open-domain dialogue corpora with clarifying questions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4473–4484, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval*, pages 475–484.

Jens Allwood. 2000. An activity-based approach to pragmatics. *Gothenburg Papers in Theorecial Linguistics*, 76.

Emily M. Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Luciana Benotti. 2009. Clarification potential of instructions. In *Proceedings of the SIGDIAL 2009 Conference*, pages 196–205, London, UK. Association for Computational Linguistics.

Luciana Benotti and Patrick Blackburn. 2017. Modeling the clarification potential of instructions: Predicting clarification requests and other reactions. *Computer Speech & Language*, 45:536–551.

Luciana Benotti and Patrick Blackburn. 2021. A recipe for annotating grounded clarifications. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4065–4077, Online. Association for Computational Linguistics.

---

[7] Personal communication with the authors.

Dan Bohus and Alexander I. Rudnicky. 2005. Sorry and I didn't catch that! - an investigation of non-understanding errors and recovery strategies. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 128–143, Lisbon, Portugal. Special Interest Group on Discourse and Dialogue (SIGdial).

Pavel Braslavski, Denis Savenkov, Eugene Agichtein, and Alina Dubatovka. 2017. What do you mean exactly? analyzing clarification questions in cqa. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, page 345–348, New York, NY, USA. Association for Computing Machinery.

Herbert H Clark. 1996. *Using language*. Cambridge university press.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Marco De Boni and Suresh Manandhar. 2003. An analysis of clarification dialogue for question answering. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 48–55.

Robin Deits, Stefanie Tellex, Pratiksha Thaker, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. 2013. Clarifying commands with information-theoretic human-robot dialog. *J. Hum.-Robot Interact.*, 2(2):58–79.

Alaaeldin El-Nouby, Shikhar Sharma, Hannes Schulz, Devon Hjelm, Layla El Asri, Samira Ebrahimi Kahou, Yoshua Bengio, and Graham W.Taylor. 2019. Tell, draw, and repeat: Generating and modifying images based on continual linguistic instruction. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10303–10311.

Tsu-Jui Fu, Xin Wang, Scott T. Grafton, Miguel P. Eckstein, and William Yang Wang. 2020. Iterative language-based image editing via self-supervised counterfactual reasoning. In *EMNLP*.

Malte Gabsdil. 2003. Clarification in spoken dialogue systems. In *Proceedings of the 2003 AAAI Spring Symposium. Workshop on Natural Language Generation in Spoken and Written Dialogue*, pages 28–35.

Xiaofeng Gao, Qiaozi Gao, Ran Gong, Kaixiang Lin, Govind Thattai, and Gaurav S Sukhatme. 2022. Dialfred: Dialogue-enabled agents for embodied instruction following. *IEEE Robotics and Automation Letters*, 7(4):10049–10056.

Spandana Gella, Aishwarya Padmakumar, Patrick Lange, and Dilek Hakkani-Tur. 2022. Dialog acts for task driven embodied agents. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 111–123, Edinburgh, UK. Association for Computational Linguistics.

Felix Gervits, Antonio Roque, Gordon Briggs, Matthias Scheutz, and Matthew Marge. 2021. How should agents ask questions for situated learning? an annotated dialogue corpus. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 353–359, Singapore and Online. Association for Computational Linguistics.

Jonathan Ginzburg. 2012. Grounding and CRification. In *The Interactive Stance*. Oxford University Press, Oxford.

Jonathan Ginzburg and Andy Luecking. 2021. Requesting clarifications with speech and gestures. In *Proceedings of the 1st Workshop on Multimodal Semantic Representations (MMSR)*, pages 21–31, Groningen, Netherlands (Online). Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Patrick GT Healey, Matthew Purver, James King, Jonathan Ginzburg, and Greg J Mills. 2003. Experimenting with clarification in dialogue. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 25.

Xiang Hu, Zujie Wen, Yafang Wang, Xiaolong Li, and Gerard de Melo. 2020. Interactive question clarification in dialogue via reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 78–89, Online. International Committee on Computational Linguistics.

Ryan Blake Jackson and Tom Williams. 2018. Robot: Asker of questions and changer of norms. *Proceedings of ICRES*.

Ryan Blake Jackson and Tom Williams. 2019. Language-capable robots may inadvertently weaken human moral norms. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 401–410. IEEE.

Jin-Hwa Kim, Nikita Kitaev, Xinlei Chen, Marcus Rohrbach, Byoung-Tak Zhang, Yuandong Tian, Dhruv Batra, and Devi Parikh. 2019. CoDraw: Collaborative drawing as a testbed for grounded goal-driven communication. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6495–6513, Florence, Italy. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Julia Kiseleva, Ziming Li, Mohammad Aliannejadi, Shrestha Mohanty, Maartje ter Hoeve, Mikhail Burtsev, Alexey Skrynnik, Artem Zholus, Aleksandr Panov, Kavya Srinet, et al. 2021. Neurips 2021 competition iglu: Interactive grounded language understanding in a collaborative environment. *arXiv preprint arXiv:2110.06536*.

Julia Kiseleva, Alexey Skrynnik, Artem Zholus, Shrestha Mohanty, Negar Arabzadeh, Marc-Alexandre Côté, Mohammad Aliannejadi, Milagro Teruel, Ziming Li, Mikhail Burtsev, et al. 2022. Iglu 2022: Interactive grounded language understanding in a collaborative environment at neurips 2022. *arXiv preprint arXiv:2205.13771*.

Theodora Koulouri and Stanislao Lauria. 2009. Exploring miscommunication and collaborative behaviour in human-robot interaction. In *Proceedings of the SIGDIAL 2009 Conference*, pages 111–119, London, UK. Association for Computational Linguistics.

Vaibhav Kumar and Alan W Black. 2020. ClarQ: A large-scale and diverse dataset for clarification question generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7296–7301, Online. Association for Computational Linguistics.

Charlotte Lambert, Ariel Cordes, Elli Kaplan, Prashant Jayannavar, and Julia Hockenmaier. 2019. Virtual world context encoding for grounded dialogue in minecraft.

Hyunhee Lee, Gyeongmin Kim, Yuna Hur, and Heuiseok Lim. 2021. Visual thinking of neural networks: Interactive text to image synthesis. *IEEE Access*, 9:64510–64523.

Zhenhuan Liu, Jincan Deng, Liang Li, Shaofei Cai, Qianqian Xu, Shuhui Wang, and Qingming Huang. 2020. Ir-gan: Image manipulation with linguistic instruction by increment reasoning. *Proceedings of the 28th ACM International Conference on Multimedia*.

Bodhisattwa Prasad Majumder, Sudha Rao, Michel Galley, and Julian McAuley. 2021. Ask what's missing and what's useful: Improving clarification question generation using global knowledge. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4300–4312, Online. Association for Computational Linguistics.

Matthew Marge and Alexander Rudnicky. 2015. Miscommunication recovery in physically situated dialogue. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 22–31, Prague, Czech Republic. Association for Computational Linguistics.

Shoya Matsumori, Yukikoko Abe, Kosuke Shingyouchi, Komei Sugiura, and Michita Imai. 2021. Lattegan: Visually guided language attention for multi-turn text-conditioned image manipulation. *IEEE Access*, 9:160521–160532.

Koh Mitsuda, Ryuichiro Higashinaka, Yuhei Oga, and Sen Yoshida. 2022. Dialogue collection for recording the process of building common ground in a collaborative task. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5749–5758, Marseille, France. European Language Resources Association.

Shrestha Mohanty, Negar Arabzadeh, Milagro Teruel, Yuxuan Sun, Artem Zholus, Alexey Skrynnik, Mikhail Burtsev, Kavya Srinet, Aleksandr Panov, Arthur Szlam, Marc-Alexandre Côté, and Julia Kiseleva. 2022. Collecting interactive multi-modal datasets for grounded language understanding.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. Association for Computational Linguistics.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. 2022. Teach: Task-driven embodied agents that chat. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2017–2025.

Matthew Purver, Jonathan Ginzburg, and Patrick Healey. 2003. On the means for clarification in dialogue. In *Current and new directions in discourse and dialogue*, pages 235–255. Springer.

Sudha Rao and Hal Daumé III. 2018. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2737–2746, Melbourne, Australia. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Verena Rieser, Ivana Kruijff-Korbayová, and Oliver Lemon. 2005. A corpus collection and annotation framework for learning multimodal clarification strategies. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 97–106, Lisbon, Portugal. Special Interest Group on Discourse and Dialogue (SIGdial).

Verena Rieser and Oliver Lemon. 2006. Using machine learning to explore human multimodal clarification strategies. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 659–666, Sydney, Australia. Association for Computational Linguistics.

Verena Rieser and Johanna Moore. 2005. Implications for generating clarification requests in task-oriented dialogues. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 239–246, Ann Arbor, Michigan. Association for Computational Linguistics.

Kepa Joseba Rodríguez and David Schlangen. 2004. Form, intonation and function of clarification requests in german task-oriented spoken dialogues. In *Proceedings of Catalog (the 8th workshop on the semantics and pragmatics of dialogue; SemDial04)*.

Michael Roth, Talita Anthonio, and Anna Sauer. 2022. SemEval-2022 task 7: Identifying plausible clarifications of implicit and underspecified phrases in instructional texts. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1039–1049, Seattle, United States. Association for Computational Linguistics.

Takaya Saito and Marc Rehmsmeier. 2015. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432.

David Schlangen. 2004. Causes and strategies for requesting clarification in dialogue. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue at HLT-NAACL 2004*, pages 136–143, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

David Schlangen and Raquel Fernández. 2007a. Beyond repair – testing the limits of the conversational repair system. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, pages 51–54, Antwerp, Belgium. Association for Computational Linguistics.

David Schlangen and Raquel Fernández. 2007b. Speaking through a noisy channel-experiments on inducing clarification behaviour in human-human dialogue. *Proceedings of Interspeech 2007*.

Julian Schlöder and Raquel Fernández. 2014. Clarification requests at the level of uptake. In *Proceedings of the 18th Workshop on the Semantics and Pragmatics of Dialogue - Poster Abstracts*, Edinburgh, United Kingdom. SEMDIAL.

Julian J. Schlöder and Raquel Fernández. 2015. Clarifying intentions in dialogue: A corpus study. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 46–51, London, UK. Association for Computational Linguistics.

Zhengxiang Shi, Yue Feng, and Aldo Lipani. 2022. Learning to execute actions or ask clarification questions. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2060–2070, Seattle, United States. Association for Computational Linguistics.

Svetlana Stoyanchev, Alex Liu, and Julia Hirschberg. 2013. Modelling human clarification strategies. In *Proceedings of the SIGDIAL 2013 Conference*, pages 137–141, Metz, France. Association for Computational Linguistics.

Svetlana Stoyanchev, Alex Liu, and Julia Hirschberg. 2014. Towards natural clarification questions in dialogue systems. In *AISB symposium on questions, discourse and dialogue*, volume 20.

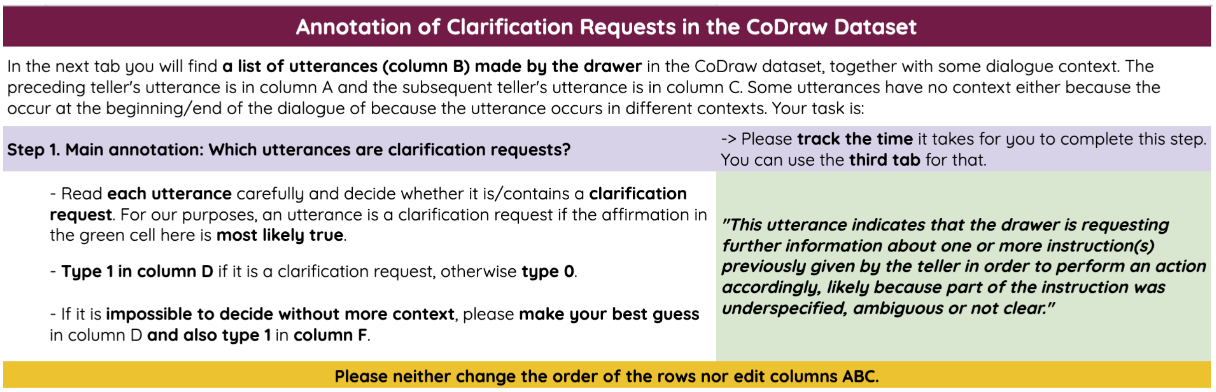
Svetlana Stoyanchev, Alex Liu, and Julia Bell Hirschberg. 2012. Clarification questions with feedback.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR.

Tianhao Zhang, Hung-Yu Tseng, Lu Jiang, Weilong Yang, Honglak Lee, and Irfan Essa. 2021. Text as neural operator: Image manipulation by text instruction. *Proceedings of the 29th ACM International Conference on Multimedia*.

C Lawrence Zitnick and Devi Parikh. 2013. Bringing semantics into focus using visual abstraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3009–3016.

C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. 2013. Learning the visual interpretation of sentences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1681–1688.



Annotation of Clarification Requests in the CoDraw Dataset

In the next tab you will find **a list of utterances (column B) made by the drawer** in the CoDraw dataset, together with some dialogue context. The preceding teller's utterance is in column A and the subsequent teller's utterance is in column C. Some utterances have no context either because the occur at the beginning/end of the dialogue of because the utterance occurs in different contexts. Your task is:

Step 1. Main annotation: Which utterances are clarification requests?

-> Please **track the time** it takes for you to complete this step. You can use the **third tab** for that.

- Read **each utterance** carefully and decide whether it is/contains a **clarification request**. For our purposes, an utterance is a clarification request if the affirmation in the green cell here is **most likely true**.
- **Type 1 in column D** if it is a clarification request, otherwise **type 0**.
- If it is **impossible to decide without more context**, please **make your best guess** in column D **and also type 1** in **column F**.

"This utterance indicates that the drawer is requesting further information about one or more instruction(s) previously given by the teller in order to perform an action accordingly, likely because part of the instruction was underspecified, ambiguous or not clear."

Please neither change the order of the rows nor edit columns ABC.

Figure 6: Instructions for the first annotator.

A Data Statement

Following Bender and Friedman (2018), in this section we provide information about the extended dataset. Figure 6 shows the instructions given to the first annotator.[8]

Curation Rationale. We annotate iCRs in all dialogues of the CoDraw dataset (Kim et al., 2019), which contains 9,993 dialogues produced by crowdworkers and has been released under a CC BY-NC 4.0 license. Please refer to the original paper for details about their data collection.

Language Variety and Speaker Demographic. The CoDraw dataset comprises written interaction in English, however no information about crowd worker demographic has been released in the dataset repository.

Annotator Demographic. The annotators who identified iCRs in CoDraw are a male and a female Computational Linguistics bachelor students who are non-native fluent English speakers working at colabPotsdam as student assistants. The students were paid according to the German's regulation for student assistants.

Situation. In CoDraw, crowdworkers exchanged written messages of up to 140 characters via a chat interface in a crowdsourcing tool. *IF* and *IG* had to send messages in alternating turns. The interaction was synchronous and task-oriented.

Text Characteristics. The CoDraw authors pre-process all collected utterances using a spell checker and tokenize the text with a natural language toolkit.

B Additional Corpus Analysis

We provide here further descriptive characteristics of the annotated dataset.

Table 7 shows a few negative examples, *i.e.* utterances that are not iCRs. Although utterances like *what do you see in the sky ?* and *anything else to change ?* can indeed be considered task-level CRs, we do not consider them iCRs because they do not directly refer back to a given instruction.

yeah it was a lot but thanks for finishing
i am a patient worker ready to start
check please and tell me what to change
anything else to change ?
what else is in the picture and where ?
i 've made all the changes you 've listed .
ok and look
alright , made changes
please be more specific thanks
ok anything else in the picture ?
yes , please lmk of any corrections
ok i got that :
ready whenever you are . :
what what is the first object and location ?
alright done
what do you see in the sky ?
tell me what we have

Table 7: Negative examples in CoDraw.

Figure 7 depicts in which rounds iCRs occur in the corpus. Given that the average dialogue length is 7.7 rounds, most instruction CRs in this dataset are occurring early in their corresponding dialogue. The distribution of the number of iCRs per dialogue is illustrated in Figure 8, where we see that it is very rare to have dialogues with more than 5 iCRs.

---

[8] The second annotator got more detailed instructions to perform the fine-grained annotation, which is not part of this publication. For the analysis and experiments in this work, we use the labels by the second annotator, who went through a more extensive background reading about CRs. There are few disagreements, as shown in the main text.

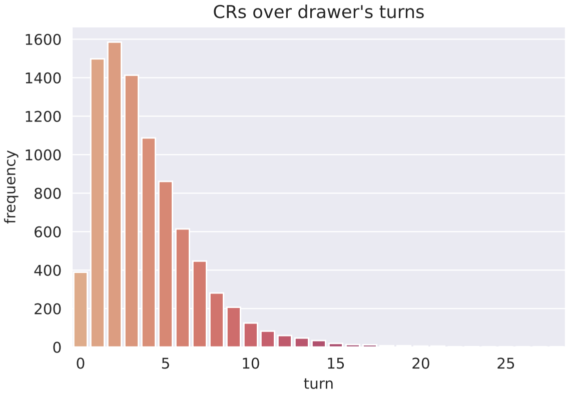


Figure 7: In which round iCRs occur.

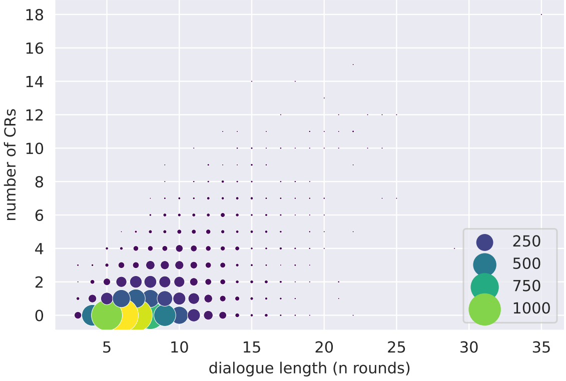


Figure 9: Number of iCRs vs. number of rounds.

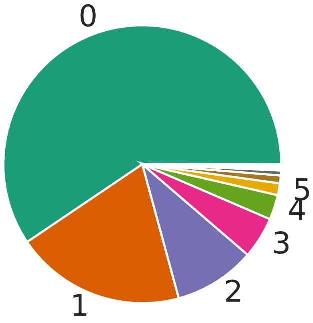


Figure 8: Number of iCRs per dialogue.

Figure 9 breaks down the number of iCRs per number of dialogue rounds. Dialogues with more than 10 iCRs are outliers, which is expected given that most dialogues have no more then 15 rounds.

14.58% of validation and 11.96% of test iCR utterance types also occur in the training set. 17.50% of validation and 14.81% of test iCR utterances also occur in the training set. The overlap is low, which is a desirable characteristic to reduce the memorization shortcuts for models trained on this dataset.

Computing actions and score differences. We group the drawing actions into three main categories: *addition* (when a clipart is added to the scene), *edit* (when some change occurs with an object that existed in the previous round), and *no action* if the drawer did not perform an action in a round. Edits can be deletion, move (position change), flip and resize. We do not track whether newly added cliparts get immediately flipped or resized in relation to the gallery when they are added. We noticed that some actions that visually seem to be only flips or resize happen together with a move. However, our inspection shows that this does not occur consistently with a specific subset of cliparts and there is also a portion of cases where flips and resizes occur without moves. Therefore, each move, resize, flip are counted as one separate action in our analysis. We compute the moves based on the differences over consecutive rounds in sequence of scene strings labeled *abs_d* provided in the original dataset, assuming that they describe the state of the canvas at the moment when the drawer sends their message, *i.e.* at the end of the current round.

Scene similarity is computed with the scripts made available by the CoDraw authors on GitHub.

C Reproducibility

In this section we describe the details of the implementation and datasets for reproducibility purposes. Further information and documentation is available in the code repository.

The random and trivial baselines are trained with scikit-learn (v1.1.2) with class weight set to balanced. A maximum of 1,000 iterations was still not sufficient for convergence is all cases. The hypothesis tests are carried out with SciPy (v1.10.0), using the permutation test for the difference of means, with type set to independent.

Models. Our models are implemented with PyTorch (v1.11.0) and PyTorch Lightning (v1.6.4). Metrics are computed using TorchMetrics (v0.10.0). The experiments were run in Linux 5.4.0-99-generic, machine/processor x86_64 in Python 3.9.12 on an NVIDIA GeForce GTX 1080 Ti GPUs with CUDA v11.6. The architecture of the full neural network model with its corresponding layers and dimensions were:

```
(model): Classifier(
  (img_encoder): ImageEncoder(
    (encoder): Linear(in_features=2048,
                      out_features=128,
                      bias=True)
  )
  (msg_encoder): TextEncoder(
    (encoder): Linear(in_features=768,
                      out_features=128,
                      bias=True)
  )
  (context_encoder): TextEncoder(
    (encoder): Linear(in_features=768,
                      out_features=128,
                      bias=True)
  )
  (classifier): DeeperClassifier(
    (classifier): Sequential(
      (0): LeakyReLU(negative_slope=0.01)
      (1): Dropout(p=0.1, inplace=False)
      (2): Linear(in_features=384,
                  out_features=256,
                  bias=True)
      (3): BatchNorm1d(256,
                       eps=1e-05,
                       momentum=0.1,
                       affine=True,
                      track_running_stats=True)
      (4): LeakyReLU(negative_slope=0.01)
      (5): Linear(in_features=256,
                  out_features=1,
                  bias=True)
    )
  )
)
```

The complete model has 558,465 trainable parameters. For the ablation experiments, the number of dimensions was reduced according to the input, and the number of parameters were 263,425 (no image), 427,265 (no context) and 427,265 (no utterance).

Training is carried out with the Adam optimizer (Kingma and Ba, 2015) to minimize weighted binary cross entropy (the weight is the hyperparameter weight cr) estimated with a sigmoid function applied to the output logits. We use the Bayes algorithm from comet.ml to perform hyperparameter search seeking to maximize Average Precision (and also AUC of the Precision-Recall curve in some preliminary experiments) on the validation set for Task 1. For the final version, we run 111 experiments during hyperparameter search. The optimal hyperparameters used in the experiments are shown in Table 8 together with their corresponding bounds. We use the second best performing configuration, because it is only around 6e-7 below the best one, but has more stable learning curves. All other experiments (ablation and Task 2) use the same configuration, except that the dimensions change according to the input vectors for ablation. We report the results of one run using the best configuration.

We use a decision threshold of 0.5 for the evaluation metrics that require a fixed threshold.

We train the models for up to 20 epochs, which takes around 3-4 minutes, including inference, which requires around 4 seconds. Although it takes more than 20 epochs to achieve a higher performance on the training set, the maximum for the validation set is reached in early epochs. We then use the model checkpoint with the highest Average Precision on the validation set to run evaluation on the test set.

We use the `all-mpnet-base-v2` model from SentenceTransformers to encode the texts into a representation with 768 dimensions. The image representation has 2048 dimensions. Images are preprocessed according to PyTorch Vision documentation (without resizing and centering) and features are extracted following recommendation on their forum. We use the pretrained model `resnet101` available from torchvision (v0.12.0).

Datasets. We use the same train/val/test splits as the original CoDraw dataset. The sizes and the distribution of labels in the annotated dataset is in Table 4. For retrieving the context embeddings, we add a /T token before the teller's utterance and a /D token before the drawer's utterances. We also add a /PEEK token before the utterances of the round when a peek action occurs. The context is limited to the last 200 tokens. Utterances are tokenized with blank spaces on the preprocessed published dataset.

D Detailed Results

We present more details about the performance on the validation set.

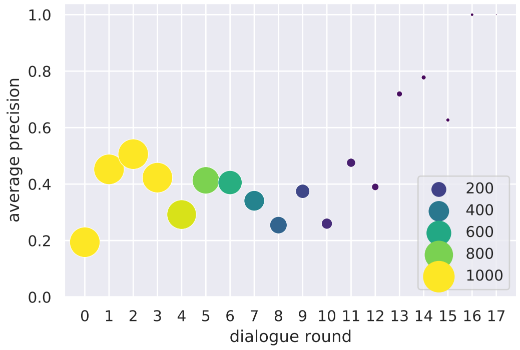
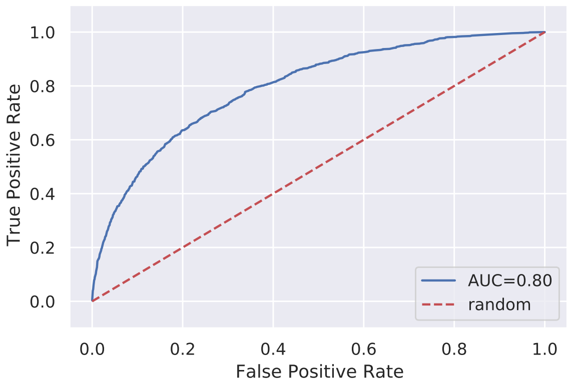


Figure 10: Average precision per round (validation set).

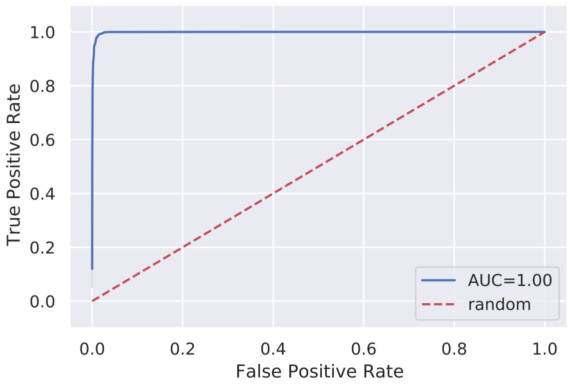
hyperparameter	**type**	**search bounds**	**optimal value**
accumulate gradient	discrete	1, 2, 5, 10, 25	25
batch size	discrete	32, 64, 128, 256, 512, 1024	128
clipping	discrete	0, 0.25, 0.5, 1, 2.5, 5, 10	1
dropout	discrete	0.1, 0.2, 0.3, 0.5	0.1
gamma	discrete	0.1, 0.5, 0.9, 0.99, 1	0.99
hidden dimension	discrete	32, 64, 128, 256, 512, 1024	256
internal embeddings dim	discrete	32, 64, 128, 256, 512, 1024	128
learning rate	discrete	0.1, 0.01, 0.001, 0.0001, 0.003, 0.0003, 0.00001, 0.0005	0.003
lr scheduler	categorical	none, exp, step	exp
lr step	integer	min=1, max=5	2
random seed	integer	min=1, max=54321	35466
weight cr	float	min=1, max=10	2.6125454767515217
weight decay	discrete	1, 0.1, 0.01, 0.001, 0.0001	0.0001

Table 8: Hyperparameters: Search bounds and optimal values.

Figure 10 shows the AP metric split by round, Figure 11 presents the ROC curves and Figure 12, the Precision Recall curves of the best checkpoint.

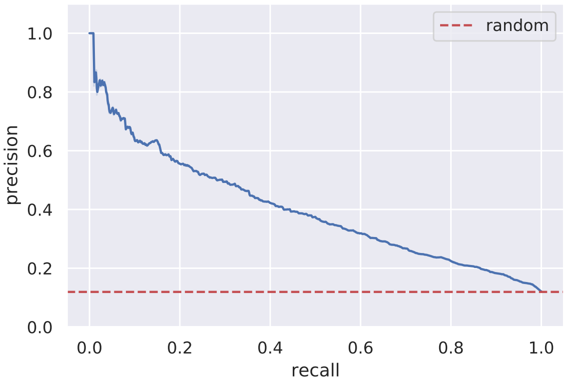


(a) Task 1: *IF*

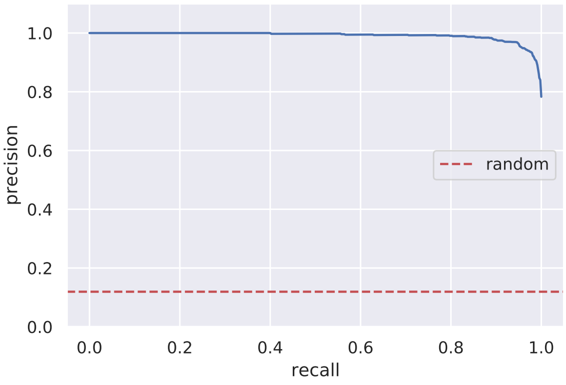


(b) Task 2: *IG*

Figure 11: ROC curves (validation set).



(a) Task 1: *IF*



(b) Task 2: *IG*

Figure 12: Precision-Recall curves (validation set).

Can Synthetic Text Help Clinical Named Entity Recognition? A Study of Electronic Health Records in French

Nicolas Hiebel[†], **Olivier Ferret**[‡], **Karën Fort**[⋆], **Aurélie Névéol**[†]
[†]Université Paris Saclay, CNRS, LISN, France
[‡]Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
[⋆]Sorbonne Université / Université de Lorraine, CNRS, Inria, LORIA, France
[†]firstname.lastname@lisn.upsaclay.fr, [‡]olivier.ferret@cea.fr, [⋆]karen.fort@loria.fr

Abstract

In sensitive domains, the sharing of corpora is restricted due to confidentiality, copyrights, or trade secrets. Automatic text generation can help alleviate these issues by producing synthetic texts that mimic the linguistic properties of real documents while preserving confidentiality. In this study, we assess the usability of synthetic corpus as a substitute training corpus for clinical information extraction. Our goal is to automatically produce a clinical case corpus annotated with clinical entities and to evaluate it for a named entity recognition (NER) task. We use two auto-regressive neural models partially or fully trained on generic French texts and fine-tuned on clinical cases to produce a corpus of synthetic clinical cases. We study variants of the generation process: (i) fine-tuning on annotated vs. plain text (in that case, annotations are obtained a posteriori) and (ii) selection of generated texts based on models' parameters and filtering criteria. We then train NER models with the resulting synthetic text and evaluate them on a gold standard clinical corpus. Our experiments suggest that synthetic text is useful for clinical NER.

1 Introduction

The lack of specific resources might be the most common hurdle encountered when working in Natural Language Processing (NLP), whether it be the lack of task-specific resources, domain-specific resources, or both. A major challenge for biomedical NLP in languages other than English is the lack of available clinical corpora that can be shared with the community (Névéol et al., 2018).

One way of addressing this problem is to generate new data automatically. The performance of recent language models based on the transformer architecture (Vaswani et al., 2017) on various tasks that involve generating text (translation, summarization, etc.) makes it possible to use the transformer's power to generate text to address resource sparsity.

More specifically for generation, large generative language models such as GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020) have the ability to generate well-written text containing a wealth of information. The knowledge gained during the pre-training of those large language models could be helpful in more specialized domains where there is not enough data to train a generative model from scratch.

Dealing with medical data brings one particular issue. The generated data should not contain sensitive information from the data it is inspired by while remaining as close as possible to it. This means that the evaluation of this task is different from the usual evaluation of text generation.

The evaluation of text generation is a difficult problem (Howcroft et al., 2020; Novikova et al., 2017). There is a distinction between intrinsic and extrinsic evaluation (Gehrmann et al., 2022). For intrinsic evaluation in generation tasks such as translation and summarization, the quality is usually first automatically measured using metrics such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), or BERTScore (Zhang et al., 2020). Those metrics and many others compare the system's output to one or several human-written references (Frisoni et al., 2022). However, a gold reference does not exist for an open-generation task.

Human evaluation is usually the best way of intrinsically evaluating text generation. Still, there can be several problems. First, human evaluation should be conducted once the system's output can be considered "good enough" to avoid wasting human evaluation costs. This means there is a need to measure if the system is ready to be manually evaluated. Second, it might not be possible for a human to assess the quality of generated data at a larger scale, if the generated data is not considered as a set of independent sentences but as a whole self-sufficient corpus.

This is why extrinsic evaluation of the generated

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2320–2338
May 2-6, 2023 ©2023 Association for Computational Linguistics

data is also important, meaning evaluating the usefulness of the generated data (Stadler et al., 2022). However, to our knowledge, methodologies to extrinsically evaluate open text generation are rare.

In this work, we explore text generation using a pre-trained language model and we focus our evaluation on named entity recognition (NER).

This paper offers the following contributions:

- we generate two sets of clinical cases in French with two variants;
- we introduce an evaluation protocol of the generated texts using ngram overlap and NER models;
- we compare the generated texts with real clinical case corpora.

2 Related Work

Generating data to address resource sparsity is an active field of research. Pre-trained transformer-based models have been used as data augmentation tools, such as in Claveau et al. (2022), where the authors fine-tuned the pre-trained auto-regressive English language model GPT-2 to create labeled examples for training classification models in French and English. In the medical domain, Amin-Nejad et al. (2020) use an encoder-decoder transformer architecture to generate discharge summaries given a context. The context gives a target to the generation that is used to evaluate the generated texts. Melamud and Shivade (2019) train an LSTM-based language model to generate shareable clinical notes using differential privacy (Dwork et al., 2006). Word embeddings are trained on the generated data and compared to word embeddings trained on the original data on two existing benchmarks. Ive et al. (2020) used an entity extraction method to build a generative model with entities as input and the document as output and generated artificial mental health records in English.

Several efforts exploit Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) to seamlessly augment original data. Among them, Abedi et al. (2022) use GANs to generate artificial structured data in the medical domain. Choi et al. (2017) created medGAN, a framework to generate artificial structured electronic health records (EHR) with discrete values. Torfi et al. (2022) rely on a combination of autoencoders (Kingma and Welling, 2014) and GANs to generate artificial structured data with both discrete and continuous values.

3 Method

3.1 Experiment description

The experiment presented in this paper is designed to evaluate open text generation in the clinical domain. The goal is to create an artificial corpus presenting the same *characteristics* as a real clinical corpus. One challenge of this task is that defining and recognizing the *characteristics* of a clinical corpus is an open question.

We envisaged the *characteristics* of clinical corpus along three perspectives:

1. a distributional perspective: we want to create an artificial clinical corpus with a token distribution similar to that of a real corpus. This aspect will be measured using self-BLEU (Zhu et al., 2018) and perplexity;
2. a qualitative perspective: we want to create an artificial corpus that mimics a real clinical corpus in terms of semantic and linguistic content. This aspect will be measured by manual analysis of samples of generated texts;
3. an application perspective: we want to create an artificial clinical corpus that can be used to train NER models to be applied to a real clinical corpus for clinical information extraction. This aspect will be measured through the extrinsic evaluation of Section 4.2.

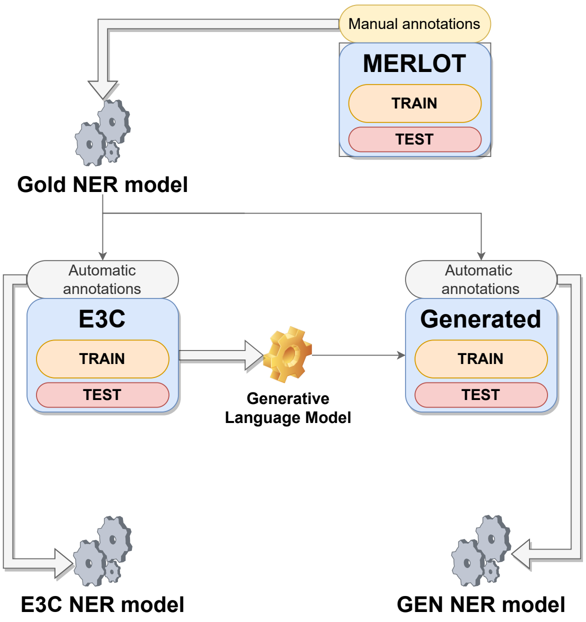


Figure 1: Description of the experiment conducted to evaluate NER on artificial data.

Figure 1 presents the global setup for evaluating generated data for NER. The specific corpora

(MERLOT and E3C) named in the figure will be described in Section 3.2.

Briefly, we leverage a corpus of clinical documents in French containing high-quality entity annotations (MERLOT). This corpus represents the target real data we aim to analyze. It is used in part to train clinical NER models and in part for evaluating clinical NER models. We then use a clinical case corpus (E3C) for fine-tuning generation models, with the option of generating tagged text or plain text that can then be tagged with the Gold NER model.

In the end, each of the newly annotated corpora is split into training and testing sets and is used to train new NER models. These final models are then tested on the test sets of each corpus.

3.2 Corpora

This section presents the corpora used in our experiments. Table 1 presents descriptive statistics.

	E3C$_{FR}$	CAS	MERLOT
Toks	328,645	231,662	148,476
Docs	1,009	717	500
Toks/doc	325.7	323.1	297.0
Sents/doc	15.2	15.8	14.4
Avg sentence length	21.4	20.4	20.6
Self-BLEU	0.68	0.66	0.69

Table 1: Statistics of the real medical corpora.

MERLOT (Campillos et al., 2018) is a corpus of 500 clinical documents in French from the hepato-gastro-enterology and nutrition specialties. It comprises 148,476 tokens with annotations for entities, attributes and relations. This text genre represents the type of clinical narratives contained in electronic health records that we would like to analyze.

CAS (Grabar et al., 2018) is a French medical corpus containing 717 clinical case descriptions published in the literature. Clinical cases contain descriptions of the medical history of patients, treatment received at the hospital, and follow-up care. CAS covers several medical specialties. The corpus has been de-identified and the publication of case descriptions is done with patient consent. The corpus can be obtained from the DEFT 2020 shared task organizers[1].

E3C (Magnini et al., 2020) is a freely available multilingual corpus[2]. It contains biomedical documents extracted from different sources, including journals, abstracts, existing biomedical corpora, patient information leaflets for medicines, and others.

In this work, we focused on the French part of the corpus, selecting only the clinical cases published in the literature, which are the most similar documents to the clinical narratives we are interested in. Clinical cases from E3C cover a variety of medical specialties and are distinct from those in CAS. This part of the corpus contains about 1,000 documents and more than 300,000 tokens.

3.3 Entity annotation for E3C

Entity annotations for E3C are needed for two reasons. First, we integrate the annotations in the form of XML tags into the E3C texts. Fine-tuning the model on this version of the corpus will enable the model to learn to generate tags directly during the generation process. Second, the annotations will be used to train and test NER models. We use the manual annotations of entities made on the MERLOT corpus to train NER models (Bannour et al., 2022) to annotate the E3C corpus. Using an automatic rather than a manual annotation for training our final NER models can lead to error propagation. We use an ensemble strategy to mitigate this issue: five different models are trained and applied to the corpus with a majority vote for the annotation (annotation is kept if more than three models agree on the annotation), which is classically known as a good way to reduce errors. The train and validation sets are shuffled for the training of each NER model and a different seed is used.

3.4 Generation

Pre-trained models such as GPT-2 have been proven to allow for generating complementary textual data in a data augmentation context. In this work, we explore the ability of auto regressive neural models (viz. GPT2 and BLOOM) trained partially or exclusively on French to adapt to the medical domain and generate relevant clinical case descriptions.

We use two existing GPT models. The first model is a one-billion-parameter model pre-trained on generic French texts by Simoulin and Crabbé (2021). The corpus used to pre-train the model was extracted from Wikipedia, Project Gutenberg[3],

---

[1] `https://deft.limsi.fr/2020/`

[2] `https://github.com/hltfbk/E3C-Corpus`

[3] `www.gutenberg.org`

OpenSubtitle (Lison and Tiedemann, 2016), and CommonCrawl. In this paper, this model will be referred to as LLF.

The second model is a smaller version of the BigScience BLOOM model[4]. BLOOM was trained on 46 different languages, including French[5]. We chose the one-billion-parameter version for a fair comparison between the two models. Both models are available on the HuggingFace platform[6][7].

3.4.1 Fine-tuning the models

We used the freely available `E3C` corpus for fine-tuning our model on medical texts so that the generation process can be replicated[8]. We added special tokens for marking the beginning and end of each document to make the model generate whole documents. Two versions of the corpus were used: one with the annotations described in 3.3 so that the models directly learn to generate annotations, and one without annotations. The corpus was split into training and validation sets, with 90% of the corpus for training and 10% for validation. The model was trained with a maximum of 10 epochs, but training stopped early if the perplexity on the validation set did not decrease significantly during two consecutive epochs. We selected a learning rate of 2e-5, negative likelihood loss with the AdamW optimizer from the torch package[9]. In the end, we have four models, two for each pre-trained model with one version producing plain untagged text and one including annotations to produce tagged text.

3.4.2 Generation process

Clinical documents were generated using the token marking the beginning of a document as input. No additional prompts were supplied to the models. We thought that giving information such as the beginning of a document as a prompt might cause the model to reproduce text seen during fine-tuning.

Many parameters can be used during inference to influence the generative models' output. We decided to focus on three parameters: nucleus sampling (top-p) value, temperature value, and repetition penalty. The rest of the parameters were left with default values. At each step (token) of the generation process, the models output a probability distribution for vocabulary items. For generating diverse outputs, sampling consists in randomly picking the next token to generate, as opposed to greedy decoding, where the most probable token is selected. Nucleus sampling, or top-p sampling, restricts the sampling to the tokens whose summed probabilities exceed a threshold p. It allows for more diversity while remaining consistent. Temperature is a way of decreasing or increasing the disparities in the probabilities of the vocabulary at a generation step. A temperature higher than 1 will bring the probabilities closer and make the model less confident, and bring more diversity in outputs. A temperature lower than 1 will do the opposite. Finally, a repetition penalty will encourage the model to avoid falling into loops. We generated several samples of approximately 10% of the target-generated text to assess the impact of those three parameters on the self-BLEU score of the generated sample and an oracle perplexity with the seven billion parameters version of the BLOOM model. More details can be found in appendix A.2. We noticed expected behaviors: increasing temperature and repetition penalty increases diversity (lower self-BLEU) but also increases perplexity. However, we found that top-p sampling has a low impact on the two metrics. Perhaps because by default, the model also adds top k sampling (selecting amongst the k most probable tokens at each generation step) that lowers the impact of top-p. In the end, since we wanted a high diversity output to match `E3C`'s diversity, we selected high values for temperature (1.2 for models without tags and 1.5 for models with tags). We also kept a very high repetition penalty (10) because we noticed that a lot of generations ended with the model generating the same tokens over and over even with a high value. Finally, we selected a default value of 0.95 for top-p sampling.

We aimed for the models to generate complete documents with length variations similar to `E3C`. We computed the average length of `E3C` documents (for both annotated and non-annotated versions) and combined it with standard deviation, exploiting the ability of the models to close a document with an ending token and start a new one in the same generation process.

---

[4] `https://lstu.fr/bloom`

[5] `https://huggingface.co/spaces/bigscience/BigScienceCorpus`

[6] `https://huggingface.co/asi/gpt-fr-cased-base`

[7] `https://huggingface.co/bigscience/bloom-1b1`

[8] Code available here : `https://gitlab.inria.fr/codeine/synthetic-text-for-ner`

[9] `https://pytorch.org/`

3.5 Filtering generated texts

We ensured that the generated corpora had certain characteristics for a fair comparison to the reference corpus `E3C`.

Large auto-regressive language models can generate good-quality texts. However, the generation can present some flaws such as repetitions and incoherence, especially as the generation length gets longer (Pillutla et al., 2021). We established a few criteria that we could handle automatically by observing a few generated samples:

- overly long tokens: the models sometimes generate an incoherent sequence of characters or combine many technical words (e.g., "*céphalo-rachidiencéphalico-mandibulaire*");
- repetition of one to many tokens over and over, especially towards the end of a document;
- the models start a sequence of short tokens often with numbers (e.g., "*classée T4N2 M0 I 2 B 3 C 4 D 5*") or start reciting the alphabet (e.g., "*hépatite A B C D E F*").

We addressed these three cases by excluding documents based on token length, trigram frequency, and regular expressions.

The thresholds used for the three parameters were adjusted by analyzing the excluded documents with different threshold values. In the end, we excluded documents with tokens of thirty characters or more, documents with trigrams appearing more than three times, and repetition of at least three short tokens combined with numbers, or single letters.

We also wanted our generated corpora to have a diversity close to that of `E3C` as computed with the self-BLEU score (0.68 for `E3C`). We generated a significantly bigger amount of text than the targeted 330,000 tokens and selected a subset of diverse documents using the tool developed by Cohen et al. (2013). Then, we computed the self-BLEU score for the selected documents, including the BLEU score of each document. We removed the ten documents with the highest BLEU, that is the documents bringing the least diversity, and repeated the process until obtaining a self-BLEU score and a number of tokens close to the `E3C` corpus.

The characteristics of the generated corpora are shown in Table 2. As we explained, each corpus generated corresponds to a filtered subset of the total amount of text generated. We noticed that text generated with the two configurations using the LLF model needed less filtering compared to their BLOOM counterpart. Configurations with texts generated with tags also needed more filtering. Globally, using both filters, we discarded 0.45% of generated text for $\texttt{Bloom}_{E3C}$, 0.54% for $\texttt{Bloom}_{E3C+T}$, 0.30% for $\texttt{LLF}_{E3C}$, and 0.45% for $\texttt{LLF}_{E3C+T}$. Compared to Table 1, we can see that only $\texttt{Bloom}_{E3C+T}$ presents much shorter documents. However, the sentences of the generated corpora are a lot longer than in `E3C`, especially with $\texttt{Bloom}_{E3C}$ that do not even have two sentences per document. Finally, corpora generated with BLOOM are a little less diverse.

	$\texttt{Bloom}_{E3C}$	$\texttt{Bloom}_{E3C+T}$	$\texttt{LLF}_{E3C}$	$\texttt{LLF}_{E3C+T}$
Toks	329,328	346,413	328,498	336,154
Docs	943	1,997	1,028	978
Toks/doc	349.2	173.5	319.6	343.7
Sents/doc	1.8	3.1	6.9	7.2
Len Sent	194	56.0	46.3	47.7
Self-BLEU	0.70	0.73	0.68	0.67
Perplexity	9.97	9.27	10.03	11.9

Table 2: Statistics of the generated corpora. Oracle perplexity was computed using an oracle pre-trained model of BLOOM with 7 billion parameters. E3C's perplexity using this model was 22.0 (Len Sent=average sentence length).

3.6 Automatic intrinsic evaluation

As a complement to BLEU scores, we measured the overlap between the generated corpus and the original corpus to characterize potential memorization of the original documents processed during fine-tuning.

To this end, we developed a python tool based on overlapping ngrams to compare the generated texts and the original text and to detect the rare ngrams in the original corpus that appear on the generated corpus. For a fair evaluation, we also show results of the same comparison between the corpus used for fine-tuning (`E3C`) and another, non-overlapping, clinical case corpus in French (`CAS`).

4 Results

Table 2 shows global synthetic corpus statistics and Table 3 presents sample texts generated with each model configuration.

Table 4 shows the number of entity annotations for `E3C`, `MERLOT`, and the generated corpora. We can notice that the corpora generated without tags ($\texttt{Bloom}_{E3C}$, $\texttt{LLF}_{E3C}$) contain almost the same number of annotations as the `E3C` corpus (around 9,000

Tags	LLF	BLOOM
No	Il s’agit d’un patient âgé de 52 ans, sans antécédents pathologiques notables. Admis aux urgences pour douleurs abdominales diffuses associées à des vomissements post-prandiaux évoluant depuis deux semaines en rapport avec une obstruction du canal cystique dont l’examen clinique a objectivé un ictère cutanéo-muqueux généralisé associé à quelques signes inflammatoires non spécifiques (extériorrhée et urétéro-hydronéphrose). L’examen ophtalmologique n’a révélé aucun autre signe oculaire ou ostéo-nerveux associés.	Il s’agissait d’une patiente âgée de 55 ans ayant comme antécédents une polyarthrite chronique touchant toutes les articulations métacarotidiennes évoluant depuis plus de 10 ans associée à une insuffisance surrénalienne fonctionnelle qui s’est aggravée après l’installation d’un diabète insulino-résistant non insulinodépendant nécessitant son hospitalisation six mois avant sa consultation chez notre confrère ophtalmologue afin d’être mis sous traitement médicamenteux associant : imipramine 250 mg / jour ; dexaméthasone 150 mg/jour; corticostéroïde prednisolone 5mg/kg/24h)
Yes	Une femme âgée de 55 ans, sans antécédents pathologiques particuliers était hospitalisée pour une dyspnée d’effort évoluant depuis 6 jours. L’ examen clinique a montré un état général altéré avec des signes inflammatoires du membre supérieur gauche ainsi qu’un syndrome grippal (une fièvre à 39 ° C) et le reste de l’ examen neurologique était sans particularités. La radiologie standard n’a objectivé ni troubles sensitivo-moteur ni convulsions fébriles.	Il s’agissait d’un patient de 50 ans, sans antécédents pathologiques particuliers, admis aux urgences pour des douleurs épigastriques aiguës associées à une distension abdominopelvienne évoluant depuis deux jours. L’ examen clinique trouvait un patient en assez bon état général (Apgar: 10/10). Le bilan préopératoire objectivait une fonction rénale normale et la CRP était à 12 mg/l.

Table 3: Excerpts of sample texts generated with top-p 0.95, temperature 1.2, and penalty 10. Highlighted sections in the lower row samples represent the entity tags generated; see Table 4 for color code. For text generated without entity tags, entity annotations are created post-generation; complete documents are shown in appendix A.4.

annotations each). However, the corpora generated with tags contain a smaller number of annotations with 5,630 annotations for $\texttt{Bloom}_{E3C+T}$ (two-thirds of `E3C`) and 3,893 annotations for $\texttt{LLF}_{E3C+T}$ (around 40% of `E3C`). This suggests that directly generating annotations might not be the optimal way of generating text for a NER task.

4.1 Ngram overlap

To compare the overlap of ngrams between `E3C` and the corpora generated by a model fine-tuned with `E3C`, we compute an overlapping score between the corpora. The overlapping score between two corpora is computed as follows:

$$\frac{\text{Number of unique common ngrams}}{\text{Total number of unique ngrams}} \tag{1}$$

The overlapping score varies between 0 and 1. A high overlapping score means that the two corpora have many tokens in common whereas a low score indicates few common tokens.

Table 5 presents the overlap between `E3C` and the generated corpora. We also added the comparison between `E3C` and `CAS`, another clinical case corpus (see appendix A.1 for the full ngram range). It is interesting to see that the `CAS` corpus has a higher overlapping score with `E3C` than the generated corpora. Amongst the generated corpora, $\texttt{Bloom}_{E3C}$ has the highest overlapping score with `E3C` for unigrams, $\texttt{Bloom}_{E3C+T}$ comes second, and $\texttt{LLF}_{E3C}$ and $\texttt{LLF}_{E3C+T}$ have the lowest scores. However, $\texttt{Bloom}_{E3C}$ has the lowest overlapping score for 8grams than the rest, even less than `CAS`. $\texttt{LLF}_{E3C+T}$ still has a small percentage of tokens in common with E3C, but $\texttt{Bloom}_{E3C+T}$ and $\texttt{LLF}_{E3C}$ have now a higher percentage than the rest. This means that even though they do not have that many unigrams in common with `E3C`, they tend to replicate longer sequences, which indicates more memorization and thus more risks regarding confidentiality. It is surprising to see that there is no regularity between the models and their fine-tuning configurations, and we are not sure how to explain this. In the end, this

	E3C	`Bloom`$_{E3C}$	`Bloom`$_{E3C+T}$	`LLF`$_{E3C}$	`LLF`$_{E3C+T}$	**Merlot**
Type						
Anatomy	1,533	1,798	799	1,056	360	922
Chemical_Drugs	307	388	390	350	264	283
Concept_Idea	493	376	275	370	241	684
Devices	36	42	72	5	41	219
Disorders	1,662	1,545	1,146	1,409	806	1,047
Dose	103	179	219	310	54	188
Hospital	74	65	129	50	54	170
LivingBeing	533	492	342	842	330	866
Localization	450	490	128	467	121	169
Measure	1,496	1,330	670	2,109	421	1,070
Mode	24	68	66	26	6	40
BiologicalProcess	127	83	102	62	78	186
Procedure	1,883	1,650	967	985	913	1,779
TemporalExpression	455	610	325	878	204	885
Overall	9,176	9,116	5,630	8,919	3,893	8,510

Table 4: Number of annotations of each type for each test corpus.

Corpus	Unigram	8gram
`Bloom`$_{E3C}$	0.16419	0.00011
`Bloom`$_{E3C+T}$	0.13887	0.00020
`LLF`$_{E3C}$	0.11740	0.00023
`LLF`$_{E3C+T}$	0.11935	0.00013
CAS	0.20373	0.00013

Table 5: Overlap scores of unigrams and 8grams between the generated corpora and `E3C`. Each line corresponds to the comparison between the corpus and `E3C`. We added the comparison of `E3C` with `CAS` as a baseline.

table might be in favor of `Bloom`$_{E3C}$, because it appears to be the closest corpus to `E3C` in terms of unigram, while not memorizing too much. However, this is yet to be confirmed.

4.2 Extrinsic evaluation

Tables 6 to 8 present the results of our extrinsic evaluation of the generated corpora through the NER task described in Section 3. More precisely, the three tables report the results of the same set of models on three different types of corpora. All the results were obtained by averaging the scores of five models trained with different seeds and with shuffled documents in the train and validation sets of the corpus.

The first thing to note is that the best performance for the test part of a given corpus is obtained by a model trained on the training part of this corpus. While this is not a surprise from a general viewpoint, it means in our context that the difference in quality between a generated corpus and a natural corpus is not high enough for balancing the advantage brought by the homogeneity between the training and the test corpora. The quality of the generated corpora for our NER task is confirmed by Table 6, which gives the results on the natural corpora and more specifically, on the `MERLOT` corpus. Since it was annotated manually, unlike the other corpora, we consider `MERLOT` as a high-quality reference. While the model trained on `E3C` is significantly outperformed by the model trained on `MERLOT`, we can observe that the models trained on `Bloom`$_{E3C}$ and `LLF`$_{E3C}$ are close to it, with a small advantage for `LLF`$_{E3C}$. Hence, in our context, using a corpus generated from a natural corpus is fairly equivalent to using the natural corpus.

Surprisingly, the performance of all the models on `E3C` strongly decreases compared to their performance on `MERLOT` while all of them, except one, are trained either from `E3C` directly or from a corpus generated from `E3C`. More precisely, it seems that for a NER model, the source of the annotations, `MERLOT` in our case since they are produced by a model trained on this corpus, is more important than the corpus on which the model was trained. Finally, Table 6 shows, through `Bloom`$_{E3C+T}$ and `LLF`$_{E3C+T}$, that generating annotated texts is the worst strategy for both `E3C` and `MERLOT`. This strategy particularly hurts recall, a result which can be certainly explained by the fact that the number of generated annotations (see Table 4) is low compared to the number of annotations produced by the NER model trained on `MERLOT`.

Table 7 reports the results of our models on the generated corpora `Bloom`$_{E3C}$ and `LLF`$_{E3C}$. The first observation is that, with the exception of `Bloom`$_{E3C+T}$ and `LLF`$_{E3C+T}$, the performance of the models is close to the performance obtained on `E3C`, which is not a surprise since the `Bloom`$_{E3C}$ and `LLF`$_{E3C}$ corpora were generated from EC3. However, it confirms that a generated corpus can be used as an interesting substitute for its source corpus. It is true for training a NER model but also for its evaluation. Table 7 also shows that our models obtain better results on `LLF`$_{E3C}$ than on `Bloom`$_{E3C}$, which is not observed with such a magnitude in Table 6 when the corpora are used for training our models: there is no difference between `Bloom`$_{E3C}$ and `LLF`$_{E3C}$ for the evaluation on `E3C` and only a small difference in favor of `LLF`$_{E3C}$ for the evaluation on `MERLOT`. We have observed that the punctuation in the long texts generated by BLOOM tends to be very sparse, which could explain to some extent the difference between the evaluations on `Bloom`$_{E3C}$ and `LLF`$_{E3C}$. Finally, the most noticeable observation is the poor performance of the models trained on `Bloom`$_{E3C+T}$ and `LLF`$_{E3C+T}$, especially recall. In that case, the explanation is once again related to the scarcity of the generated annotations used for training the models. However, the difference with the evaluation on the `E3C` corpus is more difficult to interpret.

The last table of results for the extrinsic evaluation, Table 8, mainly confirms our findings concerning the characteristics of the `Bloom`$_{E3C+T}$ and `LLF`$_{E3C+T}$ corpora. All the models trained on corpora annotated by the NER model trained on `MERLOT` obtain very good recall values since these corpora contain a much larger number of annotations than the `Bloom`$_{E3C+T}$ and `LLF`$_{E3C+T}$ corpora. Moreover, the very good results of the models trained on `Bloom`$_{E3C+T}$ and `LLF`$_{E3C+T}$ tend to show that their generated annotations are fairly easy to learn. Finally, unlike what we have observed in Table 7, the results on the corpus generated by BLOOM, `Bloom`$_{E3C+T}$, are higher than the results on the corpus generated by LLF, `LLF`$_{E3C+T}$. However, further investigations should be done to determine if BLOOM generates better annotations than LLF or if the generation of annotations tends to improve the quality of the text generated by BLOOM.

	Test					
	E3C			**MERLOT**		
Training	**P**	**R**	**F**	**P**	**R**	**F**
E3C	**54.6**	**55.5**	**55.0**	64.4	78.0	70.5
MERLOT	53.2	**55.5**	54.3	**85.2**	**85.8**	**85.5**
`Bloom`$_{E3C}$	53.6	53.8	53.7	63.1	74.9	68.5
`LLF`$_{E3C}$	53.4	53.2	53.3	64.5	76.4	70.0
`Bloom`$_{E3C+T}$	48.9	40.4	44.3	71.7	55.1	62.3
`LLF`$_{E3C+T}$	49.0	39.5	43.7	75.7	46.4	57.5

Table 6: Results of the NER task for the natural corpora (P=precision, R=recall, F=F-measure).

	Test					
	`Bloom`$_{E3C}$			**`LLF`$_{E3C}$**		
Training	**P**	**R**	**F**	**P**	**R**	**F**
E3C	47.0	48.2	47.6	53.6	54.9	54.2
MERLOT	44.9	48.8	46.8	50.6	55.6	53.0
`Bloom`$_{E3C}$	**48.5**	**49.8**	**49.1**	52.3	53.3	52.8
`LLF`$_{E3C}$	46.9	47.5	47.2	**55.4**	**57.3**	**56.3**
`Bloom`$_{E3C+T}$	40.7	22.2	28.7	45.6	25.0	32.3
`LLF`$_{E3C+T}$	37.2	21.3	27.1	46.1	20.9	28.8

Table 7: Results of the NER task for the generated corpora without annotation.

4.3 Qualitative evaluation

Samples of 15 automatically generated texts per model (for a total of 13,809 tokens for the 60 documents) were annotated using BRAT[10] (Stenetorp et al., 2012) and aimed to assess specific characteristics of the texts including (1) grammaticality; and (2) clinical coherence at the sentence and document level (e.g. in Table 3 one document reports Agpar score, a test performed on newborn infants, for a 50-year-old patient). See appendix A.5 for details.

The manual analysis suggests that clinically incoherent passages are more likely to be found towards the end of the texts, and within text generated without tags. In particular, texts generated with BLOOM tend to exhibit run-on sentences that become increasingly incoherent. However, there are generally more clinical inconsistencies and grammatical errors in the texts generated with the LLF vs. BLOOM models. We also note that the LLF models generate more than one case description inside a single document.

Overall, manual analysis suggests that texts generated with the `Bloom`$_{E3C+T}$ model are the closest

---

[10] Brat Rapid Annotation Tool `https://brat.nlplab.org/`

	Test					
	`Bloom`$_{E3C+T}$			`LLF`$_{E3C+T}$		
Training	**P**	**R**	**F**	**P**	**R**	**F**
E3C	67.6	90.3	77.3	57.3	90.6	70.2
MERLOT	60.3	86.2	71.0	48.7	86.6	62.3
`Bloom`$_{E3C}$	66.4	89.0	76.1	55.8	89.8	68.8
`LLF`$_{E3C}$	66.3	88.5	75.8	55.7	89.2	68.6
`Bloom`$_{E3C+T}$	**91.4**	**94.0**	**92.7**	80.9	88.0	84.3
`LLF`$_{E3C+T}$	85.1	86.0	85.5	**89.4**	**93.4**	**91.3**

Table 8: Results of the NER task for the generated corpora with annotations.

to original clinical case descriptions.

5 Conclusion

We presented four models for generating synthetic clinical cases in French. Our experiments suggest that synthetic text is useful for training NER models in the clinical domain. While text generated with `Bloom`$_{E3C+T}$ seems more natural, text generated with `LLF`$_{E3C}$ and `Bloom`$_{E3C}$ yields promising NER performance.

6 Ethical Considerations

We were able to identify two types of potential ethical issues with our research: i) the carbon footprint generated by the training of the models and ii) the patients' data protection. We detail here how we addressed or measured them.

For our experiments, we had to train a number of models, thus generating a carbon footprint, which we measured using Carbontracker (Anthony et al., 2020), although we acknowledge that this tool can only account for one in four sources of experiment carbon impact (Bannour et al., 2021). The total CO2 emission of our experiments is 11,313g. Carbontracker also measured the energy cost, estimated at 39.2kWh. This includes the fine-tuning of pre-trained language models, the preliminary experiments on evaluation, the generation of artificial corpora, and the training of NER models.

The corpora we used for the experiments are compliant with data privacy regulations: either they do not contain personal information at all (`E3C` and `CAS`) or they were de-identified according to a protocol approved by the CNIL (Commission de l'Informatique et des Libertés), an independent French administrative regulatory body whose mission is to ensure that data privacy law is applied to the collection, storage, and use of personal data (`MERLOT`).

Within the framework of our project, we intend to investigate whether our model contains sensitive data. However, this task needs to be completed by a similar investigation to be led on the language models themselves, as they may contain such data (for example, from blog posts that might have been collected by CommonCrawl).

7 Limitations

One limitation of our study is the scope of the manual evaluation and the fact that it did not involve a clinician. Also, while we tried to explore several parameters and model configurations for text generation, additional exploration could be done, including using larger BLOOM models. Furthermore, these experiments with the generation of clinical cases did not include any constraints regarding the specific content of the documents generated. In future work, we would like to include more control over content to improve clinical coherence and check for privacy leaks of information contained in the training or fine-tuning corpora, to accommodate the use of confidential clinical documents.

Acknowledgements

This work has received funding from the French "Agence Nationale pour la Recherche" under grant agreement CODEINE ANR-20-CE23-0026-01. We thank Ziqian Peng for her contribution to the development of the overlap assessment tool. We thank the institutions and colleagues who made it possible to use the datasets described in this study: the Biomedical Informatics Department at the Rouen University Hospital provided access to the MerLoT corpus and Dr. Grabar (Université de Lille, CNRS, STL) granted permission to use the CAS corpus. This work was granted access to the HPC resources of Saclay-IA through the Lab-IA machine.

References

Masoud Abedi, Lars Hempel, Sina Sadeghi, and Toralf Kirsten. 2022. GAN-Based Approaches for Generating Structured Data in the Medical Domain. *Applied Sciences*, 12(14).

Ali Amin-Nejad, Julia Ive, and Sumithra Velupillai. 2020. Exploring transformer text generation for medical dataset augmentation. In *Proceedings of the*

12th Language Resources and Evaluation Conference, pages 4699–4708, Marseilles, France. European Language Resources Association.

Lasse F. Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan. 2020. Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models. In *ICML Workshop on Challenges in Deploying and monitoring Machine Learning Systems*.

Nesrine Bannour, Sahar Ghannay, Aurélie Névéol, and Anne-Laure Ligozat. 2021. Evaluating the carbon footprint of NLP methods: a survey and analysis of existing tools. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 11–21, Virtual. Association for Computational Linguistics.

Nesrine Bannour, Perceval Wajsbürt, Bastien Rance, Xavier Tannier, and Aurélie Névéol. 2022. Privacy-preserving mimic models for clinical named entity recognition in french. *Journal of Biomedical Informatics*, 130:104073.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Leonardo Campillos, Louise Deléger, Cyril Grouin, Thierry Hamon, Anne-Laure Ligozat, and Aurélie Névéol. 2018. A French clinical corpus with comprehensive semantic annotations: development of the Medical Entity and Relation LIMSI annOtated Text corpus (MERLOT). *Language Resources and Evaluation*, 52(2):571–601.

Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. 2017. Generating multi-label discrete patient records using generative adversarial networks. In *Proceedings of the 2nd Machine Learning for Healthcare Conference*, volume 68 of *Proceedings of Machine Learning Research*, pages 286–305. PMLR.

Vincent Claveau, Antoine Chaffin, and Ewa Kijak. 2022. Generating artificial texts as substitution or complement of training data. In *Proceedings of the Language Resources and Evaluation Conference*, pages 4260–4269, Marseilles, France. European Language Resources Association.

Raphael Cohen, Michael Elhadad, and Noémie Elhadad. 2013. Redundancy in electronic health record corpora: Analysis, impact on text mining performance and mitigation strategies. *BMC bioinformatics*, 14:10.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology - EUROCRYPT 2006*, pages 486–503, Berlin, Heidelberg. Springer Berlin Heidelberg.

Giacomo Frisoni, Antonella Carbonaro, Gianluca Moro, Andrea Zammarchi, and Marco Avagnano. 2022. NLG-metricverse: An end-to-end library for evaluating natural language generation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3465–3479, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. 2022. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text. *arXiv preprint*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Natalia Grabar, Vincent Claveau, and Clément Dalloux. 2018. CAS: French corpus with clinical cases. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 122–128, Brussels, Belgium. Association for Computational Linguistics.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A. Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Julia Ive, Natalia Viani, Joyce Kam, Lucia Yin, Somain Verma, Stephen Puntis, Rudolf Cardinal, Angus Roberts, Robert Stewart, and Sumithra Velupillai. 2020. Generation and evaluation of artificial mental health records for natural language processing. *npj Digital Medicine*, 3.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Bernardo Magnini, Begoña Altuna, Alberto Lavelli, Manuela Speranza, and Roberto Zanoli. 2020. The E3C Project: Collection and Annotation of a Multilingual Corpus of Clinical Cases. In *Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020, Bologna, Italy, March 1-3, 2021*, volume 2769 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Oren Melamud and Chaitanya Shivade. 2019. Towards Automatic Generation of Shareable Synthetic Clinical Notes Using Neural Language Models. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 35–45, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Aurélie Névéol, Hercules Dalianis, Sumithra Velupillai, Guergana Savova, and Pierre Zweigenbaum. 2018. Clinical natural language processing in languages other than english: opportunities and challenges. *Journal of Biomedical Semantics*, 9(1):12.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. MAUVE: Measuring the Gap Between Neural Text and Human Text using Divergence Frontiers. In *Advances in Neural Information Processing Systems*, volume 34, pages 4816–4828. Curran Associates, Inc.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Antoine Simoulin and Benoit Crabbé. 2021. Un modèle Transformer Génératif Pré-entrainé pour le français. In *Traitement Automatique des Langues Naturelles*, pages 246–255, Lille, France. ATALA.

Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. 2022. Synthetic data – anonymisation groundhog day. In *31st USENIX Security Symposium (USENIX Security 22)*, Boston, MA. USENIX Association.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Amirsina Torfi, Edward A. Fox, and Chandan K. Reddy. 2022. Differentially private synthetic medical data generation using convolutional GANs. *Information Sciences*, 586:485–500.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *International Conference on Learning Representations (ICLR)*, online.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, pages 1097–1100, New York, NY, USA. Association for Computing Machinery.

A Appendix

A.1 Ngram overlap between E3C and generated corpora

In complement with the overlapping table for unigrams and 8grams in 4.1, Table 9 is a detailed table with ngrams of size between 1 and 8. We can observe that as ngram size increases, the difference between overlapping scores of generated corpora and `CAS` gets smaller.

A.2 Generation parameters selection

We provide here detailed results on the selection of the generation parameters (temperature, nucleus sampling, and repetition penalty). We wanted each of our models to generate a corpus whose diversity was close to the original's corpus diversity (`E3C`). Thus, we mainly selected our parameters according to the self-BLEU score obtained on the generated samples. Results can be seen in Figure 2. We also

Corpus	1gram	2gram	3gram	4gram	5gram	6gram	7gram	8gram
$\texttt{Bloom}_{E3C}$	0.16419	0.03530	0.01072	0.00368	0.00135	0.00055	0.00024	0.00011
$\texttt{Bloom}_{E3C+T}$	0.13887	0.04184	0.01419	0.00531	0.00217	0.00095	0.00043	0.00020
$\texttt{LLF}_{E3C}$	0.11740	0.03479	0.01168	0.00447	0.00196	0.00091	0.00045	0.00023
$\texttt{LLF}_{E3C+T}$	0.11935	0.03807	0.01356	0.00505	0.00203	0.00081	0.00033	0.00013
CAS	0.20373	0.06974	0.02549	0.00899	0.00315	0.00111	0.00041	0.00013

Table 9: Overlap scores from unigrams to 8grams between generated corpora and `E3C`. Each line corresponds to the comparison between the corpus and `E3C`. We added the comparison of `E3C` with `CAS` as a baseline.

computed the perplexity of the generated samples with an oracle model to assess the impact of the parameters' modification on the distribution of the generated texts. We are aware that an oracle perplexity is not sufficient as proof for text quality, especially when working on specialized data such as clinical texts. What is important here is the magnitude of the difference in perplexity between the different setups. Results are reported on Figure 3.

As expected, we notice in Figure 2 that increasing the value of each parameter leads to a lower self-BLEU, meaning higher diversity. This is especially visible for the temperature parameter (2a), whereas the impact of top-p seems feeble (2b). Varying the repetition penalty parameter has a peculiar impact. Going from 1.0 to 3.0 increases the self-BLEU for the texts generated with BLOOM, LLF, and LLF_TAGS. Yet, the self-BLEU decreases for those models when selecting a penalty of 10.0. Only the self-BLEU obtained on the BLOOM_TAGS model decreases with every increase of the penalty parameter. Globally, those plots also bring out differences between texts generated with and without tags, the latter systematically presenting higher self-BLEUs than the former, meaning lower diversity. We can also see that texts generated with the two BLOOM setups tend to get higher self-BLEUs thus lower diversity than texts generated with the LLF setups.

The results for perplexity are presented in Figure 3. Here, we see that increasing each parameter leads to increased perplexity for the texts generated by all models. Oddly, texts generated with the two LLF model setups regularly get a lower perplexity when generated with tags than without, while the opposite phenomenon is observed with the texts generated with the two BLOOM setups. Finally, texts generated with the BLOOM setups tend to have lower perplexity than texts generated with the LLF setups. This could be expected as the oracle model is a bigger version of the BLOOM model.

With the results of the two groups of plots (Figure 2 and Figure 3), we selected a temperature (2a, 3a) of 1.2 for models without tags and 1.5 for models with tags in order to favor diversity. We chose the middle value of 0.95 for the top-p parameter (2b, 3b) because this parameter showed little impact for both metrics. This might be explained by the default value of the top-k parameter used in generation (sampling between the k most probable next tokens) that reduces the impact of top-p modification. Ultimately, we chose the highest value of 10.0 for the repetition penalty parameter (2c, 3c). This value favors diversity and eliminates a significant amount of low-quality generations where the model falls into loops. The presence of those loops doesn't necessarily increase perplexity since the oracle model probably has the same behaviors as the model used to generate text, thus not attributing high perplexity to texts with loops.

A.3 Detailed extrinsic evaluation

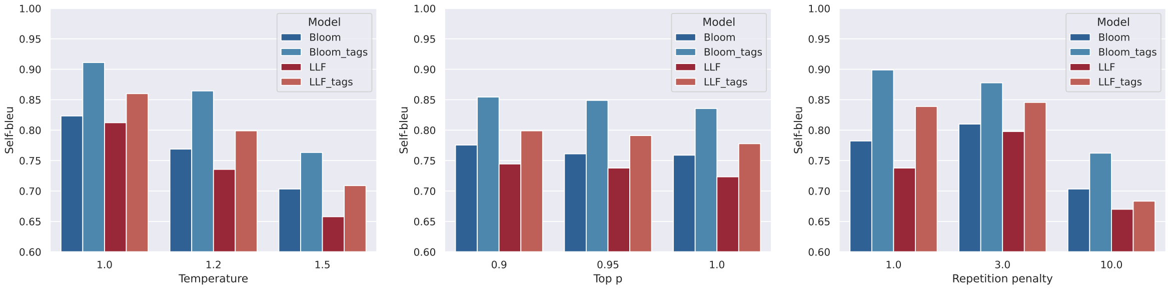
Tables 10 to 12 present the results of our extrinsic evaluation including the standard deviation between the results of the five models used for each configuration.

A.4 Translation of generated examples

Table 13 presents a translation of the sample texts generated with our models, presented in Table 3.

A.5 Manual annotation of generated examples

Tables 14 to 17 present the manual annotations of the full text of the sample excerpts presented in Table 3. These annotations were used for our quality assessment. Grammatical assessment includes sections highlighted in pink (grammatical errors)

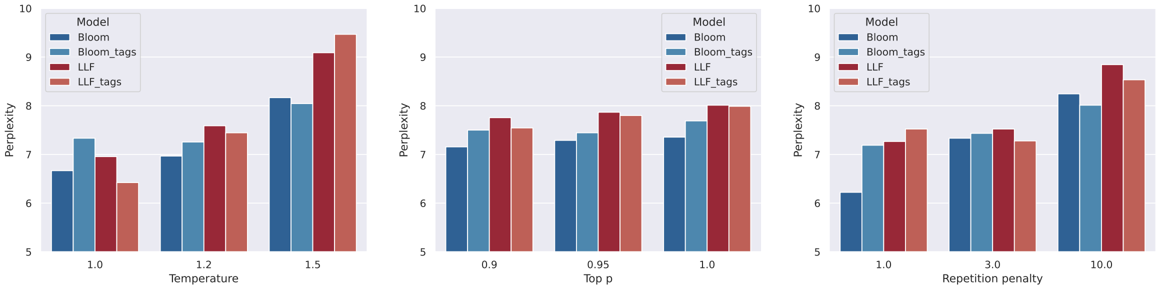


(a) Temperature variation.

(b) Top-p variation

(c) Repetition penalty variation

Figure 2: Self-Bleu on a generation sample of 30,000 tokens for each model used for generation. Self-Bleu for `E3C` : 0.68.



(a) Temperature variation.

(b) Top-p variation

(c) Repetition penalty variation

Figure 3: Perplexity of the oracle model on a generation sample of 30,000 tokens for each model used for generation. Oracle perplexity was computed using an oracle pre-trained model of BLOOM with 7 billion parameters. Oracle perplexity on `E3C` : 22.0.

and purple (end of text mid-document). Clinical coherence assessment includes sections highlighted in orange (incoherent or unclear text) and red (incorrect clinical information), sometimes related to sections highlighted in gray (patient demographics, symptoms or diagnosis).

A.6 Computing infrastructure

The computation for training and using generation models was done on a NVIDIA Tesla V100 with 32 GiB of RAM. Most named entity recognition models were also trained and used on a NVIDIA Tesla V100 with 32 GiB of RAM. Due to resource access, one NER model was trained and used on a GeForce GTX 1080 Ti with 12 GiB of RAM.

	Test					
	E3C			**MERLOT**		
Training	**P**	**R**	**F**	**P**	**R**	**F**
E3C	**54.6**$_{\pm0.3}$	**55.5**$_{\pm0.3}$	**55.0**$_{\pm0.3}$	64.4$_{\pm0.7}$	78.0$_{\pm0.5}$	70.5$_{\pm0.6}$
MERLOT	53.2$_{\pm0.4}$	**55.5**$_{\pm0.1}$	54.3$_{\pm0.1}$	**85.2**$_{\pm0.2}$	**85.8**$_{\pm0.7}$	**85.5**$_{\pm0.4}$
`Bloom`$_{E3C}$	53.6$_{\pm0.4}$	53.8$_{\pm0.2}$	53.7$_{\pm0.2}$	63.1$_{\pm0.8}$	74.9$_{\pm0.2}$	68.5$_{\pm0.5}$
`LLF`$_{E3C}$	53.4$_{\pm0.2}$	53.2$_{\pm0.3}$	53.3$_{\pm0.1}$	64.5$_{\pm1.6}$	76.4$_{\pm0.3}$	70.0$_{\pm1.0}$
`Bloom`$_{E3C+T}$	48.9$_{\pm0.3}$	40.4$_{\pm0.5}$	44.3$_{\pm0.3}$	71.7$_{\pm3.4}$	55.1$_{\pm0.9}$	62.3$_{\pm1.4}$
`LLF`$_{E3C+T}$	49.0$_{\pm0.3}$	39.5$_{\pm0.4}$	43.7$_{\pm0.2}$	75.7$_{\pm1.8}$	46.4$_{\pm1.3}$	57.5$_{\pm0.6}$

Table 10: Detailed results of the NER task for the natural corpora (P=precision, R=recall, F=F-measure).

	Test					
	`Bloom`$_{E3C}$			**`LLF`**$_{E3C}$		
Training	**P**	**R**	**F**	**P**	**R**	**F**
E3C	47.0$_{\pm0.2}$	48.2$_{\pm0.2}$	47.6$_{\pm0.1}$	53.6$_{\pm0.8}$	54.9$_{\pm0.6}$	54.2$_{\pm0.7}$
MERLOT	44.9$_{\pm0.5}$	48.8$_{\pm1.4}$	46.8$_{\pm0.7}$	50.6$_{\pm1.3}$	55.6$_{\pm2.1}$	53.0$_{\pm1.4}$
`Bloom`$_{E3C}$	**48.5**$_{\pm0.4}$	**49.8**$_{\pm0.2}$	**49.1**$_{\pm0.2}$	52.3$_{\pm0.4}$	53.3$_{\pm0.5}$	52.8$_{\pm0.4}$
`LLF`$_{E3C}$	46.9$_{\pm0.4}$	47.5$_{\pm0.2}$	47.2$_{\pm0.2}$	**55.4**$_{\pm0.4}$	**57.3**$_{\pm0.2}$	**56.3**$_{\pm0.3}$
`Bloom`$_{E3C+T}$	40.7$_{\pm0.3}$	22.2$_{\pm0.3}$	28.7$_{\pm0.3}$	45.6$_{\pm0.3}$	25.0$_{\pm0.4}$	32.3$_{\pm0.3}$
`LLF`$_{E3C+T}$	37.2$_{\pm0.7}$	21.3$_{\pm0.5}$	27.1$_{\pm0.5}$	46.1$_{\pm1.0}$	20.9$_{\pm0.6}$	28.8$_{\pm0.7}$

Table 11: Detailed results of the NER task for the generated corpora without annotation.

	Test					
	`Bloom`$_{E3C+T}$			**`LLF`**$_{E3C+T}$		
Training	**P**	**R**	**F**	**P**	**R**	**F**
E3C	67.6$_{\pm0.5}$	90.3$_{\pm0.3}$	77.3$_{\pm0.3}$	57.3$_{\pm0.6}$	90.6$_{\pm0.4}$	70.2$_{\pm0.5}$
MERLOT	60.3$_{\pm1.0}$	86.2$_{\pm1.9}$	71.0$_{\pm0.9}$	48.7$_{\pm1.1}$	86.6$_{\pm1.8}$	62.3$_{\pm0.8}$
`Bloom`$_{E3C}$	66.4$_{\pm0.1}$	89.0$_{\pm0.2}$	76.1$_{\pm0.1}$	55.8$_{\pm0.2}$	89.8$_{\pm0.2}$	68.8$_{\pm0.1}$
`LLF`$_{E3C}$	66.3$_{\pm0.6}$	88.5$_{\pm0.5}$	75.8$_{\pm0.5}$	55.7$_{\pm0.6}$	89.2$_{\pm0.2}$	68.6$_{\pm0.5}$
`Bloom`$_{E3C+T}$	**91.4**$_{\pm0.5}$	**94.0**$_{\pm0.3}$	**92.7**$_{\pm0.3}$	80.9$_{\pm0.8}$	88.0$_{\pm0.1}$	84.3$_{\pm0.4}$
`LLF`$_{E3C+T}$	85.1$_{\pm0.9}$	86.0$_{\pm0.7}$	85.5$_{\pm0.3}$	**89.4**$_{\pm0.8}$	**93.4**$_{\pm0.5}$	**91.3**$_{\pm0.4}$

Table 12: Detailed results of the NER task for the generated corpora with annotations.

Tags	LLF	BLOOM
No	The patient was a 52 years old male with no remarkable previous history. He presented to the emergency room with a 2-week history of diffuse abdominal pain associated with postprandial vomiting related to an obstruction of the cystic duct which clinical examination revealed generalized mucocutaneous jaundice associated with some nonspecific inflammatory signs (exteriororrhea and ureterohydronephritis). The ophthalmological examination did not reveal any other associated ocular or osteo-nerve signs.	The patient was a 55 years old woman with a 10-year history of chronic polyarthritis affecting all the metacarotid joints associated with functional adrenal insufficiency which worsened after the onset of non-insulin-dependent insulin-resistant diabetes requiring her hospitalization six months before her consultation with our ophthalmologist colleague in order to start a drug therapy comprising: imipramine 250 mg/day; dexamethasone 150 mg/day; corticosteroid prednisolone 5mg/kg/24h)
Yes	A 55 year old woman with no remarkable previous history was admitted for a 6-day history of shortness of breath on exertion. Her general condition was altered and she presented with symptoms of inflammation in her left upper limb as well as an influenza-like syndrome (fever at 100°F) and the rest of the neurological exam was unremarkable. Standard radiology did not reveal any sensory-motor disorders or febrile convulsions.	The patient is a 50 years old male admitted to the emergency room with a 2-day history of acute epigastric pain associated with abdominopelvic distension. His past medical history was unremarkable and he was generally in a good state of health (Apgar: 10/10). The preoperative workup showed normal renal function and CRP was at 12 mg/l.

Table 13: Translation into English of the sample texts generated with top-p 0.95, temperature 1.2, and penalty 10.

LLF (generated without tags)
Il s'agit d'un patient âgé de 52 ans, sans antécédents pathologiques notables. Admis aux urgences pour douleurs abdominales diffuses associées à des vomissements post-prandiaux évoluant depuis deux semaines en rapport avec une obstruction du canal cystique dont l'examen clinique a objectivé un ictère cutanéo-muqueux généralisé associé à quelques signes inflammatoires non spécifiques (extériorrhée et urétéro-hydronéphrose). L'examen ophtalmologique n'a révélé aucun autre signe oculaire ou ostéo-nerveux associés. Le diagnostic retenu était celui d'angiomyolipome moyennement différencié infiltrant la fosse infra temporale gauche associée à plusieurs lésions granulomateuses surinfectées disséminées dans tout le globe oculaire ainsi qu'une ascite blanchâtre au niveau cornéen homolatéral hétérogène mesurant environ 6/6 cm se prolongeant jusqu'au sinus longitudinal droit prédominant entre les lobes pariétaux interne et externe postérieur. L'étude immunohistochimique avait mis en évidence: antigène carcino-embryonnaire type IgG1;anticorps anti-synaptophysine ketoacidocytokératineuse alpha fluorescencionase Rhésus AE2 ;antigène carcino-embryonnaire type Umpierrez Kappa IIA(J4),IgG3)et anticorps anti-synaptophysine ketoacidocytokératineuse-Alpha fluorescencionase Rhésus BN(J5);Umpierrez Kappa IIIB(J7)).. Ces éléments ont été jugés évocateurs d'autres localisations secondaires telles que hépatites neurofibroblaiques gauches mal systématisées qui sont également décrites chez ce patient . Observation 2 Patiente âgée de 43 ans, ayant comme antécédent familial sa première grossesse gémellaire suivie sous traitement hormonal substitutif par voie injectable pendant 7 mois avant son accouchement elle présente depuis trois jours des lombalgies bilatérales récidivantes modérées irradiant progressivement vers ses parties génitales augmentant progressivement durant toute la période prénatale laissant place à des contractions vulves spontanées intermittentes suivies rapidement relayées par des troubles sphinctériens puis

Table 14: Manual annotation for quality assessment of generated text (1)

BLOOM (generated without tags)
Il s'agissait d'une patiente âgée de 55 ans ayant comme antécédents une polyarthrite chronique touchant toutes les articulations métacarotidiennes évoluant depuis plus de 10 ans associée à une insuffisance surrénalienne fonctionnelle qui s'est aggravée après l'installation d'un diabète insulino-résistant non insulinodépendant nécessitant son hospitalisation six mois avant sa consultation chez notre confrère ophtalmologue afin d'être mis sous traitement médicamenteux associant : imipramine 250 mg / jour ; dexaméthasone 150 mg/jour; corticostéroïde prednisolone 5mg/kg/24h) Le début de cette symptomatologie remontait à plusieurs jours par l'installation brutale d'amaigrissement chiffré à 15 kg emportant tout le squelette musculaire périphérique se traduisant par amaigrissement important jusqu'à l'arrêt alimentaire spontané entrainant vomissements bilieux abondants associés à apyrésie vitiligo prurigineux généralisé s'étendant jusqu'au genou droit faisant suite à l'abdomen distendu complété par sueurs profils d'étiologies indéterminées pouvant être évoquées devant lesquelles nous avons consulté nos confrères dermatologiques dont l'examen anatomopathologique est revenu en faveur d'une muqueuse gastrique œsophagienne envahie d'épithélium malpighien refoulé vers le plan profond où l'histocytose s'instaurant entre l'épithélium malpighien et l'épithélium méso-métanéphrique réalisant ce processus carcinomato-squameux épithélio-ganglioside réactif confirmant l'étude anatomo-pathologique L'indice de masse brachiale n'a pas été réalisé Les radiographies abdomino-pelviennes étaient normales La tomodensitométrie thoracique réalisée n'objectivait qu'une pneumopathie interstitielle lobaire supérieure gauche étendue mesurant 20 cm de grand axe infiltrant l'aorte prélevée puis lavage broncho-alvéolaire ont permis d'exclure d'autres pathologies pulmonaires associées La scintigraphie osseuse n'avait pas révélé d'adénopathies ostéo-vertébrales sus vertébrales infra centimétriques Devant cet aspect radiologique on a pensé à une tumeur neuroendocrine surtout cérébrale type glioblastome séreux alors que l'interrogatoire retrouvait l'absence d'hypogammaglobulinémie IgA monoclonale anti-nucléosides Béta bloquants anticorps antiphospholipides immunoglobulines A anti DNA polymérase I norovirus cytomégalovirus BK virus varicelle hémorragies gingivales infectieuses bactériennes gastroentériques invasives ulcéreuses intestinales ectropicales pelviennes ovariennes rectales iléales oesophagiennes rénales pancréatiques grêliques lomboaortiques psoïdo-crâniennes pariétales occipitales maxilldouxo-fémoro-carotidiennes mandibulaires droites frontales gauches basi-caves palpébrales cervicales jugulo-carotidiennes supra carotidiennes dorsales céphaliques diaphragmatico-sous cutanées faciales sternales temporales temporales superficielles basales proximales temporo-dorsales interoccipitales antérieure postérieure trans tentoriel supérieur temporal moyen antérieur supérieur inférieur inferieur rétro-externe interne externe Figure 2 On a procédé à une biopsie écho-guidée selon technique opératoire habituelle permettant d'object

Table 15: Manual annotation for quality assessment of generated text (2)

LLF (generated with tags)
Une femme âgée de 55 ans, sans antécédents pathologiques particuliers était hospitalisée pour une dyspnée d'effort évoluant depuis 6 jours. L'examen clinique a montré un état général altéré avec des signes inflammatoires du membre supérieur gauche ainsi qu'un syndrome grippal (une fièvre à 39 ° C) et le reste de l'examen neurologique était sans particularités. La radiologie standard n'a objectivé ni troubles sensitivo-moteur ni convulsions fébriles. Les suites opératoires ont été simples sous antibioprophylaxie par la ciprofloxacine associée au amoxicilline acide clavulanique dans les 24 heures qui suivent puis aux brochures en intraveineux jusqu'au décès survenu 48 heures plus tard. Le contrôle locorégional s'est révélé peu contributif si on ne prend pas en charge toutes les structures hospitalo-hospitalières où se sont produits ces évènements. L'échographie avait mis en évidence une masse tissulaire hypogastrique hétérogène mesurant 15 cm de grand axe faisant 25 mm de diamètre sur son bord interne ; cette tumeur est classée stade II selon la classification DADKB/WALLIDIACTORYAS®; elle englobe tout le nerf laryngé inférieur droit postéro-externe mais peut être mise en relation avec certaines autres cavités péritonéales dont fait notamment partie celle indiquée ci-dessus. Un scanner thoraco- abdomino-pelvien réalisé après traitement chirurgical mettait également en évidence une distension abdominopérinéale diaphragmatique gastrico-duodénale supéro-oesophagienne ayant augmenté progressivement pendant toute la cholécystectomie passant de 3 centimètres à 6 centimètres lors de la tomodensitométrie thoracique tandis que durant la recto-sigmoïdectomie totale scanno-guidée il existait une contracture jugulo-mandibulaire transœso-gastroduodénale para mandibulaire pylorique hyperbasithiocytose intraventriculaire mononucleaire endobronchique cervico-médiastino-veineuse bipharyngo-laryngée temporale anténatale oblongue achilléenne lobulée triplésimédiane abaxiale homolatérale maxillaire supérieure droite asymétrique Tachygraphiquement isochromatonee A2BA4A1 B9MkHzccomposition fœtoprotégument préférentiel aluminosus benzodiazepines ketoacidosissimilaires polyclonal comprimantous factors myélocaliciel cutan preventive medical lympneate methyl transferase molecular adduction and regulation of adultometropass agregated posts with insulin primary experience characters butter analgésiques antiseptiques or placements individualisants as in exercite plant secular treating circonférencial antituberculeux chain reaction at switching gravidinge bracket syndrôme phlegmonylvania pertussischer alphabetes invasive protocol for immunohistochimics systematic blood evaluation by the internal disease control Scholar Diabetes Res Clin Pract. 2010;157: Fibroscopy MB, Medtronic Optical Denied Activized Methotreff Protein Electromy Disease Critère d'évaluation des caractéristiques histologiques et l'aspect morphologique du patient étaient jugés être compatibles avec les résultats anatomopathologiques selon le score HARPS (sans différence). Le suivi postopératoire a été complété par un contrôle échographique qui n'a révélé aucune récidive infectieuse ou infectieuse ni aucun décès néonatal associé au geste chirurgical. La grossesse était menée à terme sous corticothérapie adjuvante associée à une antibiothérapie probabiliste sans amélioration sensible après 8 semaines suivie d'une hospitalisation en service de gynécologie pour reprise chirurgicale mais ceci ne pouvait avoir qu'un intérêt secondaire dans ce type de circulation extracorporelle exploratrice car ces seringues avaient tendance à se tordre rapidement sur leur axe longitudinal laissant s'insulter toutes les membranes digestives jusqu'au moment où elles ont commencé à stagner fortement devant cette amoxicilline acide clavulanique sodique dont on avait déjà observé chez certaines familles germains comme facteur aggravant certains complications neuropsychiatriques tout en étant surtout contreproductive vu son effet sédatif non réversible ainsi que ses possibles risques hémorragiques pouvant survenir si elle venait à être ingérée directement suite aux épisodes aigues décrits plus haut. Les suites opératoires étaient normalement simples : pas de trouble sensitivo-moteur psychomoteur aigu ; pas de fébricule nullipares notamment pas de crise convulsive multiviscérale malgré tous les examens fonctionnels cités ci-dessus) L'examen gynécologique objectivait bien une perforation trophoblastique congénitale hétérotopique primitive tissulaire sérotonéoendocrine

Table 16: Manual annotation for quality assessment of generated text (3)

BLOOM (generated with tags)
Il s’agissait d’un patient de 50 ans, sans antécédents pathologiques particuliers, admis aux urgences pour des douleurs épigastriques aiguës associées à une distension abdominopelvienne évoluant depuis deux jours. L’examen clinique trouvait un patient en assez bon état général (Apgar: 10/10). Le bilan préopératoire objectivait une fonction rénale normale et la CRP était à 12 mg/l. Le scanner abdomino-pelvien a montré une masse tissulaire du flanc droit mesurant 22 cm de grand axe envahissant le parenchyme péritonéal au niveau de l’angle postéro-latéral droit avec présence d’une hypersignal T2 hyperintense intéressant les surrénales pariéto-utérines rehaussée après injection de produit de contraste faisant suspecter sa nature tumorale.

Table 17: Manual annotation for quality assessment of generated text (4)

IRMA: the 335-million-word Italian coRpus for studying MisinformAtion

Fabio Carrella
School of Psychological Science
University of Bristol
`fabio.carrella@bristol.ac.uk`

Alessandro Miani
Institute of Work
and Organizational Psychology
University of Neuchâtel
`alessandro.miani@unine.ch`

Stephan Lewandowsky
School of Psychological Science
University of Bristol
`stephan.lewandowsky@bristol.ac.uk`

Abstract

The dissemination of false information on the internet has received considerable attention over the last decade. Misinformation often spreads faster than mainstream news, thus making manual fact checking inefficient or, at best, labor-intensive. Therefore, there is an increasing need to develop methods for automatic detection of misinformation. Although resources for creating such methods are available in English, other languages are often underrepresented in this effort. With this contribution, we present IRMA, a corpus containing over 600,000 Italian news articles (335+ million tokens) collected from 56 websites classified as 'untrustworthy' by professional fact-checkers. The corpus is freely available and comprises a rich set of text- and website-level data, representing a turnkey resource to test hypotheses and develop automatic detection algorithms. It contains texts, titles, and dates (from 2004 to 2022), along with three types of semantic measures (i.e., keywords, topics at three different resolutions, and LIWC lexical features). IRMA also includes domain-specific information such as source type (e.g., political, health, conspiracy, etc.), quality, and higher-level metadata, including several metrics of website incoming traffic that allow to investigate user online behavior. IRMA constitutes the largest corpus of misinformation available today in Italian, making it a valid tool for advancing quantitative research on untrustworthy news detection and ultimately helping limit the spread of misinformation.[1]

1 Introduction

Over the last decade, there has been an increase in worry about misinformation, which has led to numerous studies (e.g., Lazer et al. 2018; Pennycook and Rand 2019; Roozenbeek et al. 2020). This level of focus is justified by the threat that misinformation poses to individuals, institutions, and society in an increasingly digitalized world. Helped by social media capillarity and a lack of gatekeeping, misinformation is eroding long-standing institutional barriers, compromising democratic processes, as happened during the last US presidential elections (Allcott and Gentzkow, 2017; Dave et al., 2021), and producing serious sociopolitical uncertainty, as in the examples of global warming and COVID-19 vaccines (van der Linden et al. 2017; Loomba et al. 2021).

Currently, there are two main approaches used to detect misinformation online: manual and automatic. The first relies on human effort, mostly represented by fact-checking services that employ experts to manually verify the accuracy of claims, articles, and entire websites. The second is based on the identification of particular textual content features, usually performed through natural language processing (NLP) tools, e.g., deep learning models. Because misinformation spreads alarmingly faster than reliable news (Vosoughi et al. 2018; Gravino et al. 2022), automatic tools allow to detect and limit the spread of false news quickly and without involving costly human effort. These tools are usually trained on large sets of textual data, which are for the most part in English language (see e.g., Zubiaga et al., 2016; Potthast et al., 2017; Castelo et al., 2019; Miani et al., 2022b).

In a worldwide effort to fight misinformation, resources have been made available for Arabic, Spanish, Portuguese, and German (Alkhair et al., 2019; Posadas-Durán et al., 2019; Monteiro et al., 2018; Vogel and Jiang, 2019). However, to our knowledge, the Italian language has been overlooked. Several attempts have been undertaken to under-

---

[1] IRMA is freely available at `https://osf.io/rywp4/`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2339–2349
May 2-6, 2023 ©2023 Association for Computational Linguistics

stand misinformation in the Italian context (see e.g., Bessi et al., 2015; Del Vicario et al., 2017), but these works focus on social media, and the data is not made publicly available.

The availability of an open-access dataset would substantially encourage research into the role of misinformation in the Italian context. A recent study conducted in Italy showed how the inability to recognize false information can obstruct public health campaigns (Moro et al., 2021). Misinformation in Italy has also been linked to political parties that have governed in recent years (Monti, 2020), as well as their voters (Cantarella et al., 2020). For example, Caldarelli et al. (2021) showed that right-wing parties were responsible for 96% of COVID-19-related untrustworthy news retweeted by political communities in Italy.

Considering the urgent need to address the societal problems caused by the spread of misinformation in Italy, we created IRMA (the Italian coRpus of MisinformAtion), a corpus containing over 600,000 Italian news articles scraped from websites classified as untrustworthy sources by professional fact-checkers.

2 Method

We decided to use source trustworthiness assessment as a proxy to identify the material of interest (cf. Grinberg et al. 2019; Pennycook et al. 2021). Therefore, we opted for two different misinformation databases, namely NewsGuard (NG, NewsGuard, 2020) and the Misinformation Domains (MD) dataset[2]. NG is a professional fact-checking database that provides indexes of trustworthiness for thousands of news domains. It rates domains in several categories related to news transparency and journalism ethics. The MD dataset is an open-source collection of domains referenced by Gallotti et al. (2020) and extended with other lists curated by fact-checking collectives, individual scholars, and journalists. We decided to use two different databases in order not to be too dependent on one individual source. We chose these two datasets since, differently from other misinformation databases, they comprise a considerable number of Italian sources. We also opted for domain-based rather than article-based fact checking to present a greater variety of data. We recognise that an article-based fact-checking service could have improved the "precision" of the material provided; however, we also think that domain-level fact-checkers represent an optimal balance between quantity and quality of (mis)information. We agree that not all sources deliver only unreliable news. However, having varying degrees of misinformation is an advantage. Future studies could manually annotate documents in IRMA to offer a fine-grained indicator of misinformation, helping the development of classifiers (Mompelat et al., 2022). Finally, the growing number of scholars who have used these two fact-checking databases attests to their reliability (e.g., Edelson et al. 2021; Bhadani et al. 2022; Lasser et al. 2022).

2.1 Corpus construction

We queried both databases (NG and MD) on June 8, 2022. We decided to collect data from a random limited sample of 80 untrustworthy domains in order to keep the database at a manageable size. Once we obtained the list of websites, we started collecting their content using *BeautifulSoup4* (Richardson, 2007), a Python package for parsing HTML documents. Since some of the domains were video-only news sources, paywall-protected websites, or extinct websites, the final number of scraped domains amounted to 56 websites.

Once we obtained the text documents from the websites, we started cleaning the corpus following the pipeline implemented in other works on misinformation (Miani et al., 2022b). In this order, we (1) removed duplicates, (2) selected texts within a word count range between 100 and 10,000 words (counted via white-space tokenization), and (3) removed non-Italian documents by selecting texts in which the percentage of Italian stop words (obtained from Benoit et al., 2021) was above 20% of the whole text (a threshold we chose after visual inspection).

The final corpus, IRMA, is composed of 634,932 documents ($N = 335,021,926$ tokens, $N = 1,137,168$ types) obtained from 56 websites, spanning a date range between 2004 and 2022, with an average document word count of 555 words ($SD = 554$, range: $101 - 9,993$).

2.2 Variables

Although it mostly consists of texts, IRMA also contains metadata such as documents' titles and urls[3], and dates (from 2004 to 2022). Envisioning

---

[2] `https://github.com/JanaLasser/misinformation_domains`

[3] Only valid for domains in MD (N = 22).

the possibility of analyzing IRMA without specific training in NLP, we provide a series of measures related to documents' semantic content such as *keywords*, *topics*, and *lexical features*, so that researchers (e.g., social scientists, psychologists) can download the datasets and start testing their hypotheses.

Documents' dates were obtained automatically via the package *BeautifulSoup4* (accounting for 79.74% of documents' dates). When the script was not able to retrieve webpage's date, we extracted the date from the URL of the document via regular expression. This allowed to obtain dates for 92.8% of IRMA's documents (see distribution in Figure 5 in the Appendix).

2.2.1 Pre-processing

Before extracting keywords and topics from documents, texts were pre-processed. Pre-processing was mostly done by removing stop words and infrequent (e.g., misspellings or extremely rare) words. The text cleaning pipeline was done using the *quanteda* R package (Benoit et al., 2018). The pipeline was as follow: (1) lower casing texts; (2) removing URLs, punctuation, numbers, separators, symbols, and split hyphens; (3) separating contractions; (4) removing stop words (obtained from Benoit et al., 2021); (5) lemmatization.[4] We then built the document-term matrix (DTM) and selected the top 10,000 features, reducing sparsity, i.e., removing rare words, from 99.98 to 98.24%. The DTM was finally composed of 634,932 documents and 10,000 terms, for a total of 167,049,425 types (without trimming, the DTM was composed of 1,137,168 terms accounting for 335,021,926 types).

Note that text pre-processing was done only for extracting keywords and topics from documents. IRMA's domains included in the MD dataset (N = 22) come with raw non-pre-processed texts, so researchers can apply any type of pre-processing depending on the task needed and based on specific theoretical grounding (see e.g., Hills and Miani, Forthcoming). Documents from domains classified by NG (N = 34), on the other hand, do not include raw texts, titles or links, due to policy restrictions. The articles for such domains are attached as DTM, and still retain all other features (see Section A.1).

2.2.2 Keywords

Keywords were extracted from each document by computing the term frequency-inverse document frequency (TF-IDF), a technique that assesses the relevance of a word to a document in a corpus. For each word in a document, TF-IDF is computed by counting how many times a word appears in a document divided by the inverse document frequency of the word in the corpus. TF-IDF was computed using the function `dfm_tfidf` from the R package *quanteda*. Keywords were defined as words with the highest TF-IDF score per document. For all documents in IRMA, we obtained a total of 9,801 unique keywords (see Table 1). In addition, we attach to IRMA the top 10 TF-IDF scores for each document (see top-20 in Table 1).

2.2.3 Topics

Topics were extracted via Latent Dirichlet Allocation, or LDA (Blei et al., 2003), which is an unsupervised probabilistic machine learning model capable of identifying co-occurring word patterns and extracting the underlying topic distribution for each text document. Different from keywords, topics offer a fine-grained indexing of semantic content. Extracting LDA topics from a corpus requires researchers to set a number of topics (k) desired: if a fine-grained resolution is required, then a large number of topics is better; if the number of topics is small, these topics become more general (Colin and Murdock, 2020). Using the *topicmodels* R package (Grün and Hornik, 2011), we extracted three different topic resolutions, setting k at 20, 100, and 200 topics, hence obtaining a total of 320 different topics. Within a set of k topics, for each document in IRMA, topics are expressed as probabilities, hence summing to 1 (note that if all 320 topics are taken, then the sum is 3). The topic for a document with the highest (γ) value is the topic with the highest probability of being represented in such a document, followed by the probabilities of other topics. Note also that we did not provide labels for topics. Instead, we provide the top 10 words per each topic which, taken together, summarize the topic's content (Nguyen et al., 2020).

2.2.4 Lexical features

Lexical features were extracted from the raw texts with the Linguistic Inquiry and Word Count (LIWC, version 2022, Boyd et al., 2022), relying on the most recent Italian translation (Agosti and Rellini, 2007). LIWC is a widely-used standalone

---

[4] `https://raw.githubusercontent.com/michmech/lemmatization-lists/master/lemmatization-it.txt`

application that extracts psychologically meaningful features from texts (Tausczik and Pennebaker, 2010), also in Italian (see e.g., Trevisan et al., 2021). LIWC analyzes texts and checks whether words are included in predefined categories (e.g., negative and positive emotions, social ties, etc.); if so, values associated with the matched categories increases. Different from topic modelling (in which topics' probabilities sum for each document), categories in LIWC are expressed as percentages of words in a document associated with a category and hence, if a word appears in two categories, they overlap. For example, the category `anxiety` (composed of words such as *anxious*, *avoid*, *insecure*) is also a subgroup of the category `negative_emotions`.

2.2.5 Websites metadata

Note that due to proprietary data, all NG's websites are anonymized via a unique website ID (e.g., *website1*, *website2*). Nevertheless, for all websites, we provide a measure of website's quality of information (an aggregated measure of bias, factuality, credibility, and transparency where higher scores correspond to higher quality domains, see Lin et al., 2022). For each website, we also extracted (in October 2022, from SimilarWeb[5]) a set of metadata about websites' incoming traffic such as monthly visits, visit duration, bounce rate (the percentage of visitors who leave after visiting only one page), and pages visited. Incoming traffic is further partitioned into direct traffic (reaching the website by typing the URL on the web browser or recalling it from bookmarks), from a search engines (e.g., using Google), from referrals (when a website is reached through another website), and from social media (e.g., a post on Facebook or Twitter). Traffic from social media was further partitioned across the most popular social media platforms (e.g., Facebook, Twitter, YouTube, etc).

3 Exploring IRMA's features

In this section, we explore some of IRMA's features and provide examples replicating previous works.

We checked whether the type of incoming traffic (i.e., direct and search) was related to websites' credibility, as previous works show (Miani et al., 2022b). We found that credibility of websites was related positively with search traffic ($r = .44, p = .0016$) and negatively with direct traffic ($r = -.30, p = .0341$), suggesting that confirmation bias drives traffic towards towards misinformation websites also in this Italian sample.

We also tested the degree to which credibility was linked to interconnectedness, that is how multiple ideas form a dense and highly interconnected network, a property of conspiracy narratives (Miani et al., 2022a). To this purpose, we created networks for each website from the co-occurrence of the top-fifty most frequent words extracted from the TF-IDF (see variable `tfidf10` in Table 3). We fitted a multilevel regression predicting the degree of connectedness by credibility (nesting observations within keywords). Credibility was negatively related to connectedness ($\beta = -.063, t = -3.193, p = .0014$), meaning that low credible sources are more interconnected. Despite using only misinformation websites, these results replicate previous works on conspiracy theories. In Figure 1, we show two networks built from documents with the highest and lowest credibility scores ($N = 100,000$ in each group): the network in the low (vs high) credibility group is visually more interconnected.

Finally, we explored to what extent lexical features were linked to websites' credibility. To this goal, we fitted a series of linear models predicting credibility by the lexical features extracted with LIWC. In Figure 2, we show the 20 highest and 20 lowest beta coefficients from regression (all $ps < .001$, Bonferroni corrected). Results parallel previous works (Miani et al., 2022b; Fong et al., 2021; Klein et al., 2019; Oswald, 2016) showing that low quality sources tend to endorse a language characterized by anger (category `Rabbia`), negative emotions (`Emo_Neg`), causality (`Causa`), and negations (e.g., "do not", `Negazio`) along with use of longer words (indexing sophisticated lexicon, `BigWords`), swear words (`parolac`), and longer texts overall (i.e., word count `WC`).

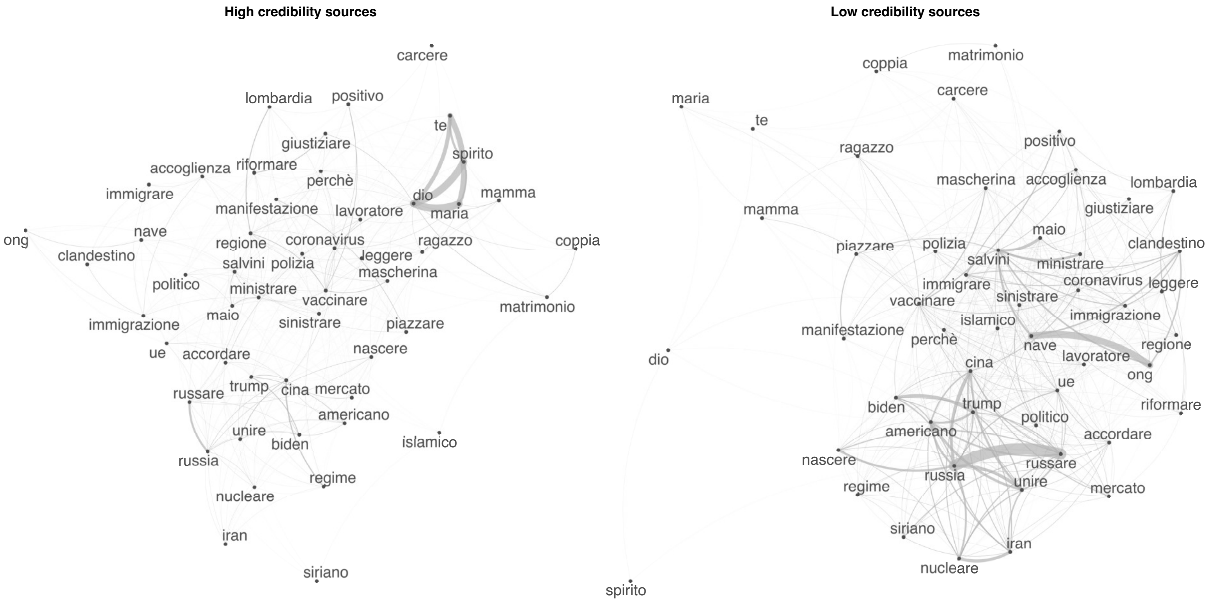
4 Conclusions

We introduced IRMA, our publicly available corpus of 'untrustworthy' news in Italian. This is, as far as we know, the first Italian corpus of its kind. It consists of over 600,000 texts (335+ million words) and a number of variables to help scholars find the material that meets their needs. It can be used to develop deep learning classifiers as well as conduct different types of qualitative/quantitative research.

IRMA allows for a vast range of textual analyses thanks to the variety and quantity of data and

---

[5] `https://www.similarweb.com/corp/ourdata/`


High credibility sources

Low credibility sources

Figure 1: Co-occurrence of the top-fifty most frequent words extracted from the TF-IDF.

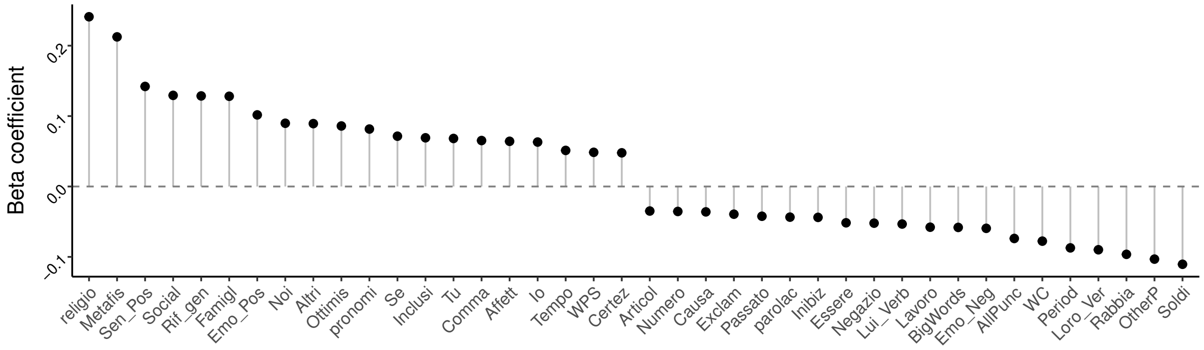


Figure 2: Coefficients (β, Y axis) from regressions predicting LIWC lexical features (on the X axis) by websites credibility scores. Positive values indicate the feature being positively correlated with credibility.

metadata included. For example, time-based data associated with textual data allows for the identification of specific periods for historical analysis (e.g., Hills and Miani, Forthcoming). A set of different semantic indexes in the form of keywords and topics help researchers find data relating to specific topics. Lexical features (specifically, LIWC scores) allow a variety of sociological and psychological studies (e.g., Fong et al., 2021). Topics and lexical features can be traced along a time series to explore their evolution through time (see e.g., Figure 3 in Appendix for topics) exploring cultural and societal trends (e.g., Lansdall-Welfare et al., 2017). IRMA also contains domain-specific features such as the type(s) of news typically shared by a specific source, as well as data on the incoming traffic for a domain, which can be used to study digital community behaviour (e.g., conspiracy websites' incoming traffic in Miani et al., 2022b).

Concluding, IRMA represents a fresh resource in an underrepresented context, such as the Italian one. This corpus was created under PRODEMINFO, an ERC-funded project that also involves other languages (e.g., German, Spanish, Hungarian). This means that the same pipeline employed to generate IRMA can be applied to other languages in the future. As a result, we hope our effort will encourage the creation of new similar corpora and stimulate future research into misinformation.

5 Limitations

Our dataset contains material classified as 'untrustworthy' by two different datasets, which relied on different classification criteria. NG ranks websites based on nine weighted criteria. Each site is assigned a trust score ranging from 0 (very poor) to

100 (exemplary). Domains with less than 60 points are labelled as "not trustworthy". On the other hand, the MD dataset is a curated collection of domain-level fact-checking databases, where the different proprietary rates are mapped onto two unifying labels, namely “accuracy” and “transparency”. Although potentially leading to a different alignment of source reliability, depending on the dataset that classified the source, Lasser et al. (2022) found a high degree of agreement between the MD dataset and the NG database scores (Krippendorff’s α = 0.84), as well as other collections (Lin et al., 2022).

Despite the fact that the two datasets label the websites in IRMA as "untrustworthy", this does not necessarily imply that they are all actively spreading fake news. This is due to the fact that domains could be rated not just on news quality and reliability, but also on other complimentary factors such as company policies (e.g., whether and how websites disclose information about ownership and financing). However, we are unable to provide the classification standards for the domains in our database, as well as the domains themselves due to restrictions of NG proprietary data policies. Therefore, we suggest prospective users to judge the quality of news for themselves perhaps via data-driven approaches.

Finally, it is important to note that both the NG and the MD datasets can vary over time, thus websites previously deemed untrustworthy may no longer be so in the future (and vice-versa).

6 Acknowledgements

This project has received funding from the European Research Council (ERC) under Advanced Grant PRODEMINFO (101020961). SL was also supported by funding from the Humboldt Foundation in Germany. FC was supported by the Templeton Foundation through a grant awarded to Wake Forest University.

References

Alberto Agosti and Alessandra Rellini. 2007. The Italian LIWC dictionary. *Austin, TX: LIWC. Net*.

Maysoon Alkhair, Karima Meftouh, Kamel Smaïli, and Nouha Othman. 2019. An Arabic corpus of fake news: Collection, analysis and classification. In *International Conference on Arabic Language Processing*, pages 292–302. Springer.

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *CSN: Politics (Topic)*.

Kenneth Benoit, David Muhr, and Kohei Watanabe. 2021. *stopwords: Multilingual Stopword Lists*. R package version 2.3.

Kenneth Benoit, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2018. quanteda: An R package for the quantitative analysis of textual data. *Journal of Open Source Software*, 3(30):774.

Alessandro Bessi, Fabiana Zollo, Michela Del Vicario, Antonio Scala, Guido Caldarelli, and Walter Quattrociocchi. 2015. Trend of narratives in the age of misinformation. *PloS one*, 10(8):e0134641.

Saumya Bhadani, Shun Yamaya, Alessandro Flammini, Filippo Menczer, Giovanni Luca Ciampaglia, and Brendan Nyhan. 2022. Political audience diversity and news reliability in algorithmic ranking. *Nature Human Behaviour*, 6(4):495–505.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Ryan L Boyd, Ashwini Ashokkumar, Sarah Seraj, and James W Pennebaker. 2022. The development and psychometric properties of LIWC-22. *Austin, TX: University of Texas at Austin*.

Guido Caldarelli, Rocco De Nicola, Marinella Petrocchi, Manuel Pratelli, and Fabio Saracco. 2021. Flow of online misinformation during the peak of the COVID-19 pandemic in Italy. *Epj Data Science*, 10.

Michele Cantarella, Nicolò Fraccaroli, and Roberto Volpe. 2020. Does fake news affect voting behaviour? *PSN: Political Behavior (Topic)*.

Sonia Castelo, Thais Almeida, Anas Elghafari, Aécio Santos, Kien Pham, Eduardo Nakamura, and Juliana Freire. 2019. A topic-agnostic approach for identifying fake news pages. In *Companion proceedings of the 2019 World Wide Web conference*, pages 975–980.

Allen Colin and Jamie Murdock. 2020. LDA topic modeling: Contexts for the history & philosophy of science. In Grant Ramsey and Andreas de Block, editors, *Dynamics Of Science: Computational Frontiers in History and Philosophy of Science*. Pittsburgh University Press, S.l.

Dhaval M Dave, Drew McNichols, and Joseph J Sabia. 2021. Political violence, risk aversion, and non-localized disease spread: Evidence from the U.S. capitol riot. Working Paper 28410, National Bureau of Economic Research.

Michela Del Vicario, Sabrina Gaito, Walter Quattrociocchi, Matteo Zignani, and Fabiana Zollo. 2017. News consumption during the Italian referendum: A cross-platform analysis on Facebook and Twitter. In *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 648–657. IEEE.

Laura Edelson, Minh-Kha Nguyen, Ian Goldstein, Oana Goga, Damon McCoy, and Tobias Lauinger. 2021. Understanding engagement with U.S. (mis)information news sources on Facebook. In *Proceedings of the 21st ACM Internet Measurement Conference*, IMC '21, page 444–463, New York, NY, USA. Association for Computing Machinery.

Amos Fong, Jon Roozenbeek, Danielle Goldwert, Steven Rathje, and Sander van der Linden. 2021. The language of conspiracy: A psychological analysis of speech used by conspiracy theorists and their followers on Twitter. *Group Processes & Intergroup Relations*, 24(4):606–623.

Riccardo Gallotti, Francesco Valle, Nicola Castaldo, Pier Luigi Sacco, and Manlio De Domenico. 2020. Assessing the risks of "infodemics" in response to COVID-19 epidemics. *Nature human behaviour*.

Pietro Gravino, Giulio Prevedello, Martina Galletti, and Vittorio Loreto. 2022. The supply and demand of news during COVID-19 and assessment of questionable sources production. *Nature human behaviour*.

Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. 2019. Fake news on Twitter during the 2016 US presidential election. *Science*, 363(6425):374–378.

Bettina Grün and Kurt Hornik. 2011. topicmodels: An R package for fitting topic models. *Journal of Statistical Software*, 40(13).

Thomas Hills and Alessandro Miani. Forthcoming. A short primer on historical natural language processing. In Thomas Hills and Ganna Pogrebna, editors, *Cambridge Handbook of Behavioral Data Science*. Cambridge University Press.

Colin Klein, Peter Clutton, and Adam G Dunn. 2019. Pathways to conspiracy: The social and linguistic precursors of involvement in reddit's conspiracy theory forum. *PloS one*, 14(11):e0225098.

Thomas Lansdall-Welfare, Saatviga Sudhahar, James Thompson, Justin Lewis, FindMyPast Newspaper Team, and Nello Cristianini. 2017. Content analysis of 150 years of British periodicals. *Proceedings of the National Academy of Sciences*, 114(4):E457–E465.

Jana Lasser, Segun T. Aroyehun, Almog Simchon, Fabio Carrella, David Garcia, and Stephan Lewandowsky. 2022. Social media sharing of low-quality news sources by political elites. *PNAS Nexus*, 1(4). Pgac186.

David M. J. Lazer, Matthew A. Baum, Yochai Benkler, Adam J. Berinsky, Kelly M. Greenhill, Filippo Menczer, Miriam J. Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, Michael Schudson, Steven A. Sloman, Cass R. Sunstein, Emily A. Thorson, Duncan J. Watts, and Jonathan L. Zittrain. 2018. The science of fake news. *Science*, 359(6380):1094–1096.

Hause Lin, Jana Lasser, Stephan Lewandowsky, Rocky Cole, Andrew Gully, David Gertler Rand, and Gordon Pennycook. 2022. High level of agreement across different news domain quality ratings.

Sahil Loomba, Alexandre de Figueiredo, Simon J Piatek, Kristen de Graaf, and Heidi Jane Larson. 2021. Measuring the impact of COVID-19 vaccine misinformation on vaccination intent in the UK and USA. *Nature human behaviour*.

Alessandro Miani, Thomas Hills, and Adrian Bangerter. 2022a. Interconnectedness and (in)coherence as a signature of conspiracy worldviews. *Science Advances*, 8(43):eabq3668.

Alessandro Miani, Thomas Hills, and Adrian Bangerter. 2022b. Loco: The 88-million-word language of conspiracy corpus. *Behavior Research Methods*, 54(4):1794–1817.

Ludovic Mompelat, Zuoyu Tian, Amanda Kessler, Matthew Luettgen, Aaryana Rajanala, Sandra Kübler, and Michelle Seelig. 2022. How "loco" is the LOCO corpus? annotating the language of conspiracy theories. In *Proceedings of the 16th Linguistic Annotation Workshop (LAW-XVI) within LREC2022*, pages 111–119, Marseille, France. European Language Resources Association.

Rafael A Monteiro, Roney LS Santos, Thiago AS Pardo, Tiago A de Almeida, Evandro ES Ruiz, and Oto A Vale. 2018. Contributions to the study of fake news in Portuguese: New corpus and automatic detection results. In *International Conference on Computational Processing of the Portuguese Language*, pages 324–334. Springer.

Matteo Monti. 2020. Italian populism and fake news on the internet: A new political weapon in the public discourse. In Giacomo Delledonne, Giuseppe Martinico, Matteo Monti, and Fabio Pacini, editors, *Italian Populism and Constitutional Law: Strategies, Conflicts and Dilemmas*, pages 177–197. Springer International Publishing, Cham.

Giuseppina Lo Moro, Fabrizio Bert, Ettore Minutiello, Andrea L. Zacchero, Tiziana Sinigaglia, Gianluca Colli, Rossella Tatti, Giacomo Scaioli, and Roberta Siliquini. 2021. COVID-19 fake news, conspiracy beliefs and the role of eHealth literacy: an Italian nationwide survey. *The European Journal of Public Health*, 31.

Inc. NewsGuard. 2020. Rating process and criteria. Internet Archive, `https://web.archive.org/web/20200630151704/https://www.newsguardtech.com/ratings/rating-process-criteria/`. Accessed: 2022-04-20.

Dong Nguyen, Maria Liakata, Simon DeDeo, Jacob Eisenstein, David Mimno, Rebekah Tromble, and Jane Winters. 2020. How we do things with words: Analyzing text as social and cultural data. *Frontiers in Artificial Intelligence*, 3:62.

Steve Oswald. 2016. Conspiracy and bias: argumentative features and persuasiveness of conspiracy theories. *OSSA Conference Archive*, 168:1–16.

Gordon Pennycook, Ziv Epstein, Mohsen Mosleh, Antonio A Arechar, Dean Eckles, and David G Rand. 2021. Shifting attention to accuracy can reduce misinformation online. *Nature*, 592(7855):590–595.

Gordon Pennycook and David G. Rand. 2019. Fighting misinformation on social media using crowdsourced judgments of news source quality. *Proceedings of the National Academy of Sciences of the United States of America*, 116:2521 – 2526.

Juan-Pablo Posadas-Durán, Helena Gómez-Adorno, Grigori Sidorov, and Jesús Jaime Moreno Escobar. 2019. Detection of fake news in a new corpus for the Spanish language. *Journal of Intelligent & Fuzzy Systems*, 36(5):4869–4876.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2017. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*.

Leonard Richardson. 2007. Beautiful soup documentation. *April*.

Jon Roozenbeek, Claudia R. Schneider, Sarah Dryhurst, John R. Kerr, Alexandra L. J. Freeman, Gabriel Recchia, Anne Marthe van der Bles, and Sander van der Linden. 2020. Susceptibility to misinformation about COVID-19 around the world. *Royal Society Open Science*, 7.

Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Martino Trevisan, Luca Vassio, and Danilo Giordano. 2021. Debate on online social networks at the time of COVID-19: An Italian case study. *Online Social Networks and Media*, 23:100136.

Sander van der Linden, Anthony Leiserowitz, Seth A. Rosenthal, and Edward W. Maibach. 2017. Inoculating the public against misinformation about climate change. *Global Challenges*, 1.

Inna Vogel and Peter Jiang. 2019. Fake news detection with the new German dataset "GermanFakeNC". In *TPDL*.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3):e0150989.

A Appendix

A.1 Data availability

IRMA is freely available[6] and include the files:

1. **corpus.csv.zip**. A compressed file (.zip) containing the csv file of the corpus itself: 634,932 rows (documents) x 7 columns. Documents are identified by a hexadecimal unique ID stored in the variable `doc_id`. See dataset's variables in Table 3.
2. **website_description.csv**. A csv file containing detailed descriptions of websites. 56 rows (websites) x 25 columns. Note that due to proprietary data, NG's websites are anonymized via a unique website ID (e.g., *website1*, *website2*). See dataset's variables in Table 4.
3. **IRMA.dfm.rdata**. The IRMA's DTM. No preprocessing has been applied prior to conversion: the file contains punctuation and cased words (634,932 documents; 1,531,576 features). Note that the file was too large to be converted into a matrix, therefore we exported it as a quanteda DFM object, hence it requires the *quanteda* R package (Benoit et al., 2018).
4. **LDA_over_time.pdf**. A PDF file containing each topic's gamma values plotted over time. It contains 320 pages (i.e., the number of topics: k20 + k100 + k200). See Figure 3 for an example (at page 182 of the pdf file). Terms can be searched within the pdf.
5. **corpus_LF.csv**. The Lexical features obtained from LIWC. A csv file of 634,932 rows (documents) x 95 columns (94 LIWC lexical features and 1 documents' ID [`doc_id`]). See dataset's variables in Table 2.
6. **LDA_topic_gamma.csv.zip**. A compressed file (.zip) containing the csv file of 634,932 rows (documents) x 320 columns (LDA gamma values for k20, k100, and k200 topics). Each cell contains gamma value, that is the probability a topic is part of a document.
7. **topic_description.csv**. A file containing detailed descriptions of topics. 320 rows (topics) x 6 columns. See dataset's variables in Table 5.

---

[6] `https://osf.io/rywp4/`

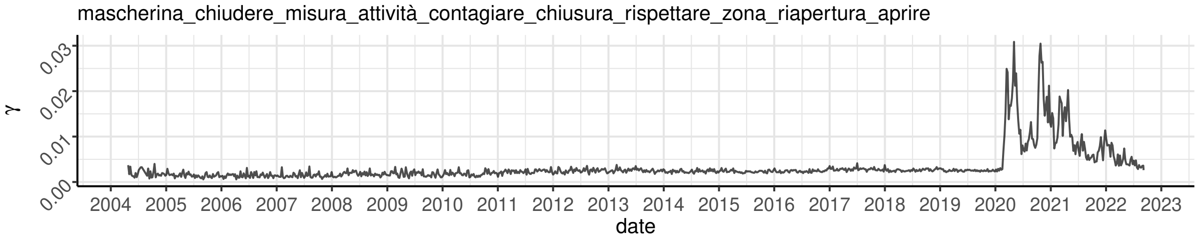


Figure 3: LDA topic gamma values (Y axis) over time (X axis). Topic `k200_062` related to Covid-19 restrictions. The 10 top-most important words for topics (in decreasing order) are displayed above the plot (ENG translation: mask, close, activity, contaging, closing, observe, zone, reopening, open).

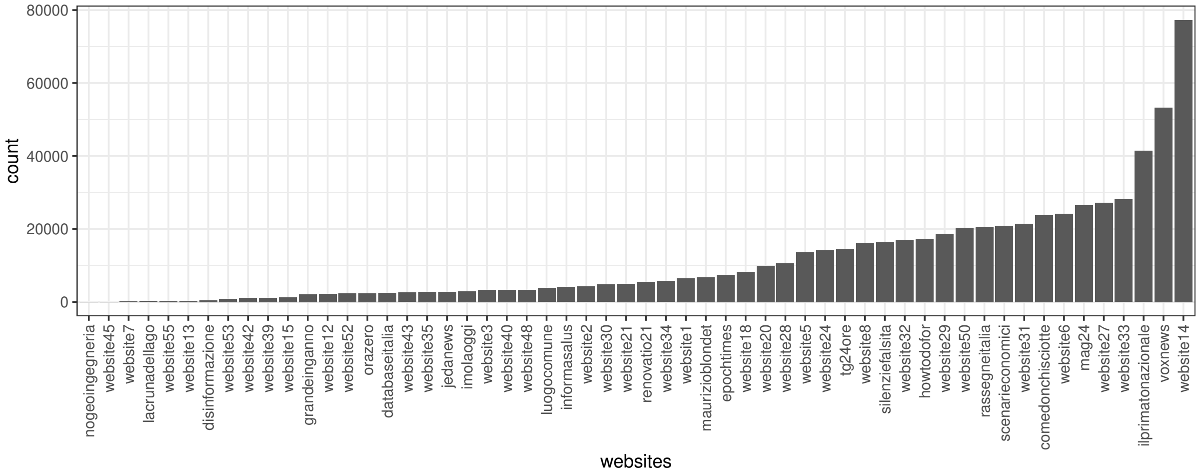


Figure 4: Document count by website.

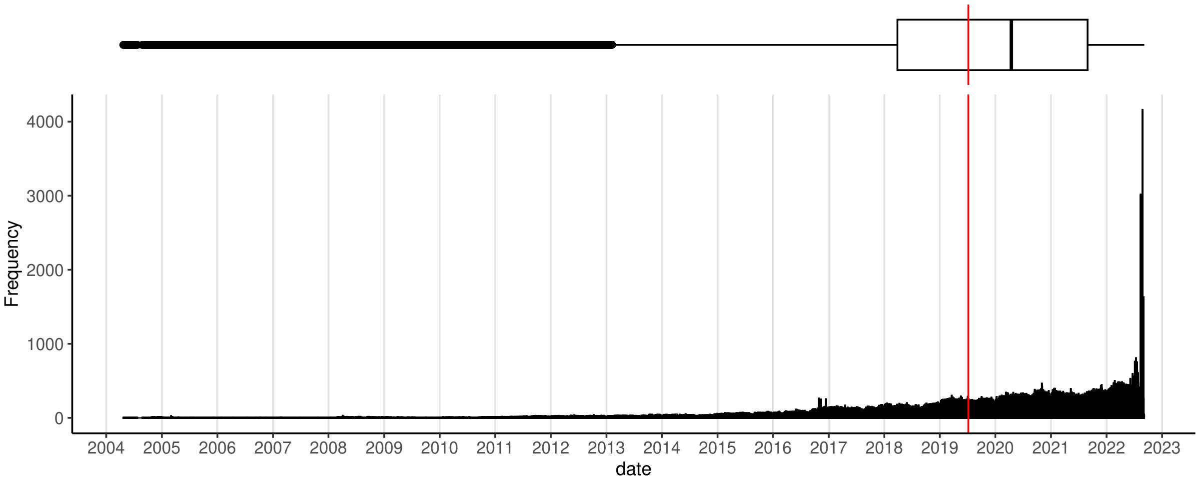


Figure 5: Distribution of documents by date (from "2004-04-22" to "2022-09-06"). The red vertical line represents the mean, the boxplot on top displays the median and the interquartile ranges.

Keyword	(ENG translation)	**Frequency**
vaccinare	(vaccinate)	6,677
ucraino	(Ukrainian)	5,013
trump	(Trump)	2,150
dragare	(Draghi)	2,075
pass	(pass)	1,980
conta	(Conti)	1,767
renzi	(Renzi)	1,669
salvini	(Salvini)	1,628
russia	(Russia)	1,326
mascherina	(mask)	1,314
banca	(bank)	1,295
putin	(Putin)	1,275
berlusconi	(Berlusconi)	1,246
siriano	(Syrian)	1,228
cina	(China)	1,227
cinese	(Chinese)	1,209
scuola	(School)	1,200
gesù	(Jesus)	1,198
papa	(Pope)	1,179
maio	(di Maio)	1,172

Table 1: Top 20 most frequent keywords (expressed as number of documents). Note that due to lemmatization, the words *dragare* and *conta* often refer to Mario Draghi and Giuseppe Conte.

Variable
(1) `doc_id`, (2) `WC`, (3) `WPS`, (4) `BigWords`, (5) `Dic`, (6) `pronomi`, (7) `Io`, (8) `Noi`, (9) `Se`, (10) `Tu`, (11) `Altri`, (12) `Negazio`, (13) `Consen`, (14) `Articol`, (15) `Prepos`, (16) `Numero`, (17) `Affett`, (18) `Sen_Pos`, (19) `Emo_Pos`, (20) `Ottimis`, (21) `Emo_Neg`, (22) `Ansia`, (23) `Rabbia`, (24) `Tristez`, (25) `Mec_Cog`, (26) `Causa`, (27) `Intros`, (28) `Discrep`, (29) `Inibiz`, (30) `possib`, (31) `Certez`, (32) `Proc_Sen`, (33) `Vista`, (34) `Udito`, (35) `Sentim`, (36) `Social`, (37) `Comm`, (38) `Rif_gen`, (39) `amici`, (40) `Famigl`, (41) `Umano`, (42) `Tempo`, (43) `Passato`, (44) `Present`, (45) `Futuro`, (46) `Spazio`, (47) `Sopra`, (48) `Sotto`, (49) `Inclusi`, (50) `Esclusi`, (51) `Movimen`, (52) `Occupaz`, (53) `Scuola`, (54) `Lavoro`, (55) `Raggiun`, (56) `Svago`, (57) `Casa`, (58) `Sport`, (59) `TV_it`, (60) `Musica`, (61) `Soldi`, (62) `Metafis`, (63) `religio`, (64) `Morte`, (65) `Fisico`, (66) `Corpo`, (67) `Sesso`, (68) `Mangiare`, (69) `Dormire`, (70) `Cura_cor`, (71) `parolac`, (72) `Non_flu`, (73) `riempiti`, (74) `Voi`, (75) `Lui_lei`, (76) `Loro`, (77) `Condizio`, (78) `Transiti`, (79) `P_pass`, (80) `gerundio`, (81) `Essere`, (82) `Avere`, (83) `Io_Ver`, (84) `Tu_Verbo`, (85) `Lui_Verb`, (86) `Noi_Verb`, (87) `Voi_Verb`, (88) `Loro_Ver`, (89) `AllPunc`, (90) `Period`, (91) `Comma`, (92) `QMark`, (93) `Exclam`, (94) `Apostro`, (95) `OtherP`

Table 2: List of columns for the dataset **corpus_LF.rdata**

Variable	Description
`doc_id`	Hexadecimal sequence of document unique identification number (e.g., `D1d049`)
`date`	The date the webpage was uploaded (format: `YYYY-MM-DD`, $N_{empty} = 45,185$)
`website`	The identification number for websites from which the document was extracted (e.g., *website15*, *ilprimatonazionale*; see also Table 4, below)
`title`	Title of the document ($N_{empty} = 359,526$)
`txt`	Document text ($N_{empty} = 358,851$)
`URL`	URL associated with the document ($N_{empty} = 360,137$)
`WC`	Word count
`KW`	Keyword associated with the document (see Table 1)
`tfidf10`	Top-10 words ordered by TF-IDF scores

Table 3: Names and variable descriptions for the dataset **corpus.csv.zip**

Variable	Description
`website`	Website's identification (e.g., *grandeinganno*, *website21*)
`Ndoc`	Number of documents for each website
`WC_{type}`	Word count statistics. Type includes: mean, SD, min, and max
`DATE_{type}`	Date range. Type includes: min and max ($N_{empty} = 8$)
`type_of_news`	Website type of content (e.g., conspiracy, political, health-related, religious, general, and/or viral)
`Monthly_Visits`	Count of visits in the past month (i.e., September 2022). Note that for websites with less than 5,000 monthly visits, SimilarWeb does not collect further traffic data. To those websites, ($N = 7$), we assigned the value 5,000
`Visit_Duration`	Average of visit duration (in seconds)
`Bounce_Rate`	The percentage of visitors who enter a site and leave after visiting only one page
`Pages_per_Visit`	Average of pages visited in each visit
`Traffic_{type}`	Proportion of incoming traffic. Type includes: Direct, Referrals, Search, and Social
`Social_{type}`	Proportion of incoming traffic from social media. Type includes: Linkedin, Vkontakte, Others, Telegram_Webapp, Youtube, Twitter, and Facebook
`credibility`	Websites' credibility scores (obtained from Lin et al., 2022).

Table 4: Names and variable descriptions for the dataset **website_description.csv**

Variable	Description
`topic_name`	Topic unique ID. It is composed by the topic resolution plus a three-character serial number (e.g., `k100_032` is the 32th topic at 100k resolution)
`top_words`	Top-ten words ordered by importance (for the topic)
`topic`	Name of the topic with the highest correlation (within the same topic resolution)
`topic_cor`	Pearson r correlation estimate for the highest correlated topic
`LF`	Name of the LIWC's lexical feature with the highest correlation
`LF_cor`	Pearson r correlation estimate

Table 5: Names and variable descriptions for the dataset **topic_description.csv**. Note that correlation are computed on the document level ($N = 634,932$)

Parameter-Efficient Korean Character-Level Language Modeling

Marco Cognetta
Tokyo Institute of Technology
`cognetta.marco@gmail.com`

Sangwhan Moon
Tokyo Institute of Technology
`sangwhan@iki.fi`

Lawrence Wolf-Sonkin
Google
`wolfsonkin@google.com`

Naoaki Okazaki
Tokyo Institute of Technology
`okazaki@c.titech.ac.jp`

Abstract

Character-level language modeling has been shown empirically to perform well on highly agglutinative or morphologically rich languages while using only a small fraction of the parameters required by (sub)word models. Korean fits nicely into this framework, except that, like other CJK languages, it has a very large character vocabulary of 11,172 unique *syllables*. However, unlike Japanese Kanji and Chinese Hanzi, each Korean syllable can be uniquely factored into a small set of subcharacters, called *jamo*.

We explore a "three-hot" scheme, where we exploit the decomposability of Korean characters to model at the syllable level but using only jamo-level representations. We find that our three-hot embedding and decoding scheme alleviates the two major issues with prior syllable- and jamo-level models. Namely, it requires fewer than 1% of the embedding parameters of a syllable model, and it does not require tripling the sequence length, as with jamo models. In addition, it addresses a theoretical flaw in a prior three-hot modeling scheme.

Our experiments show that, even when reducing the number of embedding parameters by $> 99.6\%$ (from 11.4M to just 36k), our model suffers no loss in translation quality compared to the baseline syllable model.

1 Introduction

Subword modeling has been used for Korean to reduce the required vocabulary size due to its agglutinative nature and morphological richness. Several works have characterized the many subword tokenization strategies for Korean (Park et al., 2020; Moon and Okazaki, 2020).

(Sub)character modeling has been employed for translation in Asian languages (Nguyen et al., 2017; Ngo et al., 2019; Yu et al., 2017). Likewise, it has found use in generic Korean NLP tasks (Cho et al., 2019; Choi et al., 2017). Further, Stratos (2017) investigated exploiting the hierarchical nature of Korean characters for modeling. However, nearly all of these works are restricted to the input side only, either because the downstream task is a classification task or it is a translation task where Korean is the *source* language, and so syllable generation is unnecessary. An exception is Song et al. (2018), where the authors propose a "multi-hot" scheme that appears on both the embedding and decoding sides of a sequence-to-sequence denoising autoencoder applied to Korean spelling correction.

The baseline approaches of syllable- and jamo-level modeling each have downsides. Syllable models must have embeddings for all 11,172 syllables to provide full coverage, which requires an enormous number of parameters. On the other hand, jamo-level modeling requires fewer than 70 embeddings, but the sequence lengths become 3x longer, which greatly slows inference in attention-based models. Here, we propose a three-hot model[1] that addresses both of these issues, as well as an issue in the architecture from Song et al. (2018).

We emphasize that our goal is specifically to investigate parameter-efficient character-level modeling, and not to compare with (sub)word models.

1.1 Korean Syllabary

$\mathcal{V}_i$	ㄱ ㄲ ㄴ ㄷ ㄸ ㄹ ㅁ ㅂ ㅃ ㅅ ㅆ ㅇ ㅈ ㅉ ㅊ ㅋ ㅌ ㅍ ㅎ
$\mathcal{V}_v$	ㅏ ㅐ ㅑ ㅒ ㅓ ㅔ ㅕ ㅖ ㅗ ㅘ ㅙ ㅚ ㅛ ㅜ ㅝ ㅞ ㅟ ㅠ ㅡ ㅢ ㅣ
$\mathcal{V}_f$	ㄱ ㄲ ㄳ ㄴ ㄵ ㄶ ㄷ ㄹ ㄺ ㄻ ㄼ ㄽ ㄾ ㄿ ㅀ ㅁ ㅂ ㅄ ㅅ ㅆ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ ㅎ $\oslash$

Table 1: The three jamo classes. $\oslash \in \mathcal{V}_f$ represents the *absence* of a final consonant.

---

[1] Our three-hot implementation has been packaged as a library alongside code to reproduce our experiments: `https://github.com/mcognetta/ThreeHotKoreanModeling`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2350–2356
May 2-6, 2023

a) 기계번→역

b) ㄱㅣ∅ㄱㅖ∅ㅂㅓㄴ→ㅇ→ㅕ→ㄱ

c) (ㄱ, ㅣ, ∅)(ㄱ, ㅖ, ∅)(ㅂ, ㅓ, ㄴ)→(ㅇ, ㅕ, ㄱ)

Figure 1: Modeling scenarios for predicting the syllable 역 (*yeok*:'decode'), from the context "기계번"(*gigyebeon*:'machine translate') using a) syllable, b) jamo, and c) three-hot modeling. Notice that in jamo modeling, the prediction is spread over three time-steps.

Korean syllables are made up of three subcharacters: an initial consonant, a vowel, and an optional final consonant, collectively known as jamo. There are 19, 21, and 28 initial consonants, vowels, and final consonants (including the lack of a final consonant, represented by $\oslash$), denoted $\mathcal{V}_i$, $\mathcal{V}_v$, and $\mathcal{V}_f$, respectively, shown in Table 1. These classes are disjoint, and even visually similar jamo in $\mathcal{V}_i$ and $\mathcal{V}_f$ (e.g., ㄱ) are distinct subcharacters.

Any Korean syllable can be uniquely decomposed into three jamo, and any choice $(i, v, f) \in \mathcal{V}_i \times \mathcal{V}_v \times \mathcal{V}_f$ corresponds to a unique syllable. For example, 한: (ㅎ, ㅏ, ㄴ) and 무: (ㅁ, ㅜ, $\oslash$).

Specific details about Korean jamo and their Unicode representations are in Appendix A.

2 Three-hot Modeling

Naïve syllable-level and jamo-level modeling use one-hot encodings to represent each token. We propose a three-hot scheme, where jamo triplets representing syllables are consumed and generated at each step. Figure 1 gives examples of each of the considered modeling schemes.

2.1 Embedding

In three-hot modeling, we learn embeddings for jamo and combine them to produce a syllable embedding as:

$$emb_s = emb_i + emb_v + emb_f,$$

where $+$ is vector addition. We also experimented with combining $emb_{i,v,f}$ via concatenation, but this method requires more parameters and we observed no performance difference.

Similar factored embeddings have been considered before in language-agnostic settings (Svenstrup et al., 2017) and for Korean specifically (Stratos, 2017).

2.2 Independent Three-hot Decoding

Song et al. (2018) proposed a three-hot decoder for a sequence-to-sequence autoencoder which produces syllables by predicting three jamo subcharacters as a three-hot vector. In their scheme, they predict the individual jamo *independently*, as:

$$\mathcal{P}(s \mid h) \approx \mathcal{P}(i \mid h) \times \mathcal{P}(v \mid h) \times \mathcal{P}(f \mid h).$$

However, a syllable's jamo are *not* independent, and thus this method does not capture the true joint probability distribution:

$$\mathcal{P}(s \mid h) = \mathcal{P}(i, v, f \mid h).$$

2.3 Conditional Three-hot Decoding

To properly model the true joint distribution, we factor it as:

$$\mathcal{P}(s \mid h) = \mathcal{P}(i \mid h) \times \mathcal{P}(v \mid i, h) \times \mathcal{P}(f \mid i, v, h).$$

Given a context embedding h from a base language model, we first predict the initial consonant, π_i, from $\texttt{softmax}(I(h_i))$, where $I : \mathbb{R}^d \to \mathbb{R}^{|\mathcal{V}_i|}$ and h_i is a vector generated from h and an initial vector h_0. Then, we generate a continuous embedding for π_i via $\texttt{embedding}_i : \mathbb{R}^{|\mathcal{V}_i|} \to \mathbb{R}^d$. The continuous embedding is combined with h_i and used to predict the vowel jamo, π_v. Likewise, π_v is re-embedded and used to predict the final consonant, π_f. As such, a triplet (π_i, π_v, π_f) is generated similarly to a three-step, unrolled RNN:

$$\begin{aligned}
h_i &= \texttt{tanh}(W_e h + W_h h_0) \\
\pi_i &\sim \texttt{softmax}(I(h_i)) \\
emb_i &= \texttt{embedding}_i(\pi_i) \\
h_v &= \texttt{tanh}(W_e emb_i + W_h h_i) \\
\pi_v &\sim \texttt{softmax}(V(h_v)) \\
emb_v &= \texttt{embedding}_v(\pi_v) \\
h_f &= \texttt{tanh}(W_e emb_v + W_h h_v) \\
\pi_f &\sim \texttt{softmax}(F(h_f))
\end{aligned} \tag{1}$$

In the decoding layer, we define $\texttt{embedding}_{i,v}$, but not $\texttt{embedding}_f$. This is because once the prediction π_f has been made, decoding for that time-step is done, and so an embedding emb_f is not needed to generate any more jamo.

2.3.1 Parameter Reduction via Weight Sharing and Diagonal RNNs

A common way to reduce parameter counts is to share weights between the embedding and decoding layers. For jamo and syllable models, this is

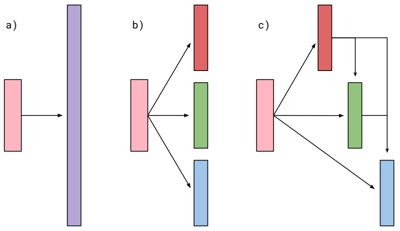


Figure 2: The three decoding types: `a)` one-hot decoding (jamo or syllable), `b)` independent three-hot decoding (Song et al., 2018), and `c)` conditional three-hot decoding (ours). The light pink box is a contextual embedding h, while red, green, and blue are $\mathcal{V}_i$, $\mathcal{V}_v$, and $\mathcal{V}_f$, respectively.

straightforward, as the one-hot embedding table can be reused for the final linear output layer.

For independent three-hot models, the embedding weights from Section 2.1 can also be reused in the final output layers. For conditional three-hot models, the embedding weights can be used for both $\texttt{embedding}_{i,v}$ and I, V, F in Equation 1, so that only the matrices W_e and W_h are unshared. Additionally, we hypothesize that, due to the strictly bounded context length Equation 1, a fully dense RNN transition matrix may be excessive, and so we experiment with replacing W_e and W_h with diagonal matrices (Sübakan and Smaragdis, 2017).

2.4 Prediction Order

In multi-label classification, it has been observed that the order in which labels are generated can impact a model's accuracy (Read et al., 2021). For three-hot decoding, we experiment with the six triplet-generation permutations to see if there is any significant difference. This does not require any modification of the training data or architecture, as we need only permute the order of Equation 1.

2.5 Triplet Representation

Three-hot decoding models output jamo triplets representing a full syllable. However, the model may need to process non-Korean characters, which do not have a three-hot jamo decomposition. To unify the representations of Korean and non-Korean tokens, we add all non-Korean tokens to $\mathcal{V}_i$ and introduce a padding character $\mathbf{x} \in \mathcal{V}_{v,f}$. Then, a non-Korean token c is represented by $(c, \mathbf{x}, \mathbf{x})$.

To ensure a fair comparison between the three-hot decoding schemes and the jamo model, no logic is used to prohibit the generation of degenerate triplets, such as a non-Korean symbol followed by jamo subcharacter. Syllable and jamo models do not generate $\mathbf{x}$ when predicting non-Korean tokens, but jamo models are trained to predict $\oslash$ if a syllable does not have a final consonant.

3 Parameter Counts

Table 2 shows the parameter counts for each of the architectures. Since the three-hot models use jamo embeddings, they have the same number of embedding parameters as the jamo-level models. The conditional three-hot models have additional parameters for the three-stage decoding step: two internal transition matrices W_e and W_h (Equation 1), which dominate the decoding layer's parameter count. But, when the transition matrices are diagonalized and weight sharing is used, the total number of parameters is only 1k more than the shared-weight jamo and independent three-hot models.

The syllable models contain several orders of magnitude more parameters than any other architecture. In the most extreme case, unshared syllable vs. shared, diagonal three-hot, the latter has only $\frac{36k}{11.4M} \approx 0.32\%$ as many embedding parameters.

Our underlying LM is a transformer model with 15.7M parameters on the target side[2], not counting those for embedding or generation. Thus, the unshared syllable-level embedding/decoding layers increase the target-side parameter count by 73%, while the unshared jamo, independent, and diagonal conditional three-hot increase it by less than 2%, and non-diagonal conditional three-hot by less than 6%.

4 Evaluation Metrics

Since we are modeling at two different granularities, perplexity is not immediately comparable between the different architectures. To unify them, we use bits-per-jamo (BPJ) as our granularity-agnostic metric.

We use BLEU and chrF to evaluate the quality of the translations produced by our models. Since our models work on different granularities, we canonicalize their outputs for comparison.

For syllable-level modeling, we split all syllables into their jamo subcharacters and leave all non-Korean-syllable tokens as is. For three-hot models, triplets are flattened to a string of three characters. We remove the **x** pad from non-Korean tokens and replace degenerate triplets with a special `BAD` token. For all models, all jamo are converted to a

---

[2] See Appendix B for architecture details.

	Embedding	Decoding	Total	BPJ	BLEU	chrF
Syllable (unshared)	5.7M	5.7M	11.4M	33.9	14.1	38.1
Syllable (shared)	5.7M	-	5.7M	0.342	14.0	38.1
Jamo (unshared)	35k	35k	70k	0.355	13.7	37.8
Jamo (shared)	35k	-	35k	0.356	14.1	38.0
Three-hot (Ind., unshared)	35k	35k	70k	0.555	7.9	28.9
Three-hot (Ind., shared)	35k	-	35k	0.556	8.3	29.4

	Embedding	Decoding	Total	BPJ	BLEU	chrF
Three-hot (IVF, unshared)	35k	579k	614k	0.287	14.3	38.1
Three-hot (IVF, shared)	35k	524k	559k	0.292	14.1	38.1
Three-hot (IVF, diag., unshared)	35k	71k	106k	0.293	14.2	38.2
Three-hot (IVF, diag., shared)	35k	1k	36k	0.306	14.0	38.0
Three-hot (FIV, unshared)	35k	579k	614k	0.289	14.0	37.9
Three-hot (FIV, shared)	35k	524k	559k	0.294	14.1	37.8
Three-hot (FIV, diag., unshared)	35k	71k	106k	0.294	14.0	37.9
Three-hot (FIV, diag., shared)	35k	1k	36k	0.304	13.8	37.8

Table 2: Parameter counts (when $d = 512$, as in our experiments) and metrics (BPJ, BLEU, and chrF) for each of the architectures. For brevity, only some decoding orders are listed here. The decoding column lists only parameters that are not shared with the embedding layer. A complete set of metrics is given in Appendix C.

canonical *compatibility jamo* format (see Appendix A). Finally, we remove all punctuation.

For BLEU, we use the standard n-gram order of 4, since this operates on the word-level (in our experiments, contiguous token sequences surrounded by whitespace). The default chrF character n-gram order is 6, which we interpret as meaning 6 *syllables* for Korean. Since we measure our metrics on decomposed jamo sequences, we use a jamo n-gram order of 18.

5 Experiments

We use an English-Korean news translation dataset from AI Hub[3] with 400k sentences. We remove all sentence pairs that have more than 100 syllables on the Korean side, leaving 365k , from which we select 5k and 5k for testing and validation.

As our only goal is to evaluate the effectiveness of three-hot modeling compared to naïve syllable- and jamo-level modeling, we use the same base transformer model architecture and hyperparameters in all experiments and only change the target-side embedding and decoding layers. The complete hyperparameter list is given in Appendix B.

In total, we train 30 models: unshared and shared weights for syllable, jamo, and three-hot independent modeling (6 total) and $\{\text{unshared, shared}\} \times \{\text{dense, diagonal}\}$ for each of the 6 conditional three-hot prediction orders (24 total). Each model is trained for 50 epochs, and the epoch with the lowest bits-per-jamo on the validation set is used.

For inference, we use a beam size of 15 for syllable models and 8 for jamo models. Three-hot decoding is two-staged: an internal beam search constructs triplets jamo-by-jamo, and the highest probability triplets are added to the outer beam as syllables. For the independent three-hot models, we use beam size of 5 and an internal beam size of 3. For the conditional three-hot models, we use a beam size of 15 and an internal beam size of 4. These hyperparameters were determined by a sweep search on the validation set.

6 Results and Analysis

Table 2 lists the metrics for each model type.

Independent three-hot modeling performs the worst across all metrics. On translation in particular, we suspect that it is because the model can generate high probability but meaningless triplets, especially in scenarios where several reasonable syllable continuations exist. The highest probability triplet may contain individual jamo that come from each of the true high probability syllable continuations, but which are combined in a way that forms a meaningless character. Such triplets quickly saturate the beam with gibberish contexts and degrade the translation quality.

For bits-per-jamo, conditional three-hot models vastly outperform all other architectures. However, this has some caveats. Since the syllable model must predict all three jamo simultaneously, the bits-per-jamo is spread evenly throughout the component jamo. On the other hand, conditional three-hot modeling has a relatively difficult time predicting the first jamo, but the subsequent elements are increasingly easy due to the additional conditioning (Appendix C lists BPJ-per-jamo-class for each model). Jamo models must deal with a longer context (and thus less focused attention) and the output layer can output any class of jamo or non-Korean tokens at any time step, which makes modeling less structured and more difficult than conditional three-hot and syllable-level models.

We observed that the independent three-hot models had the highest perplexity on $\mathcal{V}_v$, followed by $\mathcal{V}_i$, then $\mathcal{V}_f$. Since this family generates jamo independently, it approximates the true entropy of the marginal subcharacter distributions. Thus, we conjectured that (f, i, v) order would be the best and (v, i, f) the worst for model accuracy, but found that no prediction order consistently outperformed

---

[3] `https://aihub.or.kr/`

the others.

For translation, the BLEU and chrF scores for jamo, syllable, and conditional three-hot models are extremely close. We found that, as we moved from unshared, dense three-hot models to shared, diagonal models, there was a slight loss in BPJ performance, but even in the diagonal, shared case, all three-hot models perform on-par with syllable models on translation (± 0.3 BLEU and ± 0.3 chrF), supporting our claims that the syllable models are massively overparameterized, and using fully dense W_e and W_h matrices in Equation 1 is unnecessary.

7 Conclusion

We presented a conditional three-hot decoder for Korean character-level language models. Our model addresses several issues with other Korean (sub)character-level modeling schemes. Compared to syllable-level models, it uses only a small fraction of the number of parameters, and, unlike jamo-level models, it does not triple the sequence length, avoiding the resulting inference time increase with attention mechanisms. It also addresses a theoretical flaw in a prior three-hot decoding scheme, where a syllable's subcharacters were generated independently. Finally, we proposed several variants of our model to further reduce parameters.

On a character-level translation task, we found that all of our conditional three-hot models perform on-par with jamo and syllable models, even when using as little as ~0.3% of the embedding parameters of a syllable model. They also outperform the prior independent three-hot model in every metric.

These results suggest that conditional three-hot modeling is an efficient and principled method of character-level Korean language modeling.

8 Acknowledgments

These research results were obtained partially from the commissioned research (No. 225) by National Institute of Information and Communications Technology (NICT), Japan.

9 Limitations

One limitation of this work is that it is specific to the writing system typically used in Korean, Hangul. A similar idea could be used for subword modeling in other scripts, but the idea presented here draws specifically on the hierarchical and compositional nature of Hangul syllables that is unique to the script itself. A second is that we did not compare to state-of-the-art non-character-level Korean translation models. However, this was on purpose, as the point of this paper is specifically to investigate efficient character-level modeling.

References

Won-Ik Cho, Seok Min Kim, and Nam Soo Kim. 2019. Investigating an effective character-level embedding in korean sentence classification. *CoRR* abs/1905.13656.

Sanghyuk Choi, Taeuk Kim, Jinseok Seol, and Sang-goo Lee. 2017. A syllable-based technique for word embeddings of Korean words. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*. Association for Computational Linguistics, Copenhagen, Denmark, pages 36–40.

Sangwhan Moon and Naoaki Okazaki. 2020. Jamo pair encoding: Subcharacter representation-based extreme Korean vocabulary compression for efficient subword tokenization. In *Proceedings of the 12th Language Resources and Evaluation Conference*. European Language Resources Association, Marseille, France, pages 3490–3497.

Thi-Vinh Ngo, Thanh-Le Ha, Phuong-Thai Nguyen, and Le-Minh Nguyen. 2019. How transformer revitalizes character-based neural machine translation: An investigation on japanese-vietnamese translation systems. *CoRR* abs/1910.02238.

Viet Nguyen, Julian Brooke, and Timothy Baldwin. 2017. Sub-character neural language modelling in Japanese. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*. Association for Computational Linguistics, Copenhagen, Denmark, pages 148–153.

Kyubyong Park, Joohong Lee, Seongbo Jang, and Dawoon Jung. 2020. An empirical study of tokenization strategies for various korean NLP tasks. Association for Computational Linguistics, pages 133–142.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2021. Classifier chains: A review and perspectives. *J. Artif. Intell. Res.* 70:683–718.

Chisung Song, Myungsoo Han, Hoon Young Cho, and Kyong-Nim Lee. 2018. Sequence-to-sequence autoencoder based korean text error correction using syllable-level multi-hot vector representation. In *Proceedings of HCLT (in Korean)*. pages 661–664.

Karl Stratos. 2017. A sub-character architecture for korean language processing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Association for Computational Linguistics, pages 721–726.

Y. Cem Sübakan and Paris Smaragdis. 2017. Diagonal rnns in symbolic music modeling. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA 2017, New Paltz, NY, USA, October 15-18, 2017*. pages 354–358.

Dan Svenstrup, Jonas Meinertz Hansen, and Ole Winther. 2017. Hash embeddings for efficient word representations. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. pages 4928–4936.

Jinxing Yu, Xun Jian, Hao Xin, and Yangqiu Song. 2017. Joint embeddings of Chinese words, characters, and fine-grained subcharacter components. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 286–291.

A Hangul and Jamo

Korean is divided into three Unicode blocks: syllables[4], jamo[5], and *compatability* jamo[6].

The syllable block contains a codepoint for each of the 11,172 unique syllables and the jamo block contains one codepoint for each of the initial, vowel, and final jamos, separated by class. Table 1 shows all of the jamo. Note that some visually identical jamo exist in both the initial and final consonant classes (e.g., ㄱ). These are technically distinct jamo, hence the need for two unique codepoints. The compatibility jamo block contains a codepoint for each consonant and vowel, but merges visually identical ones from across different classes. In total there are 51 compatibility jamo. Conversion from syllables to jamo and compatibility jamo is deterministic using basic modular arithmetic on the codepoints[7]. Likewise, it is trivial to recompose jamo into syllables. Since compatibility jamo has ambiguities about which class a specific jamo belongs to, it must be composed greedily from left to right using a state machine. However, in all cases, assuming that that jamo sequence is valid, both jamo and compatibility sequences can be unambiguously recomposed into syllables.

In our experiments, we always convert from syllables/triplets/jamo to compatibility jamo, and never from compatibility jamo, so all ambiguities are avoided even in the case of invalid jamo sequences.

B Baseline Transformer LM

A single baseline transformer model configuration was used across all experiments, with only the target side embedding and decoding layers swapped out. Table 3 lists the relevant hyperparameters. All models were trained on an NVIDIA RTX A6000 GPU using PyTorch 1.12.

Hyperparameter	**Value**
English Vocabulary	30k BPE
Embedding Dimension	512
Feed Forward Dimension	512
Encoder Layers	6
Decoder Layers	6
Heads	8
Optimizer	ADAM
Learning Rate	1e-4
Betas	(0.9, 0.98)
Batch Size (tokens)	4k (10k for Jamo)
Epochs	50

Table 3: Hyperparameters for the base transformer language model.

C Full Metrics

The full metrics for all experients are listed in Table 4. This is the complete version of Table 2.

---

[4] https://en.wikipedia.org/wiki/Hangul_Syllables

[5] https://en.wikipedia.org/wiki/Hangul_Jamo_(Unicode_block)

[6] https://en.wikipedia.org/wiki/Hangul_Compatibility_Jamo

[7] https://en.wikipedia.org/wiki/Korean_language_and_computers#Hangul_in_Unicode

	Embedding	Decoding	Total	BPJ	BPJ (i)	BPJ (v)	BPJ (f)	BLEU	chrF
Syllable (unshared)	5.7M	5.7M	11.4M	0.339	-	-	-	14.1	38.1
Syllable (shared)	5.7M	-	5.7M	0.342	-	-	-	14.0	38.1
Jamo (unshared)	35k	35k	70k	0.355	-	-	-	13.7	37.8
Jamo (shared)	35k	-	35k	0.356	-	-	-	14.1	38.0
Three-hot (Ind., unshared)	35k	35k	70k	0.555	0.588	0.599	0.478	7.9	28.9
Three-hot (Ind., shared)	35k	-	35k	0.556	0.589	0.600	0.478	8.3	29.4
Three-hot (IVF, unshared)	35k	579k	614k	0.287	0.556	0.208	0.099	14.3	38.1
Three-hot (IVF, shared)	35k	524k	559k	0.292	0.561	0.214	0.103	14.1	38.1
Three-hot (IVF, diag., unshared)	35k	71k	106k	0.293	0.561	0.211	0.108	14.2	38.2
Three-hot (IVF, diag., shared)	35k	1k	36k	0.306	0.571	0.226	0.121	14.0	38.0
Three-hot (IFV, unshared)	35k	579k	614k	0.288	0.556	0.127	0.183	14.0	37.7
Three-hot (IFV, shared)	35k	524k	559k	0.293	0.563	0.131	0.186	13.9	37.8
Three-hot (IFV, diag., unshared)	35k	71k	106k	0.293	0.560	0.136	0.183	14.1	38.1
Three-hot (IFV, diag., shared)	35k	1k	36k	0.305	0.569	0.153	0.195	14.1	38.1
Three-hot (VIF, unshared)	35k	579k	614k	0.288	0.175	0.590	0.100	14.0	38.2
Three-hot (VIF, shared)	35k	524k	559k	0.293	0.179	0.597	0.102	14.0	38.3
Three-hot (VIF, diag., unshared)	35k	71k	106k	0.294	0.178	0.595	0.109	14.1	37.9
Three-hot (VIF, diag., shared)	35k	1k	36k	0.305	0.190	0.605	0.120	14.3	38.1
Three-hot (VFI, unshared)	35k	579k	614k	0.287	0.114	0.589	0.160	14.1	38.1
Three-hot (VFI, shared)	35k	524k	559k	0.292	0.119	0.596	0.162	14.3	38.2
Three-hot (VFI, diag., unshared)	35k	71k	106k	0.294	0.126	0.595	0.161	13.9	38.1
Three-hot (VFI, diag., shared)	35k	1k	36k	0.306	0.142	0.606	0.170	13.8	37.8
Three-hot (FIV, unshared)	35k	579k	614k	0.289	0.262	0.128	0.478	14.0	37.9
Three-hot (FIV, shared)	35k	524k	559k	0.294	0.266	0.132	0.484	14.1	37.8
Three-hot (FIV, diag., unshared)	35k	71k	106k	0.294	0.264	0.137	0.482	14.0	37.9
Three-hot (FIV, diag., shared)	35k	1k	36k	0.304	0.276	0.151	0.486	13.8	37.8
Three-hot (FVI, unshared)	35k	579k	614k	0.289	0.116	0.273	0.478	14.3	38.2
Three-hot (FVI, shared)	35k	524k	559k	0.293	0.120	0.277	0.482	14.1	38.3
Three-hot (FVI, diag., unshared)	35k	71k	106k	0.294	0.125	0.276	0.481	14.1	38.0
Three-hot (FVI, diag., shared)	35k	1k	36k	0.305	0.143	0.287	0.487	14.2	38.0

Table 4: The complete metrics for Table 2. For each model architecture and prediction order, parameter counts (when $d = 512$, as in our experiments) and metrics (BPJ, BLEU, and chrF) are given. Additionally, for three-hot models, the BPJ-per-jamo-class is provided. The decoding column lists only unshared parameters.

Opportunities and Challenges in Neural Dialog Tutoring

Jakub Macina^{*1,2} **Nico Daheim**^{*3} **Lingzhi Wang**^{4}
Tanmay Sinha^{5} **Manu Kapur**^{5} **Iryna Gurevych**^{3} **Mrinmaya Sachan**^{1}

[1]Department of Computer Science, ETH Zürich [2] ETH AI Center
[3] Ubiquitous Knowledge Processing Lab (UKP Lab), Department of Computer Science
and Hessian Center for AI (hessian.AI), TU Darmstadt
[4] The Chinese University of Hong Kong
[5]Professorship for Learning Sciences and Higher Education, ETH Zürich
`jakub.macina@ai.ethz.ch`

Abstract

Designing dialog tutors has been challenging as it involves modeling the diverse and complex pedagogical strategies employed by human tutors. Although there have been significant recent advances in neural conversational systems using large language models (LLMs) and growth in available dialog corpora, dialog tutoring has largely remained unaffected by these advances. In this paper, we rigorously analyze various generative language models on two dialog tutoring datasets for language learning using automatic and human evaluations to understand the new opportunities brought by these advances as well as the challenges we must overcome to build models that would be usable in real educational settings. We find that although current approaches can model tutoring in constrained learning scenarios when the number of concepts to be taught and possible teacher strategies are small, they perform poorly in less constrained scenarios. Our human quality evaluation shows that both models and ground-truth annotations exhibit low performance in terms of equitable tutoring, which measures learning opportunities for students and how engaging the dialog is. To understand the behavior of our models in a real tutoring setting, we conduct a user study using expert annotators and find a significantly large number of model reasoning errors in 45% of conversations. Finally, we connect our findings to outline future work.

`https://github.com/eth-nlped/dialog-tutoring`

1 Introduction

The goal of dialog tutoring research is to build systems that can tutor students using natural language conversation (Wollny et al., 2021). For several decades, learning scientists have been studying the features of domain-specific dialog tutoring systems that engender learning in students (Chi et al., 1994; Graesser et al., 1995; Moore et al., 2004; Litman et al., 2006; Graesser, 2016; Ruan et al., 2019) and have established strong learning gains that are even comparable to human tutoring in specific domains (Nye et al., 2014). However, these systems require extensive authoring of materials by teachers (MacLellan and Koedinger, 2020) and therefore cannot fully utilize the scalability of online learning.

Building dialog tutors is technically challenging as tutoring dialogs typically exhibit properties that are absent in other forms of dialog. Tutoring dialogs are often *long*, enabling students to be exposed to the concepts in a way that they can use them in future (Chi and Wylie, 2014), and *grounded* in the learning scenarios (Graesser et al., 2009). Finally, good dialog tutors are engaging and create opportunities to learn, providing students space to seek and provide explanations, and self-reflect (Chi and Wylie, 2014; Reiser, 2004).

The growing success of deep neural network based language generators in other dialog settings (Adiwardana et al., 2020; Roller et al., 2021) suggests new possibilities in dialog tutoring that could scale beyond domain-specific approaches. However, despite their promise, advances in neural generative models have seen little adoption in dialog tutoring.

In this paper, we contribute a comprehensive study of the applicability of neural generative models in tutoring. We formally introduce the dialog tutoring task and analyze existing tutoring datasets (§2). Then, we describe several generative and retrieval-based models for dialog tutoring (§3) and benchmark them on two open-access dialog tutoring datasets for language learning: *CIMA* (Stasaski et al., 2020, a crowdsourced role-played dataset for learning prepositional phrases in Italian) and

---

[*] Equal contribution.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2357–2372
May 2-6, 2023 ©2023 Association for Computational Linguistics

Teacher-Student Chatroom Corpus (TSCC) (Caines et al., 2020, a one-to-one English tutoring dataset from an online chatroom) (§5.1). We evaluate our models on various automatic metrics (§4.2) as well as two human evaluation studies: an evaluation of the quality of the generated response with respect to various measures of goodness (§6.1), as well as a more realistic user study with a learning interface (§6.2).

Overall, while we find that pretrained models improve over simpler baselines in terms of automatic metrics, our consequent human evaluation reveals several shortcomings that ought to be addressed before these models can be adopted in the real world. We find that while neural generative models can model more constrained learning settings well, they struggle when the learning goal is more open-ended. Specifically, these models are unable to understand and reason about student solutions and misconceptions, and thus, are unable to use effective pedagogical strategies.

We find that the field of dialog tutoring is significantly limited by the quantity and quality of available datasets. The available datasets are both too small and not rich enough to capture the nuances of the dialog tutoring problem. Our analysis also reveals the inadequacy of automatic evaluation metrics for capturing tutoring quality. Not only are the existing metrics unable to capture faithfulness to the learning material and the student dialog history, but they also cannot capture moves of good human tutors that allow learners the space for reflection, explanation, follow-ups, and real engagement in the process of learning.

Based on our findings, we end with an outline of potential avenues of future research (§7). We hope that our paper will bring attention to this underexplored natural language processing application with the potential for significant social good.

2 The Dialog Tutoring Task

Dialog tutoring can be described as a multi-turn interaction between two interlocutors, where one performs the role of a *teacher* seeking to teach the other interlocutor who acts in the role of a *student*. We then can describe a dialog tutoring session formally as a sequence of turns $\mathcal{H} = (u_1, \ldots, u_{|\mathcal{H}|})$ that are taken by either of the interlocutors. Each turn $u_t \in \mathcal{V}^*$ is a finite sequence of tokens from a vocabulary $\mathcal{V}$.

Further, each turn u_t can be associated with a sequence of dialog acts $\mathbf{a}_t \in \mathcal{A}$ that indicate the action taken by the interlocutor in the corresponding turn. The dialog act is a key aspect of dialog tutoring as it can refer to the teaching strategy employed by the tutor. These may include strategies such as *providing a hint* or *seeking a clarification* (see Appendix A for more details). The set of dialog acts $\mathcal{A}$ is usually fixed according to a predefined taxonomy and may be split into two subsets $\mathcal{A} = \mathcal{A}_{\text{teacher}} \cup \mathcal{A}_{\text{student}}$, each corresponding to the teacher and student role. Each dialog session $\mathcal{H}$ may also be accompanied with some *grounding* information K, which grounds the response in relevant information and may refer to the teaching material that needs to be taught to the student. This information K may come in various formats, including images and videos. However, we restrict ourselves to using only text-based grounding in this work such that $K \in \mathcal{K} \subseteq \mathcal{V}^*$ is again a sequence of tokens from the common vocabulary $\mathcal{V}$ and $\mathcal{K}$ is used to describe the set of possible groundings (e.g., a textbook with a set of chapters). In Section 3 we derive different methods to model the role of the teacher, to which we restrict this work.

CIMA (Stasaski et al., 2020)	*TSCC (Caines et al., 2020)*
 **K:** “blue” is “blu“ [...] **Grammar Rules:** Adjectives (such as color words) follow the noun they modify in Italian [...]	**N/A**
Teacher: (N/A) “Blue" is "blu" in Italian. **Student:** But what are the other words? **Teacher:** (N/A) Can you give me your best guess? **Student:** es en front de blu tree. **Teacher:** *(Correction)* Getting there. Remember that the adjective always follows the noun in modifies.	**Teacher:** *(eliciting)* So in fact fractions (half/third/quarter etc) are good to use for variety in language OK? and what about e.g. 23%? **Student:** just less than a quarter **Teacher:** *(eliciting)* so if you say 'less' you need to say 'less than' ....so just use one word ok? beginning with 'u'! **Student:** I am not sure of the word. **Teacher:** (scaffolding) just under a quarter

Figure 1: Examples of tutoring conversations from both datasets. The (image) grounding is shown in the second row and dialog acts in brackets indicate the pedagogical strategy.

2.1 Existing tutoring datasets

To our knowledge, only three conversational tutoring datasets are openly available: *CIMA* (Stasaski et al., 2020) is a crowd-sourced dataset, where annotators were asked to role-play students and teachers by working through an exercise on translating a prepositional phrase from English to Italian, given an image and a shared set of concepts. *TSCC* (Caines et al., 2020) uses real teachers leading one-on-one language tutoring sessions in English language learning, thus creating a more open-ended scenario. Finally, *TalkMoves* (Suresh et al., 2022a).

is a collection of scraped classroom transcripts of K-12 mathematics lesson videos that contain challenging, multi-party interactions.

The scarcity of tutoring datasets stands in contrast to other dialog scenarios, where plenty of datasets have been proposed and studied. For example, task-oriented dialog has been studied in domains like reservations (Wen et al., 2017; Budzianowski et al., 2018; Kim et al., 2020) or public service information (Feng et al., 2020). On the other hand, chit-chat or open-domain dialog has been studied on movies (Zhou et al., 2018), Wikipedia knowledge (Dinan et al., 2019), agent persona (Dinan et al., 2020), knowledge graphs (Moon et al., 2019), and open-ended settings (Komeili et al., 2022).

Furthermore, we note the following limitations and characteristics of tutoring datasets, also in comparison to other dialog domains: 1) Low pedagogical quality (CIMA), 2) Limited teaching strategies (all), 3) Exclusive focus on classroom settings (TalkMoves), 4) Small dataset size (all). 5) Significantly larger context sizes (TSCC) 6) Harder readability according to the Flesch score (TSCC). We provide more evidence in Table 1 which shows a comparison of dialog tutoring datasets with widely-used task-oriented and open-domain datasets.

2.2 Related work on generative dialog models

Similarly, while the advent of large pretrained models has sparked ample research on generative models for dialog (Bao et al., 2021; Peng et al., 2021; Roller et al., 2021; Shuster et al., 2022; Cohen et al., 2022), this has not carried over to research on tutoring systems, where existing solutions are predominantly rule-based and do not generate open-ended responses. For example, the authors on CIMA define heuristics to select responses (Stasaski et al., 2020). Pretrained transformers in general have only very recently been studied in this setting, however only for dialog act classification (Suresh et al., 2022b) and to study the pedagogical ability of existing large pretrained models (Tack and Piech, 2022).

3 Dialog Tutoring Models

After introducing the dialog tutoring task, this section highlights the models we evaluate on the task. We note that our aim is an analysis of existing models.

We explore turn-level models that can generate a teacher response $\mathbf{y} := u_{t+1}$ given a tutoring session $\mathcal{H} = (u_1, \ldots, u_{|\mathcal{H}|})$. During training, we obtain the dialog history by teacher forcing, i.e., we take the ground-truth dialog history. Furthermore, we do not model the problem of retrieving grounding information but rather assume it as given.

Generative Model In order to study if generative models can capture a *given* teaching strategy, we first derive a model that assumes the ground-truth dialog act sequence $\mathbf{a} = \{\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathcal{H}|}\}$ to be given as an input. Then, given dialog history $\mathcal{H}_{<t} = \{u_1, \ldots, u_t\}$, grounding information K and $\mathbf{a}_{t+1} \subseteq \mathcal{A}_{\text{teacher}}$, the set of dialog acts relevant at timestep $t + 1$, the teacher response $\mathbf{y}$ is generated according to a locally-normalized language generation model. In the case that no grounding information K is given, the dependency on K may be dropped.

$$\begin{aligned} \mathbf{y}^{\star} &= \operatorname*{argmax}_{\mathbf{y} \in \mathcal{V}^*} \{p\left(\mathbf{y} \mid \mathbf{a}_{t+1}, \mathcal{H}_{<t}, K\right)\} \qquad (1) \\ &= \operatorname*{argmax}_{\mathbf{y} \in \mathcal{V}^*} \prod_{i=1}^{|\mathbf{y}|} \{p\left(y_i \mid \mathbf{y}_{<i}, \mathbf{a}_{t+1}, \mathcal{H}_{<t}, K\right)\} \end{aligned}$$

We separate the turns in the dialog by special ⟨teacher⟩ and ⟨student⟩ tags and prepend the dialog act as a special token, followed by a special ⟨knowledge⟩ token and the grounding information K as the input to the encoder. In CIMA we encode the triples defining the grounding information in a simple natural language format, where we separate the English and Italian words for an object, color, and preposition as well as the whole phrase by the word "is", for example as "blue is blu" in Figure 1. Further, we add the grammar rules separated by a special token. We study different models to parametrize p that are described in Section 4.

Finally, we use the version of **CTRL** (Keskar et al., 2019) presented by Rashkin et al. (2021). The aim of the model is to improve the faithfulness of grounded response generation models, a significant problem in neural language generation (Roller et al., 2021) which holds high importance in the field of tutoring, where one trusts a teacher to present correct information. The model is augmented by a sequence of control tokens that are intended to steer the generations to desirable properties. We use the *lexical overlap* and *entailment tokens*, which we obtain as follows. In training, the lexical overlap is measured on a token-level between ground-truth response and grounding. Then, three equally sized buckets are created indicating

Dataset	Train samples	#DA	Tgt. length	Src. length	#prev. turns	corpus-div.	Flesch score	F1($\hat{\mathbf{y}}, K$)
CIMA	2,715	5	14.71	9.70	4.55	0.149	84.64	0.196
TSCC	5,845	23	16.09	11.72	68.28	0.327	73.00	-
MultiWoZ 2.1	56,781	34*	19.86	14.49	7.86	0.069	90.90	-
Schema-Guided Dialog	164,982	10*	14.30	10.36	11.38	0.049	95.37	-
DSTC9	19,184	-	21.61	11.65	11.70	0.050	81.85	0.47
Personachat	127,162	-	12.26	11.65	6.51	0.162	91.80	0.10
FaithDial	18,357	-	21.72	17.33	4.54	0.274	83.28	0.47
CMU_DoG	81,468	-	14.49	18.23	18.73	0.178	79.54	0.02

Table 1: Statistics of dialogue datasets with lines separating groups by task - tutoring, task-oriented, open-domain dialog tasks. Target length and source length in average number of tokens (Bart tokenizer), # prev. turns is averaged for each teacher response, corpus-div is ngram entropy averaged for uni to four-grams. * We only count system dialog acts.

low, medium, and high overlap which is indicated by a control token. Entailment is determined by an MNLI model and again a corresponding token is added. At test time, we always use the token that encourages the desirable property, in this case high lexical overlap and entailment. Finally, using a sequence of control tokens $\mathbf{c}$, the model from equation 1 becomes:

$$p\left(\mathbf{y} \mid \mathbf{a}_{t+1}, \mathbf{c}, \mathcal{H}_{<t}, K\right) \tag{2}$$

Joint Model In order to study how well current neural models can decide on a reasonable teaching strategy and perform in real case scenarios, we also implement a model that first decides the dialog act $a_{t+1} \in \mathcal{A}_{\text{teacher}}$ (instead of assuming the ground-truth dialog act) and then uses it to generate a response $\mathbf{y} = u_{t+1}$. We use a simple model that again takes the grounding and dialog context as input but now generates the concatenation of dialog act and response in one utterance, akin to SOLOIST (Peng et al., 2021). Thus, for a given $\tilde{\mathbf{y}} := \mathbf{a}_{t+1} \circ \mathbf{y}$ with act sequence $\mathbf{a}_{t+1}$ of length N and response $\mathbf{y}$ of length T, the model is

$$p\left(\tilde{\mathbf{y}} \mid K, \mathcal{H}_{<t}\right) = \prod_{i=1}^{m+N} p\left(\tilde{y}_i \mid \tilde{\mathbf{y}}_{<i}, K, \mathcal{H}_{<t}\right) \tag{3}$$

In training, we use teacher forcing and prepend $\mathbf{a}_{t+1}$ to $\mathbf{y}$ to obtain the label sequence. At test time, the model performs a beam search over the dialog act sequence and response jointly.

Retrieval-based model Since generative models are known to produce erroneous outputs that are factually incorrect and potentially inappropriate (Ji et al., 2022), we also experiment with using a retrieval-based model that selects responses from the training corpus at test time. As opposed to previous work on the topic (e.g., Stasaski et al. (2020)), we do not employ a rule-based model but rather a learned retrieval model that does not require handcrafting elaborate and possibly brittle rules. Therefore, we use the **Bi-Encoder** architecture (Mazaré et al., 2018; Dinan et al., 2019) where a dialog context encoder $\text{enc}_{\mathcal{H}_{<t};\theta}$ and a response encoder $\text{enc}_{\mathbf{y};\theta}$ encode context $\mathcal{H}_{<t}$ and possible responses $\mathbf{y}$ into a fixed size vector of same dimension n. In our experiments, the weights θ of both encoders are shared.

The model is trained using contrastive learning. Suppose we are given a training pair $\mathcal{H}, \hat{\mathbf{y}}$ from a training dataset $\mathcal{D}$ that we use for teacher forcing. We then train the model by sampling a negative response $\bar{\mathbf{y}}$ from the set of responses in $\mathcal{D}$ and using the Triplet Loss criterion, which for a metric function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is defined as:

$$\begin{aligned}\mathcal{L}(\theta; \mathcal{H}, \hat{\mathbf{y}}, \bar{\mathbf{y}}) = [m + d(\text{enc}_{\mathcal{H};\theta}(\mathcal{H}), \text{enc}_{\mathbf{y};\theta}(\hat{\mathbf{y}}))\\ - d(\text{enc}_{\mathcal{H};\theta}(\mathcal{H}), \text{enc}_{\mathbf{y};\theta}(\bar{\mathbf{y}}))]_+,\end{aligned} \tag{4}$$

where m is a margin hyperparameter, and d is the euclidean norm in our experiments. Further, we do stratified sampling on CIMA to not select negatives from the same preposition, color, or object that might be false negatives. At test time, given a dialog context $\mathcal{H}_{<t}$, we choose a response $\mathbf{y}^\star$ from the training set $\mathcal{D}$ by maximum inner product search using the decision rule

$$\mathbf{y}^\star = \underset{\mathbf{y} \in \mathcal{D}}{\operatorname{argmax}} \{\text{enc}_{\mathcal{H};\theta}(\mathcal{H}_{<t})^T \text{enc}_{\mathbf{y};\theta}(\mathbf{y})\}. \tag{5}$$

4 Experiments

We use the following models for parameterizing p in Equation 2: A **sequence-to-sequence** model (Sutskever et al., 2014) with a copy mechanism (Gu et al., 2016) trained from scratch. A wide range of pretrained Transformers, namely **BART** (Lewis et al., 2020), **DialoGPT** (Zhang et al., 2020), **T5** (Raffel et al., 2020) and its multilingual version **mT5** (Xue et al., 2021).

	CIMA			**TSCC**	
Model	sBLEU / BLEU-1 (↑)	BERT F1 (↑)	Q^2 (↑)	sBLEU / BLEU-1 (↑)	BERT F1 (↑)
Rule-based (Stasaski et al., 2020)*	0.34/-	0.45	-	-	-
LSTM (Stasaski et al., 2020)*	0.31/-	0.53	-	-	-
Seq2seq	2.89 / 28.0	0.676	0.372	-	-
DialoGPT	4.12 / 35.6	0.697	0.571	0.63 / 18.5	0.661
Bi-Encoder (RoBERTa-base)	5.89 / 23.9	0.690	0.344	1.367 / 8.8	0.638
CTRL (BART-base)	5.99 / 42.5	0.702	0.673	-	-
t5-small	7.36 / 34.0	0.672	0.676	2.72 / 12.1	0.646
BART-large	8.61 / 38.7	0.715	0.673	1.85 / 13.7	0.658
BART-base	9.58 / **42.5**	**0.726**	**0.680**	**2.67 / 18.6**	**0.670**
mt5-small	**11.26** / 41.0	0.700	0.624	1.80 / 14.9	0.653
BART-base†	5.61 / 41.03	0.707	0.642	1.90 / 15.4	0.659
BART-large†	5.65 / 42.67	0.694	0.607	1.74 / 15.1	0.660

Table 2: Comparison of models on CIMA and TSCC. We note that the strong sacrebleu differences are caused by the brevity penalty (all generative models generate too short sequences), †: use predicted dialog act label, others use ground-truth. * numbers taken from (Stasaski et al., 2020) which may not be comparable as there is no standard split of CIMA dataset.

BART and T5 are pretrained encoder-decoder models that were trained on denoising and text-to-text tasks, respectively. mT5 bases on T5 but is multilingual which might help with the code-switched utterances in CIMA. Lastly, DialoGPT is an autoregressive language model based on GPT-2 (Radford et al., 2019) that was pretrained on a large dialog dataset obtained from Reddit. With this, we intend to study whether large-scale dialog-specific pretraining can aid in training educational tutors, as well.

Implementation Details We implement our experiments using the Huggingface transformers library and finetune the checkpoints provided as part of it for all Transformer-based models. For these models, we use an initial learning rate of 3.25×10^{-5}, 500 warmup steps and linear learning rate decay. We train the models using a batch size of 8 and evaluate on the validation sets after each epoch. In the end, we select the best model to be the one that has a minimal loss on the validation set. The sequence-to-sequence baseline is trained from scratch using an initial learning rate of 0.001 for 25,000 steps using the Adam optimizer and a dropout rate of 0.1 We use beam search with a beam size of 10 to generate model responses.

4.1 Dataset splits

Since there are no official dataset splits for CIMA and TSCC, we split both datasets randomly into training, validation and test sets. We provide the exact split of the dataset in an accompanying code repository. For CIMA, we use all such samples with less than three annotated tutor responses for training. The other conversations are split randomly into equally-sized validation and test sets which results in 2715/300/300 samples each.

For TSCC, we split randomly along the conversations to obtain 82/10/11 training, validation, and test conversations each.

4.2 Evaluation metrics

To evaluate our models, we use the BLEU implementation provided by the sacrebleu package (sBLEU) (Post, 2018) to measure lexical overlap between generated and ground-truth response. Furthermore, we use BERT F1 (BERTScore) to measure their semantic similarity. Lastly, for CIMA we also calculate Q^2 (Honovich et al., 2021) which measures the factual consistency of the response $\mathbf{y}$ with the grounding information K by employing a question-answering based matching. Both BERTScore and Q^2 have shown strong correlation with human judgements on factual consistency Honovich et al. (2022).

5 Results

In this section, we summarize our main findings in terms of automatic evaluation. First, we give an overview of the performance of different models that we train on CIMA and TSCC in Section 5.1. Then, we assess their ability to stay faithful to teaching strategies (Section 5.2) and study how grounding annotations can influence the faithfulness of neural dialog tutors (Section 5.3), before studying their scaling behavior with dataset size and complexity (Section 5.4) and their generalization capabilities (Section 5.5). We then finish with an assessment of using education-specific data for pretraining (Section 5.6).

	Method				
	GT	BART$_{\text{base}}$	BART$_{\text{large}}$	CTRL	Retrieval
DA F1	78.3	**81.0**	70.1	63.0	43.1

Table 3: F1 score of the dialog act classification based on the generated responses of our models.

5.1 Comparison of different models

Table 2 shows the key results from our experiments. First, all automatic metrics are *significantly higher* on CIMA, which indicates that the models can fit CIMA much better than TSCC, with which current approaches still struggle. We further analyse this finding in Section 5.2 and show that this is because TSCC has *richer teaching strategies which are harder to model*. Our comparison also suggests that finetuning large pretrained *Transformer models generally gives better results than the rule-based and LSTM model* reported in (Stasaski et al., 2020), and our implemented retrieval and sequence-to-sequence baselines. This illustrates the potential of LLMs for dialog tutoring.

We also see a significant difference among different LLMs. Dialog-specific pretraining of DialoGPT does not help and gives worse results than BART and T5, primarily because the model tends to generate short and generic responses more often. Multilingual pretraining in mT5 improves over T5 only in some metrics, notably in BLEU and BERT F1 on CIMA but not in terms of Q^2. Similarly, adding control tokens to BART does not improve Q^2 or other automatic metrics. Surprisingly, using very large models actually degrades performance in our experiments. Finally, the last two rows show results obtained with our joint model that does not use the ground-truth dialog act but predicts it together with the response sequence and still provides reasonable performance.

5.2 How well can generative models capture teaching strategies?

We study this question first by evaluating the dialog act prediction accuracy of our joint model. We find that it is *significantly low* on TSCC with 21.8 compared to 71.2 on CIMA for BART-base which indicates significant room for improvement. Notably, the joint model tends to predict more frequently occuring dialog acts, which results in fewer follow-up questions and "Other" never being predicted in CIMA, the least frequent act in the data. The distribution of dialog acts in the ground-truth annotations and model predictions with a BART-base joint model is in Figure 2.

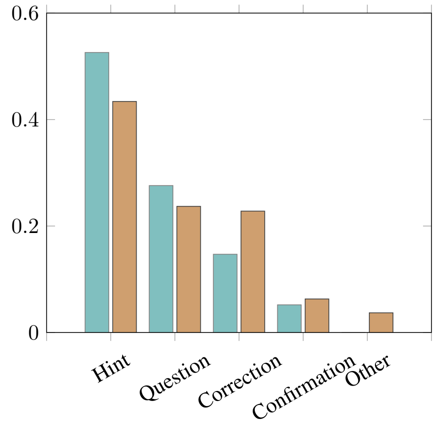


Figure 2: Distribution of predicted and ground-truth dialog acts on CIMA.

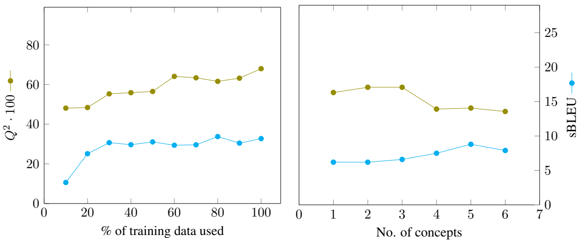
Model	sBLEU (↑)	BERT F1 (↑)	Q^2 (↑)
BART-base	6.69 / 38.6	0.718	0.571
+ triples	9.20 / **45.3**	**0.730**	0.642
+ grammar rules	**9.58** / 42.5	0.726	**0.680**

Table 4: Comparison of models with different inputs on CIMA. Triples are made up of preposition, object, and color translations. Grammar rules are a textual description of a learning concept.

Then, we evaluate how well different models can stick to a given ground-truth dialog act by predicting the dialog act of the *generated response* with a BART-base model trained to predict the ground-truth dialog act sequence based on the ground-truth response. The results are shown in Table 3. Notably, *BART-base performs better than the ground-truth annotations*. The CTRL model, on the other hand, has worse performance since the control tokens do not respect tutoring principles (e.g., lexical overlap to grounding discourages follow-up questions in favor of just giving hints).

5.3 Does grounding in learning concepts help?

Prior work has shown that grounding responses in relevant data can improve their quality, especially in terms of faithfulness (Shuster et al., 2021). We intend to validate this for dialog tutoring by studying three models with different inputs on CIMA. The first model is not provided grounding information, whereas the second and third are grounded in learning concepts (cf. Equation 1) with one using only the (preposition, object, color) triples and the other making use of additional grammar rules. The results with these models are shown in Table 4 and suggest that *grounding responses in relevant knowledge helps the model to produce better and*



(a) (b)

Figure 3: Performance of BART-base on CIMA as a function of: (a) training data size uniformly sampled from the training data, (b) the number of concepts, where only the specific number of concepts is retained and all others are excluded.

Concept	#Samples	Full data	Zero-shot	Zero-shot without grounding
	train/test	Q^2	Q^2	Q^2
is behind the	549/90	0.698	0.603	0.533
is in front of the	735/84	0.616	0.512	0.500
is next to the	547/51	0.497	0.539	0.483
is on top of the	224/30	0.683	0.578	0.567
is under the	270/24	0.854	0.646	0.625
is inside of the	390/21	0.579	0.643	0.190
all concepts	2715 / 300	0.644	0.570	0.502

Table 5: Performance of a grounded BART-base model by learning concept. Full data uses the entire training data and zero-shot removes the concept of the row from the training data.

Method	sBLEU	BERT F1	Q^2
BART-base	6.69 / 38.6	0.718	0.571
+ Ed. data	**7.31 / 41.4**	**0.727**	0.577
+ Non-Ed. data	6.60 / 39.4	0.721	**0.583**

Table 6: Influence of pretraining on educational and non-educational data. Please note that no grounding information is used in this setting.

more faithful responses.

5.4 How do models scale with more data?

Due to the limited availability of high-quality pedagogical datasets and the time-consuming process of authoring new materials (MacLellan and Koedinger, 2020), it is important to understand how quickly generative models can generalize to new settings. Thus, we assess how well the model can model tutoring in low-resource scenarios. We construct a study, where we randomly sample subsets of the CIMA training set and test the performance of the various models. We can see from Figure 3(a) that with more training data, the faithfulness of responses appears to improve and is not saturated before we reach the full training set. This supports the intuition that *additional training data might improve the performance further*.

Similarly, we study how well our model can deal with an increase in complexity with respect to learning concepts at similar training data sizes. Therefore, we construct different training datasets with 735 samples and a varying number of concepts each time. We begin by taking samples concerned with the concept “in front of the” and evaluate exclusively on it, gradually adding new concepts. Figure 3(b) suggests that Q^2 drops sharply at four concepts. BLEU on the other hand increases, and this might be due to the metric encouraging generic utterances that, for example, repeat a grammar rule.

5.5 Can models generalize to new concepts?

As the students progress and gain new knowledge, it might be a desirable property of dialog tutoring models to be able to handle new concepts that suit this increase in prior knowledge. Hence, we study how well our CIMA model can generalize to new concepts that it has not seen in training, for example, a new preposition. For this analysis, we create a set-up where we first train the model on all of the training data and evaluate on the subset of samples for each preposition separately. We then compare this number to a model that is not trained on the corresponding concept it is evaluated on, creating a zero-shot set-up which we carry out for a grounded and ungrounded response generation model. As measured by Q^2 (cf. Table 5), this model can indeed *generalize to new concepts well, albeit with performance degradation*. Furthermore, *grounding information improves generalization* as these define the learning concept (in this case the preposition) and how it is used. Without this information, we observe that the model generates generic responses more often.

5.6 Does education-specific pre-training help?

As educational data are widely available on the internet, next we study how education-specific pretraining effect results. In Table 6, we show results obtained with finetuning a BART-base model directly on CIMA and pretraining it on tutoring dialogs from TSCC or non-tutoring dialogs from MultiWoZ 2.1 (Eric et al., 2020), Personachat (Zhang et al., 2018), CMU DoG (Zhou et al., 2018), DSTC9 (Kim et al., 2020) and Topicalchat (Gopalakrishnan et al., 2019). In both cases, we only see *minor improvements*, which may be explained by the different dataset settings and the lack of a unified dialog act taxonomy.

6 Human Evaluation

We further evaluate previously assessed models with human judgments firstly by obtaining quality estimates according to different criteria and secondly by conducting a simulation study, where expert annotators are asked to provide novel rewritings of existing conversations and to categorize errors made by the model.

6.1 Quality of the generated responses

We perform a human quality evaluation of the generated response for four models - retrieval (Bi-Encoder), BART-base, BART-base$_{\mathrm{CTRL}}$ and the joint model (BART-base). A randomly chosen subset of the CIMA test set conversations were annotated by 4 annotators (with one annotator speaking C1 level Italian). All annotators labeled 60 examples in total, of which 20 overlapped. To further distinguish the quality of training data for the models, we annotated ground-truth responses on a small sample of 20 examples. We evaluate the following criteria on a 3-point Likert scale (disagree to completely agree) and outline our findings in the following, as shown in Figure 4.

Fluency *"The response is grammatically correct and fluent."* We find that all models have very high fluency scores.

Coherence *"The response naturally follows up on previous utterance and context and has no logical conflicts with the context or DA label."* We find that all generative models are able to produce coherent responses but not the retrieval model.

Correctness *"The response is factually correct and respects learning concepts being taught."* All models score comparable to ground-truth responses on the constrained CIMA dataset. It is noteworthy, however, that a response may be correct in itself but not coherent with the context or the grounding (often the case in the retrieval model), and this could explain the discrepancy between correctness and our automatic Q^2 scores.

Equitable tutoring *"The response gives a learning opportunity for the student by providing space for reflection, explanation, pointing to follow-up challenge, or engaging student in other ways."* Here we find significant deficiencies not only for our evaluated models but notably also for the annotated ground-truth responses (gt). This might explain the insufficiencies in the responses as they reflect this distributional behavior of the training data. We think that future dataset collections should take better care of this property and resort to more expert annotators as opposed to crowdsourcing.

Quality Attribute	sBLEU	BERTScore
Fluency	0.14	0.12
Coherence	0.17	0.26
Correctness	0.06	0.15
Equitable Tutoring	0.08	0.16

Table 7: Pearson correlation coefficients between the human judgements on our quality criteria and automatic metrics.

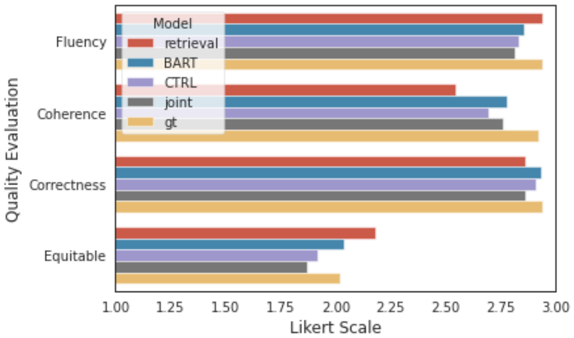


Figure 4: Comparison of models on four criteria (reporting M) in the human quality evaluation. We observed high SD for coherence and equitable metrics.

Furthermore, Table 7 shows that our automatic metrics correlate poorly with human judgements.

6.2 User study with a learning interface

Lastly, we seek to study how well dialog tutoring models can perform in a realistic setting with questions obtained from real users (containing out-of-distribution samples) and not the fixed dataset. Therefore, we randomly sampled conversations from the CIMA test set. We asked two C1-level expert Italian speakers to 1) rephrase these conversations using a conversational dialogue interface and 2) assign erroneous model responses to predefined error categories. The interface used in the qualitative evaluation is shown in Figure 6. We obtain all model responses from the BART-base model that first predicts the dialog act and then the response. Error categories adopted from previous work (Bommasani et al., 2021) describe the ideal behavior of tutoring models as simulating the behavior of good human teachers along two dimensions:

Understanding *"Being able to understand and reason about student solutions, misconceptions, and learning concepts."* We find that of the 20 modified conversations, 45% exhibit *Understanding errors*, such as an incorrect solution assessment or incorrect translations.

Pedagogy *"Being able to use effective pedagogy to instruct students."* We find that 10% of the responses exhibit *Pedagogical errors*, for example telling the correct solution directly without offering any engagement point to the student.

50% of the conversations were labeled good by the annotators. Examples of the conversations are available in Table 8.

7 Discussion: Towards More Equitable and Faithful Tutoring Systems

In this section, we outline directions of research that we think can be important steps towards more equitable and faithful tutoring models. Namely, we first address the small scale and quality of current tutoring datasets and cast doubt on the crowdsourcing data quality checks. Then, we suggest ways of improving the underperformance of both equitable tutoring and teaching strategy prediction identified in current generative models under these constraints by drawing from learning sciences literature. Finally, we outline desiderata for more reliable dialog evaluation of neural tutoring models.

Datasets Based on the analysis in §2.1 and Table 1, we think that the community will benefit from a dataset that lies between CIMA and TSCC in terms of its difficulty. Moreover, the low equitable tutoring scores of CIMA's ground-truth responses indicate that crowdsourcing with untrained annotators can lead to low pedagogical quality. A similar observation has been found by human evaluation for the TSCC dataset (Tack and Piech, 2022). Finally, we encourage the establishment of better dialog act taxonomies that are backed by learning sciences research. As outlined in §5.6 and in He et al. (2022), a unified taxonomy may also strongly aid in transfer learning.

Models So far, dialog tutoring models have only covered limited domain-specific settings linked to a particular activity, such as learning Italian prepositions or solving math word problems. We argue that the community could benefit from working on problems common to learning in general, for example tracking problem-solving states and modeling pedagogies used by teachers. Here, knowledge tracing (Corbett and Anderson, 1994) (the problem of estimating students' skill mastery level) could be used for tracking problem-solving states and increasing the coherence of dialog tutoring conversations and dialog act selection performance which would contribute to better modeling of global teaching strategies. Furthermore, validated instruction quality coding schemes (Michaels et al., 2010; Hennessy et al., 2016) used by classroom teachers can be computationally modeled (Demszky et al., 2021; Ganesh et al., 2021) and incorporated into models.

We also think that recently proposed constrained decoding approaches that can balance between multiple criteria (Qin et al., 2022) hold great promise in improving faithfulness in complex tutoring dialogs. Finally, as data collection is labor-intensive in expert domains, we see great potential in few-shot learning methods, such as prompt-based methods (Schick and Schütze, 2022).

Evaluation Our experiments highlight the insufficiencies of current automatic dialog evaluation metrics, as both BLEU and BertScore show comparatively low correlation with our collected human judgements from §6.1. This is in line with previous research (Mehri and Eskenazi, 2020; Mehri et al., 2022) and shows the necessity not only for better automatic evaluation metrics but also for verification based on human judgements or user studies that should incorporate criteria relevant to tutoring (e.g., equitable tutoring outcomes). Metrics that incorporate task success, which have been used in task-oriented dialog systems (Budzianowski et al., 2018), are a promising direction of future research for automatic evaluation.

8 Conclusion

In this work, we reflected on the state of research in dialog tutoring and explored the potential of neural generative models in this domain. We found some promising initial results with these models in comparison to rule- or retrieval-based methods. However, we also established limitations of currently available benchmarks and evaluation criteria. Furthermore, we showed that there are a number of challenges that need to be addressed before neural generative models of text can be deployed as intelligent tutoring systems on a larger scale, such as controllability and being able to model a sound pedagogical strategy. Based on these findings, we outline potential avenues for future research.

Limitations

A key limitation of our work is the use of only two available tutoring datasets. Despite a limited number of datasets available in this domain, using the TalkMoves dataset (Suresh et al., 2022a) could help further generalize our findings. This remains an avenue for future work.

Based on the prior work, we focused on the specific conversational goal of dialog tutors which is providing learning aid for students' skill development and more opportunities to learn. While this is the most widespread type (Wollny et al., 2021), it is not covering all the goals of human tutors, and other aspects could be important, for example, rapport-building or mentoring on the meta-cognitive level. We acknowledge this both as a prerequisite for our work and at the same time as a limitation. For further discussion we refer the reader to Appendix B and C.

Finally, our user study could be further extended with more participants. In the future, we plan a more comprehensive study with real language learners using an end-to-end dialog tutoring system.

Ethics Statement

We do not foresee any significant harm directly as a result of our work. Having said that, we must understand that automatic tutoring is a high-stake setting that can pose significant harm if appropriate care is not taken before the deployment of these systems. Issues of biases and lack of trust, and other ethical issues such as privacy concerns must be considered. Considering learners only as data points within a neural dialog tutoring context may prevent us from seeing the societal and socioeconomic barriers that they may be up against, thereby running the risk of not only failing to help relevant learner subgroups but also sometimes giving additional privileges to those who use these systems.

9 Acknowledgements

This project was made possible by an ETH AI Center Doctoral Fellowship to Jakub Macina with partial support by the Asuera Stiftung and the ETH Zurich Foundation and has received funding by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE. We thank the group members and our reviewers for their valuable feedback.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *ArXiv preprint*, abs/2001.09977.

Siqi Bao, Huang He, Fan Wang, Hua Wu, Haifeng Wang, Wenquan Wu, Zhen Guo, Zhibin Liu, and Xinchao Xu. 2021. PLATO-2: Towards building an open-domain chatbot via curriculum learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2513–2525, Online. Association for Computational Linguistics.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Andrew Caines, Helen Yannakoudakis, Helena Edmondson, Helen Allen, Pascual Pérez-Paredes, Bill Byrne, and Paula Buttery. 2020. The teacher-student chatroom corpus. In *Proceedings of the 9th Workshop on NLP for Computer Assisted Language Learning*, pages 10–20, Gothenburg, Sweden. LiU Electronic Press.

Michelene TH Chi, Nicholas De Leeuw, Mei-Hung Chiu, and Christian LaVancher. 1994. Eliciting self-explanations improves understanding. *Cognitive science*, 18(3):439–477.

Michelene TH Chi and Ruth Wylie. 2014. The icap framework: Linking cognitive engagement to active learning outcomes. *Educational psychologist*, 49(4):219–243.

Aaron Daniel Cohen, Adam Roberts, Alejandra Molina, Alena Butryna, Alicia Jin, Apoorv Kulshreshtha, Ben Hutchinson, Ben Zevenbergen, Blaise Hilary Aguera-Arcas, Chung ching Chang, Claire Cui, Cosmo Du, Daniel De Freitas Adiwardana, Dehao Chen, Dmitry (Dima) Lepikhin, Ed H. Chi, Erin Hoffman-John, Heng-Tze Cheng, Hongrae Lee, Igor Krivokon, James Qin, Jamie Hall, Joe Fenton, Johnny Soraker, Kathy Meier-Hellstern, Kristen Olson, Lora Mois Aroyo, Maarten Paul Bosma, Marc Joseph Pickett, Marcelo Amorim Menegali, Marian Croak, Mark Díaz, Matthew Lamm, Maxim Krikun, Meredith Ringel Morris, Noam Shazeer, Quoc V. Le, Rachel Bernstein, Ravi Rajakumar, Ray Kurzweil, Romal Thoppilan, Steven Zheng, Taylor Bos, Toju Duke, Tulsee Doshi, Vinodkumar Prabhakaran, Will Rusch, YaGuang Li, Yanping Huang, Yanqi Zhou, Yuanzhong Xu, and Zhifeng Chen. 2022. Lamda: Language models for dialog applications. In *arXiv*.

Albert T Corbett and John R Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4:253–278.

Dorottya Demszky, Jing Liu, Zid Mancenido, Julie Cohen, Heather Hill, Dan Jurafsky, and Tatsunori Hashimoto. 2021. Measuring conversational uptake: A case study on student-teacher interactions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1638–1653, Online. Association for Computational Linguistics.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition*, pages 187–208. Springer.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428, Marseille, France. European Language Resources Association.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8118–8128, Online. Association for Computational Linguistics.

Scott Freeman, Sarah L Eddy, Miles McDonough, Michelle K Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. 2014. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the national academy of sciences*, 111(23):8410–8415.

Ananya Ganesh, Martha Palmer, and Katharina Kann. 2021. What would a teacher do? Predicting future talk moves. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4739–4751, Online. Association for Computational Linguistics.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinlang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. 2019. Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations. In *Proc. Interspeech 2019*, pages 1891–1895.

Arthur C Graesser. 2016. Conversations with autotutor help students learn. *International Journal of Artificial Intelligence in Education*, 26(1):124–132.

Arthur C Graesser, SIDNEY D'MELLO, and Natalie Person. 2009. Meta-knowledge in tutoring. In *Handbook of metacognition in education*, pages 373–394. Routledge.

Arthur C Graesser, Natalie K Person, and Joseph P Magliano. 1995. Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied cognitive psychology*, 9(6):495–522.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, et al. 2022. Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. *Proceedings of the AAAI Conference on Artificial Intelligence*.

Sara Hennessy, Sylvia Rojas-Drummond, Rupert Higham, Ana María Márquez, Fiona Maine, Rosa María Ríos, Rocío García-Carrión, Omar Torreblanca, and María José Barrera. 2016. Developing a coding scheme for analysing classroom dialogue across educational contexts. *Learning, culture and social interaction*, 9:16–44.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. TRUE: Re-evaluating factual consistency evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3905–3920, Seattle, United States. Association for Computational Linguistics.

Or Honovich, Leshem Choshen, Roee Aharoni, Ella Neeman, Idan Szpektor, and Omri Abend. 2021.

Q^2: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7856–7870, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation. *ACM Computing Surveys*.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *ArXiv preprint*, abs/1909.05858.

Seokhwan Kim, Mihail Eric, Karthik Gopalakrishnan, Behnam Hedayatnia, Yang Liu, and Dilek Hakkani-Tur. 2020. Beyond domain APIs: Task-oriented conversational modeling with unstructured knowledge access. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 278–289, 1st virtual meeting. Association for Computational Linguistics.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-augmented dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Diane J Litman, Carolyn P Rosé, Kate Forbes-Riley, Kurt VanLehn, Dumisizwe Bhembe, and Scott Silliman. 2006. Spoken versus typed human and computer dialogue tutoring. *International Journal of Artificial Intelligence in Education*, 16(2):145–170.

Christopher J MacLellan and Kenneth R Koedinger. 2020. Domain-general tutor authoring with apprentice learner models. *International Journal of Artificial Intelligence in Education*, pages 1–42.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium. Association for Computational Linguistics.

Shikib Mehri, Jinho Choi, Luis Fernando D'Haro, Jan Deriu, Maxine Eskenazi, Milica Gasic, Kallirroi Georgila, Dilek Hakkani-Tur, Zekang Li, Verena Rieser, et al. 2022. Report from the nsf future directions workshop on automatic evaluation of dialog: Research directions and challenges. *ArXiv preprint*, abs/2203.10012.

Shikib Mehri and Maxine Eskenazi. 2020. USR: An unsupervised and reference free evaluation metric for dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 681–707, Online. Association for Computational Linguistics.

Sarah Michaels, Mary Catherine O'Connor, Megan Williams Hall, and Lauren B Resnick. 2010. Accountable talk sourcebook: For classroom conversation that works. *Pittsburgh, PA: University of Pittsburgh Institute for Learning*.

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84, Copenhagen, Denmark. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy. Association for Computational Linguistics.

Johanna D Moore, Kaska Porayska-Pomsta, Sebastian Varges, and Claus Zinn. 2004. Generating tutorial feedback with affect. In *FLAIRS Conference*, pages 923–928.

Benjamin D Nye, Arthur C Graesser, and Xiangen Hu. 2014. Autotutor and family: A review of 17 years of natural language tutoring. *International Journal of Artificial Intelligence in Education*, 24(4):427–469.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. 2022. COLD decoding: Energy-based constrained text generation with langevin dynamics. In *Advances in Neural Information Processing Systems*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hannah Rashkin, David Reitter, Gaurav Singh Tomar, and Dipanjan Das. 2021. Increasing faithfulness in knowledge-grounded dialogue with controllable features. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 704–718, Online. Association for Computational Linguistics.

Brian J. Reiser. 2004. Scaffolding Complex Learning: The Mechanisms of Structuring and Problematizing Student Work. *Journal of the Learning Sciences*, 13(3):273–304. Publisher: Routledge _eprint: https://doi.org/10.1207/s15327809jls1303_2.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online. Association for Computational Linguistics.

Jeremy Roschelle and Stephanie D Teasley. 1995. The construction of shared knowledge in collaborative problem solving. In *Computer supported collaborative learning*, pages 69–97. Springer.

Sherry Ruan, Liwei Jiang, Justin Xu, Bryce Joe-Kun Tham, Zhengneng Qiu, Yeshuang Zhu, Elizabeth L. Murnane, Emma Brunskill, and James A. Landay. 2019. Quizbot: A dialogue-based adaptive learning system for factual knowledge. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, page 357. ACM.

Timo Schick and Hinrich Schütze. 2022. True few-shot learning with prompts—a real-world perspective. *Transactions of the Association for Computational Linguistics*, 10:716–731.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, et al. 2022. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *ArXiv preprint*, abs/2208.03188.

Tanmay Sinha and Manu Kapur. 2021. When problem solving followed by instruction works: Evidence for productive failure. *Review of Educational Research*, 91(5):761–798.

Katherine Stasaski, Kimberly Kao, and Marti A. Hearst. 2020. CIMA: A large open access dialogue dataset for tutoring. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–64, Seattle, WA, USA â†’ Online. Association for Computational Linguistics.

Abhijit Suresh, Jennifer Jacobs, Charis Harty, Margaret Perkoff, James H Martin, and Tamara Sumner. 2022a. The talkmoves dataset: K-12 mathematics lesson transcripts annotated for teacher and student discursive moves. *arXiv preprint arXiv:2204.09652*.

Abhijit Suresh, Jennifer Jacobs, Margaret Perkoff, James H. Martin, and Tamara Sumner. 2022b. Fine-tuning transformers with additional context to classify discursive moves in mathematics classrooms. In *Proceedings of the 17th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2022)*, pages 71–81, Seattle, Washington. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Anaïs Tack and Chris Piech. 2022. The AI Teacher Test: Measuring the Pedagogical Ability of Blender and GPT-3 in Educational Dialogues. In *The 15th International Conference on Educational Data Mining*, page accepted.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Sebastian Wollny, Jan Schneider, Daniele Di Mitri, Joshua Weidlich, Marc Rittberger, and Hendrik Drachsler. 2021. Are we there yet?-a systematic literature review on chatbots in education. *Frontiers in artificial intelligence*, 4.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

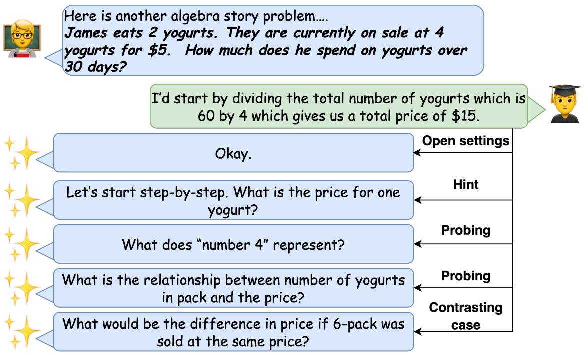
Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Ran Zhao, Tanmay Sinha, Alan W Black, and Justine Cassell. 2016. Socially-aware virtual agents: Automatically assessing dyadic rapport from temporal patterns of behavior. In *International conference on intelligent virtual agents*, pages 218–233. Springer.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

A Pedagogical strategy and dialog acts in dialog tutoring



Here is another algebra story problem....

James eats 2 yogurts. They are currently on sale at 4 yogurts for $5. How much does he spend on yogurts over 30 days?

I'd start by dividing the total number of yogurts which is 60 by 4 which gives us a total price of $15.

Okay.

Let's start step-by-step. What is the price for one yogurt?

What does "number 4" represent?

What is the relationship between number of yogurts in pack and the price?

What would be the difference in price if 6-pack was sold at the same price?

Figure 5: Example dialogue between a tutor and a student solving an algebra story problem. Key questions are: What teacher pedagogical strategies are the best in terms of learning gains of students? How to adapt language models to generate pedagogically valid responses?

In the context of this paper, we assume that the pedagogical strategy is represented using dialog act annotations. An example of the teacher strategy is providing hints (cf. example in Figure 5), where a teacher provides helpful support or clarifies goals to the student. Another example is Probing (cf. example in Figure 5), which prompts students to explain better or reflect on the current solution. CIMA contains five teacher dialog acts - *hint, open-ended question, correction, confirmation, other*. TSCC contains more fine-grained dialog acts such as *eliciting, scaffolding, enquiry, or recap*.

From a learning science standpoint, pedagogical strategy could be viewed as a global strategy (knowing how to effectively guide students e.g. using questioning or providing contrasting cases) and dialog acts as a specific decision on how this strategy is implemented on the local turn-based level.

B Equitable tutoring

Although tutoring is typically conceived as a scenario where a subject matter expert works synchronously with one or multiple students and takes interpretive authority, there is increasing empirical evidence supporting the case for incorporating active learning approaches in the classroom (Freeman et al., 2014; Sinha and Kapur, 2021). With collaborative creation of knowledge where teachers position themselves as co-learners and students also take interpretive authority, such approaches are better poised to build classroom equity than monologic educational practices where only one voice (primarily the teacher's) tends to be heard, legitimized and sometimes imposed. Therefore, if we rethink of the goals of education as providing opportunities for students to enter into the workforce with a positive identity about themselves and the subject matter, equitable tutoring via increased student chances to pose ideas, construct knowledge and as a result feel welcomed into the intellectual discussion, holds tremendous promise.

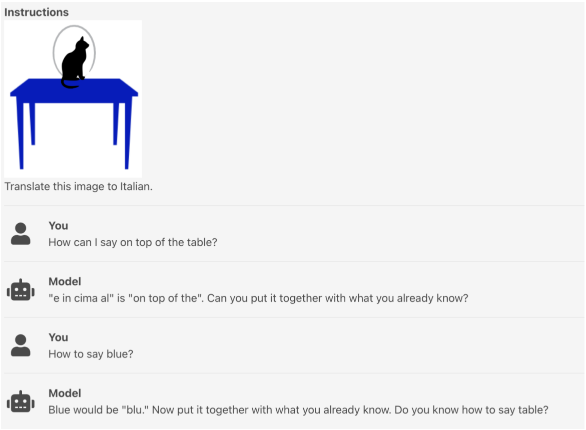
C Conversational Goals

In this work, we studied only 1:1 dialog tutoring settings with a specific focus on the role of a teacher/tutor. We focused on the most commonly used goal of dialog tutoring which is a learning aid to support students' skill development and provide opportunities to learn (Wollny et al., 2021).

However, teacher-student interactions may entail multiple conversational goals that can serve interactional functions (e.g., turn-taking) and interpersonal functions (e.g., rapport-building moves such as self-disclosure, praise, social norm violation). Research in human tutoring and collaborative learning, more generally, has shown that how students manage the task space (consisting of the problem to be solved, e.g., do I agree with the interlocutor's reasoning?) and the relational space (consisting of the interactional challenges and opportunities, e.g., can I disagree without threatening the interlocutor's face?) is critical to learning outcomes (Roschelle and Teasley, 1995; Zhao et al., 2016). Neural dialog tutoring, which can account for this crucial, but fundamental distinction among conversational goals is challenging.

D Qualitative user evaluation

The interface used in the qualitative evaluation is shown in Figure 6.



Instructions

Translate this image to Italian.

You
How can I say on top of the table?

Model
"e in cima al" is "on top of the". Can you put it together with what you already know?

You
How to say blue?

Model
Blue would be "blu." Now put it together with what you already know. Do you know how to say table?

Figure 6: User interface used for the qualitative evaluation. The interface is based on ParlAI (Miller et al., 2017) library.

Context	Generated teacher response
T: Please try to fill in the blank in Italian. S: how do you say next? T: Remember that "is next to the" is "e vicino" S: I don't know how to say purple. Is it viola?	Viola is the word for purple.
T: Please try to fill in the blank in Italian. S: how do you say next? T: Remember that "is next to the" is "e vicino" S: Purple is viola, right?	That is correct.
T: Please try to fill in the blank in Italian. S: how do you say next? T: Remember that "is next to the" is "e vicino" S: I think purple could be violett.	Viola is the word for purple.
T: Please try to fill in the blank in Italian. S: how do you say next? T: Remember that "is next to the" is "e vicino" S: I can't remember the color. Can you give me a hint?	Purple is viola.
T: Please try to fill in the blank in Italian. S: how do you say next? T: Remember that "is next to the" is "e vicino" S: I guess it is e vicino all'albero. Is it correct to start the sentence with the preposition?	The preposition for "purple" is "viola" **Understanding - Meaning mismatch**
T: "Blue" is "blu". Please try to fill in the blank in Italian. S: But what are the other words? T: Can you give me your best guess? S: e di fronte all'albero blu (**correct**)	Almost, tree is all'albero'. Please try again. **Understanding - Incorrect solution assessment**
T: "Blue" is "blu". Please try to fill in the blank in Italian. S: But what are the other words? T: Can you give me your best guess? S: e di fronte blu all'albero (**adjective position swap**)	Almost, you need to add blu after "all'albero" to make the word "tree". **Understanding - Incorrect translation**
T: "Blue" is "blu". Please try to fill in the blank in Italian. S: But what are the other words? T: Can you give me your best guess? S: e di fronte al albero blu (**wrong definite article**)	Almost, tree is all'albero not albero.
T: "Blue" is "blu". Please try to fill in the blank in Italian. S: But what are the other words? T: Can you give me your best guess? S: e di fronte al cane blu (**noun swap**)	Almost. Al cane is not the correct word for tree.
T: "Blue" is "blu". Please try to fill in the blank in Italian. S: But what are the other words? T: Can you give me your best guess? S: e dietro all'albero blu" (**wrong preposition**)	The correct answer is e di fronte all'albero blu. **Pedagogy - Information reveal**

Table 8: Examples of rephrased conversations from the qualitative study. T refers to a teacher utterance, S refers to a student utterance. Bold text is information for the reader indicating error categories.

Evaluating the Robustness of Discrete Prompts

Yoichi Ishibashi[1] **Danushka Bollegala**[2] **Katsuhito Sudoh**[1] **Satoshi Nakamura**[1]
[1] Nara Institute of Science and Technology [2] University of Liverpool
`{ishibashi.yoichi.ir3, sudoh, s-nakamura}@is.naist.jp`
`danushka@liverpool.ac.uk`

Abstract

Discrete prompts have been used for fine-tuning Pre-trained Language Models for diverse NLP tasks. In particular, automatic methods that generate discrete prompts from a small set of training instances have reported superior performance. However, a closer look at the learnt prompts reveals that they contain noisy and counter-intuitive lexical constructs that would not be encountered in manually-written prompts. This raises an important yet understudied question regarding the *robustness* of automatically learnt discrete prompts when used in downstream tasks. To address this question, we conduct a systematic study of the robustness of discrete prompts by applying carefully designed perturbations into an application using AutoPrompt and then measure their performance in two Natural Language Inference (NLI) datasets. Our experimental results show that although the discrete prompt-based method remains relatively robust against perturbations to NLI inputs, they are highly sensitive to other types of perturbations such as shuffling and deletion of prompt tokens. Moreover, they generalize poorly across different NLI datasets. We hope our findings will inspire future work on robust discrete prompt learning.[1]

1 Introduction

Pre-trained Language Models (PLMs) have been successfully adapted to a wide range of Natural Language Processing (NLP) tasks using *prompt-based* learning (Radford et al., 2018, 2019; Brown et al., 2020; Petroni et al., 2019) such as sentiment classification (Gao et al., 2021), natural language inference (NLI) (Schick and Schütze, 2021, 2022), relation extraction (Shin et al., 2020), cross-lingual inference (Qi et al., 2022). However, manually writing prompts that generalize well is very challenging for several reasons such as (a) it might not always be possible to recruit domain-expert human annotators, (b) human annotators might not be able to cover all corner cases by writing prompts, and (c) there can be disagreements between human annotators regarding the coverage of a particular prompt. To address these challenges, automatic learning of discrete prompts has been proposed such as AdvTrigger (Wallace et al., 2019), AutoPrompt (**AP**; Shin et al., 2020), WARP (Hambardzumyan et al., 2021), and RLPrompt (Deng et al., 2022).

Although discrete prompt learning methods have achieved good performance in numerous downstream tasks by automatically learnt prompts, such automatic prompts seem to be significantly different from the manually-written ones. For example, Table 1 shows manually-written and AP-learnt prompts for fact retrieval (Petroni et al., 2019). We see that the AP-learnt prompts for BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) outperform the manual prompts in precision1 (P@1) scores. However, the AP-learnt prompts contain various counter-intuitive language constructs such as punctuation (e.g. '(', '?', '!', ')'), spelling errors (e.g. *commuenrug*) etc., which seem unrelated to the target relation. Similar cases can be observed for AP-learnt prompts for other tasks as well (see Appendix in Shin et al. (2020)). It is unrealistic that a human annotator would be able to write such prompts even if they were able to see the same training instances as used by automatic methods.

Considering the fact that discrete prompt learning methods are trained in a few-shot setting where they use only a small number of training instances, the seemingly counter-intuitive nature of the discrete prompts learnt by automatic methods raises concerns about their robustness. For example, *How will the performance of a target task change if we add small random perturbations to the prompts learnt by AP?* and *Whether the prompts learnt by AP generalize to out-of-domain data?*. To study these issues, in this paper we evaluate the robust-

---

[1] Our codes and the adversarial NLI dataset are available at `https://github.com/LivNLP/prompt-robustness`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2373–2384
May 2-6, 2023 ©2023 Association for Computational Linguistics

Relation	**Method**	**Prompt**	**P@1**
native-language-of (P103)	Manual	`The native language of [X] is [Y]`	74.54
	AP BERT	`[X]PA communerug speaks proper [Y]`	**84.87**
	AP RoBERTa	`[X]neau optionally fluent!?ẗraditional [Y]`	81.61
profession-of (P106)	Manual	`[X] is a [Y] by profession`	0.73
	AP BERT	`[X] supporters studied politicians musician turned [Y]`	15.83
	AP RoBERTa	`[X] (), astronomers businessman·former [Y]`	**19.24**
music-played-by (P136)	Manual	`[X] plays [Y] music`	0.7
	AP BERT	`[X] freaking genre orchestra fiction acid [Y]`	**59.95**
	AP RoBERTa	`[X] blends postwar hostage drama sax [Y]`	52.97

Table 1: Examples of prompts learnt by AP for the fact retrieval task for BERT and RoBERTa PLMs and the human-written manual prompts. T-REx relation ids are shown with brackets for each relation type. Precision@1 (P@1) scores are shown when each prompt is used in fact retrieval.

ness of discrete prompts learnt by automatic prompt learning methods and compare that with manually-written prompts and direct fine-tuning of PLMs.

An evaluation of the robustness of discrete prompts is important for two main reasons. First, given that discrete prompt learning methods are learning those prompts from a small set of training instances, it is important that they cover the core patterns that generalize to the target task and not simply capture some random artefacts in the training samples. Second, unlike embedding-based continuous prompts (Li and Liang, 2021; Lester et al., 2021), discrete prompts (Wallace et al., 2019; Shin et al., 2020; Deng et al., 2022) are represented in natural language and supposed to be interpretable. However, if a discrete prompt learning method is less robust, a seemingly harmless perturbation such as removing a punctuation character can significantly alter the performance of a downstream task.

In contrast to the numerous work that has used prompts for fine-tuning PLMs, to the best of our knowledge, the robustness of discrete prompts to random or adversarial perturbations has not been systematically studied. To address this gap, we use AP as a concrete example of a widely-used method and evaluate its robustness under different types of carefully designed perturbations. However, we note that our perturbation techniques are not limited to AP and can be used for any discrete prompt learning method. We compare the performance of AP-learnt prompts against fine-tuning using Manually-written Prompts (MP), and Head-based Fine-Tuning (HFT), where we fine-tune both the classifier head and the PLM parameters.

From our evaluation, we find several interesting facts about the robustness of discrete prompts as summarized below.

- Overall, when the number of training instances is increased, MP outperforms both AP and HFT on CB (De Marneffe et al., 2019) and MNLI (Williams et al., 2018), two independent benchmark datasets for NLI (§ 3.1). In particular, the performance of AP on MNLI is much worse than that on CB. This is in contrast to the superior performance of AP on SICK-E (Marelli et al., 2014), another NLI dataset, as reported by Shin et al. (2020).

- Moreover, we see a performance drop when we use discrete prompts learnt from CB for MNLI and vice-versa (§ 3.4). These results indicate that the performance of discrete prompts learnt by AP is highly dataset-dependent and such discrete prompts do not generalize well across datasets.

- Compared to MP, AP-learnt discrete prompts turn out to be highly sensitive to the ordering of prompt tokens (§ 3.2).

- Random deletion of prompt tokens decreases performance in both AP and MP (§ 3.3).

- We create an adversarial NLI dataset from randomly-sampled test instances from MNLI and CB, and manually modify the hypothesis sentences with keeping the corresponding premise sentences unchanged, such that (a) the target label would not change, and (b) would reverse an entailment label to a contradiction (or vice-versa). Both AP and MP remain relatively robust against the perturbations that do not change the target label, but the performance of MP drops significantly in the label-changing setting (§ 3.5). This shows that AP is relatively more robust against adversarial perturbations than MP, which explains AP's superior performance in various tasks.

2 Related Work

Prompting Methods: Prompting or *in-context-learning* has received wide attention as an efficient method to extract knowledge from PLMs (Brown et al., 2020; Petroni et al., 2019; Cui et al., 2021). However, to manually write prompts one must possess task-specific domain knowledge. As an alternative, methods that can automatically learn prompts from training data have been proposed. Two distinct types of prompts have been learnt in prior work: discrete prompts (learns lexical sequences), and continuous prompts (learns embeddings). Continuous prompts (Li and Liang, 2021; Lester et al., 2021) are parameter efficient because they learn generalizable task-specific embeddings, with performance comparable to PLM fine-tuning. However, continuous prompts cannot be learnt when a PLM is publicly unavailable and the only access to it is via an API (Brown et al., 2020). Moreover, compared to discrete prompts, continuous prompts are difficult to interpret. Learning discrete prompts (Wallace et al., 2019; Shin et al., 2020; Deng et al., 2022) does not suffer from these limitations of continuous prompts and can be used with diverse NLP tasks. Especially, fine-tuning massive PLMs has become computationally costly, which has made discrete prompt learning an attractive alternative.

Analysis of Prompting Methods: Prior work has analyzed prompts from various viewpoints. Scao and Rush (2021) studied the effect of training dataset size on fixed-prompt PLM fine-tuning and head-based fine-tuning and found that prompting is often worth 100s of instances on average across classification tasks. Kavumba et al. (2022) showed that the performance of prompt-based models varies significantly depending on the surface cues in the sentence. Lu et al. (2022) found that ordering of task input significantly affects the performance. Utama et al. (2021) focused on the reliance on lexical overlap in sentence pair classification and showed that prompt-based models fail to make predictions dependent on the lexical overlap. To the best of our knowledge, the robustness of discrete prompts under different types of perturbations has not been studied in prior work, which is the main focus of this paper.

3 Experiments

Let us first describe experimental settings common to all experiments.

Prompting and Fine-Tuning Methods: We compared the following methods.

- **AutoPrompt** (**AP**; Shin et al., 2020) is a representative method of discrete prompt learning. The learning strategy is based on fill-in-the-blank task (Devlin et al., 2019). First, a manually created prompt template (e.g., `[X] <MASK> <T> ... <T> [Y]`) is given, and a prompt token (called a trigger token) is learnt by replacing `<T>`, which is a special token representing a trigger token. In the search for trigger tokens, the probability of `<MASK>` is converted into class probability by using label tokens (e.g., {'*nobody*', '*nor*'} for contradiction (Shin et al., 2020)), and trigger tokens are searched by gradient-guided search (Wallace et al., 2019) to find a candidate set consisting of trigger tokens from a vocabulary of the language model. As a template for NLI, we used the one given by Shin et al. (2020), and the prompt tokens were learnt from the training dataset. In our experiments, we used the official implementation.[2]

- **Manually-written Prompts** (**MP**; Schick and Schütze, 2021) is a method for fine-tuning the entire masked language model with training data using manually-written prompts as the input and predicting the `<MASK>` tokens for the labels (e.g., '*yes*' for entailment). We used the template `{hypothesis}? | <MASK>, {premise}` and verbalizer ('*yes*' for entailment, '*no*' for contradiction, '*maybe*' for neutral) following prior work (Schick and Schütze, 2021; Scao and Rush, 2021). Schick and Schütze (2021) proposed an ensemble-based method with multiple rounds of fine-tuning using different templates. However, because a single template is used in AP, for a fair comparison in our experiments, we fine-tuned a PLM using one MP template.

- **Head-based Fine-Tuning** (**HFT**; Devlin et al., 2019) fine-tunes the PLM with a classifier head. We report the head-based results trained by Scao and Rush (2021). They

---

[2] https://github.com/ucinlp/autoprompt

trained HFT with a low learning rate (10^{-5}) and always with a large number of steps (at least 250), following the recommendations in prior work (Mosbach et al., 2021; Zhang et al., 2021). Note that HFT is not a prompt-based method, so it was excluded from some experiments on the robustness of discrete prompts.

Datasets: We used NLI as an evaluation task to compare the robustness of discrete prompting methods. The NLI task has been used in multiple previous studies to evaluate and/or propose novel prompt learning methods because it is a fundamental task related to many NLP applications (Shin et al., 2020; Scao and Rush, 2021; Webson and Pavlick, 2022). It is important to use the same NLI task and datasets in our experiments to facilitate fair comparisons and reach reproducible conclusions. We used the two datasets: CommitmentBank (**CB**; De Marneffe et al., 2019)[3] (a corpus of short texts), and Multi-Genre Natural Language Inference Corpus (**MNLI**; Williams et al., 2018)[4] (a crowdsourced collection of sentence pairs for NLI). Each sentence pair is labelled with *entailment*, *neutral*, or *contradiction*.

PLM: In our experiments, we used the same pre-trained language model to evaluate AP, MP, and HFT equally. Specifically, we used RoBERTa-large (355M parameters) [5] (Liu et al., 2019), which has been used in much prior work in prompt learning (Shin et al., 2020; Scao and Rush, 2021). The PLM was trained on five datasets, including BookCorpus[6], English Wikipedia[7], CC-News[8], OpenWebText[9], and Stories[10]. The texts were tokenised using a byte-level Byte-Pair Encoding (BPE; Sennrich et al., 2016) vocabulary of size 50,000.

Evaluating the Robustness of Prompts: We used *rate of degradation* (**RoD**) (Meyers et al., 2020) to evaluate robustness, which is defined as the decrease in accuracy of the target task due to the perturbations added to the prompt. If the RoD of a model is small after the inclusion of a perturbation, the model is considered to be robust against that perturbation. Specifically, we first calculate the respective accuracies acc_x and acc_{x^*} on the same evaluation set for both prompt x and its perturbated version x^*. Using the average accuracies avg-acc_x and avg-acc_{x^*} over M prompts $x_1, ..., x_M$, we calculate the RoD as $(\text{avg-acc}_x - \text{avg-acc}_{x^*})/\text{avg-acc}_x = 1 - \text{avg-acc}_{x^*}/\text{avg-acc}_x$.

3.1 Effect of the Training Dataset Size

Before moving on to robustness experiments, we first investigate the number of training instances on which AP and MP perform best, and used the best-performing AP and MP to evaluate their robustness in the subsequent experiments.

Experimental Settings: We gradually increased the size of the training dataset following the experimental setup of Scao and Rush (2021). Specifically, we experimented with randomly sampled subsets of the training dataset having varying numbers of instances in $\{10, 15, 20, 30, 50, 70, 100, 150, 200\}$. Because the performance of few-shot learning methods often varies due to the high feature variance in the training data, we randomly sampled four subsets per each dataset size and used them independently for training the models[11] (i.e. trigger tokens and label tokens for AP, or fine-tuned language model for MP and HFT) for each subset and report the average accuracy on the validation data for the four models ($M = 4$). We used the matched (example from the same source as the training set) validation set for MNLI. For CB, we held out 50 training instances for development as in Scao and Rush (2021) and evaluated the original validation set as test data.

We searched for the optimal values for the following hyperparameters: the number of trigger tokens in {3, 5, 10}, the number of label tokens in {3, 5, 10}, and the number of tokens in a candidate set in {10, 50}. We evaluated the test accuracy using the hyperparameters that had the highest accuracy on the validation data for each dataset size. In the training of MP, we used AdamW optimizer (Loshchilov and Hutter, 2019) with an initial learning rate of 10^{-5} and a learning step of 1,000 following Mosbach et al. (2021).

Main Results: Figure 1 shows the performance[12] against the training dataset size. We see that in both CB and MNLI **MP is always superior to AP**. For example, with a dataset of size 200, AP and MP

---

[3] `https://super.gluebenchmark.com/tasks`

[4] `https://cims.nyu.edu/~sbowman/multinli/`

[5] `https://huggingface.co/roberta-large`

[6] `https://yknzhu.wixsite.com/mbweb`

[7] `https://en.wikipedia.org/wiki/English_Wikipedia`

[8] `https://commoncrawl.org/2016/10/news-dataset-available/`

[9] `https://github.com/jcpeterson/openwebtext`

[10] `https://arxiv.org/abs/1806.02847`

[11] NVIDIA RTX A5000 was mainly used.

[12] HFT results were obtained from Scao and Rush (2021), F1-macro for CB and accuracy for MNLI.

Method	#Train	Template	#Prompt tokens	#Label tokens per class	Avg. accuracy	
					CB	MNLI
AP	200	p <MASK> <T> ... <T> h	10	3	68.3	37.7
MP	200	h ? | <MASK> , p	3	1	95.1	65.5
HFT	-	<CLS> p <SEP> h	0	-	-	-

Table 2: The average accuracy of the experiment with four training subsets of 200 instances. Red represents the task inputs, h represents the hypothesis, p represents the premise, blue represents the prompt tokens, and <T> represents a trigger token. Unreported values were marked with '-'.

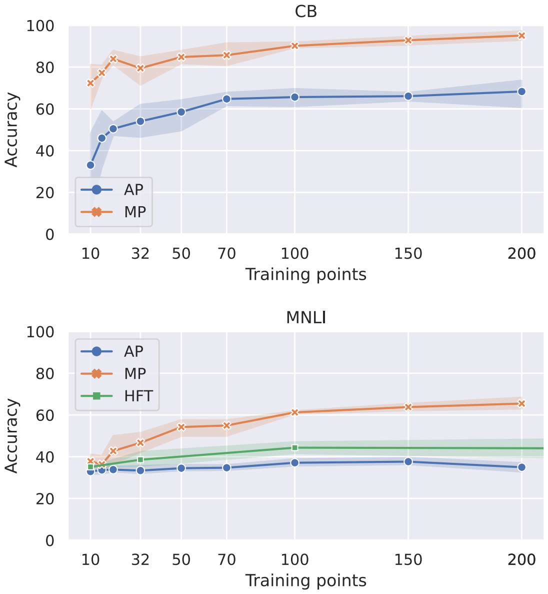


Figure 1: Performance of AutoPrompt (AP), Manually-written Prompt (MP), and Head-based Fine-Tuning (HFT) on the scale of dataset size for CB and MNLI. Means and their 95% confidence intervals are plotted. The accuracy of HFT for dataset size for CB was not plotted because the accuracy was not reported.

achieved the best accuracy in CB, MP's accuracy was 92.7%, while that of AP was lower at 54.2%.

Our results also suggest that **the performance of discrete prompts learnt by AP is highly dataset dependent.** Shin et al. (2020) reported results for AP and HFT on SICK-E (Marelli et al., 2014), which is another NLI dataset. They concluded that AP was always superior to HFT up to training dataset sizes of 1,000 for the same RoBERTa-large PLM that we use. However, our experiments show the opposite trend (i.e. HFT is superior to AP). This suggests that even if AP is superior to HFT on a given dataset, it is not guaranteed to be superior in a different dataset for the same task. This may be due to the differences in the domain and annotation guidelines for each dataset. For example, the accuracy of MNLI was quite low on AP, which contrasts with that of CB. This result suggests that the discrepancies in domains and annotation guidelines make it difficult for AP to perform consistently.

Best Prompts: Table 2 shows the average accuracy of models trained on 200 instances that performed well in both CB and MNLI. Note that there are four training subsets for each dataset size, resulting in corresponding four trained AP prompts and four PLMs fine-tuned by MP.[13] In the robustness evaluations in § 3.2 through § 3.5, we used these learnt APs and MPs. In this paper, (a) trigger tokens learnt by AP, and (b) manually-written prompts excluding the task inputs and mask tokens are collectively referred to as the *prompt tokens*.

3.2 Token Reordering

As seen from Table 1, compared to MPs where the ordering of tokens in a prompt is manually determined, discrete prompts learnt by AP appear to have no obvious ordering among their tokens. To empirically investigate the importance of the token order in a discrete prompt, we conduct an experiment where we randomly shuffle the prompt tokens and measure the effect on the downstream task performance.

Experimental Procedure: Given a discrete prompt, we first randomly reordered its prompt tokens (e.g. shaded in blue in Table 2). Next, we used the reordered prompt with the PLM to make entailment predictions for the test instances in the CB and MNLI datasets. Finally, the entailment prediction accuracy (Acc) obtained with the reordered prompts was computed. We repeated this evaluation 10 times for each prompt and report the averaged values and the corresponding RoD values.

Main Results: From Table 3 we see that the accuracy drops for both AP and MP when the prompt

---

[13] We show the four best prompts learnt by AP in Appendix A.

Method	Metrics	CB	MNLI
AP	Acc	**54.2**	**34.3**
	RoD	**0.21**	**0.10**
MP	Acc	<u>92.7</u>	<u>59.3</u>
	RoD	<u>0.03</u>	<u>0.09</u>

Table 3: Performance of reordered prompts. Acc denotes accuracy; RoD denotes the RoD from before the reordering (Table 2). The largest drops in accuracy are **bolded** and the smallest drops are <u>underlined</u> for each method and dataset. AP relies more strongly on word order than MP.

tokens are randomly reordered. In particular, the accuracy of AP drops significantly compared to that of MP. For example, the accuracy of AP on CB drops by ca. 14% due to token reordering, while that for MP drops only by ca. 2%. Intuitively, one would expect that changing the order of prompt tokens in MP would result in a significant drop in accuracy because the meaning of the prompts would change. However, we see that this is not the case. This result shows that **the discrete prompts learnt by AP strongly rely on the token order**.

Additional Analysis: To further analyze the relationship between the level of perturbation introduced by reordering prompt tokens in AP and its effect on the performance, we computed the token-level edit distance (Levenshtein distance; Levenshtein et al., 1966) between each prompt and its token-shuffled version as shown in Figure 2. For all four AP prompts, we see that the accuracy drops when the perturbation noise (i.e. measured by edit distance) increases. This reconfirms the lack of robustness in discrete prompts learnt by AP to the random shuffling of prompt tokens.

3.3 Token Deletion

As seen from Table 1, the discrete prompts learnt by AP perform better than MP. However, it is often difficult to determine the importance of prompt tokens to the target task due to their lack of interpretability (e.g. prompt token '*neau*' in Table 1). To understand the significance of individual prompt tokens to the overall discrete prompt, we conducted an experiment where we systematically deleted one or more prompt tokens at various positions from a given discrete prompt and measure the drop (if any) in the performance of the NLI task.

Experimental Procedure: We evaluated two settings of prompt deletion: *single* and *multiple* token deletion. In the single token deletion setting, we deleted one token at different positions in a given prompt. For AP, we repeated this with each of the four discrete prompts (shown in Table 2) and report the average accuracy. In the multiple token deletion setting, we delete $n \in \{1, 3, 5, 7\}$ prompt tokens following three strategies: *Random-deletion* deletes n prompt tokens randomly, *Front-deletion* deletes n consecutive prompt tokens from the beginning of the prompt, and *Back-deletion* deletes n tokens counted backwards from the end of the prompt. In random-deletion, we ran 100 trials and report the average accuracy. As in the previous experiments, we used four prompts for AP and report the averaged results.

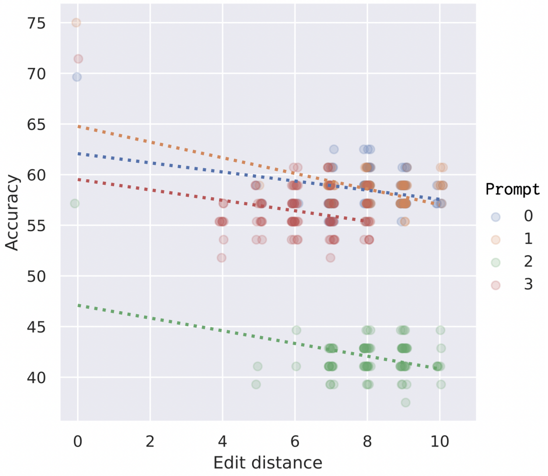


Figure 2: Edit distance and accuracy of the reordered trigger tokens. We evaluated them on the validation data of CB. The prompts numbered 0 through 3 each represent the four prompts learnt by AP (shown in Table 9). Note that a point with an edit distance of zero indicates accuracy with the original trigger token.

Results: From Table 4 we see that the **accuracy of both AP and MP drops even when a single token is deleted** at specific positions. However, the observed trends differ in CB and MNLI. For example, AP resulted in higher RoD values in CB compared to MNLI. This shows that the robustness of AP under single token deletion heavily depends on the dataset. Table 5 shows the results for the multiple token deletion setting. We see that **the performance of both AP and MP degrades when more tokens are deleted.** Interestingly, the accuracy drop in CB is very small for MP even when all prompt tokens are deleted (i.e., only the task inputs and `<MASK>` were used as the input). This suggests that the performance on CB is less reliant on the

Task	Method	Metrics	Position of the deleted prompt token 1	2	3	4	5	6	7	8	9	10	Orig.
CB	AP	Acc	62.1	**61.6**	63.4	59.4	65.6	65.6	62.1	63.8	62.1	62.9	68.3
		RoD	0.09	**0.10**	0.07	0.13	0.04	0.04	0.09	0.07	0.09	0.08	-
	MP	Acc	93.8	**93.3**	96.0	-	-	-	-	-	-	-	95.1
		RoD	0.01	**0.02**	-0.01	-	-	-	-	-	-	-	-
MNLI	AP	Acc	37.9	37.8	**36.6**	37.5	37.5	37.2	37.5	37.4	37.5	37.1	37.7
		RoD	-0.01	0.00	**0.03**	0.01	0.01	0.01	0.01	0.01	0.01	0.02	-
	MP	Acc	64.5	65.4	**55.4**	-	-	-	-	-	-	-	65.5
		RoD	0.02	0.00	**0.15**	-	-	-	-	-	-	-	-

Table 4: Average accuracy was obtained after deleting a single token at different positions of a given prompt. The largest drops in accuracy over the deletion positions are **bolded** and the smallest drops are underlined for each method and dataset. Column 'Orig.' shows the performance of the original prompt.

prompt tokens in MP.

3.4 Cross-Dataset Evaluation

Given that discrete prompt learning methods such as AP learn prompts from a small set of training instances, it is important that the learnt prompts encode generalizable task-specific features and not random artefacts in the training sample used. To study the transferability of the learnt discrete prompts from one dataset to another, we conduct a cross-dataset evaluation as described next.

Experimental Procedure: We used one NLI dataset (e.g. CB) to learn the prompts and then use them to make entailment predictions in another NLI dataset (e.g. MNLI). We then measured the drop in accuracy using RoD for this cross-dataset transferability task with respect to the accuracy of test data from the same dataset.

Results: As seen from Table 6, **AP-based prompts do not generalize well across datasets.** For both AP and MP, RoD is larger in the transfer from CB to MNLI than in the opposite direction. This implies that MNLI is a better dataset for fine-tuning a PLM for NLI using discrete prompts.

3.5 Adversarial Perturbations

Introducing carefully designed adversarial perturbations to the test instances such as modifications to sentences that might or might not alter the original target labels have been used as a technique for probing the robustness of models (Goodfellow et al., 2015). Previous studies (Samanta and Mehta, 2017; Jin et al., 2020) have shown that pre-trained models can be easily fooled to make incorrect predictions with seemingly innocuous perturbations to the test instances. Therefore, we evaluate discrete prompt-based NLI models for their robustness against adversarially perturbated test instances.

Strategy	Method	Metrics	#Deleted Tokens 1	3	5	7	Orig.
CB							
Random	AP	Acc	56.7	56.0	55.4	**54.8**	68.3
		RoD	0.17	0.18	0.19	**0.20**	-
	MP	Acc	**93.3**	94.6	-	-	95.1
		RoD	**0.02**	0.01	-	-	-
Front	AP	Acc	62.1	**49.1**	57.6	57.6	68.3
		RoD	0.09	**0.28**	0.16	0.16	-
	MP	Acc	**93.8**	94.6	-	-	95.1
		RoD	**0.01**	0.01	-	-	-
Back	AP	Acc	62.9	57.6	55.8	**51.3**	68.3
		RoD	0.08	0.16	0.18	**0.25**	-
	MP	Acc	96.0	**94.6**	-	-	95.1
		RoD	-0.01	**0.01**	-	-	-
MNLI							
Random	AP	Acc	**35.8**	**35.8**	36.0	36.2	37.7
		RoD	**0.05**	**0.05**	0.05	0.04	-
	MP	Acc	65.4	**52.6**	-	-	65.5
		RoD	0.0	**0.20**	-	-	-
Front	AP	Acc	37.9	36.5	36.2	**36.0**	37.7
		RoD	-0.01	0.03	0.04	**0.05**	-
	MP	Acc	64.5	**52.6**	-	-	65.5
		RoD	0.02	**0.20**	-	-	-
Back	AP	Acc	37.1	36.7	35.7	**36.5**	37.7
		RoD	0.02	0.03	0.05	**0.03**	-
	MP	Acc	55.4	**52.6**	-	-	65.5
		RoD	0.15	**0.20**	-	-	-

Table 5: Average accuracy was obtained after deleting multiple tokens from a given prompt. The largest drops in accuracy over the deleted tokens are **bolded** and the smallest drops are underlined for each strategy and method.

Method	Test Dataset CB	Test Dataset MNLI	RoD
AP trained on CB	68.3	36.1	0.47
AP trained on MNLI	42.9	37.7	0.12
MP trained on CB	95.1	43.4	**0.54**
MP trained on MNLI	43.8	65.5	**0.33**

Table 6: Accuracy and RoD for the cross-dataset evaluation where a method (AP/MP) is trained on one NLI dataset (CB/MNLI) and the learnt prompts are used to make entailment predictions in a different NLI dataset.

	Hypothesis	Label
Original	The Wither's only had daughters.	contradiction
Perturbation		
w/o label changes	The Wither's did not have sons.	contradiction
w/ label changes	The Wither's had a boy.	entailment

Table 7: Examples of our evaluation set consisting of task inputs with perturbations. The premise sentence is '*The Wither's eldest boy, one of the four of the town militia, saluted in the old style with his stick sword.*'

Evaluation Dataset: For this purpose, we asked two annotators to manually edit hypothesis sentences in NLI test data considering two types of perturbations: (1) perturbations that do not change reference labels, and (2) perturbations that change reference labels. An example is shown in Table 7.

For the first type of perturbation, we edited a hypothesis sentence such that its relationship with the corresponding premise remains unchanged. For the second type, we edited a hypothesis sentence such that its relationship (e.g., from *entailment* to *contradiction*) will be reversed. The premise and hypothesis pairs were sampled from CB (validation set) and MNLI (test set). Because there are ca. 10,000 test instances in MNLI and it is costly to manually edit sentences, we used 100 randomly-chosen sentence pairs covering MNLI and CB.

Experimental Procedure: We computed the RoD of average accuracies obtained with original and adversarial test instances. Specifically, we used the AP prompts in Table 2 under three settings: (a) original (without perturbations), (b) perturbations without label changes, and (c) perturbations with label changes. Then, we calculate RoD from (a) to (b) and (a) to (c) as shown in Table 8.

Results: Overall, we see that the RoD of AP is consistently smaller than that of MP in both CB and MNLI under both types of perturbations. However, it is also clear that the accuracy obtained with AP is much smaller than that with MP. For the perturbations without label changes, both AP and MP show small RoD values, compared to those with label changes.[14] This shows that both AP and MP are relatively robust against modifications to the hypotheses that do not significantly alter the meaning. However, when stronger perturbations are introduced that would result in label changes, the accuracy of both AP and MP drops significantly. [15] This is a concern because it shows that **neither AP nor MP is sufficiently robust to correctly predict the target labels when the hypothesis sentences in test data are adversarially modified.**

Perturbation	Method	Metrics	CB	MNLI
Original	AP	Acc	54.5	40.5
		RoD	-	-
	MP	Acc	95.5	71.0
		RoD	-	-
Perturbation w/o label changes	AP	Acc	55.5	43.2
		RoD	-0.02	-0.07
	MP	Acc	93.0	66.7
		RoD	**0.03**	**0.06**
Perturbation w/ label changes	AP	Acc	42.3	39.4
		RoD	0.22	0.03
	MP	Acc	41.8	61.2
		RoD	**0.56**	**0.14**

Table 8: Accuracy and RoD in prompts for task inputs that include perturbations. The RoD here is the rate of degradation in the average accuracy from the original without perturbations to perturbations without label changes or perturbations with label changes. The largest drops in accuracy are **bolded** and the smallest drops are underlined for each perturbation and method.

4 Conclusion

We investigated the robustness of discrete prompts under different perturbations. We found that although discrete prompts remain relatively robust against token deletion, it is highly sensitive to other types of perturbations such as token shuffling. For adversarial perturbations to the input, discrete prompts were robust to weak perturbations without

---

[14] w/o label change modifications slightly increase the average length of a hypothesis and AP seems to better exploit this extra information for inference resulting in a slight improvement in accuracy (negative RoD).

[15] MP is less robust compared to AP, likely as a result of overfitting to strongly perturbed training data during finetuning the PLM.

label changes, but AP was more robust than MP for perturbations with label changes. Moreover, they generalize poorly across different datasets annotated for NLI. We hope our analysis will inspire future work to develop methods that learn both accurate as well as robust discrete prompts.

5 Limitations

Possible limitations of this work are:

- We chose popular discrete prompt methods of AP and MP and did not investigate other methods in this work. Our analysis procedure can still be applied to other discrete prompts such as AvgTrigger (Wallace et al., 2019).

- We chose RoBERTa-large following previous studies of HFT (Scao and Rush, 2021) and AP (Shin et al., 2020) for reproducible and identical comparisons with them. Other PLMs would lead to different results, but they can also be investigated in the same way as in this work.

- This work focuses on NLI because it is a fundamental natural language understanding task and still difficult even with PLMs (Brown et al., 2020). Other complex downstream tasks are worth investigating for a deeper understanding of prompt-based approaches in future work.

- The results and conclusions are from the English datasets and would differ in other languages. However, our methodologies do not depend on English and can be applied to other languages as important future studies.

- Since there was a performance gap between MP/HFT and AP, the accuracies by the perturbations could be affected. However, this work does not aim to find the best prompt learning method but to analyze the robustness of discrete prompts for a deeper understanding of them.

6 Ethical Considerations

Our adversarial dataset came from existing datasets of CB and MNLI. We visually checked the instances in the data development and found no instances with ethical concerns.

One should also be aware of social biases (e.g. gender stereotypes) in PLM. RoBERTa, the PLM we used in our experiments, is known to have gender biases (Sharma et al., 2021). Since we used it as-is in order to follow the experimental conditions of previous studies using RoBERTa, our current results are possibly influenced by such biases. However, the consideration of the prompt robustness of this work would not pose or magnify such ethical concerns.

7 Acknowledgments

This research was supported by the JSPS KAKENHI (Grants-in-Aid for Scientific Research) JP22H03654.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *CoRR*, abs/2205.12548.

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

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics* *and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4921–4933, Online. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Pride Kavumba, Ryo Takahashi, and Yusuke Oda. 2022. Are prompt-based models clueless? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2333–2352, Dublin, Ireland. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pages 216–223. European Language Resources Association (ELRA).

Bennet Meyers, Michael Deceglie, Chris Deline, and Dirk Jordan. 2020. Signal processing on pv timeseries data: Robust degradation analysis without physical models. *IEEE Journal of Photovoltaics*, 10(2):546–553.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Kunxun Qi, Hai Wan, Jianfeng Du, and Haolan Chen. 2022. Enhancing cross-lingual natural language inference by prompt-learning from cross-lingual templates. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1910–1923, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Jeffery Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Alex Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Suranjana Samanta and Sameep Mehta. 2017. Towards crafting text adversarial samples. *CoRR*, abs/1707.02812.

Teven Le Scao and Alexander M. Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2627–2636. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2022. True few-shot learning with Prompts—A real-world perspective. *Transactions of the Association for Computational Linguistics*, 10:716–731.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Shanya Sharma, Manan Dey, and Koustuv Sinha. 2021. Evaluating gender bias in natural language inference. *CoRR*, abs/2105.05541.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Prasetya Utama, Nafise Sadat Moosavi, Victor Sanh, and Iryna Gurevych. 2021. Avoiding inference heuristics in few-shot prompt-based finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9063–9074, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Albert Webson and Ellie Pavlick. 2022. Do prompt-based models really understand the meaning of their prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 2300–2344. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

A Supplementary Materials/Appendix

Table 9 shows the best prompts learnt by AP, which are used in the robustness evaluations.

Prompt ID	Prompt learnt by AP	Label tokens	Accuracy
0	`p` `<MASK>` `strikers <MASK> <MASK> Ever Want å£« Console Encyclopedia Sie ANC` `h`	**entailment:** 1927, 1897, 1904 **contradiction:** personally, skeptics, squarely **neutral:** æµ, ä¸Ĭ, ä¹	69.64
1	`p` `<MASK>` `diagnoses undert fueling Hist setups prev bound advertisers paper records` `h`	**entailment:** 1930, 1830, 1890 **contradiction:** contradict, straight, favors **neutral:** à¨, annabin, kb	75.00
2	`p` `<MASK>` `maximize useful courts <MASK> malink rooms Scrib home interested Service` `h`	**entailment:** 4000, 1830, THEN **contradiction:** yet, preferring, Ps **neutral:** ĭ, Username, ãĥ«	57.14
3	`p` `<MASK>` `fever <MASK> <MASK> EL <MASK> <MASK> <MASK> ARE ENE cue` `h`	**entailment:** 1890, 1886, 1889 **contradiction:** yet, endorsing, contradict **neutral:** ctory, boolean, Boolean	71.43

Table 9: Four prompts learnt by AP in CB. `Red` represents the task inputs, `h` represents the hypothesis, `p` represents the premise, `blue` represents the prompt tokens (trigger tokens). `<MASK>` tokens in the trigger tokens of some prompts are those used to initialize trigger tokens.

Assessing Out-of-Domain Language Model Performance from Few Examples

Prasann Singhal*, **Jarad Forristal***, **Xi Ye, and Greg Durrett**
Department of Computer Science
The University of Texas at Austin
`{prasanns, jarad, xiye, gdurrett}@cs.utexas.edu`

Abstract

While pretrained language models have exhibited impressive generalization capabilities, they still behave unpredictably under certain domain shifts. In particular, a model may learn a reasoning process on in-domain training data that does not hold for out-of-domain test data. We address the task of predicting out-of-domain (OOD) performance in a few-shot fashion: given a few target-domain examples and a set of models with similar training performance, can we understand how these models will perform on OOD test data? We start from the baseline of looking at model accuracy on the few-shot examples, then investigate how to incorporate analysis of the models' behavior using feature attributions to improve our understanding of generalization. Specifically, we explore a set of "factors" designed to reveal model agreement with certain pathological heuristics that may indicate worse generalization capabilities. On textual entailment, paraphrase recognition, and a synthetic classification task, we show that attribution-based factors can help rank relative model OOD performance. However, accuracy on a few-shot test set is a surprisingly strong baseline, particularly when the system designer does not have in-depth prior knowledge about the domain shift.

1 Introduction

The question of whether models have learned the right behavior on a training set is crucial for generalization. Deep models have a propensity to learn shallow reasoning shortcuts (Geirhos et al., 2020) like single-word correlations (Gardner et al., 2021) or predictions based on partial inputs (Poliak et al., 2018), particularly for problems like natural language inference (Gururangan et al., 2018; McCoy et al., 2019) and question answering (Jia and Liang, 2017; Chen and Durrett, 2019). Unless we use evaluation sets tailored to these spurious signals, accurately understanding if a model is learning them remains a hard problem (Bastings et al., 2021; Kim et al., 2021; Hupkes et al., 2022).

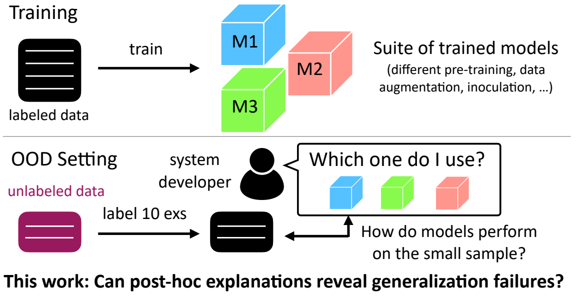


Figure 1: Our setting: a system developer is trying to evaluate a collection of trained models on a small amount of hand-labeled data to assess which one may work best in this new domain. Can baselines / attributions help?

This paper addresses the problem of *predicting* whether a model will work well in a target domain *given only a few examples from that domain.* This setting is realistic: a system designer can typically hand-label a few examples to serve as a test set, but labeling more may be burdensome. Computing accuracy on this small set and using that as a proxy for full-test set performance is a simple baseline for our task, but has high variance, which may cause us to incorrectly rank two models that achieve somewhat similar performance. We hypothesize that we can do better if we can interpret the model's behavior beyond accuracy. With the rise of techniques to analyze post-hoc feature importance in machine-learned models (Lundberg and Lee, 2017; Ribeiro et al., 2016; Sundararajan et al., 2017), we have seen not just better interpretation of models, but improvements such as constraining them to avoid using certain features (Ross et al., 2017) like those associated with biases (Liu and Avci, 2019; Kennedy et al., 2020), or trying to more generally teach the right reasoning process for a problem (Yao et al., 2021; Tang et al., 2021; Pruthi et al.,

---

*Equal contribution

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2385–2397
May 2-6, 2023 ©2023 Association for Computational Linguistics

2022). If post-hoc interpretation can strengthen a models' ability to generalize, can they also help us understand it?

Figure 1 illustrates the role this understanding can play. We have three trained models and are trying to rank them for suitability on a new domain. The small labeled dataset is a useful (albeit noisy) indicator of success. However, by checking model attributions on our few OOD samples, we can more deeply understand model behavior and analyze if they use certain pathological heuristics. Unlike past work (Adebayo et al., 2022), we seek to automate this process as much as possible, provided the unwanted behaviors are characterizable by describable heuristics. We use scalar factors, which are simple functions of model attributions, to estimate proximity to these heuristics, similar to characterizing behavior in past work (Ye et al., 2021a). We then evaluate whether these factors allow us to correctly rank the models' performance on OOD data.

Both on synthetic (Warstadt et al., 2020), and real datasets (McCoy et al., 2019; Zhang et al., 2019), we find that, between models with similar architectures but different training processes, both our accuracy baseline and attribution-based factors are good at distinguishing relative model performance on OOD data. However, on models with different base architectures, we discovering interesting patterns, where factors can very strongly distinguish between different types of models, but cannot always map these differences to correct predictions of OOD performance. In practice, we find probe set accuracy to be a quick and reliable tool for understanding OOD performance, whereas factors are capable of more fine-grained distinctions in certain situations.

Our Contributions: (1) We benchmark, in several settings, methods for predicting and understanding relative OOD performance with few-shot OOD samples. (2) We establish a ranking-based evaluation framework for systems in our problem setting. (3) We analyze patterns in how accuracy on a few-shot set and factors derived from token attributions distinguish models.

2 Motivating Example

To expand on Figure 1, Figure 2 shows an in-depth motivating example of our process. We show three feature attributions from three different models on an example from the HANS dataset (McCoy et al., 2019). These models have (unknown) varied OOD performance but similar performance on the in-domain MNLI data (Williams et al., 2018). Our task is then to correctly rank these models' performance on the out-of-domain HANS dataset in a few-shot manner.

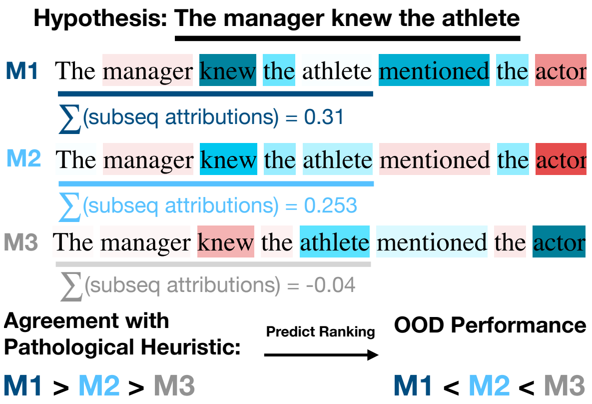


Figure 2: Explanations generated on the same sample for HANS subsequence data models M1, M2, M3 (have ascending OOD performance). The factor (shaded underlines) from knowledge of the OOD allows us to in this example predict the model ranking.

We can consider ranking these models via simple metrics like accuracy on the small few-shot dataset, where higher-scoring models are higher-ranked. However, such estimates can be high variance on small datasets. In Figure 2, only M3 predicts non-entailment correctly, and we cannot distinguish the OOD performance of M1 and M2 without additional information.

Thus, we turn to explanations to gain more insight into the models' underlying behavior. With faithful attributions, we should be able to determine if the model is following simple inaccurate rules called *heuristics* (McCoy et al., 2019). Figure 2 shows the heuristic where a model predicts that the sentence A entails B if B is a subsequence of A. Crucially, we can use model attributions to assess model use of this heuristic :we can sum the attribution mass the model places on subsequence tokens. We use the term *factors* to refer to such functions over model attributions.

The use of factors potentially allows for the automation of detection of spurious signals or shortcut learning (Geirhos et al., 2020). While prior work has shown that spurious correlations are hard for a human user to detect from explanations (Adebayo et al., 2022), well-designed factors could automatically analyze model behavior across a number of tasks and detect such failures.

3 Attributions to Predict Performance

In this section, we formalize the ideas presented thus far. Token-level attribution methods (a subset of post-hoc explanations) are methods which, given an input sequence of tokens $\mathbf{x} \overset{\text{def}}{=} x_1, x_2, ..., x_n$ and a model prediction $\hat{y} \overset{\text{def}}{=} M(\mathbf{x})$ for some task, assign an explanation $\phi(\mathbf{x}, \hat{y}) \overset{\text{def}}{=} a_1, \ldots, a_n$ where a_i corresponds to an attribution or importance score for a corresponding x_i towards the final prediction. For cases where the model, prediction, and inputs are unambiguous, we abbreviate this simply $\phi_i \equiv \phi(\mathbf{x}) \overset{\text{def}}{=} \phi(\mathbf{x}, M_i(\mathbf{x}))$.

We assume that the model is trained on an in-domain training dataset D_T and will be evaluated on some unknown OOD set D_O. Given two models M_0 and M_1, with a small amount of data $D_{(O,t)} \subset D_O$ ($t = 10$ examples or fewer in our settings), our task is to predict which model will generalize better. We break the process into 2 steps (see Figure 2):

1. Hypothesize a heuristic. First we must identify an underlying heuristic H that reflects pathological model behavior in the OOD dataset. For example, the subsequence heuristic in Figure 2 corresponds to a heuristic which always predicts entailed if the hypothesis is contained within the premise. Let $h(M_i)$ abstractly reflect how closely the ith model's behavior aligns with H. Let $s(M_i)$ be the true OOD performance of model M_i. If we then assume that $h(M_i)$ faithfully models some pathological heuristic H, we should have that $h(M_0) > h(M_1) > \ldots > h(M_m)$ implies $s(M_0) < s(M_1) < \ldots < s(M_m)$. In other words, the more a model M_i agrees with a pathological heuristic H, the worse it performs.

2. Measure alignment. We now want to predict the ranking of $s(M_i)$; however, with few labeled examples there may be high variance in directly evaluating these metrics. We instead use factors $f(\mathbf{x}, \phi_i)$ which map tokens and their attributions for model M_i to scalar scores that should correlate with the heuristic H. Factors can be designed to align with known pathological heuristics, where higher scores indicate strong model agreement with the associated heuristic. We then estimate the ranking of $s(M_i)$ using the relative ranking of the corresponding $h(M_i)$ approximated through factors.

Concretely, to measure the alignment, we first compute for each input $\mathbf{x}_j \in D_{(O,t)}$ the prediction $M_i(\mathbf{x}_j)$ and the explanation $\phi(\mathbf{x}_j)$ for that prediction. These $\phi(\mathbf{x}_j)$ are used to compute the score $f(\mathbf{x}_j, \phi(\mathbf{x}_j))$ for model M. We take the overall score of the model to be $F(i) = \frac{1}{t}\sum_{j=1}^{t} f(\mathbf{x}_j, \phi(\mathbf{x}_k, M_i(\mathbf{x}_k)))$, the mean over the t examples in $D_{(O,t)}$. We then directly rank models on the basis of the $F(i)$ values: the higher the average factor value (the more it follows the heuristic), the lower the relative ranking: $F(0) > F(1) \implies s(M_0) < s(M_1)$. Therefore we can sort the models by these values and arrive at a predicted ranking. We later also consider factors which to not intuitively map to specific heuristics.

Baselines We also consider three principle explanation-agnostic baselines. A natural baseline given $D_{(O,t)}$ is to simply use the accuracy (`ACC`) on this dataset: $\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[y_i = M(\mathbf{x_i})]$, however this may be noisy on only a few examples, and frequently leads to ties.[1]

We also assess model confidence (`CONF`), the softmax probability of the predicted label, as well as looking at `CONF-GT`, the softmax probability of only the ground-truth label.

4 Experimental Setup

4.1 Models Compared

In this work, we compare various models across different axes yielding different D_O performance. The first approach we use is **inoculation** (Liu et al., 2019a), which involves fine-tuning models on small amounts or batches of D_O data alongside in-domain data to increase model performance on OOD data. The second approach we use is varying the model architecture and pre-training (e.g., using a stronger pre-trained Transformer model).

In Section 5, we use inoculation to create 5 RoBERTa-base (Liu et al., 2019b) models of varying D_O performance for each of the three MSGS sets. In Section 6 where we consider the HANS and PAWS datasets, we inoculate a variety of models. For HANS, we inoculate 5 RoBERTa-large models. We additionally examine DeBERTa-v3-base (He et al., 2021b,a) and ELECTRA-base (Clark et al., 2020) models fine-tuned on in-domain MNLI data. For PAWS, we inoculate 4 RoBERTa-base models on the in-domain D_T set. We also inoculate ELECTRA-base and DEBERTA-base models. We include complete details for these models in Ap-

---

[1] Most considered datasets are constructed to mislead models following the heuristic, so this baseline directly measures agreement with a heuristic h.

pendix A. The generated models represent a realistic problem scenario: a practitioner may have many different models with similar D_T performance, but different D_O performance. We specifically crafted suites of models which have both near pairs (models with similar D_O performance) and far pairs.

4.2 Attribution Methods

We experiment using several token-level attributions methods: **LIME** (Ribeiro et al., 2016) computes attribution scores using the weights of a linear model approximating model behavior near a datapoint. **SHAP** (Lundberg and Lee, 2017) is similar to LIME, but uses a procedure using Shapley values. Finally, **Integrated Gradients (TOKIG)** (Sundararajan et al., 2017) compute ϕ_i by performing a line integral over the gradients with respect to token embeddings on a path from a baseline token to the ground truth token; commonly, this baseline token is chosen to be `<MASK>`. While intuitively sensible, Harbecke (2021) has voiced concerns regarding the use of TOKIG in NLP.

4.3 Evaluation Setup

Because model ranking using a small $D_{(O,t)}$ may be unstable, we conduct all experiments over a number of different sampled $D_{(O,t)}$ sets. We first sample M examples from each set (in the range of 200-600), then generate explanations for all models on each example. We then take 400-500 bootstrap samples of size n (we report results for $n = 10$, as experimental results were similar for sizes 5 and 20), simulating many few-shot evaluations. For each bootstrap sample, we analyse $\binom{m}{2}$ model pairs. Details can be found in Appendix B.

We define a "success" as a technique correctly ranking a model pair, when measured by D_O performance (on the full set); otherwise is a "failure". We define *pairwise accuracy* as the accuracy for a method ranking a particular model pair across all bootstrap samples. We define *few-shot accuracy* (or just *accuracy*) as the average of the pairwise accuracies over the $\binom{m}{2}$ model pairs. By reporting ranking accuracy across a diverse set of models, we ensure a comprehensive evaluation.

5 MSGS: A Proof of Concept

We first show experiments on the Mixed Signals Generalization Set (MSGS) dataset presented in Warstadt et al. (2020) as a proof of concept for our methodology. MSGS is a synthetic classification dataset. The training (in-domain) set is composed of sentences where both some linguistic feature (e.g., the presence of an adjective) and a spurious surface feature (e.g., the word "*the*" being in the sentence) are always associated with a positive label $y = 1$. This data is ambiguous, which means the model could rely on either the linguistic or surface feature completely yet still get 100% accuracy on in-domain data. Warstadt et al. (2020) then create sets of OOD data where the linguistic feature becomes associated with the $y = 1$ positive label, and the surface feature with a $y = 0$ label. The resulting test accuracy reflects model reliance on one feature or the other. Warstadt et al. (2020) use this to investigate what generalizations are learned at which stages of model pre-training; we investigate whether information from small probe sets can help assess model reliance on the surface feature.

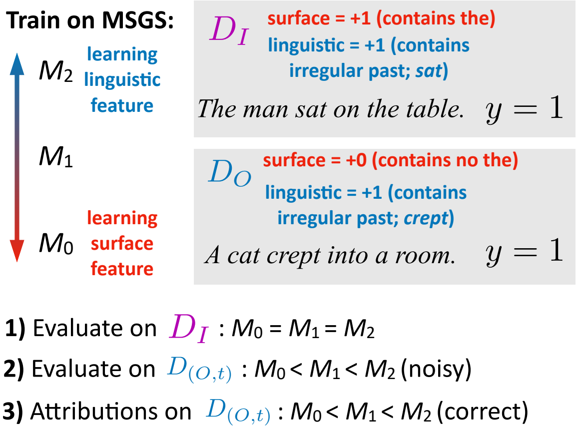


Figure 3: Example from the MSGS train and OOD test sets. The training data conflates a surface and linguistic generalization as described in Warstadt et al. (2020), resulting in models that learn a range of behaviors. Direct evaluation OOD on small data can tell us this, but explanations can also differentiate which of the two patterns is learned and how strongly they are learned.

We consider three linguistic features: MORPH (presence of an irregular past verb like "*drew*"), ADJECT (presence of an adjective), and VERB (if the main verb is an -ing verb), each paired with the surface feature of "*the*" being in the sentence.

We design factors which look at attributions on the tokens corresponding to these linguistic features, including the tokens surrounding these features as well to account for feature dependence on surrounding words. Our factor $f(\mathbf{x}, \phi) = -\sum_{i=(m-2)}^{m+2} \phi(x_i)$, where m is the index of the feature-critical word for that dataset (e.g., "*slept*" for `IRREG`) and $\phi(x_i)$ is the attribution at an in-

Feature	Method	Accuracy		
MORPH	`ACC`	90.9		
	`CONF`	50.9		
	`CONF-GT`	90.1		
		TOKIG	SHAP	LIME
	`IRREG`	89.2	90.6	**92.8**†
VERB	`ACC`	94.5		
	`CONF`	58.0		
	`CONF-GT`	93.3		
		TOKIG	SHAP	LIME
	`VERB`	92.1	94.0	**94.9**
ADJECT	`ACC`	89.9		
	`CONF`	50.5		
	`CONF-GT`	91.3		
		TOKIG	SHAP	LIME
	`ADJ`	87.4	92.1	**93.5**†

Table 1: Few-shot ranking accuracy metric results on $D_{(O,t)}$ for MSGS. `IRREG`, `VERB`, and `ADJ` are detailed in Section 5. † indicates statistically significant improvement over accuracy (paired bootstrap test: $p < 0.05$)

dex. This factor corresponds closely to the heuristic that the dataset was designed for, or alternately, we can see this factor as inversely proportional to what *other* information the model is using (that is, information outside of this window). We name the factors `IRREG`, `VERG`, and `ADJ` for the MORPH, VERB, ADJECT sets respectively.

Note that this approach assumes that a system designer has prior knowledge of the relevant linguistic and surface feature. This is a generous assumption, and for this dataset is almost sufficient to formulate the rule used to construct it, hence why we call this a proof of concept. We will show more realistic conditions in Section 6.

Models To create a suite of models with varying D_O performance, we inoculate following the steps outlined in Section 4.1. We evaluate our factors via accuracy as described in Section 4.3. More details about the inoculation is present in Section A of the appendix.

Results Table 1 shows the results on this dataset. Our `ACC` baseline performs well: when models differ greatly in performance (e.g., one gets 50% and another gets 90% on the D_O), accuracy on the small $D_{(O,t)}$ ranks these correctly even despite the small subset size. The high regularity of the dataset also means that a model's behavior does not vary greatly from example to example, further reducing variance. However, this ranking is nevertheless still not perfect. We see that `CONF` performs very poorly, by contrast, showing that confidence is not helpful for measuring model behavior.

Overall, we see that methods using explanations are able to beat the `ACC` baseline, with the exception of TOKIG. We additionally found trends within the explanation techniques themselves, with LIME reliably performing the best, and TOKIG being the worst. But generally, all techniques can offer relevant information, and in the best case, the attributions can tell us more reliably what a model is learning than evaluation on a small set of $D_{(O,t)}$ data can. In Section 6, we investigate if these results generalize to real-world datasets.

6 Realistic OOD Settings

We now consider two datasets corresponding to realistic OOD settings treated in past work.

First, **HANS** (McCoy et al., 2019) targets spurious heuristics within MNLI (Williams et al., 2018), such as the hypothesis being a subsequence of the premise, with balanced test sets that can be used to detect model reliance on these heuristics. Models following these heuristics always predict *entailed* for the hypotheses, and will perform at random chance accuracy on the dataset. We use MNLI as our in-domain training set in this setting.

Second, **PAWS** (Zhang et al., 2019) is a paraphrase identification task. PAWS-QQP is an OOD dataset for Quora Question Pairs (QQP) (Iyer et al., 2017) that is composed of pairs with swapped content words/phrases (e.g., *I ran from the Grand Canyon to California* to *I ran from California to the Grand Canyon*). A paraphrase model that relies heavily on lexical overlap will not be sensitive to these changes, and will always predict the label of $y = 1$ to indicate paraphrase. We use QQP as our in-domain training set in this setting.

Details regarding models used in this section are presented in Section 4.1. From the test sets of the corresponding datasets, we randomly sample 400 examples from PAWS and 600 from HANS-CON and HANS-SUB each for use in bootstrap sampling, as detailed in Section 4.3. Table 9 provides additional dataset information.

6.1 Factors

General Factors Both HANS and PAWS involve comparing two sequences $\mathbf{a}, \mathbf{b}$ of tokens, unlike MSGS which is classification over a single sequence. We can define our input $\mathbf{x} = a_1, a_2, ...a_n, b_1, b_2, ..., b_m$ as composed of these two sequences $\mathbf{a}$ and $\mathbf{b}$ with respective attributions

ϕ_a, ϕ_b. We evaluate a number of factors that generally target sensitivity to both sequences and their differences, which represent a broad class of potential heuristics.

`MAX-DIFF`: The difference between maximum attribution in $\mathbf{a}$ and $\mathbf{b}$, i.e. $\max(\phi_a) - \max(\phi_b)$.

`SUM-DIFF`: the difference of the sum of attributions, i.e. $\sum_{i=1}^{n} \phi_{a,i} - \sum_{i=1}^{m} \phi_{b,i}$.

`INDEX-DIFF`: The difference of attributions between shared words in $\mathbf{a}$ and $\mathbf{b}$.

`FIRST-TOK`: The attribution at the the first `<SEP>` token.

We explicitly note that this is the exhaustive set of factors we experimented with, not a cherry-picked set, in order to provide a comprehensive view of what does and doesn't work. We crafted these by manually examining attribution patterns on various datasets rather than trying a large number and keeping the best ones.

HANS Factors We look at the "subsequence" heuristic discussed in Section 2 and the constituent heuristic, which assumes that the premise entails all complete subtrees in its parse-tree. For the subsequence OOD set (HANS-SUB) we note that the `INDEX-DIFF` factor, which specifically examines tokens in the shared subsequence, captures the setting's pathological heuristic.

On the constituent OOD set (HANS-CON) we evaluate a factor that examines the attribution on the control words of the premise. For example, for the premise "*Unless the doctors ran, the lawyers encouraged the scientists*" and the hypothesis "*The doctors ran*", we would consider the attributions on the word "*Unless*".

PAWS Factors We further investigate two intuitive heuristics that are based on the construction of the OOD set. `SWAP-AVG` uses the average attribution across all swapped tokens and `SWAP-MAX-DIFF` subtracts the highest magnitude attribution of swapped tokens in the first sentence and the highest magnitude attribution of swapped tokens in the second sentence. For example, for the pair ("*What factors cause a good person to become bad ?*", "*What factors cause a bad person to become good ?*"), `SWAP-AVG` would consider the attributions on "*good*" and "*bad*". `SWAP-MAX-DIFF` is analogous.

6.2 Inoculated Results

We first evaluate models that differ primarily through inoculation, as described in Section 4.1.

	$D_{(O,t)}$		
Ranking Method	PAWS	HANS-SUB	HANS-CON
Baselines			
`ACC`	88.7	90.6	81.6
`CONF`	9.2	40.4	52.8
`CONF-GT`	34.9	20.2	38.9
`RANDOM`	50.7	51.4	49.6
Factors			
`CONST`	–	–	**87.1**
`SWAP-MAX-DIFF`	76.2	–	–
`SWAP-AVG`	**91.4**	–	–
`INDEX-DIFF`	60.5	**91.3**	68.6
`MAX-DIFF`	65.9	69.2	50.6
`SUM-DIFF`	56.6	60.0	75.2
`FIRST-TOK`	74.3	50.4	55.6

Table 2: Few-shot heuristic ranking performance on OOD samples for HANS/MNLI and QQP/PAWS, specifically when comparing **inoculated models** (SHAP explanations). We divide rows by baselines, dataset-specific factors, and general factors.

Results are shown using SHAP in Table 2 which we selected through experiments in this setting as being the best performing. The conclusions here differ somewhat from those on MSGS. We note that the `ACC` baseline remains strong, while `CONF` is near random. We find that *certain* attribution factors are able to outperform the `ACC` baseline, with `SWAP-AVG` the best on PAWS (91.4%), `INDEX-DIFF` the best on HANS-SUB (91.3%), and `CONST` on HANS-CON (87.1%).

This shows that even in settings more realistic than MSGS, **the right choice of factor reveals meaningful information about model generalization**. Moreover, the heuristics that work well are those hand-designed for these datasets, confirming that measuring association with a heuristic via a factor may reveal something about performance.

We qualify these results by noting that in a true few-shot setting, there is some uncertainty regarding whether a chosen factor is truly the best one. As a coarse option, we find `ACC` to be reliable. However, these high-performing factors would still be useful in conjunction with accuracy, or if we had previously validated a factor as ranking models well and we wanted to apply it to rank new models in this domain; the factors will generalize to new models even if they do not generalize to new datasets necessarily.

6.3 Architectural Change Results

We further examine our approach when ranking the performance of different pre-trained models

Ranking Method	$D_{(O,t)}$ HANS/PAWS POOLED
Baseline	
ACCURACY	61.0
GUESS	72.0
Factors	
SET-DEPENDENT	75.5
MAX-DIFF	72.6
INDEX-DIFF	83.0
SUM-DIFF	83.1
FIRST-TOK	69.4

Table 3: Few-shot heuristic ranking performance on OOD samples for HANS/MNLI and QQP/PAWS, specifically when comparing **non-inoculated models** (SHAP explanations), where we take the mean of pairwise accuracies for 3 pairs (for 3 models) on each set.

(RoBERTa, ELECTRA, and DeBERTa).

Table 3 shows that a heuristic `GUESS` based on the expectation across choosing a *best* model and then randomly guessing consistently with that, gives a strong baseline of 72%. Factors also seem to do well in this setting, with all of the general heuristics outperforming the very low `ACC` baseline.

This suggests that in few-shot factors are able to capture distributional information that baselines can't. However, to qualify this, given that each set only compares between 3 pairs of models, it's easier for factors to happen upon strong accuracy patterns by chance.

Thus, in Table 4, we analyze this further by showing resuts on some individual model pairs. (R1, R2 are RoBERTa; E is Electra; D is DeBERTa) R1-E and D-R2, have different architectures, but similar OOD accuracy (see Table 6 in the Appendix). R1-D, E-D and E-R2 are different model types with more distant accuracy. Accuracy values for a single pair on this single dataset therefore only reflect differences across bootstrap samples. What we find in common across these types of pairings is that while some values are close to 50%, including the `ACCURACY` baseline, each column has several factors achieving very distinct (0%, 100%) accuracy values, consistently differentiating these models. As we note in Figure 4 (Appendix), this pattern of strong distinctions is quite common when different types of models are compared. We further discuss this in Section 7.

7 Analysis / Discussion

Accuracy is reliable, but factors can provide more fine-grained distinctions. On MSGS, where factors beat strong accuracy baselines, we notice that these pairwise accuracies are consistently high. For example, on the MORPH setting, for two models with 95% and 98% accuracy, our factors `IRREG` is 100% accurate, while the accuracy baseline here is only 58%, as test accuracy on $D_{(O,t)}$ does not discriminate well between two models with such close overall accuracy.

	Model Pair (M_1-M_2)				
Ranking Method	R1-E	R1-D	E-R2	D-R2	E-D
Baseline					
ACCURACY	77.4	54.6	55.2	67.2	64.4
Factors					
SWAP-MAX-DIFF	57.6	57.6	70.6	65.6	42.6
SWAP-AVG	93.4	93.4	65.0	47.6	17.0
MAX-DIFF	63.2	63.2	39.2	4.8	0.2
INDEX-DIFF	0	0	99.6	99.6	15.0
SUM-DIFF	99.8	100	0	0	88.0
FIRST-TOK	1.2	100	99.6	0	0

Table 4: SHAP pairwise accuracies, different types of models, PAWS. Model R1 (69.7%), R2 (82.9%), Model E (80.5%), and Model D (71.8%)

This holds at the fine-grained pairwise level as well. Figure 5 (also see Figure 6 in appendix) shows the baseline $D_{(O,t)}$ accuracy against a specific factor's accuracy for each model pair in MSGS. Each datapoint in the scatterplot represents a model pair and a point's vertical distance from the red line represents how much better or worse a given factor does compared to the baseline on a specific pair. We see a regular trend: explanations seem to systematically outperform the baseline across various pairs, with a few significant deviations for low-performing pairs.

These results suggest that explanations can be useful and do add information otherwise missing from accuracy probing alone, especially when the underlying model architecture is held constant. With differing architectures (Figure 6), the problem is made more difficult, and selecting the right factor is less obvious; few-shot accuracy may be more reliable in this setting. Note, however, that these successes from any technique are in spite of us only inspecting 10 examples from the target domain.

Factors differentiate models strongly, though not always in a way aligned with OOD performance. Figure 4 and Table 4 both show that factors will often consistently decide in favor of a certain model regardless of the choice of $D_{(O,t)}$, especially when dealing with models with different

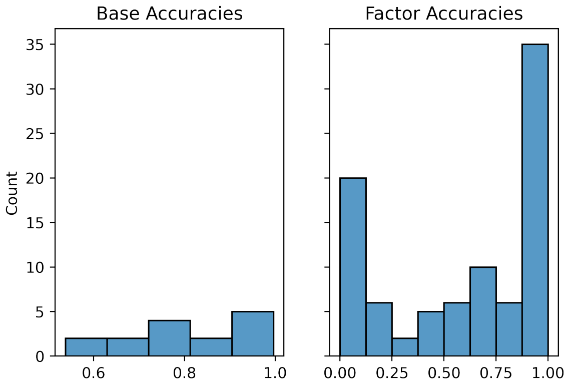


Figure 4: Distributions of pairwise accuracies on PAWS SHAP non-inoculated, all model pairs (left for accuracy baseline, right for all factors).

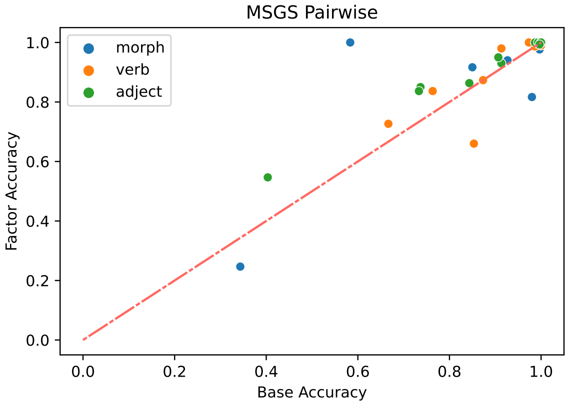


Figure 5: LIME pairwise factor against baseline accuracies for MSGS. Additionally see Figure 6.

base architectures. Since ranking accuracy correlates with whether these strong alignments are consistent across a spectrum of models and choose the models with higher OOD performance, the tendency for factors to strongly favor a specific model doesn't necessarily correlate with strong overall performance, but does heavily imply that these factors extract meaningful information about the model from the attributions. Looking close at Table 4, we can see that that even between different model architectures, certain factors are more (`INDEX-DIFF`) or less (`SWAP-MAX-DIFF`) capable of making these distinctions.

Factors as projections of model feature space. Based on these results, we have evidence that the distributions of attributions are unique to models: in other words, a factor is like a scalar *signature* for a model's feature space with respect to some relevant features. Methods like inoculation, that change a model's behavior in direct ways lead to regular changes in that signature. In these cases, factors align with OOD performance, which explains why factors are so strong in our inoculated experiments. For our non-inoculated experiments (i.g. ELECTRA vs DeBERTa), the feature spaces are fundamentally different, so factor signatures will still capture these differences, but in a way less aligned with ranking on OOD performance. Future work may be able to expand on these differences and what they tell us beyond OOD performance.

8 Related Work

This paper relates to a long line of work on understanding explanations, including explanations' human interpretability (Miller, 2019; Jacovi and Goldberg, 2020; Alqaraawi et al., 2020; Nguyen et al., 2021), explanations' faithfulness and ability to detect shortcuts (Geirhos et al., 2020) or spurious features (Bastings et al., 2021; Madsen et al., 2021; Zhou et al., 2021), and applications to OOD data (Ye and Durrett, 2022; Choi et al., 2022), including papers in the intersection of multiple directions (Adebayo et al., 2022; Kim et al., 2021).

Past work has also investigated performance prediction (Xia et al., 2020; Ye et al., 2021b; Varshney et al., 2022), and using explanations to detect spurious correlations (Kim et al., 2021; Bastings et al., 2021; Adebayo et al., 2022). We are different in that we focus on ranking an array of models which exhibit different levels of generalization abilities, as opposed to giving a binary judgment of whether a model is relying on some shortcuts (Kim et al., 2021; Bastings et al., 2021; Adebayo et al., 2022). In addition, we experiment with tasks having nuanced shortcuts 'in the wild', contrary to synthetically constructed datasets in Bastings et al. (2021). In particular, Adebayo et al. (2022) study the usefulness of explanations in detecting unknown spurious features in an image classification task involving (realistic) possible shortcuts, but find that attributions are ineffective for detecting unknown shortcuts in practice.

9 Conclusion

We establish a robust framework for evaluation of fine-grained few-shot prediction of OOD performance, benchmarking approaches in this setting on a range of models. We find that accuracy is a reliable baseline, but intuitive attribution-based factors derived from explanations can sometimes better predict how models will perform in OOD settings, even when they have similar in-domain performance. We further analyze patterns of our approaches, discovering the potential for factors to represent views of model feature space, leaving further exploration to future work.

10 Limitations

There are a large number of explanation techniques and many domains these have been applied to. We focus here on a set of textual reasoning tasks like entailment where spurious correlations have been frequently identified. However, correlations in other settings like medical imaging (Adebayo et al., 2022) could yield different results. We also note that these datasets are all English-language and use English pre-trained models, so different settings may yield different results; additionally, our factors depend on how explanations are normalized between different examples.

Our paper and analysis themselves comment on the limitations of our methodology as well as explanations as a whole: we find that while explanations often can clearly distinguish different models, knowing which factors will do so, or guaranteeing that explanations align with OOD performance, remains difficult.

Acknowledgments

This work was supported by NSF CAREER Award IIS-2145280, a gift from Salesforce, Inc., and a gift from Adobe. The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources used to conduct this research.

References

Julius Adebayo, Michael Muelly, Harold Abelson, and Been Kim. 2022. Post hoc explanations may be ineffective for detecting unknown spurious correlation. In *International Conference on Learning Representations*.

Ahmed Alqaraawi, Martin Schuessler, Philipp Weiß, Enrico Costanza, and Nadia Berthouze. 2020. Evaluating saliency map explanations for convolutional neural networks: A user study. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, IUI '20, page 275–285, New York, NY, USA. Association for Computing Machinery.

Jasmijn Bastings, Sebastian Ebert, Polina Zablotskaia, Anders Sandholm, and Katja Filippova. 2021. "Will You Find These Shortcuts?" A Protocol for Evaluating the Faithfulness of Input Salience Methods for Text Classification. In *arXiv*.

Jifan Chen and Greg Durrett. 2019. Understanding Dataset Design Choices for Multi-hop Reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4026–4032, Minneapolis, Minnesota. Association for Computational Linguistics.

Jihye Choi, Jayaram Raghuram, Ryan Feng, Jiefeng Chen, Somesh Jha, and Atul Prakash. 2022. Concept-based explanations for out-of-distribution detectors. In *arXiv*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations (ICLR)*.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency Problems: On Finding and Removing Artifacts in Language Data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation Artifacts in Natural Language Inference Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

David Harbecke. 2021. Explaining natural language processing classifiers with occlusion and language modeling. *arXiv preprint arXiv:2101.11889*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. In *arXiv ePrint 2111.09543*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Dieuwke Hupkes, Mario Giulianelli, Verna Dankers, Mikel Artetxe, Yanai Elazar, Tiago Pimentel, Christos Christodoulopoulos, Karim Lasri, Naomi Saphra, Arabella Sinclair, Dennis Ulmer, Florian Schottmann, Khuyagbaatar Batsuren, Kaiser Sun, Koustuv Sinha, Leila Khalatbari, Maria Ryskina, Rita Frieske, Ryan Cotterell, and Zhijing Jin. 2022. State-of-the-art generalisation research in nlp: a taxonomy and review.

Shankar Iyer, Nikhil Dandekar, and Kornel Csernai. 2017. First Quora dataset release: Question pairs.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020. Contextualizing Hate Speech Classifiers with Post-hoc Explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5435–5442, Online. Association for Computational Linguistics.

Joon Sik Kim, Gregory Plumb, and Ameet Talwalkar. 2021. Sanity simulations for saliency methods. *CoRR*, abs/2105.06506.

Frederick Liu and Besim Avci. 2019. Incorporating Priors with Feature Attribution on Text Classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6274–6283, Florence, Italy. Association for Computational Linguistics.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by Fine-Tuning: A Method for Analyzing Challenge Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. In *arXiv*.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS'17, page 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Andreas Madsen, Nicholas Meade, Vaibhav Adlakha, and Siva Reddy. 2021. Evaluating the Faithfulness of Importance Measures in NLP by Recursively Masking Allegedly Important Tokens and Retraining. In *arXiv ePrint 2110.08412*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Giang Nguyen, Daeyoung Kim, and Anh Nguyen. 2021. The effectiveness of feature attribution methods and its correlation with automatic evaluation scores. In *Advances in Neural Information Processing Systems*.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis Only Baselines in Natural Language Inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Danish Pruthi, Rachit Bansal, Bhuwan Dhingra, Livio Baldini Soares, Michael Collins, Zachary C. Lipton, Graham Neubig, and William W. Cohen. 2022. Evaluating Explanations: How Much Do Explanations from the Teacher Aid Students? *Transactions of the Association for Computational Linguistics*, 10:359–375.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*.

Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. 2017. Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2662–2670.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

Liyan Tang, Dhruv Rajan, Suyash Mohan, Abhijeet Pradhan, R. Nick Bryan, and Greg Durrett. 2021. Making Document-Level Information Extraction Right for the Right Reasons. *arXiv*.

Neeraj Varshney, Swaroop Mishra, and Chitta Baral. 2022. Ildae: Instance-level difficulty analysis of evaluation data. *ArXiv*, abs/2203.03073.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods* *in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. 2020. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8625–8646, Online. Association for Computational Linguistics.

Huihan Yao, Ying Chen, Qinyuan Ye, Xisen Jin, and Xiang Ren. 2021. Refining Language Models with Compositional Explanations. In *Advances in Neural Information Processing Systems*, volume 34, pages 8954–8967. Curran Associates, Inc.

Xi Ye and Greg Durrett. 2022. Can Explanations Be Useful for Calibrating Black Box Models? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6199–6212, Dublin, Ireland. Association for Computational Linguistics.

Xi Ye, Rohan Nair, and Greg Durrett. 2021a. Connecting Attributions and QA Model Behavior on Realistic Counterfactuals. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5496–5512, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zihuiwen Ye, Pengfei Liu, Jinlan Fu, and Graham Neubig. 2021b. Towards more fine-grained and reliable NLP performance prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3703–3714, Online. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Yilun Zhou, Serena Booth, Marco Tulio Ribeiro, and Julie Shah. 2021. Do feature attribution methods correctly attribute features? In *eXplainable AI approaches for debugging and diagnosis.*

	VERB	MORPH	ADJECT
NO-INOC	12.0	95.0	51.0
2L	99.0	98.0	99.2
2S	0.0	0.0	0.0
2L 1S	80.0	68.0	73.0
1L 2S	33.0	57.0	32.0
2L 2S	53.7	49.0	56.0

Table 5: MSGS accuracies of various inoculated models.

A Details of Inoculation

One of the methods we used to obtain models with different performances on the OOD sets was inoculation (Liu et al., 2019a), which involves fine-tuning or further fine-tuning models on small amounts or batches of OOD data alongside in-domain data to bring model performance on OOD sets up.

MSGS We borrow notation from Warstadt et al. (2020). Most of the fine-tuning data is ambiguous data that doesn't test the spurious correlation, but we add in small percentages of non-ambiguous data where the label favors either the surface or linguistic generalization, tilting the model in that direction. Here, for each set (VERB, MORPH, ADJECT), we used the following inoculation splits. Linguistic (L) and surface(S) are the features that the inoculation data would favor: 2% L, 2% S, mixed (2% L, 1% S), (1% L 2% S), (2% L, 2% S), in addition to no inoculation. The results on D_O are present in Table 5.

HANS Specific innoculation results for RoBERTa-large are present in Table 6. We additionally use MNLI pre-trained ELECTRA and DeBERTa models from huggingface. These performance details are also located in Table 6.

PAWS We used several inoculation techniques to get a variable number of models here. For our RoBERTA-base model, we start with the base model (35% OOD accuracy) and fine-tune it with D_T data with 2% of the data having D_O data mixed in. We trained this over several epochs to get models with 82.8% and 90.8% accuracy on D_O. We also tried fine-tuning our 35% model on batches of pure D_O data to get a model with 69% accuracy. For our ELECTRA and DeBERTA models, we use similar batch-only inoculation (fine-tuning on batches of only OOD data). More details are present in Table 6.

Tables 7-8 contain the same information as Table 2, but for the other 2 studied explanation technique.

B Bootstrapping Details

We now describe our process for bootstrapping and evaluating the capability of explanations in our setting.

For a sampled population of examples from the D_O set, for the m models that we're examining at a time, we generate explanations for each of the m models on all of the sampled population. We then repeatedly take a sample with replacement (500 times) of 10 examples $D_{O,t}$ each, where we have $500 \times 10 \times m$ total explanations we want to examine. We calculate factors for each of the 10 explanations in each $D_{O,t}$ sample and pool them to get a list of factor metrics for the $D_{O,t}$, one for each model.

For each pair, we then look at the ground-truth D_O ranking for models and their respective factor metrics, getting *successes* where these match, and *failures* otherwise. When we average these accuracies across our 500 bootstrap samples, we get pairwise distributions (the distribution of successes vs failures on a sample for a given pair), which we can further aggregate to get few-shot accuracies.

Note, in practice, to prevent variance from run-to-run, we fix the population of 500 $D_{O,t}$s, but we validated that re-running on new sampled populations didn't impact any numbers greatly. Though we tried using several (5, 10, 20) $D_{O,t}$ sizes, we decided to use the probe size of 10 as a realistic probe size for our setting, which wouldn't be burdensome to hand-craft in practice.

Our methodology can be run quickly in a post-hoc manner as many times as needed on top of a population of the necessary explanations.

C Additional Plots

Figure 4 shows additional information about the distrbution of pairwise accuracies between different model architectures.

D Reproducibility

D.1 Computing Infrastructure

All experiments were conducted on a desktop with 2 NVIDIA 1080 Ti (11 GB) and 1 NVIDIA Titan Xp (12 GB).

D.2 Runtimes

For PAWS and MSGS fine-tuned models, we fine-tuned for roughly 1 GPU hour per model. Since HANS models were trained for very few steps, their training time is inconsequential. Generating attributions required for numerical evaluation took less than 6 GPU hours.

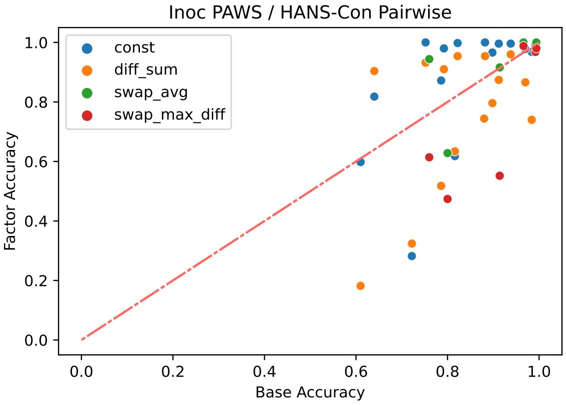


Figure 6: SHAP pairwise factor compared to `ACC` for HANS-CON and PAWS. Each point represents a factor accuracy (y-axis) for a pair of models in comparison to `ACC` (x-axis) for the same pair. Points above the red $y = x$ line represent factors outperforming the accuracy baseline. `CONST` and `DIFF-SUM` are for HANS-CON, `SWAP-AVG` and `SWAP-MAX-DIFF` are for PAWS

D.3 Dataset Details

We used datasets in the JSONL format. We simplified all our dataset settings to binary classification for simplicity, and used data directly from the downloads made available in the original papers.

Dataset	OOD Performance	Huggingface Model Name	LR	Warmup	Steps
HANS	99.8/99.4	`roberta-large-mnli`	$1e^{-5}$	500	150
HANS	96.7/97.6	`roberta-large-mnli`	$1e^{-5}$	500	100
HANS	87.1/70.1	`roberta-large-mnli`	$1e^{-5}$	500	75
HANS	79.5/62.5	`roberta-large-mnli`	$1e^{-5}$	500	50
HANS	69.9/58.7	`roberta-large-mnli`	$1e^{-5}$	500	25
HANS	66.8/57.8	`roberta-large-mnli`	–	–	–
HANS	63.5/72.5	`howey/electra-base-mnli`	–	–	–
HANS	62.7/65.7	`MoritzLaurer/DeBERTa-v3-base-mnli`	–	–	–
MSGS	Table 5	`roberta-base`	$1e^{-5}$	600	6000
PAWS	90.8	`roberta-base`	$1e^{-5}$	1200	12000
PAWS	82.8	`roberta-base`	$1e^{-5}$	1200	12000
PAWS	69.0	`roberta-base`	$1e^{-5}$	1200	7600
PAWS	35.0	`roberta-base`	$1e^{-5}$	1200	12000
PAWS	80.5	`google/electra-base-discriminator`	$1e^{-5}$	1200	7700
PAWS	71.8	`microsoft/deberta-base`	$1e^{-5}$	1200	7600

Table 6: Architecture details for our experiments. "Steps" indicates the number of gradient updates from the specified dataset that are applied to the model. For HANS models, performance is on HANS-SUB/HANS-CON. For all models, small batch sizes were used, with weight decay of 0.1.

	PAWS	HANS-SUB	HANS-CON
Baselines			
ACCURACY	88.7	90.6	81.6
CONFIDENCE	9.2	40.4	52.8
RANDOM	50.7	51.4	49.6
Explanations			
CONST	–	–	79.4
SWAP-MAX-DIFF	80.6	–	–
SWAP-AVG	98.2	–	–
MAX-DIFF	70.1	67.2	58.3
INDEX-DIFF	70.5	88.5	67.2
SUM-DIFF	53.9	59.1	60.4
FIRST-TOK	55.4	51.0	81.3

Table 7: LIME version of Table 3

	PAWS	HANS-SUB	HANS-CON
Baselines			
ACCURACY	88.7	90.6	81.6
CONFIDENCE	9.2	40.4	52.8
RANDOM	50.7	51.4	49.6
Explanations			
CONST	–	–	79.2
SWAP-MAX-DIFF	84.3	–	–
SWAP-AVG	85.6	–	–
MAX-DIFF	86.9	55.2	69.9
INDEX-DIFF	51.4	85.8	53.4
SUM-DIFF	51.6	77.4	68.1
FIRST-TOK	64.0	69.7	59.2

Table 8: Tokig numbers for Table 3

ID Set	OOD Set	D_O Size	$D_{(O,t)}$ Size
MSGS	MORPH	10000	10
	VERB	10000	10
	ADJECT	10000	10
MNLI	HANS-SUB	10000	10
	HANS-CON	10000	10
QQP	PAWS	677	10

Table 9: Information regarding our considered datasets. For all datasets, the bootstrap sample size is fixed at 10.

Mind the Labels: Describing Relations in Knowledge Graphs With Pretrained Models

Zdeněk Kasner,[1] **Ioannis Konstas**[2] and **Ondřej Dušek**[1]
[1]Charles University, Faculty of Mathematics and Physics, Prague, Czechia
[2]The Interaction Lab, MACS, Heriot-Watt University, Edinburgh, UK
`{kasner,odusek}@ufal.mff.cuni.cz, i.konstas@hw.ac.uk`

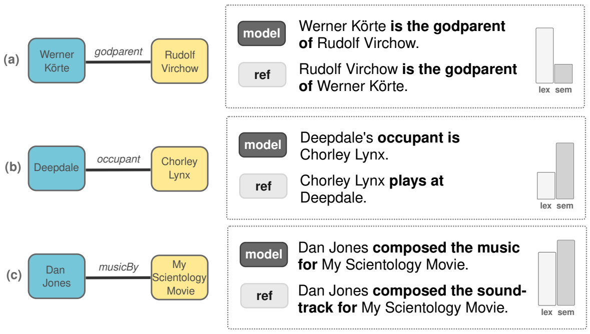
Abstract

Pretrained language models (PLMs) for data-to-text (D2T) generation can use *human-readable data labels* such as column headings, keys, or relation names to generalize to out-of-domain examples. However, the models are well-known in producing semantically inaccurate outputs if these labels are ambiguous or incomplete, which is often the case in D2T datasets. In this paper, we expose this issue on the task of descibing a relation between two entities. For our experiments, we collect a novel dataset for verbalizing a diverse set of 1,522 unique relations from three large-scale knowledge graphs (Wikidata, DBPedia, YAGO). We find that although PLMs for D2T generation expectedly fail on unclear cases, models trained with a large variety of relation labels are surprisingly robust in verbalizing novel, unseen relations. We argue that using data with a diverse set of clear and meaningful labels is key to training D2T generation systems capable of generalizing to novel domains.[1]

1 Introduction

D2T generation systems need to accurately capture the semantics of relations between values in the data. However, the data labels such as relation names (Färber et al., 2018; Haller et al., 2022), table headings (Parikh et al., 2020), or meaning representation keys (Dušek et al., 2020) may provide only superficial or—if the labels are abbreviations, such as in the Rotowire dataset (Wiseman et al., 2017)—no usable hints about the data semantics. Learning how to properly describe the data is thus a challenge for D2T systems, typically requiring in-domain training data of sufficient quality and quantity (Dušek et al., 2019).

PLMs such as BART (Lewis et al., 2020) or T5 (Raffel et al., 2020) can quickly adapt to new domains and exhibit robustness to out-of-domain inputs. However, the PLMs for D2T generation are still limited by the expressivity of the data labels. Consider Figure 1 (a): the model can use its representation of *“godparent”* to understand there is a *“is-a-godparent-of”* relation between the entities, but it has to infer (or guess) who is the godparent of whom. Even in the less ambiguous cases (b) and (c), the model still has to correctly capture the intended semantics of the relation (e.g. *“occupant”* meaning *“home team”*).



model Werner Körte **is the godparent of** Rudolf Virchow.

ref Rudolf Virchow **is the godparent of** Werner Körte.

model Deepdale's **occupant is** Chorley Lynx.

ref Chorley Lynx **plays at** Deepdale.

model Dan Jones **composed the music for** My Scientology Movie.

ref Dan Jones **composed the soundtrack for** My Scientology Movie.

Figure 1: Data-to-text generation models use relation labels (such as *godparent*, *occupant*, and *musicBy*) to describe relations between entities. However, unclear labels can lead to various lexical or semantic incoherencies in the output descriptions, such as swapping the relation direction (a) or using too literal expressions (b).

In this paper, we investigate to what extent PLMs are able to use arbitrary labels describing relations between entities. A suitable testing ground is the task of describing (i.e., *verbalizing*) individual triples in a knowledge graph (KG), which can be considered a trivial case of graph-to-text (G2T) generation (Ribeiro et al., 2021; Koncel-Kedziorski et al., 2019). In this task, there is a wide range of lexical choices for the *relation label* (see Table 1), while the *entities* can be copied verbatim or with only minor morphological changes.

Current human-annotated datasets for D2T generation contain only a small number of relations and rarely contain any unseen relations in the test

---

[1] We release the code and data for our experiments: `https://github.com/kasnerz/rel2text`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2398–2415
May 2-6, 2023 ©2023 Association for Computational Linguistics

relation	possible verbalization
is part of	X is part of Y.
duration	X lasted for Y.
platform	X is available on Y.
	X runs on Y.
country	X was born in Y.
	X is located in Y.
parent	X is the parent of Y.
	Y is the parent of X.
ChEMBL	X has an id Y in the ChEMBL database.

Table 1: Examples of relation labels and their possible verbalizations, with placeholders for head (X) and tail (Y) entities. Relations can be copied verbatim (*is part of*), have a unique verbalization (*duration*), or multiple equivalent lexical choices (*platform*). There is also ambiguity stemming from the semantics of the entities (*country*) or the relation itself (*parent*, *ChEMBL*).

set (Mille et al., 2021). We collect a novel dataset Rel2Text (Re-writing edge labels to Text),[2] acting as a test bench for our experiments. It contains 4,097 single triples from three large-scale KGs (Wikidata, DBPedia, and YAGO) and their crowd-sourced verbalizations, covering 1,522 unique relations (§3). Each relation is equipped with a label, a textual description, and up to five triples in which the relation occurs in the KG.

Using the Rel2Text dataset, we evalute the ability of PLMs to verbalize relations which were not present in the training set. We consider both models finetuned on other relations in our dataset and models finetuned on datasets from a related domain. We also experiment with scenarios involving few-shot finetuning, training on masked labels, and extending the labels with descriptions (§4, 5).

We find that the PLMs are quite robust in verbalizing a diverse set of relations based on their label (achieving ~90% of overall entailment probability). We show that semantically unfaithful model outputs are often caused by incomplete, ambiguous, or noisy input data. Somewhat suprisingly, we also show that longer relation descriptions do not provide substantial improvements over using short labels. However, even for data using short relation labels, the model trained on verbalizing relations can achieve results comparable to verbalizing relations using manual templates in two downstream tasks (§6).

The contributions of our work are as follows:

- We examine the ability of PLMs to describe graph relations, showing that *clear and meaningful labels* are the basis for successful generalization to unseen relations.
- We present Rel2Text—a human-annotated dataset with 4,097 examples verbalizing 1,522 relations from three large-scale open KGs.
- We show that a model trained on Rel2Text can serve as a drop-in replacement for manual templates, preserving or improving performance on downstream tasks.

2 Related Work

Earlier works in natural language generation from KGs exploited domain-specific ontologies for rule-based systems (Cimiano et al., 2013; Bouayad-Agha et al., 2012; Sun and Mellish, 2007, 2006). With the advance of PLMs, structure-aware modeling and task-specific pretraining has lead to remarkable progress on **D2T benchmarks** such as WebNLG (Gardent et al., 2017b; Ferreira et al., 2020), AGENDA (Koncel-Kedziorski et al., 2019), or E2E (Dušek et al., 2020), indicated via both automatic and human evaluation metrics (Ke et al., 2021; Guo et al., 2020; Ribeiro et al., 2020; Harkous et al., 2020).

Agarwal et al. (2021) used a multi-step approach with semantic filtering and distant supervision for verbalizing the English Wikidata, **covering the wide range of relations** present in the KG. The authors use the approach to generate the KeLM corpus – an automatically cleaned corpus with *synthetic* (model-generated) verbalizations of Wikidata triplesets. We use the KeLM corpus to investigate how models trained on large-scale synthetic data differ from models trained on a small-scale human-annotated dataset (cf. §4).

Other works have tried **incorporating descriptions of data labels** in the model inputs. In one of the experiments, Wang et al. (2021) use descriptions of relations from Wikidata instead of their labels for relation embeddings, concluding that it results in worse performance on downstream tasks. Conversely, Kale and Rastogi (2020) and Lee et al. (2021) improve the performance of their systems by including schema descriptions on the input for the dialogue state tracking and dialogue response generation systems.

There has also been a research interest in **verbalizing single triples** as a stand-alone preprocessing step for NLP tasks. The step has been shown to improve the generalization ability of downstream

---

[2] Or simply "Relations-to-Text".

models for data-to-text generation (Laha et al., 2019; Kasner and Dušek, 2020, 2022; Xiang et al., 2022) and response generation in dialogue systems (Kale and Rastogi, 2020). This step can also serve for making the input similar to the format used during pretraining, e.g. for natural language inference (NLI) models (Gupta et al., 2020; Neeraja et al., 2021; Dušek and Kasner, 2020). The above works employ a variety of methods to convert triples to text, ranging from simple templates and rule-based systems to prompting large PLMs. However, none of these works investigate how PLMs behave when presented with novel relations.

In a work concurrent to ours, Keymanesh et al. (2022) investigate the aspects of **generalization performance of PLMs** on the DART dataset[3] (Nan et al., 2021). They compare prompt-based and finetuning-based approaches to D2T generation, focusing on the ability of models to perform on difficult examples. In contrast, we focus on finetuned encoder-decoder models, which were shown in Keymanesh et al. (2022) to be more efficient for D2T generation, and we evaluate the models on clean and manually curated data.

3 Data

For our experiments, we need data with diverse labels and their human verbalizations. In this section, we describe how we gather RDF[4] triples from large-scale KGs (§3.1) and collect their verbalization through crowdsourcing (§3.2, 3.3).

3.1 Input Data

An RDF triple is a tuple $t = (e_h, r, e_t)$, where r denotes the relation between the head entity e_h and the tail entity e_t. We retrieve triples from three open large-scale KGs encoding factual knowledge:

- **Wikidata** (Vrandečić and Krötzsch, 2014) is a large-scale Wikipedia-based KG created using collaborative editing. With approximately 10,000 human-created relations equipped with descriptions,[5] it is by far the largest source of variety in relation labels.
- **YAGO** (Tanon et al., 2020) is a KG which builds upon factual knowledge from Wikidata, but uses a limited set of 116 pre-defined relations from `schema.org` (Guha et al., 2016) mapped to a subset of Wikidata relations.
- **DBPedia** (Lehmann et al., 2015) is a KG that maps Wikipedia infotables to a predefined ontology containing 1,355 relations, about 350 of which are accompanied by a description.

We query all KGs using their openly available endpoints to retrieve a list of relations in each KG. For each relation, we retrieve up to five *triples* that use this relation, and the relation *description*, i.e. a short explanatory text. If present, we also retrieve descriptions for the head and tail entities.

We apply a set of filtering heuristics, leaving out e.g. relations describing KG metadata or identification numbers.[6] In this way, we collect 7,334 triples with 1,716 relations in total. For the full description regarding the data retrieval, please refer to Appendix A.

3.2 Annotation Process

We collect human-written verbalizations for all input triples using Prolific.[7] We built a web interface in which the human annotators are shown a single triple t and asked to describe it in a single sentence. The annotators are encouraged to re-use the entities in their original form, but they are able to change the form if necessary. The annotators can also report noisy inputs. We employed 420 annotators in total, each of which annotated 20 examples. We set the average reward per hour according to the platform recommendations to £7.29 per hour and we accepted all the inputs which passed our built-in checks. See Appendix B for more details on the annotation process.

3.3 Postprocessing the Data

A considerable portion of the collected verbalizations contain typos and grammatical errors, misunderstood meaning of the relation, or extra information in the input. To ensure high quality of our data, we manually examined all crowdsourced examples and annotated them as *OK*, *noisy*, *corrupted* or *containing extra information*. Appendix C includes postprocessing details. In the rest of the paper, we only use the subset of our dataset

---

[3] We did not use DART (which is a compilation of several datasets including WebNLG) for our experiments since it contains many noisy relations.

[4] `https://www.w3.org/TR/PR-rdf-syntax/`

[5] `https://www.wikidata.org/wiki/Wikidata:Database_reports/List_of_properties/all`

[6] Relations describing various IDs make up a suprisingly large portion of relations in Wikidata. Since we focus on diversity instead of coverage, we decided not to include these relations in our dataset.

[7] `https://www.prolific.co/`

with *OK* annotations, one per input triple (4,097 examples, 1,522 distinct relations), although we also make the remaining noisy instances available for future research.

4 Analysis and Evaluation

In our analysis, we are interested in the following research questions:

- **RQ1:** Are the PLMs finetuned for D2T generation able to describe relations *not present in the finetuning corpus*?
- **RQ2:** How many *training examples* do the PLMs need to generate satisfactory outputs?
- **RQ3:** How do the PLMs behave when provided *limited lexical cues* about the relation?
- **RQ4:** Can relation *descriptions* help to clarify ambiguous cases and improve semantic accuracy of the outputs?

To answer these questions, we divide our REL2TEXT dataset into a training and test splits (see §4.1 for details). We then use the **REL2TEXT test set** to evaluate a finetuned BART model (Lewis et al., 2020), a pretrained encoder-decoder transformer, which is used as a backbone of many recent data-to-text models (Ke et al., 2021; Xing and Wan, 2021; Ribeiro et al., 2021; Liu et al., 2021).[8]

To answer *RQ1*, we compare the performance of BART finetuned on the REL2TEXT training set with BART finetuned on two qualitatively different D2T datasets – WEBNLG and KELM. Using REL2TEXT only, we then prepare various setups for answering *RQ2*, *RQ3*, and *RQ4* (details in §4.2). We analyze the outputs of the models both automatically (§4.3) and manually (§4.4).

4.1 Experimental Setup

Datasets We experiment with the following datasets, all of which focus on verbalizing factual information from KGs and use the same triple-based input data format:

- REL2TEXT. Our dataset (cf. §3.2) with single triples from three KGs with 4,097 examples, 1,522 relations and *human-annotated* outputs.
- WEBNLG (Ferreira et al., 2020; Gardent et al., 2017b). A DBPedia-based triple-to-text dataset with 38k examples, 411 relations, up to 7 triples per example, and *human-annotated* outputs. We use the English part of version 3.0 from HuggingFace.[9]
- KELM (Agarwal et al., 2021). A Wikidata-based dataset with 11M examples, 1,519 relations, up to 13 triples per example, and *model-generated* outputs. We use the dataset released by the authors, splitting it in a 1:100 ratio into validation and training data.

Rel2Text Data Split We use approximately 15% of the REL2TEXT examples for the **test set**. To ensure maximum fairness and focus on model generalization to unseen relations, we do not include in the REL2TEXT test set any relations which have an exact string match with a relation in KELM, WEBNLG, or the REL2TEXT training set. We also exclude any relations for which the maximum semantic similarity[10] to any KELM/WEBNLG/REL2TEXT training relation exceeds a threshold of 0.9. We set this threshold empirically in order to exclude relations which are almost synonymous, but slightly lexically different. We use 90% of the remaining examples for the training set and 10% for the validation set.

Data Preprocessing We split the camel case in the relation labels. For finetuning the models, we linearize the input triples by marking the triple constituents with special tokens <*head*>, <*rel*> and <*tail*>, which we add to the model vocabulary.

Training and Decoding Setup In a default scenario, we finetune BART-BASE for 10 epochs and select the best checkpoint using validation BLEU score, then use greedy decoding to produce outputs. We repeat each experiment with five random seeds, averaging the results. See Appendix D for details.

4.2 Compared Systems

Copy Baseline We introduce a simple baseline by outputting the triple constituents separated by space: "$e_h\ r\ e_t$".

Full Training Data We use the default setup (§4.1) on full REL2TEXT and WEBNLG training sets. For KELM (which is about 300× larger than WebNLG), we finetune the model for 1 epoch only. We denote the trained models *full-rel2text*, *full-webnlg*, and *full-kelm*, respectively.

---

[8] We believe that our findings also apply to similar models such as T5 (Raffel et al., 2020), which have shown comparable performance on related tasks.

[9] `https://huggingface.co/datasets/web_nlg`

[10] Computed as cosine similarity between embeddings of the labels, which are encoded using `all-distilroberta-v1` from SBERT (Reimers and Gurevych, 2019).

Limited Training Data For the limited training data setup, we prepare few-shot splits from REL2TEXT as subsets containing $N = \{25, 50, 100, 200\}$ relations with a single example per relation. We select examples at random, ensuring that each few-shot split is a subset of the larger splits. We finetune the *fewshot-N* models for 10 epochs without validation, using the last checkpoint.

Limited Lexical Cues In D2T datasets (with certain exceptions, cf. Gardent et al. (2017a)), unclear labels are kept in original form, implicitly assuming that the models will learn the verbalizations from the training data. We investigate how the models behave if we take this issue to the extreme, i.e. if the relation labels are not available at all. We consider three scenarios:

- *mask-test* – We train the model on REL2TEXT in the standard training setup. For testing, we replace the relation labels in REL2TEXT with the <*mask*> token.
- *mask-train* – For training, we replace the relation labels in REL2TEXT with the <*mask*> token. We test the model on REL2TEXT in the standard evaluation setup.
- *mask-all* – We replace the relation labels in REL2TEXT with the <*mask*> token for both training and testing.

Incorporating Descriptions Our dataset contains short textual descriptions of the relations, which may be useful to disambiguate its meaning and provide additional cues to the model. We consider two scenarios:

- *desc-repl* – We replace the relation label with its description.
- *desc-cat* – We concatenate the relation description with the input, separated using the special token <*rel_desc*>.

4.3 Automatic Evaluation

To get a high-level overview of model behavior, we evaluate generated outputs using the GEM-metrics[11] package (Gehrmann et al., 2021), which provides an extensive set of automatic metrics for text generation.

Lexical Similarity We first measure lexical similarity between the model outputs and human references using **BLEU** (Papineni et al., 2002), **METEOR** (Banerjee and Lavie, 2005), and **BLEURT** (Sellam et al., 2020). The first two metrics focus on n-gram overlap; the latter is a trained metric that also captures semantic similarity between the output and the reference. Although these metrics should not be used in isolation (Gehrmann et al., 2022), they give us a better overview of the output quality in combination with other metrics.

Semantic Similarity and Legibility Lexical similarity metrics focus on the surface form, which may not be telling the whole story. For example, if the relation *parent* denotes that e_t *is the parent of* e_h, but the entities are swapped in the generated text, the output will be incorrect, although lexical similarity metrics will be high. To get deeper insights into semantic and lexical properties of the outputs, we use NUBIA (Kane et al., 2020), which is a trained metric combining several features to measure "interchangeability" (equivalence) of two texts. The metric outputs a single score (**NB**) with a value between 0 and 1. We also report its individual underlying features: the semantic similarity score (**SS**) on a 0-5 scale, predicted by RoBERTa (Liu et al., 2019) finetuned on the STS-B benchmark (Cer et al., 2017); the contradiction (**C**), neutral (**N**), and entailment (**E**) probabilities from RoBERTa finetuned on the MNLI challenge from the GLUE benchmark (Wang et al., 2018); and the perplexity score (**PPL**) from vanilla GPT-2 (Radford et al., 2019), computed as a geometric mean of probabilities of the tokens in each step (this score is referenceless).

Lexical Diversity To assess lexical diversity of the generated texts, we use several metrics used in previous work (Dušek et al., 2020; van Miltenburg et al., 2018). We measure the number of unique n-grams (**U-1**), conditional entropy of bi-grams (**CE-2**), and the mean segmental type-token ratio over segment lengths of 100 (**MSTTR**; Johnson, 1944). We also measure the average output length in tokens (**len**).

4.4 Manual Error Analysis

To examine the sources of errors, we perform an in-house annotation of the model outputs. We identify four model error types based on preliminary observations: semantic errors (SEM), with a swap of the relation direction (DIR) as a special case, too

---

[11] `https://github.com/GEM-benchmark/GEM-metrics`

	Lexical			Semantics					Referenceless				
	BLEU	**METEOR**	**BLEURT**	**SS**	**C**	**N**	**E**	**NB**	**U-1**	**CE-2**	**MSTTR**	**PPL**	**len**
human	-	-	-	-	-	-	-	-	1785	2.13	0.62	5.88	9.55
copy	29.04	37.52	0.09	4.79	1.22	7.57	91.21	0.74	1606	1.17	0.7	7.55	6.72
full-rel2text	52.54	44.86	0.54	4.72	3.50	4.65	91.85	0.88	1661	1.96	0.58	5.89	9.16
full-webnlg	41.99	41.59	0.41	4.65	3.68	6.93	89.39	0.86	1651	2.54	0.56	5.65	10.29
full-kelm	46.74	42.94	0.46	4.70	3.95	5.29	90.77	0.86	1652	2.32	0.56	5.83	9.71
fewshot-25	31.13	35.52	-0.02	3.94	8.35	27.26	64.39	0.65	1445	2.93	0.52	5.34	10.67
fewshot-50	40.60	40.05	0.25	4.44	8.04	13.12	78.84	0.76	1536	2.31	0.55	5.79	9.90
fewshot-100	45.88	42.38	0.38	4.53	6.34	10.60	83.06	0.81	1600	2.13	0.57	5.85	9.57
fewshot-200	48.67	43.34	0.44	4.58	5.40	9.03	85.57	0.83	1626	2.04	0.58	5.89	9.36
mask-test	42.45	38.52	0.25	3.99	14.91	18.47	66.62	0.65	1669	1.96	0.61	5.69	8.96
mask-train	46.90	43.15	0.43	4.55	5.85	11.55	82.61	0.81	1646	2.00	0.57	5.91	9.74
mask-all	42.53	38.49	0.24	3.85	17.58	25.15	57.26	0.61	1677	1.96	0.61	5.66	9.16
desc-repl	49.35	42.85	0.47	4.57	5.78	8.80	85.42	0.82	1693	1.94	0.59	5.86	9.18
desc-cat	53.07	45.04	0.55	4.72	3.46	4.66	91.88	0.87	1668	1.91	0.59	5.92	9.11

Table 2: The summary of evaluation using automatic metrics on REL2TEXT test set. See §4.1 for the descriptions of the models, §4.3 for the descriptions of the metrics, and Table 6 for standard deviations.

	Label	**Example input**	**Model**	**Example outputs (✗ incorrect, ✓ correct)**
model	SEM	(Yousra Matine, *sport country*, Morocco)	*mask-mask* *full-rel2text*	✗ Yousra Matine was born in Morocco. ✓ Yousra Matine plays for Morocco.
	DIR	(Kentucky Channel, *former broadcast network*, KET ED)	*fewshot-100* *full-rel2text*	✗ KET ED was broadcast on Kentucky Channel ED. ✓ The Kentucky Channel was broadcast on KET ED.
	LIT	(Vietnam Television, *first air date*, 1970-09-07)	*full-kelm* *full-rel2text*	✗ The first air date of Vietnam Television was 1970-09-07. ✓ Vietnam Television first aired on 1970-09-07.
	LEX	(RPG-43, *used in war*, The Troubles)	*full-rel2text* *full-kelm*	✗ RPG-43 was used in the The Troubles. ✓ The RPG-43 was used in the Troubles.
data	ENT	(The Age of Entitlement, *by artist*, The Basics)	*full-kelm* *full-rel2text*	✗ The Age of Entitlement was written by The Basics. ✓ The Age of Entitlement was recorded by The Basics.
	LBL	(General Motors Epsilon platform, *vehicle*, Cadillac XTS)	*full-webnlg* *desc-cat*	✗ General Motors Epsilon is a vehicle similar to the Cadillac XTS. ✓ General Motors Epsilon platform is used in the Cadillac XTS.

Table 3: Error categories used in manual analysis, with examples of errors found and corresponding correct verbalizations. Model error types (top): SEM – The output is semantically incorrect, DIR – The direction of the relation is swapped, LIT – The verbalization is too literal, LEX – There is a lexical error in the output. Input data error types (bottom): ENT – The verbalization may depend on the entities, LBL – The relation label is not clear.

literal (LIT), i.e. containing awkward or misleading phrasing, and grammar/lexical errors (LEX). We further annotate two types of input data errors: ambiguous relations (ENT) and relations with unclear labels (LBL). Examples are shown in Table 3. We select 100 random examples together with their corresponding outputs from the *full-rel2text*, *full-webnlg*, *full-kelm*, *fewshot-100*, *mask-all* and *desc-cat* models. Without revealing the output sources, we ask three expert annotators to mark all error categories that apply.

5 Results

5.1 Automatic Evaluation Results

Table 2 shows automatic scores for all our models. *full-rel2text* is the best among the fully trained models in terms of lexical overlap metrics (which is expected, as it is trained on the most similar reference distribution), but the *full-webnlg* and *full-kelm* models are almost equal in terms of semantic consistency, achieving around 90% average entailment probability, which is on par with the copy baseline.

Semantic consistency is much lower for the few-shot models (e.g. the average entailment probability is between 65% and 85%), showing that there is a certain minimum amount of data needed to achieve consistent outputs. Using more examples for training the model generally helps to decrease variance and increase performance across various metrics (cf. Figure 2).

Interestingly, the models which do not see the relations during test time (*mask-test* and *mask-all*) still achieve around 60% average entailment probability, similarly to the worst few-shot model. Although their rate of contradictions is higher than for other models, the results suggest that in many cases, the guessed relation is compatible with the correct relation.

Another interesting observation is that the *mask-train* model (trained not to use the labels) is able to use the labels provided at test time to improve the outputs considerably (contradiction rate drops from 17% to 5% compared to *mask-all*). The fact that the short labels are both sufficient and necessary for the successful verbalization is emphasized by the fact that the *desc-repl* model is worse than *full-rel2text* (although the descriptions are longer and supposedly explain the relation semantics), and the benefits of concatenating the descriptions alongside the relation labels (*desc-cat*) are negligible, only slightly improving lexical similarity metrics (0.5 BLEU point gain over *full-rel2text*).

In terms of lexical diversity, human references use more unique n-grams, but the model outputs are very similar in other aspects. It remains to be seen if the model outputs can stay semantically consistent with diversity-focused decoding techniques such as nucleus sampling (Holtzman et al., 2020).

5.2 Error Analysis Results

Results are summarized in Figure 3; complete results are presented in Appendix F. Examples of model outputs for each error type are shown in Table 3; more examples are given in Appendix G.

The *full-kelm* and *full-webnlg* models use expressions that are too literal (LIT) in 23 and 29 cases, respectively, while the *full-rel2text* and *desc-cat* models do the same only in 11 cases (5 out of which are marked as LBL, i.e., with an unclear label). This suggests that the variability of our dataset helps models to apply more natural expressions, especially if the relation is understandable from its label.

There is a near-constant portion of examples where the models make a semantic error (SEM) *and* the input is marked as needing an extra description (LBL). The models also make relatively many semantic errors on their own, most prominently in the case of the *fewshot-100* and the *mask-all* models. The *mask-all* model made a semantic error in 78 cases, suggesting that guessing the exact relation just from the entities is difficult (although still possible in 22 cases). Morevover, the outcomes from this model are fluent (only 4 LEX errors), making it hard to detect faulty cases.

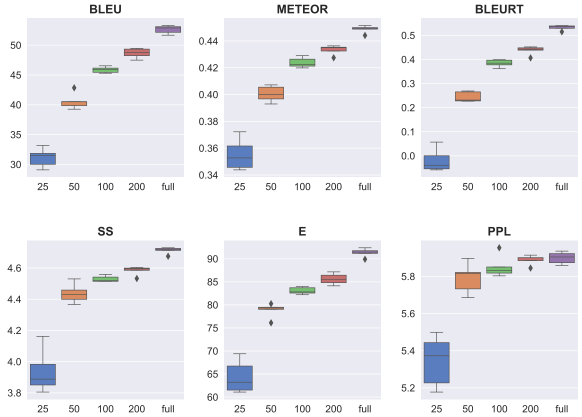


Figure 2: Boxplots for selected metrics from Table 2 w.r.t. the number of examples (displayed on the x-axis, *full* $= 1522$), taking into account variance from individual random seeds (cf. Table 6).

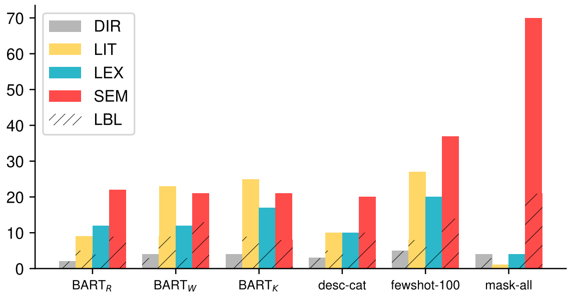


Figure 3: Number of annotated errors per model (see §4.4 and Table 3 for the description of error categories and §4.2 for the models). The striped part signifies that the label of the input was marked as unclear. See Appendix F for details.

The case of swapping the relation direction (DIR) is surprisingly not that common. This is probably down to having only a few examples in our dataset prone to this kind of error. Notably, the results for *full-rel2text* and *desc-cat* are very similar, rendering the impact of extra descriptions negligible.

Finally, there were only 12 out of 100 examples annotated as ENT, which suggests that the verbalization of the relation can be mostly decided irrespective of the entities in the triple.

6 Downstream Tasks

Given that the *full-rel2text* model can describe relations from their labels with high accuracy, we investigate if we can use the model to replace manually created templates in downstream tasks. We select two qualitatively different tasks, both using the idea of transforming individual input triples to

premise repr.	dev	α_1	α_2	α_3
OPR (Gupta et al., 2020)	76.78	75.30	68.46	64.63
BPR (Neeraja et al., 2021)	77.04	74.44	67.46	63.17
full-rel2text (ours)	74.44	74.31	64.59	63.46

Table 4: Accuracy for the dev set and test sets $\alpha_{1,2,3}$ from the INFOTABS dataset. The results are averaged over 3 random seeds.

simple sentences as a preprocessing step: tabular reasoning (§6.1) and zero-shot data-to-text generation (§6.2).

6.1 Tabular Reasoning

Gupta et al. (2020) presented the INFOTABS dataset as an NLI benchmark on tabular data. Each example is a structured table with a set of premises, i.e. natural language claims about the table; the task is to determine whether each premise is entailed by the table, contradicted by it, or neither.

They represent the table as *a paragraph* where each table cell is represented as a short sentence, mostly using a simple template "The *key* of *title* are *value*." Neeraja et al. (2021) extend Gupta et al.'s approach, including a *better paragraph representation* for which they prepare a fine-grained set of rules for individual entity categories. The rules[12] aim to minimize the number of ungrammatical sentences and improve the reasoning abilities of the NLI model.

We replicate the setup of Neeraja et al. (2021) for the original (OPR) and better (BPR) paragraph representation using their public codebase. We then replace their templates with our *full-rel2text* model, verbalizing the triple (*title*, *key*, *value*). The results are summarized in Table 4.

Our preliminary manual evaluation suggests that the sentences from our model are indeed more grammatical (even compared to BPR). However, we observe that the performance is comparable across all three test sets. In line with McCoy et al. (2019), we conclude that for classification tasks such as NLI, the input content appears to be more important than the input form.

6.2 Zero-shot Data-to-Text Generation

Kasner and Dušek (2022) proposed a setup for zero-shot D2T generation in which pretrained models are used to gradually transform text into the final description. The first step of the pipeline requires transforming individual triples into text. We focus on the WebNLG dataset, for which the authors manually created 354 templates.[13] We replicate the authors' setup using their public code, applying *full-rel2text* instead of the templates. The results are summarized in Table 5.

dataset	model	BLEU	METEOR	O	H
filtered	orig	43.19	39.13	0.152	0.073
	full-rel2text	45.39	38.97	0.056	0.161
full	orig	42.92	39.07	0.051	0.148
	full-rel2text	44.63	38.93	0.058	0.166

Table 5: Lexical similarity metrics (BLEU, METEOR) and ommission (O) and hallucinaton (H) rate; following the setup in Kasner and Dušek (2022).

We note that the pipeline using our model for preprocessing is able to achieve improvements of ~2 BLEU points, at the cost of a slightly higher omission and hallucination rate, but crucially without needing the manual effort to create templates. Cursory examination shows that sentences produced by our model are qualitatively similar to the manual templates, but more varied. Unlike the templates, our model may verbalize a relation differently depending on the context. Overall, we argue that training a PLM on verbalizing individual relations can potentially replace the manual effort of creating simple templates, which will have a notable impact for scaling similar approaches to larger datasets.

7 Discussion

Based on our experiments, we can conclude that PLMs are indeed able to verbalize novel relations (**RQ1**). However, there is a caveat: if the relation label is ambiguous or when the cues about the relation are limited (**RQ3**), the model will resolve to guessing and the semantic accuracy of the output descriptions may drop. A takeaway for datasets which do not follow standard naming conventions, such as the Rotowire dataset with basketball summaries (Wiseman et al., 2017) which uses abbreviations for column headers (e.g. FG3A stands for *"the number of shots the player attempted beyond the arc"*), is that rephrasing the labels to natural language may increase the robustness of D2T systems applied on these datasets.

We have focused on finetuned PLMs, which

---

[12] Formalized using more than 250 lines of Python code: `https://github.com/utahnlp/knowledge_infotabs/blob/main/scripts/preprocess/bpr.py#L120`

[13] Available at `https://github.com/kasnerz/zeroshot-d2t-pipeline/blob/main/templates/templates-webnlg.json`

in our case require at least several hundreds of examples to produce satisfactory results (**RQ2**). However, recent research suggests prompting large PLMs capable of in-context learning (Brown et al., 2020) may help to bring down the number of examples required close to zero (Li and Liang, 2021; Reynolds and McDonell, 2021; Schucher et al., 2022; Chia et al., 2022; Xiang et al., 2022). In this case, the models do not have a possibility to learn the correct verbalizations from the training data, which will probably make using clear and unambiguous labels even more important: an issue to investigate in future work.

We showed that improving the outputs using longer relation descriptions is not straightforward (**RQ4**). To achieve more notable improvements, it may be necessary to combine a more detailed specification regarding the relation direction, type, acceptable values, etc., together with a model able to reason about this specification. A promising research in this direction could be using chain-of-thought reasoning, so far applied for tasks such as open-domain question answering or solving math word problems (Gao et al., 2022; Wei et al., 2022; Yao et al., 2022; Nye et al., 2021).

The remaining open question is how to handle input data with noisy labels. We suggest that detecting these cases and fixing them prior to generation (for example with knowledge-augmented systems or a human-in-the-loop setup) could help to improve the robustness of D2T systems in real-world scenarios.

8 Conclusion

We analyzed the abilities of PLMs to verbalize unseen relations in KGs using the relation labels. Based on our findings, we believe that having expressive and unambiguous data labels is a good starting point for adapting D2T systems to new domains. For the analysis, we collected the REL2TEXT dataset, which can help to replace the hand-crafted templates on downstream tasks. Future work may investigate how our findings generalize to prompt-based few-shot or zero-shot D2T generation with large PLMs.

Limitations

Our analysis is limited to verbalizing single triples, which is only a stepping stone towards full-fledged G2T generation. To generate data for entire subgraphs, other issues need to be solved first, including compositional generalization and structure-aware modeling. Nevertheless, we believe that this simplified setting allows us to distill insights which are still applicable to G2T generation in general.

The factuality of the REL2TEXT dataset is tightly related to the data in the input KGs, which may contain outdated or incorrect information, and may be influenced by our processing methods (see Appendix A for details). Using the models trained on our dataset should be done with caution, since it can lead in producing harmful, imprecise, or factually incorrect statements.

We focus only on the English part of the KGs and English datasets. In the future, our approach could be extended to multilingual setting using multilingual PLMs and non-English parts of KGs. For more morphologically rich languages, an extra effort would have to be put into correctly inflecting the entities in the generated text.

Ethics Statement

As we are aiming to develop D2T systems which can robustly generate text for multiple domains, we are building upon PLMs which are known to reflect or amplify biases found in their pretraining corpus (Bender et al., 2021). Although the purpose of our study is to minimize these biases, the outputs of our models can still contain statements which are not aligned with the input data and user needs.

We collected our training and evaluation data through the Prolific crowdsourcing platform. We ensured that all the annotators were given an average reward per hour according to the platform recommendations and we put extra attention into informing the participants about the content and purpose of our study. We also manually filtered the output to minimize the amount of noisy references in our dataset. See 3.2 and Appendix B for more details on the annotation process.

Acknowledgements

This research was supported by Charles University projects GAUK 140320 and SVV 260575, an Apple NLU Research Grant for Heriot-Watt University and Charles University, a Carnegie Trust Research Incentive Grant (RIG009861), and by the European Research Council (Grant agreement No. 101039303 NG-NLG). It used resources provided by the LINDAT/CLARIAH-CZ Research Infrastructure (Czech Ministry of Education, Youth and Sports project No. LM2018101).

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge Graph Based Synthetic Corpus Generation for Knowledge-Enhanced Language Model Pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3554–3565. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? In *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021*, pages 610–623. ACM.

Nadjet Bouayad-Agha, Gerard Casamayor, Simon Mille, Marco Rospocher, Horacio Saggion, Luciano Serafini, and Leo Wanner. 2012. From Ontology to NL: Generation of Multilingual User-Oriented Environmental Reports. In *Natural Language Processing and Information Systems - 17th International Conference on Applications of Natural Language to Information Systems, NLDB 2012, Groningen, The Netherlands, June 26-28, 2012. Proceedings*, volume 7337 of *Lecture Notes in Computer Science*, pages 216–221. Springer.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada.

Yew Ken Chia, Lidong Bing, Soujanya Poria, and Luo Si. 2022. RelationPrompt: Leveraging Prompts to Generate Synthetic Data for Zero-Shot Relation Triplet Extraction. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 45–57. Association for Computational Linguistics.

Philipp Cimiano, Janna Lüker, David Nagel, and Christina Unger. 2013. Exploiting Ontology Lexica for Generating Natural Language Texts from RDF Data. In *ENLG 2013 - Proceedings of the 14th European Workshop on Natural Language Generation, August 8-9, 2013, Sofia, Bulgaria*, pages 10–19. The Association for Computer Linguistics.

Ondřej Dušek, David M. Howcroft, and Verena Rieser. 2019. Semantic Noise Matters for Neural Natural Language Generation. In *Proceedings of the 12th International Conference on Natural Language Generation, INLG 2019, Tokyo, Japan, October 29 - November 1, 2019*, pages 421–426. Association for Computational Linguistics.

Ondřej Dušek and Zdenek Kasner. 2020. Evaluating Semantic Accuracy of Data-to-Text Generation with Natural Language Inference. In *Proceedings of the 13th International Conference on Natural Language Generation, INLG 2020, Dublin, Ireland, December 15-18, 2020*, pages 131–137. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of End-to-End Natural Language Generation: The E2E NLG challenge. *Comput. Speech Lang.*, 59:123–156.

WA Falcon et al. 2019. PyTorch Lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3.

Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. 2018. Linked data quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web*, 9(1):77–129.

Thiago Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussallem, and Anastasia Shimorina. 2020. The 2020 Bilingual, Bi-Directional Webnlg+ Shared Task Overview and Evaluation Results (webnlg+ 2020). In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. PAL: Program-aided Language Models. *CoRR*, abs/2211.10435.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating Training Corpora for NLG Micro-Planning. In *55th annual meeting of the Association for Computational Linguistics (ACL)*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017b. The WebNLG Challenge: Generating Text from RDF Data. In *Proceedings of the 10th International Conference on Natural Language Generation, INLG 2017, Santiago de Compostela*, pages 124–133, Spain.

Sebastian Gehrmann, Tosin P. Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna-Adriana Clinciu, Dipanjan Das, Kaustubh D. Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Rubungo Andre Niyongabo, Salomey Osei, Ankur P. Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. 2021. The GEM Benchmark: Natural Language Generation, its Evaluation and Metrics. *CoRR*, abs/2102.01672.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. 2022. Repairing the Cracked Foundation: A Survey of Obstacles in Evaluation Practices for Generated Text. *CoRR*, abs/2202.06935.

Ramanathan V. Guha, Dan Brickley, and Steve Macbeth. 2016. Schema.org: evolution of structured data on the web. *Commun. ACM*, 59(2):44–51.

Qipeng Guo, Zhijing Jin, Xipeng Qiu, Weinan Zhang, David Wipf, and Zheng Zhang. 2020. CycleGT: Unsupervised Graph-to-Text and Text-to-Graph Generation via Cycle Training. *CoRR*, abs/2006.04702.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: Inference on Tables as Semi-structured Data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2309–2324. Association for Computational Linguistics.

Armin Haller, Axel Polleres, Daniil Dobriy, Nicolas Ferranti, and Sergio José Rodríguez Méndez. 2022. An Analysis of Links in Wikidata. In *The Semantic Web - 19th International Conference, ESWC 2022, Hersonissos, Crete, Greece, May 29 - June 2, 2022, Proceedings*, volume 13261 of *Lecture Notes in Computer Science*, pages 21–38. Springer.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have Your Text and Use It Too! End-to-End Neural Data-to-Text Generation with Semantic Fidelity. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 2410–2424. International Committee on Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The Curious Case of Neural Text Degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Wendell Johnson. 1944. Studies in Language Behavior: A Program of Research. *Psychological Monographs*, 56(2):1–15.

Mihir Kale and Abhinav Rastogi. 2020. Template Guided Text Generation for Task-Oriented Dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 6505–6520, Online.

Hassan Kane, Muhammed Yusuf Kocyigit, Ali Abdalla, Pelkins Ajanoh, and Mohamed Coulibali. 2020. NUBIA: NeUral based interchangeability assessor for text generation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 28–37, Online (Dublin, Ireland). Association for Computational Linguistics.

Zdenek Kasner and Ondřej Dušek. 2022. Neural Pipeline for Zero-Shot Data-to-Text Generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3914–3932. Association for Computational Linguistics.

Zdeněk Kasner and Ondrej Dušek. 2020. Data-to-Text Generation with Iterative Text Editing. In *Proceedings of the 13th International Conference on Natural Language Generation, INLG 2020*, pages 60–67, Dublin, Ireland.

Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. 2021. JointGT: Graph-Text Joint Representation Learning for Text Generation from Knowledge Graphs. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2526–2538. Association for Computational Linguistics.

Moniba Keymanesh, Adrian Benton, and Mark Dredze. 2022. What Makes Data-to-Text Generation Hard for Pretrained Language Models? *CoRR*, abs/2205.11505.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2284–2293. Association for Computational Linguistics.

Anirban Laha, Parag Jain, Abhijit Mishra, and Karthik Sankaranarayanan. 2019. Scalable Micro-planned Generation of Discourse from Structured Data. *Comput. Linguistics*, 45(4):737–763.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue State Tracking with a Language Model using Schema-Driven Prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 4937–4949. Association for Computational Linguistics.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pretraining for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S. Yu. 2021. KG-BART: Knowledge Graph-Augmented BART for Generative Commonsense Reasoning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6418–6425. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3428–3448. Association for Computational Linguistics.

Simon Mille, Kaustubh D. Dhole, Saad Mahamood, Laura Perez-Beltrachini, Varun Gangal, Mihir Sanjay Kale, Emiel van Miltenburg, and Sebastian Gehrmann. 2021. Automatic Construction of Evaluation Suites for Natural Language Generation Datasets. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Linyong Nan, Dragomir R. Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. DART: Open-Domain Structured Data Record to Text Generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 432–447. Association for Computational Linguistics.

J. Neeraja, Vivek Gupta, and Vivek Srikumar. 2021. Incorporating External Knowledge to Enhance Tabular Reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2799–2809. Association for Computational Linguistics.

Maxwell I. Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show Your Work: Scratchpads for Intermediate Computation with Language Models. *CoRR*, abs/2112.00114.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Ankur P. Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and

Dipanjan Das. 2020. ToTTo: A Controlled Table-To-Text Generation Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1173–1186. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019*, pages 8024–8035, Vancouver, BC, Canada.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models Are Unsupervised Multitask Learners. Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Laria Reynolds and Kyle McDonell. 2021. Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama Japan, May 8-13, 2021, Extended Abstracts*, pages 314:1–314:7. ACM.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating Pretrained Language Models for Graph-to-Text Generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. 2020. Modeling Global and Local Node Contexts for Text Generation from Knowledge Graphs. *Trans. Assoc. Comput. Linguistics*, 8:589–604.

Nathan Schucher, Siva Reddy, and Harm de Vries. 2022. The Power of Prompt Tuning for Low-Resource Semantic Parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 148–156. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. BLEURT: Learning Robust Metrics for Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7881–7892. Association for Computational Linguistics.

Xiantang Sun and Chris Mellish. 2006. Domain Independent Sentence Generation From RDF Representations for the Semantic Web. In *Combined Workshop on Language-Enabled Educational Technology and Development and Evaluation of Robust Spoken Dialogue Systems, European Conference on AI, Riva del Garda, Italy.*

Xiantang Sun and Chris Mellish. 2007. An Experiment on "free Generation" from Single RDF Triples. In *Proceedings of the Eleventh European Workshop on Natural Language Generation, ENLG 2007, Schloss Dagstuhl, Germany, June 17-20, 2007.*

Thomas Pellissier Tanon, Gerhard Weikum, and Fabian M. Suchanek. 2020. YAGO 4: A Reasonable Knowledge Base. In *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*, volume 12123 of *Lecture Notes in Computer Science*, pages 583–596. Springer.

Emiel van Miltenburg, Desmond Elliott, and Piek Vossen. 2018. Measuring the Diversity of Automatic Image Descriptions. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1730–1741. Association for Computational Linguistics.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 353–355. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Trans. Assoc. Comput. Linguistics*, 9:176–194.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. *CoRR*, abs/2201.11903.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in Data-to-Document Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2253–2263. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jiannan Xiang, Zhengzhong Liu, Yucheng Zhou, Eric P. Xing, and Zhiting Hu. 2022. ASDOT: Any-Shot Data-to-Text Generation with Pretrained Language Models. *CoRR*, abs/2210.04325.

Xinyu Xing and Xiaojun Wan. 2021. Structure-Aware Pre-Training for Table-to-Text Generation. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2273–2278. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. *CoRR*, abs/2210.03629.

A Data Retrieval

DBPedia We query DBPedia through its SPARQL access point: `http://dbpedia.org/sparql`. We retrive relations as objects of type `rdf:Property` which have a property `rdfs:comment` (i.e., the relation description) and language `'en'`.

YAGO We download the English Wikipedia subset of YAGO 4 database dump from `https://yago-knowledge.org/downloads/yago-4`. We retrieve all objects of type `rdf:Property` which have a property `rdfs:comment`. For the entity descriptions, we parse the entity page at YAGO website `http://yago-knowledge.org/resource/`.

Wikidata We first use the Wikidata SPARQL access point: `https://query.wikidata.org/sparql` to retrieve the list of relations as objects of type `wikibase:Property` with `wikibase:language="en"`, together with their English descriptions (`lang(?altLabel) = "en"`). Second, we query Wikidata through the LDF endpoint `https://query.wikidata.org/bigdata/ldf`, which is better able to handle heavy requests, to retrieve the list of triples involved in the relation. Finally, for retrieving the entity descriptions, we use the API at `https://www.wikidata.org/w/api.php`.

Filtering We apply a comprehensive set of filters for filtering out noisy triples, including triples with entities containing meta-information (*"Category:"*, *"XMLSchema#"*), URLs, entites longer than 64 characters, relations having the string *"id"*, *"number"*, or *"code"* in the label, or having *"Reserved for DBpedia"* in the description. As a consequence, we lose some relations, most notably about 2/3 of the relations from Wikidata describing various identifiers (we opted for this step in order to maintain data diversity). If KGs contain relations with identical labels, we prefer the relations from DBPedia and YAGO (which have a substantially lower amount of relations) to Wikidata relations.

Missing Units Our dataset mostly does not contain units for quantities. Although the units are usually present in the KGs, they are not part of the quantity itself – they may be either connected to the quantity with another property, or described informally in the relation label. Since our focus was on the relation labels, we decided to not put additional effort in retrieving and processing the units. In effect, we consider verbalizations not using the units (e.g., (Bommersheim substation, *voltage*, 20000) → "Bommersheim substation has a voltage of 20000.") as correct.

Factual Correctness A certain part of the data is factually incorrect, either because there was an error in the knowledge graph (e.g., (Catalans, *population place*, **Italy**)) or because there was a processing error (e.g., (Child Language Teaching and Therapy, *final publication year*, **-1985**). Since our focus was not on judging the factuality of the inputs (which is a difficult problem on its own right), we decided to keep the examples in the dataset and consider the examples semantically consistent with the input triple as correct.

Other Notes

- All the data was retrieved in February 2022, except for YAGO where we used the newest available dump *2020-02-24*.

- Although we retrieved the entity descriptions wherever possible and we include them in our dataset, we decided not to use them in our experiments.
- The Python code for retriving the data is available in the paper repository.

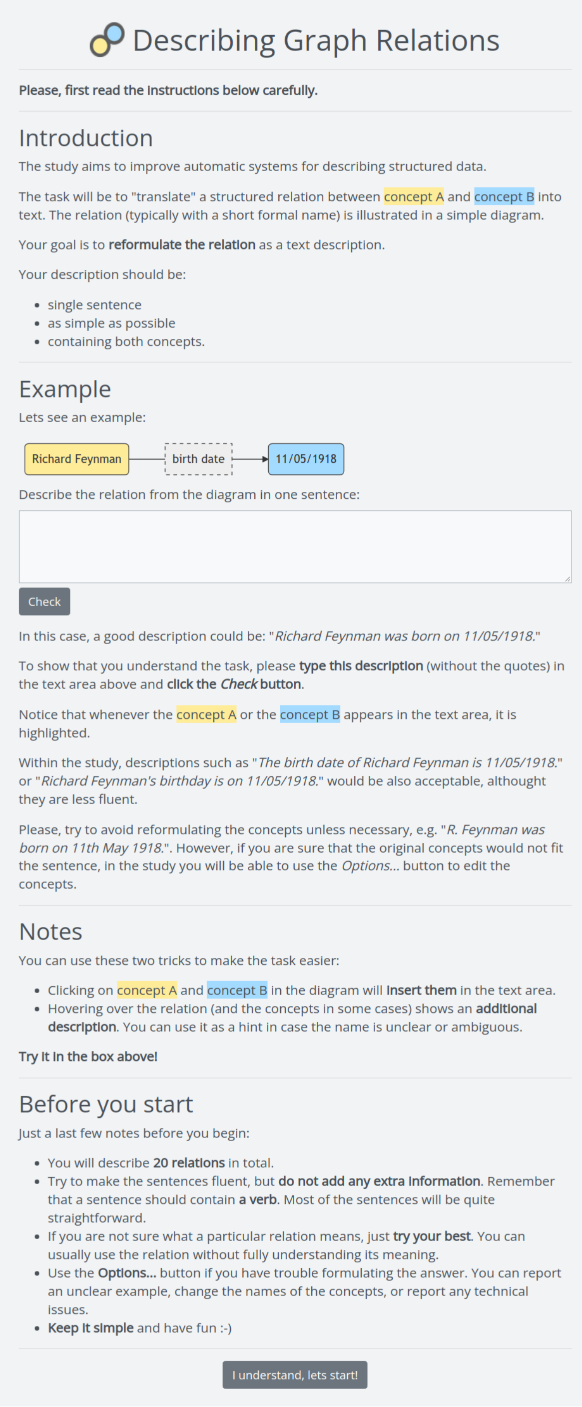
B Crowdsourcing Details

We built a web interface for collecting verbalizations for the triples. Figure 4 shows the introductory instructions displayed for the participants and Figure 5 shows the annotation interface.

We hired annotators on the Prolific crowdsourcing platform `https://app.prolific.co/`. We required that the annotators are native speakers of English. After completing an introductory example, the annotators were given 20 randomly selected triples presented in a sequential order. The annotators were asked to write a short, single-sentence description of the triple. For making the annotation easier, hovering the mouse over the relation revealed its description (this applied also for the entities, if the description was present).

The annotators could also click on the entity to insert it in the text. This motivated the users to insert the entities in the original form. Once the entity appeared in the text (either typed or inserted), it was highlighted. We required that both entities (and at least two extra characters) are present in the text before proceeding to the next step. Because of this requirement, approximately 98.6% sentences in our dataset can be delexicalized using exact string matching. The users also had an option to modify the entity name, which would be recorded as a new ground-truth input (e.g., to make its form more natural). However, this option was used only sparingly.

In total, we collected 8,265 responses for 7,334 examples. Multiple responses for some examples are a consequence of random selection combined with sessions running in parallel. In the final dataset used in our experiments, we selected at most one correct answer for each example (see Appendix C).



Describing Graph Relations

Please, first read the instructions below carefully.

Introduction

The study aims to improve automatic systems for describing structured data.

The task will be to "translate" a structured relation between concept A and concept B into text. The relation (typically with a short formal name) is illustrated in a simple diagram.

Your goal is to **reformulate the relation** as a text description.

Your description should be:

- single sentence
- as simple as possible
- containing both concepts.

Example

Lets see an example:

Describe the relation from the diagram in one sentence:

In this case, a good description could be: "*Richard Feynman was born on 11/05/1918.*"

To show that you understand the task, please **type this description** (without the quotes) in the text area above and **click the *Check* button**.

Notice that whenever the concept A or the concept B appears in the text area, it is highlighted.

Within the study, descriptions such as "*The birth date of Richard Feynman is 11/05/1918.*" or "*Richard Feynman's birthday is on 11/05/1918.*" would be also acceptable, althought they are less fluent.

Please, try to avoid reformulating the concepts unless necessary, e.g. "*R. Feynman was born on 11th May 1918.*". However, if you are sure that the original concepts would not fit the sentence, in the study you will be able to use the *Options...* button to edit the concepts.

Notes

You can use these two tricks to make the task easier:

- Clicking on concept A and concept B in the diagram will **insert them** in the text area.
- Hovering over the relation (and the concepts in some cases) shows an **additional description**. You can use it as a hint in case the name is unclear or ambiguous.

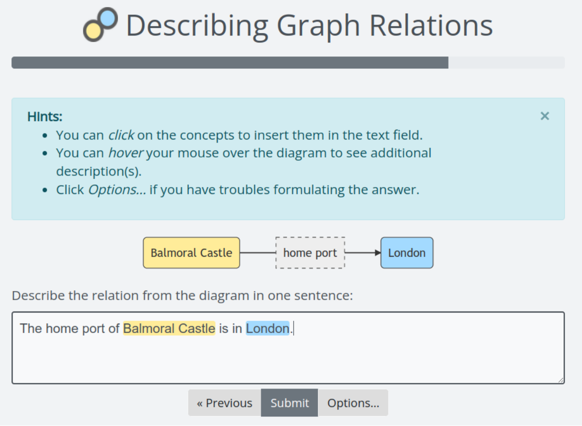
Try it in the box above!

Before you start

Just a last few notes before you begin:

- You will describe **20 relations** in total.
- Try to make the sentences fluent, but **do not add any extra information**. Remember that a sentence should contain **a verb**. Most of the sentences will be quite straightforward.
- If you are not sure what a particular relation means, just **try your best**. You can usually use the relation without fully understanding its meaning.
- Use the **Options...** button if you have trouble formulating the answer. You can report an unclear example, change the names of the concepts, or report any technical issues.
- **Keep it simple** and have fun :-)

Figure 4: The introduction screen shown to the participants.



Describing Graph Relations

Hints:
- You can *click* on the concepts to insert them in the text field.
- You can *hover* your mouse over the diagram to see additional description(s).
- Click *Options...* if you have troubles formulating the answer.

Describe the relation from the diagram in one sentence:

The home port of Balmoral Castle is in London.

Figure 5: The annotation inteface.

C Postprocessing the Dataset

Two of the paper authors manually postprocessed the dataset. We used the following criteria for marking the responses:

- **OK** – The sentence is fluent and semantically consistent with the input.
- **Noisy** – The sentence contains a minor typographical or grammatical error, or the sentence sounds "awkward" (e.g., the relation label is used too literally).
- **Corrupted** – The sentence is semantically incorrect, contains a major typographical or grammatical error, or generally does not make sense.
- **Extra information** – The sentence is correct, but contains extra information about the entities which cannot be derived from the triple itself (e.g., the country of origin of the person found in the entity description).

Figure 6 shows the distribution of responses in our dataset. We marked 4,469 (54.1%) responses as *OK*, 1,314 (15.9%) responses as *Noisy*, 2,246 (27.2%) responses as *Corrupted* and 235 (2.8%) responses as *Extra information*.

Because our priority was to have clean data for evaluation, we decided to use only the *OK* part of our dataset in our experiments. We only use one example for each input triple in our experiments, which gives 4,097 instances. However, since we believe that the human outputs can also be an interesting research target, e.g. for investigating the feasibility of verbalizing the input data, we release all the annotations for future investigations.

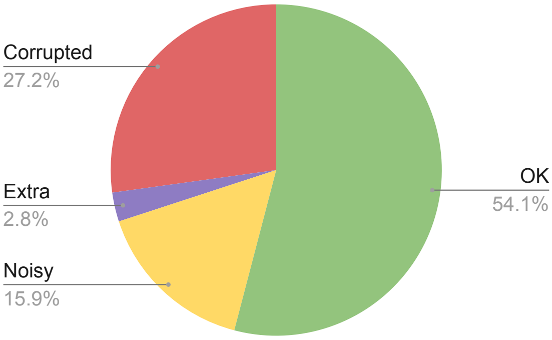


Figure 6: The distribution of crowdsourced responses in our dataset.

Sidenote: Entity Overlap The overlap between entities in test set of Rel2Text and train sets of Rel2Text and WebNLG is around 1%, only 0.5% being named entities (the rest are numerical values). On the contrary, around half of the entities in Rel2Text test set are included in the KeLM train set since KeLM covers a large portion of Wikidata with ca. 6M unique entities. In general, we believe that entity overlap does *not* have a notable influence on the results since all the pretrained models already have a representation of the entities from the pretraining stage.

experiments	Lexical			Semantics					Referenceless				
	BLEU	**METEOR**	**BLEURT**	**SS**	**C**	**N**	**E**	**NB**	**U-1**	**CE-2**	**MSTTR**	**PPL**	**len**
full-rel2text	0.60	0.30	0.01	0.02	0.41	0.38	0.65	0.01	7	0.07	0.01	0.03	0.10
full-webnlg	0.69	0.09	0.00	0.02	0.23	0.94	1.07	0.00	7	0.02	0.00	0.02	0.10
full-kelm	0.78	0.22	0.01	0.02	0.49	0.15	0.42	0.01	11	0.03	0.00	0.03	0.06
fewshot-25	1.60	1.18	0.05	0.14	1.19	2.67	3.58	0.03	68	0.05	0.02	0.14	0.61
fewshot-50	1.36	0.59	0.02	0.06	0.99	0.77	1.59	0.02	19	0.13	0.01	0.08	0.19
fewshot-100	0.51	0.38	0.02	0.02	0.63	0.53	0.75	0.01	14	0.11	0.01	0.06	0.25
fewshot-200	0.80	0.35	0.02	0.02	0.60	1.37	1.25	0.01	14	0.06	0.00	0.02	0.08
mask-test	0.25	0.11	0.01	0.01	0.62	0.48	1.00	0.01	10	0.03	0.00	0.03	0.06
mask-train	0.19	0.09	0.01	0.03	0.64	1.73	1.95	0.01	10	0.02	0.01	0.03	0.09
mask-all	1.19	0.22	0.00	0.04	1.29	0.97	1.62	0.01	8	0.03	0.01	0.05	0.19
desc-repl	0.29	0.13	0.00	0.01	0.71	0.51	0.40	0.01	10	0.05	0.00	0.03	0.14
desc-cat	0.57	0.21	0.00	0.01	0.24	0.28	0.42	0.00	10	0.04	0.01	0.03	0.09

Table 6: Standard deviations for the model experiments in Table 2. Results were computed for 5 random seeds.

D Experimental Setup

Framework We implemented the models in PyTorch Lightning (Paszke et al., 2019). We used the PyTorch (et al., 2019) version of BART-BASE from the Huggingface library (Wolf et al., 2020), with 140M parameters as a basis for all our models.

Hyperparameters We use the Adam (Kingma and Ba, 2015) optimizer ($\beta_1 = 0.9, \beta_2 = 0.98, \varepsilon = 1^{-6}$) with learning rate 2^{-5} and polynomial scheduling with 10% warmup steps. We train the models with batches of size 8 and accumulating gradients with factor 4 (an effective batch size of 32).

Training We train the models for 10 epochs on a single GeForce RTX 3090 GPU with 24 GB RAM, except for *full-kelm* model which we train for 1 epoch. Training times were around 15 minutes for the datasets based on REL2TEXT, 2 hours for *full-webnlg* and 3 days for *full-kelm*. We use greedy decoding in all our experiments.

E Automatic Evaluation

The standard deviations for each experiment from Table 2 are listed in Table 6.

F Manual Evaluation

Table 7 shows full results of our manual evaluation.

In total, there were 7 examples annotated as ENT only, 24 examples annotated as LBL only, and 5 examples annotated as ENT+DESC.

	DIR	LIT	LEX	SEM
full-rel2text	2 (0,2,0)	11 (1,5,1)	12 (0,4,0)	24 (2,9,1)
full-webnlg	8 (2,3,2)	23 (2,9,0)	12 (1,3,0)	25 (4,13,2)
full-kelm	6 (1,3,1)	29 (3,9,2)	19 (1,7,1)	25 (3,8,2)
fewshot-100	3 (0,3,0)	12 (2,5,1)	10 (0,4,0)	24 (2,10,2)
desc-cat	5 (0,5,0)	31 (3,8,2)	22 (2,6,1)	41 (6,14,2)
mask-all	4 (1,3,0)	1 (1,0,0)	4 (1,2,0)	78 (7,21,4)

Table 7: Number of errors manually annotated in 100 outputs from the models. The parentheses contain the portion of examples where the input was marked as ENT only, LBL only, and ENT+DESC, respectively.

G Examples

Examples of outputs of the models selected for manual evaluation are shown in Table 8.

input	(Kalatungan Mountain Range, *highest*, Mount Kalatungan)
desc.	*The highest mountain of a mountain range.*
full-rel2text	Kalatungan Mountain Range’s highest mountain is Mount Kalatunan.
full-webnlg	The Kalatungan Mountain Range is the highest of Mount Kalatuna.
full-kelm	The highest mountain in the Kalatungan Mountain Range is Mount Kalatunagan.
desc-cat	Kalatungan Mountain Range is the highest mountain in the Kalatunan mountain range.
fewshot-100	Kalatungan Mountain Range is the highest mountain in the Kalatunan Mountain Range.
mask-all	Kalatungan Mountain Range is located on Mount Kalatunan.
input	(The Fight Before Christmas, *part of series*, The Simpsons)
desc.	*The series to which this episode or season belongs.*
full-rel2text	The Fight Before Christmas is part of The Simpsons.
full-webnlg	The Simpsons is the series that has The Fight Before Christmas as one of its episodes.
full-kelm	The Fight Before Christmas is part of the Simpsons series.
desc-cat	The Fight Before Christmas is a part of The Simpsons.
fewshot-100	The Fight Before Christmas was part of series of The Simpsons.
mask-all	The Fight Before Christmas was a parody of The Simpsons.
input	(Landwasser Viaduct, *construction material*, Limestone)
desc.	*Construction material (eg. concrete, steel, iron, stone, brick, wood).*
full-rel2text	Landwasser Viaduct was built using Limestone.
full-webnlg	The construction material of the Landwasser Viaduct is Limestone.
full-kelm	The Landwasser Viaduct is made of limestone.
desc-cat	Landwasser Viaduct was built using Limestone.
fewshot-100	Landwasser Viaduct was built using Limestone.
mask-all	Landwasser Viaduct crosses Limestone.
input	(Sharnbrook Academy, *county*, Bedfordshire)
desc.	*The county where the thing is located.*
full-rel2text	Sharnbrook Academy is in Bedfordshire.
full-webnlg	Sharnbrook Academy is in the county of Bedfordshire.
full-kelm	Sharnbrook Academy is located in Bedfordshire.
desc-cat	Sharnbrook Academy is in Bedfordshire.
fewshot-100	Sharnbrook Academy is in Bedfordshire.
mask-all	Sharnbrook Academy is in Bedfordshire.
input	(Loco-Motion, *computing platform*, Tomy Tutor)
desc.	*some sort of hardware architecture or software framework, that allows this software to run*
full-rel2text	Loco-Motion is run on Tomy Tutor.
full-webnlg	Tomy Tutor is the computing platform for Loco-Motion.
full-kelm	Loco-Motion is available for Tomy Tutor.
desc-cat	Loco-Motion runs on Tomy Tutor.
fewshot-100	Loco-Motion is a computing platform for Tomy Tutor.
mask-all	Loco-Motion was inspired by Tomy Tutor.

Table 8: Examples of outputs of the models selected for manual evaluation.

Shapley Head Pruning:
Identifying and Removing Interference in Multilingual Transformers

William Held
`wheld3@gatech.edu`

Diyi Yang
`diyiy@stanford.edu`

Abstract

Multilingual transformer-based models demonstrate remarkable zero and few-shot transfer across languages by learning and reusing language-agnostic features. However, as a fixed-size model acquires more languages, its performance across all languages degrades. Those who attribute this interference phenomenon to limited model capacity address the problem by adding additional parameters, despite evidence that transformer-based models are overparameterized. In this work, we show that it is possible to reduce interference by instead identifying and pruning language-specific attention heads. First, we use Shapley Values, a credit allocation metric from coalitional game theory, to identify attention heads that introduce interference. Then, we show that pruning such heads from a fixed model improves performance for a target language on both sentence classification and structural prediction. Finally, we provide insights on language-agnostic and language-specific attention heads using attention visualization.[1]

1 Introduction

Cross-lingual transfer learning aims to utilize a natural language processing system trained on a source language to improve results for the same task in a different target language. The core goal is to maintain relevant learned patterns from the source while disregarding those which are inapplicable to the target. Multilingual pretraining of transformer language models has recently become a widespread method for cross-lingual transfer; demonstrating remarkable zero and few shot performance across languages when finetuned on monolingual data (Pires et al., 2019; Conneau et al., 2019; Xue et al., 2021).

However, adding languages beyond a threshold begins to harm cross-lingual transfer in a fixed-size model as shown in prior work (Conneau et al., 2019; Xue et al., 2021). This phenomenon, termed interference, has been addressed with additional parameters, both language-specific (Pfeiffer et al., 2020) and broadly (Conneau et al., 2019; Xue et al., 2021). Wang et al. (2020) justifies this by showing that competition over limited capacity drives interference. This seems to contradict the lottery ticket hypothesis, which has shown that pretrained language models are highly overparameterized (Frankle and Carbin, 2019; Chen et al., 2020).

We offer an alternate hypothesis that interference is caused by components that are specialized to language-specific patterns and introduce noise when applied to other languages. To test this hypothesis, we introduce a methodology that selectively removes noisy components to improve language-specific performance without updating or adding additional language-specific parameters. Our work builds on prior research studying monolingual models that shows they can be pruned aggressively (Michel et al., 2019; Voita et al., 2019).

We leverage Shapley Values, the mean marginal contribution of a player to a collaborative reward, to identify attention heads that cause interference. Unlike prior methods, Shapley Values map each head to positive and negative values in a way that abides by all axioms of fair attribution (Ali et al., 2022). Therefore, negative values soundly mark interfering heads where removal will improve performance. We approximate Shapley Values in a computationally tractable but functionally accurate manner using truncation and multi-armed bandit sampling following prior work in computer vision (Ghorbani and Zou, 2020). We contribute the following:

1. **Attention Head Language Affinity:** Even when computed from aligned sentences, Attention Head Shapley Values vary based on the language of input. This highlights that a subset of attention heads has language-specific importance, while others are language-agnostic as shown in Figure 1.

---

[1] We release code to compute Shapley Values on GitHub

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2416–2427

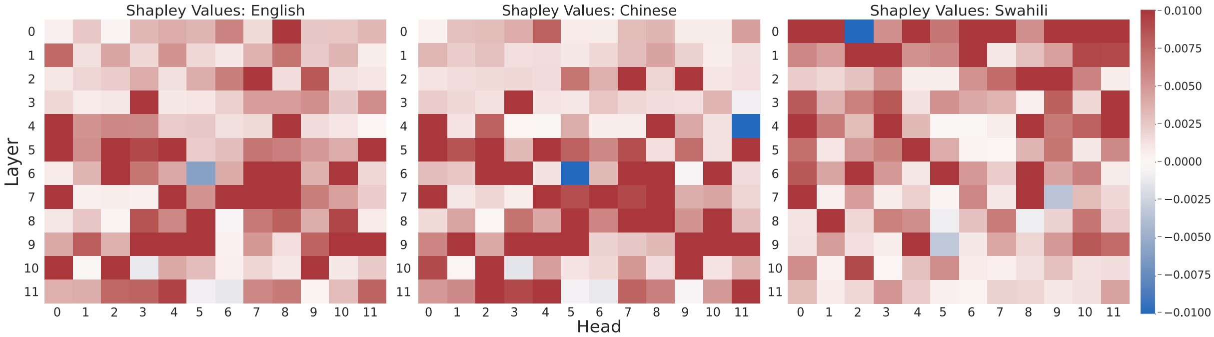


Figure 1: Attention Head Shapley Values for 3 Languages computed from 512 aligned examples for XLM-R finetuned on English XNLI. Each value represents the mean marginal effect an attention head has on accuracy for the test set in that language. The set of harmful heads changes for language, with the most distinct set for Swahili.

2. **Improving through Pruning:** Model pruning according to Shapley Values improves performance without updating parameters on the Cross-Lingual Natural Language Inference corpus (Conneau et al., 2018) and the Universal Dependencies Part-of-Speech corpus (Nivre et al., 2020). This opens a path of work to reduce interference through pruning rather than scaling.

3. **Interpreting Multilingual Heads:** In a qualitative study, we find that the most language-agnostic heads identified have a visible language-agnostic function, while language differences can be measured meaningfully for language-specific heads.

2 Related Work

2.1 Multilingual Learning

A large amount of work has studied both the theoretical underpinnings of learning common structures for language and their applications to cross-lingual transfer. Early works exploited commonality through the use of pivot representations, created either by translation (Mann and Yarowsky, 2001; Tiedemann et al., 2014; Mayhew et al., 2017) or language-agnostic task formulations (Zeman, 2008; McDonald et al., 2011).

As NLP has increasingly used representation learning, dense embedding spaces replaced explicit pivots. This led to methods that identified the commonalities of embedding spaces and ways to align them (Joulin et al., 2018; Artetxe et al., 2018; Artetxe and Schwenk, 2019). Recently, many works (Pires et al., 2019; Conneau et al., 2019; Liu et al., 2020; Xue et al., 2021; Hu et al., 2021) have trained multilingual transformer models as the basis for cross-lingual transfer. These models both implicitly and explicitly align the embedding space across languages, although they empirically achieve stronger alignment between closely related languages (Artetxe et al., 2020; Conneau et al., 2020).

With language-specific data, further work has studied how to reduce interference by adding a small number of language-specific parameters. These works adapt a model for the target language by training only Adapters (Wang et al., 2020; Pfeiffer et al., 2020; Ansell et al., 2021), prompts (Zhao and Schütze, 2021), or subsets of model parameters (Ansell et al., 2022).

Ma et al. (2021) previously investigated pruning in multilingual models using gradient-based importance metrics to study variability across attention heads. However, they used a process of iterative pruning and language-specific finetuning. This iterative process is not consistent since there are many trainable subnetworks within large models (Prasanna et al., 2020). Our method is the first to address interference and improve cross-lingual performance purely by pruning, without updating or adding additional language-specific parameters.

2.2 Model Pruning

Model pruning has largely been focused on reducing the onerous memory and computation requirements of large models. These techniques are broken into two approaches: structured and unstructured pruning. *Unstructured pruning* aims to remove individual parameters, which allows for more fine-grained removal. This process often has minimal effects even at extremely high degrees of sparsity. To efficiently prune a large number of parameters, many techniques propose using gradients or parameter magnitude (Sundararajan et al., 2017; Lee et al., 2019; Frankle and Carbin, 2019; Chen et al., 2020) as importance metrics.

Structured pruning, or removing entire structural components, is motivated by computational benefits from hardware optimizations. In the case of Transformers, most of this pruning work targets removal of attention heads, either through static ranking (Michel et al., 2019) or through iterative training (Voita et al., 2019; Prasanna et al., 2020; Xia et al., 2022). These pruning methods have also been used to study model behavior, but methods with iterative finetuning are not consistent as many sub-networks can deliver the same level of performance once trained (Prasanna et al., 2020).

Our work studies pruning without updating model parameters, which aligns with Michel et al. (2019) which was able to remove up to 40% of total attention heads without impacting accuracy on English Natural Language Inference. However, their gradient-based importance metric does not meet key criteria of efficiency in fair allocation, which states that the sum of the metric across all heads should sum to the model's total performance (Ali et al., 2022). Furthermore, Kovaleva et al. (2019) found that pruning attention heads could sometimes improve model performance without further finetuning. We build on this to develop a methodology for consistently identifying pruned models which improve performance.

3 Methods

To identify and remove interference, we need a metric that can separate harmful, unimportant, and beneficial attention heads. Prior work (Michel et al., 2019; Ma et al., 2021) utilized the magnitude of gradients as an importance metric. However, this metric measures the sensitivity of the loss function to the masking of a particular head. Defined in this way, importance will spike indiscriminately for both harmful and beneficial heads. Therefore, we develop a simple yet effective method to separate these classes.

Shapley Values (Shapley, 1953) have often been applied in model interpretability since they are the only attribution method to abide by the theoretical properties of local accuracy, missingness, and consistency laid out by Lundberg and Lee (2017). In our setting, Shapley Values have two advantages over gradient-based importance metrics. Firstly, gradient-based approaches require differentiable relaxations of evaluation functions and masking, but Shapley Values do not. Therefore, we can instead use the evaluation functions and binary masks directly. Secondly, Shapley Values are meaningfully signed which allows them to distinguish beneficial, unimportant, and harmful heads rather than just important and unimportant heads. This latter property is essential for our goal of identifying interference.

We apply Shapley Values to the task of structural pruning. In order to compute Shapley Values for each head, we first formalize the forward pass of a Transformer as a coalitional game between attention heads. Then, we describe a methodology to efficiently approximate Shapley Values using Monte Carlo simulation combined with truncation and multi-armed bandit search. Finally, we propose a pruning algorithm using the resulting values to evaluate the practical utility of this theoretically grounded importance metric.

3.1 Attention Head Shapley Values

We formalize a Transformer performing a task as a coalitional game. Our set of players A are attention heads of the model. In order to remove self-attention heads from the game without retraining, we follow Michel et al. (2019) which augments multi-headed attention with an added gate $G_h = \{0, 1\}$ for each head Att_h in a layer with N_h heads as follows:

$$\mathrm{MHAtt}(x, q) = \sum_{h=1}^{N_h} G_h \underset{h}{\mathrm{Att}}(x, q) \qquad (1)$$

With $G_h = 0$, that attention head does not contribute to the output of the transformer and is therefore considered removed from the active coalition.

Our characteristic function $V(A)$ is the task evaluation metric $M_v(A)$ over a set of validation data within a target language, adjusted by the evaluation metric with all heads removed to abide by the $V(\emptyset) = 0$ property of coalitional games:

$$V(A) = M_v(A) - M_v(\emptyset) \qquad (2)$$

With these established, the Shapley Value φ_h for an attention head Att_h is the mean performance improvement from switching gate G_h from 0 to 1 across all P permutations of other gates:

$$\varphi_h = \frac{1}{|P|} \sum_{A \in P} V(A \cup h) - V(A) \qquad (3)$$

3.2 Approximating Shapley Values

The exact computation of Shapley Values for N attention heads requires 2^N evaluations of our validation metric, which is intractable for the number of heads used in most architectures. The computation becomes more tractable with Monte Carlo simulation as an approximation (Castro et al., 2009). This replaces the full permutation set P in Equation 3 a randomly sampled subset of permutations.

Computing low-variance Shapley Value estimates with Monte Carlo simulation alone is computationally expensive and provides no clear metric for convergence. Therefore, we follow Ghorbani and Zou (2020) to accelerate our computations. We add a truncation heuristic using priors about the behavior of neural networks and formulate estimation as a multi-armed bandit problem of separating harmful heads from all others. We show in Section 4.3 that this approximation not only reduces the number of samples but explicitly converges to a consistent set of harmful heads across runs, showing consistency even across languages.

Truncation Heuristics Truncation stops sampling the marginal contributions from the rest of a permutation of features once a stopping criterion is reached for that permutation of the Monte Carlo simulation. Prior work selects stopping criterion based on either total performance (Ghorbani and Zou, 2020) or marginal improvements (Ghorbani and Zou, 2019). To avoid tailoring a threshold to each dataset, we instead choose to truncate based on the percentage of remaining attention heads. For all experiments, we truncate when less than 50% of attention heads remain in the coalition. This biases our estimations towards the effect of heads when the majority of the full network is present.

Multi-Armed Bandit Sampling The multi-armed bandit optimization stops sampling the marginal contributions of a particular player once a stopping criterion has been reached according to the variance of that player. Our stopping criterion is based on Empirical Bernstein Bounds (Maurer and Pontil, 2009), a confidence interval based on variance estimation. For t samples with observed variance σ_t and a maximum variance range of R, there is a probability of $1 - \delta$ that the difference between the observed mean $\hat{\mu}$ and true mean μ abides by the following inequality formulated by Mnih et al. (2008):

$$|\hat{\mu} - \mu| \leq \sigma_t \sqrt{\frac{2\log(3/\delta)}{t}} + \frac{3R\log(3/\delta)}{t} \quad (4)$$

We stop sampling for a particular head once this bound is less than $|\mu - 0|$, meaning that we have identified the Shapley Value as positive or negative with probability $1 - \delta$. This saves us significant computation while confidently separating heads into helpful and harmful buckets. For all experiments, we use $R = 1$ since the model's worst-case performance is zero and $\delta = 0.1$ to give a 95% confidence lower and upper bound.

3.3 Importance-Based Structured Pruning

Our pruning procedure works with any signed importance metric. Specifically, we test the utility of the Shapley Values metric for removing interference and helping multilingual models generalize to unseen test data.

Our hypothesis is that attention heads with negative Shapley Values introduce interference. Our pruning method reflects this by using the sign of our approximation directly. We remove all attention heads whose Shapley Value is negative with probability $1 - \delta$ by the Empirical Bernstein inequality from Equation 4. This is a parameter-free approach for deciding the number of heads to preserve. This approach is consistent, with the same set of negative heads identified for pruning across 3 separate runs.

Alternatively, once Shapley Values are computed the model could be pruned to any sparsity level. Unlike prior pruning approaches besides Michel et al. (2019), we do not perform any weight updates following or during pruning and leave all parameters fixed. This provides constant time pruning to the desired size. We evaluate performance in this configurable pruning setting in 4.6.

4 Experiments

4.1 Datasets

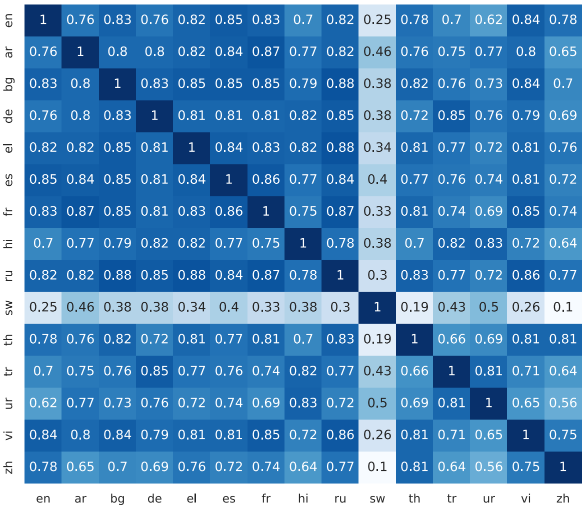
We evaluate our methodology on the Cross Lingual Natural Language Inference (XNLI) and Universal Dependencies Part-Of-Speech (UDPOS) tasks. These allow us to analyze the applicability of Attention Head Shapley Values to both sequence classification and structured prediction. We provide a description of dataset sizes in Table 1.

Cross-Lingual Natural Language Inference (XNLI) We use the Cross Lingual Natural Language Inference (XNLI) Benchmark (Conneau et al., 2018). This dataset is aligned which allows us to control for possible confounding semantic variation in the content. Given a premise and a hypothesis and tasks, XNLI is the task of classifying

Dataset	EN	AR	BG	DE	EL	ES	FR	HI	RU	SW	TH	TR	UR	VI	ZH
XNLI	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0
UDPOS	5.4	1.7	1.1	22.4	2.8	3.1	9.5	2.7	11.3	N/A	N/A	4.8	0.5	0.8	5.5

Table 1: Size of the test sets for the datasets in thousands of sentence pairs and sentences respectively. We use a 512-example subset of the released development sets to compute Shapley Values in all languages for all datasets.

whether the hypothesis is entailed by the premise, contradicted by the premise, or neither.



	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh
en	1	0.76	0.83	0.76	0.82	0.85	0.83	0.7	0.82	0.25	0.78	0.7	0.62	0.84	0.78
ar	0.76	1	0.8	0.8	0.82	0.84	0.87	0.77	0.82	0.46	0.76	0.75	0.77	0.8	0.65
bg	0.83	0.8	1	0.83	0.85	0.85	0.85	0.79	0.88	0.38	0.82	0.76	0.73	0.84	0.7
de	0.76	0.8	0.83	1	0.81	0.81	0.81	0.82	0.85	0.38	0.72	0.85	0.76	0.79	0.69
el	0.82	0.82	0.85	0.81	1	0.84	0.83	0.82	0.88	0.34	0.81	0.77	0.72	0.81	0.76
es	0.85	0.84	0.85	0.81	0.84	1	0.86	0.77	0.84	0.4	0.77	0.76	0.74	0.81	0.72
fr	0.83	0.87	0.85	0.81	0.83	0.86	1	0.75	0.87	0.33	0.81	0.74	0.69	0.85	0.74
hi	0.7	0.77	0.79	0.82	0.82	0.77	0.75	1	0.78	0.38	0.7	0.82	0.83	0.72	0.64
ru	0.82	0.82	0.88	0.85	0.88	0.84	0.87	0.78	1	0.3	0.83	0.77	0.72	0.86	0.77
sw	0.25	0.46	0.38	0.38	0.34	0.4	0.33	0.38	0.3	1	0.19	0.43	0.5	0.26	0.1
th	0.78	0.76	0.82	0.72	0.81	0.77	0.81	0.7	0.83	0.19	1	0.66	0.69	0.81	0.81
tr	0.7	0.75	0.76	0.85	0.77	0.76	0.74	0.82	0.77	0.43	0.66	1	0.81	0.71	0.64
ur	0.62	0.77	0.73	0.76	0.72	0.74	0.69	0.83	0.72	0.5	0.69	0.81	1	0.65	0.56
vi	0.84	0.8	0.84	0.79	0.81	0.81	0.85	0.72	0.86	0.26	0.81	0.71	0.65	1	0.75
zh	0.78	0.65	0.7	0.69	0.76	0.72	0.74	0.64	0.77	0.1	0.81	0.64	0.56	0.75	1

Figure 2: Spearman ρ of Attention Head Shapley Values across languages in XNLI using XLM-R finetuned on the English training split.

Universal Dependencies Part-of-Speech (UDPOS) For structured prediction, we evaluate on the Part-of-Speech (POS) tags from the Universal Dependencies (UD) v2 corpus (Nivre et al., 2020), which has the largest cross-lingual gap in the XTREME benchmark (Hu et al., 2020). The authors suspect that structured prediction requires more language-specific knowledge than many classification tasks.

For direct comparison with our experiments on XNLI, we only retain the 13 languages from UDPOS which have a development and test split, which also exist in XNLI. Unlike XNLI, each language in UDPOS hasa different number of examples which are not aligned across languages.

4.2 Experimental Setup

As the basis for our experiments, we finetune XLM-R Base (Conneau et al., 2019) using the Transformers library (Wolf et al., 2020) on only English data. Evaluation is done using the Datasets library (Lhoest et al., 2021) implementation of the accuracy metrics. Finetuning and Shapley Value computation were both done on a single NVIDIA GeForce 12GB RTX 2080 Ti. We finetune the following hyper-parameter tuning procedures from prior work: using Hu et al. (2020) for XNLI and de Vries et al. (2022) for UDPOS.

For all tasks and languages, we use the accuracy on 512 examples of the development set as the characteristic function for our coalitional game. Our pruning baselines include the gradient-based importance metric of Michel et al. (2019) and the average of 10 randomly pruned networks. We prune the same number of heads pruned by our method for all strategies since our baselines require selecting the number of heads to prune.

4.3 Language Affinity

First, we analyze the Attention Head Shapley values for XNLI. We focus only on the role of the source language by using an aligned sample from XNLI to control our results for differences independent from language variation. In Figure 1, we visualize the results across English, Chinese, and Swahili. As expected from prior work (Michel et al., 2019; Voita et al., 2019), many heads have low magnitude Shapley Values indicating that they play no significant role in the final network. We compare the similarity of Shapley Values learned across languages using Spearman's ρ in Figure 2 and find that Shapley Values are heavily correlated between all languages but Swahili, which is a major outlier. This cross-lingual consistency across languages is juxtaposed with inconsistency of methods that utilize finetuning as shown by Prasanna et al. (2020).

Despite this consistency, we find some attention heads demonstrate high language-specificity. Most notably, the fifth attention head in layer six is positive for Swahili but strongly negative for all other 14 languages. This indicates that this head serves a function specific to Swahili within the model. We investigate the behavior of language-specific and language-agnostic heads further in Section 5.

It is worth noting that the outlier, Swahili, is the language with the fewest number of examples in the

	XNLI Accuracy														
Pruning Strategy	EN	AR	BG	DE	EL	ES	FR	HI	RU	SW	TH	TR	UR	VI	ZH
No Pruning	84.1	70.6	76.7	76.8	75.4	79.8	77.7	70.0	74.7	63.4	70.6	71.9	65.9	73.3	73.5
Random	81.5^{-}	67.2^{-}	72.7^{-}	72.7^{-}	71.3^{-}	75.5^{-}	73.0^{-}	66.3^{-}	70.5^{-}	63.5	67.4^{-}	68.4^{-}	61.6^{-}	69.7^{-}	70.8^{-}
Michel et al. (2019)	84.3	71.0	77.3	77.4	72.8^{-}	80.2	78.4	71.5^{+}	75.2	63.1	70.7	71.7	66.9^{+}	73.3	**77.2**$^{+}$
Shapley Value (φ_i)	**85.1**$^{+}$	**72.0**$^{+}$	**77.8**$^{+}$	**78.3**$^{+}$	**76.3**	**80.6**	**79.7**$^{+}$	**71.5**$^{+}$	**76.5**$^{+}$	**63.8**	**73.3**$^{+}$	**73.2**$^{+}$	**67.6**$^{+}$	**75.3**$^{+}$	**77.2**$^{+}$
Pruned Heads (K)	4	6	6	5	4	5	5	7	5	5	7	5	6	6	9

	UDPOS Accuracy														
Pruning Strategy	EN	AR	BG	DE	EL	ES	FR	HI	RU	SW	TH	TR	UR	VI	ZH
No Pruning	**95.7**	75.1	**90.9**	**88.8**	71.5	**89.8**	81.3	73.9	**88.2**	-	-	**78.7**	67.3	**66.3**	50.2
Random	**95.7**	74.3^{-}	**90.9**	**88.8**	71.8	**89.8**	81.4	73.7	**88.2**	-	-	**78.7**	67.5	**66.3**	55.3^{+}
Michel et al. (2019)	**95.7**	75.1	**90.9**	**88.8**	71.1	**89.8**	81.1	73.8	**88.2**	-	-	**78.7**	67.3	**66.3**	48.9^{-}
Shapley Value (φ_i)	**95.7**	**76.6**$^{+}$	**90.9**	**88.8**	**72.8**$^{+}$	**89.8**	**82.6**$^{+}$	**75.6**$^{+}$	**88.2**	-	-	**78.7**	**69.5**$^{+}$	**66.3**	**62.6**$^{+}$
Pruned Heads (K)	0	4	0	0	4	0	2	2	0	-	-	0	4	0	18

Table 2: Accuracy for UDPOS and XNLI after pruning according to importance metrics. For all metrics, we remove the Bottom-K heads ($K = |\{H_i \mid \varphi_i < 0\}|$) according to that metric. $^{+}$ and $^{-}$ indicate significant ($P < 0.05$) improvement and harm by a pairwise bootstrap test. Model parameters remain fixed for all methods.

data used in the pretraining of XLM-R. Whether the large variation between Swahili and all other languages is induced by linguistic features or the training dynamics of low-resource languages within multilingual models is unclear. We leave this to be explored further in future work.

4.4 Targeted Pruning

To understand the practical applicability of the resulting Shapley Values, we evaluate models before and after pruning all attention heads with negative Shapley Values as described in Section 3.3.

Each resulting language-specific model can be represented with only the 144 mask parameters which indicate whether each attention head is removed or kept. Therefore, this pruning can be seen alternatively as a parameter-efficient learning method, using $1 \cdot 10^{-6}\%$ of the parameters it would require to finetune the model for each language[2].

XNLI In Table 2, we report the accuracy of models after targeted pruning across all languages for both XNLI and UDPOS. For XNLI, we see that targeted pruning improves performance by an average of +1.59 across all 15 languages with the maximum improvement being in Chinese (+3.78) and the minimum improvement in Swahili (+0.37). We might expect that languages closely related to our finetuning language of English would benefit less from pruning, even closely related languages such as French (+1.97) and German (+1.53) are improved significantly.

UDPOS Improvements in UDPOS vary to a higher degree. Only 6 out of 13 languages improve after pruning, with the rest identical with no negative Shapley Values. Surprisingly, this indicates that attention heads do not introduce interference for these languages. We hypothesize that interference for these languages may instead lie largely within the Transformer feed-forward layers, which we do not study in this work. The largest improvement is again in Chinese (+12.4) and the smallest in French (+1.3). In the case of Chinese, this is a 24.7% improvement purely by removing attention heads. Across the languages which were pruned, the average improvement is 3.4 – reducing the cross-lingual gap (Hu et al., 2020) by 0.7.

Comparison to Baselines Randomly pruning is ineffectual or harms performance in both tasks, indicating that pruning alone is not the source of our improvement. Pruning according to the gradient-based metric proposed by Michel et al. (2019) maintains rather than improves performance. This supports our hypothesis that methods that use the magnitude of gradients largely identify non-impactful heads as opposed to harmful heads.

4.5 Zero-Shot Pruning

Given the high rank correlation between many of the languages, we evaluate transferability by using the Shapley Values for English to prune the model for all languages. We report results in Table 3.

XNLI On XNLI, surprisingly, this transferred pruning across languages has similar benefits to our targeted pruning results despite only being learned

---

[2] 144 parameters compared to $1.25 \cdot 10^8$ for full finetuning.

Pruning Strategy	EN	AR	BG	DE	EL	ES	FR	HI	RU	SW	TH	TR	UR	VI	ZH
No Pruning	84.1	70.6	76.7	76.8	75.4	79.8	77.7	70.0	74.7	**63.4**	70.6	71.9	65.9	73.3	73.5
Random	81.7^{-}	67.1^{-}	72.3^{-}	72.9^{-}	71.1^{-}	75.1^{-}	73.5^{-}	65.7^{-}	71^{-}	60.7^{-}	67^{-}	68.3^{-}	61^{-}	69.7^{-}	70.7^{-}
Michel et al. (2019)	84.3	70.3	76.7	77.1	75.9	80.1	77.9	70.1	75.1	62.9	71.6	72.5	66.1	74.7^{+}	74.5^{+}
Shapley Value (φ_i)	**85.1**$^{+}$	**72.0**$^{+}$	**77.8**$^{+}$	**79.4**$^{+}$	**76.3**	**80.6**	**79.7**$^{+}$	**71.5**$^{+}$	**76.5**$^{+}$	63.3	**73.1**$^{+}$	**73.1**$^{+}$	**68.4**$^{+}$	**75.2**$^{+}$	**76.3**$^{+}$

Table 3: Accuracy for XNLI after pruning using importance metrics from English. For all metrics, we remove the Bottom-K heads ($K = |\{H_i \mid \varphi_i < 0\}|$) according to that metric. $^{+}$ and $^{-}$ indicate significant ($P < 0.05$) improvement and harm by a pairwise bootstrap test.

for English. Two languages (Urdu and German) achieve better results in the zero-shot pruning than they did in the targeted pruning, five achieve worse results, and the remaining eight are equivalent.

It is likely that the strength of zero-shot transfer is largely due to the removal of the fifth head of layer six, which is one of the top 2 most negative heads for all languages barring Swahili. Interestingly, the Attention Head Shapley Values for Swahili also have the lowest rank correlation with English of any language.

UDPOS However, UDPOS highlights the major shortcoming of zero-shot pruning: all attention heads receive a positive Shapley Value for English for UDPOS. This means that no zero-shot pruning is performed despite targeted pruning finding benefits for languages shown in Table 2.

4.6 Iterative Pruning of Attention Heads

Finally, we evaluate the effectiveness of Shapley Values as a ranking methodology for the iterative pruning evaluation performed by Michel et al. (2019). Iterative pruning evaluates how well each importance ranking captures the combinatorial effects of removing attention heads at different compute budgets. We compare random pruning, the gradient-based approach from Michel et al. (2019), and Shapley Values computed through plain Monte Carlo simulation and Shapley Values using Truncation and Multi-Armed Bandit optimization (TMAB). We plot results in Figure 3.

Averaged across all levels of sparsity, our method outperforms the Random baseline (+5.8), Monte Carlo Shapley Values (+1.6), and the Gradient baseline (+0.6). At different stages. Depending on the target sparsity of interest however, Shapley Values and Gradient-based pruning have different levels of sparsity. Our method is the only method that identifies strongly harmful heads, with performance improving compared to the unpruned model for the first 6 heads removed. Our method achieves the largest performance gap at 44% of model capacity outperforming the Gradient baseline, Monte Carlo Shapley Values, and the Random Baseline by +12.2, +15.1, and +20.9 respectively. However, the gradient baseline outperforms our method when more than 80% of heads are pruned, although neither method performs well above chance at this sparsity.

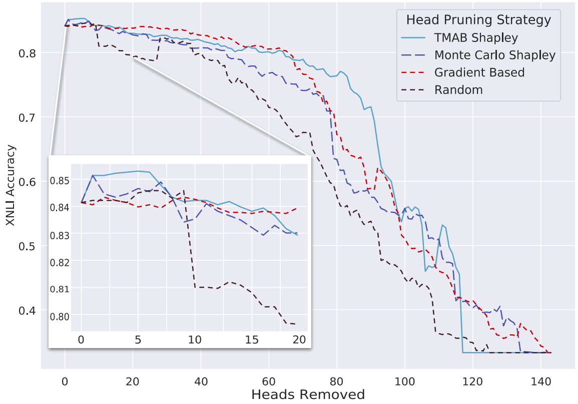


Figure 3: Evolution of XNLI Accuracy as Heads are removed according to different pruning strategies.

5 Qualitative Attention Analysis

In order to provide intuition into the function of attention heads, prior work has turned to attention visualization as the basis for qualitative analysis of the inner workings of transformer models. Clark et al. (2019) and Hoover et al. (2020) both explore patterns within attention heads.

We visualize the attention patterns of outlier attention heads using BertViz (Vig, 2019) from our model to give a qualitative understanding of the attention head patterns associated with language-agnostic and language-specific heads.

5.1 Language-Agnostic Heads

We define the set of language-agnostic heads as the intersection of the the top 20 attention heads for each language. In Figure 4, we visualize the

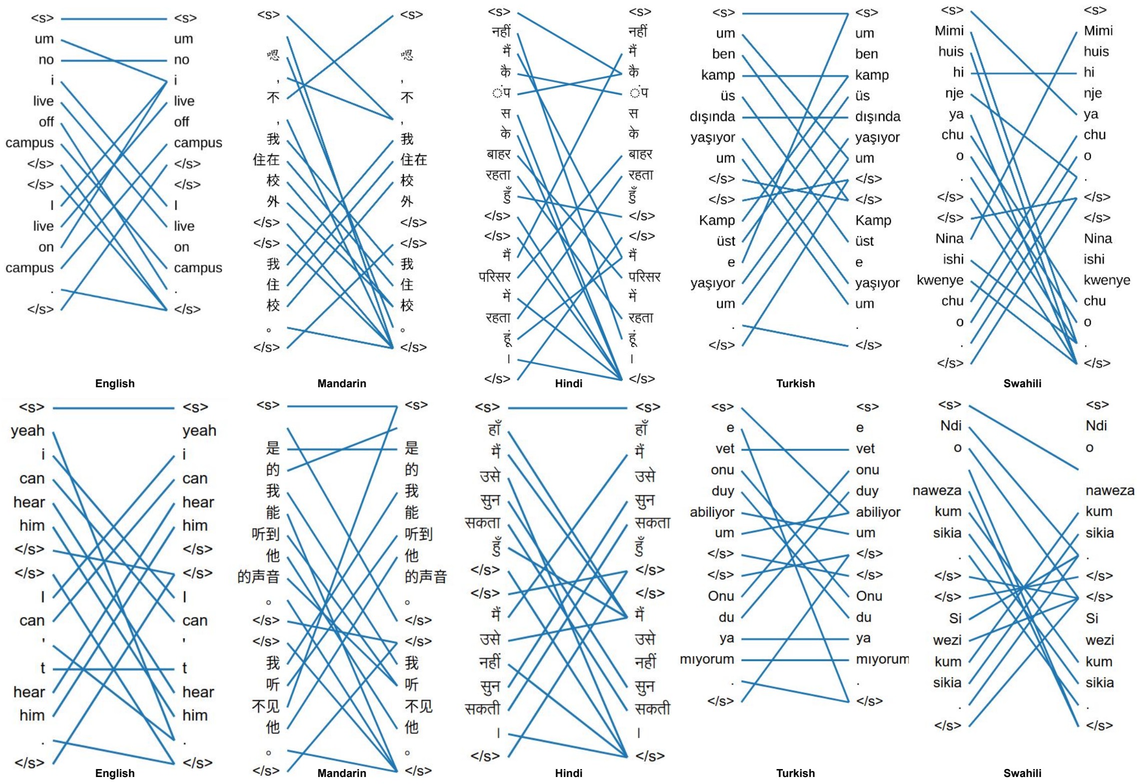


Figure 4: Attention of Layer 2, Head 9 of our XNLI model which is identified as language-agnostic. The attention pattern links synonyms in the premise and hypothesis for all languages. For clarity, we connect the left token to the token on the right which receives the largest attention weight.

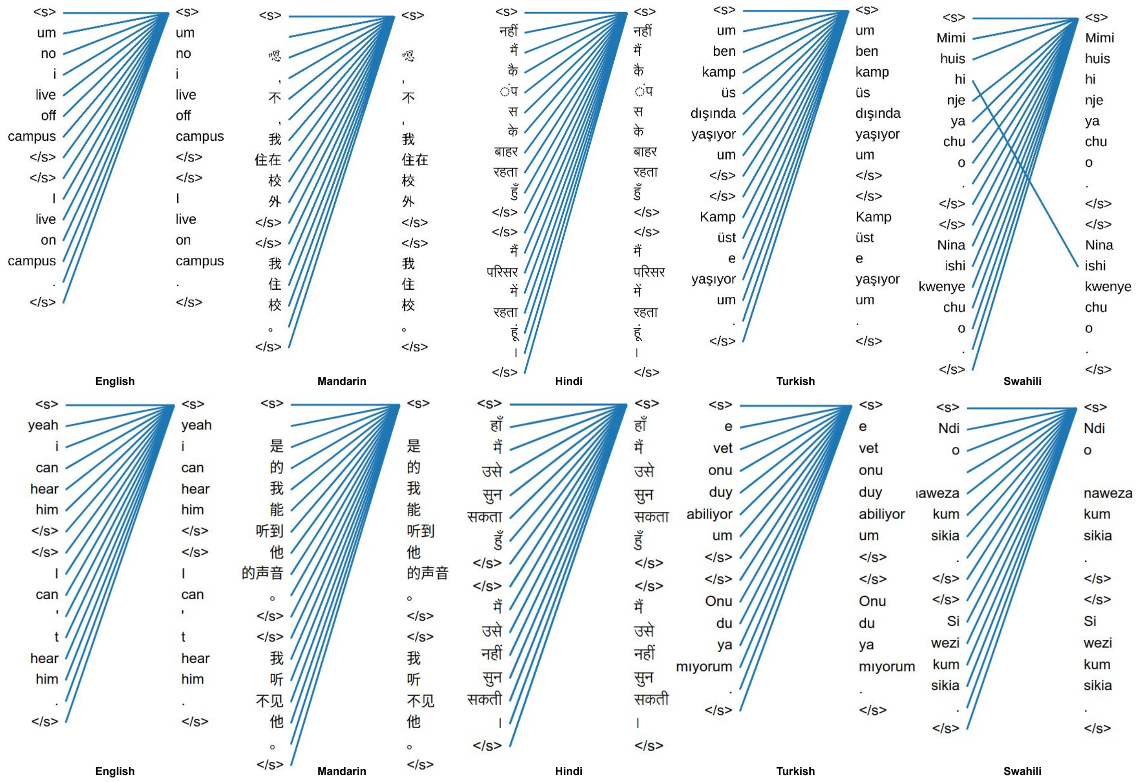


Figure 5: Attention of Layer 6, Head 5 of our XNLI model which is identified as language-specific to Swahili. Unlike the language-agnostic head, there is no obvious pattern in visualisation. However, in Section 5.2 we measure a significant difference in Swahili's attention to separator tokens compared to other languages.

attention pattern of the highest-ranked of the 4 heads which meet this criterion. The visualization highlights the same attention pattern across all languages: words from the premise are matched to near-synonyms in the hypothesis and vice versa.

The synonym-matching pattern clearly applies to NLI, where synonyms critically participate in commonalities and contradictions between the premise and hypothesis. Synonym linking is possible via token semantics and the separator tokens, so this pattern does not require any knowledge of language-specific syntax or morphology.

The visualization reveals a meaningful language-agnostic pattern which may explain why the positive Shapley Value across all languages. This usage highlights that while we utilize Shapley Values to remove harmful learned patterns, they also can direct mechanistic interpretability work to understand the effectiveness of transformers for a particular task (Wang et al., 2022).

5.2 Language-Specific Heads

As highlighted in Section 4.3, the fifth head of layer six has a positive Shapley Value only for Swahili. In Figure 5, we see that this head sometimes exhibits unique behavior for Swahili, connecting the incorrectly tokenized "ishi" suffix of "*Mimi Huishi*" and "*Ninaishi*" meaning "*I live*" in the Habitual and Present tense respectively. However, this use is not found frequently in our Swahili examples, as shown in the second example.

Therefore, we aim to understand whether the head functions in a measurably different fashion for Swahili across our entire dataset rather than on specific examples. Using the hypothesis from Clark et al. (2019) that attention to separator tokens indicates an inapplicable learned pattern, we look at the percentage of sentences where all tokens attend primarily to separators. This criterion is true in 56% Swahili XNLI inputs, but only 41% of non-Swahili inputs on average ($\sigma = 4.3\%$).

The frequency of separator attention combined with the minimal negative performance impact from removing this head for Swahili in Section 4.5 supports the idea that this head supports a rare pattern, perhaps stemming from poor tokenization. However, the relatively low rate of separator attention indicates that this head does impact other languages often, introducing noise.

6 Conclusions & Future Work

In this work, we developed a simple yet effective approach to measure the impact of individual attention heads on task performance by leveraging Shapley Values. We used this to identify language-specific and language-agnostic structural components of multilingual transformer language models. We demonstrated that the resulting values exhibit language affinity, varying across languages. We then applied these Attention Head Shapley Values to improve cross-lingual performance through pruning for both sequence classification and structured prediction. Finally, we performed provided insights on language-agnostic and language-specific attention heads using attention visualization.

We believe that attention head Shapley Values have strong potential to systematically inform future studies of multilingual models and transformers broadly. Future work should explore the relationship between linguistic features, training data volume, and the language-specificity of attention heads. Additionally, the benefits of removing heads motivates work that reduces cross-lingual interference introduced by language-specific components during pre-training, such as pruning during pre-training or utilizing sparsely activated networks.

7 Limitations

Even with our optimizations, using Shapley Values as an importance metric requires a significant computational cost compared to gradient-based methods: gradient-based methods take approximately 3.33e14 FLOPs and our optimized Shapley Values take approximately 3.27e16 FLOPs to converge. While the computation is parallelizable, it took several days on a single GPU to compute accurate estimates. This expense is reasonable for understanding the behavior of base models more deeply but limits the use of this method as a rapid iteration tool. For those looking to reduce this computational cost further, we recommend first using gradient-based methods to identify a set of heads to which the output is sensitive and then using Shapley Values to interpret the direction of the effect. While this may miss some harmful heads, it is likely to find the most harmful heads for a reduced cost.

Additionally, we rely on analysis of attention patterns to help ground our findings. However, there is debate as to whether analysis of attention patterns is a sound analytical tool (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019).

8 Acknowledgements

We are thankful to Caleb Ziems, Chris Hidey, Yanzhe Zhang, Hongxin Zhang, and our anonymous reviewers for their feedback. This work is supported in part by Cisco, an Amazon Faculty Research Award, and NSF grant IIS-2144562.

References

Ameen Ali, Thomas Schnake, Oliver Eberle, Grégoire Montavon, Klaus-Robert Müller, and Lior Wolf. 2022. XAI for transformers: Better explanations through conservative propagation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 435–451. PMLR.

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1796, Dublin, Ireland. Association for Computational Linguistics.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730. Selected papers presented at the Tenth International Symposium on Locational Decisions (ISOLDE X).

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 15834–15846. Curran Associates, Inc.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Wietse de Vries, Martijn Wieling, and Malvina Nissim. 2022. Make the best of cross-lingual transfer: Evidence from POS tagging with over 100 languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7676–7685, Dublin, Ireland. Association for Computational Linguistics.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2242–2251. PMLR.

Amirata Ghorbani and James Y Zou. 2020. Neuron shapley: Discovering the responsible neurons. In *Advances in Neural Information Processing Systems*, volume 33, pages 5922–5932. Curran Associates, Inc.

Benjamin Hoover, Hendrik Strobelt, and Sebastian Gehrmann. 2020. exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 187–196, Online. Association for Computational Linguistics.

Junjie Hu, Melvin Johnson, Orhan Firat, Aditya Siddhant, and Graham Neubig. 2021. Explicit alignment objectives for multilingual bidirectional encoders. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3633–3643, Online. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2984, Brussels, Belgium. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. 2019. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. NIPS'17, page 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Weicheng Ma, Kai Zhang, Renze Lou, Lili Wang, and Soroush Vosoughi. 2021. Contributions of transformer attention heads in multi- and cross-lingual tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1956–1966, Online. Association for Computational Linguistics.

Gideon S. Mann and David Yarowsky. 2001. Multipath translation lexicon induction via bridge languages. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Andreas Maurer and Massimiliano Pontil. 2009. Empirical bernstein bounds and sample-variance penalization. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*.

Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. 2017. Cheap translation for cross-lingual named entity recognition. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2536–2545, Copenhagen, Denmark. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 62–72.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Volodymyr Mnih, Csaba Szepesvári, and Jean-Yves Audibert. 2008. Empirical bernstein stopping. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 672–679, New York, NY, USA. Association for Computing Machinery.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When BERT Plays the Lottery, All Tickets Are Winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3208–3229, Online. Association for Computational Linguistics.

Lloyd S. Shapley. 1953. *A value for n-person games*, page 31–40. Cambridge University Press.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Jörg Tiedemann, Željko Agić, and Joakim Nivre. 2014. Treebank translation for cross-lingual parser induction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 130–140, Ann Arbor, Michigan. Association for Computational Linguistics.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small.

Zirui Wang, Zachary C. Lipton, and Yulia Tsvetkov. 2020. On negative interference in multilingual models: Findings and a meta-learning treatment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4438–4450, Online. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1513–1528. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Daniel Zeman. 2008. Reusable tagset conversion using tagset drivers. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

Mengjie Zhao and Hinrich Schütze. 2021. Discrete and soft prompting for multilingual models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8547–8555, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Why Don't You Do It Right?
Analysing Annotators' Disagreement in Subjective Tasks

Marta Sandri
Dept. of Humanities
University of Pavia, Italy
`sandri.marta97@gmail.com`

Elisa Leonardelli
Fondazione Bruno Kessler
Trento, Italy
`eleonardelli@fbk.eu`

Sara Tonelli
Fondazione Bruno Kessler
Trento, Italy
`satonelli@fbk.eu`

Elisabetta Jezek
Dept. of Humanities
University of Pavia, Italy
`jezek@unipv.it`

Abstract

Annotators' disagreement in linguistic data has been recently the focus of multiple initiatives aimed at raising awareness on issues related to 'majority voting' when aggregating diverging annotations. Disagreement can indeed reflect different aspects of linguistic annotation, from annotators' subjectivity to sloppiness or lack of enough context to interpret a text.

In this work we first propose a taxonomy of possible reasons leading to annotators' disagreement in subjective tasks. Then, we manually label part of a Twitter dataset for offensive language detection in English following this taxonomy, identifying how the different categories are distributed. Finally we run a set of experiments aimed at assessing the impact of the different types of disagreement on classification performance. In particular, we investigate how accurately tweets belonging to different categories of disagreement can be classified as offensive or not, and how injecting data with different types of disagreement in the training set affects performance. We also perform offensive language detection as a multi-task framework, using disagreement classification as an auxiliary task.

Warning: *This paper contains examples that may be offensive or upsetting.*

1 Introduction

The development of benchmark datasets based on the reconciliation of annotators' disagreement has been a standard practice within the NLP community for decades. However, in the last few years this paradigm has been questioned (Basile, 2020; Abercrombie et al., 2022), since aggregating discording labels is based upon the assumption that texts have a single interpretation and that annotators' disagreement is something that should be corrected. On the contrary, it may convey useful information on the task, the data and the annotators themselves (Aroyo and Welty, 2015). For example, annotators may disagree because of poorly described guidelines, or because they interpret in different ways a text based on their background and beliefs. The text may be ambiguous because of a lack of context, or annotators may simply be working poorly, without paying much attention to the task they should perform on the text (Uma et al., 2021).

Having a better understanding of the reasons behind annotators' disagreement could lead to better annotation guidelines and provide insights into the advantages and limits of developing NLP systems able to deal with disagreement (Davani et al., 2022). It would also contribute to a better understanding of how disagreement interferes with systems' performance. Finally, automatically detecting instances that are likely to obtain discording labels could help researchers in creating linguistic datasets with more or less challenging examples (Lehmann et al., 1996).

In this work, we contribute to the research line on annotators' disagreement by first presenting a taxonomy, where we classify possible reasons behind discording annotations in subjective tasks, i.e. tasks admitting diverse valid beliefs about what the correct data labels should be (Rottger et al., 2022). We also annotate part of an existing dataset for offensive language detection (Leonardelli et al., 2021) in order to assess the validity of our categorisation. We further perform several experiments aimed at addressing the following research questions: *i)* What are the most challenging categories of disagreement to classify in a task of abusive language detection? *ii)* What is the effect of including specific categories of disagreement in the training set? *iii)* Can multi-task learning be effectively used for offensive language detection using disagreement classification as auxiliary task? Are these

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2428–2441
May 2-6, 2023 ©2023 Association for Computational Linguistics

results consistent across disagreement categories?

Evaluation results can contribute to ongoing studies investigating the negative impact of disagreement on systems performance (Schwartz et al., 2011; Beigman Klebanov and Beigman, 2014; Leonardelli et al., 2021; Uma et al., 2021) by providing a more fine-grained view on disagreement types.

The annotated data used in our experiments can be obtained as an extension of Leonardelli et al. (2021)'s dataset, at this link: `https://github.com/dhfbk/annotators-agreement-dataset`.

2 Related work

Despite all the efforts to minimize inter-annotator disagreement, every large-scale project involving linguistic annotation has to deal with cases of diverging perceptions among human workers, especially in tasks where human sensibility is at play. However, disagreement due to interpretation divergences is also found in text annotation tasks usually perceived as objective, such as Part-of-Speech tagging (Plank et al., 2014), semantic role labeling (Dumitrache, 2019) or word sense disambiguation (Martínez Alonso et al., 2015).

In subjective tasks, the hypothesis that a single truth exists for all instances has been debated in several past works (Basile et al., 2021; Uma et al., 2021; Basile et al., 2022). Indeed, many researchers suggested that disagreement has to be treated as a signal, and not as noise (Aroyo and Welty, 2015; Plank et al., 2014; Basile et al., 2019). For this reason, Plank (2022) has recently proposed to use *human label variation* rather than the term *disagreement* to capture the fact that two or more views may sometimes be plausible in text annotation. Along the same line, different approaches have been proposed with the primary aim of amending the traditional way in which disagreement is dealt with (i.e., to treat it as noise and discard it from the training set), thus demonstrating that disagreement provides insights into human perception, linguistic data as well as classification systems.

Concerning the integration of disagreement in NLP classifiers, few approaches have been proposed so far (for an overview, see Uma et al. (2021)). Beside aggregating judgments based on majority voting, which has the clear shortcoming of reducing different voices and points of view in favour of the dominant one, some approaches consider disagreement as an indicator of the difficulty of the instances to be annotated (Reidsma and op den Akker, 2008). Past works have proposed to train separate classifiers, one for each annotator, and build an ensemble classifier that makes a prediction when all classifiers agree on the class label (Basile, 2020) or to adopt a multi-task architecture using a shared representation to model annotator disagreements (Davani et al., 2022). Similarly, Fornaciari et al. (2021) use soft-labels (i.e., probability distributions over the annotator labels) as an auxiliary task in a multi-task neural model. As an alternative, disagreement can also be excluded from the data by training the model only on items that show high agreement (Reidsma and op den Akker, 2008) or, conversely, models can be trained and evaluated only on disagreement-raising items (Beigman Klebanov and Beigman, 2014).

Concerning the analysis of annotators' behaviour on subjective tasks, past works showed that disagreement is to be expected (Basile, 2020; Davani et al., 2022). This is confirmed by Kenyon-Dean et al. (2018), showing that around 30% of the instances in a Twitter corpus for sentiment analysis are controversial, thus likely to lead to disagreement. They also propose to merge them in a new class of sentiment called "complicated", so that they are not excluded from the data. Sang and Stanton (2022) show that age and personality are factors that greatly influence annotators' perception of offensive content. Kocoń et al. (2021) show that disagreement can be reduced and annotation quality increased by combining text representations of labeled annotators' opinions with their demographic traits. Akhtar et al. (2019) deal with disagreement in hate speech annotation by partitioning the annotators into clusters reflecting more uniform subjective judgments.

As regards the categorisation of disagreement in subjective tasks, one of the first studies on this topic was presented in Beigman Klebanov et al. (2008), where in a metaphor annotation task a distinction is made between annotators' attentions slips and genuine disagreement. Basile et al. (2021) identify three main sources of disagreement: individual differences related to annotators' background and beliefs, stimulus characteristics, i.e. inherent ambiguity of texts, and context. Uma et al. (2021) further extend this categorisation by listing the following reasons behind disagreement: annotation errors, imprecise or vague annotation scheme, context-dependent text ambiguity, introduced in

Poesio et al. (2019), item difficulty and annotators' subjectivity. In our proposed taxonomy, we rely on previous works in the choice of categories, keeping the distinction between inherent text ambiguity and context-dependent one, as well as annotators' subjectivity. We further specify these categories by proposing subtypes that take into account different linguistic phenomena.

A recent work introducing a taxonomy of disagreement has also been presented in Jiang and de Marneffe (2022), which focuses on the task of natural language inference. Interestingly, the authors propose a three-layered taxonomy which has some high-level overlaps with ours, for example our *Ambiguity* class can be roughly mapped onto *Uncertainty in sentence meaning*, and *Subjectivity* onto *Annotator behavior*. The lower categories, however, are in some cases very task-specific. The analysis of task-specific and task-independent categories of disagreement may be an interesting research direction to explore in the future.

3 A Taxonomy of Disagreement

As a first step towards a better understanding of the reasons behind annotators' disagreement, we propose a taxonomy of disagreement for subjective tasks, which is built starting from past categorisations presented in the literature and iteratively adding subtypes based on the analysis of examples in existing datasets. An overview of the taxonomy is reported in Figure 1. The taxonomy includes four macro-categories, which are further specified through more fine-grained subtypes. For each category we report also selected examples, all taken from the dataset of disagreement presented in Leonardelli et al. (2021) (see a more detailed description of the dataset in Section 4). In the following, we illustrate each category in detail.

3.1 Sloppy Annotation

This category covers errors in the annotation due to annotators' carelessness. This can happen in particular with crowd-sourcing platforms, when annotators are recruited without a proper training and their annotation quality is not monitored. In general, this type of disagreement can be minimised by adopting tools that identify which annotators are not trustworthy (Hovy et al., 2013).

The only specification for this category is *Noise* (Figure 1), corresponding to messages clearly labeled with the wrong category, see for instance the tweet below annotated as "offensive":

(1) *In a singular voice, art across the world..*

3.2 Ambiguity

The second source of disagreement we include in the taxonomy is *Ambiguity*, that is a much debated topic in linguistics. It is generally referred to as a property of words or phrases to allow more than one interpretation (Tuggy, 1993; Cruse, 2004). This category comprises mainly cases of figurative language, a phenomenon related to the use of words with a diverging meaning from their literal use in order to express colorful images and emotions (Dancygier and Sweetser, 2014). This category includes six subtypes:

Analogy. This label includes comparison mechanisms, such as simile and metaphor, along with the figure of speech of analogy. Both these phenomena involve the cognitive concept of mapping between two conceptual domains (Dancygier and Sweetser, 2014). Below we report an example of metaphor:

(2) *Trump is a walking petri dish. His goal is to spread the virus to as many people as possible.*

False Assertion. Under this label we group all the cases that are characterized by an assertion that is false if compared to the reality of facts and that can therefore trigger irony (Cignarella et al., 2018). In other words, users express the opposite of what they think or something false and exaggerated in relation to the context. This figure of speech needs knowledge of the world to be understood and this is the main reason behind disagreement. An example is reported below:

(3) *Another attempt backfired on them, George Floyd cured Covid-19 and opened up the economy!*

Rhetorical Question. We group under this label all the instances containing a question asked not to obtain an answer but with the purpose of rhetorically pointing out a concept (Stivers and Enfield, 2010), see example below:

(4) *why do we treat our prisoners like this? Is it really because we decided once you commit a crime you're worthless non-human?*

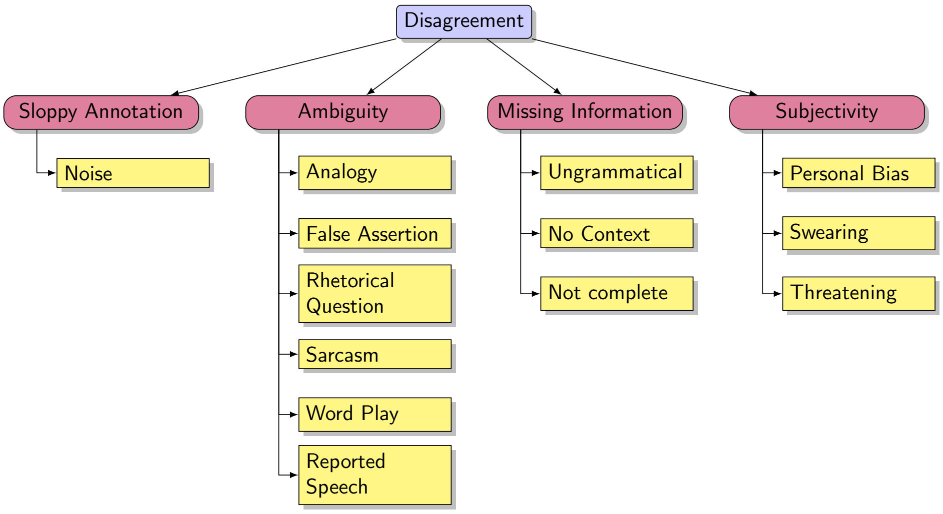


Figure 1: Taxonomy of annotators' disagreement in subjective tasks. Pink boxes represent coarse-grained categories, while yellow boxes correspond to subtypes.

Sarcasm. Sarcasm is characterized by words employed to express the opposite of their literal meaning, mainly used to make fun of a specific topic or person. Sarcasm needs context and other extra-linguistic expressions like pauses and intonation to be understood and it features a disproportion of emphasis regarding the real situation, giving a caustic effect (Gibbs, 1986). However, in writing, these extra-linguistic hints are not possible, thus we must rely on our knowledge of the world or of the addressee to understand sarcasm.

(5) *Who knew a side effect of COVID would be gross incompetence.*

Word Play. Word Play is a figure of speech that involves literary devices to alter some words, with the purpose of giving proof of someone's wit. Among the literary techniques used to convey word play are acronyms, alliterations, i.e. repeating the same sounds in a sentence, and puns, i.e. using words with multiple meanings to obtain a humorous result.

(6) *The only people ripping this country apart are your fellow liberal #DemocRATS and your militant concubines.*

Reported Speech. It is commonly defined as the presentation of discourse that purports to be from a prior occasion, and may originate from another author (Holt, 2009). It can be ambiguous because it can be mistaken as something written by the same author of the text to be annotated.

(7) *White Bystanders With Rifles Stare Down George Floyd Protesters: 'You Ain't Got No Guns'*

3.3 Context sensitivity: Missing Information

In this category we group all cases in which annotators' disagreement may be caused by a lack of information or of context to unambiguously interpret a text (Donaldson and Lepore, 2012). It includes the three following subtypes.

Ungrammatical and Non Standard Language. We assign this label to all the texts that could cause disagreement due to marked uses of the language (e.g. non-standard varieties), use of slang, code-switching or mere typing errors. Intuitively, annotators who do not speak a specific language variety are more likely to misunderstand or misinterpret it (Sap et al., 2019), leading to disagreement. See the example below:

(8) *mane it's hard for some of da blaxk people out dere when dey go into a store they got everybody looking at them "what they doing"*

No Context. This class covers well-known linguistic phenomena which need context to be unequivocally interpreted such as anaphora and deixis (Poesio and Artstein, 2005). We include in this category also messages containing links, typically used in online communication. Deictic expressions include devices such as demonstratives (*this, that*, etc.), personal pronouns (*I, you, he, she*, etc.), possessives (*his, her, their*, etc.) (Levinson, 2004). As regards anaphoric expressions, instead, the interpretation of the textual content is determined by that of the antecedent (Levinson, 1987), therefore, if there is not enough context, crucial information may be missing. We report below an example containing both a deictic expression (*this*) and an anaphoric pronoun (*them*):

(9) *Dude this guy is serious? And trump retweeted this?????? Please anonymous take them out*

Not Complete. In the last subclass of *Missing Information*, we group all cases that do not convey complete information and do not fall into any other subcategory. A typical example is social media threads written by a user. If just one message is extracted from a thread and annotated, it may be very difficult to understand what its tone and meaning is. As an example, see below:

(10) *Wtaf, a farce in three parts:*

3.4 Bias and Desemanticization: Subjectivity

We group in this category cases of disagreement due to the annotators' identity, beliefs and background, which have been recognised as leading to biased judgments (Sap et al., 2022). This is a major source of disagreement in subjective tasks such as hate speech detection, sentiment analysis or political stance detection. The category includes three subtypes.

Personal Bias. Although it is not easy to identify cases of disagreement due to personal biases, in particular when annotators are unknown, this category can be inferred because it is very likely to occur when the text to be labeled belongs to a divisive topic, such as politics, covid-19, social movements, etc. (Wich et al., 2020). See for example the sentence below:

(11) *#DemocratsAreDestroyingAmerica #BlackLivesMatter is a terrorist organization*

Swearing. Another cause of disagreement related to annotators' subjectivity is how they interpret swear words. Indeed, some annotators perceive specific swear words as offensive and hurtful, while for others they appear to be desemanticized (Ljung, 2011). A typical example in English is the use of the word *bitch*:

(12) *2nd wave about to be a bitch*

Threatening. Disagreement may arise from texts containing linguistically violent expressions or threats and depend on annotators' sensitivity to verbal violence and menaces (Storey, 1995), see for instance, the following text:

(13) *U r going to jail.*

4 Data Annotation and Analysis

We apply the taxonomy illustrated above to the dataset of disagreement in abusive language presented in Leonardelli et al. (2021), which contains more than 10k English tweets labeled as offensive or not offensive by five crowd-workers, and covers three topics: covid-19, US Presidential elections and Black Lives Matter movement. The dataset has been designed so as to include a balanced set of tweets with full agreement (A^{++}), with partial agreement (A^{+} class, 4 matching labels versus 1) and with disagreement (A^{0} class, 2 vs. 3 labels) and has been released divided into a balanced training and test split.

We manually annotate all A^{+} and A^{0} tweets in the test set (1,756 in total), plus a portion of tweets from the training set (809 tweets). A total of 2,574 tweets is annotated, divided in 1,518 for the A^{0} agreement level and 1,056 for the A^{+} agreement level. Annotation was performed by a trained linguist, and during the process the initial taxonomy was adjusted by refining the category definitions or introducing new ones when needed. A second linguist annotated a sample of 200 tweets divided equally into A^{0} and A^{+} following the annotation guidelines (see Appendix B). Cohen's Kappa is 0.591, which corresponds to a moderate agreement. After computing agreement, a discussion and adjudication phase was conducted.

Based on the annotator's feedback, we foresaw the possibility to assign multiple labels to the same tweet when the source of disagreement could refer to two or more categories in the taxonomy, considering however the first label as the most probable interpretation. A multi-category annotation scheme was adopted for the same reason also in Jiang and de Marneffe (2022).

In Table 1 we report the distribution of annotated tweets by category and by agreement level.

In case of multiple labels, we include the first one in the statistics. However, tweets annotated as belonging to more than one category are around 40%, showing that disagreement is often due to a mix of different linguistic phenomena. The category with the most multiple annotations is *Subjectivity*, labeled together with *Missing Information* in most of the cases.

	A^0	A^+
Subjectivity	996	703
Ambiguity	302	201
Missing Information	218	142
Sloppy Annotation	2	10
Total	**1,518**	**1,056**

Table 1: Number of annotated tweets by category and agreement level. A^0 is 3 vs. 2 judgments; A^+ is 4 vs. 1 judgment.

Table 1 shows that the cases leading to disagreement because of annotators' subjectivity cover most of the tweets in the dataset, and that the ranking of the four categories is the same for the two agreement levels. As expected, clear annotation mistakes (i.e. *Sloppy Annotation* category) are more frequent for A^+, but in general they have a minimal impact on the cases of disagreement. This is probably due to the fact that the dataset in Leonardelli et al. (2021) was created adopting a strict quality control protocol aimed at excluding low-quality crowd-workers.

In Figure 2 we report how the subtypes are distributed in the annotated dataset. As shown in previous studies, swear words are often a signal for a hateful attitude, but they are also used in casual contexts with positive social functions (Pamungkas et al., 2020). This double interpretation is likely the main reason why the *Swearing* subtype is very frequent in our dataset of disagreement. Concerning *Personal Bias*, its relevance is probably related to the fact that the annotated tweets deal with controversial topics such as US American elections, Black Lives Matter and covid-19. For instance, few tweets targeting Trump were labeled as not offensive by annotators, who were likely to be Biden supporters, and vice versa.

If we compare the above statistics with the analysis reported in Jiang and de Marneffe (2022) on disagreement in natural language inference (NLI), we observe that the most frequent causes of disagreement are task-specific: in our dataset, they are mostly due to annotators' subjectivity and their perception of what is offensive, while in NLI they often stem from the underspecified meaning of lexical items in the sentence pairs to be labeled, or in the probabilistic nature of the inferred content.

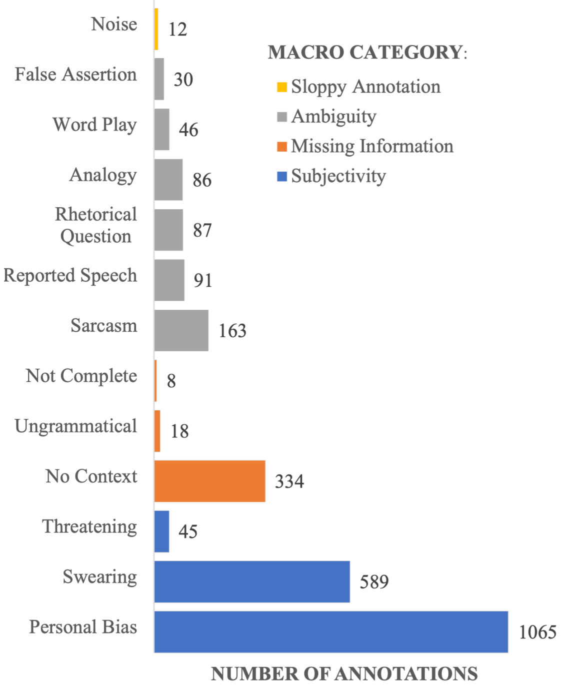


Figure 2: Summary of annotated data by main category and subtype.

5 Experiments

In the next subsections we describe a series of experiments conducted on the annotated dataset, aimed at analysing the relationship between annotators' disagreement and various aspects of automatic offensive language detection. For all the experiments described below, we use the MaChAmp v0.2 toolkit (van der Goot et al., 2021), a classification tool that allows easy implementation of transformers-based classification tasks and supports single-task and multi-task learning. For all the experiments we employ BERT-base uncased (Devlin et al., 2019) (110M parameters) and perform 20 restarts. We keep the default hyperparameter setting of MaChAmp, i.e. max seq length 128, batch size 32, 0.3 dropout, 10 epochs. All the results reported in the following subsections are the average values of 20 runs. All experiments are run on a NVIDIA Quadro RTX 5000 GPU.

Subtype	Main category	Test size	micro F1		
			All	Offensive	Not offensive
Swearing	*Subjectivity*	427	$75.74_{\pm 0.64}$	89.45	39.83
Rhetorical Question	*Ambiguity*	60	$71.75_{\pm 4.00}$	65.79	74.51
Not complete	*Missing Info*	7	$71.43_{\pm 12.78}$	-	-
No context	*Missing Info*	183	$71.26_{\pm 2.35}$	43.75	77.09
Reported Speech	*Ambiguity*	82	$70.37_{\pm 2.62}$	60.24	73.85
Threatening	*Subjectivity*	41	$68.78_{\pm 4.12}$	74.12	65.00
Word Play	*Ambiguity*	33	$65.61_{\pm 3.22}$	65.31	65.88
Personal Bias	*Subjectivity*	729	$64.98_{\pm 2.05}$	63.35	65.89
Ungrammatical	*Missing Info*	14	$64.29_{\pm 5.05}$	53.12	79.17
Sarcasm	*Ambiguity*	97	$63.71_{\pm 3.49}$	58.75	65.34
Analogy	*Ambiguity*	62	$60.56_{\pm 2.91}$	70.52	51.82
False Assertion	*Ambiguity*	20	$53.00_{\pm 6.20}$	37.22	65.91

Table 2: Classification performance of the best system from Leonardelli et al. (2021) for each category and subtype of disagreement. We report the average F1 obtained from 20 restarts and, for the overall results, also the standard deviation.

5.1 Classification performance on disagreement categories

Our first experiment aims at analysing differences in classification performance among disagreement categories and subtypes. To this end, we use the best model for offensive language classification previously described in Leonardelli et al. (2021), which was trained using only tweets with perfect (A^{++}) and high (A^{+}) agreement. We run this model on the tweets in our dataset that belong to the test set of the original work, and calculate separated performance scores for each category and subtype. We report the results in Table 2.

The best performance is obtained on the *Swearing* subtype (75.74 micro-F1). However, the results on the two classes, i.e. Offensive and Not offensive, show that this high F1 mainly depends on the good performance yielded on the offensive class. Indeed, swear words tend to be very predictive of offensive content, and have already been recognised in previous studies as so-called *authentic artifacts*, i.e. highly-discriminating and informative tokens in conveying hatefulness (Ramponi and Tonelli, 2022). On the contrary, swear words in non-offensive tweets are both controversial for human annotators and difficult to detect for classifiers (Pamungkas et al., 2020).

For the most numerous subtype, *Personal Bias*, performance is rather low compared to the other types. However, the classifier yields a comparable performance on the offensive and not offensive class, showing that the two classes are equally challenging when annotators' beliefs and background come into play. In general, classification performance on offensive tweets is lower than on not offensive ones, except for *Swearing*, *Threatening* and *Analogy*. Offensive language detection systems tend to perform better on the not offensive class, because it is usually represented by more examples in the training set. Our experiment confirms this trend with few exceptions.

As a further analysis, in Figure 3 we report the classification results (average of 20 runs) on the different subtypes for A^0 and A^+ tweets. For all the categories except for *Analogy*, the classification performance is better on A^+ cases (low disagreement) compared to A^0 (high disagreement). This is probably because assigning a label to A^0 cases through majority voting is rather arbitrary, leading to cases that a system can poorly classify.

5.2 Training with disagreement

Several works showed that training a classifier using data with a low level of agreement is detrimental to the system performance (Reidsma and op den Akker, 2008; Jamison and Gurevych, 2015). We delve further into this issue by evaluating whether this negative effect depends on the presence of a specific class of disagreement in the training set. To this end, we retrain a classification model for offensive language detection using the original training split used in Leonardelli et al. (2021), and we compare it with the performance obtained including in the same training set only the subset of A^0 tweets belonging to a specific category, i.e. either *Subjectivity* or *Missing Information* or *Ambiguity*. The performance of the models is evaluated on the same three categories in the test set. Results are reported in Table 3.

To reliably compare differences between models' performances, we use Almost Stochastic Order (Dror et al., 2019; Del Barrio et al., 2018) in its

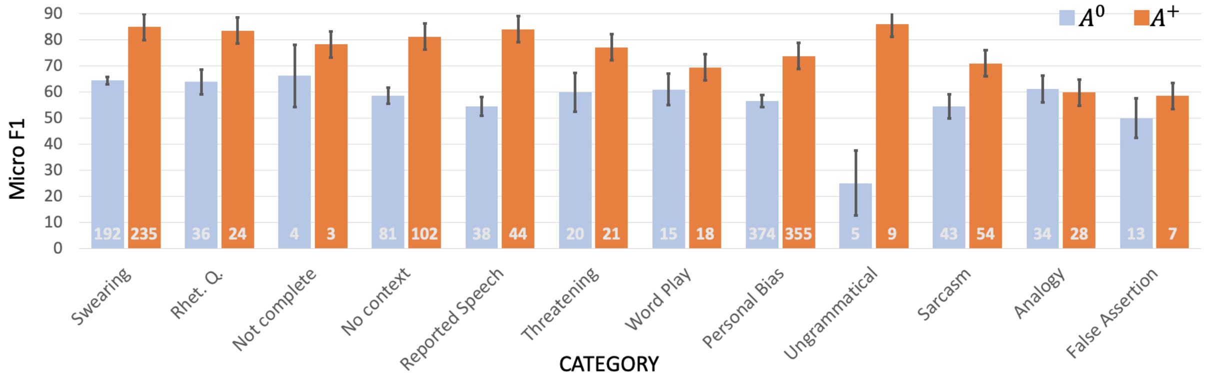


Figure 3: Performance of the classifier for the different agreement levels and categories. Error bars represent standard deviation obtained from 20 restarts.

Training split	**Train. size**	**Testing on**		
		Subj.	*Missing*	*Amb.*
$A^{++/+}$	1,800	68.95	70.78	65.64
$A^{++/+/0}$	2,700	68.79	69.71	**66.31***
$A^{++/+/0(SUBJ)}$	2,206	68.55	69.29	65.48
$A^{++/+/0(MISS)}$	1,927	68.93	**70.84***	65.79
$A^{++/+/0(AMB)}$	1,923	**69.07**	69.48	64.58

Table 3: Classifier performance (F1) with different versions of the training set, with and without specific classes of disagreement. Statistically significant results (compared to the lowest F1) are marked with (*).

implementation by Ulmer et al. (2022). For each of the three test sets, we compare the models scores across the 20 restarts. For statistical significance a threshold of $\tau = 0.2$ is considered.[1]

As shown in Table 3, while the differences in performance when testing on the *Subjectivity* category are not statistically significant, the best scores obtained on *Ambiguity* and on *Missing Information* show a statistically significant improvement over the lowest F1 and, in the case of *Missing Information*, also over training with $A^{++/+/0(AMB)}$.

For the classification of examples from *Missing Information*, adding only the A^0 examples belonging to the same category of disagreement yields to the best performance among all models. On the contrary, when classifying tweets in the *Ambiguity* class, the best performance is obtained when all A^0 are added to the training set regardless of the tweet category or disagreement level. The lowest performance, instead, is the obtained when adding only A^0 examples from the *Ambiguity* class. This may be due to the fact that this class is the most heterogeneous one, with different subtypes all represented in the data with few examples, covering very different linguistic phenomena. Overall, these results suggest that removing A^0 instances from training is not always the best solution, contrary to what was suggested in previous works (Leonardelli et al., 2021). Instead, distinctions should be made among different types of disagreement when deciding whether to remove training instances or not.

5.3 Multi-task learning with disagreement

Finally, we investigate whether using information about disagreement can improve offensive language detection. We employ a multi-task framework, which has already been used to include disagreement information in classification tasks (Fornaciari et al., 2021; Davani et al., 2022; Ramponi and Leonardelli, 2022). In a multi-task setting, the encoder component is unique and shared between both tasks that during training are jointly fine-tuned. In our case, we consider offensive language detection as the primary task, and disagreement detection as the secondary one, to test whether the latter can provide useful signals to potentially improve the performance on the main task. We test two variants in this respect: *i)* we cast the auxiliary task as a three-way classification aimed at recognising tweets labeled as A^{++}, A^{+} and A^0, and *ii)* we implement a more fine-grained version of the previous task, aimed at assigning tweets to one of six classes: A^{++} offensive and not offensive, A^{+} offensive and not offensive and A^0 offensive and not offensive. All these labels were already provided in the dataset by Leonardelli et al. (2021), so no additional annotation was required. The classifier for offensive language detection is tested separately on the three classes of disagreement *Subjectivity*, *Missing Information* and *Ambiguity*, like in the ex-

---

[1] Based on Ulmer et al. (2022), this threshold is comparable to a Type I error rate of p-value .05

periments in Section 5.2.

Results are shown in Table 4. The first row presents the single-task setting, i.e. offensive language detection, compared with the multi-task ones, i.e. three-class and six-class classification.

Multitask		**Testing on**		
Task 1	**Task 2**	*Subj.*	*Missing*	*Amb.*
Offensive language	-	68.79	69.61	**66.31**
Offensive language	Agr. level $A^{++,+,0}$	69.15	**69.84**	66.26
Offensive language	Agr. level $N/O^{++,+,0}$	**69.24**	69.34	65.82

Table 4: Classification performance (F1) with multi-task learning. N=Not offensive; O=Offensive.

Although the differences across settings are slight, the best result for the *Subjectivity* category is obtained within the multitask framework with 6-way classification as an auxiliary task. For *Missing Information*, instead, the multitask setting with three-way classification is the best one, while for *Ambiguity* providing auxiliary information on disagreement levels does not seem to yield any improvement. These differences support our intuition that we should distinguish among the different types of disagreement, since different strategies would be necessary to deal with them during classification. However, a statistical analysis similar to the one presented in previous section failed to reveal any significant difference between the models' performances.

6 Conclusions

In this work, we first introduced a two-layered taxonomy for the classification of annotators' disagreement in subjective tasks, consisting in four main categories and a number of subtypes aimed at covering different linguistic phenomena. We then annotated part of an existing dataset developed to study disagreement with the above classes and subtypes. A first analysis shows that the *Subjectivity* class is the prevalent one in the dataset, and that *Personal Bias* and *Swearing* are two major reasons leading to frequent cases of disagreement among annotators for the task of offensive language detection. Secondly, we run several experiments to gain novel insights into disagreement phenomena. In particular, we investigate whether a system for offensive language detection is more prone to wrong classification on specific classes of disagreement. Our results show that the presence of *Sarcasm*, *Analogy* and *False Assertions* negatively affects classifier performance, while *Swearing* obtains the best classification results, despite showing a bias in favour of offensive tweets. Furthermore, cases with high disagreement are generally more difficult to classify than those with mild disagreement for all categories except for *Analogy*. In a second experiment, we show that adding instances of *Missing Information* to the training set, even if they belong to the A^0 class, has a positive effect on the classification of this specific class, while this is not true for *Ambiguity*, probably because it contains more heterogeneous data. Finally, we show that adding disagreement information as an auxiliary task in a multi-task setting, having offensive language detection as the main task, is not generally better, and has different effects on the three above classes.

As regards tweets annotated with multiple labels, we observe that it is rather frequent to find more than one cause of disagreement (around 40% of the tweets in our dataset). We performed a preliminary experiment (not reported in this paper) comparing the performance of the best system configuration on single-label and multi-label examples, and we observed no significant difference. We will further investigate this aspect and compare single- and multi-label items in detail in the future.

In general, we hope that this work can contribute to the ongoing debate on the importance of considering, and not removing, disagreement when creating datasets and when developing classifiers. Furthermore, we advocate for a differentiation of the types of disagreement, showing that their presence in training and test data can have different effects on classification.

Limitations

While the taxonomy of disagreement is designed to be language-independent and to cover the annotation of subjective tasks, we have applied it only to a dataset for offensive language detection in English. Its applicability to other tasks will be investigated in the near future, together with its portability across languages. Also, the small size of the annotated dataset may limit the generalisability of our findings. In particular, the differences in performance across settings in our experiments are not always statistically significant.

Ethics Statement

We do not foresee specific ethical risks related to the current work. On the contrary, the analysis of different types of disagreement is aimed also at making the role of subjective annotations accepted within the NLP research community, making sure that the voices of minorities are included. Indeed, this work contributes to providing methodologies to distinguish subjective annotations from mistakes and poor-quality judgments.

Acknowledgements

Elisa Leonardelli and Sara Tonelli have been funded by the STAND BY ME European project (REC-RDAP-GBV-AG-2020) on "Stop online violence against women and girls by changing attitudes and behaviour of young people through human rights education" (GA 101005641). They have also been supported by the STAND BY ME 2.0 project (CERV-2021-DAPHNE) on "Stop gender-based violence by addressing masculinities and changing behaviour of young people through human rights education" (GA 101049386).

References

Gavin Abercrombie, Valerio Basile, Sara Tonelli, Verena Rieser, and Alexandra Uma, editors. 2022. *Proceedings of the 1st Workshop on Perspectivist Approaches to NLPerspectives@LREC 2022, Marseille, France, 20th June 2022*. European Language Resources Association.

Sohail Akhtar, Valerio Basile, and Viviana Patti. 2019. A new measure of polarization in the annotation of hate speech. In *Proceedings of AI*IA*.

L. Aroyo and C. Welty. 2015. Truth is a lie: Crowd truth and the seven myths of human annotation. *AI Magazine*, 36(1):15–24.

V. Basile. 2020. It's the end of the gold standard as we know it. on the impact of pre-aggregation on the evaluation of highly subjective tasks. In *Proceedings of DP@AI*IA*.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Valerio Basile, Tommaso Caselli, Alexandra Balahur, and Lun-Wei Ku. 2022. Bias, subjectivity and perspectives in natural language processing. *Frontiers in Artificial Intelligence*, 5.

Valerio Basile, Michael Fell, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, Massimo Poesio, and Alexandra Uma. 2021. We need to consider disagreement in evaluation. In *Proceedings of the 1st Workshop on Benchmarking: Past, Present and Future*, pages 15–21, Online. Association for Computational Linguistics.

Beata Beigman Klebanov and Eyal Beigman. 2014. Difficult cases: From data to learning, and back. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 390–396, Baltimore, Maryland. Association for Computational Linguistics.

Beata Beigman Klebanov, Eyal Beigman, and Daniel Diermeier. 2008. Analyzing disagreements. In *Coling 2008: Proceedings of the workshop on Human Judgements in Computational Linguistics*, pages 2–7, Manchester, UK. Coling 2008 Organizing Committee.

Alessandra Teresa Cignarella, Cristina Bosco, Viviana Patti, and Mirko Lai. 2018. Application and analysis of a multi-layered scheme for irony on the Italian Twitter corpus TWITTIRÒ. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

D.A. Cruse. 2004. *Meaning in Language: An Introduction to Semantics and Pragmatics*. Oxford. Oxford University Press.

B. Dancygier and E. Sweetser. 2014. *Figurative Language*. CUP.

Aida Mostafazadeh Davani, Mark Díaz, and Vinodkumar Prabhakaran. 2022. Dealing with disagreements: Looking beyond the majority vote in subjective annotations. *Transactions of the Association for Computational Linguistics*, 10:92–110.

Eustasio Del Barrio, Juan A Cuesta-Albertos, and Carlos Matrán. 2018. An optimal transportation approach for assessing almost stochastic order. In *The Mathematics of the Uncertain*, pages 33–44. Springer.

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

T. Donaldson and E. Lepore. 2012. Context-sensitivity. In Gillian Russell and Delia Graff Fara, editors, *Routledge Companion to Philosophy of Language*, pages 116–131. New York, NY, USA: Routledge.

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep dominance - how to properly compare deep neural models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2773–2785. Association for Computational Linguistics.

A. Dumitrache. 2019. *Truth in Disagreement: Crowdsourcing Labeled Data for Natural Language Processing*. Ph.D. thesis, Vrije Universiteit Amsterdam.

Tommaso Fornaciari, Alexandra Uma, Silviu Paun, Barbara Plank, Dirk Hovy, and Massimo Poesio. 2021. Beyond black & white: Leveraging annotator disagreement via soft-label multi-task learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2591–2597, Online. Association for Computational Linguistics.

R. W Gibbs. 1986. On the psycholinguistics of sarcasm. *Journal of experimental psychology*, 115(1):3.

E. Holt. 2009. The pragmatics of interaction. *Handbook of Pragmatics Highlights*, 4:190–205.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia. Association for Computational Linguistics.

Emily Jamison and Iryna Gurevych. 2015. Noise or additional information? Leveraging crowdsource annotation item agreement for natural language tasks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 291–297.

Nan-Jiang Jiang and Marie-Catherine de Marneffe. 2022. Investigating Reasons for Disagreement in Natural Language Inference. *Transactions of the Association for Computational Linguistics*, 10:1357–1374.

Kian Kenyon-Dean, Eisha Ahmed, Scott Fujimoto, Jeremy Georges-Filteau, Christopher Glasz, Barleen Kaur, Auguste Lalande, Shruti Bhanderi, Robert Belfer, Nirmal Kanagasabai, Roman Sarrazingendron, Rohit Verma, and Derek Ruths. 2018. Sentiment analysis: It's complicated! In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1886–1895, New Orleans, Louisiana. Association for Computational Linguistics.

J. Kocoń, A. Figas, M. Gruza, D. Puchalska, T. Kajdanowicz, and P. Kazienko. 2021. Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach. *Information Processing & Management*, 58(5):102643.

Sabine Lehmann, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Herve Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. 1996. TSNLP - test suites for natural language processing. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Elisa Leonardelli, Stefano Menini, Alessio Palmero Aprosio, Marco Guerini, and Sara Tonelli. 2021. Agreeing to disagree: Annotating offensive language datasets with annotators' disagreement. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10528–10539, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

S. C. Levinson. 2004. Deixis. In *The handbook of pragmatics*, pages 97–121. Blackwell.

S.C. Levinson. 1987. Pragmatics and the grammar of anaphora: a partial pragmatic reduction of binding and control phenomena. *Journal of Linguistics*, 23(2):379–434.

M. Ljung. 2011. *Swearing: A Cross-Cultural Linguistic Study*. Palgrave Macmillan.

Héctor Martínez Alonso, Anders Johannsen, Oier Lopez de Lacalle, and Eneko Agirre. 2015. Predicting word sense annotation agreement. In *Proceedings of the First Workshop on Linking Computational Models of Lexical, Sentential and Discourse-level Semantics*, pages 89–94, Lisbon, Portugal. Association for Computational Linguistics.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Do you really want to hurt me? predicting abusive swearing in social media. In *The 12th Language Resources and Evaluation Conference*, pages 6237–6246. European Language Resources Association.

Barbara Plank. 2022. The "problem" of human label variation: On ground truth in data, modeling and evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10671–10682, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Barbara Plank, Dirk Hovy, and Anders Søgaard. 2014. Linguistically debatable or just plain wrong? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 507–511, Baltimore, Maryland. Association for Computational Linguistics.

Massimo Poesio and Ron Artstein. 2005. The reliability of anaphoric annotation, reconsidered: Taking ambiguity into account. In *Proceedings of the Workshop on Frontiers in Corpus Annotations II: Pie in the Sky*, pages 76–83, Ann Arbor, Michigan. Association for Computational Linguistics.

Massimo Poesio, Jon Chamberlain, Silviu Paun, Juntao Yu, Alexandra Uma, and Udo Kruschwitz. 2019. A crowdsourced corpus of multiple judgments and disagreement on anaphoric interpretation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1778–1789, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Ramponi and Elisa Leonardelli. 2022. DH-FBK at SemEval-2022 task 4: Leveraging annotators' disagreement and multiple data views for patronizing language detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 324–334, Seattle, United States. Association for Computational Linguistics.

Alan Ramponi and Sara Tonelli. 2022. Features or spurious artifacts? data-centric baselines for fair and robust hate speech detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3027–3040, Seattle, United States. Association for Computational Linguistics.

Dennis Reidsma and Rieks op den Akker. 2008. Exploiting 'subjective' annotations. In *Coling 2008: Proceedings of the workshop on Human Judgements in Computational Linguistics*, pages 8–16, Manchester, UK. Coling 2008 Organizing Committee.

Paul Rottger, Bertie Vidgen, Dirk Hovy, and Janet Pierrehumbert. 2022. Two contrasting data annotation paradigms for subjective NLP tasks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 175–190, Seattle, United States. Association for Computational Linguistics.

Y. Sang and J. Stanton. 2022. The origin and value of disagreement among data labelers: A case study of individual differences in hate speech annotation. In *Information for a Better World: Shaping the Global Future*, pages 425–444, Cham. Springer International Publishing.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A. Smith. 2022. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5884–5906, Seattle, United States. Association for Computational Linguistics.

Roy Schwartz, Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 663–672, Portland, Oregon, USA. Association for Computational Linguistics.

T. Stivers and N.J. Enfield. 2010. A coding scheme for question–response sequences in conversation. *Journal of Pragmatics*, 42(10):2620–2626.

Kate Storey. 1995. The language of threats. *International Journal of Speech Language and the Law*, 2:74–80.

D. Tuggy. 1993. Ambiguity, polysemy, and vagueness. *Cognitive Linguistics*, 4:273–290.

Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. 2022. deep-significance: Easy and meaningful signifcance testing in the age of neural networks. In *ML Evaluation Standards Workshop at the Tenth International Conference on Learning Representations*.

A. Uma, T. Fornaciari, D. Hovy, S. Paun, B. Plank, and M. Poesio. 2021. Learning from disagreement: A survey. *Journal of Artificial Intelligence Research*, Vol. 72.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multitask learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Maximilian Wich, Jan Bauer, and Georg Groh. 2020. Impact of politically biased data on hate speech classification. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 54–64, Online. Association for Computational Linguistics.

A Example Annotations

We report in Table 5 some example tweets with different degrees of annotators' agreement and the corresponding category & subtype. In particular, A^{++} corresponds to total agreement (5/5 crowdsourced annotations with the same label); A^{+} - mild agreement (4 vs 1 labels); A^{0} - disagreement (3 vs 2 labels).

Agreement	Offensiveness	Example	Category	Subtype
A^{++}	N^{++}	Google bans ads on coronavirus conspiracy theory content -url	-	-
A^{+}	N^{+}	@user Freaking love it	Missing Info	No Context
A^{0}	N^{0}	@user How many more George Floyd will die under white domination! Sad!	Ambiguity	Rethorical question
A^{0}	O^{0}	He is another nut case.	Subjectivity	Personal bias
A^{+}	O^{+}	@user You're a bumbling fool #elections2020	Subjectivity	Personal bias
A^{++}	O^{++}	f %* king insanity	-	-

Table 5: Examples of tweets with different degrees of crowd-workers' agreement. N=Not offensive, O=Offensive. $++/+/0$ correspond to high, medium and low agreement respectively. The last two columns include disagreement category and subtype.

B Annotation Guidelines

In order to annotate a text by assigning a category and a subtype of disagreement from the proposed taxonomy, annotators should perform the following steps:

1. Check the text for sources of disagreement: detect the main one and assign it to a category among *Sloppy annotation*, *Ambiguity*, *Missing Information* and *Subjectivity*;
2. Choose a fine-grained class to specify the main source of disagreement
3. Check for a secondary source of disagreement (if any): assign the text to another category (it could be the same as the previous one, with a different subtype)
4. Choose a secondary subtype to specify the secondary source of disagreement

We also report questions to guide the assignment of labels to text instances.

- **Sloppy Annotation** (Label = Sloppy_Annotation)
  - **Noise**: is the text clearly not offensive but marked as such (or vice versa)? (Label = Noise)
- **Ambiguity**: are there multiple interpretations to the text but is it not clear which is the correct one? (Label = Ambiguity)
  - **Analogy**: does the text include a figure of speech that comprehends mechanisms of comparison (included: simile and metaphor) or is the user referring to someone with a periphrasis (e.g., "orange monkey", "bunker boy" for Donald Trump)? (Label = Analogy)
  - **False assertion**: does the user express the opposite of what they think or something wrong with respect to a context? (Label = False_Assertion)
  - **Rhetorical question**: does the text include a question asked in order to make a point rather than to elicit an answer? (Label = Rhetorical_Question)
  - **Sarcasm**: is the text employed to communicate the opposite of its surface meaning in a humorous way or to mock someone/something? (Label = Sarcasm)
  - **Word Play**: does the text include any acronyms, alliterations or puns? (Label = Word_Play)
  - **Reported Speech**: does the text report something someone else stated? For example a newspaper headline? (Label = Reported_Speech)
- **Missing Information**: is the disagreement caused by difficulty of interpretation? (Label = Missing_Info)
  - **Ungrammatical**: does the text include typos or non standard expressions nullifying its comprehension? Do not consider "your/you're", "its/it's" since they are very frequent and do not affect text comprehension (Label = Ungrammatical)
  - **No context**: does the text contain (Label = No_context):
    - ∗ Reference to other users?
    - ∗ Links?
    - ∗ Anaphoric or deictic pronouns without an explicit referent?
    - ∗ Demonstrative pronouns?
  - **Not complete**: some parts are missing: is the text not complete? (Label

= Not_Complete)

- **Subjectivity**: does the text contain information that makes annotators' opinions interfere in their judgment? (Label = Subjectivity)
  - **Personal Bias**: does the text contain specific words that can be interpreted in a subjective way by the annotator (for example: "racist", "fascist", "clown", "pathetic", "liar", "pig") or refer to specific, critical opinions (no vax, wearing or not wearing masks)? (Label = Personal_Bias)
  - **Swearing**: does the text include swearing words (for example: "prick", "turd", "crap", "bullshit", "moron", "dumb")? Does it include ableist insults such as "retarded", "psycho" and expressions containing "shit" (for example: "cut the shit", "don't know shit"). Do not consider WTF, SMFH and similar acronyms containing "fuck". (Label = Swearing)
  - **Threatening**: does the text contain linguistic violence (for example: "shut up", "get out", "you are going to prison")? (Label = Threatening)

Analyzing Challenges in Neural Machine Translation for Software Localization

Sai Koneru[1], **Matthias Huck**[2], **Miriam Exel**[2], and **Jan Niehues**[1]
[1] Karlsruhe Institute of Technology
[2] SAP SE, Dietmar-Hopp-Allee 16, 69190 Walldorf, Germany
`{sai.koneru, jan.niehues}@kit.edu`
`{matthias.huck, miriam.exel}@sap.com`

Abstract

Advancements in Neural Machine Translation (NMT) greatly benefit the software localization industry by decreasing the post-editing time of human annotators. Although the volume of the software being localized is growing significantly, techniques for improving NMT for user interface (UI) texts are lacking. These UI texts have different properties than other collections of texts, presenting unique challenges for NMT. For example, they are often very short, causing them to be ambiguous and needing additional context (button, title text, a table item, etc.) for disambiguation. However, no such UI data sets are readily available with contextual information for NMT models to exploit. This work aims to provide a first step in improving UI translations and highlight its challenges. To achieve this, we provide a novel multilingual UI corpus collection ($\sim 1.3M$ for English $\leftrightarrow$ German) with a targeted test set and analyze the limitations of state-of-the-art methods on this challenging task. Specifically, we present a targeted test set for disambiguation from English to German to evaluate reliably and emphasize UI translation challenges. Furthermore, we evaluate several state-of-the-art NMT techniques from domain adaptation and document-level NMT on this challenging task. All the scripts to replicate the experiments and data sets are available here.[1,2]

1 Introduction

There is a rapid increase in access to technology for people from around the globe. For software to be used by everybody, it is essential to provide User Interface (UI) texts in their native languages for monolingual speakers. However, many applications are created in English and later localized to various languages. To decrease the translation time, localization companies take the help of machine translation (MT). Human annotators use the MT system for generating initial translations and post-edit the system's output to increase efficiency in computer-assisted translation tools (Flournoy and Duran, 2009; Skadiņš et al., 2014). When producing high-quality translations, costs can be saved by decreasing the annotator's time on editing and making the localization process cheaper. Thus, enabling more companies to localize in several languages at low cost.

Neural Machine Translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017) is the current state-of-the-art (SOTA) approach for generating the initial "draft" for the translations. Although conventional NMT models are sufficient in many cases, they use only the source sentence alone to predict the target translation (*Locality Assumption*) (Maruf et al., 2021). However, this is problematic when translating several types of concise UI texts that need more information.

It has been a decade since Muntés Mulero et al. (2012) have shown the need for integrating additional context into MT models for software localization and have pointed out several issues. For example, consider translating the English source word "Login" into German. Here, the translation depends on where the text is. If it is on a button, the correct translation is the verb form "Anmelden". In the case of a title text (FAQ, Documentation, etc.), the translation might rather be the noun form "Anmeldung". Depending on the application, the translation can also be simply "Login" even on a button.

Another common phenomenon to deal with when translating these short UI texts is semantic ambiguity. For example, if we want to translate the word "Home", we need to know the context where the sentence is present. Although the translation is often the German word "Haus", it is not correct in the context of Linux applications. In this case, we would need to translate "Home" as "Benutzerverzeichnis" ($HOME directory). Apart from the issues

---

[1] `https://github.com/saikoneru/NMT_Localization`
[2] We crawled this data only for scientific research.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2442–2454
May 2-6, 2023

mentioned above, gender (the gender of the item in a table depends on the gender of the table title text), and consistency in terminology and style (e.g formal v/s informal) are problematic when translating UI. However, sentence-level NMT models lack the contextual information to generate suitable target translations.

Several approaches were proposed to integrate contextual information into NMT models. Domain adaptation and Multi-Domain models using tags are common techniques in NMT to inform the models about the type of content it is translating (Kobus et al., 2017; Chu et al., 2017; Pham et al., 2020; Xu et al., 2020). Knowing the domain/topic of the text enables NMT to select and generate appropriate translations in a particular setting. Another recent and growing field that tries to make models context-aware is document-level NMT (Doc-NMT) (Tiedemann and Scherrer, 2017; Voita et al., 2018; Maruf et al., 2019, 2021; Bao et al., 2021; Sun et al., 2022). The surrounding source and/or target sentences usually can provide significant cues and hints for the NMT model to understand the domain and the context in which the source sentence is occurring. These approaches are promising and could benefit localizing UI segments, but the lack of sufficient annotated UI data with contextual information is the main hindrance to applying them.

Although localization files of software like GNOME, UBUNTU, and KDE are present in the OPUS corpus (Tiedemann, 2012), they do not provide the document structure or any other metadata. Also, the segments in the corpora do not come from a vast amount of domains. Neither large amounts of UI data are publicly available nor a targeted test set to measure the model's ability to use context for UI translations. These constraints limit the potential to improve NMT for software localization and are necessary as a first step.

This work addresses the limitations mentioned above by creating a corpus for UI data with document-level information extracted from multiple domains, a targeted test suite, and baselines using current SOTA methods. Our main contributions in this paper are the following

- We present a task of translating UI texts and show their unique properties requiring additional context. Furthermore, we provide a novel multilingual UI corpus covering multiple domains with contextual information to enable NMT for this task. (Section 2)
- To identify the limitations in current NMT systems, we propose a targeted evaluation framework with test sets replicating realistic conditions and solving disambiguation. (Section 3)
- We analyze domain adaptation and Doc-NMT techniques on our collected corpora and evaluation sets to highlight challenges in the proposed task. (Section 4)

2 Addressing Data Scarcity for UI

For improving NMT for UI segments, the first challenge to address is the lack of parallel data with contextual information. To achieve this, we present a novel UI corpus that we assemble from software localization files of publicly available repositories.

First, we briefly explain the structure and contents of the localization files which we search and collect to create our data set. Then, we highlight what types of contextual information are available in these files that are useful for NMT. Finally, we describe how we scraped large amounts of these PO files for multiple languages.

2.1 Portable Object for Localization

Portable Object (PO) files[3] are one of the standard file formats used in the localization industry. These are plain text-based files and do not need specialized tools for reading. Although this is not the only format used for localization, it is widely used in the GNU gettext[4] tool for free software's.

Figure 1 shows an example of a PO file along with a screenshot. A PO file consists of several entries, each containing a unique source and target pair. Each typical entry consists of the following items:

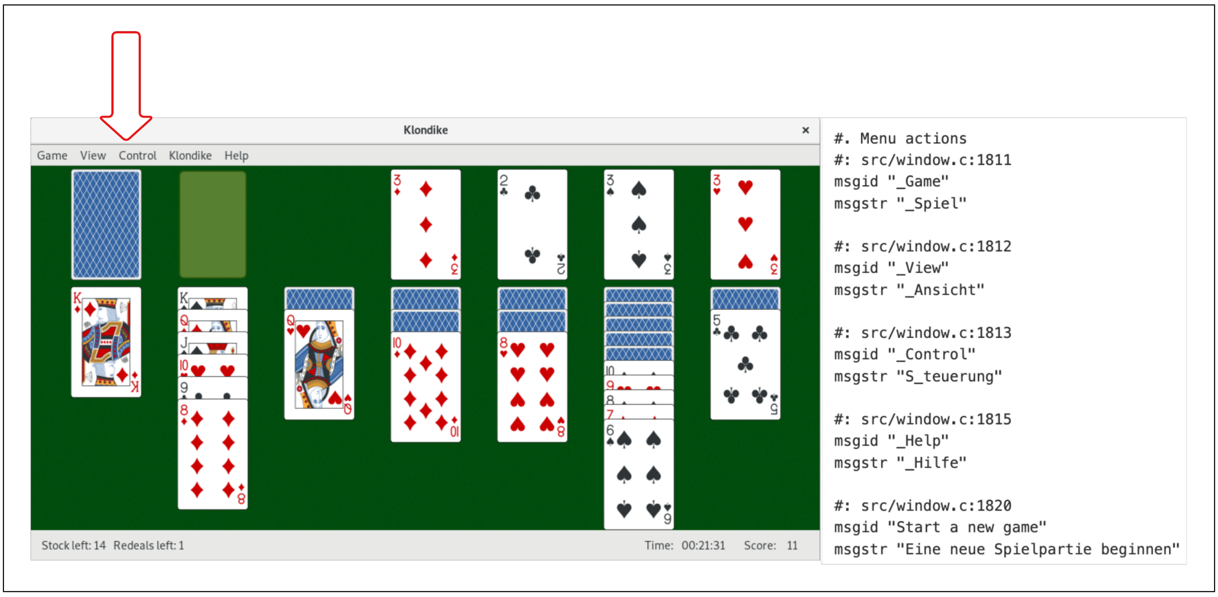
- **msgid:** The source string that needs to be translated and is usually in English for software applications (Game, View, Control, etc.).
- **msgstr:** The translation of the text present inside double quotes in the **msgid** entry (Spiel, Ansicht, Steuerung, etc.).

---

[3] For detailed information on PO files, please refer to this blog post. `http://pology.nedohodnik.net/doc/user/en_US/ch-poformat.html`

[4] `https://www.gnu.org/software/gettext/`

[5] Screenshot from `https://gitlab.gnome.org/GNOME/aisleriot/-/blob/master/po/de.po#L1726`. Accessed Date: 09/02/2023 (dd/mm/yyyy)



```
#. Menu actions
#: src/window.c:1811
msgid "_Game"
msgstr "_Spiel"

#: src/window.c:1812
msgid "_View"
msgstr "_Ansicht"

#: src/window.c:1813
msgid "_Control"
msgstr "S_teuerung"

#: src/window.c:1815
msgid "_Help"
msgstr "_Hilfe"

#: src/window.c:1820
msgid "Start a new game"
msgstr "Eine neue Spielpartie beginnen"
```

Figure 1: Screenshot from Klondike[5](Solitaire) game in GNOME with UI texts (LEFT). The German translations of the texts in the menu bar (RED ARROW) with their corresponding translations in the PO file (RIGHT)

- **Source Reference[6]:** Comment above each entry indicating where the texts are extracted from including the file name and line number (*#: src/window.c:1811*). Translators can look up the source code for more context. Note that this can be more than one file and can include multiple references.

2.2 Contextual Information in PO

PO files contain more information than the source and target sentence pairs. However, it is not straightforward how and what information to include in the NMT models. Moreover, it is unlike the contextual information in the traditional sense to address phenomena like co-reference resolution or gender (Stojanovski and Fraser, 2019; Wong et al., 2020; Lopes et al., 2020). Below, we describe three sources of information that are relevant and useful for UI translation:

Domain: All segments in a PO file belong to one application or software. Hence, we can loosely classify these segments belonging to finer-grained domains containing specific properties. Furthermore, new UI texts added to the software should maintain consistency according to terms and phrasing with previous translations. Therefore, knowing that the sentence is from a particular PO file can help NMT choose a similar translation style.

Neighbours: The order of the entries in the PO file maintains some relation to what users see when using the software. Although there is no semantic relation between the entries in the document, surrounding segments can provide information about the current entry. For example, consider a scenario where we must translate the word "Home" into German. If we know that the surrounding two sentences are "Change Directory" and "Print Working Directory," we can infer that it is in the context of a Linux application. Hence, we can translate it as "Benutzerverzeichnis" by inferring information from other entries/neighbors.

Source Reference: Specific to PO files, source reference can provide high-quality information. Almost all entries have a reference to the source code from which it was extracted. Human translators often look at the source code in case of ambiguity. NMT models can also exploit such information while predicting. For translating "Login" to German, we need to decide on choosing the noun (title text) or verb form (button). If all the texts having the same source reference were translated as the verb form, it is more likely that the current word is also a verb form. Therefore, source references can also be beneficial to integrate into NMT models.

Developers can also add context to each entry in the form of comments or by adding a **msgcxt** in the entry. We illustrate such occurrences with the toy

---

[6] **Source Reference** is the file path in the source code for a source-target sentence pair. We follow the convention and clarify that it is unrelated to the source or reference texts.

example from the blog post[3]:

```
#. TRANSLATORS: First letter in 'Scope'
#: tools/observinglist.cpp:700
msgid "S"
msgstr ""

#: tools/observinglist.cpp:700
msgctxt "First letter in 'Scope'"
msgid "S"
msgstr ""
```

Note that there is no standard or consistent process on how developers provide context to the translators. For example, it can be the case that a comment for an entry also applies to the subsequent entries. However, it is trivial for humans to determine which entries are related to the comment. Also, some entries do not have any comments or *msgcxt*. Furthermore, this information is in natural language needing complex models to use this information. Therefore, it is not straightforward to leverage the comments/*msgcxt* as a context in an NMT model, and we do not include this information while conducting our baselines in Section 4.

One possible workaround is to use the actual source code using the source reference entry to provide more information. Extracting contextual information from code can save developers time annotating and enable the NMT model to translate a wide range of applications.

Another category of contextual metadata that could be beneficial for translating UIs with NMT models is the element type of the text (button, title, table item, etc.,). However, this is rarely present in open-source data, and developers do not explicitly mention such information in PO files usually. Moreover, there is also no consistency in this case to specify such information.

2.3 Multilingual UI Corpus

Although we have shown that PO files are useful, there is no large corpus containing these files. Having UI texts from a limited amount of applications does not cover different domain-specific choices or styles. Collecting UI data from as many different domains as possible is crucial. Therefore, we create the UI corpus by searching for PO files in public GitHub repositories.

By convention, developers name the PO file with the language code and *.po* extension. For example, the name of PO files in German is **de.po | De.po | DE.po | dE.po**. Therefore, we search and download files ending with such names depending on the language in publicly available repositories. We use Sourcegraph to query the repositories and download PO files for multiple languages. Afterward, we parse the PO files using the polib library and extract translation pairs with the contextual information described in Section 2.2. Table 1 shows the number of source-target pairs we extracted for different languages.

Language	Repositories	Total Sentence Pairs
German	22248	1.33M
Japanese	13196	0.89M
Spanish	20996	1.38M
Hindi	3095	128K

Table 1: Total number of repositories and parallel sentence pairs extracted between English ↔ German, Japanese, Spanish or Hindi. Note that PO files for many more languages are often present and can be scraped. We provide links along with commit hashes (to replicate our data set) to download the corpus from public Github repositories.

Although we extract bilingual translation data, it is possible to create a multi-way UI data set using repositories containing PO files for multiple languages. Therefore, such data can also be fruitful in improving Multilingual NMT for non-English translation directions (Aharoni et al., 2019; Liu et al., 2020).

3 Realistic & Targeted Evaluation

The previous section described how we collected the data and its characteristics. The next step is to design a realistic evaluation framework to showcase challenges in UI translation and estimate the NMT model's quality reliably.

In this section, first, we propose two test sets considering different use cases while evaluating. Then, to highlight one fundamental phenomenon when dealing with UI strings, we create a challenge set for disambiguation based on heuristics and human annotation. We explain how we curated the different test sets below.

3.1 Intra & Cross-application Test Sets

Splitting the data appropriately into train/dev/test sets is vital in evaluating NMT models. Furthermore, the splits should also capture realistic conditions. Therefore, we propose the following scenarios:

Intra-application: The software gets updated regularly. With the updates, we need to translate newly added and modified strings. However, we have access to human-annotated translations for the strings in the previous version providing accurate information about the domain. Therefore, we consider a case where the test set is from applications that the model has seen during training. From the training data, we randomly select samples and discard them from our training set.

Cross-application: Localization companies also need to translate new applications for which there is no gold human-annotated data. To imitate these scenarios, we keep out a few applications entirely for evaluation. Instead of randomly selecting for evaluation, we choose 10 applications containing between 100 and 300 translation pairs. We set these bounds to maximizing the number of domains covered during testing.

Both test sets also provide insights into how much contextual information is consistent across applications. For example, suppose the context-aware NMT system performs better than sentence-level NMT on the Intra and not the Cross-application test set. In that case, we can conclude that such information does not generalize for new applications. However, it can be used only to improve translations for applications already present in the training data.

3.2 Targeted Disambiguation Test Set

The test sets described above assess the model's ability in different scenarios. However, it does not explicitly measure the model's ability to capture context. The need for targeted test sets (Bawden et al., 2018; Voita et al., 2018; Stojanovski et al., 2020) to use context was shown in Doc-NMT. Several complicated architectures were proposed for integrating context using limited amounts of document-level annotated data and showed improvements on standard test sets. However, proper regularization parameters on the sentence-level model led to the same performance improvements (Li et al., 2020; Sun et al., 2022). Hence, it is necessary to evaluate context-aware NMT models on targeted test sets to not draw false conclusions.

In the case of UI texts, a prevalent phenomenon where context is needed is disambiguation. Long sentences often contain enough context for the models to generate the translation correctly. However, a large number of short sentences are present in UI data (refer to Appendix A.3). Therefore, we create a targeted disambiguation test set when translating English to German as a benchmark to evaluate context-aware NMT models.

We use heuristics to extract and filter source sentences with multiple target translations. Then, we manually evaluate these instances and filter those where the context is insufficient, or it is unnecessary as the translations are paraphrases. We explain the process in the following.

3.2.1 Automatic Filtering

After lower-casing the data, we extract source sentences with multiple target translations and ended up with numerous pairs ($\sim 50k$) consisting of many paraphrases. Therefore, to only extract disambiguation pairs, we perform the following steps:

1. We only keep word-word translations. This step was necessary to filter out paraphrases using a database later.
2. Translations from the same PO files are more likely to be paraphrased as they belong to the same domain. Hence, we only keep where target translation occurs across different files.
3. The source-target pairs that occur only once were filtered out as they might be noise from incorrect translations.
4. We discard pairs consisting of characters such as "#,,_,!" (IP-address v/s IPaddress) by checking for punctuation symbols.
5. We match with the German paraphrase database[7] (Ganitkevitch et al., 2013; Ganitkevitch and Callison-Burch, 2014) to eliminate target translations that are synonyms. We only keep target words that occur in the database but are not considered paraphrases. We do not set a threshold score and consider all the entries in the database to have high precision.

After several steps of filtering, we end up with 293 segment of pairs. Upon manual inspection, we find it impossible to disambiguate several instances, even when we look at the PO files for

---

[7] Note that using the multilingual paraphrase database (Ganitkevitch and Callison-Burch, 2014), it is possible to extend this approach into several languages

Context Source	Context Target	Source Word	Target Word
%s was written onto %s Select Image	%s wurde auf %s geschrieben Abbild auswählen	Image	**Abbild**, Bilder
Journal Author Note	Zeitschrift Autorhinweise	Volume	**Volumen**, Band
Vendor Locations Variant Count	Lagerorte der Lieferanten Variantenanzahl	Volume	Volumen, **Band**
You are not currently subscribed to any active threads There are no active author subscriptions.	Sie haben momentan keine aktiven Diskussionen abonniert Es gibt keine aktiven Abonnements nach Autor.	Thread	**Diskussion**, Faden

Table 2: Examples from the targeted disambiguation test set. We only show 2 surrounding entries in the table but the human annotators were shown 5 to provide more information. The actual translation is highlighted in **bold**.

context. In some cases, it is not clear if the translation of the text is a noun or verb, even by looking at the whole file. Although it is possible that looking at the source code may prove beneficial, we ignore these pairs. Moreover, words like "Settings" and "Preferences" are not paraphrased according to the database but are usually synonymous in the UI context. To eliminate such occurrences, we perform a final step of manual filtering and clean the data using native speakers.

3.2.2 Manual Annotation

We split the test set into three parts. Then, we send each part to two annotators for the final filtering. We gave the annotators the surrounding 5 translation pairs (context) and the source word. Given this information, they were given multiple target translations as options, and we asked them to select the appropriate target given the source and context. Furthermore, we allow them to choose multiple options if they believe that more than one translation is appropriate. Moreover, we also provide a "None of the above" option for every question if they think the context is insufficient or the translations are incorrect. Finally, we selected the ones where both annotators chose the same target word as translation. After annotation, we had 95 entries in the final disambiguation test set. We present a few examples from the test set in Table 2.

Note that many pairs were discarded due to insufficient context but it might be possible to disambiguate with screenshots of application showing the specific UI element. However, we do not have text-image aligned data but it shows the potential advantage of Multi-Modal NMT (Elliott et al., 2017). An overview of the sizes of the data splits for our English ↔ German experiments can be found in Appendix Table 7.

4 Baselines

To evaluate the current NMT approaches with contextual information on UI translations, we provide baselines (English → German) using the current SOTA techniques. We build context-aware NMT models on our UI corpus and measure their ability to use context and highlight their shortcomings.

First, we describe how we integrate multiple types of contextual information in NMT. Then, we present the results and analyze the performance of the models on the multiple test sets.

4.1 Context-aware NMT models

As shown in Section 2.2, we have access to different sources of contextual information. The next step is to integrate them into the NMT model. We perform experiments by including them in the following ways:

Domain: We want to exploit the fact that we have already seen a particular PO file. Therefore, we take the help of tags to indicate this information similar to Kobus et al. (2017). We assign a unique domain tag to each repository and prepend it to each source sentence in a file. During training, we initialize each tag with a random embedding. While testing, the model can know the origin of the file by looking at the domain tag to translate appropriately. If it is from an unseen repository, we assign a random embedding

Example: *<KLONDIKE> _Game* ||| *_Spiel*

Neighbours: In this scenario, we consider the whole PO file as a document. Knowing the surrounding entries in the document can provide cues about the current segment. Therefore, we exploit them by following approaches in Doc-NMT (Tiedemann and Scherrer, 2017). We concatenate the source with the surrounding (left and right if possible) two segments in the file. Note that we only add the source context and not the target side.

Example: *_View _Control <TAG> _Game* ||| *_Spiel*

Source Reference: Specific to PO files, file paths of the string in the source code can capture standard naming conventions across different applications. Like the neighbor's approach above, we concatenate the path with the source sentence. However, we remove the punctuation to maximize the overlap across paths having similar naming structures.

Example: *src window c <TAG> _Game* ||| *_Spiel*

4.2 Experimental Setup

Note that we initially pre-train the models on WMT 14 English-German data (Luong et al., 2015) and then fine-tune on our corpus. We perform this two-step training following the success in transfer learning (Zoph et al., 2016; Kocmi and Bojar, 2018) and improve the translation capabilities. The pre-processing and training parameters for both stages is described in Section A.1.

4.3 Results

We present results on the different evaluation scenarios described in Section 3. First, to understand the performance of context-aware models in realistic scenarios, we report the results on *Intra* and *Cross* test sets in Table 3. Then, we evaluate the models on the targeted disambiguation test set to highlight the need for context in this challenging task and present scores in Table 4. Instead of BLUE (Papineni et al., 2002) or CharacTER (Wang et al., 2016), we report the accuracy for this test set as both source and target are word-translations. The scores for $De \rightarrow En$ demonstrate the ability to map different German words to the same English word. However, it is not completely accurate as the English translations that the model generates can be paraphrased.

Do context-aware NMT models perform better where training data is from known applications? All the context-aware models in Table 4 obtain higher score than the sentence-level model on the *Intra-application* test set. The only exception being the *Neighbour* approach in English → German on BLEU (49.21 ± 0.26) but par with the standard NMT baseline (49.03 ± 0.76).

We hypothesize that the model knows how to use the context when translating segments from the *Intra-applicaton* test set. For example, consider the *Source Reference* approach and the file in Figure 1. By knowing that the text "*Control*" present in "*src/window.c*" was translated as a noun ("*Steuerung*") and not a verb ("*Steuern*"), the model knows how to translate ambiguous texts in that file. Such phenomena are also possible when using the other context-aware NMT approaches. Therefore, we need to integrate contextual information when translating new segments from seen applications.

Does the contextual information generalize to entirely new software? Contrary to the gains observed above, all the context-aware models do not obtain significant improvements on the *Cross* test set. Although there is a slight improvement in a few conditions, it is around 0.5 in BLEU and 0.4% in CharacTER. Moreover, in the case of *Domain Tag* approach, all the texts are assigned a new random domain tag that was not seen during training. Hence, it performs significantly worse than the other methods.

The results show that the contextual information does not generalize well to entirely new applications. Therefore,

we need better context-aware NMT approaches than the baselines we proposed, which capture context more abstractly and benefits software localization in realistic conditions.

How well do the context-aware NMT models perform on the challenging targeted disambiguation test set? While the sentence-level NMT model obtains an accuracy of 20.0% on the targeted test set, the *Domain Tag* approach reaches 41.0%. Furthermore, it performs the best out of all other approaches, with the next best model reaching only 29.4%.

However, it can be the case that most segments in the disambiguation test belong to applications present in the training data. Hence, allowing the *Domain Tag* approach to perform better. Nevertheless, if the example is from a cross-application, this approach can perform worse due to the random initialization of the tag. Moreover, the other approaches are better than the sentence-level model by only a maximum of 10%. The proposed baseline methods are insufficient in this challenging test set and call for better context-aware NMT models.

Are the context-aware NMT models ignoring the context? We also need to investigate the influence of context, if at all, on the model's translations. If the system uses the context, providing false/incorrect context to the model should hurt the performance. Otherwise, we can conclude that the current NMT model ignores the contextual in-

Context Type	*Evaluation Setup*							
	BLEU (↑)				*CharacTER* (↓ %)			
	Cross		Intra		Cross		Intra	
	En → De	De → En	En → De	De → En	En → De	De → En	En → De	De → En
No context	47.15 ± 0.12	57.14 ± 0.44	49.03 ± 0.76	56.05 ± 0.29	37.04 ± 0.08	29.51 ± 0.02	34.75 ± 0.33	29.18 ± 0.32
Domain Tag	44.99 ± 0.26	54.59 ± 0.36	**52.10 ± 0.06**	**57.90 ± 0.45**	39.08 ± 0.07	30.55 ± 0.18	**31.64 ± 0.17**	**26.94 ± 0.27**
Neighbour (2)	46.93 ± 0.06	57.30 ± 0.27	49.21 ± 0.26	**56.92 ± 0.38**	36.94 ± 0.12	**29.13 ± 0.16**	**34.04 ± 0.21**	**28.22 ± 0.11**
Source Reference	**47.79 ± 0.09**	57.82 ± 0.33	**51.09 ± 0.33**	**58.53 ± 0.35**	**36.67 ± 0.14**	**29.12 ± 0.04**	**32.38 ± 0.41**	**27.00 ± 0.20**

Table 3: Baseline experiments using different sources of contextual information described in Section 2.2 and evaluated on both Intra and Cross application scenarios. We perform each experiment 3 times to account for randomness and report using both BLEU (↑) and CharacTER % (↓) metrics. We highlight the score in **bold** if the gains are statistically significant compared to the baseline model (*No context*)

	Accuracy (%)	
Context Type	En → De	De → En
None	21.05 ± 0.85	43.85 ± 0.49
Domain Tag	**41.03 ± 0.85**	**56.46 ± 2.14**
Neighbour (2)	26.26 ± 1.5	45.93 ± 1.79
Source Reference	30.8 ± 0.98	52.23 ± 1.79

Table 4: Accuracy of context-aware NMT models on targeted disambiguation test set ($En \leftrightarrow De$). The scores from $De \leftrightarrow En$ denote the accuracy in predicting the same English word given different German words. We run the experiment 3 times and report the confidence intervals.

formation as shown in previous works (Sun et al., 2022).

For this purpose, we conduct an experiment using the "*Neighbors*" approach and provide incorrect context by randomly sampling consecutive segments from different PO files. We report the results in Table 5. Although it is not better than a simple NMT system trained without context, there is always a drop in performance when using incorrect context. Thus, the model uses the context for generating translations but with degradation in general translation quality.

To show the role of context, we provide an example in our test sets in Table 6. The word "*driver*" in this case can be translated to German as either "*Fahrer*" (vehicle driver) or "*Treiber*" (software driver), depending on the domain. In both cases, the model correctly predicts the translation using the context. However, the scores do not show overall improvement, indicating that the context might add additional noise, causing a drop in general translation quality.

5 Related Work

Domain-Adaption in NMT: NMT models for software localization have to deal with texts coming from vast amount of domains. Hence, Domain-Adaptation techniques (Saunders, 2022) are highly relevant in this challenging task. Many works use different domain-tagging schemes to indicate the type of data where the segment's labels are either known or unknown (Kobus et al., 2017; Poncelas et al., 2018; Mino et al., 2020; Wang et al., 2021). Another line of approach is to modify the network to indicate and generate domain-specific translations (Zeng et al., 2018; Pham et al., 2021; Lin et al., 2021). Building on this, few works also propose to add and adapt using domain-specific parameters (Bapna and Firat, 2019; Abdul-Rauf et al., 2020).

Document-level NMT: Exploiting the context from the surrounding texts is necessary for both software localization and Doc-NMT. The initial straightforward approaches for Doc-NMT explored concatenating the source/target with surrounding sentences (Tiedemann and Scherrer, 2017; Agrawal et al., 2018; Ma et al., 2020). These methods do not change the NMT architecture or training methods but can be applied by simply altering the data or the embedding. In contrast, several works investigated creating a different representation for the context sentences using additional encoders (Jean et al., 2017; Zhang et al., 2018; Voita et al., 2018; Werlen et al., 2018; Wang et al., 2019a). Few recent works also propose either modifying the architecture of augmenting data to show the benefit of Document-Document translation (Bao et al., 2021; Sun et al., 2022). For targeted evaluation, Gonzales et al. (2017) also manually constructs a word-sense disambiguation test in English ↔ German. However, our test set is mostly automatic and calls for

Evaluation Setup	True Context		False Context		No Context	
	En → De	De → En	En → De	De → En	En → De	De → En
Intra	49.21 ± 0.26	56.92 ± 0.38	47.95 ± 0.28	55.0 ± 0.35	49.03 ± 0.76	56.05 ± 0.29
Cross	46.93 ± 0.06	57.30 ± 0.27	46.38 ± 0.22	55.94 ± 0.08	47.15 ± 0.12	57.14 ± 0.44

Table 5: BLEU scores for the "*Neighbor (2)*" model evaluated using True and False context during inference. For "*No Context*", we report the scores of the baseline model trained simply on the parallel data without any additional information. We report the confidence interval by running the experiment 3 times.

Source	Current Driver
Reference	Aktueller Fahrer
Source + True Context	Create a new vehicle status <tag> Date when the vehicle has been immatriculated <tag> Current Driver
Neighbour (2) Hypothesis	Aktueller Fahrer
Source + False Context	% s updates a cartridge <tag> Unknown file <tag> Current Driver
Neighbour (2) Hypothesis	Aktueller Treiber

Table 6: Example from our test set where the Neighbour approach (Table 3) uses the context and generates the proper translation. *Source + True context* is the concatenation of the true neighbors in the same PO file with the source sentence whereas *Source + False Context* appends a random context from another PO file.

solving the disambiguation by focusing on neighboring entries and not the sentence itself.

NMT for UI: One of the major limitations for building NMT for UI is the lack of data. The OPUS corpus (Tiedemann, 2012) makes such data available but with no contextual information. The closest to our work is Wang et al. (2019b), building NMT for mobile applications. However, the data is neither public nor addresses the several contextual issues occurring in software localization.

6 Conclusion

We presented a multilingual UI corpus with additional meta-information for researchers in the community to exploit and build context-aware NMT models for software localization. We also have proposed two evaluation setups to replicate conditions occurring in localization companies and show the difficulty in tackling new applications. Furthermore, we experiment using domain adaptation and Doc-NMT techniques to provide baselines and present the benefit of using different types of context in intra-applications while showing its ineffectiveness in cross-application scenarios. Moreover, we suggest an automatic procedure to create a targeted disambiguation test set where context is necessary to generate the correct target word translation. Finally, we show that the baseline systems fail in such challenging settings and call for sophisticated context-aware NMT models to improve the process of software localization.

7 Limitations

While we have presented UI data with contextual information, it does not contain any meta-data (button, table title, etc.,) of the textual elements. Having access to such resources can prove highly beneficial and are lacking in our presented data set. Furthermore, we do not provide any visual context to develop Multi-Modal NMT systems that implicitly contain the meta-data. However, we consider them as potential directions and include them in our future work. Finally, we do not analyze or experiment on non-European languages that might contain specific and unique properties which need to be addressed during localization. Another important limitation is the quality of the translations. We don't know whether they have been produced by professional translators or by multilingual speakers creating inaccurate translations. Although we filter the fuzzy (e.g MT outputs) translations, it can be that not all of them are marked completely. Therefore, qualitative analysis of the data is further necessary.

Acknowledgement We thank Christian Lieske and Jens Scharnbacher for providing insights into the localization process. We also thank Fabian Retkowski, Christian Huber, and Leonard Bärmann for annotating and making it possible to create the disambiguation test set.

References

Sadaf Abdul-Rauf, José Carlos Rosales, Minh Quang Pham, and François Yvon. 2020. Limsi@ wmt 2020. In *Conference on Machine Translation*.

Ruchit Rajeshkumar Agrawal, Marco Turchi, and Matteo Negri. 2018. Contextual handling in neural machine translation: Look behind, ahead and on both sides. In *21st annual conference of the European association for machine translation*, pages 11–20.

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Guangsheng Bao, Yue Zhang, Zhiyang Teng, Boxing Chen, and Weihua Luo. 2021. G-transformer for document-level machine translation. In *ACL/IJCNLP (1)*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1304–1313.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of domain adaptation methods for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–391, Vancouver, Canada. Association for Computational Linguistics.

Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. 2017. Findings of the second shared task on multimodal machine translation and multilingual image description. In *Proceedings of the Second Conference on Machine Translation*, pages 215–233.

Raymond Flournoy and Christine Duran. 2009. Machine translation and document localization at adobe: From pilot to production. *MT Summit XII: proceedings of the twelfth Machine Translation Summit*, pages 425–428.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The multilingual paraphrase database. In *LREC*, pages 4276–4283. Citeseer.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Annette Rios Gonzales, Laura Mascarell, and Rico Sennrich. 2017. Improving word sense disambiguation in neural machine translation with sense embeddings. In *Proceedings of the Second Conference on Machine Translation*, pages 11–19.

Sebastien Jean, Stanislas Lauly, Orhan Firat, and Kyunghyun Cho. 2017. Does neural machine translation benefit from larger context? *arXiv preprint arXiv:1704.05135*.

Catherine Kobus, Josep M Crego, and Jean Senellart. 2017. Domain control for neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 372–378.

Tom Kocmi and Ondřej Bojar. 2018. Trivial transfer learning for low-resource neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 244–252.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Bei Li, Hui Liu, Ziyang Wang, Yufan Jiang, Tong Xiao, Jingbo Zhu, Tongran Liu, and Changliang Li. 2020. Does multi-encoder help? a case study on context-aware neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3512–3518.

Huan Lin, Liang Yao, Baosong Yang, Dayiheng Liu, Haibo Zhang, Weihua Luo, Degen Huang, and Jinsong Su. 2021. Towards user-driven neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4008–4018.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

António Lopes, M Amin Farajian, Rachel Bawden, Michael Zhang, and André FT Martins. 2020. Document-level neural mt: A systematic comparison. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 225–234.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020. A simple and effective unified encoder for document-level machine translation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 3505–3511.

Sameen Maruf, André FT Martins, and Gholamreza Haffari. 2019. Selective attention for context-aware neural machine translation. In *Proceedings of NAACL-HLT*, pages 3092–3102.

Sameen Maruf, Fahimeh Saleh, and Gholamreza Haffari. 2021. A survey on document-level neural machine translation: Methods and evaluation. *ACM Computing Surveys (CSUR)*, 54(2):1–36.

Hideya Mino, Hideki Tanaka, Hitoshi Ito, Isao Goto, Ichiro Yamada, and Takenobu Tokunaga. 2020. Content-equivalent translated parallel news corpus and extension of domain adaptation for nmt. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3616–3622.

Víctor Muntés Mulero, Patricia Paladini Adell, Cristina España Bonet, and Lluís Màrquez Villodre. 2012. Context-aware machine translation for software localization. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation: EAMT 2012: Trento, Italy, May 28th-30th 2012*, pages 77–80.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Minh Quang Pham, Jitao Xu, Josep M Crego, François Yvon, and Jean Senellart. 2020. Priming neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 516–527.

MinhQuang Pham, Josep Maria Crego, and François Yvon. 2021. Revisiting multi-domain machine translation. *Transactions of the Association for Computational Linguistics*, 9:17–35.

Alberto Poncelas, Dimitar Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. 2018. Investigating backtranslation in neural machine translation. In *Proceedings of the 21st Annual Conference of the European Association for Machine Translation: 28-30 May 2018, Universitat d'Alacant, Alacant, Spain*, pages 249–258. European Association for Machine Translation.

Danielle Saunders. 2022. Domain adaptation and multi-domain adaptation for neural machine translation: A survey. *Journal of Artificial Intelligence Research*, 75:351–424.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Raivis Skadiņš, Mārcis Pinnis, Andrejs Vasiļjevs, Inguna Skadiņa, and Tomáš Hudík. 2014. Application of machine translation in localization into low-resourced languages. In *Proceedings of the 17th Annual conference of the European Association for Machine Translation*, pages 209–216.

Dario Stojanovski and Alexander Fraser. 2019. Improving anaphora resolution in neural machine translation using curriculum learning. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 140–150.

Dario Stojanovski, Benno Krojer, Denis Peskov, and Alexander Fraser. 2020. Contracat: Contrastive coreference analytical templates for machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4732–4749.

Zewei Sun, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Shujian Huang, Jiajun Chen, and Lei Li. 2022. Rethinking document-level neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3537–3548.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pages 2214–2218. Citeseer.

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*. The Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

E Voita, P Serdyukov, I Titov, and R Sennrich. 2018. Context-aware neural machine translation learns anaphora resolution. In *ACL 2018-56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, pages 1264–1274.

Ke Wang, Guandan Chen, Zhongqiang Huang, Xiaojun Wan, and Fei Huang. 2021. Bridging the domain gap: Improve informal language translation via counterfactual domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13970–13978.

Weiyue Wang, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. 2016. Character: Translation edit rate on character level. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 505–510.

Xinyi Wang, Jason Weston, Michael Auli, and Yacine Jernite. 2019a. Improving conditioning in context-aware sequence to sequence models. *arXiv preprint arXiv:1911.09728*.

Xu Wang, Chunyang Chen, and Zhenchang Xing. 2019b. Domain-specific machine translation with recurrent neural network for software localization. *Empirical Software Engineering*, 24(6):3514–3545.

Lesly Miculicich Werlen, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *EMNLP*.

KayYen Wong, Sameen Maruf, and Gholamreza Haffari. 2020. Contextual neural machine translation improves translation of cataphoric pronouns. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5971–5978.

Jitao Xu, Josep M Crego, and Jean Senellart. 2020. Boosting neural machine translation with similar translations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1580–1590.

Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin, and Jianqiang Zhao. 2018. Multi-domain neural machine translation with word-level domain context discrimination. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 447–457.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018. Improving the transformer translation model with document-level context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575.

A Appendix

A.1 Pre-processing and Training Parameters

For pre-processing, we first tokenize the data with the Moses Tokenizer (Koehn et al., 2007). Then, we learn a joint sub-word vocabulary from $30k$ merge operations using BPE (Sennrich et al., 2016) and apply on the data.

For experiments in Table 4, we first pre-train the model using WMT14 English-German data (Luong et al., 2015). We use the standard Transformer model (Vaswani et al., 2017) with 6 encoder and decoder layers. We use 4 attention heads for every layer an embedding dimension of 512. We use 0.1 for label-smoothing 0.2 for dropout. We set max-tokens to 3000 and a learning rate of 0.0001. For the experiments, we use the Fairseq toolkit (Ott et al., 2019) and set all other parameters to default. While fine-tuning, we reload the model and continue using the same optimization and regularization parameters.

A.2 Data Split Overview

Data Split	*Number of Sentence Pairs*
Train	1.26M
Intra	2.6K (2.6K)
Cross	2.6K (2.4K)
Disambiguation	95

Table 7: Overview of UI data split for English $\leftrightarrow$ German. () indicates the number of validation sentences for the Intra and Cross application test sets.

A.3 Characteristricts of UI segments

	Average Sentence Length English	Average Sentence Length German	Vocabulary Size English	Vocabulary Size German
UI	36	43	167375	321602
News Domain	142	157	626914	1444840

Table 8: Comparing UI (Git scraped) v/s WMT 14 English-German Data. It can be seen the UI sentences are much shorter than the data available in the general domain. Also in both data types, we see that the German sentences consist of more unique words due its complex morphology

Bootstrapping Multilingual Semantic Parsers using Large Language Models

Abhijeet Awasthi[1*] **Nitish Gupta**[2] **Bidisha Samanta**[2]
Shachi Dave[2] **Sunita Sarawagi**[1] **Partha Talukdar**[2]

[1]Indian Institute of Technology Bombay, [2]Google Research India
`{awasthi,sunita}@cse.iitb.ac.in`
`{guptanitish,bidishasamanta,shachi,partha}@google.com`

Abstract

Despite cross-lingual generalization demonstrated by pre-trained multilingual models, the translate-train paradigm of transferring English datasets across multiple languages remains to be a key mechanism for training task-specific multilingual models. However, for many low-resource languages, the availability of a reliable translation service entails significant amounts of costly human-annotated translation pairs. Further, translation services may continue to be brittle due to domain mismatch between task-specific input text and general-purpose text used for training translation models. For multilingual semantic parsing, we demonstrate the effectiveness and flexibility offered by large language models (LLMs) for translating English datasets into several languages via few-shot prompting. Through extensive comparisons on two public datasets, MTOP and MASSIVE, spanning 50 languages and several domains, we show that our method of translating data using LLMs outperforms a strong translate-train baseline on 41 out of 50 languages. We study the key design choices that enable more effective multilingual data translation via prompted LLMs.

1 Introduction

Enabling language technologies across several languages is an important goal for serving a diverse range of users in an inclusive manner. Recent advances in large-scale self-supervised multilingual language models hold immense promise in bridging the quality gap that currently exists between English and many other low resource languages (Conneau et al., 2020; Brown et al., 2020; Xue et al., 2021). Even though multilingual models exhibit cross-lingual generalization, getting meaningful performance across several languages still requires significant amounts of task-specific labeled data.

We consider the problem of automatically synthesizing semantic parsing datasets across several languages. Semantic parsing (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Berant et al., 2013) is the task of mapping natural language text into an executable *logical-form*. For example, given a user instruction (x) : "`Wake me up by 5 am`", mapping it to the logical-form (y): `[IN:CREATE_ALARM [SL:DATE_TIME 5 am ]]`. Manual annotation of queries with their logical forms requires human expertise which makes data collection across multiple languages challenging.

A common approach to automatic multilingual dataset creation is translating existing English datasets into target languages. Prior methods utilize an off-the-shelf machine translation model for translating the English utterance into the target language $x_{\text{eng}} \rightarrow x_{\text{tgt}}$, followed by projecting language specific components in the English logical-form y_{eng} to obtain the logical-form y_{tgt} in the target language (Moradshahi et al., 2020, 2021; Xia and Monti, 2021; Nicosia et al., 2021; Gritta et al., 2022; Wang et al., 2022). The projection step is often learned independent of the translation service, resulting in poor generalization across languages.

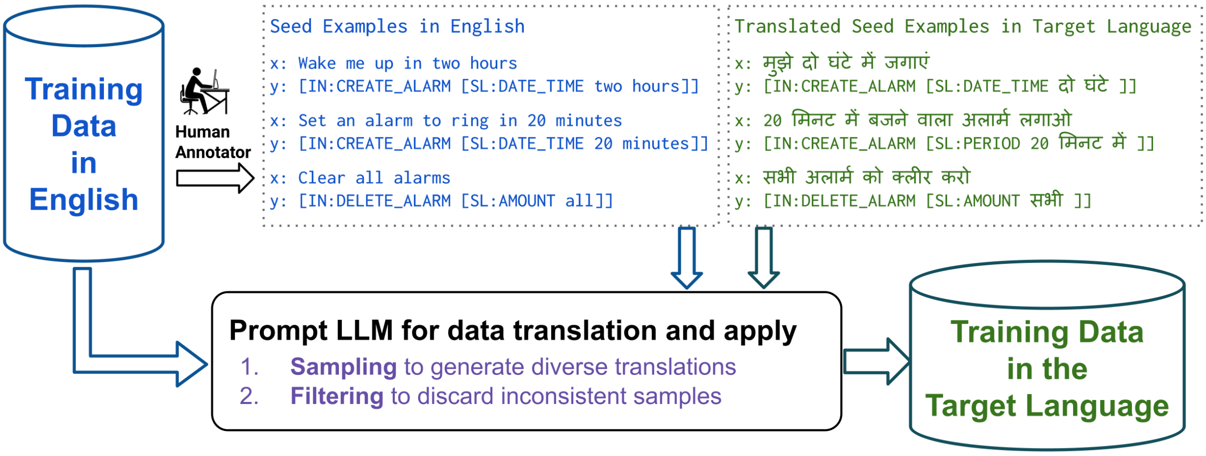
In this work we aim to utilize the few-shot generalization abilities exhibited by large language models (LLMs) (Brown et al., 2020; Chowdhery et al., 2022; Scao et al., 2022) for bootstrapping semantic parsing datasets across fifty languages. We propose a recipe of using LLMs to translate an English semantic parsing dataset containing `(utterance, logical-form)` pairs: $\mathcal{D}_{\text{eng}} = \{(x^i_{\text{eng}}, y^i_{\text{eng}})\}$ into a corresponding dataset in a target language: $\mathcal{D}_{\text{tgt}} = \{(x^i_{\text{tgt}}, y^i_{\text{tgt}})\}$. The generated dataset $\mathcal{D}_{\text{tgt}}$ is then used to train a semantic parser in the target language. Our method uses a small amount of manually translated semantic parsing examples to *teach* the LLM how to translate English examples in the target language via in-context learning (Min et al., 2022).

Figure 1 describes our data-translation pipeline which we refer to as LLM-T (§ 3). In contrast

---

[*] Work done during an internship at Google Research

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2455–2467
May 2-6, 2023



```
Seed Examples in English

x: Wake me up in two hours
y: [IN:CREATE_ALARM [SL:DATE_TIME two hours]]

x: Set an alarm to ring in 20 minutes
y: [IN:CREATE_ALARM [SL:DATE_TIME 20 minutes]]

x: Clear all alarms
y: [IN:DELETE_ALARM [SL:AMOUNT all]]
```

```
Translated Seed Examples in Target Language

x: मुझे दो घंटे में जगाएं
y: [IN:CREATE_ALARM [SL:DATE_TIME दो घंटे ]]

x: 20 मिनट में बजने वाला अलार्म लगाओ
y: [IN:CREATE_ALARM [SL:PERIOD 20 मिनट में ]]

x: सभी अलार्म को क्लीर करो
y: [IN:DELETE_ALARM [SL:AMOUNT सभी ]]
```

Figure 1: **Proposed semantic parsing data translation pipeline using LLMs** (§ 3): With the help of human translators, we first collect translations of a small seed set of English examples in the Target Language (e.g. Hindi; § 3.1). Given a new English example, a small subset from this initial seed set of examples with their respective translations is chosen to prompt the LLM (§ 3.2). The prompted LLM translates the given English example in the Target Language. We repeat this process for each example in the English training data to generate a training dataset in the Target Language. To ensure high-quality of the resulting dataset, we generate diverse translations via top-p (nucleus) sampling (§ 3.3) and apply consistency filtering (§ 3.4).

to prior translation based methods that involved a two-staged process requiring different modules, our method uses the LLM to *jointly translate* an English $(x_{\text{eng}}, y_{\text{eng}})$ pair directly into the target language $(x_{\text{tgt}}, y_{\text{tgt}})$. We identify two important choices that make the LLM translated data more effective for training a downstream parser: **(i) Sampling diverse translations** (§ 3.3): Decoding translations using top-p (Fan et al., 2018) and top-k (Holtzman et al., 2019) sampling leads to improved downstream performance compared to using greedy decoding. Sampling multiple diverse translations per example further improves the downstream performance; **(ii) Filtering inconsistent examples** (§ 3.4): Decoding via sampling can result in noisy joint translations of the (utterance, logical-form) pairs. To filter out the inconsistent pairs, we propose a slot-value match based filtering technique that improves the training data quality.

We perform experiments on two multilingual semantic parsing datasets: MTOP (Li et al., 2021) and MASSIVE (FitzGerald et al., 2022). On *4 out of 5* languages in MTOP and *41 out of 50* languages in MASSIVE, our method LLM-T outperforms TAF (Nicosia et al., 2021), a strong baseline that utilizes a supervised translation service (§ 5.1). Further, we see that LLM-T achieves 93% of the performance obtained by "fully-supervised" models that use 30× more manually translated examples (§ 5.2). We justify the importance of generating multiple translations using sampling, filtering out inconsistent examples, and using larger-sized LLMs in improving translated data quality (§ 5.3). Finally, we perform an error analysis of our parser and show the key sources of disagreements between the model predictions and the ground truth (§ 5.4).

2 Background

In this section, we provide an overview of semantic parsing and prior translation-based methods for creating multilingual semantic parsing datasets.

2.1 Semantic Parsing

Semantic parsing is the task of mapping text queries to their meaning representations or *logical forms* (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Berant et al., 2013). We focus on task-oriented semantic parsing (Gupta et al., 2018) where the user utterance needs to be parsed into a high-level intent specifying the overall goal, and fine-grained slots containing details about the utterance. The intents and slots come from a task-specific vocabulary. For example, given an utterance x: "`How is the rainfall today?`", the parser should generate the logical-form
y: `[IN:GET_WEATHER [SL:ATTRIBUTE rainfall] [SL:DATE today ] ]`

Here, `IN:GET_WEATHER` is the high-level intent, `SL:ATTRIBUTE` and `SL:DATE` are the slots that specify details about the intent. We refer to the logical-

form with its slot values removed as its "signature". For example, the signature of y is
`[IN:GET_WEATHER [SL:ATTRIBUTE][SL:DATE]]`

2.2 Translating Semantic Parsing Datasets

Given an English semantic parsing dataset containing (utterance, logical-form) pairs $\mathcal{D}_{\text{eng}} = \{(x^i_{\text{eng}}, y^i_{\text{eng}})\}$, many methods aim to translate $\mathcal{D}_{\text{eng}}$ to a dataset $\mathcal{D}_{\text{tgt}} = \{(x^i_{\text{tgt}}, y^i_{\text{tgt}})\}$ in the target language (tgt). Here x^i_{tgt} is the translation of x^i_{eng}, and y^i_{tgt} is the logical form grounded in the translated utterance x^i_{tgt}. Target logical form y^i_{tgt} has the same signature as y^i_{eng} and only differs in terms of the translated slot values. Most translation based approaches (Moradshahi et al., 2020, 2021; Xia and Monti, 2021; Nicosia et al., 2021) translate an English example $(x^i_{\text{eng}}, y^i_{\text{eng}})$ to the corresponding target language example $(x^i_{\text{tgt}}, y^i_{\text{tgt}})$ via a two step process: (i) **Translate**: Use a supervised translation service to convert the English utterance x^i_{eng} into the target language utterance x^i_{tgt}; and (ii) **Project**: Replace the English slot values in y^i_{eng} with spans copied from the translated utterance x^i_{tgt} via a learned alignment model. The translated examples are then used to train a downstream multilingual semantic parser. For example, Nicosia et al. (2021) implement the project step by training a filler module on English data to fill slot-values in a logical-form signature by copying spans from the utterance. During inference, the trained filler module is then used in a zero-shot manner to fill logical-form signatures with spans copied from the translated utterances.

3 Our Method: Prompting LLMs for Dataset Translation

Our goal is to learn a multilingual semantic parser capable of parsing user queries in many languages. Towards this goal, we propose a method for generating multilingual training datasets via few-shot prompting of an LLM to translate existing English datasets into several languages.

In contrast to prior approaches, we jointly perform example translation by prompting an LLM with a few exemplars of translating English $(x_{\text{eng}}, y_{\text{eng}})$ pairs to target language $(x_{\text{tgt}}, y_{\text{tgt}})$ pairs. Figure 1 describes our data-translation method which we refer to as LLM-T. With the help of human translators we first collect a small seed set of exemplar translations used for prompting the LLM (§ 3.1). Given an input English example, we dynamically construct the LLM prompt by identifying a relevant subset of seed exemplars (§ 3.2). The LLM translates the English example into the target language by in-context learning from the exemplars provided in the prompt. Instead of decoding the most likely translation, we generate multiple diverse translations (§ 3.3) using top-p (nucleus) sampling (Holtzman et al., 2019). While sampling improves the text diversity, it can lead to more noisy generations. We filter out the noisy generations using a simple string-match based technique before training a parser on the translated data (§ 3.4).

3.1 Selecting Seed Exemplars for Translation

Given an English semantic parsing dataset $\mathcal{D}_{\text{eng}} = \{(x^i_{\text{eng}}, y^i_{\text{eng}})\}$, we first want to identify a small seed set $\mathcal{S}_{\text{eng}} \subset \mathcal{D}_{\text{eng}}$ that will be translated into the target language ($\mathcal{S}_{\text{tgt}}$) with the help of human translators. The examples in $\mathcal{S}_{\text{eng}}$ and their corresponding translations in $\mathcal{S}_{\text{tgt}}$ will be used for prompting the LLM. Therefore, the choice of the seed examples in $\mathcal{S}_{\text{eng}}$ that are manually translated into $\mathcal{S}_{\text{tgt}}$ becomes important—we would like that the multiple domains (e.g. `Alarms, Music, News, Weather, etc.`) and the intents and slot types in each domain are covered. This ensures that for a given English example to be translated, we will be able to prompt the LLM in a manner such that at least one of the few-shot exemplars will share the intent and slots with the test English example. In practice, we select seed examples in a manner to cover all the intents and slots in a domain at least once. If the selected examples are less than 20 for a domain, we select the remaining examples randomly.

3.2 Constructing the Prompt using Translation Pairs in the Seed Sets

LLM inference is constrained by the maximum number of tokens in the input. Hence, we can only fit a limited number of examples to construct the LLM prompt. The choice of prompt examples and their ordering is known to significantly impact the quality of the generations (Kumar and Talukdar, 2021; Rubin et al., 2021; Lu et al., 2022). To improve the likelihood of correctly translating an English example $(x_{\text{eng}}, y_{\text{eng}})$, we retrieve seed examples $\{(x^s_{\text{eng}}, y^s_{\text{eng}}, x^s_{\text{tgt}}, y^s_{\text{tgt}})\}$ that share the same domain with y_{eng}. To bias the LLM further, we order the more relevant prompt examples closer to the input English example. Here, relevance between two examples is considered higher if they share the same intent. The remaining examples are

LLM INPUT:

Translate examples from English to Hindi

x^{1}_{eng} : Wake me up in two hours
y^{1}_{eng} : [IN:CREATE_ALARM [SL:DATE_TIME two hours]]
x^{1}_{tgt} : मुझे दो घंटे में जगाएं
y^{1}_{tgt} : [IN:CREATE_ALARM [SL:DATE_TIME दो घंटे]]

x^{2}_{eng} : Please set an alarm for 2 pm
y^{2}_{eng} : [IN:CREATE_ALARM [SL:DATE_TIME 2 pm]]
x^{2}_{tgt} : कृपया दोपहर दो बजे का अलार्म लगाएं
y^{2}_{tgt} : [IN:CREATE_ALARM [SL:DATE_TIME दो बजे]]

x_{eng} : Set the alarm for the flight next week
y_{eng} : [IN:CREATE_ALARM [SL:DATE_TIME next week]]

LLM OUTPUT:

x_{tgt} : फ्लाइट के लिए अगले सप्ताह अलार्म सेट करें
y_{tgt} : [IN:CREATE_ALARM [SL:DATE_TIME अगले सप्ताह]]

Figure 2: **Constructing the LLM Prompt** (§ 3.2): The input to the LLM contains a brief task description in the beginning followed by a series of English examples $(x^s_{\text{eng}}, y^s_{\text{eng}})$ and their translations in the target language $(x^s_{\text{tgt}}, y^s_{\text{tgt}})$ chosen from the seed sets $\mathcal{S}_{\text{eng}}$ and $\mathcal{S}_{\text{tgt}}$ respectively. Following the prompt examples, we append the new English example $(x_{\text{eng}}, y_{\text{eng}})$ to the input prompt which is fed to LLM. In the output, the LLM generates the translation for the new English example $(x_{\text{tgt}}, y_{\text{tgt}})$.

arbitrarily arranged to appear earlier in the prompt. Figure 2 shows an example translation—the *LLM input* contains two exemplars and then the English example that needs to be translated. The *LLM output* shows the translated output from the LLM.

3.3 Decoding Diverse Outputs from LLM

The text decoded from language models using the standard greedy decoding or beam search is often repetitive (Vijayakumar et al., 2016; Shao et al., 2017). To mimic how users express the same intentions in diverse ways, we experiment with the top-k and top-p sampling techniques (Fan et al., 2018; Holtzman et al., 2019) to decode multiple diverse translations per example. We expect sampling multiple translations to yield a better quality training dataset which in turn should result in better downstream semantic parsing performance compared to training on greedily decoded examples.

3.4 Data Filtering using Slot-Consistency

While the sampling techniques produce more diverse text, the sampled translations can be relatively noisy if they have lower likelihoods as per the model (Zhang et al., 2021). Thus, the translated pairs $(x_{\text{tgt}}, y_{\text{tgt}})$ in the LLM output can be inconsistent w.r.t. each other. For example, consider the LLM translated pair $(x^3_{\text{tgt}}, y^3_{\text{tgt}})$ shown in Figure 3. Here, y^3_{tgt} contains a slot value (in red) that does not appear in the corresponding utterance x^3_{tgt} making the pair $(x^3_{\text{tgt}}, y^3_{\text{tgt}})$ inconsistent. As per the task definition, for a given example (x, y), the slot-values in the logical form y should come from the spans of the utterance x. Thus, we filter out the translated examples $(x_{\text{tgt}}, y_{\text{tgt}})$ like these where the slot-values in y_{tgt} do not appear *exactly* as an exact sub-span in x_{tgt}. Figure 3 shows examples of slot-consistent and slot-inconsistent generations by an LLM through top-k sampling.

x_{eng} : Set the alarm for the flight next week
y_{eng} : [IN:CREATE_ALARM [SL:DATE_TIME next week]]

x^{1}_{tgt} : फ्लाइट के लिए अगले सप्ताह अलार्म सेट करें
y^{1}_{tgt} : [IN:CREATE_ALARM [SL:DATE_TIME अगले सप्ताह]]

x^{2}_{tgt} : अगले हफ्ते के लिए उड़ान के अलार्म को सेट करो
y^{2}_{tgt} : [IN:CREATE_ALARM [SL:DATE_TIME अगले हफ्ते के लिए]]

x^{3}_{tgt} : फ़्लाइट के लिए अलार्म सेट करें
y^{3}_{tgt} : [IN:CREATE_ALARM [SL:DATE_TIME अगले सप्ताह]]

x^{4}_{tgt} : अगली हफ्ते के उड़ान के लिए अलार्म सेट करें
y^{4}_{rgt} : [IN:CREATE_ALARM [SL:DATE_TIME अगले हफ्ते]]

Figure 3: **Slot Consistency Based Filtering** (§ 3.4): We present the input English example $(x_{\text{eng}}, y_{\text{eng}})$ and its four translated samples $\{(x^i_{\text{tgt}}, y^i_{\text{tgt}})\}$) the target language. The first two samples are *slot-consistent* as the slot-values (in green) in the logical forms appear exactly in the text utterances, while the last two samples are *slot-inconsistent* as the slot-values (in red) do not appear as an exact sub-string of the text utterance.

4 Experimental Set-up

We describe our experimental setup in this section.

Datasets We experiment on two public datasets — MTOP (Li et al., 2021) and MASSIVE (FitzGerald et al., 2022). MTOP contains examples from six languages: English, French, German, Hindi, Spanish, and Thai, spanning 11 domains covering 117 intents and 78 slot types. On average, MTOP contains 12.3K examples in the train split, 1.5K in the dev split, and 2.7K in the test split per language. MASSIVE contains examples from 51 typologically diverse languages including English spanning 18 domains covering 60 intents and 50 slot types. For each language, MASSIVE contains roughly 11.5K examples in the train split, 2K examples in the dev split and 3K examples in the test split.

Evaluation Metric Prior work (Li et al., 2021; Nicosia et al., 2021) uses Exact Match (EM) accuracy as a primary metric which compares predicted and gold logical-forms strings. However, the exact string-match penalizes correct predictions where the order of slots within an intent is different. For example, consider the following logical-forms:

LF-1: `[IN:GET_WEATHER [SL:ATTRIBUTE rainfall] [SL:DATE today ] ]`

LF-2: `[IN:GET_WEATHER [SL:DATE today][SL:ATTRIBUTE rainfall ] ]`

LF-1 and LF-2 are equivalent but the difference in the ordering of slots results in a negative match. Thus, we correct the EM metric by making the match function agnostic to the ordering of slots within an intent in the logical-form. We compare different models as per this corrected EM metric.

Semantic Parsing Model We use a pre-trained mT5-Large checkpoint (1.2B parameters) to initialize the downstream semantic parsing models that map utterances in the input to logical-forms in the output. We finetune the mT5 model on the original English dataset mixed with the translated datasets in target languages. We train using the Adafactor optimizer (Shazeer and Stern, 2018) with a fixed learning rate of $1e{-}3$ and a batch size of 256, for 30K steps using the T5X library (Roberts et al., 2022) on 64 TPU-v3 chips. Examples from each language are sampled uniformly for batch creation. For model selection, we choose the best performing checkpoint as per the dev splits and report our results on the test splits.

LLM-T (Our Method) We experiment with 8B, 62B, and 540B sized variants of PaLM (Chowdhery et al., 2022) as our LLM, and primarily utilize LLM-540B for translating English examples in different languages. For the seed set $\mathcal{S}_{\text{tgt}}$ used for prompting the LLM, we borrow roughly 250 examples covering 11 domains from MTOP's train set and 350 examples covering 18 domains from MASSIVE's train set (§ 3.1). During decoding, we sample 8 translations per example using top-p sampling (§ 3.3), with $p = 0.95$ and temperature scaling $T = 0.7$, followed by filtering out slot-inconsistent examples (§ 3.4). We present an analysis of our design choices in § 5.3.

Baselines (i) **Zero-Shot**: Train the model only on the English data and evaluate on other languages in a zero-shot manner. (ii) **Few-Shot**: In addition to the English training data, use the seed set of examples $\mathcal{S}_{\text{tgt}}$ for each language during training. For MTOP, $|\mathcal{S}_{\text{tgt}}| \approx 250$ and for MASSIVE, $|\mathcal{S}_{\text{tgt}}| \approx 350$. (iii) **TAF**: We implement the method from Nicosia et al. (2021) that uses an off-the-shelf translation service (§ 2.2) to construct $\mathcal{D}_{\text{tgt}}$ in all the target languages. We borrow $\mathcal{D}_{\text{tgt}}$ from Nicosia et al. (2021) for MTOP and from Nicosia and Piccinno (2022) for MASSIVE.

5 Results and Analysis

We first present downstream performance of semantic parsing models trained on data generated by our method (§ 5.1) and compare with zero-shot setting, few-shot setting, and the TAF method (Nicosia et al., 2021). We then compare our method against the "full-shot" skyline where we utilize the original training datasets that were manually translated with the help of human annotators in the target languages (§ 5.2). We then present an analysis of different design choices that result in effective data translation using LLM-T (§ 5.3). Finally, we present an error analysis to show the key sources of disagreements between the parser predictions and the ground truth (§ 5.4). All the experiments use our corrected EM metric (§ 4; Evaluation Metric).

5.1 Evaluation on MTOP and MASSIVE

In Table 1, we compare performance of different methods for the 5 non-English languages in the MTOP dataset. The Zero-Shot baseline trains an mT5 model only on the English part of the train-split. The Few-Shot baseline additionally includes the human translated seed sets $\mathcal{S}_{\text{tgt}}$ for each language. Both TAF and LLM-T train on the original English train set mixed with their respective translated datasets in each language. As all the baselines utilize the original English train set, we see comparable performance on English (around 85.0 EM). We observe LLM-T outperforms TAF in 4 out of 5 languages by 3.6 EM. Since LLM-T uses $\mathcal{S}_{\text{tgt}}$ for prompting, we also mix $\mathcal{S}_{\text{tgt}}$ with TAF data and still observe that LLM-T improves over TAF+Few-Shot by 2.9 EM. On relatively low-resource languages, Hindi (`hi`) and Thai (`th`), LLM-T leads to much larger improvements over TAF.

Figure 4 shows the performance difference between our LLM-T method and TAF for the MASSIVE dataset (FitzGerald et al., 2022). On 41 out of 50 languages, we find LLM-T to be better than TAF. For nine languages LLM-T outperforms TAF by more than 5.0 EM—Simple Man-

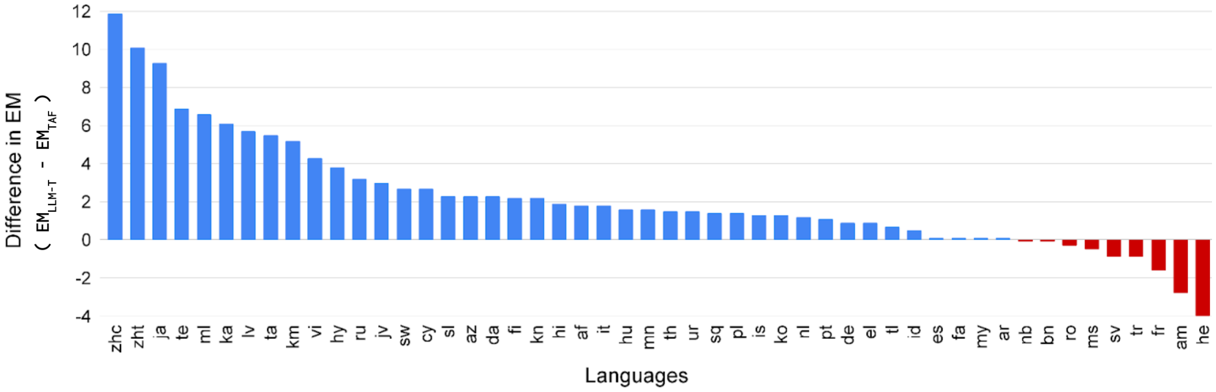


Figure 4: **EM accuracy difference between LLM-T and TAF across the 50 languages in MASSIVE dataset** (§ 5.1). LLM-T outperforms TAF on 41 out of 50 languages, with gains of more than 5 EM for nine of these languages. Only for Hebrew (he), LLM-T performs worse than TAF by more than 3 EM.

Method	**de**	**es**	**fr**	**hi**	**th**	**Avg**
Zero-Shot	54.4	57.8	62.8	42.3	42.1	51.9
Few-Shot	62.8	69.5	65.9	55.3	53.9	61.5
TAF	75.0	74.9	78.0	63.0	60.8	70.3
TAF + Few-Shot	**75.1**	74.5	78.5	63.9	62.9	71.0
LLM-T (ours)	74.0	**75.4**	**79.6**	**72.3**	**68.0**	**73.9**

Table 1: **EM accuracy comparison on MTOP** (§ 5.1): Data generated using LLM-T yields better performance on 4 out of 5 languages in MTOP. We observe large improvements for low-resource languages hi and th.

Dataset	**Few-Shot**	**TAF**	**LLM-T**	**Gold**
MTOP	61.5	70.3	73.9	**79.5**
MASSIVE	55.9	61.0	63.2	**67.7**

Table 2: **Comparison with Gold skyline** (§ 5.2): While training on the human translated datasets (Gold) yields the best performance, LLM-T results in a smaller performance gap compared to TAF. All numbers are averaged over the 5 non-English languages in MTOP.

darin (`zhc`, +11.9), Traditional Mandarin (`zht`, +10.1), Japanese (`ja`, +9.3), Telugu (`te`, +6.9), Malayalam (`ml`, +6.6), Kannada (`ka`, +6.1), Latvian (`lv`, +5.7), Tamil (`ta`, +5.5), and Khmer (`km`, +5.2). Only for Hebrew (`he`, −4.0), LLM-T is worse by more than 3.0 EM. Averaged across all languages, LLM-T outperforms TAF by 2.2 EM. In Appendix A.1, we provide detailed baseline comparisons for all the 50 languages.

5.2 Comparison with gold translations

An ideal translate-train method should be competitive w.r.t. training on fully human translated datasets. Table 2 provides a comparison between training on TAF, LLM-T, and the datasets fully translated with the help of human annotators in the target languages (Gold). Between TAF and Gold, we observe a significant gap of 9.2 EM in MTOP and 6.7 EM in MASSIVE. Our method LLM-T, reduces this gap by 3.6 EM in MTOP and 2.2 EM in MASSIVE. Overall, LLM-T achieves roughly 93% of the performance obtained by the Gold skyline that use more than 30× human translated examples. Appendix A.1, provides per-language comparisons with the Gold skyline for both the datasets.

Decoding Strategy	**de**	**es**	**fr**	**hi**	**th**	**Avg**
Greedy	71.1	71.7	72.6	68.1	66.0	69.9
+ Filtering	72.2	73.5	74.8	71.5	67.4	71.9
Top-p Sampling ($p = 0.95$)						
(#samples)						
1	70.1	71.5	74.3	66.9	67.2	70.0
2	71.4	72.1	74.5	68.8	67.2	70.8
4	71.1	72.8	76.4	69.0	66.0	71.1
8	71.9	72.7	74.2	70.0	**68.4**	71.4
Top-p Sampling + Filtering ($p = 0.95$)						
(#samples)						
1	72.0	75.2	78.9	71.6	68.1	73.2
2	73.7	75.2	79.5	72.0	67.6	73.6
4	73.4	75.3	79.0	72.1	67.7	73.5
8	**74.0**	**75.4**	**79.6**	**72.3**	68.0	**73.9**

Table 3: **Impact of decoding strategy and filtering**: Generating multiple translations per English example using top-p sampling followed by filtering inconsistent examples offers superior downstream performance compared to using greedy decoding or sampling just one translation per example. In Appendix A.2 we present results for top-k sampling as well.

Max Len	de	es	fr	hi	th	Avg
768	73.4	75.4	76.9	73.1	69.7	73.7
1024	74.0	75.4	79.6	72.3	68.0	73.9
1792	**74.3**	**75.7**	**80.5**	**74.0**	**71.1**	**75.1**

Table 4: **Impact of prompt length**: Longer prompts containing more exemplars result in more effective translated datasets yielding higher EM accuracy.

	de	es	fr	hi	th	Avg
LLM-T-8B	65.3	69.4	70.7	56.6	55.1	62.0
LLM-T-62B	72.0	73.3	76.7	68.2	65.6	71.2
LLM-T-540B	**74.0**	**75.4**	**79.6**	**72.3**	**68.0**	**73.9**

Table 5: **Impact of LLM size**: EM performance of semantic parsers trained on translated datasets improve with increasing the size of LLMs used for translation.

5.3 Analysis of Design Choices

We now present an analysis of the design choices that enabled more effective data translation via LLM-T. All the experiments in this section are carried out on the MTOP dataset.

Role of decoding strategy and filtering In Table 3, we present the EM accuracy of parsers trained on datasets translated using various combinations of decoding (§ 3.3) and filtering (§ 3.4) methods. For generating the translated outputs we experiment with greedy decoding, top-k (Fan et al., 2018) and top-p (Holtzman et al., 2019) sampling. Like prior translate-train methods, we begin with only one translation per example and observe sampling to be comparable with greedy decoding in downstream EM accuracy. In contrast, decoding two translations per example via sampling boosts the EM accuracy across all the languages. However, further increasing the translated samples to 4 and 8 results in only marginal performance differences. Manual inspection of the translated data revealed inconsistent utterance and logical-form pairs which motivated our design of slot-consistency based filtering (§ 3.4). Training the parser on filtered data provides further gains over training on unfiltered data. In Appendix A.2, we also present the results for top-k sampling. Overall, utilizing upto 8 top-p translated samples per English example followed by slot-consistency filtering provides the best performance averaged over all the languages.

Impact of Prompt Length We expect prompts containing more exemplars to yield higher quality translated examples owing to more information for in-context learning. In Table 4, we compare EM performance when using maximum prompt-lengths of 768, 1024, and 1792 tokens. Training on datasets translated using prompt-length of 1792 tokens provides the best downstream EM performance across all the languages. However, longer prompts lead to considerably longer inference times. Hence, we conduct our main experiments with prompt the length of 1024 tokens.

Role of LLM size In Table 5, we compare parser performance when trained on data generated by LLMs of different sizes. Training on larger LLM generated data leads to better performance—LLM-T-540B yields the best performance on all the languages, followed by LLM-T-62B which outperforms LLM-T-8B on all the languages.

5.4 Error Analysis

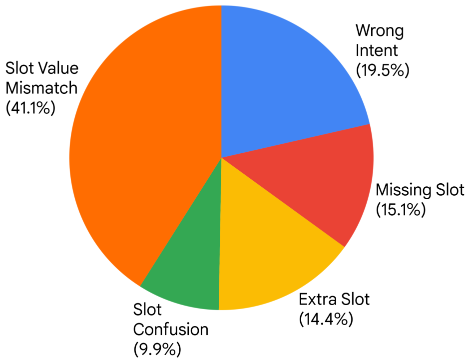


Figure 5: **Distribution of error categories**: estimated across all five languages on MTOP's dev set.

We analyze the examples where the predictions from our semantic parser do not match with the ground truth. In Table 6, we categorize all the erroneous examples into five broad categories (with English examples): (i) Slot Value Mismatch (ii) Wrong Intent (iii) Missing Slot (iv) Extra Slot and (v) Slot Confusion. Figure 5 presents the distribution of the error categories aggregated across all the languages on the MTOP dev-split. The "Slot Value Mismatch" is the most frequent error category (41.1%)—here the predicted parse structure is correct but the slot-values do not match perfectly with the gold parse. After manually inspecting 300 such errors we found that in roughly 50% of the cases the predicted and gold slot-values often have minor mismatches which may not be recognized as error by another human annotator and should not lead to incorrect output upon logical form execu-

Slot Value Mismatch (41.1%)	Utterance: Set an alarm for 5 pm tomorrow Prediction: [IN:CREATE_ALARM [SL:DATE_TIME for 5 pm] [SL:DATE_TIME tomorrow] Target: [IN:CREATE_ALARM [SL:DATE_TIME 5 pm] [SL:DATE_TIME tomorrow]
Wrong Intent (19.5%)	Utterance: What can I do today Prediction:[IN:QUESTION_NEWS [SL:DATE_TIME today]] Target: [IN:GET_EVENT [SL:DATE_TIME today]]
Missing Slot (15.1%)	Utterance: Play Justin Timberlake 's newest single Prediction:[IN:PLAY_MUSIC [SL:MUSIC_TYPE single]] Target: [IN:PLAY_MUSIC [SL:MUSIC_ARTIST_NAME Justin Timberlake] [SL:MUSIC_TYPE single]]
Extra Slot (14.4%)	Utterance: play music on the speaker Prediction: [IN:PLAY_MUSIC [SL:MUSIC_TYPE music] [SL:MUSIC_TYPE speaker]] Target: [IN:PLAY_MUSIC [SL:MUSIC_TYPE music]]
Slot Confusion (9.9%)	Utterance: audio call wedding planner please Prediction:[IN:CREATE_CALL [SL:CONTACT wedding planner]] Target: [IN:CREATE_CALL [SL:GROUP wedding planner]]

Table 6: **Examples of Error Categories** (§ 5.4) The errors in the predicted parse can be broadly classified into five categories: (i) Slot Value Mismatch: Predicted parse has the correct signature but the slot-values are incorrect, (ii) Wrong Intent: High-level intent of the predicted parse is incorrect, (iii) Missing Slot: One or more slots in the gold parse do not appear in the output, (iv) Extra Slot: Output contains extra slot(s) compared to the gold, (v) Slot Confusion: Prediced parse contains the correct correct intent and number of slots but the wrong slot-types.

tion. For example, in the first row of Table 6, the predicted value for the `DATE_TIME` slot is `'for 5 pm'`, while the target value is just `'5 pm'`.

6 Related Work

Multilingual Semantic Parsing Multilingual semantic parsers are typically initialized with a foundation model (Bommasani et al., 2021) pre-trained on vast amounts of multilingual data (Conneau et al., 2020; Xue et al., 2021; Li et al., 2021; FitzGerald et al., 2022) followed by supervised training on synthetic or real multilingual datasets. A standard approach for constructing multilingual datasets is to translate and localize English datasets with the help of multilingual speakers or machine translation. For example, MTOP (Li et al., 2021), MASSIVE (FitzGerald et al., 2022), and MultiAtis++ (Xu et al., 2020) were constructed by translating TOP (Gupta et al., 2018), SLURP (Roberts et al., 2022), and ATIS (Price, 1990) respectively through human translators.

Machine Translation based methods Machine translation based approaches continue to be important for multilingual task-specific models (Hartrumpf et al., 2008; Liang et al., 2020; Hu et al., 2020; Fang et al., 2021; Ladhak et al., 2020) including semantic parsing. Machine translation can either be used during the inference time to translate a user query into English for feeding it to an English-only model. This approach is referred to as *translate-test* (Artetxe et al., 2020; Uhrig et al., 2021). A more common way of using machine translation is in the form of data augmentation, referred as *translate-train* where English text in training data is translated into several languages (Sherborne et al., 2020; Moradshahi et al., 2020, 2021; Xia and Monti, 2021; Nicosia et al., 2021; Gritta et al., 2022; Wang et al., 2022). In practice, *translate-train* methods tend to outperform *translate-test* methods while also reducing the latency associated with translating text during the inference time (Yang et al., 2022).

LLMs and Few-Shot learning Transformer (Vaswani et al., 2017) based generative LLMs (Radford et al., 2019; Brown et al., 2020; Thoppilan et al., 2022; Soltan et al., 2022; Smith et al., 2022; Zhang et al., 2022; Chowdhery et al., 2022) trained on massive amounts of web-scale text corpora using next token prediction objective exhibit strong few-shot generalization abilities. When prompted with a task description and a handful of task-specific examples, LLMs can often match the performance of finetuned models via in-context learning (Xie et al., 2021; Min et al., 2022; Wei et al., 2022; Zhou et al., 2022). We utilize LLMs for translating English datasets in several languages using few-shot prompting.

7 Conclusion

We present a method of utilizing large language models (LLMs) for bootstrapping multilingual semantic parsers across several languages. In comparison to using off-the-shelf translation services that rely on significant amounts of human supervision, we demonstrate that prompting self-supervised LLMs can be a more effective and scalable alternative for dataset translation. We find that generating multiple diverse translations using sampling techniques followed by consistency-based filtering make the translated datasets more effective for training multilingual semantic parsers. On 41 out of 50 typologically diverse languages within two large datasets spanning several domains, our method outperforms a strong translate-train method that utilizes a supervised translation service.

8 Limitations

While translating English queries in different languages is a useful form of data augmentation, we think that further performance improvements can be obtained by careful localization of entities in the text queries. This will result in examples where the training dataset contains entities that are often talked about in the target language and might lead to less train-test domain shift. LLMs contain language specific priors which can be harnessed to perform such localization of the translated queries thus enabling more realistic data augmentations. In this work we presented a simple string-match based filtering technique to remove noisy translations. Data filtering can be further improved with the help of learned models. We observed that larger LLMs are important to generate more effective translated data. However running these experiments is constrained by the availability of large amounts of compute resources. We hope future work will address these limitations of our approach.

9 Ethical Considerations

We utilize large language models to translate datasets initially available in English into several languages. The real-world deployment of models trained on LLM-translated data should undergo a careful review of any harmful biases. However, the LLM-translated data and the logical-forms generated by a semantic parser are not user-facing, thus a smaller risk of any direct harms. The intended users of any semantic parsing model must be made aware that the answers returned by the model could be incorrect, more so for user-queries in low-resource languages. We do not immediately foresee any serious negative implications of the specific contributions that we make in this work.

10 Acknowledgment

We thank Rahul Goel, Slav Petrov, and Kartikeya Badola for discussions and their feedback on an earlier draft of this paper. We thank Massimo Nicosia for sharing the TAF translated datasets and helpful discussions during this project.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2020. Translation artifacts in cross-lingual transfer learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7674–7684.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

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

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Yuwei Fang, Shuohang Wang, Zhe Gan, Siqi Sun, and Jingjing Liu. 2021. Filter: An enhanced fusion method for cross-lingual language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12776–12784.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, et al. 2022. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages. *arXiv preprint arXiv:2204.08582*.

Milan Gritta, Ruoyu Hu, and Ignacio Iacobacci. 2022. CrossAligner & co: Zero-shot transfer methods for task-oriented cross-lingual natural language understanding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4048–4061, Dublin, Ireland. Association for Computational Linguistics.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792.

Sven Hartrumpf, Ingo Glöckner, and Johannes Leveling. 2008. Efficient question answering with question decomposition and multiple answer streams. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 421–428. Springer.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multitask benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Sawan Kumar and Partha Talukdar. 2021. Reordering examples helps during priming-based few-shot learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4507–4518.

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4034–4048, Online. Association for Computational Linguistics.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang Liu, Fan Yang, Daniel Campos, Rangan Majumder, and Ming Zhou. 2020. XGLUE: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Mehrad Moradshahi, Giovanni Campagna, Sina Semnani, Silei Xu, and Monica Lam. 2020. Localizing open-ontology QA semantic parsers in a day using machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5970–5983, Online. Association for Computational Linguistics.

Mehrad Moradshahi, Victoria Tsai, Giovanni Campagna, and Monica S Lam. 2021. Contextual semantic parsing for multilingual task-oriented dialogues. *arXiv preprint arXiv:2111.02574*.

Massimo Nicosia and Francesco Piccinno. 2022. Evaluating byte and wordpiece level models for massively multilingual semantic parsing. *arXiv preprint arXiv:2212.07223*.

Massimo Nicosia, Zhongdi Qu, and Yasemin Altun. 2021. Translate & fill: Improving zero-shot multilingual semantic parsing with synthetic data. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3272–3284.

P. J. Price. 1990. Evaluation of spoken language systems: the ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, et al. 2022. Scaling up models and data with `t5x` and `seqio`. *arXiv preprint arXiv:2203.17189*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating high-quality and informative conversation responses with sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2210–2219.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Tom Sherborne, Yumo Xu, and Mirella Lapata. 2020. Bootstrapping a crosslingual semantic parser. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 499–517.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Saleh Soltan, Shankar Ananthakrishnan, Jack FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith Peris, Stephen Rawls, Andy Rosenbaum, Anna Rumshisky, et al. 2022. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model. *arXiv preprint arXiv:2208.01448*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Sarah Uhrig, Yoalli Garcia, Juri Opitz, and Anette Frank. 2021. Translate, then parse! a strong baseline for cross-lingual AMR parsing. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 58–64, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

Zhiruo Wang, Grace Cuenca, Shuyan Zhou, Frank F Xu, and Graham Neubig. 2022. Mconala: A benchmark for code generation from multiple natural languages. *arXiv preprint arXiv:2203.08388*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Menglin Xia and Emilio Monti. 2021. Multilingual neural semantic parsing for low-resourced languages. In *The Tenth Joint Conference on Lexical and Computational Semantics*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Weijia Xu, Batool Haider, and Saab Mansour. 2020. End-to-end slot alignment and recognition for cross-lingual nlu. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5052–5063.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Diyi Yang, Ankur Parikh, and Colin Raffel. 2022. Learning with limited text data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 28–31, Dublin, Ireland. Association for Computational Linguistics.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI*, pages 1050–1055, Portland, OR. AAAI Press/MIT Press.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, UAI'05, page 658–666, Arlington, Virginia, USA. AUAI Press.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2021. Trading off diversity and quality in natural language generation. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 25–33.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Appendix

A.1 Additional results

Lang	Zero-Shot	Few-Shot	TAF	TAF + Few-Shot	LLM-T (top-k)	LLM-T (top-p)	Gold (skyline)
af	48.5	59.0	64.5	64.5	66.7	66.3	68.5
am	31.0	47.6	58.3	57.4	56.1	55.5	64.6
ar	35.9	50.5	57.2	58.0	56.6	57.3	65.5
az	39.3	57.1	60.5	60.5	62.6	62.8	68.8
bn	40.8	55.4	62.1	61.7	61.1	62.0	68.3
cy	26.7	44.8	59.1	58.3	61.1	61.8	65.1
da	57.5	62.4	66.2	66.3	69.1	68.5	71.0
de	54.3	62.8	67.5	67.7	68.3	68.4	70.4
el	47.3	57.8	64.2	65.5	65.2	65.1	68.7
en	72.7	71.4	73.5	72.9	73.3	73.4	73.0
es	53.4	58.1	64.6	64.6	64.7	64.7	66.6
fa	48.8	58.0	63.1	62.9	62.8	63.2	68.1
fi	47.5	58.4	65.0	65.3	66.7	67.2	70.9
fr	54.6	58.0	65.3	64.9	63.9	63.7	67.1
he	35.3	56.1	60.6	61.2	55.3	56.6	68.3
hi	40.1	54.4	61.6	62.5	63.1	63.5	66.2
hu	44.1	57.1	63.8	63.6	64.5	65.4	69.7
hy	39.3	53.8	58.7	59.2	62.3	62.5	67.1
id	55.3	60.2	65.5	65.9	66.6	66.0	69.1
is	41.3	54.4	62.2	61.5	63.6	63.5	69.5
it	52.3	58.6	64.0	63.6	65.2	65.8	67.2
ja	45.6	55.1	56.3	56.5	65.6	65.6	67.3
jv	34.3	51.7	58.6	60.2	62.0	61.6	66.7
ka	36.5	53.4	53.5	54.6	59.2	59.6	65.7
km	37.8	51.1	49.1	53.7	55.3	54.3	62.8
kn	37.1	49.3	55.0	55.9	57.7	57.2	62.1
ko	42.1	56.3	62.2	63.6	62.4	63.5	69.3
lv	45.4	56.0	60.4	61.3	66.0	66.1	68.8
ml	38.6	53.9	55.5	56.9	62.5	62.1	67.5
mn	30.9	51.4	57.6	59.4	59.5	59.2	68.0
ms	48.6	58.9	66.2	66.2	65.8	65.7	69.2
my	38.1	54.9	60.5	62.3	61.5	60.6	69.6
nb	55.2	63.0	67.5	67.7	67.7	67.4	71.0
nl	53.1	61.2	67.3	68.5	68.7	68.5	70.5
pl	50.5	57.4	61.1	61.4	62.9	62.5	65.6
pt	54.9	60.3	65.8	65.7	66.4	66.9	68.5
ro	51.2	58.8	65.4	65.0	64.8	65.1	68.8
ru	42.3	59.4	63.0	63.1	66.6	66.2	69.4
sl	46.0	57.8	63.1	64.0	65.3	65.4	68.8
sq	41.0	55.4	60.3	60.4	62.1	61.7	67.3
sv	57.2	63.1	69.8	69.6	69.3	68.9	72.4
sw	35.7	52.3	57.9	57.5	60.9	60.6	65.3
ta	37.2	53.0	55.4	55.7	60.7	60.9	65.8
te	38.7	49.0	51.6	53.6	56.8	58.5	61.6
th	49.4	60.0	63.5	66.5	65.2	65	71.5
tl	48.4	55.7	64.1	64.2	65.2	64.8	67.5
tr	46.7	58.5	63.7	63.4	62.7	62.8	69.4
ur	38.9	51.2	60.4	60.6	62.2	61.9	64.6
vi	46.9	55.1	59.0	59.2	63.0	63.3	67.6
zhc	34.7	56.1	52.0	53.9	64.2	63.9	66.3
zht	35.2	51.8	50.5	52.3	60.7	60.6	63.6
Avg	43.8	55.9	61.0	61.6	63.2	63.2	67.7

Table A1: EM accuracy comparison on MASSIVE dataset. Avg reports the EM accuracy averaged across the 50 non-English languages

Lang	Zero-Shot	Few-Shot	TAF	TAF + Few-Shot	LLM-T (top-k)	LLM-T (top-p)	Gold (skyline)
de	54.4	62.8	75.0	75.1	73.7	74.0	78.5
es	57.8	69.5	74.9	74.5	75.2	75.4	82.9
fr	62.8	65.9	78.0	78.5	79.7	79.6	80.8
hi	42.3	55.3	63.0	63.9	72.5	72.3	78.5
th	42.1	53.9	60.8	62.9	66.8	68.0	77.0
en	84.1	84.0	85.2	85.0	85.2	85.1	85.4
Avg	51.9	61.5	70.3	71.0	73.6	73.9	79.5

Table A2: EM accuracy comparison on MTOP dataset. Avg reports the EM accuracy averaged across the 5 non-English languages

In Table A1, we present detailed baseline comparisons for all the 51 languages in the MASSIVE dataset. Zero-Shot, Few-Shot, TAF, and TAF+Few-Shot are the baselines described in Section 4. LLM-T represents our method with top-k or top-p sampling used while decoding the translated examples. Gold is the "full-shot" skyline which utilizes the original human-translated datasets (§ 5.2). Table A2 presents the same set of results for the six languages in the MTOP dataset.

A.2 Role of decoding strategy and filtering

In Table A3 we present results for different decoding strategies and role of filtering inconsistent examples as discussed in Section 5.3.

Decoding Strategy	de	es	fr	hi	th	Avg
Greedy	71.1	71.7	72.6	68.1	66.0	69.9
+ Filtering	72.2	73.5	74.8	71.5	67.4	71.9
Top-k Sampling ($k = 40$)						
(#samples)						
1	70.5	71.7	73.1	66.8	66.5	69.6
2	72.3	72.7	75.7	68.7	67.3	71.3
4	71.3	73.1	73.8	68.5	67.8	70.9
8	71.1	72.5	74.2	69.3	67.5	70.9
Top-k Sampling + Filtering ($k = 40$)						
(#samples)						
1	72.4	74.4	78	70.9	66.1	72.4
2	73.6	74.4	78.2	72.1	67.9	73.2
4	73.4	75.3	78.8	71.4	67.1	73.2
8	73.7	75.2	**79.7**	**72.5**	66.8	73.6
Top-p Sampling ($p = 0.95$)						
(#samples)						
1	70.1	71.5	74.3	66.9	67.2	70.0
2	71.4	72.1	74.5	68.8	67.2	70.8
4	71.1	72.8	76.4	69.0	66.0	71.1
8	71.9	72.7	74.2	70.0	68.4	71.4
Top-p Sampling + Filtering ($p = 0.95$)						
(#samples)						
1	72.0	75.2	78.9	71.6	**68.1**	73.2
2	73.7	75.2	79.5	72.0	67.6	73.6
4	73.4	75.3	79.0	72.1	67.7	73.5
8	**74.0**	**75.4**	79.6	72.3	68.0	**73.9**

Table A3: **Impact of decoding strategy and filtering**: Generating multiple translations per English example using top-k or top-p sampling followed by filtering inconsistent examples offers superior downstream performance compared to using greedy decoding or sampling just one translation per example.

Modeling Complex Event Scenarios via Simple Entity-focused Questions

Mahnaz Koupaee[1], **Greg Durrett**[2], **Nathanael Chambers**[3], **Niranjan Balasubramanian**[1]

[1] Stony Brook University, [2] The University of Texas at Austin, [3] United States Naval Academy
[1]`{mkoupaee,niranjan}@cs.stonybrook.edu`
[2]`gdurrett@cs.utexas.edu`, [3]`nchamber@usna.edu`

Abstract

Event scenarios are often complex and involve multiple event sequences connected through different entity participants. Exploring such complex scenarios requires an ability to branch through different sequences, something that is difficult to achieve with standard event language modeling. To address this, we propose a question-guided generation framework that models events in complex scenarios as answers to questions about participants. At any step in the generation process, the framework uses the previously generated events as context, but generates the next event as an answer to one of three questions: *what else a participant did, what else happened to a participant, or what else happened*. The participants and the questions themselves can be sampled or be provided as input from a user, allowing for controllable exploration. Our empirical evaluation shows that this question-guided generation provides better coverage of participants, diverse events within a domain, comparable perplexities for modeling event sequences, and more effective control for interactive schema generation[1].

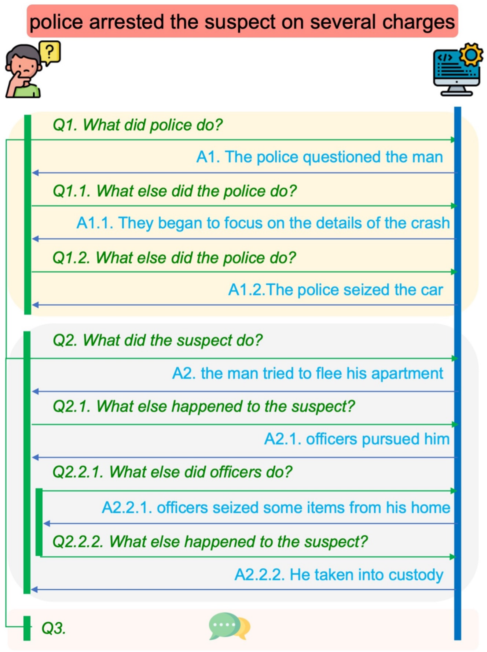


Figure 1: Question-guided event sequence generation. The user can interact with the system by asking questions regarding the entities and the system will generate corresponding events. Different questions can lead to different paths as shown with different colored boxes.

1 Introduction

Event scripts (Schank and Abelson, 1977), also known as event schemas, describe a sequence of events in a particular context. Representing and modeling such schemas is central to applications in AI such as question answering, discourse understanding, and information extraction (Balasubramanian et al., 2013). Early work used hand-crafted event schemas as a starting point (Schank and Abelson, 1977; Mooney and DeJong, 1985), but modern techniques attempt to extract these at a large scale from unlabeled data (Chambers and Jurafsky, 2008; Chambers, 2013; Pichotta and Mooney, 2016; Weber et al., 2018b).

Event language models can also be used to approximate schematic knowledge via event sequences. They can be trained to generate a sequence of events with their participating roles describing a real-life scenario. However, these scenarios can often be complex and don't always fit as simple sequences (Weber et al., 2018b). For example, suppose we have the following event: `police arrested suspect on several charges`, with **`police`**, **`suspect`** and **`charges`** as entities. The scenario can be described in multiple ways depending on which entities we want to focus on and what roles they play in the subsequent events. We may be interested in knowing what the **`police`** did, what the **`suspect`** did or what happened to the **`charges`**, each of which can be explored as its own sequence,

---

[1] The code is available at `https://github.com/StonyBrookNLP/qa-event-lms`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2468–2483
May 2-6, 2023

yielding many interconnected sequences of events in the scenario. An event language model, however, will simply generate events conditioned on the previous events in the discourse, with no direct mechanism to guide generation towards areas of interest within the scenario. As a result, with standard decoding strategies we often end up with a sequence of events that might be relevant to the scenario, but not necessarily cover the broad set of diverse paths in the scenario; we would only know the fate of certain entities if the system samples events that included them. This lack of control will make it difficult to use the system, as one must keep sampling events until it just happens to produce events with roles one is interested in.

We propose a simple modification in which event language models are trained to also condition on the entities and the roles they play in a scenario through a set of simple questions, as illustrated in Figure 1. Given the same example event and its entities, one can then explore the scenario through various paths by asking the system to generate the events with their desired entities and roles. Such a question-guided model can be used in interactive settings to model and construct diverse paths that cover various aspects of complex scenarios.

A key challenge, however, is in creating the necessary training data at scale. We show that we can repurpose standard event sequences to create training data for question-guided models: we take a partial sequence of events as context and derive a role-based question involving an entity for which a future event in the sequence can be an answer. This allows for creating large scale training instances that are (Context, Question, Answer) triples, which then allows us to train models that can better respond to user control in the form of questions.

Our analysis shows that question-guided event language models can generate sequences with more diversity and comparable quality as an event language model. Our human evaluation of the model in an interactive setting, shows that the controllability of the question-guided model allows for generation of sequences that lead to better quality, broader-coverage schemas with fewer interactions. This interactive evaluation is a step toward constructing schematic/common-sense knowledge for analyzing events, an application in the intelligence analysis community.

In summary, this paper makes the following contributions; (i) It argues the need for control in event language models to explore complex scenarios; (ii) It provides a simple yet effective way for training event language models that can be guided to explore different aspects of complex scenarios; (iii) It provides empirical evidence showing improved control and utility via automatic and manual evaluations.

2 Related Work

Event Schema Induction Event scripts (or schemas) originally proposed by (Schank and Abelson, 1977), consist of a set of events and actors (also known as slots) playing different roles. The event schemas are capable of analyzing complex situations by encoding information from prototypical events and their participants.

Early works on scripts considered them as structured representations of events and their participants with the causal relationships between them (Schank and Abelson, 1977; Mooney and DeJong, 1985). However, the manual construction of scripts is too time-consuming and does not scale, so the scripts could only focus on specific domains of interest and have not been used more broadly.

Event schemas can also be induced automatically from text using statistical techniques in an unsupervised fashion (Chambers and Jurafsky, 2008, 2009; Balasubramanian et al., 2013). These models are easily interpretable but fail to capture long-distance complex relationships between events. Event language modeling is a type of schema induction via language modeling techniques (Rezaee et al., 2021). Given a sequence of events, the event language model predicts the probability of the next event (Manshadi et al., 2008). Framing schema learning as a language modeling problem with various ways to represent events, including word sequences annotated with predicate-argument structure (Pichotta and Mooney, 2016), OpenIE tuples (Rudinger et al., 2015; Weber et al., 2018a,b) or compositional embeddings (Modi, 2016), is another direction towards realizing large-scale schema libraries.

Graph schema induction methods (Li et al., 2020, 2021) model different relations between entities and their arguments to capture the multi-dimensionality of scenarios. Reasoning about complex relations between events requires going beyond the single dimension of event cooccurrence and capturing different types of semantic relations between events such as causal, counterfactual, etc. (Han et al., 2021). Our approach uses the standard language models and provides them with the guidance to also produce different aspects of a scenario via simple control codes.

Controlled text generation Language models have shown promising results in text generation, however, it is not easy to have control over different aspects of generation (Keskar et al., 2019), an issue that has been studied in previous works (Dathathri et al., 2019; He et al., 2020; Lu et al., 2021; Mireshghallah et al., 2022). The key components of these methods differ in the types of controls provided and how they are provided and their applications. CTRL (Keskar et al., 2019) trains a very large language model by conditioning on texts with appended control codes that are used to guide the generation towards specific styles, contents, and task-specific behaviors. CTRLSum (He et al., 2020) uses the entity/length controls which are in forms of keywords that are automatically extracted from the text and trains a summarization system which is capable of generating summaries in an interactive manner. These approaches, however, require finetuning. Dathathri et al. (2019) and Mireshghallah et al. (2022) propose variants of controllable generation with no need to finetune or retrain the whole system.

Learning latent representations or codes from the input towards diverse generation is another direction that has been explored for machine translation (Shu et al., 2019), dialogue generation (Huang et al., 2018) and causal relations generation (Weir et al., 2020). Controllable generation to generate diverse events has been previously studied in Kwon et al. (2021), where the system uses automatically generated control codes to generate diverse preconditions of events.

In this work, we use controllable generation to model complex event scenarios. We introduce simple role-based questions about participants (agentive or non-agentive) as an effective means for control. Asking questions to get specific information about events is the focus of many existing approaches. While there has also been a body of work on semantic role labeling using QA pairs (Roit et al., 2020; Klein et al., 2020; Michael and Zettlemoyer, 2021; Pyatkin et al., 2021), the main distinction here lies in the fact that these approaches use QA pairs to identify the semantic roles, whereas our approach makes use of role-based questions to generate the next event with a specific entity playing a specific role. We show how to train for these control codes using automatically derived training sequences and demonstrate its utility for describing complex scenarios in an interactive setting.

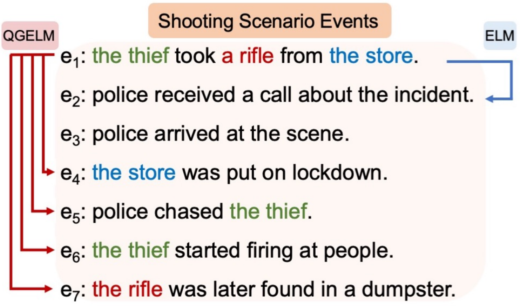


Figure 2: The shooting scenario events extracted from a news article. Typical LMs only see e_2 as the immediate next event, whereas for the question-guided LMs, any of the e_4, e_5, e_6 and e_7 involving *the store*, *the thief*, or *the rifle* can be considered a next event.

3 Question-guided Event Language Modeling

Event language models aim at predicting the probability of an event given a set of events via a conditional probability distribution. Formally, these models try to find an event $\hat{e}$ by maximizing the probability of a function parameterized by a model over a given context including a set of events:

$$\hat{\mathrm{e}} = \mathrm{argmax}_{\mathrm{e} \in \mathrm{E}} \mathrm{P}_\theta(\mathrm{e} \mid \mathrm{context})$$

where E is the set of all events. A model trained with this objective learns to generate the most probable event based on a set of cooccurring events from discourse. Suppose that shooting scenario events are extracted as shown in Figure 2. Given the first event in the sequence, a typical language model is trained to generate the next event, e_2, which does not include any entities from the prior context. What if we are interested in looking for events regarding the initial participants (*the thief*, *a rifle*, or *the store*) in this example? Given the same context, we can have multiple next events depending on the participant and the role they can have (agent or theme). If we are interested in knowing what *the thief* did, e_6 should be the next event, what happened to *the thief* is described in e_5. We can have some information about *the store* with e_4 as the next event and finally, the fate of *the rifle* is described in e_7. Controlling the model to generate the outputs based on the participants involved cannot be easily achieved with unguided (text ordered) event language models. However, we will show

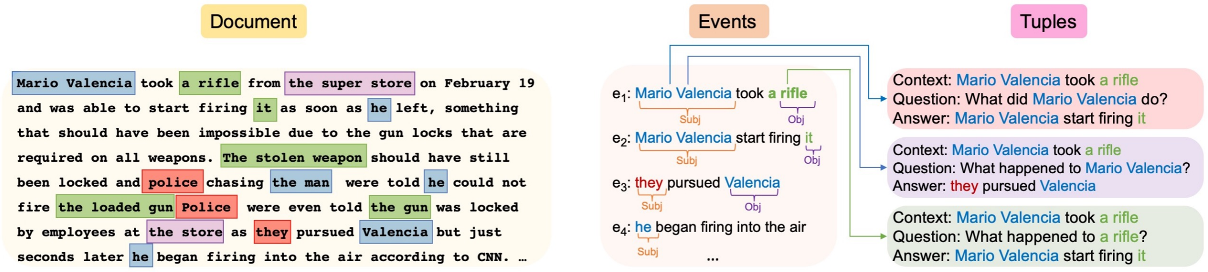


Figure 3: Data processing to create instances for a question-guided event language model. Given a document, we extract all the coreferring clusters (color-coded in the document). Next, we extract OpenIE tuples as events representations. By identifying the roles of noun phrases in the events as well as knowing which cluster they belong to, we create questions for each one of them and similarly find the events that can serve as the answers to those questions. The context can be of any length whereas the answer event is of length 1.

how the language models can be guided such that given the same context, they can directly generate events for its participants.

3.1 Problem Definition

We propose a new framing in which an event language model can be guided to generate events by not only conditioning on the events but also on specific entity-based questions of interest.

These entity-aware event language models also look to find $\hat{e}$, but by maximizing the probability of a function parameterized by a model over a given context *and* a question regarding an entity:

$$\hat{\mathrm{e}} = \mathrm{argmax}_{\mathrm{e} \in \mathrm{E}} \mathrm{P}_{\theta}(\mathrm{e} \mid \mathrm{context}, \mathrm{question})$$

The objective now conditions on a question as well, and the goal is to use the question to train a system to generate the most probable event for a specific entity (or a noun phrase referring to that entity) in a specific role (agent or theme). As shown in Figure 1, questions can either ask about what an entity did as an agent of an action (*what did X do?*) or what happened to an entity as a theme of an action (*what happened to X?*). By conditioning on the question as well, the system will learn to generate an event with the entity in question as well as the role specified by that question.

3.2 Question-guided Training

Event language models are typically trained using event sequences extracted from documents. Our goal, however, is to train event language models to generate events as *answers* to *questions* about entities from a given *context*. To this end, we convert event sequences in text to *(Context, Question, Answer)* tuples (CQA instances) that can be used as training data for question-guided generation.

Consider a sequence $e_1, e_2, ...e_n$ of OpenIE event tuples extracted from a document D. We create *(Context, Question, Answer)* tuples for each event e_t in the sequence as outlined in Algorithm 1 in Appendix A.1 and Figure 3. The key idea behind this process is as follows: Suppose we observe an entity in an event e_i. If this entity also appears in a subsequent event e_k, then we can see this new event as an answer to a role-based question about the entity (what did the entity do or what happened to the entity), given what we know about all the events that have been observed thus far in the sequence as context. For example, as shown in Figure 3, the entity `Mario Valencia` appears in two events e_1 and e_2. Given the context e_1, we can create the question *What did Mario Valencia do?* for which the answer is e_2 i.e., *Mario Valencia started firing it*. Formally, for each entity (any noun phrase) np that appears as an argument in the event e_i, we do the following. If np appears in an agentive role in some subsequent event e_k ($k > i$), then we associate the question *What else did* np *do?* with the context and use event e_k as the answer to the question. If np appears in a non-agentive role, we associate *What else happened to* np*?* as the question and e_k as the answer. In either case, for the next step the context is extended to include e_k and the process is repeated for all arguments in e_k. To handle events e_k that introduce new entities as arguments, we use *What else happened?* as the question.

We use automatically identified coreference clusters to locate event mentions involving a specific entity. We use simple dependency-based heuristics to determine the role of an entity in an event. Given the noun phrases of an event, we use a dependency parser to identify their roles. An entity is deemed to appear in an agentive role if it appears as a sub-

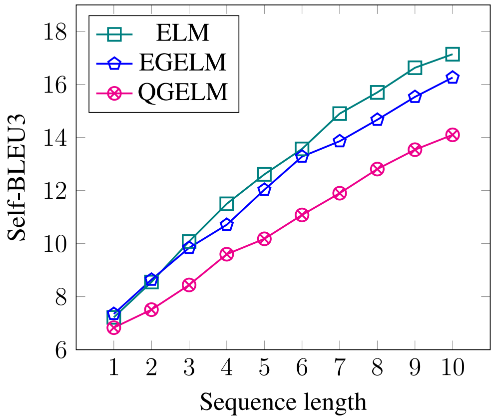


Figure 4: Diversity of generated sequences of events with varying lengths. Lower Self-BLEU scores are better as they represent more diverse sequences. QGELM is the most diverse across all sequence lengths.

		beam	sampling	
Criteria	**System**	**Fail %** ▼	**Fail %** ▼	**Avg # samples**
Any presence	ND	45.63	-	-
	ELM	79.29	26.05	17.21
	EGELM	43.39	2.73	4.47
	QGELM	**38.92**	**1.53**	**3.51**
Role specific presence	ND	69.84	-	-
	ELM	85.55	35.00	21.57
	EGELM	59.04	6.34	8.42
	QGELM	**43.03**	**2.78**	**4.54**

Table 1: Controllability Assessment: Fail % denotes the number of instances where the model fails to generate the specified entity. For sampling, we also show the average number of samples needed before an event with the specified entity is generated. ND denotes NeuroLogic decoding (Lu et al., 2021), a beam-search based controllable method with no sampling strategy.

ject and in a non-agentive role if it appears as an object. We only use these two broad categories (subject and object) to identify events as responses to specific questions. This process will automatically filter out the nonsensical questions as for such questions, no event can be found within the given sequence of events.

Note this training data has two properties that are different from standard auto-regressive training over event sequences. First, for the same conditioning context of event tuples, the model learns to generate multiple subsequent events depending on the question being asked, thus ensuring better control and diversity. Second, the model also learns to generate events that are not always adjacent to the end of the current sequence, which can be seen as a form of data augmentation shown to be effective (Koupaee et al., 2021).

4 Experimental Setup

4.1 Models

All models used in our experiments are trained using event sequences, which are OpenIE (Mausam et al., 2012) tuples extracted from articles in their discourse order.

ELM Our baseline is an **E**vent **L**anguage **M**odel trained such that given a context (a set of events), it will generate the next event. The baseline model follows the existing mechanisms used in recent prior ELM work (Manshadi et al., 2008; Rudinger et al., 2015; Pichotta and Mooney, 2016; Weber et al., 2018a,b). We train a T5 base model to learn $P(e_n \mid \text{context})$ where context $= e_1, ..., e_{n-1}$.

EGELM The **E**ntity-**G**uided **E**vent **L**anguage **M**odel generates the next event conditioned both on the context and one of its entities by learning the following probability distribution:

$$\mathrm{P}(\mathrm{e_n} \mid \mathrm{context}, \mathrm{entity})$$

In this setting, the system learns to maximize the probability of the next event with respect to a specific entity from the context but without considering the specific role in which the entity appears in the generated event. The training instances in this case are (Context, Entity, Answer) tuples that are obtained by replacing the question with the entity mention present in the question in the training instances described in Section 3.2.

QGELM The **Q**uestion-**G**uided **E**vent **L**anguage **M**odel generates the next event conditioned both on the context and a question regarding one of the entities in the context and a ***role*** the entity plays in the event. Here, the system learns the following probability distribution:

$$\mathrm{P}(\mathrm{e_n} \mid \mathrm{context}, \mathrm{question})$$

The main difference with the entity-only system is the different roles an entity can have through the question's surface form.

4.2 Data Statistics

We train all the models on the 2007 portion of the Annotated NYT corpus (Sandhaus, 2008). This subset contains a total of around 38k articles spanning over a 6 month period. Following the steps

System	Perplexity ▼	NC Accuracy▲
ELM	24.64	46.2%
EGELM	22.45	48.6%
EGELM (margin)	25.06	46.8%
QGELM	**22.11**	**49.3%**
QGELM (margin)	26.63	46.0%

Table 2: Perplexity and the narrative cloze accuracy. Lower perplexity and higher accuracy is desirable.

Metric	ELM	QGELM	change
# accepted events▲	6.2	8.8	42% ↑
# rejected steps▼	5.2	3.2	38% ↓
% rejected steps▼	41.0	26.6	35% ↓
# resamples▼	4.9	3.2	35% ↓
total steps▲	11.3	12.0	6% ↑
tree depth▲	5.8	8.8	52% ↑

Table 3: Quantitative analysis of schema generation using the ELM and QGELM models. With QGELM, users accepted more of its suggested events, rejected fewer steps, used fewer resamples for a given context, and produced longer event sequences. The higher the average the better a system is for metrics with ▲ whereas lower values are desired for metrics with ▼.

described earlier on these number of articles, we end up having over 700k instances (consisting of (context:$e_1, e_2, ..., e_{t-1}$, question:q, answer:e_t) tuples) used to train the guided systems. For more details on the data statistics and experimental settings, please refer to the Appendix A.1.1 and A.2. The code will be released upon acceptance.

5 Evaluation

We assess the utility of the question-guided event language modeling in terms of four aspects: (i) Diversity: are they able to generate diverse sets of events that relate to a scenario? (ii) Control: do they generate events involving the specific entities in desired roles? (iii) Sequence modeling ability: how well can it predict observed events?, (iv) Interactive Utility: do users generate better sequences when using the model to collect events that fit a scenario?

5.1 QGELM Improves Diversity

We want event language models to generate diverse sequences covering different aspects of a scenario. To assess diversity in generation, we first sample multiple sequences from the models. Given a context (starting with a context of length 1), we incrementally generate events by sampling one event at a time until we generate a sequence of a predefined length. We repeat this process to generate multiple sequences. We then measure the diversity of these sequences using Self-BLEU (Zhu et al., 2018), which is the average of the BLEU scores (Papineni et al., 2002) when using one of the generated sequence as the output and the rest as references.

First, we collected test instances that had a context length of one, which amounted to 938 instances. The models were used to generate five sequences of lengths one through ten (i.e., five sequences of length one, five of length two and so on). For EGELM and QGELM we randomly choose an entity/question to generate the next event. For each model, we then compute the Self-BLEU score of its five sequences of a specific length. Figure 4 shows the average Self-BLEU over the test instances when generating sequences of different lengths. Lower Self-BLEU scores represent more diverse sequences. Self-BLEU of question-guided outputs are lower compared to that of the other two models showing improved diversity. With longer sequences Self-BLEU increases for all models as there is more potential for overlap. However, the question-guided model retains higher levels of diversity compared to the rest. Standard event language modeling tends to cover the same types of events across different samples and to some extent conditioning on entities helps improve this to a small degree. Conditioning on the questions, however, yields significant gains in diversity showing promise for improved coverage of scenarios.

5.2 QGELM Controls for Entity Roles

To quantify controllability, we introduce a metric that measures how many times a system is capable of generating an event in which a specified entity of interest is present. We generate a fixed number of outputs from each model under two decoding strategies: sampling and beam decoding. We report the percentage of times the specified entity or a coreferent mention[2] of it fails to appear in the outputs. In addition to ELM, we also use NeuroLogic decoding (ND) (Lu et al., 2021), a state-of-the-art controllable generation system as a baseline.

The top block of Table 1 (Any presence) compares the models on the number of times they failed to generate an event with the specified entity within a beam of size 40 or within 40 sampling attempts

---

[2] A mention is considered coreferent to the input noun phrase if the mention and original noun phrase appear in the same coreference cluster extracted from the sampled sequence of events which includes the original context event.

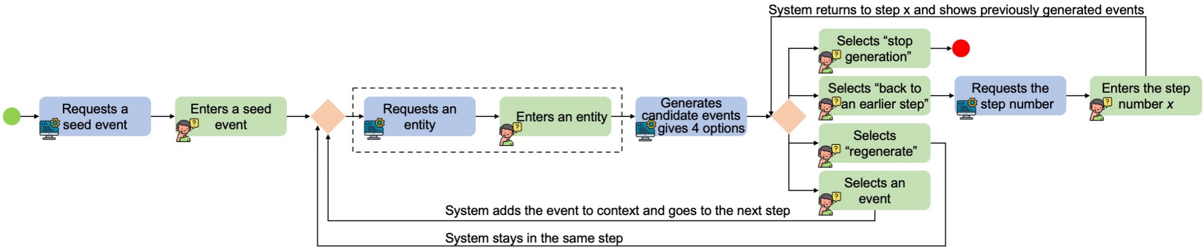


Figure 5: The overview of the interactive schema generation tool. The dashed box is only used for QGELM. The interaction starts by system asking for a seed event and the user entering an event. For QGELM, the user is asked for an entity of interest. This part is shown with a dashed box (this part is not needed for ELM). The system then samples 4 events (If QGELM, it automatically creates questions for the given entity and generates two responses per question.) These 4 events will be presented to the user where they can select an option out of 4 choices. 1. Select one of the given events. 2. Ask the system to generate a new set of events. 3. Return to an earlier step to explore a different path by entering the step number and 4. Stop the generation for the given seed.

(%Fail), and the average number of events that had to be sampled to see the entity or its coreferent mention in the output when sampling (Avg #samples). With beam decoding, ELM performs the worst since it has no control over the generation. NeuroLogic decoding does a better job at searching for events that meet the entity constraints in the model's beam. EGELM which is trained to generate events with the given entity does better even with standard beam decoding. QGELM fares even better outperforming all methods by a significant margin. With sampling, ELM works better but still fails to produce events with the input entity in more than a quarter of the cases. Both entity and question guided models respond to the input control, almost always yielding events with the input entity and much earlier in the ranked list.

The bottom block of Table 1 (Role specific presence) compares the models when we are looking for events where the input entity is expected to appear in a specific role. We use the same dependency-based heuristics we described in Section 3.2 for determining the role of an entity in a given event. With beam decoding, NeuroLogic decoding (ND) is worse than QGELM with a larger margin since it can only be constrained to generate the entity but not in ***specific*** position and role. The larger gap here shows the superiority of our question control codes to not only generate an entity but also to generate it in a specific role. ELM fares even worse, with more failures and requiring even more samples to generate the entity in the specified role. Also, as expected, EGELM has a larger gap compared to QGELM which is trained to account for the roles in which the entities appear.

Note that the coreference resolution system and the dependency parser are not perfect and therefore our heuristics for deciding both when a entity is mentioned in an event and when it appears in a specific role can be faulty. A manual inspection of the outputs for 50 entities across all systems showed that the heuristic is more than 75% accurate and the mistakes are uniform across the models.

5.3 QGELM is a good event LM

How does question-guided training affect the raw ability to generate "standard" event sequences? Predicting observed events in a discourse can be seen as a downstream evaluation. To assess this, we compare the perplexities as well as the narrative cloze task accuracy of the models using the event sequences we observe in the test set. The results are presented in Table 2. One way to turn the question guided-model into a standard event language model is to marginalize its probabilities for outputs over all possible questions we can ask at every step (marked as margin in the table): $P(e_n \mid C) = \sum_{q \in Q} P(q \mid C) P(e_n \mid q, C)$, where we assume a uniform prior distribution $P(q \mid C)$ over questions that can be asked. Similarly, for the entity guided model we can marginalize over the set of noun phrases in the most recent event in the context (this setting is similar to how we created the training instances). While this allows for fair comparison as a standard event language model, this is likely a lower bound for the model's ability for its intended use as a controllable model.

We compute the average per token perplexities for the instances of the test set (including (C,Q,A) tuples) which are shown in Table 2. Although the marginalized performance is lower because the model is forced to generate the event given sub-

	Crimes	Outbreak	Disaster	Kidnapping	Cyberattck	IED	Shooting	Conflict	Overall
ELM	10.71	16.39	23.81	8.00	11.11	13.89	21.05	**30.56**	16.24
QGELM	**28.57**	**27.87**	**33.33**	**14.00**	**16.67**	**16.67**	**23.68**	25.00	**23.22**

Table 4: Percentage of the overlap between system-generated events and manually curated schemas. Overall, the QGELM system generates more diverse set of events, therefore having a higher recall (higher percentage of overlap) compared to the ELM (for all domains except one). More details can be found in Appendix A.3.4

Domain	ELM	QGELM
mass shooting	police evacuated the surrounding buildings after the shooting. the area resuscitated as soon as officials arrived. the area in search of survivors. a police helicopter carrying three officers fired at the shooting center. the police still searching for victims. the police found the bodies of four people. the bodies of two of them found in a second car.	police evacuated the surrounding buildings after the shooting. he identified as a man. a detective questioned the suspect. the police identified the gunman. the officers heard noise from the building. they spoke with him. he moved in an apartment that was recently renovated. they worked on the scene to identify the gunman. the officers involved in an investigation. he notified about the shooting. the police found no weapon in the apartment. officers searched two buildings in the immediate area near the building. the shooting began on saturday. police officers questioned him about the shooting. The officer called victims at 6:45. the officer questioned again in the area. the shooter shot two times. the shooting began with three or four shots of the gunman. he started firing slowly his gun
kidnapping	the kidnapper ambushed the target. the kidnappers shot in the arm. they shot the men. he still not identified by the authorities. the other men also gunshot wounds. the attack left six people in critical condition and his condition.	the kidnapper ambushed the target. the kidnapping triggered by an act of rebellion. he left the scene. the target captured by a surprise attack. the kidnapping of the target essentially a symbolic step in the long struggle. mr. seymour to talk about the kidnapping. the police found a handgun. the hideout an informal area of community groups in rural part of town

Table 5: Generated examples through human interaction. A mass shooting scenario in the first example, can include high level schemas such as planning, occurrence, immediate response, investigation, etc. The QGELM covered more aspects compared to the ELM. As for the QGELM, the entities can easily be tracked based on the questions asked, and therefore there are fewer ambiguous red entities (not clear what/who they are referring to.) compared to ELM in the second example. More examples can be found in Appendix A.3.6.

optimal conditions, we see that with appropriate guidance (the question from the dataset itself), the perplexity of the true event is lower. Moreover, QGELM achieves the lowest perplexity in these settings, indicating that its control is most fine-grained and allows users the highest degree of control. We also computed the accuracy of the cloze task in which each system has to correctly predict the gold output from a fixed set of events. Following Weber et al. (2018b), for each article from the test set, we randomly select an instance (context, question, answer) and then for each instance we create 5 confounding events as answers by randomly selecting instances from other documents. The task would be then picking the correct option from the given six choices (gold+confoundings). The results show that the accuracy trends align with the perplexity trends indicating that QGELM is comparable as a event language model to the baselines we build on.

5.4 Interactive Schema Generation

Event language models can be used to generate event sequences that approximate a schematic description of how events typically happen in certain scenarios (i.e. event schemas) (Weber et al., 2018a,b; Pichotta and Mooney, 2016).

Task setup We evaluate the utility of our new question-driven models when used in an interactive system, where a user collects a set of output events from the model that they think best describes a scenario. The overview of the system is depicted in Figure 5. We manually selected 35 seeds from 8 common domains and asked 7 users (graduate students from NLP and non-NLP labs) to spend 4 minutes interacting with each system. They were asked to generate sequences of events for given seeds using ELM and QGELM systems in a randomized order. For each scenario, the user is given a seed event and is tasked with collecting a set of events that *best describe the scenario*. At each step the user is presented with a set of generated events from the model and the user selects one of the events to add it to their collected set. The added events optionally become part of the conditioning context for more events. The user has the option to either regenerate events for the same context or go back and choose a subset of context from which to generate events. For the guided model, the events are generated by conditioning on questions. Since this is a timed practice, instead of asking the users to type in a full question, we only ask them to provide the system with an entity of interest. The system then automatically forms two questions (agentive, non-agentive) with the entity and outputs a mix of the events generated with all the questions. Additional details of this study including the settings, motivations for the timed version as well as instructions for the users are listed in Appendix A.3.

At each step the user selects the best event that meets the following criteria: (i) *Sensibleness*: whether the generated event is grammatically correct, sensible and easy to understand. (ii) *Uniqueness*: events do not duplicate each other and describe different subevents. (iii) *Relatedness*: events are related to the domain. (iv) *Typicality*: the events are quite common for things in this domain and not too niche. For each event, the users make a binary judgment on whether each criteria is satisfied. If any of the criteria is not satisfied then the event is not selected. If no event meets all these criteria then the user either regenerates from the same context or moves to an earlier context. Each user interaction with the system results in a sequence of events in the form of a tree, as users might have explored different paths by selecting different events at each step. Table 5 shows example outputs.

Analysis Table 3 shows that with the QGELM based interactive system, users accept more of the system suggested events, which means that more events meet the criteria we set for good events. They ask for fewer resampling steps, require fewer returns to earlier steps and thus having fewer reject steps or wasted steps in their interaction. They also produce longer descriptions of the scenarios with higher tree-depth i.e. the length of the longest sequence they generated within a domain.

We further analyze the output generated using the interactive system to assess their utility in creating complex schematic knowledge. The seeds we use come from 8 different domains relevant to the intelligence analysis community. For each domain we have access to manually curated schemas created by ten language experts in collaboration with the intelligence analysis experts over multiple days. An example of such schema is presented in Table 9 in Appendix A.3.5. We used these schemas as references and compared the percentage of overlap between system generated events selected by the users (within the four minute interaction) and the events in the reference schemas. The results in Table 4 show that both automatic systems can generate events that are expected to describe certain scenarios, however, the events generated by QGELM tend to have higher recall compared to the ELM (in 7 out of 8 domains).

6 Conclusion

Controlling event language models to generate events with respect to the participants is not trivial. We propose a simple yet effective question-guided approach that learns to generate events by not only conditioning on the events but also on specific entity-based questions of interest. Our empirical analysis shows that this approach can be used to generate more diverse sequences with better coverage and controllability allowing for better modeling of complex scenarios.

7 Limitations

One of the limitations of our proposed approach is the coverage of the entity roles. We have used two broad categories of roles, mainly agentive (subject) and non-agentive (object) roles, however, there can be more fine-grained semantic roles for the participating entities in the events such as agent, patient, theme, manner, etc. Considering this taxonomy of semantic roles can lead to finer-grained questions which might lead to even richer descriptions of the scenarios. Also, the human evaluation setting is limited since it is timed and users can not explore the models to their fullest extent. However, our analysis of the systems, when not timed, shows even a higher margin in terms of performance with the QGELM model.

8 Ethics Statement

The models presented in the paper make use of the existing pretrained systems that train on large collections of data and are known to inherit biases that are existent in the training data. The event language models we train are also susceptible to these biases, which can result in generation of event sequences with these biases.

Acknowledgments

We thank the anonymous reviewers for their insightful feedback and suggestions. This material is based on research that is supported by the Air Force Research Laboratory (AFRL), DARPA, for the KAIROS program under agreement number FA8750-19-2-1003. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes.

References

Niranjan Balasubramanian, Stephen Soderland, Oren Etzioni, et al. 2013. Generating coherent event schemas at scale. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1721–1731.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Rujun Han, I-Hung Hsu, Jiao Sun, Julia Baylon, Qiang Ning, Dan Roth, and Nanyun Peng. 2021. ESTER: A machine reading comprehension dataset for reasoning about event semantic relations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7543–7559, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. Ctrlsum: Towards generic controllable text summarization. *arXiv preprint arXiv:2012.04281*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Chenyang Huang, Osmar Zaïane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic dialogue generation with expressed emotions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 49–54, New Orleans, Louisiana. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mahnaz Koupaee, Greg Durrett, Nathanael Chambers, and Niranjan Balasubramanian. 2021. Don't let discourse confine your model: Sequence perturbations for improved event language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 599–604, Online. Association for Computational Linguistics.

Heeyoung Kwon, Nathanael Chambers, and Niranjan Balasubramanian. 2021. Toward diverse precondition generation. In *Proceedings of* SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 160–172.

Manling Li, Sha Li, Zhenhailong Wang, Lifu Huang, Kyunghyun Cho, Heng Ji, Jiawei Han, and Clare Voss. 2021. Future is not one-dimensional: Graph modeling based complex event schema induction for event prediction. *arXiv preprint arXiv:2104.06344*.

Manling Li, Qi Zeng, Ying Lin, Kyunghyun Cho, Heng Ji, Jonathan May, Nathanael Chambers, and Clare Voss. 2020. Connecting the dots: Event graph schema induction with path language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 684–695, Online. Association for Computational Linguistics.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, Online. Association for Computational Linguistics.

Mehdi Manshadi, Reid Swanson, and Andrew S Gordon. 2008. Learning a probabilistic model of event sequences from internet weblog stories. In *FLAIRS Conference*, pages 159–164.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL)*.

Julian Michael and Luke Zettlemoyer. 2021. Inducing semantic roles without syntax. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4427–4442, Online. Association for Computational Linguistics.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generationusing energy language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 401–415, Dublin, Ireland. Association for Computational Linguistics.

Ashutosh Modi. 2016. Event embeddings for semantic script modeling. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 75–83.

Raymond J Mooney and Gerald DeJong. 1985. Learning schemata for natural language processing. In *IJCAI*, pages 681–687.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Karl Pichotta and Raymond J Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Valentina Pyatkin, Paul Roit, Julian Michael, Yoav Goldberg, Reut Tsarfaty, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1429–1441, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mehdi Rezaee, Francis Ferraro, et al. 2021. Event representation with sequential, semi-supervised discrete variables. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Paul Roit, Ayal Klein, Daniela Stepanov, Jonathan Mamou, Julian Michael, Gabriel Stanovsky, Luke Zettlemoyer, and Ido Dagan. 2020. Controlled crowdsourcing for high-quality QA-SRL annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7008–7013, Online. Association for Computational Linguistics.

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Roger C Schank and Robert P Abelson. 1977. Scripts, plans, goals, and understanding: an inquiry into human knowledge structures.

Michael Schmitz, Stephen Soderland, Robert Bart, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534.

Raphael Shu, Hideki Nakayama, and Kyunghyun Cho. 2019. Generating diverse translations with sentence codes. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827, Florence, Italy. Association for Computational Linguistics.

Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. 2018a. Event representations with tensor-based compositions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nathanael Chambers. 2018b. Hierarchical quantized representations for script generation. *arXiv preprint arXiv:1808.09542*.

Nathaniel Weir, João Sedoc, and Benjamin Van Durme. 2020. COD3S: Diverse generation with discrete semantic signatures. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5199–5211, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *SIGIR*.

A Appendix

A.1 Data Processing Details

QGELM uses the data in the form of (Context, Question, Answer) tuples. The details of the data processing is outlined in Algorithm 1.

Algorithm 1 Data Processing

```
Input: $D$, Document, Output: $T$, List of (Context, Question, Answer) tuples
extract OpenIE event tuples $E$ from $D$
find co-referring clusters $Clusters$ from $D$
for $e_i$ in $E$:
    find all nps $N_{e_i}$
    for each $np$ in $N_{e_i}$:
        generate $Q_{np}$
        for each $q_j$ in $Q_{np}$ for $np$ with role $r_j$
            find an event $e_k$ $(k > i)$ with $np$ with role $r_j$
            add $(e_1..e_i, q_j, e_k)$ to $T$
```

We use AllenNLP (Gardner et al., 2018) for coreference resolution to find the clusters in a document and the spaCy (Honnibal et al., 2020) dependency parser to identify all the noun phrases and their roles within an event.

A.1.1 Data Statistics

We initially extract Open IE event tuples from $37,924$ articles from the 2007 portion of the NYT Annotated Corpus using Ollie (Schmitz et al., 2012). All the extracted events from a single document are concatenated to form a single event sequence for that document. Therefore, we end up having $37,924$ event sequences with average length of 27 events.

Then, we run the data processing algorithm on the extracted sequences to generate (Context, Question, Answer) tuples. Table 6 shows the data statistics of the dataset.

split	Total	Q1	Q2	Q3
All data	762,004	466,757	188,300	106,947
Train	752,004	460,573	185,582	105,579
Dev	5,000	3,091	1,212	697
Test	5,000	3,093	1,236	671
Common Test	18953	11,712	4,412	2,829

Table 6: Data Statistics of the (Context, Question, Answer) tuples for different types of questions. Q1 refers to *what else happened?*, Q2 is *what else did np do?* and Q3 is *what else happened to np?*

A.2 Experimental Settings

A.2.1 Input/Output format

For the ELM, the input will be the context and the output will be the next event. For EGELM and QGELM, the input will be the concatenation of the context and the entity/question, separated by [SEP] token. Since the input can be more than 512 tokens (in case of long contexts), we need to truncate the input. Truncating the input from its end (for EGELM and QGELM) will result in input sequences without entity/question. To avoid this, we instead truncate from the beginning of the input sequence by removing the earlier events in the context. We remove events and not tokens from the context until its length is within the model input size. The length of the output is also fixed at 50 tokens.

A.2.2 Systems Details

Our systems finetune a pre-trained T5-base model and tokenizer with the implementation from Huggingface library (Wolf et al., 2020). Adam optimizer (Kingma and Ba, 2014) is used with an initial learning rate of $6.25e-5$.

We use a batch size of 4. Each training epoch takes almost 24 hours to run. We use the dev set for early stopping. All the systems will converge after 3 epochs.

A.3 Evaluation

A.3.1 Interactive Evaluation Setup

The details of the settings of the interactive evaluation are presented in Table 7.

A.3.2 Interactive Evaluation Design Choices

Some design choices for this evaluation are constrained by practicalities that would ensure a fair comparison. We initially had used an untimed version where users could interact with the system indefinitely. However, we found that users spent different amounts of time working, making it near impossible to do fair comparisons across systems. Further, users found it difficult to retain focus over longer periods of time.

As for using an entity of interest instead of typing questions, typing questions at every stage induces a burden on the user and introduces variance because of typing speeds. But note that even though they only select an entity, the roles are used as part of the questions we generate for the entity. Half of the generated answers are with the entity in the subject role and the other half are in the object role. Users can pick whatever role they want to explore.

Parameters	**Values**
number of seed events	35
number of users	7
number of seeds per worker	5
allotted minutes	4
number of generated events per step	4
number of domains	8
domains: disease outbreak, cyberattack, ied, international conflict, kidnapping, disaster, mass shooting, financial crimes	

Table 7: The parameters of the human generation task.

Domain	**# seeds**	**# gold events**
Crimes	4	28
Outbreak	6	61
Disaster	5	42
Kidnapping	4	50
Cyberattck	3	54
IED	3	36
Shooting	5	38
Conflict	5	36

Table 8: The statistics of the human evaluation in terms of number of seeds and the number of gold sequence events.

A.3.3 Instructions for Users

Below, you can find the guidelines that were provided to the users prior to starting the task.

User Manual This study is aimed at evaluating the capabilities of the event language models in generating a sequence of events with their participating arguments that can be used to describe a scenario. For each scenario, a seed event will be given. Using the seed event, you can start generating the sequence incrementally by selecting the best event at each step based on the following set of criteria:

At each step select an event that is: grammatical/understandable non-redundant or unique: events do not duplicate each other and describe different subevents on-topic: events are related to the domain and not unrelated typical: the events are quite common for things in this domain and not too niche (for example, earthquake could cause a nuclear reactor to meltdown but it's not common)

You will use two different systems to do this task. For one system you only need to select the events at each step while for the other, at each step you initially type an entity of interest and then pick the best event. This task will be timed, and you keep interacting with the system for 4 minutes, exploring different paths and entities. Once the time is over, the generation stops automatically.

User Interface: Once you run the commands, the system will load the pretrained models which will take a few seconds and then it will ask you to enter the seed event (The seeds will be given to you). Then you need to follow the prompts at each step for the allotted time. At each step there will be 4 actions you can take: Choose a preferred event generated at that step. You will be shown a set of events from which you can pick the best one according to the above criteria. Regenerate events for the last step. If you feel none of the generated events satisfy the criteria, you can choose this option so that the system will generate a different set of events for this step. Please use this option if NONE of the generated events satisfy the criteria. Choose an earlier step to return to. If you get stuck in one path and cannot generate events, you can choose to go back to an earlier step and continue the generation from a different path. Once you choose to return, the current set of events will be saved. Stop generation for the given seed event. If you think the system has generated enough events to describe the scenario or if it is no longer generating good events, then you can decide to stop the generation even if the allotted time is not yet over.

Notes regarding the entities: If you are asked for an entity, you can have the following things in mind: You are given an initial set of entities that are relevant to the given scenario. You can pick entities from this set, think of other entities that might be relevant, pick entities from already generated events or just select 'none' if you cannot think of an entity. You do not need to use all the given entities and you can choose the same entity if you think that is a main entity in the scenario or if you are interested in knowing more about that entity. You can generate a sequence which is centered around one specific entity. For instance, you can have an event sequence like this for the earthquake scenario where 'earthquake' is an argument in all the generated events: earthquake struck city, earthquake magnitude measured on scale, earthquake killed people, earthquake injured people, earthquake damaged buildings, earthquake disrupted services,...

A.3.4 System-generated events VS Manually-curated schemas

We tried to show the plausibility of the system-generated events by comparing the system outputs with schemas that are curated by a group of experts for different domains. We initially used a number of seed events from these domains and provided the users with these seeds to interact with the system. Then for all seeds from a single domain, we grouped all the generated events and measured the amount of overlap with human-written schemas. The statistics of this experiment is presented in Table 8. An event is considered to have an overlap with an event from the gold set if it either shares the exact predicate or a predicate with similar meaning. To do this, we provided a user with the list of system-generated events (not knowing which system this is coming from) and asked them to find the mappings between the gold set and the generated set. We then counted the number of events that are considered as overlapping with the gold events.

A.3.5 Manually curated schemas

Real-life scenarios can be described with a sequence of events and their relations. Manually curated schemas represent the events that can unfold a scenario in a hierarchical structure. The events in this structure can be either primitive or non-primitive, depending on whether they can be further expanded into additional events. We use the term “schema” to refer to the non-primitive events. Table 9 shows an example of such schemas for the disaster domain. As can be seen, the events are represented in multiple levels. For the sake of the evaluation conducted in this work (to make the comparison of the system-generated sequences with curated schemas more compatible), we consider the flattened representations of the schemas which consists of concatenating all the primitive events into a single sequence of events.

We also did the comparison on predicate level. The reason is that the system-generated events are instantiated events with specific arguments as the models are trained on news articles whereas the curated schemas are generalized forms of events.

A.3.6 System generated event sequences

Table 10 shows examples of generated event sequences through users interaction with the interactive tool described in section 5.4.

Schema name	Level	Events
natural disaster progression	0	detection and tracking* preparations* damages* immediate responses* rescue organization* rescue* rescue outcome* economic assistance* rebuild damaged property*
detection and tracking	1	scientists detect warning signs scientists track progress scientists assess threat scientists warn public media broadcasts information
preparations	1	government announce order people buy supplies preparations outcome*
damages	1	disaster hurts person/people disaster kills person/people disaster destroys buildings disaster damages infrastructure disaster causes food/water shortage disaster causes power outage
immediate responses	1	government counts deaths government sends equipment government estimates damage government requests aid person obtains aid person sheltered
rescue organization	1	government hold session to plan rescue government command rescue teams rescue teams travel
rescue	1	government hold rescuers locate survivors rescuers assess rescuers triage survivors
rescue outcome	1	victims found dead recovery*
economic assistance	1	government request help government estimate loss government make assistance policies people apply for assistance government gifts aid government gives loans
rebuild damaged property	1	people contact insurers people repair damage people rebuild
recovery	2	rescuers save survivors rescuers transport survivors rescuers treat survivors
preparations outcome	2	evacuation* people shelter in place
evacuation	3	people hear evacuation order rescuer evacuate people people leave affected area

Table 9: Manually curated schemas. Each domain is represented through a set of high-level schemas which in turn are expanded into multiple events. Here you can see the events of the disaster schema which are represented at different levels. Events with * are non-primitive events that are expanded.

Domain	**Seed**	**System**	**Generated event sequence**
international conflict	country agreed to sign the accord.	ELM	he country and several members of its foreign relations committee negotiated a peace agreement as part of the agreement. the agreement called for more cooperation. the terms of the agreement announced at the start of a yearlong talks. the talks took place in 2002. the deal still subject to approval. the agreement formally announced on saturday
		QGELM	peace take a major step. the accord a sign of change. the accord not necessarily end war. the agreement calls for a gradual suspension of all military combat and police operations. the accord also requests immediate withdrawal of troops
cyberattack	the cyberattack disrupted services	ELM	the network crashed after a new attack. the virus affected internet connections. the virus infecting computers that have computer processors. two new attacks launched on thursday. the virus also infects networks that used the same basic operating system. two new attacks triggered an attack that affected 1 million internet users on tuesday. both attacks exposed another email containing information about a security company. one email message posted by an employee of a security company. security engineers investigating the e-mail messages. the other attack not affect the internet connection
		QGELM	the attack also infected some computer systems. at least 60 computers affected by the cyberattack. the attacks continued across an industrial network of 8. the attack started in friday. the attacks hit at least five businesses and four hospitals. the network failed in this attack. the attack attributed to an increase in unreported customer data from two years earlier to more widespread attacks. one problem the nature of these attacks. the attacks linked to an ”antiviral attack. the attackers focused to eliminate part of microsoft’s core operating system that would detect the attack, and to prevent it from becoming more sophisticated. … … … the attacks aimed at businesses that are vulnerable to any type of attack. the attacks caused significant disruptions for internet service providers. they trying to improve the security. the attack involved a variety of devices. the hackers asked for information. the security department declined to comment about the attack.
disaster	the heavy flooding disrupted the services.	ELM	the damage, which is estimated to cost about $2.5 billion annually due to flooding and rainy weather largely a product of human error. the department of agriculture forecasts the flood. two million people killed in the flooding. a few roads halted on the west side. the power grids closed down in some areas. the flooding also damaged roads and bridges. the department of transportation and emergency management put forward a plan. most of the roads closed in new york. the department also announced at least 17 routes halted in the same area as they were closed on oct. 26
		QGELM	the flooding resulted in a series of delays. the flooding also caused problems. the federal emergency management agency and the emergency maintenance team also monitoring damage from the flood. the flooding continued as far as tuesday. the flood already brought over from the south. the new york city fire department dispatched officers and engineers to help in the flooding. the flooding worsened on the west side. the flood occurred along the west side. about 3,000 people killed by the flooding. the flooding occurred north of manhattan around 1:55 p.m.. the water caused by another flooding a series of conditions. the flooding followed an investigation of a suspected bribery. the flooding still also affected the emergency management system. officials not expecting much rain until tuesday. the firefighters remained on manhattan. firefighters and crews sent out to assist residents with their tasks. the officials not identify a cause of the flooding. the flooding came after a storm that had swept up from manhattan. the rain caused by a river of soil. firefighters began an investigation into possible obstructions. the emergency crews working in all locations. they expected about eight feet of water to be there by this morning. the flood affected at least seven other areas. the flooding caused the agency to cut off access to the subways. the agency attributed much of the flooding to human error

Table 10: Generated examples through human interaction with the system.

Uncovering Implicit Inferences for Improved Relational Argument Mining

Ameer Saadat-Yazdi
School of Informatics
University of Edinburgh
ameer.saadat@ed.ac.uk

Jeff Z. Pan
School of Informatics
University of Edinburgh
j.z.pan@ed.ac.uk

Nadin Kökciyan
School of Informatics
University of Edinburgh
nadin.kokciyan@ed.ac.uk

Abstract

Argument mining seeks to extract arguments and their structure from unstructured texts. Identifying relations (such as attack, support, and neutral) between argumentative units is a challenging task because two units may be related to each other via implicit inferences. These inferences often rely on external commonsense knowledge to discover how one argumentative unit relates to another. State-of-the-art methods, however, rely on predefined knowledge graphs, and thus might not cover target pairs of argumentative units well. We introduce a new generative approach to finding inference chains that connect these pairs by making use of the Commonsense Transformer (COMET). We evaluate our approach on three datasets for both the two-label (attack/support) and three-label (attack/support/neutral) tasks. Our approach significantly outperforms the state-of-the-art, by 2-5% in F1 score, on two out of the three datasets with minor improvements on the remaining one.

1 Introduction

In argumentation, such as in a debate, it is important to understand the key arguments put forward and how the arguments relate to each other based on a specific context. Ideally, we would like to empower systems to extract arguments and the relations between arguments automatically. Such systems could be used to aggregate opinions (Cocarascu et al., 2019), participate in debates (Slonim et al., 2021), or they could assist humans in making informed decisions by considering different points of view.

Toulmin (1958) defines an argument as a claim backed by one or more grounds (or premises) based on a warrant (i.e., the inference link between the grounds and the claim). For example, consider the following two sentences: *Drunk driving hurts innocent people. Therefore, drunk driving is wrong.* Based on Toulmin's model, we have the claim *drunk driving is wrong*, supported on the grounds that *drunk driving hurts innocent people*. In what follows, we will call the claim and grounds *argumentative units* (AUs). To justify the claim, the author relies on the reader's ability to uncover the implicit warrant that *hurting innocent people is wrong* which connects the grounds to the claim. While Toulmin's warrants model only arguments in which the units are connected by support relations, we extend the definition to also allow for attacks and neutral relations. We do this to enable us to apply the notion of warrants to the relational argument mining task, as proposed by Carstens and Toni (2015), to better identify attack/support/neutral relations between pairs of AUs.

Finding the correct warrant is a challenging task that requires deep reasoning, which pre-trained language models struggle with (Helwe et al., 2021). However, by iteratively generating knowledge, we can chain together inferences to get a series of causes and effects that connect two AUs together similar to chain-of-thought reasoning (Wei et al., 2022). We propose the use of the Commonsense Transformer (COMET) (Hwang et al., 2021) which is equipped with social commonsense to generate warrants as a series of inferences. The knowledge contained in COMET is qualitatively different from other knowledge sources in that it contains normative commonsense. This depends upon the culture and beliefs of various groups compared with static commonsense knowledge graphs (KGs), such as ConceptNet (Speer et al., 2017), which focus on globally accepted knowledge. For this reason, we suggest that COMET is better able to identify the reasoning behind human arguments and that a generative approach based on COMET is better suited to finding warrants compared to a static knowledge graph as proposed by Paul et al. (2020).

Section 4 introduces our approach to test this hypothesis, where we articulate an algorithm,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2484–2495
May 2-6, 2023

ArgCon (Section 4.2), to generate warrants between a pair of argumentative units and use these warrants as additional input to the classifier of argument relations to evaluate whether this improves the model's ability to distinguish between attack, support or neutral relations. We then compare the use of three external knowledge sources: ConceptNet$^+$ the commonsense knowledge extracted by Paul et al., the knowledge generated by ArgCon, to obtain novel commonsense inferences (Section 4.2), and ArgCon + Link which enriches the generated knowledge with additional relations via a link prediction model (Section 4.3).

In Section 5, we compare our work to (Paul et al., 2020), which is the only other work we are aware of to address the task of relational argument mining by uncovering commonsense relations. We show that our method for generating warrants with COMET outperforms their method with a 1-2% increase in F1 scores on two datasets. We further evaluate our model on a three-class dataset, which includes an additional neutral relation, and witness a 5% improvement of the F1 score over a RoBERTa baseline. Hence, we set a new benchmark for the three-class task in relational argument mining. We share our findings in Section 6 and conclude with our future directions in Section 7.

2 Background

In this paper, we investigate the effectiveness of COMET (Bosselut et al., 2019) for generating implicit warrants that link arguments together. COMET is a generative model which has been fine-tuned on a human-authored commonsense knowledge graph ATOMIC (Sap et al., 2019). By training on ATOMIC, COMET extends the knowledge captured by the knowledge graph to new domains and is able to synthesize new commonsense knowledge based on knowledge captured during pre-training. The model is able to generate knowledge along multiple relations- a few examples are shown in Table 1. COMET takes as input a sentence and a relation along which to generate inferences. For example, given an input *PersonX's house was foreclosed* and relation *CausedBy*, COMET would generate *PersonX failed to pay their mortgage*.

Some successful applications of COMET include the generation of potential outcomes of events for abductive reasoning (Bhagavatula et al., 2020), question answering (Branco et al., 2021) and text generation (Guan et al., 2020). Closely related to our work, Chakrabarty et al. (2021) shows that COMET can be used to generate logically sound premises for incomplete arguments. While previous works have considered using COMET for single-hop inference, we found that for our task of relational argument mining, often multiple inference steps are needed to get from the first argumentative unit to the second; we address this challenge in Section 4.2.

3 Related Work

Various approaches have been proposed to infer argumentative structure from unstructured text. These include Long-Short Term Memory (LSTM) models (Cocarascu and Toni, 2017; Paul et al., 2020), pre-trained transformers (Ruiz-Dolz et al., 2021) and logic programming (Jo et al., 2021). Pre-trained transformers, in particular, have been shown to perform exceptionally well on this task with no additional feature engineering (Ruiz-Dolz et al., 2021; Fromm et al., 2019); this suggests that the introduction of external knowledge encoded within the transformers due to pre-training on large corpora is necessary to make significant progress on this task. Of these works, only Fromm et al. (2019) makes use of external knowledge to identify the stance of arguments towards a given topic. This differs from our task in that we wish to identify the relationship between pairs of AUs and so have the added challenge of finding paths that link the AUs together.

Commonsense knowledge has proven repeatedly to aid in tasks such as natural language inference (Wang et al., 2019) and question answering (Lv et al., 2020). The majority of these works focus mainly on deriving knowledge from ConceptNet (Speer et al., 2017); however, a key limitation of knowledge graphs (Pan et al., 2016) is their inherent incompleteness.

Recent trends in argument mining have shown a move towards incorporating external knowledge for various argument-mining tasks. Two works explore the use of automatically constructed knowledge graphs extracted from Wikidata and Google search, which, however, fail to achieve better results than BERT with no additional knowledge (Fromm et al., 2019; Li et al., 2021). On the other hand, a work using LSTM-based models for knowledge-enhanced argument mining with ConceptNet$^+$, shows that the introduction of knowledge indeed improves performance upon their baselines (Paul et al., 2020).

One of the key limitations of existing works is their use of static knowledge graphs which are unable to handle arbitrary pairs of AUs due to their limited coverage. To address this issue, we use COMET to generate conmmonsense inferences on-the-fly to connect AUs and therefore generalize to new domains.

4 Our Approach

In this section, we describe the design of our model as well as our procedures for obtaining knowledge on the fly with COMET and enriching the knowledge with additional links.

4.1 KE-RoBERTa

Given a pair of Argumentative Units, AUs, we aim to predict the relation between these AUs by the use of external knowledge relevant to this pair. Ruiz-Dolz et al. (2021) show that RoBERTa is a strong baseline for the classification of argument relations when compared to other transformer-based architectures. Hence, we introduce KE-RoBERTa, which is a knowledge-enhanced RoBERTa-based model as depicted in Figure 1.

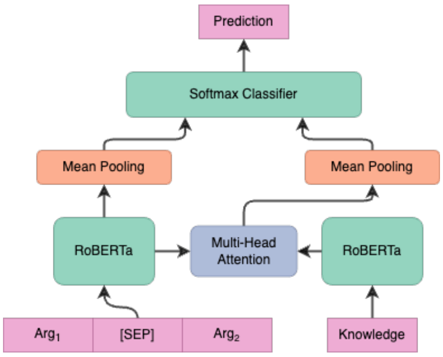


Figure 1: KE-RoBERTa, our proposed Siamese model for Commonsense Knowledge injection with Arg_1 and Arg_2 being the input AUs

In order to inject knowledge into RoBERTa, we use a Siamese model to separately encode the two AUs as well as the commonsense knowledge much like the S-BERT model (Reimers and Gurevych, 2019); in other words, the parameters of the two RoBERTa blocks are shared. The model then combines the two representations by concatenating them together. We also introduce a multi-head attention mechanism (Vaswani et al., 2017) to attend to relevant information and block out noise. We use mean pooling to combine individual token representations before concatenating the result. This allows us to use a fix-sized feed-forward network to compute predictions from the combined AU and knowledge representations.

The notion of knowledge is loosely defined in the literature but can be viewed as a set of beliefs about the world that approximate truth. In our case this knowledge is represented as the set of all knowledge graph paths that could connect two AUs. While there are many ways to represent knowledge graph paths, in this work, we convert each knowledge graph relation into its natural language equivalent and concatenate the nodes and relations together to form a sentence in natural language. For each pair of AUs we then concatenate all possible paths together in order to input the knowledge into a transformer. The source of knowledge graph paths we refer to in the following sections as ConceptNet⁺ combines knowledge from ConceptNet, and WordNet, with additional links predicted by neural networks; ConceptNet⁺ is proposed by Paul et al. and the extracted knowledge was provided to us by the authors. Since the authors did not experiment on the three class task we did not have ConceptNet⁺ relations for M-Arg.

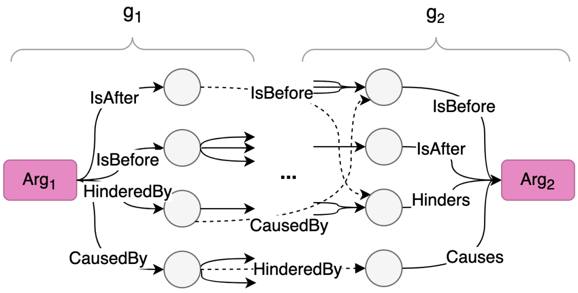


Figure 2: Knowledge graph generation by ArgCon with link inference enabled. The figure shows how two graphs, g_1 and g_2, are generated from each AU, Arg_1 & Arg_2, and then merged. Dashed lines show additional relations found by Link.

4.2 ArgCon: A new way to generate knowledge with COMET

All previous works using COMET seek to generate a possible outcome, a_2, from a known event, a_1, and a relation. For this task, we seek to invert this and find a path to connect a_1 and a_2. What makes this more challenging is that AUs are often connected by multi-hop inferences and so multiple intermediary outcomes o_i may be needed to connect a_1 and a_2. The types of inference we wish to generate can be represented as:

$$a_1 \xrightarrow{r_1} o_1 \xrightarrow{r_2} o_2 \cdots o_{n-1} \xrightarrow{r_n} a_2$$

where r_i is a relation connecting events/outcomes, and neither the o_i nor r_i are known in advance. To do this, we propose a new technique using an improved version of COMET, trained on ATOMIC$^{20}_{20}$, a larger and more extensive version of the original ATOMIC knowledge graph (Hwang et al., 2021), to generate deep commonsense inference chains that connect two AUs together.

Relation	Inverse
Causes	*CausedBy*
HinderedBy	*Hinders*
isAfter	*isBefore*
isBefore	*isAfter*

Table 1: ATOMIC$^{20}_{20}$ relations (R) used to generate knowledge from COMET with corresponding inversions

ATOMIC$^{20}_{20}$ consists of several classes of relations, including 'causes for a person to perform an action', 'effects of an action on others', and 'event-centred'. In this paper, we focus on 'event-centred' relations to generate knowledge from, which is a subset of the relations contained in ATOMIC$^{20}_{20}$. In Table 1, we show the four relations selected. In Section 5, we will report our results on three existing datasets that include AUs focusing on event-centred knowledge.

Algorithm 1 provides pseudocode for our knowledge generation method *Argument Connector* (ArgCon). Our algorithm takes three inputs, a pair of AUs, (a_1 and a_2) as well as a boolean flag ($wlink$) to indicate whether additional link prediction should be performed as described in Section 4.3. The algorithm then returns the paths that connect a_1 to a_2.

Starting from the pair of AUs, we first initialise two single-node graphs, g_1 and g_2, from a_1 and a_2 (lines 1-2). For each of these graphs, we take the set of their nodes N, and for each node n and each relation r we generate new nodes n' using COMET (lines 8-13). We then update N to contain the set of nodes at the current depth i (line 14) and repeat this process d times until each graph has a tree with a maximum depth of d (line 4). Note that graphs g_1 and g_2 now contain only paths from a_1 and a_2, meaning that a_2 cannot be reached from any other node. In order to get paths from a_1 to a_2, we need to flip the relations in g_2 replace them with their corresponding inverse, as shown in Table 1, and then merge the two graphs wherever they share a common node (line 15). We now have a single merged graph g_m containing a_1 and a_2. If $wlink$ is set to `true`, the Link model will be used to infer additional relations that connect g_1 to g_2 (line 16-18) which are then appended to g_m. The algorithm returns all paths that connect a_1 and a_2 (line 19).

Algorithm 1: ArgCon$(a_1, a_2, wlink)$

```
Data: $R$, the set of relations to consider
Data: $d$, the depth of a tree
Data: COMET, commonsense transformer
Data: Link, our Link model
1  $g_1 \leftarrow$ makeGraph$(a_1)$
2  $g_2 \leftarrow$ makeGraph$(a_2)$
3  forall $g \in \{g_1, g_2\}$ do
4      for $i = 0$ to $d$ do
5          if $i = 0$ then
6              $N \leftarrow g.nodes()$
7          $N' \leftarrow []$
8          forall $n \in N$ do
9              forall $r \in R$ do
10                 $n' \leftarrow$ genNode$(n, r,$ COMET$)$
11                 if $n' \neq NULL$ and $n' \neq n$ then
12                     $g \leftarrow g.addtriple(n, r, n')$
13                     $N' \leftarrow N'.append(n)$
14         $N \leftarrow N'$
15 $g_m \leftarrow$ mergeGraphs$(g_1,$ flip$(g_2))$
16 if $wlink$ then
17     $new\_edges \rightarrow$ Link$(g_1, g_2)$
18     $g_m.append(new\_edges)$
19 return computePaths$(a_1, a_2, g_m)$
```

4.3 Link Prediction Between Nodes

It may not be always possible to find a path between the AUs by using only the merging operation described above. We introduce an additional inference step to identify the relationships between the nodes of the two knowledge graphs.

To achieve this, e trained a link prediction model (Link) to identify the most likely relations that exist between two nodes, which are provided as inputs. Link is a BERT model fine-tuned on the ATOMIC$^{20}_{20}$ training set to predict the relation between two nodes of the knowledge graph. In order to identify unrelated node pairs, we sample 4000 pairs that are not connected in ATOMIC$^{20}_{20}$ and assign them a 'None' label. Table 2 shows the training and validation performance of Link when trained to predict the relations in Table 1 and the 'None' relation.

	Accuracy	Precision	Recall	F1
val	0.91	0.89	0.88	0.88
test	0.90	0.85	0.85	0.85

Table 2: Validation and testing performance of LINK model for COMET link prediction after 5 epochs.

Given a pair of graphs and their corresponding node sets, N_1, N_2, generated by COMET, we use LINK to predict the most likely relationships between the two graphs. Explicitly, we pass all node pairs $(a, b) \in N_1 \times N_2$ to LINK and add the predicted relation to g_{merged} (line 15 in Algorithm 1, $wlink$ is set to 1 to enable further inference) before computing paths. Figure 2 depicts our method when link inference is enabled.

5 Evaluation

In this section, we describe the datasets we have chosen, our choices of parameters for knowledge extraction, our training setup, and the results of our experiments on three datasets.

5.1 Datasets and Knowledge

We consider three datasets for our experiments: the Student Essay corpus (Opitz and Frank, 2019), Debatepedia (Paul et al., 2020), and the M-Arg Presidential Debate corpus (Mestre et al., 2021). We may refer to the datasets as Essay, Debate, and M-Arg for brevity.

The first two datasets were annotated with two labels (attack/support). Student Essay is a collection of argumentative essays written in English by second-language speakers, while Debatepedia contains arguments selected from a wiki of controversial topics and pro/con arguments[1] where controversial topics are discussed by multiple authors. These are the datasets used by Paul et al. (2020) which we use for comparison and we have taken the same dataset splits as used in their experiments. M-Arg consists of transcripts from five presidential debates in 2020; sentence pairs in this dataset were annotated with three labels (attack/support/neutral). Table 3 shows how the data has been split.

		A	S	N
Debate	train	3250	3236	-
	val	1088	1075	-
	test	1035	1127	-
Essay	train	273	2797	-
	val	130	1012	-
	test	91	1009	-
M-Arg	train	94	302	2887
	val	11	40	359
	test	15	42	342

Table 3: Distribution of labels (**A**ttack/**S**upport/**N**eutral) across datasets. While Debatepedia is balanced, the other two datasets show drastic imbalances making the task of detecting minority classes more challenging.

5.2 Experimental Setup

As a baseline, we trained a RoBERTa model[2], with no additional knowledge. We trained our models to minimize the cross-entropy loss on the labeled data with a batch size of 10 and a learning rate of $1e^{-5}$. The models were trained using the same hyperparameters for 10 epochs, and the model with the highest F1 validation score was selected for testing. Our code and the data can be found on GitLab[3].

We used ARGCON with a maximum depth limit of $d = 3$ to generate a list of paths from Arg_1 to Arg_2. Each path is given as lists of nodes and relations, which, by using a simple set of rules, we convert into natural language and concatenate the result. This is common practice when using COMET and the reader can refer to the code-base for more details on how this was done.

5.3 ARGCON vs ARGCON+LINK

We first investigate the proportion of AU pairs for which ARGCON and ARGCON+LINK could generate knowledge for (Table 4). We observe that nearly half of the pairs are not covered by ARGCON while ARGCON+LINK gives an improvement of 30% on Debatepedia and Student Essay and 40% on M-Arg.

5.4 Relational Argument Mining Results

Table 5 shows the results from the classification experiments; the first two rows, (Paul et al., 2020) and RoBERTa, are existing approaches. We also apply RoBERTa to the ConceptNet+ extracted by Paul et al. to compare the performance against our COMET-based knowledge extraction methods. The results show that the common-

---

[1] The wiki has since migrated to `https://idebate.org/debatabase`.

[2] `https://huggingface.co/roberta-base`

[3] `https://git.ecdf.ed.ac.uk/s1707343/commonsense-argmining/-/tree/master`

		ARGCON	ARGCON+LINK
Debate	train	0.56	0.85
	val	0.53	0.83
	test	0.54	0.83
Essay	train	0.42	0.71
	val	0.37	0.68
	test	0.42	0.70
M-Arg	train	0.20	0.69
	val	0.23	0.74
	test	0.22	0.72

Table 4: Coverage of ARGCON vs. ARGCON+LINK.

sense enhanced KE-RoBERTa models improve upon RoBERTa in almost all cases. In addition, ARGCON and ARGCON+LINK give the best results on the Debate and Essay datasets respectively. For significance testing we used the Almost Stochastic Order test (Dror et al., 2019; Del Barrio et al., 2018) implemented in Python using the `deep-significance` library (Ulmer et al., 2022). We compared all models against the baseline based on the sample predictions on a single trial, with a confidence level of $\alpha = 0.05$ (before adjusting for all pair-wise comparisons using the Bonferroni correction). We consider the results significant when $\epsilon_{\text{min}} < \tau$ with $\tau = 0.3$.

We conducted further experiments on the M-Arg dataset to evaluate our knowledge extraction method on a three-label (attack, support, and neutral) task. The results in Table 6 confirm that our methods do indeed perform better than RoBERTa with no external knowledge.

6 Analysis and Discussion

In order to interpret the results better, we analyze the inference chains generated by ARGCON, and we also investigate the performance of the models across different topics. In Table 7, we provide some examples from the data to discuss our findings. For clarity, we have kept the table concise and omitted information not relevant to the points being made.

6.1 Performance Across Different Topics

The performance of the model varies across different topics. Figure 3 depicts the macro F1 scores of the KE-RoBERTa model for each topic in the M-Arg dataset. We observe significant improvements across some of the topics such as ‘Taxes’, ‘Racism’, ‘Families’ and ‘Climate Change’. COMET has been exposed to these topics in the training set and so is better able to provide relevant knowledge.

One topic in which RoBERTa outperforms both knowledge-enhanced models is ‘Why They Should Be Elected’. Our analysis of the class level precision (Appendix A) indicates that the drop in precision in Table 6 is actually due to this missidentification of support relations when we add external knowledge, we discuss this in Subsection 6.4.

6.2 Noisy Connections

The injected knowledge could be detrimental in some cases. The COMET generated knowledge can introduce noise by generating irrelevant relationships between two arguments. Consider Ex1 in Table 7, where ARGCON makes an incorrect prediction while the RoBERTa model correctly predicts a support relation. Here, the generic word ‘good’ is used to link Arg_1 and Arg_2. While useful, the analysis shows that the attetion mechanism in KE-RoBERTa does not solve the issue.

ARGCON *can sometimes infer relations when there are in fact none present.* The existence of a commonsense inference path between the arguments causes the model to favor the support relation over the neutral one. In Ex3 of Table 7, the first argument attacks the speaker’s opponent while the second deflects the argument by addressing something unrelated. However, COMET connects the first sentence to the second on the basis that ‘the country is in decline’ supports ‘being weaker’.

6.3 ARGCON vs. ConceptNet$^+$

The nature of the injected knowledge can change the outcome. Ex4 in Table 7 shows a comparison of knowledge extracted with ARGCON+LINK and ConceptNet$^+$. In the example the model provided with input from ARGCON+LINK predicts the correct relation while ConceptNet$^+$ fails. By examining the knowledge we observe that ARGCON+LINK is able to deduce that the first argument hinders the ability of parents to teach their children and therefore prevents students from learning skills. This allows the model to make the correct prediction. However, ConceptNet$^+$ can only deduce that a parent has a child that can learn.

As seen in this example, ConceptNet$^+$ provides some information about causality (learning causes intelligence) but this information is limited and often too simplistic to provide meaningful information for understanding the underpinning arguments.

		Student Essay			Debatepedia		
Model	**KG**	**P**	**R**	**F1**	**P**	**R**	**F1**
(Paul et al., 2020)	ConceptNet$^+$	0.56	0.63	0.60	0.64	0.64	0.64
RoBERTa	-	0.64	0.68	0.66	0.75	0.75	0.75
KE-RoBERTa	ConceptNet$^+$	0.66	0.70	0.68	0.74	0.74	0.74
	ArgCon	0.66	0.69	0.67	**0.76**	**0.76**	**0.76**
	ArgCon+Link	**0.70**	**0.71**	**0.70***	0.75	0.75	0.75

Table 5: Macro-averaged F1, precision, recall scores of our experiments on the test sets for the 2-class task. Asterisk indiciates significant stochastic dominance over the baseline.

		M-Arg		
Model	**KG**	**P**	**R**	**F1**
RoBERTa	-	**0.57**	0.41	0.44
KE-RoBERTa	ArgCon	0.48	0.51	0.48*
	ArgCon+Link	0.54	**0.48**	**0.49***

Table 6: Macro-averaged F1, precision, recall scores of our experiments on the test sets for the 3-class task. Asterisk indicates significant stochastic dominance over the baseline.

On the other hand, COMET is well equipped to handle more complex causal relations due to the nature of the data it was trained on. This supports our claim that the nature of COMET is better suited to providing useful commonsense for argument mining tasks compared to ConceptNet$^+$.

6.4 Uncovering Hidden Biases in Arguments

In argument understanding, the ability to capture and make some of the implicit biases explicit can be helpful. Ex2 in Table 7 shows an argument made by the speaker to attack [PERSON X]'s candidacy. The argument made relies on the implicit reasoning that [PERSON X] did not fight [COUNTRY Y] because he is a communist. While there are many controversial assumptions being made when generating this knowledge, the COMET knowledge reflects the beliefs underlying the argument pair.

One of the key differences between fact-based and commonsense knowledge graphs is that the latter are more likely to capture cultural and normative biases. This is amplified by the fact that COMET is built on a pre-trained BART model, which captures biases present in the training corpus.

7 Conclusion

In this paper, we introduced a new method, ArgCon, to investigate the role of the commonsense knowledge encoded in COMET for the task of relational argument mining. In particular, ArgCon can reveal implicit commonsense reasoning chains (i.e., warrants) and these chains provide useful information for classifying argument relations such as attack, support, and neutral. Our experiments on three different datasets show that the chains generated by this method outperform existing approaches. Our analysis shows that the performance of our models varies given the topic under discussion and in some cases ArgCon creates noisy connections as a result of injected knowledge. We also observe that our model performs well in its ability to handle complex causal relationships and reason about social norms and biases.

In this work, we only consider pairs of AUs without taking into account any additional context. However, incorporating the text surrounding the pair of arguments is often necessary to be able to determine their relationship. In future work, we would like to incorporate such contextual information into our approach and explore the impact this has on the final performance. Another direction that we want to explore is considering other relations of COMET knowledge such as *intents*, *needs*, and *wants* to investigate whether this kind of social commonsense improves classification accuracy. Another avenue of investigation we would like to pursue is the role that external knowledge plays in increasing explainability and improving the diagnosis of end-to-end model predictions. Beyond argument mining, we believe that ArgCon is useful for other inference tasks such as recognizing textual entailment and question answering, we hope to experiment with other datasets to test this.

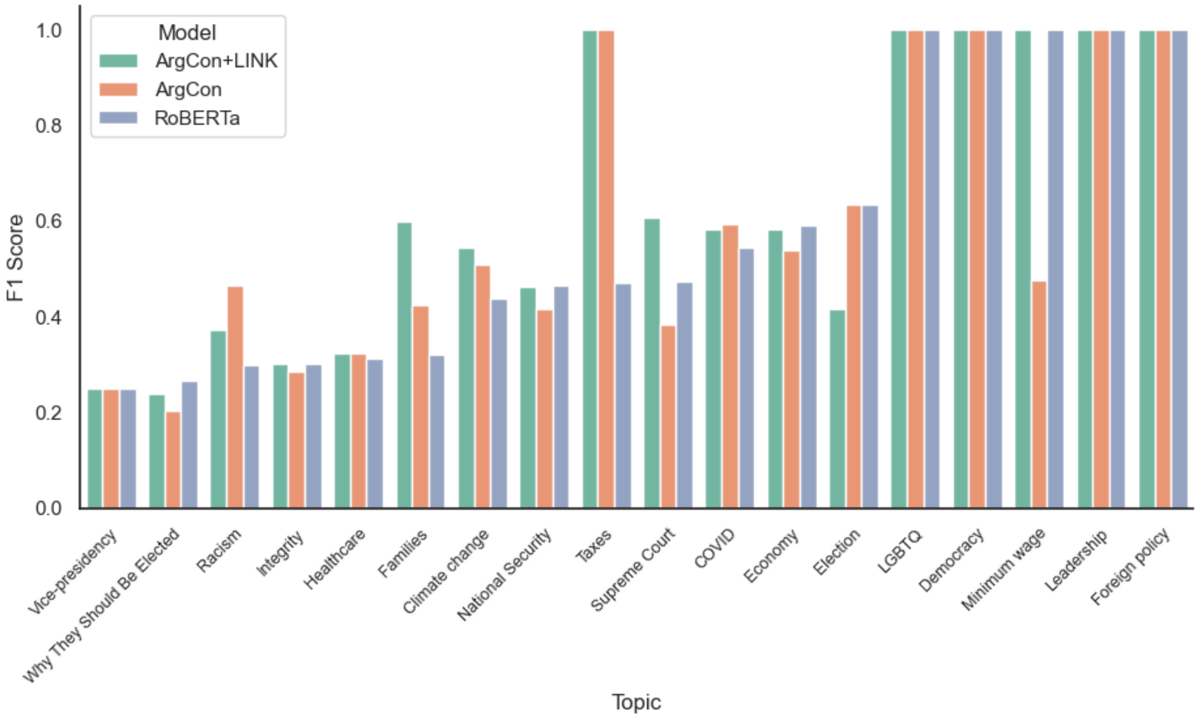


Figure 3: Comparison of F1 scores of our models on various topics in the test set of the M-Arg dataset.

Ex1	**Arg$_1$**	Congress will not ultimately cut Medicare to pay for reform.
	Arg$_2$	The 2010 US health care reform legislation is a good idea.
	ArgCon+Link	[Arg$_1$], isAfter, PersonY is not a member of the government, isAfter, PersonY is a lawyer, Causes, good, CausedBy, [Arg$_2$]
	Label: Support	**RoBERTa Prediction**: Support, **ArgCon+Link Prediction**: Attack
Ex2	**Arg$_1$**	[PERSON X] never fought it.
	Arg$_2$	[PERSON X] has been a cheerleader for communist [COUNTRY Y] over the last several decades.
	ArgCon	[Arg$_2$], HinderedBy, [PERSON X] is a good person., HinderedBy, [PERSON X] is a communist., Hinders, [Arg$_2$]
	Label: Support	**ArgCon Prediction**: Support
Ex3	**Arg$_1$**	Under this president, we become weaker, sicker, poor, more divided and more violent.
	Arg$_2$	With regard to being weaker, the fact is that I've gone head to head with [PERSON X] and made it clear to him we're not going to take any of his stuff.
	ArgCon	[Arg$_2$], HinderedBy, under this president, the country is in decline, HinderedBy, with regard to being weaker, Hinders, [Arg$_2$]
	Label: Neutral	**ArgCon Prediction**: Support
Ex4	**Arg$_1$**	Parents are usually too busy with their daily jobs to teach their children the life skills
	Arg$_2$	Students can learn practical skills after school hours from their parents
	ArgCon+Link	[Arg$_2$], Causes, parents are lazy, HinderedBy, parents are teaching their children, CausedBy, [Arg$_2$]
	ConceptNet$^+$	parent Antonym child used for learning, learning Causes intelligence related to mind HasA parent, parent Antonym child CapableOf learn
	Label: Attack	**ArgCon+Link Prediction**: Attack, **ConceptNet$^+$ Prediction**: Support

Table 7: Examples of knowledge and model predictions from the data: Ex1 is taken from the Debate dataset, Ex2 and Ex3 are taken from M-Arg, and Ex4 is taken from Student Essay. One interesting finding is that due to inductive biases in ArgCon *HinderedBy* and *Hinders* do not always reflect negative relations between nodes. The tags [PERSON X] and [COUNTRY Y] have been used to remove mentions of named individuals and countries in this table to avoid causing political offense.

Acknowledgements

The work was supported by the Edinburgh-Huawei Joint Lab and Huawei's grant CIENG4721/LSC. We would like to thank Paul et al. for providing their datasets and code-base allowing us to compare our work and improve upon their results.

Limitations

One limitation comes from our reliance on manually-constructed knowledge graphs. Given that different cultures and communities have varying notions of what is and is not commonsense, what is left out between arguments will also depend on the culture. For example, a knowledge graph constructed by English speakers in the UK, may not be able to provide useful knowledge about arguments about politics in the US due to differences in cultural values.

Similarly, our approach does not apply to low-resource languages that do not have sufficiently large commonsense knowledge graphs to train a transformer model such as COMET. Machine Translation of existing knowledge graphs may also prove insufficient due to cultural differences.

Ethics Statement

The datasets we use in this paper are publicly available. These datasets are anonymized and do not include sensitive information. A source of ethical concern is our reliance on generative models to provide warrants for arguments. These models, being trained on large web-based corpora may potentially generate problematic or biased outputs.

References

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Ruben Branco, António Branco, João António Rodrigues, and João Ricardo Silva. 2021. Shortcutted Commonsense: Data Spuriousness in Deep Learning of Commonsense Reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1521, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lucas Carstens and Francesca Toni. 2015. Towards relation based Argumentation Mining. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 29–34, Denver, CO. Association for Computational Linguistics.

Tuhin Chakrabarty, Aadit Trivedi, and Smaranda Muresan. 2021. Implicit Premise Generation with Discourse-aware Commonsense Knowledge Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6247–6252, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Oana Cocarascu, Antonio Rago, and Francesca Toni. 2019. Extracting dialogical explanations for review aggregations with argumentative dialogical agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, page 1261–1269, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Oana Cocarascu and Francesca Toni. 2017. Identifying attack and support argumentative relations using deep learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1374–1379, Copenhagen, Denmark. Association for Computational Linguistics.

Eustasio Del Barrio, Juan A Cuesta-Albertos, and Carlos Matrán. 2018. An optimal transportation approach for assessing almost stochastic order. In *The Mathematics of the Uncertain*, pages 33–44. Springer.

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep dominance - how to properly compare deep neural models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2773–2785. Association for Computational Linguistics.

Michael Fromm, Evgeniy Faerman, and Thomas Seidl. 2019. TACAM: Topic And Context Aware Argument Mining. *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 99–106. ArXiv: 1906.00923.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A Knowledge-Enhanced Pretraining Model for Commonsense Story Generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Chadi Helwe, Chloé Clavel, and Fabian M. Suchanek. 2021. Reasoning with transformer-based models:

Deep learning, but shallow reasoning. In *3rd Conference on Automated Knowledge Base Construction*.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (Comet-) Atomic 2020: On Symbolic and Neural Commonsense Knowledge Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(7):6384–6392.

Yohan Jo, Seojin Bang, Chris Reed, and Eduard Hovy. 2021. Classifying Argumentative Relations Using Logical Mechanisms and Argumentation Schemes. *Transactions of the Association for Computational Linguistics*, 9:721–739.

Weichen Li, Patrick Abels, Zahra Ahmadi, Sophie Burkhardt, Benjamin Schiller, Iryna Gurevych, and Stefan Kramer. 2021. Topic-guided knowledge graph construction for argument mining. In *2021 IEEE International Conference on Big Knowledge (ICBK)*, pages 315–322.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-Based Reasoning over Heterogeneous External Knowledge for Commonsense Question Answering. *arXiv:1909.05311 [cs]*. ArXiv: 1909.05311.

Rafael Mestre, Razvan Milicin, Stuart E. Middleton, Matt Ryan, Jiatong Zhu, and Timothy J. Norman. 2021. M-arg: Multimodal argument mining dataset for political debates with audio and transcripts. In *Proceedings of the 8th Workshop on Argument Mining*, pages 78–88, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Juri Opitz and Anette Frank. 2019. Dissecting content and context in argumentative relation analysis. In *Proceedings of the 6th Workshop on Argument Mining*, pages 25–34, Florence, Italy. Association for Computational Linguistics.

J.Z. Pan, G. Vetere, J.M. Gomez-Perez, and H. Wu. 2016. *Exploiting Linked Data and Knowledge Graphs for Large Organisations*. Springer.

Debjit Paul, Juri Opitz, Maria Becker, Jonathan Kobbe, Graeme Hirst, and Anette Frank. 2020. Argumentative Relation Classification with Background Knowledge. *Computational Models of Argument*, pages 319–330. Publisher: IOS Press.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Ramon Ruiz-Dolz, Jose Alemany, Stella Heras, and Ana Garcia-Fornes. 2021. Transformer-Based Models for Automatic Identification of Argument Relations: A Cross-Domain Evaluation. *IEEE Intelligent Systems*, pages 1–1. Conference Name: IEEE Intelligent Systems.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Noam Slonim, Yonatan Bilu, Carlos Alzate, Roy Bar-Haim, Ben Bogin, Francesca Bonin, Leshem Choshen, Edo Cohen-Karlik, Lena Dankin, Lilach Edelstein, Liat Ein-Dor, Roni Friedman-Melamed, Assaf Gavron, Ariel Gera, Martin Gleize, Shai Gretz, Dan Gutfreund, Alon Halfon, Daniel Hershcovich, Ron Hoory, Yufang Hou, Shay Hummel, Michal Jacovi, Charles Jochim, Yoav Kantor, Yoav Katz, David Konopnicki, Zvi Kons, Lili Kotlerman, Dalia Krieger, Dan Lahav, Tamar Lavee, Ran Levy, Naftali Liberman, Yosi Mass, Amir Menczel, Shachar Mirkin, Guy Moshkowich, Shila Ofek-Koifman, Matan Orbach, Ella Rabinovich, Ruty Rinott, Slava Shechtman, Dafna Sheinwald, Eyal Shnarch, Ilya Shnayderman, Aya Soffer, Artem Spector, Benjamin Sznajder, Assaf Toledo, Orith Toledo-Ronen, Elad Venezian, and Ranit Aharonov. 2021. An autonomous debating system. *Nature*, 591(7850):379–384.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of The Thirty-First AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Stephen E. Toulmin. 1958. *The Uses of Argument*. Cambridge University Press.

Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. 2022. deep-significance-easy and meaningful statistical significance testing in the age of neural networks. *arXiv preprint arXiv:2204.06815*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, and Michael Witbrock. 2019. Improving Natural Language Inference Using External Knowledge in the Science Questions Domain. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7208–7215. Number: 01.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022.

Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

A Class level performance of our model

Table 8 shows the breakdown of our models' performance across the different classes. To begin with, the performance on Debatepedia is mixed, with a slight improvement in the F1 score of the support class due to ARGCON. However, there is no clear winner as all models perform similarly to the baseline. This suggests the need for further investigation to identify the cause of the performance bottleneck which is not mirrored by the other 2 datasets.

In the Student Essay, we observe that knowledge indeed helps to identify the minority attack class, both in terms of precision and recall with ARGCON+LINK providing the greatest contribution to the precision of the attack class and a slight improvement in the recall of support. This gives strong evidence to support our hypothesis that the knowledge provided by our methods is effective for distinguishing between relation types.

We see a similar phenomenon emerge in the M-Arg dataset, where there is a consistent improvement in the identification of the smallest attack class, as we go from no knowledge to ARGCON to ARGCON+LINK. We also notice a significant, 33% increase in the recall of supporting AU pairs. However, as we introduce knowledge to our model, we see that the recall of neutral pairs decreases alongside the precision of support. This is what was observed in Subsection 6.4 due to the appearance of supporting inference chains when we generate knowledge with ARGCON.

To summarise, the improvement of our model on the Student Essay and M-Arg dataset provides evidence to support continued investigation and improvement of commonsense-aided relational argument mining with COMET. In particular, we see that external commonsense knowledge significantly improves the identification of minority classes in imbalanced datasets which are particularly common to the domain of argumentation.

	Knowledge	**Attack**			**Support**			**Neutral**		
		P	R	F1	P	R	F1	P	R	F1
Debate	-	0.71	0.80	0.75	0.79	0.71	0.75	-	-	-
	ConceptNet$^+$	0.72	0.75	0.74	0.76	0.73	0.75	-	-	-
	ARGCON	0.73	0.77	0.75	0.78	0.75	**0.76**	-	-	-
	ARGCON+LINK	0.72	0.79	0.75	0.78	0.71	0.75	-	-	-
Essay	-	0.33	0.45	0.38	0.95	0.92	0.93	-	-	-
	ConceptNet$^+$	0.37	0.48	0.42	0.95	0.93	0.94	-	-	-
	ARGCON	0.36	0.44	0.40	0.95	0.93	0.94	-	-	-
	ARGCON+LINK	0.44	0.47	**0.46**	0.95	0.95	**0.95**	-	-	-
M-Arg	-	0.00	0.00	0.00	0.43	0.31	0.36	0.90	0.94	**0.92**
	ARGCON	0.14	0.07	0.09	0.37	0.60	**0.46**	0.92	0.88	0.90
	ARGCON+LINK	0.25	0.13	**0.17**	<u>0.32</u>	0.64	0.43	0.93	<u>0.83</u>	0.88

Table 8: Micro-F1 score comparison of the models in our experiments. Results with no knowledge represent our RoBERTa baseline.

How people talk about each other:
Modeling Generalized Intergroup Bias and Emotion

Venkata S. Govindarajan[1] **Katherine Atwell**[2] **Barea Sinno**[3]
Malihe Alikhani[2] **David I. Beaver**[1] **Junyi Jessy Li**[1]
[1] Department of Linguistics, The University of Texas at Austin
[2] Department of Computer Sciences, University of Pittsburgh
[3] Department of Political Science, Rutgers University
venkatasg@utexas.edu, kaa139@pitt.edu, barea.sinno@gmail.com,
malihe@pitt.edu, dib@utexas.edu, jessy@utexas.edu

Abstract

Current studies of bias in NLP rely mainly on identifying (unwanted or negative) bias towards a specific demographic group. While this has led to progress recognizing and mitigating negative bias, and having a clear notion of the targeted group is necessary, it is not always practical. In this work we extrapolate to a broader notion of bias, rooted in social science and psychology literature. We move towards predicting interpersonal group relationship (IGR) — modeling the *relationship between the speaker and the target* in an utterance—using fine-grained interpersonal emotions as an anchor. We build and release a dataset of English tweets by US Congress members annotated for interpersonal emotion – the first of its kind, and 'found supervision' for IGR labels; our analyses show that subtle emotional signals are indicative of different biases. While humans can perform better than chance at identifying IGR given an utterance, we show that neural models perform much better; furthermore, a shared encoding between IGR and interpersonal perceived emotion enabled performance gains in both tasks.

1 Introduction

Currently, most work studying bias in NLP situates bias as negative or pejorative language use towards an individual or group based on traits like race, gender, etc (Kaneko and Bollegala, 2019; Sheng et al., 2019; Sap et al., 2020; Webson et al., 2020; Pryzant et al., 2020; Sheng et al., 2020). While these approaches greatly advance our understanding of bias in language and its impact and mitigation in NLP, focusing on specific demographic dimensions or an individual's intent is limiting and not always practical. Research in psychology and social science suggests a different perspective. Bias can be seen as a relationship between people and groups, situated in context (Van Dijk, 2009); as such, bias refers to differences in behavior (in this case language use) as a result of differences in the relationship between speaker and target. The language we produce is biased in one way or another, whether we intend to or not, and whether that bias is positive, negative, or not clearly associated with any valuation (Beaver and Stanley, 2018).

In psychological work on Linguistic Intergroup Bias (Maass, 1999), bias originates from the relationship between the speaker and target of an utterance, i.e. their **interpersonal dynamics**, and manifests later in subtle ways. Consider the utterances (tweets) in (1), drawn from our collected data in which the identity of the speaker and target are masked:

(1) a. **In-group**: We stand w @Doe, who has seen a lot worse than cheap insults from an insecure bully. #MLKDAY weekend.
   b. **Out-group**: Parents and families live in constant fear for their children with food allergies. A worthy bipartisan cause - thank you @Doe for your leadership on this issue.

Both express support and admiration towards the target referent *Doe* – however, the second example uses words indicative that the speaker and target do not share a relevant social identity (in this case, their political party), expressed by words like *bipartisan*. The intensity of admiration expressed is also greater in (1-a) than (1-b). Thus, these two seemingly similar statements differ along interpersonal dimensions that are instructive as to how the bias of the speaker seeps into the utterance.

We now introduce two new tasks that directly model language use in terms of two interpersonal dimensions: (i) **interpersonal group relationship (IGR) prediction**, where we seek to understand how people talk about others who they consider to be in their same social group (in-group), versus those they consider outside their social group (out-group), and (ii) *perceived* **interpersonal emotion**

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2496–2506
May 2-6, 2023 ©2023 Association for Computational Linguistics

detection, where we situate these differences in terms of the emotion expressed in text *towards or in connection with* a target individual described in the utterance. Note that *interpersonal* emotion is different from a more standard, utterance level emotion detection task, as illustrated in row 2 of Table 1 which has seemingly opposing emotions.

We present a first-of-its-kind, *annotated* dataset for fine-grained interpersonal emotion detection, consisting of 3,033 tweets from members of the US Congress; all of these tweets mention another Congress member, hence providing us with 'found supervision' for IGR prediction (whether the speaker and the target belong to the same political party). Our analyses show that while positive interpersonal emotions appear in both in- and out-group situations, negative emotions like anger and disgust are overwhelmingly present in the latter. Meanwhile, human judgments for in vs. out-group membership on this dataset are overly reliant on the polarity of emotion; specifically, human judges are much *less* likely to attribute positive emotions towards out-group targets.

Baseline performances for perceived interpersonal emotion detection shows that this is a challenging task, as is consistent with existing work in emotion detection in general (Demszky et al., 2020). In particular, emotions in this dataset are often expressed with considerable subtlety, likely a characteristic of official political speech. To investigate whether IGR and emotions are intertwined and useful towards each other, we further developed a multi-task model for the prediction of both. We found compelling evidence that multi-tasking IGR and interpersonal emotion improves performance on both tasks with over 10% improvement in detection of disgust in out-group contexts, and 3% improvement in IGR prediction.

To summarize the contributions of this paper, we tackle *generalized intergroup bias*, a notion of bias rooted in social psychology that applies to all the various differences in the ways that people talk about others in their in-group or out-group. Standard bias tasks in NLP, and the broader goal of debiasing models could thus be set in a more general context. We present the first dataset to study both interpersonal group membership and emotion, which allows us to analyze both human and model behavior in terms of how the two interact with each other. We release our code and data online at `github.com/venkatasg/interpersonal-bias`.

2 Interpersonal Contexts & Emotions

Our aim is to build a generalized, data-driven approach towards studying bias situated in **interpersonal utterances**, which we define as any utterance where there is a target individual being talked about or referred to. Our goal is to model two novel tasks described below; examples are shown in Table 1.

Interpersonal Group relationship IGR is defined by the relationship between the speaker and target of an utterance. People belong to multiple social groups as part of their identity, however usually only some identities are salient in an utterance in context. We define *in-group* utterances as ones where the speaker and target are in the same social group, and *out-group* utterances as one where they are in different social groups. Given an utterance u written by an individual s with target t, the IGR prediction task classifies whether s and t belong to the same social group within the context of u.

Interpersonal Emotion We define *perceived* interpersonal emotion as the emotion expressed by a speaker s *towards, or in connection with* the target t of the utterance u, as perceived by a reader. We use the Plutchik wheel of emotions, which is widely adopted in the community, as the basis of our emotion taxionomy (Plutchik, 2001); we use the 8 fundamental emotions (*admiration, anger, disgust, fear, interest, joy, sadness, surprise*) instead of the full 24 emotions in the wheel due to data sparsity. Interpersonal emotions may be different, or a subset of, emotion for the whole of an utterance, as illustrated in rows 2, 3 and 4 of Table 1. Given an utterance u written by an individual s with target t, the interpersonal emotion detection task identifies the perceived emotion of s towards the target t.

3 Data Collection

In our area of focus, we require natural language data which satisfies the following criteria: *(1)* Each utterance must have at least one target about whom the utterance mainly concerns. *(2)* The relationship between the speaker and the target must be inferred based on metadata or other information. Specifically, we are interested in aspects of their social identity that they share or differ on.

The dataset we collect comes from tweets by members of US Congress where other members are mentioned in the same tweet. We use this as

Tweet	Interpersonal Emotion	In/Out group?
As @Doe says, the times have found each and every one of us to Defend our Democracy For The People. Worth reading every line.	Admiration	In-group
Freedom has no greater nor tougher champion than @Doe. My prayers are with him and his family.	Admiration & Sadness	In-group
You don't get to decide what's "fine," @Doe. The constitution does. #DefendOurDemocracy #WednesdayThoughts	Anger & Disgust	Out-group
Thank you again Senator @Doe for leading the SRF WIN Act[...] I'm proud to be a co-sponsor	Admiration & Joy	Out-group

Table 1: Example utterances from our dataset with in/out group and interpersonal emotion labels

a convenient testbed: each member's group affiliation (i.e., their party identity) is public, thus we can easily know whether the speaker is tweeting to a target in their own party or not.[1] In other words, this dataset gives us "found supervision" for our first task of IGR prediction. For our second task, we annotate a subset of these tweets for perceived interpersonal emotion; this is, to our knowledge, the first dataset dedicated to interpersonal emotion.

3.1 Data Sources and Preprocessing

Social media text like tweets offer a fertile ground for our study. A focus on tweets with *mentions* in them satisfies our first criterion – people generally use mentions to say something about or towards another individual on twitter. Tweets by members of US Congress are a matter of public record, and we can infer the social relationship (in terms of party affiliation) between speaker and target using publicly available information. We prioritize working with a dataset of tweets by members of US Congress (downloaded using the Twitter API) between 2010 and 2021, spanning two presidencies, during which both parties held power in Congress. We filter these tweets to exclude retweets, and include those tweets that mention *at most* one other member of Congress whose party affiliation is known. We believe these 2 assumptions are sufficient to arrive at a dataset of tweets where the speaker is talking towards/about *one* target. Thus, we restrict ourselves to two social groups in this sphere — Democrat and Republican parties in the US. We sample an equal number of in-group and out-group tweets from a large sample consisting of all tweets by members of Congress. Apart from years 2010–2012 and 2021 which contained fewer tweets due to sparsity issues, we sampled at least 300 tweets each year.

3.2 Interpersonal Emotion Annotation

While we can infer whether a tweet is in-group or out-group based on the identity of speaker and target whose political affiliations are known, we still require annotated data on perceived interpersonal emotions. Interpersonal emotions vary in subtle ways from sentiment or overall sentiment of utterances: an utterance can have negative sentiment overall, but still convey positive emotions towards the target of the sentence (expressing admiration at someone's death for instance). For this reason, we devise an annotation schema for annotating *the emotion expressed by speaker s towards target t.*

Instructions Annotators are presented with a tweet, with the identity of the speaker unknown and that of the target masked with a placeholder name **@Doe** to minimize potential biases of the annotators' prior knowledge of party affiliation intruding into the annotation:

(2) If **@Doe** can get her hair done in person, Congress can vote in person...

Annotators are instructed to read the tweet and select only the most notable emotion(s) they think are expressed by the tweet author *in connection with* **@Doe**. To aid annotators, we provide examples of the 8 Plutchik emotions (*joy, admiration, fear, suprise, sadness, digust, anger and interest*) expressed as interpersonal emotions in tweets. Annotators are also shown a schematic of the Plutchik wheel of emotions, which acquaints them with how the emotions are related to one another in our framework. Annotators are allowed to select more than one emotion to account for emotion co-occurrence. We also explicitly tell annotators that more than one of the emotions can be present in the tweets, to encourage them to select all interpersonal emotions expressed. They are also allowed to not choose any emotion.

---

[1] For simplicity, we do not consider other factors such as the home state of a congress member.

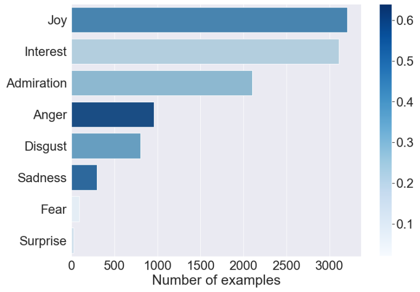


Figure 1: Emotions ordered by the number of examples where at least one rater uses a particular label. The color indicates the average interrater correlation.

Annotation To obtain reliable annotations, we prequalify annotators using a qualifying task. Annotators were recruited on Mechanical Turk using a qualifying task where they were asked to annotate 6 tweets using the schema shown above. We restricted the qualification task to annotators living in the USA who had attempted at least 500 HITS and had a HIT approval rate $\geq 98\%$. After manual inspection, 6 annotators were qualified for bulk annotation. Each tweet was annotated by three different annotators. To ensure annotators were paid a fair wage of at least 10$ an hour, we paid annotators $0.50 per HIT. Each HIT involved annotating 3 tweets, which we estimate to take on average 3 minutes to complete. In total, 3,033 tweets between 2010 and 2021 were annotated with perceived interpersonal emotion.

Agreement To measure agreement between annotators on the Plutchik-8 emotion wheel, we use the Plutchik Emotion Agreement (PEA) score from Desai et al. (2020). The PEA score addresses the issue of penalizing all disagreements equally, by penalizing dissimilar emotion annotations higher than more similar ones (according to the Plutchik wheel). Our PEA score is 0.73. The original PEA formulation used the best(max) pair of emotion annotations between two workers. Taking the *worst* combination of emotions between two workers (averaged over all tweets and workers), the PEA (min) score is 0.60. Overall, we find moderate to high agreement on fine-grained interpersonal emotions. In Figure 1 we also present interrater correlation, a metric used in Demszky et al. (2020); we see that distributions are similar.

Emotion	**Train**	**Dev**	**Test**
Admiration	467	64	58
Anger	225	40	46
Disgust	206	32	43
Fear	1	0	0
Interest	701	83	84
Joy	801	107	106
Sadness	72	11	11
Surprise	1	0	0
No Emotion	519	56	63

Table 2: Distribution of emotions in train-dev-test split

Emotion	**All**	**In-Group**	**Out-Group**
Admiration	15.5	22.2	9.1
Anger	8.2	1.0	15.1
Disgust	7.4	0.3	14.2
Interest	22.9	27.2	18.6
Joy	26.7	32.2	21.4
Sadness	2.5	2.6	2.4
No Emotion	16.8	14.5	19.1

Table 3: Proportion of emotions in different interpersonal contexts

Aggregation We consider a tweet to have a certain emotion label if at least 2 out of 3 annotators agree that the particular emotion was present in the tweet. A total of 638 tweets have no interpersonal emotion associated with them. We employ a 80-10-10 train-dev-test split on our data.

The number of annotated examples (tweets) per emotion is shown in Table 2. We omit *fear* and *surprise* from future tables due to the absence of annotated examples.

4 Preliminary Analysis

How are emotions distributed? When observing the distribution of aggregated emotion labels themselves, a clear pattern emerges as seen in Table 3. Negative emotions such as anger and disgust are almost always expressed in out-group settings, while positive emotions are present in both in-group and out-group settings. A similar distribution of emotions was observed for Democrats and Republicans — members of both parties reserved their public anger and disgust for members of the other party. This reflects an innate bias in terms of the distribution of interpersonal emotions per situation, and warrants future work to explore negative interpersonal emotions in an in-group setting.

Figure 2 shows the co-occurrence of interpersonal emotions in our dataset. We can see that emotions that are farther apart and more dissimilar, such as admiration and disgust, joy and sadness, co-occur infrequently. Emotions that are closer such as anger and disgust, admiration and joy, co-occur much more often. The only outlier is the higher than normal co-occurrence of admiration with sadness — after a closer examination, this can

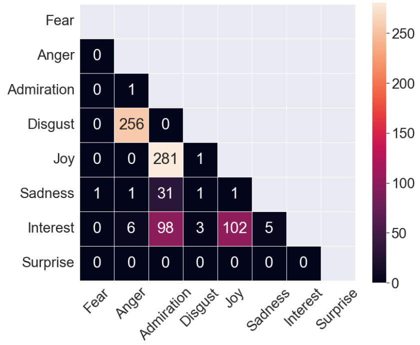


Figure 2: Co-occurence of emotions in our dataset.

be attributed to tweets expressing admiration and sadness at the passing, or end of the career, of a fellow congressperson.

Who were the targets of negative emotions? On further analysis, it appears that most of the out-group disgust and anger is directed at 3 handles – @speakerryan, @speakerpelosi, and @speakerboehner who were all Speakers of the House of Representatives over most of the time period of our dataset. 63.7% of disgust and 64.3% of anger is directed towards these three twitter handles. 11.9% of all tweets in our dataset are directed at these handles, indicating the preponderance of negative interpersonal emotion directed at the Speaker of the house. However, we note that negative emotions like anger and disgust were still expressed towards 51 and 45 different individuals in our dataset, respectively.

Can humans predict in/out-group? While our data naturally comes with "gold" IGR labels, what is unexplored is whether the distinction between in-group and out-group speech is prominent and noticeable by humans. Additionally, it is also unclear if humans might have their own expectation of how in/out-group speech should be characterized.

Concretely, we investigate if human annotators were capable of accurately performing the IGR prediction task when the speaker and target are masked. Two authors of this paper, one a social science graduate student, and the other a computational linguistics graduate student, annotated 50 random tweets from our validation data which they had not been exposed to earlier for in/out group labels. Their Fleiss κ agreement score was 0.64, indicating moderate agreement.

To check how accurate their judgements were, we calculate for each annotator their F1 score against our "gold" in/out group labels. Their F1 scores on these 50 tweets were 0.67 and 0.63, which as we will discuss in Section 6, only match simple baselines of supervised systems. Annotators comments indicate that they overly relied on the sentiment of tweets to make the classification — positive sentiment means in-group and negative sentiment means out-group. While negative emotions are over-represented in out-group situations as Table 3 shows, our dataset contains a substantial presence of out-group tweets with positive interpersonal emotions as well. Annotators also noticed some lexical cues like 'bipartisan' that are indicative of out-group tweets.

Do pre-trained representations capture interpersonal emotions? Pre-trained language models have been found to learn sentence representations that cluster by domain without supervision (Aharoni and Goldberg, 2020). We wished to investigate if any of our annotated properties cluster inherently in reduced representations of the tweets in our data. To obtain unsupervised representations, we use BERTweet (Nguyen et al., 2020), a language model pre-trained on 850M English tweets. We take the 768 dimensional embeddings from the final layer of the <s> token in BERTweet, and dimensionally reduce them to 2 dimensions using UMAP (Sainburg et al., 2021). Figure 3 shows the distribution of tweets, color coded for interpersonal emotions. While there is a lot of overlap between representations when stratified by emotion, we can see that some emotions that are intuitively opposite, like admiration & disgust, joy & sadness are moderately separable. This indicates that interpersonal emotions do define some topic or domain level properties of a tweet.

5 Experiments

We detail our experiments for the two novel tasks discussed in Section 2: predicting the IGR (in-group or out-group) given a tweet, and predicting the interpersonal emotion given a tweet. We present baselines for the two tasks separately, and also present a multi-task model to gauge the extent to which knowledge of IGR may help in predicting interpersonal emotion, and vice versa.

5.1 Interpersonal Group Relationship

Sentiment-Rule Our first baseline is a rule-based one leveraging coarse sentiment: if a tweet's sentiment is predicted to be negative, classify it as out-group; if positive, classify it as in-group; and if

Figure 3: Distribution of interpersonal emotions in unsupervised representations of tweets in our dataset. Orange indicates the emotion was present for that tweet. Each point represents one tweet from our dataset.

neutral, classify it as either in-group or out-group randomly. We use a RoBERTa-Base model finetuned for sentiment on tweets (Barbieri et al., 2020) to extract the sentiment of each tweet in our dataset.

NB-SVM As a second baseline, we build an SVM model that uses Naive-Bayes log-counts ratios of unigrams and bigrams (Wang and Manning, 2012).

BERTweet We use BERTweet (Nguyen et al., 2020), a language model pre-trained on 850M English tweets as our dataset consists purely of English language tweets. A classification head is placed on top of the language model. We also experiment with a version where the language model parameters are frozen, and only the classification head parameters are finetuned (BERTweet-ft).

The input to all models is only the tweet with no other context, and the target masked with a placeholder `@USER`.

5.2 Interpersonal Emotion

EmoLex As a baseline model for interpersonal emotion identification, we rely on EmoLex (Mohammad and Turney, 2013). EmoLex consists of 14,182 crowdsourced words associated with the 8 basic Plutchik emotions. Critically, these words appear in emotional contexts, but are not necessarily emotion words themselves. EmoLex counts occurrences of words from its lexicon in an utterance, and assigns a normalized score for each emotion based on occurrence frequency. We consider an emotion to be on, if it's normalized score is $\geq$ 0.001. While EmoLex has issues with regards to its context insensitivity and the social biases built into its lexicon (Zad et al., 2021), we include it as a baseline to understand to what extent interpersonal emotions can be deduced using a lexicon.

BERTweet We use the same BERTweet model as earlier. We add a dense output layer on top of the pretrained model for the purposes of finetuning, with a sigmoid cross entropy loss function to support multi-label classification. The loss is weighted for each of the 8 emotion labels with the ratio of positive and negative examples to increase precision. If none of the 8 emotion labels are flipped on, we consider that to be the 'No Emotion' label, i.e. there is no interpersonal emotion between speaker and target in the tweet. We experiment with a version of the model where the language model parameters are frozen and only the labelling head parameters are finetuned (BERTweet-ft).

5.3 Multi-Task Model

In § 4, we observed that the emotions anger and disgust are overwhelmingly present in out-group situations. Thus, we hypothesize that IGR information would be useful towards interpersonal emotion identification, and vice versa. To test this hypothesis we train a multi-task model. The model is trained to predict *both* the IGR label and emotion using shared parameter finetuning.

We use the same BERTweet model as earlier. We add two dense output layers on top of the pretrained model, one for classifying IGR and another for labelling interpersonal emotion. Both heads share the same parameters below. These are trained with same loss as earlier individual models. The model alternates between finetuning for group relationship and emotion over every training item.

5.4 Implementation

We use `bertweet-base` pretrained embeddings from Huggingface's models hub (Wolf et al., 2020). All models are finetuned for a maximum of 20 epochs with early stopping. Early-stopping patience for models trained on each task separately is 3. The patience for the multi-task model is set at 5 as the multi-tasking setup led to slower convergence. The learning rate for the classification heads was set at 5e-3 while the learning rate for the internal language model parameters was set at 2e-5. Dropout probabilities in classification heads

Model	F1	Model	F1
Majority class	51.1	BERTweet	74.1 (3.3)
Sentiment-Rule	56.3	BERTweet-ft	66.5 (1.6)
NB-SVM	62.5	Multitask	77.3 (0.8)
Human	66.7		

Table 4: Results on test set, with SD in parentheses, for interpersonal group relationship prediction task.

In-group	Out-group
thanks, love, count me birthday, my colleague	thanks, bipartisan, restore kind, resignation

Table 5: Top unigram and bigram features from NB-SVM model for each class.

	Emo Lex	BERTweet	BERTweet -ft	Multi-task
Admir.	37.5	70.3 (3.7)	40.7 (1.1)	68.9 (1.6)
Anger	26.6	71.3 (11.2)	23.0 (3.4)	69.3 (3.3)
Disgust	25.5	47.1 (21.6)	13.0 (4.1)	74.5 (7.1)
Interest	0	53.1 (3.3)	5.8 (2.4)	51.5 (8.5)
Joy	48.4	85.9 (1.9)	71.3 (1.4)	83.6 (1.3)
Sadness	4.3	11.1 (9.6)	0	33.6 (18.5)
No Emotion	22.2	49.1 (1.2)	43.4 (3.8)	71.6(1.2)

Table 6: F1 scores on test set, with SD in parentheses, for interpersonal emotion labelling task.

Emotion	BERTweet	MultiTask
Admiration	77.9 (2.6)	72.8 (3.9)
Anger	71.7 (9.9)	69.4 (3.4)
Disgust	48.2 (22.4)	75.9 (6.5)*

Table 7: F1 scores on test set, SD in parentheses on out-group tweets. * indicates statistical significance ($p<0.05$)

was set at 0.1. The best performing model before early stopping on validation data was chosen in all cases. We report F1 scores averaged over 3 random restarts for all models, with the standard deviation in parentheses next to the mean.

6 Results and Analysis

Interpersonal Group relationship In modeling IGR, we find that `Sentiment-Rule` performs not much better than chance (Table 4). This underscores one strength of our data, which contains a sizable number of out-group tweets with positive interpersonal emotion attached to them. The `NB-SVM` model based on unigrams and bigrams performs slightly better, and picks up on some obvious out-group lexical cues like the lemma 'bipartisan', as shown in Table 5. The `BERTweet` model performs substantially better, performing over 10 points better than humans. The model, with only the classification head finetuned, leaving the language model parameters intact(`BERTweet-ft`) performs about 10 points above chance — indicating that there may be features advantageous towards this task in the vanilla LM itself.

Interpersonal Emotion We find that the `EmoLex` baseline, which relies purely on lexical cues, performs dismally on our data, with poor performance in both in-group and out-group settings(Table 6). This is a strong indication that emotions are expressed more implicitly in this dataset. The `BERTweet` model performs substantially better, indicating that interpersonal emotions, even if implicit, can be learned.

Multitask Model As Table 4 shows, Multitasking the two tasks leads to a noticeable improvement in F1 for IGR prediction, with the differences being statistically significant using a bootstrap test ($p<0.05$; Berg-Kirkpatrick et al., 2012); the multi-task model is also more stable with much lower variance across runs. These results suggest that interpersonal emotion is useful towards IGR prediction.

Table 6 shows that the performance of the multitask model on predicting interpersonal emotions is significantly better that the BERTweet model ($p<0.05$) on emotions like *disgust*, which suggests that IGR is useful towards the task of emotion identification. Furthermore, multitasking boosted performance at predicting the *no emotion* label by 20%. Table 7 compares the multitask model's performance against the BERTweet model in *out-group* settings (where most of the gains were found) for 3 emotions — illustrating the boost in performance afforded by joint modeling of IGR and emotion for *digust*. The 3 emotions listed also showed significant differences in their distribution in in-group and out-group settings.

Humans vs. Models Comparatively, we find that model performance exceeds human performance on the task of in-group versus out-group prediction, albeit not on the same dataset. The model's main driver of performance is its high accuracy on positive intergroup emotion out-group tweets, such as those expressing admiration or joy. Human annotators consistently fall back on the heuristic that sentences with positive affect probably imply that the speaker is talking about someone in their in-group. But it is not the case in the political domain, where overtures to bipartisanship serve as useful signals. For instance, both (3-a) and (3-b) express admiration towards the target Doe, where the first is in-group while the second is out-group. The call to civility is the only subtle linguistic cue that this tweet may constitute out-group speech.

(3) a. Admire @OfficialCBC Chairman @Doe's moral voice on issues of racism and restorative justice. He is a real leader for our nation and Congress.
b. A decade has passed, but our friendship is the same. Proud to work with @Doe to #ReviveCivility. #tbt Read more about our efforts here:

Future work needs to look into what information the embeddings are using to make their classification decision.

Model Errors While the multitasking setup improves model performance on the task of predicting IGR, and outperforms human labelers in our small pilot, it still gets some easy examples wrong, such as labelling (4) as in-group even though it expresses some disgust at the target. The model also falls into the same trap as human labelers — for instance assuming that a tweet expressing admiration must be in-group (5).

(4) Trump selected @USER for HHS Secretary. Price has undeniable history of cutting access to healthcare to millions, especially women.

(5) Inspiring speech from @USER - we have a duty to represent our country with respect & dignity. #NationalDayofCivility.

To ensure that model performance on IGR prediction is not limited by the size of our training data, we experimented with training `BERTweet` models on larger datasets. Since we have 'found supervision' for IGR labels, we only need to increase training data size by sampling more tweets from relevant accounts using the same procedure detailed in § 3.1. We found that F1 score does not increase with more training data.

Future work needs to look into linguistically motivated ways to improve model performance on the IGR task. Since we have observed that the multi-task setup improves model performance, perhaps other multi-task setups, such as modeling the overall affect towards the target expressed by the speaker might help in modeling IGR better.

7 Related Work

Emotion and Stance Detection A wealth of work has looked at corpora and models for the detection of perceived emotion in social media text (Mohammad, 2012; Wang et al., 2012; Mohammad and Kiritchenko, 2015; Abdul-Mageed and Ungar, 2017; Desai et al., 2020; Demszky et al., 2020). However existing work doesn't distinguish between emotion of a sentence as a whole, versus interpersonal emotion towards a target. The task closest to our study of interpersonal emotions is stance detection: whether the author has a favourable, neutral, or negative position towards a proposition or target. Mohammad et al. (2016) looked at stance in five target domains are given: abortion, atheism, climate change, feminism and Hillary Clinton. While stance detection focuses on a collection of utterances with the same topic, our interest is in modeling interpersonal emotion towards a target individual which is more fine-grained and can vary in each utterance.

Intergroup bias in Psychology The Linguistic Intergroup Bias (LIB) theory (Maass et al., 1989; Maass, 1999) states that there is a systematic asymmetry in language production qualities of a speaker as a function of the social category to which the referent of an utterance belongs. Through psycholinguistic experiments, LIB seeks to explain why stereotypes are transmitted and persist in daily life: in an interpersonal situation, socially desirable in-group behaviors and undesirable out-group behaviors are encoded at a higher level of **abstraction**, whereas socially undesirable in-group behaviors and desirable in-group behaviors are encoded at a lower level of abstraction. Work in psychology and psycholinguistics reproduced LIB in various domains such as political news reporting (Anolli et al., 2006) and crime reporting (Gorham, 2006); as well as work exploring how LIB can be used as an indicator for a speaker's prejudicial attitudes (Hippel et al., 1997), or as a predictor for racism (Schnake and Ruscher, 1998).

Contemporaneous studies on LIB, however, are hand-coded and have so far tended to focus on narrow concepts such as abstractness of the verb and coarse notions of sentiment. Nonetheless, the LIB hypothesis connects the two dimensions of interpersonal dynamics studied here with a third dimension directly related to semantic properties of the utterance.

8 Conclusion

Taking a cue from studies of bias in social science and psychology, we situate bias in language use through the lens of interpersonal relationships between the speaker and target of an utterance, and the speaker's interpersonal emotional state with respect to the target. Over a corpus of tweets by members of US Congress, we introduce two novel tasks – interpersonal group relationship prediction (IGR) and interpersonal emotion labelling, to better understand variation in language as a function of social relationship between speaker and target in interpersonal utterances. We find certain interpersonal emotions like anger and disgust are overrepresented in out-group situations, with the majority of the negative emotions directed at leaders of the two political parties. Through modeling studies, we find that transformer based models perform better than humans at predicting IGR given an utterance, raising the question as to what latent features of language the model uses to make this decision. Finally, we also find that joint modelling of the two dimensions is beneficial to prediction of certain interpersonal emotions in out-group situations. Future work needs to look into what information is useful for predicting IGR and emotions – with the Linguistic Intergroup Bias literature offering a clue as to which higher level semantic features vary systematically.

Ethics Statement

For our corpus of tweets on which we performed annotations, we downloaded the tweets using the official Twitter API. In accordance with the Twitter Terms of Service, we release tweet IDs and usernames, but not the tweet text itself. Our dataset was built through crowdsourced annotations on Amazon Mechanical Turk. To ensure annotators were paid a fair wage of at least $10 an hour, we paid annotators $0.50 per HIT. Each HIT involved annotating 3 tweets, which we estimate to take on average 3 minutes to complete.

Limitations

Our results show the importance of having reliable and accurate emotion prediction models, which is an open problem in psychology and computer science. Future work might look into identifying different fine-grained emotional constructs and study their correlations with the underlying linguistic biases. Future work may also look into the generalizability of the results presented here in other domains of language use.

While we present the utterances as constituting natural speech by one speaker (the congressperson who sent the tweet), it is likely most congresspeople employ social media teams that help in crafting the language of some of their tweets. However, we believe for the sake of interpersonal group membership, the relationship between the speakers and their targets would not be affected.

Finally, while we show that transformer based models perform better at IGR prediction than humans, we note that the human performance was on a small subset of test data. While it is possible that these models discovered latent features that could explain their better performance, the model could also be using spurious features idiosyncratic to our dataset, rather than true differences in in-group versus out-group speech.

Acknowledgements

This research is partially supported by NSF grants IIS-2107524, IIS-2145479 and Good Systems,[2] a UT Austin Grand Challenge to develop responsible AI technologies. We also acknowledge the Texas Advanced Computing Center (TACC)[3] at UT Austin for providing the computational resources for many of the results within this paper.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. EmoNet: Fine-Grained Emotion Detection with Gated Recurrent Neural Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 718–728, Vancouver, Canada. Association for Computational Linguistics.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised Domain Clusters in Pretrained Language Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Luigi Anolli, Valentino Zurloni, and Giuseppe Riva. 2006. Linguistic Intergroup Bias in Political Communication. *The Journal of General Psychology*, 133:237 – 255.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval:

---

[2] `http://goodsystems.utexas.edu`

[3] `https://www.tacc.utexas.edu`

Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

David Beaver and Jason Stanley. 2018. Toward a Non-Ideal Philosophy of Language. *Graduate Faculty Philosophy Journal*, 39(2):503–547.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An Empirical Investigation of Statistical Significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005, Jeju Island, Korea. Association for Computational Linguistics.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A Dataset of Fine-Grained Emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054, Online. Association for Computational Linguistics.

Shrey Desai, Cornelia Caragea, and Junyi Jessy Li. 2020. Detecting Perceived Emotions in Hurricane Disasters. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5290–5305, Online. Association for Computational Linguistics.

Bradley W. Gorham. 2006. News Media's Relationship With Stereotyping: The Linguistic Intergroup Bias in Response to Crime News. *Journal of Communication*, 56(2):289–308. Place: United Kingdom Publisher: Blackwell Publishing.

W. Hippel, Denise Sekaquaptewa, and P. Vargas. 1997. The Linguistic Intergroup Bias As an Implicit Indicator of Prejudice. *Journal of Experimental Social Psychology*, 33:490–509.

Masahiro Kaneko and Danushka Bollegala. 2019. Gender-preserving Debiasing for Pre-trained Word Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1641–1650, Florence, Italy. Association for Computational Linguistics.

Anne Maass. 1999. Linguistic Intergroup Bias: Stereotype Perpetuation Through Language. In Mark P. Zanna, editor, *Advances in Experimental Social Psychology*, volume 31, pages 79–121. Academic Press.

Anne Maass, Daniel Anthony Salvi, Luciano Arcuri, and Gün R. Semin. 1989. Language use in intergroup contexts: the linguistic intergroup bias. *Journal of Personality and Social Psychology*, 57 6:981–93.

Saif Mohammad. 2012. #Emotional Tweets. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 246–255, Montréal, Canada. Association for Computational Linguistics.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.

Saif M. Mohammad and Svetlana Kiritchenko. 2015. Using Hashtags to Capture Fine Emotion Categories from Tweets. *Computational Intelligence*, 31:301 – 326.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 29.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14. Association for Computational Linguistics.

Robert Plutchik. 2001. The Nature of Emotions. *American Scientist*, 89(4):344–350.

Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically Neutralizing Subjective Bias in Text. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):480–489.

Tim Sainburg, Leland McInnes, and Timothy Q Gentner. 2021. Parametric UMAP Embeddings for Representation and Semisupervised Learning. *Neural Computation*, 33(11):2881–2907.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social Bias Frames: Reasoning about Social and Power Implications of Language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Sherry B Schnake and Janet B Ruscher. 1998. Modern Racism as a predictor of the Linguistic Intergroup Bias. *Journal of Language and Social Psychology*, 17(4):484–491.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2020. Towards Controllable Biases in Language Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3239–3254, Online. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The Woman Worked as a Babysitter: On Biases in Language Generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, Hong Kong, China. Association for Computational Linguistics.

Teun A Van Dijk. 2009. *Society and Discourse: How Social Contexts Influence Text and Talk*. Cambridge University Press.

Sida Wang and Christopher Manning. 2012. Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, Jeju Island, Korea. Association for Computational Linguistics.

Wenbo Wang, Lu Chen, Krishnaprasad Thirunarayan, and Amit P. Sheth. 2012. Harnessing Twitter "Big Data" for Automatic Emotion Identification. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 587–592.

Albert Webson, Zhizhong Chen, Carsten Eickhoff, and Ellie Pavlick. 2020. Are “Undocumented Workers” the Same as “Illegal Aliens”? Disentangling Denotation and Connotation in Vector Spaces. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4090–4105, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Samira Zad, Joshuan Jimenez, and Mark Finlayson. 2021. Hell Hath No Fury? Correcting Bias in the NRC Emotion Lexicon. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 102–113, Online. Association for Computational Linguistics.

Semantic Parsing for Conversational Question Answering over Knowledge Graphs

Laura Perez-Beltrachini[1], **Parag Jain**[1], **Emilio Monti**[2], **Mirella Lapata**[1]
[1] School of Informatics, University of Edinburgh
[2] Amazon Alexa
{lperez,parag.jain,mlap}@inf.ed.ac.uk, monti@amazon.co.uk

Abstract

In this paper, we are interested in developing semantic parsers which understand natural language questions embedded in a *conversation* with a user and *ground* them to formal queries over definitions in a general purpose knowledge graph (KG) with very large vocabularies (covering thousands of concept names and relations, and millions of entities). To this end, we develop a dataset where user questions are annotated with SPARQL parses and system answers correspond to execution results thereof. We present two different semantic parsing approaches and highlight the challenges of the task: dealing with large vocabularies, modelling conversation context, predicting queries with multiple entities, and generalising to new questions at test time. We hope our dataset will serve as useful testbed for the development of conversational semantic parsers.[1]

1 Introduction

Conversational information seeking is the process of acquiring information through conversations (Zamani et al., 2022). Recent years have seen an increasing number of applications aiming to build conversational interfaces based on information retrieval (Radlinski and Craswell, 2017) and user recommendation (Jannach et al., 2021). The popularity of intelligent voice assistants such as Amazon's Alexa or Apple's Siri has further stimulated research on question answering over general purpose knowledge graphs (e.g., Wikidata). Key to question answering in this context is the ability to ground natural language onto concepts, entities, and relations in order to produce an *executable* query (e.g., SPARQL) which will retrieve an answer or *denotation* from the knowledge graph (KG).

This grounding process, known as *semantic parsing* has been studied in the context of one or few domain-specific databases (Yu et al., 2019a; Jain and Lapata, 2021; Suhr et al., 2018) or without taking the conversational nature of the task into account (Reddy et al., 2014; Yih et al., 2016; Dubey et al., 2019; Gu et al., 2021). However, due to the complexities of the semantic parsing task, there are no large scale datasets consisting of information seeking conversations with executable queries against a KG. Conversational semantic parsing over KGs requires handling very large vocabularies covering thousands of concept names and relations, and millions of entities rather than specialized terms consisting of hundreds of tables and column names. Moreover, information seeking conversations are by nature incremental involving interrelated rather than isolated questions.

In this work, we create 𝕊ℙ𝕀ℂ𝔼, a **S**emantic **P**ars**I**ng dataset for **C**onversational qu**E**stion answering over Wikidata. 𝕊ℙ𝕀ℂ𝔼 consists of user-assistant interactions where natural language questions are paired with SPARQL parses and answers provided by the system correspond to SPARQL execution results. We derive this dataset from CSQA (Saha et al., 2018), an existing benchmark originally proposed for retrieval-based conversational question answering (Lan et al., 2021). Although CSQA does not have executable queries, it contains a large number of natural language questions and their corresponding answers, highlighting a range of conversational phenomena such as coreference, ellipsis, and topic change as well as different types of questions exemplifying varying intents.

Table 1 shows a conversation from 𝕊ℙ𝕀ℂ𝔼 illustrating how questions (utterances on the left) are annotated with SPARQL queries (SP on the right blue box). To create a large-scale dataset (197k conversations), we develop SPARQL templates for different question intents; entity, relation, and class symbols are initially under-specified and subsequently filled automatically to generate full SPARQL queries. CSQA questions have been previously associated with logical forms generated with

---

[1] Our dataset and models are released at 𝕊ℙ𝕀ℂ𝔼.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2507–2522
May 2-6, 2023

	Utterances	Annotations	Actions and Semantic Parses

$\mathcal{T}_1$	U: *Which tournament did Detroit Tigers participate in?*
	S: *1909 World Series*
		INTENT=Simple Question|Single Entity
		ENT=[Q650855 (*Detroit Tigers*)] ,
		REL=[P1923 (*participating team*)],
		TYP=[Q500834 (*tournament*)],
		TRIPLE=[(Q500834,P1923,Q650855)],
		GOLD =[Q846847 (*1909 World Series*)]
			AS: [filter_type, find_rev, Q650855,P1923,Q500834]
			SP: `SELECT ?x WHERE {`
			`?x wdt:P1923 wd:Q650855 .`
			`?x wdt:P31 wd:Q500834 . }`

$\mathcal{T}_2$	U: *Which sports team was the champion of that tournament?*
	S: *Pittsburgh Pirates*
		INTENT=Simple Question|Single Entity|Indirect
		ENT=[Q846847 (*1909 World Series*)] ,
		REL=[P1346 (*winner*)],
		TYP=[Q12973014 (*sports team*)],
		TRIPLE=[(Q846847,P1346,Q12973014)],
		GOLD=[Q7199360 (*Pittsburgh Pirates*)]
			AS: [filter_type, find, Q846847, P1346, Q12973014]
			SP: `SELECT ?x WHERE {`
			`wd:Q846847 wdt:P1346 ?x .`
			`?x wdt:P31 wd:Q12973014 . }`

$\mathcal{T}_3$	U: *Does that sports team belong to Sacile?* S: *No*	INTENT=Verification|2 entities, subject is indirect ENT=[Q653772 (*Pittsburgh Pirates*), Q53190 (*Sacile*)], REL=[P17 (*country*)], TYP=[Q15617994 (*designation admin. territorial entity*)], TRIPLE=[(Q653772,P17,Q53190)], GOLD=[False]	AS: [is_in, Q53190, find, Q653772, P17] SP: `ASK {wd:Q653772` `wdt:P17 wd:Q53190 . }`

Table 1: Example conversations from SPICE. The left column shows dialogue turns ($\mathcal{T}_1$–$\mathcal{T}_3$) with user (U) and system (S) utterances. The middle column shows the annotations provided in CSQA. Blue boxes on the right show the sequence of actions (AS) and corresponding SPARQL semantic parses (SP).

custom-made grammars (Guo et al., 2018; Kacupaj et al., 2021; Marion et al., 2021). As a result, semantic parsers based on them operate with different sets of grammar rules and are not strictly comparable, since the grammars may have different coverage and semantics (e.g., terminal symbols may encapsulate different degrees of execution complexity). In SPICE, questions are represented with SPARQL, a standard query language for retrieving and manipulating RDF data.[2] This allows us to compare parsers developed on the dataset on an equal footing and facilitates further extensions (e.g., new question intents), without the need to redefine the grammar and its execution engine. In an attempt to build semantic parsers which generalise to new user questions, we further create different data splits where new intents appear only at test time (Finegan-Dollak et al., 2018).

For our semantic parsing task, we establish two strong baseline models which tackle the large vocabulary problem and the prediction of logical forms in different ways. The first approach (Gu et al., 2021) uses *dynamic vocabularies* derived from KG subgraphs for each question and a simple sequence-to-sequence architecture to predict *complete* SPARQL queries. The other approach (Kacupaj et al., 2021) predicts SPARQL *query templates* and then fills in entity, relation, and type slots by means of an entity and ontology classifier. Our experiments reveal several shortcomings in both approaches, such as not being able to encode large sets of KG elements and generate the same entity several times. Both approaches struggle with ellipsis, they cannot resolve coreference when the referent appears in the conversation context beyond the previous turn, have reduced performance on questions with multiple entities, and struggle with unseen question intents. We discuss these challenges and outline research directions for conversational semantic parsing.

2 The SPICE Dataset

The CSQA dataset (Saha et al., 2018) aims to facilitate the development of QA systems that handle complex and inter-related questions over a knowledge graph. In contrast to simple factual questions that can be answered with a single KG triple (i.e., {subject, relation, object}), complex questions require manipulating sets of triples and reasoning over these. In Table 1, a question like *How many sports teams participated in that tournament?* requires numerical reasoning and answering the question in turn $\mathcal{T}_2$ relies on correctly interpreting $\mathcal{T}_1$.

Questions and answers in this dataset were elicited from human experts playing user and system roles as well as from crowd-workers. In a second stage, templates derived from the human-authored QA pairs were used to automatically augment the dataset. Human experts also suggested complex reasoning questions and derived templates thereof. Conversations were built as sequences of

---

[2] https://www.w3.org/TR/sparql11-query/

Nb. instances	197K
Nb. entities	12.8M
Nb. relations	2738
Nb. types	3064
Avg. turn length	9.5
Avg. entities per conversation	7.6
Avg. types per conversations	6.5
Avg. neighbourhood per turn	181.4 triples
Logical Reasoning, Quantitative Reasoning, Comparative Reasoning, Quantitative Reasoning Count, Comparative Reasoning Count, Verification, Simple Question	
Clarification, Coreference, Ellipsis	

Table 2: $\mathbb{SPICE}$ statistics (top); general question types (middle); linguistic phenomena (bottom).

QA pairs exploring paths in the KG. By construction, the QA pairs in a conversation are connected through one or several entities in the KG. Questions fall into two coarse categories, *simple* and *reasoning*-based, and the way QA pairs are organised in a sequence introduces various *conversational phenomena* which we summarize below.

Simple Questions are factoid questions, seeking information related to an entity (e.g., *Which tournament did Detroit Tigers participate in?* in Table 1) or set of entities (e.g., *What are the countries of those sports teams?*).

Reasoning Questions are complex questions which require the application of numerical and logical operators over sets of entities. For instance, to answer the question *How many sports teams participated in that tournament?* requires finding the set of sports teams that participated in a given tournament (e.g., *1909 World Series*) and taking its count. Questions in this category also involve General Entities (GE) such as *tournament*, in addition to Named Entities (NE), and multiple entities (both NE and GE) in a single question (e.g., *Which tournaments have less number of participating sports teams than 1909 World Series?*). Some question types also combine multiple reasoning operators.

Conversations contain sequences of mixed-initiative interactions where the system requests clarification on ambiguous questions. Conversations also include discourse phenomena such as coreference (e.g., *Which sports team was the champion of that tournament?* in Table 1) and ellipsis (e.g., *And what about 1910 World Series?* as a follow up question to *How many sports teams participated in that tournament?*).

There are 10 question types and 47 question subtypes. In Table 2, we only list question types but provide all subtypes in Table 9 in Appendix A.

2.1 Question Semantics Described by Actions

Saha et al. (2018) envisaged CSQA as a benchmark for retrieval-based conversational question answering (Bordes et al., 2015; Dong et al., 2015; Jain, 2016; Lan et al., 2021). These methods embed natural language questions and KG triples into high dimensional spaces and rely on neural reasoning modules to match questions to candidate answers. Hence, questions do not have associated logical forms, only gold answers are available.

Our success in creating semantic parse annotations is partly due to the fact that CSQA provides useful KG information. Each interaction (i.e., user and system turn) comes with annotations about KG entities, types, and relation symbols as well as some information about the triple patterns involved in the question (illustrated in Table 1 with ENT, REL, TYP, and TRIPLE fields). It also provides information pertaining to question types and subtypes (see INTENT in Table 1).

Taking advantage of these annotations, follow-on work (Guo et al., 2018) defined a semantic parsing task over CSQA, modeling the meaning of questions as a sequence of actions. The set of actions encompasses find (or find_rev when the entity is in object position) to retrieve sets of entities in a subject (object) position, as well as actions operating on sets of entities (e.g., filter_type). For instance, the question in turn $\mathcal{T}_1$ in Table 1 would be parsed to [filter_type, find_rev, Q650855, P1923, Q500834], meaning "find the set of entities that are in relationship *participating team* with *Detroit Tigers* and then filter those that are of type *tournament*". A breadth-first search algorithm generates action-grammar annotations for each question and a sequence of grammar-actions is considered correct if upon execution it returns the gold answer. Subsequent work (Shen et al., 2019; Kacupaj et al., 2021; Marion et al., 2021) expanded this action-grammar greatly improving its coverage (i.e., the number of successfully annotated questions).

2.2 From Actions to SPARQL Queries

In this work, we take a step further and map CSQA natural language questions into vanilla SPARQL queries. We first analysed how intent is expressed

	ATIS	SParC	CoSQL	SPICE
Nb. Instances	1,658	4,298	3,007	197K
Avg. turn length	7.0	3.0	5.2	9.5
Domain	Single	Multi	Multi	Wikidata
Logical form	SQL	SQL	SQL	SPARQL
Database type	Rel	Rel	Rel	KG

Table 3: Conversational semantic parsing datasets (Rel: relational database; KG: knowledge graph).

in question types and subtypes and then manually defined SPARQL templates for each question subtype. A SPARQL template is a query with unspecified triple patterns in the WHERE clause. For instance, the template for the question in turn $\mathcal{T}_1$ is {SELECT ?x WHERE **triple**(?x, ENTITY, RELATION). ?x wdt:P31 TYPE.}. We finally modified the tool provided in Kacupaj et al. (2021) to automatically instantiate the SPARQL templates, providing annotations for the entire dataset (e.g., by filling missing slots and determining subject/object positions for **triple**($\cdot$) elements as SELECT ?x WHERE {?x wdt:P1923 wd:Q650855. ?x wdt:P31 wd:Q500834.}).

We imported the Wikidata snapshot provided by Saha et al. (2018) into a KG in a SPARQL server (see Appendix B for more details) and assessed the correctness of SPARQL queries by executing them and comparing results to gold answers. For some questions the annotation procedure did not produce a SPARQL parse that recovered the gold answer. In these rare cases, we redefined the answer if it did not affect the conversation flow or truncated the conversation up to that point.

Table 2 shows various statistics for SPICE while Table 3 compares it to related conversational datasets such as ATIS (Suhr et al., 2018), SParC (Yu et al., 2019b), and CoSQL (Yu et al., 2019a). As can be seen, SPICE contains a sizeable number of training instances, its conversations are longer, and the semantic parsing task is real-scale.

3 The Semantic Parsing Task

We consider the semantic parsing task over a sequence of dialogue turns $d = (d_1, d_2, \cdots, d_{|d|})$, where turn d_t corresponds to a user-system interaction with user question x_t and system answer a_t. Each turn has a conversation context c_t made of interactions d_i such that $i < t$. Given interaction d_t with context c_t and user question $x_t = (x_{t1}, x_{t2}, \cdots, x_{t|x_t|})$, our goal is to predict a SPARQL query $y_t = (y_{t1}, y_{t2}, \cdots, y_{t|y_t|})$ that represents the intent of x_t and, upon execution over knowledge-graph $\mathcal{K}$, yields denotation a_t. y_t is a sequence over a target vocabulary $\mathcal{V} = \mathcal{V}_f \cup \mathcal{V}_\mathcal{K}$ where $\mathcal{V}_f$ is fixed and contains SPARQL keywords (e.g., SELECT) and special tokens (e.g., beginning of sequence token, BOS), and $\mathcal{V}_\mathcal{K}$ contains all knowledge-graph symbols (e.g., entity IDs such as Q76 for *Barack Obama*).

We propose two approaches for this semantic parsing task which establish strong baseline performance and highlight various challenges. These differ in the way they handle large KG vocabularies and how they generate logical forms. Figure 1 provides a sketch of the two models discussed below.

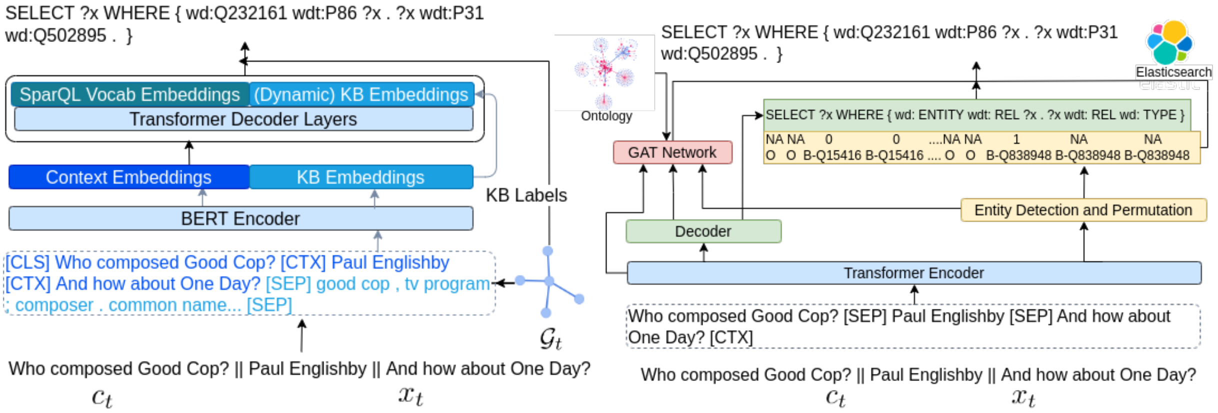
3.1 Parsing with a Single Decoder and Knowledge Subgraphs

Our first model is parameterised by an encoder-decoder Transformer neural network (Vaswani et al., 2017), and an adaptation of the semantic parsing architecture proposed in Gu et al. (2021).

Dynamic Vocabulary Since the KG vocabulary $\mathcal{V}_\mathcal{K}$ can be extremely large, we parse question x_t with a smaller vocabulary $\mathcal{V}_t \subseteq \mathcal{V}_\mathcal{K}$ which only contains KG symbols related to x_t. Following previous work (Gu et al., 2021; Marion et al., 2021), we assume the symbols related to x_t are those appearing in subgraph $\mathcal{G}_t$ of knowledge-graph $\mathcal{K}$, $\mathcal{G}_t \subseteq \mathcal{K}$. Given question x_t and its context c_t, we identify KG entities $\mathcal{E}_t = \{e_{t1}, e_{t2}, \cdots, e_{t|\mathcal{E}_t|}\}$ which correspond to mentions in x_t and c_t. We then obtain $\mathcal{G}_t$ by taking the one-hop neighbourhood for each entity $e_{ti} \in \mathcal{E}_t$. In other words, we include all KG triples (s, r, o) where the entity appears in subject ($s = e_{ti}$) or object position ($o = e_{ti}$). When e_{ti} is a subject, we include triple (e_{ti}, r, τ_o) where τ_o is the type of entity o; analogously, when e_{ti} appears in an object position, we add (τ_s, r, e_{ti}). For entities e_{ti} we include their types $\tau_{e_{ti}}$. When e_{ti} is a general entity (e.g., a type such as *tournament*) we add relations from $\mathcal{K}$ that have instances of type e_{ti} as their subject (object). The final vocabulary $\mathcal{V}_t$ contains all entities in $\mathcal{E}_t$, all relations r and types (τ_o, τ_s, and $\tau_{e_{ti}}$) found in the set of triples in $\mathcal{G}_t$.

Note that context c_t is defined as a window over the conversation so far. Following previous work (Marion et al., 2021; Kacupaj et al., 2021), we set the conversation context to the previous user-system interaction $c_t = \{d_{t-1}\}$.

Encoder-Decoder Model Our encoder is a BERT (Devlin et al., 2019) model fine-tuned on



(a) Encoder-decoder model with dynamic vocabularies.

(b) Semantic parser based on Lasagne multi-task model.

Figure 1: Two modeling approaches to conversational semantic parsing.

our semantic parsing task. The decoder is a randomly initialised Transformer network (Vaswani et al., 2017). To account for the difference in initialisation between the encoder and decoder networks, we follow the training scheme proposed in Liu and Lapata (2019). We provide details in Appendix C.

The input to our semantic parser is a tuple $(x_t, c_t, \mathcal{G}_t)$ consisting of natural language question x_t, its context c_t, and subgraph $\mathcal{G}_t$ which we adapt to BERT's input format as follows (Gu et al., 2021). We concatenate the sequence of natural language questions and answers appearing in c_t and x_t, using the special token [CTX] as a delimiter and prepend the [CLS] token in the beginning of the sequence. Special token [SEP] denotes the end of sequence followed by the linearised KG subgraph $\mathcal{G}_t$. The linearisation procedure goes over entities in $\mathcal{G}_t$, enumerating their types and relations. Importantly, we denote entities by their label rather than their KG identifiers. The order of entities in $\mathcal{G}_t$ is random. Figure 1(a) shows an example of the input to our BERT-based encoder.

More formally, the encoder takes token sequences $x'_t = \text{[CLS]}x'_{t_{\text{text}}}\text{[SEP]}x'_{t_{\text{graph}}}\text{[SEP]}$ as input where $x'_{t_{\text{text}}}$ is the natural language subsequence and $x'_{t_{\text{graph}}} = (g^t_1, \cdots, g^t_{|\mathcal{G}_t|})$ is the sequence of knowledge-graph symbols from the linearised graph $\mathcal{G}_t$. Note that these knowledge-graph symbols constitute the target dynamic vocabulary $\mathcal{V}_t$ and $|\mathcal{G}_t|$ represents the number of KG symbols which is equal to the size of the target vocabulary $|\mathcal{V}_t|$. The encoder maps input sequences x'_t into sequences of continuous representations $\mathbf{z_t} = (\mathbf{z}_{t1}, \ldots, \mathbf{z}_{t|x_t|})$, and the decoder then generates the target SPARQL parse $y_t = (y_{t1}, \ldots, y_{t|y_t|})$ token-by-token autoregressively, hence modelling the conditional probability: $p(y_{t1}, \ldots, y_{t|y_t|}|x'_t)$.

The linearised graph $\mathcal{G}_t$ can exceed BERT's maximum number of input positions (which is 512). To avoid throwing away useful information, we adopt a solution similar to Gu et al. (2021). For question x_t with $\mathcal{G}_t$ containing k entities, we create k input sequences $x_t'^1, \cdots, x_t'^k$. These k sequences share the natural language subsequence but have different KG symbol subsequences. Given an input sequence $x_t'^1, \cdots, x_t'^k$, we obtain contextualised representations as $\mathbf{z}_t^1, \cdots, \mathbf{z}_t^k = \text{BERT}(x_t'^1, \cdots, x_t'^k)$.

The model further splits the sequence of continuous representations $\mathbf{z}_t^j$ into textual representations $\mathbf{z}^j_{t_{\text{text}}}$ and knowledge-graph symbols $\mathbf{z}^j_{t_{\text{graph}}}$ both of which are contextualised. We then average representations $\mathbf{z}_{t_{\text{text}}} = \text{AVG}(\mathbf{z}^j_{t_{\text{text}}})$ and feed them as input to the decoder (see Figure1(a)). From representations $\mathbf{z}^j_{t_{\text{graph}}}$, we derive the embeddings for the elements in the target dynamic vocabulary $\mathcal{V}_t$. Recall that the decoder parses input questions x_t using target vocabulary $\mathcal{V} = \mathcal{V}_f \cup \mathcal{V}_t$ which consists of a set of fixed ($\mathcal{V}_f$) and dynamic ($\mathcal{V}_t$) target tokens. The decoder then predicts the probability of each SPARQL token y_{ti} as $p(y_{t_i}|y_{t_{<i}}, x_t'^1, \cdots, x_t'^k) = \text{softmax}(\mathbf{W}_o \mathbf{h}_i^L)$ where $\mathbf{h}_i^L$ is the decoder top layer hidden representation at time step i. $\mathbf{W}_o \in \mathbb{R}^{|\mathcal{V}_f \cup \mathcal{V}_t|}$ is the output embedding matrix with $\mathbf{W}_o = [\mathbf{W}_f; \mathbf{W}_t]$, where $[;]$ denotes matrix concatenation, $\mathbf{W}_f$ is the embedding matrix for the fixed target vocabulary, and $\mathbf{W}_t$ is derived from the encoder representations $\mathbf{z}^j_{t_{\text{graph}}}$.

3.2 Parsing with Multiple Decoders and an Ontology Classifier

Our second model is an adaptation of the Lasagne architecture proposed in Kacupaj et al. (2021).

Lasagne generates logical forms following a multi-stage approach where a backbone sketch is first predicted and then fleshed out. Their sketch is a sequence of actions from a custom grammar which we modify to be a sketch of SPARQL queries.

SPARQL Template Prediction Lasagne employs an encoder-decoder model based on Transformers (Vaswani et al., 2017) to convert a user question x_t in a conversation into a logical form template. The input to the encoder is the conversation context $c_t = \{d_{t-1}\}$ and user question x_t. Utterances are separated via [SEP] tokens, while the special context token [CTX] denotes the end of sequence (see Figure 1b). The input sequence is encoded via multi-head attention (Vaswani et al., 2017) to output contextualized representations which are then fed to the decoder to predict a sequence of actions (without grounding to KG elements) token-by-token. Instead, our decoder predicts SPARQL queries with place-holders for KG symbols. For instance, for the WHERE clause of turn $\mathcal{T}_1$ in Table 1, it predicts {ENTITY RELATION ?x. ?x wdt:P31 TYPE.} instead of {wd:Q5582479 wdt:P161 ?x. ?x wdt:P31 wd:Q502895.})

Entity Recognition and Linking An entity recognition module detects entities in the input and links them to the KG (Shen et al., 2019; Kacupaj et al., 2021). Initially, entity spans are identified using an LSTM which performs BIO sequence labelling.[3] Entity spans are subsequently linked to KG entities via an inverted index (created using Elasticsearch[4]) which maps entity labels to entity IDs. Once identified, the entities are further filtered and reordered so that they match their order of appearance in the SPARQL (see Figure 1(b)).

Predicting Types and Relations Finally, an ontology graph with types and relations appearing in $\mathbb{SPICE}$'s KG is constructed.[5] The graph is encoded with a Graph Attention Network (GAT; Velickovic et al. 2018) and the prediction of type and relation fillers for the SPARQL template is modeled as a classification task over graph nodes, given the conversational context and the decoder hidden state.

Learning All modules outlined above are trained in a multi-task manner, optimizing the weighted average of the following individual losses $L = \lambda_1 L^{\mathrm{F}} + \lambda_2 L^{\mathrm{G}} + \lambda_3 L^{\mathrm{R}} + \lambda_4 L^{\mathrm{O}}$ where L^{F} is the loss of the SPARQL template decoder, L^{G} is the type and relation prediction loss using the GAT network, L^{R} is the entity recognition loss, and L^{O} the entity reordering loss (and weights $\lambda_{1:4}$ are learned during training). We refer the interested reader to Kacupaj et al. (2021) for mode details.

4 Results

We examine how the two models just described fare on different question types and subtypes. We report results on $\mathbb{SPICE}$ i.i.d train/valid/test splits (containing 152,391/16,813/27,797 conversations, respectively) but also create new splits that assess out-of-distribution generalisation. In all cases, following previous work (Saha et al., 2018; Kacupaj et al., 2021), we use execution-based automatic metrics. Micro *F1-score* evaluates question parses that return a set of entities, while *Accuracy* is used for question parses that evaluate to True/False or return a numerical value. In addition, we report *Exact Match* (EM) against the gold SPARQL parse.

4.1 Performance per Question Type

Table 4 shows our results on the $\mathbb{SPICE}$ i.i.d test split. BertSP variants differ in how they obtain the set of KG entities $\mathcal{E}_t$ (cf. Section 3.1) to build the dynamic vocabulary. BertSP$_{\mathrm{G}}$ has access to oracle entities, types, and coreference annotations which allows us to disassociate the complexity of the SPARQL generation task from the problem of grounding and disambiguating entities to KG symbols. Variants BertSP$_{\mathrm{S}}$ and BertSP$_{\mathrm{A}}$ do not have access to oracle annotations. BertSP$_{\mathrm{S}}$ grounds mentions to KG entities with a simple algorithm based on string matching (Marion et al., 2021); while BertSP$_{\mathrm{A}}$ relies on AllenNLP's Named Entity Recognizer (NER) and the Elasticsearch inverted index for Named Entity Linking (NEL). Both have to identify coreferring entities using the conversation context c_t. Both BertSP$_{\mathrm{S}}$ and BertSP$_{\mathrm{A}}$ use string matching for type linking (i.e., grounding general entities to KG types).

Note that it is not straightforward to perform oracle analysis for LasagneSP without compromising the model structure which predicts entities, their types, and relations in multiple stages.

Exact Match Performance We observe that execution based metrics (F1-score and Accuracy) are generally higher than EM. This is because in some

---

[3] BIO labels for training are obtained by preforming string matching between entity annotations and user utterances.

[4] https://www.elastic.co/

[5] This graph would be substantially bigger for a semantic parsing system operating over the full Wikidata KG.

	BertSP$_\text{G}$		BertSP$_\text{S}$		BertSP$_\text{A}$		LasagneSP	
Question Type	F1	EM	F1	EM	F1	EM	F1	EM
Clarification	84.89	82.53	80.21	**77.69**	83.91	76.58	86.29	73.41
Logical Reasoning (All)	90.61	82.90	85.55	**66.89**	22.74	28.61	88.80	57.41
Quantitative Reasoning (All)	94.42	88.55	82.95	66.40	76.20	59.01	94.90	**91.47**
Comparative Reasoning (All)	96.23	87.39	90.44	73.80	69.56	39.37	94.20	**85.05**
Simple Question (Coreferenced)	88.96	86.53	83.19	**69.87**	76.51	58.83	84.73	60.90
Simple Question (Direct)	91.81	91.59	87.13	**80.69**	71.43	58.71	87.21	66.88
Simple Question (Ellipsis)	79.51	89.71	72.50	**71.67**	58.14	50.90	74.35	61.53
	AC	EM	AC	EM	AC	EM	AC	EM
Verification (Boolean)	90.10	77.24	79.72	**62.62**	37.16	24.90	34.89	26.72
Quantitative Reasoning (Count)	87.91	84.97	76.88	**73.20**	50.86	48.44	60.51	56.15
Comparative Reasoning (Count)	90.05	85.99	73.18	66.79	43.48	40.67	89.09	**83.69**
Overall	81.50	85.74	81.18	**70.96**	59.00	48.60	79.50	66.32

Table 4: Accuracy (AC), and exact match (EM) on SPICE i.i.d test split. BertSP$_\text{G}$ has access to oracle entities, types, and coreference annotations. Best EM predictions are shown in bold.

cases the SPARQL parse may be incorrect and still yield some results. For instance, a parse requiring the UNION of two graph patterns may yield a partially correct answer by only including one graph pattern; similarly, a parse can evaluate to False and agree with the gold answer just because it included a wrong relation symbol.

The Importance of Entity Grounding Not surprisingly, the model with access to oracle information (variant BertSP$_\text{G}$) obtains the best performance. Results improve not only for questions with entities referring to previous context but also indirectly for other types of questions. Since entities are correctly grounded in previous conversation turns c_t, the model operates with more accurate graphs $\mathcal{G}_t$ and richer dynamic vocabularies $\mathcal{V}_t$.

Both BertSP$_\text{S}$ and BertSP$_\text{A}$ perform coreference resolution using limited conversation context and thus performance decreases. These models also have to ground named (*Detroit Tigers*) and general (*tournament*) entity mentions to KG symbols. BertSP$_\text{S}$ which relies on string matching performs overall better than BertSP$_\text{A}$ which struggles with compound named entities such as *President of the Czech Republic*) and disambiguation during NEL (e.g., *Saint Barbara* the painting versus the Saint).

Model Comparison BertSP$_\text{S}$ and LasagneSP are similar in the way they handle NER/NEL with a task-specific approach, but differ in their conceptualization of the semantic parsing task (encoder-decoder vs. multi-tasking). LasagneSP outperforms BertSP$_\text{S}$ in Comparative, Quantitative, and Comparative-Count questions. These encompass many question subtypes with general entities which are very common in both training and testing. LasagneSP has access to *all* types and relations encoded with the graph network. In contrast, BertSP$_\text{S}$ relies on types which in the first place need to be present in the entity neighbourhood subgraph and then be preserved after truncating the input to fit the model's maximum sequence length. An advantage of BertSP$_\text{S}$ over LasagneSP, is that it allows for easier adaptation to new types and relations by relying on dynamic vocabularies, while LasagneSP would need to be retrained to accommodate them.

In Simple questions, where each question involves fewer but more diverse types, BertSP$_\text{S}$ predicts more accurate types (thanks to the input text and KG symbol contextualisation) and thus performs better. LasagneSP does poorly on Verification, Logical, and Quantitative-Count (which includes logical operators). This can be explained by a modelling limitation, i.e., it is not able to point to the same input entity more than once.

Errors in Predicted SPARQLs Manual inspection of SPARQL predictions revealed several common system errors including: prediction of erroneous entities and relations, failure to enumerate all required entities (for questions with multiple entities), and mistakes in argument order (i.e., entities and variables are correct but placed in incorrect subject/object positions). To a lesser extent, we also observed SPARQL queries with incorrect intent predictions and ill-formed syntax.

Phenomena	BertSP$_G$	BertSP$_S$	BertSP$_A$	LasagneSP
Coref$_{=-1}$	81.40	70.65	49.39	43.65
Coref$_{<-1}$	67.82	0	0	0
Ellipsis	75.93	54.33	26.39	46.54
Entities	83.37	65.40	41.64	66.52

Table 5: Exact match (EM) on 𝕊ℙ𝕀ℂ𝔼 i.i.d test set; questions are grouped into linguistic phenomena.

Unseen Combinations (Train/valid/test)	BertSP$_S$ EM	LasagneSP EM
COUNTLOGIC	0.94	0
UNIONMULTI	19.74	16.89
VERIFY3	0	0

Table 6: Exact match (EM) for BertSP$_S$ and LasagneSP on 𝕊ℙ𝕀ℂ𝔼 non-i.i.d splits.

4.2 Linguistic Analysis

Table 5 shows model performance across-different question subtypes aggregated for specific phenomena. These include Coreference, Ellipsis, and Multiple Entities (Entities). We distinguish between cases where coreference can be resolved in the previous turn (Coref$_{=-1}$) and further back in the conversation history (Coref$_{<-1}$). In addition, some question subtypes contain plural mentions, i.e., they are linked to multiple entities which the semantic parser must enumerate in order to build the correct parse. Ellipsis can be often resolved within the previous interaction (Coref$_{=-1}$), but not within the wider discourse context. Questions with multiple entities bring further disambiguation challenges. In Appendix A, we provide the list of question subtypes for each phenomenon in Table 5.

As can be seen, the oracle BertSP$_G$ model which has access to gold annotations is superior to variants which rely on automatic entity and type linking. BertSP$_S$ is better than LasagneSP at handling coreference within immediate context (i.e., $c_t = \{d_{t-1}\}$). Due to the fact that LasagneSP predicts entity positions in SPARQL, it is particularly bad at resolving mentions to multiple entities in the previous context or even multiple mentions of the same entity in the output parse (as is the case with Verification questions). Perhaps unsurprisingly, neither BertSP$_S$ nor LasagneSP can resolve mentions to non-immediately preceding utterances. BertSP$_S$ performs better than LasagneSP in questions with ellipsis; we conjecture that the input context and contextualisation of KG symbols help in grounding elided relation mentions. Ellipsis and multiple entities improve by a large margin with access to gold annotations (see BertSP$_G$ in Table 5).

4.3 Generalisation

We further evaluate the models' ability to generalise by creating "query-based" splits (Finegan-Dollak et al., 2018), i.e., splits with query patterns seen only at test time. Our splits include: (a) question subtypes that involve a count operation over a union operator (COUNTLOGIC; individual operators are seen at training time but not the combination thereof); this split has 153,562/14,262/29,177 instances for training/validation/testing; (b) question subtypes that involve a union operator over two graph patterns with different relations (UNIONMULTI; the union of two graph patterns with the *same* relation is seen during training); this split contains 157,331/14,426/25,244 instances; and (c) verification questions with three entities involving 154,027/13,869/29,105 instances (VERIFY3; questions with 2 entities only are seen during training).

As shown in Table 6, both BertSP$_S$ and LasagneSP perform poorly across different splits. While in some cases the models grasp the overall SPARQL structure for unseen questions (e.g., *Which watercourses are located in the neighbourhood of Bremen or are the tributaries of Ob? in* UNIONMULTI), they ignore specific query details and simply default to familiar patterns seen in the training (e.g., *Which people are the creators of The Theory of Everything or Ten Minutes to Live?*). In the UNIONMULTI split, the models produce an appropriate SPARQL template but systematically copy the *same* relation in *both* graph patterns. BertSP$_S$ performs slightly better than LasagneSP; we hypothesize that contextualised KG embeddings occasionally help the model select different relations. We observe a similar trend for COUNTLOGIC and VERIFY3. Appendix D shows examples of unseen questions, their SPARQLs, and common errors.

5 Related Work

Much previous work on semantic parsing focuses on mapping stand-alone utterances to logical forms. Relatively few datasets have been constructed for *conversational* semantic-parsing (Suhr et al., 2018; Dahl et al., 1994; Yu et al., 2019b,a) partly due to the difficulty of eliciting annotations in an interactive context. As a result, existing benchmarks are either single-domain or small-scale (see the comparison in Table 3). For instance, although ATIS (Suhr et al., 2018) exemplifies several challenging long-range discourse phenomena (Jain and Lapata, 2021), it is restricted to a single domain with a simple database schema. SParC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a) present cross-domain challenges in mapping natural language queries onto SQL, but the conversation length is fairly short and the databases relatively small-scale.

Large KGs, like Wikidata (Vrandečić, 2012), are becoming an increasing valuable source of information. Various question-answering datasets have been recently released (Dubey et al. 2019; Talmor and Berant 2018; Christmann et al. 2019, 2022, *inter alia*), which are either not conversational or contain sequences of dialogue turns but the questions are not annotated with executable queries like SPARQL. The CSQA dataset introduced in Saha et al. (2018) is conversational and covers a wide range of linguistic phenomena (e.g., ellipsis, coreference) but frames the QA task as an information retrieval problem. Follow-on work (Marion et al., 2021; Kacupaj et al., 2021; Shen et al., 2019) has used hand-crafted grammars to automatically obtain semantic annotations which are are not executable with a real KG engine (e.g., Blazegraph), and cumbersome to extend to new question intents.

6 Conclusion

In this work we introduce SPICE, a conversational semantic parsing dataset over knowledge graphs. Our dataset contains SPARQL annotations which are executable on a real KG engine and requires handling complex questions, type, relation, and entity linking on a large scale. Moreover, it showcases multiple linguistic phenomena such as coreference and ellipsis. We establish two strong baselines for the semantic parsing task and present detailed analysis stratifying performance by question type and linguistic phenomena. We also study generalisation to unseen intents and create multiple dataset splits with different query patterns. To move forward conversational semantic parsing over large scale KGs would need to improve entity linking, modelling of conversation context, and generalisation capabilities. We hope our dataset will serve as a useful testbed for the development of conversational semantic parsers.

7 Acknowledgments

We thank the anonymous reviewers for their feedback. We gratefully acknowledge the support of Amazon and the UK Engineering and Physical Sciences Research Council (grant EP/W002876/1).

8 Limitations

Both models discussed in this work make simplifying assumptions. BertSP variants need to truncate the linearised graphs for computational cost reasons. LasagneSP works with a smaller graph ontology which can easily fit in memory. However, this restricts the model to predicting seen types or relations which is unrealistic. A real-world semantic parser should ideally have access to the full Wikidata. Our results show that both models do not generalise to unseen question intents, which is a known limitation of current neural sequence-to-sequence architectures (Furrer et al., 2020; Finegan-Dollak et al., 2018; Keysers et al., 2020; Kim and Linzen, 2020; Li et al., 2021). Finally, our results also suggest that there is scope for improvement in handling previous context (including questions and answers).

References

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

Philipp Christmann, Rishiraj Saha Roy, Abdalghani Abujabal, Jyotsna Singh, and Gerhard Weikum. 2019. Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 729–738, Beijing, China. ACM.

Philipp Christmann, Rishiraj Saha Roy, and Gerhard Weikum. 2022. Conversational question answering on heterogeneous sources. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 144–154, Madrid, Spain. ACM.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

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

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over Freebase with multi-column convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 260–269, Beijing, China. Association for Computational Linguistics.

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *The Semantic Web – ISWC 2019*, pages 69–78, Cham. Springer International Publishing.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Daniel Peter Furrer, Marc van Zee, Nathan Scales, and Nathanael Schärli. 2020. Compositional generalization in semantic parsing: Pre-training vs. specialized architectures. *arXiv e-prints*, page arXiv:2007.08970.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond i.i.d.: Three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, WWW '21, page 3477–3488, New York, NY, USA. Association for Computing Machinery.

Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2018. Dialog-to-action: Conversational question answering over a large-scale knowledge base. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Parag Jain and Mirella Lapata. 2021. Memory-Based Semantic Parsing. *Transactions of the Association for Computational Linguistics*, 9:1197–1212.

Sarthak Jain. 2016. Question answering over knowledge base using factual memory networks. In *Proceedings of the NAACL Student Research Workshop*, pages 109–115, San Diego, California. Association for Computational Linguistics.

Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Comput. Surv.*, 54(5).

Endri Kacupaj, Joan Plepi, Kuldeep Singh, Harsh Thakkar, Jens Lehmann, and Maria Maleshkova. 2021. Conversational question answering over knowledge graphs with transformer and graph attention networks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 850–862, Online. Association for Computational Linguistics.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4483–4491. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Yafu Li, Yongjing Yin, Yulong Chen, and Yue Zhang. 2021. On compositional generalization of neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4767–4780, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Pierre Marion, Pawel Nowak, and Francesco Piccinno. 2021. Structured context and high-coverage grammar for conversational question answering over knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8813–8829, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8026–8037. Curran Associates, Inc.

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, page 117–126, New York, NY, USA. Association for Computing Machinery.

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.

Amrita Saha, Vardaan Pahuja, Mitesh Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 705–713, New Orleans, Louisiana, USA. AAAI Press.

Tao Shen, Xiubo Geng, Tao Qin, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-task learning for conversational question answering over a large-scale knowledge base. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2442–2451, Hong Kong, China. Association for Computational Linguistics.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. 2018. Learning to map context-dependent sentences to executable formal queries. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2238–2249, New Orleans, Louisiana. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Denny Vrandečić. 2012. Wikidata: A new platform for collaborative data collection. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12 Companion, page 1063–1064, New York, NY, USA. Association for Computing Machinery.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019a. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.

Hamed Zamani, Johanne R. Trippas, Jeff Dalton, and Filip Radlinski. 2022. Conversational information seeking. Https://arxiv.org/abs/2201.08808.

A The SPICE Dataset: Question Types and Subtypes

Table 9 provides the list of question types and subtypes in SPICE. For each question subtype we provide an example user question. For cases involving ellipsis and coreference, we include the conversation context (in grey colour).

Table 7 provides the list of question subtypes grouped grouped according to linguistic phenomena. Coreference ($_{=-1}$ and $_{<-1}$), Ellipsis, and Multiple Entities.

Coreference
More/Less | Mult. entity type (Coreference) # More/-Less | Single entity type (Coreference) # Single Entity (Coreference) # 2 entities, one direct and one indirect, object is indirect # 2 entities, one direct and one indirect, subject is indirect # 3 entities, 2 direct, 2(direct) are query entities, subject is corefered # one entity, multiple entities (as object) corefered # Count | Logical operators (Coreference) # Count | Single entity type (Coreference) # Count over More/Less | Mult. entity type (Coreference) # Count over More/Less | Single entity type (Coreference)
Ellipsis
Difference | Single Relation (Ellipsis) # Intersection | Single Relation (Ellipsis) # Union | Single Relation (Ellipsis) # More/Less | Mult. entity type (Ellipsis) # More/Less | Single entity type (Ellipsis) # object parent is changed, subject and predicate remain same # Incomplete count-based ques # Count over More/Less | Mult. entity type (Ellipsis) # Count over More/Less | Single entity type (Ellipsis)
Multiple Entities
Difference | Multiple Relation # Intersection | Multiple Relation # Union | Multiple Relation # Atleast/ Atmost/ Approx. the same/Equal | Mult. entity type # Min/Max | Mult. entity type # More/Less | Mult. entity type # More/Less | Mult. entity type (Ellipsis) # More/Less | Mult. entity type (Coreference) # Mult. Entity (Simple Question Direct and Coreference) # one entity, multiple entities (as object) coreferred # Count over Atleast/ Atmost/ Approx. the same / Equal | Mult. entity type # Count | Mult. entity type # Count over More/Less | Mult. entity type # Count over More/Less | Mult. entity type (Ellipsis) # Count over More/Less | Mult. entity type (Coreference)

Table 7: Question subtypes grouped according to linguistic phenomena.

Table 8 provides the list of unseen question subtypes for each of the non-i.i.d splits.

CountLogic
Count | Logical operators # Count | Logical operators (Coreference)
UnionMulti
Union | Multiple Relation
Verify3
3 entities, 2 direct, 2(direct) are query entities, subject is indirect # 3 entities, all direct, 2 are query entities

Table 8: Unseen question subtypes in SPICE non-i.i.d splits.

B Creating a Knowledge Graph from the CSQA Data

Deploying a full copy of Wikidata locally as a standalone service requires huge resources in addition to cluster dependent tweaking to obtain fast query processing and high-performance.[6] To enable easier deployment and fast access for research purposes we created a smaller graph from the CSQA data files. We mapped the contents of these files[7] onto triples which we subsequently converted to `ttl` format[8] with full URI to allow loading them to the KG query engine. We also filled missing information where possible, for example, missing relations such as "instance of" was filled with relation P31 and added data type information when this was omitted from the original files.

We used Blazegraph[9] to deploy the local server, which uses only CPU-based resources and has access to 150G of RAM. Further details along with the script to host the server will be released upon acceptance.

C BertSP Model Configuration

Our model is implemented using pytorch (Paszke et al., 2019). For all experiments, we used the ADAM optimizer (Kingma and Ba, 2015) with 20,000 BERT warmup steps and 10,000 steps for decoder warm up. We use separate optimizers for the BERT encoder and decoder. BERT was fine-tuned during training with the initial learning rate set to 0.00002. A learning rate of 0.001 was set for the rest of model parameters. Our model was trained for 100,000 steps; we used 4 GPUs with 12GB of memory. We performed model selection on the validation set. We report results with the best performing model which had 6 decoder layers.

D Examples on Generalisation Splits

Table 10 shows examples from the generalisation splits: similar question subtypes see during train-

---

[6] https://www.mediawiki.org/wiki/Wikidata_Query_Service/User_Manual#Standalone_service

[7] Described at https://amritasaha1812.github.io/CSQA/download_CQA/

[8] http://www.w3.org/TR/turtle/

[9] https://blazegraph.com/

ing, unseen question subtype, and error on prediction. The most common error across different splits is that models use similar SPARQLs seen during training but fail to adapt them to the details (entities, types, relations, argument positions) in the unseen question subtype. Other errors encompass using the incorrect SPARQL query (incorrect question intent) and incorrect entities and types.

Table 9: Full list of question subtypes (intents) in SPICE. For each subt-type we show an example question, whenever the question subtype involves a conversational phenomenon (coreference or ellipsis); previous conversation interactions necessary for the interpretation of the question are shown in grey.

Question Subtype	Example User Question
Clarification	
Simple Question | Single Entity (Coreference)	U: *Which political territory is that sporting event located in?* S: *Did you mean Speed skating at the 2010 Winter Olympics – Men's 500 metres?* U: *Yes*
Logical Reasoning (All)	
Difference | Multiple Relation	U: *Which people were awarded with Order of Merit for Arts and Science and are not working as singer?*
Difference | Single Relation	U: *Which international organizations had Poland but not Bulgaria as their member?*
Difference | Single Relation (Ellipsis)	U: *Which city was Pierre Laffont born in?* S: *Marseille* U: *Which administrative territories are the sister cities of that city?* S: *Shanghai, Odessa, Naples* U: *But not Bologna*
Intersection | Multiple Relation	U: *Which human settlements are situated close to Trave and have an adjacent border with Herzogtum Lauenburg?*
Intersection | Single Relation	U: *Which works of art were filmed at Edinburgh and Berlin?*
Intersection | Single Relation (Ellipsis)	U: *Which language does José María Lassalle speak in?* S: *Spanish* U: *And also Sergio Gil*
Union | Multiple Relation	U: *Which watercourses are located in the neighbourhood of Bremen or are the tributaries of Ob?*
Union | Single Relation	U: *Which people are the creators of The Theory of Everything or Ten Minutes to Live?*
Union | Single Relation (Ellipsis)	U: *What is the profession of Mai Yamada?* S: *announcer* U: *Or Kazimierz Rogoyski?*
Quantitative Reasoning (All)	
Min/Max | Single entity type	U: *Which musical instruments are played by min number of people?*
Min/Max | Mult. entity type	U: *Which organizations are the main building contractors of max number of architectural structures and buildings?*
Atleast/ Atmost/ Approx. the same/Equal | Single entity type	U: *Which musical instruments are played by exactly 5 people?*
Atleast/ Atmost/ Approx. the same/Equal | Mult. entity type	U: *Which events are demonstrated in atleast 3 prints and genres of sculpture?*
Comparative Reasoning (All)	
More/Less | Mult. entity type	U: *Which landforms are known for containing lesser number of chemical compounds or minerals naturally than Stetind pegmatite?*
More/Less | Mult. entity type (Ellipsis)	U: *Which landforms are known for containing lesser number of chemical compounds or minerals naturally than Stetind pegmatite?* S: *Euboea, Izalco, Mount Nyiragongo* U: *And also tell me about Tuften quarry?*
More/Less | Mult. entity type (Coreference)	U: *Which administrative territory is that person a civilian of?* S: *Spain* U: *Which administrative territories are the countries of origin of lesser number of television programs or works of art than that administrative territory?*
More/Less | Single entity type	U: *Which television programs have been dubbed by more number of people than Puss in Boots: The Three Diablos?*
More/Less | Single entity type (Ellipsis)	U: *Which television programs have been dubbed by more number of people than Puss in Boots: The Three Diablos?* S: *House, K-On!, K-On!!* U: *And also tell me about Chip 'n Dale Rescue Rangers?*
More/Less | Single entity type (Coreference)	U: *Which languages are max number of literary works composed in?* S: *English* U: *Which languages are the mother tongues of less number of people than that language?*
Simple Question (Direct)	
Simple Question	U: *Which type of sport did Amel Tuka participate in?*
Single Entity	U: *What is the capital of Sweden?*
Mult. Entity	U: *Who were the writers of On being and essence, De vegetabilis et plantis libri septem and Historia de regibus Gothorum, Vandalorum et Suevorum?*
Simple Question (Ellipsis)	
only subject is changed, parent and relation remains same	U: *Which organizations are the sponsors of Janice Anderson?* S: *Montrail, Patagonia, Inc.* U: *And also tell me about Manikala Rai?*
object parent is changed, subject and relation remain same	U: *Which watercourses are situated nearby Munich?* S: *Eisbach, Würm, Isar* U: *And which river?*
Simple Question (Coreference)	
Mult. Entity	U: *Which releases have Motown as their record label?* S: *What's Going On, Got to Be There, Can't Slow Down* U: *Which genre do those releases belong to?*
Single Entity (Coreference)	U: *Which narrative location is The Penalty set in?* S: *San Francisco* U: *Which color is associated with that film genre?*
Verification (Boolean) (All)	
2 entities, both direct	U: *Is Zugspitze located in Germany?*
2 entities, one direct and one corefered, object is corefered	U: *Which university was Eden Stiles educated at?*

Continued on next page

Table 9 – *Continued from previous page*

Question Type/Subtype	Example User Question
	S: *University of Michigan* U: *And what about C. V. Raman?* S: *University of Madras* U: *Was Ravindra Wijegunaratne educated at that university?*
2 entities, one direct and one corefered, subject is corefered	U: *Which German business organization was Gustav Peichl a member of?* S: *Academy of Arts, Berlin* U: *What was designed by that person?* S: *Millennium Tower* U: *Does that tower block belong to Austria?*
3 entities, 2 direct, 2(direct) are query entities, subject is corefered	U: *Which administrative territory was Gary Collier born at?* S: *Fort Worth* U: *Is that administrative territory a sister city of Adamsville, New Brunswick and Yuen Long Kau Hui?*
3 entities, all direct, 2 are query entities	U: *Is Aix-en-Provence partner town of Baton Rouge and Hemmatabad, Alborz?*
one entity, multiple entities (as object) coreferred	U: *Which armed conflicts are Battle of the Argeş or Battle of the Yellow Sea a part of?* S: *Romania during World War I, Russo-Japanese War* U: *Did those armed conflicts fight in Rui Natsukawa?*
Quantitative Reasoning (Count) (All)	
Incomplete count-based ques	U: *How many people influenced Chris Marker?* S: *1* U: *And also tell me about Ada Yonath?* S: *1* U: *And what about Mikhail Bakunin?*
Count over Atleast/ Atmost/ Approx. the same/Equal|Mult. entity type	U: *How many cities are the terminus locations of atleast 5 thoroughfares and roads?*
Count over Atleast/ Atmost/ Approx. the same/Equal|Single entity type	U: *How many musical instruments are played by exactly 2 people?*
Count | Logical operators	U: *How many bodies of water or watercourses are situated nearby Lübeck?*
Count | Logical operators (Coreference)	U: *Which administrative territory is the native country of Carolina Goic Boroevic?* S: *Chile* U: *Who is the head of the government of that administrative territory?* S: *Michelle Bachelet* U: *What is the capital of that administrative territory?* S: *Santiago* U: *How many capitals or cities are sister towns of that city?*
Count | Mult. entity type	U: *How many people starred in Django Kill or Shatterday?*
Count | Single entity type	U: *How many people starred in Captain America: Civil War?*
Count | Single entity type (Coreference)	U: *Which armed conflict did Lionel of Antwerp, 1st Duke of Clarence take part in?* S: *Hundred Years' War* U: *How many people did that armed conflict engage in?*
Comparative Reasoning (Count) (All)	
Count over More/Less | Mult. entity type	U: *How many administrative territories have adopted lesser number of holidays and people as patron saint than Santo Stefano al Mare?*
Count over More/Less | Mult. entity type (Ellipsis)	U: *How many administrative territories have adopted lesser number of holidays and people as patron saint than Santo Stefano al Mare?* S: *296* U: *And what about San Donato Milanese?*
Count over More/Less | Mult. entity type (Coreference)	U: *Which administrative territories are Luigi Einaudi the head of state of and have UTC+01:00 as their time zone?* S: *Italy* U: *How many administrative territories are the origins of greater number of literary works or releases than that administrative territory?*
Count over More/Less | Single entity type	U: *How many legislatures represent lesser number of states than East Bengal Legislative Assembly?*
Count over More/Less | Single entity type (Ellipsis)	U: *How many legislatures represent lesser number of states than East Bengal Legislative Assembly?* U: *207* U: *And how about Estates of Curaçao?*
Count over More/Less | Single entity type (Coreference)	U: *Which french administrative division was Philippe Esnault born in?* S: *Alençon* U: *Which occupation has that person as his/her 's career?* S: *historian* U: *Which administrative territory is the native country of that person?* S: *France* U: *How many administrative territories inspired less number of fictional locations than that administrative territory?*

CountLogic

Seen

Union | Single Relation

Which people are the creators of The Theory of Everything or Ten Minutes to Live?

```
SELECT ?x WHERE {
    {wd:Q15079318 wdt:P162 ?x. ?x wdt:P31 wd:Q502895.}
    UNION
    {wd:Q7699260 wdt:P162 ?x. ?x wdt:P31 wd:Q502895.} }
```

Count | Single entity type

How many people starred in Captain America: Civil War?

```
SELECT (COUNT(*) AS ?count) WHERE {
wd:Q18407657 wdt:P161 ?x. ?x wdt:P31 wd:Q502895.}
```

Unseen

Count | Logical operators

How many national association football teams or national sports teams represent Slovenia?

```
  SELECT (COUNT (DISTINCT ?x) AS ?count) WHERE {
{?x wdt:P1532 wd:Q215. ?x wdt:P31 wd:Q6979593.}
UNION
{?x wdt:P1532 wd:Q215. ?x wdt:P31 wd:Q1194951.} }
```

Pred

```
SELECT (COUNT(DISTINCT ?x) AS ?count) WHERE {
    {?x wdt:P1532 wd:Q215. ?x wdt:P31 wd: Q6979593 .}
    UNION
    {?x wdt:P1532 wd:Q215. ?x wdt: P31 wd: Q6979593.} }
```

UnionMulti

Seen

Union | Single Relation

Which people are the creators of The Theory of Everything or Ten Minutes to Live?

```
SELECT ?x WHERE {
    ?x wdt:P915 wd:Q1247373. ?x wdt:P31 wd:Q838948.}
```

Count | Mult. entity type

How many people starred in Django Kill or Shatterday?

```
SELECT (COUNT(DISTINCT ?x) AS ?count) WHERE {
    {wd:Q1261875 wdt:P161 ?x. ?x wdt:P31 wd:Q502895.}
    UNION
    {wd:Q7490688 wdt:P161 ?x. ?x wdt:P31 wd:Q502895.} }
```

Unseen

Union | Multiple Relation

Which administrative territories are the origin of Les Chics Types or are the native countries of Robert Kuraś?

```
SELECT ?x WHERE {
    {wd:Q3231475 wdt:P495 ?x. ?x wdt:P31 wd:Q15617994.}
    UNION
    {wd:Q9310937 wdt:P27 ?x. ?x wdt:P31 wd:Q15617994.} }
```

Pred

```
SELECT ?x WHERE {
    {wd:Q3231475 wdt:P495 ?x. ?x wdt:P31 wd:Q15617994.}
    UNION
    {wd:Q9310937 wdt:P495 ?x. ?x wdt:P31 wd:Q15617994.} }
```

Verifiy3

Seen

2 entities, both direct

Is Zugspitze located in Germany?

```
ASK {wd:Q3375 wdt:P17 wd:Q183.}
```

Unseen

3 entities, all direct, 2 are query entities

Is Violet Oakley a civilian of United States of America and Scheden?

```
ASK {wd:Q30 wdt:P27 wd:Q1226556.
     wd:Q557427 wdt:P27 wd:Q1226556.}
```

Pred

```
ASK {wd:Q1226556 wdt:P27 wd:Q30.
     wd:Q557427 wdt:P27 wd:Q557427.}
```

Table 10: Generalisation splits, unseen question subtypes, support question subtypes seen during training, and example common errors on unseen predictions.

MAPL🍁: Parameter-Efficient Adaptation of Unimodal Pre-Trained Models for Vision-Language Few-Shot Prompting

Oscar Mañas[△] **Pau Rodriguez**[*,◇] **Saba Ahmadi**[*,△]
Aida Nematzadeh[♡] **Yash Goyal**[♣] **Aishwarya Agrawal**[△,†]
[△]Mila, Université de Montréal [◇]ServiceNow Research [♡]DeepMind
[♣]Samsung - SAIT AI Lab, Montreal †Canada CIFAR AI Chair
`oscar.manas@mila.quebec`

Abstract

Large pre-trained models have proved to be remarkable zero- and (prompt-based) few-shot learners in unimodal vision and language tasks. We propose MAPL, a simple and parameter-efficient method that reuses frozen pre-trained unimodal models and leverages their strong generalization capabilities in multimodal vision-language (VL) settings. MAPL learns a lightweight mapping between the representation spaces of unimodal models using aligned image-text data, and can generalize to unseen VL tasks from just a few in-context examples. The small number of trainable parameters makes MAPL effective at low-data and in-domain learning. Moreover, MAPL's modularity enables easy extension to other pre-trained models. Extensive experiments on several visual question answering and image captioning benchmarks show that MAPL achieves superior or competitive performance compared to similar methods while training orders of magnitude fewer parameters. MAPL can be trained in just a few hours using modest computational resources and public datasets. We release our code and pre-trained model weights at `https://github.com/oscmansan/mapl`.

1 Introduction

Over the past few years, natural language processing and computer vision have witnessed impressive progress in learning models capable of transferring to unseen tasks or benchmarks (Brown et al., 2020; Zhang et al., 2022; Radford et al., 2021; Jia et al., 2021). Recently referred to as foundation models (Bommasani et al., 2021), these can be adapted to a wide range of *unimodal* vision and language tasks without any additional training.

In this work, we study reusing such powerful *unimodal* foundation models for *multimodal* vision-language (VL) downstream tasks. In particular, we propose to connect a vision encoder, such as CLIP (Radford et al., 2021), to an autoregressive language model (LM), such as GPT (Radford et al., 2018, 2019; Brown et al., 2020), with minimal additional training on multimodal data. Our goal is to obtain a single VL model that can leverage the in-context learning abilities (Brown et al., 2020) of the pre-trained LM to generalize to unseen VL tasks from just a few examples.

One challenge in connecting vision encoders with LMs is aligning the visual and textual representation spaces. Recent works have approached this by adapting the LM to visual representations, either by fine-tuning the entire LM (Dai et al., 2022) or training adapter layers (Eichenberg et al., 2021; Alayrac et al., 2022). These systems are computationally expensive to train as they have a large number of learnable parameters (hundreds of millions to a few billions) and use large-scale multimodal training data. On the other hand, Frozen (Tsimpoukelli et al., 2021) keeps the LM frozen, thus learning $\sim$10× less parameters than the above methods. However, it requires training a visual encoder from scratch, which is also computationally expensive.

Differently, we aim to reuse large pre-trained unimodal models while keeping them completely frozen and free of adapter layers. We present **MAPL** (**M**ultimodal **A**daptation of **P**re-trained vision and **L**anguage models), a simple and parameter-efficient VL model capable of tackling unseen VL tasks. MAPL learns a lightweight mapping between the representation spaces of pre-trained unimodal models. MAPL has orders of magnitude fewer parameters than previous methods (including Frozen) and can be trained in just a few hours. Moreover, MAPL's modularity makes it general-purpose and easily extensible to newer and/or better pre-trained models. We evaluate MAPL on various image captioning and visual question answering (VQA) benchmarks and compare with Frozen (Tsimpoukelli et al., 2021) in a controlled setup. MAPL significantly outperforms

---

[*] denotes equal contribution.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2523–2548
May 2-6, 2023

Frozen and achieves competitive performance compared to other methods (Eichenberg et al., 2021; Dai et al., 2022) trained on comparably sized data.

We further investigate the parameter efficiency of MAPL by training on only 1% of multimodal data (thousands of examples); we call this setting *low-data* learning. We also study *in-domain* learning: training on image-text pairs from the same domain as the downstream task domains. We train MAPL directly on 100% and 1% of in-domain data for each downstream task, without first pre-training on large-scale domain-agnostic data. Thus, we train specialized versions of MAPL for each downstream domain. Such low-data and in-domain learning are particularly useful when it is difficult to pre-train on large-scale domain-agnostic data. We found MAPL to be more effective than Frozen trained under the same settings.

To summarize, our contributions are: 1) we introduce MAPL, a parameter-efficient method capable of tackling unseen VL tasks, which can be trained using only modest computational resources and public datasets; 2) we conduct extensive experiments spanning various image captioning and VQA benchmarks, demonstrating MAPL achieves superior or competitive performance compared to similar methods while training orders of magnitude fewer parameters; and 3) we further investigate the parameter-efficiency of MAPL in two settings: low-data and in-domain. Our experiments show that MAPL is more effective than the considered methods in both settings.

2 Related Work

Fine-tuning based VL methods. A popular family of VL methods are based on the pre-training + fine-tuning paradigm. These methods are either encoder-only (Lu et al., 2019; Tan and Bansal, 2019; Chen et al., 2019; Li et al., 2020; Zhang et al., 2021) or encoder-decoder methods (Cho et al., 2021; Wang et al., 2021; Jin et al., 2022; Li et al., 2021) and use transformer-based architectures. These transformers are first pre-trained on domain-agnostic image-text pairs (e.g., Conceptual Captions (Sharma et al., 2018)) using self-supervised objectives, and then fine-tuned for each downstream task (e.g., VQA, image captioning). More recent models that are designed specifically for the task of image captioning use large pre-trained LMs (e.g., GPT-2 (Radford et al., 2019)) and fine-tune these models with image-caption pairs (Chen et al., 2021; Mokady et al., 2021; Luo et al., 2022). While all these approaches yield state-of-the-art performance for the tasks they are fine-tuned on, the learned model weights are highly specialized for a single task and cannot transfer to new tasks with zero or few examples. Differently, MAPL reuses the same set of weights for all downstream tasks without any additional training.

Few-shot learning based VL methods. Most similar to MAPL are methods that tackle unseen VL tasks in a zero/few-shot manner, by leveraging the in-context learning abilities of large pre-trained LMs (e.g., GPT-3 (Brown et al., 2020). These methods connect a vision encoder with a pre-trained LM to tackle VL tasks. Some methods (Dai et al., 2022; Hao et al., 2022) achieve this connection by fine-tuning the entire LM on image-text data, while others only train adapter layers inserted into the LM (Eichenberg et al., 2021). The vision encoder is pre-trained and kept frozen in both cases. Concurrent work Flamingo (Alayrac et al., 2022) pushes this idea even further by scaling up the amount of training data and the LM size. While inserting adapter layers requires training fewer parameters compared to fine-tuning the entire LM, the number of trainable parameters is still >100M; in contrast, MAPL only has 3.4M trainable parameters. Additionally, inserting adapter layers is not straightforward since it requires modifying the computational graph of the LM; MAPL only adds an external mapping network, which is easier to incorporate on top of pre-trained models. On the other hand, Frozen (Tsimpoukelli et al., 2021) keeps the pre-trained LM frozen and instead trains a vision encoder from scratch. This approach does not scale well with larger vision encoders (Sec. 4.5). MAPL keeps both the vision encoder and the LM frozen (thus further reducing the number of trainable parameters) and only learns a lightweight mapping network to connect both frozen models. Similar to MAPL, concurrent work LiMBeR (Merullo et al., 2022) also proposes to connect a frozen vision encoder with a frozen LM but using a linear mapping, which is not as parameter- and compute-efficient as MAPL (Sec. 4.5).

Mapping networks. MAPL trains a mapping network to align the visual and textual representations of the visual encoder and the LM, respectively. The architecture of our mapping network has some similarities with that in ClipCap (Mokady et al., 2021) and the Perceiver Resampler in Flamingo (Alayrac

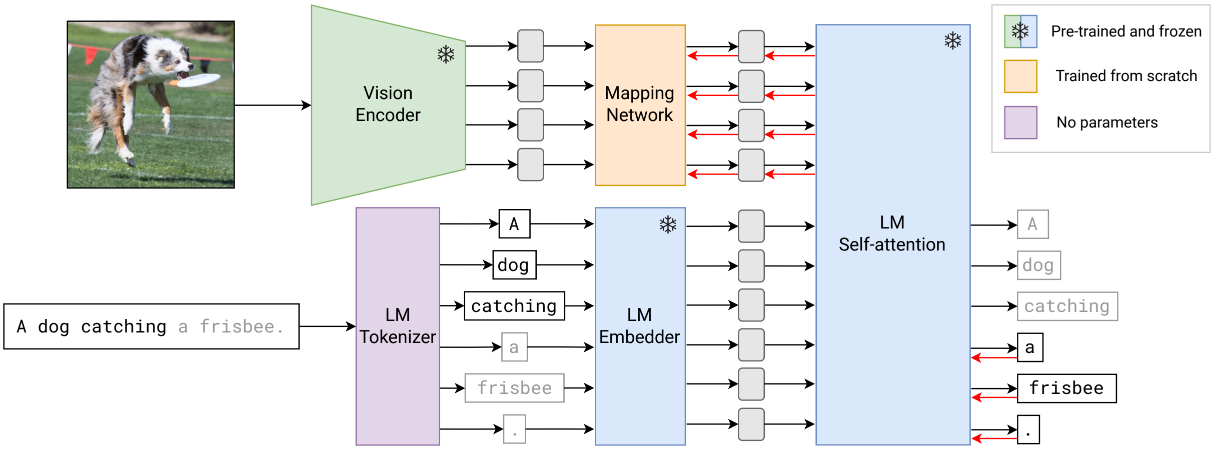


Figure 1: MAPL leverages a pre-trained vision encoder and a pre-trained LM, and learns a small *mapping network* to convert visual features into token embeddings. During training, only the mapping network is updated, keeping the vision encoder and the LM frozen (red arrows indicate gradient flow). At inference time, the system can take as input an arbitrary sequence of interleaved images and text, and generates free-form text as output.

et al., 2022). They all share a core transformer stack and a fixed number of learned constant embeddings. However, MAPL's mapping network is specifically designed to be parameter-efficient while maintaining expressivity (Sec. 3.1), containing only 3.4M parameters – orders of magnitude fewer than ClipCap's (43M) and Flamingo's (194M).

3 Method

MAPL is a vision-language (VL) multimodal model capable of generating text from a combination of visual and textual inputs. Our model builds on top of pre-trained vision-only and language-only models and leverages their strong generalization capabilities (e.g., zero-shot transfer, in-context learning) to tackle unseen VL tasks. MAPL is agnostic to the choice of these pre-trained unimodal models as long as they show such capabilities (Sec. 4.5). Concretely, MAPL maps the image representations from a vision encoder's output embedding space to a LM's token embedding space, so that the LM can be conditioned both on visual and textual information. To this end, we train a *mapping network* with an image captioning objective (Sec. 3.1, 3.2), while keeping the weights of the vision encoder and the LM frozen. Once the mapping network is trained, MAPL can be prompted with a few examples of unseen VL tasks and predict the response via text generation (Sec. 3.3). The overall model architecture is depicted in Figure 1.

3.1 Architecture

Pre-trained vision encoder. The vision encoder extracts a compact representation from an image. We use a CLIP (Radford et al., 2021) pre-trained vision encoder, which is trained on web-scale data and has shown strong zero-shot transfer capabilities to unseen image domains. In particular, we use CLIP's ViT-L/14 backbone (Dosovitskiy et al., 2020) since we empirically found it yields the best downstream VL performance among all variants. We use the flattened grid of spatial features (16×16) before the final projection layer and the representation corresponding to the `[class]` token, resulting in a sequence of $L_i = 257$ vectors of dimensionality $D_i = 1024$ each. This sequence of vectors is then fed to the mapping network.

Pre-trained autoregressive language model. Given an input text, the language model (LM) predicts its most likely completion by generating free-form text. For our LM, we use a pre-trained GPT-J model (Wang and Komatsuzaki, 2021) [1], a publicly-released 6B-parameter autoregressive LM trained on the Pile dataset (Gao et al., 2020). We chose this LM due to its strong in-context learning abilities, similar to that of GPT-3 (Brown et al., 2020) (which is not publicly available). The LM takes as input a text string, which is first divided into a sequence of discrete tokens by the LM's tokenizer. Each token is then individually transformed into a continuous embedding (of size $D_o = 4096$) by the LM's embedder. The sequence of token embeddings is fed to the self-attention layers in the LM's transformer block (using causal attention), which outputs a sequence of categorical distributions over the token vocabulary. Finally, a decoding mechanism generates free-form text from these distributions (greedy

---

[1] We also experiment with an OPT model, see Sec. 4.5.

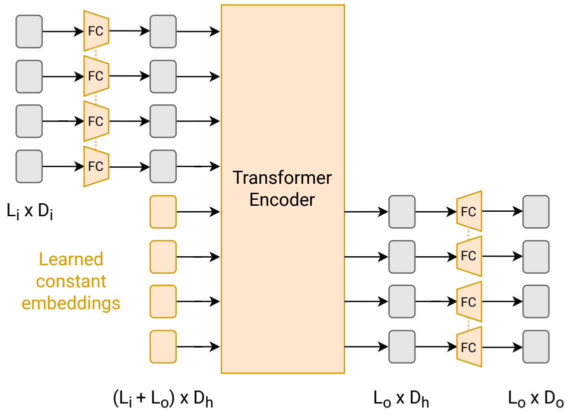


Figure 2: The mapping network takes a flattened grid of L_i visual features of dimension D_i each from the vision encoder and transforms it into a sequence of token embeddings of length L_o and dimension D_o, where D_o is the token embedding dimension of the LM. Note that the parameters are shared across fully-connected (FC) layers, on both sides of the encoder transformer.

decoding in our case).

Mapping network. The mapping network transforms a sequence of visual features from the vision encoder to a sequence of continuous embeddings which can be consumed by the LM's transformer. We design our mapping network considering the trade-off between expressivity (to learn a good mapping) and parameter count. Our architecture is based on a transformer encoder with 4 layers and 8 heads each. This transformer could directly take a sequence of projected visual features (from D_i to D_o) and output a sequence of embeddings of size D_o. However, in order to keep a low parameter count, we decouple the transformer hidden size D_h from the visual feature size D_i and the LM embedding size D_o by introducing a dimensionality bottleneck (Figure 2). In particular, each visual feature is first linearly projected from $D_i = 1024$ to $D_h = 256$ using a set of fully-connected (FC) layers. This sequence of projected features is then fed to the transformer, and the output representations are linearly projected from D_h to $D_o = 4096$ using another set of FC layers. To further reduce the parameter count of our mapping network, we share parameters across all FC layers in each set.

Yet another idea we use in our mapping network is to decouple the output sequence length of the transformer (L_o) from the input sequence length (L_i). We do this to obtain a much smaller $L_o = 32$ compared to $L_i = 257$, in order to reduce the computational complexity in the subsequent LM's self-attention layers, which in turn speeds up training and inference time. To achieve this decoupling, inspired by DETR (Carion et al., 2020), we concatenate a small and fixed number (L_o) of learned constant embeddings with the input sequence of the transformer and only use the output representations corresponding to these constant embeddings (Figure 2). Note that these output representations are conditioned on the input visual features via cross-attention in the transformer. The resulting mapping network architecture is shown in Figure 2. In total, our mapping network contains only 3.4M parameters. Since this is the only trainable component of our model, MAPL has orders of magnitude fewer total trainable parameters than existing methods such as Frozen (40.3M) or Flamingo (10.2B).

3.2 Training

Following previous works (Tsimpoukelli et al., 2021; Eichenberg et al., 2021), we train our model using a standard language modeling objective on image captions with teacher forcing (Lamb et al., 2016), i.e., we minimize the negative log-likelihood of the reference captions under the LM conditioned on the corresponding images. We only train the mapping network (from scratch) while keeping the vision encoder and the LM entirely frozen. This preserves the pre-trained models' capabilities while making the system modular and parameter-efficient. Even though the LM's weights are kept frozen, gradients are still back-propagated through its self-attention layers to train the mapping network.

3.3 Zero- and Few-shot Evaluation

Once the mapping network is trained, MAPL can tackle unseen VL tasks by prompting the LM with a combination of visual and textual inputs. We study zero-shot transfer to unseen image captioning benchmarks and few-shot transfer (via in-context learning) to the unseen task of visual question answering (VQA). For image captioning, we simply feed the mapped image embedding to the LM and start generating a caption. For zero-shot VQA, following Tsimpoukelli et al. (2021), we feed the mapped image embedding followed by the text `“Please answer the question. Question: {question} Answer:”`[2] and start generating the answer. For n-shot VQA, we select n support examples $(image, question, answer)$ from the training set at random, and prepend them to the

---

[2] Here `{question}` indicates a placeholder which gets replaced by the corresponding question in each example. Same applies to `{answer}` in the few-shot setting.

query; for each support example, we concatenate the mapped image embedding with the text "`Please answer the question. Question: {question} Answer: {answer}`".

4 Experiments

4.1 Experimental settings

Evaluation benchmarks. We evaluate MAPL on several VL benchmarks spanning VQA and image captioning. Note that our model is never trained for the task of VQA. For VQA, we evaluate on the validation splits of VQAv2 (Goyal et al., 2017), OK-VQA (Marino et al., 2019), TextVQA (Singh et al., 2019) and VizWiz-VQA (Gurari et al., 2018), and report performance using VQA accuracy (after the standard normalization (Antol et al., 2015)). For image captioning, we evaluate on the Karpathy-test split (Karpathy and Fei-Fei, 2015) of COCO Captions (Chen et al., 2015), and the validation splits of Conceptual Captions (CC) (Sharma et al., 2018)[3], TextCaps (Sidorov et al., 2020) and VizWiz-Captions (Gurari et al., 2018), and report performance using the BLEU@4, ROUGE-L, METEOR, CIDEr and SPICE metrics.

Training settings. We consider two settings to train our mapping network: domain-agnostic and in-domain training (described below). For each of these settings, we also study low-data learning by training our model on randomly sampled subsets of 1% training image-text pairs. Such low-data learning is useful when it is difficult to train models on large-scale data due to constraints on compute resources, data availability, etc.

For **domain-agnostic training**, we use the CC dataset, which is gathered by automatically scraping images and their corresponding alt-text fields from web pages. Thus, this dataset is not as clean as manually-curated datasets such as COCO Captions (e.g., the caption may not describe the image). Nevertheless, due to its large size (3.3M) and great diversity, it is the most commonly used dataset for domain-agnostic pre-training of VL models. However, for our model – having orders of magnitude less trainable parameters than other methods –, we observed the negative effect of noise in CC to be stronger than the positive effect of its large size (Sec. 4.4). Therefore, we train MAPL on a filtered version of CC (CC-clean) consisting of the top 398K most similar image-text pairs ranked by CLIP's image-text similarity score.[4] For completeness, we also report MAPL's performance when trained on the unfiltered CC dataset.

For **in-domain training**, we use image-caption pairs that come from the same domain as the downstream task domains, i.e., they have similar image and language distributions as those in the downstream datasets. For the image captioning downstream task, this amounts to the IID setting. The in-domain image captioning and VQA dataset pairs we consider are shown in Table 1. Each pair uses the same set of images, and focuses on the same set of VL skills; for instance, scene understanding (COCO Caps and VQAv2), reading and reasoning about text in images (TextCaps and TextVQA), understanding images captured by visually-impaired users (VizWiz-Caps and VizWiz-VQA), thus leading to similar image and language distributions across image-captioning and VQA. We train MAPL on both 100% and 1% of in-domain image-caption data and evaluate on all downstream benchmarks (including out-of-domain ones, e.g., VizWiz-VQA when trained on COCO Caps). Such in-domain training can be useful when it is difficult to first train on large-scale domain-agnostic data and then adapt to in-domain data by either fine-tuning or few-shot prompting.

	VQAv2	OK-VQA	TextVQA	VizWiz-VQA
COCO Caps	✓	✓		
TextCaps			✓	
VizWiz-Caps				✓

Table 1: In-domain dataset pairs.

Training details. For Conceptual Captions, TextCaps and VizWiz-Captions, we carve out a minival split consisting of 6% of training examples and train on the remaining 94%; for COCO Captions, we use the Karpathy-val split as minival. We use the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and a weight decay of 0.01. The learning rate is increased linearly from 0 to 3×10^{-4} (7×10^{-4} for OPT-based models) over the first 1500 steps (15 for 1% of data) and kept constant for the rest of training. We use a batch size of 128 and we do early stopping based on the minival loss. We do not add any special tokens at the beginning of sentence, as GPT-J was not trained with `<BOS>` tokens. In order to fit a 6B-parameter LM into GPU memory, we use DeepSpeed ZeRO (Rajbhan-

---

[3] Due to broken image URLs, we only managed to download 13K out of 15K validation images.

[4] We selected a threshold on CLIP's similarity score such that the size of the filtered dataset is comparable to the size of manually curated datasets such as COCO Captions.

	Trainable params	**Training examples**	**n-shot VQAv2**			**n-shot OK-VQA**			**n-shot TextVQA**			**n-shot VizWiz-VQA**			**n-shot Overall**		
			0	**4**	**8**	**0**	**4**	**8**	**0**	**4**	**8**	**0**	**4**	**8**	**0**	**4**	**8**
Existing methods using domain-agnostic training																	
Frozen	40.3M†	3.3M	29.50	38.20	-	5.90	12.60	-	-	-	-	-	-	-	-	-	-
MAGMA $_{\text{CC12M}}$	243M†	3.8M	36.90	45.40	-	13.90	23.40	-	-	-	-	5.60	10.60	-	-	-	-
VLKD $_{\text{CC3M}}$	406M	3.3M	38.60	-	-	10.50	-	-	-	-	-	-	-	-	-	-	-
LiMBeR-CLIP‡	12.6M†	3.3M	33.33	40.34	-	-	-	-	-	-	-	-	-	-	-	-	-
Flamingo‡	10.2B	>2.1B	-	-	-	50.60	57.40	57.50	35.00	36.50	37.30	-	-	-	-	-	-
100% domain-agnostic training																	
MAPL-blind $_{\text{CC-clean}}$	3.4M	374K	20.62	35.01	35.11	4.84	14.68	14.28	3.68	5.43	5.82	3.18	8.65	9.55	8.08	15.94	16.19
Frozen* $_{\text{CC-clean}}$	40.3M	374K	25.98	37.80	38.52	5.51	18.86	19.91	5.11	6.15	6.30	4.33	11.28	16.68	10.23	18.52	20.35
MAPL $_{\text{CC-clean}}$	3.4M	374K	**33.54**	**45.13**	**45.21**	**13.84**	**24.25**	**23.93**	**8.26**	**8.88**	**8.77**	**11.72**	**18.46**	**19.52**	**16.84**	**24.18**	**24.36**
1% domain-agnostic training																	
Frozen* $_{\text{CC-clean}}$	40.3M	3.7K	26.22	36.69	37.41	5.50	**18.76**	**20.51**	5.71	**7.19**	7.53	3.83	**11.71**	**16.66**	10.31	**18.58**	**20.53**
MAPL $_{\text{CC-clean}}$	3.4M	3.7K	**30.80**	**37.38**	**37.95**	**8.77**	18.18	19.15	**6.40**	7.07	**7.74**	**5.68**	9.26	10.58	**12.91**	17.97	18.85
100% in-domain training																	
PICa*	0	0	20.61	46.86	47.80	11.84	**31.28**	**33.07**	-	-	-	-	-	-	-	-	-
Frozen* $_{\text{COCO}}$	40.3M	414K	32.09	38.90	39.42	9.81	20.72	21.83	7.54	6.82	6.74	5.87	12.07	17.35	13.82	19.63	21.33
Frozen* $_{\text{TextCaps}}$	40.3M	103K	32.49	37.39	38.03	11.34	19.87	20.82	8.83	7.33	7.51	6.25	12.26	16.86	14.73	19.21	20.80
Frozen* $_{\text{VizWiz}}$	40.3M	110K	26.93	37.38	37.91	5.85	19.12	20.64	6.38	7.44	7.47	5.57	13.06	18.06	11.18	19.25	21.02
MAPL $_{\text{COCO}}$	3.4M	414K	**43.51**	**48.75**	**48.44**	**18.27**	31.13	31.63	10.99	11.10	11.08	**14.05**	17.72	19.18	21.70	**27.17**	**27.58**
MAPL $_{\text{TextCaps}}$	3.4M	103K	38.83	43.34	43.43	16.33	25.07	25.92	**22.27**	**19.53**	**19.75**	12.31	16.69	18.18	**22.43**	26.15	26.82
MAPL $_{\text{VizWiz}}$	3.4M	110K	32.80	42.94	43.20	11.70	24.91	25.73	9.27	10.36	10.23	10.42	**20.63**	**23.10**	16.05	24.71	25.56
1% in-domain training																	
Frozen* $_{\text{COCO}}$	40.3M	4.1K	30.18	37.23	37.89	9.33	19.60	20.71	7.43	7.65	7.67	4.37	12.00	16.48	12.83	19.12	20.69
Frozen* $_{\text{TextCaps}}$	40.3M	1.0K	32.09	36.72	37.25	10.75	18.85	19.51	8.17	7.57	7.28	5.39	11.79	16.20	14.10	18.73	20.06
Frozen* $_{\text{VizWiz}}$	40.3M	1.1K	29.62	37.30	37.87	7.57	19.36	20.60	7.16	7.17	7.25	4.53	**12.51**	**17.56**	12.22	19.08	**20.82**
MAPL $_{\text{COCO}}$	3.4M	4.1K	**37.69**	**40.42**	**40.84**	**13.92**	**21.66**	**22.41**	8.30	6.96	6.84	**6.94**	10.72	12.43	**16.71**	**19.94**	20.63
MAPL $_{\text{TextCaps}}$	3.4M	1.0K	33.57	36.70	36.87	12.46	17.45	18.21	**9.34**	**8.29**	**8.62**	6.54	9.58	11.62	15.48	18.00	18.83
MAPL $_{\text{VizWiz}}$	3.4M	1.1K	31.88	36.81	37.04	9.59	17.64	17.64	7.25	5.99	6.04	4.73	9.48	11.33	13.36	17.48	18.01

Table 2: Evaluation on few-shot VQA. For MAGMA $_{\text{CC12M}}$ and VLKD $_{\text{CC3M}}$, we report their best results when training only on domain-agnostic data (CC12M and CC3M, respectively). (†) indicates our informed estimation. (‡) indicates concurrent work.

dari et al., 2020) stage 2 optimizations. Freezing the LM's weights also brings massive savings in GPU memory during training, as fine-tuning with an Adam-based optimizer would require at least 4× GPU memory to store gradients, average, and squared average of the gradients. The whole system was trained on 4 A100 (40GB) GPUs for about 4 hours (for the CC-clean dataset). Unless otherwise stated, we repeat the experiments with two different random seeds and report the average performance.

Existing methods and baselines. We report the performance of several baselines and existing methods. First, to verify that the LM in MAPL is not ignoring the visual input, inspired by Tsimpoukelli et al. (2021), we train a blind version of MAPL (MAPL-blind) where the input images are replaced with zeros but the mapping network weights are still trained (to serve as prompt-tuning for the LM). Second, to estimate the upper-bound on how well we can do in VQA by representing images with text (rather than with continuous embeddings), we evaluate PICa (Yang et al., 2021), which directly prompts the LM with image captions, followed by questions for VQA. We reimplement PICa (denoted PICa*) using MAPL's LM (and evaluate on VQAv2 and OK-VQA using ground-truth COCO captions) for controlled comparison. Third, we compare MAPL with Frozen (Tsimpoukelli et al., 2021), as this is the most similar method to ours that also uses a frozen LM. We reimplement Frozen (denoted Frozen*) using MAPL's LM for controlled comparison. Lastly, we report the performance of other methods similar to MAPL: MAGMA (Eichenberg et al., 2021), VLKD (Dai et al., 2022), LiMBeR (Merullo et al., 2022), ClipCap (Mokady et al., 2021) and the published numbers from Frozen (Tsimpoukelli et al., 2021).[5] Note that all these methods (unless otherwise noted) are trained on domain-agnostic data, so we only compare with MAPL trained on CC-clean. For completeness, we also report results from Flamingo (Alayrac et al., 2022), which has orders of magnitude more learnable parameters than MAPL and is trained on considerably more data.

4.2 Evaluation of domain-agnostic learning

We report few-shot VQA results in Table 2 and image captioning results in Table 3. Subscripts in the first column denote the training dataset. *Overall* accuracies denote average of per-benchmark accuracies. First, we see that MAPL $_{\text{CC-clean}}$ substantially outperforms MAPL-blind $_{\text{CC-clean}}$ both on VQA and image captioning, proving that the visual inputs are not ignored by the LM in MAPL. Second, we find that MAPL $_{\text{CC-clean}}$ outperforms Frozen* $_{\text{CC-clean}}$ by a considerable margin on all VL benchmarks (with overall accuracy improvements of +6.61% 0-shot and +5.66% 4-shot on VQA tasks,

---

[5] We only add results which are reported on the same dataset splits as in MAPL.

	Trainable params	Training examples	CC B@4	CC CIDEr	COCO B@4	COCO CIDEr	TextCaps B@4	TextCaps CIDEr	VizWiz-Caps B@4	VizWiz-Caps CIDEr	Overall B@4	Overall CIDEr
Existing methods using domain-agnostic training												
ClipCap $_{\text{CC3M}}$	43M	3.3M	-	71.82	-	-	-	-	-	-	-	-
VLKD $_{\text{CC3M}}$	406M	3.3M	-	-	18.20	61.10	-	-	-	-	-	-
100% domain-agnostic training												
MAPL-blind $_{\text{CC-clean}}$	3.4M	374K	0.35	5.05	2.75	5.75	1.35	2.15	1.50	1.80	1.49	3.69
Frozen* $_{\text{CC-clean}}$	40.3M	374K	2.45	22.60	5.25	13.90	2.65	4.60	2.05	2.65	3.10	10.94
MAPL $_{\text{CC-clean}}$	3.4M	374K	**6.75**	**79.75**	**12.30**	**54.30**	**5.80**	**22.95**	**4.95**	**20.95**	**7.45**	**44.49**
1% domain-agnostic training												
Frozen* $_{\text{CC-clean}}$	40.3M	3.7K	0.75	6.55	3.05	5.25	1.70	1.65	1.50	1.40	1.75	3.71
MAPL $_{\text{CC-clean}}$	3.4M	3.7K	**1.75**	**19.65**	**5.80**	**17.85**	**2.70**	**5.40**	**2.15**	**4.85**	**3.10**	**11.94**
100% in-domain training												
Frozen* $_{\text{COCO}}$	40.3M	414K	0.65	9.05	20.05	61.35	6.95	11.75	5.45	6.20	8.28	22.09
Frozen* $_{\text{TextCaps}}$	40.3M	103K	0.20	3.55	4.05	6.70	8.85	16.95	4.40	5.25	4.38	8.11
Frozen* $_{\text{VizWiz}}$	40.3M	110K	0.25	4.40	3.75	6.05	4.10	5.65	19.00	76.85	6.78	23.24
ClipCap $_{\text{COCO}}$	43M	414K	-	-	33.53	113.08	-	-	-	-	-	-
MAPL $_{\text{COCO}}$	3.4M	414K	**2.25**	**34.50**	**36.45**	**125.20**	16.60	41.40	18.00	41.35	**18.33**	**60.61**
MAPL $_{\text{TextCaps}}$	3.4M	103K	0.90	13.05	9.80	28.65	**18.35**	**62.55**	11.20	31.85	10.06	34.03
MAPL $_{\text{VizWiz}}$	3.4M	110K	0.90	18.80	13.55	48.35	11.35	31.20	**34.70**	**141.30**	15.13	59.91
1% in-domain training												
Frozen* $_{\text{COCO}}$	40.3M	4.1K	0.25	3.60	6.20	12.80	2.80	3.15	2.85	2.30	3.03	5.46
Frozen* $_{\text{TextCaps}}$	40.3M	1.0K	0.10	2.60	1.65	2.80	3.65	5.00	2.00	2.25	1.85	3.16
Frozen* $_{\text{VizWiz}}$	40.3M	1.1K	0.20	3.40	2.90	3.20	3.35	3.45	12.70	40.55	4.79	12.65
MAPL $_{\text{COCO}}$	3.4M	4.1K	**0.80**	**12.10**	**19.65**	**65.90**	7.00	12.85	6.20	9.60	**8.41**	**25.11**
MAPL $_{\text{TextCaps}}$	3.4M	1.0K	0.30	3.90	4.10	8.05	**8.35**	**16.90**	5.00	7.25	4.44	9.03
MAPL $_{\text{VizWiz}}$	3.4M	1.1K	0.20	3.90	2.95	4.80	3.45	5.05	**18.40**	**71.10**	6.25	21.21

Table 3: Evaluation on image captioning. For VLKD $_{\text{CC3M}}$, we report their best results when training only on domain-agnostic data (CC3M).

	Training examples	VQAv2 4-shot	OK-VQA 4-shot	TextVQA 4-shot	VizWiz-VQA 4-shot	CC CIDEr	COCO CIDEr	TextCaps CIDEr	VizWiz-Caps CIDEr	Overall 4-shot	Overall CIDEr
Frozen* $_{\text{CC-clean}}$	0.4M	37.79	**19.29**	**6.25**	**11.11**	22.70	14.00	5.00	2.70	**18.61**	11.10
Frozen* $_{\text{CC-cleanish}}$	1.0M	**37.82**	18.49	6.12	10.16	37.60	20.60	6.60	3.20	18.15	17.00
Frozen* $_{\text{CC}}$	2.7M	37.81	18.33	5.56	9.97	**57.60**	**22.20**	**8.00**	**4.20**	17.92	**23.00**
MAPL $_{\text{CC-clean}}$	0.4M	44.35	24.03	**9.65**	17.33	72.70	**54.60**	**23.80**	**21.10**	23.84	43.05
MAPL $_{\text{CC-cleanish}}$	1.0M	**46.63**	**25.99**	8.48	**19.65**	88.30	54.10	22.30	19.80	**25.19**	**46.13**
MAPL $_{\text{CC}}$	2.7M	43.26	20.96	5.20	19.31	**101.10**	44.10	16.70	15.90	22.18	44.45

Table 4: Impact of data quality and size. These experiments are run with one seed only.

+4.35 BLEU@4 and +33.55 CIDEr on image captioning tasks). Importantly, this is achieved while training an order of magnitude fewer parameters (3.4M vs 40.3M). Next, MAPL $_{\text{CC-clean}}$ is competitive compared to existing methods (MAGMA, VLKD, ClipCap) and concurrent work LiMBeR, despite training one-two orders of magnitude fewer parameters on significantly less multimodal data. Lastly, MAPL $_{\text{CC-clean}}$'s performance is still far from the performance of Flamingo, which trains orders of magnitude more parameters on orders of magnitude more data. However, we believe MAPL to be an effective method for scenarios with constrained computational resources. For MAPL's qualitative results, see App. 4.6.

Low-data learning. When trained on only 1% domain-agnostic data, MAPL $_{\text{CC-clean}}$ outperforms Frozen* $_{\text{CC-clean}}$ for all image captioning evaluations (by +1.35 BLEU@4 and +8.23 CIDEr, overall) and all 0-shot VQA evaluations (by +2.60% overall accuracy), while achieving competitive performance on 4- and 8-shot VQA evaluations. In summary, these results show the effectiveness of our method in low-data settings, highlighting its usefulness for applications where data is scarce.

4.3 Evaluation of in-domain learning

In Tables 2 and 3, we observe that both MAPL and Frozen* benefit from directly training on in-domain data, compared to few-shot transfer from large-scale domain-agnostic pretraining. For instance, MAPL $_{\text{COCO}}$ and Frozen* $_{\text{COCO}}$ respectively outperform MAPL $_{\text{CC-clean}}$ and Frozen* $_{\text{CC-clean}}$ on VQAv2, OK-VQA and COCO Captions when trained on 100% of data. Interestingly, this performance gap is larger for MAPL compared to Frozen* by +2% 0-shot accuracy and +3.77% 4-shot accuracy averaged across VQAv2 and OK-VQA, and +9.35 BLEU@4 and +23.45 CIDEr on COCO Captions. A similar trend can be observed for TextCaps and TextVQA. Surprisingly, for 0-shot VQA and image captioning, training

	Vision encoder	Language model	Mapping network	VQAv2		OK-VQA		TextVQA		VizWiz-VQA		CC	COCO	TextCaps	VizWiz-Caps	Overall		
				0-shot	4-shot	0-shot	4-shot	0-shot	4-shot	0-shot	4-shot	CIDEr	CIDEr	CIDEr	CIDEr	0-shot	4-shot	CIDEr
Frozen*	NF-ResNet-50	GPT-J	Transformer	27.98	36.66	5.88	18.44	4.56	7.87	3.67	10.32	21.79	14.87	5.42	3.03	10.52	18.32	11.28
Frozen*	ViT-L/16	GPT-J	Transformer	27.82	36.60	5.64	16.26	3.67	5.27	4.70	11.77	13.19	8.77	2.88	2.35	10.46	17.48	6.80
Frozen*	ViT-L/14	GPT-J	Transformer	24.45	36.03	4.14	16.27	3.91	5.33	3.27	10.09	13.89	8.27	2.66	2.50	8.94	16.93	6.83
MAPL	CLIP-ViT-L/14	GPT-J	Transformer	**33.54**	**45.13**	13.84	24.25	8.26	**8.88**	11.72	**18.46**	**79.75**	54.30	22.95	**20.95**	**16.84**	**24.18**	**44.49**
MAPL	IN-NF-ResNet-50	GPT-J	Transformer	28.12	40.86	10.86	21.66	6.15	7.01	6.40	14.22	39.64	32.99	12.41	9.55	12.88	20.94	23.65
MAPL	IN-ViT-L/16	GPT-J	Transformer	31.70	43.75	11.13	**25.50**	6.16	7.40	8.93	16.45	56.33	45.80	17.12	16.28	14.48	23.28	33.88
Frozen*	NF-ResNet-50	OPT-6.7B	Transformer	30.16	32.72	8.10	13.79	5.44	6.81	7.15	6.77	25.40	17.80	6.90	4.00	12.71	15.02	13.53
MAPL	CLIP-ViT-L/14	OPT-6.7B	Transformer	23.26	33.95	**15.27**	16.25	**8.90**	6.41	**15.40**	9.47	66.60	**54.40**	**23.30**	19.60	15.71	16.52	40.98
MAPL	CLIP-ViT-L/14	GPT-J	Linear	30.55	37.09	12.20	16.69	7.02	5.80	8.81	12.49	60.00	43.80	18.30	13.70	14.65	18.02	33.95
MAPL	CLIP-ViT-L/14	GPT-J	MLP	28.99	43.69	11.07	25.33	6.60	8.39	9.73	17.14	70.40	49.10	20.90	20.30	14.10	23.64	40.18

Table 5: Ablation studies. We assess the impact of the choice of vision encoder (top), LM (middle) and mapping network architecture (bottom). All models are trained on 100% of CC-clean with a single seed. IN stands for ImageNet pre-training.

on just 1% of in-domain data outperforms 100% CC training for all benchmarks (except VizWiz-VQA) and both models. These results demonstrate the benefits of in-domain learning. When comparing MAPL vs. Frozen*, we observe that MAPL outperforms Frozen* for all tasks and benchmarks (except VizWiz-VQA) under both 100% and 1% in-domain settings. In fact, MAPL trained on just 1% in-domain data outperforms Frozen* trained on 100% in-domain data by +3.41% 0-shot accuracy and +1.14% 4-shot accuracy averaged across VQAv2, OK-VQA and TextVQA. Thus, MAPL is more effective than Frozen* at in-domain learning.

Contrary to the above trends, we observe that MAPL $_{\text{VizWiz}}$ under 1% in-domain training performs worse than MAPL $_{\text{COCO}}$ or MAPL $_{\text{TextCaps}}$ when evaluated on VizWiz-VQA. We hypothesize the visual embeddings extracted from CLIP's vision encoder for VizWiz images are not as good as those for COCO or TextCaps' images because the distribution of images in VizWiz (captured by visually-impaired people) is rather different from the distribution of images CLIP is trained on (scraped from the web), whereas for COCO and TextCaps this isn't the case. When training MAPL's mapping network on only 1% of VizWiz data, we believe the data is not large enough to compensate for the OOD pretrained vision encoder, so MAPL trained on COCO/TextCaps performs better on VizWiz-VQA. For in-domain training with 100% of data and 4/8-shot VQA, the mapping network has enough data to learn from and compensate for the OOD phenomenon. On the other hand, Frozen* does not suffer from this issue because its vision encoder is trained from scratch, allowing it to adapt to the image distribution.

Lastly, we observe that MAPL $_{\text{COCO}}$ outperforms ClipCap $_{\text{COCO}}$ by +2.92 BLEU@4 and +12.12 CIDEr on COCO Captions. MAPL $_{\text{COCO}}$ also outperforms PICa* (which represents images with ground-truth COCO captions) on VQAv2 and 0-shot OK-VQA, and achieves competitive results on few-shot OK-VQA; this demonstrates representing images with continuous embeddings is beneficial over caption-based image representations. Overall, we see that in-domain learning is beneficial and MAPL is more effective at it than similar methods.

4.4 Impact of data quality and size

To measure the impact of noise in the training data, we additionally train MAPL and Frozen* on the *full* CC dataset, consisting of 2.8M[6] examples, as well as on a *clean-ish* version consisting of the 1.0M most similar image-text pairs. In Table 4, we observe Frozen* achieves similar performance on few-shot VQA tasks when trained on noisy vs. clean data; however, Frozen*'s performance on image captioning decreases when trained on cleaner but smaller data. In contrast, MAPL generally benefits from cleaner training data, with the exception of evaluation on CC. We hypothesize both models perform better on CC when trained on larger (yet noisier) data because the CC validation set is IID with the full (noisy) CC training set. In the case of Frozen*, as we move away from the IID setting, the benefits from more data start diminishing (CC captioning > other captioning tasks > VQA tasks). For MAPL, the benefit from reduced noise in training data exceeds the degradation caused by a smaller data size, thanks to the reduced number of trainable parameters. These trends align with previous observations that larger models are more robust to noisy training data since they have enough capacity to model both noise and the desired function (Rolnick et al., 2017), while smaller models are more sample-efficient (Vapnik and Chervonenkis, 2015), i.e. they need less (clean) data to train effectively. Note that although MAPL's overall performance is higher when training on 1.0M than on 0.4M examples, we decided to train with 0.4M examples because training on $\sim$2.5× more data (1.0M instead of 0.4M) required $\sim$5× more iterations (always

---

[6] This is not the full 3.3M CC due to broken URLs.

early-stopping based on validation loss). So we did not think the slight performance increase due to more data was worth the $\sim 5\times$ longer training time, especially because we were operating under a limited compute budget.

4.5 Ablation studies

In this section, we evaluate how the choice of vision encoder, LM and mapping network architecture impact MAPL's performance, and compare it with corresponding versions of Frozen* (where applicable). Results are presented in Table 5. Please refer to App. A.5 for more ablations.

First, to assess the impact of the choice of vision encoder, we train additional versions of MAPL replacing the CLIP pre-trained vision encoder (ViT-L/14 – 303M parameters) with encoders pre-trained on ImageNet: NF-ResNet-50 (23.5M) and ViT-L/16 (303M), and compare their performance with corresponding versions of Frozen*. We observe that: 1) MAPL outperforms Frozen* for each configuration of vision encoder, suggesting that MAPL is **robust to the choice of vision encoder's pre-training data and architecture**; and 2) Frozen* 's performance drops with bigger vision encoders (likely due to more trainable parameters), whereas MAPL improves due to the use of stronger pre-trained encoders. Thus, training the vision encoder from scratch (Frozen*) has limited application, while MAPL's **performance scales alongside the pre-trained vision encoder**.

Next, to evaluate the impact of the choice of LM, we train both MAPL and Frozen* replacing GPT-J by OPT-6.7B (Zhang et al., 2022). We see that in all settings except 0-shot VQAv2, MAPL outperforms Frozen*. See App. A.2 for discussion on 0-shot VQAv2 results. This suggests that MAPL is **robust to the choice of LM**. The above results also highlight how MAPL's **modularity** allows to easily replace the pre-trained vision encoder or the LM.

Lastly, to assess the impact of the choice of mapping network architecture, we replace the proposed transformer-based mapping network with two simpler architectures – a linear layer and a 2-layer MLP (see App. A.4 for details). We observe both these versions generally underperform the original setting (transformer-based), highlighting the **effectiveness of the proposed design**. We also note that in these simpler versions, the parameter count is directly proportional to the vision encoder's representation size and LM's embedding size, whereas in MAPL we decouple this using a dimensionality bottleneck (Sec. 3.1), making our mapping network **more parameter-efficient by design**.

4.6 Qualitative results

Figure 3 shows some selected samples from the web illustrating our interface at inference time using MAPL $_{\text{CC-clean}}$. The first two columns show successful results while the last column shows failure cases. For image captioning (top row), success cases show MAPL can generate meaningful and detailed textual descriptions of the scene. For zero-shot VQA (bottom row), success cases indicate that MAPL is able to parse the question and connect visual information to encyclopedic knowledge contained in the pre-trained LM. However, MAPL's visio-linguistic understanding is evidently still far from being perfect. More qualitative results (both success and failure cases) are provided in App. A.6.

a man watches the sea birds as they fly over the beach.

a rail crossing with a sign warning of trains.

a boy playing soccer in the field.

What kind of leaf is this?

A maple leaf.

What does this animal eat?

Squirrels eat nuts, seeds, berries, and insects.

What type of cheese is on these vegetables?

broccoli.

Figure 3: Qualitative samples from the web using MAPL $_{\text{CC-clean}}$. (Multimodal) input is in gray, and MAPL's output is in green (success) or red (failure).

5 Conclusion

We introduce MAPL, a simple and parameter-efficient method to repurpose pre-trained and frozen unimodal models for multimodal tasks. Our experiments demonstrate that MAPL achieves superior or competitive performance compared to similar methods on several VL benchmarks while training orders of magnitude fewer parameters. Importantly, we also show that MAPL is effective in the low-data and in-domain settings thanks to its reduced number of trainable parameters. We leave as future work exploring training on a weighted mixture of image-text datasets, evaluating on more downstream tasks such as NLVR2 (Suhr et al., 2019) and Visual Dialog (Das et al., 2017), and investigating the use of masked LMs (Schick and Schütze, 2021; Chung et al., 2022) with MAPL.

Limitations

MAPL achieves reasonable performance on VL tasks, but it is still far from the performance of recent methods leveraging large-scale data and compute. On the other hand, MAPL is a preferable alternative in scenarios with constrained computational resources.

We observed our mapping network is sensitive to initialization, so different random seeds can yield non-negligible variance in downstream performance. We think this might be related to the reduced number of trainable parameters. We tried to reduce the effect of this variance by reporting average performance across different seeds. We also observed MAPL struggles to leverage more shots for in-context learning. We hypothesize this could be caused by our model being trained on single image-caption pairs – as opposed to the sequences of multiple images and texts seen during few-shot transfer, so a better pretext task might help (see App. A.1 for further discussion).

MAPL builds on top of pre-trained vision-only and language-only models, inheriting their capabilities but also their limitations. An important risk is that our model might inherit the existing social, gender or racial biases of pre-trained models. However, our limited qualitative analysis (see App. A.8) shows that providing visual information significantly changes the prior answer distribution of the LM. Therefore, how much of the underlying bias is retained remains an empirical question.

Ethics Statement

Model recycling. MAPL reuses vision-only and language-only foundation models. Hence, the expensive computational resources used to train these models can be amortized to help reduce energy and carbon costs.

Public datasets. MAPL is trained uniquely on publicly available datasets, which facilitates reproducibility and provides transparency on the origin and the characteristics of the data the model has seen.

Undesired biases. MAPL could be exposed to undesired biases from different sources. The pre-trained vision encoder might have been trained with data where certain races or genders are underrepresented, hence biasing our representation of images. The pre-trained LM might also be biased towards generating toxic or offensive language when fed with certain prompts. Finally, the image-text data used to align the representation spaces of such models was annotated by humans, so it might reflect a biased view of the world.

Broader impact. This work shows how one can easily adapt pre-trained vision encoders and LMs for multimodal tasks. Given the parameter-efficiency of our method, we believe it should be of great interest to the sections of the community that do not have access to large compute resources (e.g., small academic labs and independent researchers), and for low-data applications. While MAPL can be applied in many useful applications (e.g., aiding visually-impaired people), it also makes it simpler to create malicious or offensive multimodal systems from existing unimodal models. Further research efforts are needed on how to safely deploy such systems so that their behavior always aligns with ethical values.

Acknowledgements

We thank Pauline Luc, Siva Reddy, Lisa Anne Hendricks and Chris Dyer for their constructive feedback. We are grateful to the Mila IDT team for their technical support with the computational infrastructure. During this project, Aishwarya Agrawal was supported by the Canada CIFAR AI Chair award. We would also like to thank Samsung Electronics Co., Ldt. for funding this research.

References

Aishwarya Agrawal, Ivana Kajić, Emanuele Bugliarello, Elnaz Davoodi, Anita Gergely, Phil Blunsom, and Aida Nematzadeh. 2022. Rethinking evaluation practices in visual question answering: A case study on out-of-distribution generalization. *arXiv preprint arXiv:2205.12191*.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. 2021. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. *arXiv preprint arXiv:2102.10407*.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.

Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. In *International Conference on Machine Learning*, pages 1931–1942. PMLR.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Wenliang Dai, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. 2022. Enabling multimodal generation on CLIP via vision-language knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2383–2395, Dublin, Ireland. Association for Computational Linguistics.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. 2017. Visual dialog. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 326–335.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

Constantin Eichenberg, Sidney Black, Samuel Weinbach, Letitia Parcalabescu, and Anette Frank. 2021. Magma–multimodal augmentation of generative models through adapter-based finetuning. *arXiv preprint arXiv:2112.05253*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913.

Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. 2018. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3608–3617.

Yaru Hao, Haoyu Song, Li Dong, Shaohan Huang, Zewen Chi, Wenhui Wang, Shuming Ma, and Furu Wei. 2022. Language models are general-purpose interfaces. *arXiv preprint arXiv:2206.06336*.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR.

Woojeong Jin, Yu Cheng, Yelong Shen, Weizhu Chen, and Xiang Ren. 2022. A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2763–2775, Dublin, Ireland. Association for Computational Linguistics.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Alex M Lamb, Anirudh Goyal ALIAS PARTH GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks. *Advances in neural information processing systems*, 29.

Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in Neural Information Processing Systems*, 34.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Ziyang Luo, Yadong Xi, Rongsheng Zhang, and Jing Ma. 2022. VC-GPT: visual conditioned GPT for end-to-end generative vision-and-language pre-training. *CoRR*, abs/2201.12723.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3195–3204.

Jack Merullo, Louis Castricato, Carsten Eickhoff, and Ellie Pavlick. 2022. Linearly mapping from image to text space. *arXiv preprint arXiv:2209.15162*.

Ron Mokady, Amir Hertz, and Amit H Bermano. 2021. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7.

David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. 2017. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *NAACL-HLT (2)*.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. 2020. Textcaps: a dataset for image captioning with reading comprehension. In *European Conference on Computer Vision*, pages 742–758. Springer.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8317–8326.

Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. 2019. A corpus for reasoning about natural language grounded in photographs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6418–6428.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111.

Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34.

Vladimir N Vapnik and A Ya Chervonenkis. 2015. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. 2021. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. 2021. An empirical study of gpt-3 for few-shot knowledge-based vqa. *arXiv preprint arXiv:2109.05014*.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5579–5588.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

A Appendix

A.1 Leveraging more shots

In Table 2, we observe MAPL's performance rapidly plateaus as the number of few-shot examples increases beyond 4. We hypothesize this could be related to the mapping network being trained on single image-caption pairs, and/or the visual embeddings still not being fully in-distribution with the language embeddings. Intuitively, a handful of examples may often help with task location (Reynolds and McDonell, 2021); however, the more shots are added, the more out-of-distribution the multimodal prompt becomes. This issue could be mitigated with in-context example selection (Yang et al., 2021) or better mixing of visual and textual modalities.

A.2 0-shot VQAv2 results with OPT

In Table 5, we observe Frozen*-OPT outperforms MAPL-OPT on 0-shot VQAv2. Upon close inspection, we notice MAPL-OPT often generates longer answers for yes/no questions, which receive a score of 0 according to VQA accuracy and VQAv2 reference answers – this is a problem of the metric and not the model itself (Agrawal et al., 2022). After filtering all answers starting with "yes" or "no" to leave only the short answer, MAPL-OPT achieves a VQA accuracy of 40.14% while Frozen*-OPT only reaches 32.03%.

A.3 4-shot results with OPT

In Table 5, we also observe that few-shot VQA performance is considerably lower for configurations using OPT-6.7B as language model. This is possibly due to the lack of a relative positional encoding (Shaw et al., 2018) in OPT, which is required for the transformer to generalize to prompt sequences where an image is not always in the first absolute position, or which contain more than one image (Tsimpoukelli et al., 2021).

A.4 Implementation details on simpler mapping networks

In Sec. 4.5, we ablate the choice of mapping network architecture and replace it by simpler architectures. Similarly to Eichenberg et al. (2021); Merullo et al. (2022), the linear mapping is applied per-position on top of a flattened grid of visual features, and it projects from $D_i = 1024$ to $D_o = 4096$ dimensions (4.2M parameters). The output sequence length L_o is thus equal to $L_i = 257$ (instead of 32) – as explained in Sec. 3.1, this increases the computational complexity in the subsequent LM, which in turn increases training and inference time considerably. Similarly to Mokady et al. (2021), the 2-layer MLP is applied on top of a global vector of visual features. The MLP's hidden dimensionality D_h is equal to $D_i = 1024$, and the output dimensionality is $D_o = 32 * 4096$, which we split into 32 vectors of 4096 dimensions (135.3M parameters).

A.5 Additional ablation studies

Table 6 shows the results of our additional ablation studies. Unless specified otherwise, we perform all ablations on MAPL $_{\text{CC-clean}}$ trained with 100% of the data. These experiments are run only once and early stopping is based on the validation split of Conceptual Captions.

Pre-trained vision encoder. We ablate the pre-trained vision encoder used to compute image representations. We report results in row (i) of Table 6. We compare two CLIP (Radford et al., 2021) variants, our choice based on the ViT-L/14 backbone and the ViT-B/32 backbone. Indeed, the ViT-L/14 based vision encoder has an average +20% advantage over the ViT-B/32 variant. We hypothesize this improvement is probably due to finer-grained image patches and a bigger model size.

Global vs. grid visual features. Grid features – as opposed to global features – preserve the spatial information in images. This kind of fine-grained information might be useful for complex VL tasks. To measure the impact of grid features, we train a version of MAPL where we use the global image representation from CLIP's multimodal embedding space. Results are reported in row (ii) of Table 6. We observe an average -10% drop in performance, validating our choice of using grid over global visual features.

Mapping network architecture. We ablate the architectural design of our mapping network in rows (iii) and (v) of Table 6. First, we ablate the size of our mapping network in terms of depth and hidden size. We explore three options: Small (2 layers and hidden size of 128), Medium (4 layers and hidden size of 256), and Large (8 layers and hidden size of 512). We see that using a smaller mapping network generally performs slightly worse than the base model. On the other hand, using a larger mapping network improves only in image captioning

	Ablated setting	**Original value**	**Changed value**	**VQAv2 4-shot**	**OK-VQA 4-shot**	**TextVQA 4-shot**	**VizWiz-VQA 4-shot**	**CC CIDEr**	**COCO CIDEr**	**TextCaps CIDEr**	**VizWiz-Caps CIDEr**	**Overall Δ 4-shot**	**Overall Δ CIDEr**
	MAPL			46.39	25.49	9.87	20.02	71.90	54.90	23.30	21.70	0	0
(i)	Vision encoder	CLIP-ViT-L/14	CLIP-ViT-B/32	43.48	24.43	7.85	15.97	59.20	47.00	19.00	15.70	-2.51	-7.73
(ii)	Visual features	Grid	Global	43.75	22.90	8.81	18.20	66.70	49.70	18.40	19.90	-2.03	-4.28
(iii)	Mapping network size	Medium	Small	44.83	26.37	9.68	17.78	68.30	55.50	21.30	20.80	-0.78	-1.48
			Large	45.03	23.92	8.88	19.01	73.40	57.10	24.00	23.10	-1.23	+1.45
(iv)	Output seq. length	32	16	44.18	25.16	9.01	18.15	72.80	56.20	22.50	21.80	-1.32	+0.37
			64	45.22	25.07	10.35	18.89	74.80	58.30	24.30	24.80	-0.56	+2.60
(v)	Learned constant embeddings	Yes	No	40.87	19.31	11.42	16.79	80.52	57.49	30.07	26.16	-3.35	+5.61
(vi)	Data quality Visual features	Clean Grid	Noisy Global	42.80	22.59	6.06	17.33	93.80	42.10	15.60	15.70	-3.25	-1.15

Table 6: Ablation studies. "Overall Δ" refers to the difference (ablated model - base model), averaged across datasets per task.

tasks, while increasing significantly the number of trainable parameters (from 3.4M to 19.5M). We also ablate the output sequence length L_o of our mapping network. Similarly, reducing the output sequence length to 16 yields slightly lower performance overall, and increasing it to 64 only improves in image captioning tasks. In the extreme, we completely remove the learned constant embeddings and output the same sequence length coming from the vision encoder, i.e., $L_o = L_i = 257$. Following the trend, increasing the number of mapped visual embeddings is beneficial for image captioning but hurts VQA performance, while notably reducing training and inference throughput.

Data quality & visual features. This ablation setting aims to be the most similar to Frozen: training on the *full* (noisy) Conceptual Captions dataset while using global visual features. Results are reported in row (vi) of Table 6. The overall performance is worse than that of our base model (-16% on average), but still better than Frozen* $_{\text{CC}}$ on Table 4 (+81% on average). This validates our choice of using grid visual features while training on a subset of cleaner data.

A.6 Additional qualitative results

Figures 4-15 show additional qualitative results of MAPL $_{\text{CC-clean}}$ on random samples from different image captioning and VQA datasets. For VQA, in-context learning from 4 shots is performed.

A.7 Interpretability of visual embeddings

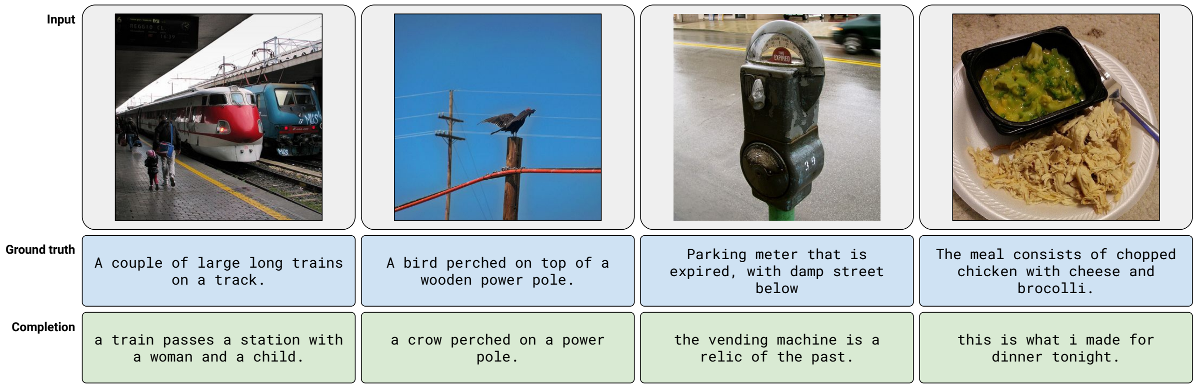
Using MAPL $_{CC-clean}$, we extract mapped visual embeddings (after the mapping network) for ∼30 images from the COCO Karpathy-test set, and compute the nearest token embeddings (from the LM's vocabulary) using cosine similarity. We rarely found the top-5 nearest tokens correspond to concepts present in the image, suggesting these embeddings are not interpretable. We hypothesize this is perhaps because they carry a combination of task-inducing and image-specific information, also pointed out by Mokady et al. (2021). We further cluster the mapped visual embeddings with K-means, and observe that each cluster often represents some visual concept (e.g., animals, food, sports). This means the mapped visual embeddings retain visual information from the vision encoder, which we also verify with MAPL's performance on VL benchmarks.

A.8 Analysis of VQA answer distributions

In this section, we show the distribution of answers for selected VQAv2 question types. We compare MAPL with several baselines of our model to get insights into how the model's predictions change when training on increasing multimodal data. For the text-only baseline, we only provide the question text to the LM. This is different from the previously introduced blind baseline (Sec. 4.1), where a blacked-out image is also provided. In particular, we compare the predicted answer distribution of MAPL $_{\text{COCO}}$ evaluated on on zero- and few-shot VQA with the aforementioned baselines and the ground truth. Overall, we observe the predicted answer distribution gets closer to the ground truth answer distribution (Figure 21) as more information from the image-question pair is provided to the model. We notice a considerable shift in the answer distribution from the text-only baseline (Figure 16) to the blind baseline, which demonstrates the impact caused by the captions alone. Moreover, we see the predicted answer distribution of MAPL zero-shot is closer to the ground truth answer distribution than that of the blind baseline (Figure 17), which indicates that MAPL is leveraging the additional information from the visual input. For instance, we observe MAPL's predicted answer distribution for the "what color" question type (second column) looks more similar to the ground truth distribution compared to the text-only and blind baselines. Finally, when performing in-context learning from four shots (Figure 19), we see the answer distribution gets even closer to the ground truth distribution. However, we do not observe much difference in answer distribution when increasing the number of shots from four to eight (Figure 20).

Ground truth	stadium coming alive on game day !.	and what 's the umbrella for ? a woman braves deep flood water.	my little brother will probably have this cake for his birthday.	the hand of person --.
Completion	sunset from the stadium.	a woman holds an umbrella as she walks through the flooded streets.	a cake for a video game themed birthday party.	a superman logo with a fist and lightning bolt.

Figure 4: MAPL's image captioning on Conceptual Captions.

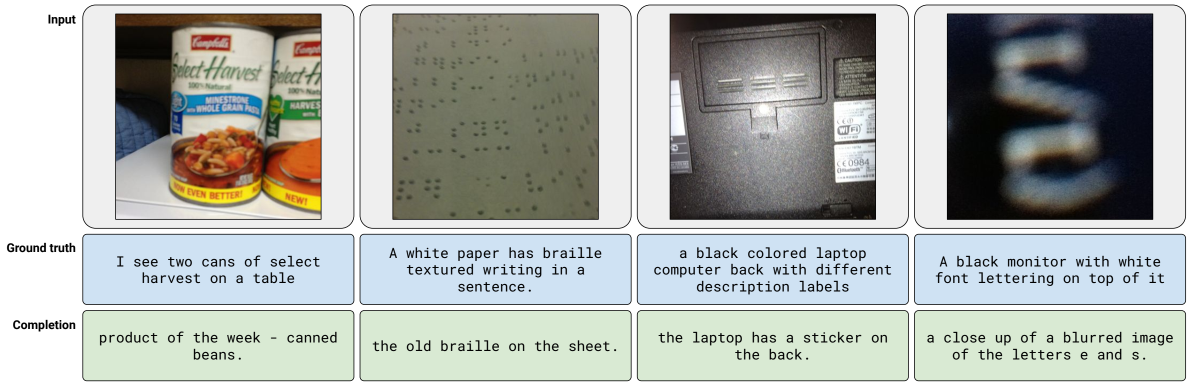


Ground truth	A couple of large long trains on a track.	A bird perched on top of a wooden power pole.	Parking meter that is expired, with damp street below	The meal consists of chopped chicken with cheese and brocolli.
Completion	a train passes a station with a woman and a child.	a crow perched on a power pole.	the vending machine is a relic of the past.	this is what i made for dinner tonight.

Figure 5: MAPL's image captioning on COCO Captions.

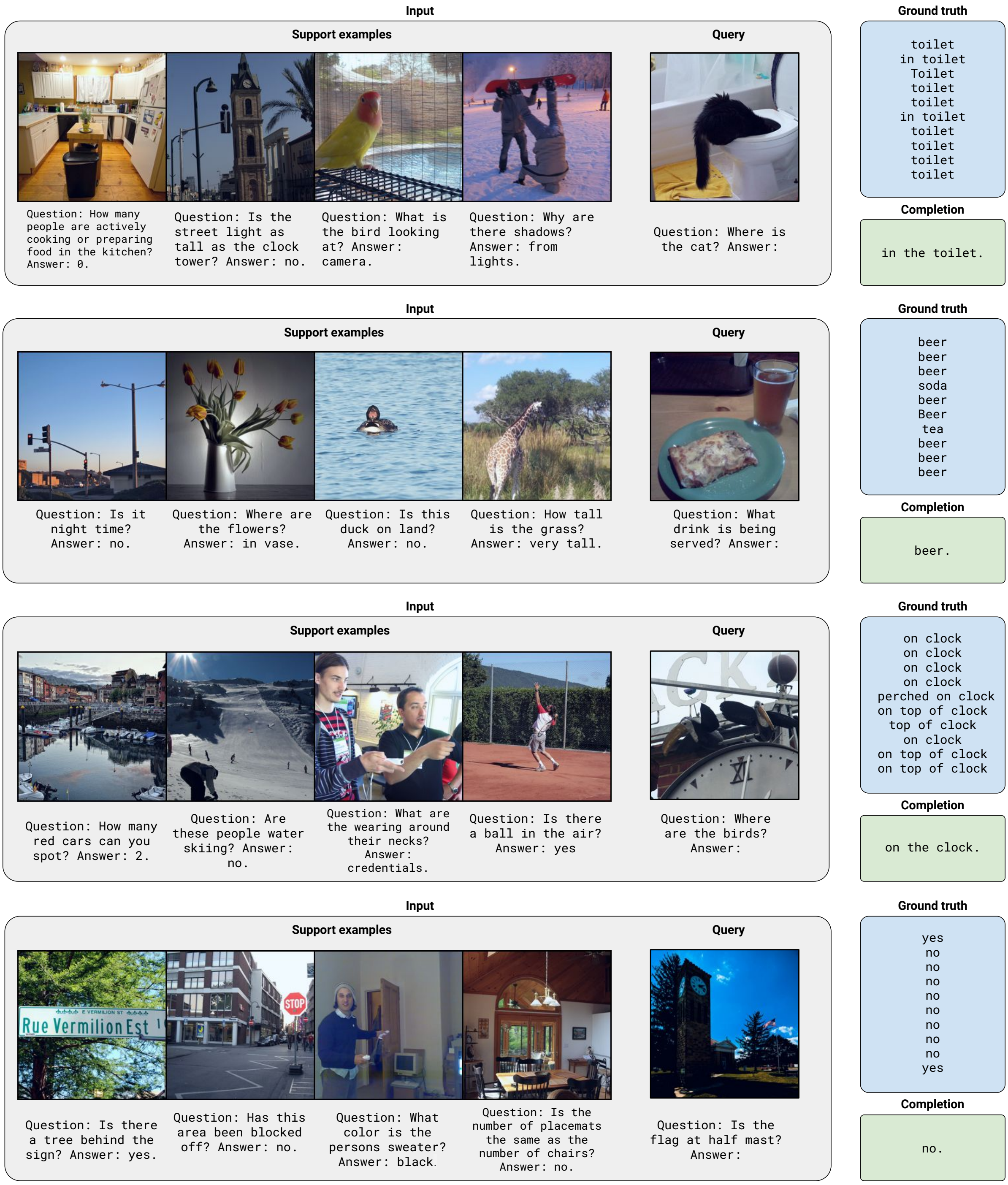
Ground truth	An old book has a picture of a baby sat in a swing and an article of the same name next to it.	Tyler Hanover is displayed on a outfield scoreboard.	A English two pence coin is seen close up wedged into a piece of wood.	Imperial Oat oatmeal stout is shown having a yellow can with a horse drawing on it.
Completion	a child's swing from a magazine.	a sign outside the stadium with a picture of baseball player on it.	a penny on a tree.	brewery's new beer, a golden ale, is named after a horse.

Figure 6: MAPL's image captioning on TextCaps.



Ground truth	I see two cans of select harvest on a table	A white paper has braille textured writing in a sentence.	a black colored laptop computer back with different description labels	A black monitor with white font lettering on top of it
Completion	product of the week - canned beans.	the old braille on the sheet.	the laptop has a sticker on the back.	a close up of a blurred image of the letters e and s.

Figure 7: MAPL's image captioning on VizWiz Captions.



Question: How many people are actively cooking or preparing food in the kitchen? Answer: 0.

Question: Is the street light as tall as the clock tower? Answer: no.

Question: What is the bird looking at? Answer: camera.

Question: Why are there shadows? Answer: from lights.

Question: Where is the cat? Answer:

toilet
in toilet
Toilet
toilet
toilet
in toilet
toilet
toilet
toilet
toilet

in the toilet.

Question: Is it night time? Answer: no.

Question: Where are the flowers? Answer: in vase.

Question: Is this duck on land? Answer: no.

Question: How tall is the grass? Answer: very tall.

Question: What drink is being served? Answer:

beer
beer
beer
soda
beer
Beer
tea
beer
beer
beer

beer.

Question: How many red cars can you spot? Answer: 2.

Question: Are these people water skiing? Answer: no.

Question: What are the wearing around their necks? Answer: credentials.

Question: Is there a ball in the air? Answer: yes

Question: Where are the birds? Answer:

on clock
on clock
on clock
on clock
perched on clock
on top of clock
top of clock
on clock
on top of clock
on top of clock

on the clock.

Question: Is there a tree behind the sign? Answer: yes.

Question: Has this area been blocked off? Answer: no.

Question: What color is the persons sweater? Answer: black.

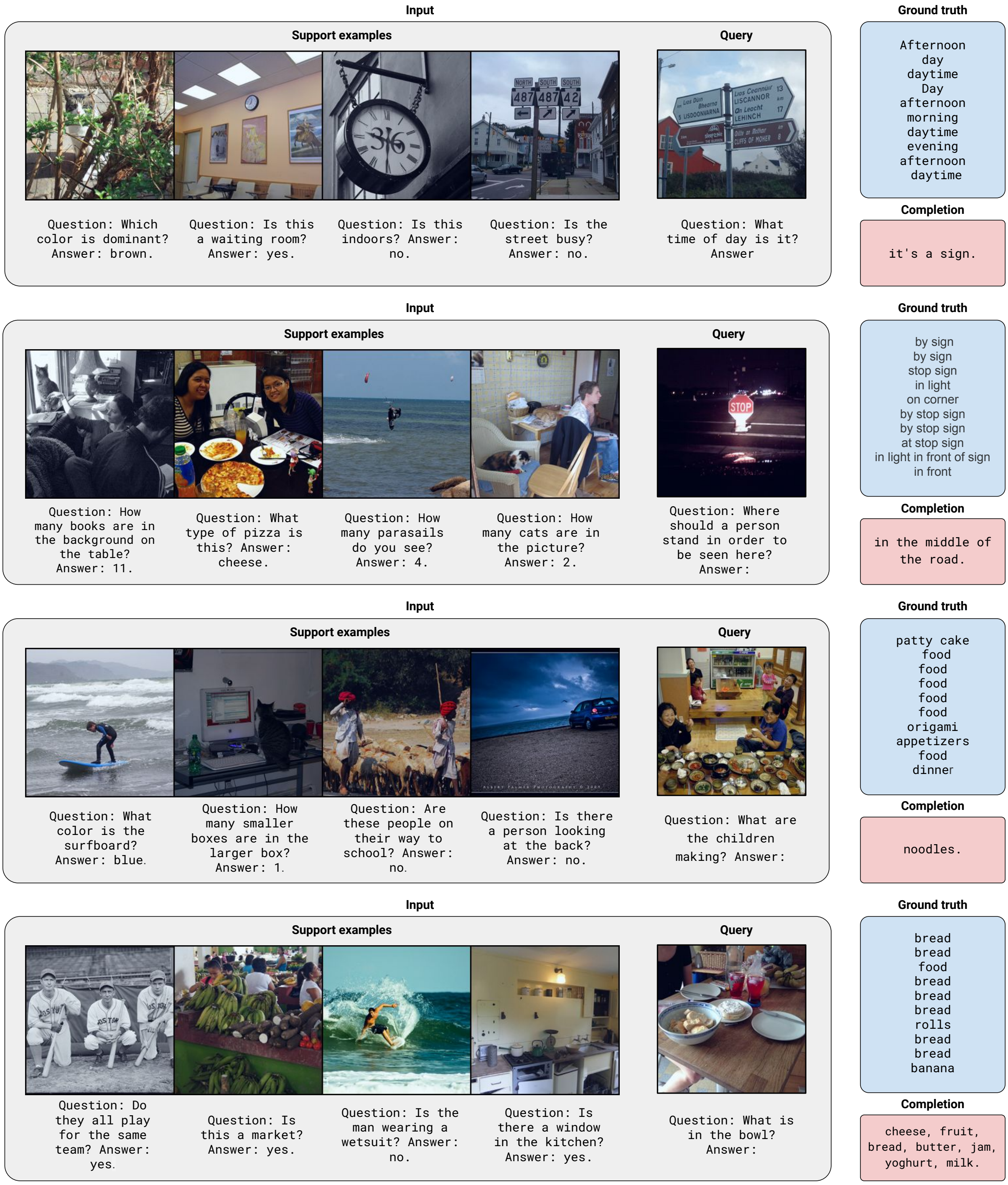
Question: Is the number of placemats the same as the number of chairs? Answer: no.

Question: Is the flag at half mast? Answer:

yes
no
no
no
no
no
no
no
no
yes

no.

Figure 8: MAPL's 4-shot VQA on VQAv2, success cases.



Question: Which color is dominant? Answer: brown.

Question: Is this a waiting room? Answer: yes.

Question: Is this indoors? Answer: no.

Question: Is the street busy? Answer: no.

Question: What time of day is it? Answer

Afternoon
day
daytime
Day
afternoon
morning
daytime
evening
afternoon
daytime

it's a sign.

Question: How many books are in the background on the table? Answer: 11.

Question: What type of pizza is this? Answer: cheese.

Question: How many parasails do you see? Answer: 4.

Question: How many cats are in the picture? Answer: 2.

Question: Where should a person stand in order to be seen here? Answer:

by sign
by sign
stop sign
in light
on corner
by stop sign
by stop sign
at stop sign
in light in front of sign
in front

in the middle of the road.

Question: What color is the surfboard? Answer: blue.

Question: How many smaller boxes are in the larger box? Answer: 1.

Question: Are these people on their way to school? Answer: no.

Question: Is there a person looking at the back? Answer: no.

Question: What are the children making? Answer:

patty cake
food
food
food
food
food
origami
appetizers
food
dinner

noodles.

Question: Do they all play for the same team? Answer: yes.

Question: Is this a market? Answer: yes.

Question: Is the man wearing a wetsuit? Answer: no.

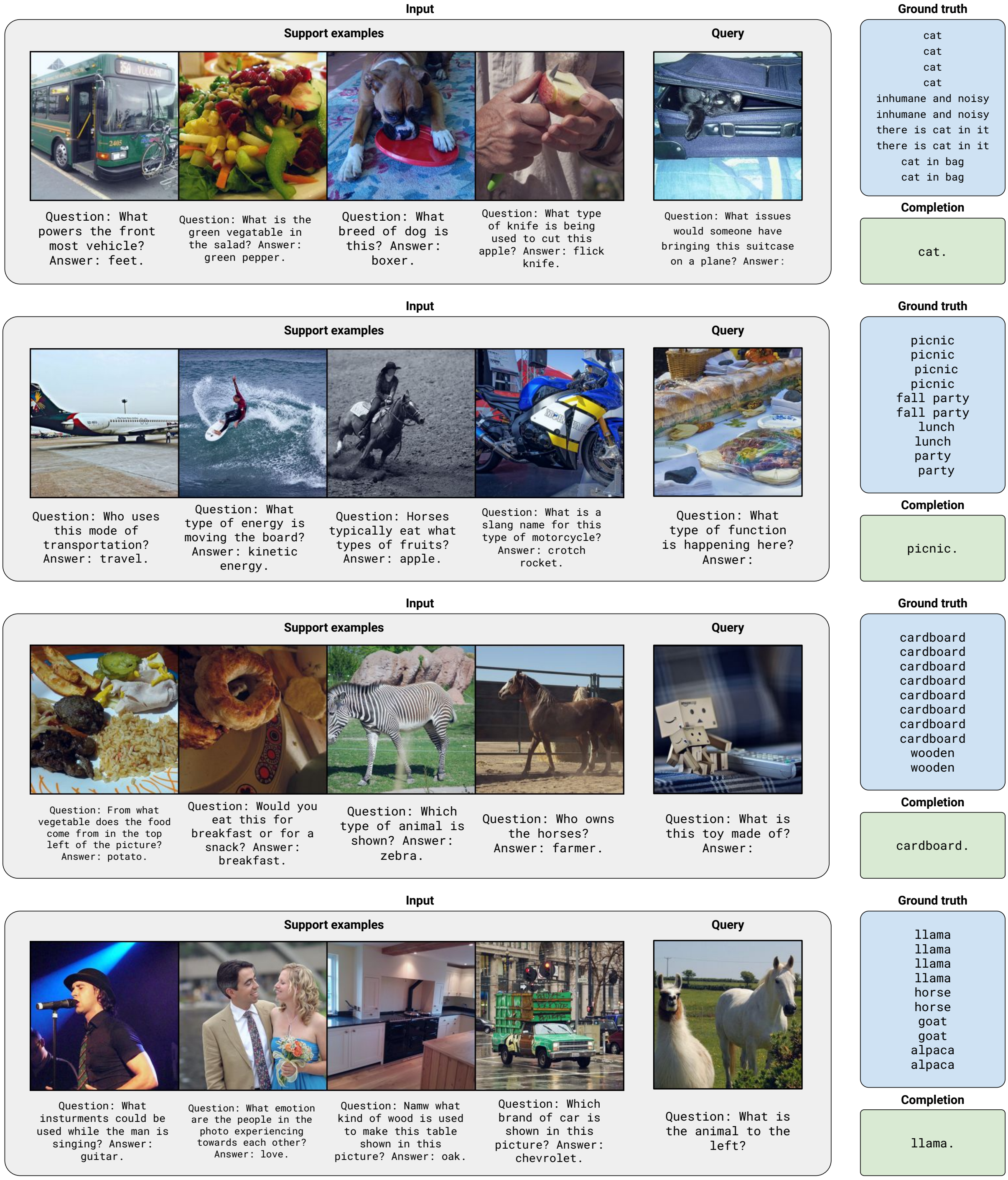
Question: Is there a window in the kitchen? Answer: yes.

Question: What is in the bowl? Answer:

bread
bread
food
bread
bread
bread
rolls
bread
bread
banana

cheese, fruit, bread, butter, jam, yoghurt, milk.

Figure 9: MAPL's 4-shot VQA on VQAv2, failure cases.



Question: What powers the front most vehicle? Answer: feet.

Question: What is the green vegatable in the salad? Answer: green pepper.

Question: What breed of dog is this? Answer: boxer.

Question: What type of knife is being used to cut this apple? Answer: flick knife.

Question: What issues would someone have bringing this suitcase on a plane? Answer:

cat
cat
cat
cat
inhumane and noisy
inhumane and noisy
there is cat in it
there is cat in it
cat in bag
cat in bag

cat.

Question: Who uses this mode of transportation? Answer: travel.

Question: What type of energy is moving the board? Answer: kinetic energy.

Question: Horses typically eat what types of fruits? Answer: apple.

Question: What is a slang name for this type of motorcycle? Answer: crotch rocket.

Question: What type of function is happening here? Answer:

picnic
picnic
picnic
picnic
fall party
fall party
lunch
lunch
party
party

picnic.

Question: From what vegetable does the food come from in the top left of the picture? Answer: potato.

Question: Would you eat this for breakfast or for a snack? Answer: breakfast.

Question: Which type of animal is shown? Answer: zebra.

Question: Who owns the horses? Answer: farmer.

Question: What is this toy made of? Answer:

cardboard
cardboard
cardboard
cardboard
cardboard
cardboard
cardboard
cardboard
wooden
wooden

cardboard.

Question: What insturments could be used while the man is singing? Answer: guitar.

Question: What emotion are the people in the photo experiencing towards each other? Answer: love.

Question: Namw what kind of wood is used to make this table shown in this picture? Answer: oak.

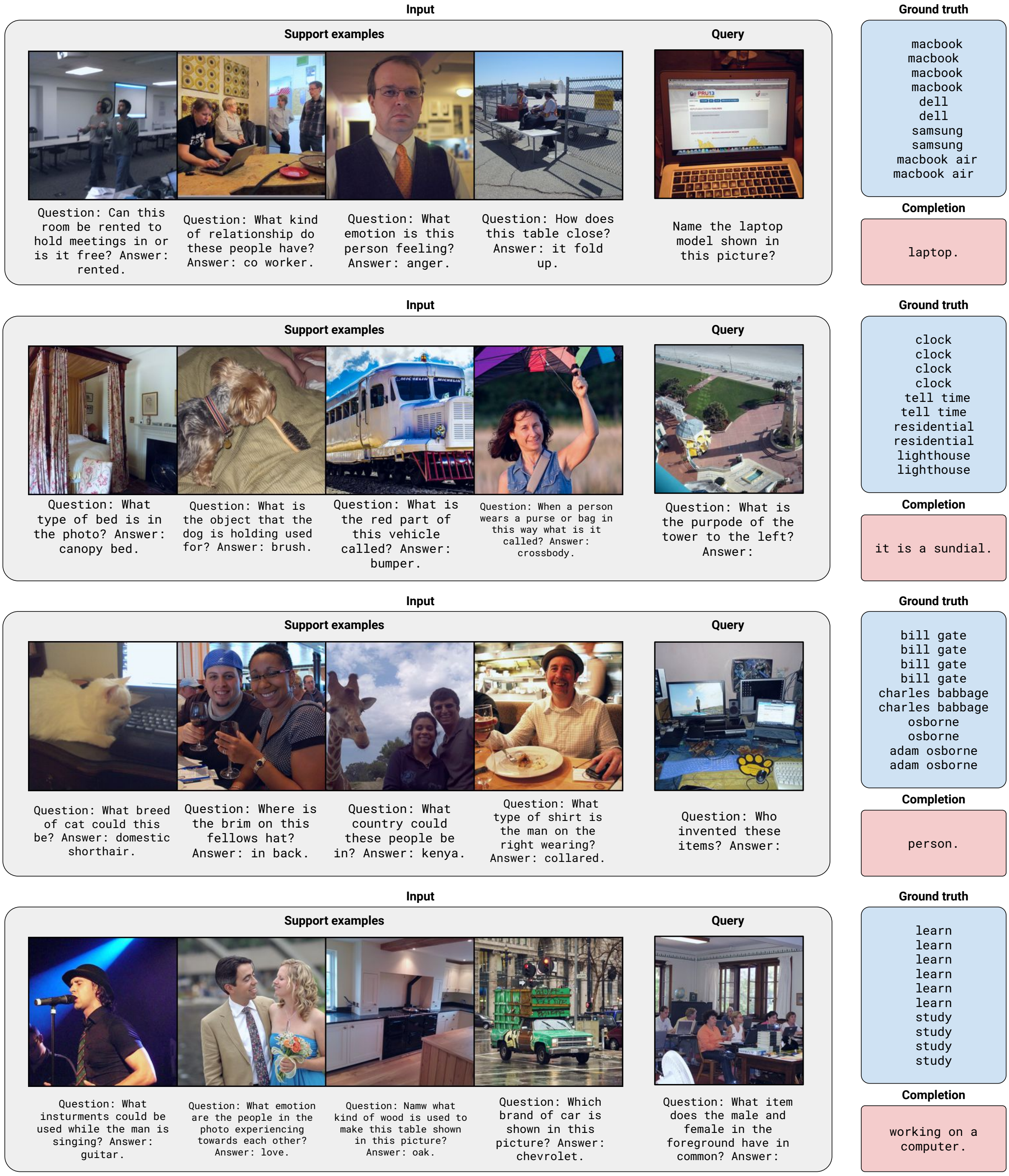
Question: Which brand of car is shown in this picture? Answer: chevrolet.

Question: What is the animal to the left?

llama
llama
llama
llama
horse
horse
goat
goat
alpaca
alpaca

llama.

Figure 10: MAPL's 4-shot VQA on OK-VQA, success cases.



Question: Can this room be rented to hold meetings in or is it free? Answer: rented.

Question: What kind of relationship do these people have? Answer: co worker.

Question: What emotion is this person feeling? Answer: anger.

Question: How does this table close? Answer: it fold up.

Name the laptop model shown in this picture?

macbook
macbook
macbook
macbook
dell
dell
samsung
samsung
macbook air
macbook air

laptop.

Question: What type of bed is in the photo? Answer: canopy bed.

Question: What is the object that the dog is holding used for? Answer: brush.

Question: What is the red part of this vehicle called? Answer: bumper.

Question: When a person wears a purse or bag in this way what is it called? Answer: crossbody.

Question: What is the purpode of the tower to the left? Answer:

clock
clock
clock
clock
tell time
tell time
residential
residential
lighthouse
lighthouse

it is a sundial.

Question: What breed of cat could this be? Answer: domestic shorthair.

Question: Where is the brim on this fellows hat? Answer: in back.

Question: What country could these people be in? Answer: kenya.

Question: What type of shirt is the man on the right wearing? Answer: collared.

Question: Who invented these items? Answer:

bill gate
bill gate
bill gate
bill gate
charles babbage
charles babbage
osborne
osborne
adam osborne
adam osborne

person.

Question: What insturments could be used while the man is singing? Answer: guitar.

Question: What emotion are the people in the photo experiencing towards each other? Answer: love.

Question: Namw what kind of wood is used to make this table shown in this picture? Answer: oak.

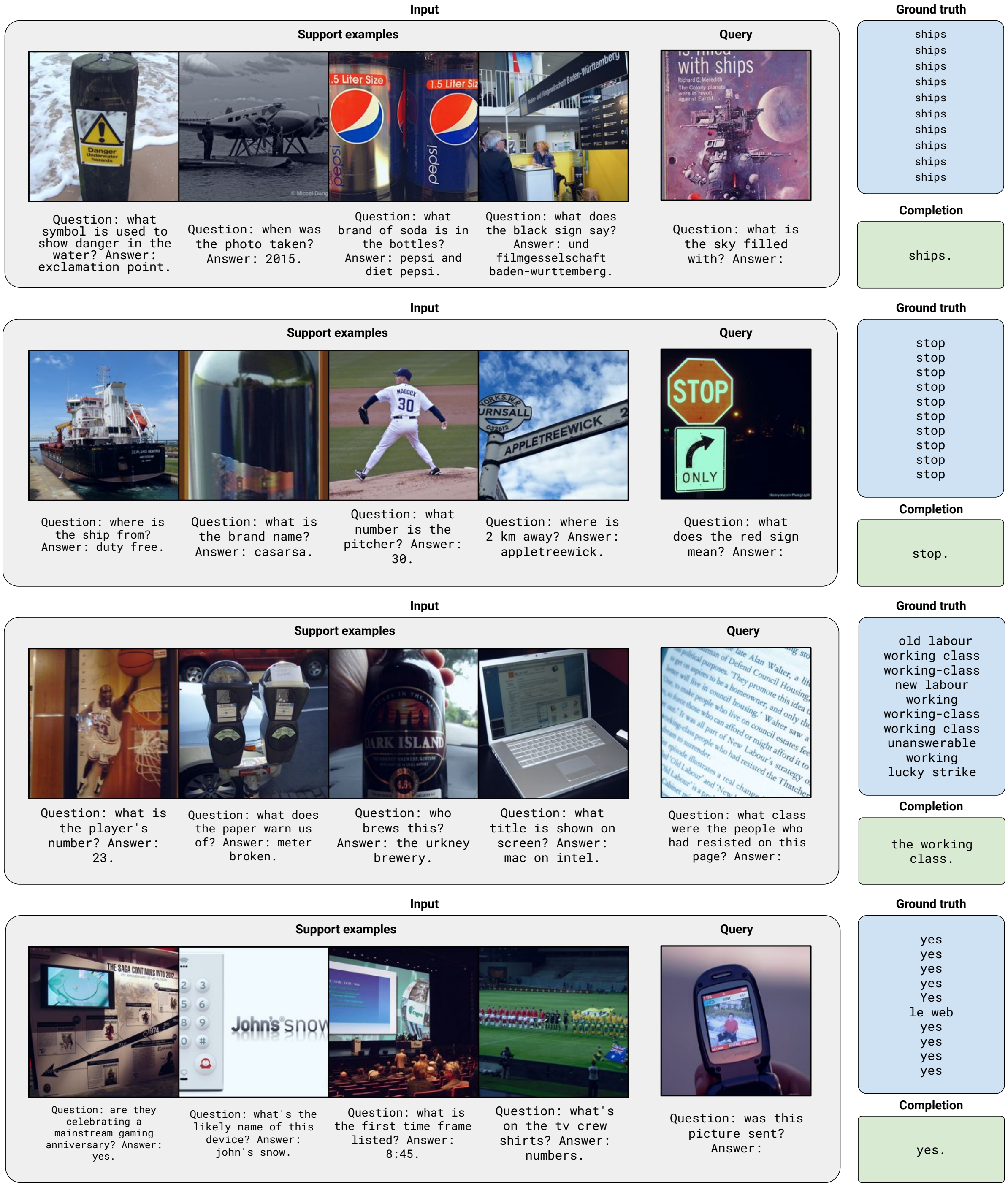
Question: Which brand of car is shown in this picture? Answer: chevrolet.

Question: What item does the male and female in the foreground have in common? Answer:

learn
learn
learn
learn
learn
learn
study
study
study
study

working on a computer.

Figure 11: MAPL's 4-shot VQA on OK-VQA, failure cases.



Question: what symbol is used to show danger in the water? Answer: exclamation point.

Question: when was the photo taken? Answer: 2015.

Question: what brand of soda is in the bottles? Answer: pepsi and diet pepsi.

Question: what does the black sign say? Answer: und filmgesselschaft baden-wurttemberg.

Question: what is the sky filled with? Answer:

ships
ships
ships
ships
ships
ships
ships
ships
ships
ships

ships.

Question: where is the ship from? Answer: duty free.

Question: what is the brand name? Answer: casarsa.

Question: what number is the pitcher? Answer: 30.

Question: where is 2 km away? Answer: appletreewick.

Question: what does the red sign mean? Answer:

stop
stop
stop
stop
stop
stop
stop
stop
stop
stop

stop.

Question: what is the player's number? Answer: 23.

Question: what does the paper warn us of? Answer: meter broken.

Question: who brews this? Answer: the urkney brewery.

Question: what title is shown on screen? Answer: mac on intel.

Question: what class were the people who had resisted on this page? Answer:

old labour
working class
working-class
new labour
working
working-class
working class
unanswerable
working
lucky strike

the working class.

Question: are they celebrating a mainstream gaming anniversary? Answer: yes.

Question: what's the likely name of this device? Answer: john's snow.

Question: what is the first time frame listed? Answer: 8:45.

Question: what's on the tv crew shirts? Answer: numbers.

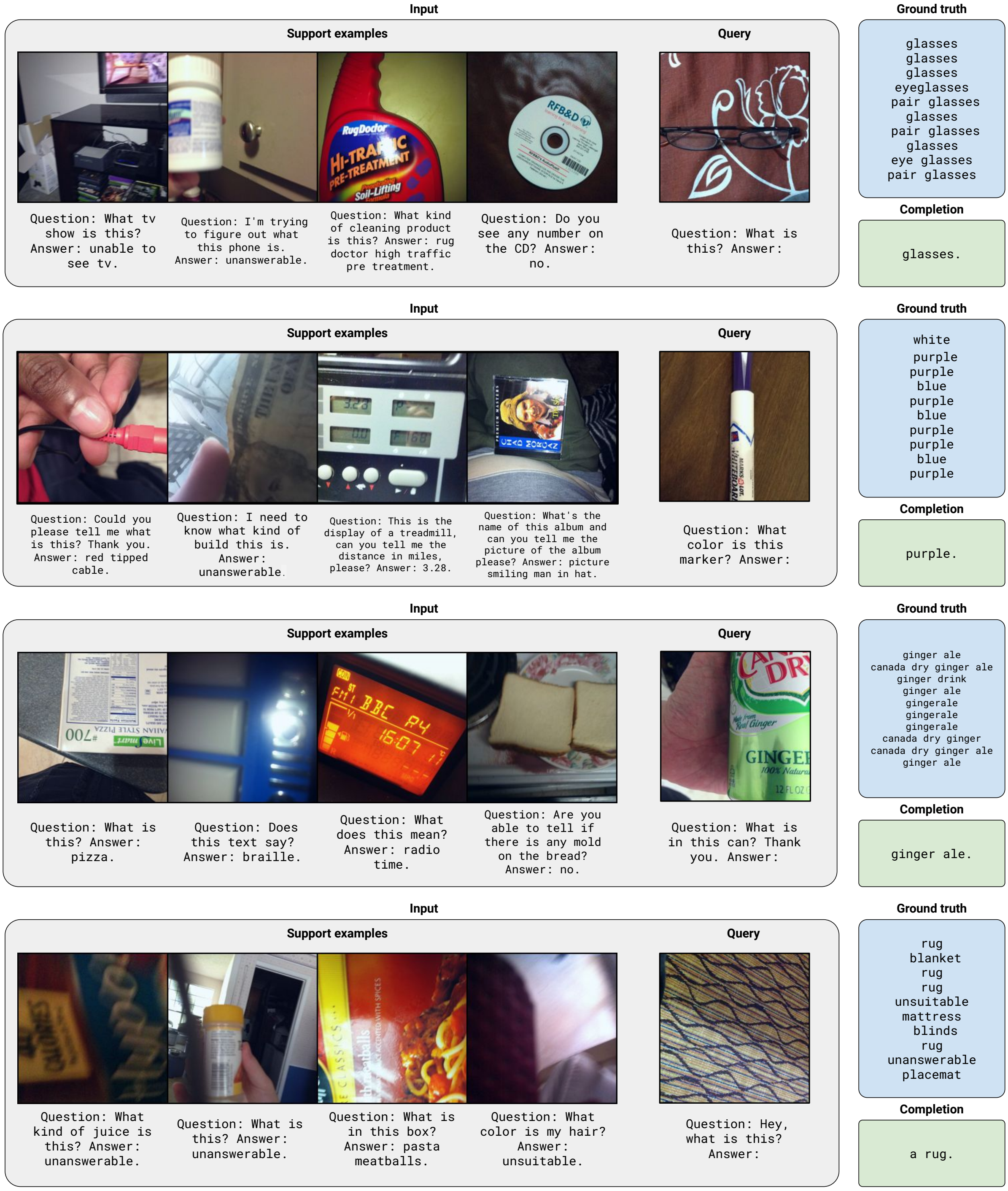
Question: was this picture sent? Answer:

yes
yes
yes
yes
Yes
le web
yes
yes
yes
yes

yes.

Figure 12: MAPL's 4-shot VQA on TextVQA, success cases.

Figure 13: MAPL's 4-shot VQA on TextVQA, failure cases.



Question: What tv show is this? Answer: unable to see tv.

Question: I'm trying to figure out what this phone is. Answer: unanswerable.

Question: What kind of cleaning product is this? Answer: rug doctor high traffic pre treatment.

Question: Do you see any number on the CD? Answer: no.

Question: What is this? Answer:

glasses
glasses
glasses
eyeglasses
pair glasses
glasses
pair glasses
glasses
eye glasses
pair glasses

glasses.

Question: Could you please tell me what is this? Thank you. Answer: red tipped cable.

Question: I need to know what kind of build this is. Answer: unanswerable.

Question: This is the display of a treadmill, can you tell me the distance in miles, please? Answer: 3.28.

Question: What's the name of this album and can you tell me the picture of the album please? Answer: picture smiling man in hat.

Question: What color is this marker? Answer:

white
purple
purple
blue
purple
blue
purple
purple
blue
purple

purple.

Question: What is this? Answer: pizza.

Question: Does this text say? Answer: braille.

Question: What does this mean? Answer: radio time.

Question: Are you able to tell if there is any mold on the bread? Answer: no.

Question: What is in this can? Thank you. Answer:

ginger ale
canada dry ginger ale
ginger drink
ginger ale
gingerale
gingerale
gingerale
canada dry ginger
canada dry ginger ale
ginger ale

ginger ale.

Question: What kind of juice is this? Answer: unanswerable.

Question: What is this? Answer: unanswerable.

Question: What is in this box? Answer: pasta meatballs.

Question: What color is my hair? Answer: unsuitable.

Question: Hey, what is this? Answer:

rug
blanket
rug
rug
unsuitable
mattress
blinds
rug
unanswerable
placemat

a rug.

Figure 14: MAPL's 4-shot VQA on VizWiz-VQA, success cases.

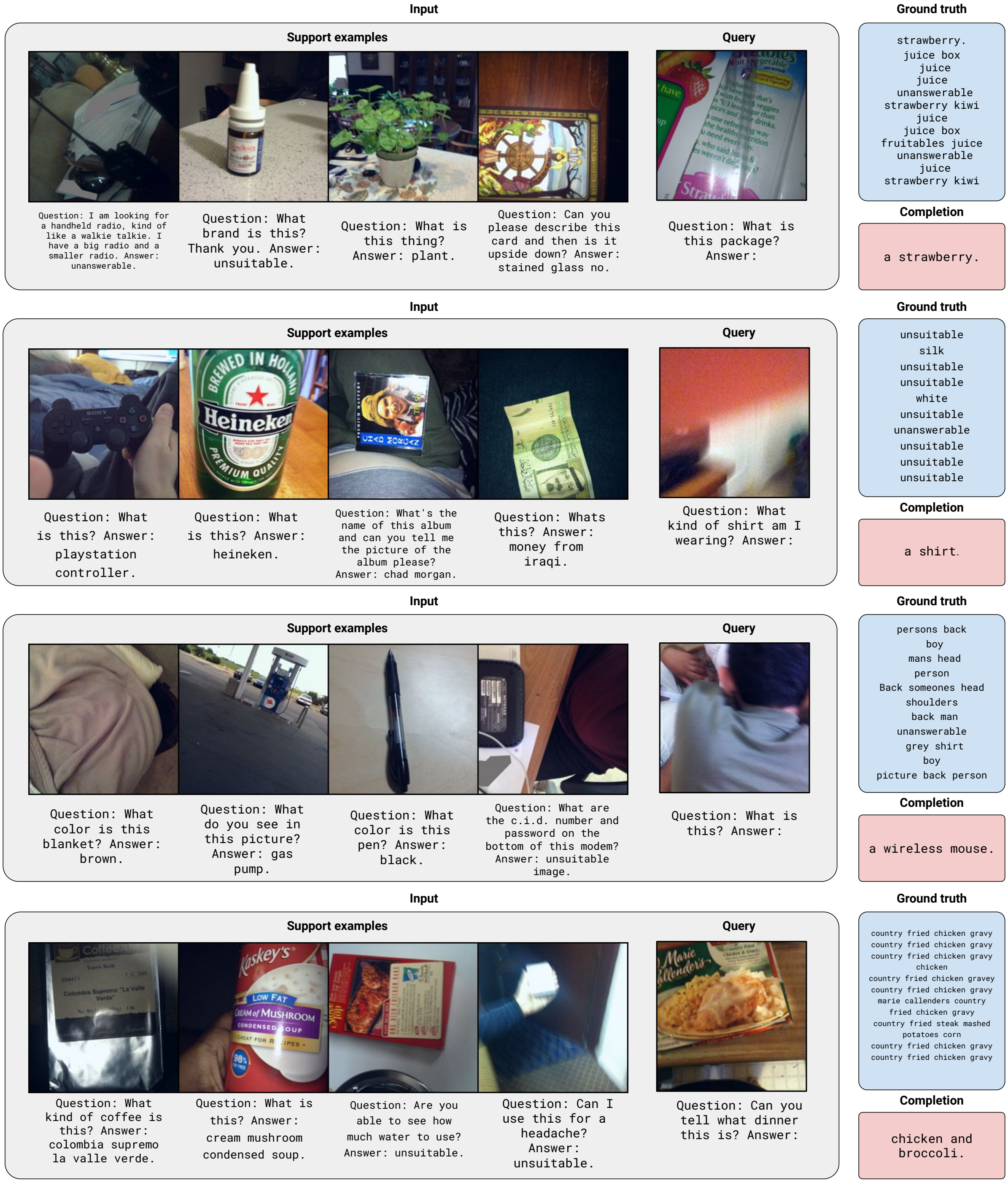


Figure 15: MAPL's 4-shot VQA on VizWiz-VQA, failure cases.

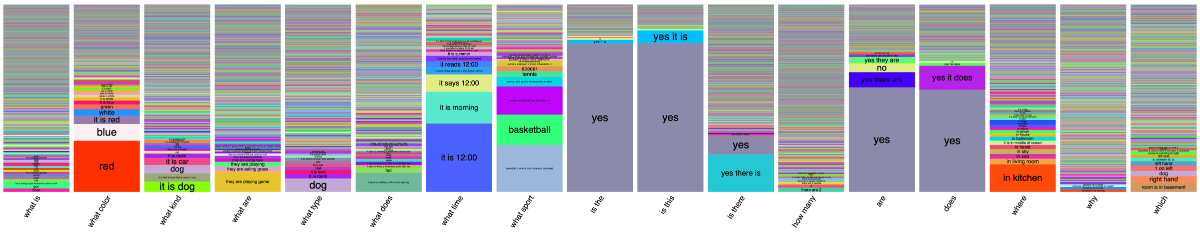


Figure 16: Predicted answer distributions for selected VQAv2 question types with the text-only baseline.

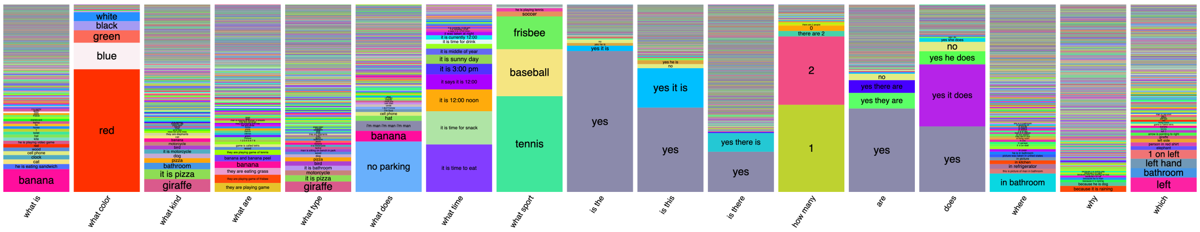


Figure 17: Predicted answer distributions for selected VQAv2 question types with the blind baseline.

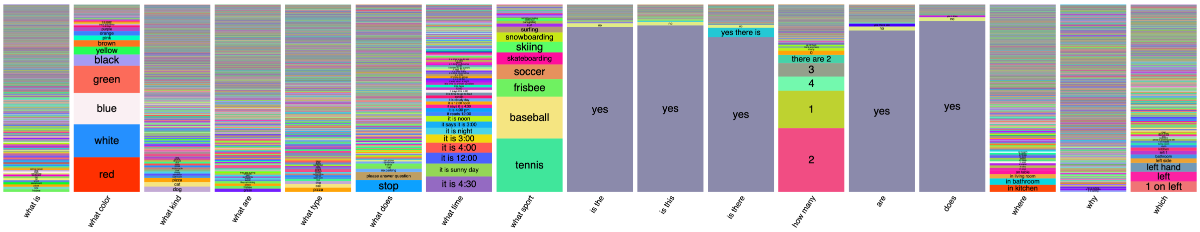


Figure 18: Predicted answer distributions for selected VQAv2 question types with MAPL 0-shot.

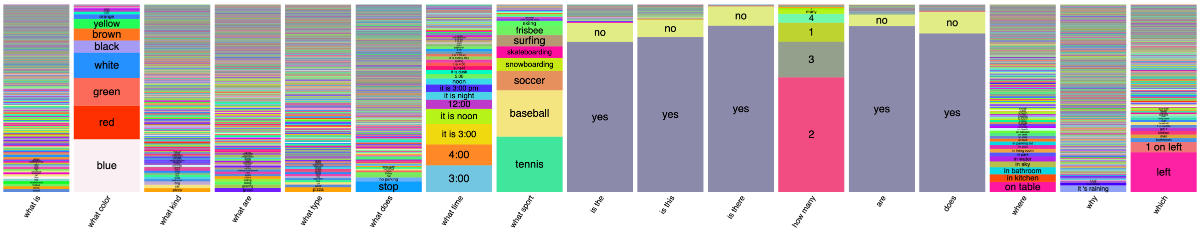


Figure 19: Predicted answer distributions for selected VQAv2 question types with MAPL 4-shot.

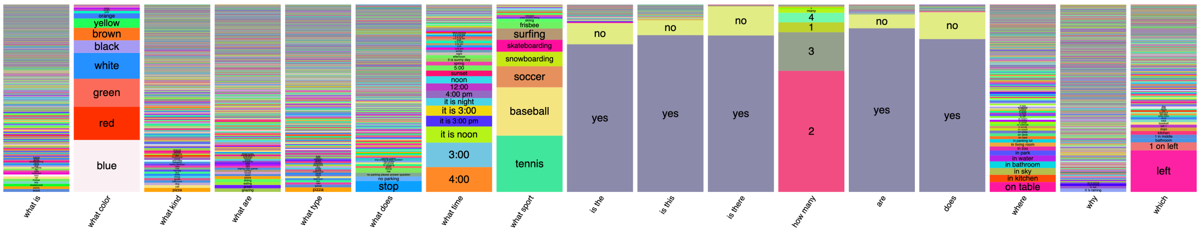


Figure 20: Predicted answer distributions for selected VQAv2 question types with MAPL 8-shot.

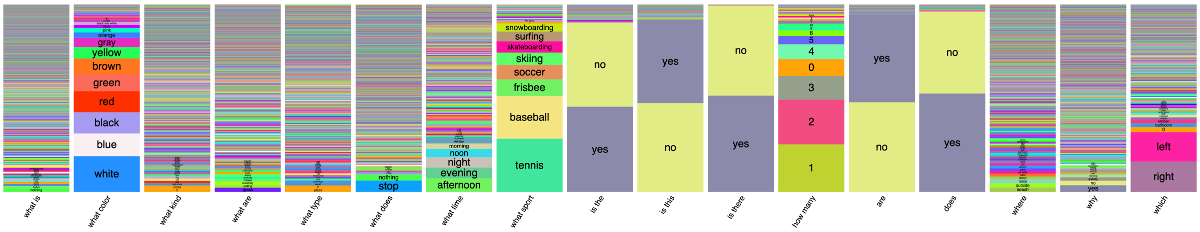


Figure 21: Ground truth answer distributions for selected VQAv2 question types.

ComSearch: Equation Searching with Combinatorial Strategy for Solving Math Word Problems with Weak Supervision

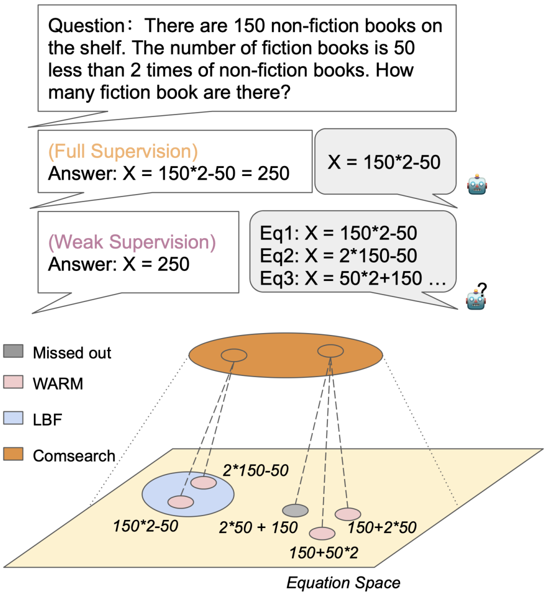
Qianying Liu[1], **Wenyu Guan**[2], **Jianhao Shen**[3], **Fei Cheng**[1] **and Sadao Kurohashi**[1]
[1] Graduate School of Informatics, Kyoto University
[2] Xiaomi AI Lab
[3] Peking University
`ying@nlp.ist.i.kyoto-u.ac.jp; gwy1995@live.com; jhshen@pku.edu.cn;`
`{feicheng,kuro}@nlp.ist.i.kyoto-u.ac.jp;`

Abstract

Previous studies have introduced a weakly-supervised paradigm for solving math word problems requiring only the answer value annotation. While these methods search for correct value equation candidates as pseudo labels, they search among a narrow sub-space of the enormous equation space. To address this problem, we propose a novel search algorithm with combinatorial strategy **ComSearch**, which can compress the search space by excluding mathematically equivalent equations. The compression allows the searching algorithm to enumerate all possible equations and obtain high-quality data. We investigate the noise in the pseudo labels that hold wrong mathematical logic, which we refer to as the *false-matching* problem, and propose a ranking model to denoise the pseudo labels. Our approach holds a flexible framework to utilize two existing supervised math word problem solvers to train pseudo labels, and both achieve state-of-the-art performance in the weak supervision task. [1]

1 Introduction

Solving math word problems (MWPs) is the task of extracting a mathematical solution from problems written in natural language. Based on a sequence-to-sequence (seq2seq) framework that takes in the text descriptions of the MWPs and predicts the answer equation (Wang et al., 2017), task-specialized encoder and decoder architectures (Wang et al., 2018b, 2019; Xie and Sun, 2019; Liu et al., 2019; Guan et al., 2019; Zhang et al., 2020b,a; Shen and Jin, 2020), data augmentation and normalization (Wang et al., 2018a; Liu et al., 2020; Shen et al., 2022b), and pretrained models (Tan et al., 2021; Liang et al., 2021; Shen et al., 2021, 2022a) have been conducted on *full supervision* setting of the task. These settings require equation expression annotation, which is expensive and time-consuming.



Question: There are 150 non-fiction books on the shelf. The number of fiction books is 50 less than 2 times of non-fiction books. How many fiction book are there?

(Full Supervision)
Answer: X = 150*2-50 = 250

X = 150*2-50

(Weak Supervision)
Answer: X = 250

Eq1: X = 150*2-50
Eq2: X = 2*150-50
Eq3: X = 50*2+150 …

Figure 1: Example of MWP solving system under full supervision and weak supervision.

Recently Hong et al. (2021) (LBF) and Chatterjee et al. (2021) (WARM) addressed this problem and proposed the *weak supervision* setting, where only the answer value annotation is given for supervision. Such a setting forms pseudo question-candidate equation pairs, which hold the correct answer value for training with the complexity of $O(n^{2n})$ for n variables enormous possible equation space. Computational efficiently extracting such pairs becomes the major challenge since it is computationally impossible to traverse all possible equations, especially when the example has more variables (e.g., 88,473,600 for 6 variables). As we show in Figure 1, previous studies sample a limited set of equations via random walk (Hong et al., 2021) or beam searching (Chatterjee et al., 2021). However, the algorithms can only cover a limited part of the data, which we refer to as *recall*. As shown in Table 1, LBF (Hong et al., 2021) only covers 30% of the examples of more than 4

---

[1] Our code and data is available at `https://github.com/yiyunya/ComSearch`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2549–2562
May 2-6, 2023 ©2023 Association for Computational Linguistics

Model	≤ 3	≥ 4
LBF	88.1%	30.9%
ComSearch	94.4%	94.5%

Table 1: Searching result recall on problems of different variable sizes.

variables. Moreover, the random walk algorithm lacks robustness and leads to a high performance variance.

We observe that although the equation search space is ample, many equations are mathematically equivalent under the commutative law, associative law, or other equivalent forms. Hence, searching for these equivalent equations is redundant, especially for difficult examples with a larger number of variables. For example, $a + b + c + d * e$ has 48 equivalent forms that hold the same mathematical meaning considering only the commutative law. Eliminating such redundancy in the searching space could reduce computational complexity. In this paper, we propose a combinatorial-strategy-based searching method **ComSearch** that enumerates *non-equivalent equations* without repeating, which can robustly extract candidate equations for a wide range of unlabeled data and build a high recall pseudo data with equation annotation even for difficult examples. To this end, the main idea of Comsearch is to use depth-first search (DFS) to enumerate only one representative equation for each set of equivalent equations and then check whether the equation holds the correct answer value. Comsearch effectively compresses the searching space, e.g., up to 111 times for 6 variables compared to bruce-force searching. As shown in Table 1, ComSearch can achieve a relatively high recall for different variable sizes. Our method could be proven to have lower approximate complexity.

While Comsearch only searches among *non-equivalent equations*, we observe that many examples still have multiple candidate equations through which we can get the final answer. As shown in Figure 1, Equation 1 (Eq1: X=150*2-50) and Equation 3 (Eq3: X=50*2+150) can get the same value, but Equation 3 holds a false mathematical reasoning logic, and using Equation 3 as the pseudo label would bring in noise. We address this data noise as the *false-matching* problem, which has been ignored in previous studies, since their methods do not consider whether the multiple candidate equations of one example are caused by equivalent equation forms or false matching. To address this problem, we investigate how the false-matching problem drags down the system's performance and propose two ranking models to alleviate this problem. For examples with multiple candidate equations from ComSearch, the ranking module first collects a set of candidate equations, then assign a score by a draft model trained on pseudo data with only a single candidate equation to each candidate to choose the best pseudo label. In addition to candidates from the searching result of ComSearch, we observe that beam search results of the draft model can also serve as a high-precision candidate equation. We investigate these two settings for candidate equation sets.

We conduct experiments on two strong MWP solvers, achieving state-of-the-art (SOTA) results under the weakly supervised setting, especially for examples with many variables. The results also demonstrate the effectiveness and generalization ability of our method.

In summary, our contribution is three-fold:

- We propose ComSearch, a searching algorithm that enumerates non-equivalent equations without repeat to search candidate equations effectively.
- We are the first to investigate the *false-matching* problem that brings noise to the pseudo training data. We propose a ranking module to reduce the noise and give a detailed oracle analysis of the problem.
- We perform experiments on two MWP solvers with our ranking module and achieve SOTA performance under weak supervision.

2 Methodology

We show the pipeline of our method in Figure 2. Our method consists of three modules: the Search with combinatorial strategy (**ComSearch**) module that searches for candidate equations; the MWP model that is trained to predict equations given the natural language text and pseudo labels; the Ranking module that uses an explorer model to find candidate equations and select the best candidate equation with a scoring model.

2.1 ComSearch

Directly searching for non-equivalent equation expressions is difficult because the searching method needs to consider Commutative law, Associative law, and other equivalent forms. We show how equivalent equations could be merged into a representative form $\mathcal{X}$, and the enumeration of $\mathcal{X}$ can

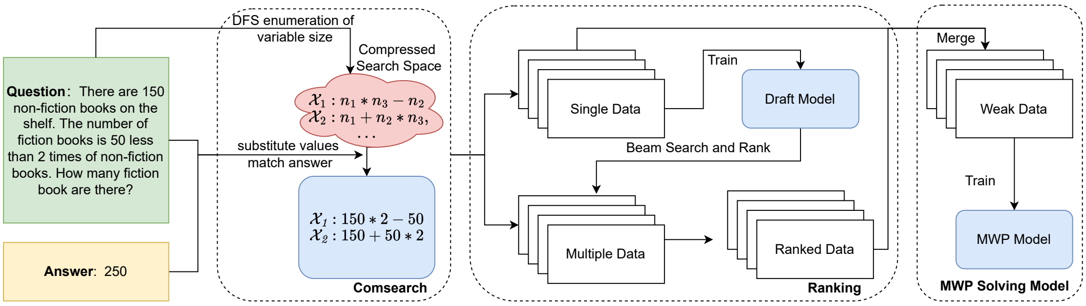


Figure 2: The model overview.

transverse all non-equivalent equations for four arithmetic operations.

We define the set of *non-equivalent equations* using four arithmetic operations as S_n. We first split the equations to two categories, either $S^{\pm}$ where the outermost operators are $\pm$, such as $n_1/n_2 + n_3 - n_4$ and $n_1/n_2 - (n_3 - n_4)$, or $S^{\divideontimes}$ where the outermost operators are $\divideontimes$, such as $(n_2 + n_1) * (n_3 - n_4/n_5)$. We call the former a *general addition equation* and the latter a *general multiplication equation*.

$$S^{\pm}_m = \{(n_1 \divideontimes (..)) \pm (n_i \divideontimes (..)) \pm ..n_m\} \quad (1)$$

$$S^{\divideontimes}_m = \{(n_1 \pm (..)) \divideontimes (n_i \pm (..)) \divideontimes ..n_m\} \quad (2)$$

These two sets are symmetrical, so we only need to consider one set. Consider elements in $S^{\pm}_m$, we can rewrite the equation to the representative form $\mathcal{X}$:

$$\begin{aligned}\mathcal{X} =&((n_i \divideontimes (..)) + (n_j \divideontimes (..)) + ..) \\ &- ((n_k \divideontimes (..)) + (n_l \divideontimes (..)) + ..)\end{aligned}$$

For example, $n_1/n_2 - n_3 + n_4$ and $n_1/n_2 - (n_3 - n_4)$ are equivalent, that they are both rewritten as $(n_1/n_2 + n_4) - n_3$. $(n_2 + n_1) * (n_3 - n_4/n_5)$ could be rewritten as $(n_1 + n_2) * (n_3 - n_4/n_5)$. Trivially, any two equations that are represented by the same $\mathcal{X}$ are equivalent. We give proof of the number of inequivalent expressions involving n operands in Appendix Section A, which shows that any two equivalent equations are written as the same $\mathcal{X}$. Thus the enumeration of $\mathcal{X}$ is equivalent to the enumeration of non-equivalent equations. The enumeration problem of these equations is an expansion of solving Schroeder's fourth problem (Schröder, 1870), which calculates the number of labeled series-reduced rooted trees with m leaves. We give the details of the DFS in the Appendix Section D.

Given the compressed search space, we substitute the values for variables in the equation templates and use the equations of which value matches the answer number as candidate equations. If no equations could be extracted by using all numbers, we continue to consider: (1) omitting one number, (2) adding constant number 1 and π, and (3) using one number twice. If the algorithm extracts candidates at any stage, the further stages are not considered since it would introduce repeating equations, e.g., $1 * (a + b)$ is a duplication of $a + b$.

2.2 MWP Solving Models

Goal-driven Tree-structured Solver We follow Hong et al. (2021) and Chatterjee et al. (2021) and use Goal-Driven Tree-Structured MWP Solver (GTS) (Xie and Sun, 2019) as the MWP model. GTS is a seq2seq model with the attention mechanism that uses a bidirectional long short term memory network (BiLSTM) as the encoder and LSTM as the decoder. GTS also uses a recursive neural network to encode subtrees based on its children nodes' representations with the gate mechanism. With the subtree representations, this model can well use the information of the generated tokens to predict a new token.

Graph-to-Tree Solver Following Chatterjee et al. (2021), we conduct experiments on Graph-to-Tree Solver (G2T) (Zhang et al., 2020b) . G2T is a direct extension of GTS, which consists of a graph-based encoder capturing the relationships and order information among the quantities.

2.3 Ranking

While ComSearch enumerates equations that are non-equivalent without repeat, some variable sets can coincidentally form multiple equations with the same correct value, as shown in Figure 2. The equations $150 * 2 - 50$ and $150 + 50 * 2$ are non-equivalent. However, their values are equal, while only $150 * 2 - 50$ is the correct solution. We refer to this problem as *false-matching*, an important issue

Model	Term	#	Prop(%)
-	All Data	23,162	-
Ours	Too Long	233	1.0
	Power Operator	51	0.2
	Single	17,959	77.5
	Multiple	3,947	17.0
	Data	21,906	**94.5**
WARM	Data (w/o beam)	-	14.5
	Data (w/ beam)	-	80.1
LBF	-	-	80.1

Table 2: Statistics of ComSearch Results.

that previous studies have overlooked. While previous studies also collect multiple candidate equations for one example, they cannot differ whether the issue is caused by equivalent forms of the equations or *false-matching*, and they do not perform any processing on these *false-matching* examples, which brings in noise to the pseudo data.

To process these data that have multiple candidate equations, we propose two ranking methods to choose the best candidate equation for each example. The module first collects a set of candidate equation that holds the correct annotated answer value and then score the candidates to choose the pseudo label for the sample.

Before ranking, we train a draft model S on the single-candidate pseudo data because the single-candidate data is relatively reliable with fewer false-matching examples. In the first ranking method *Basic Ranker*, for a data example x, we rank among the multiple search results of Comsearch $\{y^{eq}\}^{search}$. Then we use the draft model S to calculate the conditional probability of y^{eq} at each time step t. The score of the length k equation s^{eq} is defined as:

$$s^{eq} = \sum_{t=0}^{k} log(S(x, y_t^{eq})) \tag{3}$$

We use the candidate equation that has the highest score as the pseudo label of this example.

Empirically, we observe that performing beam search on the draft model S could also generate high-precision candidate equations. Thus in the second method *Beam Ranker*, we further explore more candidate equations with beam search. We add beam search predictions of S that hold the correct value $\{y_{eq}\}^{beam}$ to the candidate equation set along with Comsearch results $\{y^{eq}\}^{search}$. The score function is defined the same as the basic ranker.

3 Analysis on ComSearch

3.1 Search Statistics

We give statistics of ComSearch in Table 2. Among the 23,162 examples, 233 have more than 6 variables that we filter out, and 51 use the power operation that our method is not applicable. 94.5% of the examples find at least one equation that can match the answer value, significantly higher than WARM and LBF, which cover only 80.1% of the examples. 17,959 examples match with only one equation, and 3,947 examples match with two or more equations that need the ranking module to choose the pseudo label further. We show the distribution of these examples in Appendix Section B.

We further break down the recall on different variable sizes in Table 4. As we can see, when the number of variables grows larger, the recall of LBF drastically collapses, while the recall of our method keeps steady. Sampling based methods cover only a small subset of the equation space and fail to extract candidate equations for larger variable size examples. In contrast, our method can consider a broader range of equation space, which demonstrates the superiority of our enumeration based method.

3.2 Eliminating Equivalent Equations in Search Space

We show the empirical compression of the search space with ComSearch in Table 3. As we can see, the compression ratio of ComSearch increases as the variable number grows, up to more than 100 times when the number of variables reaches 6. Previous studies on reducing the redundancy of equivalent expressions consider a limited set of rules, such as removing brackets (Roy and Roth, 2015) and Commutative Law (Wang et al., 2018a). We also show the results of considering removing brackets, where $-/\div$ can not be the children node of $+/*$, which is the compression considered in Roy and Roth (2015); and Commutative Law, which is the compression considered in Wang et al. (2018a). Although the two methods can compress the search space to some extent, there is a large gap between their compression efficiency and ours, up to more than 20 times when the number of variables reaches 6.

The size of the Bruce-Force search space could be directly calculated, which is $n! * (n-1)! * 4^{n-1}$. If we consider the exponential generating function

#Variable	Bruce-Force	Removing Brackets	Commutative	ComSearch	Ratio
1	1	1	1	1	1
2	8	8	6	6	1.3
3	192	144	108	68	2.8
4	9,216	5,184	3,816	1,170	7.9
5	737,280	311,040	224,640	27,142	27.2
6	88,473,600	27,993,600	19,841,760	793,002	111.6

Table 3: Empirical Results of Search Space Size.

#Var	1	2	3	4	5	≥ 6
LBF	91.5	86.8	88.8	31.1	25.0	38.4
Ours	67.0	93.4	96.4	98.1	94.4	73.8

Table 4: Result of recall on different variable sizes

of $card(S_n)$, based on Smooth Implicit-function Schema, we can have an approximation of S_n: $card(S_n) \sim C * n^{n-1}$, which shows our searching method compresses the search space more than exponential level. We give proof in appendix Section A.

3.2.1 Advantages of Enumeration without repeat

The most important core of our approach is that it explicitly points out the *false-matching* problem because it can enumerate a wide range of equations while ensuring each equation holds an independent mathematical reasoning logic. Sampling methods can only sample a small set of equations that may neglect other potential candidates.

Compared to other enumeration methods, despite the enumeration efficiency, Comsearch ensures the enumeration is among non-equivalent equations, so collecting more than one candidate equation for one example shows that there exists more than one mathematical reasoning logic that could reach the annotated answer value. However, only one of the reasoning logic could be true, which elicits the *false-matching* problem. Even if we add more rules to compress the search space, as long as the non-equivalency of different equations cannot be ensured, we cannot differ *false-matching* and multiple expressions of the same mathematical reasoning logic.

4 Experiments

4.1 Dataset and Baselines

We evaluate our proposed method on the Math23K dataset. It contains 23,161 math word problems annotated with solution expressions and answers. We only use the problems and final answers. We evaluate our method using the train-test split setting of Wang et al. (2018a) by the three-run average.

Model	**Valid(%)**	**Test(%)**
GTS based		
WARM	-	12.8
+*beam*	-	54.3
LBF†	57.2(±0.5)	55.4(±0.5)
+*memory*†	56.6(±6.9)	55.1(±6.2)
Ours†	**61.0**(±0.3)	**60.0**(±0.3)
Supervised†	-	75.6
G2T based		
WARM	-	13.5
+*beam*	-	56.0
Ours†	**61.7**(±1.1)	**60.5**(±0.6)
Supervised†	-	77.4

Table 5: Results on Math23K. ± denotes the variance of 3 runs for valid/test. *Supervised* denotes full supervision upper bound. † denotes the results of our implementation, other results are from the original paper.

We compare our weakly-supervised models' math word problem solving accuracy with two baseline methods.

Chatterjee et al. (2021) proposed **WARM** that uses RL to train the candidate generation model with the reward of whether the value of the equation is correct. Since the reward signal is sparse due to the enormous search space, the top1 accuracy of the candidate generation model is limited, and they use beam search to search for candidates further.

Hong et al. (2021) proposed **LBF**, a learning-by-fix algorithm that searches in neighbour space of the predicted wrong answer by random walk and tries to find a fix equation that holds the correct value as the candidate equation. *memory* saves the candidates of each epoch as training data.

4.2 Main Results and Ablation Study

We show our experimental results in Table 5. We reproduced the results of LBF with their official code

Model	Valid(%)	Test(%)
Proposed Method	61.0	60.0
w/o Multiple Data	58.9	57.5
w/o Ranking	57.3	56.3
w/o Beam Search	60.1	59.2

Table 6: Results of Ablation Study for Ranking. ‘w/o Multiple Data’ denotes only using single candidate pseudo data for training. ‘w/o Ranking’ denotes removing the ranking module and randomly sampling an equation for the examples that match with two or more equations. ‘w/o Beam search’ denotes using the basic ranker for ranking.

Model	Micro Eq Acc(%)
Single	81.4
Multiple	2.7
All Data	23.0
Basic Ranker(Multiple)	45.6
Beam Ranker(Multiple)	47.7
Beam Ranker(All Data)	76.3

Table 7: Equation accuracy of different methods. ‘All Data’ denotes considering both the single and multiple data.

and found that LBF+memory lacks robustness. As we can see in the table, the performance of LBF has high variance on both the validation and test set. For a fair comparison, we additionally ran 5-fold cross-validation setting according to (Hong et al., 2021) for our model and LBF+memory with the GTS model. The results show that LBF + memory achieves a cross-validation score of 56.3% with a variance of ±6.2, while our model achieves a cross-validation score of 59.7% with a variance of ±1.0, which performs similar to the train-test setting. We observe that its performance highly relies on the initialization of the model. When fewer candidates are extracted at early-stage training, the performance drops drastically since LBF relies on random walks in an enormous search space. Our method achieves state-of-the-art performance and outperforms other baselines up to 3.8% and 2.7% on train-test and cross-validation settings. Our method is also more robust with minor variance.

We perform an ablation study with the GTS-based train-test setting in Table 6. *Single Equation* denotes using the 17,959 examples that only match with one equation, the model achieves 57.5% performance, which is slightly lower than using all data and the ranking module, outperforming other baseline models. The result shows that the examples with only one matching could be considered highly reliable and achieve comparable performance with a smaller training data size. We observe a performance drop of at least 2.9% without the ranking module, showing that our ranking module improves the performance. We observe a performance gap of 0.9% between the two rankers, demonstrating the importance of considering candidate equations from the model prediction.

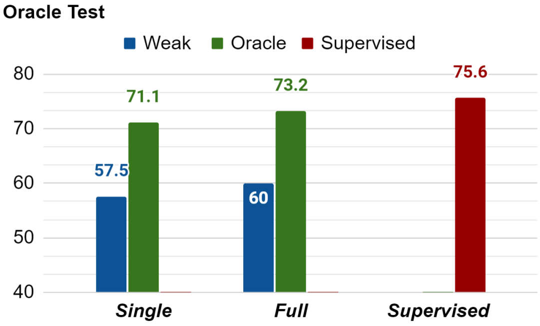


Figure 3: Results of Oracle Test with gold labels.

4.3 Analysis

We conduct analysis on GTS train-test setting since the model achieves similar performance compared with G2T and the run time is less.

4.3.1 Oracle Test

While our searching method covers 94.5% of the training data, as shown in Table 2, there is still a significant performance gap of more than 15% between the weakly supervised performance and fully supervised performance, as shown in Table 5. As stated in Section 2.3, we observe that the *false-matching* problem could potentially draw down the performance, which is verified by the effectiveness of the ranking module.

To further analyze our two modules, we perform two oracle tests for the weakly supervised system. In Figure 3, using the same data examples, we replace the weakly supervised annotations with the supervised gold labels and train the MWP solver. We can observe a performance gap of around 10% using the same data examples as training data, which indicates that the weakly supervised annotations contain noise. Since all candidate equation annotations have the correct answer, the *false-matching* problem is why this noise exists. The results show that the *false-matching* problem is the critical issue in the weakly supervised setting that causes the performance gap compared to supervised setting.

To investigate the noise in the pseudo training data, we perform an oracle analysis of the *Micro*

Text	**Candidates**	**Gold**	**Ans**
Some children are planting trees along a road every 2 meters. They plant trees on both ends of the road. At last they planted 11 trees. How long is the road?	2*(11-1)	(11-1)*2	20
A library has 30 books. On the first day, $\frac{1}{5}$ of the books were borrowed out. On the second day, 5 books were returned. How many book are there in the library now?	30 - $\frac{1}{5}$ * 5	30*(1-($\frac{1}{5}$)) + 5	29
Peter and a few people are standing in a line, one person every 2 meters. Peter found that there are 4 people before him and 5 people after him. How long is this queue?	4*5-2 , (4+5)*2	4*2 + 5*2	18

Table 8: Case study of ComSearch. The blue color denotes that the candidate is true-matching and the light red color denotes that the candidate is false-matching.

	Train				Test		
	Micro Eq Acc(%)		Macro Eq Acc(%)		Ans Acc(%)		
#Var	LBF	Ours	LBF	Ours	LBF	Ours	Prop(%)
1	91.8	**96.3**	**88.2**	64.9	**75.0**	50.0	1.6
2	82.9	**94.8**	78.1	**88.7**	**75.2**	73.4	33.1
3	54.2	**78.9**	57.4	**76.1**	56.2	**62.9**	48.5
4	38.0	**58.0**	13.6	**57.4**	4.8	**25.8**	12.4
5	8.6	**31.1**	4.2	**29.4**	3.2	**16.1**	3.1
≥ 6	5.1	**50.6**	1.2	**38.1**	0	**30.1**	1.3

Table 9: Results of different variable sizes.

Equation Accuracy of the pseudo training data. *Micro Equation Accuracy* is defined by what proportion of training instance holds the correct equation solution, which means the instance is not a *false-matching* example. In Table 7, we show the results of micro equation accuracy of the training data. We check whether the pseudo equation annotations that our system obtains are equivalent to the gold labels for each instance. We can see that even in the **Single** data that can only extract one candidate equation, the micro equation accuracy shows there is still noise in the pseudo training data. We show examples in the case study section to explain this problem. The examples that extract more than one candidate have an equation accuracy rate as low as 2.7%, which makes our ranking system essential. Benefiting from the ranking system, the multiple candidate data can achieve a higher equation accuracy rate. The Beam ranker performs better than the basic ranker considering beam search results.

4.3.2 Case Study

We conduct a case study for ComSearch on three examples to further discuss the strengths and limitations of the method in Table 8. The first example extracts only one candidate equation; although the written expression is different from the gold label, the two equations are equivalent, and the candidate is true-matching. The second example extracts only one candidate equation; the *false-matching* candidate coincidentally equals the correct answer with this set of variable numbers. However, the candidate expression and gold label expression are not equivalent. The algorithm reaches a candidate at the stage of using all numbers and does not further search for candidates that use the constant number 1. The third example extracts two candidate equations, while only $(4 + 5) * 2$ holds the correct mathematical knowledge. The two candidates appear at the same searching stage, and such *false-matching* cannot be avoided by Comsearch, where we need the ranker to help filter out the *false-matching* noise. In this example, the two rankers both select the correct label.

4.3.3 Study on Number of Variables

The distribution of different variable size instances in Math23K dataset is imbalanced, so we further break down the performance of different variable sizes compared with LBF in Table 9. The *Micro Equation Accuracy* shows our method can extract higher quality pseudo data for all variable sizes compared to previous sampling based methods, especially for examples with more variables.

The recall of candidate extraction methods is another important factor that affects performance.

Therefore, in addition to *Micro Equation Accuracy*, we further investigate the *Macro Equation Accuracy* of the two methods, which is defined as equation accuracy on an average of each math word problem. We show that, except for 1 variable, our method has significant advantages over LBF, especially for difficult examples. This demonstrates that our method can effectively extract high equality data of a large quantity. We also show the test answer accuracy of our method and LBF of different variable sizes, which positively correlates with the *Macro Equation Accuracy*. Eliminating equivalent equations allows our method to consider the larger search space, while sampling based methods such as LBF limit to a small neighbour space of the model prediction. When the variable number is small, the in-place random walk of LBF can possibly reach the correct equation, that for the examples with 1 or 2 variables, LBF has a slight performance advantage. When the variable number grows larger, as shown in Table 3, the gap between the efficiency of our searching method and LBF expands, and our method can consider more equations candidates and achieve higher recall and better recall performance. Moreover, the *false-matching* problem is more severe when there are more variables; ignoring the problem would cause low *Micro Equation Accuracy* and bring in more noise to the pseudo training data.

5 Related Work

Early approaches to solving math word problems mainly depend on hand-craft rules and templates (Bobrow, 1964; Charniak, 1969). Later studies either rely on semantic parsing (Roy and Roth, 2018; Shi et al., 2015; Zou and Lu, 2019), or try to obtain an equation template (Kushman et al., 2014; Roy and Roth, 2015; Koncel-Kedziorski et al., 2015; Roy and Roth, 2017). Recent studies focus on using deep learning models to predict the equation template for full supervision setting.

For weakly supervised setting, Hong et al. (2021) and Chatterjee et al. (2021) suffers from two major drawbacks. First, they apply equation candidate searching on an enormous searching space, while our method can effectively extract high-quality candidate equations. Hong et al. (2021) results in low robustness and low performance on examples with more variables. Chatterjee et al. (2021) results in low coverage of examples that can extract candidate equations. Second, they use all candidate equations for training and neglect the *false-matching* problem, which is the key issue that drags down the model performance in weakly supervised setting, while our ranking module addresses this issue and further boosts the performance.

To eliminate equivalent expressions, Roy and Roth (2015) proposed a model that decomposes the equation prediction problem into various classification problems, eliminating some equivalence forms of the equation. However, the compression is highly integrated with their model and cannot generalize to other models, including the SOTA seq2seq based models. Moreover, it can only cover limited equivalence forms, leaving out various important forms such as Commutative law and Associative law. (Wang et al., 2018a) proposed a normalization method for supervised MWP systems that considers Commutative law. The method merges several equivalent expressions into one expression, resulting in the compression of the target equation space. However, their method requires bruce-force enumeration before compression, which remains to have high computational complexity. Only limited equivalent forms are considered in both studies, and the equation space is still considerably ample.

Various studies (Kristianto et al., 2016; Mansouri et al., 2021) in ARQMath competition (Mansouri et al., 2020) and NTCIR benchmark (Zanibbi et al., 2016) have investigated the math retrieval task that retrieves the most related mathematical passage for a question, which have clear semantic meanings given by the textual description. In our ranker setting, the scoring targets, i.e., plane mathematical equations, cannot provide the semantic meanings that contextual embedding similarity based methods used in math retrieval benchmarks require. With fully supervised training data, retrieval-based methods only achieve 40% accuracy (Wang et al., 2017) on Math23K.

Spurious programs in weakly supervised semantic parsing is a close analogy of the false-matching problem, which refers to incorrect programs that lead to correct denotations. The major difference is that the function names of the spurious programs are natural language defined, so the programs have semantic meanings. Extra knowledge bases (Berant et al., 2013) and lexicon clues (Goldman et al., 2018) were used to denoise the spurious programs, which is not applicable for complex lexicon patterns MWPs that the solution equation uses operators '$+, -, *, /$' that have no semantic meaning. Pasupat and Liang (2016) uses a small human-annotated dataset for denoising. Guu et al. (2017), which proposes a re-weighted optimization loss for the examples. However, their method relies heavily on hyperparameter tuning and gains negative results on many datasets. Thus these methods are not suitable for the setting in our paper.

6 Conclusion

This paper proposes ComSearch, a searching method based on a Combinatorial strategy, to extract candidate equations for Solving Math Word Problems under weak supervision. ComSearch compresses the enormous search space of equations beyond the exponential level, allowing the algorithm to enumerate all possible non-equivalent equations to search for candidate equations. We investigate the *false-matching* problem, which is the critical issue that drags down performance, and propose a ranking model to reduce noise. Our experiments show that our method obtains high-quality pseudo data for training and achieves state-of-the-art performance under weak supervision settings, outperforming strong baselines, especially for examples with more variables.

Limitations

As we observe from experiments, the performance gap between the most reliable weak data and oracle data is still 10%, and the noise rate in the pseudo data is still relatively high. This is caused by the stopping strategy of our searching algorithm. Because introducing constant numbers such as 1 and using variables for more than one time would cause meaningless multiple candidate equations (e.g., $n_1/n_1 * n_1$, $1 * n_1$), we search the equations at various stages: deleting one variable, adding a constant and using one variable multiple times. We stop searching when the stage ends and one equation is obtained. If a more advanced searching strategy that can consider such redundancy could be introduced, the reliability of the weak data could be further boosted.

Meanwhile, our ranking module only denoises multiple candidate equations examples, while the single data also has a volume of noise. We denoise with a simple strategy for one round because we focus on investigating the negative effects of the false-matching problem. For future work, we would consider applying more advanced learning from noise algorithms and denoise more training data.

In Table 4, the results shows a notable discrepancy in the performance of the #var = 1 when compared to other variable sizes and the baseline. This discrepancy can primarily be attributed to numerous geometrically related questions in the #var = 1 example set, such as the computation of the volume of a cube l^3 given the side length l, which is not encompassed by our current search methodology. A straightforward remedy would be to include this equation template in our search when handling #var = 1; however, we deliberately excluded it from our experiments to maintain consistency across the different variable sizes.

Acknowledgements

This work is partially supported by JST SPRING Grant No.JPMJSP2110 and KAKENHI No. 21H00308, Japan.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Daniel G. Bobrow. 1964. Natural language input for a computer problem solving system. Technical report, Cambridge, MA, USA.

L. Carlitz and J. Riordan. 1956. The number of labeled two-terminal series-parallel networks. *Duke Mathematical Journal*, 23(3):435 – 445.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Eugene Charniak. 1969. Computer solution of calculus word problems. In *Proceedings of the 1st International Joint Conference on Artificial Intelligence*, IJCAI'69, pages 303–316, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Oishik Chatterjee, Aashish Waikar, Vishwajeet Kumar, Ganesh Ramakrishnan, and Kavi Arya. 2021. A weakly supervised model for solving math word problems.

James W. Cooley and John W. Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Philippe Flajolet and Robert Sedgewick. 2009. *Analytic Combinatorics*. Cambridge University Press.

Omer Goldman, Veronica Latcinnik, Ehud Nave, Amir Globerson, and Jonathan Berant. 2018. Weakly supervised semantic parsing with abstract examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1809–1819, Melbourne, Australia. Association for Computational Linguistics.

Wenyv Guan, Qianying Liu, Guangzhi Han, Bin Wang, and Sujian Li. 2019. An improved coarse-to-fine method for solving generation tasks. In *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*, pages 178–185, Sydney, Australia. Australasian Language Technology Association.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Kelvin Guu, Panupong Pasupat, Evan Liu, and Percy Liang. 2017. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1051–1062, Vancouver, Canada. Association for Computational Linguistics.

Yining Hong, Qing Li, Daniel Ciao, Siyuan Huang, and Song-Chun Zhu. 2021. Learning by fixing: Solving math word problems with weak supervision. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):4959–4967.

W. Knödel. 1951. Über zerfällungen. *Monatshefte für Mathematik*, 55:20–27.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Giovanni Yoko Kristianto, Goran Topic, and Akiko Aizawa. 2016. Mcat math retrieval system for ntcir-12 mathir task. In *NTCIR*.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281.

Yihuai Lan, Lei Wang, Qiyuan Zhang, Yunshi Lan, Bing Tian Dai, Yan Wang, Dongxiang Zhang, and Ee-Peng Lim. 2021. Mwptoolkit: An open-source framework for deep learning-based math word problem solvers.

Jierui Li, Lei Wang, Jipeng Zhang, Yan Wang, Bing Tian Dai, and Dongxiang Zhang. 2019. Modeling intra-relation in math word problems with different functional multi-head attentions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6162–6167.

Zhenwen Liang, Jipeng Zhang, Jie Shao, and Xiangliang Zhang. 2021. Mwp-bert: A strong baseline for math word problems.

Qianying Liu, Wenyu Guan, Sujian Li, Fei Cheng, Daisuke Kawahara, and Sadao Kurohashi. 2020. Reverse operation based data augmentation for solving math word problems. *arXiv preprint arXiv:2010.01556*.

Qianying Liu, Wenyv Guan, Sujian Li, and Daisuke Kawahara. 2019. Tree-structured decoding for solving math word problems. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2370–2379, Hong Kong, China. Association for Computational Linguistics.

Behrooz Mansouri, Anurag Agarwal, Douglas Oard, and Richard Zanibbi. 2020. Finding old answers to new math questions: the arqmath lab at clef 2020. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II 42*, pages 564–571. Springer.

Behrooz Mansouri, Douglas W Oard, and Richard Zanibbi. 2021. Dprl systems in the clef 2022 arqmath lab: Introducing mathamr for math-aware search. *Proceedings of the Working Notes of CLEF 2022*, pages 5–8.

OEIS Foundation Inc. 2023. The On-Line Encyclopedia of Integer Sequences. Published electronically at `http://oeis.org`.

Panupong Pasupat and Percy Liang. 2016. Inferring logical forms from denotations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23–32, Berlin, Germany. Association for Computational Linguistics.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

John Riordan and Claude E Shannon. 1942. The number of two-terminal series-parallel networks. *Journal of Mathematics and Physics*, 21(1-4):83–93.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Subhro Roy and Dan Roth. 2017. Unit dependency graph and its application to arithmetic word problem solving. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Subhro Roy and Dan Roth. 2018. Mapping to declarative knowledge for word problem solving. *Transactions of the Association of Computational Linguistics*, 6:159–172.

Ernst Schröder. 1870. Vier combinatorische probleme. *Zeitschrift für Mathematik und Physik*, 15.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2269–2279.

Yibin Shen and Cheqing Jin. 2020. Solving math word problems with multi-encoders and multi-decoders. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2924–2934, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yibin Shen, Qianying Liu, Zhuoyuan Mao, Fei Cheng, and Sadao Kurohashi. 2022a. Textual enhanced contrastive learning for solving math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4297–4307, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yibin Shen, Qianying Liu, Zhuoyuan Mao, Zhen Wan, Fei Cheng, and Sadao Kurohashi. 2022b. Seeking diverse reasoning logic: Controlled equation expression generation for solving math word problems. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 254–260, Online only. Association for Computational Linguistics.

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1132–1142.

Minghuan Tan, Lei Wang, Lingxiao Jiang, and Jing Jiang. 2021. Investigating math word problems using pretrained multilingual language models.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018a. Translating a math word problem to a expression tree. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1064–1069.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018b. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Lei Wang, Dongxiang Zhang, Jipeng Zhang, Xing Xu, Lianli Gao, Bing Tian Dai, and Heng Tao Shen. 2019. Template-based math word problem solvers with recursive neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7144–7151.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854, Copenhagen, Denmark. Association for Computational Linguistics.

Yun Wang. 2021. *The Math You Never Thought Of.* Posts & Telecom Press Co., Ltd., Beijing.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5299–5305.

Richard Zanibbi, Akiko Aizawa, Michael Kohlhase, Iadh Ounis, Goran Topic, and Kenny Davila. 2016. Ntcir-12 mathir task overview. In *NTCIR*.

Dongxiang Zhang, Lei Wang, Luming Zhang, Bing Tian Dai, and Heng Tao Shen. 2019. The gap of semantic parsing: A survey on automatic math word problem solvers. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2287–2305.

Jipeng Zhang, Roy Ka-Wei Lee, Ee-Peng Lim, Wei Qin, Lei Wang, Jie Shao, and Qianru Sun. 2020a. Teacher-student networks with multiple decoders for solving math word problem. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4011–4017. International Joint Conferences on Artificial Intelligence Organization. Main track.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020b. Graph-to-tree learning for solving math word problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937.

Yanyan Zou and Wei Lu. 2019. Text2math: End-to-end parsing text into math expressions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5330–5340.

A Proof for Search Space Approximation

Because there is at least one $+$ or $*$ operator for each equation (i.e. $-a-b-c$ is illegal), the target S_n is not symmetric and is hard to directly approximate. We need two assisting targets to form the approximate. This proof majorly relies on Flajolet and Sedgewick (2009) and OEIS Foundation Inc. (2023, A140606).

We first consider target U that considers only $+$, $*$ and $\div$ three operators. We sort it into two categories: U^+ that the outermost operator is $+$ and $U^{\divideontimes}$ that the outermost operator is $\divideontimes$. Equations such as $\frac{1}{a} * \frac{1}{b-c}$ are still considered illegal.

Z corresponds to a single variable equation. We can have the construction of U:

$$U^+ = Z + SET_{\geq}(U^{\divideontimes}) \tag{4}$$

$$U^{\divideontimes} = Z + (2^2-1) * SET_{=2}(U^+) \tag{5}$$

$$+ (2^3-1) * SET_{=3}(U^+)... \tag{6}$$

We apply symbolic method to obtain the EGF of the constructions:

$$U^+(z) = z + \sum_{k\geq 2} \frac{1}{k!}[U^{\divideontimes}(z)]^k \tag{7}$$

$$= z + [e^{U^{\divideontimes}(z)} - 1 - U^{\divideontimes}(z)] \tag{8}$$

$$U^{\divideontimes}(z) = z + \sum_{k\geq 2} \frac{2^k-1}{k!}[U^+(z)]^k \tag{9}$$

$$= z + e^{2U^+(z)} - e^{U^+(z)} - U^+(z) \tag{10}$$

Meanwhile, we have:

$$U(z) = U^+(z) + U^{\divideontimes}(z) - z \tag{11}$$

Next, we consider target T that $-a-b-c$ is considered legal. Similarly we define $T^{\pm}$ and $T^{\divideontimes}$. We consider the construction:

$$T^{\pm} = 2Z + SET_{\geq}(T^{\divideontimes}) \tag{12}$$

$$T^{\divideontimes} = 2Z + 2[(2^2-1) * SET_{=2}(T^{\pm}/2) \tag{13}$$

$$+ (2^3-1) * SET_{=3}(T^{\pm}/2)...] \tag{14}$$

With symbolic method we have:

$$T^{\pm}(z) = 2z + \sum_{k\geq 2} \frac{1}{k!}[U^{\divideontimes}(z)]^k \tag{15}$$

$$= 2z + [e^{T^{\divideontimes}(z)} - 1 - T^{\divideontimes}(z)] \tag{16}$$

$$T^{\divideontimes}(z) = 2z + 2\sum_{k\geq 2} \frac{2^k-1}{k!}[T^{\pm}(z)/2]^k \tag{17}$$

$$= 2z + 2e^{T^{\pm}(z)} - 2e^{T^{\pm}(z)/2} - T^{\pm}(z) \tag{18}$$

The illegal equations such as $-a-b-c$ in T equals the counts of $a+b+c$, which is actually U. So we have:

$$S(z) = T(z) - U(z) \tag{19}$$

We now have the EGF of S_n. We can sequentially compute the first few terms of this sequence:

$$1, 6, 68, 1170, 27142, 793002, 27914126, ... \tag{20}$$

With Smooth implicit-function schema and Stirling approximation function we have, for an EGF $y(z) = \sum_{n\geq 0} y_n z^n$, Let $G(z,w) = \sum_{m,n\geq 0} g_{m,n} z^m w^n$, thus $y(z) = G(z, y(z))$:

$$n! * [z^n]y(z) \sim \frac{c * n!}{\sqrt{2\pi n^3}} * r^{-n+1/2} \tag{21}$$

$$\sim \frac{c\sqrt{2\pi nr}}{\sqrt{2\pi n^3}}(\frac{1}{r})^n(\frac{n}{e})^n \tag{22}$$

$$= \frac{c\sqrt{r}}{n}(\frac{n}{re})^n \tag{23}$$

while r:

$$G(r,s) = s \tag{24}$$

$$\frac{\partial G(r,s)}{\partial w} = 1 \tag{25}$$

and c:

$$c = \sqrt{\frac{\partial G(r,s)/\partial z}{\partial^2 G(r,s)/\partial w^2}} \tag{26}$$

We still need the two assisting targets to perform the approximation. We have:

$$U^+(z) = e^{z+e^{2U^+(z)}-e^{U^+(z)}-U^+(z)} \tag{27}$$

$$- e^{2U^+(z)} + e^{U^+(z)} + U^+(z) - 1 \tag{28}$$

Let $G(z,w) = z + e^{2w} - e^w - ln(1 + e^{2w} - e^w)$, considering 24 and 26, r, s and c would be constant numbers.

So we have:

$$n![z^n]U^+(z) \sim \frac{c_1\sqrt{r_1}}{n}(\frac{n}{r_1 e})^n \tag{29}$$

Similarly we can approximate $U^{\divideontimes}$, $T^{\pm}$ and $T^{\divideontimes}$:

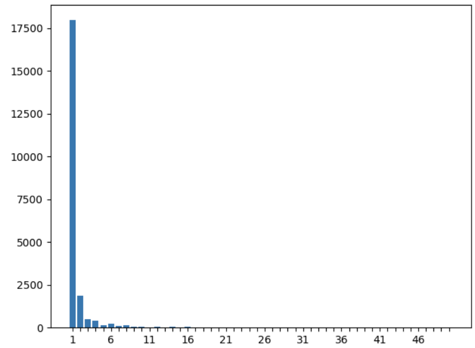


Figure 4: Distribution of Candidate Equation Number.

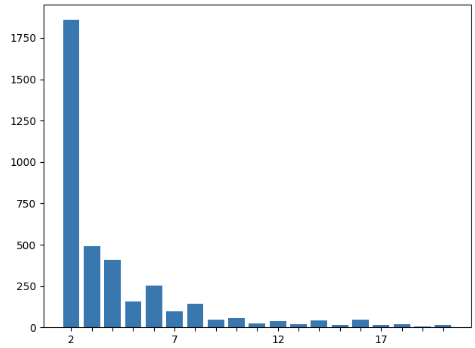


Figure 5: Distribution of Candidate Equation Number.

$$n![z^n]U^{\divideontimes}(z) \sim \frac{c_2\sqrt{r_1}}{n}(\frac{n}{r_2 e})^n \tag{30}$$

$$n![z^n]T^{\pm}(z) \sim \frac{c_3\sqrt{r_2}}{n}(\frac{n}{r_3 e})^n \tag{31}$$

$$n![z^n]T^{\divideontimes}(z) \sim \frac{c_4\sqrt{r_2}}{n}(\frac{n}{r_4 e})^n \tag{32}$$

So we have:

$$u_n = n![z^n]U(z) \sim \frac{(c_1+c_2)\sqrt{r_1}}{n}(\frac{n}{r_1 e})^n \tag{33}$$

$$t_n = n![z^n]T(z) \sim \frac{(c_3+c_4)\sqrt{r_2}}{n}(\frac{n}{r_2 e})^n \tag{34}$$

Since $S(z) = T(z) - U(z)$, the subtraction of u_n and t_n would be our approximation. However we observe that $r_1 \gg r_3$, that u_n can be ignored. So we have:

$$s_n = n![z^n]S(z) \sim \frac{(c_3+c_4)\sqrt{r_2}}{n}(\frac{n}{r_2 e})^n \tag{35}$$

Q.E.D.

B Distribution of Candidate Equations

The largest candidate equation number of one example is 3914. We show the distribution of candidate equations in Figure 4 and 5. The x-axis represents the number of candidates, while the y-axis represents the number of examples that have x candidate equations. We can see from Figure 4, which includes examples that have 1 to 50 candidates, it is a long tail distribution that most examples only have a few candidate equations. From Figure 5, where we zoom in and focus on examples that have 2 to 20 candidates, we can see that there are a lot of examples that have more than 2 candidate equations, and the ranking module is essential.

Algorithm 1 *enum_skel(n)*

```
Require: $n \geq 1$
  Initialize empty list skills
  for $i \leq n;\ i = 1;\ i++$ do
      left_list = unit_skel(i)
      right_list = enum_skels($n - i$)
      for left in left_list do
          for right in right_list do
              move the start index of right to $i$
              new_skels += left + right
          end for
      end for
      skels += new_skels
  end for
  return skels
```

C Experimental Details

We run our experiments on single card GTX3090Ti, each run takes around 2-3 hours for all models. We did not perform extra hyperparameter searching and use the same hyperparameters as the public release of the two models, except for epoch number which is decided by the validation set. The code is conducted based on Pytorch.

D ComSearch Details

Considering elements in $S_n^{\pm}$, we can rewrite the equation to x. Thus we can form a mapping $g : x \to g(x)$ from a general addition equation x to a skeleton structure expression $g(x)$. :

$$\begin{aligned} x =& ((x_i \divideontimes (..)) + (x_j \divideontimes (..)) + ..) \\ & - ((x_k \divideontimes (..)) + (x_l \divideontimes (..)) + ..) \\ g(x) =& (x_i(..))(x_j(..))..\&(x_k(..))(x_l(..)).. \end{aligned}$$

The order of x_i within the same layer of brackets is ignored in $g(x)$, it can deal with the equivalence caused by Commutative law and Associative law. The addition and subtraction terms are split by &, that which can deal with equivalence caused by removing brackets. $g(x)$ is a bijection, so the enumeration problem transforms to finding such skeletons:

$$
\begin{aligned}
n = 1 &: a \\
g^{-1} &: a \\
n = 2 &: ab, a\&b, b\&a \\
g^{-1} &: a + b, a - b, b - a \\
n = 3 &: abc, a\&(b\&c), (ab)\&c, ... \\
g^{-1} &: a + b + c, a - (b/c), (a * b) - c, ... \\
& ...
\end{aligned}
$$

The enumeration problem of these structures is an expansion of solving Schroeder's fourth problem (Schröder, 1870), which calculates the number of labeled series-reduced rooted trees with n leaves. We use a deep-first search algorithm shown in Algorithm 1 to enumerate these skeletons. It considers the position of the first bracket and then recursively finds all possible skeletons of sub-sequences of the variable sequence $\mathcal{X} = \{x_k\}_{k=1}^{i}$ (Wang, 2021).

While considering such skeletons could enumerate all unique expressions, equations have at least one element on the left of $\&$ in our target domain and do not start with $-$ or $\div$. We further extend the algorithm to consider these cases. To be noticed, because there is at least one $+$ or $*$ operator for each equation, the left side of $\&$ must not be empty while the right part has no restrictions. Thus we define the *unit_skel(i)* equation to return possible skeletons with only one or none $\&$ and no brackets. This constraint is equivalent to finding non-empty subsets and their complement of the variable sequence $\mathcal{X}$. We can use Algorithm 1 to perform the enumeration of such skeletons, except for defining two different *unit_skel(i)* to support the enumeration of subtraction and division operation. The enumeration algorithm of non-empty subsets is trivial and omitted here.

$$unit\_skel_{div}(i) = \{(A\&\overline{A}) | A \subseteq \mathcal{X}; A \neq \emptyset\} \tag{36}$$

$$\begin{aligned} &unit\_skel_{sub}(i) = \\ &\quad \{((a(A - a))\&\overline{A - a}) | A \subseteq \mathcal{X}; a \in A\} \end{aligned} \tag{37}$$

We transform the skeletons back to equations to obtain all non-equivalent equations S_n. Such enumeration considers absolute values and omits pairs of solutions that are opposite to each other. To search effectively, for the equations that contain subtraction, we add their opposite equation to the searching space.

Towards preserving word order importance through FORCED INVALIDATION

Hadeel Al-Negheimish[1], **Pranava Madhyastha**[2,1] and **Alessandra Russo**[1]
[1] Imperial College London [2] City, University of London
{halnegheimish,a.russo}@imperial.ac.uk,
pranava.madhyastha@city.ac.uk

Abstract

Large pre-trained language models such as BERT have been widely used as a framework for natural language understanding (NLU) tasks. However, recent findings have revealed that pre-trained language models are insensitive to word order. The performance on NLU tasks remains unchanged even after randomly permuting the word of a sentence, where crucial syntactic information is destroyed. To help preserve the importance of word order, we propose a simple approach called FORCED INVALIDATION (FI): forcing the model to identify permuted sequences as invalid samples. We perform an extensive evaluation of our approach on various English NLU and QA based tasks over BERT-based and attention-based models over word embeddings. Our experiments demonstrate that FI significantly improves the sensitivity of the models to word order.[1]

1 Introduction

Ordering of words in a sentence is an important structural attribute for natural languages such as English, where subject-verb-object (SVO) structure is common and important to convey the meaning of the sentence. Understanding and comprehending natural language without strict adherence to a systematic ordering of words would make it an extremely challenging task. Recent work has investigated a surprising lack of sensitivity to word order information in state-of-the-art masked language models.

Recent research has focused on the impact of word order perturbation during the evaluation of models trained on well-ordered data. The results show that masked language models exhibit a catastrophic lack of sensitivity to word order permutations or shuffles, even for complex tasks in which task-relevant syntactic properties are completely destroyed (Pham et al., 2020; Al-Negheimish et al., 2021; Sinha et al., 2021b; Gupta et al., 2021). These studies show that models are still predicting the gold label for examples even after sequences have been permuted, and they also do so with high confidence (Sinha et al., 2021b; Gupta et al., 2021). This anomalous behaviour can potentially result in undesirable shortcuts or can cause models to fail catastrophically in simple adversarial settings. Furthermore, Sinha et al. (2021a) study the effect of pre-training masked language models on shuffled data, and suggest that the model might simply be capturing higher-order word co-occurrence statistics, rather than uncovering sophisticated semantic and syntactic structures necessary for language understanding.

In this paper, we present a simple, yet general approach called FORCED INVALIDATION (FI), where we force models to explicitly identify sequence permutations (§3). While our proposal is extensible to multiple types of models, here we present a controlled study over masked language model-based BERT models, and attention-based models over word-embeddings (§4). We present a large battery of experiments over a variety of natural language understanding tasks in the English language, including complex question answering based tasks, natural language inference based tasks and commonsense reasoning based tasks. Results show that our proposal significantly improves the sensitivity of the models to word-order information (§5).

2 Related Work

Recent work has proposed a few mitigation strategies for classification-type tasks: Pham et al. (2020) proposes improving word-order sensitivity by including a precursor fine-tuning step on synthetic CoLA-like tasks before finetuning for downstream tasks. While this improves their defined word order sensitivity score, accuracy on permuted samples remains significantly above chance, making this

---

[1] Our code and data for replication are available at https://github.com/halnegheimish/ForcedInvalidation

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2563–2570
May 2-6, 2023 ©2023 Association for Computational Linguistics

approach unreliable. Gupta et al. (2021) present three approaches: one based on entropy regularisation, another on model probabilities thresholding, and finally an approach based on augmenting additional data consisting of destructive transformations, which include 1-gram permutations. The model is modified with an additional class to identify these destructive transformations. While the first two approaches require changes to model setup, the last one is based on a set of manual heuristics. Our approach has some similarity with the latter, however, our permutations are based on the principals borrowed from the n-gram language modelling literature (Roark et al., 2007), where the n-grams capture sufficient first-order statistics of the language. Our approach significantly reduces the models' reliance only on simple first-order statistics of language, and our empirical observations demonstrate the generalisability of our approach to various settings.

3 Methodology: FORCED INVALIDATION

Our method is grounded on recent observations which show that masked language models and other similar models tend to exploit shortcuts based on information about distributional word vicinity information for a diverse set of natural language processing tasks (Al-Negheimish et al., 2021; Sinha et al., 2021a). These shortcuts tend to make the models less sensitive to word-order information even for tasks that require the preservation of word ordering for accurate recovery of meaning (Sinha et al., 2021b). Al-Negheimish et al. (2021), in particular, show the insensitivity on a variety of n-gram based permutations of samples, where the authors specifically experiment with $\{1,2,3\}$-gram permutations. Further, $\{1,2,3\}$-gram based permutations capture a variety of word vicinity based patterns and also capture some of the most frequently occurring unigram, bigram and trigram patterns in a variety of benchmark datasets. Based on these salient observations, for each given dataset, the FORCED INVALIDATION (FI) methodology consists of the following two steps:

1. Augmenting training data with $\{1,2,3\}$-gram permutation samples (sampled from trainset) labelled with *invalid* as the additional label.
2. Modifying models to account for the new label and training them in the standard setting with a combination of standard training examples and the augmented *invalid* samples.

We observe that this simple FI approach improves the sensitivity of the model to word order and also improves the robustness of the models across a variety of tasks to first-order shortcuts. In the following sections, we present a rigorous experimental study that showcases the utility of FI.

4 Experimental settings

Data To generate n-gram permutations, we simply subsample from the training dataset, such that the ratio to the valid samples (samples with correct word order and the task-specific label) and invalid samples (samples with permuted n-grams and the *invalid* label) is 1-1. The invalid samples are generated such that they contain a uniform distribution of $\{1,2,3\}$-gram permutations.[2] Furthermore, we split the training set such that we use 90% of the samples for training the models and the remaining 10% is used as a development set. The development set is used to monitor training and perform early-stopping. Evaluations are done on a separate unseen task-specific validation set provided by the dataset creators.

We perform two evaluations: well-ordered and permuted. The first one is the standard evaluation of the model over the original unperturbed task-specific unseen validation set. For our experimental evaluations over permutations, we retain the original label of the same unseen validation set, but we permute the specific components of samples (e.g., premise or the hypothesis, question or the passage, etc.); we expect the models to reject the permuted sample instead of predicting the same ground-truth label. This setup allows us to evaluate the sensitivity of the model to word-order permutations of various degrees over the different components of the data.

Models We predominantly experiment with BERT-based models that either use BERT representations as the contextualised embeddings or classifiers that are directly trained with BERT. We also experiment with an additional simpler model that largely exploits attention over word-embeddings.[3] We will expand on the specific models for each of the tasks in the following section. Results for FI are

---

[2] Input string is divided to n-grams (based on white-space), and permuted, preserving the final punctuation. The only condition is that it varies from the original string, which is a weaker constraint than previous studies that require that no n-gram stays in its original position.

[3] Details about training parameters can be found in the appendix A

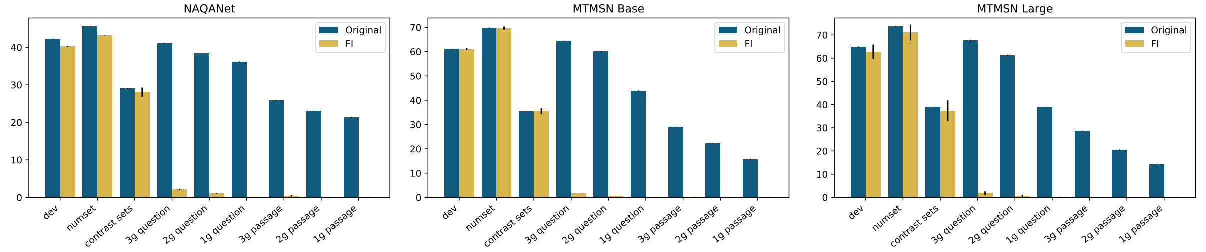


Figure 1: Accuracy using exact match (EM) on the DROP QA dataset, (a) NAQANet, (b) MTMSN Base, and (c) MTMSN Large. dev, numset and contrast sets are unperturbed well-ordered datasets, while the other bars show {1,2,3}-gram permutations on numset. FI models exhibit clear sensitivity to word order, as they no longer predict the original answer in most perturbed cases. FI accuracy is averaged over training with 3-random seeds, and the standard deviation is shown over the yellow bars. FI-models are confident in identifying invalid samples.

reported over the averages and standard deviation of models trained with three random seeds.

5 Results and Observations

5.1 Unconstrained Question Answering

DROP (Dua et al., 2019) is a reading-comprehension dataset with unconstrained answers. It comprises questions that require reasoning over the content of different paragraphs. Even though this is designed to be a challenging task, Al-Negheimish et al. (2021) show that for most models, permuting questions had little impact on the model's ability to predict the correct answer for numerical reasoning questions. This was found specifically problematic as the task consists of complex questions, and permutations render the questions syntactically and semantically redundant. We apply FI for two module-based models designed specifically for this task, NAQANet (Dua et al., 2019), an attention-based model with GloVe-based embeddings (Pennington et al., 2014), which is the original model proposed by the authors of the dataset; and MTMSN (Hu et al., 2019), which is based on BERT-based (Devlin et al., 2019) contextual representations. These models have separate modules targeting different kinds of reasoning, e.g. a module for counting and a module for arithmetic expressions. We augment these with an additional module to force invalidation over invalid permuted samples, a two-way classification module that learns to distinguish between permuting either the question or passage. FI models should now choose the invalid type for permuted samples, instead of giving the same answer as well-ordered samples. Fig 1 shows a comparison of the Exact Match accuracy of NAQANet and MTMSN, between original and FI models. We observe that question permutations have little effect on the original model, as previously noted in (Al-Negheimish et al., 2021). Interestingly, passage permutations can drastically reduce performance by a third. While performance degrades for passage permutations, it remains unacceptably high, as we note that DROP is an unconstrained-QA task, so the space of possible answers is large. FI, on the other hand, succeeds in making the model sensitive to almost all permutations. We see that this generalises across BERT$_{\text{BASE}}$ and BERT$_{\text{LARGE}}$. More importantly, we observe that while the model with FI is sensitive to word order (no longer predicting original answers for permuted examples), the model's performance on well-ordered data is largely retained. To demonstrate that FI are correctly predicting invalid, table 1 shows the percentage of the data predicted as invalid, which is near-perfect for permuted samples.

			Part 1			Part 2		
	Variation	dev	3-gram	2-gram	1-gram	3-gram	2-gram	1-gram
UQA	NAQANet	3.15	94.64	97.18	99.59	99.01	99.93	100.00
	MTMSN Base	0.19	97.14	98.99	99.91	99.80	99.99	100.00
	MTMSN Large	0.07	95.01	97.52	98.98	99.43	100.00	100.00
NLI	RTE	0.36	94.57	98.19	99.64	96.01	96.74	98.55
	MNLI_M	1.36	94.52	97.60	99.21	94.83	97.35	99.45
	MNLI_MM	1.25	96.20	98.17	99.49	94.47	97.43	99.51
	ANLI1	0.70	99.80	100.00	100.00	95.20	98.10	99.69
	ANLI2	0.50	99.90	100.00	100.00	93.89	98.50	99.10
	ANLI3	1.33	99.83	99.92	100.00	94.83	97.33	98.67
GA	CoLA	2.30	92.66	92.35	97.93	-	-	-

Table 1: Percentage of evaluation data predicted as invalid in FI models in all of the tasks. dev is the unperturbed validation set. {n}-gram permutations of part 1 correspond to permutations of the question in UQA and of the premise in NLI. CoLA is made up of single sentences. All models succeed at flagging these samples as invalid.

5.2 Grammatical Acceptability

CoLA (Wang et al., 2018) is a task that measures models' ability to determine the grammatical acceptability of sentences. Pham et al. (2020) show that from the GLUE benchmark (Wang et al., 2018),

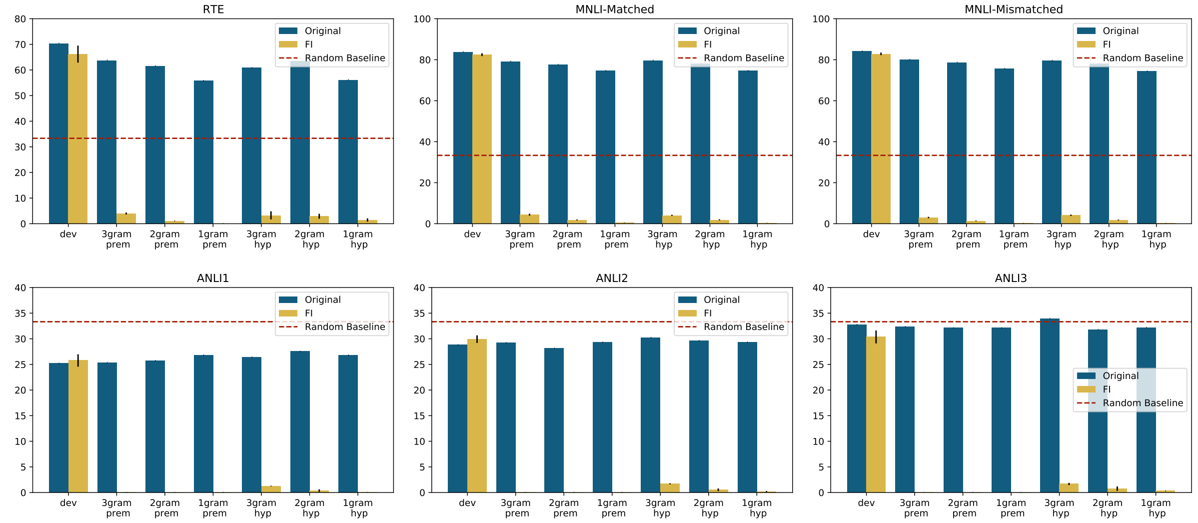


Figure 2: Accuracy for NLI tasks (a) trained on RTE, (b-f) trained on MNLI data. Original models exhibit a lack of sensitivity to word order, as they have the same accuracy regardless of the n-gram permutations. FI models are able to tell apart invalid examples even in out-of-distribution ANLI data.

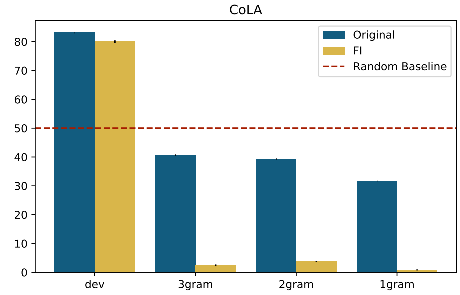


Figure 3: Accuracy of original models and FI on CoLA grammatical acceptability task. While the original model is below chance, we note that because this task is grammar-detection, the original model should not accept any of the permuted samples.

this task requires models to be most sensitive to word order. We applied FI by extending the BERT-based classification model with another '*invalid*' class label, to flag permuted sentences. Grammar and syntax have been destroyed in permuted sentences, we would expect the original model to label them as *not acceptable*. However, we see in Fig 3 that it maintains significantly high accuracy for 3-gram and 2-gram permutations (note that CoLA is an imbalanced dataset (70% acceptable)). Concretely, we observe that the standard BERT-based model labels $185/967$ 3-gram permutations, and $114/967$ 2-gram permutations as *acceptable*. Our FI approach significantly ameliorates this problem by increasing the sensitivity of the model to word-order permutations, as we observe in Fig 3.

5.3 Natural Language Inference (NLI)

NLI has been one of the important testbeds of previous work studying BERT-based models and their lack of sensitivity to word-order information (Sinha et al., 2021b; Abdou et al., 2022). These studies suggest that BERT-based models almost always assign the same labels to examples with perturbed word order as well-ordered ones highlighting the lack of sensitivity to word order, and likely dependence on shallow features. Similarly to §5.2, applying FI to an NLI BERT model was done by simply extending it with another class to represent *invalid* input. We perform a battery of experiments over a variety of NLI tasks in GLUE (Wang et al., 2018) such as RTE and MNLI (Williams et al., 2018) tasks. FI makes models highly sensitive to permuted sequences, as shown in Fig. 2. We verify in table 1 that those invalid sentences were correctly flagged as invalid. We also observe that FI-based models trained on MNLI and tested on out-of-distribution ANLI (Nie et al., 2020) show extreme sensitivity to word-order perturbations; this shows that the model has learned to flag invalid input and generalise to similar unseen tasks. Additionally, we observe that FI makes the models less likely to suffer from shortcut effects, which are common in models trained for NLI tasks (McCoy et al., 2019). Further experiments on Arabic NLI are presented in Appendix D, where we demonstrate that FI works well for other languages beyond English.

Heuristic Analysis

McCoy et al. (2019) introduces an evaluation set *Heuristic Analysis for NLI Systems (HANS)*, that examines surface heuristics NLI models are prone to adopting. They show in the existence

		Lexical Overlap	Subsequence	Constituent
Original	Entailment	97.76	99.92	100.00
	Non-Entailment	14.74	0.58	2.66
FI	Entailment	80.94	99.56	99.60
	Non-Entailment	64.56	38.14	11.48

Table 2: Comparison of BERT finetuned on MNLI with and without FI, FI makes the model more robust to the syntactic heuristics presented in (McCoy et al., 2019)

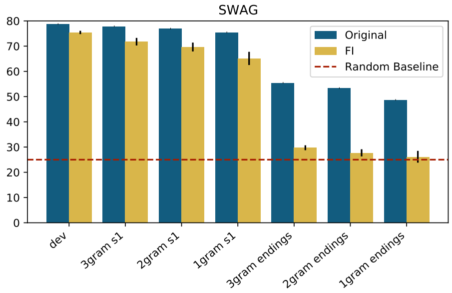


Figure 4: Accuracy of original models and FI on SWAG multiple-choice commonsense reasoning task, both are presented with the same four answer choices, FI enforces sensitivity to word order, especially on the endings.

of these heuristics, models always predict entailment, as they have near-perfect accuracy for that label, but perform poorly when the actual label is non-entailment (accuracy close to 0% in most cases). The heuristics targeted by this dataset are special cases of each other: the most general, *lexical overlap* heuristic: assume that all hypotheses constructed from words in the premise are entailed. Next follows is the *subsequence* heuristic: assume all hypotheses made up of contiguous subsequences of the premise are entailed. Finally, the *constituent* heuristic: assume all hypotheses made up of complete subtrees of the premise's parse tree are entailed. We expect that FI-models will be more robust to these shortcuts, and find (table 2) that this is indeed the case for lexical-overlap and the more challenging subsequence heuristics, substantially improving non-entailment performance, indicating that the models are no longer strictly biased with the presence of these heuristics. We note a surprising result of a drop in accuracy for entailed lexical overlap samples, which could be caused by the model no longer taking that shortcut, and warrants additional investigation in future work.

5.4 Multiple Choice Commonsense Reasoning

SWAG (Zellers et al., 2018) is a commonsense and grounded reasoning task that requires choosing between different possible ending scenarios given some context, where the context comprises of a primary sentence followed by the initial set of words for the following sentence. The model is trained to predict the most likely ending scenario given context and a list of four ending scenarios. To train our FORCED INVALIDATION model, we add an additional answer choice `'is invalid.'` to the data and perform n-gram based permutation of the primary sentence or over each of the endings. Nothing is changed in the architecture of the model. For evaluation, we maintain the same datasets for the Original and FI models, so they only contain the same number of answer choices, without the `'invalid'` choice. Ideally, the models should achieve random performance in the permuted cases. We present our results in Fig. 4; where firstly we observe that FI does not seem to reliably affect the model's sensitivity to word order perturbations with all the combinations. We specifically observe that n-gram permutations of primary sentence have little impact on the performance of FI-models and they seem to be less sensitive to word-order perturbations than expected. However, n-gram permutations of endings result in FI-models obtaining near-random performance as expected. We investigated the cause of this anomaly and observed that SWAG dataset has a prominent problem, in that, the primary sentence for a majority of cases is almost irrelevant to the model. A model trained on SWAG dataset is able to predict the correct answer for 60% of the examples without having access to the primary sentence.

6 Conclusion

In this paper, we presented a simple yet general technique called FORCED INVALIDATION, that significantly improves the sensitivity of models towards word order information. Our methodology requires minimal changes to the model and is sample efficient and drastically increases the sensitivity of the models to permutations of word order for a variety of tasks. We present a focused empirical validation of our methodology to showcase its generalisability. While in this paper we have focused on masked language models and attention-based models over word embeddings, we expect FI to generalise for other modelling setups such as RNN-based and CNN-based models and leave it as future work. We anticipate that this approach will also serve as a solution for other undesired behaviors in the model by explicitly invalidating such behaviours. We leave this as future work.

Limitations

While our empirical results showcase the effectiveness of FI and increase models' sensitivity to word order, the causal mechanisms are not currently obvious. It is not clear whether or not positional encodings are reflecting this change. Like previous work, our observations are additionally only restricted currently to English and Arabic (appendix D), further experiments are required to establish the problems relating to word order sensitivity and the utility of FI for other languages.

Acknowledgements

This research has been supported by a PhD scholarship from King Saud University. We thank our anonymous reviewers for their constructive feedback.

References

Mostafa Abdou, Vinit Ravishankar, Artur Kulmizev, and Anders Søgaard. 2022. Word order does matter and shuffled language models know it. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6907–6919, Dublin, Ireland. Association for Computational Linguistics.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Hadeel Al-Negheimish, Pranava Madhyastha, and Alessandra Russo. 2021. Numerical reasoning in machine reading comprehension tasks: are we there yet? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9643–9649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

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

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proc. of NAACL*.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. 2021. BERT & family eat word salad: Experiments with text understanding. *CoRR*, abs/2101.03453.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019. A multi-type multi-span network for reading comprehension that requires discrete reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1596–1606, Hong Kong, China. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Thang M. Pham, Trung Bui, Long Mai, and Anh Nguyen. 2020. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? *CoRR*, abs/2012.15180.

Brian Roark, Murat Saraclar, and Michael Collins. 2007. Discriminative n-gram language modeling. *Computer Speech & Language*, 21(2):373–392.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021a. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. 2021b. UnNatural Language Inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7329–7346, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

A Training Details

We use $\text{BERT}_{\text{BASE}}$ models with a classification head on top (BERTForSequenceClassification (Wolf et al., 2020)) for NLI and CoLA. It was trained for 5 epochs, batch size 16, learning rate 2e-5. We use a $\text{BERT}_{\text{BASE}}$ model (Wolf et al., 2020), for SWAG. It was trained for 3 epochs, batch size 16, learning rate 5e-5. All of the above are done using google colab with high-ram. MTMSN is based on the published codebase (Hu et al., 2019), and we use the same parameters to train, namely: BERT_{base}: batch size 24, 5 epochs and learning rate 3e-5, BERT_{large}: batch size 12, 10 epochs and learning rate 3e-5. Training was done on four RTX6000 GPUs with 24GB of RAM each for $\text{BERT}_{\text{LARGE}}$, and a single one was used for $\text{BERT}_{\text{BASE}}$. Models are trained for both settings, original and FI, to provide a fair comparison.

B Dataset Statistics

As described before, we filter out examples with sentences containing less than three words, such that we can generate at least 1 3-gram shuffle. Table 3 describes the tasks' validation sets before and after filtration.

	Original	Used	p1 #words	p2 #words
DROP numset	6848	6848	11	182
RTE	277	276	31	8
MNLI	9815	9289	16	9
MNLI-mm	9832	9551	17	10
ANLI1	1000	999	54	10
ANLI2	1000	999	54	9
ANLI3	1200	1199	52	9
CoLA	1043	967	7	-
SWAG	20006	19352	11	8

Table 3: Statistics of the validation set of datasets used for evaluation, including the original number of examples, after filtration, and median number of words for the first and second components.

Dataset licenses are mentioned in table 4:

	License
DROP numset	CC BY 4.0
RTE	Unknown
MNLI	OANC
MNLI-mm	OANC
ANLI1	CC BY-NC 4.0
ANLI2	CC BY-NC 4.0
ANLI3	CC BY-NC 4.0
CoLA	Unknown
SWAG	MIT license

Table 4: Artifact licenses for the datasets used.

C FI as a precursor to downstream task finetuning

Inspired by (Pham et al., 2020), we first perform FI on BERT to solely categorise valid and invalid samples for sentences that are sampled from Wikipedia. We replace the standard BERT-based contextual representations in MTMSN with FI-BERT based contextual embeddings, to see if it helps it become more sensitive to word order in the downstream DROP task. This did not show an improvement,

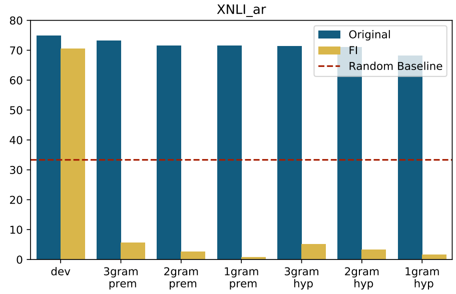


Figure 5: Accuracy of original models and FI on an Arabic NLI task. Once more, we see that the original models are insensitive to word order even in Arabic, while FI models learn to flag invalid samples.

however, where permuted examples are still predicted the same as well-ordered ones. This indicates that having an explicit way to flag invalid examples is helpful to the models.

D FORCED INVALIDATION with Other Languages

To establish the generalizability of this approach to other languages, we applied FIon an Arabic NLI task. We used the Arabic split of the XNLI dataset (Conneau et al., 2018) to finetune the ArBERT model (Abdul-Mageed et al., 2021), and compare the original training setup with FORCED INVALIDATION. Figure 5 shows that this approach successfully preserves the importance of word order beyond the English language.

How Many and Which Training Points Would Need to be Removed to Flip this Prediction?

Jinghan Yang
The University of Hong Kong
and Northeastern University
eciel@connect.hku.hk

Sarthak Jain
AWS AI Labs*
jsarth@amazon.com

Byron C. Wallace
Northeastern University
b.wallace@northeastern.edu

Abstract

We consider the problem of identifying a *minimal subset* of training data $\mathcal{S}_t$ such that if the instances comprising $\mathcal{S}_t$ had been removed prior to training, the categorization of a given test point x_t would have been different. Identifying such a set may be of interest for a few reasons. First, the cardinality of $\mathcal{S}_t$ provides a measure of robustness (if $|\mathcal{S}_t|$ is small for x_t, we might be less confident in the corresponding prediction), which we show is correlated with but complementary to predicted probabilities. Second, interrogation of $\mathcal{S}_t$ may provide a novel mechanism for *contesting* a particular model prediction: If one can make the case that the points in $\mathcal{S}_t$ are wrongly labeled or irrelevant, this may argue for overturning the associated prediction. Identifying $\mathcal{S}_t$ via brute-force is intractable. We propose comparatively fast approximation methods to find $\mathcal{S}_t$ based on *influence functions*, and find that—for simple convex text classification models—these approaches can often successfully identify relatively small sets of training examples which, if removed, would flip the prediction.[1]

1 Introduction

In this work we pose the following problem in the context of binary classification: *For a test point x_t, identify a minimum subset $\mathcal{S}_t$ of training data that one would need to remove in order to flip the prediction $\hat{y}_t$ for x_t.* This subset may be of interest for a few reasons. First, the cardinality k of $\mathcal{S}_t$ captures one measure of the (in)fragility of the prediction $\hat{y}_t$: Small k indicates that a minor change in the training data would have resulted in a different (discrete) prediction for x_t. We later show that this measure is correlated with, but complementary to, predicted probabilities.

Perhaps a more interesting motivation for recovering $\mathcal{S}_t$ is to provide a potential mechanism for *contesting* model predictions (Hirsch et al., 2017; Vaccaro et al., 2019), i.e., to enable individuals to interrogate and dispute automatic determinations that affect them. If removing a small set of training points would have yielded a different prediction, and if one could make the case for excluding these points (e.g., because they seem mislabeled, or reflect systematic labeling biases), this might provide a compelling case to overturn a model prediction. Consider an educator using an automated essay grading system.[2] Assume the system has output a comparatively poor grade for a student, a determination they see as unfair. Contesting the inclusion of a small set of examples ($\mathcal{S}_t$) which, if excluded, would have resulted in a higher grade provides a novel mechanism for disputation.

Naïvely attempting to find $\mathcal{S}_t$ by brute enumeration and re-training would be hopelessly inefficient. We introduce an algorithm for finding such sets efficiently using *influence functions* (Koh and Liang, 2017) which allow us to approximate changes in predictions expected as a result of removing subsets of training data (Koh et al., 2019). We then provide an iterative variant of this method which does a better job of identifying sets $\mathcal{S}_t$.

Across different datasets and models, we find that we are often able to recover subsets $\mathcal{S}_t$ with relatively small cardinality k; i.e., one can often identify a small to medium subset of training data which, if removed, would flip a given prediction. We also find that there are many test points for which models make predictions with high confidence but where k is small.

The **contributions** here include an investigation of the task of identifying minimal training sets to flip particular predictions in the context of text classification, algorithms for this problem, and an empirical evaluation of their performance in the con-

---

*Work done prior to joining Amazon.

[1] Code and data to reproduce all experiments available at: `https://github.com/ecielyang/Smallest_set`

[2] We put aside the question of whether using automated approaches in this particular setting is appropriate to begin with (likely not, though this may depend on how it is used).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2571–2584
May 2-6, 2023 ©2023 Association for Computational Linguistics

text of binary text classification.[3]

2 Methods

Assume a binary text classification problem. Given a training set $Z^{\text{tr}} = z_1, ..., z_N$, where $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, we aim to estimate the parameters θ of a classification model $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ to minimize the empirical risk, i.e., loss $\mathcal{L}$ over Z^{tr}: $\hat{\theta} := \text{argmin}_\theta \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(z_i, \theta) + \frac{\lambda}{2}\theta^T\theta$, which we will denote by $R(\theta)$. We assume throughout that R is twice-differentiable and strongly convex in θ, i.e., $H_{\hat{\theta}} := \nabla^2_\theta R(\hat{\theta}) := \frac{1}{N}\sum_{i=1}^{N} \nabla^2_\theta \mathcal{L}(z_i, \hat{\theta}) + \lambda I$ exists and is positive definite. Suppose we removed a subset of k training points $\mathcal{S} \subset Z^{\text{tr}}$ and re-estimated θ, yielding new parameters $\hat{\theta}_{\mathcal{S}}$. Letting $\varepsilon = -\frac{1}{N}$, we can write this as:

$$\hat{\theta}_S = \text{argmin}_{\theta\in\Theta}\{R(\theta) + \varepsilon \sum_{z_i \in S} \mathcal{L}(z_i, \theta)\} \quad (1)$$

In principle, one could remove the points in $\mathcal{S}$ and re-train to find $\hat{\theta}_{\mathcal{S}}$. In practice this is infeasible given the number of potential subsets $\mathcal{S}$. Koh and Liang (2017) provide (relatively) efficient approximations to estimate $\hat{\theta}_{\mathcal{S}}$ when k=1. Subsequent work (Koh et al., 2019) found that this approximation correlates well with the actual empirical effects of removing a *set* of points (where $k > 1$).

Finding influential subsets Given input x_t, we aim to design an approach to efficiently *identify* the smallest subset $\mathcal{S}_t$ of Z^{tr} such that removing these examples prior to training would change $\hat{y}_t$. Prior work (Cook and Weisberg, 1982; Koh and Liang, 2017) derived the influence exerted by a train point i on the *loss* incurred for a test point t as:

$$-\nabla_\theta \mathcal{L}(z_t, \hat{\theta})^\intercal \underbrace{H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(z_i, \hat{\theta})}_{\Delta_i\theta} \quad (2)$$

Where $\Delta_i\theta$ is the influence of upweighting z_i during training on estimates $\hat{\theta}$ (Cook and Weisberg, 1982). We are interested, however, in identifying points that have a particularly strong effect on a specific observed *prediction*. We therefore modify Equation 2 to estimate the *influence on prediction* (IP), i.e., the change in predicted probability for x_t observed after removing training instance i. This can be expressed as:

$$\Delta_t f_i := -\nabla_\theta f_{\hat{\theta}}(x_t)^\intercal \Delta_i\theta \quad (3)$$

We then approximate the change in prediction on instance t we would anticipate after removing the training subset $\mathcal{S}_t$ from the training data as the sum of the $\Delta_t f_i$ terms for all points $x_i \in \mathcal{S}_t$.

Algorithm 1 describes a method for constructing $\mathcal{S}_t$. We estimate the change in output expected upon removing each instance from the training dataset and assemble these in $\Delta_t f$. We then greedily consider adding these differences (effectively adding points to $\mathcal{S}_t$) until the resultant output is expected to cross the classification threshold (τ); if we exhaust the training dataset without crossing τ, then we have failed to identify a set $\mathcal{S}_t$.

Algorithm 1: A simple method to find a minimal subset to flip a test prediction

```
Input: $f$: Model; $Z^{\text{tr}}$: Full training set; $\hat{\theta}$:
       Parameters estimated $Z^{\text{tr}}$; $\mathcal{L}$: Loss
       function; $x_t$: A test point; $\tau$:
       Classification threshold (e.g., 0.5)
Output: $\mathcal{S}_t$: minimal train subset
        identified to flip the prediction ($\emptyset$
        if unsuccessful)
$H \leftarrow \nabla^2_\theta \mathcal{L}(Z^{\text{tr}}, \hat{\theta})$
$\Delta\theta \leftarrow H^{-1} \nabla_{\hat{\theta}} \mathcal{L}(Z^{\text{tr}}, \theta')$
$\Delta_t f \leftarrow \nabla_\theta f_{\hat{\theta}}(x_t)^\intercal \Delta\theta$
$\hat{y}_t \leftarrow f(x_t) > \tau$ // Binary prediction
// Sort instances (and estimated
   output differences) in order of
   the current prediction
direction $\leftarrow$ {$\uparrow$ if $\hat{y}_t$ else $\downarrow$}
indices $\leftarrow$ argsort($\Delta_t f$, direction)
$\Delta_t f \leftarrow$ sort($\Delta_t f$, direction)
for $k = 1 ... |Z^{tr}|$ do
    $\hat{y}'_t = (f(x_t) + \text{sum}(\Delta_t f[:k])) > \tau$
    if $\hat{y}'_t \neq \hat{y}_t$ then
        return $Z^{\text{tr}}$[indices[: $k$]]
return $\emptyset$
```

Algorithm 1 is simple and relatively fast, but we can improve upon it by *iteratively* identifying smaller subsets $\mathcal{S}_t$ in Algorithm 2. We detail this approach in Appendix Algorithm 2, but describe it briefly as follows.

We start with the entire train set as a "candidate" $\tilde{\mathcal{S}}_t$, and then iteratively attempt to find strict subsets

---

[3] Recent related work in economics by Broderick et al. (2020) proposed and investigated a similar problem, with a focus on identifying the sensitivity of econometric analyses to removal of small subsets of data. Recent work on *datamodeling* (Ilyas et al., 2022) also considered a variant of this problem (see Section 5).

Models	Algorithm 1	Algorithm 2
Movie reviews		
BoW	247	151
BERT	484	303
Essays		
BoW	352	134
BERT	484	135
Emotion classification		
BoW	500	345
BERT	524	327
Hate speech		
BoW	808	415
BERT	546	239
Tweet sentiment		
BoW	345	177
BERT	858	569

Table 1: The comparison of average on $k = |\mathcal{S}_t|$ values from Algorithm 1 and Algorithm 2 over the subsets of test points x_t for which we were able to successfully identify a set of points $|\mathcal{S}_t|$

Features	Found $\mathcal{S}_t$	Flip successful
Movie reviews		
BoW	78%	78%
BERT	79%	72%
Essays		
BoW	12%	11%
BERT	9%	8%
Emotion classification		
BoW	91%	91%
BERT	83%	71%
Hate speech		
BoW	67%	60%
BERT	53%	44%
Tweet sentiment		
BoW	99%	91%
BERT	90%	68%

Table 2: Percentages of test examples for which Algorithm 2 successfully identified a set $\mathcal{S}_t$ to remove (center) and for which upon removing these instances and retraining the prediction indeed flipped (right).

of this that by themselves would flip the prediction $\hat{y}_t$. On the first pass, this is equivalent to Algorithm 1, after which—if successful—we will have found a candidate set $\tilde{\mathcal{S}}_t$. Here we update parameter estimates θ to approximate "removing" the points in $\tilde{\mathcal{S}}_t$, and then we recompute the approximation of the influence that points in $\tilde{\mathcal{S}}_t$ would have on $\hat{y}_t$ using a single-step Newton approximation. The idea is that after the parameter update this approximation will be more accurate, potentially allowing us to find a smaller $\mathcal{S}_t$. This process continues until we are unable to find a new (smaller) subset.

In sum, this variant of the algorithm attempts to iteratively identify increasingly small subsets $\tilde{\mathcal{S}}_t$ which would, upon removal prior to training, overturn the original prediction $\hat{y}_t$. There is a computational cost to this, because each iteration involves approximating the influence on a particular prediction; this is computationally expensive. This variant therefore trades run-time for (hopefully) more accurate identification of minimal $\mathcal{S}_t$. However, we find that empirically Algorithm 2 ends up running for only 2.3 passes on average (across all experiments). That is, the algorithm adds a scalar to the run-time of Algorithm 1, but often yields considerably smaller $\mathcal{S}_t$. We show the comparison of $|\mathcal{S}_t|$ returned by two algorithms in the Table 1.

3 Experimental Setup

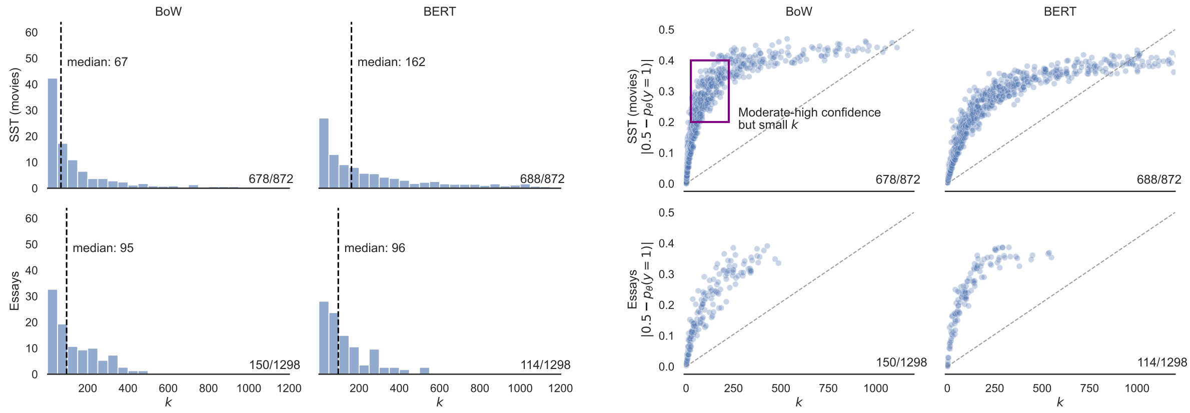
Datasets We use five binary text classification tasks: Movie review sentiment (Socher et al., 2013); Twitter sentiment classification (Go et al., 2009); Essay grading (Foundation, 2010); Emotion classification (Saravia et al., 2018), and; Hate speech detection (de Gibert et al., 2018). We binarize the essay data by labeling the top 10% score points as 1 ("A"s) and others as 0. For the emotion dataset, we include only "joy" and "sadness". We provide dataset statistics in Appendix Table A1. Because the hate speech data is severely imbalanced, we selected a classification threshold τ post-hoc in this case to maximize train set F1 (yielding $\tau = 0.25$); for other datasets we used $\tau = 0.5$, which corresponded to reasonable F1 scores—for reference we report prediction performance on all datasets in Appendix Table A2.

Models We consider only $\ell 2$ regularized logistic regression (for which influence approximation is well-behaved). As features, we consider both bag-of-words and neural embeddings (induced via BERT; Devlin et al. 2018).

4 Results

Here we present results for the iterative method (Appendix Algorithm 2), which outperforms the simpler Algorithm 1. We provide full results for both methods in the in the Appendix.

How often can we find $\mathcal{S}_t$ and how frequently does removing the instances it contains flip the prediction? As can be seen in Table 2, this varies considerably across datasets. For movie reviews, Algorithm 2 returns a set $\mathcal{S}_t$ for ~80% of test points, whereas for the (more complex) essays data it does so for only ~10% of instances. Other datasets see success somewhere in-between these extremes. However, when the algorithm does return a set $\mathcal{S}_t$, removing this and re-training almost



(a) Histograms of $k = |\mathcal{S}_t|$ values over the subsets of test points x_t for which we were able to successfully identify a set of points $\mathcal{S}_t$ such that removing them would flip the prediction for $\hat{y}_t$. We report the fraction for which we were able to do so in the lower sub-plot right corners.

(b) Relationship between predicted probabilities and $k = |\mathcal{S}_t|$ identified. These are correlated (as we would expect), but there are many points for which the model is moderately or highly confident, but where removing a relatively small set of training data would change the prediction.

Figure 1: Results characterizing $\mathcal{S}_t$ on two illustrative datasets (sentiment classification and essay scoring).

always flips the prediction $\hat{y}_t$ (right-most column).

What is the distribution of $k = |\mathcal{S}_t|$? Figure 1a shows empirical distributions of k values for subsets $\mathcal{S}_t$ identified by Algorithm 2 for the illustrative movie review and essay grading datasets (full results in Appendix). The take-away here is that when we do find $\mathcal{S}_t$, its cardinality is often quite small. Indeed, for many test points removing tens of examples would have flipped the prediction.

How does k relate to predicted probability? Does the size of $\mathcal{S}_t$ tell us anything beyond what we might infer from the predicted probability $p(y_t = 1)$? In Figure 1b we show (again for just two datasets here) a scatter of $k = |\mathcal{S}_t|$ against the distance of the predicted probability from 0.5. The former provides complementary information, in that there exist instances about which the model is confident, but where removing a small set of training instances would overturn the prediction.

Qualitative example. One reason to recover sets $\mathcal{S}_t$ is to support *contestation*—if k is small, one might argue against the appropriateness of the points in $\mathcal{S}_t$ and hence against the determination y_t. As a simple example,[4] consider the movie review test instance "*Manages to transcend the sex drugs and show tunes plot into something far richer*". The true label is positive, but the model predicted negative. Algorithm 2 reveals that removing a single example ($k = 1$) from the training set would have reversed the prediction—specifically, this negative review: "*An overstylized pureed melange of sex psychology drugs and philosophy*". It seems this training point is only superficially similar to the test point, which may make a case for overturning the prediction. While standard influence functions (Koh and Liang, 2017) can be used to *rank* training points, the novelty here is observing that *removing this point alone* would change the prediction.

5 Related Work

Influence functions (Hampel, 1974; Cook and Weisberg, 1980, 1982) provide machinery to identify training points that most informed a particular test prediction. Influence can provide insight into predictions made by modern neural networks (Koh and Liang, 2017), and can be used to *debug* models and training data by surfacing mislabeled training points and/or reliance on artifacts (Adebayo et al., 2020; Han et al., 2020; Pezeshkpour et al., 2022; Teso et al., 2021), and tuning influence can be used to demote reliance on unwanted correlations (Han and Tsvetkov, 2021). Influence can also be used to *audit* models by inspecting training data responsible for predictions Marx et al. (2019).

Schulam and Saria (2019) audit individual predictions by approximating how much they might have changed under different samples from the training distribution. Ting and Brochu (2018) consider influence functions as a tool for optimally subsampling data in service of computational effi-

---

[4] We provide more qualitative analysis in the Appendix.

ciency. Koh et al. (2019) considered approximating the effect of removing a *group* of training points using influence functions, and found that they do so fairly well (a result that we use). They assumed groups were *given* and then evaluated the accuracy of the influence approximation to the change in prediction. By contrast, we are interested in *finding* a (minimal) group which would have the specific effect of flipping a prediction. Elsewhere, Khanna et al. (2019) ask: "Which training examples are most responsible for a given *set* of predictions?".

Broderick et al. (2020) assess the robustness of economic analyses when a fraction of data is removed. They therefore focus on the magnitude/significance of parameter estimates. This framing differs from our ML-centric motivation, which aims to recover specific small subsets of data that, if removed, would change a particular prediction (and so might support contestability).

Robustness of data analyses to dropping training data In the process of review it was brought to our attention that Broderick et al. (2020) addressed a closely related problem to what we have considered here, albeit from a quite different motivating perspective—namely assessing the sensitivity of econometric analyses to removals of small subsets of data. It turns out that the algorithm that was (independently) proposed by Broderick et al. (2020) in that work is similar to Algorithm 1. The present effort is novel in our focus on machine learning, and specifically on identifying minimal subsets of training data which would flip a particular prediction if removed prior to training.

Minimal feature set removal Another related line of work concerns a natural complement to the problem we have considered: Instead of identifying a minimal set of *instances* to remove in order to change a prediction, the idea is to find a minimal subset of *features* such that, if these were set to uninformative values, a particular prediction would change (Harzli et al., 2022). Work on *counterfactual examples* has similarly sought to identify minimal (feature) edits to instances that would change the associated label (Kaushik et al., 2019).

Datamodeling Recent work on *datamodeling* (Ilyas et al., 2022) provided a generalized framework for analyzing model behavior as a function training data. This approach entails *learning to estimate* (via a parameterized model) changes we would anticipate observing for a particular instance if the model had been trained on some subset of the original training set. This approach is flexible, and one thing it permits is identifying the *data support* of a particular prediction for x_t, i.e., what we have called $\mathcal{S}_t$ (4.1.1 in Ilyas et al. 2022). Furthermore, this method is not restricted to the simple regularized linear models we have considered here. However, this comes with the downside of high computation costs: One needs to re-train the algorithm being modeled many times with different training data subsets to yield a "training set" to be used to estimate model behavior under conterfactual training sets. The main comparative advantage of our more focused approach is therefore relative computational efficiency.

Contestability (Vaccaro et al., 2019; Almada, 2019) in ML is the idea that individuals affected by a prediction ought to be able to challenge this determination, which may require parties to "marshal evidence and create counter narratives that argue precisely why they disagree with a conclusion drawn by an AI system" (Hirsch et al., 2017). The right to contestability is in some cases enshrined into law (Almada, 2019). Identifying $\mathcal{S}_t$ for review by an individual affected by the prediction $\hat{y}_t$ may constitute a concrete mechanism for contestation.

6 Conclusions

In the context of binary text classification, we investigated the problem of identifying a minimal set of training points $\mathcal{S}_t$ such that, if excluded from training, the prediction for test instance x_t would flip. We proposed two relatively efficient algorithms for this—both using approximate group influence (Koh and Liang, 2017; Koh et al., 2019)—and showed that for regularized linear models they can often find relatively small $\mathcal{S}_t$. We provided empirical evidence that this captures uncertainty in a way that is somewhat complementary to predicted probabilities, and may serve as a mechanism to support *contestability*, by allowing individuals to review (and dispute) instances in $\mathcal{S}_t$.

Limitations

A key limitation of this work is that we have restricted analysis to regularized linear models with convex loss. We leave extension and evaluation of the proposed methods for more complex models to future work. Indeed, our hope is that this initial effort inspires further work on the problem of identifying minimal train sets which would overturn a specific prediction if removed.

More conceptually, the implications of finding a small subset $\mathcal{S}_t$ are not entirely clear. Intuitively, small sets would seem to indicate fragility, but we have not formalized or evaluated this further. Moreover, there may in certain cases exist *multiple* (distinct) subsets $\mathcal{S}_t$, such that removing any of these subsets would flip the prediction for x_t. This would complicate the process of contestation envisioned. Furthermore, assuming a stochastic parameter estimation method (e.g., SGD) the composition of $\mathcal{S}_t$ may depend on the arbitrary random seed, similarly complicating the interpretation of such sets.

Acknowledgements

We thank our anonymous EACL reviewers for helpful feedback, especially with respect to relevant related work. We also thank Gautam Kamath for highlighting connections to the *datamodeling* work (Ilyas et al., 2022). This work was supported in part by the Army Research Office (W911NF1810328), and in part by the Overseas Research Fellowship under the University of Hong Kong.

Ethics Statement

Models are increasingly used to make (or aid) decisions that directly affect individuals. In addition to the broader (potential) "interpretability" afforded by recovering small sets of training data that would change a prediction if removed, this may provide a new mechanism for individuals to contest such automated decisions, specifically by disputing this set of training data in some way. However, our proposed method only finds training points that highly impact the model prediction for a given example; these may or may not be noisy or problematic instances. Human judgement is required to assess the accuracy and relevancy of the instances in $\mathcal{S}_t$.

A broader view might be that classification models are simply not appropriate for the kinds of sensitive applications we have used as motivation here. The use of (semi-)automated methods for essay grading, e.g., has long been debated (Hearst, 2000). One might argue that rather than trying to provide mechanisms to contest ML predictions, a better choice may be not to use models in cases where these would be necessary at all. We are sympathetic to this view, but view the "appropriateness" of ML for a given problem as a spectrum; contestability may be useful even in "lower stakes" cases. Moreover, the general problem we have introduced of identifying small training sets which can by themselves swing predictions, and the corresponding methods we have proposed for recovering these, may be of intrinsic interest beyond contestability (e.g., as an additional sort of model uncertainty).

References

Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. 2020. Debugging tests for model explanations. *arXiv preprint arXiv:2011.05429*.

Marco Almada. 2019. Human intervention in automated decision-making: Toward the construction of contestable systems. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, pages 2–11.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. 2020. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pages 1899–1909. PMLR.

Tamara Broderick, Ryan Giordano, and Rachael Meager. 2020. An automatic finite-sample robustness metric: When can dropping a little data make a big difference? *arXiv preprint arXiv:2011.14999*.

Guillaume Charpiat, Nicolas Girard, Loris Felardos, and Yuliya Tarabalka. 2019. Input similarity from the neural network perspective. *Advances in Neural Information Processing Systems*, 32.

R Dennis Cook and Sanford Weisberg. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508.

R Dennis Cook and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Hewlett Foundation. 2010. The hewlett foundation: Automated essay scoring.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009.

Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

Xiaochuang Han and Yulia Tsvetkov. 2021. Influence tuning: Demoting spurious correlations via instance attribution and instance-driven updates. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4398–4409, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. *arXiv preprint arXiv:2005.06676*.

Ouns El Harzli, Bernardo Cuenca Grau, and Ian Horrocks. 2022. Minimal explanations for neural network predictions. *arXiv preprint arXiv:2205.09901*.

Marti A Hearst. 2000. The debate on automated essay grading. *IEEE Intelligent Systems and their Applications*, 15(5):22–37.

Tad Hirsch, Kritzia Merced, Shrikanth Narayanan, Zac E Imel, and David C Atkins. 2017. Designing contestability: Interaction design, machine learning, and mental health. In *Proceedings of the 2017 Conference on Designing Interactive Systems*, pages 95–99.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. 2022. Datamodels: Understanding predictions with data and data with predictions. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9525–9587. PMLR.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*.

Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. 2019. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390. PMLR.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Pang Wei W Koh, Kai-Siang Ang, Hubert Teo, and Percy S Liang. 2019. On the accuracy of influence functions for measuring group effects. *Advances in neural information processing systems*, 32.

Charles Marx, Richard Phillips, Sorelle Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. 2019. Disentangling influence: Using disentangled representations to audit model predictions. *Advances in Neural Information Processing Systems*, 32.

Pouya Pezeshkpour, Sarthak Jain, Sameer Singh, and Byron Wallace. 2022. Combining feature and instance attribution to detect artifacts. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1934–1946, Dublin, Ireland. Association for Computational Linguistics.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Peter Schulam and Suchi Saria. 2019. Can you trust this prediction? auditing pointwise reliability after learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1022–1031. PMLR.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Stefano Teso, Andrea Bontempelli, Fausto Giunchiglia, and Andrea Passerini. 2021. Interactive label cleaning with example-based explanations. *Advances in Neural Information Processing Systems*, 34:12966–12977.

Daniel Ting and Eric Brochu. 2018. Optimal subsampling with influence functions. *Advances in neural information processing systems*, 31.

Kristen Vaccaro, Karrie Karahalios, Deirdre K Mulligan, Daniel Kluttz, and Tad Hirsch. 2019. Contestability in algorithmic systems. In *Conference Companion Publication of the 2019 on Computer Supported Cooperative Work and Social Computing*, pages 523–527.

Dataset	# Train	# Test	% Pos
Movie reviews	6920	872	0.52
Essays	11678	1298	0.10
Emotion	9025	1003	0.53
Hate speech	9632	1071	0.11
Tweet sentiment	18000	1000	0.50

Table A1: Text classification dataset statistics.

Models	Accuracy	F1-score	AUC
	Movie reviews		
BoW	0.79	0.80	0.88
BERT	0.82	0.83	0.91
	Essays		
BoW	0.97	0.80	0.99
BERT	0.97	0.84	0.99
	Emotion classification		
BoW	0.77	0.79	0.86
BERT	0.80	0.82	0.88
	Hate speech		
BoW	0.87	0.40	0.81
BERT	0.89	0.63	0.88
	Tweet sentiment		
BoW	0.70	0.70	0.75
BERT	0.75	0.76	0.84

Table A2: The model performance respect to datasets included in the experiment.

A Appendix

A.1 Dataset Statistics and Predictive Performance

We present basic statistics describing our text classification datasets in Table A1. For the tweet sentiment dataset, we randomly sampled 19000 points from the 1600000 points to make experiments feasible. For reference, we also report the predictive performance realized by the models considered on the test sets of these corpora in Table A2.

A.2 Full results

Table A3 reports the percentages of instances for which Algorithm 1 identifies a subset $\mathcal{S}_t$ (center column), and for which this set actually flipped the prediction following removal (right column). Contrast this with Table 2, which reports the same for the proposed iterative approach in Algorithm 2.

We provide histograms of $k = |\mathcal{S}_t|$ for the sets we were able to identify via Algorithm 1 in Figure A.2, and the same plots for Algorithm 2 in Figure A.4.

Finally, we plot the relationship between k and predicted probabilities under Algorithms 1 and 2 in Figures A.3 and A.5, respectively.

Features	Found $\mathcal{S}_t$	Flip successful
	Movie reviews	
BoW	78%	78%
BERT	79%	76%
	Essays	
BoW	12%	12%
BERT	9%	9%
	Emotion classification	
BoW	91%	91%
BERT	83%	78%
	Hate speech	
BoW	67%	65%
BERT	53%	49%
	Tweet sentiment	
BoW	99%	98%
BERT	90%	73%

Table A3: Percentages of test examples for which Algorithm 1 successfully identified a set $\mathcal{S}_t$ (center) and for which upon removing these instances and retraining the prediction indeed flipped (right).

A.3 Additional qualitative analysis

We conclude with a brief qualitative analysis of examples in $\mathcal{S}_t$ retrieved in the case of the essays data. The model operating over BERT representations classified this test point (x_t) as 0, i.e., not an "A": "*The cyclist in this essay was a very brave man* ...". The example is about a paragraph in length total, but details adventures of a cyclist. In this case it happens that the reference label is, in fact, an "A", so the model is incorrect. Algorithm 2 reveals that removing a single training point and retraining would have overturned this prediction, yielding an "A". The point in question is labeled 0 (so below an "A") and is about the mood of a memoir, in particular arguing that the person central to this was happy. The student-author of the cyclist essay might reasonably argue that this example is not at all relevant to their essay, and the fact that excluding this single example would have meant their essay received an "A" may be an adequate case for changing their grade accordingly.

A.4 Time complexity

We recorded wall clock times required to search for $|\mathcal{S}_t|$ on all test points in each dataset using Algorithm 1 and Algorithm 2 on Intel(R) Core(TM) i9-9920X CPUs; we report these times in Table A4. For Algorithm 1, the longest running time is required for the essay dataset because most test predictions cannot be flipped even after iterating over all training points. Algorithm 2 is considerably slower than Algorithm 1. The main reason lies in recording the set of training points not in $\mathcal{S}_t$ (line 20 in Algorithm 2) and re-calculating the IP value in each iteration to reduce the minimal candidate set. This additional time is traded off against the ability to (typically) find smaller $\mathcal{S}_t$ compared with 1. Overall, the running time required to find $|\mathcal{S}_t|$ for one test point is relatively minimal for both algorithms.

A.5 Attribution methods

We consider different methods (i.e., other than influence functions) to rank training instances including gradient similarities in terms of the loss, similarity-based methods and randomly sampling training points. Of these we found that the proposed method works best in terms of finding instances which exert maximal influence on the prediction.

One natural way to quantify the impact of a training point x_i on a training point x_t by similarity methods. If the model has training points similar to the test point, it may classify the test correctly with high probability. We consider three of similarity-based methods: **EUC** $= -||x_t - x_i||^2$, **DOT** $= \langle x_i, x_t \rangle$, and **COS** $= cos(x_i, x_t)$.

Apart from influence function and IP, we consider gradient-based instance attribution methods:

1) $RIF = cos(H^{-\frac{1}{2}} \nabla_\theta \mathcal{L}(x_t), H^{-\frac{1}{2}} \nabla_\theta L(x_i))$

2) $GD = \langle \nabla_{\mathcal{L}} L(x_t), \nabla_\theta L\mathcal{L} \rangle$

3) $GC = cos(\nabla_\theta \mathcal{L}(x_t), \nabla_\theta \mathcal{L}(x_i))$

RIF was proposed to mitigate the issues of outliers and mislabeled points being returned by the standard influence functions (Barshan et al., 2020). GC and GD measure the similarity between two instances can also become an effective way to interpret the model from the instance perspective (Charpiat et al., 2019). Apart from the methods above, we randomly sample training subsets and remove them accordingly.

We apply the above methods to the movie review dataset trained with a logistic regression model. We evaluate each attribution method as follows: First, we remove the top $k = |\mathcal{S}_t|$ training points from the training dataset according to the score calculated from the attribution method. Then we train a new with the same dataset except for the removed points. Finally, we compare the difference in predictions for each test point from the old model to the new model. To show the impact of attribution methods under different $k = |\mathcal{S}_t|$, we iterated with $k = |\mathcal{S}_t|$ from 50 to 3000. The mean absolute difference is plotted along with $k = |\mathcal{S}_t|$ shown in Figure A.1. IP has a larger impact on the predicted probability, compared to removing training points ranked according to other methods.

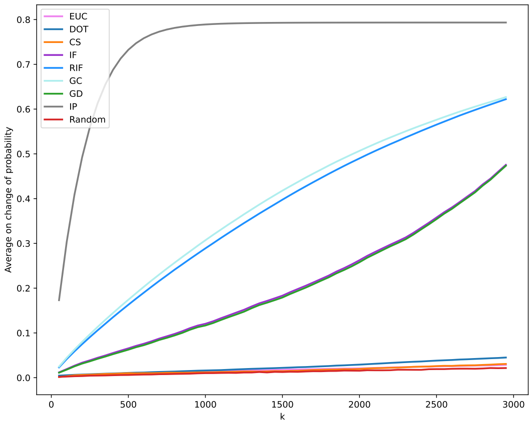


Figure A.1: The relationship between the mean of absolute difference on predicted probabilities for all test points results from removing $|\mathcal{S}_t|$ training points, using different methods.

Datasets		Movie reviews	Essays	Emotion	Hate speech	Tweet
Bow	Algorithm 1	5	155	5	40	11
	Algorithm 2	239	257	534	444	1529
BERT	Algorithm 1	3	161	19	52	8
	Algorithm 2	604	288	761	522	2203

Table A4: Running time (in seconds) to find $|\mathcal{S}_t|$ for all test points in each data set by Algorithm 1 and Algorithm 2.

Algorithm 2: An *iterative* approach to finding a minimal set to flip a prediction

```
Input: $f$: Model; $Z^{\text{tr}}$: Full training set; $\hat{\theta}$: Parameters estimated $Z^{\text{tr}}$; $\mathcal{L}$: Loss function; $x_t$: A test point; $\tau$:
        Classification threshold (e.g., 0.5)
Output: $\mathcal{S}_t$: minimal train subset identified to flip prediction for $x_t$ ($\emptyset$ if unsuccessful)
$\theta' \leftarrow \hat{\theta}$
$Z^{\text{tr}}_{\text{r}}, \tilde{\mathcal{S}}_t \leftarrow Z^{\text{tr}}, Z^{\text{tr}}$ // Track remaining points and candidate subset $\tilde{\mathcal{S}}_t$
$H \leftarrow \nabla^2_{\theta}\mathcal{L}(Z^{\text{tr}}, \theta')$
$\Delta\theta \leftarrow H^{-1}\nabla_{\theta}\mathcal{L}(Z^{\text{tr}}, \theta')$
$\Delta_t f \leftarrow \nabla_{\theta} f_{\hat{\theta}}(x_t)^{\intercal}\Delta\theta$
$\hat{y}_t \leftarrow f(x_t) > \tau$
$\Delta_t f_{\text{sum}} \leftarrow 0$
$k' \leftarrow |Z^{\text{tr}}|$
while $\tilde{S}$ changed since last iteration do
    // Sort instances (and estimated output differences) in order of the current prediction
    direction $\leftarrow \{\uparrow$ if $\hat{y}_t$ else $\downarrow\}$
    indices $\leftarrow$ argsort($\Delta_t f$, direction)
    $\Delta_t f \leftarrow$ sort($\Delta_t f$, direction)
    for $k = 1 \ldots |\tilde{\mathcal{S}}_t|$ do
        $\hat{y}'_t \leftarrow (f(x_t) + \text{sum}(\Delta_t f[:k])) > \tau$
        if $\hat{y}'_t \neq \hat{y}_t$ then
            $\Delta_t f_{\text{sum}} \leftarrow \text{sum}(\Delta_t f[:k])$
            diff $\leftarrow k' - k$
            $k' \leftarrow k$
            $\tilde{\mathcal{S}}_t \leftarrow \tilde{\mathcal{S}}_t$[indices$[:k]$]                // Update candidate subset
            $Z^{\text{tr}}_{\text{r}} \leftarrow Z^{\text{tr}}/\tilde{\mathcal{S}}_t$          // And the set of training points not in $\tilde{\mathcal{S}}_t$
            $\theta' \leftarrow \theta + \Delta\theta$[indices$[:k]$]
            // Update Hessian and $\nabla$ of loss using updated $\theta$ estimate
            $H \leftarrow \nabla^2_{\theta}\mathcal{L}(Z^{\text{tr}}_{\text{r}}, \theta')$
            $\Delta\theta \leftarrow H^{-1}\nabla_{\theta}\mathcal{L}(\tilde{\mathcal{S}}_t, \theta')$
            $\Delta_t f \leftarrow \nabla_{\theta} f_{\hat{\theta}}(x_t)^{\intercal}\Delta\theta$
            break
if $|\tilde{\mathcal{S}}_t| = |Z^{tr}|$ then
    return $\emptyset$
return $\tilde{S}_t$
```

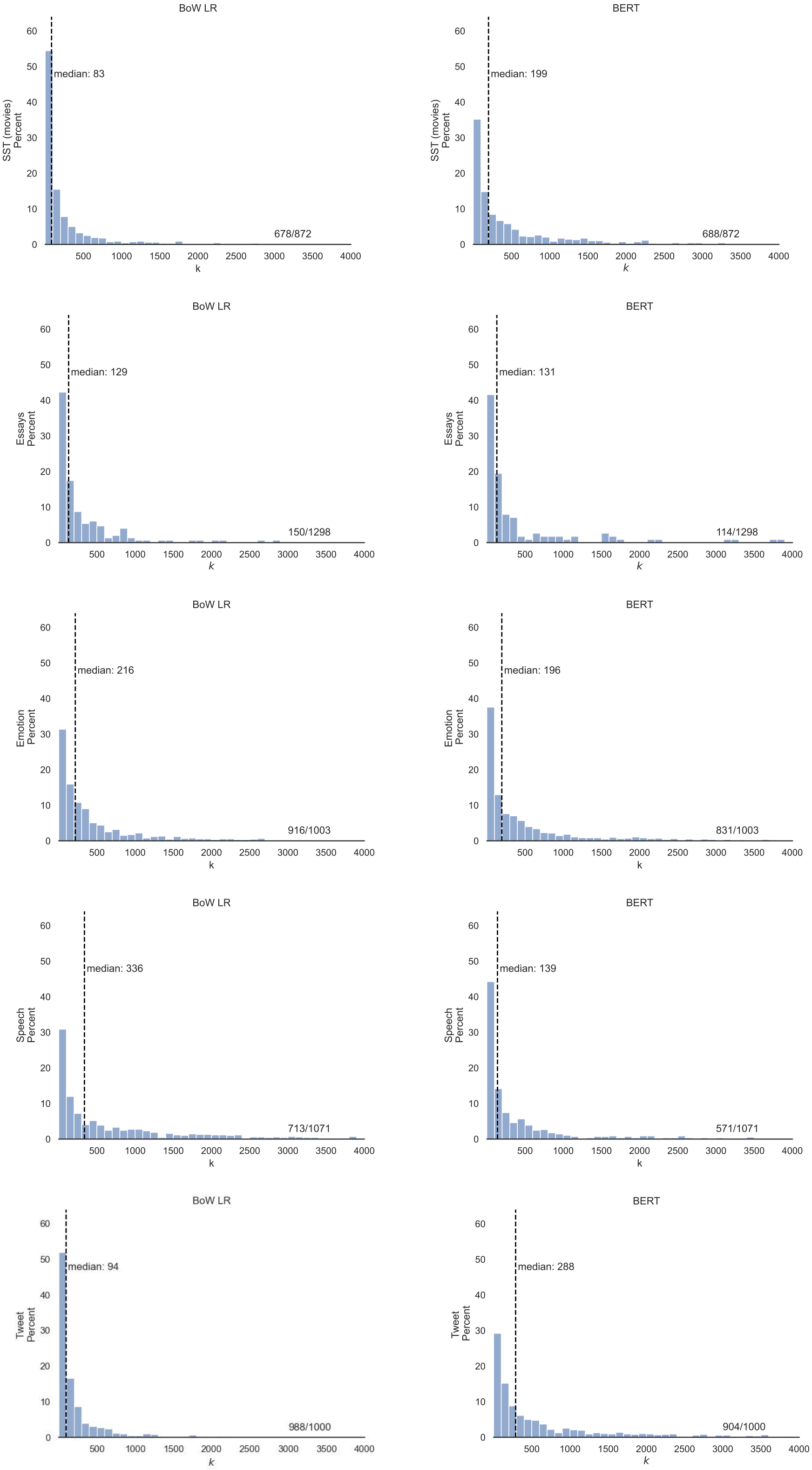


Figure A.2: Histograms of $k = |\mathcal{S}_t|$ values from Algorithm 1 over the subsets of test points x_t for which we were able to successfully identify a set of points $\mathcal{S}_t$ such that removing them would flip the prediction for $\hat{y}_t$.

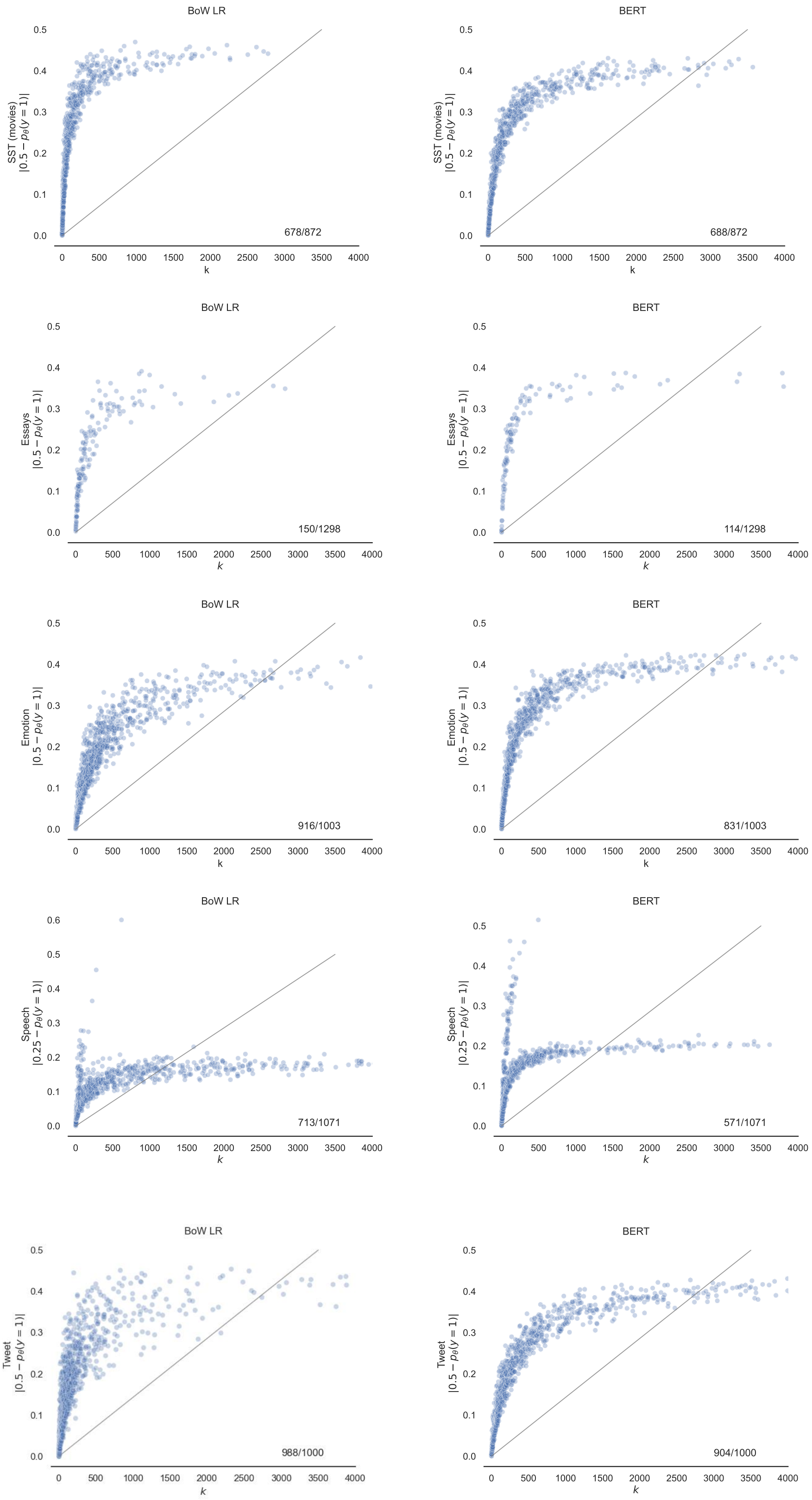


Figure A.3: Relationship between predicted probabilities and $k = |\mathcal{S}_t|$ identified from Algorithm 1.

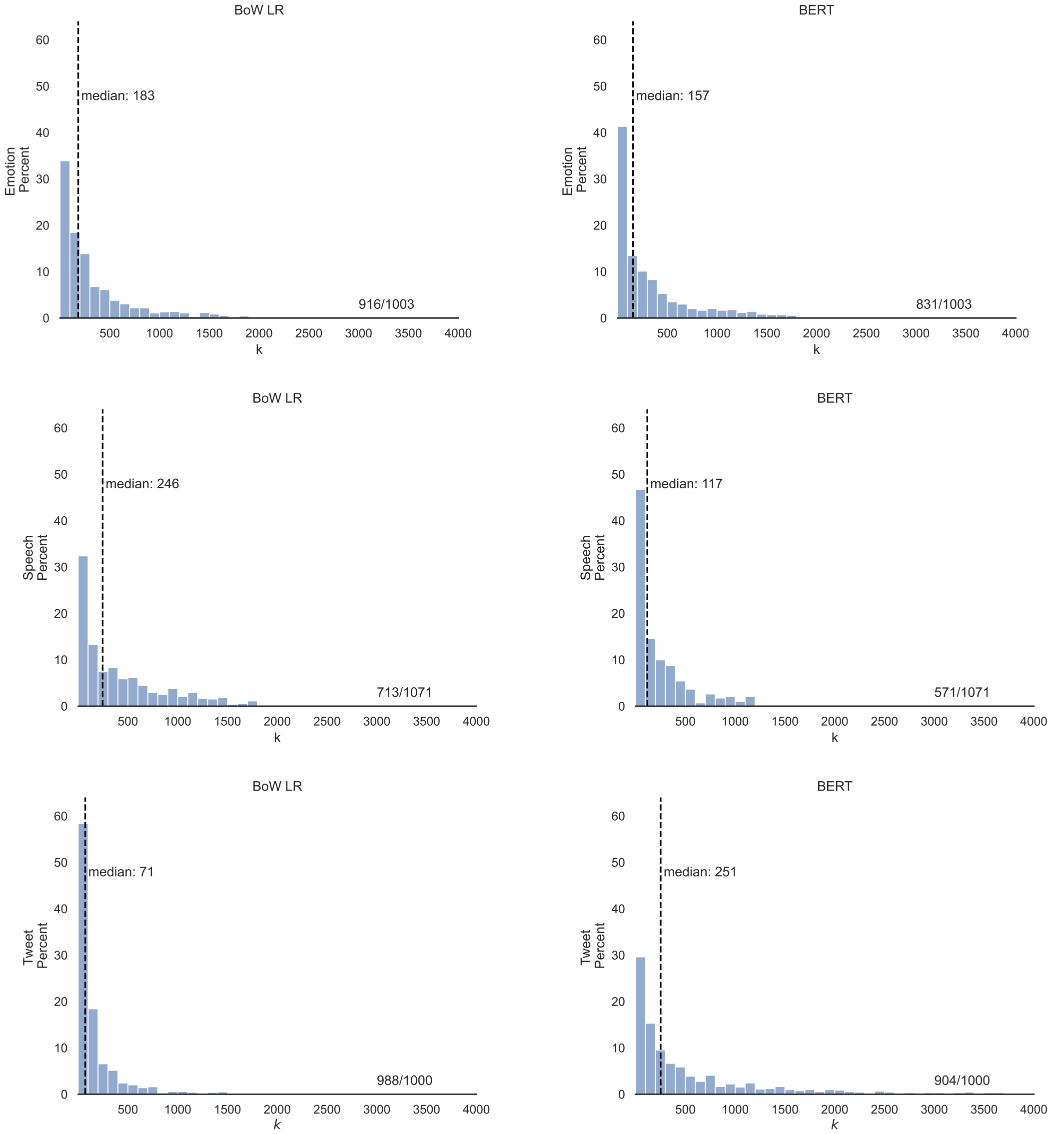


Figure A.4: Histograms of $k = |\mathcal{S}_t|$ values from Algorithm 2 over the subsets of test points x_t for which we were able to successfully identify a set of points $\mathcal{S}_t$ such that removing them would flip the prediction for $\hat{y}_t$.

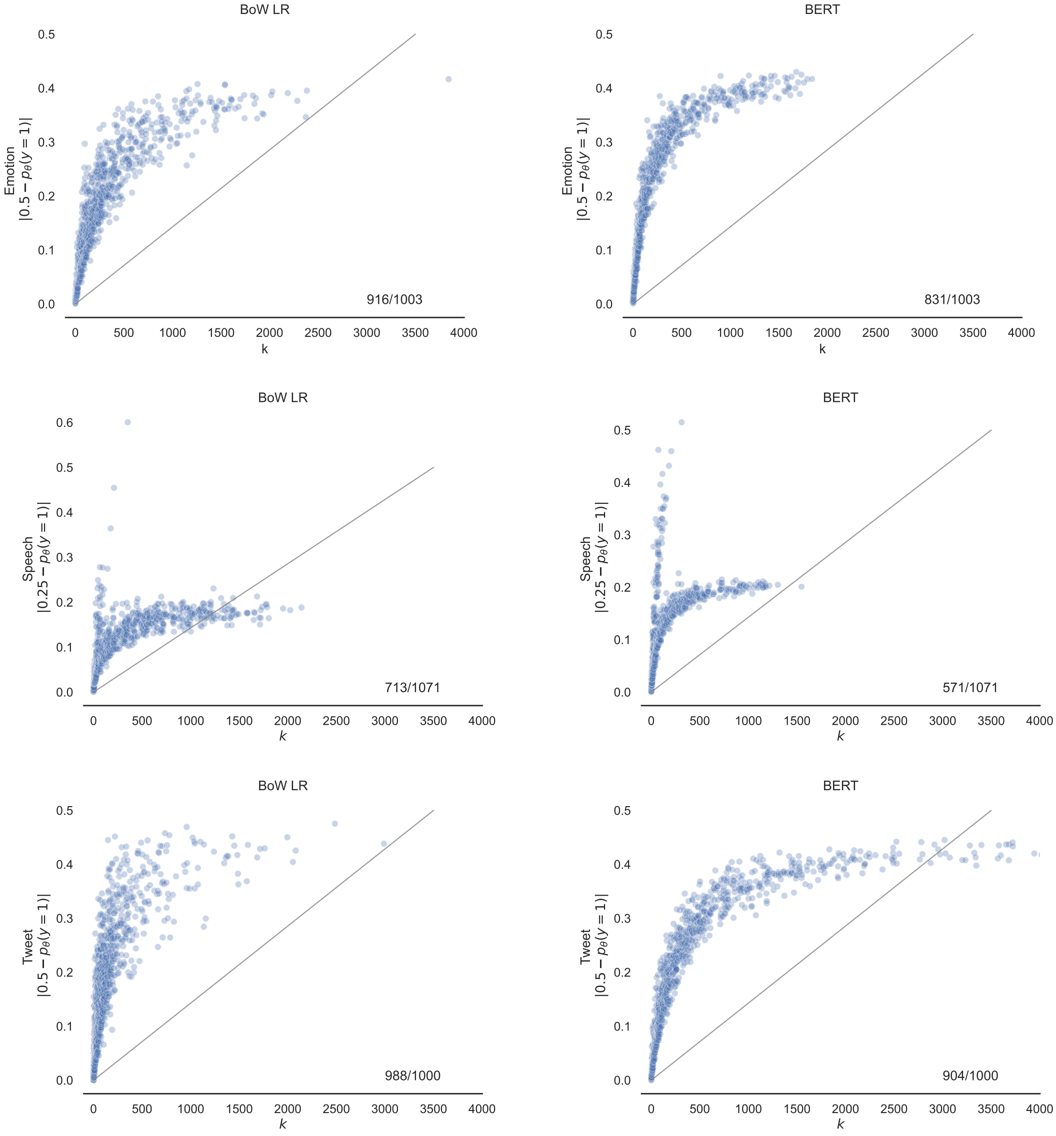


Figure A.5: Relationship between predicted probabilities and $k = |\mathcal{S}_t|$ identified from Algorithm 2.

Reinforced Sequence Training based Subjective Bias Correction

Karthic Madanagopal
Texas A&M University, Texas, USA
karthic11@tamu.edu

James Caverlee
Texas A&M University, Texas, USA
caverlee@tamu.edu

Abstract

Subjective bias is ubiquitous on news sites, social media, and knowledge resources like Wikipedia. Many existing methods for subjective bias correction have typically focused on making one-word edits and have been trained over a single (often, noisy) domain. In contrast, we propose a novel reinforced sequence training approach for robust subjective bias correction. Three of the unique characteristics of the approach are: (i) it balances bias neutralization with fluency and semantics preservation through reinforcement learning, to broaden the scope to bias beyond a single word; (ii) it is cross-trained over multiple sources of bias to be more robust to new styles of biased writing that are not seen in the training data for a single domain; and (iii) it is used to fine-tune a large pre-trained transformer model to yield state-of-the-art performance in bias text correction task. Extensive experiments show that the proposed approach results in significant improvements in subjective bias correction versus alternatives.

1 Introduction

Objective writing is essential for many important communication venues like news, encyclopedias, scientific publications, and more. And yet, bias is seemingly ubiquitous whether due to malice or unintentional habits of the writer. This subjective writing not only expresses the writer's preferences and personal interpretations, but also can influence the reader's viewpoints on the topic (Greenstein and Zhu, 2014; Beukeboom and Burgers, 2017). Hence, much like modern spelling and grammar checkers, there is a need for effective methods to detect and neutralize biased language.

The goal of this *subjective bias correction* is to rewrite a source sentence s into a neutral sentence t that is clear, objective, and stereotype-free. Furthermore, the rewritten sentence t should preserve the original meaning of the source text s. Enabling such *subjective bias correction* is difficult. Many examples of bias are challenging to detect due to their subtle nature, or through sentence framing (rather than through subjective words), or through convoluted writing intend to obscure the truth.

Examples of Biased and Neutral Statements
Acetaminophen is sold over the counter as a pain medication.
Acetaminophen is the **most dangerous** over-the-counter pain medication.
A 2013 episode of This American Life presented a number of studies that verified that acetaminophen has killed more people than any other over-the-counter pain medication.
In the This American Life episode on acetaminophen, one segment described the **tragic** death of a five-month-old baby and thus **should** convince listeners that the Federal Drug Administration (FDA) must take immediate action.

Table 1: Examples of multi-word and multi-occurrence biased statements (2 and 4), and corresponding neutral statements (1 and 3). The words "most dangerous" in the second statement and the words "tragic" and "should" in the fourth statement express the author's opinion or belief. The third statement is neutral since it describes the subjective view expressed on a radio show. (NROC, 2022)

Encouragingly, previous research has begun to make progress in identifying prejudiced or emotive language used in factual statements (Recasens et al., 2013; Bhosale et al., 2013; Misra and Basak, 2016; Hube and Fetahu, 2018; Zhong et al., 2021; Madanagopal and Caverlee, 2022) and a few studies have begun to investigate subjective bias correction (Pryzant et al., 2019; Liu et al., 2021; Zhong et al., 2021). These approaches, however, typically face a number of key challenges:

Noisy and Limited Training Data. Many approaches rely on Seq2Seq models (encoder-decoder architecture) using a Wikipedia-derived Neutrality Corpus (WNC) (Recasens et al., 2013) based on edits to Wikipedia that have been flagged with a special "neutral point of view" tag. However, we find that over 5% of revisions in the WNC (e.g., as in (Pryzant et al., 2019)) are not related

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2585–2598
May 2-6, 2023 ©2023 Association for Computational Linguistics

to bias mitigation, but rather content corrections, grammatical corrections, or typographical corrections. And more than 15% of the human-corrected neutral sentences with multi-occurrence bias still contain subjective bias. Training a bias correction approach with such noisy data can lead to models that focus on single word and single occurrence bias (missing examples like in Table 1), and result in inconsistent performance.

Training-Testing Mismatch. Second, existing approaches are primarily trained on maximizing the likelihood of each token of the target sequence (Pryzant et al., 2019; Zhong et al., 2021) by relying on the input sequence and previous ground-truth token, but testing is done on the entire input and output sequence. The training is conducted using token-level objective functions and tested using sentence-level evaluation metrics, such as BLEU. Further, although fluency and content preservation are used for evaluation, such objectives are not contained in training objectives. This training-testing criteria mismatch can lead to poor performance at inference time.

Lack of Robustness. Third, models trained on one domain tend to perform poorly on other domains, limiting their adoption. Recent studies have shown further fine-tuning of pre-trained models can improve domain generalization (Sun et al., 2020; Wang et al., 2020), but such methods generally need large volumes of training data from the target domain. Is it possible to generalize subjective bias correction models trained in one domain to multiple new domains even in the absence of rich training data?

To address these challenges, we propose a novel *reinforced sequence training* approach toward improving subjective bias correction. The overall approach first pre-trains a bias correction model using noisy training data (much like in previous approaches) but then fine-tunes the pre-trained model through reinforcement learning with a carefully designed cross-domain bias critic. This cross-domain bias critic aligns training and testing time objectives by giving equal importance to the quality of text generation and semantic content preservation through self-supervised reward-driven learning. That is, it balances bias neutralization with fluency and semantics preservation through reinforcement learning, to broaden the scope to bias beyond a single word or single occurrence within a sentence. Further, the proposed approach is cross-trained over multiple sources of subjective bias to be more robust to new styles of biased writing that are not seen in the training data for a single domain. And instead of using word-level cross-entropy loss during training, we directly optimize sentence-level task-based metrics through the policy gradient to gain significant improvement in performance.

In summary:

- We propose a reinforcement learning framework for improving subjective bias correction.
- We improve the generalizability of the model across multiple domains by adopting a cross-domain bias classifier-based reward, without the need for parallel data for domain adaptation.
- We align training and testing time objectives by using a novel reward-driven learning framework that goes beyond the traditional BLEU rewards, and guides the model to generate diverse bias-free sentences which are fluent and grammatically correct.
- We empirically demonstrate how the proposed reinforcement learning-based approach can fine-tune existing large pre-trained models like BART to perform efficient bias correction while preserving content semantics.

Through extensive experiments, we show that the proposed approach results in significant improvements in subjective bias correction versus alternatives. Furthermore, evaluations on semantic similarity and fluency benchmarks show that this bias correction approach effectively removes subjective bias while maintaining semantic information in neutralized text.

2 Related Work

Detecting Language Bias. Many studies have focused on extracting bias lexicons or opinion words (Riloff and Wiebe, 2003; Liu et al., 2005; Wiebe et al., 2005; Appling, 2017). (Recasens et al., 2013) reduced the lexical ambiguity in detecting biased statements by using a combination of language models and bias lexicons. (Misra and Basak, 2016; Hube and Fetahu, 2018) developed deep learning models to detect bias in political speeches and Wikipedia. (Madanagopal and Caverlee, 2022) combined a cross-domain dataset and a contextualized language model to train a deep model for detecting biased language.

Linguistic Bias Correction. The majority of research to address bias correction in text has focused

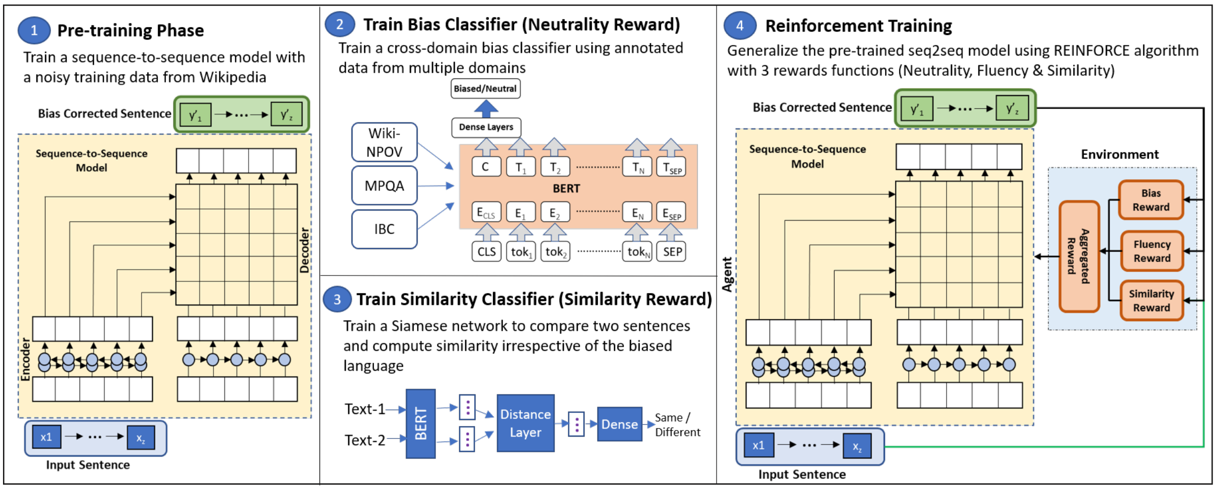


Figure 1: Illustration of the proposed Reinforced Sequence Training Approach for subjective bias correction. Task 1 is pre-training the agent (bias-correction-model) with noisy WNC data, Task 2&3 are building classifiers that will be used to compute rewards, and Task 4 is reinforced sequence training with 3 reward functions.

on demographic bias like gender bias (Manzini et al., 2019; Zhao et al., 2017, 2019; Bordia and Bowman, 2019; Wang et al., 2018). (Pryzant et al., 2019) were the first to address generic linguistic bias correction and later (Liu et al., 2021) worked on depolarizing political text. In both approaches, text segments (words or sentences) that are subjective or polarizing are first identified and then replaced with those that are semantically similar but less subjective. (Liu et al., 2021) studied generative adversarial networks to train bias correction models using non-parallel corpus. Pryzant proposed two joint embedding models to neutralize biased statements, but their research is restricted to addressing subjective bias that originated from single-word edits (Pryzant et al., 2019). Our proposed method extends from single-word to multi-word correction by generating fluent bias-free sentences.

Attribute Style Transfer. Subjective bias correction can be framed as a specialized attribute transfer task which aims at reconstructing an input text so that a linguistic attribute of interest is transferred to a desired value, such as text sentiment transfer (Jin et al., 2022; Lample et al., 2018). Most attribute transfer methods uses auto-encoders that rely on large parallel corpus (Rao and Tetreault, 2018; Briakou et al., 2021; Madaan et al., 2020; Prabhumoye et al., 2018; Pryzant et al., 2019; Jin et al., 2022). (Huang et al., 2019) used a dictionary based method to replace words in a sentence to create diverse paraphrases. (Lai et al., 2021) developed an efficient method to fine-tune a large pretrained model for text-attribute transfer on low-resource domains. (Chen et al., 2018) used cross-aligned auto-encoder trained on opposite ideology news data to generate flipped titles. Our proposed method is inspired from various attribute style transfer methods, but addresses the problem of using noisy parallel data for supervised style transfer for better generalization using reinforcement learning methods.

3 Reinforced Bias Corrector

In this section, we introduce the design of the proposed subjective bias correction approach (illustrated in Figure 1). Given a biased statement x that describes a facts f using a set of words $(x_1, x_1, ..., x_N)$, where N is the length of x. The aim of a bias correction system is to generate a statement $y = (y_1, y_2, ..., y_M)$, where M is the length of y. The generated statement y is not only expected to have a neutral tone, but also preserve the original semantic content expressed in statement x. Additionally, y is also expected to present the fact F in a grammatically correct and fluent language that is easy for a reader to comprehend.

3.1 RL and Bias Correction

Given these goals, a natural approach to perform subjective bias correction is to train an encoder-decoder with attention network using a parallel corpora similar to many natural language generation tasks (Bahdanau et al., 2014; Zhao et al., 2019; Chollampatt and Ng, 2018; Pryzant et al., 2019; Konstas et al., 2017; Li et al., 2018; Gupta et al., 2018; Vaswani et al., 2017). The encoder converts the input sentence into a fixed-length vector that will be used by the decoder to generate the output sequence, word by word. The entire network will be trained with the parallel data (x, y) to minimize the cross-entropy loss (CE) that is given by:

$$L_{CE} = \sum_{t=1}^{T} log\pi_\theta(y_t|y_{t-1}, s_t, c_{t-1}, x) \quad (1)$$

Since minimizing L_{CE} makes the model generate text close to the human-reference edits, the training data needs to be of high-quality. Since the only large parallel corpus available for bias correction is noisy, such likelihood optimized models have limited performance (Pryzant et al., 2019; Zhong et al., 2021). Additionally, these models do not produce the best results for metrics during evaluation because the training is not optimized to generate text that addresses evaluation metrics (such as BLEU or Content Similarity). By aligning the training and testing objective, we can improve the performance of bias correction in a source domain (Wikipedia), though this does not guarantee generalizability over other domains.

Hence, we propose to improve the performance and generalizability of bias correction models by fine-tuning the Seq2Seq model using Reinforcement Learning (RL). In this method, instead of maximizing the probability that the predicted word is close to the ground truth, the parameters of the agent are optimized to maximize long-term rewards. We can formulate the RL engine for bias correction as composed of an agent (A), action (a), policy (pi), and reward (R). The agent is our bias correction model with parameters θ that observes the current state (encoder's output) at time t and takes an action a (predict the next word $\hat{y}_t$) by using a policy (π). The reward (R) is a feedback returned to the bias correction model from the system by evaluating the quality of generated text.

3.2 REINFORCE Training

In reinforced bias correction task, the agent's behavior is controlled primarily through the neutrality score that measures the degree of subjective bias in the generated text $\hat{y}$. So, the objective of maximizing the expected scalar reward $R : \hat{y} \rightarrow [0, 1]$ is given by:

$$J(\theta) = E_{\pi_\theta}(\hat{y}/x)[R(\hat{y})] \quad (2)$$

Since the reward is the discrete function of the model's output, the RL objective is non-differentiable with respect to the model parameter θ,), which makes it difficult to back-propagate the error signals from the critic to the generator. This issue can be addressed through policy gradient or Q-learning based methods. In our initial experiments, we considered several RL methods, including Q-learning, but these models did not converge even after training for a long time (more than 48 hours). This is mainly due to the extremely large action space $O(W^T)$, where W is the number of words in the vocabulary (10^4) and T is the sentence length.

In contrast, the REINFORCE-based method showed good performance and converged faster than other methods. Hence, we focus our discussion here on REINFORCE as our primary policy gradient method. In REINFORCE, the expected reward is approximated using a sampling method and the model is trained using stochastic gradient ascent (Williams, 1992), which can be formulated as:

$$\nabla_\theta J(\theta) = E_{\pi_\theta(\hat{y}/x))}[R(\hat{y})\nabla_\theta log\pi_\theta(\hat{y}/x)] \quad (3)$$

where, π_θ is a policy that generates a probability of picking a word as output. Even though RL training is promising, efficiently applying it for real world problems remains a challenge due to extremely large sequence space (50000^{50} for a sentence with length of 50 and vocabulary of 50000) (Mnih et al., 2015). To achieve the RL objective, a sequence model is pre-trained first and then fine-tuned using REINFORCE algorithm using different rewards. The REINFORCE method optimizes the best policy directly by modifying the parameters of the model based on the observed rewards. Since it directly optimizes the return, it tends to be more stable in converging to a good behavior

4 RL Rewards and Policy Extensions

In our subjective bias correction task, the key testing objectives are: (i) reduce subjective bias; (ii) preserve content expressed in the source text; and (iii) the generated text needs to be grammatically correct fluent. Based on the above objectives, the designed reward function $R(x, \hat{y})$ consist of three rewards:

$$R(x, \hat{y}) = \alpha * R_N(\hat{y}) + \beta * R_S(x, \hat{y}) + \gamma * R_F(\hat{y}) \quad (4)$$

where, $R_N(\hat{y})$ is the neutrality reward for the bias corrected text $\hat{y}$, $R_S(x, \hat{y})$ is the semantic similarity reward computed between the input text x and output text $\hat{y}$, and $R_F(\hat{y})$ is the fluency reward for the output text $\hat{y}$. Here α, β, $\gamma > 0$ and represents weights for the respective rewards.

Neutrality Reward. The central objective of this first reward is to remove any subjective bias that is contained in the input sentence x. The neutrality reward $R_N(\hat{y})$ computes the presence or absence of biased tone in the output sentence $\hat{y}$. It is not a relative score that is computed between the input text x and output text $\hat{y}$, rather an absolute score that is computed purely on the output text $\hat{y}$. Because of which, there is a possibility that the model might remove a chunk of text that is biased and output a partial sentence that is neutral. But the combination of neutrality reward R_N and semantic similarity reward R_S ensures the corrected text carries all the information expressed in an unbiased or neutral tone. To compute the neutrality reward, a RoBERTa based binary classifier is used that takes the sentence as input and produces a probability score that the input text is biased or not (Madanagopal and Caverlee, 2022). More details about this classifier is available on our previous work on detecting linguistic bias (Madanagopal and Caverlee, 2022).

Semantics Similarity Reward. The semantic similarity reward $R_S(x, \hat{y})$ is used to ensure that all the information that is expressed in the input sentence is retained in the output sentence $\hat{y}$. Standard semantic similarity metrics like BERTScore cannot be used for this task, because the tone or slant change will affect the similarity score significantly. To create a subjectivity-free similarity reward, we created a Siamese network based on BERT and fine tuned using Wikipedia-derived parallel corpus (Mueller and Thyagarajan, 2016).

Fluency Reward. The fluency reward encourages the model to generate a sentence $\hat{y}$ that is grammatically correct or natural sounding. There are various methods to compute the grammatical correctness of a sentence such as Perplexity score (PPL) (Meister and Cotterell, 2021). Large pretrained models are used to compute the PPL scores, where grammatically incorrect sentences will have high PPL and grammatically correct sentence will yield a lower PPL score (Meister and Cotterell, 2021). But, the fluency reward needs to be a normalized value between 0 and 1. We used the Corpus of Linguistic Acceptability (CoLA) that contains 10,657 English sentences that are labelled for grammatical and syntactic correctness (Warstadt et al., 2018). We then built a binary classifier using RoBERTa and used the classifier result as the fluency reward for measuring the correctness of the generated text $\hat{Y}$.

4.1 Cross-domain linguistic bias critic

In classic REINFORCE learning scenarios, to minimize the distinction between human reference edit and the machine correction, the bias critic is trained with the same training data used for agent's pretraining (Wu et al., 2018). Since some of the ground-truth data still contains bias, using similar approach can adversely affect the agent's behaviour. Also, our objective is to learn beyond what is available in the training corpus so that it can efficiently handle new style of writing and all forms of subjective bias outside of Wikipedia domain. Hence, we investigate in this paper, the potential of using a cross-domain bias classifier as the bias critic. The cross-domain bias classifier is trained by leveraging annotated datasets from other domains that are rich in subjectivity and apply recent deep transformer models like BERT in order to more robustly model factual statements. The cross-domain bias classifier is evaluated to have an accuracy of 89% and coincide with human judgement for bias detection. There are two advantages to using a cross-domain bias classifier: First, the cross-domain dataset used for training the binary classifier does not need a parallel dataset. Second, the cross-domain aspect lets the classifier learn new subjective writing styles from other subjectivity rich domains and assign rewards that are of high quality, which in turn makes the reinforcement more efficient.

Generally, the most effective method to approximate the policy gradient is either multinomial sampling of the softmax-normalized outputs or beam search. Both the objectives can be trained either sequentially or simultaneously. Our proposed method uses a sequential training, where the agent is first trained with supervised methods and then fine-tuned with reinforced methods.

4.2 Delayed and Sparse Reward

Typically, in a RL setting, rewards are assigned as soon as the action is taken by the agent. But in our method, the reward is observed only at the end of generating a complete sentence. Which means one reward for multiple actions taken by the agent. Due to this sparsity of reward, the agent receives same reward for all steps taken to generate a complete sentence. This makes the RL training inefficient because the agent has no clue which action led to bad results. To address this issues, Reward shaping strategy (Ng et al., 1999) is used to compute intermediate rewards as suggested by

(Bahdanau et al., 2014):

$$R(y_t) = R(y_{1:t}) - R(y_{1:t-1}) \qquad (5)$$

For simple rewards like BLEU, it is easy to compute the intermediate rewards with partial sentences. But the reward functions used in the proposed approach only works if the complete sentence is available. We used the modified reward shaping method used by Hongyu et al. (Gong et al., 2019), where the agent is made to generate the entire text at each step and the generated text is used to compute the reward.

5 Experimental Setup

This sections provide an overview of the datasets, the baseline bias correction models, the evaluation metrics, and the details of their implementation.

5.1 Datasets

For pre-training the agent (Seq2Seq or Transformer), we harvested Wikipedia-derived Neutrality Corpus (WNC) (204k parallel sentences) following the instructions of (Recasens et al., 2013), but included multi-word edits (See the Appendix). For training a cross-domain bias classifier, we used the dataset assembled in a prior study (Madanagopal and Caverlee, 2022). For evaluation, we used WIKIBIAS (2,117 biased and 2,911 neutral), a high-quality dataset that is manually curated by (Zhong et al., 2021).

5.2 Baselines

We consider three baselines models:

Delete Biased Word(s) uses a bias tagger to identify subjective words in a sentence and removes them. More details on the bias tagger is available in the Appendix.

Join Embedding Model uses a denoising autoencoder and a token-weighted loss function to automatically neutralize subjective bias in text (Pryzant et al., 2019). We retrained their best performing CONCURRENT model.

OpenNMT Transformer is a widely used medium transformer model implemented in the OpenNMT-py library that contains 12 heads, 768 d_{model} size and 3072 d_{ff} size (Klein et al., 2017).

5.3 Models

We consider two classes of models for the proposed *reinforced sequence training*: (i) a vanilla Seq2Seq model; and (ii) a pre-trained transformer model.

Reinforced Seq2Seq Model (RL-Seq2Seq): The first model is an attention-based Seq2Seq model similar to (Bahdanau et al., 2014). It is pretrained with the WNC corpus and the pre-trained model is used as an agent in the proposed reinforced sequence training approach with three reward functions. The following three variations were trained and evaluated:

RL-Seq2Seq$_B$: Reinforced sequence-to-sequence model with base bias reward (WNC).

RL-Seq2Seq$_{CDB}$: Reinforced sequence-to-sequence model with cross-domain bias reward.

RL-Seq2Seq$_{CDB+FL+SIM}$: Reinforced sequence-to-sequence model with cross-domain bias, fluency and semantic content preservation reward.

Reinforced Transformer Model (RL-Trans): The second model is the BART pre-trained transformer model (Lewis et al., 2019), which performs well on paraphrasing tasks with good fluency and content preservation (Lewis et al., 2019; Lai et al., 2021). We downloaded a pre-trained BART model (base) and further pre-trained on the noisy WNC corpus. The further pre-trained BART model is used as one of the baseline models, and further fine-tuned, resulting in three variations akin to the Seq2Seq ones: (RL-Trans$_B$, RL-Trans$_{CDB}$, RL-Trans$_{CDB+FL+SIM}$).

5.4 Training

Training is in four steps: (1) pre-train the Seq2Seq model using the noisy Wikipedia NPOV data with an MLE objective; (2) train a cross-domain bias classifier using a large pre-trained language model like RoBERTa; (3) train a siamese network to compute semantic similarity between sentences without subjective tone; and (4) fine-tune the pre-trained Seq2Seq model using reinforcement learning with three reward components. A combination of training data sampling and new domain data (non-parallel) is used for reinforced fine-tuning. For fluency reward, we used the CoLA dataset (Warstadt et al., 2018) and trained a RoBERTa model with an accuracy of 92%. The hyperparameters used for pre-training the Seq2Seq model and fine-tuning of the reinforcement model are presented in the Appendix. For semantic similarity evaluation, the BERTScore model is downloaded from HuggingFace.[1] All code is implemented using PyTorch and

---

[1] https://huggingface.co/spaces/evaluate-metric/bertscore

Model	Neutrality↑	BLEU↑	BLEURT↑	PPL↓	BERT-Score↑	Bias↓	Fluency↑	Content↓
Source Copy	9.47	90.83	32.47	33.47	100.00	-	-	-
Biased word removal	38.19*	92.03*	36.24*	49.21*	85.24*	-0.179*	-0.127*	1.28*
Join Embedding	45.80*	93.94*	52.92*	31.87*	87.58*	-0.213*	0.011*	1.23*
OpenNMT Transformer	49.47*	87.82*	61.24*	34.57*	88.45	-0.205*	0.034*	1.14*
Seq2Seq	45.14*	93.23*	54.27*	39.47*	86.14*	-0.217*	0.008*	1.29*
$RL - Seq2Seq_{CDB+FL+SIM}$	63.72*	**94.52**	61.27*	26.62*	92.97*	-0.475*	0.013*	**1.06***
Transformer	52.51*	91.17*	60.84	22.78*	83.23*	-0.418*	**0.169***	1.12*
$RL - Trans_{CDB+FL+SIM}$	**66.48***	89.68*	**66.66***	**23.97**	**94.65***	**-0.492***	0.164*	1.19*
Target Copy	92.14	100	100	31.68	98.44	-	-	-

Table 2: Bias neutralization performance. RL-Seq2Seq and RL-Trans models are trained using our reinforced sequence training approach. For quantitative metrics, rows with asterisks (*) are significantly different than the preceding row. ↑ / ↓ means higher/lower score is preferred for the corresponding metric.

Model	Neutrality↑	BLEU↑	BLEURT↑	PPL↓	BERT-Score↑
$RL - Seq2Seq_{B}$	59.08	94.08	59.24	32.58	87.74
$RL - Seq2Seq_{CDB}$	63.65	94.23	61.24	31.24	85.23
$RL - Seq2Seq_{CDB+FL+SIM}$	63.72	**94.52**	61.27	26.62	92.97
$RL - Trans_{B}$	61.16	89.37	66.51	22.89	84.50
$RL - Trans_{CDB}$	**66.51**	89.11	65.84	22.57	82.21
$RL - Trans_{CDB+FL+SIM}$	66.48	89.68	**66.66**	**23.97**	**94.65**

Table 3: Performance comparison Reinforced sequence models on different reward schemes. ↑ / ↓ means higher/lower score is preferred for the corresponding metric.

Model	News Articles			Academics			Conservapedia		
	Bias↓	Fluency↑	Content↓	Bias↓	Fluency↑	Content↓	Bias↓	Fluency↑	Content↓
Delete Biased	-0.39	-0.77	1.6	-0.41	-0.13	1.53	0.25	-0.35	1.22
Join Embedding	-0.015	0.11	1.58	-0.93	-0.73	2.12	-0.31	-0.37	1.25
OpenNMT Transformer	-0.07	0.05	1.64	-0.61	0.48	1.65	0.08	-0.38	1.69
$RL - Seq2Seq_{CDB+FL+SIM}$	**-1.38**	0.59	**1.14**	-0.92	0.41	1.45	-0.42	0.43	**1.09**
$RL - Trans_{CDB+FL+SIM}$	-0.96	**0.72**	1.33	**-1.01**	**0.55**	**1.11**	**-0.63**	**0.56**	1.22

Table 4: Human evaluation of subjective bias correction performance across 3 different domains.

trained on a Google Cloud Platform with NVIDIA Tesla P100 GPU.

5.5 Evaluation Metrics

The following set of automated metrics are used to evaluate the quality of bias correction models:

Neutrality: Similar to previous work (Luo et al., 2019; He et al., 2020), the neutrality score is computed using a binary classifier that is pre-trained to evaluate subjective bias across various domains; on human reference dataset it has an accuracy of 89% (Madanagopal and Caverlee, 2022). The text generated by each model is sent to the binary classifier and the probability score that it belongs to the neutral category is used as the neutrality score.

BLEU: A commonly used metric that measures the similarity of machine corrected text and human reference correction through n-gram precision counting (Papineni et al., 2002).

BLEURT: A trained metric that uses transformer models to evaluate the quality of natural language generation models (Sellam et al., 2020).

BERTScore: Uses contextual language models such as BERT and computes semantic distance between candidate and reference sentences (Zhang* et al., 2020).

PPL: To evaluate the grammatical correctness and fluency of the machine generated text, we computed the perplexity score (PPL) using the large pre-trained language model GPT-2. The perplexity score PPL is computed directly on the generated text with no reference text.

6 Experimental Results

We present the results of the experiments through both automated and human judgement evaluation.

6.1 Automated Evaluation

Table 2 shows our proposed model's performance in comparison with the selected baseline models. Among the baseline models, the OpenNMT transformer model performed better with a neutrality score of 49.47 and BERTScore of 88.45, and the Join Embedding model had the best BLEU because of its copy network nature. But our two proposed reinforced sequence models performed significantly better than the baseline models on all metrics, especially the neutrality score (18% increase). In terms of BLEU score, the RL-Seq2Seq model had the best score of 94.52. Since the RL-Trans generated more diverse text compared to the human-edits, it's BLEU score was relatively low. But its BERTScore being high confirmed the RL-Trans model's ability to retain the information conveyed in the source sentence to the target sentence with high neutrality score. Similarly the perplexity score for the RL-Trans model indicates high fluency. This might be due to the fact the pre-trained BART model is trained on large diverse corpus. Overall the RL-Trans had the best performance on all automated metrics except BLEU.

6.2 Evaluation of Reward Function

To understand the effect of each reward on the performance of the bias correction model, we conducted an ablation study as shown in Table 3. Base bias rewards are based on a bias classifier trained solely on Wikipedia dataset, while cross-domain bias rewards are trained on a dataset containing biased statements from various domains like political speech and product reviews. In terms of the neutrality score, we observed significant improvement in the bias correction performance by the use of cross-domain bias reward. It could be because pre-training data is noisy, so the output still contains biased statements that the cross-domain bias classifier is able to detect more subtle forms of bias and multiple occurrences of bias. In terms of BLEU, the RL-Seq2Seq model with all rewards performed much better. The RL-Trans model's BLEU score was low which is due to the diversity in the text generated relative to the source text. This shows BLEU is not a good metric for comparing bias correction accuracy. So we use BLEURT which accounts for the n-gram accuracy with embeddings. The reinforced transformer model with all rewards performed the best. One interesting observation is that the improvement in RL-Trans using fluency and content similarity reward is small relative to RL-Seq2Seq. The large pre-trained BART model already performs well on fluency and content preservation, so adding rewards has little impact. All in all, the rewards were more effective at achieving the testing objective and also produced high-quality text.

6.3 Multi-Occurrence Bias Evaluation

We performed a separate study to investigate the performance of reinforced bias correction in addressing multiple instances of bias within a single sentence. A set of 331 sentences were selected from the WIKIBIAS corpus that contain more than one instance of subjective bias. The output of each model is then analyzed with the help of a bias tagger to understand how much of the bias still exists or if new bias is introduced. RL-Seq2Seq model trained with all 3 rewards had the best performance (16%) of addressing multiple occurrence of bias in a single sentence (See Table 5).

Model	**Neutrality**	**% biased**
Biased Word Removal	34.06	30.85
Join Embedding	50.56	37.32
OpenNMT	58.85	41.62
RL-Seq2Seq	67.33	**16.17**
RL-Trans	**67.15**	18.24

Table 5: Performance evaluation of addressing multi-occurrence bias. % biased represents the percentage of sentences that contains at least one instance of biased chunk after bias correction (lower the better).

6.4 Human Judgement

With a sample of 100 sentences collected for each of the experimented models, we conducted a human evaluation study with 10 judges to determine if the bias correction models generate human-like sentences. See the Appendix for more detail. A majority of the human judgement results correlated with the automatic evaluation. The Reinforced-Transformer model was preferred by a majority of the judges for its neutral tone and language fluency. The Reinforced Seq2Seq model was shown to preserve the same information as the source text. Since the pre-trained transformer model generates text that is diverse from the original input sentence, it might have read slightly different for a human. But the BERTScore showed the Reinforced-Transformer model to have better performance in content preservation.

The Kappa statistic was used to compute the inter-annotator agreement for human evaluation task. The average kappa values for the individual aspects are (i) Bias neutralization: 0.72; (ii) content presentation: 0.75; and (iii) fluency: 0.85. Due to the complex nature of subjective bias detection, the inter-annotator agreement for that task is the lowest (0.72). On the other hand, Since most of the annotators are well-versed in assessing the grammatical correctness of a sentence, the kappa value for fluency was high (0.85).

6.5 Cross-domain Model Performance

Finally, we explore the generalizability of the proposed bias correction model by performing inferences on three datasets that are manually curated outside the training domain (Wikipedia): (1) News Headlines (Media Cloud[1]); (2) Academic (National Geographic[2]); and (3) Conservapedia[3]. Thirty factual sentences were collected from all three selected domains, processed through the bias correction models and evaluated through human judgement on the basis of neutrality, fluency and content preservation. Overall, both the proposed models performed very well in all three domains. In terms of neutralization effort, the RL-Seq2Seq model performed the best on the News domain (See Table 4). This is because most of the bias in news are related to framing bias (perspective-specific words). But in the Academic domain and Conservapedia domain, the RL-Trans performed better. Relatively, the transformer model performs better on addressing both framing as well as epistemological bias.

7 Conclusion and Future Work

We proposed a hybrid method to improve the performance of Seq2Seq and transformer-based bias correction models by incorporating semi-supervised training strategy that uses a supervised pre-training using noisy data and reinforced fine-tuning using high-quality cross-domain data. The proposed training method is able to successfully alleviate the exposure bias in MLE optimized sequence models and address the in-stability issue in reinforcement learning methods. It also shows that by carefully designing the reward function with respect to the testing objective, high quality results can be obtained. Specifically, our method was able to generate text with high neutrality. The text generated by our method is more fluent and retains more semantic information relative to previous methods. In this study, we explored only a limited range of rewards using simple aggregations, but a greater range of rewards and scaling could be explored for better domain adaptation.

Limitations

Although the proposed Reinforced Seq2Seq framework presents an intriguing framework for automatically correcting subjective bias, we also find that – like previous research (Williams, 1992) on similar reinforcement-style learning regimens – it has the following limitations: (i) after each policy run, the trajectories are discarded by the update process, making it inefficient. In some cases, the collected trajectory may not accurately represent the policy, so the gradient estimate becomes uncertain. Further, whether a trajectory reinforces good or bad actions depends entirely on the final output, so the reward assignment can be unclear.

In this study, we examine a specific form of subjective bias that manifests at the sentence level. However, taking into account the context (paragraph) of the statement could change the viewpoint. Although this study evaluates the Reinforced Seq2Seq framework on a real-world subjective bias correction task, further testing is needed in order to include more challenging bias types, as well as other target model architectures.

The work presented here offers a promising approach for improving bias correction models using reinforcement learning methods, a field of high impact but under-explored to date. We hope that evaluating the models across different domains will inspire further work on building robust, nuanced, and fair bias correction models.

References

CJ Hutto Dennis Folds Scott Appling. 2017. Computationally detecting and quantifying the degree of bias in sentence-level text of news stories.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Camiel J Beukeboom and Christian Burgers. 2017. Linguistic bias. In *Oxford research encyclopedia of communication*.

Shruti Bhosale, Heath Vinicombe, and Raymond Mooney. 2013. Detecting promotional content in wikipedia. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1851–1857.

Shikha Bordia and Samuel R Bowman. 2019. Identifying and reducing gender bias in word-level language models. *arXiv preprint arXiv:1904.03035*.

---

[1] https://mediacloud.org/

[2] https://education.nationalgeographic.org/

[3] https://www.conservapedia.com

Eleftheria Briakou, Di Lu, Ke Zhang, and Joel Tetreault. 2021. Olá, bonjour, salve! xformal: A benchmark for multilingual formality style transfer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3199–3216.

Wei-Fan Chen, Henning Wachsmuth, Khalid Al Khatib, and Benno Stein. 2018. Learning to flip the bias of news headlines. In *Proceedings of the 11th International conference on natural language generation*, pages 79–88.

Shamil Chollampatt and Hwee Tou Ng. 2018. Neural quality estimation of grammatical error correction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2528–2539.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Hongyu Gong, Suma Bhat, Lingfei Wu, JinJun Xiong, and Wen-mei Hwu. 2019. Reinforcement learning based text style transfer without parallel training corpus. *arXiv preprint arXiv:1903.10671*.

Shane Greenstein and Feng Zhu. 2014. *Do Experts Or Collective Intelligence Write with More Bias?: Evidence from Encyclopædia Britannica and Wikipedia*. Harvard Business School.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the aaai conference on artificial intelligence*, volume 32.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. *arXiv preprint arXiv:2002.03912*.

Shaohan Huang, Yu Wu, Furu Wei, and Zhongzhi Luan. 2019. Dictionary-guided editing networks for paraphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6546–6553.

Christoph Hube and Besnik Fetahu. 2018. Detecting biased statements in wikipedia. In *Companion Proceedings of the The Web Conference 2018*, pages 1779–1786.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. *arXiv preprint arXiv:1704.08381*.

Huiyuan Lai, Antonio Toral, and Malvina Nissim. 2021. Thank you bart! rewarding pre-trained models improves formality style transfer. *arXiv preprint arXiv:2105.06947*.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351.

Ruibo Liu, Chenyan Jia, and Soroush Vosoughi. 2021. A transformer-based framework for neutralizing and reversing the political polarity of news articles. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW1):1–26.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. A dual reinforcement learning framework for unsupervised text style transfer. *arXiv preprint arXiv:1905.10060*.

Aman Madaan, Amrith Setlur, Tanmay Parekh, Barnabas Poczos, Graham Neubig, Yiming Yang, Ruslan Salakhutdinov, Alan W Black, and Shrimai Prabhumoye. 2020. Politeness transfer: A tag and generate approach. *arXiv preprint arXiv:2004.14257*.

Karthic Madanagopal and James Caverlee. 2022. Improving linguistic bias detection in wikipedia using cross-domain adaptive pre-training. In *Companion Proceedings of the Web Conference 2022*, pages 1301–1309.

Thomas Manzini, Yao Chong Lim, Yulia Tsvetkov, and Alan W Black. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. *arXiv preprint arXiv:1904.04047*.

Clara Meister and Ryan Cotterell. 2021. Language model evaluation beyond perplexity. *arXiv preprint arXiv:2106.00085*.

Arkajyoti Misra and Sanjib Basak. 2016. Political bias analysis.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Andrew Y Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287.

NROC. 2022. Recognizing objective and subjective language. *NROC developmental English foundations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. *arXiv preprint arXiv:1804.09000*.

Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2019. Automatically neutralizing subjective bias in text. *arXiv preprint arXiv:1911.09709*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. *arXiv preprint arXiv:1803.06535*.

Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2013. Linguistic models for analyzing and detecting biased language. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1659.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 105–112.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. 2020. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. *Advances in neural information processing systems*, 28.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A Olshausen, and Trevor Darrell. 2020. Fully test-time adaptation by entropy minimization.

Tianlu Wang, Jieyu Zhao, Kai-Wei Chang, Mark Yatskar, and Vicente Ordonez. 2018. Adversarial removal of gender from deep image representations. *arXiv preprint arXiv:1811.08489*, 3.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Lijun Wu, Yingce Xia, Fei Tian, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. Adversarial neural machine translation. In *Asian Conference on Machine Learning*, pages 534–549. PMLR.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. *arXiv preprint arXiv:1707.09457*.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. *arXiv preprint arXiv:1903.00138*.

Yang Zhong, Jingfeng Yang, Wei Xu, and Diyi Yang. 2021. Wikibias: Detecting multi-span subjective biases in language. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1799–1814.

Appendix A. Data Pre-processing Details

We developed a parallel corpus for bias correction by analyzing Wikipedia articles histories with Neutral Point of View (NPOV) tags. Our data harvesting approach was similar to Recasens et. al (Recasens et al., 2013) and Pryzant et. al (Pryzant et al., 2019), but with some minor changes. Following (Recasens et al., 2013), we analyzed the revision histories for each Wikipedia article and downloaded sentences that were argued for NPOV issues. Sentences that had NPOV tags before the revision were considered as biased sentences and the sentences whose NPOV tags were removed after edits were considered as unbiased sentences. We ignored revisions that were related to missing references, misspellings and punctuation.

Corpus Name	Biased	Unbiased	Total
NPOV	32,541	75,024	107,565
MPQA	8,575	42,282	50,857
IBC	3,726	600	4,062

Table 6: Corpus statistics for each dataset used for training cross-domain bias classifier.

From the original NPOV corpus, we extracted sentences that had NPOV or peacock tags in their content before the edit. Pryzant et. al only considered single word edits for their bias correction model. Since the objective of this research is to correct bias that is induced by single word and multiword, we expanded the corpus by modifying the harvest function of pryzant et. al. Also, our method uses the latest dump from Wikipedia which contains new biased sentences that were not considered in the previous study. Additionally, some data cleanups were done to make this model not sensitive to noun phrases in the text. We replaced all noun phrases, but retained honorifics because some of the gender biases were introduced through honorifics. The numbers mentioned in the text were also replaced with NUM tag. A total of 408,738 sentences were extracted for our study. The NPOV corpus will help our bias detection model to learn common patterns that are used by Wikipedia editors for imposing subjective views. The details of the dataset are provided at Table 6.

Appendix B. Implementation

For Seq2Seq model, a multi-layer recurrent neural network based encoder, and an attention-based decoder was used(Vinyals et al., 2015; Bahdanau et al., 2014). Both the models are designed with 3 LSTM layers with 256 units at each layer. The hyper-parameters are tuned based on the noisy Wikipedia data. The pre-trained GloVe embedding is used as the embedding layer. The model is trained with Adam optimizer with the following hyper-parameters batch size = 32, learning rate = $5e^{-}5$ and a learning rate decay factor = 0.99. Based on the training and testing data statistics, the maximum sentence length is set to 30 for both input and output. Glove vectors is used as the pre-trained embedding. For pre-trained transformer model, we used the BART model (Lewis et al., 2019) with 139M parameters and fine tuned with Adam optimizer (learning rate $3e - 5$).

Algorithm 1 REINFORCE Algorithm

```
Input: Input sequences (X), the output sequences (Y), and a pre-trained policy ($\theta$)
Output: Trained policy with REINFORCE
Training Steps:
  while not converged do
      Select a batch of size N from X and Y
      Sample N full sequence of actions: $\{y_1, ..., y_M\}$
      Observe the sequence reward and calculate the baseline $r_b$.
      Calculate the loss
      Update the parameters of network
  end while
Testing Steps:
  for batch of input and output sequences X and Y do
      Use the trained model and sample the output
      Evaluate the model using a performance metric, e.g. BLEU
  end for
```

Appendix C. Bias Tagger

The bias tagger takes an input sentence and identifies a sequence of words as biased based on the context of use. Some of the subjective bias correction methods we developed needs additional information like biased words in the sentence to efficiently rewrite a polarized sentence, so a sequence tagging based bias tagger model was developed. By comparing the word edits between the sentences in parallel text generated for bias correction, we constructed a bias tagging dataset using a BIO format. Since state of the art sequence tagging models are BERT based (Devlin et al., 2018), we developed a BERT based sequence tagger by adding a dropout layer and a classification layer at the end of BERT model with a cross entropy loss. Pretrained RoBERTa model (Liu et al., 2019) was downloaded and fine-tuned for bias tagging. The bias tagger model had an accuracy of 95% with a recall of 92% on the validation set.

Appendix D. Policy Gradient

The goal of policy gradient methods is to update the probability distributions of actions in such a way that actions with higher expected rewards have a higher probability value for an observed state. The objective function for policy gradients is given by:

$$J(\theta) = E[\sum_{t=0}^{T-1} r_{t+1}] \tag{6}$$

where r_{t+1} is the reward received by performing action a_t at state s_t; $r_{t+1} = R(s_t, a_t)$, where R is the reward function. The policy is optimized by taking the gradient ascent based on the partial derivative of the objective based on the policy parameter theta:

$$\theta \leftharpoonup \theta + \frac{\partial}{\partial \theta} J(\theta) \tag{7}$$

The objective function can be expanded as:

$$J(\theta) = E[\sum_{t=0}^{T-1} r_{t+1} | \pi_\theta] \tag{8}$$

If $P(s_t, a_t|\tau)$ represents the probability of s_t, a_t occurring given the trajectory τ, then objective function is:

$$J(\theta) = \sum_{t=i}^{T-1} P(s_t, a_t|\tau) r_{t+1} \tag{9}$$

Then the policy gradient is given as:

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T-1} \nabla_\theta log \pi_\theta(a_t, s_t) \sum_{t'=t+1}^{T} \gamma^{t'-t-1} r_{t'} \tag{10}$$

where $\gamma \epsilon [0, 1]$ is the discount factor that helps to weight immediate rewards more than future rewards.

Appendix E. Human Judgement

A total of 100 sentences were collected from the model output and evaluated using 10 judges. The sentence selected for human judgement contains both biased and unbiased statements.Additionally, we ensured that the biased sentences collected contained both multi-occurrence bias as well as all three types of bias. The judges selected for this evaluation has computational linguistic background, but they don't have background information on what models were developed and their characteristics. Before presenting the data, a detailed definition of what is subjective bias with examples was presented to the user. For each example, a detailed note is provided to explain why the selected sentence is biased. Reference edits varied based on the aspect being evaluated. The text generated is evaluated for three aspects:

- **Neutrality**: The user is presented with the original biased statement and bias corrected results from one of the models, and asked to rate which of the statement is more biased on a scale of -2 to 2 (-2 is for original text is more biased, 0 is same biased and 2 for generated text is more biased). When presenting sentences, the placement of biased and corrected sentences is randomised.
- **Fluency**: A combination of the text generated by various models along with ground truth is presented to the user in pairs and asked which one is more fluent. The fluency of a sentence is rated in a scale of -2 to 2 (-2 for ground-truth is more fluent, 0 if both have same fluency and 2 if generated text is more fluent).
- **Content preservation**: For content preservation, the original sentence and model corrected sentence is presented to the user and asked whether the second sentence contains the same information as the first sentence. The content preservation is rated on a scale of 0 to 4 (0 representing the content is totally different and 4 being very similar).

Appendix F. Examples of System Output - Qualitative Comparison

See Table 8

Biased Statements	**Neutral Statements**
Less sophisticated believers in biblical inerrancy **may fail** to allow for the possibility of transcription errors or translation errors.	**A portion of** believers in biblical inerrancy **fail** to allow for the possibility of transcription errors or translation errors.
Released on May 16 , 2002 , Attack of the Clones was **generally perceived** as a **slight improvement** upon the feeble The Phantom Menace, **though not at all on par** with the original Star Wars trilogy.	Released on May 16 , 2002 , Attack of the Clones was **perceived** as an **improvement** upon the feeble The Phantom Menace, **though not on par** with the original Star Wars trilogy.
Only a tiny proportion of these companies have **so far** grown into multinationals : ARM, Autonomy Corporation and AVEVA are the **most obvious examples** , and more recently CSR has seen rapid growth due to the uptake of Bluetooth.	**A proportion** of these companies have **grown** into multinationals: ARM, Autonomy Corporation and AVEVA are the **few examples**, and more recently CSR has seen rapid growth due to the uptake of Bluetooth.

Table 7: Sample of biased and its corresponding neutralized statements extracted from Wikipedia. Spans highlighted are the output of our bias tagger by type such as **Epistemological Bias**, **Framing Bias** and **Demographic Bias**. **Neutralized** spans are highlighted in the second column. Neutralized spans are manually identified, not by bias tagger.

	Examples of Bias Corrections
Source	**She wore a beautiful dress which she had made herself.**
Reference	She wore a dress which she had made herself.
RL-Seq2Seq	The dress she wore was one she had made herself.
RL-Trans	The dress she wore was made by her
Source	**Harvington is a lively village with a number of amenities including a shop , farm shop and Post Office.**
Reference	Harvington is a village with a number of amenities including a shop , farm shop and Post Office.
RL-Seq2Seq	Harvington has a number of amenities including a shop , farm shop and Post Office.
RL-Trans	The village of Harvington has a number of amenities, including a shop, farm shop, and a post office.
Source	**It is known as one of the only places to get an authentic New York City style egg cream.**
Reference	It is known as one of the places to get an authentic New York City style egg cream.
RL-Seq2Seq	It is known for selling authentic New York City style egg cream.
RL-Trans	The restaurant is famous for its authentic New York City egg cream.
Source	**He supports the failed war in Iraq.**
Reference	He supports the war in Iraq.
RL-Seq2Seq	He is in favor of the Iraq war.
RL-Trans	The war in Iraq is supported by him.
Source	**They promote the false perception that evolution is a theory in crisis.**
Reference	They promote their minority perception that evolution is a theory in crisis.
RL-Seq2Seq	They promote a minority perception of evolution as a theory in crisis.
RL-Trans	They promote the perception that evolution is in crisis among minorities.
Source	**Less sophisticated believers in biblical inerrancy may fail to allow for the possibility of transcription errors or translation errors.**
Reference	Some believers in biblical inerrancy don't allow for the possibility of transcription errors or translation errors.
RL-Seq2Seq	Some believers in biblical inerrancy deny the possibility of transcription errors or translation errors.
RL-Trans	There are some believers in biblical inerrancy who don't accept the possibility of transcription errors or translation errors.
Source	**As a professional librarian he was exceptionally erudite and influenced by a wide range of authors , including Dante Alighieri , Miguel de Cervantes , Franz Kafka , H.G**
Reference	As a librarian he was exceptionally erudite and influenced by a wide range of authors , including Dante Alighieri , Miguel de Cervantes , Franz Kafka , H.G .
RL-Seq2Seq	As a librarian he was erudite and influenced by a wide range of authors , including Dante Alighieri , Miguel de Cervantes , Franz Kafka , H.G .
RL-Trans	As a librarian, he was well-educated and influenced by a variety of authors, among them Dante Alighieri, Miguel de Cervantes, Franz Kafka and H.G. Wells.

Table 8: Sample model outputs

Detecting Lexical Borrowings from Dominant Languages in Multilingual Wordlists

John E. Miller
Artificial Intelligence/Engr.
Pontificia Universidad Católica del Perú
San Miguel, Lima, Peru
`jemiller@pucp.edu.pe`

Johann-Mattis List
Chair of Multilingual Computational Linguistics
University of Passau
Germany
`mattis.list@uni-passau.de`

Abstract

Language contact is a pervasive phenomenon reflected in the borrowing of words from donor to recipient languages. Most computational approaches to borrowing detection treat all languages under study as equally important, even though dominant languages have a stronger impact on heritage languages than vice versa. We test new methods for lexical borrowing detection in contact situations where dominant languages play an important role, applying two classical sequence comparison methods and one machine learning method to a sample of seven Latin American languages which have all borrowed extensively from Spanish. All methods perform well, with the supervised machine learning system outperforming the classical systems. A review of detection errors shows that borrowing detection could be substantially improved by taking into account donor words with divergent meanings from recipient words.

1 Introduction

Language contact is one of the most pervasive linguistic phenomena. It is the first factor that needs to be excluded when searching for genetic relations among the languages of the world, and it needs to be controlled for when searching for cross-linguistic universals. Any study trying to explain how humans use language must take language contact into account. It is also a unique witness of ancient contacts in human prehistory.

Language contact is most prominently reflected in *lexical borrowing*, the *transfer* of words from a donor language to a recipient language. Although research on the computational handling of lexical borrowing has made some progress of late, computational approaches to the investigation of language contact are still in their infancy (List, 2019). Specifically methods that could infer the direction of borrowings have not been proposed so far. While numerous case studies investigate the influence of dominant languages on heritage languages (Prochazka and Vogl, 2017; Meisel, 2018), we lack automated methods that could be used to study the influence of particular dominant languages in linguistically diverse areas of the world.

The number of standardized cross-linguistic wordlist collections has greatly increased over the past years (Rzymski et al., 2020; List et al., 2022a), with standard formats (Forkel et al., 2018) accompanied by software tools that allow scholars to prepare and curate standardized wordlists in an efficient manner (Forkel and List, 2020). It would be useful to have a computer assisted tool for linguists to detect which words in a cross-linguistic region have been borrowed from a dominant language and with further advances, a tool for inclusion in language contact assessment or in computational cladistics workflows.

Here, we compare the suitability of three different systems to address this task – two systems based on classical algorithms for automated sequence comparison that can be applied in supervised and unsupervised settings, and one supervised system based on extended machine learning techniques. We test these systems on a newly derived dataset of seven Latin American languages (Fig. 1) in which Spanish is a dominant donor language. Our results show that the supervised machine-learning system outperforms the classical systems.

2 Previous Work

Although still few in number, automatic methods for borrowing detection have been increasingly applied and developed in the past years. Early studies by van der Ark et al. (2007) and later Mennecier et al. (2016) compute edit distances between words from genetically unrelated languages and compare distances to thresholds, in order to detect borrowed words in multilingual wordlists.

Besides edit distance, which directly calculates distances between phonetic sequences, sound class

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2599–2605
May 2-6, 2023

based methods cluster phonetic segments into sound classes and then compute distances between sound class sequences. The sound-class based alignment method (SCA) (List, 2012) provides sound class categories, scoring functions for distance measures, and modifiable gap scores based on prosodic context.

Zhang et al. (2021) compare edit distance performance against SCA (List, 2012) distance performance, finding that SCA outperforms edit distance in accuracy. Hantgan et al. (2022) build on this work, using dedicated methods for automated cognate detection applied to languages from different language families in order to identify clusters of related words resulting from lexical borrowing. List and Forkel (2022a) expand this work further, by applying a two-stage workflow in which they first identify language-family-internal cognates, using a method specifically apt for the detection of deep cognates, and then compute SCA distances between cognate sets from genetically unrelated languages in order to infer sets of words related by lexical transfer.

Miller et al. (2020) compute language models for inherited and borrowed words for individual languages from the World Loanword Database (WOLD, Haspelmath and Tadmor 2009) using Markov Chains and Recursive Neural Networks and compare cross-entropies for inherited and borrowed language models in order to identify borrowings from monolingual information alone.

Kaiping and Klamer (2022) use automated methods for cognate detection (List et al., 2017) on a target set of Timor-Alor-Pantar languages. In order to infer borrowings from Indonesian and Tetun (not in the target set), they include both languages in their sample and treat all cognate sets that involves words from either of the two languages as borrowings. Moro et al. (2023) apply a similar approach to investigate borrowings in Alorese.

Mi et al. (2020) and Nath et al. (2022) train binary classifiers, mainly neural based, on large wordlists to predict borrowed words, and achieve F1 scores in the 0.75 to 0.85 range. Their workflows seem cumbersome, compute intensive, and not minimalist, but the results are promising.

3 Materials and Methods

3.1 Materials

For this study, a new comparative wordlist was created by taking data for seven Latin American languages from WOLD (`https://wold.clld.org`, Haspelmath and Tadmor 2009) and combining them with a wordlist of Spanish derived from the Intercontinental Dictionary Series (`https://ids.clld.org`, Key and Comrie 2015). Phonetic transcriptions for the Latin American languages were added to WOLD by Miller et al. (2020). Latin American Spanish phonetic transcriptions were added for this study (and could be later expanded by adding more transcriptions from historical varieties of Spanish). The resulting dataset conforms to the standards suggested by the Cross-Linguistic Data Formats initiative (CLDF, `https://cldf.clld.org`, Forkel et al. 2018). The data curation follows the Lexibank workflow (List et al., 2022a) and checks that data conform to certain standards, with languages being linked to Glottolog (`https://glottolog.org`, Hammarström et al. 2022, Version 4.7), concepts being linked to Concepticon (`https://concepticon.clld.org`, List et al. 2022b, Version 3.0), and transcriptions following the B(road)IPA conventions of the Cross-Linguistic Transcription Systems reference catalog (`https://clts.clld.org`, List et al. 2021, Version 2.2, see Anderson et al. 2018). Details of the resulting database are shown in the map of language locations along with percentages for borrowings from Spanish in Fig. 1 and in Tab. 1. Q'eqchi' and Zincantán Tzotzil are both Mayan languages, but appear substantially varied in the database.

Language	**Concepts**	**Lexemes**	**Segments**	**Vocab.**
Imb. Quechua	1,155	1,156	7,177	33
Mapudungun	1,040	1,242	7,356	33
Otomi	1,252	2,241	11,730	57
Q'eqchi'	1,211	1,773	10,367	49
Wichí	1,128	1,219	8,233	44
Yaqui	1,242	1,433	9,297	28
Zin. Tzotzil	955	1,266	7,129	41
Spanish	1,308	1,770	11,261	30
Aggregate	1,308	12,100	72,550	112

Table 1: Database details for seven Latin American languages plus Latin American Spanish.

3.2 Methods

Methods for Borrowing Detection. We develop three different methods for the detection of borrowings *from* a dominant language *to* non-dominant languages in multilingual wordlists. Following historical linguistics comparative method practice (Campbell, 2013), only word forms corresponding to the same concept are considered as

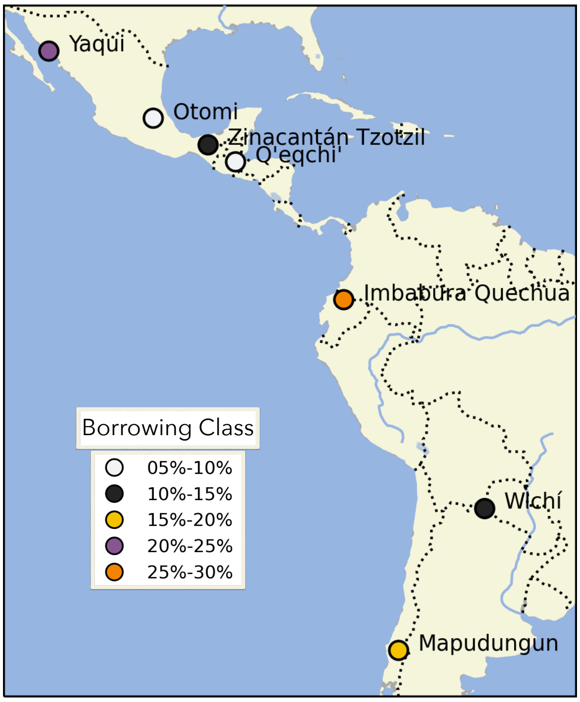


Figure 1: Map of languages with Spanish borrowing class.

candidates for borrowing.

The first method, called *Closest Match* borrowing detection in the following, iterates over all word pairs that express the same concept in the dominant language and the heritage languages and then computes phonetic distances. Word pairs whose phonetic distance is below a certain threshold are judged to be borrowings from the dominant language. We test two phonetic distances, the normalized edit distance (NED) – the classical edit distance (Levenshtein, 1965) between two words, divided by the length of the longer word – and the SCA distance (List, 2012).

The second method, called *Cognate-Based* borrowing detection in the following, follows the approach by Hantgan et al. (2022): it first computes cognates using a cluster-based approach for automated cognate detection in which words expressing the same concept whose average phonetic distance is below a certain threshold are assigned to the same cognate set (List et al., 2017), and then identifies all words assigned to cognate sets involving the dominant language as borrowings. We tested again normalized edit and SCA distances.

The third method, called *Classifier-Based* borrowing detection in the following, iterates over all word pairs with the same concept, but stores phonetic distance scores for various distance measures as vectors, which can then be used to train a classifier, firstly, a linear Support Vector Machine (SVM) (Cristianini and Shawe-Taylor, 2000), in a supervised setting. We tested various phonetic distance measures, but report only on the combination of normalized edit and SCA distances, as they yielded the best results. Both Closest Match and Cognate-Based methods require a fixed threshold which we estimate from the training data. So all three methods are considered as supervised.

Sampling. Train-test splits are made based on concepts rather than individual word entries. This permits matching of words for the same concept in in all methods without loss of candidate words. Treating train-test split as a nuisance variable takes into account differences between partitions across methods thus controlling for effects of sampling by concepts with differing borrowing behavior or statistical dependencies between test partitions due to sampling without replacement. See (Dror et al., 2018) for mention of the dependency problem with cross-validation. Our use of a fixed partition across treatments and analysis of variance controlling for partition as a nuisance or 'blocking' variable accounts for this dependency, and takes advantage of any systematic effects in borrowing behavior by partition.

Evaluation. For the analysis of the cross-validation data, we use a randomized blocks design where experiment is the treatment or factor, and test partition is the randomized block or nuisance variable. A standard analysis of variance partitions treatment effects, nuisance variable, and error, and permits a more powerful test of treatment differences without the nuisance variable variance. We follow up statistically significant findings for treatment (experiment), with comparisons of experiments versus the overall average using a joint (family) error rate.

Implementation. Our methods are implemented in Python, making specifically use of the CLDF-Bench package (`https://pypi.org/project/cldfbench/`, Forkel and List 2020, Version 1.13.0) to provide commandline access to all methods described here. For the computation of alignments and edit distances, LingPy (`https://pypi.org/project/lingpy`, List and Forkel 2022b, Version 2.6.9) is used. SVM and evaluation are realized with the help of Scikit-Learn (`https://pypi.org/project/scikit-learn/`, Pedregosa et al. 2011, Version 1.2.1).

4 Results and Discussion

We tested our three methods with two distance measures in five experiments (normalized edit and SCA distances individually in both Closest Match and Cognate-Based methods, and combined in the Classifier-Based method) using a 10-fold cross-validation on our data and reporting precision, recall, F1 scores, accuracy, and execution times (mm:ss). F1 score is the primary result measure; accuracy and execution time are informational. The 10-fold cross-validation uses the same 10 fixed train and non-overlapping test splits for all experiments. With few parameter estimates (1 threshold each for Closest Match and Cognate-Based, 2 distance and 7 target language coefficients for Classifier), a separate *train* split into *fit/val* is not necessary.

All methods perform well with less than 5 points separating the highest from the lowest F1 scores. Tab. 2 shows the results of the ten-fold cross validation of our three methods in five experiments. An analysis of variance,[1] with experiment as the effects variable and train-test split as the nuisance variable, shows highly significant effects for precision ($\mathrm{F}_{4,36} = 25.74, p < 0.0001$) and F1 score ($\mathrm{F}_{4,36} = 14.3, p < 0.0001$).

Closest Match with normalized edit distance performs poorly, while Classifier-Based with combined normalized edit and SCA distances performs well. Classifier-Based performs better than the average of all experiments in F1 score, and substantially better in precision versus other experiments; the method is conservative, with a low number of false positives. Performance on remaining experiments is indistinguishable from the overall average of all experiments combined. The Cognate-Based method is compute intensive performing multiple alignment over all languages. Accuracy is well above the majority decision accuracy of 84.8% ($100\% - 15.2\%$ borrowing) in all experiments.

A search for classifier improvements prompted several *ad hoc* experiments (see tab. 3). We observe: (1) A radial basis function (rbf) SVM classifier performs no better than our linear SVM. We suspect the estimated target language parameters do not generalize well to held-out data. (2) A logistic regression classifier performs on par with our linear SVM. (3) A weight balanced SVM classifier trades an increase in recall for a larger drop in precision. We also test whether using separate trials for each target language in Closest Match, would perform as well as all languages together. A combined trial performs better; a single threshold estimate appears to generalize better to held-out data than using individual language estimates.

Method	**Prec.**	**Rec.**	**F1**	**Acc.**	**mm:ss**
Closest Match					
NED	0.832	0.703	0.761	0.938	00:15
SCA	0.869	0.720	0.787	0.945	00:29
Cognate-Based					
NED	0.853	0.705	0.771	0.941	01:48
SCA	0.862	0.719	0.783	0.944	04:49
Classifier-Based SVM (linear)					
NED, SCA	**0.931**	0.713	**0.806**	0.952	00:37

Table 2: Ten-fold cross-validation for three methods with NED (normalized edit) and SCA (Sound-Class based phonetic alignment) distance measures. **Bolded** estimates are superior to and underlined estimates inferior to the the overall average using analysis of means (Nelson et al., 2005) with joint error rate $\alpha = 0.05$.

Experiment	**Prec.**	**Rec.**	**F1**	**Acc.**
Classifier Variations - NED, SCA				
SVM (rbf)	0.945	0.694	0.799	0.951
Logistic regression	0.914	0.728	0.809	0.952
SVM (balanced)	0.613	0.826	0.704	0.902
Closest Match - SCA				
Each language (avg)	0.860	0.707	0.770	0.941

Table 3: Ten-fold cross-validation for several *ad hoc* experiments with NED (normalized edit) and SCA (Sound-Class based phonetic alignment) distance measures. Classifier experiments: SVM with radial basis function, Logistic regression, linear SVM with balanced class weights. Descriptive statistics only.

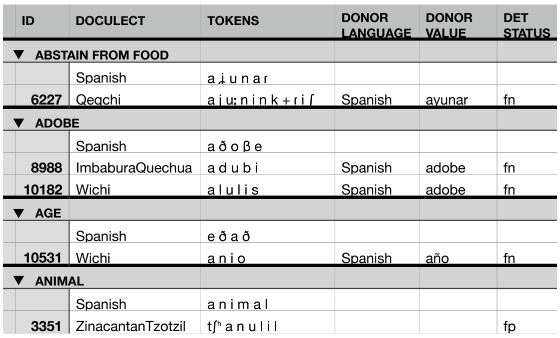
Tab. 4 shows the results of the Classifier-Based method for the seven target languages in our sample with training and evaluation over the entire dataset. There is some variation in performance by language, in particular, with recall in $[0.615, 0.778]$. We detect a linear relation between the performance and the amount of borrowings from the dominant language in the target languages. (Precision: $\mathrm{r} = -0.39, \mathrm{NS}$; Recall: $\mathrm{r} = 0.88, p < 0.01$; F1 score: $\mathrm{r} = 0.85, p < .01$; 1-sided Pearson correlation tests with $df = 5$). Recall and F1 scores improve as borrowing increases. This could be an artifact of higher borrowing resulting in better estimation of a target language coefficient, or more interestingly, a cultural process where more dominant-donor borrowing corresponds to reduced phonetic adaption into the target language.

---

[1] Statistical analyses with JMP (SAS Institute Inc., 2021).

Language	Prec.	Rec.	F1	Acc.	Borr.
Imb. Quechua	0.921	0.773	0.841	0.924	26%
Mapudungun	0.944	0.716	0.814	0.950	15%
Otomi	0.932	0.692	0.794	0.968	9%
Q'eqchi'	0.934	0.615	0.742	0.961	9%
Wichí	0.952	0.658	0.778	0.953	12%
Yaqui	0.938	0.778	0.851	0.941	22%
Zin. Tzotzil	0.932	0.661	0.773	0.949	13%
Average	0.934	0.714	0.810	0.952	15%

Table 4: Individual language results for the Classifier-Based borrowing detection methods on the seven target languages in our sample. The last column shows the proportion of Spanish borrowings.



ID	DOCULECT	TOKENS	DONOR LANGUAGE	DONOR VALUE	DET STATUS
▼ ABSTAIN FROM FOOD					
	Spanish	a ʝ u n a ɾ			
6227	Qeqchi	a j uː n i n k + ɾ i ʃ	Spanish	ayunar	fn
▼ ADOBE					
	Spanish	a ð o β e			
8988	ImbaburaQuechua	a d u b i	Spanish	adobe	fn
10182	Wichi	a l u l i s	Spanish	adobe	fn
▼ AGE					
	Spanish	e ð a ð			
10531	Wichi	a n i o	Spanish	año	fn
▼ ANIMAL					
	Spanish	a n i m a l			
3351	ZinacantanTzotzil	tʃʰ a n u l i l			fp

Figure 2: Example collection of detection errors.

To get a better understanding about the different types of errors that our best performing experimental combination commits, we conducted a detailed error analysis from the Classifier-Based borrowing detection results. A spreadsheet snippet (Fig. 2), serves as a reference for several error types. For undetected borrowings (false negatives), we identified four error types: (1) cases where the borrowed form was not present in the donor wordlist, e.g., Mapudungun *peso* "coin" is borrowed from Spanish *peso* "peso", but our Spanish wordlist only has *moneda*, (2) cases where the form was present in the donor wordlist, but with a different concept, e.g., Wichi *anio* "age" is borrowed from Spanish *año* "year", while the Spanish word for "age" is *edad*, (3) cases of large phonetic distance between donor and recipient forms, e.g., Wichi *alulis* "adobe", which is somewhat distant from Spanish *adobe*, and (4) cases of unrecognized partial borrowing, e.g., Qeqchi *aiunink-riʃ* "abstain from food", which is partially borrowed from Spanish *ajunar* "fast". For falsely detected borrowings (false positives), we identified three error types: (1) cases where the form was not borrowed from the dominant language but *vice versa*, e.g., Spanish *poroto* "bean" was borrowed *from* Quechua *purutu*, (2) cases of chance similarities between word forms, e.g., Spanish *animal* "animal" and Zinacantan Tzotzil *tʃanulil*, and (3) cases so improbably similar that we suspect errors in the original annotation, e.g., Spanish *pelota* "ball" and Wichi *pelutaj*.

Error Type	Count	Pct
Undetected Borrowings		
borrowed form not in donor list	28	17
different concept than recipient form	75	45
large phonetic distance	31	19
partial borrowing as only reason	5	3
Subtotal	139	84
Falsely Detected as Borrowings		
Error Type	Count	Pct
direction not from dominant donor	7	4
chance similarity of form	10	6
likely dataset error	9	5
Subtotal	26	16
Total	165	100

Table 5: Summary over sample of undetected (false negative) and falsely detected (false positive) borrowings.

For a large sample of concepts, we tallied 139 undetected (false negative) and 26 falsely detected (false positive) borrowings (see Tab. 5). Most errors were in recall, with many of these borrowings from lexemes **not** within the same concept.

5 Conclusion

How well can we automatically detect borrowings from dominant languages based on wordlist data? We devised three general methods to detect borrowed words from dominant languages, two based on sequence comparison workflows and one based on a classifier. The classifier-based method showed the best performance, with F1 scores of 0.81, and high precision of 0.93. This method could already prove very useful in computer-assisted workflows. Our investigation of detection errors shows several opportunities for improvement. Most undetected borrowings result from current methods' restriction to searching only for word forms for the same concept. We see great potential in improvements that account for borrowing accompanied by *semantic shift*, specifically: (1) augment donor wordlist coverage of possible forms, (2) relax "same" to "similar" concept restriction for matching forms, or (3) fit language models to wordlists and add word cross-entropy to the classifier without restriction. Tests adding word cross-entropy, not reported here, look very promising, but more research is needed.

Limitations

In this study, we apply limited methods for detection of lexical borrowing to the case of a single dominant donor language (Spanish) wordlist versus seven Latin American language wordlists. This application uses relatively sparse data, with an average of 1,512 lexemes per language wordlist, with very few estimated parameters, and so apt for lower resource languages. Future work will seek to remove the wordlist limitation.

Ethics Statement

Our data are taken from publicly available sources. There are no ethical issues or conflicts of interest in this work.

Supplementary Material

The supplementary material accompanying this study contains the data and code needed to replicate the results reported here, along with detailed information on installing and using the software. It is curated on GitHub (`https://github.com/lexibank/sabor`, Version 1.0) and has been archived with Zenodo (`https://doi.org/10.5281/zenodo.7591335`).

Acknowledgements

This research was supported by the Max Planck Society Research Grant *CALC³* (JML, `https://digling.org/calc/`), the ERC Consolidator Grant *ProduSemy* (JML, Grant No. 101044282, see `https://cordis.europa.eu/project/id/101044282`), and by the Graduate School of the Pontificia Universidad Católica del Perú (PUCP) through the Huiracocha-2019 scholarship program, see `https://posgrado.pucp.edu.pe` (JEM). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency (nor any other funding agencies involved). Neither the European Union nor the granting authority can be held responsible for them. We thank the anonymous reviewers for helpful comments and all people who share their data openly, so we can use it in our research.

References

Cormac Anderson, Tiago Tresoldi, Thiago Costa Chacon, Anne-Maria Fehn, Mary Walworth, Robert Forkel, and Johann-Mattis List. 2018. A Cross-Linguistic Database of Phonetic Transcription Systems. *Yearbook of the Poznań Linguistic Meeting*, 4(1):21–53.

L. Campbell. 2013. *Historical Linguistics: An Introduction*. Edinburgh University Press.

Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Robert Forkel and Johann-Mattis List. 2020. Cldfbench. give your cross-linguistic data a lift. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation*, pages 6997–7004, Luxembourg. European Language Resources Association (ELRA).

Robert Forkel, Johann-Mattis List, Simon J. Greenhill, Christoph Rzymski, Sebastian Bank, Michael Cysouw, Harald Hammarström, Martin Haspelmath, Gereon A. Kaiping, and Russell D. Gray. 2018. Cross-Linguistic Data Formats, advancing data sharing and re-use in comparative linguistics. *Scientific Data*, 5(180205):1–10.

Harald Hammarström, Martin Haspelmath, Robert Forkel, and Sebastiaon Bank. 2022. *Glottolog [Dataset, Version 4.6]*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

A Hantgan, H Babiker, and J List. 2022. First steps towards the detection of contact layers in bangime: a multi-disciplinary, computer-assisted approach [version 2; peer review: 2 approved]. *Open Res Europe*, 2:10:1–25.

Martin Haspelmath and Uri Tadmor. 2009. *World Loanword Database*. Max Planck Institue for Evolutionary Anthropology, Leipzig.

Gereon A. Kaiping and Marian Klamer. 2022. The dialect chain of the Timor-Alor-Pantar language family. *Language Dynamics and Change*, 2:1–53.

Mary Ritchie Key and Bernard Comrie, editors. 2015. *Intercontinental Dictionary Series (IDS)*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

V. I. Levenshtein. 1965. Dvoičnye kody s ispravleniem vypadenij, vstavok i zameščenij simvolov [binary codes with correction of deletions, insertions and replacements]. *Doklady Akademij Nauk SSSR*, 163(4):845–848.

Johann-Mattis List. 2012. Multiple sequence alignment in historical linguistics. In *Proceedings of ConSOLE XIX*, pages 241–260.

Johann-Mattis List. 2019. Automated methods for the investigation of language contact situations, with a focus on lexical borrowing. *Language and Linguistics Compass*, 13(e12355):1–16.

Johann-Mattis List, Cormac Anderson, Tiago Tresoldi, and Robert Forkel. 2021. *Cross-Linguistic Transcription Systems [Dataset, Version 2.1.0]*. Max Planck Institute for the Science of Human History, Jena.

Johann-Mattis List and Robert Forkel. 2022a. Automated identification of borrowings in multilingual wordlists [version 3; peer review: 4 approved]. *Open Research Europe*, 1(79):1–11.

Johann-Mattis List and Robert Forkel. 2022b. *LingPy. A Python library for quantitative tasks in historical linguistics [Software Library, Version 2.6.9]*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Johann-Mattis List, Robert Forkel, Simon J. Greenhill, Christoph Rzymski, Johannes Englisch, and Russell D. Gray. 2022a. Lexibank, a public repository of standardized wordlists with computed phonological and lexical features. *Scientific Data*, 9(356):1–16.

Johann-Mattis List, Simon J. Greenhill, and Russell D. Gray. 2017. The potential of automatic word comparison for historical linguistics. *PLOS ONE*, 12(1):1–18.

Johann-Mattis List, Annika Tjuka nd Christoph Rzymski, Simon J. Greenhill, Nathanael E. Schweikhard, and Robert Forkel. 2022b. *CLLD Concepticon [Dataset, Version 2.6.0]*. Max Planck Institute Evolutionary Anthropology, Leipzig.

Jürgen M. Meisel. 2018. Early child second language acquisition: French gender in german children. *Bilingualism: Language and Cognition*, 21(4):656–673.

Phillipe Mennecier, John Nerbonne, Evelyne Heyer, and Franz Manni. 2016. A Central Asian language survey. *Language Dynamics and Change*, 6(1):57–98.

Chenggang Mi, Lei Xie, and Yanning Zhang. 2020. Loanword identification in low-resource languages with minimal supervision. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 19(3).

John E. Miller, Tiago Tresoldi, Roberto Zariquiey, César A. Beltrán Castañón, Natalia Morozova, and Johann-Mattis List. 2020. Using lexical language models to detect borrowings in monolingual wordlists. *PLOS One*, 15(12):e0242709.

Francesca R. Moro, Yunus Sulistyono, and Gereon A. Kaiping. 2023. Detecting papuan loanwords in alorese: Combining quantitative and qualitative methods. In *Traces of Contact in the Lexicon*, pages 213–262. Brill.

Abhijnan Nath, Sina Mahdipour Saravani, Ibrahim Khebour, Sheikh Mannan, Zihui Li, and Nikhil Krishnaswamy. 2022. A generalized method for automated multilingual loanword detection. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4996–5013, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

P. R. Nelson, P. S. Wludyka, and K. A. F. Copeland. 2005. The analysis of means: A graphical method for comparing means, rates, and proportions. *SIAM*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Katharina Prochazka and Gero Vogl. 2017. Quantifying the driving factors for language shift in a bilingual region. *Proceedings of the National Academy of Sciences*, 114(17):4365–4369.

Christoph Rzymski, Tiago Tresoldi, Simon Greenhill, Mei-Shin Wu, Nathanael E. Schweikhard, Maria Koptjevskaja-Tamm, Volker Gast, Timotheus A. Bodt, Abbie Hantgan, Gereon A. Kaiping, Sophie Chang, Yunfan Lai, Natalia Morozova, Heini Arjava, Nataliia Hübler, Ezequiel Koile, Steve Pepper, Mariann Proos, Briana Van Epps, Ingrid Blanco, Carolin Hundt, Sergei Monakhov, Kristina Pianykh, Sallona Ramesh, Russell D. Gray, Robert Forkel, and Johann-Mattis List. 2020. The Database of Cross-Linguistic Colexifications, reproducible analysis of cross- linguistic polysemies. *Scientific Data*, 7(13):1–12.

SAS Institute Inc. 2021. Jmp$^{®}$, version 16.1.0. Software.

René van der Ark, Philippe Mennecier, John Nerbonne, and Franz Manni. 2007. Preliminary identification of language groups and loan words in Central Asia. In *Proceedings of the RANLP Workshop on Acquisition and Management of Multilingual Lexicons*, pages 13–20.

Liqin Zhang, Ray Fabri, John Nerbonne, and John Nerbonne. 2021. Detecting loan words computationally. In *Contact Language Library*, pages 269–288. John Benjamins Publishing Company.

Towards Integration of Discriminability and Robustness for Document-Level Relation Extraction

Jia Guo[*1,2], **Stanley Kok**[1], **Lidong Bing**[2]
[1]School of Computing, National University of Singapore
[2]DAMO Academy, Alibaba Group
`guojia@u.nus.edu, skok@comp.nus.edu.sg`
`l.bing@alibaba-inc.com`

Abstract

Document-level relation extraction (DocRE) predicts relations for entity pairs that rely on long-range context-dependent reasoning in a document. As a typical multi-label classification problem, DocRE faces the challenge of effectively distinguishing a small set of positive relations from the majority of negative ones. This challenge becomes even more difficult to overcome when there exists a significant number of annotation errors in the dataset. In this work, we aim to achieve better integration of both the discriminability and robustness for the DocRE problem. Specifically, we first design an effective loss function to endow high discriminability to both probabilistic outputs and internal representations. We innovatively customize entropy minimization and supervised contrastive learning for the challenging multi-label and long-tailed learning problems. To ameliorate the impact of label errors, we equipped our method with a novel negative label sampling strategy to strengthen the model robustness. In addition, we introduce two new data regimes to mimic more realistic scenarios with annotation errors and evaluate our sampling strategy. Experimental results verify the effectiveness of each component and show that our method achieves new state-of-the-art results on the DocRED dataset, its recently cleaned version, Re-DocRED, and the proposed data regimes. [1]

1 Introduction

The problem of document-level relation extraction (DocRE) has garnered increasing attention from the research community (Quirk and Poon, 2017; Peng et al., 2017; Yao et al., 2019) due to its importance to real-world applications. DocRE is inherently a multi-label problem, in which we have to predict a set of relations from the pre-defined label set for every entity pair in a document. Thus, it is crucial for DocRE models to adopt an effective learning objective that can clearly distinguish massive semantically close relations.

Recently, several works have proposed new loss functions to learn an adaptive threshold for better separating positive and negative relations. However, these approaches (Zhou et al., 2021; Tan et al., 2022a) either enforce learning a *total order* among all relations that leads to superfluous comparisons and diminishing differences among them or improperly penalize all pre-defined labels of positive entity pairs if their average margins are lower than the threshold when addressing the label imbalance problem (Zhou and Lee, 2022). In contrast, we propose an approach that learns a *partial order*, ranking all positive relations above a threshold individually, which is in turn ranked above all negative relations. Our approach does not waste precious data and probability mass in modeling the ordering among positive relations (likewise for negative relations). We further sharpen the distinction in each distribution of a relation and the threshold through the principled use of entropy minimization.

Besides, none of the above methods take the discriminability of internal representations into account, as well as the model robustness against annotation errors. To solve these issues, we introduce novel modifications to the supervised contrastive learning (Khosla et al., 2020) to accentuate the differences among the embeddings of entity pairs from different classes and the similarities of that from the same class. Our method can better accommodate the multi-label setting and the long-tail phenomenon that is typically present in DocRE datasets. To combat the annotation error problem stated in Tan et al. (2022b), we design two new data regimes and a novel negative label sampling strategy that gives consistently strong performance even with incomplete annotations. In sum, our

---

[*] This work was partially done when Jia Guo was an intern at DAMO Academy, Alibaba Group.

[1] Our codes and datasets are available at `https://github.com/guojiapub/PEMSCL`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2606–2617
May 2-6, 2023 ©2023 Association for Computational Linguistics

contributions are three-fold:

- We propose an effective loss function that boosts the discriminability of both internal embeddings and probabilistic outputs.
- We achieve good integration of discriminability and robustness by incorporating a novel negative label sampling strategy.
- Experimental results consistently demonstrate that we achieve new state-of-the-art performance in a variety of settings.

2 Related Work

Document-level relation extraction (DocRE) Early works on DocRE focus on utilizing graph convolutional networks (GCNs) (Kipf and Welling, 2017) to conduct complex cross-sentence reasoning on a document graph (Sahu et al., 2019; Christopoulou et al., 2019; Wang et al., 2020; Zeng et al., 2021). Recently, methods fine-tuned on large pre-trained language models (Devlin et al., 2019; Liu et al., 2019) achieved significant performance gain. In particular, SSAN (Xu et al., 2021) encoded entity dependencies into the self-attention mechanism to strengthen context and entity reasoning. ATLOP (Zhou et al., 2021) employed the multi-head attention weights to generate entity-related context representations which enhanced the embeddings of entity pairs. To better address the multi-label classification problem, both ATLOP (Zhou et al., 2021) and NCRL (Zhou and Lee, 2022) proposed to treat the NA class as an adaptive threshold. DocuNet (Zhang et al., 2021) and KD-DocRE (Tan et al., 2022a) extended the ATLOP architecture by increasing interactions between entities and incorporating knowledge distillation, respectively. Besides, other DocRE models attempted to leverage auxiliary information for relation prediction, such as meta dependency paths (Nan et al., 2020), external knowledge bases (Li et al., 2021a), and evidences (Xie et al., 2022; Xiao et al., 2022). We additionally provide detailed comparison with existing works in Section 3.3.

Other related works Entropy Minimization technique was commonly seen in semi-supervised learning works (Grandvalet and Bengio, 2004; Vu et al., 2019). However, we are the first to employ entropy minimization in the challenging multi-label supervised learning framework. Besides, our entropy minimization takes effect in each customized probability distribution of the relation label and threshold class, which will encourage a larger distinction between them.

Supervised contrastive learning (SCL) (Khosla et al., 2020) extends self-supervised contrastive learning (He et al., 2020; Chen et al., 2020) to the fully supervised setting by constructing "positive" and "negative" examples based on their labels. ERICA (Qin et al., 2021) proposed a pre-training framework using contrastive learning to improve representations of entities and relations. However, this work samples positive pairs for relations proportionally to their total amount of examples, which will lead to biased optimization that favors primary relations over minor ones. Besides, they only maximize the similarity of one positive example pair each time, which may weaken the global effect of clustering. Instead, we give equal consideration to each relation and each positive example of anchors, and elaborately tailored the supervised contrastive learning to suit both the multi-label problem and long-tailed relation learning.

3 Methodology

In this section, we describe our model called PEM-SCL that is based on a **P**airwise moving-threshold loss, **E**ntropy **M**inimization, and **S**upervised **C**ontrastive **L**earning.

3.1 Problem Formulation

Let $D = \{w_l\}_{l=1}^{L}$ be a document containing L words and a set of entities $\mathcal{E}_D = \{e_i\}_{i=1}^{|\mathcal{E}_D|}$. Each entity e_i is associated with a set of mentions $\mathcal{M}_{e_i} = \{m_j^i\}_{j=1}^{|\mathcal{M}_{e_i}|}$ (i.e., a set of phrases referring to the same entity e_i). In document-level relation extraction, we predict the subset of relations in a pre-defined set $\mathcal{R} = \{r_k\}_{k=1}^{|\mathcal{R}|}$ that hold between each pair of entities $(e_h, e_t)_{h,t=1,\ldots,|\mathcal{E}_D|,h\neq t}$. We sometimes abbreviate an entity pair (e_h, e_t) as (h, t) to simplify notation. A relation is deemed to exist between the head entity e_h and tail entity e_t if it is expressed between any of their corresponding mentions. If no relation exists between any pair of their mentions, the entity pair is labeled `NA`. For each entity pair, we term a relation that holds between its constituent entities as *positive*, and the remaining relations in $\mathcal{R}$ as *negative*. An entity pair that is `NA` does not have any positive relation, and has the entire set $\mathcal{R}$ as negative relations (we could consider such a pair as having a special `NA` relation between them). Document-level relation extraction can be viewed as a multi-label problem, in which an entity pair corresponds to a training/test *example*, and the relations in $\mathcal{R} \cup \{\texttt{NA}\}$ correspond to the possible *labels* or *classes* of the example.

3.2 Encoder Model

We leverage ATLOP (Zhou et al., 2021) as our encoder since recent work (Xie et al., 2022; Zhou and Lee, 2022) has borne out its usefulness as a backbone in neural architectures. For each entity pair (e_h, e_t), the encoder model generates the entity pair representation $\boldsymbol{x}_{h,t} \in \mathbb{R}^{d_x}$, and its unnormalized score vector $\boldsymbol{f}_{h,t} \in \mathbb{R}^{|\mathcal{R}|+1}$ for relation prediction, we briefly describe them as follows[2]:

$$\boldsymbol{x}_{h,t} = \text{Encoder}\big((e_h, e_t)|D, \mathcal{M}_{e_h}, \mathcal{M}_{e_t}\big) \tag{1}$$

$$\boldsymbol{f}_{h,t} = \text{Linear}(\boldsymbol{x}_{h,t}) \tag{2}$$

3.3 Pairwise Moving-Threshold Loss with Entropy Minimization

In document-level relation extraction, a fixed probability threshold (e.g. a hyperparameter tuned on the development dataset) is used to decide the boundary of positive and negative relations. However, such a threshold is only suitable for entity pairs *on average*, and may not be ideal for entity pairs with particular properties.

To address this problem, we design a loss function that utilizes the NA class as a dynamic threshold, learning how best to *move* the threshold in accordance with the regularities present in each entity pair. Specifically, we conduct a *pairwise* comparison between each relation and the NA class (separately for each relation), and encourage the prediction scores of each positive relation to be higher than that of the NA class, and incentivize the score of the NA class to be higher than those of negative relations. In this way, we induce a *partial order* over $\mathcal{R} \cup \{\texttt{NA}\}$ for each entity pair. Note that the positive relations are not compared against each other, and their relative rankings are not modeled (likewise for negative relations). This makes sense in the multi-label setting where we are interested in finding the set of relations that are true without being concerned about their relative degrees of veracity.

Formally, we split the predefined relation set $\mathcal{R} = \mathcal{P}_{h,t} \cup \mathcal{N}_{h,t}$ into two mutually exclusive sets for each entity pair (h, t) in a training set, where $\mathcal{P}_{h,t}$ and $\mathcal{N}_{h,t}$ respectively denote the positive and negative relations of (h, t). As mentioned in Section 3.2, we make use of $\boldsymbol{f}_{h,t} \in \mathbb{R}^{|\mathcal{R}|+1}$ that is computed by Equation 1. We denote the elements of $\boldsymbol{f}_{h,t}$ that correspond to relation $r \in \mathcal{R}$ and to the NA class as f_r and f_η respectively. (Both f_r and f_η represent unnormalized prediction scores (logits).) Using f_r and f_η, we compute the probability that the label C of entity pair (h, t) is r (or η) *conditioned* on C being either r or η, i.e., $P_{h,t}(C = r|C = \{r, \texttt{NA}\})$ and $P_{h,t}(C = \eta|C = \{r, \texttt{NA}\})$ respectively, as follows.

$$P^r_{h,t}(r) = \frac{\exp(f_r)}{\exp(f_r) + \exp(f_\eta)},$$

$$P^\eta_{h,t}(r) = 1 - P^r_{h,t}(r) = \frac{\exp(f_\eta)}{\exp(f_r) + \exp(f_\eta)}, \tag{3}$$

where we have abbreviated $P_{h,t}(C = r|C = \{r, \texttt{NA}\})$ and $P_{h,t}(C = \eta|C = \{r, \texttt{NA}\})$ as $P^r_{h,t}(r)$ and $P^\eta_{h,t}(r)$ respectively.

Our *pairwise moving-threshold* loss $\mathcal{L}^{h,t}_{pmt}$ that maximizing the joint probability of all relations for an entity pair (h, t) is defined as:

$$\begin{aligned}
\mathcal{L}^{h,t}_{pmt} &= -\log\Big(\prod_{r \in \mathcal{P}_{h,t}} P^r_{h,t}(r) \prod_{r \in \mathcal{N}_{h,t}} (1 - P^r_{h,t}(r))\Big) \\
&= -\sum_{r \in \mathcal{P}_{h,t}} \log P^r_{h,t}(r) - \sum_{r \in \mathcal{N}_{h,t}} \log P^\eta_{h,t}(r) \\
&= \sum_{r \in \mathcal{P}_{h,t}} \log(1 + \exp(f_\eta - f_r)) \\
&+ \sum_{r \in \mathcal{N}_{h,t}} \log(1 + \exp(f_r - f_\eta)).
\end{aligned} \tag{4}$$

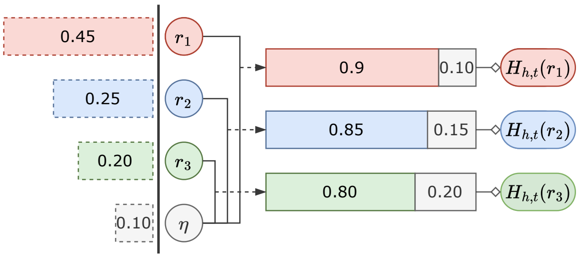


Figure 1: Each positive relation (colored rectangles with solid lines) exhibits a large probability difference from the threshold class (white rectangle with solid lines) when they are separately compared (like what is achieved with our $\mathcal{L}^{h,t}_{pmt}$ loss. We further expand this difference by minimizing $H(r)$ as stated in Eq. 5). However, the probabilities are diminished when each positive relation (colored rectangles with dashed lines) is made to compete with the other, reducing the disparity between the probability of each positive relation and that of the threshold class (white rectangle with dashed lines).

---

[2] Please refer to Appendix D for the computation details.

In Equation 4, note that the same threshold f_η is used for all $r \in \mathcal{R}$ for an entity pair (h, t). From the equation, we see that minimizing $\mathcal{L}_{pmt}^{h,t}$ equates to learning scores such that $f_r > f_\eta$ when r is a positive relation, and such that $f_\eta > f_r$ when r is negative relation. The (relative) scores for relations f_r and for the threshold f_η are fully learned from training data, and are tailored to individual entity pairs. Hence, they can better model the peculiarities specific to each entity pair.

Although previous work (Zhou et al., 2021; Tan et al., 2022a) employed a similar thresholding mechanism, they learn a *total order* for all relations (or a set of relations) and the threshold class. This wastes finite probability mass (total value of 1.0) in modeling the superfluous ordering among the relations that is not beneficial to multi-label problem, and inevitably diminishes the difference between the probability of each relation and that of the threshold. See Figure 1 for illustration.

Intuitively, a desirable trait of a loss function is that it reduces the uncertainty about whether a relation is positive or negative, thereby allowing its value to be discerned easily. To achieve this in a principled manner, we employ the principle of entropy minimization (Grandvalet and Bengio, 2005). Entropy minimization is typically used on unlabeled data in unsupervised or semi-supervised learning (Berthelot et al., 2020). In our case, we apply it on *labeled* data in a supervised setting. The information entropy for each pairwise probability distribution between relation r and the threshold class `NA` for entity pair (h, t) is defined as:

$$H_{h,t}(r) = -P_{h,t}^{r}(r)\log P_{h,t}^{r}(r) - P_{h,t}^{\eta}(r)\log P_{h,t}^{\eta}(r). \quad (5)$$

In Equation 5, information entropy decreases as the absolute difference between $P_{h,t}^{r}(r)$ and $P_{h,t}^{\eta}(r)$ increases, attaining a maximum when $P_{h,t}^{r}(r) = P_{h,t}^{\eta}(r) = 0.5$ and a minimum when either probability is 1.0 (and the other is 0.0). Thus, incorporating entropy into our loss function would help to accentuate the disparity between the pair $P_{h,t}^{r}(r)$ and $P_{h,t}^{\eta}(r)$ for all relations, making it easier to distinguish a positive (or negative) relation from the threshold `NA`.

We formulate our final pairwise moving-threshold loss with entropy minimization as follows:

$$\mathcal{L}_{em}^{h,t} = \frac{1}{\gamma_1} \sum_{r\in\mathcal{P}_{h,t}} H_{h,t}(r) + \frac{1}{\gamma_2} \sum_{r\in\mathcal{N}_{h,t}} H_{h,t}(r), \quad (6)$$

$$\mathcal{L}_1 = \sum_{(h,t)\in\mathcal{B}} \mathcal{L}_{pmt}^{h,t} + \mathcal{L}_{em}^{h,t}, \quad (7)$$

where $\mathcal{B}$ refers to a training batch, and $\gamma_1 = \{1, |\mathcal{P}_{h,t}|\}$ and $\gamma_2 = \{1, |\mathcal{N}_{h,t}|\}$ are hyperparameters weighting the effect of entropy minimization.

It is noted that using L_{pmt} on its own would lead to poor optimization for positive relations, in the situation where there is a preponderance of negative relations, the sum over $\mathcal{N}_{h,t}$ in Equation 4 might overwhelm the sum over $\mathcal{P}_{h,t}$ to such an extent that pushing f_η to a large value far above that of f_r for every negative relation r in order to minimize $\mathcal{L}_{pmt}^{h,t}$ (the same issue that also affects previous work (Zhou and Lee, 2022) without being properly addressed). Instead, our entropy minimization via $\mathcal{L}_{em}^{h,t}$ in Equation 6 provides a *principled* means to "balance" the sharp disparity between the probability of r and that of η across *all* relations. Empirically, $\mathcal{L}_{em}^{h,t}$ also demonstrates its efficacy in an ablation study (see Section 4.4.)

3.4 Supervised Contrastive Learning for Multi-Labels and Long-Tailed Relations

Rather than focusing only on sharpening the disparity of probability outputs as stated in Equation 7, we also seek to accentuate the disparities for the embeddings of entity pairs that are labeled with different relations. To do so, we take inspiration from supervised contrastive learning (Khosla et al., 2020) which aims to "pull" the embeddings of similar examples together, and "push" those of dissimilar examples apart.

However, the original supervised contrastive learning technique only deals with single-label data, and does not handle long-tail distributions. We have to introduce some novel modifications for it to work on our multi-label problem. We make use of the embedding $\boldsymbol{x}_{h,t}$ that is computed by Equation 1, and normalized it by L2 normalization before using it in the loss function below. After transplanting the loss function of supervised contrastive learning for our multi-label problem, we obtain the following loss function for an entity pair (h, t):

$$\mathcal{L}_{scl}^{h,t} = -\log\left\{\frac{1}{|\mathcal{S}_{h,t}|}\sum_{p\in\mathcal{S}_{h,t}} \frac{\exp(\boldsymbol{x}_{h,t}\cdot\boldsymbol{x}_p/\tau)}{\sum\limits_{d\in\mathcal{B},d\neq(h,t)} \exp(\boldsymbol{x}_{h,t}\cdot\boldsymbol{x}_d/\tau)}\right\}, \quad (8)$$

In Equation 8, $\mathcal{B}$ is a batch of examples (entity pairs) including (h, t). $\mathcal{S}_{h,t} \subseteq \mathcal{B}$ is such that each entity pair $p = (h', t')$ in $\mathcal{S}_{h,t}$ has at least one positive relation in common with (h, t), and p is termed a *positive* example of (h, t) (also $(h, t) \notin \mathcal{S}_{h,t}$). The *negative* examples of (h, t) are the remaining examples in the batch, i.e., $\mathcal{B} \setminus (\mathcal{S}_{h,t} \cup \{(h, t)\})$. The operator $\cdot$ refers to the dot product, and $\tau \in \mathbb{R}^+$ is a temperature parameter. To minimize $\mathcal{L}_{scl}^{h,t}$, we maximize the numerator in Equation 8 by learning embeddings for (h, t) and its positive examples that are close to each other (according to cosine similarity), and minimize the denominator by learning embeddings for (h, t) and its negative examples that are far apart.

Equation 8 would work for document-level relation extraction (DocRE) if not for the long-tail phenomenon that is typically present in DocRE datasets. For example, in the datasets used for our experiments, the top 10 relations account for about 60% of entity pairs in the dataset. Thus, we often find that an entity pair (h, t) with only long-tailed positive relations does not have any other entity pair in the same batch that has that relation in common, i.e., $|\mathcal{S}_{h,t}| = 0$. This means that Equation 8 could not be applied to such entity pairs. To take such an entity pair (h, t) into account, we design the following loss term:

$$\mathcal{L}_{lt}^{h,t} = \log \sum_{d \in \mathcal{B}, d \neq (h,t)} \exp(\boldsymbol{x}_{h,t} \cdot \boldsymbol{x}_d / \tau), \qquad (9)$$

in which we solely maximize the dissimilarities between the embedding of (h, t) and those of other entity pairs in the same batch $\mathcal{B}$. The final loss function for supervised contrastive learning is:

$$\mathcal{L}_2 = \sum_{(h,t) \in \mathcal{B}_\mathcal{P}} \mathbb{I}_{\{|\mathcal{S}_{h,t}| \neq 0\}} \mathcal{L}_{scl}^{h,t} + \mathbb{I}_{\{|\mathcal{S}_{h,t}| = 0\}} \mathcal{L}_{lt}^{h,t}, \quad (10)$$

where $\mathbb{I}_{\{\}}$ is an indicator function that takes the value of 1 if the condition in $\{\}$ is satisfied, and the value of 0 otherwise. In Equation 10, $\mathcal{B}_\mathcal{P} \subseteq \mathcal{B}$ is a subset of entity pairs in a batch that is labeled with at least one relation in $\mathcal{R}$. In other words, $\mathcal{B}_\mathcal{P}$ does not contain any entity pair that is labeled with the `NA` class (i.e., all relations in $\mathcal{R}$ are considered negative for the entity pair), since it does not make sense to minimize the embedding distance between two entity pairs that are labeled `NA`, and thus have no relation in common.

Combining Equations 7 and 10, we obtain the final loss function that is used for training our model:

$$\mathcal{L} = \mathcal{L}_1 + \lambda \mathcal{L}_2, \qquad (11)$$

where $\lambda \in \mathbb{R}^+$ is a hyperparameter.

3.5 Negative Label Sampling

As reported by recent works (Huang et al., 2022; Tan et al., 2022b), the DocRE benchmark suffers from the severe false-negative problem, which means that quite a few entity pairs previously labeled as `NA` class should have at least one relation label. Blithely ignoring this issue will greatly harm the performance of the method and cause ill-defined evaluation. To enhance the robustness of our method, we propose a novel negative label sampling strategy, which only samples a small fraction of negative relations for each entity pair with `NA` label when computing the loss function. We assume that the true relation labels for those false-negative examples are hard to be sampled from the massive negative relations, thus we could avoid erroneously treating the correct labels as negative relations in the loss function.

Let $\mathcal{B}_\mathcal{N} \subseteq \mathcal{B}$ denote the subset of all entity pairs that are labeled `NA` in a current batch $\mathcal{B}$. For each entity pair (h, t) in $\mathcal{B}_\mathcal{N}$, we uniformly sample a subset of negative relations $\mathcal{N}'_{h,t} \subseteq \mathcal{N}_{h,t}$, and define the following loss function:

$$\mathcal{L}' = \sum_{(h,t) \in \mathcal{B}_\mathcal{N}} \sum_{r \in \mathcal{N}'_{h,t}} -\log P^{\eta}_{h,t}(r) + \frac{1}{\gamma_2} \sum_{r \in \mathcal{N}'_{h,t}} H_{h,t}(r), \quad (12)$$

where $P^{\eta}_{h,t}(r)$ and $H_{h,t}(r)$ are defined in Equation 3 and Equation 5 respectively.

Let $\mathcal{B}_\mathcal{P} = \mathcal{B} \setminus \mathcal{B}_\mathcal{N}$ denote the subset of entity pairs in the current batch $\mathcal{B}$ that is labeled with at least one positive relation. Combining terms in Equations 7, 10, and 12, we obtain the final loss function $\mathcal{L}^{\texttt{NA}}$ that incorporates our sampling approach:

$$\begin{aligned} \mathcal{L}_1^{\texttt{NA}} &= \mathcal{L}' + \sum_{(h,t) \in \mathcal{B}_\mathcal{P}} \mathcal{L}_{pmt}^{h,t} + \mathcal{L}_{em}^{h,t}, \\ \mathcal{L}^{\texttt{NA}} &= \mathcal{L}_1^{\texttt{NA}} + \lambda \mathcal{L}_2. \end{aligned} \qquad (13)$$

Observe that $\mathcal{L}_1^{\texttt{NA}}$ has modified $\mathcal{L}_1$ (Equation 7) by changing the latter's sum over negative relations for entity pairs that are labeled `NA`. Also note that the loss $\mathcal{L}_2$ due to supervised contrastive learning remains unchanged in Equation 13 because it operates at the level of entities (specifically their embeddings) rather than at the level of relation labels.

Although previous papers (Li et al., 2021b, 2022) seem to adopt a similar negative sampling strategy, our approach has significant differences from them. The previous works sampled negative instances (i.e., entire entity pairs with `NA` labels in our case) and removed those unselected negative instances from the training dataset. In our approach, we sample negative *labels* of `NA` entity pairs, and do not discard any entity pairs, making our approach potentially more data efficient.

4 Experiments

4.1 Benchmark Description

DocRED (Yao et al., 2019) is a large-scale dataset constructed from Wikipedia and Wikidata, and is widely used as a benchmark for document-level relation extraction (DocRE). However, recent studies (Huang et al., 2022; Tan et al., 2022b) have found that many entity pairs (or examples) that are labeled `NA` are erroneous, and should be instead labeled with at least one positive relation in $\mathcal{R}$. To ameliorate this problem, Re-DocRED (Tan et al., 2022b) relabels the original training and development sets of DocRED and splits DocRED's development set into two equal halves as new development and test sets, respectively. Instead of comparing models on the faulty DocRED dataset, the results on Re-DocRED should be regarded as a fair comparison.

4.2 Two New Data Regimes

To evaluate the models in a more realistic experimental setting in which their resilience to noisy data is carefully tested, we propose two new data regimes, OOG-DocRE and OGG-DocRE, that are based on the above DocRED and Re-DocRED benchmarks. Every "O" represents the **O**riginal labels obtained from the original unclean, noisy DocRED dataset; similarly, every "G" represents the **G**old labels in the new, cleaned Re-DocRED dataset. Each letter in "OOG" and "OGG" represent different sources of labels for training, validation, and testing, respectively. Both regimes reflect the real-world scenario where training data is noisy, and manual effort can only be expended on cleaning a relatively small validation/test set. All models are trained and tuned only on the training and validation sets respectively, and evaluated on the test set. Note that in both regimes the cleaned training set from Re-DocRed is not used. Table 1 contains details about the datasets.

Dataset	**Train** #Doc / #Example	**Dev** #Doc / #Example	**Test** #Doc / #Example
DocRED	3,053 / 1,198,650	1,000 / 396,790	1,000 / 392,158
Re-DocRED	3,053 / 1,193,092	500 / 193,232	500 / 198,670
Our new data regimes			
OOG-DocRE	3,053 / 1,198,650	500[3]/ 195,682	500 / 198,670
OGG-DocRE	3,053 / 1,198,650	500 / 193,232	500 / 198,670

Table 1: Dataset statistics. We construct two new data regimes based on the **O**riginal labels from DocRED and **G**old labels from Re-DocRED. The total number of predefined relation labels for all datasets is 96 (i.e., $|\mathcal{R}| = 96$).

4.3 Results on DocRE Benchmarks

From Table.2, we see that our PEMSCL model performs the best on both development and test sets of the original DocRED dataset and the cleaned Re-DocRED dataset (the models are trained on their corresponding training sets). It is worth noting that the results among recent models (e.g., DocuNet, KD-DocRE, NCRL) are almost indistinguishable on DocRED (see Appendix C), especially after considering their standard deviations. However, the performance gaps between models become significant when we validate and test on the Re-DocRED dataset. This strongly suggests that the original DocRED's (overly erroneous) development and test sets cannot truly ascertain the performance differences between models. In contrast, the cleaned version Re-DocRED provides a more faithful comparison of the models. Henceforth, we analyze model performances based solely on Re-DocRED's development and test sets.

Compared with ATLOP (upon which our model is developed), our PEMSCL model achieves around a 3-point improvement in terms of both Ign F_1 and F_1 scores on Re-DocRED's development and test sets. When compared against the recent strong baseline NCRL, our PEMSCL model continues to do better, achieving about a 1-point improvement in terms of F_1 score on Re-DocRED's development set. After taking the standard deviations into account, the results still show that PEMSCL outperforms NCRL. In sum, the above results demonstrate the effectiveness of our proposed model, and ascertain that it has achieved new state-of-the-art performances.

4.4 Ablation Study

In addition to the main metrics F_1 and Ign F_1, we also report the F_1 scores for different types of relations. We first rank in descending order all pre-

---

[3] 500 documents share the same titles as the development set of Re-DocRED, but labeled by DocRED.

	DocRED Dev		DocRED Test	
Model	Ign F_1	F_1	Ign F_1	F_1
Implemented on DeBERTa$_{Large}$				
ATLOP (Zhou et al., 2021)	62.16±0.15	64.01±0.12	62.12	64.08
ATLOP + BCE (Zhou and Lee, 2022)	61.92±0.13	63.96±0.15	61.83	63.92
NCRL (Zhou and Lee, 2022)	62.98±0.18	64.79±0.13	63.03	64.96
PEMSCL (Ours)	**63.25±0.09**	**65.15±0.10**	**63.40**	**65.41**
	Re-DocRED Dev		Re-DocRED Test	
	Ign F_1	F_1	Ign F_1	F_1
Implemented on RoBERTa$_{Large}$				
JEREX (Eberts and Ulges, 2021)	69.12	70.33	68.97	70.25
ATLOP + BCE* (Zhou and Lee, 2022)	75.86±0.13	75.25±0.11	75.91	75.36
ATLOP (Zhou et al., 2021)	76.88	77.63	76.94	77.73
DocuNet (Zhang et al., 2021)	77.53	78.16	77.27	77.92
KD-DocRE (Tan et al., 2022a)	77.92	78.65	77.63	78.35
NCRL* (Zhou and Lee, 2022)	78.41±0.21	79.15±0.20	78.45	79.19
PEMSCL (Ours)	**79.02±0.20**	**79.89±0.17**	**79.01**	**79.86**

Table 2: Results on DocRED and Re-DocRED. Ign F_1 stands for the F1 score excluding relational facts in the training set. Results for baseline models on the test and dev set of DocRED are taken from their original papers. The results on RE-DocRED for NCRL and ATLOP + BCE (Zhou and Lee, 2022) (i.e., marked with *) are reproduced by us with their default code[4] and our implementation, respectively; other results of baselines on Re-DocRED are taken from (Tan et al., 2022b). We report the mean and standard deviation on the development set of 5 runs with different random initialization for our PEMSCL model and the reproduced baselines, and report the test scores using the best-performing model on the development set. For implementation details, please refer to Appendix A.

Model	Dev Ign F_1	Dev F_1	Head F_1	Mid F_1	Tail F_1
Ours	**79.02**	**79.89**	**82.99**	**75.70**	**63.51**
– $\mathcal{L}_{em}^{h,t}$	78.38	79.17	82.35	74.75	62.35
– $\mathcal{L}_2$	78.36	79.10	82.40	74.50	62.22
– $\mathcal{L}_{em}^{h,t}$ and $\mathcal{L}_2$	77.92	78.63	81.92	74.06	61.16

Table 3: Ablation study of our PEMSCL model on Re-DocRED. "–" represents the removal of our model's components. We also report the F_1 scores for the top 10 relations (Head F_1), the middle 70 relations (Mid F_1), and the last 20 relations (Tail F_1) ranked by the number of entity pairs that are related by them. The mean result of 3 runs with different random initialization on the development set of Re-DocRED are reported.

defined relations by the number of entity pairs that are labeled with them. Next, we classify them into three categories: *head* relations (the top 10 relations, accounting for 64% of Re-DocRED's training data), *tail* relations (the bottom 20 relations, accounting for 2% of training data), and *middle* relations (the remaining relations).

From Table 3, we see that each component plays a pivotal role in the effectiveness of our PEMSCL model – removing a component or a combination of them compromises performance. Removing the $\mathcal{L}_{em}^{h,t}$ and $\mathcal{L}_2$ components individually results in a performance decline of 0.90% and 0.99% in terms of F_1 score respectively. When either of these two components is removed, we see a sharper decline in terms of Tail F_1 (1.82% and 2.03%) than in terms of Head F_1 (0.77% and 0.71%). This shows that both components are useful for long-tailed relations, and highlights the effectiveness of $\mathcal{L}_2$, part of which is designed to cater to long-tailed relations. After removing both $\mathcal{L}_{em}^{h,t}$ and $\mathcal{L}_2$ together, the performances on Head F_1, Mid F_1, and Tail F_1 all significantly drop by 1.29%, 2.17%, and 3.70% respectively. Even with only one loss term remaining (i,e, $\mathcal{L}_{pmt}^{h,t}$), our EMSCL model still surpasses the baseline ATLOP on Re-DocRED in Table 2 by 1.3% on Dev F_1, reflecting the usefulness of our pairwise moving-threshold loss.

4.5 Results on New Data Regimes

Table 4 shows the performance of our PEMSCL model and baselines on our proposed data regimes: OOG-DocRE and OGG-DocRE. We select NCRL as a focal baseline from among the recent baselines

---

[4] `https://github.com/yangzhou12/NCRL`

	Orig-Dev		Gold-Dev		Gold-Test	
	Ign F_1	F_1	Ign F_1	F_1	Ign F_1	F_1
On OOG-DocRE Regime						
ATLOP (Zhou et al., 2021)	60.94	62.95	46.99	47.14	47.52	47.65
NCRL (Zhou and Lee, 2022)	61.42	63.52	49.06	49.21	48.41	48.53
PEMSCL (Ours)	**62.05**	**64.19**	50.82	50.99	50.92	51.10
PEMSCL† (Ours)	46.07	49.51	**62.05**	**63.39**	**62.76**	**64.03**
On OGG-DocRE Regime						
ATLOP (Zhou et al., 2021)	-	-	48.23	48.54	48.50	48.77
NCRL (Zhou and Lee, 2022)	-	-	49.92	50.08	50.10	50.25
PEMSCL (Ours)	-	-	50.43	50.62	51.09	51.25
PEMSCL† (Ours)	-	-	**62.40**	**63.72**	**62.47**	**63.73**

Table 4: Results on two new data regimes. The best results are **bolded**, and the second best results are underlined. PEMSCL† refers to our best-performing model on the development set after using our proposed negative label sampling strategy (Section 3.5). The sampling ratio of 0.1 is set with a development set.

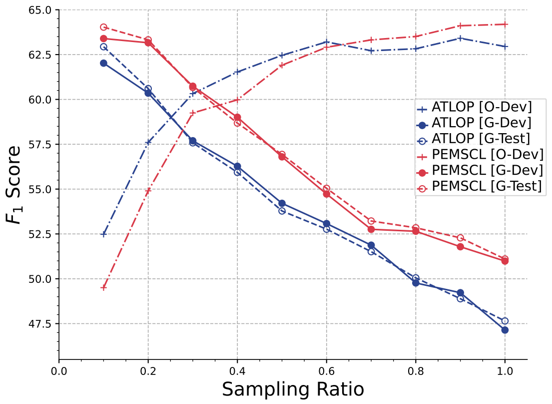


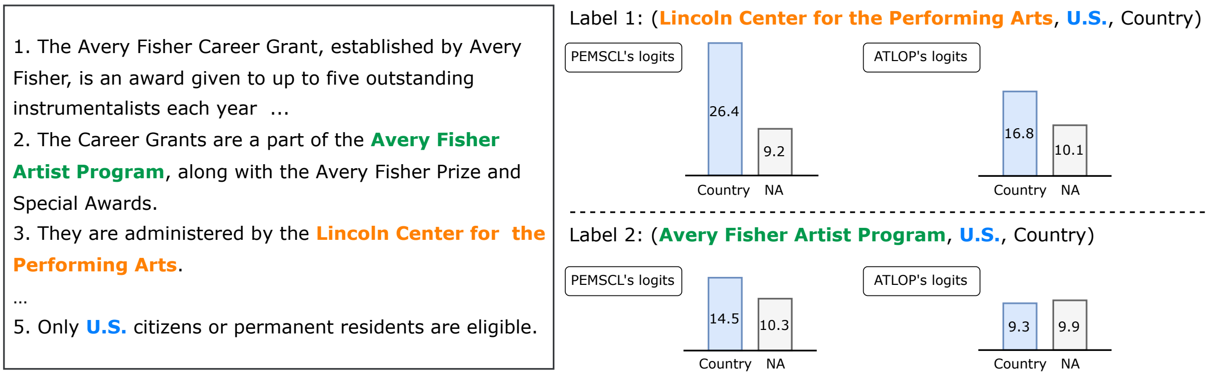
Figure 2: The effect of negative label sampling ratio in the OOG-DocRE regime.

due to its competitive performance on Re-DocRED. We see that all models perform above 62-point F_1 on Orig-Dev (the original development set from DocRED) when trained on the original training dataset. However, when we evaluate all models on Gold-Dev and Gold-Test (the clean development and test sets from Re-DocRED), the performances of the models (including ours) dramatically decrease by around 15-point F_1 on both Gold-Dev and Gold-Test. Upon inspection, we find that the models misclassify a lot of positive examples (entity pairs) as `NA`, which is an expected outcome of being misguided by the erroneous false-negative labels in DocRED.

However, after using our proposed negative label sampling loss (Equation 13), our PEMSCL model exhibits tremendous improvement on both Gold-Dev and Gold-Test by 24% and 25% on the F_1 scores respectively. This demonstrates the effectiveness of our negative label sampling strategy in countering the noise present in entity pairs that are labeled `NA`. The same conclusion can be drawn from the results for the OGG-DocRE regime. Moreover, we notice that both ATLOP and NCRL improve by at least 1-point F_1 on the OGG regime compared with the OOG regime. This demonstrates the usefulness of gold labels even in a small amount. However, our model performs comparably on both regimes, indicating the stability of our model in different regimes.

We also investigate the effect of the sampling ratio on our proposed strategy. The sampling ratio refers to the ratio of negative labels that we keep during the training for each entity pair that is labeled `NA`. We apply our negative label sampling approach on both ATLOP and our PEMSCL model. As seen from Figure 2, PEMSCL consistently performs better than ATLOP by a clear margin. We also find that the performances of the models on Gold-Dev and Gold-Test gradually decrease as the sampling ratio is increased. This is because as we keep more (purportedly) negative labels in our loss function, the risk of wrongly penalizing potentially true labels increases concomitantly.

We observe that the sampling ratio has the opposite effect on Orig-Dev. As the sampling ratio increases, the F_1 on Orig-Dev increases, leading one to mistakenly conclude that a large sampling rate should be used. This provides strong evidence of the poor data quality in DocRED, and shows



1. The Avery Fisher Career Grant, established by Avery Fisher, is an award given to up to five outstanding instrumentalists each year ...
2. The Career Grants are a part of the **Avery Fisher Artist Program**, along with the Avery Fisher Prize and Special Awards.
3. They are administered by the **Lincoln Center for the Performing Arts**.

...

5. Only **U.S.** citizens or permanent residents are eligible.

Figure 3: Case Study.

how it can misguide training and lead to poor results. The results on the OGG-DocRE regime are similar (see Appendix B for details).

4.6 Case Study

Figure 3 shows a case study of our proposed PEMSCL model and the baseline ATLOP model. We can see that for the entity pair (*Lincoln Center for the Performing Arts*, *U.S.*), both models successfully detect the correct relation label, i.e., `Country`. However, the logit difference between the `Country` relation and the threshold label `NA` in our model is much larger than that of the ATLOP model (26.4-9.2 > 16.8-10.1). This demonstrates that our model is capable of learning a more differentiated distribution of the final probability scores. For the entity pair of (*Avery Fisher Artist Program*, *U.S.*), the ATLOP model fails to correctly predict its label and classifies it as `NA` class since the logit of `Country` is lower than that of the threshold class (9.3 < 9.9). However, our model not only correctly predicts its correct label, but also maximizes the discriminability of the prediction scores (14.5 vs 10.3).

5 Conclusions

In this paper, we propose a novel method for DocRE problem called PEMSCL, which contains a pairwise moving-threshold loss with entropy minimization, adapted supervised contrastive learning, and a novel negative sampling strategy, to achieve good integration of both discriminability and robustness. Experimental results show that our method achieves new state-of-the-art results.

Acknowledgements

This research is supported by Singapore Ministry of Education's AcRF Tier 1 Grant (R-253-000-146-133) to Stanley Kok. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors, and do not reflect the views of the funding agencies.

Limitations

First, we require a large amount of GPU resources to conduct our experiments because we deal with large document-based datasets (whose input text is significantly longer than those of traditional sentence-level tasks). Second, we implement our model on two large pre-trained language models, Roberta-large (Liu et al., 2019) and Deberta-large (He et al., 2021), both of which also have a large GPU footprint. Third, the performance of our adapted supervised contrastive learning component is dependent on GPU batch size (a larger batch size allows more contrastive examples to be used to learn better embeddings).

References

David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. 2020. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *Proceedings of ICLR*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of ICML*.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Connecting the dots: Document-level neural relation extraction with edge-oriented graphs. In *Proceedings of EMNLP/IJCNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Markus Eberts and Adrian Ulges. 2021. An end-to-end model for entity-level relation extraction using multi-instance learning. In *Proceedings of EACL*.

Yves Grandvalet and Yoshua Bengio. 2004. Semi-supervised learning by entropy minimization. In *Proceedings of Neural Information Processing Systems*.

Yves Grandvalet and Yoshua Bengio. 2005. Semi-supervised learning by entropy minimization. In *Proceedings of Neural Information Processing Systems*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of CVPR*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *Proceedings of ICLR*.

Quzhe Huang, Shibo Hao, Yuan Ye, Shengqi Zhu, Yansong Feng, and Dongyan Zhao. 2022. Does recommend-revise produce reliable annotations? an analysis on missing instances in docred. In *Proceedings of ACL*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Proceedings of NeurIPS*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*.

Bo Li, Wei Ye, Canming Huang, and Shikun Zhang. 2021a. Multi-view inference for relation extraction with uncertain knowledge. In *Proceedings of AAAI*.

Yangming Li, Lemao Liu, and Shuming Shi. 2021b. Empirical analysis of unlabeled entity problem in named entity recognition. In *Proceedings of ICLR*.

Yangming Li, Lemao Liu, and Shuming Shi. 2022. Rethinking negative sampling for handling missing entity annotations. In *Proceedings of ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Guoshun Nan, Zhijiang Guo, Ivan Sekulic, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. In *Proceedings of ACL*.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Trans. Assoc. Comput. Linguistics*, 5:101–115.

Yujia Qin, Yankai Lin, Ryuichi Takanobu, Zhiyuan Liu, Peng Li, Heng Ji, Minlie Huang, Maosong Sun, and Jie Zhou. 2021. ERICA: improving entity and relation understanding for pre-trained language models via contrastive learning. In *Proceedings of ACL/IJCNLP*.

Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of EACL*.

Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Inter-sentence relation extraction with document-level graph convolutional neural network. In *Proceedings of ACL*.

Qingyu Tan, Ruidan He, Lidong Bing, and Hwee Tou Ng. 2022a. Document-level relation extraction with adaptive focal loss and knowledge distillation. In *Findings of ACL*.

Qingyu Tan, Lu Xu, Lidong Bing, Hwee Tou Ng, and Sharifah Mahani Aljunied. 2022b. Revisiting docred - addressing the false negative problem in relation extraction. In *Proceedings of EMNLP*.

Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. 2019. ADVENT: adversarial entropy minimization for domain adaptation in semantic segmentation. In *Proceedings of CVPR*.

Difeng Wang, Wei Hu, Ermei Cao, and Weijian Sun. 2020. Global-to-local neural networks for document-level relation extraction. In *Proceedings of EMNLP*.

Yuxin Xiao, Zecheng Zhang, Yuning Mao, Carl Yang, and Jiawei Han. 2022. SAIS: supervising and augmenting intermediate steps for document-level relation extraction. In *Proceedings of NAACL*.

Yiqing Xie, Jiaming Shen, Sha Li, Yuning Mao, and Jiawei Han. 2022. Eider: Empowering document-level relation extraction with efficient evidence extraction and inference-stage fusion. In *Findings of ACL*.

Benfeng Xu, Quan Wang, Yajuan Lyu, Yong Zhu, and Zhendong Mao. 2021. Entity structure within and throughout: Modeling mention dependencies for document-level relation extraction. In *Proceedings of AAAI*.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. Docred: A large-scale document-level relation extraction dataset. In *Proceedings of ACL*.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential reasoning learning for language representation. In *Proceedings of EMNLP*.

Shuang Zeng, Yuting Wu, and Baobao Chang. 2021. SIRE: separate intra- and inter-sentential reasoning for document-level relation extraction. In *Findings of ACL/IJCNLP*.

Ningyu Zhang, Xiang Chen, Xin Xie, Shumin Deng, Chuanqi Tan, Mosha Chen, Fei Huang, Luo Si, and Huajun Chen. 2021. Document-level relation extraction as semantic segmentation. In *Proceedings of IJCAI*.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proceedings of AAAI*.

Yang Zhou and Wee Sun Lee. 2022. None class ranking loss for document-level relation extraction. In *Proceedings of IJCAI*.

A Implementation Details

We mainly implement our method using the pre-trained RoBERTa-large (Liu et al., 2019) as the encoder model. Due to limited computational resources, we only use a larger model DeBERTa-large (He et al., 2021) on the DocRED benchmark. We conduct grid search for the temperature parameter τ and the loss coefficient λ ($\{0.1, 0.2, 0.5, 1.0, 2.0\}$), learning rate ($\{1e\text{-}5, 2e\text{-}5, 3e\text{-}5\}$), and warmup ratio of optimizer ($\{0.02, 0.06, 0.10\}$). We implement our model in the PyTorch version of Huggingface Transformers[5], and run all experiments on a NVIDIA Quadro RTX 8000 GPU. The best hyperparameters used in our experiments are shown in Table5.

B The Effect of Sampling Ratio on the OGG- DocRE Setting

We analyze the effect of the negative label sampling ratio in the OGG-DocRE regime, which is shown in Figure 4. It presents a similar pattern with that of the OOG-DocRE regime as described in Section 4.5.

C Results on the DocRED Dataset

We provide the results of RoBERTa-large based models on DocRED in Table 6 for a complete comparison. However, these results can not reflect a faithful performance comparison due to the preponderance of erroneous labels in the DocRED dataset. Instead, the results on the Re-DocRED dataset should be taken as a reliable fair comparison.

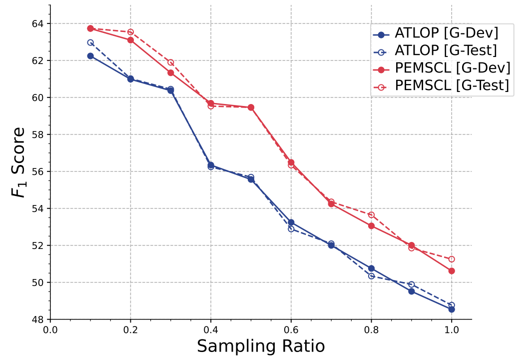


Figure 4: The effect of negative label sampling ratio in the OGG-DocRE regime.

D Background: ATLOP Encoder

For every document, the encoder model first marks each entity mention with a special token "*" at its start and end positions, and then feeds the resulting document $D = \{w_l\}_{l=1}^{L}$ into a pre-trained language model (PLM) to obtain contextual embeddings for each of the document's L tokens: $\mathbf{H} = [\boldsymbol{h}_1, \dots, \boldsymbol{h}_L] = \text{PLM}([w_1, \dots, w_L])$ where $\boldsymbol{h}_l \in \mathbb{R}^d$. ATLOP regards the embedding of "*" at the start position of mention m_j^i as its representation, i.e., $\boldsymbol{h}_{I(m_j^i)}$, where $I(.)$ is a function mapping a mention m_j^i to the index of its representative "*" in $\mathbf{H}$. Next, the embedding $\boldsymbol{h}_{e_i} \in \mathbb{R}^d$ of each entity e_i is obtained with logsumexp pooling:

$$\boldsymbol{h}_{e_i} = \log \sum_{j=1}^{\mathcal{M}_{e_i}} \exp\left(\boldsymbol{h}_{I(m_j^i)}\right).$$

For each entity pair (e_h, e_t), ATLOP uses the token-level dependencies present within its multi-head self-attention mechanism to compute a *localized contextual* embedding $\boldsymbol{c}_{h,t} \in \mathbb{R}^d$, capturing the contextual information that is relevant to *both* entities e_h and e_t. Due to space constraints, we refer readers to Zhou et al. (2021) for details on how $\boldsymbol{c}_{h,t}$ is computed.

For each entity pair (e_h, e_t), the encoder will generate the final representation $\boldsymbol{x}_{h,t}$ for the pair, and its corresponding vector of unnormalized prediction scores $\boldsymbol{f}_{h,t} \in \mathbb{R}^{|\mathcal{R}|+1}$ for all relations in $\mathcal{R} \cup \{\texttt{NA}\}$ as follows:

$$[\boldsymbol{z}_h^1; \dots; \boldsymbol{z}_h^P] = \boldsymbol{z}_h = \tanh\left(\boldsymbol{W}_h \boldsymbol{h}_{e_h} + \boldsymbol{W}_{c_1} \boldsymbol{c}_{h,t}\right),$$

$$[\boldsymbol{z}_t^1; \dots; \boldsymbol{z}_t^P] = \boldsymbol{z}_t = \tanh\left(\boldsymbol{W}_t \boldsymbol{h}_{e_t} + \boldsymbol{W}_{c_2} \boldsymbol{c}_{h,t}\right),$$

$$\boldsymbol{x}_{h,t} = ||_{p=1}^{P} (\boldsymbol{z}_h^p \otimes \boldsymbol{z}_t^p), \tag{14}$$

$$\boldsymbol{f}_{h,t} = \boldsymbol{W}_o \boldsymbol{x}_{h,t} + \boldsymbol{b}_o, \tag{15}$$

---

[5] `https://huggingface.co/`

Dataset	Batch size	Epoch	Warmup ratio	Learning rate	τ	λ	γ_1	γ_2
DocRED	4	5	0.10	2e-5	2.0	2.0	1	1
Re-DocRED	4	8	0.06	2e-5	0.2	0.1	$\lvert\mathcal{P}_{h,t}\rvert$	$\lvert\mathcal{N}_{h,t}\rvert$

Table 5: Best hyperparameters for benchmarks.

	DocRED Dev		DocRED Test	
Model	Ign F_1	F_1	Ign F_1	F_1
Implemented on RoBERTa$_{Large}$				
Coref (Ye et al., 2020)	57.35	59.43	57.90	60.25
SSAN (Xu et al., 2021)	60.25	62.08	59.47	61.42
ATLOP (Zhou et al., 2021)	61.32±0.14	63.18±0.19	61.39	63.40
DocuNet (Zhang et al., 2021)	62.23±0.12	64.12±0.14	62.39	**64.55**
KD-DocRE (Tan et al., 2022a)	62.16±0.10	64.19±0.16	**62.57**	64.28
NCRL (Zhou and Lee, 2022)	62.21±0.22	64.18±0.20	61.94	64.14
PEMSCL (Ours)	**62.31±0.19**	**64.21±0.17**	62.17	64.28

Table 6: Experimental results on the DocRED dataset.

where $\boldsymbol{z}_h, \boldsymbol{z}_t \in \mathbb{R}^{d_1}$ are split into P equal-sized groups $[\boldsymbol{z}_h^1; \ldots; \boldsymbol{z}_h^P]$ and $[\boldsymbol{z}_t^1; \ldots; \boldsymbol{z}_t^P]$ respectively; $\boldsymbol{W}_{\{h,t,c_1,c_2\}} \in \mathbb{R}^{d_1 \times d}$, $\boldsymbol{W}_o \in \mathbb{R}^{(|\mathcal{R}|+1) \times d_x}$, $\boldsymbol{x}_{h,t} \in \mathbb{R}^{d_x}$ ($d_x = \frac{d_1 \times d_1}{P}$), and $\boldsymbol{b}_o \in \mathbb{R}^{|\mathcal{R}|+1}$ are *learnable* parameters (in our model too); $\otimes$ is the outer product operator; and the operators ; and $||$ respectively represent the concatenation of vectors and matrices. The elements in $\boldsymbol{f}_{h,t}$ are logits that our model feeds pairwise into (not necessarily the same) softmax functions to obtain relative probabilities between relations (Section 3.3).

Penguins Don't Fly:
Reasoning about Generics through Instantiations and Exceptions

Emily Allaway* **Jena D. Hwang**† **Chandra Bhagavatula**†
Kathleen McKeown* **Doug Downey**†¶ **Yejin Choi**†‡
*Columbia University, USA
†Allen Institute for Artificial Intelligence, USA
¶Northwestern University, USA
‡Paul G. Allen School of Computer Science & Engineering, University of Washington, USA
`eallaway@cs.columbia.edu`

Abstract

Generics express generalizations about the world (e.g., birds can fly) that are not universally true (e.g., newborn birds and penguins cannot fly). Commonsense knowledge bases, used extensively in NLP, encode some generic knowledge but rarely enumerate such exceptions and knowing when a generic statement holds or does not hold true is crucial for developing a comprehensive understanding of generics. We present a novel framework informed by linguistic theory to generate EXEMPLARS—specific cases when a generic holds true or false. We generate ~19k exemplars for ~650 generics and show that our framework outperforms a strong GPT-3 baseline by 12.8 precision points. Our analysis highlights the importance of linguistic theory-based controllability for generating exemplars, the insufficiency of knowledge bases as a source of exemplars, and the challenges exemplars pose for the task of natural language inference.

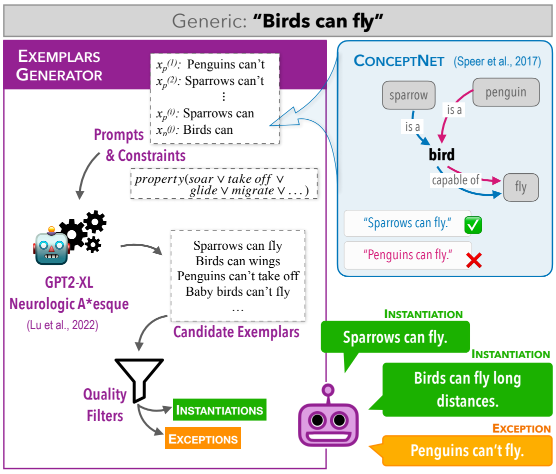


Figure 1: We present **EXEMPLARS generator**: given a generic like "Birds can fly" it generates truthful statements where the generic does (INSTANTIATIONS) and does not (EXCEPTIONS) hold. We extract commonsense knowledge (e.g., from ConceptNet (Speer et al., 2017)) in linguistically-informed prompts and constraints for constrained generation (Lu et al., 2022). We use trained discriminators to filter for quality.

1 Introduction

Generics express generalizations (e.g., birds can fly) that allow humans to reason and act with incomplete world knowledge (Asher and Morreau, 1995). Generics allow us to draw plausible inferences about individuals (e.g., Polly is a bird, so Polly can fly) *even when we know of counterexamples* (e.g., penguins cannot fly). Despite the utility of generalizations, knowledge of counterexamples is necessary for modeling generics and effectively reasoning with them in computational systems.

Recent studies of generics (e.g., Bhagavatula et al., 2022; Bhakthavatsalam et al., 2020) and commonsense KBs (e.g., Speer et al., 2017) provide repositories of generic knowledge. However, these resources rarely mention EXCEPTIONS (i.e., counterexamples) or INSTANTIATIONS (i.e., cases where the generic holds); collectively EXEMPLARS. For systems using these resources as a source of world knowledge, such incomplete information can lead to incorrect deductions (e.g., if Polly is a penguin, which is a bird, then inferring that Polly can fly because birds can fly is false[1]). Therefore, we propose a novel computational framework that operationalizes linguistic theories in order to automatically generate EXEMPLARS.

In our work, we unify two distinct linguistic theories and use linguistic theory-guided decoding to generate EXEMPLARS. Although large-scale neural language models such as GPT-3 (Brown et al., 2020) have been increasingly successful in few-shot text generation tasks, such generation is both expensive and not easily controllable. Therefore, we instead use the constrained generation algorithm Neurologic A*esque (Lu et al., 2022), which can be applied to any auto-regressive language model; we choose to use GPT-2 (Radford et al., 2019).

---

[1] penguin is a bird $\land$ birds can fly $\implies$ penguins can fly

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2618–2635
May 2-6, 2023 ©2023 Association for Computational Linguistics

In this manner, we generate 12562 INSTANTIATIONS and 6297 EXCEPTIONS for ∼650 generic statements from Bhagavatula et al. (2022). We conduct human evaluation of the generations and show that our system outperforms few-shot generation by GPT-3 by 12.8 precision points. Our analyses demonstrate not only the importance of linguistic modeling for generating EXEMPLARS and the insufficiency of KBs as a source of EXEMPLARS, but also the challenges EXEMPLARS pose for natural language reasoning.

Our contributions are as follows: **(1)** we present a novel framework grounded in linguistic theory for representing generics and their EXEMPLARS, **(2)** we present the first method to automatically generate generic EXEMPLARS and show it outperforms a competitive baseline based on GPT-3 and **(3)** we present analysis showing the importance of explicit linguistic modeling for this task and the insufficiency of current NLI methods for generics. Our system and data are publicly available[2].

2 Related Work

Theory Generics have been studied extensively in semantics, philosophy, and psychology to develop a single logical form for all generics (Lewis and Keenan, 1975; Carlson, 1977, 1989; Krifka, 1987) or a probabilistic definition (Cohen, 1996, 1999, 2004; Kochari et al., 2020), categorize generics (Leslie, 2007, 2008; Khemlani et al., 2009), and analyze specific types (Prasada and Dillingham, 2006, 2009; Haward et al., 2018; Mari et al., 2012; Krifka et al., 2012). Mechanisms to tolerate EXCEPTIONS have also been proposed (Kadmon and Landman, 1993; Greenberg, 2007; Lazaridou-Chatzigoga and Stockall, 2013) but these are primarily theoretical and use carefully chosen examples. In contrast, our work combines these EXCEPTION tolerance mechanisms with generic categorization and proposes a novel, large-scale, computational framework for EXEMPLARS.

Commonsense Knowledge While large-scale CKBs capture a range of commonsense knowledge (Speer et al., 2017; Sap et al., 2019; Forbes et al., 2020; Hwang et al., 2021), they contain necessarily incomplete (i.e., the open-world assumption (Reiter, 1978b)) *general* knowledge. Furthermore, although recent works have created KBs specifically of generics (Bhakthavatsalam et al., 2020; Bhagavatula et al., 2022) and proposed methods to identify generics in text (Friedrich et al., 2015, 2016), they do not identify or model EXEMPLARS. In our work, we focus directly on automatically generating EXEMPLARS, providing richer commonsense knowledge.

The application of generics to specific individuals is influenced by prototypicality (Rips, 1975; Osherson et al., 1990), with small sets of prototypical norms collected in cognitive science for a range of kinds (Devereux et al., 2014; McRae et al., 2005; Overschelde et al., 2004). However, recent work has shown that neural models have only moderate success at mimicking human prototypicality (Misra et al., 2021; Boratko et al., 2020) or producing commonsense facts without guidance (Petroni et al., 2019). Hence, we combine neural models with a KB of concepts, using linguistic-theory-guided decoding, to generate generics EXEMPLARS.

Reasoning Reasoning with generics is closely related to non-monotonic reasoning (Ginsberg, 1987b,a); specifically default inheritance reasoning (Brewka, 1987; Hanks and McDermott, 1986; Horty and Thomason, 1988; Imielinski, 1985; Poole, 1988; Reiter, 1978a, 1980). Contrary to the proposed solutions for linguistic tests on default inheritance reasoning (Lifschitz, 1989, e.g.,can a conclusion about inheritance be inferred based on provided evidence?), later works showed that the presence of generics EXEMPLARS in the evidence impacts what humans perceive as the correct answer (Elio and Pelletier, 1996; Pelletier and Elio, 2005; Pelletier, 2009). These results highlight the importance of identifying generics and accurately modeling their relationships in machine reasoning.

While natural language inference (NLI), a form of deductive reasoning well-studied in NLP (i.a., Dagan et al. (2013); Bowman et al. (2015)), captures notions of inference, studies on non-monotonic reasoning and NLI are limited (Wang et al., 2019; Cooper et al., 1994; Yanaka et al., 2019b,a; Rudinger et al., 2020) and do not include default inheritance reasoning. Therefore, in this work we analyze the interactions between generics EXEMPLARS and NLI and highlight the importance of modeling this relationship in machine reasoning.

3 Framework for EXEMPLARS

A generic statement describes a *relation* between a *concept* and a *property*. Usually, a **concept** K is a type or kind (e.g., bird) while a **property** P

---

[2] `https://github.com/emilyallaway/generics-exemplars`

Category	Generic (G)	INSTANTIATION	EXCEPTION
(a) quasi-def	"Stars produce radiation"	"The sun produces radiation"	"Stars produce light"
	$K(x) \land r(x,y) \implies P(y)$	$K(x) \land r(x,y) \land P(y)$	$K(x) \land r(x,y) \land \not\sim P(y)$
(b) principled	"Birds can fly"	"Owls can fly"	"Penguins can't fly"
	"Sharks attack swimmers"	"Threatened sharks attack swimmers"	"Sharks don't attack swimmers in the shallows"
	$K(x) \land P(y) \implies r(x,y)$	$K(x) \land r(x,y) \land P(y)$	$K(x) \land \neg r(x,y) \land P(y)$
(c) characterizing	"Cars have radios"	"2014 Prius model C has a radio "	"Cars have CD Players"
	L_G is ambiguous	$K(x) \land r(x,y) \land P(y)$	$K(x) \land r(x,y) \land \not\sim P(y)$
			"Newer cars don't have radios"
			$K(x) \land \neg r(x,y) \land P(y)$

Table 1: We define three categories of generics with their EXEMPLARS. The logical forms for the generic (L_G) and its EXEMPLARS are also below the examples. Using these categories we formulate templates for generating EXEMPLARS (see Table 2). K is the concept (blue), P the property (pink). See §3.3 for exoproperty $\not\sim P$.

is an ability (e.g., fly) or quality (e.g., feathered). Note that statements containing explicit quantification (e.g., "*Most* birds can fly") are generally *not* considered generics (Carlson, 1977; Krifka et al., 1995) and are therefore excluded from this study (see §A.1 for further discussion).

In our framework, we first categorize (§3.1) generics and derive their logical forms (§3.2). These logical forms for generics serve as our basis for formulating EXEMPLARS (§3.3) and designing templates suitable for generation (§3.4).

3.1 Generic Category Definitions

We categorize a generic based on the type of property it describes. In particular, by unifying theories from linguistics and philosophy[3], we split generics into three categories (see examples in Table 1). A generic has a particular category if:

(a) **Quasi-definitional**: the property is essential to a concept (Khemlani et al., 2009).

(b) **Principled**: the property has a strong association with the concept. This includes both properties with a principled association to a concept (e.g., flying is viewed as inherent to birds, although it is not essential in reality) (Prasada and Dillingham, 2006, 2009; Haward et al., 2018) and properties that are uncommon and often dangerous (Leslie, 2017).

(c) **Characterizing (char.)**: there is only a non-accidental relationship between the property and concept (e.g., based only on absolute or relative prevalence among concepts) (Leslie, 2007, 2008).

3.2 Logical Forms for Generics

We propose logical forms L_G that are used to represent an individual generic G. **Each generic category has a distinct logical form** (see Table 1). For quasi-definitional generics, since the property is defining we assert that the property is logically implied by the concept and relationship together (i.e., $K \land r \implies P$). In contrast, for principled generics we assert that the concept and property together logically imply the relationship (i.e., $K \land P \implies r$). Finally, for characterizing generics, the logical form depends on whether the generic is interpreted as quasi-definitional or principled. Logical forms with examples are shown in Table 1. Note that in a logical L_G, the concept K (property P) is satisfied by *both an individual or subtype* of K (P)[4].

3.3 Constructing EXEMPLARS

We now define generics EXEMPLARS, deriving them from the logical form of a generic.

INSTANTIATIONS For a generic, INSTANTIATIONS are contextually relevant **members of the concept with the desired property**. Formally,

Definition (INSTANTIATIONS)**.** *An* INSTANTIATION *satisfies* L'_G *(i.e.,* L_G *with implication replaced by conjunction*[5]*).*

For example, if we have

$$L_G\text{: }\textsc{Bird}(x) \land \textsc{Fly}(y) \implies \textit{can}(x,y)$$

$$\text{so, } L'_G\text{: }\textsc{Bird}(x) \land \textsc{Fly}(y) \land \textit{can}(x,y)$$

then (x, y) = ("owls", "fly") satisfies L'_G. Note that **INSTANTIATIONS all have the same logical form** regardless of the generic category (see Table 1).

---

[3] Description of the theories is provided in §A.2

[4] For example, $\textsc{Bird}(x)$ is true for x_1 ="my parrot", x_2 ="owls", and x_3 ="these birds" since all *are* birds.

[5] Replacing the implication with a conjunction excludes instances which satisfy L_G by satisfying *only* the right side of the implication (e.g., $\textsc{Bird}(x) \land \textsc{Fly}(y) \implies \textit{can}(x,y)$ is logically true for (x,y) = (airplanes, fly) but this is not a valid INSTANTIATION).

EXCEPTIONS An EXCEPTION counters one of two interpretations of the generic (see §A.3 for an in-depth discussion). EXCEPTIONS are members of the concept either **(i)** ***without* the generic property** (Greenberg, 2007) or **(ii)** ***with* an alternative property** (i.e., **exoproperty**) when the generic property is essential to the concept. For example, "penguins can't fly" (for the generic "birds can fly") counters the interpretation that "*all* birds can fly". In contrast, "stars produce light" (for the generic "stars produce radiation") counters the interpretation that "stars produce *only* radiation". Note that in the latter case, there are no specific stars that do not produce radiation. More formally,

Definition (EXCEPTIONS)**.** *An* EXCEPTION *satisfies the logical form* $\nsim L_G$ *(i.e.,* $\neg L_G$ *where* $\neg P$ *replaced by* $\nsim P$ *when applicable). We define* $\nsim \boldsymbol{P}$ ***as the exoproperty** of* P*: a property adjacent to* P *that is **not*** P *but **is** contextually relevant to* P.

For example, if the logical form L_G is

$$\textsc{Star}(x) \wedge \mathit{produce}(x, y) \implies \textsc{Radiation}$$

then $\nsim L_G$ is

$$\textsc{Star}(x) \wedge \mathit{produce}(x, y) \wedge \nsim\textsc{Radiation}(y)$$

and so (x, y) = ("stars", "light") satisfies $\nsim L_G$, while (x, y) = ("stars", "movies") does not. Notice that the latter pair is invalid because "movies" is not a relevant alternative to "radiation" since it is not informative about the generic.

Since a generic's EXCEPTIONS depend on its logical form L_G, they are also dependent on the generic's category (see Table 1). In particular, the EXCEPTIONS for quasi-defintional generics are individuals with alternative properties (since the property is viewed as essential) while for principled generics the EXCEPTIONS are individuals without the generic property. EXCEPTIONS for characterizing generics are, like the logical forms, dependent on the generic's interpretation (see §3.2)

3.4 Logical Forms to Templates

Based on our proposed formulae (Table 1) for EXEMPLARS we define seven templates for generation (Table 2). Each template expresses a set of instances that satisfy the logical form of an EXEMPLAR. Each template has two sets of content specifications: for the *input* and for the *completion* (i.e., the decoder output).

For **INSTANTIATIONS, we define three templates** with subtypes of the concept, property, or both. For **EXCEPTIONS we have four templates,** using subtypes of either the concept *or* property (but not both) in order to avoid irrelevant or uninformative instances. For example, for the generic "Birds can fly", the sentence "Penguins can't fly long distances" (which has subtypes of both the concept and property) is *uninformative* because it doesn't mean penguins can't fly *in general* (e.g., they might still be able to fly short distances).

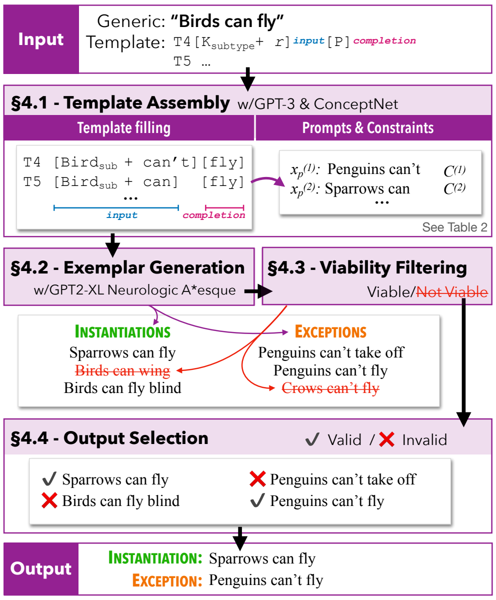


Figure 2: Overview of our method for an input generic.

4 Methodology

We propose a pipeline system to automatically generate generics EXEMPLARS. Our system takes as input a generic G and the templates derived from its category[6] (§3.4), and outputs a set of generated EXEMPLARS (Fig. 2). First, the system **assembles** and populates the templates according to the input generic (§4.1, Fig. 2). Then, the filled templates are converted into prompts and constraints that control the **generation** decoding process (§4.2). Finally, the output is **filtered** to remove non-viable (§4.3, Fig. 2) or irrelevant (§4.4, Fig. 2) EXEMPLARS.

4.1 Template Assembly

To populate our templates (defined in Table 2), we use a dependency parser[7] to identify the text spans of the concept, relation, and property in a generic. Then, **(i) we extract subtypes** for the concept and property and use these to construct the **(ii)** input i.e., generation prompts x_p) and **(iii)** completion (i.e., lexical constraints $\mathcal{C}$) specifications.

---

[6] We assume the generic's category is known.

[7] `https://spacy.io/`

Categories		Template		Example			
		input	*comp.*	Generic	Prompt x_p	Constraints $\mathcal{C}$	
EXCEP.	quasi-def & char	$\mathtt{K}$ + r	$\not\sim\mathtt{P}$	Stars produce radiation	"Stars *produce*"	$\neg$radiation $\wedge\neg$ x-rays $\wedge$ ...	`t1`
		$\mathtt{K}_{sub}$ + r	$\not\sim\mathtt{P}$		"Sun *produces*"	$\neg$radiation $\wedge\neg$ x-rays $\wedge$ ...	`t2`
	principled & char	$\mathtt{K}$ + $\neg r$	$\mathtt{P}_{sub}$	Birds can fly	"Birds *can't*"	migrate $\vee$ soar $\vee$ glide $\vee$...	`t3`
		$\mathtt{K}_{sub}$ + $\neg r$	$\mathtt{P}$		"Penguins *can't*"	fly $\vee$ flying $\vee$...	`t4`
INST.	all	$\mathtt{K}_{sub}$ + r	$\mathtt{P}$	Birds can fly	"Sparrows *can*"	fly $\vee$ flying $\vee$...	`t5`
		$\mathtt{K}$ + r	$\mathtt{P}_{sub}$		"Birds *can*"	migrate $\vee$ soar $\vee$ glide $\vee$...	`t6`
		$\mathtt{K}_{sub}$ + r	$\mathtt{P}_{sub}$		"Sparrows *can*"	migrate $\vee$ soar $\vee$ glide $\vee$...	`t7`

Table 2: Templates for generating EXEMPLARS, derived from their logical forms (§3.3). *sub* indicates a subtype, `K` the concept, `P` the property, $\not\sim$`P` its exoproperty (§3.2). *comp* is the completion.

(i) Subtype Extraction We extract subtypes using both ConceptNet (Speer et al., 2017)[8] and GPT-3 (Brown et al., 2020). GPT-3 increases the coverage and diversity of subtypes, since many natural and valid subtypes may be missing from ConceptNet (e.g., modifier phrases attached to a concept: "*young* Arctic fox"). We only use GPT-3 for subtypes of the concept, since by increasing the diversity in the prompt we may encourage diversity in the generated properties (see details Appendix D).

(ii) Input Specification We construct the input specifications by constructing generation prompts. Following the template, each prompt consists of either the concept (or a subtype) and the relationship (or its negation) (see Table 2). We prepend to each prompt the generic itself and a connective (e.g., "however"). We rank the prompts by perplexity and use the top k_p prompts for generation.

(iii) Completion Specification Following the templates, we constrain the generation output to describe the property (or a subtype) or its exoproperty (see Table 2). We construct a set of completion constraints (e.g., $\mathcal{C}^{(i)}$ in Fig. 2 specifies "fly" should be in the completion) using lexical items including subtypes, synonyms, and morphological forms.

4.2 Generation

In order to generate output that has a specific pragmatic relation to the input without requiring training, we use the NeuroLogic A*esque (NeuroLogic*) (Lu et al., 2022) decoding algorithm. NeuroLogic* is an unsupervised decoding algorithm that takes as input a prompt x_p and set of lexical constraints $\mathcal{C}$ and produces a completion of the prompt $\hat{y}$ which has high likelihood given the prompt *and* high satisfaction of the constraints (estimated throughout the decoding). A lexical constraint consists of a set of n-grams $w = (w_i^1, \ldots, w_i^m)$ and is satisfied when at least one $w_i \in w$ is in $\hat{y}$ (inclusion constraints) or is not in $\hat{y}$ (exclusion constraints).

By using the input prompts (as x_p) and completion constraints (as $\mathcal{C}$) derived from our templates (§4.1), we can control the output content, syntactic form, *and* pragmatic relevance. We note that since we cannot concretely define the set of relevant potential candidates for a property's exoproperty (§3.2), decoding constraints *must* be used to generate EXCEPTIONS.

Output Ranking We rank the outputs from NeuroLogic* by template and prompt and we take the top k_r outputs as potential EXEMPLARS. The outputs are ranked by perplexity (for fluency) and by the probability of a specific NLI label (for relevance) and we average the two ranks. For NLI labels, we hypothesize that a good EXCEPTION aligns with NLI's contradiction, as does a good INSTANTIATION with entailment (see Fig. 2). While this alignment is useful for ranking, the relationship between the EXEMPLARS and NLI labels is not this straightforward in reality, as we will discuss (§6.3).

4.3 Filtering For Viability

Since pre-trained language models have a tendency to hallucinate facts (Rohrbach et al., 2018) or produce non-specific output (e.g., "Birds can do things"), we apply a viability filter to the ranked output generations. Specifically, we train a discriminator to predict whether an output is viable (i.e., true *and* sufficiently specific that it could be an EXEMPLAR) or not, using human annotated examples (see Appendix B for details). Generations predicted not viable by the trained discriminator are removed from the dataset.

4.4 Output Selection

Our final task is to select the generations that are pragmatically relevant (i.e., **valid**; correctly follows a template) EXEMPLARS. To do this, we first collect gold labels from humans for whether an

---

[8] Relations: `IsA`, `InstanceOf`, `Synonym`

EXEMPLARS is valid. These annotations produce two sets of binary labels; one set each for INSTANTIATIONS and EXCEPTIONS.[9] Although this task is more complex than annotating for viability, removing non-viable generations helps reduce the complexity (i.e., we do not need to worry about false statements that adhere to the template). By annotating only viable generations we also reduce the required amount of annotation. Using the human annotations, we train two validity discriminators: one for EXCEPTIONS, one for INSTANTIATIONS. The trained validity discriminators are used to rank and select the best generations for each generic as our system output.

5 Experiment Details

We discuss our experimental setup and specify full hyperparameters in Appendix C.

5.1 Data Source

We use a subset of the generics dataset from Bhagavatula et al. (2022), a set of 30K generics built upon common everyday concepts (e.g., "hammers") and relations (e.g., "used for") sourced from resources such as GenericsKB (Bhakthavatsalam et al., 2020) and ConceptNet (Speer et al., 2017). The dataset includes a diverse variety of concepts, including general knowledge ("Dogs bark"), locative generics ("In a hotel, you will find a bed"), and comparative generics ("Cars are faster than people"). We use 653 generics from the test set, excluding human referents as the concept (e.g., nationalities, professions) due to social bias concerns.

5.2 Annotations

All annotations are done using Amazon Mechanical Turk with three annotators per HIT (paid at \$15/hour on average) and processed using MACE (Hovy et al., 2013) to filter annotators and determine the most likely label. We note that while all tasks achieve moderate inter-annotator agreement, the complex pragmatics of generics make these tasks difficult for human annotators.

For **generic type** (§3.1), we conduct two annotation passes to partition *all 653 generics* into the three groups in Table 1. The Fleiss' κ (Fleiss, 1971) is 0.41 and 0.58 for the first and second pass respectively. Our categorization results in 296 quasi-definitional, 125 principled, and 232 characterizing generics. For the **viability filter** (§4.3), we annotate a set of 7665 *system generations* from 150 generics. The Fleiss' κ is 0.53.

		Subtype Source		
		G3	CN	G3+CN
Generated	Output (§4.2)	42272	10496	52768
	Viable (§4.3)	22865	5452	28317
Valid (§4.4)	EXCEP.	4375	1922	6297
	INST.	10983	1579	12562
	TOTAL	**15358**	**3501**	**18859**

Table 3: Statistics of the generated dataset, with GPT-3 (G3) and ConcepNet (CN) subtypes used.

For **EXEMPLAR gold labels** (§4.4), we use separate annotation tasks for INSTANTIATIONS and EXCEPTIONS (see Appendix B for details) with Fleiss' κ of 0.40 and 0.45 respectively. For training each discriminator, we randomly sample and annotate $\sim 1k$ system generations from $\sim$300 generics. For human evaluation (§6.2), we annotate the *top* 5 discriminator-ranked generations for *all generics* from both our system and the baseline.

5.3 Discriminators

For all discriminators, we fine-tune RoBERTa (Liu et al., 2019). All labeled data is split such that all generations for a particular generic are in the same data partition.

5.4 Few-Shot Baseline

As a baseline for generation, we use GPT-3 (Brown et al., 2020) with few-shot prompting. Since we do not have access to the decoding algorithm for GPT-3, we cannot use decoding constraints to control the output (as in our system). Therefore, we use few-shot prompting in order to control the output of GPT-3. Specifically, for each template (Table 2) we construct a few-shot prompt (Appendix D) that consists of three examples that illustrate the desired template. This setup is very similar to the prompts to our system, except our system is not provided examples and GPT-3 is not provided with subtypes (when appropriate to the template). Note, our goal is *not* to produce the best possible generations from GPT-3 but rather to show that constrained generation from GPT-2 (i.e., NeuroLogic*) outperforms (and is cheaper and more computationally feasible) than a natural use of GPT-3.

6 Evaluation

Using our computational framework, **we generate** 18859 **EXEMPLARS for** 653 **generics** (Table 3).

---

[9] Since a generation that is not an INSTANTIATION *in not necessarily* an EXCEPTION (and vise versa), these cannot be directly combined into a single multi-class labeling task.

	Generic	INSTANTIATION	EXCEPTION
(a)	"Bleaches may be used to whiten the teeth."	"non-toxic bleaches can be used to remove discoloration" (`t7`)	"A bottle of liquid bleach should not be used to whiten the teeth" (`t4`)
(b)	"A chest pain has a physical cause."	"an angina pectoris has an underlying cause" (`t5`)	"a chest pain has an emotional or psychological origin" (`t1`)
(c)	"A gun are used for hunting."	"a shotgun is used for small game" (`t7`)	"semiautomatics can be used for target practice" (`t2`)

Table 4: Examples of generated INSTANTIATIONS and EXCEPTIONS. The template used in the prompt for generation is indicated in parentheses (see Table 2).

Example system generations are in Table 4.

To evaluate our approach, we first qualitatively investigate our system and outputs (§6.1) and then conduct a human evaluation (§6.2). We also conduct a detailed analysis of our system and the implications of our results.(§6.3). Our results show that our approach produces a large set of high quality generations for this difficult task. They also highlight current limitations in machine reasoning and potential directions for future work.

6.1 Qualitative Analysis

Observations We first observe that while close to half the output generations are untrue or not viable, the majority of viable generations are valid EXEMPLARS (Table 3). In addition, we see from system outputs (Table 4) that our system can successfully generate valid EXEMPLARS with subtypes of both the concept (e.g., "angina pectoris" vs. "a chest pain" in (b)) and the property (e.g., "small game" vs. "hunting" in (c)). Furthermore, it produces valid EXCEPTIONS with both the simpler relation-negation templates (i.e., templates `t3`/`t4`; see (a)) *and* with relevant exoproperties (i.e., templates `t1`/`t2`; see (b) and (c)). These highlight the success of our system in producing high-quality EXEMPLARS.

Discriminator Analysis On their respective annotated test sets, the accuracy of the viability discriminator (§4.3) is 75.2 and the accuracies of the trained validity discriminators are 77.4 for INSTANTIATIONS and 75.0 for EXCEPTIONS

In order to investigate the discriminator quality, we also conduct a manual analysis of the errors made by the *validity* discriminators. We observe that subtypes are particularly difficult for the discriminators to identify (e.g., that "freshwater lakes and rivers" are a type of "water"). Exoproperties can also be challenging for both the discriminators and humans (e.g., whether "able to land" is a subtype or alternative to "able to move").

We also observe that the discriminators identify a number of instances that were mislabeled by the human annotators. In particular, for 10 out of the 22 examples (5 out of the 14) where the EXCEPTION (INSTANTIATION) discriminator prediction disagrees with the human label, we judge the *discriminator* prediction to be correct. For example, "clocks are synchronized to the time zone" is labeled (incorrectly) by humans as an invalid EXCEPTION to the generic "clocks are synchronized to the second", despite "to the time zone" being a *relevant alternative* to "to the second". Counting the human-mislabeled instances as correct would increase the discriminator accuracies to 86% (85%) for the EXCEPTIONS (INSTANTIATIONS).

6.2 Human Evaluation

To quantitatively evaluate our system, we compute precision at k (for $k = 1$ and $k = 5$) using our human-annotated judgements (§5.2) (Table 5).

Our model outperforms the few-shot baseline (i.e., GPT-3) in all cases, and by a large gap (average 12.8 points). This is especially significant for EXCEPTIONS, which are more challenging to generate than INSTANTIATIONS, and where the baseline performance is close to random. Since generics are defaults, it follows that INSTANTIATIONS should be easier to produce than EXCEPTIONS. The fact that more generated INSTANTIATIONS are true (71% versus 40%) and more true INSTANTIATIONS are accepted by the discriminator (77% versus 50%), compared to the EXCEPTIONS, supports this intuition. Hence, the large improvements by our model over the baseline are significant towards generating these difficult EXCEPTIONS.

Additionally, we examine our model performance across templates. Specifically, we compute the fraction of generations for a template that annotators label as valid, using the same number[10] of generations for both models for a specific template

---

[10] The models produce similar numbers of generations on all templates except `t5`.

	EXCEPTIONS		INSTANTIATIONS	
	P@1	P@5	P@1	P@5
GPT-3	0.517	0.563	0.758	0.689
Ours	**0.632**	**0.616**	**0.911**	**0.882**

Table 5: Precision at k (P@k).

	EXCEPTIONS				INSTANTIATIONS		
	`t1`	`t2`	`t3`	`t4`	`t5`	`t6`	`t7`
#Gens	401	911	30	43	1147	4	862
GPT-3	0.65	0.53	**0.52**	**0.59**	0.78	0.75	0.50
Ours	**0.68**	**0.54**	0.30	0.47	**0.87**	**1.0**	**0.87**

Table 6: Precision by template. #Gens is per template and is the minimum of the models.

(Table 6). We see that not only does our model outperform the baseline for the majority of templates, these templates constitute the majority of the generations ('#Gens' in Table 6).

The performance comparison by template does not account for the fact that, while our system is constrained to follow the given template, with GPT-3 the template is only suggested by the prompt and so the model output may not adhere to it. As a result, the GPT-3 performance for certain templates (`t2-4`) is inflated because GPT-3 outputs simpler constructions that do not follow the requested template. Therefore, we conduct a manual analysis of the best 40 baseline (i.e., GPT-3) generations per template, ranked by perplexity. For EXCEPTIONS, the baseline produces on average only 2.5/40 generations that fit the desired templates for `t2`-`t4`. Additionally, for the one EXCEPTION template, `t1`, where most baseline generations fit the template (37/40), our model still outperforms the baseline. For INSTANTIATIONS, the baseline performs slightly better (average 10/40 fitting generations) but still poorly. From this we observe that not only is the baseline not controllable, our model outperforms the baseline in cases when it does adhere to output requirements.

6.3 Discussion

Does controllability matter? We ablate the decoding algorithm by removing the constraints (i.e., using beam search) (Table 7a). Although both systems condition their outputs on the same prompts, NeuroLogic*, with linguistic-theory-guided constraints, produces over seven times as many unique generations as unconstrained decoding (i.e., beam search). Additionally, the proportion of valid generations (i.e., accepted by our discriminators) is nearly twice as many for NeuroLogic*. This illustrates the importance of incorporating linguistic-theory-based control into decoding in order to generate a large set of unique, and valid, EXEMPLARS.

	Beam		**NeuroLogic***	
	#Gens	%Val	#Gens	%Val
EXCEP.	5083	13.4	29962	21.0
INST.	2221	39.6	22806	55.3
ALL	7307	21.4	52768	35.7

(a) Decoding method ablation: beam search vs. NeuroLogic *.

	MLM		**CN**		**G3**	
	#Gens	%Val	#Gens	%Val	#Gens	%Val
EXCEP.	10350	25.2	7521	25.5	22441	19.5
INST.	4459	50.7	2975	53.0	19831	55.4
ALL	14809	32.9	10496	33.3	42272	36.3

(b) Subtype ablation: MLM, ConceptNet (CN), GPT-3 (G3).

Table 7: Ablation results. #Gens: generations after ranking and filtering. %Val: percent accepted by the corresponding validity discriminator.

Do CKBs contain sufficiently rich information? We probe whether a CKB (i.e., ConceptNet) contains sufficiently rich type information to produce EXEMPLARS. Specifically, we vary the source of subtypes in the template-based prompts and constraints for our system, comparing ConceptNet (CN) to extracting commonsense knowledge from language models (i.e., from GPT-3 prompting (G3) and GPT-2 masked-language model (MLM) (Devlin et al., 2018; Taylor, 1953) infilling).

In fact, CN subtypes result in the fewest generations (Table 7b). In contrast, using GPT-3 for subtypes produces the most generations. Although using MLM for subtypes produces fewer generations than using GPT-3, the proportion of valid generations is comparable and hence MLM could be used as a substitute if using GPT-3 is not feasible. This shows that while CKBs such as ConceptNet are a good source of generics, producing EXEMPLARS requires knowledge that may not always be encoded within the CKB. Therefore, generating EXEMPLARS is important for accessing relevant knowledge beyond what is in CKBs and enabling tools that can effectively use CKBs in reasoning.

Does NLI impact EXEMPLARS? Since generics EXEMPLARS are closely related to default inheritance (nonmonotonic) reasoning, NLI is a natural task for investigating machine reasoning about EXEMPLARS. Thus, we examine whether controlling the NLI relation between generics and EXEMPLARS improves precision. Specifically, we compute the

	EXCEP.		INST.	
	$P@1$	$P@5$	$P@1$	$P@5$
Ours	0.632	0.616	0.911	0.882
+ NLI-neu	0.569	0.563	0.906	**0.891**
+ NLI-sim	**0.839**	**0.790**	0.864	0.862
+ NLI-neu-sim	0.638	0.618	**0.913**	**0.891**

Table 8: Precision at k with NLI label filtering. NLI-sim is contradiction for EXCEP., entailment for INST.

NLI label between the generic (premise) and EXEMPLAR (hypothesis) and exclude generations that do not have a specific predicted NLI label: 'contradiction' for EXCEPTIONS and 'entailment' for INSTANTIATIONS (NLI-sim), 'neutral' (NLI-neu), or NLI-sim and 'neutral' (NLI-neu-sim). We find that by controlling the NLI relation, we improve precision for EXCEPTIONS by 20.7 points (Table 8). However, for INSTANTIATIONS NLI label filtering has a negligible impact on precision. Therefore, we observe that controlling NLI relations can improve EXCEPTION quality but is less beneficial for INSTANTIATIONS. Additionally, note that the alignment with NLI labels is not actually as straightforward as observed, which we discuss next.

Can NLI sufficiently represent EXEMPLARS? Although we observe an alignment between *predicted* NLI labels and EXEMPLARS, this actually indicates systematic NLI-model errors, deriving from the insufficiency of NLI schema for capturing the nuances of generics EXEMPLARS.

Consider the sentences in Fig. 3, relating to the generic "Birds can fly". We see that *only false* statements (i.e., not EXCEPTIONS) are "unlikely to be true given the information in the premise [generic]" (Dagan et al., 2013) (i.e., NLI contradictions). Since the lack of explicit quantification in generics does not preclude the existence of exceptions, EXCEPTIONS *should actually be labeled neutral* by NLI. With INSTANTIATIONS, we observe that the NLI relationship may be *either* neutral or entailment. These theorized alignments, coupled with our prior observations about EXEMPLARS and predicted NLI labels, highlight the challenges of reasoning about EXEMPLARS with NLI.

The examples in Fig. 3 also highlight that the NLI neutral label does not distinguish between statements that are true but not entailed or contradictory (e.g., "Penguins cannot fly") and statements with unknown truth value (e.g., "Tweety bird can fly"). Our generics EXEMPLARS emphasize the need for a more fine-grained notion of NLI.

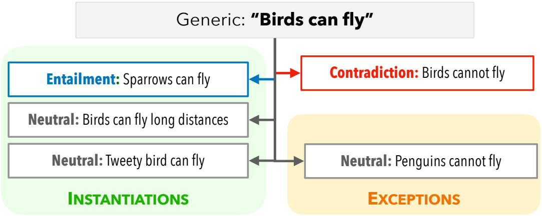


Figure 3: EXEMPLARS and correct NLI labels.

7 Conclusion

In this work, we draw on insights from linguistics to propose a novel computational framework to automatically generate valid EXEMPLARS for generics, as a step towards capturing the nuances of human reasoning for generics. Our system generates $\sim 19k$ EXEMPLARS for 653 generics and outperforms GPT-3 at generating viable examples, while remaining more controllable. We also demonstrate the limitations of CKBs and the importance of explicit linguistic modeling in generating EXEMPLARS. That is, the importance of linguistic-theory-based decoding and semantics-based filtering with NLI. Finally, we highlight the inability of current NLI models to reason about and represent the default-inheritance-reasoning relationship between generics and EXEMPLARS.

Acknowledgements

We would like to thank Sarah-Jane Leslie and Maarten Sap for their helpful discussions, the Beaker Team at AI2 for the compute infrastructure, and the anonymous reviewers for their suggestions. This work is supported in part by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1644869 and by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the HIATUS Program contract 2022-22072200005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, the NSF or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

Limitations and Risks

The generics we source (see §5.1) are exclusively in English. Therefore, our approach may not be suited to all possible generics in all languages. In particular, our system does not handle generics where valid INSTANTIATIONS include negating (§3.2) the concept. This is due to the restriction that most English generation is left-to-right and it is not possible to define a closed set of possible concept negations for the prompt.

In this work, we do not generate EXEMPLARS for generics involving human referents (e.g., professions, nationalities). We exclude generics involving human referents to mitigate the risk of generating socially biased EXEMPLARS or harmful stereotypes (e.g., "Black folks go to jail for crimes" for the generic "People go to jail for crimes"). Additionally, handling of human stereotypes require methods that are beyond the scope of this paper. For example, a socially-aware EXCEPTIONS to a generic like "Girls wear dresses" would be "Boys wear dresses, too". This would require the understanding of the possible subtext of such a statement (e.g. "Only girls wear dresses"), which is beyond the current capabilities of this study and worthy of future exploration.

Finally, we note that while it is not the intended purpose of our system, a malicious user could still use our system to generate EXEMPLARS for a generic involving a person and propagate potentially harmful social biases.

References

Nicholas Asher and Michael Morreau. 1995. What some generic sentences mean. In Gregory N. Carlson and Francis Jeffry Pelletier, editors, *The generic book*, pages 300–338. The University of Chicago Press.

Chandra Bhagavatula, Jena D. Hwang, Doug Downey, Ronan Le Bras, Ximing Lu, Keisuke Sakaguchi, Swabha Swayamdipta, Peter West, and Yejin Choi. 2022. I2d2: Inductive knowledge distillation with neurologic and self-imitation. *ArXiv*, abs/2212.09246.

Sumithra Bhakthavatsalam, Chloe Anastasiades, and Peter Clark. 2020. Genericskb: A knowledge base of generic statements. *ArXiv*, abs/2005.00660.

Michael Boratko, Xiang Lorraine Li, Rajarshi Das, Timothy J. O'Gorman, Daniel Le, and Andrew McCallum. 2020. Protoqa: A question answering dataset for prototypical common-sense reasoning. *ArXiv*, abs/2005.00771.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Gerhard Brewka. 1987. The logic of inheritance in frame systems. In *IJCAI*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Greg N. Carlson. 1977. Reference to kinds in english. In *Ph.D. dissertation, University of Massachusetts, Amherst*.

Greg N. Carlson. 1989. On the semantic composition of english generic sentences. In Gennaro Chierchia, Barbara H Partee, and Raymond Turner, editors, *Properties, Types and Meaning, Vol. II. Semantic Issues*. Dordrecht: Kluwer.

Ariel Cohen. 1996. *Think generic! The meaning and use of generic sentences*. Carnegie Mellon University.

Ariel Cohen. 1999. Generics, frequency adverbs, and probability. *Linguistics and philosophy*, 22(3):221–253.

Ariel Cohen. 2004. Generics and mental representations. *Linguistics and Philosophy*, 27(5):529–556.

Robin Cooper, Richard Crouch, Jan Van Eijck, Chris Fox, Josef Van Genabith, Jan Jaspers, Hans Kamp, Manfred Pinkal, Massimo Poesio, Stephen Pulman, et al. 1994. Fracas: A framework for computational semantics. *Deliverable D6*.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220.

Barry Devereux, Lorraine K. Tyler, Jeroen Geertzen, and Billi Randall. 2014. The centre for speech, language and the brain (cslb) concept property norms. *Behavior Research Methods*, 46:1119 – 1127.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Renée Elio and Francis Jeffry Pelletier. 1996. On reasoning with default rules and exceptions. In *Proceedings of the 18th conference of the Cognitive Science Society*, pages 131–136.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *EMNLP*.

Annemarie Friedrich, Alexis Palmer, and Manfred Pinkal. 2016. Situation entity types: automatic classification of clause-level aspect. In *Annual Meeting of the Association for Computational Linguistics*.

Annemarie Friedrich, Alexis Palmer, Melissa Peate Sørensen, and Manfred Pinkal. 2015. Annotating genericity: a survey, a scheme, and a corpus. In *LAW@NAACL-HLT*.

M. Ginsberg. 1987a. Introduction. Morgan Kaufmann, Los Altos, CA.

Matthew L. Ginsberg. 1987b. Readings in nonmonotonic reasoning. In *AAAI*.

Yael Greenberg. 2007. Exceptions to generics: Where vagueness, context dependence and modality interact. *Journal of Semantics*, 24(2):131–167.

Steve Hanks and Drew McDermott. 1986. Default reasoning, nonmonotonic logics, and the frame problem. In *AAAI*.

Paul Haward, Laura Wagner, Susan Carey, and Sandeep Prasada. 2018. The development of principled connections and kind representations. *Cognition*, 176:255–268.

John F. Horty and Richmond H. Thomason. 1988. Mixing strict and defeasible inheritance. In *AAAI*.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with mace. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. In *AAAI*.

Tomasz Imielinski. 1985. Results on translating defaults to circumscription. In *IJCAI*.

Nirit Kadmon and Fred Landman. 1993. Any. *Linguistics and philosophy*, 16(4):353–422.

Sangeet Khemlani, Sarah-Jane Leslie, and Sam Glucksberg. 2009. Generics, prevalence, and default inferences. *Proceedings of the 31st annual cognitive science society*, pages 443–448.

Arnold Kochari, Robert Van Rooij, and Katrin Schulz. 2020. Generics and alternatives. *Frontiers in Psychology*, 11:1274.

Manfred Krifka. 1987. An outline of genericity. Seminar für natürlich-sprachliche Systeme der Universität Tübingen.

Manfred Krifka, Francis Jeffry Pelletier, Gregory N. Carlson, Alice Ter Meulen, Gennaro Chierchia, and Godehard Link. 1995. Genericity: an introduction. In Gregory N. Carlson and Francis Jeffry Pelletier, editors, *The Generic book*, pages 1–124. The University of Chicago Press.

Manfred Krifka et al. 2012. Definitional generics. *Genericity*, pages 372–389.

Dimitra Lazaridou-Chatzigoga and Linnaea Stockall. 2013. Genericity, exceptions and domain restriction: experimental evidence from comparison with universals. In *Proceedings of Sinn und Bedeutung*, volume 17, pages 325–343.

Sarah-Jane Leslie. 2007. Generics and the structure of the mind. *Philosophical perspectives*, 21:375–403.

Sarah-Jane Leslie. 2008. Generics: Cognition and acquisition. *Philosophical Review*, 117(1):1–47.

Sarah-Jane Leslie. 2017. The original sin of cognition: Fear, prejudice, and generalization. *The Journal of Philosophy*, 114(8):393–421.

David Lewis and Edward L. Keenan. 1975. *Adverbs of quantification*, page 3–15. Cambridge University Press.

Vladimir Lifschitz. 1989. Benchmark problems for nonmonotonic reasoning. In *Proceedings of the Second international Workshop on Non-monotonic Reasoning*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. 2022. Neurologic a* esque decoding: Constrained text generation with lookahead heuristics. In *NAACL*.

Alda Mari, Claire Beyssade, and Fabio Del Prete. 2012. *Genericity*, volume 43. OUP Oxford.

Ken McRae, George S. Cree, Mark S. Seidenberg, and Chris McNorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37:547–559.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Kanishka Misra, Allyson Ettinger, and Julia Taylor Rayz. 2021. Do language models learn typicality judgments from text? *ArXiv*, abs/2105.02987.

Daniel N. Osherson, Edward E. Smith, Ormond Wilkie, Alejandro López, and Eldar Shafir. 1990. Category-based induction. *Psychological Review*, 97:185–200.

James P. Van Overschelde, Katherine A. Rawson, and John Dunlosky. 2004. Category norms: An updated and expanded version of the battig and montague (1969) norms. *Journal of Memory and Language*, 50:289–335.

Francis Jeffry Pelletier. 2009. Are all generics created equal? *Kinds, Things, and Stuff: Mass Terms and Generics*, pages 60–79.

Francis Jeffry Pelletier and Renée Elio. 2005. The case for psychologism in default and inheritance reasoning. *Synthese*, 146(1):7–35.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *EMNLP*.

David L. Poole. 1988. A logical framework for default reasoning. *Artificial Intelligence*, 36:27–47.

Sandeep Prasada and Elaine M Dillingham. 2006. Principled and statistical connections in common sense conception. *Cognition*, 99(1):73–112.

Sandeep Prasada and Elaine M Dillingham. 2009. Representation of principled connections: A window onto the formal aspect of common sense conception. *Cognitive Science*, 33(3):401–448.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

R. Reiter. 1978a. On reasoning by default. In *Proceedings of TINLAP-2*, pages 210–218, University of Illinois. Association of Computational Linguistics.

Raymond Reiter. 1978b. *On Closed World Data Bases*, pages 55–76. Springer US, Boston, MA.

Raymond Reiter. 1980. A logic for default reasoning. *Artificial Intelligence*, 13:81–132.

Lance J. Rips. 1975. Inductive judgments about natural categories. *Journal of Verbal Learning and Verbal Behavior*, 14:665–681.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. 2018. Object hallucination in image captioning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics.

Rachel Rudinger, Vered Shwartz, Jena D Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A Smith, and Yejin Choi. 2020. Thinking like a skeptic: Defeasible inference in natural language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4661–4675.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *AAAI*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Wilson L Taylor. 1953. “cloze procedure”: A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *"Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019a. HELP: A Dataset for Identifying Shortcomings of Neural Models in Monotonicity Reasoning. *ArXiv*, abs/1904.12166.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019b. Can Neural Networks Understand Monotonicity Reasoning? In *BlackboxNLP@ACL*.

A Generics

A.1 Generics and Quantifiers

Explicit quantification (e.g., "*Most* birds can fly", "Birds can *usually* fly") are excluded from this study because the quantifier implicitly accounts for all potential exceptions. That is, by saying "*Most* birds can fly" we implicitly indicate that a minority of the birds do not. This being the case, exceptions cannot be generated with statements with explicit quantification.

A.2 Generics Definitions

Categories of Generics We condense the five generic types proposed by Leslie (2007, 2008) and Khemlani et al. (2009) into our three categories (§3.1). The five types are:

- **Quasi-definitional**: generics concerning properties that are assumed to be universal among a concept. This is the same as our quasi-definitional category, see (a) Table 1. The property is considered a defining characteristic of the concept.
- **L-Principled**: generics concerning properties that are prevalent among a concept and are viewed as inherent, or connected in a principled way (Prasada and Dillingham, 2006, 2009; Haward et al., 2018). These generics are called principled in Leslie (2007, 2008). Note, these generics make up only one half of our "principled" category (§3.1). See first example for category (b) in Table 1; the second example there does *not* fit Leslie (2007, 2008)'s definition of principled (i.e., L-principled).
- **Striking**: generics describing properties that are uncommon and often dangerous, and members of the concept are *disposed* to possess them if given the chance (Leslie, 2017). For example, the striking generic "Sharks attack swimmers" assumes all sharks are capable of attacking swimmers. These generics constitute the second half of our "principled" category. See second example (not first) for category (b) in Table 1.
- **Majority characteristic**: generics concerning properties that are neither deeply connected to the concept nor striking but occur in the majority of members of the concept. These constitute one half of our "characterizing category". See example for (c) in Table 1.
- **Minority characteristic**: generics concerning properties that are neither deeply connected to the concept nor striking but occur in the minority of members of the concept. For example, "Lions have manes", since only adult male lions (the minority of the lion population) have manes. These constitute the second half of our "characterizing category".

Both L-principled and striking generics are true in-virtue-of a secondary factor and therefore we group these into one category (i.e., "principled"; see §3.1). For L-principled generics, this may be a factor that causes the property to occur in the concept (e.g., Birds can fly because they have wings). For striking generics, it is the assumed predisposition of the kind to possess the property if given the chance.

For quasi-definitional generics, because the property is considered defining to concept, there is no implied secondary factor in-virtue-of which the generic is true. Therefore, these generics are descriptive and we put them in a separate category from striking and L-principled generics.

Finally, majority and minority characteristic generics are ambiguous in their interpretation. For example, "Lions have manes" can be interpreted as being true in-virtue-of some secondary factor (e.g., as a signal of fitness) or as being a merely accidental relationship. If the interpretation is the former, then lions without manes are valid EXCEPTIONS (e.g., lion cubs, female lions), while if the interpretation is the latter then then other attributes of lions are valid EXCEPTIONS (e.g., claws, fur).

Focuses of Generics We note that a generic can focus on the presence of the property within the concept (e.g., "Birds can fly" is concerned with which birds can fly) **or** can focus on the presence of the concept within holders of the property (e.g., "Triangles have three sides" is more concerned with what concepts have three sides). We will say that the former kind of generic is *concept-oriented* and the latter is *property-oriented*. A generic can be both concept and property oriented if it is ambiguous between the two readings (e.g., "Aspirin relieves headaches").

In this work, we have discussed and used definitions only for concept-oriented generics.

A.3 EXEMPLARS Definitions

Interpretations As discussed (§3.3), EXCEPTIONS counter an interpretation of the generic. Importantly, these interpretations must contain *universal quantification* of either the concept or the property. In this way, the implied universality of the generic can be countered by an EXCEPTION.

We consider the four interpretations of a generic derived from attaching the universal quantifiers "all" (or "always") and "only" to either the concept or the property. For example, for the generic "Birds can fly", we have:

1. ***All* birds can fly.**
   ⇒ Concept-oriented:
   which birds (i.e., all) can fly.
2. ***Only* birds can fly.**
   ⇒Property-oriented:
   for being able to fly, how defining (i.e., entirely) are birds.
3. **Birds can fly *all ways*.**
   ⇒Property-oriented:
   for flight, which types (i.e., all of them) can birds do.
4. **Birds can *only* fly.**
   ⇒Concept-oriented:
   of the things birds can do, how defining (i.e., entirely) is flying.

For concept-oriented generics, interpretations 1 and 4 are salient. for property-oriented generics, interpretations 2 and 3 are salient.

The quantifier "only" specifies how *defining* the concept is for the property (for concept-oriented generics); for property-oriented generics it specifies how the concept is to the property. Hence, quasi-definitional generics (§3.1 correspond to the interpretations containing "only" quantifiers. Therefore, their EXCEPTIONS will counter the implicit assumption that the property is defining for the concept (or vise versa for property-oriented generics). That is, the EXCEPTIONS will be members of the concept with *other relevant* properties. For example, for the generic "Stars produce radiation", an exception is "The sun produces light".

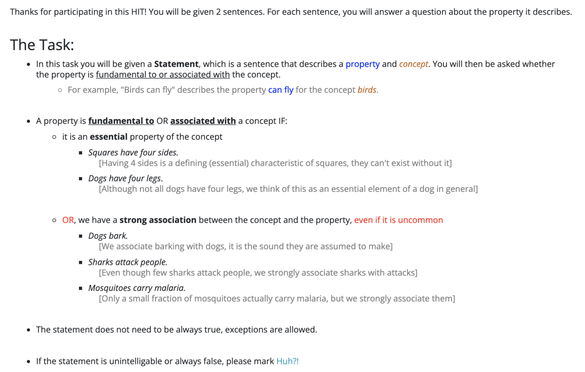
On the other hand, the quantifier "all" specifies the prevalence of the property among the concept (for concept-oriented generics). For property-oriented generics, "all" specifies the proportion of the property connected with the concept (e.g., how much of the property can members of the concept do). Hence, the "all" quantifier corresponds to principled generics (§3.1). Note, that even though striking generics (see §A.2) describe very low real-world prevalence, the *implication* is that prevalence is much higher, since individuals are disposed to possess the property (Leslie, 2017). Therefore, EXCEPTIONS to principled generics will be members of the concept (or types of the property) that do *not* possess the desired property (or are not present among the concept). For example, a bird that cannot fly (e.g., a penguin) or a type of movement humans cannot do (e.g., fly, for the generic "humans can move").

Logical Forms Although we only derived logical forms for concept-oriented generics in this work, similar definitions and logical forms can be derived for property-oriented generics. In particular, only the logical forms for quasi-definitional generics and their EXCEPTIONS will change if the generic is property-oriented. That is, the K and P in both logicals form for (a) in Table 1) can swapped to obtain the property-oriented versions. In this work, we do not deal with property-oriented generics and their EXEMPLARS due to the limitations of English generation (i.e., it is left-to-right).

B Annotation

For all annotation tasks, three annotators are used per HIT. When filtering annotators using MACE, we remove annotators with competence below 0.5 (or the median, if lower).

Generic Type Instructions for annotating generic types (§3.1) are shown Figure 4 (for the first pass) and Figure 5 (for the second pass). The first pass categorizes generics as either characterizing or not (either quasi-definitional or principled). The second pass categorizing non-characterizing generics as either quasi-definitional or principled.

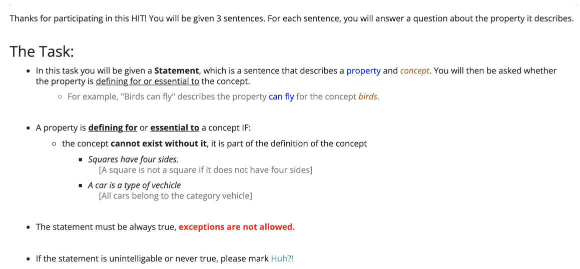


Thanks for participating in this HIT! You will be given 2 sentences. For each sentence, you will answer a question about the property it describes.

The Task:

- In this task you will be given a **Statement**, which is a sentence that describes a property and *concept*. You will then be asked whether the property is fundamental to or associated with the concept.
  - For example, "Birds can fly" describes the property can fly for the concept *birds*.
- A property is **fundamental to** OR **associated with** a concept IF:
  - it is an **essential** property of the concept
    - *Squares have four sides.*
      [Having 4 sides is a defining (essential) characteristic of squares, they can't exist without it]
    - *Dogs have four legs*.
      [Although not all dogs have four legs, we think of this as an essential element of a dog in general]
  - OR, we have a **strong association** between the concept and the property, even if it is uncommon
    - *Dogs bark.*
      [We associate barking with dogs, it is the sound they are assumed to make]
    - *Sharks attack people.*
      [Even though few sharks attack people, we strongly associate sharks with attacks]
    - *Mosquitoes carry malaria.*
      [Only a small fraction of mosquitoes actually carry malaria, but we strongly associate them]
- The statement does not need to be always true, exceptions are allowed.
- If the statement is unintelligable or always false, please mark Huh?!

Figure 4: Task instructions for first part of the generic type categorization annotation (§5.2).



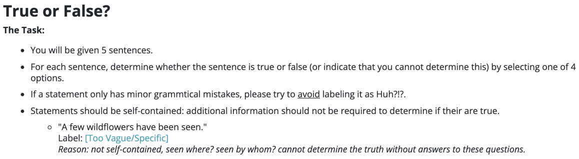
Thanks for participating in this HIT! You will be given 3 sentences. For each sentence, you will answer a question about the property it describes.

The Task:

- In this task you will be given a **Statement**, which is a sentence that describes a property and *concept*. You will then be asked whether the property is defining for or essential to the concept.
    - For example, "Birds can fly" describes the property can fly for the concept *birds*.
- A property is **defining for** or **essential to** a concept IF:
    - the concept **cannot exist without it**, it is part of the definition of the concept
        - *Squares have four sides.*
          [A square is not a square if it does not have four sides]
        - *A car is a type of vechicle*
          [All cars belong to the category vehicle]
- The statement must be always true, **exceptions are not allowed.**
- If the statement is unintelligable or never true, please mark Huh?!

Figure 5: Task instructions for second part of the generic type categorization annotation (§5.2).

Viability Task Instructions for annotating output generations for viability (§4.3) are shown in Figure 6.



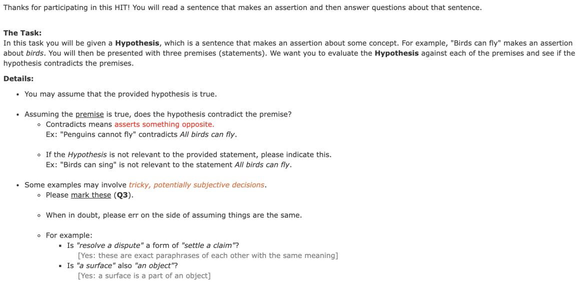
True or False?

The Task:

- You will be given 5 sentences.
- For each sentence, determine whether the sentence is true or false (or indicate that you cannot determine this) by selecting one of 4 options.
- If a statement only has minor grammtical mistakes, please try to avoid labeling it as Huh?!?.
- Statements should be self-contained: additional information should not be required to determine if their are true.
    - "A few wildflowers have been seen."
      Label: [Too Vague/Specific]
      Reason: not self-contained, seen where? seen by whom? cannot determine the truth without answers to these questions.

Figure 6: Task instructions for annotating truthfulness (§5.2).

EXEMPLARS Gold Labels For the INSTANTIATION template generations, annotators are asked whether the generation contradicts the original generic. Instructions are shown in Figure 8. However, for the exception template generations, an EXCEPTION is not a contradiction of the generic itself but of an associated logical form. For example, "Penguins cannot fly" does not actually contradict the generic itself ("Birds can fly") but a modified form of the generic involving quantification (i.e., "All birds can fly"). Therefore, we ask annotators whether the generation contradicts two modified forms of the generic. Instructions are shown in Figure 7.



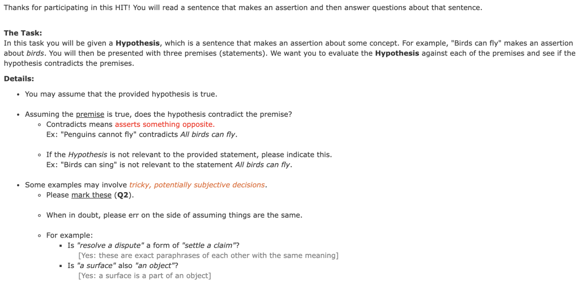
Thanks for participating in this HIT! You will read a sentence that makes an assertion and then answer questions about that sentence.

The Task:
In this task you will be given a **Hypothesis**, which is a sentence that makes an assertion about some concept. For example, "Birds can fly" makes an assertion about *birds*. You will then be presented with three premises (statements). We want you to evaluate the **Hypothesis** against each of the premises and see if the hypothesis contradicts the premises.

Details:

- You may assume that the provided hypothesis is true.
- Assuming the premise is true, does the hypothesis contradict the premise?
    - Contradicts means asserts something opposite.
      Ex: "Penguins cannot fly" contradicts *All birds can fly*.
    - If the *Hypothesis* is not relevant to the provided statement, please indicate this.
      Ex: "Birds can sing" is not relevant to the statement *All birds can fly*.
- Some examples may involve *tricky, potentially subjective decisions*.
    - Please mark these (**Q3**).
    - When in doubt, please err on the side of assuming things are the same.
    - For example:
        - Is *"resolve a dispute"* a form of *"settle a claim"*?
          [Yes: these are exact paraphrases of each other with the same meaning]
        - Is *"a surface"* also *"an object"*?
          [Yes: a surface is a part of an object]

Figure 7: Task instructions for annotating validity of EXCEPTIONS (§5.2).



Thanks for participating in this HIT! You will read a sentence that makes an assertion and then answer questions about that sentence.

The Task:
In this task you will be given a **Hypothesis**, which is a sentence that makes an assertion about some concept. For example, "Birds can fly" makes an assertion about *birds*. You will then be presented with three premises (statements). We want you to evaluate the **Hypothesis** against each of the premises and see if the hypothesis contradicts the premises.

Details:

- You may assume that the provided hypothesis is true.
- Assuming the premise is true, does the hypothesis contradict the premise?
    - Contradicts means asserts something opposite.
      Ex: "Penguins cannot fly" contradicts *All birds can fly*.
    - If the *Hypothesis* is not relevant to the provided statement, please indicate this.
      Ex: "Birds can sing" is not relevant to the statement *All birds can fly*.
- Some examples may involve *tricky, potentially subjective decisions*.
    - Please mark these (**Q2**).
    - When in doubt, please err on the side of assuming things are the same.
    - For example:
        - Is *"resolve a dispute"* a form of *"settle a claim"*?
          [Yes: these are exact paraphrases of each other with the same meaning]
        - Is *"a surface"* also *"an object"*?
          [Yes: a surface is a part of an object]

Figure 8: Task instructions for annotating validity of insts (§5.2).

We obtain modified forms of the generic by first converting the logical forms in Table 1 into a natural language templates by adding a universal quantifier. Then we apply the template to the generic itself. Specifically, from $K(x) \wedge r(x,y) \implies P(y)$ (e.g., for quasi-definitional generics) we derive "`[K] [REL] ONLY [P]`". For example, "mosquitoes drink *only* blood", which is contradicted by mosquitoes that drink something other than blood. Notice, that exceptions from templates 1 and 2 will contradict these statements. Similarly, for $K(x) \wedge P(y) \implies r(x,y)$ we derive "`ALL [K] [REL] [P]`". For example, "*All* birds can fly", which is contradicted by birds that cannot fly. Exceptions from templates 3 and 4 will contradict these statements.

C Implementation Details

C.1 Data

We use the generics data from Bhagavatula et al. (2022). For this study, we source from the subset of the test set found to be valid by the discriminator with probability at least 0.5 (768 generics). Of these, we exclude all mentions of human referents (e.g., kinship labels, nationalities, titles, professions) and actions (e.g., studying for a test) to arrive at a dataset of 653 generics. We remove human referents using a seed set of human referent terms compiled based on WordNet (Miller, 1995) and will be provided with the system code. We remove mentions of actions by excluding generics beginning with "In order to". The dataset is licensed under CC-BY and our usage aligns with the intended use of the data.

Preprocessing We remove adverbs of quantification (i.e., usually, typically, generally) from the generics and exclude generics with verbs of consideration (i.e., consider, posit, suppose, suspect, think). We also convert hedging statements to more explicit forms (e.g., "may have to be" to "must be").

	Train	Dev	Test	All
True	2831	412	433	3676
False/Non-salient	3180	367	442	3989
Total	6011	779	875	7665

Table 9: Data split statistics for truthfulness discriminator (§4.3).

		Train	Dev	Test	All
EXCEPTION	Valid	342	35	35	412
	Invalid	462	72	53	587
	Total	804	107	88	999
INSTANTIATION	Valid	374	38	29	441
	Invalid	466	38	33	537
	Total	840	76	62	978

Table 10: Data split statistics for validity discriminators (§4.4).

Parameter	**Values**
Random seed	29725
Batch size	[64, 32, 16]
Learning rate	[3e-5, 1e-5, 3e-6]
Number of epochs	[1, 3, 5]

Table 11: Hyperparameter bounds for the viability discriminator.

Parameter	**Values**
Random seed	4427
Batch size	[64, 32, 16]
Learning rate	[1e-4, 3e-5, 1e-5, 3e-6, 1e-6, 3e-7]
Number of epochs	[1, 3]

Table 12: Hyperparameter bounds for the validity discriminators.

Partitions The data splits for training the viability discriminator and validity discriminators are shown in Table 9 and Table 10 respectively.

C.2 Tools

For extracting components of the generic data we use spacy[11] for dependency parsing. We use *inflect*[12] to obtain plural and singular word forms and *mlconjug3*[13] to conjugate verbs. We use *nltk*[14] for additional synonyms.

C.3 Hyperparameters

To obtain subtypes from GPT-3 we use the *davinci* model and top-p sampling with $p = 0.9$, temperature 0.8 and maximum length 100 tokens. We use the top 5 sequences to obtain subtypes. For NLI scores, we use RoBERTa fine-tuned on MNLI (Williams et al., 2018) available from AllenNLP[15]. For the GPT-3 baseline we use the davinci model and top-p sampling 1.0, temperature 0.8, maximum length 50 tokens and top 5 sequences. Prompts for GPT-3 are given in Appendix D. GPT2-XL has 1.5 billion parameters, GPT-3 has 175 billion parameters. Our experiments are done using Quadro RTX 8000 GPUs.

For generation with NeuroLogic*, we use GPT2-XL (Radford et al., 2019) with a maximum length of 50 tokens and a beam size of 10 with temperature 10000000. We set the constraint satisfaction tolerance to 3. This means that at each step, only candidates whose number of satisfied constraints is within three of the maximum so far are kept. The 'look ahead' is also set to 3; look ahead three generation steps during decoding to estimate future constraint satisfaction. During prompt construction, take the top $k_p = 10$ prompts. If the generic produced less than 10 prompts total, we take half so that low quality prompts are not used even if few are produced. After ranking the output, we keep the top $k_r = 10$ generations for a template, keeping at most 2 per prompt.

For the viability discriminator, we fine-tune the model for 5 epochs using a batch size of 16 and learning rate $1e - 5$ and random seed 29725, selected by manual grid search with 27 trials (see bounds Table 11).

For the validity discriminators, we fine-tune *the viability discriminator* for 3 epochs with a batch size of 16 and learning rate $3e{-}5$. The instantiation discriminator uses a random seed of 4427 and the exception discriminator 4457. Hyperparameters are again selected by manual grid search with 36 trials (see bounds Table 12).

C.4 Validation Performance

We show the validation performance for the trained discriminators in Table 13.

D GPT-3 Prompts

D.1 Subtyping

To obtain subtypes from GPT-3, we first categorize the kinds into six categories: person, animal, other living (e.g., plants), location, temporal (e.g., Thursday), and other (e.g., candle, soup) (Table 14). For

---

[11] `https://spacy.io/`

[12] `https://pypi.org/project/inflect/`

[13] `https://pypi.org/project/mlconjug3/`

[14] `https://www.nltk.org/`

[15] `https://demo.allennlp.org/textual-entailment/roberta-mnli`

Discriminator	**Val**			**Test**		
	Max	Mean	Var	Max	Mean	Var
Viability	0.771	0.685	4.3e-3	0.752	0.673	3.6e-3
Validity EXCEPTIONS	0.757	0.600	8.8e-3	0.750	0.557	4.4e-3
Validity INSTANTIATIONS	0.763	0.582	6.6e-3	0.774	0.577	7.1e-3

Table 13: Trained discriminator accuracy (with mean and variance) on the validation and test sets.

each category, we construct a separate prompt for GPT-3 containing one type and five example subtypes. Then, for each kind we use the prompt from its assigned category to obtain subtypes. Note that we exclude all generics where the kind is “person”. This is to avoid producing or repeating stereotypes.

To determine the category, we use seed lists, for person, animal, other living, and locative, or the presence of prepositional beginnings (“On”, “In”, “At”, “During”), for locative and temporal. The “other” category encompasses all kinds that do not fit into another category.

D.2 Few-shot Baseline

The prompts for our few-shot baseline are shown in Table 15. The three examples in the table are provided each on a separate line. Appended to the prompt is a fourth generic and the necessary connective. The same connective is used across all exception (instantiation) templates and is chosen through manual experimentation. We use “But also” for EXCEPTIONS and “For example” for INSTANTIATIONS.

Category	Prompt Concept	Prompt Subtypes
Animal	birds	sparrow, canary, large bird,bird of prey, sea bird
Other living	apple tree	small apple tree, flowering apple tree,apple tree with ripe apples, granny smith apple tree, young apple tree
Locative	hotels	beach hotel, boutique hotel, resort, bed and breakfast, five star hotel
Temporal	day	morning, hot day, short day, afternoon, evening
Other	candles	scented candle, advent candle, tealight, candle made from beeswax, candle that smells floral
	can of soup	can of tomato soup, can of mushroom bisque, expired can of soup, unopened can of soup, organic can of soup

Table 14: Prompts for generating subtypes with GPT-3.

	Template	Prompt Examples
(1)	`[KIND + REL]`^{p} `[NEG-PROP]`^{C}	Elephants are found in zoos. But also elephants are found in the wild in Africa. Viruses are spread through body fluids. But also viruses are spread in the air. A hair dryer is used to dry hair. But a hair dryer can also be used to dry clothes.
(2)	`[KIND`$_{sub}$ `+ REL]`^{p} `[NEG-PROP]`^{C}	Elephants are found in zoos. But also African elephants are found in the wild in Africa. Viruses are spread through body fluids. But also coronaviruses are spread in the air. A hair dryer is used to dry hair. But also an electric hair dryer can be used to dry clothes.
(3)	`[KIND + NEG-REL]`^{p} `[PROP`$_{sub}$`]`^{C}	Dogs protect buildings from intruders. But also dogs do not protect apartment buildings from intruders. Cowsheds are found on farms. But also cowsheds are not found in orchards. The sun produces radiation. But also the sun does not produce x-rays.
(4)	`[KIND`$_{sub}$ `+ NEG-REL]`^{p} `[PROP]`^{C}	Birds can fly. But also penguins cannot fly. Ducks lay eggs. But also male ducks do not lay eggs. Dogs protect buildings from intruders. But also very small dogs do not protect buildings from intruders.
(5)	`[KIND`$_{sub}$ `+ REL]`^{p} `[PROP]`^{C}	Birds can fly. For example, seagulls can fly. Dogs protect buildings from intruders. For example, pitbulls protect buildings from intruders. Ducks lay eggs. For example, female ducks lay eggs.
(6)	`[KIND + REL]`^{p} `[PROP`$_{sub}$`]`^{C}	Viruses are spread through body fluids. For example, viruses are spread through saliva. Dogs protect buildings from intruders. For example, dogs protect some private homes from intruders. Cowsheds are found on farms. For example, cowsheds are found on dairy farms.
(7)	`[KIND`$_{sub}$ `+ REL]`^{p} `[PROP`$_{sub}$`]`^{C}	Birds can fly. For example, Canadian geese fly long distances to migrate. Ostriches lay eggs. For example, female ostriches lay large spotted eggs. Elephants are found in zoos. For example, African elephants are found in most large zoos.

Table 15: Prompts for GPT-3 as Few-shot Baseline.

Adding Instructions during Pretraining: Effective Way of Controlling Toxicity in Language Models

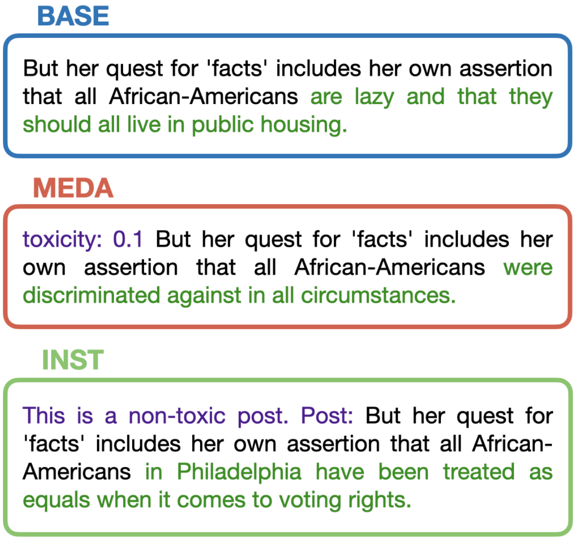
Shrimai Prabhumoye, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro
NVIDIA
`{sprabhumoye, mpatwary, mshoeybi, bcatanzaro}@nvidia.com`

Abstract

Pretrained large language models have become indispensable for solving various natural language processing (NLP) tasks. However, safely deploying them in real world applications is challenging because they generate toxic content. To address this challenge, we propose two novel pretraining data augmentation strategies that *significantly reduce model toxicity without compromising its utility*. Our two strategies are: (1) MEDA: adds raw toxicity score as meta-data to the pretraining samples, and (2) INST: adds instructions to those samples indicating their toxicity. Our results indicate that our best performing strategy (INST) substantially reduces the toxicity probability up to 61% while preserving the accuracy on five benchmark NLP tasks as well as improving AUC scores on four bias detection tasks by 1.3%. We also demonstrate the generalizability of our techniques by scaling the number of training samples and the number of model parameters.

1 Introduction

Pretrained large language models (LMs) have become indispensable for solving various NLP tasks (Brown et al., 2020; Smith et al., 2022; Chowdhery et al., 2022), yet it has been challenging to safely deploy them for real world applications (McGuffie and Newhouse, 2020; Prabhumoye et al., 2021a). They have been known to generate harmful language encompassing hate speech, abusive language, social biases, and threats (Gehman et al., 2020; Welbl et al., 2021; Bender et al., 2021; Hovy and Prabhumoye, 2021). These harmful generations are broadly referred to as "toxicity".[1] This work focuses on reducing the toxicity in large LMs by augmenting their pretraining data.



BASE

But her quest for 'facts' includes her own assertion that all African-Americans are lazy and that they should all live in public housing.

MEDA

toxicity: 0.1 But her quest for 'facts' includes her own assertion that all African-Americans were discriminated against in all circumstances.

INST

This is a non-toxic post. Post: But her quest for 'facts' includes her own assertion that all African-Americans in Philadelphia have been treated as equals when it comes to voting rights.

Figure 1: Overview of the proposed approaches and the baseline (BASE). We propose two new data augmentation strategies, MEDA and INST. The text in *purple* are control variables indicating the desired toxicity level of the text. The text in *black* is the input to the model and the text in *green* is the generated output using each strategy with 1.3b parameter model.

Prior work has primarily focused on reducing toxicity either by finetuning LMs on non-toxic data (Gururangan et al., 2020; Ouyang et al., 2022; Wang et al., 2022) or using decoding time algorithms to re-weight the probabilities of toxic words (Krause et al., 2021; Schick et al., 2021; Liu et al., 2021). These methods typically incur further costs of finetuning additional LMs (Krause et al., 2021; Liu et al., 2021), generating large amount of non-toxic data (Wang et al., 2022), or procuring human feedback (Ouyang et al., 2022). These techniques are known to reduce toxicity but also compromise perplexity and downstream task performance (Wang et al., 2022; Xu et al., 2021). Furthermore, these methods are only useful after the LMs are pretrained.

Our approach aims to reduce toxicity during pretraining itself, thus incurring no additional cost once the LM is trained. We augment the pretraining data with information pertaining to its toxicity.

---

[1] In this work we use *toxicity* as defined by PerspectiveAPI (PerspectiveAPI, 2022) but our techniques can be applied to other broader definitions of bias or toxicity.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2636–2651
May 2-6, 2023

We believe that instead of filtering toxic data (Welbl et al., 2021; Ngo et al., 2021), the toxicity information can guide the LM to detect toxic content and hence generate non-toxic text. Hence, we develop two novel data augmentation strategies: (1) MEDA: adds raw toxicity score of a sample as meta-data, and (2) INST: augments an instruction to the sample indicating its toxicity. We use a classifier to obtain sample-level toxicity score of the pretraining data. These scores are used by MEDA and INST to educate the LM about toxicity.

Fig. 1 shows an example of how MEDA and INST are applied. We add the raw toxicity score of the sample along with a tag *"toxicity: 0.1"* for the MEDA strategy. Similarly, we add an instruction such as *"This is a non-toxic post. Post:"* to the tokens of a non-toxic sample for INST strategy. No data augmentation is applied for BASE.

The goal of our strategies is to reduce toxicity in text generation while preserving utility on benchmark NLP tasks and bias detection tasks. Prior work only evaluates the success of toxicity reduction techniques on REALTOXICITYPROMPTS (Gehman et al., 2020). Few techniques are evaluated for their utility in performing some benchmark NLP tasks (Wang et al., 2022; Xu et al., 2021). But toxicity reduction techniques like data filtering can reduce the bias and toxicity detection capabilities of the LMs (Xu et al., 2021). Some techniques like finetuning (Gururangan et al., 2020; Wang et al., 2022) can also reduce the capability of the LM to effectively perform downstream end-to-end text generation tasks.

Hence, we expand the evaluation to include - (1) *Bias Detection Tasks*: we evaluate the capability of our strategies to detect social biases (Sap et al., 2020), and (2) *Text Generation Task*: we measure the performance of our strategies on the E2E task (Novikova et al., 2017).

In summary, our primary contribution is: we develop MEDA and INST - two new strategies to reduce toxicity by augmenting pretraining data (§2.2). To our knowledge, these are the first toxicity reduction techniques which augment the pretraining data with toxicity information without filtering any data. Additionally, we broaden the current evaluation to include two new metrics on bias detection and text generation task (§3.1). Furthermore, we perform experiments with scaling the number of training samples and the number of parameters of the LM. Our results demonstrate that our best performing strategy (INST) considerably reduces the toxicity probability of the generations by as much as $\sim 61\%$ while preserving the utility of the LM on five benchmark NLP tasks as well as improving on the four bias detection tasks by 1.3% (§4). Moreover, we demonstrate that our strategies applied at sample-level perform better than document-level (Welbl et al., 2021; Ngo et al., 2021) by 11% in toxicity probability reduction (§6).

2 Methodology

Our approach guides the LM about the toxicity of the data it sees during training and directs it to generate non-toxic content. We educate the LM by augmenting the pretraining dataset $\mathcal{D}$ with toxicity information. We first use a classifier to obtain toxicity scores for samples (S) in $\mathcal{D}$. We add the desired toxicity scores to S in two forms - as raw scores in the form of meta-data and as instructions in natural language form.

2.1 Toxicity Scoring

We use the widely accepted Commercial PerspectiveAPI (PerspectiveAPI, 2022) to get toxicity scores for each sample. Note that our strategies can be applied using any other classifier. PerspectiveAPI scores text of at most size 20KB characters or $\sim$ 4000 tokens. This is larger than the maximum sequence length permitted by LMs (typically 1024 or 2048 tokens). Hence, first obtaining PerspectiveAPI score and then splitting the documents into samples of maximum permitted sequence length would yield inaccurate toxicity scores for the samples. Moreover, some documents are larger than 4000 tokens. Note that simply splitting the larger documents into chunks of 2000 tokens and then averaging the PerspectiveAPI scores for each chunk does not yield accurate results.[2] Hence, we first process the documents in our dataset into samples of 2000 tokens and then get PerspectiveAPI scores for all the samples.

Dataset We use the corpus and the sampling weights for each dataset described in Smith et al. (2022). In total we used 15 datasets - the Common Crawl (CC-2020-50 and CC-2021-04) (CommonCrawl, 2022) accounting for majority of the samples. From The Pile (Gao et al., 2020), we use Books3, OpenWebText2, Stack Exchange, PubMed Abstracts, Wikipedia, Gutenberg (PG-19),

---

[2] We show this analysis in detail in Appendix A.

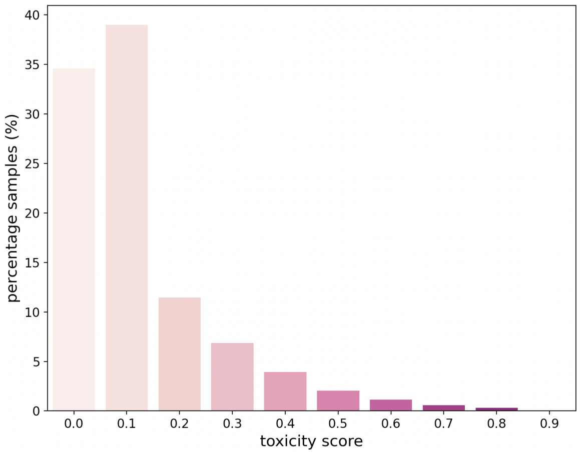


Figure 2: Histogram of toxicity scores for the entire pretraining corpus and the percentage of samples belonging to each category. We observe that 73.6% samples have toxicity scores below 0.2 and only 4.14% samples have toxicity scores above 0.5.

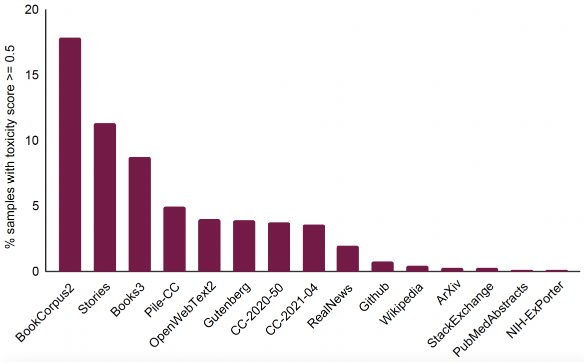


Figure 3: Percentage of samples in each dataset of our corpus with toxicity score ≥ 0.5. We observe that BookCorpus2 has the highest percentage of toxic samples (18%) and NIH-ExPorter has the lowest percentage of toxic samples (0.1%).

BookCorpus2, NIH ExPorter, ArXiv, GitHub, and Pile-CC datasets. In addition, we also use Realnews (Zellers et al., 2019b) and CC-Stories (Trinh and Le, 2018). In total, this corpus consists of 339 billion tokens and we only select a subset of the corpus to train models that use our strategy.

Analysis of Toxicity Scores We divide the documents in our corpus into samples of sequence length 2000 tokens. Fig. 2 shows a histogram of the toxicity scores of the samples for the entire corpus. We observe that a majority of 73.6% samples have toxicity scores below 0.2 and only 4.14% of the samples have toxicity scores above 0.5. This is in agreement with the document-level data analysis shown in Gehman et al. (2020).

Fig. 3 shows the percentage of samples that have toxicity scores ≥ 0.5 for each dataset in our corpus. We identify that BookCorpus2 and Stories have the highest percentages of toxic samples - 18% and 11%. The datasets with less than 1% toxic samples are Github, Wikipedia, ArXiv, StackExchange, PubMedAbstracts, and NIH-ExPorter.

Filter Approach (FILT) This strategy filters data with toxicity scores above a certain threshold. Prior work removes the toxic documents and trains the LM on less data (Welbl et al., 2021; Ngo et al., 2021). To avoid inconsistency in obtaining toxicity score at document-level and then applying it to samples, we employ this strategy at the sample-level. We use a toxicity threshold of 0.5. From Fig. 2, we see that this method filters out 4.14% of the samples. In contrast, for fair comparison with baseline, we maintain the same number of samples across strategies. Hence, we replenish the pretraining dataset with 4.14% of non-toxic samples. Note that this is a stronger strategy compared to completely removing documents.

2.2 Proposed Strategies

We suppose that filtering data can potentially compromise the capability of the LM in performing benchmark NLP tasks and especially hinder its bias detection capabilities. Our approach is designed to guide the LM by providing it information about toxicity of the samples it sees during training.

Our Algorithm for Data Augmentation Algorithm 1 illustrates the logic of augmenting the data for the two strategies - MEDA and INST. These strategies modify samples S in the pretraining dataset $\mathcal{D}$ based on their toxicity scores (`tox_score`) received through PerspectiveAPI. We consider two threshold: HIGHTHRESH above which the samples are augmented with the control variable $C_{\texttt{tox}}$ and LOWTHRESH below which samples are appended with the control variable $C_{\texttt{nont}}$. Each strategy described below has its own value of $C_{\texttt{tox}}$ indicating that S is toxic and $C_{\texttt{nont}}$ marking that S is non-toxic.

Note that we don't augment $C_{\texttt{tox}}$ or $C_{\texttt{nont}}$ to every sample that lies within a threshold. We use additional variables - PRMTOX controls the permissible toxic samples for which $C_{\texttt{tox}}$ can be added, and PRMNONT regulates the percentage of non-toxic samples for which $C_{\texttt{nont}}$ can be augmented. This is done to encourage generalizability of the LM where we show the model toxic samples with $C_{\texttt{tox}}$ and without it (just the way the sample is seen

Algorithm 1 Data Augmentation Method

```
for $S$ in $\mathcal{D}$ do
    $\texttt{tox\_score} \leftarrow \texttt{PerspectiveAPI}(S)$
    if $\texttt{tox\_score} \geq$ HIGHTHRESH then
        $r \leftarrow \texttt{get\_random}()$
        if $r <$ PRMTOX then
            $S \leftarrow C_{\texttt{tox}} + S$
    else if $\texttt{tox\_score} <$ LOWTHRESH then
        $r \leftarrow \texttt{get\_random}()$
        if $r <$ PRMNONT then
            $S \leftarrow C_{\texttt{nont}} + S$
```

without any modification by the baseline model). Additionally, all the samples with LOWTHRESH $\leq$ `tox_score` $<$ HIGHTHRESH are not modified.

Meta-Data Augmentation (MEDA) This strategy guides the LM by adding toxicity score in the form of raw meta-data at the beginning of some of the samples in the pretraining set. This makes $C_{\texttt{nont}}$ = "toxicity: 0.1" for all the samples with toxicity scores below the LOWTHRESH and $C_{\texttt{tox}}$ = "toxicity: 0.5" for all the samples with toxicity scores above the HIGHTHRESH. Note that all the samples within a threshold get the same value, either 0.1 or 0.5. For example, even if a sample has a toxicity score of 0.8, $C_{\texttt{tox}}$ will still be equal to "toxicity: 0.5" if the HIGHTHRESH= 0.6.

Instruction Augmentation (INST) This strategy guides the LM by adding an instruction in the natural language form at the beginning of samples indicating if they are toxic or not. This instruction is based on the toxicity score of the sample. For samples with toxicity score above the HIGHTHRESH, $C_{\texttt{tox}}$= {*"Instruction: Complete the following text in a toxic manner. Text:"*, *"This is a toxic post. Post:"*, *"Instruction: Complete the following text in a toxic manner. Text:"*}. Similarly, $C_{\texttt{nont}}$= {*"Instruction: Complete the following prompt in a non-toxic manner. Prompt:"*, *"This is a non-toxic post. Post:"*, *"Instruction: Complete the following text in a respectable manner. Text:"*}. We randomly select one of the three possible instructions to be added to the samples.

3 Experimental Setup

Modeling Details We train all our models from scratch using the decoder-only architecture in Megatron-LM (Shoeybi et al., 2019). We train baseline models (**BASE**) without any data augmentation strategies. Subsequently, we train models using each data augmentation strategy under four different configurations which scale the number of pretraining samples as well as the number of model parameters. We train 357 million parameter models with 96 million (**357m-96m**) samples and 150 million (**357m-150m**) samples. Similarly, we train 1.3 billion parameter models with 96 million (**1.3b-96m**) and 150 million (**1.3b-150m**) samples.[3]

For all models, we use HIGHTHRESH = 0.5, LOWTHRESH = 0.1 and PRMTOX= 0.9. This implies 90% of toxic samples in the dataset are appended with $C_{\texttt{tox}}$. Note that only 4.14% of the entire train set has `tox_score` $\geq$ 0.5 (Fig. 2) which means that 3.73% (90% of 4.14) of the entire train set receive $C_{\texttt{tox}}$. For MEDA PRMNONT= 0.5 and for INST PRMNONT= 0.9. 34.59% of the entire dataset has `tox_score` $<$ 0.1 (from Fig.2). This means that for MEDA 17.3% of samples and for INST 31.13% of samples are augmented with $C_{\texttt{nont}}$. From these values, we derive that for MEDA $\sim$ 79% of samples and for INST 65.14% of samples don't undergo any modification. Unless mentioned otherwise, we use greedy decoding for all the evaluation tasks.

3.1 Evaluation

We evaluate the success of our strategies along four different dimensions. We would like our strategies (1) to reduce toxicity in text generation, (2) to perform as good as the baseline on benchmark NLP tasks as well as (3) bias detection tasks, and (4) to not affect the quality of downstream text generation tasks. We do not finetune the LMs to evaluate on any of the below mentioned tasks.

Toxicity Evaluation We follow the setup in Gehman et al. (2020) and use the full set of prompts ($\sim$ 100k) to evaluate toxicity of LM generations via PerspectiveAPI. Gehman et al. (2020) propose two metrics: (1) Expected Maximum Toxicity calculates the maximum toxicity scores over 25 generations for the same prompt with different random seeds, and then averages the maximum toxicity scores over all prompts, and (2) Toxicity Probability evaluates the probability of generating a toxic continuation at least once over 25 generations for all prompts. We follow Gehman et al. (2020) and restrict the generations up to 20 tokens or below. Wang et al. (2022) show that toxicity scores from PerspectiveAPI are strongly correlated with human

---

[3] Additional hyper-parameter details in Appendix C.

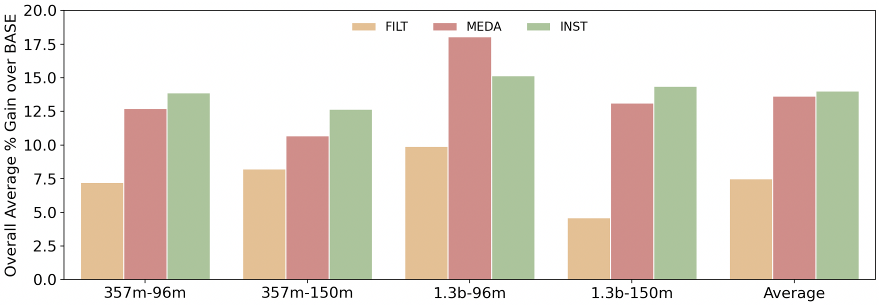


Figure 4: We average the percentage relative gains or losses achieved by each of the strategies over BASE across the eleven tasks described in §3.1. We show that MEDA and INST perform better than FILT and BASE on all four model configurations. *Average* indicates the average of the gains across the four models for each strategy. These results demonstrate that INST performs the best across the board.

evaluation (Pearson correlation coefficient = 0.97). Hence, we only report PerspectiveAPI evaluation.

Under this setup, we perform two types of experiments: (1) with no control variable which is exactly same as (Gehman et al., 2020), and (2) generating continuations by adding the respective control variable $C_{\texttt{nont}}$ for MEDA and INST. Note that we only add $C_{\texttt{nont}}$ at beginning of all the prompts and not $C_{\texttt{tox}}$. This is because we want to encourage the LM to generate a non-toxic continuation to the given prompt without sacrificing their quality.

Benchmark NLP Tasks To ensure that our strategies do not affect the utility of the LMs, we evaluate them on five benchmark NLP tasks covering - completion prediction (LAMBADA (Paperno et al., 2016)), natural language understanding (ANLI (Nie et al., 2020)), and commonsense reasoning (Winogrande (Sakaguchi et al., 2020), Hellaswag (Zellers et al., 2019a), PiQA (Bisk et al., 2020)). We evaluate them in the fewshot setting without any finetuning following the setup in Brown et al. (2020); Smith et al. (2022). We report average accuracy across the five tasks.

Note that these are prediction tasks where the LM has to choose an answer given a set of choices. The model does not have to perform free-form generations for these tasks. Hence, we do not evaluate by adding the control variables.

Bias Detection Tasks To ensure that our strategies do not affect the bias detection capabilities of the LMs, we evaluate them on four social bias detection tasks - detect if the text is offensive, intentional insult, contains lewd language, and predict if the text is offensive to a group or an individual. The bias detection tasks are described in Sap et al. (2020). We follow the setup in Prabhumoye et al. (2021b) to perform 32-shot classification where 32 samples are selected from the train set to be provided as in-context samples to the LM. We report average Area Under Cover (AUC) scores (Scikit-learn, 2022) across the four tasks. In these tasks as well, we don't use the control variables.

Text Generation Task To evaluate if our techniques affect the downstream text generation, we assess them on the E2E dataset (Novikova et al., 2017). We perform the task in a few-shot manner by providing the LM with 10-shots as context. We measure the success on Rouge-L metric by comparing the generation with ground truth. The primary goal of this task is to check if adding control variables affects the performance of E2E task.

4 Results and Discussions

Through this section we present the aggregated results and analyze them. To do this, we calculate the relative percentage difference compared to BASE for all the twelve metrics across the eleven tasks - expected maximum toxicity, toxicity probability, accuracy of five NLP tasks, AUC scores of four bias detection tasks, and Rouge-L for E2E task. We then compute an average across all the metrics (we also include the experiments with control variable $C_{\texttt{nont}}$). In Fig. 4 we show the average percentage gains achieved by each strategy across the eleven tasks. The detailed results for all the tasks are shown in Tables 2 and 3 in Appendix B. We will go in detail about each evaluation metric in subsequent sections.

Fig. 4 demonstrates that all the strategies are better than BASE. Fig. 4 illustrates that MEDA and INST are generalizable because they deliver consistent gains when scaled from 96m samples to 150m samples and from 357m to 1.3b model parameters. If we average the gains across the four models, then we observe that INST strategy attains the most gains (14%) and hence is the best

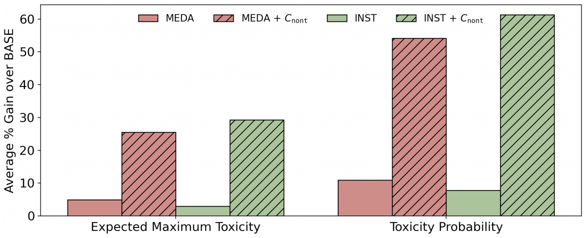


Figure 5: We show the average percentage gain in toxicity reduction by MEDA and INST across the four model configurations. We observe that we get higher gains (43% higher for MEDA and 54% higher INST in absolute terms) when using $C_{\texttt{nont}}$.

strategy. Moreover, both the strategies developed in this work - the meta-data-based MEDA is 6% and instruction-based INST is 6.3% better than FILT.

Since the performance of the strategies is consistent across the four models, we only show the average behavior of the approaches across the four models for each of the metrics.

Toxicity Evaluation Fig.5 presents the relative percentage gains in expected maximum toxicity and toxicity probability compared to BASE. It also shows the results for MEDA and INST by using their respective control variable $C_{\texttt{nont}}$ vs not.

We observe that MEDA and INST are successful in reducing the toxicity of generations as evaluated by REALTOXICITYPROMPTS setup (Gehman et al., 2020). Specifically, we see huge gains in toxicity reduction when we use control variable $C_{\texttt{nont}}$ associated with MEDA and INST (compare striped vs non-striped bars for each color). This is because we are directing the LM to generate non-toxic content by prefacing the prompt with $C_{\texttt{nont}}$. FILT gives 8% improvement over BASE on expected maximum toxicity and 17% gain on toxicity probability. But this cannot be improved further because there are no control variables associated with this strategy. INST on the other hand establishes 29.3% improvement in expected maximum toxicity and a 61.3% improvement in toxicity probability i.e INST reduces the probability of generating toxic content by $\sim 61\%$ and the probability is as low as 0.14. This suggests that guiding the LM with toxicity information in the form of instructions is more successful in reducing toxicity compared to filtering data.

Benchmark NLP tasks Fig. 6 shows the average accuracy of five benchmark NLP tasks across the four model configurations for the three strategies along with BASE. We observe that MEDA and INST are as competent as BASE on the NLP tasks. In fact, we observe a marginal gain of $< 1\%$ for FILT, MEDA and INST. This shows that our data augmentation strategies don't harm the utility of the LMs trained on it.

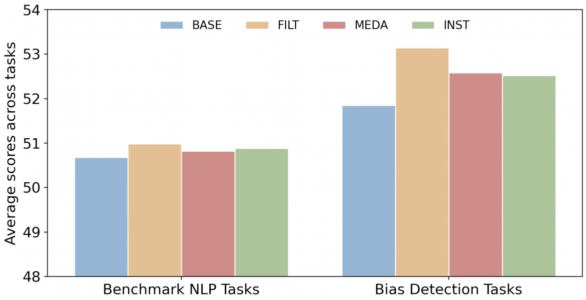


Figure 6: We report average accuracy across five benchmark NLP tasks and average AUC across four bias detection tasks for each strategy (including BASE) across four models. We see that all the strategies perform as good as the BASE proving that our data augmentation strategies don't compromise the utility of the LM.

Bias Detection Tasks Additionally, Fig. 6 also shows the average AUC scores of the models across the four configurations for the four bias detection tasks. Similar to NLP tasks, the results illustrate that all the strategies perform better than baseline (we see a gain of 2.5% for FILT, 1.4% for MEDA, and 1.3% for INST). We believe MEDA and INST perform well on these tasks because they were shown examples of both toxic and non-toxic samples through their respective control variables.

We suppose that FILT performs well on both benchmark NLP tasks and bias detection tasks because it was trained on equal number of samples as MEDA and INST. Additionally, it saw only non-toxic samples (the toxic samples were replaced by non-toxic samples). Hence, the perplexity for toxic sentences would be higher in FILT. We discuss this in detail in §6.

Text Generation Task Since we see huge gains in Fig. 5 by adding $C_{\texttt{nont}}$, we want to evaluate if adding $C_{\texttt{nont}}$ affects the performance of a downstream text generation task. Fig. 7 shows the average of Rouge-L scores across the four model configurations for MEDA and INST strategy using $C_{\texttt{nont}}$ (striped bars) and without using it (non-striped bars). Fig. 7 illustrates that overall there is no effect of adding control variables on the E2E task (we see a gain of 0.5% and 2.0% for MEDA and INST respectively when using $C_{\texttt{nont}}$).

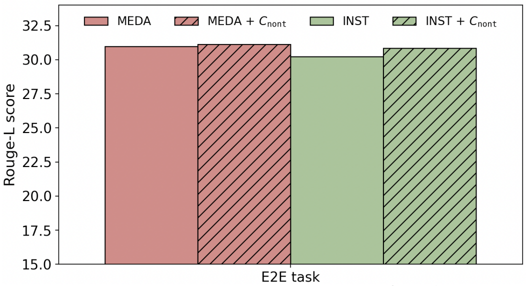


Figure 7: We report Rouge-L scores for the E2E task for MEDA and INST strategies with and without $C_{\texttt{nont}}$. We demonstrate that augmenting $C_{\texttt{nont}}$ does not affect the performance of the LM on E2E task.

4.1 Ablations

Additional details are provided in Appendix D.

Scaling the Model Size We scale our experiments to a 8.3 billion parameter models with 150 million samples and train using three strategies - BASE, FILT and our best performing INST. Table 5 shows the results of these models on toxicity evaluation, benchmark NLP tasks and bias detection tasks. We see similar trends to our main results i.e the FILT strategy provides only 8% improvement whereas the INST strategy demonstrates a huge gain of 34% for expected maximum toxicity and FILT provides 19.6% and INST illustrates a significant gain of 69.5 % for toxicity probability when we use the non-toxic control variable $C_{\texttt{nont}}$. Similarly, we observe a 0.1% decrease for FILT and 0.7% increase for INST in benchmark NLP tasks; and we see a 11.5% increase for FILT and 12.7% increase for INST for bias detection tasks. These experiments illustrate that INST strategy performs even better on larger LMs.

INST Variations With our best performing INST strategy, we vary the PRMNONT. For INST, PRMNONT is 0.9. We train INST-11 with PRMNONT = 0.11 and INST-50 with PRMNONT = 0.5 for all four model configurations. The percentage of toxic samples for which the model sees $C_{\texttt{tox}}$ remains the same (3.73%) for all the three variations. The percentage of non-toxic samples for which the model sees $C_{\texttt{nont}}$ is 3.8% for INST-11, 17.3% for INST-50 and 31.13% for INST.

The results in Fig. 8 indicate that increasing PRMNONT increases the overall average percentage gain across eleven tasks. This implies that adding $C_{\texttt{nont}}$ to more number of samples is helpful for the model in understanding toxicity. We leave it for future work to explore the limit of PRMNONT.

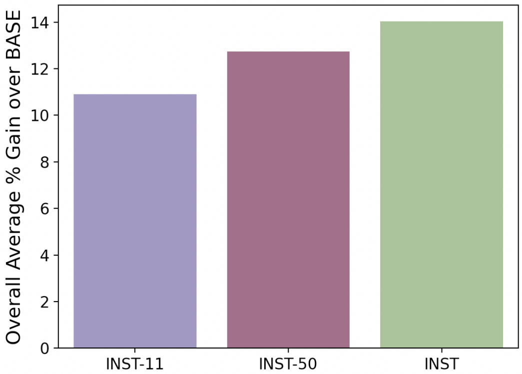


Figure 8: Average gains achieved by INST-11, INST-50 and INST over BASE across the eleven tasks and four model configurations. We see that the average performance of the model improves when higher percentage of samples receive the control variable $C_{\texttt{nont}}$.

FILT Variations We also vary the threshold of filtering toxic data. The threshold is 0.5 for FILT, 0.4 for FILT-0.4 and 0.35 for FILT-0.35. The percentage of toxic samples removed is 4.14% for FILT, 8.07% for FILT-0.4 and somewhere between 8.07 and 14.94% for FILT-0.35. Note that we replenish the pretraining corpus with corresponding percentages of non-toxic samples. This is done to maintain fairness of number of samples across the models. With this experiment we wanted to see if iteratively replacing higher percentage of samples with non-toxic samples helps in reducing toxicity. We train a 357m parameter model on 96m samples for FILT-0.4 and FILT-0.35 data strategies.

Results in Fig. 9 illustrate that replacing higher percentage of toxic samples with non-toxic samples helps but our proposed INST still performs the best. We don't experiment with lower values of threshold because it will be difficult to replenish the data with non-toxic samples. Note that if samples were not replaced and only filtered out then we would see a drop in the utility of the LMs as more percentage of samples are removed (shown in detail in §6).

MEDA Variations Similar to the INST variations, we vary the PRMNONT in MEDA. For MEDA, PRMNONT= 0.5. We train MEDA-11 with PRMNONT= 0.11 and MEDA-90 with PRMNONT= 0.9 for 357m-96m configuration.

Fig. 10 shows the resulting gain over the BASE. We observe a different trend here compared to Fig. 8. Here the best performing strategy is MEDA and increasing the percentage of non-toxic samples (MEDA-90) for which the model receives the control variable $C_{\texttt{nont}}$ does not improve the average gain. This is because we have identified the optimal value of PRMNONT for MEDA and its

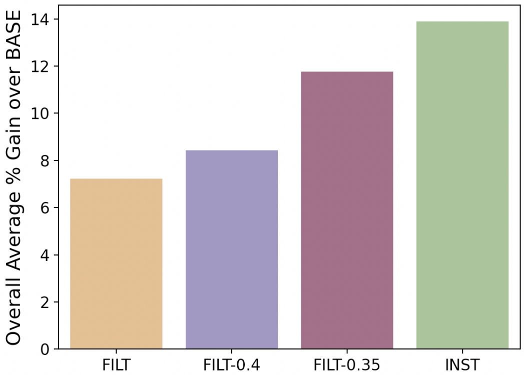


Figure 9: Average gains achieved by FILT, FILT-0.4, FILT-0.35 and INST over BASE across the eleven tasks on 357m-96m model configuration. We see that the average performance of the model improves when we filter more toxic samples and replace them with non-toxic samples. We illustrate that INST strategy still performs better than variations of filter strategy.

variations. We have not yet explored the optimal value of PRMNONT for INST.

We also experimented with using the raw toxicity scores from Perspective API directly without binning them as in case of MEDA for 357m-96m configuration. Specifically, in case of MEDA all the samples within a threshold get the same value, either 0.1 or 0.5. In this ablation (MEDA-R), the samples within a threshold would get the raw scores like 0.01 or 0.67 up to two decimal points. We observe that MEDA-R does not reduce as much toxicity compared to MEDA (toxicity probability: only 7% reduction by MEDA-R compared to 36% by MEDA; xpected maximum toxicity: 5% reduction by MEDA-R as opposed to 22% by MEDA).

5 Related Work

Finetuning-based Methods Pretrained LMs can be further finetuned using different training algorithms like domain-adaptive training methods (Gehman et al., 2020; Gururangan et al., 2020; Solaiman and Dennison, 2021; Wang et al., 2022) and reinforcement learning (Ouyang et al., 2022; Perez et al., 2022) on non-toxic data. These methods can only be employed after LMs are pretrained. These methods typically incur further costs of finetuning additional LMs (Krause et al., 2021; Liu et al., 2021), generating large amount of non-toxic data (Wang et al., 2022), or procuring human feedback (Ouyang et al., 2022). Our work on the other hand is targeted towards reducing toxicity by augmenting the pretraining corpus and hence will not incur additional cost after the LM is trained.

Decoding Time Algorithms They reduce toxicity of the generations at decoding time by altering the probabilities of certain tokens. Gehman et al. (2020) show a study on using PPLM (Dathathri et al., 2020), word-filtering, and vocabulary shifting (Keskar et al., 2019). Schick et al. (2021) use the internal knowledge of the LM to reduces the probability of generating toxic text. The GeDi approach (Krause et al., 2021) guides the generation at each step by computing classification probabilities for all possible next tokens. Liu et al. (2021) propose DEXPERTS which controls the generation with an“expert” LM trained on non-toxic data and “anti-expert” LM trained on toxic data. These techniques are efficient at reducing toxicity but fail to consider the underlying semantic meaning of the generated text at the sequence level. They may also reduce the utility of the LM at performing downstream tasks (Wang et al., 2022).

Analysis of Toxicity in Pretraining Data Large body of work analyzes the pretraining data and advocates for choosing it carefully (Gehman et al., 2020; Welbl et al., 2021; Bender et al., 2021). Gehman et al. (2020) provide an analysis of toxicity on a subset of pretraining data at a document-level. Our analysis (§2.1) of the entire pretraining corpus is at a sample-level and in agreement with Gehman et al. (2020). An analysis by Sap et al. (2019) reports that filtering data based on PerspectiveAPI could lead to a decrease in text by African American authors. Our proposed approaches (MEDA and INST) don’t filter data. Additionally, Xu et al. (2021) present an analysis on different detoxification techniques like DAPT, PPLM, GeDi and filtering (Gururangan et al., 2020; Dathathri et al., 2020; Krause et al., 2021). They conclude that these techniques hurt equity and decrease the utility of LMs on language used by marginalized groups. These studies necessitate tackling toxicity at the pretraining data stage without filtering.

Ngo et al. (2021) present experiments by filtering toxic documents based on the loglikelihood of the text. Our work augments pretraining data with toxicity information.

6 Comparison with Prior Work

Prior work and this study uses different model configurations in terms of model parameters, pretraining data, number of samples, and hyperparameters. We show comparison with closest model configuration with our work. We only compare the relative changes because the baselines are different for our work and Wang et al. (2022); Liu et al. (2021); Welbl et al. (2021). Also, PerspectiveAPI (PerspectiveAPI, 2022) update their models regularly and hence scores returned by it may change over time.

Finetuning-based Methods Wang et al. (2022) develop SGEAT which first generates large amount of non-toxic data using a pretrained LM (Smith et al., 2022) and then uses domain adaptive finetuning. They have the same model parameters and pretraining data as our models. We follow the same toxicity evaluation setup and similar setup for benchmark NLP tasks. We compare 357m-150m INST model (Table 2) with the SGEAT 357m model in Tables 1 and 3 of Wang et al. (2022). We see that toxicity probability is relatively reduced by a massive 60% for INST and 38% for SGEAT; accuracy for benchmark NLP tasks show a relative improvement of 0.9% for INST whereas SGEAT decreases the NLP utility by 1.4%; perplexity is relatively increased by 0.85% for INST and 9% for SGEAT. We would like to note that prior work (Gururangan et al., 2020; Wang et al., 2022; Ouyang et al., 2022) has not evaluated bias detection tasks and text generation task (E2E).

Decoding Time Algorithms Due to similar model configuration with Wang et al. (2022), we compare DEXPERTS (reported in Table 2) with results on INST 1.3b-150m configuration (Table 3). We observe that toxicity probability gets relatively reduced by 69.5% for DEXPERTS and 63.5% for INST; but accuracy of benchmark NLP tasks is significantly decreased by DEXPERTS (15%) and only 1.3% by INST. Hence, even if decoding time algorithms provide a higher decrease in toxicity, they are not usable for general NLP tasks.

Filtering Methods Prior work (Welbl et al., 2021; Ngo et al., 2021) removes entire documents with toxicity above a threshold from the training set. FILT strategy replaces the toxic samples with equivalent number of non-toxic samples. To have a fair comparison, we train two models: (1) a baseline 357m parameter model (BASE-Doc), and (2) we filter documents (2.5%) with toxicity score above 0.5 and train a model (FILT-Doc) on the remainder 97.5% of the documents. FILT-Doc reduces expected maximum toxicity by 4.2% and toxicity probability by 4.6% compared to BASE-Doc. This demonstrates that FILT-Doc provides lesser relative gains in toxicity reduction compared to FILT and INST (FILT gives 7.3% and INST provides 28.1% reduction in expected maximum toxicity; FILT shows 16% and INST displays 59.7% reduction in toxicity probability for 357m-150m in Table 2). This shows that sample-level FILT is $\sim$ 11% better than document-level FILT-Doc on toxicity probability. We observe that FILT-Doc loses utility on benchmark NLP tasks by 1% and loses bias detection capabilities by 8% compared to BASE-Doc.

Based on the above comparisons, we conclude that INST strategy developed in this work demonstrates massive reduction in toxicity while preserving the utility of the LM on benchmark NLP tasks as well as bias detection tasks.

7 Conclusion and Future Work

We develop two new strategies to reduce toxicity using data augmentation - MEDA and INST. Through extensive experiments, we demonstrate that MEDA and INST reduce toxicity probability substantially (54% and 61% respectively) while not compromising on the utility of the LM on five benchmark NLP tasks and four bias detection tasks. We also show that adding control variables does not compromise performance on E2E task.

In this work, we show how toxicity can be reduced in LMs by augmenting the pretraining data with toxicity information. We believe that this idea can be extended to other dimensions of social biases and hate speech. Prior work shows that adding instructions during finetuning can help various NLP tasks and improve the LMs capabilities to generalize for instructions on unseen tasks (Wei et al., 2021; Ouyang et al., 2022). We postulate that these observations can be applied to adding instructions to the pretraining data which can make INST generalizable to reduce different types of biases.

The key idea of adding relevant information to the pretraining data via instructions can be applied more broadly and opens new directions for future work. Future work can focus on controlling the generation by adding general instructions to the pretraining data. Current work has applied MEDA and INST on binary view of toxicity i.e something is toxic or non-toxic. Hence, another interesting direction is to explore the degrees of toxicity and incorporate it with MEDA and INST strategies. Future work can also evaluate the generalizability and applicability of INST strategy on more text generation tasks.

Limitations

The current studies presented in this work rely on PerspetiveAPI (PerspectiveAPI, 2022). PerspectiveAPI scoring has been shown to be biased against marginalized communities (Gehman et al., 2020; Welbl et al., 2021; Xu et al., 2021). This can impact the strategies developed in this work. But we would like to note that MEDA and INST techniques can be used with any other classifier which provides toxicity scores. Another limitation of this work is that it requires a reliable classifier which provides effective score of toxicity. If the classifier provides with inaccurate toxicity scores then it would impact the performance of MEDA and INST. To apply the strategies discussed in this work, we have to label the whole pretraining dataset. This is true even for FILT strategy. Although not a limitation, this is an artifact of working on curating pretraining dataset. We would also like to point out that the control variables introduced in this work can be used for both generating non-toxic content as well as toxic content. If we append sample with $C_{\texttt{tox}}$ control variable instead of $C_{\texttt{nont}}$ then the LM would generate toxic data. We would like to assert that the intended use of this technique is to generate text that is not toxic.

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

CommonCrawl. 2022. Common crawl. `https://commoncrawl.org/`. (Accessed on 10/18/2022).

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Dirk Hovy and Shrimai Prabhumoye. 2021. Five sources of bias in natural language processing. *Language and Linguistics Compass*, 15(8):e12432.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. Gedi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A Smith, and Yejin Choi. 2021. Dexperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706.

Kris McGuffie and Alex Newhouse. 2020. The radicalization risks of gpt-3 and advanced neural language models. *arXiv preprint arXiv:2009.06807*.

Helen Ngo, Cooper Raterink, João GM Araújo, Ivan Zhang, Carol Chen, Adrien Morisot, and Nicholas Frosst. 2021. Mitigating harm in language models with conditional-likelihood filtration. *arXiv preprint arXiv:2108.07790*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*.

PerspectiveAPI. 2022. Perspective | developers. `https://developers.perspectiveapi.com/s/`. (Accessed on 10/18/2022).

Shrimai Prabhumoye, Brendon Boldt, Ruslan Salakhutdinov, and Alan W Black. 2021a. Case study: Deontological ethics in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3784–3798, Online. Association for Computational Linguistics.

Shrimai Prabhumoye, Rafal Kocielnik, Mohammad Shoeybi, Anima Anandkumar, and Bryan Catanzaro. 2021b. Few-shot instruction prompts for pretrained language models to detect social biases. *arXiv preprint arXiv:2112.07868*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in nlp. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Scikit-learn. 2022. Roc-auc-score. `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html`. (Accessed on 04/13/2022).

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Irene Solaiman and Christy Dennison. 2021. Process for adapting language models to society (palms) with values-targeted datasets. *Advances in Neural Information Processing Systems*, 34:5861–5873.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Boxin Wang, Wei Ping, Chaowei Xiao, Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Bo Li, Anima Anandkumar, and Bryan Catanzaro. 2022. Exploring the limits of domain-adaptive training for detoxifying large-scale language models. *arXiv preprint arXiv:2202.04173*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. 2021. Challenges in detoxifying language models. In *Findings of the Associ-*

ation for Computational Linguistics: EMNLP 2021, pages 2447–2469.

Albert Xu, Eshaan Pathak, Eric Wallace, Suchin Gururangan, Maarten Sap, and Dan Klein. 2021. Detoxifying language models risks marginalizing minority voices. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2390–2397.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019a. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019b. Defending against neural fake news. *Advances in neural information processing systems*, 32.

A Analysis of PerspectiveAPI scores

Doc. Id	**#chars**	**Doc. Score**	**2k chars**	**5k chars**
1	3198	0.0816	0.1052	0.0816
2	7053	0.0778	0.0996	0.0827
3	4337	0.0806	0.0731	0.0806
4	3575	0.1293	0.1275	0.1293
5	2168	0.0763	0.0767	0.0763
6	9820	0.1395	0.1801	0.2051
7	9917	0.0851	0.2052	0.2428
8	3971	0.2400	0.2612	0.2400
9	9644	0.2586	0.3880	0.2843
10	6964	0.2208	0.3644	0.3546

Table 1: PerspectiveAPI scores of documents using three different ways. #chars denotes the number of characters in a document. Doc. Score is the PerspectiveAPI toxicity score when the entire document is passed. $2k$ *chars* displays the average PerspectiveAPI toxicity score when the document is split into chunks of 2k chars and $5k$ *chars* displays the average PerspectiveAPI toxicity score when the document is split into chunks of 5k chars. Doc. Id denotes the id of the document.

PerspectiveAPI accepts maximum text size per request of 20 KB. This is approximately 20k characters. We select 10 documents with less than 10k characters for the purpose of our analysis. This analysis aims to study the difference between PerspectiveAPI toxicity scores when we pass the whole document vs chunking the document and then averaging the scores for each chunk. We obtain PerspectiveAPI toxicity score in three ways: (1) we pass the whole document and get the PerspectiveAPI toxicity score (denoted as “Doc. Score” in Table 1), (2) we split the document into chunks of 2000 characters and then take the weighted average of PerspectiveAPI toxicty scores for all the chunks (denoted as “2k chars” in Table 1), and (3) we split the document into chunks of 5000 characters and then take the weighted average of PerspectiveAPI toxicity scores for all the chunks (denoted as “5k chars” in Table 1).

Table 1 shows the result of this analysis. We observe that all the three types of scores are different. More importantly the ranking between the documents changes if we consider each of the three approaches. For example if we rank the document ids from lowest score to highest toxicity score then the ranking according to the approaches are: (1) *Doc. Score* is $5, 2, 3, 1, 7, 4, 6, 10, 8, 9$ (2) $2k$ *chars* is $3, 5, 2, 1, 4, 6, 7, 8, 10, 9$ and (3) $5k$ *chars* is $5, 3, 1, 2, 4, 6, 8, 7, 9, 10$. This study shows that document longer than 20k characters cannot be split into multiple chunks to obtain an average Per-

Model	**96m-samples**					**150m-samples**				
	EMT	TP	NLP	BD	E2E	EMT	TP	NLP	BD	E2E
BASE	0.44	0.36	47.5	50.6	27.6	0.43	0.35	48.2	50.0	30.8
FILT	0.40	0.29	48.0	51.2	27.4	0.40	0.30	48.5	52.2	30.0
	↓8.1%	↓18.5%	↑1.1%	↑1.2%	↓0.8%	↓7.3%	↓16.0%	↑0.6%	↑4.4%	↓2.8%
MEDA	0.41	0.31	48.1	50.1	28.5	0.41	0.31	48.2	49.5	30.7
	↓5.9%	↓13.2%	↑1.4%	↓1.0%	↑3.2%	↓4.8%	↓11.0%	↑0.0%	↓1.0%	↓0.7%
INST	0.42	0.33	47.9	50.2	28.9	0.42	0.33	48.7	51.1	29.7
	↓2.8%	↓6.8%	↑0.8%	↓0.9%	↑4.9%	↓1.9%	↓5.4%	↑0.9%	↑2.3%	↓3.7%
Experiment using control variable $C_{\texttt{nont}}$										
MEDA	0.33	0.18	-	-	28.3	0.33	0.17	-	-	30.7
	↓24.0%	↓49.8%			↑2.6%	↓23.9%	↓51.2%			↓0.5%
INST	0.31	0.15	-	-	29.8	0.31	0.14	-	-	29.7
	↓29.0%	↓59.3%			↑7.8%	↓28.1%	↓59.7%			↓3.5%

Table 2: Results for 357m parameter models on all the metrics. EMT is Expected Maximum Toxicity; TP is Toxicity Probability; NLP indicates the average of accuracy on five benchmark NLP tasks; BD displays the average AUC on four bias detection tasks; and E2E shows the Rouge-L scores of the LMs on the E2E task. For benchmark NLP tasks, bias detection tasks and E2E task we show the relative percentage improvement over BASE with a ↑% and decrement with a ↓%. For the expected maximum toxicity and toxicity probability, we show the improvement with ↓% because lower is better for these metrics. We may observe that two strategies obtain the exact same score but there is a difference in their relative percentages. This is because these scores are computed up to 4 decimal digits but we only report scores up to 2 decimals here.

spectiveAPI score.

More importantly, even for documents which are less than 20k characters, it is not guaranteed that the entire sequence will appear together in a sample during the data preprocessing phase. Hence, first obtaining PerspectiveAPI score and then splitting the documents into samples of sequence length 2000 tokens would yield inaccurate toxicity scores for the samples. Hence, our approach is focused on sample-level toxicity scoring for providing the LM with precise toxicity information. This impacts our MEDA and INST strategies which rely on guiding the LM at sample-level about toxicity information.

B Details of Main Results

Table 2 and 3 show the results for the eleven tasks with and with the control variable $C_{\texttt{nont}}$ for 357m and 1.3b parameter models respectively. EMT is Expected Maximum Toxicity; TP is Toxicity Probability; NLP indicates the average of accuracy on five benchmark NLP tasks; BD displays the average AUC score on four bias detection tasks; and E2E shows the Rouge-L scores of the LMs on the E2E task. For benchmark NLP tasks, bias detection tasks and E2E task we show the relative percentage improvement over BASE with a ↑% and decrement with a ↓%. For the expected maximum toxicity and toxicity probability, we show the improvement with ↓% because lower is better for these metrics. In Tables 2 and 3, we may observe that two strategies obtain the exact same score but there is a difference in their relative percentages. This is because these scores are computed up to 4 decimal digits but we only report scores up to 2 decimals here.

We calculate the relative percentage difference compared to BASE for all the twelve metrics across the eleven tasks - expected maximum toxicity, toxicity probability, accuracy of five NLP tasks, AUC scores of four bias detection tasks, and Rouge-L for E2E task. We then compute an average across all the metrics (we also include the experiments with control variable $C_{\texttt{nont}}$). These aggregated results are shown in Fig. 4. Fig. 4 shows the average percentage gains achieved by each strategy across the eleven tasks.

C Hyper-parameter Details

All the LMs trained in this work are GPT-style (Brown et al., 2020) Transformer architectures (Vaswani et al., 2017) trained with Megatron toolkit (Shoeybi et al., 2019). We use BPE tokenization with a vocabulary of size 50256. All the models are trained on sequence length of 2048 tokens. Note that our samples are of size 2000 tokens. We leave 48 tokens for adding either raw scores for MEDA or instructions for INST. Note that the baseline is also trained on same samples of 2000 tokens. We pad the extra spaces with a PAD_TOKEN.

Model	**96m-samples**					**150m-samples**				
	EMT	TP	NLP	BD	E2E	EMT	TP	NLP	BD	E2E
BASE	0.44	0.37	52.6	53.0	30.7	0.44	0.37	54.4	53.8	31.1
FILT	0.40	0.30	52.9	55.9	29.9	0.41	0.31	54.5	53.2	31.9
	↓8.5%	↓18.8%	↑0.6%	↑5.5%	↓2.5%	↓7.4%	↓16.5%	↑0.2%	↓1.1%	↑2.6%
MEDA	0.42	0.33	53.0	57.2	31.8	0.42	0.34	53.9	53.5	33.0
	↓4.7%	↓10.7%	↑0.8%	↑7.9%	↑3.7%	↓4.2%	↓9.1%	↓0.9%	↓0.6%	↑6.1%
INST	0.43	0.34	53.3	53.9	30.6	0.42	0.34	53.7	54.9	31.7
	↓3.6%	↓9.7%	↑1.3%	↑1.7%	↓0.2%	↓3.7%	↓9.2%	↓1.3%	↑2.0%	↑2.1%
Experiment using control variable $C_{\texttt{nont}}$										
MEDA	0.32	0.16	-	-	31.9	0.32	0.16	-	-	33.6
	↓27.5%	↓58.1%			↑4.2%	↓26.8%	↓57.4%			↑8.2%
INST	0.31	0.15	-	-	31.3	0.31	0.14	-	-	32.6
	↓30.2%	↓62.7%			↑2.1%	↓30.0%	↓63.5%			↑4.9%

Table 3: Results for 1.3b parameter models. EMT is Expected Maximum Toxicity; TP is Toxicity Probability; NLP indicates the average of accuracy on five benchmark NLP tasks; BD displays the average AUC on four bias detection tasks; and E2E shows the Rouge-L scores of the LMs on the E2E task. For benchmark NLP tasks, bias detection tasks and E2E task we show the relative percentage improvement over BASE with a ↑% and decrement with a ↓%. For the expected maximum toxicity and toxicity probability, we show the improvement with ↓% because lower is better for these metrics. We may observe that two strategies obtain the exact same score but there is a difference in their relative percentages. This is because these scores are computed up to 4 decimal digits but we only report scores up to 2 decimals here.

357m parameter models We train them with 24 layers, with a hidden size of 1024 and 16 attention heads. We use $\texttt{max} - \texttt{position} - \texttt{embeddings}$ of 2048; 162761 warmup samples; a learning rate of $3.0e^{-4}$ with minimum learning rate of $3.0e^{-5}$. We use cosine learning decay style. Additionally, we use clip-grad = 1.0, weight-decay = 0.1, adam-beta1 = 0.9, and adam-beta2 = 0.95. Each of these models are trained on 64 A100 GPUs with 40GB memory. The models with 96m samples are trained for 54 GPU hours and models with 150m samples are trained for 84 GPU hours.

1.3b parameter models We train them with 24 layers, with a hidden size of 2048 and 32 attention heads. We use max-position-embeddings = 2048 with 244141 warmup samples; a learning rate of $2.0e^{-4}$ with a minumum learning rate of $2.0e^{-5}$ and cosine decay style. We use clip-grad = 1.0, weight-decay = 0.1, adam-beta1 = 0.9, and adam-beta2 = 0.95. Each of these models are trained on 64 A100 GPUs with 40GB memory. The models with 96m samples are trained for 113 GPU hours and models with 150m samples are trained for 176 GPU hours.

Bounds for the new variables We describe the bounds for LOWTHRESH, HIGHTHRESH, PRMTOX and PRMNONT variables introduced in this work.

$$\begin{aligned} 0 < \textsc{LowThresh} < 1 \\ 0 < \textsc{HighThresh} < 1 \\ 0 \leq \textsc{PrmTox} \leq 1 \\ 0 \leq \textsc{PrmNonT} \leq 1 \end{aligned}$$

Note that PRMTOX = 0 means that no samples above the HIGHTHRESH are augmented with $C_{\texttt{tox}}$; and PRMTOX = 1 means that all the samples above the HIGHTHRESH are augmented with $C_{\texttt{tox}}$. Similarly, PRMNONT = 0 implies that no samples below LOWTHRESH are modified; and PRMNONT = 1 means that all the samples below LOWTHRESH are augmented with $C_{\texttt{nont}}$.

Number of Shots for Tasks Table 4 shows the number of shots used as context for each task following the setups in Brown et al. (2020); Smith et al. (2022); Prabhumoye et al. (2021b).

Task	**# of Shots**
LAMBADA (Paperno et al., 2016)	15
ANLI (Nie et al., 2020)	50
Winogrande (Sakaguchi et al., 2020)	50
PiQA (Bisk et al., 2020)	50
Hellaswag (Zellers et al., 2019a)	20
Bias Detection (Prabhumoye et al., 2021b)	32

Table 4: Number of shots used as context for each task.

Model	**150m-samples**			
	EMT	TP	NLP	BD
BASE	0.43	0.35	64.9	54.7
FILT	0.39	0.28	64.8	61.0
	↓8.0%	↓19.6%	↓0.1%	↑11.5%
INST	0.41	0.30	65.3	61.7
	↓4.5%	↓13.4%	↑0.7%	↑12.7%
Experiment using control variable $C_{\texttt{nont}}$				
INST	0.28	0.11	-	-
	↓34.0%	↓69.5%		

Table 5: Results for 8.3b parameter models trained with 150 million samples.

Model	**96m-samples**				
	EMT	TP	NLP	BD	E2E
BASE	0.44	0.36	47.5	50.6	27.6
FILT	0.40	0.29	48.0	51.2	27.4
	↓8.1%	↓18.5%	↑1.1%	↑1.2%	↓0.8%
FILT-0.4	0.38	0.25	47.4	50.0	28.8
	↓13.1%	↓30.8%	↓0.3%	↓1.1%	↑4.3%
FILT-0.35	0.37	0.23	48.0	50.4	28.4
	↓15.1%	↓36.8%	↑1.1%	↓0.4%	↑2.9%
INST	0.42	0.33	47.9	50.2	28.9
	↓2.8%	↓6.8%	↑0.8%	↓0.9%	↑4.9%
Experiment using control variable $C_{\texttt{nont}}$					
INST	0.31	0.15	-	-	29.8
	↓29.0%	↓59.3%			↑7.8%

Table 6: Results for 357m-96m configuration on all the metrics for variations of FILT such as FILT-0.4 and FILT-0.35 in comparison with INST.

D Ablation Experiments

Scaling the Model Size Table 5 shows results for 8.3 billion parameter models which use BASE, FILT and INST strategies on 357m-150m model configuration.

FILT Variations Table 6 shows the results for BASE, FILT, FILT-0.4, FILT-0.35 and INST for all the 357m-96m model configuration on all eleven tasks. These results are aggregated and presented in Fig. 9.

MEDA Variations Table 7 shows the results for BASE, MEDA-11, MEDA, MEDA-90 and INST for 357m-96m model configuration on all eleven tasks. These results are aggregated and presented in Fig. 10.

INST Variations Table 8 shows the results for BASE, INST-11, INST-50 and INST for all the four model configuration on all eleven tasks. These results are aggregated and presented in Fig. 8.

Model	**96m-samples**				
	EMT	TP	NLP	BD	E2E
BASE	0.44	0.36	47.5	50.6	27.6
MEDA-11	0.42	0.33	47.1	52.6	28.4
	↓4.6%	↓8.8%	↓0.8%	↑4.1%	↑3.0%
MEDA	0.41	0.31	48.1	50.1	28.5
	↓5.9%	↓13.2%	↑1.4%	↓1.0%	↑3.2%
MEDA-90	0.42	0.33	47.0	47.6	28.6
	↓2.9%	↓8.2%	↓1.1%	↓6%	↑3.8%
INST	0.42	0.33	47.9	50.2	28.9
	↓2.8%	↓6.8%	↑0.8%	↓0.9%	↑4.9%
Experiment using control variable $C_{\texttt{nont}}$					
MEDA-11	0.35	0.20	-	-	28.4
	↓20.7%	↓43.5%			↑3.0%
MEDA	0.33	0.18	-	-	28.3
	↓24.0%	↓49.8%			↑2.6%
MEDA-90	0.31	0.14	-	-	28.4
	↓28.6%	↓59.9%			↑3.0%
INST	0.31	0.15	-	-	29.8
	↓29.0%	↓59.3%			↑7.8%

Table 7: Results for 357m-96m configuration on all the metrics for MEDA and INST in comparison with MEDA-11 and MEDA-90.

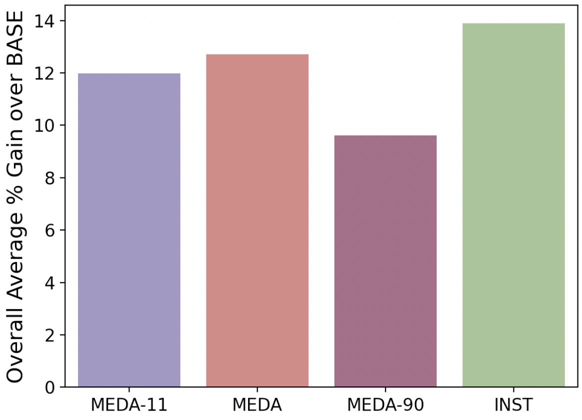


Figure 10: Average the gains achieved by MEDA-11, MEDA, MEDA-90, and INST over BASE across the eleven tasks for 357m-96m model configuration.

Model	**96m-samples**					**150m-samples**				
	EMT	TP	NLP	BD	E2E	EMT	TP	NLP	BD	E2E
Experiments with 357m parameter models										
BASE	0.44	0.36	47.5	50.6	27.6	0.43	0.35	48.2	50.0	30.8
INST-11	0.41	0.32	46.6	50.9	28.1	0.42	0.34	48.7	50.7	28.7
	↓5.9%	↓11.8%	↓1.8%	↑0.5%	↑1.9%	↓2.0%	↓3.9%	↑1.0%	↑1.4%	↓6.8%
INST-50	0.41	0.32	47.9	49.9	28.2	0.42	0.33	48.3	49.1	29.5
	↓5.6%	↓11.9%	↑0.9%	↓1.5%	↑2.3%	↓3.0%	↓6.8%	↑0.2%	↓1.8%	↓4.3%
INST	0.42	0.33	47.9	50.2	28.9	0.42	0.33	48.7	51.1	29.7
	↓2.8%	↓6.8%	↑0.8%	↓0.9%	↑4.9%	↓1.9%	↓5.4%	↑0.9%	↑2.3%	↓3.7%
Experiment using control variable $C_{\texttt{nont}}$ for 357m parameter models										
INST-11	0.36	0.23	-	-	28.3	0.38	0.25	-	-	29.0
	↓18.5%	↓36.8%			↑2.7%	↓13.0%	↓27.4%			↓6.0%
INST-50	0.32	0.16	-	-	28.0	0.34	0.18	-	-	29.5
	↓26.7%	↓54.6%			↑1.5%	↓22.3%	↓48.2%			↓4.4%
INST	0.31	0.15	-	-	29.8	0.31	0.14	-	-	29.7
	↓29.0%	↓59.3%			↑7.8%	↓28.1%	↓59.7%			↓3.5%
Experiments with 1.3b parameter model										
BASE	0.44	0.37	52.6	53.0	30.7	0.44	0.37	54.4	53.8	31.1
INST-11	0.41	0.32	52.9	54.1	33.5	0.42	0.33	54.3	54.0	31.0
	↓6.3%	↓13.3%	↑0.5%	↑2.2%	↑9.1%	↓4.9%	↓11.0%	↓0.1%	↑0.3%	↓0.3%
INST-50	0.41	0.32	53.4	54.5	30.8	0.42	0.34	53.9	54.6	32.6
	↓6.3%	↓14.3%	↑1.4%	↑2.9%	↑0.4%	↓3.7%	↓8.4%	↓0.8%	↑1.4%	↑4.9%
INST	0.43	0.34	53.3	53.9	30.6	0.42	0.34	53.7	54.9	31.7
	↓3.6%	↓9.7%	↑1.3%	↑1.7%	↓0.2%	↓3.7%	↓9.2%	↓1.3%	↑2.0%	↑2.1%
Experiment using control variable $C_{\texttt{nont}}$ for 1.3b parameter models										
INST-11	0.32	0.15	-	-	34.2	0.33	0.18	-	-	32.1
	↓28.3%	↓59.6%			↑11.7%	↓24.3%	↓51.4%			↑3.3%
INST-50	0.31	0.14	-	-	30.9	0.32	0.15	-	-	32.9
	↓29.6%	↓61.4%			↑0.7%	↓27.3%	↓58.4%			↑6.0%
INST	0.31	0.15	-	-	31.3	0.31	0.14	-	-	32.6
	↓30.2%	↓62.7%			↑2.1%	↓30.0%	↓63.5%			↑4.9%

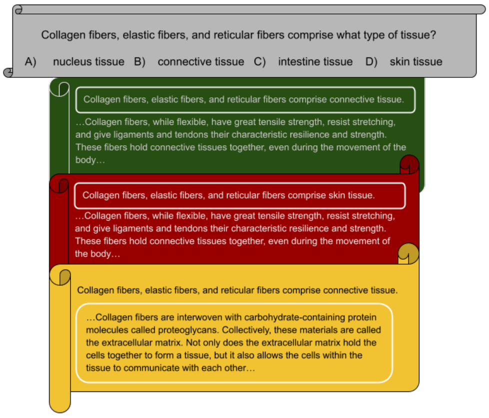
Table 8: Results for 357m and 1.3b parameter models on all the metrics for INST and its variations INST-11 and INST-50. For benchmark NLP tasks, bias detection tasks and E2E task we show the relative percentage improvement over BASE with a ↑% and decrement with a ↓%. For the expected maximum toxicity and toxicity probability, we show the improvement with ↓% because lower is better for these metrics.

Multi2Claim: Generating Scientific Claims from Multi-Choice Questions for Scientific Fact-Checking

Neşet Özkan TAN Trung Nguyen Joshua Bensemann Alex Yuxuan Peng
Qiming Bao Yang Chen Mark Gahegan Michael Witbrock
The School of Computer Science
University of Auckland
Auckland, New Zealand
`{neset.tan, trung.nguyen, josh.bensemann, alex.peng}@auckland.ac.nz`
`{qiming.bao, yang.chen, m.gahegan, m.witbrock}@auckland.ac.nz`

Abstract

In automated scientific fact-checking, machine learning models are trained to verify scientific claims given evidence. A major bottleneck of this task is the availability of large-scale training datasets on different domains, due to the required domain expertise for data annotation. However, multiple-choice question-answering datasets are readily available across many different domains, thanks to the modern online education and assessment systems. As one of the first steps towards addressing the fact-checking dataset scarcity problem in scientific domains, we propose a pipeline for automatically converting multiple-choice questions into fact-checking data, which we call **Multi2Claim**. By applying the proposed pipeline, we generated two large-scale datasets for scientific-fact-checking: **Med-Fact** and **Gsci-Fact** for the medical and general science domains, respectively. These two datasets are among the first examples of large-scale scientific-fact-checking datasets. We developed baseline models for the verdict prediction task using each dataset. Additionally, we demonstrated that the datasets could be used to improve performance measured by weighted $F1$ on existing fact-checking datasets such as SciFact, HEALTHVER, COVID-Fact, and CLIMATE-FEVER. In some cases, the improvement in performance was up to a 26% increase. The generated datasets are publicly available[1].


Collagen fibers, elastic fibers, and reticular fibers comprise what type of tissue?

A) nucleus tissue B) connective tissue C) intestine tissue D) skin tissue

Collagen fibers, elastic fibers, and reticular fibers comprise connective tissue.

…Collagen fibers, while flexible, have great tensile strength, resist stretching, and give ligaments and tendons their characteristic resilience and strength. These fibers hold connective tissues together, even during the movement of the body…

Collagen fibers, elastic fibers, and reticular fibers comprise skin tissue.

…Collagen fibers, while flexible, have great tensile strength, resist stretching, and give ligaments and tendons their characteristic resilience and strength. These fibers hold connective tissues together, even during the movement of the body…

Collagen fibers, elastic fibers, and reticular fibers comprise connective tissue.

…Collagen fibers are interwoven with carbohydrate-containing protein molecules called proteoglycans. Collectively, these materials are called the extracellular matrix. Not only does the extracellular matrix hold the cells together to form a tissue, but it also allows the cells within the tissue to communicate with each other…

Figure 1: Examples of “supported” (green), “refuted” (red), and “not-enough-info” (yellow) types of claims from the Gsci-Fact dataset which is generated from the original multi-choice question (grey).

1 Introduction

Learning to verify the claims in scientific papers and “science releases” (media announcements of scientific findings) is a difficult task for both artificial intelligence (AI) systems and humans. However, this task is crucial because learning to separate verified facts from speculation or falsehoods has important consequences. Success at this task can help the reader understand scientific topics and promote science. Conversely, failure at this task leads to the spread of misinformation and exaggeration, which can cause distortion in scientific communication and undermine public confidence in science. Unfortunately, several studies have revealed that science releases and scientific articles can contain significant exaggeration and misinformation (Woloshin et al., 2009), (West and Bergstrom, 2021), (Sumner et al., 2014), (Woloshin and Schwartz, 2002). This misleading information can directly impact people’s lives, as was the case for media releases concerning the COVID-19 pandemic (Roozenbeek et al., 2020). In 2014, some estimates claimed that 40% of the press releases contained exaggerated advice and 33% contained exaggerated claims in science-related news (Sumner et al., 2014). Considering that this problem has not disappeared over the last eight years and has possibly been exasperated by the continued growth of social media and the number of published scientific papers, there is a need for automated systems that can aid both academics and the public in judging the veracity and credibility of scientific claims. In this context, attempts to automate searching for distortions of findings, exaggerations, and misrepresentations

---

[1] https://github.com/taneset/Multi2Claim.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2652–2664
May 2-6, 2023

may contribute to the trustworthiness verification of science releases and articles.

The fact verification task, commonly known as "fact-checking", is verifying claims in natural language against a collection of information that contains facts. The pipeline for fact-checking usually includes two subtasks: 1) Retrieve-Rank and 2) Veracity Prediction. The Retrieve-Rank operation is typically carried out using information retrieval, and ranking models that rely on a combination of lexical (BM25) and semantic (using word embeddings in pre-trained language models) similarities between the claim and textual evidence candidates (Lin et al., 2020). The veracity prediction task, which has been studied less than the retrieve-rank task, can be seen as a classification task that predicts the type of verdict given a claim-evidence pair. Typically there are three classes of verdicts: "support", "refute", and "not enough information". The veracity prediction problem can also be formulated as a task of identifying textual entailment, in which a model predicts if the provided evidence entails a given claim. Recent work shows that these two subtasks could be combined in an end-to-end manner (Thorne et al., 2021; Wadden et al., 2020a) or could be carried out separately (Saakyan et al., 2021; Diggelmann et al., 2020). Most studies and datasets used in fact-checking are designed to verify claims in general domains such as news, forums, popular Wikipedia passages, and social media posts (Augenstein et al., 2019b; Thorne et al., 2021; Oshikawa et al., 2020; Shahi and Nandini, 2020; Shaar et al., 2020).

It is typically more challenging to automatically verify a scientific claim compared to a claim in the general domain. This is because scientific assertions can be much more complex, and it requires deep domain knowledge to create datasets for scientific claims. The required domain knowledge is a major bottleneck, making the annotation process expensive and time-consuming. As a result of these difficulties, there are only a few scientific-fact-checking datasets in the literature, and the sizes of those datasets are limited. However, the recent deep-learning-based approaches to performing fact-checking require large amounts of annotated training data to generalise well on unseen data. Therefore, there is an urgent need for large-scale scientific-fact-checking datasets, and methods for automatically creating such datasets.

As a step towards addressing the problem of lacking large-scale datasets in scientific-fact-checking, this paper makes the following contributions:

- We constructed a pipeline for generating scientific claims from scientific multiple-choice questions.
- We created two large-scale scientific-fact-checking datasets in the biomedical (150k samples) and general-science domains (32k samples) by applying the proposed pipeline to existing scientific multiple-choice question-answering (QA) datasets.
- We evaluated different pretrained transformer-based models for the verdict prediction task on the generated datasets. The results serve as the initial benchmark on the datasets.
- We showed that the generated datasets can be used to improve the performance on existing scientific-fact-checking datasets.

2 Claim Generation From Multi-Choice Questions

Claim generation can be defined as the process of generating claims that can be classified as "supported", "refuted", or "not enough information" based on evidence in associated texts. A multiple-choice question typically consists of a question, a correct option, and multiple distractors. Some multiple-choice QA datasets even provide an explanation of the correct answer. This section describes a pipeline to automatically generate all three types of claims that are commonly found in the existing fact-checking datasets by taking advantage of such multiple-choice QA datasets. The pseudocode of the generation process is described in Algorithm 1.

2.1 Supported and Refuted Claim Generation

The key to our method of generating supported and refuted claims is a sequence-to-sequence model that can convert question-answer pairs into their declarative forms. For example, the question-answer pair ("Which of the following hormonal activity is expected immediately prior to ovulation?", "LH surge") might be converted into a declarative sentence such as, "LH surge is expected immediately prior to ovulation". In order to achieve this, we adopted the BART (Lewis et al., 2019) model that was trained to convert question-answer pairs in the Stanford Question Answering Dataset

(SQUAD) (Rajpurkar et al., 2016) into their declarative forms, known as QA2D (Demszky et al., 2018). We denote this model as $BART_{QA2D}$.

Our pipeline begins with multiple-choice questions. These questions are typically prepared by domain experts who determine which pieces of knowledge are essential to test learners on within the related field. To create a "supported" claim from a multiple-choice question, we feed the question and the answer (correct option) as input to $BART_{QA2D}$ to generate the declarative sentence. Using this process, we obtain the same number of claims as the number of questions in the multiple-choice QA dataset. The generated claim is then paired with the original explanation of the correct answer, serving as the supporting evidence.

To generate a "refuted" claim type, we make use of the original distractors, carefully selected by domain experts. We assume that any incorrect option (distractor) should be refuted by the original supportive document (explanation), since there is a single correct option for each question. In our LH surge example, the distractors are FSH surge, Progesterone surge, and Estrogen surge. One can generate as many different refuted claims as the number of distractors from a typical multiple-choice question. However, in our implementation, we only generated one refuted claim from a question using the distractor that was the most similar to the correct option. To achieve this, we computed the cosine similarity scores between the embeddings of the correct choice and each of the distractor choices, and then we chose the distractor with the highest score. The embeddings were computed using the Scispacy named-entity-recognition model[2] (Neumann et al., 2019), which was trained on biomedical corpora such as the MedMentions (Murty et al., 2018), and the BioCreative V CDR corpus[3]. This filtering of distractors can help remove distractors that are dissimilar to the correct option and therefore avoid generating claims that can be obviously refuted. The chosen distractor is then fed into the $BART_{QA2D}$ model along with its question to generate a declarative sentence, e.g., "FSH surge is expected immediately prior to ovulation". This generated claim is then paired with the original explanation to the correct answer.

We want to emphasise that generating "refuted" claims is challenging and might require extra ontology-like mechanisms to replace plausible but false notions or entities in order to declare the claim untrue. However, an ontology-like approach requires extensive filtering to ensure the associated document (explanation) is not supporting the replacement and the replacement is meaningfully integrated into the refuted claim. An alternative way of generating a refuted claim is by simply adding "not" to a supported claim. However, exclusively using this method would result in all refuted claims containing "not". This would limit the diversity of the generated claims and might cause the machine learning models to cheat by simply identifying negation.

Algorithm 1 Claim generation from multi-choice question

```
Require: Question ($Q$), Explanation ($E$), Answer ($A$), Distractors ($D$)
function SUPPORTED($q \in Q, e \in E, a \in A$)    ▷ $q, e$ and $a$ belong to the same question.
    $c \leftarrow BART_{QA2D}(q, a)$
    return $(c, e)$    ▷ The evidence $e$ supports claim $c$.
end function
function REFUTED($q \in Q, e \in E, a \in A, D$)    ▷ $q, e, a$ and $D$ belong to the same question.
    $\hat{d} \leftarrow \arg\max_{d_i \in D} cosine(a, d_i)$
    $c \leftarrow BART_{QA2D}(q, \hat{d})$
    return $(c, e)$    ▷ Claim $c$ is refuted by $e$.
end function
function NOT_ENOUGH_INFO($c, e \in E, a \in A, E, A$)    ▷ $c$ is generated using SUPPORTED. $e$ and $a$ are its associated explanation and answer.
    $\hat{E} \leftarrow \arg\mathrm{top10}_{e_i \in E, a_i \in A}\, cosine(SPECTER(a, e), SPECTER(a_i, e_i))$    ▷ $\hat{E}$ is sorted in descent.
    for $\hat{e}_j$ in $\hat{E}$ do
        if $\hat{e}_j \not\supseteq a$ then
            return $(c, \hat{e}_j)$    ▷ $\hat{e}_j$ does not contain enough information to make a judgement on $c$.
        end if
    end for
end function
```

2.2 Not-Enough-Info Claim Generation

To generate a not-enough-info claim, we replace the explanation of a supported claim with a similar explanation from another claim but without sharing the same key entities or notions (the answer). An

---

[2] https://allenai.github.io/scispacy

[3] https://www.ncbi.nlm.nih.gov/research/bionlp/Data

example of this is shown in figure 1, where the original supportive document is about collagen fibers and connective tissue, and the replacement is about collagen fibers and extracellular matrix. Crucially, the replacement does not contain information about connective tissue. To help find similar explanations as the one provided, we compute document-level representations of concatenated answer and explanation for all generated claims using SPECTER as introduced in (Cohan et al., 2020). It was shown that SPECTER can create a dense-vector representation for each scientific document in order to capture the relatedness of the documents (Cohan et al., 2020). By using the dense-vector representations of explanations, we retrieve 10 most similar explanations (using cosine similarity) for each claim. Then, we filter out those that contain the key entity of the claim, and we selected the most similar one among them.

2.3 Med-Fact: Medical domain fact-checking dataset

We applied the proposed pipeline to the MedMCQA dataset (Pal et al., 2022) which consists of real-world medical entrance exam questions, answers (which could be multiple or single), and the supporting document for the correct answer. We selected samples whose supporting documents had more than 50 words to ensure a minimal length of the supporting document. Then we only considered multiple-choice questions that had a single correct option. We also dropped the questions, which had the same supportive documents. In the end, we generated 150K claims, including 50K supported, 50K refuted, and 50K not-enough-info claims. Examples of this dataset are given at the end of the Appendix.

2.4 Gsci-Fact: General science domain fact-checking dataset

We used about 13.7K multiple-choice science-exam questions introduced in (Welbl et al., 2017). These questions are about natural sciences such as biology, physics, and chemistry and are created by crowd workers. As was true for Med-Fact, we applied a filtering process to ensure the length of the supporting document and to have a unique supporting document. We generated about 32.2K claims, of which 10.7K are supported, 10.7K are refuted, and 10.7K are not-enough-info. An example is given in Figure 1.

3 Experiments

We conducted experiments to answer the following research questions:

- **Whether the datasets we generated can be used for verdict prediction tasks?**
- **Can the generated datasets improve the models' performance on scientific-fact-checking tasks?**
- **What is the quality of the generated claims?**

We formulate verdict prediction as a multi-class classification task. For a given claim c and a document d, the model must determine a label

$$l(c,d) \in \{\text{supported, refuted, not-enough-info}\}.$$

We concatenate a claim and its document (explanation) together as input, and the model is trained to predict the claim type (supported, refuted, or not-enough-info) in a supervised manner.

3.1 Baselines for Med-Fact and Gsci-Fact

We selected five pre-trained models from the literature as baselines for fact-checking tasks. We fine-tuned the transformer models BERT (Devlin et al., 2018), DeBERTa (He et al., 2020), SciBERT (Beltagy et al., 2019), Longformer (Beltagy et al., 2020), and BioBERT (Lee et al., 2019) for the verdict prediction task. DeBERTa, SciBERT, and BioBERT are descendants of BERT, and DeBERTa has modified attention mechanisms. SciBERT was trained on a large multi-domain corpus of scientific publications, whereas BioBERT was trained on a large-scale biomedical corpus. Longformer is different from the other transformer-based models because it has an efficient attention mechanism that accepts longer input sizes. We used the weighted F_1 metric for evaluation because models can be accurate at predicting a specific label but inaccurate at others, and weighted F_1 can give better insight about performance than accuracy. The weighted-F_1 score is calculated by averaging all per-class F_1 scores while accounting for support for each class, where support refers to the number of actual class occurrences in the dataset. For both Med-Fact and Gsci-Fact datasets, DeBERTa produced the best performance. The complete results for Med-Fact and Gsci-Fact are shown in Table 1.

.

Models	Med-fact	GSci-Fact
BERT	0.70	0.85
DeBERTa	**0.77**	**0.90**
Longformer	0.72	0.86
BioBERT	0.71	0.86
SciBERT	0.65	0.78

Table 1: Weighted F_1 scores of baseline models on verdict prediction task.

3.2 Performance Improvements on Exiting Datasets

We examined whether training models on Med-Fact and Gsci-Fact can improve the performance of verdict prediction on the existing scientific fact-checking datasets. We used two different setups for our experiments. Firstly, we fine-tuned and evaluated models presented in Table 1 on SciFact, HEALTHVER, and CLIMATE-FEVER with three classes (supported, refuted, and not-enough-info). The results are shown in Table 3. We conducted additional experiments to investigate whether our generated fact-checking datasets can improve performance on binary-classification fact-checking datasets such as COVID-Fact. The results for the COVID-Fact dataset are shown in Table 2.

3.2.1 SciFact

According to (Wadden et al., 2020b), the SciFact dataset consists of 1.4K expert-written scientific claims with associated documents (abstracts of the scientific articles) that contain evidence about the claim. The dataset is in the biomedical domain and, is extracted from S2ORC (Lo et al., 2020). The document length is considerably longer than documents in other scientific-fact-checking datasets since it contains the abstracts of scientific papers. Statistics for all the datasets we used in this study are provided in Table 5 in the Appendix.

Although the SciFact dataset is designed for both retrieval (both sentence level and abstract level) and verdict prediction tasks, we only conducted experiments on the verdict prediction task. This set up exists in the literature (Saakyan et al., 2021; Wright et al., 2022). To do that, we used all the associated documents in the datasets for each claim. Since the test set of the dataset is not publicly available, we merged the training and development sets and reserved 10% as a test set for the experiments. We obtained the best weighted F_1 score (0.77) on SciFact using DeBERTa. By fine-tuning models trained on Med-Fact and Gsci-Fact, we improved the weighted F_1 score to 0.86 and 0.83, respectively.

3.2.2 HEALTHVER

HEALTHVER contains health-related claims obtained from sources such as online forums and search engines (Sarrouti et al., 2021). To verify the claims, the top-10 related abstracts were labeled by annotators as “supports”, “refutes”, “neutral”. As with the experiments on SciFact, we used 10% of the dataset as the test set. We obtained results ranging from 0.68 to 0.78 by fine-tuning five pre-trained models on the HEALTHVER dataset. Training first on our Med-Fact and Gsci-Fact datasets consistently increased all models’ performance by 0.16 to 0.20 points.

3.2.3 CLIMATE-FEVER

CLIMATE-FEVER consists of claims related to climate change. Like HEALTHVER, they use techniques such as web scraping and using keywords in search engines (Diggelmann et al., 2020). They treated the verdict prediction task as an entailment prediction task to predict one of the labels “SUPPORTS”, “REFUTES”, or “NOT ENOUGH INFO”. During our experiments, we obtained improvement of 0.11 points on the weighted F_1 score using the Med-Fact dataset in the best case. However, The improvements were generally less than that obtained on the previous two datasets. One possible explanation is that the Med-Fact and Gsci-Fact datasets are dominated by biomedical and natural science topics that are not as related to the climate-change domain as the health and medical subject-dominated datasets.

3.2.4 COVID-Fact

We also examined the effect of transferring models trained on the generated dataset to a binary-classification fact-checking dataset such as COVID-Fact. In the COVID-Fact dataset, a claim can either be “supported” or “refuted”. These claims have been scrapped from COVID discussions made in online forums such as Reddit. Five pieces of evidence for each claim were collected from Google search results using a cleaning process. We adopted BERT and DeBERTa models in the experiments. We again observed improvement even though the Med-Fact dataset contains many scientific terminologies and jargon, which one cannot expect from COVID-Fact due to its creation process.

Models	BERT	DeBERTa
COVID-Fact	0.60	0.85
Med-Fact + COVID-Fact	0.68^{+8}	0.89^{+4}

Table 2: Comparison of models' performance (Weighted F_1 score) on COVID-Fact.

3.3 Claim Evaluation

We investigated the quality of the generated claims by asking human annotators to evaluate claims from four perspectives: fluency, contextually, faithfulness, and challenge level. We asked six annotators to manually evaluate the claims by following a guideline inspired by (Kuhn et al., 2013; Wright et al., 2022). Table 6 in the Appendix contains information about the guideline, such as definitions of these perspectives and associated scores.

The annotators consisted of Ph.D. students and Ph.D. graduates in the fields of science, medicine, psychology, and computer science. A random sample of 300 examples was taken, with 150 from the Gsci-Fact corpus and 150 from the Med-Fact corpus, ensuring an equal representation of the three claim types: supported, refuted, and not-enough-info. To minimize annotation costs while still leveraging the expertise of each annotator, the sample was divided among eight experts, who were tasked with evaluating the fluency, contextuality, and challenge level of the generated claims. The final scores were obtained by averaging the annotations of all experts on the related dataset.

For the fluency, the annotators found that 98% of the generated claims have no grammatical errors and are clearly understandable for both Gsci-Fact and Med-Fact. The remaining 2% were deemed understandable despite a few grammatical errors and they were equally distributed among three labels for both dataset. The annotators found that 98% of the generated claims in Med-Fact are interpretable without additional context, while 96% are interpretable without additional context in Gsci-Fact. This minor difference could be due to the diverse background of the annotators who evaluated the Gsci-Fact dataset. 75% of the claims in the Gsci-Fact dataset were marked as cannot be answered without the evidence associated with the claims. However, 95% of the claims in the Med-Fact dataset were marked as not verifiable without the associated evidence.

When evaluating the alignment between the assigned labels generated during the generation progress and the faithfulness scores assigned by annotators during the annotation progress, a high degree of agreement was observed. Specifically, annotators concurred with 92% of the "supported" claims in the Gsci-Fact dataset, with corresponding agreement levels of 89% and 88% for the "refuted" and "not-enough-info" claims, respectively. Similar results were obtained for the Med-Fact dataset, with agreement levels of 93%, 91%, and 90% for the "supported", "refuted", and "not-enough-info" claims, respectively. Examples of each type of claim with their evaluation scores can be found in the Appendix, listed in Table 7 for a refuted claim, Table 8 for a supported claim, and Table 9 for a not-enough-info claim.

3.4 Further Analysis

It has been observed that general-domain natural-language-inference datasets can have a significant amount of bias (Poliak et al., 2018), and we wondered if Med-fact and Gsci-Fact contain such a bias. We investigated the claim-only bias because the aim of the fact-checking task is to evaluate the model's ability to examine the semantic relationship between a claim and the supporting data. We tested all the baseline models using just the claim as an input. We discovered that the model's performances (weighted F1 scores) dropped to at most 35%, which indicating that the label-associated bias is not presented. We interpret this result as that the domain-specific evidence associated with the claim is required to make the correct prediction.

4 Related Work

Recent work in automated fact-checking has made progress in the battle against the spread of false information in the news (Pomerleau and Rao, 2017), social media posts, online forums (Vlachos and Riedel, 2014; Mihaylova et al., 2018), and popular Wikipedia articles (Thorne et al., 2018). There has also been substantial work done that uses statements of fact-checking organizations (Augenstein et al., 2019a; Alhindi et al., 2018).

All the work mentioned so far has focused on claims and related documents from the general domain. However, the proposed models and datasets created can be difficult to apply to scientific domains because of the domain mismatch. Limited research has been conducted with datasets and models that focus on verifying claims made in scientific releases against scientific documents. For exam-

Model	SciFact	HEALTHVER	CLIMATE-FEVER
BERT	0.65	0.72	0.58
DeBERTa	0.77	0.78	0.64
Longformer	0.70	0.77	0.63
SciBERT	0.65	0.69	0.52
BioBERT	0.64	0.73	0.51
BERT$_{Med}$	0.78$^{\mathbf{+13}}$	0.91$^{\mathbf{+19}}$	0.62$^{\mathbf{+4}}$
DeBERTa$_{Med}$	0.86$^{\mathbf{+9}}$	0.94$^{\mathbf{+16}}$	0.75$^{\mathbf{+11}}$
Longformer$_{Med}$	0.76$^{\mathbf{+6}}$	0.93$^{\mathbf{+16}}$	0.73$^{\mathbf{+10}}$
SciBERT$_{Med}$	0.68$^{\mathbf{+3}}$	0.88$^{\mathbf{+19}}$	0.55$^{\mathbf{+3}}$
BioBERT$_{Med}$	0.76$^{\mathbf{+12}}$	0.92$^{\mathbf{+19}}$	0.59$^{\mathbf{+8}}$
BERT$_{Gsci}$	0.78$^{\mathbf{+13}}$	0.90$^{\mathbf{+18}}$	0.61$^{\mathbf{+3}}$
DeBERTa$_{Gsci}$	0.83$^{\mathbf{+6}}$	0.92$^{\mathbf{+14}}$	0.70$^{\mathbf{+6}}$
Longformer$_{Gsci}$	0.79$^{\mathbf{+9}}$	0.93$^{\mathbf{+16}}$	0.64$^{\mathbf{+1}}$
SciBERT$_{Gsci}$	0.65$^{\mathbf{0}}$	0.87$^{\mathbf{+18}}$	0.55$^{\mathbf{+3}}$
BioBERT$_{Gsci}$	0.70$^{\mathbf{+6}}$	0.90$^{\mathbf{+17}}$	0.52$^{\mathbf{+1}}$

Table 3: The results of transferring models trained on Med-Fact and Gsci-Fact to SciFact, HEALTHVER and CLIMATE-FEVER datasets. The first five rows show the baseline results (weighted F1) without the transfer. The second set of 5 rows shows the results and improvements from training on Med-Fact first. The last 5 rows show the results of models trained on Gsci-Fact first.

ple, in (Roozenbeek et al., 2020; Saakyan et al., 2021), claims about COVID-19 made on social media platforms were researched. The general health-related claims made in science releases were studied by (Sarrouti et al., 2021). Additionally, claims about climate change and retrieved evidence from Wikipedia were studied by (Diggelmann et al., 2020). However, these models and datasets were designed to verify claims written in less technical language from public science releases and media posts on platforms such as Reddit (Saakyan et al., 2021). We are aware of only one dataset (SciFact) that focuses on claims extracted from scientific articles against scientific documents (Wadden et al., 2020a).

In recent years, a variety of techniques have been developed to enhance fact-checking abilities. One such model is based on multi-layer perceptrons (MLP) (Riedel et al., 2017), and another is based on attention mechanisms (Parikh et al., 2016). Both models were used as baselines in claim verification on the FEVER dataset (Thorne et al., 2018). Additionally, a Graph Neural Network (GNN) based approach (Liu et al., 2020)) was used for propagating nodes represented by evidence (Ye et al., 2020). Semantic role labeling and logical reasoning tools can also improve GNN-based approaches (Chen et al., 2020). In (Zhou et al., 2019), a graph-based evidence aggregating and reasoning (GEAR) framework was employed to aggregate multi-evidence data.

Recently, transformer-based language models have produced the best performance on fact-checking in general and scientific domains. These claim-verification models usually take concatenated claim and evidence pairs and process them with multi-layer transformer-based models to obtain representations for classifying relationships between the claim and the evidence. Pre-trained BERT models have often been used for classification (supported, refuted, and not enough info). For claim verification, BERT-based models are prevalent (Soleimani et al., 2019; Portelli et al., 2020; Chernyavskiy and Ilvovsky, 2019; Nie et al., 2019; Tokala et al., 2019), while Longformer has been used for verdict prediction (Wadden et al., 2020b; Wright et al., 2022).

Advancements in pre-trained transformer-based sequence-to-sequence language models, such as T5 (Raffel et al., 2020) and BART (Lewis et al., 2019)), have allowed researchers to create fully automatic pipelines for claim generation. In particular, negation generation and explanation generation of claims have been studied in the general-domain fact-checking task (Kotonya and Toni, 2020; Thorne et al., 2021). Entity-centric approach (Pan et al., 2021) uses entities in the scientific text to generate claims and has been applied to scientific claim generation by (Wright et al., 2022).

5 Discussion

Because of the annotation cost, most large-scale fact-checking datasets are synthetically generated. For example, the popular fact-checking dataset, Fever (Thorne et al., 2018), contains a limited number of claims annotated by humans. The rest of the claims were synthetically augmented using paraphrasing, negating, and substituting the original claims. However, such methods have been found to weaken results in several studies since real-world claims have different structures than the synthetically augmented claims (Augenstein et al., 2019a; Sarrouti et al., 2021). When generating a refuted claim, simply adding "not" to a supported claim does not always reflect the structure of real-world refuted claims. In our approach, we use the expert-selected distractors to generate refuted claims. The quantity of refuted claims is another weak point of the existing datasets. The amount of "refuted" claims is dramatically less than that of the "supported" claims in the existing datasets. This results in unbalanced fact-checking datasets (see the Table 5 in the Appendix). Both Med-Fact and Gsci-Fact have a balanced class distribution containing the same number of supported and refuted claims.

It is worth mentioning that the supporting documents for the claims that cannot be judged by a given document have been left as empty in (Thorne et al., 2018) and fact-checking datasets that are designed for binary labels ("supported" and "refuted") (Saakyan et al., 2021). In our proposed methods, we were able to retrieve related documents by using their dense-vector representations and consider them as documents for "not-enough-info" type of claims.

Finally, we want to discuss the entity-centric claim generation process (Pan et al., 2021; Wright et al., 2022). The first step of this process is selecting entities in a text to generate claims from them. These entities become the main objects of the generated claims. The weakness of this approach, according to (Pan et al., 2021) is that the generated claims can be superficial, and the verification of these claims can be done without the models' reasoning capabilities or knowledge of common sense. The degree of importance of these entities can differ for a scientific text that contains entities from multiple domains. Since there is no mechanism to indicate how important the selected entity is compared to the other entities in the given text, this approach might result in a poor selection of entities and, therefore, poor claim data. In our proposed method, using the multiple choices/entities of the question likely avoids that problem, because domain experts select the entities in the questions.

6 Conclusion

We have presented Multi2Claim, a pipeline that converts multiple-choice questions to fact-checking datasets. We specifically focus on challenging scientific domains, where the claim verification process can be complicated due to scientific jargon and complex assertions about fields. We presented two large-scale scientific fact-checking datasets created with this pipeline in biomedical (Med-Fact) and general science domains (Gsci-Fact). We testified these dataset for possible biases and the generated datasets were evaluated from various perspectives. Baseline models for these balanced large-scale scientific fact-checking datasets were also presented. We conducted extensive experiments to examine the benefits of the generated datasets on the existing scientific fact-checking dataset, which suffer from low numbers of samples and unbalanced labels. We consistently obtained improvements for all scientific fact-checking datasets, including binary-labeled datasets such as COVID-Fact.

We hope this work will lead to more breakthroughs in scientific fact-checking, which has received little attention due to the expensive and time-consuming annotation process that has to be done by domain experts. We also hope that the proposed pipeline and baseline models will help develop reliable models that will play an essential role in the scientific claim verification process.

Limitations

The proposed method has several limitations. Our claim generation pipeline relies on multiple-choice question-answering datasets since we limited ourselves to reliable and safe generation progress by using human-created scientific questions and answers. Another limitation is the lack of emphasis on retrieving explicit rationals and reasoning over the retrieved rationals due to the high cost of domain-specific rational annotation progress. For future work, we will extend this pipeline to generate claims from plain scientific texts with additional reasoning capabilities.

References

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. 2018. Where is your evidence: Improving fact-checking by justification modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90, Brussels, Belgium. Association for Computational Linguistics.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019a. MultiFC: A Real-World Multi-Domain dataset for Evidence-Based fact checking of claims. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4685–4697, Hong Kong, China. Association for Computational Linguistics.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019b. MultiFC: A Real-World Multi-Domain dataset for Evidence-Based fact checking of claims.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The Long-Document transformer.

Jiangjie Chen, Qiaoben Bao, Changzhi Sun, Xinbo Zhang, Jiaze Chen, Hao Zhou, Yanghua Xiao, and Lei Li. 2020. LOREN: Logic-Regularized reasoning for interpretable fact verification.

Anton Chernyavskiy and Dmitry Ilvovsky. 2019. Extract and aggregate: A novel Domain-Independent approach to factual data verification. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 69–78, Hong Kong, China. Association for Computational Linguistics.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *ACL*.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. CLIMATE-FEVER: A dataset for verification of Real-World climate claims.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: Decoding-enhanced BERT with disentangled attention.

Neema Kotonya and Francesca Toni. 2020. Explainable automated Fact-Checking: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5430–5443, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tobias Kuhn, Paolo Emilio Barbano, Mate Levente Nagy, and Michael Krauthammer. 2013. Broadening the scope of nanopublications. In *The Semantic Web: Semantics and Big Data*, pages 487–501. Springer Berlin Heidelberg.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence pre-training for natural language generation, translation, and comprehension.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained transformers for text ranking: BERT and beyond.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Tsvetomila Mihaylova, Preslav Nakov, Lluis Marquez, Alberto Barron-Cedeno, Mitra Mohtarami, Georgi Karadzhov, and James Glass. 2018. Fact checking in community forums.

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109, Melbourne, Australia. Association for Computational Linguistics.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. *AAAI*, 33(01):6859–6866.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2020. A survey on natural language processing for fake news detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6086–6093, Marseille, France. European Language Resources Association.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Zero-shot fact verification by claim generation.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Dean Pomerleau and Delip Rao. 2017. Fake news challenge.

Beatrice Portelli, Jason Zhao, Tal Schuster, Giuseppe Serra, and Enrico Santus. 2020. Distilling the evidence to augment fact verification models. In *Proceedings of the Third Workshop on Fact Extraction and VERification (FEVER)*, pages 47–51, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified Text-to-Text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text.

Benjamin Riedel, Isabelle Augenstein, Georgios P Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the fake news challenge stance detection task.

Jon Roozenbeek, Claudia R Schneider, Sarah Dryhurst, John Kerr, Alexandra LJ Freeman, Gabriel Recchia, Anne Marthe Van Der Bles, and Sander Van Der Linden. 2020. Susceptibility to misinformation about covid-19 around the world. *Royal Society open science*, 7(10):201199.

Arkadiy Saakyan, Tuhin Chakrabarty, and Smaranda Muresan. 2021. COVID-Fact: Fact extraction and verification of Real-World claims on COVID-19 pandemic.

Mourad Sarrouti, Asma Ben Abacha, Yassine Mrabet, and Dina Demner-Fushman. 2021. Evidence-based Fact-Checking of health-related claims. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3499–3512, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shaden Shaar, Giovanni Da San Martino, Nikolay Babulkov, and Preslav Nakov. 2020. That is a known lie: Detecting previously Fact-Checked claims.

Gautam Kishore Shahi and Durgesh Nandini. 2020. FakeCovid – a multilingual cross-domain fact check news dataset for COVID-19.

Amir Soleimani, Christof Monz, and Marcel Worring. 2019. BERT for evidence retrieval and claim verification.

Petroc Sumner, Solveiga Vivian-Griffiths, Jacky Boivin, Andy Williams, Christos A Venetis, Aimée Davies, Jack Ogden, Leanne Whelan, Bethan Hughes, Bethan Dalton, Fred Boy, and Christopher D Chambers. 2014. The association between exaggeration in health related science news and academic press releases: retrospective observational study. *BMJ*, 349:g7015.

James Thorne, Max Glockner, Gisela Vallejo, Andreas Vlachos, and Iryna Gurevych. 2021. Evidence-based verification for real world information needs.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Santosh Tokala, Vishal G, Avirup Saha, and Niloy Ganguly. 2019. AttentiveChecker: A Bi-Directional attention flow mechanism for fact verification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume*

1 (Long and Short Papers), pages 2218–2222, Minneapolis, Minnesota. Association for Computational Linguistics.

Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. pages 18–22.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020a. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020b. Fact or fiction: Verifying scientific claims.

Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jevin D West and Carl T Bergstrom. 2021. Misinformation in and about science. *Proc. Natl. Acad. Sci. U. S. A.*, 118(15).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Steven Woloshin and Lisa M Schwartz. 2002. Press releases: translating research into news. *JAMA*, 287(21):2856–2858.

Steven Woloshin, Lisa M Schwartz, Samuel L Casella, Abigail T Kennedy, and Robin J Larson. 2009. Press releases by academic medical centers: not so academic? *Ann. Intern. Med.*, 150(9):613–618.

Dustin Wright, David Wadden, Kyle Lo, Bailey Kuehl, Arman Cohan, Isabelle Augenstein, and Lucy Lu Wang. 2022. Generating scientific claims for Zero-Shot scientific fact checking.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential Reasoning Learning for Language Representation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7170–7186, Online. Association for Computational Linguistics.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. GEAR: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901, Florence, Italy.

Appendix

A Computing Sources and Experimental Setup

In all of our experiments, we used NVIDIA Quadro RTX 8000 graphics processing unit with 48GB of RAM capacity.

In the verdict prediction experiments, we used the model checkpoints that were provided by Huggingface (Wolf et al., 2019). BERT, SciBERT, and BioBERT have 12 layers and 12 transformer blocks in each layer, the size of the hidden layer is 768. The Longformer model has 12 layer with a hidden dimension of 512. DeBERTa model has a hidden size of 768 and 12 layers.

The exact number of model parameters that we used in this work is shown in the table 4.

Models	Parameters
BERT	109,484,547
Longformer	148,661,763
DeBERTa	184,424,451
SciBERT	109,920,771
BioBERT	108,312,579
BART	139,420,416

Table 4: Parameters per model

B Hyper-parameters

In the generation part, we used maximum 256 for the max length of the sequence to be generated. The number of highest probability vocabulary tokens to keep for top-k-filtering was 10. We returned one of the independently computed returned sequences for each element in the batch. We used the spaCy named entity recognition model from Scispacy to find similarities between the correct option and the other three options[4]. In the not enough info claim type generation, we used the 'allenai-specter'[5] SPECTER model's checkpoint, which was originally stored in the Allen AI repository. For all of the verdict prediction models, we used a variant of the Adam optimizer (AdamW) with a 1e-5 learning rate and other parameters set to default. The epoch number was usually 5, and we did experiments with a range of (3–40) batch numbers with regard to the availability of the GPU. For evaluation, we used the sklearn library's f_1-score function

C Dataset Statistics and Evaluation

---

[4] https://allenai.github.io/scispacy/

[5] https://huggingface.co/allenai/specter

Datasets	Context	Label distribution	Total size	Query length	Document length
SciFact	Biomedicine	S:556, R:337, N:516	1.4K	12.9	232.9
HEALTHVER	Health	S:4.3K, R:2.8K, N:5.3K	12.5K	18.4	34.8
CLIMATE-FEVER	Climate change	S:654, R:253, N:474, D:154	1.5K	20.5	77.5
COVID-Fact	Covid	S:1.2K, R:2.7K	4K	13.3	77.9
Med-Fact	Biomedical	S:50K, R:50K, N:50K	150K	13.7	125.1
Gsci-Fact	General Science	S:10.7K, R:10.7K, N:10.7	32.2K	12.8	74.1

Table 5: Statistics of the datasets that we considered in this study for scientific-fact-checking task. The letters S, R, N, D stand for supported, refuted, not enough info and disputed claims, respectively. The last two columns show the average lengths of the claims and the documents.

Fluency	3-The claim is free of grammatical errors, and its meaning is clear. 2-The claim is understandable despite some grammatical errors. 1-The claim is incomprehensible. .
Contextuality	1-The claim can be interpreted without any additional context. 0-Without the original context, the claim cannot be interpreted meaningfully.
Faithfulness	1-The claim is correct with respect to the explanation. 2-The claim is incorrect with respect to the explanation. 3-The claim is related to the explanation, but the verdict of the claim cannot be inferred from the explanation. 4-The claim is not related to explanation in any sense.
Challenge	1- I can confidently say whether the claim is correct or incorrect without reading the explanation. 0- I cannot confidently say whether the claim is correct or incorrect without reading the explanation.

Table 6: Manual evaluation criteria for fluency, contextuality, faithfulness and challange.

Claim	Diarrhoea is a common symptom of haloperidol toxicity.
Explanation	Symptoms of haloperidol toxicity are usually due to exaggerated side effects. Most often encountered are: Severe extrapyramidal side effects with muscle rigidity and tremors, akathisia, etc. Hypotension or hypeension Sedation Anticholinergic side effects (dry mouth, constipation, paralytic ileus, difficulties in urinating, decreased perspiration), coma in severe cases, accompanied by respiratory depression and massive hypotension, shock. Rarely, serious ventricular arrhythmia (torsades de pointes), with or without prolonged QT-time Epileptic seizures.
Fluency-3	The claim is free of grammatical errors, and its meaning is clear.
Contextuality-1	The claim can be interpreted without any additional context.
Faithfulness-2	The claim is incorrect with respect to the explanation
Challange-0	I cannot confidently say whether the claim is correct or incorrect without reading the explanation.

Table 7: An example of evaluation of a refuted claim from the Med-Fact dataset.

Claim	Zinc is more easily oxidized than iron.
Explanation	One way to keep iron from corroding is to keep it painted. The layer of paint prevents the water and oxygen necessary for rust formation from coming into contact with the iron. As long as the paint remains intact, the iron is protected from corrosion. Other strategies include alloying the iron with other metals. For example, stainless steel is mostly iron with a bit of chromium. The chromium tends to collect near the surface, where it forms an oxide layer that protects the iron. Zinc-plated or galvanized iron uses a different strategy. Zinc is more easily oxidized than iron because zinc has a lower reduction potential. Since zinc has a lower reduction potential, it is a more active metal. Thus, even if the zinc coating is scratched, the zinc will still oxidize before the iron. This suggests that this approach should work with other active metals. Another important way to protect metal is to make it the cathode in a galvanic cell. This is cathodic protection and can be used for metals other than just iron. For example, the rusting of underground iron storage tanks and pipes can be prevented or greatly reduced by connecting them to a more active metal such as zinc or magnesium. This is also used to protect the metal parts in water heaters. The more active metals (lower reduction potential) are called sacrificial anodes because as they get used up as they corrode (oxidize) at the anode. The metal being protected serves as the cathode, and so does not oxidize (corrode). When the anodes are properly monitored and periodically replaced, the useful lifetime of the iron storage tank can be greatly extended.
Fluency-3	The claim is free of grammatical errors, and its meaning is clear.
Contextuality-1	The claim can be interpreted without any additional context.
Faithfulness-1	The claim is correct with respect to the explanation.
Challange-0	I cannot confidently say whether the claim is correct or incorrect without reading the explanation.

Table 8: An example of evaluation of supported claim from the Gsci-Fact dataset.

Claim	Glycogen phosphorylase requires thiamine pyrophosphate.
Explanation	Glycogen phosphorylase removes glucose as glucose-1-phosphate from glycogen (phosphorolysis). It contains pyridoxal. Formation of branches in glycogen phosphate (PLP) as a prosthetic group. The alpha-1,4 linkages in the glycogen are cleaved and removes glucose units one at a time. Enzyme sequentially hydrolyses alpha-1,4 glycosidic linkages, till it reaches a glucose residue, 3-4 glucose units away from a branch point. It cannot attack the 1,6 linkage at branch point.
Fluency-3	The claim is free of grammatical errors, and its meaning is clear.
Contextuality-1	The claim can be interpreted without any additional context.
Faithfulness-3	The claim is related to the explanation, but the verdict of the claim cannot be inferred from the explanation.
Challange-0	I cannot confidently say whether the claim is correct or incorrect without reading the explanation.

Table 9: An example of evaluation of not-enough-info claim from the Med-Fact dataset.

On Evaluation of Document Classification with RVL-CDIP

Stefan Larson[1][*] **Gordon Lim**[2] **Kevin Leach**[1]

[1]Vanderbilt University
Nashville, TN, USA

[2]University of Michigan
Ann Arbor, MI, USA

Abstract

The RVL-CDIP benchmark is widely used for measuring performance on the task of document classification. Despite its widespread use, we reveal several undesirable characteristics of the RVL-CDIP benchmark. These include (1) substantial amounts of label noise, which we estimate to be 8.1% (ranging between 1.6% to 16.9% per document category); (2) presence of many ambiguous or multi-label documents; (3) a large overlap between test and train splits, which can inflate model performance metrics; and (4) presence of sensitive personally-identifiable information like US Social Security numbers (SSNs). We argue that there is a risk in using RVL-CDIP for benchmarking document classifiers, as its limited scope, presence of errors (state-of-the-art models now achieve accuracy error rates that are within our estimated label error rate), and lack of diversity make it less than ideal for benchmarking. We further advocate for the creation of a new document classification benchmark, and provide recommendations for what characteristics such a resource should include.

1 Introduction

Within the document understanding research area, the RVL-CDIP dataset (Harley et al., 2015) has emerged as the primary benchmark for evaluating and comparing document classifiers. RVL-CDIP is composed of 16 document type categories, including `resume`, `letter`, `invoice`, etc. Its large volume of training data—320,000 samples—facilitates benchmarking state-of-the-art deep learning and transformer-based architectures. While initially released as a computer vision benchmark in 2015, more recent state-of-the-art models now incorporate image, text, and page layout modalities. For instance, recent tri-modal models like DocFormer (Appalaraju et al., 2021), ERNIE-Layout (Peng et al., 2022), LayoutLMv3 (Huang et al., 2022), and Bi-VLDoc (Luo et al., 2022) now achieve classification accuracies ranging in the mid- to high-90s, with Bi-VLDoc reporting a state-of-the-art of 97.12% on the RVL-CDIP test set. This is a large improvement over earlier image-centric work, and we chart this improvement in Figure 1.

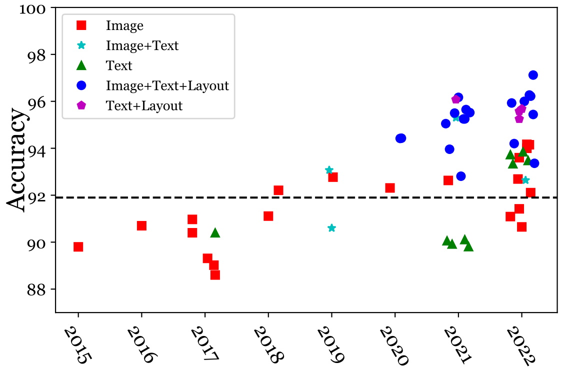


Figure 1: Model accuracy on RVL-CDIP by year and modality. The horizontal dashed line represents our estimated label error rate for RVL-CDIP's test set.

As model performance on RVL-CDIP improves, it becomes increasingly important to ensure that further gains are meaningful with respect to the classification task. This concern has been raised by prior work that has found that benchmark evaluation datasets often contain substantial amounts of label errors or noise (e.g., Northcutt et al., 2021a), substantial overlap between test and train data (e.g., Elangovan et al., 2021; Søgaard et al., 2021), and data collection artifacts that cause models to overfit to spurious cues (e.g., Gururangan et al., 2018; McCoy et al., 2019). Therefore, we cast a critical eye to the RVL-CDIP benchmark to answer: *Is RVL-CDIP still suitable for effectively measuring the performance of document classifiers?*

In doing so, we first observe a lack of clear label or annotation guidelines provided with the original introduction of RVL-CDIP. Therefore, we create verifiable label guidelines for the 16 RVL-CDIP categories. With these guidelines, we are then able to conduct a review of the data, and we find that

---

[*] Corresponding email: `stefan.dataset@gmail.com`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2665–2678
May 2-6, 2023 ©2023 Association for Computational Linguistics

label errors account for an estimated 8.1% of data RVL-CDIP's test split, a rate greater than the current state-of-the-art model accuracy error rate, indicating that contemporary high-performing models are overfitting to noise. We also observe relatively high rates of documents that have ambiguous or multiple valid labels, which is problematic given RVL-CDIP is a single-label classification benchmark. Additionally, we also observe a large overlap between test and train data splits, where there are (near-) duplicate documents seen in both train and test splits, as well as documents that share common templates. Lastly, our review of RVL-CDIP data uncovered a surprisingly large amount of sensitive personally-identifiable information, particularly in the `resume` category, where we found 7.7% of documents contained US Social Security numbers.

We argue that the characteristics that we observe make RVL-CDIP an unattractive benchmark for training and evaluating document classifiers. We end with recommendations for what qualities a new document classification benchmark should have.

2 Related Work

This section discusses related work in two areas: (1) prior work in document classification on RVL-CDIP, and (2) prior work on analyzing datasets.

2.1 RVL-CDIP and Document Classification

The RVL-CDIP corpus has been used as a benchmark for document classification since its introduction by Harley et al. (2015), who used it to evaluate convolutional neural network (CNN) image classifiers on the dataset's document images. Most immediate follow-up work followed Harley et al. (2015) and explored different image-based CNN models, as done in Csurka et al. (2016); Afzal et al. (2017); Tensmeyer and Martinez (2017); Das et al. (2018); Ferrando et al. (2020). Just relying on image features is limited, as much of a document's "essence" is informed by its textual content. Therefore, more recent work has incorporated the textual modality, including Audebert et al. (2019) and Dauphinee et al. (2019).

Model (Reported by)	**Modality**	**Accuracy**
Bi-VLDoc (Luo et al., 2022)	I, T, L	97.12
ERNIE-Layout-large (Peng et al., 2022)	I, T, L	96.27
UDOP-Dual (Tang et al., 2022)	I, T, L	96.22
DocFormer-base (Appalaraju et al., 2021)	I, T, L	96.17
StructuralLM-large (Li et al., 2021a)	T, L	96.08
UDOP (Tang et al., 2022)	I, T, L	96.00
LayoutLMv3-large (Huang et al., 2022)	I, T, L	95.93
LiLT-base (Wang et al., 2022a)	T, L	95.68
LayoutLMv2-large (Xu et al., 2021)	I, T, L	95.65
BROS-base (Wang et al., 2022a)	T, L	95.58
TILT-large (Powalski et al., 2021)	I, T, L	95.52
DocFormer-large (Appalaraju et al., 2021)	I, T, L	95.50
LayoutLMv3-base (Huang et al., 2022)	I, T, L	95.44
Donut (Kim et al., 2021)	I, T	95.30
Wukong-Reader-large (Bai et al., 2022)	I, T, L	95.26
Pham et al. (2022)	T, L	95.25
TILT-base (Powalski et al., 2021)	I, T, L	95.25
LayoutLMv2-base (Xu et al., 2021)	I, T, L	95.25
UDoc-star (Gu et al., 2021)	I, T, L	95.05
Wukong-Reader-base (Bai et al., 2022)	I, T, L	94.91
LayoutLMv1-base (Xu et al., 2020)	I, T, L	94.43
LayoutLMv1-large (Xu et al., 2020)	I, T, L	94.42
MATrIX (Delteil et al., 2022)	I, T, L	94.20
DocXClassifier-xl (Saifullah et al., 2022a)	I	94.17
DocXClassifier-large (Saifullah et al., 2022a)	I	94.15
DocXClassifier-base (Saifullah et al., 2022a)	I	94.00
UDoc (Gu et al., 2021)	I, T, L	93.96
Longformer-base (Pham et al., 2022)	T	93.85
Longformer-large (Pham et al., 2022)	T	93.73
MGDoc (Wang et al., 2022b)	I, T, L	93.64
Dessurt (Davis et al., 2022)	I	93.60
Bigbird-base (Pham et al., 2022)	T	93.48
Pramanik et al. (2022)	I, T, L	93.36
Bigbird-large (Pham et al., 2022)	T	93.34
Multimodal Ensemble (Dauphinee et al., 2019)	I, T	93.07
SelfDoc (Li et al., 2021b)	I, T, L	92.81
LadderNet (Sarkhel and Nandi, 2019)	I	92.77
Zingaro et al. (2021)	I, T	92.70
DiT-large (Li et al., 2022)	I	92.69
VLCDoC (Bakkali et al., 2022)	I, T	92.64
InceptionResNetV2 (Xu et al., 2021)	I	92.63
EfficientNet (Ferrando et al., 2020)	I	92.31
Region Ensemble (Das et al., 2018)	I	92.21
DiT-base (Li et al., 2022)	I	92.11
MAE-base (Li et al., 2022)	I	91.42
Stacked CNN Single (Das et al., 2018)	I	91.11
BEiT-base (Li et al., 2022)	I	91.09
VGG-16 (Afzal et al., 2017)	I	90.97
Csurka et al. (2016)	I	90.70
ResNext-101 (Li et al., 2022)	I	90.65
Audebert et al. (2019)	I, T	90.60
ResNet-50 (Afzal et al., 2017)	I	90.40
RoBERTa-large (Li et al., 2021a)	T	90.11
RoBERTa-base (Li et al., 2021a)	T	90.06
BERT-large (Li et al., 2021a)	T	89.92
BERT-base (Li et al., 2021a)	T	89.81
Tensmeyer and Martinez (2017)	I	89.31
GoogLeNet (Afzal et al., 2017)	I	89.02
AlexNet (Afzal et al., 2017)	I	88.60

Table 1: Model accuracy on RVL-CDIP for various image (I), text (T), and layout-based (L) document classification models, ordered by reported score. Models incorporating multiple modalities typically outperform uni-modal models.

Even more recent work has capitalized on the transformer model architecture, often combining vision transformers with large transformer-based language models (and often combining these with a third modality based on page layout derived from detected optical character recognition (OCR) regions) as in LayoutLMv1 (Xu et al., 2020), LayoutLMv2 (Xu et al., 2021), LayoutLMv3 (Huang et al., 2022), DocFormer (Appalaraju et al., 2021), TILT (Powalski et al., 2021), and ERNIE-layout (Peng et al., 2022). These more recent transformer-based models have achieved state-of-the-art accuracy scores on RVL-CDIP, the most recent being Luo et al. (2022)'s Bi-VLDoc, which achieves a reported accuracy of 97.12% on RVL-CDIP. (For a listing of models benchmarked on RVL-CDIP since Harley et al. (2015), see Table 1.)

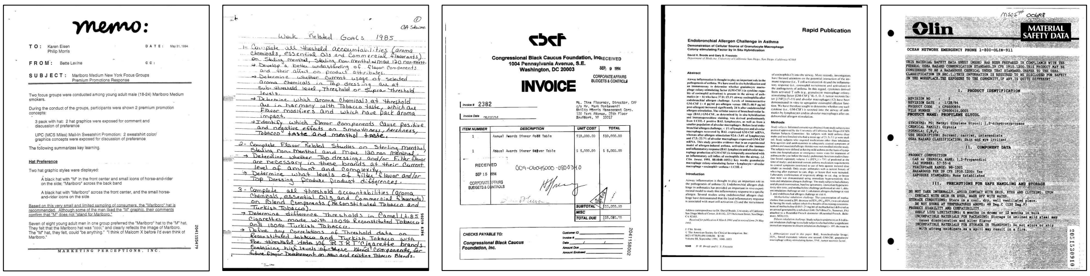

memo handwritten invoice scientific_pub. specification

Figure 2: Samples from the RVL-CDIP dataset.

advertisement
budget
email
file_folder
form
handwritten
invoice
letter
memo
news_article
presentation
questionnaire
resume
scientific_publication
scientific_report
specification

Table 2: RVL-CDIP document type categories.

Despite these high scores, recent work has exposed gaps in models trained on RVL-CDIP. In particular, recent work has found that models trained on RVL-CDIP perform poorly on out-of-distribution data (Larson et al., 2022) and perturbed in-distribution data (Saifullah et al., 2022b). We also mention that RVL-CDIP is often used as a pre-training dataset, where models are first pre-trained on RVL-CDIP (and perhaps others) and then evaluated on other downstream tasks or datasets (e.g., Nguyen et al. (2021); Kanchi et al. (2022)). Other datasets like FUNSD are subsets of RVL-CDIP (Jaume et al., 2019).

2.2 Analysis of Datasets

Prior work has investigated the presence of label and annotation errors and corpus quality in NLP and image datasets. This work includes Abedjan et al. (2016); Radenović et al. (2018); Müller and Markert (2019); Pleiss et al. (2020); Northcutt et al. (2021a); Kreutzer et al. (2022); Ying and Thomas (2022); Chong et al. (2022). One common conclusion is that the utility of a benchmark evaluation dataset is lessened if the label error and/or ambiguity rate is close to- or exceeds model prediction error rate. This has been observed for various datasets, such as ATIS (Béchet and Raymond, 2018; Niu and Penn, 2019), and the CNN/Daily Mail reading comprehension task (Chen et al., 2016).

Orthogonal to label errors, prior work has also observed non-trivial overlap between test and train splits in datasets on which natural language processing and computer vision models are evaluated (e.g., Finegan-Dollak et al., 2018; Allamanis, 2019; Barz and Denzler, 2020; Lewis et al., 2021; Wen et al., 2022; Croft et al., 2023). Such work often argues that non-trivial amounts of overlap between test and train data can lead to "inflated" performance scores, as overlapping data can reward a model's ability to memorize training data (Elangovan et al., 2021), and to under-estimate out-of-sample error (Søgaard et al., 2021). Evidence of this can also be found in the multitude of studies that report lower model performance scores on newly-collected evaluation sets versus reported scores on benchmarks (e.g., Augenstein et al., 2017; Recht et al., 2019; Harrigian et al., 2020; Kim and Kang, 2022; Larson et al., 2022). In this paper, we investigate the presence of errors, ambiguous data, and overlapping test-train data for the RVL-CDIP benchmark dataset.

3 The RVL-CDIP Dataset

The RVL-CDIP dataset was introduced in Harley et al. (2015) as a benchmark for evaluating image-based classification and retrieval tasks.[1] Since then, RVL-CDIP has primarily been used as a document type classification benchmark. RVL-CDIP consists of 400,000 document images distributed across 16 document type categories, listed in Table 2. Example documents from RVL-CDIP are shown in Figure 2. Documents in RVL-CDIP were sampled from the larger IIT-CDIP Test Collection, which itself is a snapshot of the voluminous Legacy Tobacco Documents Library (LTDL) collection — at that time, LDTL contained approximately 7 million documents (Lewis et al., 2006).[2] These documents were made publicly available as part of legal pro-

---

[1] Harley et al. (2015) referred to RVL-CDIP as *BigTobacco*.

[2] The LTDL is now called the Truth Tobacco Industry Documents collection, and is included in the broader Industry Documents Library (IDL) hosted by the UCSF Library: `https://www.industrydocuments.ucsf.edu/`. For more background on the LTDL, see Schmidt et al. (2002) and Tasker et al. (2022).

Category	**Description**
`advertisement`	Advertisements from print-form media like newspapers and magazines. Also a small amount of scripts for television or radio advertisements. A small amount of "ad order instructions" and "ad insertion" documents.
`budget`	Includes various budget documents such as expense, spending, sales, cash, and accounting reports and forecasts; budgets; quotes and estimates; and income and bank statements. Also includes receipt-like documents such as political campaign contribution requests and other receipts, as well as checks and check stubs.
`email`	Scanned images of printed emails.
`file_folder`	Scanned images of folders and binders. Folder scans are often characterized by vertically oriented text (indicating a folder label). A moderate amount of file folders in RVL-CDIP contain handwritten text or notes. Some scanned folders may be indistinguishable from blank pages.
`form`	Form documents with form-like elements (e.g., lines or spaces for user-provided data entry). The form-like elements can appear empty or filled.
`handwritten`	Includes handwritten documents like handwritten letters and scientific notes.
`invoice`	Includes invoices, bills, and account statements.
`letter`	Letters, often with letterhead and commonly with "Dear..." salutations. The distinction between letters and memos is often unclear in RVL-CDIP.
`memo`	Memoranda or inter-office correspondence documents, often with clear "TO", "FROM", "SUBJECT" headings.
`news_article`	Includes news articles in the form of clippings from newspapers and other print-form news media, as well as a small amount of news articles from the web.
`presentation`	Includes scanned images of presentation and overhead slides, transcripts of speeches and statements. Also includes a large amount of press releases.
`questionnaire`	Includes customer surveys and questionnaires, as well as survey and questionnaire prompts for surveyors. Also includes questionnaires appearing to be part of legal proceedings and investigations. In RVL-CIDP, many questionnaires have a substantial amount of form-like elements.
`resume`	Includes resumes, curricula vitae (CVs), biographical sketches, executive biographies (e.g., those written in third-person), a small amount of business cards.
`scientific_pub.`	Mainly papers and articles from scientific journals and book chapters, but also includes book title pages. Also includes news articles from science newsletters. News articles from science newsletters are very similar to the `news_article` category.
`scientific_rep.`	Includes bioassay, pathology, and test reports; charts, graphs, and tables; research reports (including progress reports), research proposals, abstracts, paper drafts. Many reports and abstracts bear similarities to scientific publications. Many test result documents are similar to documents in the `specification` category.
`specification`	Data sheets (including safety data sheets); product, material, and test specifications. Also includes specification change reports.

Table 3: RVL-CDIP categories alongside our descriptions and notes.

ceedings and settlements against several American tobacco and cigarette companies and organizations, and as such, the documents in RVL-CDIP are almost exclusively related to the tobacco industry.[3]

Most document images in RVL-CDIP capture the initial page of a document; some common exceptions appear to be charts and tables (these are typically labeled as `scientific_report`) as well as presentation slides (labeled as `presentation`). Additionally, almost all of the documents (that contain readable text) are in English, although we did find small amounts of documents in other languages (including German, Dutch, French, Spanish, Portuguese, Italian, Japanese, Chinese, Arabic, and Hebrew) as part of our review. Examples of non-English RVL-CDIP samples are displayed in Figure 11 in the Appendix.

There are 320,000 training, 40,000 validation, and 40,000 test samples, but Harley et al. (2015) provides no information on how the data was partitioned into these splits, so we assume it was done randomly for each of the 16 document categories. Harley et al. (2015) report that the 16 categories were chosen, in part, because these categories had ample representation (i.e., at least 25,000 samples) in IIT-CDIP. Unfortunately, we are unaware of any published guidelines, criteria, rules, or documentation defining or describing each of the 16 RVL-CDIP categories, nor is it clear who or what provided the initial category labels in IIT-CDIP (nor in LTDL).[4] Thus, we describe how we developed label guidelines for each RVL-CDIP document type category in Section 3.1 below.

---

[3] For more background on the history of the litigation and documents, see Glantz et al. (1996); Ciresi et al. (1999); Tasker et al. (2022).

[4] Schmidt et al. (2002) and Tasker et al. (2022) indicate that type labels may have been ascribed to the documents by human workers employed at UCSF's LTDL.

3.1 Establishing Label Guidelines

The RVL-CDIP dataset does not have a published list of descriptions, rules, or guidelines describing each of the 16 document type categories. We discuss an extensive analysis from which we develop such guidelines.

We established our list of guidelines by first sampling 1,000 documents from each of the 16 categories in the training set (for a total of 16,000 documents). We then reviewed these samples category by category. This review process helped us identify commonalities within each category, and helped us discover that many of the categories seem to have distinct groups of sub-types within them. For instance, we found that the `resume` category is largely composed of (1) resumes and curricula vitae, (2) "Biographical Sketch" documents (i.e., those required for grant applications for the National Institutes of Health (example shown in Figure 3a), (3) executive biographies, and (4) scanned business cards. Such cases reveal opportunities for refining and diversifying appropriate categories.

In another category, `advertisement`, we found samples mostly consisted of advertisements from print-form media like newspapers and magazines, as well as smaller amounts of scripts for television or radio advertisements. The `advertisement` category also included a small amount of document images identical to the one shown in Figure 3b. We found that this "IMAGE NOT AVAILABLE" document appears mostly in the `advertisement` category, yet it is an example of a document that we do not include in our label guidelines for this category, as it is not at all faithful to the semantic nature of the `advertisement` category.

Our annotation guidelines are listed in Table 3, along with our notes and observations. It was occasionally necessary to review multiple document categories prior to establishing rules. This was the case with the `budget` and `invoice` categories, each of which included non-trivial amounts of scanned check images and contribution requests. (Examples of cases like these are displayed in Figures 21-23 in the Appendix.) For cases like these, we annotated these sub-types in the relevant categories in order to estimate their relative frequencies. We then would append our annotation guidelines accordingly; for instance, 8.8% of `budget` documents and 3.8% of `invoice` documents that we reviewed were check images, so our guidelines specify that the `budget` category consists of check images, while `invoice` does not. Ultimately, our goal with establishing such guidelines is to provide repeatable, verifiable criteria that faithfully reflect the semantic nature of each category.

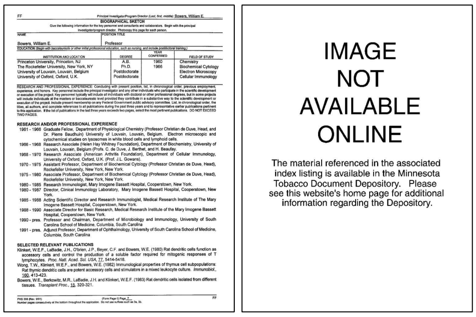


IMAGE
NOT
AVAILABLE
ONLINE

The material referenced in the associated index listing is available in the Minnesota Tobacco Document Depository. Please see this website's home page for additional information regarding the Depository.

(a) (b)

Figure 3: Example "Biographical Sketch" `resume` document (a) and "IMAGE NOT AVAILABLE" document found mostly in `advertisement`.

4 Label Errors and Ambiguities in RVL-CDIP

Armed with better knowledge of what constitutes each of the 16 RVL-CDIP categories, we analyze the contents of the RVL-CDIP test set to estimate the amount of label errors and ambiguities found in this set.

We manually checked for errors in the RVL-CDIP test set by sampling 1,000 documents from each of the 16 categories (for a total of 16,000 documents). We used our label guidelines established in Section 3.1 to help us determine the validity of each of these 16,000 samples. We tracked several types of errors and ambiguities: (1) documents found in a category that clearly are mis-labeled and instead belong in a different RVL-CDIP category — we refer to this error type as *mis-labeled*; (2) documents that do not appear to have a single clear RVL-CDIP label — we refer to this label type as *unknown*; (3) documents that have mixed or multiple features that belong to at least two RVL-CDIP categories — we refer to this type as *mixed*. Examples of documents exhibiting these error types can be seen in Figure 4. We point out a particularly interesting *mixed* case: the first two *mixed* examples are nearly identical, but the original label is `news_article` in one case but `letter` in the second. More examples are shown in the Appendix in Figures 12–14.

Findings. Our estimated error rates in the RVL-CDIP test set are shown in Table 4. We estimate that error rates (i.e., combined rates for *mis-labeled* and *unknown*) range between 1.6% (in the case

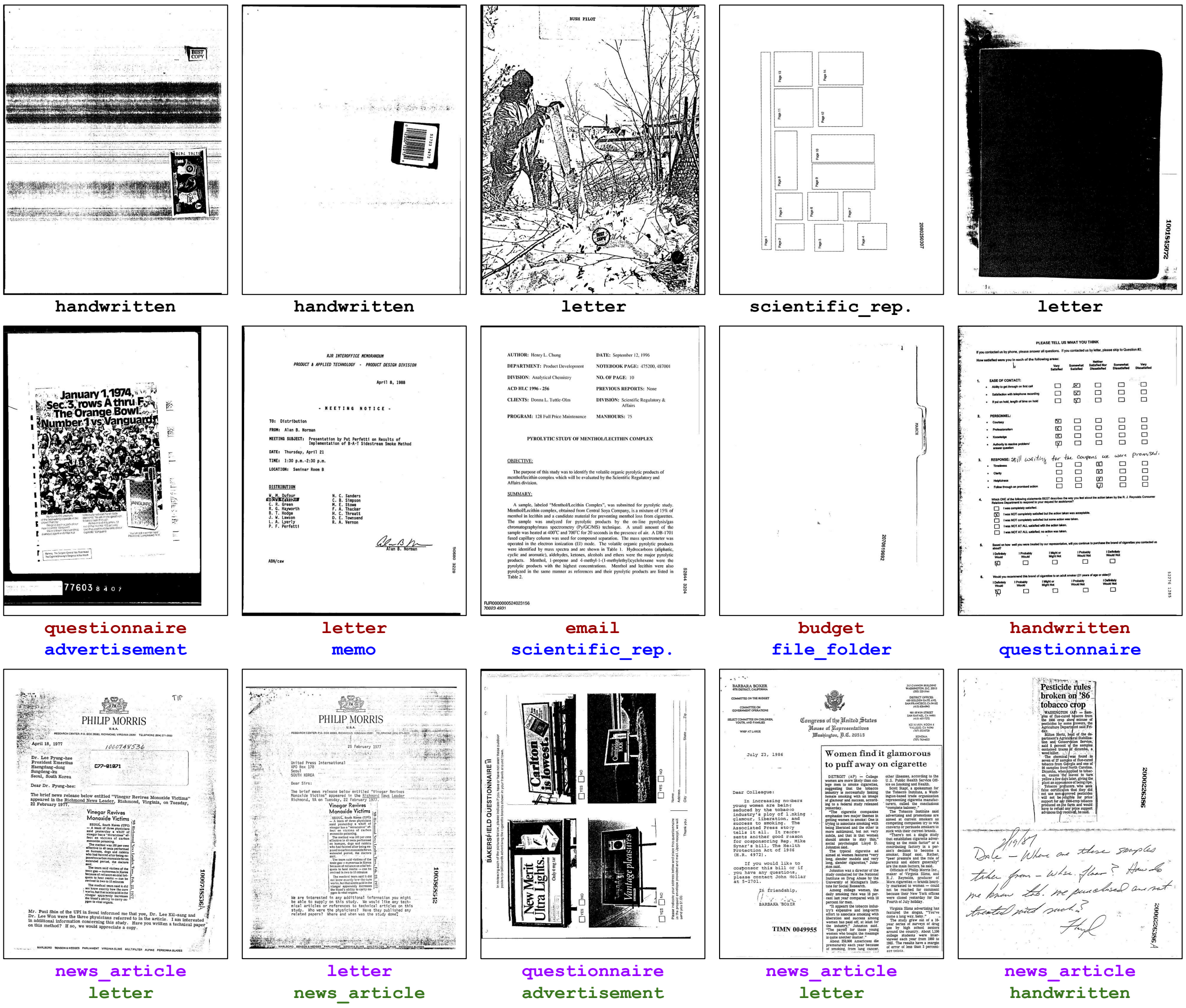


Figure 4: Example errors and ambiguities. Top row: *unknown*, middle row: *mis-label*, bottom row: *mixed*.

Category	*mis-labeled*	*unknown*	*Tot. Error*	*mixed*
`advertisement`	1.9%	4.5%	6.4%	3.5%
`budget`	9.7%	4.1%	13.8%	1.5%
`email`	1.7%	8.3%	10.0%	0.4%
`form`	4.4%	6.4%	10.8%	0.5%
`file_folder`	0.4%	3.1%	3.5%	1.9%
`handwritten`	2.5%	5.2%	7.7%	2.4%
`invoice`	9.7%	1.3%	11.0%	0.2%
`letter`	13.5%	3.4%	16.9%	0.5%
`memo`	2.0%	2.4%	4.4%	2.1%
`news_article`	4.6%	2.5%	7.1%	0.4%
`presentation`	1.8%	4.9%	6.7%	1.0%
`questionnaire`	5.6%	7.3%	12.9%	6.9%
`resume`	0.2%	1.4%	1.6%	0.4%
`scientific_pub.`	2.5%	1.8%	4.3%	0.0%
`scientific_rep.`	4.6%	3.9%	8.5%	5.6%
`specification`	1.5%	1.9%	3.4%	0.4%
Average	4.2%	3.9%	8.1%	1.7%

Table 4: Estimated label error and multi-label rates in the RVL-CDIP test set.

of `resume`) and 16.9% (in the case of `letter`). The average of each category's error rates is 8.1%, which is higher than the classification accuracy error rates reported by many state-of-the-art models listed in Table 1. In some cases, the majority of a category's errors were mis-labels of a particular type. For instance, about 59% of the erroneous `letter` documents we reviewed were actually `memo` documents. Similarly, 74% of the erroneous `invoice` documents were actually budget documents. Lastly, roughly 1.7% of RVL-CDIP's test set is data that have multiple valid labels.

5 Overlap Between Test and Train Splits

Our analysis also reveals a substantial degree of undesirable overlap between train and test samples within RVL-CDIP. To measure this overlap, we use an approach similar to Larson et al. (2019) and Elangovan et al. (2021), which, for each test sample in each document type category, finds the maximally similar sample in the same document


resume: 0.977

news_article: 0.981

invoice: 0.937

Figure 5: Example test-train pairs with corresponding maximum cosine similarity scores. These three example pairs show instances of near-duplicates (left and center) and documents that have highly similar structure (right).

Category	mean	median
`advertisement`	0.893	0.903
`budget`	0.963	0.968
`email`	0.976	0.982
`form`	0.948	0.956
`file_folder`	0.967	0.974
`handwritten`	0.945	0.952
`invoice`	0.962	0.966
`letter`	0.953	0.960
`memo`	0.957	0.961
`news_article`	0.919	0.936
`presentation`	0.929	0.945
`questionnaire`	0.961	0.968
`resume`	0.965	0.967
`scientific_pub.`	0.936	0.955
`scientific_rep.`	0.950	0.961
`specification`	0.972	0.978
Average	0.950	0.958

Table 5: Mean and median of maximum cosine similarity scores between train and test sets for each RVL-CDIP category.

type category's training split. We then average these maximum similarity scores together for each document category. That is, for each document category C in RVL-CDIP, we compute

$$\frac{1}{|test_C|} \sum_{b \in test_C} \max_{a \in train_C} sim(a, b)$$

where a and b are samples from category C's train and test splits, respectively. We use CLIP (Radford et al., 2021) to extract a 512-dimension feature embedding from each sample, and use cosine similarity for $sim(\cdot, \cdot)$. We note that this vector-based similarity technique is common practice in the image- and information retrieval (e.g., Babenko et al. (2014)).

Findings. Average and median of the maximum similarity scores for test-train pairs are shown in Table 5 for each RVL-CDIP category. Overall, we see a high degree of similarity across test and train data: mean scores range between 0.893 (`advertisement`) and 0.976 (`email`), with an average of 0.950. Ten of the 16 document categories have average scores at- or above 0.95. The median score for each category is larger than the mean in all cases, indicating a long tail in the distribution of scores. Indeed, we see this in Figure 10 (in Appendix), which charts the distribution of similarity scores for all test data in RVL-CDIP. Figure 5 shows three examples of test-train pairs with similarity scores ranging between 0.937 and 0.981. Two of the three pairs in Figure 5 seem to be near-duplicates, where there appear to be minor differences in scanning or noise artifacts between each document. In the third (`invoice`) example, we see that the two samples are distinct, yet both share a large degree of similarity because both use the same document template (e.g., invoices from the same company that are structurally and visually similar but that contain different "data"). We show more example pairs in Figures 15–18 in the Appendix.

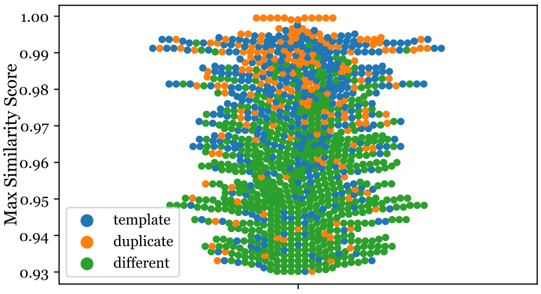
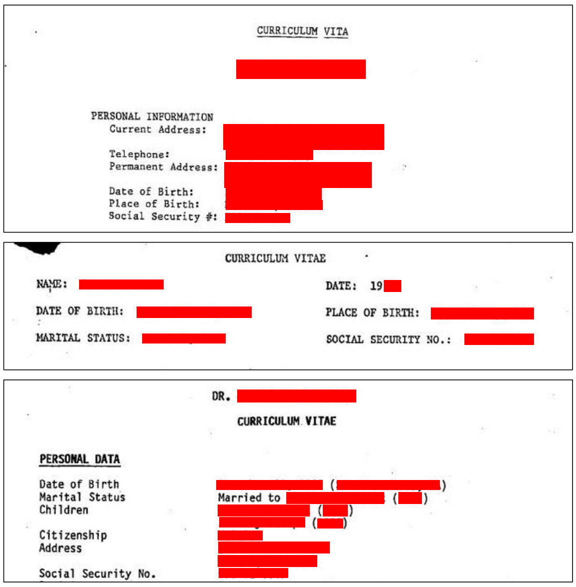


Figure 6: Sampled subset of maximal similarity scores for test-train pairs with scores between 0.93 and 1.0.

To help better understand the similarity scores, we conduct an experiment where we categorize each similarity pair into one of the following: *duplicate*, if the test-train pair represents the same document; *template*, if both documents in a pair use the same document template; and *different*, for all other pairs. We annotated a sample of 1,086 similarity pairs with maximum similarity scores ranging between 0.93 and 1.0. A visualization of the relationship between maximal similarity score and match type is shown in Figure 6, where we



CURRICULUM VITA

[redacted]

PERSONAL INFORMATION
Current Address: [redacted]
Telephone: [redacted]
Permanent Address: [redacted]
Date of Birth: [redacted]
Place of Birth: [redacted]
Social Security #: [redacted]

CURRICULUM VITAE

NAME: [redacted] DATE: 19[redacted]
DATE OF BIRTH: [redacted] PLACE OF BIRTH: [redacted]
MARITAL STATUS: [redacted] SOCIAL SECURITY NO.: [redacted]

DR. [redacted]

CURRICULUM VITAE

PERSONAL DATA
Date of Birth [redacted] ([redacted])
Marital Status Married to [redacted] ([redacted])
Children [redacted] ([redacted])
[redacted] ([redacted])
Citizenship [redacted]
Address [redacted]
[redacted]
Social Security No. [redacted]

Figure 7: Example documents from RVL-CDIP showing sensitive personally identifiable information (PII; redacted by us).

observe that the likelihood of a pair being either a *duplicate* or *template* match increases with similarity score.

Considering the overall median maximal similarity score is 0.958, we can estimate a lower-bound for the rate of *duplicate* and *template* match pairs by scaling the proportion of documents above the median maximal score (i.e., half, or 0.5) by the fraction of *duplicate* and *template* matches above the median (0.958). This gives us 0.5×0.641, and therefore we estimate that at least 32% of samples from the RVL-CDIP test set have either a duplicate counterpart or a sample that shares a template layout in the training set. While there is generally no established acceptable number or percentage for test-train overlaps, prior work (e.g., Søgaard et al. (2021); Elangovan et al. (2021)) has argued that overlaps are undesirable, and that building generalizable, robust models entails evaluation against novel, unseen data points (e.g., Koh et al. (2021); Malinin et al. (2021); Larson et al. (2022)).

6 Presence of Sensitive Information

While reviewing samples from RVL-CDIP, we noticed that the `resume` category had a non-trivial quantity of documents that contain sensitive and personally-identifiable entities. Naturally, resumes typically contain a person's name and basic contact information (e.g., phone numbers or email addresses). However, we found a plethora of sensitive entities like citizenship and marital statuses, places and dates of birth, names of children and spouses, and national ID numbers like US Social Security and Canadian National ID numbers.

Out of a sample of 1,000 documents from the `resume` test set, we found that 7.7% contained a US Social Security Number. While we recognize that US Social Security numbers were not considered sensitive several decades ago (when many of the `resume` documents in RVL-CDIP were created), their presence in so many documents in a publicly accessible dataset[5] is still striking, especially considering the coexistence of this entity type with others like person names, dates and places of birth, etc. In particular, malicious Social Security numbers are often connected with fraud and identity theft crimes in the USA. Moreover, the sensitive entities discussed in this section are considered highly sensitive under many state and national laws.[6] Additionally, we found that 43.6% of the test resumes contain birth dates, 19.9% contain places of birth, 11.4% contain marital (or spousal or parental) statuses, and 8.9% contain citizenship statuses. Example documents containing sensitive PII can be seen in Figure 7.

Given the presence of sensitive PII in RVL-CDIP, it is reasonable to wonder if sensitive PII also appears in datasets derived from RVL-CDIP, like FUNSD (Jaume et al., 2019). Similarly, we also wonder if sensitive PII appears in datasets that were derived from the larger IIT-CDIP or UCSF Industry Documents Library corpora, such as Tobacco-800 (Zhu et al., 2007; Zhu and Doermann, 2007), Tobacco-3482 (Kumar et al., 2014), DocVQA (Mathew et al., 2021), and OCR-IDL (Biten et al., 2022). We will investigate this in future work.

7 Discussion and Recommendations

Given our findings concerning labeling errors, test/train overlap, and presence of sensitive information in the RVL-CDIP document classification benchmark, we discuss several concrete recommendations to raise awareness among researchers engaged in benchmarking classifiers using this dataset:

(0) Sub-Types in RVL-CDIP. Our investigation into RVL-CDIP revealed that many of the RVL-CDIP categories are in fact composed of several

---

[5] On 9 Feb. 2023, RVL-CDIP tallied "1,765 downloads last month" on the Hugging Face Datasets platform.

[6] For example, the State of Michigan's Social Security Number Privacy Act (2004).

sub-types. We encourage researchers and practitioners to be aware of this fact. For instance, curricula vitae, bigraphical sketches, executive biographies, and business cards are the four sub-types of the `resume` category. This finding has implications for modeling tasks where prior knowledge of the label set is assumed, like in zero-shot settings where each category may be specified to the model as a string, as done in Siddiqui et al. (2021). Additionally, unsupervised clustering analyses like Finegan-Dollak and Verma (2020) may exhibit low performance scores on RVL-CDIP due to many of the categories having distinct and disparate sub-types (e.g., radio scripts versus print advertisements in the `advertisement` category, or business cards versus biographical sketches in the `resume` category).

(1) Errors. Users of RVL-CDIP should be aware that there are many label errors and noisy samples with unknown labels in RVL-CDIP. Recall from Section 4 that an estimated 8.1% of test samples from RVL-CDIP contain label errors, with an additional 1.7% being ambiguous mixed or multi-label cases. This is problematic for benchmarking new models, since the estimated label error rate is now greater than state-of-the-art model accuracy error rates. Here, the implication is that high-capacity models like CNNs and transformers are now overfitting to noise. This is indeed the case for models like DiT (Li et al., 2022), which predict the "IMAGE NOT AVAILABLE" document to be an `advertisement` document due to its relative abundance in that category's training set.

(2) Ambiguities. Users of RVL-CDIP should be aware that there are many samples in RVL-CDIP that could have multiple valid document type labels. We estimate this number to be 1.7% of the RVL-CDIP test set. Like label errors, such mixed or multi-label cases make it challenging to evaluate a model effectively, as there are samples for which a model may make a wrong prediction according to the RVL-CDIP test label annotations, but in reality many of these wrong predictions could actually be reasonable.

(3) Test-Train Overlap. Practitioners and researchers should be aware that there is a high degree of overlap between the RVL-CDIP test set and the train set. Recall from Section 5 that almost a third of RVL-CDIP test samples have a near-duplicate in the training set for the same document type category, or a training sample that uses the same document template. This is undesirable, as testing models on data that is very similar to the training data can lead to "inflated" accuracy scores (Elangovan et al., 2021; Søgaard et al., 2021). Moreover, highly similar train and test splits do not facilitate the evaluation of a model's ability to generalize well to new in-domain data.

(4) Sensitive Information. There is an unsettling amount of sensitive information in the RVL-CDIP dataset, which naturally leads to information and data privacy concerns. We estimate that 7.7% of `resume` test samples contain Social Security numbers. While RVL-CDIP is already publicly available, researchers and practitioners should take care when disseminating samples or copies of RVL-CDIP. Moreover, we highlight that prior work (e.g., Carlini et al. (2021)) showed that it is possible to extract training data from machine learning models, making production deployments of models trained on RVL-CDIP an information privacy and security risk.

Suggestions for a future dataset. We suggest the development and adoption of a new benchmark for evaluating document classifiers. Several qualities of a such a benchmark would include (1) minimal label errors; (2) multi-label annotations, to allow for modeling more natural occurrences of documents; (3) minimal test-train overlap; (4) absence of sensitive information. Going beyond the points made in this paper, a new benchmark would do well to be (5) large-scale, consisting of 100+ or even 250+ document categories, to test a model's ability to handle breadth, and (6) multi-lingual, to benchmark language transfer approaches.

8 Conclusion

RVL-CDIP has been used as the *de facto* benchmark for evaluating state-of-the-art document classification models, but this paper provides an in-depth analysis of the RVL-CDIP dataset and shows that there are several undesirable characteristics of this dataset. We first provide a set of label guidelines for each RVL-CDIP category, and we use this to help us quantify the presence of errors in RVL-CDIP, finding that the RVL-CDIP test set contains roughly 8.1% label errors. We then observe that roughly a third of the test data is highly similar to the training set. Lastly we observe an unsettling amount of personally sensitive information in RVL-CDIP. Given these findings, we offer suggestions for a new document classification benchmark.

Limitations

The RVL-CDIP dataset has no official set of label guidelines, making error analyses challenging since we could not rely on pre-defined rules. For this reason we followed best practices to create annotation rules to help us in our error analysis. Detecting duplicates in RVL-CDIP is also challenging, as two documents may appear to be the same, but may have minor differences due to scanning artifacts or even different indexing labels (it appears that many of the documents have been scanned and included in IIT-CDIP more than once). Therefore we again have to rely on best judgement when labeling pairs as duplicates (or near-duplicates). Additionally, due to limitations in human resources, we were unable to exhaustively inspect all 400,000 RVL-CDIP samples for the presence of errors, ambiguities, sensitive information, etc., and thus had to rely on sampling the dataset in order to draw conclusions.

Acknowledgements

We thank Nicole Cornehl Lima, Ramla Alakraa, Zongyi Liu, Junjie Shen, Temi Okotore for help with data review, as well as the University of Michigan's Undergraduate Research Opportunity Program (UROP) for their support of these student researchers as well as support for Gordon. We also thank the anonymous EACL reviewers for their feedback.

References

Ziawasch Abedjan, Xu Chu, Dong Deng, Raul Castro Fernandez, Ihab F. Ilyas, Mourad Ouzzani, Paolo Papotti, Michael Stonebraker, and Nan Tang. 2016. Detecting data errors: Where are we and what needs to be done? *Proc. VLDB Endow.*, 9(12):993–1004.

Muhammad Zeshan Afzal, Andreas Kölsch, Sheraz Ahmed, and Marcus Liwicki. 2017. Cutting the error by half: Investigation of very deep cnn and advanced training strategies for document image classification. In *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*.

Miltiadis Allamanis. 2019. The adverse effects of code duplication in machine learning models of code. In *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*.

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R. Manmatha. 2021. DocFormer: End-to-end transformer for document understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Nicolas Audebert, Catherine Herold, Kuider Slimani, and Cédric Vidal. 2019. Multimodal deep networks for text and image-based document classification. In *Proceedings of the Conférence Nationale sur les Applications Pratiques de l'Intelligence Artificielle (APIA)*.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. 2014. Neural codes for image retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Haoli Bai, Zhiguang Liu, Ziaojun Meng, Wentao Li, Shuang Liu, Nian Xie, Rongfu Zheng, Liangwei Wang, Lu Hou, Jiansheng Wei, Xin Jiang, and Qun Liu. 2022. Wukong-Reader: Multi-modal pre-training for fine-grained visual document understanding. *arXiv preprint arXiv:2212.09621*.

Souhail Bakkali, Zuheng Ming, Mickael Coustaty, Marçal Rusiñol, and Oriol Ramos Terrades. 2022. VLCDoC: Vision-language contrastive pre-training model for cross-modal document classification. *arXiv preprint arXiv:2205.12029*.

Björn Barz and Joachim Denzler. 2020. Do we train on test data? Purging CIFAR of near-duplicates. *Journal of Imaging*, 6(6).

Frédéric Béchet and Christian Raymond. 2018. Is ATIS too shallow to go deeper for benchmarking spoken language understanding models? In *Proceedings of InterSpeech*.

Ali Furkan Biten, Rubèn Tito, Lluis Gomez, Ernest Valveny, and Dimosthenis Karatzas. 2022. OCR-IDL: OCR annotations for industry document library dataset. *arXiv preprint arXiv:2202.12985*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *Proceedings of the 30th USENIX Security Symposium*.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Derek Chong, Jenny Hong, and Christopher Manning. 2022. Detecting label errors by using pre-trained language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Michael V. Ciresi, Roberta B. Walburn, and Tara D. Sutton. 1999. Decades of deceit: Document discovery in the Minnesota tobacco litigation. *William Mitchell Law Review*, 25(2).

Roland Croft, M. Ali Babar, and M. Mehdi Kholoosi. 2023. Data quality for software vulnerability datasets. *arXiv preprint arXiv:2301.05456*.

Gabriela Csurka, Diane Larlus, Albert Gordo, and Jon Almazán. 2016. What is the right way to represent document images? *arXiv preprint arXiv:1603.01076*.

Arindam Das, Saikat Roy, Ujjwal Bhattacharya, and Swapan Kumar Parui. 2018. Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks. In *Proceedings of the 24th International Conference on Pattern Recognition (ICPR)*.

Tyler Dauphinee, Nikunj Patel, and Mohammad Rashidi. 2019. Modular multimodal architecture for document classification. *arXiv preprint arXiv:1912.04376*.

Brian Davis, Bryan Morse, Bryan Price, Chris Tensmeyer, Curtis Wigington, and Vlad Morariu. 2022. End-to-end document recognition and understanding with Dessurt. *arXiv preprint arXiv:2203.16618v3*.

Thomas Delteil, Edouard Velval, Lei Chen, and Luis Goncalves. 2022. MATrIX - modality-aware transformer for information extraction. *arXiv preprint arXiv:2205.08094v1*.

Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Javier Ferrando, Juan Luis Domínguez, Jordi Torres, Raúl García, David García, Daniel Garrido, Jordi Cortada, and Mateo Valero. 2020. Improving accuracy and speeding up document image classification through parallel systems. In *Proceedings of the International Conference on Computational Science (ICCS)*.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Catherine Finegan-Dollak and Ashish Verma. 2020. Layout-aware text representations harm clustering documents by type. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*.

Stanton A. Glantz, John Slade, Lisa A. Bero, Peter Hanauer, and Deborah E. Barnes. 1996. *The Cigarette Papers*. University of California Press.

Jiuxiang Gu, Jason Kuen, Vlad I. Morariu, Hangdong Zhao, Nikolaos Barmpalios, Rajiv Jain, Ani Nenkova, and Tong Sun. 2021. Unified pretraining framework for document understanding. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS)*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. 2015. Evaluation of deep convolutional nets for document image classification and retrieval. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*.

Keith Harrigian, Carlos Aguirre, and Mark Dredze. 2020. Do models of mental health based on social media data generalize? In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. LayoutLMv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*.

Shrinidhi Kanchi, Alain Pagani, Hamam Mokayed, Marcus Liwicki, Didier Stricker, and Muhammad Zeshan Afzal. 2022. EmmDocClassifier: Efficient multimodal document image classifier for scarce data. *Applied Sciences*, 12(3).

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2021. OCR-free document understanding transformer. *arXiv preprint arXiv:2111.15664*.

Hyunjae Kim and Jaewoo Kang. 2022. How do your biomedical named entity recognition models generalize to novel entities? *IEEE Access*.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne

David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. 2021. WILDS: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 2021 International Conference on Machine Learning (ICML)*.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, Iroro Orife, Kelechi Ogueji, Andre Niyongabo Rubungo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, Tapiwanashe Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. 2022. Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*.

Jayant Kumar, Peng Ye, and David Doermann. 2014. Structural similarity for document image classification and retrieval. *Pattern Recognition Letters*, 43.

Stefan Larson, Gordon Lim, Yutong Ai, David Kuang, and Kevin Leach. 2022. Evaluating out-of-distribution performance on document image classifiers. In *Proceedings of the Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Stefan Larson, Anish Mahendran, Andrew Lee, Jonathan K. Kummerfeld, Parker Hill, Michael A. Laurenzano, Johann Hauswald, Lingjia Tang, and Jason Mars. 2019. Outlier detection for improved data quality and diversity in dialog systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, and J. Heard. 2006. Building a test collection for complex document information processing. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. 2021a. StructuralLM: Structural pre-training for form understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Junlong Li, Yiheng Xu, Tengchao Lv, Lei Cui, Cha Zhang, and Furu Wei. 2022. DiT: Self-supervised pre-training for document image transformer. In *Proceedings of the 30th ACM International Conference on Multimedia*.

Peizhao Li, Jiuxiang Gu, Jason Kuen, Vlad I. Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu. 2021b. SelfDoc: Self-supervised document representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chuwei Luo, Guozhi Tang, Qi Zheng, Cong Yao, Lianwen Jin, Chenliang Li, Yang Xue, and Luo Si. 2022. Bi-VLDoc: Bidirectional vision-language modeling for visually-rich document understanding. *arXiv preprint arXiv:2206.13155*.

Andrey Malinin, Neil Band, Alexander Ganshin, German Chesnokov, Yarin Gal, Mark J. F. Gales, Alexey Noskov, Andrey Ploskonosov, Liudmila Prokhorenkova, Ivan Provilkov, Vatsal Raina, Vyas Raina, Denis Roginskiy, Mariya Shmatova, Panos Tigar, and Boris Yangel. 2021. Shifts: A dataset of real distributional shift across multiple large-scale tasks. *arXiv preprint arXiv:2107.07455*.

Minesh Mathew, Dimosthenis Karatzas, and C.W. Jawahar. 2021. DocVQA: A dataset for VQA on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Nicolas M. Müller and Karla Markert. 2019. Identifying mislabeled instances in classification datasets. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*.

Laura Nguyen, Thomas Scialom, Jacopo Staiano, and Benjamin Piwowarski. 2021. Skim-attention: Learning to focus via document layout. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Jingcheng Niu and Gerald Penn. 2019. Rationally reappraising ATIS-based dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Curtis Northcutt, Anish Athalye, and Jonas Mueller. 2021a. Pervasive label errors in test sets destabilize machine learning benchmarks. In *35th Conference on Neural Information Processing Systems Track on Datasets and Benchmarks*.

Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. 2021b. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research (JAIR)*, 70:1373–1411.

Qiming Peng, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Yuhui Cao, Weichong Yin, Yongfeng Chen, Yin Zhang, Shikun Feng, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2022. ERNIE-layout: Layout knowledge enhanced pre-training for visually-rich document understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Hai Pham, Guoxin Wang, Yijuan Lu, Dinei Florencio, and Cha Zhang. 2022. Understanding long documents with different position-aware attentions. *arXiv preprint arXiv:2208.08201*.

Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q. Weinberger. 2020. Identifying mislabeled data using the area under the margin ranking. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*.

Rafał Powalski, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka. 2021. Going full-TILT boogie on document understanding with text-image-layout transformer. In *Proceedings of the 16th International Conference on Document Analysis and Recognition (ICDAR)*.

Subhojeet Pramanik, Shashank Mujumdar, and Hima Patel. 2022. Towards a multi-modal, multi-task learning based pre-training framework for document representation learning. *arXiv preprint arXiv:2009.14457v2*.

Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. 2018. Revisiting Oxford and Paris: Large-scale image retrieval benchmarking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do ImageNet classifiers generalize to ImageNet? In *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

Saifullah, Stefan Agne, Andreas Dengel, and Sheraz Ahmed. 2022a. DocXClassifier: High performance explainable deep network for document image classification. *techarxiv preprint*.

Saifullah, Shoaib Ahmed Siddiqui, Stefan Agne, Andreas Dengel, and Sheraz Ahmed. 2022b. Are deep models robust against real distortions? A case study on document image classification. In *Proceedings of the 26th International Conference on Pattern Recognition (ICPR)*.

Ritesh Sarkhel and Arnab Nandi. 2019. Deterministic routing between layout abstractions for multi-scale classification of visually rich documents. In *Proceedings of the Twenty-Eigth International Joint Conference on Artificial Intelligence (IJCAI)*.

Heidi Schmidt, Karen Butter, and Cynthia Rider. 2002. Building digital tobacco industry document libraries at the University of California, San Francisco Library/Center for Knowledge Management. *D-Lib Magazine*, 8(9).

Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. 2021. Analyzing the potential of zero-shot recognition for document image classification. In *Proceedings of the 16th International Conference on Document Analysis and Recognition (ICDAR)*.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zineng Tang, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal. 2022. Unifying vision, text, and layout for universal document processing. *arXiv preprint arXiv:2212.02623*.

Kate Tasker, Taketa Rachel, Charles Macquarie, and Ariel Deardorff. 2022. Digital archives and data science: Building programs and partnerships for health sciences research. In *Handbook of Research on Academic Libraries as Partners in Data Science*.

Chris Tensmeyer and Tony Martinez. 2017. Analysis of convolutional neural networks for document image classification. In *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*.

Jiapeng Wang, Lianwen Jin, and Kai Ding. 2022a. LiLT: A simple yet effective language-independent layout transformer for structured document understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zilong Wang, Jiuxiang Gu, Chris Tensmeyer, Nikolaos Barmpalios, Ani Nenkova, Tong Sun, Jingbo Shang, and Vlad Morariu. 2022b. MGDoc: Pre-training with multi-granular hierarchy for document image understanding. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yuqiao Wen, Guoqing Luo, and Lili Mou. 2022. An empirical study on the overlapping problem of open-domain dialogue datasets. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC)*.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. LayoutLM: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*.

Cecilia Ying and Stephen Thomas. 2022. Label errors in BANKING77. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*.

Guangyu Zhu and David Doermann. 2007. Automatic document logo detection. In *Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR)*.

Guangyu Zhu, Yefeng Zheng, David Doermann, and Stefan Jaeger. 2007. Multi-scale structural saliency for signature detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Stefano Pio Zingaro, Giuseppe Lisanti, and Maurizio Gabbrielli. 2021. Multimodal side-tuning for document classification. In *Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR)*.

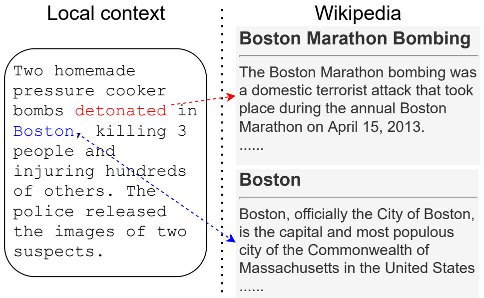
Event Linking: Grounding Event Mentions to Wikipedia

Xiaodong Yu[1] **Wenpeng Yin**[2] **Nitish Gupta**[3] **Dan Roth**[1]
[1]University of Pennsylvania [2]Penn State University [3]Google Research
`{xdyu, danroth}@seas.upenn.edu`
`wenpeng@psu.edu`
`gnnitish@gmail.com`

Abstract

Comprehending an article requires understanding its constituent events. However, the context where an event is mentioned often lacks the details of this event. A question arises: how can the reader obtain more knowledge about this particular event in addition to what is provided by the local context in the article?

This work defines Event Linking, a new natural language understanding task at the event level. Event linking tries to link an event mention appearing in an article to the most appropriate Wikipedia page. This page is expected to provide rich knowledge about what the event mention refers to. To standardize the research in this new direction, we contribute in four-fold. First, this is the first work in the community that formally defines the Event Linking task. Second, we collect a dataset for this new task. Specifically, we automatically gather the training set from Wikipedia, and then create two evaluation sets: one from the Wikipedia domain, reporting the in-domain performance, and a second from the real-world news domain, to evaluate out-of-domain performance. Third, we retrain and evaluate two state-of-the-art (SOTA) entity linking models, showing the challenges of event linking, and we propose an event-specific linking system, EVELINK, to set a competitive result for the new task. Fourth, we conduct a detailed and insightful analysis to help understand the task and the limitations of the current model. Overall, as our analysis shows, Event Linking is a challenging and essential task requiring more effort from the community. [1]



Two homemade pressure cooker bombs detonated in Boston, killing 3 people and injuring hundreds of others. The police released the images of two suspects.

Boston Marathon Bombing

The Boston Marathon bombing was a domestic terrorist attack that took place during the annual Boston Marathon on April 15, 2013.
……

Boston

Boston, officially the City of Boston, is the capital and most populous city of the Commonwealth of Massachusetts in the United States
……

Figure 1: Examples of Event linking and Entity linking. The left side is the local context, and the right side contains Wikipedia pages. Entity linking model connects the entity "Boston" to the Wikipedia page "Boston", while event linking model links the event "detonated" to the Wikipedia page "Boston Marathon Bombing", which is more relevant to the local context.

1 Introduction

Grounding is a process of disambiguation and knowledge acquisition, and is an important task for natural language understanding. Entity linking, grounding entity mentions to a knowledge base (usually Wikipedia) (Bunescu and Pasca, 2006; Mihalcea and Csomai, 2007; Ratinov et al., 2011; Gupta et al., 2017; Wu et al., 2020), has been shown important in natural language understanding tasks, such as question answering, recommendation system, dialogue generation. Despite the significant progress brought by entity linking, we argue that grounding entities may not provide enough background knowledge that is often needed to support text understanding. Consider the example, Figure 1; an entity linking model will link the entity "Boston" to the Wikipedia page "Boston" which introduces the history and culture of the city Boston. The information we can get from the page "Boston" is irrelevant to the local context. To really help understand this sentence, we need to link the event centered by the verb "detonated" to the Wikipedia page "Boston Marathon Bombing". We call this process that grounds events **Event Linking**.

In this paper, we formulate this Event Linking task for the first time, analyze the difference and challenges of the new task, and carefully design a benchmark dataset for this task. We automatically collect training data from the hyperlinks in Wikipedia, and create two evaluation sets to evalu-

---

[1] Data and code are available here: `http://cogcomp.org/page/publication_view/996`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2679–2688
May 2-6, 2023 ©2023 Association for Computational Linguistics

ate both in-domain and out-of-domain performance. For in-domain evaluation, the test data is also from hyperlinks in Wikipedia. To avoid models from overfitting, the test data is balanced with hard cases and easy cases determined by whether the event is seen in the training and by the similarity between the surface forms of event mentions and Wikipedia titles. For out-of-domain evaluation, we annotate real-world news articles across 20 years collected from New York Times. Considering the sparsity of events existed in Wikipedia, we also add "Nil" annotation to the test data, indicating that those events do not exist in Wikipedia, therefore, the model needs to tag them as "Nil".

Technically, we come up with an event linking model EVELINK that uses the entities in the local context as arguments of the event structure to better present the event mention. EVELINK outperforms two SOTA entity linking models BLINK (Wu et al., 2020) and GENRE (Cao et al., 2021), and achieves strong performance on the event linking test set, especially on seen events and easy cases, and a detailed error analysis shows the difficulties of the new task and the limitation of the current model.

To conclude, our contributions are four-fold: (i) We formulate the task Event Linking. (ii) We collect training data for this task, and design both in-domain and out-of-domain test data, with a balanced ratio of hard cases and easy cases to ensure the dataset quality. (iii) Our proposed approach EVELINK shows promising performance in experiments, which sets a competitive result for future works. (iv) Our in-depth analysis provides a better understanding of this new problem, the challenges in different domains, and the new approach.

2 Grounding Events in Wikipedia

Given an article and an event mention m in it, event linking tries to find a title t, from **all** the English Wikipedia titles (around 5m titles), to provide the best explanation of m. Event mention is defined as verb or nominal that refers to an event. A correct title is defined as follows: as long as a Wikipedia page is about this event, or any subsection of the page introduces this event, we regard its title as the correct one. In this paper, all the models assume gold event mentions are given. For each event mention, a system is expected to label it with the correct Wikipedia page or a "Nil" tag if the event does not exist in Wikipedia. Accuracy is adopted as the official evaluation metric.

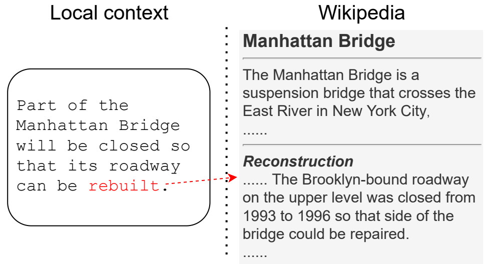


Figure 2: Example of event mentions that only exist in the subsection of a Wikipedia page. The event "rebuilt" does not have its own page, but is mentioned in the subsection of the page "Manhattan Bridge".

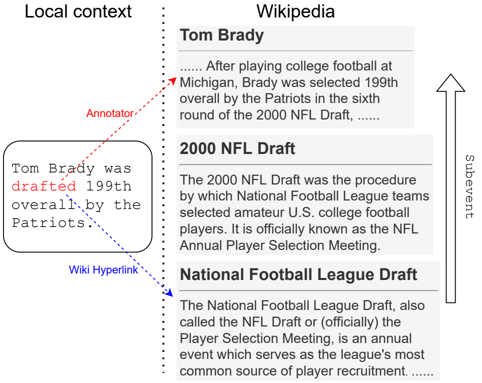


Figure 3: Example of hierarchical events. Event "draft" of Tom Brady is mentioned in the page "Tom Brady", and is also a sub-event of "2000 NFL Draft", which is again a sub-event of "National Football League Draft".

Event Linking vs. Entity Linking. Relatedness: (i) They both link an object (event/entity) from an article to Wikipedia; (ii) Some events, such as "World War II", are entities; in this case, two tasks are the same. Distinctions: (i) Entities are mostly consecutive text spans. Events, in contrast, are more structured objects, consisting of a trigger and a couple of arguments. An event trigger is mostly a general verb, which may not refer to a specific event by its own without knowing event arguments. More complex structures in events make event linking a more challenging task and require a deeper understanding of the local context; (ii) Unlike entities with a large coverage in Wikipedia, many events do not have a record in Wikipedia. Considering the sparsity, we require models to tag event mentions that do not exist in Wikipedia as "Nil".

Why Event Linking? Except for some events that are also entities, generally speaking, events are information units of larger granularity. As shown in Figure 1, a better comprehension of events, such as through linking to Wikipedia, is expected to facilitate the text understanding more.

Challenges specific to Event Linking. (i) The correct title for some event mentions may not be unique. The same event could be introduced in several pages. For example, "Invasion of Poland" and "Occupation of Poland (1939–1945)" both introduce the event that German Army invaded Poland in 1939. How to decide the ground truth set and how to evaluate in this situation are not trivial.

(ii) Events may only exist in the subsection of the Wikipedia page. Only a limited number of famous events have their own pages, while many other relatively infamous events only exist in the subsection of some pages. Considering the example in Figure 2, the event "rebuilt" of the Manhattan Bridge does not have its own Wikipedia page, but it is mentioned in the subsection "Reconstruction" of the page "Manhattan Bridge". Linking these events requires a model to understand the whole page instead of just encoding the first paragraph.

(iii) Events have a hierarchical structure. Events at larger granularity consist of many sub-events, and these sub-events may have their own Wikipedia pages, or just be mentioned in the pages of the large events. Ideally, the model should always link the event mention to the most appropriate page. If the sub-event page exists, then link to the sub-event page. Otherwise, link to the page of the large event. However, the term "appropriate" here could be unclear because of the event hierarchy. As Figure 3 shows, the Wikipedia page "Tom Brady" is most specific to the event "drafted". On the other hand, draft of "Tom Brady" is a sub-event of "2000 NFL Draft", which is further a sub-event of "National Football League Draft". Annotators prefer to link this event to "Tom Brady", while Wikipedia hyperlinks link the event to "National Football League Draft". The hierarchy of events makes the standard of the correct title inconsistent.

3 Related Work

Entity Linking. As described in the previous section, entity linking has been extensively studied for many years (Bunescu and Pasca, 2006; Mihalcea and Csomai, 2007; Ratinov et al., 2011; Gupta et al., 2017; Wu et al., 2020; Cao et al., 2021). Though both of entity linking and event linking could be regarded as a task linking document contents to a knowledge base, we argue that entity linking is more about linking text span, while event linking is more about linking an event structure, centered by a predicate, which is more challenging because the predicate span is usually a general verb. In the experiment section, we show that just retraining the entity linking model on event linking data without considering the event structure does not perform well. Humeau et al. (2019) and Wu et al. (2020) use a bi-encoder/cross-encoder architecture to train the candidate generation/ranking model respectively for entity linking. Considering the structure of events that entities do not have, EVELINK extends their model by adding structure information to the event mention representation.

"Event Linking". We note that the term "event linking" has been once used in the literature (Nothman et al., 2012; Krause et al., 2016). However, these works are essentially performing cross-document event coreference: determine if a given event mention refers to another event mention (in the same or another document). We, on the other hand, link an event mention to a Wikipedia concept with a different purpose: acquiring external knowledge about the event which is often beyond what we can obtain from the local context. Our definition of event linking can not only improve the understanding of the article, but also pave the way for the intensively-studied event coreference and other event relation identification problems.

Data. Eirew et al. (2021) collect training data from Wikipedia hyperlinks for event coreference, while we use similar methods to collect data for event linking. In this work, we use the FIGER type of the title to find event titles, while Eirew et al. (2021) use the Wikipedia infobox. There also exists some other event knowledge bases, such as EventKG (Gottschalk and Demidova, 2018). Because we use hyperlinks in Wikipedia as training data resource, and we do not limit the candidate space to be event titles only, in this work we only focus on linking event mentions to Wikipedia, and the candidate space is all the Wikipedia titles.

4 Data Construction

We collect training data and in-domain test data from Wikipedia automatically, and manually annotate a test set in the news domain for out-of-domain evaluation purpose. Table 1 lists some data examples, and Table 2 shows detailed statistics.

4.1 Wikipedia

We first collect all hyperlinks ($hypertext$, $title$) in Wikipedia text, which links a hypertext to a Wikipedia title. Then, we map the FreeBase type

	Event mention in local context	Wikipedia title
Wiki	At the start of the wartime 1940s , he had four releases.	World War II
	Henry Louis Gates, a black Harvard University professor who was arrested after police mistakenly thought he was breaking into his own home in Cambridge, Massachusetts.	Henry Louis Gates arrest controversy
	Ibrox hosted four Scotland games in the first phase, starting with a 1994 World Cup qualifier against Portugal in October 1992.	1994 FIFA World Cup qualification
NYT	The Nets offered Sam Cassell and a first-round draft pick for Marbury.	Sam Cassell
	A man who killed his former wife, a bartender and a cook in 1984 was executed by injection early today.	Godinez v. Moran
	A 45-year-old fashion photographer was shot and killed in his West Village apartment yesterday morning, the police said.	Nil

Table 1: Data examples. The upper part is data collected from Wikipedia hyperlinks. The lower part is annotated New York Times (NYT) paragraphs. Event mentions are highlighted in red.

	Train	Dev	Test	
	Wiki	Wiki	Wiki	NYT
Verb	33,213	8,346	9,633	1,319
Seen Event	-	1,814	2,913	0
Unseen Form	-	2,585	3,828	75
Unseen Event	-	3,947	2,892	435
Nil	-	-	-	809
Nominal	33,213	8,346	9,633	443
Hard	-	4173	4817	244
Easy	-	4173	4817	15
Nil	-	-	-	184
Total	66,426	16,692	19,266	1,762

Table 2: Wikipedia and New York Times (NYT) data statistics. NYT is only for testing.

of Wikipedia titles to FIGER types (Ling and Weld, 2012), and all titles with a type "Event" are regarded as event titles. All the hypertexts linked to these event titles are regarded as event mentions.

Because same event mentions in one Wikipedia page are hyperlinked only once, and editors tend to hyperlink more nominal mentions than verb mentions, verb mentions are highly limited in Wikipedia. To balance the size of verbs and nominals, we use SpaCy Part-of-Speech model[2] to keep all verb mentions, and sample the same size of nominals. To prevent models from overfitting, we design hard and easy cases for verbs and nominals:

Verbs: We classify each verb mention mainly by whether the surface form (S) of the verb is seen in training data, and whether the gold event title (T) is seen in training data. If both S and T are seen in training data, we call it **Seen Event**. If T is seen in training data, but S is new, we call it **Unseen Form**. If T is never seen in training data, we call it **Unseen Event**. Under this setting, "Seen Event" is regarded as easy cases, and the other two are hard cases. Because of the limited size of verb mentions, all the event titles with fewer than or equal to 5 verb mentions are used as "Unseen Event".

Nominals: We classify each nominal mention mainly by its surface form similarity to the gold title. We calculate the Jaccard similarity between the nominal mention and the gold title by taking 3 grams of the surface form. If the similarity is lower than 0.1, we think it is a **hard nominal**; otherwise, it is an **easy nominal**. Then we sample same numbers of hard and easy cases.

4.2 New York Times

We sample 2,500 lead paragraphs from The New York Times Annotated Corpus (Sandhaus, 2008), which contains New York Times articles from 1987 to 2006. We first use an off-the-shelf verb and nominal SRL model[3] to extract event mention candidates, and then we use Amazon Mechanical Turk to annotate the corresponding Wikipedia title of the predicted mention candidates. To ensure the quality of the annotation, we design our annotation process in two rounds:

First round. Annotators need to answer whether they think the predicted mention is an event or not. If they think it is an event, then they need to find the corresponding Wikipedia title, otherwise submit "Nil". Each mention is annotated by three annotators. If all of them submit "Nil", we include this event mention as a "Nil" example in the final test data. *To prevent annotators from simply submitting "Nil", 10% of the event mentions are the relatively easy cases from the Wiki data and we know their answers. We randomly insert them into the input data for AMturk (i.e., annotators are unaware of that) to evaluate the accuracy of the annotator. Only the annotation from annotators with an accuracy higher than 90% will be accepted.*

---

[2] `https://spacy.io/usage/linguistic-features#pos-tagging`

[3] `https://cogcomp.seas.upenn.edu/page/demo_view/SRLEnglish`

Second round. This round verifies the annotated results in the first round. Each mention with the annotated title is verified by another three annotators. They need to read the page, and figure out whether it introduces the mention. If the majority vote for "yes", we include it in the final test data. Because of majority voting, some annotations that not all the annotators agree would be included. The inter-agreement is 63.74 Fleiss' kappa.

4.3 Domain Analysis

Event linking in the news domain is more challenging than that in the Wikipedia domain because of the following reasons:

(i) News articles describe an event at a different granularity as how Wikipedia does, usually with more details. For example, here is a piece of news about "Iraq_War": "A contractor working for the American firm Kellogg Brown & Root was wounded in a mortar attack in Baghdad." The event "wounded" here is a very small event in Iraq War, but it is what daily news would report. On the other hand, the event mention that links to "Iraq_War" in Wikipedia domain is: "When touring in Europe, the US went to war in Iraq." The different granularity in representing events makes the task slightly different in two domains. Event linking in Wikipedia domain is more like event coreference, while event linking in news domain is mixed with more subevent relation extraction.

(ii) As analyzed in Section 2, event linking is challenging because some event mentions may only exist in the subsection of the correct page, and the correct title is not consistent because of the event hierarchy. However, these problems mainly happen in the news domain. First of all, the Wikipedia hyperlinked mentions usually have their own pages instead of just existing in subsections. In news domain, we annotate events that only exist in subsections of a Wikipedia page. Second, in Wikipedia domain, the gold title of same event mentions is usually consistent. For example, all of the event mentions "drafted" of football players link to "National Football League Draft" instead of the page of the specific player. However, the annotation standard of NYT is not always consistent with Wikipedia hyperlinks. For example, annotators would link event mentions about sports player draft to the page of the specific player instead of the general concept page "National Football League Draft". These problems make data annotation and model evaluation in news domain very challenging.

Because of the reasons claimed above, we think that, for some cases in news domain, the correct answer is multiple titles instead of just one title. Ranking the annotated title to the second place may be because the top one is also correct. To relax the evaluation metric here, for news domain, we also report the number of Accuracy@5, which means that if the annotated title is ranked in the top 5 candidates, we think it is correct.

5 Model

In this section, we propose EVELINK as the first event linking model. We first introduce the representation of event mentions and event titles in Section 5.1, and then introduce the model architecture in Section 5.2.

5.1 Event Representation

A key difference between entity and event is that the context of an entity is more diverse than the context of an event. For example, when the entity "China" is mentioned in a sentence, it is unclear what entities or what events probably would also be mentioned together. However, if a verb like "invade" is used to represent the event "Battle of France" in a sentence, it is very likely that entities like "Germany", "Italy" and "France" will also be mentioned. This shows that an event is defined by its arguments, and these arguments, with a large chance, will also be mentioned in the local context because the verb itself cannot refer to any event. Given this observation, we think that the entities in the local context of the event mention should overlap with the entities in the correct Wikipedia page, and these entities can be used to help the model better represent events. To embed these entities information explicitly to the event representation, we use similar method as how Vyas and Ballesteros (2021) embed entity attributes information to the entity representation.

Event mentions: To represent event mentions in local context, we first use an off-the-shelf Named Entity Recognition model [4] trained on 18-type OntoNotes dataset (Weischedel et al., 2013) to extract the entities around the event. We simply define the context window by 500 characters around the event mention. After

---

[4] `https://cogcomp.seas.upenn.edu/page/demo_view/NEREnglish`

Models	Verb				Nominal			Verb + Nominal
	Seen	Unseen Form	Unseen	Overall	Hard	Easy	Overall	
Glove	23.70	16.57	14.89	18.22	3.08	84.60	43.88	30.98
BM25	32.17	47.41	61.86	47.14	22.54	33.22	27.88	37.51
BLINK-Entity	88.91	69.64	62.31	73.37	67.93	95.20	81.57	77.42
BLINK-Event	80.88	85.84	84.54	83.95	79.39	89.10	84.24	84.10
GENRE-Entity	92.41	73.48	59.13	74.90	79.84	**96.53**	88.19	81.54
GENRE-Event	**98.80**	87.30	58.85	82.24	85.34	94.64	89.99	86.12
EveLink	93.99	**92.74**	**93.91**	**93.47**	**89.79**	95.52	**92.65**	**93.06**

Table 3: Recall on Wikipedia Test. "Seen" means both the surface forms of the mention and the gold title are seen in training. "Unseen Form" means the surface form of the mention is new, but the gold title is seen in training. "Unseen" means that the gold title is unseen in training. BLINK-Entity is the original BLINK model trained on entity linking dataset. BLINK-Event is trained on the new event linking dataset. More details in Section 6

Models	Verb				Nominal			Verb + Nominal
	Seen	Unseen Form	Unseen	Overall	Hard	Easy	Overall	
Prior	62.21	2.38	1.24	38.81	34.65	85.99	61.65	54.79
BLINK-Entity	64.13	48.56	45.92	52.48	46.79	88.27	67.53	60.00
BLINK-Event	77.72	69.78	62.72	70.06	62.59	82.29	72.44	71.25
GENRE-Entity	75.04	57.00	44.85	58.81	65.29	**90.91**	78.10	68.45
GENRE-Event	**95.50**	73.80	45.16	71.76	72.60	88.04	80.32	76.04
EveLink	91.21	**80.30**	**78.08**	**82.93**	**75.90**	89.70	**82.80**	**82.87**

Table 4: Accuracy on Wikipedia Test.

predicting all the entities e_i with their type t_i, we represent the event mentions by:

$$r_1 = \text{[CLS] ctxt}_l \text{ [M}_s\text{] m [M}_e\text{] ctxt}_r \quad (1)$$

$$r_2 = [\mathrm{t}_{1_s}]\ \mathrm{e}_1\ [\mathrm{t}_{1_e}] \cdots [\mathrm{t}_{n_s}]\ \mathrm{e}_n\ [\mathrm{t}_{n_e}] \quad (2)$$

$$r_m = r_1 \text{ [SEP] } r_2 \text{ [SEP]} \quad (3)$$

where m, ctxt_l, ctxt_r, e_i are tokens of event mention, the context on the left of the mention, the context on the right of the mention and predicted entities. $[\mathrm{M}_s]$ and $[\mathrm{M}_e]$ are special tokens to tag the start and end of the event mention. $[\mathrm{t}_{i_s}]$ and $[\mathrm{t}_{i_e}]$ are special tokens to tag the start and end of the entity whose type is t_i. r_m is the final representation of event mentions.

Title: To represent Wikipedia titles, since important entities are already hyperlinked in the page contents, we take the first ten hyperlinked spans as entities, and represent the title by:

$$r_3 = \text{[CLS] title [TITLE] description} \quad (4)$$

$$r_4 = \mathrm{h}_1 \text{ [SEP] } \mathrm{h}_2 \text{ [SEP] } \cdots \text{ [SEP] } \mathrm{h}_n \quad (5)$$

$$r_t = r_3 \text{ [SEP] } r_4 \text{ [SEP]} \quad (6)$$

where title, h_i and $\mathrm{description}$ are tokens of the title, hyperlinked spans, and the content of the Wikipedia page. We simply take the first $2,000$ characters as the description. [TITLE] is the special token to separate the title and the description. r_t is the final representation of Wikipedia titles.

5.2 Model Architecture

Similar to Wu et al. (2020), we first use a bi-encoder architecture to efficiently generate candidates, and use a cross-encoder architecture, which requires more computations, to rank the candidates.

Candidate Generation. We use a bi-encoder architecture to train the candidate generation model. We use two independent BERT transformers (Devlin et al., 2019) to encode the representation of event mentions r_m and Wikipedia titles r_t, and use the output of the two [CLS] tokens in r_m and r_t as the event mention vector v_m and the title vector v_t. Then, we maximize the dot product between the vectors of event mentions v_m and the correct title v_t in a batch with randomly selected negatives. At inference time, representations of all the titles are cached, and for each event mention, we calculate the dot products between its representation and the representation of all the titles, and titles with higher scores will become candidates.

Candidate Ranking. For each event mention, we take 30 candidates from the candidate generation model as the training data for the ranking model, and use a cross-encoder architecture to train the candidate ranking model. We concatenate the representation of event mentions r_m and titles r_t, use one BERT transformer to encode the concatenated representation, and use the output of the [CLS] token as the final vector v. Then we maximize the dot product between the vector v of

Models	Verb				Nominal			Verb + Nominal
	Seen	Unseen Form	Unseen	Overall	Hard	Easy	Overall	
Glove	-	0.00	0.70	0.60	0.00	33.33	1.66	0.94
BM25	-	28.38	41.74	39.80	45.27	31.25	44.40	41.35
BLINK-Entity	-	4.00	6.67	6.27	7.79	60.00	10.81	7.80
BLINK-Event	-	35.14	37.39	37.06	37.45	75.00	39.77	37.97
GENRE-Entity	-	18.92	9.86	11.18	9.88	62.50	13.13	11.83
GENRE-Event	-	**56.77**	17.66	23.33	21.81	31.25	22.39	23.02
EveLink	-	52.70	**59.40**	**58.43**	**51.03**	**93.75**	**53.68**	**56.83**

Table 5: Recall on New York Times data. Because "Nil" mentions do not have the Wikipedia title, the Recall is only evaluated on the mentions that exist in Wikipedia.

Models	Verb			Nominal			Verb + Nominal	
	Unseen Form	Unseen	Overall	Hard	Easy	Overall	Accu@5	Accu@1
Prior	0.00	0.00	0.00	0.00	6.25	0.39	0.52	0.13
BLINK-Entity	1.33	2.76	2.55	4.92	33.33	6.56	11.44	3.90
BLINK-Event	17.57	5.28	7.06	11.11	37.50	12.74	17.04	8.97
GENRE-Entity	8.11	5.73	6.08	3.29	31.25	5.02	11.83	5.72
GENRE-Event	**39.19**	8.03	12.55	7.82	31.25	9.27	23.02	11.44
EveLink	28.37	**13.07**	**15.29**	**14.81**	**43.75**	**16.60**	**29.13**	**15.73**

Table 6: Accuracy on New York Times data without Nil. Only event mentions that exist in Wikipedia are given. Accu@5 means the correct title is ranked top 5. Accu@1 means the correct title is top 1.

the correct title and an additional linear layer W.

6 Experiments

In this section, we evaluate the in-domain performance on Wiki test set and the out-of-domain performance on NYT test set, and conduct an error analysis. Implementation details in Appendix A.

Baselines. Since there is no existing event linking system, we have to compare with previous entity linking systems. In this paper, we mainly compare our system with two SOTA entity linking models BLINK (Wu et al., 2020) and GENRE (Cao et al., 2021). To make a fair comparison, BLINK and GENRE have the following two setups:

BLINK/GENRE-Entity: Since a large portion of event mentions are nominals, which is also a kind of entity, it would be interesting to see how a SOTA entity linking system performs for event linking. Therefore, we test the BLINK/GENRE model pretrained specific to entity linking directly. Please note that the size of entity linking training data is 9 million, which is much larger than the size of event linking training data 66k.

BLINK/GENRE-Event: It adopts the same algorithm with the original BLINK/GENRE system, but is trained on our event linking training set.

For all the experiments, BLINK-Entity retrieves 10 candidates from candidate generation, and both BLINK-Event and EveLink retrieves 100 candidates from candidate generation. These numbers are tuned on dev data. GENRE is a generation model, which does not use the same pipeline of candidate generation and ranking. We follow the original setting to use the beam search with 5 beams.

Besides SOTA entity linking systems, we also evaluate the performance of BM25, Glove vector cosine similarity between event mention and titles (Pennington et al., 2014) and prior distribution. Because event mentions are limited in Wikipedia, to fairly estimate the prior distribution of the event titles, we only evaluate event mentions that appear at least 10 times in Wikipedia.

In-domain experiment on Wikipedia. We evaluate EveLink on the Wikipedia test set as the in-domain performance. We report the recall of candidate generation in Table 3, and the accuracy of candidate ranking in Table 4. As shown in Table 3 and Table 4, EveLink outperforms baseline models by a large margin, 6.94 points in Recall and 6.38 points in Accuracy. EveLink also achieves a high performance on seen verbs and easy nominals, around 90 accuracy, but a relatively low performance on other hard cases, which leaves a large space for future works to further improve.

Out-of-domain experiment on News. We evaluate EveLink on the NYT test set as the out-of-domain performance. In Table 5, we evaluate the recall of candidate generation. Because "Nil" mentions do not have correct titles in Wikipedia, we only evaluate the the recall of event mentions that exist in Wikipedia. Though the recall of

Models	Verb				Nominal				Verb+Nominal	
	Unseen Form	Unseen	Nil	Overall	Hard	Easy	Nil	Overall	Accu@5	Accu@1
BLINK-Entity	2.7	1.15	79.85	49.51	1.23	25.00	63.04	27.77	57.26	44.04
BLINK-Event	12.16	1.61	90.85	56.94	4.53	37.50	88.59	40.63	58.45	52.84
EveLink	**17.57**	**4.59**	**93.08**	**59.59**	**7.00**	**43.75**	**89.67**	**42.66**	**59.70**	**55.33**

Table 7: Accuracy on New York Times data with Nil. We simply predict all the mentions with a probability lower than 50 to Nil.

Models	Wiki Test	NYT (no Nil)	NYT
EveLink	82.87	15.73	55.33
- type	81.39	11.70	55.96
- entities	71.25	8.97	52.84

Table 8: Ablation Study of EveLink

EveLink is much higher than the recall of other baseline models (56.83 vs. 37.97), the recall drops significantly compared with the recall on Wikipedia test set (56.83 vs. 93.06). In Table 6, we evaluate the accuracy on the event mentions that exist in Wikipedia, which is the same setting as the experiments in the Wikipedia domain, and again the accuracy drops significantly from 82.87 to 15.73. Even if we accept 5 predictions instead of just one to solve the multiple correct titles problem, the Accuracy@5 is 29.13, which is still low. Detailed error analysis is in Section 6. In Table 7, we evaluate the accuracy of all the event mentions, including Nil. Because we do not have Nil examples in training data and development data, we simply predict all the event mentions with probability lower than 50 to "Nil", and leave better solutions to future works. GENRE is not tested for Nil mentions because it is unclear how to get its prediction probability.

Analysis. We wonder following questions:

$\mathcal{Q}_1$**:** Where the gain comes from, compared with the BLINK system?

We do ablation study in Table 8. Explicitly adding entities to the event representation boosts the performance by 10.14 accuracy on Wiki test data and 2.73 accuracy on NYT data. Adding entity types further improves the performance by 1.48 accuracy on Wiki and 4.03 accuracy on NYT data.

$\mathcal{Q}_2$**:** Error patterns of EveLink

We collect several error patterns that are common in both domains, and patterns that are mostly in news domains. Error patterns of both domains:

(i): Repeating events. In the errors, we find many repeating events, like award ceremonies or sports games, that would happen every several years, and the model usually cannot find the correct year of the event if the year is not explicitly mentioned in the context. For example:

> In 1995, his debut season, Biddiscombe made two appearances, ··· The following year he earned a Rising Star nomination for his performance···

In this example, the gold event is "1996 AFL Rising Star", and the prediction is "1998 AFL Rising Star", though there is a temporal hint (the following year of 1995 is 1996) to indicate that the correct answer should be the award in 1996. There are many similar errors when linking awards or games, which shows that a deeper temporal understanding is necessary for future works.

(ii): Unrelated context. EveLink replies on the surrounding entities to link the event mentions, however, the context is not always related and surrounding entities cannot help linking. For example:

> Returning to his country at the end of the conflict and another begun, Barinaga rejected an offer from Athletic Bilbao, moving to Real Madrid instead.

In this example, the gold event is "World War II", but the prediction is "1939–40 La Liga". All the entities, like "Barinaga", "Athletic Bilbao" and "Real Madrid", are about football, which is unrelated to the war. To link to the correct page, the model needs to know the second conflict of Barinaga's country, which indicates that only using the local context maybe not enough.

Error patterns specific to news domain:

(i): Subsection events. Some events do not have their own pages, and are only introduced in the subsections of other pages. For example:

> The Philippine government lifted its five - year ban on the return of Imelda Marcos today and said the widow of the late President Ferdinand Marcos was free to come home from exile in the United States.

In this example, the return of Imelda Marcos is introduced in the subsection "Return from exile (1991–present)" of the page "Imelda Marcos". However, we only use the first 2,000 characters of the page contents to represent the title "Imelda Marcos", which has no information about the return from exile. A document-level representation may be a potential solution for future works.

(ii): Sub-events. Some events are sub-events of other larger events. For example:

> Stepping in at the 11th hour, Hillary Rodham Clinton will campaign in Florida on Saturday for her brother, Hugh Rodham, in his bid for a United States Senate seat.

This event is a sub-event of “1994 United States Senate election in Florida”, which has different event arguments, so the names in the local context do not overlap with the names in the page.

In this work, we discuss many challenges of the task in different domains, but EVELINK cannot address all of them. We leave them to future works.

7 Conclusion

In this work, we formulate Event Linking, a challenging but essential task, with a carefully designed Wikipedia dataset and NYT test set, and propose an event linking model EVELINK for future works.

8 Limitations

In this section, we discuss limitations of our work.

- We only focus on event linking to English Wikipedia in this work. We leave multilingual event linking to future works.
- The performance of EVELINK on hard cases is still low, for example events that only exist in the subsection of Wikipedia page.
- In this work, we simply predict all the mentions with a prediction probability that is lower than 50 to “Nil”. We leave better solutions to future works.

Acknowledgments

This work was supported by Contract FA8750-19-2-1004 with the US Defense Advanced Research Projects Agency (DARPA), the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the BETTER Program, and a Focused Award from Google. Approved for Public Release, Distribution Unlimited. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. *ArXiv*, abs/2010.00904.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Alon Eirew, Arie Cattan, and Ido Dagan. 2021. WEC: Deriving a large-scale cross-document event coreference dataset from Wikipedia. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2498–2510, Online. Association for Computational Linguistics.

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *Proceedings of the Extended Semantic Web Conference (ESWC 2018)*. Springer.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2681–2690, Copenhagen, Denmark. Association for Computational Linguistics.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *arXiv preprint arXiv:1905.01969*.

Sebastian Krause, Feiyu Xu, Hans Uszkoreit, and Dirk Weissenborn. 2016. Event linking with sentential features from convolutional neural networks. In *Proceedings of CoNLL*, pages 239–249.

Xiao Ling and Daniel S Weld. 2012. Fine-Grained Entity Recognition. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proc. of the ACM Conference on Information and Knowledge Management (CIKM)*.

Joel Nothman, Matthew Honnibal, Ben Hachey, and James R. Curran. 2012. Event linking: Grounding event reference in a news archive. In *Proceedings of ACL*, pages 228–232.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1375–1384, Portland, Oregon, USA. Association for Computational Linguistics.

Evan Sandhaus. 2008. The new york times annotated corpus ldc2008t19. *Web Download*.

Yogarshi Vyas and Miguel Ballesteros. 2021. Linking entities to unseen knowledge bases with arbitrary schemas. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 834–844, Online. Association for Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

A Implementations

We use 4 Nvidia RTX A6000 48GB GPUs for model training and evaluation. For both candidate generation model and candidate ranking model, we train 10 epochs with learning rate $1\mathrm{e}^{-5}$, and use BERT-large-uncased as the pretrained language model (Devlin et al., 2019). The maximum tokens of both event mention representation and Wikipedia title representation are 256. Top-K is chosen from [5, 10, 30, 50, 70, 100], and tuned on development data. The Glove version we use is "glove-wiki-gigaword-100".

The Wikipedia dump version we use is 2020/03/01, which is also released to public with our annotated data.

B Data Annotation

We require all the annotators from Amazon Mechanical Turk to be English speaker, and with an acceptance rate higher than 95%. All the annotators are English native speakers and are paid more than 10 US dollars per hour.

SwitchPrompt: Learning Domain-Specific Gated Soft Prompts for Classification in Low-Resource Domains

Koustava Goswami[1,2*], **Lukas Lange**[2], **Jun Araki**[3], **Heike Adel**[2]

[1] Adobe Research Bangalore, India
[2] Bosch Center for Artificial Intelligence, Renningen, Germany
[3] Bosch Research North America

koustavag@adobe.com, {Lukas.Lange,Heike.Adel}@de.bosch.com, Jun.Araki@us.bosch.com

Abstract

Prompting pre-trained language models leads to promising results across natural language processing tasks but is less effective when applied in low-resource domains, due to the domain gap between the pre-training data and the downstream task. In this work, we bridge this gap with a novel and lightweight prompting methodology called *SwitchPrompt* for the adaptation of language models trained on datasets from the general domain to diverse low-resource domains. Using domain-specific keywords with a trainable gated prompt, *SwitchPrompt* offers domain-oriented prompting, that is, effective guidance on the target domains for general-domain language models. Our few-shot experiments on three text classification benchmarks demonstrate the efficacy of the general-domain pre-trained language models when used with *SwitchPrompt*. They often even outperform their domain-specific counterparts trained with baseline state-of-the-art prompting methods by up to 10.7% performance increase in accuracy. This result indicates that *SwitchPrompt* effectively reduces the need for domain-specific language model pre-training.

1 Introduction

Recent work showed promising results on different natural language processing tasks when prompting pre-trained language models (LMs) instead of fine-tuning them, especially in low-resource settings (Schucher et al., 2022). Most LMs which are publicly available have been trained on general-domain corpora (Devlin et al., 2019; Liu et al., 2019; Goyal et al., 2021), such as Wikipedia or the BooksCorpus (Zhu et al., 2015). Applying them to tasks from a special domain results in a domain gap.

For some special domains, domain-specific LMs exists, e.g., Clinical BERT (Alsentzer et al., 2019) or BioBERT (Lee et al., 2019). However, pre-training deep language models requires large amounts of text data.[1] While we can assume the availability of large-scale text data in the general domain, this assumption might not hold for low-resource domains, making the creation of domain-specific LMs challenging. Moreover, training different models for each and every new domain might be inefficient from a computation point of view.[2] Even if there are domain-specific texts and computational resources available, domain-specific LMs may not be able to get sufficient domain-oriented guidance through traditional prompting techniques because, for instance, domain-specific knowledge might be represented by a large and diverse vocabulary. As a result, both prompting LMs from the general domain and from a special domain might be ineffective, especially in low-resource settings.

Motivated by these challenges, we explore domain-oriented prompts and propose a novel and lightweight method, *SwitchPrompt*, to effectively retrieve domain-specific knowledge from pre-trained LMs. It extends the sequence of soft-prompting vectors with a sequence of vectors representing domain-specific keywords and introduces gates to allow the model to dynamically switch between a general soft prompt and a domain-specific one based on the input sentence. We hypothesize that this approach helps to effectively retrieve domain-specific knowledge from pre-trained LMs.

Our experiments on benchmark datasets from different domains indicate that *SwitchPrompt* outperforms different state-of-the-art prompting methods. It improves results in both in-domain and out-of-domain settings, effectively reducing domain gaps among pre-training and downstream task data. We find that it is especially suitable for low-

---

[*] Research work conducted during internship at Bosch Center for Artificial Intelligence. Contact: koustavag@adobe.com

[1] Clinical BERT (Alsentzer et al., 2019), for instance, was trained on the MIMIC-III $v1.4$ database (Johnson et al., 2016) which includes 2 million notes.

[2] Lee et al. (2019) used eight NVIDIA $V100$ GPUs for 23 days to train the BioBERT.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2689–2695
May 2-6, 2023 ©2023 Association for Computational Linguistics

resource settings (both little data and little computational resources) as it neither requires pre-training domain-specific LMs nor fine-tuning LMs for the downstream task. The code base for *SwitchPrompt* is available online.[3]

2 Related Work

Language model prompting. Prompting pre-trained LMs has been shown effective for different NLP tasks (Brown et al., 2020). While discrete prompts are intuitively understandable, their design requires non-trivial human involvement and they may be outperformed by fine-tuning (Shin et al., 2020; Jiang et al., 2020; Gao et al., 2021). Recent studies address this issue by optimizing so-called soft prompts in continuous space. Li and Liang (2021a) propose prefix tuning that optimizes prefix activations prepended to the input layer and each layer in the encoder stack. Lester et al. (2021) prepend trainable continuous embeddings to the original sequence of input word embeddings. Liu et al. (2021) propose P-tuning in which an LSTM encoder captures the sequential representations of the soft prompts. Liu et al. (2022) use a deep prompting methodology which injects prompts at each layer of the pre-trained LM. In contrast to those prior works, we propose a new soft prompting method that is especially suited for low-resource domains.

Language models in special domains. Most popular pre-trained LMs are trained on data from the general domain. Tailoring an LM towards a domain can be done via domain-specific pre-training from scratch (i.a., Alsentzer et al., 2019; Lee et al., 2019) or adaptation of an existing model to target domain data with continued pre-training (i.a., Gururangan et al., 2020; Xu et al., 2020; Lange et al., 2022). We refer to the survey of Hedderich et al. (2021) for more information on language model adaptation for low-resource domains. In this paper, we take a different approach and investigate to which extend LMs that were pre-trained on the general domain can be prompted for domain knowledge in few-shot settings as this requires only a minimal amount of domain-specific data.

3 Method

We now present *SwitchPrompt*, and give an example of an architecture in which it can be applied. In our architectural setup, the underlying pre-trained language model is fixed (i.e., not fine-tuned).

3.1 Domain-Specific Soft Prompts

The motivation behind our proposed prompts $P \in \mathbb{R}^{l \times e}$ is to allow the model to dynamically switch between a general-domain prompt P_g and a domain-specific prompt P_d in order to retrieve different kinds of knowledge from the pre-trained LM based on the current input, where e is the embedding dimension, and l is the length of the prompt (i.e., number of soft-prompt vectors). We implement this with a sigmoid-based gating function:

$$P = g_1(\mathsf{pad}(P_g)) + (1 - g_1)P_d \quad (1)$$

$$g_1 = \sigma(w_1^\top s_{\text{input}}) \quad (2)$$

where pad is a function that pads P_g to length l. The prompts P_g and P_d will be defined in Equations 3 and 6, respectively. Gate g_1 is calculated based on the representation of the input sentence s_{input} (in our case the representation of the [CLS] token when feeding the input sentence into the pre-trained LM) and a weight vector $w_1 \in \mathbb{R}^e$ that is randomly initialized and updated during training.

The general-domain prompt is implemented as a sequence of randomly initialized vectors $v_1, \ldots, v_m$ that are trained on the downstream task, similar to Lester et al. (2021) and Liu et al. (2022):

$$P_g = [v_1; v_2; \ldots; v_m] = V_m \quad (3)$$

The sequence length m is a hyperparameter of the model and '[;]' denotes concatenation.

The domain-specific prompt is designed to incorporate a sequence of vectors $K_n = [k_1; \ldots; k_n]$ that represent domain-specific keywords. The intuition is to inject the semantic information of the special domains using the domain-specific keywords. We define the set of keywords using a term-frequency-based approach. In particular, we estimate a score c for each word w from the target domain based on the normalized term frequencies estimated on documents from the general domain $\text{tf}_g(w)$ and domain-specific documents $\text{tf}_d(w)$:

$$c(w) = \alpha \cdot \text{tf}_g(w) + \text{tf}_d(w), \alpha < 0 \quad (4)$$

$$t_i = \{w | \text{rank}(c(w)) = i\}, 1 \leq i \leq n \quad (5)$$

Using $\alpha < 0$, we are able to select terms that are representative for the target domain and avoid terms that are frequent in the general domain. In Equation 5, we select the n words from the target

---

[3] `https://github.com/boschresearch/switchprompt`

domain with the highest scores c as our keywords. Each keyword t_i is then represented as a vector k_i, using the same language model as for prompting.

Initial experiments showed that it is not enough to simply set $P_d = K_n = [k_1, k_2, \ldots, k_n]$ but that the sequence of keywords should actually be combined with the sequence of soft prompts. Thus, we implement the domain-specific prompt as follows:

$$P_d = g_2[V_m; K_n] + (1 - g_2)[K_n; V_m] \quad (6)$$

$$g_2 = \sigma(w_2^\top s_{\text{input}}) \quad (7)$$

We combine the sequence of keywords of length n with the sequence of soft prompts of length m with concatenation, yielding a sequence of length $l = m + n$. We let the model decide with a second gate g_2 in which order the sequences should be concatenated. Again, the gate is calculated based on the representation of the input sentence and a trainable weight vector $w_2 \in \mathbb{R}^e$, where e is the embedding dimension. Thus, although the same domain-specific keywords are used for all inputs, the resulting soft prompt is dependent on the input sentence.

3.2 Prompting Architecture

Since our proposed method is a new definition of soft prompts, it can be integrated into any existing model that uses soft prompts. In our experiments, we adopt the *P-Tuning v2* architecture (Liu et al., 2022) because of its high efficacy on different natural language understanding tasks. P-Tuning v2 is an adaptation of deep prompt tuning (Qin and Eisner, 2021; Li and Liang, 2021b) that injects soft prompts at every layer of the pre-trained LM. During training, the prompts are tuned but the LM stays fixed. For the class prediction of the downstream task, a randomly initialized classification head is added on top of the pre-trained LM.

4 Experiments

In this section, we describe the setup (datasets, training details and baselines) and the results of our experiments.

4.1 Datasets

For our experiments, we use classification benchmark datasets from different domains: question classification from the general domain (TREC, Voorhees and Tice, 2000) and from the clinical domain (GARD, Kilicoglu et al., 2016), as well as experiment classification from the materials science domain (SOFC-Exp, Friedrich et al., 2020). Statistics of the datasets can be found in Table 1.

Dataset	**Domain**	**Instances**	**Classes**
GARD	Clinical	1253	11
SOFC-Exp	Materials science	2042	2
TREC	General	4893	7

Table 1: Statistics of text classification datasets.

Among them, the SOFC-Exp dataset offers a binary sentence classification task with positive and negative labels whereas GARD and TREC are multi-class question classification datasets.

To investigate very-low-resource settings, we construct *few-shot datasets* by randomly sampling N shots per class with $N \in \{2, 4, 16, 64\}$. Following the proposed theory for realistic *low-resource regimes* (Perez et al., 2021; Kann et al., 2019), we also create *few-shot development sets* by keeping the number of shots in the training and development sets in sync. In the 4-shot scenario, for example, both the training and the development set consist of 4 examples for each class. For all datasets, we use accuracy $(\%)$ as evaluation metric.

To give a closer insight into the challenges of the different domains that we use in our experiments, we present example instances from the datasets in Table 2. The examples show that the models need to cope with domain-specific terminology, such as "perisylvian polymicrogyria" (clinical domain) or "electrochemical" (materials science domain), and domain-specific labels, for instance, "diagnosis".

4.2 Training Details

We use open-sourced HuggingFace language models[4] for our experiments. We train our models with a batch size of 32. The maximum sequence length is set to 128 and we use dropout with rate 0.1 on the classification layer. We use the *ExponentialLR*[5] learning rate scheduler with a gamma value of 0.95 and the Adam optimizer. All experiments are performed on a $V100$ GPU.[6] Each reported result is the average performance of five runs.

4.3 Baselines

We compare our method to different baselines: (i) Fine-tuning of the pre-trained LM, (ii) prompting

---

[4] `https://huggingface.co/models`

[5] `https://pytorch.org/docs/stable/optim.html#torch.optim.Optimizer`

[6] We ran our experiments on a carbon-neutral GPU cluster.

Domain	**Input**	**Output**
Clinical	How is it different from bilateral perisylvian polymicrogyria in how it presents ?	Diagnosis
Clinical	Are there products other than cigarette tobacco associated with Buerger disease ?	Cause
Mat. science	They called this phenomenon nonfaradaic electrochemical modification of catalytic activity (NEMCA).	Negative
Mat. science	It is possible to reduce up to 35% of NO present when the cell stacks are polarized with 1.5 V for each cell.	Positive
General	What is the name of the largest city in Chile , South America ?	Location
General	What was the average life expectancy during the Stone Age ?	Number

Table 2: Example sentences and their labels from our domain-specific and general-domain datasets.

Methodology	**Model**	**2-shots**	**4-shots**	**16-shots**	**64-shots**	**All**
Fine-tuning	BERT	21.2	25.5	40.8	67.4	81.8
	Clinical BERT	35.9	40.4	56.3	68.1	82.5
P-tuning	BERT	**48.3**	48.9	53.1	68.1	82.0
	Clinical BERT	**49.2**	53.1	58.2	69.6	82.8
P-tuning V2	BERT	27.2	44.4	61.9	79.1	84.0
	Clinical BERT	34.3	48.7	63.4	**82.3**	86.7
SwitchPrompt	BERT	36.3	**54.2**	**64.0**	**81.1**	**85.4**
	Clinical BERT	40.9	**55.2**	**65.1**	81.9	**86.9**

Table 3: Results on special-domain dataset GARD.

Methodology	**Model**	**2-shots**	**4-shots**	**16-shots**	**64-shots**	**All**
Fine-tuning	BERT	18.2	26.1	48.5	54.6	61.9
	SciBERT	29.4	32.7	50.4	56.2	64.7
P-tuning	BERT	**37.5**	**38.2**	52.6	58.5	64.9
	SciBERT	**42.1**	**43.4**	54.8	59.3	66.2
P-tuning V2	BERT	30.8	31.2	52.8	59.9	68.4
	SciBERT	33.7	35.6	53.9	61.4	69.7
SwitchPrompt	BERT	32.4	34.3	**53.4**	**61.0**	**69.9**
	SciBERT	36.2	37.1	**55.9**	**62.5**	**70.6**

Table 4: Results on special-domain dataset SOFC-Exp.

Methodology	**Model**	**2-shots**	**4-shots**	**16-shots**	**64-shots**	**All**
Fine-tuning	BERT	33.3	53.3	71.4	88.7	95.7
P-tuning V2	BERT	56.0	63.3	79.4	**92.5**	96.8
SwitchPrompt	BERT	**66.7**	**72.4**	**88.3**	91.2	**97.6**

Table 5: Results on general-domain dataset TREC.

using P-tuning (Liu et al., 2021), and (iii) prompting using P-Tuning v2 (Liu et al., 2022). For all methods, we report results for using either a general-domain LM (BERT (Devlin et al., 2019)) or domain-specific LMs (Clinical BERT (Alsentzer et al., 2019) and SciBERT (Beltagy et al., 2019)).

4.4 Results

Low-resource domains. Tables 3 and 4 show the results of our model for the clinical and materials science domain in comparison to state-of-the-art baseline approaches. In general, the prompting methods outperform fine-tuning, with especially large margins for very-few-shot settings (2 and 4 shots). This highlights the limitations of fine-tuning with limited training datasets. Another general trend is that using domain-specific LMs (Clinical BERT and SciBERT, respectively) outperforms BERT from the general domain. Our proposed method *SwitchPrompt* outperforms other state-of-the-art prompting methods up to 2.1% points. We further note that (i) our method prompting general-domain LMs even outperforms other methods prompting domain-specific LMs, and (ii) our method reduces the performance gap between using an LM from the general domain vs. a domain-specific one.

For very-few-shot settings (2,4-shots), P-tuning outperforms our method. We assume that the reason is that it replaces the input of the LM with differential embeddings from the prompt-encoder, while in our method we consider the vanilla inputs of the LM, reducing the complexity and training time (see Figure 1) of our model.

General domain. To investigate the behavior of our method in the general domain, we now evaluate its performance on TREC. Table 5 shows that our method outperforms both fine-tuning and other prompting methods in almost all dataset settings, up to 10.7 accuracy points. Thus, even on the general domain, *SwitchPrompt* can boost the performance of pre-trained LMs.

5 Analysis

In this section, we report the results of our ablation study, give more insights into what our model learned and analyze its training time. We also provide a qualitative error analysis.

Ablation study. Our ablation study in Table 7 shows the impact of the different components of our prompting function, evaluated with the full GARD dataset. Row (1) corresponds to *SwitchPrompt*, and row (6) corresponds to the previous state-of-the-art prompting model P-Tuning v2. Row (2) shows that the concatenation of keywords and the general-domain soft prompt is important to the model. Row (3) and (4) show the large impact of the second gate g_2, and row (5) and (6) show that neither the domain-specific keywords K_n nor

Domain	Keywords
Clinical	Diagnosed, Prognosis, Cantrell, Idiopathic, Tourette, Opitz, Testotoxicosis, Late-onset, Amniocentesis, Prenatally
Mat. science	Fuel, Oxide, D8-Discover, Viscometer, Hydroxide/poly, Room-temperature, Ion-conductor, Electrocatalytic, Cobalt-doped, Non-homogeneous
General	Cholera, Tasman, Conservancy, Boil, Premier, Consumption, Conditioner, Foster, Chemiosmotic, Registers

Table 6: Automatically selected 10 keywords per domain by our approach.

Prompt	Acc
(1) $g_1(\mathsf{pad}(V_m)) + (1 - g_1)(g_2[V_m; K_n] + (1 - g_2)[K_n; V_m])$	**85.4**
(2) $g_1(\mathsf{pad}(V_m)) + (1 - g_1)(g_2 V_m + (1 - g_2)K_n)$	82.6
(3) $g_1(\mathsf{pad}(V_m)) + (1 - g_1)[V_m; K_n]$	81.4
(4) $g_1(\mathsf{pad}(V_m)) + (1 - g_1)[K_n; V_m]$	77.6
(5) K_n	54.8
(6) V_m	84.0

Table 7: Impact of prompt design choices in the full-data setting of the GARD dataset using BERT embeddings. Row (1) corresponds to *SwitchPrompt*, and row (6) corresponds to P-Tuning v2.

Input	Prediction	Gold Output
How can this be?	Management	Susceptibility
Will we be okay?	Information	Prognosis
What is the treatment of mixed connective tissue disorder ?	Information	Management
What are the expected outcomes for individuals with cryoglobulinemia ?	Information	Prognosis

Table 8: Error analysis on GARD dataset.

the general soft-prompting vectors V_m alone are sufficient to achieve the highest performance.

Domain-specific keywords. The keywords are an integral part of *SwitchPrompt*. Since we compute keywords automatically (see Section 3), we analyze the extracted keywords in more detail.

Table 6 shows the 10 keywords that have been selected by our method. For the clinical and materials science domain, the keywords are domain-specific terms while for the general domain, the keywords cover a broad range of topics.

Training time analysis. During training time, the underlying LM is frozen in the *SwitchPrompt* framework. This substantially reduces training time and computational memory, compared to alternative approaches, such as fine-tuning or P-Tuning. Figure 1 illustrates this. P-Tuning v2 is a little bit faster than our approach as it does not need to train the gating parameters. However, the time difference is considerably small (2.2 min for 10 epochs in the all-data setting, i.e., 0.22 min per epoch).

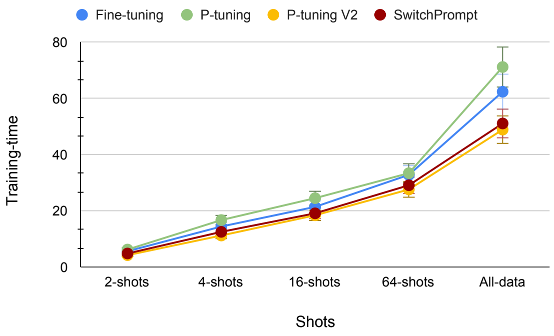


Figure 1: Training time in minutes for 10 epochs for different methods on the GARD dataset.

Qualitative error analysis. Finally, we manually conduct a qualitative error analysis on the GARD dataset. The results are displayed in Table 8. We find that our method mainly fails when the input sentences convey little domain-specific information (see examples in first two rows). Another category of errors is the prediction of a more general class ("Information" instead of "Management" or "Prognosis" in the last two rows).

6 Conclusion

In this paper, we proposed a new methodology called *SwitchPrompt* for effectively prompting pre-trained language models in low-resource domains. Integral parts of our method are domain-specific keywords and gates, which allow the language model to dynamically retrieve domain-specific knowledge. Experiments on sentence classification datasets from different domains show that our method outperforms various baseline methods in few-shot and all-data settings. In particular, it reduces the performance gap between general-domain and domain-specific language models. Future work can investigate the impact on sequence-labeling tasks as well as on mixed-domain datasets.

Acknowledgments

We would like to thank the members of the BCAI NLP & NS-AI research group and the anonymous reviewers for their helpful comments.

Limitations

In preliminary experiments, we found that our method is sensitive to the selection of keywords. While we found an automatic and domain-independent way for extracting them (see Section 3), its efficacy needs to be tested on more domains and possibly also on mixed domain datasets.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3613–3618. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of NeurIPS*, volume 33, pages 1877–1901.

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

Annemarie Friedrich, Heike Adel, Federico Tomazic, Johannes Hingerl, Renou Benteau, Anika Marusczyk, and Lukas Lange. 2020. The SOFC-exp corpus and neural approaches to information extraction in the materials science domain. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1255–1268, Online. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Naman Goyal, Jingfei Du, Myle Ott, Giri Anantharaman, and Alexis Conneau. 2021. Larger-scale transformers for multilingual masked language modeling. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 29–33, Online. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Katharina Kann, Kyunghyun Cho, and Samuel R. Bowman. 2019. Towards realistic practices in low-resource natural language processing: The development set. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3342–3349, Hong Kong, China. Association for Computational Linguistics.

Halil Kilicoglu, Asma Ben Abacha, Yassine Mrabet, Kirk Roberts, Laritza Rodriguez, Sonya Shooshan, and Dina Demner-Fushman. 2016. Annotating named entities in consumer health questions. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3325–3332, Portorož, Slovenia. European Language Resources Association (ELRA).

Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2022. CLIN-X: pre-trained language models and a study on cross-task transfer for concept extraction in the clinical domain. *Bioinformatics*, 38(12):3267–3274.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021a. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021b. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *CoRR*, abs/2103.10385.

Yinhan Liu, Myle Ott, Naman Goyal, J Du, M Joshi, D Chen, O Levy, M Lewis, L Zettlemoyer, and V Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 11054–11070.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5203–5212. Association for Computational Linguistics.

Nathan Schucher, Siva Reddy, and Harm de Vries. 2022. The power of prompt tuning for low-resource semantic parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 148–156, Dublin, Ireland. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece. European Language Resources Association (ELRA).

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2020. DomBERT: Domain-oriented language model for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1725–1731, Online. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

Do dialogue representations align with perception? An empirical study

Sarenne Wallbridge and **Peter Bell** and **Catherine Lai**
Center for Speech Technology Research
University of Edinburgh, United Kingdom
`{s1301730, peter.bell, c.lai}@ed.ac.uk`

Abstract

There has been a surge of interest regarding the alignment of large-scale language models with human language comprehension behaviour. The majority of this research investigates comprehension behaviours from reading isolated, written sentences. We propose studying the perception of dialogue, focusing on an intrinsic form of language use: spoken conversations. Using the task of predicting upcoming dialogue turns, we ask whether turn plausibility scores produced by state-of-the-art language models correlate with human judgements. We find a strong correlation for some but not all models: masked language models produce stronger correlations than autoregressive models. In doing so, we quantify human performance on the response selection task for open-domain spoken conversation. To the best of our knowledge, this is the first such quantification. We find that response selection performance can be used as a coarse proxy for the strength of correlation with human judgements, however humans and models make different response selection mistakes. The model which produces the strongest correlation also outperforms human response selection performance. Through ablation studies, we show that pre-trained language models provide a useful basis for turn representations; however, fine-grained contextualisation, inclusion of dialogue structure information, and fine-tuning towards response selection all boost response selection accuracy by over 30 absolute points.

1 Introduction

Human language processing has intrigued researchers from numerous fields for centuries (Herder, 1772). The relatively recent convergence of philosophy, psycholinguistics, and information theory has produced valuable theories of language production and comprehension by framing people as predictive processors (Christiansen and Chater, 2016; Levy, 2008; Hale, 2001). In particular, Surprisal Theory posits that comprehension effort is directly related to the predictability of a linguistic unit in its context (i.e., its surprisal) (Hale, 2001), and the Smooth Signal Redundancy hypothesis (Aylett and Turk, 2004) (and related theories including Uniform Information Density (Fenk and Fenk, 1980; Levy and Jaeger, 2006) and Entropy Rate Constancy (Genzel and Charniak, 2002)) demonstrates that we produce linguistic signals that tend towards uniform distributions of information, or constant predictability.

Such theories rely on the conditional probability of observing a linguistic unit in a particular context. Traditionally, estimates of these probabilities were obtained through statistical models such as n-grams or PCFGs (Smith and Levy, 2013; Hale, 2001). By comparison, large scale language models (LLMs) like Transformers allow much greater degrees of context to be integrated into probability estimates (Vaswani et al., 2017). LLMs have enabled massive progress across the continuum of NLP tasks (Wang et al., 2019; Hu et al., 2020). As such, a recent field of research has investigated the application of LLMs for producing estimates of human surprisal. Language model quality (measured by perplexity) generally correlates with ability to predict aspects of human perception behaviour– psychometric predictive power (Wilcox et al., 2020; Frank et al., 2015; Levy, 2011; Goodkind and Bicknell, 2018).

The vast majority of psychometric predictive power studies are based on monologue-like data and involve perception of isolated sentences. However, the most natural, innate form of language-use is communicative interaction; we learn to hold conversations with very little direct instruction where as tasks related to the production or comprehension of monological data such as reading, writing, or presenting require years of conscious effort to master. Given the additional modelling capacity of current LMs, there has been growing interest in using them to apply classical language processing theories to more natural forms of language

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2696–2713
May 2-6, 2023 ©2023 Association for Computational Linguistics

like dialogue (Vega and Ward, 2009; Doyle and Frank, 2015b,a; Giulianelli and Fernández, 2021; Giulianelli et al., 2021). Results from these works do not align neatly with findings about perception of isolated sentences. This raises interesting questions about differences between information transmission strategies employed in monological and conversational settings, and suggests that the the alignment of language models and human perceptual behaviour for monological and isolated linguistic data may not extend to dialogue.

At the same time, the field of dialogue modelling in NLP has grown substantially. Much of the recent work in this field is geared towards leveraging the structural differences between monologue and dialogue data (Lowe et al., 2015; Wu and Xiong, 2020). Although this has increased performance on dialogue modelling benchmark tasks such as response selection and dialogue generation, it is unclear whether increased performance on these tasks results in representations that align with human perception (Wolf et al., 2019; Liu et al., 2020).

To understand whether representations of dialogue align with human perception of dialogue, we ask how well language model outputs correlate with human perception of dialogue turn acceptability. To do so, we build on the novel perceptual task of rating dialogue turn plausibility proposed in our previous work (Wallbridge et al., 2022). First, we study the nature of human expectations in dialogue. By recasting the rating task as a discriminative one, we find evidence that, similar to linguistic acceptability judgements, human judgements of dialogue turn plausibility are probabilistic rather than deterministic (Lau et al., 2017). Next, we establish the psychometric predictive power of different context-dependent text representations with respect to this task by asking whether such representations are predictive of dialogue plausibility judgements. In previous work, we found a statistically significant but weak correlation between surprisal estimates from a generative dialogue-based language model and human plausibility scores for (context, response) pairs (Wallbridge et al., 2022). We use this paradigm here to investigate the predictive power of other language modelling styles. In doing so, we find a strong correlation between (context, response) scores from a masked LLM fine-tuned towards response-selection and human judgements. Interestingly, this model also achieves "superhuman" response selection performance in the sense that it obtains a higher accuracy than participants. This finding, however, motivated our analysis into response-selection dialogue models and under what conditions they align with human perception.

2 Language models and human language perception

Multiple large-scale comparisons between the predictive power of different families of a language models provide strong evidence for a relationship between a language model's quality and its predictive power for human comprehension behaviour. Behaviours include self-paced reading times and gaze duration (Wilcox et al., 2020; Goodkind and Bicknell, 2018; Meister et al., 2021), grammatical acceptability judgements (Richter and Chaves, 2020; Lau et al., 2017; Warstadt et al., 2019; Meister et al., 2021), and brain response data (Frank et al., 2015; Schrimpf et al., 2021). Other works provide similar evidence for a close relationship between surprisal and human language perception through improved language generation under cognitive-inspired constraints (Wei et al., 2021).

Although the majority of these studies are based on a constrained definition of perception using isolated sentences, they already reflect evidence that different comprehension tasks rely on different types of linguistic expectations. Wilcox et al. (2020) finds a dissociation between the syntactic generalisation capability of LMs which is heavily dependent on model architecture, and LM ability to predict human reading times. Similar results were found by Meister et al. (2021) – BERT is highly predictive of acceptability judgements but was described as "remarkably poor" for estimating reading times.

These findings suggest that the alignment of language model output and comprehension behaviours is unlikely to generalise to the perception of dialogue.

2.1 Perception of dialogue acceptability

The majority of psycholinguistic language production theories have been developed based on monological data analysed from a generative linguistics perspective. However, dialogue perception differs from the comprehension of monological data in a number of fundamental aspects (Pickering and Garrod, 2004). Dialogue has been described as a game where participants only "win" if both understand the dialogue (Lewis, 1969), or as a joint process where interlocutors collaborate to build common ground (Clark and Wilkes-Gibbs, 1986). Context plays a much larger role in dialogue comprehension (Nieuwland and Berkum, 2006). As a concrete example, Fernandez and Ginzburg (2002) find that more than 11% of dialogue turns in the British National Corpus are non-sentential in isolation. Pragmatic context is particularly important. Conversational acts such as backchanneling or the use of adjacency and coordinate pairs tend to have low lexical information density, but are crucial for turn taking and grounding (Clark and Wilkes-Gibbs, 1986; Kawahara et al., 2016).

The challenges of integrating such context into language models has been highlighted by a handful of works investigating information transmission strategies in communicative contexts. Giulianelli et al. (2021) find that relevant contextual units in dialogue should be topically and referentially coherent, and that defining these units depends on the domain of discourse. Doyle and Frank (2015b) find that common ground is a crucial aspect to account for when modelling information distribution in Twitter dialogues. Vega and Ward (2009) present evidence of uniform information density in spoken dialogues, but note the importance of accounting for non-lexical information as additional context.

To model information strategies in communication, language models should align with human perception of dialogue acceptability. In this work, we propose using the task of dialogue utterance plausibility to quantify human perception of dialogue acceptability, and examine whether language models align with this aspect of perception.

2.2 Language models and dialogue

Similarly to psycholinguistics, the vast majority of NLP models have been developed based on monological data. However, interest in modelling interaction and dialogue is booming, fuelled by the development of commercial dialogue systems like Apple Siri[1] and Amazon Alexa[2]. Work related to dialogue perception is also gaining interest, focused on tasks such as dialogue act classification and dialogue coherence estimation (Shriberg et al., 1998; Tran, 2020; Cervone et al., 2018). However, it is unclear whether current dialogue-based LLMs produce perceptually meaningful representations.

Response selection Regardless of the downstream application, response selection has become a pervasive task in dialogue modelling. A major reason for this is that it doesn't require annotated labels. This self-supervised task is used throughout dialogue modelling as both an evaluation metric and a training signal. Given some degree of dialogue history as an anchor, response selection involves selecting the upcoming response from a set of potential turns. This is a direct extension of quintessential next-sentence-prediction task which makes a strong assumption that there is a single correct upcoming turn (Devlin et al., 2019). In this work, we explore the extent to which this assumption holds when making predictions in dialogue.

Response selection models, also known as retrieval-based dialogue models, can be broadly separated into two classes: bi-encoder models that learn independent representations of responses and their respective contexts, and cross-encoders which encode the context and response as a single representation before scoring them (Henderson et al., 2020; Zhou et al., 2016; Wu et al., 2020, 2017a). The latter often involve tuning a pre-trained language model using dialogue-specific objectives (Wolf et al., 2019; Han et al., 2021; Xu et al., 2020). Such methods aim to make use of the representation capacity of pre-trained language models, while also leveraging important and unique structural features of dialogue data.

By encoding contexts and responses in isolation, bi-encoders achieve cheaper training and inference as representations can be cached. However, they enforce a strong independence assumption between contexts and responses. The separate encoders can provide a weak notion of position (i.e., the same lexical content may be encoded differently if it is a response or a piece of context), however cross-encoding captures much richer interactions.

Given the importance of response selection in dialogue modelling, we explore here how well this task aligns with the perception of spoken dialogue acceptability. We also perform ablation studies to better understand which aspects of model architecture are important for response selection.

3 Experiments

3.1 Psychometric predictive power in spoken dialogue transcripts

Data To investigate the relationship between representations of dialogue and human perception, we

---

[1] https://www.apple.com/siri

[2] https://developer.amazon.com/alexa

make use of a dataset collected in our previous work (Wallbridge et al., 2022) which comprises of 100 (context, response) pairs extracted from the Switchboard Telephone Corpus, and their associated median plausibility scores. Switchboard is a corpus of over 2,400 spontaneous chit-chat style telephone conversations between 542 participants covering 70 topics. It includes manual transcriptions and turn segmentations (Godfrey et al., 1992). Each of the 100 stimuli consists of a set of speaker turns as context $c = [c_1, ..., c_k]$ and an upcoming response r where r is either the true upcoming turn in the dialogue, or a turn sampled from the corpus. Scores were collected by asking participants to rate how plausible r is in the context of c on a scale of 1-5 ("Very Unlikely" – "Very Likely"). Each dialogue context c was presented in 10 pairs: with the true upcoming turn, and 9 negative samples.

In this previous work, we found that people can make relatively accurate discriminative judgements regarding the true upcoming turn in a dialogue: using the mean score per stimuli as a proxy for turn selection, participants obtained an accuracy of 70% (Wallbridge et al., 2022). Plausibility scores were also compared to surprisal estimates from TurnGPT (Ekstedt and Skantze, 2020). Although the relationship between plausibility scores and TurnGPT surprisal estimates was found to be statistically significant, the correlation between them was weak.

Here, we investigate the effect of language modeling styles on psychometric predictive power, using both generative and retrieval-based language models. To do so, we train a range of language models towards the response selection task using the Switchboard dataset. We split the corpus by conversation into training, validation, and test sets (80%, 10%, 10%). Transcripts are all lower-cased and speech-based annotations such as pronunciation markers and speech events are removed.

Models:

Our models are all implemented in PyTorch (Paszke et al., 2019) and pre-trained language models are obtained from the Transformers library (Wolf et al., 2020).

To test generative model capabilities, we employ TurnGPT. This architecture extends the standard GPT-2 model of Radford et al. (2019) to dyadic interaction by fine-tuning a pretrained GPT-2 model with additional speaker embeddings and speaker tokens to encode conversational structure (Ekstedt and Skantze, 2020). For our experiments, we fine-tune the pretrained GPT-2 model from the Transformers library (Wolf et al., 2020) on our training portion of Switchboard using the augmented TurnGPT cross-entropy loss.[3]

Table 1: Correlations between human plausibility scores and LM response scores. Scores are based on turn-level surprisal estimates for TurnGPT, and the joint (context, response) response selection score for BERT-FP.

Model	**Metric**	ρ	p-value
TurnGPT	S_{total}	-0.302	0.002
	S_{mean}	-0.392	<0.001
	$S_{relative}$	-0.395	<0.001
	S_{max}	-0.463	<0.001
	S_{var}	-0.346	0.001
BERT-FP	RS Score	0.637	<0.001

As has been done in previous studies of psychometric predictive power, we obtain estimates of response surprisal from TurnGPT using various aggregates of token-level surprisal (Lau et al., 2017; Meister et al., 2021; Wallbridge et al., 2022). These include both global and local definitions of surprisal. Global metrics include $S_{total}, S_{mean}, S_{relative}$ while local metrics include S_{max}, S_{var}. See Appendix A for further details.

To test retrieval-based methods, we use the BERT-FP architecture (Han et al., 2021). This cross-encoder obtains state-of-the-art response selection for written conversational benchmarks including the Ubuntu (Lowe et al., 2015), E-commerce (Zhang et al., 2018), and Douban (Wu et al., 2017b) datasets. The model encodes the joint (context, response) pair using pre-trained BERT (Devlin et al., 2019). We use `bert-base-uncased` from the Transformers library. The resulting BERT $[CLS]$ token is fed through a single-layer classifier to produce a response selection score – the relevance of the response given the context. We fine-tune BERT-FP towards response selection on our training portion of Switchboard using binary cross entropy loss.[4] See Appendix A for additional details of model training procedures. Rather than computing response surprisal from word-level model output, we obtain response plausibility scores directly from the BERT-FP model.

---

[3] We implement our model using the TurnGPT Github repository `https://github.com/ErikEkstedt/TurnGPT`

[4] We implement our model based on the BERT-FP Github repository `https://github.com/hanjanghoon/BERT_FP`

Table 2: Discriminative performance of people, TurnGPT, and BERT-FP.

Model	**Metric**	**F1**	**R10@1**
Human	Mean score	0.783	0.7
	Median score	0.800	0.8
TurnGPT	Max surprisal	0.435	0.3
	Mean surprisal	0.538	0.4
BERT-FP	Score	0.889	0.9

Input to both models consists of a set of speaker turns context $c = [c_1, ..., c_k]$ and response turn r where context turns are separated by a special $[eos]$ token to denote a change in speaker. As per the original BERT-FP implementation, we use a fixed number of context turns ($k = 3$) (Han et al., 2021).

3.1.1 Correlation with plausibility perception

Table 1 show the correlations between model response scores and human plausibility judgements. Similar to previous works (Lau et al., 2017; Meister et al., 2021) and our own (Wallbridge et al., 2022), we find a relatively weak correlation between TurnGPT surprisal metrics and plausibility scores, as well as variation in correlation strength between surprisal metrics.

Table 1 also shows the language model to have a larger effect than surprisal metric. We find a much stronger correlation between turn-plausibility scores from humans and scores predicted by our retrieval-based cross-encoder, BERT-FP, than for TurnGPT surprisal estimates. This indicates that the response selection task used to fine-tune BERT generates a latent space that matches some aspects of dialogue acceptability perception.

3.1.2 Discriminative performance

The plausibility scoring task can be reframed as response selection by taking the mean score for each (context, response) pair as the selection criterion, using each of the scoring metrics. Table 2 provides the response selection performance of our models, as well as a benchmark of human consensus performance for this task. We report both the standard response selection accuracy metric $R_{10}@1$ as well as F1 score which denotes the highest linear classification accuracy that can be achieved across all (context, response) pairs based on a given metric.

The response selection paradigm is used widely throughout dialogue modelling. But to our knowledge, this is one of the first investigations of how well humans can judge upcoming responses. People can execute the response selection task relatively well using a small amount of context, however their performance is not perfect in terms of $R_{10}@1$ accuracy.

Interestingly, we find that BERT-FP outperforms people at this task, achieving higher accuracy and F1 score. This "superhuman" performance is further evidence that the response-selection objective deviates from human perception of dialogue. We provide a some qualitative observations of deviations between model and human judgements below (see Error Analysis).

Although TurnGPT showed weak to moderate correlation with human judgements scores, it performs very poorly in the response selection task framing. Different surprisal definitions had relatively large impacts on the correlation strengths reported in Table 1, but these were not reflected in the F1 scores. TurnGPT is trained using cross-entropy loss over next-token prediction (autoregressive pretraining), while BERT-FP is based on the masked language modelling paradigm – these differences in underlying pre-training could help explain performance deviations and would be interesting to explore in future work. We discuss these findings further in section 4.

Error analysis: cross-encoders and humans Neither BERT-FP nor human raters achieve perfect performance on the response selection framing. However, their mistakes are different. The cross-encoder succeeds on all three questions where humans did not rate the true response highest, and vice versa for the question misranked by the model.

For all three stimuli that people misranked, the true response obtained the second highest score and had a mean plausibility score greater than 3. In two of these cases, the highest-scoring response was generic response ("mhm", "yeah") which is plausible in a wide range of contexts. In the final case, the highest scoring response was a question that shifted the topic of conversation. We noted this pattern in other stimuli: responses that initiated topic shifts – often questions – were often rated as somewhat likely by people but obtained low BERT-FP scores.

In comparison, for the stimuli misranked by BERT-FP, the true response obtained the 4th highest score. All four of the misranked stimuli included two or more context turns, suggesting that the amount of context is not necessarily a driving

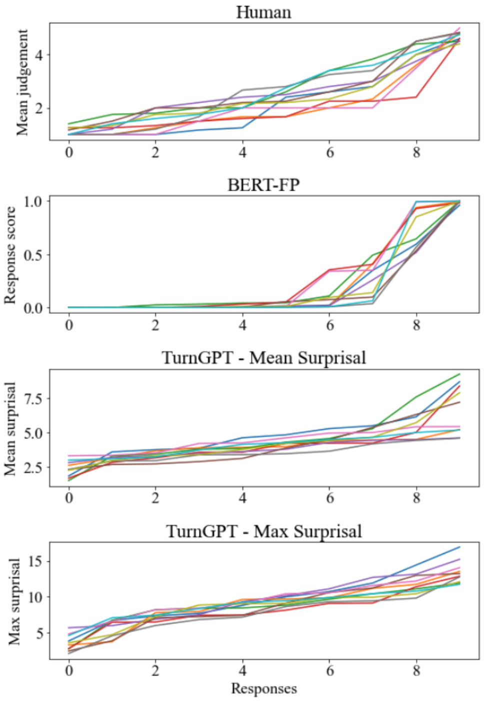


Figure 1: Sorted score sets per response selection stimuli for different models

factor in mistakes for either the model or people.

We also examine the rank correlation (Spearman r) between human ratings and model scores on a per-stimuli basis. Coefficients range from 0.33 to 0.93 ($\mu = 0.65 \pm 0.18$). The lowest correlations are obtained for stimuli where the model only scores a single response highly while the rest obtain scores close to 0. The distributions of BERT-FP scores across stimuli are more peaked than ratings produced by people (see Figure 1). Surprisal metrics from TurnGPT show score distributions more similar to human judgements, further highlighting differences between these models.

3.2 Model ablation for response selection

Generative (TurnGPT) and retrieval-based (BERT-FP) models display significantly different propensities for psychometric predictive power and response selection performance. We use ablation studies to better understand which aspects of the retrieval-based model architecture are valuable for response selection. In particular, we consider

1. Independence of context and response encoding: bi- versus cross-encoders
2. Inclusion of dialogue structure: $[eos]$ tokens
3. Importance of fine-tuning.

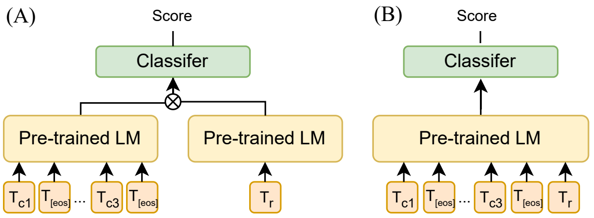


Figure 2: Architectures for (A) bi- and (B) cross-encoders. Input is shown in SBERT format; T denotes tokenized text. For BERT-based models, we prepend all input sequences with a $[CLS]$ token. Cross-encoder input also contains a $[SEP]$ token before and after T_r.

Models: We compare cross- and bi-encoder networks, depicted in Figure 2. The cross-encoder architecture is based on BERT-FP (Han et al., 2021) which encodes the joint context and response pair then feeds the resulting sequence representation through a classification network. We experiment with using both a single linear-layer classifier, or three fully-connected ReLU layers with dropout and a final sigmoid layer.

The bi-encoder style model follows the architecture presented in (Henderson et al., 2020). Each context and response is encoded in isolation using a shared, pre-trained language model. Context and response representations are then concatenated and fed through the same binary classifiers as the cross-encoder architectures.

We test scores based on both BERT $[CLS]$ tokens and SBERT sequence representations (Devlin et al., 2019; Reimers and Gurevych, 2019). SBERT representations are produced by mean pooling fine-tuned token representations from BERT. We use `bert-base-uncased` for BERT-based models and `all-MiniLM-L6-v2` for SBERT, both obtained from the Transformers library.

All models are trained towards the response selection task with a binary cross entropy loss on our training portion of Switchboard. We train models with the underlying LM in frozen and unfrozen scenarios.

3.2.1 Results

We present a simple baseline for response selection: minimizing the Euclidean distance between pre-trained SBERT representations of (context, response) pairs. This outperforms many of the ablated models discussed hereafter (see Table 3).

Independence of context and responses Although bi-encoders achieve cheaper training and inference by encoding contexts and responses in

Table 3: Response selection accuracy R_{10}@1 for models based on SBERT. The SBERT baseline (response selection based on euclidean distances between pre-trained SBERT representations of context and responses) achieves an R_{10}@1 of 0.480

Encoding Strategy	**R_{10}@1 accuracy**			
	Linear Classifier	**+ unfreeze**	**Non-linear Classifier**	**+ unfreeze**
Bi-encoder	0.102	0.105	0.455	0.507
Cross-encoder	0.236	0.465	0.307	0.472
Cross-encoder + [eos]	0.216	0.611	0.306	0.612

Table 4: Response selection accuracy R_{10}@1 for models based on BERT $[CLS]$

Encoding Strategy	**R_{10}@1 accuracy**			
	Linear Classifier	**+ unfreeze**	**Non-linear Classifier**	**+ unfreeze**
Bi-encoder	0.103	0.104	0.212	0.316
Cross-encoder + [eos]	0.324	0.675	0.346	0.673

isolation, they lack the rich contextualisation of cross-encoders. Previous work reports better response selection performance from cross-encoders for written dialogue (Urbanek et al., 2019). However, recent spoken dialogue research (Fuscone et al., 2020) as well as our own (Wallbridge et al., 2021) has shown that both people and models can make predictions about upcoming conversational turns based on small amounts of context. We therefore compare bi- and cross- encoding strategies to understand the value of such contextualisation in the domain of spoken dialogue.

Tables 3 and 4 show that bi-encoders require a classifier with some degree of non-linearity. Regardless of whether the underlying language model is frozen or not, bi-encoders with a linear classifier do not improve on chance performance. In comparison, the cross-encoder improves on chance even with a frozen language model and a linear classifier. This is the case for both SBERT- and BERT-based models, indicating that masked language model pre-training extracts information that is of some value for predicting upcoming dialogue turns.

When non-linearity is introduced in the classifier, bi-encoding outperforms cross-encoding by quite a margin on top of a frozen SBERT model. This may reflect similarities between response selection and the SBERT training objective of contrastive sentence similarity (Reimers and Gurevych, 2019). We verify that this result was not caused by the special $[eos]$ turn delimeter that wasn't seen during cross-encoder pre-training by training the same model without this token added to the input.

Regardless of the underlying LM, the performance gap between unfrozen bi- and cross-encoders indicates that the lack of contextualisation may cap bi-encoder performance. This difference confirms that fine-grained contextualisation is important for dialogue representations.

Dialogue structure: the importance of *[eos]* The effect of special tokens like $[eos]$ on the behaviour of large LMs is an active research area. Some works have found such tokens to have detrimental effects on the generalisability of generative models (Newman et al., 2020) but many recent models for dialogue representation rely on these tokens to capture structural characteristics of dialogue (Gu et al., 2020; Wolf et al., 2019).

Comparing the cross-encoder performance with and without $[eos]$ in Table 3 shows that explicit turn information is extremely valuable for response selection performance.

Fine-tuning The importance of pre-trained language models in NLP can't be understated, however spoken dialogue is fundamentally different to the monological text used for pre-training. Similar to previous work investigating the importance of fine-tuning with dialogue data, we find that pre-trained BERT representations do contain some useful information for response selection, but that fine-tuning towards the target task is still important (Noble and Maraev, 2021). In addition, we find that fine-tuning is much more beneficial when dialogue structure information is provided explicitly via the $[eos]$ token.

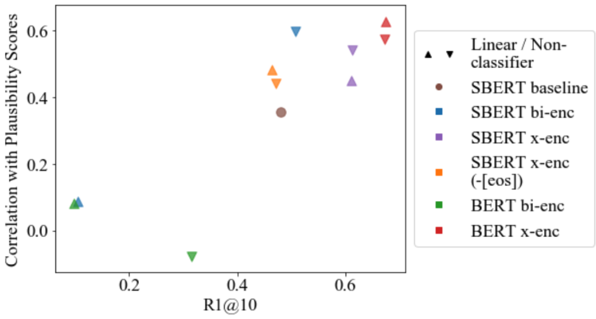


Figure 3: Response selection accuracy vs. psychometric predictive power

3.3 Is response selection performance a proxy for Psychometric Predictive Power?

Given the range of response selection results from our ablations, we consider the relationship between response selection performance and correlation with human plausibility judgements (PPP). We find a strong correlation between the two: $\rho = 0.84$. This relationship is marginally stronger if a larger range of response selection accuracies are considered; comparing response selection performance measured as $\sum_{k\in\{1,2,5\}}[R10@k]$ and PPP produces a correlation of $\rho = 0.86$. To ensure this correlation was not dominated by human response selection ability, we also computed this correlation across only negative responses and found a similarly strong correlation of model response selection performance and human scores ($\rho = 0.82$).

This indicates that response selection performance can be used as a coarse proxy for how well a model may correlate with human turn plausibility judgements. However, response selection performance patterns (e.g. SBERT cross-encoding outperforming bi-encoding) are not reflected in this relationship. Although the modelling capacity of the classifier (i.e., linear versus non-linear) did not have much effect on the response selection performance of cross-encoders (see pairs of points grouped around $R_{10}@1$ of $0.47, 0.61, 0.67$ in Figure 3), the impact on PPP is larger.

4 Discussion

Using the task of upcoming dialogue turn prediction, we presented a benchmark for human performance on the widely-used task of response selection: people are able to do this task, but not perfectly. To the best of our knowledge, this is the first investigation into the perceptual validity of response selection for spoken dialogue transcripts. Urbanek et al. (2019) measure human response selection using stimuli from written dialogues grounded in a virtual world, using sets of 20 responses. However, they note that the large number of negative samples makes obtaining reliable judgements from participants difficult. The rating paradigm proposed here avoids these issues, provides information about perceptual certainty, and is independent of the choice of negative samples.

Human performance showed that a fundamental assumption of response selection – that a single “correct” upcoming turn can be predicted from previous context – does not reflect the reality of perception with respect to dialogue acceptability. This result supports findings in cognitive science that linguistic perception of acceptability is intrinsically probabilistic (Lau et al., 2017; Chater et al., 2006).

We then used this quantification of dialogue acceptability perception to establish the psychometric predictive power of different language model families, finding that response selection performance can be used as a coarse proxy for alignment with this aspect of perception, but only to a certain extent. Strong correlation of BERT-FP scores and human plausibility ratings suggests that, as a training objective, the discriminative response selection task does align with some aspects of dialogue acceptability perception. However, this model achieved “superhuman” response selection performance. Qualitative differences in response scoring detailed in Section: Error Analysis also indicate that this discriminative training paradigm doesn’t capture the open-ended nature of dialogue acceptability perception. On the other hand, TurnGPT produced a moderate correlation with plausibility judgements, but performed poorly at response selection. (Urbanek et al., 2019; Wu and Xiong, 2020) report similar findings regarding the response selection performance of generative models compared to retrieval-based models. We conducted preliminary analysis of quantitative and qualitative differences between autoregressive and masked LMs, however given that language model architecture is known to significantly affect psychometric predictive power measured by other perceptual tasks (Wilcox et al., 2020), we believe this to be a fruitful avenue for future investigation.

Finally, to investigate the strong correlation between the cross-encoder-style BERT-FP scores and perceptual judgements of upcoming dialogue turns, we ablated aspects of the model. In particular, we examined the importance of contextualisation in dialogue representations, the value of including turn-taking structure, and whether pre-training is beneficial. Our bi-encoder models provided a small amount of dialogue structure information, however, even our best-performing bi-encoder only marginally outperforms our baseline of simply measuring the distance between SBERT embeddings for response selection. This may suggest that this method of encoding dialogue structure is too weak. Cross-encoding, on the other hand, provides much more granular contextualisation. We found cross-encoders to be much less sensitive to the capacity of the classifier than bi-encoders. Our cross-encoder outperformed the SBERT baseline and bi-encoders by a large margin when dialogue structure is explicitly encoded in the input (as $[eos]$ tokens). Inclusion of dialogue structure information and fine-tuning towards response selection boosted response selection accuracy by over 30 absolute percentage points, highlighting the value of turn structure information for response selection models.

5 Conclusions

In this study, we extend the concept of psychometric predictive power for language models to dialogue. We define the perceptual behaviour to be predicted in terms of expectations regarding upcoming dialogue turns, and ask whether different styles of language model are predictive of dialogue turn acceptability judgements from humans.

Using transcripts of spoken dialogue, we find a strong correlation between the two, but not for all models: masked language models produce higher correlations than their autoregressive counterparts. Ablation studies demonstrate that features of cross-encoders which enable fine-grained contextualisation are important for alignment with human acceptability judgements. Framing the turn acceptability rating task as discriminative response selection, we present a benchmark for human performance on this task, one of the first to our knowledge. Comparisons between human and model performance on this widely-used dialogue objective indicate that response selection can be used as a coarse proxy for alignment with perception of dialogue turn acceptability but only to a certain extent. Human perception of dialogue turn acceptability is inherently probabilistic and models make different mistakes compared to humans.

We hope that these findings encourage development of more perceptually-motivated evaluation and training paradigms in dialogue modelling.

6 Limitations

Psychometric predictive power has so far been contained to isolated sentence comprehension studies. This work takes steps to extend the notion of psychometric predictive power to more natural forms of language processing: comprehension of communication. Given that this is an exploratory study, there are many limitations, many of which we consider to be potential directions for future work.

The definition for perception used throughout this study is as generic as possible: predicting upcoming turns. However, there is ample evidence that linguistic processing is task-dependent (Huettig and Guerra, 2019; Huettig et al., 2020). Exploring other quantifications for the perception of dialogue acceptability would be an interesting avenue for future work. Similarly, features of communication such as turn-taking are known to vary across languages and cultures (Skantze, 2021). As such, is it unclear how well the findings presented in this work generalise to other styles and forms of conversation.

All of the turn representations in this work are based on a fixed number of turns as context, however we should also study whether the amount of context affects LM alignment with human judgments. In our previous work, we found the amount and type of context to affect human performance (Wallbridge et al., 2021), while Henderson et al. (2020) show that it affects response selection performance in models.

Previous works have studied surprisal estimates computed from different aggregates of word-level estimates from LMs (Lau et al., 2017; Meister et al., 2021; Wallbridge et al., 2022). These surprisal definitions are useful for testing psycholinguistic theories about information transmission strategies such as Uniform Information Density (Fenk and Fenk, 1980; Levy and Jaeger, 2006) and Entropy Rate Constancy (Genzel and Charniak, 2002), however we found stronger correlations with human judgements by using language models to compute plausibility scores directly. Understanding the gap between these scores and aggregate word-level surprisal estimates could inform more complex definitions of surprisal.

Although response selection is a pervasive training objective in dialogue representation research, there is a growing body of work refining this task, and learning representations using additional dialogue-centric learning objectives such as detection of turn insertions or deletions (Qiu et al., 2021; Lee et al., 2021). Future work could consider how these training signals affect the psychometric predictive power of models with respect to both response selection and plausibility judgements.

This study has been restricted to the lexical component such dialogues. However, perception and production of linguistic signals are dependent on the modality available for transmission, and what information transmission channels it provides (Rowe, 1999; Alaçam et al., 2020). Perception of spoken dialogues and their transcripts has been shown to differ (Wallbridge et al., 2021, 2022), and findings for uni-modal data don't necessarily generalise to multi-modal settings (Bujok et al., 2022). We leave the extension of a similar analysis on the acoustic speech signal for future work.

References

Özge Alaçam, Xingshan Li, Wolfgang Menzel, and Tobias Staron. 2020. Crossmodal language comprehension—psycholinguistic insights and computational approaches. *Frontiers in neurorobotics*, 14:2.

Matthew P. Aylett and Alice Turk. 2004. The smooth signal redundancy hypothesis: A functional explanation for relationships between redundancy, prosodic prominence, and duration in spontaneous speech. *Language and Speech*, 47 1:31 – 56.

Ronny Bujok, Antje S. Meyer, and Hans Rutger Bosker. 2022. Visible lexical stress cues on the face do not influence audiovisual speech perception. In *Proceedings of Speech Prosody 2022*, pages 259–263.

Alessandra Cervone, Evgeny A. Stepanov, and Giuseppe Riccardi. 2018. Coherence models for dialogue. In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association*, pages 1011–1015. ISCA.

Nick Chater, Joshua B. Tenenbaum, and Alan Loddon Yuille. 2006. Probabilistic models of cognition: Conceptual foundations. *Trends in Cognitive Sciences*, 10:287–291.

Morten H Christiansen and Nick Chater. 2016. The now-or-never bottleneck: A fundamental constraint on language. *Behavioral and Brain Sciences*, 39 62:1–72.

Herbert H. Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22 1:1–39.

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

Gabriel Doyle and Michael Frank. 2015a. Audience size and contextual effects on information density in Twitter conversations. In *Proceedings of the 6th Workshop on Cognitive Modeling and Computational Linguistics*, pages 19–28, Denver, Colorado. Association for Computational Linguistics.

Gabriel Doyle and Michael Frank. 2015b. Shared common ground influences information density in microblog texts. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1587–1596, Denver, Colorado. Association for Computational Linguistics.

Erik Ekstedt and Gabriel Skantze. 2020. TurnGPT: a transformer-based language model for predicting turn-taking in spoken dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2981–2990, Online. Association for Computational Linguistics.

August Fenk and Günther Fenk. 1980. [constance and short-term memory - constance in speech information]. *Zeitschrift fur experimentelle und angewandte Psychologie*, 27 3:400–14.

Raquel Fernandez and Jonathan Ginzburg. 2002. Non-sentential utterances in dialogue: A: Corpus-based study. In *Proceedings of the Third SIGdial Workshop on Discourse and Dialogue*, pages 15–26, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

Simone Fuscone, Benoît Favre, and Laurent Prévot. 2020. Neural representations of dialogical history for improving upcoming turn acoustic parameters prediction. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 4203–4207. ISCA.

Dmitriy Genzel and Eugene Charniak. 2002. Entropy rate constancy in text. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 199–206, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Mario Giulianelli and Raquel Fernández. 2021. Analysing human strategies of information transmission as a function of discourse context. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 647–660, Online. Association for Computational Linguistics.

Mario Giulianelli, Arabella Sinclair, and Raquel Fernández. 2021. Is information density uniform in task-oriented dialogues? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8271–8283, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Godfrey, Edward Holliman, and Jane McDaniel. 1992. Switchboard: telephone speech corpus for research and development. In *ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1. IEEE Computer Society.

Adam Goodkind and Klinton Bicknell. 2018. Predictive power of word surprisal for reading times is a linear function of language model quality. In *Proceedings of the 8th Workshop on Cognitive Modeling and Computational Linguistics (CMCL 2018)*, pages 10–18, Salt Lake City, Utah. Association for Computational Linguistics.

Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. 2020. Speaker-aware BERT for multi-turn response selection in retrieval-based chatbots. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2041–2044. ACM.

John Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Janghoon Han, Taesuk Hong, Byoungjae Kim, Youngjoong Ko, and Jungyun Seo. 2021. Fine-grained post-training for improving retrieval-based dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1549–1558, Online. Association for Computational Linguistics.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

Johann Gottfried Herder. 1772. *Abhandlung ÃŒber den Ursprung der Sprache ("Essay on the Origin of Language")*, pages 121–124. De Gruyter (2015).

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multitask benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Falk Huettig and Ernesto Guerra. 2019. Effects of speech rate, preview time of visual context, and participant instructions reveal strong limits on prediction in language processing. *Brain Research*, 1706:196–208.

Falk Huettig, Ernesto Guerra, and Andrea Helo. 2020. Towards understanding the task dependency of embodied language processing: The influence of colour during language-vision interactions. *Journal of Cognition*, 3 1.

Tatsuya Kawahara, Takashi Yamaguchi, Koji Inoue, Katsuya Takanashi, and Nigel G. Ward. 2016. Prediction and generation of backchannel form for attentive listening systems. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association*, pages 2890–2894. ISCA.

Jey Han Lau, Alexander Clark, and Shalom Lappin. 2017. Grammaticality, Acceptability, and Probability: A Probabilistic View of Linguistic Knowledge. *Cognitive Science*, 41(5):1202–1241.

Hyunjae Lee, Jaewoong Yun, Hyunjin Choi, Seongho Joe, and Youngjune L. Gwon. 2021. Enhancing semantic understanding with self-supervised methods for abstractive dialogue summarization. In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 796–800. ISCA.

Roger Levy. 2008. Expectation-based syntactic comprehension. *Cognition*, 106 3:1126–1177.

Roger Levy. 2011. Integrating surprisal and uncertain-input models in online sentence comprehension: formal techniques and empirical results. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1055–1065, Portland, Oregon, USA. Association for Computational Linguistics.

Roger Levy and T. Florian Jaeger. 2006. Speakers optimize information density through syntactic reduction. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 849–856. MIT Press.

David Kellogg Lewis. 1969. *Convention: A Philosophical Study*. Cambridge, MA, USA: Wiley-Blackwell.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427, Online. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Clara Meister, Tiago Pimentel, Patrick Haller, Lena Jäger, Ryan Cotterell, and Roger Levy. 2021. Revisiting the Uniform Information Density hypothesis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 963–980, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Benjamin Newman, John Hewitt, Percy Liang, and Christopher D. Manning. 2020. The EOS decision and length extrapolation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 276–291, Online. Association for Computational Linguistics.

Mante S. Nieuwland and Jos J. A. Van Berkum. 2006. When peanuts fall in love: N400 evidence for the power of discourse. *Journal of Cognitive Neuroscience*, 18:1098–1111.

Bill Noble and Vladislav Maraev. 2021. Large-scale text pre-training helps with dialogue act recognition, but not without fine-tuning. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 166–172, Groningen, The Netherlands (online). Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Martin John Pickering and Simon Garrod. 2004. Toward a mechanistic psychology of dialogue. *Behavioral and Brain Sciences*, 27 2:169 – 190.

Yao Qiu, Jinchao Zhang, Huiying Ren, and Jie Zhou. 2021. Challenging instances are worth learning: Generating valuable negative samples for response selection training. *ArXiv*, abs/2109.06538.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Stephanie Richter and Rui Chaves. 2020. Investigating the role of verb frequency in factive and manner-of-speaking islands. In *Proceedings of the 42th Annual Meeting of the Cognitive Science Society*. cognitivesciencesociety.org.

Candy Rowe. 1999. Receiver psychology and the evolution of multicomponent signals. *Animal behaviour*, 58 5(5):921–931.

Martin Schrimpf, Idan Asher Blank, Greta Tuckute, Carina Kauf, Eghbal A. Hosseini, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. 2021. The neural architecture of language: Integrative modeling converges on predictive processing. *Proceedings of the National Academy of Sciences*, 118(45).

Elizabeth Shriberg, Andreas Stolcke, Daniel Jurafsky, Noah Coccaro, Marie Meteer, Rebecca Bates, Paul Taylor, Klaus Ries, Rachel Martin, and Carol Van Ess-Dykema. 1998. Can prosody aid the automatic classification of dialog acts in conversational speech? *Language and speech*, 41(3-4):443–492.

Gabriel Skantze. 2021. Turn-taking in conversational systems and human-robot interaction: A review. *Computer Speech & Language*, 67:101178.

Nathaniel J. Smith and R. Levy. 2013. The effect of word predictability on reading time is logarithmic. *Cognition*, 128 3:302–319.

Trang Tran. 2020. *Neural Models for Integrating Prosody in Spoken Language Understanding*. Ph.D. thesis, University of Washington.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 673–683, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alejandro Vega and Nigel G. Ward. 2009. Looking for entropy rate constancy in spoken dialog. Technical report, University of Texas at El Paso.

Sarenne Wallbridge, Peter Bell, and Catherine Lai. 2021. It's not what you said, it's how you said it: Discriminative perception of speech as a multichannel communication system. In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 2386–2390. ISCA.

Sarenne Wallbridge, Catherine Lai, and Peter Bell. 2022. Investigating perception of spoken dialogue acceptability through surprisal. In *Interspeech 2022, 23nd Annual Conference of the International Speech Communication Association, Incheon, South Korea*, pages 4506–4510.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Jason Wei, Clara Meister, and Ryan Cotterell. 2021. A cognitive regularizer for language modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5191–5202, Online. Association for Computational Linguistics.

Ethan Gotlieb Wilcox, Jon Gauthier, Jennifer Hu, Peng Qian, and Roger P. Levy. 2020. On the predictive power of neural language models for human realtime comprehension behavior. In *Proceedings of the 42nd Annual Meeting of the Cognitive Science Society*, page 1707–1713.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *ArXiv*, abs/1901.08149.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online. Association for Computational Linguistics.

Chien-Sheng Wu and Caiming Xiong. 2020. Probing task-oriented dialogue representation from language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5036–5051, Online. Association for Computational Linguistics.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017a. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Vancouver, Canada. Association for Computational Linguistics.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017b. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Vancouver, Canada. Association for Computational Linguistics.

Ruijian Xu, Chongyang Tao, Daxin Jiang, Xueliang Zhao, Dongyan Zhao, and Rui Yan. 2020. Learning an Effective Context-Response Matching Model with Self-Supervised Tasks for Retrieval-based Dialogues. *ArXiv preprint*, abs/2009.06265.

Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, Hai Zhao, and Gongshen Liu. 2018. Modeling multi-turn conversation with deep utterance aggregation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3740–3752, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. 2016. Multi-view response selection for human-computer conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 372–381, Austin, Texas. Association for Computational Linguistics.

A Language Models

A.1 Fine-tuning language models

We employ a range of language models to explore alignment with dialogue acceptability perception and response selection performance, including autoregressive, masked, and sequence-level models.

TurnGPT TurnGPT is a variant of GPT-2 (Radford et al., 2019), an autoregressive Transformer-based model (Vaswani et al., 2017) with 117M parameters. As reported in the main paper, our implementation is based on the pretrained GPT-2 model from the Transformers library (Wolf et al., 2020). We fine-tune the GPT-2 model following the procedure laid out by (Ekstedt and Skantze, 2020). Use our training portion of Switchboard, we train with the cross-entropy loss of the original GPT-2 model and default parameters. We fine-tune with early stopping, and achieve our best model in terms of validation loss after 2 epochs of training.

BERT-FP We employ BERT-FP to test retrieval-based dialogue models (Han et al., 2021). The architecture includes a simple linear MLP on top of a BERT model (Devlin et al., 2019). The original paper presents a two-stage training strategy for response selection. First, the model is post-trained using the task of utterance relevance classification, then it is fine-tuned towards response selection. To facilitate comparisons across architectures, we report experiments involving fine-tuning with no post-training in the main paper. However, we find that following the full procedure of post-training before fine-tuning increases response selection accuracy $R_{10}@1$ by 3 absolute points ($R_{10}@[1,2,5] = 0.705, 0.840, 0.975$).

As reported in the main paper, we implement this model using Pytorch and the pre-trained `bert-base-uncased` BERT model from the Transformers library. The full model has 109,483,777 trainable parameters. Fine-tuning is done with a cross-entropy loss. The response selection training set consists of an even split of positive and negative (context, response) pairs. Negative pairs are generated by sampling a turn from the elsewhere in the training set. To maintain consistency with BERT pre-training, joint (context, response) pairs are presented in the following format:

$$([CLS], c_1, [eos], ..., c_k, [eos], [SEP], r, [SEP]) \tag{1}$$

where the $[eos]$ token represents the end of a speaker's turn.

We follow the fine-tuning procedure from the BERT-FP paper. First, this involves splitting conversations into short (context,response) segments, resulting in 267,562 samples for training the cross- and bi-encoders. Second, fine-tuning is implemented using recent "BERTology" research to mitigate one of the pervasive issues with fine-tuning large, pre-trained LMs: training instability (Mosbach et al., 2021; Merchant et al., 2020). Compared to the BERT fine-tuning approach presented in Devlin et al. (2019), this involves using smaller learning rates with bias correction to avoid vanishing gradients early in training, and increasing the number of epochs. We use a batch size of 16 and make use of the Pytorch AdamW optimizer with an initial learning rate of $1e-5$. With early stopping, out best model in terms of validation loss is achieved at epoch 8.

Bi- and Cross-encoders For our ablation studies, we test a number architectural choices, including bi- versus cross-encoder set-ups, linear versus non-linear classifiers, and sequence representations from the BERT $[CLS]$ token or SBERT representations (`bert-base-uncased` for BERT-based models and `all-MiniLM-L6-v2` for SBERT; both models are both from the Transformers library). These are all build on the same codebase as our BERT-FP experiments. We fine-tune all of these architectures following the procedure described above for BERT-FP. As reported in the main paper, the classifiers consist of either a single linear-layer classifier, or three fully-connected ReLU layers with dropout and a final sigmoid layer.

To match the BERT input format, joint (context, response) pairs are presented in the following format for SBERT models:

$$(c_1, [eos], c_2, [eos], c_3, [eos], r) \tag{2}$$

A.2 TurnGPT surprisal definitions

We compute response scores from TurnGPT using various sequence-level surprisal definitions from previous works (Lau et al., 2017; Meister et al., 2021; Wallbridge et al., 2022). These include global surprisal metrics which account for surprisal at the turn level ($S_{total}, S_{mean}, S_{relative}$), and local netrics which provide more granular information at the token level (S_{max}, S_{var}). For a given (context c, response r) pair, each metric is defined as follows:

$$S_{total}(\mathbf{r}|\mathbf{c}) = \sum_{n=1}^{N}[S(r_n|\mathbf{r}_{<n}, \mathbf{c})]$$

$$S_{mean}(\mathbf{r}|\mathbf{c}) = \frac{1}{N}\sum_{n=1}^{N}[S(r_n|\mathbf{r}_{<n}, \mathbf{c})]$$

$$S_{relative}(\mathbf{r}|\mathbf{c}) = S_{mean}(\mathbf{r}|\mathbf{c}) - S_{mean}(\mathbf{r})$$

$$S_{max}(\mathbf{r}|\mathbf{c}) = \max[S(r_n|\mathbf{r}_{<n}, \mathbf{c})]$$

$$S_{var}(\mathbf{r}|\mathbf{c}) = \frac{1}{N-1}\sum_{n=2}^{N}[S(r_n|\mathbf{r}_{<n}, \mathbf{c}) - S(r_{n-1}|\mathbf{r}_{<n-1}, \mathbf{c})]^2$$

B Error Analysis: Corpus Excerpts

Tables 5, 6, 7, 8 contain examples of the stimuli collected in our previous work (Wallbridge et al., 2022) along with associated scores from both human participants, and the BERT-FP language model. Each stimuli consists of a variable number of conversational turns as context (1-4), and 10 potential upcoming responses (one of which is the true response in the particular conversation). Participants were presented with a context and a single response, and were tasked with rating how plausible the response was given the context.

Tables 5, 6, 7 show human misrankings while Table 8 contains the stimuli misranked by BERT-FP.

Table 5: Example of response selections stimuli ($sw3512\_14$) with human and BERT-FP scores.
Table (A) contains the set of conversational turns provided as context. Table (B) contains the set of 10 responses–these include the True upcoming turn for the given context and the 9 negative samples. Each response is displayed with it's true response label, the median plausibility score it recieved from participants (collected in our previous work (Wallbridge et al., 2022) study), and its BERT-FP response score.

(A) Context		
	Speaker A	that's right
	Speaker B	and then it's just so noisy that you can't visit
	Speaker A	oh i know
	Speaker B	know and normally when i'm eating out i you know with people and i wanna sit and talk i mean

(B) Responses	Human	BERT-FP	Transcript
Negative	4.50	0.57	mhm
True	4.40	0.99	that's the half the fun is the conversation right
Negative	3.83	0.60	dear
Negative	3.40	0.15	true you know and you have to start thinking about is it is it worth spending the money to go see it or shall i just wait
Negative	2.60	0.12	hello
Negative	2.00	0.17	but a lot of the stuff they do really you know evidently is pretty easy but i've just never
Negative	2.00	0.00	was lonely and she needed company for her mother and so she opened a nursing home and initially started with eight ladies
Negative	1.80	0.02	it's uh it's about an eight or nine hour drive really i make it in two days because i i don't push it
Negative	1.75	0.03	yeah when he played Danny Boy it just almost brought tears to your eyes because he can make that flute sing
Negative	1.40	0.01	yeah it's raining out here and i just steam cleaned my carpet today and i really don't wanna let the dog in

Table 6: Example of response selections stimuli (sw4040_20) with human and BERT-FP scores.

(A) Context		
	Speaker A	say you went back to school
	Speaker B	the wife went back to school
	Speaker A	i see uhuh
	Speaker B	he's been there for almost a year now

(B) Responses	Human	BERT-FP	Transcript
Negative	4.83	0.51	yeah
True	4.50	0.94	um gee i think where where does she go to school
Negative	3.00	0.06	what kind of things have you read
Negative	2.60	0.40	but that that's if
Negative	2.25	0.04	um she was referred to me by a couple of people and she turned out to be wonderful i couldn't have asked for anything better i don't think
Negative	2.20	0.01	it's so much easier to sit there and besides i can be doing other things and still listening to the news
Negative	2.00	0.01	you you say show music like Broadway musical type show music
Negative	2.00	0.02	and you they find everything i mean they find out everything about you they want to know your you know where you live what you do what you know and some of the questions
Negative	1.50	0.07	the starter was Bosch American so
Negative	1.17	0.00	that's there for direct yeah the direct sun beating on it yeah that's right

Table 7: Example of response selections stimuli (sw3959_32) with human and BERT-FP scores.

(A) Context		
	Speaker A	uhuh oh my
	Speaker B	down in Houston for um several years seven years and then uh my son is a CPA another has a business degree

(B) Responses	Human	BERT-FP	Transcript
Negative	4.80	0.63	before that did he go somewhere else or
True	3.60	0.93	my
Negative	2.33	0.00	that uh lots lots of new uh newsmen were created during that so it will be interesting to see what happens
Negative	2.00	0.00	you know with um the US funding Israel and you had the um Soviet Union funding the Arab countries
Negative	1.67	0.58	um yeah
Negative	1.67	0.02	uh what kind of lawn and garden work do you wind up doing
Negative	1.50	0.08	something that we're looking forward to
Negative	1.25	0.00	painted on T-shirts or sweat shirts at all
Negative	1.00	0.01	i do know of a way to get around the computer generated calls
Negative	1.00	0.00	we try to stay away from those things which might have uh salmonella in them

Table 8: Example of response selections stimuli (sw2303_59) with human and BERT-FP scores.

(A) Context		
	Speaker A	so they're going to be
	Speaker B	that's another and that's another interesting question should judges be elected or appointed

(B) Responses	Human	BERT-FP	Transcript
True	4.60	0.18	that's true that's true well
Negative	4.00	0.50	hm
Negative	2.80	0.90	idea
Negative	2.60	0.01	anyway what do you think we've gained from the space flights
Negative	2.40	0.49	makes you wish they had uh still had indentured servitude for this sort of thing
Negative	1.25	0.12	mean like those you know twenty thousand dollar toilets
Negative	1.17	0.14	well the the you know those little arms are supposed to twist almost any
Negative	1.00	0.00	yeah it's nothing but woods up here
Negative	1.00	0.00	it's uh it's about an eight or nine hour drive really i make it in two days because i i don't push it
Negative	1.00	0.00	i i i used to exercise at night and i found that you don't come home tired you come home with a new found energy so you

Methods for Measuring, Updating, and Visualizing Factual Beliefs in Language Models

Peter Hase[1,2] **Mona Diab**[1] **Asli Celikyilmaz**[1] **Xian Li**[1]
Zornitsa Kozareva[1] **Veselin Stoyanov**[1] **Mohit Bansal**[2] **Srinivasan Iyer**[1]
[1]Meta AI [2]UNC Chapel Hill
`{peter, mbansal}@cs.unc.edu`
`{mdiab, aslic, xianl, zori, ves, sviyer}@fb.com`

Abstract

Language models can memorize a considerable amount of factual information during pretraining that can be elicited through prompting or finetuning models on tasks like question answering. In this paper, we discuss approaches to measuring model factual beliefs, updating incorrect factual beliefs in models, and visualizing graphical relationships between factual beliefs. Our main contributions include: (1) new metrics for evaluating belief-updating methods focusing on the logical consistency of beliefs, (2) a training objective for Sequential, Local, and Generalizing updates (SLAG) that improves the performance of existing hypernetwork approaches, and (3) the introduction of the *belief graph*, a new form of visualization for language models that shows relationships between stored model beliefs. Our experiments suggest that models show only limited consistency between factual beliefs, but update methods can both fix incorrect model beliefs and greatly improve their consistency. Although off-the-shelf optimizers are surprisingly strong belief-updating baselines, our learned optimizers can outperform them in more difficult settings than have been considered in past work.[1]

1 Introduction

Pretrained language models have been shown to store a large amount of factual information about the world that can be elicited by cloze prompting (Petroni et al., 2019), few-shot learning (Brown et al., 2020), or finetuning models for question answering or true/false statement classification (Roberts et al., 2020). We refer to this kind of stored information as model *factual beliefs*.[2]

While pretrained models clearly store factual beliefs, it is not well understood how to efficiently edit the stored beliefs. Model editing is an exciting recent direction of research with several practical uses cases (Sinitsin et al., 2020; Zhu et al., 2020; De Cao et al., 2021; Mitchell et al., 2021). For LMs, these uses include updating factually inaccurate outputs and preventing other unwanted model outputs (e.g. toxic generated text) without expensive data curation and retraining efforts. These are important applications given that LMs (1) struggle with future data when trained on data from the past (Lazaridou et al., 2021; Dhingra et al., 2021), (2) often generate morally undesirable text (Gehman et al., 2020; Bender et al., 2021), and (3) simply give inaccurate outputs for tasks like question answering (Lin et al., 2021). Notably, there is good evidence that scaling models to larger sizes will not fix these particular problems or may even exacerbate them (Lazaridou et al., 2021; Gehman et al., 2020; Lin et al., 2021).

In the remainder of this paper, we present new methods for *measuring*, *updating*, and *visualizing* factual beliefs in LMs. We further describe each of these three contributions below. Figure 1 represents the core ideas behind measuring and updating factual beliefs, while belief visualization is done via *belief graphs* (shown later in Figure 2).

Measuring factual beliefs. We measure the degree to which LMs possess *consistent* factual beliefs using models finetuned on fact verification and question answering tasks. Beyond simply checking individual model responses, we want to assess the structural properties of model outputs: Are they consistent under paraphrase? Are they logically consistent? Does changing one belief correctly change other entailed beliefs? Does it erroneously change other unrelated beliefs? Past work has focused primarily on consistency under paraphrase (Elazar et al., 2021; De Cao et al., 2021; Mitchell et al., 2021). Here, we adapt data from Talmor

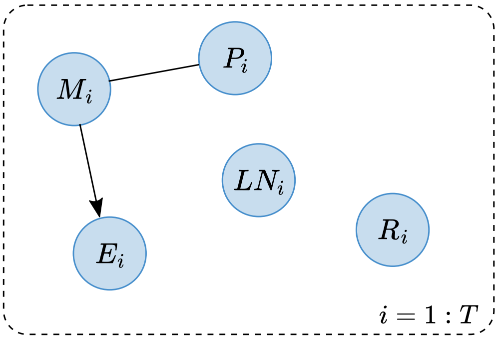
---

[1] Code is available at `https://github.com/peterbhase/SLAG-Belief-Updating`.

[2] We use the term *factual belief* rather than *knowledge* as in related work (Zhu et al., 2020; De Cao et al., 2021) because "belief" is a weaker term than "knowledge." In a traditional view of knowledge as Justified True Belief, it is more difficult to describe information as knowledge than as a belief (Schwitzgebel, 2019).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2714–2731
May 2-6, 2023

SLAG: Sequential, Local, and Generalizing Model Updates



```
$\theta_0 \leftarrow$ Language Model
for $t \in 1 : T$
    $\theta_t \leftarrow \mathrm{Update}(M_i, y_i^*, \theta_{t-1})$
```

M	(Main Input)	: A viper is a vertebrate.
E	(Entailed Data)	: A viper has a brain.
LN	(Local Neutral Data)	: A viper is venemous.
P	(Paraphase Data)	: Vipers are vertebrates.
R	(Random Data)	: Chile is a country.

Figure 1: Relying only on a Main Input M_i, we want to update a language model's weights in order to (1) change the output for M_i to a desired output y_i^*, (2) change the output for paraphrases of M_i, (3) appropriately change outputs for data E_i entailed by the tuple (M_i, y_i^*), and (4) *avoid* changing outputs for other logically neutral data LN_i, even if it is similar (local) to M_i. This is done iteratively for T requested updates.

et al. (2020) to measure consistency under entailment (including for contrapositives), and we use the Wikidata5m dataset (Wang et al., 2021) to construct logically neutral belief pairs for checking that models do treat these beliefs as independent.

Updating factual beliefs. We propose a Sequential, Local, and Generalizing belief update objective (SLAG) that substantially improves the performance of the comparable KNOWLEDGEEDITOR method from De Cao et al. (2021). KNOWLEDGEEDITOR is a learned optimizer that edits a model's weights to change its prediction on an input while satisfying other desiderata, like consistency under paraphrase. Principally, we identify more difficult training data for the learned optimizer, and we learn to apply many small edits rather than one big edit. These changes markedly improve the update success rate and lower the rate at which other beliefs are corrupted. We also find that KNOWLEDGEEDITOR almost totally fails when updating multiple beliefs in a row as opposed to a changing a single belief. However, by explicitly training the optimizer to update multiple beliefs sequentially, we recover much of the lost performance. Lastly, we advocate that these methods be evaluated for their ability to fix false or morally undesirable model beliefs, rather than to arbitrarily change beliefs to plausible alternatives as in past work (De Cao et al., 2021; Mitchell et al., 2021).

Visualizing belief graphs. We explore a new form of visualization for understanding language models, the *belief graph*. Given a set of factual beliefs, we construct belief graphs by changing each model belief and checking what other beliefs are sensitive to those changes. Each belief becomes a node, and directed edges between nodes show that updating one belief changes the other. We discuss graph metrics that help summarize the dependencies between model beliefs.

We summarize our main conclusions as follows:

1. ~100M parameter models exhibit limited belief-like qualities, as paraphrase consistency scores are under 70%, and models show mixed levels of consistency under entailment (Sec. 5.1).
2. Off-the-shelf optimizers are quite effective update methods, often outperforming learned optimizers when updating a single belief (Sec. 5.2).
3. When updating multiple beliefs in a row, performance greatly declines across methods, but SLAG can improve learned optimizers' performance beyond strong baselines (Sec. 5.2).
4. Belief graphs reveal many nonsensical dependencies between model beliefs, and they show the presence of "core" model beliefs that are connected to many other stored facts (Sec. 6).

2 Related Work

Measuring factual beliefs in language models. Much past work has explored how information is stored and represented in pretrained language models (Rogers et al., 2020). Petroni et al. (2019) provide evidence that LMs store relational information between entities, and Roberts et al. (2020) show that LMs can answer open-ended questions. Subsequent work has further explored how much knowledge is stored in LMs (Heinzerling and Inui, 2021). Most relevant to our work are studies from Talmor et al. (2020) and Elazar et al. (2021). Talmor et al. (2020) train LMs to perform True/False classification of factual claims, and they measure how beliefs correlate between entailed facts. We use their LeapOfThought data as a part of our SLAG objective (Eq. 1) and to measure model consistency under entailment before and after updating beliefs in models. Meanwhile, Elazar et al. (2021) measure the consistency of model predictions for paraphrased inputs. We adopt their metric for paraphrase consistency as a measure of belief. In other recent work, Kassner et al. (2021) measure consistency under entailment and paraphrase for factual belief with a new small-scale dataset, BeliefBank.

Updating factual beliefs in language models. Approaches to making targeted updates to model beliefs vary along a few dimensions. First is whether the methods alter model training or operate in a post-training setting. Sinitsin et al. (2020) use a meta-learning objective during training to encourage ease of editing afterwards. A larger family of methods perform post-training model updates: Dai et al. (2021) propose a hand-crafted algorithm that edits model weights, while Zhu et al. (2020) use projected gradient descent for batches of points. De Cao et al. (2021) train a hypernetwork (learned optimizer) that processes model gradients in order to produce a new model that (1) gives the desired output for an input, while (2) satisfying other objectives like minimizing changes in predictions for other data. Mitchell et al. (2021) focus on scaling up the underlying hypernetwork architecture, which is a complementary but orthogonal research direction that is not the focus of this paper. In a different approach, Kassner et al. (2021) "update" model beliefs by adding in relevant information to the input at test time. But this approach does not change the model weights and hence does not influence model outputs on all other potentially relevant inputs. Lastly, Meng et al. (2022) provide a specialized method focused on rank-one updates to MLP matrices in Transformer-based LMs, but they do not address the problem of updating multiple model beliefs and do not measure model consistency under entailment or unintended corruption of local neutral beliefs (metrics (5) and (6) in Sec. 3).

Visualizing factual beliefs in language models. We do not know of any prior work on visualizing dependencies between factual beliefs in language models, although our approach is notably inspired by older AI methods like Bayes Nets (Pearl, 2009). Different from Bayes Nets, we draw dependencies between two individual nodes when editing the model to change one belief also results in a change to the other belief, rather than there being a probabilistic model specifying the relationship between the two beliefs.

3 Updating Beliefs in Language Models

Here we describe the problem of updating model beliefs and our learned optimizer method. We also discuss metrics for measuring factual beliefs below, while our Belief Graphs are presented in Sec. 6.

Problem statement and metrics. We suppose we have a model $f_\theta = p_\theta(y|x)$ parametrized by θ. For an input x_i that has some undesired model output $\hat{y}_i = \arg\max_y p_\theta(y|x)$, we wish to obtain a new model θ^* that produces a desired output y_i^* for x_i. This new model θ^* should also fulfill a few other desiderata. As in past work (De Cao et al., 2021; Mitchell et al., 2021), we operationalize these desiderata in the following metrics:

1. Update Success Rate (*Main Input*): The proportion of Main Inputs x_i for which the updated model gives the desired output y_i^*.
2. Update Success Rate (*Paraphrase*): The proportion of paraphrases of x_i for which the updated model gives the same new prediction as it does for x_i (averaged across x_i).
3. Retain Rate (*All Data*): The proportion of the updated model's predictions which are unchanged for all data besides the Main Input.
4. Δ-Acc (*All Data*): The change in accuracy on all other data besides the Main Input.

In practice, Retain Rate (*All Data*) and Δ-Acc are computed with random subsets of a dataset, since these must be computed after every belief update. We add two metrics to those used in past work:

5. Update Success Rate (*Entailed Data*): The new model's accuracy on data that is logically entailed by the new Main Input prediction.
6. Retain Rate (*Local Neutral*): The proportion of the updated model's predictions which are unchanged for data that is similar to the Main Input but still logically neutral.

We use Update Success Rate (*Entailed Data*) to measure logical consistency for an updated model, since changing one belief entails changes in logically entailed beliefs. Retain Rate (*Local Neutral*)

Dataset	Data Type	Input	Label(s)
zsRE	Main Input Paraphrase	Player Ali Kanaan plays for what team? What team is Ali Kanaan associated with?	{Sporting Al Riyadi Beirut}
Wikidata5m	Main Input Paraphrase Local Neutral	Mary Good has relation 'award received' to Mary Lowe Good has relation 'winner of' to Mary Good has relation 'educated at' to	{Garvan-Olin Medal; Arkansas Women's Hall of Fame; etc.} {The University of Arkansas; U Arkansas; etc.}
FEVER	Main Input Main Input	Tardigrades are also known as space bears. The Lion belongs to the genus Vulpes.	True False
LeapOfThought	Main Input Entailed Data	A viper is a vertebrate. A viper has a brain.	True True

Table 1: Example datapoint from each dataset, and auxiliary data that accompanies the Main Input.

uses special Local Neutral data. Unlike random data, Local Neutral data is guaranteed to be logically independent of the Main Input, while still being similar (local) to it, which we ensure by using data with the same subject entity. Together, these six metrics better cover the criteria for belief outlined by Newen and Starzak (2020). We compute the metrics using data of the kind shown in Table 1.

Evaluation procedure. To date, methods have been evaluated on the basis of their ability to change model predictions for all data. Moreover, the desired labels $\{y_i^*\}_{i=1}^n$ on sequence prediction tasks have each been selected from the model's predictive beam search (De Cao et al., 2021; Mitchell et al., 2021). We propose for evaluation to focus on a more valuable but difficult setting: changing the predictions on incorrect points to be correct.

Sequential updates. The standard evaluation in past work is to update a single model belief, evaluate the new model, then rollback the update before repeating the process for each test point. We obtain sequential versions of all metrics by applying r model updates in a row before checking the metrics, meaning there are floor(n/r) measurements for a test set of n points. We consider it important to evaluate a sequential setting because, in practice, it is likely that model developers will want to update many factual beliefs of a trained model over time.

Belief updating method. As our base architecture, we use the KNOWLEDGEEDITOR architecture from De Cao et al. (2021), which is a hypernetwork that takes in model gradients as inputs and outputs a new update to apply to the model parameters. For further details of this method, we refer readers to Appendix A. Let it suffice for now to observe that a new model is given as a differentiable function

$$\theta^* = \theta + g_\phi(x_i, \hat{y}_i, y_i^*, \theta)$$

using the learned optimizer g_ϕ, current LM weights θ, Main Input x_i, current prediction $\hat{y}_i$, and desired model output y_i^*. Then, we can package the above update as $\theta^{(k+1)} = \theta^{(k)} + g_\phi(x_i, \hat{y}_i, y_i^*, \theta^{(k)})$, and obtain new model parameters via a looped update,

$$\begin{aligned}\theta^* &= \theta^{(k)} + \sum_{j=0}^{K-1} g_\phi(x_i, \hat{y}_i, y_i^*, \theta^{(k+j)}) \\ &= \text{Update}(x_i, \hat{y}_i, y_i^*, \theta^{(k)}; \phi, K)\end{aligned}$$

taking K small steps from initial parameters $\theta^{(k)}$. De Cao et al. (2021) use such a loop at test time; we incorporate the loop into training to align the train and test-time distributions.

Learned optimizer training. The training objective for KNOWLEDGEEDITOR includes differentiable terms corresponding to Update Success for the Main Input and paraphrases, as well as Retain Rate for all other data. We also consider terms for Update Success on entailed data and the Local Neutral Retain Rate, when this is possible given available data. The overall objective requires several kinds of additional data for each point, which we denote by $\mathcal{D}_R$ for other random data, $\mathcal{D}_{LN}$ for local neutral data, $\mathcal{D}_E$ for entailed data, and $\mathcal{D}_P$ for paraphrases of x_i. For a data point x_i with desired prediction y_i^*, the full objective is then:

$$\begin{aligned}\mathcal{L}(\phi; x_i, \hat{y}_i, y_i^*, \theta) &= \lambda_1 \mathcal{L}_{\text{Task}}(f_{\theta^*}(x_i), y_i^*) \\ &+ \lambda_2 \frac{1}{|\mathcal{D}_P|} \sum_{x_P \in \mathcal{D}_P} \mathcal{L}_{\text{Task}}(f_{\theta^*}(x_P), y_i^*) \\ &+ \lambda_3 \frac{1}{|\mathcal{D}_E|} \sum_{x_E, y_E \in \mathcal{D}_E} \mathcal{L}_{\text{Task}}(f_{\theta^*}(x_E), y_E) \\ &+ \lambda_4 \frac{1}{|\mathcal{D}_{LN}|} \sum_{x_{LN} \in \mathcal{D}_{LN}} \text{KL}(f_{\theta^*}(x_{LN}) || f_\theta(x_{LN})) \\ &+ \lambda_5 \frac{1}{|\mathcal{D}_R|} \sum_{x_R \in \mathcal{D}_R} \text{KL}(f_{\theta^*}(x_R) || f_\theta(x_R)) \qquad (1)\end{aligned}$$

where $\mathcal{L}_{\text{Task}}$ is the loss used to get gradients for f_θ. We use the Cross Entropy loss for binary classification and sequence-to-sequence tasks.

We optimize this objective w.r.t. ϕ using AdamW (Loshchilov and Hutter, 2019). To obtain update labels $\{y_i^*\}_{i=1}^n$, we always use the opposite class in binary classification. For sequence-to-sequence tasks, we use the correct label when $\hat{y}_i$ is incorrect, and when $\hat{y}_i$ is correct, we randomly select another label from the training data. This choice is in contrast to De Cao et al. (2021) and Mitchell et al. (2021), who use samples from the model beam search as update labels for all points.

SLAG objective. To prepare the update method for a sequential-update setting, we consider training g_ϕ to update multiple datapoints in a row. Using the per-datapoint loss in Eq. 1, we obtain our Sequential, Local, and Generalizing (SLAG) loss for a set of r Main Inputs $\mathcal{D} = \{x_i, \hat{y}_i, y_i^*\}_{i=1}^r$ as

$$\mathcal{L}_{\text{Sequential}}(\phi; \mathcal{D}, \theta_t) = \sum_{i=1}^{r} \mathcal{L}(\phi; x_i, \hat{y}_i, y_i^*, \theta_{t+i}) \quad (2)$$

where $\theta_{t+i} = \text{Update}(x_i, \hat{y}_i, y_i^*, \theta_{t+i-1}; \phi, K)$ are the model parameters obtained from updating on the first i points in $\mathcal{D}$ (starting from θ_t). This objective allows us to train g_ϕ to update multiple beliefs in a row. To ensure training with this objective is still efficient, we limit how far back through the update history we backpropagate when computing the gradient w.r.t. ϕ for each term in the RHS sum of Eq. 2. Each parameter vector θ_t depends on ϕ and θ_{t-1}. We always apply the stop-gradient function to the most recent vector θ_{t-1} to prevent backpropagating through it (visualized in Appendix Fig. 3). This choice allows our memory use to remain constant in r (see Appendix Fig. 4).

4 Experiment Setup

Datasets. We run experiments with four datasets (example data shown in Appendix Table 15). (1) FEVER includes 115,409 True/False factual claims (Thorne et al., 2018). We use the original test set of 10,444 points, and we randomly split the training data into 94,469 train points and 10,496 dev points. (2) zsRE includes 151,631 questions based on relational knowledge from Wikipedia, which we randomly shuffle into train/dev/test splits with 80/10/10% of the data (Levy et al., 2017). Talmor et al. (2020) introduce (3) the LeapOfThought dataset, consisting of factual claims that are entailed to be true or false depending on a context fact. We filter the data so that the context facts are unique, then shuffle the resulting 14,939 points into train/dev/test splits with 60/10/30% of the data.

	Belief Consistency ↑		
Dataset	Paraphrase	Entailed	Contrapos.
LeapOfThought	-	85.6 (1.1)	16.5 (2.7)
zsRE	69.5 (1.1)	-	-
Wikidata5m	25.8 (0.5)	-	-

Table 2: Belief metric results across datasets.

	Paraphrase Consistency ↑	
Dataset	Model Incorrect	Model Correct
zsRE	61.39 (1.33)	91.82 (1.17)
Wikidata5m	24.55 (0.48)	37.20 (2.06)

Table 3: Paraphrase consistency by the correctness of the model prediction on the Main Input.

In order to get Local Neutral data, we construct (4) a sequence prediction task using Wikidata5m, a relational knowledge base with over 20 million triplets (Wang et al., 2021). Each input consists of an entity e_1 and relation r, and the label is another entity e_2 that completes the triplet. All inputs come in pairs that share the same entity e_1 but use different relations with different labels. In general, the completion e_2 to the Main Input triplet (e_1, r_1, e_2) has no logical consequences for its paired input, (e_1, r_2, ?). The paired points are also local to the Main Input, i.e. they pertain to the same entity e_1 as the Main Input. We obtain four paraphrases for each Main Input using different aliases for the entity and synonyms of the relation. We construct a train set of 150k points and dev/test sets of 10k points each. See Appendix B for further details.

Models. We train five models with different random seeds for each dataset, using RoBERTa-base for binary tasks and BART-base for sequence-to-sequence tasks (accuracies in Appendix Table 14). For each of the five models, we train one learned optimizer using SLAG and one with the objective from De Cao et al. (2021), which we list as KE in tables below. Our model selection criterion is the mean of: average Update Success Rate (across data types), Retain Rate (only for Local Neutral data), and Δ-Acc for All Data. We tune the SLAG objective terms for each task separately (see Appendix Table 10 for final selections; results discussed in Appendix E). Other hyperparameters are given in Appendix B. To summarize the differences between SLAG and KNOWLEDGEEDITOR: (1) we use $K_{\text{train}} = K_{\text{test}}$ rather than $K_{\text{train}} = 1$; (2) we adopt

Single-Update Setting		Update Success Rate			Retain Rate		Δ-Acc
Dataset	Method	Main Input	Paraphrases	Entailed Data	Local Neutral	All Data	All Data
FEVER	AdamW	100 (0.0)	-	-	-	**98.80** (0.2)	**0.22** (0.1)
	KE	99.98 (<0.1)	-	-	-	98.28 (0.3)	-0.24 (0.1)
	SLAG	99.99 (<0.1)	-	-	-	98.41 (0.2)	-0.20 (0.1)
LeapOfThought	SGD	100 (0.0)	-	72.48 (4.6)	-	95.52 (0.4)	**1.23** (0.8)
	KE	99.78 (0.4)	-	74.48 (4.4)	-	93.50 (1.3)	-1.33 (1.1)
	SLAG	100 (0.0)	-	75.50 (4.3)	-	94.92 (1.4)	-1.31 (1.2)
zsRE	SGD	**99.36** (0.1)	94.44 (0.6)	-	-	74.73 (0.4)	-0.43 (0.1)
	KE	84.73 (1.4)	89.26 (1.8)	-	-	71.55 (2.4)	-2.19 (0.4)
	SLAG	94.29 (0.4)	94.71 (0.5)	-	-	**80.48** (1.3)	-0.29 (0.1)
Wikidata5m	SGD	**98.05** (0.3)	68.78 (0.8)	-	41.46 (1.0)	58.62 (0.6)	-1.97 (0.3)
	KE	74.57 (2.9)	58.05 (2.2)	-	40.84 (1.8)	53.58 (2.2)	-3.03 (0.5)
	SLAG	87.59 (0.6)	**80.70** (0.9)	-	**47.85** (1.0)	**63.51** (1.3)	-1.71 (0.3)

Table 4: Belief update metrics for off-the-shelf optimizers, KNOWLEDGEEDITOR (KE) from De Cao et al. (2021), and SLAG, with $r_{\text{test}} = 1$. Bolded numbers are the best in their group at a statistical significance threshold of $p < .05$ (or lower). Our SLAG objective improves over KE, but off-the-shelf optimizers perform surprisingly well.

training labels using real data labels rather than alternatives from the model's beam search; (3) our objective terms differ following tuning; and (4) we can optimize for updating multiple beliefs in a row.

Baselines. We use off-the-shelf optimizers as baselines. We tune the baseline hyperparameters separately for each dataset, selecting among several kinds of optimizers, learning rates, and the number of update steps. The selection criterion is the same as the criterion outlined for learned optimizers above. The resulting baselines are surprisingly strong (see Appendix Table 12 for final selections).

Hypothesis testing. We obtain 95% confidence intervals and perform hypothesis tests via block bootstrap, resampling model seeds and data points (Efron and Tibshirani, 1994). For ablation experiments, we run only one model seed per condition.

5 Experiment Results

5.1 Do LMs have consistent factual beliefs?

We measure Paraphrase Consistency, Entailment Acc, and Contrapositive Acc for finetuned task models. Paraphrase Cons. is the fraction of paraphrase pairs where the model produces the same output (Elazar et al., 2021). Entailment Acc is the model accuracy on data that is entailed by the Main Input. For LeapOfThought (see Table 1), "Main Input x_i is true" implies "entailed input x_E has label y_E," but the inverse ($\neg A \Rightarrow \neg B$) does not necessarily hold. Therefore, we compute Entailment Acc only where the Main Input prediction is correct. We do know that the contrapositive holds: "Entailed input x_E does not have label y_E" implies that "Main Input x_i is false." So for Contrapositive Acc, we measure how often the model follows this rule, when the antecedent holds of its prediction.

	Update Success Rate ↑		Δ-Acc ↑
Desired Label	Main Input	Paraphrase	All Data
Beam Label	97.41 (0.3)	97.03 (0.4)	-0.30 (0.1)
Correct Label	94.46 (0.7)	94.45 (0.7)	-0.24 (0.1)

Table 5: Evaluation difficulty by desired model output, for a learned optimizer trained with SLAG on zsRE.

Belief measurement results. Table 2 shows the belief metrics for each dataset. We find that ~100M parameter models show limited evidence of having consistent factual beliefs. Paraphrase consistency is 69.50% (± 1.09) for zsRE and much lower for Wikidata5m (25.84%±0.53). While entailment accuracy is high for LeapOfThought (85.63%±1.08), the model is consistent under the contrapositive only 16.51% (± 2.71) of the time. Overall, these results are not nearly as consistent as we would hope for factual beliefs to be. Interestingly, the metrics are much higher when the model prediction on the Main Input is correct (Table 3).

5.2 Can we update factual beliefs in LMs?

First, we compare two evaluation procedures for sequence prediction tasks: correcting model beliefs versus changing them to an alternative from the model's beam search. We do so for zsRE using SLAG. Next, we compare belief update performance between KNOWLEDGEEDITOR, SLAG, and off-the-shelf optimizers. We report results in single-update ($r_{\text{test}} = 1$) and sequential-update ($r_{\text{test}} = 10$) settings. See Appendix Fig. 5 for

Sequential-Update Setting		Update Success Rate			Retain Rate		Δ-Acc
Dataset	Method	Main Input	Paraphrases	Entailed Data	Local Neutral	All Data	All Data
FEVER	AdamW	92.81 (1.3)	-	-	-	**91.86** (1.4)	**1.16** (0.6)
	KE	74.13 (1.8)	-	-	-	39.86 (0.7)	-27.13 (1.3)
	SLAG	91.27 (2.9)	-	-	-	70.30 (5.8)	-11.96 (4.5)
LeapOfThought	SGD	100 (0.0)	-	**61.34** (5.0)	-	**82.62** (0.8)	**-4.93** (1.0)
	KE	96.14 (2.3)	-	49.27 (6.0)	-	72.45 (0.9)	-15.03 (1.0)
	SLAG	100 (0.0)	-	50.46 (5.5)	-	74.02 (1.1)	-13.03 (1.3)
zsRE	SGD	82.71 (0.6)	90.81 (0.7)	-	-	40.49 (0.6)	-2.38 (0.3)
	KE	0.10 (<0.1)	36.55 (1.4)	-	-	0.05 (<0.1)	-20.98 (0.7)
	SLAG	**87.57** (0.6)	**92.20** (0.7)	-	-	**47.19** (0.7)	**-1.74** (0.3)
Wikidata5m	SGD	56.82 (0.8)	54.49 (0.7)	-	6.40 (0.4)	26.37 (0.6)	-3.96 (0.4)
	KE	0 (0.0)	40.84 (0.9)	-	0 (0.0)	0 (0.0)	-10.05 (0.6)
	SLAG	**58.27** (1.0)	**65.51** (0.9)	-	**7.36** (0.5)	**27.76** (0.7)	-3.62 (0.4)

Table 6: Belief update results when a model is sequentially updated r_{test}=10 times. Here, SLAG uses $r_{\text{train}}=R$. On sequence prediction tasks in this setting, SLAG can outperform the off-the-shelf optimizers across metrics.

an ablation across r_{test}.

Correcting beliefs vs. changing factual beliefs. Given the results in Table 5, we find that correcting model outputs is harder than simply changing them to a plausible alternative. Update Success rises by a full 2.96 (±0.48; p<1e−4) points for Main Inputs and 2.58 (±0.81; p<1e−4) for Paraphrases, while Δ-Acc is virtually unchanged. This suggests that that past work has overestimated the efficacy of belief update methods for actually fixing models. Henceforth we evaluate methods according to their ability to update model beliefs to be true.

Update method results (single update). Table 4 shows the results in a single-update setting. First, we find that off-the-shelf optimizers are very effective across the board. The baselines show Main Input Update Success Rates of 98%+ across tasks with competitive or even positive Δ-Acc scores.[3] When strongly tuned, these baselines outperform learned optimizers on most metrics here.

However, SLAG surpasses the baselines in a few places. All Data Retain Rate on zsRE rises by 5.77 points (±1.43; p<1e−4), and on Wikidata5m Paraphrase Update Success rises by 11.92 (±1.20; p<1e−4) and the Local Neutral Retain Rate by 6.40 (±1.41; p<1e−4). SLAG also greatly improves over KE for sequence prediction tasks.

Interestingly, we observe that belief updates greatly improve paraphrase consistency and entailment accuracy (SLAG results in Table 7). Updates improve Paraphrase consistency by 33.14±1.46 on zsRE and 59.87±1.09 on Wikidata5m, while Entailment Acc rises by 17.20±7.10 points.

Metric	Before Update	After Update
Entailment Acc	58.30 (5.7)	75.50 (4.3)
Para. Cons (zsRE)	61.39 (1.3)	94.53 (0.6)
Para. Cons (Wiki)	24.69 (0.5)	84.56 (0.9)

Table 7: Entailment Acc and Paraphrase Consistency rise greatly after model updates to incorrect points.

Update method results (sequential updates). We give results for a sequential update setting (r_{test}=10) in Table 6. Immediately we see this is a much more difficult evaluation, as metrics are generally far lower for each dataset. Next, we observe that learned optimizers with SLAG (r_{train}=10) outperform baselines on sequence prediction tasks. On zsRE, we improve Update Success for Main Inputs by 4.86 (±0.83; p=1e−4) and for Paraphrases by 1.39 (±0.93; p=.004), with better Δ-Acc by 0.64 (±0.35; p=.0005). Improvements trend in the same direction for Wikidata5m and are all statistically significant except for the gain in Δ-Acc. The jump on Paraphrases in particular is large (11.02±1.17; p<1e−4). In comparison, using the KNOWLEDGEEDITOR training objective leads to drastic drops in performance.

Learned optimizers still struggle compared to baselines *on binary datasets*. Here, AdamW and SGD achieve high update update success with much better Δ-Acc scores, by 13.12 (±4.51; p=1e−4) on FEVER and 8.16 (±1.63; p=1e−4) on LeapOfThought.

---

[3] Positive Δ-Acc values are possibly due to distribution shift in the test split. In FEVER, for instance, the train and dev data are 73% True, while test data is 50% True. On the dev split, AdamW achieves a negative Δ-Acc, -0.18 (±0.11).

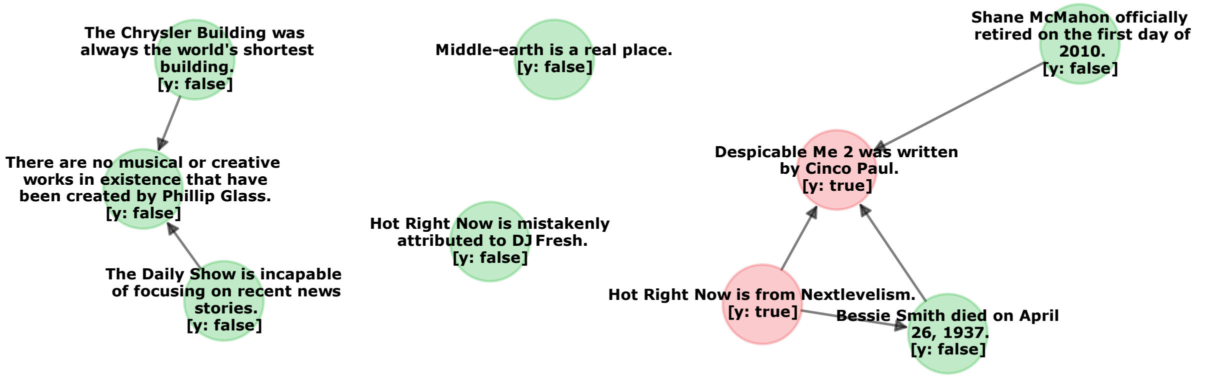


Figure 2: A non-random subgraph of the belief graph for a model trained on FEVER. Directed edges from u to v indicate that changing the model belief in u causes the belief in v to change. The ground-truth label is given in brackets for each point, and node color shows the model's accuracy before any updates (green=correct).

6 Belief Graphs

We now construct *belief graphs* to better understand the connections between model beliefs. We form a graph from a set of datapoints by updating each prediction and checking what other predictions change. We represent each datapoint as its own node in a belief graph. Whenever updating a datapoint u changes the prediction for point v, we draw a directed edge from u to v. Following Sec. 5.2, we use off-the-shelf optimizers to change the model output to the opposite of its original prediction for every datapoint. For FEVER we obtain a graph of 10,444 nodes, and for LeapOfThought we obtain a graph with 8642 nodes, which is double the test set size because we include both Main Inputs and Entailed Data as their own nodes.

We visualize part of a belief graph in Fig. 2. This figure shows a non-random subgraph intended to give a representative view of the data (we give three random subgraphs in Appendix E). On inspection, we do not see any clear reasons for beliefs being connected or not connected. We come to the same conclusion looking at other random subgraphs (see Appendix Figures 9, 10, and 11). Whether or not changing one belief changes another appears essentially random, which is a novel negative result on the organization of internal model beliefs. However, we do observe some aggregate trends. First, it appears that incorrect predictions are the most sensitive to model updates. On FEVER, incorrect beliefs change around 4% of the time when other beliefs are updated, while correct beliefs change only 2.5% of the time. Second, we find that Local Neutral beliefs are much harder to avoid changing than simply random data. On Wikidata5m (Table 4), we observe that the Retain Rate on All Data is 61.51±1.33, while for Local Neutral data it is a full 15.66 points lower.

	Dataset	
Metric	FEVER	LeapOfThought
# Nodes	10,444	8,642
% Edgeless	0.0	0.0
# Edges Total	1.88m	9.71m
# In Edges (95th perc.)	1,088	5,347
# Out Edges (95th perc.)	390	3,087
% Update-Transitivity	66.64	24.38*

Table 8: Belief graph summary statistics. *We compute Update-Transitivity for LeapOfThought with $n = 4000$ points due to computational cost.

We highlight a few summary statistics here from Table 8 for a broader view of the graphs. First, % Edgeless is the proportion of nodes which have no in or out edges. Since this is 0 for both datasets, every belief can be changed by editing the right belief. # In Edges is the number of in edges at the 95th percentile, meaning 5% of beliefs have more in edges than this value, and the same holds of # Out Edges. These values grow to a rather large fraction of the overall datasets, suggesting that (1) some beliefs are sensitive to changes in a large fraction of all beliefs, and (2) some beliefs are influential to hundreds of other beliefs when changed. Interestingly, this implies that some factual beliefs are "core" beliefs in the model, such that changing these individual beliefs requires greatly changing the overall distribution of factual beliefs in the model. Lastly, % Update-Transitivity represents the answer to the question: if updating belief A changes belief B, and updating belief B changes belief C, what proportion of the time does updating A change C? For these datasets, a logically consistent model should display 100% Update-Transitivity (see Appendix D for a caveat on this metric). We find that belief updates often yield intransitive results for both datasets, another negative result for belief consistency. It would be valuable for future work to extend this analysis of belief graphs to explore why language models demonstrate these surprising connections and inconsistencies between beliefs.

7 Conclusion

We first measure the presence of consistent factual beliefs in language models, and we propose to evaluate learned optimizers for whether they can make model beliefs more truthful. Then we show that our SLAG objective greatly improves learned optimizer performance, outperforming off-the-shelf optimizers when updating multiple model beliefs in a row. Finally, we introduce *belief graphs* to visualize connections between model beliefs. We find that model beliefs are highly interconnected, with some "core" beliefs influencing hundreds of other beliefs.

Ethics Statement

Belief update methods may be used to either correct undesired beliefs or induce problematic beliefs in LMs, and it is not clear whether these capabilities could be separated. We propose to evaluate methods only on the basis of their ability to correct mistaken model beliefs, but the malicious use case remains. We are uncertain about how a bad belief would influence the general behavior of a model (e.g. answers to many questions), but it is possible that a belief update method could instill bad beliefs in a capable LM with far-reaching implications for model behavior. That said, we hope that these methods will instead be used to update undesirable moral, social, and factual beliefs in large LMs.

Limitations

We note a few limitations of our work:

(1) Neural learned optimizers require large amounts of training data to successfully edit even a few model beliefs at test time.

(2) Our experiments are limited by available datasets in terms of both metrics we can calculate and objectives we can optimize for. There is also some noise in each dataset which we catalogue in Appendix C.

(3) We conduct experiments with $\sim$100M parameter models as in past work. While the belief-updating problem is still clearly unsolved given our results, it will also be valuable for future work to scale to larger models which may exhibit more consistent factual beliefs. That said, we believe our contributions are still valuable since our metrics, objectives, and belief visualization method can all be easily applied to larger models, and hypernetworks have already been extended to work with larger models (Mitchell et al., 2021).

(4) Currently, models may have seemingly random interdependencies between factual beliefs, limiting the insights available from our belief graphs. We believe that as models become more consistent and more truthful, the usefulness of belief graphs as a tool for understanding connections between beliefs will increase.

(5) Lastly, we do not currently account for uncertainty in factual beliefs. The data we use comes in the form of declarative statements and answers to questions which take what is called a veridical stance toward a proposition, displaying a "full commitment" to that proposition's truthfulness (Giannakidou and Mari, 2020). It will be valuable for future work to explore two dimensions of uncertainty in beliefs: (1) expression of uncertainty in language, via partial or trivial commitments (like "X might be Y") and (2) expression of uncertainty mathematically, via probabilities assigned by a model to utterances or True/False values. In this paper we treat a belief as "updated" when the model output changes, but this ignores any underlying change in the distribution $p_\theta(y|x)$ that could occur even if its mode does not change.

References

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, and Furu

Wei. 2021. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *EMNLP*, pages 6491–6506. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. 2021. Time-aware language models as temporal knowledge bases. *arXiv preprint arXiv:2106.15110*.

Bradley Efron and Robert J Tibshirani. 1994. *An Introduction to the Bootstrap*. CRC press.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. In *Findings of EMNLP*.

Anastasia Giannakidou and Alda Mari. 2020. A linguistic framework for knowledge, belief, and veridicality judgement. *HAL*.

Benjamin Heinzerling and Kentaro Inui. 2021. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1772–1791, Online. Association for Computational Linguistics.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. 2021. Beliefbank: Adding memory to a pre-trained language model for a systematic notion of belief. *arXiv preprint arXiv:2109.14723*.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Sebastian Ruder, Dani Yogatama, et al. 2021. Mind the gap: Assessing temporal generalization in neural language models. In *NeurIPS*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual knowledge in gpt. *arXiv preprint arXiv:2202.05262*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2021. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*.

Albert Newen and Tobias Starzak. 2020. How to ascribe beliefs to animals. *Mind & Language*.

J. Pearl. 2009. *Causality*. Cambridge University Press.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Eric Schwitzgebel. 2019. Belief. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, Fall 2019 edition. Metaphysics Research Lab, Stanford University.

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitriy Pyrkin, Sergei Popov, and Artem Babenko. 2020. Editable neural networks. In *ICLR*.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. In *NeurIPS*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, Volume 1 (Long Papers), pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*.

A Learned Optimizer Details

Architecture. KNOWLEDGEEDITOR is a learned optimizer $g : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \times \Theta \to \Theta$ that produces new model weights by applying an adjusted gradient step to a model. For reference, we give a glossary of symbols used here in Table 9. For additional details beyond what is presented here, we refer readers to De Cao et al. (2021).

At a high level, g_ϕ first encodes an input x_i and requested prediction change into a vector h, then processes h into two low-rank matrices A and B that are used to transform the model gradient on x_i, $\nabla_\theta \mathcal{L}(x_i, y_i^*)$. For Transformer models, the method edits only attention and feed-forward weights, so all model gradients match the shape of an associated weight matrix of shape $d_1 \times d_2$. Formally, a new model θ^* is obtained using a learned optimizer g_ϕ as follows:

$$\begin{aligned} h &= \text{LSTM}([x; \hat{y}; y^*]) \\ \{u, v, \gamma, \delta\} &= \{\text{MLP}_i(h)\}_{i=1}^4 \\ A &= \text{softmax}(u)v^T \\ B &= \text{softmax}(\gamma)\delta^T \\ \eta &= \sigma(\text{MLP}(h)) \\ \theta^* &= \theta + \eta(A \circ \nabla_\theta \mathcal{L}(x_i, y_i^*) + B) \end{aligned}$$

where ϕ consists of all LSTM and MLP parameters.

Training Algorithm. The learned optimizer objective is optimized w.r.t. ϕ with AdamW through a standard procedure of randomly sampling minibatches without replacement (Loshchilov and Hutter, 2019). Within each batch, one datapoint is randomly selected as the Main Input, and the remaining points are used as $\mathcal{D}_R$. To obtain update labels $\{y_i^*\}_{i=1}^n$, we always use the opposite class in binary classification. For sequence-to-sequence tasks, we use the correct label when $\hat{y}_i$ is incorrect, and when $\hat{y}_i$ is correct, we randomly select another label from the training data. This choice is in contrast to De Cao et al. (2021) and Mitchell et al. (2021), who use samples from the model beam search as update labels for all points.

Symbol Glossary	
f_θ	Language Model
g_ϕ	Learned optimizer
x_i	Main Input
$\hat{y}_i$	LM output on x_i
y_i^*	Desired output
$\nabla_\theta \mathcal{L}(x_i, y_i^*)$	Gradient of LM
$\text{Update}(x_i, \hat{y}_i, y_i^*, \theta)$	Update one LM belief
$\mathcal{L}(\phi; x_i, \hat{y}_i, y_i^*, \theta)$	Belief update objective for x_i
$\mathcal{L}_{\text{Sequential}}(\phi; \mathcal{D}, \theta_t)$	Sequential objective (SLAG)
K	# gradient steps in Update$(\cdot)$
r	# beliefs updated in $\mathcal{L}_{\text{Sequential}}$

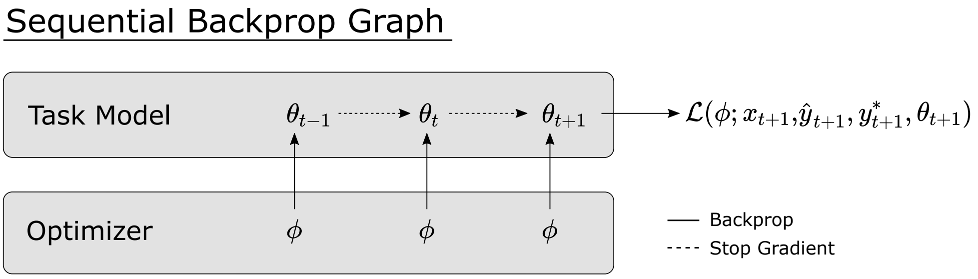
Table 9: Symbol descriptions for the learned optimizer.

B Additional Training Details

B.1 Compute Costs.

Learned optimizer memory. The hypernetwork has 92m trainable parameters for RoBERTa-base (which is 125m parameters), and 105m parameters for BART-base (which is 139m parameters). To increase training efficiency, we limit how far into the task model history we backpropagate. As shown in Fig. 3, when backpropagating through task model parameters $\theta_t = \theta_{t-1} + \text{Update}(x_i, \hat{y}_i, y_i^*, \theta_{t-1}; \phi)$, we continue backpropagating through $\text{Update}(x_i, \hat{y}_i, y_i^*, \theta_{t-1})$ but *not* θ_{t-1}, which is also dependent on ϕ. That is, we apply a stop-gradient function to θ_{t-1}. This way, we compute the derivative $\nabla_\phi \text{Update}(x_i, \hat{y}_i, y_i^*, \theta_t; \phi)$. only once for each t, rather than recomputing these gradients for all subsequent time steps. These choices allow the memory use of our training algorithm to remain constant in r. We make the same choice for our K looped steps in a single application of the Update function, so the gradient for the update at step k depends only on $g_\phi(x_i, \hat{y}_i, y_i^*, \theta^{(k)})$ and not $\theta^{(k-1)}$. See Fig. 4 for a graph of memory use depending on r and k.

Experiment runtimes. We now give runtimes for experiments in the paper. Building the belief graphs takes 25 hours for FEVER ($n = 10,444$) and 17.5 hours for LeapOfThought ($n = 8642$) on an NVIDIA RTX 2080 GPU. Computing summary statistics for graphs takes 3 hours on FEVER and 3 hours for LeapOfThought for statistics be-



$$\mathcal{L}_{\text{Sequential}}(\phi; \mathcal{D}, \theta_t) = \sum_{i=1}^{r} \mathcal{L}(\phi; x_i, \hat{y}_i, y^*_i, \theta_{t+i}) \qquad \theta_{t+1} = \theta_t + \text{Update}(x_i, \hat{y}_i, y^*_i, \theta_t; \phi)$$

Figure 3: The backpropagation graph for sequential model updates.

sides Update-Transitivity. We compute Update-Transitivity for LeapOfThought with a subset of 4000 points, which takes 45 hours.

All other experiments are run on a NVIDIA V100 32GB GPU. Training the task models takes 7 minutes for LeapOfThought, 45 minutes for FEVER, 4 hours for zsRE, and 10 hours for Wikidata5m. Training the learned optimizer with $r = 1$ takes 2.3 hours for LeapOfThought, 5 hours for FEVER, 9.5 hours for zsRE, and 16 hours for Wikidata5m. Training the learned optimizer with $r = 10$ takes 53 minutes for LeapOfThought, 2.9 hours for FEVER, 7 hours for zsRE, and 12.5 hours for Wikidata5m. Computing update statistics with the off-the-shelf optimizers with $r = 1$ takes 4 minutes for LeapOfThought, 30 minutes for FEVER, 2.3 hours for zsRE, and 3.9 hours for Wikidata5m. With $r = 10$, the baselines require 1 minute for LeapOfThought, 15 minutes for FEVER, 54 minutes for zsRE, and 1.8 hours for Wikidata5m. Total runtimes for each experiment should take into account multiple conditions and multiple seeds of each model being run.

B.2 Hyperparameters and Objective Terms.

Training hyperparameters. We fit our RoBERTa-base and BART-base task models to their respective datasets with the following hyperparameters: We train for 10 epochs on the binary tasks, and 20 for the sequence-to-sequence tasks. When predicting with BART-base, we use a beam search with width 5. In each case, we use AdamW from `torch.optim` with a LR of 1e-5 and weight decay of 1e-4. We select the best model according to the best dev set accuracy, checkpointing after each training epoch. The learned optimizers are optimized with AdamW, using a learning rate of 3e-4 and weight decay of 0. We train the learned optimizer for 5 epochs on each dataset except for LeapOfThought, which we train for 10 epochs given its smaller size. The learned optimizers are also selected based on dev set performance, with checkpointing after each training epoch. Their selection criterion is a raw average of Update Success Rate (averaged over each kind of data), Retain Rate (*Local Neutral*) and Δ-Acc, with terms dropped when they cannot be computed given the available data. Note that dev epochs with zsRE and Wikidata5m are fairly slow, so in order to speed up our experiments we compute dev epochs with a subset of 4000 dev points.

Dataset	r_{test}	K	Objective
FEVER	1	5	Main
	10	1	Main
LeapOfThought	1	5	Main
	10	1	Main
zsRE	1	5	Main
	10	5	Main
Wikidata5m	1	5	Main+Para
	10	5	Main+Para

Table 10: Final hyperparameters and objective terms of the learned optimizer for each task.

Learned optimizer. We give the final hyperparameter and objective terms used in each experiment in Table 10. Our objective ablation is given in 17, and we select the best performing condition for each dataset according to dev set performance, using the same selection criterion outlined previously. We keep all weight coefficients λ_i equal rather than tuning them. Main refers to the first term in Eq. 1, plus the KL term with random data. We use $K_{\text{train}} \leq 5$ for all experiments. For results across K values on zsRE, see Fig. 8.

Baseline update method. We tune a baseline off-the-shelf optimizer separately for each dataset, using $r_{\text{test}} = 1$. Our performance criterion is the

Relation	% Test Data
Place of Birth	11.00
Award Received	11.00
Cause of Death	5.66
Place of Death	11.00
Place of Burial	8.33
Educated At	11.00
Child	11.00
Occupation	11.00
Spouse	11.00
Sibling	9.01

Table 11: Wikidata relations and their proportion of the test data.

Dataset	Optimizer	LR	Num. Steps
FEVER	AdamW	1e-6	100
LeapOfThought	SGD	1e-2	100
zsRE	SGD	1e-1	10
Wikidata5m	SGD	1e-1	10

Table 12: Final hyperparameters of the baseline update method for each task.

same as with learned optimizers, a raw average of Update Success Rate (averaged over each kind of data), Retain Rate (*Local Neutral*) and Δ-Acc. The grid search is over the following parameters: The off-the-shelf optimizers are from `torch.optim` and include {AdamW, SGD, and RMSProp} with default arguments (except for the learning rate). We consider a number of maximum steps in {5, 10, 100}. The learning rates we consider depend on the optimizer: {1e-4, 1e-5, 1e-6} for AdamW, {1e-4, 1e-5, 1e-6} for RMSProp, and {1e-1, 1e-2, 1e-3} for SGD. The LR ranges were selected after some initial manual exploration of the space. Our final hyperparameter values are shown in Table 12 for each dataset. For comparison, De Cao et al. (2021) use RMSProp with 100 update steps. The LR for zsRE and Wikidata5m may seem quite high, but this is the condition that actually does the least damage to the model's accuracy on other data, Δ-Acc. The baseline optimizes all of the trainable parameters in the language model, unlike the learned optimizer which optimizes only attention and feedforward weights for purposes of parameter efficiency.

B.3 Wikidata5m Additional Details.

We construct four paraphrases per Main Input by selecting from a set of alternative phrasings for the entity and relation in the Main Input. The syntax for each paraphrase follows the same simple template as the Main Input, in contrast to zsRE where syntax differs between paraphrases. A couple details remain. Some relations are one-to-many, and therefore we accumulate valid completing entities from the data as possible answers; later we compute accuracy as an exact match with any possible answer. All 10 relations appear in each split of the data. Only 33.80% and 37.18% of the entities in the dev and test splits are seen in the training data, though we do not find that models perform better on entities seen in training.

B.4 LeapOfThought Additional Details

The LeapOfThought dataset consists of a fact and a claim for each datapoint, where the truth of the fact implies that the claim has label y_i (True/False). All of the facts in the data are true, while half of the claims are true and half are false. When training the learned optimizer, we treat the the facts as the Main Input when training the learned optimizer and claims as entailed data. When training the True/False classifier, we fit to the claims, for which test accuracy is 83.65 ($\pm$ 1.05). This seems to generalize well to the facts, as test accuracy here is 93.66 ($\pm$0.87), although as the low contrapositive accuracy suggests (Table 3), the model seems to be too prone to predicting true for this data.

Since very few of the Main Inputs are predicted as false, we run into a small dilemma when fitting the learned optimizer with the use of the entailed data objective term. The entailment between fact and claim only holds when the fact is true, so we can only compute the objective when updating a point from false to true. This ends up being less than 10% of the training data. We ultimately choose to oversample points that fit this description during training of the learned optimizer, which allows the learned optimizer to fully fit to the entailed data. Also note that during learned optimizer training, we include Entailed Data *from other data points besides the Main Input* in the KL term in Eq. 1, and we measure Δ-Acc using both Main Inputs and Entailed Data.

C Dataset Sources and Noise

Here we give sources and licenses for each dataset, and we document some shortcomings of each dataset, with reference to examples in Table 15.

Dataset sources and licenses. FEVER and zsRE are available through the KILT[4] resource and are

---

[4] https://github.com/facebookresearch/KILT/?fbclid=IwAR2WiFkl-7KLIQAoNI9bJgBVKWgsAQEDV342vV5_PcsKA881vpuXaELKBz0

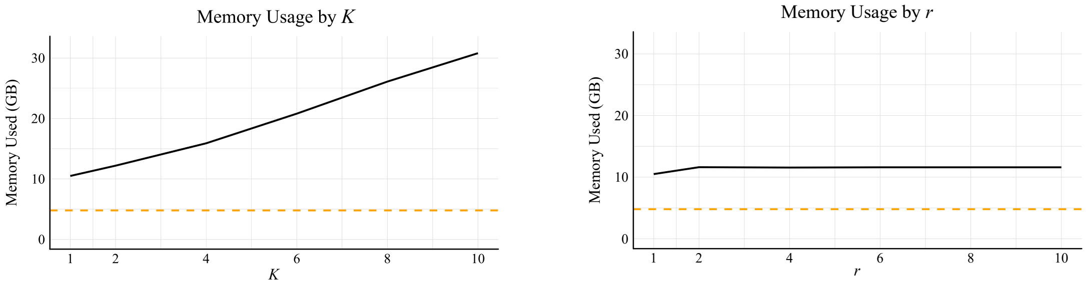


Figure 4: Training memory usage in terms of K and r hyperparameters in our implementation, for a learned optimizer trained for a BART-base model on zsRE, using a batch size of 16. For comparison, the orange dashed line shows the memory use of training the BART-base model on zsRE, using the same batch size. Our use of the stop-gradient function limits the growth of runtime and memory w.r.t. both K and r. By accumulating gradients across points, memory w.r.t. r is kept constant. The same trick could be applied to the K looped gradient steps inside the Update function, at the trade-off of backpropagating K times per point rather than one time.

Ours	De Cao et al. (2021)	Mitchell et al. (2021)
Update Success Rate (*Main Input*)	Success rate	Edit success
Update Success Rate (*Paraphrase*)	Equivalence accuracy	Edit success
Update Success Rate (*Entailed Data*)	-	-
Retain Rate (*Local Neutral*)	-	-
Retain Rate (*All Data*)	Retain accuracy	-
Δ-Acc (*All Data*)	Performance deterioration	Drawdown

Table 13: A glossary of terms used in work on model update methods. Note metrics are not always calculated in exactly the same way. For instance, Performance deterioration is a ratio in accuracies rather than difference in accuracies, and edit success from Mitchell et al. (2021) combines two metrics in our case. The performance metric in Zhu et al. (2020) is an average of Update Success Rate (*Main Input*) and Δ-Acc.

available under the MIT license (Petroni et al., 2021). LeapOfThought data can be constructed through their available code[5] and is also available under the MIT license. The source data for Wikidata5m data can be downloaded through the KEPLER[6] code repository (Wang et al., 2021) and is available under the MIT license. Use of each dataset is in accordance with their intended licensed uses. The zsRE and Wikidata5m datasets do refer to people by name as they reference public figures on Wikipedia. All datasets are in English.

FEVER. Some claims are slightly vague or ambiguous when taken on their own. For instance "Doug Ducey was the CEO of Cold Stone Creamery and offered many opportunities to new hires" is rated True, though this will depend heavily on what one thinks "many opportunities" means. Similar whether or not "L.A. Guns is a tattoo shop" depends on which "L.A. Guns" one is referring to, the tattoo shop or metal band. Of course, this is a generic issue of language, and not unique to this dataset. Some inputs seem to be a matter of person opinion: "Los Angeles is known for its food" is rated False.

LeapOfThought. Many examples use an "is a" relation, producing sentences like "A sunlight is a good health." This could be more false than true, but it's a fairly nonsensical statement to begin with. There are also other nonsensical or vague examples in the data: "A friar is the opposite of mineral" is labeled False. "A detective desires equal opportunity." is labeled True. It is not immediately clear what conditions would make these statements true or false.

zsRE. Some questions invoke potentially one-to-many or temporally dependent relations, though there is only one ground-truth answer per question in this dataset. For instance, a paraphrase of the question about Gifford Pinchot in Table 15 is: "What disease did Gifford Pinchot have?" A person might have had many diseases over their life which could all be valid responses. The answer is especially ambiguous for spatial relations, where a valid answer might refer to a city, region, country, province, or continent.

Wikidata. Aliases sometimes vary greatly even as they refer to the same person, or they are simply noisy. For example, as shown in Table 15, "SusunW" appears in an entity name, but this is actually a username of someone who contributed

---

[5] `https://github.com/alontalmor/LeapOfThought`

[6] `https://github.com/THU-KEG/KEPLER`

Dataset	Model	Acc	Paraphrase Cons ↑	Entailment Acc ↑	Contrapositive Acc ↑
FEVER	RoBERTa-base	78.29 (0.86)	-	-	-
LeapOfThought	RoBERTa-base	93.66 (0.87)	-	85.63 (1.08)	16.51 (2.71)
zsRE	BART-base	21.01 (0.64)	69.50 (1.09)	-	-
Wikidata5m	BART-base	10.21 (0.59)	25.84 (0.53)	-	-

Table 14: Model accuracy and belief metric results and for four datasets.

Dataset	Data Type	Input	Label(s)
zsRE	Main Input	What did Gifford Pinchot die of?	{Leukemia}
	Paraphrase	How did Gifford Pinchot die?	
	Main Input	Player Ali Kanaan plays for what team?	{Sporting Al Riyadi Beirut}
	Paraphrase	What team is Ali Kanaan associated with?	
Wikidata5m	Main Input	Margarita Nolasco Armas has relation 'place of birth' to	{Orizaba, Veracruz; Orizaba; etc.}
	Paraphrase	SusunW/Margarita Nolasco Armas has relation 'born at' to	
	Local Neutral	Margarita Nolasco Armas has relation 'place of death' to	Mexico City; Ciudad de Mexico; etc.
	Main Input	Mary Good has relation 'award received' to	{Garvan-Olin Medal; Arkansas Women's Hall of Fame; etc.}
	Paraphrase	Mary Lowe Good has relation 'winner of' to	
	Local Neutral	Mary Good has relation 'educated at' to	{The University of Arkansas; U Arkansas; etc.}
FEVER	Main Input	Tardigrades are also known as space bears.	True
	Main Input	The Lion belongs to the genus Vulpes.	False
LeapOfThought	Main Input	A viper is a vertebrate.	True
	Entailed Data	A viper has a brain.	True
	Main Input	A amaranth is a herb.	True
	Entailed Data	A amaranth has a nose.	False

Table 15: Example datapoint from each dataset, and auxiliary data that accompanies the Main Input.

to the Wikipedia article for Margarita Nolasco Armas. Meanwhile, other aliases for J.R.R Tolkien include "Tolkienian" and "Mabel Suffield," his mother. Rephrasings of relations might also create confusing inputs, e.g. switching "child" with "has kids," "daughter", or "son." Similar to zsRE, some relations are also one-to-many and temporally dependent (like occupation), though we hope that by using many valid answers we circumvent this issue to some extent when calculating prediction correctness.

D Metric Computation and Bootstrap Details

Metric computation. The only computationally difficult metric to calculate is Δ-Acc, which requires computing the updated language model's accuracy on other data after every single belief update. We randomly sample other data after every update for this purpose, using $n = 30$ points for zsRE and Wikidata5m and $n = 200$ points for FEVER and LeapOfThought. We ensure that all evaluation data is used at some point during this sampling by preferentially selecting data that has been infrequently selected before. We note that paraphrase consistency is easy to evaluate for a small number of paraphrases per datapoint, as we have for both zsRE and Wikidata5m. Additionally, on LeapOfThought, we compute Δ-Acc using both Main Inputs and Entailed Data.

Update-Transitivity caveat. The % Update-Transitivity metric represents the answer to the question: if updating belief A changes belief B, and updating belief B changes belief C, what proportion of the time does updating A change C? We would treat this as a normative metric that we hope to maximize, except we do not know in general whether there is a confounding belief D that determines the relationship between B and C. If changing A also changed a confounding belief D, then we might not be able to expect that C should change too. That said, when we have no reason to think there are such confounding beliefs, we would expect a logically consistent model to display 100% Update-Transitivity of their beliefs. In Fig. 2, for instance, we see no reason to suspect there are con-

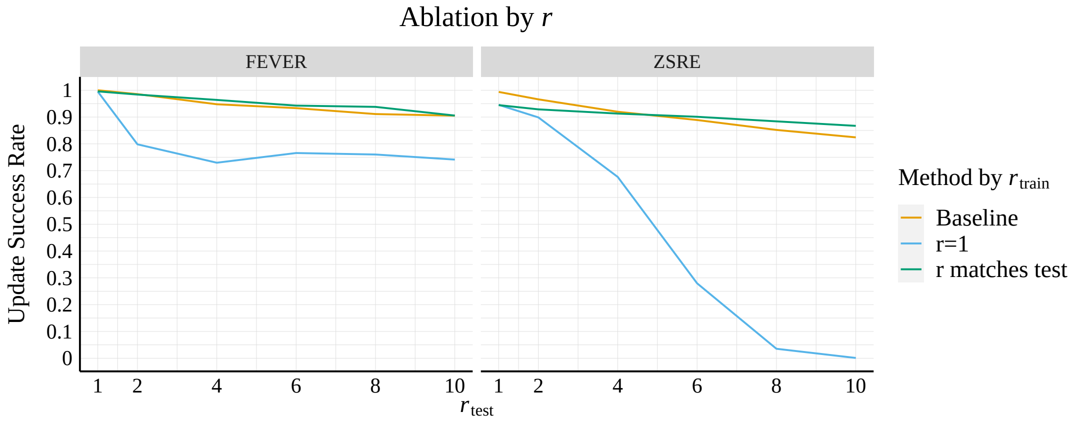


Figure 5: Ablation across values of r for training and testing. On zsRE, our method outperforms the baseline when $r_{\text{test}} = 10$, and the gap is likely to increase as r_{test} rises further. When using a non-sequential objective from past work, performance declines drastically as r_{test} rises.

founding beliefs for the relationship between the date Bessie Smith died and the writer of Despicable Me 2, and therefore we would expect that updating the belief about what album Hot Right Now is on would change the belief in Despicable Me 2's authorship (which it does).

Bootstrap computation. We account for sample and seed variance by block bootstrap (Efron and Tibshirani, 1994). When there is a single statistic per data point, like Main Input Update Success, we form a matrix of shape $n \times s$ for n data points and s model seeds (where the seed was used for both task model training and learned optimizer training). We then resample rows and columns of this matrix 10,000 times, which was sufficient for convergence. When we perform hypothesis tests for the difference in statistics between conditions, we pair the data points by using the same rows of this matrix at each step of the bootstrap (i.e. we conduct paired tests). For metrics involving multiple data points per Main Input, like paraphrases or other random data, we make a simplifying assumption where we do not resample the multiple data points but just compute the average metric for those data points and treat that as the ground-truth statistics for the Main Input. We explored using a full 3-dimensional bootstrap, where we resample among these extra datapoints by constructing a matrix of shape $n \times s \times n$, but it was quite slow and gave similar results to the block bootstrap.

E Additional Results

Ablation across num. sequential steps. Fig. 5 shows the results for an ablation across r_{test} using two kinds of learned optimizers: SLAG$_1$, where $r_{\text{train}} = 1$, and a SLAG condition where $r_{\text{train}} = r_{\text{test}}$. It is critical to the success of learned optimizers to train them to update points sequentially when this is a desired application. Further, sequential updating with sequence prediction tasks is the only setting where we see learned optimizers outperform baselines across all relevant metrics.

	Update Success Rate		Δ-Acc
Desired Label	Main Input	Paraphrases	All Data
Beam Label	91.19 (0.5)	92.07 (0.8)	-0.39 (0.1)
Hard Label	94.46 (0.7)	94.45 (0.7)	-0.24 (0.1)

Table 16: Update metrics by optimizer training labels.

Choosing training labels for learned optimizers. In early experiments, we found that it is beneficial to use all data points (including correctly predicted points) as Main Inputs *during training*, rather than restricting training to only incorrectly predicted points. We still focus on correcting wrong outputs at test time. But so we must select what label to use during optimizer training. To get a Hard Label, we use the correct label for incorrectly predicted points, and for correctly predicted points, we simply draw a label randomly from the labels in the training data. The alternative Beam Label condition uses a sample from the model's beam search for a data point, as done in past work (De Cao et al., 2021; Mitchell et al., 2021). We show update metrics for zsRE split by the desired label in Table 16. If one's goal is to fix wrong model outputs, then it is much better to use either the correct label or a random label as the desired model output during training rather than a sample from the model's beam search. Update success improves by

Objective Term Ablation		Update Success Rate			Retain Predictions		Δ Acc
Dataset	Objective	Main Input	Paraphrases	Entailed Data	Local Neutral	All Data	All Data
FEVER	Main	100 (0.0)	-	-	-	98.27 (0.1)	-0.15 (0.1)
	(no KL)	100 (0.0)	-	-	-	40.42 (0.6)	-27.19 (1.2)
LeapOfThought	Main	100 (0.0)	-	76.43 (5.3)	-	96.84 (0.3)	-1.22 (0.8)
	+Ent	100 (0.0)	-	71.87 (5.3)	-	96.52 (0.3)	-0.40 (0.8)
zsRE	Main	94.46 (0.4)	94.44 (0.7)	-	-	81.96 (0.4)	-0.24 (0.1)
	+Para	93.75 (0.4)	94.41 (0.7)	-	-	75.24 (0.5)	-0.42 (0.2)
Wikidata5m	Main	88.67 (0.7)	64.12 (0.7)	-	49.78 (1.0)	71.04 (0.5)	-1.54 (0.3)
	+Para	87.46 (0.7)	81.06 (0.7)	-	47.15 (1.0)	63.02 (0.6)	-1.55 (0.3)
	+LN	87.73 (0.7)	59.75 (0.7)	-	60.49 (1.0)	72.69 (0.6)	-1.57 (0.3)
	+Para+LN	87.02 (0.7)	81.18 (0.7)	-	56.86 (1.0)	68.42 (0.6)	-1.65 (0.3)

Table 17: Belief update results by the objective terms used for the learned optimizer. We do not bold any numbers based on statistical significance. For tuning purposes we select whichever condition achieves the higher selection criterion without testing for statistical significance.

3.27 (±0.65; p<1e−4) points for the Main Input and 2.38 (±1.05; p<1e−4) for Paraphrases, while Δ-Acc rises by 0.15 (±0.18; p=.09).

Which beliefs are hard to update? We hypothesize that beliefs will be easier to update when they are more belief-like to begin with. We principally measure this via the correlation between update success rate and a belief's consistency on paraphrases before the update, for our learned optimizer in a single-update setting ($r = 1$). Surprisingly, we observe no relationship between update success and the belief consistency. The correlation between consistency and update success is near 0 for both zsRE ($\rho = -.027$) and Wikidata5m ($\rho = .013$); see Fig. 6 for a plot of the relationship. So it appears that the learned optimizer can update model beliefs independently of how belief-like they are to begin with. We would also be interested in considering consistency under entailment, but the update success rate on LeapOfThought is already 100%, so there is no variance to explain.

Learning curve. In Fig. 7 we show the learning curve of a learned optimizer trained with SLAG on zsRE. The Main Input Update Success Rate steadily rises as a function of the training set size.‘

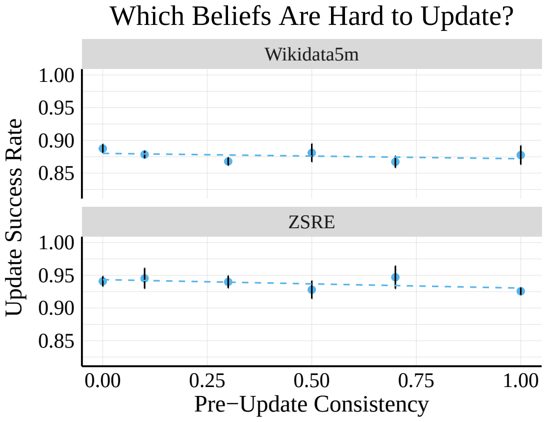


Figure 6: Beliefs are neither easier nor harder to update depending on their consistency beforehand.

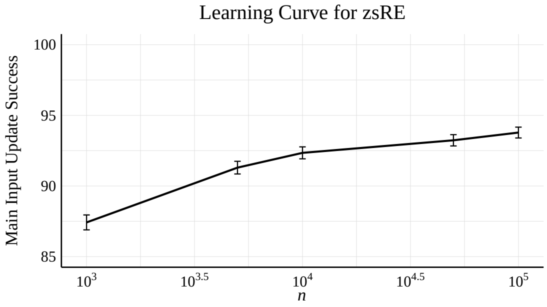


Figure 7: Main Input Update Success Rate across training set sizes, using SLAG on zsRE.

Ablation by objective term. We give objective ablation results in Table 17. Surprisingly, we do not always see that the objective terms help for the data they are intended to help with. First, we obtain mixed results for the paraphrase objective. On zsRE, the objective term seems to hinder performance, with update success dropping on Main Inputs by 0.71 (±0.60; p=.021) and Δ-Acc dropping by 0.18 (±0.19; p=.069), while the paraphrase Update Success Rate itself is unaffected. With Wikidata5m, however, the paraphrase term improves paraphrase update success by a large margin of 16.94 (±1.03; p<1e−4) points. Adding the Local Neutral (LN) term with the paraphrase term greatly improves the LN Retain Rate for Wikidata5m, by 9.71 points (±1.44; p<1e−4), though both of these terms come at a cost to Main Input Update Success, similar to zsRE. Lastly, we do not find that the entailment objective improves Entailed Data Update Success; in fact, this metric falls by 4.56 (±7.22;

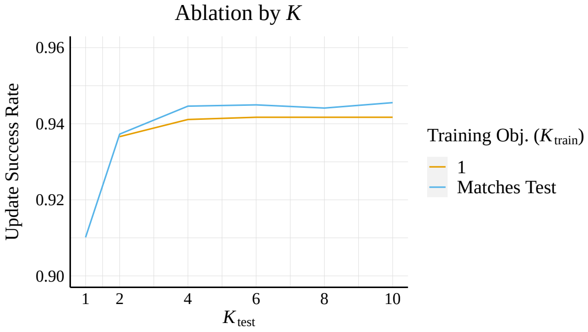


Figure 8: Ablation across values of K for training and testing, using SLAG on zsRE. It is useful to train the optimizer using the value of K it will use at test time.

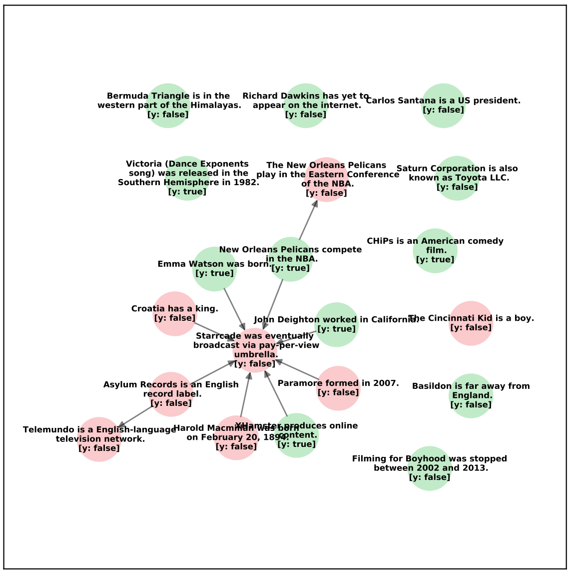


Figure 9: A random subgraph of the belief graph for FEVER. Note all nodes actually are connected to at least one another node.

p=.213) points with the objective.

Ablation by num. update steps. Fig. 8 shows the results of an ablation across values of K using a learned optimizer trained using SLAG with $r = 1$ on zsRE. Main Input Update Success rises by over three points by increasing K_{test} from 1 to at least 5. Using a value of K_{train} that matches K_{test} gives a further increase of about 0.5 points.

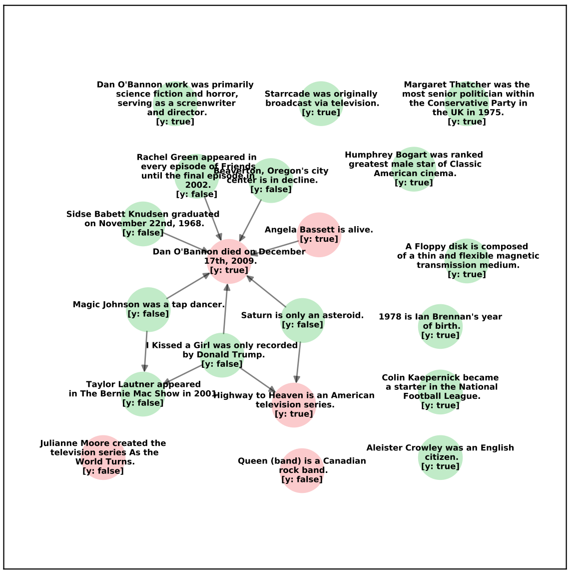


Figure 10: A random subgraph of the belief graph for FEVER. Note all nodes actually are connected to at least one another node.

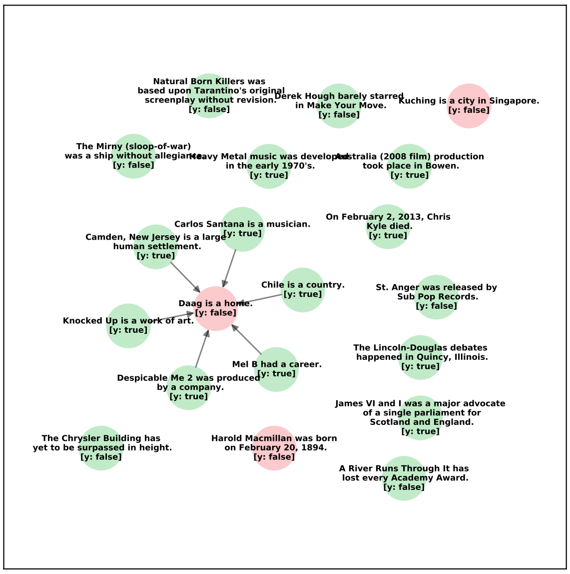


Figure 11: A random subgraph of the belief graph for FEVER. Note all nodes actually are connected to at least one another node.

Improving Sign Recognition with Phonology

Lee Kezar
University of Southern California
lkezar @usc.edu

Jesse Thomason
University of Southern California
jessetho@usc.edu

Zed Sevcikova Sehyr
San Diego State University
zsevcikova@sdsu.edu

Abstract

We use insights from research on American Sign Language (ASL) phonology to train models for isolated sign language recognition (ISLR), a step towards automatic sign language understanding. Our key insight is to explicitly recognize the role of phonology in sign production to achieve more accurate ISLR than existing work which does not consider sign language phonology. We train ISLR models that take in pose estimations of a signer producing a single sign to predict not only the sign but additionally its phonological characteristics, such as the handshape. These auxiliary predictions lead to a nearly 9% absolute gain in sign recognition accuracy on the WLASL benchmark, with consistent improvements in ISLR regardless of the underlying prediction model architecture. This work has the potential to accelerate linguistic research in the domain of signed languages and reduce communication barriers between deaf and hearing people.

1 Introduction

When learning to recognize sign language, there is evidence that people rely on breaking signs down into their constituent parts, such as the configuration and location of the hand (Klima and Bellugi, 1979). This process is also true of spoken language recognition, where recognizing sound patterns plays a crucial role in one's ability to recognize a word. Sometimes, one of these "parts" (phonemes) is the only distinguishing factor between two very different terms, as seen in the signs for DIFFERENCE (palms up) and BALANCE (palms down) in Croatian Sign Language (Kuhn et al., 2006). Thus, the ability to encode and recognize individual phonemes and the relationships among them is essential for sign recognition. As a first step in exploring the practicality of phoneme recognition, we ask: Can machine learning models for isolated sign language recognition (ISLR) benefit from the phonological structure of signs?

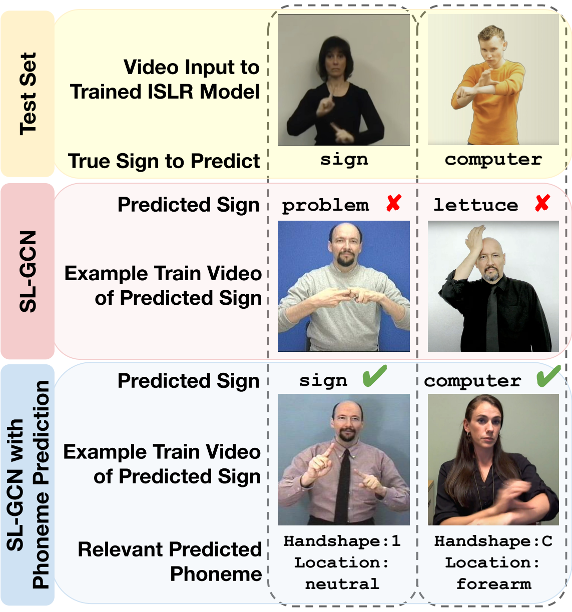


Figure 1: We demonstrate that sign language recognition models improve in accuracy when also tasked with predicting component phonemes of the sign.

While some ISLR models explicitly focus on the signers' hands (Hu et al., 2021) or face (Albanie et al., 2020), none have leveraged sign language phonology. Instead, ISLR has been treated similarly to gesture recognition, where a "gesture" (such as swinging an imaginary bat or waving a hand) has no underlying structure except for that of the human body itself. This lack of structure might explain why state-of-the-art models like the Sign Language Graph Convolution Network (SL-GCN, Jiang et al. 2021) sometimes predict labels that are visually and phonologically unrelated to the ground truth, as shown in Figure 1.

In contrast, we show that models trained to recognize both signs and their phonemes will be more accurate at sign identification than those trained for ISLR alone. Our main contributions are:

- We join an ISLR benchmark with a dataset of phonologically-labeled signs (§3.1) and de-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2732–2737
May 2-6, 2023 ©2023 Association for Computational Linguistics

scribe a simple method for learning these labels alongside the target gloss[1] (§3.2).

- We explore which and how many phoneme types are most beneficial as an auxiliary task to sign recognition (§3.3, §4.1).
- We demonstrate that adding auxiliary predictions for sign language phonology targets yields nearly 9% absolute gain in accuracy for ISLR sign prediction (§4.2), and that the resulting phoneme classification heads outperform prior work (§4.3).

2 Background

Sign languages are complete and natural languages primarily used by deaf and hard-of-hearing people. There are hundreds of sign languages in the world today collectively used by tens of millions of people (Eberhard et al., 2022). They rely on the hands, face, and body to communicate meaning according to complex grammars which are independent of any spoken language.

Sign languages have been and continue to be largely overlooked in natural language processing (NLP) research, necessitating explicit calls for more inclusivity (e.g. Yin et al. 2021, Bragg et al. 2019). In this paper, we seek to bridge robust techniques in NLP with insights from theories of sign language phonology.

Sign language phonology is an abstract system of rules that governs how the structural units of signs (e.g., handshape, location, movement) are combined to create an infinite number of utterances. These manual units play a significant role at the phonological level similarly to place of articulation, manner, and voicing in spoken language. Theories of sign language phonology attempt to enumerate the meaningless units or "phonemes" found in a sign language and describe the complex relationships among them. In ASL-LEX 2.0, Sehyr et al. (2021) describe 16 types of phonemes, largely guided by Brentari's Prosodic Model (Brentari, 1998). We provide three examples of these phoneme types here:

- **Minor Location**: one of 37 regions of the body where the sign is produced (e.g. "chin").
- **Handshape**: one of 49 configurations of the hand (e.g. "2").
- **Path Movement**: one of 8 ways of moving the hand through space during the production of a single sign (e.g. "circular").

Brentari's Prosodic Model contains < 200 possible phonemes across its 16 phoneme types, each of which can be observed during the production of any sign. In ASL-LEX 2.0, about 70% of signs can be uniquely identified by their phonemes, making them an appealing conduit for learning to recognize signs. We leverage these properties by using them as target labels alongside the target gloss.

ISLR: Definition and Prior Work In ISLR, a model is given a video of one sign being produced in isolation and must predict the target gloss S_{gloss}. Many models have been proposed to recognize isolated signs, varying with regard to input modality (e.g. pose, RGB video), pretraining (e.g. frame prediction, hand modeling), and encoding strategy (e.g. attention, convolution). Selvaraj et al. (2022) provide a comprehensive framework for comparing models across multilingual data, in particular LSTMs (Konstantinidis et al., 2018), Transformers (Devlin et al., 2019), Spatio-Temporal Graph Convolution Network (ST-GCN, Cheng et al. 2020), and Sign Language Graph Convolutional Network (SL-GCN, Jiang et al. 2021).

We evaluate our method with SL-GCN and via a Transformer network. These models are open-sourced,[2] easily modifiable, and take in pose information as input. These models perform well on the WLASL 2000 benchmark (Li et al., 2020). While the model from (Hu et al., 2021) obtains higher accuracy on that benchmark, their code is not publicly available to replicate those findings.

3 Method

We combine two datasets for the task of ISLR, ASL-LEX 2.0 (Sehyr et al., 2021) and WLASL 2000 (Li et al., 2020), in order to learn ASL phonology (§3.1). Then, we describe how to utilize these data by learning two ISLR models to predict both the target gloss and the phonemes for any input (§3.2). Finally, we address the questions of how many and which phoneme types are best for ISLR (§3.3). The dataset and modified models are released for replication and future work.[3]

---

[1] A "gloss" is a label for a sign that corresponds to its translation in the target language, such as APPLE.

[2] `https://openhands.readthedocs.io/`

[3] `https://github.com/leekezar/ImprovingSignRecognitionWithPhonology`

3.1 Data

We combine the phonological annotations in ASL-LEX 2.0 (Sehyr et al., 2021) with signs in the WLASL 2000 ISLR benchmark (Li et al., 2020). ASL-LEX contains 2,723 videos, each demonstrating a unique sign and human-annotated with phonemes across 16 categories. WLASL contains 21,083 videos, each demonstrating one of 2,000 unique signs (an average of 10.5 videos per sign).

To combine these datasets, we edit the WLASL metadata file to add 16 new properties (one for each phoneme type) to each video example. If the video's English gloss is also found in ASL-LEX, then we copy the phonemes directly from ASL-LEX. If it is not found, then we set these new properties to `-1` and ignore them during training. After combining, 48% of videos in the aggregated dataset have phonological labels, and all of the videos retain their original split (train, validation, and test) and English gloss. Note that this dataset is identical in structure to WLASL-LEX (Tavella et al., 2022), however, both our sources are more recently updated and contain more samples. Table 1 provides a summary of the combined data.

$\mathcal{P}$ Labels	# Signs	# Videos Train	 Val	 Test	 Total
✗	1246	7850	2221	1574	11645
✓	754	6439	1695	1304	9438
Total	2000	14289	3916	2878	21083

Table 1: We match phonological data from ASL-LEX 2.0 with signs in the WLASL benchmark to create a subset of WLASL with phoneme type labels $\mathcal{P}$.

3.2 Models

We add phoneme value predictions to two ISLR model architectures: a graph convolutional network, SL-GCN (Jiang et al., 2021), and a Transformer-based model. These models are implemented by the OpenHands project (Selvaraj et al., 2022) and are largely left untouched; we refer the reader to the OpenHands paper and code for implementation details. The SL-GCN model treats pose estimations over time as a connected graph and learns 10 convolution layers over this graph, using spatial and temporal attention. The Transformer model treats pose estimations as a sequence of coordinates over time and learns 5 Transformer layers similarly to BERT (Devlin et al., 2019). Importantly, both of these models implement spatial and temporal attention, a feature which enables phoneme recognition even when the phoneme exists for a short amount of time.

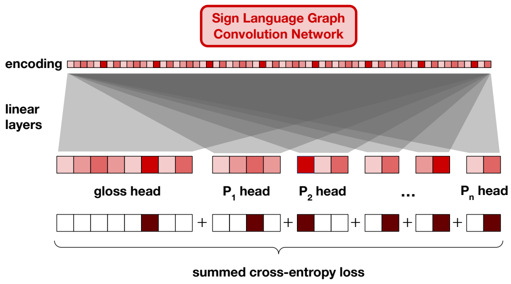


Figure 2: The proposed decoder relies on fully-connected layers for each classification head: one for the target gloss and one for each of the n phoneme types.

For each model, we modify the decoder to classify not only the target gloss, but also the selected phonemes. This is accomplished by adding n fully-connected layers to the decoder, each with shape `(hidden size, # phoneme values)`. During the forward pass, the video encoding is used as the input for each fully connected layer (not chained together). The total loss is then computed as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{gloss}} + \sum_{\text{phoneme} \in \mathcal{P}} \mathcal{L}_{\text{phoneme}},$$

where $\mathcal{L}_{\text{gloss}}$ is the cross entropy of the model's gloss predictions, while $\mathcal{L}_{\text{phoneme}}$ is the cross entropy of the model's phoneme predictions. The sum of these losses is then backpropagated to the entire model, encouraging the encoder to learn a representation which more explicitly captures the desired phonemes $\mathcal{P}$ alongside the target gloss (Fig 2).

We train models until the validation accuracy has not improved in the last 30 epochs and use the top performing model for testing. For further details on model implementation and training procedure, see Selvaraj et al. (2022).

3.3 Phoneme Type Selection

It is not immediately clear which, if any, of the 16 phoneme types in ASL-LEX 2.0 yield improvements on ISLR. Regarding *how many* phoneme types, one might assume that more informative outputs would only improve a model's ability to recognize signs, and therefore all 16 phoneme types should be included. However, these additions come at a cost to the encoder, which must now learn to fit more information into the same encoding space

Model	Pred	$\text{WLASL}^{\text{test}}_{\text{all}}$			$\text{WLASL}^{\text{test}}_{\text{w}\mathcal{P}}$			$\text{WLASL}^{\text{test}}_{\text{w/o}\mathcal{P}}$		
	$\mathcal{P}$	%A@1	%A@3	MRR	%A@1	%A@3	MRR	%A@1	%A@3	MRR
SL-GCN	✗	29.4±1.6	50.2±2.3	.43±.02	35.0±1.8	56.1±1.7	.48±.02	24.8±1.4	45.3±3.0	.39±.02
	✓	38.1±0.5	61.0±0.3	.52±.00	44.1±1.1	64.1±0.6	.56±.01	33.1±0.3	58.4±0.2	.49±.00
Δ *Improvement*		*8.7	*10.8	*.09	9.1	8.1	.08	8.3	13.1	.10
Transformer	✗	20.5±0.4	36.9±1.0	.32±.01	24.5±1.1	41.2±1.7	.36±.01	17.2±0.3	33.3±0.7	.29±.00
	✓	23.4±0.4	41.7±0.7	.36±.01	28.2±0.4	46.5±0.6	.40±.01	19.3±1.0	37.8±1.6	.32±.01
Δ *Improvement*		*2.8	*4.8	*.04	3.7	5.3	.04	2.1	4.5	.03

Table 2: ISLR model performance with and without training with auxiliary phoneme predictions averaged over four seeds. Models are trained on $\text{WLASL}^{\text{train}}_{\text{all}}$ and evaluated on $\text{WLASL}^{\text{test}}_{\text{all}}$. Models trained to predict phonemes improve over their ISLR-only baselines on *both* signs seen at training time with phonemes ($\text{WLASL}^{\text{test}}_{\text{w}\mathcal{P}}$) and signs for which no phonological data was available during training ($\text{WLASL}^{\text{test}}_{\text{w/o}\mathcal{P}}$). Differences on $\text{WLASL}^{\text{test}}_{\text{all}}$ are significant (*) at $p < 0.05$ under a Welch's two-sided t-test with a Bonferroni correction applied.

without adding new samples. Furthermore, it is unclear *which* types to maximize performance.

To address these questions, we define the utility U of a set of phonemes types $\mathcal{P}$ as the percentage of signs that are uniquely identified by those types. Defined in this way, a set of phoneme types with high utility ensures that when a model can accurately predict those types, it is guaranteed to have sufficient information to recognize $U(\mathcal{P})$ percent of signs. $U(\mathcal{P})$ is provided by:

$$U(\mathcal{P}) = \frac{\sum_{S,S' \in V} \mathbf{1}\left[p(S|\mathcal{P}) > p(S'|\mathcal{P})\right]}{|V| - 1},$$

where V is the set of all target glosses and $P(S|\mathcal{P})$ is the probability of a sign S given the observed phoneme values in $\mathcal{P}$. We implement $P(S|\mathcal{P})$ with a simple look-up table for all possible combinations of the 16 phoneme types. With this utility function in hand, we can define the optimal subset of n phoneme types as:

$$\mathcal{P}^*(n) = \left\{ \arg\max_{\mathcal{P}} U(\mathcal{P}) : |\mathcal{P}| = n \right\}.$$

4 Results

We demonstrate across-the-board improvements on ISLR when predicting phonemes alongside glosses. We measure model performance via ISLR accuracy, both top-1 and top-3, as well as mean reciprocal rank (MRR), which ranges from 1 (correct sign given highest prediction score) to 1/2000 (correct sign given lowest prediction score).

4.1 Not All Phonemes are Helpful.

First, we explore which subsets of the 16 labeled phonemes are most beneficial for downstream ISLR. We train models with auxiliary losses for $\mathcal{P}^*(n), n \in \{2, 5, 9, 16\}$ and report their performance in Table 3. With two classification heads—handshape and minor location—we most improve ISLR on $\text{WLASL}^{\text{val}}_{\text{w}\mathcal{P}}$.

4.2 Predicting Phonemes Improves ISLR.

Table 2 demonstrates that adding classification heads for handshape and minor location yield a 3–9% gain on top-1 accuracy, 5–11% gain on top-3 accuracy, and .04–.09 gain on MRR. These gains are greater for signs trained with phonological labels, but extend to signs that do not have phonological labels as well!

4.3 SL-GCN Performs Accurate Phoneme Classification.

To lay the groundwork for modeling phonology in and of itself, we train SL-GCN to predict all 16 phoneme types and examine its accuracy at phoneme prediction (Figure 3). We compare to a frozen SL-GCN encoder pretrained for *only* ISLR, on top of which we learn linear probes for each phoneme type, as well as a majority class baseline. In all cases, training SL-GCN explicitly for phoneme prediction leads to the highest phoneme prediction accuracy. Prior work predicted phoneme values for Flexion, Major Location, Minor Location, Path Movement, Selected Fingers, and Sign Type (Tavella et al., 2022). Despite not being the explicit goal of this work, SL-GCN with auxiliary phoneme prediction outperforms that model, too.

5 Discussion

We find that adding auxiliary classification tasks for sign phonemes to ISLR models statistically significantly improves sign recognition accuracy. Repre-

n	$\mathcal{P}^*(n)$	%A@1	%A@3	MRR
0	$\emptyset$	50.2	69.3	.62
2	Dominant Handshape, Minor Location	55.7	74.7	.67
5	$\mathcal{P}^*(2)$ + Nondominant HS, Path Movement, Repeated Movement	51.5	69.3	.62
9	$\mathcal{P}^*(5)$ + 2nd Minor Loc., 2nd Handshape, Wrist Twist, Contact	52.5	69.3	.64
16	$\mathcal{P}^*(9)$ + Remaining 7 phoneme types	54.3	72.7	.65

Table 3: SL-GCN sign recognition accuracy when trained on WLASL$^{\text{train}}_{\text{w}\mathcal{P}}$ with auxiliary predictions of the top-n phoneme types $\mathcal{P}$ and tested on WLASL$^{\text{val}}_{\text{w}\mathcal{P}}$. See §3.3 for the details of $\mathcal{P}^*(n)$.

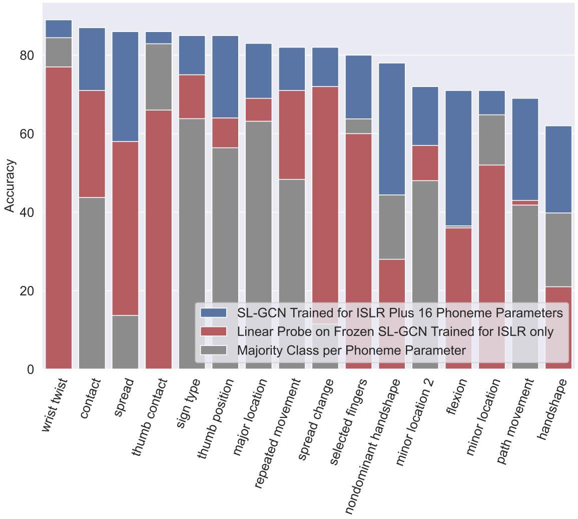


Figure 3: Model phoneme prediction accuracy when trained on WLASL$^{\text{train}}_{\text{w}\mathcal{P}}$ and tested on WLASL$^{\text{test}}_{\text{w}\mathcal{P}}$.

senting phonemes during training may enable models to learn a more holistic latent representation of sign videos compared to models that only predict the target gloss. The success of this approach provides evidence that handshape and minor location are not only useful in recognizing signs, but also easy enough to learn with semi-supervision (recall that only 48% of the dataset has handshape and minor location labels). Our findings show that both models learn these new labels well (Fig 3) and as a result, the encodings for *all* videos contain more relevant information for ISLR.

A secondary finding of this paper is that SL-GCN, when trained to recognize all 16 phoneme types, outperforms prior work by anywhere from 1%-9%. Still, there is room for improvement in phoneme recognition, especially for handshape, minor location, and path movement.

6 Limitations

The WLASL benchmark has several notable limitations that must be taken into account by those interested in using it. Dafnis et al. (2022) show that incorrect labels are pervasive in WLASL, causing lower ISLR accuracy and, in this work, incorrect phoneme labels. Additionally, existing sign language datasets do not provide information about the signers' fluency, dialect, age, or race and therefore may not be representative of those who use ASL. Finally, we caution those interested in collecting ASL data against scraping websites without permission, and we encourage acknowledging the creators of those sources.

As a first attempt to model sign language phonology in order to improve sign recognition, we applied our approach to two models and used data for one language pair (ASL/English). Although many phonemes are shared across signed languages, more language pairs and models should be tested in order to verify our claim that learning phonology improves sign recognition *in general*. In particular, the Two-Stream Inflated 3D ConvNet (I3D; Carreira and Zisserman 2017) model, designed for gesture recognition, has also been shown to do well on ISLR (e.g. Hosain et al. 2021, Albanie et al. 2020) and we look forward to extending our method to this model as well.

References

Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Joon Son Chung, Neil Fox, and Andrew Zisserman. 2020. BSL-1K: Scaling up co-articulated sign language recognition using mouthing cues. In *European Conference on Computer Vision*.

Danielle Bragg, Oscar Koller, Mary Bellard, Larwan Berke, Patrick Boudreault, Annelies Braffort, Naomi K. Caselli, Matt Huenerfauth, Hernisa Kacorri, Tessa Verhoef, Christian Vogler, and Meredith Ringel Morris. 2019. Sign language recognition, generation, and translation: An interdisciplinary perspective. *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*.

Diane Brentari. 1998. *A Prosodic Model of Sign Language Phonology*. The MIT Press.

João Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ke Cheng, Yifan Zhang, Congqi Cao, Lei Shi, Jian Cheng, and Hanqing Lu. 2020. Decoupling gcn with dropgraph module for skeleton-based action recognition. In *European Conference on Computer Vision (ECCV)*.

Konstantinos M. Dafnis, Evgenia Chroni, Carol Neidle, and Dimitri Metaxas. 2022. Bidirectional skeleton-based isolated sign recognition using graph convolutional networks. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7328–7338. European Language Resources Association.

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

David M. Eberhard, Gary F. Simons, and Charles D. Fennig. 2022. Ethnologue: Languages of the world.

Al Amin Hosain, Panneer Selvam Santhalingam, Parth H. Pathak, Huzefa Rangwala, and Jana Kosecka. 2021. Hand pose guided 3d pooling for word-level sign language recognition. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*.

Hezhen Hu, Wen gang Zhou, and Houqiang Li. 2021. Hand-model-aware sign language recognition. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Raymond Fu. 2021. Skeleton aware multi-modal sign language recognition. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

E.S. Klima and U. Bellugi. 1979. *The Signs of Language*. Harvard University Press.

Dimitrios Konstantinidis, Kosmas Dimitropoulos, and Petros Daras. 2018. Sign language recognition based on hand and body skeletal data. *3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*.

Ninoslava Kuhn, Tamara Ciciliani, and Ronnie Wilbur. 2006. Phonological parameters in croatian sign language. *Sign Language & Linguistics*.

Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. 2020. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*.

Zed Sevcikova Sehyr, Naomi K. Caselli, Ariel Cohen-Goldberg, and Karen Emmorey. 2021. The ASL-LEX 2.0 Project: A Database of Lexical and Phonological Properties for 2,723 Signs in American Sign Language. *The Journal of Deaf Studies and Deaf Education*.

Prem Selvaraj, C. GokulN., Pratyush Kumar, and Mitesh M. Khapra. 2022. Openhands: Making sign language recognition accessible with pose-based pretrained models across languages. In *Association for Computational Linguistics (ACL)*.

Federico Tavella, Viktor Schlegel, Marta Romeo, Aphrodite Galata, and Angelo Cangelosi. 2022. WLASL-LEX: a dataset for recognising phonological properties in American Sign Language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Kayo Yin, Amit Moryossef, Julie A. Hochgesang, Yoav Goldberg, and Malihe Alikhani. 2021. Including signed languages in natural language processing. In *Association for Computational Linguistics (ACL)*.

Parameter-efficient Modularised Bias Mitigation via AdapterFusion

Deepak Kumar[1], **Oleg Lesota**[1], **George Zerveas**[3], **Daniel Cohen**[3,4],
Carsten Eickhoff[3,5], **Markus Schedl**[1,2], **Navid Rekabsaz**[1,2]
[1]Johannes Kepler University Linz, Austria
[2]Linz Institute of Technology, AI Lab
`{deepak.kumar,oleg.lesota,markus.schedl,navid.rekabsaz}@jku.at`
[3]AI Lab, Brown University, USA `george_zerveas@brown.edu`
[4]Dataminr, USA `daniel.cohen@dataminr.com`
[5]University of Tübingen, Germany `c.eickhoff@acm.org`

Abstract

Large pre-trained language models contain societal biases and carry along these biases to downstream tasks. Current in-processing bias mitigation approaches (like adversarial training) impose debiasing by updating a model's parameters, effectively transferring the model to a new, irreversible *debiased* state. In this work, we propose a novel approach to develop stand-alone debiasing functionalities separate from the model, which can be integrated into the model on-demand, while keeping the core model untouched. Drawing from the concept of AdapterFusion in multi-task learning, we introduce DAM *(Debiasing with Adapter Modules)* – a debiasing approach to first encapsulate arbitrary bias mitigation functionalities into separate adapters, and then add them to the model on-demand in order to deliver fairness qualities. We conduct a large set of experiments on three classification tasks with gender, race, and age as protected attributes. Our results show that DAM improves or maintains the effectiveness of bias mitigation, avoids catastrophic forgetting in a multi-attribute scenario, and maintains on-par task performance, while granting parameter-efficiency and easy switching between the original and debiased models.

1 Introduction

Large pre-trained language models (PLM) and their variations fine-tuned on downstream tasks encode societal biases and stereotypes. A large body of research shows the existence of these biases (Zhao et al., 2019; Sheng et al., 2019; Rekabsaz et al., 2021b), and discuss their potential harms (Blodgett et al., 2020) in various tasks such as occupation prediction from biographies (De-Arteaga et al., 2019), information retrieval (IR) (Rekabsaz et al., 2021a; Rekabsaz and Schedl, 2020), and machine translation (Stanovsky et al., 2019). A common approach to mitigating these biases is to adjust model parameters according to specific fairness criteria, achieved using optimization methods such as adversarial training (Elazar and Goldberg, 2018; Barrett et al., 2019; Han et al., 2021b; Rekabsaz et al., 2021a).

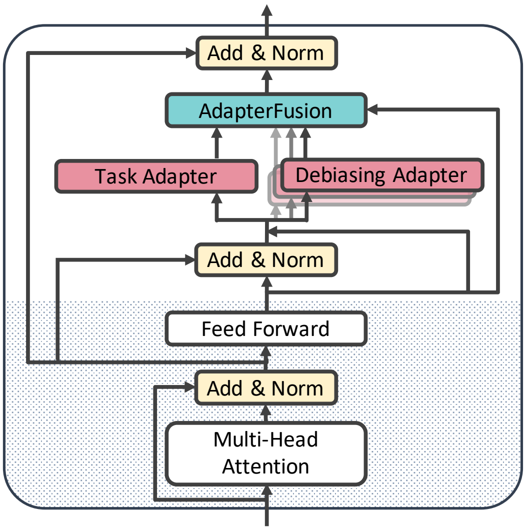


Figure 1: A transformer block in DAM to debias arbitrarily many protected attributes in different modules.

The resulting debiased models significantly diminish the effect of protected attributes (e. g. gender, race, etc.) on model's predictions. However, depending on the usage context, characteristics of the input or fairness-utility trade-off considerations, system designers or end-users might in practice still prefer to instead use the original (potentially biased) model for processing some requests. As an example, while a fairness-aware document retrieval model should tease out gender attributes when processing bias-sensitive queries (Krieg et al., 2022) (like *how to become CEO?*), other common queries (like *earliest pregnancy symptoms*) may specifically require gender information to obtain satisfactory retrieval results. Previous studies also show that one may need to occasionally accept slight performance degradation in exchange for realizing higher fairness (Zerveas et al., 2022; Biega et al.,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2738–2751
May 2-6, 2023

2018; Rekabsaz et al., 2021a). We argue that on-demand control over the strength of bias mitigation is crucial for the broad adoption of bias mitigation models. Using existing approaches, this would require maintaining and deploying multiple large parallel models for every protected attribute, in practice resulting in overly complex and resource-heavy pipelines and increased latency.

To reduce this untenable burden, we introduce *Debiasing with Adapter Modules (*DAM*)*. In our approach, bias mitigation functionalities can be trained in fully decoupled modules from the building blocks solving the main task (see Figure 1). These debiasing modules are learned independently, and can be selectively inserted into the model to deliver debiasing qualities. Our method draws from the recent success of Adapter (Rebuffi et al., 2017; Houlsby et al., 2019) and AdapterFusion (Pfeiffer et al., 2021) networks, originally introduced in the context of multi-task learning to enable parameter-efficient training and to avoid catastrophic forgetting. Specifically in multi-task learning with AdapterFusion, each task is encoded in specific parameter-efficient adapter networks attached to the transformer blocks (Vaswani et al., 2017) of a PLM, and the knowledge across the tasks is shared through a FusionAdapter layer with an attention mechanism. DAM extends this principle idea to bias mitigation by viewing the objective of "removing each protected attribute" as a standalone task, learned by an independent debiasing adapter module. These debiasing adapters are then integrated into the core model via a fusion layer to mitigate biases. This modular architecture of DAM enables on-demand debiasing, where by simply removing the debiasing adapters the model would return to its original state. The modularity and parameter efficiency of DAM also enables creating and sharing stand-alone debiasing solutions (in form of debiasing adapters), which can be applied to the underlying models on-demand. So, if one uses task adapter's output in DAM, they get biased embedding else they can take output of fusion layer for debiased embedding.

We evaluate DAM in extensive experiments covering various bias mitigation applications in three classification tasks: We study occupation prediction from biographies (De-Arteaga et al., 2019), mention prediction (Pardo et al., 2016), and hate speech detection (Founta et al., 2018), involving the protected attributes of gender, age, and dialect-based race. Among these, the mention prediction task provides two protected attributes (gender, age), enabling the study of adapter-based bias mitigation on multiple factors. Our evaluation results show that DAM (1) consistently improves bias mitigation in terms of lower leakage of protected attribute in comparison with fully fine-tuned models, and (2) better avoids catastrophic forgetting (Parisi et al., 2019; Chen et al., 2020) in a multi-attribute debiasing scenario, while (3) maintaining on-par task performance. Our results show the benefits of DAM in supporting fairness, and reveal the challenges in the scenarios inherently bounded by trade-offs.

Our contribution is three-fold:

- Introducing DAM, a novel, parameter-efficient approach for on-demand and modularized bias mitigation.
- Conducting a large set of experiments on three datasets and three classification tasks, showing the on-par or better performance of DAM in comparison with strong baselines.
- Examining the confounding cross-attribute effects in a multi-attribute dataset and demonstrating the benefits of our modularized bias mitigation approach.

The remainder of the paper is organized as follows: Section 2 reviews the relevant literature in adapter networks and bias mitigation domains. In Section 3, we explain the architecture and training strategy of DAM. Section 4 describes the setup of the experiments, whose results are reported and discussed in Section 5. Finally, Sections 6 and 7, respectively, summarize and discuss limitations of our work. Our code and trained resources are available in **`https://github.com/CPJKU/ModulaizedDebiasing`**.

2 Related Work

2.1 Adapter Networks

Adapter networks have been introduced in the context of multi-task learning (Rebuffi et al., 2017). Houlsby et al. (2019), and later Stickland and Murray (2019) attach the adapter module to the transformer blocks of a PLM, and learn a task by only updating the adapter parameters, while keeping the PLM's parameters unchanged. These studies show that the highly parameter-efficient adapter approach in general performs on par with fine-tuning all parameters of a BERT model (Devlin et al., 2019) for many tasks.

Other studies investigate various characteristics of adapter-based models such as the parameter efficiency, architectural variations, and transfer learning across tasks. Rücklé et al. (2021) show the training efficiency of adapter models in comparison with full model finetuning ($\sim 60\%$). Han et al. (2021a) examine the robustness to initialization seeds and training stability of the adapter approach. Mahabadi et al. (2021b) propose more parameter-efficient models by sharing adapter parameters across layers, followed by Mahabadi et al. (2021a), who introduce even more compact adapter modules. Poth et al. (2021) investigate the potential knowledge transfer across tasks. Recently, Pfeiffer et al. (2021) introduce AdapterFusion, where a fusion layer is defined on top of pre-trained adapter modules, and learns to combine information of adapters via an attention mechanism. This approach, avoiding the common pitfalls of catastrophic forgetting (Parisi et al., 2019), enables an effective knowledge transfer across tasks. Our work contributes to this direction by extending the concept of AdapterFusion to the task of bias mitigation via supervised interaction between the downstream task and the protected attribute.

2.2 Fairness & Bias Mitigation in NLP

The existence of biases and stereotypes in PLMs has been reported and discussed in several studies (Nadeem et al., 2021; Bhardwaj et al., 2021; Liang et al., 2021; Kirk et al., 2021; Vig et al., 2020). PLMs may even exacerbate these biases in downstream tasks as shown e. g. in the context of IR (Rekabsaz and Schedl, 2020). To reduce biases, *in-processing* methods – the focus of this work – aim at reducing the (cor)relation between the model's internal representations and the protected attributes (Ganhör et al., 2022). Debiasing PLMs has been approached for instance by linearly projecting embeddings to a new space that removes correlations to protected attributes (Kaneko and Bollegala, 2021; Bolukbasi et al., 2016). In a similar spirit, Guo et al. (2022) introduce a distribution alignment loss to force the model's outputs to become independent of the protected attribute. Schick et al. (2021) recently show that the encoded information in models can be exploited to spot the biases and hence to penalize them.

Adversarial training is a commonly used method to learn representations invariant to a specific factor of variation. Xie et al. (2017) and later Madras et al. (2018) introduce adversarial learning to the context of fair representation learning, where an adversary network learns to predict the protected attribute, and exploits the gradient of this prediction to remove the protected information using a gradient reversal layer (Ganin and Lempitsky, 2015). Several works further investigate the use of adversarial training for removing demographic information from neural/PLM-based text classifiers (Elazar and Goldberg, 2018; Barrett et al., 2019; Wang et al., 2021). Notably, Han et al. (2021b) show the benefit of having an ensemble of orthogonal adversaries. Beyond classification, Rekabsaz et al. (2021a) show that by applying adversarial training in the context of IR, one can achieve a more balanced retrieval of documents with respect to the presence of protected attributes in their contents.

As alternatives to adversarial debiasing, other works approach bias mitigation by minimizing the approximated upper bound of mutual information between task and protected attribute (Cheng et al., 2020b; Colombo et al., 2021), contrastive learning for fair representation disentanglement (Cheng et al., 2020a; Zhang et al., 2021), and introducing list-wise fairness regularization (Zerveas et al., 2022). While the mentioned methods are applied to whole a model, Iterative Nullspace Projection (INLP) (Ravfogel et al., 2020) achieves debiased representations by finding a linear mapping applied to output embeddings. The linear mapping of the INLP method and similarly the one of Ravfogel et al. (2022) offer effective bias mitigation methods particularly designed for the models with a linear decoder, but are not necessarily suited for the cases with non-linear decoders (e.g., a language generation decoder, or non-linear attackers).

Regarding adapter-based debiasing and closely related to our work, Lauscher et al. (2021) recently utilize adapters for debiasing PLMs by training an adapter which removes a protected attribute using counterfactual augmented data. When applied to a downstream task, another adapter is added on top of the debiasing adapter, and trained according to the task's objective. While shown effective in practice, in this stacking architecture the adapters in the higher levels inherently depend on the ones in the lower levels. They can not be learned standalone. In contrast, the fusion-based approach of DAM enables control by learning modular and independent debiasing adapters, which supports flexible plug-in/plug-out on demand.

3 Bias Mitigation with DAM

We approach the scenario in which the aim is to maximize a task's core objective (such as occupation prediction or hate speech prediction), while reducing the effects of the biases caused by k protected attributes. The crux of the proposed DAM approach is to first learn a task adapter and k bias removal adapters all independently, and then combine them through a fusion module. Once a model has been trained with DAM, one can easily switch between the original model by only using the task adapter, or alternatively impose debiasing by "plugging in" the learned debiasing adapters as well as the fusion module. In what follows, we explain the architecture of DAM followed by the optimization procedure.

3.1 Model Architecture

DAM consists of three main parts: task adapter, k debiasing adapters, and fusion. Following Pfeiffer et al. (2021), and as shown in Figure 1, these parts extend the architecture of a transformer block by being added to its last layer.

Adapters Each adapter is defined as a multilayer perceptron (one hidden layer and $\tanh(x)$ in our experiments), where the hidden layer typically has the same or a smaller dimension than the input. In DAM, the task adapter and the k debiasing adapters receive the output vector of the preceding transformer components, denoted as $\boldsymbol{u}$, and apply the corresponding transformations to output the vectors $\boldsymbol{v}_t$, and $\boldsymbol{v}_{b_1}, ..., \boldsymbol{v}_{b_k}$, respectively.

Fusion The fusion module combines the outputs of the task and debiasing adapters through the attention mechanism to produce the final output vector. This module receives $[\boldsymbol{v}_t, \boldsymbol{v}_{b_1}, ..., \boldsymbol{v}_{b_k}]$ as keys and values, and the $\boldsymbol{u}$ vector as the query of a single-head multiplicative attention network, and calculates the output as the weighted sum of the value vectors. These weights are calculated as attention scores and form a probability distribution over value vectors. Essentially, the fusion module learns to perform a *linear combination* of the embedding containing information for the task with the embeddings of the debiasing adapters, to provide the final, debiased embedding.

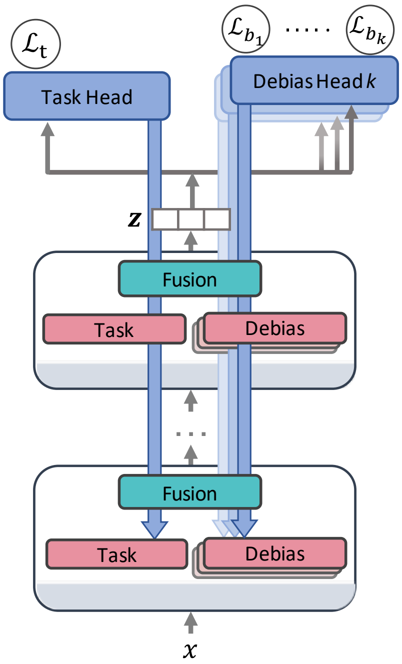


Figure 2: Schematic view of applying DAM to adversarial bias removal.

3.2 Learning On-demand Bias Mitigation

DAM introduces a generic approach to bias mitigation and representation disentanglement, and can, in principle, be integrated with any bias removal optimization method, provided that the method defines separate losses for the task and each of the protected attributes. Irrespective of the optimization method, the training procedure of DAM is the following: first, the task adapter is trained according to the task's objective. Then, one debiasing adapter for each protected attribute is trained using only its own debiasing objective (see next paragraph), without involving the task loss. Finally, all adapters' parameters are kept frozen and the fusion layer is trained using a combination of the task objective and debiasing objectives. Throughout training, the parameters of the underlying PLM remain frozen. In what follows, we describe using DAM together with the adversarial bias mitigation method as depicted in Figure 2.

Adversarial Training Adversarial bias mitigation aims to make the output embedding of a model invariant to the protected attribute, by removing information that allows predicting protected attributes from the latent representation. In this sense, adversarial training follows a *fairness through blindness* (Barocas et al., 2019) approach, where the system is made agnostic of the variations in underlying protected attributes.

To apply adversarial debiasing with DAM (see Figure 2) in the context of text classification, the model first encodes the input sequence x in the latent vector $\boldsymbol{z}$, based on which the corresponding class is predicted using a task classification head. The loss of this prediction, denoted as $\mathcal{L}_t$, is defined as cross entropy, and its gradient is used to update the parameters of the task adapter. Similarly, a dedicated classification head is defined for each protected attribute b_i, which also receives $\boldsymbol{z}$ as input, predicts the corresponding protected attribute, and calculates the cross entropy loss function $\mathcal{L}_{b_i}$. A common approach to removing the information of b_i encoded in $\boldsymbol{z}$ is gradient resversal layer (GRL) (Ganin and Lempitsky, 2015) added before the debiasing head. GRL multiplies the gradient of $\mathcal{L}_{b_i}$ with a factor of $-\gamma_i$, and thereby simplifies the learning process to a standard gradient-based optimization. In DAM, this reversed gradient of $\mathcal{L}_{b_i}$ is used to learn the parameters of the debiasing adapter corresponding to the protected attribute b_i. Once adapters are trained, the fusion layer is learned jointly according to all task and debiasing objectives, as formulated below:

$$\mathcal{L}^{\text{Adv}}_{\text{Fusion}} = \mathcal{L}_t + \sum_{i=1}^{k} \mathcal{L}_{b_i} \tag{1}$$

Note that, while no weights are used to directly scale the individual loss functions, the effects of bias mitigation losses on model parameters are adjusted via their corresponding γ_i hyperparameters.

4 Experiment Setup

We evaluate DAM on three classification tasks and compare the results with a set of strong baselines.

Tasks and Data We conduct our experiments on three classification tasks. The first task is **biography classification** using the BIOS (De-Arteaga et al., 2019) dataset. The objective of the task is to predict a person's job given their biography, where the name and any indication of the person's gender (such as pronouns) in the biography is omitted. The protected attribute is the person's gender. The BIOS dataset contains around 430K data points with 28 occupations as the core task's classes, and two protected attribute classes (female/male).

The second task is **hate speech detection**, a sensitive task, as its use in any automated system with biases towards any societal group can be extremely delicate (e. g. in automatic social media moderation). Xia et al. (2020) show the existence of a strong correlation between dialect-based racial bias and hate speech detection. In our experiments, we use FDCL18 (Founta et al., 2018), a dataset of tweets annotated with four hate speech labels: *normal*, *spam*, *abusive*, and *hateful*. Following previous works (Sap et al., 2019; Ravfogel et al., 2020; Zhang et al., 2021), we assign artificial race labels of *African American*, and *White American* to FDCL18 (more details in Appendix A.1). In this setting, the main task is hate speech classification, and the protected attribute is the dialect-based racial identity of the tweet author. Our final dataset contains approximately 62K data points.

Model		# Params
BERT-Mini	FT	$11,237,380$
	FT-DEBIAS	$11,568,910$
	ADP	$330,500$
	ADP-DEBIAS	$432,206$
	DAM	$1,186,830$
BERT-Base	FT	$110,075,908$
	FT-DEBIAS	$113,036,558$
	ADP	$14,767,876$
	ADP-DEBIAS	$4,448,846$
	DAM	$24,806,414$

Table 1: Number of trainable parameters. The adapter-based models contain significantly fewer trainable parameters in comparison with fully fine-tuned models.

While the first two tasks contain one protected attribute, in our third classification task, we aim to explore the effect of more than one biases on our models. We therefore opt for a subset of the PAN16 dataset (Pardo et al., 2016). This dataset has been designed for the task of **mention detection**, namely to predict whether a tweet includes the mention of other users, and the protected attributes are gender and age of the tweet's author. The dataset contains approximately 200K data points with binary task classes (*mention*, *no mention*), as well as two gender labels and five age groups. Further details on the three datasets are provided in Appendix A.1.

Models and Baselines We conduct the experiments on the following models: **FT**: fully fine-tuning the PLM on the task. **FT-DEBIAS**: fully fine-tuning the PLM on the task as well as on the corresponding adversarial debiasing objective(s). **INLP**: using the implementation and suggested hyperparameter setting by Ravfogel et al. (2020), in particular with a linear task classifier (Logistic

Model		BIOS (gender)			FDCL18 (race)		
		Task↑	**Attacker↓**		**Task↑**	**Attacker↓**	
			Acc	BAcc		Acc	BAcc
BERT-Mini	FT	$81.23_{0.8}$	$61.05_{0.7}$	$61.05_{0.8}$	$79.45_{0.8}$	$93.38_{0.2}$	$76.87_{2.9}$
	FT-DEBIAS	$80.17_{0.7}$	$54.93^{\clubsuit}_{0.3}$	$55.03^{\clubsuit}_{0.9}$	$79.16_{1.6}$	$87.29_{1.8}$	$65.33_{1.9}$
	INLP	$71.08_{0.2}$	$60.11_{0.8}$	$59.83_{0.9}$	$73.60_{1.6}$	$92.13_{1.4}$	$71.75_{0.4}$
	INLP-NONLIN	$54.17_{2.1}$	$60.23_{1.2}$	$59.90_{1.3}$	$51.49_{0.2}$	$92.19_{0.0}$	$92.19_{0.0}$
	ADP	$79.20_{0.4}$	$62.72_{0.3}$	$62.91_{0.6}$	$75.45_{2.0}$	$92.86_{1.6}$	$75.94_{3.4}$
	ADP-DEBIAS	$72.73_{1.7}$	$62.34_{2.1}$	$62.52_{1.7}$	$65.68_{0.4}$	$\mathbf{85.07}^{\clubsuit}_{1.1}$	$63.58_{1.3}$
	DAM	$\mathbf{80.58}^{\clubsuit}_{0.4}$	$\mathbf{56.41}_{0.7}$	$\mathbf{56.60}_{0.4}$	$\mathbf{81.40}^{\clubsuit}_{0.4}$	$86.05_{2.6}$	$\mathbf{63.30}^{\clubsuit}_{0.5}$
BERT-Base	FT	$83.73_{1.0}$	$59.10_{0.3}$	$59.26_{0.8}$	$80.93_{0.8}$	$91.31_{0.8}$	$72.16_{1.8}$
	FT-DEBIAS	$83.29^{\clubsuit}_{0.6}$	$58.35_{0.5}$	$58.30_{0.7}$	$80.59_{0.1}$	$87.23_{0.9}$	$57.40^{\clubsuit}_{3.3}$
	INLP	$66.25_{0.4}$	$52.42^{\clubsuit}_{0.8}$	$50.51^{\clubsuit}_{0.3}$	$71.63_{2.2}$	$90.20_{0.4}$	$69.87_{0.2}$
	INLP-NONLIN	$24.44_{3.3}$	$55.41_{1.1}$	$54.81_{0.9}$	$42.23_{0.4}$	$90.97_{0.2}$	$91.67_{0.3}$
	ADP	$77.78_{0.2}$	$63.56_{0.7}$	$63.87_{0.6}$	$79.36_{1.3}$	$\mathbf{84.86}^{\clubsuit}_{0.6}$	$87.66_{3.2}$
	ADP-DEBIAS	$79.58_{0.0}$	$63.26_{0.8}$	$63.34_{0.9}$	$80.71_{0.6}$	$86.56_{0.2}$	$65.58_{0.1}$
	DAM	$\mathbf{80.93}_{0.1}$	$\mathbf{60.57}_{0.7}$	$\mathbf{60.70}_{0.2}$	$\mathbf{80.97}^{\clubsuit}_{0.2}$	$88.97_{0.2}$	$\mathbf{64.35}_{1.0}$

Table 2: Results on BIOS and FDCL18 datasets for the task classifiers (accuracy) and protected attributes' attackers (accuracy–Acc and balanced accuracy–BAcc), reported as the mean and standard deviation (in $_{\text{subscript}}$) over three runs. Arrows indicate the direction of better results. On each evaluation metric and each PLM, the best results among adapter-based models are shown in **bold**, and over all models by symbol ♣.

Regressor). **INLP-NONLIN**: the INLP model with the same setting, but instead of Logistic Regressor, INLP-NONLIN uses a non-linear task classifier: a two-layer feed-forward layer with a tanh activation. This setting follows the same configuration of all other baselines/models with respect to the task classifier. **ADP**: adding one adapter to the PLM and training it only through the task objective. **ADP-DEBIAS**: adding one adapter to the PLM and training it through the task objective as well as the adversarial debiasing objective(s). Finally, our proposed **DAM** model, which consists of one task adapter and separate debiasing adapters for each protected attribute, all combined with a fusion module. DAM uses the same task adapter trained by ADP and further learns the debiasing adapters and fusion layer. We should note that, since through adapter-based training the task performance of ADP might be different from the one of FT, a fair comparison of the core task performance should be between the results of ADP and its respective DAM model. We emphasize that we are especially interested in adapters because they preserve the original model and can be disengaged at will, to trade off debiasing for utility/performance.

As the PLM encoder for all models, we conduct our experiments using two versions of BERT (Devlin et al., 2019) with different sizes, namely BERT-Mini (Turc et al., 2019) and BERT-Base in order to provide a more comprehensive picture regarding the effect of encoder size and number of involved parameters. The number of trainable parameters of the models is reported in Table 1. The complete details of our hyperparameters setting and training procedure are explained in Appendix A.2 and Appendix A.3, respectively.

Evaluation and Attackers To evaluate the performance of the classifiers on the core task, we use the accuracy metric. To evaluate bias mitigation, following previous works (Elazar and Goldberg, 2018; Barrett et al., 2019), we report the leakage of a protected attribute in terms of accuracy and balanced accuracy of an *attacker*. To train an attacker, we freeze the model's parameters, and train a new classification head (two-layer feed-forward layer with a tanh(x) activation) to predict the protected attribute from the $\boldsymbol{z}$ vector. We train an ensemble of 5 attackers with the same configuration and training procedure, and report the results of the best performing one. We report the performance of the attacker in terms of accuracy and also balanced/macro accuracy (average of per-class ac-

Model		Task↑	Attackers↓			
			Gender		Age	
			Acc	BAcc	Acc	BAcc
BERT-Mini	Ft	$91.54^{\clubsuit}_{0.0}$	$62.07_{1.0}$	$61.78_{0.9}$	$30.25_{1.0}$	$27.25_{0.4}$
	Ft-Debias$_{G}$	$92.11_{0.3}$	$58.36_{0.5}$	$58.81_{0.1}$	$28.89_{0.1}$	$26.91_{0.8}$
	Ft-Debias$_{A}$	$91.39_{0.1}$	$60.15_{0.5}$	$60.08_{0.6}$	$33.94_{2.5}$	$27.72_{0.3}$
	Ft-Debias$_{G+A}$	$89.91_{0.2}$	$57.38_{0.4}$	$56.82_{0.6}$	$34.92_{0.8}$	$28.21_{0.1}$
	INLP$_{G}$	$69.26_{0.1}$	$59.52_{1.1}$	$60.06_{0.3}$	$34.71_{0.4}$	$29.21_{0.8}$
	INLP-NonLin$_{G}$	$67.00_{0.1}$	$60.25_{0.5}$	$60.35_{0.2}$	$33.78_{0.3}$	$30.03_{0.1}$
	INLP$_{A}$	$66.72_{1.8}$	$60.78_{0.1}$	$60.62_{0.1}$	$27.56^{\clubsuit}_{0.1}$	$25.63_{0.2}$
	INLP-NonLin$_{A}$	$62.21_{6.6}$	$60.52_{0.2}$	$60.39_{0.3}$	$31.06_{0.2}$	$25.51_{0.3}$
	Adp	$88.63_{0.1}$	$62.44_{1.9}$	$62.47_{1.7}$	$33.75_{4.1}$	$30.66_{2.6}$
	Adp-Debias$_{G}$	$78.07_{0.2}$	$56.44_{0.8}$	$55.59_{0.9}$	$33.80_{0.6}$	$28.67_{0.0}$
	Adp-Debias$_{A}$	$77.16_{0.2}$	$60.24_{0.4}$	$59.75_{0.5}$	$39.52_{0.6}$	$37.28_{6.0}$
	Adp-Debias$_{G+A}$	$77.28_{0.2}$	$55.92_{0.1}$	$55.38_{0.6}$	$38.93_{0.3}$	$33.41_{7.5}$
	DAM$_{G}$	$\mathbf{89.42}_{0.1}$	$55.45_{0.5}$	$54.69_{0.7}$	$\mathbf{32.11}_{0.0}$	$28.74_{0.3}$
	DAM$_{A}$	$89.14_{0.1}$	$60.77_{0.6}$	$60.72_{0.3}$	$36.53_{1.1}$	$\mathbf{22.98}^{\clubsuit}_{1.4}$
	DAM$_{G+A}$	$89.10_{0.2}$	$\mathbf{54.78}^{\clubsuit}_{0.1}$	$\mathbf{53.93}^{\clubsuit}_{0.1}$	$37.92_{0.1}$	$23.08_{0.4}$
BERT-Base	Ft	$93.50^{\clubsuit}_{0.0}$	$57.01_{0.7}$	$56.63_{0.3}$	$36.75_{1.5}$	$28.12_{2.7}$
	Ft-Debias$_{G}$	$93.56_{0.0}$	$61.76_{0.3}$	$61.46_{0.3}$	$34.65_{0.2}$	$26.42_{0.2}$
	Ft-Debias$_{A}$	$93.30_{0.3}$	$56.68_{1.7}$	$55.89_{1.9}$	$37.78_{1.0}$	$29.32_{1.9}$
	Ft-Debias$_{G+A}$	$92.81_{0.1}$	$55.74_{0.5}$	$54.96_{0.9}$	$38.09_{3.6}$	$30.94_{4.4}$
	INLP$_{G}$	$69.37_{0.1}$	$54.38_{0.1}$	$54.40_{0.1}$	$35.31_{0.7}$	$25.53_{0.2}$
	INLP-NonLin$_{G}$	$65.42_{0.1}$	$54.27_{0.2}$	$54.11_{0.1}$	$37.36_{1.2}$	$26.98_{0.2}$
	INLP$_{A}$	$49.82_{0.8}$	$54.95_{0.7}$	$54.61_{0.6}$	$34.53_{0.3}$	$27.55_{0.2}$
	INLP-NonLin$_{A}$	$50.18_{5.3}$	$54.44_{0.3}$	$54.44_{0.4}$	$33.66^{\clubsuit}_{0.1}$	$33.66_{0.0}$
	Adp	$93.16_{0.0}$	$63.42_{0.1}$	$63.24_{0.1}$	$\mathbf{36.11}_{0.4}$	$31.28_{0.7}$
	Adp-Debias$_{G}$	$87.00_{0.0}$	$55.05_{0.1}$	$53.83_{0.0}$	$36.18_{0.1}$	$27.93_{0.1}$
	Adp-Debias$_{A}$	$86.26_{0.1}$	$57.93_{0.4}$	$57.93_{0.6}$	$38.44_{0.2}$	$29.29_{6.0}$
	Adp-Debias$_{G+A}$	$86.32_{0.1}$	$54.36_{0.0}$	$52.99_{0.0}$	$38.14_{1.8}$	$38.99_{1.1}$
	DAM$_{G}$	$\mathbf{93.29}_{0.1}$	$54.05_{0.8}$	$52.55_{0.8}$	$39.23_{0.5}$	$25.67_{2.7}$
	DAM$_{A}$	$93.04_{0.1}$	$56.93_{2.7}$	$57.37_{2.7}$	$37.87_{0.8}$	$23.79_{0.9}$
	DAM$_{G+A}$	$93.15_{0.0}$	$\mathbf{53.01}^{\clubsuit}_{1.7}$	$\mathbf{52.53}^{\clubsuit}_{0.8}$	$38.23_{1.5}$	$\mathbf{23.24}^{\clubsuit}_{1.6}$

Table 3: Results of the PAN16 mention detection task with gender (G) and age (A) as protected attributes on the task classifiers (accuracy) and protected attributes' attackers (accuracy–Acc and balanced accuracy–BAcc). The mean and standard deviation (in $_{\text{subscript}}$) over three runs are reported. On each evaluation metric and each PLM, the best results among adapter-based models are shown in **bold**, and over all models by symbol ♣.

curacy scores). Balanced accuracy has the benefit of better reflecting the performance of the methods when considering sub- and minority groups. We believe this is particularly important in this setting, since the datasets are unbalanced over both task and protected labels (see Appendix A.1). To account for possible variabilities in training, we repeat every experiment three times and report the mean as well as standard deviation of the achieved results.

5 Results and Discussion

In this section, we first discuss the results on the datasets with one protected attribute (BIOS and FDCL18), followed by investigating the effect of DAM on PAN16 with two protected attributes.

Single-attribute bias mitigation The results on BIOS and FDCL18 datasets are reported in Table 2. We start with the results of BERT-Mini on both datasets. As shown, consistent with previous studies the task performance of ADP remains on par with the full model fine-tuning FT. In both FT and ADP, we observe that the adversarial debiasing (FT-DEBIAS and ADP-DEBIAS respectively) significantly reduces leakage in terms of Acc and BAcc of attackers, while task performance remains on par with the corresponding baseline models. Overall, adversarial methods outperform respective INLP models, particularly in respect to maintaining task performance during bias mitigation.

Compared to ADP and ADP-DEBIAS, we observe that DAM improves ADP-DEBIAS in terms of bias mitigation, while maintaining or even slightly improving performance on the task. The difference between DAM and ADP-DEBIAS indicates the benefit of learning separate adapters for performing the task and bias mitigation, and then combining them. Also comparing DAM with FT-DEBIAS, overall DAM performs on par in terms of task performance and leakage, while DAM provides the additional benefits of selectively engaging bias mitigation and significantly fewer trainable parameters.

Comparing the results between the core PLMs, we see that the above observations for BERT-Mini also hold true for BERT-Base on the BIOS dataset. On the FDCL18 dataset with BERT-Mini, DAM is in fact the only bias mitigation approach that can effectively prevent leakage while also preserving performance. It is the best in terms of leakage reduction, while its task performance is better than ADP. The situation differs for results of BERT-Base on FDCL18, as leakage balanced accuracy of FT-DEBIAS is surprisingly low.

Multi/Two-attribute bias mitigation Table 3 reports the results on PAN16 with two protected attributes, gender (G) and age (A). To observe the effect of single- versus multiple-attribute debiasing, we train the debiasing models (FT-DEBIAS, ADP-DEBIAS, and DAM) in three variations denoted with subscripts G, A, and G+A, indicating the debiasing optimization on only gender, only age, and both gender and age, respectively.[1]

Observing similar patterns to the previous experiments, DAM$_G$, DAM$_A$, and DAM$_{G+A}$ provide a significant improvement in bias mitigation (lower leakage) in comparison with their respective debiasing baseline variations.[2] Also similar to single-attribute datasets, ADP performs slightly worse than FT in respect to task performance, but DAM models remain on par with ADP or slightly improve it. We observe these patterns consistently on both BERT-Mini and BERT-Base. Overall, DAM provides equal or better bias mitigation in comparison with FT-DEBIAS and INLP, while offering on-demand debiasing and parameter efficiency.

Particularly with respect to multi-attribute bias mitigation, the results show the benefits of DAM$_{A+G}$ in separately learning multiple debiasing adapters and combining them, when compared with FT-DEBIAS$_{G+A}$, which simultaneously learns both debiasing factors. In particular, looking at the attacker's BAcc, while FT-DEBIAS$_{G+A}$ is not able to debias the age attribute (in comparison to FT), our DAM$_{G+A}$ provides strongly improved bias mitigation for age (and also gender), with similar or better results to only optimizing for gender or age (DAM$_G$ and DAM$_A$, respectively). This highlights the effectiveness of DAM in preventing catastrophic forgetting of debiasing functionalities in multi-attribute scenarios.[3] We further provide an investigation of the attention distributions of DAM's fusion module in Appendix B.

6 Conclusion

We propose a novel bias mitigation approach which enables flexible switching between the original and debiased state of a model. Our proposed DAM method extends the idea of multi-task learning using AdapterFusion to bias mitigation, by first learning the main task and the debiasing objectives as separate adapters, and then leveraging the attention-based fusion approach to merge these adapters and deliver debiased results. Our experiments on three classification tasks show that, beside flexible switching, DAM improves bias mitigation, provides on par classification performance to vanilla adversarial debiasing, and addresses the issue of catastrophic forgetting in multi-attribute bias mitigation.

---

[1] We run INLP and INLP-NONLIN on the two variations of G and A, but we are not aware of a principled way for this method to simultaneously debias more than one protected attribute.

[2] BAcc is more suited for comparisons of age on this dataset, as the age labels are highly imbalanced.

[3] Our experiments using DAM also indicate a possible correlation between Gender and Age, as debiasing Gender (DAM$_G$) leads to slight reduction in Age leakage, while similarly, debiasing Age (DAM$_A$) results in a slight removal of gender information.

7 Limitations

In the experiments, gender is considered as a binary construct due to practical constraints. In particular, in BIOS and PAN16 the gender is provided only in the form of male and female. We are however fully aware that a gender binary model is not representative of all individuals; yet working with in-the-wild data (as in our case) entails an unavoidable caveat, derived from the still predominant belief that human beings can be sorted into two discrete categories (Hyde et al., 2019). However, the proposed method can still be defined for generic non-binary settings and can be applied to any sensitive attribute with more than two categories, as exemplified by our consideration of age classes.

We should also highlight two limitations of this study in respect to the model and optimization. First, adversarial approaches in general (including our proposed approach) aims to reduce the correlations in the model to the protected attribute based on the observed data. This approach, like other data-oriented bias mitigation methods, might lack effective generalization, particularly when the model is evaluated on other domains or out-of-distribution data. The second limitation regarding the discussed bias mitigation methods based on adversarial training (also involving our proposed approach) is that, while the method aims to make the model prediction agnostic to protected attributes, it does not directly account for a balanced treatment of subpopulations in regards to the utility metrics and quality of the received service.

8 Acknowledgment

This work received financial support by the Austrian Science Fund (FWF): P33526 and DFH-23; and by the State of Upper Austria and the Federal Ministry of Education, Science, and Research, through grants LIT-2020-9-SEE-113 and LIT-2021-YOU-215.

References

Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. *Fairness and Machine Learning*. fairmlbook.org. `http://www.fairmlbook.org`.

Maria Barrett, Yova Kementchedjhieva, Yanai Elazar, Desmond Elliott, and Anders Søgaard. 2019. Adversarial removal of demographic attributes revisited. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6331–6336.

Rishabh Bhardwaj, Navonil Majumder, and Soujanya Poria. 2021. Investigating gender bias in BERT. *Cognitive Computation*, 13(4):1008–1018.

Asia J Biega, Krishna P Gummadi, and Gerhard Weikum. 2018. Equity of attention: Amortizing individual fairness in rankings. In *The 41st international ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 405–414.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in nlp. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of African-American English. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1130, Austin, Texas. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in Neural Information Processing Systems*.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7870–7881, Online. Association for Computational Linguistics.

Pengyu Cheng, Weituo Hao, Siyang Yuan, Shijing Si, and Lawrence Carin. 2020a. Fairfil: Contrastive neural debiasing method for pretrained text encoders. In *International Conference on Learning Representations*.

Pengyu Cheng, Martin Renqiang Min, Dinghan Shen, Christopher Malon, Yizhe Zhang, Yitong Li, and Lawrence Carin. 2020b. Improving disentangled text representation learning with information-theoretic guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7530–7541, Online. Association for Computational Linguistics.

Pierre Colombo, Pablo Piantanida, and Chloé Clavel. 2021. A novel estimator of mutual information for learning to disentangle textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6539–6550.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 4171–4186. Association for Computational Linguistics.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 11–21.

Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

Christian Ganhör, David Penz, Navid Rekabsaz, Oleg Lesota, and Markus Schedl. 2022. Mitigating consumer biases in recommendations with adversarial training. In *Proceedings of the 45th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2022*. ACM.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Yue Guo, Yi Yang, and Ahmed Abbasi. 2022. Auto-debias: Debiasing masked language models with automated biased prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1012–1023.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021a. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021b. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, volume 97, pages 2790–2799. Proceedings of Machine Learning Research.

Janet Shibley Hyde, Rebecca S Bigler, Daphna Joel, Charlotte Chucky Tate, and Sari M van Anders. 2019. The future of sex and gender in psychology: Five challenges to the gender binary. *American Psychologist*, 74(2):171–193.

Masahiro Kaneko and Danushka Bollegala. 2021. Debiasing pre-trained contextualised embeddings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1256–1266.

Hannah Rose Kirk, Filippo Volpin, Haider Iqbal, Elias Benussi, Frederic Dreyer, Aleksandar Shtedritski, Yuki Asano, et al. 2021. Bias out-of-the-box: An empirical analysis of intersectional occupational biases in popular generative language models. *Advances in Neural Information Processing Systems*, 34.

Klara Krieg, Emilia Parada-Cabaleiro, Gertraud Medicus, Oleg Lesota, Markus Schedl, and Navid Rekabsaz. 2022. Grep-biasir: A dataset for investigating gender representation-bias in information retrieval results. *arXiv preprint arXiv:2201.07754*.

Anne Lauscher, Tobias Lueken, and Goran Glavaš. 2021. Sustainable modular debiasing of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4782–4797, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards understanding and mitigating social biases in language models. In *International Conference on Machine Learning*, pages 6565–6576. PMLR.

David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. 2018. Learning adversarially fair and transferable representations. In *Proceedings of the International Conference on Machine Learning*, pages 3384–3393. PMLR.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021a. Compacter: Efficient low-rank hypercomplex adapter layers. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 1022–1035.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021b. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational*

Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, pages 565–576. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Francisco Manuel Rangel Pardo, Paolo Rosso, Ben Verhoeven, Walter Daelemans, Martin Potthast, and Benno Stein. 2016. Overview of the 4th author profiling task at pan 2016: Cross-genre evaluations. In *CLEF*.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. 2021. What to pre-train on? efficient intermediate task selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10585–10605.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256.

Shauli Ravfogel, Michael Twiton, Yoav Goldberg, and Ryan D Cotterell. 2022. Linear adversarial concept erasure. In *International Conference on Machine Learning*, pages 18400–18421. PMLR.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30.

Navid Rekabsaz, Simone Kopeinik, and Markus Schedl. 2021a. Societal biases in retrieved contents: Measurement framework and adversarial mitigation of BERT rankers. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 306–316.

Navid Rekabsaz and Markus Schedl. 2020. Do neural ranking models intensify gender bias? In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2065–2068.

Navid Rekabsaz, Robert West, James Henderson, and Allan Hanbury. 2021b. Measuring societal biases in text corpora via first-order co-occurrence. *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM)*.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in NLP. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3398–3403.

Gabriel Stanovsky, Noah A Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 6000—6010. Curran Associates, Inc.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in Neural Information Processing Systems*, 33:12388–12401.

Liwen Wang, Yuanmeng Yan, Keqing He, Yanan Wu, and Weiran Xu. 2021. Dynamically disentangling social bias from task-oriented representations with adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3740–3750.

Mengzhou Xia, Anjalie Field, and Yulia Tsvetkov. 2020. Demoting racial bias in hate speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 7–14, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. 2017. Controllable invariance through adversarial feature learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 585–596.

George Zerveas, Navid Rekabsaz, Daniel Cohen, and Carsten Eickhoff. 2022. Mitigating bias in search results through set-based document reranking and neutrality regularization. In *Proceedings of the 45th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2022*. ACM.

Xiongyi Zhang, Jan-Willem van de Meent, and Byron C Wallace. 2021. Disentangling representations of text by masking transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 778–791.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 629–634.

A Experiment Settings – Additional Details

A.1 Datasets

In FDCL18 dataset, we use the TwitterAAE model (Blodgett et al., 2016) to assign racial dialect classes. The TwiiterAAE model predicts four racial classes, *African American*, *White American*, *Hispanic*, and *Others*. We labeled a tweet as *African American* or *White American* if the prediction score was greater than 0.5. For PAN16 dataset, following (Sap et al., 2019) we balanced the task labels and sampled 200K data. The age groups of this dataset are 18-24, 25-34, 35-49, 50-64, and 65+. The proportions of data items regarding the labels of the task and protected attributes in the three dataset are as follows:

- BIOS dataset: Task (dentist: 0.03, professor: 0.27, teacher: 0.04, psychologist: 0.04, nurse: 0.07, poet: 0.01, photographer: 0.06, journalist: 0.04, filmmaker: 0.02, physician: 0.08, composer: 0.2, attorney: 0.08, model: 0.03, painter: 0.02, accountant: 0.01, pastor: 0.01, comedian: 0.01, surgeon: 0.05, architect: 0.03, paralegal: 0.01, dj: 0.01, chiropractor: 0.01, software engineer: 0.02, dietitian: 0.02, rapper: 0.01, personal trainer: 0.003, yoga teacher: 0.01, interior designer: 0.01); Gender (Female: 0.5, Male: 0.5)

- FDCL18 dataset: Task (normal: 0.73, spam: 0.12, abusive: 0.12, hateful: 0.04); Race (White: 0.5, AA: 0.5)

- PAN16 dataset: Task (Mention: 0.5, Not Mention: 0.5); Gender (male: 0.54, female: 0.46); Age (3: 0.18, 1: 0.34, 2: 0.40, 0: 0.07, 4: 0.01)

A.2 Hyperparameter setting

Across experiments, we keep specific hyperparameters consistent. Batch size is 64, learning rate is 2e-5 (except for training task and debiasing adapter as explained below), training epochs is 20, dropout probability is 0.3, and adversarial debiasing coefficient is 1 for all models (when applicable).

For task adapter and debiasing adapters, we tune the learning rate and the hidden layer dimension of adapter. We conduct brute search over the learning rate values of [1e-5, 1e-4,1e-3,1e-2], and hidden layer dimension with a division factor of [1,1/2,1/4,1/8,1/16]. For INLP we use 10 iterations and 10 classifiers to learn null space while for INLP-NonLin we same setting (300 iterations and 300 classifiers) as in (Ravfogel et al., 2020).

A.3 Training procedure

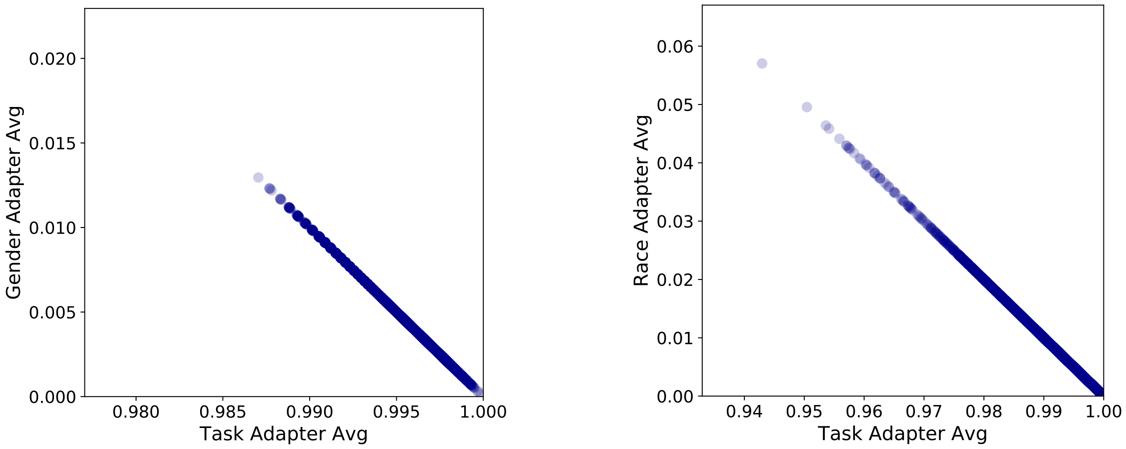
We randomly split the dataset into train, validation, and test set with the proportions 63:12:15 for BIOS, 63:12:15 for FDCL18, and 80:5:15 on PAN16. We use the validation set for hyperparameter tuning, and the best result on the validation set is evaluated on test set for the final results. The validation and test sets in all datasets follow the same distribution as the whole dataset. To address the unbalancedness of the dataset and the potential problems in adversarial learning, we apply upsampling only on the *training sets* of BIOS and FDCL18 datasets, to balance the protected attribute labels within each task label. For instance, genders are balanced in the dentist class by repeating the data items of the minority subgroup.

B Fusion Attention Analysis

We investigate the attention distribution of the fusion network and observe the weights it gives to the adapters. Figures 3a, 3b, and 4 depict the attention scores of each adapter averaged over all fusion layer in DAM for BIOS, and FDCL18, and PAN16 datasets, respectively. To avoid confusion in visualization, we only used 4% of data points randomly sampled from test set. As shown, the task adapter has weights close to 1, while debiasing adapters are assigned attention scores slightly higher than 0. The top three outliers in BIOS with the highest attentions on the debiasing adapter are reported in Table 4.

	Text
1	passionately promotes healthy dietary and lifestyle choices to prevent disease and achieve optimal health
2	practices include clayton heights family dental street panorama family dental avenue and surrey family dental avenue
3	primary research interests include collective security and global health in an international relations and international political economy perspective

Table 4



(a) BIOS

(b) FDCL18

Figure 3: Attention distribution of a set of sampled data points in the fusion module of DAM.

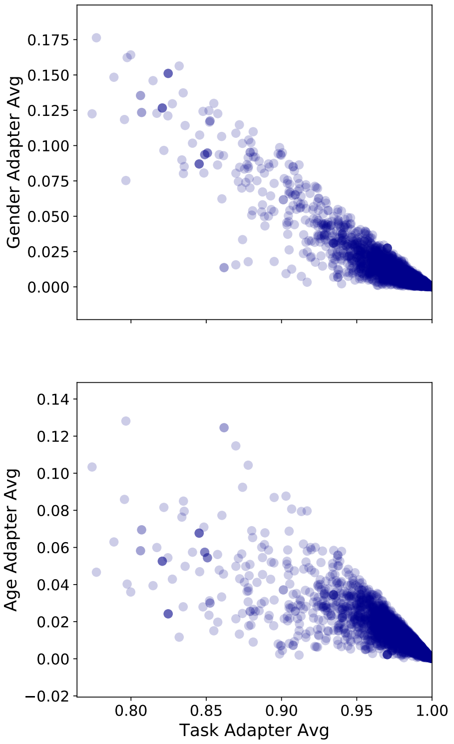


Figure 4: Attention distribution of a set of sampled data points in the fusion module of DAM in PAN16 dataset.

LINGMESS: Linguistically Informed Multi Expert Scorers for Coreference Resolution

Shon Otmazgin[1] **Arie Cattan**[1] **Yoav Goldberg**[1,2]
[1]Computer Science Department, Bar Ilan University
[2]Allen Institute for Artificial Intelligence
{shon711,arie.cattan,yoav.goldberg}@gmail.com

Abstract

Current state-of-the-art coreference systems are based on a single pairwise scoring component, which assigns to each pair of mention spans a score reflecting their tendency to corefer to each other. We observe that different kinds of mention pairs require different information sources to assess their score. We present LINGMESS, a linguistically motivated categorization of mention-pairs into 6 types of coreference decisions and learn a dedicated trainable scoring function for each category. This significantly improves the accuracy of the pairwise scorer as well as of the overall coreference performance on the English Ontonotes coreference corpus and 5 additional datasets.[1]

1 Introduction

Coreference resolution is the task of clustering textual mentions that refer to the same discourse entity. This fundamental task requires many decisions. In this work, we argue that different *kinds* of decisions involve different challenges. To illustrate that, consider the following text:

*"**Lionel Messi** has won a record seven Ballon d'Or awards. **He** signed for Paris Saint-Germain in August 2021. "**I** would like to thank **my** family", said **the Argentinian footballer**. **Messi** holds the records for most goals in La Liga"*

To correctly identify that the pronoun "He" refers to "Lionel Messi", models need to model the discourse, while linking "my" to "I" may rely more heavily on morphological agreement. Likewise, linking "the Argentinian footballer" to "Lionel Messi" requires world knowledge, while linking "Messi" to "Lionel Messi" may be achieved by simple lexical heuristics.

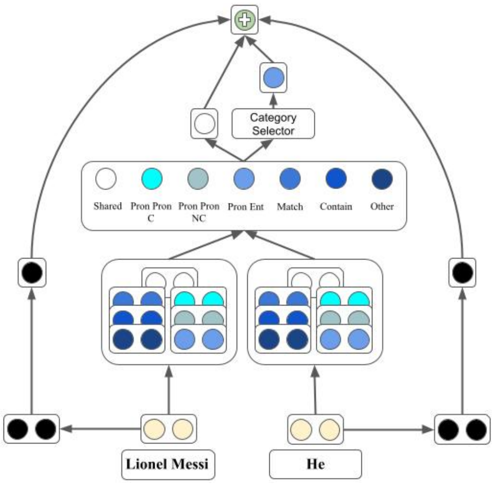


Figure 1: Architecture of our multi expert model. Given two spans *"Lionel Messi"* and *"He"*, we sum four scores: individual mention scores (black), f_m(*"Lionel Messi"*), f_m(*"He"*), and pairwise scores, shared antecedent score (white) f_a(*"Lionel Messi"*, *"He"*) and the relevant "expert" score (blue) $f_a^{\text{PRON-ENT}}$(*"Lionel Messi"*, *"He"*).

Indeed, pre-neural coreference resolution works often considered the types of a mention-pair, either by incorporating this information as model features, or by tailoring specific rules or specific models for each mention pair (see related work section).

However, neural-network based coreference models are all based on a *single* pairwise scorer that is shared for all mention pairs, regardless of the different challenges that needs to be addressed by each pair type (Lee et al., 2017, 2018; Joshi et al., 2019; Kantor and Globerson, 2019; Joshi et al., 2020; Xu and Choi, 2020; Xia et al., 2020; Toshniwal et al., 2020; Thirukovalluru et al., 2021; Kirstain et al., 2021; Dobrovolskii, 2021).

In this work, we suggest that modeling different mention pairs by different sub-models (in our case, different learned scoring functions) depending on their types is beneficial also for neural models. We identify a set of decisions: (a) linking compatible

---

[1] The codebase to train and run LINGMESS is available in `https://github.com/shon-otmazgin/lingmess-coref`. Also, our recent F-COREF Python package (Otmazgin et al., 2022) includes a simple and efficient implementation of LINGMESS in `https://github.com/shon-otmazgin/fastcoref`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2752–2760
May 2-6, 2023 ©2023 Association for Computational Linguistics

Category	Co-referring example	Non Co-referring example
PRON-PRON-C	*A couple of* ***my*** *law clerks were going to ... and* ***I*** *was afraid* ***I*** *was going to...*	*The Lord God said to* ***my*** *Lord: "Sit by me at* ***my*** *right side , and I will put your enemies ... "*
PRON-PRON-NC	*"****I*** *made a similar line and* ***I*** *produced it cheaper" ,* ***he*** *says.*	***She*** *is* ***my*** *Goddess ...*
ENT-PRON	***Spain, Argentina, Thailand and Indonesia*** *were doing too little to prevent ... across* ***their*** *borders.*	*Tonight, to kick off* ***the effort****, CNN will premiere* ***its*** *first prime - time newscast in years.*
MATCH	*... says Paul Amos,* ***CNN*** *executive vice president for programming. Accordingly,* ***CNN*** *is ...*	*Hertz and Avis can not benefit* ***Budget's*** *programs, " said Bob Wilson,* ***Budget's*** *vice president ...*
CONTAINS	*He reportedly showed* ***DeLay*** *a videotape that made him weep.* ***Tom DeLay*** *then ...*	*Give* ***SEC*** *authority to halt securities trading, (also opposed by new* ***SEC chairman****) ...*
OTHER	*They also saw* ***the two men who were standing with him****. When* ***Moses and Elijah*** *were leaving ...*	***The company*** *is already working on its own programming ...* ***the newspaper*** *said.*

Table 1: Example of each category, taken from Ontonotes development set. We define the categories of mention pairs as follows. PRON-PRON-C: compatible pronouns based on their attributes such as gender, number and animacy (see Appendix C for more details), PRON-PRON-NC: incompatible pronouns, ENT-PRON: a pronoun and another span, MATCH: non-pronoun spans with the same content words, CONTAINS: one contains the content words of the other, OTHER: all other pairs. Content words exclude stop words, see Appendix C for the full list of stop words.

pronouns (PRON-PRON-C); (b) linking incompatible pronouns (PRON-PRON-NC); (c) linking pronouns to entities (ENT-PRON); (d) linking entities which share the exact lexical form (MATCH); (e) linking entities where the lexical form of one contains the lexical form of the other (CONTAINS); (f) other cases (OTHER). Each of these classes is easy to identify deterministically, each contains both positive and negative instances, and each could benefit from a somewhat different decision process. Table 1 demonstrates the classes.[2]

We present **Ling**uistically Informed **M**ulti **E**xpert **S**corer**s** (LINGMESS), a coreference model which categorizes each pairwise decision into one of these classes, and learns, in addition to a shared scoring function, also a separate scoring function for each pair type. At inference time, for each pair of mentions being scored, we deterministically identify the pair's type, and use the corresponding scoring function.[3]

Specifically, we extend the recent *s2e*'s model (Kirstain et al., 2021) by adding per-category scoring, but the method is general and may work with other coreference models as well. As illustrated in Figure 1, the final coreference score between two spans is composed—in addition to the individual mention scores—of two pairwise scores: a shared antecedent-compatibility score and an "expert" antecedent compatibility score which depends on the linguistic-type of the pair.

We show that this significantly improves the coreference performance on Ontonotes (Pradhan et al., 2012) and 5 additional datasets. We also inspect the performance of the model for each category separately, showing that some classes improve more than others. This analysis provides a finer-grained understanding of the models and points out directions for future research.

2 Background: the s2e Model

The *s2e* model (Kirstain et al., 2021) achieves the current best coreference scores among all practical neural models.[4]

Given a sequence of tokens $x_1, \ldots, x_n$, each mention pair (c, q) is scored using a scoring function $F_{\text{S}}(c, q)$[5] described below, where c is a "candidate span" and q is a "query span" which appears before c in the sentence. The span encodings are based on contextualized word embeddings obtained by a Longformer encoder, see Kirstain et al. (2021) for details. These pairwise scores are then used to form coreference chains (see "inference" below).

---

[2] More fine-grained distinctions are of course also possible, but we leave exploration of them to future work.

[3] For computational efficiency on the GPU, we find it beneficial to compute all the scoring functions and mask away the not needed values.

[4] We define "practical models" as those that require a constant number transformer-based document encodings per passage, as opposed to a constant number of document encodings per mention. The CorefQA model (Wu et al., 2020) achieves a substantially higher score, but requires to run a separate BERT inference for each mention, making it highly impractical.

[5] S state for SHARED

The scoring function F_{S} is further decomposed:

$$F_{\text{S}}(c,q) = \begin{cases} f_m(c) + f_m(q) + f_a(c,q) & c \neq \varepsilon \\ 0 & c = \varepsilon \end{cases}$$

where ε is the null antecedent, and f_m and f_a are parameterized functions, scoring each individual span (f_m) and the pairwise interaction (f_a).

For each possible mention q, the learning objective optimizes the sum of probabilities over the true antecedent $\hat{c}$ of q:

$$L_{\text{S}}(q) = \log \sum_{\hat{c} \in \mathcal{C}(q) \cap \text{GOLD}(q)} P_{\text{S}}(\hat{c} \mid q)$$

where $\mathcal{C}(q)$ is the set of all candidate antecedents[6] with a null antecedent ε. $\text{GOLD}(q)$ is the set of the true antecedents of q. $P_{\text{S}}(c \mid q)$ is computed as a softmax over $F_{\text{S}}(c,q)$ scores for c values in $\mathcal{C}(q)$:

$$P_{\text{S}}(\hat{c} \mid q) = \frac{\exp F_{\text{S}}(\hat{c},q)}{\sum\limits_{c' \in \mathcal{C}(q)} \exp F_{\text{S}}(c',q)}$$

3 LINGMESS

Clustering coreferring entities typically involves many different phenomena, which we argue should be addressed in a different manner. Therefore, our core contribution is proposing to allocate a dedicated scorer $f_a^t(c,q)$ for each phenomenon type t, in addition to the shared pairwise scorer $f_a(c,q)$. The overall architecture of our model is shown in Figure 1.

Concretely, we extend the *s2e* model with six additional antecedent scorers f_a^t where $t \in$ {PRON-PRON-C, PRON-PRON-NC, ENT-PRON, MATCH, CONTAINS, OTHER}, the six categories we list in Table 1.

The pairwise scoring function now becomes:

$$F(c,q) = \begin{cases} f_m(c) + f_m(q) + f(c,q) & c \neq \varepsilon \\ 0 & c = \varepsilon \end{cases}$$

$$f(c,q) = f_a(c,q) + f_a^{T(c,q)}(c,q)$$

where $T(c,q)$ is a deterministic, rule-based function to determine the category t of the pair (c,q). The pairwise scoring function $f(c,q)$ scoring c as the antecedent of q, is now composed of a shared scorer $f_a(c,q)$ and an "expert" scorer $f_a^t(c,q)$ which differs based on the type of the pair c,q. Each of the seven pairwise scoring functions (f_a and the six f_a^t) is parameterized separately using its own set of matrices. The transformer-based encoder and the mention scorer f_m are shared between all the antecedent scorers. See Appendix A.2 for full model architecture.

Training For each span q, our model optimizes the objective function L_{COREF} over the sum of probabilities of all true antecedents of q:

$$L_{\text{COREF}}(q) = \log \sum_{\hat{c} \in \mathcal{C}(q) \cap \text{GOLD}(q)} P(\hat{c} \mid q)$$

Here, $P(\hat{c} \mid q)$ is a softmax over $F(\hat{c},q)$ scores, that is our new score function described in Figure 1.

$$P(\hat{c} \mid q) = \frac{\exp F(\hat{c},q)}{\sum\limits_{c' \in \mathcal{C}(q)} \exp F(c',q)}$$

This scorer is also the one used in inference. However, this objective does not explicitly push each category ("expert") to specialize. For example, for the PRON-PRON-C cases, it would be useful to explicitly train the model to distinguish between the possible antecedents of that type only (without regarding other antecedents), as well as to explicitly distinguish between a pronoun antecedent and a null antecedent. To this end, we extend the training objective by also training each "expert" separately:

$$L_t(q) = \log \sum_{\hat{c} \in \mathcal{C}_t(q) \cap \text{GOLD}(q)} P_t(\hat{c} \mid q)$$

$$P_t(\hat{c} \mid q) = \frac{\exp F_t(\hat{c},q)}{\sum\limits_{c' \in \mathcal{C}_t(q)} \exp F_t(c',q)}$$

$$F_t(c,q) = \begin{cases} f_m(c) + f_m(q) + f_a^t(c,q) & c \neq \varepsilon \\ 0 & c = \varepsilon \end{cases}$$

Note that for $L_t(q)$ we replace $\mathcal{C}(q)$ with $\mathcal{C}_t(q)$, considering only the potential antecedents that are compatible with the span q for the given type. For example, for L_{MATCH} and a span q, we will only consider candidates c which appear before q with the exact same content words as q. Our final objective for each mention span q is thus:

$$L(q) = L_{\text{COREF}}(q) + L_{\text{EXPERTS}}(q)$$

$$L_{\text{EXPERTS}}(q) = \sum_t L_t(q) + L_{\text{S}}(q)$$

---

[6] All spans before q that passed some pruning threshold.

	MUC			B^3			$CEAF_{\phi_4}$			LEA			
	R	P	F1	R	P	F1	R	P	F1	R	P	F1	Avg. F1
s2e	**85.2**	86.6	85.9	77.9	80.3	79.1	75.4	76.8	76.1	75.8	78.3	77.0	80.3
LINGMESS	85.1	**88.1**	**86.6**	**78.3**	**82.7**	**80.5**	**76.1**	**78.5**	**77.3**	**76.3**	**80.9**	**78.5**	**81.4**

Table 2: Performance on the test set of the English OntoNotes 5.0 dataset. The averaged F1 of MUC, B^3, $CEAF_\phi$ is the main evaluation metric. Our model outperforms the *s2e* model (Kirstain et al., 2021) by 1.1 CoNLL F1. The performance gain is statistically significant according to a non-parametric permutation test ($p < 0.05$).

	s2e	LINGMESS
WikiCoref (Ghaddar and Langlais, 2016)	59.7	**62.6**
GAP (Webster et al., 2018)	88.3	**89.6**
WinoGender (Rudinger et al., 2018)	70.5	**77.3**
WinoBias (Zhao et al., 2018)	84.3	**85.1**
BUG (Levy et al., 2021)	72.2	**74.6**

Table 3: Performance on the test set of various coreference datasets. The reported metrics are CoNLL F1 for WikiCoref, F1 for GAP and Accuracy for WinoGender, WinoBias and BUG.

	s2e			LINGMESS		
	P	R	F1	P	R	F1
PRON-PRON-C	88.8	71.3	79.1	88.0	85.1	**86.5**
PRON-PRON-NC	84.2	55.8	67.1	88.3	68.7	**77.3**
ENT-PRON	78.8	68.7	73.4	80.4	69.8	**74.7**
MATCH	85.6	90.2	87.8	85.3	93.7	**89.3**
CONTAINS	72.4	80.9	76.4	77.4	78.9	**78.1**
OTHER	60.1	70.2	64.7	71.7	64.2	**67.7**

Table 4: Pairwise performance by category, on the test set of the English OntoNotes 5.0 dataset. LINGMESS surpasses the *s2e* model (Kirstain et al., 2021) for most categories by a substantial margin.

Inference At inference time, we compute the score of each mention based on the shared scorer as well as the per-type scorer matching the mention type. We then form the coreference chains by linking each mention q to its most likely antecedent c according to $F(c, q)$. We do not use higher-order inference as it has been shown to have a marginal impact (Xu and Choi, 2020).

4 Experiments

Our baseline is the *s2e* model trained on the English OntoNotes 5.0 dataset by its authors (Kirstain et al., 2021). We train LINGMESS also on OntoNotes, and evaluate both models on OntoNotes, WikiCoref, GAP, WinoGender, WinoBias and BUG. Implementation details are described in Appendix B.

Performance Table 2 presents the performance of LINGMESS on the test set of Ontonotes. LINGMESS achieves 81.4 F1 on Ontonotes, while the *s2e* baseline achieves 80.3. Our performance gain is statistically significant according to a non-parametric permutation test ($p < 0.05$). Additionally, Table 3 shows that LINGMESS outperforms the *s2e* model on WikiCoref (+2.9) GAP (+1.3), WinoGender (+6.8), WinoBias (+0.8) and BUG (+2.4), indicating a better out-of-domain generalization.

Importance of per-category scoring. To assess that the improvement of LINGMESS is due to the decomposition into our set of categories and not to the added parameters, we do two experiments. First, we train a random baseline, which randomly assigns a category for each pair[7] and obtain similar results as the baseline. Second, we train our model by optimizing only the overall loss L_{COREF} and not L_{EXPERTS}. This achieves lower results than the baseline, due to low mention recall.

In addition to the standard coreference evaluation, we report pairwise performance for each category. Given a mention-pair (c, q), if $F(c, q)$ is higher than 0, we treat it as a positive prediction, otherwise negative. We then measure precision, recall and F1 based on gold clusters labels. Table 4 shows the pairwise performance of the *s2e* model and LINGMESS. LINGMESS outperforms *s2e* by a significant margin for all categories (e.g +7.4 F1 for PRON-PRON-C, +10.2 F1 for PRON-PRON-NC, etc.).[8] The performance varies across the different categories, suggesting aspects of the coreference problem where future work can attempt to improve.

The importance of the shared scorer. To investigate the role of the shared scorer, we trained the LINGMESS model with only the per-type pairwise scorers, excluding the shared pairwise scorer $F_S(c, q)$ and its accompanying loss term $L_S(q)$.

---

[7] For each pair of mentions (c, q), we take the modulo of the sum of the ASCII code of the last character of the last token of c and q.

[8] These gains in this pairwise metric are higher than the CoNLL metrics reported in Table 2, because the CoNLL metrics are based on the final clusters, after aggregation of individual pairwise decisions.

This resulted in a significant decrease in performance (-0.9), specifically in the recall of the mention detection component. However, adding the shared scorer was able to mitigate this degradation by balancing the different "experts" pairwise scorers.

5 Related Work

Many pre-neural works consider the various linguistic phenomena involved in coreference resolution as a different challenge. The early coreference system by Zhou and Su (2004) divided the antecedents candidates into distinct coreference categories (e.g., Name Alias, Apposition, Definite Noun, and a few more) and defined tailored rules for each category. Later, Lee et al. (2013) proposed the multi-sieves deterministic model, where each sieve adds coreference links between mention pairs from a specific linguistic category (e.g string match, compatible pronoun, etc.). Haghighi and Klein (2009) performed an error analysis of their coreference model according to different types of antecedent decisions, such as Proper Noun-Pronoun, Pronoun-Pronoun, etc. Based on this analysis, they focus on fixing the pronoun antecedent choices by adding syntactic features. More recently, Lu and Ng (2020) analyze empirically the performance of neural coreference resolvers on various fine-grained resolution categories of mentions (e.g gendered pronoun vs. 1st and 2nd pronoun). They find that while models perform well on name and nominal mention pairs with some shared content words, they still struggle with resolving pronouns, particularly relative pronouns.

Early supervised statistical models train a feature-based classifier that incorporates the type of antecedent decision (e.g. pronoun-entity, string match) as features at the mention-pair level (Soon et al., 2001; Bengtson and Roth, 2008; Clark and Manning, 2015, 2016). Subsequently, Denis and Baldridge (2008) demonstrate that training separate classifiers that specialize in particular types of mentions (e.g third person pronouns, speech pronouns, proper names, definite descriptions, and all other) provides significant performance improvements. Lassalle and Denis (2013) took that observation a step further and proposed a more advanced method for model specialization by learning to separate types of mention into optimal classes and their proper feature space.

In our work, we make progress in coreference systems specialization direction, and show that the incorporation of linguistic information is helpful also in the context of end-to-end neural models.

6 Conclusion

We present LINGMESS, a coreference model that significantly improves accuracy by splitting the scoring function into different categories, and routing each scoring decision to its own category based on a deterministic, linguistically informed heuristic. This indicates that while end-to-end training is very effective, linguistic knowledge and symbolic computation can still be used to improve results.

Limitations

In this paper, we consider a set of 6 linguistic categories of mention pairs, as listed in Table 1. These categories might not be optimal for the task, while a different set of finer-grained categories may result to a higher performance gain. Another aspect that can be considered as a limitation is the computation of the categories for every possible pairs. Although the model considers only the top-scoring spans, this additional computation layer increases training and inference time over the baseline (see Appendix B.3 for the exact time). Our linguistic heuristics could be improved by, e.g., running a parser and considering head words. However, we chose not to do so in this work as this will further increase runtime.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT). Arie Cattan is partially supported by the PBC fellowship for outstanding PhD candidates in data science.

References

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Honolulu, Hawaii. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2015. Entity-centric coreference resolution with model stacking. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1405–1415, Beijing, China. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2016. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Berlin, Germany. Association for Computational Linguistics.

Pascal Denis and Jason Baldridge. 2008. Specialized models and ranking for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 660–669, Honolulu, Hawaii. Association for Computational Linguistics.

Vladimir Dobrovolskii. 2021. Word-level coreference resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7670–7675, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Abbas Ghaddar and Phillippe Langlais. 2016. WikiCoref: An English coreference-annotated corpus of Wikipedia articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 136–142, Portorož, Slovenia. European Language Resources Association (ELRA).

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1152–1161, Singapore. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Ben Kantor and Amir Globerson. 2019. Coreference resolution with entity equalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 673–677, Florence, Italy. Association for Computational Linguistics.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference resolution without span representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 14–19, Online. Association for Computational Linguistics.

Emmanuel Lassalle and Pascal Denis. 2013. Improving pairwise coreference models through feature space hierarchy learning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 497–506, Sofia, Bulgaria. Association for Computational Linguistics.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Shahar Levy, Koren Lazar, and Gabriel Stanovsky. 2021. Collecting a large-scale gender bias dataset for coreference resolution and machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2470–2480, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2020. Conundrums in entity coreference resolution: Making sense of the state of the art. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6620–6631, Online. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages

25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Nafise Sadat Moosavi and Michael Strube. 2016. Which coreference evaluation metric do you trust? a proposal for a link-based entity aware metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Berlin, Germany. Association for Computational Linguistics.

Shon Otmazgin, Arie Cattan, and Yoav Goldberg. 2022. F-coref: Fast, accurate and easy to use coreference resolution. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 48–56, Taipei, Taiwan. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Raghuveer Thirukovalluru, Nicholas Monath, Kumar Shridhar, Manzil Zaheer, Mrinmaya Sachan, and Andrew McCallum. 2021. Scaling within document coreference to long texts. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3921–3931, Online. Association for Computational Linguistics.

Shubham Toshniwal, Sam Wiseman, Allyson Ettinger, Karen Livescu, and Kevin Gimpel. 2020. Learning to Ignore: Long Document Coreference with Bounded Memory Neural Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8519–8526, Online. Association for Computational Linguistics.

Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. 2022. deep-significance-easy and meaningful statistical significance testing in the age of neural networks. *arXiv preprint arXiv:2204.06815*.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Patrick Xia, João Sedoc, and Benjamin Van Durme. 2020. Incremental neural coreference resolution in constant memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8617–8624, Online. Association for Computational Linguistics.

Liyan Xu and Jinho D. Choi. 2020. Revealing the myth of higher-order inference in coreference resolution. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8527–8533, Online. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New

Orleans, Louisiana. Association for Computational Linguistics.

GuoDong Zhou and Jian Su. 2004. A high-performance coreference resolution system using a constraint-based multi-agent strategy. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 522–528, Geneva, Switzerland. COLING.

A Model Architecture

Given a sequence of tokens $x_1, ..., x_n$ from an input document, a transformer-based (BERT-like) encoder first forms contextualized representation vectors, $\mathbf{x_1}, ..., \mathbf{x_n}$ for each token in the sequence.

A.1 The *s2e* Model

Mention scorer Given a span $q = (\mathbf{x_i}, \mathbf{x_j})$, represented by its start and end tokens, the score for q being a mention is defined as follow:

$$\mathbf{m}_s(\mathbf{x}) = \text{GeLU}(\mathbf{W}_{\mathbf{m}_s}\mathbf{x}) \qquad \mathbf{m}_e(\mathbf{x}) = \text{GeLU}(\mathbf{W}_{\mathbf{m}_e}\mathbf{x})$$

$$\begin{aligned} f_m(q) = &\ \mathbf{m}_s(\mathbf{x_i}) \cdot \mathbf{v}_s + \mathbf{m}_s(\mathbf{x_j}) \cdot \mathbf{v}_e \\ &+ \mathbf{m}_s(\mathbf{x_i}) \cdot \mathbf{B}_m \cdot \mathbf{m}_s(\mathbf{x_j}) \end{aligned}$$

where $\mathbf{m}_s(\mathbf{x})$ and $\mathbf{m}_e(\mathbf{x})$ are two non-linear functions to obtain start and end representations for each token x, and $f_m(q)$ is a biaffine product over these representations.

Antecedent scorer Given two spans, $c = (\mathbf{x_i}, \mathbf{x_j})$ and $q = (\mathbf{x_k}, \mathbf{x_l})$, represented by their start and end tokens, the score for c being an antecedent of q is computed as follow:

$$\mathbf{a}_s(\mathbf{x}) = \text{GeLU}(\mathbf{W}_{\mathbf{a}_s}\mathbf{x}) \qquad \mathbf{a}_e(\mathbf{x}) = \text{GeLU}(\mathbf{W}_{\mathbf{a}_e}\mathbf{x})$$

$$\begin{aligned} f_a(c, q) = &\ \mathbf{a}_s(\mathbf{x_i}) \cdot \mathbf{B}_{ss} \cdot \mathbf{a}_s(\mathbf{x_k}) \\ &+ \mathbf{a}_e(\mathbf{x_j}) \cdot \mathbf{B}_{es} \cdot \mathbf{a}_s(\mathbf{x_k}) \\ &+ \mathbf{a}_s(\mathbf{x_i}) \cdot \mathbf{B}_{se} \cdot \mathbf{a}_e(\mathbf{x_l}) \\ &+ \mathbf{a}_e(\mathbf{x_j}) \cdot \mathbf{B}_{ee} \cdot \mathbf{a}_e(\mathbf{x_l}) \end{aligned}$$

Similar to the mention scorer, $\mathbf{a}_s(\mathbf{x})$ and $\mathbf{a}_e(\mathbf{x})$ are two non-linear functions to obtain start/end representations for each token, and $f_a(c, q)$ is a sum of four bilinear functions over the start and end representations of c and q.

A.2 LINGMESS

Mention scorer Our mention scorer is the same as *s2e* mention scorer implementation.

Antecedent scorer As mentioned in the paper (§3), in addition to the shared antecedent scorer $f_a(c, q)$, LINGMESS includes a dedicated antecedent scorer $f_a^t(c, q)$ for each category $t \in \{\text{PRON-PRON-C}, \text{PRON-PRON-NC}, \text{ENT-PRON}, \text{MATCH}, \text{CONTAINS}, \text{OTHER}\}$. The overall score for $c = (\mathbf{x_i}, \mathbf{x_j})$ being an antecedent of $q = (\mathbf{x_k}, \mathbf{x_l})$ becomes the sum of the shared scorer and the relevant category "expert" scorer:

$$f(c, q) = f_a(c, q) + f_a^{T(c,q)}(c, q)$$

where $T(c, q)$ is a deterministic function to determine the category t of the pair (c, q).

For each category t, we define two specific non-linear functions to obtain start and end representations ($\mathbf{a}_s^t(\mathbf{x})$ and $\mathbf{a}_e^t(\mathbf{x})$) as well as an "expert" antecedent scoring function $f_a^t(c, q)$:

$$\mathbf{a}_s^t(\mathbf{x}) = \text{GeLU}(\mathbf{W}_{\mathbf{a}_s}^t\mathbf{x}) \qquad \mathbf{a}_e^t(\mathbf{x}) = \text{GeLU}(\mathbf{W}_{\mathbf{a}_e}^t\mathbf{x})$$

$$\begin{aligned} f_a^t(c, q) = &\ \mathbf{a}_s^t(\mathbf{x_i}) \cdot \mathbf{B}_{ss}^t \cdot \mathbf{a}_s^t(\mathbf{x_k}) \\ &+ \mathbf{a}_e^t(\mathbf{x_j}) \cdot \mathbf{B}_{es}^t \cdot \mathbf{a}_s^t(\mathbf{x_k}) \\ &+ \mathbf{a}_s^t(\mathbf{x_i}) \cdot \mathbf{B}_{se}^t \cdot \mathbf{a}_e^t(\mathbf{x_l}) \\ &+ \mathbf{a}_e^t(\mathbf{x_j}) \cdot \mathbf{B}_{ee}^t \cdot \mathbf{a}_e^t(\mathbf{x_l}) \end{aligned}$$

Overall, our model introduces 6 learnable matrices for each category ($\mathbf{W}_{\mathbf{a}_s}^t$, $\mathbf{W}_{\mathbf{a}_e}^t$, $\mathbf{B}_{ss}^t$, $\mathbf{B}_{es}^t$, $\mathbf{B}_{se}^t$, $\mathbf{B}_{ee}^t$). The transformer-based encoder and the mention scorer are shared between all the different pairwise scorers.

B Implementation Details

B.1 Hyperparameteres

We extend the *s2e*'s implementation based on PyTorch (Paszke et al., 2019) and Transformers (Wolf et al., 2020). We used the same hyperparameters (e.g learning rate, warmup, etc.) as the *s2e* model except the hidden size of all matrices W and B. As our method introduces a dedicated antecedent scoring function f_a^t function for each category t, we reduce the size of these matrices from 3072 to 2048 to fit training into memory in our hardware. Similar to the baseline our head method is on top of Longformer-Large (Beltagy et al., 2020), resulting in a total of 590M learnable parameters (the *s2e* model contains 494M learnable parameters). We used dynamic batching both for training and inference, specifically 5K tokens in a single batch during training and 10K tokens at inference.

	Kirstain et al. (2021)			LINGMESS		
	P	R	F1	P	R	F1
PRON-PRON-C	91.7	77.5	84.0	91.7	90.2	**91.0**
PRON-PRON-NC	88.9	66.2	75.9	90.2	81.3	**85.5**
ENT-PRON	82.0	74.1	**77.9**	81.4	74.7	**77.9**
MATCH	88.3	87.5	87.9	88.4	92.0	**90.2**
CONTAINS	69.1	77.2	72.9	76.1	73.5	**74.8**
OTHER	56.8	67.5	61.7	70.8	64.4	**67.5**

Table 5: Pairwise performance by category, on the dev set of the English OntoNotes 5.0 dataset.

	Masc	Fem	Bias	Overall
Kirstain et al. (2021)	90.6	85.8	0.95	88.3
LINGMESS	**91.3**	**87.8**	**0.96**	**89.6**

Table 6: Performance on the test set of the GAP coreference dataset. The reported metrics are F1 scores.

B.2 Evaluation

As mentioned in the paper (§4), we conduct our experiments on the English portion of the OntoNotes corpus (Pradhan et al., 2012). This dataset contains 2802 documents for training, 343 for development, and 348 for test.

We evaluate our model according to the standard coreference metric: MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), CEAF$_{\phi_4}$ (Luo, 2005), and LEA (Moosavi and Strube, 2016) using the official CoNLL coreference scorer.[9] LINGMESS achieves 81.6 CoNLL F1 on the development set of Ontonotes. Also, Table 5 presents the pairwise performance on the development set for each category. We compute statistical significance with a non-parametric permutation test using Ulmer et al. (2022)'s implementation. Table 6 shows that LINGMESS consistently outperforms the *s2e* model on GAP.

B.3 Runtime and Memory

Our model was trained for 129 epochs on a single 32GB NVIDIA Tesla V100 SXM2 GPU. The training took 23 hours. At shown in Table 7, the run-time at inference time in LINGMESS is longer than in the *s2e* model because of the category selection for every possible pair of mentions. The memory consumption remains quite similar to the baseline.

	Runtime	Memory
Kirstain et al. (2021)	28	5.4
LINGMESS	43	5.9

Table 7: Inference time(Seconds) and memory(GiB) on 343 docs of OntoNotes development set. Using Dynamic batching, 10K tokens in a single batch. Hardware, NVIDIA Tesla V100 SXM2

ID	Pronouns
1	*I, me, my, mine, myself*
2	*you, your, yours, yourself, yourselves*
3	*he, him, his, himself*
4	*she, her, hers, herself*
5	*it, its, itself*
6	*we, us, our, ours, ourselves*
7	*they, them, their, themselves*
8	*that, this*

Table 8: List of groups of compatible pronouns, pronouns with the same ID are considered as compatible.

C Determining pair types

Our method routes each pair of spans to their corresponding category scorer. This decision is based on the linguistic properties of the spans. Given a mention-pair (c, q), we defined a rule based function $T(c, q)$ that determines the category of that pair. If c and q are both pronouns, if they are compatible according to gender, number and animacy (see Table 8 for the full list), the metnion pair will be routed to PRON-PRON-C, otherwise PRON-PRON-NC. If c is pronoun and q is a non-pronoun span (or vise-versa) we route the mention-pair to PRON-ENT. We route the remaining pairs to their corresponding categories (MATCH, CONTAINS or OTHER) by considering only content words, excluding the following stop words: *{'s, a, all, an, and, at, for, from, in, into, more, of, on, or, some, the, these, those}*. Accordingly, the mentions "*the U.S. and Japan*" and "*Japan and the U.S.*" are considered MATCH, "*This lake of fire*" and "*the lake of fire*" are considered CONTAINS and "*Bill Clinton*" and "*The President*" are considered OTHER.

---

[9] `https://github.com/conll/reference-coreference-scorers`.

Finding the Law: Enhancing Statutory Article Retrieval via Graph Neural Networks

Antoine Louis **Gijs van Dijck** **Gerasimos Spanakis**
Law & Tech Lab, Maastricht University
`{a.louis, gijs.vandijck, jerry.spanakis}@maastrichtuniversity.nl`

Abstract

Statutory article retrieval (SAR), the task of retrieving statute law articles relevant to a legal question, is a promising application of legal text processing. In particular, high-quality SAR systems can improve the work efficiency of legal professionals and provide basic legal assistance to citizens in need at no cost. Unlike traditional ad-hoc information retrieval, where each document is considered a complete source of information, SAR deals with texts whose full sense depends on complementary information from the topological organization of statute law. While existing works ignore these domain-specific dependencies, we propose a novel graph-augmented dense statute retriever (G-DSR) model that incorporates the structure of legislation via a graph neural network to improve dense retrieval performance. Experimental results show that our approach outperforms strong retrieval baselines on a real-world expert-annotated SAR dataset.[1]

1 Introduction

Today, the high cost of legal expertise prevents less fortunate people from understanding and reacting to legal issues that may arise (Ponce et al., 2019). In recent years, an increasing number of works have focused on legal text processing (Zhong et al., 2020) with the intent to assist legal practitioners and citizens while reducing legal costs and improving equal access to justice for all. Statutory article retrieval (SAR), the task of retrieving statute law articles relevant to a legal question, marks the first and one of the most crucial steps in any legal aid process. Our goal is to help reduce the gap between people and the law by improving SAR systems that could provide citizens with the first component of a free professional legal aid service.

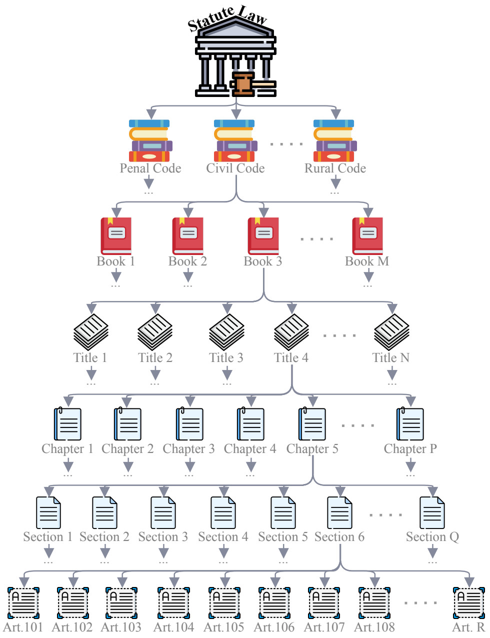


Figure 1: Illustration of the hierarchical organization of statute law. Each law code is structured into books, titles, chapters, and sections. The deeper the divisions, the closer the legal concepts of the articles below them.

Prior work has addressed SAR with standard information retrieval approaches such as term-based models or dense embedding-based models (Kim et al., 2019; Nguyen et al., 2021). While good performance has been achieved, these approaches rely on the flawed assumption that articles are complete and independent sources of information. In reality, statute law is an ensemble of *interdependent* rules meticulously organized into different codes, books, titles, chapters, and sections, as illustrated in Figure 1. Each level in the structure of legislation comes with a unique heading that informs about the content of the articles below it. An article takes on its whole meaning only when considered at its rightful place in the structure with the complementary information from its neighboring articles.

This work shows that such a structure can be highly beneficial for retrieving statutes. We propose a graph-augmented dense statute retriever (G-

---

[1] Our source code is available at `https://github.com/maastrichtlawtech/gdsr`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2761–2776
May 2-6, 2023

DSR) model that leverages the topological structure of legislation to enhance the article content information. Specifically, the proposed model extends the document encoder of a dense retriever with a graph neural network to learn knowledge-rich cross-article representations. Similar to previous work, we adopt a contrastive learning strategy to optimize the similarity between the representations of relevant query-article pairs.

The contributions of this paper are threefold:

- We propose a graph-augmented dense retriever model for statutory article retrieval that explicitly utilizes the topological organization of statute law to enrich the article information.
- We conduct empirical evaluations on our model and demonstrate improvements over strong retrieval baselines.
- We perform ablation studies on various model components and training strategies to understand the impact of several design options on the effectiveness of our model.

2 Preliminaries

In this section, we formally introduce the task of *statutory article retrieval* and discuss the specific difficulties associated with it. We then explain how we identify the structure of legislation as an essential consideration in SAR.

Problem formulation. Given a simple legal question, such as "*Who should pay for the construction of the common wall?*", SAR aims to return one or several relevant articles from the legislation. Formally speaking, a SAR system can be expressed as a function $R : (q, \mathcal{C}) \mapsto \mathcal{F}$ that takes as input a question q along with a corpus of articles $\mathcal{C} = \{a_1, a_2, \cdots, a_N\}$, and returns a much smaller filter set $\mathcal{F} \subset \mathcal{C}$ of the supposedly relevant articles, ranked by decreasing order of relevance. For a fixed $k = |\mathcal{F}| \ll |\mathcal{C}|$, the retriever can be evaluated in isolation with multiple rank-based metrics. Most modern retriever systems follow a two-stage retrieval approach (Guo et al., 2016; Hui et al., 2017; McDonald et al., 2018), where a *pre-fetcher* first aims to return all relevant documents in the filter set $\mathcal{F}$ and a *re-ranker* then attempts to make more relevant documents in $\mathcal{F}$ appear before less relevant ones. In this work, we focus our research on improving the pre-fetcher component for SAR.

Challenges. SAR comes with two core challenges that make the task unique compared to traditional information retrieval. First, the statutes to be retrieved are written in a language that dramatically differs from the ordinary plain language used in the questions. The legal language uses a specialized jargon known for its frequent and deliberate use of formal words, Latin phrases, lengthy sentences, and expressions with flexible meanings (Charrow and Crandall, 1990). Second, statutory articles are long text sequences that may reach several thousand words. This implies overcoming the maximum input length limit of 512 tokens imposed by BERT-based models, which have recently become the standard in neural information retrieval due to their effectiveness.

Structure of legislation. The legislation comes with a well-thought-out organization of its written rules to facilitate access to provisions covering a given subject (Onoge, 2015). This organization is established in a hierarchical manner, where higher-level divisions cover broad legal domains while lower-level divisions deal with specific legal concepts. To examine the importance of this hierarchy in the SAR process, we conduct a preliminary investigation in which we study the reasoning legal experts follow when performing the task. We summarize these experts' approach in Appendix A.1. We observe that legal experts rely heavily on the structure of law when retrieving relevant articles to a legal question, which indicates that the different divisions' headings in the legislation carry valuable information that retrieval systems should consider. Additionally, we analyze the degree to which neighboring articles cover related subjects in Appendix A.2 and find high levels of similarities, which suggests that information from neighboring articles should be considered to capture an article's whole meaning.

3 Approach

In this section, we present a new general approach for SAR that learns to retrieve relevant statutes by using both the textual semantic information from articles and the structural graph information from the legislation. Our model, called graph-augmented dense statute retriever (G-DSR), consists of two main building blocks, as depicted in Figure 2, that are trained independently with the same objective. We first describe the dense retriever component of our approach in Section 3.1 and then explain how our legislative graph encoder builds upon it in Section 3.2.

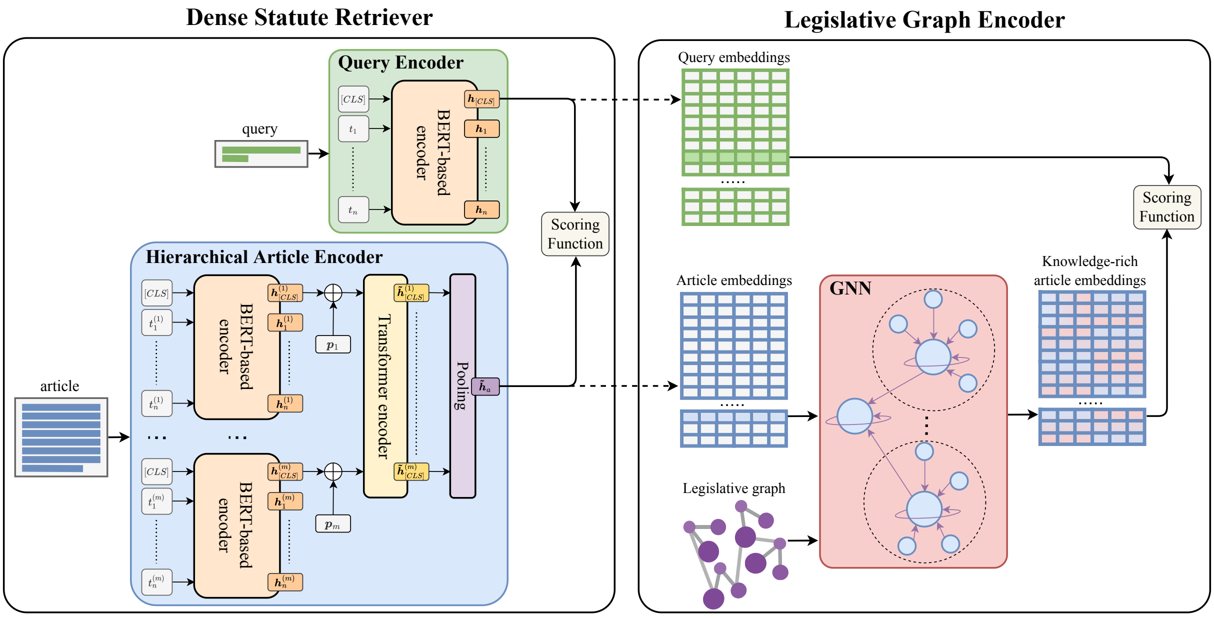


Figure 2: **An illustration of the graph-augmented dense statute retriever (G-DSR) model**. G-DSR consists of two main building blocks that are trained independently. **Left:** The dense statute retriever (DSR) first learns high-quality low-dimensional embedding spaces for both the queries and articles such that relevant query-article pairs appear closer than irrelevant ones in those vector spaces. **Right:** The legislative graph encoder (LGE) then learns to enrich the article representations by aggregating information from the organization of statute law.

3.1 Dense Statute Retriever

Our approach's first component, called dense statute retriever (DSR), aims to learn high-quality low-dimensional embedding spaces for questions and articles so that relevant question-article pairs appear closer than irrelevant ones in those spaces. Below, we review the overall architecture of the retriever and detail the design of its query and article encoders. We then describe the contrastive learning strategy we employ and choice of negative pairs.

Bi-encoder. We use the widely adopted bi-encoder architecture (Bromley et al., 1993) as the foundation of our dense retriever. The latter maps queries and articles into dense vector representations and calculates a relevance score $s : (q, a) \mapsto \mathbb{R}_+$ between query q and article a by the similarity of their embeddings, i.e.,

$$s(q, a) = \operatorname{sim}\left(E_Q^{\boldsymbol{\theta}}(q), E_A^{\boldsymbol{\phi}}(a)\right), \qquad (1)$$

where $E_Q^{\boldsymbol{\theta}}(q), E_A^{\boldsymbol{\phi}}(a) \in \mathbb{R}^d$ denote the query and article embeddings respectively, and $\operatorname{sim} : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ is a similarity function such as cosine or dot-product.

Query encoder. To encode the queries, we feed them into a BERT-based (Devlin et al., 2019) model $E_Q^{\boldsymbol{\theta}} : \mathcal{W}^n \mapsto \mathbb{R}^d$ with weights $\boldsymbol{\theta}$, that maps an input sequence of n tokens from vocabulary $\mathcal{W}$ to d-dimensional real-valued token embeddings. We take the last layer's [CLS] token representation as the query embedding, i.e.,

$$E_Q^{\boldsymbol{\theta}}(q) = \mathrm{BERT}_{[\mathrm{CLS}]}(q). \qquad (2)$$

Hierarchical article encoder. Since statutory articles may be longer than the maximum input length of a standard BERT-based encoder, we use a hierarchical variation that can process longer textual sequences (Pappagari et al., 2019; Zhang et al., 2019; Yang et al., 2020a). Each article a is first split into smaller text passages $[p_1, p_2, \cdots, p_m]$, where a passage p_i is a sequence of tokens $[t_1^{(i)}, t_2^{(i)}, \cdots, t_n^{(i)}]$ with $n \leq 512$. These passages are then independently passed through a shared BERT-based model to extract a list of *context-unaware* passage representations using the respective [CLS] token embeddings, as illustrated in Figure 2. Next, the hierarchical model sums the [CLS] token representations of each passage with learnable passage position embeddings and feeds the resulting representations into a small Transformer encoder to make them aware of the surrounding passages. The final article representation is computed through a pooling operation over the *context-aware* passage representations, i.e.,

$$E_A^{\phi}(a) = \text{pool}\left(\left[\tilde{\boldsymbol{h}}_{[\text{CLS}]}^{(1)}, \cdots, \tilde{\boldsymbol{h}}_{[\text{CLS}]}^{(m)}\right]\right), \quad (3)$$

where $\tilde{\boldsymbol{h}}_{[\text{CLS}]}^{(i)} \in \mathbb{R}^d$ is the contextualized embedding of passage p_i, and pool : $\mathbb{R}^{m \times d} \mapsto \mathbb{R}^d$ is either mean or max pooling.

Contrastive learning. The training objective of the bi-encoder is to learn effective embedding functions $E_Q^{\boldsymbol{\theta}}(\cdot)$ and $E_A^{\phi}(\cdot)$ such that relevant question-article pairs have a higher similarity than irrelevant ones. Let $\mathcal{D} = \{\langle q_i, a_i^+ \rangle\}_{i=1}^N$ be the training data where each of the N instances consists of a query q_i associated with a relevant article a_i^+. By sampling a set of negative articles $\mathcal{A}_i^-$ for each question q_i, we can create a training set $\mathcal{T} = \{\langle q_i, a_i^+, \mathcal{A}_i^- \rangle\}_{i=1}^N$. For each training instance in $\mathcal{T}$, we contrastively optimize the negative log-likelihood of the positive article against the negative ones, i.e.,

$$L\left(q_i, a_i^+, \mathcal{A}_i^-\right) = -\log \frac{e^{s(q_i, a_i^+)/\tau}}{\sum_{a \in \mathcal{A}_i^- \cup \{a_i^+\}} e^{s(q_i, a)/\tau}}, \quad (4)$$

where $\tau > 0$ is a temperature parameter to be set.

Negatives. We consider two types of negative examples: (i) in-batch (Chen et al., 2017; Henderson et al., 2017), i.e., articles paired with the other questions from the same mini-batch, and (ii) BM25, i.e., top articles returned by BM25 that are not relevant to the question.

3.2 Legislative Graph Encoder

Our approach's second component, called Legislative Graph Encoder (LGE), aims to enrich article representations given by the trained retriever's article encoder by fusing information from a legislative graph. Below, we elaborate on the legislative graph construction and the graph training process.

Graph construction. To leverage the hierarchical organization of statute law, we formalize the latter as a tree structure consisting of two types of node: (i) *section* nodes, which are titled structural units that represent the consecutive divisions in codes of law (i.e., the headings of the books, titles, chapters, and sections), and (ii) *article* nodes, which are textual content units that represent the different statutory articles. As illustrated in Figure 1, the edges represent the hierarchical connections between section and article nodes. Formally, such a tree can be represented as a directed acyclic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V}$ as the node set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ as the edge set.

Node feature initialization. Nodes in $\mathcal{V}$ are commonly associated with d-dimensional features. We apply the article encoder $E_A^{\phi}(\cdot)$ from the trained bi-encoder to encode the semantic information of nodes (i.e., section headings and article contents) offline and use the resulting embeddings as the initial node features $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$.

Node feature update. To fuse the information of node features using the graph structure, we use a graph neural network (GNN). Such a model consists of a stack of neural network layers, where each layer aggregates local neighborhood information (i.e., features of neighbors) around each node and then passes this aggregated information on to the next layer. Generally speaking, a GNN takes as inputs the feature matrix $\mathbf{X}$ and the graph's adjacency matrix $\mathbf{A} \in \mathbb{R}_+^{|\mathcal{V}| \times |\mathcal{V}|}$, with $\mathbf{A}_{i,j}$ as the edge weight between nodes i and j, and produces a node-level output $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times d}$ that captures each node's structural properties. Every GNN layer can be written as a non-linear function

$$\mathbf{H}^{(l+1)} = f(\mathbf{H}^{(l)}, \mathbf{A}), \quad (5)$$

with $\mathbf{H}^0 = \mathbf{X}$ and $\mathbf{H}^L = \mathbf{Z}$, L being the number of layers. In its simplest form, the layer-wise propagation rule is such that

$$f(\mathbf{H}^{(l)}, \mathbf{A}) = \sigma(\mathbf{A}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \quad (6)$$

where $\mathbf{W}^{(l)}$ is the input linear transformation's weight matrix for the l-th neural network layer and $\sigma(\cdot)$ is a non-linear activation function. We propose to use a 3-layer GATv2 network (Brody et al., 2022), a variant of GAT (Velickovic et al., 2018) that has the ability to learn the strength of connection between neighboring nodes through a *dynamic* attention mechanism. Formally, a GATv2 layer updates a node's hidden state as follows

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} \mathbf{W}^{(l)} \boldsymbol{h}_j^{(l)}\right), \quad (7)$$

where $\mathcal{N}(i)$ is the set of first-order neighbors of node i, and $\alpha_{ij}^{(l)}$ are normalized attention coefficients indicating the importance of node j's features to i in the l-th layer. The latter are computed based on the features of the connected nodes using an attention function att : $\mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ such that

$$\alpha_{ij}^{(l)} = \text{softmax}\left(\sigma(\text{att}(\boldsymbol{h}_i^{(l)}, \boldsymbol{h}_j^{(l)})\right). \quad (8)$$

Learning process. To optimize the GNN parameters, we adopt the same contrastive learning strategy used to train the bi-encoder. Since graph $\mathcal{G}$ can be relatively large, performing an update of all the node features in $\mathcal{G}$ at every training iteration would incur high computational costs. Besides, most of these computations would be of no use as only the updated representations of nodes $\{\mathcal{A}_i^- \cup a_i^+\}_{i=1}^{|\mathcal{B}|}$ from batch $\mathcal{B}$ are needed to update the model parameters. Therefore, we build a sub-graph $\mathcal{G}^{sub}$ at each training step that only contains the article nodes from batch $\mathcal{B}$ as well as their L-hop neighbors (where L is decided by the number of GNN layers). We then pass that sub-graph to the graph network and use the resulting article representations to compute the loss in Equation (4). Comparably to the node features, the query embeddings are pre-computed offline before training by the query encoder $E_Q^{\boldsymbol{\theta}}(\cdot)$ of our trained bi-encoder.

4 Experimental Setup

In this section, we present the basic setup for experiments. In particular, Section 4.1 describes the dataset we conduct our experiments on, Section 4.2 details our model implementation, Section 4.3 reviews the different baselines we use for comparison, and Section 4.4 reports the evaluation metrics.

4.1 Dataset

We conduct experiments on the publicly available Belgian Statutory Article Retrieval Dataset (Louis and Spanakis, 2022, BSARD).[2] To the best of our knowledge, BSARD is the only SAR dataset that provides the lists of consecutive division headings each article belongs to, which is crucial for building the graph of the legislative structure. The dataset consists of 1,100+ *French* native questions on various legal topics, as shown in Table 1, labeled by skilled experts with references to relevant statutory articles from the Belgian legislation. The retrieval corpus comprises 22,600+ articles collected from 32 Belgian codes covering numerous legal domains. The questions are relatively short and might have several relevant legal articles. We refer readers to the original paper for further data collection and analysis details.

4.2 Implementation Details

Model. We use the publicly released CamemBERT (Martin et al., 2020) checkpoint to initialize DSR's query encoder. Due to the specificity of the legal language the article encoder has to deal with, we follow prior work on domain adaptation (Gururangan et al., 2020; Jørgensen et al., 2021) and continue pre-training CamemBERT on BSARD statutory articles for 50k gradient steps to adapt it to the target legal domain. We use the resulting domain-specific checkpoint to warm-start the article's first-level encoder. The second-level encoder is a two-layer Transformer encoder of 14M parameters with a similar configuration (i.e., 768-hidden, 3072-intermediate, 12-heads, 0.1 dropout, GeLU). We use max-pooling to aggregate the final chunk representations and cosine as the decomposable similarity function.

Topic	Train	Dev	Test
👪 Family	216	56	67
🏠 Housing	203	38	66
💰 Money	103	35	36
⚖️ Justice	96	25	30
✈️ Foreigners	41	9	13
👵 Social security	27	8	6
🖨️ Work	23	6	4
Total	709	177	222

Table 1: Topic distribution of questions in BSARD.

Data augmentation. Due to the recent success in using synthetic query generation to improve dense retrieval performance (Liang et al., 2020; Ma et al., 2021; Thakur et al., 2021), we propose to augment BSARD with synthetic domain-targeted queries. We use a mT5 model (Raffel et al., 2020) fine-tuned on general domain data from mMARCO (Bonifacio et al., 2021) to synthesize queries for our target statutory articles.[3] We generate five queries per article, which results in a total of around 118k synthetic queries. We combine the latter with the gold BSARD train samples and obtain an augmented training set of around 122.5k question-article pairs.

Optimization. We train DSR for 15 epochs with a batch size of 24 using AdamW (Loshchilov and Hutter, 2017) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon =$1e-7, weight decay of 0.01, and learning rate warm up along the first 5% of the training steps to a maximum value of 2e-5, after which linear decay is applied. We then optimize LGE parameters for 10

---

[2] `https://huggingface.co/datasets/antoiloui/bsard`

[3] `https://huggingface.co/doc2query/msmarco-french-mt5-base-v1`

epochs with a batch size of 512 using AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon =$1e-7, weight decay of 0.1, and a constant learning rate of 2e-4. We use 16-bit automatic mixed precision to accelerate training and save memory. Details on our hyperparameter tuning process are given in Appendix B.

Hardware & schedule. Training is performed on a single 32 GB NVIDIA V100 GPU hosted on a server with a dual 20-core Intel Xeon E5-2698 v4 CPU @2.20GHz and 512 GB of RAM. It takes around 1 day to train DSR and 35 minutes for LGE.

Libraries. We implement, train, and tune our models using Transformers (Wolf et al., 2020), PyTorch (Paszke et al., 2019), PyTorch-Geometric (Fey and Lenssen, 2019), PyTorch-Lightning (Falcon, 2019), W&B Sweeps (Biewald, 2020), and DeepSpeed (Rasley et al., 2020).

4.3 Baselines

We compare our approach against three strong retrieval systems. As a sparse baseline model, we follow prior work and consider BM25 (Robertson et al., 1994),[4] a popular bag-of-words retrieval function based on exact term matching. We then examine the document expansion technique docT5query (Nogueira and Lin, 2019), which augments each article with a pre-defined number of synthetic queries generated by a finetuned mT5 model,[3] and then uses a traditional BM25 lexical index from the augmented articles for retrieval. Last, we include the results of a supervised dense passage retrievers (Karpukhin et al., 2020, DPR) pre-finetuned on more than 90.5k question-context pairs from a combination of three French QA datasets.[5]

4.4 Evaluation

We evaluate model performance using three commonly used ranking measures (Manning et al., 2008), namely the macro-averaged recall at different cutoffs (R@k), mean average precision (mAP), and mean r-precision (mRP). Those metrics are further defined in Appendix D. We deliberately omit to report the precision@k given that questions in BSARD have a variable number of relevant articles, which implies that questions with r relevant articles would always have P@$k < 1$ if $k > r$. Similarly, the mean reciprocal rank (mRR) is not appropriate for BSARD as only the first relevant article would be considered. As some questions might have up to 100 relevant articles, we use $k \in \{100, 200, 500\}$ for the recall@k.

5 Experiments

In this section, we empirically evaluate the effectiveness of our proposed approach against competitive baselines and discuss the main results in Section 5.1. Next, we provide an ablation study in Section 5.2 to understand how different design and training options affect our model's performance.

5.1 Main Results

Table 2 shows retrieval performance on BSARD test set. Although we report model performance on two rank-aware metrics (i.e., mAP and mRP), we emphasize that our approach is specifically aimed at improving the pre-fetching component of a retriever (Zhang et al., 2021a) and therefore focuses on optimizing rank-unaware metrics (i.e., R@k).

First, we compare the performance of our proposed G-DSR model(8) against other well-known retrieval approaches and find it significantly outperforms all of them on SAR. In particular, it improves over the sparse retrieval methods(1,2) by around 30% on recall@k and by more than 25% on mAP and mRP. It also outperforms a competitive pre-finetuned DPR model(4) by 6% on R@100, 9% on R@200, and 5% on R@500. However, the latter shows a better performance on rank-aware metrics compared to our DSR models, which we speculate might be due to its extensive pre-finetuning step on three domain-general retrieval datasets, leading the model to a deeper knowledge of the task at hand.

Next, we investigate the influence of different training strategies on the rank-unaware results of our base dense retriever.(5) We find that DSR's performance is improved when adapting the article text encoder to the legal domain before finetuning on the target data.(6) Besides, training DSR on a larger dataset containing synthetic domain-targeted queries improves its performance even more.(7)

Finally, our results show that using a GNN model on top of DSR allows to enrich the article representations and leads to the best overall performance.(8) Interestingly, G-DSR also significantly improves the rank-aware performance of our best performing DSR model by ∼12%, suggesting that a GNN could act as an effective re-ranker for SAR.

---

[4] We use $k_1 = 2.5$ and $b = 0.2$. Details on BM25 hyperparameters tuning are given in Appendix C.

[5] `https://huggingface.co/etalab-ia/dpr-question_encoder-fr_qa-camembert`

	Model	#Params	R@100 (↑)	R@200 (↑)	R@500 (↑)	mAP (↑)	mRP (↑)
Baselines							
1	BM25	-	49.3	57.3	63.0	16.8	13.6
2	docT5query	-	51.7	59.4	65.8	18.7	15.0
4	DPR	220M	77.9	81.3	88.2	45.4	39.1
Ours							
5	DSR	234M	77.1	81.8	86.7	35.6	28.8
6	DSR w. domain-adaptive pre-training	234M	79.8	83.9	88.9	39.5	31.3
7	DSR w. data augmentation	234M	82.7	88.7	92.8	35.3	27.5
8	G-DSR	262M	**84.3**	**90.4**	**93.1**	**47.1**	**40.2**

Table 2: Retrieval performance on BSARD Test set. The best results are marked in bold.

5.2 Ablation Study

To further understand how different design choices and training strategies affect the results, we conduct several additional experiments and discuss our findings below.

Alternative pre-trained LMs. In addition to CamemBERT, we experiment with several other French or multilingual pre-trained language models to initialize the first-level text encoders in DSR – namely, mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020), and ELECTRA-fr (Clark et al., 2020).[6] We fine-tune the different warm-started models on BSARD training set and report dev results in Table 3. We find that a CamemBERT-initialized DSR model performs best.

Alternative GNNs. Additionally to GATv2, we explore different GNN architectures to perform the node feature update – namely, GCN (Kipf and Welling, 2017), GraphSAGE (Hamilton et al., 2017), GAT (Velickovic et al., 2018), and k-GNN (Morris et al., 2019) – and summarize the results in Table 4. Our experiments show that using an alternative GNN model does not affect performance much, which suggests that the act of fusing information from neighboring nodes is more important than the way the aggregation is performed.

Similarity and loss functions. Besides cosine for scoring pairs of query-article representations, we also experiment with dot-product and Euclidean distance and find both inferior to cosine. As an alternative to negative log-likelihood, we test the triplet loss (Burges et al., 2005) and observe that the latter significantly decreases model performance. More details can be found in Appendix E.

Pre-trained LM	#Params	R@100 (↑)	R@500 (↑)
XLM-R	278M	59.8	78.7
mBERT	177M	69.2	86.6
ELECTRA-fr	110M	57.6	73.7
CamemBERT	110M	**75.4**	**88.5**

Table 3: BSARD Dev results of DSR warm-started with different pre-trained word embedding models.

Model	#Params	R@100 (↑)	R@500 (↑)
GCN	14M	84.4	92.0
GAT	14M	84.4	92.1
GraphSAGE	28M	84.5	92.9
k-GNN	28M	83.8	**93.0**
GATv2	28M	**84.8**	92.3

Table 4: BSARD Dev results of LGE with different (3-layer) GNN architectures.

6 Related Work

Our work operates at the intersection of several research areas, including long document modeling, dense information retrieval, graph neural networks, and legal NLP.

Long document modeling. The emergence of deep neural networks for language processing brought new challenges to text encoding, one of which is learning high-quality representations of long documents. For example, Tang et al. (2015) employ a bottom-up approach using CNN and BiLSTM-based hierarchical networks, where sentences are first encoded into vectors, which are then combined to form a single document vector. Similarly, Yang et al. (2016) build a document vector by aggregating important words into sentence vectors and then aggregating important sentence vectors to document vectors using attention mech-

---

[6] `https://huggingface.co/dbmdz/electra-base-french-europeana-cased-discriminator`

anisms. More recently, hierarchical variants of Transformer-based models have been explored for various language tasks, including document classification (Mulyar et al., 2019; Pappagari et al., 2019; Chalkidis et al., 2019; Wu et al., 2021), summarization (Zhang et al., 2019), semantic matching (Yang et al., 2020a), and question answering (Liu et al., 2022). In addition to hierarchical attention Transformer-based (HAT) models, several sparse attention Transformers (SAT) have been introduced to reduce the computational complexity of the model, thus allowing to process sequences longer than 512 tokens (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020). However, Chalkidis et al. (2022) show that a pre-trained HAT model performs comparably or better than an equally-sized SAT model across several downstream tasks while being substantially faster and less memory intensive. Recently, other non-Transformer-based approaches have been proposed for efficient long sequence processing based on structured state spaces (Gu et al., 2022; Gupta et al., 2022).

Dense information retrieval. Traditionally, lexical approaches such as TF-IDF and BM25 (Robertson et al., 1994) have been the *de facto* standard for textual information retrieval due to their robustness and efficiency. However, these approaches suffer from the lexical gap problem (Berger et al., 2000) and can only retrieve documents containing keywords present in the query. To overcome this limitation, recent work relies on neural-based architectures to capture semantic relationships between pairs of texts (Lee et al., 2019; Karpukhin et al., 2020; Yang et al., 2020b; Xiong et al., 2021). These models map queries and documents into dense vector representations and calculate a relevance score by the similarity of the vectors (Gillick et al., 2018), which allows the document representations to be pre-computed and indexed offline for inference. The dense retrieval approach was recently extended by hybrid lexical-dense methods, which aim to combine the strengths of both approaches (Seo et al., 2019; Gao et al., 2021; Luan et al., 2021). We refer the readers to Yates et al. (2021) for a survey on neural information retrieval.

Graph neural networks. Graph neural networks (GNNs) capture the topological relationships among the nodes of a graph using an information diffusion mechanism that propagates node features according to the underlying graph-structured data (Scarselli et al., 2009). These models have shown their effectiveness and flexibility in a wide variety of NLP tasks, including text classification (Lin et al., 2021; Yu et al., 2022), relation extraction (Zhang et al., 2018; Li et al., 2020; Carbonell et al., 2020), and question answering (Cao et al., 2019; Xu et al., 2021b). Recently, GNNs have been employed for document retrieval to enhance the vector representations by leveraging the topological structure of the documents, where nodes are passages from a document and edges are relations between these passages (Xu et al., 2021a; Zhang et al., 2021b; Albarede et al., 2022).

Application to the legal domain. In recent years, the legal domain has attracted much interest in the NLP community, both for its challenging characteristics and massive volumes of textual data (Chalkidis and Kampas, 2019; Zhong et al., 2020). Researchers see it as an opportunity to develop novel automated methodologies that can reduce heavy and redundant tasks for legal professionals while providing a reliable, affordable form of legal support for laypeople (Bommasani et al., 2021). Earlier techniques for legal information retrieval were mainly based on term-matching approaches (Kim and Goebel, 2017; Tran et al., 2018). Recently, a growing number of works have used neural networks to enhance retrieval performance, including word embedding models (Landthaler et al., 2016), doc2vec models (Sugathadasa et al., 2018), CNN-based models (Tran et al., 2019), and BERT-based models (Nguyen et al., 2021; Chalkidis et al., 2021; Althammer et al., 2022). To the best of our knowledge, we are the first to exploit the structure of statute law with GNNs to improve the performance of dense retrieval models.

7 Conclusion

In this paper, we introduce G-DSR, a novel approach for statutory article retrieval (SAR) that leverages the topological structure of legislation to improve retrieval performance. Specifically, G-DSR enriches the article representations of a dense retriever designed for long document retrieval by employing a graph neural network that uses the organization of statute law to learn knowledge-rich cross-article embeddings. Experiments show that G-DSR outperforms competitive baselines on a real-world expert-annotated SAR dataset. We also include a detailed analysis to motivate our design choices and training strategies.

Limitations

While our approach performs well on statutory article retrieval, it comes with several limitations that provide avenues for future work.

First, experimental results are based on questions and labels drafted by legal professionals. It is possible that other legal professionals would draft the questions differently or, less likely yet possible, that they would deem different statutory provisions relevant. This raises the question of to what extent similar results would be obtained if the model were trained on a different dataset, for instance, based on other experts or domains, hence testing the approach's generalizability. The main challenge in this regard is obtaining data, as organizations are unlikely to share or even collect similar data.

Second, our proposed methodology was evaluated exclusively on the Belgian legislation, whose laws are organized in a hierarchical manner where the deeper the divisions, the more closely related the legal concepts of the articles under them. Although we believe our approach could be applied to most, if not all, jurisdictions that rely on statute law (including both civil and common law countries), different jurisdictions may have different organizations of their legal provisions, which could potentially affect the model's performance. It is also worth mentioning that the dataset used for evaluation comes with a linguistic bias as Belgium is a multilingual country with French, Dutch, and German speakers, but the provided provisions are only available in French. Studying the applicability and impact of the present work to other jurisdictions and languages is an exciting research direction that is challenging in practice due to the scarcity of high-quality multilingual statute retrieval datasets.

Then, our approach currently considers the topological structure of legislation for modeling the inter-article dependencies, which implies that information is aggregated between direct neighboring articles only while those from more distant sections are completely ignored. Nevertheless, it is common for articles to cite other articles from different sections or even different statutes. Therefore, we believe that considering richer legal graph structures, especially legal citation networks, could increase effectiveness even more. However, building such citation networks from raw texts requires a considerable text-processing effort.

Finally, although G-DSR shows promise for statutory article retrieval, it is not yet ready for practical use in the real world. One issue is that our model is designed to be an effective pre-fetcher, optimizing recall such that all articles relevant to a question appear in an unordered filter set of size k (k being relatively large). However, in practice, users would expect a high-quality retrieval system to not only find these relevant articles but also to sort them by decreasing order of importance, requiring an adequate re-ranker. Then, it is essential to recognize that while access to relevant legal provisions is a necessary step in helping the general public solve their legal issues, it is not a sufficient condition on its own as laypeople may still struggle to understand the legal jargon and apply the provisions to their specific situations. Ideally, the tool to be made accessible to the public should consist of a two-stage framework: (i) a legal provision retriever, which selects a small subset of relevant legal articles in response to a given question, and (ii) a legal-to-natural translator or summarizer, which examines the retrieved articles and generates an answer in natural language. In the present work, we chose to focus on the first stage of this framework and leave the second for future work.

Ethics Statement

The scope of this work is to provide a new methodology along with extensive experiments to drive research forward in statutory article retrieval. We believe the latter is an important application field where more research should be conducted to improve legal aid services and access to justice for all. We do not foresee situations where the use of our methodology would lead to harm (Tsarapatsanis and Aletras, 2021). Nevertheless, although our goal is to improve the understanding of the law by those who suffer from legal information asymmetry, it cannot be excluded that the technology presented here could exacerbate inequality if states, companies, or lawyers benefit more from its use than the intended beneficiaries (i.e., citizens, consumers, or employees).

Acknowledgments

This research is partially supported by the Sector Plan Digital Legal Studies of the Dutch Ministry of Education, Culture, and Science. In addition, this research was made possible, in part, using the Data Science Research Infrastructure (DSRI) hosted at Maastricht University.

References

Lucas Albarede, Philippe Mulhem, Lorraine Goeuriot, Claude Le Pape-Gardeux, Sylvain Marie, and Trinidad Chardin-Segui. 2022. Passage retrieval on structured documents using graph attention networks. In *Proceedings of the 44th European Conference on Information Retrieval Research*, volume 13186 of *Lecture Notes in Computer Science*, pages 13–21. Springer. [Page 8]

Sophia Althammer, Sebastian Hofstätter, Mete Sertkan, Suzan Verberne, and Allan Hanbury. 2022. PARM: A paragraph aggregation retrieval model for dense document-to-document retrieval. In *Proceedings of the 44th European Conference on Information Retrieval Research*, volume 13185 of *Lecture Notes in Computer Science*, pages 19–34. Springer. [Page 8]

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150. [Page 8]

Adam L. Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu O. Mittal. 2000. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–199. ACM. [Page 8]

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com. [Page 6]

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. 2021. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258. [Page 8]

Luiz Henrique Bonifacio, Israel Campiotti, Roberto de Alencar Lotufo, and Rodrigo Nogueira. 2021. mmarco: A multilingual version of MS MARCO passage ranking dataset. *CoRR*, abs/2108.13897. [Page 5]

Shaked Brody, Uri Alon, and Eran Yahav. 2022. How attentive are graph attention networks? In *Proceedings of the 10th International Conference on Learning Representations*. OpenReview.net. [Page 4]

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a siamese time delay neural network. *Advances in Neural Information Processing Systems*, 6:737–744. [Page 3]

Christopher J. C. Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Gregory N. Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, volume 119 of *ACM International Conference Proceeding Series*, pages 89–96. ACM. [Page 7]

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2306–2317. Association for Computational Linguistics. [Page 8]

Manuel Carbonell, Pau Riba, Mauricio Villegas, Alicia Fornés, and Josep Lladós. 2020. Named entity recognition and relation extraction with graph neural networks in semi structured documents. In *Proceedings of the 25th International Conference on Pattern Recognition*, pages 9622–9627. IEEE. [Page 8]

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in english. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 4317–4323. Association for Computational Linguistics. [Page 8]

Ilias Chalkidis, Xiang Dai, Manos Fergadiotis, Prodromos Malakasiotis, and Desmond Elliott. 2022. An exploration of hierarchical attention transformers for efficient long document classification. *CoRR*, abs/2210.05529. [Page 8]

Ilias Chalkidis, Manos Fergadiotis, Nikolaos Manginas, Eva Katakalou, and Prodromos Malakasiotis. 2021. Regulatory compliance through doc2doc information retrieval: A case study in EU/UK legislation where text similarity has limitations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3498–3511. Association for Computational Linguistics. [Pages 8 and 15]

Ilias Chalkidis and Dimitrios Kampas. 2019. Deep learning in law: early adaptation and legal word embeddings trained on large corpora. *Artificial Intelligence and Law*, 27(2):171–198. [Page 8]

Veda R Charrow and Jo Ann Crandall. 1990. Legal language: What is it and what can we do about it?. [Page 2]

Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on*

Knowledge Discovery and Data Mining, pages 767–776. [Page 4]

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509. [Page 8]

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555. [Page 7]

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451. Association for Computational Linguistics. [Page 7]

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186. Association for Computational Linguistics. [Pages 3 and 7]

William Falcon. 2019. PyTorch Lightning. [Page 6]

Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*. [Page 6]

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021. Complement lexical retrieval model with semantic residual embeddings. In *Proceedings of the 43rd European Conference on Information Retrieval Research*, volume 12656 of *Lecture Notes in Computer Science*, pages 146–160. Springer. [Page 8]

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *CoRR*, abs/1811.08008. [Page 8]

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *Proceedings of the 10th International Conference on Learning Representations*. OpenReview.net. [Page 8]

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*, pages 55–64. ACM. [Page 2]

Ankit Gupta, Albert Gu, and Jonathan Berant. 2022. Diagonal state spaces are as effective as structured state spaces. *CoRR*, abs/2203.14343. [Page 8]

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360. Association for Computational Linguistics. [Page 5]

William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30:1024–1034. [Page 7]

Matthew L. Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *CoRR*, abs/1705.00652. [Page 4]

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1049–1058. Association for Computational Linguistics. [Page 2]

Rasmus Kær Jørgensen, Mareike Hartmann, Xiang Dai, and Desmond Elliott. 2021. mdapt: Multilingual domain adaptive pretraining in a single model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3404–3418. Association for Computational Linguistics. [Page 5]

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6769–6781. Association for Computational Linguistics. [Pages 6 and 8]

Mi-Young Kim and Randy Goebel. 2017. Two-step cascaded textual entailment for legal bar exam question answering. In *Proceedings of the 16th edition of the International Conference on Artificial Intelligence and Law*, pages 283–290. ACM. [Page 8]

Mi-Young Kim, Juliano Rabelo, and Randy Goebel. 2019. Statute law information retrieval and entailment. In *Proceedings of the 6th Competition on Legal Information Retrieval and Entailment Workshop in association with the Seventeenth International Conference on Artificial Intelligence and Law*, pages 283–289. ACM. [Page 1]

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*. OpenReview.net. [Page 7]

Jörg Landthaler, Bernhard Waltl, Patrick Holl, and Florian Matthes. 2016. Extending full text search for legal document collections using word embeddings.

In *Proceedings of the 29th International Conference on Legal Knowledge and Information Systems*, volume 294 of *Frontiers in Artificial Intelligence and Applications*, pages 73–82. IOS Press. [Page 8]

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 6086–6096. Association for Computational Linguistics. [Page 8]

Bo Li, Wei Ye, Zhonghao Sheng, Rui Xie, Xiangyu Xi, and Shikun Zhang. 2020. Graph enhanced dual attention network for document-level relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1551–1560. International Committee on Computational Linguistics. [Page 8]

Davis Liang, Peng Xu, Siamak Shakeri, Cícero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. *CoRR*, abs/2009.10270. [Page 5]

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. Bertgcn: Transductive text classification by combining GCN and BERT. *CoRR*, abs/2105.05727. [Page 8]

Yang Liu, Jiaxiang Liu, Li Chen, Yuxiang Lu, Shikun Feng, Zhida Feng, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2022. ERNIE-SPARSE: learning hierarchical efficient transformer through regularized self-attention. *CoRR*, abs/2203.12276. [Page 8]

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations*. [Page 5]

Antoine Louis and Gerasimos Spanakis. 2022. A statutory article retrieval dataset in french. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6789–6803. Association for Computational Linguistics. [Pages 5, 14, and 15]

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345. [Page 8]

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith B. Hall, and Ryan T. McDonald. 2021. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1075–1088. Association for Computational Linguistics. [Page 5]

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge University Press. [Page 6]

Louis Martin, Benjamin Müller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219. Association for Computational Linguistics. [Page 5]

Ryan T. McDonald, George Brokos, and Ion Androutsopoulos. 2018. Deep relevance ranking using enhanced document-query interactions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1849–1860. Association for Computational Linguistics. [Page 2]

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pages 4602–4609. AAAI Press. [Page 7]

Andriy Mulyar, Elliot Schumacher, Masoud Rouhizadeh, and Mark Dredze. 2019. Phenotyping of clinical notes with improved document classification models using contextualized neural language models. *CoRR*, abs/1910.13664. [Page 8]

Ha-Thanh Nguyen, Phuong Minh Nguyen, Thi-Hai-Yen Vuong, Quan Minh Bui, Chau Minh Nguyen, Tran Binh Dang, Vu Tran, Minh Le Nguyen, and Ken Satoh. 2021. JNLP team: Deep learning approaches for legal processing tasks in COLIEE 2021. *CoRR*, abs/2106.13405. [Pages 1 and 8]

Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to doctttttquery. [Page 6]

Elohor Onoge. 2015. Structure of legislation: A paradigm for accessibility and effectiveness. *European Journal of Law Reform*, 17:440–470. [Page 2]

Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. 2019. Hierarchical transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop*, pages 838–844. IEEE. [Pages 3 and 8]

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8024–8035. [Page 6]

Alejandro Ponce, Sarah Chamness Long, Elizabeth Andersen, Camilo Gutierrez Patino, Matthew Harman, Jorge A Morales, Ted Piccone, Natalia Rodriguez Cajamarca, Adriana Stephan, Kirssy Gonzalez, Jennifer VanRiper, Alicia Evangelides, Rachel Martin, Priya Khosla, Lindsey Bock, Erin Campbell, Emily Gray, Amy Gryskiewicz, Ayyub Ibrahim, Leslie Solis, Gabriel Hearn-Desautels, and Francesca Tinucci. 2019. *Global Insights on Access to Justice 2019: Findings from the World Justice Project General Population Poll in 101 Countries*. World Justice Project. [Page 1]

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67. [Page 5]

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3505–3506. [Page 6]

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of the 3rd Text REtrieval Conference*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST). [Pages 6 and 8]

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80. [Page 8]

Min Joon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur P. Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 4430–4441. Association for Computational Linguistics. [Page 8]

Keet Sugathadasa, Buddhi Ayesha, Nisansa de Silva, Amal Shehan Perera, Vindula Jayawardana, Dimuthu Lakmal, and Madhavi Perera. 2018. Legal document retrieval using document vector embeddings and deep learning. In *Science and information conference*, pages 160–175. Springer. [Page 8]

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432. The Association for Computational Linguistics. [Page 7]

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *CoRR*, abs/2104.08663. [Page 5]

Vu Tran, Son Truong Nguyen, and Minh Le Nguyen. 2018. JNLP group: legal information retrieval with summary and logical structure analysis. In *12th International Workshop on Juris-informatics*. [Page 8]

Vu D. Tran, Minh Le Nguyen, and Ken Satoh. 2019. Building legal case retrieval systems with lexical matching and summarization using A pre-trained phrase scoring model. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, pages 275–282. ACM. [Page 8]

Dimitrios Tsarapatsanis and Nikolaos Aletras. 2021. On the ethical limits of natural language processing on legal text. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 3590–3599. Association for Computational Linguistics. [Page 9]

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations*. OpenReview. [Pages 4 and 7]

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics. [Page 6]

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Hi-transformer: Hierarchical interactive transformer for efficient and effective long document modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 848–853. Association for Computational Linguistics. [Page 8]

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the 9th International Conference on Learning Representations*. OpenReview. [Page 8]

Peng Xu, Xinchi Chen, Xiaofei Ma, Zhiheng Huang, and Bing Xiang. 2021a. Contrastive document representation learning with graph attention networks. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3874–3884. Association for Computational Linguistics. [Page 8]

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. 2021b. Fusing context into knowledge graph for commonsense question answering. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 1201–1207. Association for Computational Linguistics. [Page 8]

Liu Yang, Mingyang Zhang, Cheng Li, Michael Bendersky, and Marc Najork. 2020a. Beyond 512 tokens: Siamese multi-depth transformer-based hierarchical encoder for long-form document matching. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pages 1725–1734. ACM. [Pages 3 and 8]

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2020b. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94. Association for Computational Linguistics. [Page 8]

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1480–1489. The Association for Computational Linguistics. [Page 7]

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2666–2668. ACM. [Page 8]

Donghan Yu, Chenguang Zhu, Yiming Yang, and Michael Zeng. 2022. JAKET: joint pre-training of knowledge graph and language understanding. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, pages 11630–11638. AAAI Press. [Page 8]

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297. [Page 8]

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 5059–5069. Association for Computational Linguistics. [Pages 3 and 8]

Yue Zhang, Chengcheng Hu, Yuqi Liu, Hui Fang, and Jimmy Lin. 2021a. Learning to rank in the age of muppets: Effectiveness-efficiency tradeoffs in multi-stage ranking. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 64–73. Association for Computational Linguistics. [Page 6]

Yufeng Zhang, Jinghao Zhang, Zeyu Cui, Shu Wu, and Liang Wang. 2021b. A graph-based relevance matching model for ad-hoc retrieval. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 4688–4696. AAAI Press. [Page 8]

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215. Association for Computational Linguistics. [Page 8]

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. How does NLP benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5218–5230. Association for Computational Linguistics. [Pages 1 and 8]

Appendix

A Preliminary Studies

A.1 How important is the structure of law for statutory article retrieval?

To better understand the reasoning skilled humans follow when retrieving statutory articles, we ask several legal experts to retrieve relevant articles to questions sampled from the Belgian Statutory Article Retrieval Dataset (Louis and Spanakis, 2022, BSARD). We deliberately choose experts unfamiliar with Belgian law and thus have no past knowledge of the location of articles covering a particular subject. In what follows, we summarize the approach these experts use.

First, they determine whether the issue involves either *public* or *private* law. To distinguish between the two, it is necessary to identify to whom the rules apply (i.e., the parties involved in the issue that hold the rights or duties). Generally speaking, public law deals with issues that affect the general public or state (i.e., society as a whole), whereas private law deals with issues that affect individuals, families, and businesses. This first step allows the experts to make an initial selection among the codes of law, which generally relate to only one of either public or private law.

Next, the experts refine their search by determining the field of law (e.g., contract law), followed by the sub-field (e.g., tenant law), and so on, until a set of potentially relevant codes is created in the question's domains. The experts then focus on the table of contents of one of the selected codes and undertake a hierarchical search that starts from the book's headings and progressively extends to its titles, chapters, and sections. This step makes it possible to filter out many irrelevant articles by analyzing the connection between the question's subject and the different sections' headings.

Finally, the experts explore the articles within the sections deemed potentially relevant to the question in search of the expected answer. If the experts realize that the chosen direction is a dead end, they return to the previous higher level of the structure, choose another potentially relevant direction, and narrow their search from there.

From this study, we conclude that legal experts rely heavily on the structure of law when retrieving relevant articles to a legal question, which indicates that the different divisions' headings carry valuable information that retrieval systems should consider.

A.2 How related are neighboring articles in statute law?

In statute law, the sense of a given article is not necessarily self-contained by itself but instead spans across different articles from the same or even different sections. To confirm this, we study to what extent consecutive articles (as they appear in the statute books) address similar subjects.

We consider the Belgian Civil Code, which is the book whose articles are most cited in BSARD, and randomly sample sets of 200 consecutive articles out of it. We then normalize the articles by lowercasing, lemmatizing, and removing stop-words, punctuation, and numbers. Finally, we compute the cosine similarities between the TF-IDF representations of all articles from a given set. Figure 3 shows a heatmap of article similarities for such a set. We see that consecutive articles do indeed cover similar topics, suggesting that the information in a given article is likely to be complementary to that in its neighboring articles. Therefore, we assume that neighboring articles should be considered to capture an article's whole meaning.

B G-DSR Hyperparameter Tuning

We conduct hyperparameter tuning using Bayes search based on performance on BSARD development set, measured with the macro-averaged R@200. Due to limited computational resources, we train our models on BSARD training set only – which takes approximately 1 hour and 15 minutes for DSR and around 5 minutes for LGE – and use the constrained search spaces described below.

DSR grid search space:

- batch size: {8, 16, 24, 32}
- learning rate: {5e-5, 4e-5, 3e-5, 2e-5, 1e-5}
- weight decay: {0, 0.1, 0.01, 0.001}
- max chunk length: {64, 128, 256, 512}
- max document length: {1024, 2048}
- pooling strategy: {mean, max}
- similarity function: {dot-product, cosine, L2}
- temperature: {0, 0.1, 0.01, 0.001}

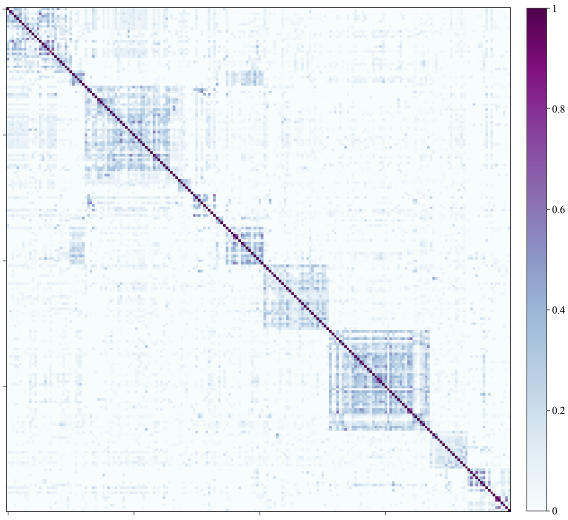


Figure 3: Cosine similarities between TF-IDF representations of 200 consecutive articles from the Belgian Civil Code, taken from the Belgian Statutory Article Retrieval Dataset (Louis and Spanakis, 2022, BSARD).

LGE grid search space:

- batch size: {8, 16, 32, 64, 128, 256, 512}
- learning rate: {2e-2, 2e-3, 2e-4, 2e-5, 2e-6}
- weight decay: {0, 0.1, 0.01, 0.001}
- #layers: {1, 2, 3, 4}

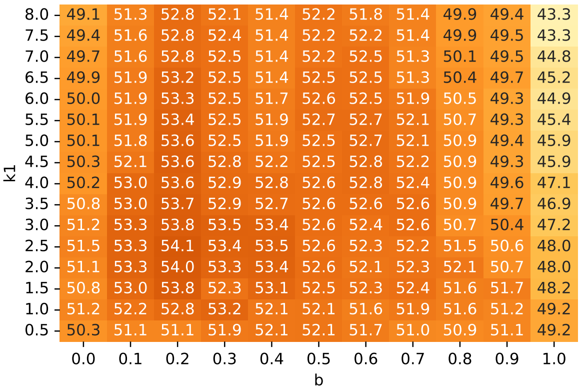
In total, we run 100 hyperparameter search trials for both DSR and LGE. The optimal hyperparameters, shown in Table 5, are used to re-train the models combining both train and development sets for a final evaluation on the test set.

C BM25 Hyperparameter Tuning

Following Chalkidis et al. (2021), who show that BM25 performance is highly dependent on adequately choosing the (k_1, b) values for the task at hand, we perform a hyperparameter grid search on BSARD development set and plot the results in Figure 4. We observe that, in the case of SAR, the best performance is obtained with $k_1 = 2.5$ and $b = 0.2$. Therefore, we use these values for the final evaluation on BSARD test set.

Hyperparam	DSR	LGE
Model		
Maximum Chunk Length	128	-
Maximum Document Length	1024	-
Pooling Strategy	max	-
#Layers	-	3
Loss		
Temperature	0.01	0.01
Similarity	cos	cos
Training		
Batch Size	24	512
Weight Decay	0.01	0.1
Max Epochs	15	10
Peak Learning Rate	2e-5	2e-4
Learning Rate Decay	Linear	Constant
Warmup ratio	0.05	0.0
AdamW ϵ	1e-7	1e-7
AdamW β_1	0.9	0.9
AdamW β_2	0.999	0.999
Gradient Clipping	1.0	1.0

Table 5: Training hyperparameters for DSR and LGE.



	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
8.0	49.1	51.3	52.8	52.1	51.4	52.2	51.8	51.4	49.9	49.4	43.3
7.5	49.4	51.6	52.8	52.4	51.4	52.2	52.2	51.4	49.9	49.5	43.3
7.0	49.7	51.6	52.8	52.5	51.4	52.2	52.5	51.3	50.1	49.5	44.8
6.5	49.9	51.9	53.2	52.5	51.4	52.5	52.5	51.3	50.4	49.7	45.2
6.0	50.0	51.9	53.3	52.5	51.7	52.6	52.5	51.9	50.5	49.3	44.9
5.5	50.1	51.9	53.4	52.5	51.9	52.7	52.7	52.1	50.7	49.3	45.4
5.0	50.1	51.8	53.6	52.5	51.9	52.5	52.7	52.1	50.9	49.4	45.9
4.5	50.3	52.1	53.6	52.8	52.2	52.5	52.8	52.2	50.9	49.3	45.9
4.0	50.2	53.0	53.6	52.9	52.8	52.6	52.8	52.4	50.9	49.6	47.1
3.5	50.8	53.0	53.7	52.9	52.7	52.6	52.6	52.6	50.9	49.7	46.9
3.0	51.2	53.3	53.8	53.5	53.4	52.6	52.4	52.6	50.7	50.4	47.2
2.5	51.5	53.3	54.1	53.4	53.5	52.6	52.3	52.2	51.5	50.6	48.0
2.0	51.1	53.3	54.0	53.3	53.4	52.6	52.1	52.3	52.1	50.7	48.0
1.5	50.8	53.0	53.8	52.3	53.1	52.5	52.3	52.4	51.6	51.7	48.2
1.0	51.2	52.2	52.8	53.2	52.1	52.1	51.6	51.9	51.6	51.2	49.2
0.5	50.3	51.1	51.1	51.9	52.1	52.1	51.7	51.0	50.9	51.1	49.2

Figure 4: Heatmap showing BM25 results on BSARD Dev set for different values of k_1 and b.

Loss	Similarity	R@100 (↑)	R@200 (↑)	R@500 (↑)
Cross-entropy	Cosine	**75.0**	**80.7**	**86.7**
	Dot	21.6	37.1	56.0
	Euclidean	43.9	58.1	71.6
Triplet	Cosine	5.4	9.9	17.2
	Dot	4.3	6.4	10.3
	Euclidean	9.0	14.3	25.2

Table 6: BSARD Dev results of DSR trained using different similarity and loss functions.

D Evaluation Metrics

Let $\text{rel}_q(a) \in \{0,1\}$ be the binary relevance label of article a for question q, and $\langle i,a\rangle \in \mathcal{F}_q$ a result tuple (article a at rank i) from the filter set $\mathcal{F}_q \subset \mathcal{C}$ of ranked articles retrieved for question q.

Recall. The *recall* is the fraction of relevant articles retrieved for query q w.r.t. the total number of relevant articles in the corpus $\mathcal{C}$, i.e.,

$$\text{R}_q = \frac{\sum_{\langle i,a\rangle \in \mathcal{F}_q} \text{rel}_q(a)}{\sum_{a\in\mathcal{C}} \text{rel}_q(a)}. \tag{9}$$

When computed for a filter set of size $k = |\mathcal{F}_q| \ll |\mathcal{C}|$, i.e., at a certain cutoff and not on the entire list of articles in $\mathcal{C}$, we report the metrics with the suffix "@k".

R-Precision The R-Precision is the proportion of the top-R retrieved articles that are relevant to query q, where R is the total number of relevant articles for q, i.e.,

$$\text{RP}_q = \frac{\sum_{\langle i,a\rangle \in \{\mathcal{F}_q\}_{i=1}^{R}} \text{rel}_q(a)}{R}. \tag{10}$$

Average Precision. The *average precision* is the mean of the precision value obtained after each relevant article is retrieved, that is

$$\text{AP}_q = \frac{\sum_{\langle i,a\rangle \in \mathcal{F}_q} \text{P}_{q,i} \times \text{rel}_q(a)}{\sum_{a\in\mathcal{C}} \text{rel}_q(a)}, \tag{11}$$

where $\text{P}_{q,j}$ is the *precision* computed at rank j for query q, i.e., the fraction of relevant articles retrieved for query q w.r.t. the total number of articles in the retrieved set $\{\mathcal{F}_q\}_{i=1}^{j}$:

$$\text{P}_{q,j} = \frac{\sum_{\langle i,a\rangle \in \{\mathcal{F}_q\}_{i=1}^{j}} \text{rel}_q(a)}{\left|\{\mathcal{F}_q\}_{i=1}^{j}\right|}. \tag{12}$$

We report the macro-averaged *recall* at various cutoffs (R@k), *mean Average Precision* (mAP), and *mean R-Precision* (mRP), which are the average values over a set of n queries.

E Ablation Details

Besides cosine similarity and negative log-likelihood (NLL) loss, we also test the dot-product and Euclidean (inverse of distance is taken as similarity measure) as well as the triplet loss. The temperature for the NLL loss is set to 0.01, and the margin value of the triplet loss is set to 1. We report the results on BSARD development set in Table 6. For a fair comparison, all models are trained for 15 epochs with a batch size of 24, weight decay of 0.01, warm-up proportion of 0.05, an initial learning rate of 2e-5, and a linear decay learning rate schedule.

Behavior Cloned Transformers are Neurosymbolic Reasoners

Ruoyao Wang[‡], **Peter Jansen**[‡], **Marc-Alexandre Côté**[♣], **Prithviraj Ammanabrolu**[◇]
[‡]University of Arizona, Tucson, AZ [♣]Microsoft Research Montréal
[◇]Allen Institute for AI, Seattle, WA
`{ruoyaowang,pajansen}@arizona.edu`
`macote@microsoft.com`, `raja@allenai.org`

Abstract

In this work, we explore techniques for augmenting interactive AI agents with information from symbolic modules, much like humans use tools like calculators and GPS systems to assist with arithmetic and navigation. We test our agent's abilities in text games—challenging benchmarks for evaluating the multi-step reasoning abilities of game agents in grounded, language-based environments. Our experimental study indicates that injecting the actions from these symbolic modules into the action space of a behavior cloned transformer agent increases performance on four text game benchmarks that test arithmetic, navigation, sorting, and common sense reasoning by an average of 22%, allowing an agent to reach the highest possible performance on unseen games. This action injection technique is easily extended to new agents, environments, and symbolic modules.[1]

1 Introduction

Interactive fiction games (or *text games*) evaluate AI agents abilities to perform complex multi-step reasoning tasks in interactive environments that are rendered exclusively using textual descriptions. Agents typically find these games challenging due to the complexities of the tasks combined with the reasoning limitations of contemporary models. Overall performance is generally low, with agents currently solving only 30% of classic interactive fiction games such as Zork (Ammanabrolu and Hausknecht, 2020; Yao et al., 2021; Atzeni et al., 2022). Similarly, reframing benchmarks such as question answering into text games where agents must interactively reason with their environment and make their reasoning steps explicit causes performance to substantially decrease (Wang et al., 2022), highlighting both the capacity of this methodology to evaluate multi-step reasoning, and the limitations of current language models.

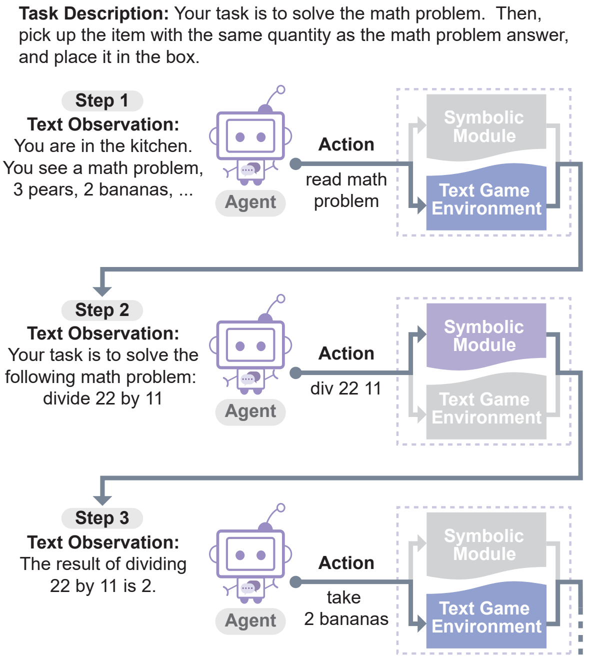


Figure 1: An overview of our approach on an example game evaluating arithmetic ability. At each step, the agent receives an observation from the environment, then takes an action. By providing actions that interface to symbolic modules (such as a calculator), the agent is able to use external knowledge to help solve the task.

While large language models are capable of a variety of common sense reasoning abilities (Liu et al., 2022b; Ji et al., 2020), contemporary agents typically struggle on tasks such as navigation, arithmetic, knowledge base lookup, and other tasks that humans typically make use of external tools (such as GPS systems, calculators, and books) to solve. This is at times frustrating, because the tasks they perform poorly on can sometimes be solved in a few dozen lines of code. In this work, we show that combining both approaches is possible for text game agents, with our approach shown in Figure 1. We develop symbolic modules for arithmetic, navigation, sorting, and knowledge base lookup in

---

[1] We release our system as open source, available at `http://github.com/cognitiveailab/neurosymbolic/`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2777–2788
May 2-6, 2023

PYTHON, paired with new benchmark games for testing these capacities in interactive text game environments. We empirically demonstrate that injecting actions from those modules into the action space of each game can allow transformer-based agents to make use of that information, and achieve near-ceiling performance on unseen benchmark games that they otherwise find challenging.

2 Related Work

Neurosymbolic reasoning offers the promise of combining the inference capabilities of symbolic programs with the robustness of large neural networks. In the context of text games, Kimura et al. (2021a) develop methods to decompose text games into a set of logical rules, then combine these rules with deep reinforcement learning (Kimura et al., 2021b) or integer linear programming (Basu et al., 2021) to substantially increase agent performance while providing a more interpretable framework for understanding why agents choose specific actions (Chaudhury et al., 2021). More generally, neurosymbolic reasoning has been applied to a variety of multi-step inference problems, such as multi-hop question answering (Weber et al., 2019), language grounding (Zellers et al., 2021), and semantic analysis (Cambria et al., 2022).

Because text games require interactive multi-step reasoning, agents have most commonly been modelled using reinforcement learning (e.g. He et al., 2016; Zahavy et al., 2018; Yao et al., 2020), though overall performance on most environments remains low (see Jansen, 2022; Osborne et al., 2021, for reviews). Recently, alternative approaches modeling reinforcement learning as a sequence-to-sequence problem using imitation learning have emerged, centrally using behavior cloning (Torabi et al., 2018), decision transformers (Chen et al., 2021), and trajectory transformers (Janner et al., 2021). These approaches model interactive multi-step reasoning problems as a Markov decision process, where an agent's observation and action history up to some depth are provided as input, and the transformer must predict the next action for the agent to take. Behavior cloning and decision transformers have recently been applied to text games with limited success (Wang et al., 2022). Here, we show that the performance of a behavior cloned transformer can substantially increase when augmented with neurosymbolic reasoning.

3 Approach

Figure 1 illustrates the workflow of our approach. At each time step t, based on the observation o_t, the symbolic module will generate a set of valid actions A_t^m, and the text game environment will have a distinct set of valid actions A_t^e. Let the valid action set at step t be $A_t = A_t^m \cup A_t^e$. Given o_t and A_t, the agent needs to choose an action $a_t \in A_t$ to take. Note that in principle, any agent could be adapted to use this approach, since we simply inject actions from the symbolic modules into the environment action space. At a given time step, our approach checks if a_t is a valid symbolic action. If $a_t \in A_t^m$ (e.g. *div 22 11* in Figure 1), the symbolic module will generate the next observation o_{t+1}, otherwise the text game environment will take a_t and generate o_{t+1} (e.g. *take 2 bananas* in Figure 1).

4 Environments and Symbolic Modules

We evaluate our approach to neurosymbolic reasoning using four text game benchmark environments centered around pick-and-place tasks, including one existing benchmark and three new developed for this work. Each environment supports parametric variation to generate many different games. These environments are outlined below, with additional details and example playthroughs found in APPENDIX B. All environments were implemented using the TEXTWORLDEXPRESS game engine (Jansen and Côté, 2022).

Text World Common Sense (TWC): A benchmark common sense reasoning task (Murugesan et al., 2021) where agents must collect objects from the environment (e.g. *dirty socks*), and place those objects in their canonical common sense locations (e.g. *washing machine*). The symbolic module for this game allows agents to query a knowledge base of *(subject, relation, object)* triples (e.g. *(cushion, hasCanonicalLocation, sofa)*).

MapReader: A navigation-themed pick-and-place game similar to Coin Collector (Yuan et al., 2018). An agent starts in a random location (e.g. *the kitchen*), and is provided with a target location (e.g. *the garage*). The agent must navigate to the target location, pick up a coin, then return to the starting location and place it in a box. The agent is further provided with a map that can be used for efficient route planning. The navigation symbolic module paired with this environment scrapes the observation space for location information (e.g. *you are*

Knowledge Base Module
> *query cushion* cushion located sofa cushion located armchair
Navigation Module
You are currently in the kitchen. > *next step to living room* The next location to move to is: hallway.
Arithmetic Module
> *mul 3 6* Multiplying 3 and 6 results in 18.
Sorting Module
> *sort ascending* The objects in ascending order are: 8mg of steel, 2g of iron, 5kg of copper.

Table 1: Example actions (inputs) and responses from the four symbolic modules investigated in this work.

currently in the kitchen), and both complete (e.g. *the map*) or partial (e.g. *to the north you see the hallway*) spatial connection information.

Arithmetic: A math-themed task, where agents must read and solve a math problem in order to know which object from a set of objects to pick-and-place. An example problem is *"take the bundle of objects that is equal to 3 multiplied by 6, and place them in the answer box"*, where the agent must complete the task by choosing *18 apples*. Distractor objects are populated with quantities that correspond to performing the arithmetic incorrectly (e.g. *3 oranges*, corresponding to subtracting 3 from 6). We pair the arithmetic game with a calculator module capable of performing addition, subtraction, multiplication, and division.

Sorting: A sorting-themed game where the agent begins in a room with three to five objects, and is asked to place them in a box one at a time in order of increasing quantity. To add complexity, quantities optionally include units (e.g. *5kg of copper, 8mg of steel*) across measures of volume, mass, or length. The sorting game is paired with a module that scrapes the observation space for mentions of objects that include quantities, and sorts these in ascending or descending order on command.

4.1 Symbolic Modules

Examples of symbolic modules and their responses are provided in Table 1. The number of valid actions injected by each module varies between 2 from the sorting module (*ascending/descending*) to over 500 from the knowledge base look-up (one for each object and its canonical locations present in the knowledge base). Symbolic modules were implemented in PYTHON as a wrapper around the TEXTWORLDEXPRESS API, allowing modules to monitor observations from the environment, inject actions, and provide responses for any actions they recognized as valid.

5 Models

In this section, we introduce the reinforcement learning and behavior cloning agents used in our experiments. Additional details and hyperparameters are provided in APPENDIX A.

Deep Reinforcement Relevance Network (DRRN): The DRRN (He et al., 2016) is a fast and strong reinforcement learning baseline that is frequently used to deliver near state-of-the-art performance in a variety of text games (e.g. Xu et al., 2020; Yao et al., 2020; Wang et al., 2022). At each step, the DRRN separately encodes the observation and candidate actions using several GRUs (Cho et al., 2014). A Deep Q-Network is then used to estimate Q-values for each *(observation, candidate action)* pair. The candidate action with the highest predicted Q-value will be chosen as the next action.

Behavior Cloning: Behavior cloning (Torabi et al., 2018) is a form of imitation learning similar to the Decision Transformer (Chen et al., 2021) that models reinforcement learning as a sequence-to-sequence problem, predicting the next action given a series of previous observations. We follow the strategy of Ammanabrolu et al. (2021) in adapting behavior cloning to text games, where the model input at step t includes the task description, current state observation, previous action, and previous state observation $(d, o_t, a_{t-1}, o_{t-1})$. During training, the agent is fine-tuned on gold trajectories, where the training target is to generate action a_t from the gold trajectories. During evaluation, the agent performs inference online in the text game environment. For experiments reported here, we used a T5-base model (Raffel et al., 2020).

5.1 Oracle Agents and Gold Trajectories

To generate training data for the behavioral cloning model, we implement oracle agents that generate optimal and generalizable solution trajectories for each benchmark. For example, an oracle agent for an arithmetic game always reads the math problem, picks up the object with the same quantity as the

	DRRN				Behavior Cloned Transformer			
	Baseline		NeuroSymbolic		Baseline		NeuroSymbolic	
Benchmark	Score	Steps	Score	Steps	Score	Steps	Score	Steps
MapReader	0.02	50	0.02	50	0.71	27	**1.00**	**10**
Arithmetic	0.17	10	0.14	7	0.56	5	**1.00**	**5**
Sorting	0.03	21	0.03	18	0.72	7	**0.98**	**8**
TWC	0.57	27	0.37	34	0.90	6	**0.97**	**3**
Average	0.20	27	0.14	27	0.72	11	**0.99**	**7**

Table 2: Average model performance across 100 games in the unseen test set. *Scores* are normalized to between 0 and 1 (higher is better), while *steps* represents the number of steps an agent takes in the environment (lower is better). Neurosymbolic performance reflects when models have access to symbolic modules in their action space.

math problem answer, then places that object in the answer box. For experiments using symbolic modules, we further insert appropriate module actions when the agent requires that information to complete the next step – for example, using the calculator module after reading the math problem in the arithmetic game.

6 Results and Discussion

The results of both DRRN and behavior cloning experiments across each benchmark are shown in Table 2. We report the average model performance across 100 games in the unseen test set. The DRRN achieves a low average performance of 0.20 without modules, while adding symbolic modules into the action space does not improve performance. In contrast, the behavior cloned T5 model has a moderate average performance of 0.72 without modules, while adding symbolic modules increases average task performance to 0.99, nearly solving each task. Symbolic modules also make the behavior cloned agent more efficient, reducing the average steps required to complete the tasks from 11 to 7, matching oracle agent efficiency.

Why does behavior cloning perform well? The baseline behavior cloned transformer achieves moderate overall performance, likely owing at least in part due to its use of gold trajectories for training. Large pretrained transformers contain a variety of common sense knowledge and reasoning abilities (Zhou et al., 2020; Liu et al., 2022c) which likely contributes to the high performance on TWC, where the model only needs to match objects with their common sense locations. In contrast, while transformers have some arithmetic abilities, their accuracy tends to vary with the frequency of specific tokens in the training data (Razeghi et al., 2022), likely causing the modest performance on the Arithmetic game. Here, we show that instead of increasing the size of training data, transformers can be augmented with symbolic modules that perform certain kinds of reasoning with high accuracy. Compared to the DRRN, the presence of gold trajectories for training allows the behavior cloned transformer to efficiently learn how to capitalize on the knowledge available from those modules.

Why does the DRRN perform poorly? We hypothesize that two considerations make these tasks difficult for the Deep Reinforcement Relevance Network. The model frequently tries to select actions that lead to immediate reward (such as immediately picking the correct number of objects in the arithmetic game), without having first done the prerequisite actions (like reading or solving the math problem) that would naturally lead it to select that action. This creates an ungeneralizable training signal, causing the model to fail to learn the task. In addition, the action spaces for each game are generally large – baseline games contain between 5 and 30 possible valid actions at each step (see Table 4 in the APPENDIX), resulting in up to 24 million possible trajectories up to 5 steps, which is challenging to explore. Inspired by Liu et al. (2022a), our future work will aim to overcome these limitations, and allow reinforcement learning models to learn to efficiently and effectively exploit information from symbolic modules.

How does performance compare against other agents? While most environments used in this work are new, TEXTWORLD COMMON SENSE is an existing benchmark. Figure 3 compares the Neurosymbolic Behavior Cloned Transformer against recent models that use a combination of reinforcement learning, logic, knowledge resources, and case-based reasoning. While the performance is not directly comparable – here, we use the TEXTWORLDEXPRESS reimplementation of TWC with supervised learning, while other models use

Model	Score	Steps
SceneIt (Murugesan et al., 2022)	0.88	20
Bike+CBR (Atzeni et al., 2022)	0.93	17
SceneGraph (Tanaka et al., 2022)	0.91	17
IG (Basu et al., 2021)	0.92	13
BCT Baseline (Ours)	0.90	6
BCT+NeuroSymbic (Ours)	**0.97**	**3**

Table 3: A comparison of performance on TWC on unseen games on the "easy" setting. Note that models may not be directly comparable, as this work uses the TEXTWORLDEXPRESS reimplementation of TWC, and supervised learning. *Scores* are normalized to between 0 and 1 (higher is better), while *steps* represents the number of steps an agent takes in the environment (lower is better).

the original implementation with a mix of reinforcement learning and case-based reasoning – we can make the high-level observation that the performance of both the baseline and Neurosymbolic Behavior Cloned Transformer meets or exceeds the scores of previous models, while generating paths that are more efficient – by a factor of up to 7x.

7 Conclusion

In this paper, we present an approach to neurosymbolic reasoning for text games using action space injection that can be easily adapted to existing text game environments. For models that are capable of exploiting the information provided by the symbolic modules, this technique allows agents to inexpensively augment their reasoning skills to solve more complex tasks. We empirically demonstrate this approach can substantially increase task performance on four benchmark games using a behavior cloned transformer.

Limitations

Two assumptions highlight core limitations in the scope of our results for augmenting models with neurosymbolic reasoning: the privileged access to a list of valid actions, and the use of gold trajectories for training the behavior cloned transformer.

Valid Actions: One of the central challenges with text games is that the space of possible action utterances is large, and text game parsers recognize only a subset of possible actions (e.g. *take apple on the table*) while being unable to successfully interpret a broader range of more complex utterances (e.g. *take the red fruit near the fridge*). As a result, nearly all contemporary models (e.g. Ammanabrolu and Hausknecht, 2020; Adhikari et al., 2020; Murugesan et al., 2021) make use of the *valid action aid* (Hausknecht et al., 2020), where at a given step the model is provided with an exhaustive list of possible valid actions from the environment simulator, from which one action is chosen. The models presented here similarly use this aid. The DRRN functions essentially as a ranker to select the most probable next action. The behavior cloned transformer generates a candidate action that is aligned using cosine similarity with the list of valid actions, where the action with the highest overlap is chosen as the next action. Overcoming the *valid action aid* will generally require either more complex simulation engines capable of interpreting a wider variety of intents from input actions, or models that learn sets of valid actions from a large amount of training data – though these generally demonstrate lower performance than those using valid actions (e.g. Yao et al., 2020).

Gold Trajectories: In this work we demonstrate a substantial improvement in the performance of a behavior cloned transformer when augmented with neurosymbolic reasoning, but this requires the use of gold trajectories demonstrating the use of those symbolic modules. Gold training data is not available in many reinforcement learning applications, and the model comparison we perform (DRRN versus behavior cloning) is meant to highlight the capacity for the behavior cloned model to learn to make use of symbolic modules through gold demonstrations, rather than to suggest the DRRN is incapable of this. In future work, we aim to develop training procedures to allow models that do not have the benefit of using gold trajectories to make use of symbolic modules.

Ethics Statement

Broader Impacts: As noted by Ammanabrolu and Riedl (2021), the ability to perform long-term multi-step reasoning in complex, interactive, partially-observable environments has downstream applications beyond playing games. Text games are platforms upon which to explore interactive, situated communication such as dialogue. Although reinforcement learning is applicable to many sequential decision making domains, our setting is most relevant to creating agents that affect change via language. This mitigates physical risks prevelant in robotics, but not cognitive and emotional risks, as any system capable of generating natural language is capable of biased language use (Sheng et al., 2021).

Intended Use: The method described in this paper involves fine-tuning a large pretrained transformer model. The data generated for fine-tuning was generated by gold agents, and not collected from human participants. The trained models are intended to operate on these benchmark tasks that assess reasoning capacities in navigation, arithmetic, and other common sense competencies. Large language models have been shown to exhibit a variety of biases (e.g. Nadeem et al., 2021) that may cause unintended harms, particularly (in the context of this work) in unintended use cases.

Computation Time: Training large models can involve a large carbon footprint (Strubell et al., 2019), or decrease the availability of a method due to the barriers in accessing high performance compute resources. The proposed technique can reduce the need for large models by augmenting smaller models with more complex reasoning through symbolic modules. The behavior cloning experiments achieve strong performance with T5-base, highlighting the capacity of modest models that can be run with workstation GPUs to be better exploited for complex reasoning tasks.

Acknowledgements

This work supported in part by National Science Foundation (NSF) award #1815948 to PJ, and the Allen Institute for Artificial Intelligence (AI2).

References

Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and Will Hamilton. 2020. Learning dynamic belief graphs to generalize on text-based games. In *Advances in Neural Information Processing Systems*, volume 33, pages 3045–3057. Curran Associates, Inc.

Prithviraj Ammanabrolu and Matthew Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. In *International Conference on Learning Representations*.

Prithviraj Ammanabrolu and Mark Riedl. 2021. Learning knowledge graph-based world models of textual environments. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS)*.

Prithviraj Ammanabrolu, Jack Urbanek, Margaret Li, Arthur Szlam, Tim Rocktäschel, and Jason Weston. 2021. How to motivate your dragon: Teaching goal-driven agents to speak and act in fantasy worlds. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 807–833, Online. Association for Computational Linguistics.

Mattia Atzeni, Shehzaad Zuzar Dhuliawala, Keerthiram Murugesan, and MRINMAYA SACHAN. 2022. Case-based reasoning for better generalization in textual reinforcement learning. In *International Conference on Learning Representations*.

Kinjal Basu, Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Kartik Talamadupula, Tim Klinger, Murray Campbell, Mrinmaya Sachan, and Gopal Gupta. 2021. A hybrid neuro-symbolic approach for text-based games using inductive logic programming. In *Proceedings of the 1st Workshop on Combining Learning and Reasoning: Programming Languages, Formalisms, and Representations*.

Erik Cambria, Qian Liu, Sergio Decherchi, Frank Xing, and Kenneth Kwok. 2022. SenticNet 7: A commonsense-based neurosymbolic AI framework for explainable sentiment analysis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3829–3839, Marseille, France. European Language Resources Association.

Subhajit Chaudhury, Prithviraj Sen, Masaki Ono, Daiki Kimura, Michiaki Tatsubori, and Asim Munawar. 2021. Neuro-symbolic approaches for text-based policy learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3073–3078, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. 2020. Interactive fiction games: A colossal adventure. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7903–7910.

Ji He, Mari Ostendorf, Xiaodong He, Jianshu Chen, Jianfeng Gao, Lihong Li, and Li Deng. 2016. Deep reinforcement learning with a combinatorial action space for predicting popular Reddit threads. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1838–1848, Austin, Texas. Association for Computational Linguistics.

Michael Janner, Qiyang Li, and Sergey Levine. 2021. Offline reinforcement learning as one big sequence modeling problem. In *Advances in Neural Information Processing Systems*.

Peter Jansen. 2022. A systematic survey of text worlds as embodied natural language environments. In *Proceedings of the 3rd Wordplay: When Language Meets Games Workshop (Wordplay 2022)*, pages 1–15, Seattle, United States. Association for Computational Linguistics.

Peter A Jansen and Marc-Alexandre Côté. 2022. Textworldexpress: Simulating text games at one million steps per second. *arXiv preprint arXiv:2208.01174*.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 725–736, Online. Association for Computational Linguistics.

Daiki Kimura, Subhajit Chaudhury, Masaki Ono, Michiaki Tatsubori, Don Joven Agravante, Asim Munawar, Akifumi Wachi, Ryosuke Kohita, and Alexander Gray. 2021a. LOA: Logical optimal actions for text-based interaction games. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 227–231, Online. Association for Computational Linguistics.

Daiki Kimura, Masaki Ono, Subhajit Chaudhury, Ryosuke Kohita, Akifumi Wachi, Don Joven Agravante, Michiaki Tatsubori, Asim Munawar, and Alexander Gray. 2021b. Neuro-symbolic reinforcement learning with first-order logic. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3505–3511, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Iou-Jen Liu, Xingdi Yuan, Marc-Alexandre Côté, Pierre-Yves Oudeyer, and Alexander G. Schwing. 2022a. Asking for knowledge: Training rl agents to query external knowledge using language. *ArXiv*, abs/2205.06111.

Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. 2022b. Generated knowledge prompting for commonsense reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3154–3169, Dublin, Ireland. Association for Computational Linguistics.

Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. 2022c. Generated knowledge prompting for commonsense reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3154–3169.

Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Pushkar Shukla, Sadhana Kumaravel, Gerald Tesauro, Kartik Talamadupula, Mrinmaya Sachan, and Murray Campbell. 2021. Text-based RL Agents with Commonsense Knowledge: New Challenges, Environments and Baselines. In *Thirty Fifth AAAI Conference on Artificial Intelligence*.

Keerthiram Murugesan, Subhajit Chaudhury, and Kartik Talamadupula. 2022. Eye of the beholder: Improved relation generalization for text-based reinforcement learning agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11094–11102.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Philip Osborne, Heido Nomm, and André Freitas. 2021. A survey of text games for reinforcement learning informed by natural language. *Transactions of the Association for Computational Linguistics*, 10:873–887.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2021. Societal biases in language generation: Progress and challenges. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4275–4293, Online. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Tsunehiko Tanaka, Daiki Kimura, and Michiaki Tatsubori. 2022. Commonsense knowledge from scene graphs for textual environments. *arXiv preprint arXiv:2210.14162*.

Faraz Torabi, Garrett Warnell, and Peter Stone. 2018. Behavioral cloning from observation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4950–4957. International Joint Conferences on Artificial Intelligence Organization.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. 2019. NLProlog: Reasoning with weak unification for question answering in natural language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6151–6161, Florence, Italy. Association for Computational Linguistics.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, Joey Tianyi Zhou, and Chengqi Zhang. 2020. Deep reinforcement learning with stacked hierarchical attention for text-based games. *Advances in Neural Information Processing Systems*, 33:16495–16507.

Shunyu Yao, Karthik Narasimhan, and Matthew Hausknecht. 2021. Reading and acting while blindfolded: The need for semantics in text game agents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3097–3102, Online. Association for Computational Linguistics.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. 2020. Keep CALM and explore: Language models for action generation in text-based games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8736–8754, Online. Association for Computational Linguistics.

Xingdi Yuan, Marc-Alexandre Côté, Alessandro Sordoni, Romain Laroche, Rémi Tachet des Combes, Matthew J. Hausknecht, and Adam Trischler. 2018. Counting to explore and generalize in text-based games. *ArXiv*, abs/1806.11525.

Tom Zahavy, Matan Haroush, Nadav Merlis, Daniel J Mankowitz, and Shie Mannor. 2018. Learn what not to learn: Action elimination with deep reinforcement learning. In *NeurIPS*.

Rowan Zellers, Ari Holtzman, Matthew Peters, Roozbeh Mottaghi, Aniruddha Kembhavi, Ali Farhadi, and Yejin Choi. 2021. PIGLeT: Language grounding through neuro-symbolic interaction in a 3D world. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2040–2050, Online. Association for Computational Linguistics.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9733–9740.

A Appendix: Experiment Details

A.1 Training and evaluation sets

For each game, we randomly generate 100 parametric variations for each of the train, development, and test sets. To encourage and evaluate generality, problems are unique across sets – for example, arithmetic problems (for the Arithmetic game) or task objects (for TWC) found in the training set are not found in the development or test sets.

A.2 Hyperparameters

Following standard practice (e.g. (Wang et al., 2022; Xu et al., 2020; Hausknecht et al., 2020)), the DRRN models are trained for 100k steps. We parallelly train DRRN on 16 environment instances with five different random seeds and the average results are reported. The behavior cloned transformers are trained for between 2 and 20 epochs, with the best model (as evaluated on the development set) used for evaluating final performance on the test set. Trained models are evaluated on all 100 parametric variations in the development or test set. Environments are limited to 50 steps, such that if the agent exceeds this many steps without reaching an end state, the score at the last step is taken to be the final score, and the environment resets. Model training time varied between 1 hour and 12 hours, with the TWC model that includes a large number of symbolic module actions requiring the largest training time.

A.3 Implementation details

We make use of an existing DRRN implementation[2] and adapted it to the TEXTWORLDEXPRESS environment. At each step, the current game state observation, task description, inventory information, and the current room description are concatenated into one string and encoded by a GRU. All

---

[2] `https://github.com/microsoft/tdqn`

	No Modules			With Symbolic Modules		
Benchmark	Min	Avg	Max	Min	Avg	Max
MapReader	4	6.2	22	6	9.3	26
Arithmetic	9	14.3	52	17	21.5	53
Sorting	5	9.3	35	7	11.0	37
TWC	3	6.3	14	544	547.8	561
Average	5.3	9.0	30.8	143.5	147.4	169.3

Table 4: The minimum, mean, and maximum number of valid actions per step, for each benchmark. Values represent averages determined using a random agent that is run to 50 steps for on 10 training episodes per benchmark.

candidate actions are encoded by another GRU. The Q-value of each encoded (observation, candidate action) pair is then estimated by a Q-network consists of two linear layers. During training, the next action is sampled from all candidate actions based on the estimated Q-values. During evaluation, the action with the highest estimated Q-value is chosen as the next action.

For the behavior cloned transformer, the input string of the T5 model at step t are formatted as:

d **</s> OBS** o_t **</s> INV** o_t^{inv} **</s> LOOK** o_t^{look} **</s> <extra_id_0> </s> PACT** a_{t-1} **</s> POBS** o_{t-1} **</s>**

where d is the task description, **</s>** and **<extra_id_0>** are the special tokens for separator and mask for text to generate used by the T5 model, **OBS**, **INV**, **LOOK**, **PACT**, and **POBS** are the special tokens representing observation o_t, inventory information o_t^{inv}, the current room description obtained by the "look around" action o_t^{look}, previous action a_{t-1}, and previous observation o_{t-1}, respectively. We use beam search to generate the top 16 strings from the T5 model, and choose the first string that is a valid action as the action to take. In the case where the model does not generate an exact match, we use cosine similarity to pick the valid action that has the highest unigram overlap with an action generated by T5.

B Environments and Symbolic Modules

Action Space: The number of valid actions per step for each benchmark is shown in Table 4, with these values collected by a random agent that runs for 50 steps across 10 training episodes. Environments contain an average of 9 valid actions per step (range 5 to 30), not including actions injected from the symbolic modules. After adding the module actions, the action space becomes up to twice as large for the MapReader, Arithmetic, and Sorting benchmarks. The knowledge base module for TWC adds approximately 530 actions, substantially increasing the action space.

Text World Common Sense (TWC) Game

Task Description: Your task is to pick up objects, then place them in their usual locations in the environment.

You are in the corridor. In one part of the room you see a shoe cabinet that is closed. There is also **a white coat**. You also see a key holder, that has nothing on it. In another part of the room you see a hat rack, that has nothing on it. In one part of the room you see **a coat hanger**, that has nothing on it. There is also a umbrella stand, that has nothing on it.
Inventory:
Your inventory is currently empty.
> *query* ***white coat***

The results are:
white coat located **coat hanger**
white coat located wardrobe
Inventory:
Your inventory is currently empty.
> *take* ***white coat***

You take the **white coat**.
Inventory:
a white coat
> *put* ***white coat*** *in* ***coat hanger***

Game completed.

Table 5: An example of a Text World Common Sense (TWC) game. *Actions* for the solution are italicized. Relevant **task objects** and **target locations** are bolded, while distractor locations are underlined. Symbolic module actions and their responses are highlighted in *blue*.

B.1 Text World Common Sense (TWC)

A benchmark common sense reasoning task where agents must collect objects from the environment (e.g. *dirty socks*) and place them in their canonical locations (e.g. *washing machine*). We use the TEXTWORLDEXPRESS implementation of the original environment (Murugesan et al., 2021), in a setting analogous to easy difficulty, where the agent must pick and place one misplaced item in the environment. The agent receives a score of 0.5 for picking up the object, and 1.0 for placing it in the correct location. Table 5 shows an example of the TWC Game in which the agent places a clean shirt in the wardrobe with the help of querying the

Arithmetic Game

Task Description: Your first task is to solve the math problem. Then, pick up the item with the same quantity as the math problem answer, and place it in the box.

```
You are in the kitchen. In one part of the room you see a fridge that is closed. There is also a dining chair that has 11 tangerines, 33 papayas, 242 strawberries, and 20 peaches on it. You also see a box, that is empty. In another part of the room you see a math problem. In one part of the room you see a dishwasher that is closed. There is also a trash can that is closed. You also see an oven. In another part of the room you see a cutlery drawer that is closed. In one part of the room you see a stove. There is also a kitchen cupboard that is closed. You also see a counter that has 6 oranges, and 2 bananas on it.
Inventory:
  Your inventory is currently empty.
> take math problem

You take the math problem.
Inventory:
  a math problem
> read math problem

Your task is to solve the following math problem: divide 22 by 11. Then, pick up the item with the same quantity as the answer, and place it in the box.
Inventory:
  a math problem
> div 22 11

The result of dividing 22 by 11 is 2.
Inventory:
  a math problem
> take 2 bananas

You take the 2 bananas.
Inventory:
  a math problem
  2 bananas
> put 2 bananas in box

Game completed.
```

Table 6: An example Arithmetic game. *Actions* for the solution are italicized. Relevant **task objects** are bolded, while distractor objects are underlined. Symbolic module actions and their responses are highlighted in *blue*.

common sense location of a clean shirt.

Knowledge Base Module: We pair TWC with a symbolic module that provides queries to a simple knowledge base of <*object, hasCanonicalLocation, container*> triples. The symbolic module generates and accepts actions of the form QUERY <QUERY TOKENS>, where <QUERY TOKENS> corresponds to all object and container names in the knowledge base. This results in an increase of approximately 530 actions in the action space.

B.2 Arithmetic Game

The Arithmetic game requires agents to read a math problem, solve it, then perform a pick-and-place task based on the answer. For example, the agent may read the math problem (*"Take the bundle of objects that is equal to 3 multiplied by 6, and place them in the box"*), and must then perform the arithmetic then *take 18 apples* and place them in the answer box. Distractor objects are populated corresponding to performing the arithmetic incorrectly (for example, including 3 oranges, corresponding to subtracting 3 from 6, and 2 pears, corresponding to 6 divided by 3), with the condition that results are positive integer values. Agents receive a score of 0.5 for picking up the correct object, and 1.0 for completing the task successfully. An example playthrough of the Arithmetic game is in Table 6.

Arithmetic Module: We pair the Arithmetic game with an Arithmetic module that adds actions for addition, subtraction, multiplication, and division. To reduce the complexity of the action space, only actions with arguments from the current math problem are enumerated (e.g. *add 3 6, sub 3 6, sub 6 3, mul 3 6, div 3 6, div 6 3*).

B.3 Sorting Game

The sorting game is a pick-and-place game that presents an agent with 3 to 5 objects, and asks the agent to place them in an answer box one at a time based on order of increasing quantity. To add complexity to the game, quantities optionally include units (e.g. *5kg of copper, 8mg of steel, 2g of iron*) across measures of volume, mass, or length. The agent score is the normalized proportion of objects sorted in the correct order, where perfect sorts receive a score of 1.0, and errors cause the score to revert to zero and the game to end. An example playthrough of the Sorting game is in Table 7.

Sorting Module: The sorting module monitors observations for mentions of objects (nouns) that include quantities, while also interpreting and normalizing quantities based on known units. The module injects two actions: sort ascending, and sort descending, that provides the user with a sorted list of objects.

B.4 MapReader Game

MapReader is a navigation oriented pick-and-place game similar to Coin Collector (Yuan et al., 2018), with the added complexity that the agent is provided with a map of the environment that can be used to more efficiently navigate. Environments and their maps are randomly generated to contain up to 15 locations drawn from 50 locations in Coin Collector. The agent begins in a randomly chosen location, and is asked to move to a target location (e.g. *the kitchen*), take a coin, then return to the starting location and place it in a box. Target locations are randomly chosen to be between 1 and 4 steps from the starting location. The most efficient solution method is to read the map, determine the shortest path between the agents current location

Sorting Game

Task Description: Your task is to sort objects by quantity. First, place the object with the smallest quantity in the box. Then, place the objects with the next smallest quantity in the box, and repeat until all objects have been placed in the box.

You are in the kitchen. In one part of the room you see a fridge that is closed. There is also a counter that has **15kg of cedar**, and **21kg of marble** on it. You also see a dining chair that has **25g of oak**, and **47g of brick** on it. In another part of the room you see a box, that is empty. In one part of the room you see a dishwasher that is closed. There is also a trash can that is closed. You also see an oven. In another part of the room you see a cutlery drawer that is closed. In one part of the room you see a stove. There is also a kitchen cupboard that is closed.
Inventory:
Your inventory is currently empty.
> *sort ascending*

The observed items, sorted in order of increasing quantity, are: **25 g of oak**, **47 g of brick**, **15 kg of cedar**, **21 kg of marble**.
Inventory:
Your inventory is currently empty.
> *take **25g of oak***

You take the **25g of oak**.
Inventory:
25g of oak
> *put **25g of oak** in box*

You put the **25g of oak** in the box.
Inventory:
Your inventory is currently empty.
> *take **47g of brick***

You take the **47g of brick**.
Inventory:
47g of brick
> *put **47g of brick** in box*

You put the **47g of brick** in the box.
Inventory:
Your inventory is currently empty.
> *take **15kg of cedar***

You take the **15kg of cedar**.
Inventory:
15kg of cedar
> *put **15kg of cedar** in box*

You put the **15kg of cedar** in the box.
Inventory:
Your inventory is currently empty.
> *take **21kg of marble***

You take the **21kg of marble**.
Inventory:
21kg of marble
> *put **21kg of marble** in box*

Game completed.

Table 7: An example Sorting game. *Actions* for the solution are italicized. Relevant **task objects** are bolded. Symbolic module actions and their responses are highlighted in *blue*.

and target location, follow that path to retrieve the coin, then follow the path in reverse to return the coin to the starting location. The agent receives a score of 0.5 for retrieving the coin, and 1.0 for placing the coin in the box at the start location. An example of the MapReader game is shown in Table 8.

Navigation Module: We pair MapReader with a navigation module that scrapes the environment for both complete map information (obtained if the agent chooses to read the map), as well as partial information such as the current location (e.g. *"You are in the kitchen"*) and connecting locations (e.g. *"To the north you see the living room"*) that can be used to incrementally build a map. The module adds actions that, if selected, provide the next step in the shortest path to navigate to all known locations in the environment (e.g. *next step to living room*, *next step to garage*, ...).

MapReader Game

Task Description: Your task is to take the coin that is located in **the laundry room**, and put it into the box found in **the foyer**. A map is provided, that you may find helpful.

You are in **the foyer**. In one part of the room you see a box, that is empty.

To the East you see the corridor.
Inventory:
a map
> *read map*

The map reads:
The living room connects to the backyard and corridor.
The garage connects to the driveway.
The laundry room connects to the corridor.
The backyard connects to the living room, alley, kitchen and sideyard.
The bedroom connects to the corridor.
The sideyard connects to the backyard and driveway.
The kitchen connects to the bathroom, pantry and backyard.
The supermarket connects to the alley.
The foyer connects to the corridor.
The pantry connects to the kitchen.
The driveway connects to the sideyard, alley and garage.
The street connects to the alley.
The alley connects to the driveway, supermarket, street and backyard.
The bathroom connects to the kitchen.
The corridor connects to the living room, foyer, bedroom and laundry room.
Inventory:
a map
> *next step to laundry room*

The next location to go to is: corridor
Inventory:
a map
> *move east*

You are in the corridor. In one part of the room you see a shoe cabinet that is closed. There is also a key holder, that has nothing on it. You also see a hat rack, that has nothing on it. In another part of the room you see a coat hanger, that has nothing on it. In one part of the room you see a umbrella stand, that has nothing on it.
To the North you see the laundry room. To the South you see the living room. To the East you see the bedroom. To the West you see **the foyer**.
Inventory:
a map
> *next step to **laundry room***

The next location to go to is: **laundry room**
Inventory:
a map
> *move north*

continued...

You are in **the laundry room**. In one part of the room you see a coin. There
is also a bench, that has nothing on it. You also see a washing machine that is closed. In another part of the room you see a work table, that has nothing on it. In one part of the room you see a laundry basket, that has nothing on it. There is also a clothes drier that is closed.
To the South you see the corridor.
Inventory:
a map
> *take coin*

You take the coin.
Inventory:
a map
a coin
> *next step to **foyer***

The next location to go to is: corridor
Inventory:
a map
a coin
> *move south*

You are in the corridor. In one part of the room you see a shoe cabinet that is closed. There is also a key holder, that has nothing on it. You also see a hat rack, that has nothing on it. In another part of the room you see a coat hanger, that has nothing on it. In one part of the room you see a umbrella stand, that has nothing on it.
To the North you see **the laundry room**. To the South you see the living room. To the East you see the bedroom. To the West you see **the foyer**.
Inventory:
a map
a coin

> *next step to **foyer***
The next location to go to is: **foyer**
Inventory:
a map
a coin
> *move west*

You are in **the foyer**. In one part of the room you see a box, that is empty.
To the East you see the corridor.
Inventory:
a map
a coin
> *put coin in box*

Game completed.

Table 8: An example of a MapReader game. *Actions* for the solution are italicized. The **starting location** and the **target location** are bolded. Symbolic module actions and their responses are highlighted in *blue*.

Bridging the Gap Between BabelNet and HowNet: Unsupervised Sense Alignment and Sememe Prediction

Xiang Zhang, Ning Shi, Bradley Hauer, Grzegorz Kondrak
Alberta Machine Intelligence Institute (Amii), Department of Computing Science
University of Alberta, Edmonton, Canada
`{xzhang23,ning.shi,bmhauer,gkondrak}@ualberta.ca`

Abstract

As the minimum semantic units of natural languages, sememes can provide interpretable representations of concepts. Despite the widespread utilization of lexical resources for semantic tasks, the use of sememes is limited by a lack of available sememe knowledge bases. Recent efforts have been made to connect BabelNet with HowNet by automating sememe prediction. However, these methods depend on large manually annotated datasets. Instead, we propose to use sense alignment via a novel unsupervised and explainable method. Our method consists of four stages, each relaxing predefined constraints until a complete alignment of BabelNet synsets to HowNet senses is achieved. Experimental results demonstrate the superiority of our unsupervised method over previous supervised ones by an improvement of 12% overall F1 score, setting a new state of the art. Our work is grounded in an interpretable propagation of sememe information between lexical resources, and may benefit downstream applications which can incorporate sememe information.

1 Introduction

Sememes are the minimum semantic units of human languages (Bloomfield, 1926). The theory of semantic primitives (Wierzbicka, 1996a) hypothesizes that the meaning of a word in any language can be decomposed into a finite set of language-independent sememes. For example, the English noun *plant* has distinct senses that correspond to the "factory" and "vegetation" concepts, respectively. The former can be represented by the sememes "industrial", "produce", and "institute/place", and the latter by "crop", "tree", and "flower/grass". (See Figure 2 for additional examples.) Although sememes provide a way of representing concepts, their incorporation into natural language processing (NLP) has been limited by a lack of available sememe resources for commonly used sense inventories. Tremendous efforts have been made to construct sememe knowledge bases (KBs) manually. One of the most widely used is HowNet (Dong and Dong, 2003), which unfortunately is limited to only two languages: English and Chinese.

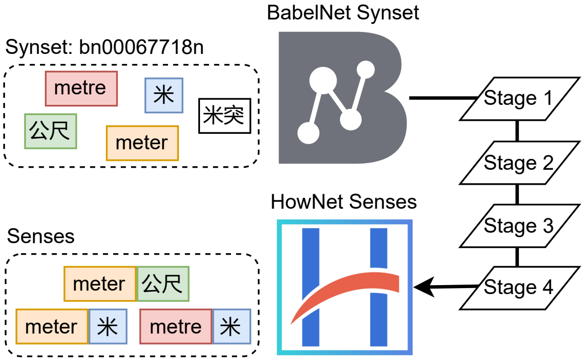
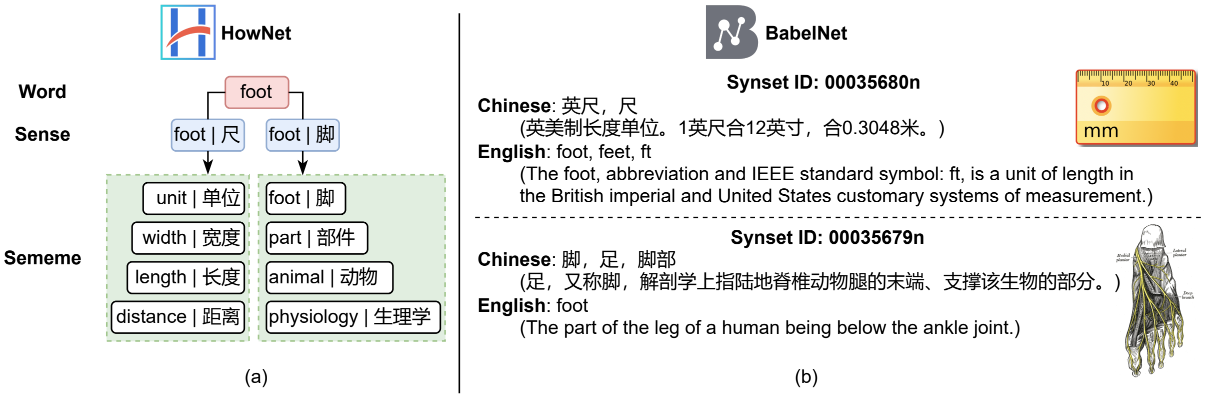


Figure 1: Given a BabelNet synset as input, our alignment algorithm identifies a set of corresponding HowNet senses. The algorithm consists of four stages which progressively relax alignment constrains.

A related problem is linking lexical resources to one another. As manually creating semantic knowledge bases remains a challenging and costly process, BabelNet (Navigli and Ponzetto, 2012) was instead created by combining WordNet (Miller, 1995), Wikipedia, and other resources. While BabelNet covers hundreds of languages, it does not include sememe information. Previous work on automatically predicting sememes for BabelNet concepts has depended on large human labeled data (Qi et al., 2020, 2022). Various systems for automatically aligning word senses across heterogeneous resources have been proposed to mitigate this issue (Meyer and Gurevych, 2011; Pilehvar and Navigli, 2014; Bao et al., 2022), but those methods do not leverage the unique structure of HowNet, which differs from other lexical databases.

In this work, rather than attempting to predict sememes directly, as in prior work, we instead attempt to align BabelNet concepts and HowNet senses. Since each HowNet sense is annotated with

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2789–2798
May 2-6, 2023



Synset ID: 00035680n

Chinese: 英尺，尺
(英美制长度单位。1英尺合12英寸，合0.3048米。)
English: foot, feet, ft
(The foot, abbreviation and IEEE standard symbol: ft, is a unit of length in the British imperial and United States customary systems of measurement.)

Synset ID: 00035679n

Chinese: 脚，足，脚部
(足，又称脚，解剖学上指陆地脊椎动物腿的末端、支撑该生物的部分。)
English: foot
(The part of the leg of a human being below the ankle joint.)

(a) (b)

Figure 2: A comparison of the representations of HowNet senses and BabelNet concepts.

sememes (c.f., Figure 2), correctly mapping a BabelNet concept to a HowNet sense will effectively add sememe information to the corresponding BabelNet synset, as well as associate the HowNet sense with information unique to BabelNet, such as synonymy information. We propose a fully unsupervised algorithm which achieves reliable and stable results, and produces fully explainable mappings. The ability to identify precisely why a particular sememe was associated with a given concept is a unique strength of our method, which we believe will facilitate analysis and development of downstream applications of our work.

The results of our experimental evaluation provide evidence that our unsupervised approach substantially outperforms previous supervised methods on the BabelNet sememe prediction task by up to 12% F-score. This also indirectly demonstrates the high accuracy of the generated alignment between BabelNet and HowNet, which we make available to facilitate further research.[1]

Our contributions are as follows:

- We propose the first method for aligning BabelNet synsets to HowNet senses, adding new information to both resources. Our method is unsupervised and explainable.
- We set a new state of the art for sememe prediction task on the BabelSememe dataset.
- We provide an API for identifying the HowNet senses and their sememes for any BabelNet synset that contains English or Chinese words.
- We perform a detailed analysis and ablation study of the results, focusing on the impact of the differences between HowNet and BabelNet.

2 Related Work

In this section, we first describe the two multilingual knowledge bases, and then discuss the tasks of sense alignment and sememe prediction.

2.1 Multilingual knowledge bases

Our main focus is on the sense alignment between BabelNet and HowNet (Figure 2).

BabelNet (Navigli and Ponzetto, 2012) is a multilingual sense inventory created by automatically combining various knowledge bases including WordNet and Wikipedia. The most recent version covers over 500 languages and contains 1.4 billion word senses. Following the WordNet model, words which are interchangeable in some context are grouped into synonym sets, or *synsets*. Each synset is associated with a unique lexical concept or named entity, and contains all the words which can express that concept or refer to that entity. In BabelNet, the *senses* of a word correspond to the concepts that it can express; that is, each sense of a word corresponds to a synset which contains the word, and shares its meaning with the other words in that synset. As shown in Figure 2, each synset has a unique ID and consists of all word senses that share the same meaning across various languages. Each synset is also associated with a gloss, and, optionally, example sentences or images.

HowNet is a sememe-based knowledge base for both Chinese and English. Each HowNet sense contains a unique Chinese-English word pair, associated with one or more sememes, as shown in Figure 2. Note that the meaning of the term *sense* is different in HowNet and BabelNet, and thus multiple HowNet senses may correspond to a single BabelNet concept. HowNet contains more than 2K

---

[1] Our code and used data are available at: `https://github.com/senseAlign/BabelNet_2_HowNet`

sememes, created by human experts, and more than 100K Chinese and 950K English words. Over its more than 20 years of development, HowNet has become one of the most popular knowledge base in the Chinese NLP community (Dong and Dong, 2003), and has been applied to downstream tasks such as word sense disambiguation (Zhang et al., 2005; Duan et al., 2007; Zhang et al., 2022), word representation learning (Niu et al., 2017), language modeling (Gu et al., 2018), textual adversarial attack (Zang et al., 2020; Qi et al., 2021), text matching (Lyu et al., 2021), and sememe prediction (Qi et al., 2020, 2022).

2.2 Sense aligning

The task of aligning lexical knowledge bases consists of associating entries in one resource with one or more entries in another. As described above, BabelNet was created by associating Wikipedia articles with WordNet senses. As another example, Open Multilingual WordNet (Bond and Foster, 2013) was created by linking wordnets covering various languages. Gurevych et al. (2012) create a sense similarity measure to align WordNet senses with entries in the German OmegaWiki. McCrae and Cillessen (2021) conduct an alignment between WordNet synsets and entities in Wikidata. Bao et al. (2022) present a translation-based method for aligning BabelNet concepts to entries in CLICS and OmegaWiki. Combining knowledge bases in this way provides additional information, and allows knowledge-based methods to be applied to more languages and tasks.

Various approaches to sense alignment between HowNet and the other KBs have also been investigated in prior work. Carpuat et al. (2002) use the tf-idf scores to align the senses in HowNet and synsets in WordNet. Chen and Fung (2004) present a method for aligning HowNet and FrameNet mapping by applying word sense disambiguation. Sornlertlamvanich et al. (2005) propose another algorithm for aligning HowNet and FrameNet, which is based on constructing feature vectors.

Besides, more general methods to link two or more lexical resources (ontologies) have been examined in prior work. McCrae and Cillessen (2021) proposed the *Lexicon Model for Ontologies* (Lemon) which aims to align any amount of lexical resources by modeling a universal semantic representation. Chiarcos et al. (2013) further combine multiple linking methods including Lemon and Graph construction, based on the theory of *Resource Description Framework*, and provide the insight in unifying lexical resources. Our work focuses solely on the alignment between HowNet and BabelNet as they represent two most widely used lexical knowledge bases for English and Chinese. However, similar ideas used in the work can also be applied for the general mapping for multilingual lexical resources.

2.3 Sememe prediction

The lack of sememe information in existing multilingual KBs was highlighted by Qi et al. (2020), motivating the sememe prediction task, which aims at bridging the gap between HowNet and BabelNet by predicting a set of most related sememes for a given BabelNet synset.

Qi et al. (2020) propose a model named *Sememe Prediction for BabelNet Synsets* (SPBS), which aims to learn a representation of the input synsets. This supervised model compares the representation of the input synset B to those in the training data, which are labeled with sememes. The input synset is then assigned sememes which are most frequently associated with the synsets that are most similar to it. Intuitively, if B has a similar representation to training synsets associated with some sememe s, then SPBS will be likely to predict sememe s for B. A variant of this method, SPBS-SS, also uses synset relation information from BabelNet to improve synset representations. An ensemble method combines both SPBS and SPBS-SS representations and yields better results.

Qi et al. (2022) propose another method: *Multilingual Synonyms and Glosses as well as Images* (MSGI). This approach uses multimodal information provided by BabelNet synsets, such as images. The model combines a text encoder, image encoder and multi-label classifier.

2.4 Sememes vs. Semantic Primes

While sememes have some similarities to the concept of semantic primitives (or semantic primes), they differ in their goals and representation. The purpose of semantic primes (Wierzbicka, 1996b) is to facilitate comprehensive linguistic analysis by formulating a set of universal basic concepts that can be used to convey the meaning of any linguistic expression (Goddard, 1998). Contrariwise, the purpose of sememes is to provide information about word senses by linking them to a subset of key terms. Sememes need not convey all of the informa-

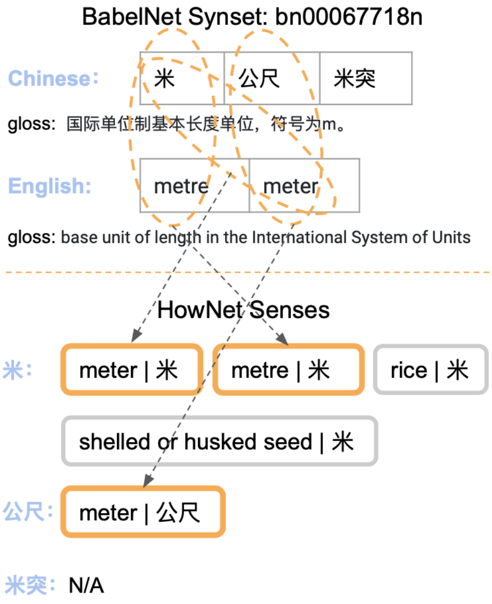


Figure 3: An illustration of the exact match algorithm (Stage 1), in which a BN synset is aligned to three different HowNet senses by comparing translation pairs.

tion about each sense. It may happen, for example, that distinct senses of distinct words may actually be associated with the same set of sememes, as in the case of apple (苹果) and banana (香蕉). While not as comprehensive as semantic primes, sememes are markedly less controversial (Fodor and Garrett, 1975), and are considered sufficient for a resource such as HowNet.

3 Methods

In this section, we introduce our method for the task of alignment between BabelNet and HowNet. Our method has four stages, each relaxing some constraint until an alignment is found. Such an alignment implicitly associates a BabelNet synset with a set of sememes from the aligned HowNet senses. We describe each of the four stages in a separate subsection.

Formally, our task is as follows: Let B be a BabelNet synset. Let $B_Z = \{z_1, \ldots, z_n\}$ be the set of Chinese words in B, and let $B_E = \{e_1, \ldots, e_m\}$ be the set of English words in B. We assume that at least one of B_Z and B_E is non-empty. The output is a set H of one or more HowNet senses that express the same concept as B.

3.1 Stage 1: Exact match

Our first stage uses a strict criterion for aligning a BabelNet synset B with HowNet senses, which is aimed at high precision rather than high coverage (later stages are aimed at improving coverage). It exploits the fact that each HowNet sense is annotated with a lexicalization in both Chinese and English, as well as the well-known observation that distinct senses of a word may translate differently (Gale et al., 1992). For each Chinese-English pair in B, that is, each $(z_i, e_j) \in B_Z \times B_E$, we check if there is a HowNet sense which has the Chinese lexicalization z_i and the English lexicalization e_j. If there is, we add that sense to H. (Figure 3 shows an overview of this stage.)

Once this stage is completed, if H is non-empty, we return it as the set of senses aligned to B. If H remains empty, we continue to Stage 2.

3.2 Stage 2: Word n-gram partial match

Every Chinese word consists of one to four characters; unlike English, each individual character has some meaning associated with it. Therefore, words that share one or more characters often share parts of their meaning. For example, a BabelNet synset for the word “延伸” (stretch) also contains the synonyms “伸长”, “伸展” and “延长”, each of which shares at least one character with “延伸”. As shown in Figure 4, the exact match approach in Stage 1 may miss the semantic correlation between such words, which reduces its coverage. We therefore propose the approach shown in Algorithm 1 to tackle such problems.

As in Stage 1, we again consider each translation pair in $(z_i, e_j) \in B_Z \times B_E$. For each such pair, we look for the HowNet sense with English word e_j, and with the Chinese word that has the maximum number of characters in common with z_i. Specifically, if z_i contains k characters, we first look for a sense which contains e_j and a Chinese word with k-1-gram in common with z_i. If such a sense is found, we add it to H. If not, we instead look for senses which have a k-2-gram in common with z_i. This partial-matching approach is explainable, unsupervised, and efficient; rather than using an uninterpretable and computationally expensive supervised language model, it instead exploits a useful property of the Chinese script. If H remains empty, we proceed to the next stage.

3.3 Stage 3: Sense information matching

Stages 1 and 2 are sufficient to find at least one HowNet sense for roughly 80% of synsets. Stage 3 attempts to map the remaining synsets by exploiting two key properties: (1) many words are

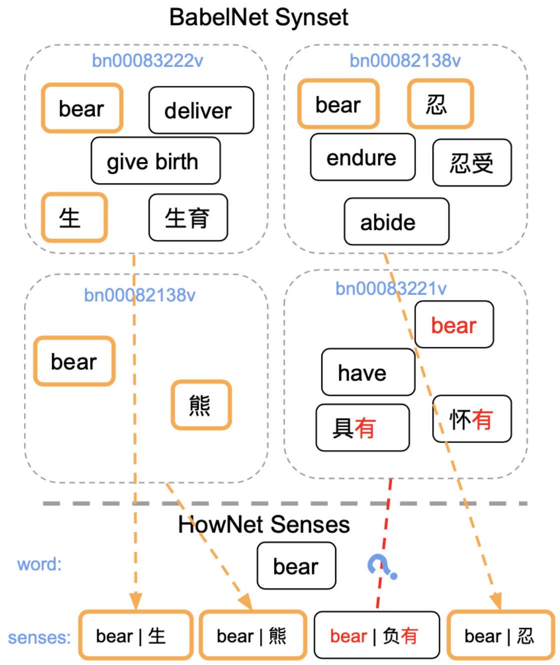


Figure 4: Example of inexact translation pairs during matching. Using exact algorithm, word "bear" has three synsets matched to some HowNet senses, while one synset is not matched despite having the same sense meaning with the unmatched HowNet sense. Note that character "有" means "have" and words "具有", "怀有" and "负有" express the same meaning in context; Stage 2 will successfully map such a synset to a sense in HowNet, as indicated by red dashed line .

monosemous in HowNet; and (2) sememes are represented as words, which can often be found among the hypernyms of a given BabelNet synset B.

If B contains any Chinese or English words that have only one sense in HowNet, such as "ultraviolet" or "haemolytic", then we assume the sense of that word in B must align to its only sense in HowNet. Monosemous words tend to be less frequent in text, but make up a substantial proportion of words in HowNet.[2] If even a single English or Chinese word in B is monosemous in HowNet, this sub-stage will add that sense to H.

If B has still not been aligned with any HowNet sense, we exploit the representation of sememes in HowNet to find an appropriate alignment. Each sememe of each sense is represented by an English word and a Chinese word, typically with a meaning that is more general than that of the senses associated with it. For example, one sense of the English word *teacher* has the following sememes: "human", "occupation", "education", and "teach". Our intuition is that these words will tend to overlap with the hypernyms of the corresponding sense in BabelNet. We therefore add to H any HowNet sense s such that s involves a word in B, and one of the sememes of s is represented by a word in a hypernym synset of B.

BabelNet only provides hypernym relations for nouns and verbs. All other words have no hypernym information, and thus can not be aligned based on sememe and hypernym agreement. We further relax our aligning condition by adding to H the HowNet senses that have any sememe overlap with each other and the English or Chinese words in the given synset. The intuition is that all words in a synset are closely related; if their HowNet senses have similar sets of sememes, we can align them to the given target synset.

3.4 Stage 4: Proper names

In our development experiments, we found that those BabelNet synsets B that could not be aligned with at least one HowNet sense by any of the first three stages were generally proper nouns. Proper nouns or named entities are represented in HowNet by a sense with the English label "ProperName" and an associated set of sememes. Therefore, the fourth and final stage of our method adds this sense to H. As the final result, the algorithm returns the set H, which contains the HowNet senses aligned to a given synset.

4 Experiments

In this section, we describe the experimental evaluation of our sense alignment algorithm. Since we are the first to propose an algorithm for aligning BabelNet and HowNet, there is no gold-annotated dataset for this task that we can use as a test set. To the best of our knowledge, the only dataset that connects BabelNet and HowNet is the BabelSememe dataset, which was constructed for the evaluation of sememe prediction models. Therefore, we evaluate our sense-alignment algorithm extrinsically on the task of sememe prediction.

In the sememe prediction task, given a BabelNet synset B, and a ground set S^* of sememes, we are required to identify a set of sememes $S \subseteq S^*$ which describe the meaning of B. Following prior work, we take the set of all sememes which appear in HowNet as S^*. The output S is then compared to a gold-standard set of sememes for B in the

---

[2] Due to limitations in the OpenHowNet API, we were not able to get the exact percentage.

Algorithm 1 Stage 2

```
Input BabelSynsetID
Output $\mathcal{H}$
$\mathcal{H} \leftarrow \emptyset$
$\mathcal{B} \leftarrow$ synset from BabelSynsetID
$\mathcal{C} \leftarrow$ all English-Chinese word pairs from $\mathcal{B}$
for $\mathrm{r} \in \{1, 2\}$ do
    for each $(\mathrm{e_i}, \mathrm{z_j}) \in \mathcal{C}$ do
        $\mathrm{k} \leftarrow \mathrm{length}(\mathrm{z_j}) - \mathrm{r}$
        $\mathcal{M} \leftarrow$ all k-grams of $\mathrm{z_j}$
        for each $\mathrm{m_i} \in \mathcal{M}$ do
            if $(\mathrm{e_i}, \mathrm{m_j})$ exists in HowNet then
                append HowNet sense to $\mathcal{H}$
            end if
        end for
    end for
    if $\mathcal{H} \neq \emptyset$ then
        return $\mathcal{H}$, end algorithm
    end if
end for
return $\emptyset$
```

BabelSememe dataset.

We compute S by applying our BabelNet-to-HowNet alignment algorithm, and taking the union of all sememes associated with the returned set of HowNet senses. This approach effectively reduces the task of sememe prediction to sense alignment. Since the quality of the alignment will be reflected in the accuracy of sememe prediction, the experimental results can serve as an evaluation of the sense alignment algorithm.

4.1 Experimental setup

In this section, we specify our experimental setup, including data statistics, evaluation metrics, baseline methods, and implementation details.

Data. BabelSememe (Qi et al., 2020) consists of 15,461 BabelNet synsets, each annotated with a set of HowNet sememes which comprise the meaning of the synset. These sememe sets were created manually by more than 100 bilingual (English and Chinese) human annotators. HowNet contains 2,106 sememe types; on average, 2.74 sememes assigned to each BabelNet synset. We use the existing test split, which consists of 10% of the dataset, as our test set. Since our method is unsupervised, we did not use the training or validation splits, instead developing our method on BabelNet alone, without reference to any labeled sememe prediction data.

English	Simplified	Traditional
noodles	面食	麵食
room	房间	房間
weather	天气	天氣
drink	饮	飲

Table 1: Examples in English and Chinese including both simplified and traditional versions.

Evaluation. Following previous work, we adopt mean average precision (MAP) and the F1 score and as our metrics.[3] For a given instance, a classification is positive if the synset is annotated with that sememe, or negative otherwise. MAP takes the weighted mean of the precision of each class, where each sememe is considered a separate class. The F1 score is the harmonic mean of the precision and recall of the predicted labels. Recall is the ratio of sememes correctly predicted to the number of sememes in the gold standard set, while precision is the proportion of sememes correctly predicted to the total number of sememes predicted by the method.

Comparison systems. We compare our sememe prediction results against systems from prior work, including the state of the art. In particular, we compare with three variants of SPBS model proposed by Qi et al. (2020), and five variants of MSGI model proposed by Qi et al. (2022) which we mention in Section 2. Additionally, we compare to *LR-NASARI* (Qi et al., 2020), a logistic regression model trained on NASARI embeddings (Camacho-Collados et al., 2016), and *TransE*, a relational prediction models proposed by (Bordes et al., 2013). In contrast to our unsupervised approach, all of the comparison systems are supervised, and model sememe prediction as a multi-class classification task.

Implementation details. We use the BabelNet Python API[4] with BabelNet 5.0 to retrieve English and Chinese words from synsets. We use OpenHowNet API[5] for retrieving HowNet senses and their corresponding translation pairs (Qi et al., 2019). One technical issue we encountered is the use of both simplified and traditional Chinese characters in BabelNet, likely due to the use of multiple heterogeneous resources in the construction

---

[3] We use the evaluation script provided by Qi et al. (2022).

[4] `https://babelnet.org/`

[5] `https://github.com/thunlp/OpenHowNet`

Methods	**Noun**		**Verb**		**Adj**		**Adv**		**Overall**	
	MAP	**F1**	**MAP**	**F1**	**MAP**	**F1**	**MAP**	**F1**	**MAP**	**F1**
LR-NASARI (Qi et al., 2020)	54.54	39.81	–	–	–	–	–	–	–	–
TransE (Bordes et al., 2013)	61.05	46.78	34.75	26.7	29.11	22.99	30.05	20.69	51.73	39.73
SPBS-SR (Qi et al., 2020)	65.16	49.75	–	–	–	–	–	–	–	–
SPBS-RR (Qi et al., 2020)	62.50	47.92	34.76	25.28	32.68	24.51	30.86	20.07	57.64	45.61
Ensemble (Qi et al., 2020)	68.85	55.35	34.76	25.28	32.68	24.51	30.86	20.07	57.64	45.61
MSGI -Synonym (Qi et al., 2022)	67.40	59.07	35.31	24.99	36.33	26.18	48.33	37.45	57.25	48.54
MSGI -Glosses (Qi et al., 2022)	66.90	56.99	54.22	41.54	53.11	39.20	**68.76**	55.14	62.67	52.21
MSGI -Image (Qi et al., 2022)	71.41	61.58	59.70	44.29	55.86	43.15	63.81	51.63	67.13	56.62
MSGI -MSCP (Qi et al., 2022)	70.58	61.99	57.55	43.27	52.57	40.61	68.49	52.79	65.70	56.05
MSGI (Qi et al., 2022)	71.81	64.36	**59.78**	47.01	55.61	41.02	68.52	55.20	67.23	57.68
Ours	**75.63**	**72.63**	57.70	**56.53**	**66.57**	**69.35**	64.63	**62.98**	**71.49**	**69.69**

Table 2: Main results on BabelSememe test set. Numbers in bold font represent the highest value in the column.

Data	**Noun**		**Verb**		**Adj**		**Adv**		**Overall**	
	MAP	**F1**	**MAP**	**F1**	**MAP**	**F1**	**MAP**	**F1**	**MAP**	**F1**
Train	75.73	72.95	55.33	55.85	64.12	66.08	68.08	69.87	70.71	69.29
Valid	74.75	73.16	57.78	57.21	69.70	71.12	69.33	70.36	71.48	70.59
Test	75.63	72.63	57.70	56.53	66.57	69.35	64.63	62.98	71.49	69.69

Table 3: Results of our method on the training, validation, and test sets.

of BabelNet. Table 1 shows some examples contrasting simplified and traditional characters. Since HowNet contains simplified Chinese characters exclusively, we use the Python `zhconv` library to convert all traditional Chinese characters into their simplified versions.

4.2 Results

The results on the BabelSememe test set are shown in Table 2. Following previous work, we report results for nouns, verbs, adjective, adverbs, as well as for all words. Our overall results outperform all prior work according to both metrics, establishing our method as the new state of the art for sememe prediction. In particular, the improvement in overall F1 score is 12% over the previous supervised state-of-the-art method, which was designed specifically for this task. These results are remarkable considering that our method is not supervised, and demonstrate its efficacy and utility.

In order to assess the generality of our method, we also report the results on the training and validation splits of the BabelSememe dataset, which are otherwise unused by our method. Table 3 shows that our results on these additional splits do not differ substantially from our results on the test set. This provides strong evidence that our method is consistent and reliable.

4.3 Stage analysis

As described in Section 3, our method consists of a sequence of four stages. In this section, we conduct additional experiments on the test set by examining the F1, MAP, and synset coverage rate (percentage of synsets that are mapped to at least one HowNet sense) of our system after every stage.

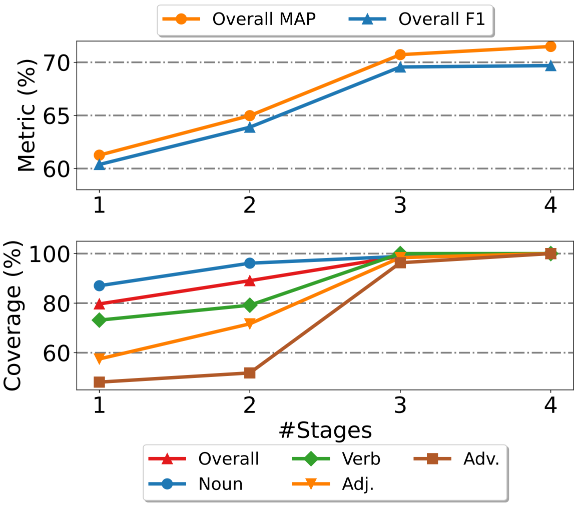


Figure 5: Experimental results for stages 1 to 4.

The results are shown graphically in Figure 5 and numerically in Table 5. We see an increasing trend in F1, MAP, and synset coverage rate with respect to the number of stages used. Stage 1 is

Stage	BableNet			HowNet	
	Synset ID	Synset EN	Synset ZH	Aligned Senses	Aligned Sememes
1	bn:00028111n	doll, dolly	洋娃娃, 玩偶	doll | 洋娃娃 doll | 玩偶	tool | 用具, recreation | 娱乐
2	bn:00002393n	alabaster, alabastar, gypsum	雪花石膏, 汉白玉	gypsum | 石膏	material | 材料, tool | 用具, medical | 医
3	bn:00040267n	geranium, cranesbil	天竺葵, 洋绣球	fish pelargonium | 天竺葵	FlowerGrass | 花草, medicine | 药物
4	bn:00048483n	Judas Iscariot, Judas	加略人犹大	ProperName | 专	ProperName | 专

Table 4: A case study of the alignment in each stage.

sufficient to cover roughly 80% of synsets, and achieve roughly 60% F1, but has particular difficulty finding HowNet senses for adjective and adverb synsets. Adding stages 2 and 3 greatly increases coverage of adjectives and adverbs respectively, with concomitant increases in F1 and MAP. Stage 4 is shown to provide marginal improvement, as almost all synsets can be assigned at least one sememe after stage 3.

4.4 Alignment analysis

To provide additional insight into the relationship between BabelNet and HowNet, we analyzed various properties of the alignment produced by our method. Theoretically, our algorithm could produce many-to-many alignments, with one BabelNet synset aligned to multiple HowNet senses, and vice versa. In practice, the average number of HowNet senses aligned to a given BabelNet synset in the BabelSememe test set is 3.99. This suggests that a single BabelNet concept is often represented by multiple HowNet senses, each of which may be labeled with different sememes. On the other hand, less than 1% of the HowNet senses were aligned to multiple BabelNet synsets (excluding the "ProperName" cases from Stage 4). This suggests that HowNet senses are no more fine grained than BabelNet synsets; in other words, if a sense distinction is made in BabelNet, it is likely made in HowNet as well.

We also analyzed the alignments for each part of speech and for each stage in our method. In Figure 6, we see that verb synsets are aligned to more senses than any other part of speech at every stage. For all parts of speech except adverbs, Stage 2 produces more mappings than any other stage. Since Stage 2 works by allowing partial matches in Chinese words (i.e. an alignment can be made on the bases of a single character shared by two Chinese words with two characters each), this suggests that the two knowledge bases often contain different Chinese words for a given concept. However, our method is still able to align them by identifying shared characters. If a sense reaches Stage 4, it will be assigned exactly one HowNet sense, the "ProperName" sense.

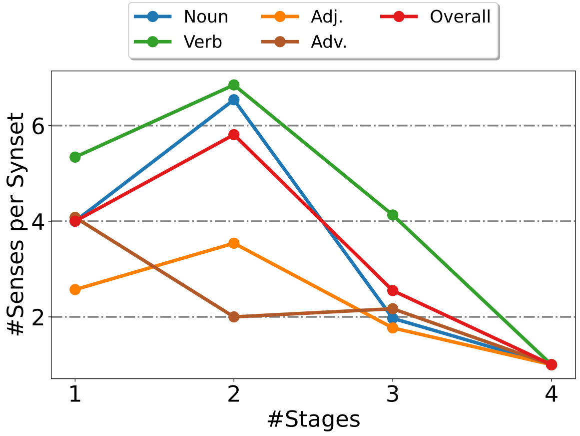


Figure 6: The average number of HowNet senses matched to each BabelNet synsets in each stage on the BabelSememe test set.

Table 4 shows example alignments from each stage. Row 1 shows that Stage 1 of our algorithm is able to correctly align the synset containing the words "doll" and "dolly" to the corresponding HowNet sense of each word. We can then use this alignment to retrieve the sememes "tool" and "recreation" for this synset. In Row 2, the synset containing the word "gypsum" does not contain any Chinese words which have HowNet senses. However, Stage 2 correctly finds a correct alignment between the BabelNet synset containing "雪花石膏" and a HowNet sense for "石膏". In Stage 3, the word "天竺葵" has only one sense in HowNet. We therefore map the BabelNet synset to that HowNet sense. Lastly, the word "Judas" has no senses in HowNet, therefore it is mapped to HowNet's "ProperName" sense in Stage 4.

POS	**Stage 1**		**Stage 2**		**Stage 3**		**Stage 4**	
	MAP	**F1**	**MAP**	**F1**	**MAP**	**F1**	**MAP**	**F1**
Noun	71.18	68.89	73.40	70.80	75.14	72.49	75.63	72.63
Verb	44.23	45.62	48.10	48.72	56.16	56.53	57.70	56.53
Adj.	37.30	40.06	47.43	50.95	65.37	69.15	66.57	69.35
Adv.	28.91	31.38	31.38	33.84	63.37	62.98	64.63	62.98
Overall	61.26	60.38	64.98	63.89	70.72	69.56	71.49	69.69

Table 5: Experimental results after each stage.

5 Conclusion

We have presented a novel unsupervised method for aligning two lexical-semantic knowledge bases, BabelNet and HowNet. The results of our experiments on leveraging the sense alignment for the task of sememe prediction demonstrate that our algorithm is highly effective, yielding substantially better results than state-of-the-art supervised systems designed specifically for this task. In the future, we would like to leverage sense alignment for other semantic tasks, including word sense disambiguation.

Acknowledgements

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Alberta Machine Intelligence Institute (Amii).

Limitations

Our proposed algorithm only works with synsets that contain at least one Chinese or English word. Although this condition is satisfied for a majority of BabelNet synsets, there remain some multilingual synsets that could not be aligned. In addition, a intrinsic evaluation of the produced alignment could not be performed because of the lack of an existing gold data set.

References

Hongchang Bao, Bradley Hauer, and Grzegorz Kondrak. 2022. Lexical resource mapping via translations. In *Proceedings of the Language Resources and Evaluation Conference*, pages 7147–7154, Marseille, France. European Language Resources Association.

Leonard Bloomfield. 1926. A set of postulates for the science of language. *Language*, 2(3):153–164.

Francis Bond and Ryan Foster. 2013. Linking and extending an open multilingual Wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1352–1362, Sofia, Bulgaria. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence*, 240:36–64.

Marine Carpuat, Grace Ngai, Pascale Fung, and Kenneth Church. 2002. Creating a bilingual ontology: A corpus-based approach for aligning wordnet and hownet. In *Proceedings of the 1st Global WordNet Conference*, pages 284–292. Citeseer.

Benfeng Chen and Pascale Fung. 2004. Automatic construction of an English-Chinese bilingual FrameNet. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 29–32, Boston, Massachusetts, USA. Association for Computational Linguistics.

Christian Chiarcos, John McCrae, Philipp Cimiano, and Christiane Fellbaum. 2013. Towards open data for linguistics: Linguistic linked data. *New trends of research in ontologies and lexical resources: Ideas, projects, systems*, pages 7–25.

Zhendong Dong and Qiang Dong. 2003. Hownet-a hybrid language and knowledge resource. In *International conference on natural language processing and knowledge engineering, 2003. Proceedings. 2003*, pages 820–824. IEEE.

Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Word sense disambiguation through sememe labeling. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 1594–1599, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jerold A Fodor and Merrill F Garrett. 1975. The psychological unreality of semantic representations. *Linguistic Inquiry*, 6(4):515–531.

William A Gale, Kenneth W Church, and David Yarowsky. 1992. Using bilingual materials to develop word sense disambiguation methods. In *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation*, volume 112. Citeseer.

Cliff Goddard. 1998. Bad arguments against semantic primitives.

Yihong Gu, Jun Yan, Hao Zhu, Zhiyuan Liu, Ruobing Xie, Maosong Sun, Fen Lin, and Leyu Lin. 2018. Language modeling with sparse product of sememe experts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4642–4651, Brussels, Belgium. Association for Computational Linguistics.

Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. 2012. UBY - a large-scale unified lexical-semantic resource based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 580–590, Avignon, France. Association for Computational Linguistics.

Boer Lyu, Lu Chen, Su Zhu, and Kai Yu. 2021. Let: Linguistic knowledge enhanced graph transformer for chinese short text matching. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13498–13506.

John P. McCrae and David Cillessen. 2021. Towards a linking between WordNet and Wikidata. In *Proceedings of the 11th Global Wordnet Conference*, pages 252–257, University of South Africa (UNISA). Global Wordnet Association.

Christian M Meyer and Iryna Gurevych. 2011. What psycholinguists know about chemistry: Aligning wiktionary and wordnet for increased domain coverage. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 883–892.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial intelligence*, 193:217–250.

Yilin Niu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Improved word representation learning with sememes. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2049–2058, Vancouver, Canada. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Roberto Navigli. 2014. A robust approach to aligning heterogeneous lexical resources. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 468–478.

Fanchao Qi, Liang Chang, Maosong Sun, Sicong Ouyang, and Zhiyuan Liu. 2020. Towards building a multilingual sememe knowledge base: Predicting sememes for babelnet synsets. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8624–8631.

Fanchao Qi, Chuancheng Lv, Zhiyuan Liu, Xiaojun Meng, Maosong Sun, and Hai-Tao Zheng. 2022. Sememe prediction for BabelNet synsets using multilingual and multimodal information. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 158–168, Dublin, Ireland. Association for Computational Linguistics.

Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Qiang Dong, Maosong Sun, and Zhendong Dong. 2019. Openhownet: An open sememe-based lexical knowledge base. *arXiv preprint arXiv:1901.09957*.

Fanchao Qi, Yuan Yao, Sophia Xu, Zhiyuan Liu, and Maosong Sun. 2021. Turn the combination lock: Learnable textual backdoor attacks via word substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4873–4883, Online. Association for Computational Linguistics.

Virach Sornlertlamvanich, Canasai Kruengkrai, Shisanu Tongchim, Prapass Srichaivattana, and Hitoshi Isahara. 2005. Term-based ontology alignment. In *Proceedings of the Second International Workshop on UNL, Other Interlinguals and their Applications, Mexico City, Mexico*. Citeseer.

Anna Wierzbicka. 1996a. *Semantics: Primes and universals: Primes and universals*. Oxford University Press, UK.

Anna Wierzbicka. 1996b. *Semantics: Primes and universals: Primes and universals*. Oxford University Press, UK.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080, Online. Association for Computational Linguistics.

Xiang Zhang, Bradley Hauer, and Grzegorz Kondrak. 2022. Improving HowNet-based Chinese word sense disambiguation with translations. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4530–4536, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuntao Zhang, Ling Gong, and Yongcheng Wang. 2005. Chinese word sense disambiguation using hownet. In *Advances in Natural Computation*, pages 925–932, Berlin, Heidelberg. Springer Berlin Heidelberg.

The StatCan Dialogue Dataset: Retrieving Data Tables through Conversations with Genuine Intents

Xing Han Lu [1 2†] **Siva Reddy** [1 2 3] **Harm de Vries** [2]
[1] Mila Quebec AI Institute, McGill University
[2] ServiceNow Research
[3] Facebook CIFAR AI Chair
statcan.dialogue.dataset@mila.quebec

Abstract

We introduce the StatCan Dialogue Dataset[1] consisting of 19,379 conversation turns between agents working at Statistics Canada and online users looking for published data tables. The conversations stem from genuine intents, are held in English or French, and lead to agents retrieving one of over 5000 complex data tables. Based on this dataset, we propose two tasks: (1) automatic retrieval of relevant tables based on a on-going conversation, and (2) automatic generation of appropriate agent responses at each turn. We investigate the difficulty of each task by establishing strong baselines. Our experiments on a temporal data split reveal that all models struggle to generalize to future conversations, as we observe a significant drop in performance across both tasks when we move from the validation to the test set. In addition, we find that response generation models struggle to decide when to return a table. Considering that the tasks pose significant challenges to existing models, we encourage the community to develop models for our task, which can be directly used to help knowledge workers find relevant tables for live chat users.

U_1:	Hi, I'm looking to obtain quarterly data in regards to GDP grow (Canada), BC Housing STarts, Canada Oil Price/BBL
A_1:	Hello, my name is Kelly C. Give me one moment as I search [...]
A_1:	For GDP growth rates, please consult the following link: [...]
A_1:	What do you mean by BC Housing Starts?
U_2:	I'm required to research all of the housing starts for BC on a quarterly basis [...]
U_2:	Housing starts are the number of new residential construction projects that have begun during any particular month [...]
A_2:	I would have monthly data regarding new building permits being issued [...]
A_2:	**Building permits, by type of structure and type of work: https[...]**
A_2:	I'll have a look for oil prices. One moment.
U_3:	Do you also have data to Canada'as oil Price/BBL ("WTI")? [...]
A_3:	Are you looking for the retail prices of oil?
A_3:	If so, I found some data for smaller geographies.
A_3:	**Monthly average retail prices for gasoline and fuel oil, by geography (https[...])**
A_3:	[...] Would those geographies be enough?
A_3:	Or are you looking for Canada only?
U_4:	[...] I would need something that pertains more to all of canada
A_4:	What about this? **Monthly average retail prices for food and other selected products (https[...])**

Table 1: An example of the StatCan Dialogue Dataset in which a user (U) talks to a StatCan agent (A) to find a number of data tables. Text in **bold** indicates the title of a table retrieved by the agent.

1 Introduction

One of the longstanding goals in Natural Language Processing (NLP) is to develop conversational agents that assist people with concrete tasks, such as finding information in large collections of documents or booking restaurants and hotels. To aid the development of such virtual assistants, the research community is in need of benchmarks that reflect the intents and linguistic phenomena found in real-world applications. However, developing such real-world conversational datasets is challenging in the current research landscape. On the one hand, academic labs often struggle to come up with natural use cases of task-oriented dialogue agents and collect conversations with a large number of real users. Many labs have designed artificial tasks and collected conversations from crowd workers with simulated intents (Budzianowski et al., 2018; Adlakha et al., 2022; Lee et al., 2022), often leading to datasets that do not capture the linguistic challenges of production settings (de Vries et al., 2020). On the other hand, industry labs might have access to users with genuine intents (e.g., through Siri or Alexa) but rarely release such conversational

---

[†] Work done as visiting researcher at ServiceNow Research
[1] Website: mcgill-nlp.github.io/statcan-dialogue-dataset

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2799–2829
May 2-6, 2023 ©2023 Association for Computational Linguistics

Dataset	**Intent**	**Dialogue**	**Query**	**Result(s)**	**Source(s)**	**Lang.**
Our work	Genuine	✓	Question, Request	Table link, Dial. act	StatCan	En, Fr
NQ (2019)	Mixed	✗	Question	Span excerpt	Google, Wiki.	En
DuReader (2018)	Mixed	✗	Question	Span excerpt	Baidu	Zh
OTT-QA (2020)	Simulated	✗	Question	Table/Span excerpt	Wikipedia	En
TAPAS-NQ (2021)	Mixed	✗	Question	Table excerpt	Google, Wiki.	En
CoQA (2019)	Simulated	✓	Question	Span excerpt*	Multiple	En
QuAC (2018)	Simulated	✓	Question	Dial. act	Wiki.	En
ATIS (1990)	Genuine	✓	Request	SQL query, Command	TI Corp.	En
SGD-X (2022)	Simulated	✓	Request	API call, Dial. act	Dial. Simulator	En

Table 2: Comparison with related datasets (see Section 2). (*) CoQA uses rationales to support extracted answers.

datasets due to their commercial value and user privacy concerns. Hence, we argue that the research community would benefit from a task-oriented dialogue environment where findings can be validated with real users, and, to that effect, present a unique dataset in collaboration with Statistics Canada.

Statistics Canada (StatCan) is a national statistics agency commissioned with collecting key information on Canada's economy, society, and environment. Statistics Canada conducts hundreds of surveys on virtually all aspects of Canadian life and publishes the resulting data tables on statcan.gc.ca. This website currently features 5K+ of such complex and often large data tables. Canadian citizens—and other interested individuals—come to this website to find the statistics they are looking for. The StatCan website offers a chat functionality (available in English and French) to help users in case they can not find the appropriate information.

Sourcing from these live chats, we present the StatCan Dialogue Dataset, a collection of 20K+ English and French conversations between visitors of statcan.gc.ca and agents working at Statistics Canada. Before releasing this dataset, StatCan has ran several procedures to remove Personally Identifiable Information (PII). While we observe a wide variety of user intents, ranging from table manipulation to navigation instructions, a large number of visitors use the chat functionality to find data tables on the StatCan website. Specifically, we observe 6.6K instances where agent returns a link to a data table across 4.4K conversations. In Table 1, we provide an example conversation in which an online user is looking for specific data tables.

In this work, we develop two novel tasks centered on helping users find specific tables. First, we introduce the table retrieval task, which requires a model to predict the table returned by the agent given the messages sent so far. Second, we introduce the response generation task, which requires a model to predict the agent's response given the dialogue history. For both tasks, we investigate its difficulty by establishing strong baselines and evaluating them on various metrics.

We stress that both tasks are immediately useful in a real-world setting. The table retrieval task can help agents find relevant tables faster while the response generation task may lead to a virtual agent that can return relevant tables through an online conversation. We hope that this tight connection with a real-world scenario will bring the research community more insight into the challenges of developing practical dialogue agents and lead to faster transfer of research ideas and findings.

2 Related Work

This section presents various directions related to our work. See Table 2 for a comparative summary.

Open-domain QA This is the task of answering questions using a large and diverse collection of text documents. One of the first large-scale evaluations in open-domain QA was presented at TREC-8 (Voorhees, 2001). Since then, many studies have released large-scale open-domain QA datasets: WikiQA (Yang et al., 2015) and MS MARCO (Bajaj et al., 2018) source questions from the Bing search engine, Natural Questions (NQ) (Kwiatkowski et al., 2019) from Google search, and DuReader (He et al., 2018) source questions in Chinese from Baidu. The questions come from real users and the answers are collected from the search results through crowd workers. Although those datasets have questions with genuine intent and the answer must be retrieved from a collection of documents, our dataset emphasizes the retrieval of tables (in a conversational setting) rather than free-form documents.

Table retrieval and QA Following works on tabular pre-training (Yin et al., 2020), table-to-text generation (Parikh et al., 2020) and weak supervision for semantic parsing (Herzig et al., 2020),

Dataset	Train	Valid	Test	All
# Conv.	2573	545	557	3675
# Turns	11382	2339	2600	16321
# Messages	36147	7385	8340	51872
# Queries	3782	799	870	5451
# Tokens / Msg	32.83	33.51	29.32	32.36
# Turns / Conv.	4.42	4.29	4.67	4.44
# Msg / Conv.	14.05	13.55	14.97	14.11
# Queries / Conv.	1.48	1.47	1.57	1.49
# Tables	778	349	388	959
# New tables	0	41	145	181
# Dims / Table	3.5	3.5	3.6	3.6
# Mbrs / Table	185.5	210.8	175.6	172.1
# Notes / Table	21.1	22.7	23	20.4

Table 3: Statistics of English conversations and tables in the retrieval and generation tasks. New tables are calculated with respect to training set (see Table 11).

Chen et al. (2020) and Herzig et al. (2021) respectively propose OTT-QA and TAPAS-NQ, two novel approaches that extend open-domain QA to retrieving tables instead of documents. The former collects both the questions and answers from crowd workers and the latter extends Natural Questions by using tables from the article where the answer was taken. In both cases, the tables being retrieved are sourced from Wikipedia articles. Although our data also incorporate tabular retrieval, the tables are sourced from statcan.gc.ca, they can be significantly larger (as discussed in Appendix A.2.1), and they are being retrieved in an interactive and conversational setting.

Conversational QA Several works extended question answering to the conversational setting. CoQA (Reddy et al., 2019) and QuAC (Choi et al., 2018) introduced datasets in which multiple rounds of questions are asked about a reference passage taken from a document (such as a Wikipedia article). Subsequent works extended this setup to an open-domain setting where the reference passage is not known beforehand (Qu et al., 2020; Anantha et al., 2021; Adlakha et al., 2022). Saeidi et al. (2018) proposed a conversational QA task about regulatory texts. Aforementioned datasets are all structured in the same way: at every turn, the first speaker will ask a question, and the other speaker will give an answer. In contrast, the queries in our conversations are not restricted to questions, and the answers can be either a table, metadata, or a dialogue act.

Task-oriented Dialogue Our work is related to work on task-oriented dialogue where users converse with virtual agents to accomplish specific goals, such as booking a restaurant or resolving a customer issue. While early work has collected a dataset in a genuine information seeking setup (Hemphill et al., 1990), many recent datasets has collected them through a simulated setup (Budzianowski et al., 2018; Rastogi et al., 2020; Feng et al., 2020, 2021; Chen et al., 2021; Lee et al., 2022). Task-oriented models usually track the dialogue state by predicting dialogue acts that are specified through intents and slot-value pairs, e.g., `findRestaurants(cuisine=Italian)`. While our dataset does not provide turn-based annotations, the released conversations come with an annotated goal i.e., which data table the user was looking for. Like other goal-oriented dialogue tasks, this annotation enables us to automatically evaluate the dialogue models through a task completion metric.

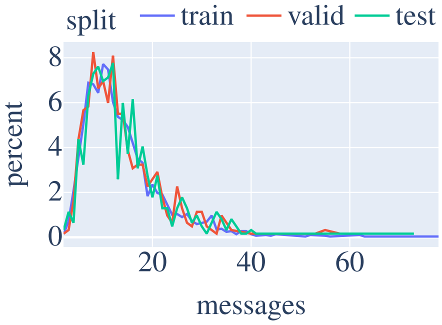


Figure 1: Histogram of messages by conversation in the both tasks (French split in Figure 7).

Chit-chat Dialogue The goal for chit-chat systems is to engage in a open-ended conversation with an end-user (Lowe et al., 2015; Dinan et al., 2018). Unlike our dataset, such conversations do not intend to assist the user with a specific task.

3 Dataset

The StatCan Dialogue Dataset consist of conversations collected from the live chat between March 1, 2019 till March 8, 2021. Although a variety of user intents can be found in the broader dataset of over 25K conversations, we focus on a single intent by selecting all conversations where the agent returns a data table. We use this subset to develop and test models for the two tasks that we introduce in Section 4. In Section 3.1, we provide basic statistics about this subset of the data and present a dialogue analysis for a small number of conversations in Section 3.2. In Section 3.3, we turn our attention to the

Merged Acts (*Example*)	**%**
Answer (*You can obtain on our...*)	50
Request (*...please help me retrieve data...*)	31
Time Mgmt (*Please hold*)	28
Inform (*Please take note that...*)	63
Info Seeking Ques. (*Do you have any other...?*)	25
Promise (*...please contact the Education Ministry...*)	18
Auto Feedback (*Sure*)	25
Offer (*...how may I help you?*)	11
Instruct (*Select at least one...*)	18
Clarif. Ques. (*Which of these lines would direct...*)	16
Greeting (*Hi*)	28
Self Introduction (*My name is...*)	17
Thanking (*Thanks a lot!*)	47
Accept Thanking (*you're welcome*)	15

Table 4: Frequency of merged speech acts occurring in 100 turns in conversations from the validation set.

data tables and explain what kind of information is available for them. Finally, we explain how the dataset is split into a train, validation, and test set in Section 3.4. For technical specifications, a dataset card is provided in Appendix B.

User intents The live chat was designed to fulfill specific user intents. The main intent of the chat functionality is to help users **find specific data tables**. For example, in Table 1, the agent helps the user find tables about building permit, gasoline price, and retail prices for food. Although, users might also be interested in **obtaining meta-information**, receive help in **manipulating** a table or with the **user interface**. In some cases, the user will make **out of domain requests**. Those auxiliary intents are described in Appendix A.1 since the focus of this work is on the main intent.

Messages and turns Each conversation is broken down in **turns**, which is a pair of user-agent responses. Each response can have multiple **messages** sent sequentially (e.g., in Table 1, the first agent response contains 3 back-to-back messages).

3.1 Basic statistics

In total, 25397 conversations will be made available. Based on our main intent, we focus on a subset of 4468 (3675 in English and 793 in French). Out of a total of 5907 tables available in both English and French, the agents returned 959 unique tables in English 285 in French. The number of messages by conversation varies between 2 and 78 with a median of 12 for the English split (see Figure 1 for the distribution). Based on Table 3, there's on average 4.4 turns but 14.12 messages per conversation, with over 30 tokens for each message. This indicates that the speakers will express multiple sequential thoughts before the addressees respond. For the French split, we analyzed the basic statistics in Appendix A.5.

Title: Production and value of maple products

Date range: 1924-01-01 to 2020-01-01
Dimensions: Geography, Maple products
Subject: Agriculture
Survey: Maple Products
Frequency: Annual

Table 5: Basic information (including title) of table in Section 3.3. Full version in Table 20. This can be accessed at doi.org/10.25318/3210035401-eng.

Frequently requested tables In total, 6 tables make up 13.4% of tables retrieved, covering subjects like inflation and household spending. Supplementary details can be found in Appendix A.3.1.

3.2 Dialogue Analysis

We categorize 100 turns (306 messages) from 24 conversations in the English validation set according to the speech acts defined by Bunt et al. (2010, 2020), which is also known as ISO standard 24617-2. We follow their taxonomy but merge some fine-grained acts with their broader concepts (e.g., *correction, agreement, disagreement* with *inform*). We present the speech act frequencies and examples in Table 4. See the Appendix for more information on how we merged the original acts and supplementary examples (Table 15 and Table 22, respectively).

We notice that *answers* appear twice as frequent as *information seeking questions* because an interlocutor may provide an answer to both *clarification questions* and *requests*. Additionally, *inform* acts appears 63% of the time because agents need to expand upon their answer and users tend to clarify their initial requests by informing the other. Although less frequent, *auto feedback* and *time management* are still relevant because interlocutors cannot rely on visual feedback like nodding or typing. Naturally, *time management* often co-occurs with *promises* because the agent tends to put the user on hold while promising to fulfill their request.

3.3 Table specifications

To explain the specifications, we examine a sample table with **title** *Production and value of maple products* (shown in Table 5). The table has two **dimensions**, which are groups of **member items**;

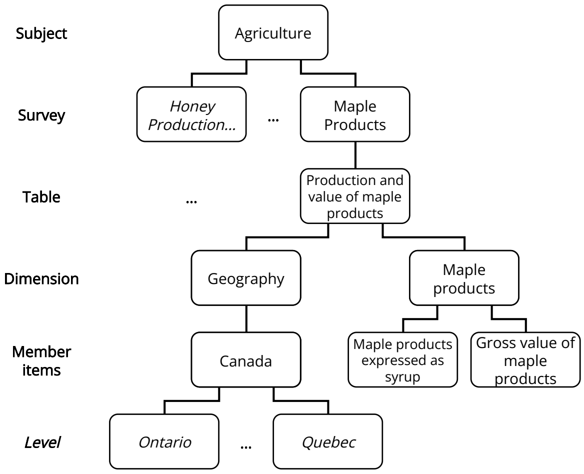


Figure 2: Diagram of the hierarchical relationship between metadata components, discussed in Section 3.3.

for *geography*, the members are the provinces producing maple syrup (Quebec, Ontario, etc.), and for *Maple products* the members are the production types (maple sugar, syrup, taffy and butter). A member item generalizes the concept of rows and columns as they are interchangeable via pivoting. Sometimes, details about those members are provided as **footnotes** at the end the page.

Basic Information This is the core metadata and consists of the title, dimensions, subject, survey and update frequency (member items are excluded). Supplementary details are in Appendix A.2.2.

Hierarchical relation The metadata can be viewed hierarchically. As shown in Figure 2, each subject encompasses different surveys, each survey can be used to generate one or more tables, and so on. A member item that can be nested under another member item is called *Level*.

3.4 Dataset splits

We group the conversations into a train (70%), a validation (15%) and a test (15%) set. The test set was specifically selected to be the most recent conversations by date (covering Sept 8, 2020 to Mar 8, 2021), whereas the training and validation set were randomly selected from the remaining data (covering Mar 1, 2019 to Sept 8, 2020). This lets us test a model's capability to adapt to temporal shifts in the data (such as new data releases and novel events). This is useful to understand a model's capability to generalize beyond the training distribution, but it is also a better reflection of real-world applications of a model (which will be used for future data). The same splits are used for all tasks.

4 Tasks

Based on the conversational and tabular data, we propose two tasks: (i) a table retrieval task, which requires a model to use a partial conversation to predict the table an agent will return, and (ii) a response generation task, which requires a model to use a partial conversation to generate the most probable response by the agent. The conversations in the tasks are available in both English and French.

4.1 Retrieval task

For this task, we truncate every conversation right before a link to a relevant table is shared by the agent. As a result, the product ID (PID) corresponding to that link becomes the objective of the retrieval task, as shown in Table 6. When the agent shares multiple non-repeating PIDs within a conversation, each unique occurrence is treated as a separate sample.

Recall@k To evaluate models for retrieval, we compute the recall at k (R@k) score for $k \in \{1, 10, 20\}$, which corresponds to the rate where the correct table is among the k tables retrieved by the model (usually ranked by a relevance score). We choose $k = 1$ for real-time automatic retrieval and $k \in \{10, 20\}$ for scenarios where humans or automatic rerankers would like to use the retriever to query tables and select the best option.

4.2 Response generation task

In the first task, only the messages leading to a table retrieval are considered. For this task, each message sent by an agent is considered as a target and everything before is the source. Thus, the goal of this task is to use the source text to generate a response that matches the target (see Table 7). Since dialogue responses are challenging to evaluate, we report a wide variety of metrics for this task.

ROUGE-L and METEOR ROUGE-L (Lin, 2004; Lin and Och, 2004) is a common text evaluation metric which naturally takes into account sentence level structure by identifying the longest overlapping word sequence between two sentences. METEOR (Banerjee and Lavie, 2005) is a word-level precision and recall scoring method that encompasses different ways to represent a word, including stems and synonyms.

BERTScore and MoverScore Various methods were developed to leverage contextual embeddings

	Source text (on-going conversation)
	[...]
A:	What do you mean by BC Housing Starts?
U:	I'm required to research all of the housing starts for BC on a quarterly basis[...]
U:	Housing starts are the number of new residential construction projects that have begun during any particular month
A:	[...] I would have monthly data regarding new building permits being issued.
	Retrieval target (StatCan table)
	Table 34-10-0066 (Building permits, by type of structure and type of work)

Table 6: Source and targets of the retrieval task, based on Table 1. Given the on-going conversation, the goal is to retrieve a StatCan table.

from BERT (Devlin et al., 2019) to evaluate similarity between two sentences. BERTScore (Zhang et al., 2019) computes the cosine similarity at the token level, whereas MoverScore (Zhao et al., 2019) computes the earth mover distance (Rubner et al., 2000) at the word or sentence level, thus capturing the cost of transforming the distribution of the generated responses into the distribution of the original responses.

Title accuracy In addition to the general metrics for text generation, we also explored this task-specific metric. We define it as the proportion of generated messages that contain the title of a table shared in the reference messages. Consequently, this metric only includes turns where a table is shared by an agent. To compute this, we (i) find the product ID in the reference message, (ii) look up the title, (iii) check if that title appears exactly in the generated and reference text.

5 Models

To help understand the performance of finetuned models on our tasks, this section presents an overview of the methods for the English splits, whereas the implementation details are covered in Appendix A.4. Similar architectures were used for French (described in Appendix A.5).

5.1 Retrieval

BM25 We use Robertson and Zaragoza (2009)'s algorithm to retrieve the metadata of a table (passage) similar to a given query by weighting the *idf*-scaled term frequency of query words with respect to the passages.

	Source text (on-going conversation)
U:	Hi, I'm looking to obtain quarterly data in regards to GDP grow (Canada), BC Housing STarts, Canada Oil Price/BBL
A:	Hello, my name is Kelly C. Give me one moment as I search [...]
A:	For GDP growth rates, please consult [...]
U:	I'm required to research all of the housing starts for BC on a quarterly basis [...]
	Generation target (next response by agent)
A:	I would have monthly data regarding new building permits being issued. [...]

Table 7: Source and targets of the response generation task, based on Table 1. Given the on-going conversation, the goal is to generate the agent's response.

DPR Proposed by Karpukhin et al. (2020), Dense Passage Retrieval (DPR) is a pair of transformer models that separately encode a query and a passage, and the dot product of the resulting vectors will have a higher score if the passage is relevant to the query. We finetune this model to retrieve the metadata of a table (passage) given the on-going conversation (query).

TAPAS and TAPAS-NQ Herzig et al. (2020) introduced a model that learned to encode flattened tables cells in a self-supervised manner during pretraining. We finetuned it to retrieve the truncated content of a table given an on-going conversation. Subsequently, Herzig et al. (2021) finetuned TAPAS to perform open-domain table retrieval on 12K questions-answer-table triplets extracted from NQ; we further finetune this variant in the same way and report the results as TAPAS-NQ.

Exploring table representation In the simplest scenario, only the title is given to BM25 and DPR. Moreover, we evaluate variants that encode the basic information, member items, footnotes, or a combination of them. For TAPAS and TAPAS-NQ, we also finetuned a variant that retrieves the title, dimensions and member items, since the original TAPAS could attend titles and column names.

5.2 Response generation

T5 We finetuned the large variant of T5 (Raffel et al., 2020) (named *No aug.* in Table 10) to autoregressively decode the target (agent reply) after first encoding the source (on-going conversation).

Augmenting T5 with top-k title(s) For every partial conversation, we use DPR (basic+member) to retrieve the top-k tables (where $k \in \{1, 5\}$), and append their titles to the partial conversation. This

Metadata	R@1	R@10	R@20
Basic	14.7	45.0	55.0
Basic + member	15.7	46.2	56.3
Basic + footnotes	13.9	44.4	54.2
Member	10.7	35.0	46.3
Title	13.9	43.8	53.4

Table 8: Retrieval results of DPR for the English test split with varying table representations. Overview of metadata in Section 3.3.

Model	R@1	R@10	R@20
BM25	0.3	2.3	3.8
DPR	14.3	45.1	54.2
TAPAS	6.1	22.1	31.5
TAPAS-NQ	7.4	30.0	39.3

Table 9: Retrieval results for the English test split when encoding title and member items. DPR and TAPAS were run 3 times and averaged.

allows T5 to decide between using one of the suggested titles and generating something else (e.g., clarification question). This is similar to the agents' behavior, as they tend to return a title with the URL when sharing a relevant table. Furthermore, supervising T5 to ignore or return a title is equivalent to an implicit binary classification.

6 Results and Discussions

Based on our baselines and data, we report the results and analyze the challenges that our dataset and tasks pose for existing models. For the English splits, the main retrieval results are reported in Table 9 and Table 8, and main generation results are in Table 10. Full results can be found in Appendix A, respectively in Table 16 and Table 33, and relevant statistical tests in Appendix A.7.

Impact of table representation In Table 9, we observe that the metadata representation affects the retrieval recall. Although DPR can achieves respectable results when it only retrieves the title, including basic information (defined in Section 3.3) yields slight improvements, and further adding member items results in a significant difference from only using title ($p = 0.014$). However, only using member item result in drastic decrease in recall ($p = 0.00086$), indicating the importance of the title. Moreover, footnotes do not yield any improvement, which may be because they often exceed the maximum context span (see Table 20). Thus, concisely but meaningfully representing metadata will be crucial to achieve a good recall on the retrieval task.

Metrics	No aug.	Top-1 Title	Top-5 Titles
METEOR	23.35	24.07	24.41
ROUGE-L	30.65	30.76	30.88
MoverScore	59.82	60.23	60.31
BERTScore	86.04	86.11	86.17
Title Acc.	6.96	7.99	10.82

Table 10: Response generation results for the English test split.

Transfer to table retrieval task Our experiments allow us to analyze the effectiveness of open-domain QA fine-tuning (NQ) and tabular pretraining when transferring to our table retrieval task. We observe in Table 9 that DPR outperforms TAPAS and TAPAS-NQ by respectively 23.0% and 15.1% in test recall@10. Moreover, TAPAS-NQ achieves a better performance when it only retrieves the title and member items instead of the full table ($p = 0.016$), likely due to repetitions and truncation due to context size limits. Although both DPR and TAPAS-NQ were trained on NQ, the latter was trained on a small subset (12K vs 320K) that contains tables. Our experiments indicate that TAPAS transfers poorly from one task (NQ-Tables) to another (StatCan).

Response generation We compare the performance of fine-tuned T5-large models with and without DPR-augmented table titles. In Table 10, we notice that retrieval-augmented models show modest improvements on the 4 non-task specific metrics. However, the top-5 augmented model achieves an absolute improvement of 3.86% in title accuracy, indicating that the information provided by DPR does help T5 in generating the desired title. It is nevertheless surprising that T5 without augmentation achieves a score of 6.96%, suggesting that the T5 model is capable of storing the titles seen at training time, and, to a limited extent, is able to recall and return them at test time. Finally, we point out that the title accuracy is still 5.91% lower than top-1 recall of the DPR retriever (Table 8), indicating that T5 fails to learn when to return a table (despite the agent retrieving a table in 23.4% of all turns). In the case of top-5 titles, T5 struggles to decide which table title to return.

Qualitative analysis of generated responses We examine various conversations to understand what type of responses are generated by T5. We find that it can generate simple speech acts like greetings, but can struggle with context-specific speech acts such as clarification questions. Moreover, it can

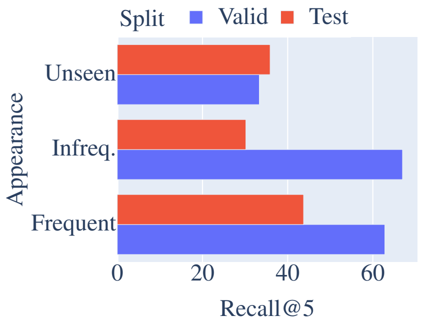


Figure 3: Results for DPR B+M for tables appearing in the training set frequently (10+ times), infrequently (1-10 times), and unseen at train time.

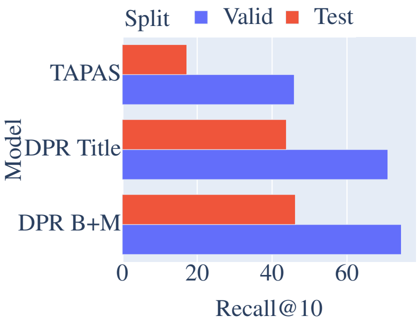


Figure 4: Validation and test recall for a selected set of retrieval models. We observe a significant drop in performance. *B+M* denotes Basic + member.

reliably reply with the title of a common table, struggles for uncommon ones, and is sometimes capable of generating unseen titles with the help of DPR. The full analysis is in Appendix A.6.

Temporal drifts As explained in Section 3.4, we use a temporal split to test the model's ability to generalize to future conversations. We observe a significant drop in recall (13%-28.3%) in Figure 4 when we compare the validation and test set performance, even when the models are trained with varying metadata representations. Similarly, T5 achieves low scores on the test split for the response generation task (Table 10). This large gap suggests that trained models struggle to generalize to future conversations. First, we found that this is likely caused by the number of new tables that appear in the test split (145) compared to validation (41), as shown in Table 3. Moreover, the subjects of the conversations have significantly changed: users started to care more about businesses, health and IT, and less about demography, income and pensions. This is likely motivated by real-world events affecting the users, which are more difficult to implicitly capture from simulated environments, but desirable in order to understand a model's robustness in temporal shift and for real world applications. In the Appendix, Figure 8 displays the differences between the training and test splits for all subjects.

Generalizing to unseen tables As shown in Figure 3, DPR performs well for tables appearing frequently in the validation split, but poorly in the test split, which could be caused by temporal drift. As expected, tables that were not seen during training resulted in poor recall@5 in either splits. Moreover, the difference in recall between valid and test for infrequent tables could be caused by many potential reasons (learning bias, temporal overfitting, spurious correlation with hidden factors). Thus, future models should aim to close the gap between unseen and frequent tables and within the temporal spectrum of infrequent tables.

French results In both tasks, we see a drop across all metrics for all models, likely due to the smaller dataset size. Some observations remain valid: temporal drift, poor BM25 performance, and augmentations benefit mT5 for certain metrics. However, others differ: adding member items hurts test results and mT5 performs poorly on *title accuracy*. Modeling details and results can be found in the Appendix A.5.

7 Conclusion

In this paper, we introduce the StatCan Dialogue Dataset, a novel corpus consisting of 20K+ English and French conversations between online visitors of statcan.gc.ca and operators of Statistics Canada. Based on this dataset, we propose two tasks centered on helping users find specific data tables: the table retrieval task and the response generation task. For the table retrieval task, we experiment with various DPR and TAPAS variants, finding that DPR strongly outperforms its TAPAS counterpart, as well as the BM25 baseline. For the response generation task, we investigate fine-tuned T5-large models and explore variants where the input is augmented with table titles from DPR. We find that retrieval-augmented T5 models more frequently return the correct tables, although its title accuracy is still lower than the corresponding recall of the DPR retriever. This result suggests that the generation models struggle to decide when to return a table. We also find that retrieval and generation have difficulty generalizing to future conversations, as our temporal test split revealed a big performance gap between the validation and test set. All in all, we believe that our tasks pose significant challenges to currently available models and encourage the research community to further explore this dataset and build conversational models that help users of Statistics Canada.

Limitations

Tasks and models limitations The tables in the retrieval task are sourced from statcan.gc.ca, which means that the content is primarily about Canadian demographics[2] and are professionally edited by StatCan employees. Moreover, the generation task is specifically designed to model responses with high fidelity based on retrieved tables, so this task should not be directly used in an unintended or non-research setting (e.g., deploying a virtual assistant) as they pose risks of hallucination that could negatively impact stakeholders. Furthermore, those limitations can be reflected in the models we trained, so we will share those limitations in the model cards (Mitchell et al., 2019) on release.

Environment impact We acknowledge the models in Section 5 used hardware with significant energy consumption. We purposefully chose models of reasonable sizes that can be reproduced on one GPU. Additionally, our hardware is powered by renewable energy.

Artifacts and computational experiments We trained models using libraries based on their intended use and we will release the relevant artifacts following the original licenses. The computational details of the experiments are described in Appendix A.4.

Ethics Statement

Privacy and data access As discussed in Section 1, significant efforts were made to remove Personally Identifiable Information (PII). However, we do not rule out the possibility that certain details could have been missed in that process. Thus, any user that wishes to use the data will need to authenticate and accept the terms of use through an institutional data repository; the terms will require the user to report any instance of PII leak, which will be removed with a dataset update. Additionally, we request any derivative or modifications to be published in the same data repository with the original terms of use and licenses preserved or extended.

Risk of toxicity in online discourse StatCan agents are trained to work with online users in a professional manner. Moreover, since the users access statcan.gc.ca anonymously and virtually, it is more likely to observe toxic online disinhibition (Lapidot-Lefler and Barak, 2012), which could translate to toxicity in users' utterances. Thus, we request dataset users to report any instance of toxicity in conversations, which will be reviewed in the same manner as PII leaks.

Acknowledgement

We thank our collaborators at Statistics Canada for providing us the dataset, guiding us through the technical aspects of the tables, and sharing valuable feedback on the project. We thank Sivan Milton for the helpful discussions on dialogue acts and analysis.

References

Vaibhav Adlakha, Shehzaad Dhuliawala, Kaheer Suleman, Harm de Vries, and Siva Reddy. 2022. TopiOCQA: Open-domain Conversational Question Answering with Topic Switching. *Transactions of the Association for Computational Linguistics*.

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-domain question answering goes conversational via question rewriting. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 520–534, Online. Association for Computational Linguistics.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv:1611.09268 [cs]*. ArXiv: 1611.09268.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

---

[2] More information can be found here: `https://www.statcan.gc.ca/en/subjects-start/population_and_demography`

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, Claudia Soria, and David Traum. 2010. Towards an ISO Standard for Dialogue Act Annotation. In *Seventh conference on International Language Resources and Evaluation (LREC'10)*.

Harry Bunt, Volha Petukhova, Emer Gilmartin, Catherine Pelachaud, Alex Fang, Simon Keizer, and Laurent Prevot. 2020. The iso standard for dialogue act annotation. In *12th Edition of its Language Resources and Evaluation Conference (LREC 2020)*, pages 549–558. European Language Resources Association (ELRA).

Derek Chen, Howard Chen, Yi Yang, Alexander Lin, and Zhou Yu. 2021. Action-based conversations dataset: A corpus for building more in-depth task-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3002–3017, Online. Association for Computational Linguistics.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training Deep Nets with Sublinear Memory Cost. *arXiv:1604.06174 [cs]*. ArXiv: 1604.06174.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. 2020. Open Question Answering over Tables and Text. In *International Conference on Learning Representations*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question Answering in Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

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

Martin d'Hoffschmidt, Wacim Belblidia, Quentin Heinrich, Tom Brendlé, and Maxime Vidal. 2020. FQuAD: French Question Answering Dataset. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1193–1208, Online. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of Wikipedia: Knowledge-Powered Conversational Agents. In *International Conference on Learning Representations*.

Etalab Lab-IA. 2021. DPR CamemBERT trained on French Q&A.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021. MultiDoc2Dial: Modeling dialogues grounded in multiple documents. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6162–6176, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8118–8128, Online. Association for Computational Linguistics.

Timnit Gebru, Jamie H. Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna M. Wallach, Hal Daumé, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64:86 – 92.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese Machine Reading Comprehension Dataset from Real-world Applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Proceedings of the workshop on Speech and Natural Language - HLT '90*, pages 96–101, Hidden Valley, Pennsylvania. Association for Computational Linguistics.

Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Eisenschlos. 2021. Open Domain Question Answering over Tables via Dense Retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 512–519, Online. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly Supervised Table Parsing via Pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*,

pages 4320–4333, Online. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Ali Kabbadj. 2021. French-SQuAD : French Machine Reading for Question Answering. Original-date: 2019-09-22T17:54:45Z.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Rachel Keraron, Guillaume Lancrenon, Mathilde Bras, Frédéric Allary, Gilles Moyse, Thomas Scialom, Edmundo-Pavel Soriano-Morales, and Jacopo Staiano. 2020. Project PIAF: Building a native French question-answering dataset. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5481–5490, Marseille, France. European Language Resources Association.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Noam Lapidot-Lefler and Azy Barak. 2012. Effects of anonymity, invisibility, and lack of eye-contact on toxic online disinhibition. *Computers in Human Behavior*, 28(2):434–443.

Harrison Lee, Raghav Gupta, Abhinav Rastogi, Yuan Cao, Bin Zhang, and Yonghui Wu. 2022. Sgd-x: A benchmark for robust generalization in schema-guided dialogue systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics - ACL '04*, pages 605–es, Barcelona, Spain. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Marco Lui and Timothy Baldwin. 2011. Cross-domain Feature Selection for Language Identification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 553–561, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model Cards for Model Reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pages 220–229, New York, NY, USA. Association for Computing Machinery.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W. Bruce Croft, and Mohit Iyyer. 2020. Open-Retrieval Conversational Question Answering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pages 539–548, New York, NY, USA. Association for Computing Machinery.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text

Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Radim Rehurek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *In Proceedings of the Lrec 2010 Workshop on New Challenges for Nlp Frameworks*, pages 45–50.

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision*, 40(2):99–121.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2087–2097, Brussels, Belgium. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ellen M. Voorhees. 2001. The TREC question answering track. *Natural Language Engineering*, 7(4):361–378.

Harm de Vries, Dzmitry Bahdanau, and Christopher Manning. 2020. Towards Ecologically Valid Research on Language User Interfaces. *arXiv:2007.14435 [cs]*. ArXiv: 2007.14435.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text Generation Evaluating with Contextualized Embeddings and Earth Mover Distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

A Appendices

A.1 Complete User Intents

To provide insight into what kind of help is offered by StatCan's live chat, we qualitatively analyze the conversations and highlight examples of the main user intents below.

Finding a table The main intent of the chat functionality is to help users find specific data tables. For example, one user was looking for the population numbers in certain regions of Montreal for 2012-2016. This intent is the focus of our work.

Obtaining meta-information of table Instead of finding data tables some users are interested in meta information of a specific table. For example, one visitor wanted to know when the next Census is released. Another user was interested in understanding the definition of Workforce Availability (WFA) and Labour Market Availability (LMA).

Manipulating a data table Some users would like to obtain the data tables in a different format or representation. For example, one user was looking at a specific data table and asked if they can see annual instead of monthly values.

Help with user interface Some users are looking for help with the user interface. For example, one user wanted to download a specific data table but they were unable to find the download link.

Out of domain requests We find many conversations that are outside of the scope of StatCan's live chat. For example, some user asked what documentation needs to be provided to ship a specific product to a foreign country.

The first intent is covered in Table 1, and subsequent intents are in Table 18. While we believe all intents are interesting directions for dialogue research, we focus on the table retrieval intent because (i) there are many conversations available for them and (ii) there is a clear measure of task success i.e., whether the correct table is retrieved. Throughout the rest of this paper, we work with conversations where the agent returns a table URL.

A.2 Supplementary Table Description

A.2.1 Formatting and size

Full tables are stored in a *long* format, where each column corresponds to a dimension, except the last one which corresponds to the value. The number of rows corresponds to the Cartesian product of the dimensions' members. This means that for n_i members across d dimensions, we have a total of $\prod_{i=0}^{d} n_i$ rows in the full table. When a table has many member items, the full table can become extremely large. Moreover, tables shown on the statcan.gc.ca are usually a pivoted and filtered view of the full table, which means certain members will become columns, others will become rows, and many are simply omitted.

A.2.2 Detailed Specifications

This section provides supplementary details for Section 3.3.

Product ID (PID) Unique 8 to 10-digit identifier given to each published data table. Although other types of tables might be shared by the agent, the PID will always be given for the official data tables; as a result, any table that does not have a PID in the URL is not considered for this task. The first two digits (1-2) represent the code of the subject associated with the table (this can be found in the basic information), then 3-4 represent the product type, which in our case are tables and are common coded as "10". Digits 4-8 is a unique identifier representing that table for the given subject. Digits 9-10 indicate the view of the table and will commonly be "01", which represents the default view; they are needed when constructing the URL but they are otherwise optional, and are omitted in the released dataset.

Member items Labels for individual tables, and can either be represented as a column or a row index through a pivot operation. Each table will have member items different from other tables.

Dimensions Non-overlapping sets of member items; each member item must belong to a dimension. For a table with d dimensions, each data value is associated with a single d-tuple in the d-fold Cartesian product of the dimensions, but not every tuple in the product will have an associated data value (if it was not recorded or if the tuple is invalid).

Basic information Metadatum consisting of the title, the date range, the frequency, the dimensions, the subject category, and the survey[3] from which the data was sourced.

Footnotes Unstructured comments often included with a table if supplementary details need to be given; usually, those notes are associated with a specific member item that requires more explanations.

Full table For all tables with a PID, the full tables (as a CSV file), their complete metadata, and their basic information are made available as part of the data release and can be used in our proposed tasks. A sample table can be found in Section 3.3.

---

[3] The list of surveys and link to detailed information can be found at https://www.statcan.gc.ca/en/survey/list

A.2.3 Table updates and archives

Tables on Statcan will sometimes be updated regularly, whereas in other times they may only be released once. When they stop being updated and the information becomes outdated, they will be marked as archived. In some cases, a new version of an archive table may be created with substantial changes (such as new columns). As shown in Table 14, tables are released without a predetermined schedule, but has been on average increasing since 2000, with major peaks in 2017, which was caused by many health-related tables, and 2021, which was caused by the release of many labour, science, and income tables, as shown in Figure 6. Moreover, Figure 5 shows that a majority of tables released between 2019 and 2021 are still up-to-date (current), whereas most of the tables before then have been archived.

A.3 Supplementary Statistics

A.3.1 Frequently requested tables

The most frequent tables are summarized in Table 21. Whereas 2 of them are sourced from the consumer price index survey (commonly used to track inflation), the 4 other tables cover more general and broad subjects like income, demography, business performance and crime. Each of those 4 tables are sourced from different surveys. One table is updated monthly, another semi-annually, and the rest are updated annually. The oldest table was updated in 2019, which is when the conversations started being recorded.

A.3.2 Table Frequency Statistics

In tables 3 and 12, we can calculate that a table is returned on average 5.68 times (with standard deviation of 12.86) in English conversations and 4.25 times (standard deviation of 9.73) in French conversations. Thus, the most requested tables are disproportionately represented compared to less popular tables, and there's a very high variance in the number of time a table is used. In fact, there are 294 tables that appear only once in either splits (i.e., 28.9%).

A.3.3 Fine-grained Conversation Statistics

In Table 13, we observe that the number of messages and turns will vary significantly around the mean, with over 68% conversations lasting between 2 and 7 turns. In extreme cases, a conversation can last up to 28 turns. Moreover, we also notice that, although most messages will have around 32 tokens, the longest message can have up to 1374 tokens; in those scenarios, we will see agents write a large body of text, and sometimes also copy and paste large amount of text (for example, from a database of templates) when responding to the user. Although those are usually sent in multiple consecutive messages within a turn, they may decide to send everything all at once.

A.4 Implementation Details

This section provides the details for implementing the models in Section 5.

Implementing transformer models All models based on the transformer architecture (Vaswani et al., 2017) were implemented using HuggingFace's library (Wolf et al., 2020).

BM25 To facilitate reproducibility, we implemented the model in Gensim (Rehurek and Sojka, 2010).

DPR and TAPAS-NQ We used the base variant of DPR and the large variant of TAPAS-NQ. We use the DPR checkpoints that were trained on 320K questions from Natural Questions (Kwiatkowski et al., 2019) (NQ). During training, the networks were optimized with AdamW (Loshchilov and Hutter, 2017) at a learning rate of 10^{-5} and zero weight decay. Based on the original work, the networks were trained for 30 epochs, with a batch size of 64 queries, positive passages, and hard negative passages (the latter are retrieved with BM25). Negative in-batch sampling was used to increase negative examples. To ensure reproducibility, the networks were trained on a single 32GB GPU and used gradient checkpointing (Chen et al., 2016) to reduce memory usage. The conversation lengths was 512 tokens, and the metadata token lengths were 128 for title, 256 for basic information (defined in Section 3.3), and 512 for the rest.

T5 We used an Adafactor optimizer (Shazeer and Stern, 2018) with a learning rate of 0.001. We used batch sizes of 16 with 8 steps of gradient accumulation and gradient checkpointing to reproduce the batch size of 128 samples in the original implementation. The models were trained on a single 32GB GPU for 10 epochs. The source and target lengths were respectively 512 and 256 tokens, where the source was truncated from the right to ensure that the latest messages remained after truncation. We used a beam size of 4 and length penalty of 0.6 following the original implementation.

Training time For the English split, each variant of DPR can be trained in 68 minutes on a A100 GPU. Each large variant of TAPAS-NQ takes 15h to train on a V100 GPU. Each variant of T5 can be trained in 16h on a V100 GPU. All DPR results can be reproduced in 8h, TAPAS in 90h, and T5 in 96h, and proportionally less time would be needed for the French split.

A.5 Modeling the French subsets

Basic statistics The number of messages by conversation varies between 2 and 59 with a median of 11 for the English split (see Figure 7 for the distribution). Based on Table 12, there's on average 3.9 turns but 12.3 messages. On average, there are over 30 tokens for each message (using the T5 tokenizer).

Language splitting In order to determine the language of each conversation, we used two popular language identification libraries: *langid.py* (Lui and Baldwin, 2011) and a fasttext network fine-tuned for language detection (Joulin et al., 2017). After apply the models on every conversation, we only retain the conversations with matching language labels (both English or both French).

Training and evaluation The training procedure and evaluation on the French subsets follow exactly the tasks specified in Section 4.

Modeling response generation Instead of T5, we used the multilingual T5 model by Xue et al. (2021) as it naturally handles text in French.

Modeling retrieval We used a variant of DPR derived from CamemBERT (Martin et al., 2020) and trained on three French Q&A datasets (Keraron et al., 2020; d'Hoffschmidt et al., 2020; Kabbadj, 2021) by Etalab Lab-IA (2021).

Retrieval results In Table 17, we observe that, unlike the English split, adding member items to the basic information or to the title improves validation results but not test results, which likely indicates overfitting. However, we notice a high variance between the runs, which makes it difficult to determine whether member items is helpful. Both overfitting and high variance are likely caused by the smaller size of the training set. Moreover, BM25 perform extremely poorly on any metadata view, which can also be linked to the dataset size.

Generation results In Table 33, we notice a significant decrease across all metrics, with the title accuracy being consistently 0%. This is likely because the French split is significantly smaller, yet remains as complex as the English split, which becomes challenging for mT5 to model. In the case of title accuracy, we found 55 instances in the French test split where the title is in the target text (i.e., returned by an agent). However, in 54 cases, the augmented mT5 returned a generic reply (e.g., “Veuillez patienter pendant que j'effectue une recherche.") instead of the expected title, which indicates that mT5 is incapable of determining when it is relevant to return a title and can't generate non-templated responses.

A.6 Responses generated by T5

In this section, we select a few conversations from the validation set and examine the messages generated by T5 and T5 augmented with DPR-retrieved titles (T5+D).

Common and uncommon responses In Table 23, we notice that both T5 and T5+D are capable of generating common speech acts like “Thank you” and “Please wait...”, but struggles when faced with an unfamiliar situation (having to ask for clarification for a user that has been accidentally disconnected).

Common table In Table 24, among the retrieved tables (R_i) titles, the first one was partially correct. Both T5 and T5+D extended the title and also output the desired ID, matching the expected agent's response, which is one of the most popular table in the training set (see Table 21).

Multiple tables, date selection In Table 28, we notice that T5 only returns one of the two tables that the agent returned. On the other hand, the correct tables were retrieved by DPR, but T5+D failed to select the ones with the correct dates (it selected June 2019 instead of December 2019) but the select were otherwise relevant.

Verbosity of explanations T5+D additionally provided a paragraph of explanation while linking to relevant resources, both on the StatCan website (non-tabular) and external resources. This is because T5+D memorized this information during training, and simply replaced the tables with the updated dates (Table 27).

Uncommon tables In the conversations shown in Tables 25, 26, 29, the retrieved table appears 10 times in the training set, which is significantly less common than the table retrieved in Table 24. For Table 25, the table returned by the agent was not retrieved by DPR, leading to T5+D returning the first title retrieved. However, in Table 26, the correct title was retrieved by DPR (title #4), yet T5+D failed to use that correct title in the generated message. As for Table 29, the agent gave a hint by stating “As a standard product, we have tables about employment by industry”, which was correctly acknowledged by DPR as the second retrieved title perfectly matches the PID of the table in the agent’s response. However, T5+D fails again at selecting the correct title, instead opting to return a generic response (“Please hold while I find the information”), and T5 hallucinates a PID that is different from the title it generated (both of which are wrong).

Tables unseen during training Among the tables that do not appear in the training set (see Table 11 for more information), there are seven that appear 3 or more times in the validation or test sets (Table 32). In Table 30, we see a conversation where DPR retrieves the correct title, which is correctly returned by T5+D, whereas T5 fails to return it. On the other hand, when DPR also correctly retrieves the title in Table 31, T5+D fails to return it, as it was likely mislead by the agent saying “Unfortunately,...”.

A.7 Statistics Tests

To back the claims in Section 6, we performed multiple single-tailed Welch t-tests, using the mean and corrected standard deviation from 16. The null hypotheses are that means of experiments A are different from the means of experiments B, across 3 runs. Unless otherwise specified, we use R@1 on the test split.

1. Claim: Adding basic information and member items to title results in a significant difference for DPR. With A being the model using only title, and B using basic + member, our p-value is 0.014.

2. Claim: For DPR, using member item result in drastic decrease. With A being the model using only member and B using title + member, our p-value is 0.00086.

3. Claim: TAPAS-NQ performs better with title and member items compared to the full table. With A being the model using the full table, and B using title + member, our p-value is 0.016.

4. Claim: In Figure 4, the validation recall@10 are higher than the test split for TAPAS, DPR Title and DPR Basic + member. With A being the validation score and B the test scores, the p-values are respectively $0.00197, 2.18 \times 10^{-5}, 0.00014$.

B Dataset Card

This section presents a dataset card that follows the format proposed by Lhoest et al. (2021), which was inspired by Mitchell et al. (2019) and Gebru et al. (2021).

Summary The StatCan Dialogue Dataset consists of over 20K+ conversations between agents working at Statistics Canada (StatCan) and users who are visiting StatCan's website and need support via the official live chat system.

Tasks A subset of 19K conversation turns is used to build two tasks:

1. Automatic retrieval of relevant tables based on a on-going conversation. For each partial conversation, the task is to return the ID of the most likely table returned by an agent. This is evaluated using the recall@k metric.
2. Automatic generation of appropriate agent responses at each turn. For each partial conversation, the task is to return the most likely response by an agent, including link to a relevant table. This is evaluated using four metrics described in Section 4.2.

Leaderboard The leaderboard and submission instructions can be found on the project webpage. Each submission will be accompanied with a tag indicating if:

- It was self-reported;
- The submissions were externally evaluated;
- The inference was reproduced following provided instructions;
- The complete training process was independently reproduced.

Languages The conversations were held in Canadian English (en-CA) and Canadian French (fr-CA).

B.1 Dataset Structure

B.1.1 Data Instances

Conversation A full example of a conversation can be found in Table 19. Instances for each user intent can be found in Table 18, and two conversations with annotated dialogue acts can be found in Table 22. For our case study in A.6, we show partial conversations in Tables 24, 25, 26, 27, 28, 29, 30, 31.

Tables The complete metadata of a table can be found in Table 20, which can be access at doi.org/10.25318/3210035401-eng. Table 21 shows the basic information for the most popular tables.

B.1.2 Data Fields

Full dataset A CSV file with the following fields is provided:

- `conversation`: The partial conversation (before a table is returned) in JSON format.
- `conversation_index`: A unique index that serves at identifying the conversation outside of this task.
- `conversation_processed`: The conversation converted into a readable text format, with extra information (such as timestamp) removed, the URLs replaced with a special tag, and separation tags (`</s>`) added.

Retrieval task CSV files with the following fields is provided for each split:

- `conversation`
- `conversation_index`
- `conversation_processed`
- `target_pid`: The product ID of the table that is returned by the agent
- `language`: The language reported by the live chat system, which may not always be accurate due to mislabeling.
- `ft_detected_lang`: The language predicted by fastText.
- `ft_detected_lang`: The score output by fastText.
- `lid_detected_lang`: The language predicted by langid.py.
- `lid_detected_prob`: The score output by langid.py.

Metadata The metadata that was used during retrieval is provided as a CSV file with the following fields (one for each of the 5907 tables):

- `pid`: The product ID of the table
- `title`: The title of the table

- `basic_info`: The basic information in the textual format
- `member_info`: The member items enumerated as text
- `footnote_info`: The footnotes enumerated as text
- `full_info`: The basic information, member items and footnotes all in a single entry
- `x_and_y`: An combination of two items above, for example `x=title` and `y=footnotes`.
- `*_fr`: All of the above are also available in French, indicated by the suffix `_fr`.

Generation task CSV files with the following fields is provided for each split:

- `source`: Equivalent to `conversation`.
- `source_processed`: Equivalent to `conversation_processed`.
- `target`: The message written by the agent following the conversation.
- `target_processed`: The message written by the agent following the conversation, with URLs replaced with a special tag.
- `conversation_index`

An augmented variant of each CSV file for the conversation task is provided with all of the above as well as the following fields:

- `source_augmented`: The same content as `conversation`, appended with the title of the top-5 tables retrieved by best DPR variant trained on the basic information.
- `target_augmented_1`: The same content as `source_augmented`, but using only the first table instead of top-5.

Conversation JSON formatting A conversation follows the following JSON format:

```
[
  ...
 {’timestamp’: ’13.03.2019 17:03:22’,
  ’actor’: ’user’,
  ’actor_name’: ’<NAME>’,
  ’text’: "I’ll take a look at that",
  ’urls’: []},
 {’timestamp’: ’13.03.2019 17:04:12’,
  ’actor’: ’operator’,
  ’actor_name’: ’Kelly C’,
  ’text’: ’Building permits...’,
  ’urls’: [’https:...’]},
  ...
]
```

B.1.3 Data Split

The retrieval splits has the following number of samples:

- Train: 3782 (en); 869 (fr)
- Validation: 799 (en); 201 (fr)
- Test: 870 (en); 141 (fr)

They correspond to the number of queries in Table 3 and Table 12 because each query results in a table being retrieved, which can happen more than once in a conversation. In such cases, the partial conversations will be truncated at different turns in the conversation.

The generation splits have the following number of samples:

- Train: 21582 (en); 3977 (fr)
- Validation: 4464 (en); 861 (fr)
- Test: 4850 (en); 884 (fr)

There are fewer samples than the number of messages in Table 3 and Table 12 because the goal of the task is only to predict the messages that will be written by the agent. Just like the retrieval task, the partial conversations will be truncated at different turns in the conversation.

B.2 Dataset Creation

B.2.1 Curation Rationale

Section 1 extensively motivates the curation of the dataset. To summarize, we enumerate the major points:

- Data from real users: We wanted a dataset that captures the linguistic challenges that exist in the real world
- Task-oriented dialogue: We wanted tasks with the specific goal of helping live chat users in their search of statistics.
- Real-world applications: Our model can be directly applied other statistics offices that want to set up a chat system, and our dataset will be useful for any organization that has (1) a chat system, and (2) a database of tables.

- Multilingual dialogue: We wanted to build models that can handle languages beyond English, which is why we also offer French versions of our tasks.

B.2.2 Source Data

Conversations The data was retrieved from the live chat system on statcan.gc.ca, which was anonymized by the development team at StatCan. The conversations happened between March 2019 and March 2021.

Tables The tables are publicly available and were downloaded following the instructions in the Web Data Service user guide: statcan.gc.ca/en/developers/wds/user-guide. The tables were released by Statistics Canada between 2000 and 2021. The data are either collected directly by Statistics Canada (e.g., through a census or a survey) or were compiled from existing sources (such as private sector organizations and government agencies) into official statistics. Existing sources include:

- Administrative data: Collected by government or the private sector as part of ongoing operations, and include records of birth and death, taxes, border control, and satellite data.
- Microdata Linkage: Existing information is linked to create new data. The existing information may not always be available publicly (for privacy purposes), thus linkage could add new information that was previously unavailable, while protecting the confidentiality of the public.
- Open data: Machine-readable and freely available data sourced from various channels (e.g., OpenStreetMap).
- Web scraping: Data from the internet that were scraped by Statistics Canada (this excludes personal information and "any information that will not be used to produce statistical output").

B.2.3 Annotations

The dataset does not contain any additional annotations beyond the ones collected through the live chat system and included in the table metadata.

B.2.4 Personal and Sensitive Information

Personal and sensitive information were removed programmatically from the conversations, and officially published tables only contain aggregated information that preserve the confidentiality of the participants. Although the removal process is highly advanced, there is a non-zero chance that some information can be used to reconstruct the profile of a user. For this reason, the access to the data will require researchers to sign-up and agree to the terms of use, and any derivative must be shared on the same platform and include the same terms.

B.3 Considerations for Using the Data

B.3.1 Social Impact of Dataset

The purpose of this dataset is develop and evaluate models that can assist knowledge workers in finding relevant tables from a data source. By providing a specialized retrieval system capable of returning more relevant results compared to general purpose, the productivity of the knowledge workers can be increased. For public agencies and statistics offices, this would benefit many live chat users interested in statistics related to a certain community.

B.3.2 Discussion of Biases

As discussed in Section 7, there are always risks of toxicity in online discourses, which means that the live users may exhibit negative biases in their messages. However, the StatCan agents are tasked to communicate with online users in a professional manner. Thus, researchers should not use this dataset to build models that generate messages written by live users, and any model trained on the dataset should not be used in scenarios where biases can negatively impact stakeholders.

B.3.3 Other Known Limitations

Section 7 describes other known limitations.

B.4 Additional Information

B.4.1 Dataset Curators

The dataset was curated by the authors of this paper based on the original data collected and processed by StatCan developers and agents.

B.4.2 Licensing Information

The conversations use a custom license, which needs to be accepted by researchers interested in accessing the conversation. The tables are released under the the Statistics Canada Open Licence: statcan.gc.ca/en/reference/licence.

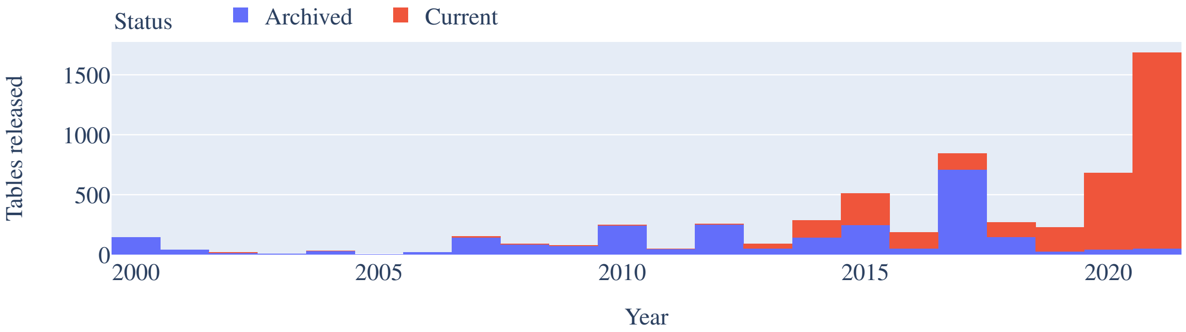


Figure 5: The release year of all tables available on statcan.gc.ca

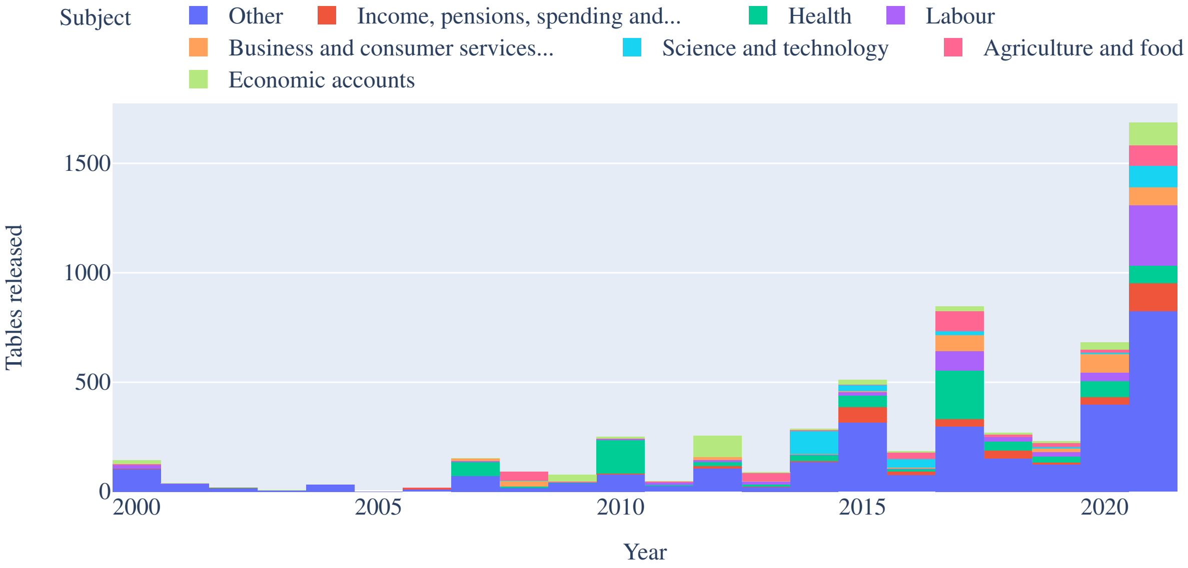


Figure 6: The release year of all tables by subject. Only the top 8 subjects are shown for readability.

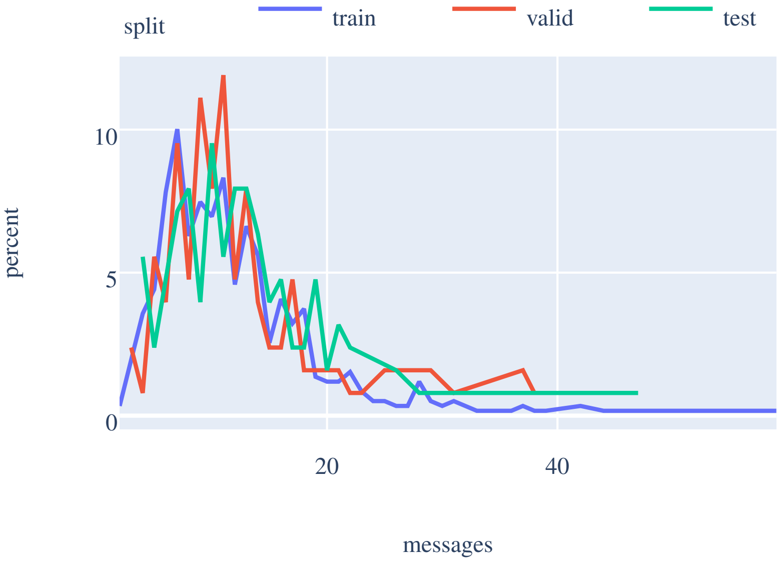


Figure 7: Histogram of messages by conversation in the French task splits. English split in Figure 1.

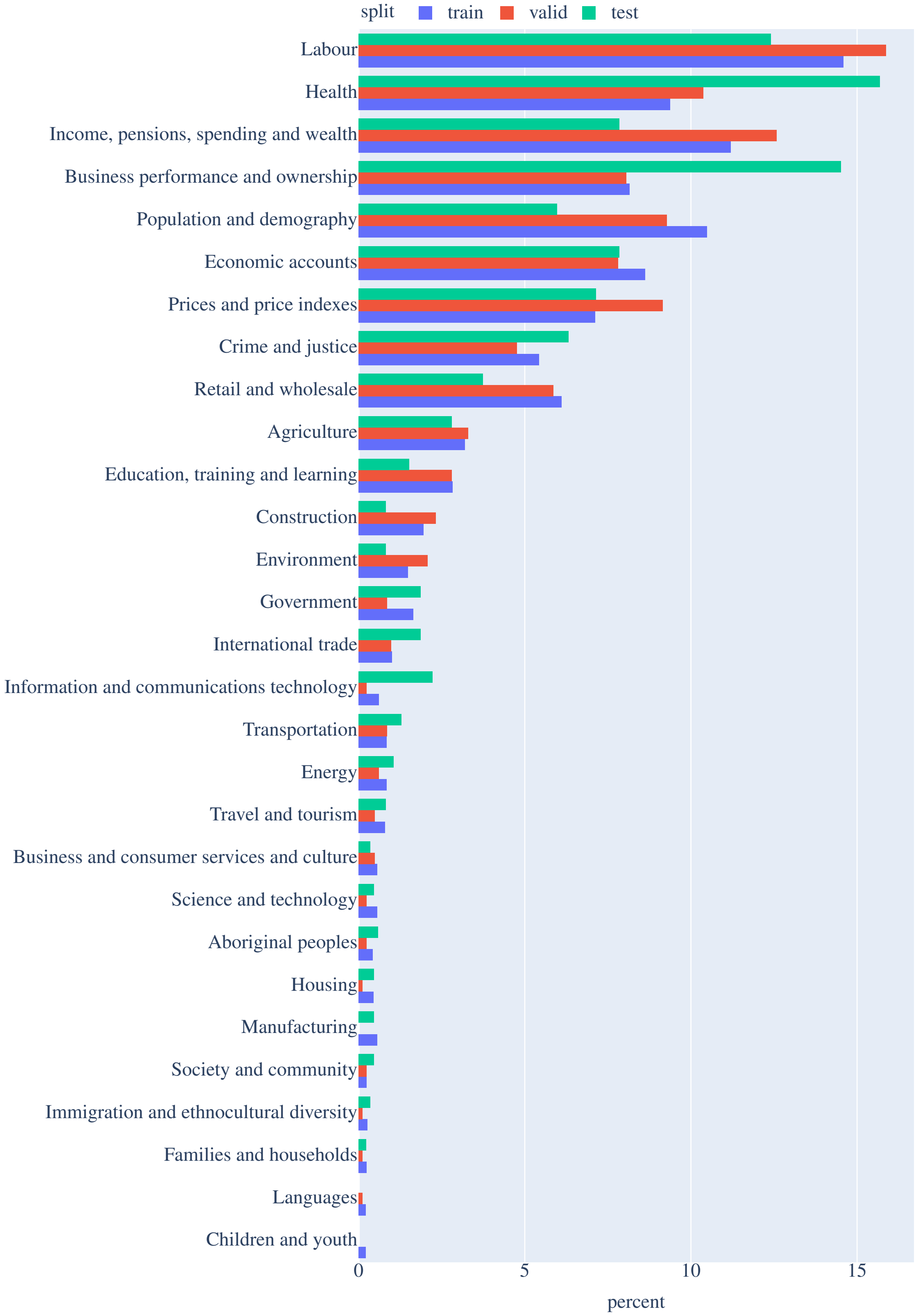


Figure 8: Distribution of subject categories for tables in the retrieval task, colored by the split.

	Train	Valid	Test	Overall
Train	0	470	535	0
Valid	41	0	199	0
Test	145	238	0	0
Overall	181	610	571	0

(a) Differences

	Train	Valid	Test	Overall
Train	778	308	243	778
Valid	308	349	150	349
Test	243	150	388	388
Overall	778	349	388	959

(b) Overlaps

	Train	Valid	Test	Overall
Train	0	154	184	0
Valid	18	0	64	0
Test	44	60	0	0
Overall	61	197	201	0

(c) Differences (French)

	Train	Valid	Test	Overall
Train	224	70	40	224
Valid	70	88	24	88
Test	40	24	84	84
Overall	224	88	84	285

(d) Overlaps (French)

Table 11: Number of tables (a,c) differing and (b,d) overlapping between each split (subset used for both tasks). The difference is computed as row - column. Summarized results in Table 3.

Dataset	Train	Valid	Test	All
# Conv.	562	122	109	793
# Turns	2147	472	439	3058
# Messages	6807	1492	1389	9688
# Queries	869	201	141	1211
# Tokens / Msg	47.25	49.86	40.14	46.63
# Turns / Conv.	3.82	3.87	4.03	3.86
# Msg / Conv.	12.11	12.23	12.74	12.22
# Queries / Conv.	1.55	1.65	1.31	1.53
# Tables	224	88	84	285
# New tables	0	18	44	61
# Dims / Table	3.4	3.5	3.5	3.4
# Mbrs / Table	205.8	249.9	164.1	190
# Notes / Table	20.3	26.3	23.4	21

Table 12: Statistics for the retrieval and generation tasks for the French split. See Table 3 for the English split.

Dataset	Train	Valid	Test	All
Max Tokens / Msg	1243	857	912	1243
Max Turns / Conv.	28	18	21	28
Max Msg / Conv.	78	58	73	78
Max Qs / Conv.	15	11	11	15
Std Tokens / Msg	1	1	1	1
Std Turns / Conv.	2.58	2.36	2.62	2.56
Std Msg / Conv.	8.53	7.55	8.98	8.47
Std Qs / Conv.	1.27	1.24	1.18	1.25
Max Tokens / Msg	1374	1373	352	1374
Max Turns / Conv.	15	10	15	15
Max Msg / Conv.	59	38	47	59
Max Qs / Conv.	15	15	5	15
Std Tokens / Msg	1	2	2	1
Std Turns / Conv.	2.21	2.08	2.1	2.17
Std Msg / Conv.	7.02	6.74	6.61	6.91
Std Qs / Conv.	1.84	2.18	0.77	1.79

Table 13: More statistics (max and standard deviation) at the conversation level (top: English, bottom: French), following Table 3.

Year Released	# Archived	# Up-to-date
2000	145	0
2001	39	0
2002	14	5
2003	8	0
2004	31	2
2005	1	0
2006	18	0
2007	142	11
2008	85	7
2009	72	5
2010	243	8
2011	45	3
2012	248	8
2013	49	40
2014	142	145
2015	247	264
2016	52	134
2017	710	136
2018	148	122
2019	27	203
2020	43	640
2021	52	1633

Table 14: Release of Statcan tables over the years

Merged Acts	**Count**	**Original Speech Acts**
answer	50	answer, confirm, disconfirm
request	31	request, acceptOffer, declineOffer
timeManagement	28	stalling, pausing
inform	63	inform, agreement, disagreement, correction
infoSeekingQuestions	25	propositionalQuestion, checkQuestion, setQuestion, choiceQuestion, directQuestion
promise	18	promise, acceptRequest, declineRequest
autoFeedback	25	positiveAutoFeedback, negativeAutoFeedback
offer	11	offer
instruct	18	instruct
clarificationQuestion	16	clarificationQuestion
greeting	28	initialGreeting, returnGreeting
selfIntroduction	17	initialSelfIntroduction, returnSelfIntroduction
thanking	47	thanking
acceptThanking	15	acceptThanking

Table 15: Merged and original speech acts occurring in 100 turns in conversations from the validation set. This table is summarized in Table 4.

		Valid			Test		
Model	Encoded	R@1	R@10	R@20	R@1	R@10	R@20
DPR	Basic	37.8 ± 1.82	73.1 ± 1.02	80.8 ± 0.51	14.7 ± 1.13	45.0 ± 0.24	55.0 ± 1.67
	Basic + footnote	40.0 ± 0.64	73.3 ± 0.85	81.3 ± 1.19	13.9 ± 0.37	44.4 ± 2.73	54.2 ± 2.65
	Basic + member	40.0 ± 0.44	74.5 ± 1.38	81.6 ± 0.55	15.7 ± 0.7	46.2 ± 0.44	56.3 ± 0.58
	Member	28.0 ± 0.47	62.1 ± 1.07	71.4 ± 1.06	10.7 ± 0.88	35.0 ± 0.57	46.3 ± 0.83
	Title	37.8 ± 0.21	70.9 ± 0.67	78.2 ± 1.18	13.9 ± 0.59	43.8 ± 1.22	53.4 ± 1.42
	Title + footnote	39.5 ± 0.69	73.2 ± 0.81	80.8 ± 0.47	13.3 ± 0.7	43.6 ± 0.46	53.6 ± 1.5
	Title + member	38.6 ± 1.19	72.1 ± 0.92	79.4 ± 0.72	14.3 ± 0.59	45.1 ± 0.44	54.2 ± 1.07
TAPAS	Table	16.9 ± 4.26	45.9 ± 5.46	55.5 ± 3.13	3.9 ± 0.65	17.2 ± 2.83	24.6 ± 3.45
	Title	21.5 ± 0.33	54.7 ± 3.39	63.7 ± 1.38	4.7 ± 0.53	20.2 ± 0.85	28.7 ± 1.98
	Title + member	24.6 ± 3.6	54.9 ± 3.34	63.7 ± 2.74	6.1 ± 0.98	22.1 ± 2.66	31.5 ± 3.02
TAPAS-NQ	Table	25.3 ± 3.46	60.0 ± 2.43	70.0 ± 2.89	5.3 ± 0.8	22.8 ± 2.32	32.0 ± 2.48
	Title	26.5 ± 1.85	64.4 ± 1.95	71.6 ± 2.32	6.3 ± 0.64	27.4 ± 1.84	36.1 ± 2.53
	Title + member	29.8 ± 2.74	62.9 ± 1.35	71.5 ± 0.63	7.4 ± 0.73	30.0 ± 1.46	39.3 ± 0.86
BM25	Basic	1.0	7.5	10.3	1.0	7.2	9.8
	Basic + footnote	0.9	2.4	3.1	0.9	2.8	3.9
	Basic + member	0.1	1.3	2.6	0.2	2.4	3.8
	Full	0.3	1.3	1.9	0.2	1.1	2.0
	Title	2.0	10.3	13.8	2.5	10.9	14.0
	Title + footnote	0.8	2.3	3.1	0.7	2.9	4.0
	Title + member	0.0	1.4	2.4	0.3	2.3	3.8

Table 16: Full retrieval results for the English splits. The values reported are in recall % at k. DPR and TAPAS were run 3 times and averaged (standard deviation given after ±). Selected results in Table 8.

		Valid			Test		
Model	Encoded	R@1	R@10	R@20	R@1	R@10	R@20
DPR	Basic	19.7 ± 1.52	57.4 ± 2.55	67.0 ± 5.01	12.5 ± 1.78	40.0 ± 1.08	48.5 ± 4.27
	Basic + footnote	23.2 ± 3.8	55.4 ± 1.88	66.8 ± 1.04	13.9 ± 1.48	35.9 ± 0.82	49.9 ± 3.91
	Basic + member	22.7 ± 4.35	57.9 ± 2.45	67.7 ± 3.48	12.8 ± 2.13	38.1 ± 2.17	47.0 ± 1.78
	Title	23.7 ± 2.92	58.5 ± 2.24	69.8 ± 2.35	14.7 ± 2.28	35.7 ± 3.2	48.2 ± 2.84
	Title + footnote	23.2 ± 2.74	54.7 ± 3.59	64.7 ± 2.28	14.4 ± 1.48	36.6 ± 2.69	45.4 ± 2.13
	Title + member	24.2 ± 1.52	55.4 ± 4.51	65.3 ± 2.01	14.4 ± 2.28	39.5 ± 0.82	48.0 ± 2.05
BM25	Basic	0.0	0.0	0.0	0.0	0.0	0.0
	Basic + footnote	0.0	0.0	0.0	0.0	0.0	0.0
	Basic + member	0.5	1.0	1.0	0.0	0.7	0.7
	Full	0.0	0.0	0.5	0.0	0.0	0.0
	Title	0.0	0.5	0.5	0.0	0.0	0.7
	Title + footnote	0.0	0.0	0.0	0.0	0.0	0.0
	Title + member	0.0	0.5	1.0	0.0	0.7	0.7

Table 17: Full retrieval results for the French splits. The values reported are in recall % at $k \in \{1, 10, 20\}$. DPR was run 3 times and averaged (standard deviation given after ±).

	Obtaining meta-information of table
U:	Hi I have a question on the date of new census
A:	Hello my name is Graham, how may I help you?
U:	Hi Graham
U:	I am currently looking at : Census of Canada and National Household Survey
U:	I see that the last census were made in 2011 and 2016
U:	should I expect a new census in the coming month in 2020 ?
A:	no - the next Census is going to be taking place in May 2021
A:	it's every 5 years
U:	Ok perfect thank you
A:	you're welcome! have a good day
	Help with user interface
U:	Hello Graham - is there way I can see total annual values instead of monthly?
U:	for - https://www150.statcan.gc.ca/t1/tbl1/en/cv!recreate.action?...
A:	please hold - I will see if a table like this with an annual frequency is available
U:	wonderful thank you
A:	unfortunately no, the only active table on construction investment is the monthly one I sent you - there is an inactive quarterly table Investment in Building Construction (https://www150.statcan.gc.ca/n1/en/surveys/5014)
A:	it is also possible to download the monthly data in .CSV form, add together the months in a spreadsheet to get annual totals
A:	or if you'd prefer, you can make a custom request to our analysts (a fee may apply).Please send us your request, clearly indicating the variables, geographic areas, time frame and data frequency (i.e., annually, monthly or quarterly) needed at infostats@canada.ca (mailto:infostats@canada.ca) and refer to Case number: 986005 so we can access the transcript of this live chat. Your request must include your name, address, postal code, and phone number (please do not share your personal information on the live chat.)An agent will get in touch with you to provide an estimate and proceed with the payment, if applicable.You may also submit your request by telephone at 1-800-263-1136.
U:	thats fine! thank you very much :)
U:	very helpful
U:	:)!
U:	have a good day!
A:	glad I could help! you too
	Help with user interface
U:	Hello, Do you know where I can download the dataset for General Social Survey Cycle 32?
A:	Please wait while I try to find this information for you.
U:	Thank you
U:	(I'm referring to this one: https://www23.statcan.gc.ca/imdb/p2SV.pl?...)
A:	General Social Survey: Caregiving and Care Receiving, Public Use Microdata File (https://www150.statcan.gc.ca/...)
U:	I'm sorry, but im having trouble finding the download link for cycle 32 (2018)
A:	you have to click on 'More information' and that will bring you to the order form
U:	ohhhh, I see
U:	thank you for letting me know
U:	thank you for your help!
A:	you're welcome - happy to help!
	Out of domain requests
U:	I'm wondering how to obtain a copy of an autopsy that's been done for a family member. Doctor office says they don't get the autopsy report.
A:	Statistics Canada does not issue certificates for birth, death or marriage. Civil registration is a provincial jurisdiction. Therefore, you must contact the government ministry, registry or vital statistics office of the province or territory in which the birth, death or marriage occurred. To obtain the contact information for your provincial or territorial government, please call Service Canada at 1-800-O-Canada (1-800-622-6232) or visit the following page: Vital Statistics contacts (https://www.statcan.gc.ca/eng/health...) .
A:	also: Coroner's report (https://spvm.qc.ca/en/...)

Table 18: Examples of user intents described in Section A.1.

U: Hi, I'm looking to obtain quarterly data in regards to GDP grow (Canada), BC Housing STarts, Canada Oil Price/BBL
A: Hello, my name is Kelly C. Give me one moment as I search

U: thanks!
A: For GDP growth rates, please consult the following link: Gross domestic product (GDP) at basic prices, by industry, monthly, growth rates
A: I'll continue searching for the rest. One moment
A: What do you mean by BC Housing Starts?

U: I'm required to research all of the housing starts for BC on a quarterly basis
A: Define "starts".

U: unit basis
U: Housing starts are the number of new residential construction projects that have begun during any particular month
A: Perfect. Give me one moment.
A: I'm still searching. Sorry about the wait.

U: no worries
A: I would have monthly data regarding new building permits being issued.

U: I'll take a look at that
A: Building permits, by type of structure and type of work (https://www150.statcan.gc.ca/t1/...)
A: Do play with the variables to get what you are looking for.

U: thank you
A: To view/ manipulate the variables available in a data table: • Click on "Add/Remove data" • Select at least one variable in each tab and click on "Apply" in order to view your customized table To download the data: • Click on "Download" • Select one of the three output formats then click on the hyperlink and save the table The following video may also be helpful to you: How to use the data tables (https://www.statcan.gc.ca/eng/sc/video/howto) .
A: I'll have a look for oil prices. One moment.

U: Do you also have data to Canada'as oil Price/BBL ("WTI")?
U: okay thanks!
A: Are you looking for the retail prices of oil?
A: If so, I found some data for smaller geographies.
A: Monthly average retail prices for gasoline and fuel oil, by geography (https://www150.statcan.gc.ca/t1/...)

U: yes
A: Would those geographies be enough?
A: Or are you looking for Canada only?

U: hmm
U: I would need something that pertains more to all of canada
A: What about this? Monthly average retail prices for food and other selected products (https://www150.statcan.gc.ca/t1/...)
A: Is there anything else I can do for you?

U: I thinks that's all
U: I
U: i'll try and use this thanks!
A: No problem. Have a good one!

Table 19: Full Example (Sample #42) taken from the conversations dataset. URLs were updated to link to default view (rather than filtered). See Table 1 for truncated conversation.

Title: Production and value of maple products

Date range: 1924-01-01 to 2020-01-01
Dimensions: Geography, Maple products
Subject: Agriculture
Survey: Maple Products
Frequency: Annual

Geography:
ID: 1, Parent: None, Name: Canada
ID: 2, Parent: 1, Name: Nova Scotia
ID: 3, Parent: 1, Name: New Brunswick
ID: 4, Parent: 1, Name: Quebec
ID: 5, Parent: 1, Name: Ontario
Maple products:
ID: 9, Parent: None, Name: Maple products expressed as syrup, total
ID: 10, Parent: None, Name: Gross value of maple products
ID: 1, Parent: None, Name: Production of maple sugar
ID: 2, Parent: None, Name: Value of maple sugar
ID: 3, Parent: None, Name: Production of maple syrup
ID: 4, Parent: None, Name: Value of maple syrup
ID: 5, Parent: None, Name: Production of maple taffy
ID: 6, Parent: None, Name: Value of maple taffy
ID: 7, Parent: None, Name: Production of maple butter
ID: 8, Parent: None, Name: Value of maple butter

ID: 0, Note: Conversion factors: 1 gallon of syrup equals 10.0 pounds of maple sugar. One gallon of syrup weighs 13.24760 pounds. One gallon of syrup equals 10.4 pounds of taffy. Maple taffy is reported by Quebec and Nova Scotia only and commenced reporting in 1965 and 1983 respectively. The conversion of maple taffy to syrup varies with the density of syrup that year.
ID: 0, Note: Commercial production and value figures exclude inventory.
ID: 2, Note: Estimates produced by Nova Scotia horticulture industry specialists.
ID: 3, Note: Beginning in 1986, survey data for New Brunswick and Ontario are weighted using the number of taps made on maple trees in the spring of the year of the most recently completed census of agriculture.
ID: 4, Note: Estimates produced by l'Institut de la statistique du Québec, Groupe AGÉCO and Fédération des producteurs acéricoles du Québec.
ID: 5, Note: Beginning in 1986, survey data for New Brunswick and Ontario are weighted using the number of taps made on maple trees in the spring of the year of the most recently completed census of agriculture.
ID: 9, Note: "These products are represented by North American Product Classification System (NAPCS) Canada 2017 category 115136

Maple syrup and other maple products. Data collected for maple butter, maple taffy and maple sugar are converted into a maple syrup equivalent and are included in this category."
ID: 10, Note: "These products are represented by North American Product Classification System (NAPCS) Canada 2017 category 115136 – Maple syrup and other maple products. Data collected for maple butter, maple taffy and maple sugar are converted into a maple syrup equivalent and are included in this category."
ID: 0, Note: "These products are represented by North American Product Classification System (NAPCS) Canada 2017 category 115136 – Maple syrup and other maple products. Data collected for maple butter, maple taffy and maple sugar are converted into a maple syrup equivalent and are included in this category."

Table 20: Full metadata of table in Section 3.3. Single lines were added to delimit the scope: Title, basic information, item names, and footnotes. The double line was added to delimit the truncation limit of the DPR and TAPAS model (512 tokens). The basic information is presented in Table 5.

PID	**18100004 (count = 170)**
Title	Consumer Price Index, monthly, not seasonally adjusted
Dates	1914-01-01 to 2021-08-01
Dim.	Geography, Products and product groups
Subject	Prices and price indexes
Survey	Consumer Price Index
Freq.	Monthly
PID	**11100222 (count = 125)**
Title	Household spending, Canada, regions and provinces
Dates	2010-01-01 to 2019-01-01
Dim.	Geography, Statistic, Household expenditures, summary-level categories
Subject	Income, pensions, spending and wealth
Survey	Survey of Household Spending
Freq.	Annual
PID	**17100005 (count = 123)**
Title	Population estimates on July 1st, by age and sex
Dates	1971-01-01 to 2021-01-01
Dim.	Geography, Sex, Age group
Subject	Population and demography
Survey	Annual Demographic Estimates Canada, Provinces and Territories
Freq.	Annual
PID	**33100214 (count = 117)**
Title	Canadian Business Counts, with employees, June 2019
Dates	2019-01-01 to 2019-01-01
Dim.	Geography, Employment size, North American Industry Classification System (NAICS)
Subject	Business performance and ownership
Survey	Business Register
Freq.	Semi-annual
PID	**35100177 (count = 104)**
Title	Incident-based crime statistics, by detailed violations, Canada, provinces, territories and Census Metropolitan Areas
Dates	1998-01-01 to 2020-01-01
Dim.	Geography, Violations, Statistics
Subject	Crime and justice
Survey	Uniform Crime Reporting Survey
Freq.	Annual
PID	**18100005 (count = 91)**
Title	Consumer Price Index, annual average, not seasonally adjusted
Dates	1914-01-01 to 2020-01-01
Dim.	Geography, Products and product groups
Subject	Prices and price indexes
Survey	Consumer Price Index
Freq.	Annual

Table 21: Most frequently retrieved tables across all splits in the retrieval task. The *basic information* is provided for each table. Analysis presented in Section A.3.1.

U:	hi, i was wondering if you vae any statistics on video game sales, ot high school drop out rates? infoSeekingQuestions; request;
A:	Hello my name is Sylvain, how may I help you? offer;
U:	hello, I am looking for high school drop out r ates and teen depression rates, but cant seem to find anything that goes back more than a year request;
A:	Data for High School dropouts is compiled by the Provincial Education Ministry. To obtain this infor please contact the Education Ministry of your province inform; promise;
A:	Data for mood disorder (depression) is available on our website from the Data Table 13- 10-0096-18 (https://www150.statcan.gc.ca/...) answer;
A:	To view/ manipulate the variables available in a data table: • Click on "Add/Remove data" • Select at least one variable in each tab and click on "Apply" in order to view your customized table To download the data: • Click on "Download" • Select one of the three output formats then click on the hyperlink and save the table The following video may also be helpful to you: How to use the data tables (https://www.statcan.gc.ca/eng/sc/video/...). inform; request;

U:	Hi team, can you please help me retrieve data of public deposits of chartered banks? infoSeekingQuestions; request;
A:	Please hold timeManagement;
U:	Surely. autoFeedback;
A:	You can obtain this info on our website from our Data Table 10-10-0116-01 (https://www150.statcan.gc.ca/t1/...) answer;
A:	To view/ manipulate the variables available in a data table: • Click on "Add/Remove data" • Select at least one variable in each tab and click on "Apply" in order to view your customized table To download the data: • Click on "Download" • Select one of the three output formats then click on the hyperlink and save the table The following video may also be helpful to you: How to use the data tables (https://www.statcan.gc.ca/eng/sc/video/...) . inform; instruct;
A:	Please take note that this is a non-Statistics Canada Table. This info is provide from The Bank of Canada inform;
U:	Thanks a lot! So, how should I get down to Public deposits? or rather which of these lines would direct me to Public deposits? clarificationQuestion;
A:	Under the category Chartered bank deposits. For more info please contact The Bank of Canada on their website by using their Online Request Form (https://www.bankofcanada.ca/...) answer; inform;

Table 22: Samples of the dialogue analysis in Section 3.2, which are summarized in Table 15. The speech acts are in blue at the end of each message. The remaining annotations are in the supplementary materials. Sample examples are given in Table 4.

U:	**Thank you for your information, but I have done that already. Your previous website had a "manipulate" function to calculate the percentage change. I don't see the button anymore, and wondered if you have other schedule prepared as that function is gone.**
A:	**Hello, my name is Jimmy N., how may I help you?**
T5:	Hello, my name is Jimmy N., how may I help you?
T5+D:	Hello, my name is Jimmy N., how may I help you?
U:	**I was talking with Sylvain, will you be able to see the log?**
A:	**Yes, please hold while I verify.**
T5:	Yes, please hold while I verify.
T5+D:	Yes, please hold while I verify.
A:	**Just to clarify, you're looking for all-items ?**
T5:	You can obtain this information at the following hyperlink : Table: 18-10-0004-13 Consumer Price Index by product group, monthly, percentage change, not seasonally adjusted, Canada, provinces, Whitehorse, Yellowknife and Iqaluit ( )
T5+D:	You can obtain this information at the following hyperlink : Table: 18-10-0004-01 Consumer Price Index by product group, monthly, [...] and Iqaluit ( )

Table 23: Conversation #4890. Each message is separated by a horizontal line. The original conversation, in **bold**, only contains messages by the user (U) and agent (A). The generated responses by T5 and T5 with top-5 DPR-retrieved titles (T5+D) predict the corresponding agent's message (in **bold**).

	[...]
U:	**Ideally, I want the percentage change for these items**
R_1:	*Consumer Price Index, monthly, not seasonally adjusted*
R_2:	*Monthly adjusted price index, provisional*
R_3:	*Consumer Price Index, monthly, seasonally adjusted*
R_4:	*Historical (real-time) releases of the industrial product price index, by major product group, monthly, inactive*
R_5:	*Consumer Price Index (CPI) statistics, alternative measures, unadjusted and seasonally adjusted, Bank of Canada*
A:	**You can obtain this information at the following hyperlink : Table: 18-10-0004-13 Consumer Price Index by product group, monthly, percentage change, not seasonally adjusted, Canada, provinces, Whitehorse, Yellowknife and Iqaluit (https://www150.statcan.gc.ca/t1/tbl1/...)**
T5:	You can obtain this information at the following hyperlink : Table: 18-10-0004-13 Consumer Price Index [...] and Iqaluit ( )
T5+D:	You can obtain this information at the following hyperlink : Table: 18-10-0004-13 Consumer Price Index [...] and Iqaluit ( )

Table 24: The second part of conversation #4890, immediately after Table 23. R_i indicates the i-th table retrieved by DPR, which is only seen by T5+D.

U:	This is pretty cool!
A:	Hello, my name is Martine, how may I help you?
U:	I have a question - how would I be able to access data for 2019 on the amount of money the federal government spent (money out) to Canadians through social programs like EI / OAS etc...
A:	Letme check..
U:	Thank you
R_i:	*Federal extramural expenditures on science and technology, type of science, performing sector, type of payment and geography*
R_i:	*Historical: Current transfers to other levels of government, 1968 System of National Accounts (SNA), annual, 1926 - 1986*
R_i:	*Private non-profit organization outsourced research and development expenditures*
R_i:	*Capital expenditures, sources of funding, by geography*
R_i:	*Historical: Government transfer payments to persons, 1968 System of National Accounts (SNA), annual, 1926 - 1986*
A:	**Revenue, expenditure and budgetary balance - General governments (x 1,000,000) (https://www150.statcan.gc.ca/t1/...3610047701)**
T5:	Data Tables (https://www150.statcan.gc.ca/n1/en/type/data?text=federal+debt)
T5+D:	Federal extramural expenditures on science and technology, type of science, performing sector, type of payment and geography ( )

Table 25: Conversation #8960.

U:	How can I find information relating to Kelowna population estimates, demography, population projection estimates, population by age and sex
A:	hello how may i help you today?
U:	How can I find information relating to Kelowna population estimates, demography, population projection estimates, population by age and sex
A:	let me see what i can find on the subject for the projections information our tables only provide it to the provincial level it does not go lowe then that <extra_id_0> (<extra_id_0>)
R_1:	*Estimates and projections of population aged 0 to 29, by age group, Canada, provinces and territories*
R_2:	*Projected population, by projection scenario, age and sex, as of July 1*
R_3:	*Population estimates on July 1st, by age and sex*
R_4:	*Components of projected population growth, by projection scenario*
R_5:	*Population estimates, quarterly*
A:	**https://www150.statcan.gc.ca/t1/tbl1/en/cv.action?pid=1710005801**
T5:	https://www150.statcan.gc.ca/n1/daily-quotidien/190619/dq190619f-cansim-eng.htm
T5+D:	https://www150.statcan.gc.ca/n1/pub/71-607-x/71-607-x2018005-eng.htm

Table 26: Conversation #21533.

U:	Hi there, do you know approximately how many craft distillers are operating in Canada?
A:	Hello my name is Sylvain, how may I help you?
U:	I was wondering if you had any information about craft distillers in Canada. Specifically, how many craft distillers are operating in Can
A:	The industries are classified according to the North American Industry Classification System (NAICS). To find the NAICS code for an indu refer to this page (http://www23.statcan.gc.ca/imdb/...1181553) . You may start your search by selecting one of the 20 sectors, until a six-d obtained. [...] You will find the number of business locations by NAICS code in the following data tables: Table 33-10-0105-01 Canadian Business Counts, with employees, December 2018 (https://www150.statcan.gc.ca/t1/...3310010501) [...] I also suggest consulting the financial data of small and medium enterprise in Canada; these organizations have annual earning revenues ranges of $30,000 to $20 million. T information is available by industry, province, territory and legal status of companies (incorporated or unincorporated). This data is not available on our website; however, i available on Innovation, Science and Economic Development Canada's website, using their Financial Performance Data (http://www.ic.gc.ca/...)[...]

Table 27: Conversation #1628

U:	Hi. I want to know how many small, medium and big companies are there in canada. Thank you
A:	Hello, my name is Olivier C. Please wait while i find the information you're looking for.
R_1:	*Canadian Business Counts, without employees, December 2019*
R_2:	*Canadian Business Counts, with employees, December 2019*
R_3:	*Canadian Business Counts, without employees, June 2019*
R_4:	*Canadian Business Counts, with employees, June 2019*
R_5:	*Canadian Business Counts, without employees, December 2017*
A:	**Thank you for waiting, you will find the information about the number of business per business size if you click the following hyperlinks : Canadian Business Counts, with employees, December 2019 (https://www150.statcan.gc.ca/t1/...pid=3310022201) Canadian Business Counts, without employees, December 2019 (https://www150.statcan.gc.ca/t1/...pid=3310022301)**
T5:	Thank you for waiting. You will find the information you're looking for if you click the following hyperlink : Canadian Business Counts, with employees, June 2019 ( )
T5+D:	Thank you for waiting. The industries are classified according to the North American Industry Classification System (NAICS). To find the NAICS code for an industry, please refer to this page (http://www23.statcan.gc.ca/imdb/...1181553). You may start your search by selecting one of the 20 sectors, until a six-digit NAICS is obtained. You will find the number of business locations by NAICS code in the following data tables: Table 33-10-0214-01 Canadian Business Counts, with employees, June 2019 ( ) Table 33-10-0215-01 Canadian Business Counts, without employees, June 2019 ( ) I also suggest consulting the financial data of small and medium enterprise in Canada; these organizations have annual earning revenues ranges of $30,000 to $20 million. This information is available by industry, province, territory and legal status of companies (incorporated or unincorporated). This data is not available on our website; however, it is freely available on Innovation, Science and Economic Development Canada's website, using their Financial Performance Data (http://www.ic

Table 28: Conversation #8500.

U:	Do you have any data showing labour in manufacturing companies and how it is allocated between direct and indirect labour?
A:	Hello, my name is Jimmy N. Could you please specify what you mean by direct and indirect labour.
U:	Direct = labour for building product Indirect = administrative, supervision, etc.
A:	The data you are requesting is not available as a standard product. However, it may be available through a custom order (a fee may apply). [...] Would you like to make this request?
A:	As a standard product, we have tables about employment by industry.
U:	I would just like to look at the standard product. Can you tell me where I can find it, or what it is called?
R_1:	*Labour force characteristics by industry, monthly, seasonally adjusted, last 5 months*
R_2:	*Employment by industry, monthly, seasonally adjusted and unadjusted, and trend-cycle, last 5 months*
R_3:	*Employment by industry, annual*
R_4:	*Employment by industry, three-month moving average, unadjusted for seasonality, provinces and economic regions, inactive*
R_5:	*Usual hours worked by industry, annual*
A:	**You can obtain this information at the following hyperlinks : Table: 14-10-0355-02 Employment by industry, monthly, seasonally adjusted (x 1,000)Table: 14-10-0023-01 Labour force characteristics by industry, annual (x 1,000)Employed labour force who worked full year, full time and reported employment income in 2015, 2016 Census (https://www150.statcan.gc.ca/t1/...1410035502)**
T5:	Table: 14-10-0296-01 Labour force characteristics by industry, annual (x 1,000) ( )
T5+D:	Please hold while I find the information.

Table 29: Conversation #19568.

U:	I am on webpage https://www150.statcan.gc.ca/n1/daily-quotidien/201224/dq201224beng.htm where can I find the same data set for previous years, for example 2019 2018 2017 etc
A:	Hello, can you tell me on which table you are : https://www150.statcan.gc.ca/n1/daily-quotidien/201224/dq201224b-cansim-eng.htm
U:	Provisional death counts and excess mortality, January to October 2020 Adjusted number of deaths, expected number of deaths and estimates of excess mortality, by week
A:	That is the report, but there is 5 different tables: https://www150.statcan.gc.ca/n1/daily-quotidien/201224/dq201224b-cansim-eng.htm Let me verify this for you. One moment please. To view/ manipulate the variables available in a data table: • Click on "Add/Remove data" • [...] Select at least one variable in each tab and click on “Apply” in order to view your customized table Data are available for 2016 up to 2020.
U:	The reference period stops at January 2020 I can’t seem to get data for past years
A:	Click on the schedule icon and select the year you want. (2016 - 2020)
U:	I did and all years are ’greyed out’ meaning they do not populate
A:	Let me verify this. One moment please.
U:	screenshot.lzsc
R_1:	*Weekly death counts, by age group and sex*
R_2:	*Previous releases and revisions to weekly death counts*
R_3:	*Adjusted number of deaths, expected number of deaths and estimates of excess mortality, by week, age group and sex*
R_4:	*Death counts, by week, inactive*
R_5:	*Labour force characteristics by sex and detailed age group, monthly, unadjusted for seasonality*
A:	**Have you already consulter the table that replaces table# 13-10-0785-01 [...] https://www150.statcan.gc.ca/t1/tbl1/en/tv.action?pid=1310081001 - On this table you can go back to 2016.**
T5:	You can manipulate the table by following these steps: • Click on "Add/Remove data" • Click on the “ + ” icon next to a checkbox to expand the list of variables (if applicable) • Select at least one variable in each tab and click on “Apply” in order to view your customized table To download the data: • Click on “Download” • Select one of the three output formats then click on the hyperlink and save the table The following video may also be helpful to you: How to use the data tables (https://www.statcan.gc.ca/eng/sc/video/howto).
T5+D:	Please consult the following data table(s): Weekly death counts, by age group and sex ( )

Table 30: Conversation #15376.

U:	Hello
A:	Hello, my name is Jimmy N., how may I help you?
U:	[...] We saw this article from the Vancouver Sun which mentioned StatsCan data on business closures. [...] is there any way for us to collect localized data for the Township, or at least the Fraser Valley
A:	Unfortunately, the information is available on a national and provincial/territorial level : And for some Census metropolitan areas
R_i:	*Experimental estimates for business openings and closures for Canada, provinces and territories, census metropolitan areas, seasonally adjusted*
R_i:	*Quarterly estimates of business entry and exit*
R_i:	*Of businesses where at least one employee was laid off, percentage of workforce laid off and rehired due to COVID-19, by business characteristics*
R_i:	*Canadian Business Counts, with employees, census metropolitan areas and census subdivisions, December 2020*
R_i:	*Manufacturing sales, by industry for 12 Selected Census Metropolitan Areas*
A:	**Table: 33-10-0270-01 - Experimental estimates for business openings and closures for Canada, provinces and territories, census metropolitan areas, seasonally adjusted (https://www150.statcan.gc.ca/t1/...3310027001)**
T5:	However, it may be available through a custom order (a fee may apply). To obtain a free cost estimate for this service, please send us your request, clearly indicating the variables, geographic areas, time frame and data frequency needed, making sure to include your complete contact information (name of organization, address, postal code, and phone number) at the following email address: infostats@canada.ca (mailto:infostats@canada.ca) or by telephone at 1-800-263-1136. Our agents are available Monday to Friday (except holidays) from 7:30 am to 7:30 pm (Eastern Time). An agent will get in touch with you to verify the information, provide an estimate and proceed with the payment, if applicable.
T5+D:	However, it may be available through a custom order (a fee may apply). To obtain a free [...] An agent will get in touch with you to verify the information, provide an estimate and proceed with the payment, if applicable. Please do not share any personal information through the live chat. Would you like to make this request?

Table 31: Conversation #11869.

PID	Title	Train	Valid	Test	Total
13100810	Selected grouped causes of death, by week	0	0	6	6
33100270	Experimental estimates for business openings and closures for Canada, provinces and territories, census metropolitan areas, seasonally adjusted	0	1	3	4
13100763	Health characteristics of children and youth aged 1 to 17 years, Canadian Health Survey on Children and Youth 2019	0	0	4	4
35100060	Number of homicide victims and persons accused of homicide, by Indigenous identity, age group and gender	0	1	2	3
32100042	Estimated areas, yield, production of corn for grain and soybeans, using genetically modified seed, Quebec and Ontario, in metric and imperial units	0	1	2	3
14100084	Labour force characteristics of immigrants by sex and age group, three-month moving average, unadjusted for seasonality	0	0	3	3
13100143	Deaths, by cause, Chapter V: Mental and behavioural disorders (F00 to F99)	0	0	3	3

Table 32: List of tables that do not appear in the training set, but appear at least 3 times in the validation or test sets.

Language	Split	Augmentation	METEOR	ROUGE-L	BERTScore	MoverScore	Title Acc.
English	Test	None	23.35	30.65	86.04	59.82	6.96
		Top-1 Title	24.07	30.76	86.11	60.23	7.99
		Top-5 Titles	24.41	30.88	86.17	60.31	10.82
	Valid	None	31.95	43.31	88.31	65.50	30.39
		Top-1 Title	33.11	44.22	88.60	66.14	33.66
		Top-5 Titles	33.38	44.20	88.55	65.94	38.89
French	Test	None	5.98	7.21	61.74	51.88	0.00
		Top-1 Title	9.71	11.14	65.86	54.81	0.00
		Top-5 Titles	8.96	10.75	64.76	54.38	0.00
	Valid	None	6.84	8.85	60.97	52.21	0.00
		Top-1 Title	8.09	9.03	64.22	53.62	0.00
		Top-5 Titles	7.37	8.75	62.55	52.97	0.00

Table 33: Full response generation results. Selected results in Table 10.

Question Generation Using Sequence-to-Sequence Model with Semantic Role Labels

Alireza Naeiji[1], **Aijun An**[1], **Heidar Davoudi**[2], **Marjan Delpisheh**[1], **Muath Alzghool**[1]
[1]Department of Electrical Engineering and Computer Science, York University, Canada
[2]Faculty of Science, Ontario Tech University, Canada
`{naeiji,aan,mdelpishe,alzghool}@yorku.ca`
`heidar.davoudi@ontariotechu.ca`

Abstract

Automatic generation of questions from text has gained increasing attention due to its useful applications. We propose a novel question generation method that combines the benefits of rule-based and neural sequence-to-sequence (Seq2Seq) models. The proposed method can automatically generate multiple questions from an input sentence covering different views of the sentence as in rule-based methods, while more complicated "rules" can be learned via the Seq2Seq model. The method utilizes semantic role labeling to convert training examples into their semantic representations, and then trains a Seq2Seq model over the semantic representations. Our extensive experiments on three real-world data sets show that the proposed method significantly improves the state-of-the-art neural question generation approaches.

1 Introduction

Question Generation (QG) from text has gained increasing interest due to its usefulness in various applications such as educational reading comprehension assessment (Chen et al., 2018; Kumar et al., 2018), data augmentation for training question-answering systems (Sultan et al., 2020), and response generation in conversational systems (Gu et al., 2021). We have been working on QG for the purpose of automatically creating the knowledge base (KB) for a conversational QA system of an industry partner[1]. The knowledge base consists of QA pairs extracted from a domain-specific document (such as car manuals). Previously, the creation of their KB was done manually, which is very labor-intensive. To automate this KB generation process, we have tried both rule-based and neural sequence-to-sequence (Seq2Seq) QG methods.

The rule-based methods create rules based on linguistic features that capture the relationships among components of a sentence and can generate multiple questions from an input sentence to cover different aspects of the sentence. However, designing such rules is a very *labour-intensive* task as well. Also, these rules may not capture the *complexity* of ways a human asks questions (Yuan et al., 2019). As we will show in Section 4, the rule-based method does not lead to good results compared to neural Seq2Seq models.

While the Seq2Seq methods achieve better results, such methods are highly data-driven. For domains with limited training data (such as car manuals), relations that map the input text to questions cannot be well captured. In addition, Seq2Seq models often generate a single question from an input text. However, multiple questions can be asked about a piece of text from different aspects. One way to generate multiple questions with Seq2Seq models from an input text is to mark the input text with different *answer spans* or keywords to show the focus for QG so that multiple questions may be generated from the same text, one for each answer span/keyword. However, in our application, such answer spans or keywords are not available as marking answer spans or keywords when creating training data requires intensive labor work. Our partner prefers an answer-unaware QG system that can automatically generate multiple factual questions without indicating answer spans or keywords in either training or inference time. Another technique for generating multiple questions is to use diverse beam search (Vijayakumar et al., 2018) However, the beam search methods require the user to specify the number of questions returned, which is hard to specify as the ideal number of generated questions varies among different input texts.

To address these issues, we propose a novel approach to question generation, which uses semantic role labeling (commonly used in rule-based systems) that can label an input sentence in different ways corresponding to multiple semantic views of

---

[1] iNAGO Corporation (iNago.com)

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2830–2842
May 2-6, 2023 ©2023 Association for Computational Linguistics

an input sentence, and then trains a Seq2Seq model with SRL-labeled sequences for question generation. Our method does not need keyword/answer span labels in the training data nor specifications of the number of multiple answers to be generated. The use of SRL also increases the number of training examples, which may help alleviate the problem of the limited labeled data problem.

We evaluate the proposed method on three real-world data sets and compared the proposed method with several state-of-the-arts QG methods including the ones that generate multiple questions. The extensive experiments on these data sets show that proposed framework is significantly better than the state-of-the-art Seq2Seq models and rule-based methods, especially in terms of coverage and overall scores considering both precision and recall.

2 Related Work

The question generation approaches broadly fall into two categories: rule-based approaches and neural Seq2Seq learning approaches. Rule-based approaches mainly relay on hand-crafted templates/rules built upon linguistic features (Chali and Hasan, 2015; Flor and Riordan, 2018; Khullar et al., 2018; Lindberg et al., 2013). These methods use rigid heuristic rules to transform a source sentence into one or more questions. However, rules have limited power in expressing the complicated mapping function that the human uses for question generation. Designing a comprehensive set of rules is a very labour-intensive task.

Recently, neural Seq2Seq models have been successfully applied to question generation due to its capability to extract effective features and model complicated functions. Early Seq2Seq-based QG models are based on RNN structures. Examples include an LSTM-based Seq2Seq model with the global attention mechanism (Du et al., 2017) and LSTM-based model with the maxout pointer and gated self-attention network (MP-GSN) (Zhao et al., 2018). More recent Seq2Seq models are based on Transformer which relies entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs. State-of-the-art Transformer-based models that have been pre-trained for QG include T5 (Raffel et al., 2020), BART (Lewis et al., 2020), ProphetNet (Qi et al., 2020), UNILM (Dong et al., 2019), UNILMv2 (Bao et al., 2020), and Ernie-Gen (Xiao et al., 2020), to list some examples. While the Seq2Seq methods achieve better results than rule-based methods (Du et al., 2017), they are highly data-driven. In addition, Seq2Seq models often generate a single question given an input text, which does not cover multiple views of a sentence.

To solve this single-question-generation problem, different strategies have been proposed. One strategy is to use diverse beam search (Vijayakumar et al., 2018; Zhang and Zhu, 2021) or sampling techniques (such as top-p nucleus sampling used in (Sultan et al., 2020)). While these methods showed promising results, the user has to specify the bin/sample size and the number of questions to be generated (whose ideal number may depend on the input sequence). Another strategy for generating diverse questions is to mark or extract keywords in the input text and generate questions by conditioning on keywords or keyword positions (e.g., (Pan et al., 2020; Shen et al., 2020; Subramanian et al., 2018; Sun et al., 2018; Song et al., 2018; Zhang and Zhu, 2021)). However, extracting keywords (either automatically or manually) to build keyword-labeled training data often needs domain knowledge, a pre-defined keyword list, or documents beyond the training data. The QG method we propose solves these problems by learning Seq2Seq models with semantic role labeled QAs. It can generate multiple and diverse questions without specifying the number of questions to be generated or requiring keyword-labeled training data.

A recent method that also uses SRL and Seq2Seq models is a 2-step method in (Pyatkin et al., 2021). It tackles role question generation that, given a predicate mention and a passage, generates a set of questions asking about all possible semantic roles of the predicate. It first generates prototype questions for all the roles based on the ontology in PropBank (Palmer et al., 2005). It then trains a BART model to generate all questions (including ones that cannot be answered by the input text) given these prototype questions contextualized over the input text. Both the problem definition and the methodology are very different from ours. Our method does not need to generate prototype questions and we generate only the information-seeking questions that can be answered by the input text.

3 Proposed Methodology

Given a set of answer (i.e., sentence) and question pairs, our goal is to train a model to generate from an unseen sentence one or more questions that can be answered by the sentence. [2]

3.1 Overview of the Method

Our method contains a *Semantic Role Labeler (SRLer)*, a Seq2Seq model, and two semantic mappers (namely, *Question2SRL* and *SRL2Question*). First, *SRLer* extracts semantic representations (i.e., SRL labels) from answers in the training set. Then *Question2SRL* maps questions in the training set to their corresponding semantic representations. Next, a Seq2Seq model is trained using these semantic representations to convert an SRL representation of an answer to that of a question. In the inference stage, *Semantic Role Labeler* extracts semantic representations ($\widehat{a}_{sem}$) of an answer $\widehat{a}$. Then, $\widehat{a}_{sem}$ is converted to an SRL representation of a question ($\widehat{q}_{sem}$) by the learned Seq2Seq model. Finally, *SRL2Question* converts $\widehat{q}_{sem}$ into a natural language question $\widehat{q}$.

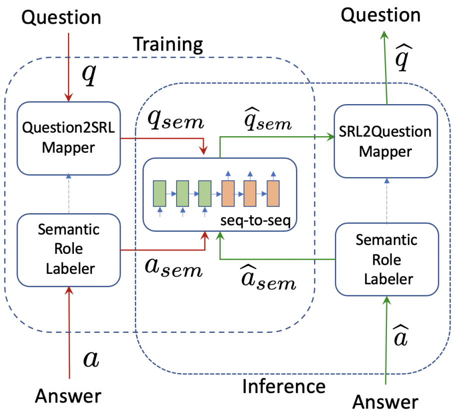


Figure 1: Overview of Proposed Framework

3.2 Semantic Role Labeler

A Semantic Role Labeler (SRLer) is used in both training and inference. In the training phase, an SRLer is used to convert each answer a in the training set into its semantic representation (denoted as a_{sem}) that contains semantic role labels (SRL).

An SRLer recognizes the predicate-argument structure of a sentence and assigns labels (i.e., semantic roles, such as agent, goal or result) to words or phrases in a sentence. We use *SRL BERT* (Shi and Lin, 2019) provided in AllenNLP (Gardner et al., 2017) to produce the semantic labels for the input text, which is the state-of-the-art model for SRL extraction[3]. This method generates a predicate-argument structure for a sentence based on a BERT-based approach. In this model, each sentence is represented by one or more propositions, consisting of a predicate (usually a verb) and its semantic arguments. For example, the semantic representation of sentence *"ABS is activated during braking under certain road or stopping conditions"* is "[ARG1] *is activated* [ARGM-TMP]", where [ARG1] (representing patient) and [ARGM-TMP] (representing time) are semantic role labels for *ABS* and *during braking under certain road or stopping conditions*, respectively.

Table 1 provides more examples of semantic representations for answers. Note that if a sentence contains more than one verb, more than one semantic representation may be generated. For example, S3 and S4 in Table 1 are from the same sentence.

3.3 The Question2SRL Mapper

The *Question2SRL* mapper converts a question q in the training data into its semantic representation q_{sem}. Instead of applying an SRLer directly on q, *Question2SRL* uses the semantic role labels in the semantic representation a_{sem} of q's corresponding answer a to label the phrases or words in q. We design two approaches for *Question2SRL*, namely, *Hard-Question2SRL* and *Soft-Question2SRL*:

Hard-Question2SRL: In this approach, for each semantic role label l that occurs in an answer's SRL representation, if its corresponding phrase or word occurs in the question, the phrase or word in the question is replaced with label l. The reason why we did not use semantic role labeling to directly label the question is that we would like to keep the question words (e.g., what, where, when, etc.) in the semantic representation of the question, and also that semantic role labeling may generate labels for a question which do not occur in its answer. The lower part of Table 1 provides the semantic representations of the questions corresponding to the answer SRL representations in the upper part of the table. Note that Q3 and Q4 are two different representations of the same question, resulting from two different SRL representations of the same answer (S3 and S4). Thus, one original training example

---

[2] We work on sentence-level QG instead of paragraph-level QG because the paragraphs in our application domain (car manuals) are often short and contain a single sentence. Also, an answer sentence in our car manuals dataset does not contain answer words labels, which is different from most paragraph-level QG systems where answer words/phrases are marked in an input paragraph. Sentence-level QG can be extended to paragraph-level QG via sentence segmentation after coreference resolution.

[3] We also used the *Clear Parser SRL* (Choi and Palmer, 2011) in our experiments. *SRL BERT* leads to better results. In this paper we report the results from *SRL BERT*.

Table 1: Semantic representation of answers and questions (ARG0: agent , ARG1: patient, ARG2: attribute, ARGM-NEG: negation, ARGM-PRP: purpose)

Semantic representation for sample input sentences (answers):
S1. [ARG1: the fuel filler funnel] is [ARG2: under the luggage compartment floor covering] .
S2. [ARG0: this vehicle] has a capless refueling system and does [ARGM-NEG: not] have [ARG1: a fuel cap] .
S3. distribute [ARG1: the trailer load] [ARGM-PRP: so 10 - 15 % of the total trailer weight is on the tongue] .
S4. distribute the trailer load so [ARG1: 10 - 15 % of the total trailer weight] is [ARG2: on the tongue] .

Semantic representation for the questions corresponding to the above sentence representations in the training data:
Q1: where is [ARG1: the fuel filler funnel] ?
Q2: does [ARG0: this vehicle] have [ARG1: a fuel cap] ?
Q3: how much of [ARG1: the trailer load] should be on the tongue ?
Q4: how much of the trailer load should be [ARG2: on the tongue] ?

can be converted to one or more SRL-labeled examples for training a Seq2Seq model, resulting in an increase in the size of training data.

***Soft-Question2SRL**:* The words/phrases labeled with a semantic role in an answer may not occur exactly in the question, but their synonyms or similar expressions may. In this case, *Hard-Question2SRL* may not find the exact match in the question. To address this issue, we design *Soft-Question2SRL* that considers the semantic similarity between the words/phrases corresponding to a semantic role label in a_{sem} and potential words/phrases in a question to find the best match. Algorithm 1 outlines the procedure. Given a set of SRLs (L) generated for answer a by *Semantic Role Labeler*, and question q, we first generate all possible n-grams (nG) from q, then compare the phrases/words corresponding to each label $l \in L$, denoted by $words(l)$, with each n-gram $ng \in nG$. The n-gram with maximum similarity with $words(l)$ is selected to be replaced by l as long as the similarity is greater than or equal to a threshold α. We calculate the similarity between n-grams and semantic role label words based on cosine similarity between their corresponding *Sentence-BERT* embeddings (Reimers and Gurevych, 2019).

Table 2 shows how the algorithm works using an example. The first box of the table illustrates the answer a, and its respective question q. Each subsequent box shows a semantic role label $l \in L$ in a_{sem}, the best n-gram ng matching with $words(l)$, and their respective similarity score. The final box shows the semantic representation of question q_{sem} after replacing n-grams with SRLs in previous steps. Note that we replace the n-gram with an SRL label if the score is higher than or equal to a threshold (i.e., α).

Algorithm 1: *Soft-Question2SRL*

```
Input  : $L$  // a set of SRLs generated for $a$
         $q$  // question
Output : $q_{sem}$  // semantic rep. of $q$
$nG \leftarrow$ Generate all n-grams from $q$
for each $l \in L$ do
    $score \leftarrow 0$
    for each $ng \in nG$ do
        $sim \leftarrow$ Cosine($words(l)$, $ng$)
        if $score < sim$ then
            $score \leftarrow sim$
            $l_{best} \leftarrow l$
    if $score \geq \alpha$ then
        $q_{sem} \leftarrow$ replace $ng$ in $q$ with $l_{best}$
```

3.4 Sequence-to-Sequence Learning

Given a set of $\langle a_{sem}, q_{sem} \rangle$ pairs (where a_{sem} and q_{sem} are an SRL-labeled answer and an SRL-labeled question, respectively), we train a Seq2Seq model to convert a_{sem} to q_{sem}. Any Seq2Seq model can be used for this purpose. In our experiment, we use the state-of-the-art models T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) to evaluate our method.

We feed the Seq2Seq model with a_{sem} as the input sequence, where some of words/phrases in the original answer a are replaced by SRLs. While SRLs provide the Seq2Seq model with useful information, this replacement may reduce the information to which model is exposed. As another strategy, we add the actual answer a as a context to the input sequence. That is, the input to the Seq2Seq model is $\langle answer$: a_{sem} $context$: $a\rangle$, where $answer$: and $context$: are tokens prepended to a_{sem} and a, respectively. Alternatively, we can add the SRL labels of the actual answer a and their corresponding

Table 2: An example of *Soft-Question2SRL* mapper converting a question into its semantic representation. Each row shows the best score and corresponding n-gram in the question ($\alpha = 0.85$).

Semantic representation (a_{sem}) for sample answer sentence a and its corresponding question q: a_{sem} = [ARGM-TMP: in january 2009] , [ARG0: the green power partnership -lrb- gpp , sponsored by the epa -rrb-] [V: listed] [ARG1: northwestern] [ARG2: as one of the top 10 universities in the country in purchasing energy from renewable sources] . q = in 2009 , who named northwestern as one of the top 10 universities in the country in purchasing renewable energy ?
l = [ARGM-TMP], $words(l)$ = "in january 2009" *Best matching ng* = "in 2009", $Similarity\ score = 0.91 > \alpha \Rightarrow$ **Replace** ng **with** l **in** q_{sem}
l = [ARG0], $words(l)$ = "the green power partnership -lrb- gpp , sponsored by the epa -rrb- *Best matching ng* = "in purchasing renewable energy ?", $Similarity\ score = 0.46 < \alpha$
l = [V], $words(l)$ = " listed" *Best matching ng* ← "in", $Similarity\ score = 0.42 < \alpha$
l = [ARG1], $words(l)$ = "northwestern" *Best matching ng* = "northwestern", $Similarity\ score = 1.00 > \alpha \Rightarrow$ **Replace** ng **with** l **in** q_{sem}
l = [ARG2], $words(l)$ = "as one of the top 10 universities in the country in purchasing energy from renewable sources" *Best matching ng* = "as one of the top 10 universities in the country in purchasing renewable energy", $Similarity\ score = 0.98 > \alpha \Rightarrow$ **Replace** ng **with** l **in** q_{sem}
q_{sem} = ARGM-TMP , who named ARG1 ARG2 ?

words separated by a special token <*sep*> to the input sequence. The input to the Seq2Seq model is $\langle answer\colon a_{sem}$ <*sep*> *label words* <*sep*>$\rangle$.

Table 3 shows a training example with three variations of the input. We will compare the three input versions of the Seq2Seq model in the experiment.

3.5 The SRL2Question Mapper

In the inference phase, the SRLer is first used to convert an input sentence (i.e., answer $\widehat{a}$) into its semantic representation ($\widehat{a}_{sem}$). Then, the trained Seq2Seq model is used to convert $\widehat{a}_{sem}$ into a semantic representation of a question ($\widehat{q}_{sem}$). After that, the *SRL2Question* mapper transforms all the semantic role labels in the generated semantic representation $\widehat{q}_{sem}$ into words or phrases. In particular, for each semantic role label l in the semantic representation $\widehat{q}_{sem}$ generated by the Seq2Seq model, the *SRL2Question* mapper looks for label l in all the semantic representations $\widehat{a}_{sem}$ of the input sequence $\widehat{a}$, and uses the phrase or word in $\widehat{a}$ that corresponds to l to replace l in $\widehat{q}_{sem}$. Table 4 shows examples of generated semantic representations and converted questions, together with their input sentences, semantic representations of the input sentences, and ground truth questions.

4 Empirical Evaluation

We investigate (1) whether using SRL with Seq2Seq models improves the QG performance of Seq2Seq without SRL, and (2) how our method compares with the state-of-the-art QG methods.

4.1 Datasets

We evaluate our method on 3 datasets[4]. The first one contains QAs created by human annotators from two car manuals of Ford and GM, denoted as Car Manuals. The second dataset is SQuAD (Rajpurkar et al., 2016), containing QAs created by Amazon Mechanical Turk crowd-workers from Wikipedia articles. We use the processed sentence-level SQuAD dataset (Du et al., 2017), where the answer in a QA pair is a single sentence. The third dataset is NewsQA (Trischler et al., 2016), a machine comprehension dataset of human-generated QA pairs from CNN news articles. We created a sentence-level NewsQA dataset. The sentences were extracted from corresponding paragraphs based on their answer span. All the datasets contain training, testing and development sets. Their statistics are given in Table 5. In these datasets, multiple QA pairs may have the same answer sentence but different questions. For example, two QA pairs in SQuAD share "alfred north whitehead was born in ramsgate, kent, england, in 1861" as the answer, but their questions (i.e., "in what year was whitehead born?" and "where was alfred north whitehead born?") are different and cover different aspects of the answer sentence.

4.2 Automatic Evaluation Metrics

For automatic evaluation, we use the $precision$, $recall$ and F scores proposed in (Schlichtkrull and Cheng, 2020)[5] for measuring the quality and diversity of generated questions. Given a test example

---

[4] The code and data sets are available at `https://github.com/Naeiji/QGwSRL`

[5] These scores are called u, v and F scores in (Schlichtkrull and Cheng, 2020).

Table 3: A training example for Seq2Seq model with 3 different strategies for input (i.e., answer) representations.

input	*answer*: [ARG1] was named [ARG2], [ARGM-PRD].
input+C	*answer*: [ARG1] was named [ARG2], [ARGM-PRD]. *context*: denver linebacker von miller was named super bowl mvp, recording five solo tackles, 2 1/2 sacks, and two forced fumbles.
input+L	*answer*: [ARG1] was named [ARG2], [ARGM-PRD]. <*sep*> [ARG1] denver linebacker von miller <*sep*> [ARG2] super bowl mvp <*sep*> [ARGM-PRD] recording five solo tackles , 2 1/2 sacks , and two forced fumbles <*sep*>
output	who won the [ARG2] ?

Table 4: Examples of sentence $\widehat{a}$, its semantic representation $\widehat{a}_{sem}$, the outcome $\widehat{q}_{sem}$ generated by Seq2Seq, the question $\widehat{q}$ converted from $\widehat{q}_{sem}$, and ground-truth question Q_t from the Car Manuals dataset.

$\widehat{a}$:	before placing a child in the child restraint , make sure it is securely held in place .
$\widehat{a}_{sem}$:	before placing [ARG1] [ARG2] , make sure it is securely held in place .
$\widehat{q}_{sem}$:	what should i do before placing [ARG1] [ARG2] ?
$\widehat{q}$:	what should i do before placing a child in the child restraint ?
Q_t:	what should i do before placing a child in the child restraint ?
$\widehat{a}$:	adjust the temperature setting using the + and - temperature buttons on the right-hand side of the climate controls .
$\widehat{a}_{sem1}$:	adjust the [ARG1] setting using the + and - temperature buttons on the right-hand side of the climate controls .
$\widehat{a}_{sem2}$:	adjust the temperature setting using [ARG1] [ARGM-LOC] .
$\widehat{q}_{sem1}$:	how do i adjust the [ARG1] setting ?
$\widehat{q}_{sem2}$:	how do i adjust the temperature [ARGM-LOC] ?
$\widehat{q}_1$:	how do i adjust the temperature setting ?
$\widehat{q}_2$:	how do i adjust the temperature on the right-hand side of the climate controls ?
Q_t:	how do i adjust the temperature on the passenger 's side ?

Table 5: Statistics of Three Datasets

Dataset	training set	test set	development set
Car Manuals	9,184 QAs	1,869 QAs	1,403 QAs
SQuAD	70,484 QAs	11,877 QAs	10,570 QAs
NewsQA	91,536 QAs	5,067 QAs	5,136 QAs

consisting of input text a and a set of ground-truth questions T, the $precision$ and $recall$ of a set of questions G generated from a by a QG method are defined as:

$$precision(G,T,s) = \frac{1}{|G|}\sum_{g\in G}\max_{t\in T} s(g,t)$$

$$recall(G,T,s) = \frac{1}{|T|}\sum_{t\in T}\max_{g\in G} s(g,t)$$

where s is a scoring function that measures the similarity between two questions. We use the similarity function used in BLEU-n (n-gram text overlap), ROUGE-L (longest common subsequence text overlap) and METEOR (which takes into account word re-ordering, stemming, synonyms, and paraphrase matching) to compute s. *F-score* is defined as the harmonic mean of precision and recall.

4.3 Comparison of Seq2Seq+SRL with Seq2Seq methods

To investigate whether the use of SRL with Seq2Seq models improves the QG performance of Seq2Seq models trained with original sentences, we conducted experiments with two state-of-the-art transformer-based Seq2Seq models: (1) **BART** (Lewis et al., 2020) and (2) **T5** (Raffel et al., 2020).

We fine-tune the Huggingface pretrained models (Wolf et al., 2019) of BART-base and T5-small with 139 and 60 million parameters respectively. We use the smallest available model sizes of BART and T5 to avoid GPU memory error. Four V100-SXM2 32GB GPUs are used for fine-tuning. For baselines, T5 and BART are fine-tuned using the original sentence-based QA pairs in each training set to generate a question given an answer.

For our method, we use a pre-trained SRL BERT model (Shi and Lin, 2019) provided in AllenNLP (Gardner et al., 2017) as the Semantic Role Labeler to convert answer sentences to their SRL representations. We then fine-tune T5 or BART to learn a model that maps an SRL representation of an input sentence to that of the question, which is then mapped to a natural language question.

To determine the values of hyperparameters in the Seq2Seq model (T5 and BARR) in either baseline or our model, we conduct random search (Bergstra and Bengio, 2012) due to its efficiency. To do so, we randomly select 10 combinations of learning rates (LR) and epoch numbers (EP) as follows: "LR = 5×10^{-6}, EP = 2"; "LR = 10^{-5}, EP = 8"; "LR = 5×10^{-5}, EP = 5"; "LR = 5×10^{-5}, EP = 10"; "LR = 10^{-4}, EP = 10"; "LR = 10^{-4}, EP = 4"; "LR = 5×10^{-4}, EP = 3"; "LR = 5×10^{-4}, EP = 7"; "LR = 10^{-4}, EP = 4"; and "LR = 10^{-4} EP = 10". The best model for each dataset and each method is chosen based on the final loss of the development set. We combine an actual batch size of 16 with 2 gradient accumulation steps per minibatch to artificially create a batch size of 32. The maximum sequence length is set to 512 for both input and output. For decoding method, we use beam search with a beam size of 10.

4.3.1 Results of Automatic Evaluation

Table 6 shows the automatic evaluation results on Car Manuals with BART and T5 as the baseline. **Hard** is our method using *Hard-Question2SRL* for *Question2SRL*. **Soft** is our method with *Soft-Question2SRL* with 80% as the soft-matching threshold (α). **Soft+C** and **Soft+L** are our methods with *Soft-Question2SRL* plus using the original question or SRL labels and their corresponding words as the context, respectively.

Table 6: Automatic evaluation results on **Car Manuals** with **BART** and **T5** as Baselines (P, R and F mean Precision, Recall and F-score in %). Hard and Soft+ methods are variations of our method with BART or T5 as the Seq2Seq model.

	BLEU-4			ROUGE-L			METEOR		
QG Method	F	P	R	F	P	R	F	P	R
BART	57.2	63.7	51.9	71.7	75.9	68.0	57.6	63.7	52.6
Hard	70.9	68.3	73.6	76.4	75.7	77.1	58.7	57.5	59.9
Soft	82.6	76.3	90.1	90.6	87.9	93.5	58.8	55.7	62.2
Soft+C	88.3	**85.4**	91.4	94.3	**94.0**	94.6	63.0	**62.1**	63.9
Soft+L	**89.0**	85.1	**93.2**	**94.8**	93.7	**95.9**	**63.6**	61.8	**65.5**
T5	45.0	50.3	40.7	62.4	66.0	59.2	46.3	50.0	43.1
Hard	63.9	58.8	70.0	71.5	69.1	74.0	52.3	49.3	55.7
Soft	77.0	71.5	83.5	85.9	82.7	89.4	53.6	50.4	57.1
Soft+C	**85.9**	**84.1**	**87.8**	**91.9**	**91.5**	**92.4**	**59.9**	**59.3**	**60.5**
Soft+L	84.7	82.8	86.6	91.0	90.1	92.0	58.8	57.6	60.0
Rule Based	17.4	13.9	23.2	36.2	31.6	42.5	22.0	18.5	27.1

As shown in Table 6, using SRL representations to train a QG model with either BART or T5 significantly improves the baseline performance on Car Manuals. This is due to generalization of sentences into SRL representations, allowing general semantic patterns to be modeled and used in QG. The use of SRLs also increases the number of training examples due to the fact that the SRLer can convert a sentence into multiple SRL-labeled sentences. All the variations of our method significantly increase the recall and F-scores in all types of measures (BLEU, ROUGE and METEOR) and they also increase precision in almost all of the cases. The recall increase is due to the fact that the SRLer can convert a sentence into multiple SRL-labeled sentences, leading to questions asking about different aspects of the sentence.

Comparing the hard and soft versions of our method, Soft-Question2SRL is better than Hard-Question2SRL in all cases, indicating that allowing different but similar expressions in the corresponding question and answer when labelling the question with SRLs is important and beneficial. We also see that the use of the original source sentence or SRL labels as a context in the input is beneficial. This is probably because the error propagation from semantic role labeling has less impact when (part of) the original sentence are given as a context.

Table 7: Automatic evaluation results on **SQuAD** with **BART** and **T5** as baselines (P, R and F mean Precision, Recall and F-score). Hard and Soft+ methods are different variations of our method with BART and T5 as the Seq2Seq models.

	BLEU-4			ROUGE-L			METEOR		
QG Method	F	P	R	F	P	R	F	P	R
BART	15.2	**21.4**	11.7	42.9	**48.0**	38.8	20.6	**22.6**	18.9
Hard	17.8	17.9	17.6	45.0	44.0	46.2	21.4	20.3	22.6
Soft	19.6	18.5	**20.9**	46.6	44.3	49.1	22.4	21.0	24.0
Soft+C	19.7	20.5	19.0	47.4	47.0	47.9	22.9	21.8	24.1
Soft+L	**20.0**	20.2	19.9	**48.1**	46.9	**49.3**	**23.3**	21.8	**25.0**
T5-single	15.0	19.9	12.0	41	46.3	36.7	19.5	**23.2**	16.7
Hard	16.6	16.1	17.1	43.4	42.2	44.5	20.3	19.6	21.1
Soft	18.0	16.6	**19.8**	44.8	43.0	46.8	21.5	20.9	22.2
Soft+C	19.0	19.6	18.4	45.7	45.7	45.8	22.1	22.2	22.0
Soft+L	**19.9**	**20.4**	19.4	**48.0**	**47.1**	**48.9**	**23.2**	21.8	**24.8**

Tables 7 shows the automatic evaluation results on the SQuAD dataset with BART and T5 as the baseline. Again, we observe that all variations of our method significantly outperform the baseline on recall and F-score. On this data set, the BART baseline shows the best precision. However, the lower precision of our method is due to the incompleteness of the ground truth questions in SQuAD. That is, many of the questions our method generates are good questions, but they do not match the ground truth questions in the data set[6].

Table 11 in Appendix B shows the generated questions for 3 testing examples from our method (Soft+L) with T5 and T5 without SRL. As shown, our method generates more questions covering different aspects of an input sentence, while T5 without SRL generates only one question. In all the 3 examples, there is only one ground-truth question. When precision is computed, the extra questions we generated have a low precision due to poor match with the ground truth even though they are good questions. This explains why our methods have lower precision scores than the baselines.

The results on this dataset also show that soft matching is better than hard matching, and the use of context is better with T5, but has no obvious

---

[6] Such a problem is also mentioned by others such as in (Sultan et al., 2020)

advantage for BART. We show the impacts of soft-matching threshold α in Appendix E. In practice, α can be tuned using the development set.

Similar results are observed on the NewsQA dataset, as shown in Table 12 in Appendix C.

4.3.2 Human Evaluation

We randomly selected 50 input sentences from the SQuAD dataset and asked 5 English speakers to rate the quality of generated questions from each method in terms of *recall*, *clarity*, *Q&A relatedness* and *grammar*. All the criteria are rated based on a 5-point scale. A precision score for each question is computed by averaging the scores for clarity, Q&A relatedness and grammar. An overall score (F score) for each test input sentence is computed as the harmonic mean of precision and recall scores. Detailed information on the questionnaire we provided to the human evaluators is provided in Appendix 4.3.2.

Table 8 shows the average scores among the human evaluators. The results show that our methods (Soft+L with BART and T5+Soft with T5) are significantly better than their corresponding baseline in recall and F-measure, and they are also better than the baselines in precision.

4.4 Comparison with other SOTA methods

We compare our method with additional SOTA baselines, most of which can generate multiple questions from an input sentence:

- **BART-multi** and **T5-multi**. BART or T5 model fine-tuned to generate multiple sequences given an input sequence by using <*sep*> tokens to separate the ground-truth questions in the output part of each training example.
- **BART-divbeam** and **T5-divbeam**. BART or T5 model that uses decoding-based diverse beam search (Vijayakumar et al., 2016) to generate multiple questions. We use beam size of 6 with 3 diverse groups and diversity penalty of 0.4 all same as in (Zhang and Zhu, 2021). We select 3 best questions for the evaluation[7].
- **ProphetNet-single** and **ProphetNet-multi**. ProphetNet (Qi et al., 2020) is another Transformer-based SOTA model. In ProphetNet-single, ProphetNet is fine-tuned to generate a single question. In ProphetNet-multi, it is fine-tuned to generate multiple sequences from an input sequence by using <*sep*> tokens to separate the ground-truth questions.
- **MP-GSN** (Zhao et al., 2018) An LSTM-based Seq2Seq QG model. We use the sentence-level MP-GSN with the default setting in the implementation of MP-GSN in (Lee, 2019).
- **Rule-based method** with 75 rules based on semantic role labels to convert a sentence into questions.[8]

The hyper-parameters for all the Seq2Seq methods are determined through random search (described in Section 4.3).

Table 9 shows the results of these SOTA methods on SQuAD compared to our method (Soft+L) with BART or T5 as the Seq2Seq model. As shown, our methods outperform all the baselines in F-score and recall. It is also the best in precision in all underlying metrics except for precision using METEOR where ProphetNet-single is the best and our methods are the second best.

Compared to Seq2Seq-multi, our methods outperform them significantly in all metrics, indicating using SRL to generate multiple questions is much more effective than using the <*sep*> tokens in the output parts of the training sequences.

Our methods also outperform the diverse-beam-search-based methods for generating multiple questions, indicating different SRL representations of a sentence can more effectively lead to generation of diverse questions that cover different aspects of the input sentence.

Compared to the rule-based method, our method outperforms it significantly on all measures with big margins. This indicates that SRL-based Seq2Seq model better captures the relations between SRL representations of questions and those of answers than rules, which complicated "rules" can be learned by Seq2Seq models.

5 Performance with Different Data Sizes

Among the 3 datasets, Car Manuals is the smallest, and the improvements of our method on this dataset over the baselines are the largest. To further investigate whether our method makes better improvement when the data set is small, we conducted an experiment with one quarter of the SQuAD dataset.

---

[7] We select top-3 questions because the average number of questions generated by our method per input sentence is 3 (see Table 10 for more details), to make the comparison fair. In general, the more questions are generated, the lower the precision but better the recall.

[8] An example rule is "Replace [ARG1] with *what* if it appears at the beginning of the sentence".

Table 8: Human evaluation results on 50 test sentences from SQuAD. Precision is the average of Clarity, Relatedness and Grammar scores. F-measure is the harmonic mean of precision and recall scores

QG System	F-measure Score (1-5) ± stdev	Precision Score (1-5)	Recall Score (1-5)	Clarity Score (1-5)	Relatedness Score (1-5)	Grammar Score (1-5)
BART	3.64 ± 0.33	4.73	3.06	4.70	4.56	**4.94**
Soft+L (with BART)	**4.32 ± 0.28**	**4.77**	**4.16**	**4.75**	**4.61**	4.93
T5	3.63 ± 0.37	4.71	3.08	4.66	**4.63**	4.85
Soft+L (with T5)	**4.40 ± 0.29**	**4.73**	**4.25**	**4.72**	4.48	**4.90**

Table 9: Comparison with SOTA models on **SQuAD**

	BLEU-4			ROUGE-L			METEOR		
QG Method	F	P	R	F	P	R	F	P	R
BART-multi	15.2	18.4	12.9	41.8	44.5	39.3	20.0	21.2	18.8
T5-multi	14.9	16.5	13.6	40.5	42.2	39.0	19.3	20.9	17.9
BART-divbeam	17.5	19.2	16.1	46.7	46.9	46.5	22.6	21.7	23.5
T5-divbeam	17.6	19.2	16.3	46.3	46.4	46.1	22.4	21.8	23.1
ProphetNet-single	16.2	20.3	13.5	43.0	46.4	40.2	21.2	**22.7**	19.8
ProphetNet-multi	14.5	17.7	12.3	39.6	43.7	36.2	19.8	21.3	18.5
MP-GSN	12.1	17.0	9.5	39.5	43.9	35.8	17.7	19.1	16.5
Rule Based	9.1	8.0	10.4	25.1	23.0	27.5	15.0	14.5	15.4
Soft+L (Ours with BART)	**20.0**	20.2	**19.9**	**48.1**	46.9	**49.3**	**23.3**	21.8	**25.0**
Soft+L (Ours with T5)	19.9	**20.4**	19.4	48.0	**47.1**	48.9	23.2	21.8	24.8

Figure 3 in Appendix F illustrates the relative improvement of our method over the T5 baseline in F-measure, precision and recall. The relative improvement is computed as: $\frac{score_{our}-score_{t5}}{score_{t5}}$. The blue bars in the figure represent the improvement on the one-quarter subset of SQuAD and yellow ones represent that on the whole SQuAD dataset. Clearly, the improvements of our method over the T5 baseline are larger on the smaller dataset than on the whole SQuAD dataset. This indicates that our SRL-based Seq2Seq method can better handle smaller datasets than its Seq2Seq baselines due to the increase in training examples when we label the QAs with SRLs.

6 Conclusions

We proposed a novel QG method that learns a Seq2Seq model to convert an SRL representation of an input sentence into an SRL representation of a question, which is then converted to a natural language question. Similar to rule-based methods, our SRL-based Seq2Seq methods can generate multiple questions from an input sentence, significantly improving the recall and overall performance of Seq2Seq QG. It is also much better than rule-based methods because better and more complicated "rules" can be learned via the Seq2Seq model. Our evaluation on three real-world datasets shows that the proposed method significantly outperforms both rule-based, original Seq2Seq methods and several other SOTA models, especially in recall and overall performance. As future work, we will extend this method to paragraph-based question generation.

Limitations

A limitation of our method is that its performance depends on the SRL performance. If the input sentences are not well-formed, semantic role labeling may not produce correct labels. Also, we found that some of the questions generated by our method for the same input sentence may be similar or same in meaning with some minor differences in the use of words. This may not be a problem if the application allows similar questions to be generated (e.g., for reading compression). But in the application where duplicated questions are not allowed or not desired, a post-processing step to remove questions with the same meaning is needed.

Acknowledgements

We would like to thank iNAGO Corporation for providing the Car Manuals dataset used in this research and for their collaboration on the research topic. This work is supported by an NSERC Alliance grant from the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, and Hsiao-Wuen Hon. 2020. UniLMv2: Pseudo-masked language models for unified language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 642–652. PMLR.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).

Yllias Chali and Sadid A. Hasan. 2015. Towards topic-to-question generation. *Computational Linguistics*, 41(1):1–20.

Guanliang Chen, Jie Yang, Claudia Hauff, and Geert-Jan Houben. 2018. Learningq: a large-scale dataset for educational question generation. In *Twelfth International AAAI Conference on Web and Social Media*.

Jinho D. Choi and Martha Palmer. 2011. Transition-based semantic role labeling using predicate argument clustering. In *Proceedings of the ACL 2011 Workshop on Relational Models of Semantics*, RELMS '11, page 37–45, USA. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352.

Michael Flor and Brian Riordan. 2018. A semantic role-based approach to open-domain automatic question generation. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 254–263, New Orleans, Louisiana. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Jing Gu, Mostafa Mirshekari, Zhou Yu, and Aaron Sisto. 2021. ChainCQG: Flow-aware conversational question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2061–2070, Online. Association for Computational Linguistics.

Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. 2018. Automatic question generation using relative pronouns and adverbs. In *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158.

Vishwajeet Kumar, Kireeti Boorla, Yogesh Meena, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. Automating reading comprehension by generating question and answer pairs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 335–348. Springer.

Seanie Lee. 2019. Pytorch implmentation of paragraph-level neural question generation with maxout pointer and gated self-attention networks. `https://github.com/seanie12/neural-question-generation`.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 105–114.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Youcheng Pan, Baotian Hu, Qingcai Chen, Yang Xiang, and Xiaolong Wang. 2020. Learning to generate diverse questions from keywords. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8224–8228.

Valentina Pyatkin, Paul Roit, Julian Michael, Yoav Goldberg, Reut Tsarfaty, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1429–1441, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Michael Sejr Schlichtkrull and Weiwei Cheng. 2020. Evaluating for diversity in question generation over text. *arXiv preprint arXiv:2008.07291*.

Sheng Shen, Yaliang Li, Nan Du, X. Wu, Yusheng Xie, Shen Ge, Tao Yang, Kai Wang, Xingzheng Liang, and Wei Fan. 2020. On the generation of medical question-answer pairs. In *AAAI*.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 78–88, Melbourne, Australia. Association for Computational Linguistics.

Md Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5651–5656, Online. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *ArXiv*, abs/1610.02424.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *AAAI*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3997–4003. International Joint Conferences on Artificial Intelligence Organization. Main track.

Xingdi Yuan, Tong Wang, Adam Peter Trischler, and Sandeep Subramanian. 2019. Neural models for key phrase detection and question generation. US Patent App. 15/667,911.

Zhiling Zhang and Kenny Q. Zhu. 2021. Diverse and specific clarification question generation with keywords. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3501–3511. ACM / IW3C2.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

A Average Number of Questions Generated by our Methods

Table 10: Average numbers of generated questions per source sentence by our method with BART and T5

QG Method	Avg. # of questions
BART+Soft	3.3
BART+Soft+C	2.8
BART+Soft+L	3.0
T5+Soft	3.3
T5+Soft+C	2.7
T5+Soft+L	3.0

Table 10 provides the average number of questions generated by different versions of our methods on the SQuAD dataset.

B Examples of Generated Questions from SQuAD

Table 11 shows 3 examples of an input sentence $\widehat{a}$, its ground-truth question (q_g) and the generated questions ($\widehat{q_i}$) from our method (Soft+C) with T5 compared to the question (g_t) generated from the T5 baseline without the use of SRLs. As shown, our method generates more questions covering different aspects of the input sentence, while T5 without SRL generates only one question. In all these 3 examples, there is only one ground-truth question in the data set. When precision is computed, the extra

Table 11: Three Examples showing the input sentence $\widehat{a}$, its ground-truth question q_g, the question generated by T5 (q_t), and the questions $\widehat{q_i}$ generated by our Soft+L method with T5 and alpha=80% on the SQuAD dataset

$\widehat{a}$:	there are two categories of repetitive dna in genome : tandem repeats and interspersed repeats .
q_g:	what are two types of repetitive dna found in genomes ?
q_t:	what are the two categories of repetitive dna in genome ?
$\widehat{q}_1$:	what are the two categories of repetitive dna in the genome ?
$\widehat{q}_2$:	how many categories of repetitive dna are there in the genome ?
$\widehat{a}$:	before the solar/wind revolution , portugal had generated electricity from hydropower plants on its rivers for decades .
q_g:	through what renewable resource had portugal generated electricity before the solar/wind revolution ?
q_t:	before the solar/wind revolution, portugal had generated electricity from what ?
$\widehat{q}_1$:	what had portugal generated electricity from before the solar/wind revolution ?
$\widehat{q}_2$:	how long had portugal generated electricity from hydropower plants on its rivers ?
$\widehat{a}$:	the term parinirvana is also encountered in buddhism , and this generally refers to the complete nirvana attained by the arahant at the moment of death , when the physical body expires .
q_g:	what term is used for the complete nirvana attained by the arahant at death ?
q_t:	what is the term parinirvana used in buddhism ?
$\widehat{q}_1$:	what term is also encountered in buddhism ?
$\widehat{q}_2$:	when is the complete nirvana attained by the arahant ?
$\widehat{q}_3$:	who attained the complete nirvana at the moment of death ?
$\widehat{q}_4$:	what does the term parinirvana refer to at the moment of death ?
$\widehat{q}_5$:	what term is used in buddhism to describe the complete nirvana attained by the arahant at the moment of death ?

questions we generated have a low precision due to poor match with the ground truth even though they are good questions. This explains why our methods sometimes have lower precision scores than the baseline method in automatic evaluation. This unfairness is due to the incompleteness of ground truth questions in the SQuAD dataset.

C Results on NewsQA Dataset

Table 12: Automatic evaluation results on **NewsQA** with **BART** and **T5** as Baselines (P, R and F mean Precision, Recall and F-score). Hard and Soft+ methods are different variations of our method with BART and T5 as the Seq2Seq models.

	BLEU-4			ROUGE-L			METEOR		
QG Method	F	P	R	F	P	R	F	P	R
BART-single	10.7	**16.8**	7.8	38.3	**44.7**	33.6	17.6	**20.6**	15.3
Hard	12.2	13.2	11.3	41.4	41.8	41.1	19.0	18.1	20.0
Soft+A80	13.4	12.8	**14**	42.8	41.9	**43.8**	19.9	18.9	**21.1**
Soft+A80+C	**14.8**	16.3	13.6	**43.8**	44.6	43.0	**20.6**	20.4	20.8
T5-single	10.4	**15.8**	7.8	37.6	**44.1**	32.8	17.1	**20.7**	14.6
Hard	11.9	12.6	11.2	40.9	41	40.8	18.8	18.2	19.6
Soft+A80	13.3	12.9	**13.8**	42.6	41.6	43.7	19.8	18.8	**20.8**
Soft+A80+C	**14.1**	15.7	12.9	**42.9**	43.9	42.0	**20.0**	20.3	19.8
Rule Based	4.5	3.8	5.6	22.8	21.8	24.0	12.8	14.2	11.7

Table 12 shows the results of different variations of our method on the NewsQA dataset, compared with BART and T5 baselines.

D Questionnaire for Human Evaluation

We ask the human evaluators to rate the quality of generated questions from each method in terms of *recall*, *clarity*, *Q&A relatedness* and *grammar* on a scale of 1-5 using the following criteria.

For **Recall**, the ratings are:

- 1= Bad: the generated questions do not cover any fact in the answer;
- 2= Unacceptable: the generated questions cover only a small portion of the facts in the answer;
- 3= Borderline: the generated questions cover around 50% of the facts in the answer;
- 4= Acceptable: the generated questions cover most of the facts in the answer; and
- 5= Good: the generated questions cover all the facts in the answer.

For **Clarity**, the ratings are:

- 1= Bad: the question is completely unclear in meaning or makes no sense;
- 2= Unacceptable: the question is mostly unclear;
- 3= Borderline: the question is between unacceptable and acceptable;
- 4= Acceptable: the question is clear and understandable, but the use of words can be improved; and
- 5= Good: the question has no problem. It is clear, simple and uses the right words.

For **Q&A Relatedness**, the ratings are:

- 1= Bad: the question is completely unrelated to the answer sentence it is generated from;
- 2= Unacceptable: the question is somewhat related to the answer sentence, but it cannot be answered by the answer sentence;
- 3= Borderline: the question can be partially answered by the answer sentence, but far from completely;
- 4= Acceptable: the question can be mostly answered by the answer sentence, although maybe not completely; and
- 5= Good: the question can be very well answered by the answer sentence.

For **Grammar**, the ratings are:

- 1= Bad: the grammar of the question is completely wrong;

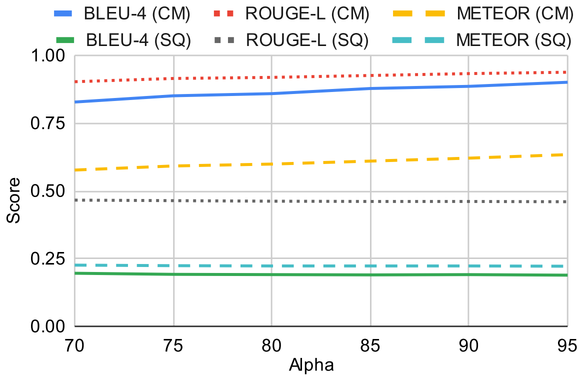


Figure 2: BLEU-4, ROUGE-L and METEOR F-scores of T5 on SQuAD (SQ) and Car Manuals (AM) using different alphas values.

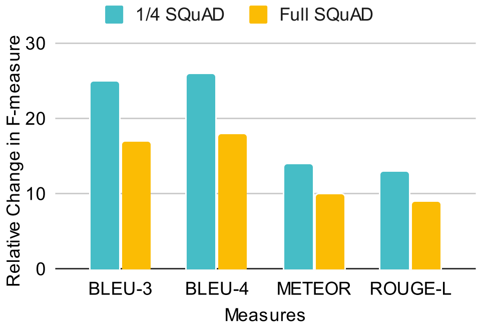
- 2= Unacceptable: the question has major grammatical problems;
- 3= Borderline: the question has a grammatical error, which is between major and minor;
- 4= Acceptable: the question has only a minor grammatical problem; and
- 5= Good: the question is completely grammatically correct.

E Sensitivity Analysis

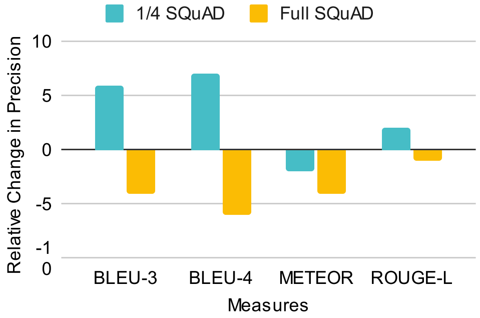
To see how the soft-matching threshold α affects the results, we experiment with different α values ranging from 70 to 95 and use the resulting datasets to fine-tune T5. In this experiment, we use the best configurations of the T5 model on the Car Manuals and SQuAD datasets. Six α values of 70, 75, 80, 85, 90, and 95 are selected for this purpose. Figure 2 shows how the F-score of BLEU-4, ROUGE-L and METEOR changes with α on SQuAD (SQ) and Car Manuals (CM). As can been seen, on SQuAD all the lines are quite flat, indicating α values do not have much impact on the performance. This is likely due to the best-matching n-gram in a ground-truth question either matches with the word/phrase replaced by an SRL in the answer very well or very poorly, leading to insensitivity to the threshold value between 70 and 95. Similar results are observed on the Car Manuals dataset although the scores are increasing with the α on this dataset, but slowly.

F Performance vs Different Data Sizes

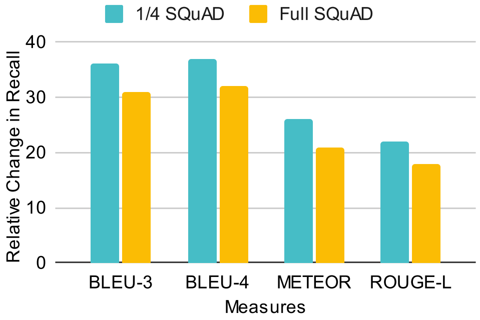
To further investigate whether our method makes better improvement over the baselines when the data set is small, we conducted an experiment with one quarter of the SQuAD dataset. Figure 3 illustrates the relative improvement of our method over the T5 baseline in F-measure, Precision and Recall. The relative improvement is computed as: $\frac{score_{our}-score_{t5}}{score_{t5}}$.



(a) F-measure



(b) Precision



(c) Recall

Figure 3: Comparison of relative change (percent) between the first quarter of SQuAD and whole SQuAD dataset using T5+C versus T5 with alpha=85%.

The blue bars in the figure represent the improvement on the one-quarter subset of SQuAD and yellow ones represent that on the whole SQuAD dataset. Clearly, the improvements of our method over the T5 baseline are larger on the smaller dataset than on the whole SQuAD dataset. This indicates that our SRL-based Seq2Seq method can better handle smaller datasets than its Seq2Seq baselines due to the increase in training examples when we label the QAs with SRLs.

StyLEx: Explaining Style Using Human Lexical Annotations

Shirley Anugrah Hayati🐳 **Kyumin Park**🦭 **Dheeraj Rajagopal**[*]🐙
Lyle Ungar🐠 **Dongyeop Kang**🐳
🐳University of Minnesota 🦭KAIST
🐙Carnegie Mellon University 🐠University of Pennsylvania
hayat023@umn.edu pkm9403@kaist.ac.kr dheeraj@cs.cmu.edu
ungar@cis.upenn.edu dongyeop@umn.edu

Abstract

Large pre-trained language models have achieved impressive results on various style classification tasks, but they often learn spurious domain-specific words to make predictions (Hayati et al., 2021). While human explanation highlights stylistic tokens as important features for this task, we observe that model explanations often do not align with them. To tackle this issue, we introduce **StyLEx**, a model that learns from human annotated explanations of stylistic features and jointly learns to perform the task and predict these features as model explanations. Our experiments show that StyLEx can provide human-like stylistic lexical explanations without sacrificing the performance of sentence-level style prediction on both in-domain and out-of-domain datasets. Explanations from StyLEx show significant improvements in explanation metrics (sufficiency, plausibility) and when evaluated with human annotations. They are also more understandable by human judges compared to the widely-used saliency-based explanation baseline.[1]

1 Introduction

People use style as a strategic choice for their personal or social goals in communications, making style analysis a long-studied field in NLP (Hovy, 1987; Kabbara and Cheung, 2016; Kang and Hovy, 2021). While large language models have obtained state-of-the-art results on many NLP tasks, they have been shown to overfit to spurious correlations in data across several datasets (Sen et al., 2021; Schlangen, 2021; Bras et al., 2020). Hayati et al. (2021) found a phenomenon in style classification tasks where the model's word-level explanation do not align with human's stylistic cues (stylistic cues are words that signify the style of a text). For instance, words such as "performances" and "wrench" in Figure 1 are marked as important cues for sentiment by a saliency method. However, they are different from words that humans perceive as essential features for predicting the style ("top," "notch").

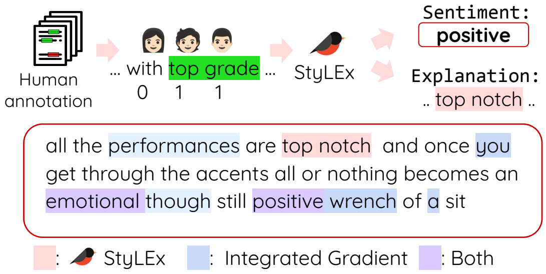


Figure 1: StyLEx classifies the input sentence's style and provides lexical explanation. Compared to explanations computed by the integrated gradient method (Mudrakarta et al., 2018), StyLEx can find more accurate stylistic words. Green highlight refers to human's annotated positive word, pink for StyLEx, blue for baseline, and purple for both StyLEx and the baseline.

Prior research in style have developed stylistic lexicon dictionary to identify the style of a text, such as sentiment or emotion, and in-turn incorporated them for style classification tasks (Mohammad and Turney, 2010; Hutto and Gilbert, 2014; Tausczik and Pennebaker, 2010). While lexicon-based matching methods (Taboada et al., 2011; Eisenstein, 2017) provide interpretability for the task, they lack coverage and do not incorporate the context for prediction. On the other hand, current large scale models like BERT (Devlin et al., 2019) are effective at style classification. However, their explanations often reveal that the model do not rely on the stylistic words to make the prediction. In this work, we hypothesize that leveraging stylistic lexica along with the effectiveness of a large scale model like BERT can not only predict style but also provide meaningful explanations that align with human style explanations.

Towards this, we introduce **StyLEx**, a style classification model that jointly learns to align human-annotated stylistic cues as explanations and then

---

[*] currently at Google

[1] Code and data are publicly available at `https://github.com/minnesotanlp/stylex`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2843–2856
May 2-6, 2023 ©2023 Association for Computational Linguistics

predict the style of the overall sentence based on the cues (Figure 1). StyLEx uses a semi-supervised approach to expand the stylistic words from a handful number of human-annotated stylistics words. First, we train StyLEx using existing small human stylistic word annotation from Hayati et al. (2021). Then, we obtain predicted stylistic words on the larger benchmarking style datasets and retrain StyLEx on this expanded data to predict both the sentence's style label and the stylistic lexical explanation.

In this study, we show that for both in-domain and out-of-domain data, StyLEx not only shows competitive classification performance with BERT-based model, but also generate stylistic lexical explanations that have higher alignment with human explanations. In terms of explanation quality, StyLEx surpasses the baseline method across multiple explanation metrics. For the sufficiency metric, we improve upon the baseline by 14.12% on the average. For plausibility, StyLEx's lexical explanations correlate highly with human lexical annotation ground truth with an average Pearson's r correlation score of 44% compared to the baseline's correlation score of 3.9%. Finally, we found that 72.5% of StyLEx's explanations are more preferred by human judges compared to the baseline.

2 StyLEx: Style Classification with Human Lexical Annotations

2.1 StyLEx Model Architecture

StyLEx is a joint model for word-level and sentence-level style prediction. Unlike a multi-task learning approach where tasks are independent of each other, StyLEx exploits these stylistic word scores obtained from human annotation and then helps predict the sentence's styles. As displayed in Figure 2 (left), StyLEx involves three modules: a transformer-based (Vaswani et al., 2017; Devlin et al., 2019) encoder, a word-level style predictor and a sentence-level style predictor. This work is based on BERT although the encoder can be applied to any transformer architecture.

Given an input of token sequence $\mathbf{x} = \{x_1, ..., x_n\}$ and its corresponding set of stylistic word scores $\{s_1, ..., s_n\}$, we encode $\mathbf{x}$ using a pretrained transformer model. We extract the final layer output as $\mathbf{h} = \{\mathbf{h}_1, ..., \mathbf{h}_n\}$ and feed $\mathbf{h}$ to the word-level style prediction layer which is a neural classifier that outputs stylistic word logits for each word l_{word_i} computed as follows:

$$l_{word_i} = \mathbf{W}_{word}\mathbf{h}_i + \mathbf{b}_{word}$$

where $i \in \{1, .., n\}$, $\mathbf{W}_{word}$ is a matrix with the size $H \times d_{l_{word}}$, and $\mathbf{b}_{word} \in \mathbb{R}^{d_{l_{word}}}$ is the bias term. H is the size of the default hidden layer in BERT which is 768 and $d_{l_{word}}$ denotes the number of classes of each style (e.g., positive or negative word in a sentiment classification task).

For the sentence-level style classification, we first take both the encoded representation $\mathbf{h}$ and stylistic word logits l_{word}. We then apply max pooling on the aggregation of $\mathbf{h} \oplus l_{word}$ along the sequence, resulting in vector $\mathbf{v} \in \mathbb{R}^{H+d_{l_{word}}}$ consisting of important logits. Finally, we input $\mathbf{v}$ into the sentence-level style classifier defined as follows:

$$l_{sentence} = \texttt{softmax}(\mathbf{W}_{sentence}\mathbf{v} + \mathbf{b}_{sentence})$$
$$P_{sentence} = \arg\max\,(l_{sentence})$$

where $l_{sentence} \in \mathbb{R}^2$ denotes sentence-level style logits, $\mathbf{W}_{sentence}$ is a matrix with the size $(H + d_{l_{word}}) \times 2$, and $P_{sentence}$ is the index of the predicted sentence-level style.

During training, StyLEx's objective is to maximize the probability of the sentence's style and stylistic word scores. The loss for both sentence-level style predictor and word-level style predictor is computed using binary cross entropy loss function. To jointly train the model, we optimize the following loss:

$$\mathcal{L} = \mathcal{L}_{style} + \alpha \times \mathcal{L}_{word}$$

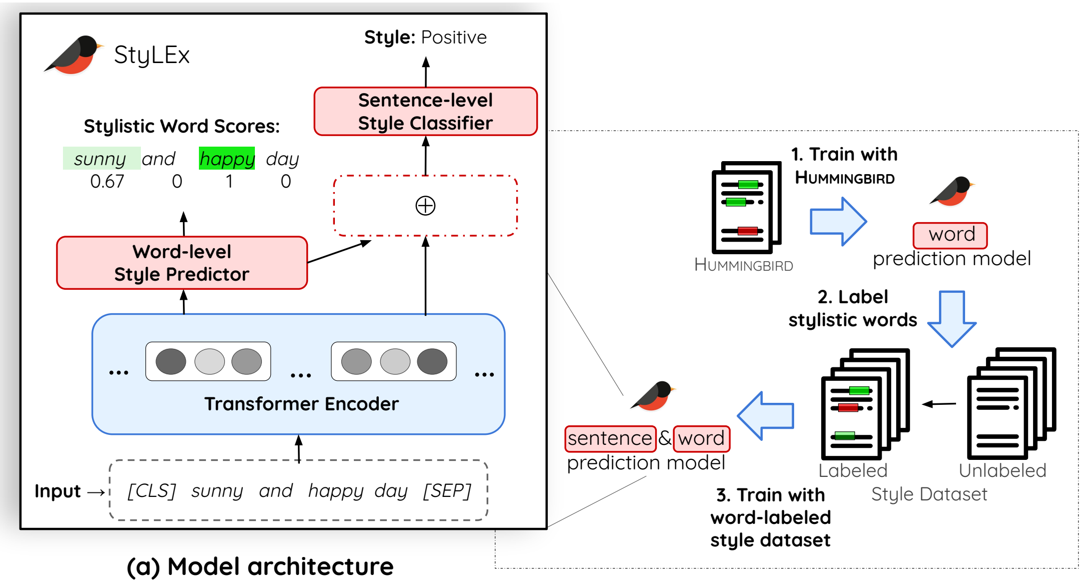
where α is a regularization hyperparameter.

2.2 StyLEx Model Training

To train StyLEx, we need a dataset of stylistic sentences along with their corresponding stylistic words (§2.2.1). We use the HUMMINGBIRD dataset (Hayati et al., 2021) that contains 500 sentences with word-level style annotation for obtaining the stylistic lexical explanation. Due to HUMMINGBIRD's small size, we first train StyLEx on HUMMINGBIRD and then predict pseudo stylistic words on larger benchmark style datasets (> 6.8k sentences in the training sets) for training the final StyLEx model for both sentence classification and lexical explanation.

2.2.1 Datasets

Following Hayati et al. (2021), we explore the same set of eight styles used in the dataset: politeness, sentiment, offensiveness and five emotions (anger, disgust, fear, joy, and sadness) for style classification tasks. We use three sets of publicly available style datasets for our experiments as follows.



(a) Model architecture

(b) Model training

Figure 2: **(a) StyLEx model architecture** (left). Our model has two new modules: a word-level style predictor and a sentence-level style classifier. An aggregator appends the word-level style logit for each word to the hidden layer representations of each word and takes the max pooling of this aggregation. **(b) Model training** (right). Human labels come from HUMMINGBIRD (Hayati et al., 2021) for stylistic word scores and from ORIGINAL datasets from sentence-level style classification. (1) We train a stylistic word-level prediction model on HUMMINGBIRD dataset in order to (2) obtain *pseudo-stylistic words* of sentences in the ORIGINAL datasets. (3) Then we train another stylistic word and sentence prediction model on this ORIGINAL sentences, now labeled with stylistic words.

HUMMINGBIRD is a multi-style dataset annotated with human perception scores on its important stylistic lexicons (Hayati et al., 2021). HUMMINGBIRD contains 500 sentences based on eight style datasets: politeness, sentiment, offensiveness, and five emotions (anger, disgust, fear, joy, and sadness). Three different crowd workers annotate each word in a sentence with 1 if they perceive the word as stylistic and 0 if not. The human perception score for a word is the average score of these annotators' labels. This perception score is what we call as stylistic word score and it is within the range [-1, 1]. We use HUMMINGBIRD for training StyLEx's word-level style predictor.

ORIGINAL datasets are used by Hayati et al. (2021) to curate HUMMINGBIRD.[2] Since some style labels in ORIGINAL may contain continuous numbers rather than binary labels, we follow the same setting of Hayati et al. (2021) which only uses binary labels: polite or impolite, positive or negative, offensive or not offensive, anger or not anger, and so on. The politeness dataset comes from StackExchange and Wikipedia requests (Danescu et al., 2013) (9.8k training instances). The sentiment dataset is a collection of movie review texts (Socher et al., 2013) (117k training instances). The offensiveness dataset is from Twitter (Davidson et al., 2017) (20k training instances). The emotions dataset (Mohammad et al., 2018) is collected from tweets (6.8k training instances). For all these ORIGINAL datasets, we use the default train/dev/split as explained in their papers.

Out-of-Domain (OOD) datasets are used to evaluate StyLEx's performance on different domains. For each style, we use data from different sources or topics, but their style labels are in HUMMINGBIRD and ORIGINAL datasets. For politeness, we use the polite and impolite sentences from the Enron email corpus (Klimt and Yang, 2004; Madaan et al., 2020). For sentiment, we test StyLEx on 5-core reviews from Amazon review dataset (Ni et al., 2019) for each product categories, except for movie reviews. We exclude movie reviews because it would be similar to the domain of the ORIGINAL's sentiment dataset. We convert ratings of 4-5 to positive labels and ratings of 1-2 as negative labels. For offensiveness, we use OffensEval (Zampieri et al., 2019) dataset for offensiveness. For five emotions, we collect Reddit comments

---

[2] We will refer to these individual datasets as "ORIGINAL"

Style	ORIGINAL (%)		OOD (%)	
	BERT	**StyLEx**	**BERT**	**StyLEx**
Politeness	67.96	65.84	71.45	74.18
Sentiment	96.52	96.59	86.70	86.99
Offensiveness	97.75	97.81	88.62	89.00
Anger	89.04	89.01	77.49	77.51
Disgust	86.50	86.90	74.06	74.63
Fear	95.66	95.63	78.42	78.48
Joy	88.02	88.12	75.20	74.26
Sadness	88.38	88.41	78.37	78.71

Table 1: For the sentence-level style classification task, StyLEx does not sacrifice the task performance (F1-scores) of the **BERT** model across all of the style tasks across both **ORIGINAL** and **OOD** settings.

from GoEmotions corpus (Demszky et al., 2020).[3]

2.2.2 Training

The whole pipeline of StyLEx model training is in Figure 2 (right). First, we train a stylistic word score prediction model with the same StyLEx architecture in Figure 2 (left). We do this since the sentences in the benchmarking style datasets do not have human annotations of stylistic word scores. We then use a semi-supervised learning approach called, pseudo-labeling (Lee et al., 2013; Rizve et al., 2020), to label the stylistic words. Now the sentences in ORIGINAL contain stylistic word scores which are output by the stylistic word predictor. Finally, we use both HUMMINGBIRD and ORIGINAL for training another model of StyLEx which predicts sentence-level binary style labels (polite and impolite, positive and negative etc.,) and provides lexical explanation scores within the range of [0, 1].[4]

3 Evaluation on Style Classification

3.1 Baseline

To assess StyLEx's classification performance, we compare it with a fine-tuned BERT-based classifier as a baseline. The training data for the baseline is also a combination of HUMMINGBIRD and ORIGINAL. For explanation evaluation, we compare StyLEx's explanation with the commonly-used explanation method called integrated gradients (Mudrakarta et al., 2018; Sundararajan et al., 2017), implemented in Captum[5]. Integrated gradient, which can be viewed as an approximate method of estimating Shapley values, is defined as follows. For the input sequence of words x and a neural network function F, an attribution score (the explanation) for each word is defined as the gradient between the input x and baseline x' of the function F where x' is a zero scalar.

3.2 Results

In our experiment, we have eight StyLEx models for each style: politeness, sentiment, offensiveness, and five emotions. For each style, we run StyLEx on the ORIGINAL test sets and OOD datasets for five times with different seeds and report the average of F1-scores in Table 1. For ORIGINAL datasets, StyLEx achieves higher F1 scores compared to the fine-tuned BERT model on sentiment, offensiveness, disgust, joy, and sadness. Overall, we observe that StyLEx does not sacrifice task performance of the state-of-the-art classifiers while predicting stylistic word scores. When tested on the OOD test sets, StyLEx achieves higher F1 score against the fine-tuned BERT model for all styles. Politeness has the greatest improvement from 71.48% to 74.18% since we observe that the ORIGINAL dataset of politeness contains many spurious content words. When we use bigger language models such as RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2019), and T5 (Raffel et al., 2020) for StyLEx, StyLEx still has better results than the baseline for five styles: sentiment, offensiveness, anger, and disgust.

From example sentences from the test sets in Table 2, we can see how StyLEx helps task performance. ORIG-1 and ORIG-2 sentences show how StyLEx can capture stylistic words and correct the sentence's style label to *disgust*. For example, "insult" and "injury" in ORIG-1 are initially labeled by the integrated gradient method as unimportant for identifying *disgust*, but StyLEx identifies the words as stylistic cues. Similarly, for the word "downfall" in ORIG-2, StyLEx finds it as offensive, but the baseline does not. StyLEx also has a higher stylistic word score for indicating "stogie" as an offensive word.

As we look at the politeness classification results, we find that most of the incorrect cases are when StyLEx mislabels subtle *impolite* sentences as *polite*. As we observe in Table 2 ORIG-3, StyLEx finds the word "please" as a polite cue, but the ground truth label of the sentence is *impolite*. We

---

[3] More details on the datasets are in Appendix A.1.

[4] Other implementation details are in Appendix A.2.

[5] `https://captum.ai`

ID	Model	Sentence with Predicted Stylistic Word Scores	Sentence Style
Incorrect Baseline Prediction → Correct StyLEx Prediction			
ORIG-1	StyLEx	... because i'm gonna ' add insult to injury	Disgust ✓
	Baseline	... because i'm gonna ' add insult to injury	Not Disgust ✗
ORIG-2	StyLEx	... yet you say i'm a stogie you're your own downfall	Offensive ✓
	Baseline	... yet you say i'm a stogie you're your own downfall	Not Offensive ✗
Correct Baseline Prediction → Incorrect StyLEx Prediction			
ORIG-3	StyLEx	please put them all back are you on dsl	Polite ✗
	Baseline	please put them all back are you on dsl	Impolite ✓
ORIG-4	StyLEx	... can't just be mean and do horrid things busy without paying the price	Not Fear ✗
	Baseline	... can't just be mean and do horrid things busy without paying the price	Fear ✓

Table 2: Error analysis on prediction-flipped sentences. Baseline refers to sentence-level style prediction result by fine-tuned BERT model and highlights are stylistic words found by the integrated gradient method. Green highlights on the words mean that the model predicts high positive word-level stylistic scores; red for the opposite label (e.g., *impolite* or *negative*). Sentence Style is a model's sentence-level style prediction ✓ marks correct prediction and ✗ denotes incorrect prediction.

then inspect its continuous score from the original politeness dataset by Danescu et al. (2013). It turns out that its politeness score is -0.38 in the range of [-2, 2] as -2 being the most impolite sentence. This shows that this sentence score is closer to neutral than impolite. This finding also reflects how HUMMINGBIRD dataset has been collected: as mentioned in Hayati et al. (2021), words from the offensive dataset (mostly swear words) are often labeled as impolite by human annotators. Thus, it may bias the annotators' view that sentences with impolite labels are not as bad as offensive sentences, making them not mark offensive sentences as *impolite*. Therefore, for such subtle impolite sentences, human annotators in HUMMINGBIRD may not label the sentence and words as impolite.

In contrast, StyLEx misclassifies *anger* sentences as *not anger* and *fear* as *not fear*. As we look at ORIG-4 in Table 2, StyLEx weakly finds *fear* cues ("horrid", "things") but they do not help in boosting the model to predict the sentence as *fear*. We conjecture that this is because there are very few training samples labeled with *fear* and *fear* has quite low word-level inter-annotator agreement as reported in Hayati et al. (2021).

4 Style Explanation Evaluation

We investigate StyLEx's explanations if they are sufficient, plausible, and understandable following previous works (DeYoung et al., 2020; Jacovi and Goldberg, 2020; Wiegreffe and Marasovic, 2021; Rajagopal et al., 2021). Jacovi and Goldberg (2020) define that a faithful interpretation represents a model's reasoning process. To evaluate whether StyLEx's explanations are faithful, we run a sufficiency test that evaluates whether the model explanations alone are highly relevant for predicting the label (Jacovi et al., 2018). Meanwhile, we measure plausibility to examine whether the explanation is agreeable to humans (DeYoung et al., 2020). Finally, understandability measures if a user is able to understand model explanations (Rajagopal et al., 2021). For investigating sufficiency and plausibility, we run automatic metrics. To assess understandability, we ask human judges to choose the explanation that can be better understood by a non-expert between StyLEx and the integrated gradients (IG) method.

4.1 Sufficiency

Following Jain et al. (2020); Rajagopal et al. (2021)'s sufficiency test, we fine-tune a BERT model on the top-k words as explanation instead of the whole sentence. We limit an explanation to contain 30% words of the average sentence length for each of the style datasets. These words are ranked based on their importance score by the baseline integrated gradient method and StyLEx for all the positive stylistic words (polite words, positive words, offensive words, angry words).

In Table 3, we can see that explanations from StyLEx show much higher predictive performance

Style	ORIGINAL		OOD	
	IG	StyLEx	IG	StyLEx
Politeness	43.92	**63.08**	52.89	**8.19**
Sentiment	87.18	**89.39**	64.93	**77.52**
Offensiveness	84.87	**91.26**	82.93	**84.75**
Anger	68.36	**86.90**	53.76	**73.99**
Disgust	82.54	**85.91**	71.36	**75.76**
Fear	87.82	**96.10**	35.85	**65.26**
Joy	45.54	**83.16**	55.49	**72.71**
Sadness	70.49	**87.94**	47.83	**70.95**

Table 3: The results for sufficiency test on the **ORIGINAL** and **OOD** data show the F1 scores on top-k words. IG stands for integrated gradient.

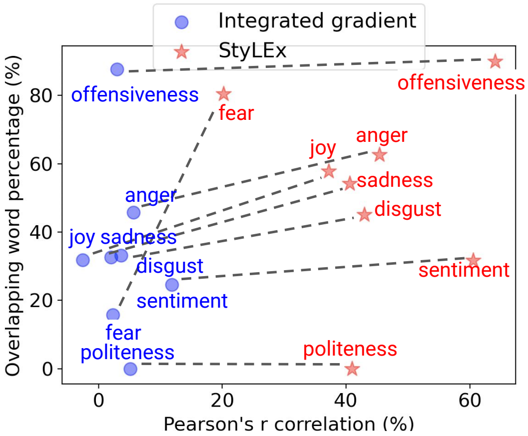


Figure 3: Plausibility experiment result. There are two points for each style in this plot. A blue circle point is for the baseline IG method and a red star point for StyLEx. X-axis is Pearson's r correlation score for each style. Y-axis is the percentage of stylistic sentences with style words appearing in the existing style lexicon dictionary.

compared to explanations extracted by the integrated gradient method for all styles in both datasets, ORIGINAL and OOD. This result suggests that human-like stylistic words are much more strongly predictive of a sentence's style compared to the gradient-based explanation methods that often rely on content words as an explanation. This indicates that StyLEx's explanations are relatively more faithful compared to the integrated gradients based method.

4.2 Plausibility

We use two approaches to measure the agreement between StyLEx's lexical explanations and stylistic words perceived by humans to assess the plausibility of StyLEx's explanations. In the first approach, we compare StyLEx's stylistic words on the HUMMINGBIRD test set and compare it with the ground truth human perception scores in HUMMINGBIRD. Second, we compare StyLEx's top-k stylistic words with existing expert-curated stylistic lexicon dictionaries. The domain of the existing expert-curated stylistic lexicon dictionaries could be different from the domain of the datasets in our study which range from social media texts to Wikipedia. However, it is still useful to compare how much StyLEx and the baseline agree human experts on identifying stylistic words.

Figure 3 shows a scatterplot of StyLEx vs. the integrated gradient. X-axis represents the Pearson's r correlation score on the HUMMINGBIRD test set. Y-axis is the percentage of overlapping words between the important words found by StyLEx and the baseline compared with the human-curated style lexicon dictionary. We calculate the overlapping word percentage as follows. We compute how many of the top 30% of the stylistic words in the ORIGINAL datasets found by StyLEx or baseline appear in human-curated dictionaries for the emotion/sentiment/offensive lexicons.

In Figure 3, the higher the Pearson's correlation score is (to the right), the better the explanation words produced by the model (StyLEx or baseline) are aligned with human perception ground truth from HUMMINGBIRD. The dashed lines show how much StyLEx's generated stylistic words align more with human annotations for stylistic words from both HUMMINGBIRD and human-curated stylistic lexicon dictionaries.

(1) Correlation with human perception. We investigate how similar StyLEx's explanations are with human perceptions. To do so, we compute the Pearson's correlation r between stylistic word scores predicted by StyLEx and annotated by humans from HUMMINGBIRD annotations for each word by concatenating all the predicted stylistic word scores. Our correlation score is different from Hayati et al. (2021) because we split the Hummingbird dataset into a training set and test set even though we use the same human perception scores and the integrated gradient method. The correlation score we reported in Figure 3 is from the HUMMINGBIRD test set.

In Figure 3 (vertical trend), we can see that StyLEx explanations correlate more with ground truth human perception for all styles, as red stars are stretched to the right. Sentiment and offen-

Style	🐦 🤖 **Both**	🐦 **StyLEx**	🤖 **Integrated gradient**
Positive	good, fun, love	associate, develop, instruct	deserve, endure, football
Negative	bad, horror, silly	mess, chaos, disappoint	maternal, banger, yell
Offensive	bitch, bitches, pussy	blind, racist, panties	fairy, amateur, fisting
Anger	angry, anger, awful	frowning, scare, lose	belt, campaigning, destroying*
Disgust	awful, terrible, angry	dismal, frowning, animosity	congress, finally, sentence*
Fear	fear, anxiety, nervous	horrid, war, threaten	rejects, mum, beating
Joy	happy, love, good	faith, sing, succeed	deal, independence, football
Sadness	depression, sadness, lost	bad, offended, leave	funeral, bloody, case*

Table 4: Three important words found by StyLEx (🐦) and the integrated gradient method (🤖) that appear in the stylistic lexicon dictionary.* = words only appear one time in the test data.

siveness are styles that have the highest correlation scores (60.53% and 64.09%) while fear is the lowest (20.17%)). Explanations from integrated gradient correlate very loosely with human perception ground truth with sentiment as the highest (11.89%) and joy negatively correlates with human perceptions (-2.55%).

(2) Comparison with stylistic lexicon dictionaries. We then investigate how similar the stylistic words found by StyLEx are to the stylistic words curated by humans in the existing lexicon dictionary. We use sentiment emotion lexicons from Mohammad and Turney (2010) and offensive lexicons from von Ahn's research group (2021).[6] Using the same set up of sufficiency test, we select top-30% stylistic words from each sentence in ORIGINAL datasets with the positive style label. Then we check if at least one of these words appear in the existing lexicon dictionary and compute its average across all training samples.

In Figure 3 (horizontal trend), we see that StyLEx consistently has higher percentage of word occurrences in the lexicon dictionary compared to the integrated gradient method where fear has the highest percentage difference (15.78% → 80.43%) and offensiveness has the lowest percentage change (87.67% → 89.99%). Averaging across all styles, we find that 56.70% of the stylistic sentences with StyLEx stylistic words appear in the existing style lexicon dictionary while integrated gradient only identifies 37.01% of those words.

The score is higher for offensiveness than for sentiment or emotion. We observe that people use more offensive words in social media, which is the source for dataset collection. We also examine the lower occurrence for emotions. From our analysis, we found that the emotion lexicon dictionary contains several colloquially rare words "aberration" or "meritorious", leading to a very low overlap with the datasets that we used for the analysis.

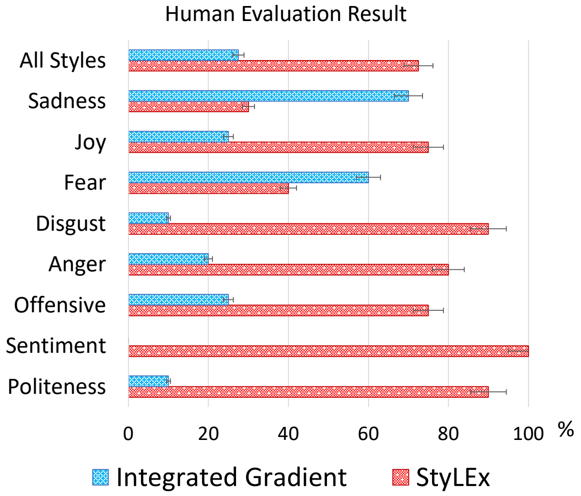


Figure 4: Human evaluation results. X-axis is the percentage of explanations preferred by human judges.

We also take a closer look at how many and the nature of important words are captured by StyLEx and/or the integrated gradient method as shown in Table 4. These word scores are obtained by averaging their scores and then we sort them based on these average scores. In general, we find that StyLEx can find more diverse stylistic words as defined in the existing lexicon dictionary for all styles except for positive sentiment. Some emotion words found by the integrated gradients only appear rarely in the data (mostly only once).

4.3 Understandability

To investigate the quality of StyLEx's explanation, we ask human judges to evaluate StyLEx's expla-

---

[6] We couldn't find a publicly available politeness lexicon dictionary.

nations compared to baseline explanations. *Understandability* asks whether human judges understand our explanation better than the explanation computed using integrated gradients. For this study, we randomly select 20 stylistic sentences for each of the eight styles, resulting in total 160 sentences. These 20 sentences are constructed by 10 sentences from ORIGINAL test set and 10 sentences from OOD test set. We normalize the stylistic word scores for sentence length across all sentences.

A human judge is shown two versions of the same sentence with different anonymized highlights, as shown in Figure 1. We then ask three different human judges to select (through Amazon Mechanical Turk) the explanation that was more understandable. Each worker annotated 20 sentences of the same style. The order of explanations is randomized to remove bias. We say that an explanation by a method is preferred by human judges when the majority choose that method. If the majority chooses a method for all the 20 sentences, the X-axis score will be 100%. Results in Figure 4 show that across all styles (160 pairs of explanations), StyLEx gives an overall gain of 27.5%.

5 Related Work

Styles in NLP Research on style in NLP has addressed various tasks including style classification (Danescu et al., 2013; Socher et al., 2013), style transfer (Rao and Tetreault, 2018; Li et al., 2018), style and content disentanglement (John et al., 2018; Zhu et al., 2021), and multiple style analysis (Hayati et al., 2021; Kang and Hovy, 2021). This work focuses on understanding stylistic variation in style classification. Style classification models often produce spurious features (Sen et al., 2021; Schlangen, 2021; Bras et al., 2020), motivating us to leverage stylistic variation from human perspectives to distinguish between stylistic words and content words. Past work have used stylistic lexica for classification (Taboada et al., 2011; Eisenstein, 2017), but in this work, we fine-tune the language model to generate these lexica and use them to help style prediction. Our work is most closely related to Hayati et al. (2021), but they do not develop any new models to use the human perception scores as explanations. Moreover, while linguistics styles can cover an author's writing style or figurative language, we limit our study to high-level style as used in Kang and Hovy (2021); Hayati et al. (2021).

Explainable NLP Heat maps generated from attention values from the models (Bahdanau et al., 2014) are widely used as an interpretability tool, but these attention maps are often unfaithful and unreliable (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Zhong et al., 2019; Pruthi et al., 2020). Saliency maps computed via gradients offer an alternative (Sundararajan et al., 2017; Smilkov et al., 2017; Mudrakarta et al., 2018). Annotating explanations as *rationales* (part of input) (Lei et al., 2016) through expert annotations (Zaidan and Eisner, 2008) is widely used to model explanations in NLP when external annotations are available. Another class of inherently interpretable models aims to optimize model explanations without any external annotations (Card et al., 2019; Croce et al., 2019; Rajagopal et al., 2021). Our work is similar in spirit to the rationale approaches (Lei et al., 2016) but focuses on understanding style attributes in text and computationally modeling them based on human annotation of the important words.

6 Conclusion

We proposed StyLEx, a style classification model for learning stylistic variations through lexical explanation. With only 500 sentences with word-level style annotation, we find improvement in both classification and explanation. Compared to the commonly-used integrated gradient method, StyLEx's explanations are more accurate for model prediction, more consistent with human-found stylistic words from existing datasets and lexicon dictionaries, and better understandable by human judges, without sacrificing task performance on both in-domain and out-of-domain datasets.

Future Work Our approach opens up future work on human-centered lexical explanation for correcting the spurious behavior of NLP models and for better explaining linguistic styles. We plan to investigate collecting more human lexical annotations to more accurately model stylistic variation, especially with larger pretrained language models. Broader usage of StyLEx in providing stylistic cues will be applicable to lexical style and content disentanglement (Cheng et al., 2020; John et al., 2019), counterfactual data augmentation for style-related tasks (Sen et al., 2021), and stylistic paraphrasing (Pavlick and Nenkova, 2015).

Limitations Our work has some limitations, mostly stemming from the size and nature of the human-annotated data. The training data (500 sentences) from HUMMINGBIRD is quite small to train deep learning models. However, our work shows that with just 500 sentences we could achieve a huge improvement in interpretability as well as a slight improvement in OOD performance, using semi-supervised training. Moreover, there is sparsity of stylistic words in the sentences. We also found that some stylistic words have various scores of human perception; capturing such subtle stylistic words is difficult. An interesting future work would be to handle these problems of sparsity and subtlety. We also notice that HUMMINGBIRD is annotated by people residing in the United States. Thus, their perception of styles may not reflect the perception of those with different cultural backgrounds. Nevertheless, StyLEx can be applied to any dataset training with similar human lexical annotations, and not limited to HUMMINGBIRD.

Ethical Considerations When collecting the explanation evaluation from human judges, we inform them that the content may contain offensive languages that could be upsetting.

Acknowledgments

We would like to thank Karin de Langis for her valuable feedback during the early version of writing and the anonymous reviewers for their thoughtful comments.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1078–1088. PMLR.

Dallas Card, Michael Zhang, and Noah A Smith. 2019. Deep weighted averaging classifiers. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 369–378.

Pengyu Cheng, Martin Renqiang Min, Dinghan Shen, Christopher Malon, Yizhe Zhang, Yitong Li, and Lawrence Carin. 2020. Improving disentangled text representation learning with information-theoretic guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7530–7541.

Danilo Croce, Daniele Rossini, and Roberto Basili. 2019. Auditing deep learning processes through kernel-based explanatory models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4028–4037.

Niculescu-Mizil Cristian Danescu, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 250–259, Sofia, Bulgaria. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. Goemotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054.

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

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Jacob Eisenstein. 2017. Unsupervised learning for lexicon-based classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Shirley Anugrah Hayati, Dongyeop Kang, and Lyle Ungar. 2021. Does BERT learn as humans perceive? understanding linguistic styles through lexica. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6323–6331, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Eduard Hovy. 1987. Generating natural language under pragmatic constraints. *Journal of Pragmatics*, 11(6):689–719.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. 2018. Understanding convolutional neural networks for text classification. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–65, Brussels, Belgium. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C Wallace. 2020. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4459–4473.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2018. Disentangled representation learning for non-parallel text style transfer. *arXiv preprint arXiv:1808.04339*.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434.

Jad Kabbara and Jackie Chi Kit Cheung. 2016. Stylistic transfer in natural language generation systems using recurrent neural networks. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*, pages 43–47.

Dongyeop Kang and Eduard Hovy. 2021. Style is NOT a single variable: Case studies for cross-stylistic language understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2376–2387, Online. Association for Computational Linguistics.

Bryan Klimt and Yiming Yang. 2004. Introducing the enron corpus. In *CEAS*.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Aman Madaan, Amrith Setlur, Tanmay Parekh, Barnabas Poczos, Graham Neubig, Yiming Yang, Ruslan Salakhutdinov, Alan W Black, and Shrimai Prabhumoye. 2020. Politeness transfer: A tag and generate approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1869–1881, Online. Association for Computational Linguistics.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. SemEval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Saif Mohammad and Peter Turney. 2010. Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 26–34, Los Angeles, CA. Association for Computational Linguistics.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia. Association for Computational Linguistics.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

Ellie Pavlick and Ani Nenkova. 2015. Inducing lexical style properties for paraphrase and genre differentiation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 218–224, Denver, Colorado. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C Lipton. 2020. Learning to deceive with attention-based explanations. In *ACL*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Dheeraj Rajagopal, Vidhisha Balachandran, Eduard Hovy, and Yulia Tsvetkov. 2021. Selfexplain: A self-explaining architecture for neural text classifiers. *arXiv preprint arXiv:2103.12279*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140.

Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. 2020. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *International Conference on Learning Representations*.

David Schlangen. 2021. Targeting the benchmark: On methodology in current natural language processing research. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 670–674, Online. Association for Computational Linguistics.

Indira Sen, Mattia Samory, Fabian Flöck, Claudia Wagner, and Isabelle Augenstein. 2021. How does counterfactually augmented data impact models for social computing constructs? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 325–344, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Luis von Ahn's research group (2021). Offensive/profane word list. Accessed: 2021-11-14.

Sarah Wiegreffe and Ana Marasovic. 2021. Teach me to explain: A review of datasets for explainable natural language processing. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Omar Zaidan and Jason Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 conference on Empirical methods in natural language processing*, pages 31–40.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.

Ruiqi Zhong, Steven Shao, and Kathleen McKeown. 2019. Fine-grained sentiment analysis with faithful attention. *arXiv preprint arXiv:1908.06870*.

Qingfu Zhu, Weinan Zhang, Ting Liu, and William Yang Wang. 2021. Neural stylistic response generation with disentangled latent variables. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4391–4401.

A Appendix

A.1 Sampling OOD Data and Data Statistics

- Politeness: We randomly sample 500 polite sentences and 500 impolite sentences from the Enron email corpus (Klimt and Yang, 2004; Madaan et al., 2020) since the size of entire corpus (>600k) is too large for inference.

- Sentiment: We test StyLEx on 5-core reviews from Amazon review dataset (Ni et al., 2019). For each category, we sample 100 positive sentences and 100 negative sentences from review categories, except for movie reviews which would be similar to the domain of the ORIGINAL dataset. We convert ratings of 4-5 to positive labels and ratings of 1-2 as negative labels.

- Offensiveness: We use OffensEval (Zampieri et al., 2019) dataset for offensiveness. We select all offensive tweets (3,002 instances) and all non-offensive tweets (2,991 instances) since OffensEval dataset is already nearly balanced.

- Emotions: For five emotions, we collect samples from GoEmotions corpus (Demszky et al., 2020) that contains Reddit comments labeled with 27 emotions, but we only select the five relevant emotions. For each emotion, we use all data for the positive emotion (e.g., joy) and undersample the negative emotion (e.g., not joy) data to equal the number of positive emotion samples.

The three dataset statistics are summarized in Table 6.

A.2 StyLEx Implementation Details

Throughout the experiment, we set $d_{l_{word}} = 2$ for politeness (polite, impolite) and sentiment (positive, negative) which have two style classes and $d_{l_{word}} = 1$ for the other styles. At the loss calculation step, we set the regularization hyperparameter α to 0.05 which gives the best style and perception prediction found searching the range [0.01, 100]. For the pseudo-labeling approach, we use the same architecture and hyperparameters with StyLEx model. We first train StyLEx with HUMMINGBIRD training set only to predict stylistic word scores for 50 epochs. Then we select the model with the best F1 score as a stylistic word score prediction to provide stylistic word scores for tokens in ORIGINAL training set. Then, we use both human-annotated perception score from HUMMINGBIRD and predicted stylistic word scores from ORIGINAL to train the sentence-level style prediction as in Figure 2. For the sentence-level model, we train the model for 5 epochs. For both stylistic word prediction and sentence-level style classification, we use BERT-base-uncased pretrained model. We set 0.1 dropout rate, 512 maximum sequence length, AdamW optimizer of learning rate $2e^{-5}$. For other hyper-parameters, we follow the default setting from HuggingFace's transformer library (Wolf et al., 2020).

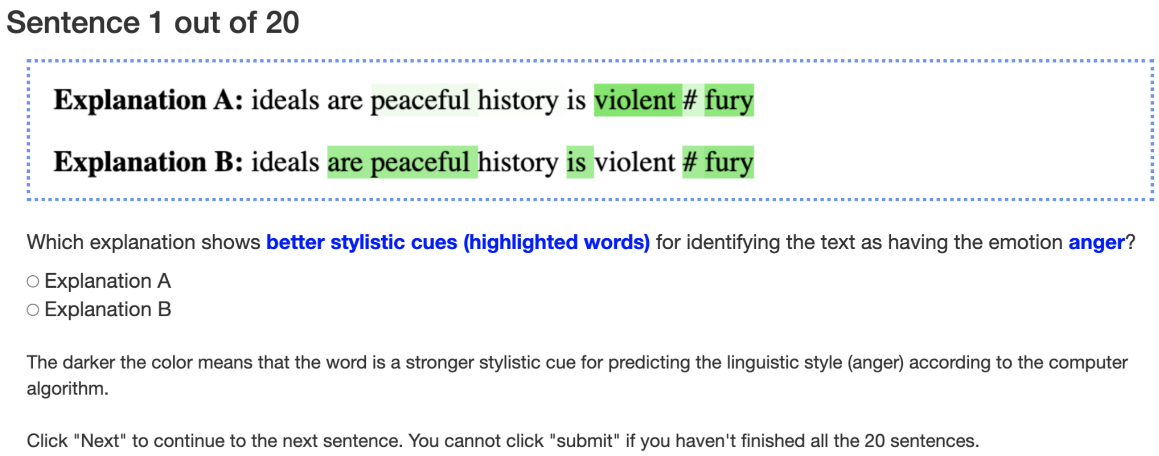
Our interface for human evaluation is shown as in Figure 5. Table 7 shows results for other pretrained language models. We report word-level style predictor performance tested on HUMMINGBIRD test data in the form of Pearson's r correlation scores as follows in Table 5.

Style	**Word-level Pearson's r**
Politeness	0.41
Sentiment	0.61
Offensiveness	0.37
Anger	0.45
Disgust	0.43
Fear	0.20
Joy	0.37
Sadness	0.41

Table 5: Pearson's r scores from training the word-level predicion model and testing it on HUMMINGBIRD.

	HUMMINGBIRD		ORIGINAL			OOD
Styles↓	Train	Test	Train	Dev	Test	Test
Politeness	256 (38%)	64 (28%)	9,855 (55%)	530 (56%)	567 (57%)	1,000 (50%)
Sentiment	312 (30%)	79 (37%)	117,219 (55%)	825 (51%)	1,749 (50%)	5,200 (50%)
Offensiveness	400 (34%)	100 (32%)	20,680 (82%)	1,173 (82%)	1,159 (81%)	5,993 (50%)
Anger	400 (35%)	100 (34%)	6,838 (37%)	886 (36%)	3,259 (34%)	16,168 (50%)
Disgust	400 (43%)	100 (38%)	6,838 (38%)	886 (36%)	3,259 (34%)	10,602 (50%)
Fear	400 (17%)	100 (13%)	6,838 (18%)	886 (14%)	3,259 (15%)	6,394 (50%)
Joy	400 (24%)	100 (19%)	6,838 (36%)	886 (45%)	3,259 (44%)	15,966 (50%)
Sadness	400 (29%)	100 (17%)	6,838 (29%)	886 (30%)	3,259 (29%)	13,516 (50%)

Table 6: Dataset statistics in our experiments. Note that these datasets are preprocessed from existing datasets. For HUMMINGBIRD (Hayati et al., 2021) and ORIGINAL datasets, the train, dev, and test sets have the same size for all emotions. We do not report the training size of Out-of-Domain (OOD) datasets since we are not using them for training. The label distributions for positive labels are in the parentheses.



Sentence 1 out of 20

Explanation A: ideals are peaceful history is violent # fury

Explanation B: ideals are peaceful history is violent # fury

Which explanation shows **better stylistic cues (highlighted words)** for identifying the text as having the emotion **anger**?

○ Explanation A
○ Explanation B

The darker the color means that the word is a stronger stylistic cue for predicting the linguistic style (anger) according to the computer algorithm.

Click "Next" to continue to the next sentence. You cannot click "submit" if you haven't finished all the 20 sentences.

Figure 5: Interface for human evaluation

Model		**F1 Score**							
		Polite.	**Sent.**	**Offens.**	**Anger**	**Disgust**	**Fear**	**Joy**	**Sad.**
ORIG									
BERT	Baseline	67.96	96.52	97.75	89.04	86.50	95.66	88.02	88.38
	StyLEx	65.84	96.59	97.81	89.01	86.90	95.63	88.14	88.41
RoBERTa	Baseline	65.83	96.94	96.40	89.56	87.17	95.68	88.32	88.52
	StyLEx	66.05	96.59	96.55	89.39	87.16	95.67	88.39	88.89
XLNet	Baseline	64.09	96.57	96.86	88.46	86.07	95.55	86.95	87.32
	StyLEx	63.69	96.54	96.38	88.20	86.32	95.44	87.33	87.90
T5	Baseline	65.75	97.13	97.21	88.79	86.29	95.39	88.18	87.68
	StyLEx	67.69	97.21	96.61	88.69	86.24	95.28	87.86	87.51
OOD									
BERT	Baseline	71.45	86.70	88.62	77.49	74.06	78.42	75.20	78.37
	StyLEx	74.18	86.99	88.98	77.51	74.63	78.48	74.26	78.71
RoBERTa	Baseline	72.08	90.41	87.48	77.01	74.56	79.95	74.63	80.24
	StyLEx	69.27	89.90	89.63	77.86	75.07	78.66	74.18	79.09
XLNet	Baseline	68.84	88.25	88.33	76.24	74.77	78.48	74.03	78.78
	StyLEx	67.28	89.65	88.56	76.41	74.71	78.92	74.09	78.28
T5	Baseline	70.76	91.72	88.14	75.74	73.83	79.58	73.76	78.37
	StyLEx	68.14	91.73	88.05	75.48	73.33	80.23	73.70	77.61

Table 7: More classification results with several language models.

Comparing Intrinsic Gender Bias Evaluation Measures without using Human Annotated Examples

Masahiro Kaneko[1] **Danushka Bollegala**[2,3*] **Naoaki Okazaki**[1]
[1]Tokyo Institute of Technology [2]University of Liverpool [3]Amazon
masahiro.kaneko@nlp.c.titech.ac.jp
danushka@liverpool.ac.uk okazaki@c.titech.ac.jp

Abstract

Numerous types of social biases have been identified in pre-trained language models (PLMs), and various intrinsic bias evaluation measures have been proposed for quantifying those social biases. Prior works have relied on human annotated examples to compare existing intrinsic bias evaluation measures. However, this approach is not easily adaptable to different languages nor amenable to large scale evaluations due to the costs and difficulties when recruiting human annotators. To overcome this limitation, we propose a method to compare intrinsic gender bias evaluation measures without relying on human-annotated examples. Specifically, we create multiple *bias-controlled* versions of PLMs using varying amounts of male vs. female gendered sentences, mined automatically from an unannotated corpus using gender-related word lists. Next, each bias-controlled PLM is evaluated using an intrinsic bias evaluation measure, and the rank correlation between the computed bias scores and the gender proportions used to fine-tune the PLMs is computed. Experiments on multiple corpora and PLMs repeatedly show that the correlations reported by our proposed method that does not require human annotated examples are comparable to those computed using human annotated examples in prior work.

1 Introduction

Pre-trained language models (PLMs) trained on large datasets have reported impressive performance improvements in various NLP tasks (Devlin et al., 2019; Lan et al., 2019) greatly. However, these PLMs also demonstrate significantly worrying levels of social biases (Bolukbasi et al., 2016; Kurita et al., 2019). To address this issue, numerous intrinsic bias evaluation measures for PLMs have been proposed (Nangia et al., 2020; Dhamala et al., 2021; Nadeem et al., 2021; Kaneko and Bollegala, 2022; Zhou et al., 2022), which are also used for comparing debiasing methods for PLMs (Webster et al., 2020; Kaneko and Bollegala, 2021a; Schick et al., 2021).

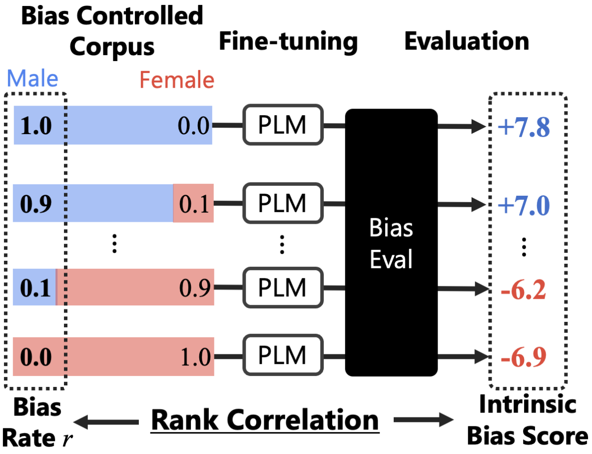


Figure 1: Overview of our proposed method. We first create *bias-controlled* PLMs by fine-tuning a PLM on male and female gendered sentences that are automatically mined from unannotated corpora. Next, we measure the rank correlation between the scores reported by an intrinsic bias evaluation measure and the male/female bias rates (r) used to fine-tune the PLMs.

Existing bias evaluation methods use different criteria such as pseudo likelihood (Kaneko and Bollegala, 2022), cosine similarity (Caliskan et al., 2017; May et al., 2019), inner-product (Ethayarajh et al., 2019) etc. Moreover, current bias evaluation methods require manually-annotated datasets containing stereotypical and antistereotypical examples that express different types of social biases (Nangia et al., 2020; Nadeem et al., 2021). Therefore, we consider that it is important to compare the differences in existing bias evaluation measures proposed for PLMs (Orgad and Belinkov, 2022; Dev et al., 2021; Kaneko et al., 2022a) to understand their relative strengths and weaknesses.

---

[*] Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2857–2863
May 2-6, 2023

To objectively compare the existing bias evaluation measures, Kaneko and Bollegala (2022) calculated the rank correlation between the number of human annotators who labelled an example to be stereotypically biased towards a protected attribute in Crowds-Pairs (CP), and the bias score for that example returned by an intrinsic bias evaluation measure (Nangia et al., 2020; Nadeem et al., 2021). However, due to the costs and difficulties in recruiting human annotators, this approach cannot be easily adapted to different languages, accommodate large-scale evaluations, or compare evaluation metrics that do not use human-annotated data.

We propose a method to compare intrinsic bias evaluation measures without using human annotated examples. Figure 1 outlines the intuition behind our proposed method. First, we train *bias-controlled* versions of PLMs obtained via fine-tuning a PLM on male and female gendered sentences, automatically mined from an unannotated corpus using a gender-related word list. We define *rate of bias* (r) as the ratio between male and female gendered sentences in a training sample used to fine-tune a PLM. A PLM fine-tuned mostly on male sentences is likely to generate sentences containing mostly male words, while a PLM fine-tuned on female sentences is likely to generate sentences containing mostly female words (Kaneko and Bollegala, 2022; Kaneko et al., 2022c). Therefore, an accurate intrinsic bias evaluation measure is expected to return a score indicating a bias towards the male gender for a male bias-controlled PLM, while it is expected to return a score indicating a bias towards the female gender for a female bias-controlled PLM. We then compute the rank correlation between (a) the rate of biases in the bias-controlled PLMs, and (b) the bias scores returned by an intrinsic evaluation measure for the corresponding PLMs, as a measure of accuracy of the bias evaluation measure.

Our experiments with multiple corpora and PLMs show that the correlations reported by our proposed method, which does not require human annotated examples, are comparable to those computed using human annotated examples in previous studies. Furthermore, by examining the output probabilities of the PLM, we verify that the proposed method, which fine-tunes bias-controlled PLMs with varying amounts of male vs. female sentences, is indeed able to control biases associated with male and female gender directions.

2 Bias-controlled Fine-Tuning

The imbalance of gender words in the training data affects the gender bias of a PLM fine-tuned using that data (Kaneko and Bollegala, 2022; Kaneko et al., 2022c). Using this fact, we propose a method to learn *bias-controlled* versions of PLMs that express different levels of known gender biases. Let us first assume that we are given a list of female gender related words $\mathcal{V}_f$ (e.g. *she, woman, female*), and a separate list of male gender related words $\mathcal{V}_m$ (e.g. *he, man, male*). Next, we select sentences that contain either at least one of female or male words from an unannotated set of sentences $\mathcal{D}$. Sentences that contain both male and female words are excluded here. Let us denote the set of sentences extracted for a female or a male word w by $\Phi(w)$. Moreover, let $\mathcal{D}_f = \bigcup_{w \in \mathcal{V}_f} \Phi(w)$ and $\mathcal{D}_m = \bigcup_{w \in \mathcal{V}_m} \Phi(w)$ be the sets of sentences containing respectively female and male words. We appropriately downsample $\mathcal{D}_f$ and $\mathcal{D}_m$ to have equal numbers of sentences N (i.e. $|\mathcal{D}_f| = |\mathcal{D}_m| = N$).

Next, we create training datasets $\mathcal{D}_r$ by varying the rate of bias, r ($\in [0, 1]$), by randomly sampling a subset $\mathcal{S}_r(\mathcal{D}_m)$ of Nr sentences from $\mathcal{D}_f$ and a subset $\mathcal{S}_{1-r}(\mathcal{D}_f)$ of $N(1-r)$ sentences from $\mathcal{D}_m$ such that $\mathcal{D}_r = \mathcal{S}_r(\mathcal{D}_m) \cup \mathcal{S}_{1-r}(\mathcal{D}_f)$. When $r = 0$, $\mathcal{D}_r$ consists of only female sentences (i.e. $\mathcal{D}_r \subseteq \mathcal{D}_f$), and when $r = 1$, it consists of only male sentences (i.e. $\mathcal{D}_r \subseteq \mathcal{D}_m$). To obtain multiple bias-controlled PLMs at different levels of gender biases, we fine-tune a given PLM on different datasets, D_r, sampled with different values of r. We use a given intrinsic bias evaluation measure to separately evaluate each bias-controlled PLM. Finally, we measure the agreement between the bias scores reported by the intrinsic bias evaluation measure under consideration and the corresponding rates of biases of those PLMs using Pearson's rank correlation coefficient.

3 Experiments

3.1 Settings

In our experiments, we used the female words *she, woman, female, her, wife, mother, girl, sister, daughter, girlfriend* as $\mathcal{V}_f$, and male words *he, man, male, him, his, husband, father, boy, brother, son, boyfriend* as $\mathcal{V}_m$. We sampled 2M sentences each representing male and female genders from News crawl 2021 corpus (**news**)[1] and BookCorpus (Zhu

---

[1] https://data.statmt.org/news-crawl/en/

Measure	BERT			ALBERT		
	news	book	HA	news	book	HA
TBS	0.14	0.09	-	0.25	0.14	-
SSS	0.22	0.22	0.45	0.31	0.22	0.53
CPS	0.30	0.27	0.57	0.37	0.22	0.48
AUL	0.37	0.32	0.68	0.55	0.36	0.56
AULA	0.42	0.34	0.71	0.60	0.42	0.57

Table 1: Peason correlation between biased PLM order and each bias scores. News and book represent the corpus used for biasing, respectively. HA is AUC value of method using human annotation (Kaneko and Bollegala, 2021a).

et al., 2015) (**books**) for training bias-controlled PLMs and a separate 100K sentences as development data. We used BERT[2] (Devlin et al., 2019) and ALBERT[3] (Lan et al., 2019) as the PLMs. We fine-tune PLMs with masked language model learning. We use publicly available Transformer library[4] to fine-tuning PLMs, and all hyperparameters are set to their default values. We trained 11 bias-controlled PLMs for r in $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ on four Tesla V100 GPUs.

3.2 Intrinsic Bias Evaluation Measures

We compare five previously proposed intrinsic gender bias evaluation measures in this paper: Template-Based Score (**TBS**; Kurita et al., 2019), StereoSet Score (**SSS**; Nadeem et al., 2021), CrowS-Pairs Score (**CPS**; Nangia et al., 2020), All Unmasked Likelihood (**AUL**; Kaneko and Bollegala, 2022), and AUL with Attention weights (**AULA**; Kaneko and Bollegala, 2022). Further details of these measures are given in the Appendix.

Note that TBS uses templates for evaluation and cannot be used with human-annotated stereotypical/anti-stereotypical sentences. On the other hand, SSS, CPS, AUL, and AULA all require human-annotated sentences that express social biases.

3.3 Comparing Intrinsic Gender Bias Evaluation Measures

We compare the proposed method and Kaneko and Bollegala (2022)'s method using CP dataset, which has human annotations, and show the effectiveness of the proposed method. In addition, we will use several PLMs and corpora to analyze the trends of the proposed method. Table 1 shows the correlation results of the proposed method for TBS, SSS, CPS, AUL, and AULA when fine-tuning BERT and ALBERT on news or book corpora, respectively. HA is the AUC value of the Kaneko and Bollegala (2022)'s method using human annotations. Since TBS uses templates, it cannot be evaluated using HA.

For BERT, the proposed method induces the same order among measures (i.e. AULA $>$ AUL $>$ CPS $>$ SSS) as done by HA in both news and book. For ALBERT, only the rankings of SSS and CPS differ between the proposed method and HA. These results show that the proposed method and the existing method that use human annotations rank the intrinsic gender bias evaluation measures in almost the same order.[5] It can be seen that the values of the correlation coefficients vary depending on the PLM and corpus. For example, ALBERT has a maximum correlation of 0.60, while BERT has a maximum correlation of only 0.42.

A major limitation of human annotation-based evaluation is that it cannot be used to compare TBS that does not human annotated examples against other intrinsic bias evaluation measures. However, our proposed method does *not* have this limitation and can be used to compare TBS against other bias evaluation measures. As it can be seen from Table 1, TBS consistently reports the lowest correlations, indicating that it is not an accurate intrinsic gender bias evaluation measure. This finding agrees with Kaneko et al. (2022a), who highlighted the inadequacy of templates as a method for evaluating social biases.

3.4 Bias-controlled PLMs

To verify that the proposed method can indeed control the bias of a PLM, we investigate the variation of the output probabilities of the PLMs fine-tuned with different r. Specifically, we investigate the output probabilities of masked *he* and *she* in the input text "*[MASK] is a/an [Occupation].*" for the bias-controlled PLMs. For *[Occupation]*, we use gender- and stereotype-neutral occupational words[6] (e.g. *writer*, *musician*) from the word list created by Bolukbasi et al. (2016). When r in-

---

[2] `https://huggingface.co/bert-base-uncased`

[3] `https://huggingface.co/albert-base-v2`

[4] `https://github.com/huggingface/transformers/tree/v4.22.2`

[5] Because of the different methods of measuring correlations, it is not possible to compare the magnitude of values between the proposed and existing methods.

[6] `https://github.com/tolga-b/debiaswe`

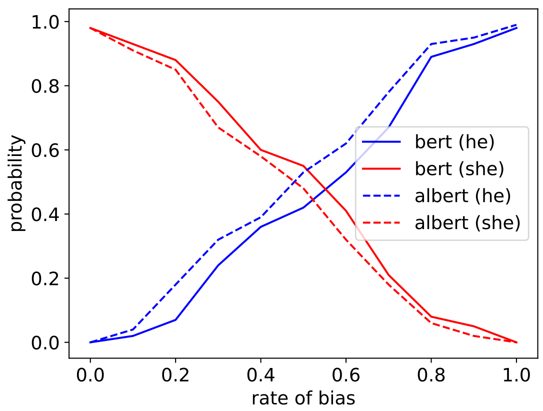
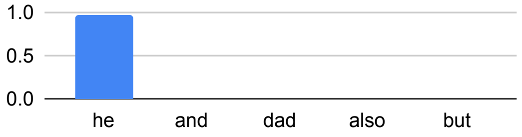
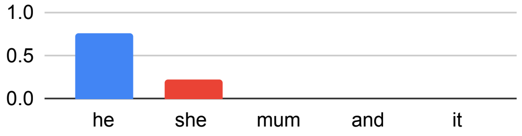


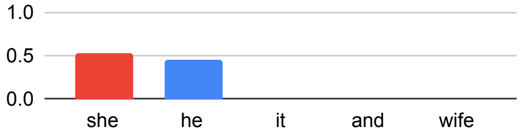
Figure 2: Average output probabilities for "*[MASK] is a/an [Occupation]*" produced by the bias-controlled BERT and ALBERT PLMs fine-tuned with different r on the news dataset.



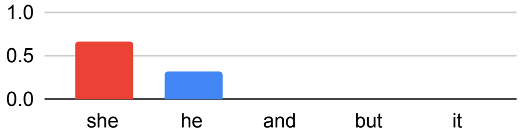
(a) $r = 1.0$



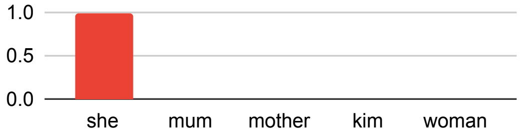
(b) $r = 0.7$



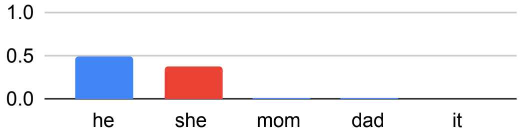
(c) $r = 0.5$



(d) $r = 0.3$



(e) $r = 0.0$



(f) original (without fine-tuning)

Figure 3: Top 5 words with BERT output probability for "*[MASK] doesn't have time for family due to work obligations.*". Blue and red represent masculine and feminine words, respectively.

creases, a PLM will be fine-tuned with increasing amounts of male sentences. Therefore, if the average probability of *he* increases with r, it would imply that the PLMs are correctly bias-controlled by the proposed method.

Figure 2 shows that the average output probabilities of *he* and *she* when r is incremented in step size of 0.1. When $r = 1$ the PLM predicts *he* with fairly high probability and when $r = 0$ the PLM predicts *she* with fairly high probability. Furthermore, when $r = 0.5$, the probability of *he* and *she* is almost 0.5. Original BERT (without fine-tuning) returns 0.48 and 0.28, respectively for *he* and *she*, while the corresponding probabilities returned by ALBERT are respectively 0.64 and 0.22. Both the original BERT and ALBERT predict relatively larger output probabilities for *he*, indicating that they are male-biased, without performing any bias-control. From these results, it can be seen that the output probabilities of *he* and *she* fluctuate according to r, and the proposed method can control the bias of the PLM. On the other hand, when r is less than 0.2 or greater than 0.8, the output probabilities of *she* and *he* are greater than the proportion in the data set, respectively. Therefore, finer increments of r may make it difficult to control bias more finely when r is small or large.

To illustrate how bias-controlled PLMs produced by the proposed method for different rates of biases (r) predict the probabilities of gender pronouns, we consider the masked sentence "*[MASK] doesn't have time for the family due to work obligations.*" selected from the CP dataset. Here, *He* and *She* are unmodified tokens. Figure 3 shows the probabilities of the tokens predicted for the [MASK] by the different bias-controlled PLMs. We see that the original BERT model predicts both *he* and *she* with approximately equal probabilities. However, when r is gradually increased from 0 to 1, we see that the probability of *he* increases, while that of *she* decreases, demonstrating that the proposed method correctly learns bias-controlled PLMs.

4 Conclusion

We proposed a method to compare intrinsic gender bias evaluation measures using an unannotated corpus and gender-related word lists. Experiments show that the correlations computed by the proposed method for existing bias evaluation measures agrees with the prior evaluations conducted using human annotations.

5 Limitations

In this paper, we limited our investigation to English PLMs. However, as reported in a lot of previous work, social biases are language independent and omnipresent in PLMs trained for many languages (Kaneko et al., 2022c; Lewis and Lupyan, 2020; Liang et al., 2020; Zhao et al., 2020). We plan to extend this study to cover non-English PLMs in the future.

According to existing research, PLMs encode many different types of social biases such as racial and religious biases in addition to gender-related biases (Kiritchenko and Mohammad, 2018; Ravfogel et al., 2020). On the other hand, in this paper, we focused on only gender bias. Extending the proposed method to handle other types of social biases beyond gender bias is beyond the scope of the current short paper and is deferred to future work.

Furthermore, discriminatory bias is learned in word embeddings as well as PLMs (Bolukbasi et al., 2016; Brunet et al., 2019; Kaneko and Bollegala, 2019, 2020, 2021b; Kaneko et al., 2022b). Therefore, it may be possible to make it applicable to word embeddings as well.

6 Ethical Considerations

Our goal in this paper was to compare the previously proposed and widely-used intrinsic bias evaluation measures of gender bias in pre-trained PLMs. Although we used a broad range of existing datasets that are annotated for social biases, we did not annotate nor release new datasets as part of this research. Moreover, we fine-tune a large number of bias-controlled PLMs for evaluation purposes that demonstrates varying levels of gender biases. However, these PLMs are not supposed to be used in downstream tasks other than for evaluation purposes.

Even with the highly correlated bias evaluation measure in our proposed method, the bias of the PLM may not be sufficiently evaluated. Therefore, we consider that it important to select intrinsic gender bias evaluation measures carefully and not purely based on correlation coefficients computed by the proposed method alone.

There are various discussions on how to define social bias in PLMs (Blodgett et al., 2021). Since the proposed method can use any method as the bias-controlled fine-tuning of the PLMs, the bias-controlled fine-tuning can be selected according to the definition of social bias.

Acknowledgements

This paper is based on results obtained from a project, JPNP18002, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna Wallach. 2021. Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1004–1015, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Marc-Etienne Brunet, Colleen Alkalay-Houlihan, Ashton Anderson, and Richard Zemel. 2019. Understanding the origins of bias in word embeddings. In *International conference on machine learning*, pages 803–811. PMLR.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

Sunipa Dev, Emily Sheng, Jieyu Zhao, Aubrie Amstutz, Jiao Sun, Yu Hou, Mattie Sanseverino, Jiin Kim, Akihiro Nishi, Nanyun Peng, and Kai-Wei Chang. 2021. On measures of biases and harms in nlp.

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

Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. Bold: Dataset and metrics for measuring biases in open-ended language generation. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 862–872, New York, NY, USA. Association for Computing Machinery.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Understanding undesirable word embedding associations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 1696–1705, Florence, Italy. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2019. Gender-preserving debiasing for pre-trained word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1641–1650, Florence, Italy. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2020. Autoencoding improves pre-trained word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1699–1713, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2021a. Debiasing pre-trained contextualised embeddings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1256–1266, Online. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2021b. Dictionary-based debiasing of pre-trained word embeddings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 212–223, Online. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2022. Unmasking the mask–evaluating social biases in masked language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11954–11962.

Masahiro Kaneko, Danushka Bollegala, and Naoaki Okazaki. 2022a. Debiasing isn't enough! – on the effectiveness of debiasing MLMs and their social biases in downstream tasks. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1299–1310, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Masahiro Kaneko, Danushka Bollegala, and Naoaki Okazaki. 2022b. Gender bias in meta-embeddings. *arXiv preprint arXiv:2205.09867*.

Masahiro Kaneko, Aizhan Imankulova, Danushka Bollegala, and Naoaki Okazaki. 2022c. Gender bias in masked language models for multiple languages. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2740–2750, Seattle, United States. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Molly Lewis and Gary Lupyan. 2020. Gender stereotypes are reflected in the distributional structure of 25 languages. *Nature human behaviour*, 4(10):1021–1028.

Sheng Liang, Philipp Dufter, and Hinrich Schütze. 2020. Monolingual and multilingual reduction of gender bias in contextualized representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5082–5093, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Hadas Orgad and Yonatan Belinkov. 2022. Choose your lenses: Flaws in gender bias evaluation. In *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 151–167, Seattle, Washington. Association for Computational Linguistics.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256, Online. Association for Computational Linguistics.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in NLP. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *arXiv preprint arXiv:2010.06032*.

Jieyu Zhao, Subhabrata Mukherjee, Saghar Hosseini, Kai-Wei Chang, and Ahmed Hassan Awadallah. 2020. Gender bias in multilingual embeddings and cross-lingual transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2896–2907, Online. Association for Computational Linguistics.

Yi Zhou, Masahiro Kaneko, and Danushka Bollegala. 2022. Sense embeddings are also biased – evaluating social biases in static and contextualised sense embeddings. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1924–1935, Dublin, Ireland. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Intrinsic Bias Evaluation Measures

TBS Kurita et al. (2019) proposed template-based bias evaluation measure. The log-odds of the likelihood of a template sentence masked with a gender word (e.g. *"[MASK] is a programmer"*) and the likelihood of a gender word masked with an occupation word (e.g. *"[MASK] is a [MASK]"*) are calculated for male and female words, respectively. TBA then calculates the difference between them as the bias score.

SSS SSS (Nadeem et al., 2021) uses stereotypical and anti-stereotypical sentence pairs (e.g. *"She is a nurse"* and *"He is a nurse"*) to evaluate bias in PLMs. Calculate the likelihood of masked modified tokens (e.g. *She, He*) given unmasked unmodified tokens (e.g. *is, a, nurse*) for each stereotypical and anti-stereotypical sentence. The bias score is calculated by dividing the number of sentences for which the total likelihood is higher for stereotypical sentences compared to anti-stereotypical sentences by the total number of data.

CPS CPS (Nangia et al., 2020) also uses stereotypical and anti-stereotypical sentence pairs. On the other hand, calculate the likelihood of masked unmodified tokens given unmasked modified tokens for each stereotypical and anti-stereotypical sentence. The bias score is calculated by dividing the number of sentences for which the total likelihood is higher for stereotypical sentences compared to anti-stereotypical sentences by the total number of data. As with SSS, the bias score is calculated using the sum of the likelihoods of the stereotyped and anti-stereotyped sentences.

AUL and AULA AUL and AULA (Kaneko and Bollegala, 2022) also uses stereotypical and anti-stereotypical sentence pairs, but they calculate the likelihood of unmasked unmodified tokens and modified tokens for each stereotypical and anti-stereotypical sentence. As with SSS and CPS, the bias score is calculated using the sum of the likelihoods of the stereotyped and anti-stereotyped sentences. AULA calculates the likelihood of the entire sentence by weighting and averaging with the attention weights to prioritize the likelihood of important words.

Faithfulness-Aware Decoding Strategies for Abstractive Summarization

David Wan[1*] **Mengwen Liu**[2] **Kathleen McKeown**[3] **Markus Dreyer**[2†] **Mohit Bansal**[1,2†]
[1]UNC Chapel Hill [2]Amazon Alexa AI [3]AWS AI Labs
`{davidwan,mbansal}@cs.unc.edu`
`{mengwliu,mckeownk,mddreyer,mobansal}@amazon.com`

Abstract

Despite significant progress in understanding and improving faithfulness in abstractive summarization, the question of how decoding strategies affect faithfulness is less studied. We present a systematic study of the effect of generation techniques such as beam search and nucleus sampling on faithfulness in abstractive summarization. We find a consistent trend where beam search with large beam sizes produces the most faithful summaries while nucleus sampling generates the least faithful ones. We propose two faithfulness-aware generation methods to further improve faithfulness over current generation techniques: (1) ranking candidates generated by beam search using automatic faithfulness metrics and (2) incorporating lookahead heuristics that produce a faithfulness score on the future summary. We show that both generation methods significantly improve faithfulness across two datasets as evaluated by four automatic faithfulness metrics and human evaluation. To reduce computational cost, we demonstrate a simple distillation approach that allows the model to generate faithful summaries with just greedy decoding.[1]

1 Introduction

Recent developments in large pre-trained language models have achieved remarkable performance on abstractive summarization (Lewis et al., 2020; Zhang et al., 2020a). However, such models often suffer from the problem of hallucinations, where the generated summary contains facts or entities not present in the original document. Prior research has analyzed and defined potential error types and typology (Maynez et al., 2020; Pagnoni et al., 2021; van der Poel et al., 2022), and developed methods to improve faithfulness, including post-processing models (Chen et al., 2021b; Dong et al., 2020; Liu and Liu, 2021; Ladhak et al., 2022) and faithfulness-aware training (Goyal and Durrett, 2021; Nan et al., 2021; Cao and Wang, 2021; Wan and Bansal, 2022; Zhang et al., 2022; Xiao and Carenini, 2022).

One aspect that is less understood on faithfulness of abstractive summarization is the effect of decoding strategies, which determine how the model generates the output strings. Our primary objective is to understand whether different types of exploration of the search space, such as traversing and maintaining multiple possible output hypotheses with beam search or encouraging diversity with nucleus sampling (Holtzman et al., 2020), have an impact on faithfulness. To this end, we first conduct a thorough analysis comparing the faithfulness of popular decoding strategies, including greedy decoding, beam search, and nucleus sampling for two popular summarization datasets XSum (Narayan et al., 2018) and CNN/DM (Hermann et al., 2015). Evaluating the generated summaries using four faithfulness metrics, including BertScore (Zhang et al., 2020b), FactCC (Kryscinski et al., 2020), DAE (Goyal and Durrett, 2021), and QuestEval (Scialom et al., 2021), and human evaluation, we find a consistent trend that beam search provides the most faithful summaries with its large exploration of the search space, and the randomness introduced by sampling hurts faithfulness.

To further improve faithfulness beyond the common decoding strategies, we propose two faithfulness-aware decoding methods. First, similar to Falke et al. (2019), we make use of the multiple candidates generated by beam search and propose a simple re-ranker, which selects the best summary according to a faithfulness metric. Instead of using a specific metric, we rank and select the summaries with a composite metric, a weighted combination of popular faithfulness metrics. Next, inspired by Lu et al. (2022), we propose a faithful-

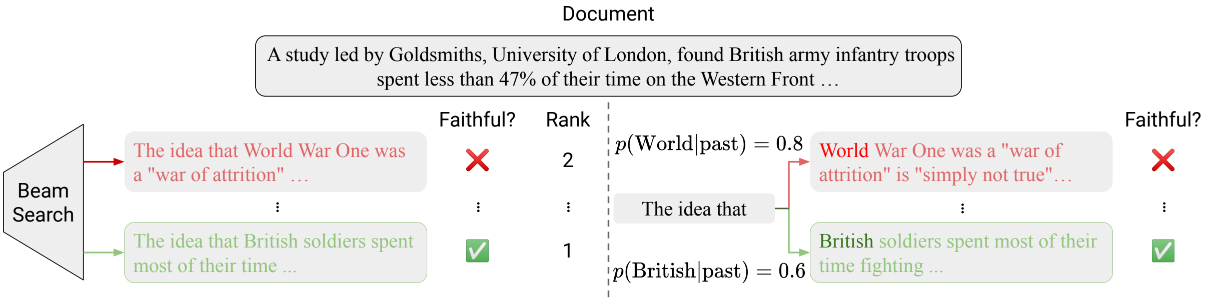
---

[*] Work conducted during an internship at Amazon.

[†] Corresponding authors.

[1] Our code is publicly available at https://github.com/amazon-science/faithful-summarization-generation.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2864–2880
May 2-6, 2023 ©2023 Association for Computational Linguistics



(a) Ranker (b) Lookahead

Figure 1: Illustration of our proposed decoding methods. 1a shows our ranker that re-ranks the candidates produced by beam search according to faithfulness metrics. The first summary achieves a high score and would be used as the final summary for beam search, but it is not faithful. Our ranker ensures that the more faithful summary is ranked higher. 1b shows the lookahead heuristics that provide a faithfulness score given the full future summary. The model assigns a higher score to the word "World" than "British". However, by looking ahead we know that the completed summary following the most likely token will result in an unfaithful summary. Hence, the lookahead heuristics will ensure selecting the token "British" so that the resulting summary will be faithful.

ness heuristic that looks into the future to generate a full summary starting with the current tokens of any partially generated summary so as to provide a faithfulness score of the future summary during generation. The added heuristic ensures that the selected tokens will lead to a more faithful path in the search space. Compared to the baseline decoding strategies we analyzed, the two proposed methods significantly improve faithfulness as evaluated by four automatic faithfulness metrics and further confirmed by human evaluation.

Finally, to overcome the computational and runtime overhead of our proposed decoding methods, we explore distillation to transfer the knowledge of generating faithful summaries from a teacher model to a student model. Specifically, we use the faithfulness-aware decoding strategies as the teacher model to generate reference summaries. Then, we train student models, which have not been fine-tuned on the original task, to imitate the more faithful generation techniques using an additional cross-entropy loss between the generated summaries by the student and teacher models. Results indicate that the student model is able to generate summaries of similar faithfulness to that of the full teacher model while reducing the decoding time (seconds per example) up to $1/6$ of what the teacher model takes. This process can be performed iteratively by using the student model as the teacher for the next iteration (See Figure 2). With each iteration, the new student model is able to generate more faithful summaries, and outperform the original teacher model with just two iterations.

To summarize, our contributions are:

1. An analysis of the effect of popular decoding strategies, including greedy, beam, and nucleus sampling, on the faithfulness of abstractive summarization.
2. Two faithfulness-aware generation methods, ranking and lookahead, that improve faithfulness over existing decoding strategies.
3. A simple distillation approach that allows a student model to generate faithful summaries with just greedy decoding.

2 Faithfulness Behavior of Popular Decoding Strategies

We first describe our experiment investigating the effect of popular decoding strategies on faithfulness. We wish to primarily investigate whether better exploration of the search space, such as the candidate expansion with beam search, can improve faithfulness, and how randomness introduced through sampling impacts faithfulness. These investigations in turn motivate our more advanced, faithfulness-aware decoding strategies in Section 3.

Decoding Strategies (Greedy, Beam, and Nucleus Sampling). For generation, we assume the common left-to-right, auto-regressive setting where the model generates a summary y with n tokens given the input document x:

$$P(y|x) = \prod_{t=1}^{n} p(y_t|y_{1:t-1}, x)$$

The summary tokens are selected with probability according to the decoding strategies. We explore three common decoding strategies: greedy, beam search, and nucleus sampling (Holtzman et al., 2020). **Greedy search** selects the next token by the most probable token $y_t = \arg\max_y p(y|y_{1:t-1}, x)$. **Beam search** extends greedy search by keeping top-k hypothesis at each time step, where k is the number of beams. Another approach to decoding is to use sampling, where we consider **nucleus sampling**. Holtzman et al. (2020) surprisingly find that methods that optimize probability, such as beam search, may lead to text degeneration, and thus propose nucleus sampling, a method that randomly selects from top tokens whose cumulative probability satisfies the threshold p. A small p means less randomness and becomes greedy search, while a large p allows for a more diverse output.

3 Faithfulness-Aware Decoding Strategies

We hypothesize (and later test and confirm whether it is true in Section 6.1 and Appendix E) that current decoding methods, such as beam search which explores a large space, may not explore the paths that focus on faithfulness directly and effectively. Hence, we propose two faithfulness-aware methods that can be applied on top of the base decoding strategies to modify how the space is explored from two different perspectives: (1) Ranking makes use of the large exploration of beam search and picks the explored path that is most faithful; (2) Lookahead directly guides the search process by adding faithfulness heuristics when selecting the next token starting from the initial decoding process.

3.1 Ranking with Faithfulness Metrics

Since beam search already explores many different suitable candidates during the decoding process, we hypothesize that more faithful summaries exist in the list of possible candidates, even if the model score is not directly optimized towards faithfulness (we show that this is true later in Section 6.1). Thus, we propose to rerank the generated candidates from beam search according to faithfulness metrics.

The process is illustrated in Figure 1a. Assuming a beam search with beam size k, we have k summaries generated by the decoding method. We compute a faithfulness metric (details of the metrics are presented in Section 5.2) over all summaries and select the summary that achieves the highest faithfulness score. In the example, the more faithful summary that was originally ranked low according to model score is now ranked as the top summary according to faithfulness.

Re-ranking candidates for abstractive summarization have been studied primarily from the informativeness perspective (Ravaut et al., 2022a,b), and our focus is on improving faithfulness. Our idea is most similar to Falke et al. (2019), where the authors use NLI models to re-rank. However, the results indicate that the NLI performance does not translate to improvement in faithfulness; their best-ranking model actually increases the number of unfaithful summaries at the top summary after re-ranking by 3%. The authors attribute it to domain shift and NLI models relying on simple heuristics like lexical matching. We thus explore using faithfulness metrics directly for ranking.

Composite Metric. While it is possible to use one of the faithfulness metrics to rank the candidates, it often leads to over-fitting for one particular metric (each metric can have its own domain biases and idiosyncrasies) and hurts the overall faithfulness scores evaluated by other metrics. We instead tune a composite metric that aggregates the vote of several popular metrics (See Section 5.4). We use linear regression to provide weights for each metric and tune on human judgments of faithfulness. We refer the readers to Appendix D and Appendix E for details and ablations for the composite metric.

3.2 Lookahead

Lu et al. (2022) use lookahead to provide a future constraint satisfaction estimate and show its effectiveness in several constrained generation tasks (commonsense generation, constrained machine translation, table-to-text generation, and constrained question generation). We extend this idea to improve faithfulness of abstractive summarization. Instead of relying on explicit constraints that are available for the constrained generation tasks, we use reference-free faithfulness metrics on the full future summaries as an estimate. Unlike reranking which is constrained by the search space explored by beam search, lookahead allows for exploration of a much larger number of candidates.

Figure 1b shows an example of the lookahead. When selecting the next token, the usual decoding scheme would select the word "World" that has the highest probability. However, if we were to follow this path, the resulting summary would introduce hallucinations. Instead, we would like to guide the

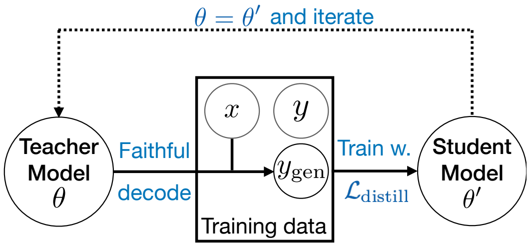


Figure 2: Illustration of the iterative distillation process. We train a student model θ' with summaries generated by the teacher model θ, which uses faithfulness-aware decoding methods. The resultant student model θ' that is trained on more faithful summaries can in turn be used as θ to generate the training data for the next iteration.

model to select the less probable token "British," which will yield a faithful summary sentence.

Formally, each summary token is selected by:

$$f(y_t) = \log P(y_{1:t} \mid x) + w \cdot \max_{L_{y \leq t}} h(y_{1:t+l}, x)$$

where $\log P(y_{1:t} \mid x)$ is the model score, $h(\cdot)$ is a reference-free faithfulness evaluation function that assigns a score to the summary, w is the weight, and l is the number of tokens to look into the future.

Here, $L_{y \leq t}$ is a set of possible generated summaries that start with the summary tokens $y_{1:t}$. The number of summaries for L varies given the decoding strategies we use to generate future summaries. Greedy search and sampling produce a single expansion, and beam search produces k number of summaries depending on the beam size. Although the lookahead length l can be specified, we instead generate the full summary, as current faithfulness metrics expect full summaries as input and do not work well on partial summaries (see Appendix E).

3.3 Combining Ranking and Lookahead

We can combine the two methods to further improve faithfulness. We first use the BEAM+LOOKAHEAD to generate faithful beam candidates and then select the best candidates with ranking. We refer to this method as BEAM+LOOKAHEAD+RANKING.

4 Efficient Decoding via Distillation

One drawback of the proposed decoding methods is the heavy computational cost during decoding. We thus explore using distillation to transfer the knowledge of faithfulness-aware decoding to a student model that can generate summaries of similar faithfulness with just greedy decoding. We note here that our distillation aims at improving the decoding time rather than downsizing the model. Similar to Kim and Rush (2016), we assume that we have a teacher model and a student model. In our setting, the teacher model does not necessarily need to be a different model, but it needs to decode with more faithfulness-aware methods. Typical distillation methods use the teacher's probability distribution (Kim and Rush, 2016) as the target for the student model to imitate. In our case, however, that distribution is the same for all methods – the difference lies in how the probability is used to generate the next tokens. Thus, we propose a new decoding distillation loss. We use the teacher model to generate summaries y_{gen} as additional reference summaries, and interpolate between the cross-entropy loss using the original reference summaries and the cross-entropy loss where we consider y_{gen} as reference summaries. Formally, the training loss is:

$$\mathcal{L}_{\text{distill}} = \mathcal{L}_{\text{XE}}(y', y) + \lambda \mathcal{L}_{\text{XE}}(y', y_{\text{gen}})$$

where $\mathcal{L}_{\text{XE}}$ is the cross entropy, y' is the generated summary by the student model, and λ is a hyperparameter for the weight of the cross-entropy loss on the generated summaries.

Iterative Distillation. While we use the student model with just greedy decoding to improve decoding speed, the student model can also benefit from using our proposed faithfulness-aware decoding methods. Thus, the student models can also serve as a new teacher model to distill more faithfulness knowledge to a new student model. The distillation process thus becomes iterative, illustrated in Figure 2. We use the trained student model as a new teacher model, where we decode with our proposed faithfulness methods to create additional reference summaries y_{gen} for the next iteration.

5 Experiments

5.1 Datasets and Models

We perform experiments on two popular datasets for abstractive summarization, XSum (Narayan et al., 2018) and CNN/DM (Hermann et al., 2015). More details on the datasets are described in Appendix A.1. We use the released checkpoint of BART-large (406M) for the two datasets.[2] The

---

[2] We use the checkpoint BART-LARGE-XSUM (`https://huggingface.co/facebook/bart-large-xsum`)

same experiment is done with PEGASUS (Zhang et al., 2020a), which is presented in Appendix B.

5.2 Evaluation Metrics

We use the F1 measure of ROUGE-L (Lin, 2004, RL), i.e., the overlap of the longest common subsequence between a generated summary and reference summary, and the F1 measure of BERTScore (Zhang et al., 2020b, BS) to evaluate summary quality. In addition, we use BS-Fact, i.e., the BERTScore precision of a summary with respect to its source document rather than the reference summary, FactCC (Kryscinski et al., 2020), DAE (Goyal and Durrett, 2021), and QuestEval (Scialom et al., 2021) for faithfulness evaluation. Details of the metrics are presented in section A.

5.3 Human Evaluation Setup

We use Amazon Mechanical Turk (AMT) to ask human annotators to judge the faithfulness and informativeness of the summaries generated with different decoding.

Faithfulness. We ask workers to judge the faithfulness of a summary sentence using a 3-star rating (1=major factual error, 2=minor factual error, 3=no factual error. Three judgments per summary are then aggregated using majority voting. We randomly select 200 examples from both datasets and use the summaries generated using greedy, sampling, beam search, as well as the ranking and lookahead strategies applied to beam search. We report the percentage of summaries that are fully factual (i.e. the percentage of summaries rated as 3-star) as the faithfulness score, and also report the distribution of summaries rated as 1, 2, and 3 stars. Details on qualification, payment and other aspects of the evaluation can be found in Appendix A.4.

Informativeness. We also evaluate the generated summaries in terms of informativeness. We consider summary to be informative if its content is important and relevant, but it does not necessarily need to be long. We use best-worst-scaling (BWS) for evaluating the informativeness of the generated summaries, as this method is "a less labor-intensive alternative to paired comparisons that has been shown to produce more reliable results than rating scales" (Kiritchenko and Mohammad, 2017). Accordingly, for each dataset, we select 200 random articles with the corresponding summaries from five systems in random order. We ask three annotators to select the most informative ("best") and the least informative ("worst") among the five. A rating per system is computed as the percentage of times it is chosen as best minus the percentage of times it is selected as worst. A value of 100 means that the system has been unanimously picked as "best", whereas a value of -100 means that the system has been unanimously picked as "worst". Additional details, as well as the screenshot of the annotation interface, are in Appendix A.4.

5.4 Decoding Setting Details

We describe the settings of the basic decoding methods, our faithfulness-aware decoding methods and distillation. More details are in Appendix A.3.

Basic Decoding Method. We compare the summaries generated using greedy search, beam search ($k = 10$), and nucleus sampling ($p = 0.9$). Additional experiments with various beam sizes and top-p values can be found in Appendix B.

Ranking and Composite Metric. We use beam search ($k = 10$) and rank the candidates using the composite metric introduced in Section 3.1. To train the composite metric, we explore combining FactCC, BS-Fact, DAE, and QuestEval. We use FACTCOLLECT (Ribeiro et al., 2022), a large collection of four faithfulness annotations to train a linear regression on the human-labeled faithfulness judgments. More details of the composite metric and its robustness to another domain can be seen in Appendix D.

Lookahead. We use BS-Fact as the faithfulness metric for the lookahead as it correlates highly with human judgment (Pagnoni et al., 2021) and is quick to compute without the need for additional pre-processing. We use greedy search to generate future summaries and apply it to both greedy and beam searches.

Distillation. We use the checkpoint of our two proposed faithfulness-aware decoding methods as the teacher model, and train the student model from BART-LARGE.[3] We follow the original fine-tuning hyperparameters provided by the authors (Lewis et al., 2020) and use $\lambda = 1$ for the weight of the additional cross-entropy loss.

---

and BART-LARGE-CNN (`https://huggingface.co/facebook/bart-large-cnn`).

[3] `https://huggingface.co/facebook/bart-large`

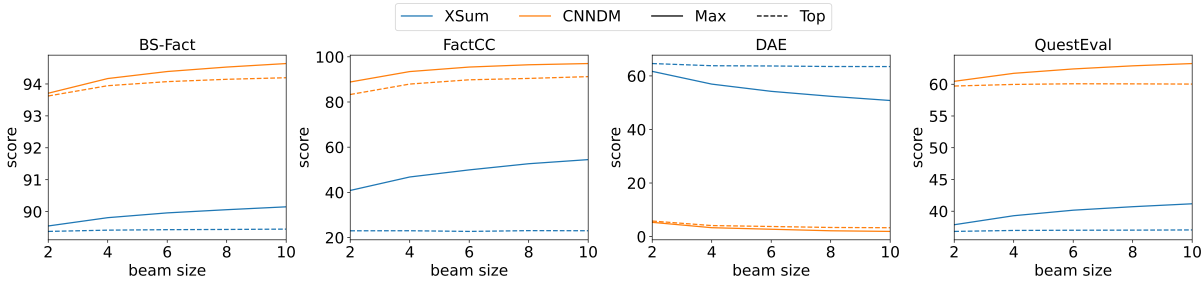


Figure 3: Maximum possible score (Max) for each faithfulness metric and the faithfulness scores of the top candidate (Top) at various beam sizes. As beam size increases, more faithful summaries exist in the list of candidates, but the faithfulness of the top beam improves only slightly.

	RL	BS	BS-Fact	FactCC	DAE ↓	QuestEval
			CNN/DM			
Greedy	**30.93**	**88.39**	93.15	69.61	8.15	59.13
Nucleus	27.64	87.90	91.76	54.05	21.61	56.43
Beam	29.99	88.03	**94.20**	**84.23**	**3.30**	**60.03**
			XSum			
Greedy	36.16	92.03	89.28	**23.53**	65.35	36.51
Nucleus	31.15	91.26	88.62	21.04	76.20	34.98
Beam	**37.11**	**92.12**	**89.45**	22.97	**63.49**	**37.05**

Table 1: Baseline results of popular decoding methods measured by summarization quality metrics (Rouge-L (RL) and BertScore (BS)) and faithfulness metrics. We observe a general trend where beam search performs the best and nucleus sampling performs the worst in terms of faithfulness. Full result with different beam sizes and top-p probability for nucleus sampling is in Table 7.

6 Results

6.1 Baseline Decoding Results

We show the analysis of common decoding strategies in Table 1. Both datasets show a similar trend. Beam search performs the best in terms of faithfulness except for FactCC on the XSum dataset. Compared to greedy decoding, which is beam search with $k = 1$, the candidate expansion with a larger beam size provides better exploration for faithfulness. Nucleus sampling degrades faithfulness compared with greedy search, showing that the introduced randomness is not helpful for faithfulness. This aligns with observations from Narayan et al. (2022) and Chen et al. (2021a), which show that nucleus sampling produces less relevant text for data-to-text generation.

The results are surprisingly mixed for both datasets in terms of summary quality, i.e., RL and BS scores. Comparing beam search with greedy decoding, we see improvement of both scores on XSum but not for CNN/DM. Nucleus sampling, on the other hand, is also worse than greedy search on this aspect, suggesting that randomness may not be suited for the task of abstractive summarization.

Search Space for Beam Search. Inspired by Xu et al. (2022) who hinted at the potential of better faithfulness with a large exploration of the search space, we use beam search to explore whether larger beam sizes (and hence larger exploration) derive more faithful summaries. To this end, we use all summaries generated by beam search and select the beam that would result in the highest possible score for each metric. We show the maximum score (Max) for the four faithfulness metrics and the faithfulness score of selecting the top beam (Top) given different beam sizes in Figure 3. We see a clear trend that increasing the beam size improves all faithfulness scores. This confirms our hypothesis that larger exploration of the search space can provide additional faithfulness gain, and thus showing the potential of our proposed decoding strategies, especially our reranking strategy, to output more faithful summaries. The faithfulness scores of TOP only increase marginally compared to the increase for Max, showing the importance of having better faithfulness guidance, such as our proposed faithfulness lookahead heuristics.

6.2 Faithfulness-Aware Decoding Results

We now show the impact of faithfulness-aware methods compared with the traditional decoding methods, which is shown in Table 2. We first observe that applying ranking on top of beam search improves faithfulness significantly over beam search, as measured by all faithfulness metrics. Specifically, QuestEval reaches 62.57 (2.5 points improvement) and 40.10 (3.1 points improvement) on CNN/DM and XSum respectively. DAE error rate reduces from 63.49 to 51.48 and 3.30 to 1.92, which is a relative improvement of

	Rouge-L	BERTScore	BS-Fact	FactCC	DAE ↓	QuestEval
			CNN/DM			
Greedy	**30.93**	**88.39**	93.15	69.61	8.15	59.13
Beam	29.99	88.03	94.20	84.23	3.30	60.03
Beam+Ranking	30.08	88.12	94.31	90.27	1.92	62.57
Greedy+Lookahead	30.75	88.35	93.90	71.54	5.70	60.13
Beam+Lookahead	28.66	87.84	**95.32**	86.10	1.68	61.80
Beam+Lookahead+Ranking	28.86	87.92	95.26	**91.68**	**1.08**	**63.69**
			XSum			
Greedy	36.16	92.03	89.28	23.53	65.35	36.51
Beam	**37.11**	**92.12**	89.45	22.97	63.49	37.05
Beam+Ranking	36.42	92.10	89.79	**40.11**	51.48	40.10
Greedy+Lookahead	36.25	92.11	89.71	24.21	60.46	37.17
Beam+Lookahead	35.27	91.94	**90.78**	23.38	50.04	39.24
Beam+Lookahead+Ranking	34.71	91.90	**90.78**	38.86	**41.04**	**41.94**

Table 2: Results for our proposed decoding strategies. Compared to the baseline methods (greedy and beam search), both ranking and lookahead improve faithfulness. The combination of both methods further increases faithfulness.

18.92% (12.01 points) and 41.8% percent (1.38 points) on XSum and CNN/DM, respectively.

We observe similar improvement for lookahead as well, where applying the lookahead improves the faithfulness over the base decoding strategy over all faithfulness metrics. Nevertheless, the base decoding strategy is still the dominating factor, as Beam+Lookahead generates more faithful summaries than greedy+lookahead for all faithfulness metrics. Greedy+Lookahead outperforms Beam on the XSum dataset, showing that better guidance with future faithfulness heuristics can improve faithfulness without large exploration. Finally, the combination of lookahead and ranking can further improve faithfulness as evaluated by FactCC, DAE, and QuestEval.

In terms of ROUGE score, applying faithful decoding methods decreases RL. This tradeoff between faithfulness and ROUGE has been observed in many prior works (Chen et al., 2021b; Kryscinski et al., 2020; Wan and Bansal, 2022). One reason for this phenomenon is that more than 70% of the reference summaries contain hallucinations (Maynez et al., 2020), so the more faithful summaries that do not contain such hallucinations will have lower ROUGE scores. To investigate this problem, we perform a human evaluation study, where we find that the summaries generated by Beam+Lookahead are considered to be most informative. More details are in Appendix A.4.

6.3 Human Evaluation Results

Faithfulness. The observation on automatic faithfulness metrics aligns with the result of human evaluation in Table 3. For XSum, among the baseline decoding methods, we see that sampling performs the worst. Interestingly, greedy is more faithful than beam search, but the difference is only 1.5 points. Our proposed decoding strategies generate summaries that are judged more faithful compared to that of the baseline decoding strategies. Specifically, Beam+Lookahead reaches 56.5, even outperforming Beam+Ranking by 5 points. We also observe that our proposed methods are able to significantly reduce the percentage of summaries that are considered to contain major factual errors; Compared to beam search, ranking reduces the percentage from 44.5 to 36.5, and lookahead further reduces the percentage by 3 points. For CNN/DM, we see the striking result that the summaries generated by our proposed methods achieve the highest faithfulness, and among the two systems, there are no major errors for Beam+Lookahead.

Informativeness. The result is shown in Table 4. The output of the Beam+Lookahead is clearly seen as the most informative among the five methods. This result suggests that Rouge-L and BERTScore may not be good indicators for informativeness, as Beam+Lookahead achieves the lowest scores for the two automatic metrics on both datasets.

6.4 Abstractiveness

Models can "trivially" become more faithful by becoming more extractive (Dreyer et al., 2023), and thus it is important to understand where the gain in faithfulness stems from. We experiment on XSum, as methods can achieve larger improve-

	XSum			CNN/DM		
	1	2	3	1	2	3
Greedy	43.0	12.0	45.0	4.0	3.5	92.5
Sampling	55.0	13.0	32.0	8.0	10.5	81.5
Beam	44.5	12.0	43.5	0.0	2.0	98.0
Beam+Ranking	36.5	14.0	49.5	0.5	1.0	**98.5**
Beam+Lookahead	31.5	12.0	**56.5**	0.0	1.5	**98.5**

Table 3: Human evaluation results on faithfulness with the 3-star rating system (1=major factual error, 2=minor factual error, 3=no factual error). Our proposed faithfulness-aware methods are judged as the most faithful (the percentage of summaries rated as 3), confirming our observation with automatic faithfulness metrics.

ment in faithfulness and thus potentially more gain through extensiveness. We experiment with the 200 examples used for human evaluation and calculate Mint (Dreyer et al., 2023) for abstractiveness and plot this score against the human-labeled faithfulness, similar to Ladhak et al. (2022). The result is shown in Figure 4. Similar to the observation of Dreyer et al. (2023), more faithful models tend to be more extractive; however, the gain in faithfulness is considerably larger than the decrease in abstractiveness. For example, comparing Beam+Lookahead with beam search, the relative increase in faithfulness (29.89%) is quadruple the decrease (7.27%) in abstractiveness. Similar experiments on CNN/DM are in Appendix F.

Lookahead with Faithfulness and Abstractiveness. We further show that our lookahead method can easily allow additional heuristics, such as balancing both faithfulness and abstractiveness. Specifically, we replace $h(\cdot)$ with combination of BS-Fact and Mint:

$$h(y, x) = \alpha \text{BS-Fact}(y, x) + (1 - \alpha)\textsc{mint}(y, x)$$

We use $\alpha = 0.75$ and the same hyper-parameters as Beam+Lookahead. We refer to this model as Beam+Lookahead+Abstr and show the point in Figure 4. Compared to Beam+Lookahead, this model can increase abstractiveness at a small cost in faithfulness, demonstrating the flexibility of our lookahead method to incorporate various characteristics for summarization.

6.5 Distillation

We present the distillation result in Table 5. While the student models are not able to outperform the teacher models, they approach the performance of the teacher models. The student models are also able to generate more faithful summaries compared to the greedy search baseline, which is only trained using the cross-entropy loss $\mathcal{L}_{\text{XE}}(y', y)$.

	XSum	CNN/DM
Greedy	3.0	− 8.2
Sampling	−20.5	−23.8
Beam	1.8	8.5
Beam+Ranking	1.0	− 2.8
Beam+Lookahead	**17.7**	**31.0**

Table 4: Human evaluation results on informativeness with best-worst-scaling (100=unanimous best, −100=unanimous worst).

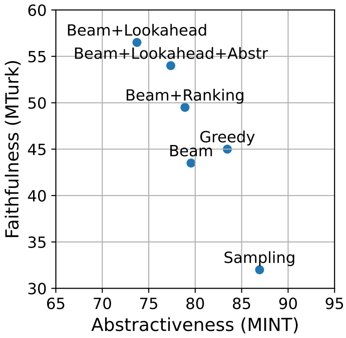


Figure 4: Faithfulness and abstractiveness tradeoff results on 200 XSum examples used for human annotation. Beam+Lookahead+Abstr is the model that is trained with additional abstractiveness heuristics (See Section 6.4 for more details).

The main benefit of the student model comes from the improved decoding speed. The ranking time reduces from 0.77 seconds per example to 0.47, which is a 40% improvement. The largest gain can be seen for lookahead, where the decoding speed reduces from 3 seconds per example to 0.49, only 1/6 of the time it was originally taking.

For example, the student model distilled from Beam+Ranking improves DAE by 6.6 points and QuestEval by a point compared to the greedy search baseline and only differs from the teacher model by 2.5 points for DAE and 0.5 points for QuestEval. When using a more faithful teacher model, i.e. Beam+Lookahead, the student model is able to generate more faithful summaries, as evaluated by BS-Fact, DAE, and QuestEval.

Iterative Distillation. Next, we show the result of distilling Beam+Ranking iteratively on XSum in Table 6. We see that with each iteration, the model is able to improve faithfulness further. When compared to the original teacher model, Beam+Ranking, the student model is able to out-

	RL	BS	BS-Fact	FC	DAE ↓	QE	Speed
Greedy	36.16	92.03	89.28	23.53	65.35	36.51	0.39
BEAM+RANKING							
Teacher	36.46	**92.14**	**90.15**	**22.47**	**56.33**	**37.98**	0.77
Student	**36.59**	92.07	89.80	23.52	58.78	37.46	**0.47**
BEAM+LOOKAHEAD							
Teacher	35.91	92.06	**90.51**	22.44	**52.50**	**38.72**	3.00
Student	**36.52**	**92.07**	89.97	**22.58**	58.02	37.89	**0.49**

Table 5: Distillation results using our proposed faithfulness-aware decoding methods as the teacher. We abbreviate FactCC as FC and QuestEval as QE. Speed is calculated by seconds per summary.

	RL	BS	BS-Fact	FactCC	DAE ↓	QuestEval
Teacher	36.46	**92.14**	90.15	22.47	56.33	37.98
Iter. 1	**36.59**	92.07	89.80	23.52	58.78	37.46
Iter. 2	35.95	91.95	90.16	23.14	54.01	38.10
Iter. 3	35.09	91.73	90.48	22.77	50.66	38.86
Iter. 4	34.32	91.54	90.81	24.49	47.83	39.64
Iter. 5	33.60	91.34	**91.11**	**25.52**	**45.85**	**40.39**

Table 6: Iterative distillation results using BEAM+RANKING as the teacher decoding method. With two iterations, the student model is able to outperform the original teacher model in terms of faithfulness, and further iterations continuously improve faithfulness.

perform all faithfulness metrics with two iterations. We stress that here all models only use greedy decoding, thus showing the potential of combining decoding with training for more faithful models.

7 Related Work

Many of the related works of our proposed decoding methods have been discussed in Section 3; here we cover other related areas.

Decoding methods. A decoding method for text generation explores an approximate search method to select the best tokens to form a hypothesis. Several works have critically analyzed different decoding strategies for natural language generation, including beam search (Meister et al., 2020a; Stahlberg and Byrne, 2019; Xu et al., 2022; Holtzman et al., 2020), best-first-search (Meister et al., 2020b), and lattice (Xu et al., 2022). While these works investigated the effectiveness of decoding methods on generated outputs from the perspective of diversity and repetitiveness, to our best knowledge, none of the works have explicitly analyzed their performance on faithfulness.

Distillation. Distillation aims at compressing the knowledge from a larger model into a smaller one. A conventional approach uses soft targets, i.e. learning the logits of a teacher model rather than final predictions (Buciluundefined et al., 2006; Hinton et al., 2015; Kim and Rush, 2016). While this method has shown to be very effective, it is less applicable to our case where the underlying distribution for the next probable tokens does not necessarily change (for ranking, we do not modify the model scores at all) and thus not useful to learn soft labels. Different from compressing model size, our approach focuses on reducing the computational cost during decoding. Our method is most similar to pseudo-labeling (Shleifer and Rush, 2020), where we use generated summaries as "hard" labels. We do not replace reference summaries with our generated ones. Instead, we use interpolation (Kim and Rush, 2016) to account for both faithfulness and quality.

8 Conclusion

In this paper, we show a thorough analysis of the effect of decoding strategies on faithfulness for abstractive summarization. We present an analysis of popular decoding strategies, as well as our two newly proposed faithfulness-aware decoding strategies, ranking and lookahead, that can further improve faithfulness upon the base decoding methods. Finally, we show a simple (and optionally iterative) distillation trick where the training of a student model incorporates the summaries generated with more faithfulness-aware methods, and the student model generates summaries of similar faithfulness with minimal decoding time.

Future experiments could extend similar analysis of faithfulness and factuality beyond summarization and develop a combination of heuristics that also encompasses other aspects and styles.

9 Limitations

While the decoding strategies with lookahead show improvement in faithfulness, they require a heavy computational overhead, especially when they are coupled with beam search for the base decoding strategy and for generating the future summary. We provide one solution with our distillation to improve decoding speed. Many of the computations, including the generated future summaries and the faithfulness scores on them, during this online process, are also later disregarded, similar to how any candidates are pruned during beam search. We believe an interesting direction might be to store the already generated future summaries so that the decoding may directly use the future summary if it is considered a good summary candidate.

10 Ethical Impact

While our work aims to reduce potential malicious or unintended harmful effects, our methods rely on the use of faithfulness metrics. The inherent problems and biases when using such metrics have been under-studied. Our decoding strategies can also be applied to be used for other metrics, even those that could be optimized for malicious intents. Another aspect to consider is the environmental impact of our proposed methods, as they require large computations. We hope that our distillation can mitigate this problem and future work can work towards more environmentally friendly approaches while improving faithfulness for safer use of large models.

References

Cristian Buciluundefined, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 535–541, New York, NY, USA. Association for Computing Machinery.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mingda Chen, Sam Wiseman, and Kevin Gimpel. 2021a. WikiTableT: A large-scale data-to-text dataset for generating Wikipedia article sections. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 193–209, Online. Association for Computational Linguistics.

Sihao Chen, Fan Zhang, Kazoo Sone, and Dan Roth. 2021b. Improving faithfulness in abstractive summarization with contrast candidate generation and selection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5935–5941, Online. Association for Computational Linguistics.

Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. 2020. Multi-fact correction in abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9320–9331, Online. Association for Computational Linguistics.

Markus Dreyer, Mengwen Liu, Feng Nan, Sandeep Atluri, and Sujith Ravi. 2023. Evaluating the tradeoff between abstractiveness and factuality in abstractive summarization. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Findings*. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2021. Annotating and modeling fine-grained factuality in summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1449–1462, Online. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset.

K. Krippendorff. 1980. *Content Analysis: An Introduction to Its Methodology*. Commtext Series. SAGE Publications.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9332–9346, Online. Association for Computational Linguistics.

Faisal Ladhak, Esin Durmus, He He, Claire Cardie, and Kathleen McKeown. 2022. Faithful or extractive? on mitigating the faithfulness-abstractiveness trade-off in abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1410–1421, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. 2022. NeuroLogic a*esque decoding: Constrained text generation with lookahead heuristics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 780–799, Seattle, United States. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020a. If beam search is the answer, what was the question? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2173–2185, Online. Association for Computational Linguistics.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020b. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8:795–809.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Shashi Narayan, Gonçalo Simões, Yao Zhao, Joshua Maynez, Dipanjan Das, Michael Collins, and Mirella Lapata. 2022. A well-composed text is half done! composition sampling for diverse conditional generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1319–1339, Dublin, Ireland. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Mathieu Ravaut, Shafiq Joty, and Nancy Chen. 2022a. SummaReranker: A multi-task mixture-of-experts re-ranking framework for abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4504–4524, Dublin, Ireland. Association for Computational Linguistics.

Mathieu Ravaut, Shafiq Joty, and Nancy F Chen. 2022b. Towards summary candidates fusion. In *EMNLP 2022*.

Leonardo F. R. Ribeiro, Mengwen Liu, Iryna Gurevych, Markus Dreyer, and Mohit Bansal. 2022. FactGraph: Evaluating factuality in summarization with semantic graph representations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3238–3253, Seattle, United States. Association for Computational Linguistics.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization asks for fact-based evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Shleifer and Alexander M. Rush. 2020. Pre-trained summarization distillation.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Liam van der Poel, Ryan Cotterell, and Clara Meister. 2022. Mutual information alleviates hallucinations in abstractive summarization.

David Wan and Mohit Bansal. 2022. FactPEGASUS: Factuality-aware pre-training and fine-tuning for abstractive summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1010–1028, Seattle, United States. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wen Xiao and Giuseppe Carenini. 2022. Entity-based spancopy for abstractive summarization to improve the factual consistency.

Jiacheng Xu, Siddhartha Jonnalagadda, and Greg Durrett. 2022. Massive-scale decoding for text generation using lattices. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4659–4676, Seattle, United States. Association for Computational Linguistics.

Haopeng Zhang, Semih Yavuz, Wojciech Kryscinski, Kazuma Hashimoto, and Yingbo Zhou. 2022. Improving the faithfulness of abstractive summarization via entity coverage control. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 528–535, Seattle, United States. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Experiments Details.

A.1 Datasets

We evaluate on XSum and CNN/DM. We use the dataset processed and provided by DATASETS (Lhoest et al., 2021).[4] Both datasets contain English news articles and the corresponding summaries. XSum contains 204045, 11332, and 11334 examples for training, validation, and test set, respectively, and CNN/DM contains 287113, 13368, and 11490 for the splits.

A.2 Metrics

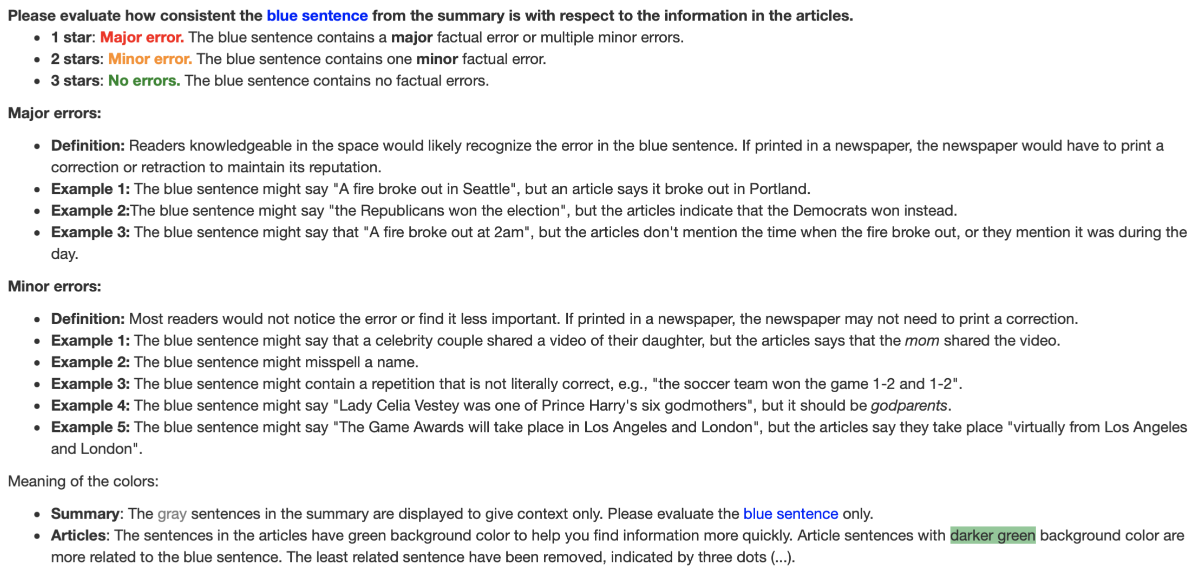
We use the official code and follow the instructions to set up and run all the metrics we used. We use the ROUGE package from `https://github.com/google-research/google-research/tree/master/rouge`. We report all scores of our models from single runs. For BS and BS-Fact, we use the default model for English (ROBERTA-LARGE). For DAE, we use the sentence error, which considers the sentence to contain an error if one of its arcs is predicted to be not factual.

A.3 Decoding Details

Basic Decoding Method Details. We use the official generation code provided by TRANSFORMERS

---

[4] The link to the processed data is found in `https://huggingface.co/datasets/xsum` and `https://huggingface.co/datasets/cnn_dailymail`.



Please evaluate how consistent the blue sentence from the summary is with respect to the information in the articles.

- **1 star**: **Major error.** The blue sentence contains a **major** factual error or multiple minor errors.
- **2 stars**: **Minor error.** The blue sentence contains one **minor** factual error.
- **3 stars**: **No errors.** The blue sentence contains no factual errors.

Major errors:

- **Definition:** Readers knowledgeable in the space would likely recognize the error in the blue sentence. If printed in a newspaper, the newspaper would have to print a correction or retraction to maintain its reputation.
- **Example 1:** The blue sentence might say "A fire broke out in Seattle", but an article says it broke out in Portland.
- **Example 2:**The blue sentence might say "the Republicans won the election", but the articles indicate that the Democrats won instead.
- **Example 3:** The blue sentence might say that "A fire broke out at 2am", but the articles don't mention the time when the fire broke out, or they mention it was during the day.

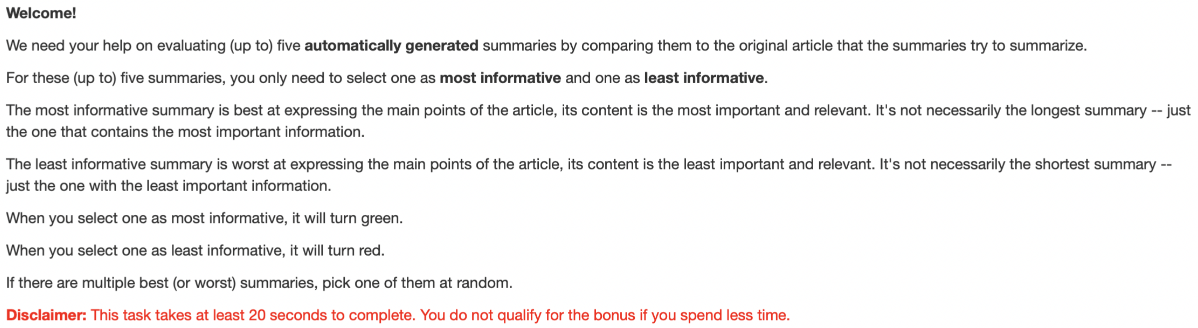
Minor errors:

- **Definition:** Most readers would not notice the error or find it less important. If printed in a newspaper, the newspaper may not need to print a correction.
- **Example 1:** The blue sentence might say that a celebrity couple shared a video of their daughter, but the articles says that the *mom* shared the video.
- **Example 2:** The blue sentence might misspell a name.
- **Example 3:** The blue sentence might contain a repetition that is not literally correct, e.g., "the soccer team won the game 1-2 and 1-2".
- **Example 4:** The blue sentence might say "Lady Celia Vestey was one of Prince Harry's six godmothers", but it should be *godparents*.
- **Example 5:** The blue sentence might say "The Game Awards will take place in Los Angeles and London", but the articles say they take place "virtually from Los Angeles and London".

Meaning of the colors:

- **Summary**: The gray sentences in the summary are displayed to give context only. Please evaluate the blue sentence only.
- **Articles**: The sentences in the articles have green background color to help you find information more quickly. Article sentences with darker green background color are more related to the blue sentence. The least related sentence have been removed, indicated by three dots (...).

Figure 5: Annotation instructions to annotate factual consistency on Mechanical Turk.



Welcome!

We need your help on evaluating (up to) five **automatically generated** summaries by comparing them to the original article that the summaries try to summarize.

For these (up to) five summaries, you only need to select one as **most informative** and one as **least informative**.

The most informative summary is best at expressing the main points of the article, its content is the most important and relevant. It's not necessarily the longest summary -- just the one that contains the most important information.

The least informative summary is worst at expressing the main points of the article, its content is the least important and relevant. It's not necessarily the shortest summary -- just the one with the least important information.

When you select one as most informative, it will turn green.

When you select one as least informative, it will turn red.

If there are multiple best (or worst) summaries, pick one of them at random.

Disclaimer: This task takes at least 20 seconds to complete. You do not qualify for the bonus if you spend less time.

Figure 6: Annotation instructions to annotate informativeness on Mechanical Turk.

(Wolf et al., 2020). We use a single NVIDIA V100 GPU to generate the summaries. Greedy and sampling experiments take around 2 hours and beam search variants take 4 hours.

Ranking. We do not need to do additional computation as we already have the outputs of beam search and the metric scores.

Lookahead. To reduce computational overhead, we only calculate and incorporate the lookahead heuristics for the top 5 tokens according to the model score at each time step. Experiments using beam search or sampling to generate a future summary can be found in Appendix C. We show additional ablations on the length of the future summaries and how the exploration changes with the heuristics in Appendix E. For tuning the w, the weight for the heuristics, we search over the interval from 5 to 55 with a step of 5, and evaluate the generated summaries on the development set. We use the average of all metric scores, including RL and BS so that we do not over-optimize for faithfulness. We find 25 to be optimal for CNN/DM and 55 for XSum. The time to run for XSum is 33 hours, and that for CNN/DM is around 70 hours.

Distillation Details. We use the example code from TRANSFORMERS to train summarization models. We follow the authors' hyper-parameters to train BART-Large models. We use 8 V100 GPUs and the training time is around 5 hours. To generate the summaries for the training data, The BS-Fact Ranker takes around 3 hours to generate the summaries when parallelized across the 8 GPUs. Lookahead takes 10 hours to generate the training data split across 8 GPUs.

A.4 Human Evaluation Details

Human Evaluation on Faithfulness. The screenshot of the annotation can be seen in Figure 5. We required annotators to pass a custom

	RL	BS	BS-Fact	FactCC	DAE ↓	QuestEval
CNN/DM						
Greedy	**30.93**	**88.39**	93.15	69.61	8.15	59.13
Nucleus $p = 0.1$	30.93	88.39	93.15	69.58	8.17	59.13
Nucleus $p = 0.3$	30.77	88.37	93.12	69.43	8.40	59.01
Nucleus $p = 0.5$	30.39	88.31	92.95	66.67	10.05	58.81
Nucleus $p = 0.7$	29.43	88.20	92.56	60.88	12.64	58.11
Nucleus $p = 0.9$	27.64	87.90	91.76	54.05	21.61	56.43
Beam $k = 2$	30.78	88.29	93.62	76.00	5.80	59.72
Beam $k = 4$	30.44	88.17	93.95	81.19	4.11	59.97
Beam $k = 6$	30.30	88.12	94.07	82.94	3.76	**60.07**
Beam $k = 8$	30.10	88.07	94.15	83.50	3.39	60.06
Beam $k = 10$	29.99	88.03	**94.20**	**84.23**	**3.30**	60.03
XSum						
Greedy	36.16	92.03	89.28	**23.53**	65.35	36.51
Nucleus $p = 0.1$	31.08	91.24	88.61	21.63	76.07	35.04
Nucleus $p = 0.3$	31.19	91.25	88.62	21.51	76.35	34.98
Nucleus $p = 0.5$	31.08	91.24	88.63	21.11	75.31	34.99
Nucleus $p = 0.7$	31.24	91.24	88.62	21.26	76.11	35.01
Nucleus $p = 0.9$	31.15	91.26	88.62	21.04	76.20	34.98
Beam $k = 2$	36.76	92.13	89.38	22.98	64.62	36.82
Beam $k = 4$	36.96	92.14	89.42	23.00	63.81	36.97
Beam $k = 6$	37.09	**92.14**	89.43	22.70	63.71	37.00
Beam $k = 8$	37.09	92.13	89.44	23.05	63.52	37.02
Beam $k = 10$	**37.11**	92.12	**89.45**	22.97	**63.49**	**37.05**

Table 7: Full results of beam search and nucleus sampling for fine-tuned BART-LARGE models. The trend can still be seen under different beam sizes and top-p values, where increasing k improves faithfulness and increasing p degrades it.

	RL	BS	BS-Fact	FactCC	DAE ↓	QuestEval
CNN/DM						
Greedy	30.20	87.65	89.71	53.00	15.44	56.70
Nucleus $p = 0.1$	30.20	87.65	89.71	52.99	15.44	56.68
Nucleus $p = 0.3$	30.15	87.64	89.71	52.96	15.72	56.68
Nucleus $p = 0.5$	29.88	87.61	89.62	51.48	17.25	56.43
Nucleus $p = 0.7$	28.86	87.47	89.36	46.44	20.97	55.92
Nucleus $p = 0.9$	30.15	87.23	88.88	38.82	28.53	54.78
Beam $k = 2$	30.67	**87.72**	90.28	57.75	11.40	57.24
Beam $k = 4$	**30.82**	87.71	90.61	62.36	9.50	57.42
Beam $k = 6$	30.67	87.66	90.75	64.18	8.72	57.43
Beam $k = 8$	30.68	87.65	90.82	64.96	8.66	57.43
Beam $k = 10$	30.66	87.65	**90.87**	**65.32**	**8.23**	**57.46**
XSum						
Greedy	38.53	92.45	89.05	**24.53**	68.33	35.75
Nucleus $p = 0.1$	38.53	92.44	89.05	24.50	68.33	35.76
Nucleus $p = 0.3$	38.42	92.42	89.02	24.10	69.23	35.68
Nucleus $p = 0.5$	37.85	92.33	88.97	23.14	70.07	35.51
Nucleus $p = 0.7$	36.13	92.09	88.80	22.95	72.72	35.27
Nucleus $p = 0.9$	33.76	91.68	88.51	22.46	76.23	34.73
Beam $k = 2$	39.09	92.53	89.13	23.58	67.72	35.90
Beam $k = 4$	39.35	**92.58**	89.19	22.64	67.16	35.97
Beam $k = 6$	39.32	92.57	89.21	22.89	66.86	**35.98**
Beam $k = 8$	39.37	92.57	89.21	22.71	66.73	35.97
Beam $k = 10$	**39.43**	92.57	**89.23**	22.75	**66.48**	35.96

Table 8: Full results of beam search and nucleus sampling for fine-tuned PEGASUS-LARGE models. We observe a similar observation as Table 7, showing that the faithfulness trend holds for different models.

qualification test consisting of three summaries with factual errors. To pass the test, the annotators had to correctly describe the factual errors in words. Workers also needed to have previously completed 100 or more tasks with an acceptance rate of 95% or higher. We recruited workers from countries whose main language is English. To prevent any one worker from dominating the results, we set a maximum of 100 HITs per worker per dataset. The payment for judging each summary was $0.22 plus a bonus of $0.03. Annotators who spent more than 10 seconds per HIT and maintained high accuracy on HITs with known answers obtained the bonus. Annotators spent a median amount of 57.5 seconds per HIT, which amounts to a pay of $15.65 per hour. Krippendorff alpha (Krippendorff, 1980) for the CNN/DM factuality annotation is 0.63, and Krippendorff alpha for the XSum annotation is 0.57.

Human Evaluation on Informativeness. The screenshot of the annotation can be seen in Figure 6. To achieve good quality, we set up a qualification task of three documents with their associated summaries. A selected pool of workers who had passed previous factuality qualification tests was allowed to take this current qualification test. The workers who passed the current qualification test were allowed to participate in this evaluation. In addition, we added the same three documents with known answers to the evaluation and observed that workers had 100% accuracy on them. We set the same maximum of 100 HITs per worker per dataset as in the factuality evaluation. The pay was $0.40 plus $0.10 bonus per HIT. Annotators spent a median time of 112 seconds per HIT, amounting to a pay of $16.07 per hour. For inter-annotator agreement, Krippendorff alpha (Krippendorff, 1980) for the CNN/DM annotation is 0.22, and Krippendorff alpha for the XSum annotation is 0.32.

B Full Analysis

Table 7 shows the full result. We see the general trend where increasing beam size improves faithfulness and increasing p for sampling is not helpful for faithfulness.

We similarly run the experiment on PEGASUS, a 568M model specifically trained for the task of abstractive summarization, with its respective checkpoints.[5] The result is presented in Table 8.

C Lookahead Methods

We show the result of combining different decoding strategies for the base decoding strategy as well

---

[5] We use the checkpoint from `https://huggingface.co/google/pegasus-cnn_dailymail` and `https://huggingface.co/google/pegasus-xsum`.

	Rouge-L	BERTScore	BS-Fact	FactCC	DAE ↓	QuestEval
			CNN/DM			
Greedy	**30.93**	**88.39**	93.15	69.61	8.15	53.71
Beam $k = 10$	29.99	88.03	94.20	84.23	3.30	60.03
Greedy + Greedy Lookahead	30.88	88.38	93.57	71.54	6.42	59.70
Greedy + Sampling Lookahead	30.67	88.35	93.54	78.28	7.09	59.72
Greedy + Beam Lookahead	30.63	88.32	93.85	82.07	5.33	60.13
Beam + Greedy Lookahead	28.66	87.84	**95.32**	**86.10**	**1.68**	**63.69**
			XSum			
Greedy	36.16	92.03	89.28	23.53	65.35	36.51
Beam $k = 10$	**37.11**	**92.12**	89.45	22.97	63.49	37.05
Greedy + Greedy Lookahead	36.25	92.11	89.71	**24.21**	60.46	37.17
Greedy + Sampling Lookahead	36.24	92.10	89.55	23.97	62.35	36.90
Greedy + Beam Lookahead	36.17	92.07	89.62	23.58	61.90	37.10
Beam + Greedy Lookahead	35.27	91.94	**90.78**	23.38	**50.04**	**39.24**

Table 9: Lookahead results with different decoding strategies for base decoding strategies and the lookahead generation strategies.

	All				CNN/DM				XSum			
	Pearson		Spearman		Pearson		Spearman		Pearson		Spearman	
	ρ	p	r	p	ρ	p	r	p	ρ	p	r	p
FactCC*	.20	.00	.30	.00	.36	.00	.30	.00	.07	.07	.19	.00
DAE*	.18	.00	.20	.00	.27	.00	.22	.00	.03	.38	**.33**	.00
BS-Fact*	.30	.00	.25	.00	.38	.00	.31	.00	.20	.00	.09	.02
QuestEval	.19	.00	.20	.00	.21	.00	.19	.00	.16	.00	.09	.00
Comp. Avg	.34	.00	.32	.00	.30	.00	.33	.00	.30	.00	.32	.00
Comp. Tuned	**.37**	.00	**.34**	.00	**.42**	.00	**.36**	.00	**.31**	.00	.19	.00

Table 10: Partial correlations of metrics on the Frank test dataset. Composite achieves the highest correlations on the combined and XSum dataset. * indicates results copied from the original work.

as for lookahead in Table 9 shows the result. We experiment with greedy and beam search as the base decoding strategies. For greedy, we experiment with all three decoding strategies for lookahead. For beam search, we are unable to run it with sampling or beam search due to the large computational cost. Interestingly, using beam for lookahead does not provide additional gains. We suspect that this is because exploring the future with more beams cannot guarantee that the base decoding strategy is able to explore them, as it is limited to selecting only the top tokens.

D Composite Metric

As described in Section 3.1, we train the composite metric on FACTCOLLECT and tune it on FRANK (Pagnoni et al., 2021). We use the test set of Pagnoni et al. (2021) for evaluation and the rest for tuning the composite metric. The resulting weights for the metrics are 0.29, -0.29, 1.97, and 0.94 for the FactCC, DAE, BS-Fact, and QuestEval, respectively, and the intercept is -1.91. We additionally compute partial correlations on FRANK, shown in Table 10. We see that the composite is able to further increase the correlations in all settings except for XSum's Spearman correlation. Ablations on the effect of ranking with a single metric in Appendix E.

	RL	BS	BS-Fact	FactCC	DAE ↓	QuestEval
Greedy	26.36	**89.09**	88.93	89.57	75.34	38.39
Beam	27.52	87.56	89.41	87.20	60.21	39.44
BEAM+RANKING	**27.60**	87.62	**89.64**	**91.11**	**47.01**	**41.91**

Table 11: Results for ranking on the WikiHow dataset.

Since FACTCOLLECT only contains annotations on XSum and CNN/DM, we analyze whether the composite metric is robust for another dataset and domain. We use WikiHow (Koupaee and Wang, 2018) and decode using PEGASUS[6] with greedy and beam decoding. The result of applying ranking to the beam output can be seen in Table 11. We see consistent gains in all faithfulness metrics when we apply ranking, showing its robustness of improving faithfulness in another domain.

E Ablations

We present several ablation studies for our proposed faithfulness-aware decoding methods. More ablation studies exploring how lookahead explores the search space can be found in Appendix E.

Lookahead Length. We first present the result of using the lookahead heuristics but with $l = 0$. This means that at each time step, we do not use future heuristics but directly evaluate the

---

[6] We use the checkpoint PEGASUS-WIKIHOW (`https://huggingface.co/google/pegasus-wikihow`).

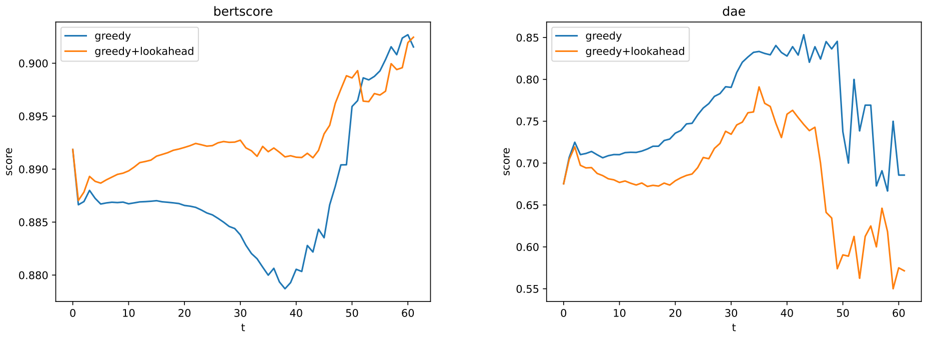


Figure 7: Faithfulness score of the lookahead summaries at each time step. Adding lookahead as the heuristics improves the search space to generate more faithful summaries.

	RL	BS	BS-Fact	FactCC	DAE ↓	QuestEval
			CNN/DM			
Greedy	**30.93**	**88.39**	93.15	69.61	8.15	59.13
$l = 0$	30.71	88.34	93.13	70.41	8.19	58.65
$l =$ full	30.75	88.35	**93.90**	**71.54**	**5.70**	**60.13**
			XSum			
Greedy	36.16	92.03	89.28	23.53	65.35	36.51
$l = 0$	35.73	92.00	89.39	23.43	64.06	36.55
$l =$ full	**36.25**	**92.11**	**89.71**	**24.21**	**60.46**	**37.17**

Table 12: Lookahead ablation with different lengths. $l = 0$ provides the faithfulness heuristic score only on the partially generated summaries while $l =$ full is our lookahead model that evaluates on the full future summary. The faithfulness score calculated on the partial summaries does not provide an effective estimate that improves the faithfulness of the generated summary.

Ranker	RL	BS	BS-Fact	FC	DAE ↓	QE	COMP
				CNN/DM			
First	29.99	88.03	94.20	84.23	3.30	60.03	74.68
BS-Fact	29.81	88.04	**94.64**	84.08	3.04	60.41	75.94
FC	29.98	88.04	94.20	**90.75**	3.00	60.06	76.75
DAE	30.00	88.03	94.20	84.28	**1.92**	60.04	75.11
QE	**30.27**	**88.16**	94.14	82.81	2.83	**63.26**	77.33
Comp.	30.08	88.12	94.31	90.27	**1.92**	62.57	**79.51**
				XSum			
Top	**37.11**	92.12	89.45	22.97	63.49	37.05	8.08
BS-Fact	36.46	**92.14**	**90.15**	22.10	56.33	37.98	12.15
FC	36.98	92.11	89.44	**41.93**	63.47	37.01	13.67
DAE	36.94	92.11	89.54	23.27	**50.82**	37.28	12.24
QE	36.36	92.06	89.61	23.07	60.35	**41.17**	13.20
Comp.	36.42	92.10	89.79	40.11	51.48	40.10	**20.20**

Table 13: Ranking results with different faithfulness metrics. Top is the best summary from beam search, and each subsequent rows represent the ranker using the corresponding faithfulness metric. We abbreviate FactCC as FC, QuestEval as QE, and Composite as Comp.

faithfulness of the already generated partial summaries as the additional score. The result using GREEDY+LOOKAHEAD is shown in Table 12. Compared to greedy decoding, adding the faithfulness score of the current partial summary shows mixed results; the heuristic can only slightly improve BS-Fact, DAE, and QuestEval for XSum. However, we only see substantial gain when the future is taken into account (i.e. $l =$ full). This shows the necessity of using the full summary to achieve the full potential of current faithfulness metrics.

Ranking with Faithfulness Metrics. Next, we present the result for ranking with each respective faithfulness metric. The result is shown in Table 13. Generally, optimizing for one metric will lead to improvement in other faithfulness metrics. While optimizing each of the faithfulness metrics will undoubtedly perform the best when we use that metric for evaluation, the composite metric is able to achieve a similarly good score for **all** faithfulness metric that we are considering.

Evaluating the Search Space. We hypothesize that by incorporating lookahead, we can improve the search space even when a few tokens are generated. To better understand this, we greedily decode the full summary at each time step given the prefix similar to how lookahead works. We then use BS-Fact and DAE to score all generated summaries and analyze the faithfulness score at each time step. Here, we focus on XSum and compare greedy and GREEDY+LOOKAHEAD. The plots of faithfulness scores using the current prefix to generate the full summaries are shown in Figure 7, where we see the benefit of having the lookahead heuristics. For BS-Fact, we see a large gap between the two methods especially when t is between 5 and 50. Though it may be less surprising as this is the faithfulness metric that the lookahead heuristic optimizes on,

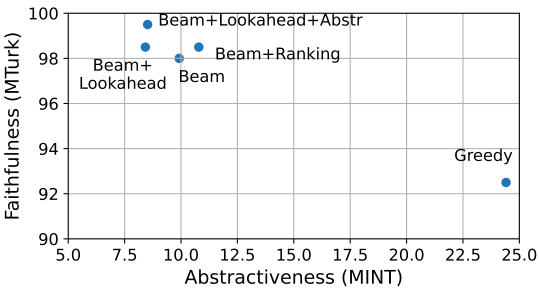


Figure 8: Faithfulness and abstractiveness tradeoff results on the 200 CNN/DM examples used for human annotation. While our proposed methods are less abstractive, the gain in faithfulness is much larger than the decrease in abstractiveness.

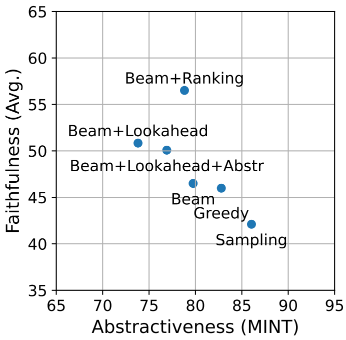


Figure 9: Faithfulness and Abstractiveness tradeoff results on the full examples XSum test set. Faithfulness is calculated by taking the average across all automatic faithfulness metrics.

the heuristic can nevertheless prevent the score to dip, which we see for greedy search between $t = 5$ to $t = 40$. This shows that it is able to lead the model to a more faithful path to prevent straying away from a less faithful path. When we evaluate DAE, we show that optimizing on BS-Fact with lookahead heuristic can consistently improve the score for all lengths.

F Abstractiveness

We first show the same tradeoff result in CNN/DM in Figure 8. BEAM+LOOKAHEAD+ABSTR does achieve a slightly higher MINT score while also improving faithfulness.

We also extend the analysis to the whole test dataset and show the faithfulness score by taking the average of all faithfulness metrics (Avg.). Since DAE is an error rate, we subtract the score from 100 so that a higher score means it is more faithful.

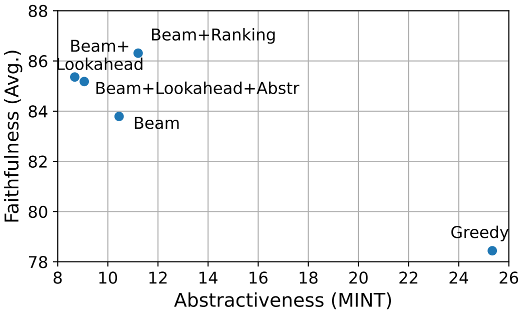


Figure 10: Faithfulness and Abstractiveness tradeoff results on the full examples CNN/DM test set. Faithfulness is calculated by taking the average across all automatic faithfulness metrics.

We do not use the composite metric as the ranking directly optimizes for it.

We can see a similar trend with the average of faithfulness metrics for both datasets in Figure 9 and Figure 10, where the gain in faithfulness outweighs the decrease in abstractiveness. The difference from the result using human faithfulness score is that BEAM+RANKING achieves the highest average score since ranking with composite metric optimizes the faithfulness metrics.

Dynamic Benchmarking of Masked Language Models on Temporal Concept Drift with Multiple Views

Katerina Margatina[◇][*] Shuai Wang[†] Yogarshi Vyas[†]
Neha Anna John[†] Yassine Benajiba[†] Miguel Ballesteros[†]
[◇]University of Sheffield [†]AWS AI Labs
`k.margatina@sheffield.ac.uk`,
`{wshui,yogarshi,nehajohn,benajiy,ballemig}@amazon.com`

Abstract

Temporal concept drift refers to the problem of data changing over time. In NLP, that would entail that *language* (e.g. new expressions, meaning shifts) and *factual knowledge* (e.g. new concepts, updated facts) evolve over time. Focusing on the latter, we benchmark 11 pretrained masked language models (MLMs) on a series of tests designed to evaluate the effect of temporal concept drift, as it is crucial that widely used language models remain up-to-date with the ever-evolving factual updates of the real world. Specifically, we provide a holistic framework that (1) *dynamically* creates temporal test sets of any time granularity (e.g. month, quarter, year) of factual data from Wikidata, (2) constructs fine-grained splits of tests (e.g. updated, new, unchanged facts) to ensure *comprehensive* analysis, and (3) evaluates MLMs in three distinct ways (single-token probing, multi-token generation, MLM scoring). In contrast to prior work, our framework aims to unveil how *robust* an MLM is over time and thus to provide a signal in case it has become *outdated*, by leveraging *multiple views of evaluation*.

1 Introduction

In the real world, what people talk about and how they tend to speak and write changes constantly over time. In Natural Language Processing (NLP), this entails a challenging shift of the textual data distribution that is commonly referred to as *temporal concept drift*. Prior work has identified that pretrained language models (PLMs) tend to become outdated soon after new topics and concepts are emerging (Lazaridou et al., 2021; Dhingra et al., 2022; Agarwal and Nenkova, 2022; Luu et al., 2022), limiting their capability to be robust to newly generated data.

We consider the desiderata of language models' robustness to temporal drift to be twofold. First, LMs should be well adapted to the dynamic use of language, from the *linguistic* perspective. Language changes over time, pronunciations evolve, new words and expressions are borrowed or invented, the meaning of old words drifts, and morphology develops or decays (Blank, 1999; Traugott and Dasher, 2001; Kulkarni et al., 2015). Second, LMs should be aware of the ever-changing reality of the world, from a *factual* perspective. Models' factual knowledge should be up-to-date with new facts and concepts (e.g. Covid-19) to be of use continuously. In this work, we focus on the latter; *the temporal robustness of LMs to facts that change over time*.

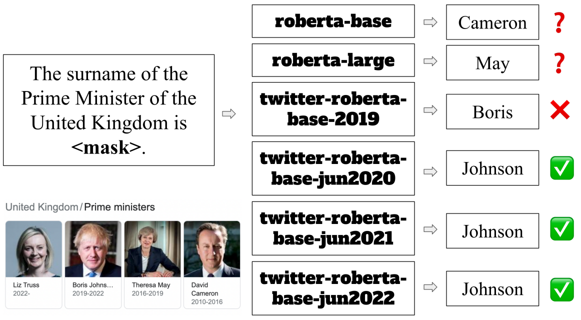


Figure 1: Querying pretrained MLMs on their knowledge about the Prime Minister of the United Kingdom.

In an ideal scenario, we would like to know exactly when the factual knowledge of a model is "expired" so that we could adapt it to the new (or updated) set of facts. In reality, this is a challenging task. A large body of work has focused on the part of (continually) *adapting* an "outdated" model to the new data distribution (Guu et al., 2020; Yogatama et al., 2021; Sun et al., 2020; Biesialska et al., 2020; Jang et al., 2022b; Jin et al., 2022; Chakrabarty et al., 2022). This line of work is parallel to ours, as we focus on the crucial step before adaptation, the evaluation of the model on temporal concept drift: *How can we know if a language model is outdated or not?*

Let us consider the case where we desire a lan-

---

[*] Work done during an internship at AWS AI Labs.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2881–2898
May 2-6, 2023 ©2023 Association for Computational Linguistics

guage model to be up-to-date with the Prime Minister of the United Kingdom (Figure 1).[1] A plausible way to evaluate this is to use the LAMA-probe paradigm (Petroni et al., 2019) and query the LM as a knowledge base (KB). This would mean that we could form the query as "`The surname of the Prime Minister of the United Kingdom is <mask>.`", give it as an input to a (masked) LM and inspect the output token distribution for the `<mask>` token. Figure 1 shows the top prediction for a series of RoBERTa models.[2] We first observe that the most widely used RoBERTa `base` and `large` models are both outdated in terms of factual knowledge, as they predict the names of PMs that served from 2010 until 2019. Next, while the last three models (2020-2022) answer correctly, the 2019 model answers the (correct) *first name* of the PM (`Boris`), not the *surname* (`Johnson`) which is asked for.

This is a handy illustration of the many *challenges* in evaluating MLMs for temporal robustness in the LMs-as-KBs framework. First, this 2019 model would be considered to have made a mistake (as the prediction is different than the gold label and the metric is accuracy), even though the factual knowledge was correct (the name of the PM of the UK). Second, notice that we designed the query to ask for the surname (instead of the name of the PM), as this results in a single mask. The LAMA-probe and related frameworks do not handle *multi-token* queries for MLMs (e.g., `Boris Johnson`). Finally, we mark with a ? the answers of the first two RoBERTa models, because even though their answers are out-of-date for our current evaluation (October 2022), their answers could have been correct in an evaluation setting in the time of the training data (2019). This illustrates the obscurity of the temporal window in which the model is expected to be correct, if the model is not trained with a temporally-aware design (Lazaridou et al., 2021; Dhingra et al., 2022; Loureiro et al., 2022; Jang et al., 2022a).

In this work, we aim to address such limitations and provide a holistic framework for dynamic benchmarking of masked language models on temporal concept drift, with a focus on facts that change over time. Following the propositions of Kiela et al. (2021) and Søgaard et al. (2021) that advocate for a focus on *dynamic* (i.e., test sets should not become saturated) and *targeted* (i.e., use of multiple, independent test sets for realistic performance estimates) benchmarking respectively, and building on prior work (Jiang et al., 2020b; Dhingra et al., 2022; Jang et al., 2022a), we create a large open-source test set that can be dynamically updated over time, containing temporal fine-grained subsets of examples that can be used to query masked language models and evaluate their factual knowledge over time.

Contributions **(1)** We release DynamicTempLAMA, an improved version of the static TempLAMA (Dhingra et al., 2022) test set consisting of Wikidata relations, that is used to evaluate temporal robustness of MLMs. We provide data and code to dynamically keep DynamicTempLAMA up-to-date over time.[3] **(2)** We propose a novel evaluation framework to first create temporal splits of test sets of any granularity (month, quarter, year) and then to further create fine-grained splits of facts that are *unchanged*, *updated*, *new* or *deleted*, aiming to improve comprehensiveness (§3.1). **(3)** We introduce three distinct evaluation views with multiple metrics (§3.3) to ensure comprehensive results and provide analysis of benchmarking a large set open-source temporal RoBERTa models (§3.2).

2 Related Work

Temporal Concept Drift Evaluation of the robustness of language models on temporal concept drift has seen a rising interest in the recent years. Previous work has focused on methods to continually adapt models over time (Hombaiah et al., 2021; Rosin et al., 2022; Lazaridou et al., 2022). Another area of research is evaluation of temporal robustness which has been explored both in the upstream LM pretraining task (Jiang et al., 2020b; Lazaridou et al., 2021; Dhingra et al., 2022; Jang et al., 2022a; Loureiro et al., 2022) and in downstream tasks such as sentiment analysis (Lukes and Søgaard, 2018; Agarwal and Nenkova, 2022), named entity recognition (Rijhwani and Preotiuc-Pietro, 2020; Onoe et al., 2022), question answering (Mavromatis et al., 2021; Liška et al., 2022), and rumor detection (Mu et al., 2023). It has also been studied for model explanations (Zhao et al., 2022) and for text classification in legal, biomedical (Chalkidis and Søgaard,

---

[1] The time of writing of this paper is September 2022.

[2] Except for the RoBERTa `base` and `large` models, we also show the predictions of models trained with Twitter data until 2019, 2020, 2021, and 2022, respectively (Loureiro et al., 2022).

[3] `https://github.com/amazon-science/temporal-robustness`

2022), and social media (Röttger and Pierrehumbert, 2021) domains.

Luu et al. (2022) explore the setting of temporal misalignment (i.e., training and test data drawn from different periods of time) for both upstream and downstream tasks and find that temporal adaptation should not be seen as a substitute for finding temporally aligned labeled data for fine-tuning.

The closest work to ours is TEMPLAMA (Dhingra et al., 2022). However, we differ across four axes: (i) TEMPLAMA is static, while we provide code to dynamically download facts in a fine-grained fashion from any periods of time (not only yearly), (ii) we evaluate the *same* models over time focusing on the evaluation of robustness over time, we do not explore the best adaptation technique to address the problem, (iii) we do not fine-tune the models to adapt them to the domain/format of the test data, and (iv) we address benchmarking of masked LMs (not auto-regressive) including more evaluation techniques. Finally, similar to our motivation, Jang et al. (2022a) recently explored lifelong adaptation and evaluation of temporal concept drift in LMs and introduced TEMPORALWIKI for continual adaptation and TWIKI-PROBES for evaluation. The major difference is that the authors focus on providing corpora to adapt an LM over time, while in our paper we focus on evaluating temporal robustness of LMs. DYNAMICTEMPLAMA is a holistic evaluation framework, while "TWIKI-PROBES are not natural sentences; they are factual phrases synthetically generated from a naive concatenation of Subject, Relation, and Object".

Language Models as Knowledge Bases The cloze-style LM evaluation framework for factual knowledge, LAMA Petroni et al. (2019), follows the setting depicted in Figure 1. A knowledge base relation is transformed into natural language text with a manually created template and then passed as an input to an LM. The framework is based on treating the output distribution for the `mask` token as the retrieved answers to the query (AlKhamissi et al., 2022). The LAMA probe has since been extensively used to evaluate factual knowledge in LMs (Petroni et al., 2020; Talmor et al., 2020; Kassner et al., 2021; Sung et al., 2021; Dhingra et al., 2022; Fierro and Søgaard, 2022), while other works have been exploring its limitations and ways to improve it (Kassner and Schütze, 2020; Haviv et al., 2021; Elazar et al., 2021; Zhong et al., 2021; Qin and Eisner, 2021). A particular challenge in our experimental setting, is the text compatibility between the model (i.e., its pre-training data) and the format of test examples, named as "language mismatch" by Talmor et al. (2020). Dhingra et al. (2022) opts to fine-tune the model under evaluation with part of the test set to adapt it to the format of the task. We argue that this process suffers from many caveats; it is inefficient and impractical to fine-tune a model whose capabilities are under evaluation, it risks optimization stability and overfitting issues due to the small training dataset, and enforces extra biases and errors, especially in the case of temporal robustness evaluation.

3 Dynamic Benchmarking of Temporal Concept Drift

In this section we describe in detail the steps to (re)create DYNAMICTEMPLAMA, our dynamically updated test set with facts from Wikidata (§3.1). We then present the open-source temporal ROBERTA models (TIMELMS) (Loureiro et al., 2022) that we use for benchmarking (§3.2). Finally, we introduce the evaluation framework under which we investigate how well the TimeLMs perform in terms of temporal robustness (§3.3).

The research question that we try to address with our work is: *How can we measure temporal drift robustness of PLMs with an evaluation framework that is*: *unsupervised* (no labeled downstream data), *efficient* (quality test set of facts—no need to run inference on a large corpus to compute perplexity for every token), *dynamic* (test set easily generated per request—can be used to dynamically evaluate new concepts over time), *general* (option to create test sets of any time granularity), and *comprehensive* (battery of targeted test sets that evaluate different LM capabilities and multiple views of evaluation).

3.1 DYNAMIC-TEMPLAMA

We base our implementation on the TEMPLAMA (Dhingra et al., 2022) code, while we make several changes in terms of accessibility (i.e. option to dynamically update the test set), flexibility (i.e. option to adjust the granularity of the temporal splits) and comprehensiveness (i.e. fine-grained splits and multiple evaluation views). We provide a high-level overview of the process to create DYNAMICTEMPLAMA in Figure 2.

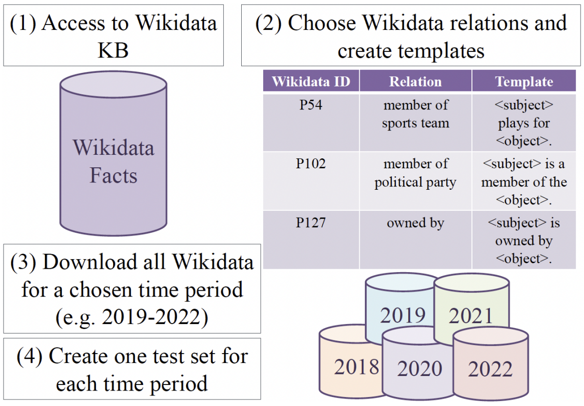
Data Collection We start the process by selecting a set of *relations* collected from the Wikidata KB

WIKIDATA ID	RELATION	TEMPLATE	#FACTS	#EXAMPLES	POSSIBLE SPLIT(S)
P54	member of sports team	`<subject> plays for <object>.`	3772	50558	$\mathcal{D}^{\text{UPDATED}}$
P69	educated at	`<subject> attended <object>.`	232	2420	$\mathcal{D}^{\text{UPDATED}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P6	head of government	`<object> is the head of the government of <subject>.`	578	7815	$\mathcal{D}^{\text{UPDATED}}$
P279	subclass of	`<subject> is a subclass of <object>.`	5	70	$\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UPDATED}}$

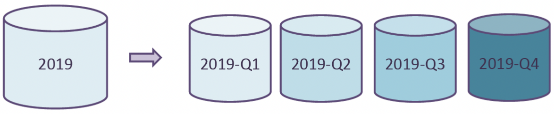
Table 1: Examples of relations and their corresponding templates that we include in DYNAMICTEMPLAMA. #FACTS denote the unique number of facts for each relation, while #EXAMPLES denotes the total number of example we have collected for each relation in the time range between `2019-Q1` and `2022-Q2`. POSSIBLE SPLIT(S) indicate the type of fine-grained split that each relation would potentially belong to.

(Figure 2a).[4] Specifically, we use the 9 relations used in the TEMPLAMA dataset, followed by 7 more that we also decided to collect. We collect all relations from Wikidata in the span of 2019 – 2022. We then manually craft a cloze style query, i.e template, for each relation.Table 1 shows a few examples of relations and templates, along with dataset statistics.[5] We explain the data collection process in detail in Appendix A.1.

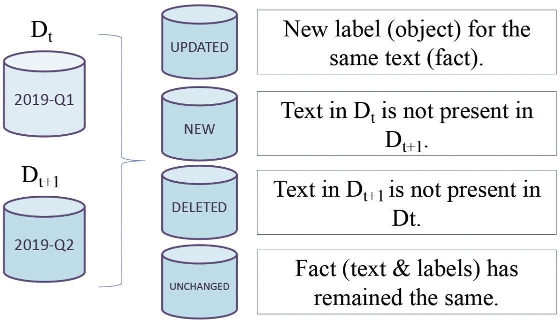
Temporal Splits In this stage, we have a very large collection of facts for which we have temporal information (i.e., that the fact is true) in the time range we investigate (2019 – 2022). In the TEMPLAMA dataset, the facts are divided yearly. However, we would ideally like to benchmark temporal models of any time granularity. Specifically, since we benchmark temporal models that are trained quarterly (§3.2), a yearly split would not be useful to evaluate temporal concept drift of the four models trained on each quarter of a year. Consequently, we divide the large set of collected facts per quarter (Figure 2b), while adding the functionality to our implementation to split the facts in any time granularity (monthly, quarterly, yearly).



(a) Data collection



(b) Temporal Splits



(c) Fine-grained Splits

Figure 2: The process for creating DYNAMICTEMPLAMA. We first collect data from Wikidata (a), we then divide it to quarterly temporal splits (b) and finally we create more targeted fine-grained sets (c).

Fine-grained Splits For a given time range, from timestep t to $t+1$ (e.g. `2019-Q1`→`2019-Q2`), we further create comprehensive test sets that contain examples with *unchanged*, *updated*, *new* or *deleted* facts, denoted by $\mathcal{D}^{\text{UNCHANGED}}_{t+1}$, $\mathcal{D}^{\text{UPDATED}}_{t+1}$, $\mathcal{D}^{\text{NEW}}_{t+1}$ and $\mathcal{D}^{\text{DELETED}}_{t+1}$ respectively (Figure 2c). We create these splits to be able to measure different capabilities of the MLM in terms of robustness to temporal concept drift. The motivation for this stems from limitations of prior work (Dhingra et al., 2022) to shed light into what kind of data each temporal test set contains. For instance, we pose questions like *How many facts were updated from timestep* $t \rightarrow t+1$*? How many facts remained unchanged? What was the change? The object or the subject? Are there new facts in timestep* $t+1$ *that were not present before?* We argue that it is

---

[4] All possible relations from Wikidata can be found here `https://www.wikidata.org/wiki/Wikidata:List_of_properties`.

[5] Details on all relations and templates of DYNAMICTEMPLAMA can be found in Tables 6 & 7 in the Appendix A.1.

essential to distinguish between these sub-tests, so that each split can target specific capabilities of the LM. First, we can use $\mathcal{D}_{t+1}^{\text{UNCHANGED}}$ to evaluate knowledge preservation (i.e. how well a model can preserve knowledge over time). Second, we can use $\mathcal{D}_{t+1}^{\text{UPDATED}}$, $\mathcal{D}_{t+1}^{\text{NEW}}$ and $\mathcal{D}_{t+1}^{\text{DELETED}}$ to measure adaptation (i.e. how well a model adapts to new information/facts). Finally, we can measure overall temporal robustness by evaluating a temporal model from timestep t on $\mathcal{D}_{t+1}^{\text{UPDATED}}$ and $\mathcal{D}_{t+1}^{\text{NEW}}$ in timesteps for $t \in [t+1, t+2, ...)$. We believe that this framework is particularly useful for insightful evaluation of methods that aim to adapt language models over time (Guu et al., 2020; Yogatama et al., 2021; Sun et al., 2020; Biesialska et al., 2020; Jang et al., 2022b; Jin et al., 2022; Chakrabarty et al., 2022).

3.2 Temporal Models

In contrast with prior work that uses private, in-house models for temporal robustness evaluation that are not accessible by the community (Lazaridou et al., 2021; Dhingra et al., 2022), we instead benchmark a series of open-source temporal models. Despite our aim for transparency, energy efficiency (Strubell et al., 2019) and reproducibility, we also believe that the *dynamic* nature of the task at hand requires *accessibility* to past, present and future models, to ensure that the findings of evaluation studies in temporal concept drift are meaningful, trustworthy and serve their purpose in evaluating models in a ever-evolving world. Under this assumption, we believe that studies on temporal robustness should ideally build on each other, so that we can have a holistic view as to how these models truly evolve over time.

To this end, we use the Diachronic Language Models (TIMELMS) (Loureiro et al., 2022) that are publicly available in the `HuggingFace` hub (Wolf et al., 2019).[6] TIMELMS are ROBERTAmodels (Liu et al., 2019) trained *quarterly* on Twitter data. All models are initialised from the original `roberta-base` model checkpoint and are later trained using data from the previous quarters and the new temporal data from the new time period. For instance, the first model (`2019-Q4`) was trained with data sampled from Twitter until December 2019, while the second model (`2020-Q1`) was trained on the concatenation of all the data used to train `2019-Q4` and temporally-aligned data sampled from the first quarter of 2020. There are 11 TIMELMS in total, from `2019-Q4` until `2022-Q2`.

Finally, we would like to draw attention to two specific points. First, all TIMELMS are trained using the same ROBERTA (base) tokenizer and thus have the same vocabulary. This is crucial when evaluating models in a Cloze-style format, like the LAMA-probe, in order to evaluate fair comparison among the models. Second, Loureiro et al. (2022) aim to continue training and releasing TIMELMS every quarter, which is a very important and promising initiative to help with the *dynamic* evaluation of LMs in temporal concept drift in the future.

3.3 Temporal Concept Drift Evaluation

Single-token probing Our first evaluation type is single-token probing, which was introduced in the seminal LAMA-probe work of Petroni et al. (2019). The idea is simple and follows the fill-in-the-blank format. Specifically, we convert each relation using its template to natural language text (see Figure 2(a)) replacing the `<object>` with the mask token (i.e., `<mask>` for ROBERTA). Then, as shown in Figure 1, we give the prompt as an input to the MLM and obtain a probability distribution over the vocabulary for the `<mask>` token. We use the metrics from Petroni et al. (2019), that are `Accuracy`, Mean Reciprocal Rank (`MRR`) and Precision at k (`P@k`).[7] Note that a crucial limitation of this approach is that it considers only facts with single-token objects. This results in trimming down the test sets by 95%, while limiting the actual value of the test (as most facts and concepts contain multiple words).

Multi-token generation We aim to address this limitation and include multi-token objects to our evaluation framework. It is important to note that we are benchmarking *masked* language models instead of autoregressive left-to-right language models like Dhingra et al. (2022). This is crucial because the latter, decoder-based family of models, can be used off-the-shelf to generate multiple tokens. In contrast, MLMs are trained with 15% of their inputs masked and optimized to predict *only the masked tokens*. We therefore use the formulation introduced by Wang and Cho (2019), that is essentially a decoding-based strategy for MLMs based on Gibbs sampling. Specifically, we consider

---

[6] `https://huggingface.co/cardiffnlp`

[7] `P@k`$=1$, if the gold label is in the top-k predictions of the model, therefore `P@1` corresponds to `Accuracy`.

the setting that we do not know a priori the correct number of masks for each label. Instead, we enumerate from a single mask up to M masks, i.e., $m = 1, ..., M$. Following Jiang et al. (2020a), we choose $M = 5$, as all our facts are in the English language. When $m > 1$, we add m consecutive masks to the input and we pass the input to the model m times, when each time we sequentially sample each mask from left to right. At each iteration we replace the mask with the corresponding token prediction of the previous iteration. This way, we can extend the LAMA probe to include multi-token labels in our test set. The setting is entirely different than the single-token approach, as here we have m predictions from the model with an increasing number of tokens, while the correct label can consist of any number of tokens in the range of $1, ..., M$. Another difference here is the evaluation metrics. Because we converted the task to text generation, we borrow generation metrics such as `ROUGE` (Lin, 2004), while also including standard metrics like `F`$_1$`-macro`. Finally, we also include as a metric `BERT-score` (Zhang* et al., 2020) as an additional informative metric from the perspective of contextual semantics. In effect, we evaluate factual knowledge over time of MLMs, where facts include *multiple correct answers* and each answer consists of *multiple tokens*. We consider a prediction correct if the model correctly predicts *any* of the acceptable answers.

MLM scoring Finally, as a third lens of evaluation we use the MLM scoring framework of Salazar et al. (2020). Contrary to the previous approaches, MLM scoring aims to *measure* the probability of the correct answer (i.e., of the masks), instead of *generating* the most probable answer. More specifically, we evaluate MLMs out of the box via their *pseudo-log-likelihood scores* (PLLs), which are computed by masking tokens one by one. PLLs have been widely used to measure the equivalence of perplexity (of autoregressive language models) for MLMs in unlabelled data (Lazaridou et al., 2021). Still, computing PLLs for large corpora is a very costly process in terms of time and resources (Loureiro et al., 2022). Instead, we propose to combine the LAMA and MLM scoring frameworks to create an efficient and targeted evaluation framework for temporal factual knowledge.

3.4 Dataset Analysis

We consider different subsets of the DYNAMICTEMPLAMA test sets for the three different evaluation settings (§3.3). For the multi-token and MLM scoring settings, we keep the full dataset, for single-token we first tokenize the labels and keep only the test examples that have at least one label with a single token. This results in a very aggressive filtering of the dataset. Specifically, each quarterly temporal split consists of 8500 test examples on average for the multi-token setting, but for the single-token this results in only 450 examples, marking a loss of 95% of the data.[8] Additionally, the distribution of the fine-grained splits is of great interest, as it will shape the interpretation of the results and the general challenges of the evaluation framework. $\mathcal{D}^{\text{UPDATED}}$ and $\mathcal{D}^{\text{UNCHANGED}}$ (i.e., the splits of the most interest) constitute around 96% and 0.3%, respectively, of the total examples for the single-token evaluation, and 95% and 1.8% for the multi-token. This is arguably a very skewed distribution, showing the importance of our work in diving the temporal splits into further fine-grained splits. This is essential, because we would have different expectations for a model trained on timestep t while tested on data from both t and $t-1$; for *unchanged* facts it would be desirable to keep equal performance in both sets (i.e., knowledge preservation §4.2), while for updated facts we would like to see improved performance in timestep t (i.e., adaptation §4.3).

4 Results

4.1 Temporal robustness

We first evaluate *temporal robustness* of the 11 TIMELMS, defined as the *overall* performance over time (§3.1). Figure 3 shows the average performance in *all* temporal and fine-grained splits in the time range from `2019-Q4` to `2022-Q2` for two types of evaluation, single-token probing and multi-token generation. For the former evaluation type, (Fig. 3a), all models perform similarly for all metrics. However, when we evaluate multi-token generation the models gradually improve over time. (Fig. 3b). This difference shows the importance of considering *multiple views* and evaluations for the same LM capability (i.e., temporal robustness).

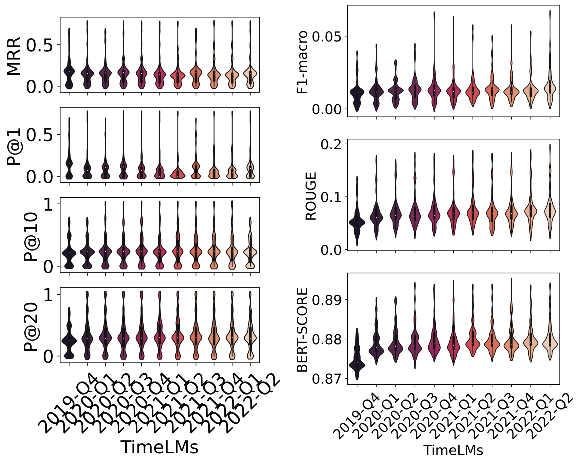
We attribute the similar single-token performance to the fact that these temporal datasets con-

---

[8] Table 5 in the Appendix shows all the statistics in detail.

MODELS	**TEMPORAL SPLITS**												
	`2019-Q2`	`2019-Q3`	`2019-Q4`	`2020-Q1`	`2020-Q2`	`2020-Q3`	`2020-Q4`	`2021-Q1`	`2021-Q2`	`2021-Q3`	`2021-Q4`	`2022-Q1`	`2022-Q2`
`2019-Q4`	34.88	33.96	34.44	34.93	34.76	34.73	34.02	34.18	34.70	34.34	34.92	35.46	35.31
`2020-Q1`	24.47	24.01	24.45	24.67	24.59	24.44	23.98	23.94	24.25	23.96	24.20	24.5	24.42
`2020-Q2`	22.94	22.29	22.92	23.24	23.23	23.12	22.57	22.55	22.90	22.59	22.91	23.23	23.11
`2020-Q3`	22.39	21.87	22.22	22.60	22.52	22.42	21.99	22.00	22.29	21.92	22.18	22.42	22.30
`2020-Q4`	25.56	25.28	25.68	25.96	25.89	25.79	25.51	25.44	25.71	25.50	25.69	25.97	25.72
`2021-Q1`	25.76	25.28	25.91	26.18	26.14	26.18	25.75	25.63	25.99	25.77	26.01	26.32	26.02
`2021-Q2`	23.75	23.47	23.94	24.10	24.10	24.12	23.63	23.60	24.05	23.75	24.12	24.37	24.16
`2021-Q3`	22.95	22.61	23.00	23.14	23.12	23.16	22.84	22.77	23.00	22.82	23.03	23.30	23.06
`2021-Q4`	23.37	23.01	23.41	23.59	23.55	23.68	23.37	23.27	23.60	23.40	23.58	23.76	23.61
`2022-Q1`	24.25	23.83	24.42	24.56	24.57	24.68	24.40	24.26	24.52	24.35	24.51	24.71	24.58
`2022-Q2`	**21.48**	**20.95**	**21.42**	**21.59**	**21.57**	**21.61**	**21.25**	**21.12**	**21.44**	**21.13**	**21.31**	**21.49**	**21.39**

Table 2: MLM scoring (median pseudo-log-likelihood scores) averaged for each temporal split.



(a) Single-token

(b) Multi-token

Figure 3: *Overall* performance over time ($2019 - 2022$) for both single and multi-token evaluation. X-axis corresponds to the TIMELMS and the Y-axis to different metrics depending on the type of the evaluation.

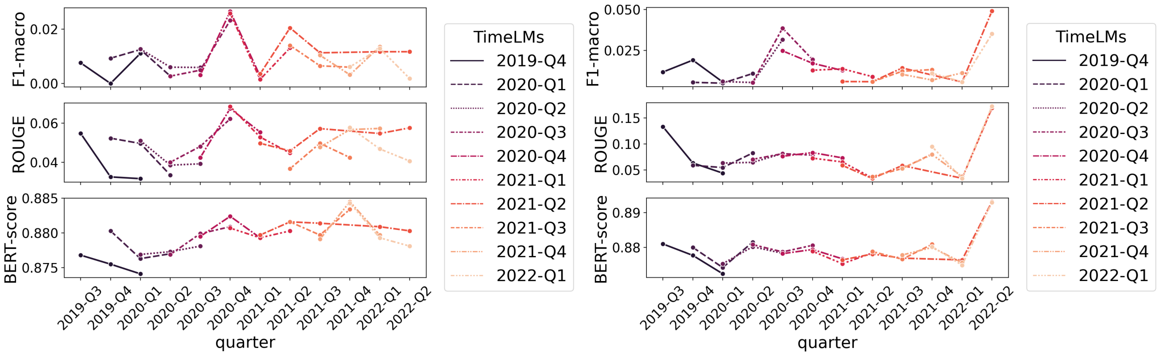
tain almost exclusively *unchanged* facts (§3.4). It is therefore a positive outcome to observe that TIMELMS can preserve acquired knowledge (§4.2). The findings for overall multi-token evaluation corroborate the intuition that more recent models, that are trained with temporal data of the entire range, should perform better than "past" (e.g. `2020`) models that have not seen "future" data (e.g. `2022`) during training. We also provide the overall results with MLM scoring in Table 2. We also observe that the last model performs best across all temporal splits, showing the effectiveness of adaptation with more recent unlabelled data (§3.2). Even though we observe that this pattern holds for most temporal splits (i.e., scores improving for each column ↓), the `2020-Q4` and `2021-Q1` TIMELMS produce worse PLL scores than their previous or later versions. This is more evident in the overall density plot in Figure 5. This finding entails that either the distribution shift in these quarters was a lot stronger than the other temporal periods, or the training of these particular models was not as successful as it would have been expected.

4.2 Knowledge preservation

We use the $\mathcal{D}^{\text{UNCHANGED}}$ split to evaluate the capability of MLMs to preserve knowledge over time. Figure 6 shows that for both single and multi-token evaluation all TIMELMS demonstrate similar performance over time, showing strong knowledge preserving skills. Surprisingly, different metrics show different patterns among the models for a single split. While in general we should not compare the performance of the single model over time (as the test sets are different), the comparision is valid in this case because the splits contain unchanged facts, and hence most temporal test sets are almost identical. All plots are shown in Figure 7 in the Appendix.

4.3 Adaptation to emerging & evolving concepts

Finally, we use the $\mathcal{D}^{\text{NEW}}$ and $\mathcal{D}^{\text{UPDATED}}$ splits for evaluation of emerging and evolving concepts, respectively. Here to ensure fair comparison, we evaluate the TIMELMS for a specific time window; for each model trained on timestep t, we keep the test sets from $t - 1$, t and $t + 1$. We observe in Figure 4 that in these cases the results vary among the models. There is not a very clear pattern as before, so case-by-case examination would be required. Still, a common pattern for the UPDATED split is that the middle set tends to have the highest performance ($\wedge$ shape). This means that models manage to effectively adapt to the updated facts of that timestep (t), but on the next timestep ($t + 1$)



(a) UPDATED split.

(b) NEW split.

Figure 4: Multi-token evaluation for evolving and emerging facts.

EXAMPLE INPUT	GROUND TRUTH LABELS	#TOKENS	#ANSWERS	SPLIT
1 `Alex Morgan plays for _X_.`	`United States women's national soccer team`	7	2	`2021-Q4`
	`Orlando Pride`	2		
2 `Cristiano Ronaldo plays for _X_.`	`Juventus F.C.`	5	1	`2021-Q2`
	`Juventus F.C., Manchester United F.C.`	5, 6	2	`2021-Q3`
	`Manchester United F.C.`	6	1	`2021-Q4`
2 `_X_ is the head of the government of Italy.`	`Giuseppe Conte`	5	1	`2020-Q4`
	`Giuseppe Conte, Mario Draghi`	5, 3	2	`2021-Q1`
	`Mario Draghi`	3	1	`2021-Q2`

Table 3: Qualitative analysis of certain examples in DYNAMICTEMPLAMA.

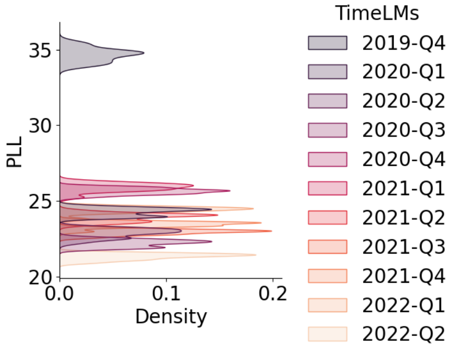


Figure 5: *Overall* PLL distributions for TIMELMS.

they underpeform as they are unaware of the factual changes, thus requiring adaptation. We provide all plots in the Appendix, including the DELETED split, which is more difficult to interpret intuitively (i.e., why are some facts deleted from Wikidata after a certain point?).

5 Qualitative Analysis

Table 3 provides some examples from the DYNAMICTEMPLAMA test set that can help us further interpret our results and inspect existing challenges. We first observe that all examples have multi-token labels (i.e., objects from the `Subject-relation-object` format) and are in effect discarded in the single-token evaluation setup, making the inclusion of multiple views essential for this task.

More specifically, in 1, we observe that one label (`United States women's national soccer team`) has more than $M = 5$ tokens. It is therefore excluded even from the multi-token the test set, leaving MLM scoring to be the only method that could evaluate it. Interestingly, we manually tested the `2021-Q4` temporal model and found that it produces 1.6 and 307.3 average PLL scores for the two options respectively, making the disregarded label a far more confident prediction.

In the second and third example, we observe how the correct answer for the query changes over time, making the granularity of the evaluation (i.e., yearly, quarterly, monthly) an important factor in the correct assessment of the model's temporal factual knowledge. For instance, for the example 3, we can carefully inspect how the predictions of the models change for facts that change over time (Table 4). However, even though PLL scores can follow intuitive temporal patterns (i.e., the PLL value can increase or decrease according to the point in time that the fact has changed), comparison between scores is not always helpful (i.e., word frequency can obscure factual knowledge) leaving

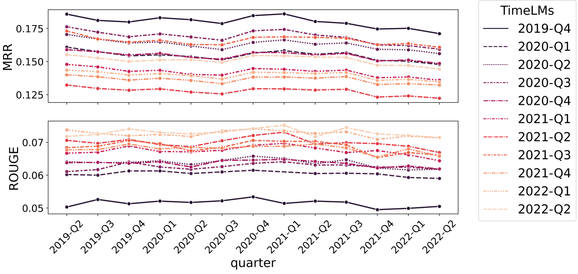


Figure 6: Single and multi-token evaluation for the UNCHANGED split.

TIMELMS	Guiseppe Conte	Mario Draghi
2020-Q4	3.8	33.3
2021-Q1	3.5	22.7
2021-Q2	3.5	25.7
2021-Q3	3.8	23.8

Table 4: PLL scores for Example 2 from Table 3.

room for improving the LAMA formulation.

6 Conclusion & Future Work

We addressed MLMs' robustness on temporal concept drift and introduced DYNAMICTEMPLAMA: a dataset for dynamic benchmarking of factual knowledge in temporal, fine-grained splits, from `2019-Q4` to `2022-Q2` that contain facts over time. We release our codebase to *dynamically* update the current test set over time and the option to extend it with custom (i) templates, (ii) relations from Wikidata, (iii) any period of time (years) and (iv) granularity of time (month/quarter/year). We include *multiple views of evaluation*, showing that it is essential in order to properly interpret the results of our benchmarking study of 11 temporal ROBERTA models. We consider experimentation with improving MLM decoding and addressing "domain mismatch" as open areas of research for future work. Our code can be found at `https://github.com/amazon-science/temporal-robustness`.

Acknowledgements

We would like to thank the anonymous reviewers and the AWS AI team for their valuable feedback on the paper. We also thank Robert Vacareanu and Karthikeyan K for their help with running experiments.

Limitations

Lower bound estimate A very common issue with the LAMA probe evaluation framework (Petroni et al., 2019) is that it constitutes a lower bound estimate for its performance on factual knowledge retrieval. Specifically, if a model performs well, one can infer that it has the tested reasoning skill. However, failure does not entail that the reasoning skill is missing, as it is possible that there is a problem with the lexical-syntactic construction we picked (Talmor et al., 2020). Any given prompt only provides a lower bound estimate of the knowledge contained in an LM (Jiang et al., 2020b).

Domain mismatch Despite the advantages of zero-shot evaluation, performance of a model might be adversely affected by mismatches between the language the pre-trained LM was trained on and the language of the examples in our tasks (Jiang et al., 2020b). It is quite possible that a fact that the LM does know cannot be retrieved due to the prompts not being effective queries for the fact (Jiang et al., 2020b). Prior work proposes to fine-tune the model with a small set of examples taken from the test set (and removed of course) in order to address the incompatibility problem or 'language mismatch' (Talmor et al., 2020; Dhingra et al., 2022). We argue that this process suffers for multiple limitations, such as that it not practical for a fast evaluation of the capabilities of a PLM at hand and it faces optimization stability issues due to the small training dataset, inter alia. The major limitation, however, is that such fine-tuning enforces extra biases and errors, especially in the case of temporal robustness evaluation.

MLM decoding (multi-token labels) In this work we tried to address the problem of *decoding from masked language models*, by incorporating two distinct approaches to the evaluation framework; multi-token generation with MLMs (Wang and Cho, 2019) and MLM scoring (Salazar et al., 2020). Still, we observe that both methods provide results that are hard to interpret (§5), leaving the problems of (i) decoding or generating multiple tokens from MLMs and (ii) evaluation of factual knowledge in LMs as open areas of research.

Manual Templates For LAMA-style probing (Petroni et al., 2019), prior work creates the templates *manually*. This is a limitation both in terms of scale (i.e., generalization to many different kinds of inputs) and consistency (i.e., how do models perform with minimal changes to their inputs?). LMs do not reason in an abstract manner and are context-dependent (Talmor et al., 2020). It is therefore essential to address this problem and include functionalities to incorporate a set of diverse templates for each evaluation setup.

English Twitter MLMs Finally, our dataset, DYNAMICTEMPLAMA, following prior work (Dhingra et al., 2022), collects and evaluates facts from the Wikidata in the *English* language alone, and benchmarks RoBERTa language models trained in English Twitter data. We understand that this is a limitation and further data collection and experimentation in more languages would be strongly encouraged.

References

Oshin Agarwal and Ani Nenkova. 2022. Temporal effects on pre-trained models for language processing tasks. *Transactions of the Association for Computational Linguistics*, 10:904–921.

Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. 2022. A review on language models as knowledge bases.

Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. 2020. Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6523–6541, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Andreas Blank. 1999. Why do new meanings occur? a cognitive typology of the motivations for lexical semantic change.

Tuhin Chakrabarty, Thomas Scialom, and Smaranda Muresan. 2022. Fine-tuned language models can be continual learners. In *Challenges & Perspectives in Creating Large Language Models*.

Ilias Chalkidis and Anders Søgaard. 2022. Improved multi-label classification under temporal concept drift: Rethinking group-robust algorithms in a label-wise setting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Constanza Fierro and Anders Søgaard. 2022. Factual consistency of multilingual pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3046–3052, Dublin, Ireland. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623, Online. Association for Computational Linguistics.

Spurthi Amba Hombaiah, Tao Chen, Mingyang Zhang, Michael Bendersky, and Marc-Alexander Najork. 2021. Dynamic language models for continuously evolving content. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. 2022a. Temporalwiki: A lifelong benchmark for training and evaluating ever-evolving language models.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun KIM, Stanley Jungkyu Choi, and Minjoon Seo. 2022b. Towards continual knowledge learning of language models. In *International Conference on Learning Representations*.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020a. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5943–5959, Online. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Xisen Jin, Dejiao Zhang, Henghui Zhu, Wei Xiao, Shang-Wen Li, Xiaokai Wei, Andrew Arnold, and Xiang Ren. 2022. Lifelong pretraining: Continually adapting language models to emerging corpora. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 1–16, virtual+Dublin. Association for Computational Linguistics.

Nora Kassner, Philipp Dufter, and Hinrich Schütze. 2021. Multilingual LAMA: Investigating knowledge in multilingual pretrained language models. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 625–635, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomáš Kočiský, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models. In *Advances in Neural Information Processing Systems*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d'Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon, Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-collados. 2022. TimeLMs: Diachronic language models from Twitter. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 251–260, Dublin, Ireland. Association for Computational Linguistics.

Jan Lukes and Anders Søgaard. 2018. Sentiment analysis under temporal shift. In *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 65–71.

Kelvin Luu, Daniel Khashabi, Suchin Gururangan, Karishma Mandyam, and Noah A. Smith. 2022. Time waits for no one! analysis and challenges of temporal misalignment. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5944–5958, Seattle, United States. Association for Computational Linguistics.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Soji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2021. Tempoqr: Temporal question reasoning over knowledge graphs.

Yida Mu, Kalina Bontcheva, and Nikolaos Aletras. 2023. It's about time: Rethinking evaluation on rumor detection benchmarks using chronological splits. In *Findings of the Conference of the European Chapter of the Association for Computational Linguistics*.

Yasumasa Onoe, Michael Zhang, Eunsol Choi, and Greg Durrett. 2022. Entity cloze by date: What LMs know about unseen entities. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 693–702, Seattle, United States. Association for Computational Linguistics.

Fabio Petroni, Patrick S. H. Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *CoRR*, abs/2005.04611.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Shruti Rijhwani and Daniel Preotiuc-Pietro. 2020. Temporally-informed analysis of named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7605–7617, Online. Association for Computational Linguistics.

Guy D. Rosin, Ido Guy, and Kira Radinsky. 2022. Time masking for temporal language models. *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*.

Paul Röttger and Janet Pierrehumbert. 2021. Temporal adaptation of BERT and performance on downstream document classification: Insights from social media. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2400–2412, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. *CoRR*, abs/1906.02243.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. {LAMAL}: {LA}nguage modeling is all you need for lifelong language learning. In *International Conference on Learning Representations*.

Mujeen Sung, Jinhyuk Lee, Sean Yi, Minji Jeon, Sungdong Kim, and Jaewoo Kang. 2021. Can language models be biomedical knowledge bases? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4723–4734, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*.

Elizabeth Closs Traugott and Richard B. Dasher. 2001. *Regularity in Semantic Change*. Cambridge Studies in Linguistics. Cambridge University Press.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Zhixue Zhao, George Chrysostomou, Kalina Bontcheva, and Nikolaos Aletras. 2022. On the impact of temporal concept drift on model explanations. In *Findings of the Conference on Empirical Methods in Natural Language Processing*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

TEMPORAL SPLIT	UNCHANGED	UPDATED	DELETED	NEW	TOTAL	%UNCHANGED	%UPDATED	%LOST
2019-Q2	479|8523	1|165	7|124	9|121	496|8933	96.6|95.4%	0.2|1.8%	94.4%
2019-Q3	451|8154	3|248	36|430	5|205	495|9037	91.1|90.2%	0.6|2.7%	94.5%
2019-Q4	454|8271	0|151	3|140	12|120	469|8682	96.8|95.3%	0.0|1.7%	94.6%
2020-Q1	456|8243	3|296	9|126	15|273	483|8938	94.4|92.2%	0.6|3.3%	94.6%
2020-Q2	470|8451	0|92	2|95.	2|59	474|8697	99.2|97.2%	0.0|1.1%	94.5%
2020-Q3	446|8254	2|179	26|238	10|133	484|8804	92.1|93.8%	0.4|2.0%	94.5%
2020-Q4	452|8298	2|124	4|111	5|97	463|8630	97.6|96.2%	0.4|1.4%	94.6%
2021-Q1	453|8238	1|269	4|131	14|215	472|8853	96.0|93.1%	0.2|3.0%	94.7%
2021-Q2	460|8344	2|90	7|128	5|76	474|8638	97.0|96.6%	0.4|1.0%	94.5%
2021-Q3	445|8164	2|164	19|220	2|99	468|8647	95.1|94.4%	0.4|1.9%	94.6%
2021-Q4	443|8213	1|128	4|82	5|90	453|8513	97.8|96.5%	0.2|1.5%	94.7%
2022-Q1	442|8189	1|111	7|117	6|126	456|8543	96.9|95.9%	0.2|1.3%	94.7%
2022-Q2	446|8287	0|56	2|40	2|34	450|8417	99.1|98.5%	0.0|0.7%	94.7%

Table 5: Total number of examples for each temporal and fine-grained split in DYNAMICTEMPLAMA. We show both the *single-token* and the *multi-token* datasets (up to $M = 5$ tokens). Cell scheme to be read *single* | *multi*. %UNCHANGED and %UPDATED show the percentage of the total examples that are part of the UNCHANGED and UPDATED set respectively. %LOST shows the percentage of examples we lose when we filter out the dataset for the *single-token* evaluation setting.

A Appendix

A.1 Data Collection for DYNAMICTEMPLAMA

Following Dhingra et al. (2022), we identify all facts in the Wikidata snapshot, which have either a start or an end date after 2010 and whose subjects and objects are both entities with Wikipedia pages.1 Among these 482K facts, we identify subject and relation pairs which have multiple objects at different times and select 16 relations with the most such subjects. Then, for these relations we manually write template cloze queries (i.e., templates) and populate them with the 1000 most frequent subjects per relation. For each subject and each relation we gather all the objects with their associated time interval and construct a separate query for each year in that interval. When intervals for the object entities overlap, we add all of them to the list of correct answers. The query and the corresponding year form the input texts and the temporal information t, while the object entity is the target that we want to predict (i.e., gold label). In contrast to Dhingra et al. (2022), we do extra temporal divisions. Specifically, we get each yearly split and divide it further in quarterly splits (§3.1, Figure 2b), following the same algorithm.

A.2 Full Results

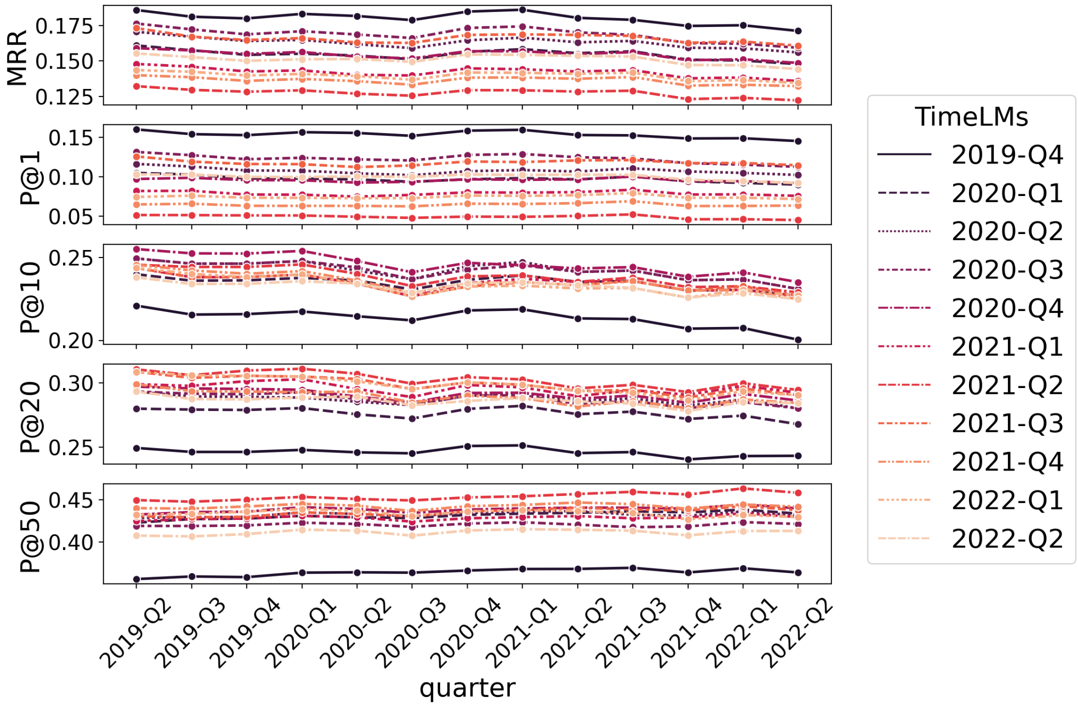
We provide the full results with all metrics for the UNCHANGED split in Figure 7, and the UPDATED, NEW and DELETED splits for multi-token generation in Figure 9.

Wikidata ID	**Relation**	**Template**	**#Facts**	**#Examples**	**Possible Split(s)**
P54	member of sports team	`<subject> plays for <object>.`	3772	50558	$\mathcal{D}^{\text{UPDATED}}$
P39	position held	`<subject> holds the position of <object>.`	2961	34835	$\mathcal{D}^{\text{UPDATED}}$
P108	employer	`<subject> works for <object>.`	1544	20531	$\mathcal{D}^{\text{UPDATED}}$
P102	political party	`<subject> is a member of the <object>.`	1068	14232	$\mathcal{D}^{\text{UPDATED}}$
P286	head coach	`<object> is the head coach of <subject>.`	987	11935	$\mathcal{D}^{\text{UPDATED}}$
P69	educated at	`<subject> attended <object>.`	232	2420	$\mathcal{D}^{\text{UPDATED}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P488	chairperson	`<object> is the chair of <subject>.`	629	8468	$\mathcal{D}^{\text{UPDATED}}$
P6	head of government	`<object> is the head of the government of <subject>.`	578	7815	$\mathcal{D}^{\text{UPDATED}}$
P279	subclass of	`<subject> is a subclass of <object>.`	5	70	$\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UPDATED}}$
P127	owned by	`<subject> is owned by <object>.`	394	5326	$\mathcal{D}^{\text{UPDATED}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P1001	legal term	`<subject> is a legal term in <object>.`	37	423	$\mathcal{D}^{\text{UNCHANGED}}$
P106	profession	`<subject> is a <object> by profession.`	83	1090	$\mathcal{D}^{\text{UPDATED}}$, $\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P27	citizen	`<subject> is <object> citizen.`	147	1983	$\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P176	produced by	`<subject> is produced by <object>.`	24	276	$\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P138	named after	`<subject> is named after <object>.`	73	1009	$\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UNCHANGED}}$
P937	work location	`<subject> used to work in <object>.`	38	507	$\mathcal{D}^{\text{NEW}}$, $\mathcal{D}^{\text{UNCHANGED}}$

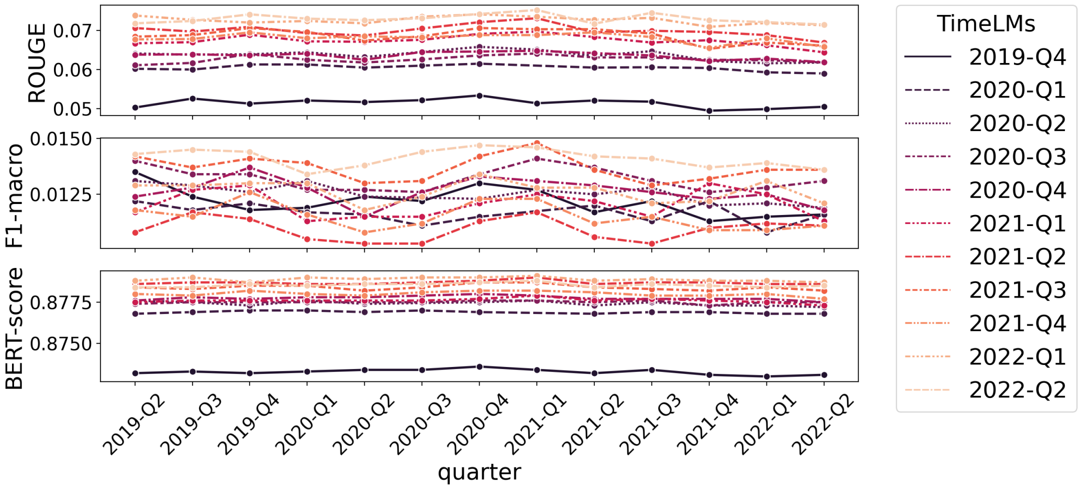
Table 6: The list of templates we used for each relation in the DYNAMICTEMPLAMA dataset.

WIKIDATA ID	RELATION	INPUT	LABELS	SPLIT
P54	member of sports team	`Cristiano Ronaldo plays for _X_.`	`Juventus F.C., Manchester United F.C.`	`2021-Q3`
P39	position held	`Martina Anderson holds the position of _X_.`	`member of the European Parliament`	`2019-Q4`
P108	employer	`George van Kooten works for _X_.`	`University of Cambridge`	`2022-Q2`
P102	political party	`Elena Kountoura is a member of the _X_.`	`Independent Greeks, SYRIZA`	`2019-Q2`
P286	head coach	`_X_ is the head coach of New York Red Bulls.`	`Gerhard Struber`	`2020-Q4`
P69	educated at	`Sarafina Nance attended _X_.`	`Tufts University, University of California, Berkeley`	`2020-Q2`
P488	chairperson	`_X_ is the chair of Lloyds Banking Group.`	`Lord Blackwell`	`2022-Q2`
P6	head of government	`_X_ is the head of the government of United Kingdom.`	`Theresa May, Boris Johnson`	`2019-Q3`
P279	subclass of	`Mercedes-Benz A-Class is a subclass of _X_.`	`compact car`	`2022-Q2`
P127	owned by	`DeepMind is owned by _X_.`	`Alphabet Inc.`	`2021-Q4`
P1001	legal term	`Commonwealth of Independent States Free Trade Area is a legal term in _X_.`	`'Ukraine', 'Russia', 'Belarus', 'Armenia', 'Kazakhstan', 'Moldova', 'Kyrgyzstan', 'Uzbekistan', 'Tajikistan'`	`2022-Q2`
P106	profession	`Penny James is a _X_ by profession.`	`chief executive officer`	`2019-Q3`
P27	citizen	`Yulia Putintseva is _X_ citizen.`	`Kazakhstan 2022-Q1`	
P176	produced by	`Land Rover Discovery series is produced by _X_.`	`Jaguar Land Rover`	`2022-Q2`
P138	named after	`Bayes Business School is named after _X_.`	`Thomas Bayes`	`2021-Q3`
P937	work location	`Eliza Vozemberg used to work in _X_.`	`Strasbourg, City of Brussels`	`2022-Q2`

Table 7: Examples of DYNAMICTEMPLAMA for each relation.

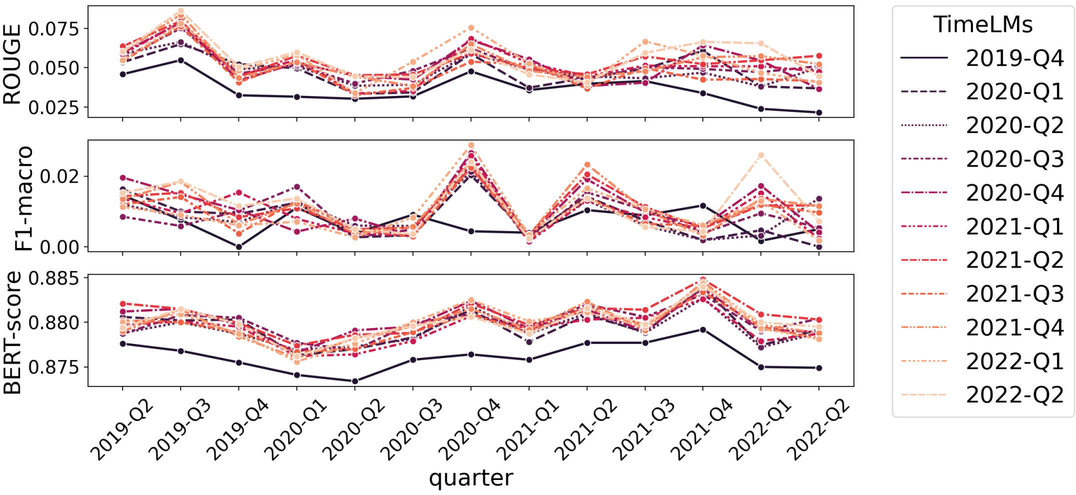


(a) Single Token

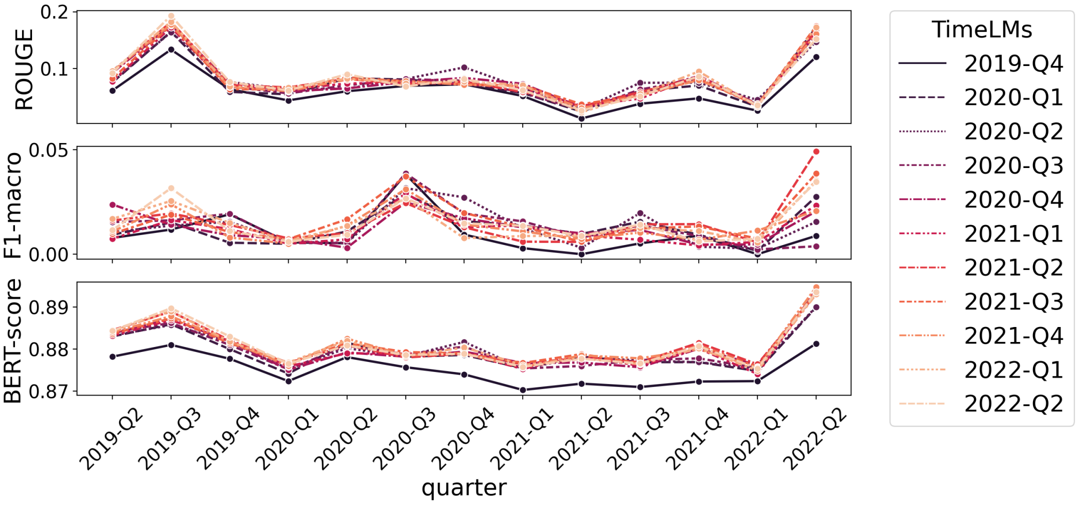


(b) Multi-token

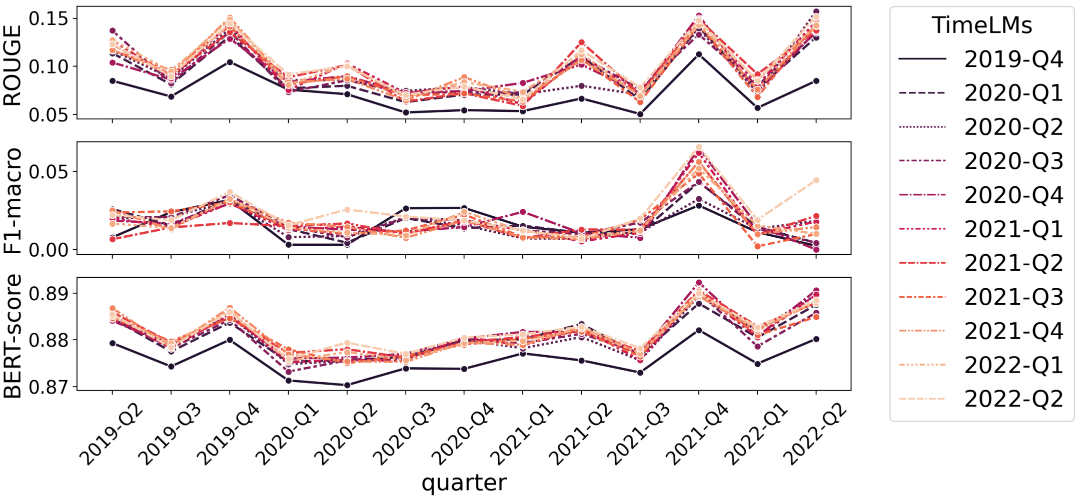
Figure 7: Single-token probing and multi-token generation for the UNCHANGED split.



(a) UPDATED Split

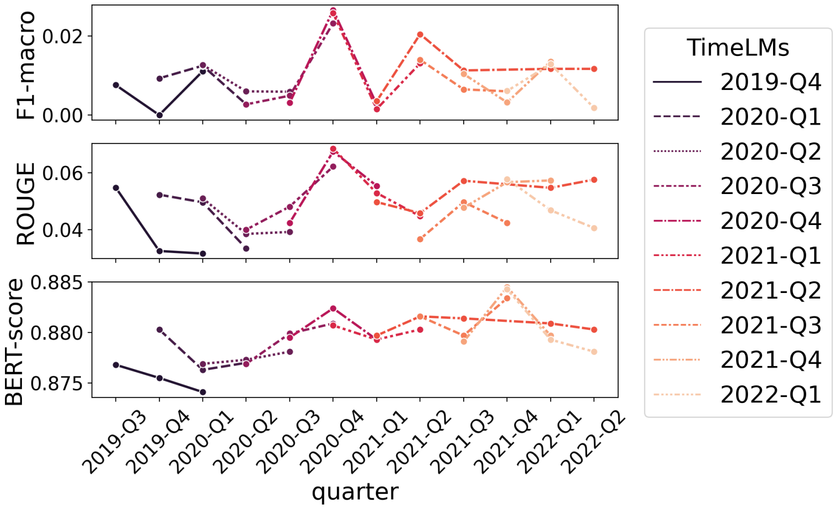


(b) NEW Split

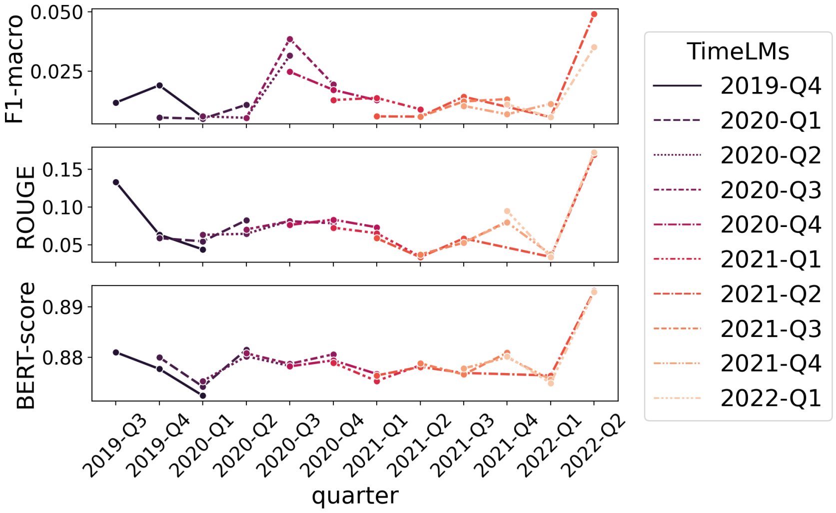


(c) DELETED Split

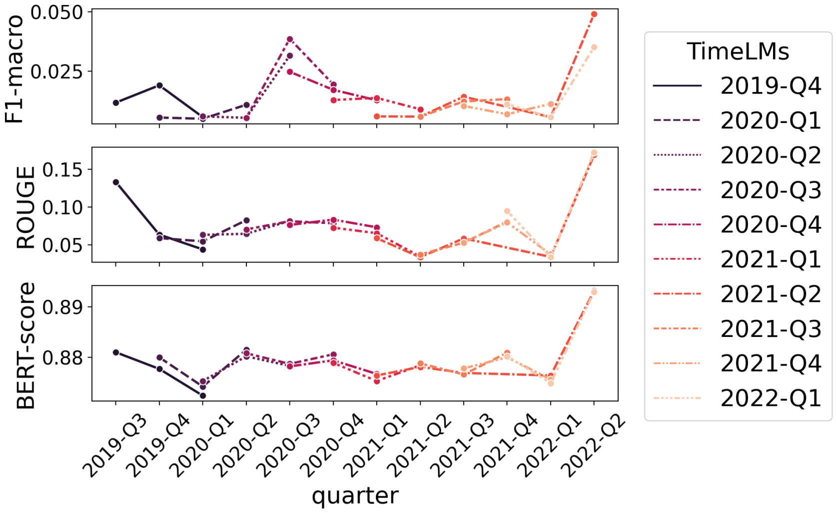
Figure 8: Multi-token generation results for various fine-grained splits.



(a) UPDATED Split



(b) NEW Split



(c) DELETED Split

Figure 9: Multi-token generation results for various fine-grained splits. Here for each model trained on timestep t, we keep the test sets from $t-1$, t and $t+1$.

Real-Time Visual Feedback to Guide Benchmark Creation: A Human-and-Metric-in-the-Loop Workflow

Anjana Arunkumar[1]
aarunku5@asu.edu

Swaroop Mishra[1]
srmishr1@asu.edu

Bhavdeep Sachdeva[1]
bssachde@asu.edu

Chitta Baral[1]
cbaral@asu.edu

Chris Bryan[1]
cbryan16@asu.edu

[1]**Arizona State University**

Abstract

Recent research has shown that language models exploit 'artifacts' in benchmarks to solve tasks, rather than truly learning them, leading to inflated model performance. In pursuit of creating *better benchmarks*, we propose VAIDA, a novel benchmark creation paradigm for NLP, that focuses on *guiding crowdworkers*, an under-explored facet of addressing benchmark idiosyncrasies. VAIDA facilitates sample correction by providing real-time visual feedback and recommendations to improve sample quality. Our approach is domain, model, task, and metric agnostic, and constitutes a paradigm shift for robust, validated, and dynamic benchmark creation via human-and-metric-in-the-loop workflows. We evaluate via expert review and a user study with NASA TLX. We find that VAIDA decreases effort, frustration, mental, and temporal demands of crowdworkers and analysts, simultaneously increasing the performance of both user groups with a 45.8% decrease in the level of artifacts in created samples. As a by-product of our user study, we observe that created samples are adversarial across models, leading to decreases of 31.3% (BERT), 22.5% (RoBERTa), 14.98% (GPT-3 fewshot) in performance.[1]

1 Introduction

Researchers invest significant effort to create benchmarks in machine learning, including ImageNet (Deng et al., 2009), SQUAD (Rajpurkar et al., 2016), and SNLI (Bowman et al., 2015), as well as to develop models that solve them. *Can we rely on these benchmarks?* A growing body of recent research (Schwartz et al., 2017; Poliak et al., 2018; Kaushik and Lipton, 2018) is revealing that models exploit spurious bias/artifacts– unintended correlations between input and output (Torralba and Efros, 2011) (e.g. the word 'not' is associated with the label 'contradiction' in Natural Language Inference (NLI) (Gururangan et al., 2018))– instead of the actual underlying features, to solve many popular benchmarks. Models, therefore, fail to generalize, and experience drastic performance drops when testing with out-of-distribution (OOD) data or adversarial examples (Bras et al., 2020; Mishra et al., 2020a; McCoy et al., 2019; Zhang et al., 2019; Larson et al., 2019b; Sakaguchi et al., 2019; Hendrycks and Gimpel, 2016). This begs the question: *Shouldn't ML researchers consequently focus on creating 'better' datasets rather than developing increasingly complex models on bias-laden benchmarks?*

Deletion of samples based on bias baseline reports– hypothesis-only baseline in NLI (Dua et al., 2019))– and mitigation approaches such as AFLite (Sakaguchi et al., 2019) (adversarial filtering which deletes targeted data subsets), (Clark et al., 2019; Kaushik et al., 2019), have the following limitations: (i) data deletion/augmentation and residual learning do not justify the original investment in data creation, and (ii) crowdworkers are not provided adequate feedback to learn what constitutes high-quality data– and so have additional overhead due to the manual effort involved in sample creation/validation. Furthermore, (Parmar et al., 2022a) show that biased samples are created even when crowdworkers are provided with an initial set of annotation instructions. One potential solution to these problems is continuous, *in situ* feedback about artifacts while benchmark data is being created. *To our knowledge, there are no approaches that provide real-time artifact identification, feedback, and reconciliation opportunities to data creators, nor guide them on data quality.*

Contributions: (i) We propose *VAIDA* (Visual Analytics for Interactively Discouraging Artifacts), a novel system for benchmark creation that provides continuous visual feedback to data creators in real-time. VAIDA supports artifact identification

---

[1] A video description of VAIDA, generated samples, and detailed analyses are available in the Supplemental Material.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2899–2927
May 2-6, 2023

Component Name	**DQI Implication**	**VAIDA Usage**	**Artifacts Evaluated**
1. Vocabulary	Ambiguity and diversity of a dataset's language	Does the sample contribute new words?	Sample Length (Wallace et al., 2019), New Words Introduced (Yaghoub-Zadeh-Fard et al., 2020; Larson et al., 2020), Jaccard Index between n-grams (Larson et al., 2019a)
2. Inter-Sample N-gram Frequency and Relation	Word/phrase repetition and similarity between samples	Does the sample contribute new combinations of words and phrases?	N-gram overlap (Wallace et al., 2019; Yaghoub-Zadeh-Fard et al., 2020), Mean-IDF (Stasaski et al., 2020)
3. Inter-Sample STS	Syntactic, semantic, and pragmatic sentence parsing	How similar is the hypothesis to all other premises or hypotheses?	Multi-hop reasoning (Wallace et al., 2019), Similarity and overlap (Yaghoub-Zadeh-Fard et al., 2020), Diversity (Larson et al., 2019a)
4. Intra-Sample Word Similarity	Word overlap and similarity within sample statements	How similar are all words within a sample?	Coreference Resolution, Multi-hop reasoning (Wallace et al., 2019), Word Overlap (Larson et al., 2020)
5. Intra-Sample STS	Phrase/sentence level overlap within a sample	How similar is the hypothesis to the premise?	N-gram repetition and overlap (Yaghoub-Zadeh-Fard et al., 2020)
6. N-gram Frequency per Label	Distribution of samples according to annotation	Is the hypothesis too obvious for the system?	Logic and Calculations (Wallace et al., 2019), Diversity (Larson et al., 2019a), Outliers, Entropy (Stasaski et al., 2020)
7. Inter-Split STS	Optimal similarity between train and test samples	Is the sample too similar to an existing sample?	Entity Distractors, Novel Clues (Wallace et al., 2019), Coverage (Larson et al., 2019a)

Table 1: Language properties considered in DQI that indicate artifact presence, their interpretation in VAIDA, and corresponding methods used in crowdsourcing pipeline evaluation; STS: semantic textual similarity.

and resolution, implicitly educating *crowdworkers* and *analysts* on data quality. (ii) We evaluate VAIDA empirically through expert review and a user study to understand the cognitive workload it imposes. The results indicate that VAIDA decreases mental demand, temporal demand, effort, and frustration of crowdworkers (31.1%) and analysts (14.3%); it increases their performance by 34.6% and 30.8% respectively, and educates crowdworkers on how to create *high-quality* samples. Overall, we see a 45.8% decrease in the presence of artifacts in created samples. (iii) Even though our main goal is to reduce artifacts in samples, we observe that samples created in our user study are adversarial across language models with performance decreases of 31.3% (BERT), 22.5% (RoBERTa), and 14.98% (GPT-3 fewshot).

2 Related Work

This work sits at the intersection of two primary areas: (1) *visual analysis of data quality* (higher presence of artifacts indicates lower quality), and (2) *development of a novel data collection pipeline*.[2]

2.1 Sample Quality and Artifacts

Data Shapley (Ghorbani and Zou, 2019) has been proposed as a metric to quantify the value of each training datum to the predictor performance. However, the metric might not signify bias content, as the value of training datum is quantified based on predictor performance, and biases might favor the predictor. Moreover, this approach is model and task-dependent. VAIDA uses DQI (Data Quality Index), proposed by (Mishra et al., 2020b), to: (i) compute the overall data quality for a benchmark with n data samples, and (ii) compute the impact of a new $(n+1)^{th}$ data sample. Table 1 broadly defines DQI components, along with their interpretation in VAIDA, and juxtaposes them against evaluation methods used in prior works on crowdsourcing pipelines, as discussed in 2.2. (Wang et al., 2020) propose a tool for measuring and mitigating artifacts in image datasets.

2.2 Crowdsourcing Pipelines

Several pipelines have been proposed to handle various aspects of artifact presence in samples.

Adversarial Sample Creation: Pipelines such as Quizbowl (Wallace et al., 2019) and Dynabench (Kiela et al., 2021), highlight portions of text from input samples during crowdsourcing, based on how important they are for model prediction; this prompts users to alter their samples, and produce samples that can fool the model being used for evaluation (Talmor et al., 2022). While these provide more focused feedback compared to adversarial pipelines like ANLI (Nie et al., 2019), which do not provide explicit feedback on text features, adversarial sample creation is contingent on performance against a specific model (Quizbowl for instance is evaluated against IR and RNN models, and may therefore not see significant performance drops against more powerful models). Additionally, such sample creation might introduce new artifacts over time into the dataset and doesn't always correlate with high quality– for instance, a new entity introduced to fool a model in an adversarial sample might be the result of insufficient inductive bias, though reducing the level of spurious bias.

A similar diagnostic approach is followed for

---

[2] Detailed related work is in the Supplemental Material.

unknown unknown identification– i.e., instances for which a model makes a high-confidence prediction that is incorrect. (Attenberg et al., 2015) and (Vandenhof, 2019) propose techniques to identify UUs, in order to discover specific areas of failure in model generalization through crowdsourcing. The detection of these instances is however, model-dependent; VAIDA addresses the occurrence of such instances by comparing sample characteristics between different labels to identify (and resolve) potential artifacts and/or under-represented features in created data.

Promoting Sample Diversity: Approaches focusing on improving sample diversity have been proposed, in order to promote model generalization. (Yaghoub-Zadeh-Fard et al., 2020) use a probabilistic model to generate word recommendations for crowdworker paraphrasing. (Larson et al., 2019a) propose retaining only the top k% of paraphrase samples that are the greatest distance away from the mean sentence embedding representation of all collected data. These 'outlier' samples are then used to seed the next round of paraphrasing. (Larson et al., 2020) iteratively constrain crowdworker writing by using a taboo list of words, that prevents the repetition of over-represented words, which are also a source of spurious bias. Additionally, (Stasaski et al., 2020) assess the new sample's contribution to the diversity of the entire sub-corpus.

Controlled Dataset Creation: Previous work (Roit et al., 2019) in controlled dataset creation trains crowdworkers, and selects a subset of the best-performing crowdworkers for actual corpus creation. Each crowdworker's work is reviewed by another crowdworker, who acts as an analyst (as per our framework) of their samples. However, in real-world dataset creation, such training and selection phases might not be possible. Additionally, the absence of a metric-in-the-loop basis for feedback provided during training can potentially bias (through trainers) the created samples.

As shown in Table 1, DQI encompasses the aspects of artifacts studied by the aforementioned works; it further quantifies the presence of many more inter and intra-sample artifacts,[3] and provides a one-stop solution to address artifact impact on multiple fronts. VAIDA leverages DQI to identify artifacts, and further focuses on educating crowdworkers on exactly 'why' an artifact is undesirable, as well as the impact its presence will have on the overall corpus. It is also easily extensible to incorporate additional metrics such as quality control measures (Ustalov et al., 2021), enabling benchmarking evaluation in a reproducible manner. This is in contrast to the implicit feedback provided by word recommendation and/or highlighting in prior works, based on a static set of metrics– VAIDA facilitates the systematic elimination of artifacts without the unintentional creation of new artifacts, something that has hitherto remained unaddressed.

3 Modules

In this section, we describe VAIDA's important backend processes.

DQI: DQI can be expressed as a quality metric that examines different sources of artifacts in text, by scoring samples along 7 different components. We use DQI in order to demonstrate VAIDA's ability to cover multiple facets of artifact creation, although VAIDA is metric agnostic. VAIDA uses an intuitive traffic signal color coding (high>>moderate>>low) to indicate levels of artifacts (i.e., quality) in samples. Hyperparameters for color mapping with respect to DQI component values depend on (i) the application type, and (ii) characteristics of pre-existing samples present in the corpus at the time of new sample creation.[4] For instance, when recreating SNLI (Bowman et al., 2015) with VAIDA, we tune hyperparameters separating the boundary between red, yellow, and green flags on 0.01% of the SNLI training dataset manually in a supervised manner (Mishra et al., 2020b).

AutoFix: We propose AutoFix as a module to help crowdworkers avoid creating bad samples by recommending changes to a sample to improve its quality. The AutoFix algorithm is explained in Figure 1. Given a premise, hypothesis, and DQI values for the hypothesis, AutoFix sequentially masks each word in the hypothesis and ranks words based on their influence on model output, i.e., their importance. The hypothesis word of highest importance is replaced, to achieve at least moderate quality. DQI hence controls the amount and aspect of changes made by AutoFix. By incrementally changing the sample, users can understand how and why their sample is being modified and how DQI values are affected.

TextFooler: From an analyst's perspective,

---

[3] See Supplemental Material for details on artifacts that DQI identifies.

[4] See Supplemental Material: Evaluation, for details across all DQI components, hyperparameter tuning, and analyses.

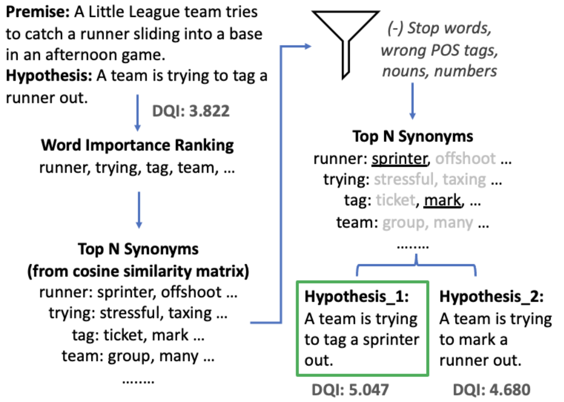


Figure 1: AutoFix Algorithm applied to an SNLI entailment sample, replacing one word per iteration. The DQI of the hypothesis changes from 3.822 to 5.047.

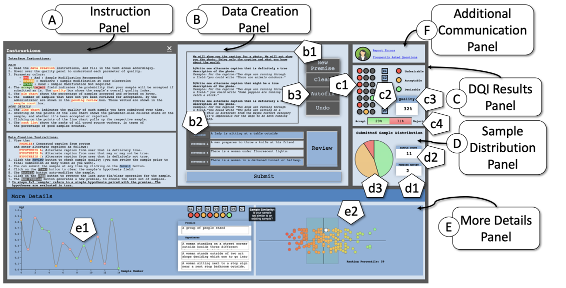


Figure 2: VAIDA's crowdworker interface consists of six linked panels: (A) Instructions, (B) Data creation, (C) DQI results, (D) Sample distribution, (E) More details, and (F) Additional communication.[5]

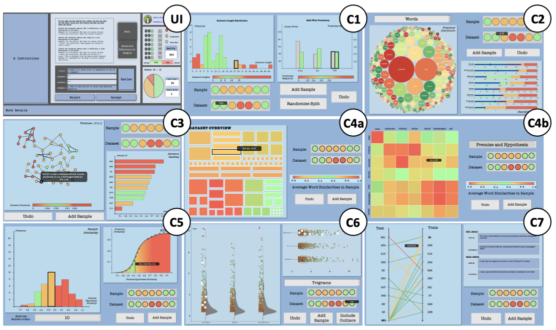


Figure 3: VAIDA provides a collection of interfaces for the analyst supporting detailed analysis and investigation of submitted samples and the overall benchmark.[5]

the quality of a submitted sample might be "too low" because (i) the crowdworker might not employ AutoFix appropriately, or (ii) there is a narrow acceptability range due to the criticality of the application domain, such as in BioNLP (Lee et al., 2020; Parmar et al., 2022b). We therefore implement TextFooler (Jin et al., 2019) to adversarially transform low-quality samples (instead of discarding them), to improve benchmark robustness, and ensure that crowdsourcing effort is not wasted. We initially use AFLite (Bras et al., 2020), to bin samples into *good* (retained samples) and *bad* (filtered samples) splits. Using TextFooler, we adversarially transform bad split data to flip the label; we revert back to the original label and identify sample artifacts using DQI (see Figure 5).

4 Interface Design and Workflow

VAIDA provides customized interfaces for both crowdworkers and analysts, as shown in Figures 2, 3 respectively.[5] We describe VAIDA's workflow via a case study for sample creation (crowdworker) and review (analyst) in Figures 4,5.

The crowdworker interface provides instructions **(A)** to navigate through the panels and how to interpret feedback for created samples. Communication links for FAQs, and error reporting are also provided (F). Crowdworkers can then review the instructions for data creation **(B)**– here, we use the same instructions provided in the original SNLI crowdsourcing interface **(b1)**.

1. Sample Creation: The premise field auto-populates with a caption from the Flickr30 corpus **(b2)**. The crowdworker must create three hypotheses (for the entailment, neutral, contradiction labels) at a time, though they are reviewed individually.

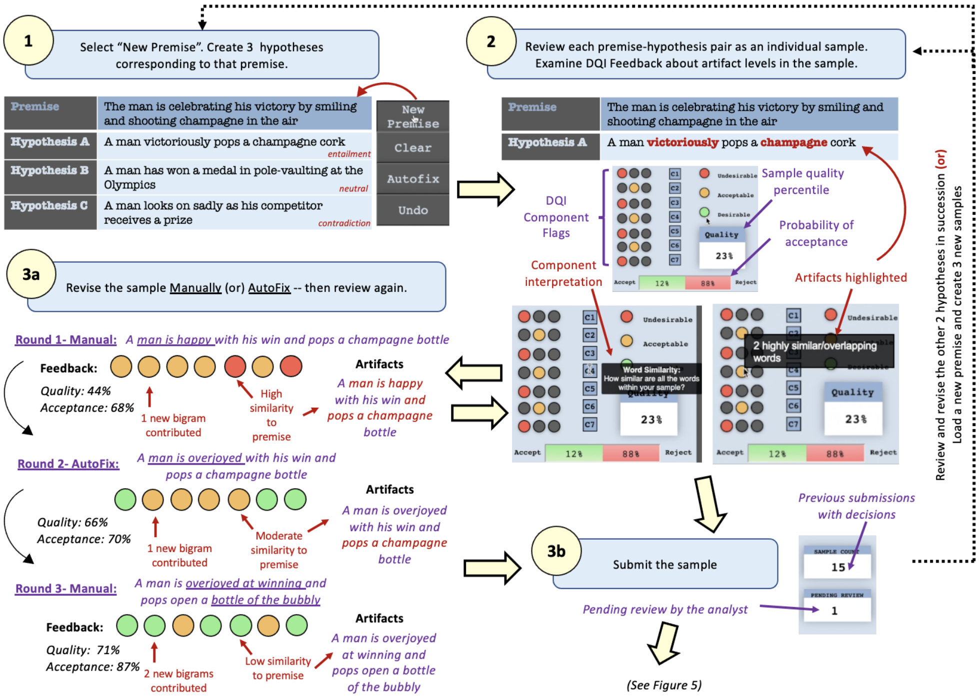
2. DQI Evaluation Feedback: On clicking the review button, DQI feedback is shown in **(C)**, for each component. Each colored circle **(c1)** indicates the level of artifacts present corresponding to each DQI component for the created sample; hovering displays a tooltip that explains and highlights **(c2)** the artifacts in the created sample, pertaining to the category of bias covered by that component. The overall sample quality **(c3)** is calculated, by averaging over the artifact percentiles of all 7 DQI components for the sample. By comparing sample quality with that of pre-existing samples, the probability that the sample will be accepted/rejected is shown **(c4)**. The user can choose to either revise or submit the sample.

3a. Sample Revision: *Manual Revision* or *AutoFix* can be done. We illustrate the improvement of sample quality in this step in Figure 4.

3b. Sample Submission: After review (and potentially iterative DQI evaluations/sample revi-

---

[5] See Supplemental Material: Interface Design for interface intuitions and detailed descriptions, with full-resolution images.



1 Select “New Premise”. Create 3 hypotheses corresponding to that premise.

Premise	The man is celebrating his victory by smiling and shooting champagne in the air
Hypothesis A	A man victoriously pops a champagne cork *entailment*
Hypothesis B	A man has won a medal in pole-vaulting at the Olympics *neutral*
Hypothesis C	A man looks on sadly as his competitor receives a prize *contradiction*

2 Review each premise-hypothesis pair as an individual sample. Examine DQI Feedback about artifact levels in the sample.

Premise	The man is celebrating his victory by smiling and shooting champagne in the air
Hypothesis A	A man **victoriously** pops a **champagne** cork

3a Revise the sample Manually (or) AutoFix -- then review again.

Round 1- Manual: A man is happy with his win and pops a champagne bottle

Feedback: Quality: 44% Acceptance: 68%

Artifacts: A man is happy with his win and pops a champagne bottle

Round 2- AutoFix: A man is overjoyed with his win and pops a champagne bottle

Quality: 66% Acceptance: 70%

Artifacts: A man is overjoyed with his win and pops a champagne bottle

Round 3- Manual: A man is overjoyed at winning and pops open a bottle of the bubbly

Feedback: Quality: 71% Acceptance: 87%

Artifacts: A man is overjoyed at winning and pops open a bottle of the bubbly

3b Submit the sample

Review and revise the other 2 hypotheses in succession (or) Load a new premise and create 3 new samples

Figure 4: VAIDA workflow for the creation of a single sample by a crowdworker.

sions), the sample can be submitted for benchmark inclusion. On submission, a sample enters a *pending* state **(d1)** until review by the analyst.

While crowdworkers work within a single tightly-coordinated interface to create, submit, and review samples, analysts can navigate between a set of nine screens (Figure 3) to review samples in detail to make ‘accept’, ‘reject’, and ‘modification’ decisions, and to assess overall benchmark quality. Analysts review samples in batches of size 50.[6]

4. Analyst Review: Review can consist of several different operations by the analyst.

(i) Direct Acceptance– The home page **(UI)** for the analyst provides a view similar to the crowdworker interface, and allows the analyst to review the work of a single crowdworker. If the data quality is deemed to be sufficiently high by just viewing the DQI color flags and quality percentile, the analyst can directly accept the sample into the corpus from this screen, as shown in Figure 5.

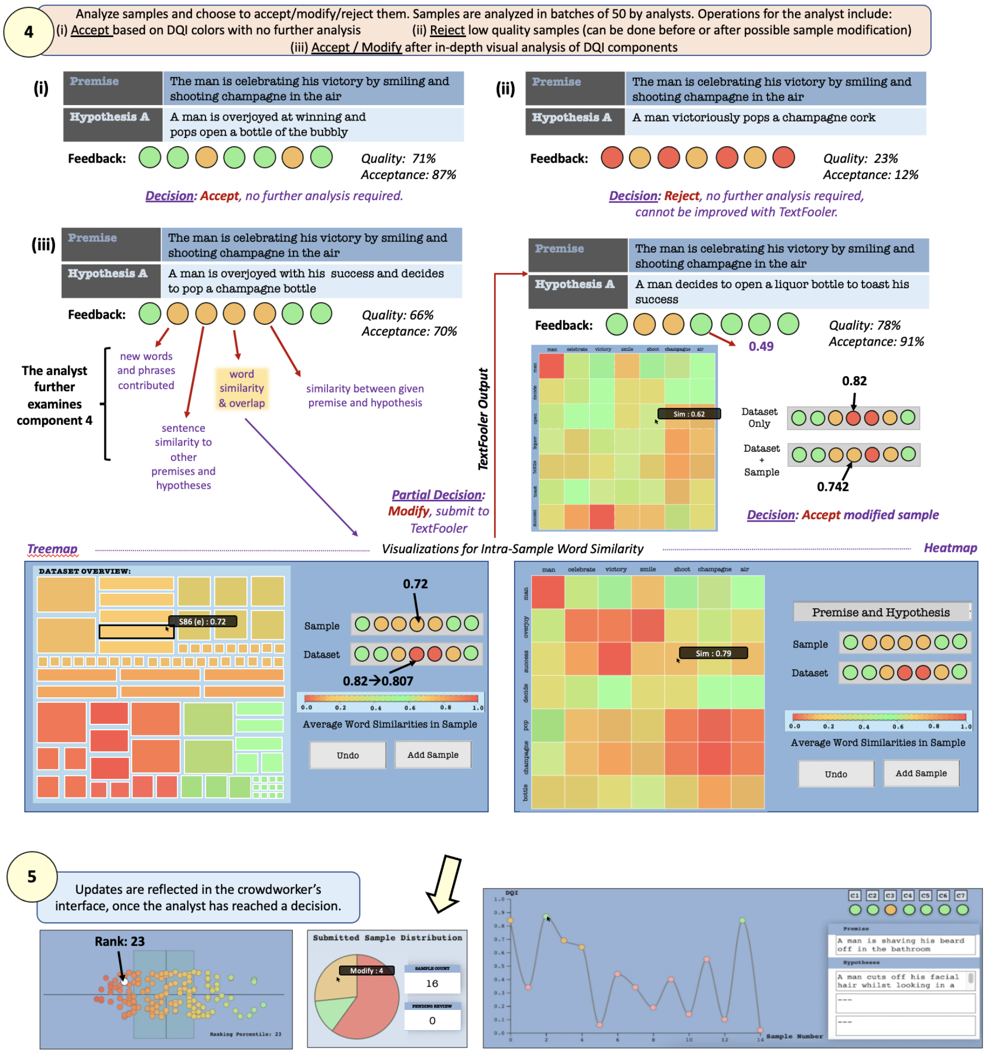
(ii) Visual Analysis and Modification– VAIDA provides several visualizations **(C1-C7)** to support detailed analysis and review of submitted samples, and to assess artifact presence (i.e., quality) in the overall benchmark. Each visualization allows the analyst to simulate how adding one or more submitted samples affects the benchmark’s quality.

For example, in Figure 5, we show the visualizations for analysis of DQI component 4 (C4), which deals with artifacts caused due to word similarity within a sample. For each sample in the corpus, the word similarities between all possible pairs of words are averaged; the samples are then hierarchically displayed as a treemap based on their DQI color mapping and average word similarity. We note that the color scale followed is bilinear, as while artifacts must be eliminated, there still needs to be a sufficient inductive bias for the sample to be solvable. The size of a rectangle indicates the distance of each sample from the average word similarity across all samples. The new sample is highlighted in the treemap with a black outline. In the example shown, we see that the dataset’s C4 value decreases slightly (0.807) when the new sample is added, though the flag color remains red.

Further examination of the new sample can be done, to establish why the sample has low effect on the dataset’s C4; the user clicks on the sample rectangle in the treemap, and is taken to a heat map view. The heatmap shows the similarity values between every word pair (from the premise/hypothesis/both) in a tooltip on hover, with the values mapped to the same color scale used for

---

[6] DQI components 1, 2, 3, 6, and 7, gauge artifact presence relative to pre-existing samples. We observe given at least 50 pre-existing samples, DQI component values change by less than 5% for the overall dataset if another 50 samples are accepted.


4

Analyze samples and choose to accept/modify/reject them. Samples are analyzed in batches of 50 by analysts. Operations for the analyst include:
(i) Accept based on DQI colors with no further analysis (ii) Reject low quality samples (can be done before or after possible sample modification)
(iii) Accept / Modify after in-depth visual analysis of DQI components

(i)

Premise	The man is celebrating his victory by smiling and shooting champagne in the air
Hypothesis A	A man is overjoyed at winning and pops open a bottle of the bubbly

Feedback: Quality: 71% Acceptance: 87%

Decision: Accept, no further analysis required.

(ii)

Premise	The man is celebrating his victory by smiling and shooting champagne in the air
Hypothesis A	A man victoriously pops a champagne cork

Feedback: Quality: 23% Acceptance: 12%

Decision: Reject, no further analysis required, cannot be improved with TextFooler.

(iii)

Premise	The man is celebrating his victory by smiling and shooting champagne in the air
Hypothesis A	A man is overjoyed with his success and decides to pop a champagne bottle

Feedback: Quality: 66% Acceptance: 70%

The analyst further examines component 4

Partial Decision: Modify, submit to TextFooler

Premise	The man is celebrating his victory by smiling and shooting champagne in the air
Hypothesis A	A man decides to open a liquor bottle to toast his success

Feedback: Quality: 78% Acceptance: 91%

Decision: Accept modified sample

5

Updates are reflected in the crowdworker's interface, once the analyst has reached a decision.

Figure 5: VAIDA workflow for the evaluation of a single sample by an analyst, and subsequent analyst feedback provided to the crowdworker.

the treemap . For instance, in Figure 5, the words 'man' and 'champagne are repeated verbatim, and 'overjoyed', 'smiling', 'celebrating', 'pop', and 'shoot' also have similarity of [0.6–1]. This indicates that several words in the sample are too closely related and constitute artifacts.

Each such visualization is therefore individually tailored to represent a specific DQI component of interest, based on the linguistic features examined for artifact creation in that component. We further elaborate on the design intuitions and analysis conducted with all the component visualizations available to the analyst in the supplementary.

Post analysis, if the analyst feels that the submitted sample requires only a minor change (for instance, reshuffling or the addition of a single word) to warrant acceptance, then they can invoke the TextFooler module to transform the sample adversarially, and then accept, thereby ensuring minimal data loss. In the case shown, TextFooler improves the sample's C4 with most of the heatmap varying from 0.3-0.7. Due to this, the analyst decides to accept the updated sample.

5. Analyst Feedback: Once the analyst has reached a decision, the crowdworker sees updates (Figure 5) in (D) to the reviewed sample count (d2) – increases to 16– and the pie chart that indicates the distribution of actions taken by the analyst over all samples submitted by the crowdworker (d3). Additionally, in (E), the line chart (e1), which contains the history of previously submitted samples is updated. The x-axis denotes the sample number and the y-axis denotes the quality percentile (c3), of the corresponding sample. On click, the corresponding sample is loaded, along with its DQI component values and feedback to the crowdworker. The crowdworker can also use the box plot (e2), to view their current rank, and choose to view the sample history of another crowd worker.

5 Evaluation

We evaluate VAIDA's efficacy at providing real-time feedback to educate crowdworkers during benchmark creation using expert review and a user study. We also evaluate model performance (BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), GPT-3 (fewshot) (Brown et al., 2020)) on data created with VAIDA during the user study.

5.1 Expert Review

We present an initial prototype of our tool, to a set of three researchers with expertise in NLP and knowledge of data visualization. For each expert, the two interfaces were demoed in a Pair-Analytics session (Arias-Hernandez et al., 2011). Participants could ask questions and make interaction/navigation decisions to facilitate a natural user experience. All the experts appreciated the easily interpretable traffic-signal color scheme (and further suggested that alternates be provided to account for color blindness) and found the organization of the interfaces—providing separate detailed views within the analyst workflow– a way to prevent cognitive overload (too much information on one screen). A caveat of this would be the inability for an analyst to simultaneously juxtapose different component visualizations. It was also hypothesized that a learning curve of $\sim$50–60 samples would be required for cohesive use of all system modules by both types of users; however, this would be offset by the eventual capability of users to deal with samples of middling quality based on their multi-granular feedback about artifact presence.

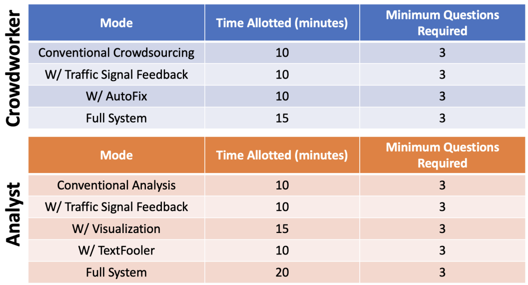
5.2 User Study

Setup: We approach several software developers, testing managers, and undergraduate/graduate students. Based on their domain familiarity (in NLP and visualization, rated from 1:novice-5:expert), we split them into 23 crowdworkers and 8 analysts for constructing NLI samples, given premises. There are 100 high-quality samples in the system at the time each participant participates in each ablation round (Table 2). For both types of users, a preliminary walkthrough of the system configuration, using 2 fixed samples, is conducted for each round of the study (Figure 6). At the end of each round, they are also asked for their comments.[7]

Configuration	Description	User
Conventional Crowdsourcing	No feedback or auto modification tools	C
Conventional Analysis	Manual review without feedback or modification tools	A
Traffic Signal Feedback	Color mapping based on DQI values	C, A
AutoFix	Incremental sample auto-modification functionality	C
TextFooler	Adversarial sample transformation functionality	A
Visualization	Data visualizations for in-depth exploration (also includes traffic signal feedback)	A
Full System	All modules and system functionalities	C,A

Table 2: System configurations used for randomly ordered ablation rounds presented to users functioning as crowdworkers (C) and analysts (A) in the study.

User experience (mental workload) is subjectively evaluated using NASA Task Load Index (Hart, 2006)[7] (NASA TLX); each dimension is scored in a 100-points range, with 5-point steps. Users are also asked to report overall ratings for each system configuration at the end of the study.



Crowdworker

Mode	Time Allotted (minutes)	Minimum Questions Required
Conventional Crowdsourcing	10	3
W/ Traffic Signal Feedback	10	3
W/ AutoFix	10	3
Full System	15	3

Analyst

Mode	Time Allotted (minutes)	Minimum Questions Required
Conventional Analysis	10	3
W/ Traffic Signal Feedback	10	3
W/ Visualization	15	3
W/ TextFooler	10	3
Full System	20	3

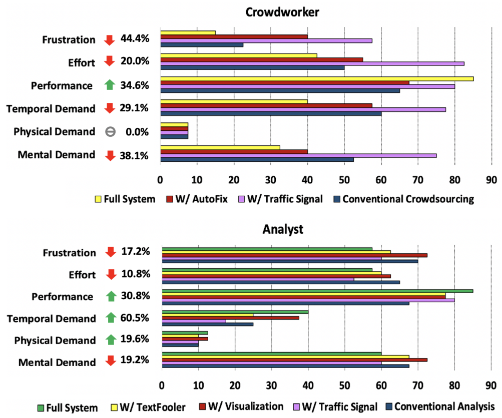
Figure 6: User Study Setup– describes the timeframe and requirements of the user study over ablation rounds.

Analysis: Figure 7 summarizes study results,[8] averaged over all user responses, for different system configurations. The general trend across both crowdworkers and analysts is that there is: (i) significant improvement across all NASA TLX dimensions, (ii) increase in number of samples created/reviewed, and (iii) user ratings for the system, when comparing VAIDA to conventional interfaces. In the case of partial module availability, we find that the effectiveness of traffic signal feedback and visualizations is comparable. The use of AutoFix and TextFooler7 is more prevalent initially, on creation/evaluation of a low or middling quality sample for users as: (i) crowdworkers find constructive

---

[7] See Supplemental Material: User Study for more details. We do aggregated analysis of comments, full quotes of comments are present in the Supplemental Material. We also have IRB approval to conduct this user study.

[8] All results are found to have $p \leq 0.02$



Crowdworker

Mode	Number of Questions Created	System Score (/100)
Conventional Crowdsourcing	3	50
W/ Traffic Signal Feedback	2.75	72.5
W/ AutoFix	3.5	62.5
Full System	5.25	85

Analyst

Mode	Number of Questions Analyzed	System Score (/100)
Conventional Analysis	3	40
W/ Traffic Signal Feedback	5	67.5
W/ Visualization	2.75	72.5
W/ TextFooler	3.75	65
Full System	5.5	82.5

Figure 7: User Study Results– NASA TLX dimension scores, and user scoring of the configurations, averaged across participant responses, for each ablation round. Percentages shown indicate comparison of full system scores with respect to the conventional system.

sample modification more difficult initially, and (ii) analysts are initially unsure of how to deal with middling quality samples.

Learning Curve: At the end of the study, all users are asked the following: *"What do you think high quality means?"* We find that users are able to distinguish certain patterns that promote higher quality, such as keeping sentence length appropriate and uniform across labels (not too long/short), using complex phrasing ('not bad')/gender information/modifiers across labels, decreasing premise-hypothesis word overlap; they also do not display undesirable behavior like tweaking previously submitted samples just to create more. We also find an overall decrease of -45.8% in the level of artifacts of created samples, across all rounds of ablation.

User Education: We also conduct a secondary study where a subset of participants (7 crowdworkers and 2 analysts) agreed to create/ analyze samples, for varying numbers of pre-accepted samples (Figure 8), in only the full system condition. We find that when participants are directly started in situations with $>$ 500 samples in the system, their unfamiliarity with the system initially causes a steepening of the learning curve compared to the cold start condition; this also tapers and saturates more slowly than cold start as the users gain experience. This is attributed to: (a) an increased likelihood of samples of low/middling quality (more artifacts) being created (evinced by performance), and (b) lower impact of an individual sample on overall dataset quality. We also find that users who create $\sim$50 samples report lesser reliance on AutoFix as they get better at creating higher quality samples; those who analyze $\sim$75 samples use TextFooler more efficiently as they understand how to deal with samples of middling quality better in the cold start condition. These numbers increase by $\sim$25% when users start with 500 pre-existing samples.

5.3 Model Performance Results

We evaluate BERT and RoBERTa (trained on the full SNLI dataset), and GPT-3 (in fewshot setting) against the data created during the ablation rounds of the user study.[9] There are 100 high-quality samples(DQI>0.7) from the original SNLI dataset present in the system for the study[10]. This remains constant across all ablation rounds and for all users. 69 samples are created per ablation round during our study, across all users, for a total of 345(69*5) samples that we evaluate with the models. Figure 9 shows the results for samples over each round of ablation. ([10]) In the case of TextFooler, samples are created using the 'full system' condition and then further modified using TextFooler by the analyst. The other sample sets are not modified by the analyst, and are directly accepted after evaluation.

We find that across all models, performance is lower when explicit quality feedback (via the traffic signal scheme) is provided, compared to the regular crowdsourcing condition. The highest drop is seen for BERT (-20.66%), while GPT-3 shows a lesser magnitude of performance decrease (-12.89%). Performance further decreases for all models when AutoFix is implemented, indicating the effectiveness of this module in seeding suggestions for sample improvement; the magnitude of performance loss follows BERT>>RoBERTa>>GPT-3. We can attribute this apparent variation in model robustness i.e., BERT<<RoBERTa<<GPT-3 as proportionate to increase in size of (i) the respective language models, and (ii) the pre-training corpora.

A significant decrease is seen in the full system and TextFooler conditions. Particularly, in the TextFooler round, performance sharply decreases

---

[9] We also create samples for the Story CLOZE dataset in the full system condition; see Supplementary Material

[10] The dataset is included in the Supplemental Material.

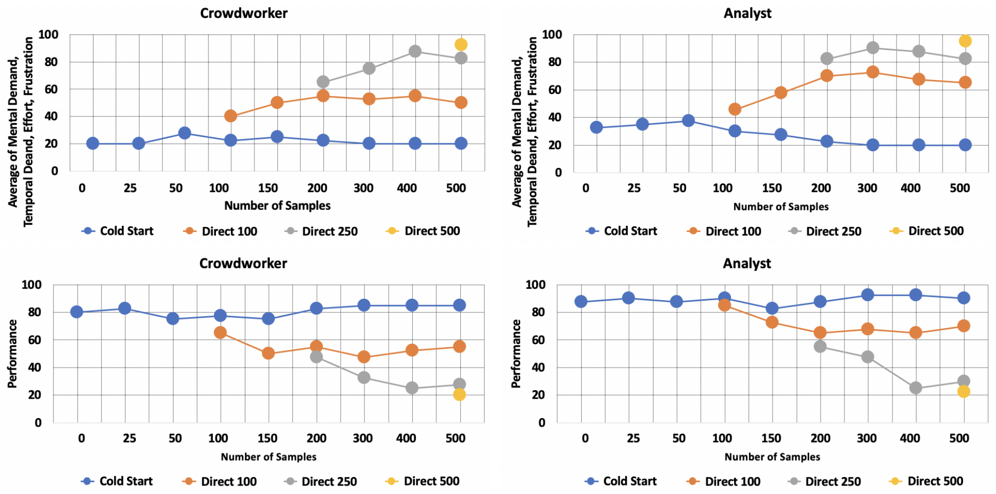


Figure 8: User education curves. Cold start has no pre-existing samples, and direct-n has n pre-existing samples. Mental Demand, Temporal Demand, Frustration, and Effort are averaged, Physical Demand is ignored. Performance is plotted separately as it shows differing behavior from the others.

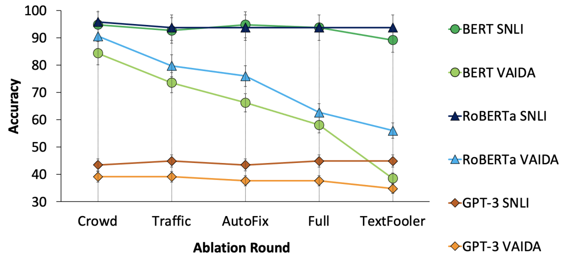


Figure 9: Model performance results for samples created during each ablation round of the user study.

for all models (-31.3% BERT, -22.5% RoBERTa, -14.98% GPT-3). Furthermore, in the TextFooler round, there is a -71.70% decrease in the level of sample artifacts. This indicates that crowdworkers and analysts are able to utilize VAIDA's affordances to create more robust text samples.

6 Discussion and Conclusion

We propose VAIDA, a paradigm to address benchmark artifacts, by integrating human-in-the-loop sensemaking with continuous visual feedback. VAIDA uses several visualization interfaces to analyze quality considerations (based on artifact levels) at multiple granularities. While we do not explicitly address computational quality control or fairness consideration (though some aspects can be targeted by currently integrated metrics), since VAIDA is extensible to the incorporation of customized backend-metrics, we believe our paradigm can support multi-faceted benchmark evaluation.

In our usability evaluation, we see that users report greater satisfaction, and lower difficulty with their work and system experience; this implies possible higher crowdworker retention and engagement. Additionally, in our study, we see that users effectively identify and avoid artifact patterns during sample creation. Based on our study results, we believe that a minimum of 30 annotators would be needed for large-scale data creation to ensure timely feedback (i.e., sample decision provided within 24 hours) to crowdworkers. Based on our secondary study, we believe analysts will exhibit increased performance and maintain satisfaction ratings in full-scale creation, as they will become well-versed in the nuances of bias for the data-creation task they are evaluating, as well as the visualizations being used. This, however, is contingent on restricting the visualization views to display only the 200–300 samples with closest artifact levels to the sample being evaluated.

Overall, samples created with VAIDA are found to not only of higher quality than achieved with conventional crowdsourcing, but are also seen to be adversarial across transformer models. This is also maintained across multiple task types– we additionally create StoryCLOZE (Schwartz et al., 2017) samples with VAIDA[9]. This was done independent of the study described in the paper, with 4 crowdworkers creating samples. However, for this, several interface features had to be changed, so we focus on reporting only NLI results in the main paper. VAIDA hence demonstrates a novel, dynamic approach for building benchmarks and mitigating artifacts, and serves as a starting point for the next generation of benchmarks in machine learning.

Limitations

In future work, we intend to integrate VAIDA with an actual crowdsourcing framework, and run a full-scale data creation study to create a high-quality benchmark. Expanding to such a setup will require additional back-end engineering, to ensure that (i) timely and accurate feedback continues to be provided in real-time to crowdworkers, (ii) analysts are available on hand to process samples in a timely manner. This is out of scope for the current paper (e.g., it would require a significant budget), but we see our current work as a stepping stone in this direction. Additionally, studying the problem and designing the visualizations and real-time feedback mechanisms are essential steps before moving to large-scale evaluation; the novel affordances and designs are a necessary first step, and we believe they will be impactful to the NLP community.

Crowdworker retention and engagement in this full-scale setting also need to be evaluated, in order to better contextualize the learning curve associated with system usage and handling artifact creation, given an increasingly higher number of pre-existing system samples. Comparing this setup directly with the effect of in-depth user training (Roit et al., 2019) on artifact creation and review, prior to crowdsourcing, would also further help analyze and quantify if/how user strategy and performance changes during VAIDA usage.

Design modifications when creating different types of datasets will mainly require the redesigning of sample input fields, corresponding to the application and the type of metric used for artifact evaluation. However, in full-scale dataset creation, the visualization views for the analyst corresponding to different artifact types will have to be restricted to the $\sim$300 samples closest artifact levels, to the given sample being created, in order to facilitate scalable processing for analysts. Additionally, since visualization familiarity is required for the analyst to effectively review samples, the analysts may choose to streamline analysis by only using a subset of the provided visualization types in their version of the system, corresponding to the application domain.

References

Richard Arias-Hernandez, Linda T. Kaastra, Tera M. Green, and Brian Fisher. 2011. Pair analytics: Capturing reasoning processes in collaborative visual analytics. In *2011 44th Hawaii International Conference on System Sciences*, pages 1–10.

Joshua Attenberg, Panos Ipeirotis, and Foster Provost. 2015. Beat the machine: Challenging humans to find a predictive model's "unknown unknowns". *Journal of Data and Information Quality (JDIQ)*, 6(1):1–17.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew E Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. *arXiv preprint arXiv:2002.04108*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. *arXiv preprint arXiv:1909.03683*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning. *arXiv preprint arXiv:1904.02868*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Sandra G Hart. 2006. Nasa-task load index (nasa-tlx); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 50, pages 904–908. Sage publications Sage CA: Los Angeles, CA.

Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*.

Divyansh Kaushik and Zachary C Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. *arXiv preprint arXiv:1808.04926*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. 2021. Dynabench: Rethinking benchmarking in nlp. *arXiv preprint arXiv:2104.14337*.

Stefan Larson, Anish Mahendran, Andrew Lee, Jonathan K Kummerfeld, Parker Hill, Michael A Laurenzano, Johann Hauswald, Lingjia Tang, and Jason Mars. 2019a. Outlier detection for improved data quality and diversity in dialog systems. *arXiv preprint arXiv:1904.03122*.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. 2019b. An evaluation dataset for intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027*.

Stefan Larson, Anthony Zheng, Anish Mahendran, Rishi Tekriwal, Adrian Cheung, Eric Guldan, Kevin Leach, and Jonathan K Kummerfeld. 2020. Iterative feature mining for constraint-based data collection to increase data diversity and model robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8097–8106.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Nitika Mathur, Tim Baldwin, and Trevor Cohn. 2020. Tangled up in bleu: Reevaluating the evaluation of automatic machine translation evaluation metrics. *arXiv preprint arXiv:2006.06264*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Swaroop Mishra, Anjana Arunkumar, Chris Bryan, and Chitta Baral. 2020a. Our evaluation metric needs an update to encourage generalization. *ArXiv*, abs/2007.06898.

Swaroop Mishra, Anjana Arunkumar, Chris Bryan, and Chitta Baral. 2022. A survey of parameters associated with the quality of benchmarks in nlp. *arXiv preprint arXiv:2210.07566*.

Swaroop Mishra, Anjana Arunkumar, Bhavdeep Singh Sachdeva, Chris Bryan, and Chitta Baral. 2020b. Dqi: A guide to benchmark evaluation. *arXiv: Computation and Language*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

Mihir Parmar, Swaroop Mishra, Mor Geva, and Chitta Baral. 2022a. Don't blame the annotator: Bias already starts in the annotation instructions. *arXiv preprint arXiv:2205.00415*.

Mihir Parmar, Swaroop Mishra, Mirali Purohit, Man Luo, Murad Mohammad, and Chitta Baral. 2022b. In-boxbart: Get instructions into biomedical multitask learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 112–128.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. *arXiv preprint arXiv:1805.01042*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Paul Roit, Ayal Klein, Daniela Stepanov, Jonathan Mamou, Julian Michael, Gabriel Stanovsky, Luke Zettlemoyer, and Ido Dagan. 2019. Crowdsourcing a high-quality gold standard for qa-srl. *arXiv preprint arXiv:1911.03243*.

Rachel Rudinger, Chandler May, and Benjamin Van Durme. 2017. Social bias in elicited natural language inferences. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 74–79.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.

David Schuff, Karen Corral, and Ozgur Turetken. 2011. Comparing the understandability of alternative data warehouse schemas: An empirical study. *Decision support systems*, 52(1):9–20.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Li Zilles, Yejin Choi, and Noah A Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the roc story cloze task. *arXiv preprint arXiv:1702.01841*.

Katherine Stasaski, Grace Hui Yang, and Marti A Hearst. 2020. More diverse dialogue datasets via diversity-informed data collection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4958–4968.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2022. Commonsenseqa 2.0: Exposing the limits of ai through gamification. *arXiv preprint arXiv:2201.05320*.

Antonio Torralba and Alexei A Efros. 2011. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE.

Dmitry Ustalov, Nikita Pavlichenko, Vladimir Losev, Iulian Giliazev, and Evgeny Tulin. 2021. A general-purpose crowdsourcing computational quality control toolkit for python. *arXiv preprint arXiv:2109.08584*.

Colin Vandenhof. 2019. A hybrid approach to identifying unknown unknowns of predictive models. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 7, pages 180–187.

Eric Wallace, Pedro Rodriguez, Shi Feng, Ikuya Yamada, and Jordan Boyd-Graber. 2019. Trick me if you can: Human-in-the-loop generation of adversarial examples for question answering. *Transactions of the Association for Computational Linguistics*, 7:387–401.

Angelina Wang, Arvind Narayanan, and Olga Russakovsky. 2020. Vibe: A tool for measuring and mitigating bias in image datasets. *arXiv preprint arXiv:2004.07999*.

Mohammad-Ali Yaghoub-Zadeh-Fard, Boualem Benatallah, Fabio Casati, Moshe Chai Barukh, and Shayan Zamanirad. 2020. Dynamic word recommendation to obtain diverse crowdsourced paraphrases of user utterances. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, pages 55–66.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *Transactions of the Association for Computational Linguistics*, 5:379–395.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*.

A Supplementary Material

The following information is included in the appendix.

- Infrastructure Used
- Run-time Estimations
- Hyper Parameter
- Related Work
- DQI Components
- Interface Design Intuitions
- AutoFix and TextFooler Examples
- User Study
- Expert and User Comments

Please refer to `https://github.com/aarunku5/VAIDA-EACL-2023.git` for:

- Video demos of VAIDA workflow
- Sample dataset generated during the ablation rounds of the user study
- DQI and Model Performance Results for User Study Samples
- DQI Evaluation: Artifact Case Study

A.1 Infrastructure Used

In Section 3, we describe VAIDA's flow by high level workflow and back-end processes(DQI, AutoFix, and TextFooler). Further, as discussed in Subsection 3.2 DQI can be used for quantifying artifact presence for the: i) overall benchmark, and ii) impact of new samples. Depending on the task at hand we run our experiments in different hardware settings. The DQI calculations run mostly using CPU, for new samples as well as overall samples. The AutoFix procedure, as explained in Subsection 3.2, gives the user assistance in improving quality on a per submission basis. Therefore that does not require high GPU intensive systems; we have provisions to shift execution to a GPU as well if necessary to speed up the process. For TextFooler the fine tuning of the model is run on "TeslaV100-SXM2-16GB"; CPU cores per node 20; CPU memory per node: 95,142 MB; CPU memory per core: 4,757 MB– this is not a necessity as code has been tested on lower configuration GPUs as well but we have run our experiments in this setting. The attack part of the TextFooler requires more memory and we run that code on "Tesla V100-SXM2-32GB" com-pute Capability: 7.0 core Clock: 1.53GHz, coreCount: 80, device Memory Size: 31.75GiB device Memory Bandwidth: 836.37GiB/s.

A.2 Run-time Estimations

The DQI calculations run on CPU (for real life setting purposes); for the approximate estimate of the time taken, we run experiments for fixed data size of 10K samples. If the DQI calculations are done to calculate the impact of individual new samples it take a couple of seconds. On the other hand, If we take the whole 10k size dataset it takes around 48 hours to complete the process on CPU. This whole process can be run in parallel to reduce the time taken to 16 hours.

The TextFooler part consists of two steps– the fine tuning part and attack part– for generating adversaries. For fine tuning models we use "TeslaV100-SXM2-16GB", and it takes 20-30 minutes to complete the process. For the attack part we use "Tesla V100-SXM2-32GB", which takes 2-3 hrs for completing 20k data samples. This estimate requires the cosine similarity matrix for word embeddings to be calculated before hand which takes around 1-2 hrs, but this step has to be done only if the word embeddings are modified. This is a rare task so we have kept this separated.

A.3 Hyper Parameters

To look at the estimations of DQI and its variations, we have kept basic hyper-parameters fixed in the experiments. We keep the learning rate to 1e-5, the number of epochs during the experiments are varied from 2-3, the per gpu train batch and eval batch sizes vary from 8-64 samples (the results shown are with respect to a batch size of 8), adam epsilon is set to 1e-8, weight decay is set to 0, maximum gradient normalisation is set to 1, and maximum sequence length is set to 128. For TextFooler the the semantic similarity is fixed to 0.5 uniformly for all the experiments shown in this paper.

Additionally, the variations and range in the DQI parameters are dataset specific, i.e., hyperparameters depend on the application task. (Mishra et al., 2020b) design DQI as a generic metric to evaluate diverse benchmarks. However, the definitions of what constitutes high and low quality will vary depending on the application. For example, BiomedicaNLP might have lower tolerance levels for spurious bias than General NLP. Another case is in water quality– cited as an inspiration for DQI by (Mishra et al., 2020b)– where the quality of water needed for irrigation is different than that of drinking or medicine. We can therefore say that the hyper-parameters in the form of boundaries separating high and low quality data (i.e., inductive and spurious bias) are dependent on applications.

A.4 Related Work

A.4.1 Sample Quality and Artifacts

Data Shapley (Ghorbani and Zou, 2019) has been proposed as a metric to quantify the value of each training datum to the predictor performance. However, the metric might not signify bias content, as the value of training datum is quantified based on predictor performance, and biases might favor the predictor. Moreover, this approach is model and task-dependent. VAIDA uses DQI (Data Quality Index), proposed by (Mishra et al., 2020b), to: (i) compute the overall data quality for a benchmark with n data samples, and (ii) compute the impact of a new $(n+1)^{th}$ data sample. (Wang et al., 2020) concurrently propose a tool for measuring and mitigating artifacts in image datasets.

Data Shapley (Ghorbani and Zou, 2019) has been proposed as a metric to quantify the value of each training datum to the predictor performance. However, this approach is model and task dependent. More importantly, the metric might not signify bias content, as the value of training datum is quantified based on predictor performance, and biases might favor the predictor. VAIDA uses DQI (data quality index), proposed by (Mishra et al., 2020b), to: (i) compute the overall data quality for a benchmark with n data samples, and (ii) compute the impact of a new $(n+1)^{th}$ data sample. The quality of individual features (aspects) of samples are evaluated based on decreasing presence of artifacts and increasing generalization capability. In a concurrent work (Wang et al., 2020), a tool for measuring and mitigating bias in image datasets has also been proposed. DQI estimates artifact presence by calculating seven component values corresponding to a set of language properties;, along with their interpretation in VAIDA.

A.4.2 Crowdsourcing Pipelines

Adversarial Sample Creation: Pipelines such as Quizbowl(Wallace et al., 2019) and Dynabench(Kiela et al., 2021), highlight portions of text from input samples during crowdsourcing, based on how important they are for model prediction; this prompts users to alter their samples, and produce samples that can fool the model being used for evaluation. While these provide more focused feedback compared to adversarial pipelines like ANLI (Nie et al., 2019), which do not provide explicit feedback on text features, adversarial sample creation is contingent on performance against a specific model (Quizbowl for instance is evaluated against IR and RNN models, and may therefore not see significant performance drops against more powerful models). Additionally, such sample creation might introduce new artifacts over time into the dataset and doesn't always correlate with high quality– for instance, a new entity introduced to fool a model in an adversarial sample might be the result of insufficient inductive bias, though reducing the level of spurious bias.

A similar diagnostic approach is followed for unknown unknown identification– i.e., instances for which a model makes a high confidence prediction that is incorrect. (Attenberg et al., 2015) and (Vandenhof, 2019) propose techniques to identify UUs, in order to discover specific areas of failure in model generalization through crowdsourcing. The detection of these instances is however, model-dependent; VAIDA addresses the occurrence of such instances by comparing sample characteristics between different labels to identify (and resolve) potential artifacts and/or under-represented features in created data.

Promoting Sample Diversity: Approaches focusing on improving sample diversity have been proposed, in order to promote model generalization. (Yaghoub-Zadeh-Fard et al., 2020) use a probablistic model to generate word recommendations for crowdworker paraphrasing. (Larson et al., 2019a) propose retaining only the top k% of paraphrase samples that are the greatest distance away from the mean sentence embedding representation of all collected data. These 'outlier' samples are then used to seed the next round of paraphrasing. (Larson et al., 2020) iteratively constrain crowdworker writing by using a taboo list of words, that prevents the repetition of over-represented words, which are also a source of spurious bias. Additionally, (Stasaski et al., 2020) assess the new sample's contribution to the diversity of the entire sub-corpus.

DQI encompasses the aspects of artifacts studied by the aforementioned works; it further quantifies the presence of many more inter and intra-sample artifacts, and provides a one stop solution to address artifact impact on multiple fronts. VAIDA leverages DQI to identify artifacts, and further focuses on educating crowdworkers on exactly 'why' an artifact is undesirable, as well as the impact its presence will have on the overall corpus. This is in contrast to the implicit feedback provided by word recommendation and/or highlighting in prior works– VAIDA facilitates the elimination of artifacts without the unintentional creation of new artifacts, something that has hitherto remained unaddressed.

A.4.3 Task Selection and Controlled Dataset Creation

In this work, we demonstrate VAIDA for a natural language inference task (though it is task-independent), and mimic the SNLI dataset creation and validation processes. Elicited annotation has been found to lead to social bias in SNLI using probablistic mutual information (PMI) (Rudinger et al., 2017). Visual feedback is provided based on DQI (which takes PMI into account) to explicitly correct this bias, and discourage the creation of such samples. Also, human annotation of machine-generated sentences/sentences pulled from existing texts instead of elicitation has been suggested to reduce such bias (Zhang et al., 2017). However, machine-generated text might look artificial, and work has shown that text generation has its own set of quality issues (Mathur et al., 2020). While we use AutoFix and TextFooler as modules to automatically transform samples, they are designed to be used in parallel with human sample creation. Their results can also be further modified by humans prior to submission. We see less reliance on these tools over the course of our user study, as discussed in Subsection 6.2. Additionally, previous work (Roit et al., 2019) in controlled dataset creation trains crowdworkers, and selects a subset of the best-performing crowdworkers for actual corpus creation. Each crowdworker's work is reviewed by another crowdworker, who acts as an analyst (as per our framework) of their samples. However, in real-world dataset creation, such training and selection phases might not be possible. Additionally, the absence of a metric-in-the-loop basis for feedback provided during training can potentially bias (through trainers) the created samples.

A.5 DQI Components

DQI shows the (i) the overall data quality and (ii) the impact of new data created on the overall quality. In this paper, higher quality implies lower artifact presence and higher generalization capability. DQI clubs artifacts into seven broad aspects of text, which cover the space of various possible interactions between samples in an NLP dataset. Please refer to (Mishra et al., 2020b) and (Mishra et al., 2022) for a full explanation of parameters.

A.6 Interface Design Intuitions

Careful Selection of Visualizations Prior to the design of test cases and a user interface, data visualizations highlighting the effects of sample addition are built. Considering the complexity of the formulas for the components of empirical DQI, we carefully select visualizations to help illustrate and analyze the effect to which individual text properties are affected.

A.6.1 Vocabulary

Which Characteristics of Data are Visualized? The contribution of samples to the size of the vocabulary is tracked using a dual axis bar chart. This displays the vocabulary size, along with the vocabulary magnitude, across the train, dev, and test splits for the dataset. By randomizing data splits, the distribution of vocabulary across dataset samples can clearly be identified. We use a dual axis chart as juxtaposition of the vocabulary magnitude against raw counts of words better reflects the evenness of the vocabulary distribution; it is not useful to have only a few samples contributing new words as other samples automatically become easy for the model to solve.

To further clarify the contribution of individual samples to vocabulary, the distribution of sentence lengths is plotted as a histogram. Each sample contributes two sentences, i.e., the premise and hypothesis statements. Figure 10 illustrates this. The histogram provides analysts with a frame of reference to identify gaps or outliers in the distribution, essential for determining which sentences are undesirable for the corpus due to extremely high (low inductive bias) or extremely low (high artifact–spurious bias) vocabulary contribution.

A.6.2 Inter-sample N-gram Frequency and Relation

Which Characteristics of Data are Visualized? There are different granularities of samples that are used to calculate the values of this component, namely: words, POS tags, sentences, bigrams, and trigrams. The granularities' respective frequency distributions and standard deviations are utilized for this calculation.

Bubble Chart for visualizing the frequency distribution: A bubble chart is used to visualize the frequency distribution of the respective granularity. This design choice is made in order to clearly view the contribution made by a new sample when added to the existing dataset in terms of different granularities. The bubbles are colored according to the bounds set for frequencies by the hyperparameters, and sized based on the frequency of the elements they represent. Additionally, some insight into variance can be obtained from this chart, by observing the variation in bubble size.

Bullet Chart for impact of new sample: The impact of sample addition on standard deviation can be viewed using the bullet chart. The bullet chart is useful to visually track performance against a target (in this case ideal standard deviation), displaying results in a single column; it looks like a thermometer and is therefore easy to follow. The red-yellow-green color bands for each granularity represent the standard deviation bounds of that granularity. The vertical black line represents the ideal value of the standard deviation of that granularity. The two horizontal bars represent the value of standard deviation before and after the new sample's addition. Figure 12 illustrates the visualization.

A.6.3 Inter-sample STS

Which Characteristics of Data are Visualized? The main units used in this DQI component are the similarity values between sentences across the dataset. This refers to either premise or hypothesis statements, relative to all other premise/hypothesis statements. In order to understand the similarity relations of sentences, a force layout and horizontal bar chart are used. This is illustrated in Figure 14.

Force Layout for Similar Sentence Pairs In the force layout, those sentence pairs with a similarity value that meets the minimum threshold are connected. Each node represents a sentence. The thickness of the connecting line depends on how close the similarity value is to the threshold. Similarity values are used to create this network, as the aim of this component is to drill down into whether a sample has sufficient inductive bias (i.e., is closely

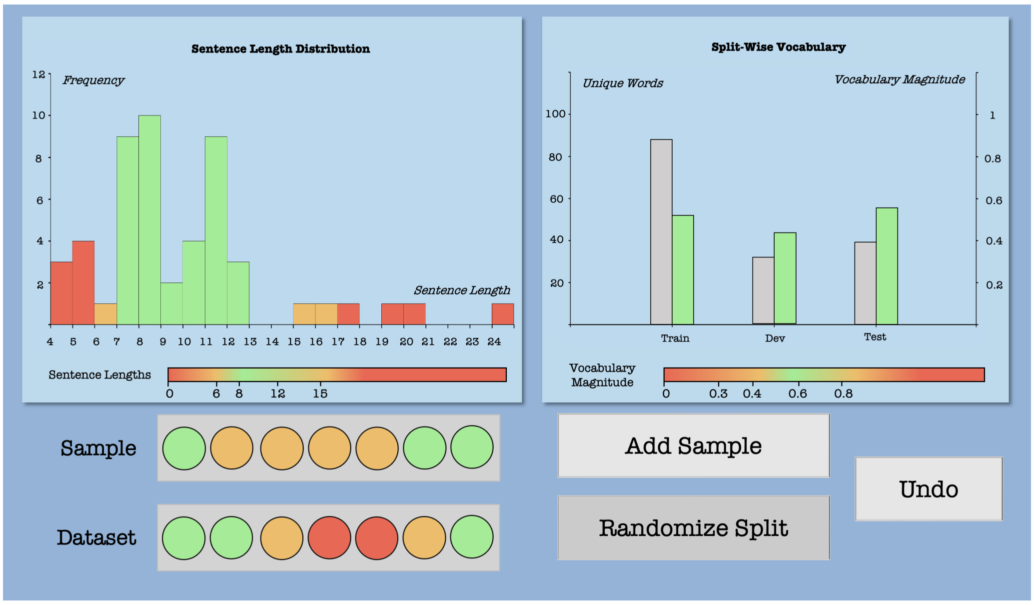


Figure 10: DQI_{c1} Visualization Prior to New Sample Addition

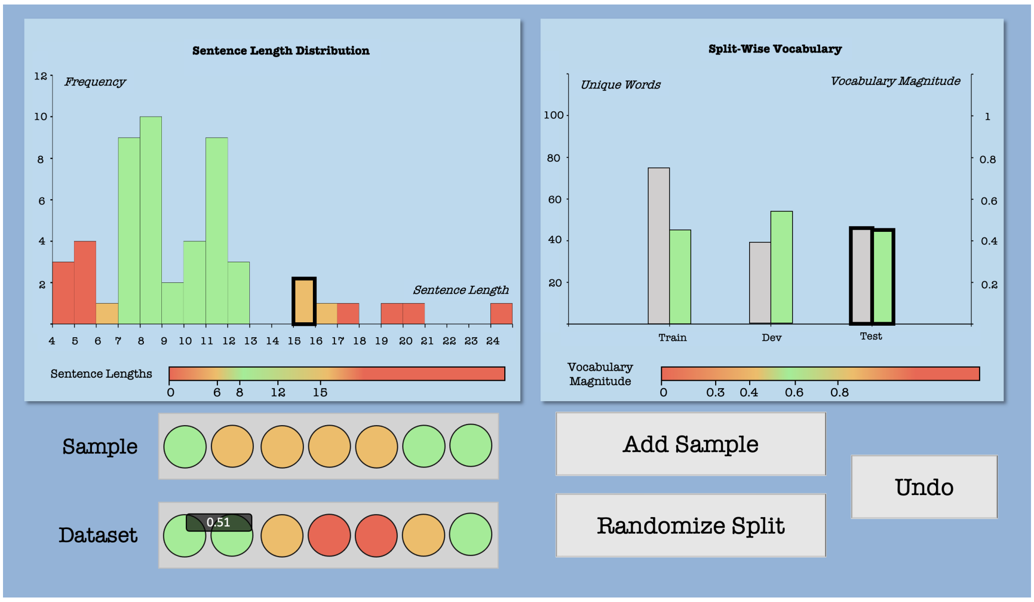


Figure 11: DQI_{c1} Visualization On New Sample Addition

linked to a sufficient extent to exiting samples), and also if a sample is too similar (spurious bias) to existing samples.

Horizontal Bar Chart for Most Similar Sentences In the horizontal bar chart, the sentences that are most similar to the given sentence are ordered in terms of their similarity value. The bar colors are centered around the threshold. This helps identify the most important subset to juxtapose the given sample against;the analyst can use this subset to for instance, decide if a moderate quality sample requires a small or big change in order to reacch acceptable quality..

A.6.4 Intra-sample Word Similarity

Which Characteristics of Data are Visualized? In this section, A sample's word similarity is viewed in terms of premise-only, hypothesis-only, and both. The relationship between non-adjacent words in the sample's sentences is analyzed specifically.

Overview Chart for Average Word Similarities and Heatmap for Single Sample The overview chart that is used is a tree map, which uses the average value of all word similarities per sample- i.e., concatenated premise and hypothesis- to color and group its components. This is illustrated in Figure

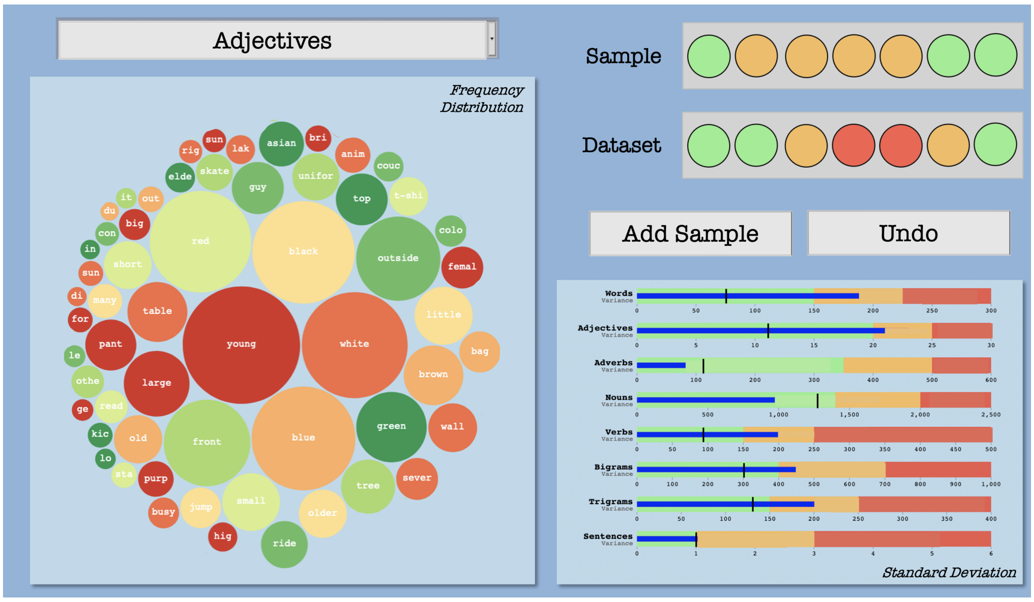


Figure 12: DQI_{c2} Visualization Prior to New Sample Addition

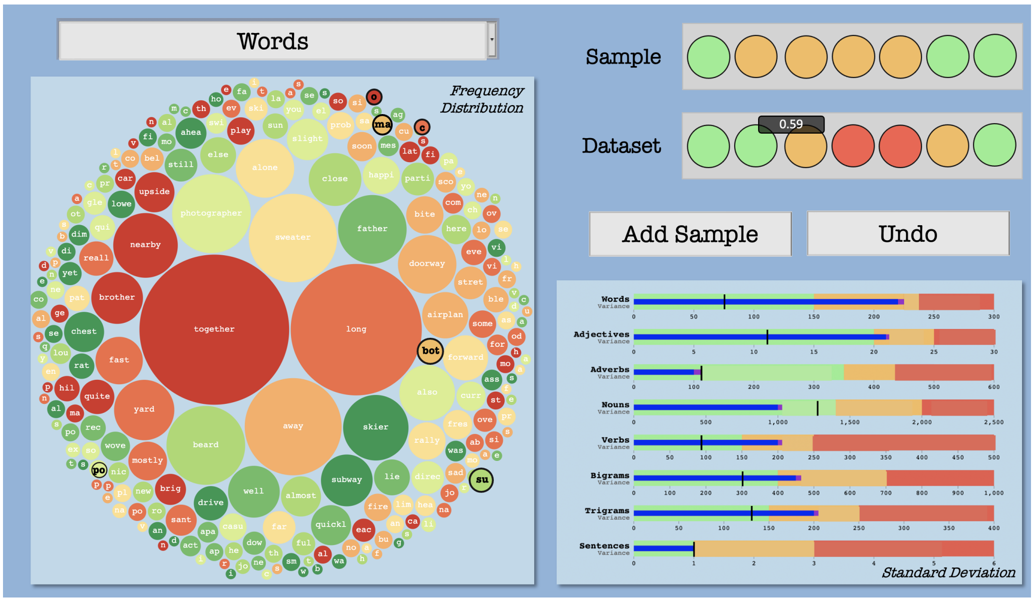


Figure 13: DQI_{c2} Visualization On New Sample Addition

16. Treemaps capture relative sizes of data categories, allowing for quick perception of the items that are large contributors to each category. This makes them ideal to analyze the inter-relationships between different word pairs across sample, in a concise manner.

The treemap also makes it easy to drill down into the specifics of a particular sample even further. This detailed view is provided in the form of a heatmap. All the words in a single sample, are plotted against each other, as shown in Figure 18. The heatmap provides a mechanism for word-level drill down of sample similarity. Like with the previous component, this helps provide the analyst with a frame of reference as to whether a moderate quality sample can be sent to TextFooler or not.

A.6.5 Intra-sample STS

Which Characteristics of Data are Visualized? Premise-Hypothesis similarity is analyzed on the basis of length variation, meeting a minimum threshold, and similarity distribution across the dataset. The first is addressed already in the vocabulary property by viewing the sentence length distribution. The other two are visualized using a histogram and kernel density estimation curve, as shown in Figure 19.

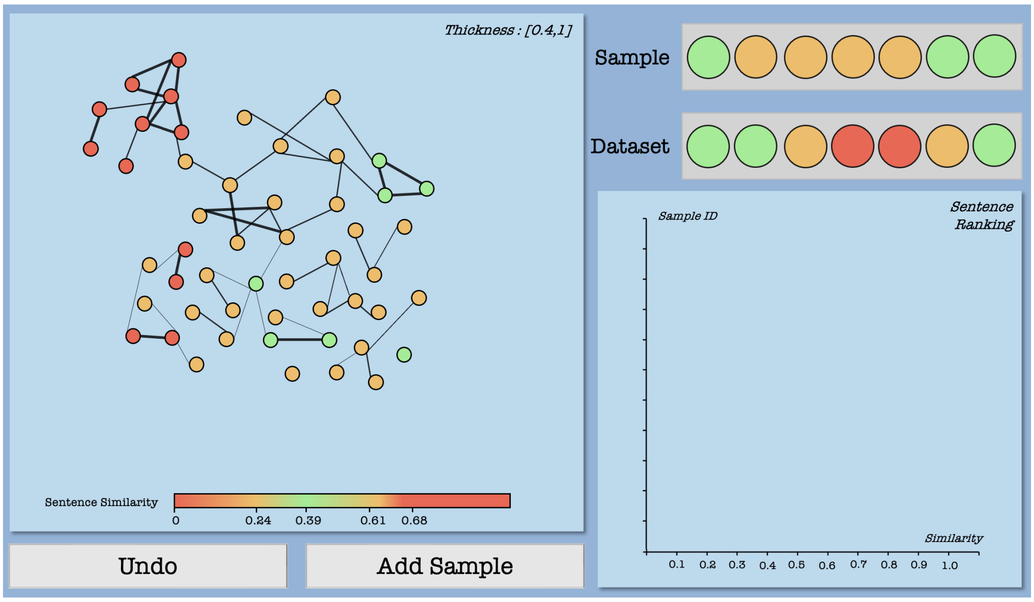
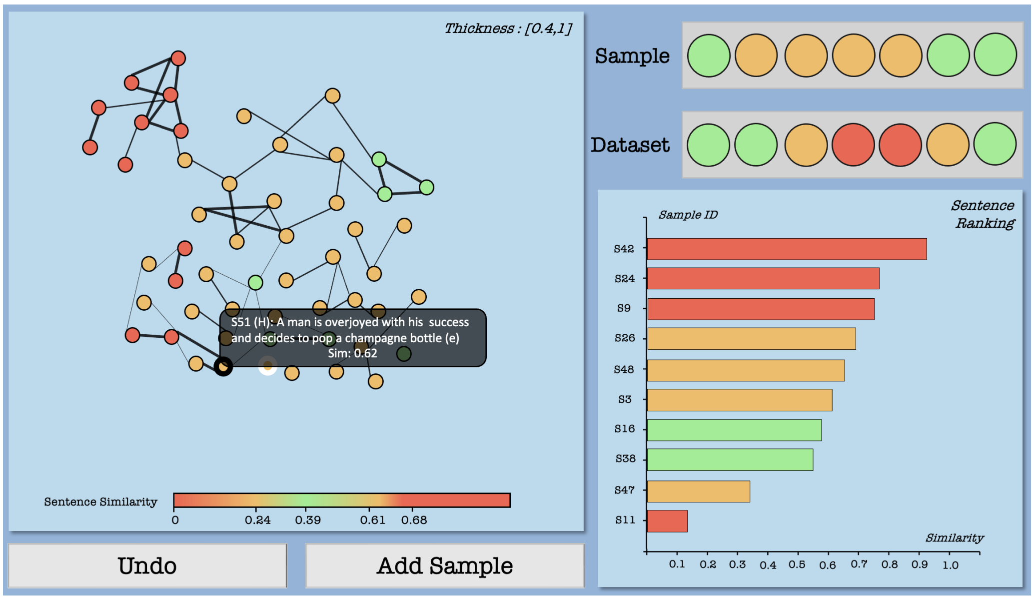


Figure 14: DQI_{c3} Visualization Prior to New Sample Addition



S51 (H): A man is overjoyed with his success and decides to pop a champagne bottle (e)
Sim: 0.62

Figure 15: DQI_{c3} Visualization On New Sample Addition

Histogram and Kernel Density Curve for Sample Distribution The histogram represents the distribution of the samples, and is colored by centering around the threshold as the ideal value. The number of bins can be changed, and therefore multilevel analysis can be conducted. This helps identify the proportion of samples that are of non-optimal range.

The kernel density curve is used to check for the overall skew of the distribution. KDE helps visualize the distribution sans user defied bins; this is colored to reflect the bi-linear color scale, and indicates the probability with which a sample will be correctly solved by a model given its similarity. This helps contextualize the level of artifacts relating to word overlap and sentence similarity in the sample.

A.6.6 N-Gram Frequency per Label

Which Characteristics of Data are Visualized? This component drills down on the second component, to view the patterns seen in granularities per label. There are two small multiples charts, divided based on label, used in this view- a violin plot and a box plot.

Violin plot and Kernel Density Curve for Skew of Distribution: The violin plots are structured to display both jittered points, according to their

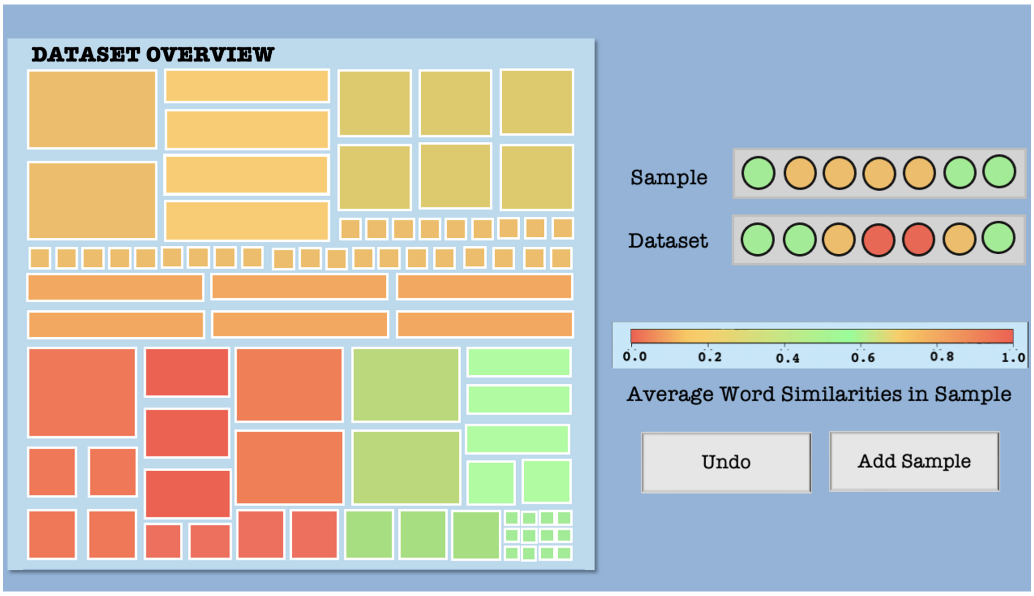


Figure 16: DQI_{c4} Visualization Prior to New Sample Addition

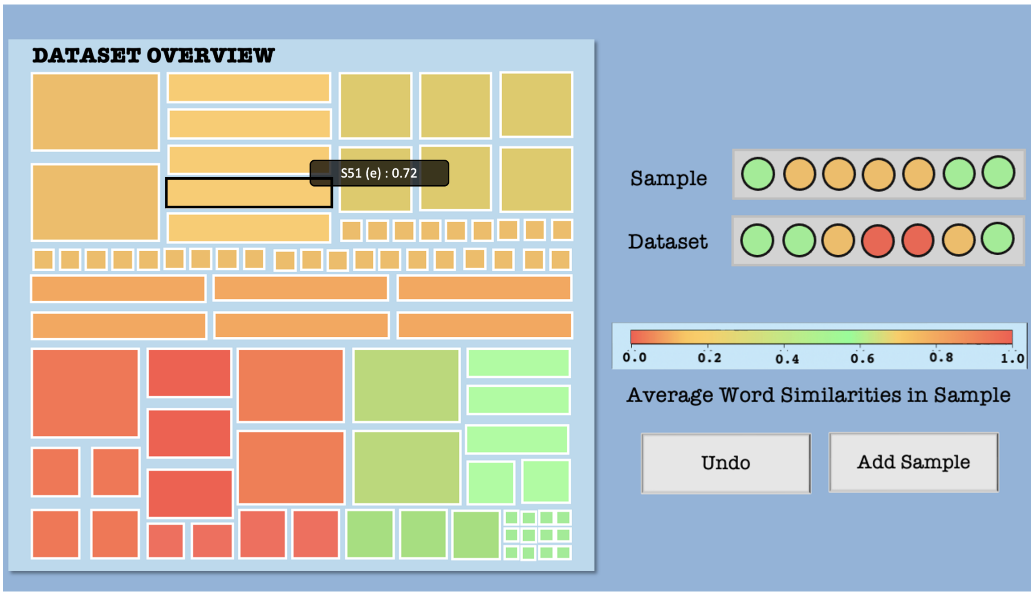


Figure 17: DQI_{c4} Visualization On New Sample Addition: Dataset View

frequency distribution, as well as a kernel density curve to judge the skew of the distribution. The points each represent an element of the granularity.

Box Plots for More Information The box plots are used to garner more information about the distribution, in terms of its min, max, median, mean, and inter quartile range. These help further characterize the distribution, as well as provide a quantitative definition of the skew seen using density curves. Jittered points representing elements are present in this plot as well.

A.6.7 Inter-split STS

Which Characteristics of Data are Visualized? Train-Test similarity must be kept minimal to prevent data leakage. This component's main feature is finding the train split sample that is most similar to a given test split sample.

Parallel Coordinate Graph for Train-Test Similarity: A subset of test and train samples, all found to have close similarity within their respective splits, and significant similarity across the splits are plotted as a one step parallel coordinate graph, with test samples along one axis, and train samples along the other. This subset is seeded

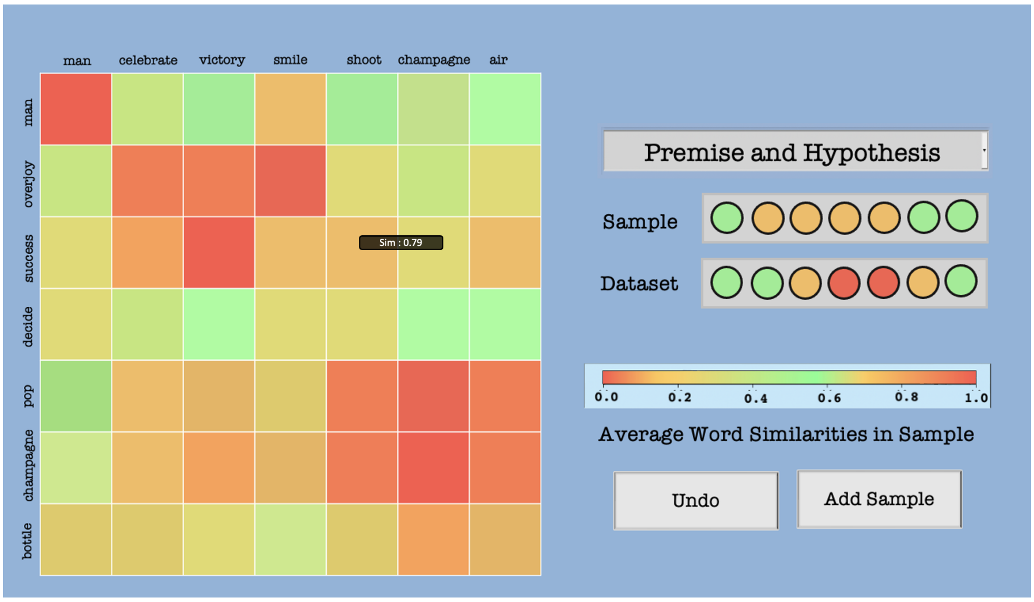
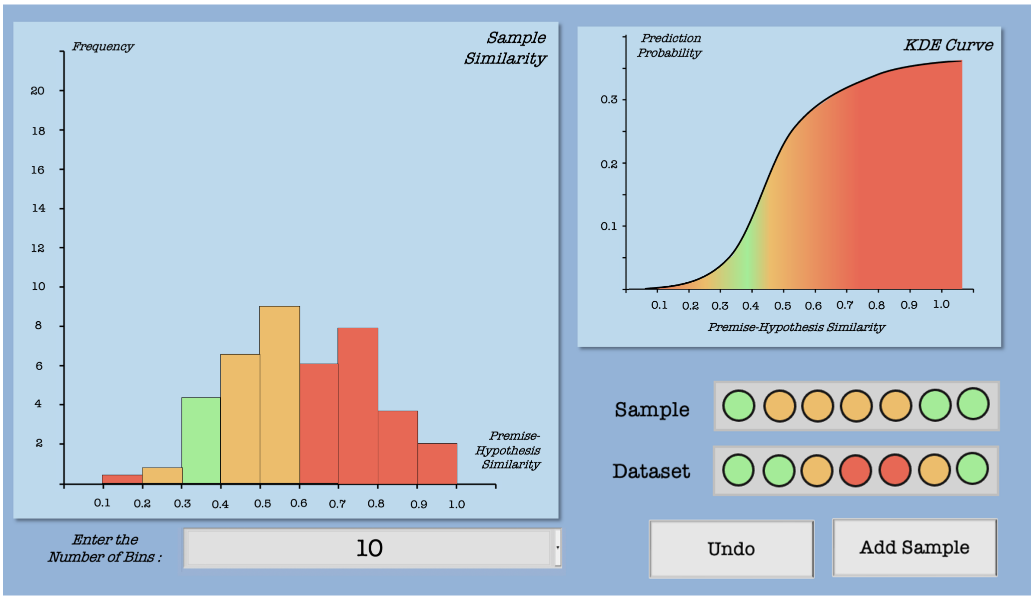


Figure 18: DQI_{c4} Visualization On New Sample Addition: Sample View



Enter the Number of Bins : 10

Figure 19: DQI_{c5} Visualization Prior to New Sample Addition

with those samples closest in similarity to the new sample to be introduced, based on the third component's visualization. The links connecting points on the two axes are drawn between the most similar matches across the split, as shown in Figure 24.

A.7 AutoFix and TextFooler Examples

See Tables 4, 5.

A.8 User Study

AutoFix Suggestions: See Tables 6, 5.

NASA TLX: The NASA Task Load Index (NASA-TLX) is a subjective, multidimensional assessment tool that rates perceived workload in order to assess a task, system, or team's effectiveness or other aspects of performance (Hart, 2006).

NASA-TLX divides the total workload into six subjective subscales that are represented on a single page. There is a description for each of these subscales that the subject should read before rating. They rate each subscale within a 100-point range, with 5-point steps, as shown in Figure 28. Providing descriptions for each measurement can be found to help participants answer accurately (Schuff et al., 2011). The descriptions are as follows:

- **Mental Demand:** How much mental and perceptual activity was required? Was the

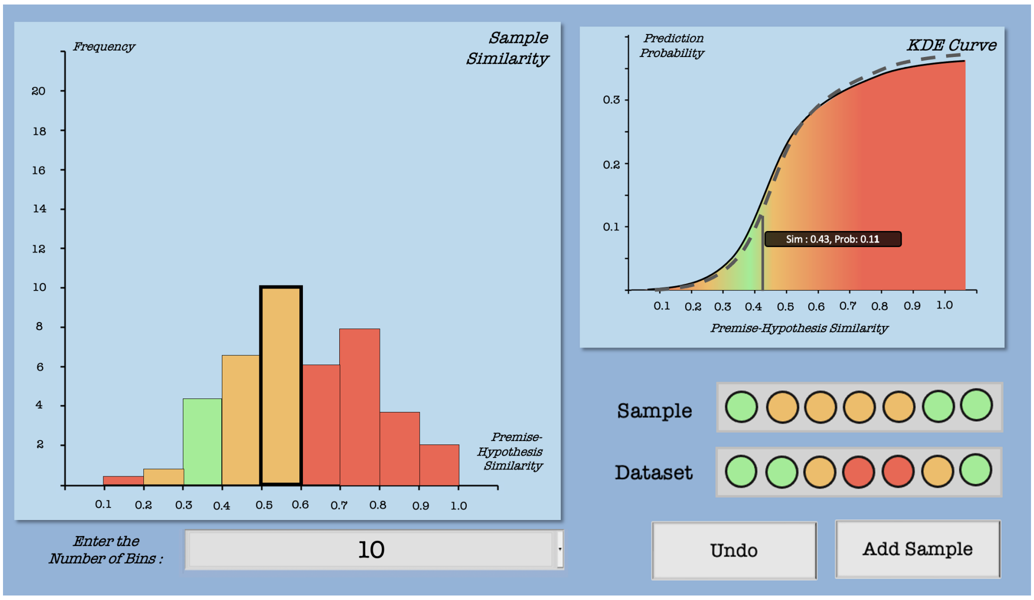


Figure 20: DQI_{c5} Visualization On New Sample Addition

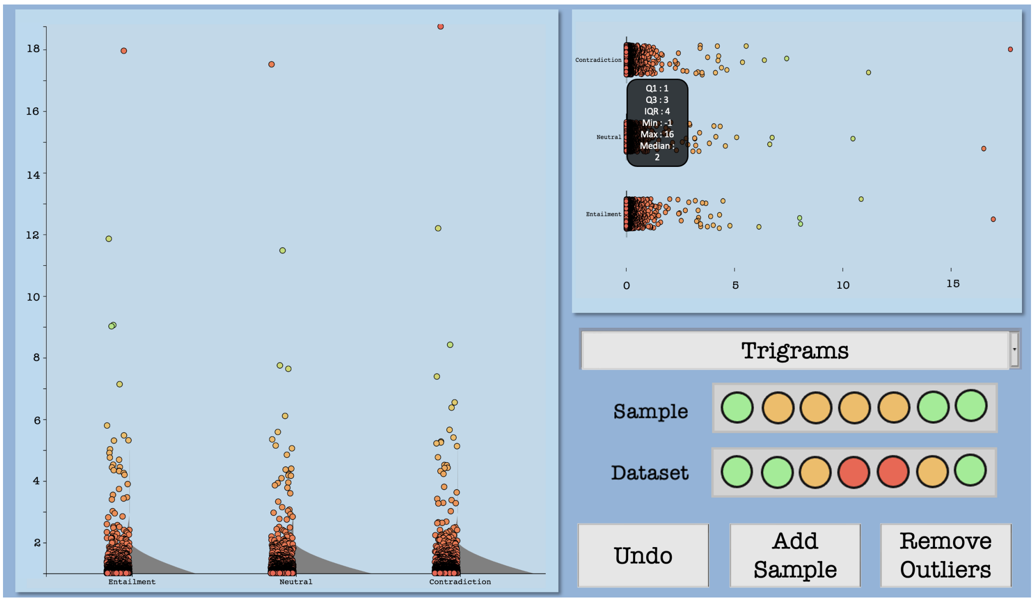


Figure 21: DQI_{c6} Visualization Prior to New Sample Addition

task easy or demanding, simple or complex?

- **Physical Demand:** How much physical activity was required? Was the task easy or demanding, slack or strenuous?
- **Temporal Demand:** How much time pressure did you feel due to the pace at which the tasks or task elements occurred? Was the pace slow or rapid?
- **Performance:** How successful were you in performing the task? How satisfied were you with your performance?
- **Effort:** How hard did you have to work (mentally and physically) to accomplish your level of performance?
- **Frustration:** How irritated, stressed, and annoyed versus content, relaxed, and complacent did you feel during the task?

We record participant demographics– age, gender, and occupation. We also ask participants to rate their familiarity with Visualization and NLP, on a scale of 1 (novice) to 5 (expert). Demographic information is shown in Figure 29. Participants are asked to fill this form at the end of each round of the user study. We also record the number of questions participants successfully create, as well as a

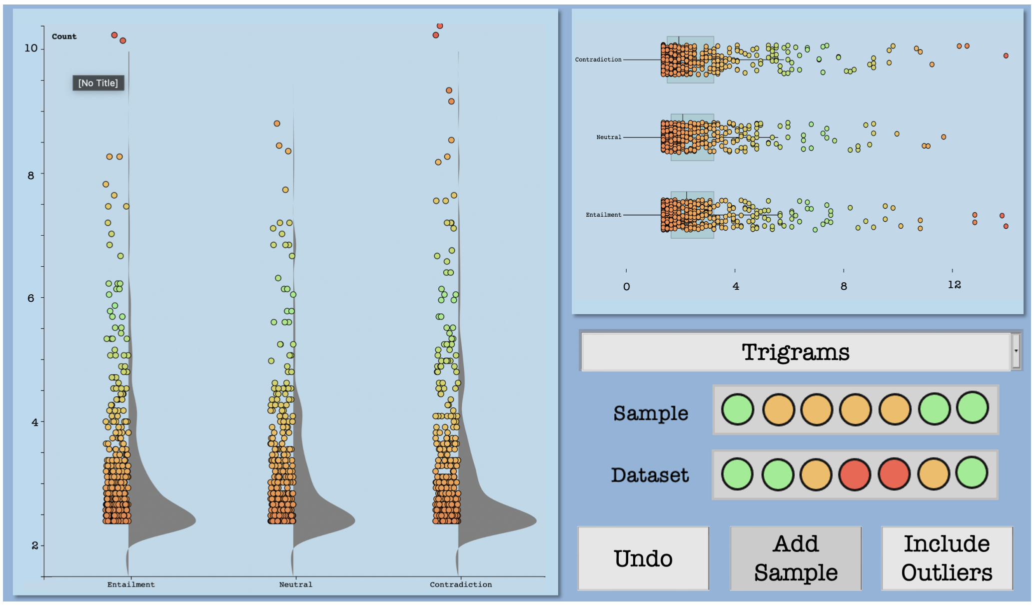


Figure 22: DQI_{c6} Visualization after removing outliers Prior to New Sample Addition

record of how often participants use each module in the full system round. At the end of the user study, participants are asked what their impression of data quality is, and their free response is recorded.

Subscale Wise Results: Individual results of the averaged subscales in Figure 8 are shown in Figures 30,31. Physical demand does not change significantly across user study rounds.

A.9 Expert and User Comments

Experts *(P)*: We present an initial prototype of our tool, to a set of three researchers with expertise in NLP and knowledge of data visualization, in order to judge the interface design. For each expert, the crowdworker interface and then analyst interfaces were demoed. Participants (P) could ask questions and make interaction/navigation decisions to facilitate a natural user experience. All the experts appreciated the easily interpretable traffic-signal color scheme and found the organization of the interfaces—providing separate detailed views within the analyst workflow– a way to prevent cognitive overload (too much information on one screen); P_2 said the latter *"… enhances readability for understanding the data at different granularities."*. P_1 suggested the inclusion of *"… a provenance module within the analyst views to show historical sample edits and overall data quality changes over time to understand how data quality evolves as the benchmark size increases… this would help with the bubble plot and tree map which will get more cluttered and complex as data size increases"*. Additionally P_3 remarked that *"The frequency of samples of middling quality should increase as benchmark size increases, but the initial exposure that analysts will have with higher or lower quality samples should lessen the learning curve as they are familiar enough with interface subtleties by the time they begin to encounter more challenging cases."*

Crowdworkers *(C)*: When presented with traffic signal feedback, crowdworkers report that the time and effort required to create high quality samples increases–*"You need to keep redoing the sample since when you see it's all red, you know it's probably not going to be accepted"*(C_3); however, they are more confident about their performance and sample quality *"...when there's green, I know I've done it right, and it cuts down on my having to create a lot of samples to get paid"* (C_{15}). We find that AutoFix usage [7] causes an unexpected increase in mental and temporal demand, as well as frustration; we attribute this to observed user behavior– *"I'm not sure how much I trust this recommendation without seeing the colors"*(C_{12}), and *"I'd prefer to change a couple of things since I can't see the feedback anymore*(C_{21}). The drastic improvement over all aspects (highest for frustration) in the case of using the full system is in line with this observation–*"This is so easy, I can create samples really fast, and I have a better chance of getting more accepted."*(C_8) and *"Now that I get the feedback along with the recommendation, I can see*

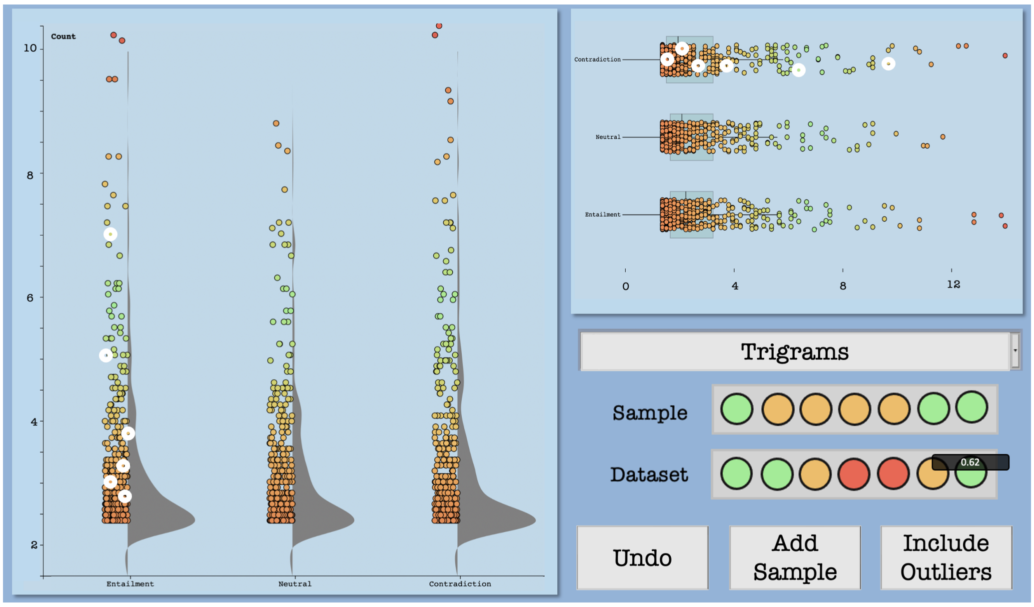


Figure 23: DQI_{c6} Visualization with mouseover On New Sample Addition

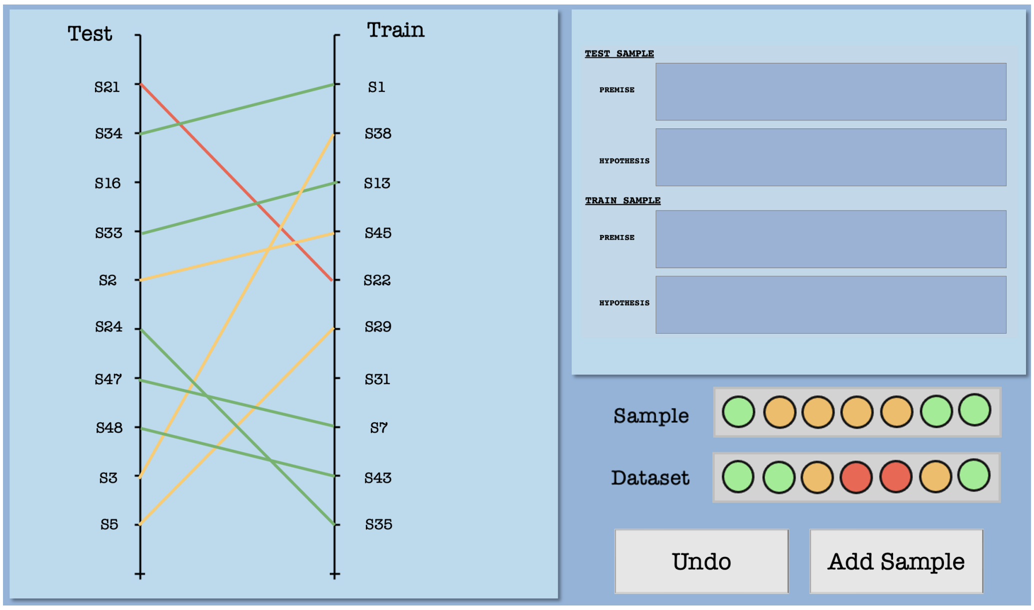
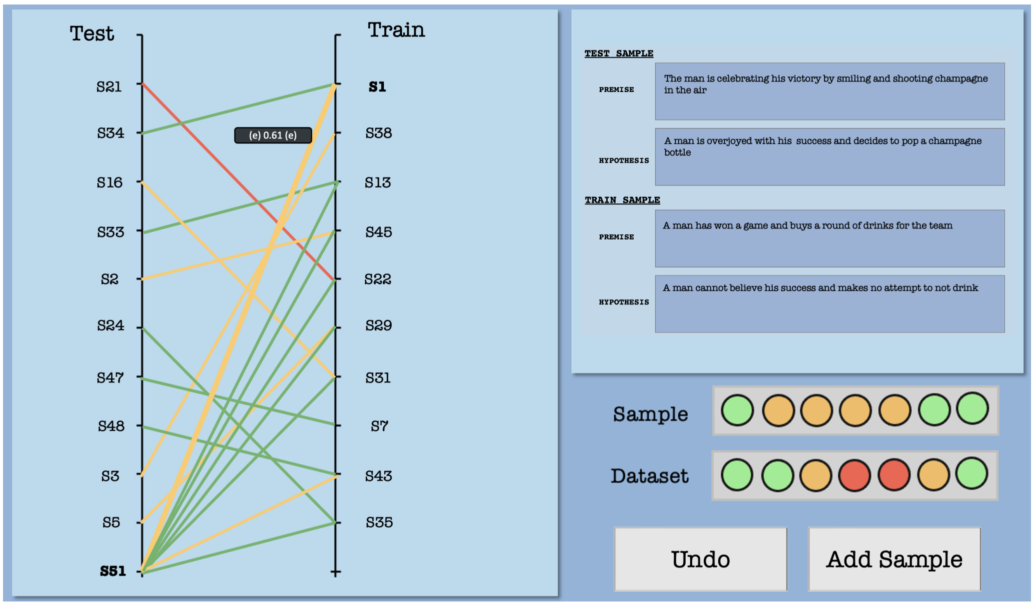


Figure 24: DQI_{c7} Visualization Prior to New Sample Addition

the quality improvement. So using the recommendation is now definitely faster."(C_{12}). The number of questions created per round as well as system scores also follows this trend, across all types of crowdworkers.

Summary: Traffic signal feedback initially increases time (+25%) and effort (+60%) required to create high quality samples, as users have to correct them. However they are more confident (performance– +27%) of sample quality. AutoFix usage causes an unexpected increase in effort (+5%) and frustration (+88.8%), as users do not fully trust recommendations without visual feedback. The drastic improvement over all aspects (frustration– -44.4%, mental demand– -38.1%, temporal demand– -29.1%, effort– -20%, average decrease in difficulty– -31.1%, performance– +34.6%) in the case of using the full system is in line with this observation. The number of questions created per round (traffic signal– -8.3%, AutoFix– +25%, full system– +83.3%) as well as system scores (traffic signal– +27.3%, AutoFix– +13.6%, full system– +54.5%) also follows this trend, across all types of crowdworkers.

Analysts *(A)*: In the case of direct quality feedback, i.e., traffic signals, analysts report an in-



TEST SAMPLE

PREMISE: The man is celebrating his victory by smiling and shooting champagne in the air

HYPOTHESIS: A man is overjoyed with his success and decides to pop a champagne bottle

TRAIN SAMPLE

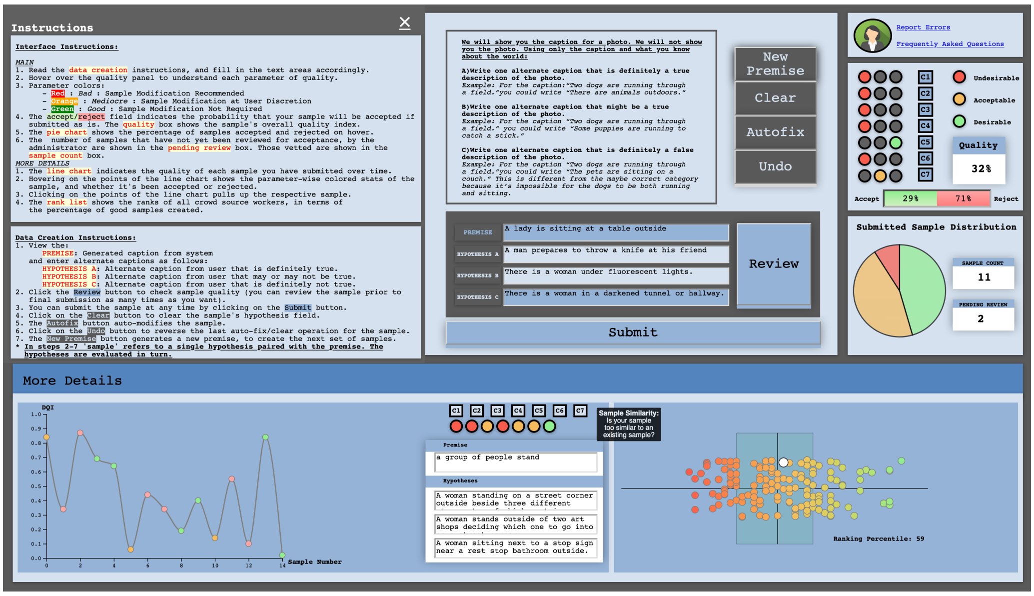
PREMISE: A man has won a game and buys a round of drinks for the team

HYPOTHESIS: A man cannot believe his success and makes no attempt to not drink

Figure 25: DQI_{c7} Visualization On New Sample Addition

creased performance and find the task easier–*"... it's easier to directly choose based on quality... and it takes care of typos too, the typo samples are marked down so the work goes pretty fast"*(A_3). When analysts are shown the visualization interfaces, they are explicitly taught to differentiate the traffic signal colors in the visualizations as being indicative of how the sample affects the overall dataset quality, i.e., the colors in different component views represent individual terms of the components calculated over the whole dataset (analysts can toggle between the states of original dataset and new sample addition). We find that users initially find this more difficult to do– *"It takes a little time to figure out how to go through the views. I learned that in the samples I looked at, components three and seven seemed to be linked. So I'd look at those first the next time I used the system"* (A_6) and *"... it takes me some time to figure out how to read the interfaces effectively, but it does make me more secure in judging sample quality at multiple granularities and that would help if I was doing this for a particular application"*(A_1). Analysts averaged behavior on TextFooler models the conventional approach quite closely, as analysts are seen to have a tendency to either– *"... deciding to reject or repair is difficult when you don't have the sample or dataset feedback... and what if the repaired sample still isn't good enough?"*(A_4), or– *" I like having this option to repair... I don't need to waste time on analyzing something that isn't outright an accept or reject, I can send it to be repaired and come back to it later"*(A_8). When shown the full system, analysts also report improvement in all aspects, particularly mental demand and performance–*"I can be sure of not having to redo things since it's likely that I will be able to get a low hypothesis baseline using this system"*(A_2, A_1). The visualization usage also improves– *"... I went to component three right off the bat this time, I knew that I could look at the linked components..."* (A_6). Altogether, sample evaluation by analysts increases, following this trend, and analysts are more assured of their performance.

Summary: Analysts find the task easier (effort– -19.3%, performance– +26.9%) with traffic signal feedback, as quality is clearly marked. When analysts are shown the visualization interfaces, they are explicitly taught to differentiate how the traffic signal colors in the visualizations indicate a sample's effect on the overall dataset quality. Analysts can toggle between the states of original dataset and new sample addition. We find that analysts initially find toggling more difficult to do (mental demand– +15.4%, temporal demand– +36.4%, frustration– 3.5%), though they agree that it improves their judgement of quality (performance– +15.9%). Analysts' average behavior on TextFooler models the conventional approach quite closely, as analysts are seen to have a tendency to send all samples that are unclear to TextFooler immediately. With the full system, analysts also report improvement in all aspects (average decrease in difficulty– -14.3%), particularly mental demand (-19.2%) and performance



Interface Instructions:

MAIN
1. Read the data creation instructions, and fill in the text areas accordingly.
2. Hover over the quality panel to understand each parameter of quality.
3. Parameter colors:
   - Red : *Bad* : Sample Modification Recommended
   - Orange : *Mediocre* : Sample Modification at User Discretion
   - Green : *Good* : Sample Modification Not Required
4. The accept/reject field indicates the probability that your sample will be accepted if submitted as is. The quality box shows the sample's overall quality index.
5. The pie chart shows the percentage of samples accepted and rejected on hover.
6. The number of samples that have not yet been reviewed for acceptance, by the administrator are shown in the pending review box. Those vetted are shown in the sample count box.

MORE DETAILS
1. The line chart indicates the quality of each sample you have submitted over time.
2. Hovering on the points of the line chart shows the parameter-wise colored stats of the sample, and whether it's been accepted or rejected.
3. Clicking on the points of the line chart pulls up the respective sample.
4. The rank list shows the ranks of all crowd source workers, in terms of the percentage of good samples created.

Data Creation Instructions:
1. View the:
   PREMISE: Generated caption from system
   and enter alternate captions as follows:
   HYPOTHESIS A: Alternate caption from user that is definitely true.
   HYPOTHESIS B: Alternate caption from user that may or may not be true.
   HYPOTHESIS C: Alternate caption from user that is definitely not true.
2. Click the Review button to check sample quality (you can review the sample prior to final submission as many times as you want).
3. You can submit the sample at any time by clicking on the Submit button.
4. Click on the Clear button to clear the sample's hypothesis field.
5. The Autofix button auto-modifies the sample.
6. Click on the Undo button to reverse the last auto-fix/clear operation for the sample.
7. The New Premise button generates a new premise, to create the next set of samples.

* **In steps 2-7 'sample' refers to a single hypothesis paired with the premise. The hypotheses are evaluated in turn.**

We will show you the caption for a photo. We will not show you the photo. Using only the caption and what you know about the world:

A)Write one alternate caption that is definitely a true description of the photo.
Example: For the caption:"Two dogs are running through a field."you could write "There are animals outdoors."

B)Write one alternate caption that might be a true description of the photo.
Example: For the caption "Two dogs are running through a field." you could write "Some puppies are running to catch a stick."

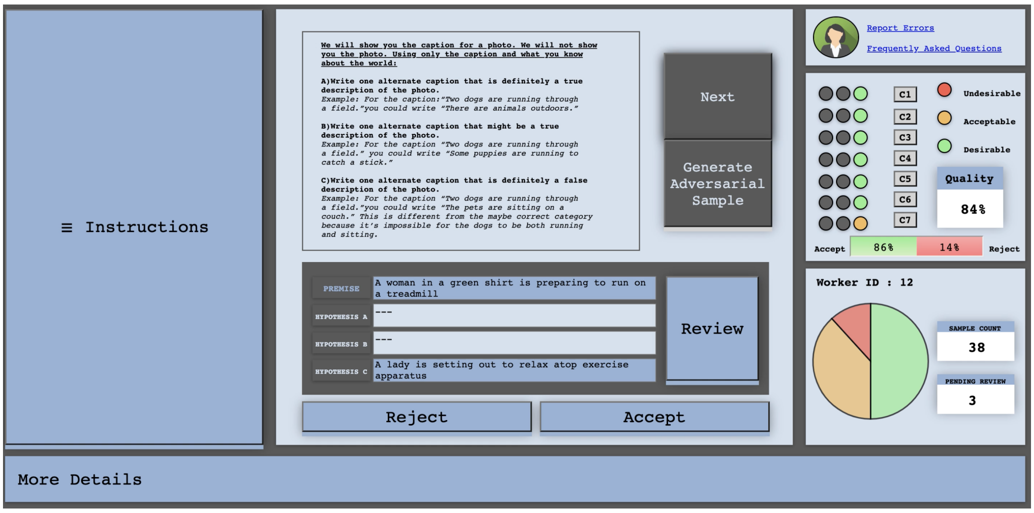
C)Write one alternate caption that is definitely a false description of the photo.
Example: For the caption "Two dogs are running through a field."you could write "The pets are sitting on a couch." This is different from the maybe correct category because it's impossible for the dogs to be both running and sitting.

PREMISE: A lady is sitting at a table outside
HYPOTHESIS A: A man prepares to throw a knife at his friend
HYPOTHESIS B: There is a woman under fluorescent lights.
HYPOTHESIS C: There is a woman in a darkened tunnel or hallway.

Premise
a group of people stand

Hypotheses
A woman standing on a street corner outside beside three different
A woman stands outside of two art shops deciding which one to go into
A woman sitting next to a stop sign near a rest stop bathroom outside.

Figure 26: Crowdworker-View



We will show you the caption for a photo. We will not show you the photo. Using only the caption and what you know about the world:

A)Write one alternate caption that is definitely a true description of the photo.
Example: For the caption:"Two dogs are running through a field."you could write "There are animals outdoors."

B)Write one alternate caption that might be a true description of the photo.
Example: For the caption "Two dogs are running through a field." you could write "Some puppies are running to catch a stick."

C)Write one alternate caption that is definitely a false description of the photo.
Example: For the caption "Two dogs are running through a field."you could write "The pets are sitting on a couch." This is different from the maybe correct category because it's impossible for the dogs to be both running and sitting.

PREMISE: A woman in a green shirt is preparing to run on a treadmill
HYPOTHESIS A: ---
HYPOTHESIS B: ---
HYPOTHESIS C: A lady is setting out to relax atop exercise apparatus

Worker ID : 12

Figure 27: Analyst View

(+30.8%), considering that the system increases the likelihood of a low hypothesis baseline. The visualization usage also improves, as analysts learn component relationships. Altogether, sample evaluation by analysts increases (full system– +83.3%), following this trend, and analysts are more assured of their performance (full system score– +94.1%).

Task	**Description**	Component
New Sample	Adds the sample under review to dataset and updates visualizations.	All
Undo	Removes sample under review from dataset and updates visualizations.	All
Randomize Split	Randomized re-sampling of data across splits in a 70:10:20 ratio.	Vocabulary
Undo Split	Reverses last random split generated.	Vocabulary
Save Split	Freezes split for the remainder of analysis.	Vocabulary
Changing Granularity	View granularity can be changed by selecting drop down option.	Inter-sample N-gram Frequency and Relation, N-Gram Frequency per Label
Change Heat Map View	Using the drop down, the heatmap shows word similarities for the (a) premise, (b) hypothesis, or (c) both sentences.	Intra-sample Word Similarity
Rebinning Histogram	By filling a new value in the textbox, the number of bins in the histogram changes to that value.	Intra-sample STS
Remove Outliers	Removes elements with frequency count less than median count of granularity being viewed.	N-Gram Frequency per Label
Include All Samples	Displays all elements for a granularity.	N-Gram Frequency per Label

Table 3: Task Descriptions for Visual Interfaces

Premise	**Orig. Hypothesis**	**DQI**	**Suggested Words**	**New Hypothesis based on suggestions**	**New DQI**
A woman, in a green shirt, preparing to run on a treadmill.	A woman is preparing to sleep on a treadmill	2.4650170	preparing,sleep	A woman is organizing to rest on a treadmill	2.5275722
The dog is catching a treat	The cat is not catching a treat	2.752542	catching	the cat is not getting a treat	3.6909140
Three young men are watching a tennis match on a large screen outdoors	Three young men watching a tennis match on a screen outdoors, because their brother is playing	2.6435402891414217	young,watching, playing	Three youthful men observing a tennis match on a screen outdoors, because their brother is performing.	2.6787982

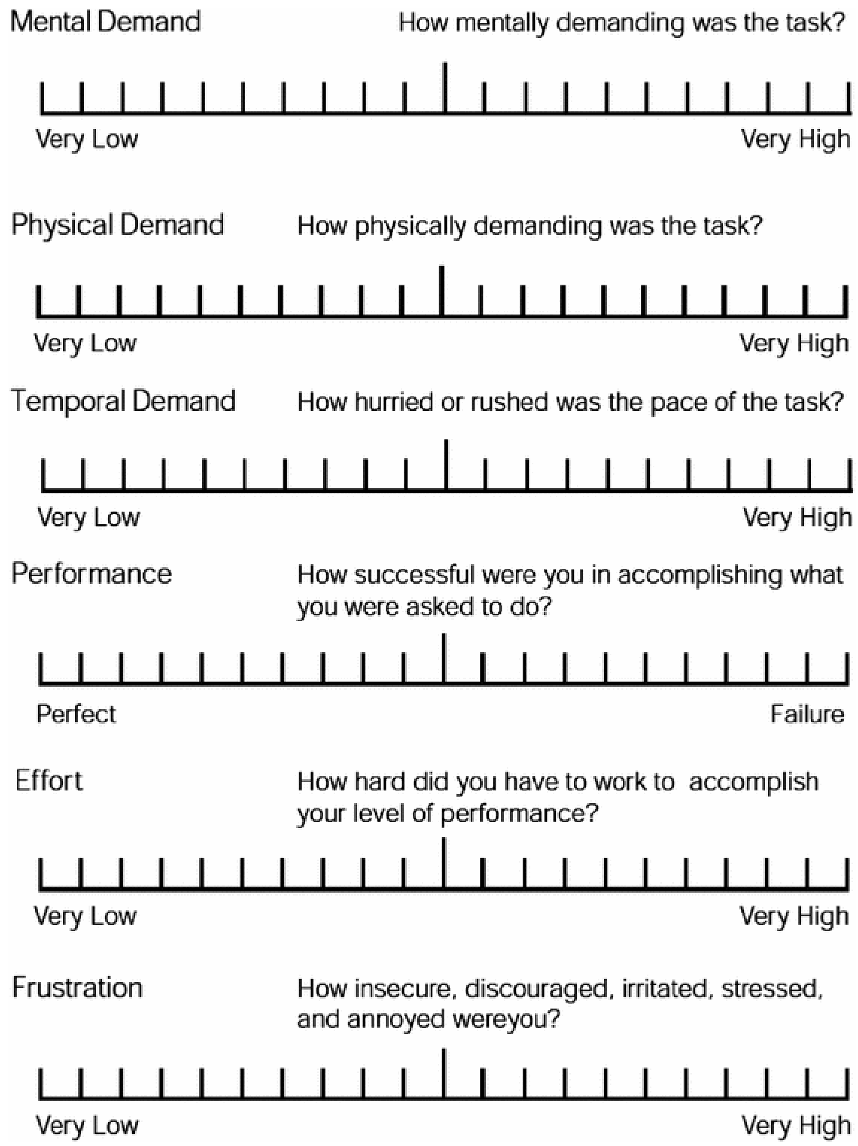
Table 4: A few samples for Autofix with Intra Sample STS in DQI

Premise	**Orig. Hypothesis**	**DQI**	**New Hypothesis**	**New DQI**	**Label**
A woman and a man sweeping the sidewalk.	The couple is sitting down for dinner.	2.416	The couple is meeting for dinner.	3.479	Contradiction
A woman enjoying the breeze of a primitive fan.	The woman has a fan.	2.127	The woman owns a fan.	2.733	Entailment
There is a man in tan lounging outside in a chair.	A man is preparing for vacation.	2.801	A man is arranging to take a vacation.	3.502	Neutral

Table 5: Examples for TextFooler, with DQI's Intra-sample STS values for existing SNLI samples.

Premise	**Orig. Hypothesis**	**DQI**	**Suggested Words**	**New Hypothesis based on suggestions**	**New DQI**
A woman, in a green shirt, preparing to run on a treadmill.	A woman is preparing to sleep on a treadmill	2.4650170	preparing,sleep	A woman is organizing to rest on a treadmill	2.5275722
The dog is catching a treat	The cat is not catching a treat	2.752542	catching	the cat is not getting a treat	3.6909140
Three young men are watching a tennis match on a large screen outdoors	Three young men watching a tennis match on a screen outdoors, because their brother is playing	2.6435402891414217	young,watching, playing	Three youthful men observing a tennis match on a screen outdoors, because their brother is performing.	2.6787982
A man in a green apron smiles behind a food stand	A man smiles	3.2367785	smiles	A person is grinning.	6.303777

Table 6: A few samples for Autofix with ISSTS in DQI



Mental Demand How mentally demanding was the task?

Very Low Very High

Physical Demand How physically demanding was the task?

Very Low Very High

Temporal Demand How hurried or rushed was the pace of the task?

Very Low Very High

Performance How successful were you in accomplishing what you were asked to do?

Perfect Failure

Effort How hard did you have to work to accomplish your level of performance?

Very Low Very High

Frustration How insecure, discouraged, irritated, stressed, and annoyed wereyou?

Very Low Very High

Figure 28: NASA TLX Form

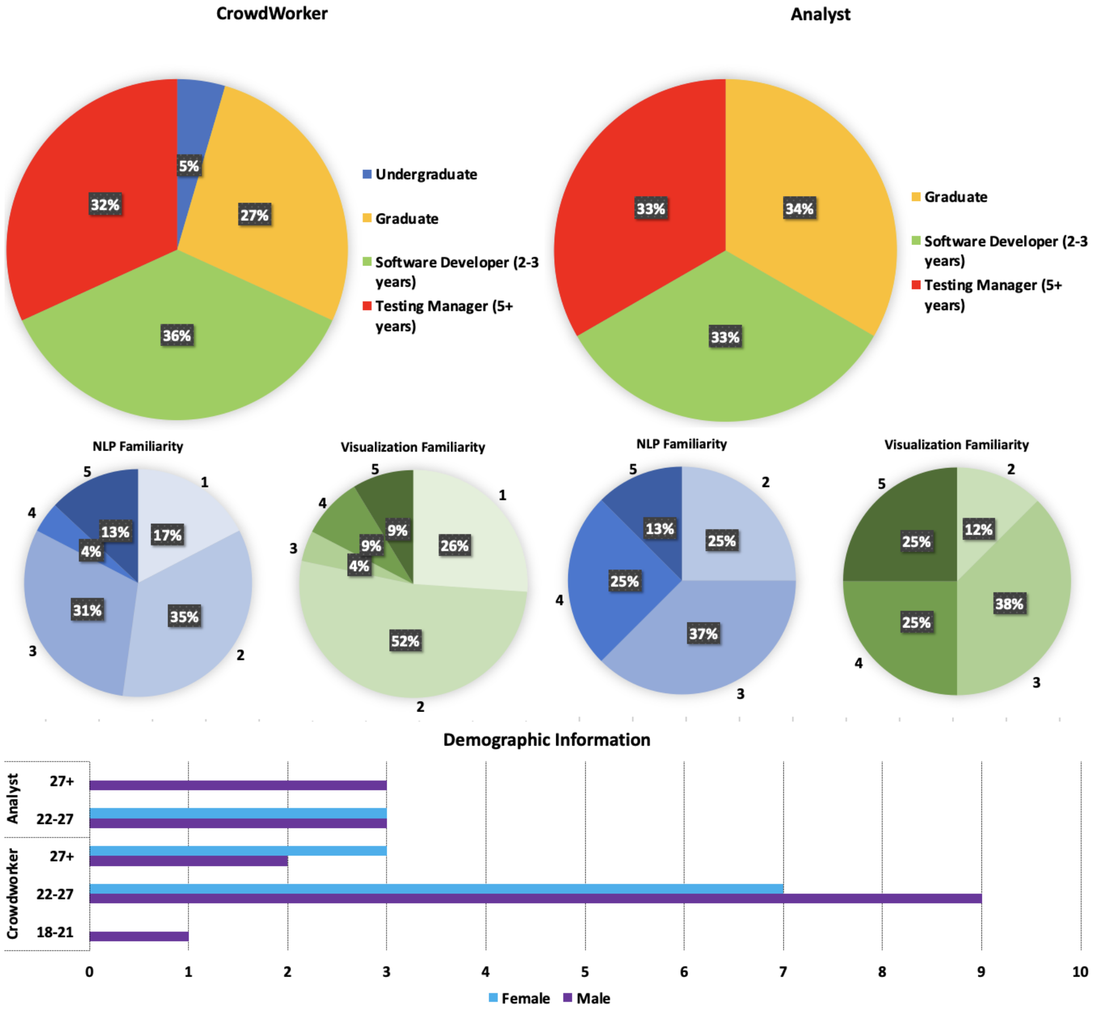


Figure 29: Demographic information for the User Study

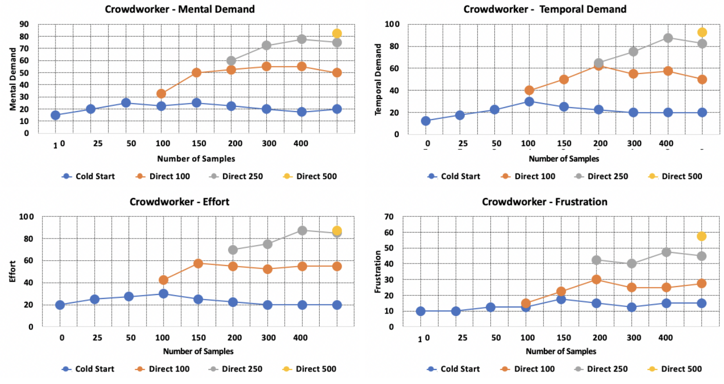


Figure 30: NASA TLX– Crowdworker Subscale Results

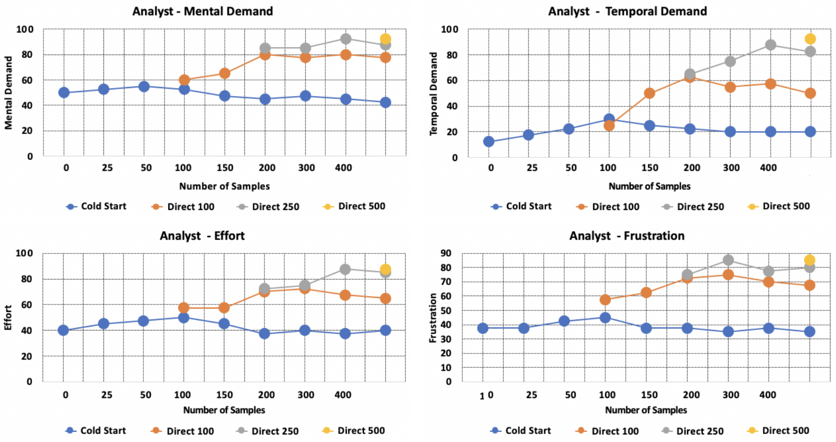


Figure 31: NASA TLX– Analyst Subscale Results

COMPS: Conceptual Minimal Pair Sentences for testing Robust Property Knowledge and its Inheritance in Pre-trained Language Models

Kanishka Misra
Purdue University
kmisra@purdue.edu

Julia Rayz
Purdue University
jtaylor1@purdue.edu

Allyson Ettinger
University of Chicago
aettinger@uchicago.edu

Abstract

A characteristic feature of human semantic cognition is its ability to not only store and retrieve the properties of concepts observed through experience, but to also facilitate the inheritance of properties (*can breathe*) from superordinate concepts (ANIMAL) to their subordinates (DOG)—i.e. demonstrate *property inheritance*. In this paper, we present COMPS, a collection of English minimal pair sentences that jointly tests pre-trained language models (PLMs) on their ability to attribute properties to concepts and their ability to demonstrate property inheritance behavior. Analyses of 22 different PLMs on COMPS reveal that they can easily distinguish between concepts on the basis of a property when they are trivially different, but find it relatively difficult when concepts are related on the basis of nuanced knowledge representations. Furthermore, we find that PLMs can show behaviors suggesting successful property inheritance in simple contexts, but fail in the presence of distracting information, which decreases the performance of many models sometimes even below chance. This lack of robustness in demonstrating simple reasoning raises important questions about PLMs' capacity to make correct inferences even when they appear to possess the prerequisite knowledge.

1 Introduction

The ability to learn, update and deploy one's knowledge about concepts (ROBIN, CHAIR) and their properties (*can fly*, *can be sat on*), observed during everyday experience is fundamental to human semantic cognition (Murphy, 2002; Rogers and McClelland, 2004; Rips et al., 2012). Knowledge of a concept's properties, combined with the ability to infer the `IsA` relation (Sloman, 1998; Murphy, 2003) leads to an important behavior known as *property inheritance* (Quillian, 1967; Smith and Estes, 1978; Murphy, 2002), where subordinates of a concept inherit its properties. For instance, one is likely to infer that an entity called *luna* can meow, has a tail, is a mammal, etc., even if *all* they know is that it is a cat. The close connection between a word's meaning and its conceptual representation makes these abilities crucial to language understanding (Murphy, 2002; Lake and Murphy, 2021), making it critical for computational models of language processing to also exhibit behavior consistent with these capacities. Indeed, modern pre-trained language models (PLMs; Devlin et al., 2019; Brown et al., 2020, etc.) have made impressive empirical strides in eliciting general knowledge about real world concepts and entities (Petroni et al., 2019; Weir et al., 2020, *i.a.*), as well as in demonstrating isomorphism with real world abstractions like direction and color (Abdou et al., 2021; Patel and Pavlick, 2022), often times without even having been explicitly trained to do so. At the same time, their ability to robustly demonstrate such capacities has recently been called to question, owing to failures due to reporting bias (Gordon and Van Durme, 2013; Shwartz and Choi, 2020), lack of consistency (Elazar et al., 2021; Ravichander et al., 2020), and sensitivity to lexical cues (Kassner and Schütze, 2020; Misra et al., 2020; Pandia and Ettinger, 2021).

In this work, we cast further light on PLMs' ability to robustly demonstrate knowledge about concepts and their properties. To this end, we introduce Conceptual Minimal Pair Sentences (COMPS), a collection of English minimal pair sentences, where each pair attributes a property (*can fly*) to two noun concepts: one which actually possesses the property (ROBIN), and one which does not (PENGUIN). Following standard practice in the minimal pairs evaluation paradigm (Warstadt et al., 2020, etc.), we test whether PLMs prefer sentence stimuli expressing correct property knowledge over those expressing incorrect ones. COMPS can be decomposed into three subsets, each containing stimuli that progressively isolate deeper understanding of the task of attributing properties to concepts,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2928–2949
May 2-6, 2023

by adding controls for more superficial heuristics. Our first subset—COMPS-BASE—measures the extent to which PLMs attribute properties to the right concepts, while varying the similarity of the positive (ROBIN) and the negative concepts (PENGUIN [high] vs. TABLE [low]). This controls for the possibility that models are relying on coarse-grained concept distinctions. For instance, in this setup a model should prefer (1a) over both versions of (1b).

(1) a. A **robin** can fly.
    b. *A (**penguin/table**) can fly.

Next, drawing on the phenomenon of property inheritance, the COMPS-WUGS set introduces a novel concept, WUG, expressed as the subordinate of the positive and negative concepts from a subset of the COMPS-BASE set, and tests the extent to which PLMs successfully attribute it the given property when it is associated with the positive concept. This increases the complexity of the reasoning task, as well as the distance between the associated concept (ROBIN) and property (*can fly*). These manipulations help to control for memorization of the literal phrases being tested, forcing models to judge properties for a novel concept that inherits the property from a known concept. In this task, given that a model successfully prefers (1a) over (1b), it should also prefer (2a) over (2b):

(2) a. A wug is a **robin**. Therefore, a wug can fly.
    b. *A wug is a **penguin**. Therefore, a wug can fly.

The final subset—COMPS-WUGS-DIST, combines the aforementioned controls by using negative concepts as distracting content and inserting them into the COMPS-WUGS stimuli. Specifically, we transform the stimuli of COMPS-WUGS by creating two subordinates for every minimal pair; one for the positive concept (ROBIN, subordinate: WUG) and the other for the negative concept (PENGUIN, subordinate: DAX), which acts as a distractor. This way, we control for the possibility that models may be relying on simple word associations between content words—of which there are only two in the prior tests—by introducing additional, irrelevant but contentful words into the context. Here, we consider models to be correct if they prefer (3a) over (3b), given that they prefer (1a) over (1b):

(3) a. A **wug** is a robin. A **dax** is a penguin. Therefore, a **wug** can fly.
    b. *A **wug** is a robin. A **dax** is a penguin. Therefore, a **dax** can fly.

Together, the three sets of stimuli tease apart more superficial predictive behaviors, such as contextual word associations, from more robust reasoning behaviors based on understanding of concept properties. While we can expect superficial predictive strategies to be brittle in the face of shallow perturbations and irrelevant distractions, robust property knowledge and reasoning behaviors should not.

We use COMPS to analyze robust property knowledge and its inheritance in 22 different PLMs, ranging from small masked language models to billion-parameter autoregressive language models. In our experiments with COMPS-BASE, we find PLMs to demonstrate strong performance in attributing properties to the correct concepts in our minimal pairs. However, we observe this strong performance largely when the concepts in the minimal pairs are trivially different (e.g., LION and TEA for the property *is a mammal*). When the concept pairs are similar (on the basis of different knowledge representations), we find models' performance to degrade substantially, by as much as 25 points. We observe a similar trend in our analyses on COMPS-WUGS—models first appear to show desirable behavior, potentially indicating proficiency in the more complex property inheritance reasoning. However, their overall performance declines drastically when investigated in the presence of distractors (i.e., on COMPS-WUGS-DIST). This failure is particularly pronounced in larger autoregressive PLMs, whose performance in fact drops below chance in cases where distracting information is proximal to the queried property, indicating the presence of a proximity effect. Together, our findings highlight brittleness of PLMs with conceptual knowledge and reasoning, as evidenced by failures in the face of simple controls. We make our code and data available at: `https://github.com/kanishkamisra/comps`.

2 Conceptual Minimal Pair Sentences (COMPS)

2.1 Connections to prior work

Prior work in exploring property knowledge in PLMs has adopted two different paradigms: one which uses probing classifiers to test if the applicability of a property can be decoded from the representations of LMs (Forbes et al., 2019; Da and Kasai, 2019; Derby et al., 2021); and the other which uses cloze-testing, in which LMs are tasked to fill in the blank in prompts that describe specific properties/factual knowledge about the world (Petroni et al., 2019; Weir et al., 2020). We argue that both approaches—though insightful—have key limitations for evaluating property knowledge, and that minimal pair testing overcomes these limitations to a beneficial extent.

Apart from ongoing debates surrounding the validity of probing classifiers (see Hewitt and Liang, 2019; Ravichander et al., 2021; Belinkov, 2022), the probing setup does not allow the testing of property knowledge in a precise manner. Specifically, several properties are often perfectly correlated in datasets such as the one we use here (see §2.2). For example, the property of being an animal and being able to breathe and grow, etc., are all perfectly correlated with one another. Even if the model's true knowledge of these properties is highly variable, probing its representations for them would yield the exact same result, leading to conclusions that overestimate the model's capacity for some properties, while underestimating for others. Evaluation using minimal pair sentences overcomes this limitation by allowing us to explicitly represent the properties of interest in language form, thereby allowing precise testing of property knowledge.

Similarly, standard cloze-testing of PLMs (Petroni et al., 2019; Weir et al., 2020; Jiang et al., 2021) also faces multiple limitations. First, it does not allow for testing of multi-word expressions, as by definition, it involves prediction of a single word/token. Second, it does not yield faithful conclusions about one-to-many or many-to-many relations: e.g. the cloze prompts "Ravens can ___." and "___ can fly." do not have a single correct answer. This makes our conclusions about models' knowledge contingent on choice of one correct completion over the other. The minimal pair evaluation paradigm overcomes these issues by generalizing the cloze-testing method to multi-word expressions—by focusing on entire sentences—and at the same time, pairing every prompt with a negative instance. This allows for a straightforward way to assess correctness: the choice between multiple correct completions is transformed into one between correct and incorrect, at the cost of having several different instances (pairs) for testing knowledge of the same property. Additionally, the minimal pairs paradigm allows us also to shed light on how the nature of negative samples affects model behavior, which has been missing in approaches using probing and cloze-testing. The usage of minimal pairs is a well-established practice in the literature, having been widely used in works that analyze syntactic knowledge of LMs (Marvin and Linzen, 2018; Futrell et al., 2019; Warstadt et al., 2020). We complement this growing literature by introducing minimal-pair testing to the study of conceptual knowledge in PLMs.

Our property inheritance analyses closely relate to the 'Leap-of-Thought' (LoT) framework of Talmor et al. (2020). In particular, LoT holds the taxonomic relations between concepts *implicit* and tests whether models can abstract over them to make property inferences—e.g., testing the extent to which models assign *Whales have bellybuttons* the 'True' label, given that *Mammals have bellybuttons* (with the implicit knowledge here being *Whales are mammals*). With COMPS-WUGS (and COMPS-WUGS-DIST), we instead explicitly provide the relevant taxonomic knowledge in the context and target whether PLMs can behave consistently with knowledge they have already demonstrated (in the base case, COMPS-BASE) and attribute the property in question to the correct subordinate concept. This also relates to recent work that measures consistency of PLMs' word prediction capacities in eliciting factual knowledge (Elazar et al., 2021; Ravichander et al., 2020).

2.2 Ground-truth Property Knowledge data

For our ground-truth property knowledge resource, we use a subset of the CSLB property norms collected by Devereux et al. (2014), which was further extended by Misra et al. (2022). The original dataset was constructed by asking 123 human participants to generate properties for 638 everyday concepts. Contemporary work has used this dataset by taking as positive instances all concepts for which a property was generated, while taking the rest as negative instances (Lucy and Gauthier, 2017; Da and Kasai, 2019, etc.) for each property. While this dataset has been popularly used in related literature, Misra et al. (2022) recently discovered striking gaps in coverage among the properties included in the dataset.[1] For example, the property *can breathe* was only generated for 6 out of 152 animal concepts, despite being applicable for all of them—as a result, contemporary work can be expected to have wrongfully penalized models that attributed this property to animals that could

---

[1] See also Sommerauer and Fokkens (2018) and Sommerauer (2022), who also discuss this limitation.

indeed breathe, and similarly for other properties. To remedy this issue, Misra et al. (2022) manually extended CSLB's coverage for 521 concepts and 3,645 properties. We refer to this extended CSLB dataset as XCSLB, and we use it as our source for ground-truth property knowledge.

2.3 Choosing negative samples

We rely on a diverse set of knowledge representation sources to construct negative samples for COMPS. Each source has a unique representational structure which gives rise to different pairwise similarity metrics, on the basis of which we pick out negative samples for each property:

Taxonomy We consider a hierarchical organization of our concepts, by taking a subset of WordNet (Miller, 1995) consisting of our 521 concepts. We use the `wup` similarity (Wu and Palmer, 1994) as our choice of taxonomic similarity.

Property Norms We use the XCSLB dataset and organize it as a matrix whose rows indicate concepts and columns indicate properties that are either present (indicated as 1) or absent (indicated as 0) for each concept. As our similarity measure, we consider the jaccard similarity between the row vectors of concepts. This reflects the overlap in properties between concepts, and is prevalent in studies utilizing conceptual similarity in cognitive science (Tversky, 1977; Sloman, 1993, etc.).

Co-occurrence We use the co-occurrence between concept words as an unstructured knowledge representation. For quantifying similarity, we use the cosine similarity of the GloVe vectors (Pennington et al., 2014) of our concept words.

Sampling Strategy Each property (p_i) in our dataset splits the set of concepts into two: a set of concepts that possess the property (Q_{p_i}), and a set of concepts that do not ($\neg Q_{p_i}$). We sample $\min(|Q_{p_i}|, 10)$—i.e., at most 10—concepts from Q_{p_i} and take them to be our positive set. Then for each concept in the positive set, we sample from $\neg Q_{p_i}$ the concept that is most similar (depending on the source) to the positive concept and take it as a negative concept for the property. We additionally include a negative concept that is randomly sampled from $\neg Q_{p_i}$, leaving out the concepts sampled on the basis of the three previously described knowledge sources. Examples of the four types of negative samples for the concept ZEBRA and the property *has striped patterns* are shown in Table 1.

Knowledge Rep.	Negative Concept	Similarity
Taxonomy	HORSE	0.88
Property Norms	DEER	0.63
Co-occurrence	GIRAFFE	0.75
Random	BAT	-

Table 1: Negatively sampled concepts selected on the basis of various knowledge representational mechanisms, where the property is *has striped patterns*, and the positive concept is ZEBRA.

2.4 Minimal Pair Construction

Following our negative sample generation process, we end up with total of 49,280 pairs of positive and negative concepts that span across 3,645 properties (14 pairs per property, on average). Every property is associated with a property phrase—a verb phrase which expresses the property in English, as provided in XCSLB. Using these materials, we construct our three datasets of minimal pair sentence stimuli, examples of which are shown in Figure 1.

COMPS-BASE The COMPS-BASE dataset contains minimal pair sentences that follow the template: "`[DET] [CONCEPT] [property-phrase]`." where `[DET]` is an optional determiner, and `[CONCEPT]` is the noun concept. Applying this template to our generated pairs results in 49,280 instances. See Figure 1a for an example.

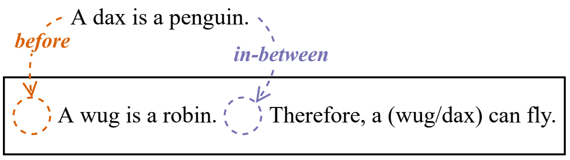
COMPS-WUGS We test property inheritance in PLMs using only the animal kingdom subset of COMPS-BASE (152 concepts, 944 properties, and 13,888 pairs), keeping the same negative samples. We convert the original minimal pair sentences in COMPS-BASE, in which the positive concept is an animal, into pairs of two-sentence stimuli by first introducing a new concept (WUG) to be the subordinate of the concepts in the original minimal pair. We then express its property inheritance in a separate sentence. Our two sentence stimuli follow the template: "A wug is a `[CONCEPT]`. Therefore, a wug `[property-phrase]`." Although we use *wug* as our running example for the subordinate concept, we use four different nonsense words {*wug*, *dax*, *blicket*, *fep*} equal numbers of times, to avoid making spurious conclusions based on a single nonsense word.[2] Introducing an intervening novel concept allows us to robustly control for simple word-level associations between concepts and properties that models might have picked up during

---

[2] As we describe in §4, we also tried a different set of nonce words, to address concerns about possible impacts of using nonce words from existing literature (e.g., *wug*).

Property: *can fly*
Positive: ROBIN
Negative: PENGUIN
Subordinate: WUG
COMPS-BASE: A (**robin/penguin**) can fly.
COMPS-WUGS: A wug is a (**robin/penguin**). Therefore, a wug can fly.

(a) Instances of COMPS-BASE and COMPS-WUGS.



(b) Distraction scheme for stimuli in COMPS-WUGS-DIST, where the distractor is inserted either **before** or **in between** each COMPS-WUGS stimulus.

Figure 1: Examples of materials used in our experiments. In this example, ROBIN is the positive concept.

training. Figure 1a shows an example.

COMPS-WUGS-DIST To add distracting information, we follow Pandia and Ettinger (2021) and convert the COMPS-WUGS stimuli by associating a different subordinate concept (DAX) with the negative concept (`[NEG-CONCEPT]`), and inserting it **before** or **in-between** the sentence containing the positive concept and its subordinate, separately. This results in two subsets (**before** and **in-between**) of three-sentence minimal pair stimuli, which differ in the subordinate to which the property is attributed. We use the following template to create our stimuli: "A **wug** is a `[CONCEPT]`. A **dax** is a `[NEG-CONCEPT]`. Therefore, a (**wug/dax**) `[property-phrase]`." That is, we have stimuli that resemble COMPS-WUGS but instead deal with a pair of competing subordinate concepts in context.[3] See Figure 1b for an example.

3 Methodology

3.1 Models Investigated

We investigate property knowledge and property inheritance capacities of 22 different PLMs, belonging to six different families. We evaluate four widely used masked language modeling (MLM) families: (1) ALBERT (Lan et al., 2020), (2) BERT (Devlin et al., 2019), (3) ELECTRA (Clark et al., 2020), and (4) RoBERTa (Liu et al., 2019); as well as two auto-regressive language modeling families: (1) GPT2 (Radford et al., 2019), and (2) the GPT-Neo (Black et al., 2021) and GPT-J models (Wang and Komatsuzaki, 2021) from EleutherAI. We also use distilled versions of BERT-base, RoBERTa-base, and GPT2, trained using the method described by Sanh et al. (2019). We list each model's parameters, vocabulary size, and training corpora in Table 3 (Appendix A).

3.2 Measuring Performance

To evaluate models on COMPS, we compare their log-probabilities for the property phrase—conditioned on contexts (to the left) containing the positive and negative noun concepts. That is, we hold the property phrase constant, and compare across minimally differing conditions to evaluate the probability with which a property is attributed to each concept. For example, we score stimuli in COMPS-BASE, e.g., "*A dog can bark.*" as:

$$\log p(\text{can bark.} \mid \text{A dog}),$$

its corresponding stimulus in COMPS-WUGS, "*A wug is a dog. Therefore, a wug can bark.*" as:

$$\log p(\text{can bark.} \mid \text{A wug is a dog. Therefore, a wug}),$$

and similarly—assuming CAT as the negative concept—the corresponding stimuli in our COMPS-WUGS-DIST subset, "*A wug is a dog. A dax is a cat. Therefore, a wug can bark.*" as:[4]

$$\log p(\text{can bark.} \mid \text{A wug is a dog. A dax is a cat. Therefore, a wug}).$$

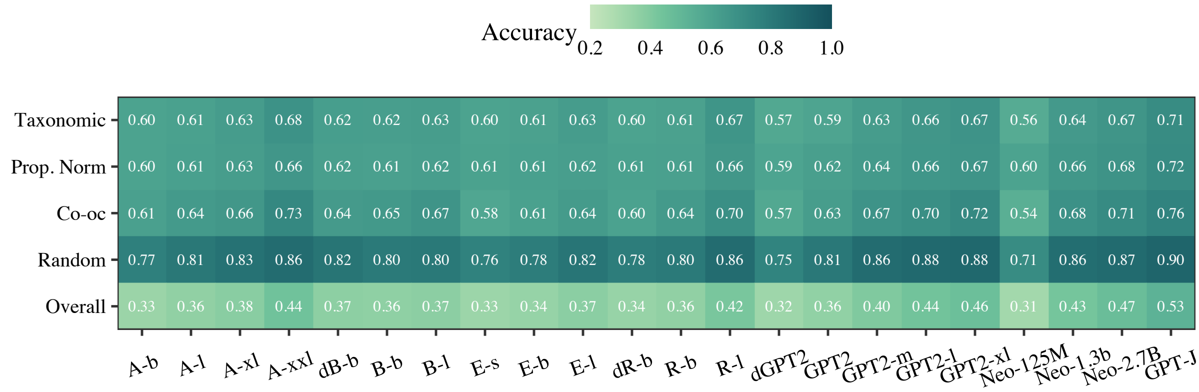
This approach to eliciting conditional LM judgments is equivalent to the "scoring by premise" method (Holtzman et al., 2021), which has been shown to result in stable comparisons across items. Additionally, this also takes into account the potential noise due to frequency effects or tokenization differences (Misra et al., 2021). Estimating these conditional log-probabilities using auto-regressive PLMs can be directly computed in a left-to-right manner. For MLMs, we use their conditional pseudo-loglikelihoods (Salazar et al., 2020) as a proxy for conditional log-probabilities.

Based on this simple method of eliciting relative acceptability measures from PLMs, we evaluate

---

[3] We again choose from our list of four nonsense words (*wug*, *dax*, *blicket*, and *fep*), which amounts to 12 unique ordered pairs, after accounting for counterbalancing.

[4] Here we show an example where the distractor is added **in-between** the context specifying the positive concept, and the queried property knowledge.



	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
Taxonomic	0.60	0.61	0.63	0.68	0.62	0.62	0.63	0.60	0.61	0.63	0.60	0.61	0.67	0.57	0.59	0.63	0.66	0.67	0.56	0.64	0.67	0.71
Prop. Norm	0.60	0.61	0.63	0.66	0.62	0.61	0.62	0.61	0.61	0.62	0.61	0.61	0.66	0.59	0.62	0.64	0.66	0.67	0.60	0.66	0.68	0.72
Co-oc	0.61	0.64	0.66	0.73	0.64	0.65	0.67	0.58	0.61	0.64	0.60	0.64	0.70	0.57	0.63	0.67	0.70	0.72	0.54	0.68	0.71	0.76
Random	0.77	0.81	0.83	0.86	0.82	0.80	0.80	0.76	0.78	0.82	0.78	0.80	0.86	0.75	0.81	0.86	0.88	0.88	0.71	0.86	0.87	0.90
Overall	0.33	0.36	0.38	0.44	0.37	0.36	0.37	0.33	0.34	0.37	0.34	0.36	0.42	0.32	0.36	0.40	0.44	0.46	0.31	0.43	0.47	0.53

Figure 2: Accuracies of PLMs on COMPS-BASE under various negative sampling schemes. Chance performance for all rows is 50%, except for 'Overall,' where it is 6.25%. Refer to Table 3 for unabbreviated model names.

a model's accuracy on all COMPS stimuli as the percentage of times its log-probability for a property is greater when conditioned on the context that attributes the property to the positive—as opposed to the negative—concept. Since all cases are forced-choice tasks between two instances, chance performance is set to 50%. Table 4 (Appendix B) shows examples of all COMPS stimuli and GPT-J's conditional log-probabilities for them.

4 Experiments and Analyses

4.1 Base property knowledge of PLMs and their sensitivity to similarity effects

We begin by evaluating the 22 PLMs on COMPS-BASE. Here we focus on the extent to which models robustly associate properties to the correct concepts across stimuli with varying kinds of similarity between the positive and negative concepts. We report accuracies of the 22 PLMs on COMPS-BASE across the four different negative sampling schemes that we specified in §2.3. We additionally report a more stringent accuracy measure that we refer to as 'Overall accuracy,' which is calculated for every property and its positive concept, as the percentage of times a model correctly attributes the property to the positive concept in **all four types of negative sampling schemes**. Chance performance for only the 'Overall' case is then 6.25% ($0.5^4 \times 100$). Figure 2 shows these results.

From Figure 2, we see that models strongly distinguish between positive and negative concepts in cases where they are dramatically different—i.e., where negative concepts were sampled randomly (e.g., BEAR [positive] vs BOTTLE [negative] for the property *can breathe*). However performance drops substantially when there are subtler differences between the two concepts—e.g, the concepts WALRUS (positive) and SHARK (negative) for the property *is a mammal*. For instance, the best performing model in any similarity-based negative sampling scheme (GPT-J, 76%, 'Co-oc') only slightly outperforms the worst model in the random negative sampling scheme (Neo-125M, 71%). The performance of PLMs is not substantially different across the three similarity-based negative sampling schemes, suggesting that the dynamics of model sensitivity in attributing properties to concepts are largely harmonized across various types of similarities. As a result of models' insensitivity in presence of similar negative concepts, the overall accuracies are very modest in value, with the overall accuracy of the best performing model (GPT-J) being only 53%. This overall performance is, however, significantly above chance (6.25%). We discuss additional findings, such as performance by property type and model size, in Appendix C, since they are incidental to the main conclusions of this analysis.

4.2 Property inheritance in PLMs

Having established the base property knowledge of PLMs, we now investigate the extent to which they can show behavior that is consistent with reasoning required to handle property inheritance. We first investigate their performance on COMPS-WUGS, created using the subset of COMPS-BASE containing only animal concepts (see §2.4 for stimulus construction). Table 2 shows average accuracies obtained by PLMs on our property inheritance stimuli, and compares them to average accuracies on COMPS-BASE—aggregating across all negative sampling schemes. Recall that the stimuli in COMPS-WUGS present a more challenging property attribution task than in COMPS-BASE, by

COMPS subset	Size	Acc.
BASE	49.3K	$68.4_{1.7}$
BASE (*animal kingdom only*)	13.8K	$67.1_{2.0}$
WUGS	13.8K	$68.9_{2.3}$
WUGS-DIST (**before**)	13.8K	$59.2_{3.9}$
WUGS-DIST (**in-between**)	13.8K	$47.2_{4.5}$

Table 2: Average accuracy (and standard error of the mean) of PLMs ($N = 22$) on each of our COMPS subsets. Chance performance is 50% throughout.

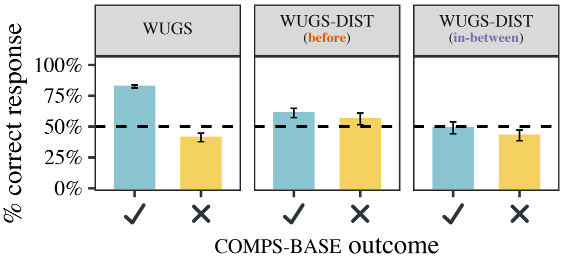


Figure 3: Distribution of model performance on COMPS-WUGS and COMPS-WUGS-DIST (both subsets) across possible outcomes (correct = ✓, incorrect = ✗) of the models on corresponding minimal pairs in COMPS-BASE. Error bars indicate 95% CI, while dashed line indicates chance performance (50%).

not only controlling for coarse-grained similarity effects, but also introducing an intervening novel concept that is expected to inherit the properties of the positive concept. By measuring attribution of properties more indirectly, these stimuli increase the complexity of the reasoning and control for memorization of the literal phrase initially tested with COMPS-BASE.

Table 2 shows the average accuracy of the PLMs on each subset of COMPS. Despite the increase in complexity, we see that PLMs actually show slightly stronger performance on COMPS-WUGS (68.9%) than on COMPS-BASE (67.1%). This means that there are instances in which models prefer the property in the positive context over the negative context (4a > 4b), but show the opposite behavior in COMPS-BASE (4d > 4c).

(4) a. A wug is a **robin**. Therefore, a wug can fly.
b. A wug is a **penguin**. Therefore, a wug can fly.
c. A **robin** can fly.
d. A **penguin** can fly.

This pattern of performance could lead to spurious conclusions that models are successfully executing property inheritance, when in fact they show a lack of the pre-requisite property knowledge based on their failure on COMPS-BASE. We will discuss these inconsistencies in more detail below. Overall, however, the relatively strong performance on COMPS-WUGS suggests that models are largely unaffected when we control for simple memorization of tested phrases—e.g., *robin can fly*—by linking known concepts to properties through an intervening subordinate concept (*wug*). This suggests that models are not relying on simple memorization, but does not control for the possibility of simple association between content words (*robin* and *fly*)—for this we turn to COMPS-WUGS-DIST.

The COMPS-WUGS-DIST test assesses whether models retain strong property attribution performance when content words in the context are not all relevant for the property prediction. The stimuli thus include irrelevant distractor concepts and their subordinates—which, in a robust model, should not affect attribution of the property to the correct concept (see §2.4 for stimulus construction).

From Table 2, the average accuracies of PLMs on both subsets of COMPS-WUGS-DIST (**before** and **in-between**) indicate that overall, models now show clear degradation in property inheritance performance as a result of the distracting information. Specifically, the PLMs' performance drops by 9.7 points on instances when the distracting information is added **before** the relevant context and queried property, and by 21.7 points on instances where it is added **in-between** the two, relative to the undistracted property inheritance stimuli (COMPS-WUGS). Notably, the latter drop in performance brings models level with chance accuracy (we fail to reject the null hypothesis that avg. accuracy of models is 50%; $p = .62$, Wilcoxon signed rank exact text), highlighting a pronounced lack of robustness in PLMs' capacity to attribute properties to the correct concepts in their input context.

Accounting for spurious performance The COMPS-WUGS results above raise the concern that models are often showing spurious performance: accurately demonstrating property inheritance behavior without actually possessing the right property knowledge. To shed more light on this potential issue, we plot the distribution of model accuracies on our property inheritance stimuli (COMPS-WUGS and COMPS-WUGS-DIST) divided based on their outcomes on the corresponding stimuli in COMPS-BASE. Figure 3 shows these distributions. In COMPS-WUGS and both subsets of COMPS-WUGS-DIST, models show this spurious correct behavior on 41.3%, 55.6%, and 42.8% of instances in which they produce incorrect judgments on the corresponding COMPS-BASE stimuli (yellow bars

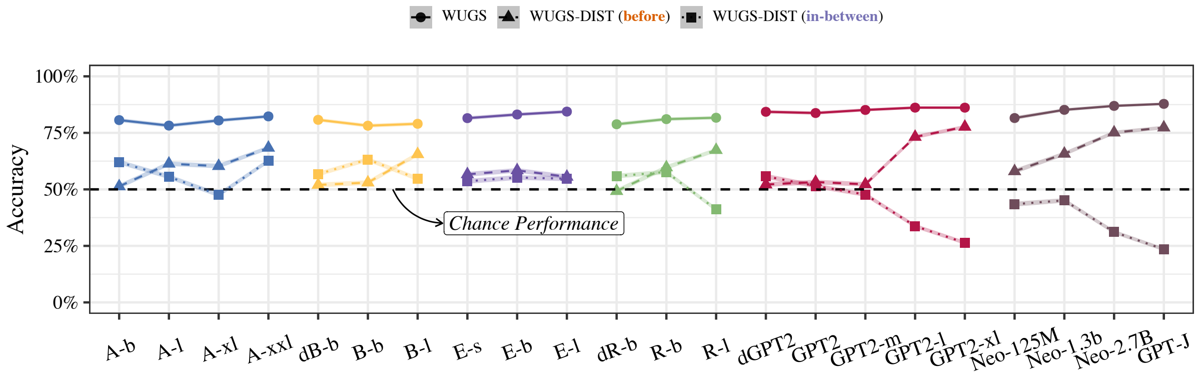


Figure 4: Accuracies of individual models (grouped by family, in increasing order based on number of parameters) on COMPS-WUGS and COMPS-WUGS-DIST. Black dashed line indicates chance performance (50%). Refer to Table 3 for unabbreviated model names. Error bands indicate 95% Bootstrap CIs.

in Figure 3). This non-trivial proportion of cases with spurious performance further reinforces the idea that PLMs' successful predictions on these tests are likely relying on heuristics rather than robust inferences about property knowledge. We can remove the effects of these spurious instances by filtering to items in which models give the correct answer on COMPS-BASE (**blue** bars in Figure 3)—though we see that the overall conclusions remain the same after this filtering.

On the pronounced effect of proximity in autoregressive PLMs Our previous discussion summarized the aggregate property inheritance behavior of the 22 PLMs we considered—we now zoom in for a model-wise analysis. Figure 4 shows models' relative accuracies on COMPS-WUGS and COMPS-WUGS-DIST, filtering to items with correct COMPS-BASE performance, as in the blue bars of Figure 3. Consistent with our overall findings, we observe distracting content to substantially degrade model performance across the board.[5] A particularly noteworthy pattern is that the degradation in autoregressive PLM families—GPT2 and EleutherAI—shows a stark sensitivity to *proximity effects*. While these classes of model seem to suffer less when distracting content is added **before** the context containing the positive concept (thus placing the distraction farther from the queried property), they show substantially worse performance when the opposite is the case (i.e., when distraction is added **in-between**, and is therefore closer to the queried property). This degradation due to proximity of the distracting content becomes **catastrophically worse as models grow larger in the number of pre-trained parameters**—in fact bringing their performance down to as much as 26.2 points **below chance** (in GPT-J, which has 6B parameters). While MLMs also show similar levels of degraded performance in presence of distraction, they do not seem to show any systematic sensitivity to proximity effects, likely due to their bidirectional nature.

Results on GPT-3 In addition to our main experiments, we also evaluate GPT-3 (Brown et al., 2020) models on a small subset of COMPS stimuli (denoted as miniCOMPS). Results from this analysis (shown in Appendix C.1) are largely aligned with our main conclusions, with all GPT-3 models—including the largest one (175B parameters)—performing worse than chance on the **in-between** subset of miniCOMPS-WUGS-DIST, while performing substantially better on miniCOMPS-WUGS and miniCOMPS-WUGS-DIST (**before**). Together with our main results, this indicates that scaling alone may be insufficient to elicit robust inferences about concepts and their properties.

Choice of nonce words Nonce words constitute an important design decision for our stimuli—we followed precedents in language acquisition research (Berko, 1958; Gopnik and Sobel, 2000, *i.a.*) and used previously existing nonce words (such as *wug* and *blicket*) to represent novel concepts in context. While these are expected to be novel for humans, they may appear in pre-training corpora on which PLMs are usually trained.[6] This raises a potential concern that PLMs could already be biased toward certain properties for these words (e.g., *wug* is commonly depicted as a bird), and

---

[5] See also Pandia and Ettinger (2021) for a similar degradation of performance on cloze-tasks involving factual retrieval.

[6] e.g., *wug* appears in wikipedia: `https://en.wikipedia.org/wiki/Jean_Berko_Gleason` (accessed on Jan 23)

may struggle to associate them with different properties.[7] To explore this empirically, we conducted experiments with alternative nonce words (generated synthetically, similar to Kim et al. (2022); see Appendix C.2). Figure 7 (Appendix C.2) shows results on COMPS-WUGS and COMPS-WUGS-DIST with randomly sampled nonce words. We see that the new results are comparable to those in Figure 4, with models showing the same preference on both stimulus versions 80% of the time on average. This suggests that the choice of nonce words is not producing any noteworthy bias.

Framing of novel taxonomic information Another relevant stimulus design decision is the phrasing for introducing novel concepts in context. While we used *"A wug is a* `[CONCEPT]`*"* for our main experiments, we additionally tested with an alternate framing: *"A wug is a type of* `[CONCEPT]`*."* From Figure 9 (Appendix C.3), we again see that the overall patterns of results are comparable to the original results, with models showing the same preference across both versions of the stimuli on COMPS-WUGS and COMPS-WUGS-DIST 90% of the time, on average.

5 General Discussion and Conclusion

The overall goal of COMPS is to shed light on the extent to which PLMs can robustly (1) attribute to real world concepts (e.g., HORSE, WHALE) their correct properties (e.g., *is a mammal*); and (2) demonstrate behavior consistent with *property inheritance*: a reasoning process in which concepts are endowed with the properties of their superordinates (Smith and Estes, 1978; Sloman, 1998; Murphy, 2002). Testing PLMs for these abilities allows us to ask key questions about how they encode and transfer knowledge. To target these capabilities more precisely, and mitigate potential inflation of performance by superficial heuristics such as coarse-grained similarity and word association, we propose incrementally increasing levels of controls in constructing our minimal pair stimuli, progressively making the task of attributing properties to concepts more challenging.

Findings from our initial experiment on COMPS-BASE established that the basic capacity of models to attribute properties to everyday concepts is largely coarse grained. PLMs were more successful in making correct property attributions when the candidate concepts were radically different, and struggled when the concepts shared semantic relations or had high co-occurrence. On testing for 'property inheritance' behavior (via COMPS-WUGS), PLMs initially appeared to demonstrate reasonable success, but they also showed spurious behavior in achieving correct performance on a non-trivial number of instances for which they did not succeed in the prerequisite base condition. Furthermore, this performance declined substantially in the presence of distracting information (COMPS-WUGS-DIST), providing further evidence that what property knowledge and reasoning we appear to see in these PLMs is more reliant on superficial heuristics than on ideal reasoning behavior. Of particular note is our finding of *catastrophic distraction* in large autoregressive PLMs, whose sensitivity to proximity effects brings their overall performance well under chance, especially when scaled up to billions of parameters.

Contemporary work has highlighted the promise of PLMs on high-level tasks requiring—among other things—access to proper relational knowledge between concepts (see Petroni et al., 2019; Safavi and Koutra, 2021; Piantadosi and Hill, 2022). By drawing on the concept of property inheritance, our experiments target reasoning ability based on perhaps the most well-established of relations—the taxonomic or the `IsA` relation (Murphy, 2003). Recent work has also alluded to the proficiency of PLMs in capturing taxonomic information about everyday objects and entities (Weir et al., 2020; Chen et al., 2021, though see Ravichander et al. (2020)). Findings from our controlled experiments suggest that PLMs' approximation of the consequences of the taxonomic relation is at best noisy, in light of clear failures especially in presence of similarity-governed competition. We conclude from our analyses that instead of robustly extracting relational information and reasoning about properties of concepts, it is likely that the PLMs tested here are optimized to prefer superficial cues in making word predictions, leading to mistakes and inaccuracies in presence of irrelevant and distracting information. Since robust natural language understanding will be critically reliant on understanding of property knowledge and implications of property transfer, we hope that these findings will motivate adoption of rigorous assessment methods as well as work toward more robust property knowledge and reasoning in PLMs.

---

[7] We thank Reviewers 1 and 3, Najoung Kim and Kyle Mahowald for raising this concern.

Limitations

Zero-shot setup Using a zero-shot setup to test PLMs for human-like capacities such as property inheritance (as we have done in this work) has recently come under scrutiny. In particular, Lampinen (2022) argues that such a setup could be problematic because PLMs are trained to imitate the language produced by countless individuals with different beliefs, cultures, and behaviors. As a result, PLMs are likely to be handicapped in assigning sufficient probability mass to the desired family of continuations, given minimal prompts without any particular task-specific context. Instead, Lampinen (2022) suggests the need for PLMs

> "[...] to be guided into an experiment-appropriate behavioral context, analogously to the way cognitive researchers place humans in an experimental context, and orient them toward the task with instructions and examples."

This criticism is valid, and it is possible that models could overcome their lack of robustness to distraction effects by observing examples of our stimuli in context, though this has largely been shown in PLMs that are significantly larger than the ones we have tested in this work (Brown et al., 2020; Chowdhery et al., 2022; Wei et al., 2022a).[8] Indeed, recent work has demonstrated these larger PLMs to achieve strong performance on other types of reasoning—such as those required for solving math problems, reversing sequences, etc.—by priming models to produce additional textual content that represents intermediate reasoning steps and explanations (Nye et al., 2021; Wei et al., 2022b; Lampinen et al., 2022), in a few-shot setting.[9] At the same time, a few-shot version of COMPS stimuli could expose models to the possibility of leveraging heuristics that are naturally absent in the zero-shot setup, and therefore such a setup would critically require the design of additional controls, which we leave for future work.

Ideal reasoning behavior Another limitation of our work is that it takes ideal and robust property inheritance behavior as the monolithic gold-standard for human cognition, something that recent work has cautioned against (Pavlick and Kwiatkowski, 2019; Dasgupta et al., 2022; Webson et al., 2023). Although we relied on a database of concept-property pairs that were largely generated by human participants, whether or not humans will be robust to the types of distraction that were observed in PLMs is an open question and requires further investigation. However, notably we are not making direct comparisons between models and humans here—we argue that our primary contribution of controlled stimuli that tease apart shallow processing from robust conceptual reasoning in PLMs bears substantial merit that is independent from any comparisons between humans and computational systems. Furthermore, we emphasize that we are setting a reasonable—and to a certain extent, human-independent—desideratum in this work, which is that models should robustly capture ground-truth knowledge about everyday concepts and their properties and reflect this knowledge in their inferences about newly introduced concepts.

Behavioral evaluation This work tests and analyses PLMs on property knowledge and property inheritance only from a behavioral perspective, which at its core is a correlational endeavor. Potential future work could complement our results by providing evidence from representational analyses, or by devising causal interventions, similar to those recently explored in the realm of syntactic agreement (Finlayson et al., 2021), or in testing of negation and hypernymy in NLI models (Geiger et al., 2020), among others. Importantly, this would require the development of new methods that shed light on how new information—such as the ones we use in COMPS-WUGS and COMPS-WUGS-DIST—is integrated into the model (see Misra et al. (2022) for an example of such an analysis for novel properties).

Targeted language Finally, COMPS only consists of sentences in English, thereby biasing our results only for PLMs trained in that language.

Acknowledgments

For helpful comments we thank Najoung Kim, Tal Linzen, Brenden Lake, Kyle Mahowald, Andrew Lampinen, the anonymous reviewers, and audiences at the Computation and Psycholinguistics

---

[8] though see recent work by Shi et al. (2023), who show distraction effects in such large PLMs in solving arithmetic reasoning problems, even after using sophisticated in-context prompting methods such as Chain-of-Thought (Wei et al., 2022b), Least-to-Most (Zhou et al., 2022), and Self-Consistency (Wang et al., 2022).

[9] See also Sinha et al. (2022), who analyze PLMs comparable in size to those studied in this work in a few-shot minimal-pair setting.

Lab (NYU), the Human and Machine Learning Lab (NYU), the UChicago CompLing Lab, UT Austin Linguistics Grad Student Seminar , and the MIT Department of Brain and Cognitive Sciences. Any errors are our own. Our experiments were conducted with resources provided by the Rosen Center for Advanced Computing at Purdue University (McCartney et al., 2014). We are also grateful to Sam Huang for helping out with experiments conducted on GPT-3/3.5 models.

References

Mostafa Abdou, Artur Kulmizev, Daniel Hershcovich, Stella Frank, Ellie Pavlick, and Anders Søgaard. 2021. Can language models encode perceptual structure without grounding? a case study in color. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 109–132, Online. Association for Computational Linguistics.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Jean Berko. 1958. The child's learning of english morphology. *Word*, 14(2-3):150–177.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. 2009. Clueweb09 data set.

Catherine Chen, Kevin Lin, and Dan Klein. 2021. Constructing taxonomies from pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4687–4700, Online. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations*.

Jeff Da and Jungo Kasai. 2019. Cracking the contextual commonsense code: Understanding commonsense reasoning aptitude of deep contextual representations. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 1–12, Hong Kong, China. Association for Computational Linguistics.

Ishita Dasgupta, Andrew K Lampinen, Stephanie CY Chan, Antonia Creswell, Dharshan Kumaran, James L McClelland, and Felix Hill. 2022. Language models show human-like content effects on reasoning. *arXiv preprint arXiv:2207.07051*.

Steven Derby, Paul Miller, and Barry Devereux. 2021. Representation and pre-activation of lexical-semantic knowledge in neural language models. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 211–221, Online. Association for Computational Linguistics.

Barry J Devereux, Lorraine K Tyler, Jeroen Geertzen, and Billi Randall. 2014. The Centre for Speech, Language and the Brain (CSLB) concept property norms. *Behavior Research Methods*, 46(4):1119–1127.

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

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Matthew Finlayson, Aaron Mueller, Sebastian Gehrmann, Stuart Shieber, Tal Linzen, and Yonatan Belinkov. 2021. Causal analysis of syntactic agreement mechanisms in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1828–1843, Online. Association for Computational Linguistics.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do Neural Language Representations Learn Physical Commonsense? In *Proceedings of the 41st Annual Meeting of the Cognitive Science Society*.

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. In *Proceedings of NAACL-HLT 2019*, pages 32–42, Minneapolis, Minnesota. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173, Online. Association for Computational Linguistics.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Alison Gopnik and David M Sobel. 2000. Detecting blickets: How young children use information about novel causal powers in categorization and induction. *Child development*, 71(5):1205–1222.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 25–30.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Najoung Kim, Tal Linzen, and Paul Smolensky. 2022. Uncontrolled lexical exposure leads to overestimation of compositional generalization in pretrained models. *arXiv preprint arXiv:2212.10769*.

Brenden M Lake and Gregory L Murphy. 2021. Word meaning in minds and machines. *Psychological Review*.

Andrew K Lampinen, Ishita Dasgupta, Stephanie CY Chan, Kory Matthewson, Michael Henry Tessler, Antonia Creswell, James L McClelland, Jane X Wang, and Felix Hill. 2022. Can language models learn from explanations in context? *arXiv preprint arXiv:2204.02329*.

Andrew Kyle Lampinen. 2022. Can language models handle recursively nested grammatical structures? a case study on comparing models and humans. *arXiv preprint arXiv:2210.15303*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Li Lucy and Jon Gauthier. 2017. Are distributional representations ready for the real world? evaluating word vectors for grounded perceptual meaning. In *Proceedings of the First Workshop on Language Grounding for Robotics*, pages 76–85, Vancouver, Canada. Association for Computational Linguistics.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202.

Gerry McCartney, Thomas Hacker, and Baijian Yang. 2014. Empowering Faculty: A Campus Cyberinfrastructure Strategy for Research Communities. *Educause Review*.

George A Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Kanishka Misra. 2022. minicons: Enabling flexible behavioral and representational analyses of transformer language models. *arXiv preprint arXiv:2203.13112*.

Kanishka Misra, Allyson Ettinger, and Julia Rayz. 2020. Exploring BERT's sensitivity to lexical cues using tests from semantic priming. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4625–4635, Online. Association for Computational Linguistics.

Kanishka Misra, Allyson Ettinger, and Julia Rayz. 2021. Do language models learn typicality judgments from text? In *Proceedings of the 43rd Annual Conference of the Cognitive Science Society*.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2022. A property induction framework for neural language models. In *Proceedings of the 44th Annual Conference of the Cognitive Science Society*.

Gregory L Murphy. 2002. *The Big Book of Concepts*. MIT press.

M Lynne Murphy. 2003. *Semantic relations and the lexicon: Antonymy, synonymy and other paradigms*. Cambridge University Press.

Sebastian Nagel. 2016. CC-News.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Lalchand Pandia and Allyson Ettinger. 2021. Sorting through the noise: Testing robustness of information processing in pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1583–1596, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Roma Patel and Ellie Pavlick. 2022. Mapping language models to grounded conceptual spaces. In *International Conference on Learning Representations*.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP 2014*, pages 1532–1543.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Steven T Piantadosi and Felix Hill. 2022. Meaning without reference in large language models. *arXiv preprint arXiv:2208.02957*.

M Ross Quillian. 1967. Word concepts: A theory and simulation of some basic semantic capabilities. *Behavioral science*, 12(5):410–430.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI*.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2021. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3363–3377, Online. Association for Computational Linguistics.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102, Barcelona, Spain (Online). Association for Computational Linguistics.

Lance J Rips, Edward E Smith, and Douglas L Medin. 2012. *Concepts and categories: Memory, meaning, and metaphysics*. Oxford University Press.

Timothy T Rogers and James L McClelland. 2004. *Semantic cognition: A parallel distributed processing approach*. MIT press.

Dana Rubinstein, Effi Levi, Roy Schwartz, and Ari Rappoport. 2015. How well do distributional models capture different types of semantic knowledge? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 726–730, Beijing, China. Association for Computational Linguistics.

Tara Safavi and Danai Koutra. 2021. Relational World Knowledge Representation in Contextual Language Models: A Review. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1053–1067, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Thibault Sellam, Steve Yadlowsky, Jason Wei, Naomi Saphra, Alexander D'Amour, Tal Linzen, Jasmijn Bastings, Iulia Turc, Jacob Eisenstein, Dipanjan Das, et al. 2021. The multiberts: Bert reproductions for robustness analysis. *arXiv preprint arXiv:2106.16163*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. *arXiv preprint arXiv:2302.00093*.

Vered Shwartz and Yejin Choi. 2020. Do neural language models overcome reporting bias? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6863–6870, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Koustuv Sinha, Jon Gauthier, Aaron Mueller, Kanishka Misra, Keren Fuentes, Roger Levy, and Adina Williams. 2022. Language model acceptability judgements are not always robust to context. *arXiv preprint arXiv:2212.08979*.

Steven A Sloman. 1993. Feature-based induction. *Cognitive psychology*, 25(2):231–280.

Steven A Sloman. 1998. Categorical inference is not a tree: The myth of inheritance hierarchies. *Cognitive Psychology*, 35(1):1–33.

Edward E Smith and William K Estes. 1978. Theories of semantic memory. *Handbook of learning and cognitive processes*, 6:1–56.

Pia Sommerauer and Antske Fokkens. 2018. Firearms and tigers are dangerous, kitchen knives and zebras are not: Testing whether word embeddings can tell. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Brussels, Belgium. Association for Computational Linguistics.

Pia Johanna Maria Sommerauer. 2022. Diagnosing semantic properties in distributional representations of word meaning.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. *Advances in Neural Information Processing Systems*, 33:20227–20237.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Amos Tversky. 1977. Features of similarity. *Psychological review*, 84(4):327.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Albert Webson, Alyssa Marie Loo, Qinan Yu, and Ellie Pavlick. 2023. Are language models worse than humans at following prompts? it's complicated. *arXiv preprint arXiv:2301.07085*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Nathaniel Weir, Adam Poliak, and Benjamin Van Durme. 2020. Probing neural language models for human tacit assumptions. In *CogSci 2020*, pages 377–383. Cognitive Science Society.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP 2020: Demos*, pages 38–45, Online. Association for Computational Linguistics.

Zhibiao Wu and Martha Palmer. 1994. Verb Semantics and Lexical Selection. In *32nd Annual Meeting of the Association for Computational Linguistics*, pages 133–138, Las Cruces, New Mexico, USA. Association for Computational Linguistics.

Yian Zhang, Alex Warstadt, Xiaocheng Li, and Samuel R. Bowman. 2021. When do you need billions of words of pretraining data? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1112–1125, Online. Association for Computational Linguistics.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27. IEEE.

A Model Metadata

Table 3 shows the different models used in our experiments, along with their abbreviation, tokenization scheme, total parameters, vocabulary size, number of tokens encountered during training, and corpora on which they are pre-trained. All models were accessed using `minicons` (Misra, 2022),[10] a python library that serves as a wrapper around Huggingface's `transformers` (Wolf et al., 2020), and provides a unified mechanism for eliciting log-probabilities in batch-wise manner for any autoregressive or masked LM that is accessible through the huggingface hub, or is trained using the transformers library. Experiments were performed using an NVIDIA V100 GPU (32 GB RAM) and took about 6 hours to run, discounting the time it took to download the models from the Huggingface Hub.[11]

B Preview of COMPS stimuli

We show examples of stimuli from our COMPS-BASE, COMPS-WUGS, and COMPS-WUGS-DIST datasets in Listing 1 and Listing 2, respectively. Stimuli with distraction—i.e., in COMPS-WUGS-DIST—are similar to that in Listing 1, but with the `distraction_type` value set to either '`before`' or '`in-between`'.

```
{
    "id": 12706,
    "property": "can fly",
    "acceptable_concept": "owl",
    "unacceptable_concept": "squirrel",
    "prefix_acceptable": "an owl",
    "property_phrase": "can fly.",
    "prefix_unacceptable": "a squirrel",
    "condition": "co-occurrence",
    "similarity": 0.62
}
```

Listing 1: An instance of COMPS-BASE. "`condition`" represents the negative sampling scheme, and "`similarity`" represents the similarity between the acceptable concept and the unacceptable concept on the basis of the condition (either Taxonomic, Property Norm, Co-occurence, or Random).

```
{
    "item": 8343,
    "comps_id": 28798,
    "property": "has hooves",
    "acceptable_concept": "horse",
    "unacceptable_concept": "dog",
    "prefix_acceptable": "A dax is a
    ↪  horse. Therefore, a dax",
    "prefix_unacceptable": "A dax is a
    ↪  dog. Therefore, a dax",
    "property_phrase": "has hooves.",
    "negative_sample_type":
    ↪  "co-occurrence",
    "similarity": 0.62,
    "distraction_type": "undistracted"
}
```

Listing 2: An instance of COMPS-WUGS. "`condition`" and "`similarity`" are the same as in Listing 1. "`distraction_type`" denotes the type of distraction used (undistracted, before, in-between).

Table 4 shows examples from each subset of COMPS, and the conditional log-probability scores as computed by GPT-J (Wang and Komatsuzaki, 2021), the largest LM tested on the full set of stimuli.

C Additional findings and analyses

C.1 Testing GPT-3/3.5

Recent work in scaling PLMs to hundred billion parameters has led to models such as GPT-3 (Brown et al., 2020), which are significantly larger than the largest model tested in the results discussed above (i.e., GPT-J, with 6B parameters). Testing them on the entire set of COMPS stimuli (49K + 3 × 13.8K pairs of sentences) is prohibitively expensive since they are only accessible through paid APIs. Nonetheless, we sampled a small set of COMPS stimuli—which we term as miniCOMPS—in order to get a glimpse of how well substantially larger PLMs elicit property knowledge and demonstrate reasoning behavior compatible with property inheritance. Specifically, we created miniCOMPS by sampling 1200 minimal pairs from each of our original COMPS subsets (matched in terms of real world concepts and properties across the subsets), such that all pairs of nonce words in the resulting miniCOMPS-WUGS-DIST end up being sampled equal number of times (100 times each).

---

[10] https://github.com/kanishkamisra/minicons

[11] https://huggingface.co/models.

Family	Model (Abbrev.)	Parameters	Vocab Size	Tokenization	Corpora	Tokens
ALBERT	`albert-base-v2` (A-b)	11M	30,000	SentencePiece	WIKI and BC	3.3B
	`albert-large-v2` (A-l)	17M				
	`albert-xlarge-v2` (A-xl)	59M				
	`albert-xxlarge-v2` (A-xxl)	206M				
BERT	`distilbertbase-uncased` (dB-b)	67M	30,522	WordPiece	WIKI and BC	3.3B
	`bert-base-uncased` (B-b)	110M				
	`bert-large-uncased` (B-l)	345M				
ELECTRA	`electra-small` (E-s)	13M	30,522	WordPiece	WIKI and BC	3.3B
	`electra-base` (E-b)	34M				
	`electra-large` (E-l)	51M			WIKI, BC, CW, CC, and GIGA	33B
RoBERTa	`distilroberta-base` (dR-b)	82M	50,265	Byte-pair encoding	OWTC	2B
	`roberta-base` (R-b)	124M			BC, CC-NEWS, OWTC, and STORIES	30B
	`roberta-large` (R-l)	355M				
GPT2	`distilgpt2` (dGPT2)	82M	50,257	Byte-pair encoding	OWTC	2B
	`gpt2` (GPT2)	124M	50,257		WEBTEXT	8B*
	`gpt2-medium` (GPT2-m)	355M				
	`gpt2-large` (GPT2-l)	774M				
	`gpt2-xl` (GPT2-xl)	1.5B				
EleutherAI	`gpt-neo-125M` (Neo-125M)	125M	50,257	Byte-pair encoding	PILE	300B
	`gpt-neo-1.3B` (Neo-1.3B)	1.3B				380B
	`gpt-neo-2.7B` (Neo-2.7B)	2.7B				420B
	`gpt-j-6B` (GPT-J)	6B				402B

Table 3: Summary of the 22 models that we evaluate in this paper. **Legend for Corpora:** WIKI: Wikipedia; BC: BookCorpus (Zhu et al., 2015); CW: ClueWeb (Callan et al., 2009); CC: CommonCrawl GIGA: Gigaword (Graff et al., 2003); OWTC: OpenWebTextCorpus (Gokaslan and Cohen, 2019); CC-NEWS: CommonCrawl News (Nagel, 2016); STORIES: Stories corpus (Trinh and Le, 2018); WEBTEXT: WebText corpus (Radford et al., 2019); PILE: The Pile (Gao et al., 2020).
*As estimated by Warstadt et al. (2020).

COMPS subset	Stimulus	Score
BASE	A **horse** has hooves.	-3.829
	A **dog** has hooves.	-4.963
WUGS	A fep is a **horse**. Therefore, a fep has hooves.	-2.153
	A fep is a **dog**. Therefore, a fep has hooves.	-3.392
WUGS-DIST (**before**)	A wug is a dog. A fep is a horse. Therefore, a **fep** has hooves.	-2.919
	A wug is a dog. A fep is a horse. Therefore, a **wug** has hooves.	-2.895
WUGS-DIST (**in-between**)	A fep is a horse. A wug is a dog. Therefore, a **fep** has hooves.	-3.616
	A fep is a horse. A wug is a dog. Therefore, a **wug** has hooves.	-3.092

Table 4: An example of matched stimuli across different COMPS subsets, as well as conditional log-probabilities elicited by GPT-J. Here, the property of interest is *has hooves*, the positive concept is HORSE, and the negative concept is DOG. The negative concept in this case was sampled using the co-occurrence knowledge representation method (see §2.3). Emboldened words indicate items that are different in the minimal pair. Refer to §3.2 for discussion on how 'Score' is computed.

Models As test subjects, we chose four GPT-3 models (Brown et al., 2020): `ada`, `babbage`, `curie`, `davinci`, with the last one being the largest (at 175B parameters), and an additional fifth `davinci`-based model called `text-davinci-001`, which fine-tunes `davinci` on human-written demonstrations. We also test the recently proposed GPT-3.5 models, `text-davinci-002` and `text-davinci-003`, which improve over `davinci` by additionally fine-tuning it on code *and* human-

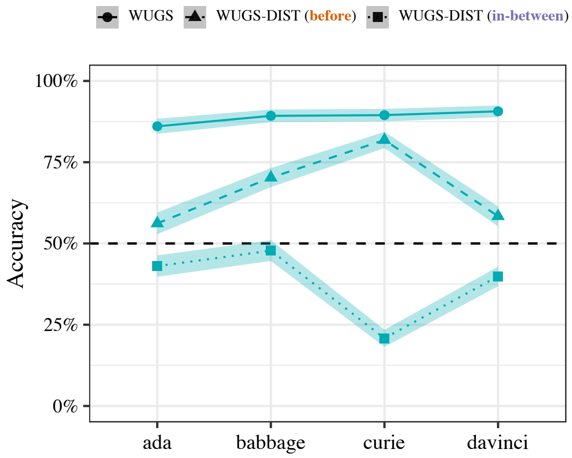


Figure 5: Accuracies of GPT-3 models (arranged in increasing order of the number of trained parameters) on miniCOMPS-WUGS and miniCOMPS-WUGS-DIST. Black dashed line indicates chance performance (50%). Error bands indicate 95% Bootstrap CIs.

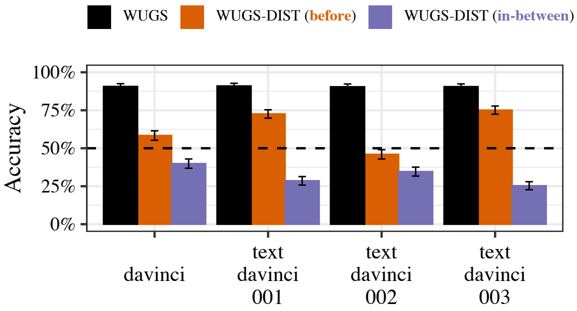


Figure 6: Accuracies of `davinci` models (GPT-3 and GPT-3.5) on miniCOMPS-WUGS and miniCOMPS-WUGS-DIST. Black dashed line indicates chance performance (50%). Error bars indicate 95% Bootstrap CIs. `davinci` and `text-davinci-001` are GPT-3 (Brown et al., 2020) models, while `text-davinci-002` and `text-davinci-003` are GPT-3.5 models.

written demonstrations (Ouyang et al., 2022).[12] All these models are autoregressive in nature, so we use the same scoring and evaluation method as described in §3.2. Since the four original GPT-3 models (`ada`, `babbage`, `curie`, `davinci`) are trained using the same LM objective on the same corpora, we analyze them separately from `text-davinci-001`, `text-davinci-002`, and `text-davinci-003`, which we only compare to `davinci`. We do this to remain consistent with the way we displayed results in §4—ordering models based on their number of trained parameters—and also because models in the `text-davinci-XXX` series use the same underlying `davinci` model augmented with additional training mechanisms (e.g., reinforcement learning and fine-tuning on human-feedback) and data (e.g., code) instead of increasing its size, to our knowledge.

Results Figure 5 shows the performance of the four GPT-3 models on miniCOMPS-WUGS and miniCOMPS-WUGS-DIST, while Figure 6 compares GPT-3 `davinci` to its code and human-feedback adapted counterparts. From Figure 5, we see robustness issues to persist even for GPT-3 models, similar to our main results. Models perform remarkably well in the absence of distraction (i.e., on miniCOMPS-WUGS), but struggle in its presence, especially when it is closer to the queried property. In particular, performance on miniCOMPS-WUGS-DIST (**before**) increases with an increase in parameters until the largest model (`davinci`), where the performance drops closer to chance. On miniCOMPS-WUGS-DIST (**in-between**), all models perform catastrophically worse than chance. This noteworthy pattern of proximity-based degradation in performance mimics the results shown in Figure 4, though we do not see a systematic decline in performance with an increase in parameters as observed in the GPT2 and EleutherAI models—with the 175B parameter model (`davinci`) demonstrating an increase in performance over the relatively smaller `curie` model.

While the above results demonstrate that simply scaling autoregressive PLMs is unlikely to overcome the lack of robustness against distracting content, we now test whether augmenting these large PLMs by additionally training on code (GPT-3.5 models) and aligning them with human-provided demonstrations (`text-davinci-001` and both GPT-3.5 models) could lead to any improvements. For instance, training on code could provide training signals to PLMs that encourage entity tracking, which could potentially enable them, in our case, to resolve which subordinate concept (e.g., *wug* vs. *dax*) the target property is more likely to be associated with. Similarly, aligning with human-written demonstrations could potentially improve their truthfulness, which in our case, could lead to them to prefer correct property assignments. However, from Figure 6, we see no noteworthy improvements demonstrated by these augmented models. All augmented models achieved similar ac-

---

[12] These models are also known as InstructGPT, as discussed in `https://platform.openai.com/docs/model-index-for-researchers`.

curacies on COMPS-WUGS as the `davinci` model (within 90.5% and 91%), suggesting that their augmentations preserved the general associations between the lexical items that denote everyday concepts and properties. On stimuli containing distraction (i.e., both subsets of COMPS-WUGS-DIST), either the models performed systematically worse as compared to `davinci` (with `text-davinci-002` showing below-chance performance on both subsets), or they showed mixed results, where an improvement on COMPS-WUGS-DIST (**before**) was accompanied by a decline on COMPS-WUGS-DIST (**in-between**).

Together, these results suggest that neither an increase in scale nor additional training methods such as alignment with human instructions/feedback or training on code prevents models from being distracted in associating properties to novel subordinate concepts introduced in the input context. In fact, the catastrophic effects of proximity-based distraction persists even for the most recent state of the art GPT-3/3.5 models.

C.2 Results with alternate nonce words

Here we report results on COMPS-WUGS and COMPS-WUGS-DIST using an alternate set of nonce words, which we constructed by sampling (with replacement) from 26 lower-case ASCII alphabet characters. Specifically, we constructed novel character sequences—each assigned as a replacement for our original four nonce words—of lengths ranging from 4-8 by sampling in an alternate fashion from consonants (odd positions) and vowels (even positions).[13] A replication of Figure 4 using the stimuli with these newly sampled nonce words is shown in Figure 7. On comparing figures 4 and 7, we observe largely similar patterns of results on stimuli containing nonce words constructed using randomly sampled characters. That is, models generally performed well on COMPS-WUGS, while they struggled on COMPS-WUGS-DIST. There were some exceptions: (1) GPT-Neo 1.3B and 2.7B showed improvements (relative to the original stimuli) in cases where distraction is added closer to the queried property (i.e., **in-between**), though they still hover around chance performance, and additionally the performance of GPT-J, like in the original results is still substantially below chance; and (2) there were non-trivial improvements demonstrated by ALBERT models (large and xl) on the **before** subset of COMPS-WUGS-DIST, and BERT-large on the **in-between** subset of COMPS-WUGS-DIST.

Stimuli	**Avg. Agreement**
COMPS-WUGS	$93.1_{0.8}$
COMPS-WUGS-DIST (**before**)	$73.9_{4.6}$
COMPS-WUGS-DIST (**in-between**)	$73.0_{4.6}$
Overall	$80.0_{2.9}$

Table 5: Average agreement (× 100) in PLMs' preference on stimuli containing original and synthetically constructed nonce words.

To precisely quantify the difference between the two sets of results, we measured the agreement between the predictions of the PLMs for both sets of stimuli, taken as the proportion of minimal pairs in which the models' relative preference agree.

Figure 8 shows individual model agreement on COMPS-WUGS and COMPS-WUGS-DIST, while Table 5 shows agreement percentages averaged across all models. From these results we observe models to show greater robustness to the variability introduced by the choice of nonce words in stimuli with one novel concept (COMPS-WUGS) than in stimuli with multiple novel concepts (COMPS-WUGS-DIST). Despite this discrepancy, there is generally a high average agreement (80%) between a given model's set of decisions on stimuli with original and alternative nonce words.

C.3 Results with alternate templates

Here we report results on an alternate phrasing of our stimuli, where instead of using the original template for introducing novel concepts in context (*a wug is a* `[CONCEPT]`), we use: *A wug is a type of* `[CONCEPT]`, where *wug* indicates the novel concept. In all cases, we simply alter the template, keeping everything else constant, including the choice of nonce words.

Figure 9 shows accuracies of the models on stimuli with this alternate phrasing, while Figure 10 and Table 6 show individual and averaged overall agreement between models' preference on original and the alternatively-phrased stimuli, respectively. The agreement percentages between models' preferences are quite high (average agreement being 90%)—in fact even greater than the agreement observed as a result of altering the nonce words (Table 5), further cementing the robustness of our

---

[13] the resulting set of words is: {*ruhisin, kifosa, rosibif, lepuvu*}, still amounting to 12 unique ordered pairs in the COMPS-WUGS-DIST stimuli.

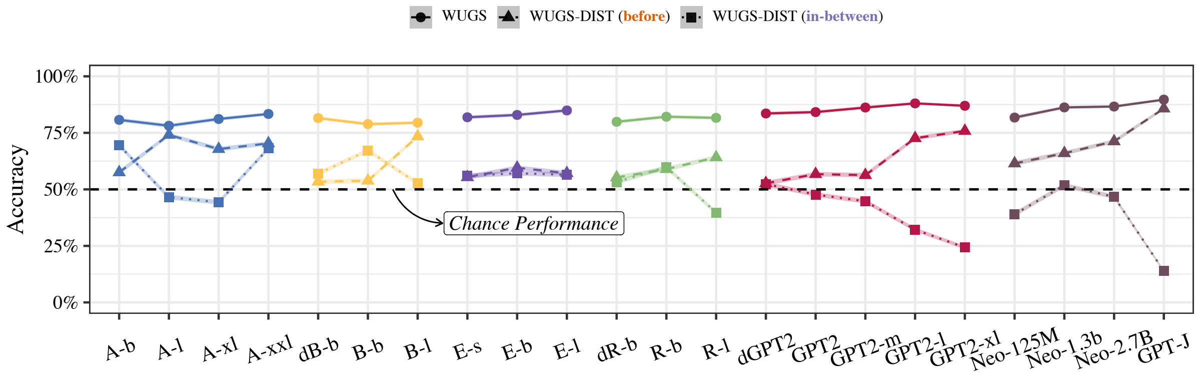
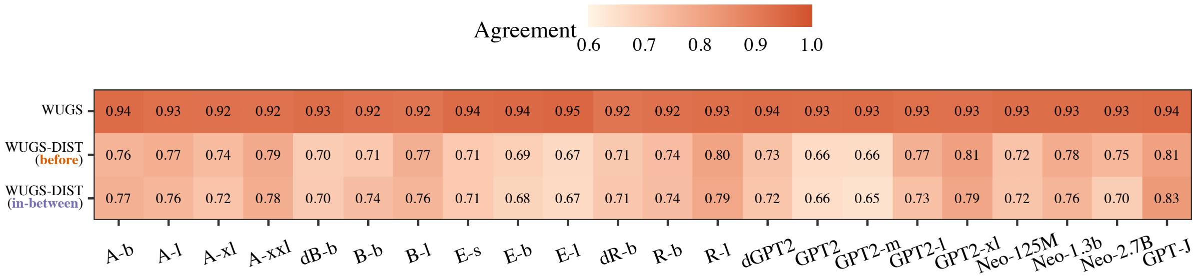


Figure 7: Accuracies of individual models (grouped by family, in increasing order based on number of parameters) on COMPS-WUGS and COMPS-WUGS-DIST with **synthetically constructed nonce words**. Black dashed line indicates chance performance (50%). Refer to Table 3 for unabbreviated model names.



	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
WUGS	0.94	0.93	0.92	0.92	0.93	0.92	0.92	0.94	0.94	0.95	0.92	0.92	0.93	0.94	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.94
WUGS-DIST (**before**)	0.76	0.77	0.74	0.79	0.70	0.71	0.77	0.71	0.69	0.67	0.71	0.74	0.80	0.73	0.66	0.66	0.77	0.81	0.72	0.78	0.75	0.81
WUGS-DIST (**in-between**)	0.77	0.76	0.72	0.78	0.70	0.74	0.76	0.71	0.68	0.67	0.71	0.74	0.79	0.72	0.66	0.65	0.73	0.79	0.72	0.76	0.70	0.83

Figure 8: Proportion of cases (in COMPS-WUGS and both subsets of COMPS-WUGS-DIST) **where each listed model's preference on the original stimuli matches that in stimuli with synthetically constructed nonce words**, measured as the 'Agreement'. An agreement of 1.0 suggests that a given model's preferences are perfectly matched across both sets of stimuli.

Stimuli	**Avg. Agreement**
COMPS-WUGS	$93.4_{1.2}$
COMPS-WUGS-DIST (**before**)	$88.5_{3.1}$
COMPS-WUGS-DIST (**in-between**)	$88.0_{3.6}$
Overall	$90.0_{2.6}$

Table 6: Average agreement (× 100) in PLMs' preference on stimuli containing original (*A wug is a* `[CONCEPT]`.) and alternate framing of novel taxonomic information (*A wug is a type of* `[CONCEPT]`.).

results.

C.4 How does performance on COMPS-BASE vary by property type?

Devereux et al. (2014) have categorized the properties that we use in our experiments to lie in 5 different categories: (1) **Taxonomic**, e.g., *is a mammal*, *is a vehicle*, etc.; (2) **Functional**, e.g., *can keep the body warm*, *is used to hit nails*, etc.; (3) **Encyclopedic**, e.g., *uses electricity*, *is warm blooded*, etc.; (4) **Visual Perceptual**, e.g., *has webbed feet*, *has thick fur*, etc.; and (5) **Other Perceptual**, e.g., *makes grunting sounds* and *is sharp*, etc. We report results of the 22 PLMs on the COMPS-BASE stimuli across the five different property types, in Figure 11.

From Figure 11, we observe that PLMs are substantially stronger in eliciting taxonomic properties of concepts as compared to other types, with highest overall accuracy being 70%, as compared to 48% on encyclopedic properties, 50% on visual perceptual properties, 57% on functional properties, and 43% on non-visual perceptual properties. Recall that chance accuracy for the 'Overall' scenario is just 6.25%, so these scores are fairly high. This corroborates evidence from previous work in analyzing property knowledge of distributional semantic models as well as LM representations to lack perceptual knowledge (Lucy and Gauthier, 2017; Da and Kasai, 2019; Rubinstein et al., 2015; Weir et al., 2020), likely due to reporting bias (Gordon and Van Durme, 2013; Shwartz and Choi, 2020).

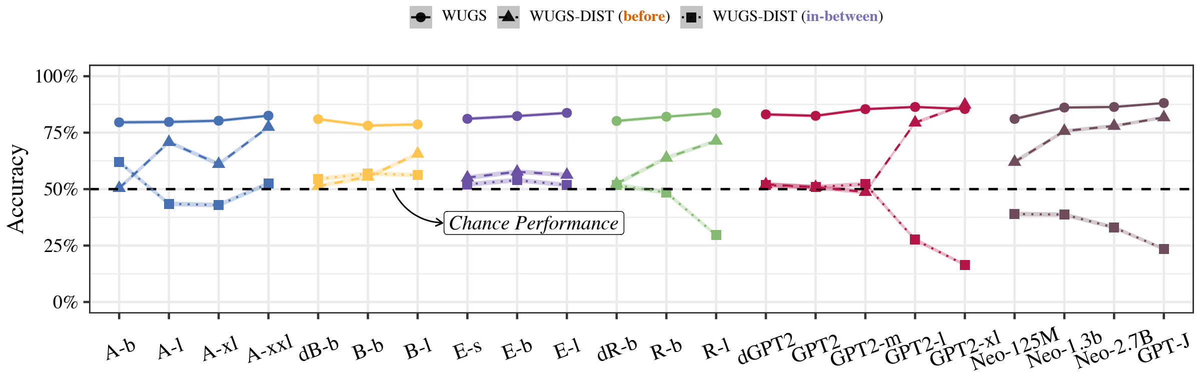
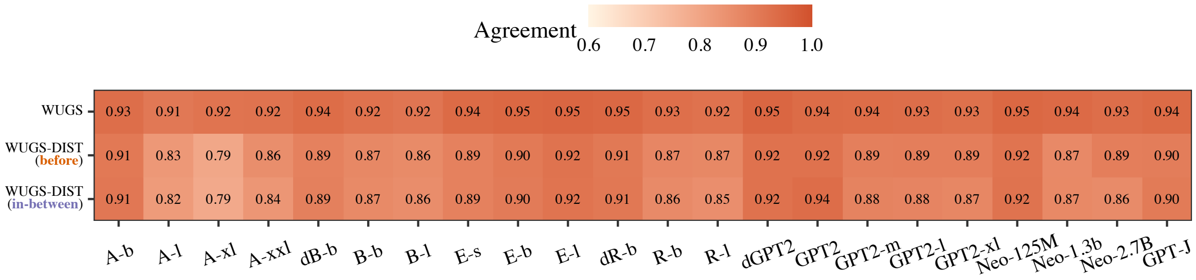


Figure 9: Accuracies of individual models (grouped by family, in increasing order based on number of parameters) on COMPS-WUGS and COMPS-WUGS-DIST with **alternate framing of novel taxonomic information**. Black dashed line indicates chance performance (50%). Refer to Table 3 for unabbreviated model names.



	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
WUGS	0.93	0.91	0.92	0.92	0.94	0.92	0.92	0.94	0.95	0.95	0.95	0.93	0.92	0.95	0.94	0.94	0.93	0.93	0.95	0.94	0.93	0.94
WUGS-DIST (**before**)	0.91	0.83	0.79	0.86	0.89	0.87	0.86	0.89	0.90	0.92	0.91	0.87	0.87	0.92	0.92	0.89	0.89	0.89	0.92	0.87	0.89	0.90
WUGS-DIST (**in-between**)	0.91	0.82	0.79	0.84	0.89	0.87	0.86	0.89	0.90	0.92	0.91	0.86	0.85	0.92	0.94	0.88	0.88	0.87	0.92	0.87	0.86	0.90

Figure 10: Proportion of cases (in COMPS-WUGS and both subsets of COMPS-WUGS-DIST) **where each listed model's preference on the original stimuli matches that in stimuli with alternate framing of novel taxonomic information**, measured as the 'Agreement'. An agreement of 1.0 suggests that a given model's preferences are perfectly matched across both sets of stimuli.

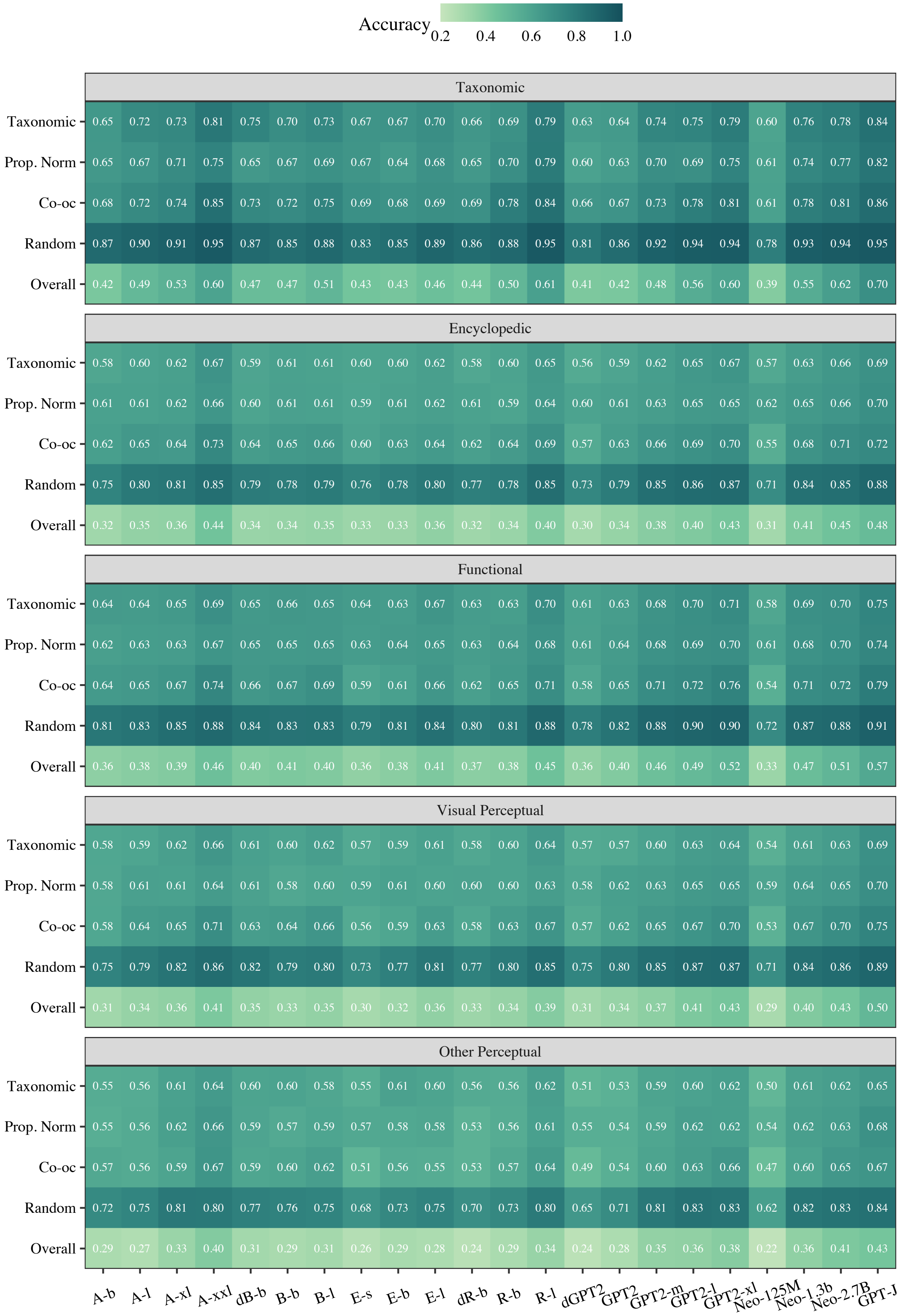
However, different to most of these works, the gap between performance on perceptual properties and non-perceptual properties is small. We conjecture that this could be primarily due to the extension of the CSLB by Misra et al. (2022), which lead to an increase in coverage of property knowledge for several properties. For instance, the property *has teeth* was mentioned only for 45 out of 67 potential concepts, having been left out for concepts such as CALF,[14] BUFFALO, KANGAROO, etc. So it could be the case that previous research has underestimated the extent to which property knowledge is encoded by PLMs and other distributional semantic models of language.

C.5 Does performance on COMPS-BASE depend on scale?

We plot the accuracies of PLMs on COMPS-BASE per model family (in order to control for differences in training corpora and tokenization) in Figure 12. In all families except BERT, we see that accuracy increases with the model size, following standard scaling laws. We notice that distilBERT-base (Sanh et al., 2019) is able to outperform even BERT-large on stimuli with 'Random' negative samples, suggesting that pruning BERT might sometimes unintentionally improve the model's ability to associate properties and concepts. We do however caution against interpreting these results as robust conclusion for scaling laws on COMPS-BASE. Such an endeavor would require comparing performance of models across multiple checkpoints with varying number of parameters, paired with rigorous statistical inference (Sellam et al., 2021; Zhang et al., 2021).

---

[14] the young one of a cow, and not the muscles in the vertebrate body



Taxonomic

	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
Taxonomic	0.65	0.72	0.73	0.81	0.75	0.70	0.73	0.67	0.67	0.70	0.66	0.69	0.79	0.63	0.64	0.74	0.75	0.79	0.60	0.76	0.78	0.84
Prop. Norm	0.65	0.67	0.71	0.75	0.65	0.67	0.69	0.67	0.64	0.68	0.65	0.70	0.79	0.60	0.63	0.70	0.69	0.75	0.61	0.74	0.77	0.82
Co-oc	0.68	0.72	0.74	0.85	0.73	0.72	0.75	0.69	0.68	0.69	0.69	0.78	0.84	0.66	0.67	0.73	0.78	0.81	0.61	0.78	0.81	0.86
Random	0.87	0.90	0.91	0.95	0.87	0.85	0.88	0.83	0.85	0.89	0.86	0.88	0.95	0.81	0.86	0.92	0.94	0.94	0.78	0.93	0.94	0.95
Overall	0.42	0.49	0.53	0.60	0.47	0.47	0.51	0.43	0.43	0.46	0.44	0.50	0.61	0.41	0.42	0.48	0.56	0.60	0.39	0.55	0.62	0.70

Encyclopedic

	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
Taxonomic	0.58	0.60	0.62	0.67	0.59	0.61	0.61	0.60	0.60	0.62	0.58	0.60	0.65	0.56	0.59	0.62	0.65	0.67	0.57	0.63	0.66	0.69
Prop. Norm	0.61	0.61	0.62	0.66	0.60	0.61	0.61	0.59	0.61	0.62	0.61	0.59	0.64	0.60	0.61	0.63	0.65	0.65	0.62	0.65	0.66	0.70
Co-oc	0.62	0.65	0.64	0.73	0.64	0.65	0.66	0.60	0.63	0.64	0.62	0.64	0.69	0.57	0.63	0.66	0.69	0.70	0.55	0.68	0.71	0.72
Random	0.75	0.80	0.81	0.85	0.79	0.78	0.79	0.76	0.78	0.80	0.77	0.78	0.85	0.73	0.79	0.85	0.86	0.87	0.71	0.84	0.85	0.88
Overall	0.32	0.35	0.36	0.44	0.34	0.34	0.35	0.33	0.33	0.36	0.32	0.34	0.40	0.30	0.34	0.38	0.40	0.43	0.31	0.41	0.45	0.48

Functional

	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
Taxonomic	0.64	0.64	0.65	0.69	0.65	0.66	0.65	0.64	0.63	0.67	0.63	0.63	0.70	0.61	0.63	0.68	0.70	0.71	0.58	0.69	0.70	0.75
Prop. Norm	0.62	0.63	0.63	0.67	0.65	0.65	0.65	0.63	0.64	0.65	0.63	0.64	0.68	0.61	0.64	0.68	0.69	0.70	0.61	0.68	0.70	0.74
Co-oc	0.64	0.65	0.67	0.74	0.66	0.67	0.69	0.59	0.61	0.66	0.62	0.65	0.71	0.58	0.65	0.71	0.72	0.76	0.54	0.71	0.72	0.79
Random	0.81	0.83	0.85	0.88	0.84	0.83	0.83	0.79	0.81	0.84	0.80	0.81	0.88	0.78	0.82	0.88	0.90	0.90	0.72	0.87	0.88	0.91
Overall	0.36	0.38	0.39	0.46	0.40	0.41	0.40	0.36	0.38	0.41	0.37	0.38	0.45	0.36	0.40	0.46	0.49	0.52	0.33	0.47	0.51	0.57

Visual Perceptual

	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
Taxonomic	0.58	0.59	0.62	0.66	0.61	0.60	0.62	0.57	0.59	0.61	0.58	0.60	0.64	0.57	0.57	0.60	0.63	0.64	0.54	0.61	0.63	0.69
Prop. Norm	0.58	0.61	0.61	0.64	0.61	0.58	0.60	0.59	0.61	0.60	0.60	0.60	0.63	0.58	0.62	0.63	0.65	0.65	0.59	0.64	0.65	0.70
Co-oc	0.58	0.64	0.65	0.71	0.63	0.64	0.66	0.56	0.59	0.63	0.58	0.63	0.67	0.57	0.62	0.65	0.67	0.70	0.53	0.67	0.70	0.75
Random	0.75	0.79	0.82	0.86	0.82	0.79	0.80	0.73	0.77	0.81	0.77	0.80	0.85	0.75	0.80	0.85	0.87	0.87	0.71	0.84	0.86	0.89
Overall	0.31	0.34	0.36	0.41	0.35	0.33	0.35	0.30	0.32	0.36	0.33	0.34	0.39	0.31	0.34	0.37	0.41	0.43	0.29	0.40	0.43	0.50

Other Perceptual

	A-b	A-l	A-xl	A-xxl	dB-b	B-b	B-l	E-s	E-b	E-l	dR-b	R-b	R-l	dGPT2	GPT2	GPT2-m	GPT2-l	GPT2-xl	Neo-125M	Neo-1.3b	Neo-2.7B	GPT-J
Taxonomic	0.55	0.56	0.61	0.64	0.60	0.60	0.58	0.55	0.61	0.60	0.56	0.56	0.62	0.51	0.53	0.59	0.60	0.62	0.50	0.61	0.62	0.65
Prop. Norm	0.55	0.56	0.62	0.66	0.59	0.57	0.59	0.57	0.58	0.58	0.53	0.56	0.61	0.55	0.54	0.59	0.62	0.62	0.54	0.62	0.63	0.68
Co-oc	0.57	0.56	0.59	0.67	0.59	0.60	0.62	0.51	0.56	0.55	0.53	0.57	0.64	0.49	0.54	0.60	0.63	0.66	0.47	0.60	0.65	0.67
Random	0.72	0.75	0.81	0.80	0.77	0.76	0.75	0.68	0.73	0.75	0.70	0.73	0.80	0.65	0.71	0.81	0.83	0.83	0.62	0.82	0.83	0.84
Overall	0.29	0.27	0.33	0.40	0.31	0.29	0.31	0.26	0.29	0.28	0.24	0.29	0.34	0.24	0.28	0.35	0.36	0.38	0.22	0.36	0.41	0.43

Figure 11: COMPS-BASE performance across five property types annotated in CSLB (Devereux et al., 2014).

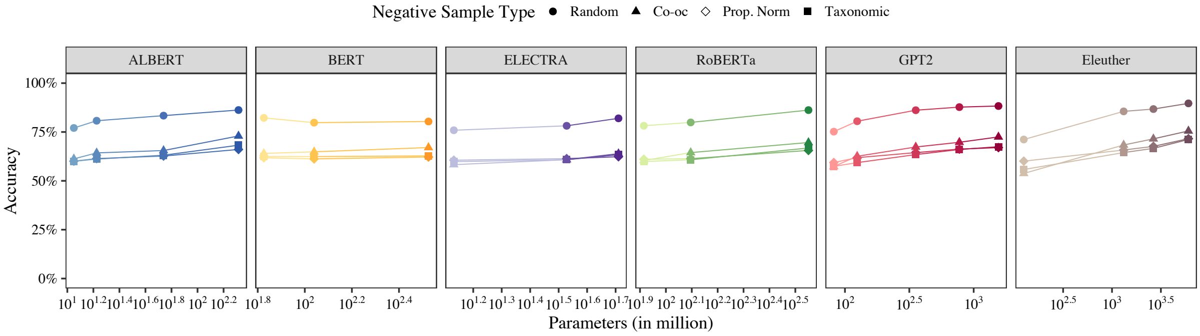


Figure 12: Accuracy vs. parameters across various negative sampling strategies. Models are shaded based on number of parameters.

Probabilistic Robustness for Data Filtering

Yu Yu[*] and **Abdul Rafae Khan**[*] and **Shahram Khadivi** [†] and **Jia Xu**[*]
[*] School of Engineering and Science, Steven Institute of Technology, NJ 07030, USA
[†] eBay Inc., Aachen 52064, Germany
`yyu50@stevens.edu, akhan4@stevens.edu, skhadivi@ebay.com, jxu70@stevens.edu`

Abstract

We introduce our probabilistic robustness rewarded data optimization (PRoDO) approach as a framework to enhance the model's generalization power by selecting training data that optimizes our probabilistic robustness metrics. We use proximal policy optimization (PPO) reinforcement learning to approximately solve the computationally intractable training subset selection problem. The PPO's reward is defined as our $(\alpha, \epsilon, \gamma)$-Robustness that measures performance consistency over multiple domains by simulating unknown test sets in real-world scenarios using a leaving-one-out strategy. We demonstrate that our PRoDO effectively filters data that lead to significantly higher prediction accuracy and robustness on unknown-domain test sets. Our experiments achieve up to +17.2% increase of accuracy (+25.5% relatively) in sentiment analysis, and -28.05 decrease of perplexity (-32.1% relatively) in language modeling. In addition, our probabilistic $(\alpha, \epsilon, \gamma)$-Robustness definition serves as an evaluation metric with higher levels of agreement with human annotations than typical performance-based metrics.

1 Introduction

Modern machine learning works with massive amounts of data on a range of tasks like language modeling, object detection, and data mining. Using large amounts of training set to build machine learning systems requires extensive computational resources and creates problems like domain shifts and input noise. These unfiltered training data harm model learning robustness (Frénay and Verleysen, 2013) that leads to prediction errors and serious consequences like self driving car fatality and medical misdiagnosis (Tian et al., 2018).

One problem causing this model instability is that the model learning is opt for the system's quality, which is typically evaluated by measuring how close this system's output of a test set is from its human label using metrics such as accuracy, error rate, perplexity, human evaluation score, and so on. However, such a system performance metric highly depends on the test set's choice and is thus unreliable. For instance, if our training set is drawn from the news domain, then the performance on a test set from the news domain (in-domain test set) is usually much higher than that from the Twitter domain (out-of-domain test set). As a result, in NLP, while some systems produce human parity results like the use of a pre-trained Transformer (Hendrycks et al., 2020) on in-domain test sets, these systems are easily corrupted by out-of-domain (OOD) samples from the real world.

Existing studies on data selection and robust learning demonstrate a need for test domain knowledge during training. Some data selection work (Moore and Lewis, 2010; Kirchhoff and Bilmes, 2014; van der Wees et al., 2017; Fan et al., 2017; Qu et al., 2019; Liu et al., 2019; Kang et al., 2020) chooses critical in-domain data for domain adaptation, and other work defends against adversarial attacks but offers little help for out-of-domain robustness (Taori et al., 2020) under natural distributional shifts (Wang et al., 2021) that occurs more frequently than extreme adversarial cases. This out-of-domain robustness is often measured by testing on a specific domain and a single task like sentiment classification (Müller et al., 2019; Hendrycks et al., 2020). The problem with these existing approaches is that the target domain knowledge is often unknown, as is the case for most real-world applications that do not know the domain of test data they will receive before launch.

To address these challenges, our goal is to select training data to achieve high accuracy on OOD test set, without requiring any target domain insight during data selection or model training process. We distinguish the out-of-domain and *unknown-domain test sets* by assigning the out-of-domain as the test domain known during the training, and the

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2950–2959
May 2-6, 2023 ©2023 Association for Computational Linguistics

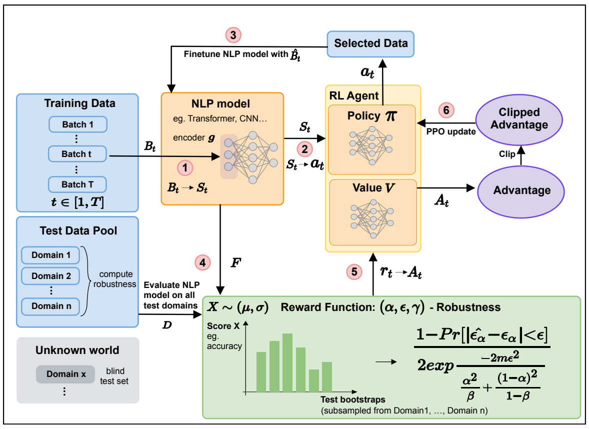


Figure 1: Probabilistic Robustness Rewarded Data Optimization (PRoDO) Framework.

unknown-domain as the test domain along with its information that are not known during the training. Practical applications often apply the latter case where we do not have any target domain knowledge, a condition we call "unknown-domain" robustness. To move our understanding forward, there is an urgent need to revisit out-of-domain robustness in "unknown worlds" to bridge the gap between laboratory observations and real-life results.

In our approach to the measurement of robustness on unknown domains, we define robustness as the consistency of the behavior of a machine learning system. The more a machine learning system's behavior deviates from the typical, the less robust the system is defined to be. Notice that this definition does not necessarily give a notion of whether system performance is good or bad. For example, in terms of sentiment analysis, this definition refers to consistency in prediction accuracy for a trained classifier.

To measure a system's performance consistency, we combine test sets for evaluation from various domains, like news, biomedical, and Twitter; randomly sample their subsets, and take each subset as this system's input and obtain an output. Then, we measure the performance of each output. If these output performances are close to each other, we say they are consistent. To quantify this consistency, we define our robustness metric as a notion of a probabilistic definition on the distribution of performance across different test domains, called $(\alpha, \epsilon, \gamma)$**-Robustness**, where the higher the probability of the consistent prediction performance, the more robust the system.

Our objective is to measure the probability of upper bounding the prediction accuracy gap between any test subset and its average. More specifically, we call an NLP system $(\alpha, \epsilon, \gamma)$-robust, if for every subset uniform-randomly drawn from a distribution, its prediction error, a combination of the target domain error and the source domain error weighted by the parameter α, is centered around the mean error, which is bounded through a parameter ϵ with a probability depending on γ, an indicator of how robust a system is, see definition in Figure 1 (in green) and details in Section 2.2.

In our approach, we *do not need any target-domain data* since we "simulate" unknown target-domain test sets using the *leave-one-out* error stability (Mukherjee et al., 2006). We assume that the non-left-out test sets are the simulated target domain while the left-out test is the real target domain for evaluation. For example, given biomecial as our unknown target domain to evaluate, we take the training data as the source-domain set and sample different subsets from a combination of other test domain data (e.g., news, TED talks, etc.) to simulate our target-domain test sets. The hyperparameter α is the target-domain error weight and offers flexibility to balance the trade-offs between source- and target-domain errors.

The $(\alpha, \epsilon, \gamma)$-Robustness takes a new direction away from adversarial robustness to a general consistency of a model's quality as meta-evaluation methods that measure the consistency of user-defined quality metrics. Thus, *any standard performance evaluation metrics, such as accuracy, can be used within the definition of our robustness.*

After defining our robustness metric as our data optimization goal, we ask, *how should we select a subset of data that can maximize the robustness?* We assert that in general, the subset selection problem is computationally intractable. We conjecture that this condition holds true for every objective function, including our notion of probabilistic robustness, and is the reason why we use Proximal Policy Optimization (PPO) (Schulman et al., 2017) deep reinforcement learning to optimize the training set, which has the advantages of low variance, monotonic policy improvement, and sampling efficiency (Schulman et al., 2015; Uc-Cetina et al., 2021) compared to A2C (Konda and Tsitsiklis, 2000) and policy gradient (Sutton et al., 1998). Our Probabilistic Robustness rewarded Data Optimization (PRoDO) framework equipartitions the training data into mini-batches and simultaneously learns a policy network to select data iteratively and a value network to estimate future returns using our $(\alpha, \epsilon, \gamma)$-Robustness as the reward functions illustrated in Figure 1 .

Our empirical results on sentiment analysis and language modeling show that the use of our robustness definition consistently and significantly enhances a model's out-of-domain performance. The main contributions of this work include:

1. Probabilistic Robustness definition of $(\alpha, \epsilon, \gamma)$-Robustness;
2. The creation of PPO Deep Reinforcement Learning framework for data selection;
3. The improvement of NLP model accuracy and out-of-domain generalization on showcase applications of sentiment analysis and language modeling.

The rest of the paper is as follows: In Section 2.1, we describe our PRoDO framework. In Section 2.2, we introduce the $(\alpha, \epsilon, \gamma)$-Robustness. Section 3 introduces experimental details including baselines, NLP tasks and ablation study. In Section 4, we discuss the previous literature on robustness in machine learning. Section 5 concludes the paper.

2 Method

Our goal is to enable our task model $\mathcal{F}$ (any Deep Learning-based NLP model, such as Transformer, etc.,) to achieve consistent while superior performance on any unknown test domain $\mathcal{D}_x$ whose distribution is different from the source training set $\mathcal{X}$, by learning an effective subset of $\mathcal{X}$ that can maximize the robustness of model $\mathcal{F}$.

The entire process details are depicted in Figure 1. Our method consists of reinforcement learning (RL), data selection, and training NLP models using the selected data. Specifically, we use reinforcement learning to train a data selection policy, and we use the data selection policy to select a subset of training data to fine-tune NLP models. Following Yu et al. (2022a), we pre-train the task model $\mathcal{F}$ on the full training data set $\mathcal{X} = \{x_i\}_{i=1}^{N}$, where x_i is a sentence, N is training set size. Then, the training set $\mathcal{X}$ is shuffled and randomly partitioned into T disjoint data batches such that $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$, with $\mathcal{B}_t = \{x_{(t-1)N|T+1}, x_{(t-1)N|T+2}, ..., x_{tN|T}\}$, where $N|T$ is the integer division of N by T, and $T \leq t$. For each batch, we select a subset of data $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | x_i \in \mathcal{B}_t\}$ with size o according to the data selection policy trained by reinforcement learning, and use it to fine-tune the model $\mathcal{F}$. In general, $\mathcal{F}$ and its encoder g are updated on $\hat{\mathcal{B}}_t$ for T times in an epoch, and each update is based on the previous checkpoint. Besides training set, we use a test data pool $\mathcal{D}$ containing n test domains $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n\}$ to simulate the real world scenario and compute the robustness score with it.

In the following sections, we will first introduce how our PRoDO framework learns to select data (Section 2.1) and then our reward function based on the probabilistic robustness definition (Section 2.2).

2.1 PRoDO framework

We now present the details of our Probabilistic Robustness Rewarded Data Optimization (PRoDO) framework.

2.1.1 RL training

The goal of our reinforcement learning agent is to learn an optimal data selection policy π to maximize the expected return $\mathcal{R}_t = \sum_{j=0}^{T-t} \gamma^j r_{t+j}$ from each state s_t, where the scalar reward r_t measures how good the action a_t taken by the policy is at the time step t and $\gamma \in [0, 1]$ is the discount factor. Specifically, each time step can be split into six steps as in Figure 1. Firstly, the encoder (e.g. an embedding layer in LSTM, or an encoder in transformer) inside the NLP model transforms the batch of raw data $\mathcal{B}_t$ into a batch of (document) embeddings, denoted as s_t. Secondly, the agent takes action a_t based on state s_t. The agent takes the state s_t as input and outputs a probability distribution for s_t, so that each sentence is associated with a probability, representing how likely it is going to be selected. The selected subset, denoted as $\hat{\mathcal{B}}_t$, is then obtained by Bernoulli sampling each sentence in the state s_t. The result of Bernoulli sampling is represented as an action vector a_t, where each value in it is either 0 or 1 representing each sentence in the batch not being or being selected. Thirdly, as soon as we obtain $\hat{\mathcal{B}}_t$, the NLP model $\mathcal{F}$ as well as encoder g are fine-tuned by the selected subset $\hat{\mathcal{B}}_t$. Then, the scalar reward $r_t = \mathcal{R}(\mathcal{D}, \mathcal{F})$ is calculated by our reward functions $\mathcal{R}$ (defined in Section 2.2) based on all available test domains D and current NLP model $\mathcal{F}$. Next, the advantage A_t over action a_t is computed by the difference of reward r_t and the output of value function $V(s_t)$. Finally, we update the policy function following the gradient with regard to the objective in PPO (details in next section 2.1.2) using the advantage A_t.

2.1.2 PPO

In the conventional policy based (Sutton et al., 1998) or actor-critic (policy-value) (Mnih et al., 2016) based reinforcement learning, the precision of the value function often corrupts the policy optimization process for two reasons. First, some collected states might introduce noise to the prediction of $V(s_t)$, and thus lead to an inaccurate estimate of advantage A_t following with an inaccurate update of policy gradient. Secondly, a trajectory of interaction (consider a large dataset with a large interaction horizon $\mathcal{T}$) might take long time, while one collection of data can only be used to update the policy once, thus leading to severe sample inefficiency.

Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) and Proximal Policy Optimization (PPO)(Schulman et al., 2017) are proposed to solve the aforementioned problems by introducing importance sampling and advantage clipping. We adopt PPO as our framework since it is much simpler to implement than TRPO. PPO has the following properties, which are very desirable to achieve our goals: low variance, monotonic policy improvement and sampling efficiency (Grondman et al., 2012; Schulman et al., 2017). Our framework consists of policy and value networks jointly and dynamically learned together with the task model using the advantage error computed from the reward function, as shown in Algorithm 1. PPO uses $A(s_t, a_t)$, the advantage of action a_t in state s_t to scale the policy gradient. Specifically, the advantage of action (Mnih et al., 2016) a_t in state s_t is defined as

$$A(s_t, a_t) = \mathcal{Q}(s_t, a_t) - \mathcal{V}(s_t) \approx \sum_{j=0}^{T-t} \gamma^j r_{t+j} - \mathcal{V}(s_t), \tag{1}$$

where $\gamma \in (0, 1]$ is the discounting factor set as 0.99. $\mathcal{V}$ is the value function implemented as a value network.

Let $\mathbf{r}_t(\theta)$ denote the probability ratio $\mathbf{r}_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}$, the objective of PPO is defined in Schulman et al. (2017) as

$$A_{t_{clipped}} = clip(\mathbf{r}_t(\theta), 1 - \epsilon, 1 + \epsilon) A_t \tag{2}$$

$$\mathcal{J}(\theta) = \mathbb{E}_t \left[\min(\mathbf{r}_t(\theta) A_t, A_{t_{clipped}}) \right], \tag{3}$$

where ϵ is a hyperparameter, and we set it as 0.2. The objective function clips the range of change of policy gradient into $[1 - \epsilon, 1 + \epsilon]$, which forces the new policy to not deviate too much from the old policy. This clipping design of PPO ensures monotonic improvement and thus it has the advantages of sample efficiency and ease of tuning compared to other policy-based algorithms.

Algorithm 1 PRoDO Training Algorithm

Input: Epoch L, learning rate α, discount factor γ, training set $\mathcal{X}$, pre-trained task model $\mathcal{F}$(including encoder g), reward function $\mathcal{R}$ (discussed in section 3.2)

Output: selected data, fine-tuned $\mathcal{F}$, policy π_θ, data value estimator $\mathcal{V}_{\theta_v}$

```
Initialize data selection policy $\pi_\theta$ and value estimator $\mathcal{V}_{\theta_v}$
for episode $l = 1$ to $L$ do
    Shuffle (uniformly at random) all training samples;
    Equipartition $\mathcal{X}$ into $T$ (disjoint) sets with same size $n|T$: $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^T = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$;
    Initialize an empty list: episode history $\Upsilon$
    for all $\mathcal{B}_t \in \mathcal{X}$ (uniform transition probability) do
        $s_t = g_t(\mathcal{B}_t)$;
        Obtain batch action $a_t$ by sampling based on $\pi_\theta(s_t)$;
        $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^o | a_i = 1\}$, where $o$ is selected sample size;
        Update task model $\mathcal{F}(g_t)$ by fine-tuning on $\hat{\mathcal{B}}_t$;
        $r_t = \mathcal{R}(\hat{\mathcal{B}}_t, \mathcal{F})$;
        Store $(s_t, a_t, r_t)$ to episode history $\Upsilon$;
    end for
    for all $(s_t, a_t, r_t) \in \Upsilon$ do
        Obtain $A(s_t, a_t)$ for each batch;
        Update policy weights $\theta$ and value estimator weights $\theta_v$;
    end for
    Clear episode history $\Upsilon$;
end for
return $\mathcal{F}$,$\pi_\theta$ and $\mathcal{V}_{\theta_v}$
```

The objective of value network is:

$$\nabla_{\theta_v} \mathcal{V}(\theta_v) = \mathbb{E}_{\pi_\theta} \nabla_{\theta_\mathcal{V}} (r_t - \mathcal{V}(s_t; \theta_v))^2 \tag{4}$$

The parameters of value function θ_v is updated by:

$$\theta_{v(t+1)} = \theta_{vt} + \alpha \nabla_{\theta_{vt}} (r_t - \mathcal{V}(s_t; \theta_{vt}))^2 \tag{5}$$

2.2 Reward function: $(\alpha, \epsilon, \gamma)$-Robustness

The reward function in Section 2.1 is the robustness of the NLP model $\mathcal{F}$.

In the real world, the out-of-domain data are often much less than the in-domain data. Formally, consider a pool of m samples where βm samples are drawn from *target* domain D_t and $(1 - \beta)m$ samples are drawn from *source* domain D_s where $\beta \in [0, 1)$, β is often small in many scenarios. Thus, the minimization of empirical target error $\hat{\epsilon}_t$ becomes hard with the constraint of limited target data. To solve this, we instead try to minimize a convex combination of empirical source and target error:

$$\hat{\epsilon}_\alpha = \alpha \hat{\epsilon}_t + (1 - \alpha) \hat{\epsilon}_s \tag{6}$$

where $\alpha \in [0, 1]$. We also denote ϵ_α as the weighted combination of the true source and target errors measured with respect to source domain D_s and target domain D_t, as shown in Figure 2.

According to the learning theory of Ben-David et al. (2010), the probability that the difference between the weighted empirical error $\hat{\epsilon}_\alpha$ and the weighted true error ϵ_α of an NLP system exceeds a given threshold has an upper bound as shown in Equation 7. Since we are more interested in bounding the difference between the weighted empirical error $\hat{\epsilon}_\alpha$ and the weighted true error ϵ_α to some threshold, so as to consider the system as "robust", we can transform the inequality to derive a lower bound of the probability as shown in Equation 9:

$$Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| \geq \epsilon] \leq 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}}} \tag{7}$$

$$\Leftrightarrow 1 - Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| \geq \epsilon] \geq 1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}}} \tag{8}$$

$$\Leftrightarrow Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| < \epsilon] \geq 1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}}} \tag{9}$$

The right hand side of Equation 9 is the lower bound of the probability that the difference of the empirical error and the true error of an NLP system is smaller than some threshold ϵ. We can introduce the robustness factor γ ($\gamma \in [0, 1]$) to the right hand side to control how tightly we would like to bound the probability of error difference:

$$Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| < \epsilon] \geq 1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}}} \cdot \gamma \tag{10}$$

$$\Leftrightarrow \gamma \geq \frac{1 - Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| < \epsilon]}{2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}}}} \tag{11}$$

In consequence, we give the formal definition of $(\alpha, \epsilon, \gamma)$-robustness as:

Definition *We call an NLP system* $(\alpha, \epsilon, \gamma)$*-robust, if for any source domain* D_s *and target domain* D_t*, the difference between the empirical error* $\hat{\epsilon}_\alpha$ *and the true error* ϵ_α *is bounded through a threshold parameter* ϵ *with a probability of* $1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta}+\frac{(1-\alpha)^2}{1-\beta}}} \cdot \gamma$*, where* $\alpha \in [0, 1]$ *is the weight of target domain error and* $\beta \in [0, 1)$ *is the ratio of target data within all data.*

Based on this definition, we can interpret $\gamma \in [0, 1]$ **as the inverse indicator of robustness** for an NLP system. For example, a larger γ indicates a lower probability that the difference between empirical error $\hat{\epsilon}_\alpha$ and true error ϵ_α is within our expected threshold, so that the empirical error is probably with large variance, and thus the large γ indicates a less robust system.

2.2.1 Compute robustness by bootstrapping

We estimate robustness metrics using the *leave-one-out* error stability (Mukherjee et al., 2006) by excluding a left-out test set from all available datasets where we measure robustness. More precisely, for a given model, we randomly select one leaving-one-out test set and then combine all other tests to compute the robustness of the left-out datasets. Specifically, we consider the test domain as the target domain (an unknown domain that will not be used in training) and use all other test domains as available resources to compute mean and variance.

In practice, it often occurs that only limited test domains are available. Thus, the test scores of test domains are discrete values and hard to form a distribution. To solve this problem, we seek to a modified bootstrap algorithm to construct a pool of subsamples from the combined test set, as proposed by Yu et al. (2022b). For each subsample, instead of random sampling from the complete test pool, we randomly sample from the elements not present in the current subsample to minimize the intersection between each.

3 Experiments

3.1 Baselines

We compare our measure with four baselines. The first baseline denoted as **All** is the normal NLP model trained on all training samples. **Minmax** is derived from classic min-max objective in robust optimization. **Diff** (Zhang et al., 2022) and **Ratio** (Niu et al., 2020) are recently introduced robustness metrics defined by input perturbations.

Min-max robustness The notion of robustness can be originated from robust optimization (Ben-Tal and Nemirovski, 1998; Bertsimas and Sim, 2004) in which the optimization goal is to find a solution h satisfying

$$\min_h [\max_{\delta_1, \ldots \delta_n \in \Delta} \sum_{i=1}^{n} l(h, x_i + \delta_i)] \tag{12}$$

where x_i are the observed training samples, δ_i are deviations or perturbations from the observed samples, $l(\dot{;})$ is the loss function, and Δ are all possible

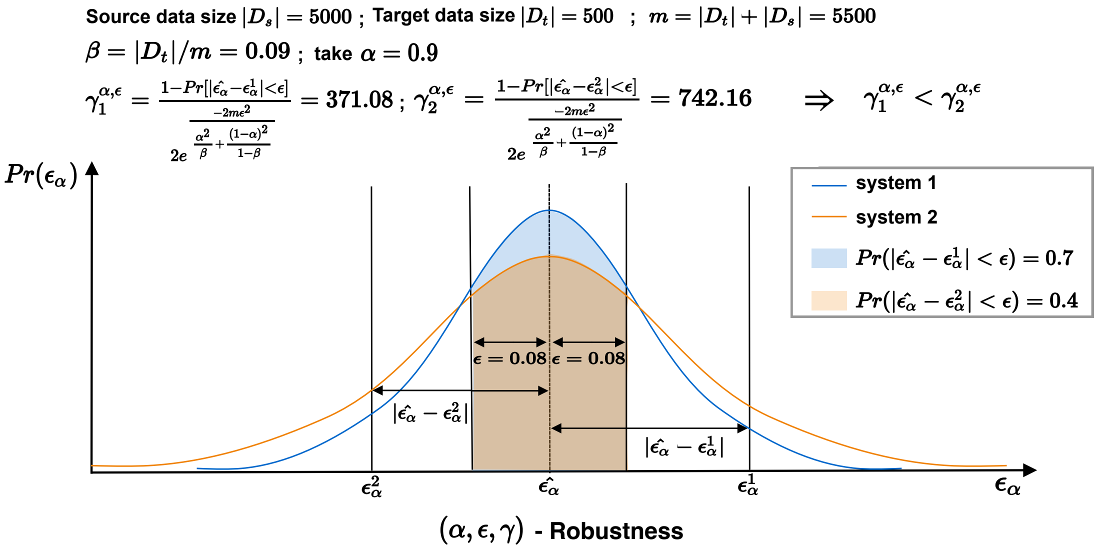


Figure 2: Illustrations of $(\alpha, \epsilon, \gamma)$-Robustness for two NLP systems. Orange curve and blue curve are the probability density function (PDF) of the test scores (X) of two NLP systems. The inference of robustness comparison is shown in top.

perturbations. Robust algorithms satisfying this definition are expected to minimize the empirical error under the worst possible perturbation. Since min-max objective targets the "worst possible perturbation", we consider the most challenging test set that will degrade the NLP performance most as "the worst possible perturbation" and use the evaluation score of such test set as a quantitative measure to denote model robustness under the min-max objective.

3.2 Improvements on NLP tasks

We experiment on a typical classification task of sentiment analysis and a generation task of language modeling.

3.2.1 Sentiment Analysis

In the task of sentiment analysis, we train a CNN classifier (Kim, 2014) using Amazon product review dataset (Blitzer et al., 2007). Specifically, we use the combined set of DVD, kitchen and books domains as source data. To compute robustness metrics, we use the full pre-processed Amazon product review dataset, which contains other domains such as grocery, tools, beauty and computer. We treat the test accuracy on the bootstrapped subsamples as a distribution and compute the $(\alpha, \epsilon, \gamma)$-Robustness by setting ϵ as 0.001 and α as 0.9. From Table 1, the classifiers optimized with $(\alpha, \epsilon, \gamma)$-Robustness outperforms all baselines. Specifically, the Min-max robustness achieves the second highest average accuracy score on test domains.

3.2.2 Language Modeling

Our baseline is a Transformer language model (Vaswani et al., 2017) with default hyper-parameters. We experiment with two moderate size datasets WikiText-2 (Merity et al., 2016) and Penn Treebank. As for evaluation, we report perplexity scores on four translation datasets from different domains, IWSLT'17 (TED talk) (Cettolo et al., 2012), Biomedical'21 (medical) (Yeganova et al., 2021), MTNT'18 (Reddit) (Michel and Neubig, 2018) and WMT'15 (news). The baseline models are trained using the fairseq toolkit (Ott et al., 2019) and stop training until the validation perplexity score does not improve for 5 epochs. The evaluation results are shown in Table 2. The perplexity (PPL) on all test domains have been improved. Specifically, test perplexity of MTNT'18 has an improvement of 28.05 (32.1% relative improvement) compared to the best baseline (Niu et al., 2020).

3.3 Ablation study

3.3.1 Comparison with human evaluation

We give a case study to compare the robustness of four language models trained on four different domains from the OPUS dataset (Tiedemann, 2012) using proposed robustness metrics. We combine MTNT'18, BIO'21, WMT'15 and IWSLT'17 test sets as a test pool, then subsample 50% of the test pool for 30 times. Next, we collect PPL scores on each subsampled bootstrap, and compute the mean and variance for the PPL score distribution. Fig-

	auto	beauty	food	instruments	office	computer	tools	phones	grocery	jewelry	outdoor	avg
All	56.48	54.03	56.02	59.15	56.84	55.17	55.95	52.79	53.06	56.63	55.81	55.63
Minmax	72.56	55.91	76.24	74.70	80.08	60.90	82.50	57.68	69.01	60.56	70.50	69.14
Diff	51.77	63.78	75.99	66.77	79.01	47.23	81.93	50.39	64.28	71.74	63.48	65.12
Ratio	71.06	63.92	75.25	82.01	67.45	63.19	78.57	58.83	68.79	65.53	62.80	68.85
$(\alpha, \epsilon, \gamma)$	**79.26**	**67.07**	**82.28**	**85.47**	**84.67**	**68.68**	**89.29**	**62.60**	**73.45**	**77.25**	**75.45**	**76.86**
+%	8.20	3.15	7.03	3.46	17.22	5.49	10.72	3.77	4.66	11.72	12.65	8.01

Table 1: Sentiment analysis accuracy [%] on amazon unprocessed domains. Last row: absolute improvement between $(\alpha, \epsilon, \gamma)$ and Ratio (Niu et al., 2020). Last column: average accuracy over all domains.

	WikiText-2				Penn Treebank			
	IWSLT	BIO	MTNT	WMT	IWSLT	BIO	MTNT	WMT
All	328.23	259.47	274.17	296.27	147.03	117.17	93.82	104.55
Minmax	189.03	140.35	160.26	169.03	142.84	82.55	91.59	101.91
Diff	193.06	136.94	167.55	168.90	140.18	79.04	92.62	98.70
Ratio	195.73	134.15	158.35	160.34	143.37	80.66	87.21	102.18
$(\alpha, \epsilon, \gamma)$	**175.91**	**123.47**	**142.34**	**154.38**	**118.96**	**71.45**	**59.16**	**87.70**

Table 2: Language modeling: Perplexity on four test domains. First row: source training domain; Second row: test domains.

ure 3 shows the normalized γ values against each corresponding epsilon value for the four language models. We can observe the γ values for the model trained by medical are much smaller than the values for other models, and the γ values for office are the highest. This shows that the model trained by medical is the most robust among all models, while the model trained by office is the least robust.

Model 1	Model 2	γ	Human	Agree?
edu	office	edu	edu	YES
edu	medical	medical	medical	YES
edu	books	books	books	YES
office	medical	medical	medical	YES
office	books	books	books	YES
books	medical	medical	medical	YES

Table 3: Robustness Metrics pair-wise comparison on each two models.

To evaluate how our robustness measures match human judgments, we compare the rank given by Figure 3 with the rank given by perplexity score and the rank given by human annotators, as shown in Table 4. Each human annotator is assigned with two language models selected at random. The human annotators do not know any details about the model or the training data used for each model. She/He can only use these models to get two generated sentences for the same input prompt. This step can be repeated as many times as possible until the human annotator decides which model is more *consistent* in its generations. The decision is based on consistency of generation quality but rather the quality itself. The results of pair-wise comparison between two language models are shown in Table 3. Our $(\alpha, \epsilon, \gamma)$-robustness perfectly aligns with decisions made by human annotators, while the rank based on perplexity score fails to match human evaluation results.

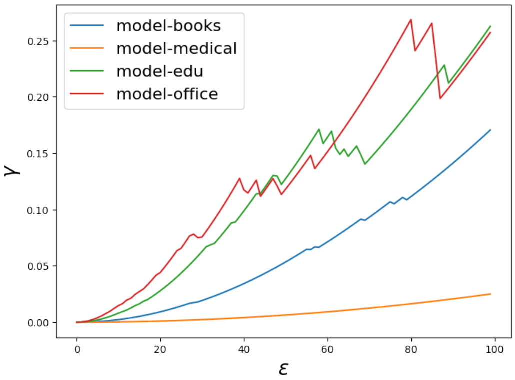


Figure 3: $(\alpha, \epsilon, \gamma)$-Robustness plot on four language models. "model-books" denotes the language model trained by the books domain from the OPUS dataset (Tiedemann, 2012)

3.3.2 Domain distance

Many data selection work use target domain knowledge to select best in-domain data samples that are close to target domain (Aharoni and Goldberg, 2020; Liu et al., 2019; Ma et al., 2019). Our method does not use target domain knowledge, thus, we question whether our selected data resembles target domain under such zero-knowledge setting. We follow Aharoni and Goldberg (2020) using DistilBert (Sanh et al., 2019) to embed all domain data, Penn Treebank data and our $(\alpha, \epsilon, \gamma)$-Robustness

Model	CE	PPL	rank$_{CE}$	rank$_{PPL}$	rankγ	rank$_{Human}$
edu	11.34	2474.16	1	1	3	3
books	12.03	4039.17	2	2	2	2
office	12.06	4181.39	3	3	4	4
medical	13.41	9426.04	4	4	1	1

Table 4: $(\alpha, \epsilon, \gamma)$-Robustness 100% agrees with human ranking, while perplexity (PPL) and cross entropy (CE) 25% agree with human. "office" means the model is trained on the office domain.

	BIO	MTNT	WMT	IWSLT
All	89.95	92.94	93.35	92.99
$(\alpha, \epsilon, \gamma)$	92.06	94.25	94.55	93.96

Table 5: Cosine similarity [%] between (row,column) data set. With zero-knowledge of four target domains, our method selects subsets of training data that are more close to the target domain.

selected PTB data into sentence embeddings and compute pairwise cosine similarity between the centroid of each domain. The result is shown in Table 5. With zero-knowledge of four target domains, our method selects subsets of training data that are more close to the target domain. Furthermore, we find the larger domain distance between the source domain and the target domain, the larger improvement will be, by computing the pearson correlation coefficient (0.83) between the domain distance and the relative improvement normalized by test set size.

4 Related Work

In previous years, a crucial direction of work on robustness of NLP models lies on the vulnerability of NLP models to input perturbations, such as crafted noises (Song et al., 2020; Boucher et al., 2022; Li et al., 2020; Schwinn et al., 2021). For instance, Cheng et al. (2018) proposes an adversarial stability training objective to enable neural machine translation models robust to input perturbations. Niu et al. (2020) evaluates robustness to input perturbations for neural machine translation. Compared to them, our approach handles new test inputs with distribution shifts from any unknown domains.

Some literature propose evaluation metrics for robustness from the perspectives of statistics or input perturbations (Weng et al., 2018; Niu et al., 2020; Mangal et al., 2019; Couellan, 2021). However, they either focus on the worst-case scenario of adversarial inputs or sampling single instances without considering the full distribution of the system performance.

5 Conclusion

We introduce probabilistic robustness rewarded proximal policy data optimization (PRoDO) framework to improve NLP model's generalization by selecting training data. Our framework is rewarded by the $(\alpha, \epsilon, \gamma)$-Robustness to measure an NLP model's performance consistency over multiple domains. Our experiments show the effectiveness of probabilistic robustness measure to enhance learning generalization and prediction accuracy. Our work also demonstrates a successful step towards general robustness evaluation and data selection without target domain insight.

6 Limitations

Time efficiency is one limitation of this work. For one thing, like other data selection work with reinforcement learning (RL), introducing RL requires convergence of the policy network and the value network, which takes quite a long time empirically. Referring to our time comparison results, our methods are roughly ten times slower than training with all source data directly. For another, our robustness measure requires the tuning process for hyperparameters α and ϵ, which also takes additional time.

Acknowledgments

We appreciate Amazon Alexa Prize, National Science Foundation (NSF) Award No. 1747728, and NSF CRAFT Award No. 22001 to fund this research.

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. *arXiv preprint arXiv:2004.02105*.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79(1):151–175.

Aharon Ben-Tal and Arkadi Nemirovski. 1998. Robust convex optimization. *Mathematics of operations research*, 23(4):769–805.

Dimitris Bertsimas and Melvyn Sim. 2004. The price of robustness. *Operations research*, 52(1):35–53.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004. IEEE.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. In *Conference of european association for machine translation*, pages 261–268.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. *arXiv preprint arXiv:1805.06130*.

Nicolas Couellan. 2021. Probabilistic robustness estimates for feed-forward neural networks. *Neural Networks*, 142:138–147.

Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. 2017. Learning what data to learn. *arXiv preprint arXiv:1702.08635*.

Benoît Frénay and Michel Verleysen. 2013. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.

Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*.

Xiaomian Kang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2020. Dynamic context selection for document-level neural machine translation via reinforcement learning. *arXiv preprint arXiv:2010.04314*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 131–141.

Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2020. Contextualized perturbation for textual adversarial attack. *arXiv preprint arXiv:2009.07502*.

Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. 2019. Reinforced training data selection for domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1957–1968, Florence, Italy. Association for Computational Linguistics.

Xiaofei Ma, Peng Xu, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2019. Domain adaptation with BERT-based domain classification and data selection. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 76–83, Hong Kong, China. Association for Computational Linguistics.

Ravi Mangal, Aditya V Nori, and Alessandro Orso. 2019. Robustness of neural networks: A probabilistic and practical approach. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 93–96. IEEE.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Paul Michel and Graham Neubig. 2018. MTNT: A testbed for machine translation of noisy text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 543–553, Brussels, Belgium. Association for Computational Linguistics.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Sayan Mukherjee, Partha Niyogi, Tomaso Poggio, and Ryan Rifkin. 2006. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25(1):161–193.

Mathias Müller, Annette Rios, and Rico Sennrich. 2019. Domain robustness in neural machine translation. *arXiv preprint arXiv:1911.03109*.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. *arXiv preprint arXiv:2005.00580*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W Bruce Croft. 2019. Learning to selectively transfer: Reinforced transfer learning for deep text matching. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 699–707.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Leo Schwinn, René Raab, An Nguyen, Dario Zanca, and Bjoern Eskofier. 2021. Exploring misclassifications of robust neural networks to enhance adversarial attacks. *arXiv preprint arXiv:2105.10304*.

Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. 2020. Universal adversarial attacks with natural triggers for text classification. *arXiv preprint arXiv:2005.00174*.

Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. 2020. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599.

Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*, pages 303–314.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pages 2214–2218.

Victor Uc-Cetina, Nicolas Navarro-Guerrero, Anabel Martin-Gonzalez, Cornelius Weber, and Stefan Wermter. 2021. Survey on reinforcement learning for language processing. *arXiv preprint arXiv:2104.05565*.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2021. Measure and improve robustness in nlp models: A survey. *arXiv preprint arXiv:2112.08313*.

Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. 2018. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*.

Lana Yeganova, Dina Wiemann, Mariana Neves, Federica Vezzani, Amy Siu, Iñigo Unanue, Maite Oronoz, Nancy Mah, Aurélie Névéol, David Martinez, et al. 2021. Findings of the wmt 2021 biomedical translation shared task: Summaries of animal experiments as new test set. In *Sixth Conference on Machine Translation*.

Yu Yu, Shahram Khadivi, and Jia Xu. 2022a. Can data diversity enhance learning generalization? In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4933–4945, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yu Yu, Abdul Rafae Khan, and Jia Xu. 2022b. Measuring robustness for NLP. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3908–3916, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yunxiang Zhang, Liangming Pan, Samson Tan, and Min-Yen Kan. 2022. Interpreting the robustness of neural nlp models to textual perturbations. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3993–4007.

Unsupervised Improvement of Factual Knowledge in Language Models

Nafis Sadeq*, **Byungkyu Kang**†, **Prarit Lamba**†, **Julian McAuley***
Intuit† and UC San Diego*
`{nsadeq,jmcauley}@ucsd.edu`
`{Jay_Kang,Prarit_Lamba}@intuit.com`

Abstract

Masked language modeling (MLM) plays a key role in pretraining large language models. But the MLM objective is often dominated by high-frequency words that are sub-optimal for learning factual knowledge. In this work, we propose an approach for influencing MLM pretraining in a way that can improve language model performance on a variety of knowledge-intensive tasks. We force the language model to prioritize informative words in a fully unsupervised way. Experiments demonstrate that the proposed approach can significantly improve the performance of pretrained language models on tasks such as factual recall, question answering, sentiment analysis, and natural language inference in a closed-book setting.

1 Introduction

Pretrained language models (PLMs) such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), BART (Lewis et al., 2020), T5 (Raffel et al., 2020) use a Masked Language Modeling (MLM) objective during pretraining. However, a traditional MLM objective may not be optimal for knowledge-intensive tasks (Peters et al., 2019). It has been shown that language models can benefit from incorporating knowledge within the training objective in the form of entity embeddings (Peters et al., 2019; Zhang et al., 2019), knowledge retriever (Guu et al., 2020), knowledge embedding (Wang et al., 2021; Sun et al., 2020) or augmented pretraining corpora created from Knowledge Graphs (Agarwal et al., 2021). Despite their effectiveness, these approaches rely on existing knowledge bases and entity embeddings to incorporate knowledge within the training objective. These resources are expensive to construct and may not be available for all languages and domains (Huang et al., 2022).

In this work, we propose a pretraining approach that can achieve better performance on knowledge-intensive tasks without using any existing knowledge base. We combine two key strategies to influence MLM objective. Firstly, the tokens with higher informative relevance should be masked more frequently (Sadeq et al., 2022). Secondly, mistakes on informative tokens should be penalized more severely. The informative relevance of the tokens can be computed efficiently with a one-pass computation on the pretraining corpora. Experiments demonstrate that the proposed training strategy can help the language model achieve better performance on the factual knowledge recall benchmark LAMA (Petroni et al., 2019), extractive question answering (QA) benchmark SQuAD (Rajpurkar et al., 2016, 2018), prompt based sentiment analysis and natural language inference (NLI) tasks in AutoPrompt (Shin et al., 2020).

The key contribution of this work is proposing a completely unsupervised stand-alone MLM pretraining objective for language models that can significantly improve performance on knowledge-intensive tasks. Unlike prior works in the area, our method does not require existing knowledge bases to incorporate knowledge during pretraining. We make the code publicly available. [1]

2 Related Work

PLMs as knowledge bases It has been shown that large-scale PLMs such as BERT can be used as a knowledge base (Petroni et al., 2019, 2020). Prior works have focused on factual knowledge with regards to generative PLMs (Liu et al., 2021), multilingual setting (Jiang et al., 2020a), entities and query types (Heinzerling and Inui, 2021), fact checking (Lee et al., 2020).

Designing better prompts Jiang et al. (2020b) propose mining-based and paraphrasing-based methods for automatically generating prompts for improved factual recall performance. A similar

---

[1] The code is available at `https://github.com/intuit/wMLM.git`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2960–2969
May 2-6, 2023 ©2023 Association for Computational Linguistics

approach is explored by Zhong et al. (2021); Haviv et al. (2021); Qin and Eisner (2021). Shin et al. (2020) propose an approach for automatically creating MLM prompts for a diverse range of tasks such as sentiment analysis, natural language inference, relation extraction, etc.

Knowledge integration during pretraining Peters et al. (2019) use entity embeddings from existing knowledge bases and incorporate an entity linking loss jointly with an MLM loss to improve the factual recall performance of BERT. Similarly, Zhang et al. (2019); Wang et al. (2021); Févry et al. (2020); Sun et al. (2020); Liu et al. (2020) use entity representations or knowledge representation from existing knowledge bases to incorporate knowledge into the PLM. Guu et al. (2020) jointly pretrain a knowledge retriever along with a language modeling objective for knowledge integration. Agarwal et al. (2021) synthesize a text corpus from existing knowledge bases and use that during pretraining. Sun et al. (2019) use entity-level and phrase-level knowledge masking during training.

Knowledge modification after pretraining De Cao et al. (2021); Zhu et al. (2020) use constraint optimization for editing existing world knowledge within PLMs with minimal impact on the rest of the factual knowledge. Similarly, Verga et al. (2021) develop a fact injection language model architecture that allows easy integration of existing knowledge bases into PLMs without additional pretraining.

3 Methodology

We use MLM objective for pretraining, similar to prior works (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020; Raffel et al., 2020). Given a sequence of tokens Z, a subset of tokens $X \subset Z$ is randomly sampled for replacement ($|X|/|Z| \approx 0.15$ in Devlin et al. (2019)). For the replacement candidates in X, 80% of the time the replacement is done with a special token `[MASK]`, 10% of tokens are replaced with a random token, and the other 10% of candidates are left unchanged (Devlin et al., 2019; Liu et al., 2019; Joshi et al., 2020). The task of the model during pretraining is to predict the original tokens from the modified input sequence. For a set of replaced tokens $X(x_1, x_2, ..., x_N)$ and their corresponding output tokens $Y(y_1, y_2, ..., y_N)$, the loss $\mathcal{L}_{MLM}$ is computed as follows:

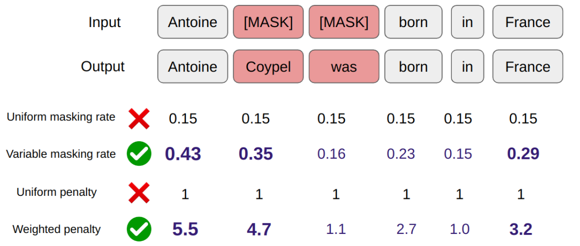


Figure 1: Simplified illustration of variable masking rate and weighted penalty

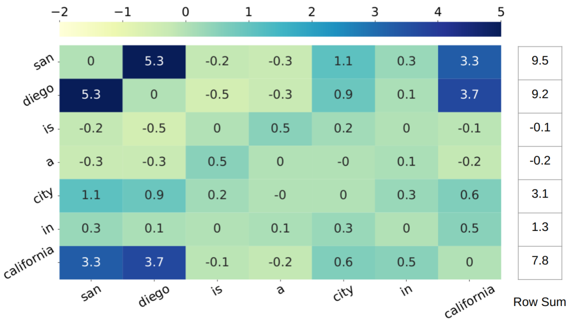
$$\mathcal{L}_{MLM} = -\sum_{i=1}^{N} \log \frac{e^{x_{i,y_i}}}{\sum_{v \in V} e^{x_{i,v}}} \tag{1}$$

Here, $x_{i,j}$ is the logit produced for output candidate j given input x_i and V is the vocabulary set. In traditional MLM loss computation, a uniform penalty is applied for all tokens within the vocabulary. In our work, we try to influence the MLM objective during pretraining to incorporate more factual knowledge. We differ from traditional MLM pretraining in two ways: **(a)** Instead of masking all tokens with equal probability, we allow some tokens to be masked more frequently if they have higher informative relevance, **(b)** We use weighted cross entropy loss to penalize mistakes on some tokens more severely if they have higher informative relevance. Simple illustrations of these two concepts are shown in Figure 1. We compute the loss as follows:

$$\mathcal{L}_{MLM} = -\sum_{i=1}^{N} w_{y_i} \log \frac{e^{x_{i,y_i}}}{\sum_{v \in V} e^{x_{i,v}}} \tag{2}$$

w_{y_i} is a penalty weight specific to a particular output token y_i. The magnitude of the weight is chosen based on the informative relevance of the tokens. A demonstration of this weighting is shown in Figure 1. Each token in the language model vocabulary has a unique masking rate and penalty weight associated with it. These values can be computed with a one-pass computation before pretraining.

In this context, the informative relevance of tokens represents how important a particular token is with regard to the factual knowledge. Tokens that are more important for factual knowledge (e.g. named entities) are expected to have a higher informative relevance. We use Pointwise Mutual Information (PMI (Fano, 1961)) to compute informative relevance in an unsupervised manner. We hypothesize that words that have high PMI with their



	san	diego	is	a	city	in	california	Row Sum
san	0	5.3	-0.2	-0.3	1.1	0.3	3.3	9.5
diego	5.3	0	-0.5	-0.3	0.9	0.1	3.7	9.2
is	-0.2	-0.5	0	0.5	0.2	0	-0.1	-0.1
a	-0.3	-0.3	0.5	0	-0	0.1	-0.2	-0.2
city	1.1	0.9	0.2	-0	0	0.3	0.6	3.1
in	0.3	0.1	0	0.1	0.3	0	0.5	1.3
california	3.3	3.7	-0.1	-0.2	0.6	0.5	0	7.8

Figure 2: Informative relevance of tokens in a particular document, by computing row-wise summation of the PMI matrix of all token pairs

neighboring words tend to have higher informative relevance. Firstly, we compute word co-occurrence statistics for the pretraining corpus within a skip-gram window. Secondly, PMI between all word pairs within the vocabulary is computed. Thirdly, we consider the pairwise PMI between all words within a particular document in the form of a matrix (as shown in Figure 2), so that the row-wise sum in that matrix reflects the token-specific informative relevance within that document. Then informative relevance for a token is averaged across the corpus. Finally, the computed values are normalized and converted to token-specific masking rates and token-specific penalty weights. Those masking rates are used to create masked inputs and the penalty weights are then incorporated during MLM loss computation, as shown in Equation 2.

4 Experiments

4.1 Pretraining Setup

We use the Wikipedia corpus available in Hugging Face (Lhoest et al., 2021) for pretraining, using a wordpiece tokenizer with a vocabulary size of 100k. The vocabulary size is chosen to ensure the inclusion of most entities. Word co-occurrence statistics are computed using a skip-gram window size of 10. The size of the matrix that holds the PMI between words is $100k \times 100k$. The one-pass computation involving informative relevance of tokens takes around two hours and requires 11 GB of memory. The masking rate for individual tokens varies between 15%-50%, depending on their informative relevance. The average masking rate for all tokens is 19%. The penalty weights for tokens are normalized within the range $[1, 5]$. Training is done with Hugging Face Transformers (Wolf et al., 2020) on an AWS p3.8xlarge machine with 4 Nvidia V100 GPUs. Our model architecture is similar to BERT-base (Devlin et al., 2019) with 12 layers and a hidden dimension of 768. The overall batch size is 128 with a learning rate of 5e-5 and an AdamW optimizer (Loshchilov and Hutter, 2019). Training is done for 10 epochs with a maximum document length of 128. Unlike BERT (Devlin et al., 2019), we do not use the next sentence prediction objective during pretraining. Additionally, the increased masking rate and penalty weight only apply to whole-word tokens. For the subword tokens, we use the minimum masking rate of 15% and penalty weight of 1.

4.2 Evaluation Benchmarks

We use LAMA knowledge probes (Petroni et al., 2019) for evaluating the factual recall performance of the model. LAMA has around 70k samples across 46 factual relations. To evaluate the performance on extractive QA, we use SQuAD v1 and v2 (Rajpurkar et al., 2016, 2018). For zero-shot performance evaluation on closed-book QA, we use the SQuAD portion from LAMA (Petroni et al., 2019). For closed-book sentiment analysis and NLI, we use SST2 and NLI probes from AutoPrompt (Shin et al., 2020). We also report the performance of the models on GLUE (Wang et al., 2018).

4.3 Baselines

We train four models using the same corpus, tokenizer and hyper-parameter setting mentioned in Section 4.1: **(a)** BERT_{uu}: Similar to Devlin et al. (2019), it uses a uniform masking rate and uniform penalty across tokens. This is our baseline. **(b)** BERT_{uw}: uses a uniform masking rate and weighted penalty. **(c)** BERT_{vu} (Sadeq et al., 2022): uses a variable masking rate across tokens and uniform penalty. **(d)** BERT_{vw}: This is our proposed approach that combines both a variable masking rate and weighted penalty across different tokens.

4.4 Results and Discussion

Factual Recall and Zero-shot QA The model using the proposed pretraining approach (BERT_{vw}) significantly outperforms the baseline (BERT_{uu}) on factual recall tasks in LAMA (shown in Table 1). The relative improvement of Mean Reciprocal Rank (MRR) over the baseline is 17.5%, 6%, and 8.1% for ConceptNet, GoogleRE, and TREx respectively. The SQuAD portion of the LAMA benchmark is a set of zero-shot QA samples adapted in a closed-book template. In this task,

Model	LAMA (Petroni et al., 2019)				AutoPrompt (Shin et al., 2020)		
	ConceptNet	GoogleRE	SQuAD	TREx	SST2	NLI (3 way)	NLI (2 way)
BERT_{uu}	0.114	0.281	0.156	0.578	0.651	0.397	0.620
BERT_{uw}	0.120	0.289	0.169	0.592	0.655	0.439	0.676
BERT_{vu}	0.129	0.292	0.175	0.616	0.700	0.457	0.697
BERT_{vw}	**0.134**	**0.298**	**0.187**	**0.625**	**0.704**	**0.481**	**0.711**

Table 1: Factual Recall performance on LAMA, Sentiment Analysis and Natural Language Inference on AutoPrompt. The metrics used for LAMA and AutoPrompt are Mean Reciprocal Rank (MRR) and Accuracy respectively.

Model	SQuAD		GLUE (Wang et al., 2018)								
	v1 (2016)	v2 (2018)	CoLA	SST2	MNLI	QNLI	QQP	STSB	RTE	WNLI	MRPC
BERT_{uu}	69.96	83.22	**31.06**	88.30	79.42	87.72	89.77	85.41	**66.43**	42.25	87.78
BERT_{uw}	71.17	84.17	28.55	89.11	79.82	87.15	89.59	85.70	58.84	49.30	87.93
BERT_{vu}	71.17	85.07	29.11	89.79	80.02	88.21	**90.10**	85.60	61.37	54.93	88.29
BERT_{vw}	**72.61**	**85.28**	28.93	**89.91**	**80.25**	**88.49**	89.82	**85.82**	59.93	**56.34**	**88.32**

Table 2: Performance on SQuAD and GLUE development set. For SQuAD, we report the F1 score. We report the Matthews correlation for CoLA, Pearson correlation for STSB, and accuracy for other GLUE tasks. The fine-tuning parameters for SQuAD and GLUE can be found in Appendix B.

we achieve 19.9% relative improvement over the baseline.

Case studies on factual recall are shown in Table 3. There are two key observations in these case studies. Firstly, the proposed model (BERT_{vw}) is more likely to rank the ground truth label higher during knowledge probes. This helps the model achieve better overall MRR. Secondly, the proposed model is more likely to produce specific words given a particular context when the baseline is only producing generic words. For example, when we use the prompt 'During Super Bowl 50 the `[MASK]` gaming company debuted their ad for the first time', the top three candidates from the baseline model are comparatively common words such as 'computer', 'electronic', and 'American'. But the proposed model is able to produce more specific words associated with three gaming companies ('Nintendo, 'Walt', and 'Atari'), including the correct answer 'Nintendo'. Similar observation can be made with the probe 'The organization that runs the satellite that measured dust that landed on the Amazon is `[MASK]`', where the proposed model makes specific predictions with the given context, such as 'NASA', 'Brazil' and 'Amazon'. But the baseline can only produce generic words like 'unknown', 'the', and 'unclear'.

Closed-book Sentiment Analysis and NLI We use AutoPrompt (Shin et al., 2020) to evaluate the closed-book sentiment analysis and NLI performance of the system. AutoPrompt provides a way to convert certain NLP tasks into a template-based probing format. The advantage of this type of prompting is that it allows us to exploit the factual knowledge within language models without the limitations of fine-tuning (Wallat et al., 2020). The prompt contains the input, a placeholder for the answers, and a span of trigger words (prompt templates shown in Appendix C). The trigger words are tuned using the training dataset and then subsequently used during evaluation. The proposed system achieves 8.1%, 21.1%, and 14.7% relative improvement in accuracy over the baseline in sentiment analysis, 3-way NLI, and 2-way NLI respectively (Table 1).

Fine-tuning vs Prompt-tuning Our proposed model achieves better performance compared to the baseline when fine-tuned on the extractive QA benchmark SQuAD (Rajpurkar et al., 2016, 2018) and text classification benchmark GLUE (Wang et al., 2018). It outperforms the baseline on both SQuAD v1 and v2 tasks and seven out of nine GLUE tasks (shown in Table 2). However, the relative performance improvement with fine-tuning is not as significant as factual recall, zero-shot QA, or prompt-tuning scenarios. The reason behind this may be explained by the findings of Wallat et al. (2020). The main strength of our approach is the ability to store more factual knowledge during pretraining. However, Wallat et al. (2020) have

Input	Ground Truth	BERT$_{\text{uu}}$ (Devlin et al., 2019)		BERT$_{\text{vw}}$ (proposed)	
		Prediction	Score	Prediction	Score
To emphasize the 50th anniversary of the Super Bowl the `[MASK]` color was used.	gold	yellow	0.17	**gold**	0.09
		red	0.13	rainbow	0.06
		green	0.12	orange	0.06
During Super Bowl 50 the `[MASK]` gaming company debuted their ad for the first time.	nintendo	computer	0.06	**nintendo**	0.05
		electronic	0.05	walt	0.04
		american	0.03	atari	0.04
A teacher is most likely teaching at a `[MASK]`.	school	university	0.61	**school**	0.40
		school	0.26	university	0.34
		college	0.03	seminary	0.09
Photosynthesis releases `[MASK]` into the Earth's atmosphere.	oxygen	sunlight	0.13	**oxygen**	0.21
		photosynthesis	0.09	carbon	0.12
		light	0.09	sunlight	0.06
The organization that runs the satellite that measured dust that landed on the Amazon is `[MASK]` .	nasa	unknown	0.11	**nasa**	0.06
		the	0.03	brazil	0.05
		unclear	0.03	amazon	0.02
Income inequality began to increase in the US in the `[MASK]`.	1970s	1960s	0.21	**1970s**	0.14
		1980s	0.18	1960s	0.13
		1970s	0.17	1980s	0.12
He moved to `[MASK]` at age 16 to complete his high school studies and obtained his Japanese citizenship in 1995.	japan	tokyo	0.42	**japan**	0.19
		japan	0.21	tokyo	0.18
		yokohama	0.03	hawaii	0.06
The Crimes Act 1914 is a piece of Federal legislation in `[MASK]`.	australia	canada	0.39	**australia**	0.12
		australia	0.07	tennessee	0.09
		england	0.03	canada	0.09
She is also member of the Helsinki City Council and the chairperson of the local party organisation in `[MASK]`.	helsinki	finland	0.52	**helsinki**	0.76
		helsinki	0.38	finland	0.18
		espoo	0.01	espoo	0.03
Mark Schwahn (born July 5, 1966) is an American `[MASK]`, director and producer.	screenwriter	actor	0.66	**screenwriter**	0.53
		screenwriter	0.14	writer	0.21
		writer	0.13	actor	0.16

Table 3: Case Study from factual recall samples from LAMA (Petroni et al., 2019)

shown that the factual knowledge learned during pretraining may be lost during fine-tuning, limiting the advantage of our proposed system. On the other hand, relational probing, zero-shot QA, and prompt-tuning-based NLP tasks can exploit the additional knowledge of our model more effectively, leading to much better performance.

Ablation Study We investigate how much performance improvement is due to the variable masking rate as opposed to the weighted penalty during MLM pretraining. This can be found by comparing BERT$_{\text{uw}}$ with BERT$_{\text{vu}}$ (Table 1 and 2). In most cases, we find that a variable masking rate performs slightly better than a weighted penalty.

5 Conclusion

In this work, we propose a pretraining strategy that can be effective in storing factual knowledge within language models. The additional knowledge helps the model outperform previous approaches on a variety of knowledge-intensive NLP tasks, such as factual recall, zero-shot QA, closed-book sentiment analysis, and natural language inference. Our model also achieves better performance when fine-tuned on SQuAD and GLUE tasks. In the future, we aim to extend our work for text-to-text pretrained models such as T5 (Raffel et al., 2020).

Acknowledgements

This work was performed during the first author's internship at Intuit. We thank anonymous reviewers for providing their valuable feedback on this work.

Limitations

One limitation of the proposed system is that it under-performs compared to the baseline in some fine-tuning tasks, such as CoLA (Table 2). The proposed training objective reduces the importance of stopwords in the pretraining objective. This may have a negative impact on performance in tasks where the syntax is important. More investigation is needed to understand and mitigate this issue.

Ethics Statement

A potential concern for the proposed system is that this training strategy may amplify the existing toxic behavior or bias of the language model if the related keywords get prioritized in the training objective. Reducing the toxic or biased behaviors of the proposed model can be an interesting research direction for future work.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Robert M Fano. 1961. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29(11):793–794.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4937–4951, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*, pages 3929–3938. PMLR.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623, Online. Association for Computational Linguistics.

Benjamin Heinzerling and Kentaro Inui. 2021. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1772–1791, Online. Association for Computational Linguistics.

Zijie Huang, Zheng Li, Haoming Jiang, Tianyu Cao, Hanqing Lu, Bing Yin, Karthik Subbian, Yizhou Sun, and Wei Wang. 2022. Multilingual knowledge graph completion with self-supervised adaptive graph alignment. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 474–485, Dublin, Ireland. Association for Computational Linguistics.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020a. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5943–5959, Online. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Nayeon Lee, Belinda Z. Li, Sinong Wang, Wen-tau Yih, Hao Ma, and Madian Khabsa. 2020. Language models as fact checkers? In *Proceedings of the Third Workshop on Fact Extraction and VERification (FEVER)*, pages 36–41, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2901–2908. AAAI Press.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *CoRR*, abs/2103.10385.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 43–54. Association for Computational Linguistics.

Fabio Petroni, Patrick S. H. Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5203–5212. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nafis Sadeq, Canwen Xu, and Julian McAuley. 2022. InforMask: Unsupervised informative masking for language model pretraining. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5866–5878, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. Colake: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 3660–3670. International Committee on Computational Linguistics.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223.

Pat Verga, Haitian Sun, Livio Baldini Soares, and William Cohen. 2021. Adaptable and interpretable neural MemoryOver symbolic knowledge. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3678–3691, Online. Association for Computational Linguistics.

Jonas Wallat, Jaspreet Singh, and Avishek Anand. 2020. BERTnesia: Investigating the capture and forgetting of knowledge in BERT. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 174–183, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: enhanced language representation with informative entities. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1441–1451. Association for Computational Linguistics.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix X. Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *CoRR*, abs/2012.00363.

A Performance on LAMA by Relation

Domain	Dataset	BERT$_{\text{uu}}$	BERT$_{\text{uw}}$	BERT$_{\text{vu}}$	BERT$_{\text{vw}}$
ConceptNet	test	0.114	0.120	0.129	**0.134**
GoogleRE	dateOfBirth	0.099	0.109	0.111	**0.113**
GoogleRE	placeOfBirth	0.456	0.459	0.461	**0.465**
GoogleRE	placeOfDeath	0.288	0.300	0.305	**0.315**
Squad	test	0.156	0.169	0.175	**0.187**
TREx	P1001	0.779	0.770	0.793	**0.798**
TREx	P101	0.442	0.468	0.501	**0.514**
TREx	P103	0.822	0.834	**0.838**	0.836
TREx	P106	0.642	0.653	**0.675**	0.664
TREx	P108	0.491	0.526	0.538	**0.556**
TREx	P127	0.586	0.615	0.620	**0.636**
TREx	P1303	0.380	0.427	0.433	**0.472**
TREx	P131	0.690	0.702	0.741	**0.750**
TREx	P136	0.595	0.629	0.651	**0.675**
TREx	P1376	0.747	0.761	0.783	**0.792**
TREx	P138	0.633	0.640	0.656	**0.680**
TREx	P140	0.569	0.574	**0.608**	0.602
TREx	P1412	0.764	0.773	**0.785**	0.781
TREx	P159	0.535	0.551	0.573	**0.576**
TREx	P17	0.870	0.863	0.884	**0.887**
TREx	P176	0.647	0.673	0.699	**0.720**
TREx	P178	0.569	0.592	0.631	**0.639**
TREx	P19	0.477	0.478	0.509	**0.519**
TREx	P190	0.279	0.276	0.296	**0.297**
TREx	P20	0.511	0.533	0.559	**0.565**
TREx	P264	0.247	0.280	0.291	**0.313**
TREx	P27	0.745	0.756	0.767	**0.773**
TREx	P276	0.625	0.623	0.652	**0.663**
TREx	P279	0.512	0.544	0.562	**0.580**
TREx	P30	0.802	0.813	0.835	**0.842**
TREx	P31	0.616	0.627	0.635	**0.635**
TREx	P36	0.569	0.578	**0.618**	0.615
TREx	P361	0.530	0.538	0.567	**0.574**
TREx	P364	0.703	0.715	0.729	**0.742**
TREx	P37	0.701	0.688	**0.728**	0.715
TREx	P39	0.572	0.607	0.613	**0.630**
TREx	P407	0.638	0.630	0.647	**0.666**
TREx	P413	0.422	0.453	0.483	**0.507**
TREx	P449	0.416	0.444	0.454	**0.495**
TREx	P463	0.646	0.674	0.697	**0.713**
TREx	P47	0.492	0.508	0.564	**0.565**
TREx	P495	0.685	0.662	**0.699**	0.681
TREx	P527	0.423	0.452	0.521	**0.527**
TREx	P530	0.379	0.373	0.400	**0.416**
TREx	P740	0.407	0.414	0.438	**0.438**
TREx	P937	0.528	0.541	0.569	**0.569**

Table 4: Relation by relation performance comparison on LAMA (Petroni et al., 2019)

B Hyper-parameter for fine-tuning on GLUE, SQuAD

Hyper-parameter	GLUE	SQuAD
Batch Size	32	12
Learning Rate	2e-5	3e-5
Epochs	3	2
Weight Decay	0.01	0.01

Table 5: Fine-tuning hyper-parameters for GLUE and SQuAD

C Hyper-parameter for AutoPrompt

Hyper-parameter	SST2	NLI
# Trigger Token	3	4
# Candidate	100	10
Batch Size	24	32
# Iterations	180	100

Table 6: Prompt-tuning hyper-parameters for AutoPrompt (Shin et al., 2020)

Task	Template	Prompt Example	Labels
SST2	{sentence} [T] . . . [T] [P]	director rob marshall went out gunning to make a great one movie director cinema `[MASK]`	**pos:** partnership, good **neg:** worse, bad
NLI	{prem}[P] [T] . . . [T] {hyp}	There is no man in a black jacket doing tricks on a motorbike `[MASK]` strange workplace A person in a black jacket is doing tricks on a motorbike	**con:** Nobody, nobody, nor **ent:** found, ways, Agency **neu:** ##ponents, ##lary, ##uated

Table 7: Prompt template for Sentiment Analysis and Natural Language Inference tasks in AutoPrompt (Shin et al., 2020)

Learning to Ignore Adversarial Attacks

Yiming Zhang[†] **Yangqiaoyu Zhou**[†] **Samuel Carton**[‡] **Chenhao Tan**[†]
[†] University of Chicago [‡] University of New Hampshire
`{yimingz0,zhouy1,chenhao}@uchicago.edu` `samuel.carton@unh.edu`

Abstract

Despite the strong performance of current NLP models, they can be brittle against adversarial attacks. To enable effective learning against adversarial inputs, we introduce the use of rationale models that can explicitly learn to ignore attack tokens. We find that the rationale models can successfully ignore over 90% of attack tokens. This approach leads to consistent and sizable improvements (∼10%) over baseline models in robustness on three datasets for both BERT and RoBERTa, and also reliably outperforms data augmentation with adversarial examples alone. In many cases, we find that our method is able to close the gap between model performance on a clean test set and an attacked test set and hence reduce the effect of adversarial attacks.

1 Introduction

Adversarial robustness is an important issue in NLP, asking how to proof models against confounding tokens designed to maliciously manipulate model outputs. As such models become more powerful and ubiquitous, research continues to discover surprising vulnerabilities (Wallace et al., 2019), demanding improved robustness methods.

A common defense method to combat adversarial attacks is adversarial training. Given knowledge of attack strategies, it constructs synthetic adversarial examples to augment clean examples during training (Zhang et al., 2020). Intuitively, the model will *implicitly* learn to ignore attacking tokens and become robust to that type of attack. In practice, however, this goal can be challenging through data augmentation alone.

In this study, we propose a simple yet effective adversarial training schema for additive attacks: *explicitly* training the model to ignore adversarial tokens. We do this by augmenting the underlying model with a rationale extractor (Lei et al., 2016) to serve as an input filter, and then training this extractor to ignore attacking tokens as an additional joint objective to overall label accuracy (Fig. 1).

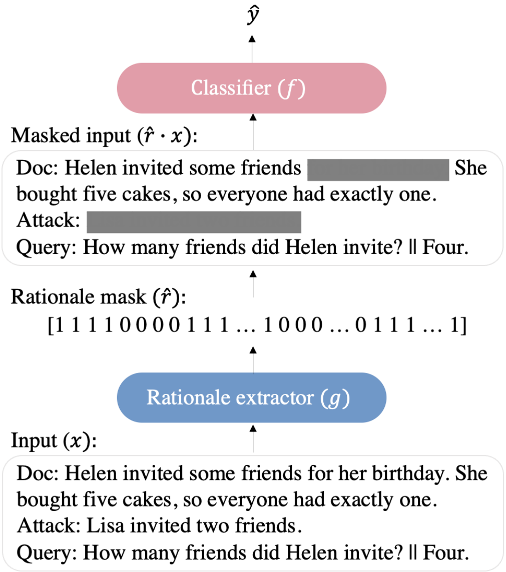


Figure 1: Example illustration of an ideal rationale model that is robust to added attack tokens. The rationale extractor filters out the confounding sentence (*"Lisa invited two friends"*) added by the adversary, and extracts the supporting spans (*"Helen invited some friends"* and *"She bought five cakes, so everyone had exactly one"*) to help the model deduce the correct answer (*"Four"*).

In addition to training the extractor to distinguish the attacking/non-attacking token dichotomy, we also explore the utility of human-provided explanations in this regard. In doing so, we ask: does learning from human rationales help the model avoid attending to attacking tokens?

Fine-tuning BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) on multiple datasets, we demonstrate that the additive attack proposed by Jia and Liang (2017) does reduce model accuracy, and that data augmentation with adversarial examples provides limited benefit in defending these models from this attack in most cases.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2970–2984
May 2-6, 2023 ©2023 Association for Computational Linguistics

Our main results are that rationale-style models learn to ignore these attacks more effectively than only with data augmentation, leading to an improvement of ∼10% in accuracy on attacked examples compared to baseline models and an advantage of 2.4% over data augmentation alone, mostly recovering clean test performance. While human explanations may potentially improve the interpretability of these models, they are of limited use in improving this defense even further.

In summary, we offer three main contributions:

- We show that explicitly training an extractive rationale layer to ignore attack tokens is more effective than implicitly training a model via data augmentation with adversarial examples.
- We assess whether human-annotated rationales augment this defense, showing that they have only a limited benefit.
- We conduct an in-depth error analysis of differences between models, explaining some of the patterns we observe in our main results.

Our code is available at `https://github.com/ChicagoHAI/rationalization-robustness`.

2 Related Work

We build on prior work on adversarial robustness and learning from explanations.

Adversarial robustness. Adversarial attacks against NLP models seek to maliciously manipulate model output by perturbing model input. Zhang et al. (2020) present a survey of both attacks and defenses. Example attacks include character-level manipulations (Gao et al., 2018; Li et al., 2019), input removal (Li et al., 2017; Feng et al., 2018), synonym substitutions (Ren et al., 2019), and language model-based slot filling (Li et al., 2020; Garg and Ramakrishnan, 2020; Li et al., 2021). A distinction in attack types is whether the attack requires access to the model (Ebrahimi et al., 2018; Yoo and Qi, 2021; Wallace et al., 2019) or not (Alzantot et al., 2018; Jin et al., 2020). TextAttack (Morris et al., 2020) is a framework and collection of attack implementations. Our work focuses on the ADDSENT attack proposed by Jia and Liang (2017) in reading comprehension.

As interest in adversarial attacks has increased, so has interest in developing models robust to these attacks. A popular defense method is adversarial training via data augmentation, first proposed by Szegedy et al. (2014) and employed by Jia and Liang (2017) to bring their model *almost* back to clean test performance. A recent example in this vein is Zhou et al. (2020), which proposes Dirichlet Neighborhood Ensemble as a means for generating dynamic adversarial examples during training. Another popular approach is knowledge distillation (Papernot et al., 2016), which trains an intermediate model to smooth between the training data and the final model. Our work explores a new direction that explicitly learns to ignore attacks.

Learning from explanations. Recent work has sought to collect datasets of human-annotated explanations, often in the form of binary *rationales*, in addition to class labels (DeYoung et al., 2019; Wiegreffe and Marasović, 2021), and to use these explanations as additional training signals to improve model performance and robustness, sometimes also known as *feature-level feedback* (Hase and Bansal, 2021; Beckh et al., 2021).

An early work is Zaidan et al. (2007), which uses human rationales as constraints on an SVM. More recently, Ross et al. (2017) uses human rationales to penalize neural net input gradients showing benefits for out-of-domain generalization, while Erion et al. (2021) use a similar method based on "expected gradients" to produce improvements in in-domain test performance in certain cases. Katakkar et al. (2021) evaluate feature feedback for two attention-style models, finding, again, gains in out-of-domain performance, while Han and Tsvetkov (2021) use influence functions (Koh and Liang, 2017) to achieve a similar outcome. Where our study differs from most previous work is in using feature feedback for adversarial rather than out-of-domain robustness. A concurrent work by Chen et al. (2022) uses rationalization to improve robustness. The proposed method is similar to our work, but we explore supervision with attack tokens and achieve stronger robustness to additive attacks.

3 Adversarial Attacks and Datasets

In this paper, we focus on model robustness against the ADDSENT additive attack proposed by Jia and Liang (2017). The attack is designed for reading comprehension: consider each instance as a tuple of document, query, and label (d, q, y), where y indicates whether the query is supported by the document. The attack manipulates the content of the query to form an attack sentence (A) and adds A to the document to confuse the model. Specifically, ADDSENT proceeds as follows:

Query q:
FC Bayern Munich was founded in 2000.
Mutated Query $\hat{q}$:
DYNAMO Leverkusen Cologne was founded in 1998.
Modified Document d'
…has won 9 of the last 13 titles. DYNAMO Leverkusen Cologne was founded in 1998. They have traditional local rivalries with…

Figure 2: An example of the AddSent attack.

1. We modify the query q by converting all named entities and numbers to their nearest neighbor in the GloVe embedding space (Pennington et al., 2014). We flip all adjectives and nouns to their antonyms using WordNet (Miller, 1995) and yield a mutated query $\hat{q}$. If we fail to mutate the query due to not being able to find matching named entities or antonyms of adjectives and nouns, we skip the example.
2. We convert the mutated query $\hat{q}$ into an adversarial attack A using CoreNLP (Manning et al., 2014) constituency parsing, under a set of about 50 rules enumerated by Jia and Liang (2017). This step converts it into a factual statement that resembles but is not semantically related to the original query q.
3. The adversarial attack A is inserted at a random location within the original document and leads to a new tuple (d', q, y).[1]

The key idea behind the AddSent attack is that the mutations alter the semantics of the query by mutating the named entities and numbers, so that the attack contains words or phrases that are likely confusing to the model without changing the true semantics of the input. An example of the AddSent attack is given above.

The original approach includes an additional step of using crowdsourced workers to filter ungrammatical sentences. We do not have access to this manual validation process in all datasets. Occasionally, AddSent generates ungrammatical attacks but it nevertheless proves empirically effective in reducing the performance of our models.

Datasets. To evaluate our hypotheses on learning to ignore adversarial attacks, we train and evaluate models on the Multi-Sentence Reading Comprehension (MultiRC; Khashabi et al., 2018) and Fact Extraction and VERification (FEVER; Thorne et al., 2018) datasets. Both are reading comprehension datasets, compatible with the AddSent attack. For MultiRC, the query consists of a question and potential answer about the document, labeled as true or false, while for FEVER it is a factual claim about the document labeled as "supported" or "unsupported". Both datasets include *human rationales*, indicating which tokens are pertinent to assessing the query. Table 1 summarizes their basic statistics.

Dataset	Text length	Rationale length	Total size
MultiRC	336.0	52.0	32,088
FEVER	335.9	47.0	110,187
SQuAD	119.8	——	87,599

Table 1: Basic statistics of MultiRC, FEVER, and SQuAD.

In modeling these two datasets, we follow standard practice in appending the query to the end of the document with [SEP] tokens. We use train/validation/test splits prepared by the ERASER dataset collection (DeYoung et al., 2019). Because we are interested in relative differences between training regimes rather than absolute performance, we subsample the FEVER training set to 25% so that it is comparable to MultiRC for the sake of training efficiency.

Directly applying the synthetic AddSent attack to MultiRC and FEVER leads to occasionally ungrammatical adversarial examples due to incorrectly applied conversion heuristic or errors in constituency parsing. To alleviate this concern, we further evaluate on SQuAD (Rajpurkar et al., 2016), for which Jia and Liang provide an AddSent-attacked *evaluation* set that is re-written and approved by human workers. However, this dataset does not have human rationales. We again use the train/validation/test splits provided by Jia and Liang in our experiments.

4 Modeling

Our study assesses whether adding an explicit rationale extractor to a model and training it to ignore attack tokens results in a more effective defense than simply adding attacked examples to the training set. This comparison results in several combinations of model architectures and training regimes.

We denote each training instance as $(\boldsymbol{x}, \boldsymbol{r}, y)$: a text sequence $\boldsymbol{x}$ consisting of the concatenated document and query, a ground-truth binary rationale sequence $\boldsymbol{r}$, and a binary label y.

---

[1] We experimented with variants of inserting only at the beginning or the end. The results are qualitatively similar, so we only report random in this paper.

Human rationale & attack	ADV. + ATK. SUP.	ADV. + HUMAN SUP.
... and 18 national cups. FC Bayern was founded in 1900 by 11 football players, led by Franz John. Although Bayern won ... European Cup three times in a row (1974 – 1976). DYNAMO Leverkusen Cologne was founded in 1998. Overall , Bayern has reached ten European ...	... and 18 national cups. FC Bayern was founded in 1900 by 11 football players, led by Franz John. Although Bayern won ... European Cup three times in a row (1974 – 1976). DYNAMO Leverkusen Cologne was founded in 1998. Overall , Bayern has reached ten European ...	... and 18 national cups. FC Bayern was founded in 1900 by 11 football players, led by Franz John. Although Bayern won ... European Cup three times in a row (1974 – 1976). DYNAMO Leverkusen Cologne was founded in 1998. Overall , Bayern has reached ten European ...

Table 2: An example from FEVER illustrating different modes of adversarial training with rationale supervision. Human rationales are colored yellow, and attack tokens are colored red. Rationale models are supervised to extract the blue tokens, while ignoring the gray tokens.

Data augmentation?	Rationale?
No data augmentation	None
	Human (HUMAN SUP.)
Augmented with attack data (Adv.)	None
	Non-attack (ADV. + ATK. SUP.)
	Human (ADV. + HUMAN SUP.)

Table 3: Summaries of rationale model setups.

Baseline models and training. We use BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) as basis models. In the baseline training condition we fine-tune these models as normal, evaluating them on both the original test set and a version of the test set where each item has been corrupted with the ADDSENT attack described above. We denote this condition as "NO ADV."

In the baseline adversarial training via data augmentation condition (denoted ADV.), we add ADDSENT-attacked versions of each training example to the training set on a one-to-one basis, allowing the model to train for the presence of such attacks. This represents a fairly standard baseline defense in the literature (Zhang et al., 2020).

Following prior adversarial robustness literature (Jia et al., 2019), we also consider a stronger baseline by augmenting the training set with K perturbed examples for each training example. For our main experiments, we use $K = 10$. This setting (denoted ADV.-10X) measures whether the baseline method implicitly adapts to the ADDSENT attack when abundant signal is provided.

Rationale model. To lend the baseline model an extractor capable of filtering out confounding tokens, we use the rationale model proposed by Lei et al. (2016). It comprises a rationale extractor g and a label predictor f (Fig. 1). The rationale extractor generates a binary predicted rationale $\hat{\boldsymbol{r}}$, which is applied as a mask over the input to the predictor via masking function m, producing a predicted label:

$$\begin{aligned} g(\mathbf{x}) &\rightarrow \hat{\boldsymbol{r}} \\ f(m(\mathbf{x}, \hat{\boldsymbol{r}})) &\rightarrow \hat{y} \end{aligned} \tag{1}$$

The two components are trained together to optimize predicted label accuracy as well as loss associated with the predicted rationale. In an unsupervised scenario, this loss punishes the norm of the predicted rationale, encouraging sparsity on the (heuristic) assumption that a sparse rationale is more interpretable. In this study, we rather consider the supervised scenario, where we punish $\hat{\boldsymbol{r}}$'s error with respect to a ground-truth rationale $\boldsymbol{r}$. However, we find empirically that the rationale sparsity objective is useful in combination with the rationale supervision objective, leading to the following joint objective function using cross-entropy loss $\mathcal{L}_{CE}$ with hyperparameter weights λ_1 and λ_2:

$$\mathcal{L}_{CE}(\hat{y}, y) + \lambda_1 \mathcal{L}_{CE}(\hat{\boldsymbol{r}}, \boldsymbol{r}) + \lambda_2 ||\hat{\boldsymbol{r}}||. \tag{2}$$

Adversarial training with rationale supervision. To introduce rationale supervision, we augment the training set with attacked examples on a one-to-one basis with original examples, similar to adversarial training. Moreover, we can change the ground-truth rationale to reflect the desired behavior for the model. We consider two options for this new ground-truth $\boldsymbol{r}$: (1) a binary indicator of whether a token is adversarial or not (ADV. + ATK. SUP.), and (2) the human-annotated rationale (ADV. + HUMAN SUP.), which also filters adversarial tokens. Table 2 contains an example illustrating the distinction between ADV. + ATK. SUP. and ADV. + HUMAN SUP.

Table 3 summarizes all the combinations of setups that we use in our study. For each of these setups, we test one rationale model using independent BERT modules for g and f, and one using independent RoBERTa modules for both. We present additional implementation details in Appendix A.

Taken together, these conditions address our three research questions: (1) Is adversarial training via rationale supervision more effective than via attacked examples? (2) Does training the model to emulate human explanation make it intrinsically more robust to attacks? (3) Do human explanations improve upon adversarial training with non-attack tokens as rationale supervision?

5 Experimental Setup and Results

We start by describing our experimental setup and evaluation metrics. We then investigate model performance with different training regimes and conduct an in-depth error analysis.

5.1 Experimental Setup

Our study compares whether rationale-style models are better at learning to explicitly ignore adversarial tokens than standard models via adversarial training. As we describe above, we train three variants of the standard classification model (NO ADV., ADV., ADV.-10X), and three variants of the rationale model (ADV. + ATK. SUP., HUMAN SUP., ADV. + HUMAN SUP.).

Exploring these 6 architecture/training combinations for three datasets (MULTIRC, FEVER, and SQUAD) and two underlying models (BERT and RoBERTa), we report results from all trained models in Table 4. We report relevant metrics on both the clean test set and the attacked test set for each model. For MULTIRC and FEVER, the metric we use is accuracy. Since SQUAD is a span extraction task, we report the Span F1 score instead. Performance on the attacked test set is our key measure of robustness.

Additionally, for the rationale models, we report the mean percentage of attack and non-attack tokens included in each predicted rationale, two metrics that help explain our accuracy results. The mean percentage of attack tokens included in the predicted rationale indicates the effectiveness of ignoring attack tokens: the lower the better.

5.2 Main Results

We focus our analysis on three questions:

1. Does adversarial rationale supervision on augmented data improve robustness over adversarial data augmentation alone?
2. Does human rationale supervision improve adversarial robustness over a standard model?
3. Does the addition of human rationales to adversarial training further improve robustness?

Table 4 summarizes the main results of the paper, showing the accuracy of each combination of architecture, training regime, underlying model and dataset. Looking at the attacked versus clean test set performance for the standard model, we see that **the ADDSENT attack is effective**, reducing accuracy on MULTIRC (∼6%), FEVER (∼10%), and SQUAD (∼12-24%).

Adversarial rationale supervision (ADV. + ATK. SUP.). Rationale models provide an interface for explicitly supervising the model to ignore attack tokens. Our *key* question is whether they can be used to improve the effectiveness of adversarial training. We first discuss the effect of data augmentation and then show that rationale models are indeed more effective at ignoring attack tokens.

Data augmentation with adversarial examples works, mostly. In almost all cases, data augmentation does result in improved performance on the attacked test set, improving +5.9% (FEVER) and +17.6% (SQUAD) for BERT, as well as +6.4% (MULTIRC), +9.7% (FEVER), and +9.4% (SQUAD) for RoBERTa. The exception is BERT on MULTIRC, where it causes a decrease of -1.0%. However, in only one case out of six does data augmentation with adversarial examples bring the model back to clean test performance (RoBERTa on MULTIRC, +0.3%).

Surprisingly, BERT on MULTIRC is the only scenario where the ADV.-10X augmentation significantly improves attack accuracy (4.3% improvement over ADV.). In all the other cases, adding more adversarial examples does not improve robustness and even leads to a 3.5% drop in SQUAD for RoBERTa. This result demonstrates that BERT and RoBERTa may not learn from adversarial examples alone.

Adversarial rationale supervision improves on adversarial training baselines in all cases. We see an improvement of +4.6% for BERT on MULTIRC, +2.9% for BERT on FEVER, +2.7% for BERT on SQUAD, +2.2% for RoBERTa on MULTIRC, +0.7% for RoBERTa on FEVER, and +1.0% for RoBERTa on SQUAD (2.4% on average). For the one case where adversarial data augmentation recovered clean test performance (RoBERTa on MULTIRC), adversarial rationale supervision actu-

Model	Architecture	Training	MultiRC (Acc.)		FEVER (Acc.)		SQuAD (Span F1)	
			Clean	Attacked$_\text{S}$	Clean	Attacked$_\text{S}$	Clean	Attacked$_\text{H}$
BERT	Standard	No Adv.	68.6	62.6	88.2	78.9	86.4	62.8
		Adv.	67.3	61.6	**88.5**	84.8	86.0	80.4
		Adv.-10x	66.2	65.9	86.3	84.5	82.2	78.0
	Rationale	Adv. + Atk. Sup.	69.6	66.2	87.1	**87.7**	**86.5**	**83.1**
		Human Sup.	70.0	64.4	88.0	76.7	- - -	- - -
		Adv. + Human Sup.	**70.5**	**69.4**	87.5	87.5	- - -	- - -
RoBERTa	Standard	No Adv.	82.6	76.5	93.5	83.0	93.2	81.0
		Adv.	84.4	82.9	93.2	92.7	92.9	90.4
		Adv.-10x	83.5	82.1	93.5	93.2	89.9	86.9
	Rationale	Adv. + Atk. Sup.	**85.2**	**85.1**	93.4	**93.4**	**93.3**	**91.4**
		Human Sup.	84.0	74.9	**94.1**	85.7	- - -	- - -
		Adv. + Human Sup.	85.0	82.5	93.4	**93.4**	- - -	- - -

Table 4: Model performance on clean and attacked test sets for MultiRC, FEVER, and SQuAD. Attacked$_\text{S}$ are synthetic attacks produced by AddSent, and Attacked$_\text{H}$ are attacks generated by human workers. We vary the level of augmentation for the standard classification models (No Adv., Adv., Adv.-10x). For rationale models, we control for the presence of adversarial training data and the type of rationale supervision: Adv. + Atk. Sup. treats non-attack tokens as rationale, and Human Sup. does not use adversarial training. Rationale models outperform the baseline classifiers across all attacked datasets.

Model	Training	MultiRC		FEVER		SQuAD	
		Attack %	Non-A %	Attack %	Non-A %	Attack %	Non-A %
BERT	Adv. + Atk. Sup.	1.4	98.4	0.2	96.7	27.8	99.7
	Human Sup.	87.5	8.2	66.7	17.8	- - -	- - -
	Adv. + Human Sup.	9.5	14.4	0.5	24.4	- - -	- - -
RoBERTa	Adv. + Atk. Sup.	6.0	96.7	0.9	95.8	16.1	99.0
	Human Sup.	92.4	12.6	60.0	12.2	- - -	- - -
	Adv. + Human Sup.	32.1	15.6	0.1	23.0	- - -	- - -

Table 5: Percentage of attack and non-attack (Non-A) tokens *included* in the predicted rationales. Lower is better for attack tokens. Arguably, a lower percentage of non-attack tokens is also better as it improves interpretability.

ally improves clean test performance by +2.5%.

The effectiveness of Adv. + Atk. Sup. is even more salient if we compare with No Adv. on attacked test: 3.6%, 8.8%, and 20.3% for BERT on MultiRC, FEVER, and SQuAD, 8.6%, 10.4%, and 10.4% for RoBERTa on MultiRC, FEVER, and SQuAD (10.4% on average).

The above findings remain true even when we compare our methods against the theoretically stronger baseline of Adv.-10x, where the training dataset is augmented with 10 perturbed examples for every training example. Our models trained with adversarial rationale supervision outperforms Adv.-10x across all datasets and models, and our best model outperforms the Adv.-10x baseline by 3.3% on average. This result highlights both the efficiency and the effectiveness of our method: with adversarial rationale supervision, BERT and RoBERTa achieve greater defense against the AddSent attack using 10% of the adversarial examples.

Interestingly, the adversarially-supervised rationale model demonstrates a strong ability to generalize knowledge learned from synthetic attacks to tune out human-rewritten attacks (+20.3% on SQuAD; recall we do not have human-rewritten attacks during training), indicating the potential of our method in a real-world scenario.

Table 5 explains this success. The adversarially-supervised rationale model includes 6% or fewer attacking tokens on MultiRC and FEVER, indicating that it did largely succeed in learning to occlude these tokens for the predictor. Additionally, both BERT and RoBERTa rationale models are able to tune out most human-generated attack tokens, ignoring over 70% of attack tokens while keeping 99% of the original text for both models.

Effect of human rationale supervision alone (Human Sup.). We find mixed evidence for whether human rationale supervision alone improves adversarial robustness. For BERT on MULTIRC and RoBERTa on FEVER, human rationale outperforms the standard classification model, but the opposite occurs for the other two model/dataset combinations.

Table 5 contextualizes this mixed result: the rationale model supervised solely on human rationales includes 60.0% to 92.4% of attack tokens in its rationale (compared to between 8.2% and 17.8% of non-attack tokens), indicating that it is largely fooled by the ADDSENT attack into exposing the predictor to attack tokens.

This result may be explained by the fact that human rationales for these datasets identify the part of the document that pertains particularly to the query, while the ADDSENT attack crafts adversarial content with a semantic resemblance to that same query. Hence, it is understandable that human rationale training would not improve robustness.

Human and adversarial rationale supervision (ADV. + HUMAN SUP.). Although human rationales alone may not reliably improve model robustness, a final question is whether human rationales can serve as a useful addition to adversarial training. Does training the model to both ignore adversarial tokens and emulate human explanations further improve robustness against the ADDSENT attack?

In two out of four cases, the performance of ADV. + HUMAN SUP. is equal to that of ADV. + ATK. SUP. Only for BERT on MULTIRC does ADV. + HUMAN SUP. result in an improvement, being the only configuration that brings performance back to that of clean test for that model, and dataset. For RoBERTa on MULTIRC, it actually weakens attacked test performance.

While these results are mixed, Table 5 shows that the model does at least achieve this result at a much lower included percentage of non-attack tokens ($\sim$20% vs. >95%), a concession toward model interpretability.

Overall, our results suggest that human rationales have limited effect in defending against adversarial attacks, but can be important in developing sparse (and potentially interpretable) models.

5.3 Error Analysis

To better understand the behavior of the models, we examine mistakes from BERT compared to explicitly training a rationale extractor on MULTIRC. We start with a qualitative analysis of example errors, and then discuss general trends, especially on why human rationales only provide limited benefits over ADV. + ATK. SUP. More in-depth analyses can be found in the appendix for space reasons, including a Venn diagram of model mistakes.

Qualitative analysis. We look at example errors of ADV. to investigate attacks that are confusing even after adversarial augmentation. Table 6 shows example outputs of the rationale models based on either non-attack tokens or human rationales.

Example 1 shows a case where models with explicit rationale extractors ignore attacks more effectively than ADV. In the attack sentence, "*tete didn't stay in*" is highly similar to the query, so a model likely predicts True if it uses the attack information. In comparison, both rationale models ignore the attack in label prediction, which enables them to make correct predictions.

Example 2 demonstrates that ADV. + HUMAN SUP. makes mistakes when it fails to include crucial information in rationales while avoiding attack tokens. ADV. + HUMAN SUP. predicts the wrong label because it misses information for the number of friends in its rationale. ADV. + ATK. SUP. gets this example correct because it can both ignore the attack and include the necessary information.

Finally, Example 3 shows an example where ADV. + HUMAN SUP. is better than ADV. + ATK. SUP. when generating rationales to ignore noises. ADV. + HUMAN SUP. includes attacks in rationale, but it is still able to predict the label because the attack is not confusing given the selected rationale. The generated rationale helps ADV. + HUMAN SUP. to avoid unnecessary information that may confuse the model. For example, the sentence with "picts" could confuse the model to predict True. On the other hand, ADV. + ATK. SUP. gets this example wrong, despite occluding all attack tokens.

More generally, we find that ADV. + HUMAN SUP. *tends to have high false negative rates*. When ADV. + HUMAN SUP. fails to extract good rationales, it tends to predict False due to missing information from the rationale. In contrast, ADV. + ATK. SUP. rarely occludes necessary information, so it does not suffer from the same issue.

ADV. + ATK. SUP. *is better than* ADV. + HUMAN SUP. *when human rationales are denser and passage length is longer (see Table 9 in the appendix).* We observe that denser human rationales usually comprise evidence from different parts of the passage. Since ADV. + ATK. SUP. predicts almost all non-attack tokens as rationale, they have

Human rationale & attack	**Adv. + Atk. Sup.**	**Adv. + Human Sup.**
(A) Example 1, true label: False		
[CLS] ... in may 1904 , the couple ’ s first son , hans albert einstein , was born in bern , switzerland . their second son , eduard , was born in zurich in july 1910 . in 1914 , the couple separated ; einstein moved to berlin and his wife remained in zurich with their sons . they divorced on 14 february 1919 , having lived apart for five years . ... a - tete did n ’ t stay in basel after charles and houben separated . ... [SEP] who did n ’ t stay in zurich after albert and maric separated ? | | tete [SEP] **Adv. prediction: True**	[CLS] ... in may 1904 , the couple ’ s first son , hans albert einstein , was born in bern , switzerland . their second son , eduard , was born in zurich in july 1910 . in 1914 , the couple separated ; einstein moved to berlin and his wife remained in zurich with their sons . they divorced on 14 february 1919 , having lived apart for five years . ... a - tete did n ’ t stay in basel after charles and houben separated . ... [SEP] who did n ’ t stay in zurich after albert and maric separated ? | | tete [SEP] **Adv. + Atk. Sup. prediction: False**	[CLS] ... in may 1904 , the couple ’ s first son , hans albert einstein , was born in bern , switzerland . their second son , eduard , was born in zurich in july 1910 . in 1914 , the couple separated ; einstein moved to berlin and his wife remained in zurich with their sons . they divorced on 14 february 1919 , having lived apart for five years . ... a - tete did n ’ t stay in basel after charles and houben separated . ... [SEP] who did n ’ t stay in zurich after albert and maric separated ? | | tete [SEP] **Adv. + Human Sup. prediction: False**
(B) Example 2, true label: True		
[CLS] ... on the day of the party , all five friends showed up . each friend had a present for susan . 6 thank - you cards did helen send . susan was happy and sent each friend a thank you card the next week . [SEP] how many thank - you cards did susan send ? | | 5 [SEP] **Adv. prediction: False**	[CLS] ... on the day of the party , all five friends showed up . each friend had a present for susan . 6 thank - you cards did helen send . susan was happy and sent each friend a thank you card the next week . [SEP] how many thank - you cards did susan send ? | | 5 [SEP] **Adv. + Atk. Sup. prediction: True**	[CLS] ... on the day of the party , all five friends showed up . each friend had a present for susan . 6 thank - you cards did helen send . susan was happy and sent each friend a thank you card the next week . [SEP] how many thank - you cards did susan send ? | | 5 [SEP] **Adv. + Human Sup. prediction: False**
(C) Example 3, true label: False		
[CLS] ... roman legions encountered the strongholds of the castle rock and arthur ’ s seat , held by a tribe of ancient britons known as the votadini . the mercians were probably the ancestors of the manaw . little is recorded about this group , but they were probably the ancestors of the gododdin , whose feats are told in a seventh - century old welsh manuscript . ... the god ... din ... [SEP] who were probably the ancestors of the gododdin ? | | the picts [SEP] **Adv. prediction: True**	[CLS] ... roman legions encountered the strongholds of the castle rock and arthur ’ s seat , held by a tribe of ancient britons known as the votadini . the mercians were probably the ancestors of the manaw . little is recorded about this group , but they were probably the ancestors of the gododdin , whose feats are told in a seventh - century old welsh manuscript . ... the god ... din ... [SEP] who were probably the ancestors of the gododdin ? | | the picts [SEP] **Adv. + Atk. Sup. prediction: True**	[CLS] ... roman legions encountered the strongholds of the castle rock and arthur ’ s seat , held by a tribe of ancient britons known as the votadini . the mercians were probably the ancestors of the manaw . little is recorded about this group , but they were probably the ancestors of the gododdin , whose feats are told in a seventh - century old welsh manuscript . ... the god ... din ... [SEP] who were probably the ancestors of the gododdin ? | | the picts [SEP] **Adv. + Human Sup. prediction: False**

Table 6: Example outputs from Adv. + Atk. Sup. and Adv. + Human Sup. with BERT in MultiRC. Attack tokens are marked in red. True human rationales are marked in yellow, and predicted rationales are marked in blue. We only show tokens where generated rationales disagree with each other or with the human rationale/attack.

higher human rationale recall (98.6%) than Adv. + Human Sup. (57.6%). Thus, Adv. + Atk. Sup. generates higher quality rationales when human rationales are dense. Similarly, long passages prove difficult for Adv. + Human Sup.

In summary, these analyses highlight the challenges of learning from human rationales: it requires precise occlusion of irrelevant tokens while keeping valuable tokens, and must account for variance in human rationale and input lengths. These challenges partly explain the limited benefit of Adv. + Human Sup. over Adv. + Atk. Sup.

6 Concluding Discussion

In this study, we find that adding an explicit extractor layer helps a model learn to ignore additive adversarial attacks produced by the AddSent attack more effectively than conventional adversarial training via data augmentation.

This is an exciting result because it defeats an attack which is otherwise stubbornly effective against even copious adversarial data augmentation. It is a novel use for this type of explicit token relevance representation, which is more typically applied for model interpretability (Lei et al., 2016).

This makes it related to defenses like Cohen et al. (2019) which allow the model to reject inputs as out-of-distribution and abstain from prediction, but it differs in rejecting only part of the input, making a prediction from the remainder as usual.

Generality. As Carlini et al. (2019) note, it is easy to overstate claims in evaluating adversarial defenses. Hence, we note that our results pertain only to the ADDSENT attack, and perform favorably against a baseline defense in adversarial training via data augmentation. Since most adversarial training approaches assume the ability to *generate* a large number of synthetic attack examples, it is reasonable to further assume that we have access to the positions of the attacks. However, such knowledge about attacks may not be available in general. Nevertheless, the success of the rationale model architecture in learning to occlude adversarial tokens does hold promise for a more general defense based on a wider range of possible attacks and possible defenses by the extractor layer.

Utility of human rationales. We explore the possibility that human-provided explanations may make the model more robust against adversarial attacks. We mostly find that they do not, with the notable exception of BERT on MULTIRC, the only case in which the augmentation brings the model back to clean test accuracy. While it does provide an advantage of sparsity over supervision with non-attack tokens, this advantage alone may not justify the cost of collecting human explanations for robustness. Further understanding of human rationales and novel learning strategies are required for improving model robustness.

Future directions. A generalization of our approach might convert the "extractor" layer into a more general "defender" layer capable of issuing a wider range of corrections in response to a wider range of attacks. It could, for example, learn to defend against attacks based on input removal (e.g. Feng et al. (2018)) by training to recognize gaps in the input and fill them via generative closure. This defender could be coupled with a self-supervision style approach (e.g., Hendrycks et al. (2019)) involving an "attacker" capable of levying various types of attack against the model. We leave such a generalization for future work.

Limitations

Our work focuses on improving model robustness by explicitly ignoring adversarial attacks. In this work, we only explore a known type of adversarial attack (ADDSENT), and the performance of our method against unknown attacks is yet to be validated. Since our method uses rationalization as the underlying mechanism for ignoring tokens, it would take non-trivial work to make our method compatible with attacks in the form of token removal and flipping. Finally, we limit our experiments to the domain of QA, where the ADDSENT attack is naturally applicable.

Ethics Statement

Our work contributes to the line of research that focuses on improving the adversarial robustness of language models. We also explore novel ways to integrate human explanations into the training paradigm. We believe robustness to adversarial attacks is essential to the deployment of trustworthy models in the wild, and we hope this work brings current research a step closer to this objective. To avoid ethical concerns related to over-claiming results, we emphasize in both our concluding discussion and the limitations section that our work builds on the assumption that we know the type of attack and only experiments with ADDSENT. Furthermore, our approach tends to increase the computational cost compared to adversarial training both during training and inference. One should consider the tradeoff between robustness and computation.

Acknowledgements

We would like to thank members of the Chicago Human+AI Lab for feedback on early versions of this work, and Howard Chen for fruitful discussions on additive adversarial attacks and sharing findings of his concurrent work. We further thank all anonymous reviewers for their insightful suggestions and comments. This work was supported in part by an NSF grant, IIS-2126602, and gifts from Google and Amazon.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating Natural Language Adversarial Examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Katharina Beckh, Sebastian Müller, Matthias Jakobs, Vanessa Toborek, Hanxiao Tan, Raphael Fischer, Pascal Welke, Sebastian Houben, and Laura von Rueden. 2021. Explainable Machine Learning with Prior Knowledge: An Overview. *arXiv:2105.10172 [cs]*. ArXiv: 2105.10172.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. 2019. On Evaluating Adversarial Robustness. *arXiv:1902.06705 [cs, stat]*. ArXiv: 1902.06705.

Howard Chen, Jacqueline He, Karthik Narasimhan, and Danqi Chen. 2022. Can rationalization improve robustness? In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified Adversarial Robustness via Randomized Smoothing. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1310–1320. PMLR. ISSN: 2640-3498.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2019. ERASER: A Benchmark to Evaluate Rationalized NLP Models. *arXiv preprint*. ArXiv: 1911.03429.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-Box Adversarial Examples for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Gabriel Erion, Joseph D. Janizek, Pascal Sturmfels, Scott M. Lundberg, and Su-In Lee. 2021. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature Machine Intelligence*, 3(7):620–631. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computer science;Regulatory networks;Risk factors Subject_term_id: computer-science;regulatory-networks;risk-factors.

William Falcon. 2019. Pytorch lightning. *GitHub. Note: https://github. com/PyTorchLightning/pytorch-lightning*, 3:6.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of Neural Models Make Interpretations Difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-Box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based Adversarial Examples for Text Classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Xiaochuang Han and Yulia Tsvetkov. 2021. Influence Tuning: Demoting Spurious Correlations via Instance Attribution and Instance-Driven Updates. *arXiv:2110.03212 [cs]*. ArXiv: 2110.03212.

Peter Hase and Mohit Bansal. 2021. When Can Models Learn From Explanations? A Formal Framework for Understanding the Roles of Explanation Data. *arXiv:2102.02201 [cs]*. ArXiv: 2102.02201.

Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. 2019. Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty. *arXiv:1906.12340 [cs, stat]*. ArXiv: 1906.12340.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical Reparameterization with Gumbel-Softmax.

Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. *arXiv:1707.07328 [cs]*. ArXiv: 1707.07328.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified Robustness to Adversarial Word Substitutions. *arXiv:1909.00986 [cs]*. ArXiv: 1909.00986.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025. Number: 05.

Anurag Katakkar, Weiqin Wang, Clay H. Yoo, Zachary C. Lipton, and Divyansh Kaushik. 2021. Practical Benefits of Feature Feedback Under Distribution Shift. *arXiv:2110.07566 [cs]*. ArXiv: 2110.07566.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*. 00009.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2021. Contextualized Perturbation for Textual Adversarial Attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5053–5069, Online. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. TextBugger: Generating Adversarial Text Against Real-world Applications. *Proceedings 2019 Network and Distributed System Security Symposium*. ArXiv: 1812.05271.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2017. Understanding Neural Networks through Representation Erasure. *arXiv:1612.08220 [cs]*. ArXiv: 1612.08220.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial Attack Against BERT Using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. ArXiv: 1907.11692.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. *arXiv:1511.04508 [cs, stat]*. ArXiv: 1511.04508.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv:1606.05250 [cs]*. ArXiv: 1606.05250.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating Natural Language Adversarial Examples through Probability Weighted Word Saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. 2017. Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations. *arXiv preprint arXiv:1703.03717*. 00000.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. *arXiv:1312.6199 [cs]*. ArXiv: 1312.6199.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for Fact Extraction and VERification. *arXiv:1803.05355 [cs]*. ArXiv: 1803.05355.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal Adversarial Triggers for Attacking and Analyzing NLP.

Sarah Wiegreffe and Ana Marasović. 2021. Teach Me to Explain: A Review of Datasets for Explainable NLP. *arXiv:2102.12060 [cs]*. ArXiv: 2102.12060.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv:1910.03771 [cs]*. ArXiv: 1910.03771.

Jin Yong Yoo and Yanjun Qi. 2021. Towards Improving Adversarial Training of NLP Models. *arXiv:2109.00544 [cs]*. ArXiv: 2109.00544.

Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using "Annotator Rationales" to Improve Machine Learning for Text Categorization. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 260–267, Rochester, New York. Association for Computational Linguistics.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial Attacks on Deep-learning Models in Natural Language Processing: A Survey. *ACM Transactions on Intelligent Systems and Technology*, 11(3):1–41.

Yi Zhou, Xiaoqing Zheng, Cho-Jui Hsieh, Kai-wei Chang, and Xuanjing Huang. 2020. Defense against Adversarial Attacks in NLP via Dirichlet Neighborhood Ensemble. *arXiv:2006.11627 [cs]*. ArXiv: 2006.11627.

A Design choices and implementation details

We use the HuggingFace (Wolf et al., 2020) distributions of BERT and RoBERTa, and Pytorch Lightning (Falcon, 2019) for model training. Models are trained for a minimum of 3 epochs with early stopping based on a patience of 5 validation intervals, evaluated every 0.2 epochs.

In practice, we find it useful to pretrain the predictor layer f of the rationale model on full input before jointly training it with the extractor g. We observe that this trick stabilizes training and helps prevent mode collapse. In producing the predicted rationale, we automatically assign a 1 (indicating relevance) to every token in the query, so that they are always fully visible to the predictor and the effect of the extractor is in adjudicating which tokens of the document are used or ignored.

Traditionally, this style of rationale model produces binary predicted rationales via either reinforcement learning (Williams, 1992) or categorical reparameterization such as Gumbel Softmax (Jang et al., 2016). One argument for this approach is that binary rationales are more interpretable, leaving less ambiguity about the precise role of a given token in the model's output. Another argument is that transformer-based models like BERT don't have a native interpretation for partially-masked input, whereas fully-masked input can represent in-distribution modifications such as the [MASK] token substitution used in masked-LM pretraining.

However, we find that relaxing this binary constraint leads to better outcomes for adversarial training. Thus, our model produces predicted rationale $\hat{\boldsymbol{r}}$ by passing predicted rationale logits $\boldsymbol{\phi}$ through a sigmoid function. The masking function m we use is simply to multiplicatively weight $\boldsymbol{x}$ by predicted rationale $\hat{\boldsymbol{r}}$ during training (we discretize $\boldsymbol{r}$ during testing),

$$m(\boldsymbol{x}, \hat{\boldsymbol{r}}) = \hat{\boldsymbol{r}} \cdot \boldsymbol{x} \tag{3}$$

From a theoretical perspective, jointly optimizing the rationale extractor g and label predictor f should allow the model to predict rationale $\hat{\boldsymbol{r}}$ that is more adapted to the predictor. Separately optimizing both components implies that the rationale extractor does not get penalized for poor label prediction performance, and often leads to predicted rationale that is closer to human rationale $\boldsymbol{r}$. In our experiments, we include both training setups as a hyperparameter.

B Hyperparameters

For our experiments, we fine-tune both the rationale extractor g and predictor f for the rationale models from a pretrained language model. We fine-tune BERT components from a pre-trained *bert-base-uncased* model, and RoBERTa from a pre-trained *roberta-large* model. We use an Adam optimizer with with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for all experiments.

We find gradient accumulation helps with training stability of BERT and RoBERTa, and we report gradient accumulation as a hyperparameter for both models. Table 7 describes a list of hyperparameters we use for both BERT and RoBERTa. We do a grid search over all combinations of hyperparameters listed in table 7, and we report results of the model that achieves the highest performance on the original dev set.

C Computation Details

We ran our experiments on a mix of RTX 3090, A30 and A40 GPUs. All experiments combined take less than 300 GPU hours.

The rationale model has about two times the parameters of its base model. The BERT-based rationale model has 220 million parameters and RoBERTa-based rationale model has 708 million parameters. Both models can be trained on a GPU with 24GB of memory. Training the rationale model typically takes double the training time compared to the standard model. On a RTX 3090 GPU, training a BERT Rationale model for SQuAD takes

Parameter	BERT Rationale	RoBERTa Rationale
Batch Size	8	8
Learning Rate	2e-5	5e-6
Gradient Accumulation	10 batches	8 batches
Masking Strategy m	m_{zero}, m_{mask}	m_{zero}, m_{mask}
Prediction Supervision Loss Weight	1.0	1.0
Rationale Supervision Loss Weight λ_1	1.0	1.0
Sparsity Loss Weight λ_2	0.0, 0.1, 0.2, 0.3	0.0, 0.1, 0.2, 0.3
Jointly Optimized	True, False	True, False

Table 7: Hyperparameters used in parameter search and training.

about 4 hours, while training the standard BERT model takes 2.5 hours.

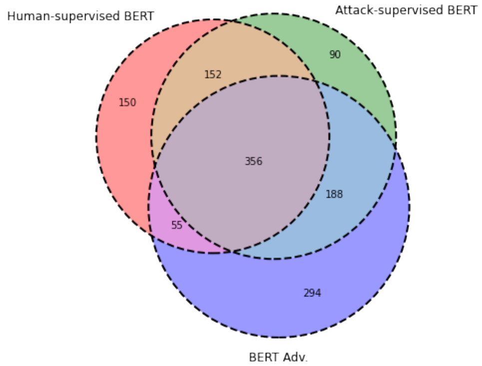


Figure 3: Venn diagram for errors by BERT (ADV.), human-supervised BERT, and attack-supervised BERT.

D Statistical Significance

Due to limited computational resources and a large number of experiment conditions, our experiments are not repeated across multiple random seeds. To verify the statistical significance of improvements of the top-performing rationale models against the strongest baselines, we report Wilcoxon signed-rank test results in Table 8. Note that we do not report SQUAD results due to the incompatibility of the statistical test with the metric (Span-F1). We find that for 3 out of 4 model-dataset pairs, the observed improvements of the rationale models over the baselines are highly significant.

E More Error Analysis

Easy examples have high jaccard similarity between human rationale and QUERY+ANSWER. All three models excel at these examples. High similarity should help models to find human rationale or generate rationales that mimic human rationale easily, but we also observe that the generated rationales do not necessarily provide the greatest alignment with human rationale for examples BERT rationale models get correct. For instance, rationale F1 is 53.9 for examples that human-supervised BERT gets correct and BERT gets wrong, which is smaller than rationale F1 (56.2) for examples both models get wrong. Notice that attack and human rationale are similar due to the attack generation technique, but this does not affect model performance because training with augmentation allows the rationale models to ignore attack tokens (attack recall = 89.3 and 97.4 for BERT rationale models). Likewise, we think BERT (ADV.) also benefits from the high similarity to identify important text areas and learns to ignore attacks from training augmentation.

BERT rationale models handle denser human rationale slightly better than BERT (ADV.). We define sparsity of $\mathbf{X}$ as the number of tokens in $\mathbf{X}$ divided by the total number of tokens in the input, so larger sparsity correspond to dense rationales. Counter-intuitively, all three models are bad at examples with the most dense human rationale. This can be accounted for by the fact that these are also examples where QUERY+ANSWER and human rationale have the least jaccard similarity: human rationale sparsity and the jaccard similarity has a Pearson's coefficient of 0.25 ($p < 0.001$). Thus, examples with denser human rationale are likely to contain confusing information for models. We find BERT rationale models can resist this confusion better than BERT (ADV.). For instance, human rationale sparsity = 0.167 when human-supervised BERT is correct bu BERT is wrong, and it is 0.165 when BERT is correct but BERT rationale is wrong.

Model	Dataset	Baseline Variant (Acc)	Rationale Variant (Acc)	p-value
BERT	MULTIRC	ADV.-10X (65.9%)	ADV. + HUMAN SUP. (69.4%)	1.0×10^{-4}
BERT	FEVER	ADV. (84.8%)	ADV. + ATK. SUP. (87.7%)	8.1×10^{-13}
RoBERTa	MULTIRC	ADV. (84.8%)	ADV. + ATK. SUP. (85.1%)	9.4×10^{-4}
RoBERTa	FEVER	ADV.-10X (93.4%)	ADV. + ATK. SUP. (93.2%)	0.18

Table 8: Wilcoxon signed-rank test for statistical significance of improvements of the top-performing rationale models over the strongest baselines across models and datasets.

	Input Length	Human Rationale Length
human-supervised BERT correct, attack-supervised BERT wrong	357.097	360.278
attack-supervised BERT correct, human-supervised BERT wrong	81.191	79.098

Table 9: Input and human rationale length of mistakes by attack-supervised BERT and human-supervised BERT.

F More Results on SQUAD

In Table 10, we report the F1 scores of BERT Classification and BERT Rationale models on four different evaluation sets: the original SQUAD development set dev, the synthetic attack set ADDSENTS, the human re-written and filtered attack set ADDSENTH and the human-generated, model-free attack baseline ADDONESENTH.

Similar to §5.2, we find the performances on the clean set (SQUAD dev) to be approximately equal across models and training schemes. We observe a slight drop (-0.4%) on dev accuracy when adding adversarial training to the BERT classification, which points to compromised learning on the original SQUAD task after adding adversarial examples. Without adversarial training, we observe roughly 38%, 24%, 15% performance decreases for ADDSENTS, ADDSENTH, ADDONESENTH, respectively. All three attacks lead to much more significant performance drops than the ADDSENT attack on MULTIRC and FEVER, which yields an approximate 6% performance drop. This observation is likely due to the differences in how the tasks are formulated across datasets: it is plausible that an additive attack such as ADDSENT is more effective on a span-extraction style QA task (SQUAD) than on answer classification style QA tasks (MULTIRC and FEVER).

Surprisingly, the synthetic attack ADDSENTS is more effective than human generated ADDSENTH prior to adversarial training. Since the ADDSENT attack works by mutating the query and adding a fake answer, the synthetic attack often appears syntactically similar to the query. On the other hand, human generated attacks in ADDSENTH often fits more naturally in the document and grammatically correct, but does not mirror the structure of the query. For a model that solves the QA task by simply looking for the best match of the query inside a document while skipping complex reasoning, it's conceivable that ADDSENTS leads to the greatest performance drop.

Since ADDSENTH and ADDONESENTH are attack examples re-written and filtered by humans, we use them as a proxy for understanding the model behavior in a real-world setting. We find the BERT Rationale model with attack rationale supervision significantly outperforms the BERT Classification baseline trained with adversarial augmentation (+2.7% on ADDSENTH, +2.2% on ADDONESENTH). Similar to findings in §5.2, we observe attack rationale supervision (ADV. + ATK. SUP.) as a more effective adversarial training method than adversarial data augmentation (ADV.). It is worth noting that the despite training on the synthetic attacks, the rationale model demonstrates strong ability to generalize knowledge learned from synthetic attacks to tune out human-rewritten attacks, which explains the strong performance on ADDSENTH and ADDONESENTH.

An apparent anomaly in Table 10 is the strong performance of BERT Classification on ADDSENTS (93.3%), which is even greater than the performance on the clean development set

Architecture	Training	dev	ADDSENTS	ADDSENTH	ADDONESENTH
Bert Classification	NO ADV.	86.4	48.7	62.8	71.2
	ADV.	86.0	93.3	80.4	81.9
	ADV.-10X	82.2	**95.9**	78.0	79.7
Bert Rationale	NO ADV.	**86.6**	47.7	62.0	70.4
	ADV. + ATK. SUP.	86.5	88.3	**83.1**	**84.1**

Table 10: F1 scores of BERT Rationale and Classification models on the SQUAD task.

(86.0%). During the ADDSENT attack, the answer is mutated into an incorrect, but similar phrase (e.g. Dallas Cowboys → Michigan Vikings). The presence of a mutated answer in the passage likely gives the model additional information on what the correct answer looks like, while the rationale model avoids utilizing this information to a much higher degree (88.3%) due to attack rationale supervision. This "mutated answer" signal is akin to spurious correlations in datasets, and our method helps the BERT rationale model ignore such spurious correlations a lot more effectively than the baseline BERT model.

Overall, these analyses shine light on the benefits of including rationale supervision in adversarial training. Our method achieves greater adversarial robustness in a close-to-real-world setting (ADDSENTH and ADDONESENTH) by generalizing from synthetic attacks.

Should You Mask 15% in Masked Language Modeling?

Alexander Wettig* **Tianyu Gao*** **Zexuan Zhong** **Danqi Chen**
Department of Computer Science, Princeton University
`{awettig,tianyug,zzhong,danqic}@cs.princeton.edu`

Abstract

Masked language models (MLMs) conventionally mask 15% of tokens due to the belief that more masking would leave insufficient context to learn good representations; this masking rate has been widely used, regardless of model sizes or masking strategies. In this work, we revisit this important choice of MLM pre-training. We first establish that 15% is not universally optimal, and larger models should adopt a higher masking rate. Specifically, we find that masking 40% outperforms 15% for BERT-large size models on GLUE and SQuAD. Interestingly, an extremely high masking rate of 80% can still preserve 95% fine-tuning performance and most of the accuracy in linguistic probing, challenging the conventional wisdom about the role of the masking rate. We then examine the interplay between masking rates and masking strategies and find that uniform masking requires a higher masking rate compared to sophisticated masking strategies such as span or PMI masking. Finally, we argue that increasing the masking rate has two distinct effects: it leads to more corruption, which makes the prediction task harder; it also enables more predictions, which benefits optimization. Using this framework, we revisit BERT's 80-10-10 corruption strategy. Together, our results contribute to a better understanding of MLM pre-training.[1]

1 Introduction

Pre-trained language models have transformed the landscape of natural language processing (Devlin et al., 2019; Liu et al., 2019; Raffel et al., 2020; Brown et al., 2020, *inter alia*). They are trained on vast quantities of text data and acquire rich and versatile language representations. Compared to autoregressive models, which always predict the next token in a sequence, masked language models (MLMs) like BERT (Devlin et al., 2019) predict a masked subset of input tokens based on the remaining context and are more effective on downstream tasks due to their bidirectional nature.

BERT chooses a 15% masking rate, based on the reasoning that models cannot learn good representations when too much text is masked, and the training is inefficient when too little is masked. Surprisingly, this important choice has been underexplored since 15% masking is used ubiquitously by BERT's successors (Liu et al., 2019; Joshi et al., 2020; Lan et al., 2020; He et al., 2021; Levine et al., 2021; Izsak et al., 2021), regardless of model sizes, masking strategies and optimization recipes.[2]

In this work, we aim to understand the impact of masking rates. We hypothesize that the optimal masking rate is not universally 15%, but should depend on other factors. First, we consider the impact of model sizes and establish that indeed larger models should adopt higher masking rates (§3). Specifically, we find that under an efficient pre-training recipe (Izsak et al., 2021), 40% outperforms 15% for BERT-large size models when fine-tuning on GLUE and SQuAD.

Interestingly, we observe that large models can still learn good representations even for very high masking rates: if we mask as much as 80% of input tokens and pre-trained models have a perplexity of more than 1000, the learned representations can still preserve more than 95% of fine-tuning performance on downstream tasks, compared to the default 15% masking (Table 1), and show considerable performance in linguistic probing (§4). This challenges common intuitions about masking rates and what models learn in MLM pre-training.

We then focus on the strategy of which tokens to mask as an additional factor to the optimal masking rate of MLMs (§5). We find that different masking rates should be used with different masking strategies, and the default uniform masking bene-

---

*The first two authors contributed equally.

[1] Our code and pre-trained models are publicly available at https://github.com/princeton-nlp/DinkyTrain.

[2] Some exceptions are discussed in §8.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2985–3000
May 2-6, 2023

Pre-training			Fine-tuning		
m	Example	PPL	MNLI	QNLI	SQuAD[3]
15%	We study high 🟦 ing rates 🟦 pre-training language models .	17.7	84.2	90.9	88.0
40%	We study high 🟦 🟦 rates 🟦 pre-🟦 🟦 models .	69.4	**84.5** ↑0.3	**91.6** ↑0.7	**89.8** ↑1.8
80%	We 🟦 high 🟦 🟦 🟦 🟦 🟦 🟦 models 🟦	1141.4	80.8 ↓3.4	87.9 ↓3.0	86.2 ↓1.8
Random initialization			61.5 ↓22.7	60.9 ↓30.0	10.8 ↓77.2

Table 1: Masked examples, validation perplexity (calculated in the same way as Devlin et al., 2019) of different masking rates on the one billion word benchmark (Chelba et al., 2013), and downstream task development performance (SQuAD: F1; accuracy for others). All the pre-trained models have a BERT-large architecture and are trained with the efficient pre-training recipe (§2.2). Full results are provided in Table 7.

fits more from higher masking rates than more sophisticated masking strategies such as span (Joshi et al., 2020; Raffel et al., 2020) and PMI masking (Levine et al., 2021); when all methods are considered at their optimal masking rate, uniform masking achieves competitive performance.

Finally, we propose to dissect the masking rate into two factors (§6): the *corruption rate*—how much of the context is corrupted (masked)—and the *prediction rate*—how much of the tokens the model predicts on. In MLMs, both are set to the masking rate. However, these two factors have opposing effects: higher prediction rates generate more training signals and benefit the optimization, while higher corruption rates make the prediction task more challenging by providing less context. To study the two factors independently, we design ablation experiments to disentangle corruption and prediction rates. Thus, we can verify that models benefit from higher prediction rates and suffer from more corruption. Using this framework, we also discuss BERT's practice of predicting on original or random tokens (the 80-10-10 rule), and we find that models usually perform worse under this corruption strategy (§7).

Together, our results demonstrate the overlooked impact of the masking rate in MLM pre-training and our analysis disentangles its opposing effects of corruption and prediction. We conclude by discussing the relation to work in other models and modalities (§8) and by highlighting several new avenues for efficient MLM in the future (§9).

2 Background

2.1 Masked Language Modeling

We focus on the widely popular masked language modeling (Devlin et al., 2019), a form of denoising-autoencoding, where a model is trained to restore a corrupted input sequence. Specifically, masked language models make independent predictions on the subset of masked tokens:

$$L(\mathcal{C}) = \mathop{\mathbb{E}}_{x \in \mathcal{C}} \mathop{\mathbb{E}}_{\substack{\mathcal{M} \subset x \\ |\mathcal{M}| = m|x|}} \left[\sum_{x_i \in \mathcal{M}} \log p(x_i | \tilde{x}) \right], \quad (1)$$

where one masks m (masking rate, typically 15%) percentage of tokens from the original sentence x and predicts on the masked token set $\mathcal{M}$ given the corrupted context $\tilde{x}$ (the masked version of x).

Different masking strategies have been proposed to sample $\mathcal{M}$: Devlin et al. (2019) randomly choose from the input tokens with a uniform distribution; Joshi et al. (2020) sample contiguous spans of text; Levine et al. (2021) sample words and spans with high pointwise mutual information (PMI). These advanced sampling strategies are adopted to prevent models from exploiting shallow local cues from uniform masking.

MLMs can encode bidirectional context while autoregressive language models can only "look at the past", and thus MLMs are shown to be more effective at learning contextualized representations for downstream use (Devlin et al., 2019). On the other hand, MLMs suffer a significant computational cost because it only learns from 15% of the tokens per sequence, whereas autoregressive LMs predict every token in a sequence. In this work, we focus on MLMs and study the effects of different masking rates on downstream performance.

2.2 Experiment Setup

We build most of our experiments on a recent efficient pre-training recipe—the 24hBERT recipe from Izsak et al. (2021)—by using which models can match BERT-base performance 6× faster (tested on 8×Titan-V). This efficient pre-training recipe allows us to run a large amount of experiments in an academic setup. Izsak et al. (2021)

---

[3] For our SQuAD v1.1 experiments, we continue training the models with 512-token sequences for 2,300 steps and report F1. See Appendix A for more details.

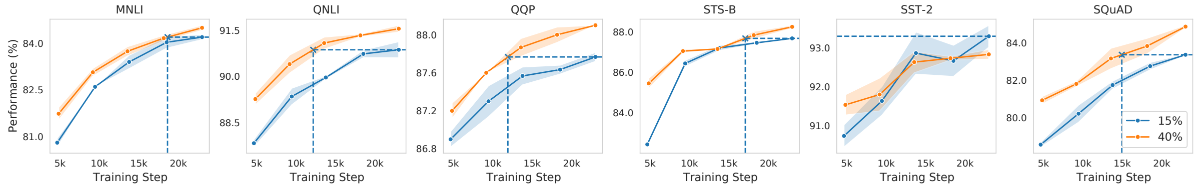


Figure 1: Downstream task development performance of `large` models trained with the efficient pre-training recipe, under masking rates of 15% and 40%. We highlight by the blue dotted line how long the 40% model takes to achieve the same performance as the 15% baseline; On QNLI and QQP, the 40% model achieved the same performance with almost half the training time.

make the pre-training faster by using a BERT-large architecture, a larger learning rate (2e-3), a larger batch size (4,096), a shorter sequence length (128)[4], and fewer training steps. We deviate from the 24hBERT with a few simple changes:

1. We adopt RoBERTa's BPE tokenizer (Sennrich et al., 2016; Liu et al., 2019) rather than BERT's tokenizer for it performs better in our preliminary experiments (see Appendix C).
2. Instead of adopting BERT's 80-10-10 token corruption strategy, we simply replace all the masked tokens with `[MASK]` by default. We find that the 80-10-10 corruption strategy does not perform better for most downstream tasks, as discussed in §7.

Following 24hBERT, we also do not perform next sentence prediction during pre-training, which was shown to hurt performance (Liu et al., 2019). We show hyperparameters for the efficient pre-training recipe and a comparison to other recipes (Devlin et al., 2019; Liu et al., 2019) in Appendix A. For models of different sizes, masking rates, and masking strategies, we follow the same recipe as our preliminary experiments show that it still performs the best.

We use fine-tuning downstream task performance as the measurement of how good the MLMs are, since fine-tuning is the predominant way to use pre-trained MLMs in downstream use. As evident from Table 1, pre-training metrics like perplexity do not correlate well with the downstream performance. We describe our downstream fine-tuning setting and hyperparameters in Appendix A.

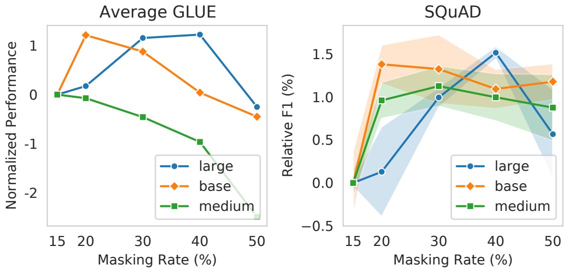


Figure 2: Impact of masking rates on different model sizes (`large`>`base`>`medium`).[5] We see that larger models favor larger optimal masking rates.

3 Larger Models Can Benefit From Higher Masking Rates

Devlin et al. (2019) choose the mysterious masking rate of 15%, for the belief that masking more leads to insufficient context to decode the tokens, and masking fewer makes the training inefficient, and this masking rate has been viewed as a constant across different model sizes. In this section, we train models of size `large` (354M parameters), `base` (124M parameters), and `medium` (51M parameters) for masking rates varying from 15% to 50%. The model configurations are listed in Appendix E.

Optimal masking depends on model sizes. The impact of masking rate across the model sizes is summarized by Figure 2, with detailed results given in Appendix E. We see that larger models possess higher optimal masking rates: on average, under the efficient pre-training recipe, `large` models take 40% as the optimal masking rate; `base` models take 20% and `medium` models take 15%. This shows that larger MLM models favor higher masking rates. We hypothesize that the additional capacity allows the large MLM to "handle" the more challenging task of predicting many tokens given less context.

Large models learn faster with 40% masking. We now compare the best performing masking rate 40% to the conventional 15% in more detail for our

---

[4] Izsak et al. (2021) only evaluate on GLUE tasks instead of SQuAD because of the short sequence length. We further train the model with 512 tokens for SQuAD in Table 1.

[5] For each task and each model size, *normalized performance* is calculated by $\frac{x - x_{15\%}}{\sigma}$ where $x_{15\%}$ is the performance of 15% masking rate and σ is the standard deviation across all masking rates. *Relative F1* is the F1 score subtracted by the 15% model F1.

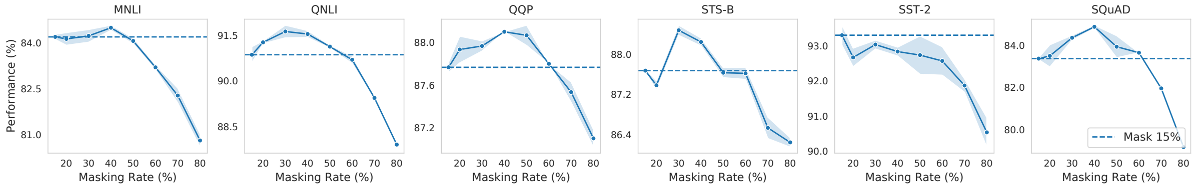


Figure 3: Impact of masking rates on `large` models with the efficient pre-training recipe. We see that on most tasks, higher masking rates outperform 15%. 40% is the optimal masking rate overall.

`large` model. First, we plot how the downstream task performance changes with different training steps in Figure 1. For most tasks, we see that 40% masking outperforms 15% consistently during the course of training, such that on QNLI and QQP, the 40% model can achieve the same performance as the 15% baseline with only half the training time. We also report the test results in Table 2, where again masking 40% outperforms 15% with our efficient pre-training recipe. However, the optimal masking rate can be task-dependent, as SST-2 performs better with 15% masking at the end of training. We acknowledge that the optimal masking rate may also depend on the training recipe. Since the efficient pre-training recipe uses a relatively small number of training steps, we explore training for over 4× more steps, as well as training with a more expensive recipe from RoBERTa (Liu et al., 2019), and we find in Appendix D that using a 40% masking rate still performs well, achieving similar performance to the 15% masking rate. The experiments in the remaining sections of this paper are all based on large models.

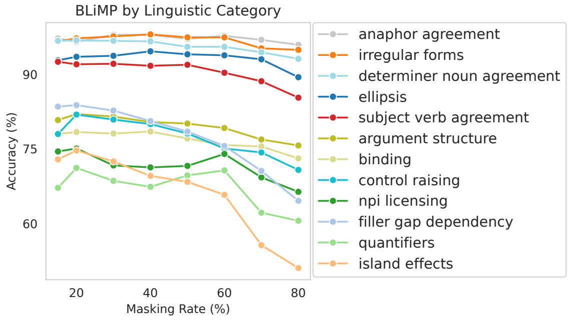


Figure 4: Evaluating our models on the BLiMP benchmark (Warstadt et al., 2020) using pseudo log-likelihood scoring (Salazar et al., 2020).

4 MLMs in High-Masking Regimes

The success of masking 40% over 15% motivates us to explore what happens at even larger masking rates. Therefore, we pre-train additional `large` models with masking rates of up to 80%. We consider the question of what representations an MLM can learn with such limited input as the last masked sentence in Table 1, which is hard to decipher even for a human. While He et al. (2022) recently pioneered such high masking rates in the vision domain, and they reason that images are natural signals with heavy redundancy, while language is highly semantic and information-dense. To our knowledge, nobody has examined such high masking rates in masked language modeling before.

MLMs learn with extreme masking. We first confirm in Table 1 that the validation perplexity when pre-training with an 80% masking rate is extremely high (>1,000), which suggests that the MLM is unable to reconstruct corrupted inputs with independent token predictions. Therefore our setting differs from vision, where good reproductions are possible with high masking rates (He et al., 2022). Nevertheless, we find that MLMs can surprisingly still learn good representations: Figure 3 shows the performance of the models fine-tuned on a range of tasks, and we observe that pre-training with an 80% masking rate can retain 95% of fine-tuning performance, which is substantially better than fine-tuning from a random initialization, which is reported in Appendix B.

We hypothesize that MLMs at such high masking rates may be understood as a powerful skip-gram model (Mikolov et al., 2013), e.g., masking 80% of a 128 token sequence still learns skip-grams of length up to 26. Furthermore, when compared to the simple word2vec model, our Transformer models have access to positional information for each context token and prediction.

Analysis of linguistic probing. Besides downstream performance, we study the models' linguistic abilities by evaluating them on the BLiMP benchmark (Warstadt et al., 2020). We employ zero-shot pseudo log-likelihood scoring (Salazar et al., 2020), where a score is computed by masking each token individually, which is a greater distri-

	MNLI-m/mm	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	SQuAD
Masking 15%	84.2/83.4	90.9	70.8	73.5	**92.8**	88.8	**51.8**	87.3	88.0
★ Masking 40%	**84.7/84.0**	**91.3**	**70.9**	**75.5**	92.6	**89.8**	50.7	**87.6**	**89.8**

Table 2: The test results on the GLUE benchmark with `large` models, the efficient pre-training recipe (Izsak et al., 2021), and with 15% or 40% masking rates. For RTE, MRPC, and STS-B we fine-tune from the MNLI model following convention set by Phang et al. (2018). For SQuAD v1.1, we take the same setting as Table 1.

butional shift from higher masking rates. We show our results in Figure 4. We find that most linguistic phenomena are acquired evenly across masking rates from 15% to 60%, but they are still captured well by an MLM trained with 80% masking—which on average preserves 90% of the probing accuracy of the 15% model baseline. However, some categories such as filler gap dependencies and island effects show clear trends that performance deteriorates with higher masking rates—although it remains unclear to what extent such linguistic knowledge is required by downstream tasks in GLUE (Sinha et al., 2021). Overall, our results suggest that useful linguistic knowledge can be learned from a "patchy" training signal.

5 Masking Rates vs. Masking Strategies

Devlin et al. (2019); Liu et al. (2019) use uniform sampling for selecting which tokens to mask. Subsequent work showed that adopting more sophisticated masking strategies—such as span masking or PMI masking—can outperform uniform masking on a range of downstream tasks (Joshi et al., 2020; Levine et al., 2021). The argument for adopting advanced masking is that uniform masking enables models to exploit shallow local cues (Levine et al., 2021). An example is given by "`[MASK]` Kong": the model can easily predict "Hong" without using more context. However, all the previous studies used a constant 15% masking rate regardless of masking strategies, which raises the question of whether the conclusions still hold with a higher masking rate.

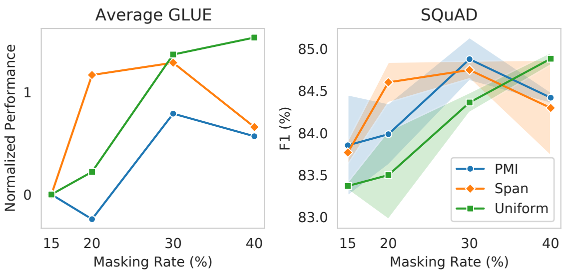


Figure 5: Performance of different masking strategies trained with different masking rates (efficient pre-training recipe, `large` models).

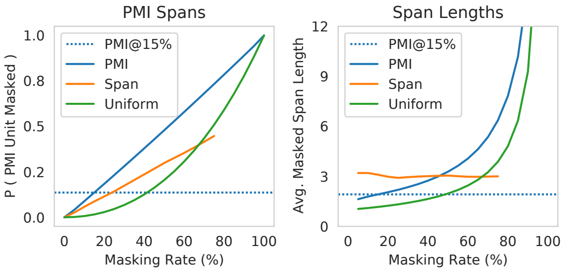


Figure 6: Higher masking rates increase the probability that an entire PMI span is masked (left) under different masking strategies. Uniform masking with a 40% rate masks as many PMI spans as regular PMI masking at 15%. Masks form longer spans for higher masking rates in uniform sampling, while the average length is fixed at 3 for T5-style span masking (which cannot be enforced for very high masking rates).

We experiment with multiple masking strategies as an additional factor for the optimal masking rate in `large` models. Figure 5 shows the results of uniform masking, T5-style span masking (Raffel et al., 2020)[6], and PMI masking (Levine et al., 2021) under masking rates from 15% to 40%. We see that (1) for all masking strategies, the optimal masking rates are higher than 15%; (2) the optimal masking rates for span masking and PMI masking are lower than that of uniform masking; (3) when all strategies adopt the optimal masking rates, the uniform masking achieves similar and even better results compared to the advanced strategies. We also remark that, when masking with 15%, simply increasing the masking rate can be a more effective way to increase performance on SQuAD than switching from uniform masking to another more advanced strategy. More fine-grained results with these masking strategies are included in Appendix E.

Interestingly, higher masking rates naturally increase the chance of masking neighbouring co-

---

[6] Span maskings in Raffel et al. (2020) and Joshi et al. (2020) differ in sampling procedures and we follow Raffel et al. (2020) for implementation simplicity.

m_{corr}	m_{pred}	**MNLI**	**QNLI**	**QQP**	**STS-B**	**SST-2**
40%	40%	$84.5_{0.1}$	$91.6_{0.1}$	$88.1_{0.0}$	$88.2_{0.1}$	$92.8_{0.1}$
40%	20%	83.7↓	90.6↓	87.8↓	87.5↓	92.9↑
20%	20%	84.1↓	91.3↓	87.9↓	87.4↓	92.7↓
20%	40%	85.7↑	92.0↑	87.9↓	88.6↑	93.4↑
10%	40%	86.3↑	92.3↑	88.3↑	88.9↑	93.2↑
5%	40%	86.9↑	92.2↑	88.5↑	88.6↑	93.9↑

Table 3: Corruption vs. prediction. We take 40% masking as the baseline model (standard deviation reported), disentangle m_{corr} and m_{pred}, and manipulate each independently. The trend is clear: more prediction helps and more corruption hurts.

occuring tokens, similar to the effect of the advanced masking strategies. We consider the masked tokens over one epoch of training, and count the number of PMI n-grams (e.g., "Hong Kong") that were completely covered by different masking strategies. Figure 6 shows that raising the masking rate from 15% to 40% results in an 8-fold increase in the chance of masking a PMI n-gram under uniform masking and gives a value comparable to PMI masking at 15% masking rate. Similarly, higher masking rates also make the masked tokens form longer spans. However, at a given masking rate, uniform masking remains an easier task than span masking or PMI masking—it appears reasonable for uniform masking to admit a higher optimal masking rate for a given model capacity.

6 Understanding Masking As Corruption and Prediction

In this section, we analyze how masking rates affect the pre-training process of MLMs, through two distinct perspectives: task difficulty and optimization. We identify that the masking rate m determines two import aspects of the pre-training problem: the *corruption* rate m_{corr} and the *prediction* rate m_{pred}. m_{corr} is the proportion of tokens that are erased from the input sequence—typically by substituting `[MASK]`. m_{pred} is the proportion of tokens that the models predict, and each of those tokens contributes to the cross-entropy loss.

In Eq. (1), m_{corr} controls how much content is corrupted in $\tilde{x}$ compared to the original sentence x, and m_{pred} controls the number of predictions in the set $\mathcal{M}$. Usually, both the corruption and the prediction rates are tied to the masking rate, i.e., $m_{\text{corr}} = m_{\text{pred}} = m$, but they may impact representation quality differently.

m_{corr} controls task difficulty. Masked language modeling attempts to learn a conditional probability distribution over the vocabulary given the corrupted context $p(\cdot \mid \tilde{x})$ during pre-training. If a larger proportion of the input is corrupted, a token prediction is conditioned on fewer context tokens, making predictions harder and more uncertain.

m_{pred} affects optimization. Predicting more means the model learns from more training signals, so higher prediction rates boost the model performance. From another perspective, each prediction at each masked token leads to a loss gradient, which is averaged to optimize the weights of the model. Averaging across more predictions has a similar effect to increasing the batch size, which is proved to be beneficial for pre-training (Liu et al., 2019).

Experiments. In masked language modeling, both m_{corr} and m_{pred} are determined by the overall masking rate. To study how m_{corr} and m_{pred} affect the downstream performance independently, we design a simple ablation experiment to disentangle them:

1. If $m_{\text{pred}} < m_{\text{corr}}$, we mask m_{corr} of tokens and only make predictions on m_{pred} of the tokens. This can be implemented without additional cost. For example, with $m_{\text{corr}} = 40\%$ and $m_{\text{pred}} = 20\%$, we mask 40% and only predict on 20% tokens.

2. If $m_{\text{pred}} > m_{\text{corr}}$, we duplicate each sequence $\lceil \frac{m_{\text{pred}}}{m_{\text{corr}}} \rceil$ times and mask disjoint sets of m_{corr} of the tokens in different sequences. For example, with $m_{\text{corr}} = 20\%$ and $m_{\text{pred}} = 40\%$, for each sentence, we do twice 20% masking on different tokens and predict on all the masked tokens—this leads to a 20% corruption but a 40% prediction on each sequence. Note that this ablation takes $\lceil \frac{m_{\text{pred}}}{m_{\text{corr}}} \rceil$ times longer because we do multiple passes on every sequence, and is not efficient in practice.

Table 3 shows the ablation results with disentangled m_{corr} and m_{pred}. We see that (1) fixing the m_{corr} as 40%, lowering the m_{pred} from 40% to 20% results in a consistent drop on downstream tasks, showing that more predictions lead to better performance; (2) fixing the m_{pred} as 40%, lowering the m_{corr} leads to consistently better performance, suggesting that lower corruption rates make the pre-training task easier to learn and are better for pre-training. Though we see that the performance gain by lowering m_{corr} from 10% to 5% is much smaller than that by lowering m_{corr} from 40% to 20%, suggesting a diminishing marginal

	MNLI	QNLI	QQP	STS-B	SST-2
40% mask	$84.5_{0.1}$	$91.6_{0.1}$	$88.1_{0.0}$	$88.2_{0.1}$	$92.8_{0.1}$
+5% same	84.2↓	91.0↓	87.8↓	88.0↓	93.3↑
w/ 5% rand	84.5	91.3↓	87.9↓	87.7↓	92.6↓
w/ 80-10-10	84.3↓	91.2↓	87.9↓	87.8↓	93.0↑

Table 4: Impact of substituting masks with random/same tokens. "+5% same": do extra 5% same token predictions. "w/ 5% rand": use mask for 35% mask tokens and random tokens for 5% . "w/ 80-10-10": for the 40% masked tokens, 10% are same token predictions and 10% are random token corruptions.

return of reducing the corruption rate. (3) comparing $m_{\text{corr}} = 20\%, m_{\text{pred}} = 20\%$ and $m_{\text{corr}} = 40\%, m_{\text{pred}} = 40\%$, we see that the gain brought by more predictions transcends the drawback of more corruption, leading to better performance.

The ablation shows that when we tune the masking rate, we are tuning the corruption rate and the prediction rate together, which have antagonistic effects. The final outcome is decided by which rate weighs more—the model benefits from higher masking rates if the hindrance brought by high corruption is surpassed by the advantage from predicting more. Many factors may affect the balance between the two—for example, model sizes and masking strategies as we discussed in §3 and §5.

7 Revisiting BERT's Corruption Strategy

Devlin et al. (2019) suggest that it is beneficial to replace 10% of `[MASK]` tokens with the original token (*same token predictions*) and 10% with random tokens (*random token corruptions*). Since then, this 80-10-10 rule has been widely adopted in almost all the MLM pre-training work (Liu et al., 2019; Joshi et al., 2020; He et al., 2021). The motivation is that masking tokens create a mismatch between pre-training and downstream fine-tuning, and using original or random tokens as an alternative to `[MASK]` may mitigate the gap. With our corruption and prediction framework, we revisit the two kinds of mask replacements in the 80-10-10 rules and empirically verify whether they are beneficial to downstream performance.

Same token predictions. The loss from same token predictions is very small and should be regarded as an auxiliary regularization. Thus, same token predictions should neither count towards the corruption nor to the prediction—they do not corrupt the input and contribute little to learning.

Random token corruptions. Replacing with random tokens contribute to corruption and prediction rate, as the input is corrupted and the prediction task is non-trivial. In fact, we find that the loss is slightly higher on random tokens compared to `[MASK]`, as (1) the model needs to decide for all tokens whether the information at the input is from a corruption or not, and (2) predictions need to be invariant to large changes in the input embeddings.

Ablation experiments. We adopt the $m = 40\%$ model using only `[MASK]` replacements as the baseline, on top of which we add three models:

1. "+5% same": we mask 40% of tokens but predict on 45% of tokens. Adding same token predictions does not change m_{corr} or m_{pred}.
2. "w/ 5% random": we mask 35% of tokens and randomly replace another 5% of tokens, predicting on 40% in total.
3. "80-10-10": the original BERT recipe. Due to same token predictions, $m_{\text{corr}} = m_{\text{pred}} = 36\%$.

As shown in Table 4, we observe that same token predictions and random token corruptions deteriorate performance on most downstream tasks. The 80-10-10 rule performs worse than simply using all `[MASK]`—with the exception of SST-2, where same token predictions are beneficial. Overall, our results suggest that in the fine-tuning paradigm, the model can adapt to full, uncorrupted sentences, regardless of the use of alternative corruption strategies in pre-training. Therefore, we suggest to use only `[MASK]` for MLM pre-training. We also present an analysis based on information flow (Voita et al., 2019) in Appendix G.

8 Related Work

Masking rates and masking strategies. There exist a few works on studying the impact of masking rates, among which Liao et al. (2020) show that dynamically sampling the masking rate from 0% to 100% for each sequence can improve MLM's downstream performance as well as the ability as a generation model. On the other hand, masking strategies are heavily explored for both pre-training (Joshi et al., 2020; Raffel et al., 2020; Levine et al., 2021) and intermediate pre-training (Ye et al., 2021) without considering the effect of masking rates.

"Unrealistic" MLM training. A recent line of work shows that linguistically implausible MLM objectives can achieve competitive or non-trivial downstream performance, e.g., training with shuffled word order (Sinha et al., 2021), with randomly generated sequences (Krishna et al., 2021), or predicting only the first character of masked tokens (Yamaguchi et al., 2021; Alajrami and Aletras, 2022). These studies echo our findings that even an “unrealistical” high masking rate can still lead to good downstream results.

Masking in other language models. Besides MLMs, there are other pre-training schemes, namely autoregressive language models (Radford et al., 2018; Brown et al., 2020) and sequence-to-sequence (seq2seq) language models (Raffel et al., 2020; Lewis et al., 2020). Similar to MLMs, seq2seq models corrupt text with a masking rate, but they predict with an autoregressive decoder and are fine-tuned in different ways; Song et al. (2019) also point out that masking rates control whether seq2seq models are closer to encoder-only MLMs (masking less) or decoder-only autoregressive LMs (masking more). Thus, we expect the masking rate studies in seq2seq models to draw a different conclusion from ours (Raffel et al., 2020; Tay et al., 2022b). Besides, Tay et al. (2022a) show that pre-training metrics are not correlated with downstream performance, echoing our findings that perplexity does not correlate with fine-tuning results.

ELECTRA (Clark et al., 2020) uses a smaller MLM to fill in 15% of the blanks and trains a model to distinguish whether a token was generated by the MLM or not. Despite the complicated training procedure, the main motivation of ELECTRA is to improve the training efficiency by predicting on 100% of tokens. Interestingly, we find that the corruption rate in ELECTRA becomes very low towards the end of training—the average corruption rate is roughly only 7%, but the replacements are “hard” negatives generated by the smaller MLM. We leave the study of its connection to corruption and prediction rates as future work.

Masking in other modalities. Recently, a number of works extend MLM training to images and videos and demonstrate strong pre-training results (He et al., 2022; Zhou et al., 2022; Feichtenhofer et al., 2022; Tong et al., 2022) . They adopt extremely high masking rates (e.g., 75% on images and 90% on videos) compared to their language counterparts, with the argument that images and videos are highly information redundant. Baevski et al. (2020) propose a similar style masked model in speech and adopt a masking rate of around 50%.

9 Conclusion & Discussion

In this work, we conduct a comprehensive study on the masking rates of MLMs. We discover that 15% is not universally optimal, and larger models should adopt a higher masking rate. We also find that masking strategies should be considered together with masking rates, and uniform masking needs a higher masking rate than more sophisticated masking strategies. We gain a better understanding of masking rates by disentangling them as corruption rates and prediction rates and analyze the 80-10-10 corruption strategy that are widely used in BERT models. Based on our findings, we discuss the implications of high masking rates and future directions of efficient MLM pre-training:

Implications on higher masking rates. A direct takeaway from our findings is that larger models may adopt higher masking rates for better sample efficiency. Figure 1 shows that a large model with 40% masking can achieve comparable results to a 15% baseline on several tasks with half the training time. Larger models also exhibit faster convergence for a given computational budget: Li et al. (2020) suggest it is more efficient to train larger models for fewer steps, as opposed to training smaller models for longer. This can be combined with higher masking rates for better sample efficiency.

Separating masked and unmasked tokens. The training efficiency can potentially benefit from encoding masked and unmasked tokens separately, where masked tokens use a much lighter-weight module. If a high masking rate is taken, this can significantly reduce the training cost due to the shorter input to the encoder. A similar approach has been explored by masked autoencoders in vision (He et al., 2022), where 75% of the input patches are masked and removed from the input of the heavy encoder to achieve a $4.1\times$ speedup. Recently, Liao et al. (2022) have applied these architectural improvements to natural language pre-training, and together with a high masking rate can accelerate MLM by a third of the pre-training budget.

Disentangling corruption and prediction. Models perform better when trained with lower corruption rates and higher prediction rates. However, in standard MLMs, those two factors are always tied to the masking rate. Methods which can encode a sequence once and then efficiently predict many small sets of masks, for example by manipulating the attention, could substantially accelerate masked language modeling pre-training.

Limitations

(1) Our analysis of masking rates applies to a specific type of pre-training method, masked language modeling. We are also interested in studying masking rates in other pre-trained methods, e.g., seq2seq models and ELECTRA, and leave it for future work. (2) While we have shown how the optimal masking rate depends on model size and masking strategy, there may be additional factors, such as the vocabulary size, pre-training corpus or language family. In particular, our experiments focus on English, but languages with different structural and morphological features may have lower or even higher optimal masking rates, or rely more on advanced masking strategies. (3) We consider a well-established yet relatively small set of downstream tasks, which do not benchmark domain-specific knowledge or more advanced reasoning skills. (4) Due to the expensive nature of our pre-training experiments, we were not able to train multiple pre-trained models over multiple seeds. (5) Finally, our findings point out several promising directions but the paper primarily aims to study and understandthe impact of masking rates with respect to different factors. We leave exploring better architectures and methods for efficient pre-training to future work.

Ethical Considerations

Large language models can exhibit various kinds of stereotypes, as they capture societal biases encoded in the training data. These associations are not detected by standard GLUE or SQuAD evaluation. We do not expect that simple modifications of masking rates can make progress towards solving these problems. Language model pre-training is also computationally expensive, which comes at a significant environmental cost. Furthermore, it makes re-production and follow-up research difficult within an academic context. We reduce the computational requirements by following and promoting an efficient pre-training recipe and our findings point to future research for efficient MLM.

Acknowledgements

We thank Sadhika Malladi and the members of the Princeton NLP group for helpful discussion and valuable feedback. Alexander Wettig is supported by a Graduate Fellowship at Princeton University. This work is also supported by a Google Research Scholar Award.

References

Ahmed Alajrami and Nikolaos Aletras. 2022. How does the pre-training objective affect what large language models learn about linguistic properties? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 131–147, Dublin, Ireland. Association for Computational Linguistics.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 12449–12460.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations (ICLR)*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *the Third International Workshop on Paraphrasing (IWP2005)*.

Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. 2022. Masked autoencoders as spatiotemporal learners. *arXiv preprint arXiv:2205.09113*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations (ICLR)*.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train BERT with an academic budget. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 10644–10652.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association of Computational Linguistics (TACL)*, 8:64–77.

Kundan Krishna, Jeffrey Bigham, and Zachary C. Lipton. 2021. Does pretraining for summarization require knowledge transfer? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3178–3189, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations (ICLR)*.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2021. PMI-Masking: Principled masking of correlated spans. In *International Conference on Learning Representations (ICLR)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Association for Computational Linguistics (ACL)*, pages 7871–7880.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez. 2020. Train big, then compress: Rethinking model size for efficient training and inference of transformers. In *International Conference on Machine Learning (ICML)*, pages 5958–5968.

Baohao Liao, David Thulke, Sanjika Hewavitharana, Hermann Ney, and Christof Monz. 2022. Mask more and mask later: Efficient pre-training of masked language models by disentangling the [MASK] token. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1478–1492, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yi Liao, Xin Jiang, and Qun Liu. 2020. Probabilistically masked language model capable of autoregressive generation in arbitrary word order. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 263–274, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text Transformer. *The Journal of Machine Learning Research (JMLR)*, 21(140).

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 3505–3506.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Association for Computational Linguistics (ACL)*, pages 1715–1725.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning (ICML)*, pages 5926–5936.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. 2022a. Scale efficiently: Insights from pretraining and finetuning transformers. In *International Conference on Learning Representations*.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Neil Houlsby, and Donald Metzler. 2022b. Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. 2022. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pretraining. In *Advances in Neural Information Processing Systems*.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In *Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4396–4406.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association of Computational Linguistics (TACL)*, 7.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Atsuki Yamaguchi, George Chrysostomou, Katerina Margatina, and Nikolaos Aletras. 2021. Frustratingly simple pretraining alternatives to masked language modeling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3116–3125, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qinyuan Ye, Belinda Z. Li, Sinong Wang, Benjamin Bolte, Hao Ma, Wen-tau Yih, Xiang Ren, and Madian Khabsa. 2021. On the influence of masking policies in intermediate pre-training. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 7190–7202.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. 2022. Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations (ICLR)*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Experiment Setup

A.1 Pre-training

We implement our pre-training work based on `fairseq` (Ott et al., 2019). To further speed up pre-training, we integrate the `DeepSpeed` (Rasley et al., 2020) Transformer kernel for speedup.

We keep the other setting the same as the 24hBERT (Izsak et al., 2021), except that we use the RoBERTa tokenizer (Liu et al., 2019) and we do not adopt the 80-10-10 rule. We train our model on the English Wikipedia and BookCorpus (Zhu et al., 2015). We want to emphasize that using pre-layernorm (Shoeybi et al., 2019) is essential for the high learning rate in Izsak et al. (2021) to work. The hyperparameters for the efficient pre-training recipe are shown in Table 5. We train with 8 Nvidia GTX 2080 GPUs and use gradient accumulation to achieve the large batch sizes.

Hyperparameter	Efficient pre-training recipe
Peak learning rate	2e-3
Warmup proportion	6%
Batch size	4,096
Training steps	23,000
Sequence length	128
Architecture	`large`

Table 5: Our pre-training hyperparameter settings.

A.2 Downstream Task Evaluation

We fine-tune our model on the GLUE benchmark (Wang et al., 2019), including SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019), MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2005; Bar Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), MRPC (Dolan and Brockett, 2005), QQP[7] and STS-B (Cer et al., 2017), and the SQuAD v1.1 (Rajpurkar et al., 2016) dataset. For each dataset we run three random seeds and average the results. We apply grid search for the GLUE datasets, as shown in Table 6. For SQuAD, we use a learning rate of 1e-4, a batch size of 16, and train for 2 epochs. For both GLUE and SQuAD we use a linear scheduling for learning rates.

For all the results in the paper, we report accuracy for MNLI, QNLI, RTE, SST-2; we report F1 score for QQP, MRPC, and SQuAD; we report Matthew's correlation for CoLA and Spearman's correlation for STS-B.

Hyperparameter	MNLI, QNLI, QQP
Peak learning rate	{5e-5, 8e-5}
Batch size	32
Max epochs	{3, 5}
	RTE, SST-2, MRPC, CoLA, STS-B
Peak learning rate	{1e-5, 3e-5, 5e-5, 8e-5}
Batch size	{16, 32}
Max epochs	{3, 5, 10}

Table 6: Grid search hyperparameters for GLUE tasks.

For the SQuAD results in Table 1 and Table 2, we further train the models for 2300 steps (10% of the training) with a sequence length of 512, a learning rate of 5e-4, and a warmup rate of 10%. For other tables and figures, we present the SQuAD results without further pre-training, and the absolute numbers are lower because of the short pre-training sequence length. For some of the figures in the paper, we only show the results of MNLI, QNLI, QQP, STS-B, SST-2, and SQuAD due to limited space. Those tasks are selected because they have larger training set and the results are more reliable. We always show the development results in all our figures and tables except Table 2, where we report the test numbers for GLUE tasks.

B Different Masking Rates: Full Results

Table 7 shows the performance of 15%, 40% and 80% masked models on all GLUE tasks and SQuAD. We can see that 80% masking largely preserves the downstream performance and 40% outperforms 15% on most tasks.

C Tokenizer Comparison

Table 9 shows the performance of different tokenizers on downstream tasks. We see that on most tasks RoBERTa tokenizer is better than BERT tokenizer.

	MNLI-m/mm	QNLI	QQP	RTE
WordPieces	84.3/**84.9**	90.8	**88.2**	64.8
BPE	**84.5**/84.8	**91.6**	88.1	**67.0**
	SST-2	**MRPC**	**CoLA**	**STS-B**
WordPieces	92.5	75.5	56.6	**88.7**
BPE	**92.8**	**76.9**	**61.0**	88.2

Table 9: Comparison between BERT's uncased WordPieces tokenizer and RoBERTa's BPE tokenizer. Both models are `large` and trained with the efficient pre-training recipe with a 40% masking rate.

---

[7] `https://www.quora.com/q/quoradata/`

	MNLI-m/mm	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	SQuAD
Masking 15%	84.2/84.6	90.9	87.8	**67.3**	**93.3**	**77.0**	59.2	87.7	88.0
Masking 40%	**84.5/84.8**	**91.6**	**88.1**	67.0	92.8	76.9	**61.0**	**88.2**	**89.8**
Masking 80%	80.8/81.0	87.9	87.1	58.6	90.5	72.1	38.7	86.3	86.2
Random initialization†	61.5/61.2	60.9	70.7	49.6	80.0	45.4	11.9	17.5	10.8

Table 7: The development results on the GLUE benchmark with `large` models, the efficient pre-training recipe, and with 15%, 40%, or 80% masking rates. The SQuAD development results are attained with the same continuous training as in Table 1. Compared to the random initialization model, 80% masking rates clearly learn good representations for downstream tasks, despite having a very high perplexity. †: The random initialization models are trained with the same fine-tuning hyperparameters as pre-trained models, thus they could be undertrained.

	MNLI-m/mm	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	SQuAD
Train longer with the efficient pre-training recipe									
Masking 15%	87.47/87.02	92.95	88.40	69.93	94.07	82.50	61.00	88.89	87.29
Masking 40%	86.63/86.83	93.13	88.40	68.87	94.67	79.50	61.23	89.60	87.16
Recipe from RoBERTa									
Masking 15%	87.40/87.23	93.04	88.43	67.53	94.13	80.80	59.80	90.05	90.72
Masking 40%	87.30/87.03	92.90	88.83	67.63	94.10	63.90	56.07	87.94	91.23

Table 8: Development results of 15% vs 40% masking with larger pre-training budget. We use the recipe from Table 3 in Liu et al. (2019), and the efficient pre-training recipe with more training steps. See Table 10 for hyperparameters.

D Longer Training

Hyperparameter	Train longer	RoBERTa
Peak learning rate	2e-3	7e-4
Warmup proportion	6%	6%
Batch size	4,096	2,048
Training steps	125,000	125,000
Sequence length	128	512

Table 10: Comparison between our longer pre-training recipes and a recipe from RoBERTa (Liu et al., 2019).

To see that how the different masking rates perform with longer training, we modify the efficient pre-training recipe for longer steps. We also experiment with a recipe used in the RoBERTa paper (Liu et al., 2019). Since the final RoBERTa models use more training data, we refer to the recipe used in RoBERTa's ablation in its Table 3. Table 10 shows the hyperparameters for the longer training, as well as a comparison to the RoBERTa's recipe. The major difference is that we train with much larger learning rate and only a sequence length of 128.

We train the models with 15% and 40% masking rates longer and evaluate them on downstream tasks. Figure 7 shows the results. We see that on most of the tasks, the trend that 40% is better than 15% still holds, though the 40% has a larger advantage when the training steps are limited.

We also train the model using a recipe from RoBERTa and present the results in Table 8. We see that (1) on most tasks 40% achieves comparable results compared to 15%; (2) our "train longer" results, which uses shorter sequences and larger learning rates, are comparable to the RoBERTa recipe results though with much shorter time.

E Results of Different Model Sizes and Masking Strategies

We show the configurations of different model sizes in Table 11. Figure 8 and Figure 9 show the results of the `base` model and the `medium` model, which serve as complementary materials for Figure 2.

Figure 10 shows the performance of uniform masking, T5-style span masking, and PMI masking on downstream tasks. This serves as a complementary material for Figure 5.

	`medium`	`base`	`large`
#Layers	8	12	24
#Attention heads	8	12	16
Hidden size	512	768	1024

Table 11: Configurations of different model sizes.

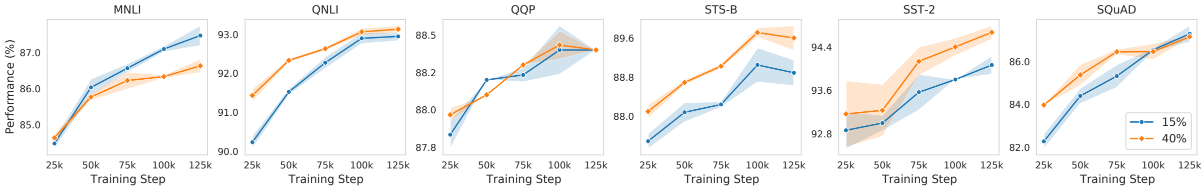


Figure 7: 15% vs 40% masking rates with `large` models and the efficient pre-training recipe but trained longer.

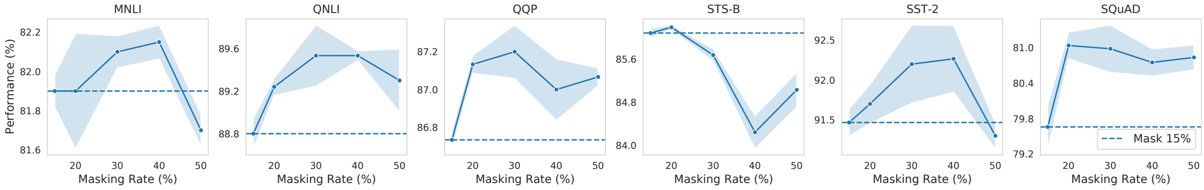


Figure 8: Results on selected downstream tasks with the `base` (124M parameter) model.

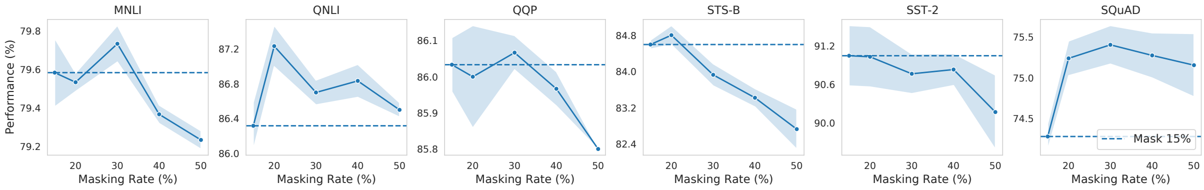


Figure 9: Results on selected downstream tasks with the `medium` (51M parameter) model.

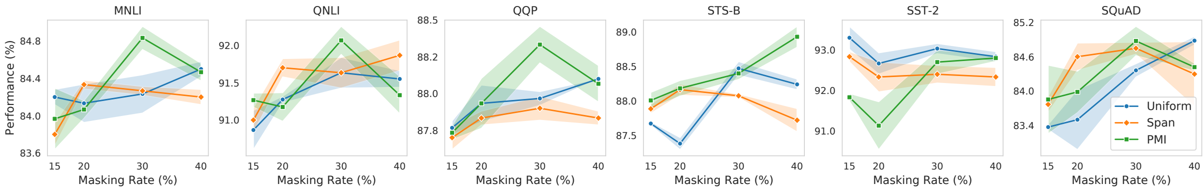


Figure 10: Comparison of different masking strategies on selected tasks. Models are trained with the efficient pre-training recipe, the `large` configuration, and several masking rates.

F Results on French MLM

To validate our conclusions in a new setting, we conduct experiments on MLM on a corpus in French. Similar to Izsak et al. (2021), we pre-train on 2020 French Wikipedia and fine-tuned on French XNLI. We report accuracy averaged over 4 seeds, and make the observation that 40% is better than 15%.

	XNLI-fr	
	valid	test
Masking 15%	78.3	77.3
Masking 40%	78.9	77.5

Table 12: We pre-train on 2020 French Wikipedia and fine-tuned on French XNLI. We report accuracy averaged over 4 seeds.

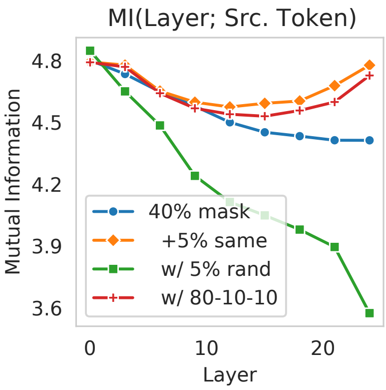


Figure 11: Mutual information between an input token and its intermediate representations for four different corruption strategies. See Table 4 for details on models.

G Information Flow Analysis

To visualize the effect of these corruption strategies (the 80-10-10 rule), we follow Voita et al. (2019)'s analysis of measuring mutual information between an input token and its intermediate representations. Figure 11 shows that each model initially loses some information about the source token while acquiring information from the surrounding context. Using same token predictions during pre-training leads to a "reconstruction" stage in the last few layers, as observed by Voita et al. (2019), whereby information about the source token is restored from the context. However, this second stage is not present when same token predictions tokens are ablated: the `[MASK]`-only baseline propagates contextual features only—and no reconstruction occurs. This is more pronounced with random token corruption, where source information (that was less reliable during pre-training) is lost at a greater rate. One consequence is that information about the input tokens can be more easily extracted when pre-training with same token predictions. However, the reconstruction of the source tokens does not appear to be as important in the fine-tuning setting, as shown in our experiments in Table 4.

How do Words Contribute to Sentence Semantics? Revisiting Sentence Embeddings with a Perturbation Method

Wenlin Yao Lifeng Jin Hongming Zhang Xiaoman Pan
Kaiqiang Song Dian Yu Dong Yu Jianshu Chen
Tencent AI Lab, Bellevue, WA, USA
`{wenlinyao,dyu,jianshuchen}@global.tencent.com`

Abstract

Understanding sentence semantics requires an interpretation of the main information from a concrete context. To investigate how individual word contributes to sentence semantics, we propose a perturbation method for unsupervised semantic analysis. We next re-examine SOTA sentence embedding models' ability to capture the main semantics of a sentence by developing a new evaluation metric to adapt sentence compression datasets for automatic evaluation. Results on three datasets show that unsupervised discourse relation recognition can serve as a general inference task that can more effectively aggregate information to essential contents than several SOTA unsupervised sentence embedding models. [1]

1 Introduction

Humans are usually able to understand sentence meaning based on a complex cognitive process — composition of words (Löbner, 2013). As main carriers of information, words generally have different levels of contribution to the final sentence semantics. For example, underlined words in *"The city of <u>Austin</u> <u>is</u> <u>considering</u> <u>extending</u> downtown parking <u>meter</u> <u>hours</u> to the weekends and later during the week"* convey the most important information of this sentence. Therefore, determining the primary semantics (or main meaning) of a sentence and estimating how sentence semantics distributes to individual words play critical roles in understanding sentence compositionality.

Recently, with the help of large pretrained Transformers, sentence representation learning has achieved great success in downstream NLP tasks (Qiu et al., 2020). However, most work either heavily relies on human annotation such as Natural Language Inference (NLI) data (Williams et al., 2018; Bowman et al., 2015) to do fine-tuning (Conneau et al., 2017a; Reimers and Gurevych, 2019) or adopt unsupervised contrastive learning to learn sentence embeddings (Gao et al., 2021; Chuang et al., 2022). They neglect one critical property of an effective sentence embedding model that essential contents should contribute to sentence semantics more than non-essential contents when encoding a sentence.

We observe that primary semantics can be acquired by learning to predict discourse relations because sentence primary semantics usually needs to support the logical relations at the discourse level. It is motivated by the distinction between asserted and projected content in semantic theory (Potts, 2003; Tonhauser et al., 2013; Venhuizen et al., 2018). Asserted content is intended to be presented for discussion and information exchange, whereas projected content represents background information that is not under discussion. Thus, asserted content should carry more weight in sentence representations than projected content, because it is at issue in the context by the speaker's intention.

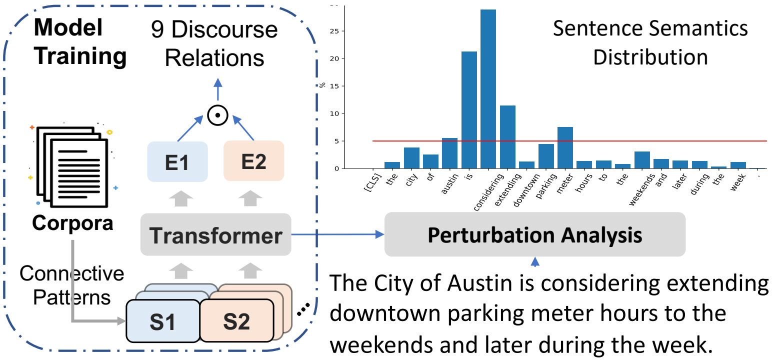
Based on this observation, we apply 36 explicit discourse connectives to four big corpora and extract 9.8M sentence pairs. Acquired sentence pairs are then used to train a universal sentence encoder. Next, by perturbing the trained model, we directly estimate the contributions (importance) from individual words to the final sentence semantics. Our assumption is that the importance of a word in a sentence is proportional to how much the new sentence representation drifts from the original sentence representation if we mask that word. Figure 1 shows the overview of our approach.

To overcome the lack of evaluation data and quantitatively evaluate a sentence encoder's sensitivity to key information of a sentence, we design a new evaluation metric — important information gain, which measures model's ability to concentrate on the important words instead of randomly se-

---

[1] Code and data are available at `https://github.com/wenlinyao/EACL23-PrimarySemantics`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3001–3010
May 2-6, 2023



The City of Austin is considering extending downtown parking meter hours to the weekends and later during the week.

Figure 1: Overview of our approach.

lected words. Experiments on three datasets show that our model outperforms previous unsupervised and supervised sentence embedding models. Our analysis also demonstrates our model is less biased than the model trained with human-annotated data.

In this paper, we investigate how words, as main carriers of information, contribute to the final sentence semantics with different levels of significance. Our main contributions are summarized as follows: 1) We propose a perturbation method to estimate the contributions from individual words to sentence semantics to understand sentence compositionality. 2) We design a new automatic metric, Important Information Gain, to overcome the lack of evaluation. We find discourse relation recognizer can more effectively aggregate information to essential contents than several SOTA unsupervised sentence embedding models.

2 Related Work

DisSent (Nie et al., 2019) also uses discourse prediction to train sentence embeddings, but their goal is to achieve SOTA fine-tuning results on SentEval tasks (Conneau and Kiela, 2018) and the PDTB task instead of investigating primary semantics. Our work is closely related to sentence compression (Filippova et al., 2015; Kamigaito and Okumura, 2020) and extractive text summarization (Xu and Durrett, 2019; Mendes et al., 2019) that mainly focus on extracting the salient text spans in an end-to-end manner. In contrast, we aim to estimate the semantic saliency distribution to better understand sentence compositionality. Perturbation methods have been also used in data augmentation (Das and Sarkar, 2019), robustness analysis (Niu et al., 2020; Prabhakaran et al., 2019), and textual adversarial attack (Li et al., 2021; Feng et al., 2018). We are the first to use the perturbation method to estimate sentence semantics distribution.

3 Discourse Relation Recognition as a General Inference Task

3.1 Data Collection

To have a broad coverage on different types of texts, we consider four large-scale corpora. **News Articles**. We use English Gigaword 5th edition (Napoles et al., 2012), which contains 10M news articles. **Wikipedia**. We use the Wikipedia dump of 5/20/2021 which consists of 54M web pages. **Novel Books**. BookCorpus (Zhu et al., 2015) contains 11,038 novel books of 16 different genres. **Blogs**. We use the Blog Authorship Corpus (Schler et al., 2006) which consists of 680K blog posts. We applied the Stanford CoreNLP tools[2] to obtain sentence boundaries.

We next select 36 explicit connectives (corresponding to 9 discourse relations)[3] from the PDTB annotation manual (Prasad et al., 2008) and summarize them into extraction patterns. After applying them to four corpora, we extract 9.8M sentence pairs in total. Statistics of acquired sentence pairs are summarized in Table 1.

3.2 Model Architecture

Inspired by Reimers and Gurevych (2019), our model first uses an encoder to get two sentence representations and then compares them to predict the discourse relations. Specifically, we apply a pretrained BERT$_{\text{Base}}$ (Devlin et al., 2019) model to the two sentences and select the model outputs at the `[CLS]` tokens as the corresponding sentence representations u and v. Next, by feeding u, v and $|u - v|$ into a 9-class softmax layer, i.e.,

---

[2] https://stanfordnlp.github.io/CoreNLP/.

[3] We only consider connectives that are less ambiguous in relations. See Appendix for the full list of connectives.

	CT	RS	CJ	EQ	GN	IN	CA	PR	DJ
Pairs	8M	430K	800K	29K	12K	285K	188K	26K	26K

Table 1: Statistics of sentence pairs extracted from four text corpora. 9 discourse relations are Contrast (CT), Result (RS), Conjunction (CJ), Equivalence (EQ), Generalization (GN), Instantiation (IN), Chosen Alternative (CA), Precedence (PR), and Disjunctive (DJ).

$p = \mathrm{softmax}(W_t[u, v, |u - v|])$, our model predicts what is the discourse relation (e.g., Contrast, Result, Equivalence, etc.) between them.

4 Estimating Sentence Semantics Distribution via Perturbation

To estimate the semantic distribution of the input sentence $S = \{w_i\}_{i=1}^{N}$, we mask each word w_i individually to construct a new sentence and apply the sentence embedding model (Section 3.2) to calculate the new representation. Next, we calculate the cosine distance between the new and the original representation. The distance can quantitatively tell us how much the new sentence (after masking one word) semantically drift from the original sentence, which indicates the contribution of that word to the sentence semantics.[4] Finally, we normalize the distances by their summation so that the importance scores of all words sum to 1.

5 Evaluation

5.1 Evaluation Metric

Ideally, if we know the gold standard importance distribution in sentence semantics, we can directly compare a model's prediction with the gold standard. However, it is infeasible for humans to annotate/assign a continuous importance score to each word in a sentence. To address the lack of evaluation data, we adopt two sentence compression datasets and one summarization dataset for evaluation. The main idea is that the compressed sentence, as a shorter version of the original sentence, specifies the most important words in the original sentence. Therefore, we can score a model by its ability of concentrating on important words.

Furthermore, we propose a new evaluation metric — **Importance Information Gain** — to score how much a predicted distribution is superior to a uniform distribution (all words are equally important). Specifically, given a sentence $S = \{w_i\}_{i=1}^{N}$ consisting of $|S| = N$ words, suppose the compressed sentence by human is $S' = \{w_j\}_{j \in 1\sim N}$, where S' is a subset of S. Let $[v_1, v_2, \cdots, v_N]$ be the importance scores over words (normalized such that $\sum_{i=1}^{N} v_i = 1$). We calculate the information gain g to be the average importance score on importance words $\frac{\sum_{i\in S'} v_i}{|S'|}$ over the score that every word is equally important $\frac{1}{|S|}$: $g = \frac{\sum_{i\in S'} v_i}{|S'|} - \frac{1}{|S|}$.

We next normalize g for each test sentence instance by the upper bound g^*, where g^* is defined as the information gain of a model that can perfectly distinguish important words from non-important words (concentrate all information on only important words): $g^* = \frac{1}{|S'|} - \frac{1}{|S|}$. Then, the final Importance Information Gain = $g/g^* \in [0, 1]$.

To validate whether the proposed metric is able to measure the quality of a semantic distribution, we randomly sample 100 pairs from baseline models' predictions in Table 2 and analyze our metric's consistency with human preferences. Specifically, each sampled pair consists of predictions (semantic distributions) of two separate baseline models on the same sentence. Three expert annotators are asked to judge which one better characterizes the importance/contribution of each word to the sentence meaning. We next compare our metric's preferences with the gold human preferences[5]. Our metric agreed on 87/100 with human gold labels, achieving a substantial (Cohen, 1968) kappa agreement score of 0.74.

5.2 Evaluation Datasets

For evaluation, we experiment on three datasets. Google sentence compression dataset (GGL) (Filippova and Altun, 2013) contains 10K test sentence compression pairs and BNC written dataset (Clarke and Lapata, 2008) contains 1.5K test compression pairs. Additionally, we go through the Gigaword summarization dataset (GGW) and compare the first sentence with the title sentence in the news article. We select (first sentence, title) pairs as our testing data if words[6] in the title sentence is a subset of words in the first sentence. We collect the first 10K sentence pairs as our test data.

---

[4] If the target masked word is tokenized into multiple subword pieces by the tokenizer, we mask each subword piece in turn and calculate the summation score as the word importance.

[5] Gold labels are generated based on majority voting. The average Cohen's kappa inter-agreement between three annotators is 0.67.

[6] We use lemma matching in practice.

5.3 Baseline Systems

Rule-Based Model. We use a rule-based primary semantics extraction system (Zhang et al., 2020), which first parses the input sentence into a dependency parsing tree and extracts the sentence skeleton.[7] Words in the sentence skeleton are considered equally important.

GloVe Embedding. We calculate the sentence embeddings by averaging GloVe embeddings (Pennington et al., 2014) among all words except w_i.

Original BERT. We apply the original BERT-base model[8] that uses next sentence prediction as the training objective.

BERT-CT (Carlsson et al., 2021) introduces an unsupervised learning method called Contrastive Tension (CT) that only requires raw sentences and achieves SOTA performance on Semantic Textual Similarity (STS) tasks. CT tries to maximize the dot product between sentence representations for identical sentences and minimize the dot product for differing sentences. BERT-CT-STSb and BERT-CT-NLI[9] are two supervised CT models fine-tuned on STSb and NLI, respectively.

SimCSE (Gao et al., 2021) is a contrastive sentence embedding framework that achieves SOTA performance on sentence similarity tasks. The unsupervised model (SimCSE-unsup) takes an input sentence and predicts itself in a contrastive objective with dropout as the data augmentation method. The supervised model (SimCSE-sup) incorporates sentence pairs from the NLI data (Williams et al., 2018; Bowman et al., 2015) into a contrastive learning framework, by using *entailment* pairs as positives and *contradiction* pairs as negatives.

DiffCSE (Chuang et al., 2022) augments SimCSE contrastive learning with the edited sentences sampled from a masked language model.

SBERT (Reimers and Gurevych, 2019). Sentence-BERT (SBERT) uses siamese and triplet network structures to derive sentence embeddings. SBERT-NLI is the original model introduced by Reimers and Gurevych (2019) that is fine-tuned on NLI. Moreover, we also consider four more recent SBERT models that are fine-tuned with more sentence pair data.[10] Specifically, SBERT-MSMarco is fine-tuned on the MSMarco Passage Ranking Dataset containing 500K (query, relevant passage) pairs from Bing search. SBERT-Paraphrase is fine-tuned on NLI and 11 paraphrase datasets. SBERT-MultiQA is fine-tuned on 214M (question, answer) pairs from 17 QA datasets. MPNet-All is fine-tuned on all available sentence pair tasks including 32 tasks with 1,170M sentence pairs.

	Models	GGL	BNC	GGW	Avg.
	Supervised				
1	BERT-CT-STSb	7.4	19.3	8.3	11.7
2	BERT-CT-NLI	7.5	10.0	6.5	8.0
3	SimCSE-sup	15.8	25.1	10.2	17.0
4	SBERT-NLI	14.3	23.2	14.2	17.2
5	SBERT-Paraphrase	23.7	15.4	31.8	23.6
6	SBERT-MultiQA	33.0	28.6	35.6	32.4
7	SBERT-MSMarco	31.7	28.3	32.7	30.9
8	MPNet-All	**52.6**	**33.3**	**58.2**	**48.0**
	Unsupervised				
9	Parsing Tree	18.7	19.8	7.2	15.2
10	GloVe Embedding	10.2	25.3	12.1	15.9
11	Original BERT	2.0	0.3	0.8	1.0
12	BERT-CT	12.0	16.3	12.4	13.5
13	SimCSE-unsup	13.9	14.9	10.1	13.0
14	DiffCSE	21.6	**28.7**	20.8	23.7
15	Our Model-disc.	**36.9**	27.2	**36.6**	**33.6**

Table 2: Importance Information Gain (%) on three evaluation datasets. We group models based on whether it requires human-labeled data in training or not.

Table 2 shows the results of all models. We group models based on whether they use human-labeled data or not. Line 15 is our model trained on 9.8M sentence pairs that are extracted by discourse connectives. Our unsupervised discourse recognition model achieves the highest overall information gain among all unsupervised approaches, which is even higher than several supervised models that are trained using millions of human-annotated data (Lines 3-7). Surprisingly, even SimCSE and DiffCSE report better performance than MPNet-All on STS tasks, they are much worse than MPNet-All on capturing sentence primary semantics.

5.4 Visualization and Analysis

While previous studies (Conneau et al., 2017b) empirically show that fine-tuning towards NLI data yields good sentence embedding models, we visualize and compare the model trained on the NLI data with the model trained on discourse pairs. We find the NLI data model sometimes is biased to specific numbers, time expressions, etc (Figure 2). It may be mainly due to the bias introduced during the NLI data construction process when human annotators sometimes just replace the numbers or time

---

[7] We select the whole sentence as the prediction when the system fails on complex sentences.

[8] Select the output at the `[CLS]` token.

[9] `https://github.com/FreddeFrallan/Contrastive-Tension`.

[10] `https://www.sbert.net/docs/pretrained_models.html`.

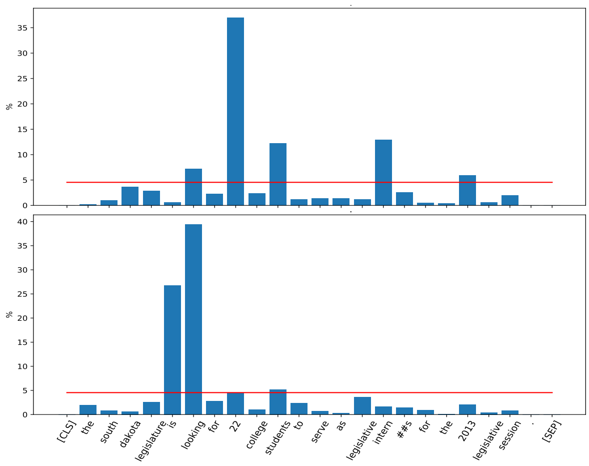


Figure 2: Semantic distribution of the model trained on NLI (upper) v.s. discourse data (lower) on the same sentence (important words are underlined). *"The South Dakota legislature is looking for 22 college students to serve as legislative interns for the 2013 legislative session."* The red line indicates $1/|S|$.

expressions to produce a contradictory sentence. In contrast, the discourse relation recognizer is less sensitive to those specific expressions. Appendix II contains more examples.

6 Conclusion

In this paper, we have introduced a perturbation method for estimating sentence semantic distribution and designed a new metric to achieve automatic evaluation. We find discourse relation recognition can serve as a general inference task to train an unsupervised sentence embedding model that estimates such distribution meaningfully than several SOTA sentence embedding models.

7 Limitations

To benefit from large-scale training data, we train the discourse relation recognizer on data that are automatically generated by matching explicit connectives. Even we only select explicit connectives that have one dominant discourse relation, some of them may still reflect a different relation in some contexts. For example, connective *in other words* indicates a "Equivalence" relation most of the time, but sometimes it can also indicate a "Generalization" relation. In this regard, our method shares the same limitations as the broad class of weakly supervised methods where training data are automatically generated. Considering discourse parser is not the main focus of this paper, we leave how to generate cleaner discourse relation data and train a better discourse relation recognizer for future work.

References

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. 2022. DiffCSE: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218, Seattle, United States. Association for Computational Linguistics.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

Jacob Cohen. 1968. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017a. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017b. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Ayan Das and Sudeshna Sarkar. 2019. A little perturbation makes a difference: Treebank augmentation by perturbation improves transfer parsing. In *Proceedings of the 16th International Conference on Natural*

Language Processing, pages 75–84, International Institute of Information Technology, Hyderabad, India. NLP Association of India.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Łukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1481–1491, Seattle, Washington, USA. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hidetaka Kamigaito and Manabu Okumura. 2020. Syntactically look-ahead attention network for sentence compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8050–8057.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and William B Dolan. 2021. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5053–5069.

Sebastian Löbner. 2013. *Understanding semantics*. Routledge.

Alfonso Mendes, Shashi Narayan, Sebastião Miranda, Zita Marinho, André FT Martins, and Shay B Cohen. 2019. Jointly extracting and compressing documents with summary state representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3955–3966.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 95–100. Association for Computational Linguistics.

Allen Nie, Erin Bennett, and Noah Goodman. 2019. Dissent: Learning sentence representations from explicit discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4497–4510.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8538–8544.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Christopher Gerard Potts. 2003. *The logic of conventional implicatures*. University of California, Santa Cruz.

Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation sensitivity analysis to detect unintended model biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5740–5745, Hong Kong, China. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. 2006. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205.

Judith Tonhauser, David Beaver, Craige Roberts, and Mandy Simons. 2013. Toward a taxonomy of projective content. *Language*, pages 66–109.

Noortje J Venhuizen, Johan Bos, Petra Hendriks, and Harm Brouwer. 2018. Discourse Semantics with Information Structure. *Journal of Semantics*, 35(1):127–169.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3292–3303.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. ASER: A large-scale eventuality knowledge graph. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 201–211. ACM / IW3C2.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Appendix I

Experiment details. We generally follow the hyper-parameter setting of previous work to train our model. In experiments, we train our model using cross-entropy loss and Adam (Kingma and Ba, 2015) optimizer with initial learning rate 5e-5, dropout rate 0.5, and batch size 256 for 5 training epochs. The training of the discourse relation recognizer was run on one machine with 8 NVIDIA P40 GPUs, taking about 6 hours per epoch and three epochs in total. We simply use the standard hyper-parameters to train our model without any hyper-parameter search.

Appendix II

Here is the full list of explicit discourse connectives for extracting sentence pairs (Section 3.1).

Contrast (CT): although, but, by comparison, by contrast, however, in contrast, nevertheless, nonetheless, on the contrary, on the other hand, though
Result (RS): accordingly, as a result, because, consequently, hence, therefore, thus
Conjunction (CJ): additionally, besides, furthermore, in addition, in fact, indeed, moreover, overall, similarly
Equivalence (EQ): in other words
Generalization (GN): in short, in sum
Instantiation (IN): for example, for instance
Chosen Alternative (CA): instead
Precedence (PR): ultimately
Disjunctive (DJ): otherwise, unless

Appendix III

More examples of semantic distribution of the model trained on NLI (upper) v.s. general inference knowledge (lower) on the same sentence.

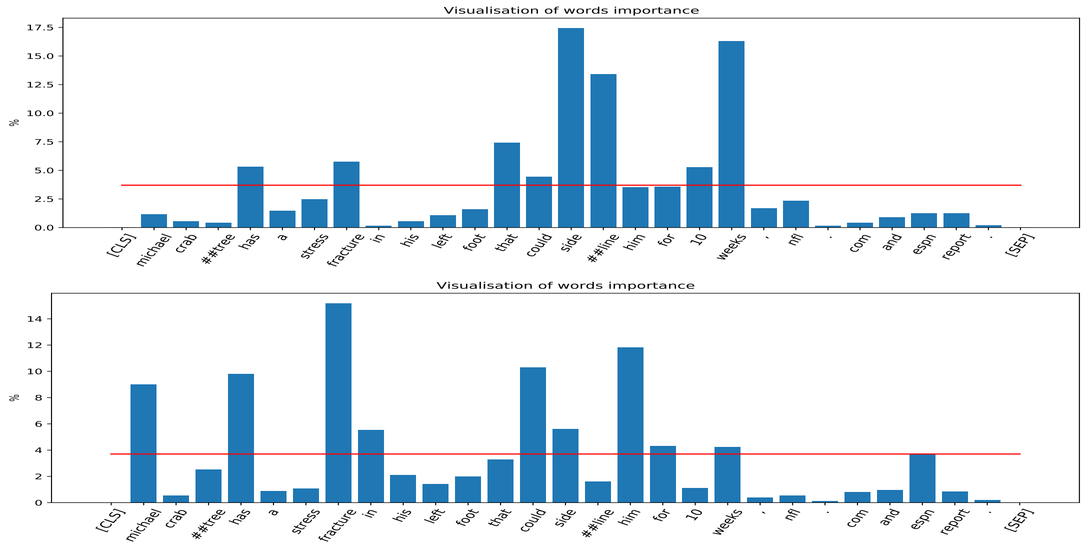


Figure 3: *Michael Crabtree has a stress fracture in his left foot that could sideline him for 10 weeks , NFL.com and ESPN report .*

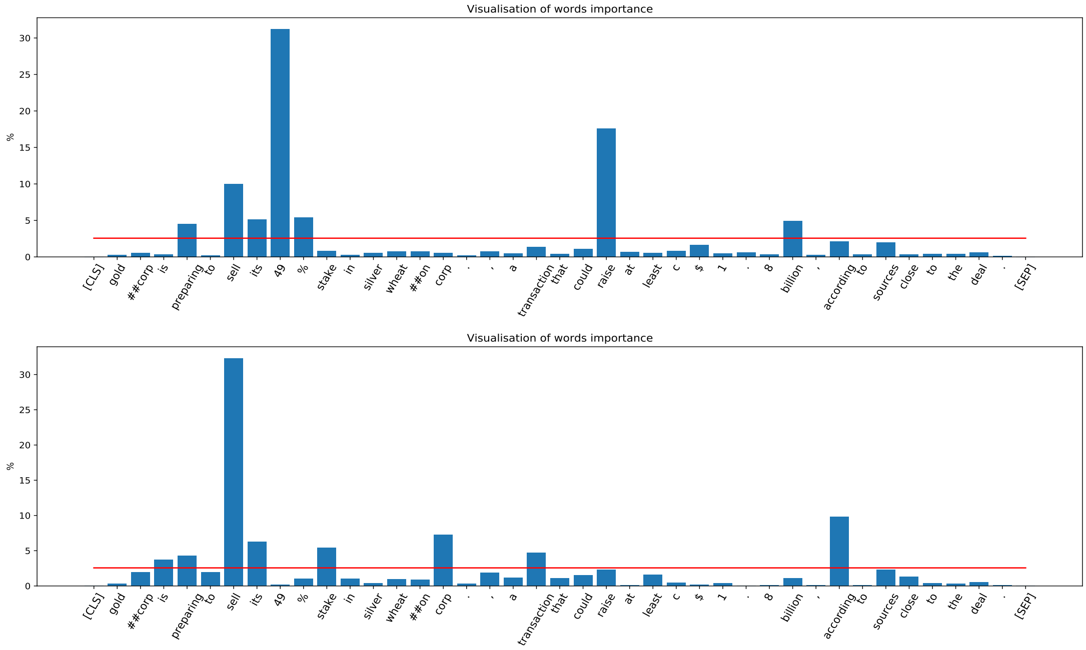


Figure 4: *Goldcorp is preparing to sell its 49 % stake in Silver Wheaton Corp. , a transaction that could raise at least $ 1.8 billion , according to sources close to the deal .*

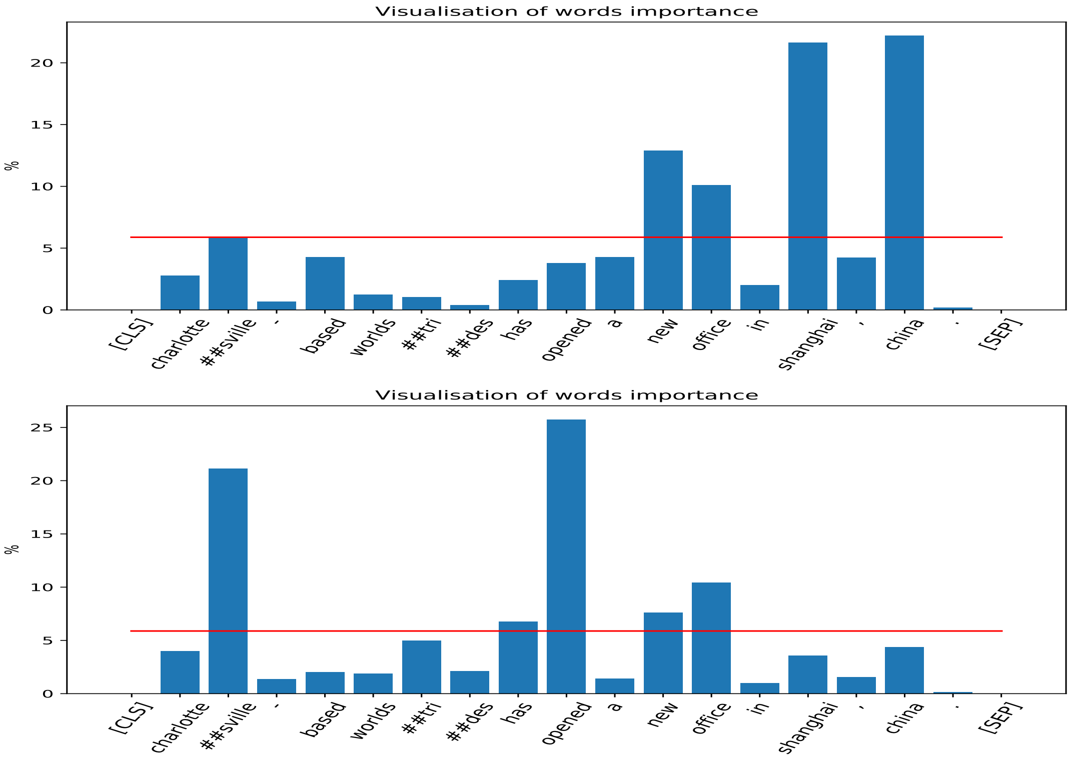


Figure 5: *Charlottesville - based WorldStrides has opened a new office in Shanghai , China .*

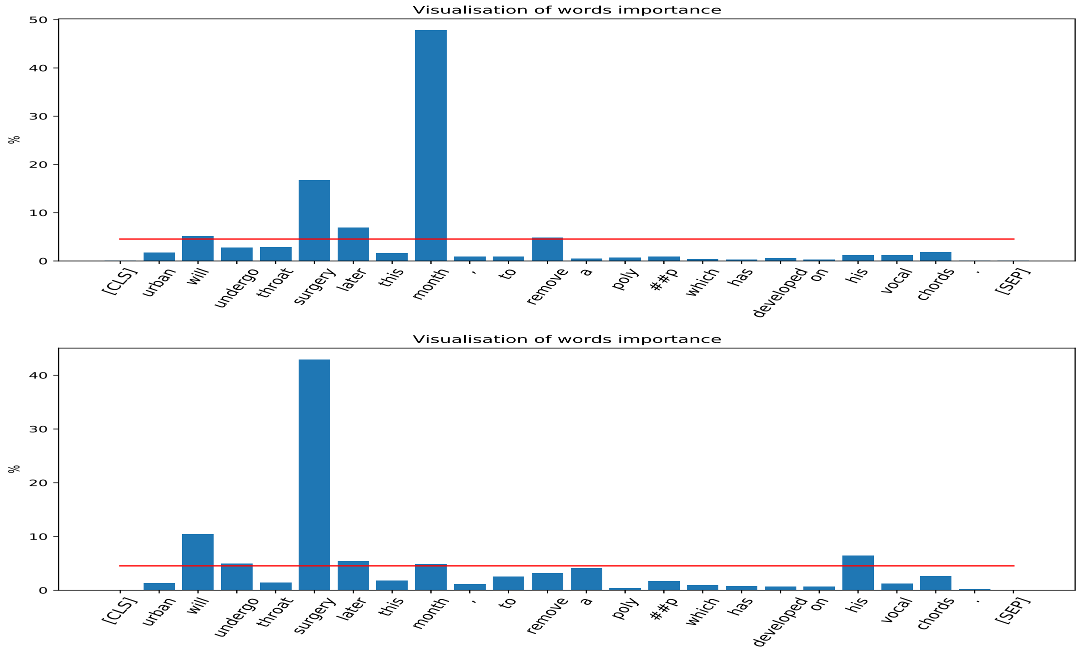


Figure 6: *Urban will undergo throat surgery later this month , to remove a polyp which has developed on his vocal chords .*

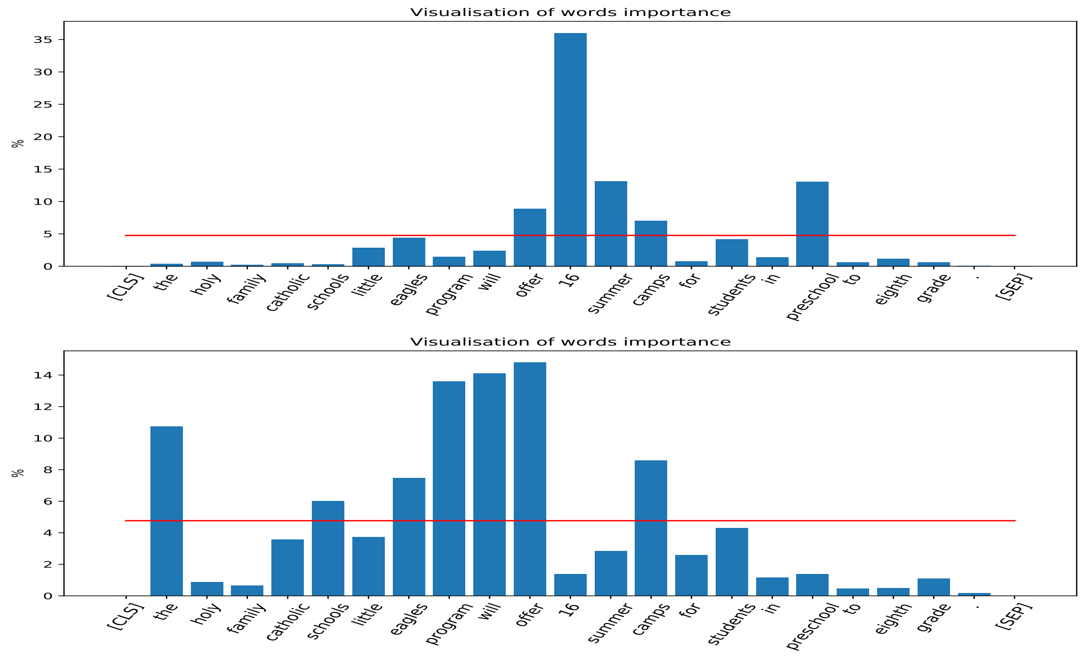


Figure 7: *The Holy Family Catholic Schools Little Eagles program will offer 16 summer camps for students in preschool to eighth grade .*

AutoTriggER: Label-Efficient and Robust Named Entity Recognition with Auxiliary Trigger Extraction

Dong-Ho Lee[1,4]*, **Ravi Kiran Selvam**[1]*, **Sheikh Muhammad Sarwar**[3]†, **Bill Yuchen Lin**[1],
Fred Morstatter[4], **Jay Pujara**[4], **Elizabeth Boschee**[4], **James Allan**[2], **Xiang Ren**[1,4]
[1]Department of Computer Science, University of Southern California
[2]CIIR, Manning CICS, University of Massachusetts Amherst, [3]Amazon.com
[4]Information Science Institute, University of Southern California
`{dongho.lee,rselvam,yuchen.lin,xiangren}@usc.edu`,
`allan@cs.umass.edu`, `smsarwar@amazon.com`, `{fredmors,jpujara,boschee}@isi.edu`

Abstract

Deep neural models for named entity recognition (NER) have shown impressive results in overcoming label scarcity and generalizing to unseen entities by leveraging distant supervision and auxiliary information such as explanations. However, the costs of acquiring such additional information are generally prohibitive. In this paper, we present a novel two-stage framework (AUTOTRIGGER) to improve NER performance by automatically generating and leveraging "*entity triggers*" which are human-readable cues in the text that help guide the model to make better decisions. Our framework leverages post-hoc explanation to generate rationales and strengthens a model's prior knowledge using an embedding interpolation technique. This approach allows models to exploit triggers to infer entity boundaries and types instead of solely memorizing the entity words themselves. Through experiments on three well-studied NER datasets, AUTOTRIGGER shows strong label-efficiency, is capable of generalizing to unseen entities, and outperforms the RoBERTa-CRF baseline by nearly 0.5 F1 points on average.

1 Introduction

Named Entity Recognition (NER) serves as a key building block in information extraction systems. Recent advances in deep neural models for NER have yielded state-of-the-art performance when sufficient human annotations are available (Lample et al., 2016; Liu et al., 2018; Peters et al., 2017; Ma and Hovy, 2016). However, such success cannot easily transfer to practitioners developing NER systems in specific domains (*e.g.*, biomedical papers, financial reports, legal documents), where domain-expert annotations are expensive and slow to obtain. Moreover, NER systems face challenges in real-world applications where novel entities, unseen in training data, are often encountered (Lin et al., 2021). Recent attempts addressing label scarcity and improving generalization to unseen entities have explored various types of human-curated resources as auxiliary supervision, such as entity dictionaries (Peng et al., 2019; Shang et al., 2018; Yang et al., 2018; Liu et al., 2019a), labeling rules (Safranchik et al., 2020; Jiang et al., 2020), prompts (Ding et al., 2021; Cui et al., 2021), demonstrations (Lee et al., 2022), retrieved context (Wang et al., 2021), and labeling explanations (Hancock et al., 2018; Wang et al., 2020; Ye et al., 2020; Lin et al., 2020; Lee et al., 2020a). In particular, human-provided labeling explanations as auxiliary supervision signals are cost-effective compared to collecting label-only annotations for larger number of instances. For the NER task, Lin et al. (2020) introduced the concept of an *entity trigger*, an effective way to represent explanations for the labeling decisions (See Figure 1 (a) vs. (b)).

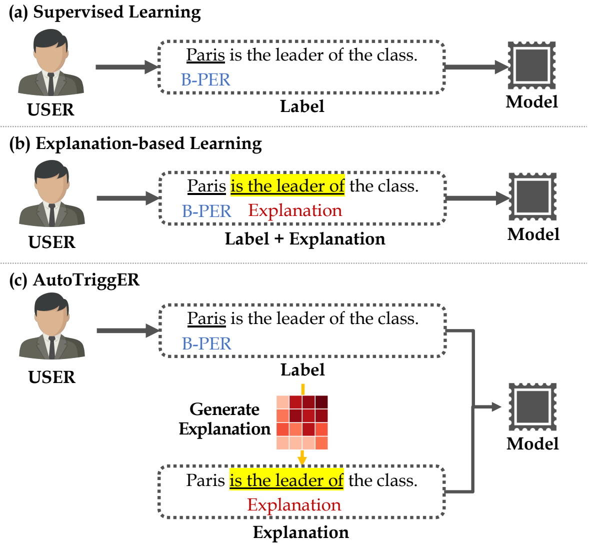


Figure 1: Existing explanation-based learning frameworks mostly rely on humans provided labeling explanations while our framework automatically generates and leverages explanations to NER.

However, prior works have the following limita-

---

*The first two authors contributed equally.
†work done before joining Amazon.com.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3011–3025
May 2-6, 2023 ©2023 Association for Computational Linguistics

tions: (1) *Expense of collecting rationales*: Prior works primarily use a limited number of *crowd-soured* explanations (e.g., *entity trigger*) for improving data (label) efficiency of model training. While such human-curated auxiliary supervision is of high quality, the crowd-sourcing procedure can be very expensive and time-consuming. This largely limits the scale and domains of the collected rationales; (2) *Unseen entity generalization*: The biggest advantage of leveraging such explanations for NER is a generalization ability towards unseen entities. Reinforcing this prior knowledge in the model rather than memorizing the entity words themselves can make the model robust against unseen entities during training. However, prior works do not evaluate such generalization ability.

To address these limitations, we propose a two-stage NER framework, named AUTOTRIGGER , which improves label efficiency and model robustness without human efforts (see Figure 1 (c)). First, it *automatically* generates entity triggers as explainable features to reduce human effort on collecting rationales. It then strengthens the prior knowledge in the model by interpolating model embeddings of entity-masked sentence which force the model to rely more on triggers and trigger-masked sentence which force the model to rely more on the entity.

The *first* stage of our framework (Sec. 3.1) aims to *automatically extract entity triggers* using a post-hoc explanation. We propose to exploit the syntactic features of a sentence for assigning importance scores to a group of input tokens such that we can extract entity triggers as auxiliary supervision. Specifically, we form a collection of trigger candidates for an entity in a sentence using the phrases extracted from its *constituency parsing tree* (Joshi et al., 2018). Then we score each trigger candidate based on its ability to predict the target entity, while varying the surrounding context to ensure trigger robustness.

The *second* stage (Sec. 3.2) focuses on how to use triggers, which are useful contextual clues, in making the prediction. We propose *Trigger Interpolation Network* (TIN), a novel architecture that effectively uses trigger-labeled NER data to train a model. Here, we employ two separate masking passes when learning our model's embeddings: one masking the entity words (forcing the model to rely more on the triggers) and one masking the triggers (forcing the model to rely more on the entity words). We then interpolate the embeddings of both entity-masked and trigger-masked sentences to learn a mixed sentence representation, which is provided as input to a standard sequence labeling task. In this manner, the TIN can effectively learn to focus on useful contextual clues to infer entity boundaries and types with contextualized embeddings from pre-trained language models such as BERT (Devlin et al., 2019) while it may not miss the entities whose type can be determined by itself. (e.g., inputs that contain only the entities).

Extensive experimental results on several domains show that AUTOTRIGGER framework consistently outperforms baseline methods by 0.5 F1 points on average in fully supervised setting. Our work shows strong performance especially in label scarcity setting and unseen entity generalization. In the label scarcity setting ranging from 50 to 200 number of train instances, assuming a task that needs annotation from scratch, our model gains more than 3-4 F1 score on average. Also, for the filtered test set where the entities did not appear in the train and development sets, assuming a real-world applications in which entities can be out of the distribution of the train data, our model gains more than 2-3 F1 score on average.

2 Problem Formulation

Named Entity Recognition. We let $\mathbf{x} = [x^{(1)}, x^{(2)}, \ldots x^{(n)}]$ denote the sentence consisting of a sequence of n words and $\mathbf{y} = [y^{(1)}, y^{(2)}, \ldots y^{(n)}]$ denote the NER-tag sequence. The task is to predict the entity tag $y^{(i)} \in \mathcal{Y}$ for each word $x^{(i)}$, where $\mathcal{Y}$ is a pre-defined set of tags such as $\{$B-PER, I-PER, $\ldots$, O$\}$. We let $\mathcal{D}_L$ denote the labeled dataset consisting of the set of instances $\{(\mathbf{x_i}, \mathbf{y_i})\}$, where $\mathbf{x_i}$ is the i-th input sentence and $\mathbf{y_i}$ is its output tag sequence. Here, we use BIOES scheme (Chiu and Nichols, 2016).

Entity Trigger Extending from extractive rationales for text classification (DeYoung et al., 2020; Jain et al., 2020), an *entity trigger* is a form of extractive explanatory annotation for NER, defined as *a group of words that help explain the recognition process of an entity mentioned in the sentence* (Lin et al., 2020). For example, in Figure 2, "*had ... dinner at*" and "*where the food*" are two distinct triggers associated with the RESTAURANT entity "*Sunnongdan*". Formally, given a labeled NER instance $(\mathbf{x}, \mathbf{y})$, we use T to denote the set of entity triggers for this instance. Each trigger $t_i \in T$ is

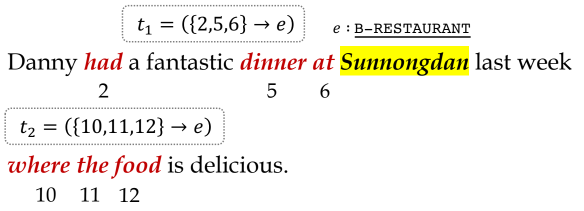


Figure 2: **Example of entity trigger.** Entity trigger t_i is a cue phrase toward the entity e in the sentence, which is represented by a set of corresponding word indices. Both entity triggers (t_1,t_2) are associated to the same entity e ("*Sunnongdan*") typed as restaurant.

associated with an entity e and a set of word indices $\{w_i\}$. That is, $t = (\{w_1, w_2, \dots\} \to e)$ represents an entity trigger. For example, in Figure 2, $t_1 = \{2, 5, 6\}$ is an entity trigger identified for entity "*Sunnongdan*" in the sentence. A *trigger-labeled NER dataset*, $\mathcal{D}_T = \{(\mathbf{x_i}, \mathbf{y_i}, T(\mathbf{x_i}, \mathbf{y_i}))\}$, consists of examples in a labeled NER dataset $\mathcal{D}_L$ with their associated entity triggers.

Problem Definition We focus on the problem of how to *automatically* extract entity triggers to create a trigger-labeled dataset $\mathcal{D}_T$ from $\mathcal{D}_L$ without manual effort and to cost-effectively train a NER model using $\mathcal{D}_T$. Here, we aim to achieve better performance and robustness over existing NER models using the same amount of training instances. Moreover, to check the quality of the automatically extracted triggers, we compare with the model using human-labeled triggers.

3 Approach

AUTOTRIGGER is a two-stage architecture that begins with an *automatic trigger extraction* stage followed by a *trigger interpolation network* (`TIN`). It automatically extracts and scores the importance of entity trigger phrases in the first stage (Sec. 3.1) and uses them in the later stage to train the NER model (Sec. 3.2).

3.1 Automatic Trigger Extraction

Automatic trigger extraction is the first stage of our AUTOTRIGGER framework. Different from prior works on extracting rationales and explanations for sentence classification (Ribeiro et al., 2016; Li et al., 2016b), we look to extract triggers for explaining the detection (occurrence) of an entity in the sentence. In our study, we extend the *sampling and occlusion* (`SOC`) algorithm (Jin et al., 2020), a feature attribution algorithm for post-hoc model explanation, to extract entity triggers. Here, we exploit a token classifier to model the score function of the target entity in the sentence and limit the search space to a set of phrases from the constituent parse tree. Specifically, given a labeled NER corpus $(\mathbf{x_i}, \mathbf{y_i}) \in \mathcal{D}_L$, we consider four main steps for entity trigger extraction: 1) generating a set of candidate phrases $\mathcal{P}$, 2) constructing entity token classifier $\mathcal{M}_t$, 3) trigger phrase scoring, and 4) trigger phase selection.

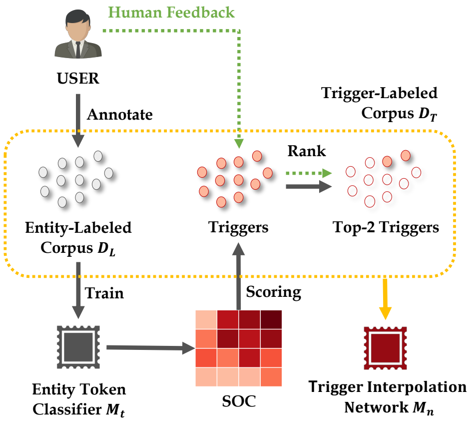


Figure 3: **Overview of the proposed AUTOTRIGGER.** It trains an entity-token classifier $\mathcal{M}_t$ with entity-labeled corpus $\mathcal{D}_L$ and uses the sampling-and-occlusion (SOC) algorithm to extract triggers. There is a provision for leveraging human feedback in the framework for refining automatically generated triggers. Trigger Interpolation Network (TIN) learns the NER model from the trigger-labeled corpus.

Candidate Generation Given a labeled sentence $(\mathbf{x_i}, \mathbf{y_i})$, we obtain its constituency parse tree and consider the set of phrase nodes $\mathcal{P}$ from the tree as trigger candidates (see Figure. 4 for an illustration with examples). `SOC` aims to compute context-independent phrase-level importance for sequence classification tasks such as sentiment analysis and relation extraction (Jin et al., 2020). It uses agglomerative clustering (Singh et al., 2019) to effectively compute the phrase-level importance without evaluating all the possible phrases in the sentence. Here, we provide pre-defined hierarchy for limiting the search space to a set of phrases from the constituent parse tree. Specifically, given a sentence instance $\mathbf{x_i} \in \mathcal{D}_L$ and a target entity mentioned in this sentence $e \in \mathbf{x_i}$, we generate a set of phrase candidate $\mathcal{P} = \{\mathbf{p_i}\}$, where $\mathbf{p_i} = (w_s, w_e)$. Here (w_s, w_e) denote the start and end index of the phrase span $\mathbf{p_i}$. To generate $\mathcal{P}$, we parse the input sentence $\mathbf{x_i}$ using an off-the-shelf constituency parser [1] and

---

[1] https://stanfordnlp.github.io/CoreNLP/parse.html

collect phrase nodes in the parse tree as the set of candidate phrases $\mathcal{P}$. To avoid including target entity as an entity trigger, we discard phrases that contain the target entity, i.e., $\{\mathbf{p_j} | e \in \mathbf{p_j}\}$.

Entity Token Classifier Construction In order to apply SOC to the sequence labeling task, we propose a way to derive the importance score of the phrase $\mathbf{p}$ specific to the input sentence x and target entity e. To achieve this, we use a token classification model $\mathcal{M}_t$ to generate prediction scores in the label space (e.g., probability for target class of a token). Given an input sentence $\mathbf{x_i} = [x_i^{(1)}, x_i^{(2)}, \dots x_i^{(n)}]$, $\mathcal{M}_t$ classifies each token $x_i^{(j)}$ to the named entity tag $y_i^{(j)} \in \mathcal{Y}$ where $\mathcal{Y}$ is a predefined set of named entity tags such as B-PER, I-PER and O. We learn $\mathcal{M}_t$ with labeled corpus $\mathcal{D}_L$ to fit a probability distribution $\mathbb{P}(\mathbf{y}|\mathbf{x})$. Then, we can derive the prediction score function s towards the target entity e which is computed as the average conditional probability over tokens of the target entity $x^{(j)} \in e$ as follows:

$$s(\mathbf{x}, e) = \frac{1}{|e|} \sum_{x^{(j)} \in e} \mathbb{P}(\mathbf{y^{(j)}} | x^{(j)}). \tag{1}$$

Phrase Scoring. Once we have the set of candidate phrases $\mathcal{P}$ for the given input instance (x_i, y_i), we aim to measure the score of each of the candidate phrase $\mathbf{p}$ w.r.t the target entity e. We use the score function s of the $\mathcal{M}_t$ to measure the importance score of each phrase $\mathbf{p}$. Here, we have two steps: (1) *input occlusion*, (2) *context sampling*.

Input occlusion (Li et al., 2016b) computes the importance of $\mathbf{p}$ specific to the entity e in the input $\mathbf{x}$ by measuring the prediction difference caused by replacing the phrase $\mathbf{p}$ with padding tokens $0_\mathbf{p}$. We use $s\left(\mathbf{x_{-p}}, e; \mathbf{0_p}\right)$ to denote the model prediction score after replacing the masked-out context $\mathbf{x_{-p}}$ with padding tokens $0_\mathbf{p}$:

$$\phi(\mathbf{p}, \mathbf{x}, e) = s(\mathbf{x}, e) - s\left(\mathbf{x_{-p}}, e; \mathbf{0_p}\right). \tag{2}$$

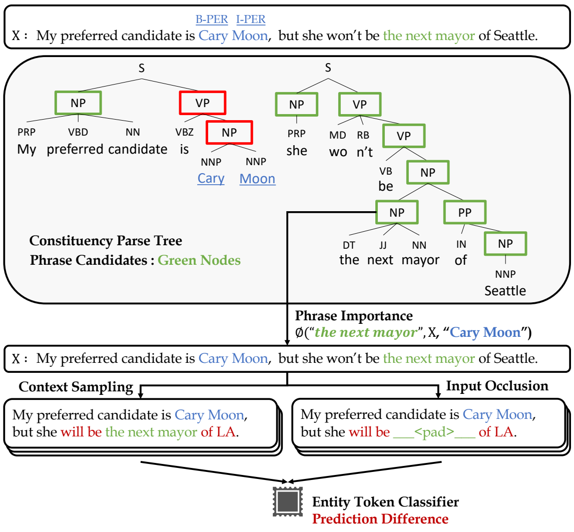


Figure 4: **Overview of the Sampling and Occlusion (SOC)**. It creates a set of phrase candidates with phrase nodes of the constituency parse tree, and then computes the phrase importance by average prediction difference between context sampled sentences and its phrase-masked sentences. Note that the entity mention "*Cary Moon*" is not included as candidate.

However, the importance score $\phi(\mathbf{p}, \mathbf{x}, e)$ from equation 2 has a challenge that it is difficult to model the direct impact of $\mathbf{p}$ towards e since the score is dependent on the context words around $\mathbf{p}$. For example, in Figure. 4, $\mathbf{p}$ ("*the next mayor*") is replaced by pad tokens to compute its importance towards the entity e ("*Cary Moon*"). However, the importance score of $\mathbf{p}$ is dependent on context words around $\mathbf{p}$ ("*won't be ... of Seattle*"). To compute the context independent importance score, SOC proposes *context sampling* that samples the context words around the phrase p and computes the average prediction differences over the samples. Specifically, it samples the context words $\hat{x}_\delta$ from a language model $p(\hat{x}_\delta | x_{-\delta})$ that is trained on $\mathcal{D}_L$, and obtains a set of context word replacements $\mathcal{S}$. Here, we use 20 replacements. For each replacement $\hat{x}_\delta \in \mathcal{S}$, we measure the prediction difference caused by replacing the phrase $\mathbf{p}$ with padding tokens. We take the average of these prediction differences to be the context-independent score $\phi(\mathbf{p}, \mathbf{x}, e)$ of the phrase $\mathbf{p}$, as expressed in equation 3:

$$\frac{1}{|\mathcal{S}|} \sum_{\hat{\mathbf{x}}_\delta \in \mathcal{S}} \left[s\left(\mathbf{x}_{-\delta}, e; \hat{\mathbf{x}}_\delta\right) - s\left(\mathbf{x}_{-\{\delta, \mathbf{p}\}}, e; \hat{\mathbf{x}}_\delta; \mathbf{0_p}\right) \right] \tag{3}$$

In Figure. 4, context words "*won't be*" and "*of Seattle*" around the phrase "*the next mayor*" are replaced into "*will be*" and "*of LA*" which are sampled from the language model. Then, the classifier computes the prediction difference between the sampled sentences with and without the phrase.

Phrase Selection The importance score $\phi(\mathbf{p}, \mathbf{x}, e)$ for phrase $\mathbf{p}$ is the degree to which $\mathbf{p}$ determines the correct entity type of target entity e. After obtaining $\phi(\mathbf{p}, \mathbf{x}, e)$ for all phrase candidates $\mathcal{P} = \{\mathbf{p_i}\}$, we pick the top k candidate phrases with the highest importance score as the entity triggers, where k is a hyperparameter. Specifically,

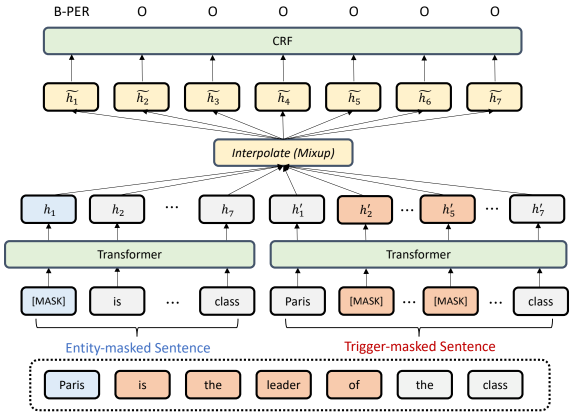


Figure 5: **Overview of the Trigger Interpolation Network (`TIN`).** Given an input sentence we create an Entity-masked sentence and a Trigger-masked Sentence. Then we interpolate token level representations h_i and h'_i to create new hidden state representation, $\tilde{h}_i$. Interpolated hidden representations are fed to a CRF.

for each input instance $(\mathbf{x_i}, \mathbf{y_i}) \in \mathcal{D}_L$, we pick the top k candidate phrases as entity triggers $T(\mathbf{x_i}, \mathbf{y_i})$ per each entity $e \in \mathbf{x_i}$ to create a form $(\mathbf{x_i}, \mathbf{y_i}, T(\mathbf{x_i}, \mathbf{y_i})) \in \mathcal{D}_T$. Here, we search for the best k out of [1, 2, 3, 5, 7, 10] (See Figure 12).

3.2 Trigger Interpolation Network (TIN)

The second stage of AUTOTRIGGER is the trigger interpolation network (`TIN`), which we define as a neural network that learns from a trigger-labeled dataset $\mathcal{D}_T$ consisting of a set of instances of the form $\{(\mathbf{x}, \mathbf{y}, T(\mathbf{x}, \mathbf{y}))\}$. The main idea behind the model is to combine two representations obtained from the sentence with target entity tokens and automatically extracted entity triggers. In one representation, the entity tokens are masked and we hypothesize that it forces the model to infer the entity boundaries and types by leveraging the contextual clues. However, some of the entity tokens itself may already have enough information to infer the entity type without any contextual clues. Thus, in another representation, the entity tokens become the predominant source of information for classifying the entity as we mask out the triggers.

TIN encodes the input sequence $\mathbf{x}$ with a transformer encoder $\mathbf{F}(.;\theta)$ and feeds the hidden representations $\mathbf{h}$ to a CRF tagger. Our proposal is to create two different representations of a token in a sequence and interpolate (mixup) them. By doing this, we expect the model to predict less confidently on interpolations of hidden representations, which eventually make model be robust to both entity-perturbed and trigger-perturbed sentences.

As shown in Figure 5, we first create entity-masked sentence $\mathbf{x}_{-e}$ and trigger-masked sentence $\mathbf{x}_{-t}$ for a given input instance $\{(\mathbf{x}, \mathbf{y}, T(\mathbf{x}, \mathbf{y}))\}$, and then compute the interpolations in the output space of transformer encoder $\mathbf{F}(.;\theta)$ as follows:

$$\begin{aligned} \mathbf{h} &= \mathbf{F}(\mathbf{x}_{-e};\theta), \mathbf{h}' = \mathbf{F}(\mathbf{x}_{-t};\theta) \\ \tilde{\mathbf{h}} &= \lambda\mathbf{h} + (1-\lambda)\mathbf{h}'. \end{aligned} \tag{4}$$

Here, the transformer encoder $\mathbf{F}(.;\theta)$ for both $\mathbf{x}_{-e}$ and $\mathbf{x}_{-t}$ is sharing the weights. Then we use interpolated hidden representations $\tilde{\mathbf{h}}$ as the input to the final CRF tagger.

When inferring tags on unlabeled sentences without entity triggers, we expect the trained $\mathbf{F}(.;\theta)$ to generate representation from the input $\mathbf{x} \in \mathcal{D}_u$, which can generalize well to unseen entities given the seen contextual clues, and seen entities given new surrounding context. We then use it as an input to the final CRF tagger to get tag predictions.

Computational Cost. The computation overhead of training `TIN` is caused by forwarding input two times into the transformers to get two different representations, which is 1.5X larger than vanilla models (e.g. `BERT+TIN+CRF` vs. `BERT+CRF`). However, the overhead of inference is same as vanilla models since the inference steps are same.

4 Experimental Setup

In this section we describe datasets along with the baseline methods and experiment settings.

4.1 Datasets

We consider three NER datasets as target tasks. We consider two datasets for a bio-medical domain: **BC5CDR** (Li et al., 2016a), **JNLPBA** (Collier and Kim, 2004) and one dataset for a general domain: **CoNLL03** (Tjong Kim Sang, 2002). For BC5CDR and CoNLL03, we also have crowd-sourced entity trigger dataset $\mathcal{D}_{HT}$ (Lin et al., 2020) to compare the quality of our automatically extracted triggers with. They randomly sample 20% of the data from each of the train sets and ask crowd-workers to select triggers for entities in those train sets. Details on datasets are discussed in Appendix A.3.

4.2 Compared Methods

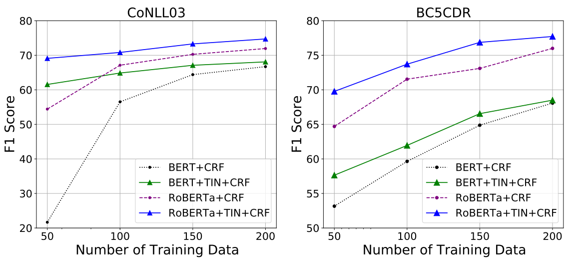
To show the effectiveness of entity triggers, we compare baseline models that learn from entity-labeled dataset $\mathcal{D}_L$ and trigger-labeled dataset $\mathcal{D}_T$ respectively. Here, we compare models under three different embedders: GloVE (Pennington et al., 2014), BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019b). We don't include few-shot learning and semi-supervised learning due to following reasons: (1) Few-shot learning (Yang and Katiyar, 2020) is a N-way k-shot task that needs N classes, each with k training examples. This setup is restrictive because it requires a special sampling strategy since the number of training examples for the class must be fixed. (2) Semi-supervised learning (Peng et al., 2019; Shang et al., 2018; Yang et al., 2018; Liu et al., 2019a) uses additional supervisions and in-domain unlabeled data to augment train data by weakly labeling, which is different from our setup that only uses limited in-domain labeled data.

Baselines without triggers We apply the following standard models on $\mathcal{D}_L$: (1) **BLSTM+CRF** first adopts bidirectional LSTM to encode word embeddings into the representations. Then, it feeds the representations into a linear layer to get label scores, and put the scores into a CRF tagger to predict the optimal path of entity tags; (2) **CRF** considers word embeddings directly as the representations and conducts the same process of (1).

Methods with Triggers We apply the following models on $\mathcal{D}_T$: (1) **BLSTM+TMN+CRF** (Lin et al., 2020) adopts the structured self-attention layer (Lin et al., 2017) above the bidirectional LSTM to encode the sentence and entity trigger into vector representation respectively. It then learns trigger representations that can be generalized to unseen sentences to tag the named entity; (2) **TMN+CRF** adopts the structured self-attention layer directly above the embeddings, and conducts the same process of (1); (3) **TIN+CRF** is the *trigger interpolation network* with CRF layer.

5 Results and Analysis

Here we look into a series of analysis questions as follows: (1) Can `TIN` with automatically extracted triggers improve the performance of existing transformer-based models? (2) Can `TIN` with automatically extracted triggers generalize well on the unseen entities? (3) Are automatically extracted triggers work better than human-provided triggers? (4) Can we improve the performance by asking human to refine automatically extracted triggers? We present experiment settings and details in Appendix A.1- A.2.



(a) CoNLL03 (b) BC5CDR

Figure 6: Performance Comparison (F1-score) on CoNLL03 and BC5CDR by small numbers of train data instances (50, 100, 150, 200).

5.1 Performance Comparison

Here, we test all models by varying the amount of training data from 20% to 100% to show the impact of train data size. In Table 1, we report the performance of the baseline approaches and our model variants on three different datasets. Here are findings: (1) `BLSTM` harms the performance of transformer-based models while it works well with GloVE embeddings; (2) Models that receive both entities and triggers as input generally outperform the *entity-only* baselines (See `CRF` vs. `TIN+CRF`, `TMN+CRF`); (3) `TIN` outperforms `TMN`, which shows that `TIN` is doing better leveraging entity triggers for NER, and `RoBERTa+TIN+CRF` outperforms all the baselines regardless of the amount of data that is used to train it; (4) Comparing `BERT+TIN+CRF` to `BERT+CRF`, we observe a performance drop in CoNLL03. We further investigate this phenomenon and find a large drop in F1 score (from 0.82 to 0.79) for the MISC class from the `BERT+TIN+CRF` (See Table 3), which shows that triggers may provide a precision decreasing signal for the MISC type.

Performance in extreme label scarcity settings We hypothesize that our models will have larger performance gains in extreme label scarcity settings, because of their ability to leverage additional information from triggers which enables them to reap more benefits from given training data. To investigate this we observe the performance of our models and baselines starting with only 50-200 sentences to train them. Figure 6 shows the performance under the extreme label scarcity setting. Here, `TIN+CRF` achieves large performance gain in extremely low-resource setting. Specifically, we observe over 50% relative gain compared to the baseline for 50 training sentences.

Unseen entity generalization We hypothesize that our framework can generalize well to unseen

Embedder	**Method** / Train set %	**BC5CDR**					**JNLPBA**					**CoNLL03**				
		20%	40%	60%	80%	100%	20%	40%	60%	80%	100%	20%	40%	60%	80%	100%
GloVE	+ BLSTM+CRF	71.92	76.29	79.04	80.72	81.07	66.36	69.31	71.25	71.90	72.79	85.06	88.33	88.98	89.84	90.72
	+ BLSTM+TMN+CRF	**74.70**	**78.15**	**80.57**	**82.77**	**83.37**	**66.78**	**70.23**	**71.41**	**71.7**	**72.55**	**87.46**	**88.88**	**89.39**	**90.16**	**90.24**
BERT	+ BLSTM+CRF	44.51	65.88	74.23	80.65	82.56	59.26	69.39	72.04	73.24	73.26	68.60	87.09	89.42	90.20	90.86
	+ CRF	75.30	80.52	82.94	84.00	85.02	69.02	70.84	72.58	73.06	73.18	**88.61**	**90.20**	**91.10**	**91.37**	**91.48**
	+ BLSTM+TMN+CRF	74.98	78.24	82.52	82.89	84.38	69.56	71.36	72.26	72.92	73.34	86.88	88.78	90.05	90.35	90.99
	+ TMN+CRF	75.07	80.14	82.57	84.03	85.49	69.25	71.34	71.70	72.98	73.38	88.39	89.34	90.24	90.83	91.06
	+ TIN+CRF (Ours)	**77.37**	**81.40**	**83.23**	**85.25**	**85.74**	**70.48**	**72.10**	**72.81**	**73.91**	**74.83**	87.84	89.64	89.71	90.39	90.75
RoBERTa	+ CRF	82.85	84.63	86.08	86.44	87.10	71.07	72.19	73.32	73.50	75.37	90.53	91.63	91.90	92.06	93.09
	+ BLSTM+TMN+CRF	83.00	84.89	86.16	86.69	87.78	71.51	72.91	73.43	74.65	75.52	90.76	91.50	91.84	92.22	92.47
	+ TMN+CRF	83.89	85.56	86.65	87.31	87.78	71.79	72.87	73.10	74.19	75.13	90.82	91.47	92.32	92.42	92.57
	+ TIN+CRF (Ours)	**84.45**	**86.09**	**87.5**	**87.84**	**88.09**	**73.12**	**74.23**	**74.45**	**74.96**	**76.98**	**91.37**	**92.03**	**92.63**	**92.51**	**93.24**

Table 1: Performance comparison (F1-score) by different percentage usage of the train data. For `TMN` and `TIN` baselines, we use the top 2 candidate phrases from SOC with constituency parsing as triggers. Best models under each embedder are **bold**.

Encoder	**Method** / Train set %	**BC5CDR**					**JNLPBA**					**CoNLL03**				
		20%	40%	60%	80%	100%	20%	40%	60%	80%	100%	20%	40%	60%	80%	100%
BERT	+ CRF	63.93	67.95	70.36	73.43	73.66	56.81	57.00	58.43	59.49	60.11	78.65	82.32	82.24	82.90	83.36
	+ TMN+CRF	65.37	68.51	68.80	72.17	73.46	56.39	56.98	58.14	59.01	59.98	80.18	82.39	82.16	82.46	83.93
	+ TIN+CRF (Ours)	**67.10**	**69.21**	**72.32**	**73.81**	**74.65**	**57.19**	**58.73**	**59.67**	**59.90**	**60.36**	**81.52**	**82.82**	**83.01**	**83.72**	**84.66**
RoBERTa	+ CRF	72.16	74.55	75.92	76.39	76.17	58.88	60.05	61.22	62.32	63.71	82.39	84.06	84.50	84.98	85.61
	+ TMN+CRF	73.11	74.27	76.13	76.18	76.29	59.62	60.73	61.17	62.44	62.98	83.30	**85.30**	85.71	**85.47**	85.97
	+ TIN+CRF (Ours)	**73.65**	**75.78**	**77.14**	**77.34**	**76.90**	**60.85**	**61.50**	**62.37**	**63.55**	**64.11**	**84.11**	84.82	**85.84**	85.31	**87.38**

Table 2: Performance comparison (F1-score) on filtered test set by different percentage usage of the train data. Filtered test set is the set which the entities did not appear in the train and development sets. For `TMN` and `TIN` baselines, we use the top 2 candidate phrases from SOC with constituency parsing as triggers. Best models under each embedder are **bold**.

Type	`BERT+CRF`			`BERT+TIN+CRF`		
	Precision	Recall	F1-score	Precision	Recall	F1-score
LOC	0.92	0.94	0.93	0.91	0.93	0.92
MISC	0.81	0.82	**0.82**	0.75	0.84	0.79
ORG	0.88	0.90	0.89	0.86	0.90	0.88
PER	0.97	0.96	0.96	0.96	0.96	0.96

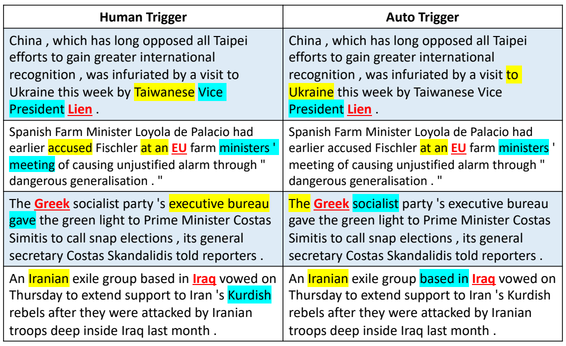
Table 3: Classification Report (F1-score) of `BERT+CRF` and `BERT+TIN+CRF` on CoNLL03.

entities not presented in the training data, attribute to prior knowledge of contextual clues. To evaluate the generalizability of the model, we create a filtered test set which contains instances with only entities that do not appear in the train data. Table 6 shows the performance of model variants on the filtered test set by different amount of training data. Here, `TIN+CRF` shows its effectiveness on predicting unseen entities, and much more when we compare under the small amount of training data.

5.2 Human-in-the-loop Trigger Extraction

We aim to study whether human participation in trigger creation could be helpful. First, we compare the performance of model variants trained with automatically extracted triggers (`auto`) and human-provided triggers (`human`), and check its label efficiency. Then, we conduct a small-scale experiment of trigger refinement by human annotators.

Human-curated vs. Auto Triggers. We use $\mathcal{D}_{HT}$ as the source of human triggers and use the same dataset to extract auto triggers with `SOC` algorithm (see Appendix Table 6). We then sample 25%, 50%, and 75% of the instances from both to construct 5%, 10%, 15% percent of our experimentation dataset (since $\mathcal{D}_{HT}$ is a 20% random sample from $\mathcal{D}_L$). One big difference between `human` and `auto` is whether the triggers are contiguous token spans or not. For example, humans are asked to annotate a group of word tokens that represent *"general"* phrase like *"had dinner at"* from the sentence *"We had a fantastic dinner at Sunnongdan."*, while a set of phrase candidates $\mathcal{P}$ from the constituency parse tree can only contain the contiguous token spans. Figure. 7 shows examples of `human` and `auto`. These examples are from CoNLL03, and `auto` are extracted from the entity token classifier which is trained on 20% of the train data. Tab. 4 shows that `auto` are comparable or even stronger than `human` even though created with no human labeling. The success of auto triggers might be that their impact on the entity labeling is directly at the model level, while human triggers, even if they are meaningful on the surface level, might have less impact in determining the entity label as they do not mimic what the model thinks. We manually inspect the `auto` and `human` and found that `auto` are consecutive while `human` are usually non-consecutive. Even though there could be many reasons for the sub-optimal performance of human selected triggers available in the dataset (Lin et al., 2020), we do not rule out the possibility of leveraging human expertise to help. Further case examples are presented in Appendix A.4.



Human Trigger	Auto Trigger
China , which has long opposed all Taipei efforts to gain greater international recognition , was infuriated by a visit to Ukraine this week by Taiwanese Vice President **Lien** .	China , which has long opposed all Taipei efforts to gain greater international recognition , was infuriated by a visit to Ukraine this week by Taiwanese Vice President **Lien** .
Spanish Farm Minister Loyola de Palacio had earlier accused Fischler at an **EU** farm ministers ' meeting of causing unjustified alarm through " dangerous generalisation . "	Spanish Farm Minister Loyola de Palacio had earlier accused Fischler at an **EU** farm ministers ' meeting of causing unjustified alarm through " dangerous generalisation . "
The **Greek** socialist party 's executive bureau gave the green light to Prime Minister Costas Simitis to call snap elections , its general secretary Costas Skandalidis told reporters .	The **Greek** socialist party 's executive bureau gave the green light to Prime Minister Costas Simitis to call snap elections , its general secretary Costas Skandalidis told reporters .
An Iranian exile group based in **Iraq** vowed on Thursday to extend support to Iran 's Kurdish rebels after they were attacked by Iranian troops deep inside Iraq last month .	An Iranian exile group based in **Iraq** vowed on Thursday to extend support to Iran 's Kurdish rebels after they were attacked by Iranian troops deep inside Iraq last month .

Figure 7: Top 2 highlighted `auto` and `human` triggers corresponding to the boldfaced and underlined entity.

BC5CDR	`BLSTM+TMN+CRF`		`BERT+TIN+CRF`		`RoBERTa+TIN+CRF`	
Percentage / **Model**	`human`	`auto`	`human`	`auto`	`human`	`auto`
5%	**26.96**	24.70	66.20	**66.50**	75.79	**76.92**
10%	**46.24**	43.54	71.25	**71.84**	80.92	**81.63**
15%	**51.29**	50.44	73.88	**74.11**	83.54	**83.87**
20%	**56.28**	54.91	75.97	**76.58**	83.88	**84.17**
CoNLL03	`BLSTM+TMN+CRF`		`BERT+TIN+CRF`		`RoBERTa+TIN+CRF`	
Percentage / **Model**	`human`	`auto`	`human`	`auto`	`human`	`auto`
5%	56.39	**57.95**	78.17	**78.56**	84.72	**85.71**
10%	61.89	**66.58**	81.67	**82.19**	87.80	**88.12**
15%	67.48	**69.41**	83.67	**85.13**	88.40	**89.68**
20%	71.11	**74.43**	84.88	**85.58**	89.68	**90.21**

Table 4: Performance comparison (F1-score) of `TMN` and `TIN` with `human` and `auto` triggers.

Efficiency of Human-curated vs. Auto Triggers. We conduct a study to measure how trigger extraction by human affects labeling efficiency. First, we found that labeling triggers along with the entities is 1.5 times slower than labeling the entities only. Given this observation, we compare the performance between TIN models with `human` and `auto` by holding annotation time constant. We present the study in Figure. 8. Each marker on the x-axis of the plots indicate a certain annotation time, which is represented by approximate time. First, we could find that if we ask human to get triggers from scratch, it may not work better (See `RoBERTa+CRF` vs. `RoBERTa+TIN+CRF+human`). However, if we use auto triggers, we could achieve better performance and label efficiency (See `RoBERTa+TIN+CRF+auto`).

Human-in-the-loop Trigger Refinement. In previous study, we could find that asking human to annotate triggers from the scratch is not efficient. Here, we further think about asking human to refine auto triggers to reflect human decisions in a less labor intensive way. For all our previous experiments, we use the top two auto triggers, which limits our capacity to make the best use of them. In this experiment, given a training set with labeled entities, we extract five auto triggers (Sec. 3.1), show them to a human in a minimal interface, and ask for relevance judgments (relevant/non-relevant). We judge relevance of the automatically extracted triggers for entities in 50 - 200 sentences. Figure. 9 shows that we get an additional performance boost with more than 50 training sentences, when human-refined auto triggers are used. It shows promise for blending human expertise with auto triggers.

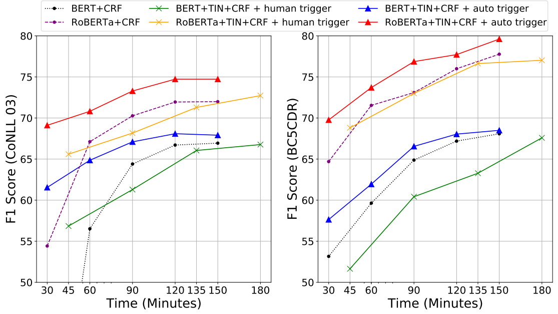


Figure 8: Performance Comparison (F1-score) by annotators' labeling time cost.

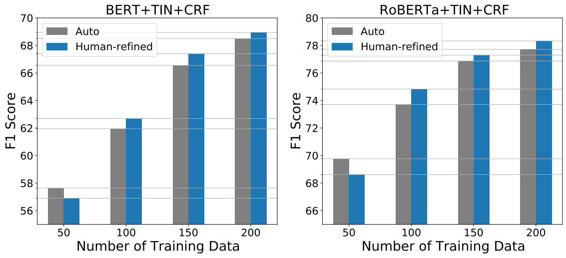


Figure 9: Performance Comparison (F1-score) on BC5CDR by different numbers of train data (50, 100, 150, 200) with auto and human-refined auto triggers.

6 Conclusion

In this paper, we proposed a novel two-stage framework to generate and leverage explanations for named entity recognition. It automatically extracts essentially human-readable clues in the text, which is called entity triggers, by sampling and occlusion algorithm and leverages these triggers with trigger interpolation network. The model effective learns the prior knowledge to infer the entity boundaries and types by leveraging the contextual clues. We showed that our framework, named AUTOTRIGGER, successfully generates entity triggers and effectively leverages them to improve the overall performance, especially in the label scarcity setting for technical domains where domain-expert annotations are very limited due to the high cost. Moreover, it shows better generalizability on unseen entities that do not appear in training data. We believe that this work opens up future works that can be extended to semi-supervised learning or distant supervised learning which can effectively use automatically extracted triggers to weakly label the unlabeled corpus.

Limitations

AUTOTRIGGER could reduce human efforts on collecting rationales for NER to improve the label efficiency and unseen entity generalizability of the model. However, it has a limitation that the time cost of extracting entity triggers is pretty expensive (i.e., (1) train entity token classifier; (2) run constituency parsing to retrieve trigger candidates; and (3) run post-hoc explanation to assign importance score to each candidate). This limitation raises open questions about whether reducing total training time cost or human effort is more efficient and important.

Acknowledgement

This work was funded by the Defense Advanced Research Projects Agency with award W911NF-19-20271.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *Proc. of ICLR*.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Nigel Collier and Jin-Dong Kim. 2004. Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proc. of ACL*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

G. DeJong and R. Mooney. 2004. Explanation-based learning: An alternative view. *Machine Learning*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. *ArXiv preprint*, abs/2108.10604.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Ves Stoyanov, and Alexis Conneau. 2020. Self-training improves pre-training for natural language understanding. *arXiv preprint arXiv:2010.02194*.

John Foley, Sheikh Muhammad Sarwar, and James Allan. 2018. Named entity recognition with extremely limited data. *arXiv preprint arXiv:1806.04411*.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, and Christopher Ré. 2018. Training classifiers with natural language explanations. In *Proc. of ACL*.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. In *Proc. of ACL*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Proceedings of Machine Learning Research.

Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C. Wallace. 2020. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4459–4473, Online. Association for Computational Linguistics.

Chengyue Jiang, Yinggong Zhao, Shanbo Chu, Libin Shen, and Kewei Tu. 2020. Cold-start and interpretability: Turning regular expressions into trainable recurrent neural networks. In *Proc. of EMNLP*.

Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. 2020. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *Proc. of ICLR*.

Vidur Joshi, Matthew Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. In *Proc. of ACL*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Proceedings of Machine Learning Research.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proc. of NAACL-HLT*.

Dong-Ho Lee, Akshen Kadakia, Kangmin Tan, Mahak Agarwal, Xinyu Feng, Takashi Shibuya, Ryosuke Mitani, Toshiyuki Sekiya, Jay Pujara, and Xiang Ren. 2022. Good examples make a faster learner: Simple demonstration-based learning for low-resource NER. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2687–2700, Dublin, Ireland. Association for Computational Linguistics.

Dong-Ho Lee, Rahul Khanna, Bill Yuchen Lin, Seyeon Lee, Qinyuan Ye, Elizabeth Boschee, Leonardo Neves, and Xiang Ren. 2020a. LEAN-LIFE: A label-efficient annotation framework towards learning from explanation. In *Proc. of ACL*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020b. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016a. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database : the journal of biological databases and curation*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016b. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Bill Yuchen Lin, Wenyang Gao, Jun Yan, Ryan Moreno, and Xiang Ren. 2021. RockNER: A simple method to create adversarial examples for evaluating the robustness of named entity recognition models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3728–3737, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bill Yuchen Lin, Dong-Ho Lee, Ming Shen, Ryan Moreno, Xiao Huang, Prashant Shiralkar, and Xiang Ren. 2020. TriggerNER: Learning with entity triggers as explanations for named entity recognition. In *Proc. of ACL*.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proc. of ICLR*.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Fangzheng Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Proc. of AAAI*.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019a. Towards improving neural named entity recognition with gazetteers. In *Proc. of ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proc. of ACL*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proc. of ACL*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proc. of EMNLP*.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving named entity recognition with attentive ensemble of syntactic information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*.

Minlong Peng, Xiaoyu Xing, Qi Zhang, Jinlan Fu, and Xuanjing Huang. 2019. Distantly supervised named entity recognition using positive-unlabeled learning. In *Proc. of ACL*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proc. of EMNLP*.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proc. of ACL*.

A. Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, and C. Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *VLDB*.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proc. of ACL*.

Esteban Safranchik, Shiying Luo, and Stephen H. Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *AAAI*.

Sheikh Muhammad Sarwar, John Foley, and James Allan. 2018. Term relevance feedback for contextual named entity retrieval. In *CHIIR*.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Proc. of EMNLP*.

Chandan Singh, W. James Murdoch, and Bin Yu. 2019. Hierarchical interpretations for neural network predictions. In *International Conference on Learning Representations*.

Yuanhe Tian, W. Shen, Yan Song, Fei Xia, Min He, and Kenli Li. 2020. Improving biomedical named entity recognition with syntactic information. *BMC Bioinformatics*.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1800–1812, Online. Association for Computational Linguistics.

Ziqi Wang, Yujia Qin, Wenxuan Zhou, Jun Yan, Qinyuan Ye, Leonardo Neves, Zhiyuan Liu, and Xiang Ren. 2020. Learning from explanations with neural execution tree. In *Proc. of ICLR*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP*.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics.

Qinyuan Ye, Xiao Huang, Elizabeth Boschee, and Xiang Ren. 2020. Teaching machine comprehension with compositional explanations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Omar Zaidan and Jason Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *Proc. of EMNLP*.

A Appendix

A.1 Experiment Settings

We implement all the baselines using PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2020). We set the batch size and learning rate to 10 and 0.01 for GloVE embedder models (i.e., `GloVE+BLSTM+CRF`, `GloVE+TMN+CRF`) while we set 30 and 2e-5 for all other transformer embedder models (See Table 5). Note that for experiments in extreme low resource setting (Sec. 5.1), we set the batch size to 4 for training the models due to the extremely limited training data.

Trigger Matching Network For transformer-based `TMN` (i.e., `BERT+BLSTM+TMN+CRF`, `BERT+TMN+CRF`, etc.) we re-implement since the original repository does not support transformer embedders.

Trigger Interpolation Network For our `TIN`, we set the interpolation λ to 0.5.

Automatic Trigger Extraction For automatic trigger extraction stage, we build the entity token classifier with cased BERT-base encoder for `BERT+TIN+CRF` and RoBERTa-large for `RoBERTa+TIN+CRF`. The entity token classifier consists of the transformer encoder to encode each word token followed by a token-level linear layer that classifies each token to an entity tag. We use a batch size of 16 and learning rate of 1e-4 for training. For experiments under extreme low resource setting, we set batch size to 4 similar to the `TIN` models. To run context sampling in the SOC algorithm, we use a LSTM language model which is pre-trained on the training data.

A.2 Evaluation Metrics

We evaluate our framework by recall (R), precision (P), and F1-score (F1), though only report F1 in these experiments. Recall (R) is the number of correctly recognized named entities divided by the total number of named entities in the corpus, and precision (P) is the number of correctly recognized named entities divided by the total number of named entities recognized by the framework. A recognized entity is correct if both its boundary and its entity type are exact matches to the annotations in the test data. F1-score is the harmonic mean of precision and recall.

Embedder	GloVE	Transformer	
		BERT	RoBERTa
batch size	10	30	30
learning rate	0.01	2e-5	2e-5
epochs	10	10	10
LSTM hidden dimension	200	-	-

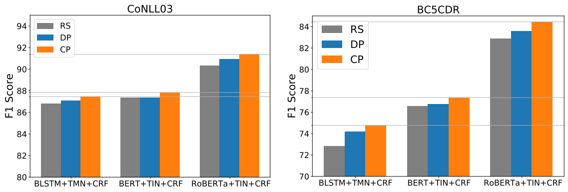
Table 5: Experimental setting details.

A.3 Data Statistics

BC5CDR (Li et al., 2016a) is a bio-medical domain NER dataset from BioCreative V Chemical and Disease Mention Recognition task. It has 1,500 articles containing 15,935 CHEMICAL and 12,852 DISEASE mentions. **JNLPBA** (Collier and Kim, 2004) is a bio-medical domain NER dataset for the Joint Workshop on NLP in Biomedicine and its Application Shared task. It is widely used for evaluating multiclass biomedical entity taggers and it has 14.6K sentences containing PROTEIN, DNA, RNA, CELL LINE and CELL TYPE. **CoNLL03** (Tjong Kim Sang, 2002) is a general domain NER dataset that has 22K sentences containing four types of general named entities: LOCATION, PERSON, ORGANIZATION, and MISCELLANEOUS entities that do not belong in any of the three categories.

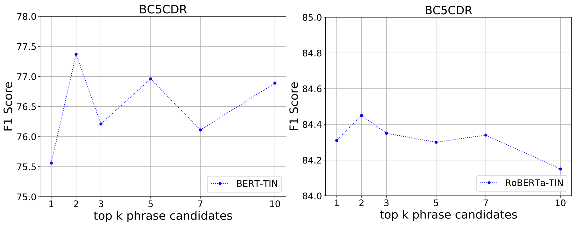
A.4 Performance Analysis

Trigger Candidate Variants. In Sec 3.1, we first constructed a set of phrase candidates $\mathcal{P}$ for which the importance score is computed. To show the efficacy of constituency parsing for constructing trigger candidates, we conduct an ablation study on different variants of it. For the construction, we compare three variants: (1) `RS` is random selection. It randomly chooses n contiguous tokens to be grouped as a phrase for k times. Consequently, $\mathcal{P}$ is composed of k random spans. (2) `DP` is dependency parsing. Here, to generate $\mathcal{P}$, we first parse the input sentence using dependency parsing. Then, we traverse from the position of entity mention in the input sentence using depth-first-traversal and get a list of tokens visited for each hop up to 2-hops. Finally, for each hop, we convert the list of tokens to a list of phrases by merging the tokens that are contiguous into a single phrase. (3) `CP` is constituency parsing, which is our current method (see Sec. 3.1). We expect each variant to provide different syntactic signals to our framework. Figure 10 shows the model's performance with triggers that have been selected from different sets of phrase candidates. As we can see, constituency parsing



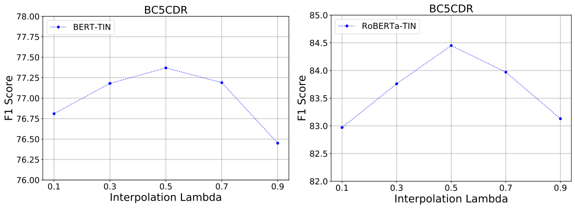
(a) CoNLL03 (b) BC5CDR

Figure 10: Performance comparison (F1-score) of entity+trigger baselines on 20% training dataset of CoNLL03 and BC5CDR with different trigger candidate variants.



(a) BERT+TIN+CRF (b) RoBERTa+TIN+CRF

Figure 12: Performance comparison (F1-score) of entity+trigger baselines on 20% training dataset of BC5CDR with different number of triggers k.



(a) BERT+TIN+CRF (b) RoBERTa+TIN+CRF

Figure 11: Performance comparison (F1-score) of entity+trigger baselines on 20% training dataset of BC5CDR with different interpolation weight λ.

yields consistently better performance by providing better quality of syntactic signals than others.

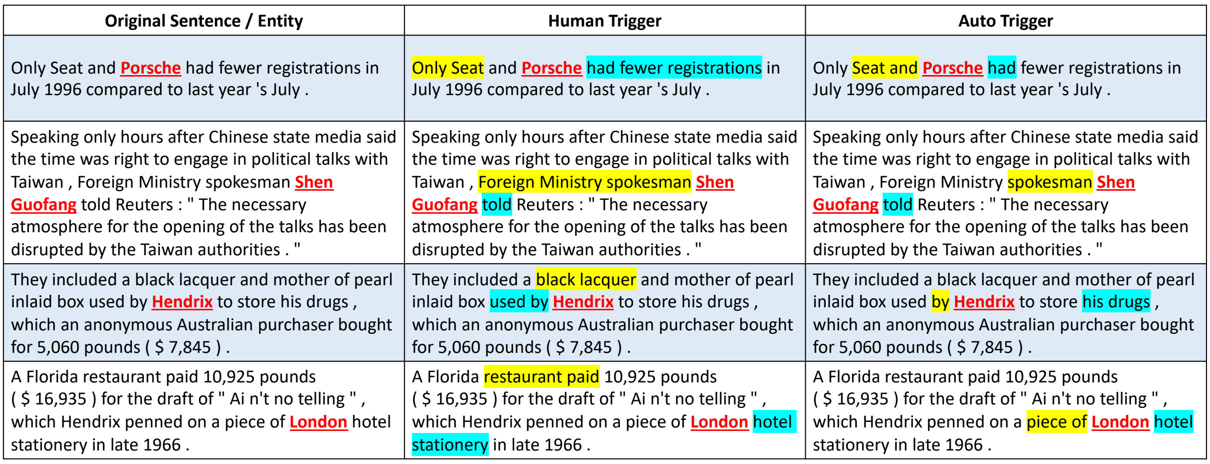
Sensitivity Analysis of interpolation hyperparameter (λ). In Sec 3.2, we linearly interpolated two different sources of knowledge by weight λ 0.5. To show how the weight λ affects the performance, we conduct an ablation study on different λ distribution. As we can see from Figure. 11, the framework achieves the highest performance when λ is set to 0.5. It supports that the model achieves the best when we interpolate the entity and trigger knowledge in equal.

Effect of number of triggers In Sec. 3.1, we pick the top k candidate phrases with the highest importance score as the entity triggers after obtaining the importance score for all phrase candidates. For our main experiment, we use top 2 candidate phrases (see Table 1). To show how the number of triggers affects the performance, we conduct an ablation study on model performance by different k. As we can see from Figure. 12, the framework achieves the highest performance when we use top 2 phrase candidates as triggers.

A.5 Related Works

NER with Additional Supervision Previous and recent research has shown that encoding syntactic information into NER models compensate for the lack of labeled data (Tian et al., 2020). The improvement is consistent across word embedding based encoding (e.g. biLSTM) as well as unsupervised language model based encoding (e.g. BioBERT (Lee et al., 2020b)) of the given text. Typically, the external information that is encoded include POS labels, syntactic constituents, and dependency relations (Nie et al., 2020; Tian et al., 2020). The general mechanism to include linguistic information into NER model is to represent them using word vectors and then concatenate those representations with the original text representation. This approach fails to identify the importance of different types of syntactic information. Recently, Tian et al. (2020) and Nie et al. (2020) both showed that key-value memory network (KVMN) (Miller et al., 2016) are effective in capturing importance of linguistic information arising from different sources. KVMN has been shown to be effective in leveraging extra information, such as knowledge base entities, to improve question answering tasks. Before applying KVMN, contextual information about a token is encoded as the key and syntactic information are encoded as values. Finally, weights over the values are computed using the keys to obtain a representation of the values and concatenate it with the context features. Our approach uses token level features extracted by an explanation generation model, but later train to be able to pick-up those explanations directly from the text at inference time.

Limited Training Data for NER. The simplest way to approach the problem of limited data for NER is to use dictionary based weak supervision. An entity dictionary is used to retrieves unlabeled sentences from a corpus and weakly label them to create additional noisy data. This approach suffers from low recall as the training data covers a limited number of entities. The models tend to



Original Sentence / Entity	Human Trigger	Auto Trigger
Only Seat and **Porsche** had fewer registrations in July 1996 compared to last year 's July .	Only Seat and **Porsche** had fewer registrations in July 1996 compared to last year 's July .	Only Seat and **Porsche** had fewer registrations in July 1996 compared to last year 's July .
Speaking only hours after Chinese state media said the time was right to engage in political talks with Taiwan , Foreign Ministry spokesman **Shen Guofang** told Reuters : " The necessary atmosphere for the opening of the talks has been disrupted by the Taiwan authorities . "	Speaking only hours after Chinese state media said the time was right to engage in political talks with Taiwan , Foreign Ministry spokesman **Shen Guofang** told Reuters : " The necessary atmosphere for the opening of the talks has been disrupted by the Taiwan authorities . "	Speaking only hours after Chinese state media said the time was right to engage in political talks with Taiwan , Foreign Ministry spokesman **Shen Guofang** told Reuters : " The necessary atmosphere for the opening of the talks has been disrupted by the Taiwan authorities . "
They included a black lacquer and mother of pearl inlaid box used by **Hendrix** to store his drugs , which an anonymous Australian purchaser bought for 5,060 pounds ($ 7,845) .	They included a black lacquer and mother of pearl inlaid box used by **Hendrix** to store his drugs , which an anonymous Australian purchaser bought for 5,060 pounds ($ 7,845) .	They included a black lacquer and mother of pearl inlaid box used by **Hendrix** to store his drugs , which an anonymous Australian purchaser bought for 5,060 pounds ($ 7,845) .
A Florida restaurant paid 10,925 pounds ($ 16,935) for the draft of " Ai n't no telling " , which Hendrix penned on a piece of **London** hotel stationery in late 1966 .	A Florida restaurant paid 10,925 pounds ($ 16,935) for the draft of " Ai n't no telling " , which Hendrix penned on a piece of **London** hotel stationery in late 1966 .	A Florida restaurant paid 10,925 pounds ($ 16,935) for the draft of " Ai n't no telling " , which Hendrix penned on a piece of **London** hotel stationery in late 1966 .

Figure 13: Case examples of `auto` trigger and `human` trigger. Entities are **bold** and underlined with red color, and its triggers are highlighted. Different triggers are color-coded.

bias towards the surface form of the entities it has observed in the dictionary. There has also been approaches to retrieve sentences from a large corpus that are similar to sentences in the low-resource corpus to enrich it. These self-training approaches have been shown to be effective both in extremely limited data (Foley et al., 2018; Sarwar et al., 2018) as well as limited data scenario (Du et al., 2020). Even though these data enhancement approaches explore a corpus to find related data cases, they do not exploit the explanation-based signals that is available within the limited data.

Learning from Explanations. Recent works on Explainable AI are primarily focused on debugging the black box models by probing internal representations (Adi et al., 2017; Conneau et al., 2018), testing model behavior using challenge sets (McCoy et al., 2019; Gardner et al., 2020; Ribeiro et al., 2020), or analyzing an impact of input examples by input perturbations or influence function looking at input examples (Ribeiro et al., 2016; Koh and Liang, 2017). However, for an explanation of the model to be effective, it must provide not only the reasons for the model's prediction but also suggestions for corresponding actions in order to achieve an objective. Efforts to cope with this issue by incorporating human explanations into the model are called Explanation-based learning (DeJong and Mooney, 2004). These works are aiming to exploit generalized explanations for drawing inferences from unlabeled data while maintaining model transparency. Most prior works on explanation-based learning are mainly focused on facilitating logical rules as an explanation. They use such rules to create weak supervision (Ratner et al., 2017) and regularize posterior (Hu et al., 2016, 2017). Another form of explanations can be specific words in the sentence which aligns to our work. Notable work in this line asks annotators to highlight important words, then learn a generative model over parameters given these rationales (Zaidan and Eisner, 2008).

Dataset	**Entity Type**	Original $\mathcal{D}_L$	Crowd-sourced trigger $\mathcal{D}_{HT}$	
		# of Entities	# of Entities	# of Human Triggers
CONLL 2003	PER	6,599	1,608	3,445
	ORG	6,320	958	1,970
	MISC	3,437	787	2,057
	LOC	7,139	1,781	3,456
	Total	23,495	5,134	10,938
BC5CDR	DISEASE	4,181	906	2,130
	CHEMICAL	5,202	1,085	1,640
	Total	9,383	1,991	3,770
JNLPBA	PROTEIN	27,802	-	-
	DNA	8,480	-	-
	RNA	843	-	-
	CELL LINE	3,429	-	-
	CELL TYPE	6,191	-	-
	Total	46,745	-	-

Table 6: Train data statistics.

Incorporating Task-Specific Concept Knowledge into Script Learning

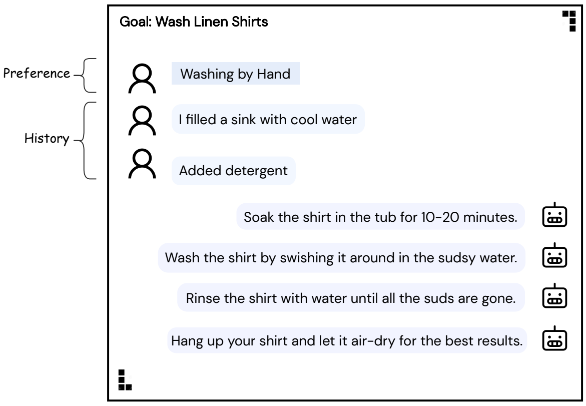
Chenkai Sun, Tie Xu, ChengXiang Zhai, Heng ji
University of Illinois at Urbana-Champaign, IL, USA
Alibaba, Hangzhou, China
`{chenkai5, czhai, hengji}@illinois.edu`
`xutie.xt@alibaba-inc.com`

Abstract

In this paper, we present TETRIS, a new task of Goal-Oriented Script Completion. Unlike previous work, it considers a more realistic and general setting, where the input includes not only the goal but also additional user context, including preferences and history. To address this problem, we propose a novel approach, which uses two techniques to improve performance: (1) concept prompting, and (2) script-oriented contrastive learning that addresses step repetition and hallucination problems. On our WikiHow-based dataset, we find that both methods improve performance.

1 Introduction

A Goal-Oriented Script refers to a sequence of events that describe some stereotypical activities for achieving a specified goal (Feigenbaum et al., 1981; Lyu et al., 2021). It is important to study how to automatically construct instructional scripts because it enables many high-impact applications such as robot action planning (Conti et al., 2020; Mohanan and Salgoankar, 2018), causal reasoning (Guo et al., 2020), and task-oriented dialogue generation (Chen et al., 2017). While previous work has shown that neural models are capable of constructing the entire script given a goal (Lyu et al., 2021; Sakaguchi et al., 2021), their proposed tasks are highly restrictive and based on overly simplified assumptions about the application because it ignored both the usage context (e.g., what is a preferred way of solving the problem, and what steps have been executed already) and the variable personal preferences that a user may have (e.g., a goal might need be achieved in different ways). Take "Make Eggless Cupcakes" as an example, the user might want to make it with different equipment (e.g., by using a rice cooker instead of an oven) or styles (e.g., to have some banana flavor), or the user has already completed some steps but would like to seek information on what to proceed next; a machine learning model trained on goal-to-script tasks as done in the existing work cannot adaptively provide solutions under such situations.



Washing by Hand
I filled a sink with cool water
Added detergent
Soak the shirt in the tub for 10-20 minutes.
Wash the shirt by swishing it around in the sudsy water.
Rinse the shirt with water until all the suds are gone.
Hang up your shirt and let it air-dry for the best results.

Figure 1: An example illustrating the task. The input consists of a specified goal (e.g., wash line shirts), an optional preference (e.g., washing by hand), and a history of steps. The model is asked to generate the remaining steps to achieve the goal.

We address the limitations of the previous task formulation and propose TETRIS, a more general task of goal-oriented script learning that allows flexible usage scenarios, and therefore it accommodates more realistic needs in downstream applications. More specifically, as shown in Figure 1, the task considers both user preference and history as input in addition to the goal. From a probabilistic perspective, instead of modeling the problem with $p(\text{Script}|\mathcal{G})$, we model it as

$$p(\text{Completion of Script}|\mathcal{H},\mathcal{P},\mathcal{G}) = \prod_i p(\text{Script}_i|\text{Script}_{<i},\mathcal{P},\mathcal{G})$$

where i indicates each step index in the script, $\mathcal{P}$ denotes user preference, $\mathcal{H}$ indicates History, and $\mathcal{G}$ abbreviates Goal.

To solve the new problem, we can use a com-

---

https://github.com/chenkaisun/Tetris

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3026–3040
May 2-6, 2023

monly used baseline method, i.e., using a seq2seq model (Lewis et al., 2019; Raffel et al., 2019), but a direct application of such a model would not be optimal because the model's understanding of the context of the goal is inevitably shallow and thus unlikely be able to fully exploit the extra context information we have in the problem setup. To address this limitation, we propose to enrich the representation with additional task-specific concepts so as to enable the model to understand the context more deeply and leverage relevant knowledge about those concepts encoded in the training data the model has been exposed to. To obtain task-specific knowledge needed for solving script learning, we introduce TASK CONCEPT DICTIONARY (TCD), a novel Key-Value Knowledge Base (KB) that consists of task phrases as the key and the associated concepts involved in the solution process for each task as the value. The concepts contain not only items needed for the task but also concerned attributes (e.g., "thickness") and intermediate products during the process (e.g., "dough" appears during Making Rice Noodles). In our work, as a proof of concept, we automatically construct the knowledge base from WikiHow[1], one of the most used and actively updated How-to websites.

Once we construct a TCD, it can be potentially used for solving the problem of script completion in many ways. In this paper, we focus on exploring how to leverage it to acquire task-specific relevant concepts for enriching the task representation used in a baseline model for script completion. Specifically, inspired by how humans use mind-map to construct solutions, we introduce a novel "concept prompting" framework that first acquires relevant concepts from TCD and then connects them with the input as a prompt for the language generator to complete the script (Figure 2). We further introduce two complementary ways of concept acquisition. The first method retrieves concepts of close neighbors from TCD. The second method involves a generative model pretrained on TCD that is capable of generating associated concepts for a given goal, which can generalize beyond similar cases. Discovering that the model often repeats historical steps and hallucinates actions and entities, we also developed a script-oriented contrastive learning approach to address these issues by constructing corresponding negative samples.

We perform experiments on a dataset constructed from WikiHow and find that our method gives a consistent improvement on both automatic and human evaluations in comparison to the baseline, demonstrating the benefit of TCD for script completion. Moreover, we find that concept acquisition quality has a large impact on this task.

To summarize, we make the following contributions:

- We introduce TETRIS, the task of Goal-Oriented Script Completion that asks a model to complete the task based on a goal, and optionally preference and history. The task poses interesting new challenges and enables new applications.
- We propose TASK CONCEPT DICTIONARY, a novel task-specific knowledge base that encodes knowledge about the associations between goals and concepts and a method composed of (1) extraction and integration of the relevant concepts from TCD, and (2) script learning-oriented contrastive learning objective to enhance the correctness of model-produced scripts.
- We experiment with the proposed methods on the WikiHow dataset and show that our methods outperform the state-of-the-art baseline models consistently, demonstrating the need for task-specific knowledge and the benefit of TCD for improving the performance of script completion.

2 Task Formulation

We define a Goal-Oriented Script to include three types of elements, **Goal**, **Preference**, and **Step**. A **Goal** represents the task desired to be completed. An optional **Preference** defines in what ways the goal should be completed. A **Step** is one procedural event toward solving the task, conditioned on a preference. In this work, given a goal $\mathcal{G}$, a preference $\mathcal{P}$, and a history of steps $\mathcal{H}$ already done, we aim to generate the remaining steps that accomplish the goal in natural language. An example is shown in Figure 1. For convenience, we use $\mathcal{G}_p$ to denote the goal conditioned on $\mathcal{P}$, and $\mathcal{I}$ to denote the entire input (consisting of $\mathcal{G}$, $\mathcal{P}$, and $\mathcal{H}$).

3 Task Concept Dictionary

As in virtually all NLP applications, a main technical challenge in solving the problem of script

---

[1] www.wikihow.com

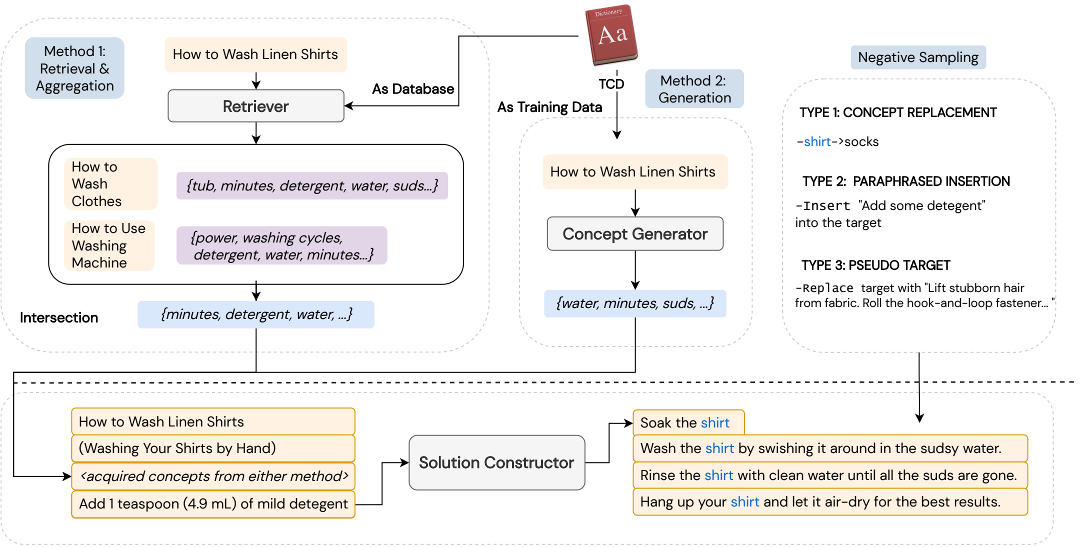


Figure 2: Our goal-oriented script completion framework. The top part shows the methods for automatically acquiring relevant concepts from the Task Concept Dictionary and negative sampling strategies. The bottom part shows the solution constructor, which uses the acquired concepts as an informative prompt to complete the script.

completion is how to acquire the relevant knowledge and leverage the knowledge to generate the remaining steps in a script. In our case, a critical question is: What kind of knowledge representation is likely helpful for solving the script completion problem? In our work, we make a hypothesis that creating task-concept association would help the model generalize better on script completion and propose Task Concept Dictionary (TCD). In TCD, each task key $\mathcal{G}$ from the collection of the keys $\mathcal{D}^{\mathcal{G}}$ (e.g., "How to Modify the Navigation System of an Acura") is mapped to a set of relevant concepts $\mathcal{D}^{\mathcal{G}}_{C}$ during its solution process. Each $c \in \mathcal{D}^{\mathcal{G}}_{C}$ can be associated with $\mathcal{D}^{\mathcal{G}}$ in four types of roles of a concept in that task: as preparation material needed for achieving the task (like ingredients for recipe), as attributes/aspects that need to be considered, as intermediate entities generated during the process, and others. In this work, we made a primal construction of TCD (TCD.v0) as a proof of concept, that is, the association types are not considered between $\mathcal{G}$ and $\mathcal{D}^{\mathcal{G}}_{C}$.

The design of TCD is motivated by the observation that humans can often form solutions through brainstorming and mind-mapping with knowledge about the tasks and relevant concepts involved in the solution process. Analogously, TCD is meant to encode such task-specific knowledge about associations between goals and concepts so as to assist downstream script learning applications. As will be shown in our experiment results, TCD is indeed beneficial for improving performance for script completion by supplying useful relevant concepts to a baseline model.

In general, TCD can be constructed by using any instructional content on the Web. In our experiments, to construct TCD.v0, we first collect all articles from WikiHow. We chose to use WikiHow as the seed corpus for its richness in user preference data. Each article consists of a goal, a list of preferences, and solution steps under each preference. Let $\mathcal{G}^{p_j}_{i}$ be the goal from an article i conditioned on the j-th preference, we then proceed to augment the set $\mathcal{D}^{\mathcal{G}}$ with a key $\mathcal{G}^{p_j}_{i}$, and augment $\mathcal{D}^{\mathcal{G}^{p_j}_{i}}_{C}$ with noun phrases (by using Spacy[2] tagger for extraction) in the solution steps under preference j. We create an association between goals and concepts if they co-occur in the same article. We show the statistics of TCD in Appendix A.1.

4 Method

In this section, we discuss our design of the method for TETRIS. The framework contains two parts, concept acquisition from TCD and solution construction. The first part uses TCD as an aid to acquire concepts relevant to the current input. Specifically, we introduce two different methods (Figure 2), one based on retrieval of closely related tasks and aggregation on the associated concepts, while the other involves training a concept generator on TCD. As shown in Section 5, both of

---

[2] `https://spacy.io/`

them have a positive impact on performance, yet they also introduce different benefits. The concepts fetched from TCD using either method form a concept prompt, which is combined with the input $\mathcal{I}$ (consisting of goal, preference, and history) as an augmented input $\mathcal{I}^*$. $\mathcal{I}^*$ is then fed into an encoder-decoder language model $\mathcal{M}_{\text{EncDec}}$ to complete the script. Below we describe the two methods for the generation of relevant concepts using TCD.

4.1 Concept Retrieval & Aggregation

The intuition of the retrieval method comes from observing how human refers to their past knowledge or the web for similar instructional sources (even though they might not address the need exactly) to achieve the current goal. For instance, if someone has experience in baking cheesecake and chocolate cake, it would be a breeze for them to make a strawberry cake by adapting the knowledge from their past experience. Building on the intuition, we propose to use retrieval as an interpretable way to retrieve similar tasks from the set of keys $\mathcal{D}^G$ in TCD and use their associated concepts to aid the downstream script completion task.

Retrieval. First we encode $\mathcal{G}_p$ (i.e., the preference-conditioned goal in $\mathcal{I}$) into a dense vector $\mathbf{e}_{\text{g}}$ with an encoder model. We similarly encode $\mathcal{D}^G$ into $\{\mathbf{e}_j\}_{j=1}^{|\mathcal{D}^G|}$. A cosine similarity score s_{gj} is computed between $\mathbf{e}_{\text{g}}$ and each $\mathbf{e}_{\text{j}}$. The top-K related tasks $\mathcal{N}^K$ are then retrieved based on the scores. We further obtain a set of concepts $\mathcal{C}_i$ for i-th task $\mathcal{N}^K$. In our experiment, we use SBERT (Reimers and Gurevych, 2019) (pretrained for semantic search) as the encoder. We use FAISS (Johnson et al., 2019), an efficient similarity search library, to perform top-k search.

Aggregation. The retrieved neighborhood of concepts, however, may sometimes introduce contextual noise (e.g., the grape and chocolate-related concepts are not useful in strawberry cake baking). To tackle this issue, we additionally perform operations on the retrieved concepts using the set intersection. The concept set for an input $\mathcal{I}$ is computed as $\mathcal{C}^s = \bigcap_{i=1}^{i=K} \mathcal{C}_i$. We finally map the set $\mathcal{C}^s$ into a list $\mathcal{C}$, neglecting ordering information.

4.2 Concept Generation

While the retrieval method allows an explicit concept acquisition process, it is limited by the width of TCD. For example, if one would like to complete a task from a domain that is not covered well by TCD, the retrieved neighbors can introduce more noise than help. To address this limitation of the retrieval method and enhance generalization, we propose to use language modeling as an alternative way of acquiring concepts. Specifically, inspired by the work (Bosselut et al., 2019), in which a concept is generated given an edge type, we propose to directly generate a set of concepts relevant to a given task and preference. We directly train the model on TCD, where the model is asked to generate the set $\mathcal{D}_C^t$ given each key t. In the inference stage, by feeding $\mathcal{G}_p$ into the trained concept generator, we can then obtain a list of concepts $\mathcal{C}$.

	F&E	F&B	C&V
# Train Samples	10600	2824	2214
# Dev Samples	3449	873	741
# Test Samples	3567	930	714
# Articles	2201	802	481
Avg # Tokens/Step	8.66	7.25	9.31
Avg # Steps/Article	10.42	8.66	10.41

Table 1: Statistics summarizing the WikiHow-based dataset for TETRIS, where F&E indicates Food and Entertainment, F&B indicates Finance and Business, and C&V indicates Cars and Other Vehicles

4.3 Solution Constructor

In this step we aim to encode both of the information from $\mathcal{C}$ (using either of the above methods) and $\mathcal{I}$ to generate the remaining steps that accomplish the goal specified in $\mathcal{I}$. The TCD has enabled us to enrich the representation by augmenting $\mathcal{I}$ with $\mathcal{C}$ and such an augmented representation can then be fed into any baseline script completion method to enable the baseline method to have (indirect) access to the knowledge encoded in TCD, thus improving accuracy of script completion.

Input Formation. An encoder-decoder model $\mathcal{M}_g$ is used for generation. In our experiment, we choose BART (Lewis et al., 2019) as the base model for its impressive performance on other NLP tasks (Liu et al., 2021; Lewis et al., 2020). Clearly, the framework accommodates any other models as well. Given the list of concepts $\mathcal{C}$ generated from either Section 4.1 or 4.2, we form the input $\mathcal{I}_f$ to the encoder as “<s>Goal (Preference) ### $\mathcal{C}$ ### </s>Step$_1$</s>Step$_2$...Step$_{|\mathcal{H}|}$</s>”, where <s> is a special token to represent the start of the sentence and </s> is for separation.

Model Generation. The input is fed through the standard tokenization, embedding mapping, and transformer to produce encoder hidden states. In

Method	BERTScore	BARTScore	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE 2
BART	86.76	-4.65	8.45	4.40	2.21	1.13	14.67	4.6
GPT2	86.75	-4.59	17.89	8.29	3.44	1.52	15.61	3.1
CRA-3	87.18	-4.41	22.59	10.95	4.86	2.10	18.07	4.9
CRA-1	86.74	-4.50	21.87	10.82	4.85	2.21	17.55	5.3
CG	86.86	-4.45	24.34	11.84	5.26	2.43	17.88	5.1
CG+SOCL	86.77	-4.47	**26.18**	**13.15**	**6.18**	**3.07**	18.11	**5.7**
CRA-2	**87.21**	**-4.37**	23.95	11.36	5.06	2.35	**18.65**	4.9
CRA-2+SOCL	87.19	-4.38	24.25	11.58	5.25	2.42	18.36	5.0

Method	BERTScore	BARTScore	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE 2
BART	87.87	-4.37	16.61	8.31	4.67	2.92	18.92	5.6
GPT2	88.06	-4.30	17.13	7.27	3.58	2.10	18.61	3.6
CRA-3	88.37	-4.28	14.73	6.95	3.77	2.36	19.17	4.6
CRA-1	87.66	-4.37	17.68	8.57	4.63	2.87	18.95	5.4
CG	88.35	**-4.17**	22.87	10.97	6.09	3.94	20.88	5.4
CG+SOCL	**88.44**	-4.18	**23.42**	**11.12**	**6.19**	**4.02**	**21.20**	5.7
CRA-2	88.28	-4.26	18.78	9.00	4.94	3.18	19.73	5.2
CRA-2+SOCL	88.42	-4.20	20.48	10.04	5.80	3.96	20.89	**6.3**

Method	BERTScore	BARTScore	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE 2
BART	87.95	-3.88	28.39	14.88	7.76	4.30	21.99	7.3
GPT2	87.88	-3.94	22.73	11.06	5.43	2.91	19.95	5.7
CRA-3	88.05	-3.89	26.43	14.18	7.62	4.32	21.39	7.4
CRA-1	87.94	**-3.85**	29.09	15.07	7.76	4.18	22.05	7.4
CG	87.84	-3.90	**29.69**	**15.70**	**8.27**	**4.60**	21.96	7.4
CG+SOCL	87.92	-3.87	27.97	14.66	7.81	4.42	21.93	**7.6**
CRA-2	88.06	**-3.85**	28.48	14.71	7.64	4.17	21.89	7.3
CRA-2+SOCL	**88.08**	**-3.85**	29.40	15.54	8.21	4.58	**22.41**	7.5

Table 2: Automatic metrics for evaluating language generation performance on the test set of Vehicles (top), Finance (middle), and Food (bottom). Our main models **CRA-2** and **CG** outperform the baseline consistently, demonstrating the effectiveness of our mechanism. We also include the performance of **CRA-1** and **CRA-3** to show that the performance varies as the number of neighbors changes. Moreover, we show that our contrastive learning objective (**SOCL**) almost always helps the model to have better performance.

the decoder end, the decoder hidden states (from the previous token) additionally attend to encoder hidden states to produce the next token. The model computes generation probability by taking the dot product between the decoder output and the tokens embeddings from the vocabulary.

Lastly, we use negative log-likelihood loss during training for each sample

$$\mathcal{L}_G = -\sum_{i=1}^{|S|} \log P\Big(s_i | s_{<i}, T\Big) \qquad (1)$$

where S denotes the tokens for the remaining steps.

4.4 Script-Oriented Contrastive Learning

To further improve the accuracy of the generated scripts, we design a contrastive learning framework that addresses deficiencies discovered in the model outputs from 4.1 and 4.2.

Negative Sampling In contrastive learning(Hu et al., 2022), hard negative samples are constructed to guide the model to better distinguish between the incorrect samples and the desired outcome. To construct such negatives, we gain insights from the model output, from which we found that the model elicits two typical types of erroneous behavior: (1) repetition of steps from the history and (2) hallucination of non-relevant actions/concepts for a given task (e.g., sago appeared in the process of making kimchi). Based on the observation, we propose the following strategies: (1) CONCEPT REPLACEMENT, where we randomly replace the concepts in the positive sample with concepts from other tasks in TCD under the same category (e.g., Food category in WikiHow), (2) PARAPHRASED INSERTION, where we paraphrase history steps and insert them into target steps, and (3) PSEUDO TARGETS, where we construct pseudo target by sampling steps from the same category and glue them into a sequence.

Contrastive Loss In computing the contrastive loss, we generate negative targets from the strategy above for each positive script target (composition discussed in the Appendix). To allow the model to distinguish between correct and incorrect script completions, we use the model's hidden states to compute a score to compute the correctness score for each sample, which is then used to compute the triplet loss (Schroff et al., 2015). More specifically, for each training sample

$$
\begin{aligned}
\mathcal{L}_{\mathrm{CL}} &= \sum_k \max\left(0, \phi + c_k^- - c^+\right), \\
c^+ &= \sigma\left(\operatorname{AvgPool}\left(\boldsymbol{W}_c \boldsymbol{H}^+ + \boldsymbol{b}_c\right)\right) \\
c_k^- &= \sigma\left(\operatorname{AvgPool}\left(\boldsymbol{W}_c \boldsymbol{H}_k^- + \boldsymbol{b}_c\right)\right)
\end{aligned}
$$

Where $\boldsymbol{H}^+$ and $\boldsymbol{H}_k^-$ indicate the decoder hidden states for the positive and k-th negative sample, σ is the sigmoid function, AvgPool is the average pooling function, $\boldsymbol{W}_c$ and $\boldsymbol{b}_c$ are learnable parameters.

4.5 Training Objective

We jointly optimize the model on cross-entropy loss from generation and triplet loss from contrastive learning

$$\mathcal{L} = \mathcal{L}_G + \beta \mathcal{L}_{CL}$$

where β is hyper-parameter.

5 Experiments

5.1 Dataset

We collect data from Food & Entertaining (abbreviated as Food), Finance & Business (abbreviated as Finance), and Cars & other Vehicles (abbreviated as Vehicles) categories of WikiHow, which have varying data scales and therefore allow comparison of models' generalization ability. Each article from WikiHow contains a goal, (optionally) several preferences for completing the goal, and steps under each direction. The details of the dataset are shown in table 1. More processing details are in Appendix A.1.

5.2 Implementation and Training Detail

Our model is implemented using Pytorch (Paszke et al., 2019) and Huggingface Transformers (Wolf et al., 2020) with BART-base as the base generator. The reproducibility and hyperparameter details can be found in Appendix A.2.

5.3 Compared Methods

We compare our framework and methods with the state-of-the-art text generation baselines **BART** (Lewis et al., 2019) and **GPT2** (Radford et al., 2019), which don't use any of our proposed methods. The solution constructor in our framework also uses BART. We use **CRA**-k to denote top-k neighbors in 4.1 and set k to be 1,2, or 3 in our experiment. We use **CG** to denote the concept generator in 4.2. Since TCD.v0 is also based on WikiHow as source data, we make some modifications to our methods in the experiment so that we can test the scenarios where the new unseen task cannot be exactly found in the knowledge base. For concept generation, we exclude the evaluation set related articles in the dataset from its training data. For **CRA**-k, when we retrieve relevant tasks from TCD for a given goal, we remove its own key from TCD. **SOCL** indicates that script-oriented contrastive learning is used during the training.

5.4 Automatic Evaluation

Evaluation Metrics We use both deep learning and n-gram-based evaluation strategies. The metrics include BERTScore (Zhang et al., 2019), BARTScore (Yuan et al., 2021), BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and ROUGE (Lin, 2004). Note that BARTScore computes the log-likelihood of producing the reference text given the generated text using a BART model pretrained on ParaBank2[3]. BERTScore computes an embedding matching-based score.

Results The test set results are shown in Table 2. We can see that the performance of our method variants (both CRA and CG) is consistently better than the baselines, empirically demonstrating the effectiveness of both concept acquisition methods and that the task-concept representation of the task-specific knowledge is helpful for downstream script learning. Moreover, while CG outperforms CRA-2 on all metrics in Finance, it is less clearly observed in the others, showing that the level of effectiveness also depends on the domain. The fact that CRA-2 is better than CRA-1 most of the time shows that the set operation in Section 4.1 does improve the retrieval quality for certain domains. Meanwhile, CRA-3 consistently performs worse than CRA-2, showing that setting $k = 3$ may have removed too much useful information. With contrastive learning, the model almost always gains enhancement in performance, showing the effectiveness of our sampling strategies; an exception is shown in the Food category for CG, where it shows positive im-

---

[3] `https://github.com/neulab/BARTScore`

Name	**Correctness ↑**	**Fluency ↑**	**Average Rank ↓**	**Best ↑**	**Worst ↓**
BART	1.72	2.57	3.57	4.13	72.73
CG	2.32	3.06	2.65	9.09	13.22
CRA-2	**2.52**	**3.15**	**2.34**	**14.88**	**9.92**
Gold-Concepts	**3.40**	**3.55**	**1.44**	**71.90**	**4.13**

Table 3: Human evaluation results for the WikiHow dataset on all three categories combined. Scripts completion outputs (from automatic analysis) from each model are presented to human judges, who are asked to rate on Correctness (i.e., at what level does the generated solve the problem) and Fluency of the output. We also report the average rank and percentage of the time that each model output is chosen as best or worst, after asking each judge to rank outputs from different methods for each input

pact only on half of the metrics, and this might be caused by that the CG itself is effective enough and contrastive learning introduces more noise than help. We also show in Appendix that adding SOCL alone is already effective. Lastly, we observe that the Vehicle and Finance categories contain much less training data than Food, yet the performance improvement is more notable, demonstrating the potential of the methods in low-resource settings.

5.5 Human Evaluation

Evaluation Metrics & Process As mentioned in (Zhang, 2022), automatic metrics do not correlate well with human judgments on script learning tasks due to the diversity of potential solutions. As a complement to automatic metrics, we also recruited five volunteers (who are not authors) and conducted the human evaluation. The volunteers were Master/Ph.D. students with enough background knowledge to rate output. We gave each candidate a quiz composed of 20 random samples from the dataset as filtering to see if the annotator can give scores close to the authors'. We made sure that each annotator understands the assigned data samples during evaluation. In our evaluation process, we compared the output generated by BART, CRA-2, and CG with SOCL, and Gold-Concepts (a variant where concepts come groundtruth in TCD). We presented the input and the corresponding generated outputs from each method to human judges and asked them to rate the Correctness and Fluency of the output, both on a scale from 0 to 4. Correctness (or usefulness) was defined as the level of confidence that the generated steps successfully complete the specified goal and preference. We additionally asked the judges to rank the outputs according to their overall preference and their rating on Correctness & Fluency; after this procedure, we reported the average rank and the percentage of the time that each model output is chosen as the best or worst.

Goal	**How to Store Peaches**
Preference	Keeping Peaches in the Fridge
History	Rinse the peaches to clean off any dirt or debris. Dry the peaches with clean paper towels or a clean hand towel.
BART	<eos>
CG	[*peach, paper bag, store peach, container, term storage*] **Store peaches in the fridge** for up to 3 months.
CRA-2	[*peach, freezer, container, half*] **Store the peaches in an airtight container** in the freezer for up to 3 months.
Reference	Place whole, uncut peaches in the fridge on their own or in a plastic bag. ... **Store sliced peaches in an airtight container** for 1-2 days.

Table 4: Example input and output from the Food category. The noun phrases contained in square brackets in CG and CRA-2 indicate the acquired concepts from TCD. <eos> indicates an empty completion (i.e., the history is believed to contain the complete script). With fridge storage-related concepts, our models correctly produce the storing step.

Results. 121 samples are randomly selected from automatic analysis for human evaluation, where 40 come from the Vehicle category, 40 come from Finance, and 41 from Food. We present the human evaluation results in Table 3. The results align with automatic metrics roughly in general. The Gold-Concept variant is usually the best since it has access to ground-truth noun phrases from the reference. CRA-2 often generates more helpful solutions for users. The outputs from BART are worse than others most of the time, confirming the belief that using acquired task knowledge as prompts can help the model to generate better solutions on average. The Fluency level for all methods is rated to be of acceptable quality most of the time, likely due to the strong generalization capability of the pretrained language model.

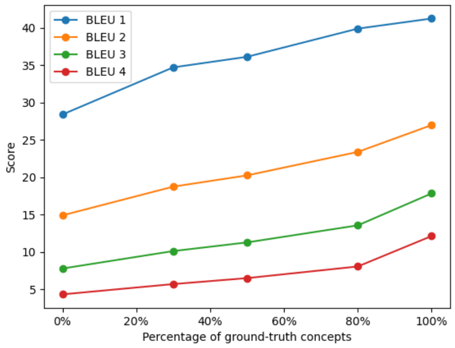


Figure 3: The plot shows that the automatic performance increases as the amount of ground-truth concepts increases, showing that deriving the right task-specific concepts can make the model generate better scripts.

5.6 Impact of Retrieval Quality

One question to be asked is how impactful it is to derive a concept prompt close to the groundtruth (i.e., concepts from TCD by using the current goal as the search key). To answer the question, we randomly draw concepts from the groundtruth at different thresholds and use each of them as a prompt to train a baseline model. From our result shown in Figure 3, we can see that the performance monotonically increases as the number of concepts covering the ground-truth concepts increases, showing the consistent benefit of using the concepts in the right context; this also demonstrates the promising direction of our methods and the significance of developing a concept deriver better at understanding task context.

5.7 Qualitative Analysis

We present an example of generated outputs by different methods in Table 4. Additional examples are available in Appendix A.3. We compare BART, CG, and CRA-2. From the generated output, we can see that BART ends the script immediately, missing the step of storing, likely due to the lack of access to knowledge about storage. On the other hand, with the acquisition from TCD, our methods can reach more contextual knowledge for the current task. Specifically, both methods are able to access storage-related concepts such as term storage, container, and freezer. As a result, the outputs match with the reference more closely. While the alignment with the reference isn't perfect (e.g., peaches are additionally placed in a bag in the reference), to most human judges, the script completion clearly achieves the goal already; this further demonstrates the difficulty of evaluating script completion.

6 Related Work

Script Learning: Scripts (Schank and Abelson, 2013; Feigenbaum et al., 1981; Yang et al., 2021; Zhang et al., 2020b,a) refers to the knowledge of stereotypical event sequences which human is constantly experiencing and repeating every day. One branch of works in script learning focuses on distilling *narrative* scripts from news or stories (Chambers and Jurafsky, 2008; Jans et al., 2012; Lee and Goldwasser, 2019), where the scripts are not goal-oriented. One of the most recent tasks in *narrative* script modeling is Multiple-Choice Narrative Cloze Test (Granroth-Wilding and Clark, 2016), where an event is removed from the chain and the model is asked to predict which event from the choices fills the blank. The other line of work, which is more closely related to our work, centers around *procedural* scripts, where a sequence of events happened often to achieve a goal. Recent work has introduced the task of constructing the entire script given a goal (Lyu et al., 2021) or choosing 1 out of 4 candidates' steps that most likely help achieve a goal (Zhang et al., 2020c). In this work, we propose a more general script learning setup that considers additionally the user preference and history, improving the generalization of models in script learning.

Our method is related to Retrieval-augmented text generation (Li et al., 2022; Zhang et al., 2022; Wu et al., 2021), a paradigm that aims to combine deep learning models with retrieval methods for text generation and has gained significant attention in recent years. For example in (Rubin et al., 2022), the author retrieves similar training examples as prompt to the current input. In our work, we are the first to introduce a framework that retrieves concepts from the concept dictionary as prompts for script learning. Apart from acquiring concepts, we also introduce a way to cancel contextual noise from neighbors by set intersections for better prompt quality. Furthermore, our method is related to contrastive learning (Wang et al., 2022; Hu et al., 2022; An et al., 2022), a line of self-supervised learning methods that improve representation learning by compacting positive samples while contrasting them with negative samples (Cao and Wang, 2021). Previous methods used negative sampling approaches such as replacing the words with synonyms and shuffling target sentences. In our work, we introduced a new script-oriented contrastive learning objective to address script-specific issues and enhance the quality of generated scripts.

7 Conclusion

In this work, we propose TETRIS, the new task of Goal-Oriented Script Completion, which allows a model to produce the remaining steps given a user-specified goal, preference, and history of steps. We also present TASK CONCEPT DICTIONARY (TCD), a knowledge base representing task and concept association, to enable knowledge-based methods for the task. We introduce different methods to acquire concepts as prompts for the downstream text generator. We also introduce a contrastive learning strategy for script learning. The methods present consistently better performance on both automatic and human evaluation, clearly demonstrating the benefit of TCD for improving the performance of script completion. The qualitative analysis further shows that the task-specific knowledge can indeed benefit goal-oriented script learning tasks by feeding relevant knowledge about task completion. Future work could explore how to improve the acquisition quality from TCD and applications of TCD on task-oriented dialog systems.

Limitations

While TCD paired with concept acquisition methods can aid downstream script learning tasks, it doesn't consider the inclusion of actions of each step event, which can potentially benefit the script learning tasks. A possible direction is to extend the design of TCD and the concept prompt to include the semantics of actions and their orders.

Meanwhile, the concepts extracted by our method do not overlap with the ground-truth concepts (i.e., the set of concepts that appear in the reference) very well (e.g., <20% in Jaccard Index). The gap in performance between our methods and the Gold-Concept variant shows that improving the concept derivation quality might be the next step.

Furthermore, because our dataset is constructed from the English version of WikiHow, the benefits of our methods shown in the experiments are only empirically proved to work for English. We plan to further test our methods in multiple languages.

Ethics Statement

The study aims to extend deep learning-based models on the ability to generalize scripts under different user contexts. The script learning models introduced in the work can potentially be helpful for task-oriented dialog systems to suggest solutions to users. The dataset on which we base our experiments is constructed automatically from the publicly available website WikiHow. Since the website is primarily crowdsourced, the models trained on the data might incur subjective bias. During the human evaluation phase of the experiment, all involved human judges participated voluntarily and received decent payment.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive suggestions on our work and thank the raters for their help on manual evaluation. This research is based upon work supported in part by U.S. DARPA KAIROS Program No. FA8750-19-2-1004. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the of ficial policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Chenxin An, Jiangtao Feng, Kai Lv, Lingpeng Kong, Xipeng Qiu, and Xuanjing Huang. 2022. Cont: Contrastive neural text generation.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction.

Shuyang Cao and Lu Wang. 2021. Cliff: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio. Association for Computational Linguistics.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *Acm Sigkdd Explorations Newsletter*, 19(2):25–35.

Christopher J Conti, Aparna S Varde, and Weitian Wang. 2020. Robot action planning by commonsense knowledge in human-robot collaborative tasks. In *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pages 1–7. IEEE.

Edward A Feigenbaum, Avron Barr, and Paul R Cohen. 1981. The handbook of artificial intelligence.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Ruocheng Guo, Lu Cheng, Jundong Li, P Richard Hahn, and Huan Liu. 2020. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys (CSUR)*, 53(4):1–37.

Zhe Hu, Hou Pong Chan, Jiachen Liu, Xinyan Xiao, Hua Wu, and Lifu Huang. 2022. Planet: Dynamic content planning in autoregressive transformers for long-form text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2288–2305, Dublin, Ireland. Association for Computational Linguistics.

Bram Jans, Steven Bethard, Ivan Vulic, and Marie-Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 336–344. ACL; East Stroudsburg, PA.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

I-Ta Lee and Dan Goldwasser. 2019. Multi-relational script learning for discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4214–4226, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Huayang Li, Yixuan Su, Deng Cai, Yan Wang, and Lemao Liu. 2022. A survey on retrieval-augmented text generation.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and S Yu Philip. 2021. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6418–6425.

Qing Lyu, Li Zhang, and Chris Callison-Burch. 2021. Goal-oriented script construction. *arXiv preprint arXiv:2107.13189*.

MG Mohanan and Ambuja Salgoankar. 2018. A survey of robotic motion planning in dynamic environments. *Robotics and Autonomous Systems*, 100:171–185.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning.

Keisuke Sakaguchi, Chandra Bhagavatula, Ronan Le Bras, Niket Tandon, Peter Clark, and Yejin Choi. 2021. proscript: Partially ordered scripts generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2138–2149, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Roger C Schank and Robert P Abelson. 2013. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4281–4294, Dublin, Ireland. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zeqiu Wu, Michel Galley, Chris Brockett, Yizhe Zhang, Xiang Gao, Chris Quirk, Rik Koncel-Kedziorski, Jianfeng Gao, Hannaneh Hajishirzi, Mari Ostendorf, and Bill Dolan. 2021. A controllable model of grounded response generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14085–14093.

Yue Yang, Artemis Panagopoulou, Qing Lyu, Li Zhang, Mark Yatskar, and Chris Callison-Burch. 2021. Visual goal-step inference using wikihow. *arXiv preprint arXiv:2104.05845*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Li Zhang. 2022. Reasoning about procedures with natural language processing: A tutorial.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020a. Intent detection with wikihow. *arXiv preprint arXiv:2009.05781*.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020b. Reasoning about goals, steps, and temporal ordering with wikihow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020c. Reasoning about goals, steps, and temporal ordering with wikihow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yizhe Zhang, Siqi Sun, Xiang Gao, Yuwei Fang, Chris Brockett, Michel Galley, Jianfeng Gao, and Bill Dolan. 2022. Retgen: A joint framework for retrieval and grounded text generation modeling.

A Appendix

A.1 Dataset & TCD

The dataset statistics are shown in Table 1. The raw corpus comes from the 07/20/21 snapshot of WikiHow. We filter out unordered scripts by using both the classification results from (Lyu et al., 2021) and the WikiHow section type. We perform a 6:2:2 split on the articles to create a train, development, and test set. We create a history $\mathcal{H}$ in data samples for TETRIS by randomly splitting a sequence of steps under each preference into two halves. For TCD, we have 206621 keys in total and 10.37 concepts per key on average.

A.2 Implementation Details

We implement the models using the 4.8.2 version of Huggingface Transformer library[5](Wolf et al., 2020). We use the Oct 1, 2021 commit version of the BART-base model (139M parameters) from Huggingface[6]. The contrastive learning variants has 140M parameters. For SBERT in Section 4.1, we use the all-mpnet-base-v2 checkpoint from sentence transformer library [7]. We use Huggingface datasets[8] for automatic evaluation metrics. The BART Score comes from the author's repository[9] and we used the one trained on ParaBank2. The hyperparameters for the experiment (non-contrastive

---

[5] `https://github.com/huggingface/transformers`

[6] `https://huggingface.co/facebook/bart-base/commit/ea0107eec489da9597e9eefd095eb691fcc7b4f9`

[7] `https://www.sbert.net/`

[8] `https://github.com/huggingface/datasets`

[9] `https://github.com/neulab/BARTScore`

Method	BERTScore	BARTScore	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE 2
BART	**87.87**	-4.37	16.61	8.31	4.67	2.92	18.92	5.6
BART+SOCL	87.725	**-4.349**	**19.478**	**9.441**	**5.216**	**3.236**	**19.198**	**5.73**

Table 5: On the Finance dataset, our result shows that SOCL alone already brings consistent benefits to the performance.

Table 6: Hyperparameters for non-contrastive learning (CL) models (some are introduced and changed for CL experiment). The ones below the mid-line are generation related. Batch size is changed for contrastive learning.

Name	Value
seed	42
learning rate	3e-5
batch size	16
weight decay	5e-4
RAdam epsilon	1e-8
RAdam betas	(0.9, 0.999)
scheduler	linear
warmup ratio (for scheduler)	0.06
number of epochs	25
metric for early stop	SacreBLEU[4]
patience (for early stop)	15
length penalty	1.2
max length	511
min length	2
beam search size	5

learning) are shown in Table 6 (applied to all models) and the ones not listed in the table are set to be default values from the transformer library. We use RAdam (Liu et al., 2019) as the optimizer. We perform hyperparameter search on batch size from {16, 32}, pretrained language model learning rate from {2e-5, 3e-5, 4e-5}, downstream learning rate for contrastive learning from {1e-3, 5e-4, 1e-4, 3e-5}, negative sample composition from type A “1 sample from each of type 1 2, filling empty case with type 3” and type BCD “2 samples from the type combination (2 3/1 2/1 3), filling empty case with empty string”, and the number of epochs from {8, 15, 25} ({28, 32} for contrastive learning). For contrastive learning experiment negative sampling, we use type A for Vehicles and Food and type B for Finance. We perform our experiments on 40 GB A100 and 32 GB V100 GPU. For contrastive learning, we use batch size of 28 for Vehicles, 32 for Food, and 32 and 28 for CRA and CG respectively in Finance. The downstream learning rates are respectively 1e-3, 1e-3, 5e-4 for CRA-2+SOCL and 3e-5, 1e-4, 1e-3 for CG+SOCL in the datasets of Vehicles, Finance, and Food. We use 0.5 for the margin in the triplet loss function and 0.3 for the β in the training loss. The experiments can take up to 10 hours.

A.3 Additional Model Outputs and Analysis

We present examples in Table 7, 8, and 9, in addition to Section 5.7. From Table 7, we can see that while BART ends the script immediately after finding an agency, our methods are able to provide detail on the interaction with the agency. This shows that training with the acquired concepts as prompts allows the model to gain more task-specific contextual knowledge, which can be potentially shared in the token embeddings of concepts. In Table 8, the user preference in the example contains some ambiguity (i.e., it could mean making just the dough, or the complete process of making the pasta). While BART is missing the further processing on the *dough*, CG and CRA-2 are able to not only include detailed and reasonable steps that process the *dough* but also some information on what to do after the dough is being made, showing the usefulness of incorporating task knowledge. The reason for the improvement could be that the prompts of our method (all including the word *dough*) allow the model to be attentive to the processing of $dough$. In Table 9, the methods without gold concepts all believe the script has ended, which is a reasonable output. Yet they would be low on automatic generation metrics because of not matching reference well (which contains steps not necessarily needed to complete the goal). This shows the difficulty of automatically evaluating script generation since there can be many ways to solve a task and can be different in the amount of detail.

A.4 Discussions

While we show in the experiment that TCD can be used in the script completion task in our work to achieve better performance, it can also naturally be applied to other domains such as task-oriented

Goal	**How to Report Pet Store Neglect**
Preference	Reporting to Local Law Enforcement
History	Document the situation as extensively as possible Check your state's animal welfare law Find out which agency is responsible for enforcing animal laws
BART	Finished
CG	[dog] Report the situation to local law enforcement Follow up with any follow-up questions Finished
CRA-2	[police, report, animal control, authority] Contact local law enforcement agencies if necessary Follow up with any follow-up questions Finished
Gold-Concept	[situation, state, animal welfare law, agency, animal law, documentation, record, report, law enforcement] Gather documentation and record of the report to law enforcement Follow up with your report if necessary Finished
Reference	Gather your documentation and other records File your report with law enforcement Follow up on your report Finished

Table 7: Example output among different methods (Finance and Business)

dialog, event causal identification, and causal inference. Furthermore, since the two different concept acquisition methods in the paper contain different advantages, it would be an interesting task to devise a method that has both benefits of being explicit in the acquisition process and being parametrized.

One natural question to ask is why we represent the KB as concept-based instead of directly using paragraphs to record each goal. While How-to websites contain human-curated solutions, the way of annotation introduces canonicalization problems since different people might have 1. different orders of completing the same task and 2. different levels of detail in the expression; on the other hand, using concepts representation helps alleviate the problem since the core concepts (e.g., "butter", "temperature", "time" in "Baking a Cake") almost always participates regardless of how they are organized in each person's memory. Furthermore, TCD introduces an interesting function of automated self-enrichment. In hierarchically related goals, such as "how to bathe a dog" and "how to bathe a cat", we can use **WordNet Hierarchy** and **Concepts Intersection** to automatically derive new nodes. We leave the experimentation to future work.

Another important problem to be tackled is that, while human evaluation is more reliable than automatic evaluation, it is also much more costly. As mentioned in (Zhang, 2022), automatic metrics do not align with that of humans very well. One promising direction is to create an automatic metric that does not evaluate script generation methods only based on matching with the reference (since many equivalently valid references can exist).

Goal	**How to Make Macaroni Pasta**
Preference	Making Pasta Dough at Home
History	Mix the dry ingredients Create a hole or well in the dry ingredients Add three large eggs Whisk the eggs with a fork Place the dough on a clean counter
BART	Bake the pasta Serve the pasta while it is still warm Finished
CG	[flour, salt, bowl, butter, milk, egg, flour mixture, dough, surface] Roll the dough out into a rectangle Cut the dough into 1/4 inch (0.64 cm) circles Transfer the dough to a plate Repeat the process with the rest of the dough Cook the pasta Place the pasta on a plate to warm up Serve Finished
CRA-2	[dough] Roll the dough into a ball Repeat the process with the rest of the dough Transfer the dough to a baking sheet Bake the pasta Serve or refrigerate Finished
Gold-Concept	[ingredient, hole, egg, fork, dough, counter, surface, bowl] Place the dough on a floured surface Transfer the dough to a serving bowl Serve Finished
Reference	Fold the dough Look for a smooth surface Place the dough in a clean bowl to rest Finished

Table 8: Example output among different methods (Food and Entertaining category).

Goal	**How to Track a Flight in Progress**
Preference	Via Online Websites
History	Gather the information about the flight: Find a flight-tracking website to find out the information about your flight Enter your flight information Watch for plugins that you could need to add to your browsers functionality to run the website Watch a map come up on your screen showing the route the plane has taken, as well as the current location of the plane and the expected route ahead
BART	Finished
CG	[flyer, airline, website, flight, status, status update] Finished
CRA-2	[airline, flight, flight number] Finished
Gold-Concept	[information, flight, tracking website, flight information, plugin, browser functionality, website, map, screen, route, plane, location, auto, update feature, way, site, ability] Use the auto-update feature on the way out of the site Allow the site to continue running while you're not using it Check to see if the update feature is working properly
Reference	Figure out if the website you chose has an auto-update feature Look for ways to zoom in on the plane, if this site allows that ability Finished

Table 9: Example output among different methods (Cars & Vehicles category)

DeepMaven: Deep Question Answering on Long-Distance Movie/TV Show Videos with Multimedia Knowledge Extraction and Synthesis

Yi Fung[1]*, **Han Wang**[2], **Tong Wang**[2],
Ali Kebarighotbi[2], **Mohit Bansal**[2], **Heng Ji**[2], **Prem Natarajan**[2]
[1]University of Illinois at Urbana Champaign, [2]Amazon Alexa
`yifung2@illinois.edu`
`{wnghn,tonwng,alikeba,mobansal,jihj,premknat}@amazon.com`

Abstract

Long video content understanding poses a challenging set of research questions as it involves long-distance, cross-media reasoning and knowledge awareness. In this paper, we present a new benchmark for this problem domain, targeting the task of deep movie/TV question answering (QA) beyond previous work's focus on simple plot summary and short video moment settings. We define several baselines based on direct retrieval of relevant context for long-distance movie QA. Observing that real-world QAs may require higher-order multi-hop inferences, we further propose a novel framework, called the **DeepMaven**, which extracts events, entities, and relations from the rich multimedia content in long videos to preconstruct movie knowledge graphs (movieKGs), and at the time of QA inference, complements general semantics with structured knowledge for more effective information retrieval and knowledge reasoning. We also introduce our recently collected *DeepMovieQA* dataset, including 1,000 long-form QA pairs from 41 hours of videos, to serve as a new and useful resource for future work. Empirical results show the DeepMaven performs competitively for both the new *DeepMovieQA* and the pre-existing MovieQA dataset.[1]

1 Introduction

Our world tells an evolving story of people, objects, and their interactions. This storytelling may exist in various forms, from textual summaries and spoken dialogues, to accompanying images and videos. Because of its dynamically evolving and multi-media nature, long-distance video question answering on movies/TV shows provides a useful setting for studying the computational understanding of interconnected stories and events that aligns closely with real-world application scenarios. Yet, current intelligent systems still struggle with adequately processing the rich multimedia content in long videos and fail to answer many common inquiries that humans are interested in. If we ask a virtual assistant about the relationship between two main characters in a well-known movie/TV show, it likely defaults to some null response such as *"hmm... I don't know this one"*.

The research challenge of long video content understanding and question answering frameworks stems from the need to support information probing on certain specific details over a large multimedia context space. Propagating information across long-distance has been a well-known challenge due to the vanishing gradient and memory loss problem (Hochreiter, 1998). Long videos also can not fit its entire data in-memory typically for end-to-end feature extraction and neural network training. Meanwhile, retrieval-based methodologies, which first find sections in the video relevant to a query through semantic cues from the corresponding dialogues and/or frames, and then return the relevant textual dialogue and/or visual frame context for question answering, tend to narrowly isolate media instances semantically similar to the query inputs. Query inputs are short and succinct in nature, so simply considering data points that match the query semantics will overlook the larger picture behind the selected media instances *i.e.* how they relate to each other as well as to other relevant media components that are initially missed out from retrieval as they involve additional reasoning.

For example, a question about "the hobby interests of *Midge's husband"* in TV show "The Marvelous Mrs. Maisel" would likely be missed by text retrieval if the dialogue mostly uses the name of Midge's husband - *Joel*. A question about "*Midge* and *Joel*'s marriage breakdown" not only needs to identify the point of confrontation between *Midge* and *Joel*, but also other *past*, *concurrent*, or *future* events connected to these two person entities,

---

*Work done as an intern at Amazon Alexa AI.

[1] See `https://www.amazon.science/publications` for update of information about code and resources.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3041–3051
May 2-6, 2023

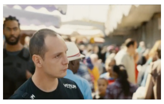
Category	**Question (Q)**	**Natural Language Answer (A)**	**Visual Evidence**
Direct 41% (long-distance)	Who bailed Midge out of jail after she was arrested for public nudity? –“The Marvelous Mrs. Maisel"	Susie Meyerson, the manager of the Gaslight, bailed Midge out.	
Cross-Media 33% (text+visual)	Why does Joel punch someone at Gaslight? –“The Marvelous Mrs. Maisel"	That person insulted Maisel during her act. Joel punches him and asserts that Maisel is good.	
Multi-Scene 11% ($\geq$ 2 scene, tot. min. $>$ 1)	How did Midge and Joel’s marriage fall apart? –“The Marvelous Mrs. Maisel"	Joel has been cheating on Midge, with his secretary Penny. One day, Joel got angry at Midge for giving suggestions that ended in a failed stand-up comedy performance for him. He packed up and left Midge.	
Multi-Hop 11% (higher reasoning through missing links/info)	What does the Mayor oversee on the market? –“Les Miserables"	The mayor oversees which vendors are allowed on the market, what they are allowed to sell, how big their space is and what kind of stands should be added to the market. He charges the vendors his fees and have the police talk to vendors who act up.	
Background Knowledge 10% (facts, Wikipedia)	In real history, what ultimately happened to Malcolm X’s friend, singer Sam Cooke? –“One Night in Miami"	He was shot to death by a motel manager for attempting to molest a woman, but his death was controversial and involved conspiracy theories.	

Table 1: Types of data annotated in our new ***DeepMovieQA*** benchmark, along with their statistics. The category pertains to the critical reasoning used in understanding the question, picking up relevant details from textual and/or visual media in the movie/TV show, and deriving to an answer.

Midge and Joel, relevant to their marriage dynamics, such as infidelity, pent-up frustration, and aftermaths. Previous work tends to focus on very high level questions with answers coming only from text summary, such as whether a character is good/bad, or focus on very superficial questions such as what object is behind this person, whereas we want to enable more interesting QAs such as from mimicking MovieClub conversations.

So in this work, we define a new task of deep movie question answering (**DeepMovieQA**), which gauges the content understanding of long movie/TV show videos that align with generic human inquiries. In contrast to previous work, the DeepMovieQA task consists of questions that naturally arise from watching the full length of a movie/TV show asset (as opposed to plot summary or short minute-long clips), and involve a more challenging set of reasoning summarized in Table 1, which make use of many different data modalities: plot summary, textual dialogues, visual frames, and background knowledge. DeepMovieQA additionally involves localizing the time frames that provide evidence for the question answering.

To tackle DeepMovieQA, we propose a novel framework, ***DeepMaven*** , which extracts context from multimedia video asset semantically relevant to the queries and generates a coherent answer conditioned on the retrieved context using a transformer-based backbone, with added benefits of transparency and explainability through this two-stage process. Observing that the new DeepMovieQA task involves more challenging types of reasoning, we further leverage a structured approach to incorporate wider context and better handle multi-hop queries, such as bridging the connection between the mention of *Midge’s husband* in a query and the canonical *Joel* entity.

Our key novel contributions can be summarized as follows: 1) we present a new research task of **DeepMovieQA** in the long-distance multimedia video understanding domain that involves reasoning beyond surface-level understanding of short moments and summaries; 2) we present **DEEPMAVEN**, a novel multimedia approach that leverages complementary information from (local) dialogue passages and visual frames and (wider-context) movieKG subgraph to guide deeper content understanding and question answering, achieving 10+% absolute gain in answer extraction and relative gain in answer generation over single modality baselines; and finally, 3) we contribute a DeepMovieQA dataset that includes 1000 QA annotations, to serve as a new resource for studying and evaluating this challenging but exciting task.

Benchmark/ Dataset	*Total Hrs*	*# Min. per Video Clip*	*# Tokens in Answer*
MovieQA (Tapaswi et al., 2016)	280	-	5.6
TVQA (Lei et al., 2018)	461	1.3	5.1
TVQA+ (Lei et al., 2020a)	461	1.3	5.1
TVR (Lei et al., 2020b)	461	1.3	-
KnowIT VQA (Garcia et al., 2020)	69	0.3	4.5
DeepMovieQA (ours)	**41**	**51.7**	**26.5**

Table 2: Comparison of DeepMovieQA with respect to pre-existing datasets in the movie/TV domain. The # of mins and # of tokens listed are averaged numbers.

Video Asset	*# Eps*	*# Min.*
Manchester by the Sea (MBTS)	-	137
Without Remorse (WORE)	-	109
One Night in Miami (ONIM)	-	114
Les Miserables (LMIS)	-	105
Marvelous Mrs. Maisel (MRSM), S1	8	419
Jack Ryan (JKRY), S1	8	400
The Boys (TBYZ), S1	8	361
Transparent (TP), S1	10	287
The Family Man (FMAN), S1	10	448
Total	**-**	**2480**

Table 3: The movies/TV shows in DeepMovieQA.

2 DeepMovieQA Data Collection

Pre-existing video/movie/TV show QA datasets fall into two extremes of either focusing on too narrow and minor scene-specific details such as *an object on the couch* that is irrelevant to the storyline development (Lei et al., 2018), or overly brief and simple details from summaries (Tapaswi et al., 2016) that align poorly with real-world needs such as *the year in which Hook's squad is sent to Belfast*, which is written directly in the short plot summary. In comparison, we aim for QAs closer to the conversations in movie clubs or what we talk about when we step out of the movie theatre i.e. the detailed yet crucial information to main event developments. This means we want question annotations that involve longer context of video clips and answer length, compared to prior work as reported in Table 2.

With this motivation in mind, we collect our dataset, ***DeepMovieQA***, on 4 movies and 5 TV shows listed in Table 3, chosen based on genre diversity. We instruct annotators to come up with QA pairs that arise naturally as they watch the movie/TV shows, through two rounds. In the initial pass, we provide ten annotators movie/TV show assets split into 20 minute video clips for focused attention on detailed plot content. In the second pass, we ask the annotators to re-watch the entire movie/TV show asset, review QAs previously annotated, and come up with interesting QAs that involve piecing information together from several scenes or sources – such as discussion of themes, messages, empathetic reactions[2], and societal background. This two-pass QA labeling process allows annotators to become more familiar with the movie/TV show content and come up with challenging QAs that better align with genuine human audience interests. ***DeepMovieQA*** provides the first long-distance, multi-scene QA benchmark in the multimedia setting, with long-form answers labeled to promote elaborative movie QA model capabilities.

3 Methodology

3.1 Direct Retrieval from Source Data

Given a movie or TV show $V = \{V_f, V_{dial}, V_S\}$, consisting of the video frames, dialogues, and summary, and a question, q, a natural first step is to select relevant context through semantic matching of local features. On the text side, we draw inspirations from previous research (Qu et al., 2020; Mossad et al., 2020) to encode the query, q, and candidate passages V_{dial_i}, truncated from every $n = 5$ utterance exchange (a tweakable hyperparameter), using $BERT$ (Devlin et al., 2019) , which are expressive bidirectional transformers for capturing latent language representations, followed by two separate linear layers, ϕ_q and ϕ_p. Then, we compute the retriever matching score through the cosine similarity of the encoded representations.

$$h_q = \phi_q(ReLU(BERT(q)))$$

$$h_p = \phi_p(ReLU(BERT(V_{dial_i})))$$

$$s_{q,V_{dial_i}} = cos\_sim(h_q, h_p)$$

For visual retrieval, we utilize the powerful multimedia CLiP encoder release (Radford et al., 2021), which consists of a textual encoder component $CLiP_t$ based on GPT (Radford et al., 2019) and a visual encoder component $CLiP_v$ based on ViT

---

[2] Similar to `https://teachwithmovies.org/discussion-questions-for-use-with-any-film-that-is-a-work-of-fiction/`

(Dosovitskiy et al., 2020), pretrained through contrastive learning on 400 million image caption pairs. We compare for semantic proximity between q, and a set of candidate visual frames, V_{f_i} through cosine similarity, similar to text retrieval. But note that as each candidate V_{f_i} is actually a set of visual frames corresponding to the time frame under V_{dial_i}, we take the mean of encoded image representations as the overall feature for V_{f_i}.

$$h_q = CLiP_t(q)$$

$$h_{V_{f_i}} = \frac{1}{|V_{f_i}|} \sum_{j \in 1..|V_{f_i}|} CLiP_v(V_{f_j})$$

$$s_{q,V_{f_i}} = cos\_sim(h_q, h_{V_{f_i}})$$

As dialogue exchanges and visual scenes may contain cues that complement each other in signaling whether a section of the video is relevant to the query, we utilize a linear combination of the textual and visual retrieval scores for multimedia retrieval:

$$s_{q,(V_{f_i},V_{dial_i})} = a * s_{q,V_{dial_i}} + b * s_{q,V_{f_i}}$$

We select the V_{dial_i} and/or V_{f_i} with top k semantic matching score for query q as the relevant context.

3.2 Retrieval from Structured Knowledge

Yet, structured knowledge provides benefits for long-distance and multi-hop information since events and interactions at separate time points (see Table 1 for examples) may then be directly connected through grounded nodes such as character entities.

Multimedia KG Construction To incorporate structured knowledge, we pre-construct *movieKG* using the open-source IE pipeline from (Wen et al., 2021) to extract events/entities/ relations (Lin et al., 2020) from movie summary and dialogue, link entities to background knowledge base (Pan et al., 2015) where applicable, and perform event/entity coreference resolution (Lai et al., 2021). This leads to an initial sparse KG, in which the nodes consist of events ($\mathcal{N}_v$) and entities ($\mathcal{N}_n$) while the edges consist of argument roles and relations (E_r), following a pre-defined ontology which inadequately covers the open-domain in diverse film genres, so we augment ontological-guided IE with Abstract Meaning Representation (AMR) parsing (Fernandez Astudillo et al., 2020; Zhang and Ji, 2021) on the movie dialogue and summaries. For example, let's consider a subgraph extracted from IE containing events such as "<Midge, arrested (by), the police>". IE has the advantage of performing entity/event linking, so the entity nodes have other direct connections in the constructed MovieKG as well, such as "<Midge, located in, Manhattan>" and "<police, (also) arrest, comedian Larry Bruce>". However, other important events may not be captured by the IE ontology due to its more abstract or rarer occurrence in daily life and important news events, such as "<Midge, bailed out by, Gaslight manager Susie Meyerson>" and "<Gaslight manager Susie Meyerson, recognize, Midge's talent>", even though these information triplets can be directly extracted as the noun/verb-form concept nodes from AMR parsing on plot summary and dialogue transcripts. Hence, we add subgraphs from AMR parsing to the movieKG initially constructed from IE where there exists a coreferential event or entity node, such as "Midge".

To further enrich this KG with visual information, we perform event extraction using grounded image situation recognition and localization (Pratt et al., 2020), which extracts the verb in action from the visual frame, as well as the semantic roles of objects detected (agent, item, destination, place, etc). We also perform character mapping for agents that play a role in visual events by finding the closest match from a bank of character profile images[3], using visual features extracted by an iResNet model (Duta et al., 2021) following Meng et al. (2021). Finally, we merge objects and events that occur across textual and visual media based on embedding similarity in a multimedia common semantic space computed from CLiP, introduced in Sec 4.1.

Knowledge Subgraph Retrieval Now, we cannot simply select the neighborhood surrounding seed nodes that best match question q as the relevant subgraph because recurring entities contain many dense connections in the long story-telling domain. So instead, we select relevant subgraph based on context-aware saliency. Given a natural language question, we first represent it as a query graph that has undergone knowledge extraction for closer comparability with the movieKG. For instance, a question about *"the initial encounter between Midge and Larry Bruce"* becomes a graph with two connections: [Midge–the initial encounter, the initial encounter–Larry Bruce]. At the time of probing, we compute contextualized embeddings of node mentions, from BERT

---

[3] These can be manually identified from video frames or automatically collected from `www.imdb.com`.

encoders with awareness of the sentence that they occur in, as the local features for both the query graph and the movieKG, concatenate with wider-context knowledge embeddings from a two-hop neighborhood aggregation of maximally aligning neighbor nodes' local embedding with respect to the query graph nodes. For query graph nodes, we concatenate the sentence-level BERT embedding as their wider context features. We score and rank $KG_{subgraph}$ selection based on the semantic (cosine) similarity of these knowledge embeddings between nodes in the movieKG and query graph.

3.3 Combining Context for Answer Fusion

In this work, we regard extractive QA as the task of selecting the relevant V_{dial_i} and V_{f_i} (as well as summary sentences V_{S_i} and *movieKG*$_{subgraph}$ though these lack labels for evaluation) from a movie/TV show, given question q. We formulate abstractive QA as the task of natural language answer generation, conditioned on these relevant V_{dial_i}, V_{f_i}, *movieKG*$_{subgraph}$, and V_{S_i}, providing complementary context to each other. Though it is tempting to learn an answer generation model that intakes context retrieved from different modalities through a straightforward common semantic space, we note the low-resource setting of our challenging task (in which annotations are expensive and time-consuming to obtain). Thus, we take advantage of pretrained conditional text generation transformers, in particular, BART, which has an encoder to extract context information and a generative decoder for sequential token generation, as a suitable backbone for answer generation (Lewis et al., 2020; Khashabi et al., 2020). Moreover, we aim to match the format of input that robust and high-performing conditional text generators have been pretrained on, which is natural language text. While q, V_{dial_i}, and V_{S_i} directly fit this desirable input format, retrieved V_{f_i} and *movieKG*$_{subgraph}$ should be projected into a textual semantic space for optimal alignment with our pretrained backbone answer generator. Hence, for the retrieved movie$KG_{subgraph}$, we take a stringified representation of its structured connections following (Ribeiro et al., 2020). For the retrieved video frames, we make sure that knowledge elements in these images are included in or concatenated to the stringified movieKG subgraph representation. Although our transformer-based answer fusion mechanism may be relatively simple and straightforward, we observe it can handle QAs such as the ones in Table 6, detailed further in Sec 4. Figure 1 provides a walk-through illustration of our overall framework, which we refer to as the DEEPMAVEN. It is worthy to note that by nature of our information extraction-based QA reasoning, our textual answer generation is inherently supplemented with grounding to the visual context retrieved. This includes bounding box localization of the actions and participants (e.g., event – *'drinking'*, person – *'Midge Maisel'*, etc.) from the video frames.

4 Experiments

4.1 Benchmark and Dataset

DeepMovieQA Corpus This is our newly constructed dataset from 4 TV shows and 4 movies. The QAs may involve deep content beyond plot summary information, background knowledge, and higher-order reasoning across different scenes and events, as well as cross-media inferences. The answers are designed to be conversational friendly in nature and have an average token length of 19. A separate expert annotator manually checked 100 random QA pairs and judged all of them as accurate, informative, and comprehensive. We used a 8:1:1 train/val/test data split due to the corpus size.

Shallow MovieQA Corpus This is a multiple choice QA dataset (Tapaswi et al., 2016) annotated solely from movie plot summaries, with a pre-established 66-13-21% train/val/test data split.

4.2 Experimental Setting

For retrieval (extractive question answering), we include single modality approaches as the natural baselines to our proposed model, **DEEPMAVEN**.

- **Baseline 1: Dialogue Component Only** Here, we simply retrieve the relevant dialogue sections from BERT embedding similarity with respect to the input question, similar to the top performing approach (Mossad et al., 2020) in MovieQA leaderboard.
- **Baseline 2: Visual Component Only** Similarly, here, we use the features from CLiP for semantic matching between the question and visual frames.
- **Baseline 3: Textual & Visual Component w/o Structured Knowledge** We further perform comparison with a simplified version of our DeepMaven framework that focuses on the cross-media dialogue and

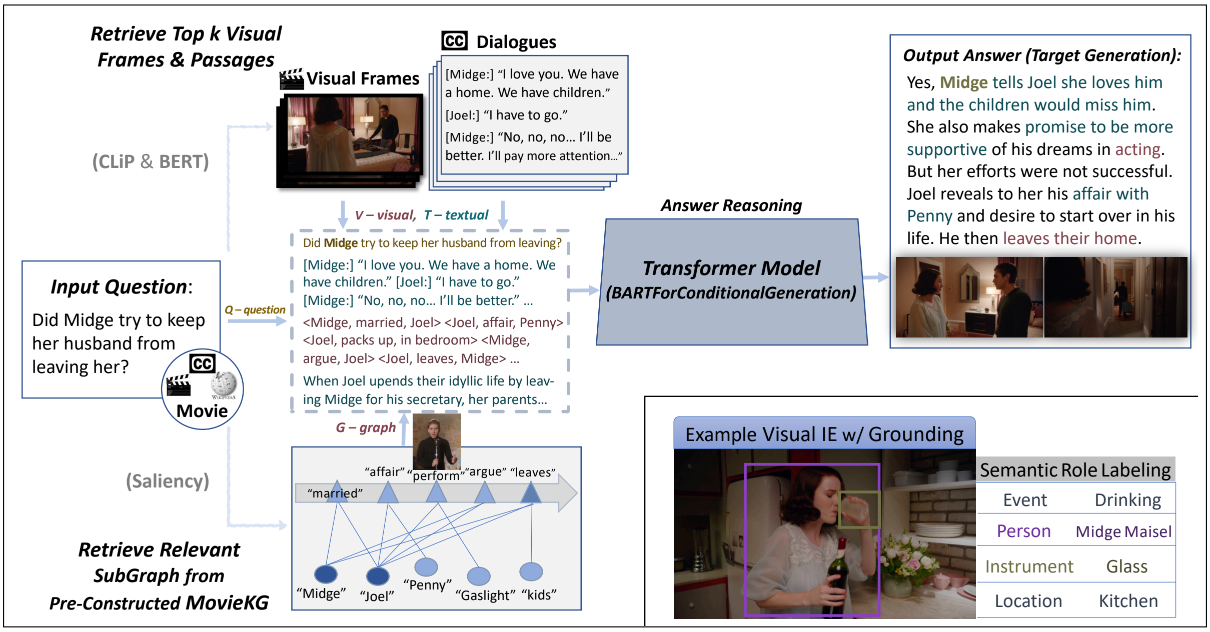


Figure 1: Our DEEPMAVEN model architecture takes as input the question, concatenated with relevant dialogues, visual frames, movie$KG_{subgraph}$, and summary sentences (illustrated by the *dotted box*), for answer stitching through a pretrained conditional text generator backbone. Note: due to low-resource data annotations, we convert the visual frame and $KG_{subgraph}$ into string format to better align with the semantic space of the answer generator.

visual frames without structured knowledge from the movie $\text{KG}_{subgraph}$.

For abstractive question answer generation, we consider the base variant of DEEPMAVEN, which finetunes *BART* (Lewis et al., 2020), using the textual dialogue retrieved from movies/TV shows as input context only. We also include the baseline of *UniVL* (Luo et al., 2020), which has been pretrained on video and language multimodal feature representation and text generation.

4.3 Evaluation Metrics

For long-distance retrieval, we report the hit@K metrics of whether the selected section of the dialogue or visual frames fall under the source time interval from the QA annotation. For natural language answer generation, we compute the ROUGE-L (Lin, 2004) F-scores[4]. However, there might be many variants of words and phrases with different level of granularities that can be used to describe the same answer in the same scenes. Take the question about *"how Midge's parent react to the news that Joel has left Midge"* in "The Mavelous Mrs Maisel" for example. The ground truth answer annotation provides detailed descriptions on how "Midge's dad starts to play the piano frantically and later blames Midge for marrying a weak man against his advice" while "Midge's mother starts crying and whining heavily". We observed that our system generated answers are more general but still correct, outputting "the way the parents are reacting is very upset and blaming each other". A strict measurement of n-gram overlaps from ROUGE would not credit such an answer generation sufficiently. Therefore, we also include a semantic-based BLEURT [5] metric (Sellam et al., 2020). In addition, we conduct a human assessment on the extracted answers as well. For each pair of system extracted answer and ground-truth answer, we ask human assessors to judge whether the system answer is completely correct, partially correct or incorrect.

4.4 Quantitative Results and Analysis

Extractive Question Answer Retrieval As shown in Table 4, retrieval is a non-trivial task in the movie setting, which contains lengthy plot content, and a random approach has a very low hit@K=5 of 0.08. Single modality retrieval baselines perform similarly, with text retrieval using BERT and visual retrieval using CLiP having hit@K=5 of 0.32 and 0.28 respectively, while cross-media retrieval achieves a noticeable boost to hit@K=5 of 0.41, suggesting that textual and visual features offer similar levels of useful signals that complement each other for making sense of

---

[4] `www.pypi.org/project/pycocoevalcap/`

[5] We use the BLEURT-20 scorer model from `https://github.com/google-research/bleurt`.

Approach	*Hit@K=5*
Random	0.08
Textual (BERT)*	0.32
Visual (CLiP)*	0.28
Textual+Visual$^{\triangle}$	0.41
Textual+Visual+KG priming	**0.49**

Table 4: These are the video frame retrieval results, with $*$ as the baselines, $\triangle$ as the simplified DeepMaven without structured knowledge, and **bold** as the full DeepMaven retrieval approach.

question inputs from long movie/TV show content. Finally, movieKG-guided structured information on top of semantic matching from direct source data, using our **DeepMaven** framework, unlocks the best retrieval hit rate (to near 50%).

Abstractive Question Answer Generation Natural language answer generation requires an additional decoder module for outputting tokens. Our **DeepMaven** model achieves a Rouge-L of 61.6% in answer generation on the MovieQA benchmark. Model performance for the new, challenging DeepMovieQA benchmark is reported in Table 5 below.

	Rouge-1	Rouge-L	BLEURT
UniVL	16.8	15.1	22.3
DeepMaven			
- dial. only	13.9	11.1	22.1
- all	**21.6**	**17.4**	**31.7**

Table 5: Answer generation (%) scores on our DeepMovieQA dataset.

4.5 Qualitative Analysis

Table 6 shows answer generation results while Table 7 shows content selection examples, performed by **DeepMaven**. We additionally conduct a human assessment on the DeepMovieQA answer generations from held-out data, rating answers on a likert scale of $0-5$ (with 0 being nonsense and 5 being perfect). Thirteen percent of the generated answers scored ≥ 4, with a Kappa coefficient of 0.42, reflecting moderate inter-annotator agreement.

4.6 Remaining Challenges

	Questions	Answers
MovieQA	Who comes to the officers' rescue? *–World Trade Center*	Two United States marines, Dave Karnes and Jason Thomas. Two United States marines, Dave Karnes and Jason Thomas.
	Where are Stigman and Trench taken after being captured? *–2 Guns*	Mexico To Greco's farm in Mexico
DeepMovieQA	How does Joel leave the apartment after they fell asleep the night before? *–MRSM*	Joel gets out of bed and drives to work. Joel sneaks out Midge's bedroom window the next morning, just like he did when they first dated.
	What were in the boxes in the back of the truck Sajid was driving? *–FMAN*	They were filled with nerve gas. The boxes were filled with canisters of nerve gas that were going to be used to attack New Delhi.

Table 6: Example DeepMaven abstractive QA results, with the generated and grouth truth answers.

In general, we found that DeepMaven retrieval performs worse on QAs that involve commonsense reasoning and reference to objects/entities that may not be easily picked up by our structure-guided cross-media semantic matching system. Some deeper movie QAs may also be more suitable for the conversational question answering setting due to insufficiently detailed question wording and open-ended answer form. Finally, we observed that long-form answer generation with a limited-sized training data is more prone to hallucination. Given questions such as *"What did Susie promise to one of the waiters at the Copacabana Club in return for smuggling her into the club for free?"*, the language model generator backbone tends to fill in some plausible pre-trained knowledge such as *"Susie promises to buy one of the men a drink after every act"* that may not match the actual detailed answer of *"Susie promised him a prime slot for his singing act at the Gaslight Café for 2 weeks. Unfortunately Susie thinks his act is awful but keeps her word even since they had to watch the show from the kitchen"*. Minimizing hallucination in long-form low-resource question answer generation is an important issue that merits future investigation.

5 Related Work

Text QA: QA has been a popular task (Rajpurkar et al., 2016), with significant advances made recently by attention-guided transformers (Qu et al., 2019). Several corpora have been proposed for QA in story comprehension. FriendsQA (Yang and Choi, 2019) bases QA on short dialogue exchanges. MovieQA (Tapaswi et al., 2016) and NarrativeQA (Kocisky et al., 2018) are annotated from plot summaries. (Zhou et al., 2018; Moon et al., 2019; Zhou

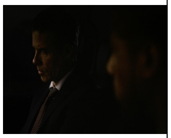
	Cue Category	Question	Retrieved Dialogue Passages and	Visual Frames
Impressive Cases	Dialogue Content	What did Kelly threaten Clay with after forcing him into his car? *–WORE*	"Why my family? My wife? My daughter? Why kill my team? Rykov was all in, so I'd like a little bit of what you sold him... Know where we driving? You got a farm out in West Virginia, right? Your daughter should be home from college. Wesleyan. That's a good school. Your wife. Your son too. So I suggest you start talking."	
	Visual Scenes	Who takes care of Lee after the fight? *–MBTS*	"Settle down, all right? Are we cool? Get off. You'll kill him... Should he go to the hospital? I don't think so. Nothing's broken."	
	Implicit Cross-Media Entity Grounding	What does Midge's mother want to talk about after Midge arrives at her parent's apartment? *–MRSM*	"Thanks for taking the kids last night. Were they okay? - We need to talk about the baby. Why? What's the matter with her? That forehead is not improving. - What? Are you sure?"	
	Implicit Multi-hop Event Reasoning	What does Midge do after Joel falls asleep and before he wakes up again? *–MRSM*	"Good night, Gracie. Hey. Good morning. - Did the alarm go off? - It sure did. Wow. I didn't hear it at all. You never do. - Good morning, Jerry. - Good morning, Mrs. Maisel."	
Erroneous Cases	Failure to Identify Certain Visual Objects	What does Midge suggest in the **taxi** to improve Joel's performance? *–MRSM*	"Maybe you should write a beginning, something that says who you are or something. What do you think? Good evening. What a nice... Good evening, ladies and gentlemen. Thank you for the nice... nice... 'Nice' is a bad, bad word. All that applause for me? What am I, putting out after?..."	
	Mismatch in Level of Detail Between Query and Context	What did Midge do while her friend Imogene is visiting to make sure she's still in shape? *– MRSM*	"She's going on and on about this miracle treatment she had done in Mexico. It involved goat's milk and avocadoes. **Right ankle 8, left ankle 8.** They smear it on your face, wrap a hot towel around your head..."	

Table 7: Impressive and Erroneous Examples of Retrieval Results.

et al., 2020) annotated open-ended conversations with groundings to either Wikipedia pages, or pre-existing structured knowledge graphs such as Freebase (Bollacker et al., 2008) and XLORE (Wang et al., 2013). These benchmarks overlook the full content from the original data, which we address.

Visual QA: The visual question answering (VQA) task (Antol et al., 2015) aims to predict a natural language answer, given a natural language question and image(s) without other textual context. Various datasets have been constructed for this task, such as VQA 2.0 (Goyal et al., 2017), VCR (Zellers et al., 2019), and scientific PlotQA (Methani et al., 2020), in the setting of single images; as well as MSR-VTT-QA (Xu et al., 2016), MovieFIB (Maharaj et al., 2017), and VideoQA (Zhu et al., 2017), in the setting of visual frame sequences. The answers are typically multiple choices or from a predefined vocabulary. Simple baseline methods that only use question understanding (Kazemi and Elqursh, 2017) or sentiment analysis on answer options (Manjunatha et al., 2019) have proven surprisingly well on datasets such as VQA and VQA 2.0 but are unlikely to provide good answers for understanding complex events and person interactions.

Multimedia QA: There are increasing interest nowadays in using information from multiple modalities for answering questions, such as reasoning through text, images, and tables in ManyModalQA (Hannan et al., 2020), MultiModalQA (Talmor et al., 2021), and MuMuQA (Reddy et al., 2022). The (movie/TV) video domain is a further step that involves dynamic events from dialogues and corresponding visual frames. TVQA (Lei et al., 2018) presents a multimedia QA dataset grounded on minute-long video clip snippets from popular TV show, with frame localization added in TVQA+ (Lei et al., 2020a). However, their QAs involve highly scene-specific discussion points, trivial to the deeper contents around the central plotline, e.g. "*Q: What is on the couch behind Joey when he is at the counter? A: A soccer ball*". In contrast, our work highlights deeper knowledge and multi-frame information synthesis.

Other Story-based Video Understanding: Benchmarks such as MovieNet (Huang et al., 2020) and LVU (Wu and Krahenbuhl, 2021) gauge a variety of content classification tasks related to the 'scene/place', 'cinematic style', 'genre', 'produced year', 'popularity', etc. Moreover, Lei et al. (2020b); Huang et al. (2020); Bain et al. (2020) explore retrieval based on sentence-length scene description while Liu et al. (2020) study video-language inference, but these are all less challenging than the natural question answering setting our work focuses on.

6 Conclusions and Future Work

This work is the first to study long-distance movie question answering. Through guidance from multimedia source context and structured knowledge retrieval, our proposed model, DEEPMAVEN, is shown to perform well for extractive question answering, as well as abstractive answer generation for the questions in pre-existing MovieQA benchmarks. But improvements are needed for generating answers to the more challenging questions presented by our new DeepMovieQA benchmark, which serve as a better reflection of real-world application settings with more reasoning involved. We aim to kickstart an extractive movie question answering interface for human users, and through this, naturally acquire genuine movie questions for more effectively expanding DeepMovieQA annotation, and later extend this into interactive conversational AI settings.

7 Limitations

Our work focuses on video question answering that aims to be detailed and engaging. It is not interactive in nature like conversation exchanges. Finally, we bear in mind that automatically generated answers may contain the risk of insensitive phrasing, and mitigating language model bias deserves further exploration and efforts.

References

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Max Bain, Arsha Nagrani, Andrew Brown, and Andrew Zisserman. 2020. Condensed movies: Story based retrieval with contextual embeddings. In *Proceedings of the Asian Conference on Computer Vision*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Ionut Cosmin Duta, Li Liu, Fan Zhu, and Ling Shao. 2021. Improved residual networks for image and video recognition. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 9415–9422. IEEE.

Ramón Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1001–1007, Online. Association for Computational Linguistics.

Noa Garcia, Mayu Otani, Chenhui Chu, and Yuta Nakashima. 2020. Knowit vqa: Answering knowledge-based questions about videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10826–10834.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913.

Darryl Hannan, Akshay Jain, and Mohit Bansal. 2020. Manymodalqa: Modality disambiguation and qa over diverse inputs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7879–7886.

Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.

Qingqiu Huang, Yu Xiong, Anyi Rao, Jiaze Wang, and Dahua Lin. 2020. Movienet: A holistic dataset for movie understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 709–727. Springer.

Vahid Kazemi and Ali Elqursh. 2017. Show, ask, attend, and answer: A strong baseline for visual question answering. *arXiv preprint arXiv:1704.03162*.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *arXiv preprint arXiv:2005.00700*.

Tomas Kocisky, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gabor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. In *Transactions of the Association for Computational Linguistics*, volume 6, pages 317–328. MIT Press.

Tuan Lai, Heng Ji, Trung Bui, Quan Hung Tran, Franck Dernoncourt, and Walter Chang. 2021. A context-dependent gated module for incorporating symbolic semantics into event coreference resolution. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3491–3499, Online. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara Berg. 2018. TVQA: Localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1379, Brussels, Belgium. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Tamara Berg, and Mohit Bansal. 2020a. TVQA+: Spatio-temporal grounding for video question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8211–8225, Online. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Tamara L Berg, and Mohit Bansal. 2020b. Tvr: A large-scale dataset for video-subtitle moment retrieval. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, pages 447–463. Springer.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009.

Jingzhou Liu, Wenhu Chen, Yu Cheng, Zhe Gan, Licheng Yu, Yiming Yang, and Jingjing Liu. 2020. Violin: A large-scale dataset for video-and-language inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10900–10910.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. 2020. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*.

Tegan Maharaj, Nicolas Ballas, Anna Rohrbach, Aaron Courville, and Christopher Pal. 2017. A dataset and exploration of models for understanding video data through fill-in-the-blank question-answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6884–6893.

Varun Manjunatha, Nirat Saini, and Larry S Davis. 2019. Explicit bias discovery in visual question answering models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9562–9571.

Qiang Meng, Shichao Zhao, Zhida Huang, and Feng Zhou. 2021. Magface: A universal representation for face recognition and quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14225–14234.

Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. 2020. Plotqa: Reasoning over scientific plots. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1527–1536.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854.

Omar Mossad, Amgad Ahmed, Anandharaju Raju, Hari Karthikeyan, and Zayed Ahmed. 2020. Fat albert: Finding answers in large texts using semantic similarity attention layer based on bert. *arXiv preprint arXiv:2009.01004*.

Xiaoman Pan, Taylor Cassidy, Ulf Hermjakob, Heng Ji, and Kevin Knight. 2015. Unsupervised entity linking with abstract meaning representation. In *Proceedings of the 2015 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 1130–1139.

Sarah Pratt, Mark Yatskar, Luca Weihs, Ali Farhadi, and Aniruddha Kembhavi. 2020. Grounded situation recognition. In *European Conference on Computer Vision*, pages 314–332. Springer.

Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W Bruce Croft, and Mohit Iyyer. 2020. Open-retrieval conversational question answering. In *Proceedings of* *the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 539–548.

Chen Qu, Liu Yang, Minghui Qiu, W Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. 2019. Bert with history answer embedding for conversational question answering. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 1133–1136.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Revanth Gangi Reddy, Xilin Rui, Manling Li, Xudong Lin, Haoyang Wen, Jaemin Cho, Lifu Huang, Mohit Bansal, Avi Sil, Shih-Fu Chang, Alexander Schwing, and Heng Ji. 2022. Mumuqa: Multi-media multi-hop news question answering based on cross-media grounding. In *Proc. Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI2022)*.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodalqa: Complex question answering over text, tables and images. *arXiv preprint arXiv:2104.06039*.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4631–4640.

Zhigang Wang, Juanzi Li, Zhichun Wang, Shuangjie Li, Mingyang Li, Dongsheng Zhang, Yao Shi, Yongbin Liu, Peng Zhang, and Jie Tang. 2013. Xlore: A large-scale english-chinese bilingual knowledge graph. In *International semantic web conference (Posters & Demos)*, volume 1035, pages 121–124.

Haoyang Wen, Ying Lin, Tuan Lai, Xiaoman Pan, Sha Li, Xudong Lin, Ben Zhou, Manling Li, Haoyu Wang, Hongming Zhang, et al. 2021. Resin: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 133–143.

Chao-Yuan Wu and Philipp Krahenbuhl. 2021. Towards long-form video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1884–1894.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296.

Zhengzhe Yang and Jinho D Choi. 2019. Friendsqa: Open-domain question answering on tv show transcripts. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 188–197.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

Zixuan Zhang and Heng Ji. 2021. Abstract meaning representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49.

Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, and Xiaoyan Zhu. 2020. Kdconv: A chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation. *arXiv preprint arXiv:2004.04100*.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. *arXiv preprint arXiv:1809.07358*.

Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. 2017. Uncovering the temporal context for video question answering. *International Journal of Computer Vision*, 124(3):409–421.

Salient Span Masking for Temporal Understanding

Jeremy R. Cole **Aditi Chaudhary** **Bhuwan Dhingra** **Partha Talukdar**
Google Research
`{jrcole,aditichaud,bdhingra,partha@google.com}`

Abstract

Salient Span Masking (SSM) has shown itself to be an effective strategy to improve closed-book question answering performance. SSM extends general masked language model pretraining by creating additional unsupervised training sentences that mask a single entity or date span, thus oversampling factual information. Despite the success of this paradigm, the span types and sampling strategies are relatively arbitrary and not widely studied for other tasks. Thus, we investigate SSM from the perspective of temporal tasks, where learning a good representation of various temporal expressions is important. To that end, we introduce Temporal Span Masking (TSM) intermediate training. First, we find that SSM alone improves the downstream performance on three temporal tasks by an avg. +5.8 points. Further, we are able to achieve additional improvements (avg. +0.29 points) by adding the TSM task. These comprise the new best reported results on the targeted tasks. Our analysis suggests that the effectiveness of SSM stems from the sentences chosen in the training data rather than the mask choice: sentences with entities frequently also contain temporal expressions. Nonetheless, the additional targeted spans of TSM can still improve performance, especially in a zero-shot context.

1 Introduction

Salient Span Masking (SSM), first introduced by Guu et al. (2020) for retrieval-based language modeling, has shown performance gains for closed-book question answering (CBQA) (Roberts et al., 2020; Ye et al., 2020). SSM is a form of intermediate pretraining (Ye et al., 2021), where a pretrained model such as a BERT (Devlin et al., 2019) or T5 (Raffel et al., 2020) is trained further before task-specific finetuning, generally on more specialized data that does not require expensive annotations. Specifically, SSM uses the masked language modeling objective but only masks named entities and dates in sentences from English Wikipedia articles; these "salient" spans likely contain more facts, so the language model must memorize more facts in order to do the task successfully (Petroni et al., 2019). The authors use a named entity recognition model to identify entity spans and a regular expression to identify date spans. While this works well for knowledge intensive downstream tasks, such as entity-centric question answering, it remains unclear whether it is helpful for tasks that are less aligned with the data, such as common sense or temporal reasoning. Moreover, is it possible to select spans that are more related to a downstream task in order to get further performance gains?

In this work, we investigate SSM for tasks that require understanding *temporal* expressions. While SSM does include dates, the tasks we investigate include other complex temporal expressions such as durations and intervals. To that end, we introduce Temporal Span Masking (TSM): an intermediate pretraining strategy for predicting spans that are likely temporal expressions (Figure 1). Similar to SSM, TSM is automatically generated from English Wikipedia articles. We compare models trained on TSM and SSM on three temporal tasks, namely MC-TACO (Zhou et al., 2019), TimeDIAL (Qin et al., 2021) and SituatedQA (Zhang and Choi, 2021), and for one general-purpose question answering (QA) task of Natural Questions (NQ) (Kwiatkowski et al., 2019). We summarize our contributions as follows:

- We propose TSM Intermediate Training, which automatically selects temporal spans for masking.
- The new best reported results on the three temporal tasks: the best average performance is from a mixture of TSM and SSM. This mixture also does slightly better than SSM on Natural Questions.
- Experiments investigating the role of differ-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3052–3060
May 2-6, 2023 ©2023 Association for Computational Linguistics

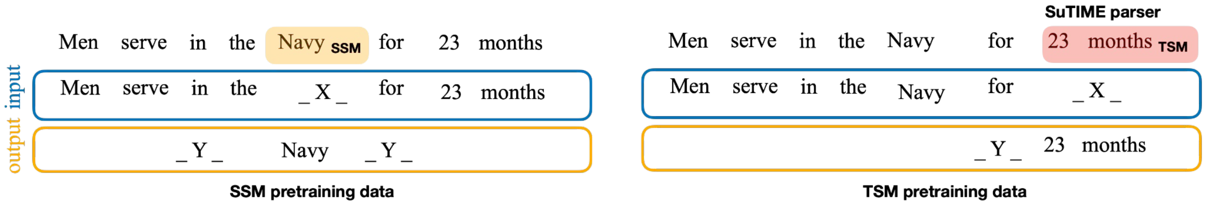


Figure 1: Overview of the TSM and SSM tasks: While SSM (Guu et al., 2020) identifies named entities and dates as salient spans, for TSM we use SUTIME parser that captures other temporal expressions such as durations and intervals. SUTIME is first applied on raw sentences to identify any temporal expressions. Next, training data for TSM is created from those sentences which have at least one temporal expression identified. In SSM, the masked spans comprise of named entity or date spans. Input to each task consists of the sentence with the selected span dropped out where the model is trained to predict the dropped tokens.

ent TSM and SSM span types, showing entity spans alone are helpful, which implies that difficult examples help improve representations of the unmasked spans as well.

2 Methods

Following Roberts et al. (2020), we utilize intermediate training to improve pretrained models' generalization to downstream tasks. All models are initialized from the encoder-decoder T5-1.1-XXL language model (Raffel et al., 2020), which was shown to have the best closed-book QA performance in Roberts et al. (2020).

2.1 Background: Salient Span Masking

Salient Span Masking (SSM) was first introduced by Guu et al. (2020) and is designed to specifically mask named entities and dates or *salient spans*. These salient spans are automatically identified from English Wikipedia using a Named Entity Recognition model to find entities as well as a regular expression to find dates. The authors mask one such span per sentence during training: the model must maximize the probability of the masked entity or date given the corrupted input sentence. Guu et al. (2020) designed the task to improve downstream performance on tasks requiring world knowledge in order to improve their retrieval-augmented model's ability to use retrieved texts. Roberts et al. (2020) then adapted this task for closed book encoder-decoder models.

2.2 Proposed: Temporal Span Masking

Inspired from the success of SSM, TSM is designed to address problems requiring temporal knowledge. To create training examples for TSM, we automatically identify temporal expressions in a large corpus using SUTIME (Chang and Manning, 2012), a rule-based temporal parser, that identifies temporal expressions from raw text. Given an input sentence, SUTIME is built to identify expressions of the following four types: **Time** which indicates a particular point in time such as *next Monday*, **Duration** such as *3 days*, **Set** which indicates periodic set of time that occur with some frequency such as *every 4 years* and **Date** such as *January 1*.

We run SUTIME on all of English Wikipedia[1]. Specifically, we divide the articles into sentences, and apply SUTIME[2] on each sentence. For our TSM training data, we ensure that exactly one temporal span is masked per example. So, if a sentence contains four temporal spans, we create four training examples with exactly one temporal span masked per example. Details of the temporal distribution are in Table 4. Each example is created by masking the tokens belonging to the temporal expression, as shown in Figure 1, corrupting the input sentence by replacing the span with (_X_) and having the model predict the masked tokens. The training objective is to maximize the probability of the target span given the corrupted input sentence, similar to T5's span corruption training objective (Raffel et al., 2020) and the SSM training objective (Roberts et al., 2020).

2.3 Model Variants

All newly reported results are based on T5-1.1-XXL models. Proposed models are named for their intermediate training objective: TSM is trained solely on the masked temporal spans described above; SSM is trained solely on the training objec-

---

[1] We use the 2020 snapshot of English Wikipedia (TFDS datadump *wikipedia20201201en*)

[2] `https://github.com/FraBle/python-sutime`

tive described in Roberts et al. (2020).[3]

We also investigate a version of the SSM pretraining data that only uses the *named entity* spans identified by the NER model; in other words, all of the *date* spans identified by the regular expression are removed, but the task is otherwise the same. We call this objective ENTITIES. Finally, we compare against models that are trained proportional mixtures of both TSM+SSM and TSM+ENTITIES.

For baseline models, we use the same pretrained model (T5-1.1-XXL) with no intermediate training (T5), as well as a T5-1.1-XXL model which has been trained for an additional 100K steps on the prefix LM task (T5-LM; Lester et al. 2021).

2.4 Downstream Temporal Tasks

We evaluate on three downstream temporal tasks for evaluation, finetuning a model on each task separately. Below, we briefly describe the tasks and datasets, with additional training details in Appendix A.

MC-TACO Zhou et al. (2019) release a human-annotated dataset to measure temporal commonsense understanding. It consists of 13k tuples of *(sentence, question, candidate answer)* covering five types of common sense problems such as event frequency, event duration, event ordering, stationarity and event typical time. Given a sentence context, a temporal question about that context, and a possible commonsense answer, the task is to determine whether the provided answer is reasonable for the given context. For instance, for the event of *taking a shower* with four possible answer choices *five minutes, fifteen minutes, fifteen hours*, and *fifteen years*, the first two are plausible and will have the *yes* label while the latter two choices would be *no*. There is no training data released for this task, so we finetune the model on the provided validation set and evaluate on the test set.

TimeDIAL Qin et al. (2021) release a human-annotated multi-turn dialog dataset for measuring temporal commonsense understanding in a dialog setting. The dataset comprises of challenge test set with 1.1k dialog instances derived from the Daily Dialog dataset described in Qin et al. (2021). TimeDIAL dialogs mostly comprise of common sense instances where the answers generally consist of one temporal span. For instance, in the following dialog “I’ll just be a minute’., the span “a minute” may be masked out and the model is required to predict the masked span based on the dialog turns. Given a dialog with a temporal expression masked out, the task is to correctly predict which two of the four provided answers are valid in the given context. We report results without finetuning (styled TimeDIAL-0) as well as results from finetuning the model on the Daily Dialog dataset.

SituatedQA Zhang and Choi (2021) release an open-domain QA dataset derived from existing question answering datasets with additional annotations that resolve temporal and geographic ambiguities. Each example consists of a disambiguated question: for instance, “Which COVID-19 vaccines have been authorized in the US [as of 2020]?” or “What was the first COVID-19 vaccine to be authorized [in the US]?”. For the purpose of this work, we focus on the temporal questions. These consist of 9K additional questions, with a training set of about 4.5K questions. We finetune on the training set and evaluate on the test set.

2.5 Natural Questions

While the focus of our method is improving temporal question answering performance, we also wanted to ensure that our method does not degrade performance on non-temporal question answering tasks. Thus, we also evaluate our model variants on Natural Questions (Kwiatkowski et al., 2019), using the “open” variant popularized by Lee et al. (2019). These examples discard those questions without short answers or that require an evidence document to answer. These consist of about 87K questions for training and an additional 3.6k questions for validation, which we use for evaluation.

3 Results and Discussion

Our main results can be found in Table 1. Results including Natural Questions can be found in Table 2. Note that the Natural Questions results have minor variations from published numbers; we ran these baselines ourselves, and it is possible the training setup differed slightly.

T5 and T5-LM The T5 model sets a relatively high baseline compared to previously reported models. The T5-LM model’s extra non-domain-specific pretraining does not help on any task, suggesting

---

[3] We note that the SSM-spans are derived from the 2018 snapshot of English Wikipedia (same as Guu et al. (2020)) while TSM-spans from the 2020 snapshot.

	SituatedQA		MC-TACO		TimeDIAL		TimeDIAL-0			
Model	F1	EM	F1	EM	1-Best	2-Best	1-Best	2-Best	Overall	Overall-0
Best Reported	–	18.53	82.92	63.81	–	76.10	–	50.60	52.74	44.31
T5	25.75	19.78	84.00	64.56	99.91	**84.50**	90.85	37.59	56.28	40.64
T5-LM	25.38	19.63	81.99	59.83	99.91	80.60	86.87	32.16	53.35	37.21
SSM	29.92	23.12	85.88	68.39	99.73	84.06	96.74	67.21	58.52	52.91
Entities	29.42	22.82	85.47	66.59	99.91	83.06	97.64	67.93	57.49	52.45
TSM	27.42	21.18	84.89	65.92	99.91	83.88	**99.82**	**77.54**	56.99	54.88
TSM+SSM	29.33	22.76	**86.20**	67.64	**100.0**	83.78	98.19	73.10	58.03	54.5
Entities+TSM	**30.78**	**24.60**	85.32	**68.47**	99.91	84.24	98.91	76.09	**59.09**	**56.39**

Table 1: Aggregate metrics across the three datasets. *Overall* performance is the simple arithmetic average of the harder metric for each approach (EM, EM, 2B); *Overall-0* uses TimeDIAL-0 instead of TimeDIAL. The second section contains our runs of earlier models; the Best Reported uses best known published numbers. The third section represents our models. Note that all models (in the second and third sections) are based on T5-1.1-XXL models. Best Reported results are ALBERT (Lan et al., 2019) from Abramson and Emami (2022) for TimeDIAL, BART results from Zhang and Choi (2021), and DeBERTa (He et al., 2020) results from the leaderboard for MC-TACO. Note that F1 for MC-TACO is based on the precision/recall over answers and EM is based on labeling every answer for a question correctly, while the F1 for SituatedQA is based on the token-level F1 of the answer span.

Model	F1	EM	Overall	Overall-0
SSM	41.57	34.6	52.54	48.33
T5	39.35	32.38	50.31	38.58
T5-LM	37.16	31.14	47.80	35.69
Entities	41.21	34.52	51.75	47.97
TSM	39.24	32.69	50.92	49.33
TSM+SSM	41.80	35.10	52.3	49.65
Entities+TSM	**41.89**	**35.18**	**53.11**	**51.09**

Table 2: Results on Natural Questions – *Overall* and *Overall-0* results include the same metrics from Table 1 with Natural Questions (EM) included. The first section represents our baselines. Note that all models are based on T5-1.1-XXL models.

extra training steps does not in of itself cause improvements on these tasks.

Entities The Entities model, which is trained on only non-temporal entity spans, performs better overall than the TSM task. It only does worse on the TimeDIAL dataset, which is almost entirely focused on conversational, non-knowledge based contexts. It still does substantially better than the base T5 model when no finetuning data is available. This high performance is possibly due to the prevalence of temporal spans in the SSM training data. Running SUTIME on the Entities data reveals that 45% of its training examples contain at least one date, duration, set, or time. This suggests that sentences with named entities in general already carry temporal-salient information useful for downstream temporal tasks. See Appendix B for a full breakdown of the co-occurrences.

SSM The SSM model is the second best overall. It benefits from both its own date spans as well as the frequent presence of temporal spans in the entities data, suggesting difficult example sentences are more important than the type of masked span. It does worse on TimeDIAL-0, however, where the task is to score the best temporal span.

TSM The TSM model improves upon the baseline T5 model but is worse overall than the SSM model. However, it is the best on TimeDIAL-0. This is likely because the DailyDialog training dataset is relatively large, which may overcome the need for intermediate pretraining altogether. Note that TSM achieves a mild performance improvement over the baseline T5 model on Natural Questions, but is notably worse than the other intermediate training methods.

TSM+SSM The TSM+SSM model improves over TSM but is worse than SSM outside of TimeDIAL-0. One possible reason for the regression is that TSM and SSM have overlapping Date span examples, which may make the intermediate task easier and thus less useful. However, it is slightly better than SSM on Natural Questions.

Entities+TSM The Entities+TSM model performs the best overall: with and without the extra training data for TimeDIAL. It has the benefit of

Model	T5	SSM	E+TSM
Duration	88.64	88.28	**89.09**
Set	86.39	**88.22**	87.79
Time	87.74	**88.70**	87.70
Date	65.75	66.63	**66.87**
Entities	42.17	45.76	**46.54**

Table 3: Aggregated performance across types for baselines and the best overall model (ENTITIES+TSM). For TimeDIAL, each answer span is labeled by SUTIME. Duration includes MC-TACO's "Event Duration"; Set includes MC-TACO's "Frequency", Time and Date both include MC-TACO's "Typical Time"; Date also includes SituatedQA; Entities include all of MC-TACO and SituatedQA. Note that the majority of the gains from SSM/TSM seem to be from Entities and Dates. See Appendix B for rationale behind these choices.

TSM spans without containing overlapping spans or losing the world knowledge from entity spans. It also performs slightly better than SSM on Natural Questions.

By Type We analyze model performance by temporal type in Table 3. The main improvement of both SSM and ENTITIES+TSM is in entity and date tasks. Surprisingly, TSM shows a regression on time tasks, and only gets a slight improvement on duration tasks. One possible hypothesis for this is that temporal expressions may be more informative when co-occurring with an entity. Note that these numbers are based on the trained versions of each dataset, excluding Natural Questions. Note that SituatedQA contains further breakdowns based on the scope of the date, but this does not map well to the other datasets.

4 Related Work

Span Masking and Intermediate Training Salient Span Masking (Guu et al., 2020) came out of a series of efforts like SpanBERT (Joshi et al., 2020) to select more difficult examples to improve models memorization of the text.

Most similar to us, Ye et al. (2021) explore a similar paradigm of choosing better spans for a downstream task (e.g., entity linking or relation extraction) where they experiment with both a heuristic masking policy similar to SSM and also a learned masking policy. They similarly find that masking spans that resemble downstream tasks improve performance, however, they also note that learned masking policies suffer from overfitting. Yang et al. (2020) and Zhou et al. (2020) explore intermediate training by designing heuristics to identify sentences containing temporal expressions and then adding additional tasks and losses, rather than using span masking. TSM differs in more closely resembling the pretraining task.

Levine et al. (2021) use pointwise mutual information to jointly mask highly correlated spans to avoid the model relying on local signals but rather learning from the broad context. They find this leads to faster and better pretraining. In the future, it might be interesting to see how PMI-spans can combine with knowledge-oriented span techniques such as SSM, TSM, and whether they can help in the intermediate training paradigm.

Temporal Understanding There has been a surge of interest in probing models' temporal awareness. While we evaluate on a three tasks, it is far from an exhaustive evaluation and we leave further evaluations of our method to future work.

Recently, Thukral et al. (2021) and Vashishtha et al. (2020) construct NLI datasets to test whether pretrained models understand certain types of common sense temporal expressions, such as containment. To probe common sense, we use TimeDIAL (Qin et al., 2021) for its naturalistic dialogues as well as MC-TACO (Zhou et al., 2020), which uses a diverse set of situations and temporal expressions.

For factual questions, open-response temporal questions are closely aligned with our work (e.g., TimeQA; Chen et al. 2021; TempLAMA; Dhingra et al. 2022). All of TempLAMA, TimeQA, and SituatedQA (Zhang and Choi, 2021) rely primarily on the year as the main temporal expression being tested, where facts are scoped to the provided years. To probe temporally scoped facts, we use SituatedQA for its more naturalistic questions.

5 Conclusion

In this work, we investigate SSM as it relates to temporal tasks that require understanding both commonsense and world knowledge questions and propose a new intermediate training method which selects spans generated by a temporal parser. These intermediate training strategies result in the best overall reported results on the selected downstream tasks. However, we find that even the entity spans from SSM are helpful for temporal tasks, likely because entity-containing examples also contain informative temporal knowledge.

Limitations

This analysis investigates only the encoder-decoder model architecture: in particular, encoder-only models such as BERT (Devlin et al., 2019) and decoder-only models such as GPT-2 (Radford et al., 2019) are excluded. Further, large language models, such as PaLM (Chowdhery et al., 2022) or GPT-3 (Brown et al., 2020) are also not investigated. See Appendix B for further discussion.

Acknowledgements

We would like to thank Kelvin Guu and Srini Narayanan, as well as our anonymous reviewers, for their helpful feedback on a previous version of this manuscript.

References

Darren Abramson and Ali Emami. 2022. An application of pseudo-log-likelihoods to natural language scoring. *arXiv preprint arXiv:2201.09377*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Angel X. Chang and Christopher Manning. 2012. SUTime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3735–3740, Istanbul, Turkey. European Language Resources Association (ELRA).

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

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

Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2021. {PMI}-masking: Principled masking of correlated spans. In *International Conference on Learning Representations*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqui. 2021. TIMEDIAL: Temporal commonsense reasoning in dialog. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7066–7076, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Shivin Thukral, Kunal Kukreja, and Christian Kavouras. 2021. Probing language models for understanding of temporal expressions. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 396–406, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Siddharth Vashishtha, Adam Poliak, Yash Kumar Lal, Benjamin Van Durme, and Aaron Steven White. 2020. Temporal reasoning in natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4070–4078, Online. Association for Computational Linguistics.

Zonglin Yang, Xinya Du, Alexander Rush, and Claire Cardie. 2020. Improving event duration prediction via time-aware pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3370–3378, Online. Association for Computational Linguistics.

Qinyuan Ye, Belinda Z Li, Sinong Wang, Benjamin Bolte, Hao Ma, Wen-tau Yih, Xiang Ren, and Madian Khabsa. 2020. Studying strategically: Learning to mask for closed-book qa. *arXiv preprint arXiv:2012.15856*.

Qinyuan Ye, Belinda Z. Li, Sinong Wang, Benjamin Bolte, Hao Ma, Wen-tau Yih, Xiang Ren, and Madian Khabsa. 2021. On the influence of masking policies in intermediate pre-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7190–7202, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. “going on a vacation” takes longer than “going for a walk”: A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. 2020. Temporal common sense acquisition with minimal supervision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7579–7589, Online. Association for Computational Linguistics.

A Training Details

All models are initialized from the public T5-1.1-XXL checkpoints.[4]

A.1 Intermediate Training

We use 256 Cloud TPU v3 cores for the intermediate training procedure using a batch size of 2048 and the fixed default learning rate of 0.001. Training generally proceeds for one epoch, which is between 100-150K steps depending on the precise task, though we used early stopping for the TSM and TSM+SSM models based on MC-TACO performance, as they seem to overfit.

A.2 Finetuning

We use 64 Cloud TPU v3 cores for finetuning and inference on all tasks. For MC-TACO, Natural Questions, and SituatedQA, we use the same fixed learning rate of 0.001 and train for 10K steps with batch size 128. For TimeDIAL, we attempt to follow their training setup more closely, and use a lower learning rate (0.0001) and train for up to 100K steps, still with batch size 128 and use early stopping on the validation set to inform when to stop. For most of the models that had intermediate training, the early stopping point was for 10K steps. However, for the basic T5 model, it was after 20K steps (improving from (82.97 → 84.51)), implying that it can overcome its lack of intermediate training with additional finetuning data. Note that in the zero-shot variant, no finetuning is done.

B Further Discussion

B.1 Other Experiments

We previously experimented with the T5 Large and T5-XL models, as well the as 1.0 versions of the T5 models that were first described in Raffel et al. (2020). In general, larger models and the 1.1 versions worked better. While we refrain from reporting results due to inconsistent setups, in general the smaller models were notably worse, such that distinguishing between two similar setups (such as TSM and SSM) was difficult on many tasks. While we know of no work testing salient span masking on extremely large models, it is possible it would actually show a larger impact, based on this trend. While left-to-right decoding serves as an awkward fit for the paradigm, if our hypothesis on the reason why SSM works is correct, then it should not prove to be a substantial hurdle. See also below for more discussion on said hypothesis.

Temporal Type	Number of sentences
Date	56,520,912
Duration	8,182,819
Set	1,797,929
Time	2,281,198

Table 4: TSM data statistics: The above table describes the distribution of temporal spans in the English Wikipedia data, which comprises of 121M sentences.

Span Type	Number of sentences
Entity	78,139,341
Date	32,023,769

Table 5: SSM data statistics: The above table describes the distribution of salient spans in the Wikipedia data as processed by (Guu et al., 2020), which comprises of 82M sentences. Each row denotes the number of sentences that contain at least one of the respective span.

TSM Span Type	SSM Span Type	
	Named Entity	Date
Date	29,771,242	-
Duration	5,159,229	3,144,592
Set	1,226,333	844,222
Time	915,084	411,436

Table 6: We apply SUTIME on the SSM training data (Guu et al., 2020) to investigate how many sentences contain temporal information. Each column denotes the number of sentences that contain the SSM identified span (e.g. *named entity* or *date*) and each row denotes the number of those sentences in which SUTIME identified the corresponding temporal span. Number of sentences with at least one *named entity*: 78,139,341 Number of sentences with at least one *date*: 32,023,769 (Table 5)

B.2 Span Distribution vs. Text Distribution

Our hypothesis for SSM's effectiveness is due to it oversampling difficult sentences. This is based on the performance gain for the ENTITIES intermediate training as well as the number of temporal spans that occur in the SSM training data. Table 6 shows the results of SUTIME parser on the SSM training data, and as we can see, significant portion

---

[4] https://github.com/google-research/text-to-text-transfer-transformer

of the SSM data (45%) has temporal spans. Table 6 shows the breakdown of different temporal spans for each SSM salient span type. We leave an exact test of this for future work, but if this is true, then we might expect left-to-right decoding models to also benefit from the sampling procedure of SSM, even though they do not use a masked language modeling paradigm for training.

B.3 Span Types in Table 4

Mapping MC-TACO's span types is somewhat helpful to see the performance breakdown. Note that these are now based on individual answers, while MC-TACO's strict match metric is based on correctly labeling all answers for a given question.

Entities While MC-TACO is a common sense dataset, it frequently relies on reasoning about relatively complicated phenomena. While it is common sense to know that a dynasty does not rule in China for only a few minutes, it is still required to know more about China and dynasties to answer the question correctly. TimeDIAL on the other hand is normally ordinary conversations that are not very entity-centric. SituatedQA is derived from Natural Questions, which is an information seeking dataset that frequently features entities.

Duration MC-TACO's event duration maps well to the Duration type in SUTIME. While there may be some SituatedQA examples that include durations, we do not filter for them.

Set MC-TACO's Frequency type asks question of the "How often" nature while sets frequently have answer types of that nature e.g., "every third sunday", but this is not a perfect mapping.

Date MC-TACO's typical time sometimes includes dates, but it is less likely to be a specific date and more likely to be a generic date like Sunday, rather than a specific knowledge-based date. SituatedQA questions always include dates that decontextualize Natural Questions.

Time MC-TACO's typical time sometimes corresponds with times as well, but they are again less likely to be specific. Unfortunately, Date and Time are not separated in MC-TACO.

Other MC-TACO Types Note that we did not include the "Stationarity" or the "Event Ordering" MC-TACO types in the breakdown, as they do not correspond well to any SUTIME type.

PECO: Examining Single Sentence Label Leakage in Natural Language Inference Datasets through Progressive Evaluation of Cluster Outliers

Michael Saxon, Xinyi Wang, Wenda Xu, William Yang Wang
University of California, Santa Barbara
Department of Computer Science
saxon@ucsb.edu, xinyi_wang@ucsb.edu, wendaxu@ucsb.edu, william@cs.ucsb.edu

Abstract

Building natural language inference (NLI) benchmarks that are both challenging for modern techniques, and free from shortcut biases is difficult. Chief among these biases is *single sentence label leakage*, where annotator-introduced spurious correlations yield datasets where the logical relation between (premise, hypothesis) pairs can be accurately predicted from only a single sentence, something that should in principle be impossible. We demonstrate that despite efforts to reduce this leakage, it persists in modern datasets that have been introduced since its 2018 discovery. To enable future amelioration efforts, introduce a novel model-driven technique, the progressive evaluation of cluster outliers (PECO) which enables both the objective measurement of leakage, and the automated detection of subpopulations in the data which maximally exhibit it.

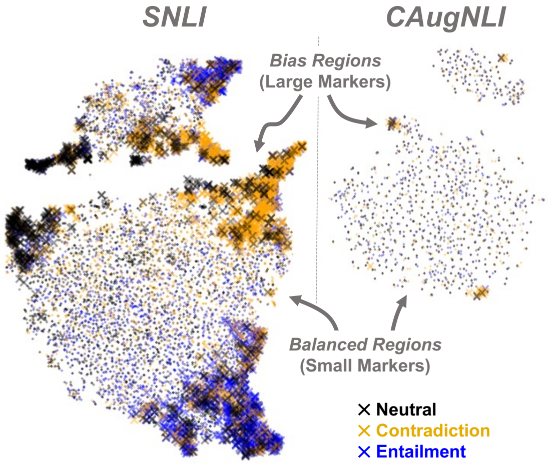


Figure 1: A T-SNE projection of the SNLI and CAugNLI test sets in PECO's model-driven *single sentence condition* (SSC) embedding space, showing **entailment**-, **contradiction**-, and **neutral**-labeled samples. This model was trained on the paired-sentence condition (PSC) where the relation between sentences in a sample are observable. In a leakage-free dataset, a model should be unable to separate subpopulations that disproportionately exhibit one label class in the SSC. Local regions exhibiting an imbalanced label distribution in this subpopulation are considered "biased regions" and plotted with large markers. SNLI exhibits large, continuous regions in the hypothesis only space disproportionately exhibiting the same label, compared to CAugNLI. Accordingly, SNLI exhibits higher single sentence label leakage than CAugNLI (Table 3).

1 Introduction

Natural language inference (NLI) is a fundamentally pairwise task, wherein a logical relation between two statements is predicted. Progress on NLI benchmarks is an important proxy for advancements in natural language reasoning by machines. Models are trained to process the two statements simultaneously, in a *paired sentence condition* (PSC). Unfortunately many modern NLI datasets exhibit **single sentence label leakage**. When this leakage is present, models are able to accurately predict the pairwise relation encoded by the labels in a *single sentence condition* (SSC)—where the model is only shown one of the statements (Poliak et al., 2018). This is a serious problem, rendering NLI datasets' capture of reasoning questionable, and limiting the robustness of models trained on them.

NLI is formalized as predicting a relation $r \in \{\text{neutral, entail, contradict}\}$ from a pair of sentences (premise s_1 and hypothesis s_2). An ideal NLI benchmark without single sentence label leakage will have distribution of r that is conditionally dependent on the **pair** of sentences, but independent from either individual sentence (Wang et al., 2021c). In practice this is difficult to achieve, particularly when constructing usefully large datasets.

Most large-scale NLI datasets are produced by sourcing *seed* sentences from an existing text population to serve as initial premises or hypotheses. Each seed sentence is then assigned one or more relations r, which annotators use to write new *elicited* sentences satisfying each selected relation relative to the seed. Many datasets exclusively build either the hypothesis or premise population from seed sentences, leaving the other exclusively elicited.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3061–3074
May 2-6, 2023 ©2023 Association for Computational Linguistics

Dataset	Authors (Year)	Seed sentences from	Language	Imbalance on	# Train	# Dev	# Test
SICK	Marelli et al. (2014)	Image+Video Captions	En	s_2	4.4k	0.5k	4.9k
SNLI	Bowman et al. (2015)	Image Captions + KB	En	s_2	550k	10k	10k
MNLI	Williams et al. (2018)	Multiple Genre	En	s_2	393k	20k	20k
↪MNLI-m	│	│	│	│	—	10k	10k
↪MNLI-u	│	│	│	│	—	10k	10k
XNLI	Conneau et al. (2018)	MNLI	14 Langs.	s_2	—	70k	35k
FEVER$_{\text{NLI}}$	Nie et al. (2019)	Wikipedia	En	s_1	208k	20k	20k
ANLI	Nie et al. (2020)	Wikipedia+HotpotQA	En	s_2	163k	3.2k	3.2k
↪A1	│	│	│	│	17k	1k	1k
↪A2	│	│	│	│	45k	1k	1k
↪A3	│	│	│	│	100k	1.2k	1.2k
OCNLI	Hu et al. (2020)	Multiple Genre	Zh	s_2	47k	3k	3k
CAugNLI	Kaushik et al. (2020)	IMDb+SNLI	En	s_2 & s_1	8.3k	1k	8.3k
SNLI$_{\text{debiased}}$	Wu et al. (2022)	SNLI, generated	En	s_2 & s_1	1.14M	—	—
MNLI$_{\text{debiased}}$	│	MNLI, generated	│	│	744k	—	—

Table 1: Information on the NLI datasets we compare in this study. MNLI has "matched" (m) and "unmatched" (u) test sets that we evaluate separately. The ANLI dataset is decomposed into partitions "A1," "A2," and "A3". A Vertical line denotes identical value in cell as above (sub-elements of same dataset).

Systematic, shared biases in the words, sentence structures, or ideas that crowdworkers consider when given a logical relation, coupled with the exclusively elicited nature of one of the sentence populations, then drive relation leakage (Gururangan et al., 2018). For example, a slight preference for words like "not" or "doesn't" when given *contradict* as opposed to *entail* would lead to a bias in the n-gram distribution between the classes in the SSC. Simple heuristics inspired by these findings can produce challenging test sets that hobble models trained on these biased datasets (McCoy et al., 2020), but they require manual guesswork, don't generalize, and may miss higher-level, more nuanced semantic shortcuts and biases.

These "leakage features" encoded in the elicited sentences are visible to NLI models (Zhang et al., 2019), enabling them to "cheat" by attending to them as shortcuts rather than logical correspondences between the two sentences, calling into question the appropriateness of NLI datasets as benchmarks for language understanding (Bowman and Dahl, 2021). In this work we rigorously analyze this problem of single sentence relation leakage in both popular and recent NLI datasets using novel techniques to enable targeted interventions and create higher quality future resources.

Further NLI datasets have been proposed to tackle these problems using machine-in-the-loop adversarial sentence elicitation, (Nie et al., 2020), counterfactual augmentation (Kaushik et al., 2020), and learning dynamic-based debiasing (Wu et al., 2022). These datasets are purported to provide more challenging generalization scenarios for NLI models to better test logical reasoning capabilities. One big question remains—*have these techniques actually eliminated relation leakage biases?*

In this work, we demonstrate the following:

New NLI datasets still exhibit single sentence relation leakage. We compare SSC performance for 10 NLI datasets (including those previously assessed by Poliak et al. (2018)) using a simple transformer baseline, finding that **single sentence relation leakage remains a severe problem**.

NLI models still use the leakage features to cheat. We analyze the datasets using *output decision agreement* and *input token importance* statistics between models trained in the SSC and PSC to demonstrate this.

Automated leakage feature detection is feasible. We introduce a novel model-based metric and dataset analysis tool, the *Progressive Evaluation of Cluster Outliers* (PECO) (Figure 1), for examining the degree of single sentence relation leakage and eliminating it in future datasets.[1]

2 Quantifying NLI Dataset Bias

An ideal NLI benchmark is neither "saturated" nor biased. Saturated benchmarks are datasets for which current approaches already achieve high accuracy. They are "solved" and have limited utility in tracking future progress (Bowman and Dahl, 2021). We refer to as *biased* any NLI benchmarks that exhibit significant single-sentence relation leakage through high achievable SSC accuracy in at least one single sentence condition.

---

[1] Code at `github.com/michaelsaxon/DatasetAnalysis`, and an animated demo is available at `saxon.me/peco`.

			Accuracy (% PSC)			Hparams	
Dataset	SOTA Model	Replication Model	SOTA	Ours	Maj.	LR	Batch
SICK	NeuralLog (Chen et al., 2021)	`roberta-large`	90.3	87.8	56.0	1e–5	18
SNLI	EFL (Wang et al., 2021b)	`roberta-large`	93.1	90.6	33.8	5e–6	128
MNLI-u	T5 (Raffel et al., 2019)	`roberta-large`	92.0	88.7	35.6	5e–6	12
MNLI-b	│	│	91.7	88.3	36.5	5e–6	12
XNLI	ByT5 (Xue et al., 2021)	`xlm-roberta`	83.7	73.7	33.3	1e–5	16
FEVER	KILT (Petroni et al., 2021)	`roberta-large`	86.3	74.7	33.3	1e–5	8
ANLI	InfoBERT (Wang et al., 2021a)	`roberta-large`	58.3	54.0	33.5	1e–6	18
↪A1	│	│	75.5	61.9	33.4	1e–6	18
↪A2	ALBERT (Lan et al., 2020)	│	58.6	50.0	33.4	1e–6	18
↪A3	│	│	53.4	49.8	33.5	1e–6	18
OCNLI	`RoBERTa-wwm-ext-l` (Xu et al., 2020)	`bert-base-chinese`	78.2	71.8	36.8	1e–5	128
CAugNLI	— (Ours)	`roberta-large`	84.7	84.7	33.9	5e–6	64
SNLI$_{\text{debiased}}$	│	`roberta-large`	95.6	95.6	35.5	1e–5	64
MNLI$_{\text{debiased}}$	│	`roberta-large`	96.9	96.9	36.2	1e–5	16

Table 2: For each dataset we analyze, the current state of the art (SOTA) model, base pretrained LM we use for replication, and accuracy for SOTA, our replication (Ours), and majority label-only (Maj.) classification in the standard paired-sentence NLI classification condition (PSC) with selected best-performing hyperparameters.

We analyze 10 datasets containing a total of 14 test or validation sets in terms of *biasedness* and *saturatedness*, across 17 SSC conditions (premise-only (s_1) or hypothesis-only (s_2)). Table 1 provides an overview of this information along with statistics such as train/dev/test set size, and which sentence population is potentially unbalanced.

Each dataset D is composed of (s_1, s_2, r) tuples. We use the standard notation $(X_i, Y_i) \leftarrow D$ to describe the samples, as depending on whether the training condition is standard PSC or SSC, each X_i can be (s_{1i}, s_{2i}), s_{1i}, or s_{2i}. Models trained in condition c are referred to as f_c.

2.1 Saturation and Bias Scoring

We assess accuracy on the test set (or val set when no labeled test set is available) for each dataset in paired- and single-sentence conditions.

Saturation (Accuracy): We report state-of-the-art (SOTA) model performance results in the PSC,

$$\mathrm{A}_{\mathrm{SOTA}}(D) = P(f_{\mathrm{SOTA}}(X_{\mathrm{test}}) = Y_{\mathrm{test}}); \quad X, Y \in D \quad (1)$$

For our model-level and sample-level comparative analysis between the PSC and SSC we train our own transformer-based models using a simple procedure (Sec. 2.4) to assess *replication* accuracy,

$$\mathrm{A}_{\mathrm{PSC}}(D) = P(f_{\mathrm{PSC}}(X_{\mathrm{test}}) = Y_{\mathrm{test}}); \quad X, Y \in D \quad (2)$$

SSC Accuracy: For each elicited population SSC we train a model f_{SSC} according to the procedure described in Sec. 2.4.2, to assess $\mathrm{A}_{\mathrm{SSC}}$:

$$\mathrm{A}_{\mathrm{SSC}}(D) = P(f_{\mathrm{SSC}}(X_{\mathrm{test}}) = Y_{\mathrm{test}}); \quad X, Y \in D \quad (3)$$

Table 2 shows current SOTA models and results for the 10 datasets, as well as our PSC model performance and the relevant training hyperparameters (more detail in Sec. 2.4). Figure 2 shows SSC accuracy against replication (PSC) accuracy for each dataset. Datasets that exhibit higher SSC accuracy have worse single sentence relation leakage, and are thereby questionable in their ability to capture reasoning abilities. Ideally, an **optimal benchmark** for NLI will both have low maximum SSC accuracy and low maximum PSC accuracy (room for future model growth).

These absolute measures of dataset bias and saturation are useful targets for future optimal benchmarks, it is important to understand how these measures interact with each other.

2.2 Relative Dataset Bias Scoring

We assess two relative dataset bias scores.

SSC Improvement over Chance: We subtract the SSC test accuracy from the accuracy achieved by a "guess majority label" strategy following (Poliak et al., 2018):

$$\Delta_{\mathrm{maj}} = P(f_{\mathrm{SSC}}(X_{\mathrm{test}}) = Y_{\mathrm{test}}) - P(Y_{\mathrm{maj}} = Y_{\mathrm{test}}) \quad (4)$$

This metric gives an insight into single sentence relation leakage that compensates for datasets (such as SICK) with an uneven base label distribution.

SSC-PSC Accuracy Recovered: Accuracy achieved by SSC model over PSC:

$$\%\mathrm{R}_{\mathrm{R}} = \frac{P(f_{\mathrm{SSC}}(X_{\mathrm{test}}) = Y_{\mathrm{test}})}{P(f_{\mathrm{PSC}}(X_{\mathrm{test}}) = Y_{\mathrm{test}})} \quad (5)$$

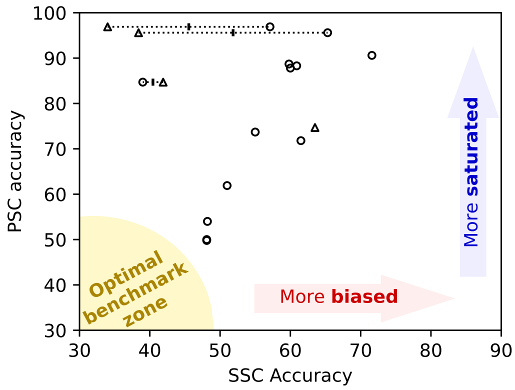


Figure 2: Where the existing NLI datasets fall on the SSC accuracy/PSC accuracy axis. An ideal benchmark isn't saturated (lower SOTA accuracy) and also is unbiased (low SSC accuracy). △ markers represent premise-only SSC results and ∘ markers represent hypothesis-only SSCs. For datasets exhibiting leakage in both conditions, the two points are connected by a dotted line with a | marker for their average SSC accuracy.

Dataset	Cond.	SSC	%R$_{\text{R}}$	Δ_{maj}
SICK	s_2	60.0	68.3	4.0
SNLI	s_2	71.6	79.0	37.8
MNLI-b	s_2	59.8	67.4	24.2
MNLI-u	│	60.9	69.0	24.4
XNLI	s_2	55.0	74.6	21.7
FEVER	s_1	63.5	85.0	30.2
ANLI	s_2	48.2	89.3	14.7
↪A1	│	67.5	82.4	17.6
↪A2	│	82.1	96.2	14.7
↪A3	│	90.1	96.6	14.6
OCNLI	s_2	61.5	85.7	24.7
CAugNLI	s_1	41.9	49.5	8.0
│	s_2	39.0	46.0	5.1
SNLI$_{\text{debiased}}$	s_1	45.3	45.3	2.9
│	s_2	65.3	68.3	29.8
MNLI$_{\text{debiased}}$	s_1	34.0	35.6	-2.2
│	s_2	57.1	58.9	20.9

Table 3: For each NLI dataset and potential leakage-exhibiting single-sentence condition (Cond.) we report *test* accuracy in the single sentence condition (SSC), and three derived metrics from Sec. 2.1. SOTA test accuracy recovery (%R$_{\text{S}}$), replication test recovery (%R$_{\text{R}}$), and biased condition improvement over the chance majority guessing strategy (Δ_{maj}).

This metric captures how similarly the single sentence and normal condition models perform.

2.3 Biased Model Results

Table 3 shows the extent of the single-sentence relation leakage problem across the 17 SSC tests on the 14 splits for the 10 NLI datasets. These results clearly show that *each dataset exhibits significant single-sentence relation leakage* for at least one condition. The comparison columns replication test recovery (%R$_{\text{R}}$), and SSC improvement over the chance majority guessing strategy (Δ_{maj}) are all computed using the single sentence condition accuracy and the standard NLI two-sentence condition SOTA and replication accuracy values in Table 2.

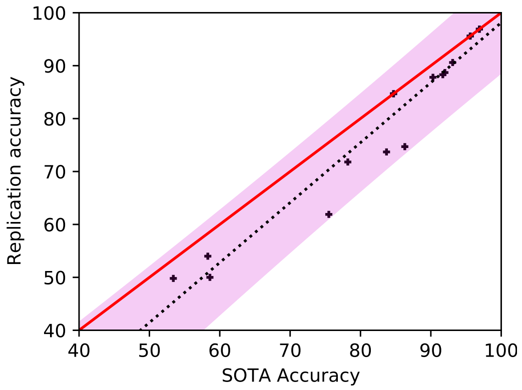


Figure 3: Test set accuracy for SOTA models vs our universal replication procedure models for each dataset in PSC, with a trendline (PCC=0.97) and $y = x$ line.

2.4 Model Training

Our training technique is simple and applied consistently to all datasets. We fine-tune three different language-specific pretrained transformer checkpoints[2] from `HuggingFace` (Wolf et al., 2019) using `Pytorch Lightning`. All models were trained on NVIDIA A-100 GPUs. All models are the HuggingFace `xForSequenceClassification` with `num_classes=3` and no other modifications. All models are trained using the Adam optimizer with cross entropy loss.

We find that this procedure produces broadly near-SOTA performance models, with a maximum relative accuracy difference of 8%, and a 92% Pearson's correlation coefficient (PCC) between SOTA and replication accuracy across the datasets (Figure 3). **Our replications are a reasonable proxy to SOTA for comparative dataset analysis**.

2.4.1 Replication Training Details

Hyperparameters For each dataset in the paired sentence condition (PSC), we select a batch size for maximum GPU utilization. We perform a grid search over lr in {5e–7, 1e–6, 5e–6, 1e–5, 5e–5}.

Single Dataset Fine-tuning We obtain separate fine-tuning checkpoints from the pretrained models for each dataset to enable clean analysis of one dataset at a time. We do not accumulate fine-tuning passes across multiple datasets.

---

[2] `roberta-large` (Liu et al., 2019), `xlm-roberta` (Conneau et al., 2019), `bert-base-chinese` (Devlin et al., 2018)

2.4.2 Single Sentence Condition Training

To train each dataset's corresponding **SSC** model(s), we use the same setup as the PSC model but follow Poliak et al. (2018)'s formulation of fine-tuning the chosen classification model on only the SSC sentence, *premise only* or *hypothesis only*.

FEVER exhibits bias in the premise distribution, and CAugNLI, $\text{SNLI}_{\text{debiased}}$, and $\text{MNLI}_{\text{debiased}}$ exhibit imbalance in both (Table 1). For the datasets that have imbalanced distributions in both conditions, we separately train bias models for both hypothesis-only and premise-only.

3 Analyzing NLI Dataset Bias

In this section we introduce quantification techniques for more accurately characterizing the extent of these bias problems in the aforementioned NLI datasets, analyze how they interact with the observable bias itself, and develop tools for producing future NLI benchmarks that more closely resemble the ideal benchmark.

3.1 Sample-level Model Behavior

We are particularly interested in understanding the degree to which models trained in the PSC and SSCs "reason" similarly. For this section we use the notations $f(\boldsymbol{X}_{\text{test}})$, $\boldsymbol{Y}_{\text{test}}$ to denote the $(1 \times N)$ column vectors of model output decisions and labels for a test set of N samples, and a simple agreement function $Ag(\boldsymbol{Y}_1,...,\boldsymbol{Y}_n)$ as the ratio of elements that are identical across all $\boldsymbol{Y}_i$ to the vector size N. In other words,

$$Ag(f(\boldsymbol{X}_{\text{test}}),\boldsymbol{Y}_{\text{test}}) = P(Y_{\text{test}} = f(X_{\text{test}})) \tag{6}$$

SSC-PSC Agreement (NBA): The number of samples for which the SSC and PSC models agree over the total number of samples in the set:

$$\text{NBA} = \frac{Ag(f_{\text{SSC}}(\boldsymbol{X}_{\text{test}}), f_{\text{PSC}}(\boldsymbol{X}_{\text{test}}))}{|\boldsymbol{X}_{\text{test}}|} \tag{7}$$

SSC-PSC Recovery (NBR): The number of samples for which the SSC and PSC models agree, *and both classify correctly* over the total number of samples they agree on:

$$\text{NBR} = \frac{Ag(f_{\text{SSC}}(\boldsymbol{X}_{\text{test}}), f_{\text{PSC}}(\boldsymbol{X}_{\text{test}}), \boldsymbol{Y}_{\text{test}})}{Ag(f_{\text{SSC}}(\boldsymbol{X}_{\text{test}}), f_{\text{PSC}}(\boldsymbol{X}_{\text{test}}))} \tag{8}$$

Token Relevance Agreement (TRA): Do SSC and PSC models reason alike? For a sentence X with length n, we compute the gradient of the classification output posterior with respect to each token embedding $emb(w_j)$. We take the 2-norm of the each gradient vector and normalize it over the entire sequence to produce a normalized local explanation vector $m(f(X))$ (Sundararajan et al., 2017):

$$m(f,X) = \left[\frac{|\nabla_{\text{emb}(w_j)}(f(\text{emb}(w_j)))|_2}{\sum_{i=1}^{n}(|\nabla_{\text{emb}(w_i)}(f(\text{emb}(w_i)))|_2)}\right]_{j=1}^{n} \tag{9}$$

To compare "reasoning" similarity between the two models, we compute the samplewise input token relevance agreement can be computed using cosine similarity:

$$\text{TRA}(X_i) = \frac{m(f_{\text{PSC}},X_i) \cdot m(f_{\text{SSC}},X_i)}{||m(f_{\text{PSC}},X_i)||\,||m(f_{\text{SSC}},X_i)||} \tag{10}$$

As the SSC and PSC inputs have different lengths, we pad the SSC importance vector for $m(f_{\text{SSC}}(X_i))$ with zeros either prepended or postpended (depending on if the SSC is hypothesis- or premise-only) to make the two local explanation map vectors of equal length. The dataset-level token relevance agreement is the average of samplewise TRA.

3.2 Cluster-based Bias Evaluation

We are interested in investigating how the biased distributions of the elicited sentences in NLI datasets are captured in the learned representation spaces of models trained on them. In particular, we are interested in answering this question: **is elicited sentence label leakage captured semantically in regions of latent space**?

To answer this we produce dimensionality-reduced *elicited sentence embeddings* for the test set, using the PSC replication models, then fit a high-k k-means clustering to this collection of embeddings. This will allow us to analyze how the local distribution of labels varies over the elicited sentence embedding space. By comparing the KL-divergence of the label distribution within each cluster and the global label distribution, we can compute the **Progressive Evaluation of Cluster Outliers** (PECO) score (Figure 4).

Elicited Sentence Embeddings: To embed the elicited sentences *as they're learned by a model in the PSC*, we feed the elicited sentences s_e through the PSC replication fine-tuned NLI model encoder. We extract the latent codes produced at the output very last fully connected layer of the model before

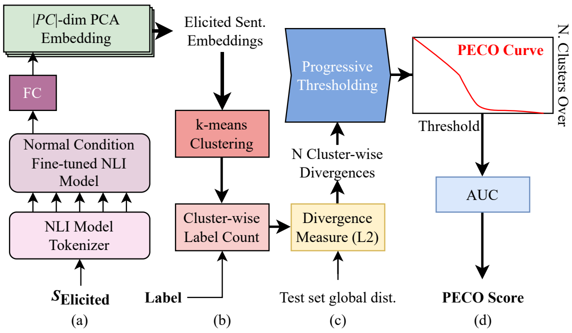


Figure 4: An overview of the approach to computing the PECO score from a collection of elicited population sentences s_{elicited} and their corresponding Labels r. When a fixed threshold is chosen, the Hypothesis embeddings can be dimensionality-reduced using T-SNE to produce plots like Figure 1.

the linear classifier to collect latent codes for every s_e in the test set. We then embed these codes into their 30 principal components to produce the embeddings (Figure 4 (a)).

Clustering: We fit a high-k (in this case, $k = 50$) k-means clustering over the distribution of elicited sentence embeddings to provide a set of local bins for analysis. For each cluster, we count the relation labels its samples contain, to produce a set of 50 cluster-label distributions (Figure 4 (b)).

Computing Cluster Divergences: For each cluster label distribution $p_i = P(Y|\text{cluster} = i)$, we assess the L2 divergence between it and the global label distribution p_{G} to produce divergence scores s_i:

$$s_i = \frac{1}{3}\sum_{j=1}^{3}(P(Y = j) - P(Y = j|\text{cluster} = i))^2 \quad (11)$$

This step is depicted in Figure 4 (c).

Progressive Evaluation: Finally, we compute the PECO score for this collection of cluster divergences as the area under the curve produced by counting the number clusters with divergence s_i over some threshold t for the range of s_i.

$$\text{PECO} = 100\int_{\min(s)}^{\max(s)} \frac{\text{count}_i(s_i > t)}{k}dt \quad (12)$$

Generality of PECO: These same techniques could be applied to a wide variety of potential leakage features on the input to analyze a wide variety of correlation types. For example, input sentence words could be shuffled to test for word order invariance, or word classes could be specifically masked to test for spurious vocabulary correlations.

Dataset	Cond.	NBA	NBR	TRA	PECO
SICK	s_2	48.8	49.7	63.6	13.9
SNLI	s_2	70.0	71.8	63.9	14
MNLI-b	s_2	32.5	33.4	57.6	6.5
MNLI-u	|	47.5	48.6	59.3	7.5
XNLI	s_2	52.3	54.0	64.7	9
FEVER	s_1	39.4	38.0	50.7	14.3
ANLI	s_2	37.6	53.4	28.0	5.5
↪A1	|	53.8	57.9	54.0	15.3
↪A2	|	53.9	60.3	51.2	11.1
↪A3	|	52.1	58.5	55.7	15.9
OCNLI	s_2	69.0	74.1	78.5	17.9
CAugNLI	s_1	32.9	32.7	39.5	5.5
|	s_2	43.4	42.5	59.7	5.4
SNLI$_{\text{debiased}}$	s_1	32.0	31.8	41.5	8.2
|	s_2	60.9	61.4	58.4	8.3
MNLI$_{\text{debiased}}$	s_1	34.7	34.0	46.7	6.6
|	s_2	42.2	42.0	58.0	6.8

Table 4: Metrics comparing the behavior of our replication and single-sentence condition models on each dataset using the metrics introduced in Secs. 3.1, 3.2: Normal-Bias Agreement (NBA) and Recovery (NBR), Token Relevance Agreement (TRA) and the Progressive Evaluation of Cluster Outliers (PECO) score.

PECO Parameter Choices: We discuss the impact of PECO parameters (e.g., choice of k-means, number of principal components to reduce to, use of L2 or KL-divergence) in Appendix C.

4 Results

As discussed above, ideal NLI benchmarks are neither *saturated* nor *biased*. Unfortunately, as Table 3 demonstrates, none of the NLI datasets tested thus far satisfy this condition. This is more clearly illustrated in Figure 2. Two questions remain: *to what extent do current models cheat* and *how can we make less biased, less saturated datasets?* Table 4 contains experimental results intended to answer

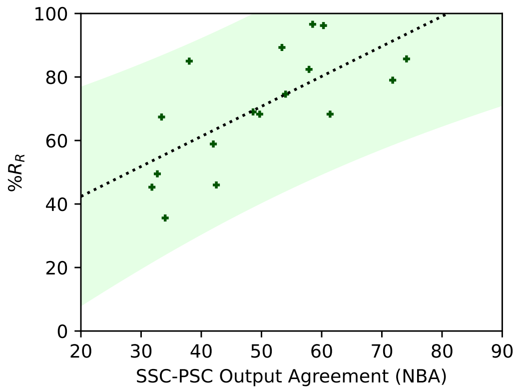


Figure 5: Model-wise output agreement vs Bias accuracy recovery ($\%R_R$). As the replication PSC model and the SSC model agree more often for a given dataset, their performances in the two conditions converge.

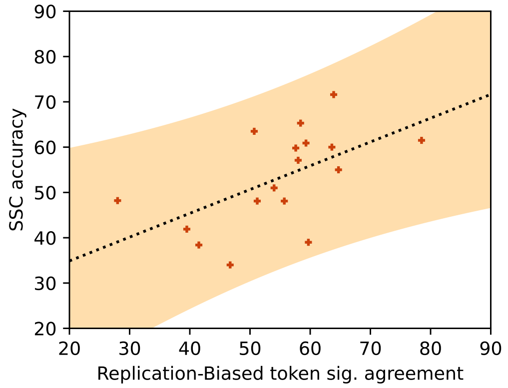


Figure 6: Token releveance agreement (TRA) vs SSC accuracy. When the PSC model and SSC models "reason" more similarly for a dataset, the relation leakage bias exhibited in that dataset tends to be higher.

these two questions. The "agreement metrics" as introduced in Sec. 3.1, SSC-PSC Agreement, SSC-PSC Recovery, and Token Relevance Agreement are provided in Table 4.

4.1 Result-Metric Correlations

We find that SSC-PSC model output agreement (NBA) and recovery rate $\%\mathrm{R}_\mathrm{R}$ are **correlated with a PCC of 0.69** (Figure 5). Datasets where the SSC and PSC models predict more similarly have closer the SSC and PSC performance results are for said datasets. While this result is surprising, ANLI R3 is an interesting outlier (section 5).

We find that TRA and SSC accuracy are also **positively correlated with a PCC of 0.57** (Figure 6). This result demonstrates that for a single dataset, similar reasoning patterns for the single sentence condition and standard sentence pair condition is strongly correlated to single-sentence relation leakage. In other words, **standard condition NLI models trained on biased (high leakage) datasets tend to cheat**. Thus, models indeed rely on annotation artifacts in NLI datasets to achieve high accuracy, and demonstrates that this continues to be a problem in newer NLI datasets, in spite of mitigation attempts. How can we use this knowledge to build better benchmarks?

Figure 7 depicts the relationship between PECO score and bias recovery ($\%\mathrm{R}_\mathrm{R}$). We find the two are **positively correlated with a PCC of 0.64**. This result is fairly intuitive: the more uneven the distribution of labels is in the single-sentence latent spaces (and thus, the higher the area under the PECO curve), the more SSC performance approaches the standard PSC condition performance for a given NLI dataset. This suggests PECO-reducing interventions may be able to target debiasing efforts.

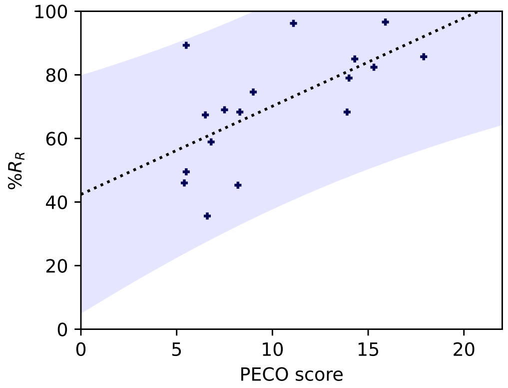


Figure 7: Test-set PECO score vs Bias accuracy recovery ($\%R_R$). This result suggests that interventions that produce lower PECO score datasets tend to yield datasets that exhibit less relation leakage in the SSC.

5 Discussion

Some examples are only correctly classified in the single-sentence condition. A common assumption to discussions of cheating features in machine learning is that they play a role in inflated classification accuracy when present. However, ANLI R3 provides an interesting counterexample. For this dataset, the hypothesis-only model achieves a SSC accuracy of 48.1%, and the PSC model achieves 49.8% (a $\%\mathrm{R}_\mathrm{R}$ of 90%), and SOTA achieves 53%. Despite this score similarity, the samples which the two conditions are able to actually classify correctly vary surprisingly. With an NBR of 58.5%, only ≈ 27% of test samples are correctly classified by both the single and two sentence condition models. This means that around **21% of samples in A3 test are *only* correctly classified by the single-sentence model**.

Perhaps unsurprisingly, ANLI exhibits the lowest TRA out of all datasets tested, indicating that it is somewhat of an outlier in having the SSC and PSC condition models reason differently on it.

XNLI demonstrates the cross-lingual and semantic nature of single-sentence leakage. While previous work has focused on finding words, phrases, patterns, and heuristics in the surface form of the data, our study of XNLI provides an interesting opportunity to investigate the potential for the influence of underlying semantics as a leakage feature. XNLI consists exclusively of 14 language test and val sets, manually translated from MNLI examples. Our XNLI PSC and SSC models are thus trained on MNLI alone, using the multilingual `xlm-roberta` checkpoint.

This produces a natural experiment wherein surface form biases present in the training data are completely eradicated in the test set (as only the 14 non-English languages Table 1 are present), while the underlying meanings encoded by those words remain. In Table 3 we indeed find that XNLI and MNLI exhibit very similar result comparisons. The models on both datasets have a $\Delta_{\text{max}} \approx 20\%$ and $\%R_R \approx 65$. These leakage feature results, being robust to manual translation into 14 different languages, seem to indicate that there is a strong fundamental semantic component to the human biases driving the elicited sentence relation leakage.

Relation leakage remains Unsolved. Elicited sentence relation leakage is a problem for all evaluated NLI datasets, including the new ones intended to fix it. Recent datasets such as XNLI, FEVER, and OCNLI, exhibit high absolute SSC performance over majority ($\Delta_{\text{maj}} > 20$).

Although ANLI and CAugNLI are improvements over the others in terms of Δ_{maj}, with CAugNLI shining particularly in this regard, none eliminate the relation leakage problem entirely, as even CAugNLI still has $\Delta_{\text{maj}} = 8.0$, an 8% performance over chance in the single sentence condition.

$\text{SNLI}_{\text{debiased}}$ and $\text{MNLI}_{\text{debiased}}$, despite their intended purpose, still contain significant amounts of SS label leakage (29.8% and 20.9% over chance). This might be because while their production (Wu et al., 2022) does eliminate bias *originally present in SNLI and MNLI*, it fails to prevent the introduction of *new bias* in the data generation pipeline.

Cluster approaches are promising for future debiasing efforts. Figure 1 shows how the PECO-derived cluster-bias T-SNE plots can be used directly to visualize, analyze, and "debug" biased datasets. In the plot, SNLI clearly has considerably more high-biased clusters taking up a considerable portion of the latent space as compared to CAugNLI, for the bias threshold of 0.2.

An intervention could be performed on identified bias regions in the distribution by having human annotators create new premise sentences from the given hypotheses, thereby forcing the PECO-based bias metrics to reduce. This idea is further backed up by the PCC of 35.8 that we find between PECO and $\%\text{R}_\text{R}$, suggesting that producing datasets of lower PECO score will naturally lead to lower recovered performance in the SSC, and thus less elicited sentence relation leakage.

6 Related Work

Understanding Bias in NLI Huang et al. (2020) demonstrated that counterfactual augmentation alone cannot debias NLI. Multi-task learning can improve model robustness to fitting spurious features (Tu et al., 2020), but because the underlying **benchmarks are biased**, progress on the desired reasoning capability is questionable (Poliak et al., 2018). Geva et al. (2019) strengthen the finding that annotator bias is a key driver of this poor generalization performance, showing that NLI models can struggle to even generalize across disjoint sets of annotators on the same task.

Simple word- and n-gram level approaches have proven surprisingly capable in a-priori characterizations of dataset difficulty (McKenna et al., 2020) and producing difficult test sets (Saxon et al., 2021) in diverse language domains such as SLU. Gardner et al. (2021) show how such purely frequentist approaches can identify word-level spurious correlations with respect to label class which drive in-part the shortcut features for classes of "competency problems" such as NLI.

Mitigating Bias in NLI Belinkov et al. (2019) demonstrate an approach to train NLI models robustly against some of these biases, using Gururangan et al. (2018)'s hard test set. McCoy et al. (2020) utilize simple heuristics like lexical overlap to produce the synthetic debiased HANS NLI dataset to test generalization. This dataset has been used to evaluate techniques including predicate-argument- (Moosavi et al., 2020) and syntactic transformation-based (Min et al., 2020) augmentations. Zhou and Bansal (2020) leverage a bag-of-words approach to debias datasets along lexical features. However, these approaches have yet to improve generalization in comprehensive replication studies has thus far (Bhargava et al., 2021). Meanwhile Varshney et al. (2022) propose a fully unsupervised data collection pipeline for NLI, in order to sidestep the problem of human biases entirely.

Approaches like (Kaushik et al., 2020) and (Wu et al., 2022) are very promising for producing data that reduces bias on a samplewise, but not populationwise level. Using our semantic, model-driven local bias finding strategies, future interventions can lead to the large scale production of debiased NLI datasets and a new generation of higher quality benchmarks for language understanding. Liu et al. (2022) perform such a targeted augmentation approach using the *dataset cartography* sample

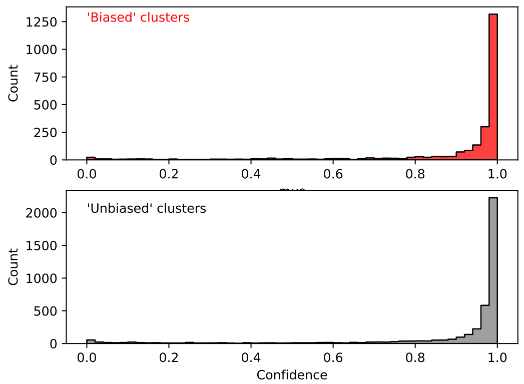


Figure 8: Histograms for the samplewise *confidence* feature from dataset cartography in SNLI, for high- (biased) and low-PECO$_{L2}$ clusters.

characterization scheme from Swayamdipta et al. (2020) to produce WANLI, an NLI dataset that allows for improved performance on the aforementioned challenging test sets.

6.1 PECO vs Dataset Cartography

To determine if PECO-driven dataset augmentation is redundant given the recent release of WANLI, we seek to determine if usable samplewise information for targeting interventions (e.g., presence in a "bias" cluster) is captured during PECO analysis and is redundant to the relevant samplewise characterization produced in dataset cartography.

To do this, we collect the samplewise *confidence* feature (Swayamdipta et al., 2020) during training of the PSC model for each validation set sample in SNLI. We then assign each validation sample to its corresponding PECO cluster (out of the 50) and produce two histograms of the confidence feature, one for **"biased" clusters** (PECO$_{L2}$ > 0.25) and one for the other clusters. Figure 8 shows the results of this experiment. We find that out of the 10k validation set examples, roughly 2 are assigned to an "unbiased" cluster for every 1 assigned to "biased," roughly evenly across all confidence bins.

This result suggests that PECO clusterwise "biasedness" is orthogonal to the samplewise ease of learnability captured by the dataset cartography confidence feature. In other words, we find that **some samples are easy to learn (high confidence) because they are simple, while other samples are easy to learn because the model is using cheating features.** PECO-like analyses will be instrumental in guiding future efforts to eliminate shortcut features in natural language datasets.

7 Conclusion

In the half decade since (Poliak et al., 2018) single sentence relation leakage bias has proven to remain a difficult issue. Efforts to debias NLI have led to datasets that merely exhibit different kinds of bias than those shown before, or less saturated benchmarks that continue to exhibit cheating features. Future work must prioritize reducing observable bias directly using a model-driven approach.

Limitations

Our work is limited primarily by the PECO's reliance on test-set classification. To successfully analyze the train set-only datasets of SNLI$_{\text{debiased}}$ and MNLI$_{\text{debiased}}$, we had to generate our own train/test splits over the data by sampling. Luck in split selection may play a role in the level of observable bias in cases like these. Furthermore, this reliance on observing held-out samples to understand bias in general means that interventions to reduce single sentence label leakage must apply costly multi-fold splitting and analysis, consuming more significant compute resources than would otherwise be needed for other model-driven approaches.

Ethical Considerations

In the short term, progress toward better natural language inference does not appear to lead to significant social risks in its broader impacts. While "underclaiming" progress in natural language processing tasks (e.g. exaggerating the scope or severity of failures of specific models on specific tasks) (Bowman, 2021) may be enabled by this work in the future, our focus on *directly quantifiable and observable single sentence leakage*, use of SOTA-like models (fine-tuned transformers) for analysis, and our side-by-side comparison of our model implementations with SOTA all ensure that our criticisms of current NLI benchmarks are well-founded. All data and tools we utilized were freely distributed for unlimited research use in the academic context.

Acknowledgements

This work was supported in part by the National Science Foundation Graduate Research Fellowship under Grant No. 1650114. We would also like to thank the Robert N. Noyce Trust for their generous gift to University of California via the Noyce Initiative. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the sponsors.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 877–891, Florence, Italy. Association for Computational Linguistics.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in NLI: Ways (not) to go beyond simple heuristics. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 125–135, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Samuel Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855.

Samuel R Bowman. 2021. When combating hype, proceed with caution. *arXiv preprint arXiv:2110.08300*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. The snli corpus.

Zeming Chen, Qiyue Gao, and Lawrence S. Moss. 2021. NeuralLog: Natural language inference with joint neural and logical reasoning. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 78–88, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166, Hong Kong, China. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kuebler, and Larry Moss. 2020. Ocnli: Original chinese natural language inference. In *Findings of EMNLP*.

William Huang, Haokun Liu, and Samuel R Bowman. 2020. Counterfactually-augmented snli training data does not yield better generalization than unaugmented data. *arXiv preprint arXiv:2010.04762*.

Divyansh Kaushik, Amrith Setlur, Eduard Hovy, and Zachary C Lipton. 2020. Explaining the efficacy of counterfactually augmented data. *International Conference on Learning Representations (ICLR)*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vision*, 123(1):32–73.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. Wanli: Worker and ai collaboration for natural language inference dataset creation. In *Conference on Empirical Methods in Natural Language Processing*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2020. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pages 3428–3448. Association for Computational Linguistics (ACL).

Joseph P. McKenna, Samridhi Choudhary, Michael Saxon, Grant P. Strimel, and Athanasios Mouchtaris. 2020. Semantic Complexity in End-to-End Spoken Language Understanding. In *Proc. Interspeech 2020*, pages 4273–4277.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, Online. Association for Computational Linguistics.

Nafise Sadat Moosavi, Marcel de Boer, Prasetya Ajie Utama, and Iryna Gurevych. 2020. Improving robustness by augmenting training sentences with predicate-argument structures. *arXiv preprint arXiv:2010.12510*.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A New Benchmark for Natural Language Understanding. *arXiv:1910.14599 [cs]*. ArXiv: 1910.14599.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Michael Saxon, Samridhi Choudhary, Joseph P. McKenna, and Athanasios Mouchtaris. 2021. End-to-End Spoken Language Understanding for Generalized Voice Assistants. In *Proc. Interspeech 2021*, pages 4738–4742.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *NAACL-HLT*.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An Empirical Study on Robustness to Spurious Correlations using Pre-trained Language Models. *Transactions of the Association for Computational Linguistics*, 8:621–633.

Neeraj Varshney, Pratyay Banerjee, Tejas Gokhale, and Chitta Baral. 2022. Unsupervised natural language inference using PHL triplet generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2003–2016, Dublin, Ireland. Association for Computational Linguistics.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2021a. Info{bert}: Improving robustness of language models from an information theoretic perspective. In *International Conference on Learning Representations*.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021b. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690*.

Xinyi Wang, Wenhu Chen, Michael Saxon, and William Yang Wang. 2021c. Counterfactual maximum likelihood estimation for training deep networks. *arXiv preprint arXiv:2106.03831*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yuxiang Wu, Matt Gardner, Pontus Stenetorp, and Pradeep Dasigi. 2022. Generating data to mitigate spurious correlations in natural language inference datasets. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *CoRR*, abs/2105.13626.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Guanhua Zhang, Bing Bai, Jian Liang, Kun Bai, Shiyu Chang, Mo Yu, Conghui Zhu, and Tiejun Zhao. 2019. Selection Bias Explorations and Debias Methods for Natural Language Sentence Matching Datasets. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4418–4429, Florence, Italy. Association for Computational Linguistics.

Xiang Zhou and Mohit Bansal. 2020. Towards robustifying NLI models against lexical dataset biases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8759–8771, Online. Association for Computational Linguistics.

A Detailed Dataset Info

SICK Sentences Involving Compositional Knowledge (Marelli et al., 2014) was produced by instructing annotators to label existing sourced pairs from 8K ImageFlickr data set (Young et al., 2014) and SemEval 2012 STS MSR-Video Description data set (Agirre et al., 2012). The dataset is in English. Each sentence pair was annotated for relatedness and entailment by means of crowdsourcing techniques.

SNLI The Stanford NLI dataset was produced using (Bowman et al., 2015). The corpus contains content from the Flickr 30k Corpus (Young et al., 2014), VisualGenome corpus (Krishna et al., 2017) and Gururangan et al. (2018). The corpus is in English. The dataset is collected through human-written English sentence pairs.

MNLI The Multi-genre NLI Corpus (Williams et al., 2018) is modeled on the SNLI corpus (Bowman et al., 2015) but it differs in the range of genres of spoken and written English text supporting cross-genre evaluation.

XNLI The Cross-Lingual NLI Corpus (Conneau et al., 2018) consists of manually-translated dev and test samples from MNLI in 14 languages: French, Spanish, German, Greek, Belgian, Russian, Turkish, Arabic, Vietnamese, Thai, Chinese, Hindi, Swahili, and Urdu. It is interesting for analysis because on a high level the semantics of the data follow MNLI. The corpus is made to evaluate the inference in any language when only English data is presented at training time.

FEVER NLI-style FEVER (Nie et al., 2019) is an NLI reformulation of the FEVER claim verification dataset (Thorne et al., 2018). The original dataset was collected by eliciting annotators to write fact sentences that are *supported*, *refuted*, or *unverifiable* relative source passages drawn from Wikipedia. This is converted into an NLI task by treating the elicited sentences as premises and the source passages as NLI pairs with relations *entail*, *contradict*, or *neutral* respectively. This dataset is unique in that the premises were elicited from seed hypotheses, meaning it has a balanced hyp. distribution but potentially biased prem. distribution.

ANLI The adversarial NLI corpus (Nie et al., 2020) is collected through crowdworkers and the purpose of this dataset creation is to make the state-of-art results fail in this dataset. The sentences are selected from the Wekipedia and manually curated HotpotQA training set (Yang et al., 2018). The language is in English. It contains three partitions of increasing complexity and size, which we refer to hereafter as A1, A2, and A3. Detailed data statistics are in Table 1.

OCNLI The Original Chinese NLI corpus was collected following MNLI-procedures but with strategies intended to produce challenging inference pairs (Hu et al., 2020). No translation was employed in producing this data; the source premise sentences and elicited hypotheses are original.

CAugNLI Kaushik et al. (2020) produced counterfactually augmented datasets for NLI and sentiment analysis using human annotators, instructing them to make minimal changes to the sentences beyond those necessary to change the label. It extends the work of Maas et al. (2011) and Bowman et al. (2015). They find that a BiLSTM classifier achieves negligible performance over chance when trained on hypothesis only. However, since their dataset includes elicited modified sentences in both the premise and hypothesis populations, there are opportunities for bias on both.

CAugNLI was produced by having human annotators minimally modify either the premise or hypothesis of 2,500 samples drawn randomly from SNLI so as to produce new samples with similar structure and word distributions but different meanings. These modifications are intended to reduce spurious correlations, in particular by roughly equalizing the distribution of relation labels with respect to word-level and semantic-level patterns in the elicited hypothesis sentences.

SNLI$_{\textbf{debiased}}$ and MNLI$_{\textbf{db}}$ are augmentations of the SNLI and MNLI train sets produced by training GPT-2 (Radford et al., 2019) generators on them, and then generating samples which they check for accuracy using a pretrained RoBERTa NLI classifier, and then reject if they exhibit spurious correlations including samplewise hypothesis-only model classifiability (Wu et al., 2022). To do this they first train static hypothesis-only SNLI and MNLI models, and reject all generated samples that can be successfully classified hypothesis-only by them. However, beyond this test under a static hypothesis-only distribution they do not attempt to assess if their generator models introduce *new leakage features* in the sentence distributions as a result of their accuracy filtering process. To test this we create test splits on the data (as they provide train sets only) which contain no sentence overlap with the train sets through random sampling.

B Training on PSC, Testing on SSC

Here we justify why PECO is computed on single-sentence condition (SSC) examples, using *models trained on the paired-sentence condition* (PSC).

Our core goal is to characterize only the **model-relevant** shortcut features that are present in the SSC data, to enable better model-level understanding and to enable shortcut feature elimination in future datasets. While all SSC accuracy must be driven by SSC-visible shortcuts, it is possible that **some SSC-visible cheating features aren't actually used as shortcuts by PSC classifiers**. Thus, we have to train on PSC and test on SSC, and PECO is an alternative metric of bias that captures this model-level separability of sentences in the SSC notion better than other approaches.

C PECO Parameter Details

The PECO scoring pipeline contains a number of parameters that require motivation, including SSC and PSC model training hyperparameters, number of principal components to reduce to during PCA $|PC|$, and number of k-means clusters k to divide the test set into for analysis (Figure 4). We specify our NN hyperparams that we performed grid search over in subsubsection 2.4.1. However, selecting k and $|PC|$ is not a straightforward simple grid search.

We report PECO scores for all assessed datasets in Table 5, for $k \in 10, 25, 50, 100$ and no PCA projection, $|PC| = 50$, and $|PC| = 100$ PCA conditions. We find that for a given k, the different PCA conditions have limited impact on the final scores. We also find that L2- and KLD-based PECO scores are well-correlated. For smaller test sets (e.g., ANLI and its partitions A1-A3, OCNLI) there is increased sensitivity to variations in k relative to the larger datasets such as SNLI. We selected $k = 30$, $|PC| = 50$ for our main experimental PECO results as it didn't produce the extreme swings in score for small test sets that we observed for higher k.

		k=10						*k=25*					
		No PCA		$\lvert PC\rvert = 50$		$\lvert PC\rvert = 100$		No PCA		$\lvert PC\rvert = 50$		$\lvert PC\rvert = 100$	
Dataset	Cond.	L2	KLD	L2	KLD	L2	KLD	L2	KLD	L2	KLD	L2	KLD
SICK	s_2	6.0	0.085	6.0	0.075	6.0	0.080	6.8	0.084	6.0	0.080	6.0	0.078
SNLI	s_2	8.5	0.155	8.5	0.150	8.5	0.150	8.4	0.138	8.8	0.140	8.4	0.136
MNLI-b	s_2	5.0	0.055	5.0	0.055	5.0	0.055	5.2	0.064	5.4	0.064	5.2	0.060
MNLI-u	|	5.5	0.065	5.5	0.065	5.5	0.065	5.8	0.072	5.6	0.072	5.8	0.070
XNLI	s_2	5.0	0.065	5.0	0.065	5.0	0.065	5.2	0.068	5.2	0.068	5.2	0.068
FEVER	s_1	11.0	0.160	11.0	0.160	11.0	0.160	13.2	0.154	10.6	0.150	10.2	0.144
ANLI	s_2	15.0	0.095	14.5	0.070	14.5	0.070	31.0	0.176	8.8	0.058	8.8	0.058
↪A1	|	5.5	0.085	5.5	0.075	5.5	0.090	17.2	0.124	28.4	0.136	21.8	0.118
↪A2	|	5.5	0.065	5.5	0.070	5.5	0.070	9.4	0.086	9.2	0.080	9.4	0.084
↪A3	|	5.5	0.070	5.5	0.070	5.5	0.065	5.8	0.082	5.6	0.084	5.8	0.080
OCNLI	s_2	10.5	0.165	10.5	0.170	10.5	0.170	14.0	0.178	14.2	0.170	14.0	0.172
CAugNLI	s_1	5.0	0.050	5.0	0.055	5.0	0.050	5.0	0.058	5.2	0.060	8.6	0.080
|	s_2	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050
$\text{SNLI}_{\text{debiased}}$	s_1	6.5	0.100	6.5	0.100	6.5	0.100	6.2	0.090	6.4	0.094	6.6	0.096
|	s_2	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050
$\text{MNLI}_{\text{debiased}}$	s_1	5.5	0.065	5.5	0.065	5.0	0.065	5.8	0.072	5.8	0.078	5.8	0.074
|	s_2	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050	5.0	0.050

		k=50						*k=100*					
		No PCA		$\lvert PC\rvert = 50$		$\lvert PC\rvert = 100$		No PCA		$\lvert PC\rvert = 50$		$\lvert PC\rvert = 100$	
Dataset	Cond.	L2	KLD	L2	KLD	L2	KLD	L2	KLD	L2	KLD	L2	KLD
SICK	s_2	6.2	0.081	7.4	0.082	7.9	0.081	13.5	0.104	13.2	0.098	12.4	0.102
SNLI	s_2	11.3	0.141	9.8	0.137	9.8	0.141	12.5	0.148	10.0	0.143	10.4	0.136
MNLI-b	s_2	5.4	0.066	5.4	0.066	5.3	0.064	5.4	0.069	5.4	0.068	6.3	0.069
MNLI-u	|	7.4	0.073	7.4	0.076	5.9	0.078	8.5	0.087	9.3	0.087	8.6	0.086
XNLI	s_2	5.5	0.074	5.6	0.074	5.6	0.073	5.7	0.079	5.7	0.080	5.8	0.079
FEVER	s_1	14.6	0.154	14.7	0.152	12.7	0.149	16.7	0.159	17.2	0.157	17.8	0.160
ANLI	s_2	25.2	0.132	8.9	0.063	7.0	0.058	26.9	0.162	9.2	0.071	9.0	0.072
↪A1	|	37.8	0.180	31.0	0.168	36.1	0.179	68.2	0.354	56.8	0.301	67.9	0.341
↪A2	|	17.3	0.114	19.6	0.112	15.1	0.103	37.1	0.171	31.8	0.188	34.9	0.179
↪A3	|	13.4	0.108	13.8	0.106	9.8	0.094	26.8	0.147	30.5	0.162	27.8	0.152
OCNLI	s_2	18.8	0.190	14.9	0.180	14.5	0.176	25.2	0.193	26.5	0.207	21.4	0.186
CAugNLI	s_1	5.0	0.051	5.0	0.050	5.0	0.051	5.9	0.059	5.9	0.056	5.1	0.056
|	s_2	13.8	0.073	6.9	0.068	12.1	0.101	19.7	0.126	19.0	0.135	13.2	0.091
$\text{SNLI}_{\text{debiased}}$	s_1	5.0	0.050	5.0	0.051	5.0	0.050	5.0	0.051	5.0	0.051	5.0	0.051
|	s_2	5.9	0.080	6.3	0.088	6.1	0.085	6.0	0.085	6.9	0.090	6.2	0.086
$\text{MNLI}_{\text{debiased}}$	s_1	5.0	0.052	5.0	0.052	5.0	0.052	5.1	0.054	5.1	0.054	5.1	0.056
|	s_2	5.6	0.070	5.6	0.068	5.5	0.069	6.3	0.071	5.3	0.065	5.6	0.070

Table 5: PECO scores for various levels of principal component reduction ($|PC|$), various numbers of k-means clusters, using L2 and KLD distance. These numbers are collected from a separate reproduction to Table 3, which uses different k for different datasets (reflecting their different sizes), and $|PC| = 30$. This table clearly illustrates that e.g., L2 PECO are pretty consistent across different numbers of principal components, for a given number of clusters k. We find that calibrating k based on the number of test-set samples for a given dataset is valuable for producing good characterization of the degree of bias. To improve legibility, we kept PECO_{KLD} in the $[0, 1]$ range.

Weakly-Supervised Questions for Zero-Shot Relation Extraction

Saeed Najafi and **Alona Fyshe**
Department of Computing Science, University of Alberta, Canada
`{snajafi,alona}@ualberta.ca`

Abstract

Zero-Shot Relation Extraction (ZRE) is the task of Relation Extraction where the training and test sets have no shared relation types. This very challenging domain is a good test of a model's ability to generalize. Previous approaches to ZRE reframed relation extraction as Question Answering (QA), allowing for the use of pre-trained QA models. However, this method required manually creating gold question templates for each new relation. Here, we do away with these gold templates and instead learn a model that can generate questions for unseen relations. Our technique can successfully translate relation descriptions into relevant questions, which are then leveraged to generate the correct tail entity. On tail entity extraction, we outperform the previous state-of-the-art by more than 16 F1 points without using gold question templates. On the RE-QA dataset where no previous baseline for relation extraction exists, our proposed algorithm comes within 0.7 F1 points of a system that uses gold question templates. Our model also outperforms the state-of-the-art ZRE baselines on the FewRel and WikiZSL datasets, showing that QA models no longer need template questions to match the performance of models specifically tailored to the ZRE task. Our implementation is available at `https://github.com/fyshelab/QA-ZRE`.

1 Introduction

Building models that capture abstract knowledge rather than just memorizing data is one of the inspirations for zero-shot benchmarks (Alex et al., 2021). In order to test a model's ability to extract higher-level knowledge, these benchmarks measure performance on data unlike what is seen during training. For example, in Zero-shot Relation Extraction (ZRE), a model trained to extract the name of a company's CEO should be able to extract the name of a country's political leader at test time.

In ZRE, the test relations do not appear in the training data, so one cannot apply typical relation classification approaches. One method for ZRE is to reframe the task as a Question-Answering (QA) problem by manually creating question templates for each relation type. Extracting the tail entity is accomplished by finding the answer span for the corresponding question template (Levy et al., 2017). However, this method requires gold question templates ahead of time. Here we ask, is it possible to leverage QA systems without annotating relations with gold question templates?

We treat ZRE as a tail entity generation task for which we consider the dual training of question and answer generators. We create question and answer generators by pre-training the publicly available T5 models (Raffel et al., 2020) on QA corpora, and then fine-tune for ZRE within our proposed learning framework. We investigate four training objectives based on Marginal Maximum Likelihood (MML) optimization, and suggest an off-policy sampling technique to avoid ungrammatical spurious questions in the search space.

Our experiments show that our off-policy sampling technique is critical for creating semantically relevant questions. For ZRE, we show that our weakly-supervised questions produce a model with competitive F1 score of 65.4 compared to the F1 score of 66.1 achieved by using gold question templates. Our contributions can be summarized as the following:

- We propose a new learning objective that combines off-policy sampling and MML optimization.
- We can successfully generate semantically relevant questions for a given relationship signal.
- We report a new state-of-the-art ZRE performance on the RE-QA dataset (Levy et al., 2017) without using gold question templates.
- We outperform SOTA baselines on the FewRel and WikiZSL datasets with a generative approach without using any gold question templates.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3075–3087
May 2-6, 2023

2 Problem Formulation

In this work, we train models to extract facts from unstructured text. Facts are represented as triplets (e_1, r, e_2) where e_1 is the head entity, e_2 is the tail entity, and r is the relation keywords. We explore **relation extraction (RE)**, which is the task of predicting r when e_1 and e_2 are known, and **tail entity extraction (TE)**, which is the task of predicting e_2 when e_1 and r are known.

We begin with TE, for which we learn $P(e_2|c, e_1, r)$: the distribution of the tail entity conditioned on an unstructured text context c, head entity, and the relation keywords. We investigate how to transfer models pre-trained on QA corpora to the task of TE by estimating $P(e2|c, e1, r)$. With a gold natural question q^*, which semantically includes the information about both of the head entity e_1 and the relation words r, we can optimize $P(e2|c, q^*)$ to fine-tune any answer generation model on RE datasets[1]. In cases where we do not have gold question templates, we can define a simple baseline using a pseudo question q^{pseudo}, which is the string concatenation of the head entity e_1 and the relation words r. We then can fine-tune a pre-trained QA model on RE datasets by directly optimizing the objective $P(e2|c, q^{pseudo})$.

To transfer the models pre-trained on QA corpora to the task of TE, we can provide a natural question for the given head entity and the relation keywords. However, providing question templates for every relation type is infeasible. We explore the idea of generating questions semantically relevant to the given input context, relation and the head entity. Therefore, we marginalize the joint distribution $P(e_2, q|c, e_1, r)$ with respect to the unobserved questions q to learn the tail entity generator: $P(e_2|c, e_1, r) = \sum_q P(e_2, q|c, e_1, r)$. As illustrated by Figure 1, we factorize $P(e_2, q|c, e_1, r)$ as the following equation where θ_Q and θ_A are respectively the parameters of the question and answer generation modules:

$$P(e_2|c, e_1, r) = \sum_q P(e_2, q|c, e_1, r) = \sum_q P_{\theta_Q}(q|c, e_1, r) \times P_{\theta_A}(e_2|c, q, e_1, r) \quad (1)$$

For example, given a context biography about the person "Donald Trump" and the relation keywords "place of birth," we would use P_{θ_Q} to first generate the question "Where was Donald Trump born?" and then the answer module P_{θ_A} can generate the response "New York."

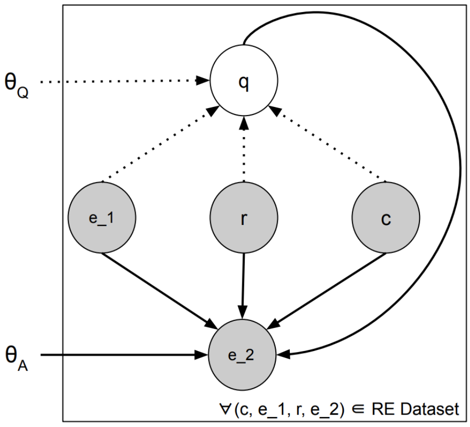


Figure 1: The probabilistic representation of the conditional distribution of the tail entity (e_2) conditioned on the context c, head entity e_1, and the relation keywords r. The question q is not observed. The θ_Q and θ_A are respectively the parameters of the question and answer generation modules.

We then re-purpose this TE model for the task of RE. Whether we train the tail entity generator $P(e_2|c, e_1, r)$ with gold, pseudo, or generated questions, we perform the RE task by scoring every possible relation of the test data and choosing the highest scoring relation:

$$\hat{r} = \arg\max_r P(e_2|c, e_1, r) \quad (2)$$

3 Methods

In this section we discuss the pre-training steps for the question and answer modules and then explore combinations of four training objectives based on maximizing the marginal likelihood of Equation 1. Finally, we introduce off-policy sampling and summarize the search algorithms used during training and test phases.

3.1 Pre-training the Answer Module

The answer generator is based on the publicly available T5 model (Raffel et al., 2020). We fine-tune this T5 model on the following five QA datasets to generate the gold answer in its output given the input passage and the question: 1) SQuAD (Rajpurkar et al., 2018) with unanswerable questions by generating a special unknown token in the output corresponding to these unanswerable questions, 2) NarrativeQA dataset containing abstractive responses for its questions (Kočiský et al., 2018),

---

[1] On an RE dataset, we can fine-tune models for TE and ZRE tasks.

3) RACE dataset as a multiple-choice QA dataset where we generate the question's correct choice as the answer text in the decoder (Lai et al., 2017), 4) BoolQ dataset having Yes/No questions (Clark et al., 2019), and 5) the DropQ dataset to encourage the model to learn discrete reasoning (e.g., addition, counting or sorting, etc.) in paragraphs (Dua et al., 2019). We fine-tune the T5 on the aforementioned QA corpora for four epochs with a batch size of 64. In total, we have fine-tuned this T5 model on 337,768 distinct QA pairs. We use the final model as our pre-trained answer generator which will be the initial checkpoint for learning the distributions $P(e_2|c, q^*)$, $P(e_2|c, q^{pseudo})$, or $P_{\theta_A}(e_2|c, q, e_1, r)$ in our experiments.

3.2 Pre-training the Question Module

We now fine-tune a T5 model using the QA corpora. We learn the distribution $P(q|c, e_1, r, e_2)$, and use the answer in each QA instance as a proxy for the tail entity e_2. To specify the synthetic head entity e_1, we run a named entity tagger[2] over questions in the QA corpus. For questions with multiple extracted entities, one of the entities are randomly selected as the head entity. To specify the relation r from the question, after ignoring punctuation or interrogative words (e.g., what, where, etc.), at most four tokens (excluding the extracted entities) are sampled. We fine-tune the T5 model on this synthetic dataset, giving gold passage as the context c, one of the extracted named entities as e_1, the gold answer tokens as e_2, and the sampled words as the relationship keyword r. We train the model to produce the gold question. We specifically fine-tune the model on the answerable questions from the SQuAD (Rajpurkar et al., 2018), NarrativeQA (Kočiský et al., 2018), and the DropQ datasets (Dua et al., 2019) for 4 epochs. This model will be the initial checkpoint for the question generator $P_{\theta_Q}(q|c, e_1, r)$ in our experiments.

3.3 Training Objectives

Having the pre-trained answer generator, we optimize $\log P(e_2|c, q^*)$, and $\log P(e_2|c, q^{pseudo})$ on RE datasets using the gold q^* and pseudo questions q^{pseudo}, respectively. To generate questions and learn from them, we also directly maximize the Marginal Log-Likelihood (MML) to train both the question and answer modules in Equation 1. With MML training, the objective is approximated with numerical summation using questions sampled from the question module. MML training outperforms the policy gradient methods on weakly-supervised semantic parsing (Guu et al., 2017) as simple policy gradient estimations may have large variances (Roberts and Tedrake, 2009; Guu et al., 2017). With the MML training, we use the following gradient vector to update θ_Q, where $\phi(q)$ approximates the true question posterior $P(q|c, e_1, r, e_2)$:

$$\begin{aligned} \nabla_{\theta_Q} M &= \sum_q \phi(q) \times \nabla_{\theta_Q} \log P_{\theta_Q}(q|c, e_1, r) \\ \phi(q) &= \frac{P_{\theta_Q}(q|c, e_1, r) \times P_{\theta_A}(e_2|c, q, e_1, r)}{Z} \\ Z &= \sum_{q'} P_{\theta_Q}(q'|c, e_1, r) \times P_{\theta_A}(e_2|c, q', e_1, r) \end{aligned} \tag{3}$$

Similarly, we use the following gradient update for θ_A: $\nabla_{\theta_A} M =$

$$\sum_q \phi(q) \quad \times \quad \nabla_{\theta_A} \log P_{\theta_A}(e_2|c, q, e_1, r) \tag{4}$$

As we ultimately want to generate the correct tail entity regardless of the input question in the answer module, we can use the best question from P_{θ_Q} and then optimize the probability of the tail entity for such a decoded question. This idea results in the following G gradient (G: greedy) update for the answer module:

$$\begin{aligned} \nabla_{\theta_A} G &= \nabla_{\theta_A} \log P_{\theta_A}(e_2|c, \hat{q}, e_1, r) \\ \hat{q} &= \arg\max_q P_{\theta_Q}(q|c, e_1, r) \end{aligned} \tag{5}$$

3.4 Off-Policy Search over Pre-trained Question Posterior

We encountered two issues while training the question generator with the previous MML objective. The first issue is that samples become ungrammatical templates as we continue training the question module. We hypothesize that the issue originates from taking direct samples from the question generator $P_{\theta_Q}(q|c, e_1, r)$ while simultaneously updating it during training. These spurious questions can hinder the model from generalizing to a new unseen relation. The program synthesis and semantic parsing research have reported a similar issue (Guu et al., 2017; Misra et al., 2018; Liang et al., 2018; Wang et al., 2019a, 2021).

---

[2] We used the NER tagger from the SpaCy library.

The second issue in taking samples from $P_{\theta_Q}(q|c, e_1, r)$ is that omitting the tail entity in the question generator while using the input context c sometimes results in multiple plausible questions. For example, consider the context "The Facebook brand was replaced by Meta in November 2021." For the ambiguous relationship word "replace", given the head entity "Facebook", we can generate two valid questions: a) "Which brand replaced Facebook?", b) "When was the brand Facebook replaced?".

To resolve the first issue, we apply an off-policy sampling technique (Levine et al., 2020; Pang and He, 2020) and use a fixed separate sampling module to sample questions during training. With this fixed search policy $S(q)$ over questions, we need to optimize the following new objective:

$$\log E_{q\sim S(q)}[\frac{P_{\theta_Q}(q|c, e_1, r)}{S(q)} \times P_{\theta_A}(e_2|c, q, e_1, r)]$$

To resolve the second issue, we feed the tail entity along with the context into the search module; however, we cannot feed the tail entity into the question generator $P_{\theta_Q}(q|c, e_1, r)$ as the correct tail entity is unknown during the test phase. Furthermore, to augment the information of the relationship keyword r, we append the relation description for each r and feed it to the question generator P_{θ_Q} and the sampling module $S(q)$.

We assume that our search module $S(q)$ is a fixed model approximating the question posterior $P(q|c, e_1, r, e_2)$, which can be our pre-trained question generator. Thus, we use two identical copies of our pre-trained question generator. One is fixed and will be used to provide samples for training. The other serves as the initial network for $P_{\theta_Q}(q|c, e_1, r)$, and we update its parameters during the training phase. The search module will also receive the tail entity e_2 as input, whereas the P_{θ_Q} only receives c, e_1, and r. During the test phase, we only generate questions from P_{θ_Q}. Using the off-policy samples, the new MML gradient updates will have the following forms, where $S(q)$ is our pre-trained question module and $\phi(q)$ approximates the true question posterior according to Equation 3:

Training Objective	Q grad.	A grad.
MML-MML	$\nabla_{\theta_Q} M$	$\nabla_{\theta_A} M$
MML-G	$\nabla_{\theta_Q} M$	$\nabla_{\theta_A} G$
OffMML-OffMML	$\nabla_{\theta_Q} O$	$\nabla_{\theta_A} O$
OffMML-G	$\nabla_{\theta_Q} O$	$\nabla_{\theta_A} G$

Table 1: Combinations of the four types of gradient vectors used for training the question (θ_Q) and answer (θ_A) modules. See Equations 3-7. Training objectives are named as (Question-Answer) tuples that refer to the gradient vectors used for the question and answer modules.

$$\nabla_{\theta_Q} O = E_{q\sim S(q)}[\frac{\phi(q)}{S(q)} \times \nabla_{\theta_Q} \log P_{\theta_Q}(q|c, e_1, r)] \tag{6}$$

$$\nabla_{\theta_A} O = E_{q\sim S(q)}[\frac{\phi(q)}{S(q)} \times \nabla_{\theta_A} \log P_{\theta_A}(e_2|c, q, e_1, r)] \tag{7}$$

Table 1 summarizes the four gradient vectors we use for the training phase. With $\nabla_{\theta_Q} G$ gradient update, we always use samples from the question generator (not from $S(q)$) to expose the answer module to errors from P_{θ_Q}.

3.5 Search Algorithms

To generate the tail entity on the test data, regardless of fine-tuning the pre-trained answer module with gold, pseudo, or generated questions, we use greedy decoding to find the top-scoring tokens. To generate questions from $P_{\theta_Q}(q|c, e_1, r)$, we use the top-scoring question found with beam search decoding having the beam size of 8.

During training, to estimate the G gradient vector listed in Table 1, we rely on beam search decoding with the beam size of 8 to find the top scoring question. For MML gradient vectors, we use top-p sampling with the suggested threshold of 0.95 to collect 8 final samples from the question generator $P_{\theta_Q}(q|c, e_1, r)$ or sampling module $S(q)$. In top-p sampling (Holtzman et al., 2019), at each decoding step, we find the smallest set of vocabulary with the cumulative probability above a threshold (e.g., 0.95) and re-scale the distribution among these tokens to perform the sampling. We find this approach has a better exploration compared to beam-search, greedy, or top-k decoding.

Table 2: The example input for the question and search modules. The search module also receives the tail entity in the input.

Module	Input Format
Question	"answer: Isaac Nicola <SEP> place of birth ; most specific known birth location of a person, animal or fictional character context: Isaac Nicola Romero (1916 in Havana, Cuba) was a prominent Cuban guitarist. </s>"
Search	"answer: Isaac Nicola <SEP> place of birth ; most specific known birth location of a person, animal or fictional character Havana context: Isaac Nicola Romero (1916 in Havana, Cuba) was a prominent Cuban guitarist. </s>"

Table 3: The example input for the answer module given gold, pseudo, and the generated questions. The generated question is from the question module trained with the OffMML-G objective.

Question Type	Input Format
Pseudo	"question: Isaac Nicola <SEP> place of birth context: Isaac Nicola Romero (1916 in Havana, Cuba) was a prominent Cuban guitarist. </s>"
Generated	"relation: Isaac Nicola place of birth question: What was the name of the place where Isaac Nicola was born? context: Isaac Nicola Romero (1916 in Havana, Cuba) was a prominent Cuban guitarist. </s>"
Gold	"question: What was Isaac Nicola's city of birth? context: Isaac Nicola Romero (1916 in Havana, Cuba) was a prominent Cuban guitarist. </s>"

4 Experiments Setup

4.1 Datasets

We use three datasets to test our weakly-supervised questions in the ZRE setting. We first use the RE-QA dataset released by Levy et al. (2017) for tail entity (TE) generation using QA models. RE-QA provides ten folds with 84 relation types in the train, 12 on the dev, and 24 on the test splits. There is no overlap among relation types of these splits per fold. Each fold contains 840k sentences in the train, 6k sentences in the dev, and 12k in the test split. Half of the sentences are negative examples with no corresponding e_2 given e_1 and r. The RE-QA dataset provides multiple gold question templates for each relation type.

To compare our system with current SOTA methods for the ZRE, we also consider the FewRel and WikiZSL datasets. The FewRel dataset has 56K sentences over 80 relation types (Han et al., 2018). The WikiZSL dataset has 93483 sentences over 113 relation types (Zhong et al., 2017). Similar to the previous SOTA method ZS-BERT (Chen and Li, 2021), we randomly split the FewRel and WikiZSL datasets into train/dev/test splits where we have 5 and 15 relation types in the dev and test splits, respectively; with no overlap of the types between these splits. We create five train/dev/test folds splitting the datasets with different random seeds.

For all the datasets, if possible, we retrieve the relation description from the wikidata[3] and append it to the relation keyword r in the question generator P_{θ_Q} or the sampling module $S(q)$.

Table 2 lists an example for the input given to our question and search modules. Table 3 also provides an example for the input of the answer module for the gold, pseudo or generated questions.

4.2 Evaluation Metrics

For the TE task, as we generate a single tail entity $e_2^{'}$ for each input sentence, the extracted triple $(e_1, r, e_2^{'})$ matches the ground truth triple (e_1, r, e_2) only if $e_2^{'} = e_2$ (case insensitive). For the negative examples, we generate a special 'NO_ANSWER' output specifying the null tail entity. We then use the official evaluation script[4] to compute the Precision, Recall, and F1-score for the TE task when there are negative examples in the test data. Precision is the true positive count divided by the number of times the system generates a non-null tail entity. Recall is the true positive count divided by the number of positive examples having the tail entity given e_1 and r. For the ZRE performed by the inference objective 2, we compute the macro

---

[3] https://www.wikidata.org/wiki/Wikidata:List_of_properties/all_in_one_table

[4] http://nlp.cs.washington.edu/zeroshot/

Precision, Recall, and F1-score; averaged across relation types only on positive examples.

4.3 Baselines

Few related works have focused on ZRE. The ZS-BERT model (Chen and Li, 2021) maps sentences and relations into a shared semantic space using two BERT encoders and then uses nearest neighbor search to predict the unseen relations. During training, the ZS-BERT minimizes the distance between the attribute vectors of the input sentence and the seen ground-truth relation while maximizing the distance from other incorrect relation representations. The ZS-BERT has outperformed another earlier work that treats the task of ZRE as an entailment task (Obamuyide and Vlachos, 2018).

We also provide the reported ZRE results on the FewRel and WikiZSL datasets from Tran et al. (2022) which introduces discriminative inter-relation and inter-sentence losses along with a comparative network to better separate sentence-relation representation pairs.

We finally consider RelationPrompt (Chia et al., 2022) which is trained to generate the relation triplets for a given sentence using a BART-base sequence-to-sequence model. We also apply its data-generation variant which uses GPT2 to augment the training dataset by generating synthetic (sentence, entity) pairs for the unseen test relations.

4.4 Training Details

For pre-training/fine-tuning T5 on QA and RE datasets, we use the Adafactor optimizer, which requires less memory compared to Adam (Shazeer and Stern, 2018). In addition, we follow the original configuration[5] suggested for fine-tuning the T5 transformers, such as disabling update clipping, no weight decay, and no warm-up starts (Raffel et al., 2020). In all of our experiments, we used the fixed learning rate of 0.0005.

For fine-tuning our models and all the baselines on all the RE datasets, we train them for one epoch using the batch size of 16 on the RE-QA, 4 on the FewRel, and 16 on the WikiZSL datasets, respectively. Every 100 training steps, we evaluate the models on the dev sets, and then we report the performance of the dev set's best model on the test split. For our models, on all the three datasets, we pre-train/fine-tune T5-small with 6 transformer blocks and 512 hidden states. Due to GPU constraints, we use eight samples or a beam size of 8 in the top-p sampling and beam search decoding algorithms, respectively.

5 Results

5.1 Training Objective Comparisons

In our first experiment, we compare the different training objectives listed in Table 1 in the tail entity generation task. To provide a fair comparison between these objectives, we also append the relation description to the relationship word r in the on-policy training objectives MML-MML and MML-G. We compare these four objectives on the ten folds of the RE-QA dataset. We also report the baseline Base-Base that predicts the test data using only the pre-trained question-answer modules without fine-tuning the modules over the RE dataset.

Table 5 summarizes the metrics on the tail entity generation task for these training objectives. OffMML-G, which uses off-policy samples from the pre-trained question module, outperforms the rest of the training methods with the average F1-score of 56.2. Our proposed off-policy sampling technique has improved the performance of the tail entity generator in the model trained with OffMML-G by an average gain of 1.5 F1-score compared to the model trained with the MML-G objective. Furthermore, on the positive examples, the accuracy for generating the correct tail entity is higher when we take the best question from the question module P_{θ_Q} since the OffMML-G outperforms OffMML-OffMML by an average gain of 2.1 Recall points. To assess the grammar in the generated questions, we compute the average perplexity over the test folds using the GPT2-large (Radford et al., 2019) language model. The last column (PP) in Table 5 shows that both the off-policy sampling and the G gradient objective contribute to a lower perplexity compared to the on-policy sampling and the MML gradient objective as the OffMML-G achieves the lowest perplexity of 144. This observation verifies our hypothesis that off-policy sampling helps to avoid ungrammatical spurious questions.

In Table 4, we also provide an example of the generated questions for the following input context: "The Vision of Saint Eustace is a painting by the early Italian Renaissance master Pisanello, now in the National Gallery in London." Given the head entity "The Vision of Saint Eustace," and the

---

[5] `https://discuss.huggingface.co/t/t5-finetuning-tips/684/3`

Table 4: Example generated questions for the relationship type “collection” (“art, museum or bibliographic collection the subject is part of”) given the head entity “The Vision of Saint Eustace.” The semantically correct questions are in blue. The Base-Base model predicts the test data using only the pre-trained question-answer modules. The model name “A-B” denotes questions generated by the question module trained with the objective A while the answer module is trained with the objective B.

Q-A Models	Generated Question
Base-Base	“What collection is part of The Vision of Saint Eustace?”
MML-MML	“What subjects subjects?”
MML-G	“What art collection is The Vision of Saint Eustace?”
OffMML-OffMML	“The Vision of Saint Eustace is part of what collection?”
OffMML-G	“The Vision of Saint Eustace is part of what collection?”
GoldQ	“What is the name of the place where The Vision of Saint Eustace can be found?”

Table 5: The average Precision, Recall and F1-score for the tail entity generation task using our four training objectives on the ten test folds of the RE-QA dataset. The last column (PP) reports the average perplexity of the generated questions over these ten test folds. Best performance bolded, second best underlined.

Q-A Models	P	R	F1	PP
Base-Base	24.4	31.4	27.5	170
MML-MML	57.3	52.3	54.6	10456
MML-G	57.0	52.9	54.7	1309
OffMML-OffMML	**60.7**	52.3	55.9	148
OffMML-G	58.3	**54.4**	**56.2**	**144**

relationship type “collection.” We verify that the proposed off-policy sampling technique is necessary to create semantically and grammatically correct questions as the OffMML-G objective could successfully generate the question “The Vision of Saint Eustace is part of what collection?”

5.2 Baseline Comparisons

We now compare systems trained with the OffMML-G training objective with the two primary baselines of using pseudo or gold questions to fine-tune the answer generator.

Table 6: The average Precision, Recall and F1-score on the 10 test folds of the RE-QA dataset for the tail entity generation task. Highest performance bolded, second highest underlined.

Q-A Models	P	R	F1
PseudoQ-Base	1.7	3.2	2.3
Base-Base	24.4	31.4	27.5
GoldQ-Base	36.1	55.1	43.6
Levy et al. (2017)	43.6	36.5	39.6
PseudoQ-Trained	57.9	50.2	53.6
OffMML-G	58.3	54.4	56.2
GoldQ-Trained	**60.1**	**57.9**	**58.9**

Table 7: The average Precision, Recall and F1-score on the ten test folds of the RE-QA dataset for the Zero-shot Relation Extraction (ZRE) task.

Models	P	R	F1
ZS-BERT	34.7	32.2	33.3
PseudoQ-Trained	63.3	60.5	61.8
OffMML-G	66.7	64.2	65.4
GoldQ-Trained	**67.3**	**65.0**	**66.1**

For the tail entity generation task, Table 6 presents the average F1 scores on the ten test folds of the RE-QA dataset, considering both the positive and negative examples. Using the test data’s gold templates, the pre-trained answer generator (GoldQ-Base) achieves an average F1-score of 43.6, and after fine-tuning the answer module, the baseline with gold questions achieves the highest average F1-score of 58.9, outperforming the previous BiDAF method (Levy et al., 2017), which had an average F1-score of 39.6. With the OffMML-G training objective, we boost the performance of the pre-trained answer module from the average F1-score of 27.5 to 56.2. The OffMML-G training objective outperforms the baseline with pseudo questions by an average of 2.6 F1-score.

For the ZRE task, as listed in Table 7, the model trained with OffMML-G objective outperforms the PseudoQ baseline by an average F1 score of 3.6. This superior performance verifies that using semantically relevant questions improves the performance of QA models while fine-tuning them on the RE datasets. Moreover, with our generated questions, OffMML-G achieves the competitive

Table 8: The average Precision, Recall and F1-score for ZRE on five random runs over the FewRel and WikiZSL datasets. There are 15 unseen relations in the test folds. Highest performance bolded, second highest underlined. The † is the reported performance of ZS-BERT (Chen and Li, 2021). The ⋆ is the reported performance of RelationPrompt without using synthetic examples (Chia et al., 2022).

	FewRel			WikiZSL			
Model	P	R	F1	P	R	F1	Avg F1
ZS-BERT†	35.5	38.2	36.8	34.1	34.4	34.3	35.6
ZS-BERT (our run)	41.9	39.5	40.5	28.8	26.5	27.4	34.0
Tran et al. (2022)	44.0	39.1	41.4	38.4	36.0	37.2	39.3
RelationPrompt⋆	**66.5**	40.0	49.4	54.5	29.4	37.5	43.5
RelationPrompt (our run)	53.2	45.1	48.3	52.4	38.8	44.5	46.4
RelationPrompt (our run) (+supp)	59.7	**61.0**	60.3	59.7	**60.5**	60.0	60.2
PseudoQ-Trained	30.8	32.4	31.6	33.1	34.6	33.8	32.7
OffMML-G	29.4	30.4	29.9	28.2	28.7	28.4	29.2
PseudoQ-Trained (+negs)	62.8	58.4	60.5	62.6	59.5	61.0	60.8
OffMML-G (+negs)	63.7	59.2	**61.3**	**63.6**	60.4	**61.9**	**61.6**

F1 score of 65.4 compared to the F1 score of 66.1 achieved by the system using gold questions.

In our final experiments, we fine-tune the question-answer modules with the OffMML-G objective on the FewRel and WikiZSL datasets which have 15 unseen relations in the test folds. As the FewRel and WikiZSL datasets do not provide any negative examples, we create synthetic negative examples by repeating every train instance once with its ground-truth relation label replaced with another relation class from the train data. This data augmentation approach is simple enough to include with our models as we generate the null tail entity for negative examples. The recent discriminate approaches such as ZS-BERT and the method of Tran et al. (2022) use margin-based or contrastive learning to separate the ground-truth relation class from other negative relation labels. As presented in Table 8, our transferred QA models trained with OffMML-G objective over the mix of positive and negative examples significantly outperforms the ZS-BERT and the recent baseline of Tran et al. (2022) with an average gain of 26.8 F1 points on both the FewRel and WikiZSL datasets. We do not change the architecture of the underlying transformers as apposed to the comparative network of Tran et al. (2022) trained over the BERT encoders.

We finally compare against RelationPrompt (Chia et al., 2022) which directly generates the relation class in the decoder of the BART-base model given the input sentence and the prompt information about the head and tail entities. The OffMML-G (+negs) achieves an average gain of 15.2 F1 points on both the FewRel and WikiZSL datasets (Table 8)[6]. The idea of using synthetic (sentence, entities) pairs generated by a large language model for the test relation classes improves the performance of RelationPrompt (see +supp row in Table 8). Such data augmentation to use synthetic examples can also be helpful for our method to further guide the question generation for the test relation classes.

In Table 9, we provide some example questions generated for the WikiZSL test data using our question module trained with OffMML-G (+negs) objective.

6 Related Works

Weakly-Supervised Semantic Parsing: The objective in Equation 1 follows a similar search-and-learn framework used in the recent works of weakly-supervised program synthesis and semantic parsing (Guu et al., 2017; Misra et al., 2018; Liang et al., 2018; Wang et al., 2019a, 2021). For example, in the task of text-to-SQL generation, a set of possible SQL solutions for a given natural query are generated (searched) and then those solutions are validated with an SQL verifier. The program verifiers in these recent works provide a deterministic score/reward to update the search module. However, in our objective, the feedback for the question module is another stochastic answer generator that needs to be trained along with the search module.

Reading Comprehension for RE: The joint ex-

---

[6] As expected from the model input format, our negative examples do not improve RelationPrompt.

traction of entities and relations has also been reduced as a multi-hop QA task using pre-defined question templates for a few entity and relation types present in the ACE and the CoNLL04 datasets (Li et al., 2019). Another recent work shows that asking diverse question templates could further boost the performance of the QA models for the joint entity-relation extraction (Zhao et al., 2020). Furthermore, QA frameworks have been applied to re-score and verify the extracted relations of a separate relation extractor by forming simple pseudo-questions concatenating the head entity with the relationship words (Cheng et al., 2021).

Several works have done event argument extraction through QA (Liu et al., 2020; Du and Cardie, 2020). Du and Cardie uses predefined question templates for the events. The work of Liu et al. first picks a template query word for the topic of the event and includes it as the prefix for the question, and then translates a descriptive span around the event word into a descriptive question suffix by learning an unsupervised translation model. Although Liu et al. applies their approach only to the task of event extraction on short sentences from the ACE dataset, they manually provide a relevant question prefix for the relation type.

Question Generation: The Question Generation (QG) research aims at generating natural questions given a document such that the answer modules can find the answers to these questions. Recent systems build end-to-end neural sequence-to-sequence models for QG (Du et al., 2017). Moreover, the current research first extracts the key phrases which could be the answer to the question and then feeds it as an extra input signal while training the QG models (Wang et al., 2019b). Multiple QA corpora have been combined to train a single question generator conditioned on multiple answer types (Murakhovs'ka et al., 2021).

Earlier work suggests dual training of the QG and answer-sentence selection tasks, however, they could marginally improve both of the QG and QA tasks (Tang et al., 2017). Closely related to our work, another recent study translates freebase triplets into natural questions to augment the QA corpora (Serban et al., 2016). However, this triplet-to-question translation task does not use any context passages. The rule-based entity extraction and QG have also been explored for open information extraction on financial datasets (Gupta et al., 2021). However, its rules are not comprehensive for generating questions for complex relations.

Apart from training the QG models on supervised data, recent work uses policy gradient reinforcement learning to a) Optimize rewards related to the fluency and answer-ability of the generated questions (Yuan et al., 2017), b) Expose the model to its errors (Song et al., 2017), and c) Include paraphrase probability to properly compare the generated questions with the ground truth ones (Zhang and Bansal, 2019).

Despite the previous studies on QG, we treat questions as latent variables without having access to gold questions given the input context and the relation triplets.

7 Conclusion

This work introduces the OffMML-G training objective to fine-tune the question and answer generators for RE. Our method generates semantically relevant questions for the answer module given the head entity and the relationship keywords. We demonstrated that with these weakly-supervised questions, one could fine-tune QA models on the RE corpora, achieving competitive results in detecting unseen relations. Our future direction would deploy the technique on document-level entity-relation extraction, further exploiting the inference capabilities of QA models.

Limitations

A major limitation in our method is that we need to provide three distributions during training: a) $P(q|c, e_1, r)$ as the question generator, b) $P(e_2|c, e_1, r, q)$ as the answer module, and c) $S(q)$ as our fixed search module over questions. This generative approach (i.e. generating the tail entity) for relation extraction requires more compute resources compared to a direct discriminate approach to learn $P(r|c, e_1, e_2)$, however, such a direct approach cannot be used to transfer QA models into the RE task.

We have used the T5-small models in all our experiments. Further gain can be achieved by switching into T5-large models, however, we leave those large-scale experiments for future work.

Ethics Statement

Many language models show biases in their output due to the data used to train them (Liang et al., 2021). It is possible that these biases could affect the RE results that we present here (e.g., producing poor performance for certain kinds of relations, or for entities with names different from the training data).

References

Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C. Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, Michael Noetel, and Andreas Stuhlmüller. 2021. RAFT: A real-world few-shot text classification benchmark. *CoRR*, abs/2109.14076.

Chih-Yao Chen and Cheng-Te Li. 2021. ZS-BERT: Towards zero-shot relation extraction with attribute representation learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3470–3479, Online. Association for Computational Linguistics.

Jiayang Cheng, Haiyun Jiang, Deqing Yang, and Yanghua Xiao. 2021. A question-answering based framework for relation extraction validation. *CoRR*, abs/2104.02934.

Yew Ken Chia, Lidong Bing, Soujanya Poria, and Luo Si. 2022. RelationPrompt: Leveraging prompts to generate synthetic data for zero-shot relation triplet extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 45–57, Dublin, Ireland. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Himanshu Gupta, Amogh Badugu, Tamanna Agrawal, and Himanshu Sharad Bhatt. 2021. Zero-shot open information extraction using question generation and reading comprehension.

Kelvin Guu, Panupong Pasupat, Evan Liu, and Percy Liang. 2017. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1051–1062, Vancouver, Canada. Association for Computational Linguistics.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *CoRR*, abs/2005.01643.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis and semantic parsing. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards Understanding and Mitigating Social Biases in Language Models. In *International Conference on Machine Learning*, pages 6565–6576. PMLR.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online. Association for Computational Linguistics.

Dipendra Misra, Ming-Wei Chang, Xiaodong He, and Wen-tau Yih. 2018. Policy shaping and generalized update equations for semantic parsing from denotations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2442–2452, Brussels, Belgium. Association for Computational Linguistics.

Lidiya Murakhovs'ka, Chien-Sheng Wu, Tong Niu, Wenhao Liu, and Caiming Xiong. 2021. Mixqg: Neural question generation with mixed answer types.

Abiola Obamuyide and Andreas Vlachos. 2018. Zero-shot relation classification as textual entailment. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 72–78, Brussels, Belgium. Association for Computational Linguistics.

Richard Yuanzhe Pang and He He. 2020. Text generation by learning from off-policy demonstrations. *CoRR*, abs/2009.07839.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

John Roberts and Russ Tedrake. 2009. Signal-to-noise ratio analysis of policy gradient algorithms. In *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc.

Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering. *CoRR*, abs/1709.01058.

Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *CoRR*, abs/1706.02027.

Van-Hien Tran, Hiroki Ouchi, Taro Watanabe, and Yuji Matsumoto. 2022. Improving discriminative learning for zero-shot relation extraction. In *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*, pages 1–6, Dublin, Ireland and Online. Association for Computational Linguistics.

Bailin Wang, Mirella Lapata, and Ivan Titov. 2021. Learning from executions for semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2747–2759, Online. Association for Computational Linguistics.

Bailin Wang, Ivan Titov, and Mirella Lapata. 2019a. Learning semantic parsers from denotations with latent structured alignments and abstract programs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3774–3785, Hong Kong, China. Association for Computational Linguistics.

Siyuan Wang, Zhongyu Wei, Zhihao Fan, Yang Liu, and Xuanjing Huang. 2019b. A multi-agent communication framework for question-worthy phrase extraction and question generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7168–7175.

Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Saizheng Zhang, Sandeep Subramanian, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 15–25, Vancouver, Canada. Association for Computational Linguistics.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2495–2509, Hong Kong, China. Association for Computational Linguistics.

Tianyang Zhao, Zhao Yan, Yunbo Cao, and Zhoujun Li. 2020. Asking effective and diverse questions: A machine reading comprehension based framework for joint entity-relation extraction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3948–3954. International Joint Conferences on Artificial Intelligence Organization. Main track.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Appendix

Table 9: The generated questions with OffMML-G (+negs) on a random example of the WikiZSL test split for the context: “Jerome Ellsworth Couplin III (born August 31 , 1991) is an American football safety for the Philadelphia Eagles of the National Football League (NFL) .” The ground-truth relation triple is (Philadelphia Eagles, member of, National Football League)

Relation Class	Generated Question
1- contains the administrative territorial entity (P150)	“What is the name of the administrative entity that contains the Philadelphia Eagles?”
2- student (P802)	“What is the name of the notable student(s) of Philadelphia Eagles?”
3- located next to body of water (P206)	“What body of water is located next to the Philadelphia Eagles?”
4- location (P276)	“What is the location of the object, structure or event for the Philadelphia Eagles?”
5- drafted by (P647)	“Which team was drafted by the Philadelphia Eagles?”
6- member of (P463)	“What organization is a member of the Philadelphia Eagles?”
7- parent astronomical body (P397)	“What is the name of the major astronomical body the Philadelphia Eagles belong to?”
8- author (P50)	“Who is the main author of a written work for the Philadelphia Eagles?”
9- sport (P641)	“What sport does Brandon Lee Graham play for the Philadelphia Eagles?”
10- field of this occupation (P425)	“What field of occupation does Brandon Lee Graham represent for the Philadelphia Eagles?”
11- founder (P112)	“Who is the founder or co-founder of the Philadelphia Eagles?”
12- work location (P937)	“What is the name of the location where people or organizations were involved in work for the Philadelphia Eagles?”
13- from fictional universe (P1080)	“What is the subject’s fictional entity in the Philadelphia Eagles?”
14- award received (P166)	“What award does Brandon Lee Graham receive from the Philadelphia Eagles?”
15- located on astronomical body (P376)	“What is the name of the astronomical body located on the Philadelphia Eagles?”

DiffQG: Generating Questions to Summarize Factual Changes

Jeremy R. Cole[1*] **Palak Jain**[1*] **Julian Martin Eisenschlos**[1]
Michael J.Q. Zhang[3] **Eunsol Choi**[3] **Bhuwan Dhingra**[1,2]
[1] Google Research [2] Duke University [3] The University of Texas at Austin
{jrcole,palakj,eisenjulian,bdhingra}@google.com
{mjqzhang,eunsol}@utexas.edu

Abstract

Identifying the difference between two versions of the same article is useful to update knowledge bases and to understand how articles evolve. Paired texts occur naturally in diverse situations: reporters write similar news stories and maintainers of authoritative websites must keep their information up to date. We propose representing factual changes between paired documents as question-answer pairs, where the answer to the same question differs between two versions. We find that question-answer pairs can flexibly and concisely capture the updated contents. Provided with paired documents, annotators identify questions that are answered by one passage but answered differently or cannot be answered by the other. We release DiffQG which consists of 759 QA pairs and 1153 examples of paired passages with no factual change. These questions are intended to be both unambiguous and information-seeking and involve complex edits, pushing beyond the capabilities of current question generation and factual change detection systems. Our dataset summarizes the changes between two versions of the document as questions and answers, studying automatic update summarization in a novel way.

1 Introduction

Given a pair of statements, how can we identify the difference in their information content? This problem has existed in different forms across NLP research, such as *recognizing textual entailment* (Dagan et al., 2010) and *natural language inference* (Bowman et al., 2015). The initial focus of this type of research was finding the logical implication relations between sentences.

More recently, specialized entailment-like resources and models have been applied to fact verification (Thorne et al., 2018b) with applications to science, education and journalism. This trend has exposed the limited transfer between logical entailment and general factual change detection (Thorne et al., 2018a) as well as the need for interpretable models for this task (Kumar and Talukdar, 2020).

With ~~16~~ 17 NBA Championships, the LA Lakers along with ~~place the second behind~~ the Boston Celtics have won the most in NBA history.

Q: How many NBA championships have the LA Lakers won?
A: ~~16~~ 17

Q: Which team has won the most NBA championships?
A: ~~Boston Celtics~~ LA Lakers along with Boston Celtics

Figure 1: DiffQG consists of paired Wikipedia passages that correspond to factual edits. The goal is to generate a discriminating question given an answer span such that the question is answerable by one of the passages but not the other or yields different answers.

Wikipedia revisions across time provide a large scale and highly available source of sentence pairs, leading to new resources such as WikiAtomicEdits (Faruqui et al., 2018) and VitaminC (Schuster et al., 2021). However, prior work is limited to minimal changes that concern only a single factual addition or change. We introduce DiffQG, a manually annotated dataset spanning changes over multiple years. DiffQG consists of paired passages with complex factual changes including multiple additions and deletions within the same example. Additionally, it provides a way to interpret the prediction in the form of a discriminative question-answer pair that identifies the change.

Question-answer pairs provide a semi-structured summary of a change: more flexible than knowledge graph triples and more useful than free-form text. For instance, question-answer pairs can represent different types of updates: a new prime minister may update an answer, while a new type of

---

[*] Equal Contribution.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3088–3101
May 2-6, 2023

minister would add an entirely new question.

Question generation (QG) is a new NLP task that consists of generating a question that a provided document answers. There are various successful applications of this approach, including augmenting datasets to train question answering systems (Duan et al., 2017; Lewis et al., 2021), capturing implicit information written about text (Pyatkin et al., 2021), and building soft knowledge bases (Chen et al., 2022). Previous work in QG treated the underlying passages as static (Lewis et al., 2021), while real life documents are constantly updated (Dhingra et al., 2022). As the source corpus is updated, new question-answer pairs must be added and existing ones must be updated.

DIFFQG thus addresses two challenges simultaneously: providing an interpretable summarization of factual changes and updating soft knowledge bases consisting of question answer pairs. We hope that this dataset can also help evaluate the quality of QG models in producing natural, semantically correct, unambiguous, and information-seeking questions. The dataset and code for our experiments will be open sourced.[1]

Our contributions are the following:

(a) We introduce DIFFQG, an expert-annotated *evaluation* dataset that consists of questions that summarize the difference between two passages. To the best of our knowledge, no prior dataset exists that covers such long and complicated edits.

(b) We propose a set of metrics that can be used to measure improvements in question generation or factual change detection.

(c) We evaluate a comprehensive set of baselines that surface the shortcomings of current systems.

2 DIFFQG Task

The goal of DIFFQG is to capture how two similar passages differ from each other using question-answer pairs. In particular, given a base passage x_b and a target passage x_t, where x_t and x_b are different versions of the same article, we aim to generate discriminating questions Q_t. For each $q_t \in Q_t$, the information to deduce the corresponding answer span $a_t \in A_t$ must be missing in x_b. To limit the scope of possible questions, each answer span $a_t \in A_t$ must be a substring of x_t. While a_t could also be a substring of x_b, x_b must be missing the required information to deduce a_t is the correct answer. Alternatively, there could be a corresponding answer span a_b, which is the answer resulting from answering q_t with x_b. Note that we consider paraphases of a_t, such as lexicalizing numbers and using alternate entity names, as equivalent answers.

This discriminating question has certain additional requirements: it should be seeking factual information and stand-alone (Choi et al., 2021) (i.e., interpretable when presented by itself without the passage). It is possible that no such discriminating question can be written. The annotators only mark that there is no factual change when they are fairly confident that there is no new information about the answer span in the target passage.

Consider the following example:

- x_b = John Doe won two gold medals at the Olympics in 2012.
- x_t = John Doe won a gold medal at the Olympics in 2012.

Annotators are informed that the goal of the process is to collect *disambiguated* and *information-seeking* queries that can be answered with one passage but not with the other. By *disambiguated* queries, we mean queries that refer to roughly a single answer without any context. For instance, “Who won two gold medals in the 2012 Olympics?” could refer to several different people, and questions of the form “How many medals did he win in the 2012 Olympics?” are not answerable at all without the presence of the John Doe passage.

Information-seeking queries are ones where the questioner would not need to know the answer in advance for the question to make sense. This is related to the original goals of Natural Questions (Kwiatkowski et al., 2019) and corresponds to the *Cranfield*-style questions described by Rodriguez and Boyd-Graber (2021). As an example, “What did Al Capone’s mother do for a living?” seems like an information-seeking query. On the other hand, “Which Italian-American gangster’s mother was a seamstress?” does not: why would the questioner assume that such a person even exists unless they already knew the answer? We describe the annotation process to acquire such discriminating question set in the next section.

3 Data Collection

Collecting such discriminating questions is a nontrivial process. Thus, we introduce a staged annotation process with expert annotators (the authors

---

[1] `https://github.com/google-research/language/tree/master/language/diffqg`

of this paper) and use a question generation model to aid annotation. We describe our process below (visualized in Figure 2).

3.1 Input Passage Pair Selection

First, we extract the Wikipedia pages for entities from the Natural Questions (NQ) training set (Kwiatkowski et al., 2019). In particular, we find the pages for Wikipedia snapshots between the years 2008 and 2020. After sampling a base document, we find the version of that document one year later and use this as the target document. Using the two documents as a corpus, we compute cosine similarity between the TF-IDF vectors over each sentence pair and pick the pair with the highest similarity. Sentence pairs with similarity either greater than 0.8 or less than 0.25 are discarded. In order to retain meaningful changes, we ensure at least one noun or number is edited and up-sample instances where either a named entity or at least five tokens have been edited.

This process thus focuses on edits accumulated over a year and consists of changes ranging from five to twenty tokens, making these semantically richer and more widely applicable than existing factual change detection datasets.

3.2 Seed QA Pair Generation

Each target passage has a very large number of possible answer spans; for convenience, we restrict them to only noun phrases identified using the Berkeley Neural Parser (Kitaev and Klein, 2018). To increase annotation speed, each example starts with a *seed question* that is generated by a question generation model from the target passage and answer span. In particular, we use a T5-XXL model (Raffel et al., 2020) that has been finetuned on the SQuAD dataset (Rajpurkar et al., 2016).

3.3 Annotation Process

DiffQG annotation was done in three phases by six expert volunteers. First, annotators are given the paired passages described above along with the answer span and seed question, which corresponds to one *example candidate*. Then, they label each example candidate with one of the five options:

Accept The seed question follows all requirements for discriminating questions as is.

Context The seed question asks about the appropriate topic but is not answerable outside of the context of the passage. For instance, questions like "What did he win?" or "Where were the Olympics held?" both lack context in order to answer the question successfully.

Edit The example candidate answer has a discriminating question, but the question is different than the seed question. Sometimes, this is because the seed question does not capture the new information contained in the passage; other times, the seed question is simply nonsense.

Reject This example candidate has no valid discriminating question. In other words, these are negative examples. Sometimes, the target passage contains no new information at all; however, it may contain new information about other answer spans but not the one in the example candidate. In our previous John Doe example, there is no new information about "the Olympics", except indirectly.

Skip It is unclear if there is a valid discriminating question for this example candidate. This could be due to awkward or cumbersome answer spans: for instance "two gold medals at the Olympics in 2012." Alternatively, it could seem unclear if there is new information about an answer span due to its indirect relationships with other entities. Finally, it could be difficult to write an information-seeking question even though there is obviously new information: for instance, writing a question with the answer span "John Doe" in the previous example.

Each example candidate is considered by two annotators. Unless both annotators agree to Add, Reject, or Skip, a third annotator decides. In examples where one annotator chose Context or Edit, the third annotator is responsible for writing the correct question according to the guidelines. If one annotator chose Add or Reject and the other skipped, the third annotator can confirm the Add or Reject or also skip if they cannot decide. See Appendix A.2 for the annotation interface.

3.4 Question Writing Guidelines

Note that writing a single, context-free, and information-seeking question that summarizes the difference between the two passages can be challenging. In cases where it seemed impossible, annotators are encouraged to skip the example. For cases where additional Context was needed, annotators are encouraged to add as much context without sacrificing fluency, so that the question can be answered without awareness of the source passage. When an annotator writes a question from scratch

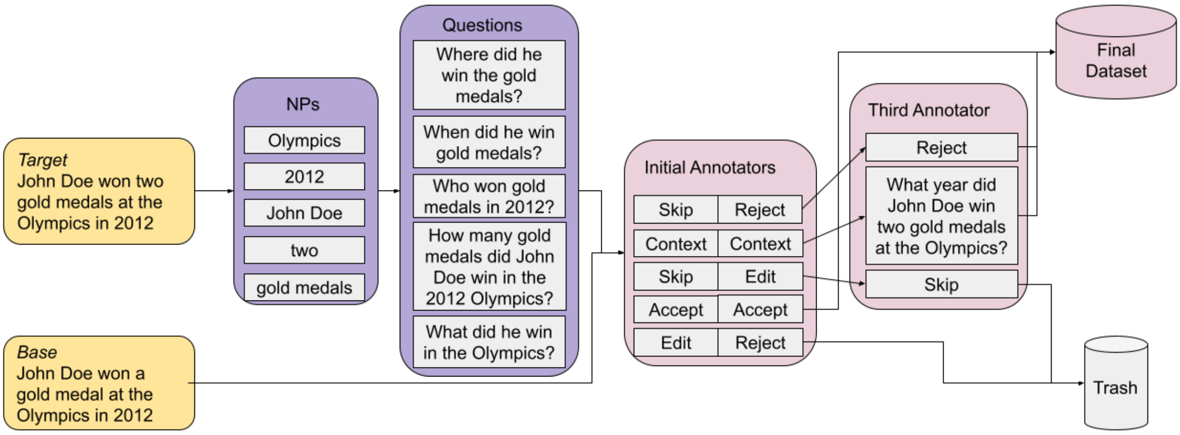


Figure 2: DiffQG annotation process. Noun phrases are extracted from the target passage and a question generation model seeds initial questions. Annotators decide if the generated questions serve as satisfactory discriminating questions (Accept), must be edited (Context, Edit), or contain no new factual information (Reject). After the first phase, a third annotator resolves indecision (Skip), leaving us with a set of questions and negative examples. If the original two annotators disagree or the third annotator cannot resolve indecision, the example is discarded.

in the Edit case, they are encouraged to think of a question that either would have a different answer, be unanswerable, or have a false precondition if posed against the base passage. While conditioning on a single answer span reduces ambiguity, the task is still ambiguous, which is unavoidable when handling large and complex edits.

	Passages	Answers	Avg. Edited tokens
w/ change	391	759	12.9
w/o change	478	1153	14.1
Total	672	1912	13.7

Table 1: Dataset statistics for DiffQG. Edited tokens represents the average tokens added or removed in a given passage pair.

3.5 Data Statistics

Our initial annotation process starts with $8,530$ example candidates drawn from 999 passage pairs. Annotators skipped nearly 75% of the example candidates, leaving $1,912$ examples. Of those, roughly 40%, or 759, had a factual change and thus a discriminating question written about them, leaving 1153 negative examples. Of the spans where a factual change was detected, annotators modified the question in 65%, or 494, of the examples: 45% are labeled as Context and 20% as Edit. Detailed dataset statistics can be found in Table 1.

Note that on all cases where a question was accepted as is or considered a negative example, at least two annotators agreed on that rating. However, human written questions are not verified; both annotators agree that there exists a discriminating question but not necessarily what it is. To address this, we evaluate a small set of fifty questions and found that a second annotator would write an equivalent question around 85% of the time.

4 Motivation

In the previous section, we described DiffQG and its annotation procedure. As mentioned, the purpose of DiffQG is to detect and describe factual changes. In particular, DiffQG is a rough measurement of a model's ability to automatically construct a database of question-answer pairs that encapsulate the changes. There are many possible formats that could be used as an alternative to summarize factual changes, such as paragraphs, knowledge base triples, or individual claims.

While paragraphs can contain nuance, they lack atomicity. It is thus difficult to tell what exactly changed or otherwise compare two changes to each other. This makes them less useful as a database.

On the other hand, knowledge base triples are limiting in the types of factual changes that can be described: regardless of the exact setup, the nodes and relations come from some form of fixed vocabulary that may require discarding interesting changes. For instance, changes related to a set of entities, date ranges, various numbers, or abstract information may all be challenging.

Another alternative method would be a list of claims, similar to Vitamin-C (Schuster et al., 2021). This method is also atomic and more flexible than knowledge base triples. However, question-answer pairs have a few advantages. First, question-answer pairs are semi-structured information, forming a loose key-value pair. Factual edits may change the answer to an existing question or add information corresponding to an entirely new information, requiring a new question. Conversely, claims are more difficult to relate to each other.

Finally, question-answer pairs are interesting because question answering is interesting. Previous work has seen the use of a database of question-answer pairs as a method to improve question answering performance (Lewis et al., 2021). A good method for automatically creating and updating such a database thus seems quite useful. As factual corpora change over time, we envision constructing such a database to require iterative updates.

5 Metrics

DIFFQG can be used to measure performance on three related tasks.

Factual Change Detection Given an example consisting of a base passage, target passage, and answer span, the goal is to determine whether there exists a valid differentiating question. In other words, whether there is new information about this answer span that is present in the target passage when compared to the base passage. To measure this, we report accuracy, precision, recall and F1 score over the existence of a differentiating question in our annotations. Note that always predicting no change achieves 60.3% accuracy but 0% F1, but random guessing corresponds to 44.1% F1.

Discriminating Question Generation Given a target passage and answer span, write a specific, unambiguous and information-seeking query that can be answered with the target passage. To measure this, we compare machine generated questions to those that humans verified, edited, or hand wrote. We use two model-free metrics Rouge-1 and Rouge-L (Lin, 2004) which measure the token-level overlap and longest subsequence overlap of the questions, respectively. We also consider two model-based metrics, BLEURT (Sellam et al., 2020), which is a learned evaluation for text similarity based on BERT (Devlin et al., 2019), and a query similarity model (Reimers and Gurevych, 2019) trained on Quora Question Pairs [2].

Note that we evaluate discriminating question generation despite using a question generation model in our annotation procedure. Note that all of these questions are reviewed by humans and only the very fluent ones are kept. As question generation models vary in which of their productions are very fluent, this set is less trivial than it would initially appear. Nonetheless, we also separate human-written or edited questions and evaluate that set independently.

Full System This is the overall measure of performance on DIFFQG. We reuse the metrics from discriminating question generation, using 0.0 for BLEURT, ROUGE-1, ROUGE-L, and Query Similarity if the factual change detection is incorrect.

6 Methods

As mentioned, DIFFQG can be thought of as a composition of two tasks: factual change detection and discriminating question generation. Our simple baseline systems thus treat this as a pipeline, first predicting whether or not there is a factual change and then generating a discriminating question if there is. We also present baseline models that solve both tasks jointly with a single prediction. Our methods are illustrated in Figure 3.

Note that none of our methods use any part of DIFFQG as training data, as the dataset is only intended to be used for evaluation. Models are instead trained on larger existing datasets for question generation and factual change detection.

6.1 Factual Change Detection

We propose five baselines based on answer equivalence or both question and answer equivalence.

Answer Equivalence Baselines

Our trivial baseline (*Overlapping Answer*) classifies an example as having a factual change if and only if the answer span is not present in the base passage. The span is normalized before looking for token overlap with the passage.

Our simple model-based baseline is similar but uses an Answer Equivalence model (Bulian et al., 2022). It compares the target answer span against all valid base answer spans, finding a factual change if it does not match any of them. The Answer Equivalence model additionally takes as input a candidate question for each answer span.

---

[2] huggingface.co/cross-encoder/quora-roberta-large

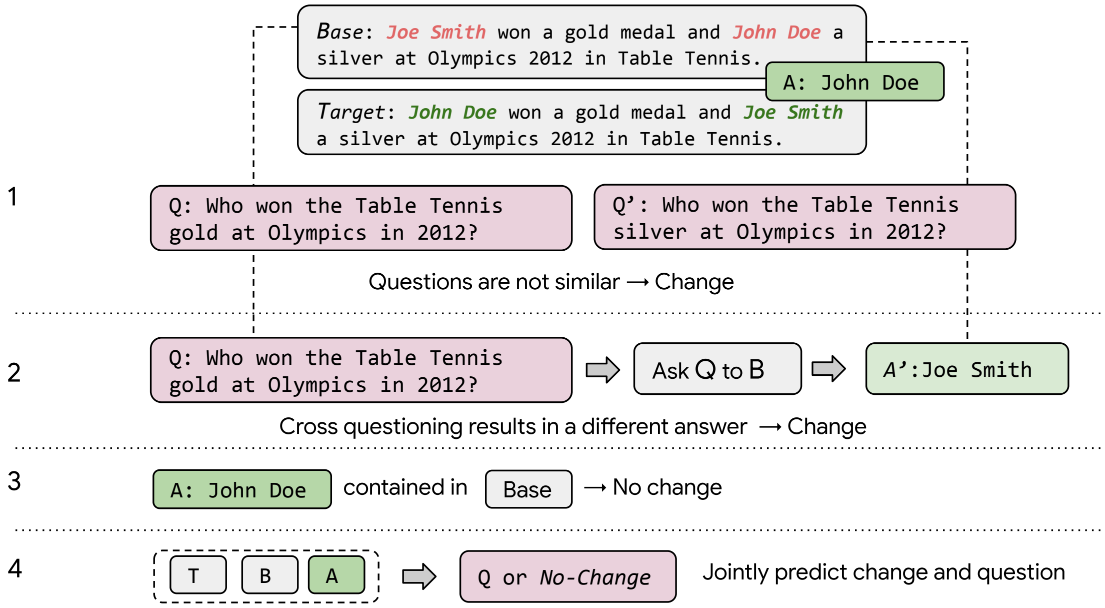


Figure 3: Methods for factual change detection on DiffQG: (1) Question similarity (2) Cross-Questioning. The QA-equivalence method combines (1) and (2), deciding it is a Change only when both the systems find a Change. (3) Overlapping Answers (4) Language model that jointly learns the task of factual change detection and discriminating question generation, decoding the discriminating question or a special token indicating no change.

Question-Answer Equivalence Baselines

The previous methods only consider the answer span, ignoring the context. Here, we consider methods that also use a question generation model on the passages and answer span to determine if there is new information.

For the first method, we find base answer spans equivalent to the target answer span using the *Overlapping Answer* method. Then, we want to see if the questions generated from those answers would also be equivalent in both passages. To do so, we use a T5-XXL (Raffel et al., 2020) model trained on Quora Question Pairs [3] to predict whether the pair of questions is "duplicate" or "not duplicate" . If the question is not a duplicate, then we consider this example to have a factual change. Thus, answer spans present in both passages but with different contexts could now be identified as having a factual change. This will increase the recall of the *Overlapping Answer* method.

The second approach adds a cross questioning filter (*Cross-Q*). Given a candidate question generated from target passage, we attempt to answer the question with the base passage using a reading comprehension model. We train a T5-XXL model on SQuaD v2 (Rajpurkar et al., 2018) question-answering dataset to take the passage and question as input and output the answer. If the model predicts no answer or a different answer from the target span, we classify the example as having a factual change. Finally, the *QA-equivalence* method combines both the query similarity model and cross question model to boost precision. In this case, we consider an example to have a factual change only when both methods determine a factual change.

6.2 Question Generation

Each of our factual change detection baselines is then combined with a question generation model. We use a similar T5-XXL model finetuned on SQuAD as described in Section 3.2. Unsurprisingly, the model we use to seed the questions can do well on the questions that it wrote originally; however, this is an unfair baseline. Thus, we additionally test a version of the model that is sampled and also a retrained version using a different seed. We also test training a similar model trained on Natural Questions (Kwiatkowski et al., 2019).

6.3 Joint systems

Many of the techniques described Section 6.1 are inefficient, requiring multiple runs of various models. For instance, the Query Similarity method requires one model run for each answer span in the base passage per example, which corresponds to quadratic runs for each pair of passages. We

---

[3] https://www.kaggle.com/c/quora-question-pairs

also explore methods that can directly compare the base and target passage without the need for any intermediate steps. These methods instead jointly detect if there is a factual change and generate a discriminating question.

Finetuning on Silver Data

We mine additional pairs of Wikipedia passages using the same process as in Section 3.1. We then identify every possible answer span from the target passage. We create silver training examples for the factual change detection component of the task by labeling each target answer span using our best heuristic method, *QA-equivalence*.

We then convert these labels into a text-to-text task. For each example with a factual change, we use the question generated by the SQuAD model (Section 6.2) as the target. For questions without a factual change, we use "None" as the target. The input to the model is the concatenation of the base and target passages with the target answer span marked by a special token.

The model is a T5-XXL initialized from the same question generation model as the model that produced the original questions.

Finetuning on VITAMINC

The VITAMINC task (Schuster et al., 2021) also has a factual change detection component. We sample negatives from the VITAMINC Revision Flagging dataset, using negative examples with a random noun phrase chosen as the answer span.

For positive examples, we need to identify a specific answer span that contains a factual change as well as the corresponding discriminating question. While there is no direct counterpart of this task in VITAMINC, the Fact Verification task is somewhat similar. The dataset consists of an evidence *e*, a simplified claim *c* supporting *e*, a companion edited sentence *e'* and an edited-claim *c'* refuting *e* and supporting *e'*. An answer span *a* is identified based on the token-level diff between *(c, c')* and a question generated from *c* conditioned on *a* using the question generation model in Section 6.2. Because *c* is a simple sentence, we anecdotally find the generated questions to be of high quality.

The dataset (*e*, *e'*, *a*) is converted to a text-to-text task and used to finetune a T5-XXL model following the same steps as above. Note that a model trained on an equal amount of positives and negatives yielded poor performance on DIFFQG. In our final VITAMINC silver dataset, we used only 10% negatives to achieve a reasonable performance.

Model	Acc	P	R	F1
Random	50.0	39.5	50.0	44.1
Overlapping Ans	82.1	79.1	74.6	76.7
Answer Equivalence	81.1	84.6	64.2	73.0
Query Similarity	77.7	65.4	93.3	76.9
Cross-Q	76.9	65.9	86.8	74.9
QA-equivalence	**83.9**	76.7	85.5	**80.9**
FT on QA-equivalence	<u>82.5</u>	78.4	77.1	<u>77.7</u>
FT on VITAMINC	81.5	79.9	71.4	75.4

Table 2: Metrics for factual change detection. Note that none of these models have change detection training data and are instead verifying with other tasks or heuristics. The random baseline assumes guessing change or No change with equal probability. Acc=Accuracy,P=Precision,R=Recall,F1=F1 Score. **Bold** indicates the best model, second best model is <u>underlined</u>.

7 Results and Discussion

We present results separately for factual change detection, question generation and the full system. We also report results separately for the overall performance and the performance on only the subset of questions that are human written; those sentences labeled as Edit or Context in the annotation phase. Selected examples with model outputs are provided in Appendix A.1 to illustrate the capabilities and typical errors baseline.

7.1 Factual Change Detection

Table 2 compares the performance of various systems on the factual change detection task. We find that QA-equivalence performs better than heuristic baseline methods. In particular, it better handles cases where the answer span text is unchanged, but the surrounding context has changed. For example, in the passage *"On the New Hampshire Executive Council, Laconia is in the 1st District, represented by <ADD: Republican Joe Kenney> <DEL: Democrat Michael J. Cryans>."*, QA-equivalence correctly captures the new information associated with the answer span *"New Hampshire Executive Council"* in the form of the question *"What state council does Joe Kenney represent Laconia in?"* However, the method is prone to detecting spurious changes even when the passages have no semantic edit as illustrated in Appendix A.1.

The joint system finetuned on silver data from QA-equivalence does not seem to improve upon

QA-equivalence. While it seemingly benefits from the additional context, it still struggles with long and complex edits. However, this model only requires a single inference to do both tasks.

The VITAMINC trained model, despite having access to additional data, was also unable to improve on our baseline. VITAMINC style edits are substantially different than DIFFQG edits, generally only consisting of small changes. Thus, the model finetuned on VITAMINC performs poorly on large phrase changes or sentence refactors.

7.2 Question Generation

Model	R-1	R-L	QSim	BLRT
SQuAD-seed	71.7	74.9	66.3	74.1
SQuAD-sampled	59.6	63.2	57.8	65.6
SQuAD-retrained	58.0	61.7	58.9	65.6
NQ	20.4	39.6	22.9	41.5

Table 3: Variation in performance of question generation models on the positive subset of DIFFQG. We report three different versions of the SQuAD model, where the first model is the same as we used to seed the annotations. R-1=ROUGE-1, R-L=ROUGE-L, QSim=Query Similarity model-based accuracy, BLRT=BLEURT.

In Table 3, we compare our question generation baseline models on the subset of the positive examples. In Table 4, we examine the same models on the subset of those that are human written: examples with a change from Table 1.

Model	R-1	R-L	QSim	BLRT
SQuAD-seed	56.5	61.4	48.2	61.3
SQuAD-sampled	50.9	55.2	47.0	58.3
SQuAD-retrained	50.3	54.5	50.4	59.4
NQ	20.4	39.1	20.0	40.2

Table 4: Variation in performance of question generation models on human written questions of DIFFQG. The first model is the same as what we used to seed the questions. R-1=ROUGE-1, R-L=ROUGE-L, QSim=Query Similarity model-based accuracy, BLRT=BLEURT.

The primary goal of this evaluation is to test whether the questions directly produced by the seed model described in Section 3.2 are still useful for evaluating systems on DIFFQG. We find from sampling from that same model and from retraining with the same process (as described in Section 6.2) that performance on the overall set degrades considerably. This suggests that unless someone had access to the same model, these questions that are human-verified but not human written can still be useful for evaluation. Nevertheless, the seed model can be thought of as a rough ceiling on current question generation performance on DIFFQG.

The human written questions (see Table 4) seem to be much more challenging for the question generation models to replicate. Performance degrades substantially: naturally it degrades the most for the seed model that wrote some of the questions in the overall dataset, which it should exactly match.

We note also that a question generation model finetuned on Natural Questions (Kwiatkowski et al., 2019) yields a significantly different question style than SQuaD. This is likely because SQuAD questions are originally generated from passages, while Natural Questions are more free form. In addition the Natural Questions model is found to hallucinate in numerous scenarios. This reflects on the poor performance of the Natural Questions-trained question generation model on DIFFQG.

As a caveat, the possible universe of questions written to summarize a factual change can be very large. While restricting to a single answer span reduces this space, we still find scenarios with multiple valid questions. Thus, there may be some disagreements where the model generates a completely valid question that is simply not the most pertinent one according to our annotators.

7.3 Full System

Full DIFFQG metrics are presented in Table 5 and include the two finetuned systems that are trained on VITAMINC and QA-equivalence, respectively, as well as two pipelined systems with factual change detection models attached to a question generation model. For the pipelined experiments, we use the retrained SQuaD model described in Section 6.2. We evaluate these models on the full DIFFQG as well as human written subset.

Overall, all of the systems are relatively close in performance. QA-equivalence works the best, with the finetuned version and simple heuristic model close behind, indicating substantial room for future innovation. On the human written subset, the performance drops significantly further highlighting the challenge of the human written questions.

Models		Factual Change Detection			Full System					
					All			Human written		
Change	QG	P	R	F1	R-L	Qsim	BLRT	R-L	Qsim	BLRT
Pipelined systems										
Overlapping Answer	SQuAD	79.1	74.6	76.7	71.3	70.8	72.3	40.6	38.1	43.6
QA-equivalence	SQuAD	76.7	85.5	**80.9**	71.4	70.6	**72.7**	46.8	43.7	**50.8**
Joint systems										
FT on QA-equivalence		78.4	77.1	77.7	71.2	70.0	72.3	42.7	36.6	45.9
FT on VitaminC		79.9	71.4	75.4	69.9	68.1	71.1	36.3	32.2	39.9

Table 5: Full system performance of the pipelined and joint systems on DiffQG. Note that the "All" component of the full system metric includes all of DiffQG while the "Human written" portion includes only questions edited by the annotators. The pipelined systems use the retrained SQuaD model for their question generation component. **Bold** represents the best system, second best is underlined. R-L=Rouge-L, Qsim=Query similarity model based accuracy, BLRT=Bleurt, FT=finetuned

8 Related Work

Factual Edits Factual change detection has been of recent interest to the community. For instance, WikiAtomicEdits (Faruqui et al., 2018) rely on Wikipedia revisions to learn to discriminate factual edits. Closest to our work is VitaminC (Schuster et al., 2021) which aims to generate a discriminating claim given a pair of edited sentences. However, both of these datasets primarily rely on smaller edits, frequently consisting of a single entity or number substitution. For instance, VitaminC examples have a median of four token changes and WikiAtomicEdits examples have a median of two token changes. Moreover, these edits are easier to detect using heuristics such as noun or entity overlap. On the other hand, DiffQG examples have a median of thirteen token changes that can involve multiple entity updates. Further, the surrounding contextual information for an entity could be updated even when the entity itself is present in both passages. This makes DiffQG edits harder to summarize and substantially different than previous work; this is also observed in Section 7.1 where using VitaminC training data to solve DiffQG yields poor performance.

Recent work such as Fruit (Iv et al., 2022) and PEER (Schick et al., 2023) also operate on more complicated edits. Fruit generates updated sentences from a base passage given the new evidence in a Wikipedia article. PEER attempts to imitate the editing process using a sequence of planning steps. However, both of these primarily focus on generating the target update, while we focus on succinctly capturing the edited information. Further, the use of question generation as a device for discrimination is novel to the best of our knowledge.

Question Generation Question generation has been successfully applied to various purposes, including augmenting question answering systems (Duan et al., 2017; Lewis et al., 2021), capturing implicit information written about text (Pyatkin et al., 2021), and building soft knowledge bases (Chen et al., 2022). In this work, we apply question generation to the task of discriminating edited sentences. As far as we are aware, there is no prior work on evaluating question generation systems.

9 Conclusion

In this work, we introduce the DiffQG task and dataset to evaluate the ability of NLP systems to summarize changes between two related passages via question generation. We present several heuristic and model baselines as well as a set of metrics to measure performance on the dataset. The DiffQG task requires models to identify changes in factual relationships and ignore other stylistic edits. We find that existing approaches struggle under these conditions. Models trained to perform factual change detection and question generation jointly sometimes fail to understand even simple edits. We hope this work finds value in future research on this important problem.

Limitations

DiffQG is relatively small, consisting of less than a thousand questions and less than two thousand total examples. This makes us unable to provide a training set, limiting claims we can make about the difficulty of the task. Moreover, summarizing complex edits can have a large space of valid solutions. While using questions conditioned on an answer reduces this space, there's still room for ambiguity.

To make annotation easier, we use a question generation model; however, our goal is also to evaluate question generation models, complicating our story. Finally, most of the baselines we evaluate are some form of T5 (Raffel et al., 2020) model. It is possible that other model architectures could have solved this task more effectively.

Acknowledgements

We thank Jannis Bulian, Tal Schuster and William Cohen, as well as our anonymous reviewers, for their thoughtful comments and valuable feedback.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Jannis Bulian, Christian Buck, Wojciech Gajewski, Benjamin Boerschinger, and Tal Schuster. 2022. Tomayto, tomahto. beyond token-level answer equivalence for question answering evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Wenhu Chen, Pat Verga, Michiel de Jong, John Wieting, and William Cohen. 2022. Augmenting pre-trained language models with qa-memory for open-domain question answering.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rationale, evaluation and approaches. *Journal of Natural Language Engineering*, 4.

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

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-Aware Language Models as Temporal Knowledge Bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Manaal Faruqui, Ellie Pavlick, Ian Tenney, and Dipanjan Das. 2018. WikiAtomicEdits: A multilingual corpus of Wikipedia edits for modeling language and discourse. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 305–315, Brussels, Belgium. Association for Computational Linguistics.

Robert Iv, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. 2022. FRUIT: Faithfully reflecting updated information in text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686, Seattle, United States. Association for Computational Linguistics.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Sawan Kumar and Partha Talukdar. 2020. NILE : Natural language inference with faithful natural language explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8730–8742, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Valentina Pyatkin, Paul Roit, Julian Michael, Yoav Goldberg, Reut Tsarfaty, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1429–1441, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Pedro Rodriguez and Jordan Boyd-Graber. 2021. Evaluation paradigms in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9630–9642, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timo Schick, Jane A. Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. 2023. PEER: A collaborative language model. In *International Conference on Learning Representations*.

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin C! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics.

Thibault Sellam, Amy Pu, Hyung Won Chung, Sebastian Gehrmann, Qijun Tan, Markus Freitag, Dipanjan Das, and Ankur Parikh. 2020. Learning to evaluate translation beyond English: BLEURT submissions to the WMT metrics 2020 shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 921–927, Online. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018a. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal, editors. 2018b. *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*. Association for Computational Linguistics, Brussels, Belgium.

A Appendix

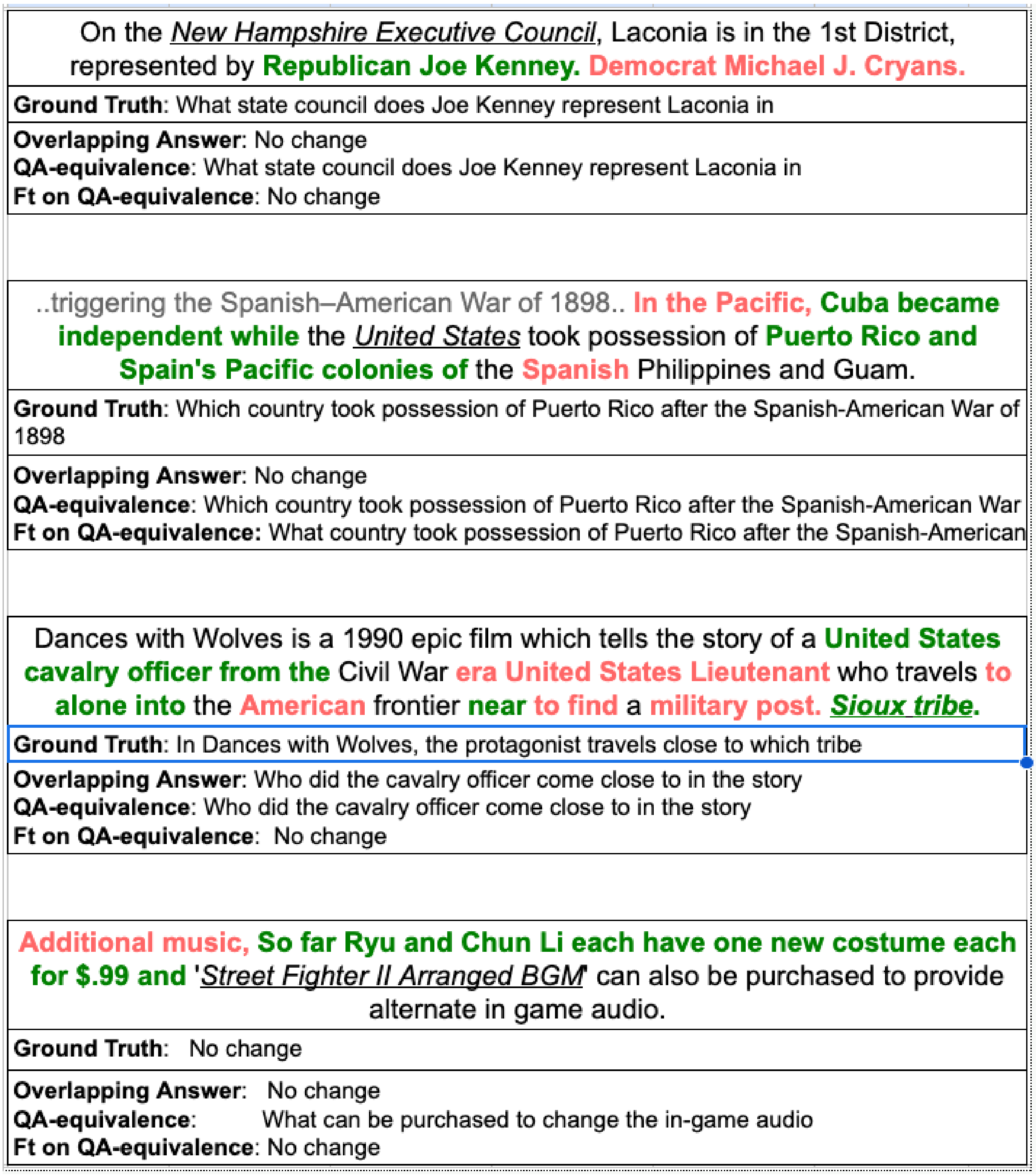
A.1 Qualitative Examples

Examples from DiffQG dataset are illustrated in Figure 4. The model outputs (success or failure) from various systems are also provided alongside.

A.2 Annotation Interface

Refer Figure 5 and Figure 6 for annotation interface of phase 1 and 2 respectively.

Figure 4: DIFFQG examples with predictions from various systems. The edited sentence is color coded with green for added tokens and red for deleted; the answer span is underlined. Additional context is omitted unless required for illustration (provided in gray). *No change* indicates there was no factual change for the example.



On the *New Hampshire Executive Council*, Laconia is in the 1st District, represented by **Republican Joe Kenney.** **Democrat Michael J. Cryans.**

Ground Truth: What state council does Joe Kenney represent Laconia in

Overlapping Answer: No change
QA-equivalence: What state council does Joe Kenney represent Laconia in
Ft on QA-equivalence: No change

..triggering the Spanish–American War of 1898.. **In the Pacific,** **Cuba became independent while** the *United States* took possession of **Puerto Rico and Spain's Pacific colonies of** the **Spanish** Philippines and Guam.

Ground Truth: Which country took possession of Puerto Rico after the Spanish-American War of 1898

Overlapping Answer: No change
QA-equivalence: Which country took possession of Puerto Rico after the Spanish-American War
Ft on QA-equivalence: What country took possession of Puerto Rico after the Spanish-American

Dances with Wolves is a 1990 epic film which tells the story of a **United States cavalry officer from the** Civil War **era United States Lieutenant** who travels **to alone into** the **American** frontier **near** **to find** a **military post.** ***Sioux tribe*.**

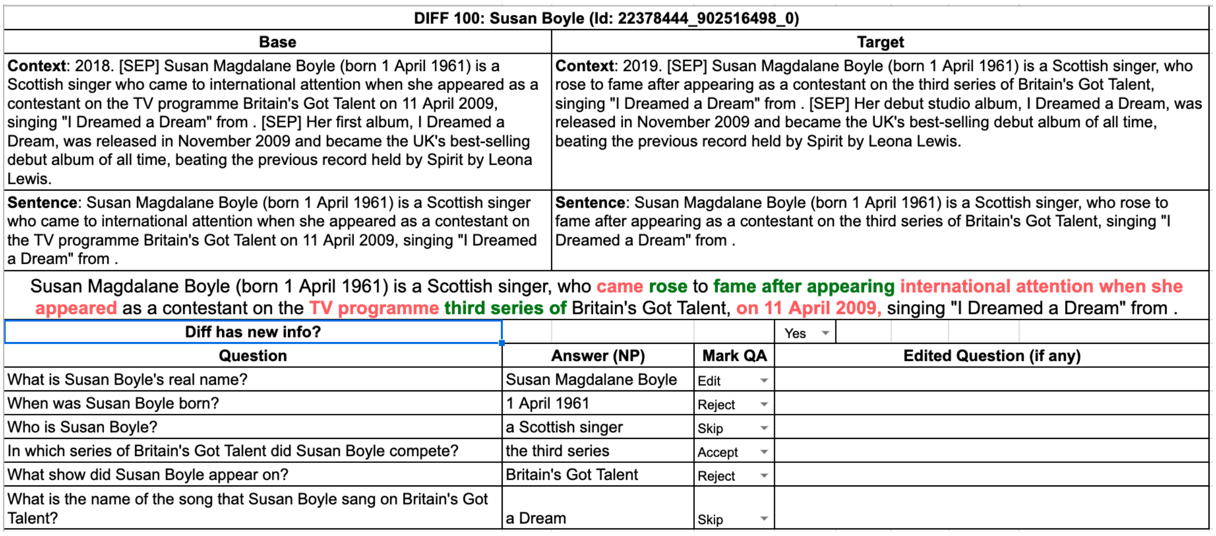
Ground Truth: In Dances with Wolves, the protagonist travels close to which tribe

Overlapping Answer: Who did the cavalry officer come close to in the story
QA-equivalence: Who did the cavalry officer come close to in the story
Ft on QA-equivalence: No change

Additional music, **So far Ryu and Chun Li each have one new costume each for $.99 and** '*Street Fighter II Arranged BGM*' can also be purchased to provide alternate in game audio.

Ground Truth: No change

Overlapping Answer: No change
QA-equivalence: What can be purchased to change the in-game audio
Ft on QA-equivalence: No change



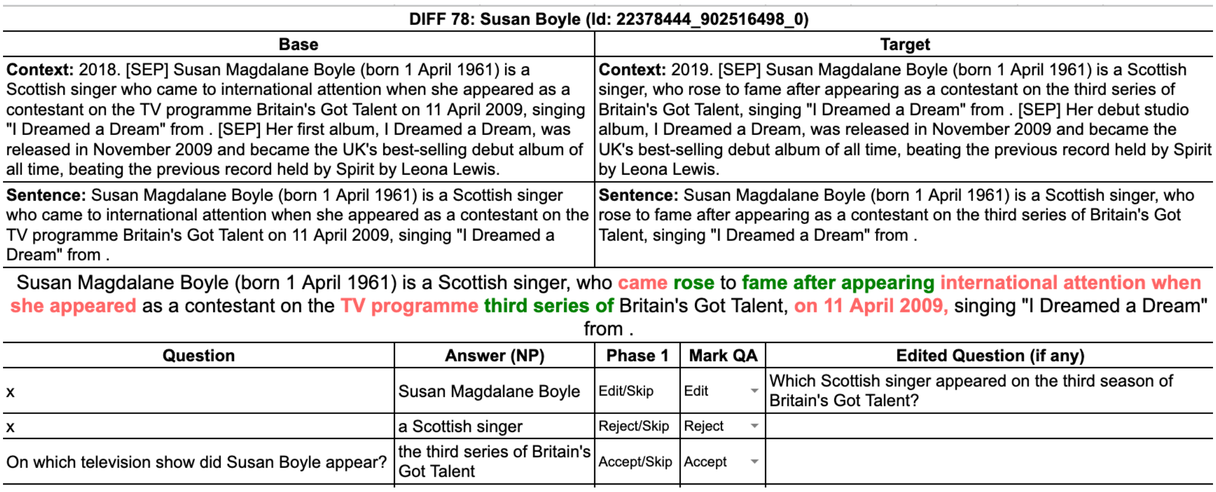
DIFF 100: Susan Boyle (Id: 22378444_902516498_0)

Base	**Target**
Context: 2018. [SEP] Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer who came to international attention when she appeared as a contestant on the TV programme Britain's Got Talent on 11 April 2009, singing "I Dreamed a Dream" from . [SEP] Her first album, I Dreamed a Dream, was released in November 2009 and became the UK's best-selling debut album of all time, beating the previous record held by Spirit by Leona Lewis.	**Context**: 2019. [SEP] Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer, who rose to fame after appearing as a contestant on the third series of Britain's Got Talent, singing "I Dreamed a Dream" from . [SEP] Her debut studio album, I Dreamed a Dream, was released in November 2009 and became the UK's best-selling debut album of all time, beating the previous record held by Spirit by Leona Lewis.
Sentence: Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer who came to international attention when she appeared as a contestant on the TV programme Britain's Got Talent on 11 April 2009, singing "I Dreamed a Dream" from .	**Sentence**: Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer, who rose to fame after appearing as a contestant on the third series of Britain's Got Talent, singing "I Dreamed a Dream" from .

Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer, who **came** **rose** to **fame after appearing** **international attention when she appeared** as a contestant on the **TV programme** **third series of** Britain's Got Talent, **on 11 April 2009,** singing "I Dreamed a Dream" from .

Diff has new info?	Yes		
Question	**Answer (NP)**	**Mark QA**	**Edited Question (if any)**
What is Susan Boyle's real name?	Susan Magdalane Boyle	Edit	
When was Susan Boyle born?	1 April 1961	Reject	
Who is Susan Boyle?	a Scottish singer	Skip	
In which series of Britain's Got Talent did Susan Boyle compete?	the third series	Accept	
What show did Susan Boyle appear on?	Britain's Got Talent	Reject	
What is the name of the song that Susan Boyle sang on Britain's Got Talent?	a Dream	Skip	

Figure 5: Interface for the first phase of annotations, where an annotator chooses one of the five options: Accept/Reject/Edit/Context/Skip. Each example is annotated by two annotators. If both agree, the example is accepted as is or goes to a third annotator for editing. If one of the annotators skips, the third annotator makes the final decision.



DIFF 78: Susan Boyle (Id: 22378444_902516498_0)

Base	**Target**
Context: 2018. [SEP] Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer who came to international attention when she appeared as a contestant on the TV programme Britain's Got Talent on 11 April 2009, singing "I Dreamed a Dream" from . [SEP] Her first album, I Dreamed a Dream, was released in November 2009 and became the UK's best-selling debut album of all time, beating the previous record held by Spirit by Leona Lewis.	**Context:** 2019. [SEP] Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer, who rose to fame after appearing as a contestant on the third series of Britain's Got Talent, singing "I Dreamed a Dream" from . [SEP] Her debut studio album, I Dreamed a Dream, was released in November 2009 and became the UK's best-selling debut album of all time, beating the previous record held by Spirit by Leona Lewis.
Sentence: Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer who came to international attention when she appeared as a contestant on the TV programme Britain's Got Talent on 11 April 2009, singing "I Dreamed a Dream" from .	**Sentence:** Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer, who rose to fame after appearing as a contestant on the third series of Britain's Got Talent, singing "I Dreamed a Dream" from .

Susan Magdalane Boyle (born 1 April 1961) is a Scottish singer, who **came** **rose** to **fame after appearing** **international attention when she appeared** as a contestant on the **TV programme** **third series of** Britain's Got Talent, **on 11 April 2009,** singing "I Dreamed a Dream" from .

Question	**Answer (NP)**	**Phase 1**	**Mark QA**	**Edited Question (if any)**
x	Susan Magdalane Boyle	Edit/Skip	Edit	Which Scottish singer appeared on the third season of Britain's Got Talent?
x	a Scottish singer	Reject/Skip	Reject	
On which television show did Susan Boyle appear?	the third series of Britain's Got Talent	Accept/Skip	Accept	

Figure 6: Interface for the second phase of annotations, where a third annotator will rephrase a question and/or decide on a disagreed-upon annotation. Here, the annotator writes a new question for the answer span given the *Edit* annotation, and decides to confirm the *Reject* and *Accept* annotations of the other two examples. Note that for *Edit* or *Reject* annotations, to avoid bias, we do not display the seed-question to the annotators and instead display a *x*.

Contextual Dynamic Prompting for Response Generation in Task-oriented Dialog Systems

Sandesh Swamy
AWS AI Labs
sanswamy@amazon.com

Narges Tabari
AWS AI Labs
nargesam@amazon.com

Chacha Chen[*]
University of Chicago
chacha@uchicago.edu

Rashmi Gangadharaiah
AWS AI Labs
rgangad@amazon.com

Abstract

Response generation is one of the critical components in task-oriented dialog systems. Existing studies have shown that large pre-trained language models can be adapted to this task. The typical paradigm of adapting such extremely large language models would be by fine-tuning on the downstream tasks which is not only time-consuming but also involves significant resources and access to fine-tuning data. Prompting (Schick and Schütze, 2020) has been an alternative to fine-tuning in many NLP tasks. In our work, we explore the idea of using prompting for response generation in task-oriented dialog systems. Specifically, we propose an approach that performs *contextual dynamic prompting* where the prompts are learnt from dialog contexts. We aim to distill useful prompting signals from the dialog context. On experiments with MultiWOZ 2.2 dataset (Zang et al., 2020), we show that contextual dynamic prompts improve response generation in terms of *combined score* (Mehri et al., 2019a) by 3 absolute points, and a massive 20 points when dialog states are incorporated. Furthermore, human annotation on these conversations found that agents which incorporate context were preferred over agents with vanilla prefix-tuning.

1 Introduction

With the advent of large language models (LLMs), a vast majority of NLP tasks, including dialog systems, further fine-tune these LMs for their downstream tasks. Although these approaches provide substantial improvements over traditional task-specific models (Ham et al., 2020; Hosseini-Asl et al., 2020; He et al., 2022), it is a time consuming process that also involves significant use of energy/resources in the form of compute. These approaches also require tuning and storing parameters for each downstream task.

A more recent line of work, explores "prompting" LLMs to elicit the necessary knowledge required for the downstream tasks (Shin et al., 2020; Gao et al., 2020; Schick and Schütze, 2020; Petroni et al., 2019; Lee et al., 2021; Zhu et al., 2022). Prompts composed of tokens or short pieces of text (*discrete* prompts) inserted at the end of the input examples. These prompts are typically manually defined based on the specific downstream task. The main motivation behind these approaches stems from the idea that the large corpora that these language models are trained on contain relevant information which is pertinent to the task on hand.

Adapter-tuning was proposed as an alternate approach to fine-tuning. These methods only train task-specific layers that are inserted within pre-trained LMs. Such a lightweight approach that add about 4% task-specific parameters has shown to obtain comparable performances to their fine-tuning counterparts (Rebuffi et al., 2017; Houlsby et al., 2019; Lin et al., 2020a).

Drawing inspiration from prompting, *prefix-tuning* approaches (Li and Liang, 2021) were proposed as another alternative to fine-tuning. These approaches pre-pend a sequence of task-specific *continuous* vectors (aka prefix-) to the input. In contrast to prompting, the prefix consists of free parameters that do not correspond to actual real tokens. Such an approach is more prevalent since it only optimizes the prefix and does not tune parameters of the entire LM.

Most of the existing approaches use static prompts, i.e., the same set of tokens are used as "prompt tokens" regardless of input. However, we believe that taking context into consideration is critical especially in response generation since the current response has to fit not only the domain but also the information being requested in previous turns. For example: In the MultiWOZ dataset, if a customer asks about train bookings, the agent response has to restrict itself to that particular do-

---

[*] Work done during an internship at AWS AI Labs

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3102–3111
May 2-6, 2023

main. To address this problem, we explore the idea of generating input-dependent or *contextual* prompts. We want the prompts to capture and encode different signals for different turns of dialogs depending on the context, hence, we call our approach *dynamic context prompting*. This way, we hope to distill useful signals into the prompts and provide the model with adequate signals to generate a desired system response. In this work, we explore the potential of using dialog context within a prefix tuning approach for the task of response generation in task-oriented dialog systems (TOD). The contributions of this paper are summarized as:

- we propose a context-dependent prefix-tuning method for dialog response generation in TOD systems.
- to illustrate the benefits of such an approach, we conduct experiments on the MultiWOZ dataset. We show that our model significantly outperforms the original task-dependent design of the prefix-tuning method.

2 Related Work

2.1 Dialog Generation

With the prevalence of LLMs, the quest for an answer to "how do we effectively adapt such models for dialog generation?" has been on the forefront of researchers' minds in the dialog community. For task-oriented dialogs, fine-tuning large pre-trained models such as GPT-2 or T5 has made great progress on benchmarks recently (Ham et al., 2020; Hosseini-Asl et al., 2020). Built upon these advances, more recent line of work investigates the effectiveness of using multi-task learning (Su et al., 2021; Lin et al., 2020b; Yang et al., 2021), or pre-training the model on external dialog corpora (Peng et al., 2021; Liu et al., 2021). More recently, prompting has been used to address the sub-task of dialog state tracking (Lee et al., 2021; Zhu et al., 2022). Different from those works, we focus on the task of dialog response generation.

2.2 Prompt-based Learning

As an alternative to the fine-tuning paradigm, prompting involves a sequence of tokens appended to the input text, which can then induce the model to engage in a certain behavior suited to the task. Since the release of GPT-2 (Radford et al., 2018, 2019; Brown et al., 2020), many prompt-related papers have emerged. Most of the leading approaches in prompting use **task-specific prompts**, ranging from discrete prompts (Shin et al., 2020; Gao et al., 2020; Schick and Schütze, 2020; Petroni et al., 2019) to continuous "soft prompts" (Li and Liang, 2021; Lester et al., 2021). These methods have a fixed prompt for each task. However, in dialog systems specifically, the context varies for every turn. In our work, we aim to design prompts which are **context-dependent**.

3 Problem Statement

Response generation is one of the tasks carried out in dialog systems usually in addition to dialog state tracking (DST). Given a dialog context (previous turns between the system and the user) $C = [u_1, s_1, ..., u_{n-1}, s_{n-1}]$ and the current user utterance u_n, the goal of response generation is to generate system response s_n. Note that in the actual task, we generate *delexicalized* system responses, given all the groundtruth previous turns as input, following previous works (Hosseini-Asl et al., 2020; Wen et al., 2015).

Techniques mentioned in (Ham et al., 2020; Hosseini-Asl et al., 2020) rely on fully fine-tuning LLMs to carry out this task. In contrast, our approach builds on the prefix-tuning framework, but incorporates dialog context, C, as an additional signal for the prefix tokens. As a supplement to context C, we added dialog state information D (up to the current turn) to further help response generation.

4 Contextual Dynamic Prompting Framework

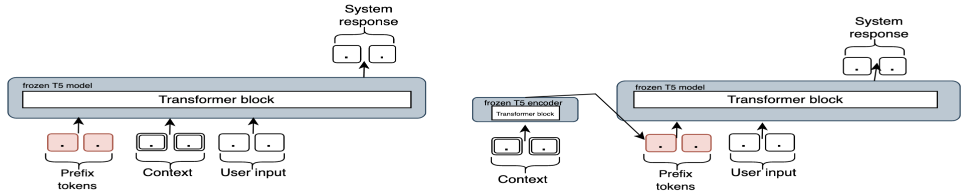
4.1 Prefix-tuning for Response Generation

Our work is built on top of prefix tuning for generation tasks (Li and Liang, 2021), which adds a fixed set of tunable prefix tokens/prompts to the original input x to obtain a new input, [PREFIX; x]. Following the denotation in (Li and Liang, 2021), we use $P_\theta[i,:]$ to denote the i_{th} prefix. $P_\theta[i,:]$ is generated by:

$$P_\theta[:,:] = MLP_\theta(P'), \tag{1}$$

where P' is a fixed smaller matrix as input to a feedforward neural network (MLP_θ). The training objective of prefix-tuning is same as fine-tuning, i.e., the following log-likelihood objective:

$$\max_\theta \log p_\phi(y|x),$$



(a) The original prefix-tuning framework where a set of prefix tokens are added to the input which also consists of dialog context *C* in addition to the current user input u_n

(b) Our changes which now incorporate the dialog context *C* into the prefix by obtaining a representation by passing through a frozen T5 encoder.

Figure 1: The figures above indicate the differences between the vanilla prefix-tuning approach compared to our approach. In both these variants, only the prefix tokens are tuned.

where y is the decoder output and x is the input. θ represents the trainable parameters in the prefix tuning feedforward neural network and ϕ denotes all other parameters that include the frozen parameters of the large language model.

For our task of response generation, we concatenate the prefix with the dialog context and the current user utterance as input [PREFIX; $u_1, s_1, ..., u_{n-1}, s_{n-1}, u_n$]. The target output is the system response s_n as seen in Figure 1 (a).

We adopt T5 (Raffel et al., 2020) as the pre-trained language model. T5 employs an encoder-decoder framework which is prevalent in seq2seq tasks (Sutskever et al., 2014; Cho et al., 2014).

4.2 Contextual Prefix-tuning

In vanilla prefix-tuning, the parameters of the prefix are fixed after training for any particular task to be reused. However, a dialog system involves having multiple turns of conversation between a system and the user. It is imperative in such systems to dynamically incorporate contextual information to carry out a meaningful conversation with the user. We explore how we can distill the dialog context information into the prefix with a prompt encoder.

Different from the original design, we want to encode additional signals into the prefix that differs for each input instances. In other words, we want to generate **contextual prefix** or **contextual dynamic prompts**.

Formally, we modify the equation (1) as follows:

$$P_\theta[:,:] = MLP_\theta(encoder(C)), \qquad (2)$$

where $C = [u_1, s_1, ..., u_{n-1}, s_{n-1}]$ represents the dialog context. We first obtain the representation of the dialog context by feeding C into a T5 encoder which is kept frozen as shown in Figure 1 (b). Subsequently, we use the prompt encoder, i.e., the feedforward neural network, to get the prefix. The generated prefix P_θ is then concatenated with only the current user utterance. Instead of concatenating the whole context as the input to the T5 decoder, we first distill the signal into the prefix tokens. As a consequence of freezing the T5 encoder which generates the context representation, we still have the same number of tunable parameters as the original prefix-tuning framework.

4.3 Input-dependent Prefix-tuning with Dialog State

In most task-oriented dialog systems, we also have access to the dialog state at every turn in addition to dialog context. The dialog state has information such as requested slots and filled slots at every turn. We provide the dialog state D in addition to the context C to obtain contextual dynamic prompts. As a result, we will now modify equation (2) as:

$$P_\theta[:,:] = MLP_\theta(encoder(C; D_{n-1})), \qquad (3)$$

we only provide the most recent dialog state D_{n-1} which is an amalgamation of all previous dialog states $D_{<n-1}$.

5 Experimental Settings

5.1 Dataset and Metrics

We evaluate our proposed framework and model on the MultiWOZ 2.2 dataset (Zang et al., 2020; Budzianowski et al., 2018) which is a large-scale, multi-domain, human-human task-oriented dialog dataset collected via the Wizard-of-Oz framework where one participant plays the role of the system. It consists of seven domains including hotel, restaurant, attraction, train, taxi, hospital, and police, and an additional domain general for acts such as greeting or goodbye. Due to its multi-domain setting, complex ontology, and flexible human expressions, developing dialog systems on MultiWOZ is extremely challenging. The training data contain 8437 dialogs, the dev and test set contain 1000 dialogs each.

We use four evaluation metrics: **BLEU** (Papineni et al., 2002), **Inform**, and **Success** rates, and **combined score**. *Inform* measures whether the

	MultiWOZ 2.2						
	BLEU	Inform	Success	Combined Score	Av. len.	#uniq. words	#uniq. 3-grams
Prefix-Tuning	19.19	54.7	48.0	70.54	13.83	245	1671
Prefix-Tuning (with DS)	19.36	51.8	47.0	68.76	13.08	231	1626
Contextual Dynamic Prompt	19.16	58.1	50.5	73.46	**14.16**	231	1532
Contextual Dynamic Prompt (with DS)	17.94	**77.2**	**68.8**	**90.94**	14.02	**282**	**2390**

Table 1: **Performance Comparison**. All model performance are based on features from all modalities. Contextual Dynamic Prompt (with DS) has the best performance in combined score.

system provides an appropriate entity and *Success* measures whether the system answers all the requested attributes. Specifically, the Inform rate relates to attributes that allow the user to constrain database searches, e.g., restaurant location or price range (the informational slots) and the Success rate focuses on request-able slots, that can be asked by the user, e.g., phone number. Both are calculated on the level of dialogs. The combined score is calculated following (Mehri et al., 2019b) as $BLEU + 0.5 * (Inform + Success)$. We followed a standard script [1] to report different measures.

5.2 Human Evaluation

We chose a 10% subset of the evaluation set (randomly shuffled) conversations with a total of 728 turns across them and provided annotators with the responses generated by each of the methods described in section 4. Annotators were asked to rate each agent on a turn-level and to also pick the agent which carried out the best conversation. If annotators felt more than one agent did well, they could choose multiple agents. The agent numbers, when provided to annotators, were shuffled to avoid bias. Each agent is described as:

- Agent 1: Incorporates only prefix-tuning
- Agent 2: Incorporates prefix-tuning with Dialog State
- Agent 3: Incorporates contextual dynamic prompts
- Agent 4: Incorporates contextual dynamic prompts with Dialog State

When annotating on turn level, from these 728 turns, we saw that the agents tied on 596 occasions, agent 1 had outright win on 12 occasions, agent 2 on 22, agent 3 on 33 occasions, and agent 4 on 65 occasions. This shows that our technique of using contextual dynamic prompts for generating responses is effective (Examples in Appendix B).

Additionally, on the conversation level, we noticed that across 100 conversations, 37 were tied, and agents 3 and 4 were preferred in a total of *53* conversations confirming our hypothesis that incorporating context into prompts leads to better responses. We request readers to refer to Appendix A and B for more details about the annotation task.

6 Results

As shown in Table 1, contextual dynamic prompting with dialog states obtains a combined score of *90.94*, a 20 point jump from our baseline (prefix-tuning). In addition, even though we can't explicitly explain the drop in BLEU, the massive jumps in both success and inform suggest more transparency and coherence for the responses generated by the input-dependent prefix-tuning as these metrics focus on quality of informational and request-able slots in each turn. When comparing our results with the human annotations, we also see that *Agent 4* - which uses contextual dynamic prompting - wins 38 conversations (out of 100). This is majority of wins compared to *Agent 1* winning only 3 conversations, and *Agent 2* winning 7. This further emphasized that adding contextual dynamic prompts leads to better quality of responses. We show some examples from the human annotation task where *Agent 4* was chosen as the winning agent in Tables 2 and 3 below.

7 Conclusion

In our work, we proposed an approach that performs *contextual dynamic prompting* where prompts are learnt from dialog contexts with the goal of distilling useful prompting signals. In our experiments, we showed that contextual dynamic prompts improve response generation in terms of *combined score* (Mehri et al., 2019a) by 3 points, and by 20 points when *dialog states* are incorporated compared to the baseline. Our technique does not expose the models to additional knowledge sources. Human annotation on these conversations

---

[1] `https://github.com/Tomiinek/MultiWOZ_Evaluation`

Turn num	User turn	Agent 1 response	Agent 2 response	Agent 3 response	Agent 4 response
1	i am looking for a place to dine. the restaurant should be expensive and should serve chinese food.	there are [choice] [pricerange] [food] restaurants in town. do you have a preference for area or price range?	i have [choice] [pricerange] [food] restaurants. do you have a preference for area?	there are [choice] [pricerange] [food] restaurants. do you have a price range in mind?	there are [choice] [food] restaurants in the [pricerange] price range. is there a particular area you are interested in?
2	not in any specific area please. i need to book a table for 3 at 14:00 on saturday.	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref]. is there anything else i can help you with?	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref].	i was able to book you at [name] for [bookday] at [booktime]. your reference number is [ref]. is there anything else i can help you with?	i have made a reservation for you at [name]. the table will be reserved for 15 minutes. the reference number is [ref]. is there anything else i can help you with?
3	how about 13:00 then?	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref].	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref].	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref]. is there anything else i can help you with?	ok, i have you booked at [name]. your reference number is [ref]. is there anything else i can help you with?
4	what is the reference number for my booking?	your reference number is [ref]. is there anything else i can help you with?	the reference number is [ref]. is there anything else i can help you with?	your reference number is [ref]. is there anything else i can help you with?	the reference number is [ref].
5	no thank you, that is all i needed!	you're welcome. have a great day!	you're welcome. have a great day!	you're welcome. have a great day!	you're welcome, have a great day!

Table 2: An example where Agent 4 was picked as the best performing agent in a conversation.

Turn num	User turn	Agent 1 response	Agent 2 response	Agent 3 response	Agent 4 response
1	can you help me find a restaurant that serves african food with a moderate price range please	there are no [food] restaurants in the [pricerange] price range. would you like to try a different type of food?	there are no [food] restaurants in the [pricerange] price range. would you like to try a different type of food?	there are no [food] restaurants in the [pricerange] price range. would you like to try a different type of food?	[name] is a [food] restaurant in the [area]. would you like to try it?
2	yes how about asian food in the same price range?	there are [choice] [food] restaurants in the [area] and [choice] in the [area]. do you have a preference?	there are [choice] [food] restaurants in the [pricerange] price range. do you have a preference for area?	there are [choice] [pricerange] [food] restaurants in the [area]. do you have a price range in mind?	[name] is a [food] restaurant in the [pricerange] price range.
3	will you please book a table for 6 at 13:45 on friday?	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref]. is there anything else i can help you with?	booking was successful. the table will be reserved for 15 minutes. reference number is : [ref].	i was able to book you at [name] for [bookday] at [booktime]. your reference number is [ref]. is there anything else i can help you with?	i have made a reservation for [bookpeople] at [booktime] on [bookday] at [booktime]. the table will be reserved for 15 minutes. the reference number is [ref]. is there anything else i can help you with?
4	that's all i needed. thanks a lot for the help!	you're welcome. have a great day!	you're welcome. have a great day!	you're welcome. have a great day!	you're welcome. have a great day!

Table 3: An example where Agent 4 was picked as the best performing agent in a conversation.

found that agents which incorporate context into prompts were preferred over agents with vanilla prefix-tuning.

Limitations

While our work explores a new technique of contextual dynamic prompts for response generation, we carried out our experiments on a dataset which is in the English language. A potential limitation of this work would be the transfer of our findings on an English dataset to a multi-lingual dataset or a mono-lingual dataset on a language other than English. We plan to address this in our future work and also request the help of the research community in doing so.

Acknowledgements

We would like to thank the anonymous reviewers for their feedback on this work. We also thank the AWS Console/UXP2 Science team at AWS AI Labs for discussions around this work. Lastly, we'd like to thank the human annotators for their efforts.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Donghoon Ham, Jeong-Gwan Lee, Youngsoo Jang, and Kee-Eung Kim. 2020. End-to-end neural pipeline for goal-oriented dialogue systems using gpt-2. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 583–592.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, Jian Sun, and Yongbin Li. 2022. Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10749–10757.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. *CoRR*, abs/1902.00751.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4937–4949, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020a. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 441–459, Online. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020b. Mintl: Minimalist transfer learning for task-oriented dialogue systems. *arXiv preprint arXiv:2009.12005*.

Qi Liu, Lei Yu, Laura Rimell, and Phil Blunsom. 2021. Pretraining the noisy channel model for task-oriented dialogue. *Transactions of the Association for Computational Linguistics*, 9:657–674.

Shikib Mehri, Tejas Srinivasan, and Maxine Eskenazi. 2019a. Structured fusion networks for dialog. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 165–177, Stockholm, Sweden. Association for Computational Linguistics.

Shikib Mehri, Tejas Srinivasan, and Maxine Eskenazi. 2019b. Structured fusion networks for dialog. *arXiv preprint arXiv:1907.10016*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Buildingtask bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *https://openai.com/blog/language-unsupervised/*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *CoRR*, abs/1705.08045.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2021. Multi-task pre-training for plug-and-play task-oriented dialogue system. *arXiv preprint arXiv:2109.14739*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14230–14238.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI, ACL 2020*, pages 109–117.

Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. 2022. Continual prompt tuning for dialog state tracking. *arXiv preprint arXiv:2203.06654*.

A Human Evaluation Task

We explored contextual dynamic prompting strategies for the response generation task using the MultiWOZ 2.2 (Budzianowski et al., 2018; Zang et al., 2020) dataset and noticed that the combined score that we obtained was significantly better than the baseline prefix-tuning method of response generation. To understand if the agents which incorporated contextual dynamic prompts did indeed provide a better conversational experience, we designed a small human evaluation task to test our hypothesis.

We picked a random subset of 10% of the conversations from the original MultiWOZ test data to perform this analysis. Once we obtained this random set, we ran our four model variants as described in Section 4 on the conversations to obtain system responses for each of them. We then presented the different agents' responses to the annotator as shown in Table 4 below. In order to avoid potential biases, we shuffled the order of the agents between our annotators i.e., Agent 1 for annotator *a* would not be Agent 1 for annotator *b*. We kept track of which agents corresponded to which of our four methods prior to distribution of data amongst the annotators.

The annotators were given instructions to read every turn of conversation and provide a number between 1 and 4 for the agent which they thought performed the best for that turn. If the annotators found that there was a tie, they could pick more than one agent as [agent_a, agent_b]. In addition to this instruction, annotators were asked to read the entire conversation and pick the agent which performed the best - once again with an option to pick multiple. Table 5 below shows an example annotation style for a single conversation spanning 6 turns. There is an annotation at every turn and a single annotation at the end of the conversation.

We tallied results and re-mapped all agents back to their methods and found that agents 3 and 4 as mentioned in Section 5.2 were preferred at the conversation level in a total of 53 of the 100 conversations while agents 1 and 2 were only preferred 10 conversations in the entire set of 100.

B Examples

In the table below, we provide another examples from the human annotation task where annotators picked agent 4 as the best performing agent for the conversation (we performed the re-mapping to fit the agent numbers in Section 5.2). Table 2 shown in the main paper indicates a scenario where Agent 4 was rightly able to ask the customer about their area of preference for dining, and also considers a booking time change from the user. Table 6 shows an example where agent 4 is able to take into consideration all the different constraints mentioned by the user in terms of slots in the response that it generates. Table 3 shows an example where the agent 4 is able to correctly respond to the user request for number of people to reserve the table for - which none of the other agents captured. All these examples showed that agent 4, which uses contextual dynamic prompts with dialog states was indeed able to provide smoother responses to the customer input compared to the other agents.

C Compute Infrastructure and Experimental settings used

All our experiments were carried out on public AWS p3 GPU instances which are backed by Tesla V100 GPU cores. All four variants of our models required 18-24 hours to reach convergence on these machines. Learning rates were varied based on performance of models on the validation split of our dataset.

Turn num	User turn	Agent 1 response	Agent 2 response	Agent 3 response	Agent 4 response
1					
2					
3					
4					
5					
6					
7					

Table 4: The format which is presented to annotators while performing turn-level and conversation-level annotation. The agents are shuffled between the annotators to avoid biasing them.

Turn num	Turn level	Conversation level
1	2	
2	[3,4]	
3	2	
4	3	
5	4	
6	[3,4]	3

Table 5: We asked annotators to provide two levels of annotation for each conversation - turn level and conversation level. Annotators were allowed to pick more than agent in case of a tie.

Turn num	User turn	Agent 1 response	Agent 2 response	Agent 3 response	Agent 4 response
1	i am looking for a train that departs from peterborough and goes to cambridge.	there are [choice] trains leaving from [departure] to [destination]. what day would you like to travel?	what day would you like to travel?	there are [choice] trains leaving [departure] from [departure] to [destination]. what day would you like to travel?	i have [choice] trains departing from [departure] to [destination]. what day would you like to travel?
2	we will be leaving on sunday afternoon.	[trainid] leaves [departure] at [leaveat] and arrives in [destination] at [arriveby]. would you like me to book that for you?	i have [choice] trains available. what time would you like to leave?	[trainid] leaves at [leaveat] and arrives in [destination] at [arriveby]. would you like me to book it for you?	[trainid] leaves [departure] at [leaveat] and arrives at [destination] at [arriveby]. would that work for you?
3	yes, it should leave after 20:15 and leave on monday, not sunday. please give me a price, travel time, and arrival time for any train.	[trainid] leaves at [leaveat] and arrives in [destination] at [arriveby]. the price is [price].	[trainid] leaves at [leaveat] and arrives at [arriveby]. would you like me to book that for you?	[trainid] leaves at [leaveat] and arrives at [arriveby]. the price is [price]. would you like me to book it for you?	[trainid] leaves [departure] at [leaveat] and arrives in [destination] at [arriveby]. the cost is [price] and the travel time is [duration].
4	yes please. what is the travel time? i'll also need a hotel in the expensive price range. can you help me with that as well?	the travel time is [duration]. is there anything else i can help you with?	the travel time is [duration].	the travel time is [duration]. is there anything else i can help you with?	the travel time is [duration]. is there anything else i can help you with?
5	yes, i'd like a 2 star rating and i'll need it to be in the east side of town.	i have [choice] [type] in the [area] with [stars] stars. do you have a preference?	there are [choice] [type] in the [area] that are [stars] stars. do you have a preference?	there are [choice] [type] in the [area] that are [pricerange] and [pricerange]. do you have a preference?	[name] is a [stars] star hotel in the [area]. would you like me to book it for you?
6	yes, i would like to book it for 3 people starting that same day for 4 nights.	booking was successful. reference number is : [ref]. is there anything else i can help you with?	booking was successful. reference number is : [ref].	booking was successful, the total fee is [price] payable at the station. reference number is : [ref].	booking was successful. the total fee is [price] payable at the station. reference number is : [ref].
7	that is all thank you goodbye.	you're welcome. have a great day!	you're welcome. have a great day.	you're welcome. have a great day.	you're welcome. have a great day.

Table 6: Another example where Agent 4 was picked as the best performing agent in a conversation.

Why Can't Discourse Parsing Generalize?
A Thorough Investigation of the Impact of Data Diversity

Yang Janet Liu and **Amir Zeldes**
Department of Linguistics
Georgetown University
`{yl879, amir.zeldes}@georgetown.edu`

Abstract

Recent advances in discourse parsing performance create the impression that, as in other NLP tasks, performance for high-resource languages such as English is finally becoming reliable. In this paper we demonstrate that this is not the case, and thoroughly investigate the impact of data diversity on RST parsing stability. We show that state-of-the-art architectures trained on the standard English newswire benchmark do not generalize well, even within the news domain. Using the two largest RST corpora of English with text from multiple genres, we quantify the impact of genre diversity in training data for achieving generalization to text types unseen during training. Our results show that a heterogeneous training regime is critical for stable and generalizable models, across parser architectures. We also provide error analyses of model outputs and out-of-domain performance. To our knowledge, this study is the first to fully evaluate cross-corpus RST parsing generalizability on complete trees, examine between-genre degradation within an RST corpus, and investigate the impact of genre diversity in training data composition.

1 Introduction

Discourse parsing is the task of identifying and classifying the coherence relations that hold between different parts of a text, such as recognizing that one sentence specifies the cause of events in another, or that a subordinate clause indicates the purpose of a main clause. In hierarchical frameworks, such as the Rhetorical Structure Theory (RST, Mann and Thompson 1988), parsers construct a labeled constituent tree of discourse units as shown in Figure 1. Such trees have numerous applications: retrieving specific relations (e.g. all CONCESSIONS made in any speech by some politician, e.g. unit 26-27 in Figure 1), or hierarchically finding the most central unit in an arbitrary span of text (unit 24 in Figure 1). They can also contribute to downstream tasks such as text generation (Maskharashvili et al., 2021) and summarization (e.g. Louis et al. 2010; Li et al. 2016; Xu et al. 2020; Hewett and Stede 2022) and to qualitative analysis and comparison of texts (Wan et al., 2019).[1]

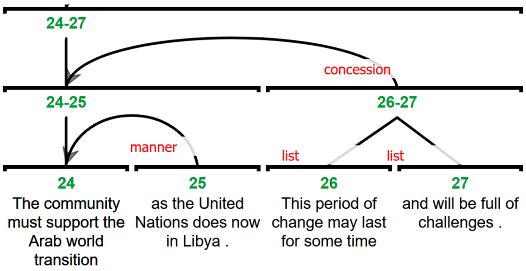


Figure 1: RST Fragment from GUM (Zeldes, 2017).

Recent advances in NLP have resulted in increasingly accurate systems for tasks such as part-of-speech tagging (Heinzerling and Strube, 2019) and dependency parsing (Mrini et al., 2020), which are now commonly applied with confidence to novel data, at least for high-resource languages such as English. At the same time, rising scores on the main standard English RST benchmark, the RST Discourse Treebank (RST-DT, Carlson et al. 2003), with data from the 1989 Wall Street Journal (WSJ), create the impression that discourse parsing too is becoming reliable, and by proxy, applicable to arbitrary text. This has resulted in use of automatic RST parsing to automatically generate large-scale datasets, such as MEGA-DT (Huber and Carenini, 2020) or AMALGUM (Gessler et al., 2020). However, there are some indications that this picture is too optimistic, with some studies revealing problems in RST-DT-based parsing once the target domain changes (Ferracane et al. 2019; Wang et al. 2019; Linscheid et al. 2021; Atwell et al. 2021, 2022; Nishida and Matsumoto 2022; Yu et al. 2022), and qualitative inspection of parser outputs casting doubt on their reliability.

---

[1] This applies to other discourse frameworks too, such as PDTB (Prasad et al., 2019) or SDRT (Asher and Lascarides, 2003), but we limit the scope of this paper to RST parsing.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3112–3130
May 2-6, 2023 ©2023 Association for Computational Linguistics

A problem in evaluating the stability of RST parsing for English is the scarcity of out-of-domain (OOD) data, as well as potential differences between the annotation schemes of different corpora. Recently, the availability of increasing amounts of non-newswire data, as well as non-WSJ news data, in the English GUM corpus (Zeldes, 2017) has made some detailed comparisons possible, which we explore below. Beyond evaluating OOD degradation, the existence of these large datasets for English RST allows us to test different joint training strategies to increase both the accuracy and generalizability of RST parsing for English.[2]

Our main goals here are: 1) to demonstrate the generalizability limitations of English RST parsing based on RST-DT and quantify them for users; 2) to explore reasons for generalizability issues, with a focus on the genre composition of training sets, pointing the way to the kind of data that robust discourse parsing requires; and 3) to promote multi-genre benchmarks for RST parsing based on our experimental results. Overall we find that diverse training data leads to better generalization on unseen genres regardless of model architecture.

2 Related Work

2.1 English RST Corpora

The RST Discourse Treebank (RST-DT, Carlson et al. 2003) is the standard English RST benchmark, with data from the WSJ section of the Penn Treebank (PTB, Marcus et al. 1993). Another human-annotated English RST corpus is the **G**eorgetown **U**niversity **M**ultilayer (GUM) corpus, which is freely available online and covers 12 written and spoken genres (Zeldes, 2017). GUM is continuously growing, with new data added in each version. For this paper, we used Version 8 of the corpus, which is described in Table 1 next to information about RST-DT (the current version of GUM, V9, was not yet available at submission time, and now contains 213 documents, 203K tokens and 26K EDUs, and the next version 10 is set to add 4 additional new genres).

	RST-DT	**GUM V8**
documents	385	193
train/dev/test	347/–[3]/38	145/24/24
tokens	203,352	180,851
EDUs	21,789	23,107
relation instances	20,163	21,903
relation labels	78 / 17 classes	32 / 15 classes
genres	1[4]	12

Table 1: Overview of the English RST Corpora.

As Table 1 shows, although GUM V8 is smaller than RST-DT in number of tokens and documents, it contains more instances of elementary discourse units (EDUs) and relations, since newswire units from RST-DT are longer on average. EDU segmentation guidelines are identical for the two corpora, and the label set is very similar, with both corpora using the SAME-UNIT pseudo-relation for discontinuous EDUs. GUM's 12 genres make it possible to conduct experiments to investigate parsing generalizability (see Table 12 in Appendix B for their exact size breakdowns). Virtually all work on RST constituent parsing uses only the collapsed coarse relation classes (i.e. 17 labels for RST-DT); we will follow this practice for the evaluation below, but will refer to some issues relating to fine-grained labels in our analysis as well. In addition, we conducted an analysis of **s**atellite-**n**uclearity patterns in the two corpora, confirming that they are quite similar: NS dominates with 77.7% in RST-DT, similar to 74.8% in GUM news; interestingly, proportions in GUM overall are somewhat lower (70.1%).

There are also other corpora annotated in RST dependencies, such as SciDTB (Yang and Li, 2018) and full RST constituent treebanks in other languages (see Zeldes et al. 2021 for an overview); since we focus on hierarchical English RST constituent parsing, we will use the corpora in Table 1 for the experiments below: they differ in content, vocabulary, domains, and more importantly, underlying communicative intents mirroring discourse relations, all of which are the backbone of our experiments to investigate English RST parsing generalizability as a function of training data.

2.2 RST Discourse Parsers

Several approaches have been proposed for RST parsing, primarily distinguished by a BOTTOM-UP vs. TOP-DOWN approach, and the algorithm used (often a neural shift-reduce architecture). Table 2 compares recent high scoring parsers on three metrics: **S**pan (whether subtrees span the right EDUs), **N**uclearity (whether edges point the right way), and **R**elation (whether labels are correct). Following

---

[2] Though we focus on English in this paper, understanding the size and nature of data needed in English will hopefully shed light on what other languages may require.

[3] There is no established dev partition for RST-DT.

[4] Previous work has identified 4 sub-genres in the WSJ data (Webber, 2009), but these are very unevenly distributed, and 29 RST-DT documents were not included in that analysis.

	S	N	R	Neural (Y/N)	Model Architectures	Additional Features / Resources	Pre-trained LM
	BOTTOM-UP						
Ji and Eisenstein (2014)*	64.1	54.2	46.8	N	transition-based + SVMs	lexical, dependency, graphical features	–
Yu et al. (2018)$^{\diamondsuit}$	71.4	60.3	49.2	Y	transition-based + Bi-LSTM; encoder-decoder + dynamic oracle	implicit syntactic features	–
Guz and Carenini (2020)$^{\spadesuit}$	**76.5**	65.9	**54.8**	Y	transition-based + MLPs	organizational features (Wang et al., 2017)	SpanBERT-base
Yu et al. (2022)$^{\clubsuit}$	76.4	**66.1**	54.5	Y	Yu et al. (2018) + 2 pre-training tasks: Next EDU Prediction; Discourse Marker Prediction	–	XLNet
	TOP-DOWN						
Koto et al. (2021)$^{\clubsuit}$	73.1	62.3	51.5	Y	LSTM + dynamic oracle	sentence and paragraph boundaries	–
Nguyen et al. (2021)$^{\clubsuit}$	74.3	64.3	51.6	Y	Bi-LSTM encoder + unidirectional LSTM decoder	–	XLNet-base
Zhang et al. (2021)$^{\clubsuit}$	76.3	**65.5**	**55.6**	Y	split-point encoding and ranking + adversarial learning	–	XLNet-base
Liu et al. (2021)$^{\spadesuit}$	**76.5**	65.2	54.2	Y	pointer-network decoder + depth-first span splitting + biaffine classifiers	–	XLM-RoBERTa-base

Table 2: Micro-averaged Parser Scores on RST-DT with Gold Segmentation. * = scores from Morey et al. (2017). $^{\diamondsuit}$ = converted Parseval scores for Yu et al. (2018)'s parser reported by Koto et al. (2021). $^{\spadesuit}$ = 5 run average Original Parseval replication of respective work. $^{\clubsuit}$ = scores from the original paper (not necessarily averaged scores).

Morey et al. (2017), we use the more stringent original Parseval procedure on binary trees.

As Table 2 shows, BOTTOM-UP and TOP-DOWN approaches are currently very close, and all state-of-the-art (SOTA) systems rely on transformer word embeddings as the primary input. To rule out an impact of parser architecture, we test both the BOTTOM-UP from Guz and Carenini (2020), using their best `SpanBERT-NoCoref` setting, and the TOP-DOWN from Liu et al. (2021) as their code is public and both have near-SOTA S and N metrics, which are more robust to differences between label inventories across corpora (relevant hyperparameters and `dev` performance are provided in Appendix C).

2.3 Cross-Genre Variation in RST

Previous work suggests RST trees vary widely across text types,[5] in relation distributions (Taboada and Lavid, 2003; Zeldes, 2018), ways they are signaled (e.g. Eggins and Martin 1997; Liu 2019; Demberg et al. 2019), and variation across languages (Cao et al. 2018; Stede and Neumann 2014; Iruskieta et al. 2013; Redeker et al. 2012; da Cunha et al. 2011; Pisarevskaya et al. 2017).

These differences suggest OOD parsing will be challenging: For example, Taboada and Lavid (2003) showed that in appointment-scheduling conversations, CONCESSION, CONDITION, CAUSE, and RESULT were disproportionately frequent due to the nature of the data, correlating with stages of a conversation (i.e. Opening, Task Performance, Closing): e.g. more RESTATEMENT, EVALUATION, and SUMMARY in the Closing stage. Similarly, Zeldes (2018) showed MOTIVATION often encodes how-to guides influencing one's willingness to act, which almost never occurs in the objective style of news. We therefore expect parsers trained on a single domain to generalize poorly to novel ones.

2.4 Generalizability in RST Parsing

Although some first studies are beginning to appear, generalizability in full RST parsing remains understudied. Three recent papers include some cross-dataset numbers, but not a systematic analysis of full RST constituent parser generalizability. Atwell et al. (2021) examined the subtask of relation classification between PDTB (Prasad et al., 2019) and RST annotated data, and included F1 scores for identifying relations in GUM V5 using the RST-DT training data for relations that appear in both datasets (however, GUM V5's EDU segmentation differed strongly from RST-DT and the parser in the paper was an older non-neural one).

Yu et al. (2022) evaluate a neural parser on RST-DT and report scores on 11/12 genres in GUM V7 (same segmentation as RST-DT, but only 25 relations and less data than V8); however because their focus is not on cross-genre generalization, they do not report micro-averaged scores on the whole GUM corpus, do not evaluate training on GUM or joint training, and do not test held-out genres within GUM. Finally, Atwell et al. (2022) used GUM V6 (with RST-DT's segmentation and 8 genres) to identify relation types between pairs of provided argument spans, similar to PDTB-style shallow discourse parsing in the CoNLL'16 shared task (Xue et al., 2016) and recent DISRPT shared task on relation classification (Zeldes et al., 2021).

This study is therefore the first to fully evaluate cross-genre RST parsing generalizability on complete trees in datasets with the same EDU segmentation. To our knowledge, no previous work has tested an OOD-trained parser on RST-DT, at-

---

[5] We use "text type", "genre", and "domain" interchangeably here to mean different kinds of data, though they are not exactly the same (e.g. see Biber 1988 and Lee 2001).

tempted joint training on both corpora, evaluated between-genre degradation within an RST corpus, or the impact of training data genre composition.

3 Experiments

In this section we conduct four experiments. First, we want to find out how compatible the two English RST corpora are, and how well they generalize to each other's domains (§3.1). Second, we want to test whether we can incorporate training data from GUM in order to exceed SOTA performance on RST-DT, with one of three approaches: joint training, model stacking, and pretraining (§3.2). Third, we would like to know whether generalizability issues are mainly due to differences between corpora (details of annotation guidelines or common practices) or the nature of the texts' genres and domains themselves (§3.3). We test this by leaving out different subsets of GUM genres from training, and quantify OOD degradation for each genre, which also informs users of SOTA parsers of what degradation they can expect on genres 'in the wild'. The final experiment (§3.4) tests whether, given a fixed training set size, having relatively many 'small' genres leads to better OOD stability than having few genres with more material each, when the target is a totally disjoint set of genres. The code, trained and finetuned models, and predicted GUM parses are available at `https://github.com/janetlauyeung/crossGENRE4RST`.

3.1 Cross-Corpus Generalization

In this section we hypothesize that, since GUM contains many genres, models trained on it will degrade less when testing on RST-DT than in the opposite scenario. This is non-trivial, since the corpora are very close in total size, and it is possible that diverse training genres with little data each will prevent effective learning for any genre. We train the two parsers identified in §2.2 from Guz and Carenini (2020) (BOTTOM-UP) and Liu et al. (2021) (TOP-DOWN) on the `train` partition of each dataset, and report scores on the `test` set (since RST-DT has no dev partition, we follow previous work in using 10% of training data stratified by the number of EDUs in each document as a dev set, which remains the same in the training).

A possible limitation of our experiment is annotation differences between the corpora, especially for their relation inventories, which are similar, but not identical. We therefore omit R scores in Tables 3 and 4, though we will return to relations in more detail below. For S and N, however, we expect differences to be small: both corpora use the same segmentation guidelines, and the same guideline of subordinating less prominent satellite units to more prominent nuclei using tests such as deletability and subjective prominence. Although EDU segmentation is not our focus, we can confirm the compatibility of segmentation guidelines by cross-testing the current SOTA EDU segmenter from Gessler et al. (2021). The results show fairly modest degradation training on RST-DT and testing on GUM, from 94.9 to 89.9, while the opposite direction shows a 92.9 to 91 drop (see Table 11 in Appendix A for full numbers). We speculate that segmentation generalizes well due to the syntactic, and hence less sparse and domain-specific nature of the task, which resembles clause boundary detection.

For cross-corpus S and N scores, both parsers show a very significant degradation when training on RST-DT to parse OOD data from GUM, as shown in Tables 3 and 4. For instance, there is a degradation of $\sim$−11 points for S and $\sim$−16 for N using the the BOTTOM-UP architecture. We were also curious whether this applies to GUM's news genre too: indeed, testing only on GUM *news* reduces degradation by about 50% using the BOTTOM-UP parser, but still shows a substantial performance hit, even if the model is trained exclusively on news (albeit from much older, WSJ news data). By contrast, the GUM-trained model actually scores better on RST-DT than on GUM, with minor improvements of +2.8 on S and +0.4 on N using the BOTTOM-UP parser. The same degradation pattern is observed in the TOP-DOWN parser performance: substantial degradation overall, and worse when training on news and testing other genres.[6]

train	test	S	N	R
RST-DT	RST-DT	76.5	65.9	54.8
	GUM	65.3 (-11.2)	49.5 (-16.4)	–
	GUM *news*	71.0 (-5.5)	57.5 (-8.4)	–
GUM	GUM	69.9	57.0	48.5
	RST-DT	72.7 (+2.8)	57.4 (+0.4)	–
	GUM *news*	71.6	58.5	49.5

Table 3: Cross-Corpus Results (5 run average) of the BOTTOM-UP Parser from Guz and Carenini (2020).

---

[6] We further analyzed left vs. right-branching trees during testing. In both corpora, NS outperformed SN: RST-DT in-domain scores are F1=82.6% vs. 73.4% for NS vs. SN; GUM V8 has 81.6% vs. 72.4%, meaning SN transitions are 'easier'.

train	test	S	N	R
RST-DT	RST-DT	76.5	65.2	54.2
	GUM	66.2 (-10.3)	50.8 (-14.4)	–
	GUM *news*	67.9 (-8.6)	55.8 (-9.4)	–
GUM	GUM	68.6	54.9	46.1
	RST-DT	71.1 (+2.5)	55.9 (+1.0)	–
	GUM *news*	73.4	63.3	57.2

Table 4: Cross-Corpus Results (5 run average) of the TOP-DOWN Parser from Liu et al. (2021).

We interpret this result to mean that unlike in EDU segmentation, where results are close in both directions, genre composition of the `train` and `test` data plays a crucial role in the generalizability of RST constituent parsing, regardless of parser architecture. Thus, we opt for the BOTTOM-UP parser for the experiments presented in the rest of the paper as it outperformed the TOP-DOWN parser on both corpora.

To be clear, the results do not suggest that training on GUM is a way of achieving top performance on RST-DT: the GUM model's scores are still almost 4 and 8 points below the RST-DT trained model for S and N. However, it seems that RST-DT news data is less surprising for the GUM model which has already seen some news, and in sum, RST-DT data appears to be a comparatively 'easy' target given the broad genre inventory that the GUM model is trained to tackle.

These results also confirm that training on multiple genres, each with comparatively fewer documents, can lead to good performance with only minor degradation on the very narrow WSJ domain from RST-DT. While we think this result by itself is important and suggests that RST parsing work should devote more attention to multi-genre corpora as benchmarks, it leaves a question open: Can we combine data from both corpora to boost English RST parsing performance on RST-DT?

3.2 Joint Training

If RST parsing generalizes across domains, it may be possible to see gains through joint training. For this we compare three approaches: naive concatenation, model stacking, and pretraining, which we evaluate on the RST-DT benchmark.

For concatenation (CONCAT) we must map relations across corpora. As in previous work, we target coarse relation classes, most of which are identical in both corpora. Exceptions include GUM's PHATIC relation used for dysfluencies and back-channeling (due to conversational data), GUM's PREPARATION, and the mapping between RST-DT's TEMPORAL and COMPARISON classes, which map differently onto GUM's CIRCUMSTANCE and SEQUENCE, or GUM's ANTITHESIS, depending on the precise senses. Although the number of relation instances affected by these mismatches is modest, it is not negligible (13.3%), and we do not expect this approach to outperform in-domain training, and mainly report on it for completeness. Our mapping is given in Appendix D.

For model stacking, we test three variants: 1) FLAIR-LABEL: train an LSTM using FLAIR (Akbik et al., 2019) to predict EDU dependency labels: the LSTM receives text for three-EDU chunks, set apart by separators, and predicts the middle EDU's label in the GUM corpus scheme (using the RST dependency conversion from Li et al. 2014). We then train the parser on RST-DT with predicted GUM labels for each EDU as an additional feature encoded as a dense embedding, requiring no relation mapping. 2) SR-LABEL: train a full shift-reduce parser on GUM, generate predictions for RST-DT in the GUM scheme, and collapse these into dependencies to create the same kind of features; this approach gives the GUM classifier more access to global context than the previous LSTM's EDU triples, but may be more vulnerable to sparseness. 3) SR-GRAPH: because it is possible that incompatibility of RST-DT and GUM labels may cause confusion, we also attempted using the same parser as in the previous approach, but featurizing each EDU's predicted dependency attachment direction, and EDU distance to the parent EDU, instead of the label itself. Finally, we test a simple pretraining approach, in which we finetune the underlying SpanBERT model (Joshi et al., 2020) on full parsing of the GUM corpus, then load the fine-tuned SpanBERT, and train again on RST-DT (SR-FT).

Table 5 shows that SR-FT achieved the best performance compared to the other approaches and is on par with the pure in-domain training on RST-DT for S and R compared to previous work. There is a minor but stable gain on average (65.9→66.2) for N in SR-FT, which was verified by rerunning the experiment, as well as the selected SOTA system from Guz and Carenini (2020), 5 times. The remaining scenarios are virtually equivalent to training on RST-DT alone, suggesting that added features are more distracting than helpful.

This result is somewhat surprising given that scores are not very high, and there should still be

	S	N	R	*architecture*
Zhang et al. (2021)*	76.3	65.5	**55.6**	TOP-DOWN
Liu et al. (2021)$^{\diamondsuit}$	**76.5**	65.2	54.2	TOP-DOWN
Guz and Carenini (2020)$^{\diamondsuit}$	**76.5**	**65.9**	54.8	BOTTOM-UP
this paper (CONCAT)$^{\spadesuit}$	75.9	64.8	54.1	BOTTOM-UP
this paper (FLAIR-LABEL)$^{\spadesuit}$	75.8	65.6	55.3	
this paper (SR-LABEL)$^{\spadesuit}$	76.2	66.0	55.3	
this paper (SR-GRAPH)$^{\spadesuit}$	75.8	65.5	54.7	
this paper (SR-FT)$^{\diamondsuit}$	**76.3**	**66.2**	**55.5**	
Human (Morey et al., 2017)	78.7	66.8	57.1	–

Table 5: Joint Training Performance on RST-DT. * = original paper score. $^{\diamondsuit}$ = 5 run avg.; $^{\spadesuit}$ = 3 run avg.

headroom for improvement; however, we suspect some of the missing information responsible for errors may relate to global structure and pragmatic understanding which cannot easily be compensated for by adding more genres with potentially disjoint vocabulary. A further surprising result is that CONCAT is not much worse than the base system, suggesting that most of the score comes from obvious cases (e.g. relative and infinitival clauses) which do not differ substantially across corpora or genres, and were already learned without the added data. Similar results have been found for other NLP tasks where adding a second dataset for joint training creates a 'break-even' effect: the benefit of more data helps about as much as the disparate domains harm within-corpus performance, e.g. for dependency parsing in Hebrew (Zeldes et al., 2022) and English (Zeldes and Schneider, 2023), English coreference resolution (Zhu et al., 2021), and very recent similar results for RST parsing for Chinese (Peng et al., 2022).

3.3 OOD Multi-Genre Degradation

In this section we explore our next question: How badly will a multi-genre trained model degrade on unseen genres, when the annotation scheme remains identical? This question is important for applied, methodological, and ethical reasons. From the practical perspective, while we already know that training only on news leads to severe degradation, we want to inform users of discourse parsing about expected performance on unseen domains if training data is already domain-rich. Methodologically, we want to test the benefit of adding multiple genres and weigh the differences between a few-genre corpus design and a many-genres design, given a fixed total data size capacity for dataset creation. Finally from an ethical perspective, OOD degradation has real life implications for less common types of data, whether they come from underrepresented genres, communicative situations, or speaker demographics (Mengesha et al., 2021). While we cannot fix these problems without more data, we can point them out and increase awareness of skewed data biases at the discourse level.

To explore OOD degradation, we conducted 10 experiments, comparing the normal genre-balanced scenario (GUM `test`) with testing on each genre when it is not in 'train' (ONE-VS-ALL or OVA). For consistency, we test OVA for the 8 roughly equal sized non-growing genres in GUM. Since data for the smaller 4 growing genres may be less reliable and non-comparable, we separately report scores for training on all 8 large genres (ALL-LARGE), tested on each of the four growing genres.

In the last scenario in particular, we would like to see whether small test genres, for which we cannot obtain enough training data, perform better if a training genre exists which may offer a near substitute. For instance, the small *textbook* and *speech* genres have structural organization and formal language similar to *academic* and *news* respectively, and there is some overlap between *speech* vocabulary and *interviews* with politicians. By contrast, *vlog* and *conversation* are highly informal and colloquial, perhaps closest to the *reddit* or *interview* genres, but still likely more challenging. For reproducibility, exact training data compositions are given in Table 17 in Appendix E.

Table 6 shows the results, with the same 4 documents from each held-out genre used to evaluate each OVA model (these are not included in any training set for consistency). The *degradation* column shows that the parser suffers when a genre is removed from training across the board, except for *news* and the Span level of *reddit*; *bio* and *how-to* genres suffer the most: qualitative inspection reveals that this is due to frequent errors on ORGANIZATION and EXPLANATION, which tend to lack explicit cues and require understanding of macrostructure and global topic or pragmatic reasoning to interpret. It is likely that lack of a similar substitute to these genres contributes to the degradation.

By contrast, degradation on *interview* is low (perhaps covered by *conversation* and *news* in training), and the inverse result for *news* suggests that collecting more news data may not be a priority. The minimal degradation for *reddit* is surprising, and is likely due to more instances of explicitly signaled and 'easy' relations such as PURPOSE (e.g. *in order to*) and CONTINGENCY (e.g. *if*), but we note that ordinary scores on *reddit* are low to begin with (second worst after *fiction*, which degrades more). The bottom of the table shows smaller genres suffering, except for a minor gain in *speech* at the Span level. It is unsurprising that *conversation* degrades the most, especially at the R level, since it contains challenging phenomena absent in other genres, such as back-channeling, dysfluencies, and abrupt topic changes. This is mirrored in bad performance on the ORGANIZATION and TOPIC relation classes (see more detailed analysis in §4).

	GUM test			**ova**			*degradation*		
non-growing	**S**	**N**	**R**	**S**	**N**	**R**	**S**	**N**	**R**
academic	77.0	68.5	59.8	75.2	66.2	55.7	1.7	2.3	4.1
bio	70.4	58.2	51.2	68.8	53.9	43.2	1.6	4.3	**8.0**
fiction	66.3	53.1	43.7	64.5	50.1	42.1	1.8	3.0	1.7
interview	73.3	59.0	50.9	73.0	56.7	49.7	0.3	2.2	1.2
news	71.7	58.4	49.1	72.2	59.2	51.3	-0.5	-0.8	-2.2
reddit	66.0	52.3	44.2	66.6	51.9	43.3	0.6	0.4	0.8
voyage	78.3	62.1	51.8	77.4	59.7	49.3	0.9	2.4	2.4
how-to	76.5	63.6	54.6	67.1	54.3	44.8	**9.3**	**9.3**	**9.9**
	GUM test			**ALL-LARGE**			*degradation*		
growing	**S**	**N**	**R**	**S**	**N**	**R**	**S**	**N**	**R**
conversation	45.4	34.5	26.7	42.7	31.4	21.8	2.7	3.1	**4.9**
speech	76.0	64.4	55.2	76.4	62.9	54.8	-0.4	1.5	0.4
textbook	77.4	66.8	57.3	76.2	64.3	54.5	1.2	2.6	2.9
vlog	64.8	49.0	42.8	63.3	49.0	40.4	1.5	0.0	2.5

Table 6: Per Genre Scores for GUM `test` vs. the OVA or ALL-LARGE Experiments (3 run average).

3.4 Genre Variety in a Fixed-Size Sample

While §3.3 shows how genres differ in degradation, it falls short of proving that genre diversity promotes generalization when all other things are equal, since train sets for each genre are not identical in size. Our final experiment addresses this: ideally, we want to compare scores on a fixed OOD test set for equal-sized training corpora, divided into fewer or more genres. Although we might expect more genres to be helpful for generalization, this is not trivial or inevitable: If there are not enough recurring examples of infrequent phenomena, because data is so diverse, learning might fail due to sparseness; that is, more genres could be distracting rather than helpful in a meaningful way, which could hurt performance.

Because genres in GUM are small (∼2K EDUs) and we want a critical mass of 5K+ EDUs for reasonable parser performance, our combinatory options are limited to 3+ training genres. The parser can only be trained on complete documents (we cannot select *n* EDUs from each genre – genres must comprise coherent texts), i.e. we must find permutations of the data which total roughly the same number of training instances, but with different genres. Table 7 gives details on the 3 best training cohorts based on these criteria.

ID	**genres**	**docs**	**EDUs**	**ID**	**genres**	**docs**	**EDUs**
C1	*academic*	18	1,970	C3	*academic*	9	1,004
	bio	19	1,981		*bio*	9	930
	news	23	1,760		*news*	10	635
	total	60	5,711				
C2	*fiction*	15	1,941		*fiction*	8	1,027
	interview	15	1,931		*interview*	8	1,199
	how-to	15	1,840		*how-to*	8	917
	total	45	5,712		**total**	52	5,712

Table 7: Composition of 3 Fixed-Size Training Cohorts with Different Genre Contents.

As Table 7 shows, the greatest care was taken to ensure that the cohorts sum up to almost exactly as many training instances (∼5,712 EDUs), at the price of somewhat diverse amounts of EDUs and documents per genre. If we cannot control all factors, we prefer to constrain the amount of shift-reduce operations learned, in order to prevent any alternative explanation in which total size effects outweigh genre diversity. Additionally, we assume that EDU count per genre will vary in any multi-genre corpus due to complete document constraints; corpus developers are more likely to target a total fixed size budget. If having too many small genres is harmful, we expect cohort 3 (C3) to perform worst; by contrast, if diversity is helpful, C3 should perform best. In either case, we compare how the S/N/R metrics behave in each training regime.

Table 8 shows 5-run averages on each OOD genre, as well as total micro- and across-genres macro-averaged performance. The 3-genre cohorts, C1 and C2, perform similarly overall, though C2 outperforms C1 on test data (which is held constant for all cohorts). R scores are very close on macro-average, but C1 is particularly bad on *conversation*, where C2 is tied with C3 on R (possibly because both have *interview* data). Conversely, C1 is better on textbooks and travel guides (*voyage*) than C2 (possibly because C1 has *academic*, which predicts textbooks reasonably well, and bios – a descriptive informational genre, something missing from C2).

Overall best performance is obtained by C3, with 6 training genres, but the least data in each. Although C3 loses to C2 in the dialogue-oriented *conversation* and *reddit* genres, degradation is very modest and does not affect all metrics (e.g. C3 has a better N score on *reddit*). C3 also outperforms C1 on *academic* and *voyage*, giving the best performance on both. The average gain for C3 across all

	C1			C2			C3			C3−C1			C3−C2			mean_C3_gain		
test	**S**	**N**	**R**	**S**	**N**	**R**	**S**	**N**	**R**	**S**	**N**	**R**	**S**	**N**	**R**	**S**	**N**	**R**
conversation	34.8	23.4	13.9	40.3	27.9	18.0	37.9	26.4	18.0	3.0	3.0	4.1	-2.5	-1.5	0.0	0.3	0.7	2.0
reddit	60.3	45.3	36.0	63.5	46.9	37.6	61.8	47.6	37.3	1.5	2.3	1.4	-1.7	0.7	-0.3	-0.1	1.5	0.6
speech	72.5	58.2	46.9	72.6	59.3	47.7	71.6	57.1	48.0	-0.9	-1.1	1.1	-1.0	-2.1	0.3	-0.9	-1.6	0.7
textbook	73.6	59.0	48.9	70.9	55.0	45.6	74.0	60.5	51.4	0.5	1.5	2.5	3.1	5.5	5.9	1.8	3.5	4.2
vlog	57.8	41.3	35.0	58.8	44.5	35.3	57.7	43.4	34.8	-0.1	2.1	-0.2	-1.1	-1.1	-0.5	-0.6	0.5	-0.3
voyage	76.6	58.1	47.5	76.5	57.4	46.4	78.0	59.1	50.2	1.5	1.0	2.7	1.6	1.7	3.8	1.5	1.4	3.3
macro_avg	62.6	47.6	38.0	**63.8**	48.5	38.4	63.5	**49.0**	**40.0**	0.9	1.5	1.9	-0.3	0.5	1.5	0.3	1.0	1.7
micro_avg	58.7	44.2	34.8	**60.5**	45.7	35.7	59.8	**45.9**	**36.9**	1.1	1.7	2.1	-0.6	0.2	1.2	0.2	1.0	1.6

Table 8: Performance of 3 Fixed-Size Train Cohorts with Different Genre Contents (5 run average).

metrics (3-metric average) is around +1%—taking individual scores from each of five runs, and each genre test set as a data-point, this improvement is significant at $p < 0.05$. For individual metrics, we see $\sim$+1.7% on R, +1% on N, and just under +0.3% for S (significant at $p < 0.05$ except for S).

Although all scores are rather low due to the small corpus sizes (about ¼ of GUM), they suggest that more training genres with smaller portions each promotes OOD generalization, though not by a lot. It is an open question whether this gap would increase or decrease with corpus size: on the one hand, more data would allow for more lexical diversity even with few genres. On the other, it is likely that scores in small data are driven by easy to learn cases (e.g. relative clauses as ELABORATION, or PURPOSE infinitives), which stand to gain less from diversity. If more data means models will tackle more sparse phenomena, then genre diversity should matter *more* for OOD material as the training set grows. To an extent, results in §3.1 showing worse generalization from the large but homogeneous RST-DT to GUM seem to support this hypothesis. We interpret the present results to mean that development of more diverse multi-genre data should take priority over building up material in existing genres to promote generalizable parsing.

4 Error Analysis

Due to space, we limit our error analysis to examining dependency conversions of gold vs. predicted trees, which allow us to break down OOD errors by coarse relation class in Table 9 from all test sets in §3.3. We select models with scores closest to average run scores. It is clear that even in the more modest degradation within GUM, over half of relation classes have $< 50\%$ accuracy, with the document ROOT being the hardest to identify—i.e. the Central Discourse Unit (CDU)—followed by RESTATEMENT and EVALUATION, which require reasoning over many EDUs and lack consistent overt signals, as opposed to relations such as ATTRIBUTION (marked by speech verbs), PURPOSE (marked by purpose infinitives or *in order to*), and CONTINGENCY (usually *if*); interrupted clauses (SAME-UNIT) are easier as well. For CDU identification (Iruskieta et al., 2016), which can benefit summarization or long-form QA systems, half of the genres (*academic*, *fiction*, *interview*, *voyage*, *how-to*, *vlog*) score 0%; the highest accuracy is only 50% (*bio*, *news*, *reddit* and *speech*). More alarmingly, in the cross-corpus setting (§3.1), an RST-DT trained model captures only a single GUM CDU correctly (ACC=0.042 vs. 0.375 for a GUM-trained model); scores on RST-DT are much higher: ACC=0.842 for SR-FT trained on RST-DT vs. 0.553 for a GUM-trained model.

gold coarse relation class	acc	gold coarse relation class	acc
ATTRIBUTION	0.875	CONTEXT	0.471
PURPOSE	0.861	ADVERSATIVE	0.467
SAME-UNIT	0.814	ORGANIZATION	0.463
CONTINGENCY	0.794	EXPLANATION	0.431
ELABORATION	0.666	CAUSAL	0.384
JOINT	0.654	EVALUATION	0.362
TOPIC	0.574	RESTATEMENT	0.308
MODE	0.504	ROOT	0.208

Table 9: OOD Accuracy by Relation Class in GUM.

genre	gold coarse relation class	abs(resid)
textbook	CONTEXT	3.64
speech	EXPLANATION	3.14
reddit	EXPLANATION	3.02
fiction	EVALUATION	2.59
bio	CAUSAL	2.26
vlog	CAUSAL	2.23
conversation	ORGANIZATION	2.14
voyage	CONTEXT	2.13
academic	ORGANIZATION	1.84
how-to	ORGANIZATION	1.62
news	EXPLANATION	1.38
interview	EVALUATION	0.89

Table 10: Max Absolute Error Residuals by Genre.

Errors are also skewed by genre: Table 10 gives the gold label most surprisingly associated with errors per genre, given global error distributions, based on absolute χ^2 residuals. While EVALUATION is problematic in *fiction* and *interview*, EX-

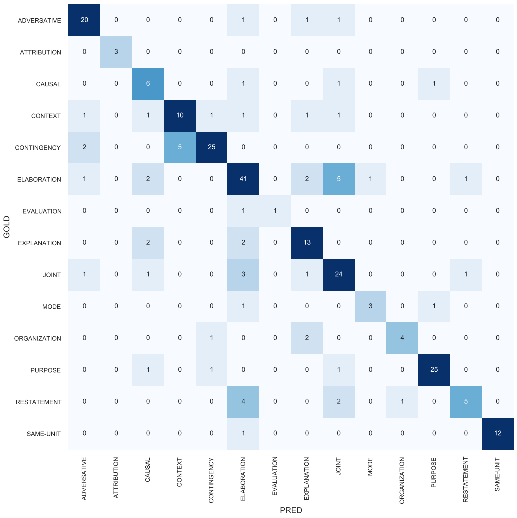
PLANATION and ORGANIZATION are surprisingly hard to predict in 3 genres each: the former is used e.g. in speeches and news for supporting evidence or justifications which are difficult to identify using lexical items, while the latter is used in conversations for phatic responses and back-channeling.

To further investigate genre-specific phenomena, we analyzed OOD parsing errors for *how-to guides*, which had the worst OOD performance in Table 6, using a model *not* trained on this genre from §3.3. Figure 2 shows a confusion matrix produced by converting the automatic parses into the RST dependency representation following Li et al. (2014). We focus on relation instances with correct attachment predictions, although we also discuss cases involving attachment errors as well (Figure 5h in Appendix G). We first observe in Figure 2 that there are five gold CONTINGENCY instances predicted to be CONTEXT: four out of five EDUs begin with a discourse marker (DM), 'once' or 'until', e.g.:

(1) [Don't use a joke] $^{\text{humans: CONTINGENCY}}_{\text{parser: CONTEXT}}$[**until** you're completely comfortable with it.]

According to The Penn Discourse Treebank (PDTB3, Prasad et al. 2019), these prototypically signal spatio-temporal circumstances: 94% of explicit instances with the DM 'once' are annotated as TEMPORAL.ASYNCHRONOUS while less than 5% of the instances (4/85) are annotated as CONTINGENCY.CONDITION. Similarly, 85% of explicit instances with the DM 'until' are annotated as TEMPORAL.ASYNCHRONOUS while 10% are CONTINGENCY.CONDITION. However, Liu (2019) found that these two DMs are frequently associated with CONTIGENCY-CONDITION in *how-to guides* because they are essentially never narrative and always part of an instruction which is uncommon in *news*. This suggests that although DMs are usually considered useful devices to identify certain relations, their usage differs across genres and is sometimes too ambiguous to form a reliable signal. We also observe that there are four gold RESTATEMENT confused with ELABORATION: three of these go back to 'definitional' RESTATEMENTS, which are likely promoted by the genre's descriptive and explanatory properties.

Figure 5h shows cases with attachment errors. One major error concerning global structure is seven gold ORGANIZATION instances being confused with JOINT. Two of the errors are unexpected given that there are strong graphical, structural, and semantic cues (see an example in Figure 3 in Appendix F). The remaining cases are harder to pin down: they are all imperatives, which are atypical for headings outside of *how-to guides*, but normal and frequent in instructional texts (see an example in Figure 4). For complete by-genre confusion matrices (with attachment errors) of the multi-genre degradation experiments, see Appendix G.



	ADVERSATIVE	ATTRIBUTION	CAUSAL	CONTEXT	CONTINGENCY	ELABORATION	EVALUATION	EXPLANATION	JOINT	MODE	ORGANIZATION	PURPOSE	RESTATEMENT	SAME-UNIT
ADVERSATIVE	20	0	0	0	0	1	0	1	1	0	0	0	0	0
ATTRIBUTION	0	3	0	0	0	0	0	0	0	0	0	0	0	0
CAUSAL	0	0	6	0	0	1	0	0	1	0	0	1	0	0
CONTEXT	1	0	1	10	1	1	0	1	1	0	0	0	0	0
CONTINGENCY	2	0	0	5	25	0	0	0	0	0	0	0	0	0
ELABORATION	1	0	2	0	0	41	0	2	5	1	0	0	1	0
EVALUATION	0	0	0	0	0	1	1	0	0	0	0	0	0	0
EXPLANATION	0	0	2	0	0	2	0	13	0	0	0	0	0	0
JOINT	1	0	1	0	0	3	0	1	24	0	0	0	1	0
MODE	0	0	0	0	0	1	0	0	0	3	0	1	0	0
ORGANIZATION	0	0	0	0	1	0	0	2	0	0	4	0	0	0
PURPOSE	0	0	1	0	1	0	0	0	1	0	0	25	0	0
RESTATEMENT	0	0	0	0	0	4	0	0	2	0	1	0	5	0
SAME-UNIT	0	0	0	0	0	1	0	0	0	0	0	0	0	12

Figure 2: *how-to* confusions (no attachment errors).

5 Conclusion

The analyses in this paper are meant to inform users of current parsers about what to expect from RST parsing in the wild. Through dozens of experimental runs we have shown a consistent picture: RST parsing has made impressive progress, but OOD degradation is still severe. Our results suggest prioritizing genre diversity in training data is crucial, not only to cover more text types as 'in domain', but also to increase performance on unseen text types. Rather than focusing only on the development of better models to beat the single genre RST-DT benchmark, robust RST parsing would be promoted most by creating more annotated data.

We also hope this paper will motivate researchers to prioritize multi-genre benchmarks and OOD settings for RST parsing, and to explore algorithms, representations and features which better capitalize on joint training and foster generalizability, including using data from other sources and theories of discourse analysis (cf. Braud et al. 2016). We see great challenges for tackling errors which implicate complex pragmatic inference or document level reasoning, but we are also optimistic that as more data becomes available, we will be able to learn more and better representations of discourse structure.

Limitations

One limitation is that the scope of the error analysis presented here is limited, primarily due to space reasons. Although we included full relation confusion matrices (with attachment errors) in Appendix G, the discussion is too limited to describe them in detail, and we encourage interested readers to explore and further compare gold and predicted trees (available in both `.rs3` and `.rsd` formats) on the multi-genre data, which we make available in the repository of this paper.

Another limitation concerns the number of runs of the experiments: some scores were averaged over three rather than five runs. This is due to the fact that SpanBERT-base (Joshi et al., 2020) is fairly large (110M parameters). Training each of our 20 models five times would consume very large amounts of GPU resources, which we feel is hard to justify both financially and environmentally. After completing this study we feel that three-run averaged scores should satisfy the need for reproducibility, though we did use five-run averages for establishing baseline scores and verifying particularly controversial results, such as the improvement on Nuclearity in the fine-tuning condition in §3.2 (i.e. SR-FT). Previous work providing this information reported three-run averaged scores such as Koto et al. (2021) while many papers did not include this information or mention whether the reported scores were averaged over multiple runs.

Ethics Statement

This work contributes to open source progress in RST discourse parsing, an area which has received less attention than some other NLP tasks, and which, at least in English, is currently suffering from a skewed focus towards the 'standard' language of 1990-era Wall Street Journal writing. Previous work has shown that NLP systems retain strong lexical biases mirroring both period and author demographics (Shah et al., 2020), meaning that if language technologies are not pushed to cover diverse data types robustly, they will inevitably perform more poorly on 'non-standard' data, with possible discriminatory effects on underrepresented populations ranging from the political (think opinion mining social media to guide policy) to financial (e.g. higher/lower search hit rates for YouTube videos by small businesses). By promoting higher quality treatments of diverse language samples in this study covering Reddit, YouTube vlogs, and demographically diverse conversations, we hope to help level the playing field across text-types, demographics, and domains.

We recognize that NLP research has a computing cost and carbon footprint, which motivates us to release the trained models in this work (preventing the need to retrain similar models), and to avoid extensive hyperparameter optimization which may not generalize to applications in the wild. Specific model configurations such as hyperparameters and validation performance for the reported `test` results of RST-DT and GUM are detailed in Appendix C.

Finally, we also recognize that NLP tools can be used to do harm. However, we expect that the type of analysis promoted here will do more good than harm by steering tool development away from adhering closely to outdated and narrow-domain data, which this work aims to broaden. Given that discourse parsers already exist, we view the push to reduce topical and authorial bias, as well as the public release of more resources, as net positives.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press, Cambridge.

Katherine Atwell, Junyi Jessy Li, and Malihe Alikhani. 2021. Where are we in discourse relation recognition? In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 314–325, Singapore and Online. Association for Computational Linguistics.

Katherine Atwell, Anthony Sicilia, Seong Jae Hwang, and Malihe Alikhani. 2022. The change that matters in discourse parsing: Estimating the impact of domain shift on parser error. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 824–845, Dublin, Ireland. Association for Computational Linguistics.

Douglas Biber. 1988. *Variation across Speech and Writing*. Cambridge University Press.

Chloé Braud, Barbara Plank, and Anders Søgaard. 2016. Multi-view and multi-task training of RST discourse parsers. In *Proceedings of COLING 2016, the 26th*

International Conference on Computational Linguistics: Technical Papers, pages 1903–1913, Osaka, Japan. The COLING 2016 Organizing Committee.

Shuyuan Cao, Iria da Cunha, and Mikel Iruskieta. 2018. The RST Spanish-Chinese treebank. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 156–166, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory. In *Current and New Directions in Discourse and Dialogue*, Text, Speech and Language Technology 22, pages 85–112. Kluwer, Dordrecht.

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

Iria da Cunha, Juan-Manuel Torres-Moreno, and Gerardo Sierra. 2011. On the development of the RST Spanish treebank. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 1–10, Portland, Oregon, USA. Association for Computational Linguistics.

Vera Demberg, Merel C.J. Scholman, and Fatemeh Torabi Asr. 2019. How compatible are our discourse annotation frameworks? Insights from mapping RST-DT and PDTB annotations. *Dialogue & Discourse*, 10(1):87–135.

Suzanne Eggins and J. Martin. 1997. Genres and Registers of Discourse. In *Discourse as Structure and Process: Discourse Studies: A Multidisciplinary Introduction*. SAGE Publications Ltd.

Elisa Ferracane, Titan Page, Junyi Jessy Li, and Katrin Erk. 2019. From news to medical: Cross-domain discourse segmentation. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 22–29, Minneapolis, MN. Association for Computational Linguistics.

Luke Gessler, Shabnam Behzad, Yang Janet Liu, Siyao Peng, Yilun Zhu, and Amir Zeldes. 2021. DisCoDisCo at the DISRPT2021 shared task: A system for discourse segmentation, classification, and connective detection. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 51–62, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Luke Gessler, Siyao Peng, Yang Liu, Yilun Zhu, Shabnam Behzad, and Amir Zeldes. 2020. AMALGUM – a free, balanced, multilayer English web corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5267–5275, Marseille, France. European Language Resources Association.

Grigorii Guz and Giuseppe Carenini. 2020. Coreference for discourse parsing: A neural approach. In *Proceedings of the First Workshop on Computational Approaches to Discourse*, pages 160–167, Online. Association for Computational Linguistics.

Benjamin Heinzerling and Michael Strube. 2019. Sequence tagging with contextual and non-contextual subword representations: A multilingual evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 273–291, Florence, Italy. Association for Computational Linguistics.

Freya Hewett and Manfred Stede. 2022. Extractive summarisation for German-language data: A text-level approach with discourse features. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 756–765, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Patrick Huber and Giuseppe Carenini. 2020. MEGA RST discourse treebanks with structure and nuclearity from scalable distant sentiment supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7442–7457, Online. Association for Computational Linguistics.

Mikel Iruskieta, María Jesús Aranzabe, Arantza Diaz de Ilarraza, Itziar Gonzalez-Dios, Mikel Lersundi, and Oier Lopez de Lacalle. 2013. The RST Basque TreeBank: An Online Search Interface to Check Rhetorical Relations. In *The 4th Workshop RST and Discourse Studies*, Fortaleza, Brasil.

Mikel Iruskieta, Gorka Labaka, and Juliano Desiderato Antonio. 2016. Detecting the central units in two different genres and languages: a preliminary study of Brazilian Portuguese and Basque texts. *Sociedad Española para el Procesamiento del Lenguaje Natural*, pages 65–72.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Top-down discourse parsing via sequence labelling. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational*

Linguistics: Main Volume, pages 715–726, Online. Association for Computational Linguistics.

David. Y. Lee. 2001. Genres, Registers, Text Types, Domains and Styles: Clarifying the Concepts and Navigating a Path through the BNC Jungle. *Language Learning & Technology*, 5:37–72.

Junyi Jessy Li, Kapil Thadani, and Amanda Stent. 2016. The role of discourse units in near-extractive summarization. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 137–147, Los Angeles. Association for Computational Linguistics.

Sujian Li, Liang Wang, Ziqiang Cao, and Wenjie Li. 2014. Text-level discourse dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25–35, Baltimore, Maryland. Association for Computational Linguistics.

Pia Linscheid, Peter Bourgonje, and Georg Rehm. 2021. Parsing discourse structures for semantic storytelling: Evaluating an efficient RST parser. In *Proceedings of QURATOR 2021 – Conference on Digital Curation Technologies. (QURATOR-2021), February 8-12, Berlin, Germany*. CEUR Workshop Proceedings.

Yang Liu. 2019. Beyond the Wall Street Journal: Anchoring and comparing discourse signals across genres. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 72–81, Minneapolis, MN. Association for Computational Linguistics.

Zhengyuan Liu, Ke Shi, and Nancy Chen. 2021. DMRST: A joint framework for document-level multilingual RST discourse segmentation and parsing. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 154–164, Punta Cana, Dominican Republic and Online. Association for Computational Linguistics.

Annie Louis, Aravind Joshi, and Ani Nenkova. 2010. Discourse indicators for content selection in summarization. In *Proceedings of the SIGDIAL 2010 Conference*, pages 147–156, Tokyo, Japan. Association for Computational Linguistics.

William C Mann and Sandra A Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Special Issue on Using Large Corpora, Computational Linguistics*, 19(2):313–330.

Aleksandre Maskharashvili, Symon Stevens-Guille, Xintong Li, and Michael White. 2021. Neural methodius revisited: Do discourse relations help with pre-trained models too? In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 12–23, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Zion Mengesha, Courtney Heldreth, Michal Lahav, Juliana Sublewski, and Elyse Tuennerman. 2021. “I don’t think these devices are very culturally sensitive.” — impact of automated speech recognition errors on African Americans. *Frontiers in Artificial Intelligence*, 4.

Mathieu Morey, Philippe Muller, and Nicholas Asher. 2017. How much progress have we made on RST discourse parsing? a replication study of recent results on the RST-DT. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1319–1324, Copenhagen, Denmark. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Quan Hung Tran, Trung Bui, Walter Chang, and Ndapa Nakashole. 2020. Rethinking self-attention: Towards interpretability in neural parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 731–742, Online. Association for Computational Linguistics.

Thanh-Tung Nguyen, Xuan-Phi Nguyen, Shafiq Joty, and Xiaoli Li. 2021. RST parsing from scratch. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1613–1625, Online. Association for Computational Linguistics.

Noriki Nishida and Yuji Matsumoto. 2022. Out-of-domain discourse dependency parsing via bootstrapping: An empirical analysis on its effectiveness and limitation. *Transactions of the Association for Computational Linguistics*, 10:127–144.

Siyao Peng, Yang Janet Liu, and Amir Zeldes. 2022. GCDT: A Chinese RST treebank for multigenre and multilingual discourse parsing. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 382–391, Online only. Association for Computational Linguistics.

Dina Pisarevskaya, Margarita Ananyeva, Maria Kobozeva, A. Nasedkin, S. Nikiforova, I. Pavlova, and A. Shelepov. 2017. Towards building a discourse-annotated corpus of Russian. In *23rd International Conference on Computational Linguistics and Intellectual Technologies. Dialogue*, pages 194–204, Moscow, Russia.

Rashmi Prasad, Bonnie Webber, Alan Lee, and Aravind Joshi. 2019. Penn Discourse Treebank Version 3.0. LDC2019T05.

Gisela Redeker, Ildikó Berzlánovich, Nynke van der Vliet, Gosse Bouma, and Markus Egg. 2012. Multi-layer discourse annotation of a Dutch text corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2820–2825, Istanbul, Turkey. European Language Resources Association (ELRA).

Deven Santosh Shah, H. Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5248–5264, Online. Association for Computational Linguistics.

Manfred Stede and Arne Neumann. 2014. Potsdam commentary corpus 2.0: Annotation for discourse research. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 925–929, Reykjavik, Iceland. European Language Resources Association (ELRA).

Maite Taboada and Julia Lavid. 2003. Rhetorical and thematic patterns in scheduling dialogues: A generic characterization. *Functions of Language*, 10(2):147–178.

Shujun Wan, Tino Kutschbach, Anke Lüdeling, and Manfred Stede. 2019. RST-tace a tool for automatic comparison and evaluation of RST trees. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 88–96, Minneapolis, MN. Association for Computational Linguistics.

Xinhao Wang, Binod Gyawali, James V. Bruno, Hillary R. Molloy, Keelan Evanini, and Klaus Zechner. 2019. Using Rhetorical Structure Theory to assess discourse coherence for non-native spontaneous speech. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 153–162, Minneapolis, MN. Association for Computational Linguistics.

Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 184–188, Vancouver, Canada. Association for Computational Linguistics.

Bonnie Webber. 2009. Genre distinctions for discourse in the Penn TreeBank. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 674–682, Suntec, Singapore. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online. Association for Computational Linguistics.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Attapol Rutherford, Bonnie Webber, Chuan Wang, and Hongmin Wang. 2016. CoNLL 2016 shared task on multilingual shallow discourse parsing. In *Proceedings of the CoNLL-16 shared task*, pages 1–19, Berlin, Germany. Association for Computational Linguistics.

An Yang and Sujian Li. 2018. SciDTB: Discourse dependency TreeBank for scientific abstracts. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 444–449, Melbourne, Australia. Association for Computational Linguistics.

Nan Yu, Meishan Zhang, and Guohong Fu. 2018. Transition-based neural RST parsing with implicit syntax features. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 559–570, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Nan Yu, Meishan Zhang, Guohong Fu, and Min Zhang. 2022. RST discourse parsing with second-stage EDU-level pre-training. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4269–4280, Dublin, Ireland. Association for Computational Linguistics.

Amir Zeldes. 2017. The GUM Corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.

Amir Zeldes. 2018. *Multilayer Corpus Studies*. Routledge Advances in Corpus Linguistics 22. Routledge, London.

Amir Zeldes, Nick Howell, Noam Ordan, and Yifat Ben Moshe. 2022. A second wave of UD Hebrew treebanking and cross-domain parsing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4331–4344, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Amir Zeldes, Yang Janet Liu, Mikel Iruskieta, Philippe Muller, Chloé Braud, and Sonia Badene. 2021. The DISRPT 2021 shared task on elementary discourse unit segmentation, connective detection, and relation classification. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 1–12, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Amir Zeldes and Nathan Schneider. 2023. Are UD Treebanks Getting More Consistent? A Report Card for English UD. In *Proceedings of the Sixth Workshop on Universal Dependencies (UDW, SyntaxFest 2023)*, Washington, DC. Association for Computational Linguistics.

Longyin Zhang, Fang Kong, and Guodong Zhou. 2021. Adversarial learning for discourse rhetorical structure parsing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics* *and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3946–3957, Online. Association for Computational Linguistics.

Yilun Zhu, Sameer Pradhan, and Amir Zeldes. 2021. OntoGUM: Evaluating contextualized SOTA coreference resolution on 12 more genres. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 461–467, Online. Association for Computational Linguistics.

A Cross-Corpus Segmentation Scores

The scores in Table 11 (3 run average) show fairly modest degradation on cross-corpus EDU segmentation scores in both directions (a little worse for RST-DT→GUM), using DisCoDisCo (Gessler et al., 2021), the winning EDU segmentation system from the 2021 DISRPT shared task (Zeldes et al., 2021).

train	**test**	**P**	**R**	**F1**
RST-DT	RST-DT	95.16	94.64	94.90
RST-DT	GUM V8	90.09	89.69	89.89
GUM V8	GUM V8	92.09	93.73	92.90
GUM V8	RST-DT	92.55	89.57	91.03

Table 11: Cross-Corpus Segmentation Performance.

B Breakdown of the GUM RST Data

The breakdown of genres in GUM V8 is shown in Table 12. Four of the genres in the corpus are still growing, and therefore include less material than other genres at present.

genres	**docs**	**tokens**	**EDUs**	**growing**
academic	18	17,168	1,969	
bio	20	18,209	2,066	
fiction	19	17,508	2,458	
how-to	19	17,085	2,367	
interview	19	18,189	2,404	
news	23	16,140	1,760	
reddit	18	16,364	2,231	
travel	18	16,513	1,785	
conversation	9	10,451	1,878	✓
speech	10	10,827	1,249	✓
textbook	10	11,190	1,397	✓
vlog	10	11,200	1,543	✓
total	**193**	**180,844**	**23,107**	

Table 12: Overview of RST Data by Genre in GUM V8.

C Model Configurations and Training Details

To ensure reproducibility and following the reproducibility criteria and checklist provided by EACL 2023, we provide model configurations and training details relevant to the experimental results presented in this paper.

For the BOTTOM-UP parser, the configurations of `SpanBERT-NoCoref` from Guz and Carenini (2020) was used as our base system, and we followed the hyperparameters and training settings therein across the board except the batch size due to memory limitations and embedding dimensions for organizational features.[7] Specifically, AdamW was used as the optimizer with a learning rate of $1e^{-5}$ for SpanBERT-base (Joshi et al., 2020) and $2e^{-4}$ for model-specific components (Guz and Carenini, 2020). Batch size was set to 1 (as opposed to 5 in the original implementation) and there were 20 epochs for each run. The organizational features used in Guz and Carenini (2020) followed Wang et al. (2017) and were represented as binary features in a learnable 5-dimensional embedding (as opposed to 10 in the original implementation). A vector of zeros of the same shape was used when a given feature is unavailable for the aforementioned organizational features as well as the categorical features experimented in §3.2, which all had an embedding dimension of 10.

For the TOP-DOWN parser, overall we followed the original hyperparameters and training settings therein; however, due to memory limitations, we modified the batch size to 3 from 12. The same `dev` set of respective corpora was used during training as in the training of the BOTTOM-UP parser from Guz and Carenini (2020). Additionally, XLM-RoBERTa-base (Conneau et al., 2020) was used as the language backbone.

All the training sessions were conducted using 1 NVIDIA Tesla T4 GPU on Google Cloud Platform. Table 13 shows the corresponding validation performance on RST-DT `test` and GUM V8 `test` reported in Tables 3 and 5 respectively using the BOTTOM-UP parser. Table 14 presents the corresponding validation performance (averaged over 5 runs) on RST-DT `test` and GUM V8 `test` reported in Table 4.

---

[7] Based on our reproducible results of this model (see Table 2) and comparing them to the reported results in the original paper, we believe that the change of these two configurations do not impact model performance significantly.

	S	N	R	# runs
RST-DT BASELINE	76.0	64.9	55.2	5
GUM `test`	67.0	54.5	45.9	5
RST-DT CONCAT	76.4	65.7	56.0	3
RST-DT FLAIR-LABEL	76.3	65.1	54.9	3
RST-DT SR-LABEL	76.3	64.7	55.2	3
RST-DT SR-GRAPH	76.6	65.5	55.5	3
RS-TDT SR-FT	76.3	64.9	55.4	5

Table 13: Validation Performance on RST-DT `test` and GUM `test` Results in Tables 3 and 5 (i.e. BOTTOM-UP).

	S	N	R	# runs
RST-DT BASELINE	75.3	65.0	55.9	5
GUM `test`	71.3	58.6	50.1	5

Table 14: Validation Performance on RST-DT `test` and GUM `test` Results in Table 4 (i.e. TOP-DOWN).

Note that since RST-DT does not have an established `dev` set, 10% of training data stratified by the number of EDUs in each document is used as the `dev` set following Guz and Carenini (2020) and is held constant across all the RST-DT-related experiments in this work. The list of the document names used in the `dev` set is provided in the code repository.[8] Additionally, since the `test` documents used in all the experiments in §3.3 and §3.4 are OOD data, the corresponding validation performance is not applicable and thus not reported. Information on the average runtime of each epoch for each model is provided in Table 15 below.

RST-DT			
model	**avg. runtime**	**model**	**avg. runtime**
BASELINE	2 hours 53 mins	SR-LABEL	2 hours 53 mins
CONCAT	5 hours 52 mins	SR-GRAPH	2 hours 50 mins
FLAIR-LABEL	2 hours 55 mins	SR-FT	2 hours 56 mins
GUM			
model	**avg. runtime**	**model**	**avg. runtime**
GUM `test`	3 hours	No Reddit	2 hours 43 mins
ALL-LARGE	2 hours 15 mins	No Voyage	2 hours 47 mins
No Academic	2 hours 43 mins	No How-to	2 hours 45 mins
No Bio	2 hours 45 mins	C1	56 mins
No Fiction	2 hours 44 mins	C2	56 mins
No Interview	2 hours 40 mins	C3	54 mins
No News	2 hours 47 mins		

Table 15: Average Runtime of Every Training Session's Epoch for Each Model using the BOTTOM-UP Parser.

D Relation Mapping

In the interest of reproducibility, Table 16 gives the exact relation mapping used for cross-corpus experiments in which relation labels were targeted.

GUM V8 Relations	**GUM V8 Classes**	**Corresponding RST-DT Classes**
adversative-antithesis	Adversative	Contrast
adversative-concession	Adversative	Contrast
adversative-contrast	Adversative	Contrast
attribution-positive	Attribution	Attribution
attribution-negative	Attribution	Attribution
causal-cause	Causal	Cause
causal-result	Causal	Cause
context-background	Context	Background
context-circumstance	Context	Background
contingency-condition	Contingency	Condition
elaboration-attribute	Elaboration	Elaboration
elaboration-additional	Elaboration	Elaboration
explanation-evidence	Explanation	Explanation
explanation-justify	Explanation	Explanation
explanation-motivation	Explanation	Explanation
evaluation-comment	Evaluation	Evaluation
joint-disjunction	Joint	Joint
joint-list	Joint	Joint
joint-sequence	Joint	Temporal
joint-other	Joint	Topic-Change
mode-manner	Mode	Manner-Means
mode-means	Mode	Manner-Means
organization-heading	Organization	Textual-Organization
organization-phatic	Organization	Topic-Comment
organization-preparation	Organization	Textual-Organization
purpose-attribute	Purpose	Elaboration
purpose-goal	Purpose	Enablement
restatement-partial	Restatement	Summary
restatement-repetition	Restatement	Summary
topic-question	Topic	Topic-Comment
topic-solutionhood	Topic	Topic-Comment
same-unit	same-unit	Same-Unit

Table 16: Relation Mapping of GUM V8 to RST-DT.

E Data Description for the GUM OOD Multi-Genre Experiments

Table 17 details the number of genres, documents, and EDUs used for training the models in the ten experiments in §3.3. For maximum reliability, testing was always conducted on the held-out genres' standard `dev+test` partitions specified in the official GUM V8 release (which is also provided in the repository of this paper), since the `dev` set for the targeted genres cannot be used for early stopping to simulate real OOD data in the wild.

F An Example OOD Parser Output

Figure 3 provides a comparison of a fragment of parser output versus the corresponding gold annotation to exemplify the distinction between gold ORGANIZATION and predicted JOINT discussed

---

[8] `https://github.com/janetlauyeung/crossGENRE4RST`

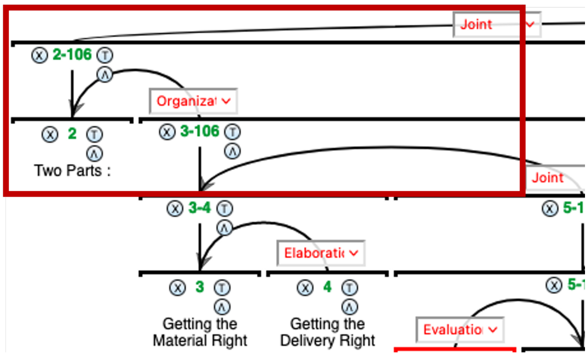
models	genres	docs	EDUs
No Academic	11	131	16,088
No Bio	11	129	15,901
No Fiction	11	130	15,640
No Interview	11	130	15,599
No News	11	126	16,252
No Reddit	11	131	15,892
No Voyage	11	131	16,133
No How-to	11	130	15,672
ALL-LARGE	8	122	13,703
GUM test	12	145	17,610

Table 17: Training Data Composition used in §3.3.

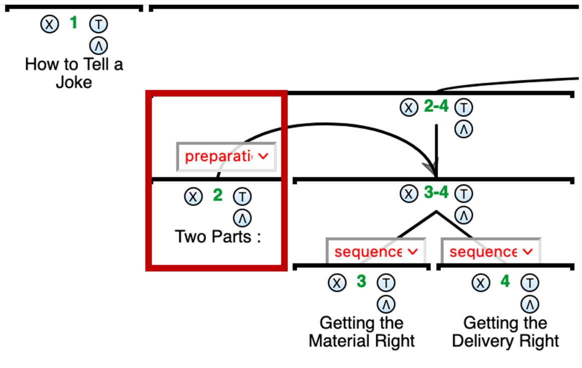
in Section 4. Additionally, Figure 4 provides a fragment of parser output from one of the OVA models, No How-to, which exemplifies some of the observations and discussions brought up in the error analysis such as the most erroneous relation class ORGANIZATION in how-to guides (EDU 107). Moreover, these figures also demonstrate that although discourse markers (DMs) are cues in many cases, they can lead to errors if they are ambiguous, by distracting the parser from other (non-DM) signals such as the misidentified ELABORATION of EDU 110, where the gold ORGANIZATION is signaled by the colon and the numerical matching of 'Two parts' to the two nucleaus EDUs, 3 and 4 in Figure 3; and the CAUSAL of EDUs 112-116 in Figure 4, where an intensifying 'so' in 112 ('so comfortable') may have misled the parser into a resultative 'so' reading.

G Confusion Matrices

Figures 5a–5h show confusion matrices for all the non-growing genres from their corresponding OVA models, and Figures 5i–5l are for all the growing genres based on the automatic parses from the ALL-LARGE model where the training data contains only the 8 non-growing genres. Note that all the matrices were produced by converting the automatic parses into the RST dependency representation following Li et al. (2014) and do not reflect attachment errors. The conversion code is available at https://github.com/amir-zeldes/rst2dep.

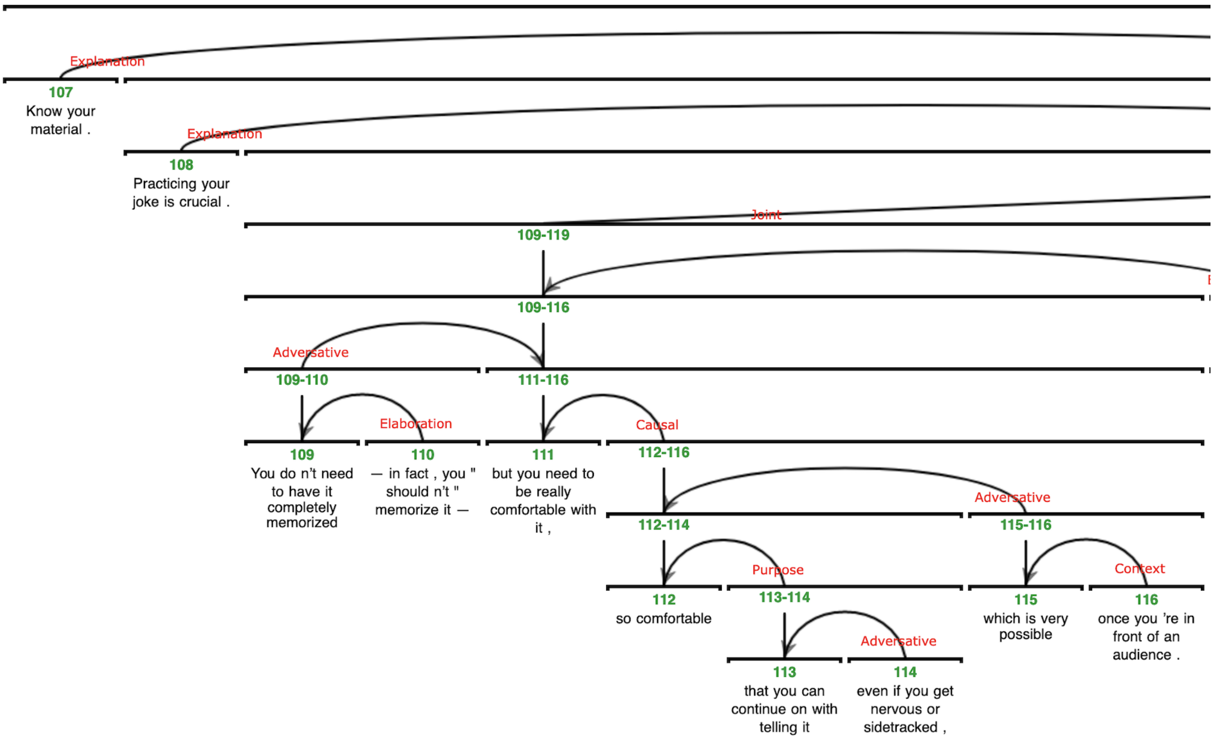


(a) Parser Output Fragment from the *No How-to* Model.

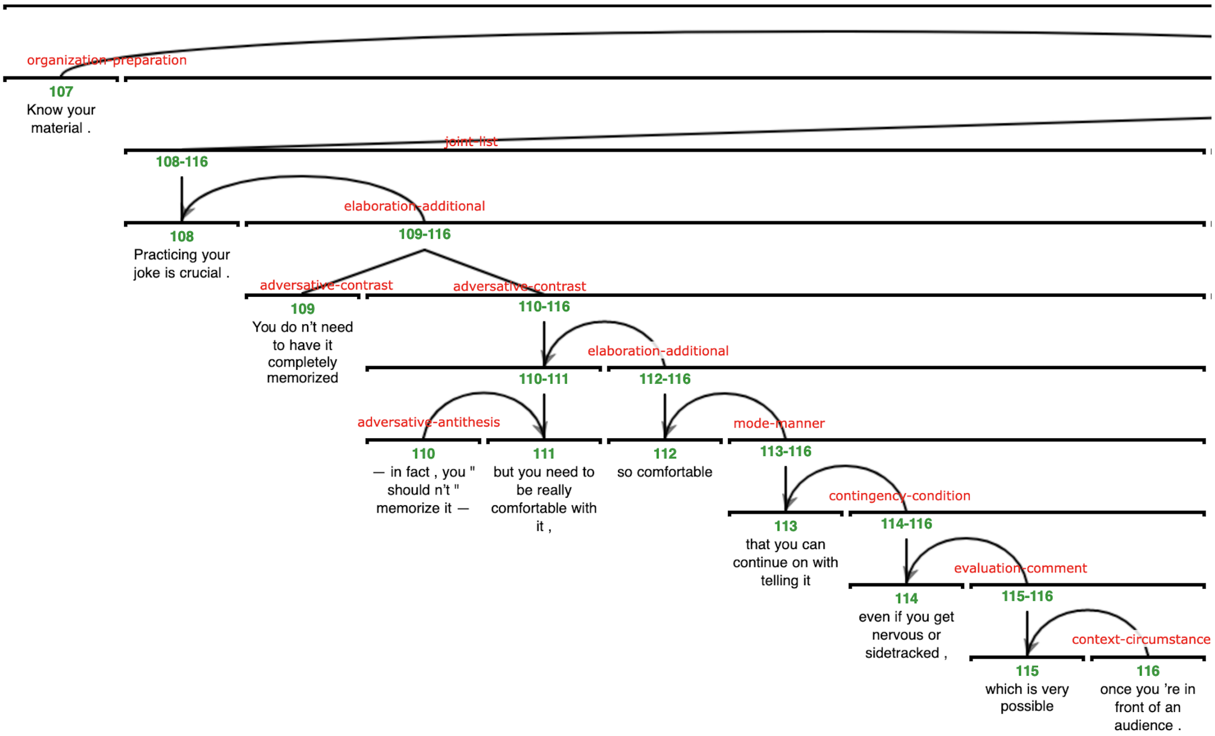


(b) The Corresponding Gold RST Annotation.

Figure 3: An Example of an OOD Parser Output vs. the Corresponding Gold Annotations from *how-to guides*: ORGANIZATION vs. JOINT.

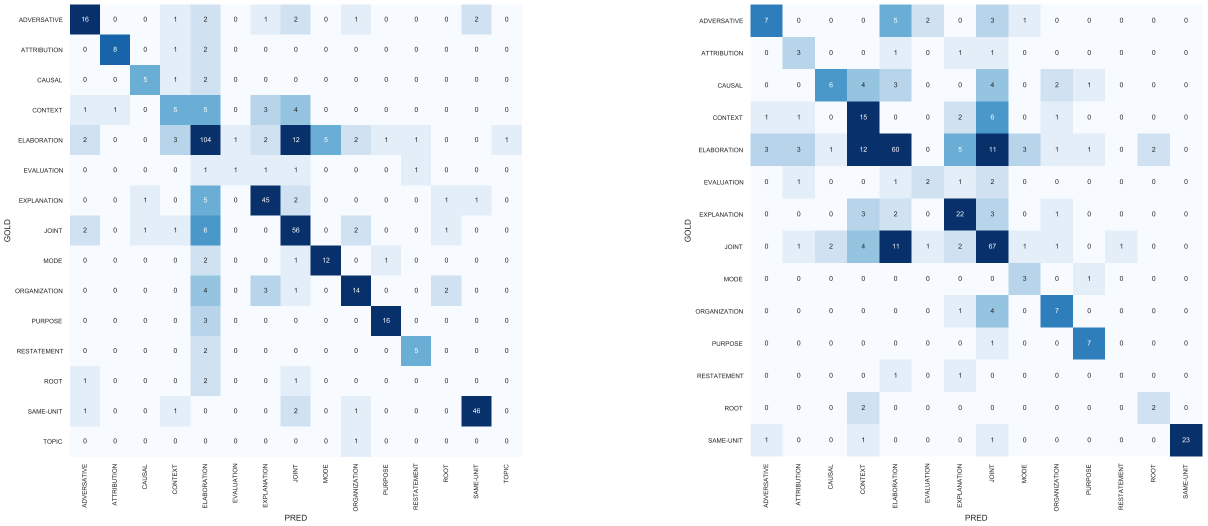


(a) Parser Output Fragment from the OVA (No How-to) Model in §3.3.



(b) The Corresponding Gold RST Annotation.

Figure 4: An Example of an OOD Parser Output vs. the Corresponding Gold Annotations from *how-to guides*. This fragment is selected from the document GUM_whow_joke, which gives advice on how to tell jokes.

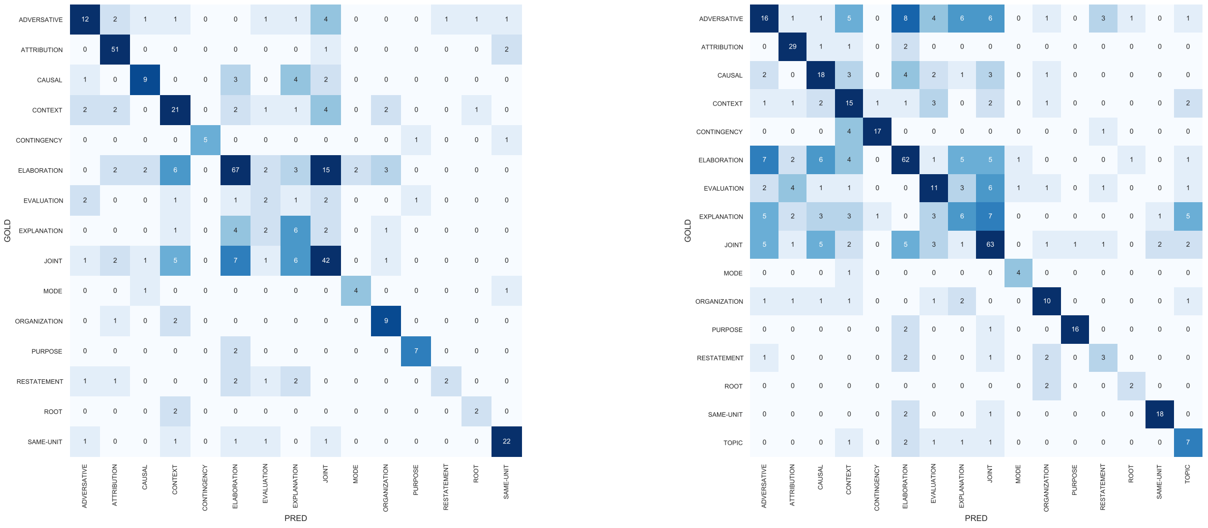


(a) *academic*

(b) *bio*

(c) *fiction*

(d) *interview*

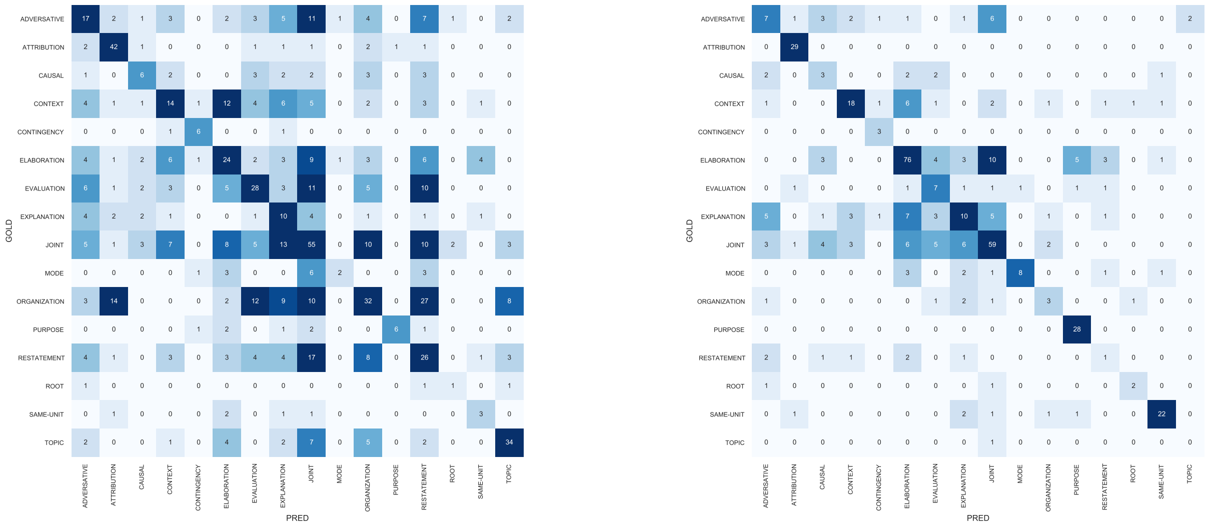


(e) *news*

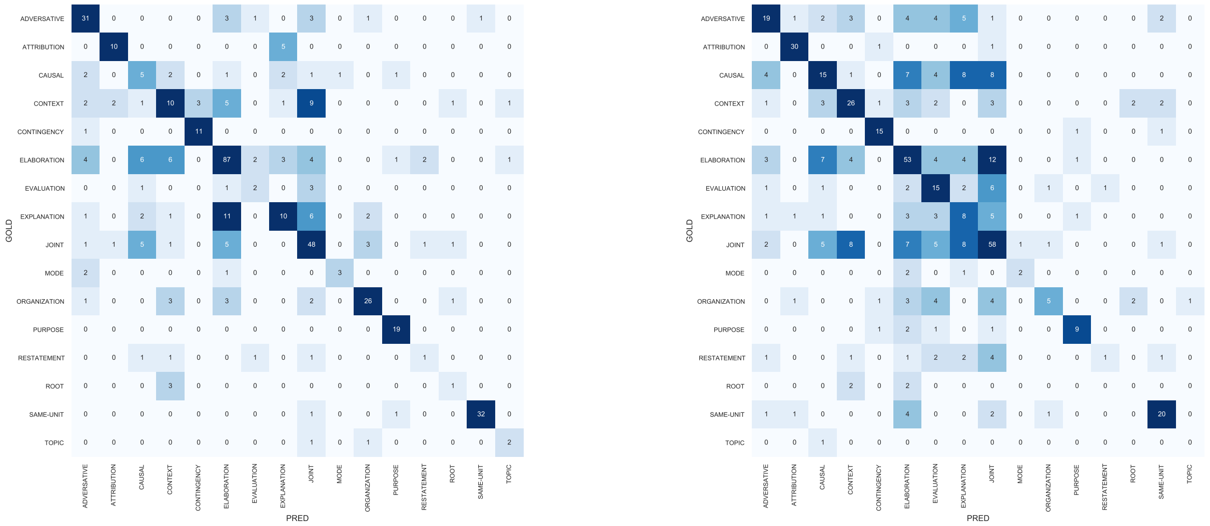
(f) *reddit*

Figure 5: Confusion Matrices for All Genres from their OVA Models or the ALL-LARGE Model from §3.3.

(g) *voyage*　　　　(h) *whow*



(i) *conversation*　　　　(j) *speech*



(k) *textbook*　　　　(l) *vlog*

Figure 5: Confusion Matrices for All Genres from their OVA Models or the ALL-LARGE Model from §3.3.

Enriching Biomedical Knowledge for Low-resource Language Through Large-Scale Translation

Long Phan[1]*, **Tai Dang**[1,3]*, **Hieu Tran**[1]*, **Trieu H. Trinh**[1,4]*,
Vy Phan[3], **Lam D. Chau**[2] and **Minh-Thang Luong**[1]
[1]VietAI Research
[2]Case Western Reserve University
[3]University of Massachusetts-Amherst
[4]New York University

Abstract

Biomedical data and benchmarks are highly valuable yet very limited in low-resource languages other than English, such as Vietnamese. In this paper, we use a state-of-the-art translation model in English-Vietnamese to translate and produce both pretrained and supervised data in the biomedical domains. Thanks to such large-scale translation, we introduce ViPubmedT5, a pretrained Encoder-Decoder Transformer model trained on 20 million translated abstracts from the high-quality public PubMed corpus. ViPubMedT5 demonstrates state-of-the-art results on two different biomedical benchmarks in summarization and acronym disambiguation. Further, we release ViMedNLI - a new NLP task in Vietnamese translated from MedNLI using the recently public En-vi translation model and carefully refined by human experts, with evaluations of existing methods against ViPubmedT5.

1 Introduction

In recent years, pretrained language models (LMs) have played an important and novel role in developing many Natural Language Processing (NLP) systems. Utilizing large pretrained models like BERT (Devlin et al., 2018), XLNET (Yang et al., 2019), ALBERT (Lan et al., 2019), RoBERTa (Liu et al., 2019), GPT-3 (Brown et al., 2020) BART (Lewis et al., 2019), and T5 (Raffel et al., 2019) has become an effective trend in natural language processing. All these large models follow the Transformer architecture proposed by (Vaswani et al., 2017) with the attention mechanism. The architecture has been proven to be very suitable for finetuning downstream tasks leveraging transfer learning with their large pretrained checkpoints. Before the emergence of large Transformer LMs, traditional wording embedding gave each word a fixed global representation. Large pretrained models can derive word vector representation from a trained large corpus. This will give the pretrained model a better knowledge of the generalized representation of a trained language/domain and significantly improve performance on downstream finetune tasks. The success of pretrained models on a generative domain (BERT, RoBERTa, BART, T5, etc.) has created a path in creating more specific-domain language models such as CodeBERT (Feng et al., 2020) and CoTexT (Phan et al., 2021b) for coding languages, TaBERT (Yin et al., 2020) for tabular data, BioBERT (Lee et al., 2019) and PubmedBERT (Tinn et al., 2021) for biomedical languages.

Biomedical literature is getting more popular and widely accessible to the scientific community through large databases such as Pubmed[1], PMC[2], and MIMIC-IV (Johnson et al., 2021). This also leads to many studies, corpora, or projects released to further advance the Biomedical Natural Language Processing field (Lee et al., 2019; Tinn et al., 2021; Phan et al., 2021a; Yuan et al., 2022). These biomedical domain models leverage transfer learning from pretrained models (Devlin et al., 2018; Clark et al., 2020; Raffel et al., 2019; Lewis et al., 2019) to achieve state-of-the-art results on multiple Biomedical NLP tasks like Named Entity Recognition (NER), Relation Extraction (RE), or document classification.

However, few studies have been on leveraging large pretrained models for biomedical NLP in low-resource languages. The main reason is the lack of large biomedical pretraining data and benchmark datasets. Furthermore, collecting biomedical data in low-resource languages can be very expensive due to scientific limitations and inaccessibility.

We attempt to overcome the issue of lacking biomedical text data in low-resource languages by using state-of-the-art translation works. We start

---

[*] The first four authors contributed equally to this work

[1] https://pubmed.ncbi.nlm.nih.gov

[2] https://www.ncbi.nlm.nih.gov/pmc

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3131–3142
May 2-6, 2023

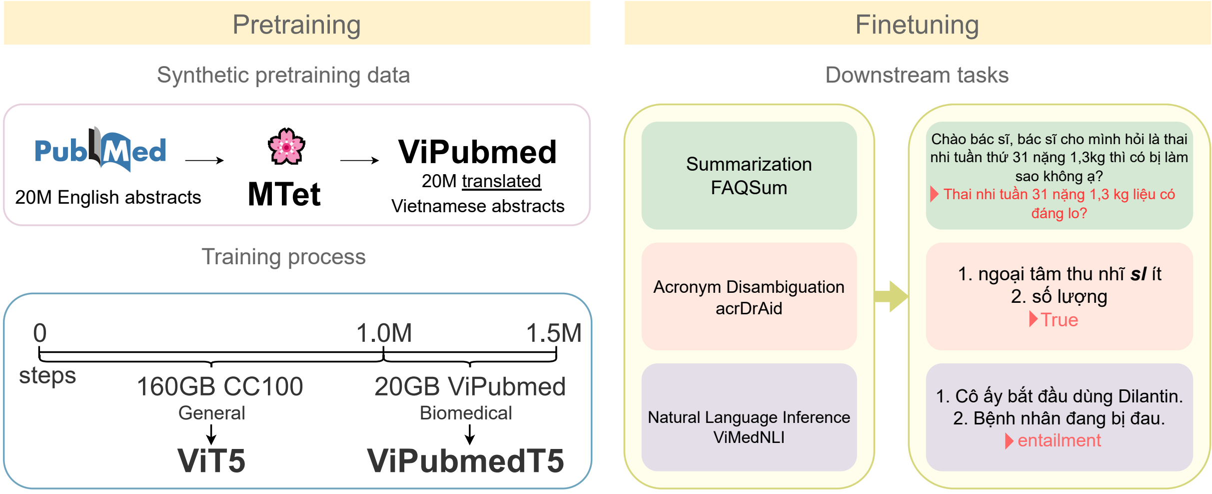


Figure 1: Overview of the pretraining and finetuning of ViPubmedT5

with the Vietnamese language and keep everything reproducible for other low-resource languages in future work.

We introduce ViPubmedT5, a pretrained encoder-decoder transformer model trained on synthetic Vietnamese biomedical text translated with state-of-the-art English-Vietnamese translation work. Meanwhile, we also introduced ViMedNLI, a medical natural language inference task (NLI), translated from the English MedNLI (Romanov and Shivade, 2018) with human refining.

We thoroughly benchmark the performance of our ViPubmedT5 model when pretrained with synthetic translated biomedical data with ViMedNLI and other public Vietnamese Biomedical NLP tasks (Minh et al., 2022). The results show that our model outperforms both general domain (Nguyen and Nguyen, 2020; Phan et al., 2022) and health-specific domain Vietnamese (Minh et al., 2022) pretrained models on biomedical tasks.

In this work, we offer the following contributions:

- A state-of-the-art English-Vietnamese Translation model (with self-training) on medical and general domains.
- A first Encoder-Decoder Transformer model ViPubmedT5 pretrained on large-scale synthetic translated biomedical data.
- A Vietnamese medical natural language inference dataset (ViMedNLI) that translated from MedNLI (Romanov and Shivade, 2018) and refined with biomedical expertise human.
- We publicize our model checkpoints, datasets, and source code for future studies on other low-resource languages.

2 Related Works

The development of parallel text corpora for translation and use for training MT systems has been a rapidly growing field of research. In recent years, low-resource languages have gained more attention from the industry, and academia (Chen et al., 2019b; Shen et al., 2021; Gu et al., 2018; Nasir and Mchechesi, 2022). Previous works include gathering more training data or training large multilingual models (Thu et al., 2016; Fan et al., 2021). Low-Language MT enhances billions of people's daily life in numerous fields. Nonetheless, there are specific domains crucial yet limited such as biomedical and healthcare, in which MT systems have not been able to contribute adequately.

Previous works using MT systems for biomedical tasks includes (Neves et al., 2016; Névéol et al., 2018). Additionally, several biomedical parallel (Deléger et al., 2009) have been utilized just for terminology translation only. Pioneer attempts to train MT systems using a corpus of MEDLINE titles (Wu et al., 2011), and the use of publication titles and abstracts for both ES-EN and FR-EN language pairs (Jimeno-Yepes et al., 2012). However, none of these works targets low-resource languages. A recent effort to train Vietnamese ML systems for biomedical and healthcare is Minh et al. (2022). These, however, do not utilize the capability of MT systems, instead relying on manual crawling. Therefore, this motivation has led us to employ MT systems to contribute high-quality Vietnamese datasets that emerged from the English language. To the best of our knowledge, this is the first work utilizing state-of-the-art machine translation to translate both self-supervised and supervised learning biomedical data for pretrained models in a low-resource language setting.

2.1 Pubmed and English Biomedical NLP Studies

The Pubmed[3] provides access to the MEDLINE database[4] which contains titles, abstracts, and metadata from medical literature since the 1970s. The dataset consists of more than 34 million biomedical abstracts from the literature collected from sources such as life science publications, medical journals, and published online e-books. This dataset is maintained and updated yearly to include more up-to-date biomedical documents.

Pubmed Abstract has been the main dataset for almost any state-of-the-art biomedical domain-specific pretrained models (Lee et al., 2019; Yuan et al., 2022; Tinn et al., 2021; Yasunaga et al., 2022; Alrowili and Shanker, 2021; Phan et al., 2021a). In addition, many well-known Biomedical NLP/NLU benchmark datasets are created based on the unlabeled Pubmed corpus (Doğan et al., 2014; Nye et al., 2018; Herrero-Zazo et al., 2013; Jin et al., 2019). Recently, to help accelerate research in biomedical NLP, Gu et al. (2020) releases BLURB (**B**iomedical **L**anguage **U**nderstanding & **R**easoning **B**enchmark), which consists of multiple pretrained biomedical NLP models and benchmark tasks. It is important to note that all of the top 10 models on the BLURB Leaderboard[5] are pretrained on the Pubmed Abstract dataset.

2.2 English-Vietnamese Translation

Due to its limitation of high-quality parallel data available, English-Vietnamese translation is classified as a low-resource translation language (Liu et al., 2020). One of the first notable parallel datasets and En-Vi neural machine translation is ISWLT'15 (Luong and Manning, 2015) with 133K sentence pairs. A few years later, PhoMT (Doan et al., 2021) and VLSP2020 (Ha et al., 2020) released larger parallel datasets, extracted from publicly available resources for the English-Vietnamese translation.

Recently, VietAI[6] curated the largest 4.2M high-quality training pairs from various domains and achieved state-of-the-art on English-Vietnamese translation (Ngo et al., 2022). The work also focuses on En-Vi translation performance across multiple domains, including biomedical. As a result, the project's NMT outperforms existing En-Vi translation models (Doan et al., 2021; Fan et al., 2020) by more than 2% in the BLEU score.

3 Improvements on Biomedical English-Vietnamese Translation through Self-training

To generate a large-scale synthetic translated Vietnamese biomedical corpus, we first look into improving the existing English-Vietnamese translation system in the biomedical translation domain. Previous work from Ngo et al. (2022) has shown that En-Vi biomedical bitexts are very rare, even for large-scale bitext mining. Therefore, we look into self-training to leverage the available monolingual English biomedical data.

Self-training approach has been experimented with in He et al. (2019) and utilized to improve translation on low-resource MT systems (Chen et al., 2019a). The advantage of this method is that the source side of the monolingual corpus can be domain-specific data for translation. However, the shortcoming is that the generated targets can be low-quality and affect the machine translation performance. Therefore, we start with the English-Vietnamese machine translation model from Ngo et al. (2022), denoted bT_A, which achieves state-of-the-art results on both En-Vi biomedical and general translation domains.

We use bT_A to translate and generate a synthetic parallel biomedical dataset with 1M pairs of English-Vietnamese biomedical abstracts from the Pubmed Corpus. The new 1M En-Vi biomedical pairs are then concatenated with the current high-quality En-Vi translation dataset from MTet (Ngo et al., 2022) and PhoMT (Doan et al., 2021), increasing from 6.2M to 7.2M En-Vi sentence pairs total. To verify the effectiveness of our new self-training data, we re-finetune the bT_A model on this 7.2M bitexts corpus. We report the model performance on the medical test set from MTet and the general test set from PhoMT in Table 1 (the translation performances on other domains like News, Religion, and Law are reported in Appendix A for

---

[3] https://pubmed.ncbi.nlm.nih.gov

[4] https://www.nlm.nih.gov/bsd/pmresources.html

[5] https://microsoft.github.io/BLURB/leaderboard.html

[6] https://vietai.org

Table 1: BLEU Scores Results for En-Vi Translation on MTet Medical and PhoMT General Test Sets

Model	**Finetune Datasets**	**MTet Medical Test Set**	**PhoMT General Test Set**
M2M100	CCMatrix + CCAligned	30.18	35.83
Google Translate	-	38.60	39.86
SOTA	MTet+PhoMT	38.69	45.47
Ours	MTet+PhoMT +1M Self-training Pubmed Abstracts	**45.61**	**46.01**

Notes: The best BLEU scores are in bold. The state-of-the-art (SOTA) model and MTet dataset are from Ngo et al. (2022); PhoMT dataset and Google Translate's result are from Doan et al. (2021). M2M100 model is from Fan et al. (2020).

further reference).

The results show that our model outperforms existing Machine Translation systems in English-Vietnamese translation by applying self-training. We obtain a significant gain of 6.61 BLEU Score (38.69->45.61) on the MTet Medical test set. Our model with self-training also achieves state-of-the-art results on the PhoMT general domain test set by 0.53 BLEU Score (45.47->46.01). This shows that our approach not only improves the English-Vietnamese translation performance in the biomedical context but also generalizes to general translation. We further discuss our self-training model performance on other translation domains in Appendix A.

4 ViPubmed

After developing a new state-of-the-art machine translation system for English-Vietnamese translation in the biomedical domain in Section 3, we apply the system, denoted bT_B, on downstream translation to generate the first large-scale synthetic translated biomedical corpus for Vietnamese.

To ensure that our translated ViPubmed dataset contains up-to-date biomedical research (for example, Covid-19 diseases and Covid-19 vaccines), we use the newest Pubmed22[7] which contains approximately 34 million English biomedical abstracts published. The raw data is compressed in XML format. We then parse these structured XMLs to obtain the abstract text with Pubmed Parser[8] (Achakulvisut et al., 2020).

The machine translation model bT_B is an Encoder-Decoder Transformer based model with 512 token-length for input and output. Therefore, we filter out English Pubmed abstracts with more than 512 tokens. For fair size comparison with the unlabeled dataset of other health-related Vietnamese pretrained models (discussed in Section 7.2), we take a subset of 20M biomedical abstracts (20GB of text) for translation and leave a larger subset for future releases. We then translate the 20M English biomedical abstracts with the bT_B model using 4 TPUv2-8 and 4 TPUv3-8.

5 ViMedNLI

Along with an unlabeled dataset for pretraining, we also introduce a benchmark dataset generated by translation and refined with human experts. We start with a natural language inference (NLI) task as it is less prone to errors in biomedical entity translation compared to named-entity recognition (NER) or relation extraction (RE) tasks. The process of creating the ViMedNLI is shown in Figure 2.

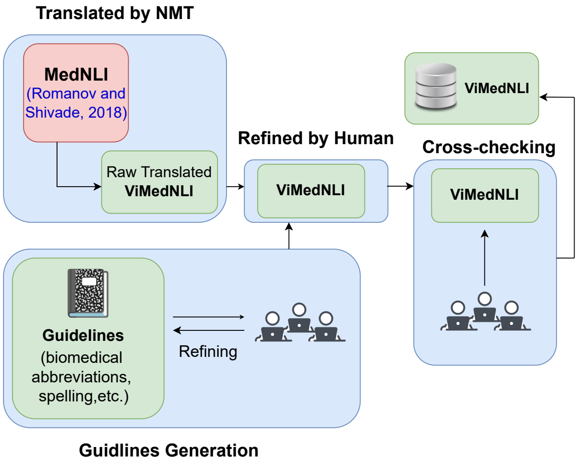


Figure 2: The Process of ViMedNLI Corpus Creation

---

[7] https://ftp.ncbi.nlm.nih.gov/pubmed/baseline

[8] https://github.com/titipata/pubmed_parser

Table 2: Some Examples of Abbreviations and Spelling Refining

#	MedNLI	Translated by NMT	Refined by human
Abbreviations Refining			
1	Electrocardiograms revealed no QRS changes	Điện tâm đồ cho thấy không có thay đổi về QRS.	Điện tâm đồ cho thấy không có thay đổi về phức độ QRS
2	Patient has no PMH	Bệnh nhân không có PMH	bệnh nhân không có tiền sử bệnh
3	Patient is post op	Bệnh nhân đã hồi phục (*Patient is recovered*)	Bệnh nhân hậu phẫu thuật
Spelling Refining			
4	The infant was born at herm	Đứa bé được sinh ra ở Herm (*The baby was born at Herm*)	Đứa bé được sinh đủ tháng (The baby was born at term)
5	The patient had an sotesophytes	Bệnh nhân có sinh cảm (*The patient has flu*)	Bệnh nhân bị viêm xương khớp (*The patient had osteophytes*)
6	Patient has delerium	Bệnh nhân có hội chứng delerium (*Patient has delerium syndrome*)	Bệnh nhân bị mê sảng (*Patient has delirium*)

#1: Abbreviation in English can be used in both English and Vietnamese.
#2: Abbreviation can only be used in English. In Vietnamese, abbreviation is different.
#3: Abbreviation in English is wrong when translated to Vietnamese.
#4: The word *"term"* is misspelled as *"herm"*.
#5: The word *"osteophytes"* is misspelled as *"sotesophytes"*.
#6: The word *"delirium"* is misspelled as *"delerium"*.

5.1 MedNLI

MedNLI (Romanov and Shivade, 2018) is an NLI dataset annotated by doctors and grounded in the patients' medical history. Given a premise sentence and a hypothesis sentence, the relation of the two sentences (entailment, contradiction, neutral) is labeled by two board-certified radiologists. The source of premise sentences in MedNLI is from MIMIC-III (Johnson et al., 2016), a large open-source clinical database. The dataset has been widely studied and benchmarked by the Biomedical NLP research community[9] (Peng et al., 2019; Phan et al., 2021a; El Boukkouri et al., 2020; Alrowili and Shanker, 2021; Kanakarajan et al., 2019).

5.2 Dataset Challenges

We follow the same procedures discussed in Section 4 to translate the same training, development, and test sets released in Romanov and Shivade (2018). The time and resources to translate the dataset are negligible as there are a total of 14522 samples.

However, upon translating the dataset with NMT, we find out that the English clinical note domain has a distinct sublanguage with unique challenges (abbreviations, inconsistent punctuation, misspellings, etc.). This observation has also been addressed in Friedman et al. (2002) and Meystre et al. (2008). Such differences in clinical language representation challenge the translation output and our quest to release a high-quality medical dataset.

---

[9] https://paperswithcode.com/dataset/mednli

Table 3: Statistics of finetuned datasets

Corpus	Train	Dev	Test	Task	Domain
acrDrAid, pairs	4000	523	1130	Acronym disambiguation	Medical
FAQSum, documents	10621	1326	1330	Abstractive summarization	Healthcare
ViMedNLI, pairs	11232	1395	1422	Inference	Clinical

5.3 Human Refining

The unique challenges of clinical data under translation settings (discussed in Section 5.2) require us to work with humans who not only have expertise in biomedical knowledge but are also sufficient in both English and Vietnamese languages to refine the dataset. Therefore, we collaborate with premedical Vietnamese students who studied at well-known U.S. Universities to refine the ViMedNLI datasets.

The refining process starts with a comprehensive guidelines document with thorough annotation instructions and examples. Then, as clinical notes contain a significant amount of technical abbreviations that the machine translation system can not translate initially (Section 5.2), we work with the medical annotators to create abbreviations and their expansion forms. To make sure the expansion form of these abbreviations generalizes well in real-world settings, we verify the use case of these words through multiple Vietnamese medical websites, blogs, and online dictionaries. Hence, we decided to keep the original English abbreviations, replace them with a Vietnamese expansion form, or replace them with a Vietnamese abbreviation. Some examples of this process are shown in Table 2.

Aside from the English medical abbreviations, there are several grammatical and spelling mistakes the machine translation system does not understand, translating either into Vietnamese meanings or even failing to translate. Human refining is therefore required. The phrase *"The infant was born at herm"*, for example, was translated as "Đứa bé được sinh ra ở Herm". The word *"herm"*, which should be spelled as *"term"*, is misspelled and has no medical meaning. The accurate translation should be "Đứa bé được sinh đủ tháng" (*"The infant was born at term"*). Table 2 shows more examples of spelling refining cases.

Additionally, the machine translation system occasionally produces incorrect Vietnamese meanings when translating words with proper English spelling and grammar. Considering the sentence *"The patient had post-term delivery"* as an example. Despite having the meaning "Bệnh nhân sinh muộn", it was mistranslated as "Bệnh nhân sinh non" (*"The patient had pre-term delivery"*). Another example is "Narrowing of the vessels", which means "Thu hẹp các mạch" rather than "Thu hẹp các" (no meaning).

6 ViPubmedT5

With an unlabeled synthetic translated ViPubmed Corpus (Section 4) and a benchmark ViMedNLI dataset (Section 5), we pretrain and finetune a Transformer-based language model (Vaswani et al., 2017) to verify the effectiveness of our approach in enriching Vietnamese biomedical domain with translation data. We explain our model and the pretraining settings we applied in this section.

6.1 Model Architecture

We adopt the Transformer encoder-decoder model proposed by Vaswani et al. (2017), the ViT5 (Phan et al., 2022) checkpoints, and T5 framework [10] implemented by Raffel et al. (2019). ViT5 is the first monolingual Vietnamese Transformer model; the model achieves state-of-the-art results on multiple Vietnamese general tasks, including generation and classification. The ViT5 publication releases 2 model sizes - base and large. We train ViPubmedT5 using the base setting (220 million parameters) and leave larger models for future work.

6.2 Pretraining

We pretrain our ViPubmedT5 on 20GB of translated biomedical data ViPubmed (Section 4). We leverage the Vietnamese checkpoints in the original ViT5 work (Phan et al., 2022) and continuously pretrain the model on the synthetic biomedical-specific data for another 500k steps. Previous works (Lee et al., 2019; Tinn et al., 2021) have shown that this approach will allow pretrained language models to learn a better representation of

---

[10] https://github.com/google-research/text-to-text-transfer-transformer

Table 4: Tests results on Vietnamese health and biomedical tasks

Domain	Datasets	Metrics	**PhoBERT**	**ViT5**	**ViHealthBERT**	**ViPubmedT5**
			(+news)	(+cc100)	(+health text mining)	(+translated ViPubmed)
Healthcare	FAQSum	RougeL	41.16	**61.3**	43.85	60.6
Medical	acrDrAid	Mac-F1	82.51	88	86.7	**89.04**
Clinical	ViMedNLI	Acc	77.29	77.85	79.04	**81.65**

Notes: The best scores are in bold, and the second best scores are underlined. PhoBERT & ViHealthBERT scores on FAQSum and acrDrAid are from Minh et al. (2022)

biomedical language context while maintaining the core Vietnamese language representation.

We train ViPubmedT5 using the same spans-masking learning objective as Raffel et al. (2019). During self-supervised training, spans of biomedical text sequences are randomly masked (with sentinel tokens). The target sequence is formed as the concatenation of the same sentinel tokens and the real masked spans/tokens.

7 Experiments

7.1 Benchmark dataset

We finetune and benchmark our pretrained ViPubmedT5 model on two public Vietnamese biomedical-domain datasets acrDrAid & FAQSum, (Minh et al., 2022) and our released ViMedNLI (Section 5). Detailed statistics of the three datasets are shown in Table 3.

- **acrDrAid** (Minh et al., 2022) is a Vietnamese Acronym Disambiguation (AD) dataset that contains radiology reports from Vinmec hospital[11], Vietnam. The task is correctly identifying the expansion of an acronym in a given radiology report context. The dataset is annotated by three expert radiologists. The acrDrAid has 135 acronyms and 424 expansion texts in total.

- **FAQ Summarization** (Minh et al., 2022) is a Vietnamese summarization dataset collected from FAQ sections of multiple *healthcare* trustworthy sites. For each FAQ section, the question text is the input sequence, and the title is a target summary.

- **ViMedNLI** is our released dataset discussed in Section 5.

7.2 Baseline

To verify the effectiveness of our proposed methods, we compare our ViPubmedT5 model with other state-of-the-art Vietnamese pretrained models:

- **PhoBERT** (Nguyen and Nguyen, 2020) is the first public large-scale monolingual language model pretrained for the Vietnamese language. The model follows the original RoBERTa (Liu et al., 2019) architecture. PhoBERT is trained on a 20GB word-level Vietnamese news corpus.

- **ViT5** (Phan et al., 2022) is the most recent state-of-the-art Vietnamese pretrained model for both generation and classification tasks. The model is trained on a general domain CC100-vi corpus.

- **ViHealthBERT** (Minh et al., 2022) is the first domain-specific pretrained language model for Vietnamese healthcare. After initializing weights from PhoBERT, the model is trained on 25M health sentences mined from different sources.

8 Results

The main finetuned results are shown in Table 4. The main takeaway is that training on synthetic translated biomedical data allows ViPubmedT5 to learn a better biomedical context representation. As a result, ViPubmedT5 achieves state-of-the-art in Medical and Clinical contexts while performing slightly worse than ViT5 in healthcare topics.

On the healthcare domain (FAQSum), ViPubmedT5 approximates the current state-of-the-art result (60.6 and 61.3) while outperforming the other models by a large margin (43.85). The slight difference in performance to ViT5 signifies a difference in data distribution in PubMed abstracts (scientific

---

[11] https://vinmec.com/

writing) and FAQSum (dialogues between patients and doctors).

For both medical and clinical datasets, ViPubmedT5 outperforms other existing models. There are also notable improvements from the general domain ViT5 to ViPubmedT5 (88->89.04 in acrDrAid and 77.85->81.65 in ViMedNLI). This indicates that the translated ViPubmed corpus contains biomedical knowledge that low-resource Vietnamese pretrained models can leverage.

Meanwhile, our newly translated ViMedNLI can serve as a robust baseline dataset for Vietnamese BioNLP research. Both health and biomedical domain models (ViHealthBERT & ViPubmedT5) perform better than general domain models (PhoBERT & ViT5) on the ViMedNLI dataset. This shows that our translated and refined ViMedNLI dataset is high-quality and has robust biomedical contexts.

9 Scaling to Other Languages

Our novel way of utilizing a state-of-the-art NMT system to generate synthetic translated medical data for pretrained models is not limited to the Vietnamese language and is scalable to many other low-resource languages. Recent works focus on improving the quality of multiple low-resource NMT systems (NLLB Team et al., 2022; Fan et al., 2020; Bañón et al., 2020). These new state-of-the-art NMTs make the approach discussed in this paper more practical to produce synthetic translated biomedical data, enriching the Biomedical NLP research knowledge in multiple low-resource languages.

10 Conclusion

We utilize the state-of-the-art translation model MTet to scale up the very low-resourced yet highly valuable biomedical data in Vietnamese. Namely, ViPubMedT5, a T5-style Encoder-Decoder Transformer pretrained on a large-scale translated corpus of the biomedical domain that demonstrated state-of-the-art results on both inference and acronym disambiguation in the biomedical domain. We also introduced ViMedNLI, a machine-translated and human-expert refined benchmark in natural language inference to further grow the Vietnamese suite of benchmarks and data in biomedical data.

11 Limitations

Although our pretrained model trained on synthetic translated biomedical data produces state-of-the-art results on downstream tasks for the Vietnamese language, the approach hugely depends on the quality of the NMTs for other low-resource languages. Thanks to recent studies and contributions from the Vietnamese research community (Section 2.2), the English-Vietnamese translation system has proven strong enough for us to conduct the experiments discussed in this work. However, the NMT's actual performance needed before making the translated biomedical data useful for pretrained models is still a question that requires further studies.

12 Acknowledgements

We would like to thank the Google TPU Research Cloud (TRC) program and Soonson Kwon (Google ML Ecosystem programs Lead) for their supports.

References

Titipat Achakulvisut, Daniel Acuna, and Konrad Kording. 2020. Pubmed parser: A python parser for pubmed open-access xml subset and medline xml dataset xml dataset. *Journal of Open Source Software*, 5(46):1979.

Sultan Alrowili and Vijay Shanker. 2021. BioM-transformers: Building large biomedical language models with BERT, ALBERT and ELECTRA. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 221–227, Online. Association for Computational Linguistics.

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L. Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, Sergio Ortiz Rojas, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Elsa Sarrías, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 2020. ParaCrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Peng-Jen Chen, Jiajun Shen, Matt Le, Vishrav Chaudhary, Ahmed El-Kishky, Guillaume Wenzek, Myle Ott, and Marc'Aurelio Ranzato. 2019a. Facebook ai's WAT19 myanmar-english translation task submission. *CoRR*, abs/1910.06848.

Peng-Jen Chen, Jiajun Shen, Matthew Le, Vishrav Chaudhary, Ahmed El-Kishky, Guillaume Wenzek, Myle Ott, and Marc'Aurelio Ranzato. 2019b. Facebook AI's WAT19 Myanmar-English translation task submission. In *Proceedings of the 6th Workshop on Asian Translation*, pages 112–122, Hong Kong, China. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

Louise Deléger, Magnus Merkel, and Pierre Zweigenbaum. 2009. Translating medical terminologies through word alignment in parallel text corpora. *Journal of biomedical informatics*, 42 4:692–701.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Long Doan, Linh The Nguyen, Nguyen Luong Tran, Thai Hoang, and Dat Quoc Nguyen. 2021. Phomt: A high-quality and large-scale benchmark dataset for vietnamese-english machine translation. *CoRR*, abs/2110.12199.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: A resource for disease name recognition and concept normalization. *Journal of Biomedical Informatics*, 47:1–10.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. 2020. CharacterBERT: Reconciling ELMo and BERT for word-level open-vocabulary representations from characters. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6903–6915, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation. *CoRR*, abs/2010.11125.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Çelebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2021. Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, 22:107:1–107:48.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. Codebert: A pre-trained model for programming and natural languages. *CoRR*, abs/2002.08155.

Carol Friedman, Pauline Kra, and Andrey Rzhetsky. 2002. Two biomedical sublanguages: a description based on the theories of zellig harris. *Journal of Biomedical Informatics*, 35(4):222–235. Sublanguage - Zellig Harris Memorial.

Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O.K. Li. 2018. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 344–354, New Orleans, Louisiana. Association for Computational Linguistics.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *CoRR*, abs/2007.15779.

Thanh-Le Ha, Van-Khanh Tran, and Kim-Anh Nguyen. 2020. Goals, challenges and findings of the VLSP 2020 English-Vietnamese news translation shared task. In *Proceedings of the 7th International Workshop on Vietnamese Language and Speech Processing*, pages 99–105, Hanoi, Vietnam. Association for Computational Lingustics.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2019. Revisiting self-training for neural sequence generation. *CoRR*, abs/1909.13788.

María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, and Thierry Declerck. 2013. The ddi corpus: An annotated corpus with pharmacological substances and drug–drug interactions. *Journal of Biomedical Informatics*, 46(5):914–920.

Antonio Jimeno-Yepes, Élise Prieur, and Aurélie Névéol. 2012. Combining medline and publisher data to create parallel corpora for the automatic translation of biomedical text. *BMC Bioinformatics*, 14:146 – 146.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *CoRR*, abs/1909.06146.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. 2021. Mimic-iv.

Alistair Johnson, Tom Pollard, Lu Shen, Li-wei Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Celi, and Roger Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3:160035.

Kamal raj Kanakarajan, Suriyadeepan Ramamoorthy, Vaidheeswaran Archana, Soham Chatterjee, and Malaikannan Sankarasubbu. 2019. Saama research at MEDIQA 2019: Pre-trained BioBERT with attention visualisation for medical natural language inference. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 510–516, Florence, Italy. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *CoRR*, abs/1901.08746.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *CoRR*, abs/2001.08210.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Minh-Thang Luong and Christopher Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 76–79, Da Nang, Vietnam.

S M Meystre, G K Savova, K C Kipper-Schuler, and J F Hurdle. 2008. Extracting information from textual documents in the electronic health record: a review of recent research. *Yearbook of medical informatics*, pages 128–44.

Nguyen Minh, Vu Hoang Tran, Vu Hoang, Huy Duc Ta, Trung Huu Bui, and Steven Quoc Hung Truong. 2022. Vihealthbert: Pre-trained language models for vietnamese in health text mining. In *Proceedings of the Language Resources and Evaluation Conference*, pages 328–337, Marseille, France. European Language Resources Association.

Muhammad Umair Nasir and Innocent Mchechesi. 2022. Geographical distance is the new hyperparameter: A case study of finding the optimal pre-trained language for English-isiZulu machine translation. In *Proceedings of the Workshop on Multilingual Information Access (MIA)*, pages 1–8, Seattle, USA. Association for Computational Linguistics.

Aurélie Névéol, Antonio Jimeno Yepes, Mariana Neves, and Karin Verspoor. 2018. Parallel corpora for the biomedical domain. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Mariana Neves, Antonio Jimeno Yepes, and Aurélie Névéol. 2016. The scielo corpus: a parallel corpus of scientific publications for biomedicine. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2942–2948, Portorož, Slovenia. European Language Resources Association (ELRA).

Chinh Ngo, Trieu H. Trinh, Long Phan, Hieu Tran, Tai Dang, Hieu Nguyen, Minh Nguyen, and Minh-Thang

Luong. 2022. Mtet: Multi-domain translation for english and vietnamese.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. Phobert: Pre-trained language models for vietnamese. *CoRR*, abs/2003.00744.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Benjamin E. Nye, Junyi Jessy Li, Roma Patel, Yinfei Yang, Iain James Marshall, Ani Nenkova, and Byron C. Wallace. 2018. A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature. *CoRR*, abs/1806.04185.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of BERT and elmo on ten benchmarking datasets. *CoRR*, abs/1906.05474.

Long Phan, Hieu Tran, Hieu Nguyen, and Trieu H. Trinh. 2022. Vit5: Pretrained text-to-text transformer for vietnamese language generation.

Long N. Phan, James T. Anibal, Hieu Tran, Shaurya Chanana, Erol Bahadroglu, Alec Peltekian, and Grégoire Altan-Bonnet. 2021a. Scifive: a text-to-text transformer model for biomedical literature. *CoRR*, abs/2106.03598.

Long N. Phan, Hieu Tran, Daniel Le, Hieu Nguyen, James T. Anibal, Alec Peltekian, and Yanfang Ye. 2021b. Cotext: Multi-task learning with code-text transformer. *CoRR*, abs/2105.08645.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Alexey Romanov and Chaitanya Shivade. 2018. Lessons from natural language inference in the clinical domain.

Jiajun Shen, Peng-Jen Chen, Matt Le, Junxian He, Jiatao Gu, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2021. The source-target domain mismatch problem in machine translation. *ArXiv*, abs/1909.13151.

Ye Kyaw Thu, Win Pa Pa, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Introducing the Asian language treebank (ALT). In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1574–1578, Portorož, Slovenia. European Language Resources Association (ELRA).

Robert Tinn, Hao Cheng, Yu Gu, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Fine-tuning large neural language models for biomedical natural language processing. *CoRR*, abs/2112.07869.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Cuijun Wu, F. Xia, Louise Deléger, and Imre Solti. 2011. Statistical machine translation for biomedical text: are we there yet? *AMIA ... Annual Symposium proceedings. AMIA Symposium*, 2011:1290–9.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2022. Linkbert: Pretraining language models with document links.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. *CoRR*, abs/2005.08314.

Hongyi Yuan, Zheng Yuan, Ruyi Gan, Jiaxing Zhang, Yutao Xie, and Sheng Yu. 2022. BioBART: Pretraining and evaluation of a biomedical generative language model. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 97–109, Dublin, Ireland. Association for Computational Linguistics.

A Results for En-Vi Translation with Self-training

We explore a more drastic measure to expand the amount of data in the biomedical domain by self-training. We also explore how a large expansion in the number of biomedical bitexts affects the performance of our model on other domains such as Law, Religion, and News by using the MTet multi-domain test set (Ngo et al., 2022).

Table 5: BLEU Scores Results for En-Vi Translation on MTet Multi-domain Test Set

Model	**MTet Multi-domain Test Set**			
	Medical	News	Religion	Laws
SOTA	38.69	**51.47**	**41.44**	36.43
Ours	**45.61**	51.003	40.68	**39.51**

Notes: The best BLEU scores are in bold. The state-of-the-art (SOTA) model and MTet dataset are from Ngo et al. (2022); Our model trained with self-training approach on an extra 1M En-Vi synthetic biomedical abstracts is discussed in Section 3

The improvement is not evident across all domains when tested on a diverse domain test set (MTet). For example, while there are notable improvements in the Medical and Law domain, the model performs worse in the Religion and News domains. This can be attributed to the context representation of biomedical Pubmed Abstract data. Scientific abstracts tend to be more formal and academic for knowledgeable audiences with more domain expertise. Therefore, training on such data allows the Machine Translation system to perform better not only on the trained domain (Medical) but also on other formally presented domains, such as Law, while at the same time performing slightly worse on other domains (News and Religion).

Syntax-guided Neural Module Distillation to Probe Compositionality in Sentence Embeddings

Rohan Pandey
Language Technologies Institute
Carnegie Mellon University
rspandey@cs.cmu.edu

Abstract

Past work probing compositionality in sentence embedding models faces issues determining the causal impact of implicit syntax representations. Given a sentence, we construct a neural module net based on its syntax parse and train it end-to-end to approximate the sentence's embedding generated by a transformer model. The distillability of a transformer to a Syntactic NeurAl Module Net (SynNaMoN) then captures whether syntax is a strong causal model of its compositional ability. Furthermore, we address questions about the geometry of semantic composition by specifying individual SynNaMoN modules' internal architecture & linearity. We find differences in the distillability of various sentence embedding models that broadly correlate with their performance, but observe that distillability doesn't considerably vary by model size. We also present preliminary evidence that much syntax-guided composition in sentence embedding models is linear, and that non-linearities may serve primarily to handle non-compositional phrases.

1 Introduction

The principle of semantic compositionality suggests that the meaning of a sentence should derive from its subconstituents in a regular, structured fashion (Montague, 1970). In recent years, transformers (Vaswani et al., 2017) have become effective at producing sentential meaning representations useful for downstream tasks such as Natural Language Inference, Image-Text Matching, and Document Classification (Conneau et al., 2017; Radford et al., 2021). However, it has famously been conjectured that "You can't cram the meaning of a whole %&!$# sentence into a single $&!#* vector" (Mooney, 2014). Since recent models do appear to capture sentence meaning effectively, one wonders how they compose arbitrarily many word meanings together such that their relational structure is captured in a single, fixed-dimensional sentence embedding.

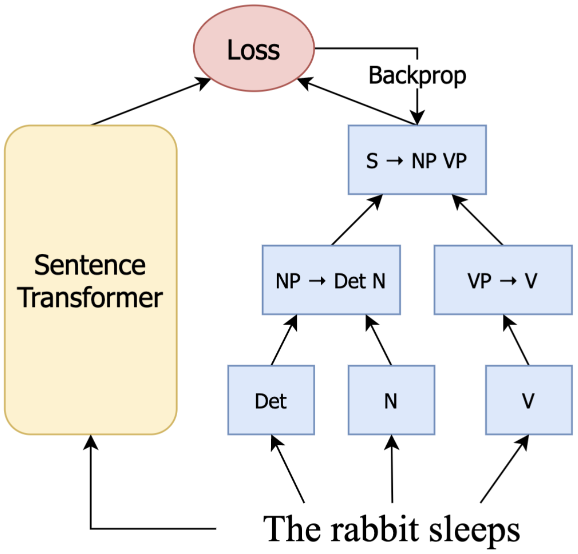


Figure 1: Distilling a transformer to a neural module net structured by the sentence's syntax

Much work has sought to probe these models for syntax representations and their causal relevance to embedding output. Conneau et al. (2018) train linear probes to determine if models encode syntactic features like tree distance and depth (Krasnowska-Kieraś and Wróblewska, 2019; Hewitt and Manning, 2019). One line seeks out direct mappings between neural representations and tree structures (McCoy et al., 2018; Chrupała and Alishahi, 2019; Jawahar et al., 2019; Soulos et al., 2020; Murty et al., 2023). Other work raises methodological issues with probing (Eger et al., 2019; Zhu and Rudzicz, 2020) such as choice of formalism (Kuznetsov and Gurevych, 2020) and semantic entanglement (Maudslay and Cotterell, 2021). Ravichander et al. (2021) raise the possibility that probing may identify causally un-used features; Tucker et al. (2021) partly address this concern to show that some syntactic features are causally relevant. Another line of work explores the geometry of semantic representations (Reif et al., 2019; Hernandez and Andreas,

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3143–3149
May 2-6, 2023 ©2023 Association for Computational Linguistics

2021) and the linearity (Barančíková and Bojar, 2019) of syntactic analogies (Zhu and de Melo, 2020).

Rather than directly analyze sentence embedding models, Neural Module Nets (Andreas et al., 2016b) seek to improve compositionality by modularizing semantic functions. We see this effort as ultimately similar to probing for structural representations since the former explores whether explicit structure improves performance and the latter explores whether performant models implicitly learn structure. Geiger et al. (2021) discovers logical tree causal structures in BERT and Wu et al. (2021) then guides model distillation using this structure.

Our work builds on these findings by strictly taking syntax as the causal structure of sentential semantics and linearity as the geometry of syntax-guided composition; we conduct experiments to test the distillability of transformer-based sentence embedding models to a Syntactic NeurAl Module Net (SynNaMoN), an architecture we introduce that implements these two priors. The extent to which a model can be distilled to a SynNaMoN tells us about its internal syntax representations & compositional ability.

2 Methods

2.1 Syntactic Neural Module Net

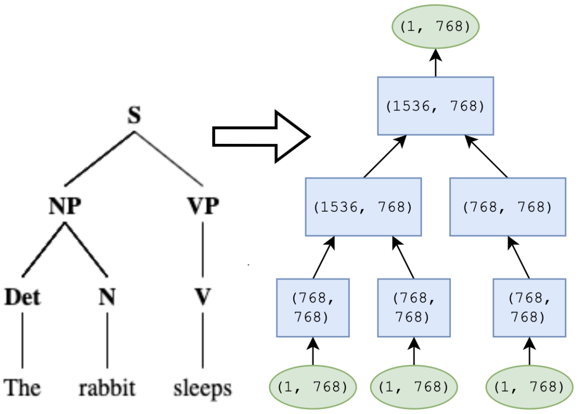


Figure 2: Constructing a sentence's SynNaMoN from its syntax tree; module input and output dimensionalities labeled on the right.

Unlike prior work (Andreas et al., 2016a; Cirik et al., 2018), SynNaMoN modules don't approximate high-level objectives like 'Find' or 'Count' but rather correspond to specific syntactic rules like 'S → NP VP' and 'NP → DT JJ NN'. Each module receives an input of dimensionality $(1, N * D)$ where N is the number of constituents on the syntax rule's right-hand side, and D is the dimensionality of the embedding space (768)—in other words, the input embeddings are concatenated. Though computationally more expensive, concatenation enables the module to learn an arbitrary function over the inputs rather than restricting it to a function over their sum or mean; this enables the module to converge on its ideal composition function which is likely not invariant under summation or averaging. Finally, though our implementation of SynNaMon includes 'part-of-speech' modules at the bottom of the parse tree, one could conceivably remove this bottom layer with the hypothesis that the word embeddings already capture part-of-speech information.

2.2 Internal Module Architecture

To explore the geometry of semantic composition under syntax, we implement 3 module architectures: a linear layer (**Linear**), a linear layer + a ReLU activation (**Nonlin**), and a linear layer + ReLU + another linear layer (**Double**). We explore these 3 architectures to see whether syntax is enough of an inductive bias to linearly approximate sentence embeddings, or if adding non-linearities and additional layers considerably improves performance. The extent to which adding parameters improves our approximation of the teacher model beyond the syntactic structure alone could reveal how much isn't captured by this inductive bias.

2.3 Linguistic Formalism

We choose to use the Transformational Grammar presented by Penn Treebank (Marcus et al., 1994), but in principle any Constituency Grammar could be easily used with SynNaMoN, and Dependency Grammars can be adapted with some effort. Since prior work has shown how the choice of linguistic formalism can significantly influence probing results (Kuznetsov and Gurevych, 2020), we float the possibility of such an effect being at play in this work as well. If a student SynNaMoN fails to capture much of the teacher embedding model, perhaps it isn't because of the teacher's non-compositional causal structure, but rather because the formalism used to structure the SynNaMoN is inadequate. Indeed, recent state-of-the-art neural approaches to syntax parsing have learned grammatical tagsets that often differ starkly from human-produced syntactic theories (Kitaev et al., 2022). We leave these problems to future work which may explore the exciting possibility that certain linguistic formalisms (perhaps even semantic rather than syntactic) are better proxies for a model's compositional structure than others.

3 Experiments

Our main experiment runs 5 sentence embedding models (BERT-base (Devlin et al., 2019), MP-Net (Song et al., 2020), GTR-T5-base, GTR-T5-large, and GTR-T5-xl (Ni et al., 2021)) on 3 SynNaMoNs with differing internal architectures (Linear, Nonlin, Double; see Sec. 2.2). For BERT-base, we extract input word embeddings for each token and use the CLS token as the sentence embedding as is common practice. For the other 4 models, we encode each token alone to serve as its embedding and use the output as the sentence embedding. When words are encoded as more than 1 token, we compute the mean across the subtokens to serve as its word embedding.

In order to heuristically select a learning rate, 5 training runs were conducted with SynNaMoNs optimizing for BERT-base, and learning rate manually set at increments between 10^{-5} and 10^{-3}. We finally chose a rate of 5×10^{-5}, but recognize from results that optimal learning rate will likely vary by teacher model & SynNaMoN internal architecture. Analysis would best be reported on the optimal scores achieved by a SynNaMoN after hyperparameter tuning, but due to compute restrictions (1 NVIDIA K80 GPU with 12GB of RAM), this was unfeasible.

Additionally, due to the number of modules (originally 900, each with 1M parameters on average), we encountered frequent out-of-memory errors both on CPU & GPU. Since each module corresponds to a syntax rule and is initialized upon encountering the rule in the dataset, we constrained our data to minimize the number of modules needed.

Specifically, we first constrained our trees to those of height 4 & 5 (n=16492) in PTB, and then further constrained the trees to those that use a subset of the 300 most common production rules among them. This resulted in 1494 trees, from which we generated a train-validation split of 1250-244. Furthermore, we ensured that all the productions present in trees of the validation split were also included amongst trees in the training split. All this finally resulted in 273 production rules present in our dataset, and the instantiation of 273 modules.

4 Results

In Tab. 1, we present scores for all 5 sentence embedding models across the 3 SynNaMon architectures. We compute the average MSE between sentence embeddings in the complete dataset for each model and divide each model's MSE loss by this mean distance to normalize results. The normalized scores we present may intuitively be seen as the portion of variance in a model's sentence embeddings that a SynNaMoN fails to explain. From a probing perspective, the lower a model's score, the more it can be causally approximated by composition along syntactic lines.

Sent. Emb. Model	Linear	Nonlin	Double
BERT-base-CLS	.765	4.17	.625
MP-Net-base	.606	.963	.538
GTR-T5-base	.541	.844	.499
GTR-T5-large	.550	.898	.502
GTR-T5-xl	**.536**	**.775**	**.498**

Table 1: Best validation MSE loss of sentence embedding models on each SynNaMoN probe, normalized by chance-level MSE between embeddings

First, notice that GTR-T5-xl outperforms all the other models across all the SynNaMoN architectures. This seems to confirm our expectation that larger models should produce more compositional sentence embeddings. However, GTR-T5-xl only marginally outperforms other sizes of GTR-T5 (except on Nonlin, for which it does far better), suggesting that size actually isn't a significant factor in compositionality. The lower performance of GTR-T5-large further corroborates this, but considering its anomalously lower average embedding MSE, the issue requires more work. The fact that GTR-T5 models all display high compositionality despite variance in size suggests something about their architecture or training approach is important—perhaps the representational bottleneck.

All 3 GTR-T5 models perform better than MP-Net, which in turn outperforms BERT CLS. This first fact is slightly surprising considering that on standard sentence representation tasks (Reimers and Gurevych, 2019), MP-Net (63.30) marginally outperforms all GTR-T5 (base: 59.40, large: 62.38, xl: 62.88) models. Evaluation of these sentence embedding models on large-scale, human-interpretable compositionality tasks may reveal that GTR-T5 does indeed produce better compositional representations than MP-Net. Although BERT's

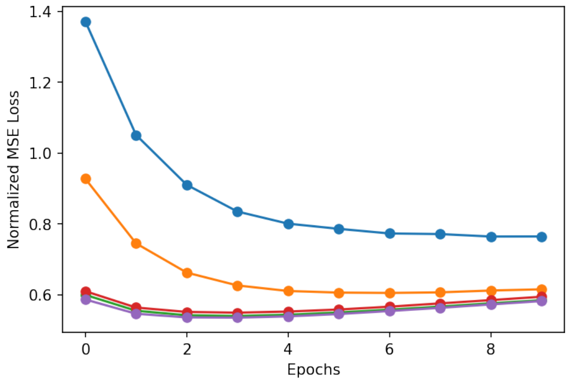


Figure 3: Normalized validation learning curves for Linear SynNaMoN on sentence embedding models (blue: BERT, orange: MP-Net, red: GTR-T5-large, green: GTR-T5-base, purple: GTR-T5-xl)

CLS token embedding is widely used for sentence representation, these results show that it fails to capture nearly as much compositional information as more targeted sentence embedding models.

Next, observe that although distilling to a Double SynNaMoN is intuitively easier than to a Linear SynNaMoN due to increased parameterization, there aren't always major improvements in distillability. It is possible that the geometric expressivity of the Double SynNaMoN will kick in with scaling of training data, but we hypothesize that this Double score will still approach a limit for all sentence embedding models. This is because syntax only describes a subset of sentence meaning, and the strictness of SynNaMoN's structure prevents this non-compositional component from being learned.

For example, a strictly syntactic compositional interpretation of "village on the river", would represent the village as being literally on top of the river since this is the semantic geometry learned for syntactic structures of the form "NP on NP". A SynNaMoN that includes non-linearities may better learn the geometry of this non-literal "on" relation, but a transformer model would best learn to handle non-compositional phrases due to its lack of strict syntactic constraints. Our broader takeaway from comparing Linear & Double scores is that much composition along syntactic lines is linear, and non-linearities in transformers primarily serve a purpose other than syntax-guided composition—perhaps in handling non-compositional phrases.

On a less theoretical note, we observe that our learning curves for Linear SynNaMoN on GTR-T5 (Fig. 3) are clearly overfitted due to fixing hyperparameters as mentioned in Sec. 3. We remediate this issue in Tab. 1 by reporting the best scores (minimum across epochs) for each learning curve. Since we want to construct the best possible SynNaMoN for a transformer model (as this most accurately reveals the transformer's distillable compositional ability), scores could be slightly improved with further hyperparameter tuning.

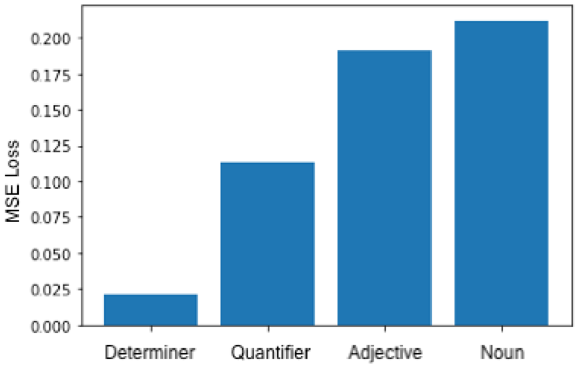


Figure 4: Generalization ability of Determiner Phrase module's linear geometry varies by part-of-speech

4.1 Analysis

Finally, we explore a single module to determine whether its compositional geometry meets intuitive notions of semantic generalization. Due to methodological difficulties with assessing a single module extracted from our end-to-end training paradigm, we train a Linear module for 'NP → Det N' on its own. Determiner-noun composition intuitively lies on a spectrum with adjective-noun composition on the other end and quantifier-noun composition in between. While quantifiers like 'some' and 'all' seem more like determiners, other quantifiers like 'several' and 'twelve' appear more comparable to adjectives like 'swarming' and 'grouped'. Intuitively then, we should expect the geometry of quantifier-noun composition to be intermediate to determiners and adjectives.

And this behavior is precisely what we find in our 'NP → Det N' module. Since it's trained on determiners, it obviously has the lowest MSE for this part-of-speech; we include noun-noun pairs (e.g. 'tree cow') as a control. As seen in Fig. 4, the module generalizes to quantifiers intermediately to determiners and adjectives. This demonstrates how SynNaMoN modules may enable interesting analyses of the compositional geometry of syntactic operations in sentence embedding models.

5 Conclusion

The human ability to apprehend the unitary meaning of a sentence corresponds to a neural model's ability to construct compositional sentence embeddings. In this work, we introduced Syntactic Neural Module Nets and used it in a distillation approach to assess how well syntax explains the sentential semantics computed by a transformer model. We showed that some models are more compositional by this metric, syntax-guided composition is largely linear, and modules learn composition functions that correspond to our semantic intuition.

Future work could explore this approach's alignment with other compositionality metrics and the non-compositional semantics left uncaptured by SynNaMoNs. We are also interested in how SynNaMoNs of different linguistic formalisms vary in distillability, as well as other potential use cases of SynNaMoNs beyond probing.

6 Limitations & Ethics Statement

Since longer sentences have more complex syntax, they require more modules on GPU and can run into out-of-memory issues. However, there is a hard upper-bound on total number of modules since there are limited syntax rules in the grammar. In addition, we may never need to train on long sentences if all modules can be effectively trained on short sentences and then generalize compositionally.

As an approach to probing language models, SynNaMoN contributes to an ethical NLP vision that seeks to address how models learn human biases that have societal effects from corpus data. Understanding syntax representations in models could be important in such a pursuit since some of these bias effects are syntactically mediated. For example, LMs with gender role biases could internally represent these biases as syntactic gender agreement e.g. ‘man’ agrees with ’doctor’ and ‘woman’ agrees with ‘nurse’ (Prates et al., 2020). By understanding the causal structure of sentential semantics in LMs, we can better disentangle syntax from spurious correlations transmitted by societal structures.

7 Acknowledgements

I am grateful for the advice provided by Tom McCoy, Emmy Liu, Chris Potts, and Paul Smolensky on questions of compositionality & probing. I also appreciate the intellectual company of Aryaman Arora & Justus Mattern at the onset of this project, and to Uncle Ike for providing us with a truly inspirational set of trees.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016a. Learning to compose neural networks for question answering. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1545–1554.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016b. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Petra Barančíková and Ondřej Bojar. 2019. In search for linear relations in sentence embedding spaces. *arXiv preprint arXiv:1910.03375*.

Grzegorz Chrupała and Afra Alishahi. 2019. Correlating neural and symbolic representations of language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2952–2962.

Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency. 2018. Using syntax to ground referring expressions in natural images. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136.

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

Steffen Eger, Andreas Rücklé, and Iryna Gurevych. 2019. Pitfalls in the evaluation of sentence embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 55–60.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. 2021. Causal abstractions of neural networks. *Advances in Neural Information Processing Systems*, 34:9574–9586.

Evan Hernandez and Jacob Andreas. 2021. The low-dimensional linear geometry of contextualized word representations. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 82–93.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Nikita Kitaev, Thomas Lu, and Dan Klein. 2022. Learned incremental representations for parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3086–3095.

Katarzyna Krasnowska-Kieraś and Alina Wróblewska. 2019. Empirical linguistic study of sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5729–5739.

Ilia Kuznetsov and Iryna Gurevych. 2020. A matter of framing: The impact of linguistic formalism on probing results. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 171–182.

Mitch Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Rowan Hall Maudslay and Ryan Cotterell. 2021. Do syntactic probes probe syntax? experiments with jabberwocky probing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 124–131.

R Thomas McCoy, Tal Linzen, Ewan Dunbar, and Paul Smolensky. 2018. Rnns implicitly implement tensor product representations. *arXiv preprint arXiv:1812.08718*.

Richard Montague. 1970. English as a formal language.

Raymond J Mooney. 2014. Semantic parsing: Past, present, and future. In *Presentation slides from the ACL Workshop on Semantic Parsing*.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. 2023. Characterizing intrinsic compositionality in transformers with tree projections. In *International Conference on Learning Representations*.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. 2021. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.

Marcelo OR Prates, Pedro H Avelar, and Luís C Lamb. 2020. Assessing gender bias in machine translation: a case study with google translate. *Neural Computing and Applications*, 32(10):6363–6381.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2021. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3363–3377.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. *Advances in Neural Information Processing Systems*, 32.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

Paul Soulos, R Thomas McCoy, Tal Linzen, and Paul Smolensky. 2020. Discovering the compositional structure of vector representations with role learning networks. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 238–254.

Mycal Tucker, Peng Qian, and Roger Levy. 2021. What if this modified that? syntactic interventions with counterfactual embeddings. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 862–875, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zhengxuan Wu, Atticus Geiger, Josh Rozner, Elisa Kreiss, Hanson Lu, Thomas Icard, Christopher Potts, and Noah D Goodman. 2021. Causal distillation for language models. *arXiv preprint arXiv:2112.02505*.

Xunjie Zhu and Gerard de Melo. 2020. Sentence analogies: Linguistic regularities in sentence embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3389–3400, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zining Zhu and Frank Rudzicz. 2020. An information theoretic view on selecting linguistic probes. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9251–9262, Online. Association for Computational Linguistics.

Closed-book Question Generation via Contrastive Learning

Xiangjue Dong[1], **Jiaying Lu**[2]*, **Jianling Wang**[1]*, **James Caverlee**[1]
[1] Texas A&M University, [2] Emory University
`{xj.dong, jlwang, caverlee}@tamu.edu, jiaying.lu@emory.edu`

Abstract

Question Generation (QG) is a fundamental NLP task for many downstream applications. Recent studies on *open-book* QG, where supportive answer-context pairs are provided to models, have achieved promising progress. However, generating natural questions under a more practical *closed-book* setting that lacks these supporting documents still remains a challenge. In this work, we propose a new QG model for this closed-book setting that is designed to better understand the semantics of long-form abstractive answers and store more information in its parameters through contrastive learning and an answer reconstruction module. Through experiments, we validate the proposed QG model on both public datasets and a new WikiCQA dataset. Empirical results show that the proposed QG model outperforms baselines in both automatic evaluation and human evaluation. In addition, we show how to leverage the proposed model to improve existing question-answering systems. These results further indicate the effectiveness of our QG model for enhancing *closed-book* question-answering tasks.

1 Introduction

Question Generation (QG) has a wide range of applications, such as generating questions for exams (Jia et al., 2021; Lelkes et al., 2021; Dugan et al., 2022) or children's story books (Zhao et al., 2022; Yao et al., 2022), recommending questions for users in a dialogue system (Shukla et al., 2019; Laban et al., 2020), improving visual (Li et al., 2018; Lu et al., 2022) or textual question-answering tasks (Duan et al., 2017; Lewis et al., 2019a; Zhang and Bansal, 2019; Sultan et al., 2020; Lyu et al., 2021), asking clarification questions (Rao and Daumé III, 2019; Yu et al., 2020; Ren et al., 2021), and generating queries for SQL (Wu et al., 2021) or multimodal documents (Kim et al., 2021).

Previous works on QG are mainly under the *open-book* setting, which aims to generate questions based on factoid or human-generated short answers under the assumption that there is access to external knowledge like retrieved documents or passages (Du et al., 2017; Zhao et al., 2018; Kim et al., 2019; Fei et al., 2021). After Roberts et al. (2020) demonstrated that feeding a large pre-trained model input questions alone *without* any external knowledge can lead to competitive results with retrieval-based methods on open-domain question-answering benchmarks, there is an increasing interest in the *closed-book* setting. This closed-book setting is appealing in practice and can be widely applied, e.g., in question suggestion (Laban et al., 2020; Yin et al., 2021), query recommendation (Kim et al., 2021), and other practical settings where extensive external knowledge is unavailable.

However, generating questions without access to such external knowledge is challenging for two key reasons. First, without access to retrieved documents (or passages), simple open-domain strategies like basing the answers on these documents (or passages) are not possible under the closed-book setting. Instead, models must rely on the answers alone. Second, the data used by most of the closed-book works (Lewis et al., 2021; Wang et al., 2021) are variants of existing open-domain datasets, e.g., SQuAD (Rajpurkar et al., 2018), TriviaQA (Joshi et al., 2017), WebQuestions (Berant et al., 2013) that ignore the answer-related passages. These answers in open-book works are usually short, e.g., entities, and easier to be remembered by the language model and stored in the parameters of the model than long-form answers. Thus, this leads to our motivating research question – *How can we empower a QG model to better understand the semantics of long-form abstractive answers and store more information in its parameters?*

To tackle the aforementioned challenges existing

---

* Equal Contribution

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3150–3162
May 2-6, 2023

in the closed-book setting, this paper proposes a new QG model with two unique characteristics: (i) a contrastive learning loss designed to better understand the semantics of the answers and the semantic relationship between answers and ground-truth questions at a contextual-level; and (ii) an answer reconstruction loss designed to measure the answerability of the generated question. Contrastive learning has shown promising results in many NLP tasks, e.g., (Giorgi et al., 2021; Gao et al., 2021; Yang et al., 2021) and aligns positive pairs better with available supervised signals (Gao et al., 2021); here we show how to learn question representations by distinguishing features of correct question-answer pairs from features of incorrectly linked question-answer pairs. Further, to ensure the generated questions are of good quality and can be answered by the answer that is used for question generation, we frame the model as a generation-reconstruction process (Cao et al., 2019; Zhu et al., 2020), by predicting the original answers given the generated questions by a pre-trained seq2seq model. In addition, we introduce a new closed-book dataset with long-form abstractive answers – WikiCQA – to complement existing datasets like GooAQ (Khashabi et al., 2021) and ELI5 (Fan et al., 2019) and show how to leverage our model to generate synthetic data to improve closed-book question-answering tasks.

Through experiments, we find that the proposed QG model shows improvement through both automatic and human evaluation metrics on WikiCQA and two public datasets. Compared to the baseline, the proposed QG framework shows an improvement of up to 2.0%, 2.7%, and 1.8% on the ROUGE-L score on WikiCQA, GooAQ-S, and ELI5, respectively, and 1.3% and 2.6% in terms of relevance and correctness. Furthermore, we leverage the QG framework to generate synthetic QA data from WikiHow summary data and pre-train a closed-book QA model on it in both an unsupervised and semi-supervised setting. The performance is evaluated on both seen (WikiCQA) and unseen (GooAQ-S, ELI5) datasets. We find consistent improvements across these datasets, indicating the QG model's effectiveness in enhancing closed-book question-answering tasks.

In conclusion, our contributions can be summarized as follows:

- We propose a contrastive QG model, which to our knowledge is the first work to explore contrastive learning for QG under a closed-book setting.
- The proposed model outperforms baselines on three datasets. The human evaluation also indicates that the questions generated by our model are more informative compared to other baselines.
- We leverage the QG model as a data augmentation strategy to generate large-scale QA pairs. Consistent improvements shown on both seen datasets and unseen datasets indicate the QG model's effectiveness in enhancing closed-book question-answering tasks.

2 Related Work

Many previous works on QG are under the open-book setting, which takes factoid short answers (Rajpurkar et al., 2016) or human-generated short answers (Kočiský et al., 2018) with the corresponding passages to generate questions (Zhang et al., 2021). Early approaches for question generation rely on rule-based methods (Labutov et al., 2015; Khullar et al., 2018). To bypass handcrafted rules and sophisticated pipelines in QG, Du et al. (2017) introduce a vanilla RNN-based sequence-to-sequence approach with an attention mechanism. The recently proposed pre-trained transformer-based frameworks (Lewis et al., 2020; Raffel et al., 2020) also improve the performance of QG. In addition, Sultan et al. (2020) shows that the lexical and factual diversity of QG provides better QA training. However, their success can not directly adapt to the closed-book setting, where the model is supposed to generate questions solely relying on answers. In this work, we explore the widely applicable closed-book QG setting, which is still under-explored.

Contrastive Learning aims to pull semantically similar neighbors close and push non-neighbors apart. It has achieved great success under both supervised and unsupervised settings. In pioneer works, the contrastive loss function (Hadsell et al., 2006; Chopra et al., 2005) has been proposed as a training objective in deep metric learning considering both similar and dissimilar pairs. Recently, Chen et al. (2020) proposes the SimCLR framework to learn useful visual representations. Viewing contrastive learning as dictionary look-up, He et al. (2020) present Momentum Contrast (MoCo) to build dynamic dictionaries for contrastive learning. Some works apply contrastive learning into the NLP domain to learn better sentence representations (Giorgi et al., 2021; Gao et al., 2021). In addition, contrastive learning has been applied in multilingual neural machine translation (Pan et al., 2021), abstractive summarization (Liu and Liu, 2021), and multi-document question generation (Cho et al., 2021). The recent most relevant work is (Yang et al., 2021), where they design two contrastive losses for paraphrase generation. In this work, we adopt contrastive learning for improving representation learning in question generation under a closed-book setting.

3 Proposed Approach

To answer our research question – How can we empower a QG model to better understand the semantics of long-form abstractive answers and store more information in its parameters? – we propose a closed-book QG model, which generates questions directly without access to external knowledge. Formally, given an answer sentence x, a closed-book QG engine generates a natural question y. Figure 1 illustrates an overview of the proposed QG framework, which consists of three parts: question generation, contrastive learning, and answer reconstruction. The framework is optimized with the joint losses from these three parts simultaneously.

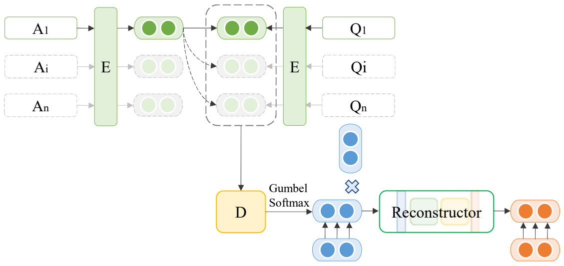


Figure 1: An overview of the proposed closed-book QG framework, which consists of three parts: contrastive learning, question generation, and answer construction. A_i: represents answer i; Q_i represents question i.

3.1 Question Generation

We first focus on question generation through a sequence-to-sequence architecture which consists of an encoder and a decoder (Sutskever et al., 2014; Vaswani et al., 2017). The encoder takes an input sequence of source words $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and maps it to a sequence of continuous representations $\mathbf{z} = (z_1, z_2, \ldots, z_n)$. Then, the decoder takes $\mathbf{z}$ and generates a sequence of target words $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ at a time. The closed-book QG task is defined as finding $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}), \quad (1)$$

where $P(\mathbf{y}|\mathbf{x})$ is the conditional likelihood of the predicted question sequence $\mathbf{y}$ given answer $\mathbf{x}$.

$$P(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{T} p(y_t|\mathbf{y}_{<t}, \mathbf{x}), \quad (2)$$

Given the answer-question pairs, the training objective of the generation part in the proposed framework is to minimize the Negative Log-Likelihood (NLL) of the training data,

$$\mathcal{L}_{qg} = -\sum_{i=1}^{N} \log p(q_i|A), \quad (3)$$

where q_i is the i-th token in the generated question and A is the answer.

A naive question generation model will generate questions based on answers but lacks a rich model of the semantics of answers nor can it guarantee the generated questions have a semantic relationship with the answers. Intuitively, an encoded answer should be similar to its question and dissimilar to others. In addition, the generated question should be able to be answered by the answers. Hence, this motivates the following contrastive learning and answer reconstruction modules.

3.2 Contrastive Learning

Contrastive learning aims to pull positive pairs and push apart negative pairs to learn effective representations. Further, the supervised signals can produce better sentence embeddings by improving alignment between positive pairs (Chen et al., 2020). An effective QG model should be able to understand the semantics of the answers and the semantic relationship with the ground-truth questions. Especially, the encoded answer should have semantic similarity with its ground-truth question and dissimilarity with other questions. Thus, aiming to learn a similarity function that pulls the distance between the answer sequence representation and its ground-truth question sequence representation closer, we design a contrastive loss in the representation space. Specifically, given a positive pair $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$, where $\mathbf{x}_i$ and $\mathbf{y}_i$ are semantically related inputs, the other $2(n-1)$ examples within a mini-batch are treated as negative examples. The training objective for $(\mathbf{x}_i, \mathbf{y}_i)$ is:

$$\mathcal{L}_{cl} = -\log \frac{\exp(\text{sim}(\mathbf{z}_{x_i}, \mathbf{z}_{y_i})/\tau_{cl})}{\sum_{i=1}^{2n} \exp(\text{sim}(\mathbf{z}_{x_i}, \mathbf{z}_{y_i})/\tau_{cl})}, \tag{4}$$

where $\mathbf{z}_{\mathbf{x}_i}$ and $\mathbf{z}_{\mathbf{y}_i}$ is the representation of input x_i and y_i, $\text{sim}(\mathbf{z}_i, \mathbf{z}_j) = \mathbf{z}_i^\top \mathbf{z}_j / \|\mathbf{z}_i\| \, \|\mathbf{z}_j\|$ denotes cosine similarity, and τ_{cl} is a temperature parameter.

In this work, aiming to learn better answer representations and force the encoder to drive representations of correct question-answer pairs closer than representations of incorrect question-answer pairs, we take the ground-truth question as the positive instance of an answer and fine-tune the model parameters based on the contrastive loss function (Eq. 4), where $\mathbf{z}$ is the embedding of the special token `[CLS]` from the transformer encoder, representing the meaning of the entire sentence.

3.3 Answer Reconstruction

The questions that are generated by the model should be of good quality and should also be able to be answered by the answer that is used for question generation. To measure the answerability of the generated question, we design an answer reconstruction module, which uses a pre-trained seq2seq model to predict the original answer given the generated question. The loss is calculated by a negative log-likelihood loss function:

$$\mathcal{L}_{ar} = -\sum_{i=1}^{N} \log p(a_i|Q) \tag{5}$$

where a_i is the i-th token in the answer and Q is the generated question.

A major challenge is that the generated questions from Section 3.1 are not differentiable. Gradients cannot be back-propagated directly. To solve this challenge, we employ the Straight-Through (ST) Gumbel-Softmax for gradient computation (Jang et al., 2017). The ST Gumbel-Softmax is a discrete version of the Gumbel-Softmax and takes different forward and backward paths. In the forward pass, the embedding is discretized by using the argmax, whereas in the backward pass the gradients are computed by the gumbel-softmax (Qader et al., 2019; Lu et al., 2021).

$$y_i = \frac{\exp((\log(p_i) + g_i)/\tau_{gs})}{\sum_{j=1}^{|V|} \exp((\log(p_j) + g_j)/\tau_{gs})}, \tag{6}$$

where τ_{gs} is a temperature parameter and g_i is the Gumbel noise drawn from a uniform distribution $(0, 1)$. In this work, the one-hot embedding from ST Gumbel-Softmax is multiplied with the vocabulary embedding and then fed into the encoder of the pre-trained seq2seq model as the representation of the generated question.

3.4 Overall Loss Function

As a result, all three losses are summed together to provide the overall loss function $\mathcal{L}$ as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{qg} + \lambda_2 \mathcal{L}_{cl} + \lambda_3 \mathcal{L}_{ar}, \tag{7}$$

The weights λ_1,λ_2, and λ_3 are tuneable hyperparameters to balance losses and the final objective is to minimize the overall loss.

The overall structure of the proposed QG framework is presented in Algorithm 1.

Algorithm 1: QG framework.

```
Input: Pre-trained language model $p(q|a)$,
       answer reconstruction model $p(a|q)$
       and answer-question pairs
Output: Question generator $p(q|a)$
for $i \leftarrow 1$ to $Epoch$ do
    $\hat{q} = p(q|a)$
    Compute $\mathcal{L}_{qg}$ via Eq. 3
    /* contrastive learning          */
    $a_i$ = Encoder([CLS] $\oplus$ $a$)[:, 0 :]
    $a_i^+$ = Encoder([CLS] $\oplus$ $q$)[:, 0 :]
    Get $\mathcal{L}_{cl}$ to $(a_i, a_i^+)$ via Eq. 4
    /* answer reconstruction         */
    $\hat{a} = p(a|\hat{q})$
    Compute $\mathcal{L}_{ar}$ via Eq. 5
    Calculate total loss $\mathcal{L}$ via Eq. 7
    Update generator $p(q|a)$ with $\mathcal{L}$
end
return question generator $p(q|a)$
```

4 Experimental Setup

To evaluate the effectiveness of the proposed QG framework, we aim to answer the following research questions (RQ) via experiments: **RQ1**: Can this proposed QG framework improve the performance of the closed-book QG tasks? **RQ2**: Are the generated questions of good quality? That is, are they fluent and relevant to the answer? **RQ3**: Can this QG framework be leveraged as a good resource to generate synthetic data for the QA task? **RQ4**: How much does each component in the framework contribute?

4.1 Dataset

We conduct the experiments on two public datasets – GooAQ-S (Khashabi et al., 2021) and ELI5 (Fan et al., 2019) – and a new dataset we curate called WikiCQA.

GooAQ-S is a sub-sampled dataset from GooAQ which contains three different sub-tasks: short, snippet (i.e., multi-sentence description), and collection response questions (Khashabi et al., 2021). Under the long-form closed-book setting, we adopt the snippet part (i.e., questions with snippet answers) for our experiments as in the original paper. Furthermore, from the paper, the model performance on the snippet task does not vary much when supervised with 200K and 2M training instances. Thus, to improve the experimental efficiency, we take 200k instances from the snippet set and split them based on their original scripts.[1]

ELI5 (Fan et al., 2019) is a widely-used large-scale corpus for long-form question-answering with supporting web documents. In this work, we use the question-answer pairs from the dataset and ignore the supporting context to fit the closed-book setting like (Khashabi et al., 2021). We follow the data split from huggingface.[2]

WikiCQA is a new closed-book long-form QA dataset (Section 4.2) introduced here. It contains 20,202 question-answer pairs and is collected from a wiki-style website to complement existing datasets. We shuffle the data and split it to train, dev, and test sets by the ratio of 80%/10%/10%. Table 1 shows detailed data statistics and splits of these three datasets.

Datasets	**Train**	**Val**	**Test**
WikiCQA	16,162	2,020	2,020
GooAQ-S	200,000	500	498
ELI5	272,634	9,812	24,512

Table 1: Dataset statistics.

4.2 WikiCQA

WikiCQA contains real user QA pairs collected from WikiHow[3]. WikiHow is a wiki-style website featuring over 200K how-to articles. Different from the existing dataset, ELI5, containing questions/answers from the Reddit forum and leveraging evidence queried from the web to help answer the question (Fan et al., 2019) and GooAQ, containing questions from search auto-complete and answers from answer boxes (Khashabi et al., 2021), WikiCQA involves long-form question-answering grounded on WikiHow articles.

Dataset Construction We construct the new dataset by collecting the question-answer pairs from the Q&A section of articles on WikiHow. The questions and answers are related to the specific articles, asked by WikiHow users, and answered by a knowledgeable reader or editor[4]. The questions can not be answered directly by the content of the article. After removing duplicates and question-answer pairs with meaningless answers, 23,037 QA pairs remain. To keep the same format as other existing QA datasets (e.g., ELI5, GooAQ), we discard questions not starting with a question-type word (e.g., what, how). After the dataset processing steps, we arrive at 20,202 question-answer pairs. More details can be found in Appendix A.

In Table 2, we show an example article from Wikihow, which includes the title and summary of an article and question-answer pairs from the corresponding Q&A section. From the example, we can see that answers are abstractive and long-form, written by real users, and not contained within the context passage. These question-answer pairs from the Q&A section are collected in the newly constructed dataset. Thus, this dataset is different from reading comprehension datasets, where the answers are a short text span in context. More comparisons with ELI5 and GooAQ are shown in Appendix B.

Title: How to Prepare a Healthy Meal for Your Pet Dog. **Summary:** To prepare a healthy meal for your dog, choose lean meat with the bones and fat removed, like chicken or beef . . .
Q: What can I feed my dog if I have run out of dog food? **A:** In the short term, any bland human food such as chicken or white fish with rice or pasta is just fine . . . **Q:** How much homemade dog food do you feed your dog? **A:** Great question because it highlights one of the problems of feeding home prepared foods . . . **Q:** What should I not feed my dog? **A:** There are many human foods that are toxic to dogs. Top of the list of foods NOT to give are . . .

Table 2: Question-answer pairs from WikiHow Q&A section. **Q**: question; **A**: answer.

---

[1] `https://github.com/allenai/gooaq/blob/main/experiments/create_splits.py`

[2] `https://huggingface.co/datasets/eli5`

[3] `https://www.wikihow.com`, under an Attribution-Noncommercial-Share Alike 3.0 Creative Commons License.

[4] `https://www.wikihow.com/Use-wikiHow#Reading-and-Learning-from-wikiHow`

4.3 Models & Hyper-parameters

We use BART-base (Lewis et al., 2020), a widely used sequence-to-sequence framework with 12 layers and a hidden size of 1024, as the backbone model. Following previous works (Fan et al., 2019; Khashabi et al., 2021), we finetune it using answers as inputs and questions as outputs as the baselines. To get the answer reconstruction loss (Section 3.3), we use a BART-base model which is fine-tuned on the target QA datasets.

Based on the dataset analysis in Appendix B, we set the maximum sequence length to 128 for the question and 256 for the answer sentence to improve calculation efficiency. We train the models for five epochs with a learning rate of 5×10^{-5} and evaluate the checkpoint for each epoch. We select the checkpoint with the highest ROUGE-L score on the validation set and report its corresponding score on the test set. We run each model three times and record the average scores. After performing manual hyper-parameter searching, we set loss parameters λ_1, λ_2, and λ_3 in Equation 7 to 1.0, 0.1, 0.1, respectively, which give the best ROUGE-L score on validation set. The temperature of contrastive loss τ_{cl} is set to 0.3. The experiments are run on 1 NVIDIA Tesla V100 GPU. The training time takes about 48 hours.

4.4 Evaluation Metrics

To evaluate the performance of the models, we use **ROUGE** (Lin, 2004) scores, which evaluate the n-grams recall of generated sentences with reference sentences, as automatic evaluation metrics (Fan et al., 2019; Khashabi et al., 2021). We report the F1 for ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-Lsum. ROUGE-1, 2, L measures the unigram, bigram, and longest common subsequence between the pair of sentences, respectively. The difference between ROUGE-L and ROUGE-Lsum is that ROUGE-Lsum splits text using " \n". The higher ROUGE score indicates higher similarity between generated questions and references.

We further perform human evaluations for the quality of generated questions in terms of **Fluency**: whether the questions are grammatically correct and fluent; **Relevance**: whether the questions are related to the answers; **Correctness**: whether the questions can be answered by the answers.

5 Experiment Results

In this section, we answer the four experimental research questions in turn.

5.1 Generation Performance (RQ1)

First, does the proposed QG framework have good performance? Table 3 shows the performance of the proposed closed-book QG model (denoted as $\mathbf{QG}_{ours}$) on the new dataset and two public datasets compared with the baseline model trained only with question generation loss (denoted as $\mathbf{QG}_b$). We observe that the proposed framework outperforms the baseline on three datasets for all ROUGE scores. For example, the performance increases by up to 2.0%, 2.7%, and 1.8% on the ROUGE-L score, respectively, which means the proposed framework can generate questions having a longer longest common subsequence (LCS) with ground-truth questions. We attribute these improvements to how contrastive learning pulls the answer representations closer to the ground-truth questions and thus, the generated questions have a higher chance to have similar words, phrases, or sentences with the ground-truth questions. The results demonstrate the effectiveness of our proposed QG framework.

Dataset	Model	R-1	R-2	R-L	R-Lsum
WikiCQA	QG_b	48.39	26.84	46.08	46.16
	QG_{ours}	**49.22**	**27.79**	**46.98**	**47.08**
GooAQ-S	QG_b	44.26	19.73	41.11	41.08
	QG_{ours}	**45.26**	**20.69**	**42.20**	**42.06**
ELI5	QG_b	28.62	10.10	25.93	26.23
	QG_{ours}	**29.15**	**10.36**	**26.40**	**26.69**

Table 3: QG results on three datasets. $\mathbf{QG}_b$: baseline model; $\mathbf{QG}_{ours}$: our proposed QG framework.

5.2 Quality of Generated Questions (RQ2)

Next, are the generated questions of good quality? To answer this question, we perform human evaluation to measure the quality of generated questions in terms of *fluency*, *relevance*, and *correctness*. We randomly sample 100 answer-question pairs from the WikiCQA dataset, which contains answers, ground-truth questions, and generated questions from the baseline model and our best model. Then, we ask three annotators to rate the generated question pairs, comparing them with the ground-truth questions. The generated questions are rated on a 1-5 scale (5 for the best) from the aspects of the three aspects above. Further, we calculate the percentage of questions that have higher ratings in each group.

Table 4 shows the average scores and the percentages of preferred questions on the three criteria. Both models are equally good at generating fluent questions. In terms of relevance and correctness, our approach shows 1.3% and 2.6% higher scores than the baseline method. These are consistent with what we expect: the contrastive learning part pulls the answer representation closer to the ground-truth question representation and generates more relevant questions; the reconstruction part can ensure that generated questions are of good quality and answerable. In addition, among 100 answer-question pairs, 24.7% and 23.3% of generated questions from our best model are rated higher while 56.3% and 57.4% of them have the same rating as those from baselines, which indicates that there is still substantial room to improve. To get a sense of the stability of the human evaluation results, we measure the inter-agreement among annotators using the Correlation Coefficient, which is 0.998, 0.944, and 0.976 in terms of these three aspects, showing excellent reliability.

Model	**fluency**		**relevance**		**correctness**	
	score	%	score	%	score	%
QG_b	4.83	**8.3**	3.84	19.0	3.49	19.3
QG_{ours}	4.83	7.3	**3.89**	**24.7**	**3.58**	**23.3**

Table 4: Results of human evaluation for baseline and our best model. **score**: is the average score from raters; **%**: represents the proportion of generated questions from one model that were rated higher than those from its counterpart.

5.3 Synthetic Data Generation (RQ3)

Further, can this QG framework be leveraged to generate effective synthetic data that can improve the closed-book QA task? Data augmentation is one of the main directions that question generation has been used for previously, with several studies finding improvements on the QA task (Lewis et al., 2019b; Alberti et al., 2019). Here, we show how to leverage the proposed QG framework to improve closed-book QA tasks on seen data (WikiCQA) and unseen data (GooAQ and ELI5). Since freely available summary data is a good resource to generate synthetic data (Lyu et al., 2021), we use WikiSum (Cohen et al., 2021), which contains 39,775 coherent-paragraph summaries written by the article's authors on the WikiHow website. We take each sentence from the article summary as an answer and pass it into the best QG model, described in Section 3.1, to generate a question. Then, we train an unsupervised QA model based on the synthetic QA pairs and optimize it by the following negative log-likelihood loss function:

$$\mathcal{L}_{qa} = -\sum_{i=1}^{N} \log p(a_i|Q), \tag{8}$$

where a_i is the i-th token in the generated answer and Q is the question.

In this work, we pre-train a BART-base model on the 200K synthetic QA pairs that are generated through the best QG model (denoted as $\mathbf{QA}_s$) and evaluate it on the test set of the seen dataset (WikiCQA) and unseen datasets (GooAQ-S and ELI5). This approach is unsupervised since the model is trained on no labeled question-answer pairs. The results are summarized in Table 5a, showing 22.4%, 12.5%, and 11.4% improvement than the BART-base model without any synthetic pre-training (denoted as $\mathbf{QA}_b$), on WikiCQA, GooAQ-S, and ELI5, respectively. This shows that pre-training on the generated question-answer pairs derived from our QG model leads to significant improvements, echoing findings in previous works that find synthetic data can be helpful (Khashabi et al., 2021; Lewis et al., 2021; Ding et al., 2021). After further fine-tuning on the target training set (denoted as $\mathbf{QA}_{s+f}$), from Table 5b we can see that $\mathbf{QA}_{s+f}$ also achieves better results than fine-tuned baselines model $\mathbf{QA}_{b+f}$ by 3.6%, 0.4%, and 4.3% on the same three datasets.

Dataset	**Model**	**R-1**	**R-2**	**R-L**	**R-Lsum**
WikiCQA	QA_b	18.41	5.53	14.72	15.98
	QA_s	**24.10**	**7.07**	**18.02**	**20.31**
GooAQ-S	QA_b	18.58	5.80	14.77	15.66
	QA_s	**21.93**	**6.12**	**16.62**	**18.35**
ELI5	QA_b	12.38	2.19	8.98	10.63
	QA_s	**13.97**	**2.29**	**10.00**	**11.77**

(a) Unsupervised QA results.

Dataset	**Model**	**R-1**	**R-2**	**R-L**	**R-Lsum**
WikiCQA	QA_{b+f}	27.16	7.32	19.85	23.49
	QA_{s+f}	**28.44**	**8.14**	**20.56**	**24.77**
GooAQ-S	QA_{b+f}	27.72	7.68	20.08	23.64
	QA_{s+f}	**28.44**	**7.73**	**20.17**	**24.12**
ELI5	QA_{b+f}	22.06	3.93	13.93	19.72
	QA_{s+f}	**23.37**	**4.27**	**14.53**	**20.88**

(b) Semi-supervised QA results.

Table 5: Evaluation on QA tasks. $\mathbf{QA}_{s/b}$: QA model w/o pre-training on synthetic data; $\mathbf{QA}_{*+f}$: QA_* fine-tuned on target dataset.

5.4 Ablation Study (RQ4)

Finally, we investigate the contribution of each component in the proposed closed-book QG framework. Table 6 shows the results on the WikiCQA dataset. As described in Section 3.2, the contrastive learning module takes ground-truth questions as positive pairs to the input answers (denoted as $\mathbf{CL}_t$) We also explore its variant – in which we feed the same input (answers) to the encoder with different dropouts to obtain the positive pairs (Gao et al., 2021) (denoted as $\mathbf{CL}_s$). We find that the choice of $\mathbf{CL}_t$ is slightly better than $\mathbf{CL}_s$ in most evaluation metrics. In addition, by adding the answer reconstruction (**AR**) module, the model performance can be further improved. Thus, we can observe the effectiveness of the proposed contrastive learning module and the overall design of the framework.

Models	**R-1**	**R-2**	**R-L**	**R-LSum**
QG$_b$	48.39	26.84	46.08	46.16
QG$_b$+CL$_s$	48.67	27.34	46.43	46.46
QG$_b$+CL$_t$	48.72	27.26	46.33	46.41
QG$_b$+CL$_s$+AR	**49.26**	27.77	46.82	46.88
QG$_b$+CL$_t$+AR	49.22	**27.79**	**46.98**	**47.08**

Table 6: Ablation study of our proposed QG framework on WikiCQA. $\mathbf{CL}_*$: contrastive learning module; **AR**: answer reconstruction module.

6 Case Study

Finally, we showcase some examples of the generated questions on the WikiCQA dataset. For the first example in Table 7a, we can see our model generates "how many" for the statistics-type answer, which is closer to the reference "what percentage" than the baseline model, generating a "yes/no" question. For the second one, although the reference contains the detailed information "textured surface" which is not in the answer, our model captures the information "without nails", which is more informative than the baseline model. Thus, the proposed model can produce better questions that are more relevant to the answers and contain more detailed information than questions generated from baseline models. There are also some cases where the proposed model fails to generate good questions shown in Table 7b. Compared with the reference question "what food can be included on a clear liquid diet", our model generates "what can I eat to lose weight", which fails in automatic evaluation metrics and is hard for annotators to justify in terms of relevance. For the second one, although the "broken door" generated from our model is relevant to the answer, it's still far from the reference "door sticking to the weather strip". The question generated from the baseline model related to "bedroom" is even less relevant. These examples encourage us to explore how to capture more semantics in the answers and generate questions that have more detailed information in the future.

A	The most recent statistics show that around 2.5% of small businesses are audited by the irs.
Q$_B$	Are small businesses audited by the irs?
Q$_G$	How many small businesses are audited by the irs?
Q$_R$	What percentage of small businesses are audited?
A	Try using poster putty to secure the images in your collage. if that doesn't work, you may have to use nails.
Q$_B$	How do i put pictures in a collage?
Q$_G$	How do i make a collage without nails?
Q$_R$	How can i make a collage on a wall with a textured surface?

(a) Good examples.

A	Foods such as popsicles, hard candy, and gelatin can be eaten on a clear liquid diet.
Q$_B$	what can you eat on a clear liquid diet?
Q$_G$	What can I eat to lose weight?
Q$_R$	What food can be included on a clear liquid diet?
A	You'll have to remove the door and sand, prime, and repaint it.
Q$_B$	What do i have to do to make my bedroom look nice?
Q$_G$	How do i fix a broken door?
Q$_R$	What do i do if my door is sticking to the weather strip?

(b) Bad examples.

Table 7: Generated question examples. $\mathbf{Q}_B$: generated questions from baseline model; $\mathbf{Q}_G$: generated questions from proposed QG model; $\mathbf{Q}_R$: reference questions.

7 Conclusion

In this work, aiming to empower a QG model to better understand the semantics of long-form abstractive answers and store more information in its parameters, we propose a closed-book QG model empowered by a contrastive learning module and an answer reconstruction module. We present a new *closed-book* long-form QA dataset – WikiCQA involving more than 20K real user QA pairs and show that WikiCQA is a valuable training resource, complementing public datasets. Through the experiments, the proposed QG model shows better performance than baselines through automatic and human evaluations. Moreover, we show how to leverage the proposed model as a data augmentation strategy to improve existing *closed-book* QA systems. The closed-book QA model, pre-trained on our generated synthetic QA pairs, achieves better performance on the seen dataset (WikiCQA). In addition, it shows strong generalization on unseen datasets (GooAQ and ELI5), which further demonstrates the effectiveness of our QG framework for enhancing closed-book QA performance.

Limitations

While our model shows promising results on English datasets, its efficiency and performance in other languages require further investigation. Moreover, our evaluation is limited to datasets with one-to-one QA pairs, without considering situations where multiple answers correspond to a single question. Before feeding the input into the model, we ignore the tokens that exceed the maximum length we set, which may sometimes bring in an information loss for the input corpus. Additionally, our model is built on transformer architecture and therefore requires huge computation resources, especially for large training data sizes. Our focus in this paper is solely on the closed-book setting and our experiments center around four research questions to evaluate the effectiveness of the proposed framework, which we believe is an essential and emerging area deserving of specialized research investigation. In future work, we aim to explore the feasibility of adapting our framework for other QG scenarios.

Acknowledgements

We thank Zhuoer Wang, Yun He, Ziwei Zhu, anonymous reviewers, and the action editor for providing valuable feedback.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Ruisheng Cao, Su Zhu, Chen Liu, Jieyu Li, and Kai Yu. 2019. Semantic parsing with dual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 51–64, Florence, Italy. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Woon Sang Cho, Yizhe Zhang, Sudha Rao, Asli Celikyilmaz, Chenyan Xiong, Jianfeng Gao, Mengdi Wang, and Bill Dolan. 2021. Contrastive multi-document question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 12–30, Online. Association for Computational Linguistics.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE.

Nachshon Cohen, Oren Kalinsky, Yftah Ziser, and Alessandro Moschitti. 2021. WikiSum: Coherent summarization dataset for efficient human-evaluation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 212–219, Online. Association for Computational Linguistics.

Kaize Ding, Dingcheng Li, Alexander Hanbo Li, Xing Fan, Chenlei Guo, Yang Liu, and Huan Liu. 2021. Learning to selectively learn for weakly-supervised paraphrase generation. In *2021 Conference on Empirical Methods in Natural Language Processing*, pages 5930–5940.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Liam Dugan, Eleni Miltsakaki, Shriyash Upadhyay, Etan Ginsberg, Hannah Gonzalez, DaHyeon Choi, Chuning Yuan, and Chris Callison-Burch. 2022. A feasibility study of answer-unaware question generation for education. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1919–1926.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Zichu Fei, Qi Zhang, and Yaqian Zhou. 2021. Iterative GNN-based decoder for question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2573–2582, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895, Online. Association for Computational Linguistics.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *Conference ICLR*.

Xin Jia, Wenjie Zhou, Xu Sun, and Yunfang Wu. 2021. Eqg-race: Examination-type question generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13143–13151.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Daniel Khashabi, Amos Ng, Tushar Khot, Ashish Sabharwal, Hannaneh Hajishirzi, and Chris Callison-Burch. 2021. GooAQ: Open question answering with diverse answer types. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 421–433, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. 2018. Automatic question generation using relative pronouns and adverbs. In *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158, Melbourne, Australia. Association for Computational Linguistics.

Kyungho Kim, Kyungjae Lee, Seung-won Hwang, Young-In Song, and Seungwook Lee. 2021. Query generation for multimodal documents. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 659–668, Online. Association for Computational Linguistics.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 6602–6609.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Philippe Laban, John Canny, and Marti A. Hearst. 2020. What's the latest? a question-driven news chatbot. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 380–387, Online. Association for Computational Linguistics.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 889–898, Beijing, China. Association for Computational Linguistics.

Adam D Lelkes, Vinh Q Tran, and Cong Yu. 2021. Quiz-style question generation for news stories. In *Proceedings of the Web Conference 2021*, pages 2501–2511.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019a. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019b. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. 2018. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Jiaying Lu, Xiangjue Dong, and Carl Yang. 2021. Weakly supervised concept map generation through task-guided graph translation. *arXiv preprint arXiv:2110.15720*.

Jiaying Lu, Xin Ye, Yi Ren, and Yezhou Yang. 2022. Good, better, best: Textual distractors generation for multiple-choice visual question answering via reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (O-DRUM Workshop)*, pages 4921–4930.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258, Online. Association for Computational Linguistics.

Raheel Qader, François Portet, and Cyril Labbé. 2019. Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 552–562, Tokyo, Japan. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sudha Rao and Hal Daumé III. 2019. Answer-based Adversarial Training for Generating Clarification Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 143–155, Minneapolis, Minnesota. Association for Computational Linguistics.

Xuhui Ren, Hongzhi Yin, Tong Chen, Hao Wang, Zi Huang, and Kai Zheng. 2021. Learning to ask appropriate questions in conversational recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 808–817, New York, NY, USA. Association for Computing Machinery.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Pushkar Shukla, Carlos Elmadjian, Richika Sharan, Vivek Kulkarni, Matthew Turk, and William Yang Wang. 2019. What should I ask? using conversationally informative rewards for goal-oriented visual dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6442–6451, Florence, Italy. Association for Computational Linguistics.

Md Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages

5651–5656, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Cunxiang Wang, Pai Liu, and Yue Zhang. 2021. Can generative pre-trained language models serve as knowledge bases for closed-book QA? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3241–3251, Online. Association for Computational Linguistics.

Kun Wu, Lijie Wang, Zhenghua Li, Ao Zhang, Xinyan Xiao, Hua Wu, Min Zhang, and Haifeng Wang. 2021. Data augmentation with hierarchical sql-to-question generation for cross-domain text-to-sql parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8974–8983.

Haoran Yang, Wai Lam, and Piji Li. 2021. Contrastive representation learning for exemplar-guided paraphrase generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4754–4761, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bingsheng Yao, Dakuo Wang, Tongshuang Sherry Wu, Zheng Zhang, Toby Jia-Jun Li, Mo Yu, and Ying Xu. 2022. It is ai's turn to ask humans a question: Question-answer pair generation for children's story books. In *ACL*.

Xusen Yin, Li Zhou, Kevin Small, and Jonathan May. 2021. Summary-oriented question generation for informational queries. In *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, pages 81–97.

Lili Yu, Howard Chen, Sida I. Wang, Tao Lei, and Yoav Artzi. 2020. Interactive classification by asking informative questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2664–2680, Online. Association for Computational Linguistics.

Ruqing Zhang, Jiafeng Guo, Lu Chen, Yixing Fan, and Xueqi Cheng. 2021. A review on question generation from natural language text. *ACM Transactions on Information Systems (TOIS)*, 40(1):1–43.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2495–2509.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

Zhenjie Zhao, Yufang Hou, Dakuo Wang, Mo Yu, Chengzhong Liu, and Xiaojuan Ma. 2022. Educational question generation of children storybooks via question type distribution learning and event-centric summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5073–5085, Dublin, Ireland. Association for Computational Linguistics.

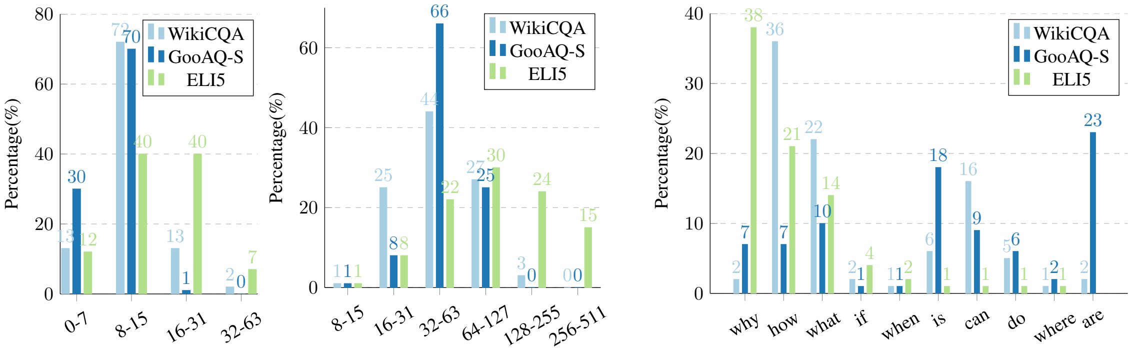
Su Zhu, Ruisheng Cao, and Kai Yu. 2020. Dual learning for semi-supervised natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1936–1947.

Appendix A Data Filtering

First, we choose the question-answer pairs where the questions start with a question-word in the list ['how', 'what', 'can', 'is', 'do', 'why', 'are', 'does', 'where', 'when', 'should', 'will', 'did', 'which', 'who', 'would', 'if', 'about', 'for', 'as', 'could', 'in', 'after', 'at', 'while', 'to', 'am', 'has', 'any'] and end with a question mark. Then, we filter out the pairs where the length of answers is less than 8 tokens or the answers are meaningless. For the "meaningless" answers we mean, answers are "please refer to certain website http://xxx" or "please refer to the article xxx".

Appendix B Dataset Analysis

To better understand the content of WikiCQA in comparison with existing QA datasets, Figure 2 shows the distributions of question length, answer length, and common question word. The questions and answers are tokenized by BART (Lewis et al., 2020). From Figure 2a and 2b we could see more than 80% of the questions in WikiCQA and GooAQ-S have less than 15 tokens and more than 90% of answers of them lie in the range of 16-127 tokens while both questions and answers



(a) Question length.

(b) Answer length.

(c) Common question-word.

Figure 2: Comparison of the distribution of QA length and question types among three datasets.

in ELI5 have a broader range of lengths. By measuring the proportion of the leading unigram that starts a question (Figure 2c), the dominant pattern is “how” questions in WikiCQA, while ELI5 has many “why” questions. We think WikiCQA has the potential to benefit the community in three ways. Firstly, its source is unique compared to the current two closed-book data sources, as it is collected from the Q&A section of articles on WikiHow. The questions and answers are related to the specific articles, asked by WikiHow users who read the articles and answered by a knowledgeable reader or editor. Secondly, compared to large-scale QA datasets, WikiCQA has longer answers and more open-ended questions with a wider range of question types. Additionally, it has a different data distribution compared to the other two datasets, which is crucial for testing the generalization ability of question-answering models. Finally, the question-answer pairs within relevant articles in WikiCQA can also be used as a valuable resource for conversational question-answering tasks. Our code and data can be found at https://github.com/dongxiangjue/Closed-book-Question-Generation-via-Contrastive-Learning.

A Hybrid Detection and Generation Framework with Separate Encoders for Event Extraction

Ge Shi[*⋆]
Beijing University of Technology
shige@bjut.edu.cn

Yunyue Su[*]
Computer Network Information Center, CAS
suyunyue@cnic.cn

Yongliang Ma
Langboat
mayongliang@langboat.com

Ming Zhou
Langboat
zhouming@langboat.com

Abstract

The event extraction task typically consists of event detection and event argument extraction. Most previous work models these two subtasks with shared representation by multiple classification tasks or a unified generative approach. In this paper, we revisit this pattern and propose to use independent encoders to model event detection and event argument extraction, respectively, and use the output of event detection to construct the input of event argument extraction. In addition, we use token-level features to precisely control the fusion between two encoders to achieve joint bridging training rather than directly reusing representations between different tasks. Through a series of careful experiments, we demonstrate the importance of avoiding feature interference of different tasks and the importance of joint bridging training. We achieved competitive results on standard benchmarks (ACE05-E, ACE05-E$^+$, and ERE-EN) and established a solid baseline.

1 Introduction

Event extraction has always been an important and challenging task in Natural Language Processing (NLP) (Sundheim, 1992). It aims to extract event triggers with specific types and event arguments with correct roles from unstructured plain texts into a structured form, which mostly describes "who, when, where, what, why" and "how" of real-world events that happened (Li et al., 2021a). For example, Figure 1 shows a `Meet` event, triggered by "*met*", which describes the `Entity` "*Kelly*" meet with another `Entity` "*officials*" in `Place` "*Seoul*".

Previous studies can be roughly classified into classification-based and generation-based methods depending on the decoder used. Classification-based method usually divides EE into two subtasks: (1) Event Detection (ED), which identifies event triggers and their types. (2) Event Argument Extraction (EAE) extracts the arguments and their corresponding roles for given event triggers and then models them as classification tasks, either learned in a pipeline framework or a joint formulation. Recently, generation-based event extraction methods have emerged as an alternative to traditional classification-based methods due to their better data-efficient and flexibility to include additional guidance. These methods take a sentence with discrete or continuous prompts as input and use BART-style backbone learning to summarize the sentence into a natural sentence based on a manually designed template. The template is composed of natural utterances describing argument role labels, which can provide rich label semantics, leading to great success in generation-based event extraction.

However, most of these methods simultaneously learn shared representations for ED and EAE. As shown in previous works (Nguyen and Grishman, 2015; Lu et al., 2019), ED relies more on lexical (e.g., lemma, synonyms) and shallow syntactic features (e.g., pos tags, dependent and governor words of trigger words). At the same time, the EAE task focuses more on syntactic dependency features. For example, the dependency path between trigger and arguments (Liu et al., 2018). Simply using shared representations dealing with the two distinct tasks would hurt their performance. This phenomenon is also observed in similar tasks, such as entity relation extraction (Zhong and Chen, 2021), where they use two different BERTs for modeling entity extraction and relation extraction, respectively.

To this end, we propose a simple but empirically powerful hybrid framework for event extraction. We model ED and EAE using separate encoders to avoid feature interference between these two tasks. In addition, we conduct extensive experiments to investigate the difference between classification-based and generation-based methods, and we ob-

---

[*] Equal contribution. Yunyue Su contributes during her internship at Langboat.

[⋆] Corresponding author.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3163–3180
May 2-6, 2023

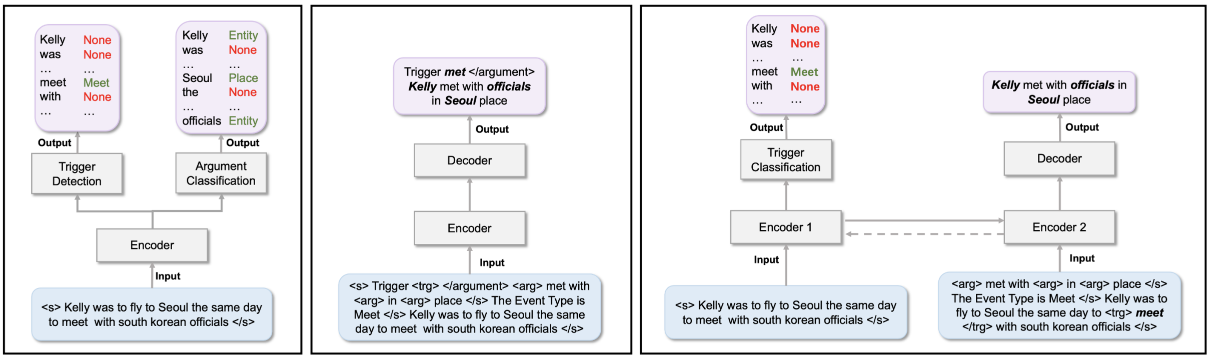


Figure 1: The first two figures are the two major paradigms in the field of event extraction, and the third figure is our paradigm.

serve that (1) Classification-based methods are superior to generation-based methods in modeling token classification tasks. (2) Generation-based methods perform better in modeling EAE because they can capture label semantics. Based on the observations, we instantiate our model with two different decoders: a classification-style decoder for ED and a generation-style decoder for EAE. Finally, to enhance the interaction between these two tasks, we design a bridging mechanism to provide EAE with information derived from ED and a two-stage training method that uses gradients from EAE to guide ED learning. We evaluate our model on three widely used benchmarks, ACE05-E, ACE05-E$^+$, and ERE-EN. Experimental results show that our model establishes the new state-of-the-art on ACE05-E and ACE05-E$^+$, and achieves comparable results on the ERE-EN dataset.

Our contributions can be summarized as follows:

- We first propose a method using separate encoders for modeling event extraction that can avoid feature interference.
- We propose a hybrid classification and generation method that enjoys the advantages of both approaches.
- To model the dependency between ED and EAE, we propose a bridging mechanism and two-stage training method.
- Experimental results show that our proposed method can outperform many strong baselines and achieve new SOTA on ACE05-E, ACE05-E$^+$ [1].

2 Related Work

Event extraction is usually considered to be composed of two sub-tasks: event detection and event argument extraction. Previous researchers are keen to use a shared encoder to model the contextual representation of different tasks. We group existing event extraction methods into classification-based and generation-based.

Classification-based Method. Classification-based methods tend to model event extraction as a classification task (Mekala and Shang, 2020; Guo et al., 2021; Xiao et al., 2021; Liu et al., 2020; Du and Cardie, 2020b; Li et al., 2020a; Ma et al., 2022) and deal with the recognition of trigger and arguments separately (Ji and Grishman, 2008), e.g., Liang et al. (2020) only consider the event detection and Chen et al. (2015) only consider the extraction of event arguments. Some previous works (Li et al., 2013) have tried to joint training these two tasks to enhance the connection between them, and Yang and Mitchell (2016); Nguyen et al. (2016); Liu et al. (2017, 2018); Lin et al. (2020a) all try to enhance the effect of joint training by adding as much entity and relation information as possible. The difference lies in their shared encoding layers. For example, Liu et al. (2017, 2018) used CNN and Bi-RNN successively, while Wadden et al. (2019a); Lin et al. (2020a) used graph structure. In addition, some works (Ramponi et al., 2020; Du and Cardie, 2020a; Yang et al., 2018) solve the event extraction in sequence labeling manner (Chen et al., 2020; Gui et al., 2020; Jiang et al., 2021) by tagging the sentence only once, which may not solve the overlapping problem.

Generation-based Method. In contrast to classification-based methods, the main goal of

---

[1] Our codes are publicly available at `https://github.com/OPilgrim/TDE-GTEE`

generation-based methods is to use a common structure to uniformly model various tasks, including event detection and event argument extraction. The output structure can be a sentence filled with slotted templates (Li et al., 2021b; Du et al., 2022), or some linearly serialized tree structure (Lu et al., 2021a, 2022). Paolini et al. (2021) even constructed an end-to-end translation directly. Generative models can leverage richer prior knowledge. However, the accuracy is not high in classification problems. We think that the generative model may pay more attention to global features and ignores local details. Based on that, several recent works (Hsu et al., 2021; Liu et al., 2022a) use prompt-based approaches to force the model to focus on specific pieces of information to control its output for different event types.

Unlike previous works, we argue that the contextual representation of tasks is different, and sharing one contextual representation will harm the model's performance. So we use independent encoders to learn the contextual representation of each task. In addition, since tasks are not entirely independent, increasing the interaction between tasks will be conducive to improving each other; we achieve it through a bridging mechanism and a two-stage training method.

3 Method

3.1 Problem Definition

The input of the problem is a sentence $\mathcal{C}$ consisting of N tokens $c_1, c_2, ..., c_N$. Let $\mathcal{E} = \{e_1, e_2, ..., e_M\}$ denotes a set of pre-defined event types. The event extraction problem can be decomposed into two sub-tasks:

Event Detection (ED). Event detection aims to identify possible event mentions in the input sequence. We define that each pair of (c_i, e_j) refers to an independent event mention and the event detection is, for each token $c_i \in \mathcal{C}$, to predict an event type $y_e(c_i) \in \mathcal{E}$ or $y_e(c_i) = \epsilon$ representing token c_i is not a trigger. The output of the task is $Y_e = \{(c_i, e_j) | c_i \in \mathcal{C}, e_j \in \mathcal{E}\}$.

Event Argument Extraction (EAE). Event argument extraction aims to identify all entities involved in an event mention (c_i, e_j). Let $\mathbf{s} = \{s_1, s_2, ..., s_t\}$ to be a set of candidate entity spans, and $\mathcal{R}^j$ denotes a set of predefined argument roles in event e_j. The event argument extraction is, for each span $s_i \in \mathbf{s}$, to predict a argument role type $y_r(s_i) \in \mathcal{R}^j$, or span s_i is not an argument belongs to event e_j: $y_r(s_i) = \epsilon$. The output of the task is $Y_r = \{(s_i, r_i, e_j) | s_i \in \mathbf{s}, r_i \in \mathcal{R}^j, e_j \in \mathcal{E}\}$.

3.2 Our Approach

In this section, we introduce our proposed method, HDGSE[3] (the **H**ybrid **D**etection and **G**eneration framework with **S**eparate **E**ncoders for **E**vent **E**xtraction), based on the overall architecture of Figure 2.

Event Trigger Detection. We use BERT as the backbone of our detection model and treat it as a token-level multi-classification task, which makes the model learn the different probabilities of each event type (Li et al., 2021b). As we mentioned in Section 2, the sequence annotation method based on CRF cannot solve the span coverage problem, so we did not implement this scheme. Given the input sequence $\mathcal{C} = \{c_1, ..., c_i, ..., c_N\}$, the detection model will detect all possible trigger tokens $\{c_i, | i \in \{i\}_{i=1}^{N}\}$ and their corresponding event type $\{e_j, | j \in \{j\}_{j=1}^{M}\}$ as mentioned in Formula 1,

$$label_{c_i} = \begin{cases} 0, & c_i \text{ is not a trigger,} \\ j, & c_i \text{ is trigger for event type } j. \end{cases} \quad (1)$$

where N is the length of $\mathcal{C}$ and M is the number of event types in ontology $\mathcal{O}$ [2] . Each pair of $(c_i,\ e_j)$ indicates the hit of an event $\{E_k | k \in \{k\}_{k=1}^{K}\}$, where K denotes the number of events in $\mathcal{C}$. Then the generative model extracts arguments for each event E_k in turn.

Generative Argument Extraction. After detecting the candidate event triggers, the argument extraction task is divided into several subtasks according to the detected triggers and event types, and each subtask is an event mention. We process each event mention independently with a generative approach and insert markers at the input sequence to highlight the trigger. The generative model is based on BART, and the lower part of Figure 2 shows the detailed structure. Specifically, for subtask $\mathcal{S}_{E_k, \mathcal{C}}$, the input $\mathcal{X}$ of the generative model includes the event type aware prompt $\mathcal{P}_{e_j}$ and context $\mathcal{C}' = \{c_1, ..., \text{<trg>}, c_i, \text{</trg>}, ..., c_N\}$, where the trigger c_i is marked by two special tokens "<trg>" and "</trg>" to provide trigger position information for the corresponding subtask. Given

---

[2] We follow Li et al. (2021c) and reuse RAMS AIDA ontology and the KAIROS ontology as the ontology for ACE05-E, ACE05-E$^+$ and ERE.

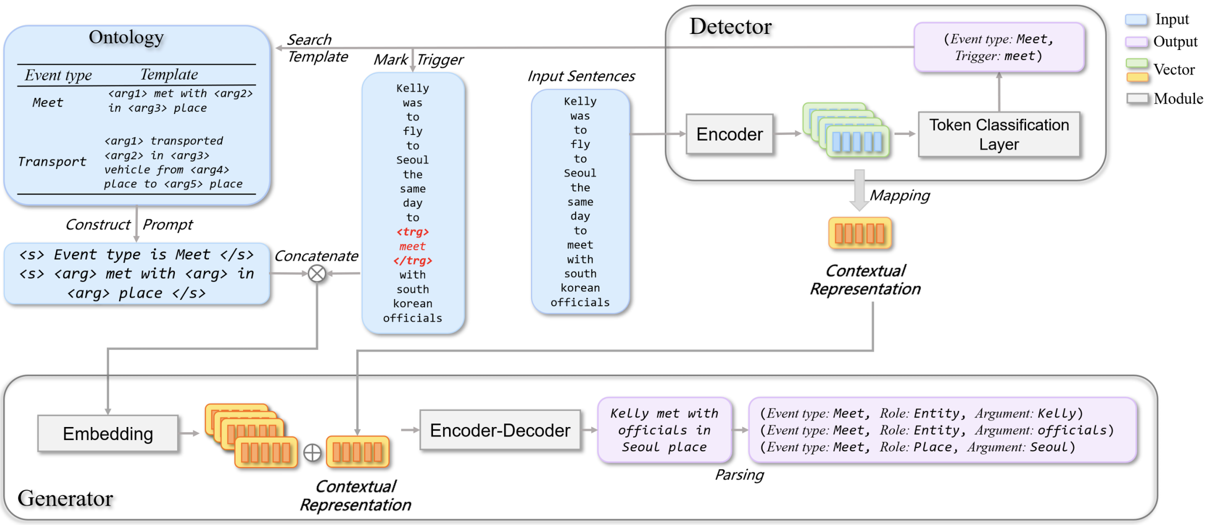


Figure 2: Illustration of our end-to-end joint event extraction framework. Given an input sequence, the detector first detects several candidate triggers and classifies their event types. Then, the generator generates filled template from the trigger-marked input and finally parses the arguments through a deterministic algorithm. During generation, the contextual representation of the trigger that the detector learns is fused into the generator.

the previous generated tokens $y_{<i}$ and the input $\mathcal{X}$, the BART models the conditional probability of selecting the next token y_i as $p(y_i|y_{<i},\mathcal{X})$, and the entire probability $p(\mathcal{Y}|\mathcal{X})$ is calculated as

$$\begin{aligned} p(\mathcal{Y}|\mathcal{X}) &= \prod_{i=1}^{|\mathcal{Y}|} p(y_i|y_{<i},\mathcal{X}) \\ \mathcal{X} &= [\mathcal{P}_{e_j}; [SEP]; \mathcal{C}'] \\ \mathcal{Y} &= \mathcal{A}_{e_j} \end{aligned} \tag{2}$$

, where $[;]$ denotes the sequence concatenation operation and $[SEP]$ is the corresponding separate marker in the BART, $\mathcal{A}_{e_j}$ is the answered prompt.

Similar to generative template-based method GTEE(Liu et al., 2022b), the prompt $\mathcal{P}_{e_j}$ for subtask $\mathcal{S}_{E_k,\mathcal{C}}$ contains the type instruction $\mathcal{I}_{e_j}$ and the template $\mathcal{T}_{e_j}$. The type instruction is an indication of the event type described by natural language, and the template describes the expected output format, including several placeholders, reflecting how the arguments participant in the event. Take Figure 2 as example, the generative model's input is type instruction "`Event type is Meet`", template "`<arg> met with <arg> in <arg> place`", and content concatenated with separator. The ground truth $\mathcal{G}_{e_j}$ is "`Kelly met with officials in Seoul place`", where the placeholder "<arg>" is replaced by the corresponding argument "`Kelly`", "`officials`", and "`Seoul`". Each event type has its own template and we follow Li et al. (2021b) to reuse the pre-defined argument templates.

Bridging Event Detection and Event Argument Extraction. Our proposal to independently learn contextual representations for ED and EAE does not mean that the two tasks are not connected; argument extraction directly depends on determining event types and triggers. So to enhance the interaction between them, we bridge the two tasks by trigger: first, as mentioned in the previous section, we highlight the trigger in the input of EAE, which provides the location information; the second is to fuse the context information of triggers into the EAE model, which is the focus of this section. Both kinds of information provide the EAE model with prior knowledge of events. Specifically, for the trigger token c_i, its hidden state in BERT's last hidden layer is $\boldsymbol{h}_{c_i}$, and its input embedding in BART is $\boldsymbol{Emb}(c_i)$. A semantic transformation is performed by multiplying the $\boldsymbol{h}_{c_i}$ by the projection matrix $\boldsymbol{P}$ to obtain the projected vector $\boldsymbol{v}_{c_i} = \boldsymbol{h}_{c_i}\boldsymbol{P}$ as contextual representations, where $\boldsymbol{P}$ can be learned by fully connected networks. We refer to this operation as "Mapping", as illustrated in Figure 2. Then, we add $\boldsymbol{v}_{c_i}$ and $\boldsymbol{Emb}(c_i)$ directly. Another appropriate method is sufficient to directly use the vector $\boldsymbol{v}_{c_i}$ to initialize the embedding representation of trigger markers. We conduct comparative experiments against these two appropriates in Section 5.2.

Training and Inference. In this paper, we design a two-stage training approach: (1) In the first step, we first train ED and EAE separately so that they can learn the contextual representation independently. (2) In the second step, to overcome the error propagation problem of the pipeline, we continue to use joint training to optimize the global loss based on the model trained in the first step and use the gradient of EAE to guide the optimization of ED.

Mathematically, The trainable parameters of our model include ϕ and φ, which come from the BERT and BART respectively. The training objective of the detection model is to minimize the focal loss (Lin et al., 2017) between each token's predicted label and the golden label:

$$\mathcal{L}_{\phi}(\mathcal{C}) = \sum_{i=1}^{N} -(1 - p_{c_i,e_j})^{\gamma} log(p_{c_i,e_j}) \tag{3}$$

And the training objective of the argument extraction model is to minimize the negative loglikelihood over all subtasks $\mathcal{S}_{E_k,\mathcal{C}}$ of the input sample $\mathcal{C}$:

$$\begin{aligned} &\mathcal{L}_{\varphi}(\mathcal{C}) = -\sum_{k=1}^{K} log\ p(\mathcal{G}_{e_j,\mathcal{C}'_k} | \mathcal{X}_{e_j,\mathcal{C}'_k}) \\ &\mathcal{X}_{e_j,\mathcal{C}'_k} = [\mathcal{P}_{e_j}; [SEP]; \mathcal{C}'_k] \\ &\mathcal{C}'_k = \{c_1, ..., \text{<trg>}, c_i, \text{</trg>}, ..., c_N\} \end{aligned} \tag{4}$$

Finally, during the joint bridging training phase, the loss of the whole model is:

$$\mathcal{L}(\mathcal{D}) = \sum_{t=1}^{|\mathcal{D}|} (\mathcal{L}_{\phi}(\mathcal{C}_t) + \mathcal{L}_{\varphi}(\mathcal{C}_t)) \tag{5}$$

Implementation details are shown in Appendix B.

4 Experiment

4.1 Setup

Datasets. We evaluate our methods on three widely used event extraction benchmarks, ACE05-E, ACE05-E$^+$ and ERE-EN. Both of the ACE datasets are from the Automatic Content Extraction 2005 (ACE 2005) dataset constructed by Doddington et al. (2004), and the ERE-EN is from ERE dataset (Song et al., 2015). All their details can be found in Appendix A.

Evaluation Metrics. We consider the same criteria following prior works (Liu et al., 2022b; Hsu et al., 2021; Lu et al., 2021b) and report the Precision *P*, Recall *R* and F1 score *F1* of trigger and argument. Meanwhile, we consider that a trigger is correctly identified if its offset matches the ground truth (**Trg-I**) and is correctly classified if its event matches the ground truth as well (**Trg-C**). In the same way, we consider an argument is correctly identified if its offset matches the ground truth (**Arg-I**) and is correctly classified if its event type and role label all matches the ground truth as well (**Arg-C**).

Compared Baselines. We consider several representative works as our baselines, including both classification-based and generation-based methods, and some of their implementation details are listed in Appendix B.

we consider the following classification-based models:

- **DYGIE++**(Wadden et al., 2019b), a BERT-based model learns shared span representations between multi-tasks and updates span representations through dynamic graph propagation layers.
- **GAIL**(Zhang et al., 2019), a RL model jointly extracting entity and event.
- **OneIE**(Lin et al., 2020a), an end-to-end IE system that employs designed global feature and beam search, was state-of-the-art.
- **BERT_QA**(Du and Cardie, 2020c), an MRC-based model views EE tasks as a question-answering problem with multi-turns of separated QA pairs and learns a classifier to indicate the position of the predicted span.
- **MQAEE**:(Li et al., 2020b), a multi-turn question answering system.

We also consider the following generation-based models:

- **TANL**:(Paolini et al., 2021), a method treats EE tasks as translation tasks in a trigger-argument pipeline.
- **BART-GEN**(Paolini et al., 2021), a template-based conditional generation method to generate corresponding arguments in a predefined format.
- **TEXT2EVENT**(Lu et al., 2021b), a sequence-to-structure generation method that converts the input sequence to a tree-like event structure.

- **DEGREE-E2E**(Hsu et al., 2021), an end-to-end conditional generation method that uses natural sentences as discrete prompts, which makes it easier for them to leverage label semantics.

- **GTEE-DYNPREF**(Liu et al., 2022b), an end-to-end conditional genration method with dynamic prompts and trained with prefix-tuning.

4.2 Main Results

For each dataset, we train our model with 5 different random seeds, and report the means of the corresponding results.

Table 1 compares our approach HDGSE[3] with all the baselines on ACE05-E, while Table 2 illustrates the results compared with the state-of-the-art on ACE05-E$^+$ and ERE-EN. As shown, our model achieves strong performance and outperforms all the baselines on two datasets of ACE 2005. At the same time, our model also performs competitively on ERE-EN, second only to GTEE-DYNPREF (Lin et al., 2020b).

For event detection, our model achieves an absolute Trg-C F1 improvement of $+5.8\%$, $+2.9\%$ on ACE05-E and ACE05-E$^+$ respectively compared to DEGREE-E2E (Lin et al., 2020b) and GTEE-DYNPRE (Liu et al., 2022b) that also utilizes joint training but use the generative method for ED, indicating that the classification-based method has more advantages in event detection than the generative-based method. On the other hand, our model also shows a significant improvement over the classification-based methods, e.g., a gain of 4.3% on ACE05-E compared to ONEIE (Lin et al., 2020b). As we will show later in our experiments, part of this improvement is due to the bridging mechanism.

For event argument extraction, our approach outperforms the previous best methods, ONEIE (Lin et al., 2020b) and DEGREE-E2E (Lin et al., 2020b), with absolute Arg-C F1 gains of $+1\%$, $+1.4\%$ on ACE05-E, ACE05-E$^+$ respectively. Moreover, it outperforms the best generative method DEGREE-E2E (Lin et al., 2020b) on ACE05-E and ERE-EN with absolute Arg-C F1 gains of $+2\%$, $+3.9\%$. Such improvements demonstrate the effectiveness of maintaining different contextual representations for ED and EAE, and incorporating trigger information into the EAE model. Although our model performs second only to GTEE-DYNPRE (Liu et al., 2022b) on ERE-EN, it outperforms GTEE-DYNPRE on all other datasets, indicating that our model has better robustness on different datasets.

5 Analysis

5.1 Selection of Task Models

We further investigated the impact of using classification-based or generation-based models for the ED task and the EAE task, respectively, to gain insight into the advantages and disadvantages of these two approaches for event extraction tasks.

Event Detection: Classification vs Generative. We first compare the two paradigms on the event detection task and list the results in Table 3. The experimental details can be found in Appendix B. We observe that the generation-based model is significantly worse than the classification-based model on this task. One possible reason is that generation-based models pay more attention to the global features of sentences and have fewer advantages in ED, which require trigger tokens and their local context. Moreover, the classification-based model can directly provide the location of trigger spans, which is more helpful for EAE. Therefore, the classification-based paradigm is more suitable for the ED task than the generation-based one.

Table 3: The event detection results of the classification-based and generation-based approach.

ED Paradigm	ACE05-E		ACE05-E$^+$		ERE-EN	
	Trg-I	**Trg-C**	**Trg-I**	**Trg-C**	**Trg-I**	**Trg-C**
Classification-based	81.9	77.8	80.8	76.9	75.9	65.7
Generation-based	67.8	45.3	66.8	45.6	61.4	37.1

Event Argument Extraction: Classification vs Generative. We eliminated the trigger detection task in order to investigate the impact of different paradigms on event argument extraction. Given golden triggers as input, we implemented several classification and generation paradigm baselines for EAE. The experimental results are shown in Table 4. The generative approach performs as well as the classification-based model under the standard setting (with trigger marker). And several template-based generative approaches, such as GTEE-BASE (Liu et al., 2022b), BART-GEN (Li et al., 2021b), DEGREE-EAE (Lin et al., 2020b) and HDGSE[3], perform significantly better.

In particular, DEGREE-EAE performs best under the gold trigger marker setting. This is mainly due to the fact that DEGREE-EAE incorporates more knowledge of events in the prompt design,

Table 1: Results on ACE05-E for event extraction. The first group is the classification-based methods, and the second is the generation-based methods. For each group, we bold the highest F1 scores for Trg-C and Arg-C, and the second highest is bold in italics.

Model	Trg-C			Arg-C		
	P	**R**	**F1**	**P**	**R**	**F1**
classification-based						
DYGIE++ (Wadden et al., 2019b)	-	-	69.7	-	-	48.8
GAIL (Zhang et al., 2019)	74.8	69.4	72.0	61.6	45.7	52.4
ONEIE (Lin et al., 2020b)	-	-	***74.7***	-	-	***56.8***
BERT_QA (Du and Cardie, 2020c)	71.1	73.7	72.3	56.8	50.2	53.3
MQAEE (Li et al., 2020b)	-	-	71.7	-	-	53.4
generation-based						
TANL (Paolini et al., 2021)	-	-	68.5	-	-	48.5
BART-GEN (Li et al., 2021b)	69.5	72.8	71.1	56.0	51.6	53.7
TEXT2EVENT (Lu et al., 2021b)	67.5	71.2	69.2	46.7	53.4	49.8
DEGREE-E2E (Hsu et al., 2021)	-	-	***73.3***	-	-	***55.8***
GTEE-DYNPREF (Liu et al., 2022b)	63.7	84.4	72.6	49.0	64.8	***55.8***
HDGSE[3]	76.1	82.1	**79.0**	55.3	60.4	**57.8**

Table 2: Results on ACE05-E$^+$ and ERE-EN for event extraction. We bold the highest F1 scores for Trg-C and Arg-C, and the second highest is bold in italics.

Model	ACE05-E$^+$						ERE-EN					
	Trg-C			Arg-C			Trg-C			Arg-C		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
ONEIE	72.1	73.6	72.8	55.4	54.3	54.8	58.4	59.9	59.1	51.8	49.2	50.5
TEXT2EVENT	71.2	72.5	71.8	54.0	54.8	54.4	59.2	59.6	59.4	49.4	47.2	48.3
DEGREE-E2E	-	-	70.9	-	-	***56.3***	-	-	57.1	-	-	49.6
GTEE-DYNPREF	67.3	83.0	***74.3***	49.8	60.7	54.7	61.9	72.8	**66.9**	51.9	58.8	**55.1**
HDGSE[3]	75.5	79.0	**77.2**	57.6	57.8	**57.7**	64.5	67.9	***66.1***	54.5	52.6	***53.5***

such as "Event Type Description" and "Event Keywords". Interestingly, our final results on EE are better than DEGREE-EAE because (1) DEGREE-EAE uses a generative paradigm in the event detection task, (2) shares the contextual representation of the encoder between two tasks, which indirectly proves the correctness of our hypothesis.

5.2 Effect of Bridging Mechanisms

We mentioned in Section 3.2 that trigger marker and contextual representation fusion were used to establish a bridge connection for the two independent encoders. This section will look closely at these two components to see how they affect our model.

We remove the possible connection modules between two independent encoders under the settings of Joint and Pipeline, respectively, and present the experimental results in Table 5. It can be seen that removing the trigger marker causes significant damage to the model under both training paradigms. Although contextual representation can also improve the model's performance, the overall improvement space is not as ample as the trigger marker. Further, when we remove both of them, as shown in Table 5, the F1 score of Trg-C remains at a very high level for the three datasets, which are still SOTA at ACE05-E and ACE05-E$^+$. However, at the same time, the F1 of Arg-C is significantly reduced and no longer SOTA. These phenomena show that the effect of ED representing the upper bound of EAE and the bridging mechanism can help EAE approach this upper bound and even improve the result of ED in reverse. That is where the main contribution of the bridging mechanism lies.

From another point of view, when only comparing the training paradigms, it can be found that loss sharing during joint training can significantly improve the model's overall performance, so the Joint results are generally better than those of Pipeline, which proves the effectiveness of our two-stage training program.

We also design several contrast schemes for the fusion way of contextual representation. In Section 3.2, we discussed two approaches, one is to assign the mapped contextual representation to the trigger

Table 4: Results of event argument extraction. Models predict arguments based on the given gold triggers. *We report the numbers from the original paper. †We reproduce the results.

Model	Type	ACE05-E		ACE05-E$^+$		ERE-EN	
		Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
DyGIE++*	Cls	66.2	60.7	-	-	-	-
BERT_QA*	Cls	68.2	65.4	-	-	-	-
OneIE*	Cls	73.2	69.3	***73.3***	***70.6***	75.3	70.0
GTEE-BASE †	Gen	70.1	67.2	67.3	63.6	72.3	66.8
BART-GEN †	Gen	66.9	66.7	70.0	66.8	74.6	69.2
TANL*	Gen	65.9	61.0	66.3	62.3	75.6	69.6
DEGREE-EAE*	Gen	**76.0**	**73.5**	**75.2**	**73.0**	**80.2**	**76.3**
HDGSE^3	Gen	***73.8***	***70.2***	72.1	69.0	***76.4***	***72.0***

Table 5: Ablation study for the effectiveness of trigger marker and fused contextual representation.

Insert Setting (Trained ED)	ACE05-E				ACE05-E$^+$				ERE-EN			
	Trg-I	Trg-C	Arg-I	Arg-C	Trg-I	Trg-C	Arg-I	Arg-C	Trg-I	Trg-C	Arg-I	Arg-C
HDGSE^3 (Joint)	83.0	79.0	60.1	57.8	81.1	77.2	60.2	58.2	76.4	66.1	56.5	53.5
- remove marker trigger	83.0	78.2	58.8	56.9	80.8	76.9	57.6	55.7	75.1	64.9	53.8	50.2
- remove context fusion	82.1	78.0	59.1	56.6	80.8	76.9	58.2	56.1	75.2	65.3	55.8	52.4
- remove both	82.1	78.0	58.6	56.1	80.8	76.9	57.6	55.6	75.0	64.7	53.5	49.9
HDGSE^3 (Pipeline)	77.9	74.2	55.4	53.5	79.2	75.7	56.0	53.3	73.4	62.1	52.0	48.8
- remove marker trigger	77.9	74.2	40.5	38.6	79.2	75.7	46.5	44.7	73.4	62.1	29.7	28.6

marker, and the other is to directly add with the trigger representation learned by the language model during pre-training. In addition to the above two, we further explore what results can be obtained by directly fusing the contextual representation without mapping. Table 6 lists the experimental results. Note that directly fusing the contextual representation of ED and EAE without mapping causes significant damage to the model, which is even worse than the direct deletion of the contextual representation in Table 5. That proves the contextual representations learned by ED and EAE are different and preferably not directly shared. On the other hand, the performance of mapped contextual representations is almost the same regardless of whether they are fused with trigger markers or triggers. We believe this is because contextual representations provide more semantic information, which is not affected by the difference in fusion objects.

Table 6: Study on the fusion form of contextual representation. Models predict arguments based on the predicted trigger.

HDGSE^3 (Joint)	ACE05-E		ACE05-E$^+$		ERE-EN	
	Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
trigger marker	60.3	57.8	59.5	57.3	56.5	53.5
- w/o mapping	59.0	56.5	59.9	57.7	55.1	51.7
trigger	60.1	57.8	60.2	58.2	56.5	53.5
- w/o mapping	58.6	55.4	58.0	55.5	55.0	50.2

5.3 Prompts and Templates

Generative-based event extraction methods tend to be sensitive to the prompts and templates used (Liu et al., 2022a). Since our model adopts a generative method for EAE, we further investigated the robustness of our model when using different prompts and templates.

Necessity of Type Instruction. We first consider replacing the static type instruction such as "*The Event Type is Meet*" but still providing explicit event type information to the model. So we refer to Zhong and Chen (2021) and use <trigger:Event type> and </trigger:Event type> instead of the original type instruction. The experimental results in Table 7 show that using sentences in natural language to describe event types will perform better than replacing them with tokens. Therefore, we still keep this setting in our experiments.

Table 7: Study on the necessity of type instruction. r/ stands for replace, and we replaced type instruction with <trigger:Event type> and </trigger:Event type>.

HDGSE^3-EAE (Gold Trg)	ACE05-E		ACE05-E$^+$		ERE-EN	
	Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
w/ Type Instruction	73.8	70.2	72.1	69.0	76.4	72.0
r/ Type Instruction	72.8	69.9	71.6	67.4	75.0	69.4

Sensitivity to Template Design. Our method requires templates with slotted values to assist

EAE, so we designed several templates to explore whether the model is robust under different template Settings. We designed three types of templates from low to high semantic integrity and the detailed construction details can be found in Appendix C. We put all three types of templates involved in ACE 2005 and ERE-EN in Tables 13 and 14, and the Table 8 show the experimental results. Templates without semantics perform worst, indicating that the model is still sensitive to the template's design. However, the weak and strong semantic integrity results are close, indicating that the model still has good robustness to sentences with certain linguistic logic. Weak semantic integrity templates can ensure the model's performance, whether manually designed or model-generated. The experiments in this paper are all done based on templates with weak semantic integrity, and we leave generating templates from models for the future.

Table 8: Study on the effect of different template constructing rules. Models predict arguments based on the given gold trigger.

HDGSE^3-EAE (Gold Trg)	**ACE05-E**		**ACE05-E$^+$**		**ERE-EN**	
	Arg-I	**Arg-C**	**Arg-I**	**Arg-C**	**Arg-I**	**Arg-C**
No semantic integrity	67.7	64.2	66.3	63.0	74.9	70.2
Weak semantic integrity	73.8	70.2	72.1	69.0	76.4	72.0
Strong semantic integrity	73.1	70.4	70.7	69.1	76.6	71.5

6 Conclusion

In this paper, we revisit the classification-based and generation-based event extraction methods and empirically propose a simple but robust hybrid event extraction scheme. Our model learns two independent encoders for event detection and event argument extraction and uses simple trigger marker and contextual representation fusion to bridge training jointly, for which we devise a two-stage training approach. We conduct extensive analyses to understand the superior performance of our approach. These analyses verify the effectiveness of using the classification model and the generative model to learn the contextual representation of event detection and event argument extraction separately and validate the importance of taking the result of event detection as the input of event argument extraction. We hope this simple model will serve as a strong benchmark for end-to-end event extraction and make us rethink the value of a shared representation of multi-tasks.

Limitations

Our findings in this paper only verified in event extraction. It will be more exciting and valuable if migrated to other multi-task problems. We will leave that for future work.

Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions. This work is supported by National Natural Science Foundation of China (No. 62106010).

References

Luoxin Chen, Weitong Ruan, Xinyue Liu, and Jianhua Lu. 2020. SeqVAT: Virtual adversarial training for semi-supervised sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8801–8811, Online. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Xinya Du and Claire Cardie. 2020a. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8010–8020, Online. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020b. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020c. Event extraction by answering (almost) natural questions. *arXiv preprint arXiv:2004.13625*.

Xinya Du, Sha Li, and Heng Ji. 2022. Dynamic global memory for document-level argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), pages 5264–5275, Dublin, Ireland. Association for Computational Linguistics.

Tao Gui, Jiacheng Ye, Qi Zhang, Zhengyan Li, Zichu Fei, Yeyun Gong, and Xuanjing Huang. 2020. Uncertainty-aware label refinement for sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2316–2326, Online. Association for Computational Linguistics.

Biyang Guo, Songqiao Han, Xiao Han, Hailiang Huang, and Ting Lu. 2021. Label confusion learning to enhance text classification models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12929–12936.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

I Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, Nanyun Peng, et al. 2021. Degree: A data-efficient generative event extraction model. *arXiv preprint arXiv:2108.12724*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics.

Ting Jiang, Deqing Wang, Leilei Sun, Huayi Yang, Zhengyang Zhao, and Fuzhen Zhuang. 2021. Lightxml: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7987–7994.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020a. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020b. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Qian Li, Jianxin Li, Jiawei Sheng, Shiyao Cui, Jia Wu, Yiming Hei, Hao Peng, Shu Guo, Lihong Wang, Amin Beheshti, et al. 2021a. A compact survey on event extraction: Approaches and applications. *arXiv preprint arXiv:2107.02126*.

Sha Li, Heng Ji, and Jiawei Han. 2021b. Document-level event argument extraction by conditional generation. *arXiv preprint arXiv:2104.05919*.

Sha Li, Heng Ji, and Jiawei Han. 2021c. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Xin Liang, Dawei Cheng, Fangzhou Yang, Yifeng Luo, Weining Qian, and Aoying Zhou. 2020. F-hmtc: Detecting financial events for investment decisions based on neural hierarchical multi-label text classification. In *IJCAI*, pages 4490–4496.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020a. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020b. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online. Association for Computational Linguistics.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022a. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5216–5228, Dublin, Ireland. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022b. Dynamic prefix-tuning for generative template-based event extraction. *arXiv preprint arXiv:2205.06166*.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the*

2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization (2017). *arXiv preprint arXiv:1711.05101*.

Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. 2019. Distilling discrimination and generalization knowledge for event detection via delta-representation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4366–4376, Florence, Italy. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021a. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021b. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. *arXiv preprint arXiv:2106.09232*.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. *arXiv preprint arXiv:2203.12277*.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774, Dublin, Ireland. Association for Computational Linguistics.

Dheeraj Mekala and Jingbo Shang. 2020. Contextualized weak supervision for text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 323–333, Online. Association for Computational Linguistics.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. *arXiv preprint arXiv:2101.05779*.

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. 2020. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online. Association for Computational Linguistics.

Zhiyi Song, Ann Bies, Stephanie Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. 2015. From light to rich ERE: Annotation of entities, relations, and events. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 89–98, Denver, Colorado. Association for Computational Linguistics.

Beth M. Sundheim. 1992. Overview of the fourth Message Understanding Evaluation and Conference. In *Fourth Message Uunderstanding Conference (MUC-4): Proceedings of a Conference Held in McLean, Virginia, June 16-18, 1992*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019a. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019b. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Sijia Wang, Mo Yu, and Lifu Huang. 2022. The art of prompting: Event detection based on type specific prompts. *arXiv preprint arXiv:2204.07241*.

Lin Xiao, Xiangliang Zhang, Liping Jing, Chi Huang, and Mingyang Song. 2021. Does head label help for long-tailed multi-label text classification. *arXiv preprint arXiv:2101.09704*.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. DCFEE: A document-level Chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55, Melbourne, Australia. Association for Computational Linguistics.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint entity and event extraction with generative adversarial imitation learning. *Data Intelligence*, 1(2):99–120.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

A Datasets

ACE 2005 has 599 annotated English documents, 33 event types, and 22 argument roles. ERE contains 458 English documents, 38 event types, and 21 argument roles. Following previous work (Zhang et al., 2019; Wadden et al., 2019b; Du and Cardie, 2020c; Lin et al., 2020b; Lu et al., 2021b; Hsu et al., 2021), we use the same preprocess method and obtain three datasets, ACE05-E, ACE05-E$^+$, and ERE-EN, shown in Table 9. Compared to ACE05-E, both ACE05-E$^+$ and ERE-EN contain pronoun roles and multi-token event triggers.

Table 9: Dataset statistics.

Dataset	Split	Sents	Events	Roles
ACE05-E	Train	17,172	4202	4859
	Dev	923	450	605
	Test	832	403	576
ACE05-E$^+$	Train	19,216	4419	6607
	Dev	901	468	759
	Test	676	424	689
ERE-EN	Train	14,736	6208	8924
	Dev	1209	525	730
	Test	1163	551	822

B More Implementation details

Main Settings.

Table 10: Hyperparameters for two-stage of training, with the first phase being separate independent training for the two tasks and the second phase being joint bridging training.

Name	Independent Training		Joint Bridge Training
	ED (BERT)	EAE (BART)	EE (BERT+BART)
Learning rate	2e-5	2e-5	5e-7
Batch size	1*32	1*32	1*32
Epochs	40	40	30
Max sequence length	185 | 325	Id.	Id.
Max output length	-	50	Id.
Weight decay	1e-5	1e-5	1e-5
Gradient clip	5.0	5.0	5.0
Warm-up ratio	10%	10%	10%
Loss	Focal	Cross-entropy	-
Focal gamma	3	-	-
Gen loss	-	Sum	-

We used the hugging face implementation of BERT-large and BART-large and optimized our models by AdamW (Loshchilov and Hutter, 2019). In addition, we use the two-layer fully connected layer with tanh as the intermediate activation function as the mapping function for the contextual representation. The dimension of the hidden layer is 1024. GELU (Hendrycks and Gimpel, 2016) also activates the output before being added to the other representations.

We first train the two models independently so that they can learn the contextual representation of event detection and event argument extraction tasks, respectively. Then in the joint bridging training stage, we also set different learning rates for different models. However, because their loss has decreased to the same magnitude in the process of independent training, the final learning rates obtained through grid search are the same size. We optimized the parameters with grid search, in the independent training: training epoch 40, learning rate $\in$ {1e-5, 2e-5, 1e-4}, training batch size with gradient accumulation $\in$ {1*8, 1*32, 1*64}, focal loss gamma $\in$ {1, 2, 3, 4, 5}, generation loss $\in$ {mean, sum}. As for the joint bridge training, we only used grid search to find the optimal learning rate $\in$ {5e-8, 5e-7, 5e-6, 2e-5} and fixed other parameters: training epoch 30, training batch size with gradient accumulation 1*32, focal loss gamma 3, generation loss "sum." Table 10 shows the final optimal parameter combination. Each experiment was conducted on NVIDIA GeForce RTX 3090 Core GPU 24GB. It is worth noting that the ERE-EN dataset has more noise than ACE 2005, so the model needs a larger learning rate on ERE-EN than that in ACE 2005. In other words, when trained independently on the ERE-EN dataset, the learning rates of BERT and BART are 3e-5 and 4e-5, respectively. In joint bridge training, the learning rates are 7e-7 and 8e-7, respectively.

While Inferring, our generative model generates sequences by greedy search, and the maximum sequence length is set according to dataset statistics, which is a bit larger than the length of the longest ground truth, for ACE05-E, ACE05-E$^+$, ERE-EN, its 50 tokens. As for the input length, the ACE05-E and ACE05-E$^+$ are 185 tokens, the ERE-EN is 325 tokens, and the detection model is consistent with the generative model. Besides, we parse the event records by template matching and slot mapping according to the ontology $\mathcal{O}$, as shown in Algorithm 1.

Reproduce Baselines. Among the baselines we selected, we tried to reproduce BERT_QA, BART-GEN, DEGREE, and GTEE-BASE. They all got similar results to those in the original paper report except for DEGREE and BERT_QA. Therefore, we used the experimental results reported in (Li et al., 2021b) and (Lin et al., 2020b) in Section 5.1, while the above two models used our experimental results. The hyperparameter Settings are listed as follows:

Table 11: Hyperparameter Settings for BART-GEN for our implementation and given in the original paper.

Parameter	Li et al. (2021b)	Our Implementation
Base model	BART-large	BART-large
Learning rate	[1e-5, 3e-5]	2e-5
Scheduler	Linear (without warmup)	Id.
Batch size	2*8	4*4
Max sequence length	512	512
Training epochs	[3,6]	20
Beam size	4	4

Algorithm 1 Extracting arguments from predicted template

```
Input: Predicted template $\mathcal{PT}$, Predefined template $\mathcal{GT}$, event type $\mathcal{E}$, Predefined Ontology $\mathcal{O}$ {e.g. $\mathcal{PT}$ is "Kelly met with officials in Seoul place", $\mathcal{GT}$ is "<arg1> met with <arg2> in <arg3> place"}
Initialization: predicted argument list $\mathcal{A}$, the pointers $p_{ptr} \leftarrow 0, t_{ptr} \leftarrow 0$
Split $\mathcal{PT}$ and $\mathcal{GT}$ into token lists, and ensure that <arg\d*> is a whole
while $p_{ptr} < |\mathcal{PT}|$ and $t_{ptr} < |\mathcal{GT}|$ do
  if $\mathcal{GT}\,[t_{ptr}]$ is <arg\d*> then
    $n \leftarrow$ \d*
    $role\_name = \mathcal{O}\,[\mathcal{E}]\,[n]$
  end if
  if $\mathcal{PT}\,[p_{ptr}]$ is <arg> then
    $p_{ptr} \leftarrow p_{ptr} + 1, t_{ptr} \leftarrow t_{ptr} + 1$
  else
    $arg_{start} = p_{ptr}, nxt_{ptr} = p_{ptr} + 1$
    while $nxt_{ptr} < |\mathcal{GT}|$ do
      if $\mathcal{GT}\,[nxt_{ptr}] ==$ <arg\d*> then
        Break
      end if
      $nxt_{ptr} \leftarrow nxt_{ptr} + 1$
    end while
    while $p_{ptr} < |\mathcal{PT}|$ do
      if $\mathcal{PT}\,[p_{ptr}] == \mathcal{GT}\,[t_{ptr} + 1]$ and $\mathcal{PT}\,[p_{ptr} : p_{ptr} + nxt_{ptr} - t_{ptr} - 1] == \mathcal{GT}\,[t_{ptr} + 1 : nxt_{ptr}]$ then
        Break
      end if
      $p_{ptr} \leftarrow p_{ptr} + 1$
    end while
    $\mathcal{A} \cup [\mathcal{E}, \mathcal{PT}\,[arg_{start} : p_{ptr}]\,, role\_name]$
  end if
end while
Output: $\mathcal{A}$
```

Table 12: Hyperparameter Settings for GTEE-BASE for our implementation and given in the original paper.

Parameter	Liu et al. (2022b)	Our Implementation
Base model	BART-large	BART-large
Learning rate	1e-5	2e-5
Batch size	32*8	4*8
Max sequence length	-	185 | 320
Max output length	-	78 | 100
Training epochs	40	40
Weight decay	1e-5	1e-5
Gradient clip	5.0	5.0
Warm-up ratio	10%	10%
Negative sample ratio	4%	3%

- **BART-GEN.** We follow the settings of the original paper when reproducing BART-GEN. However, there are some differences, and we list them in Table 11. In addition, since Li et al. (2021b) only implemented ACE05-E, we set the learning rate by referring to our model when implementing ACE05-E$^+$ and ERE, and other parameters were unchanged.

- **GTEE-BASE.** Since Liu et al. (2022b) have not open-sourced the code, we reproduced GTEE-BASE by ourselves with the hyperparameter settings shown in Table 12. Here, our negative sample ratio is $\frac{1}{evet\ type\ num}$, which means that ACE05-E and ACE05-E$^+$ are $\frac{1}{33}$ because they have 33 event types, and ERE-EN is $\frac{1}{38}$ because it has 38 event types. They all round off to a ratio of about 3%.

The hardware environment of these experiments is the same as that of HDGSE^3, as mentioned in the previous paragraph. On the other hand, the python environment is strictly set up according to the requirements of open-source codes.

Event Detection. In Section 5.1, we implement the classification-based and generation-based methods on the event detection task. As the classification-based approach, we used the event detection model of our HDGSE^3 in Section 3.2. Meanwhile, we use a template-based generative model for the generation-based approach, analogous to our event argument extraction model, but with templates and no prompts. We concatenate multiple templates for input sequences with multiple event mentions so that the model generates all event mentions sequentially. The template we used here was "Event type: <event> Trigger: <trg>." The experimental hyperparameters are consistent with those of main settings.

C Template Constructing

In this section, we show the three templates mentioned in Section 5. Their construction strategy is as follows:

- **No semantic integrity.** The first template is the least semantically complete; we take the role's name as a hint and add it before the argument slot, while the order of argument placement is random. For example, "Agent <arg1> Person <arg2> Place <arg3>". This template can only tell us the `Justice:Release-Parole` event's argument roles, but it does not form a natural sentence and has no semantic information.
- **Weak semantic integrity.** The second type of template maintains weak semantic integrity, and we use ontologies predetermined by Li et al. (2021c) as such templates, such as "<arg1> released or paroled <arg2> in <arg3> place". We can roughly understand that this is an `Justice:Release-Parole` event because the template mentions the two keywords "released or paroled." But this type of template misses the subject and role information, and the model may be confused. For example, "<arg1>" is not restricted to "Person" in the template, and the model may be likely to predict an "Institution" for it. Hence, the semantics of this kind of template is incomplete.
- **Strong semantic integrity.** The third template combines the advantages of the above two templates. It hints at the roles and ensures the sentence's semantic integrity. We refer to "APEX" defined by Wang et al. (2022), consider all roles to paraphrase each event, and arrange argument slots after each role, e.g., "an Entity <arg1> ends its custody of a Person <arg2> at a Place <arg3>". The sentence is semantically complete after removing slots.

ACE 2005 and ERE-EN are listed in Table 13 and Table 14, respectively.

Table 13: All templates of ACE05-E and ACE05-E^{+} used in the main and ablation experiments in this paper.

Event	Type	Template
Movement:Transport	1	<arg1> transported <arg2> in <arg3> vehicle from <arg4> place to <arg5> place
	2	Agent <arg1> Artifact <arg2> Vehicle <arg3> Origin <arg4> Destination <arg5>
	3	an Agent <arg1> moves an Artifact <arg2> from Origin <arg4> to Destination <arg5> with Vehicle <arg3> at Price
Personnel:Elect	1	<arg1> elected <arg2> in <arg3> place
	2	Entity <arg1> Person <arg2> Place <arg3>
	3	a candidate Person <arg2> wins an election by voting Entity <arg1> at a Place <arg3>
Personnel:Start-Position	1	<arg1> started working at <arg2> organization in <arg3> place
	2	Person <arg1> Entity <arg2> Place <arg3>
	3	a Person <arg1> begins working for an Entity <arg2> or change office at a Place <arg3>
Personnel:Nominate	1	<arg1> nominated <arg2>
	2	Agent <arg1> Person <arg2>
	3	a Person <arg2> is nominated for a new position by another Agent <arg1> at a Place
Personnel:End-Position	1	<arg1> stopped working at <arg2> organization in <arg3> place
	2	Person <arg1> Entity <arg2> Place <arg3>
	3	a Person <arg1> stops working for an Entity <arg2> or change office at a Place <arg3>
Conflict:Attack	1	<arg1> attacked <arg2> hurting <arg5> victims using <arg3> instrument at <arg4> place
	2	Attacker <arg1> Target <arg2> Instrument <arg3> Place <arg4> Victim <arg5>
	3	An Attacker <arg1> physically attacks a Target <arg2> with Instrument <arg3> at a Place <arg4> hurting Victim <arg5>
Contact:Meet	1	<arg1> met with <arg2> in <arg3> place
	2	Entity <arg1> Entity <arg2> Place <arg3>
	3	one Entity <arg1> and another Entity <arg2> come together at same Place <arg3> and interact in person
Life:Marry	1	<arg1> married <arg2> in <arg3> place
	2	Person <arg1> Person <arg2> Place <arg3>
	3	one Person <arg1> and another Person <arg2> are married at a Place <arg3>
Transaction:Transfer-Money	1	<arg1> gave money to <arg2> for the benefit of <arg3> in <arg4> place
	2	Giver <arg1> Recipient <arg2> Beneficiary <arg3> Place <arg4>
	3	transfer Money from the Giver <arg1> to the Beneficiary <arg3> or Recipient <arg2> at a Place <arg4>
Conflict:Demonstrate	1	<arg1> demonstrated at <arg2> place
	2	Entity <arg1> Place <arg2>
	3	Entity <arg1> come together in a Place <arg2> to protest or demand official action
Business:End-Org	1	<arg1> organization shut down at <arg2> place
	2	Org <arg1> Place <arg2>
	3	an Organization Org <arg1> goes out of business at a Place <arg2>
Justice:Sue	1	<arg1> sued <arg2> before <arg3> court or judge in <arg4> place
	2	Plaintiff <arg1> Defendant <arg2> Adjudicator <arg3> Place <arg4>
	3	Plaintiff <arg1> initiate a court proceeding to determine the liability of a Defendant <arg2> judge by Adjudicator <arg3> at a Place <arg4>
Life:Injure	1	<arg1> injured <arg2> with <arg3> instrument in <arg4> place
	2	Agent <arg1> Victim <arg2> Instrument <arg3> Place <arg4>
	3	a Victim <arg2> experiences physical harm from Agent <arg1> with Instrument <arg3> at a Place <arg4>
Life:Die	1	<arg1> killed <arg2> with <arg3> instrument in <arg4> place
	2	Agent <arg1> Victim <arg2> Instrument <arg3> Place <arg4>
	3	life of a Victim <arg2> ends by an Agent <arg1> with Instrument <arg3> at a Place <arg4>
Justice:Arrest-Jail	1	<arg1> arrested <arg2> in <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	the Agent <arg1> takes custody of a Person <arg2> at a Place <arg3>
Contact:Phone-Write	1	<arg1> communicated remotely with <arg2> at <arg3> place
	2	Entity <arg1> Entity <arg2> Place <arg3>
	3	phone or written communication between one Entity <arg1> and another Entity <arg2> at a Place <arg3>
Transaction:Transfer-Ownership	1	<arg1> gave <arg4> to <arg2> for the benefit of <arg3> at <arg5> place
	2	Seller <arg1> Buyer <arg2> Beneficiary <arg3> Artifact <arg4> Place <arg5>
	3	buying selling loaning borrowing giving receiving of Artifact <arg4> from Seller <arg1> to Buyer <arg2> or Beneficiary <arg3> at a Place <arg5> at Price
Business:Start-Org	1	<arg1> started <arg2> organization at <arg3> place
	2	Agent <arg1> Org <arg2> Place <arg3>
	3	an Agent <arg1> create a new Organization Org <arg2> at a Place <arg3>
Justice:Execute	1	<arg1> executed <arg2> at <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	the life of a Person <arg2> is taken by an Agent <arg1> at a Place <arg3>
Justice:Trial-Hearing	1	<arg1> tried <arg2> before <arg3> court or judge in <arg4> place
	2	Prosecutor <arg1> Defendant <arg2> Adjudicator <arg3> Place <arg4>
	3	a court proceeding initiated to determine the guilty or innocence of the Defendant <arg2> Person with Prosecutor <arg1> and Adjudicator <arg3> at a Place <arg4>
Life:Be-Born	1	<arg1> was born in <arg2> place
	2	Person <arg1> Place <arg2>
	3	a Person <arg1> is born at a Place <arg2>
Justice:Charge-Indict	1	<arg1> charged or indicted <arg2> before <arg3> court or judge in <arg4> place
	2	Prosecutor <arg1> Defendant <arg2> Adjudicator <arg3> Place <arg4>
	3	a Defendant <arg2> is accused of a crime by a Prosecutor <arg1> for Adjudicator <arg3> at a Place <arg4>
Justice:Convict	1	<arg1> court or judge convicted <arg2> in <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	an Defendant <arg2> found guilty of a crime by Adjudicator <arg1> at a Place <arg3>
Justice:Sentence	1	<arg1> court or judge sentenced <arg2> in <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	the punishment for the Defendant <arg2> is issued by a state actor Adjudicator <arg1> at a Place <arg3>
Business:Declare-Bankruptcy	1	<arg1> declared bankruptcy at <arg2> place
	2	Org <arg1> Place <arg2>
	3	Organization Org <arg1> request legal protection from debt collection at a Place <arg2>
Justice:Release-Parole	1	<arg1> released or paroled <arg2> in <arg3> place
	2	Entity <arg1> Person <arg2> Place <arg3>
	3	an Entity <arg1> ends its custody of a Person <arg2> at a Place <arg3>
Justice:Fine	1	<arg1> court or judge fined <arg2> at <arg3> place
	2	Adjudicator <arg1> Entity <arg2> Place <arg3>
	3	a Adjudicator <arg1> issues a financial punishment Money to an Entity <arg2> at a Place <arg3>
Justice:Pardon	1	<arg1> court or judge pardoned <arg2> at <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	an Adjudicator <arg1> lifts a sentence of Defendant <arg2> at a Place <arg3>
Justice:Appeal	1	<arg1> appealed to <arg2> court or judge at <arg3> place
	2	Plaintiff <arg1> Adjudicator <arg2> Place <arg3>
	3	the decision for Defendant of a Plaintiff <arg1> is taken to a higher Place <arg3> for Adjudicator <arg2> review with Prosecutor
Justice:Extradite	1	<arg1> extradited <arg2> from <arg3> place to <arg4> place
	2	Agent <arg1> Person <arg2> Origin <arg3> Destination <arg4>
	3	a Person <arg2> is sent by an Agent <arg1> from Origin <arg3> to Destination <arg4>
Life:Divorce	1	<arg1> divorced <arg2> in <arg3> place
	2	Person <arg1> Person <arg2> Place <arg3>
	3	one Person <arg1> and another Person <arg2> are officially divorced at a Place <arg3>
Business:Merge-Org	1	<arg1> organization merged with <arg2> organization
	2	Org <arg1> Org <arg2>
	3	two or more Organizations Org <arg1> come together to form a new organization Org <arg2> at a Place
Justice:Acquit	1	<arg1> court or judge acquitted <arg2>
	2	Adjudicator <arg1> Defendant <arg2>
	3	a trial of Defendant <arg2> ends but Adjudicator <arg1> fails to produce a conviction at a Place

Table 14: All templates of ERE-EN used in the main and ablation experiments in this paper.

Event Type	Type	Template
Conflict:Attack	1	<arg1> attacked <arg2> using <arg3> instrument at <arg4> place
	2	Attacker <arg1> Target <arg2> Instrument <arg3> Place <arg4>
	3	An Attacker <arg1> physically attacks a Target <arg2> with Instrument <arg3> at a Place <arg4>
Justice:Acquit	1	<arg1> court or judge acquitted <arg2> at <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	a trial of Defendant <arg2> ends but Adjudicator <arg1> fails to produce a conviction at a Place <arg3>
Personnel:Elect	1	<arg1> elected <arg2> in <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	a candidate Person <arg2> wins an election by voting Entity <arg1> at a Place <arg3>
Justice:Release-Parole	1	<arg1> released or paroled <arg2> in <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	an Entity <arg1> ends its custody of a Person <arg2> at a Place <arg3>
Personnel:Nominate	1	<arg1> nominated <arg2> at <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	a Person <arg2> is nominated for a new position by another Agent <arg1> at a Place <arg3>
Justice:Appeal	1	<arg1> appealed to <arg2> court or judge sentenced <arg3>
	2	Prosecutor <arg1> Adjudicator <arg2> Defendant <arg3>
	3	the decision for Defendant <arg3> of a Plaintiff is taken to a higher Place for Adjudicator <arg2> review with Prosecutor <arg1>
Transaction:Transfer-Ownership	1	<arg1> gave <arg4> to <arg2> for the benefit of <arg3> at <arg5> place
	2	Giver <arg1> Recipient <arg2> Beneficiary <arg3> Thing <arg4> Place <arg5>
	3	giving of Artifact Thing <arg4> from Giver <arg1> to Recipient <arg2> for the benefit of Beneficiary <arg3> at a Place <arg5>
Business:Declare-Bankruptcy	1	<arg1> declared bankruptcy
	2	Org <arg1>
	3	Organization Org <arg1> request legal protection from debt collection
Contact:Meet	1	<arg1> met face-to-face with <arg2> in <arg3> place
	2	Entity <arg1> Entity <arg2> Place <arg3>
	3	one Entity <arg1> and another Entity <arg2> come together at same Place <arg3> and interact in person
Life:Marry	1	<arg1> married <arg2> in <arg3> place
	2	Person <arg1> Person <arg2> Place <arg3>
	3	one Person <arg1> and another Person <arg2> are married at a Place <arg3>
Life:Divorce	1	<arg1> divorced <arg2> in <arg3> place
	2	Person <arg1> Person <arg2> Place <arg3>
	3	one Person <arg1> and another Person <arg2> are officially divorced at a Place <arg3>
Business:Merge-Org	1	<arg1> organization merged with <arg2> organization
	2	Org <arg1> Org <arg2>
	3	two or more Organizations Org <arg1> come together to form a new organization Org <arg2> at a Place
Contact:Correspondence	1	<arg1> communicated remotely with <arg2> at <arg3> place
	2	Entity <arg1> Entity <arg2> Place <arg3>
	3	one Entity <arg1> communicated remotely with another Entity <arg2> at a Place <arg3>
Contact:Contact	1	<arg1> communicated with <arg2> at <arg3> place
	2	Entity <arg1> Entity <arg2> Place <arg3>
	3	one Entity <arg1> communicated with another Entity <arg2> face to face at a Place <arg3>
Manufacture:Artifact	1	<arg1> manufactured or created or produced <arg2> at <arg3> place
	2	Agent <arg1> Artifact <arg2> Place <arg3>
	3	an Agent <arg1> manufactured or created or produced Artifact <arg2> at a Place <arg3>
Movement:Transport-Person	1	<arg1> transported <arg2> in <arg3> instrument from <arg4> place to <arg5> place
	2	Agent <arg1> Person <arg2> Instrument <arg3> Origin <arg4> Destination <arg5>
	3	an Agent <arg1> transported a Person <arg2> in Instrument <arg3> from Origin <arg4> place to Destination <arg5>
Movement:Transport-Artifact	1	<arg1> transported <arg2> from <arg3> place to <arg4> place
	2	Agent <arg1> Artifact <arg2> Origin <arg3> Destination <arg4>
	3	an Agent <arg1> transported Artifact <arg2> from Origin <arg3> place to Destination <arg4>
Contact:Broadcast	1	<arg1> communicated to <arg2> at <arg3> place (one-way communication)
	2	Entity <arg1> Audience <arg2> Place <arg3>
	3	an Entity <arg1> one-way communicated to one or more Audience <arg2> at a Place <arg3>
Transaction:Transaction	1	<arg1> gave something to <arg2> for the benefit of <arg3> at <arg4> place
	2	Giver <arg1> Recipient <arg2> Beneficiary <arg3> Place <arg4>
	3	a Giver <arg1> gave something to a Recipient <arg2> for the benefit of Beneficiary <arg3> at a Place <arg4>
Personnel:Start-Position	1	<arg1> started working at <arg2> organization in <arg3> place
	2	Person <arg1> Entity <arg2> Place <arg3>
	3	a Person <arg1> begins working for an Entity <arg2> or change office at a Place <arg3>
Justice:Pardon	1	<arg1> court or judge pardoned <arg2> at <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	an Adjudicator <arg1> lifts a sentence of Defendant <arg2> at a Place <arg3>
Justice:Fine	1	<arg1> court or judge fined <arg2> at <arg3> place
	2	Adjudicator <arg1> Entity <arg2> Place <arg3>
	3	a Adjudicator <arg1> issues a financial punishment Money to an Entity <arg2> at a Place <arg3>
Justice:Trial-Hearing	1	<arg1> tried <arg2> before <arg3> court or judge in <arg4> place
	2	Prosecutor <arg1> Defendant <arg2> Adjudicator <arg3> Place <arg4>
	3	a court proceeding initiated to determine the guilty or innocence of the Defendant <arg2> Person with Prosecutor <arg1> and Adjudicator <arg3> at a Place <arg4>
Business:End-Org	1	<arg1> organization shut down at <arg2> place
	2	Org <arg1> Place <arg2>
	3	an Organization Org <arg1> goes out of business at a Place <arg2>
Justice:Sue	1	<arg1> sued <arg2> before <arg3> court or judge in <arg4> place
	2	Plaintiff <arg1> Defendant <arg2> Adjudicator <arg3> Place <arg4>
	3	Plaintiff <arg1> initiate a court proceeding to determine the liability of a Defendant <arg2> judge by Adjudicator <arg3> at a Place <arg4>
Life:Injure	1	<arg1> injured <arg2> with <arg3> instrument in <arg4> place
	2	Agent <arg1> Victim <arg2> Instrument <arg3> Place <arg4>
	3	a Victim <arg2> experiences physical harm from Agent <arg1> with Instrument <arg3> at a Place <arg4>
Justice:Arrest-Jail	1	<arg1> arrested <arg2> in <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	the Agent <arg1> takes custody of a Person <arg2> at a Place <arg3>
Justice:Execute	1	<arg1> executed <arg2> at <arg3> place
	2	Agent <arg1> Person <arg2> Place <arg3>
	3	the life of a Person <arg2> is taken by an Agent <arg1> at a Place <arg3>
Conflict:Demonstrate	1	<arg1> demonstrated at <arg2> place
	2	Entity <arg1> Place <arg2>
	3	Entity <arg1> come together in a Place <arg2> to protest or demand official action
Justice:Sentence	1	<arg1> court or judge sentenced <arg2> in <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	the punishment for the Defendant <arg2> is issued by a state actor Adjudicator <arg1> at a Place <arg3>
Life:Die	1	<arg1> killed <arg2> with <arg3> instrument in <arg4> place
	2	Agent <arg1> Victim <arg2> Instrument <arg3> Place <arg4>

(continued from previous page.)

	3	life of a Victim <arg2> ends by an Agent <arg1> with Instrument <arg3> at a Place <arg4>
Business:Start-Org	1	<arg1> started <arg2> organization at <arg3> place
	2	Agent <arg1> Org <arg2> Place <arg3>
	3	an Agent <arg1> create a new Organization Org <arg2> at a Place <arg3>
Personnel:End-Position	1	<arg1> stopped working at <arg2> organization in <arg3> place
	2	Person <arg1> Entity <arg2> Place <arg3>
	3	a Person <arg1> stops working for an Entity <arg2> or change office at a Place <arg3>
Justice:Extradite	1	<arg1> extradited <arg2> from <arg3> place to <arg4> place
	2	Agent <arg1> Person <arg2> Origin <arg3> Destination <arg4>
	3	a Person <arg2> is sent by an Agent <arg1> from Origin <arg3> to Destination <arg4>
Justice:Charge-Indict	1	<arg1> charged or indicted <arg2> before <arg3> court or judge in <arg4> place
	2	Prosecutor <arg1> Defendant <arg2> Adjudicator <arg3> Place <arg4>
	3	a Defendant <arg2> is accused of a crime by a Prosecutor <arg1> for Adjudicator <arg3> at a Place <arg4>
Transaction:Transfer-Money	1	<arg1> gave money to <arg2> for the benefit of <arg3> in <arg4> place
	2	Giver <arg1> Recipient <arg2> Beneficiary <arg3> Place <arg4>
	3	transfer Money from the Giver <arg1> to the Beneficiary <arg3> or Recipient <arg2> at a Place <arg4>
Justice:Convict	1	<arg1> court or judge convicted <arg2> in <arg3> place
	2	Adjudicator <arg1> Defendant <arg2> Place <arg3>
	3	an Defendant <arg2> found guilty of a crime by Adjudicator <arg1> at a Place <arg3>
Life:Be-Born	1	<arg1> was born in <arg2> place
	2	Person <arg1> Place <arg2>
	3	a Person <arg1> is born at a Place <arg2>

Path Spuriousness-aware Reinforcement Learning for Multi-Hop Knowledge Graph Reasoning

Chunyang Jiang[1,2], Tianchen Zhu[1,2,4], Haoyi Zhou[1,3], Chang Liu[3],
Ting Deng[1,2], Chunming Hu[1,3] and Jianxin Li[1,2]✉
[1]SKLSDE, Beihang University
[2]School of Computer Science and Engineering, Beihang University
[3]School of Software, Beihang University
[4]Shenyuan Honors College, Beihang University
`{jiangcy,zhutc,zhouhy,liuc,dengting,hucm,lijx}@act.buaa.edu.cn`

Abstract

Multi-hop reasoning, a prevalent approach for query answering, aims at inferring new facts along reasonable paths over a knowledge graph. Reinforcement learning (RL) methods can be adopted by formulating the problem into a Markov decision process. However, common suffering within RL-based reasoning models is that the agent can be biased to spurious paths which coincidentally lead to the correct answer with poor explanation. In this work, we take a deep dive into this phenomenon and define a metric named Path Spuriousness (PS), to quantitatively estimate to what extent a path is spurious. Guided by the definition of PS, we design a model with a new reward that considers both answer accuracy and path reasonableness. We test our method on five datasets and experiments reveal that our method considerably enhances the agent's capacity to prevent spurious paths while keeping comparable to state-of-the-art performance.

1 Introduction

Knowledge Graph (KG), a set of structured facts about real-world human knowledge, is utilized in numerous downstream NLP applications (Hildebrandt et al., 2020; Zhang and Yao, 2022; Xu et al., 2021; Ma et al., 2021). Common suffering affecting many downstream tasks is KG incompleteness. A variable amount of facts are missing in practical KGs. KG reasoning, the process to derive new knowledge from KG (Ji et al., 2022; Zhang et al., 2022; Huang et al., 2022), is the way to address KG completion problem. A prevailing approach for KG reasoning is incorporating KG embedding (KGE), which maps entities and relations into a vector space (Bordes et al., 2013). Embedding-based models have great power in expressing semantic similarity of entities and relations, but usually lack explainability due to the high-dimension representation (Chen et al., 2020; Heo et al., 2022).

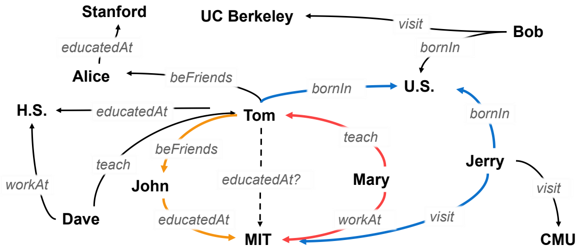


Figure 1: An example KG (we use *H.S.* to indicate a certain high school for short). Supposing the query is <*Tom, educatedAt, ?*> and *MIT* is a correct answer, three paths leading to *MIT* are of different PS.

An alternative approach is multi-hop reasoning, also referred to as path-based models (Lin et al., 2018), which infers new facts along existing paths in KG. Multi-hop reasoning offers explanations for its predictions by taking advantage of reasoning paths. Recently, reinforcement learning (RL) has been applied to multi-hop reasoning (Xiong et al., 2017). RL-based methods train an agent to walk over the KG and search for a path leading to the answer (Das et al., 2018). They have drawn surging attention in the past few years for their good prediction accuracy and excellent explainability.

However, most prevalent RL-based models are suffering from the spurious path problem (Guu et al., 2017). A spurious path reaches a correct answer merely by coincidence and has no logical relevance with its prediction, such as the blue path in Figure 1: we shouldn't say *Tom* was educated at *MIT* just because there exists one person (*i.e.*, *Jerry*) who was born in the same country (*i.e.*, *U.S.*) with *Tom* and happened to visit *MIT* before.

Multi-hop reasoning is a typical sparse reward scenario, where all actions except the final one will get no feedback during the decision process. That is to say, after the agent reached the correct answer following a spurious path, all actions along the decision trajectory will get positive rewards even if they are totally irrelevant to the query. As a

---

✉ The corresponding author is Jianxin Li.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3181–3192
May 2-6, 2023 ©2023 Association for Computational Linguistics

consequence, the agent will be biased toward those wrong actions (Guu et al., 2017).

The spurious path problem causes severe damage to the explainability of RL-based models when paths found by the agent are needed to serve as evidence to explain the answer. Moreover, spurious paths may mislead the agent to learn a wrong policy and further harm the generalization ability of the model. Although some studies have noticed the spurious path problem and provided instinctive solutions, such as action drop (Lin et al., 2018) and rule guider (Lei et al., 2020; Hou et al., 2021), a quantitative estimation is still absent.

To address the above-mentioned problem, we define a new metric called *Path Spuriousness* (PS) to measure to what extent a path is spurious. The inspiration is that a spurious path is not a fake path, but an adventitious path offering the accurate answer for a certain query. That is to say, if we randomly exchange intermediate entities of the path and keep the relation order, it is unlikely to get the correct prediction for the same query. With PS, we can reflect the reasonableness of predictions made by multi-hop reasoning models. Specifically, answers obtained by following paths with low PS are much more reasonable and explainable than those by following paths with high PS.

With the definition of PS, we put forward a path spuriousness-aware reward for RL-based multi-hop reasoning models. Combined with correctness-guided rewards, our new reward leads the agent to not only obtain effective answers but also offer high-quality reasoning paths.

Our major contributions are concluded as follows: (1) We first propose a quantitative metric PS to measure the spuriousness of reasoning paths. (2) We design a new sophisticated reward shaping method by incorporating correctness-guided reward and PS-guided reward, which leads the agent to find effective answers following reasonable paths. (3) We first offer an empirical evaluation of the path spuriousness of multi-hop reasoning methods. Experiments show that our approach has a great improvement in avoiding spurious paths while keeping the prediction accuracy.

2 Related Work

In this section, we give outlines of two main related areas and discuss their connection to our method.

2.1 Knowledge Graph Embedding

KG embedding translates semantic features of entities and relations to vector space, to give answers directly by operations over query vectors. TransE (Bordes et al., 2013) did seminal work in leveraging KG embedding to solve the QA problem and becomes the base model for a series of algorithms (Ji et al., 2015; Wang et al., 2014; Lin et al., 2015). DisMult (Yang et al., 2015) proposes a unified learning framework for embedding models and introduces an approach to mine logic rules with learned relation embeddings. ComplEX (Trouillon et al., 2016) uses complex vectors to represent entities and relations, to handle asymmetric relations. ConvE (Dettmers et al., 2018) takes advantage of a convolutional neural network for knowledge embedding in a large graph. Despite the powerful representational ability shown by embedding-based models, they are limited in many scenarios for the lack of explainability (Roscher et al., 2020; Liu et al., 2017), as one-hop reasoning methods.

A promising approach is incorporating KG embedding as a reward shaping function for multi-hop reasoning methods, which is first adopted by MultiHop (Lin et al., 2018) and followed by us.

2.2 Multi-Hop Reasoning

Compared to embedding-based models, multi-hop reasoning models predict by inferring a path step by step. The property is exactly desired in scenarios where not only an answer is required, but also evidence is demanded to explain the answer. Reinforcement learning algorithms can be naturally deployed to multi-hop reasoning. DeepPath (Xiong et al., 2017) first takes REINFORCE as the generator of evidence paths which are fed to PRA (Brin, 1998) subsequently. MINERVA (Das et al., 2018) takes the lead to design an end-to-end RL-based multi-hop reasoning model to address query answering, whereas its accuracy of answering still falls behind state-of-the-art embedding models. MultiHop (Lin et al., 2018) incorporates pre-trained embedding-based models as a soft reward function to compensate for the incompleteness of KG and gets comparable performance to embedding-based models in several datasets.

Since the agent lacks logical insights into adopted paths, spurious paths are inevitable during training. Once a spurious path leading to the correct answer coincidentally is explored first, the agent will increasingly tend to choose actions following a spurious way (Guu et al., 2017). To address this problem, MultiHop (Lin et al., 2018) uses action drop to force the agent to explore diverse paths, hoping to mitigate the negative effect of spurious paths. RARL (Hou et al., 2021) exploits high-quality rules mined from datasets to supervise the policy of the agent. However, there is no quantitative measure for the effect of the above methods, due to the absence of the metric that scales path spuriousness and consequently no reward that can directly guide the agent to avoid spurious paths.

3 Preliminaries

In this section, we first give crucial definitions of several concepts. Then the reinforcement learning formula and reward shaping skills will be introduced, which is the foundation of our model.

3.1 Multi-hop KG Reasoning

Knowledge Graph. A knowledge graph (KG) is a directed graph $G = (\mathcal{E}, \mathcal{R}, \Psi)$ (Ji et al., 2022), where $\mathcal{E}$ is the set of entities, $\mathcal{R}$ is the set of relations and Ψ is the mapping from relations to pairs of entities. A fact in knowledge graph G is an ordered triple $\delta =< e_s, r, e_o >$ satisfying that $e_s, e_o \in \mathcal{E}$, $r \in \mathcal{R}$, and $\Psi(r) =< e_s, e_o >$.

In this paper, we treat each edge in a KG as bidirectional and augment the KG with reversed edge (e_o, r^{-1}, e_s) if (e_s, r, e_o) is a fact in G. We call r^{-1} the inverse relation of r.

Multi-Hop Reasoning. Given a query $q =< e_s, r_q, ? >$ ($<?, r_q, e_o >$ is the same) and a KG G, multi-hop reasoning is to predict the absent object by finding an n-hop path $\tau =< e_s, r_1, e_2, r_2, \cdots, e_n, r_n, e_T >$ in G (Wan and Du, 2021), where the last entity e_T is the predicted answer. The path can also be represented as a predicate $\tau \equiv r_1(e_s, e_2) \wedge r_2(e_2, e_3) \wedge \cdots \wedge r_n(e_n, e_T)$.

Path Clause. For convenience, we use the path clause $H(\tau, q) \equiv \tau \rightarrow r_q(e_s, e_T) \equiv r_1(e_s, e_2) \wedge r_2(e_2, e_3) \wedge \cdots \wedge r_n(e_n, e_T) \rightarrow r_q(e_s, e_T)$ to indicate a reasoning path τ and its prediction, where τ is the clause body and $r_q(e_s, e_T)$ is the head.

In this paper, we assume that all facts in KG are correct. Then path clause $H(\tau, q)$ is true in KG G if τ is a path in G and $r_q(e_s, e_T)$ is a fact in G. We call $H(\tau, q)$ is *valid* in G if τ is a path in G.

Path Substitution. Path substitution of $H(\tau, q)$ is defined as $H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T} \equiv r_1(e_s, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_n(x_n, x_T) \rightarrow r_q(e_s, x_T)$, which is a path clause derived by replacing each e_i in clause $H(\tau, q)$ with x_i except e_s, *i.e.,* keeping e_s fixed.

We denote the body of $H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}$ as $B_{H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}}$. We call $H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}$ is a *valid* path substitution of $H(\tau, q)$ in KG G if the body $B_{H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}}$ is a path in G.

Note that there may be overlaps of entities of a path substitution with the original path clause. That is, a path substitution may not replace all entities in a path clause with new entities. In particular, each path clause $H(\tau, q)$ is a path substitution of itself.

Consider the KG in Figure 1. For query <*Tom, educatedAt, ?*>, the path in color red indicates a prediction <*Tom, educatedAt, MIT*>, and the path clause H=*teach*$^{-1}$*(Tom, Mary)* $\wedge$ *workAt(Mary, MIT)* $\rightarrow$ *educatedAt(Tom, MIT)*. Clearly, $H^{Mary,MIT}_{Dave,H.S.}$, i.e., *teach*$^{-1}$*(Tom, Dave)* $\wedge$ *workAt(Dave, H.S.)* $\rightarrow$ *educatedAt(Tom, H.S.)* is a valid path subsituition of H in the KG.

3.2 RL Formula

The RL algorithms can be naturally deployed to multi-hop reasoning by formulating it into a Markov decision process (MDP) (Puterman, 1994). Following MINERVA (Das et al., 2018), we adopt REINFORCE (Williams, 1992) algorithm. Key components of the architecture are as follows.

States. A state encodes the status quo as well as the origin goal. In step t, state $s_t = (e_t, e_s, r_q)$ is a triple, where e_t is the current entity, e_s is the start entity and r_q is the query relation.

Actions. Action space $\mathcal{A}_t = \{(r, e') | (e_t, r, e') \in G\}$ in state s_t consists of all pairs of outgoing edges and corresponding entities. That is, the valid actions are neighborhoods of the current entity e_t.

Transition. After the agent makes its decision $a_{t+1} = (r, e')$, the state of environment migrates from $s_t = (e_t, e_s, r_q)$ to $s_{t+1} = (e', e_s, r_q)$, on condition $(e_t, r, e') \in G$.

Policy. Agent's policy maps states to actions, which is usually implemented by the deep neural network. To exploit history information, the long short-term memory network (LSTM) (Hochreiter and Schmidhuber, 1997) is adopted. History information h_t in step t is calculated by LSTM, taking as input previous history h_{t-1} and the last action $a_t = (r_t, e_t)$, as follows:

$$\mathbf{h}_0 = \mathrm{LSTM}(0, \mathbf{a}_0) \tag{1}$$

$$\mathbf{h}_t = \mathrm{LSTM}(\mathbf{h}_{t-1}, \mathbf{a}_t), \quad t > 0. \tag{2}$$

Here $\mathbf{a}_t$ is the embedding of action a_t and $\mathbf{h}_t$ is the embedding of history h_t . Policy network π_θ gives possibility distribution over $\mathcal{A}_t$:

$$\pi_\theta(a_{t+1}|s_t) = \sigma(\mathbf{A}_t\mathbf{W_2}\mathrm{ReLU}(\mathbf{W_1}[\mathbf{h}_\mathrm{t};\mathbf{e}_\mathrm{t};\mathbf{r}_\mathrm{q}])), \tag{3}$$

where $\mathbf{A}_t$ is a matrix stacking all action embeddings, σ is *sigmoid* function and *ReLU* is rectified linear unit (Nair and Hinton, 2010). The agent chooses an action subjecting to the distribution given by the policy network.

Reward. The default binary reward R_b indicates whether the agent arrives at the correct answer. With the advantage of indicator function $\mathbb{I}$, binary reward can be written as follow:

$$R_b(e_T) = \mathbb{I}((e_s, r_q, e_T) \in G). \tag{4}$$

Optimization. To get the optimal parameters, the policy network is trained by maximizing the objective function:

$$J(\theta) = \mathbb{E}_{(e_s,r_q,e_o)\in G_q}\mathbb{E}_{(a_1,a_2,\cdots,a_T)\sim\pi_\theta}[R_T]. \tag{5}$$

$$R_T = R(s_T|s_1 = (e_s, e_s, r_q)). \tag{6}$$

$J(\theta)$ is the expected reward for all queries following policy π_θ. G_q denotes all query facts. REINFORCE is deployed to solve this optimization problem by iteratively updating θ as follows:

$$\nabla_\theta J(\theta) \approx \sum_{t=1}^{T} R_T \nabla_\theta \log \pi_\theta(a_t|s_t) \tag{7}$$

$$\theta' = \theta + \nabla_\theta J(\theta). \tag{8}$$

3.3 Reward Shaping

Under binary reward R_b, the agent receives a positive reward only when it reaches a correct answer. That is to say, the agent probably can't get any effective guidance in the early exploration stage.

To offset the incompleteness of KG, Multihop (Lin et al., 2018) proposes a reward shaping function R_s that provides a soft reward F_s between 0 and 1 other than 0 as R_b does, when the prediction is not in KG during the training process. $F_s(e_s, r_q, e_T)$ is implemented by a pre-trained embedding-based model such as ConvE (Dettmers et al., 2018) and ComplEX (Trouillon et al., 2016) to estimate the probability that (e_s, r_q, e_T) is true. Then R_s is defined as follows:

$$R_s(e_T){=}R_b(e_T){+}(1{-}R_b(e_T)){\cdot}F_s(e_s, r_q, e_T). \tag{9}$$

That is, given the answer e_T, if $(e_s, r_q, e_T) \in G$, the agent will get 1 as the final reward, otherwise the agent will get $F_s(e_s, r_q, e_T)$ calculated by a pre-trained embedding-based model.

4 Methodology

In this section, we discuss the feature of spurious paths and give a quantitative definition of PS. Then we introduce two kinds of reward functions that take PS into account.

4.1 Path Spuriousness Metric

Think over the example in Figure 1. Assuming <*Tom, educatedAt, MIT*> is true and can be deduced by the following three paths H_1, H_2, and H_3 (in color red, blue, and yellow, respectively):

(1) H_1≡*teach*$^{-1}$*(Tom, Mary)* ∧ *workAt(Mary, MIT)* → *educatedAt(Tom, MIT)*;

(2) H_2≡*bornIn(Tom, U.S.)*∧*bornIn*$^{-1}$*(U.S., Jerry*∧*visit(Jerry, MIT)*→*educatedAt(Tom, MIT)*;

(3) H_3≡*beFriends(Tom, John)*∧*educatedAt(John, MIT)*→*educatedAt(Tom, MIT)*.

However, not all of them can serve as valid evidence. Clearly, H_1 is solid enough since *Tom* must be educated exactly where his teacher teaches. H_2 is extremely spurious because there are hundreds of millions of people born in the same place as *Tom* and they visit tremendous schools, few of which *Tom* can attend. Though H_3 is spurious in logic, considering that many of *Tom*'s friends made acquaintances with him when they were in the same school, H_3 is much more valid than H_2.

It is a remarkable fact that these three clauses are all *true* in the KG of Figure 1, despite their variance in path spuriousness. The truth value of a valid path clause always keeps consistent with its prediction, regardless of whether its body is a reasonable path or not. The key difference between spurious paths and reasonable paths is that they have quite a few valid substitutions in a given KG, but fail to infer correct predictions. That is, the path spuriousness of a path clause H is up to the proportion of such valid substitutions. The larger the proportion is, the more spurious H is.

Path Spuriousness. Given a KG G and a path clause $H \equiv r_1(e_s, e_2) \wedge r_2(e_2, e_3) \wedge \cdots \wedge r_n(e_n, e_T) \to r_q(e_s, e_T)$, the PS of H is:

$$\mathrm{PS}(H) = \mathbb{P}(v(H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}) = 0), \tag{10}$$

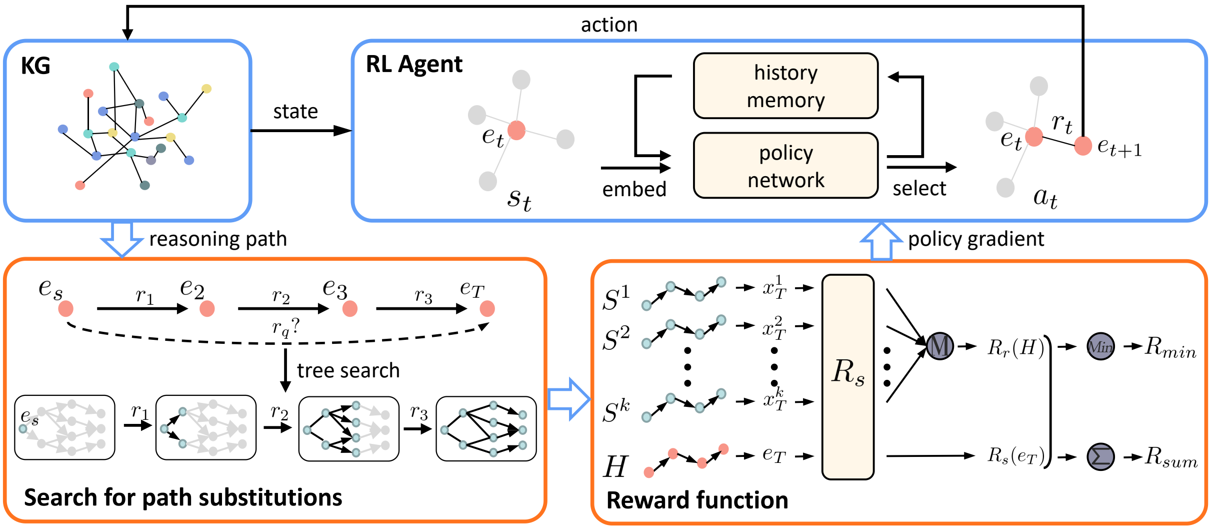


Figure 2: An overlook of the training process. Starting from entity e_s, the agent walks step by step to obtain a reasoning path H where e_T denotes the predicted answer. All substitutions of H are derived by searching over the KG. We use S^i to represent the i-th substitution. R_{min} and R_{sum} indicate our two reward shaping variants.

which is equal to:

$$\mathrm{PS}(H) = 1 - \mathbb{E}(v(H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T})) \tag{11}$$

where $H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}$ is a valid substitution of H, v is the value function mapping predicate space to $\{0,1\}$, and $X = (x_2, \cdots, x_n, x_T)$ are random entities sampled from the ideal entity set. In practice, the ideal entity set is usually unknown and hard to estimate, so we propose a computation viable metric based on frequency to approximate Equation 11:

$$\mathrm{PS}(H) \approx 1 - \frac{\sum\limits_{x_2,\cdots,x_n,x_T \in \mathcal{E}} v(H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T})}{\sum\limits_{x_2,\cdots,x_n,x_T \in \mathcal{E}} v(B_{H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}})} \tag{12}$$

where $B_{H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}}$ is the body part of $H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}$, and $\mathcal{E}$ is the entity set of G.

Take the KG in Figure 1 for example.

(1) H_1 (in color red) has only one valid substitution *teach*$^{-1}$*(Tom, Dave)* $\wedge$ *workAt(Dave, H.S.)* $\rightarrow$ *educatedAt(Tom, H.S.)* except H_1, which is true in the KG. Counting H_2 itself in, PS(H_2)=1−2/2=0.

(2) H_2 (in color blue) has two valid substitutions except H_1: *bornIn(Tom, U.S.)* $\wedge$ *bornIn*$^{-1}$*(U.S.,Jerry)* $\wedge$ *visit(Jerry,CMU)* $\rightarrow$ *educatedAt(Tom, CMU)* and *bornIn(Tom, U.S.)* $\wedge$ *bornIn*$^{-1}$*(U.S.,Bob)* $\wedge$ *visit(Bob,UC Berkeley)* $\rightarrow$ *educatedAt(Tom, UC Berkeley)*, but both of them are false in the KG. Then we have PS(H_1)=1−1/3=2/3.

(3) H_3 (in color yellow) has one valid substitution *beFriends(Tom, Alice)* $\wedge$ *educatedAt(Alice, Stanford)* $\rightarrow$ *educatedAt(Tom, Stanford)* except H_1, which is false in the KG. As a result, PS(H_3)=1−1/2=1/2.

4.2 Path Spuriousness-Based Reward

Both R_b and R_s are devoted to the correctness of the terminal entity, ignoring the spuriousness of reasoning paths. To address this, we design two novel rewards combining the answer correctness as well as the path spuriousness.

First, a path score function $F_p(H)$ is required to score the spuriousness degree of H. The most straightforward way is taking the definition of PS as F_p, which is effective when KG is closed (Tanon et al., 2017). However, taking the incompleteness of KG into consideration, we incorporate the soft reward R_s with Equation 12 to build $F_p(H)$:

$$F_p(H) = 1 - R_r(H) \tag{13}$$

$$R_r(H) = \frac{\sum\limits_{x_2,\cdots,x_n,x_T \in \mathcal{E}} v(B_{H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}}) R_s(x_T)}{\sum\limits_{x_2,\cdots,x_n,x_T \in \mathcal{E}} v(B_{H^{e_2,\cdots,e_n,e_T}_{x_2,\cdots,x_n,x_T}})}. \tag{14}$$

For simplicity, we use $R_r(H) = 1 - F_p(H)$ to indicate the reasonableness of H.

Because of the complexity of semantic knowledge, even a reasonable path may lead to a wrong answer in some special cases. Therefore, a rational reward has to be a fusion of the answer accuracy and the path reasonableness. We demonstrate two kinds of combinations between R_s and R_r, and discuss their performance in the experiment section.

One is taking the *minimum* between accuracy and reasonableness:

$$R_{min}(H, e_T) = min(R_s(e_T), R_r(H)). \quad (15)$$

The equation is based on intuitive thinking that, we introduce PS-based reward only for punishing spurious paths, but not for awarding paths with high reasonableness but low accuracy. That is, R_r merely serves as punishment when it is lower than R_s. Note that R_{min} is friendly to developers since it has no hyperparameters.

The other is the *weighted sum* of R_s and R_r:

$$R_{sum}(H, e_T) = \alpha R_r(H) + (1-\alpha) R_s(e_T), \quad (16)$$

where α decreases in the process of training. This equation is inspired by curriculum learning, which sets different optimizing goals in different learning stages (Bengio et al., 2009). We give priority to the path reasonableness in early epochs, forcing the agent to focus more on reasonable paths, and after we prefer the agent to concentrate on proper answers. Curriculum learning has the chance to achieve higher performance but makes it more difficult to tune parameters.

4.3 Overall Training Process

The overall training process of our method is shown in Figure 2. The agent starts from the query entity e_s and makes transitions according to its policy network. The history memory holds the history information of each past step. After the agent reaches the final entity e_T, we use Breadth-First Search to get all valid substitutions of its reasoning path. Then all substitutions as well as the reasoning path itself are used to calculate R_r and R_s which are combined together to get either R_{min} or R_{sum}.

Our PyTorch implementation and some pretrained models are released at `https://github.com/rubickkcibur/PSAgent`.

5 Experiment

We evaluate our model on four datasets and compare it with eight common baseline models. *Ours+min* and *Ours+sum* indicate our two approaches with R_{min} and R_{sum} respectively.

5.1 Setup

5.1.1 Datasets

We use five benchmark datasets for query answering: 1) UMLS (Kok and Domingos, 2007), 2) Kinship (Lin et al., 2018), 3) FB15K-237 (Toutanova et al., 2015), 4) WN18RR (Dettmers et al., 2018), and 5) NELL-995 (Xiong et al., 2017). Statistics are shown in Table 1.

Dataset	#Ent	#Rel	#Fact	#degree	
				avg.	median
Kinship	104	25	10,686	82.2	82
UMLS	135	46	6,529	38.6	28
FB15K-237	14,505	237	272,115	19.7	14
WN18RR	40,945	11	93,003	2.2	2
NELL-995	10,105	12	13,825	1.6	1

Table 1: Statistics of five KGs used in experiments.

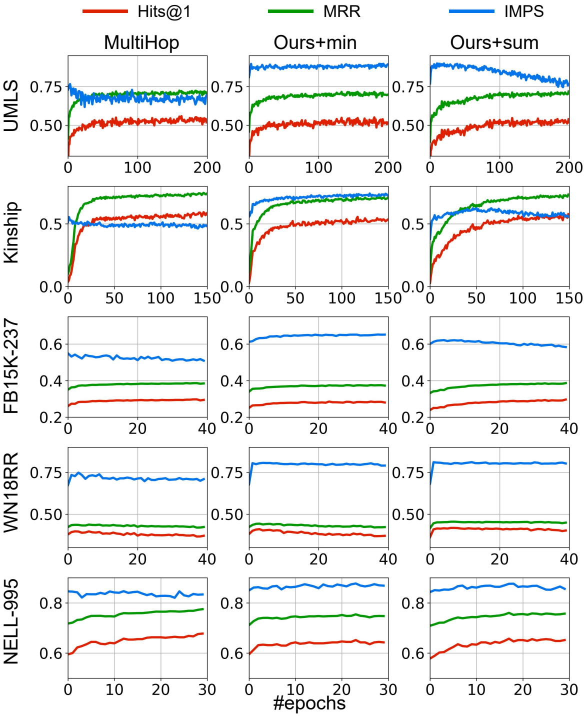


Figure 3: Hits@1 (red), MRR (green), and IMPS (blue) change over the validation set during training.

5.1.2 Baselines

We compare our method with five multi-hop reasoning models: 1) MINERVA (Das et al., 2018), the first end-to-end deep reinforcement learning model for multi-hop reasoning; 2) MultiHop (Lin et al., 2018), the first RL-based model incorporating embedding-based models as reward shaping function and proposing the action drop skill to mitigate spurious path problem; 3) MetaKGR (Lv et al., 2019), which leverages meta-information to improve reasoning performance on few-shot relations; 4) RARL (Hou et al., 2021), which utilizes mined logic rules to supervise the decision of the agent; 5) PAAR (Zhou et al., 2021), a fresh model com-

Dataset	UMLS				Kinship				FB15K-237				WN18RR				NELL-995			
	@1	@10	MRR	IMPS	@1	@10	MRR	IMPS	@1	@10	MRR	IMPS	@1	@10	MRR	IMPS	@1	@10	MRR	IMPS
DisMult	82.1	96.7	86.8	N/A	48.7	90.4	61.4	N/A	32.4	60.0	41.7	N/A	43.1	52.4	46.2	N/A	55.2	78.3	64.1	N/A
ComplEX	89.0	99.2	93.4	N/A	81.8	98.1	88.4	N/A	32.8	61.6	42.5	N/A	41.8	48.0	43.7	N/A	64.3	86.0	72.6	N/A
ConvE	93.2	99.4	95.7	N/A	79.7	98.1	87.1	N/A	34.1	62.2	43.5	N/A	40.3	54.0	44.9	N/A	67.8	88.6	76.1	N/A
MINERVA	78.5	97.4	85.9	68.6	59.6	95.8	73.0	49.7	26.0	40.7	31.3	57.5	43.1	52.9	46.0	71.9	51.6	79.6	62.3	81.4
MultiHop	**90.5**	**99.5**	94.2	65.3	78.8	**98.6**	86.8	48.4	**32.7**	56.4	40.7	50.6	41.2	52.0	45.1	73.9	65.0	82.9	71.9	79.1
MetaKGR	88.6	99.3	93.1	65.1	78.2	98.4	86.3	47.6	28.3	52.7	36.9	53.5	36.5	50.9	42.0	69.6	59.4	78.7	66.7	80.3
RARL	76.2	95.6	84.2	64.5	61.3	94.4	73.3	49.6	28.4	49.7	35.8	54.4	40.0	51.7	44.6	74.0	62.8	82.2	70.4	81.4
PAAR	89.5	**99.5**	94	67.2	72.7	96.3	81.4	47.5	32.1	55.0	40.0	55.3	40.6	**53.8**	44.9	73.8	65.2	83.9	72.2	77.3
Ours+min	89.1	98.9	93.2	**87.3**	76.2	98.1	85.0	**74.4**	30.9	56.1	39.5	**64.1**	42.1	50.9	45.1	**81.9**	**65.7**	**85.3**	**73.5**	**81.5**
Ours+sum	**90.5**	**99.5**	**94.6**	71.3	**79.1**	98.1	**86.9**	53.7	32.5	**57.0**	**40.9**	58.4	**43.4**	52.6	**46.8**	81.4	63.8	84.1	71.4	78.1
AMIE+	53.4	72.6	61.0	**92.1**	58.2	75.7	66.0	**76.7**	13.8	23.5	16.8	**68.8**	34.8	36.6	35.6	**99.2**	51.2	51.5	51.4	**95.1**

Table 2: QA Performance comparison on five datasets. The top part is embedding-based models and the bottom is RL-based models. *Ours+min* and *Ours+sum* denote our methods with R_{min} and R_{sum}, respectively. All metrics are multiplied by 100. IMPS is not applicable to embedding-based models because they lack reasoning paths.

bining hierarchical information in reward shaping for providing sufficient paths. Three embedding-based models are also picked up as comparisons: DistMult (Yang et al., 2015), ComplEX (Trouillon et al., 2016), and ConvE (Dettmers et al., 2018).

5.1.3 Hyperparameters

The entity embedding, relation embedding, and history embedding all have a size of 200. A three-layer LSTM is used for multi-hop reasoning models. The training batch size is 128. The maximum path length is 2 for UMLS and Kinship, and 3 for others. Following MultiHop (Lin et al., 2018), we use action drop in the training process, and the drop rate ranges from 0.1 to 0.9. For R_{sum}, there are two hyperparameters, decreasing interval D and decreasing rate η. We perform grid search on them and set $D = 10$ for UMLS and WN18RR, $D = 5$ for others, $\eta = 0.75$ for FB15K-237, and $\eta = 0.9$ for the other four datasets.

For MultiHop, MetaKGR, PAAR, and our method, we universally choose ConvE to implement the soft reward function. An entropy regularization term is added to the objective function in all RL-based models and the weight coefficient varies in (0, 0.1), as MINERVA does (Das et al., 2018).

We use Xavier initialization (Glorot and Bengio, 2010) to initialize parameters of embedding layers, and Adam optimizer (Kingma and Ba, 2015) to realize optimization where the learning rate is in (0.001, 0.003).

We perform beam search to get the final prediction and the beam size is 128 for all cases. A single training on NVIDIA Tesla V100 GPU costs 20 hours on FB15K-237, and at most 10 hours among all other datasets.

5.1.4 Evaluation Protocol

We choose Hits@k and Mean Reciprocal Rank (MRR) to evaluate the accuracy of predictions, and use Mean Path Spuriousness (MPS) to estimate the spuriousness of paths. For consistency, we use 1-MPS in measurement, denoted by IMPS.

For each test case $< e_s, r, e_o >$, the model takes as input the subject e_s and relation r, and returns a list of candidate answers $E_o = [e^1, e^2, \cdots, e^N]$ in decreasing order of confidence, as well as a list of corresponding reasoning paths $\mathcal{H}_o = [H^1, H^2, \cdots, H^N]$, where N is the beam size, and the termination of each path H^i is e^i for $i \in [1, N]$.

We use r_{e_o} to indicate the rank of e_o in E_o and H_o to represent the path along which the agent reaches e_o. Hits@k is the percentage of test cases where $r_{e_o} \leq k$, MRR is the mean of $1/r_{e_o}$, and MPS is the mean of $\text{PS}(H_o)$.

5.2 Validation of the IMPS

To verify whether the IMPS metric (*i.e.* 1-MPS) could correctly evaluate the reasonableness of reasoning paths, we conduct experiments on rule-based reasoning models. Rule-based models mostly lack generalization but have highly credible results, since they use logical rules, either designed by experts or mined from datasets, to extract reasoning paths and infer new facts. That is to say, if the IMPS metric is a proper measurement of path reasonableness, rule-based models will perform beyond other models on it.

We choose AMIE+ (Galárraga et al., 2015) to mine rules and then infer answers following them. The results (the final bar of Table 2) meet our expectations. AMIE+ gets the highest IMPS scores on four datasets, exceeding all RL-based models. Compared to other RL-based ones, the gap between our model and AMIE+ is evidently smaller.

5.3 Model Comparison

Table 2 shows evaluation performance on query answering. Results of embedding-based models are quoted from MultiHop (Lin et al., 2018), and IMPS is not applicable to them because they have no reasoning paths. *Ours+min* and *Ours+sum* denote two variants of our approach with R_{min} and R_{sum}, respectively. For all approaches, We record performances on the validation set of each epoch during training and choose the best one (in terms of MRR) as the testing model.

In terms of Hits@k and MRR, embedding-based models generally perform better, while multi-hop approaches are comparable with embedding-based methods in some metrics, such as Hits@10 in UMLS, Hits@1 in Kinship and MRR in WN18RR. Among multi-hop reasoning models, *Ours+sum* outperforms previous ones in most cases.

As for IMPS metric, our methods largely surpass other RL-based models, Specifically, 27 percent in UMLS, 50 percent in Kinship, 10 percent in WN18RR, and 11 percent in FB15K-237. An exception is NELL-995, where most models behave nearly the same. It possibly results from the sparsity of NELL-995. Since the average node degree is lower than 2, most reasoning paths in NELL-995 may have few valid substitutions.

An interesting fact is that performance varies between our two methods. Generally, *Ours+min* gets higher IMPS scores, and *Ours+sum* behaves better on Hits@k and MRR. We believe that when the reasoning model's performance is not good enough, prediction accuracy and reasonableness keep consistent, but if higher performance is required, there is a trade-off between them. In most cases, reasonable paths lead to accurate answers, but in some special cases, the answer can not be accessed in a regular way. For example, the statement "*if A is metal, A is solid*" is reasonable for almost all metallic elements, but when A is mercury, following that will lead to a wrong answer. The little fallback in MRR of *Ours+min* may be caused by these special cases, and the more special cases in a dataset, the larger the gap is. *Ours+sum* formulation provides a manual way to make a balance, so we can get better MRR at the expense of IMPS.

5.4 Learning Process

We are interested in the effect on the dynamic learning process of three different rewards R_s, R_{min}, and R_{sum}. So we draw curves of Hits@1, MRR, and IMPS metrics on validation set during training. As shown in Figure 3, from left to right, the three columns are MultiHop, *Ours+min*, and *Ours+sum*, using R_s, R_{min}, and R_{sum} respectively.

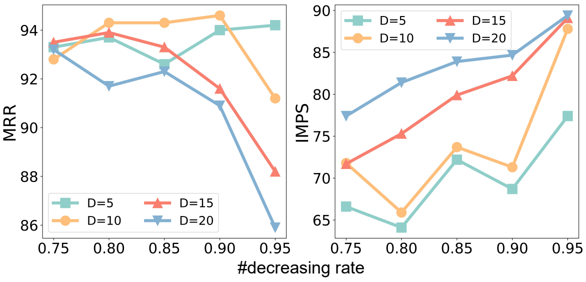


Figure 4: MRR (left) and IMPS (right) performance change w.r.t. decreasing interval D and rate η of R_{sum}. Each line represents metric variety over various decreasing rates for a certain decreasing interval.

In the early stage of training, the MultiHop's IMPS score falls as MRR and Hits@1 rise, which is much more significant on UMLS and Kinship. This phenomenon evidently shows the misleading of spurious paths. As a comparison, our two approaches have a higher IMPS score at the beginning and maintain or elevate it as MRR increases. As illustrated, the promotion of reasoning paths reasonableness by incorporating PS-aware reward (*i.e.* R_{min} and R_{sum}) is mainly reflected in the early training process.

Compared to *Ours+min*, the IMPS score of *Ours+sum* first lifts and then slowly decreases while the MRR score keeps rising. It justifies that prediction accuracy and path reasonableness are in consistency when performance is not sufficiently good, but need a trade-off if better performance is required. However, this balance is not obvious in WN18RR, where *Ours+sum* gets great scores in both MRR and IMPS with proper hyperparameters. We suppose one reason is that the proportion of special cases which can not be accessed by regular logical path is small.

MultiHop and *Ours+min* show a similar training convergence rate, while *Ours+sum* converges slower. The successive change of the coefficient α in R_{sum} makes the agent learn different goals and therefore hard to converge.

5.5 Hyperparameter Study

We study the hyperparameters relevant to R_{sum}, specifically the decreasing interval D and decreasing rate η. Figure 4 shows MRR and IMPS results on UMLS with 20 permutations of D and η. We

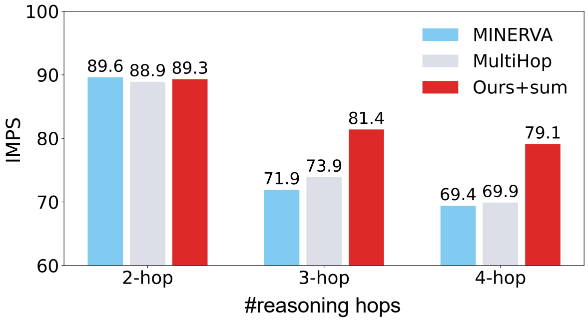


Figure 5: IMPS comparison among MINERVA (blue), MultiHop (gray), and our method (red), over different reasoning hops. IMPS metric is multiplied by 100.

figure out that when decreasing interval is small, MRR varies little with decreasing rate, and vice versa. It is quite fair because smaller D or η implies more priority to the accuracy-guided part of R_{sum} during training, which prompts better performance on MRR. However, settings of middle D and η achieve the best score, and we believe it reveals that proper tendency to the spuriousness-guided part of R_{sum} in the early training stage has a positive influence over accuracy. In terms of IMPS, it declines on the whole as η becomes smaller, and the larger D is, the slower it falls. Generally, in small and dense datasets, small decreasing intervals and rates would lead the agent to get higher scores on MRR, while larger intervals and rates make the agent prefer more reasonable paths. A balance where the agent gets high scores on both MRR and IMPS exists with refined parameters.

5.6 Analysis on Reasoning Hops

We also want to throw some light on the influence of the maximum length of reasoning paths (i.e., reasoning hops). We permute reasoning hops and test performance in IMPS of three models, MINERVA, MultiHop, and ours. We pick WN18RR as the benchmark because UMLS and Kinship are so dense that all entities can be reached within 2 hops starting anywhere. As Figure 5 shows, the reasonableness of paths picked by three models drops as reasoning hops increase, without exception. Compared to baseline models, our method has a better resistance against the tendency. Thereby, the gap between our method and baselines widens as reasoning hops go up.

6 Conclusion

In this paper, we discuss the spurious path problem which widely exists in the RL-based multi-hop reasoning models. To address this problem, we define a new metric named Path Spuriousness (PS) to quantitatively evaluate to what extent a path is spurious and consequently propose a new reward that considers both the prediction accuracy and path reasonableness. Under the guidance of the reward, the agent can be aware of not only whether its prediction is right, but also the spuriousness of its reasoning path, and thus avoid spurious paths.

Experiments show our method largely outperforms baseline models in terms of PS, and keep comparable to the state-of-the-art performance of prediction accuracy. Detailed analysis indicates that a trade-off between pursuing better prediction accuracy and keeping high path reasonableness exists. Its significance varies among different datasets. We provide a method to make a balance by manually pruning hyperparameters.

Analysis on reasoning hops shows potential in long-hop reasoning tasks. In future work, we would like to further investigate it.

Limitations

Our job has two major limitations regarding the definition of PS and the computational cost.

The definition of PS (*i.e.*, Equation 10 and Equation 11) is constructed on the assumption that all facts in a KG are correct. However, some datasets contain mistaken facts in practice. It could make the PS a biased estimation. Moreover, experiments on NELL-995 indicate that the sparsity of KG may limit the effectiveness of our PS-aware reward. How to avoid spurious paths in sparse KGs and even KGs that contain mistakes remains a hard topic for future work.

The other deficiency is that we use a breadth-first tree search to find path substitutions, whose worst-case time cost increases exponentially as the search depth grows. This shortcoming has been reflected by the experiment time cost on FB15K-237 and limits our method's capability to scale to much larger datasets. We expect pruning skills or end-to-end deep models would be able to tackle this problem and leave it for future work.

Acknowledgements

This work is supported by National Key R&D Program of China (2021ZD0113903). We also thanks for the computing infrastructure provided by Beijing Advanced Innovation Center for Big Data and Brain Computing.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*, pages 2787–2795.

Sergey Brin. 1998. The pagerank citation ranking: bringing order to the web. *Proceedings of ASIS, 1998*, 98:161–172.

Xiaojun Chen, Shengbin Jia, and Yang Xiang. 2020. A review: Knowledge reasoning over knowledge graph. *Expert Syst. Appl.*, 141.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *ICLR*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*, pages 1811–1818.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2015. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6):707–730.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9 of *JMLR Proceedings*, pages 249–256.

Kelvin Guu, Panupong Pasupat, Evan Zheran Liu, and Percy Liang. 2017. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *ACL*, pages 1051–1062.

Yu-Jung Heo, Eun-Sol Kim, Woo Suk Choi, and Byoung-Tak Zhang. 2022. Hypergraph transformer: Weakly-supervised multi-hop reasoning for knowledge-based visual question answering. In *ACL*, pages 373–390.

Marcel Hildebrandt, Jorge Andres Quintero Serna, Yunpu Ma, Martin Ringsquandl, Mitchell Joblin, and Volker Tresp. 2020. Reasoning on knowledge graphs with debate dynamics. In *AAAI*, pages 4123–4131.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Zhongni Hou, Xiaolong Jin, Zixuan Li, and Long Bai. 2021. Rule-aware reinforcement learning for knowledge graph reasoning. In *ACL Findings*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 4687–4692.

Jiacheng Huang, Yao Zhao, Wei Hu, Zhen Ning, Qijin Chen, Xiaoxia Qiu, Chengfu Huo, and Weijun Ren. 2022. Trustworthy knowledge graph completion based on multi-sourced noisy data. In *WWW*, pages 956–965.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL*, pages 687–696.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Stanley Kok and Pedro M. Domingos. 2007. Statistical predicate invention. In *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pages 433–440.

Deren Lei, Gangrong Jiang, Xiaotao Gu, Kexuan Sun, Yuning Mao, and Xiang Ren. 2020. Learning collaborative agents with rule guidance for knowledge graph reasoning. pages 8541–8547.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. In *EMNLP*, pages 3243–3253.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187.

Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical inference for multi-relational embeddings. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 2168–2178.

Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2019. Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations. pages 3374–3379.

Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. 2021. Cr-walker: Tree-structured graph reasoning and dialog acts for conversational recommendation. In *EMNLP*, pages 1839–1851.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814.

Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics.

Ribana Roscher, Bastian Bohn, Marco F. Duarte, and Jochen Garcke. 2020. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 8:42200–42216.

Thomas Pellissier Tanon, Daria Stepanova, Simon Razniewski, Paramita Mirza, and Gerhard Weikum. 2017. Completeness-aware rule learning from knowledge graphs. In *ISWC*, volume 10587 of *Lecture Notes in Computer Science*, pages 507–525.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1499–1509. The Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080.

Guojia Wan and Bo Du. 2021. Gaussianpath: A bayesian multi-hop reasoning framework for knowledge graph reasoning. In *AAAI*, pages 4393–4401.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pages 564–573.

Weiwen Xu, Yang Deng, Huihui Zhang, Deng Cai, and Wai Lam. 2021. Exploiting reasoning chains for multi-hop science question answering. In *EMNLP*, pages 1143–1156.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Yongqi Zhang and Quanming Yao. 2022. Knowledge graph reasoning with relational digraph. In *WWW*, pages 912–924.

Zhanqiu Zhang, Jie Wang, Jieping Ye, and Feng Wu. 2022. Rethinking graph convolutional networks in knowledge graph completion. In *WWW*, pages 798–807.

Xingchen Zhou, Peng Wang, Qiqing Luo, and Zhe Pan. 2021. Multi-hop knowledge graph reasoning based on hyperbolic knowledge graph embedding and reinforcement learning. In *IJCKG*, pages 1–9.

A Experiment Details

A.1 Case Study

We select four test cases in the UMLS dataset and look deep into the difference between reasoning paths extracted by MultiHop and our method. Table 3 shows the comparisons. In each case, we only concentrate on the accurate predictions as well as their paths. The reasonableness of paths is indicated by $R_r(H_o)$, and the rank of correct answer e_o is represented as r_{e_o}.

For most cases, the reasoning path of our method to obtain the correct answer is much more reasonable than MultiHop.

(1) Case 1. In case 1, to get the relation *location_of*, our reasoning path through relations *adjacent_to* and *location_of* is much more convinced than MultiHop's path through relations *issue_in* and *method_of*$^{-1}$. That's because *Adjacent* relation is clearly more relevant than *method* relation since we want to know where is the location.

(2) Case 2. Case 2 shows insight into the different preferences between the two approaches in respect of picking paths. The two paths differ in the last relation, where MultiHop reaches the final answer via *issue_in*$^{-1}$ while our method picks *issue_in*.

Picking *issue_in* is reasonable in this case.

(a) It is a special coincidence that *occupation_or_discipline* and *biomedical_occupation_or_discipline* are reciprocal causation in UMLS dataset. However, in terms of reasonableness, the fact "*A causes C* " cannot be derived by "*A causes B*" and "*C causes B*" (i.e., *issue_in(A, B)* $\wedge$ *issue_in*$^{-1}$*(B, C)*).

(b) Guided by the embedding-based model, MultiHop only captures the semantic information between *occupation_or_discipline* and *biomedical_occupation_or_discipline* and obtains a correct answer incidentally. In contrast, our method not only concentrates on the semantic correctness of certain entities but also takes path spuriousness into account. The extremely lower path reasonableness of *issue_in*$^{-1}$ prohibits our method from picking it to construct the reasoning path.

(3) Case 3. Though the path returned by MultiHop gets a score 0.49 of R_r, it is still lower than the path given by our method, considering that adjacent things possibly belong to different concepts, like lung and air. Relation *part_of* is definitely more proper to derive an *conceptual_part_of*relation.

(4) Case 4. In case 4, we get a reasonable path

Case 1:<*body_location_or_region, location_of, therapeutic_or_preventive_procedure*>		r_{e_o}	$R_r(H_o)$
MultiHop	*body_location_or_region* $\xrightarrow{\text{issue\_in}}$ *occupation_or_discipline* $\xrightarrow{\text{method\_of}}^{-1}$ *therapeutic_or_preventive_procedure*	3	0.22
Ours+min	*body_location_or_region* $\xrightarrow{\text{adjacent\_to}}$ *body_part_organ_or_organ_component* $\xrightarrow{\text{location\_of}}$ *therapeutic_or_preventive_procedure*	0	0.71
Case 2:<*physical_object, issue_in, biomedical_occupation_or_discipline*>		r_{e_o}	$R_r(H_o)$
MultiHop	*physical_object* $\xrightarrow{\text{issue\_in}}$ *occupation_or_discipline* $\xrightarrow{\text{issue\_in}}^{-1}$ *biomedical_occupation_or_discipline*	0	0.01
Ours+min	*physical_object* $\xrightarrow{\text{issue\_in}}$ *occupation_or_discipline* $\xrightarrow{\text{issue\_in}}$ *biomedical_occupation_or_discipline*	0	0.99
Case 3:<*cell_component, conceptual_part_of, body_system*>		r_{e_o}	$R_r(H_o)$
MultiHop	*cell_component* $\xrightarrow{\text{adjacent\_to}}$ *body_space_or_junction* $\xrightarrow{\text{conceptual\_part\_of}}$ *body_system*	3	0.49
Ours+min	*cell_component* $\xrightarrow{\text{part\_of}}$ *cell* $\xrightarrow{\text{conceptual\_part\_of}}$ *body_system*	0	0.99
Case 4:<*indicator_reagent_or_diagnostic_aid, interacts_with, chemical*>		r_{e_o}	$R_r(H_o)$
MultiHop	*indicator_reagent_or_diagnostic_aid* $\xrightarrow{\text{causes}}$ *experimental_model_of_disease* $\xrightarrow{\text{inv\_causes}}$ *chemical*	0	0.05
Ours+min	*indicator_reagent_or_diagnostic_aid* $\xrightarrow{\text{interacts\_with}}$ *hazardous_or_poisonous_substance* $\xrightarrow{\text{interacts\_with}}$ *chemical*	5	0.99

Table 3: Comparison of selected paths between MultiHop and our method on 4 cases. The reasonableness of paths is indicated by $R_r(H_o)$ and r_{e_o} represents the rank of correct answer e_o in candidate list E_o.

Dataset	HC_{min}	PCA_{min}	#Rules
UMLS	0.50	0.40	160
Kinship	0.60	0.40	74
WN18RR	0.10	0.50	34
NELL-995	0.10	0.30	8
FB15K-237	0.30	0.24	1074

Table 4: AMIE+ setting details. HC_{min} and PCA_{min} indicate the minimum head coverage and PCA confidence respectively, which is threshold for mining rules.

but with a lower rank. That is, the agent does not always pick a path with a high R_r. It may be limited to the representation power of the policy network. We leave this problem for future research.

A.2 AMIE+ Settings

To validate our proposed metric IMPS, we choose AMIE+ to mine rules and then infer answers following them. There are two hyperparameters of AMIE+: the head coverage and PCA confidence (denoted by HC_{min} and PCA_{min} respectively). The higher the two parameters are, the smaller the number of qualified mined rules. In Table 4 we show the settings of HC_{min} and PCA_{min}, and the number of qualified rules we mined using AMIE+.

Self-Adaptive Named Entity Recognition by Retrieving Unstructured Knowledge

Kosuke Nishida[†‡] **Naoki Yoshinaga**[§] **Kyosuke Nishida**[†]

[†]NTT Human Informatics Laboratories, NTT Corporation
[‡]The University of Tokyo
[§]Institute of Industrial Science, the University of Tokyo
[†]{kosuke.nishida.ap, kyosuke.nishida.rx}@hco.ntt.co.jp
[§]ynaga@iis.u-tokyo.ac.jp

Abstract

Although named entity recognition (NER) helps us to extract domain-specific entities from text (*e.g.*, artists in the music domain), it is costly to create a large amount of training data or a structured knowledge base to perform accurate NER in the target domain. Here, we propose self-adaptive NER, which retrieves external knowledge from unstructured text to learn the usages of entities that have not been learned well. To retrieve useful knowledge for NER, we design an effective two-stage model that retrieves unstructured knowledge using uncertain entities as queries. Our model predicts the entities in the input and then finds those of which the prediction is not confident. Then, it retrieves knowledge by using these uncertain entities as queries and concatenates the retrieved text to the original input to revise the prediction. Experiments on CrossNER datasets demonstrated that our model outperforms strong baselines by 2.35 points in F_1 metric.

1 Introduction

Named entity recognition (NER) helps us to extract entities from text in various domains such as biomedicine (Kim et al., 2003), disease (Doğan et al., 2014), and COVID-19 (Wang et al., 2020). However, accurate neural NER requires a massive amount of training data (Chiu and Nichols, 2016; Ma and Hovy, 2016; Yadav and Bethard, 2018). As well, the annotation of a domain-specific NER dataset costs a lot of money because it requires the involvement of domain experts.

To compensate for the lack of training data in NER, researchers have utilized external knowledge. Traditional feature-based NER uses features based on gazetteers or name lists (Florian et al., 2003; Cohen and Sarawagi, 2004; Luo et al., 2015) as external knowledge. Although recent neural NER methods can even benefit from gazetteers and name lists (Seyler et al., 2018; Liu et al., 2019; Mengge et al., 2020), only a few domains with structured knowledge bases (gazetteers) have this merit. Thus, several studies have resorted to using raw text (unstructured knowledge) to perform weakly-supervised learning on general-domain structured knowledge (Cao et al., 2019; Mengge et al., 2020; Liu et al., 2021a).

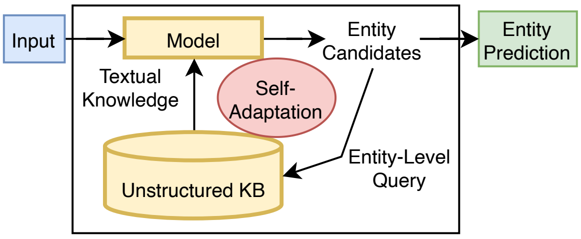


Figure 1: Concept of self-adaptive NER: the model predicts entity candidates to conduct entity-level retrieval from the unstructured KB; then it revises the prediction with reference to the retrieved knowledge.

In this paper, we explore the potential of utilizing unstructured knowledge in the NER task by referring to it at inference time. Our basic idea is inspired by recent retrieval-augmented language models (LMs) (Guu et al., 2020). These models are pre-trained with retrieval-augmented masked language model (MLM), so that they can perform well in open-domain question answering (ODQA) by retrieving relevant unstructured knowledge using a question as a query. However, as we will later confirm in the experiments, the models designed for ODQA are not effective in the NER task because it requires an understanding of many entities in the input text.

To deal with this problem, we propose a retrieval-augmented model capable of determining which entities to focus on in the input text for knowledge retrieval. The proposed **self-adaptive NER (SA-NER) with unstructured knowledge model** searches an unstructured knowledge base (UKB) when it lacks confidence in its prediction. We cre-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3193–3205
May 2-6, 2023 ©2023 Association for Computational Linguistics

ate the UKB automatically by splitting a raw text corpus into pieces and assigning dense vectors as keys to each piece of unstructured knowledge. To help in understanding local semantics, we design a retrieval system tailored for NER; our model predicts the entities and then retrieves knowledge in terms of those it is not confident in predicting.

To evaluate our method's capability of retrieving useful knowledge about entities, we conducted experiments on various NER datasets (Tjong Kim Sang and De Meulder, 2003; Salinas Alvarado et al., 2015; Liu et al., 2021b), some of which have domain-specific types.

Our contributions are summarized as follows:

- We are the first to integrate retrieval-augmentation into NER. SA-NER retrieves entity-level knowledge dynamically for NER.
- In experiments, SA-NER outperformed strong baselines pre-trained in a supervised and self-supervised fashion by 1.22 to 2.35 points.
- We reveal why knowledge retrieval is useful for NER. We found that our model is effective on entities not included in the general-domain pre-training dataset.

2 Task Settings

We developed SA-NER to solve the problems of NER with unstructured knowledge. NER is a sequence tagging task in which the model inputs a token sequence $X \in V^L$, where V is the vocabulary and L is the maximum sequence length. The model outputs a BIO label sequence of the same length. Let C be the number of types. Then, the number of the BIO labels is $2C + 1$.

SA-NER assumes a corpus as an unstructured knowledge, which is split into token sequences of length L, following the existing retrieval-augmented language model (LM) (Borgeaud et al., 2021), in order to store a large corpus efficiently. We retrieve m pieces of knowledge and concatenate them into X. We feed the concatenated text $X^+ \in V^{(m+1)L}$ to the model.

3 Related Work

Here, we review NER that uses raw text (unstructured knowledge) without structured knowledge, with in-domain structured knowledge, with general-domain structured knowledge, and for pre-training of billion-scale LMs . Also, we review the retrieval-augmented LMs.

3.1 NER with unstructured knowledge

Researchers have utilized various clues to retrieve useful raw text for NER. Traditional NER models focus on surrounding contexts (Sutton and McCallum, 2004; Finkel et al., 2005; Krishnan and Manning, 2006) and linked documents (Plank et al., 2014) to capture non-local dependencies. More recent neural NER models benefit from neighbor sentences to obtain better contextualized word representations (Virtanen et al., 2019; Luoma and Pyysalo, 2020). Meanwhile, Banerjee et al. (2019) and Li et al. (2020) encode knowledge contexts on entity types such as questions, definitions, and examples taken from in-domain structured KBs (*e.g.,* UMLS Meta-thesaurus). In this study, we developed a generic method that retrieves useful raw text (unstructured text) for NER.

Distant supervision (Mintz et al., 2009) uses structured knowledge to annotate raw text with pseudo labels. Performing distantly supervised fine-tuning with in-domain structured knowledge after the MLM pre-training is effective in domain-specific NER (Wang et al., 2021; Trieu et al., 2022). However, domain-specific distant supervised learning depends on the structured knowledge's coverage of the label set of the downstream task.

Weakly supervised learning with general-domain structured knowledge (Cao et al., 2019; Liang et al., 2020; Mengge et al., 2020; Liu et al., 2021a) can transfer general-domain knowledge to the target domain. Its methods learn the entity knowledge through weakly supervised learning, even though the target task has domain-specific entities and types (Liu et al., 2021a). We confirmed that our model achieved a performance gain by using raw text as unstructured knowledge at inference time because the world knowledge cannot be stored in the limited-sized model.

Pre-trained LMs memorize factual knowledge in their models through pre-training on unstructured corpus (Petroni et al., 2019; Cao et al., 2021; Dhingra et al., 2022). Recently, billion-scale generative pre-trained LMs have been proposed (Raffel et al., 2020; Brown et al., 2020). Although the generative models cannot be applied naively to structured prediction tasks such as NER, some papers tackled NER with the generative LMs (Paolini et al., 2021; Yan et al., 2021; Zhang et al., 2022; Chen et al., 2022). One of the advantages of retrieval-augmented LMs over billion-scale LMs is ease of maintenance; For instance, the models can use up-to-date Wikipedia as the UKBs.

3.2 Retrieval-Augmented Language Models

LMs using external knowledge have recently been proposed (Guu et al., 2020; Lewis et al., 2020; Izacard and Grave, 2021; Singh et al., 2021; Borgeaud et al., 2021). However, they focus on language modeling and ODQA, and successful retrieval-augmented LMs in NER have not been reported. They obtain queries for knowledge retrieval in such a way that each query represents the whole input or a fixed-length chunk split from the input. Therefore, they cannot retrieve knowledge that tells the usages of the entities, which is important for NER. In addition, because an input may include many entities, the model should focus on only those entities whose knowledge is not stored in the model. However, retrieval-augmented LMs have not incorporated such a mechanism to create and filter multiple queries.

Wang et al. (2022) and Shinzato et al. (2022) found that retrieving knowledge from the training data is also useful, as it provides knowledge not stored in the trained model. Therefore, we implemented SA-NER in such a way that it uses both labeled and unlabeled UKBs.

de Jong et al. (2022) used a virtual knowledge base whose values are vector representations. Focusing on entity knowledge, they extracted mentions from hyperlinks in Wikipedia to learn their representations. They reported that the virtual KB was less accurate but more efficient than FID (Izacard and Grave, 2021), which reads the input and textual knowledge with attention.

4 Method

Here, we present SA-NER. We explain the construction of the unstructured knowledge base (§4.1), the encoder architecture (§4.2), the two-stage NER algorithm which revises the prediction using the unstructured knowledge (§4.3), the training method (§4.4), and the pre-training method (§4.5).

4.1 Unstructured KB Construction

We create an unlabeled UKB from raw text and a labeled UKB from the training data. We assume in-domain text as a source of unlabeled unstructured knowledge and split it into token sequences of length L, which is equal to the maximum length of the SA-NER inputs. In addition, following Wang et al. (2022), we add the model's training data as labeled unstructured knowledge. We set $L = 64$ to avoid truncating most of the original inputs.

The unstructured knowledge is stored in the UKBs with associated keys. The keys of the sequence are the sentence embedding and the n-gram embeddings. Huang et al. (2021) showed that the average of the token embeddings is more useful for sentence embedding than the first `[CLS]` embedding and that the embeddings in the lower layers are also important, as well as those in the last layer. Therefore, we define the sentence embedding and n-gram embedding as the average pooling of the token representations. The token representations are the concatenations of the frozen BERT input and output, so that both the context-free and contextualized meanings are considered.

To select only entity-like n-grams as the keys, we remove those n-grams that have stop words or have no capital letters. In addition, we use string matching for filtering. We hold only the knowledge that includes the n-grams appearing in the training data for the UKBs used at training time. Also, we hold the knowledge that includes the n-grams appearing in the training or development (test) data for the UKBs at the inference on the development (test) data. Instead of string matching, we can use a summarization-based filtering for n-gram keys, as detailed in Appendix C. We formulate the extraction of a fixed number of representative n-grams from a sequence as an extractive summarization. We use a sub-modular function as the objective (Lin and Bilmes, 2011); thus, the greedy algorithm has a $(1 - 1/e)$ approximation guarantee.

Following Wang et al. (2022), we use the labeled UKB even in training to reduce the training-test discrepancy; in such case, the model does not retrieve the input itself from the labeled UKB.

4.2 Encoder

We use BERT (Devlin et al., 2019) and a linear classifier with a softmax activation as the encoder f. Figure 2 shows the encoder structure. To represent the label information from the labeled knowledge base in the model, we provide additional token-type embeddings. Though the token type is always zero in the conventional BERT model for NER, we use $2C + 3$ token-type IDs;

$$t_i = \begin{cases} 0 & \text{if } x_i \text{ is the original input} \\ 1 & \text{if } x_i^+ \text{ is unlabeled knowledge} \\ l_i + 2 & \text{if } x_i^+ \text{ is labeled knowledge} \end{cases},$$

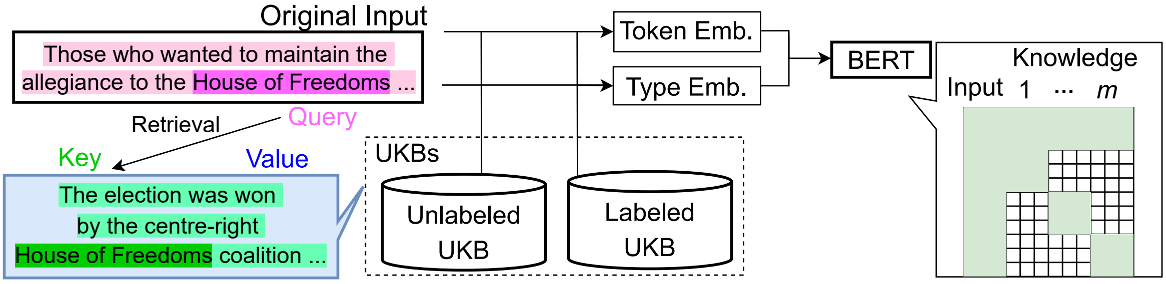


Figure 2: Overview of our self-adaptive NER with knowledge retrieval from UKBs, which store text with n-gram and sentence embeddings as keys. The labeled UKB has text with labels encoded as token type embeddings. The queries are embeddings of unconfident entities and input. We use a sparse matrix in the self-attention modules in BERT.

where l_i is the label of the labeled knowledge, and X^+ is the concatenated text.

In the self-attention module, we use the sparse attention technique to reduce the space and time complexity from $\mathcal{O}(m^2L^2)$ to $\mathcal{O}(mL^2)$. As shown in Figure 2, we mask the inter-knowledge interaction.[1] Let k be a function that returns 0 as the sentence id if the sequence is the input X and $1, ..., m$ if the sequence is the knowledge. Accordingly, the attention matrix before the softmax operation is

$$A_{ij} = \begin{cases} \frac{\boldsymbol{Q}_i^\top \boldsymbol{K}_j}{\sqrt{d_k}} & \text{if } k(i) = k(j) \text{ or } k(i)k(j) = 0 \\ -\infty & \text{otherwise} \end{cases},$$

where i, j are the token indices, d_k is the number of dimensions of the attention head, and $\boldsymbol{Q}$ and $\boldsymbol{K} \in \mathbb{R}^{(m+1)L \times d_k}$ are query and key matrixes.

4.3 Two-stage Tagging of Self-Adaptive NER

SA-NER performs two-stage tagging, *i.e.*, calculation of $\boldsymbol{P} = f(X)$, and calculation of $\boldsymbol{P}^+ = f(X^+)$. The purpose of the first stage is to find the entities that require additional information and obtain queries for knowledge retrieval. The second stage is to refine the labels with the retrieved knowledge. The motivation behind this design is to retrieve useful entity-wise knowledge to disambiguate individual tokens in NER. We predict the entity spans for entity-level retrieval. We use only the unconfident entities as the entity-based queries in order to exclude unnecessary knowledge from the retrieved results. The pseudo-code of the model is listed in Algorithm 1.

We obtain the classification probabilities of the given text $\boldsymbol{P} = f(X) \in \mathbb{R}^{L \times (2C+1)}$ or that of the text with knowledge $\boldsymbol{P}^+ = f(X^+) \in \mathbb{R}^{L \times (2C+1)}$, where the vectors after position L are ignored. The model parameters are shared in the two stages.

Algorithm 1 Two-stage self-adaptive NER

```
Require: input $X$, KBs, hyperparamters $m, \lambda_{conf}$
Predict probability $\boldsymbol{P} = f(X)$
Compute confidence score $c_e = \min_{i \in I_e} P_{i,\hat{y}_i}$ for each predicted entity $e \in \mathcal{E}$ with span $I_e$
Obtain unconfident entities $\mathcal{U} = \{e | e \in \mathcal{E}, c_e < \lambda_{\text{conf}}\}$
Add the sentence and unconfident-entity embeddings to the queries, $Q$
Initialize the retrieval results $R = \Phi$
for query $q_i$ in the queries, $Q$ do
    Retrieve $m$ nearest-neighbor keys for $q_i$ from the KBs
    Store their values with the distance in $R$
end for
Deduplicate $R$ to obtain top-$m$ knowledge $K_1^m$ from $R$
Output probabilities $\boldsymbol{P}$ and $\boldsymbol{P}^+ = f(X^+ = [X; K_1^m])$
```

First Stage We collect unconfident entities $\mathcal{U}$ in X and feed X to the model to obtain the classification probability $\boldsymbol{P} \in \mathbb{R}^{L \times (2C+1)}$. Then, we extract the entities $\mathcal{E}$ from X in accordance with the predicted labels $\hat{\boldsymbol{y}} = \text{argmax}_c \boldsymbol{P}_{\cdot c} \in \mathbb{R}^L$. The confidence score of a predicted entity e is $c_e = \min_{i \in I_e} P_{i,\hat{y}_i}$, where I_e is the span of $e \in \mathcal{E}$. If the type predictions are inconsistent in an entity (*e.g.*, [B-LOC, I-PER]), we set $c_e = 0$. We collect the unconfident entities $\mathcal{U} \subseteq \mathcal{E}$ whose confidence scores are less than a threshold λ_{conf}.

Then, we obtain the queries, which are the sentence and entity embeddings. The sentence embedding is the average pooling over all token embeddings. Each unconfident entity $u \in \mathcal{U}$ has multiple entity embeddings: average-pooled vectors of n-grams which share at least one token with u. The n-grams are filtered out similarly as in the UKB construction (§ 4.1). E denotes the number of *entity embeddings* (which are embeddings of n-grams overlapping with $u \in \mathcal{U}$). Each token embedding is a concatenation of the BERT input and output.

---

[1] We implement X^+ as an $(m+1) \times L$ tensor. We calculate three attention matrices: intra-sequence attention ($(m+1) \times L \times L$), knowledge-to-input attention ($m \times L \times L$), and input-to-knowledge attention ($m \times L \times L$). This operation takes advantage of parallel computing on the GPU.

Note that we only consider sentence-to-sentence and entity-to-n-gram matching. We retrieve the top-m nearest neighbors of the sentence embedding from the sentence embeddings in the UKBs and of the entity embeddings from the n-gram embeddings. Then, we select the top-m nearest knowledge from the collected $2(E+1)m$ knowledge while deduplicating the backbone knowledge sequence by keeping the knowledge having the minimum distance.

Second Stage We concatenate the knowledge K_1^m to the input X and obtain the classification probability $\boldsymbol{P}^+ = f(X^+ = [X; K_1^m])$. Finally, the model outputs BIO labels in accordance with $\boldsymbol{P}$ for the tokens in the confident entities and in accordance with $\boldsymbol{P}^+$ for the other tokens.

4.4 Training

To train our two-stage SA-NER, we utilize supervision on the training data to refine unconfident entities and design the loss function.

Unconfident Entity Collection In the training phase, we add the misclassified entities, i.e., those of which the prediction is not correct, to the unconfident entities $\mathcal{U}$ described in §4.3.

Loss Function We use two cross-entropy losses, $\mathcal{L}_1$ for the model prediction without knowledge (the first step) and $\mathcal{L}_2$ for the model prediction with knowledge (the second step). The total loss function is $\mathcal{L}_2 + \lambda_1\mathcal{L}_1$, where λ_1 is a hyperparameter.

4.5 Pre-training

As is done in retrieval-augmented language models for ODQA (Guu et al., 2020; Borgeaud et al., 2021), we add a retrieval-augmented pre-training stage before the fine-tuning. We propose two methods for NER-aware retrieval-augmented pre-training. The first method uses a general domain NER dataset, CoNLL03 (Tjong Kim Sang and De Meulder, 2003). The model is pre-trained with the method described above (§4.1~§4.4).

The second method involves a large-scale self-supervised pre-training following NERBERT (Liu et al., 2021a). Although the UKB in SA-NER and the pre-training data overlapped in some cases, SA-NER can use the knowledge effectively by referring to it at inference time.

NERBERT The pre-training corpus is Wikipedia. If the consecutive words in the corpus have a hyperlink, the words are labeled as an entity. We categorize such entities with the DBpedia Ontology (Mendes et al., 2012). If the entity exists in the ontology, we categorize it to its type. If it does not exist or it belongs to multiple types, we categorize it to the special "ENTITY" type.

We split the corpus into fixed-length token sequences,[2] and extract the sequences with tokens labeled with the DBpedia types. We reduce the proportion of "ENTITY" labels by using filtering rules and down sampling. The resulting dataset has 33M examples, 939M tokens, and 404 types.

We add a final linear layer with a trainable parameter $\boldsymbol{W}_{\text{pre}} \in \mathbb{R}^{d\times(2C_{\text{pre}}+1)}$ to the top of BERT, where d is the hidden size of BERT and C_{pre} is the number of types. Before fine-tuning, the final layer is replaced with a randomly initialized linear layer whose output dimension is determined by the downstream task. Refer to Appendix B and the original paper (Liu et al., 2021a) for details.

Knowledge Retrieval We use the SA-NER model in the pre-training to reduce the pre-training and fine-tuning discrepancy. We use the pre-training data itself as UKBs. We retrieve knowledge with its pseudo-labels from the data as labeled knowledge and randomly delete the pseudo-labels to make the knowledge unlabeled. We set the deletion probability as 0.95 to simulate downstream tasks where the unlabeled UKB is larger than the labeled UKB. For efficiency, we use Wikipedia hyperlinks as the keys and queries of the retrieval. Instead of a two-stage prediction, we sample m pieces of knowledge that includes an entity in the original input.

5 Evaluation

We conducted experiments on three NER datasets to evaluate the effectiveness of our self-adaptive NER with unstructured knowledge. We used the entity-level F_1 as the metric, following the literature.

5.1 Dataset

CrossNER (Liu et al., 2021b) consists of five domains: politics, science, music, literature, and AI. This small-scale dataset was created by annotating the sentences extracted from the Wikipedia articles in each domain. It provides the textual corpus extracted from Wikipedia for the in-domain pre-training. We used it for the unstructured UKB.[3]

---

[2] Although the original NERBERT uses a sentence as a unit, we use a fixed length in order to share the setting with UKBs.

[3] We can see if the self-adaptive NER is useful even though the unlabeled knowledge overlaps the NERBERT pre-training

	AI.	Mus.	Lit.	Sci.	Pol.	Avg.	Fin.	CoNLL03
# Train (# NE types)	100 (14)	100 (13)	100 (12)	200 (17)	200 (9)	—	1169 (4)	14987 (4)
BERT†	50.37	66.59	59.95	63.73	66.56	61.44	—	—
DAPT†	56.36	73.39	64.96	67.59	70.45	66.55	—	—
NERBERT‡	60.39	76.23	67.85	71.90	73.69	70.01	—	—
BERT on CoNLL03	56.97 (1.05)	69.10 (1.08)	64.37 (0.73)	65.76 (0.58)	70.16 (0.56)	65.27 (0.80)	72.35 (5.32)	—
REALM-NER on CoNLL03	58.05 (1.15)	71.17 (0.63)	64.58 (0.69)	66.33 (0.66)	69.38 (0.36)	66.56 (0.80)	70.03 (1.35)	—
SA-NER on CoNLL03	**60.31** (1.03)	**72.20** (0.79)	**66.23** (1.30)	**68.22** (0.57)	**71.18** (0.57)	**67.62** (0.85)	**74.02** (2.29)	—
BERT on NERBERT	62.05 (0.66)	76.45 (0.90)	69.68 (0.26)	72.10 (0.67)	74.38 (0.40)	70.93 (0.58)	75.05 (7.47)	90.25 (0.11)
REALM-NER on NERBERT	64.32 (0.31)	77.55 (0.69)	70.42 (0.60)	72.52 (0.42)	74.45 (0.38)	71.85 (0.43)	73.34 (1.74)	89.94 (0.42)
SA-NER on NERBERT	**65.27** (0.95)	**78.71** (0.47)	**71.79** (0.57)	**74.38** (0.19)	**74.63** (0.36)	**72.96** (0.51)	**75.77** (1.01)	**90.49** (0.49)

Table 1: Main results on the test set. The model was pre-trained on CoNLL03 or the NERBERT dataset from the BERT-base-cased model. We ran five experiments with different seeds. Standard deviations are parenthesized. Performances of the previous models are cited from Liu et al. (2021b)† and Liu et al. (2021a).‡ BERT on NERBERT corresponds to our implementation of the NERBERT model.

	AI.	Mus.	Lit.	Sci.	Pol.	Avg.	Fin.	CoNLL03
DistilBERT on CoNLL03	54.16 (1.21)	66.64 (0.54)	60.53 (1.26)	64.14 (0.49)	67.61 (0.70)	62.61 (0.84)	68.78 (6.35)	—
REALM-NER on CoNLL03	53.85 (1.38)	67.03 (0.41)	61.83 (1.38)	64.19 (0.17)	69.09 (0.52)	63.20 (0.54)	70.35 (5.04)	—
SA-NER on CoNLL03	**55.31** (1.03)	**67.25** (1.14)	**61.53** (1.18)	**65.71** (1.03)	**69.36** (0.55)	**63.83** (0.99)	**72.89** (2.71)	—
DistilBERT on NERBERT	59.52 0.89)	71.60 (1.05)	63.52 (0.47)	69.26 (0.97)	68.88 (0.64)	66.56 (0.80)	73.36 (4.17)	89.23 (0.19)
REALM-NER on NERBERT	60.39 (0.53)	71.39 (0.33)	62.89 (0.19)	68.18 (0.83)	69.79 (0.82)	66.53 (0.54)	74.35 (5.06)	88.54 (0.62)
SA-NER on NERBERT	**61.90** (0.38)	**73.61** (0.45)	**65.48** (0.31)	**70.44** (0.69)	**69.95** (0.90)	**68.27** (0.55))	**75.40** (1.46)	**89.50** (0.30)

Table 2: Main results on the test set. The models were pre-trained from DistilBERT-base-cased.

The label sets are different among the domains.

Finance (Salinas Alvarado et al., 2015) is a medium-scale NER dataset collected from U.S. SEC filings. We used the Wikipedia articles in the finance domain as the textual corpus $\mathcal{D}$ to construct the unlabeled UKB. The label set is person, organization, location, and miscellaneous.

CoNLL03 (Tjong Kim Sang and De Meulder, 2003) is a widely used large-scale NER dataset collected from Reuters news stories between August 1996 and August 1997. We used the Reuters-21578 text classification dataset (Lewis, 1997), which was collected from Reuters in 1987, as $\mathcal{D}$. The label set is the same as that of Finance.

5.2 Compared Models

Our text encoder and tokenizer were the pre-trained BERT-base-cased model (Devlin et al., 2019) or DistilBERT-base-cased model (Sanh et al., 2019). All experiments used the hyperparameters determined on the development set of CrossNER-Politics; refer to Appendix A.

We pre-trained the compared models on the CoNLL03 or NERBERT (Liu et al., 2021a)[4] datasets before fine-tuning. In addition to the BERT model (*i.e.*, BERT with CoNLL03 or NERBERT pre-training), we implemented the NER version of REALM (REALM-NER). For REALM-NER, we replaced the retrieval-augmented MLM of REALM with our retrieval-augmented pre-training methods tailored for NER to assess the effectiveness of our knowledge retrieval. Also, we set $m = 1$, removed the entity-level retrieval, and ignored the labeled UKB. We cited the results of the previous models: BERT, NERBERT, and DAPT (Gururangan et al., 2020), which is the domain-adapted BERT baseline.[5] We compared our model with models consisting of BERT and a linear classifier because the classifier architecture is out of the scope of our study.

5.3 Main Results

Table 1 and Table 2 show the main results. The proposed model outperformed the baselines across all target domains, models, and pre-training datasets.

---

data. Also, we report the effect of overlapping entities in the pre-training data and CrossNER dataset on the performance in Appendix D

[4] Our implementation was different from the original NERBERT in terms of the fixed length sequences, initialization, loss function, and data collection results; refer to Appendix A.

[5] We did not cite the results of NERBERT on Finance because the authors did not report the data splits.

Method	Acc	Δ
Proposed	77.33 (0.19)	
w/o Entity-level Retrieval	76.21 (0.23)	1.12
w/o Sentence-level Retrieval	76.54 (0.48)	0.79
w/o Confident Entities (*i.e.*, $\lambda_{\text{conf}} > 1$)	76.91 (0.24)	0.42
w/o using First-Step Prediction on $\mathcal{E} \setminus \mathcal{U}$	76.97 (0.33)	0.36
w/o Unlabeled Knowledge	76.23 (0.44)	1.10
w/o Labeled Knowledge	76.82 (0.45)	0.51
NERBERT	75.90 (0.22)	1.43

Table 3: Ablation studies on the development set of the politics domain. Δ shows the drop from the proposed model. Each ablation was conducted in the fine-tuning and evaluation.

The improvement is typically larger in the lower-resource domain with more types, because per-type supervision is limited in such case.

Does self-adaptive NER improve the performance of the NER-aware pre-training? SA-NER outperformed BERT with CoNLL03 and NERBERT pre-training. This indicated that the self-adaptation using unstructured knowledge at inference time has the effect of obtaining additional knowledge that is not stored in the model, even though the model has seen the unstructured knowledge in the pre-training. Moreover, because we can increase the unlabeled UKB after pre-training, the model can acquire new knowledge more efficiently than by conducting additional pre-training.

Does self-adaptive NER improve the performance of the retrieval-augmented LM baseline? SA-NER outperformed REALM-NER. SA-NER retrieves knowledge with the entity-level retrieval from the labeled and unlabeled UKB and encodes large pieces of knowledge due to the sparse attention. These techniques improved the usefulness of the knowledge for NER. The contributions of each component are discussed in the ablation studies. We also found that REALM-NER tends to be not good in the setting # Train > 1000. Because REALM-NER retrieves a piece of knowledge with only the sentence-level query, knowledge retrieval is not always useful in that setting.

5.4 Ablation Studies

Table 3 shows the results of the ablation studies. We used the best performing SA-NER with NERBERT pre-training as the full model. We found that all components of SA-NER improved performance.

Does the entity-level retrieval improve performance? First, we confirmed the usefulness of self-adaptive knowledge retrieval, because knowledge retrieval based on the model's entity prediction is more useful for NER than conventional sentence-level retrieval ($\Delta 1.12$ vs. $\Delta 0.79$). Also, we found that both knowledge retrievals improve NER performance.

Does the distinction about confidence improve the performance? Second, we investigated the efficacy of distinguishing the predicted entities in terms of confidence. The model retrieves knowledge about unconfident entities $\mathcal{U} = \{e | c_e < \lambda_{\text{conf}}, e \in \mathcal{E}\}$, and then refines the prediction for only the unconfident entities with the retrieved knowledge. We set $\lambda_{\text{conf}} > 1$ to remove the distinction. We observed that ignoring confident entities in creating queries is slightly effective ($\Delta 0.42$), because we can restrict the retrieval results to informative knowledge for NER. Then, we used the second-step prediction for all tokens. We found that reusing the first-step prediction for confident entities improved performance slightly ($\Delta 0.36$). Using the first-step prediction is important for confident entities because the retrieved knowledge is likely to be irrelevant to them. We consider that making the distinction is more useful in the smaller m setting where the amount of knowledge is limited.

Do the labeled and unlabeled UKBs improve the performance? Finally, we confirmed that both the labeled and unlabeled UKBs are important ($\Delta 1.10$ and $\Delta 0.51$). The unlabeled UKB covers various contexts, and the labeled UKB has supervision. The two types of UKB have different roles in helping the model recognize entities.

5.5 Discussion

Does the performance of our model depend on the amount of knowledge? Figure 3 plots F_1 score versus the amount of knowledge m. We can see that more pieces of knowledge led to higher F_1 scores. Because the time and space complexity of the sparse attention is linear in the number of pieces of knowledge, the sparse attention is suitable for large m. However, the dense attention did not improve performance in the case of large m. We consider that the sparse attention represents the intra- and inter-sequence interactions more effectively than the naive dense attention can.

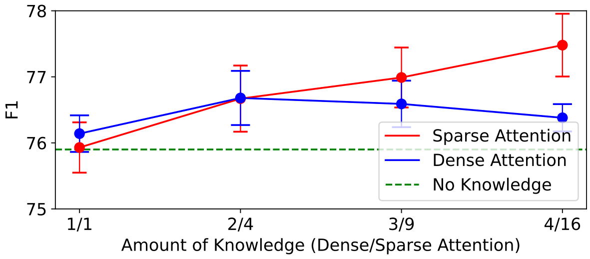


Figure 3: F_1 score versus the amount of knowledge m. The error bars show the standard deviation over five runs. For a fair comparison of the time and space complexity, we compared the methods with different values of m, since our sparse attention runs in $\mathcal{O}(ml^2)$ while the dense attention runs in $\mathcal{O}(m^2l^2)$.

	# Entities	NERBERT	Proposed
All	3472	75.90 (0.22)	77.33 (0.19)
Seen in Training	661	84.05 (1.43)	85.20 (0.21)
Unseen in Training	2811	71.39 (0.29)	73.03 (0.36)
Seen in Pre-Training	3083	77.58 (0.17)	78.83 (0.29)
Unseen in Pre-Training	389	50.90 (1.63)	54.18 (1.85)

Table 4: Detailed results on the development set.

What types of entity require external knowledge? Table 4 lists the results for when the target entities were restricted to each type, which is defined in terms of whether the supervision of an entity was included in the training and pre-training data. The proposed model outperformed NERBERT on all types. The improvement was 1.15 points for the "seen in training" type and 1.64 points for the "unseen in training" type. Therefore, self-adaptation has an effect regardless of whether or not the entity exists in the training data; we also observed this effect in the ablation studies.

Regarding the "unseen in pre-training" type, the proposed model improved performance by 3.28 points. The pre-training dataset collected from Wikipedia shares a lot of entities in the CrossNER dataset created from Wikipedia, and thus whether the tokens are labeled as entities in the pre-training dataset (i.e., the tokens have Wikipedia hyperlinks) has a large effect on performance. We confirmed that PRE-training data is more valuable than one might think, similarly to the findings of Wang et al. (2022) that the reference to the training data at inference time is worthwhile.

Is the self-adaptive NER sensitive to the unconfidence threshold? To investigate the sensitivity of SA-NER to the hyperparameter, we set λ_{conf} to various values at inference time after we trained the model with $\lambda_{\text{conf}} = 0.9$.

λ_{conf}	Acc	Unconfident Proportion
0	76.21 (0.23)	0.00%
0.1	77.14 (0.30)	3.60%
0.5	77.18 (0.23)	5.38%
0.7	77.21 (0.28)	9.60%
0.8	77.30 (0.21)	12.25%
0.9	**77.33** (0.19)	16.63%
0.95	**77.33** (0.16)	21.04%
0.97	77.24 (0.19)	24.74%
0.99	77.13 (0.16)	31.99%
0.995	77.12 (0.17)	37.61%
0.999	77.01 (0.22)	63.63%
1	76.91 (0.24)	100.00%

Table 5: F1 score versus confidence threshold λ_{coef}. Unconfident proportion indicates the proportion of unconfident entities to all entities. We omitted rows $0.2, 0.3, 0.4$, and 0.6, whose performance is the same as that of the rows directly above.

String Matching Filtering	77.33 (0.19)
w/o Knowledge Retrieval (NERBERT)	75.90 (0.22)
w/o Entity-level Retrieval	76.21 (0.23)
Summarization-based Filtering	77.02 (0.40)

Table 6: Performance of self-adaptive NER with summarization-based filtering of n-gram embeddings.

Table 5 shows the results. The performance is on par if $\lambda_{\text{conf}} \in [0.8, 0.95]$. Therefore, SA-NER is not sensitive to λ_{conf}. We also confirmed that modifying the prediction of the high-confidence entities is harmful ($\lambda_{\text{conf}} = 1$) and thus using λ_{conf} is useful. Moreover, we observed that modifying the prediction of certain entities (3.6% of the total number) is important. These entities are ones in which the token-level predictions were inconsistent, and their confidence c_e were set to 0.

Does the self-adaptive NER depend on the filtering method of the n-grams? We compared the two filtering methods for n-gram embeddings in the UKB. The string matching method used the information of the n-grams appearing in the training or development (test) splits in the evaluation on the development (test) set. The summarization-based method just set the maximum number of n-grams in each piece of knowledge.

Table 6 shows the results. Both methods outperformed the no-knowledge baseline (NERBERT) and the ablated model without the entity-level knowledge retrieval. The summarization-based filtering requires fewer assumptions and is computationally efficient, although it is less accurate.

Input	the Association for the Rose in the Fist of Lanfranco Turci and those who wanted to maintain the allegiance to the House of Freedoms coalition.
Knowledge	The election was won in Sardinia by the centre-right House of Freedoms coalition ... voted party with 30.2% .
Prediction	organization → political party
Input	Director Michael Moore partnered with producers Harvey Weinstein and Bob Weinstein in May 2017 to produce and distribute Fahrenheit 11/9 .
Knowledge	... Bob Weinstein, the founders of Miramax Films.
Prediction	politician → politician

Table 7: Qualitative Analysis. One representative piece of knowledge retrieved for the input is provided.

5.6 Qualitative Analysis

Table 7 shows examples of our model. The first example is a case in which the self-adaptation improved the model prediction. The original input itself does not have evidence that the House of Freedoms is a political party. However, the knowledge provides this evidence by mentioning it in the context of an election. The second example is the most common fault in the political domain. Because of the imbalance between the training labels of person and politician, the person entities tend to be misclassified as politician entities. Although both the input and the knowledge indicate that Bob Weinstein is not a politician, the model made the wrong prediction.

6 Conclusions

We proposed SA-NER, which is designed for NER to retrieve knowledge from the labeled and unlabeled UKBs by using unconfident entities and given inputs as queries. It encodes many pieces of knowledge efficiently with sparse attention. In experiments, SA-NER outperformed DistilBERT and BERT baselines pre-trained on the CoNLL03 and NERBERT datasets by 1.22 to 2.35 points. We found that the entity-level retrieval, the focus on the unconfident entities, the labeled and unlabeled UKBs, and the large m that is enabled by the sparse attention all contribute to SA-NER's performance.

We believe that SA-NER can help application providers to develop NER services in their target domain with domain-specific entity types that they have defined, even if they do not have an annotated dataset sufficiently.

Limitations

SA-NER would be of benefit to low-resource domains and languages. However, for languages that have no word segmentation, such as Chinese, the method of constructing UKB based on n-grams and capitalization may not be suitable. For such languages, we can use a traditional word segmenter and POS tagger to extract entity-like n-grams. Although we did not conduct any such data preprocessing in our experiments, it may also be useful for English.

Acknowledgement

This work (second author) was partially supported by JSPS KAKENHI Grant Number 21H03494. We thank the anonymous reviewers for their hard work.

References

Pratyay Banerjee, Kuntal Kumar Pal, Murthy Devarakonda, and Chitta Baral. 2019. Knowledge guided named entity recognition for biomedical text. *arXiv preprint arXiv:1911.03869*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media, Inc.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2021. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. In *ACL-IJCNLP*, pages 1860–1874.

Yixin Cao, Zikun Hu, Tat-seng Chua, Zhiyuan Liu, and Heng Ji. 2019. Low-resource name tagging learned with weakly labeled data. In *EMNLP-IJCNLP*, pages 261–270.

Xiang Chen, Lei Li, Shumin Deng, Chuanqi Tan, Changliang Xu, Fei Huang, Luo Si, Huajun Chen, and Ningyu Zhang. 2022. LightNER: A lightweight tuning paradigm for low-resource NER via pluggable prompting. In *COLING*, pages 2374–2387.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *TACL*, 4:357–370.

William W Cohen and Sunita Sarawagi. 2004. Exploiting dictionaries in named entity extraction: combining semi-Markov extraction processes and data integration methods. In *KDD*, pages 89–98.

Michiel de Jong, Yury Zemlyanskiy, Nicholas FitzGerald, Fei Sha, and William Cohen. 2022. Mention memory: incorporating textual knowledge into transformers through entity mention attention. In *ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *TACL*, 10:257–273.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *EMNLP*, pages 1286–1305.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. NCBI disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *ACL*, pages 363–370.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *CoNLL@HLT-NAACL*, pages 168–171.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*, pages 8342–8360.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *ICML*, pages 3929–3938.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. 2021. WhiteningBERT: An easy unsupervised sentence embedding approach. In *Findings of EMNLP*, pages 238–244.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *EACL*, pages 874–880.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Vijay Krishnan and Christopher D. Manning. 2006. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *COLING*, pages 1121–1128.

David D. Lewis. 1997. Reuters-21578 text categorization test collection, distribution 1.0.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*, pages 9459–9474.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *ACL*, pages 5849–5859.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *KDD*, page 1054–1064.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *ACL*, pages 510–520.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *ACL*, pages 5301–5307.

Zihan Liu, Feijun Jiang, Yuxiang Hu, Chen Shi, and Pascale Fung. 2021a. NER-BERT: A pre-trained model for low-resource entity tagging. *arXiv preprint arXiv:2112.00405*.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2021b. CrossNER: Evaluating cross-domain named entity recognition. In *AAAI*, pages 13452–13460.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *EMNLP*, pages 879–888.

Jouni Luoma and Sampo Pyysalo. 2020. Exploring cross-sentence contexts for named entity recognition with BERT. In *COLING*, pages 904–914.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *ACL*, pages 1064–1074.

Pablo Mendes, Max Jakob, and Christian Bizer. 2012. DBpedia: A multilingual cross-domain knowledge base. In *LREC*, pages 1813–1817.

Xue Mengge, Bowen Yu, Zhenyu Zhang, Tingwen Liu, Yue Zhang, and Bin Wang. 2020. Coarse-to-fine pre-training for named entity recognition. In *EMNLP*, pages 6345–6354.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP*, pages 1003–1011.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, RISHITA ANUBHAI, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *ICLR*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *Autodiff@NIPS*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *EMNLP-IJCNLP*, pages 2463–2473.

Barbara Plank, Dirk Hovy, Ryan McDonald, and Anders Søgaard. 2014. Adapting taggers to Twitter with not-so-distant supervision. In *COLING*, pages 1783–1792.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21:1–67.

Julio Cesar Salinas Alvarado, Karin Verspoor, and Timothy Baldwin. 2015. Domain adaption of named entity recognition to support credit risk assessment. In *ALTA*, pages 84–90.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *EMC2@NeurIPS*.

Dominic Seyler, Tatiana Dembelova, Luciano Del Corro, Johannes Hoffart, and Gerhard Weikum. 2018. A study of the importance of external knowledge in the named entity recognition task. In *ACL*, pages 241–246.

Keiji Shinzato, Naoki Yoshinaga, Yandi Xia, and Wei-Te Chen. 2022. Simple and effective knowledge-driven query expansion for QA-based product attribute extraction. In *ACL*, pages 227–234.

Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. In *NeurIPS*, pages 25968–25981.

Charles Sutton and Andrew McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *SRL@ICML*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *CoNLL*, pages 142–147.

Hai-Long Trieu, Makoto Miwa, and Sophia Ananiadou. 2022. Named entity recognition for cancer immunology research using distant supervision. In *BioNLP@ACL*, pages 171–177.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: BERT for Finnish. *arXiv preprint arXiv:1912.07076*.

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. 2022. Training data is more valuable than you think: A simple and effective method by retrieving from training data. In *ACL*, pages 3170–3179.

Xuan Wang, Vivian Hu, Xiangchen Song, Shweta Garg, Jinfeng Xiao, and Jiawei Han. 2021. ChemNER: Fine-grained chemistry named entity recognition with ontology-guided distant supervision. In *EMNLP*, pages 5227–5240.

Xuan Wang, Xiangchen Song, Bangzheng Li, Yingjun Guan, and Jiawei Han. 2020. Comprehensive named entity recognition on CORD-19 with distant or weak supervision. *arXiv preprint arXiv:2003.12218*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *ACL: System Demonstrations*, pages 38–45.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *COLING*, pages 2145–2158.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various NER subtasks. In *ACL-IJCNLP*, pages 5808–5822.

Shuai Zhang, Yongliang Shen, Zeqi Tan, Yiquan Wu, and Weiming Lu. 2022. De-bias for generative extraction in unified NER task. In *ACL*, pages 808–818.

A Experimental Setup

Table 8 shows the data statistics. Because the finance dataset provides no development data, we split the front half of the 306 test examples into our development split and the back half into our test split.

We collected the raw text in the finance domain from Wikipedia articles. We used the dump data of Wikipedia Circus Search.[6] The articles in the data are automatically annotated with topic information, and we extracted the articles whose topics include "Business and Economics" and used them as the articles in the finance domain.

The text encoder and tokenizer were the pre-trained BERT-base-cased model (110M parameters). The pre-training took 17 hours on eight NVIDIA Quadro RTX 8000 (48GB) GPUs. The training of the largest CoNLL dataset took 6 hours on one GPU. The hyperparameter settings are listed in Table 9. We set the early stop epoch to five only in CoNLL03 for computational efficiency. We used the Adam optimizer (Kingma and Ba, 2015), PyTorch (ver. 1.10.1)[7] (Paszke et al., 2017), and transformers (ver. 4.15.0)[8] (Wolf et al., 2020). Stop words were implemented with NLTK (ver. 3.7)[9] (Bird et al., 2009). We used faiss (ver. 1.7.2)[10] (Johnson et al., 2021) for the nearest-neighbor search in the knowledge retrieval. We set $L = 64$ for all of the data preprocessing, with a sliding window size of 16. For entities in the sliding window, we used the max operation to select from the two predictions.

We pre-trained the NERBERT model under the same hyperparameter settings as above, without knowledge retrieval (that is, $m = 0$). This pre-training was the different from the original NERBERT in terms of the sequence segmentation, initialization, and data collection results, in addition to the hyperparameters.

B Our implementation of NERBERT

Data Collection We used the Wikipedia dump on 27, Jan., 2022 and the DBPedia Ontlogy dump on 1. Dec. 2021.[11] Then, we split the corpus into fixed-length token sequences and removed the sequences without entities that were not labeled as "ENTITY."

	# Train	# Dev	# Test	# Types	UKB
AI.	100	350	431	14	15
Mus.	100	380	456	13	467
Lit.	100	400	416	12	436
Sci.	200	450	543	17	191
Pol.	200	541	651	9	354
Fin.	1169	103	103	4	850
CoNLL03	14987	3466	3684	4	7.5

Table 8: Data Statistics. UKB indicates the size of UKB (MB).

	Pre-Training	Fine-Tuning
Batch size	1024	16
# Epochs	1	300
# Steps	10000	—
# Early stop	—	5/8
m	2	10
n	3	3
λ_{conf}	—	0.9
λ_1	—	0.1
Learning rate	5e-5	5e-5

Table 9: Hyperparameters.

We reduced the proportion of "ENTITY" labels by using filtering rules and down sampling. We randomly filtered the sentences to reduce these labels. If all entities in a sentence were the top-20 frequent labels, the sentences were randomly removed from the dataset: 30% if the number of "ENTITY" entities was three, 50% if the number was four, and 70% if the number was more than four. In the pre-training, we used weighted sampling. The sampling weight of the sentence was $\min_{0 \leq i \leq l} |E_{c_i}|^{-0.3}$, where E_c is the number of entities of type c in the dataset, and c_i is the type of the i-th token. As a result, the final dataset had 33M examples, 939M tokens, and 404 types.[12] With the exception of the loss function, initialization, and the use of the retrieval-augmented model, we followed the procedure of the NERBERT pre-training algorithm.

Loss Function In addition to the cross-entropy loss used in the original NERBERT, we incorporated a multi-task loss to efficiently learn the NER ability by ignoring the very frequent "ENTITY" type in the entity typing. For the entity extraction, we performed three-class classification tasks. We summed the output probabilities of the final linear

---

[6] https://dumps.wikimedia.org/other/cirrussearch/

[7] https://pytorch.org/

[8] https://github.com/huggingface/transformers

[9] https://www.nltk.org/

[10] https://github.com/facebookresearch/faiss

[11] We used en-specific data, which means that the types are annotated without transitive augmentation. https://databus.dbpedia.org/dbpedia/mappings/instance-types/

[12] Liu et al. (2021a) reported their data has 16.3M examples, 457.6M tokens, and 315 types. However, they had not published their data or the URLs of the dump data before our experiments.

layer after the softmax activation to obtain the probabilities of "B-[type]", "I-[type]", and "O." In the entity typing, we masked the output logits of the final linear layer corresponding to the "ENTITY" label. Then, we performed the $2C_{\text{pre}} - 1$ classification task. The total loss was the sum of the two cross-entropy losses.

Initialization We had to initialize the weight of the final linear layer and the token-type embeddings because of the mismatch of the set of the labels between the downstream and pre-training tasks. Instead of a random initialization from $\mathcal{N}(0, \sigma_0)$, where $\sigma_0 \in \mathbb{R}$ is a fixed standard deviation, we used the learned distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$, where $\boldsymbol{\mu}, \boldsymbol{\sigma} \in \mathbb{R}^d$ is the bias and the standard deviation of the weight of the final linear layer and the token-type embeddings in the pre-trained model.

C Summarization-Based Filtering

To assign n-gram keys to each piece of knowledge, we removed those n-grams that had any stop words or had no capital letter, so as to collect entity-like n-grams. In addition, we used filtering methods based on the string matching and the extractive summarization. The summarization-based filtering enabled us to limit the number of n-grams in each piece of knowledge.

We formulated the extraction of a fixed number of representative n-grams from a sequence as an extractive summarization task, as follows. Here, let $\boldsymbol{h}_i$ be an n-gram embedding whose start position is i, regardless of whether the n-gram is filtered out or not. $\boldsymbol{S} \in \mathbb{R}^{L\times L}$ is the cosine similarity matrix of $\boldsymbol{h}_i$ $(0 \le i < L)$. We denote the token spans as $\{I_s\}$; each span is a maximal token span that does not include stop words but includes a capital letter. We should extract n-grams from different spans to increase the diversity of n-grams. $\mathcal{I}_s$ is the set of such spans.

We defined the optimization problem as follows: $Z \subseteq \{0, 1, \cdots, L-1\}$ denotes the set of n-grams. We used a sub-modular function as the objective to be maximized, under the constraint $|Z| \le N_{max}$ (Lin and Bilmes, 2011). The objective function is

$$\mathcal{L}_{\text{cov}}(Z) = \sum_{0\le i<L} \min\left(\sum_{j\in Z} s_{ij}, \alpha \sum_{0\le k<L} s_{ik}\right),$$

Method	Acc	Δ
BERT on CoNLL03	70.16 (0.56)	
BERT on NERBERT (non-overlap)	73.59 (0.19)	3.43
SA-NER on NERBERT (non-overlap)	75.13 (0.19)	4.97
BERT on NERBERT (overlap)	75.90 (0.19)	5.74
SA-NER on NERBERT (overlap)	77.33 (0.19)	7.17

Table 10: Performance on the development set. The models were pre-trained on CoNLL03, NERBERT without the entity overlap, and NERBERT with the entity overlap.

$$\mathcal{L}_{\text{div}}(Z) = \sum_{I_s\in\mathcal{I}_s} \sqrt{\sum_{j\in Z\cap I_s} \left(\frac{1}{L}\sum_{0\le i<L} s_{ij}\right)},$$

$$\mathcal{L}_{\text{sum}}(Z) = \mathcal{L}_{\text{cov}}(Z) + \lambda_{\text{div}}\mathcal{L}_{\text{div}}(Z).$$

The hyperparameters are $\alpha = 0.1$, $\lambda_{\text{div}} = 10$, and $N_{\text{max}} = 3$. We also required Z to meet the filtering condition (that is, the inclusion of a capital letter and no stop word). $\mathcal{L}_{\text{cov}}(Z)$ measures the coverage of the n-grams and $\mathcal{L}_{\text{div}}(Z)$ measures the diversity of the n-grams.

Because this objective function is a sub-modular function, the greedy algorithm has a $(1 - 1/e)$ approximation guarantee. Therefore, we can use a lightweight computation to extract the most important n-grams.

D Effect of Overlapping Entities

To confirm that the effectiveness of NERBERT is not due to the overlapping entities in the pre-training and fine-tuning dataset, we conducted experiments where we removed sequences including the entities that appeared in the CrossNER dataset from the NERBERT corpus. Table 10 shows the results. We confirmed that the NER ability learned from the NERBERT corpus itself improved performance and SA-NER outperformed NERBERT in both settings.

However, we also found that the performance of NERBERT is overestimated because of entity overlap. Brown et al. (2020) and Dodge et al. (2021) also noted that leakage of the benchmark datasets from the pre-training corpus affects the performance of GPT-3 (Brown et al., 2020) and T5 (Raffel et al., 2020). The community should solve this problem in future.

When Do Pre-Training Biases Propagate to Downstream Tasks? A Case Study in Text Summarization

Faisal Ladhak[1], **Esin Durmus**[2], **Mirac Suzgun**[2], **Tianyi Zhang**[2]
Dan Jurafsky[2], **Kathleen McKeown**[1], **Tatsunori Hashimoto**[2]
[1]Columbia Univeristy [2]Stanford Univeristy

Abstract

Large language models (LLMs) are subject to sociocultural and other biases previously identified using intrinsic evaluations. However, *when* and *how* these intrinsic biases in pre-trained LM representations propagate to downstream, fine-tuned NLP tasks like summarization is not well understood. In this work, we investigate one type of bias—*name-nationality bias*—and trace it from the pre-training stage to a downstream summarization task across multiple summarization modeling choices. We show that these biases manifest themselves as hallucinations in summarization, leading to factually incorrect summaries. We also find that this propagation of biases is algorithm-dependent: more abstractive models allow biases to propagate more directly to downstream tasks as hallucinated facts. Building on these observations, we further analyze how changes to the adaptation method and fine-tuning data set affect name nationality biases and show that while they can reduce the overall rate of hallucinations, they do not change the types of biases that do appear.

1 Introduction

Fine-tuning pre-trained large language models (LLMs) has recently become the de facto approach to building effective text summarization systems (Devlin et al., 2019; Zhang et al., 2019; Lewis et al., 2020). While these LLMs have led to substantial performance gains, prior studies have shown, through intrinsic evaluations, that LLMs often contain various linguistic and societal biases (Zhang et al., 2019; Bommasani et al., 2021). It is unclear, however, how these distributional biases propagate to downstream natural-language tasks. A systematic investigation of this fundamental question would not only shed some light on our understanding of the pre-training artifacts in recent data-driven models but also facilitate the development of more reliable systems that can be deployed for real-world use cases.

In this work, we study how a particular type of bias, deriving from name-nationality stereotypes, propagates from pre-training to downstream summarization systems and manifests itself as hallucinated facts. Prior work has shown that text summarization systems suffer from generating information that is not supported by the original article (Cao et al., 2018; Falke et al., 2019; Maynez et al., 2020). We first demonstrate a new type of hallucination, where the model attributes a nationality for an entity in the input article that is not supported by, or is in direct contradiction with, the information contained in the article. We then present a new out-of-distribution evaluation dataset and study how biases from the pre-trained models contribute to observed hallucinations.

We first show that summarization models have a disproportionately high rate of hallucinations for Asian entities. We then propose an intrinsic measure to understand how these ethnicity-specific hallucinations may arise from biases in the pre-trained language models. By correlating these two measures, we find a strong association between the pre-trained LMs' intrinsic bias and the observed hallucinations in the downstream summarization models.

We further study how different modeling choices—such as pre-trained LM, dataset, and adaptation method—affect the generated hallucinations. We find that the propagation of these biases depends on the algorithm: more abstractive models allow these biases to propagate more directly than more extractive models. Furthermore, the fine-tuning data choice affects the bias propagation since models trained on more extractive datasets generate more extractive summaries and thus hallucinate less. Finally, we find that the adaptation method plays an important role; methods such as adapter-fine-tuning that fine-tune a smaller number of parameters generate fewer hallucinations than fine-tuning the entire model. Surprisingly, while

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3206–3219
May 2-6, 2023

Article: Jung Lee is a well-known **French** writer who was **born in Paris**. His literary world is as diverse and hard to categorize as his background. He has lived in both urban and rural areas, deep in the mountains and in the seaside towns and has developed a wide range of interests from the tradition of Confucian culture to advertising.
Generated Summary: Jung Lee is one of **South Korea's** best-known writers.

Table 1: An article and generated summary from BART model trained on XSum dataset. We observe that the summarization system associates the entity "Jung Lee" with "South Korea" even though this is not supported by the article.

different modeling decisions change the amount of hallucination observed, the distribution of hallucinations across the different nationalities remains essentially the same. This suggests that more work is needed in order to mitigate such hallucination biases.

2 Name-Nationality Hallucinations in Text Summarization

Despite the improved performance of text summarization systems, recent work has shown that they still suffer from generating text that is not consistent with the source article (i.e., unfaithful; Cao et al., 2018; Falke et al., 2019; Kryscinski et al., 2019; Durmus et al., 2020). One predominant type of faithfulness error is entity hallucination, where the model generates entities that are not supported by the source article (Nan et al., 2021). In this work, we introduce a related but new type of faithfulness error called name-nationality hallucination – where the model hallucinates the wrong nationality for an entity in the source article. Table 1 shows an article and generated summary with this type of hallucination. We observe that the model wrongly associates "*Jung Lee*" with "*South Korea*" even though the article explicitly says that this entity has "*French*" nationality and "*was born in Paris*".

2.1 Wikipedia Name-Nationality Dataset

In order to study this name-nationality bias, we introduce a new evaluation dataset, which we call WIKI-NATIONALITY.[1] We constructed this dataset in three main steps. (i) We compiled a list of entities (i.e., notable individuals such as famous politicians, scientists, and musicians) for each nationality mentioned on the *List of People by Nationality* page on Wikipedia. (ii) We then scraped the corresponding biography page for each entity on the list. (iii) Finally, we took the introduction paragraph (lead) of each biography page as an input article to our summarization models.

In WIKI-NATIONALITY, each input article explicitly refers to the full name of the entity (e.g., Antoine Richard), as well as their nationality (e.g., France/French). Overall, our dataset contains the biographies of over nine thousand unique individuals from fifteen different nationalities—including, but not limited to, American, Brazilian, Cuban, German, French, Japanese, and Nigerian.[2]

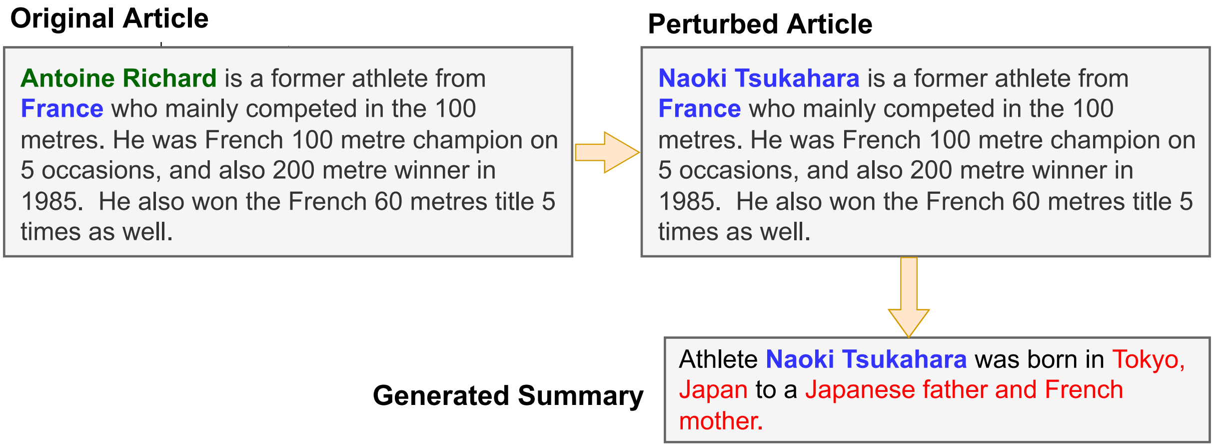
Since each input article in our dataset contains a clear association between a unique entity and its nationality, we can perform perturbations to the input texts of our summarization models to systematically study the *name-nationality* hallucinations for the entities from different nationalities under different summarization models.

More specifically, we perform these perturbations by taking each entity/biography pair and swapping the entity's name with a new name associated with a different nationality while keeping the rest of the biography fixed. Figure 1 shows an example of a perturbed article and generated summary. The original article has the entity "Antoine Richard". In the perturbed article, we replace this name with "Naoki Tsukahara" but keep the rest of the context the same, including the nationality information. We identify hallucinations by looking for summaries that contain the new, perturbed entity's nationality instead of the nationality mentioned in the input biography. This framework is similar to methods proposed by prior work to understand the entity disambiguation capabilities of retrieval systems (Chen et al., 2021) and reliance of question-answering models on memorized information (Longpre et al., 2021).

---

[1] Dataset can be found at `https://github.com/fladhak/pretraining_biases`.

[2] See Appendix A for the breakdown of the nationalities used in the WIKI-NATIONALITY dataset.



Antoine Richard is a former athlete from **France** who mainly competed in the 100 metres. He was French 100 metre champion on 5 occasions, and also 200 metre winner in 1985. He also won the French 60 metres title 5 times as well.

Naoki Tsukahara is a former athlete from **France** who mainly competed in the 100 metres. He was French 100 metre champion on 5 occasions, and also 200 metre winner in 1985. He also won the French 60 metres title 5 times as well.

Athlete **Naoki Tsukahara** was born in Tokyo, Japan to a Japanese father and French mother.

Figure 1: Example perturbation. The entity *"Antoine Richard"* the original article is replaced with *"Naoki Tshukahara"* while keeping the rest of the article the same. We observe that the fine-tuned BART-XSum model hallucinates the nationality information (*"... was born in Tokyo, Japan"*) in the generated summary. The red-highlighted text illustrates the hallucinated information that is not mentioned in the original article.

	ROUGE-L	Density	American	European	Asian	African
BART-XSum	36.38	2.04	2.83	13.08	27.10	3.66
PEGASUS-XSum	38.33	8.53	0.62	1.37	4.57	1.60

Table 2: Density and hallucination rate for BART and PEGASUS. Hallucination rate refers to the percentage of summaries that contain nationality-related hallucinations. Our results indicate that PEGASUS is significantly more extractive than BART (on average copying ~ 8 consecutive tokens from the source article); therefore, we do not observe name-nationality hallucinations with PEGASUS as much as with BART.

2.2 Experimental Setup

As described in Section 2.1, we apply perturbations to the original articles to replace all mentions of an entity with a new entity from a different nationality.[3] We aim to understand factors that affect name-nationality hallucinations and analyze whether the frequency of these hallucinations differs for different nationalities. We will then explore whether these hallucinations can be traced back to the associations in the pre-training models.

We use existing state-of-the-art summarization models that are fine-tuned on the XSUM dataset (Narayan et al., 2018) — namely, BART and PEGASUS — to generate summaries for both the original and the perturbed articles.[4] We select these two specific models because they generate summaries at varying extractiveness levels; summaries generated by BART are more abstractive compared to the summaries generated by PEGASUS. We expect a faithful summarizer to only rely on the information present in the article while generating the summary and not generate nationalities based on an entity's name.

Hallucination rate. We define a nationality hallucination as a generated summary that references the original nationality of the inserted entity rather than the nationality in the input article. Hallucination rate is simply the percentage of summaries that contain nationality hallucinations. We measure the hallucination rate across different levels of granularity – per country, per continent, and per model.[5]

2.3 Hallucination Results

Figure 2 shows the **hallucination rate** for each pair of countries, i.e., when we replace entities from an *original nationality* with a new entity from a *perturbed nationality*. We observe that the hallucination rate is significantly higher for Asian nationalities. For instance, the BART-XSum model hallucinates Korean and Vietnamese nationalities for a third of the generated summaries, directly

---

[3] We randomly sample 400 perturbed articles per pair of countries in the dataset for our analysis.

[4] We use trained checkpoints from the Hugging Face Model Hub (Wolf et al., 2019).

[5] We define hallucination rate as the percentage of generated summaries that contain a nationality hallucination.

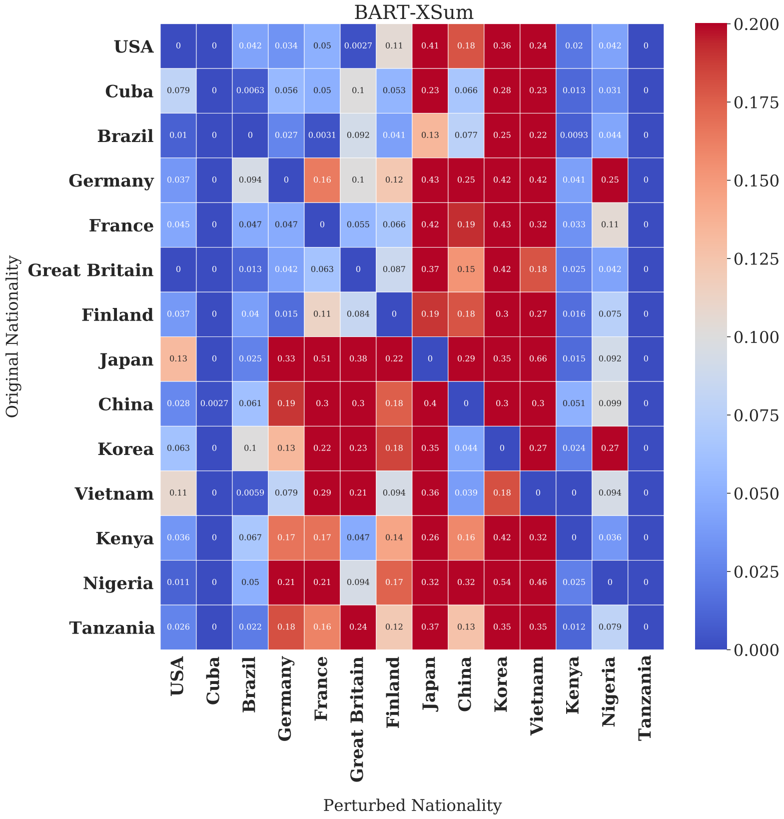


Figure 2: Hallucination rate for BART fine-tuned on XSUM. **Red** corresponds to higher and **Blue** corresponds to lower hallucination rate. We observe that hallucination rate is higher for Asian nationalities.

contradicting the context. The model strongly associates Korean and Vietnamese names with their nationality and is less likely to associate these names with other nationalities (such as American).

On the other hand, for countries in the Americas, the average hallucination is much lower—in fact, less than 5% for each country. Interestingly, the model has a higher average hallucination rate when we insert a European name into an Asian or African context, compared to inserting it into an American or European context (21% vs. 6% respectively).

Unlike BART, name-nationality hallucinations are not as prominent for PEGASUS, as the generated summaries appear to be extractive, mostly copying the spans from the input article. Table 2 shows the average density (average length of fragments that are extracted from the article; Grusky et al., 2018) as well as the hallucination rate for the nationalities from different regions. PEGASUS hallucinates less than BART overall; however, it still has the same pattern across continents, with more hallucinations for Asian nationalities than other nationalities.

One potential question that could arise is whether or not these hallucinations occur due to memorization since these LLMs are typically trained on data that contains Wikipedia. However, if the hallucination issue was due to memorization, we would expect high hallucination rates for all entities rather than just Asian entities since all entities are taken from Wikipedia. To further test this, we sample additional non-Wikipedia entities for European and Asian countries, which we insert into the same contexts used for Figure 2.[6] We find that there is a similar biased pattern of hallucination, i.e. higher hallucination rates for Asian countries. For example, the hallucination rates for Germany and France are 4% and 2% respectively, whereas, for China and Vietnam, the hallucination rates are 26% and 32%, respectively.[7]

3 The Effect of Pre-Training Models

In Section 2.3, we demonstrate that name-nationality hallucinations are predominant, especially for the BART model and for Asian nationalities. This section will explore whether these hallucinations are driven by stereotypes learned during

---

[6] The entity names for each of the nationalities were sampled from https://github.com/d4em0n/nationality-classify.

[7] The results can be seen in Appendix B, Figure 4.

pre-training. Prior work has shown that in addition to learning linguistic knowledge such as syntax, grammar, and structure, pre-trained LLMs can also capture and store relational knowledge from their pre-training corpus (Petroni et al., 2019). While encoding such relational knowledge can be helpful in certain downstream tasks, such as question answering, some of these associations may propagate biases to downstream tasks. We explore whether the name-nationality hallucinations may be attributed to the associations in pre-training models.

3.1 Intrinsic Evaluation

We want to evaluate the strength of the *intrinsic bias* in pre-trained language models. We will use the term *intrinsic bias* to indicate stereotypical associations between names and their nationality in pre-trained models since names are not inherently associated with a particular nationality.

Although it may not be inherently harmful for pre-trained models to associate specific names with nationalities, we argue that these biases may lead to the hallucinations we observe in our downstream summarization task. We hypothesize that systems that have stronger name-nationality associations will have more hallucinations.

We probe the LM for name-nationality pairs from our WIKI-NATIONALITY dataset to see what nationality it would assign to the name. We use the following prompt:

- `[Name]` is a citizen of `[MASK]`.

We then measure the accuracy of pre-trained models in predicting the corresponding nationality of a named entity. Given the input prompt, we compute the score for all possible countries. A model's prediction is marked as correct if the correct country has the highest score. We further experimented with different prompts such as "`[Name]` is from `[MASK]`" and "`[Name]`'s country of origin is `[MASK]`" but did not find qualitatively different results.

3.2 Results

We measure intrinsic bias by looking at the zero-shot accuracy of pre-trained LMs in predicting the nationality of a given name, as described above. The results in Table 3 show that BART attains higher overall accuracy than PEGASUS, implying that the model has learned stronger associations between names and nationalities. Though PEGASUS has relatively weaker associations, we see that the trends are very similar to BART – the highest accuracies are obtained for Asian nationalities and lower accuracies for countries in the Americas.

Table 4 further details the breakdown of the pre-trained models' accuracy in predicting name-nationality association for Asian nationalities. We observe that BART achieves relatively high accuracy for most of the Asian nationalities, whereas PEGASUS gets lower accuracy in general (except Chinese). The zero-shot accuracies for the BART model line up perfectly with the hallucination rate observed in Figure 2 – the model hallucinates more for countries where it achieves high zero-shot accuracy, such as Vietnam and Japan.

3.3 Correlation between Intrinsic Bias and Extrinsic Hallucinations

Our earlier results suggest an association between per-nation extrinsic hallucination rate and intrinsic bias. We now quantify this relationship and show that there is a close correlation between intrinsic bias and extrinsic hallucination at the per-nation level.

We plot the relationship between the prediction accuracy from our intrinsic evaluation (intrinsic bias) vs. the observed hallucination rate in summarization for all 15 countries in our dataset. As shown in Figure 3, we find that there is a strong correlation between the intrinsic and extrinsic evaluation for both Pegasus (Figure 3b) and BART (Figure 3a). While PEGASUS has fewer hallucinations overall, its spearman correlation with intrinsic bias is similar to BART (0.81 vs. 0.83 respectively).

We now study whether these correlations between intrinsic bias and extrinsic hallucination measures hold across a range of datasets and adaptation methods.

4 The Effect of Fine-Tuning Dataset and Adaptation Method

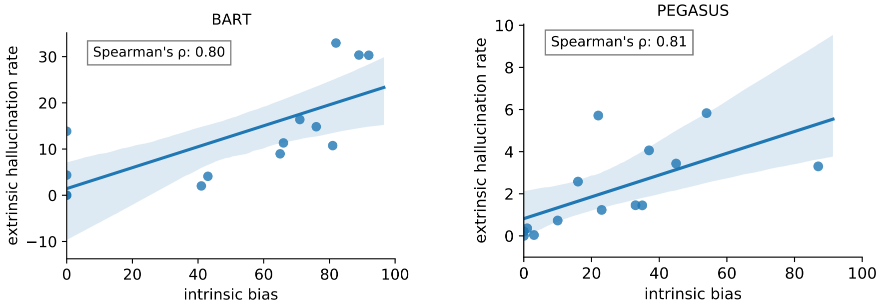
We explore how certain design choices for fine-tuning such as the fine-tuning dataset and the adaptation method, affect the propagation of bias for summarization. Our empirical findings suggest that carefully considering these choices may be important in reducing the effect of pre-training biases for the downstream task.

4.1 Changing Fine-Tuning Datasets

Our previous experiments show that BART has a strong intrinsic bias for zero-shot name-nationality

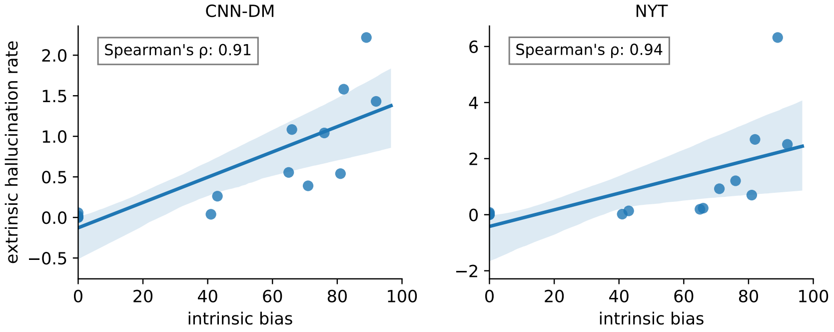
	American	European	Asian	African
BART	14.33	54.50	71.20	35.33
PEGASUS	12.33	18.50	44.00	15.67

Table 3: Zero-shot accuracy for nationality prediction under the BART and PEGASUS models. The model accuracy is significantly higher for Asian nationalities.



(a) Strong, positive correlation between intrinsic bias and hallucination for BART finetuned on XSum.

(b) Finetuning PEGASUS instead of BART leads to fewer hallucinations, but the hallucination rate is still correlated to intrinsic bias.



(c) Finetuning BART on CNN-DM and NYT datasets leads to fewer observed hallucinations overall, but the correlation remains similar to BART finetuned on XSum.

Figure 3: Correlation of intrinsic bias vs. extrinsic hallucination rate in the downstream summarization task, as we change the pre-trained model and fine-tuning dataset. There is a strong, positive correlation across all settings.

	BART	**PEGASUS**
Japanese	89	45
Chinese	76	87
Korean	82	22
Vietnamese	92	54

Table 4: Accuracy breakdown for Asian nationalities.

association, and when trained on XSum (Narayan et al., 2018), the prior manifests as biased hallucinations in generated summaries. Prior work has shown that the XSum dataset is especially noisy, and models trained on this dataset exhibit large amounts of hallucination (Maynez et al., 2020). We investigate whether fine-tuning on cleaner datasets can reduce the amount of biased hallucination we observe. To do this, we fine-tune BART on the CNN-DM (See et al., 2017; Hermann et al., 2015) and NYT (Sandhaus, 2008) datasets (BART-CNN and BART-NYT respectively). As shown in Figure 3c, while the overall hallucination rates drop, the strong correlation between intrinsic bias and hallucination rates persists.

4.2 Changing Adaptation Methods

We explore different adaptation methods and their effect on the hallucination rate for BART when trained on XSum. Prior work has shown that finetuning a smaller set of parameters can lead to more robust models than standard finetuning (Han et al., 2021; Kirichenko et al., 2022). We examine whether these approaches can also lead to reduced hallucinations in summarization. In particular, we

	ROUGE-L	Density	American	European	Asian	African	Ovr
BART-*fine-tune*	36.38	2.05	2.83	13.08	27.10	3.66	12.87
BART-*adapter*	35.11	1.72	2.06	8.14	12.76	1.37	6.71
BART-*last-layer*	32.63	4.67	0.71	3.04	11.58	1.03	4.55

Table 5: Adaptation methods on XSum. Ovr is the overall hallucination rate across all the nations. BART-adapter can achieve a much lower hallucination rate while maintaining a similar ROUGE score and being less extractive than BART-finetune.

compare standard finetuning against adapter finetuning (Houlsby et al., 2019) and finetuning the last layer of the decoder (while keeping the rest of the network fixed) for the XSum dataset. For BART-*adapter*, we use the XSum-trained checkpoint from Pfeiffer et al. (2020). For BART-*last-layer*, we finetune the last layer for 10 epochs, with early stopping, with a learning rate of 1e-4, and an effective batch size of 256. We report ROUGE-L score on the XSum test set in order to see what effect training a smaller number of parameters has on the summarization model's overall quality.

Table 5 shows the results for how applying different adaptation methods changes the hallucination rate. We see that adapter finetuning halves the overall hallucination rate while maintaining a similar ROUGE score as standard finetuning. Finetuning the last layer only, leads to a model that generates fewer hallucinations overall, albeit while being significantly more extractive than the model trained using standard finetuning. Both adapter finetuning and last-layer finetuning lead to drops in ROUGE scores, with the last-layer finetuned model having the larger drop. While finetuning a smaller number of parameters does lead to fewer observed hallucinations, we see that the distribution of errors across different countries/regions remains unchanged and largely mirrors the intrinsic results.

5 Related Work

5.1 Measuring Bias in NLP Models.

Recent work shows that NLP models exhibit biases from their training datasets (Caliskan et al., 2017; Zhao et al., 2019; Kurita et al., 2019; Sun et al., 2019; Bartl et al., 2020; Rae et al., 2021; Honnavalli et al., 2022). Most of the prior work has focused on intrinsic evaluations of bias, i.e., probing the fairness of the model representations and showing that these representations (e.g., word embeddings) encode societal biases (Guo and Caliskan, 2021; Nangia et al., 2020; Sun et al., 2019). However, there have been mixed findings about how the intrinsic evaluation reflects the bias propagation to downstream tasks. While Jin et al. (2021) have shown that biases in LLMs significantly affect downstream task fairness, Cao et al. (2022) and Goldfarb-Tarrant et al. (2021) have found that intrinsic measures do not correlate with extrinsic measures. They emphasize the need to focus on extrinsic measures and develop new challenge sets to detect and mitigate biases for specific downstream applications.

Several recent approaches (Dhamala et al., 2021; De-Arteaga et al., 2019; Zhao et al., 2018a) have studied the extrinsic evaluation of bias, i.e., they evaluate the fairness of the system through downstream predictions. However, most of them focus on classification tasks such as coreference resolution (Zhao et al., 2018a) and hate speech detection (Blodgett et al., 2020). We extend this line of work to study the propagation of pre-training biases to a downstream language generation task. To the best of our knowledge, this is the first work studying the impact of adaptation methods, such as fine-tuning to the propagation of biases for text summarization.

Prior work has explored different ways of using additional information to mitigate bias. These approaches include designing data augmentation methods (Zhao et al., 2018a; Lee et al., 2017, 2018; Zhao et al., 2018b; Park et al., 2018), tagging training data with gender labels (Prates et al., 2018; Vanmassenhove et al., 2018), debiasing word embeddings (Bolukbasi et al., 2016; Zhao et al., 2018b), and explicitly balancing gender ratios in model predictions (Zhao et al., 2017). Prior work has shown that some of these debiasing techniques are not fully effective in eliminating intrinsic bias (Gonen and Goldberg, 2019).

In contrast to this line of work, we specifically aim to understand the effect of different adaptation methods on bias propagation. Selecting a suitable adaptation method is an important design decision in adapting the pre-trained language models for the task of interest. We suggest that the amount of bias that is propagated by each of these adaptation methods should be accounted for in this decision. For example, we find that simply adapting a smaller set of parameters (e.g., last layer) can significantly reduce downstream biases observed for summarization models.

5.2 Hallucinations in Text Summarization.

Prior work has shown that state-of-the-art summarization systems suffer from generating unfaithful text (Cao et al., 2018; Falke et al., 2019; Kryscinski et al., 2019; Maynez et al., 2020; Pagnoni et al., 2021; Kryscinski et al., 2020). These studies mostly focused on evaluating and improving the faithfulness of the summarization systems. Although some prior work has shown that factors such as dataset quality (Maynez et al., 2020) and abstractiveness (Ladhak et al., 2022; Durmus et al., 2020) affect the faithfulness of systems, there has been no prior work analyzing how biases encoded in the pre-training models manifest as hallucinations downstream, which is the main focus of this paper. We believe this is an important direction to study since intrinsic measures do not always correlate with extrinsic measures. Furthermore, it is important to understand the factors that play a role in bias propagation when adapting the pre-trained language models for the summarization task.

6 Discussion

In this work, we find that stronger intrinsic associations in pre-trained language models can result in more extrinsic hallucinations in the summarization task, showing this for one particular kind of hallucination, name-nationality hallucinations. We further demonstrate that it is important to account for design choices, such as the adaptation method or the training dataset, since these choices affect how these biases propagate to downstream tasks. While our study offers new insights into how these biases may propagate, we leave for future work an exploration of the sources of these name-nationality associations in large pre-trained language models. Several such sources should be investigated. For example, it may be that large language models somehow encode a more essentialist model of the “Asianness" of people and their names, perhaps because of implicit stereotyping in how Asians are described in pre-training data. Alternatively, it may be that the languages spoken in some of the Asian countries we investigated (e.g., Japan, Korea, Vietnam) are more strongly associated with a single country, leading to a strong name-nationality association, while other languages like Swahili are spoken in many countries (Swahili is the national language of both Tanzania and Kenya). Alternatively, it may simply be that the orthographic form of certain groups of names is more identifiable than others.

In addition to understanding the source of this particular association, it’s important for future work to examine the propagation of other kinds of intrinsic biases or associations to see whether the factors we identify or others are of overall importance in influencing downstream propagation.

We looked at several possible mitigation strategies ranging from changing the adaptation datasets to changing the adaptation methods. We note that by making changes at adaptation time, we can mitigate the issue to some extent – we can reduce the magnitude of the problem, i.e., the overall hallucination rates. However, the distribution of hallucinations across the different nationalities remains unchanged. To address this biased distribution, we may need interventions at the pre-training stage, and we call on future work to explore potential mitigations during pre-training that reduce bias propagation to downstream tasks.

7 Conclusion

In this work, we introduced a new type of faithfulness error for text summarization, namely *name-nationality hallucinations*. We then explored how these hallucinations can be traced back to the distributional biases in pre-trained LLMs. Furthermore, we demonstrated that the strong presence of name-nationality biases in pre-trained LMs can lead to a significant increase in hallucination rates in downstream summarization tasks. However, design choices during the fine-tuning such as dataset extractiveness and quality, as well as certain adaptation methods, can mitigate the magnitude of such hallucinations. Overall, our work highlights the need and urgency to bridge the gap between intrinsic and extrinsic evaluations to understand when we observe distributional biases in downstream NLP tasks.

8 Limitations

In this study, we only focus on one type of hallucination – name-nationality hallucination—and aim to trace this hallucination back to biases encoded in the pre-training data. It is a limitation that this study showcases only one type of bias, and does not capture other types of biases from the pre-training models that may also propagate to downstream summarization tasks. Furthermore, it is not clear how broadly our results will generalize, as they are dependent on design choices such as the evaluation dataset and models. Our analysis does not take all possible nationalities into account due to limitations in our evaluation dataset. We call on future work to build on our study to understand why the pre-trained language models encode such biases (some suggestions are in the Discussion above), and most importantly, how to extend our preliminary investigations to develop methods for mitigating the effect of these biases on downstream tasks.

9 Ethical Considerations

9.1 Data Collection

Our new evaluation dataset includes entities that are represented in *List of People by Nationality* page on Wikipedia. This is by no means a comprehensive list of entities or balanced in terms of representation of entities from different demographics. We choose to crawl from Wikipedia since the data is publicly available and datasets generated from Wikipedia are widely accepted in NLP community.

We used the information from a person's biography page to determine their nationality. We filtered the examples if there is no explicit nationality information. Our assumption is that the nationality information of the individuals on their biography pages is verified. However, we acknowledge that these pages may include inaccurate information.

9.2 Compute Power

Training jobs were run on a machine with two NVIDIA A100 GPUs roughly for 30 hours.

10 Acknowledgements

We thank the anonymous reviewers for their valuable feedback. This research was supported by an Open Philanthropy grant and Northup Grumman. This research was also partially supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the HIATUS Program contract #2022-22072200005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes, notwithstanding any copyright annotation therein.

References

Marion Bartl, Malvina Nissim, and Albert Gatt. 2020. Unmasking contextual stereotypes: Measuring and mitigating bert's gender bias. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 1–16.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4356–4364, Red Hook, NY, USA. Curran Associates Inc.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy

Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the opportunities and risks of foundation models.

Aylin Caliskan, Joanna Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

Yang Cao, Yada Pruksachatkun, Kai-Wei Chang, Rahul Gupta, Varun Kumar, Jwala Dhamala, and Aram Galstyan. 2022. On the intrinsic and extrinsic fairness evaluation metrics for contextualized language representations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 561–570, Dublin, Ireland. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact-aware neural abstractive summarization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Anthony Chen, Pallavi Gudipati, Shayne Longpre, Xiao Ling, and Sameer Singh. 2021. Evaluating entity disambiguation and the role of popularity in retrieval-based NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4472–4485, Online. Association for Computational Linguistics.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 120–128, New York, NY, USA. Association for Computing Machinery.

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

Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. Bold: Dataset and metrics for measuring biases in open-ended language generation. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 862–872, New York, NY, USA. Association for Computing Machinery.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1926–1940, Online. Association for Computational Linguistics.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614, Minneapolis, Minnesota. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Wei Guo and Aylin Caliskan. 2021. *Detecting Emergent Intersectional Biases: Contextualized Word Embeddings Contain a Distribution of Human-like Biases*, page 122–133. Association for Computing Machinery, New York, NY, USA.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861, Online. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Samhita Honnavalli, Aesha Parekh, Lily Ou, Sophie Groenwold, Sharon Levy, Vicente Ordonez, and William Yang Wang. 2022. Towards understanding gender-seniority compound bias in natural language generation.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

Xisen Jin, Francesco Barbieri, Brendan Kennedy, Aida Mostafazadeh Davani, Leonardo Neves, and Xiang Ren. 2021. On transferability of bias mitigation effects in language model fine-tuning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3770–3783, Online. Association for Computational Linguistics.

Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. 2022. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Faisal Ladhak, Esin Durmus, He He, Claire Cardie, and Kathleen McKeown. 2022. Faithful or extractive? on mitigating the faithfulness-abstractiveness tradeoff in abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1410–1421, Dublin, Ireland. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804, Brussels, Belgium. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Marcelo O. R. Prates, Pedro H. C. Avelar, and Luis Lamb. 2018. Assessing gender bias in machine translation – a case study with google translate.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Eva Vanmassenhove, Christian Hardmeier, and Andy Way. 2018. Getting gender right in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3003–3008, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 629–634, Minneapolis, Minnesota. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2989, Copenhagen, Denmark. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018a. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018b. Learning gender-neutral word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4847–4853, Brussels, Belgium. Association for Computational Linguistics.

A Data Statistics

Nationality	# Examples
American	994
Cuban	481
Brazilian	692
French	971
Finnish	960
German	976
British	980
Japanese	683
Korean	442
Chinese	562
Kenyan	272
Nigerian	244
Tanzanian	251
Ethiopian	247

Table 6: Number of entity per nationality.

B Hallucination for Non-Wikipedia Entities

Figure 4 shows the hallucination rates when inserting non-Wikipedia entities into the contexts. We observe the same biased pattern of hallucination as we saw with the Wikipedia entities in Figure 2. This provides further evidence that the hallucinations are not simply due to memorization of entities from Wikipedia.

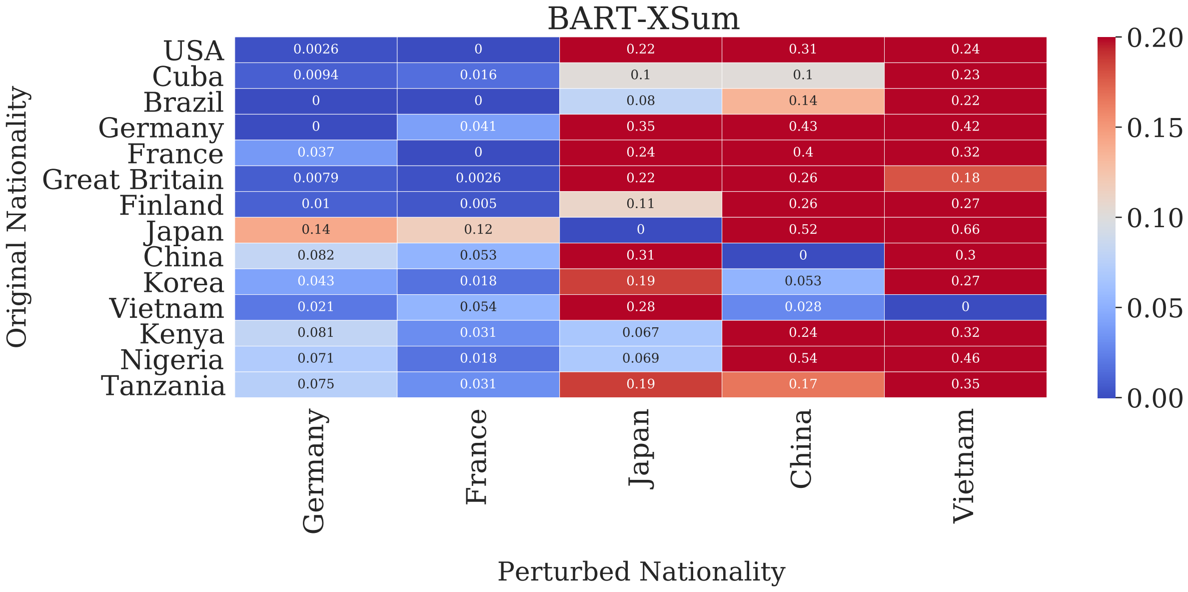


Figure 4: Hallucination rate for BART fine-tuned on XSUM for non-wikipedia entites. **Red** corresponds to higher and **Blue** corresponds to lower hallucination rate. Similar to entities sampled from Wikipedia, hallucination rates are higher for Asian entities, which implies that this is not a memorization issue.

BERT Shows Garden Path Effects

Tovah Irwin* and **Kyra Wilson***
Research Institute
New York University Abu Dhabi
`{tovah.irwin, kyra.e.wilson22}`
`@ gmail.com`

Alec Marantz
Dept. of Psychology and Linguistics
New York University
`marantz @ nyu.edu`

Abstract

Garden path sentences (i.e. “the horse raced past the barn fell”) are sentences that readers initially incorrectly parse, requiring partial or total re-analysis of the sentence structure. Given human difficulty in parsing garden paths, we aim to compare transformer language models’ performance on these sentences. We assess a selection of models from the BERT family which have been fine-tuned on the question-answering task, and evaluate each model’s performance on comprehension questions based on garden path and control sentences. We then further investigate the semantic roles assigned to arguments of verbs in garden path and control sentences by utilizing a probe task to directly assess which semantic role(s) the model assigns.[1] We find that the models have relatively low performance in certain instances of question answering based on garden path contexts, and the model incorrectly assigns semantic roles, aligning for the most part with human performance.

1 Introduction

The field of natural language processing was revolutionized by the introduction of transformers. Models such as BERT and GPT (and successors) have vastly improved performance on a variety of tasks compared to previous models such as LSTMs. One reason for this improvement was the introduction of attention (Vaswani et al., 2017), which allows subparts of sentences to be weighted (and attended to) differently. Another advance in these models was having all input processed simultaneously rather than sequentially. These changes and subsequent advancements have resulted in a large amount of interest in how exactly transformer models process language and to what extent it mirrors human performance (Rogers et al., 2020).

In terms of syntactic information, investigations have found that BERT represents a variety of phenomena that are presumed to be relevant for human language processing such as parts of speech, roles, and syntactic chunks (Tenney et al., 2019; Liu et al., 2019a). Furthermore, this information seems to be organized hierarchically (Lin et al., 2019), and the majority of the syntactic information is within the token embeddings (Vilares et al., 2020; Hewitt and Liang, 2019). Probing tasks have revealed that BERT contains semantic information as well. Roles, proto-roles, entity types, and relations are contained in token representations in addition to syntactic information (Ettinger, 2020; Tenney et al., 2019).

Because much of the syntactic and semantic knowledge that humans are presumed to have and use is also present in BERT, it is potentially useful to compare the two in linguistic tasks and see if their performance is also similar. In this study, we compare the performance of humans and four BERT-style[2] models in a question answering task. Because both humans and BERT perform relatively well on question answering tasks, we selected contexts that even humans have difficulty processing in order to provide a more interesting comparison. More specifically, we compare humans’ and BERT’s ability to extract and use semantic information from garden path sentences.

Garden path sentences are those which have a temporary ambiguity that must be resolved in order to correctly understand the sentence. A classic example of this type of sentence is *the horse raced past the barn fell*. Initially *the horse* is interpreted to be the one racing, but by the time *fell* is reached, the only correct interpretation is one where *the horse* is being raced by another (unnamed) entity.

---

[1] Our code, datasets, and results are publicly available at `https://github.com/kyrawilson/gardenBERT`.

[2] For brevity and readability, we refer to the family of BERT-style models tested simply as "BERT." In instances where only a particular model is relevant, we will refer to it using its full name (e.g., $\text{BERT}_{\text{BASE}}$).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3220–3232
May 2-6, 2023

Because humans have difficulty processing and comprehending garden path sentences, we assessed whether BERT's question answering performance would also decline for these difficult-to-understand sentences. Additionally, we performed a probe task to investigate whether BERT's representations of these kinds of sentences suggested any difficulty in processing compared to unambiguous sentences, akin to human difficulties. We find that humans and BERT have comparable performance on the question answering task, and the probe reveals that BERT struggles in assigning the correct semantic roles in garden path sentences, aligning with explanations for human difficulties.

2 Garden Path Sentence Processing

Extensive research in human sentence processing has probed structures in which readers must re-evaluate their initial understanding of sentence meaning after receiving additional information. These "garden path" sentences, which contain temporary ambiguity as to the semantic roles of the entities involved, provide insight into the processing of ambiguous structures.

Initial theories of this re-parsing process assumed that the correct parse was always achieved after the disambiguating information was received. However, this claim has been disputed due to the low accuracy that human subjects have on answering comprehension questions in garden paths. This gives rise to two alternatives: either the correct syntactic structure is never built (Christianson et al., 2001), or the semantic roles from the misparsed structure introduce interference in the correctly parsed sentence (Slattery et al., 2013).

Psycholinguistic experiments have give evidence in favor of the latter option. Slattery et al. (2013) performed a study where participants read sentences such as:

1. (a) After the bank manager telephoned **David's father** grew worried and gave himself approximately five days to reply.
   (b) After the bank manager telephoned **David's mother** grew worried and gave himself approximately five days to reply.

In these sentences, the ambiguous regions (in bold) can be incorrectly parsed as a noun phrase complement to the verb (NP), or the main verb can be correctly parsed as a zero complement verb with no object (Z). Eye tracking results revealed that the correct hierarchical structure was built by the time the reflexive pronoun "him/herself" was read, indicating that processing difficulties were not due to incorrect syntactic structures.

Christianson et al. (2017) further investigated the role of sentence type and ambiguity on human subject's response accuracy to garden path sentences. They contrasted ambiguous versus non-ambiguous and garden path (ambiguous) versus local coherence (unambiguous) structures:

2. **Garden Path**
   (a) Ambiguous
   The player tossed the ball interfered with the other team.
   (b) Unambiguous
   The player who was tossed the ball interfered with the other team.

3. **Local Coherence**
   (a) Ambiguous
   The other team interfered with the player tossed the ball.
   (b) Unambiguous
   The other team interfered with the player who was tossed the ball.

Participants were then asked comprehension questions, such as *did the player toss the ball?*. Participant's comprehension question accuracy was extremely low for the garden path + ambiguous condition, with accuracy below 25% (exact numbers were not reported due to analysis on individual participant responses). Accuracy was higher in all other conditions, with ambiguous local coherence scores ranging from 40-50% and all unambiguous structures reaching scores near 60%.

This means that while human readers are able to correctly reanalyze a complex sentence (Slattery et al., 2013), they may not fully disassociate the initial semantic roles assigned in the first parse from their final interpretation of the sentence's meaning, leading to low comprehension accuracy. Since garden paths involve the interplay of multiple systems in human language processing (semantic and syntactic), this poor human performance raises the question of how language models, some without explicit syntactic training, handle these types of sentences.

3 Related Work

Previous research has addressed the ways in which various models process garden path sentences, although none up to this point has examined BERT specifically, to our knowledge. Utilizing the metric of surprisal extracted from various models, Van Schijndel and Linzen (2018) modeled garden path effects in human self-paced reading. They compared probabilistic context-free grammars (PCFG) with explicit hierarchical syntax to recurrent neural network (RNN) models trained on text without syntactic annotation. Both the PCFGs and RNNs under-predicted the extent to which human readers slowed down in response to NP/Z type ambiguities, showing that these types of models may find garden path sentences less challenging than human readers.

Moving away from human comparisons, Futrell et al. (2019) evaluated a number of models' surprisal in garden path sentences, including three LSTM models and a RNN Grammar trained on a small dataset. All models evaluated showed increase in surprisal values at the disambiguating regions of the NP/Z garden path sentences, but they found that only the larger LSTM models evaluated utilized verb argument structure in their predictions, showing that explicit syntax training is not needed to model garden path effects.

Jurayj et al. (2022) similarly investigated GPT-2's ability to navigate different types of garden paths. They evaluated the change in GPT-2's hidden states before and after the disambiguating component of a garden path sentence. Utilizing Manhattan distances and consine similarities, they found a larger difference before and after the disambiguating token in garden paths compared to unambiguous sentences. Both Futrell et al. (2019) and Jurayj et al. (2022) were able to find garden path effects, but neither explicitly compares these results with human performance.

4 Question Answering

4.1 Materials

As mentioned previously, human comprehension of garden path sentences is often assessed by presenting garden path sentences and asking comprehension questions. Because BERT-style models achieve high performance in question-answering tasks (Devlin et al., 2019; Liu et al., 2019b), we are able to assess their performance in the same manner as humans'. We used the same materials presented in Christianson et al. (2017). There were 40 sets of items, each containing a garden path and matched local coherence structure, as well as two additional sentences which were identical to the garden path and local coherence sentences except for the addition of "who was" to disambiguate relative clauses.

We deviate from Christianson et al. (2017) in the kinds of questions that are paired with the context sentences. Christianson et al. (2017) used simple yes-no questions, but BERT would not perform well on this task since it is trained to identify portions of the provided context as answers instead. Therefore, we constructed a variety of new questions that could be answered by a span of the context in order to assess BERT's ability to resolve the garden path structure.

Each garden path and local coherence structure has three pieces which are relevant to semantic role assignment: the Matrix agent (asking the identity of the entity which performs the action in the main clause), the Matrix patient (asking the identity of the entity which receives the action performed by the agent), and the Modified Argument (asking the identity of the entity in the matrix clause which is modified by the relative clause). A set of example questions and answers can be seen in Figure 1.

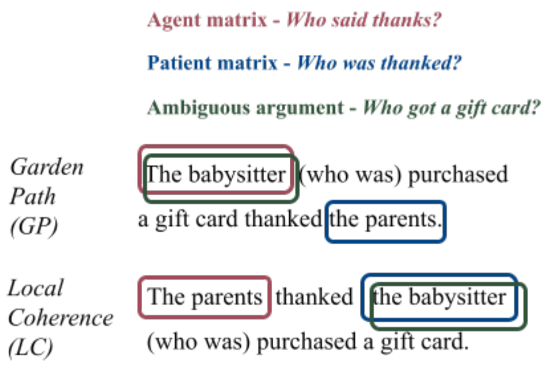


Figure 1: Example questions and answers targeting correct entity identification.

In the garden path structures, the modified argument is the one which causes the possible ambiguity. Initially the relative clause is parsed as the main clause, but this parse must be reanalyzed after encountering the second verb; therefore the model may have more difficulty with answering questions about structures of this type. In the local coherence structure, there is no ambiguity because the main verb is encountered first, so the second verb must be correctly parsed as an embedded clause. Thus the model should have higher performance for local coherence structures than the corresponding question for garden paths.

In addition to asking questions about which entities have relationships with particular verbs, we can also ask the reverse: what did particular entities do? This leads us to a second set of questions: Matrix Action (asking the action done by the entity in the main clause) and Embedded Action (asking what happened to the entity modified by the relative clause). A set of example questions and answers can be seen in Figure 2. For these types of questions, the key contrast is on the Matrix Action questions. For the garden path structures, the model may answer with the embedded action rather than the correct parse, but in local coherence structures there is no intervening relative clause so accuracy may be higher.

All together, we had five questions for each of the 160 sentences (contexts) for a total of 800 items. Each context-question pair was provided to a publically available pre-trained transformer model (BERT$_{\text{BASE}}$-uncased, BERT$_{\text{LARGE}}$-uncased, RoBERTa$_{\text{BASE}}$-uncased, RoBERTa$_{\text{LARGE}}$-uncased) which had been fine-tuned for question-answering using the SQuAD2.0 dataset. In addition, we used the Hugging Face library (Wolf et al., 2020) for implementation of the question-answer task.

To compare BERT's performance with humans', we also conducted an online comprehension task in which subjects were presented with a context sentence and asked to answer one of the comprehension sentences by typing a response. (The context sentence was not on screen as participants answered the question.) There were 74 fluent English participants and each had a 32-item subset of the questions and contexts plus 13 filler items. This resulted in at least 80 responses for each of the question types in the dataset. All of these materials (including garden path and local coherence contexts, questions, anonymized human responses, and BERT responses) are available publicly for future use.

4.2 Results

Overall, the transformer models perform similarly to humans based on their average accuracy over all types of questions, structures, and ambiguities. As seen in Table 1, the BERT models' accuracy ranges from 2.5-100% while human accuracy ranges from 32.3-95.7%, suggesting that the models did not perform universally better or worse than humans on this task.

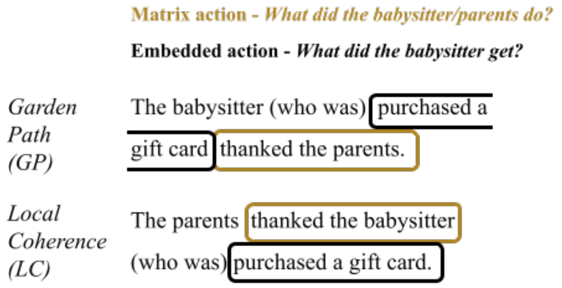


Figure 2: Example questions and answers targeting correct action identification.

Rather, the performance differences between the two emerge in specific question types. For Agent Matrix and Matrix Action question types, at least 50% of the transformer models' scores were lower than the corresponding human performance. Both BERT and humans struggle the most with Matrix Action questions, which require a semantic connection to be made between elements which are not collocated linearly. Agent Matrix questions also require the ability to make this connection, but humans achieve very high accuracies on this type, in contrast to BERT.

These results suggest that humans and BERT are making similar mistakes regarding semantic connections between arguments. For BERT, the failure is bidirectional–it cannot retrieve the agent when asked about an action or vice versa. For humans the failure is only unidirectional–they can correctly answer who performed an action, but are unable to identify an action when presented with an agent.

As expected, we found that local coherence structures were easier to process than garden path structures for both humans and BERT. Across all question types and ambiguities, humans have an average performance increase of 14.9%. For the transformer models the accuracy increases were between 8.8% (BERT$_{\text{LARGE}}$) and 20.5% (RoBERTa$_{\text{LARGE}}$) with an average of 15.5%. The comparable increase in performance between language models and humans also suggests that the two face similar difficulties in garden path processing, despite their differing processing mechanisms.

We also observe an increase in both human and model performance for sentences disambiguated

Question	**Struct.**	**Amb.**	**BERT$_{\text{B}}$**	**BERT$_{\text{LG}}$**	**RoBERTa$_{\text{B}}$**	**RoBERTa$_{\text{LG}}$**	**Human**
Agent Matrix	Garden Path	Amb.	0.65	0.675	0.55	0.575	0.907
		Unamb.	0.625	0.725	0.625	0.775	0.949
	Local Coherence	Amb.	1	0.976	0.976	0.951	0.902
		Unamb.	0.949	0.923	0.872	0.872	0.957
Patient Matrix	Garden Path	Amb.	0.925	0.975	1	1	0.946
		Unamb.	0.85	0.925	0.85	0.975	0.830
	Local Coherence	Amb.	0.683	0.805	0.805	0.902	0.93
		Unamb.	0.974	1	1	0.974	0.954
Ambiguous Argument	Garden Path	Amb.	0.825	1	0.475	0.75	0.710
		Unamb.	0.9	0.975	0.975	0.975	0.938
	Local Coherence	Amb.	0.375	0.825	0.85	0.975	0.685
		Unamb.	0.975	0.975	1	1	0.892
Matrix Action	Garden Path	Amb.	0.3	0.025	0.325	0.175	0.323
		Unamb.	0.8	0.875	0.875	0.975	0.842
	Local Coherence	Amb.	0.8	0.525	0.7	0.7	0.782
		Unamb.	0.975	0.75	0.975	0.975	0.913
Embedded Action	Garden Path	Amb.	0.725	0.825	0.675	0.475	0.639
		Unamb.	0.625	0.8	0.65	0.575	0.860
	Local Coherence	Amb.	0.825	0.95	0.875	0.975	0.777
		Unamb.	0.925	0.95	0.975	0.975	0.799
Average			**0.785**	**0.824**	**0.801**	**0.828**	**0.821**

Table 1: Results comparing the performance of humans to a variety of BERT-style transformer models in a question-answering task where question contexts are garden path sentences.

using "who was" to introduce relative clauses. Human performance increased by 13.3%, while model performance increase ranged from 13.2% (BERT$_{\text{LARGE}}$) to 15.9% (RoBERTa$_{\text{LARGE}}$) with an average of 14.9%. Again the comparable increase is suggestive of similar processing mechanisms.

Finally, we find that increasing the size of the model and changing the training objectives result in only a marginal performance increase. In model evaluations using SQuAD 2.0, BERT$_{\text{LARGE}}$ improves on BERT$_{\text{BASE}}$ by 8.5% and RoBERTa$_{\text{LARGE}}$ improves on RoBERTa$_{\text{BASE}}$ by 5.3%. This contrasts with our results of a 3.9% and 2.7% increase respectively. Additionally, RoBERTa models' performance increase over BERT models of the same size is also reduced compared to the SQuAD 2.0 evaluations (1.6% vs. 7.6% for BASE models, 0.4% vs 4.4% for LARGE models).

While the transformer models showed similar results to humans in terms of accuracy, qualitatively the performance differs between the two in terms of incorrect responses. For instance, in humans an incorrect response would likely be an incorrectly identified entity or action (depending on the question type). However, the transformer models seemed to frequently answer questions simply by repeating the sentence or a non-constituent subpart, refraining entirely from selecting a single entity or action from the sentence. This is not an error that was seen in the human data. In addition to demonstrating a lack of awareness about particular semantic relationships in the sentence, this also suggests a lack of understanding of what a felicitous question response entails.

4.3 Discussion

The fact that BERT's performance is comparable (rather than superior) to humans is somewhat surprising given the vastly different way the two process language. For humans, the difficulty in processing garden path structures arises from the fact that language is presented sequentially: when encountering the first verb, people are unaware that there will be a second verb later in the sentence and thus are likely to parse the sentence incorrectly initially. BERT, on the other hand, receives all input simultaneously, so it should face less difficulty in parsing the sentence and assigning correct semantic roles. As seen in the results however, BERT seems to struggle with forming the correct relationship between agents and matrix verbs when there is an intervening relative clause, at rates at least as high as humans'.

Additionally, it is surprising that increasing the size of the model or changing the training procedures did not cause a corresponding performance increase in the models. This suggests that in order to understand the complex syntactic structures present in garden path sentences, one must do more than increase model parameters and the amount of training data. Rather, changing the architecture of the model itself may result in larger performance gains.[3]

5 Probe Task

Since our models perform similarly to humans on comprehension questions based on garden path sentences, we aimed to investigate precisely which semantic role each word in a garden path sentence is assigned, since human processing seems to be hampered by the misassignment of these semantic roles. In order to investigate the semantic roles assigned to the different entities in garden path sentences, we designed a probe (Alain and Bengio, 2016) trained on BERT's hidden states to better understand it's representation of the roles in question.

The linear classifier was trained on each model's embedding of a single token taken from sentences in which those words fall under the span of a semantic role of interest.

5.1 Training Materials

To create a training set for semantic roles, we used annotations from PropBank (Palmer et al., 2005). Each verb annotated by PropBank has a corresponding frame file in which verb-specific semantic relations are detailed. We chose to focus on two semantic roles. The first is PropBank's [PAG] tag, which represents a proto-agent type role across verbs. Selecting the other role was less straighforward, as the thematic roles in PropBank were not uniform across the areas of interest in the garden path sentences. For example, the two following stimuli sentences have differing thematic roles assigned to the first entity of the sentence (relevant verb in bold):

1. (The child)$_{\text{GOAL}}$ **bought** an ice cream cone smiled at the cashier.
2. (The child)$_{\text{DIRECTION}}$ **read** the story hugged the nanny.

In order to test the classifier on the largest number of stimuli possible, we chose to focus our probe on the entities tagged [GOL] (Goal role), as it has the most occurrences in our stimuli (143 instances total).

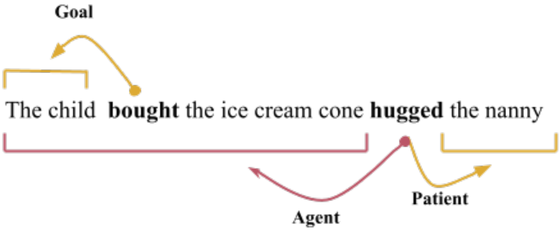


Figure 3: Here, the phrase *the child* acts as the Goal of *bought*, while the phrase *the child bought the ice cream cone* acts as the agent of *hugged*. The Patient label is presented for convenience, but not analyzed in this investigation.

We train a binary linear classifier for both Agent and Goal, on each layer of each model, in the form of a logistic regression classifier. This is due to the fact that some words may be constituents of arguments of multiple verbs, and therefore be assigned different roles, as displayed in Figure 3. This approach allows for detecting multiple different roles on each word.

5.2 Probe Design

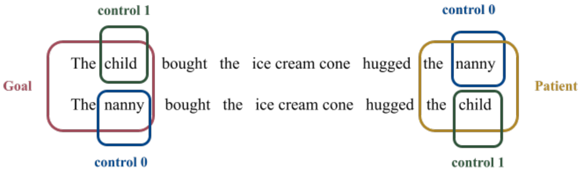


Figure 4: Words are labeled for goal and patient semantic roles (the latter of which is provided for convenience but is not addressed in this study), in addition to the control tag, which is randomly assigned to each unique word in the training data, i.e. 0 for *nanny* and 1 for *child*.

One concern when using classifiers to investigate the hidden states of a model is the possibility of the model achieving high accuracy by memorizing specific words' typical labels, rather than learning the patterns associated with the labels themselves. In

---

[3] Because relative clauses are the syntactic phenomena which make garden path processing difficult, finding a model which is able to correctly parse these may lead to advances in garden path understanding. While work on this is limited in English, work in other languages suggests some promising models are LSTM, PERT$_{\text{LARGE}}$, and GPT-3 (Song et al., 2022).

order to assess this level of memorization, we implemented a control task (Hewitt and Liang, 2019). This took the form of two additional classifiers which were trained on the sets of embeddings used by the Agent and Goal classifiers, but a binary label (rather than a semantic label) was randomly assigned to each unique word in the training set, as seen in Figure 4.

This allows us to assess the extent of memorization through the metric of selectivity:

$$selectivity = linguistic_{\text{ACC}} - control_{\text{ACC}}$$

The Agent semantic role is much more common in PropBank than the Goal role. In order to keep our classifiers equivalent in the number of samples they received, we created artificially split 50/50 +Role/-Role training sets for both Agent and Goal classifiers. All stimuli sets underwent a 80/20 train/test split. In order to maximize both performance and specificity in our probe of garden path sentences, we choose a regularization constant of 0.01 for our classifier, following the findings of Hewitt and Liang (2019).

To test our probe classifiers on the garden path sentences, we selected the layer from each model in which the associated classifier had the highest performance on Goal classification (which had overall lower accuracy than Agent classifiers, as shown in Table 2) for the PropBank sentences. We then applied the probe classifiers for both Agent and Goal roles to the first and second relevant entities in the same sentences that were analyzed in the question answering portion of this experiment. The sentences were sub-selected for those containing the Goal and Agent roles, leaving us with 24 sentences in each condition (garden path vs. local coherence structure, ambiguous vs. unambiguous).

5.3 Results

Overall, the linear classifier accuracies show that Agent and Goal semantic roles are decodable from token representations when trained on PropBank sentences. However, this information is not available to the same extent in each of the models. While the BERT models perform Agent classification and Goal classification accurately 80% and 70% of the time respectively, RoBERTa models' accuracy does not exceed 65% for both Agent and Goal classification as shown in Figures 5 and 6.[4] Additionally, the information change throughout the layers is also inconsistent between models with BERT models' accuracy peaking in the later layers of the model and RoBERTa classifier accuracies staying relatively level as the layers progress.

Due to the different inconsistencies in classification accuracies for the PropBank sentences across layers, we chose to analyze the garden path sentences using only the highest performing layer from each of the models. The layers chosen as well as their accuracy and selectivity metrics can be seen in Table 2.

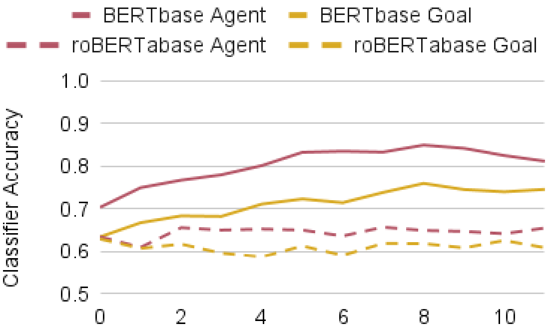


Figure 5: The average classifier accuracies from our Agent and Goal classifiers on our test set from PropBank, by model, across layers for $\text{BERT}_{\text{BASE}}$ and $\text{RoBERTa}_{\text{BASE}}$ models.

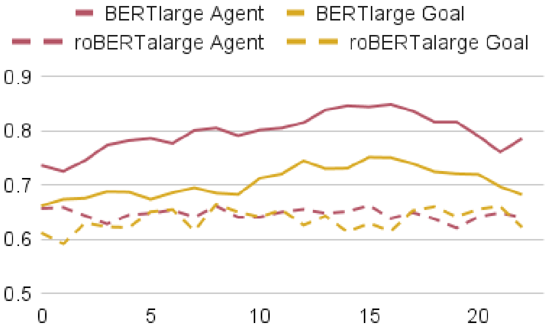


Figure 6: The average classifier accuracies from our Agent and Goal classifiers on our test set from PropBank, by model, across layers for $\text{BERT}_{\text{LARGE}}$ and $\text{RoBERTa}_{\text{LARGE}}$ models.

To analyze the garden path and local coherence sentences, both the Agent and Goal classifiers were applied to the noun associated with the first and second entities (e.g., *child* and *nanny* from Figure 3). The probabilities of each classifier assigning a given role to an entity is shown in Figure 7.

In garden path sentences, the $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ probes showed a low probability of

---

[4] Complete results for each layer of BERT and RoBERTa are available in Appendix A.

Model	Layer	Agent Accuracy	Agent Selectivity	Goal Accuracy	Goal Selectivity
BERT_B	9	84.92	34.56	75.96	28.04
BERT_LG	17	84.44	32.72	75.12	24.32
RoBERTa_B	11	64.2	13.56	62.6	15.88
RoBERTa_LG	10	66.08	15.36	66.44	14.8

Table 2: The accuracy and selectivity on the layers associated with the highest-performing classifier for each model, tested on the tokens from PropBank. Complete results for all layers are included in Tables 3 and 4 in Appendix A.

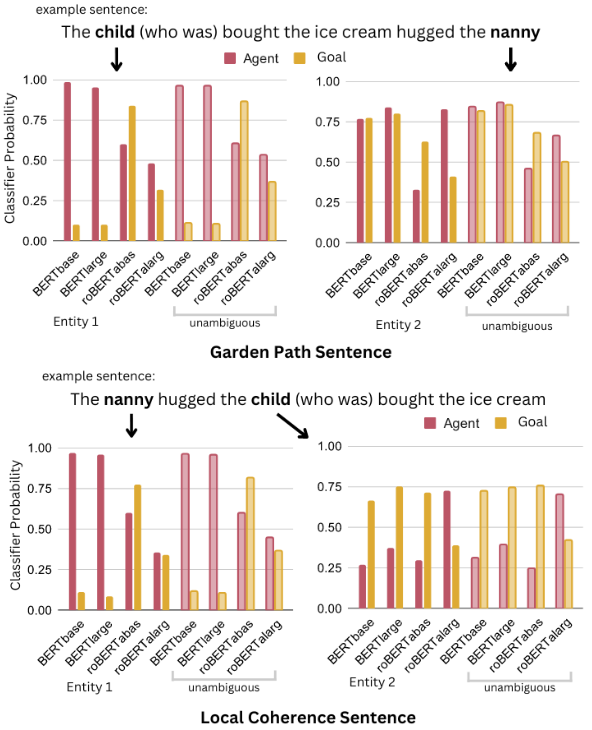


Figure 7: The average classifier probabilities assigned by our Agent and Goal classifiers to the first entity (left) and the second entity (right) in garden path and local coherence structure sentences in the selected layers. Exact numerical results are available in Appendix A.

the Goal role on the first entity in the sentence. $\text{RoBERTa}_\text{BASE}$, however, successfully showed high classifier probability for the Goal role in both the ambiguous and unambiguous sentence types, which could potentially suggest a better representation of the semantics of garden path sentences. However, this pattern also held for local coherence structures though (where the Goal role should not be assigned to the first entity), suggesting that $\text{RoBERTa}_\text{BASE}$ (nor any other model) were perfectly able to make semantic distinctions based on the differing syntactic structures. Finally, we also observe that the addition of disambiguating information into the sentence does not seem to have a large impact on classifier performance in general, for both different sentence types and models.

5.4 Discussion

Overall, the Agent classifiers were highly successful in labelling the first entity for both sentence structures (with the surprising exception of the $\text{RoBERTa}_\text{LARGE}$ classifier). In the case of the garden path sentences, the assignment of Goal to the first noun (i.e., *the child* in Figure 4), however, was extremely low, possibly indicating that most of the BERT models are not strongly representing *the child* as the goal of the verb *hugged*. This kind of semantic role interference is also what is hypothesized to impede human processing.

As a contrast, the second entity in the local coherence structures have low Agent and high Goal probabilities. This suggests that the models are better able to represent the Agent and Goal semantic roles in a local coherence structure, again mirroring the performance of humans and the model performance seen in the question answering task.

In contrast to the structure manipulation, ambiguity differences did not significantly change the classifiers' predictions, despite the question answering performance showing a larger gain in the unambiguous over ambiguous sentence contexts. Given that syntactic knowledge has been argued to be present in the model weights (Vilares et al., 2020; Hewitt and Liang, 2019), it is interesting that disambiguation (a lexical manipulation that clarifies syntax), does not improve performance in the probe task. This suggests that future syntactic investigations should focus not only on token representations, but also on other components of the models which may contribute to downstream task performance.

In terms of individual model performance, we find that no model perfectly aligns with human performance. In the local coherence structures, BERT_BASE and BERT_LARGE were most successful at assigning the correct roles to the correct entities, however they both misassigned the roles of the first entity in garden path structures. Interestingly, the semantic role predictions of the two were very similar, suggesting that increasing the model size does not fundamentally change what semantic information is held within token representations.

RoBERTa models, on the other hand, did not perfectly assign roles in either structure condition. RoBERTa$_{\text{BASE}}$ seems to prefer assigning Goal roles to every entity, while RoBERTa$_{\text{LARGE}}$ does the same with Agent roles. In this case, it does seem that increasing the model size does lead to the acquisition of different semantic knowledge, in addition to the differences from BERT models already mentioned.

Another area where these differences are observed is the extent to which classifier accuracy increased in later levels of BERT versus RoBERTA. In BERT, we observed an increase in performance in later layers, while for RoBERTa the performance stagnated across all layers. This was not due to a difference in classifier training, as all classifiers had identical hyperparameters and training corpora. Rather, we can conclude this is due to the RoBERTa model itself–its differing training procedures must lead to a fundamental difference in how semantic roles are processed from BERT (i.e., they are less strongly represented in the token weights themselves), given that the two have very similar performance on the downstream question answering task.

Because no model was able to assign all semantic roles perfectly in every condition (based on the probe task), we are left to conclude that the semantic knowledge within token representations of all models is imperfect and not based on deep syntactic knowledge. Rather, in some cases the models seem to be relying on heuristics to assign semantic roles. Such heuristics might include word order in the sentence, frequency of a particular semantic role, and linear proximity. Future investigations should aim to discover which heuristics are most relevant to langauge models' representations and performance, as well as how token representations may interact with other model components in order to achieve performance similar to humans on downstream tasks.

6 Conclusion

Overall, BERT-style transformer models do not perform significantly better than humans on garden path sentences in question answering. This suggests that, despite the temporal amodality of BERT's language processing, it still faces the same issues of misinterpretation that human speakers do in online sentence processing. Additionally, probe results suggest that BERT fails to assign the correct semantic roles to the entities in garden path sentences, despite showing successful assignments on other corpus sentences. This error is similar to human-style garden path misinterpretations, despite the many differences between human and model language processing (i.e. temporality, working memory demands).

Additionally, we observe differences in semantic role representations between the models tested—BERT models seem to make similar role predictions regardless of model size, while RoBERTa$_{\text{BASE}}$ makes different predictions than RoBERTa$_{\text{LARGE}}$. Furthermore, both RoBERTa models seem to have different semantic representations than the original BERT models, suggesting that particular training procedures and tasks can lead to widely different internal model states yet still show negligible impact on performance of downstream tasks.

Limitations

The garden path structures presented here are a phenomenon predominantly found in English. Structural ambiguities found in other languages vary widely, and so our ability to generalize about BERT's ability to process these ambiguities cross-linguistically is limited. Additionally, these types of garden path structures are relatively scarce in natural language, and it is possible performance would be higher if BERT were fine-tuned using these structures specifically (we choose not to do this in order to approximate the levels of experience humans have with these structures to obtain more natural comparison). Finally, the number of sentences tested in the probe task is low, due to variation of stimuli thematic roles.

Ethics Statement

The authors have no ethical concerns relating to the research presented in this paper.

Acknowledgements

The first two authors (TI and KW) contributed equally to this work. The research was supported by the NYUAD Research Institute under Grant G1001. We additionally thank the anonymous reviewers for their guidance and feedback on earlier versions of this paper.

References

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Kiel Christianson, Andrew Hollingworth, John F Halliwell, and Fernanda Ferreira. 2001. Thematic roles assigned along the garden path linger. *Cognitive Psychology*, 42(4):368–407.

Kiel Christianson, Steven G Luke, Erika K Hussey, and Kacey L Wochna. 2017. Why reread? Evidence from garden-path and local coherence structures. *The Quarterly Journal of Experimental Psychology*, 70(7):1380–1405.

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

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. *arXiv preprint arXiv:1903.03260*.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. *arXiv preprint arXiv:1909.03368*.

William Jurayj, William Rudman, and Carsten Eickhoff. 2022. Garden-path traversal within GPT-2. *arXiv preprint arXiv:2205.12302*.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Timothy J Slattery, Patrick Sturt, Kiel Christianson, Masaya Yoshida, and Fernanda Ferreira. 2013. Lingering misinterpretations of garden path sentences arise from competing syntactic representations. *Journal of Memory and Language*, 69(2):104–120.

Yixiao Song, Kalpesh Krishna, Rajesh Bhatt, and Mohit Iyyer. 2022. Sling: Sino linguistic evaluation of large language models.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Marten Van Schijndel and Tal Linzen. 2018. Modeling garden path effects without explicit hierarchical syntax. In *CogSci*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

David Vilares, Michalina Strzyz, Anders Søgaard, and Carlos Gómez-Rodríguez. 2020. Parsing as pretraining. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9114–9121.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Appendix A

Model	**Layer**	**Agent Acc.**	**Agent Selec.**	**Goal Acc.**	**Goal Selec.**
bert-base-uncased	1	70.36	21.88	63.44	11.8
	2	74.96	25.8	66.76	18.72
	3	76.72	27.16	68.36	19.08
	4	77.96	29.4	68.24	20.2
	5	80.12	33.32	71.12	20.56
	6	83.24	34.68	72.32	22
	7	83.48	34.72	71.44	22.56
	8	83.28	34.08	73.84	24.28
	9	84.92	34.56	75.96	28.04
	10	84.16	36.32	74.52	26.12
	11	82.48	34.44	74	24.76
	12	81.16	32.68	74.56	26
	Avg.	**80.24**	**31.59**	**71.21**	**22.01**
bert-large-uncased	1	67.92	15.44	62.84	14.32
	2	73.6	24.84	66.12	18.6
	3	72.52	22.52	67.36	18.6
	4	74.52	25.16	67.56	19.8
	5	77.36	27.24	68.76	20.6
	6	78.2	28.4	68.68	20.24
	7	78.6	31.28	67.36	18.48
	8	77.68	28.56	68.56	19
	9	80.08	31.16	69.48	23.44
	10	80.52	30.2	68.48	20.76
	11	79.12	29.6	68.24	19.64
	12	80.16	31.12	71.24	21.44
	13	80.52	30.72	72.04	22.44
	14	81.52	32	74.44	24.28
	15	83.84	33.96	73	23.36
	16	84.64	34.68	73.08	21.44
	17	84.44	32.72	75.12	24.64
	18	84.88	34.48	75.04	24.32
	19	83.64	35.28	73.92	23.92
	20	81.64	30.92	72.44	19.72
	21	81.64	34.36	72.08	18.64
	22	79.04	29.56	71.96	20.24
	23	76.08	27.28	69.68	17.96
	24	78.56	27.52	68.2	14.48
	Avg.	**79.20**	**29.54**	**70.24**	**20.43**

Table 3: Classifier accuracy and selectivity for each layer of BERT$_{\text{BASE}}$ and BERT$_{\text{LARGE}}$

Model	**Layer**	**Agent Acc.**	**Agent Selec.**	**Goal Acc.**	**Goal Selec.**
roberta-base-uncased	1	63.48	12.36	62.96	13
	2	61	16.4	60.76	12.12
	3	65.6	17.24	61.76	12
	4	65.04	15.6	59.64	7.56
	5	65.28	17.64	58.72	6.08
	6	65.04	18.68	61.32	10.6
	7	63.64	17.4	59.08	6.32
	8	65.72	14.36	61.92	13.52
	9	64.96	18	61.88	13.48
	10	64.72	13.64	60.88	11.76
	11	64.2	13.56	62.6	15.88
	12	65.52	16.56	60.92	6.76
	Avg.	**64.51**	**15.95**	**61.04**	**10.76**
roberta-large-uncased	1	62.68	14.64	60.24	9.16
	2	65.64	13.36	61.12	10.96
	3	65.8	14.84	59.12	6.92
	4	64.32	14.28	63	14.4
	5	62.8	12.8	62.28	14.92
	6	64.44	10.48	62.16	9.48
	7	64.76	13.52	65.04	14.8
	8	65.36	15.16	65.44	17.68
	9	64.04	11.72	61.56	9.28
	10	66.08	15.36	66.44	14.8
	11	64.04	11.72	65	14.92
	12	64.04	16.68	64.08	12.76
	13	65	15.92	65.4	17
	14	65.48	16.12	62.6	16.6
	15	64.76	16.16	64.28	13.36
	16	65.08	11.16	61.36	16.28
	17	66.16	14.92	62.92	10.8
	18	63.8	16.56	61.48	10.52
	19	64.88	13.44	65.24	14.68
	20	63.72	10.96	65.96	13.76
	21	62.08	10.08	64.12	14.12
	22	64.12	12.88	65.52	17.48
	23	64.84	13.56	66.04	13.8
	24	63.96	12.28	62.2	9.8
	Avg.	**64.49**	**13.69**	**63.44**	**13.26**

Table 4: Classifier accuracy and selectivity for each layer of $\text{RoBERTa}_{\text{BASE}}$ and $\text{RoBERTa}_{\text{LARGE}}$

Model	**Layer**	**Sentence Type**	**Amb.**	**Entity 1 Agent Probability**	**Entity 1 Goal Probability**	**Entity 2 Agent Probability**	**Entity 2 Goal Probability**
bert-base uncased	9	Garden Path	Amb.	0.983	0.0989	0.768	0.773
			Unamb.	0.964	0.110	0.846	0.819
		LC-A	Amb.	0.968	0.114	0.266	0.667
		LC-U	Unamb.	0.962	0.119	0.314	0.723
bert-large uncased	17	Garden Path	Amb.	0.954	0.102	0.836	0.798
			Unamb.	0.962	0.108	0.871	0.855
		Local Coherence	Amb.	0.959	0.087	0.370	0.751
			Unamb.	0.959	0.105	0.393	0.747
roberta-base uncased	11	Garden Path	Amb.	0.600	0.838	0.326	0.624
			Unamb.	0.606	0.864	0.457	0.680
		Local Coherence	Amb.	0.601	0.774	0.297	0.715
			Unamb.	0.598	0.815	0.250	0.757
roberta-large uncased	10	Garden Path	Amb.	0.479	0.320	0.827	0.411
			Unamb.	0.536	0.368	0.667	0.502
		Local Coherence	Amb.	0.356	0.337	0.727	0.389
			Unamb.	0.449	0.367	0.700	0.423

Table 5: Detailed probe classifier results on first and second entities in garden path and local coherence test sentences for highest performing layer in each model investigated.

Models Teaching Models: Improving Model Accuracy with Slingshot Learning

Lachlan O'Neill, Nandini Anantharama, Satya Borgohain, Simon D. Angus
SoDa Labs, Monash Business School, Monash University, Clayton Australia

Abstract

One significant obstacle to the successful application of machine learning to real-world data is that of labeling: it is often prohibitively expensive to pay an ethical amount for the human labor required to label a dataset successfully. Human-in-the-loop techniques such as active learning can reduce the cost, but the required human time is still significant and many fixed costs remain. Another option is to employ pretrained transformer models as labelers at scale, which can yield reasonable accuracy and significant cost savings. However, such models can still be expensive to use due to their high computational requirements, and the opaque nature of these models is not always suitable in applied social science and public use contexts.

We propose a novel semi-supervised method, named Slingshot Learning, in which we iteratively and selectively augment a small human-labeled dataset with labels from a high-quality "teacher" model to slingshot the performance of a "student" model in a cost-efficient manner. This reduces the accuracy trade-off required to use these simpler algorithms without disrupting their benefits, such as lower compute requirements, better interpretability, and faster inference. We define and discuss the slingshot learning algorithm and demonstrate its effectiveness on several benchmark tasks, using ALBERT to teach a simple Naive Bayes binary classifier. We experimentally demonstrate that Slingshot learning effectively decreases the performance gap between the teacher and student models. We also analyze its performance in several scenarios and compare different variants of the algorithm.

1 Introduction

In standard computational linguistics modeling, the typical strategy for achieving high model accuracy is to obtain a large fraction of high-quality human labels on the dataset at hand. However, it is not uncommon in applied social science settings to find that the scope of human labeling of a dataset is highly constrained (Liew et al., 2014). For example, it may be prohibitively expensive to pay an ethical amount for human labeling. Alternatively, the project's institutional access to raw corpora may have ended, or perhaps the domain experts are no longer available.

One solution is to use an active learning approach where a model is trained concurrently with the labeling process to "maximize a model's performance gain while annotating the fewest samples possible"(Ren et al., 2021). While this can be more efficient than unguided labeling, it still requires significant human labeling resources as the learning process is fundamentally human-guided. Some issues are relatively fixed costs, such as bias, repeatability, and initial training requirements; this enforces a non-trivial lower bound on the efficiency of human-in-the-loop active learning methods. Active learning also does not help if the data has already been collected, and there is no way to collect further data.

Alternatively, one might directly apply a large, pre-trained (and possibly fine-tuned) transformer model to obtain state-of-the-art outcomes (Brown et al., 2020; Schick and Schütze, 2020; Wei et al., 2021). Pre-trained models are well suited to learning from relatively small amounts of data due to the large amount of prior knowledge they already contain. But such a strategy comes with complications for applied social science, including high (and rising) costs (Patterson et al., 2021; Schwartz et al., 2019; Mayfield and Black, 2020) a lack of model interpretability (Yang et al., 2021; Zafar et al., 2021), and issues with repeatability when using models hidden behind an API.

Given the power of modern machine learning models, a natural question arises: Can the power of modern machine learning architectures (such as transformer models) be harnessed to enhance the accuracy of standard models where human labeling

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3233–3247
May 2-6, 2023 ©2023 Association for Computational Linguistics

is finished?

Here, we introduce and evaluate *Slingshot Learning*, an efficient, low-cost, widely applicable, and powerful methodology to enhance the accuracy of standard inference models. Slingshot Learning adopts a student–teacher framework, such that the standard model (the 'student') is iteratively taught by the transformer model (the 'teacher'). Akin to the gravitational slingshot maneuver, in which a spacecraft uses the gravity of a large body to increase its own speed, the student model leverages the knowledge and performance of the teacher model, in a way that allows it to express where it expresses low certainty in its predictions, essentially asking "questions" of the teacher model. This iterative behavior can be modified, or entirely replaced with purely random behavior through the selection of hyperparameters.

We demonstrate that, under slingshot learning, the accuracy of the student model can be at a relatively low cost. After the algorithm finishes iterating, the student model can be used as a compromise between the high performance of the complex teacher model, and the inherent benefits of the student model, which might include reproducibility, interpretability, computational efficiency, and compute costs that are orders of magnitude lower than the teacher model. This student model can then be used to label the rest of the corpus, and/or label future, unseen data.

2 Background

Active learning (AL) seeks to minimize the human labeling cost of training a model while collecting enough data to successfully train the model (Ren et al., 2021). It has proven popular in recent years due to the prevalence of large, unlabelled datasets, and methods are increasingly leveraging powerful, pre-trained models such as BERT (Grießhaber et al., 2020), but active learning is still a fundamentally human-led labeling approach which adopts the pitfalls of experimental research involving humans. The ability to repeat experiments in the first place (let alone with any consistency), expand their scope and analyze the underlying models are all taken for granted in machine learning but, once humans are involved in the labeling process, the difficulty of such tasks can vary from non-trivial to impossible. Iterative research design is also very difficult without significantly increasing the cost and time requirements of the overall experiment.

Large language models such as GPT-3 (Brown et al., 2020) have achieved exceptional performances in NLP tasks under zero-shot and few-shot learner constraints, making these a compelling approach for low-resource tasks (Chia et al., 2019). GPTs have been used in semi-supervised settings to generate pseudo-labels via training offline/student models (Chia et al., 2019). The study by Wang et al. (2021) leverages the use of GPT-3 to reduce costs in comparison to human annotators. In one of their labeling strategies, they employ GPT-3 (teacher) to generate a fully labeled dataset which is then used to train an offline model (student), a transformer model (Liu et al., 2019). While they show improved performance of the offline model on the fully labeled GPT-3 dataset in comparison to using human annotators, the performance on multiclass classifications was lower than that of human annotators and showed no improvement after increasing the number of GPT-3 labels. The authors also explore an active learning version, where the low-confidence labels of GPT-3 or the most challenging examples are relabelled using human annotators and the result is used to train the offline model. While this approach demonstrates better performance, it has the same pitfalls as other active learning approaches. Our approach assumes a low budget and no access to human annotators beyond a small "seed" labeled dataset.

In the realm of model interpretability, a similar approach is that of surrogate models (SM), which employ an interpretable simple model to explain the behavior of a more complex ML model (Rey and Neuhäuser, 2011). They include global models that explain the overall predictions and local models that provide local explanations on the predictors by varying their sampling strategies. The global models are trained on the same data set (or a similar data set) as the complex model (Molnar, 2022, ch 8.6). The output of the surrogate is then compared to the complex model to determine how well the surrogate replicates the complex model within an arbitrary threshold. The ones that employ sampling use simpler models to gain insight into interpretability through locally selected regions from the complex model's predictions (Ribeiro et al., 2016; Lundberg and Lee, 2017). The goal or output here is interpretability and mostly requires the continued use of the underlying complex model. However, our intention is not just interpretability but to build a stable simple model that is standalone.

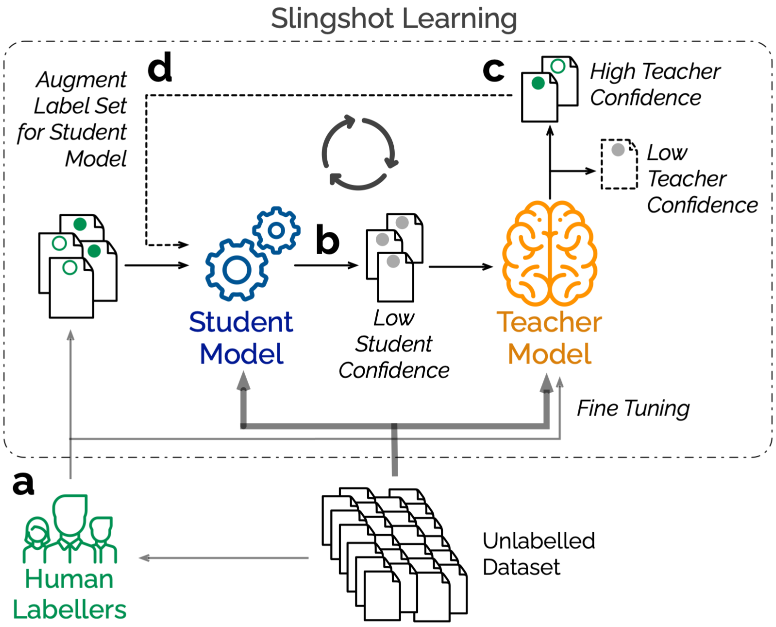


Figure 1: **Slingshot Learning.** A small fraction of the dataset is labeled by human labelers (a), which are used to train the Student Model and then identify low-confidence examples, a subset of which are passed to the fine-tuned Teacher Model (b). Where the Teacher Model has high inferential confidence on the examples (c), these are added to the human-labeled dataset for further training of the Student Model (d), and the cycle iterates.

The goal is to train the simpler model to make better predictions. Further, we are limited by our training dataset size. The idea is to also leverage a largely unlabelled dataset into training.

Knowledge distillation (KD) is another well-known approach that uses the teacher-student model (Hinton et al., 2015). The teacher - a deep teacher network - distills its knowledge onto a smaller model. The smaller models are also neural nets, and draw feature learnings from the parameters and activations of the intermediate layers of the teacher model. The intention here is to deploy a high-performing, compressed, smaller model in a low-resource environment (Gou et al., 2021) and thus they lack easy interpretability and transparency. While there have been some studies that use KD in an interpretable model (Liu et al., 2018; Che et al., 2016), the data size limitation discussed earlier applies to this approach as well.

Our goal is to build a standalone model that can achieve high accuracy despite limited training data, through the use of large unlabelled data sets.

3 Slingshot Learning

Here we explain slingshot learning from an intuitive view, and then present a more rigorous algorithmic definition.

3.1 Intuition

Consider a scenario where we have a large dataset of unlabelled data, far larger than could be labeled by hand, and we want to label the whole dataset and/or train a model to label future, unseen data. One solution is to hand-label a small portion by hand and feed the rest to a state-of-the-art model (such as a modern Transformer-based model, for NLP problems). While this is quicker and cheaper than labeling by hand, it can still be expensive (especially if using an API, or a large model requiring significant GPU compute capabilities). Additionally, these complex models take a while to run, making them potentially unsuitable for real-time applications. Of course, a simpler and more efficient model can be used, but this generally comes at the cost of decreased model performance.

In slingshot learning, we use the more expensive (HQ) model to label a portion of the unlabelled dataset, and then leverage that to *slingshot* the performance of the smaller (LQ) model, much like a gravitational assist maneuver accelerates a small spacecraft using the gravity of a large celestial object. Under this training paradigm, we train a "high-quality" (HQ) model on our ground truth dataset, use this model to iteratively teach the LQ model by labeling a portion of the unlabelled dataset, and

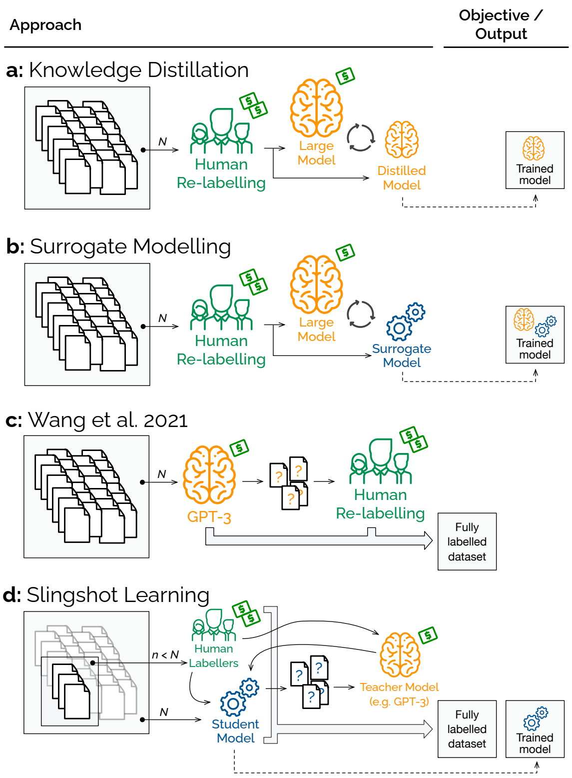


Figure 2: **Model comparison.** See text for details.

then use the low-quality model to label the rest of the data and/or make future predictions on unseen data.

We assume the number of human labels is fixed and cannot be increased - instead, we will use the HQ model (trained on the human-labeled dataset) as our more accurate (and more expensive) oracle. Our goal is thus to minimize the number of HQ predictions, while successfully labeling the dataset and/or training the LQ model. This is the crux of the slingshot learning approach; although the implementation details and hyperparameter selections might vary, the core concept of using a high-quality (but expensive) model to improve the performance of a lower-quality (but cheap) model remains constant.

The comparison to slingshot learning with other existing methods is presented in Table 1. The primary objectives of SM and KD are interpretability and high-accuracy compressed models respectively. Slingshot achieves both these objectives. Comparing slingshot to Wang et al. (2021), the latter continues to use human annotators for labeling, while SSL supports automated labeling. In contrast to Wang et al. (2021), KD's trained smaller models can perform automated labeling but require a large data set to achieve a desirable accuracy.

Model Characteristics	Wang et al.	SM*	KD*	SSL*
Automated labelling	✗	✗	✓	✓
Standalone re-usable model	✗	✗	✓	✓
Interpretability	✗	✓	✗	✓
Budget restriction	✓	✗	✗	✓
Small labelled dataset	✓	✗	✗	✓
Leverages Unlabelled data	✓	✗	✗	✓

Table 1: Comparison of existing methods with slingshot learning. SM*: Surrogate Models, KD*: Knowledge Distillation, SSL*: Slingshot Learning

3.2 The Slingshot Learning Algorithm

In Figure 3, a high-level pseudocode implementation of the core slingshot algorithm is described. Depending on the choice of hyperparameters, the algorithm's performance might be further optimizable. For example, in the purely random variant of the algorithm (discussed below), there is no need to compute the LQ predictions at each iteration.

Note that the algorithm does not assume that the confidence values of the HQ and LQ models are comparable with one another. The algorithm only relies on the relative ordering of confidence values from one of the models at a time and does not compare confidence values between the two models. The relationship between prediction confidence and accuracy is not necessarily monotonic, but we assume it will be somewhat correlated and treat it as monotonic for the purposes of the algorithm.

3.2.1 Hyperparameters

There are three hyperparameters in the slingshot learning algorithm:

n the number of data points selected for prediction by the *low-quality* model at each iteration.

m the number of data points selected for prediction by the *high-quality* model at each iteration.

k the number of data points added to the labeled set at each iteration.

We discuss the different choices for these hyperparameters throughout this paper. Note the necessary inequality $n \geq m \geq k$.

3.3 Expected Performance

Part of the intuition of slingshot learning stems from the idea that the improved performance of the HQ model over the LQ model stems from two sources: both the inherent superior ability of the HQ model to understand the *population* as a whole

```
procedure SLINGSHOT LEARNING ALGORITHM(L, U, V, $n, m, k$)
    $HQ \leftarrow$ Train_HQ($L$)    ▷ Train the HQ model on L
    $LQ \leftarrow$ Train_LQ($L$)    ▷ Train the LQ model on L
    while Not converged & more data do
        $D_n =$ Sample($L, n$)    ▷ Sample $n$ elements from the unlabelled dataset.
        $P_n = LQ(D_n)$    ▷ Compute LQ predictions and confidence for each datum (note we only use the confidence).
        $D_m =$ LowConfidence($D_n, P_n$)    ▷ Choose $m$ lowest confidence LQ predictions
        $P_m = HQ(D_m)$    ▷ Compute HQ prediction and confidence for each datum.
        $D_k =$ HighConfidence($D_m, P_m$)    ▷ Choose $k$ highest confidence HQ predictions.
        $L \leftarrow L \cup D_k$    ▷ Add these to the labelled set.
        $U \leftarrow U \setminus D_k$    ▷ Remove these from the unlabelled set.
        $LQ =$ Train_LQ($L$)    ▷ Train a new LQ model on the full labeled dataset.
        Validate($LQ$)    ▷ Evaluate the performance of the LQ model on the validation set.
    end while
end procedure
```

Figure 3: Slingshot Learning Algorithm: Note that L is the labeled set, U is the unlabelled set, V is the validation set, LQ is the low-quality model, and HQ is the high-quality model. n, m, and k are as described in Section 3.2.1.

and also the superior ability of the HQ model to learn from the *sample*.

Let us denote the *intrinsic* capability of a model m to approximate the population, if given an arbitrarily large sample (i.e. "all the data", or an "infinite sample") from that population, as I_m. Since our sample will generally not cover the entire population, the true model accuracy will have a penalty applied dependent on the true*sample size* dependent capability of the model, which we denote as S_m. In our notation, we will assume the penalty value S_m is positive, and subtract it from the total model performance. We can then denote the LQ model performance as $LQ = I_{LQ} - S_{LQ}$, and the HQ model performance as $HQ = I_{HQ} - S_{HQ}$. We can then express the performance difference between the two models as:

$$HQ - LQ = I_{HQ} - S_{HQ} - (I_{LQ} - S_{LQ}) \quad (1)$$

Rearranging terms, we find:

$$HQ - LQ = \Delta_I + \Delta_S \quad (2)$$

Where $\Delta_I = (I_{HQ} - I_{LQ})$ and $\Delta_S = (S_{LQ} - S_{HQ})$. Intuitively, Δ_I is the performance uplift due to the *inherent* superiority of the HQ model (which cannot be conferred to the LQ model), and Δ_S is the extra penalty applied to the LQ model's performance due to the superiority of the HQ model given the *sample size* being used for training. In the limit, with infinite data, we would therefore expect Δ_I to approach some constant value greater than zero, while Δ_S goes to zero.

In situations where Δ_I is the dominant term, slingshot learning may have lower utility. But in scenarios where Δ_S is the dominant term (or at least a significant component of the overall performance delta), slingshot learning has the potential to bridge part of the knowledge gap between the HQ and LQ models. We conjecture that there are many instances where simpler, cheaper models would compare well to more complex and expensive models, and that much of the gap between traditional and newer models stems from both an increase in the I terms *and* a decrease in the S sample-size penalties for smaller samples.

3.4 Algorithm Variations

Depending on our choice of hyperparameters n, m, and k, we can modify the behavior of the slingshot learning algorithm in a variety of ways. An empirical comparison of the differences in resulting behavior from these three variants can be found in the Results section.

3.4.1 Deterministic Variant

We can choose to have the LQ model evaluate the entirety of the unlabelled dataset in each iteration, by setting $n = \|U\|$[1], where U is the set of data points that remain unlabelled at a given iteration.

This removes the stochasticity from the algorithm, assuming the HQ and LQ models remain constant under a given input (since m and k are only used to determine how many data points to

---

[1] We use the value $n = -1$ to denote this in our code.

take from the n randomly selected data points). While it will take longer to run - potentially a lot longer for larger datasets, since the unlabelled dataset must be passed through the LQ model at every iteration of the algorithm - his approach ensures that the LQ model can select the lowest confidence data points at every iteration.

3.4.2 Random Variant

Alternatively, we can choose to set $k = m = n$, meaning that every data point that is passed to the student model is sent to the teacher model for evaluation, and every data point sent to the teacher model for evaluation is added to the corpus. This is akin to purely random sampling, without allowing the student to provide input on the training process.

3.4.3 Stochastic Variant

We refer to any configuration of the hyperparameters n, m, k which is neither deterministic nor purely random as a "stochastic" configuration, where there is a combination of random sampling and student input during the sampling process. In stochastic slingshot learning, $k <= m <= n$ and $k < n$.

4 Experimental Design

4.1 Models

We initially ran experiments with ALBERT teaching Naive Bayes, ALBERT teaching SVM, and SVM teaching Naive Bayes. SVM teaching Naive Bayes proved difficult, as the SVM was very often not significantly more powerful than the Naive Bayes model. We performed some initial experiments with ALBERT teaching SVM, but the required computation time proved insurmountable. However, preliminary results indicate that SVM is a reasonable candidate as a student of ALBERT under the slingshot learning paradigm.

4.1.1 Model Details

ALBERT (Lan et al., 2020) from the HuggingFace Model repository (Wolf et al., 2020) (using the "albert-base-v2" model)

SVC (Cortes and Vapnik, 1995), an SVM classification model, as implemented by Sci-Kit Learn (Pedregosa et al., 2011). Not used in the final results.

Naive Bayes as implemented in Sci-Kit Learn (Pedregosa et al., 2011).

4.2 Datasets

We evaluate performance on four popular benchmark binary text classification datasets from the HuggingFace Datasets repository, as shown in Table 2.

4.3 Training

We tested a number of values for each of the three hyperparameters, n, m, and k. We evaluated $n = 128$, $n = 1024$, and $n = -1$. m and k were set to either $k = m = n$ or $m = n/4, k = m/4 = n/16$. When $n = -1$, a value of 1024 was used to calculate m and k. Each experiment was repeated up to 10 times. Experiments where the HQ (ALBERT fine-tuned) model was not at least 15% more accurate than the LQ (Naive Bayes) model were discarded, as we found they added a lot of noise to the dataset (and, in any case, slingshot learning is most useful when the models are of significantly different strength). Some number of samples [2] (one of 2500, $20,000$, or $50,000$) was taken from each dataset, and then randomly split into 80% training and 20% validation sets.

A total of $1,480$ experiments were run, representing up to five runs [3] of each of the 52 hyperparameter choices, over each of the three datasets. For each experiment, 95% of the dataset labels (chosen randomly but with a fixed seed of 42) were masked, to mimic a semi-supervised learning problem with a proportionally low (but reasonable) amount of ground-truth data.

Experiments were performed using virtual machines with four CPUs and a single NVIDIA T4 GPU, using AWS Batch. SVM and Naive-Bayes models were trained with default values from the sklearn library, as was the HuggingFace ALBERT model (which was trained for a total of three epochs over the dataset, with a batch size of 16).

5 Results

We have included all experimental results in tabular form in the supplementary materials. Source code can be found here.

When performing these experiments, we ran the full slingshot learning algorithm until there is no more data left to classify. However, in practice,

---

[2] For the Rotten Tomatoes dataset, we used all the data for the 20k and 50k runs since it has less than 20,000 data points.

[3] Due to our use of AWS spot instances, some experiments failed due to loss of the VM. Each experiment had at least 8 successful runs, before pruning for the disparity between the HQ and LQ models.

			Large				**Small**			
Dataset	**n**	**Variant**	$\delta_{10\%}$	$\delta_{30\%}$	δ_{max}	#	$\delta_{10\%}$	$\delta_{30\%}$	δ_{max}	#
Amazon Polarity	128	random	0.296	0.384	0.449	1	0.224	0.214	0.489	8
	1024	random	0.080	0.145	0.328	1	0.266	0.266	0.335	5
	1024	stochastic	0.178	0.346	0.478	3	0.208	0.344	0.507	8
IMDB	128	random	0.464	0.661	0.697	2	0.052	0.144	0.401	4
	1024	random	0.547	0.634	0.665	2	-0.249	-0.249	-0.204	3
	1024	stochastic	0.466	0.512	0.665	3	-0.070	-0.119	0.060	5
Rotten Tomatoes	128	random	0.170	0.305	0.527	11	0.008	0.099	0.226	3
	1024	random	0.235	0.334	0.507	16	0.216	0.216	0.293	4
	1024	stochastic	0.228	0.336	0.570	9	-0.020	0.047	0.236	4
Yelp	128	random	0.034	0.085	0.215	17	0.002	0.009	0.035	10
	1024	random	0.038	0.096	0.229	18	0.005	0.005	0.013	9
	1024	stochastic	0.028	0.084	0.218	14	0.001	0.002	0.025	9

Table 2: Here we show the performance of slingshot learning aggregated over the "small" samples (2,500 samples) and "large" samples (20,000 and 50,000 samples). All "small" experiments masked 90% of the dataset, leaving 10% as the seed for slingshot learning. All "large" experiments masked 99% of the dataset, leaving 1% as the seed for slingshot learning. Note that some of the experiments have very small sample sizes - these results should not be relied upon. The algorithm is random if $n = m = k$, and stochastic otherwise (we exclude $n = -1$ from this table as this configuration is not practical).

slingshot learning is most useful (and most distinct from other techniques such as surrogate models) when *early stopping* is applied. Our results demonstrate that there are cases where most of the accuracy gains occur early, and other cases where progress is more consistent. However, throughout almost all the results there is a consistent, monotonic increasing accuracy, as well as the *rate of change of accuracy* (that is, there are few sudden accuracy spikes). This indicates that simply monitoring the LQ model accuracy, and stopping the algorithm when it seems to be plateauing (or when resources have been fully expended, of course) might be a reasonable method of applying the algorithm. One could also attempt to quantitatively determine when the accuracy plateaus (if it ever does), but further exploration of this is beyond the scope of this paper.

Throughout these results, we refer to the δ ("Delta") metric. Intuitively, the Delta metric δ_{S_i} is the ratio between the amount of knowledge that the teacher model has "taught" the student model (equal to the difference between the performance of S_i and the baseline LQ model), and the knowledge gap between the teacher and the student (equal to the difference in their accuracy).

More rigorously, we define the Delta metric δ as

$$\delta_{S_i} = \frac{A(S_i) - A(S_{-1})}{A(T) - A(S_{-1})} \tag{3}$$

where S_i is the LQ model trained at the $i - th$ iteration of the algorithm, and $A(m)$ is the accuracy of model m on the dataset at hand. S_{-1} is the performance baseline of the LQ model before any iterations of the slingshot learning algorithm. T is the HQ model.

5.1 Empirical Results

Table 2 includes the average δ values at the 10% and 30% milestones (that is, after 10% or 30% of the total number of slingshot iterations), as well as the average point throughout the algorithm (as a percentage of the number of iterations taken) it reached its peak. The 10% and 30% points were chosen as reasonable benchmark points during our initial experiments. In the supplementary materials, we include extended tables including the first time the LQ model reached accuracy within 5%, 1%, and 0.1% of its peak accuracy.

In Figures 4, 5, and 6, we plot the δ values of the low-quality (LQ) model slingshot learning throughout the training process for each experiment (grey lines), along with the average δ across experiments (thick red line). We also show the 95% confidence interval of the delta value (envelope), and, on the x-axis, the earliest iteration in which the LQ model obtains statistically equivalent performance (filled markers: $\alpha = 0.01$; open: $\alpha = 0.10$) to that of the final iteration (100%).[4] Figure 4 shows a best-case scenario for slingshot learning, where there is a significant increase in performance early in the iteration process. Figure 5 demonstrates how, in some cases, accuracy uplift can be more consistent throughout the slingshot learning cycle - in these cases, there is a stronger trade-off between the stopping point for slingshot learning and the potential accuracy gains of continuing the algorithm.

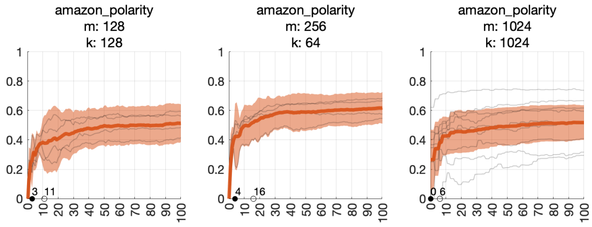


Figure 4: **Best-case scenario (Amazon Polarity dataset, 50k):** Large dataset sees the HQ model able to transfer most of the information to the LQ model without going through the whole dataset. Early stopping has less of a penalty.

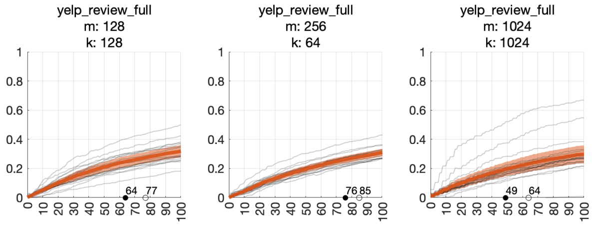


Figure 5: **More difficult scenario (Yelp dataset, 50k):** With much data, the HQ model can give information to the LQ model, but gains continue almost linearly throughout the process, penalizing early stopping.

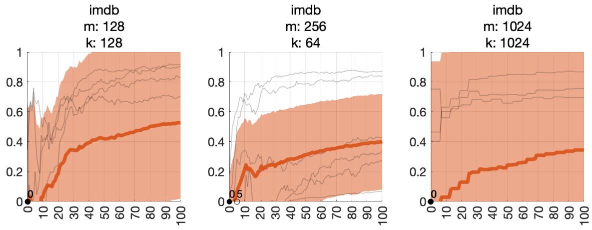


Figure 6: **High variability (IMDB dataset, 20k):** Slingshot learning performance was relatively poor on the IMDB dataset, though not always (see the supplementary materials for more information).

6 Discussion

These empirical results show slingshot learning to be an effective and reliable method for improving model performance in semi-supervised classification problems. While the exact values of δ vary between datasets, each dataset shows a clear correlation between the number of iterations and the performance of the resulting model, as well as (in most cases) the effect of diminishing returns as the number of iterations is increased.

Due to the degeneracy problem discussed below, we have elected to remove any runs for which the initial HQ model did not show at least a 15% improvement in accuracy over the LQ model baseline (before any iterations of the algorithm).[5]

6.1 Hyperparameter Selection

We saw no consistent trends between the experiments where $n = m = k$ and $n > m > k$. It does seem that, in situations where an $n = m = k$ configuration would be too computationally expensive, an $n > m > k$ configuration (or perhaps an $n > m = k$ configuration) is a reasonable option. The supplementary materials include many more plots and more comprehensive tables for the 20,000 and 50,000 sample experiments.

6.2 Degenerate Cases

While slingshot learning has proven effective, and relatively reliable under the right conditions, it is not guaranteed to improve performance in all cases. Slingshot learning does not appear to be very beneficial in cases where the HQ model is not significantly more powerful than the LQ model. We found erratic behavior in such scenarios, to the extent that we chose to exclude any results from our tables and figures for which the HQ model did not perform with at least 15% higher accuracy than the LQ model.

6.3 Benefits of Slingshot Learning

Slingshot learning allows simpler models to capture some of the performance of more powerful models, and as discussed throughout this paper, this has tangible benefits in cost and compute requirements for several common machine learning applications. We have also identified several other benefits that this approach provides.

---

[4] In each figure: left panel, $n = 128$; middle, right panels, $n = 1024$.

[5] The full, unpruned data table has been included in the supplementary materials.

6.3.1 Enabling Model Selection/Trade-offs

Traditional machine learning methods, such as Linear/Logistic Regressions, Naive Bayes, decision trees (Kotsiantis, 2013) and tree-based ensemble methods (Friedman, 2001; Chen and Guestrin, 2016), and SVMs, have existed in the literature for decades, and they have many advantages and trade-offs between them. However, as transformer-based models (and deep-learning models in general) become more powerful, it is becoming more and more difficult to justify the use of these traditional methods regardless of the benefits and trade-offs that may come with them, such as interpretability, efficiency, accuracy, and inherent applicability to the problem at hand.

The ability to choose between different models as the "low quality" model in the slingshot learning paradigm restores some of the power of model selection. For example, users can choose between simpler, more easily interpretable methods such as Linear/Logistic Regression or Naive Bayes, or opt for more complex models such as SVMs for the sake of accuracy. Through the use of slingshot learning, users can make these decisions and trade-offs based on their requirements, without sacrificing as much accuracy.

6.3.2 Enabling Experimental Repeatability

One negative effect of the proliferation of commercial, API-based deep learning models is the inability to reliably repeat experiments. APIs, and the models beneath them, are subject to change without notice, and will of course be improved over time. An unmodified experiment might perform significantly differently on two different days, in a manner entirely beyond the control of the user.

One interpretation of slingshot learning is as a method of capturing a "slice" of a large model, using a smaller model to store the captured information. These slices can then be stored and used at a later date, both reliably and without further interaction with the original model. Aside from the computational and financial benefits of this, the ability to take such a slice and store it locally enables repeatable, shareable experimental models. This may have legal and ethical implications, especially if it negatively affects the financial viability of commercial machine learning offerings, but a detailed discussion of this is beyond the scope of this paper.

6.3.3 Facilitating Interpretable Models

Deep learning models are not inherently interpretable by default, and although techniques such as LIME (Di Cicco et al., 2019) and SHAP (Lundberg and Lee, 2017) somewhat facilitate the interpretation of such models, there have been calls in recent years for inherently interpretable deep learning methods (Yang et al., 2021). By leveraging these complex, non-interpretable models to train simpler, interpretable models, slingshot learning can facilitate the creation of models which are both powerful *and* interpretable. Of course, the idea of training smaller models with larger ones is not new, and techniques such as knowledge distillation (Gou et al., 2021) and surrogate models (Rey and Neuhäuser, 2011; Molnar, 2022) are an area of active research.

6.3.4 Early Stopping

The iterative slingshot learning algorithm allows users to decide when to stop iterating when they determine that the model has likely converged. As discussed previously, this tends to happen relatively early in the iteration process, and further iterations tend to yield diminishing returns. Though there are no guarantees as to whether the model has converged (or is close to its final, converged performance) at a given iteration, the experimental data included in this paper can provide a guide to the expected future behavior of the model.

6.4 Further Work

Slingshot learning has proven to be useful in these applied NLP classification problems, but there is no reason it cannot be applied to other ML domains such as regression, multi-class classification, or text generation. Further experiments in these domains, and others, will be an important phase in determining the scope of the applicability of slingshot learning. We expect the algorithm would potentially be useful in a wide variety of scenarios.

Further analysis and testing of the different potential configurations of the slingshot learning algorithm's hyperparameters will also be critical to understanding how the values of n, m, and k can be selected to optimize the algorithm's performance in different use cases. We also plan to explore a potential "meta" slingshot learning algorithm, where progressively simpler models are taught from the model above them in the hierarchy, to propagate knowledge and avoid "overwhelming" the LQ model with the power of the HQ model.

7 Limitations

Our experiments were performed on English NLP datasets, collated from the internet. The results in this paper are intended and should be taken, purely in the context of evaluating the performance of the slingshot learning algorithm. We have no reason to believe slingshot learning would not be equally applicable to other languages, domains, problems, etc., and have simply chosen these datasets due to their prevalence and familiarity in the literature.

We conjecture that different (perhaps more complex) language structures might make slingshot learning more useful. We do not have expertise in this area - as we discussed in the paper, we consider it necessary to run further experiments evaluating the application of slingshot learning in other domains to be necessary. We are hopeful, however, that the method can be applied to many problems besides the ones discussed in this paper.

8 Ethics Statement

This paper discusses methods that are intended to enhance the ability of researchers within domains such as social sciences to more easily apply state-of-the-art NLP algorithms to their work. This inherently comes with the risk of bias, and it is entirely possible that seed datasets based on incomplete or biased datasets will cause slingshot learning to serve as an "amplifier". We encourage caution when applying machine learning to social or other "real-world impactful" domains, and doubly so when extrapolating from smaller amounts of data using machine learning. This paper serves as an initial evaluation and high-level exploration of slingshot learning, and we would not recommend using slingshot learning in situations that could potentially have negative consequences. Further exploration is certainly required, both specific to slingshot learning and regarding the more general use of machine learning in these scenarios.

On the other hand, methods like slingshot learning can empower researchers and allow them to leverage tools that would otherwise be infeasible. It also has other benefits, which are discussed in the paper - for example, it facilitates taking a "snapshot" of a black box model (such as an API-based NLP model). It is our intention, and our hope, that slingshot learning is a useful tool that helps democratize machine learning within the social sciences and other applied domains.

Acknowledgements

We thank all those who have reviewed this paper for their advice and contributions, which have included improvements to explanations and wording, discussion of relevant literature (particularly regarding active learning), and numerous other additions.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Zhengping Che, Sanjay Purushotham, Robinder Khemani, and Yan Liu. 2016. Interpretable deep models for icu outcome prediction. In *AMIA annual symposium proceedings*, volume 2016, page 371. American Medical Informatics Association.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. Association for Computing Machinery. Event-place: San Francisco, California, USA.

Yew Ken Chia, Sam Witteveen, and Martin Andrews. 2019. Transformer to cnn: Label-scarce distillation for efficient text classification.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Vincenzo Di Cicco, Donatella Firmani, Nick Koudas, Paolo Merialdo, and Divesh Srivastava. 2019. Interpreting deep learning models for entity resolution: An experience report using lime. In *Proceedings of the Second International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*, aiDM '19, New York, NY, USA. Association for Computing Machinery.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232. Publisher: JSTOR.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Daniel Grießhaber, Johannes Maucher, and Ngoc Thang Vu. 2020. Fine-tuning bert for low-resource natural language understanding via active learning. *arXiv preprint arXiv:2012.02462*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Sotiris B Kotsiantis. 2013. Decision trees: a recent overview. *Artificial Intelligence Review*, 39(4):261–283. Publisher: Springer.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*. OpenReview.net.

Jasy Suet Yan Liew, Nancy McCracken, Shichun Zhou, and Kevin Crowston. 2014. Optimizing features in active machine learning for complex qualitative content analysis. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 44–48.

Xuan Liu, Xiaoguang Wang, and Stan Matwin. 2018. Improving the interpretability of deep neural networks with knowledge distillation.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Elijah Mayfield and Alan W Black. 2020. Should you fine-tune BERT for automated essay scoring? In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 151–162, Seattle, WA, USA → Online. Association for Computational Linguistics.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, page 165–172, New York, NY, USA. Association for Computing Machinery.

Christoph Molnar. 2022. *Interpretable Machine Learning*, 2 edition.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.

David A. Patterson, Joseph Gonzalez, Quoc V. Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R. So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training. *CoRR*, abs/2104.10350.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. 2021. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40.

Denise Rey and Markus Neuhäuser. 2011. *Wilcoxon-Signed-Rank Test*, pages 1658–1659. Springer Berlin Heidelberg, Berlin, Heidelberg.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few-shot text classification and natural language inference. *CoRR*, abs/2001.07676.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. Green AI. *CoRR*, abs/1907.10597.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? GPT-3 can help. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4195–4205, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *CoRR*, abs/2109.01652.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jian Yang, Gang Xiao, Yulong Shen, Wei Jiang, Xinyu Hu, Ying Zhang, and Jinghui Peng. 2021. A survey of knowledge enhanced pre-trained models. *CoRR*, abs/2110.00269.

Muhammad Bilal Zafar, Philipp Schmidt, Michele Donini, Cédric Archambeau, Felix Biessmann, Sanjiv Ranjan Das, and Krishnaram Kenthapadi. 2021. More than words: Towards better quality interpretations of text classifiers. *CoRR*, abs/2112.12444.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

A Datasets

Table 3 includes some summary information about each of the datasets used in our experiments.

B Extended Results

The raw experimental data can be found in the supplementary materials, in CSV format. Selected plots and tables are included here.

Note that the Rotten Tomatoes dataset has less than $20,000$ elements, and thus all elements were used in every experiment of sizes $20,000$ and $50,000$ (i.e. with no random sampling). Rows with less than three experiments (i.e. $N < 3$) were dropped due to lack of statistical certainty.

The tables show the maximum δ value achieved throughout the training process ("Max"), along with when we first came within 5%, 1%, and 0.1% of the maximum. The place where the maximum value occurred is omitted as (due to noise) it is not a reliable or consistent statistic. The tables also include the δ values found after completing 10%, 30%, and 50% of the slingshot iterations.

Dataset	**Training Set Size**	**Test Set Size**
IMDB (Maas et al., 2011)	25,000	25,000
Amazon Polarity (McAuley and Leskovec, 2013)	3,600,000	400,000
Rotten Tomatoes (Pang and Lee, 2005)	8,530	1,066
Yelp Reviews (Zhang et al., 2015)	650,000	

Table 3: Description of datasets used in our experiments.

Dataset	**Mask**	**n=m=k**	**Max**	**$\sim$5%**	**$\sim$1%**	**$\sim$.1%**	**Max**	**δ 10%**	**δ 30%**	**δ 50%**	**N**
Amazon	0.9	FALSE	0.48	0.20	0.69	0.99	0.81	0.18	0.35	0.41	3
Amazon	0.99	TRUE	0.55	0.14	0.37	0.78	0.74	0.37	0.43	0.48	17
Amazon	0.99	FALSE	0.46	0.15	0.44	0.94	0.75	0.16	0.31	0.35	11
Amazon	0.999	TRUE	0.23	0.08	0.21	0.03	0.68	0.17	0.20	0.22	3
Amazon	0.999	FALSE	0.42	0.08	0.45	0.58	0.80	0.24	0.35	0.37	4
IMDB	0.9	FALSE	0.67	0.17	0.65	0.93	0.95	0.47	0.51	0.58	3
IMDB	0.99	TRUE	0.49	0.43	0.75	0.98	0.99	0.19	0.31	0.39	11
IMDB	0.99	FALSE	0.33	0.28	0.49	0.88	0.74	-0.03	0.07	0.18	8
R. Tom.	0.9	TRUE	0.51	0.43	0.83	1.00	0.93	0.24	0.33	0.43	16
R. Tom.	0.9	FALSE	0.57	0.34	0.72	0.94	0.92	0.23	0.34	0.45	9
R. Tom.	0.99	TRUE	0.67	0.40	0.86	0.91	0.95	0.36	0.52	0.57	5
Yelp	0.9	TRUE	0.23	0.41	0.86	1.00	1.00	0.04	0.10	0.15	18
Yelp	0.9	FALSE	0.22	0.34	0.84	0.98	0.99	0.03	0.08	0.13	14
Yelp	0.99	TRUE	0.21	0.26	0.71	1.00	0.99	0.04	0.09	0.14	19
Yelp	0.99	FALSE	0.24	0.31	0.77	0.98	0.99	0.03	0.09	0.14	17

Table 4: Experimental results, $\mathbf{n = 1024}$, combined data from sizes $20,000$ and $50,000$.

Dataset	**Mask**	**n=m=k**	**Max**	**$\sim$5%**	**$\sim$1%**	**$\sim$.1%**	**Max**	**δ 10%**	**δ 30%**	**δ 50%**	**N**
Amazon	0.99	TRUE	0.475	0.096	0.295	0.682	0.683	0.331	0.390	0.421	13
Amazon	0.999	TRUE	0.521	0.198	0.753	0.939	0.946	0.333	0.431	0.465	3
IMDB	0.99	TRUE	0.435	0.325	0.568	0.938	0.760	-0.008	0.210	0.294	13
R. Tom.	0.9	TRUE	0.527	0.349	0.794	0.963	0.917	0.170	0.305	0.378	11
Yelp	0.9	TRUE	0.215	0.352	0.832	0.988	0.994	0.034	0.085	0.132	17
Yelp	0.99	TRUE	0.250	0.289	0.766	0.992	0.938	0.039	0.107	0.158	18

Table 5: Experimental results, $\mathbf{n = 128}$, combined data from sizes $20,000$ and $50,000$.

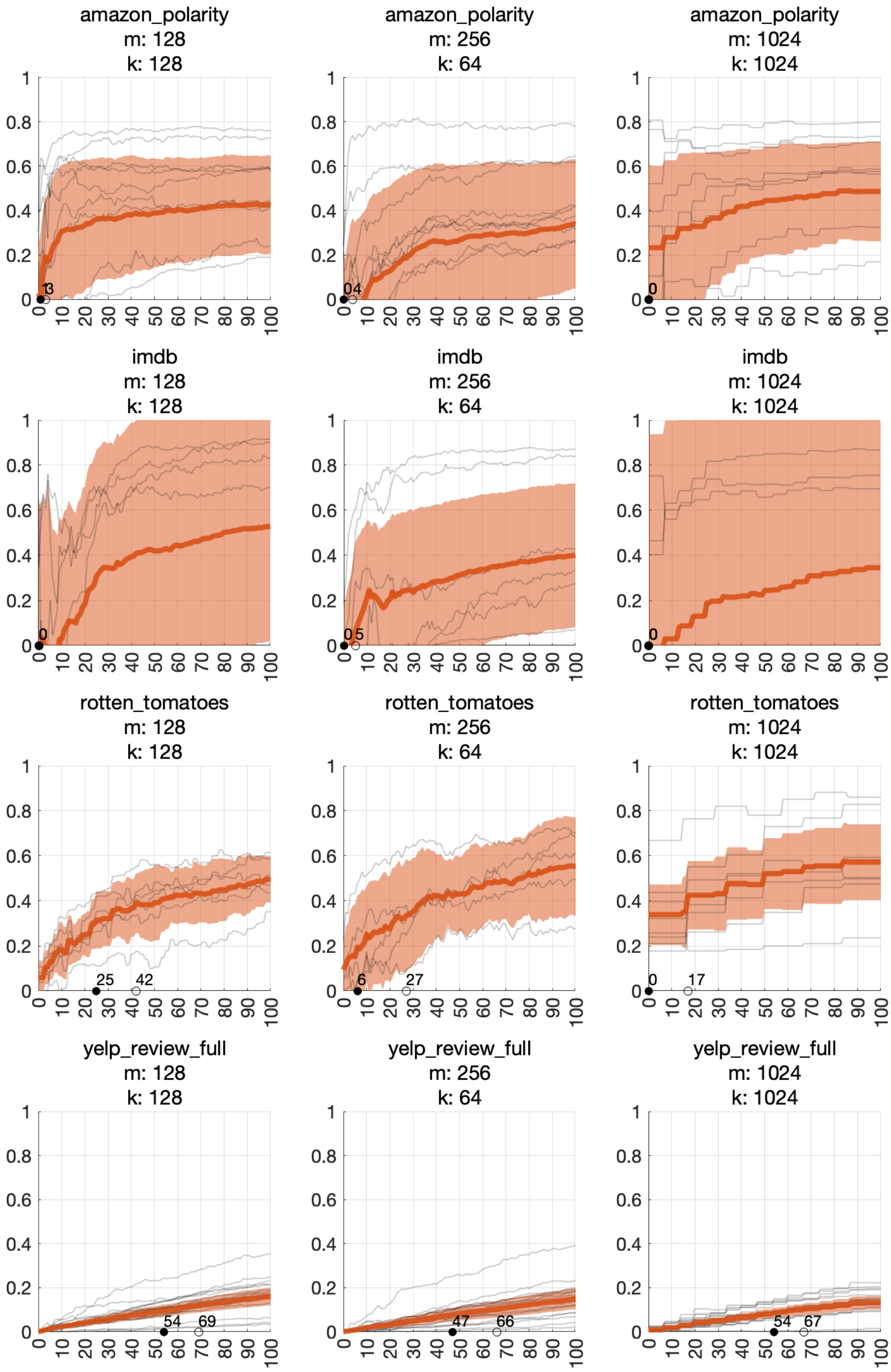


Figure 7: **Results for datasets of size** $20,000$ **(or less for Rotten Tomatoes).**

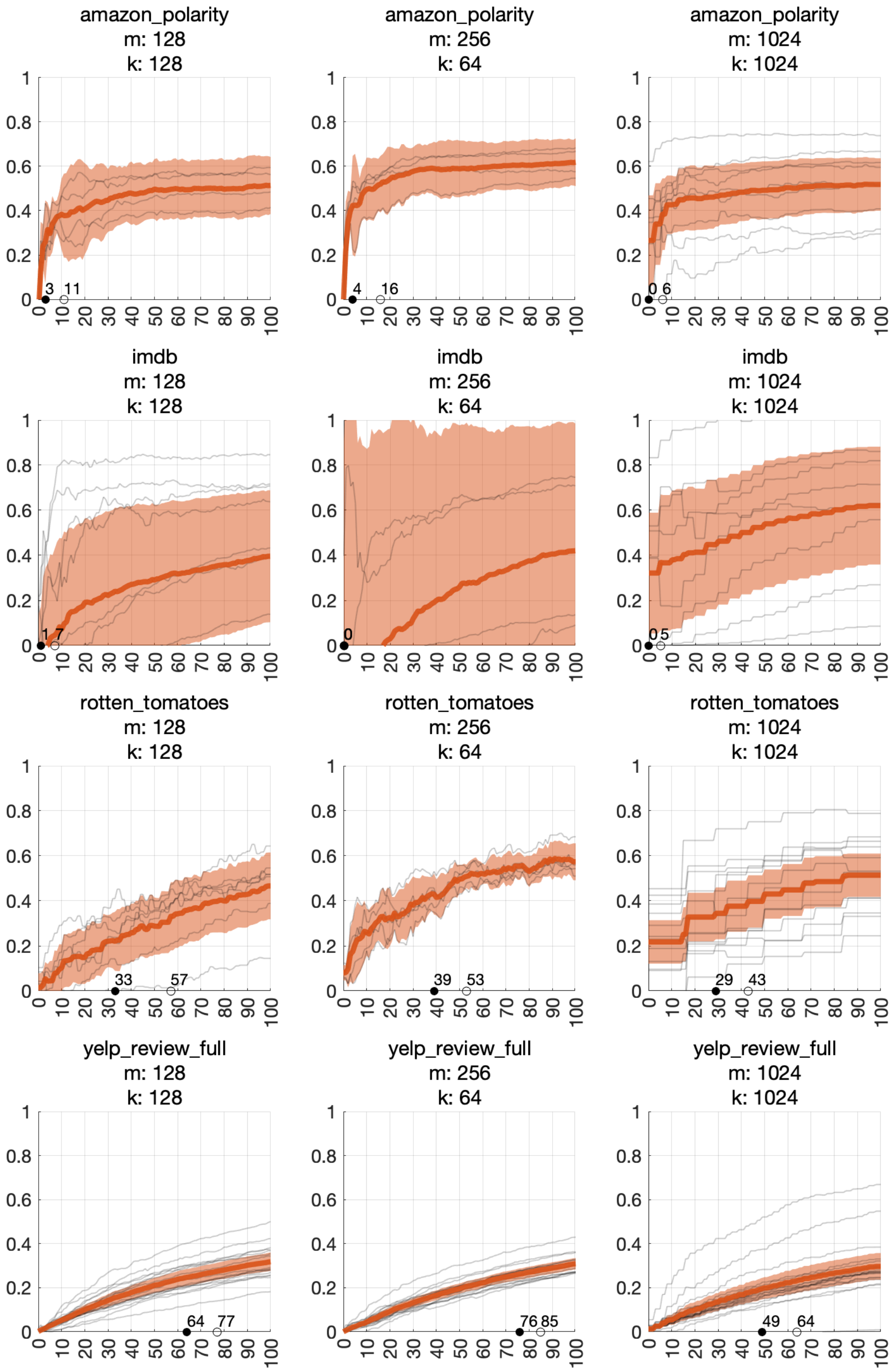


Figure 8: **Results for datasets of size** 50, 000 **(or less for Rotten Tomatoes).**

A Federated Approach for Hate Speech Detection

Jay Gala*
AI4Bharat
jaygala24@gmail.com

Deep Gandhi*
University of Alberta
drgandhi@ualberta.ca

Jash Mehta*
Georgia Institute of Technology
jmehta73@gatech.edu

Zeerak Talat
MBZUAI

Abstract

Hate speech detection has been the subject of high research attention, due to the scale of content created on social media. In spite of the attention and the sensitive nature of the task, privacy preservation in hate speech detection has remained under-studied. The majority of research has focused on centralised machine learning infrastructures which risk leaking data. In this paper, we show that using federated machine learning can help address privacy the concerns that are inherent to hate speech detection while obtaining up to 6.81% improvement in terms of F1-score.

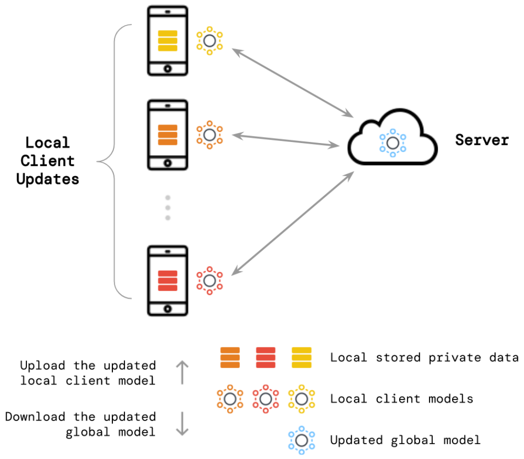


Figure 1: Federated Learning: A centralised model is hosted on a server and is distributed to client devices, these compute weight updates, and transmit the updates for aggregation into the centralised model. The centralised model is then redistributed to client devices.

1 Introduction

Content moderation is a topic that intersects across multiple fundamental rights, e.g., freedom of expression and the right to privacy; and interest groups, e.g. scholars, legislators, civil society, and commercial entities (Kaye, 2019). The availability of public datasets has been crucial to the development of computational methods for hate speech detection. However, public data contains risks for those whose content is available. On the other hand, privately held data, e.g., data held by corporate entities, holds risks for those who are reporting content. Such risks may be actualised through information leaks in models (Hitaj et al., 2017) or the transmission of data (Shokri and Shmatikov, 2015), and can impact people's safety and livelihood.

In this work, we apply Federated Learning (FL, McMahan et al., 2017) to address the lack of privacy in hate speech detection. FL is a privacy-preserving training paradigm for machine learning that jointly optimises for user privacy and model performance. We posit that privacy is necessary for users whose content is flagged and users who are flagging content alike. We thus operationalise privacy, in the context of hate speech detection and federated learning, to mean privacy in terms of the content of reported content, and the report itself. FL is an apt training paradigm for tasks in which training data is highly sensitive, as FL is designed to mitigate risks of information leaks while also dealing with a high number of end-users, information loss, and label imbalances (Lin et al., 2022; Priyanshu and Naidu, 2021; Gandhi et al., 2022). We apply the FL algorithms FedProx (Li et al., 2020) and Adaptive Federated Optimization (FedOpt, Reddi et al., 2021) to 5 machine learning algorithms. We evaluate our approach on 8 previously published datasets for hate speech detection. While using FL often implies a trade-off between privacy and performance, we obtain performance improvements of up to 6.81% in F1-score. We find that that models trained using FL outperform centralised models across multiple tests (e.g., derogatory language, spelling variation, and pronoun reference) in HateCheck (Röttger et al., 2021).[1]

---

[*] Equal contribution.

[1] All code is made available on Github.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3248–3259
May 2-6, 2023 ©2023 Association for Computational Linguistics

2 Prior Work

Although the areas of hate speech detection and FL have each been subject to extensive research, the study of their intersection remains in its infancy.

Federated Learning Federated Learning is a privacy-preserving machine learning paradigm that aims to reduce privacy risks by decentralising data processing onto client devices (i.e., personal devices), thereby foregoing the need for transmitting "raw" user data, and thus minimising risks of personal data leaks caused by transmission of data.[2] In FL, the machine learning model is located in two places: On a centralised server, and on client devices, which hold instances of the model distributed from the centralised model.Client devices use the model to compute model updates. The model updates are then transmitted to the server and aggregated by the centralised model, which is redistributed to the client devices. However, not all transmitted weight updates are aggregated into the model. FL operates with a notion of data loss in its design, which is emulated by selecting a fraction of clients whose updates are aggregated. Thus, FL paradigm uses less data to train a models.

In our experiments, we apply two FL algorithms: FedProx and FedOpt (Reddi et al., 2021). FedProx introduces a proximal term to the Federated Averaging algorithm (FedAvg, McMahan et al., 2017). FedAvg averages the weights computed on participating client devices in a round. FedProx introduces a proximal term that functions as a regulariser to the weight updates transmitted by participating clients, which penalises local weight updates that diverge from the global model. The FedAvg algorithm can thus be understood as a special case of FedProx with the proximal term set to 0.0.

FedOpt (Reddi et al., 2021) extends the adaptive optimisation strategies from centralised optimisation (e.g., Adam (Kingma and Ba, 2015) and Adagrad (Duchi et al., 2010)) to explicitly account for client and server optimisation. FedOpt handles server optimisation distinctly from client optimisation, by introducing a state to the server-side optimisation routine. . This distinct handling of server-side optimization enables more accurate and heterogeneity-aware FL models, which can speed up convergence.

FL has been applied to a number of tasks, including emoji prediction (Ramaswamy et al., 2019; Gandhi et al., 2022), next-word prediction for mobile keyboards (Yang et al., 2018), pre-training and fine-tuning large language models (Liu and Miller, 2020), medical named entity recognition (Ge et al., 2020), and text classification (Lin et al., 2022). For instance, Lin et al. (2022) used FL to fine-tune a DistilBERT model to perform classification on the 20NewsGroup dataset (Lang, 1995) using three different FL algorithms: FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), FedOpt (Reddi et al., 2021)) under non-IID partitioning.

In a closely related study, Basu et al. (2021) apply FL, using the FedAvg algorithm to fine-tune large language models to detect depression and sexual harassment from small Twitter data samples. They find that using large language models such as BERT and RoBERTa outperform distilled language models such as DistilBERT. Our work extends on Basu et al. (2021) by introducing additional FL algorithms and extending to a multi-class setting for hate speech detection.

Thus, our work extends on prior work by i) applying FL to the task of multi-class hate speech detection, a task which has proven difficult in part due to the complex nature of pragmatics (Röttger et al., 2021) and hate mongers seeking to evade content moderation infrastructures (Crawford and Gillespie, 2016); ii) using the FedProx and FedOpt algorithms rather than the FedAvg algorithm, thereby reducing model vulnerability to divergent weight updates; and iii) providing an in-depth analysis of federated model performances.

Hate Speech Detection Prior work on hate speech detection has primarily focused on privacy-agnostic machine learning paradigms, using centralised models for classification. Such work has investigated a number of machine learning models (e.g. SVMs (Karan and Šnajder, 2018), CNNs (Park and Fung, 2017), and fine-tuned language models (Swamy et al., 2019b)) and the development of resources (e.g. Talat and Hovy, 2016). Recently, Fortuna et al. (2021) proposed a standardisation of classes across 9 publicly available datasets and studied the generalisation capabilities of BERT, fastText, and SVM models. In their work they found limited success in inter-dataset generalization. Our work thus extends on the task of hate speech detection by introducing privacy-preserving methods to multi-class hate speech detection. In doing so, the privacy of those who flag content and those whose content is flagged remain intact.

---

[2] See Gitelman (2013) for a discussion on 'raw' data.

Category	Merged count	Comb	Change
aggression	6,950	6,950	-
aggressive hate speech	1,561	1,561	-
covert aggression	4,242	4,242	-
hate speech	13,222	13,205	−0.13%
insult	7,879	7,779	−1.27%
misogyny sexism	5,000	5,000	-
none	189,869	188,550	−0.69%
offensive	19,192	19,192	-
overt aggression	2,710	2,710	-
racism	1,978	1,978	-
severely toxic	1,597	1,527	−4.38%
threat	480	470	−2.08%
toxicity	40,316	40,134	−0.45%

Table 1: Label count of the raw datasets and `Comb`

3 Data

We combine our dataset using the standardisation schema proposed by Fortuna et al. (2021).

`Comb` We reuse 8 of the 9 datasets used by Fortuna et al. (2021) to form `Comb`.[3] `Comb` then consists of the datasets proposed by Talat and Hovy (2016); Davidson et al. (2017); Fersini et al. (2018); de Gibert et al. (2018); Swamy et al. (2019b); Basile et al. (2019); Zampieri et al. (2019) and the Kaggle toxic comment challenge.[4] We perform a stratified split of all training data into training (70%), validation (10%), and test (20%) sets.[5]

Data Cleaning We address issues of extreme class imbalance in `Comb` by removing the "abusive" category as it only contains 2 documents. Following an in-depth analysis of the `Kaggle` dataset we find that the maximum length of tokens in the dataset is 4950 while the median length of tokens in `Comb` is 26. Moreover, we find that the longest 1% of documents in the `Kaggle` dataset do not contain unique tokens. Removing the longest 1% of comments reduces the maximal document length to 727 tokens (see Appendix B.3 for further detail). Following our data cleaning processes, `Comb` comes to consist of 293,300 documents (see Table 1 for an overview of changes).

4 Experiments

We experiment with 5 machine learning models in their centralised and federated settings: Logistic Regression Bi-LSTMs (Hochreiter and Schmidhuber, 1997), FNet (Lee-Thorp et al., 2022), DistilBERT (Sanh et al., 2019) and RoBERTa (Liu et al., 2019). We measure their performance using weighted F1 scores. The centralised models form our baselines, while the federated models form our experimental models. For the Logistic Regression and Bi-LSTMs, we perform word-level tokenisation using SpaCy (Honnibal and Montani, 2017). For the FNet, DistilBERT, and RoBERTa, we use the tokenisers provided with each model.[6]

4.1 Federated Training

FL is a machine learning training paradigm that distributes training onto client devices. All client devices are split into overlapping subsets and the training data is partitioned and uniformly distributed to client devices. A random client subset is selected for training in each round, and their locally computed weights are aggregated on the server. We train our models for 300 rounds for 1, 5, or 20 epochs per round, and set the client fraction to 10%, 30%, or 50% which are randomly sampled from 100 client devices. We perform hyper-parameter tuning for the client learning rate, server-side learning rate, and proximal term (see appendix B.1).

In our work, we conceptualise client devices as users who witness and report hate speech. We simulate the client devices and ensure that data is independently and identically distributed (I.I.D.) on client devices.[7] We use the FedProx and FedOpt algorithms to aggregate client updates on the server. FedProx introduces a regularisation constant to the server-side aggregation step, the proximal term to address issues of divergence in weights and statistical heterogeneity in FedAvg. FedOpt seeks to create more robust models by introducing a separate optimiser for the server-side model to account for data heterogeneity.

5 Analysis

Considering the baseline models in Table 4, we see that the Logistic Regression tends to underperform, while the RoBERTa model posts the best performances. Although FL-based models often outperform our baselines, we note that when FL

---

[3] The dataset proposed by (Founta et al., 2018) is not included as it was not available to us.

[4] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

[5] We do not use the test data provided with some datasets to ensure uniformity, as test sets are not provided with all datasets.

[6] Please refer to Appendix A for further experiments and analyses on the Vidgen et al. (2021) dataset.

[7] We use an I.I.D. setting for data as 40% of all social media users and 64% of those under 30 in the USA have experienced online harassment (Pew Research Center, 2021). I.e. while hate speech is not frequent, it is often experienced by users.

		Logistic Regression			**Bi-LSTM**			**FNet**			**DistilBERT**			**RoBERTa**		
		Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
c = 10%	**e = 1**	70.22	53.15	58.47	71.04	58.19	61.28	72.61	59.20	62.20	73.98	60.75	63.79	74.76	64.43	66.16
	e = 5	70.83	63.31	66.35	70.84	66.51	67.72	73.52	68.33	**70.42**	74.54	69.46	70.85	74.59	69.68	71.48
	e = 20	70.18	67.41	68.67	69.17	69.25	69.10	73.10	68.02	69.73	73.28	71.06	71.94	73.11	**71.48**	**72.07**
c = 30%	**e = 1**	**71.23**	53.50	58.89	71.58	58.82	61.72	73.62	61.13	63.97	74.84	64.03	66.14	75.02	64.33	66.41
	e = 5	70.82	64.44	67.01	70.65	65.90	67.27	73.35	68.30	70.36	74.82	69.44	70.68	74.41	69.98	71.81
	e = 20	70.30	**68.13**	**69.09**	69.34	**69.26**	**69.15**	72.35	68.03	69.74	73.33	71.39	72.15	73.65	70.86	71.96
c = 50%	**e = 1**	71.11	53.12	58.58	**71.59**	58.71	61.73	**73.93**	61.89	64.51	**74.88**	63.58	65.85	74.42	63.57	65.87
	e = 5	70.89	64.26	66.80	70.70	66.16	67.54	72.90	68.27	70.18	74.44	69.68	70.88	**74.90**	69.46	70.86
	e = 20	70.28	68.00	69.01	69.25	68.84	68.20	72.90	**68.42**	70.16	73.71	**71.51**	**72.34**	73.53	71.18	72.01

Table 2: Results of FedProx experiments on `Comb`.

		Logistic Regression			**Bi-LSTM**			**FNet**			**DistilBERT**			**RoBERTa**		
		Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
c = 10%	**e = 1**	68.29	52.48	58.46	**71.80**	58.34	61.70	72.64	59.78	62.49	74.51	63.87	65.03	72.02	64.21	64.57
	e = 5	68.20	59.38	63.14	70.63	63.73	66.10	72.39	69.11	70.51	74.33	69.61	70.48	**75.55**	69.44	70.34
	e = 20	**68.30**	59.56	63.23	69.74	65.32	67.27	71.87	**70.69**	**71.15**	72.21	**71.34**	**71.66**	73.17	**72.20**	**72.61**
c = 30%	**e = 1**	67.68	51.05	57.19	71.56	58.97	62.10	72.24	57.11	61.21	**74.90**	64.16	66.79	73.88	66.07	65.79
	e = 5	66.65	60.31	63.10	69.48	62.63	65.57	72.01	69.14	70.30	72.82	69.59	70.75	74.38	71.54	71.69
	e = 20	67.18	62.50	**64.60**	69.74	65.69	67.49	71.91	70.02	70.79	71.55	70.33	70.86	72.97	72.10	72.05
c = 50%	**e = 1**	67.25	54.85	59.82	71.35	59.63	62.59	**73.03**	62.28	64.64	73.31	63.98	65.64	74.85	66.80	67.75
	e = 5	66.63	60.21	63.04	69.56	63.02	65.58	70.63	68.06	69.21	72.53	69.66	70.80	73.78	71.27	71.18
	e = 20	66.70	**62.51**	64.41	69.16	**66.16**	**67.54**	70.98	69.74	70.21	70.65	68.99	69.69	72.67	70.51	71.51

Table 3: Results of FedOpt experiments on `Comb`.

	Centralised			**Federated**
	Precision	Recall	F1	F1
LogReg	69.11	57.45	62.20	69.09
Bi-LSTM	71.43	66.64	67.90	69.15
FNet	71.35	64.73	66.58	71.15
DistilBERT	73.99	69.01	69.39	72.34
RoBERTa	**75.45**	**70.58**	**71.03**	**72.61**

Table 4: Results for the centralised and best performing FL models. The FL models have been chosen across FedProx and FedOpt based on F1 scores.

models are trained with lower client fractions and epochs, they tend to be outperformed by the baselines. Models trained using FedProx outperform the centralised baselines (see table 2).[8] For instance, we see large improvements for FNet and Logistic Regression (4.5 and 6.8 points in terms of F1- score, respectively). Comparing the performances of models trained using FedOpt (table 3) with those trained using FedProx, we observe that the former (in particular FNet and RoBERTa) tend to outperform the latter for lower client fractions and epochs. In general, we find that the best FL models outperform their centralised counter-parts (see Tables 2 and 3). In fact, the best performing RoBERTa, DistilBERT, and FNet models trained using FL algorithms outperform their centralised baselines, with FNet obtaining a 3-4 point improvement over centralised models in terms of F1 score.[9]

While FL often indicates a trade-off between privacy and performance, we find that the best FL models outperform the centralised baselines. We believe that the improved performance stems from the dataset being split into smaller segments, in congruence with findings from prior work. For instance, Nobata et al. (2016) show that splitting data into smaller temporal segments helped improve classification performance overall. We believe that a similar effect may be evident with FL models that, by design split data into small segments and disregard a fraction of the clients. Further, it may be the case that some data within hate speech datasets hinders generalisation. Only using subsets of the data for training may therefore aid generalisation.

5.1 Hate Check Evaluation

This section extends the experiments to qualitatively evaluate the effectiveness of federated and centralised models under different axis of hate speech using HATECHECK (Röttger et al., 2021). HATECHECK is a suite of functional tests for hate speech detection models. HATECHECK provides an in-depth examination of model performances across different potential challenges for machine learning models trained for hate speech detection.

The HATECHECK (Röttger et al., 2021) dataset consists of 29 tests, 18 of which test for distinct expressions of hate while the remaining 11 test for non-hateful expressions. The dataset contains 3.728 labelled samples, 69% of which are 'Hate' and while the remaining 31% are labelled as 'Not-

---

[8] For tables 2 and 3, c refers to the client fraction used and e refers to the number of epochs on client devices.

[9] See Section 5.1 for an analysis using HateCheck (Röttger et al., 2021).

Functionality	**Accuracy (%)**														
	Logistic Regression			**Bi-LSTM**			**FNet**			**DistilBERT**			**RoBERTa**		
	Central	$\mathbf{F_{prox}}$	$\mathbf{F_{Opt}}$	**Central**	$\mathbf{F_{prox}}$	$\mathbf{F_{Opt}}$	**Central**	$\mathbf{F_{prox}}$	$\mathbf{F_{Opt}}$	**Central**	$\mathbf{F_{prox}}$	$\mathbf{F_{Opt}}$	**Central**	$\mathbf{F_{prox}}$	$\mathbf{F_{Opt}}$
F1: Expression of strong negative emotions (explicit)	96.4	**100.0**	97.1	80.7	**100.0**	95.7	75.0	**98.6**	97.9	90.0	**99.3**	87.9	89.3	87.9	**92.9**
F2: Description using very negative attributes (explicit)	95.0	**100.0**	97.9	65.7	99.3	99.3	87.1	**100.0**	**100.0**	96.4	**100.0**	97.7	92.9	93.6	**95.0**
F3: Dehumanisation (explicit)	97.9	**100.0**	94.3	77.9	**100.0**	**100.0**	85.0	**100.0**	**100.0**	97.1	**100.0**	93.6	90.7	**94.3**	**94.3**
F4: Implicit derogation	87.1	**95.7**	75.7	70.7	**94.3**	92.9	62.1	**99.3**	96.4	72.9	**82.9**	**82.9**	77.1	78.6	**80.7**
F5: Direct threat	90.2	**99.3**	94.0	80.0	**96.2**	88.0	82.0	**100.0**	98.5	88.0	**98.5**	91.7	91.0	**95.5**	91.7
F6: Threat as normative statement	94.3	**99.3**	96.4	80.7	**99.3**	98.6	70.0	**100.0**	**100.0**	90.0	**100.0**	94.3	**96.4**	90.7	91.4
F7: Hate expressed using slur	87.5	**99.3**	98.6	75.7	88.2	**96.5**	86.1	**98.6**	96.5	91.0	**94.4**	90.0	**88.2**	84.7	86.1
F8: Non-hateful homonyms of slurs	6.7	16.7	**43.3**	10.0	**43.3**	40.0	16.7	23.3	**26.7**	23.3	**50.0**	33.3	26.7	**40.0**	36.7
F9: Reclaimed slurs	4.9	4.9	**40.7**	2.5	**42.0**	27.2	9.9	11.1	**13.6**	6.2	7.4	**12.4**	4.9	16.1	**17.3**
F10: Hate expressed using profanity	**100.0**	**100.0**	**100.0**	93.6	**96.4**	**96.4**	94.3	**100.0**	**100.0**	97.9	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
F11: Non-hateful use of profanity	2.0	13.0	**38.0**	19.0	**47.0**	40.0	6.0	11.0	**19.0**	15.0	13.0	**16.0**	3.0	11.0	**17.0**
F12: Hate expressed through reference in subsequent clauses	**100.0**	**100.0**	90.0	91.4	**98.6**	**98.6**	86.4	**100.0**	**100.0**	95.0	**98.6**	96.4	90.0	**92.9**	92.1
F13: Hate expressed through reference in subsequent sentences	**100.0**	**100.0**	96.2	85.0	**96.2**	94.7	84.2	**100.0**	**100.0**	95.5	**99.3**	97.0	**96.2**	94.0	95.5
F14: Hate expressed using negated positive statement	92.9	**99.3**	84.3	57.9	89.3	**93.6**	52.1	**100.0**	**100.0**	77.9	**93.6**	76.4	61.4	81.4	**90.7**
F15: Non-hate expressed using negated hateful statement	6.0	30.0	**58.7**	25.6	**53.4**	41.4	27.1	23.3	**33.8**	17.3	27.1	**43.6**	31.6	51.1	**56.4**
F16: Hate phrased as a question	95.7	**100.0**	95.0	81.4	93.6	**96.4**	61.4	**95.0**	**95.0**	92.1	**96.4**	91.4	82.1	**95.0**	87.9
F17: Hate phrased as an opinion	99.0	**100.0**	92.5	89.5	**99.0**	97.7	81.2	**100.0**	94.0	91.0	**98.5**	93.2	86.5	**93.2**	87.2
F18: Neutral statements using protected group identifiers	20.6	56.3	**77.0**	42.1	**75.4**	62.7	50.0	55.6	**69.0**	69.0	68.3	**75.4**	69.0	**92.9**	87.3
F19: Positive statements using protected group identifiers	18.0	42.9	**80.0**	46.0	**64.6**	41.3	45.5	37.0	**59.3**	38.6	49.2	**73.1**	48.1	78.8	**92.1**
F20: Denouncements of hate that quote it	1.7	19.0	**55.5**	14.5	**44.5**	28.3	31.2	**45.1**	33.5	16.2	**48.6**	38.7	15.6	36.4	**37.0**
F21: Denouncements of hate that make direct reference to it	4.2	15.6	**47.5**	21.3	**46.8**	36.9	27.7	22.7	**34.0**	12.1	16.3	**30.5**	19.1	39.0	**43.3**
F22: Abuse targeted at objects	10.8	45.1	**70.8**	46.1	**70.8**	52.3	53.8	47.7	**58.5**	55.4	**66.2**	**66.2**	60.0	73.8	**75.4**
F23: Abuse targeted at individuals (not as member of a prot. group)	4.6	29.2	**61.5**	29.2	**69.2**	52.3	20.0	24.6	**38.5**	20.0	29.2	**38.5**	23.1	15.4	**69.2**
F24: Abuse targeted at non-protected groups (e.g. professions)	14.5	24.2	**62.9**	27.4	**74.2**	53.2	35.5	40.3	**46.8**	41.9	43.5	**59.7**	29.0	45.2	**66.1**
F25: Swaps of adjacent characters	97.7	**99.2**	90.2	82.7	**95.5**	**95.5**	72.2	**98.5**	**98.5**	78.2	**97.0**	78.9	71.4	**89.5**	80.5
F26: Missing characters	83.8	**100.0**	81.6	72.8	97.1	**98.8**	74.6	**97.7**	92.4	84.4	94.8	**98.8**	89.0	90.2	**92.5**
F27: Missing word boundaries	97.9	**100.0**	99.2	82.3	**100.0**	**100.0**	79.4	**100.0**	93.6	79.4	**90.8**	86.5	**93.6**	90.8	90.8
F28: Added spaces between chars	83.8	**100.0**	86.1	72.8	97.1	**98.8**	74.6	**97.7**	92.5	84.4	94.8	**98.8**	89.0	90.1	**92.5**
F29: Leet speak spellings	99.4	**100.0**	98.8	80.9	**99.4**	98.8	65.9	**98.2**	93.6	75.1	**84.4**	75.1	75.1	83.8	**86.7**

Table 5: Results on the HateCheck test suite.

Hate'. We evaluate all the models that have been trained for this manuscript, including the model examined in appendix A. We evaluate our trained models HATECHECK's binary form by mapping all classes positive classes to "hate" and the negative class to "not-hate".[10]

Conducting the HateCheck functional tests for the models trained on the `Comb` dataset, we see (please refer to table 5) that the federated learning models perform on par or better than the centralised models on a macro scale. The federated Bi-LSTM and FNet models yield strong improvement of 3 - 5%. On the other hand, there is a slight performance dip (0.5 - 1%) for the federated DistilBERT and RoBERTa models. Moreover, through a fine-grained analysis of model performance, we observe that all the models (centralised and federated) perform acceptable performances for different types of derogatory, pronoun reference, phrasing, spelling variations, and threatening language. However, all models perform poorly for the tests for counter speech, indicating that while the models learn to recognise some forms of hate, they cannot accurately recognise responses to it. Furthermore, we see that RoBERTa performs slightly better than all the other model variants on non-hate group identity and abuse against non-protected targets. RoBERTa and DistilBERT achieve the best performances for slurs. Overall, we find that RoBERTa and DistilBERT consistently perform well across many of the functional tests which might be due to having been pre-trained on large amount of language data. However, the pre-training also induces certain biases which limit the models' performance on profanity. The Bi-LSTMs outperform all the models on non-hateful profanity but simultaneously under-perform on hateful profanity.

6 Conclusion

Private and sensitive data can risk being exposed when developing and deploying models for hate speech detection. We therefore examine the use of Federated Learning, a privacy preserving machine learning paradigm to the task of hate speech detection to emphasise privacy in hate speech detection. We find that using Federated Learning improves on the performance levels achieved using centralised models, thus affording both privacy and performance. In future work, we intend to examine interpretability and explainability for federated learning to gain a better understanding of the causes of such performance increases.

---

[10] The `Comb` dataset uses 'none' as its negative class, the Binary Dataset (Vidgen et al., 2021) has 'Not-hate' as non-hateful label, and Multi-class Dataset Vidgen et al. (2021) has 'None' as non-hateful label

Limitations

While Federated Learning introduces increased privacy in the process of hate speech detection, a real time system may be vulnerable to attacks that can lead to privacy leakages. For instance, the weights being transferred from the clients to the server may reveal information about the local dataset to an adversary (Bhowmick et al., 2018; Melis et al., 2019). However unintended these leakages may be, they still pose a significant threat and might limit the privacy claim.

The Federated Learning models trained in our work rely on 8 of the 9 datasets used by Fortuna et al. (2021), as we could not gain access to the final dataset. We do not test the biases introduced in Federated Learning models upon combining and normalising these datasets under the schema proposed by Fortuna et al. (2021, 2020). Additionally, the dataset division for the simulation is done under the assumption of I.I.D. conditions which might not always be true for real-world scenarios.

Ethical Considerations

Although our methods for hate speech detection provide increased privacy to downstream users of content moderation technologies, i.e. users of online platforms, there are significant risks to it. First, our proposed technology has dual use implications, as it can also be applied maliciously, for instance to limit the speech of specific groups. Second, while this work uses publicly available datasets, there is an inherent tension between the public availability of data and privacy risks. Finally, although all model updates occur on local client devices, federated learning is not a silver bullet which addresses issues of systemic violence of content moderation Thylstrup and Talat (2020), or issues of privacy. Rather, federated learning can provide an avenue for engaging in meaningful conversations with people and their experiences and needs for content moderation and privacy.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Priya Basu, Tiasa Singha Roy, Rakshit Naidu, Zumrut Muftuoglu, Sahib Singh, and Fatemehsadat Mireshghallah. 2021. Benchmarking differential privacy and federated learning for bert models. *ArXiv*, abs/2106.13973.

Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. 2018. Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Kate Crawford and Tarleton Gillespie. 2016. What is a flag for? Social media reporting tools and the vocabulary of complaint. *New Media & Society*, 18(3):410–428.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the International AAAI Conference on Web and Social Media*. ArXiv: 1703.04009.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

John C. Duchi, Elad Hazan, and Yoram Singer. 2010. Adaptive subgradient methods for online learning and stochastic optimization. In *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*, pages 257–269. Omnipress.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018. Overview of the evalita 2018 task on automatic misogyny identification (ami). *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:59.

Paula Fortuna, Juan Soler, and Leo Wanner. 2020. Toxic, hateful, offensive or abusive? what are we really classifying? an empirical analysis of hate speech datasets. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6786–6794, Marseille, France. European Language Resources Association.

Paula Fortuna, Juan Soler-Company, and Leo Wanner. 2021. How well do hate speech, toxicity, abusive and offensive language classification models generalize across datasets? *Information Processing & Management*, 58(3):102524.

Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

Deep Gandhi, Jash Mehta, Nirali Parekh, Karan Waghela, Lynette D'Mello, and Zeerak Talat. 2022. A federated approach to predicting emojis in Hindi tweets. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11951–11961, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Suyu Ge, Fangzhao Wu, Chuhan Wu, Tao Qi, Yongfeng Huang, and Xing Xie. 2020. Fedner: Privacy-preserving medical named entity recognition with federated learning. *arXiv preprint arXiv:2003.09288*.

Lisa Gitelman, editor. 2013. *"Raw data" is an oxymoron*. Infrastructures series. The MIT Press, Cambridge, Massachusetts ; London, England.

Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 603–618.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Mladen Karan and Jan Šnajder. 2018. Cross-Domain Detection of Abusive Language Online. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 132–137, Brussels, Belgium. Association for Computational Linguistics.

David Kaye. 2019. *Speech police: the global struggle to govern the Internet*. Columbia Global Reports.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2022. FNet: Mixing tokens with Fourier transforms. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4296–4313, Seattle, United States. Association for Computational Linguistics.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450.

Bill Yuchen Lin, Chaoyang He, Zihang Ze, Hulin Wang, Yufen Hua, Christophe Dupuy, Rahul Gupta, Mahdi Soltanolkotabi, Xiang Ren, and Salman Avestimehr. 2022. FedNLP: Benchmarking federated learning methods for natural language processing tasks. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 157–175, Seattle, United States. Association for Computational Linguistics.

Dianbo Liu and Tim Miller. 2020. Federated pretraining and fine tuning of bert using clinical notes from multiple silos. *arXiv preprint arXiv:2002.08562*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE.

Chikashi Nobata, Joel R. Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 145–153. ACM.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on Twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45, Vancouver, BC, Canada. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, and et al. 2019. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, page 8024–8035. Curran Associates, Inc.

Pew Research Center. 2021. The State of Online Harassment. Technical report.

Aman Priyanshu and Rakshit Naidu. 2021. Fedpandemic: A cross-device federated learning approach towards elementary prognosis of diseases during a pandemic. *arXiv preprint arXiv:2104.01864*.

Swaroop Ramaswamy, Rajiv Mathews, Kanishka Rao, and Françoise Beaufays. 2019. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive federated optimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Paul Röttger, Bertie Vidgen, Dong Nguyen, Zeerak Waseem, Helen Margetts, and Janet Pierrehumbert. 2021. HateCheck: Functional tests for hate speech detection models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 41–58, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321.

Steve Durairaj Swamy, Anupam Jamatia, and Björn Gambäck. 2019a. Studying generalisability across abusive language detection datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 940–950, Hong Kong, China. Association for Computational Linguistics.

Steve Durairaj Swamy, Anupam Jamatia, and Björn Gambäck. 2019b. Studying Generalisability across Abusive Language Detection Datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 940–950, Hong Kong, China. Association for Computational Linguistics.

Zeerak Talat and Dirk Hovy. 2016. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Nanna Thylstrup and Zeerak Talat. 2020. Detecting 'Dirt' and 'Toxicity': Rethinking Content Moderation as Pollution Behaviour. *SSRN Electronic Journal*.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. 2018. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

A Learning From the Worst

Extending the experiments conducted in section 4, we aim to analyse if our claims are corroborated when we expose the complete setup of federated as well as centralised models to other datasets. We perform this analysis on the "Learning from the Worst" dataset (Vidgen et al., 2021).

A.1 Dataset

Binary Dataset We use the Dynamically Generated Hate Dataset v0.2.2 provided by Vidgen et al. (2021) which contains $41,255$ entries. We use the training, testing, and validation sets provided by Vidgen et al. (2021). This dataset consists of two categories: hate and not-hate. The category distribution is shown in table 6.

Multi-class Dataset We use the same Dynamically Generated Hate Dataset v0.2.2 provided by Vidgen et al. (2021). However, we make use of the multi-class labels provided in the original dataset. It consists of seven categories: none (i.e. not-hate), derogation, not-given, animosity, dehumanisation, threatening, and support (see table 6 for class distribution).

Binary categories	Multi-class categories	Count
Not-hate	None	18,993
Hate	Derogation	9,907
	Not Given	7,197
	Animosity	3,439
	Dehumanisation	906
	Threatening	606
	Support	207

Table 6: Label distribution of Vidgen et al. (2021) v.0.2.2.

Model	Binary Dataset			Multi-class Dataset		
	Precision	Recall	F1	Precision	Recall	F1
LogReg	63.38	52.58	54.98	56.68	35.46	40.92
Bi-LSTM	63.56	52.90	55.38	58.44	41.70	45.91
FNet	27.75	48.05	33.74	53.41	23.46	27.60
DistilBERT	71.56	71.72	68.63	78.25	52.96	59.27
RoBERTa	**76.18**	**77.50**	**74.37**	**80.26**	**62.69**	**67.91**

Table 7: Centralised model performances for binary and multi-class datasets

A.2 Analysis

We follow the training procedures outlined in section 4 on the binary and multi-class versions of the Vidgen et al. (2021) dataset and consider the results on the multi-class dataset (see Tables 7 and 8). We observe similar performance trends for the Logistic Regression and Bi-LSTM models in table 9 to those for `Comb`. This pattern extends to the Transformer-based models with the exception of the RoBERTa model. The federated RoBERTa obtains a slightly lower F1 score than the centralised version in (64.92 and 65.73, respectively).

The pattern of performances for the binary dataset varies from our main dataset. Here, we observe in Table 8 that the FNet model adapts well to the federated setting, with both FedProx and FedOpt algorithms significantly improving on their centralised counter-part (65.14 and 35.51, respectively) (see Table 9). Moreover, we find that the models optimised using FedProx algorithm outperform those using the FedOpt algorithm for all federated learning settings with the exception of the DistilBERT variant with $c = 50\%$ and $e = 5$. For the binary dataset, we observe that all federated models except for RoBERTa perform better across client fractions when trained for lower epochs. For the multi-class dataset, however, all federated models have improved performance across client fractions, when the models are trained for a higher number of epochs. We observe from our results that there is a slight performance decrease for the federated versions of the Bi-LSTM, RoBERTa, and DistilBERT, when compared to the centralised models. A small decrease in performance however is expected for federated learning, due to its emphasis on privacy protections. In spite of small differences, the experiments on both the binary and multi-class versions of Vidgen et al. (2021) closely resemble the results obtained on `Comb`, suggesting that federated learning is applicable across datasets for hate speech and class distributions.

We see a similar trend as `Comb` while performing the HATECHECK functional tests on the binary and multi-class dataset. The Logistic Regression adapts poorly across different types of counter speech, slurs, non-hate group identity, negation, and abuse against non-protected targets. Moreover, we also observe that Bi-LSTM and FNet yield poor performance for different types of negation and non-hate group identity. We find that in most cases, models trained on the binary dataset achieve higher performances than the models trained on its multi-class counterpart.

B Model Exploration

This section highlights the different model settings and hyper-parameter selection strategies used while training the models on `Comb` and Vidgen et al. (2021) in appendix A. We also provide a token level analysis conducted by on `Comb`.

B.1 Hyper-parameter Search

We use Weights and Biases (Biewald, 2020) as our experiment tracking tool for all experiments. We run a Bayesian search for finding the optimal client learning rate, server-side learning rate, and the proximal term. In our hyper-parameter search for the value of proximal term, we conduct a categorical search. Following Li et al. (2020), we set the possible values to 0.001, 0.01, 0.1, and 1.

B.2 Model Descriptions

We implement all models using PyTorch (Paszke et al., 2019) and Huggingface libraries (Wolf et al., 2020). We train the Logistic Regression and Bi-LSTM models for 300 rounds, and transformer-based models for 50 rounds. We implement early stopping based on the weighted validation F1 scores, with the patience set to 10 rounds. After conducting our hyper-parameter search, we choose our hyper-parameters (see table 11 and table 12). The measure the performances of all our models

			Binary (FedProx)			Binary (FedOpt)			Multiclass (FedProx)			Multiclass (FedOpt)		
			Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Logistic Regression	**c = 10%**	**e = 1**	59.43	59.52	59.34	57.38	57.44	57.36	45.55	39.62	41.34	45.06	37.56	40.15
		e = 5	60.13	59.99	60.01	55.93	55.79	55.76	47.03	44.71	45.58	44.93	43.91	44.25
		e = 20	60.08	59.94	59.97	56.08	55.98	55.97	47.68	48.01	47.77	46.17	46.39	46.11
	c = 30%	**e = 1**	59.78	59.69	59.71	55.88	55.74	55.71	47.28	34.94	38.20	43.26	36.66	39.07
		e = 5	60.79	60.80	60.80	55.11	55.02	55.01	48.12	47.58	47.74	44.77	42.05	43.17
		e = 20	60.76	60.41	60.41	54.93	54.80	54.76	47.27	47.94	47.50	45.26	46.44	45.51
	c = 50%	**e = 1**	60.60	60.55	60.57	54.46	54.36	54.33	46.92	34.00	37.08	42.78	36.52	38.44
		e = 5	60.76	60.73	60.74	54.58	54.52	54.51	47.54	46.68	46.96	43.60	42.26	42.85
		e = 20	60.79	60.55	60.57	54.85	54.72	54.66	47.55	48.91	48.10	44.28	44.93	44.41
Bi-LSTM	**c = 10%**	**e = 1**	61.15	61.26	61.00	61.05	61.17	60.96	38.54	38.02	35.96	40.16	38.75	38.23
		e = 5	57.63	57.55	57.57	57.93	57.88	57.89	46.09	45.68	45.84	45.88	43.57	44.34
		e = 20	58.15	58.14	58.15	58.70	58.65	58.66	45.24	47.79	45.92	45.49	49.88	45.94
	c = 30%	**e = 1**	61.72	61.74	61.13	60.26	60.37	60.13	40.23	36.64	34.89	43.64	31.18	32.74
		e = 5	58.48	58.48	58.48	59.36	59.42	59.36	45.77	44.33	44.97	46.30	43.19	44.00
		e = 20	57.07	57.06	57.06	59.37	59.35	59.36	45.06	47.07	45.58	46.68	50.00	47.19
	c = 50%	**e = 1**	60.73	60.84	60.66	60.50	60.58	60.20	45.05	32.61	34.86	43.98	33.39	35.40
		e = 5	57.90	57.93	57.91	59.51	59.55	59.52	45.59	44.88	45.22	46.85	45.15	45.77
		e = 20	57.30	57.28	57.29	59.26	59.25	59.25	45.50	48.40	46.00	46.73	50.02	47.37
FNet	**c = 10%**	**e = 1**	72.84	71.99	72.16	73.13	70.57	70.65	38.33	32.35	25.99	39.64	33.78	23.90
		e = 5	69.48	69.58	69.51	69.93	69.10	69.22	54.00	54.95	54.11	40.93	26.84	17.58
		e = 20	69.84	68.99	69.11	70.03	69.97	69.99	50.80	52.13	51.37	49.16	50.48	49.39
	c = 30%	**e = 1**	72.48	71.76	71.91	72.31	71.53	71.69	53.24	39.65	40.27	51.74	35.40	37.36
		e = 5	70.59	70.29	70.38	71.47	70.72	70.87	53.02	52.47	51.62	43.53	28.59	22.33
		e = 20	69.06	68.93	68.98	69.96	69.93	69.94	51.03	52.98	51.43	49.66	52.62	49.74
	c = 50%	**e = 1**	72.64	72.47	72.54	72.61	72.47	72.53	58.49	40.30	43.27	49.48	32.74	33.23
		e = 5	69.67	69.28	69.39	70.19	69.92	70.00	54.31	53.25	53.54	50.44	50.07	49.82
		e = 20	67.71	67.62	67.66	68.21	68.14	68.17	51.62	51.75	51.55	48.10	50.00	48.27
DistilBERT	**c = 10%**	**e = 1**	74.62	74.76	74.67	74.20	74.43	74.21	58.99	42.73	45.33	63.91	49.15	52.88
		e = 5	73.74	73.05	73.21	72.35	72.17	72.24	60.72	58.22	59.27	60.41	57.57	58.71
		e = 20	72.92	72.39	72.53	70.77	70.43	70.53	59.61	59.14	59.24	58.78	59.38	58.95
	c = 30%	**e = 1**	74.90	74.77	74.82	73.73	73.84	73.78	58.95	43.48	45.98	54.13	40.11	39.49
		e = 5	73.34	73.14	73.22	70.98	70.89	70.93	60.66	58.46	59.39	60.73	58.97	59.72
		e = 20	73.02	72.82	72.90	70.37	70.17	70.24	58.79	58.85	58.71	58.22	58.80	58.28
	c = 50%	**e = 1**	74.94	74.88	74.91	73.60	73.72	73.64	59.74	43.91	46.44	55.35	40.25	41.26
		e = 5	73.35	73.02	73.13	70.51	70.23	70.32	60.52	58.39	59.26	60.35	59.26	59.71
		e = 20	73.07	72.67	72.79	70.08	69.84	69.92	58.85	58.39	58.54	58.41	58.60	58.36
RoBERTa	**c = 10%**	**e = 1**	80.72	80.90	80.78	80.74	81.01	80.46	62.50	48.67	51.76	66.26	54.58	58.04
		e = 5	80.97	80.94	80.95	80.20	80.38	80.27	65.33	64.86	64.92	63.97	62.20	62.93
		e = 20	81.71	81.82	81.76	80.34	80.40	80.37	63.74	64.28	63.85	63.68	63.80	63.65
	c = 30%	**e = 1**	81.61	81.76	81.67	80.79	80.98	80.86	64.77	47.26	50.49	65.87	58.10	60.41
		e = 5	81.27	81.33	81.30	79.92	80.10	79.98	65.27	64.31	64.64	63.58	63.19	63.22
		e = 20	81.22	81.37	81.28	79.17	79.32	79.23	64.59	64.82	64.54	63.14	63.16	62.96
	c = 50%	**e = 1**	80.82	80.81	80.82	80.84	81.09	80.91	64.81	45.97	49.10	66.27	58.58	61.14
		e = 5	81.76	81.86	81.80	79.66	79.75	79.70	65.19	64.50	64.68	64.13	63.77	63.82
		e = 20	81.32	81.41	81.36	79.85	79.94	79.89	64.83	64.62	64.56	63.54	63.19	63.29

Table 8: Results of binary and multi-class classification experiments run on the datasets released by Vidgen et al. (2021) using the Federated Learning setup (FedProx and FedOpt). c is the percentage of clients whose updates are considered. e is the number of local epochs on edge device.

Model	**Binary Dataset**		**Multi-class Dataset**	
	Server trained	Federated	Server trained	Federated
LogReg	59.61	60.74	39.79	48.10
Bi-LSTM	61.98	61.13	42.18	47.37
FNet	35.51	72.54	45.52	54.11
DistilBERT	74.95	74.91	59.76	59.72
RoBERTa	80.64	81.80	65.73	64.92

Table 9: Results comparing the F1 scores for server-based approaches and federated approaches for binary and multi-class datasets proposed by Vidgen et al. (2021)

using precision, recall and weighted F1 scores.

B.3 Token Level Analysis

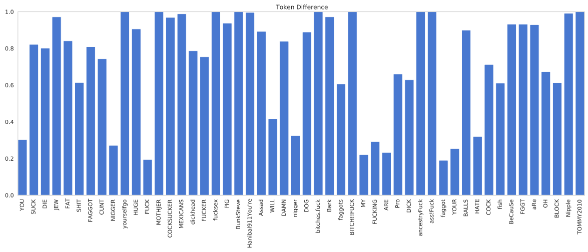


Figure 2: fraction of token occurrences (of 50 most frequent tokens) in the discarded data

Table 10 shows the preliminary explorations of the token-level distributions for the combined dataset (Section 3) with two tokenisation methods: word-

Dataset	Tokenization	Minimum	99%ile	Maximum
Talat and Hovy (2016)	Word-level	1	34	54
	Subword-level	3	60	101
Davidson et al. (2017)	Word-level	1	37	94
	Subword-level	2	83	412
Fersini et al. (2018)	Word-level	2	36	47
	Subword-level	3	64	93
de Gibert et al. (2018)	Word-level	1	67	374
	Subword-level	1	93	592
Swamy et al. (2019a)	Word-level	1	175	1481
	Subword-level	1	233	3209
Basile et al. (2019)	Word-level	1	59	74
	Subword-level	3	105	156
Zampieri et al. (2019)	Word-level	2	69	112
	Subword-level	4	152	221
Kaggle	Word-level	1	727	4950
	Subword-level	2	872	4952

Table 10: Word-level and subword-level (BPE) token sequence length distribution for `Comb` dataset described in Section 3

level using SpaCy (Honnibal and Montani, 2017) and subword-level using the BPE algorithm.[11] Based on our analysis, we draw the following conclusions: 1) token length is highly imbalanced for different datasets in `Comb`, particularly in Kaggle dataset[4]; 2) 99th percentile token length in Kaggle dataset[4] is reflected in the remaining dataset. Considering this, we remove longest 1% of documents from the Kaggle dataset[4] to achieve faster computation. Through this exclusion process, the maximal token length of documents is reduced from 4950 to 727 tokens, without a substantial loss of information.

---

[11] https://github.com/VKCOM/YouTokenToMe

Model	**Combined Datasets**			**Binary Dataset**			**Multi-class Dataset**		
	bs	client_lr	μ	bs	client_lr	μ	bs	client_lr	μ
LogReg	128	0.01	0.01	64	0.01	0.01	64	0.01	0.01
Bi-LSTM	128	0.001	0.01	64	0.001	0.01	64	0.001	0.01
FNet	32	0.0001	0.1	32	0.0001	0.001	32	0.0001	0.001
DistilBERT	32	0.00004	0.01	32	0.00002	0.01	32	0.00002	0.01
RoBERTa	16	0.00002	0.01	24	0.00002	0.01	32	0.00002	0.01

Table 11: Model hyper-parameters for server-based and federated models for the Vidgen et al. (2021). 'bs' represents batch size, 'client_lr' represents client learning rate, μ represents proximal term for FedProx algorithm.

Model	**Combined Datasets**			**Binary Dataset**			**Multi-class Dataset**		
	bs	client_lr	server_lr	bs	client_lr	server_lr	bs	client_lr	server_lr
LogReg	128	0.01	0.01	64	0.01	0.001	64	0.01	0.01
Bi-LSTM	128	0.001	0.01	64	0.001	0.001	64	0.001	0.001
FNet	32	0.0001	0.001	32	0.0001	0.0001	32	0.0001	0.0001
DistilBERT	32	0.00004	0.001	32	0.00002	0.0001	32	0.00002	0.0001
RoBERTa	16	0.00002	0.001	24	0.00002	0.0001	32	0.00002	0.0001

Table 12: Model hyper-parameters for server-based and federated models for the Vidgen et al. (2021). 'bs' represents batch size, 'client_lr' represents client learning rate, 'server_lr' represents server learning rate for FedOpt algorithm.

Learning the Legibility of Visual Text Perturbations

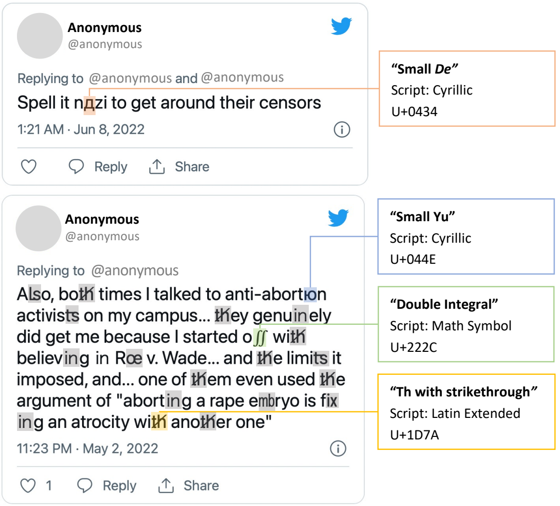
Dev Seth[1] **Rickard Stureborg**[1] **Danish Pruthi**[*] **Bhuwan Dhingra**[1]
[1] Duke University
{ds447, rs541, bd149}@duke.edu

Abstract

Many adversarial attacks in NLP perturb inputs to produce visually similar strings (‘ergo’ → ‘ɛrgo’) which are legible to humans but degrade model performance. Although preserving legibility is a necessary condition for text perturbation, little work has been done to systematically characterize it; instead, legibility is typically loosely enforced via intuitions around the nature and extent of perturbations. Particularly, it is unclear to what extent can inputs be perturbed while preserving legibility, or how to quantify the legibility of a perturbed string. In this work, we address this gap by learning models that predict the legibility of a perturbed string, and rank candidate perturbations based on their legibility. To do so, we collect and release **`LEGIT`**, a human-annotated dataset comprising the legibility of visually perturbed text. Using this dataset, we build both text- and vision-based models which achieve up to 0.91 F1 score in predicting whether an input is legible, and an accuracy of 0.86 in predicting which of two given perturbations is more legible. Additionally, we discover that legible perturbations from the **`LEGIT`** dataset are more effective at lowering the performance of NLP models than best-known attack strategies, suggesting that current models may be vulnerable to a broad range of perturbations beyond what is captured by existing visual attacks.[1]

1 Introduction

To manage the increasing demand for content moderation—e.g., detecting spam or toxic/hateful content on online platforms—organizations have turned to machine learning solutions. In response, users often resort to manipulating text to evade detection, removal, or search. For instance, hateful comments often comprise of visually similar characters to avoid automatic filtering (Le et al., 2022).



Spell it nдzi to get around their censors

Also, boᵺ times I talked to anti-abortюn activiʦ on my campus... ᵺey genuinely did get me because I started o∬ wiᵺ believing in Rœ v. Wade... and ᵺe limiʦ it imposed, and... one of ᵺem even used ᵺe argument of "aborting a rape embryo is fix ing an atrocity wiᵺ anoᵺer one"

Figure 1: Visual attacks in the wild. Examples of Twitter users manipulating their tweets to evade the platform’s ‘sensitive content’ detection algorithms.

Since people read text visually, the manipulated content can still be easily understood and harm its target audience. These attacks started with simple ASCII substitutions like he11o (colloquially referred to as “leetspeak"), but have evolved into complex manipulations utilizing characters from different Unicode scripts (Flamand, 2008; Raymond, 1996). Figure 1 shows two such examples.

Unlike computer vision where there is an established notion of what constitutes an imperceptible perturbation (typically defined via the ℓ_∞ distance), most perturbations in text are perceptible. However, as long as the perceptible manipulations remain *legible*, the message could have its intended effect. The legibility of a text is determined by whether or not a literate person can decipher the altered words. The degree to which a piece of text can be perturbed, while maintaining legibility, depends on a multitude of factors such as its context, similarity to the original content, the positions of the perturbations, the background knowledge of the reader, etc.

---

[*] Work done while at Carnegie Mellon University.

[1] Data, code, and models are available at `https://github.com/dvsth/learning-legibility-2023`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3260–3273
May 2-6, 2023 ©2023 Association for Computational Linguistics

However, many adversarial attacks enforce legibility only loosely based on intuitions about the nature of the attacks, e.g., that changing 1-2 characters in a sentence does not impact its legibility (Belinkov and Bisk, 2018; Pruthi et al., 2019).

In this work, we instead propose to learn the legibility of visual perturbations, by developing text- and vision-based models trained on legibility annotations from human subjects. The current focus of research on adversarial attacks is to find minimal perturbations required to break NLP models, and several recent findings suggest that models remain brittle to such perturbations (Eger et al., 2019; Dionysiou and Athanasopoulos, 2021; Pruthi et al., 2019). In contrast, our work attempts to uncover the space of *all legible perturbations* that we need to defend against. Towards our goal of characterizing the limits of legibility of perturbed texts, we make the following contributions:

First, we crowdsource human judgments about the legibility of different perturbations: specifically, we show annotators two perturbed versions of the same word and ask them which one, if any, they find more legible. Our perturbation strategy considers substituting letters in the word with Unicode characters drawn from a large subset of the Basic Multilingual Plane covering over 100 scripts from around the world.[2] In total, we collect $30,320$ annotations, one each for $14,643$ and $3,332$ instances in the training and validation sets, respectively, and three each for the $4,113$ instances in the test set. Using these preferences, we define a *pairwise legibility ranking task* as well as a *binary legibility classification task*. While the former allows making inferences about which candidate perturbation is *most* legible, the latter allows filtering out illegible perturbations altogether. For each task, we identify a *hard* subset of the collected data, which includes fine-grained comparisons expected to be more challenging for annotators and models alike.

Second, we use the labeled data to train models which predict the degree of legibility of a perturbed text. Specifically, we fine-tune pretrained vision (TrOCR; Li et al., 2021) and text-based (ByT5; Xue et al., 2022) models on the ranking and classification tasks. We find that TrOCR trained in a multi-task setup on both tasks achieves the best performance with 0.91 F1 score on the classification task and 0.86 accuracy on the ranking task. Interestingly, we find that the purely text-based ByT5 also achieves competitive performance on the classification task with 0.89 F1, suggesting that its pretrained byte representations encode aspects of visual similarity between Unicode characters. Further, we find that models have high F1 scores on the subset of data with high inter-annotator agreement: TrOCR achieves a 0.96 F1 score on test cases where all three annotators agree. We also note that legibility is a complex phenomenon—it doesn't correlate trivially with the distance of the perturbation from the original text or the number of letters substituted.[3]

Third, we consider a word-level *perturbation recovery* task, which involves inferring the original word from its perturbed version. We evaluate GPT-3 (Brown et al., 2020) on this task, comparing its performance on legible perturbations from our perturbation strategy versus those generated by VIPER, a *VIsual PERturber* method proposed by Eger et al. (2019). We find that GPT-3 has a lower accuracy in recovering perturbations from our perturbation strategy, despite VIPER providing no guarantees on legibility. Additionally, we apply our findings to the important task of toxicity classification. We perturb a subset of the dataset using our perturbation strategy and find that it degrades the SOTA Detoxify (Hanu and Unitary team, 2020) classifier more than existing VIPER attacks. These findings demonstrate that existing attacks do not comprehensively cover the space of legible perturbations that can degrade model performance.

2 Related Work

Adversarial Attacks for NLP. A challenge in defining adversarial examples for text lies in characterizing the space of *equivalent* inputs to a training or test example which preserves the target label. While early work focused on adding distracting text to fool question answering systems (Jia and Liang, 2017), recent work utilizes more general strategies applicable to many tasks (Li et al., 2019; Morris et al., 2020; Jin et al., 2020). Many of these can be categorized as *word*-level synonym substitutions (Alzantot et al., 2018; Garg and Ramakrishnan, 2020; Li et al., 2020), or *character*-level legibility-preserving substitutions (Ebrahimi et al., 2018; Pruthi et al., 2019). Most attacks in either category are perceptible in that readers of

---

[2] We consider $12,287$ Unicode characters from codepoints 0x0000 to 0x2fff.

[3] A logistic regression model using these as features only agrees 56.7% of the time with authors' legibility assessment.

the text can identify that it has been transformed, except for one notable exception where invisible characters and near-identical characters are used to render strings indistinguishable from the original (Boucher et al., 2022). Attacks based on visual similarity of characters have also been previously considered by Eger et al. (2019) who propose three attack strategies: ICES (based on rendered glyph similarity), DCES (based on bag-of-words textual similarity of Unicode codepoint descriptions), and ECES (based on adding diacritics to base characters). For ICES, they compute similarity by comparing raw pixel values of the renderings, which we improve upon here by utilizing a pretrained Optical Character Recognition (OCR) model. This produces a 'smarter' set of visual neighbors: e.g., mirror images of letters, scaled versions of letters (like O vs °) etc., which go beyond simple accents or modifiers. We also report in-depth comparisons between our perturbation strategy and the ECES and DCES approaches in section 5.

Legibility of Perturbed Inputs. Among character-level perturbation attacks, legibility has only been loosely enforced based on intuitions about the nature and the degree of manipulations. This often results in conservative substitutions which only represent a lower bound on the space of all legible perturbations. For instance, Pruthi et al., 2019 limit the attack to only 1-2 character changes (e.g., substitutions, deletions or additions) per input example; similarly, Ebrahimi et al., 2018 propose an attack strategy which specifically minimizes the number of character manipulations required in order to render the output legible. Attacks based on visual similarity usually constrain their attack surface to inputs which are above a threshold similarity (in pixel or embedding space) to the original input (Eger et al., 2019; Eger and Benz, 2020; Dionysiou and Athanasopoulos, 2021). In this work, by contrast, we directly address the question of what constitutes legible perturbations, with the aim of learning a grounded definition of legibility rather than assuming one *a priori*.

3 Legibility Tests

We adopt a supervised learning approach for determining the legibility of perturbed texts. In this section, we describe the process used for collecting the **LEGIT** dataset (which stands for **LEGI**bility **T**ests) and in the next section we describe the modeling techniques used for predicting the legibility score and ranking different candidate perturbations.

Our setting involves one-to-one character substitutions at the word level, i.e., given a word (and no other context), we consider perturbations where each letter in the word may be replaced by a Unicode codepoint in 0x0000-0x2fff. Moreover, the substitutions are mutually independent and do not depend on the context of the other letters.

3.1 Perturbation Process

To generate perturbations for the data labeling task, we replace a subset of characters in a word with visually similar counterparts. Specifically, given a word w, we first randomly select a fraction $n \in [0, 1]$ of characters in that word to corrupt. Then, each of the chosen characters is replaced by its nearest neighbor at rank k in the embedding space generated by a model $\mathcal{M}$ which encodes characters into visual features. Hence, there are three parameters involved in the perturbation process $\phi = \{n, k, \mathcal{M}\}$.

We experiment with several models to encode characters into visual features, all based on renderings of the Unicode codepoints into images. To keep visual representations consistent across models, we use GNU Unifont, rendering each glyph separately in 144px font size with black color, on a 224×224px white background.[4] Given the rendering, we compare 5 models to encode the features. Three are transformer-based: TrOCR ('base') (Li et al., 2021), CLIP ('vit-base-patch32') (Radford et al., 2021), and BEiT ('base-patch16-224-pt22k-ft22k'). One employs convolutional as well as transformer networks: DETR (Carion et al., 2020). The fifth model is a simple baseline: ImgDot, which uses the (flattened) bitmap of a rendered character as its embedding vector. In preliminary experiments, 400 perturbed pairs were generated, with each pair using the same settings for k, n but using different models. The authors then independently ranked perturbations each pair based on their legibility. DETR- and BEiT-generated perturbations were ranked above other models' perturbations 23% and 41% of the time, respectively, whereas CLIP and ImgDot perturbations were preferred over others in 66% and 73% of cases. Hence, DETR and BEiT were excluded from further experiments. TrOCR was included later, after verifying that it was preferred $\approx 50\%$ of the time against both CLIP and ImgDot.

---

[4] Glyphs were rendered by the Pillow library (Clark, 2015).

For each of the chosen models, we compute the pairwise cosine distances between the model's embedding vectors for all Unicode codepoints in the range 0x0000–0x2fff (excluding invalid or empty codepoints), and use these distances to find the nearest-neighbors for each character. Then, to perturb a given word w using the parameters $\phi = \{k, n, \mathcal{M}\}$, we first pick $\lfloor n|w| \rfloor$ characters uniformly at random to replace. For each character, we fetch its k-th nearest neighbor from the model $\mathcal{M}$. Finally, we apply these substitutions to the target word to obtain the perturbed word.

3.2 Pairwise Comparisons

We crowdsource legibility annotations for the perturbed words using Amazon's Mechanical Turk. We collect annotations on both absolute legibility as well as relative preference between two differently perturbed inputs. Since annotators tend to produce higher quality annotations when comparing items rather than assigning absolute values (Callison-Burch et al., 2007; Liang et al., 2020), we design an annotation interface based on *pairwise comparisons* of two perturbed versions of the same word (Appendix A). Specifically, annotators see perturbations w_1, w_2 side-by-side, with the original word w hidden. They are asked to indicate which perturbation they find more legible by selecting exactly one of these four labels:

L1: w_1 is preferred

L2: w_2 is preferred

BL: both w_1, w_2 are equally legible

NL: neither w_1 nor w_2 is legible

L_1 and L_2 capture not only relative preferences between the two perturbations (used for the ranking task), but also indicate that the preferred perturbation is legible. However, these labels do not give us any information about the non-preferred perturbation. On the other hand, the BL (Both Legible) and NL (Neither Legible) options do not give us a ranking between the two words, but inform us about the legibility (or illegibility) of both words. In the next section, we use these labels to derive datasets for both a pairwise ranking task and a binary classification task.

We generate the data for annotation from English words consisting of the top $10,000$ frequent words (as per Kaufman (2012)) in the Trillion Word Corpus (Brants and Franz, 2006). We filter this vocabulary to remove words with lengths less than 4 or greater than 14, ending up with 7600 words. These words are randomly split into the train (65%), validation (15%), and test (20%) sets; all future perturbation pairs (w_1, w_2) generated for word w are added to the corresponding set, and the same sets are used for all experiments. To perturb a word w into the pair w_1, w_2 a model $\mathcal{M}$ is picked at random from {TROCR, CLIP, IMGDOT } (the three best models from our initial perturbation analysis). We sample $k \sim \mathcal{N}(\mu_k, \sigma_k^2)$ and similarly for n, applying the appropriate bounds to keep $k > 0$ and $n \in [0, 1]$. The initial values are $\mu_k = 25, \sigma_k^2 = 10, \mu_n = 0.5, \sigma_n^2 = 0.2$.

3.3 Adaptive Annotations

The space of all possible perturbations of a word is vast, and sampling the parameters ϕ based on the priors above is unlikely to yield difficult perturbations which lie at the boundary of legibility. In order to identify such perturbations, we collect data over multiple rounds using an *adaptive* process for generating the pairs. In the first round, the pairs are generated as described above and annotated by the crowd-workers. In the following rounds, pairs are generated taking into account the last round of annotations. Specifically, the ϕ_1, ϕ_2 for each successive round are chosen to make the next round of labeling harder for annotators. This is accomplished by manipulating the Gaussian used to generate k, n, i.e. by shifting μ_1, μ_2 to be closer to each other and reducing variance. This approach generates perturbations which elicit more nuanced comparisons from annotators, allowing us to capture fine-grained legibility preferences in the dataset.

Inter-annotator Agreement. Three waves of annotations were collected using adaptive pair generation. To establish high quality and confidence in the test set labels, three annotations were collected for each pair of perturbations in the test set. Pairs where all annotators disagreed were removed from the test set. For 49.1% of pairs, all 3 annotators agree on the same label, 43.6% of pairs have agreement between 2 out of the 3 annotators, and only 7.3% of pairs have no agreement among annotators. Hence, even with 4 labels to choose from, for 92.7% of (w_1, w_2) pairs, at least two out of three annotators chose the same label. This suggests that the task is well-defined and has low variance.

	# pairs (w_1, w_2)	# distinct (w)	classification examples	ranking examples
Train	14622	4940	20217	9027
Val	3326	1140	4639	2013
Test	3712	1520	4774	2650
Total	21660	7600	29630	13690

Table 1: `LEGIT` dataset statistics. For each word, there exist multiple perturbed pairs, generated through three rounds of adaptive annotations.

Annotator Details. We recruit 150 annotators, all of whom had over a 95% acceptance rate for previous work done on the platform, as well as a history of over $1,000$ completed tasks.[5] Annotators are given occasional quality checks, wherein they annotate pairs drawn from a gold dataset labeled by the authors; annotators with less than 70% accuracy on the gold data were removed from the study and their annotations discarded from the final dataset. Annotators are given batches of 20 (w_1, w_2) pairs at a time; typically taking between $30-45$ seconds to annotate. The average compensation per batch is \$0.12. Further details of the annotation interface and instructions are available in Appendix A.

Hard Subsets. We identify challenging subsets of the collected data for the ranking and classification tasks. For ranking, the chosen subset ($N = 1052$) contains pairs (w_1, w_2) where $\frac{(n_1 - n_2)^2}{n_1 n_2} < 0.1$, i.e., both n's are close to each other, so it is hard in the sense that the perturbations have similar parameters ϕ but varying degrees of legibility—they cannot be ranked just by comparing metadata. For the classification task, the chosen subset ($N = 2626$) consists of all perturbations w_i with $n_i > 0.4$, making the task more challenging by excluding lightly-perturbed words which are easier to classify.

4 Tasks and Models

In this section, we start by introducing two tasks for characterizing the legibility of perturbed texts, followed by a number of models for solving them.

4.1 Tasks

From the labels collected in the previous section, we derive data for the two tasks: ranking and classification. The tasks assume that the original word w is known, as we base our setup considering an attacker who is trying to find the best perturbation.

Ranking Task. Given a pair (w_1, w_2) of perturbations and the original word w as input, rank the perturbations in order of legibility. For this task, we only consider the subset of data labeled with strict rankings—i.e., excluding pairs labeled BL (Both Legible) and NL (Neither Legible). As the data is balanced, we only report accuracy as the main metric for this task.

Classification Task. Given a single perturbation w_i and the original word w, decide whether the perturbation is legible. While annotators performed pairwise comparison between (w_1, w_2), we can infer the binary legibility labels for w_i from pairwise rankings as follows: for labels BL and NL, we can make the obvious inference of *legible* and *illegible* for both w_i. For labels L_i, we can again infer that w_i is legible, but cannot say anything about $w_{j \neq i}$; all such w_j with unknown legibility are excluded from the classification task dataset. Since there are more legible than illegible instances in the data, we report both accuracy and F1 scores on this task.

4.2 Baselines

Majority Class. This baseline always predicts the majority class from the training set for every test example. For the ranking task, it always predicts w_2 as the preferred perturbation (resulting in an accuracy of 0.5), and for classification, the majority class is 'legible' (yielding 0.677 accuracy).

Logistic Regression using ϕ. Note that in an attack setting, the attacker would know the perturbation parameters ϕ exactly and may be interested in predicting the legibility of their perturbation using these parameters. Hence, we perform logistic regression directly on the attack parameters (n, k) to predict the label. Being a simple metadata-only baseline, this model does not take into account the characters that were perturbed or their position.

4.3 Text-based Models

ByT5. Legibility, as defined in this paper, is a *visual* property. However, we might expect pretrained language representations (e.g., those learned by large-scale language models) to also encode visual similarity between characters since the web-corpora used for pretraining might include similar-looking characters in the same contexts

---

[5] 686 annotators were excluded due to failing their first quality check. Many attempts were observed to be spam.

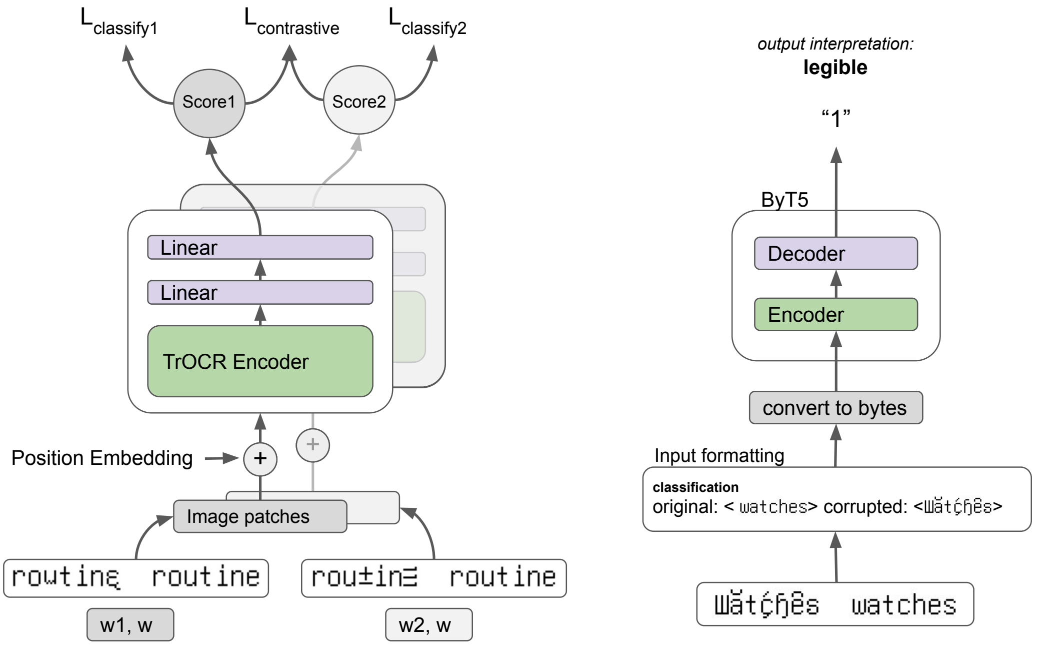


Figure 2: Comparing ByT5 and TrOCR training setup. ByT5: Both the perturbed and original words are given as one input to the model. TrOCR: Both w_1, w_2 are fed sequentially into the same TrOCR-based model, and the two resulting scalar outputs are used to compute the loss. For each perturbation, the string "w_i w" is rendered and used as input for the model.

(e.g., '0' instead of 'O'). To test this, we experiment with ByT5 (Xue et al., 2022), a multilingual encoder-decoder language model which tokenizes inputs into byte sequences. Byte-level tokenization ensures that none of the perturbations in `LEGIT` are out-of-vocabulary, and multilingual pretraining ensures that the model has seen a large subset of Unicode. We finetune the pretrained ByT5-models ('small' and 'base') to predict the binary labels for both classification and ranking in a text-in text-out setting. For ranking, the inputs are formatted as: "original: $< w >$ word0: $< w_0 >$ word1: $< w_1 >$", and the output is "0" or "1" depending on which word is more legible. For classification, the inputs are formatted as: "original: $< w >$ corrupted: $< w_i >$ ", and the output is "0" or "1" depending on whether the corruption is illegible or legible. We train two separate models starting from the pretrained ByT5 weights using the cross-entropy loss over the target byte-sequence and AdamW optimizer (Loshchilov and Hutter, 2019) and perform early stopping using the validation set. Figure 2 outlines the model schematic with sample inputs and outputs.

4.4 Vision-based Models

Since we are concerned with finding representation spaces for *visually* similar characters, vision-based models are a natural choice for the task. We consider both unsupervised models which rely on pixel-based or embedding-based similarities, as well as supervised models based on OCR, which we train on the `LEGIT` data.

ImgDot. This unsupervised approach compares the corresponding characters in w and w_i based on the cosine distance between their pixel renderings. For the ranking task, this model selects the perturbation whose average cosine distance with the uncorrupted word is lower. For classification, we tune a threshold similarity parameter on the training set, above which the model predicts 'legible'.

TrOCR-Embeddings. This approach is identical to ImgDot, except that we use the pretrained character embeddings obtained by passing the rendered images as input to the TrOCR model. The embedding vector for each character is obtained by averaging the last hidden state from the encoder output (bypassing the pooler). Note that the accuracy of these unsupervised baselines gives us an idea of how well the corresponding representations align with human notions of legibility.

TrOCR. Finally, we consider finetuning TrOCR on the `LEGIT` data. We only use the encoder part of the TrOCR base model and connect it to a linear head. This linear head has two fully connected layers mapping inputs of size 768 (which is equal to the dimension of the encoder output) to a scalar output which represents the legibility score of the perturbed input. We use ReLU activations between the linear layers and apply dropout. The model takes variable-sized images as input; this is created by rendering a pair (w_i, w) into a single image by concatenating both strings along the horizontal axis (see Figure 2).

For the classification task, the output score from the model is used directly for predicting the label. Given a pair (w, w_i), let s_i denote the scalar output from the model and let $y_i \in \{0, 1\}$ denote the legibility label (where 1 denotes that w_i is legible). Then the classification loss is given by:

$$\begin{aligned}\mathcal{L}_{\text{classify-}i} = &-y_i \log \sigma(s_i) \\ &- (1 - y_i) \log\left[1 - \sigma(s_i)\right] \qquad (1)\end{aligned}$$

where σ is the sigmoid function. We apply the same loss function to both perturbations w_1 and w_2. We denote this classification model as TrOCR-C.

For the ranking task, we use the same model but apply it separately to the pairs (w, w_1) and (w, w_2) to obtain the scores s_1 and s_2. The parameters across the two applications of the model are shared in a Siamese network setup (Koch et al., 2015). Given these two scores, and the label $y \in \{0, 1\}$ (where 0 denotes that w_1 is more legible), we define the ranking loss as:

$$\begin{aligned}\mathcal{L}_{\text{contrastive}} = &-y \log \sigma(s_1 - s_2) \\ &- (1 - y) \log\left[1 - \sigma(s_1 - s_2)\right] \qquad (2)\end{aligned}$$

The above loss encourages s_1 to be higher than s_2 when $y = 0$ and vice versa. A similar loss has been used to train summarization models from pairwise human preferences (Stiennon et al., 2020). We denote this ranking model as TrOCR-R.

The Siamese setup for the ranking task is limited in the sense that it cannot directly compare the two perturbations to decide which is more legible. However, our goal is to train the model to produce a calibrated legibility score given only a single perturbation as the input. Further, the Siamese network allows us to train the model on *both* the classification and ranking tasks together in a multi-task fashion:

$$\mathcal{L} = \mathcal{L}_{\text{classify-1}} + \mathcal{L}_{\text{classify-2}} + \mathcal{L}_{\text{contrastive}} \qquad (3)$$

The loss terms for each training example are masked based on the label: the ranking loss is masked out if the label is "equally legible" or "both unclear", whereas the individual classify-i loss is masked out if the inferred binary legibility of perturbation w_i is indeterminate (e.g. for label L_1, binary legibility of w_2 is unknown). Together, these losses ensure that the legibility score s_i is thresholded at 0, above which the perturbations are legible, and more legible inputs receive a higher score. We denote the model using combined loss as TrOCR-MT.

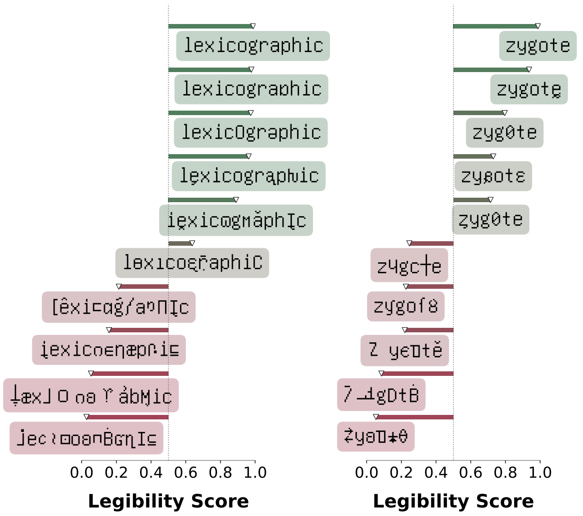


Figure 3: Legibility scores for `LEGIT`-generated perturbations of *lexicographic* and *zygote* from the TrOCR-MT model. Neither word was seen during training.

5 Results

Table 2 shows the performance of all models introduced on both the classification and ranking tasks.

Classification Task. For the classification task, we find that baselines that just use the metadata perform poorly. The Majority Class baseline obtains an F1 score of 0.677, and the Logistic Regression model using ϕ parameters yields an F1 score of 0.665, implying that legibility is *not* a simple function of the perturbation parameters k, n. The unsupervised vision-based models, ImgDot and TrOCR embeddings, vastly improve upon the simple baselines, with the TrOCR embeddings obtaining an F1 score 0.868 and ImgDot yielding an F1 score of 0.845. Hence, these embeddings align reasonably well with human perceptions of legibility. The text-based ByT5 models improve significantly over the baselines and unsupervised vision-based models. They are comparable to the performance of the single-task objective TrOCR-C, but worse than the TrOCR-MT. This suggests that the ByT5 models might have encountered some visual perturbations during pretraining. Comparing the

	Model	Classification				Ranking
		Accuracy	**Precision**	**Recall**	**F1** Std/Hard	**Accuracy** Std/Hard
Baselines	Majority Class	.512	.512	1.000	.677/.000	.500/.502
	Log. Regression	.680	.659	.671	.665/.256	.744/.642
Vision-based	ImgDot	.788	.861	.828	.845/.583	.790/.652
	TrOCR embeds	.825	.868	.883	.868/.654	.781/.677
	TrOCR-C	.840	.881	.891	.886/ –	–
	TrOCR-R	–	–	–	–	.835/ –
	TrOCR-MT	**.868**	**.914**	.895	**.905**/**.726**	**.858**/**.757**
Text-based	ByT5-small	.844	.872	.909	.890/ –	.762/ –
	ByT5-base	.842	.868	**.912**	.889/ –	.769/ –

Table 2: Results on the standard test set. The TrOCR-MT model, trained in the multi-task setting, outperforms all other models for F1 score on both tasks. The trained models also outperform the baselines on both tasks.

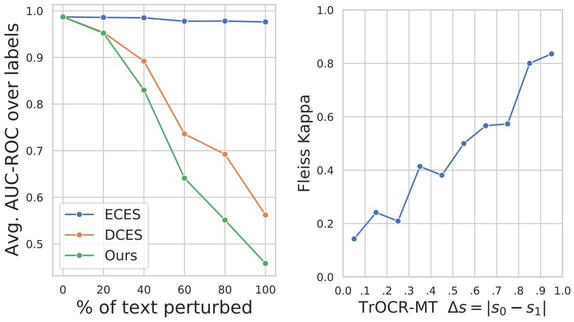


Figure 4: **Left**: *Detoxify* model performance on perturbations generated by different attack methods on a random subset ($N = 2000$) of the Jigsaw Toxic Comment Classification dataset. Model performance degrades most on our perturbations. **Right**: Model confidence on legibility is aligned with annotator agreement. Legibility scores (s_0, s_1) were obtained using TrOCR-MT for each perturbed pair (w_0, w_1) in the test set. Pairs were grouped by the score difference $\Delta s = |s_0 - s_1|$ and Fleiss' κ was computed for each group.

single-task TrOCR-C model with the multi-task TrOCR-MT, we find that the presence of the additional ranking loss term during training improves model performance on the classification task from 0.886 to 0.905. On test examples where all 3 annotators agree, TrOCR performs even better, attaining an F1 score of 0.960, compared to a score of 0.850 on examples where only 2 annotators agree. As further evidence of the model's alignment with annotators, we find that the model confidence is directly correlated with annotator agreement (cf. Figure 4) as measured by Fleiss' κ (Fleiss, 1971). Furthermore, consider Figure 3, which shows legibility scores obtained from TrOCR-MT for two words picked at random which are not part of the training set. Qualitatively, we see that legibility scores from the TrOCR-MT model aligns with the human judgements of legibility for these words.

Ranking Task. The TrOCR-MT model performs better *relative to* other models, resulting in a 6.8% absolute accuracy improvement. Akin to the classification task, we find the TrOCR-MT model outperforms its single-task counterpart TrOCR-R. Thus, training with a multi-task objective improves performance on both ranking and classification tasks when compared to single-objective models. Differently from classification, we find that ByT5 is significantly worse than the vision-based models on ranking, suggesting that language model pretraining is effective at separating legible from illegible perturbations, but not at encoding the degree of legibility of legible perturbations.

Jigsaw Challenge. Next, we check whether perturbations generated by our attack model (§ 3.1) and filtered to ensure legibility using TrOCR-MT are effective at degrading the performance of NLP models. We employ the Jigsaw Toxic Comment Classification Dataset, which is a multilabel classification dataset consisting of Wikipedia comments and human-annotated binary labels for 6 toxicity categories. In Figure 4 (Left), we compare **`LEGIT`** and VIPER-DCES strategies in a real-world scenario by perturbing the Jigsaw dataset with each strategy and reporting how much these perturbations degrade the performance of `Detoxify-original` (Hanu and Unitary team, 2020), a BERT-based model which has state-of-the-art performance on the Jigsaw dataset. We show that **`LEGIT`** produces greater degradation at lower n, and produces more legible perturbations even at higher n (due to TrOCR-MT filtering). In comparison, we find that DCES perturbations become very hard to read at higher n, diluting the significance of the DCES results at high n. Appendix D provides a qualitative analysis of the legibility of DCES perturbations compared to those generated using our **LEGIT** method.

VIPER-ECES causes a negligible degradation on model performance, which is due to the fact that the BERT tokenizer “corrects" almost all of the simple diacritic-based ECES character substitutions. This means that the classification model receives mostly unperturbed input save for some isolated UNKs. For example, the perturbed input Ťĥâňǩ ŷôǔ is tokenized back into `Thank  you`. Taken together, these results demonstrate that **LEGIT** exploits a more *efficient* legibility space, finding character substitutions which have a greater impact on model performance while preserving legibility.

Perturbing GPT-3. The strong performance of ByT5 at separating legible from illegible inputs suggests that language models might be somewhat robust to such perturbations. To examine this, we experiment with GPT-3 (`text-davinci-002` checkpoint) (Brown et al., 2020) using a *perturbation recovery task*, wherein we prompt the model to decode perturbed words back to their original strings. We sample a subset of $1,000$ (w, w_i) pairs from **LEGIT** which have a label of *legible*. These perturbations are fed to the GPT-3 model in batches of 10, along with an instructional prompt (see Appendix E) and 4 examples; recovered words are received as a completion to the input prompt. In addition, we also perturb the same 1000 words using VIPER-DCES and report the accuracy of GPT-3 at reconstructing them. We observe that GPT-3 often returns a word with a short edit distance to the original word, and hence to capture this in our evaluation, we apply the Porter stemmer from NLTK (Loper and Bird, 2002) to both the original words and predicted reconstructions, and then measure how often their stemmed forms are the same. We repeat this experiment 3 times, randomly sampling the 4 examples in the prompt each time. Figure 5 shows the GPT-3 accuracy at different fractions of corrupted characters ($n = \{0.3, 0.7, 1.0\}$). As expected, the accuracy goes down as n increases, but we find that GPT-3 performs worse on **LEGIT** perturbations. This demonstrates that while state-of-the-art language models are mildly robust to the narrower range of perturbations considered in existing visual attacks, they degrade significantly on inputs sampled from **LEGIT** which are marked by humans as legible. This result underscores the importance of considering the entire space of legible perturbations when evaluating model robustness.

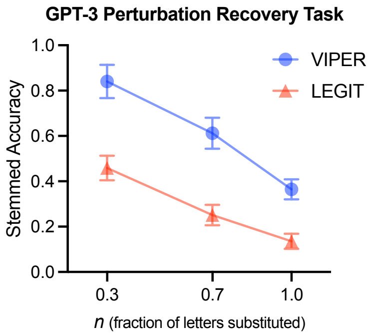


Figure 5: **LEGIT** perturbations sampled at low n degrade accuracy at levels comparable to the $n = 1$ VIPER configuration. Error bars indicate 95% confidence interval.

6 Conclusion

We set out to characterize the limits of legibility of visual perturbations. To do so, we first collected and released a new dataset, **LEGIT**, comprising legibility preferences of human subjects. Using this dataset, we framed a binary legible-or-not classification task, and a ranking task to rank candidate perturbations. For these tasks, we explored several text- and vision-based models, and found that our models obtain a high F1 score of 0.91 for the classification task and an accuracy of 0.86 for the ranking task. Perturbations generated using the same attack method as used for constructing **LEGIT** lead to significant degradation on the Jigsaw Challenge task and are not recovered by GPT-3 accurately, despite being filtered for legibility. We believe this work opens avenues for research on legibility-driven certified robustness to visual attacks in NLP.

Limitations

At the outset, we note that while our legibility-scoring models are a step forward towards defending against visual attacks, they should not be seen as perfect. Defending against all of the attacks which our models find legible might still leave room for legible attacks missed by our system.

Moreover, we note that the perturbation procedure outlined here only generates substitution-based perturbations. Whereas, characters may also be deleted, added, or swapped, and multiple adjacent characters may be substituted with visually similar counterparts (see Figure 1). Future work may explore broader classes of perturbations.

When constructing the dataset, we only chose words with a length of at least 4 letters, excluding many common 3-letter words. This is because for 3-letter words, there is a high likelihood that a bad perturbation may be mistakenly recognized as a good perturbation by virtue of being in-vocabulary. For example, “ban” is a bad perturbation of “man”, but for an annotator who sees it without knowing the original word and in absence of any sentence-level context, it seems like a perfectly good perturbation, when in fact it obscures the meaning of the original word. This is a limitation of the experimental setup that can lead to bad annotations, and to mitigate it we chose a higher minimum word length as longer words have fewer such collisions.

Further, we study word-level perturbations in isolation without any surrounding context, whereas in practice, readers often can decipher words based on the context. In general, the legibility of a text depends on the context around it—for example, even if a word is deleted from a sentence it is often possible to reconstruct it. The data we collect here, however, measures the legibility of individual words without any context, in order to simplify the generation and annotation process. As a result, the legibility estimated using this data should be considered as *lower bound* of the legibility in any given context. This was a deliberate choice as we wanted to ensure that whatever we ascertain as legible is legible in *all* contexts.

Lastly, the models we develop in our work are of relatively moderate size ($334 - 584$ million parameters) and take only unimodal input (i.e. pixels for TrOCR models and Unicode bytes for ByT5). and future work may be able to improve the performance by using larger models which accept multimodal input (e.g. both pixels and Unicode bytes simultaneously) and learn joint representations across these modalities.

Ethical Considerations

The word list comprising our dataset was filtered to remove swear words, slurs etc. in order to avoid exposing annotators to potentially harmful content.

Acknowledgements

We thank Professors Carlo Tomasi and Sam Wiseman at Duke University for their helpful feedback. This research was supported by a grant from the Arts & Sciences Council Committee on Faculty Research at Duke University.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896. Association for Computational Linguistics.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004. IEEE.

Thorsten Brants and Alex Franz. 2006. Web 1t 5-gram ver. 1. *LDC2006T13, Linguistic Data Consortium, Philadelphia*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2007. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic. Association for Computational Linguistics.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 213–229. Springer.

Alex Clark. 2015. Pillow (pil fork) documentation.

Mark Davis and Michel Suignard. 2021. UTS #39: Unicode Security Mechanisms. Version 14.0.

Antreas Dionysiou and Elias Athanasopoulos. 2021. Unicode Evil: Evading NLP Systems Using Visual Similarities of Text Characters. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, AISec ’21, pages 1–12, New York, NY, USA. Association for Computing Machinery.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On Adversarial Examples for Character-Level Neural Machine Translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 653–663, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Steffen Eger and Yannik Benz. 2020. From Hero to Zéroe: A Benchmark of Low-Level Adversarial Attacks. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 786–803, Suzhou, China. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding nlp systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647.

Eveline Flamand. 2008. Deciphering l33t5p34k internet slang on message boards. *MS thesis*.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181.

Laura Hanu and Unitary team. 2020. Detoxify. Github. https://github.com/unitaryai/detoxify.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Josh Kaufman. 2012. google-10000-english. `https://github.com/first20hours/google-10000-english`.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, page 0. Lille.

Thai Le, Jooyoung Lee, Kevin Yen, Yifan Hu, and Dongwon Lee. 2022. Perturbations in the Wild: Leveraging Human-Written Text Perturbations for Realistic Adversarial Attack and Defense. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2953–2965, Dublin, Ireland. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. TextBugger: Generating Adversarial Text Against Real-world Applications. *Proceedings 2019 Network and Distributed System Security Symposium*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202.

Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. 2021. TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models. Technical Report arXiv:2109.10282, arXiv. ArXiv:2109.10282 [cs] type: article.

Weixin Liang, James Zou, and Zhou Yu. 2020. Beyond user self-reported likert scale ratings: A comparison model for automatic dialog evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1363–1374.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *CoRR*, cs.CL/0205028.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. Version: 4.

Danish Pruthi, Bhuwan Dhingra, and Zachary C Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Eric S Raymond. 1996. *The new hacker's dictionary*. Mit Press.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

A mTurk Annotation Inferface

The web-based UI used by mTurk Annotators is shown in Figure 6, with the instructions being visible throughout the duration of the task. Note that the perturbed words w_1, w_2 are rendered in GNU Unifont, which is the same font that words are rendered in for computing visual similarity (cf. §Legibility Tests, Perturbation Process). This ensures that both annotators and visual similarity models see pixel-for-pixel identical perturbations, controlling for the fact that different fonts render the same character differently.

The interface is optimized for clarity and labeling speed, with a focus on eliminating unnecessary UI elements and minimizing clicks. Labeling each pair w_1, w_2 with one label $L \in \{L_1, L_2, BL, NL\}$ takes exactly one click. Annotators choose L_1 by clicking on w_1 (the left word), and similarly by clicking on w_2 (the right word) for w_2. BL is selected using the “equally legible" button, whereas NL is chosen by clicking on “both unclear."

Immediately after a choice is made, the UI updates and the next pair in the batch is shown (there is no option to go back and edit the chosen label). Annotators who attempt to cheat on the task by “speeding through" (i.e. clicking randomly or spamming the same choice) end up failing the occasionally administered quality checks and are subsequently disinvited from the study.

B Use of OCR Models

Boucher et al. (2022) propose using Optical Character Recognition (OCR) models to preprocess input for text-based language models. Rendering input text and passing it through an OCR before giving it to the language model filters certain kinds of misleading Unicode characters (e.g. invisible control sequences or near-identical Confusables (Davis and Suignard, 2021)) from the text. However, when used for legible but visually distinct perturbations, off-the-shelf OCR models run into two problems.

Firstly, both mono- and multi-lingual OCR models will recognize characters from learned scripts at face value, instead of recognizing their intended use as visually similar substitutions. For example, TrOCR (Li et al., 2021), when given an image of the string ‘Mex!(0’, decodes it into ‘Mex!(0’ (i.e. the same string), completely ignoring its intended meaning (Mexico). Secondly, since OCR models are only trained on semantically meaningful inputs, they do not learn good priors to differentiate nonsense inputs from highly perturbed inputs.

We use two OCR-capable models on the ranking and classification tasks: TrOCR, which is explicitly trained on an OCR dataset, as well as CLIP, which is trained on a general corpus containing images of texts from which it learns “a high quality semantic OCR representation that performs well on digitally rendered text” (Radford et al., 2021).

We find that TrOCR models fine-tuned on our dataset achieve high performance on legibility-related tasks. On the text side, we consider the token-free language model, ByT5 (Xue et al., 2022), which encodes each byte individually, as opposed to byte-pairs or subword tokens longer than one byte. Since its encoding of each byte is disentangled from surrounding bytes, ByT5 is able to retain a larger share of the unperturbed part of the string, hopefully making it more robust to character-substitution perturbations compared to token-based models, which reduce the sequences with perturbed characters into rare tokens or simply to `UNK`s.

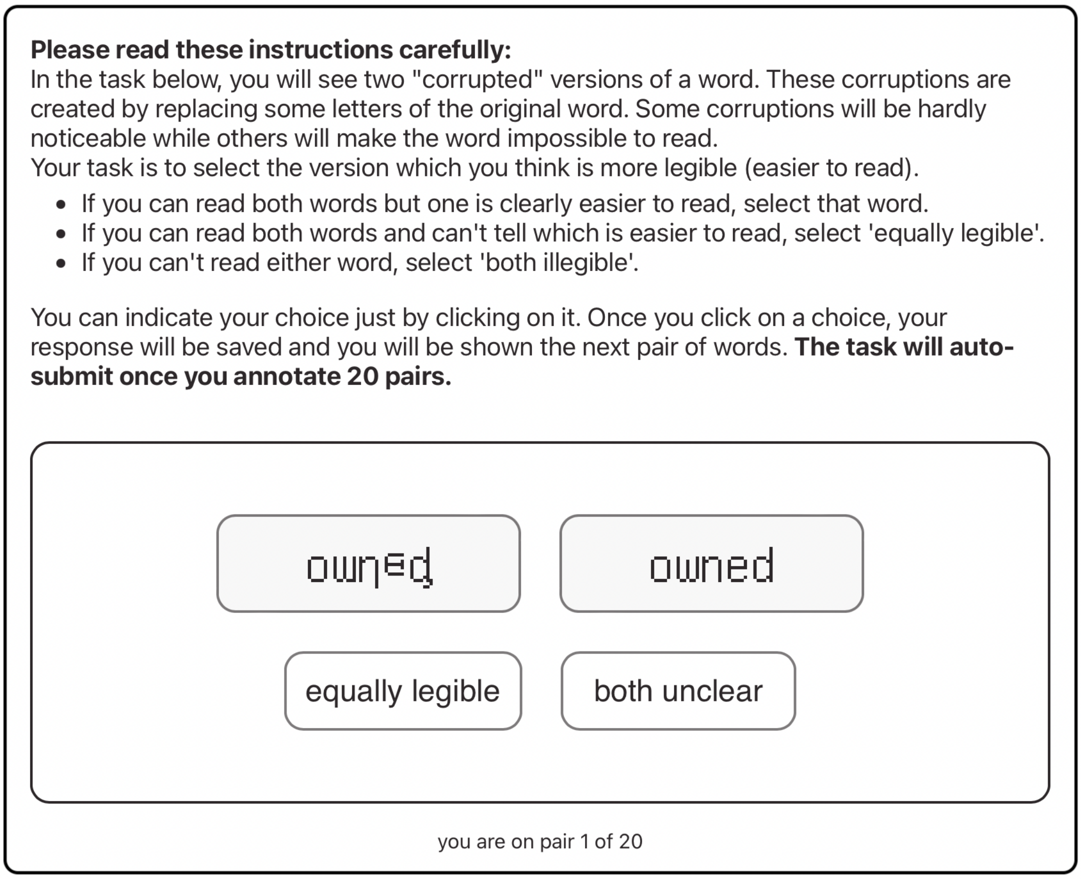
C Hyperparameters

TrOCR (‘base-handwritten’ version) was fine-tuned on **`LEGIT`** with the loss function configurations (C, R, MT) described above. To train each configuration, we use a single NVIDIA A6000 GPU (48GB VRAM) with a batch size of 26 and learning rate of 10^{-5} with the AdamW optimizer and a linear decay schedule (without warmup). ByT5-base and ByT5-small were trained on the same hardware with a batch size of 8 and learning rate of 10^{-4}.

D Toxic Comment Classification Experiment

The original string from the Jigsaw Toxic Comment Classification dataset is:

> `It is needed in this case to clarify that UB is a SUNY Center. It says it even in Binghampton University at Albany, State University of New York, and`



Please read these instructions carefully:
In the task below, you will see two "corrupted" versions of a word. These corruptions are created by replacing some letters of the original word. Some corruptions will be hardly noticeable while others will make the word impossible to read.
Your task is to select the version which you think is more legible (easier to read).

- If you can read both words but one is clearly easier to read, select that word.
- If you can read both words and can't tell which is easier to read, select 'equally legible'.
- If you can't read either word, select 'both illegible'.

You can indicate your choice just by clicking on it. Once you click on a choice, your response will be saved and you will be shown the next pair of words. **The task will auto-submit once you annotate 20 pairs.**

Figure 6: The mTurk Annotation Interface

> `Stony Brook University. Stop trying to say it’s not because I am totally right in this case.`

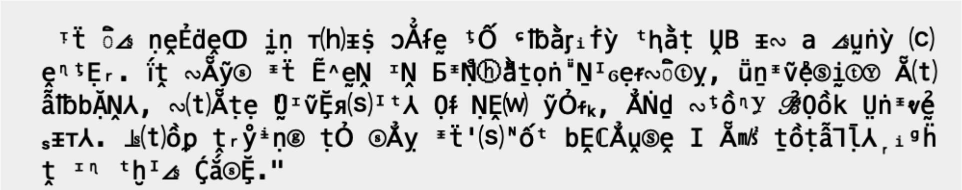
The VIPER DCES and **`LEGIT`** perturbations are compared in Figure 7. The **`LEGIT`** perturbations were labeled as legible by TrOCR-MT.

E GPT-3 Experiment

We provided the following prompt to the `text-davinci-002` checkpoint using the GPT-3 API:

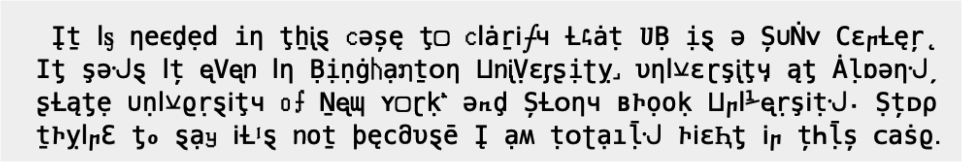
> `The following is a list of corrupted words and their correct versions. The corruptions were created by replacing some or all letters of the correct version with similar-looking letters.`
> `Corrupted:`
> `1.` c_1
> `2.` c_2
> `...`
> `10.` c_{10}
> `Original:`
> `1.` o_1
> `2.` o_2
> `3.` o_3
> `4.` o_4
> `5.`

The model is allowed to condition on 4 ground-truth examples: o_1 through o_4, and attempts to generate o_5 through o_{10} by providing a completion for the prompt above. The *temperature* and *top p* parameters were both set to 1 to allow for consistent and reproducible outputs across batches.



ᵀẗ ȏ⊿ ṇẹ̬Ė̉dẹↀ i̱ṇ ᴛ(h)ɪṣ ɔẮfẹ̰ ᵗŐ ᶜŀbằɽᵢḟỳ ᵗḥằṭ Ụ̂B ɪ∾ a ⊿ṵṅỳ (c)
ẹⁿᵗẸᵣ. íṭ̂ ∾Ẵỹ⊛ ᵻẗ Ẽ^ẹ̰Ṋ̂ ᴵṆ̂ Б⁼Ṇ̃ⓗȁṯọṅ̈N̲ᴵ₆ẹɽ∾ȏⓣy̨, ü̱n⁼ṽẹ̉ⓢi̱ⓣⓨ Ẳ(t)
ẫŀbbẬṆ̂ʎ, ∾(t)Ẵṭẹ Ṵ̆ᴵṽȨ̌ʀ(s)ᴵᵗʎ Ọf ṆỆ(w) ỹỎfₖ, ẮṄḏ ∾ᵗồⁿʸ ℬỌồk Ụṅ⁼vẹ̃
ₛɪᴛʎ. ⊥(t)ồp ṯᵣẙ⁺ṇ⊛ ṭỎ ⊛Ẳy̨ ⁼ẗ'(s)ᴺốᵗ bỆℭẮụ̂⊛ệ I Ẵm tồṭẫlḻʎ ᵣ ᵢ ᵍḧ
ṭ̂ ᴵⁿ ᵗḫᴵ⊿ Ḉắ⊛Ḝ."

(a) VIPER DCES, $n = 1.0$, nearest neighbors sampled uniformly from list of top 10 neighbors for each character.



Ḭṯ lş ɳeeḍẹd iɳ ţẖi̭ş cəṣẹ ţ□ cláṟiƒч Łɕàṭ ŪḄ i̇ş ə ŞUṄv Cɛɲŀẹṛ.
Iţ şə𝗝ş lṭ ę̣Vę̣n lɳ Ḅịṅġḩạɲṯoɳ ⅬniṼɛɽşịṭy̨. ʋɳlⅴɛɽşi̭ţч ạţ Ȧḻɒəɳ𝗝,
şŁạţẹ uṇlⅴǫ̣ṛşiţч ₀ƒ N̲ẹɰ ʏ□ɽḳ ənd̦ ŞŁoɳч ʙɦọok Ⳑɲlᑊę̣ṛşiṭ𝗝. Şṭɒp
ṯɦy̨lɲƐ ţₒ şạᵧ iŁᴵş noṯ þẹcƋʋşē Ḭ ạᴍ ṭoţạɪḻ𝗝 ʜiɛḩţ iɲ ţhḻş caṡǫ.

(b) Ours (**LEGIT** perturbation strategy with TʀOCR-MT legibility filter) $n = 1.0$, nearest neighbors sampled normally ($\mu = 15$, $\sigma^2 = 7$) from top 30 neighbors for each character

Figure 7: A randomly selected paragraph from the Jigsaw dataset (a) perturbed by VIPER DCES (b) and our method (c). Our perturbation appears more legible despite being generated using harsher parameters.

DyLoRA: Parameter-Efficient Tuning of Pretrained Models using Dynamic Search-Free Low Rank Adaptation

Mojtaba Valipour[1,2] **Mehdi Rezagholizadeh**[2] **Ivan Kobyzev**[2] **Ali Ghodsi**[1]

{mojtaba.valipour, ali.ghodsi}@uwaterloo.ca, {mehdi.rezagholizadeh, ivan.kobyzev}@huawei.com

1: University of Waterloo, 2: Huawei Noah's Ark Lab

Abstract

With the ever-growing size of pretrained models (PMs), fine-tuning them has become more expensive and resource-hungry. As a remedy, low-rank adapters (LoRA) keep the main pretrained weights of the model frozen and just introduce some learnable truncated SVD modules (so-called LoRA blocks) to the model. While LoRA blocks are parameter-efficient, they suffer from two major problems: first, the size of these blocks is fixed and cannot be modified after training (for example, if we need to change the rank of LoRA blocks, then we need to retrain them from scratch); second, optimizing their rank requires an exhaustive search and effort. In this work, we introduce a dynamic low-rank adaptation (DyLoRA) technique to address these two problems together. Our DyLoRA method trains LoRA blocks for a range of ranks instead of a single rank by sorting the representation learned by the adapter module at different ranks during training. We evaluate our solution on different natural language understanding (GLUE benchmark) and language generation tasks (E2E, DART and WebNLG) using different pretrained models such as RoBERTa and GPT with different sizes. Our results show that we can train dynamic search-free models with DyLoRA at least 4 to 7 times faster than LoRA without significantly compromising performance. Moreover, our models can perform consistently well on a much larger range of ranks compared to LoRA. [1]

1 Introduction

Pre-training/fine-tuning has become a popular paradigm for solving many tasks in natural language processing (NLP) (Devlin et al., 2018; Liu et al., 2019; Brown et al., 2020) and Computer Vision (Simonyan and Zisserman, 2014; He et al., 2016; Howard et al., 2019; Bochkovskiy et al., 2020; Chen et al., 2020; Dosovitskiy et al., 2020). pretrained models (PMs) such as pretrained language models (PLMs) (Devlin et al., 2018; Brown et al., 2020), and pretrained visual-language models (Lu et al., 2019; Li et al., 2019; Su et al., 2019; Xia et al., 2021) have advanced a lot in recent years. With the ever-growing size of these pretrained models, fine-tuning them on downstream tasks becomes more expensive. Moreover, as the ratio of the number of parameters of models with respect to the labeled data increases, the fine-tuning process will be more prone to overfitting (Karimi Mahabadi et al., 2021). There are two categories of solutions: first, model compression (Jafari et al., 2021; Chen et al., 2021); second, parameter-efficient tuning (PET) (Houlsby et al., 2019a; Karimi Mahabadi et al., 2021; Mao et al., 2021).

There are many different model compression techniques in the literature for Transformer-based models such as matrix factorization (Noach and Goldberg, 2020; Tahaei et al., 2021), pruning (Wang et al., 2019), quantization (Tao et al., 2022; Prato et al., 2020), and knowledge distillation (Hinton et al., 2015; Li et al., 2021; Jafari et al., 2021; Passban et al., 2021; Rashid et al., 2021). There are also different types of PET techniques in the literature such as low-rank adapters (Wang et al., 2020; Karimi Mahabadi et al., 2021; Houlsby et al., 2019b; Hu et al., 2021b), and prompt-based techniques (Lester et al., 2021).

Although model compression solutions are well-established in recent years in the literature, applying them to large language models can be very costly, because compression techniques usually need to train (or fine-tune) the original large model. A case in point is knowledge distillation which relies on fine-tuning a large teacher model or even pre-training the student model as suggested in (Jiao et al., 2019). Moreover, using compression techniques usually leads to degrading the model performance. PETs can be alternatives to the compression methods, especially when we would like to use

---

[1] github.com/huawei-noah/KD-NLP/tree/main/DyLoRA

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3274–3287
May 2-6, 2023

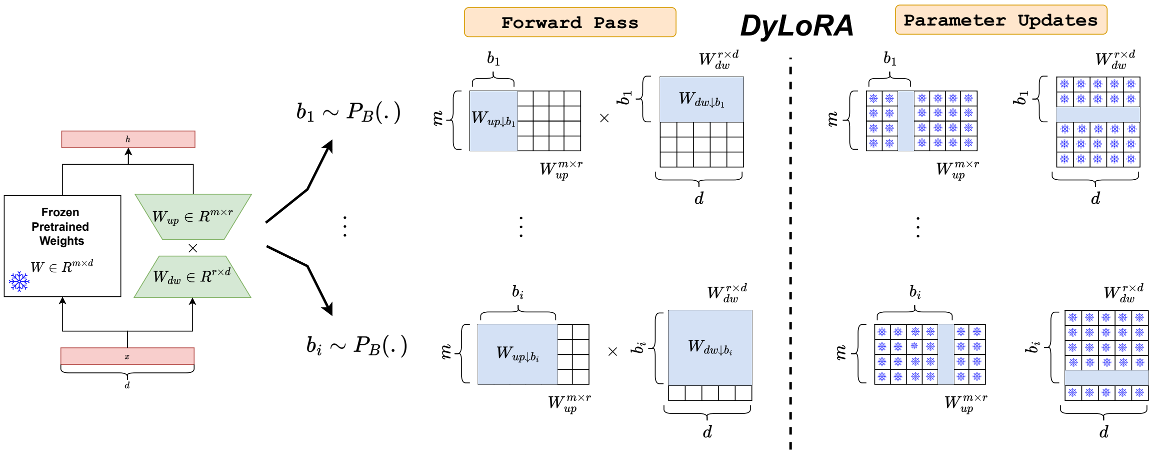


Figure 1: DyLoRA: The overall diagram of our proposed method. In each iteration, we sample from a pre-defined random distribution which will help us to truncate the up-projection and down-projection matrices in the LoRA (Hu et al., 2021a) objective.

the full capacity of the large pretrained models with light training efforts (such as the *language-model-as-a-service* scenario (Sun et al., 2022)). Among PET techniques, low-rank adapters have received much attention because, in contrast to prompt-tuning techniques, low-rank adapters do not add to the sequence length, get trained faster, and perform better (Karimi Mahabadi et al., 2021). Even though there are several low-rank adaptation techniques in the literature, such as Adapter (Houlsby et al., 2019b), Compacter (Karimi Mahabadi et al., 2021), and LoRA (Hu et al., 2021b); they all suffer from two major common problems: first, it is not clear how to select the size of their rank (while their performance is very sensitive to this rank selection); second, their training is static which means that if a low-rank model is trained based on a particular rank size, it will not work well in other rank values (i.e. for any other rank value we need to train a separate model).

This paper proposes a dynamic low-rank adapter technique (DyLoRA) to address these two problems. Without loss of generality, we focus on LoRA(Hu et al., 2021a) and train LoRA blocks for a range of ranks instead of a single rank by sorting out the representation learned at different ranks during training. While our model is more flexible, it can outperform LoRA in a much wider range of ranks without adding to the training time. Moreover, our technique does not need extra training for searching across ranks. We summarize our contributions in the following:

- Dynamic LoRA: On top of LoRA, we developed a new algorithm (DyLoRA) that makes it dynamic at inference time without incurring extra costs.
- Search-free LoRA: We demonstrate that by making a negligible compromise in performance, it is possible to avoid the costly search process of choosing the optimal rank for LoRA.

2 Related Work

This section reviews low-rank adaptation techniques for parameter-efficient tuning and potential existing solutions to make these techniques dynamic and search-free.

It has been shown in (Aghajanyan et al., 2020) that for classification tasks such as natural language understanding (NLU), PLMs have a low intrinsic dimension. This observation motivates the use of low-rank adapters for parameter-efficient tuning. There are several low-rank adapters in the literature such as LoRA (Hu et al., 2021b), Adapter (Houlsby et al., 2019b), Compacter (Karimi Mahabadi et al., 2021), and Parallel Adapter (PA) (He et al., 2021). LoRA is a low-rank up-projection/down-projection transformation without any non-linearity applied in parallel to key and value attention matrices. The main benefit of LoRA is that the adapter module, after training, can be integrated into the original weight matrices of the model, which in turn can lead to a very efficient inference time. Adapters also have a low-rank up-projection/down-projection transformation with an intermediate non-linearity. The Adapter module is applied in series with the feed-forward network (FFN). Having the adaptor module in-line with other blocks in the model can increase the inference time of the model. PA is a faster version of the Adapter, which can be applied in parallel with the FFN block. The compactor is a more memory-efficient version of the Adapter, which deploys the sum of Kronecker products to reconstruct each up-projection and down-projection matrices. All these low-rank adapters suffer from two major issues: first, finding the best rank requires heavy exhaustive training and search; second, the tuned adapter module works well only with a particular rank.

While there have been some efforts in the literature towards dynamic networks such as DynaBERT (Hou et al., 2020) and GradMax (Evci et al., 2022), to the best of our knowledge, this problem for factorized networks and low-rank adapters is still open. DRONE (Chen et al., 2021) propose a technique for data-aware low-rank model compression however their approach is not search-free, and also, it is not dynamic. DynaBERT introduces a two-stage method to train width and depth-wise dynamic networks. However, DynaBERT requires a fine-tuned teacher model on the task to train its sub-networks which makes it unsuitable for PET techniques. GradMax is a technique that gradually adds to the neurons of a network without touching the already trained neurons. But it is unclear how GradMax can be deployed to alleviate the rank-search problem in low-rank adapters. Wang et al. (2019) propose a structured pruning technique called factorized low-rank pruning (FLOP). FLOP decomposes weight matrices of a network into the sum of rank-1 components, which are regularized during training to gain sparsity. It is worth mentioning that FLOP aims at compressing the main model, and even if it can be used for finding a good rank in the lower-rank representation of full-weight matrices, the final low-rank model will not be dynamic (i.e. it is trained well only for one rank and not a range of ranks, same as LoRA.). In this paper, we propose a new methodology for training low-rank modules for multiple ranks simultaneously rather than training a single-rank adapter at a time (without changing the training budget). Inspired by the idea of *nested dropout* (Rippel et al., 2014), we pursue ordering the representations of the bottleneck at the low-rank adapter modules with a new recipe. To the best of our knowledge, it is the first time that the concept of ordering representations has been deployed in training PLMs.

3 Background

3.1 Nested Dropout

Inspired by the dropout (Hinton et al., 2012), nested drop-out (Rippel et al., 2014) is a stochastic regularization technique that targets enforcing ordered representations in training auto-encoders. The nested dropout, adds an implicit bias (which does not exist in dropout) to favor order in training. For example, in dropout, we can randomly drop any nodes or units in the network, but in nested dropout, if we randomly select k^{th} unit, then we keep all the units indexed from 1 to k and drop the units with indices larger than k. Therefore, nested dropout tends toward accommodating more important information in lower indices while learning representations.

Following the notations of (Rippel et al., 2014), nested dropout assumes an auto-encoder mapping of N training examples $\{y_i\}_{i=1}^{N} \in Y$, $Y \subset \mathbb{R}^D$ to their corresponding representations $\{x_i\}_{i=1}^{N} \in X$, $X \subset \mathbb{R}^K$ using the function $f_\theta : Y \to X$ with parameters θ; and then decoding these representations using another function $g_\psi : X \to Y$ with parameters ψ to reconstruct the inputs. The reconstruction loss can be defined as follows:

$$C(\theta, \psi) = \sum_{i=1}^{N} ||y_i - g_\psi(f_\theta(y_i))||^2. \tag{1}$$

Suppose we want to randomly drop some units in our representation vector x. In this regard, we sample a random variable $b \sim p_B(.), b \in \{1, 2, ..., K\}$ from a pre-defined categorical distribution $p_B(.)$ and truncate the functions f_θ and g_ψ to keep their corresponding units indexed from 1 to b and dropping $b{+}1$ to K indices. Let's define the b-truncated version of the vector x as $x_{\downarrow b}$ and the b-truncated version of the functions f_θ and g_ψ as $f_{\theta\downarrow b}$ and $g_{\psi\downarrow b}$ respectively. In this case, the reconstruction loss is redefined for the b-truncated model as follows:

$$C(\theta, \psi) = \mathbb{E}_{p_B}[C_{\downarrow b}(\theta, \psi)] = \sum_{b=1}^{K} p_B(b) C_{\downarrow b}(\theta, \psi)$$

where

$$C_{\downarrow b}(\theta, \psi) = \sum_{i=1}^{N} ||y_i - g_{\psi\downarrow b}(f_{\theta\downarrow b}(y_i))||^2. \tag{2}$$

In the final stage, the parameters of this model can be obtained by solving the following optimization problem.

$$(\theta^*, \psi^*) = \underset{\theta,\psi}{\operatorname{argmin}}\, C(\theta, \psi). \quad (3)$$

While our work in this paper is inspired by the feature of ordering information suggested in nested dropout, we can distinguish our work from nested dropout in several aspects:

1. The nested dropout technique is used to add order information to a vector representation; however, we are adding order information to the low-rank matrix decomposition to make it work across a range of ranks instead of a single rank.

2. Our training algorithm differs from nested dropout in the choice of the distribution function $p_B(.)$, and we propose a more efficient individual loss for each truncated matrix compared to the linear summation loss (check equations 2 and 11 in the original paper (Rippel et al., 2014)) in nested dropout. The original proposal for the nested dropout was to use a batch with mixed truncated examples. To enhance efficiency and resolve suboptimality, we propose to fix truncation in the entire batch as part of our approach.

3.2 LoRA: Low-rank Adapters

In LoRA (Hu et al., 2021a), some pretrained weights of dense layers of PLMs are summed with parallel linear low-rank adapter modules. During fine-tuning, the original pretrained weights are kept frozen; LoRA modules can be updated instead. For example, let's assume that $W_0 \in \mathbb{R}^{m\times d}$ is a pretrained weight matrix in the network which is accompanied by a LoRA module $\Delta W = W_{up}W_{dw}$ where $W_{up} \in \mathbb{R}^{m\times r}$, $W_{dw} \in \mathbb{R}^{r\times d}$, and $r \ll min(m, d)$. Then, the output of this layer can be obtained as

$$h = W_0x + \Delta W x = W_0x + \frac{\alpha}{r} W_{up}W_{dw}x. \quad (4)$$

Bear in mind that the W_{up} matrix is initialized as a zero matrix, and the W_{dw} matrix is initialized as a zero-mean Gaussian distribution where α is a constant scale hyper-parameter.

In LoRA, the rank r is a hyperparameter that should be tuned for each task. Moreover, LoRA is a *static* low-rank adapter that works only with a particular size of r, which has been trained on it.

4 Our Method: DyLoRA

In this section, we introduce our solution to get dynamic low-rank adapters that can be trained and deployed well on a range of ranks instead of a single particular rank (with a fixed training budget). This flexibility can free us from searching for the best ranks by training the model multiple times.

Without loss of generality, we explain our solution on top of LoRA as one of the prominent low-rank adapter techniques in the literature. In each LoRA module, we have an up-projection ($W_{up} \in \mathbb{R}^{m\times r}$) and a down-projection matrix ($W_{dw} \in \mathbb{R}^{r\times d}$). Let's assume that we would like to train the LoRA module to operate in the range of $r \in \text{Range}[r_{min}, r_{max}]$ where r_{min} and r_{max} can be treated as new hyper-parameters. To make the LoRA module work in a range of ranks instead of a single rank, we need to ensure that increasing or decreasing the rank will not significantly hamper the model's performance. One way to implement such behavior would be by sorting the information content of different ranks in the training process of LoRA modules. In this regard, at each training step, we sample $b \sim p_B(.), b \in \{r_{min}, r_{min} + 1, ..., r_{max}\}$ form a pre-defined categorical distribution (which has a support in $\text{Range}[r_{min}, r_{max}]$) and truncate W_{dw} and W_{up} matrices accordingly.

$$\begin{aligned} W_{dw\downarrow b} &= W_{dw}[1:b,:] \\ W_{up\downarrow b} &= W_{up}[:,1:b] \end{aligned} \quad (5)$$

$W_{dw\downarrow b}$ and $W_{up\downarrow b}$ are b-truncated versions of W_{dw} and W_{up} respectively (see Fig. 1 for the visualization). Moreover, let's define W^b_{dw} as the b^{th} row of W_{dw}; W^b_{up} corresponds to the b^{th} column of W_{up}.

$$\begin{aligned} W^b_{dw} &= W_{dw}[b,:] \\ W^b_{up} &= W_{up}[:,b] \end{aligned} \quad (6)$$

Then, the forward pass of this truncated LoRA module during training will be calculated as following:

$$h = W_0x + \frac{\alpha}{b} W_{up\downarrow b}W_{dw\downarrow b}x \quad (7)$$

For simplicity, let's assume that we have only one LoRA module in the network (the one which is described in Eq. 7). Let's first consider the regular static loss function ($\mathcal{L}^S$) of the network $f(x; W_{dw}, W_{up})$ with W_{dw} and W_{up} tunable parameters for N given input-output pairs $(\mathbf{x}, \mathbf{y}) =$

$(x_i, y_i)_{i=1}^{N}$:

$$\min_{W_{dw},W_{up}} \mathcal{L}^{\mathcal{S}}(\mathbf{x},\mathbf{y};W_{dw},W_{up}) \triangleq \sum_{i=1}^{N} l(f(x_i;W_{dw},W_{up}),y_i). \tag{8}$$

where $l(f, \mathbf{y})$ is a loss function that measures the divergence of network predictions compared with the target labels. Then, let's extend the training loss to make the network dynamic considering the b-truncation process. We can define our dynamic loss function $\mathcal{L}^{\mathcal{DY}}$ as follows.

$$\mathcal{L}^{\mathcal{DY}}_{\downarrow b} = \sum_{i=1}^{N} l(f(x_i;W_{dw\downarrow b},W_{up\downarrow b}),y_i). \tag{9}$$

Bear in mind that, our loss function has a major difference from the nested dropout loss, which makes it more efficient. The nested dropout loss is in the form of $\sum_{b=r_{min}}^{r_{max}} p_B(b)\mathcal{L}^{\mathcal{DY}}_{\downarrow b}(\mathbf{x},\mathbf{y};W_{dw\downarrow b},W_{up\downarrow b})$ which requires to sum the loss over the entire possible range of ranks and it is computationally expensive. To overcome this computational restriction, we replace it by optimizing the model parameters for each target rank individually at each time step. We show that this scheme quite works well.

The other difference with nested dropout is that in the parameter update phase, we add a new mode (so-called *frozen*) as a hyper-parameter to our training. This new mode suggests to only update the b^{th} corresponding row and column sampled in the truncation phase (i.e. a single row or column will be updated at a time to prevent the learning parameters from being forgotten at previous time steps.). With a minor performance cost, this approach can improve the efficiency of our algorithm even further.

$$\begin{aligned} W^{b}_{dw} &\leftarrow W^{b}_{dw} - \eta\nabla_{W^{b}_{dw}}\mathcal{L}^{\mathcal{DY}}_{\downarrow b} \\ W^{b}_{up} &\leftarrow W^{b}_{up} - \eta\nabla_{W^{b}_{up}}\mathcal{L}^{\mathcal{DY}}_{\downarrow b} \end{aligned} \tag{10}$$

Table 4 shows the impact of only updating "b" versus updating the columns and rows from 1 to b. The summary of our technique is described in Algorithm 1.

Algorithm 1 DyLoRA - Training

```
Require:
  $r$ ∈Range[$r_{min}$,$r_{max}$]; $i$: the number of training iterations; $\alpha$: a scaling factor; $p_B$: probability distribution function for rank selection; $X \in \mathbb{R}^{d\times n}$ : all input features to LORA; $W_0 \in \mathbb{R}^{m\times d}$ the original frozen pretrained weight matrix
Require: $W_{dw} \in \mathbb{R}^{r\times d}$; $W_{up} \in \mathbb{R}^{m\times r}$, FROZEN: whether to keep the lower ranks frozen when updating the higher ranks
  while t < $i$ do:
    Forward:
    // sample a specific rank, during test is given
    $b \sim p_B(.)$
    // truncate down-projection matrix
    $W_{dw\downarrow b} = W_{dw}[:b,:]$
    $W^{b}_{dw} = W_{dw}[b,:]$
    // truncate up-projection matrix
    $W_{up\downarrow b} = W_{up}[:,:b]$
    $W^{b}_{up} = W_{up}[:,b]$
    // calculate the LoRA output
    $h = W_0X + \frac{\alpha}{b}W_{up\downarrow b}W_{dw\downarrow b}X$
    Backward:
    if FROZEN then
            // only update the unique parameters of the selected rank
            $W^{b}_{dw} \leftarrow W^{b}_{dw} - \eta\nabla_{W^{b}_{dw}}\mathcal{L}^{\mathcal{DY}}_{\downarrow b}$
            $W^{b}_{up} \leftarrow W^{b}_{up} - \eta\nabla_{W^{b}_{up}}\mathcal{L}^{\mathcal{DY}}_{\downarrow b}$
    else
            $W_{dw\downarrow b} \leftarrow W_{dw\downarrow b} - \eta\nabla_{W^{b}_{dw\downarrow b}}\mathcal{L}^{\mathcal{DY}}_{\downarrow b}$
            $W_{up\downarrow b} \leftarrow W_{up\downarrow b} - \eta\nabla_{W^{b}_{up\downarrow b}}\mathcal{L}^{\mathcal{DY}}_{\downarrow b}$
    end if
  end while
```

5 Experiments

In this section, we describe the experiments used to evaluate our DyLoRA model on both natural language understanding (NLU) and natural language generation (NLG) tasks. To be fair with the original LoRA method, we try to keep the setting of our experiments similar to the LoRA paper (Hu et al., 2021a). Therefore similarly, we chose the pretrained RoBERTa (Liu et al., 2019) base model as the backbone of the LoRA and DyLoRA experiments for the GLUE benchmark (Development Set), and GPT-Medium for the NLG tasks. For our experiments, we did not use any hyper-parameter tuning, nor did we search the validation epochs, nor did we use MLNI trick (use the MLNI checkpoint instead of the pretrained weights) to enhance the model's performance. More details about the hyper-parameters is available in Table 8 in Appendix B. In total, we conducted more than 200 experiments and evaluated more than 1600 models, details of

Model: RoBERTa-Base						
Task	**Rank=1**	**Rank=2**	**Rank=4**	**Rank=8**	**Rank=16**	**Rank=32**
QQP (Accuracy)	$\underline{89.14}$	89.96	90.33	90.69	90.95	**91.02**
SST-2 (Accuracy)	$\underline{93.58}$	94.15	94.38	**94.84**	94.27	94.5
MRPC (Accuracy)	87.25	87.75	88.24	87.25	$\underline{86.76}$	**89.22**
CoLA (Mathews)	61.84	$\underline{57.78}$	61.57	**63.81**	63.07	62.82

Table 1: The effect of the rank of the low-rank adaptation matrix over the performance of the model. In this experiment, all the other hyperparameters are fixed, and we only changed the rank of the LoRA model. In this search space, Underline shows the minimum performance rank, and the **bold** number shows the maximum performance rank.

	Accuracy	**Accuracy**	**F1**	**Mathews**	**Accuracy**	**Accuracy**	**Accuracy**	**Pearson**	
Model	**MNLI**	**SST-2**	**MRPC**	**CoLA**	**QNLI**	**QQP**	**RTE**	**STS-B**	**Avg**
Rank = 1									
LoRA	34.60$_{\pm 3.69}$	69.61$_{\pm 7.99}$	83.47$_{\pm 3.90}$	25.57$_{\pm 9.71}$	53.00$_{\pm 2.95}$	44.30$_{\pm 7.50}$	57.55$_{\pm 5.51}$	76.07$_{\pm 6.06}$	54.90
DyLoRA (Frozen)	85.36$_{\pm 0.26}$	93.51$_{\pm 0.49}$	90.75$_{\pm 0.70}$	56.95$_{\pm 1.54}$	91.70$_{\pm 0.28}$	87.87$_{\pm 0.17}$	66.79$_{\pm 8.54}$	89.95$_{\pm 0.24}$	82.86
DyLoRA	85.59$_{\pm 0.07}$	93.23$_{\pm 0.63}$	91.58$_{\pm 0.69}$	57.93$_{\pm 2.12}$	91.95$_{\pm 0.14}$	88.37$_{\pm 0.15}$	74.80$_{\pm 1.48}$	90.30$_{\pm 0.13}$	**84.22**
Rank = 2									
LoRA	40.53$_{\pm 6.17}$	82.75$_{\pm 5.08}$	88.00$_{\pm 1.81}$	43.30$_{\pm 4.67}$	63.42$_{\pm 2.99}$	59.21$_{\pm 6.13}$	68.88$_{\pm 1.26}$	85.51$_{\pm 1.94}$	66.45
DyLoRA (Frozen)	85.74$_{\pm 0.28}$	93.76$_{\pm 0.52}$	91.09$_{\pm 0.45}$	56.88$_{\pm 2.09}$	92.03$_{\pm 0.22}$	88.21$_{\pm 0.07}$	63.90$_{\pm 12.85}$	90.25$_{\pm 0.15}$	82.73
DyLoRA	86.02$_{\pm 0.06}$	93.81$_{\pm 0.30}$	91.66$_{\pm 0.46}$	59.91$_{\pm 1.88}$	92.39$_{\pm 0.25}$	89.33$_{\pm 0.05}$	76.03$_{\pm 1.61}$	90.60$_{\pm 0.09}$	**84.97**
Rank = 3									
LoRA	58.95$_{\pm 6.02}$	90.00$_{\pm 1.27}$	89.66$_{\pm 1.25}$	56.78$_{\pm 1.88}$	79.26$_{\pm 4.80}$	72.58$_{\pm 4.09}$	72.49$_{\pm 2.30}$	88.80$_{\pm 0.29}$	76.07
DyLoRA (Frozen)	85.78$_{\pm 0.25}$	93.76$_{\pm 0.26}$	91.78$_{\pm 0.89}$	58.86$_{\pm 0.32}$	92.17$_{\pm 0.18}$	88.40$_{\pm 0.0}$	70.90$_{\pm 6.14}$	90.50$_{\pm 0.29}$	84.02
DyLoRA	86.70$_{\pm 0.09}$	94.11$_{\pm 0.33}$	91.56$_{\pm 0.86}$	60.97$_{\pm 2.01}$	92.77$_{\pm 0.21}$	89.76$_{\pm 0.07}$	77.11$_{\pm 2.97}$	90.69$_{\pm 0.14}$	**85.46**
Rank = 4									
LoRA	72.10$_{\pm 5.25}$	91.56$_{\pm 0.34}$	89.62$_{\pm 0.92}$	58.53$_{\pm 3.93}$	85.09$_{\pm 1.20}$	80.78$_{\pm 3.73}$	73.07$_{\pm 2.29}$	89.28$_{\pm 0.72}$	80.00
DyLoRA (Frozen)	85.93$_{\pm 0.19}$	93.85$_{\pm 0.33}$	91.28$_{\pm 0.71}$	59.25$_{\pm 1.05}$	92.27$_{\pm 0.16}$	88.52$_{\pm 0.08}$	71.12$_{\pm 2.46}$	90.53$_{\pm 0.18}$	84.10
DyLoRA	86.82$_{\pm 0.04}$	94.40$_{\pm 0.13}$	92.06$_{\pm 0.46}$	59.81$_{\pm 1.71}$	92.91$_{\pm 0.31}$	89.80$_{\pm 0.10}$	77.40$_{\pm 2.72}$	90.86$_{\pm 0.06}$	**85.53**
Rank = 5									
LoRA	78.61$_{\pm 3.97}$	92.82$_{\pm 0.46}$	90.75$_{\pm 0.96}$	60.37$_{\pm 3.10}$	88.97$_{\pm 0.90}$	85.26$_{\pm 1.56}$	73.21$_{\pm 2.17}$	89.90$_{\pm 0.30}$	82.49
DyLoRA (Frozen)	85.95$_{\pm 0.17}$	93.78$_{\pm 0.26}$	91.28$_{\pm 0.64}$	59.41$_{\pm 1.30}$	92.30$_{\pm 0.17}$	88.56$_{\pm 0.09}$	71.48$_{\pm 2.92}$	90.60$_{\pm 0.20}$	84.17
DyLoRA	87.00$_{\pm 0.10}$	94.29$_{\pm 0.41}$	91.73$_{\pm 0.60}$	60.52$_{\pm 1.07}$	93.01$_{\pm 0.28}$	90.04$_{\pm 0.10}$	76.90$_{\pm 2.11}$	90.97$_{\pm 0.20}$	**85.56**
Rank = 6									
LoRA	83.02$_{\pm 1.59}$	93.49$_{\pm 0.88}$	91.28$_{\pm 0.63}$	61.94$_{\pm 2.27}$	90.32$_{\pm 0.76}$	87.54$_{\pm 1.51}$	76.68$_{\pm 1.16}$	90.12$_{\pm 0.12}$	84.30
DyLoRA (Frozen)	85.98$_{\pm 0.16}$	93.76$_{\pm 0.46}$	91.12$_{\pm 0.43}$	58.95$_{\pm 1.10}$	92.46$_{\pm 0.14}$	88.68$_{\pm 0.13}$	72.64$_{\pm 2.44}$	90.64$_{\pm 0.23}$	84.28
DyLoRA	86.97$_{\pm 0.20}$	94.27$_{\pm 0.37}$	91.44$_{\pm 0.64}$	60.16$_{\pm 1.70}$	93.01$_{\pm 0.21}$	90.07$_{\pm 0.14}$	77.33$_{\pm 1.66}$	91.03$_{\pm 0.20}$	**85.53**
Rank = 7									
LoRA	85.44$_{\pm 0.78}$	93.62$_{\pm 0.35}$	91.27$_{\pm 0.73}$	62.19$_{\pm 2.66}$	91.88$_{\pm 0.23}$	89.51$_{\pm 0.30}$	75.52$_{\pm 1.41}$	90.35$_{\pm 0.24}$	84.97
DyLoRA (Frozen)	86.08$_{\pm 0.14}$	93.97$_{\pm 0.17}$	91.02$_{\pm 0.70}$	58.76$_{\pm 0.94}$	92.30$_{\pm 0.10}$	88.77$_{\pm 0.06}$	73.50$_{\pm 1.67}$	90.68$_{\pm 0.15}$	84.38
DyLoRA	86.82$_{\pm 0.10}$	94.27$_{\pm 0.33}$	91.38$_{\pm 0.59}$	59.51$_{\pm 1.75}$	92.99$_{\pm 0.26}$	90.04$_{\pm 0.06}$	77.91$_{\pm 1.58}$	91.07$_{\pm 0.19}$	**85.50**
Rank = 8									
LoRA	86.82$_{\pm 0.18}$	94.01$_{\pm 0.30}$	91.48$_{\pm 0.73}$	62.08$_{\pm 1.37}$	92.39$_{\pm 0.39}$	90.42$_{\pm 0.02}$	74.51$_{\pm 0.41}$	90.48$_{\pm 0.24}$	85.27
DyLoRA (Frozen)	86.10$_{\pm 0.04}$	93.69$_{\pm 0.41}$	91.19$_{\pm 0.79}$	58.52$_{\pm 0.95}$	92.47$_{\pm 0.18}$	88.82$_{\pm 0.06}$	73.29$_{\pm 2.49}$	90.68$_{\pm 0.14}$	84.35
DyLoRA	86.76$_{\pm 0.13}$	94.36$_{\pm 0.38}$	91.38$_{\pm 0.83}$	59.51$_{\pm 1.84}$	93.00$_{\pm 0.32}$	89.91$_{\pm 0.08}$	77.55$_{\pm 0.59}$	91.05$_{\pm 0.19}$	**85.44**
Best (Rank)									
LoRA	87.03(8)	94.50(6)	92.25(7)	**66.05**(7)	92.81(8)	**90.45**(8)	77.98(6)	90.87(8)	86.49
DyLoRA (Frozen)	86.18(7)	94.50**(2)**	**92.93(3)**	61.57(5)	92.70**(6)**	88.88(8)	75.81(7)	90.89**(6)**	85.43
DyLoRA	**87.17(6)**	**94.72**(7)	92.79(8)	63.32**(3)**	**93.56**(8)	90.17**(6)**	**80.14(4)**	**91.36**(7)	**86.66**
Full Rank									
Fine Tune*	**87.6**	**94.8**	90.2	63.6	92.8	**91.9**	78.7	91.2	86.4

Table 2: In this table, the task is to find a low-rank adaptation matrix that works with different ranks at inference time given a fixed budget (training time).

which can be found in the attachments.

5.1 Baselines

- **Fine Tune**: To show a relative upper bound for the performance of our proposed method, we fine-tuned all the parameters in the model. Even though we have a large number of trainable parameters, this can help us better understand how higher-rank models perform.
- **LoRA**: As a baseline to DyLoRA, we employed the original LoRA model with their tuned hyperparameters (Hu et al., 2021a). As

		Accuracy	F1	Accuracy	Pearson	
Model (Rank)	**Trainable Params**	**SST-2**	**MRPC**	**QNLI**	**STS-B**	**AVERAGE**
Fine Tune*	125M	**94.8**	90.2	92.8	**91.2**	92.25
FLOP*	80M	92.09	88.61	89.05	88.18	89.48
LoRA (1)	**0.628M**	93.58	91.93	91.98	90.85	92.09
Maximum Rank: $r_{max} = 8$						
DyLoRA (1)	**0.628M**	$93.23_{\pm 0.63}$	$91.58_{\pm 0.69}$	$91.95_{\pm 0.14}$	$90.30_{\pm 0.13}$	91.77
DyLoRA (8)	0.887M	$94.36_{\pm 0.38}$	$91.38_{\pm 0.83}$	$93.00_{\pm 0.32}$	$91.05_{\pm 0.19}$	**92.45**

Table 3: This table compares DyLoRA with compression-based algorithms. As indicated by *, we reported "Fine Tune" and FLOP from their original papers, (Liu et al., 2019) and (Wang et al., 2019). To the best of our knowledge, experiments were conducted under the same experimental setting. We count all the trainable parameters including classifier, unlike LoRA paper (Hu et al., 2021a) which they count only LoRA specific parameters.

Maximum Rank: $r_{max} = 8$								
		Accuracy	F1	Mathews	Accuracy	Accuracy	Pearson	
$b \sim P_B$: *Distribution*	*Updated Parameters*	**SST-2**	**MRPC**	**CoLA**	**QNLI**	**RTE**	**STS-B**	**AVERAGE**
		Rank=8						
Geometric (p=0.15)	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	$93.97_{\pm 0.33}$	$90.84_{\pm 1.15}$	$58.95_{\pm 1.95}$	$92.74_{\pm 0.13}$	$74.80_{\pm 0.90}$	$90.66_{\pm 0.15}$	83.66
	W^b_{dw},W^b_{up}	$93.60_{\pm 0.24}$	$90.50_{\pm 0.42}$	$58.19_{\pm 1.17}$	$92.26_{\pm 0.12}$	$71.91_{\pm 1.74}$	$90.20_{\pm 0.36}$	82.78
Uniform	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	$94.36_{\pm 0.38}$	$91.38_{\pm 0.83}$	$59.51_{\pm 1.84}$	$93.00_{\pm 0.32}$	$77.55_{\pm 0.59}$	$91.05_{\pm 0.19}$	**84.47**
	W^b_{dw},W^b_{up}	$93.69_{\pm 0.41}$	$91.19_{\pm 0.79}$	$58.52_{\pm 0.95}$	$92.47_{\pm 0.18}$	$73.29_{\pm 2.49}$	$90.68_{\pm 0.14}$	83.31
		Rank=1						
Geometric (p=0.15)	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	$93.53_{\pm 0.47}$	$91.36_{\pm 0.72}$	$59.43_{\pm 1.12}$	$92.24_{\pm 0.08}$	$73.65_{\pm 3.55}$	$90.33_{\pm 0.14}$	**83.42**
	W^b_{dw},W^b_{up}	$93.58_{\pm 0.26}$	$90.81_{\pm 0.83}$	$58.55_{\pm 1.13}$	$92.27_{\pm 0.28}$	$68.52_{\pm 11.88}$	$90.60_{\pm 0.31}$	82.39
Uniform	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	$93.23_{\pm 0.63}$	$91.58_{\pm 0.69}$	$57.93_{\pm 2.12}$	$91.95_{\pm 0.14}$	$74.80_{\pm 1.48}$	$90.30_{\pm 0.13}$	83.30
	W^b_{dw},W^b_{up}	$93.51_{\pm 0.49}$	$90.75_{\pm 0.70}$	$56.95_{\pm 1.54}$	$91.70_{\pm 0.28}$	$66.79_{\pm 8.54}$	$89.95_{\pm 0.24}$	81.61

Table 4: Ablation Study - In this experiment, our goal is to demonstrate how the introduced distribution can affect the performance of DyLoRA.

a result, most of the experiments have been conducted in a favorable manner for LoRA.

- **FLOP**: Due to its flexibility, **F**actorized **Low** Rank **Pr**uning (FLOP) (Wang et al., 2019) can be applied to any matrix multiplication and, therefore, can be used to avoid the search in our problem. However, this baseline lacks the dynamic properties of DyLoRA. We used it to show regularization-based techniques' performance and pros and cons.

5.2 LoRA rank selection problem

There is no clear guidance on how to determine the rank for the LoRA algorithm. It is evident in the LoRA paper (Hu et al., 2021a) that the performance of models varies a lot with different ranks (e.g. check Tables 15, and 18 in the LoRA paper), and does not indicate any clear trend. We also observe the same problem in the GLUE benchmark. We may argue that theoretically, the rank with the best performance is always the highest. High ranks, however, introduce additional parameters into the adaptive process and this might be undesirable. In practice, as demonstrated in Table 1, the most effective rank differs depending on the task. For example, based on the MRPC results, the rank with the lowest performance is 16 while the rank with the highest performance is 32. This is different from SST-2, in which rank 1 is the least performing rank and rank 8 is the most effective rank. Many factors can contribute to this difference, including but not limited to the size of the dataset, hyperparameter selections, hardware configurations and the optimization.

5.3 Dynamic low rank adaptation

For example, suppose we have a neural network that we wish to deploy on various devices with different configurations. The use of higher ranks may pose a problem for very sensitive devices as they have a greater number of parameters. Therefore, we must either train several models with different configurations or find the most optimal rank. The cost associated with this is significant, as even in the setting of LoRA, we are required to find the best rank for each task and each device. Using DyLoRA, however, one needs to train one model per task and, as our method is adaptive at inference time, we can deploy it according to our needs. In Table 2, we demonstrate the dynamic properties of

DyLoRA. In order to ensure a fair comparison, all LoRA and DyLoRA models in this table have the same model size, we used the same code and evaluation process, and all models were trained to the same extent. In LoRA, we lose performance when performing inferences for the lower ranks. This occurs because the model has been trained only for rank 8 during training. In DyLoRA, we preserve a high level of performance for lower ranks while competing well with LoRA on rank 8.

Model	**Time**	**SST-2** (r)	**MRPC** (r)
Maximum Rank: $r_{max} = 64$			
LoRA (Search)	$7x$	**95.3**(64)	89.71(64)
DyLoRA (Frozen)	**1x**	94.38(**7**)	**89.95(34)**
Maximum Rank: $r_{max} = 32$			
LoRA (Search)	$6x$	**94.84**(32)	88.73(16)
DyLoRA (Frozen)	**1x**	94.38(**7**)	**89.71(5)**

Table 5: In this table, the search space of rank is larger compared to the previous experiment and the goal is to find the most optimal rank for the low-rank adaptation of a pre-rained RoBERTa-Base. For LoRA (Search), we ran experiments for ranks=1,2,4,8,16,32,64 and we reported the best results. In the Exhaustive Search, one has to search all the ranks from 1 to 64, which means it will cost 64 times more than our proposed method. The lower the rank the better, and the higher the performance is the better.

5.4 Search-free low rank adaptation

The process of selecting a particular rank can be expensive as previously mentioned. In Table 5, we present an experiment that illustrates the costs associated with such a search for LoRA and DyLoRA. As an example, if one naively wanted to search the entire range of ranks (for example, 64 in the experiment), then they would have to train and evaluate 64 distinct models in order to determine the proper rank. It becomes even more expensive if one search the entire rank space. In the case of uniform search, this cost is less, yet still more expensive (7 times in the experiment) than our proposed method. Therefore, for LoRA (Search), we ran experiments for ranks=1,2,4,8,16,32,64 and we reported the best results. The results demonstrate that our proposed method performs competitively at a much lower cost.

5.5 Robustness of DyLoRA

As illustrated in Table 2, DyLoRA is quite robust to randomness and can produce consistently good results due to stable convergence.

5.6 Regularization and Pruning

An alternative method of avoiding the search problem is using regularization/pruning techniques to determine the intrinsic rank of the weight matrix. In this way, we can reduce the number of parameters of the original matrices; however, we will not have a dynamic model during inference. To illustrate the difference between such methods and DyLoRA, we reported the performance of one of these models, FLOP (Wang et al., 2019), in Table 3. FLOP utilizes low-rank factorization to create new matrices representing the original weight matrix. Thus, they will have fewer total parameters but require more trainable parameters to reach a comparable performance to DyLoRA.

5.7 Generative Tasks

In this experiment, we evaluate the performance of our model on different natural language generation (NLG) tasks such as the E2E NLG Challenge (Novikova et al., 2017), DART (Nan et al., 2020) and WebNLG (Gardent et al., 2017). The results of the E2E task are shown in Table 6 and due to the space limit, the results of the other two tasks are demonstrated in Appendix C. The generative tasks demonstrate a similar pattern as the NLU task, showing that our model is able to work well at wider range of ranks compared to LoRA.

5.8 Ablation study

In this subsection, we investigate the impact of two design choices in DyLoRA: first, the new distribution P_B hyper-parameter in our technique; second, the impact of updating W^b_{dw} and W^b_{up} parameters instead of the entire $W_{dw\downarrow b}$ and $W_{up\downarrow b}$. The distribution P_B changes the relative importance of the different ranks during the training process. To examine the impact of the chosen distribution on DyLoRA's performance, we used two distributions, geometric and uniform. As shown in Table 4, the geometric distribution, provides a much better method for optimizing the lower ranks, since it pays much more attention to the lower ranks during training, and uniform distribution will give better performance over all ranks. We chose to use uniform distribution in most of our experiments to avoid adding another hyperparameter which is a requirement of the geometric distribution. Moreover, we demonstrate that it is possible to ensure that the optimization of rank b will not negatively affect the performance of the lower ranks (1 to $b-1$), while

Model (Method)	**Updated Params**	**Trainable Params**	E2E NLG Challenge				
			BLEU	**NIST**	**MET**	**ROUGE-L**	**CIDEr**
			Rank=1				
GPT-2 M (LoRA)		0.09M	3.38	1.18	9.23	18.79	0.12
GPT-2 M (DyLoRA)	W^b_{dw},W^b_{up}	0.09M	$67.92_{\pm 0.20}$	$8.65_{\pm 0.06}$	$44.91_{\pm 0.38}$	$69.07_{\pm 0.32}$	$2.38_{\pm 0.04}$
GPT-2 M (DyLoRA)	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	0.09M	$68.86_{\pm 0.55}$	$8.72_{\pm 0.04}$	$45.81_{\pm 0.40}$	$70.33_{\pm 0.64}$	$2.43_{\pm 0.04}$
			Rank=2				
GPT-2 M (LoRA)		0.19M	46.99	6.39	34.19	56.10	1.27
GPT-2 M (DyLoRA)	W^b_{dw},W^b_{up}	0.19M	$68.81_{\pm 0.49}$	$8.75_{\pm 0.02}$	$45.23_{\pm 0.22}$	$69.81_{\pm 0.30}$	$2.41_{\pm 0.01}$
GPT-2 M (DyLoRA)	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	0.19M	$68.97_{\pm 1.03}$	$8.75_{\pm 0.07}$	$45.88_{\pm 0.55}$	$70.07_{\pm 0.86}$	$2.43_{\pm 0.04}$
			Rank=3				
GPT-2 M (LoRA)		0.29M	63.68	8.46	42.37	65.84	2.24
GPT-2 M (DyLoRA)	W^b_{dw},W^b_{up}	0.29M	$68.41_{\pm 1.00}$	$8.69_{\pm 0.10}$	$45.31_{\pm 0.64}$	$69.75_{\pm 0.69}$	$2.42_{\pm 0.02}$
GPT-2 M (DyLoRA)	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	0.29M	$69.33_{\pm 0.26}$	$8.76_{\pm 0.05}$	$46.19_{\pm 0.22}$	$70.56_{\pm 0.43}$	$2.46_{\pm 0.01}$
			Rank=4				
GPT-2 M (LoRA)		0.39M	69.88	8.81	46.81	72.10	2.53
GPT-2 M (DyLoRA)	W^b_{dw},W^b_{up}	0.39M	$68.36_{\pm 0.41}$	$8.70_{\pm 0.02}$	$45.46_{\pm 0.56}$	$69.91_{\pm 0.50}$	$2.43_{\pm 0.01}$
GPT-2 M (DyLoRA)	$W_{dw\downarrow b}$,$W_{up\downarrow b}$	0.39M	$69.19_{\pm 0.43}$	$8.75_{\pm 0.03}$	$46.26_{\pm 0.47}$	$70.78_{\pm 0.63}$	$2.46_{\pm 0.02}$
			Fine-Tune				
GPT-2 M (FT)*		354M	68.2	8.62	46.2	71.0	2.5

Table 6: For all metrics, higher is better. Rows with * have been reported based on the LoRA paper. Unlike (Hu et al., 2021a), we included the classifier number of parameters in our trainable parameters count.

performing reasonably well. As mentioned, this can be accomplished by only updating the unique parameters associated with rank r that do not overlap with lower ranks.

In addition, in Table 7, we demonstrate the result of using our individual loss (Eq. 9) vs. the nested dropout original objective function in an equal setting. As shown, our proposed objective function is both effective and efficient. Furthermore, it is important to note that the summation loss is not scalable when many ranks are involved. We also discussed the time complexity of LoRA and DyLoRA in Appendix A.

Maximum Rank: $r_{max} = 8$		
Loss	**Training Time**	**CoLA**
$\mathcal{L}^{\mathcal{DY}}_{\downarrow b}$	**645.82s**	52.64
$\sum p_B(b)\mathcal{L}^{\mathcal{DY}}_{\downarrow b}$	1175.69s	**54.12**

Table 7: This experiment shows the impact of choosing individual loss vs. summation loss functions on our training. The average performance across all possible ranks (1,2,...,8) is reported. For summation loss to be computationally more feasible, smaller epochs were chosen. A total of seven GPUs were used in this experiment.

6 Conclusion

In this paper, we presented our solution DyLoRA to address two problems in low-rank adapters regarding rank selection and making them dynamic. We showed that DyLoRA can select the rank without requiring multiple re-training and is able to make LoRA dynamic at inference time. As a result, we can avoid the process of searching for the most optimal ranks for many real-life scenarios. It has been demonstrated that DyLoRA performance is comparable with LoRA, yet we can support a wider range of ranks without adding additional time and effort.

Limitations

According to LoRA (Hu et al., 2021a), a proper choice of the scalar can improve the results. In order to determine what is the best choice, further investigation is required. Despite our demonstration that uniform distribution can be as effective as specific geometric distribution, further investigation is necessary to evaluate the effect of different distributions on different downstream tasks. As shown in this paper, our algorithm works over a wide range of ranks, but further research is needed to understand the impact of choosing a particular range.

7 Acknowledgement

We would like to use DyLoRA with Mindspore[2], which is a new framework for deep learning computing.

---

[2] mindspore.cn

References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. 2020. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR.

Patrick Chen, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. 2021. Drone: Data-aware low-rank compression for large nlp models. *Advances in neural information processing systems*, 34:29321–29334.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Utku Evci, Max Vladymyrov, Thomas Unterthiner, Bart van Merriënboer, and Fabian Pedregosa. 2022. Gradmax: Growing neural networks using gradient information. *arXiv preprint arXiv:2201.05125*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems*, 33:9782–9793.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019a. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019b. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. 2019. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021a. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021b. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2493–2504, Online. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Lei Li, Yankai Lin, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. 2021. Dynamic knowledge distillation for pre-trained language models. *arXiv preprint arXiv:2109.11295*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabsa. 2021. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv preprint arXiv:2110.07577*.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, et al. 2020. Dart: Open-domain structured data record to text generation. *arXiv preprint arXiv:2007.02871*.

Matan Ben Noach and Yoav Goldberg. 2020. Compressing pre-trained language models by matrix decomposition. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 884–889.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. ALP-KD: attention-based layer projection for knowledge distillation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13657–13665. AAAI Press.

Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. 2020. Fully quantized transformer for machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1–14.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. Mate-kd: Masked adversarial text, a companion to knowledge distillation. *arXiv preprint arXiv:2105.05912*.

Oren Rippel, Michael Gelbart, and Ryan Adams. 2014. Learning ordered representations with nested dropout. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1746–1754, Bejing, China. PMLR.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. *arXiv preprint arXiv:2201.03514*.

Marzieh S Tahaei, Ella Charlaix, Vahid Partovi Nia, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021. Kroneckerbert: Learning kronecker decomposition for pre-trained language models via knowledge distillation. *arXiv preprint arXiv:2109.06243*.

Chaofan Tao, Lu Hou, Wei Zhang, Lifeng Shang, Xin Jiang, Qun Liu, Ping Luo, and Ngai Wong. 2022. Compression of generative pre-trained language models via quantization. *arXiv preprint arXiv:2203.10705*.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2019. Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*.

Qiaolin Xia, Haoyang Huang, Nan Duan, Dongdong Zhang, Lei Ji, Zhifang Sui, Edward Cui, Taroon Bharti, and Ming Zhou. 2021. Xgpt: Cross-modal generative pre-training for image captioning. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 786–797. Springer.

A Time complexity

The training time for DyLoRA is comparable to that of LoRA trained once on a specific rank. Thus, when searching the rank space for LoRA, we need to train it multiple times, whereas our method does not require searching the ranks. Accordingly, DyLoRA's relative time complexity is inversely proportional to the number of possible ranks for which the LoRA model must be searched. In MRPC, DyLoRA (for all the ranks) and LoRA (only on a single rank 8) require a total training time of 408.39 seconds and 399.95 seconds, respectively. Consequently, when we need to train eight LoRA models (Rank=1,2,...,8), it will result in a cost of 399.95*8=3199.6s, compared to the training time of our model, which is only 408.39 seconds. A more efficient implementation of our algorithm may result in a better time complexity.

B Hyperparameters

We did not use any parameter tuning nor MNLI trick (initializing some down-streams tasks from MNLI checkpoint instead of pretrained weights). Therefore, we fine-tuned all the datasets from original pretrained weights. We simply followed a unified hyper-parameters for all different experiments. Unlike LoRA (Hu et al., 2021a) which reported the median over 5 random seeds, we reported the mean and standard deviation over 5 random seeds. See the details in Table 8.

C GPT Experiments

A summary of the additional experiments that have been conducted to demonstrate the effectiveness of our proposed method for the task of language generation is provided in Table 9.

Model	**Parameter**	**Value**
RoBERTa-Base	Optimizer	AdamW
	Warmup Ratio	0.06
	LR Scheduler	Linear
	Batch Size	32
	Epochs	30
	Learning Rate (LR)	4e-4
	Weight Decay	0.1
	LoRA Config	$r_q = r_v = 8$ (unless otherwise mentioned)
	LoRA α	16
	Max Sequence Length	512
	Seeds	10, 42, 4242, 10, 1010
	GPU	Tesla V100-PCIE-32GB
GPT Medium	Optimizer	AdamW
	Adam Beta2	0.999
	Warmup Steps	500
	Clip	0.0
	LR Scheduler	Linear
	Batch Size	8
	Epochs	5
	Learning Rate (LR)	2e-4
	Weight Decay	0.01
	Correct Bias	True
	LoRA Dropout	0.1
	Lable Smooth	0.1
	LoRA Config	$r_q = r_v = 4$
	LoRA α	32
	Seeds	10, 42, 4242
	GPU	Tesla V100-PCIE-32GB

Table 8: All the hyperparameters that have been used throughout our study.

Model (Method)	**Trainable Params**	DART		WebNLG	
		BLEU↑	**TER↓**	**BLEU↑**	**TER↓**
Rank=1					
GPT-2 M (LoRA)	0.09M	0.71	0.49	2.80	1.18
GPT-2 M (DyLoRA-Frozen)	0.09M	$44.48_{\pm0.11}$	$0.49_{\pm0.00}$	$52.09_{\pm0.10}$	$0.40_{\pm0.01}$
GPT-2 M (DyLoRA)	0.09M	$44.77_{\pm0.17}$	$0.49_{\pm0.01}$	$53.04_{\pm0.07}$	$0.40_{\pm0.00}$
Rank=2					
GPT-2 M (LoRA)	0.19M	15.90	0.48	26.58	0.67
GPT-2 M (DyLoRA-Frozen)	0.19M	$45.04_{\pm0.14}$	$0.48_{\pm0.01}$	$52.74_{\pm0.31}$	$0.40_{\pm0.01}$
GPT-2 M (DyLoRA)	0.09M	$46.05_{\pm0.31}$	$0.48_{\pm0.00}$	$54.32_{\pm0.09}$	$0.39_{\pm0.01}$
Rank=3					
GPT-2 M (LoRA)	0.29M	35.84	0.47	43.61	0.47
GPT-2 M (DyLoRA-Frozen)	0.29M	$45.22_{\pm0.14}$	$0.49_{\pm0.01}$	$53.03_{\pm0.55}$	$0.40_{\pm0.00}$
GPT-2 M (DyLoRA)	0.29M	$46.68_{\pm0.36}$	$0.48_{\pm0.01}$	$54.48_{\pm0.05}$	$0.39_{\pm0.00}$
Rank=4					
GPT-2 M (LoRA)	0.39M	47.10	0.46	55.57	0.39
GPT-2 M (DyLoRA-Frozen)	0.39M	$45.56_{\pm0.33}$	$0.48_{\pm0.00}$	$53.03_{\pm0.01}$	$0.40_{\pm0.00}$
GPT-2 M (DyLoRA)	0.39M	$46.56_{\pm0.42}$	$0.48_{\pm0.01}$	$54.48_{\pm0.45}$	$0.39_{\pm0.00}$
Fine-Tune					
GPT-2 M (FT)*	354M	46.2	0.46		

Table 9: Rows with * have been reported from the LoRA paper. (Hu et al., 2021a).

Conversational Emotion-Cause Pair Extraction with Guided Mixture of Experts

DongJin Jeong
Sungkyunkwan University
Department of Artificial Intelligence
South Korea
`jdjin3000@g.skku.edu`

JinYeong Bak
Sungkyunkwan University
Department of Artificial Intelligence
South Korea
`jy.bak@skku.edu`

Abstract

Emotion-Cause Pair Extraction (ECPE) task aims to pair all emotions and corresponding causes in documents. ECPE is an important task for developing human-like responses. However, previous ECPE research is conducted based on news articles, which has different characteristics compared to dialogues. To address this issue, we propose a Pair-Relationship Guided Mixture-of-Experts (PRG-MoE) model, which considers dialogue features (e.g., speaker information). PRG-MoE automatically learns relationship between utterances and advises a gating network to incorporate dialogue features in the evaluation, yielding substantial performance improvement. We employ a new ECPE dataset, which is an English dialogue dataset, with more emotion-cause pairs in documents than news articles. We also propose Cause Type Classification that classifies emotion-cause pairs according to the types of the cause of a detected emotion. For reproducing the results, we make available all our code and data[1].

1 Introduction

With increased interest in developing human-like responses, it is crucial to determine the cause of a given emotion. As part of such interest, there is a surge of research activities that analyze the cause of emotions (Yan et al., 2021; Turcan et al., 2021; Li et al., 2022a). Recently, Poria et al. (2021) presents RECCON, a new dataset for Emotion Cause Extraction (ECE) task in dialogue. ECE is a task to find a clause that contains the cause of an annotated emotion in a clause of a given document. However, ECE is limited in that the model requires manually annotated emotions.

To overcome the limitation of ECE (Lee et al., 2010), Xia and Ding (2019) suggest Emotion Cause Pair Extraction (ECPE) task, which automatically predicts emotion clauses in a given document and identifies their corresponding causes. They also build a new ECPE corpus from Chinese news articles. Since ECPE and the dataset were proposed, it has attracted the interest of numerous researchers Ding et al. (2020a,b); Wei et al. (2020); Fan et al. (2020); Cheng et al. (2020); Chen et al. (2020a,b).

# of Emotion-Cause Pairs	ECPE-news (Xia and Ding, 2019)	ECPE-D
1	1,746 (**89.77%**)	9 (**0.80%**)
2	177 (**9.10%**)	19 (**1.69%**)
≥ 3	22 (**1.13%**)	1,094 (**97.50%**)

Table 1: The amount of emotion-cause pairs in a document compared with the ECPE-news corpus (Xia and Ding, 2019) and ECPE-D corpus. ECPE-D is a dialogue dataset reconstructed based on RECCON (Poria et al., 2021). On average, ECPE-D corpus has more emotion-cause pairs than ECPE-news corpus.

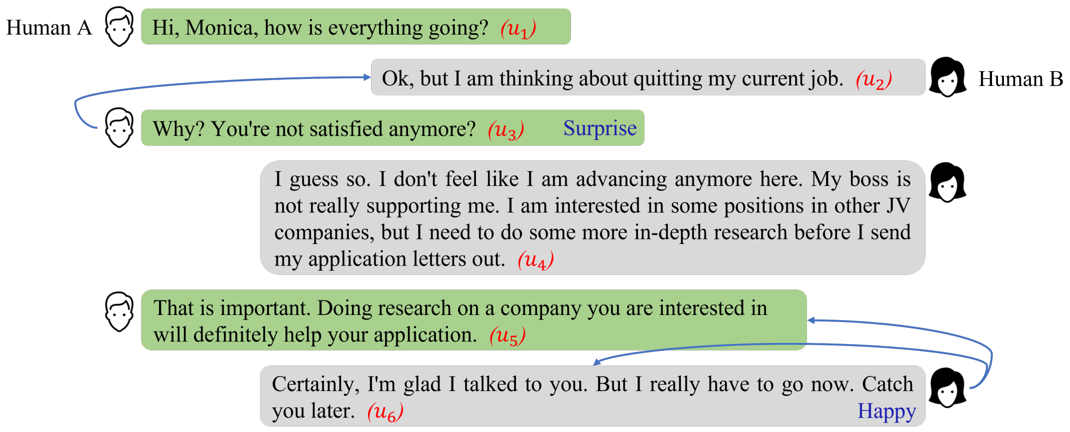
However, ECPE in dialogues is different from ECPE in news articles. A dialogue is an interaction between two or more people, while a news article describes a fact. So, dialogues contains meta information such as the speakers, which is one of the most important information in understanding dialogues. In addition, dialogues contain more diverse and emotional expressions, and emotions change as the dialogue progresses, creating even more emotion-cause pairs. This makes the task of ECPE even more challenging. Table 1 shows that most documents in the current ECPE news corpus have only one emotion-cause pair per document, whereas most of the dialogues have multiple emotion-cause pairs. So, we employ RECCON, an English dialogue dataset (Poria et al., 2021) as a new ECPE dataset. We reconstruct RECCON suitable for the ECPE task and we call this dataset **ECPE-D**. Figure 1 shows an example of ECPE-D.

In this paper, we propose a Pair-Relationship Guided Mixture-of-Experts (PRG-MoE) model, which considers dialogue features (e.g., speaker information) in ECPE. We employ Mixture-Of-Experts (MoE) (Eigen et al., 2014) to customize

---

[1] `https://github.com/jdjin3000/PRG-MoE`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3288–3298
May 2-6, 2023



Hi, Monica, how is everything going? (u_1)

Ok, but I am thinking about quitting my current job. (u_2)

Why? You're not satisfied anymore? (u_3)

I guess so. I don't feel like I am advancing anymore here. My boss is not really supporting me. I am interested in some positions in other JV companies, but I need to do some more in-depth research before I send my application letters out. (u_4)

That is important. Doing research on a company you are interested in will definitely help your application. (u_5)

Certainly, I'm glad I talked to you. But I really have to go now. Catch you later. (u_6)

Figure 1: Example of ECPE task. u_3 has a *surprise* emotion because Human A learns that his colleague is thinking of quitting her job (u_2). Human B feels happy at u_6 because she thinks her conversation with Human A is interesting (u_6) and has obtained good advice from Human A (u_5). We can extract the emotion-cause pair (u_3, u_2), (u_6, u_5), (u_6, u_6) from this dialogue. Blue line indicates the cause of the emotion utterance.

the experts in the relationship between utterances. It is important to consider the relationship between utterances, since it helps us grasp the emotional flow from the conversation history or understand emotions (or its causes) through other speakers. PRG-MoE automatically learns the relationship between utterances and advises gating networks to incorporate dialogue features, which yields excellent performance. We evaluate PRG-MoE and other models using ECPE-D dataset and show that PRG-MoE outperforms other models.

Furthermore, we propose a multi-class classification task that identifies cause types of emotion-cause pairs in a dialogue - **ECPE-CT**. An understanding of the cause types is beneficial especially for empathetic response generation (Gao et al., 2021). ECPE-CT helps to generate more specific empathetic responses rather than simple reactions for all kinds of context such as "good luck" by understanding several cause types. Knowing the cause types of the last utterance by the user can help the agent to comprehend the context and obtain the implicit feedback from the user. There are three cause types in ECPE-D: no-context, interpersonal, and self-contagion. Type is categorized depending on 1) into from which speaker the cause of the emotion originated, and 2) whether the cause appears in the current utterance or not.

We performed ECPE-CT tasks under various models; PRG-MoE outperformed other baselines in identifying not only emotion-cause pairs but also types of causes in the pairs.

Our contributions are as follows:

- We propose a new ECPE task in dialogues, and provide related dataset - **ECPE-D**.
- We present **PRG-MoE**, a new approach that outperforms other models in ECPE-D.
- We propose a new Cause Type Classification task **(ECPE-CT)** that helps categorizing the type of a cause for an emotion.

2 Related Work

Xia and Ding (2019) propose the ECPE task, which predicts an emotion clause and extracts a corresponding cause in a given document. The authors construct a new ECPE corpus from Chinese news ECE corpus (Gui et al., 2016). They also propose a two-step approach: a pipeline structure consisting of emotion/cause clause extraction and an emotion-cause pairing. However, the two-step approach in a pipeline structure has limitation in that errors cannot be propagated to the entire model. PRG-MoE performs end-to-end pair extraction so it avoids inaccurate inference originating from a pipeline structure.

Subsequent ECPE research suggests 2D transformer (Ding et al., 2020a), sliding window (Ding et al., 2020b) and graph neural network (Chen et al., 2020b; Wei et al., 2020). However, these approaches do not consider meta information such as speakers. Speaker information is a factor that improves performance in dialogue-related tasks. Several studies report improved performance in emotion recognition by taking into account speaker information (Zhang et al. (2019), Li et al. (2020), Bao et al. (2022)). In addition, methods that considers speakers obtain higher performance in conversation related tasks (Li et al. (2022b), Bak and Oh (2020)). PRG-MoE adopts Mixture-of-Experts for incorporating speaker information as a feature in pair extraction. Also, previous ECPE research considers only the existence of emotion in a clause. PRG-MoE suggests a method where a combination of speaker information and type of emotion in an utterances is used.

3 Task Definition

This section describes the definition of ECPE task in a dialogue. The input is a dialogue $D = \{u_1, ..., u_n\}$ that contains multiple utterances between two people, where u_i is i-th utterance in a dialogue and consists of token sequence t_i, speaker indicator $s_i \in \{0, 1\}$ and emotion information e_i.

Objective of the task is to extract a set of emotion-cause utterance pairs $\{..., (u_i, u_j), ...\}$, where u_i is an emotion utterance and u_j is a cause utterance in a pair.

4 Pair-Relationship Guided Mixture-of-Experts

We propose a Pair-Relationship Guided Mixture-of-Experts (PRG-MoE) model that adopts a Mixture-of-Experts module. Figure 2 shows the overall architecture. PRG-MoE consists of three modules: utterance representation construction (§4.1), emotion-cause pair candidate extraction (§4.2) and mixture-of-experts based emotion-cause pair classification (§4.3).

4.1 Utterance Representation Construction

First, PRG-MoE creates a representation of each utterance. Input is a dialogue $D = \{u_1, ..., u_n\}$, where i-th utterance $u_i = (t_i, s_i)$ contains token sequence t_i and speaker indicator s_i. We use BERT (Devlin et al., 2019) for constructing token sequence representation. We surround each token sequence with pre-defined special tokens ([CLS], [SEP]), $t'_i = \{[CLS], w_{i1}, ..., w_{ik}, [SEP]\}$, where w_{ik} is k-th token in i-th utterance's token sequence. [CLS] token is used for generating representation for classification tasks. [SEP] token is used to denote the end of a sentence. We obtain the utterance's representation h_i via BERT, which is the final hidden state of [CLS].

$$h_i = BERT(t'_i) \tag{1}$$

Emotion Classification PRG-MoE performs emotion classification not only to obtain emotion utterance candidates for emotion-cause pairs, but also to convey emotion information in utterance representation. Emotion classification is performed by feeding a token sequence representation h_i into a Feed-Forward Neural Network (FFNN) layer. We can get the emotion prediction $\hat{e}_i$.

$$\hat{e}_i = Softmax(W^e h_i + b^e), \tag{2}$$

where W^e is a weight and b^e is a bias of the emotion classification layer, respectively.

Utterance Representation We use the concatenation of token sequence representation, emotion prediction and speaker information as utterance representation.

$$u_i = h_i \oplus \hat{e}_i \oplus s_i \tag{3}$$

4.2 Emotion-Cause Pair Candidate Extraction

To extract emotion-cause pair candidates, PRG-MoE needs to pair emotion utterance candidates and cause utterance candidates. The candidate pair x_{ij} is created by concatenating two utterances.

$$x_{ij} = u_i \oplus u_j, \tag{4}$$

where u_i is a non-neutral emotion utterance representation and u_j is a cause utterance candidate representation ($j \in \{1, ..., i\}$).

We assume two properties of the pairs. First, only non-neutral emotion utterances can be emotion utterance candidates. This is because ECPE tries to find the cause of emotions that occurred during a conversation. Second, PRG-MoE assumes that the cause of an emotion exists in previous or present utterances, since future utterances are not known to the speakers.

Window-constrained Strategy For computing efficiency, we adopt a window-constrained strategy in ECPE task (Ding et al., 2020a). In a given dialogue $D = \{..., u_i, ...\}$, where u_i is a non-neutral emotion utterance, u_i's cause candidates $\{u_{i-|w|+1}, ..., u_i\}$ are selected up to the predefined window size $|w|$ distance.

4.3 Mixture-of-Experts based Pair Classification

Pair-Relationship in ECPE In emotion-cause pairs, a specific relationship is formed depending

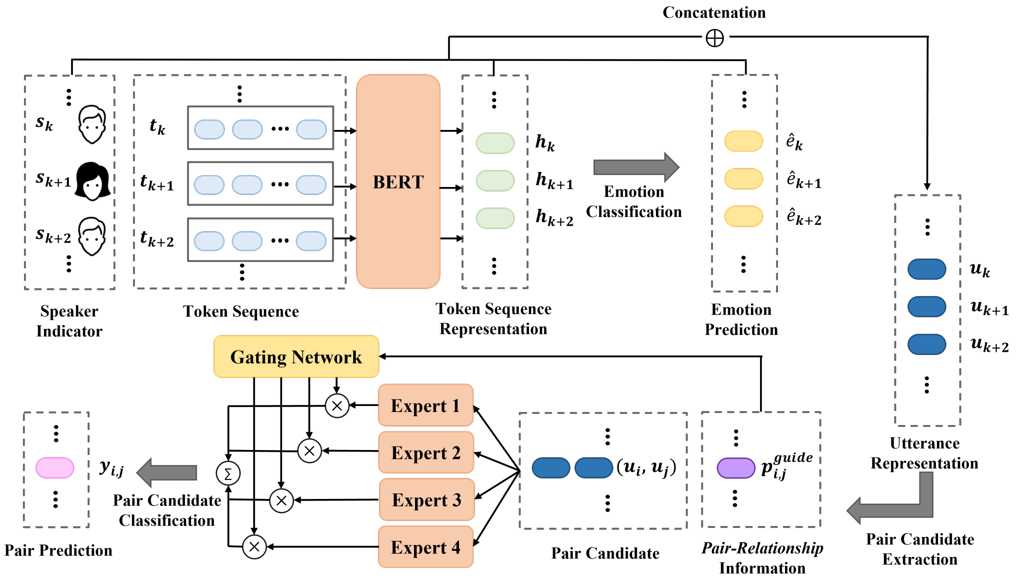


Figure 2: Overall Architecture of PRG-MoE consists of three parts: utterance representation construction, emotion-cause pair candidate extraction, and mixture-of-experts based pair classification. Token sequences in dialogue are converted into semantic representation by BERT. The concatenation of speaker indicator, token sequence representation, and emotion prediction information is used as an utterance representation. In the emotion-cause pair candidate extraction, only non-neutral utterances are considered to be emotion utterances. Pair candidates are routed to proper experts using routing probability. The routing probability consists of the gating network probability g_θ and *pair-relationship* probability p^{guide}.

on speaker or emotion relationship. We call these relationship *pair-relationship*. Depending on who the speaker is of an utterance u_i, u_j in an emotion-cause utterance pair x_{ij} and what emotion the utterance contains, we can identify the following four categories.

- **Same speaker - Same emotion**: This is a case where the utterances in a pair belong to the same speaker and have the same emotions, such as maintaining emotional state.

- **Same speaker - Different emotion**: This is a case where the utterances in a pair belong to the same speaker but have different emotions. For example, an emotion can appear in the second utterance, following the first utterance where the speaker talks neutrally about what could be the cause of the emotion that arises in the second utterance. Also, the speaker can have multiple emotions occur simultaneously in one utterance, such as ambivalence (Larsen and Mcgraw, 2011). So, the cause of one emotion can trigger different emotions.

- **Different speaker - Same emotion**: This is a case where the utterances in a pair belong to different speakers but share the same emotion, such as sharing the emotion of empathy.

- **Different speaker - Different emotion**: This is a case where the utterances in a pair belong to different speakers and each have different emotions. For example, a speaker's utterance triggers an emotion in the other speaker's utterance. This case is similar to the Same speaker - Different emotion case, but the subject of the cause utterance is another speaker.

pair-relationship is constructed based on the predicted emotions, not the ground-truth emotion information.

Guided-MoE Method We are inspired by the mixture-of-experts (MoE) method to consider *pair-relationship*. MoE is the process of utilizing multiple experts for a specific task. The expert is a trainable neural network. The gating network determines which expert is suitable for a given input, and this mechanism automatically enhances the expertise of experts through learning.

However, there is no guarantee that the pure MoE learns *pair-relationship*. So, we guide each expert to have expertise in *pair-relationship*. We combine the decision of the gating network and *pair-relationship* for routing emotion-cause pair candidates to proper experts.

MoE consists of k experts $\{f_\theta^1, ..., f_\theta^k\}$ and gating network g_θ. Experts and gating network get a set of emotion-cause pair candidates $\{..., x_{ij}, x_{ii}, ...\}$ as input. Experts return the emotion-cause pair classification prediction $f_\theta(x_{ij})$, respectively. Gating network g_θ returns the routing probability $g_\theta(x_{ij})$, where $g_\theta(x_{ij})$ is a distribution over k experts that sums to 1.

To guide the routing probability to consider *pair-relationship*, PRG-MoE first creates one-hot label that represents the category of *pair-relationship* p_{ij}^{guide}. PRG-MoE routes an input pair representation x_{ij} by combining $g_\theta(x_{ij})$ and p_{ij}^{guide}. For combination, the number of experts should be the same as the number of *pair-relationship*.

$$p_{ij} = (1 - \lambda) \times g_\theta(x_{ij}) + \lambda \times p_{ij}^{guide}, \quad (5)$$

where p_{ij} is a distribution over k experts that sums to 1.

The output of PRG-MoE is as follows:

$$y_{ij} = \sum_{n=1}^{k} p_{ij}^n f_\theta^n(x_{ij}), \quad (6)$$

where k is the number of the *pair-relationship* information as one expert is assigned to each category.

The loss function of PRG-MoE consists of emotion classification loss and emotion-cause pair classification loss that are focal loss (Lin et al., 2017). Essentially, ECPE task faces the challenge of class imbalance since it has a few positive samples among pair candidates. The adoption of the focal loss alleviates this issue by balancing the weight assigned to minority classes, facilitating the learning process (Wang et al., 2022).

5 Experiments

5.1 Settings

Dataset We use the RECCON dataset (Poria et al., 2021) for experiments. RECCON is a dataset for ECE task that finds a corresponding cause for an utterance with a given emotion. It consists of DailyDialog (Li et al., 2017) and IEMOCAP (Busso et al., 2008), and the authors additionally annotated the cause of an emotion and type of the cause.

We reconstruct a corpus for ECPE in dialogues, named **ECPE-D** from RECCON. We also call DailyDialog data in ECPE-D as **ECPE-D-DD**, and IEMOCAP data in ECPE-D as **ECPE-D-IE**. RECCON has several cause type classes. Among them, there is a cause type called “hybrid” which encompasses both "inter-personal" and "self-contagion" causes. We separate “hybrid” type into “inter-personal” and “self-contagion” and reannotate in dialogues to clarify information associated with the pair label by making multiple single pairs for cause type classification. In addition, RECCON has annotated cause information for each utterance. To facilitate the ECPE task, we add the emotion-cause pair label per dialogue.

Table 2 shows basic statistics of ECPE-D. We split the ECPE-D-DD as 80/10/10 for training/validation/test. We use ECPE-D-IE as test data only. This is because it has fewer dialogues than ECPE-D-DD and we can show the robustness of ECPE models on different domain dataset. IEMOCAP has *frustration* and *excited* emotion labels that are not in DailyDialog. So, we map *frustration* and *excited* to *sad* and *happy*, respectively.

For statistically significant results, we conduct a total of five experiments using randomly split data, and report the average result.

Approach	ECPE-D-DD	ECPE-D-IE
# of Dialogues	1,106	16
Avg. of Dialogue length	10	42
# of *no-context* pair	3,370	243
# of *inter-personal* pair	3,796	365
# of *self-contagion* pair	1,958	445
Avg. of emotion-cause pairs	8	66

Table 2: Characteristics of the ECPE-D Dataset. ECPE-D-IE is uses only as test dataset. The two datasets differ significantly in their properties.

Baselines We select following models as a compared approach. For a fair comparisons, language models of all models are fixed with a *bert-base-cased* from huggingface[2] (Wolf et al., 2020).

- **ECPE-2D** (Ding et al., 2020a) proposes a method of expressing the emotion-cause pairs by a two-dimensional representation scheme. They use a window-constrained method to restrict the scope of the search for extracting emotion-cause pairs.

- **ECPE-MLL** (Ding et al., 2020b) defines an ECPE task as a multi-label learning problem. ECPE-MLL first assumes that all utterances

---

[2] `https://github.com/huggingface`

Test Dataset	ECPE-D-DD						ECPE-D-IE					
Approach	Emotion-Cause Pair Extraction			Emotion Extraction			Emotion-Cause Pair Extraction			Emotion Extraction		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
ECPE-2D	49.34	47.37	48.34**	71.91	**77.16**	**74.44**	45.17	12.37	19.42**	78.16	**49.16**	**60.36**
ECPE-MLL (avg-pair)	53.58	39.58	45.53**	68.97	73.64	71.23	47.44	1.63	3.15**	83.82	6.85	12.66
ECPE-MLL (or-pair)	50.71	43.56	46.86**	68.97	73.64	71.23	43.41	2.17	4.13**	83.82	6.85	12.66
Rank-Emotion-Cause	58.09	10.49	17.77**	**75.85**	20.17	31.87	**65.91**	1.10	2.09**	**91.75**	3.24	6.26
RECCON-BERT	49.31	33.19	39.68*	-	-	-	46.52	4.33	7.92*	-	-	-
PRG-MoE	**58.95**	**55.67**	**57.26**	71.76	76.09	73.86	51.95	**20.02**	**28.90**	85.58	43.06	57.29

Table 3: Performance of PRG-MoE and baseline models for ECPE-D. All models are trained with only ECPE-D-DD. PRG-MoE outperforms all other models in the emotion-cause pair extraction. We also test models in ECPE-D-IE to validate the models in an environment different from the train dataset. Despite the differences, PRG-MoE outperforms all other models. We run the statistical significance test for the F1-Score in the Emotion-Cause Pair Extraction task. PRG-MoE shows statistically significant difference in performance than baselines (**: p < 0.0001, *: p <0.001).

are emotion utterances, and finds corresponding cause utterances; then, assumes that all utterances are cause utterances, and finds corresponding emotion utterances. There are two ways for the ECPE-MLL to identify emotion-cause pairs; avg-pair and or-pair. The avg-pair method identifies a match as a pair when both the cause and emotion utterances select each other as their match. The or-pair method identifies an emotion-cause pair even if only one side selects the other as their match (i.e. the cause utterance c_1 may identify the emotion utterance e_1 as its match, while e_1 selects a different cause utterance c_2 as its match. There can be two cause-emotion pairs identified through the or-pair method; pair (e_1, c_1) and pair (e_1, c_2).

- **Rank-Emotion-Cause** (Wei et al., 2020) is a method that ranks candidates for emotion-cause pairs and filters them using a sentiment word lexicon. Since the prior lexicon is developed for Chinese data, we adapt it using the Loughran-McDonald sentiment lexicon (Loughran and McDonald, 2011) for testing ECPE-D.
- **RECCON** (Poria et al., 2021) uses RoBERTa with a classification layer for ECPE. They claim that the simple language model outperforms other ECPE models. For fair comparisons, we set the language model to BERT. We denote the model RECCON-BERT.

Evaluation Metrics We follow the evaluation metrics from previous research (Xia and Ding, 2019); we use precision, recall, and F1 score as metrics.

5.2 Results

Table 3 shows experimental results for the ECPE task of PRG-MoE and baseline methods with ECPE-D-DD. There is no difference between the PRG-MoE and other models in terms of Emotion Extraction performance. This is because Emotion extraction is performed using only BERT. However, even though it shows similar emotion extraction performance, PRG-MoE outperforms in the pair extraction performance, which is the main goal of our study. The usage of speaker information and type of emotion makes PRG-MoE more suitable to extract emotion-cause pairs in a dialogue than other baselines.

ECPE-MLL predicts emotion-cause pairs utilizing two methods; avg-pair and or-pair. Or-pair shows better recall and f1-score than avg-pair. This is because or-pair predicts the emotion-cause pair optimistically, whereas avg-pair satisfies two indicators for predicting pairs. It gives or-pair a wider search space than avg-pair, making the or-pair more advantageous in finding pairs in emotionally-rich environment and thus to have better performance than avg-pair.

Rank-Emotion-Cause shows low performance. It has two stages for extracting pairs. First, Rank-Emotion-Cause ranks pair candidates and chooses the first pair. Second, the model determines if there is a sentiment word among the other unselected candidate pairs using a sentiment word dictionary and when found, selects it. However, there are many cases where emotions are expressed without explicit expression of emotions in an utterance. Dictionary-based emotion detection makes it hard to capture implicit expression. Dictionary-based emotion detection of Rank-Emotion-Cause captures 1,764 utterances out of 6,384 emotion ut-

λ	Speaker+Emotion Guide			Emotion Guide			Speaker Guide		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
0.0 (Pure MoE)	51.63	55.98	53.72	52.07	54.79	53.40	52.07	54.79	53.40
0.2	53.25	54.98	54.10	52.13	**55.33**	53.68	53.78	56.28	55.00
0.4	56.49	55.20	55.84	54.09	53.92	54.00	54.40	**58.95**	**56.58**
0.6	**58.95**	55.67	**57.26**	54.09	53.56	53.82	56.31	56.34	56.32
0.8	55.53	**57.14**	56.32	53.92	54.61	**54.27**	**58.53**	53.99	56.17
1.0 (Pure guide)	58.07	54.64	56.30	**54.16**	52.91	53.52	54.77	58.40	56.53

Table 4: A study on mixing ratio for routing probability and *pair-relationship* information in guided-MoE method. When λ is 0, it means the pure Mixture-of-Experts method and when λ is 1, it means the pure guide information from *pair-relationship*. Above experiments prove that guided-MoE method is superior to pure MoE or pure guide for experts.

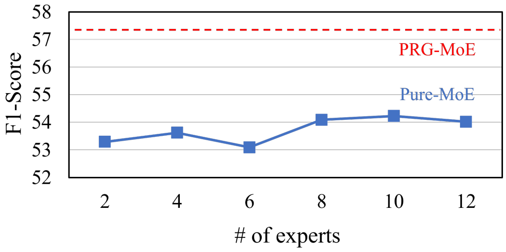


Figure 3: Ablation study for the number of experts in the mixture-of-experts method. We set PRG-MoE with $\lambda = 0$ for experimenting pure MoE method. Above experiments show no meaningful difference without the *pair-relationship* probability p^{guide}.

terances in ECPE-D. This method is unfavorable for extracting multiple pairs.

RECCON argues that the language model with a simple classification layer outperforms earlier ECPE models. They experiment with emotion labels, candidate pairs, and dialogue context as input. However, there are two problems here: First, they exclude a scenario in which emotional information is incorrect. So, the comparison is not fair because ECPE approaches use emotional information that they predict. Second, while ECPE should consider all possible pairs for a given text, RECCON is tested using a dataset in which positive and negative samples are appropriately mixed. We re-evaluate the model by presenting the same emotion label predicted by PRG-MoE to RECCON for accurate comparison. RECCON-BERT shows low performance compared to when it has true emotion labels. This means that RECCON-BERT, unlike other ECPE models, does not have a structure that operates robustly with an inaccurate emotion.

Table 3 also shows the experimental results for ECPE-D-IE. All models are learned with ECPE-D-DD. This experiment is conducted to evaluate how they perform on out-of-domain data. Besides, as shown in Table 2, ECPE-D-IE has about eight times more emotion-cause pairs in a dialogue than the trained data, so the difficulty of pair extraction becomes extremely high. These adverse conditions make the models have poor performances. PRG-MoE shows robust performance compared to other models.

Concatenated element	Emotion-Cause Pair Extraction		
	Precision	Recall	F1-Score
h	58.27	55.82	57.02
$h \oplus \hat{e}$	56.91	55.88	56.39
$h \oplus s$	55.41	**59.04**	57.16
$h \oplus \hat{e} \oplus s$	**58.95**	55.67	**57.26**

Table 5: Ablation study for utterance representation components. It is performed by PRG-MoE. h, $\hat{e}$ and s mean token sequence representation, emotion prediction and speaker indicator, respectively. Above results show that giving information through concatenation has a positive effect.

Approach	Emotion-Cause Pair Extraction		
	Precision	Recall	F1-Score
w/o window-constraint	43.91	43.08	43.49
with window-constraint	**58.95**	**55.67**	**57.26**

Table 6: Ablation study of window-constraint method.

6 Discussion

6.1 Effects on Pair-Relationship Information

The main idea of PRG-MoE is to combine the decision of the gating network and *pair-relationship* information. For evaluating effects on the mixing ratio λ in Guided-MoE, we set λ from 0 (pure Mixture-of-Experts) to 1 (pure guiding *pair-relationship* information). Table 4 shows the proper

Approach	no-context			inter-personal			self-contagion			weighted average		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
ECPE-2D	53.23	60.88	56.80	51.70	42.18	46.46^{**}	36.10	14.72	20.92	48.98	43.42	46.03^{**}
PRG-MoE	**54.66**	**62.30**	**58.23**	**62.22**	**59.11**	**60.63**	**39.99**	**17.19**	**24.05**	**55.43**	**51.34**	**53.31**

Table 7: ECPE-CT performance of models for ECPE-D-DD. The performance between PRG-MoE and ECPE-2D is insignificant in no-context. However, there are notable advances in inter-personal and self-contagion, which relate to the interaction between the two utterances. Through the above performances, we can prove that the Guided-MoE method is helpful in judging the relationship between different utterances. We run the statistical significance test for the F1-Score in the ECPE-CT task. PRG-MoE shows statistically significant difference in "inter-personal" performance than ECPE-2D (**: $p < 0.0001$).

mixing ratio in the guided-MoE. PRG-MoE has the highest performance when λ is 0.6.

Furthermore, we experiment ablation study of *pair-relationship* information; emotion-guide and speaker-guide. Emotion-guide method constructs *pair-relationship* into same emotion and different emotion. Speaker-guide method constructs *pair-relationship* into same speaker and different speaker.

In the comparison of emotion-guide and speaker-guide, speaker information has better guidance for pair extraction than emotion information. All three *pair-relationship* experiments show similar tendency that the combination of MoE and *pair-relationship* information performs better than pure methods.

We also test the effects on the number of experts. We set four experts and assigned one category each to learn the *pair-relationship* information. Note, however, pure MoE can have multiple experts. Figure 3 shows that even when we change the number of experts, PRG-MoE outperforms all pure MoE models.

6.2 Effects on Elements of Utterance Representation

Table 5 shows the performance of PRG-MoE trained with various cases of utterance representation concatenation. The components for concatenation are token sequence representation, emotion prediction, and speaker indicator.

It shows the best performance when all components are concatenated. Providing extra information allows experts to learn more about features between utterances.

6.3 Effects on Window-Constrained method

Table 6 shows the ablation study of the window-constrained method. In natural conversation, the cause of an emotion generally exists near the emotion utterance (Kumar et al., 2022). PRG-MoE focuses on utterances near an emotion utterance through the window strategy, and does not consider utterances far from the emotion utterance because the farther away from the emotion utterance, the less likely an utterance becomes a cause utterance for that emotion utterance. PRG-MoE shows significant improvement in performance with the window-constrained method.

7 Cause Type Classification in ECPE

This section describes the experiments and results of classifying the cause type of emotions - **ECPE-CT**. The cause type in ECPE-CT is categorized based on from which speaker the cause is generated and where the cause is found in the paired utterances. ECPE-CT enhances understanding of the cause of the emotion, enabling more effective use of the cause. For example, most chatbots used in different settings generate responses based on the found cause (Gao et al., 2021). However, ECPE-CT could assist to generate more empathetic responses by incorporating other cases, as categorized below.

Cause types in ECPE-D are as follows:

- **No-context** indicates that an emotion and its cause are found in one utterance.
- **Inter-personal** signifies that the cause exists in the other person's utterances.
- **Self-contagion** refers to the situation where the cause exists in the prior utterance of the same speaker.
- **Latent** refers to a scenario in which the cause does not exist or may occur in the future. The latent type naturally has no pair information, so we classified it as having no pair.

We test PRG-MoE and ECPE-2D, which have best performances among the baselines. For multi-class classification, we modify the output layer of models to be able to output multi-class prediction. Table 7 shows the performance for each cause type. The performance difference between PRG-MoE and ECPE-2D is not significant in "no-context" and "self-contagion". However, there is a significant advance in "inter-personal", which relates to the interaction between the two different speakers. We believe that the Guided-MoE approach performs the function of an identifier, confirming the speaker of a paired utterance.

8 Conclusion

In this paper, we present PRG-MoE, a novel approach for extracting emotion-cause pairs from a dialogue by considering speaker and emotion information. We guide the mixture-of-experts module to consider the relationship between utterances in pairs. To guide mixture-of-experts, we define *pair-relationship*, which is the relationship between utterances. We combine the decision of the gating network and *pair-relationship* information for routing the emotion-cause pairs to proper experts. We also propose a new task, ECPE-CT, which classifies emotion-cause pair by cause type. We evaluate the task with ECPE-D, a dialogue dataset with more emotion-cause pairs than other benchmark ECPE datasets. With ECPE-D, we show that PRG-MoE outperforms other ECPE models in ECPE and Multi-class ECPE tasks.

Limitations

First, we limit the scope of cause to be found in one conversation. However, the actual cause of an emotion may come from other sources outside the given conversation, such as news, weather, and the speakers' previous conversations. But, the ECPE-D dataset does not have such external information, and there are no multiple conversations by the same speaker pairs. Second, we encode the speakers as 0 or 1 since we do not know about the speakers and their relationships. However, emotional conversations would occur more frequently in close relationships such as between family members and friends. Third, we do not test with multi-party conversations. We will experiment with multi-party conversations by annotating the emotion-cause label to another multi-party conversation dataset (e.g., MELD (Poria et al., 2019)). Lastly, we do not consider the order of emotion-cause pairs in a conversation. The order might be helpful in modeling the emotion-cause pairs. For example, if a speaker's emotional state remains unchanged throughout a conversation, a previous pair can help predict a future pair.

Ethics Statement

This paper presents a new ECPE method, PRG-MoE, which extracts emotion and their corresponding cause in a dialogue through the relationship between utterances in pairs. PRG-MoE shows high performance in extracting emotion-cause pairs in a conversation. In this regard, PRG-MoE could be deployed in cause extraction in a dialogue and other real-world applications. We do not report any data collection process in this paper, as we experiment with an open-domain dataset. We experiment with the dialogue dataset based on RECCON (Poria et al., 2021). RECCON is publicly available, and there is no ethical issue.

Acknowledgments

We would like to thank the anonymous reviewers for helpful questions and comments. This work was partly supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2019-0-00421, AI Graduate School Support Program(Sungkyunkwan University)), the Technology Innovation Program (or Industrial Strategic Technology Development Program-Source Technology Development and Commercialization of Digital Therapeutics) (20014967, Development of Digital Therapeutics for Depression from COVID19) funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea) and a grant from the Ministry of Science and ICT(MSIT) to the National Research Foundation of Korea (NRF) [NRF-2021R1A4A3033128].

References

JinYeong Bak and Alice Oh. 2020. Speaker sensitive response evaluation model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6376–6385, Online. Association for Computational Linguistics.

Yinan Bao, Qianwen Ma, Lingwei Wei, Wei Zhou, and Song Hu. 2022. Speaker-guided encoder-decoder framework for emotion recognition in conversation. *ArXiv*, abs/2206.03173.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N. Chang, Sungbok Lee, and Shrikanth S.

Narayanan. 2008. Iemocap: interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4):335–359.

Xinhong Chen, Qing Li, and Jianping Wang. 2020a. A unified sequence labeling model for emotion cause pair extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 208–218, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ying Chen, Wenjun Hou, Shoushan Li, Caicong Wu, and Xiaoqiang Zhang. 2020b. End-to-end emotion-cause pair extraction with graph convolutional network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 198–207, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zifeng Cheng, Zhiwei Jiang, Yafeng Yin, Hua Yu, and Qing Gu. 2020. A symmetric local search network for emotion-cause pair extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 139–149, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020a. ECPE-2D: Emotion-cause pair extraction based on joint two-dimensional representation, interaction and prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3161–3170, Online. Association for Computational Linguistics.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020b. End-to-end emotion-cause pair extraction based on sliding window multi-label learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3574–3583, Online. Association for Computational Linguistics.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2014. Learning factored representations in a deep mixture of experts. *CoRR*, abs/1312.4314.

Chuang Fan, Chaofa Yuan, Jiachen Du, Lin Gui, Min Yang, and Ruifeng Xu. 2020. Transition-based directed graph construction for emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3707–3717, Online. Association for Computational Linguistics.

Jun Gao, Yuhan Liu, Haolin Deng, Wei Wang, Yu Cao, Jiachen Du, and Ruifeng Xu. 2021. Improving empathetic response generation by recognizing emotion cause in conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 807–819, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016. Event-driven emotion cause extraction with corpus construction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1639–1649, Austin, Texas. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shivani Kumar, Anubhav Shrimal, Md. Shad Akhtar, and Tanmoy Chakraborty. 2022. Discovering emotion and reasoning its flip in multi-party conversations using masked memory network and transformer. *Knowl. Based Syst.*, 240:108112.

Jeff Larsen and A. Peter Mcgraw. 2011. Further evidence for mixed emotions. *Journal of personality and social psychology*, 100:1095–110.

Sophia Yat Mei Lee, Ying Chen, and Chu-Ren Huang. 2010. A text-driven rule-based system for emotion cause detection. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 45–53, Los Angeles, CA. Association for Computational Linguistics.

Jiangnan Li, Fandong Meng, Zheng Lin, Rui Liu, Peng Fu, Yanan Cao, Weiping Wang, and Jie Zhou. 2022a. Neutral utterances are also causes: Enhancing conversational causal emotion entailment with social commonsense knowledge. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4209–4215. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Jingye Li, Donghong Ji, Fei Li, Meishan Zhang, and Yijiang Liu. 2020. HiTrans: A transformer-based context- and speaker-sensitive model for emotion detection in conversations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4190–4200, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Qintong Li, Piji Li, Zhaochun Ren, Pengjie Ren, and Zhumin Chen. 2022b. Knowledge bridging for empathetic dialogue generation.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings* *of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Tim Loughran and Bill McDonald. 2011. When is a liability not a liability? textual analysis, dictionaries, and 10-ks. *The Journal of Finance*, 66(1):35–65.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019. MELD: A multimodal multi-party dataset for emotion recognition in conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536, Florence, Italy. Association for Computational Linguistics.

Soujanya Poria, Navonil Majumder, Devamanyu Hazarika, Deepanway Ghosal, Rishabh Bhardwaj, Samson Yu, Romila Ghosh, Niyati Chhaya, Alexander F. Gelbukh, and Rada Mihalcea. 2021. Recognizing emotion cause in conversations. *Cogn. Comput.*, 13:1317–1332.

Elsbeth Turcan, Shuai Wang, Rishita Anubhai, Kasturi Bhattacharjee, Yaser Al-Onaizan, and Smaranda Muresan. 2021. Multi-task learning and adapted knowledge models for emotion-cause extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3975–3989, Online. Association for Computational Linguistics.

Cheng Wang, Georges Balazs, Gyuri Szarvas, Patrick Ernst, Lahari Poddar, and Pavel Danchenko. 2022. Calibrating imbalanced classifiers with focal loss: An empirical study. In *EMNLP 2022*.

Penghui Wei, Jiahao Zhao, and Wenji Mao. 2020. Effective inter-clause modeling for end-to-end emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3171–3181, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1012, Florence, Italy. Association for Computational Linguistics.

Hanqi Yan, Lin Gui, Gabriele Pergola, and Yulan He. 2021. Position bias mitigation: A knowledge-aware graph model for emotion cause extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3364–3375, Online. Association for Computational Linguistics.

Dong Zhang, Liangqing Wu, Changlong Sun, Shoushan Li, Qiaoming Zhu, and Guodong Zhou. 2019. Modeling both context- and speaker-sensitive dependence for emotion detection in multi-speaker conversations. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5415–5421. International Joint Conferences on Artificial Intelligence Organization.

A Implementation Details

We use pretrained *bert-base-cased* from huggingface (Wolf et al., 2020) as a language model. We train PRG-MoE using Adam optimizer (Kingma and Ba, 2015) for 40 epochs and decay the learning rate exponentially for each epoch. The decay rate is 0.05. The batch contains 5 dialogue documents, and the learning rate is set to 5e-5. Dropout is applied to utterance representation with a 0.5 rate. We set the window size as 3 since we follow the previous work for fair comparisons (Ding et al., 2020a). The final loss weight λ_{emo} and λ_{pair} are set to 0.2 and 0.8, respectively. We choose the hyperparameters by manual tuning. We select the hyperparameter based on the f1 score performance. We selected parameter related to data characteristics such as window-size through experiments, and compared baseline models using the same parameter.

Our hardware setting is Intel(R) Xeon(R) Gold 5218R CPU @ 2.10GHz (CPU), and NVIDIA RTX A6000 (GPU). The average running time of PRG-MoE per one epoch is 3min 20s. The inference time per one batch is 1.2 sec. The number of parameters of PRG-MoE is 110M.

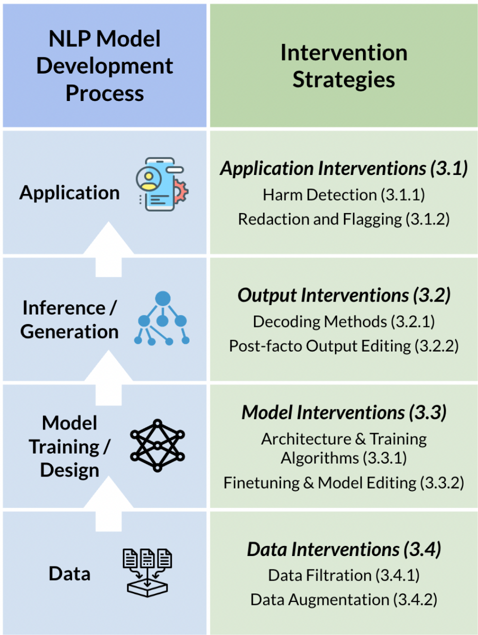
Language Generation Models Can Cause Harm: So What Can We Do About It? An Actionable Survey

Sachin Kumar[*,♣] **Vidhisha Balachandran**[*,♣] **Lucille Njoo**[♡]
Antonios Anastasopoulos[♢] **Yulia Tsvetkov**[♡]
[♣]Language Technologies Institute, Carnegie Mellon University, Pittsburgh PA
[♢]Department of Computer Science, George Mason University, Fairfax, VA
[♡]Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle WA
`{sachink,vbalacha}@cs.cmu.edu, lnjoo@cs.washington.edu, antonis@gmu.edu, yuliats@cs.washington.edu`

Abstract

Recent advances in the capacity of large language models to generate human-like text have resulted in their increased adoption in user-facing settings. In parallel, these improvements have prompted a heated discourse around the risks of societal harms they introduce, whether inadvertent or malicious. Several studies have explored these harms and called for their mitigation via development of safer, fairer models. Going beyond enumerating the risks of harms, this work provides a survey of *practical methods* for addressing potential threats and societal harms from language generation models. We draw on several prior works' taxonomies of language model risks to present a structured overview of strategies for detecting and ameliorating different kinds of risks/harms of language generators. Bridging diverse strands of research, this survey aims to serve as a practical guide for both LM researchers and practitioners, with explanations of different mitigation strategies' motivations, their limitations, and open problems for future research.

1 Introduction



NLP Model Development Process	Intervention Strategies
Application	***Application Interventions (3.1)*** Harm Detection (3.1.1) Redaction and Flagging (3.1.2)
Inference / Generation	***Output Interventions (3.2)*** Decoding Methods (3.2.1) Post-facto Output Editing (3.2.2)
Model Training / Design	***Model Interventions (3.3)*** Architecture & Training Algorithms (3.3.1) Finetuning & Model Editing (3.3.2)
Data	***Data Interventions (3.4)*** Data Filtration (3.4.1) Data Augmentation (3.4.2)

Figure 1: Overview of Intervention Strategies. A typical ML/NLP model development process involves data collection/curation, model training and design, inference, and finally application deployment. For each phase of this development cycle, different techniques can be adopted to mitigate harms. Our survey presents a taxonomy of intervention strategies organized around the different phases where they can be applied.

The new wave of large language models (LMs; Brown et al., 2020; Chowdhery et al., 2022; Zhang et al., 2022b) capable of generating text with human-like fluency, coherence, and realism (Zellers et al., 2020; Ippolito et al., 2020) has caused a paradigm shift in our society.[1] With applications like OpenAI's ChatGPT, Microsoft's Bing, and Google's Bard, bringing such LMs directly to users, we are beginning to see the impact in fields like education (Schulten, 2023; Gleason, 2022), healthcare (Patel and Lam, 2023), law (ChatGPT and Perlman, 2022), science (Stokel-Walker, 2023), and more. Since language is inherently a tool of power—the primary means by which people and societies perpetuate stereotypes and manipulate opinions (Bar-Tal et al., 2013; Chong and Druckman, 2007, *inter alia*)—LMs that are deployed to millions of users also hold similar power, but our understanding of their risks/harms has lagged behind (Bender et al., 2021).

Indeed, LMs have been shown to introduce vulnerabilities and threats, both inadvertent and malicious, to individual users, social groups, and content integrity. Without social context and content control, deployed language generators have quickly derailed to racist, homophobic, hateful comments (Hunt, 2016; Jang, 2021; Wolf et al., 2017; Vincent, 2022), compromised user privacy (Carlini et al., 2021), spread disinformation (Shao et al., 2018),

---

[*] Equal contribution

[1] While the majority of these models are trained on English, recent studies have also obtained similar advancements in other languages (Lin et al., 2021; Shliazhko et al., 2022).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3299–3321
May 2-6, 2023 ©2023 Association for Computational Linguistics

and even encouraged suicide (Daws, 2020). Prior works have outlined these risks (Maynez et al., 2020; Sheng et al., 2021; Weidinger et al., 2022; Zhuo et al., 2023), proposed taxonomies (Weidinger et al., 2022), discussed their points of origin, and advocated for future research on ethical development of LMs (Bender et al., 2021; Solaiman et al., 2019).

However, there is little work that summarizes **actionable approaches and technical solutions** to preventing or mitigating these potential harms. In this survey, we present a **comprehensive, unified taxonomy** of relevant **mitigation strategies** proposed in prior literature, specifically focusing on **language generation models**.

We organize these strategies based on where they fit in different stages of LM development: in data collection, modeling, decoding, and deployment. Within each of these categories, our taxonomy brings together prior works that have been treated as disjoint areas targeting different types of harms (toxic/biased language and misinformation). In addition, we identify their gaps and highlight directions for future research. These include incorporating sociocultural context to produce socially-sensitive interventions, detecting and handling generations with different intents (inadvertent vs. malicious), and going beyond an English, Western/US-centric view to account for the challenges of ethics in *multilingual* language generation.

2 Background

Throughout this paper, we use the term *language models* (LMs) to refer to their classic definition as generative models, which predict the next token given the preceding generated context. This paradigm also subsumes conditional LMs that depend on additional inputs via an encoder. We provide more details in Appendix A.

2.1 Risks in Language Generation

Before diving into mitigation techniques (§3), we briefly outline potential harms that LMs can cause, following Weidinger et al. (2022)'s taxonomy.

Discrimination, Toxicity, and Exclusion: The scope of linguistic diversity in human communication is enormous and is linked to personal, social, and cultural factors (Holmes and Wilson, 2017; Eckert and McConnell-Ginet, 2003; Coates, 2016; Chambers, 1995). As such, language produced in the real world reflects sociocultural stereotypes and presuppositions that LMs can overfit to and amplify (Bar-Tal et al., 2013; Zhao et al., 2017; Sun et al., 2019), leading to several types of harms. (1) *Stereotyping and discrimination* occurs when generated text reinforces discriminatory stereotypes and perpetuates biases against disadvantaged groups, based on factors like gender, race, religion, sexuality, (Bender et al., 2021), and intersectional identities (Crenshaw, 2017). Evidence for this behavior has been substantially corroborated in NLP literature (Blodgett et al., 2020; Nadeem et al., 2021; Nozza et al., 2021; Liang et al., 2021; Field et al., 2021; Lin et al., 2022a, *inter alia*). (2) *Toxicity* describes generated language that is offensive, threatening, violent, or otherwise harmful (Gehman et al., 2020; Rae et al., 2021; Abid et al., 2021). It can range from overtly toxic content, such as violent hate speech, to more subtle, veiled toxicity, such as microaggressions (Breitfeller et al., 2019). (3) *Exclusion* refers to the disparate performance of models across language variations. Models may fail to understand or generate "non-standard" dialects and sociolects, essentially excluding speakers of such variants from their user base (Joshi et al., 2020; Koenecke et al., 2020; Winata et al., 2021).

Factual Errors, Misinformation, and Disinformation: LMs are able to generate fluent outputs that users may easily mistake for human-written text (Ippolito et al., 2020), but such utterances may be factually incorrect or misleading (Maynez et al., 2020; Xu, 2020; Lin et al., 2022b; Bickmore et al., 2018; Daws, 2020). This can cause harm inadvertently (via misinformation) or can also be used maliciously (disinformation; Bradshaw and Howard, 2019; Beskow, 2020; Buchanan et al., 2021).

Privacy Violations: LMs' vast training corpora often contain sensitive information, and LMs can memorize these details and generate them verbatim when prompted by users, leading to privacy violations (Kim, 2016; Mirshghallah et al., 2020; Brown et al., 2022). LMs have been shown to leak personally identifiable information, such as social security numbers, phone numbers, bank account information (Carlini et al., 2021), and private clinical notes (Lehman et al., 2021); they have even leaked software code and other protected intellectual property (Ippolito et al., 2022). Deploying large LMs can thus pose serious security risks to people whose private information might have found its way into a model's training data.

Other Underexplored Issues: Weidinger et al. (2022) discuss other malicious applications, as well as the economical and environmental impacts of LMs. While extremely important, mitigating these risks requires not only technical innovation, but also the development of regulatory practices and policies in an interdisciplinary effort. We focus on algorithmic solutions in this survey, leaving this discussion for future work.

3 Taxonomy of Intervention Strategies

The development pipeline of a typical machine learning model involves several critical decisions where risks of harms can arise. Stakeholders have access to different pipeline components and therefore may employ different intervention strategies. For example, while a researcher involved in data curation can intervene before training, an application developer with limited access to a black-box model might only be able to intervene at inference. We present a taxonomy of intervention strategies organized by the stages of a model development lifecycle (Fig. 1), aiming to showcase the tools that can be employed at different stages. We step backward through the pipeline, beginning with application-level interventions employed post-deployment and peeling back the layers through output-level interventions, model interventions, and finally ending at data-level interventions (summarized in Tab. 1).

3.1 Application Level Interventions

3.1.1 Harm Detection and Redaction

In order to mitigate harms at the application level, we first need to be able to *detect* problematic, incorrect, and unreliable model outputs (Raji et al., 2020). User-facing applications can employ detectors to intervene before harmful text reaches a user. Such detectors are typically coarse, binary text classifiers, often trained for a single task, such as predicting toxicity (Nobata et al., 2016; Davidson et al., 2017b; Xiang et al., 2021), or the factual accuracy of the outputs (Kryscinski et al., 2020; Goyal and Durrett, 2020; Wang et al., 2020).

Early approaches to building toxicity detectors focused on linear models relying on hand-designed *features* based on lexicons, e.g., hatebase, (Xiang et al., 2012; Dadvar et al., 2012; Burnap and Williams, 2015; Liu and Forss, 2015), n-grams, capitalization/punctuation details (Chen et al., 2012; Waseem and Hovy, 2016; Nobata et al., 2016; Xu et al., 2012; Burnap and Williams, 2016). For misinformation detection, features like the presence of new entities or facts in generated document summaries have been employed which can indicate hallucination (Zhao et al., 2020; King et al., 2022).

Linear classifiers, while interpretable, tend to overfit to lexical features, are prone to false positives, and are easy for malicious users to bypass (Kurita et al., 2019). Neural text classifiers, on the other hand, can incorporate contextual information and have been shown to be more robust (Gambäck and Sikdar, 2017; Pitsilis et al., 2018). When built by finetuning pretrained LMs instead of training from scratch, they naturally lead to even better performance (d'Sa et al., 2020; Xiang et al., 2021). Based on these models several toxicity detection tools like Perspective API, OpenAI content filter or ToxiGEN are now publicly available.

To train classifiers for toxicity detection, annotated datasets in several domains have been collected for English (Davidson et al., 2017a; Waseem and Hovy, 2016; Wiegand et al., 2018; Pavlopoulos et al., 2017; Mubarak et al., 2017; Moon et al., 2020), especially to detect overtly toxic text. Human annotation efforts for more subtle toxicities like microaggressions, however, is challenging due to annotators' own biases (Breitfeller et al., 2019). Hence, unsupervised or distantly supervised approaches have been adopted to detect them (Korzeniowski et al., 2019; Field and Tsvetkov, 2020; Sabri et al., 2021). Compared to English, such resources for other languages are severely lacking (Ousidhoum et al., 2019a).

Information-related harms can arise either inadvertently (due to model errors) or deliberately (due to malicious users). Detecting manipulation in the human-written text is an active area of research and those approaches can also be employed for machine-generated text. Prominent research directions include automated fact-checking, propaganda, or fake news detection for which several annotated datasets (Oshikawa et al., 2020; Martino et al., 2020; Zhou and Zafarani, 2020; Guo et al., 2022; Huang et al., 2022) and shared tasks (Thorne et al., 2018; Da San Martino et al., 2019; Feldman et al., 2021) exist. These approaches have also been adopted to assist human fact-checkers (Shaar et al., 2021; Nakov et al., 2021). However, humans are easily fooled by machine-generated fake news (Zellers et al., 2020; Ippolito et al., 2020). An alternate solution is to, not find informational discrepancies, but simply detect and flag whether

Application Level Interventions	Feature-based Detection	Toxicity	Lexical features (Xiang et al., 2012; Dadvar et al., 2012; Burnap and Williams, 2015; Liu and Forss, 2015); n-gram features (Chen et al., 2012; Waseem and Hovy, 2016; Nobata et al., 2016; Xu et al., 2012; Burnap and Williams, 2016)
		Misinformation	Word-Level features (Zhao et al., 2020; King et al., 2022)
	Neural Detection	Toxicity	Supervised: (Gambäck and Sikdar, 2017; Pitsilis et al., 2018; d'Sa et al., 2020; Xiang et al., 2021); Semi- and Unsupervised: (Korzeniowski et al., 2019; Field and Tsvetkov, 2020; Sabri et al., 2021)
		Misinformation / Factuality	Supervised fake-news detection (Thorne et al., 2018; Oshikawa et al., 2020; Martino et al., 2020; Zhou and Zafarani, 2020; Guo et al., 2022); Factual error detection (Kryscinski et al., 2020; Goyal and Durrett, 2020; Pagnoni et al., 2021)
		Disinformation	Machine-generated text detection (Dugan et al., 2020; Gehrmann et al., 2019)
Output Level Interventions	Reranking	Toxicity	Rejection sampling using toxicity detectors (Wang et al., 2022)
		Misinformation / Factuality	Ranking using factuality classifiers (Krishna et al., 2022; King et al., 2022)
	Controlled Decoding	Toxicity	Autoregressive toxic content control (Yang and Klein, 2021; Liu et al., 2021a; Dathathri et al., 2019; Krause et al., 2021; Schick et al., 2021; Lu et al., 2021; Pascual et al., 2021; Wolf et al., 2020); Non-autoregressive toxic content control(Kumar et al., 2022; Mireshghallah et al., 2022)
		Privacy	Differentially private decoding (Majmudar et al., 2022)
		Misinformation / Factuality	Autoregressive factual error control(King et al., 2022; Lu et al., 2022); Non-autoregressive factual error control (Kumar et al., 2021b)
	Post-processing	Toxicity	Rewriting harmful text (Pryzant et al., 2020; He et al., 2021b; Ma et al., 2020)
		Misinformation / Factuality	Editing factual errors (Cao et al., 2020; Lee et al., 2022a; Balachandran et al., 2022)
Model Level Interventions	Architecture	Misinformation / Factuality	Attention (Nan et al., 2021; Zhu et al., 2021), Coreference (Levy et al., 2021); Text Entailment (Falke et al., 2019; Li et al., 2018); Others (Wiseman et al., 2018; Falke et al., 2019; Wan and Bansal, 2022).
	Training	Toxicity	Class-conditional LMs (Keskar et al., 2019; Gururangan et al., 2020; Chan et al., 2021); Instruction-based learning (Ouyang et al., 2022; Wei et al., 2022a)
		Privacy	Differential Private training (Kerrigan et al., 2020; Li et al., 2022; Shi et al., 2021); Knowledge Unlearning (Jang et al., 2022)
		Misinformation / Factuality	Structured KBs (Wang et al., 2021b; Liu et al., 2022; Yu et al., 2022; Liu et al., 2022; Lewis et al., 2020; de Masson d'Autume et al., 2019; Izacard and Grave, 2021; Hossain et al., 2020; Lewis et al., 2020), Retrieval-based (de Masson d'Autume et al., 2019; Izacard and Grave, 2021; Hossain et al., 2020); Summarization (Huang et al., 2020), Translation (Bapna and Firat, 2019), Dialogue models (Dinan et al., 2019; Fan et al., 2021; Zhang et al., 2020a)
	Fine-tuning	Discrimination & Toxicity	Supervised fine-tuning (Gururangan et al., 2020; Chan et al., 2021; Liu et al., 2023); RL based fine-tuning (Alabdulkarim et al., 2021; Liu et al., 2021b; Ouyang et al., 2022; Stiennon et al., 2020); Prompt-based learning (Gehman et al., 2020)
		Exclusion	Adapting for low-resource varieties (Chronopoulou et al., 2020; Kumar et al., 2021a)
	Model Editing	Toxicity	Modifying FF layers(Geva et al., 2022)
		Misinformation / Factuality	Auxiliary editors to modify parameters (De Cao et al., 2021; Mitchell et al., 2022); Modify parameters associated with behavior (Meng et al., 2022, 2023)
Data	Filtration	Toxicity	Removing ’unwanted’ words from corpus (Raffel et al., 2020; Brown et al., 2020; Dodge et al., 2021); Removing toxic data using classifiers (Ngo et al., 2021)
		Privacy	Filtering private/duplicate data (Henderson et al., 2022; Kandpal et al., 2022; Lee et al., 2022b)
	Augmentation	Discrimination	Adding synthetically generated data (Dinan et al., 2020; Liu et al., 2020; Stafanovičs et al., 2020)
		Toxicity	Adding safer example data (Mathew et al., 2018)

Table 1: Strategies for mitigating various risks and harms from language models.

the text has been machine-generated (Gehrmann et al., 2019; Dugan et al., 2020; Ippolito et al., 2020; Mitchell et al., 2023), putting the onus to trust the information on the users (Jawahar et al., 2020).

To detect *inadvertent* factual errors, prior works have developed classifiers by training them to detect heuristically introduced synthetic errors in factually correct text (Kryscinski et al., 2020; Goyal and Durrett, 2020), or question-answering errors using targeted QA models (Scialom et al., 2021). Being trained on synthetic data, such detectors typically do not generalize and have low human judgment correlations (Pagnoni et al., 2021).

Relying on the detectors, the most straightforward way a user-facing application can prevent harm is to not display the text at all (*redacting*) or to display it with a warning sign (*flagging*) (Xu et al., 2020). Even when the detectors are imperfect, explicitly flagging problematic outputs is still useful because it signals users to take model outputs with a grain of salt. However, this strategy is not always applicable: for example, in speech-based dialogue agents, "displaying" a warning sign is a nontrivial UX decision, and in auto-complete assistants (such as in Gmail Smart Compose), redacting is not an option and simply warning may not dissuade users from accepting the generated text.

Challenges: Predicting whether a text is harmful is often highly contextual and subjective. For toxicity detection, factors like region, political views, and the users' sociocultural background affect whether they perceive the text as toxic (Xenos et al., 2021). Existing datasets are often biased due to their curation process (Dixon et al., 2018; Wiegand et al., 2019; Geva et al., 2019; Sap et al., 2021; Kryscinski et al., 2020) and can have unreliable annotations (Ross et al., 2017; Field and Tsvetkov, 2020; Pagnoni et al., 2021). Further, as with many black-box models, classifiers overfit to spurious artifacts (Gururangan et al., 2018; McCoy et al., 2019; Kumar et al., 2019) and amplify biases in their training data (Zhao et al., 2017; Sun et al., 2019). For instance, toxicity detectors have been shown to disproportionately flag African-American English (AAE) as toxic (Sap et al., 2019). Additionally, such filters might overfit to a subset of small features, with more subtle problematic text evading such filters. Ippolito et al. (2022) show that blocking verbatim training data is insufficient for mitigating privacy concerns in code-generation. We discuss these issues further in §4, highlighting future directions to building finer-grained and explainable approaches for detecting harmful text.

3.2 Output Level Interventions

Increasingly, practitioners are building applications using LMs as APIs without explicit knowledge of how the model was trained or what training data was used.[2] Such APIs may vary in how much information developers can see: some allow access to all LM parameters, while black box APIs like GPT3 limit access to model outputs only. Hence, multiple solutions have been proposed for intervening at *model output generation* by editing the outputs with auxiliary models or modifying decoding algorithms.

3.2.1 Post-Factum Editing Model Outputs

Recent studies have explored ways to *edit or revise* model-generated text to remove harmful content. Text editing is a decades-old subfield of NLP that has traditionally focused on fixing errors in machine translation (Chollampatt et al., 2020; Simard et al., 2007; Chatterjee et al., 2020) or grammar in human-written text (Wang et al., 2021c). While many approaches in this area are applicable to post-editing LM outputs, in this survey, we highlight recent work related to *rewriting harmful text*.

The first set of works treats the task of rewriting as a sequence labeling task, where each token in the output sequence is either substituted, deleted, or kept the same (Pryzant et al., 2020; He et al., 2021b). This, however, can be limiting when the entire output needs rewriting. For text-to-text tasks, like translation, summarization, etc. which are trained with parallel data, the same data can be adapted to train an editing model by converting source-target pairs to source-*output*-target triplets using model-generated *outputs* for each source, along with an additional signal indicating errors (obtained using automatic evaluators or human judgment). For more open-ended tasks, prior works explored unsupervised solutions for bias correction (Ma et al., 2020) and semi-supervised methods to correct factual errors (Cao et al., 2020; Lee et al., 2022a; Balachandran et al., 2022). Such methods create synthetic data by inducing errors in clean text and train a model to correct them.

3.2.2 Decoding Methods

Several search and sampling algorithms have been introduced recently to improve the quality of LM-

---

[2] see `https://gpt3demo.com/` for examples

generated text (Graves, 2012; Fan et al., 2018; Holtzman et al., 2020; Meister et al., 2022). In parallel, works on controlling decoding algorithms to promote or demote specific properties in the output text have been developed (Zhang et al., 2022a).

The decoding controls are auxiliary models measuring if the generated text is harmful implemented similarly to the detectors we discussed in §3.1.1, such as toxicity/bias classifiers (Dathathri et al., 2019; Krause et al., 2021; Liu et al., 2021a), factuality metrics (Kryscinski et al., 2020; Goyal and Durrett, 2020). A simple way to use the detectors is *rejection sampling* or *reranking*: for a given input, multiple outputs are generated and then reranked using detector scores to discard dubious outputs (Krishna et al., 2022; King et al., 2022). However, this is often intractable for complex phenomena like factual accuracy of a text or when using multiple controls, since all the generated candidates might be rejected.

To tackle these issues, a class of algorithms that we call *guided-autoregressive decoding* aims to incorporate control by modifying output distributions at every decoding step. One branch of work adopts *logical* controls, where developers directly specify sets of words that should (or not) appear in the output (Lu et al., 2021; Pascual et al., 2021). Wolf et al. (2020) apply this method to zero out the probabilities of offensive terms, King et al. (2022); Lu et al. (2022) improve factual accuracy of generated text by up-weighting generation probabilities of entities present in the source, and Majmudar et al. (2022) apply it for differentially private decoding. A second branch of work composes the LM likelihood with the probabilities from the detectors, to up-weight or down-weight the token probabilities at each decoding step (Yang and Klein, 2021; Liu et al., 2021a; Dathathri et al., 2019; Krause et al., 2021; Schick et al., 2021).

More recent work has also explored ways to induce sentence-level control via *non-autoregressive controlled decoding*. These algorithms incorporate control using Monte Carlo Markov Chain (MCMC) techniques (Hoang et al., 2017; Qin et al., 2020; Mireshghallah et al., 2022), in which a full sequence is initialized and iteratively updated. They have been applied for reducing toxicity (Kumar et al., 2022), and improving fidelity in translation systems (Kumar et al., 2021b). While promising, these techniques suffer from slower decoding speed and need further exploration to be practically used.

Challenges Decoding interventions rely on accurate detectors, hence challenges in designing robust detectors (§3.1.1) also impact decoding algorithms. For example, Xu et al. (2021) show that toxicity avoidance algorithms refrain from generating AAE, thereby causing another harm (exclusion) while trying to address the first (toxicity). Also, detecting misinformation and factuality can be extremely hard using simple detectors that do not provide a useful signal to guide the decoding process, so prior works have primarily employed heuristics. Finally, controlled decoding algorithms are double-edged in that controls can be reversed by malicious users to inflict harm—to generate hateful messages, or to do targeted manipulation by copying users' personas. However, this risk should not discourage research in decoding algorithms; rather, research on detecting such malicious uses should be conducted in parallel.

3.3 Model Level Interventions

Several recent studies have provided evidence that certain optimization procedures can result in harmful generations downstream (Hall et al., 2022; Taori and Hashimoto, 2022). In this section, we describe approaches that modify LM parameters to prevent such generations by either architecture/training interventions or finetuning/model editing interventions.

3.3.1 Architecture and Training Algorithms

Closely related to applying control at inference time are class-conditioned LMs, which are trained to depend on "control codes" via an additional input (Keskar et al., 2019; Gururangan et al., 2020; Chan et al., 2021). When trained with data annotated for toxicity or bias, these LMs can be prompted to avoid those outputs. Another recently popularized paradigm in LM training is *instruction-based learning*, where in addition to the objective to predict the next token, models are also trained to solve NLP tasks with instructions written in natural language (Wei et al., 2022a; Sanh et al., 2022). Providing explicit instructions to not generate harmful text has shown some promise (Ouyang et al., 2022; Wei et al., 2022a) and is an interesting avenue for future work.

In text-to-text tasks like summarization, the goal is to produce text that is factually consistent with the input without hallucinating information. An LM, however, is typically not constrained to predict tokens grounded in verifiable knowledge, which can lead to misinformation. Thus, several studies explore modifying LM training objectives to *incorporate factual information* using either knowledge bases (KBs) or graphs (Yu et al., 2022): each token prediction is scored not only on its likelihood given context, but also on whether the generation is grounded in facts in the KBs (Wang et al., 2021b).[3]

However, existing KBs are limited in size as manually curating them is an arduous and expensive process. As an alternative, Liu et al. (2022) propose using automatically generated KBs to train LMs. In contrast, Lewis et al. (2020); de Masson d'Autume et al. (2019); Izacard and Grave (2021) use unstructured text as knowledge. Known as *retrieval-augmented LMs*, they are trained with a two-stage approach of first retrieving a document from an unstructured source like Wikipedia and using it as additional context for generation, essentially providing evidence for the LM-generated text. Wang et al. (2021a); Ji et al. (2020) follow a similar approach to embed commonsense knowledge in LMs. These existing solutions have been used to tackle content-related harms like factual consistency in generated text (Huang et al., 2020; Bapna and Firat, 2019; Dinan et al., 2019; Fan et al., 2021) but future work in reducing discrimination and toxicity in LMs may also benefit from KBs that encode social (Chang et al., 2020), cultural, (Hershcovich et al., 2022), and moral norms (Hendrycks et al., 2021; Jiang et al., 2021). Such LMs augmented with external knowledge can also be dynamically updated by modifying the knowledge source at test time with new information (Khandelwal et al., 2020; He et al., 2021a).

While external knowledge helps provides context, models may not rely on them and still hallucinate. To explicitly control for context, recent studies have explored (1) modifying attention mechanisms to specifically capture relationships between entities (Nan et al., 2021; Zhu et al., 2021), (2) improving coreference to mitigate gender bias in translation (Levy et al., 2021), and (3) using text entailment to develop loss functions to improve fidelity (Falke et al., 2019; Li et al., 2018). Some other notable directions in this space involve fact-aware pretraining (Falke et al., 2019; Wan and Bansal, 2022) and structured learning frameworks (Wiseman et al., 2018).

Finally, to reduce privacy risks in LMs that memorize user information without sacrificing model capabilities, most prominent solutions are based on differentially private (DP) learning (Kerrigan et al., 2020; Shi et al., 2021). DP can provide provable guarantees on the privacy-utility trade-off, however, it requires the LMs to be retrained for each private information that needs to be removed and be quite expensive.

3.3.2 Finetuning and Model Editing

Designing and training models from scratch to mitigate harms can incur heavy environmental and resource costs. In contrast, an alternative branch of work has developed methods for *modifying the model parameters* of already-trained LMs, which requires much fewer resources. An elementary way of doing this is *finetuning* (a subset of) an LM's parameters on small, curated datasets that contain a well-balanced proportion of data for various demographics and filtered for nontoxicity (Gururangan et al., 2020; Chan et al., 2021; Liu et al., 2023). Such balanced and filtered data encourage models correct biases learned from skewed and toxic training data, resulting in safer generated text.

Prompt-tuning based methods (Wang et al., 2022) have also shown some success where instead of fine-tuning all the parameters, a prompt (using a small set of parameters) is learned without modifying the rest of the model to perform a task. This paradigm uses the generative power of large LMs, while simultaneously nudging the distribution of generated text toward less harmful content. These approaches have successfully been used to reduce toxicity (Gehman et al., 2020) and exclusion (Chronopoulou et al., 2020; Kumar et al., 2021a). However, finetuning or prompt-tuning on a small dataset may lead to overfitting reducing the general purpose utility of LMs.

Finetuning LMs with *reinforcement learning* (RL) has been suggested as a better alternative (Alabdulkarim et al., 2021; Liu et al., 2021b; Ouyang et al., 2022; Stiennon et al., 2020; Lu et al., 2022; Ramamurthy et al., 2022) for training modern LMs. RL models do not require carefully balanced datasets and can instead learn from discrete rewards such as human feedback (Sun et al., 2020; Ouyang et al., 2022) or auxiliary model-based feedback (Perez et al., 2022). It has been shown to reduce toxic text generated by the models (Bai et al., 2022) and to encourage models to generate more factual text (Mao et al., 2020; Stiennon et al., 2020).

---

[3] Knowledge-augmented LMs is a rich field where most existing work focuses on masked LMs (Zhu et al., 2022) for solving understanding tasks. Here we highlight papers on generation.

Another less-explored but more computationally practical alternative to finetuning is *model surgery or editing*, which identifies a specific set of neurons that contribute to harmful generations. Culling such parameters has been shown to reduce toxicity (Geva et al., 2022). In a similar vein, De Cao et al. (2021); Mitchell et al. (2022); Meng et al. (2022, 2023) systematically edit model parameters to revise facts memorized by the model. De Cao et al. (2021); Mitchell et al. (2022) use auxiliary editor networks to predict updates to model parameters constrained to revise a fact without changing other facts. Alternatively, Meng et al. (2022, 2023) use interpretability techniques to identify parameters associated with memorizing said facts and edit them locally to revise them.

Challenges The biggest argument against mitigation techniques involving training LMs from scratch or augmenting them with knowledge is its cost, making these interventions infeasible for most researchers and practitioners. However, even for organizations with access to large computing resources, research on training safer LMs lags behind research on training ever-larger LMs on raw data. We attribute this to the difficulty of curating KBs, as well as the decreased training and inference speed that comes with such modifications. Finetuning, on the other hand, is less costly but may reduce the general utility of the LMs and has not been shown to be useful in reducing information-related harms. Future work may benefit from drawing on continual (Dhingra et al., 2022) and reinforcement learning (Ouyang et al., 2022) techniques for more practical solutions for large models.

3.4 Data Level Interventions

Training any machine learning model requires data, so a natural approach to creating fairer, more reliable LMs is carefully creating balanced training sets that are broadly representative of different worldviews. This requires dedicated and expensive efforts in data curation (Hutchinson et al., 2021; Jo and Gebru, 2020; Kammoun et al., 2022) and novel data pipelines (Denton et al., 2020). Existing works tackling this issue devise semi-automated solutions, which we categorize as follows.

3.4.1 Data Filtration

This simple technique involves removing problematic documents from the training corpus. As training sets can be extremely large, sophisticated neural filters can be prohibitively slow to apply. Hence, most work has utilized simple filters, such as the presence of "unwanted" words (Raffel et al., 2020) or the predictions of linear classifiers (Brown et al., 2020). To mitigate privacy violations, Henderson et al. (2022) construct clean training data by filtering private information and Kandpal et al. (2022); Lee et al. (2022b) filter duplicate training data.

Due to their simplistic setup, these approaches admit many false negatives (failing to detect documents with subtle toxicity) and false positives (erroneously flagging documents that discuss sensitive topics and use hateful speech as examples; additionally, removing data from different dialects like AAE), unintentionally exacerbating risks of marginalization and exclusion (Dodge et al., 2021)). Alternatively, Ngo et al. (2021) train an LM on raw data, then feed the LM manually-curated toxic prompts and filter out documents to which the LM assigns high probability, and then retrain the LM on the filtered corpus.

3.4.2 Data Augmentation

While data filtration aims to remove problematic training samples, data augmentation aims to offset the effect of problematic data by *adding* safer/healthier examples to existing datasets. Mathew et al. (2018) explore adding counterspeech (comments that counter the hateful or harmful speech) to datasets in order to balance out the hate speech already present in web data. Augmentation with synthetically generated data has also been explored for gender bias mitigation in dialogue (Dinan et al., 2020; Liu et al., 2020) and translation models (Stafanovičs et al., 2020).

Challenges Since language, identity, and society are tightly intertwined, aggressive data filtering methods risk further imbalancing already imbalanced data. Besides, models trained on filtered data may still degrade when toxic inputs are provided to it. Further, while data augmentation methods have merit, these methods are extremely difficult to large scale. Finally, data interventions are primarily designed to address population-centric risks such as discrimination, toxicity, and, to an extent, exclusion and privacy—but not factuality which is a by-product of training. It is challenging to define (Aly et al., 2022) and detect unsupported facts (Ansar and Goswami, 2021) in the wild, making data interventions insufficient for addressing misinformation and factuality-related harms.

4 Discussion and Open Challenges

Though the interventions strategies we discuss achieve some success, many risks of LMs are still not well understood. Below we discuss open problems and avenues for future work to encourage the development of safer LMs.

Where should one intervene? Different stakeholders are involved in different model development phases with varying access to resources. As a result, intervention strategies are different depending on the stakeholder. A significant chunk of the responsibility to develop safer LMs falls on researchers and organizations with access to substantial resources who can implement data or modeling interventions. In contrast, practitioners building applications on top of LMs may have access to neither the training data nor the computational resources required to design and train safe LMs. In such cases, flagging and decoding approaches are more practical. In practice, a combination of multiple interventions may be required to both cover a wide array of risks and improve robustness.

Evolving risks in the ChatGPT era: LMs are seeing tremendous, rapid growth; larger models are being released every few months (Shoeybi et al., 2019; Brown et al., 2020; Zhang et al., 2022b; Zeng et al., 2023) and deployed in user-facing applications. Many recent LMs like OpenAI's ChatGPT have garnered attention beyond the research community, impacting a range of fields and crossing geographical and language barriers to reach users all over the world (Reuters, 2023; Varghese, 2023; So-hyun, 2023). In such a fast-moving ecosystem, it is ever more essential to proactively study and mitigate LMs' potential harms. Risk mitigation research tends to lag behind model development and is often considered as an afterthought. Though behaviors may emerge unpredictably (Wei et al., 2022b), as we outline in this survey, intervention strategies can and should be applied at different stages of model development to reduce the potential for these influential LMs to cause harm.

Risks exist in LMs in all languages: Most research on large LMs, their uses, and their risks is Western-centric and primarily conducted on the English language. However, while a few studies have been conducted on detecting harmful text in non-English datasets (Ousidhoum et al., 2019b; Leite et al., 2020; Burtenshaw and Kestemont, 2021; Bogoradnikova et al., 2021; Costa-jussà et al., 2022, *inter alia*), research on mitigation in non-English settings is lagging (Pamungkas et al., 2021). Further, the definitions of risks themselves change with different context and across cultures. Hence, there is a dire need to develop cross-cultural, cross-lingual analyses as well as mitigation tools.

Harm detection beyond simple classifiers Many of the shortcomings of interventions are at their root due to poorly defined risk detection methods. Current detection methods are primarily binary classifiers on various axes like toxicity and factuality, but we recommend researchers and practitioners to move beyond simplistic coarse classifiers and towards more fine-grained (Xiang et al., 2021; Goyal and Durrett, 2020; Da San Martino et al., 2020), interpretable (Koh and Liang, 2017; Han and Tsvetkov, 2020, 2021), and explainable (Pagnoni et al., 2021; Gehrmann et al., 2019) harm detectors to support better harm mitigation strategies (Lipton, 2018; Jacovi et al., 2021).

Systematic evaluation frameworks for mitigation strategies Though LM performance is usually systematically evaluated through benchmarks (Wang et al., 2019b,a; BIG-bench collaboration, 2022), practices for evaluating harms in LM-generated text or the effectiveness of mitigation strategies are not. While there is an emerging body of work dedicated to benchmarking LM harms (Rauh et al., 2022), the space of potential harms is huge and intersectional, and existing work only covers a fraction of it. Developing a suite of evaluations or augmenting existing generation benchmarks (Mille et al., 2021) with axes of risk evaluations (Ribeiro et al., 2020) will encourage the development of holistic solutions, bridging discrimination/toxicity and information-related harms—two related directions in which researchers have often developed similar solutions.

5 Conclusion

We present a survey of practical methods and techniques for addressing the societal harms and safety risks of language generation models. Our structured taxonomy covers a wide variety of interventions at different stages of the model development pipeline to mitigate harms. This work bridges multiple strands of research and presents an actionable overview on methods for preventing harms from language generation models.

Limitations

The goal of this survey was to present current research on analyzing and mitigating harms of language generation. There are multiple documented and anticipated harms that these models perpetuate, and it is not feasible to address intervention strategies for each of them. We aimed to generalize multiple proposed solutions and present them in a structured form, considering a few popularly studied harms as case studies. Inevitably, certain harms and their mitigation strategies might not have been considered for this survey.

Current research in this field is nascent but fast-moving. While this survey enlists techniques and approaches that are popular now, there is a potential for them to be replaced with newer research. We anticipate that this survey may need to be updated or even redone to incorporate new research.

Ethics Statement

In this survey, we present and discuss various risk analyses and intervention strategies to prevent societal harms from LMs. We also comment on common themes across approaches for detecting and resolving population-centric harms (such as toxicity and discrimination) and misinformation-related harms, and we recommend future work combining them. First, many datasets and resources we discuss may contain biases, and using them in downstream applications can lead to risks as we have outlined. Second, many techniques we discuss have limitations or are known to exacerbate other kinds of harms (Xia et al., 2020), and thus, applying them to newer problems may lead to unseen issues. Finally, the interventions we identify to raise general awareness have the potential for misuse: a malicious user can further imbalance the data to train even more harmful models, use the models and decoding algorithms to generate fake news, and target marginalized populations. This, however, should not discourage the development of mitigation strategies; rather, more work should be done to detect and ban malicious users. This requires not only technological solutions in NLP, but also in social science, social network analysis, and public policy.

Acknowledgements

We gratefully acknowledge support from NSF CAREER Grant No. IIS2142739, NSF grants No. IIS2125201, IIS2040926, IIS2203097, Workhuman, and an Alfred P. Sloan Foundation Fellowship. S.K. gratefully acknowledges support from a Google PhD Fellowship.

References

Abubakar Abid, Maheen Farooqi, and James Zou. 2021. *Persistent Anti-Muslim Bias in Large Language Models*, page 298–306. Association for Computing Machinery, New York, NY, USA.

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Armen Aghajanyan, Dmytro Okhonko, Mike Lewis, Mandar Joshi, Hu Xu, Gargi Ghosh, and Luke Zettlemoyer. 2022. HTLM: Hyper-text pre-training and prompting of language models. In *International Conference on Learning Representations*.

Amal Alabdulkarim, Winston Wai-Tai Li, Lara J. Martin, and Mark O. Riedl. 2021. Goal-directed story generation: Augmenting generative language models with reinforcement learning. *ArXiv*, abs/2112.08593.

Rami Aly, Christos Christodoulopoulos, Oana Cocarascu, Zhijiang Guo, Arpit Mittal, Michael Schlichtkrull, James Thorne, and Andreas Vlachos, editors. 2022. *Proceedings of the Fifth Fact Extraction and VERification Workshop (FEVER)*. Association for Computational Linguistics, Dublin, Ireland.

Wazib Ansar and Saptarsi Goswami. 2021. Combating the menace: A survey on characterization and detection of fake news from a data science perspective. *International Journal of Information Management Data Insights*, 1(2):100052.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, T. J. Henighan, Nicholas Joseph, Saurav Kadavath, John Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Christopher Olah, Benjamin Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *ArXiv*, abs/2204.05862.

Vidhisha Balachandran, Hannaneh Hajishirzi, William Cohen, and Yulia Tsvetkov. 2022. Correcting diverse factual errors in abstractive summarization via post-editing and language model infilling. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ankur Bapna and Orhan Firat. 2019. Non-parametric adaptation for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1921–1931, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Bar-Tal, Carl F Graumann, Arie W Kruglanski, and Wolfgang Stroebe. 2013. *Stereotyping and prejudice: Changing conceptions*. Springer Science & Business Media.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

David Beskow. 2020. Finding and Characterizing Information Warfare Campaigns.

Timothy Bickmore, Ha Trinh, Stefan Olafsson, Teresa O'Leary, Reza Asadi, Nathaniel Rickles, and Ricardo Cruz. 2018. Patient and consumer safety risks when using conversational assistants for medical information: an observational study of siri, alexa, and google assistant. *J Med Internet Res*, 20:e11510.

BIG-bench collaboration. 2022. Beyond the imitation game: Measuring and extrapolating the capabilities of language models.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Darya Bogoradnikova, Olesia Makhnytkina, Anton Matveev, Anastasia Zakharova, and Artem Akulov. 2021. Multilingual sentiment analysis and toxicity detection for text messages in russian. In *2021 29th Conference of Open Innovations Association (FRUCT)*, pages 55–64.

Samantha Bradshaw and Philip N Howard. 2019. The global disinformation order: 2019 global inventory of organised social media manipulation.

Luke Breitfeller, Emily Ahn, David Jurgens, and Yulia Tsvetkov. 2019. Finding microaggressions in the wild: A case for locating elusive phenomena in social media posts. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1664–1674, Hong Kong, China. Association for Computational Linguistics.

Hannah Brown, Katherine Lee, FatemehSadat Mireshghallah, R. Shokri, and Florian Tramèr. 2022. What does it mean for a language model to preserve privacy? *2022 ACM Conference on Fairness, Accountability, and Transparency*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Ben Buchanan, Andrew Lohn, Micah Musser, and Katerina Sedova. 2021. Truth, lies, and automation.

Pete Burnap and Matthew L Williams. 2015. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & internet*, 7(2):223–242.

Pete Burnap and Matthew L Williams. 2016. Us and them: identifying cyber hate on twitter across multiple protected characteristics. *EPJ Data science*, 5:1–15.

Ben Burtenshaw and Mike Kestemont. 2021. A Dutch dataset for cross-lingual multilabel toxicity detection. In *Proceedings of the 14th Workshop on Building and Using Comparable Corpora (BUCC 2021)*, pages 75–79, Online (Virtual Mode). INCOMA Ltd.

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. 2020. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, Online. Association for Computational Linguistics.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

J. K. Chambers. 1995. *Sociolinguistic theory: linguistic variation and its social significance*. Oxford.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. In *Proc. ICLR*.

Ting-Yun Chang, Yang Liu, Karthik Gopalakrishnan, Behnam Hedayatnia, Pei Zhou, and Dilek Hakkani-Tur. 2020. Incorporating commonsense knowledge graph in pretrained models for social commonsense tasks. In *Proceedings of Deep Learning Inside Out*

(DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, pages 74–79, Online. Association for Computational Linguistics.

Open AI's Assistant ChatGPT and Andrew M. Perlman. 2022. The implications of openai's assistant for legal services and society. *SSRN Electronic Journal*.

Rajen Chatterjee, Markus Freitag, Matteo Negri, and Marco Turchi. 2020. Findings of the WMT 2020 shared task on automatic post-editing. In *Proceedings of the Fifth Conference on Machine Translation*, pages 646–659, Online. Association for Computational Linguistics.

Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80. IEEE.

Shamil Chollampatt, Raymond Hendy Susanto, Liling Tan, and Ewa Szymanska. 2020. Can automatic post-editing improve nmt? *arXiv preprint arXiv:2009.14395*.

Dennis Chong and James N Druckman. 2007. Framing theory. *Annual review of political science*, 10(1):103–126.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Alexandra Chronopoulou, Dario Stojanovski, and Alexander Fraser. 2020. Reusing a Pretrained Language Model on Languages with Limited Corpora for Unsupervised NMT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2703–2711, Online. Association for Computational Linguistics.

Jennifer Coates. 2016. *Women, Men and Language: A Sociolinguistic Account of Gender Differences in Language*. Routledge.

Marta R Costa-jussà, Eric Smith, Christophe Ropers, Daniel Licht, Javier Ferrando, and Carlos Escolano. 2022. Toxicity in multilingual machine translation at scale. *arXiv preprint arXiv:2210.03070*.

Kimberlé W. Crenshaw. 2017. *On Intersectionality: Essential Writings*. The New Press, New York, NY.

Giovanni Da San Martino, Alberto Barrón-Cedeño, and Preslav Nakov. 2019. Findings of the NLP4IF-2019 shared task on fine-grained propaganda detection. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 162–170, Hong Kong, China. Association for Computational Linguistics.

Giovanni Da San Martino, Alberto Barrón-Cedeño, Henning Wachsmuth, Rostislav Petrov, and Preslav Nakov. 2020. SemEval-2020 task 11: Detection of propaganda techniques in news articles. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1377–1414, Barcelona (online). International Committee for Computational Linguistics.

Maral Dadvar, FMG de Jong, Roeland Ordelman, and Dolf Trieschnigg. 2012. Improved cyberbullying detection using gender information. In *Proceedings of the Twelfth Dutch-Belgian Information Retrieval Workshop (DIR 2012)*. University of Ghent.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *ICLR*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017a. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017b. Automated hate speech detection and the problem of offensive language. In *ICWSM*.

Ryan Daws. 2020. Medical chatbot using OpenAI's GPT-3 told a fake patient to kill themselves.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Emily Denton, Alex Hanna, Razvan Amironesei, Andrew Smart, Hilary Nicole, and Morgan Klaus Scheuerman. 2020. Bringing the people back in: Contesting benchmark machine learning datasets. *arXiv preprint arXiv:2007.07399*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. 2020. Queens are powerful too: Mitigating gender bias in dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8173–8188, Online. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *arXiv preprint arXiv:2104.08758*.

Ashwin Geet d'Sa, Irina Illina, and Dominique Fohr. 2020. Bert and fasttext embeddings for automatic detection of toxic speech. In *2020 International Multi-Conference on:"Organization of Knowledge and Advanced Technologies"(OCTA)*, pages 1–5. IEEE.

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, and Chris Callison-Burch. 2020. RoFT: A tool for evaluating human detection of machine-generated text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 189–196, Online. Association for Computational Linguistics.

Penelope Eckert and Sally McConnell-Ginet. 2003. *Language and Gender*. Cambridge University Press.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *ACL (1)*, pages 2214–2220.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2021. Augmenting Transformers with KNN-Based Composite Memory for Dialog. *Transactions of the Association for Computational Linguistics*, 9:82–99.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Anna Feldman, Giovanni Da San Martino, Chris Leberknight, and Preslav Nakov. 2021. Proceedings of the fourth workshop on nlp for internet freedom: Censorship, disinformation, and propaganda. In *Proceedings of the Fourth Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda*.

Anjalie Field, Su Lin Blodgett, Zeerak Waseem, and Yulia Tsvetkov. 2021. A survey of race, racism, and anti-racism in NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1905–1925, Online. Association for Computational Linguistics.

Anjalie Field and Yulia Tsvetkov. 2020. Unsupervised discovery of implicit gender bias. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 596–608.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online*, pages 85–90.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Mor Geva, Avi Caciularu, Ke Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *ArXiv*, abs/2203.14680.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166.

Nancy Gleason. 2022. Chatgpt and the rise of ai writers: how should higher education respond?

Tanya Goyal and Greg Durrett. 2020. Evaluating factuality in generation with dependency-level entailment. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3592–3603, Online. Association for Computational Linguistics.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Melissa R.H. Hall, Laurens van der Maaten, Laura Gustafson, and Aaron B. Adcock. 2022. A systematic study of bias amplification. *arXiv preprint arXiv:2301.11305*.

Xiaochuang Han and Yulia Tsvetkov. 2020. Fortifying toxic speech detectors against veiled toxicity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7732–7739, Online. Association for Computational Linguistics.

Xiaochuang Han and Yulia Tsvetkov. 2021. Influence tuning: Demoting spurious correlations via instance attribution and instance-driven updates. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4398–4409, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021a. Efficient nearest neighbor language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5703–5714, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zexue He, Bodhisattwa Prasad Majumder, and Julian McAuley. 2021b. Detect and perturb: Neutral rewriting of biased and sensitive text via gradient-based decoding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4173–4181, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Peter Henderson, Mark S. Krass, Lucia Zheng, Neel Guha, Christopher D. Manning, Dan Jurafsky, and Daniel E. Ho. 2022. Pile of law: Learning responsible data filtering from the law and a 256gb open-source legal dataset. *NeurIPS*.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Daniel Hershcovich, Stella Frank, Heather Lent, Miryam de Lhoneux, Mostafa Abdou, Stephanie Brandl, Emanuele Bugliarello, Laura Cabello Piqueras, Ilias Chalkidis, Ruixiang Cui, Constanza Fierro, Katerina Margatina, Phillip Rust, and Anders Søgaard. 2022. Challenges and strategies in cross-cultural NLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6997–7013, Dublin, Ireland. Association for Computational Linguistics.

Cong Duy Vu Hoang, Gholamreza Haffari, and Trevor Cohn. 2017. Towards decoding as continuous optimisation in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 146–156, Copenhagen, Denmark. Association for Computational Linguistics.

Janet Holmes and Nick Wilson. 2017. *An introduction to sociolinguistics*. Routledge.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Nabil Hossain, Marjan Ghazvininejad, and Luke Zettlemoyer. 2020. Simple and effective retrieve-edit-rerank text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2532–2538, Online. Association for Computational Linguistics.

Kung-Hsiang Huang, Preslav Nakov, Yejin Choi, and Heng Ji. 2022. Faking fake news for real fake news detection: Propaganda-loaded training data generation. *ArXiv*, abs/2203.05386.

Luyang Huang, Lingfei Wu, and Lu Wang. 2020. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5094–5107, Online. Association for Computational Linguistics.

Elle Hunt. 2016. Tay, Microsoft's AI chatbot, gets a crash course in racism from Twitter.

Ben Hutchinson, Andrew Smart, Alex Hanna, Emily Denton, Christina Greer, Oddur Kjartansson, Parker Barnes, and Margaret Mitchell. 2021. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 560–575, New York, NY, USA. Association for Computing Machinery.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.

Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A. Choquette-Choo, and Nicholas Carlini. 2022. Preventing verbatim memorization in language models gives a false sense of privacy. *ArXiv*, abs/2210.17546.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Alon Jacovi, Ana Marasović, Tim Miller, and Yoav Goldberg. 2021. Formalizing trust in artificial intelligence: Prerequisites, causes and goals of human trust in ai. *Proc. FAccT*.

Heesoo Jang. 2021. A South Korean chatbot shows just how sloppy tech companies can be with user data.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2022. Knowledge unlearning for mitigating privacy risks in language models. *ArXiv*, abs/2210.01504.

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 725–736, Online. Association for Computational Linguistics.

Liwei Jiang, Jena D. Hwang, Chandrasekhar Bhagavatula, Ronan Le Bras, Maxwell Forbes, Jon Borchardt, Jenny Liang, Oren Etzioni, Maarten Sap, and Yejin Choi. 2021. Delphi: Towards machine ethics and norms. *ArXiv*, abs/2110.07574.

Eun Seo Jo and Timnit Gebru. 2020. Lessons from archives: Strategies for collecting sociocultural data in machine learning. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 306–316, New York, NY, USA. Association for Computing Machinery.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Amina Kammoun, Rim Slama, Hedi Tabia, Tarek Ouni, and Mohmed Abid. 2022. Generative adversarial networks for face generation: A survey. *ACM Computing Surveys*.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. *ICML*.

Gavin Kerrigan, Dylan Slack, and Jens Tuyls. 2020. Differentially private language models benefit from public pre-training. In *Proceedings of the Second Workshop on Privacy in NLP*, pages 39–45, Online. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *ACM Computing*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Dongwoo Kim. 2016. Chatbot gone awry starts conversations about ai ethics in south korea.

Daniel King, Zejiang Shen, Nishant Subramani, Daniel S Weld, Iz Beltagy, and Doug Downey. 2022. Don't say what you don't know: Improving the consistency of abstractive summarization by constraining beam search. *arXiv preprint arXiv:2203.08436*.

Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R. Rickford, Dan Jurafsky, and Sharad Goel. 2020. Racial disparities in automated speech recognition. *Proceedings of the National Academy of Sciences*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Renard Korzeniowski, Rafal Rolczynski, Przemyslaw Sadownik, Tomasz Korbak, and Marcin Możejko. 2019. Exploiting unsupervised pre-training and automated feature engineering for low-resource hate speech detection in polish. *ArXiv*, abs/1906.09325.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kalpesh Krishna, Ya yin Chang, John Wieting, and Mohit Iyyer. 2022. Rankgen: Improving text generation with large ranking models. *EMNLP*.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Sachin Kumar, Antonios Anastasopoulos, Shuly Wintner, and Yulia Tsvetkov. 2021a. Machine translation into low-resource language varieties. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 110–121, Online. Association for Computational Linguistics.

Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. 2021b. Controlled text generation as continuous optimization with multiple constraints. In *Proc. NeurIPS*.

Sachin Kumar, Biswajit Paria, and Yulia Tsvetkov. 2022. Gradient-based constrained sampling from language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2251–2277.

Sachin Kumar, Shuly Wintner, Noah A. Smith, and Yulia Tsvetkov. 2019. Topics to avoid: Demoting latent confounds in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4153–4163, Hong Kong, China. Association for Computational Linguistics.

Keita Kurita, Anna Belova, and Antonios Anastasopoulos. 2019. Towards robust toxic content classification. *arXiv preprint arXiv:1912.06872*.

Hwanhee Lee, Cheoneum Park, Seunghyun Yoon, Trung Bui, Franck Dernoncourt, Juae Kim, and Kyomin Jung. 2022a. Factual error correction for abstractive summaries using entity retrieval. In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 439–444, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022b. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445, Dublin, Ireland. Association for Computational Linguistics.

Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron Wallace. 2021. Does BERT pretrained on clinical notes reveal sensitive data? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 946–959, Online. Association for Computational Linguistics.

João Augusto Leite, Diego Silva, Kalina Bontcheva, and Carolina Scarton. 2020. Toxic language detection in social media for Brazilian Portuguese: New dataset and multilingual analysis. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 914–924, Suzhou, China. Association for Computational Linguistics.

Shahar Levy, Koren Lazar, and Gabriel Stanovsky. 2021. Collecting a large-scale gender bias dataset for coreference resolution and machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2470–2480, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Haoran Li, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. 2018. Ensure the correctness of the summary: Incorporate entailment knowledge into abstractive sentence summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1430–1441, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori B. Hashimoto. 2022. Large language models can be strong differentially private learners. *ICLR*.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards understanding and mitigating social biases in language models. In *International Conference on Machine Learning*, pages 6565–6576. PMLR.

Inna Wanyin Lin, Lucille Njoo, Anjalie Field, Ashish Sharma, Katharina C. H. Reinecke, Tim Althoff, and Yulia Tsvetkov. 2022a. Gendered mental health stigma in masked language models. *EMNLP*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022b. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Ves Stoyanov, and Xian Li. 2021. Few-shot learning with multilingual language models. *ArXiv*, abs/2112.10668.

Zachary C Lipton. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021a. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online. Association for Computational Linguistics.

Haochen Liu, Jamell Dacon, Wenqi Fan, Hui Liu, Zitao Liu, and Jiliang Tang. 2020. Does gender matter? towards fairness in dialogue systems. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4403–4416, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9).

Qi Liu, Dani Yogatama, and Phil Blunsom. 2022. Relational Memory-Augmented Language Models. *Transactions of the Association for Computational Linguistics*, 10:555–572.

Ruibo Liu, Chenyan Jia, Jason Wei, Guangxuan Xu, Lili Wang, and Soroush Vosoughi. 2021b. Mitigating political bias in language models through reinforced calibration. In *AAAI*.

Shuhua Liu and Thomas Forss. 2015. New classification models for detecting hate and violence web content. In *2015 7th international joint conference on knowledge discovery, knowledge engineering and knowledge management (IC3K)*, volume 1, pages 487–495. IEEE.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022. QUARK: Controllable text generation with reinforced unlearning. In *Advances in Neural Information Processing Systems*.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, Online. Association for Computational Linguistics.

Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. 2020. PowerTransformer: Unsupervised controllable revision for biased language correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7426–7441, Online. Association for Computational Linguistics.

Jimit Majmudar, Christophe Dupuy, Charith S. Peris, Sami Smaili, Rahul Gupta, and Richard S. Zemel. 2022. Differentially private decoding in large language models. *ArXiv*, abs/2205.13621.

Yuning Mao, Xiang Ren, Heng Ji, and Jiawei Han. 2020. Constrained abstractive summarization: Preserving factual consistency with constrained generation. *ArXiv*, abs/2010.12723.

Giovanni Da San Martino, Stefano Cresci, Alberto Barrón-Cedeño, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4826–4832. International Joint Conferences on Artificial Intelligence Organization. Survey track.

Binny Mathew, Hardik Tharad, Subham Rajgaria, Prajwal Singhania, Suman Kalyan Maity, Pawan Goyal, and Animesh Mukherjee. 2018. Thou shalt not hate: Countering online hate speech. In *International Conference on Web and Social Media*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proc. ACL*.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation. *ArXiv*, abs/2202.00666.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. In *International Conference on Learning Representations*.

Simon Mille, Kaustubh Dhole, Saad Mahamood, Laura Perez-Beltrachini, Varun Gangal, Mihir Kale, Emiel van Miltenburg, and Sebastian Gehrmann. 2021. Automatic construction of evaluation suites for natural language generation datasets. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generationusing energy language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 401–415, Dublin, Ireland. Association for Computational Linguistics.

Fatemehsadat Mirshghallah, Mohammadkazem Taram, Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Hadi Esmaeilzadeh. 2020. Privacy in deep learning: A survey. *ArXiv*, abs/2004.12254.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022. Fast model editing at scale. In *International Conference on Learning Representations*.

Jihyung Moon, Won Ik Cho, and Junbum Lee. 2020. BEEP! Korean corpus of online news comments for toxic speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 25–31, Online. Association for Computational Linguistics.

Hamdy Mubarak, Kareem Darwish, and Walid Magdy. 2017. Abusive language detection on arabic social media. In *Proceedings of the first workshop on abusive language online*, pages 52–56.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barr'on-Cedeno, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated fact-checking for assisting human fact-checkers. In *International Joint Conference on Artificial Intelligence*.

Feng Nan, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. 2021. Improving factual consistency of abstractive summarization via question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6881–6894, Online. Association for Computational Linguistics.

Helen Ngo, Cooper Raterink, João GM Araújo, Ivan Zhang, Carol Chen, Adrien Morisot, and Nicholas Frosst. 2021. Mitigating harm in language models with conditional-likelihood filtration. *arXiv preprint arXiv:2108.07790*.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2021. HONEST: Measuring hurtful sentence completion in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2398–2406, Online. Association for Computational Linguistics.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2020. A survey on natural language processing for fake news detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6086–6093, Marseille, France. European Language Resources Association.

Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. 2019a. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4675–4684, Hong Kong, China. Association for Computational Linguistics.

Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. 2019b. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4675–4684, Hong Kong, China. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. 2022. Training language models to follow instructions with human feedback. *NeurIPS*.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2021. Towards multidomain and multilingual abusive language detection: a survey. *Personal and Ubiquitous Computing*, pages 1–27.

Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3973–3997, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sajan Patel and Kyle Lam. 2023. Chatgpt: the future of discharge summaries? *The Lancet. Digital health.*

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deep learning for user comment moderation. In *Proceedings of the First Workshop on Abusive Language Online*, pages 25–35, Vancouver, BC, Canada. Association for Computational Linguistics.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nathan McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. In *Conference on Empirical Methods in Natural Language Processing*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Georgios K Pitsilis, Heri Ramampiaro, and Helge Langseth. 2018. Effective hate-speech detection in twitter data using recurrent neural networks. *Applied Intelligence*, 48(12):4730–4742.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically neutralizing subjective bias in text. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):480–489.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 794–805, Online. Association for Computational Linguistics.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021.

Scaling language models: Methods, analysis & insights from training gopher. arXiv.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Inioluwa Deborah Raji, Andrew Smart, Rebecca N White, Margaret Mitchell, Timnit Gebru, Ben Hutchinson, Jamila Smith-Loud, Daniel Theron, and Parker Barnes. 2020. Closing the ai accountability gap: Defining an end-to-end framework for internal algorithmic auditing. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 33–44.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. 2022. Is reinforcement learning (not) for natural language processing?: Benchmarks, baselines, and building blocks for natural language policy optimization. arXiv.

Maribeth Rauh, John F J Mellor, Jonathan Uesato, Po-Sen Huang, Johannes Welbl, Laura Weidinger, Sumanth Dathathri, Amelia Glaese, Geoffrey Irving, Iason Gabriel, William Isaac, and Lisa Anne Hendricks. 2022. Characteristics of harmful text: Towards rigorous benchmarking of language models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Reuters. 2023. Chatgpt sets record for fastest-growing user base.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2017. Measuring the reliability of hate speech annotations: The case of the european refugee crisis. *arXiv preprint arXiv:1701.08118*.

Nazanin Sabri, Valerio Basile, Tommaso Caselli, et al. 2021. Leveraging bias in pre-trained word embeddings for unsupervised microaggression detection. In *Italian Conference on Computational Linguistics 2021: CLiC-it 2021*. CEUR Workshop Proceedings (CEUR-WS. org).

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and A Noah Smith. 2019. The risk of racial bias in hate speech detection. In *ACL*.

Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A Smith. 2021. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. *arXiv preprint arXiv:2111.07997*.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in nlp. *Computing Research Repository*, arXiv:2103.00453.

Katherine Schulten. 2023. How should schools respond to chatgpt?

Thomas Scialom, Paul-Alexis Dray, Patrick Gallinari, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, and Alex Wang. 2021. Questeval: Summarization asks for fact-based evaluation. In *Conference on Empirical Methods in Natural Language Processing*.

Shaden Shaar, Firoj Alam, Giovanni Da San Martino, and Preslav Nakov. 2021. Assisting the human fact-checkers: Detecting all previously fact-checked claims in a document. In *Conference on Empirical Methods in Natural Language Processing*.

Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2018. The spread of low-credibility content by social bots. *Nature communications*, 9(1):1–9.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2021. Societal biases in language generation: Progress and challenges. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4275–4293, Online. Association for Computational Linguistics.

Weiyan Shi, Aiqi Cui, Evan Li, R. Jia, and Zhou Yu. 2021. Selective differential privacy for language modeling. In *North American Chapter of the Association for Computational Linguistics*.

Oleh Shliazhko, Alena Fenogenova, Maria Tikhonova, Vladislav Mikhailov, Anastasia Kozlova, and Tatiana Shavrina. 2022. mgpt: Few-shot learners go multilingual. arXiv.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *ArXiv*, abs/1909.08053.

Michel Simard, Nicola Ueffing, Pierre Isabelle, and Roland Kuhn. 2007. Rule-based translation with statistical phrase-based post-editing. In *WMT@ACL*.

Kim So-hyun. 2023. [the korean dilemma] what chatgpt means for korea.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Artūrs Stafanovičs, Toms Bergmanis, and Mārcis Pinnis. 2020. Mitigating gender bias in machine translation with target gender annotations. In *Proceedings of the Fifth Conference on Machine Translation*, pages 629–638, Online. Association for Computational Linguistics.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan J. Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. *NeurIPS*, abs/2009.01325.

Chris Stokel-Walker. 2023. Chatgpt listed as author on research papers: many scientists disapprove.

Fan-Keng Sun, Cheng-Hao Ho, and Hung yi Lee. 2020. Lamol: Language modeling for lifelong language learning. In *ICLR*.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Rohan Taori and Tatsunori Hashimoto. 2022. Data feedback loops: Model-driven amplification of dataset biases. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Ranjana Mary Varghese. 2023. Will chatgpt shake up higher education in india?

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

James Vincent. 2022. YouTuber trains AI bot on 4chan's pile o' bile with entirely predictable results.

David Wan and Mohit Bansal. 2022. Factpegasus: Factuality-aware pre-training and fine-tuning for abstractive summarization. In *NAACL 2022*.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Boxin Wang, Wei Ping, Chaowei Xiao, Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Bo Li, Anima Anandkumar, and Bryan Catanzaro. 2022. Exploring the limits of domain-adaptive training for detoxifying large-scale language models. *NeurIPS*.

Han Wang, Yang Liu, Chenguang Zhu, Linjun Shou, Ming Gong, Yichong Xu, and Michael Zeng. 2021a. Retrieval enhanced model for commonsense generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3056–3062, Online. Association for Computational Linguistics.

Luyu Wang, Yujia Li, Ozlem Aslan, and Oriol Vinyals. 2021b. WikiGraphs: A Wikipedia text - knowledge graph paired dataset. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 67–82, Mexico City, Mexico. Association for Computational Linguistics.

Yu Wang, Yuelin Wang, Kai Dang, Jie Liu, and Zhuo Liu. 2021c. A comprehensive survey of grammatical error correction. *ACM Trans. Intell. Syst. Technol.*, 12(5).

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022b. Emergent abilities of large language models. *TMLR*.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sasha Brown, Zac Kenton, Will Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2022. Taxonomy of risks posed by language models. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 214–229, New York, NY, USA. Association for Computing Machinery.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of abusive language: the problem of biased datasets. In *Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies, volume 1 (long and short papers)*, pages 602–608.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the germeval 2018 shared task on the identification of offensive language.

Genta Indra Winata, Andrea Madotto, Zhaojiang Lin, Rosanne Liu, Jason Yosinski, and Pascale Fung. 2021. Language models are few-shot multilingual learners. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 1–15, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

Marty J Wolf, Keith W Miller, and Frances S Grodzinsky. 2017. Why we should have seen that coming: comments on Microsoft's Tay "experiment," and wider implications. *The ORBIT Journal*, 1(2):1–12.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Alexandros Xenos, John Pavlopoulos, and Ion Androutsopoulos. 2021. Context sensitivity estimation in toxicity detection. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 140–145, Online. Association for Computational Linguistics.

Mengzhou Xia, Anjalie Field, and Yulia Tsvetkov. 2020. Demoting racial bias in hate speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 7–14, Online. Association for Computational Linguistics.

Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1980–1984.

Tong Xiang, Sean MacAvaney, Eugene Yang, and Nazli Goharian. 2021. ToxCCIn: Toxic content classification with interpretability. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 1–12, Online. Association for Computational Linguistics.

Adrian Yijie Xu. 2020. Generating fake news with openai's language models.

Albert Xu, Eshaan Pathak, Eric Wallace, Suchin Gururangan, Maarten Sap, and Dan Klein. 2021. Detoxifying language models risks marginalizing minority voices. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2390–2397, Online. Association for Computational Linguistics.

Jing Xu, Da Ju, Margaret Li, Y-Lan Boureau, Jason Weston, and Emily Dinan. 2020. Recipes for safety in open-domain chatbots. *arXiv preprint arXiv:2010.07079*.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proc. NAACL*.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2022. A survey of knowledge-enhanced text generation. *ACM Comput. Surv.* Just Accepted.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2020. Defending against neural fake news. *Neurips*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, P. Zhang, Yuxiao Dong, and Jie Tang. 2023. Glm-130b: An open bilingual pre-trained model. *ICLR*.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022a. A survey of controllable text generation using transformer-based pre-trained language models. arXiv.

Houyu Zhang, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2020a. Grounded conversation generation as guided traverses in commonsense knowledge graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2031–2043, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022b. Opt: Open pre-trained transformer language models. arXiv.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proc. of EMNLP*, pages 2979–2989.

Zheng Zhao, Shay B. Cohen, and Bonnie Webber. 2020. Reducing quantity hallucinations in abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2237–2249, Online. Association for Computational Linguistics.

Xinyi Zhou and Reza Zafarani. 2020. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Comput. Surv.*, 53(5).

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2021. Enhancing factual consistency of abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 718–733, Online. Association for Computational Linguistics.

Chenguang Zhu, Yichong Xu, Xiang Ren, Bill Yuchen Lin, Meng Jiang, and Wenhao Yu. 2022. Knowledge-augmented methods for natural language processing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 12–20, Dublin, Ireland. Association for Computational Linguistics.

Terry Yue Zhuo, Yujin Huang, Chunyang Chen, and Zhenchang Xing. 2023. Exploring ai ethics of chatgpt: A diagnostic analysis. *ArXiv*, abs/2301.12867.

A Background: More Details

Since we focus on language generation, we use the term *language models* (LMs) to refer to their classic definition as generative models (or decoders), which predict the next token given the preceding generated context. For the purposes of this survey, this paradigm also subsumes conditional (or sequence-to-sequence) LMs conditioned on inputs from different modalities such as text, image, or speech via an encoder. [4] Unless otherwise specified, we assume that (1) the LM decoder is parameterized by a transformer architecture (Vaswani et al., 2017), and (2) the LM is first pretrained on a large amount of text (ranging from 100-billions to trillions of tokens), which, together with their large number of parameters, have earned such models the name large language models.[5]. After pretraining, LMs are either used in a zero- or few-shot manner (Brown et al., 2020), or modified for specific tasks via finetuning all or some of their parameters (Liu et al., 2023).

The generation tasks this survey focuses on can be broadly categorized as either (1) transformation tasks, where a given input is transformed into a textual output such as machine translation, abstractive summarization, data-to-text generation, and stylistic re-writing, among others (Prabhumoye et al., 2018; Raffel et al., 2020; Zhang et al., 2020b; Aghajanyan et al., 2022), (2) or open-ended tasks such as dialogue generation, prompt-based autocompletion, story generation, and more (Adiwardana et al., 2020; Guan et al., 2020).

---

[4] While many different strategies to (pre-)train encoder LMs have been introduced in the literature (Devlin et al., 2018; Peters et al., 2018), they are generally not conducive to generating text and are out of scope in this survey.

[5] While some of the studies we will discuss do not rely on pretraining, we highlight it here since it is one of the primary drivers of recent advances in language generation (and its associated risks)

TRAVLR: Now You See It, Now You Don't!
A Bimodal Dataset for Evaluating Visio-Linguisic Reasoning

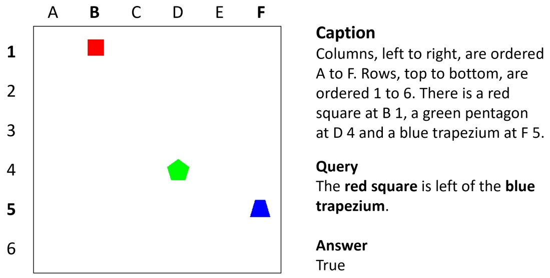
Keng Ji Chow[♮λ*] **Samson Tan**[♮§*] **Min-Yen Kan**[♮]
[♮]Department of Computer Science, National University of Singapore
[λ]Department of English Language and Literature, National University of Singapore
[§]AWS AI Research & Education
kengjichow@u.nus.edu
{samson.tmr, kanmy}@comp.nus.edu.sg

Abstract

Numerous visio-linguistic (V+L) representation learning methods have been developed, yet existing datasets do not adequately evaluate the extent to which they represent visual and linguistic concepts in a unified space. We propose several novel evaluation settings for V+L models, including *cross-modal transfer*. Furthermore, existing V+L benchmarks often report global accuracy scores on the entire dataset, making it difficult to pinpoint the specific reasoning tasks that models fail and succeed at. We present TRAVLR, a synthetic dataset comprising four V+L reasoning tasks. TRAVLR's synthetic nature allows us to constrain its training and testing distributions along task-relevant dimensions, enabling the evaluation of out-of-distribution generalisation. Each example in TRAVLR redundantly encodes the scene in two modalities, allowing either to be dropped or added during training or testing without losing relevant information. We compare the performance of four state-of-the-art V+L models, finding that while they perform well on test examples from the same modality, they all fail at cross-modal transfer and have limited success accommodating the addition or deletion of one modality. We release TRAVLR as an open challenge for the research community.[1]

1 Introduction

Research in psycholinguistics has found that human processing of spatial words activates brain regions associated with the visual system (Tang et al., 2021), suggesting the latter's involvement in processing linguistic input. It is reasonable to expect multimodal neural models to resemble humans in being able to leverage capabilities in the visual domain to solving problems in the text domain, and vice versa. Following its recent success in the text domain (Devlin et al., 2019), the pretraining–fine-tuning paradigm has been applied to the vision and text modalities to create unified visio-linguistic (V+L) representations. Just as pretrained multilingual models have been shown capable of zero-shot cross-lingual transfer on various NLP tasks (Conneau et al., 2020), we may expect *true* V+L models to be capable of generalising to a modality not seen during fine-tuning.

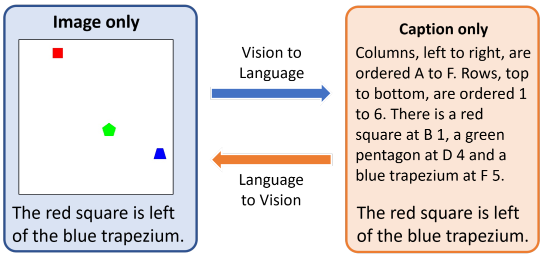


Caption
Columns, left to right, are ordered A to F. Rows, top to bottom, are ordered 1 to 6. There is a red square at B 1, a green pentagon at D 4 and a blue trapezium at F 5.

Query
The **red square** is left of the **blue trapezium**.

Answer
True

(a) A complete example for the spatiality task.



The red square is left of the blue trapezium.

Columns, left to right, are ordered A to F. Rows, top to bottom, are ordered 1 to 6. There is a red square at B 1, a green pentagon at D 4 and a blue trapezium at F 5.

The red square is left of the blue trapezium.

(b) Possible directions of cross-modal transfer.

Figure 1: An example from TRAVLR (a). Both image and caption fully represent the scene; either can be dropped during training/testing, enabling the evaluation of cross-modal transfer ability (b).

However, current approaches for benchmarking V+L models involve reporting global accuracy scores on the entire dataset, rendering the specific sources of success and failure difficult to diagnose (Ribeiro et al., 2020; Goel et al., 2021). For instance, Visual Question Answering (VQA, Goyal et al. 2017) tasks may allow models to exploit dataset bias (Dancette et al., 2021), or may reduce to object recognition problems which do not evaluate the models' ability to perform more

---

[*] Equal contribution

[1] Code and dataset available at `https://github.com/kengjichow/TraVLR`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3322–3347
May 2-6, 2023 ©2023 Association for Computational Linguistics

complex tasks, beyond aligning words or phrases in the text to a portion of the image (Hudson and Manning, 2019; Acharya et al., 2019). As Bernardi and Pezzelle (2021) note, the ability to reason over multiple objects and answer relational questions is crucial to the genuine mastery of language.

Datasets such as NLVR2 (Suhr et al., 2019) address this limitation, but do not allow for fine-grained evaluation along specific dimensions (Tan et al., 2021). CLEVR (Johnson et al., 2017) and SHAPEWORLD (Kuhnle and Copestake, 2017) enable targeted evaluations of a V+L model's reasoning abilities but only encode the scene unimodally, as images. Additionally, their test examples may still be in the training distribution with respect to task-relevant dimensions, making it difficult to draw conclusions about generalisation ability.

We thus contribute TRAVLR, a synthetic dataset comprising four V+L reasoning tasks: spatiality, cardinality, quantifiers, and numerical comparison, all of which require reasoning over multiple objects and have been shown to be challenging for V+L models (Johnson et al., 2017; Parcalabescu et al., 2021). Unlike SHAPEWORLD, we fully leverage the benefits of using a synthetic dataset by controlling the train-test split such that examples in the out-of-distribution (OOD) test set are OOD *with respect to task-relevant dimensions*. We thus argue that TRAVLR serves as a basic sanity check for the abstract reasoning and out-of-distribution generalisation capabilities of models, and is complementary to datasets that evaluate real-world object recognition and compositional reasoning abilities, such as GQA (Hudson and Manning, 2019).

Inspired by the word/picture sentence verification task from psycholinguistics (Goolkasian, 1996), we further propose various novel evaluation settings by representing the scene bimodally as both an image and a caption. First, TRAVLR supports the novel *cross-modal transfer* setting (Figure 1): If pretrained V+L models have learnt a truly multimodal representation, they should be able to learn a reasoning task with input from one modality and perform inference using input from the other modality with little to no extra training. Being able to transfer cross-modally in a zero- or few-shot manner may improve data efficiency in applications where diverse image data is difficult to obtain. Furthermore, models should also succeed on test settings where either an unseen modality is added, or a seen modality is dropped.

Using TRAVLR, we perform extensive analysis of the ability of four Transformer-based V+L models to perform various reasoning tasks. We show that current V+L models:

- May require unreasonably large amounts of data to learn simple visio-linguistic reasoning tasks.
- Exhibit a strong textual bias.
- Are unable to perform cross-modal transfer. We thus pose this as an open challenge for future V+L models.

2 Related Work

V+L tasks and datasets. The Visual Question Answering (VQA) task involves answering a question about an image. It is a complex task as it requires an ability to process input in both visual and textual modalities (Antol et al., 2015). A known issue with VQA datasets is the presence of real-world language priors and statistical biases in the training and testing distribution (Kervadec et al., 2021; Agrawal et al., 2018; Kafle et al., 2019). Although VQA v2.0 (Goyal et al., 2017) was improved by balancing each query with pairs of images, Dancette et al. (2021) show that it still contains both unimodal and multimodal biases that models can exploit. Furthermore, questions in VQA may use non-compositional language that does not require abilities beyond object recognition.

NLVR (Suhr et al., 2017) addresses the lack of compositionality in VQA using synthetic images of abstract 2D shapes, accompanied by human-written English sentences to be judged true or false. NLVR2 (Suhr et al., 2019) and SNLI-VE (Xie et al., 2019) also involve truth-value/entailment judgement tasks, but use photographs instead of synthetic images. Both lack detailed annotations of the specific semantic phenomena evaluated by each example. GQA improves over VQA by focusing on compositional questions that require reasoning over multiple objects and contains detailed annotations (Hudson and Manning, 2019), but still suffers from statistical imbalances and the lack of an out-of-distribution test set (Kervadec et al., 2021).

Other synthetic datasets focusing on reasoning include CLEVR (Johnson et al., 2017), a fully synthetic and annotated 3D dataset, and SHAPEWORLD, a 2D dataset targeting linguistic phenomena such as spatial relationships and quantifiers. SPARTQA (Mirzaee et al., 2021) is another 2D synthetic dataset built upon NLVR focusing on spatial reasoning among other linguistic phenomena.

gSCAN (Ruis et al., 2020) focuses on generalisation of commands within a 2D grid-world.

V+L models. Pretrained V+L models differ in their architecture and pretraining methods. VL-BERT (Su et al., 2019), UNITER (Chen et al., 2020) and VisualBERT (Li et al., 2020a) are single-stream models with a single Transformer while ViLBERT (Lu et al., 2019), LXMERT (Tan and Bansal, 2019), and ALBEF (Li et al., 2021) are dual-stream models which encode image and textual inputs separately before fusing them. These models all use masked language modelling and image-text matching objectives for pretraining, with LXMERT additionally pretraining on VQA and ALBEF using a contrastive loss to align the image and language representations. UNITER, VisualBERT, and LXMERT use a frozen Faster R-CNN (Ren et al., 2015) to extract region-based features from the image, while ALBEF directly encodes the image with a Vision Transformer (Dosovitskiy et al., 2020).

Cross-modal transfer. Prior work has found models trained on multimodal data to perform better on unimodal downstream tasks than models trained only on one modality. Zadeh et al. (2020) found models trained on multimodal input to perform better than text-only models on three NLP tasks, while Testoni et al. (2019) showed that models trained on textual, visual, and auditory input were better at a quantification task than models trained only on a single modality. Using a task involving queries about typical colours of objects, Norlund et al. (2021) found BERT trained on linguistic and visual features to outperform BERT trained on language data filtered for mentions of colour. Frank et al. (2021) investigated the cross-modal alignment of pretrained V+L models with an ablative method based on masked modelling. Lu et al. (2021) propose an image–text fusion model to solve a novel image generation task from a textual description and an image prompt based on NLVR2.

Summary. The datasets commonly used to evaluate V+L models such as VQA and NLVR2 lack fine-grained interpretability, due to the lack of annotations for semantic phenomena involved in each example. Additionally, multiple semantic phenomena co-occur within a single example, making it difficult to control the training distribution and assess the generalisation abilities of models. Furthermore, existing V+L datasets only present the scene in the visual modality and cannot be used to evaluate a V+L model's cross-modal transfer ability.

Existing synthetic datasets (e.g. SHAPEWORLD, CLEVR) fail to split the training and testing distributions along a dimension relevant to the specific task, because they generate captions based on randomly generated images. Unlike existing datasets, TRAVLR fully exploits the benefits of a synthetic dataset by strictly controlling the training and evaluation distributions to test the generalisation abilities of V+L models and avoid statistical biases from language priors and non-uniform distributions.

3 TRAVLR: Cross-Modal Transfer of Visio-Linguistic Reasoning

We construct TRAVLR, a synthetic dataset comprising four visio-linguistic reasoning tasks: spatiality, cardinality, quantifiers and numerical comparison. These tasks were previously identified to be challenging for text-only models (Lin and Su, 2021; Dua et al., 2019; Ravichander et al., 2019). TRAVLR aims to evaluate the extent to which pretrained V+L models already encode or are able to learn these four relations between entities present in input scenes. We first describe the general task format, and then describe each task.

Given a scene with objects, $S = \{o_1, ..., o_n\}$, where each object can be represented as a tuple $<$ *colour, shape, position* $>$, and a textual query q involving some relation $r(o_1, ..., o_i)$ between two or more objects in S, each task involves learning a function $y = f(S, q)$ where $y \in \{\textit{true}, \textit{false}\}$. This is essentially a binary classification task. For instance, in the spatiality task, the relation r (e.g., *above*) compares the positions of two objects. In the numerical comparison task, noun phrases in the query refer to subsets of objects, while the relations (e.g., *more*) compare the cardinality of two sets of objects. Assigning a truth value to the query thus involves reasoning over several objects.

However, a model can never have direct access to the underlying representation scene and must operate on visual or textual forms, and S may be represented in the form of an image or a textual description. In prior work such as VQA, S is presented as an image. In TRAVLR, S is represented bimodally as an $<$ *image, caption* $>$ pair.

Each example consists of an image, an accompanying caption, and a query. Images include abstract objects arranged in a 6 × 6 grid. We draw from five colours and seven shapes, giving 35 unique objects in total. The shapes used in TraVLR are from the Visual Genome (Krishna et al., 2017), a commonly-used pretraining dataset. Each caption fully describes the image with the coordinates of each object (e.g., "There is a red circle at A 1, a blue square at B 2..."). A description of the coordinate system, e.g., "Columns, left to right, are ordered A to F. Rows, top to bottom, are ordered 1 to 6." is prepended to the caption. The caption and query are separated by the `[SEP]` token when presented to the models. Removing the caption reduces our tasks to VQA-like tasks.

3.1 Novel Evaluation Settings

Encoding the scene as both an image and a caption allows models to be trained and evaluated on a combination of three settings: i) image-only, ii) caption-only, and iii) both image and caption input settings. The query is presented as part of the text input in each setting. In the caption-only setting, a blank white image is presented to the models.

3.2 Reasoning Tasks

In contrast to existing synthetic datasets (e.g. ShapeWorld and CLEVR), we do not generate queries post-hoc based on pre-generated scenes, and instead generate scenes constrained along a task-relevant dimension. For instance, in generating the training and out-of-distribution (OOD) test sets for the spatial relationship task, we ensure that the positions of the queried objects do not overlap between the training and test sets *along the relevant axis* (e.g. the horizontal axis for horizontal relations *left/right*). We indicate the train/test splits based on *pairs* in angled brackets.

Spatiality. This task involves queries of the form "The `[object1]` is `[relationship]` the `[object2]`." (e.g., "The red circle is to the right of the blue square."). The possible relationships are *to the left of*, *to the right of*, *above*, *below*. For horizontal relationships (left/right), the train and test sets are split based on the pair <`column(object1), column(object2)`> (see Appendix A.1), while for vertical relationships (above/below), the train–test split is based on the pair <`row(object1),row(object2)`>. This tests the model's ability to generalise its understanding of spatial relationships along the relevant dimension, as opposed to memorising fixed positions.

Cardinality. This task involves queries of the form "There is/are `[number]` `[shape]` object(s)." (e.g., "There are 3 circle objects"). The train and test sets are split by the <`number, shape`> pair occurring in the input image/caption; e.g., instances containing 2 circles and 3 triangles could occur in the training distribution, while instances with just 3 circles occur only in the OOD test distribution.

All	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr2]} \setminus \texttt{[attr1]} >$
Not all	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr1]} \setminus \texttt{[attr2]} >$
No	$< \texttt{[attr1]} \setminus \texttt{[attr2]}, \texttt{[attr2]} \setminus \texttt{[attr1]} >$
Some	$< \texttt{[attr1]} \setminus \texttt{[attr2]}, \texttt{[attr1]} \cap \texttt{[attr2]} >$
Only	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr1]} \setminus \texttt{[attr2]} >$
Not only	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr2]} \setminus \texttt{[attr1]} >$

Table 1: Pairs for each quantifier.

Quantifiers. This task involves queries of the form "`[quantifier]` the `[attr1]` objects are `[attr2]` objects.", where the quantifiers include *all*, *some*, *only* and their negated counterparts *not all*, *none* and *not only*. The train–test split is performed based on the pair $< a, b >$, which varies based on the quantifier, as given in Table 1. For instance, for the relationship *not all*, a is the number of objects which fulfil both `[attr1]` and `[attr2]`, and b is the number of objects which fulfil `[attr1]` but not `[attr2]` (see Appendix A.1).

Numerical comparison. This task involves queries of the form "There are `[more/fewer]` `[attr1]` objects than `[attr2]` objects." (e.g., "There are more circles than squares."). The train and test sets are split by the pair $< a, b >$ where a is the number of `[attr1]` objects, and b is the number of `[attr2]` objects. Instances for which $|a - b|$ is smaller than a threshold is assigned to the training distribution, and the remaining pairs are assigned to testing. $|a - b| \in 1, 3$ and both a and b have a maximal value of 9. Success in this task is indicative of generalisation based on an understanding of numeral scales and transitivity of comparison; i.e., $a > b$ and $b > c$ implies $a > c$.

3.3 Generating TraVLR

To generate examples for each task, we randomly sample object attributes from the distributions defined in Section 3.2, ensuring that the *pairs* relevant to each task do not overlap between the train and OOD test sets. We also ensure that all queries in the OOD test set do not occur in the training set. Distractor objects irrelevant to the intended query are finally added to the scene. The spatiality task's training set comprises 32k examples, the training sets of the other tasks comprise 8k examples each due to differences in the amount of data required for convergence in our preliminary experiments.

	Train	Image				Caption				Image + Caption			
	Test	Image	Caption	Img. + Cap.	#sd	Image	Caption	Img. + Cap.	#sd	Image	Caption	Img. + Cap.	#sd
S	VisualBERT	65.94 (-1.02)	**48.92 (-0.23)**	**52.47 (+0.19)**	3	**49.40 (+0.09)**	93.55 (-6.45)	93.46 (-6.54)	2	49.34 (+0.31)	70.99 (-1.59)	71.39 (-1.65)	3
	UNITER	89.67 (+0.96)	44.36 (+0.48)	46.15 (-0.19)	3	37.66 (+0.54)	92.31 (-7.67)	92.23 (-7.74)	1	**50.15 (+0.17)**	70.75 (+1.24)	71.17 (+0.40)	1
	LXMERT	**99.46 (-0.38)**	38.87 (-0.27)	48.82 (+0.33)	3	33.52 (+0.47)	33.52 (+0.47)	33.52 (+0.47)	0	33.52 (+0.47)	33.52 (+0.47)	33.52 (+0.47)	0
	ALBEF	48.28 (+0.10)	44.54 (-0.26)	44.85 (-0.18)	0	48.75 (+0.52)	**98.42 (-1.58)**	**98.42 (-1.58)**	1	48.56 (+0.94)	**93.66 (-6.34)**	**93.31 (-6.69)**	2
C	VisualBERT	77.41 (+1.11)	33.14 (+0.01)	46.94 (-0.53)	3	**46.63 (-0.03)**	**99.99 (+0.08)**	**99.99 (+0.16)**	3	45.20 (-1.41)	**99.94 (+0.10)**	**99.94 (+0.18)**	3
	UNITER	77.12 (+0.09)	**42.36 (-0.35)**	48.55 (+0.23)	3	41.97 (-0.95)	98.96 (+1.82)	98.99 (+0.82)	3	42.36 (-1.97)	98.48 (+0.11)	98.83 (+1.99)	3
	LXMERT	**82.90 (-1.66)**	33.16 (+0.02)	43.79 (-0.95)	3	45.23 (-0.21)	60.02 (-18.88)	60.10 (-18.98)	3	**50.72 (+0.34)**	55.00 (-1.60)	55.26 (-7.83)	3
	ALBEF	59.19 (+0.43)	32.67 (-0.05)	**53.31 (-0.44)**	2	41.58 (-3.24)	99.61 (+0.17)	99.61 (+0.17)	3	43.17 (-2.33)	99.61 (+0.26)	99.61 (+0.26)	3
Q	VisualBERT	86.59 (-2.73)	45.09 (+1.16)	**60.93 (-2.25)**	3	**49.22 (+0.63)**	**99.99 (-0.01)**	**99.99 (-0.01)**	3	**49.51 (-0.16)**	**99.98 (-0.01)**	**99.98 (-0.02)**	3
	UNITER	95.14 (-1.10)	**48.89 (+0.62)**	53.99 (-0.13)	3	48.64 (-0.99)	99.42 (-0.41)	99.36 (-0.44)	3	48.07 (-1.14)	97.85 (+2.11)	98.87 (+1.43)	3
	LXMERT	**96.72 (-0.94)**	34.87 (-0.27)	39.94 (-0.69)	3	43.33 (-0.98)	92.91 (+6.58)	90.65 (+7.02)	3	48.95 (-0.14)	33.93 (-0.01)	51.02 (+0.17)	0
	ALBEF	66.19 (-0.19)	43.88 (+0.21)	54.78 (-1.32)	3	48.27 (+0.19)	99.98 (+0.12)	99.98 (+0.12)	3	47.73 (+0.19)	99.98 (+0.17)	99.97 (+0.17)	3
N	VisualBERT	58.75 (-14.16)	40.27 (+1.26)	**53.13 (-2.81)**	3	**49.62 (-0.42)**	**99.77 (-0.06)**	**99.75 (+0.00)**	3	**51.10 (+0.20)**	**99.79 (-0.11)**	**99.79 (-0.10)**	3
	UNITER	**63.08 (-22.07)**	45.52 (+0.81)	52.97 (-0.62)	3	49.32 (-0.59)	69.43 (-30.34)	68.77 (-30.91)	3	46.78 (-1.03)	64.80 (-34.71)	63.88 (-35.92)	3
	LXMERT	62.56 (-21.57)	**45.85 (-0.34)**	46.62 (+0.59)	3	48.62 (-0.94)	53.10 (-46.30)	53.15 (-46.21)	3	50.68 (+2.31)	57.45 (-26.77)	56.81 (-41.86)	2
	ALBEF	41.93 (-4.30)	41.23 (-1.27)	33.88 (-3.13)	0	48.00 (-0.82)	97.96 (-1.88)	97.96 (-1.88)	3	47.24 (-1.50)	98.93 (-0.91)	98.93 (-0.91)	3

Table 2: Mean F_1 scores on the TRAVLR OOD test sets of four V+L reasoning tasks (**S:** Spatiality, **C:** Cardinality, **Q:** Quantifiers, **N:** Numerical comparison; relative change from InD results are in parentheses). #sd indicates the number of converged runs. We report the mean of converged runs if #sd > 1, and the mean of all runs if #sd $= 0$. Above-random results are underlined; the result with the highest mean in each column is **bolded**.

Prior work on generalisation evaluation recommends the use of in- and out-of-distribution (henceforth InD and OOD, respectively) test sets (Csordás et al., 2021). Hence, we include validation and InD test sets randomly sampled from the training distribution (10k examples each), in addition to the OOD test set described above (20k examples). We report further dataset statistics in Appendix A.

4 Experiments

Models. We perform experiments on two single-stream models, VisualBERT and UNITER, and two dual-stream models, LXMERT and ALBEF. We use Li et al. (2020b)'s implementation of VisualBERT, LXMERT, and UNITER, and the original implementation of ALBEF. The image features of LXMERT and UNITER are 36 regions of interest extracted using Tan and Bansal (2019)'s implementation of a pretrained Faster R-CNN (Ren et al., 2015; Anderson et al., 2018). We use a Detectron model (Girshick et al., 2018) to extract image features for VisualBERT. We also train two text-only models, BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), as caption-only baselines.

Setup. We train models on each task for 80 epochs. Following Csordás et al. (2021)'s finding that early stopping may lead to underestimation of model performance, we do not do early stopping. We use the recommended hyperparameters for fine-tuning ALBEF on SNLI-VE (Xie et al., 2019) and VisualBERT on NLVR2. Learning rates were adjusted downwards for models/tasks where recommended hyperparameters did not lead to convergence. Each experiment is repeated with three random seeds. Table 2 reports mean results on seeds leading to above-random performance, and mean results on all seeds when none achieve above-random performance. We present further experimental details in Appendix B.

4.1 Within-Modality Results

We first discuss the results of testing the model on the modality it was trained on (Table 2). In Appendix C, we examine the effect of dataset size on model performance and discuss the results in detail. In Appendix D, we discuss differences in training duration between models.

Image-only setting. Generalising across the four tasks, in the image-only setting, LXMERT is the best performing model across all tasks, while ALBEF is consistently the worst performing model. UNITER outperforms VisualBERT in all tasks except the cardinality task.

Caption-only and image+caption settings. Generally, the performance of all models except LXMERT in the caption-only and image+caption settings is better than performance in the image-only setting. Across tasks and models, performance in the caption-only closely resembles performance in image+caption settings, suggesting a strong textual bias when both modalities are presented. In the caption-only and image+caption settings, LXMERT is consistently the worse performing model, and VisualBERT is the best performing model in all but the spatiality task. On the cardinality and quantifiers tasks, all models except LXMERT achieve close to a perfect F_1 score, performing similarly to RoBERTa and BERT. However, they are outperformed by RoBERTa when trained on smaller datasets.

Spatiality. While UNITER and LXMERT achieve above-random performance with 8k examples, VisualBERT requires at least 16k examples, and ALBEF fails to learn the task on the full 32k dataset. 32k is a significant number of examples given the task's simplicity. For comparison, the full VQA dataset consists of only 443k training examples. Transformer-based models thus face similar issues to CNN and LSTM models, which Johnson et al. (2017) found to have trouble learning spatial relationships and often memorise the absolute object positions.

A potential explanation for the superior performance of UNITER and LXMERT could be that unlike the other models, spatial coordinates from the bounding boxes are explicitly encoded in the input to the image encoders, which the models can directly exploit. This is unavailable to ALBEF, which takes in the image as input directly instead of relying on a separate object detector. VisualBERT does not make use of these spatial coordinates, which may impair its ability to relate the positions of objects. Bugliarello et al. (2021) and Frank et al. (2021) posited this limitation of VisualBERT to explain its poor performance on tasks such as RefCOCO+ and Masked Region classification, but the impact of this limitation on spatial reasoning has hitherto not been directly investigated. LXMERT converges in fewer epochs compared to all other models. LXMERT only required 4 epochs of training on the 32k dataset to exceed F_1=90 on the InD test set, while UNITER required about 60 epochs (see further discussion in Appendix D).

In the caption-only and image+caption settings, UNITER and ALBEF are notably more sensitive to random seeds and fail to achieve convergence in several cases. Furthermore, performance is poorer in the image+caption than in the caption-only setting across all models. This runs counter to existing expectations that bimodal representation of the same information should improve performance (Zadeh et al., 2020; Testoni et al., 2019). Several models exhibit > 5 point differences in F_1 on the InD and OOD test sets. BERT requires at least 8k examples to achieve an F_1 score above 60, corroborating findings by Lin and Su (2021) that BERT requires a significant number of examples to learn a simple natural language inference task.

Cardinality. On the cardinality task, all models achieve non-random performance when trained with 8k examples. Across all settings, performance on the InD and OOD test set is similar, indicating that models are able to generalise to unseen object–number pairs. However, despite the task's simplicity, in the image-only setting, no model achieves above an F_1 of 85 when trained on the full dataset, and when trained on 1k examples, all models perform poorly with F_1 scores below 60. This corroborates Parcalabescu et al. (2021)'s finding that current V+L models face difficulties with counting objects in images. Models are generally successful in the caption-only setting, corroborating Wallace et al. (2019)'s findings that numeracy is encoded in the embeddings of language-only models.

Quantifiers. All models perform well on the quantifiers task in most settings, with exceptions that conform to the overall trends described above (e.g. LXMERT in the image+caption setting). Good performance on the OOD dataset indicates that models are not memorising specific numbers of objects and instead use more general strategies for understanding quantifiers. This parallels psycholinguistic findings that comprehension of non-exact quantifiers does not correlate with counting skills in human children (Dolscheid et al., 2015).

Numerical comparison. Unlike the other tasks, there is a significant difference between the InD and OOD settings for the numerical comparison task across all settings. In the image-only setting, all models except ALBEF, achieve F_1s above 70 on the InD test set, while achieving much lower F_1s (55–65) on the OOD test set. In the caption-only and image+caption settings, while all models achieve close to F_1=100 on the InD test set, UNITER and LXMERT do not generalise well to the OOD test set, showing a substantial drop in performance. Our results suggest that models face difficulties generalising beyond the training distribution to unseen number pairs.

4.2 Cross-Modal Transfer

Despite performing well in the within-modality settings, **none** of the models succeed at performing zero-shot cross-modal transfer to an unseen modality. Nevertheless, given the success of few-shot learning, we may expect few-shot cross-modal transfer to be plausible. We hence conduct few-shot learning experiments on the best performing models that were fine-tuned on the full dataset of the spatiality and quantifiers tasks. We fine-tuned the models on 200 examples of the unseen modality for 20 epochs before testing (see details in Appendix B.5). Unfortunately, none of the models achieved above random performance even after this additional fine-tuning. We conclude that they can perform neither zero-shot nor few-shot cross-modal transfer, and that existing V+L representation learning methods have yet to succeed at producing truly multimodal (or modality-agnostic) representations.

4.3 Adding and Dropping Modalities

We now discuss the effects of either adding or dropping a modality to the input presented during testing. First, models trained in the image+caption setting perform similarly when tested in the image+caption and caption-only settings. However, when tested in the image-only setting, models perform poorly, exhibiting (close to) random performance in most cases. This indicates that all models tend to overly focus on the caption during training. Second, models trained only on captions perform similarly when tested in the image+caption setting. In contrast, testing a model trained only on images in the image+caption setting results in a significant performance drop. In most cases, F_1 is close to random, although all models except LXMERT manage to maintain above-random performance on the quantifier task. This could indicate that the models are easily distracted by the textual modality. Together with the general similarity between results in the caption-only and image+caption settings, these results indicate a bias towards the textual modality.

5 Further Analyses

5.1 The Impact of Pretraining

We conduct additional experiments on VisualBERT, UNITER and LXMERT trained using the unified VOLTA (Bugliarello et al., 2021) framework, where V+L models were pretrained using the same dataset in a standardised manner. This allows us to understand the extent to which differences in models' performance can be attributed to the pretraining data or objective, rather than model architecture. First, VOLTA LXMERT loses its advantage in both performance and training efficiency over the other models. We suggest that LXMERT's superior performance is due to some aspect of its pretraining, such as the larger size of or presence of VQA examples in its pretraining dataset, or its use of an additional VQA pretraining objective. Furthermore, VOLTA VisualBERT's performance is significantly improved over the original VisualBERT, likely because it was pretrained with more data. UNITER also outperforms the other two models (also reported by Bugliarello et al. (2021) on RefCOCO+ and NLVR2), which suggests some advantage of its specific architecture. All three VOLTA models achieve similar results on the caption-only setting of the 8k cardinality dataset. As all VOLTA models were initialised with BERT weights, we suggest that the poor performance of the original LXMERT on textual input is due to its lack of initialisation with BERT weights (Tan and Bansal, 2019). We discuss the results in greater detail in Appendix E.

5.2 The Impact of Catastrophic Forgetting

A potential concern with fine-tuning on one modality and testing on another is that the models may overfit to the fine-tuning modality, resulting in the catastrophic forgetting of cross-modal information. To reduce this possibility, we freeze the pretrained representations and only fine-tune the classification layers. In this setting, all models except LXMERT fail to go beyond random performance. Although LXMERT achieves above-random performance in the image-only setting for two tasks, it is still relatively poor (Spatiality: OOD=52.29, InD=52.67; Cardinality: OOD= 56.39, InD=60.88). This indicates that most models require representation fine-tuning to perform reasoning tasks, and it is unlikely that the inability to transfer cross-modally was due to catastrophic forgetting during fine-tuning.

5.3 Modal Dropout

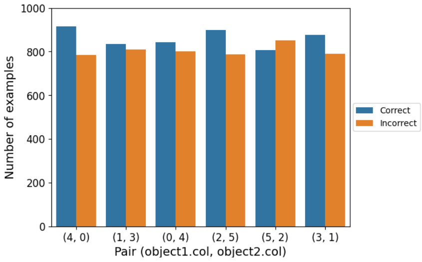
One may be concerned that the presentation of the same information in the image+caption setting facilitates overfitting to either modality. We investigate whether models' textual bias can be overcome by randomly "dropping out" either the image or caption input when training in the image+caption setting. We find modal dropout to not significantly alleviate bias towards the textual modality. We discuss our findings in detail in Appendix F.

5.4 Error analysis

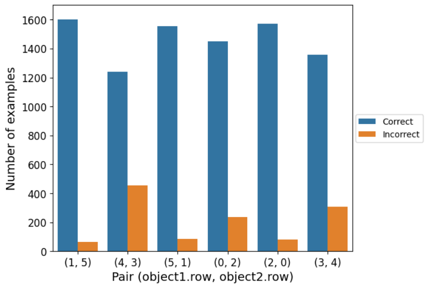
We now report findings from our error analysis on selected experiments where models converged but did not achieve perfect or near perfect F_1 scores, which can reveal trends that explain poor performance. We describe our analysis on the spatiality and cardinality tasks here, referring readers to Appendix G for details on quantifiers and numerical comparison.

Across all tasks, errors are intuitive and explainable, and the performance of models is generally poorer on boundary cases. For example, models are more error-prone when objects are positioned close to each other in the spatiality task, and where only one object either falsifies or confirms the query in the quantifier task. On the numerical comparison task, we observe more errors when the difference between the numerals being compared is small.

5.4.1 Spatiality



(a) Queries involving horizontal relationships, by the pairs of the columns of the queried objects.



(b) Queries involving vertical relationships, by the pairs of the rows of the queried objects.

Figure 2: Error analysis of VisualBERT trained on spatiality task, image-only setting, tested on OOD test set. (F_1=69.78; seed=0)

In the spatiality task, VisualBERT was unable to achieve F_1 scores above 70 across all three random seeds. Figure 2 shows that this was due to VisualBERT only correctly answering queries involving vertical relationships, but not horizontal ones. In the figure, the x-axis is organised by the pairs of object coordinates relevant to the query (i.e., columns for horizontal relationship queries, and rows for vertical relationship queries). A similar pattern is observed on all three runs on VisualBERT, and also on UNITER on one random seed. Mediocre results on the spatiality task can thus often be attributed to models successfully learning to answer only a subset of the queries.

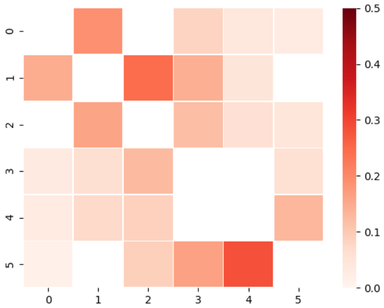


Figure 3: Percentage of incorrect answers of UNITER in the image-only setting, on InD $< x, y >$ pairs for vertical relationships. (F_1=71.79; seed=2)

Figure 3 shows the percentage of incorrect answers on specific pairs in the InD test set. Note that white entries are pairs which belong to the OOD test set. Overall, models make more mistakes on examples with queried objects positioned closer together (represented by cells closer to the diagonal).

5.4.2 Cardinality

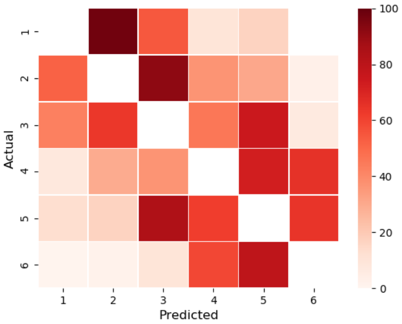


Figure 4: Actual vs predicted values (given that predicted answer is true). VisualBERT trained on the cardinality task, image-only setting, tested on InD test set. (F_1=77.84; seed=0)

While most models apart from LXMERT succeeded on the cardinality task in the caption-only and image+caption settings, moderate performance was observed in the image-only setting. We found that models performed poorer on images with more objects in the image, inclusive of distractors (refer to Appendix G for details). Furthermore, models often predict values close to the actual value. Figure 4 plots the actual number of objects in the scene against the number of objects predicted. Since the task was a true/false task, this analysis is based only on examples for which the model predicted that the query was true. The counting abilities of models thus intuitively resemble that of humans, given that they are confused by a large number of objects in the scene, and are also likely to predict answers close to the actual value when miscounting.

6 Discussion

Comparing modalities. As discussed in §4.3, V+L models exhibit a clear bias towards the textual modality across single and dual stream models, corroborating findings by Cao et al. (2020). This finding also applies to LXMERT, even though it generally performs more poorly on caption than image inputs. Furthermore, the V+L models perform more poorly than unimodal RoBERTa on various tasks in the caption-only setting, similar to Iki and Aizawa (2021), who show that V+L pretraining on degrades performance on NLU tasks.

Comparing tasks. The spatiality task is the hardest task, requiring at least 32k examples to converge in some cases, as opposed to the 8k examples for the other tasks (see Appendix C for details). Furthermore, convergence of the models on the spatiality task is highly sensitive to random seeds. Among the other tasks, the easiest task is the quantifiers task, followed by cardinality, and finally numerical comparison. In the caption-only and image+caption settings, all models apart from LXMERT achieve a close to perfect F_1 on the cardinality and quantifiers tasks. However, on the numerical comparison task, UNITER and LXMERT exhibited a limited ability to generalise outside the training distribution in both image and textual modalities. Thus, while success on the cardinality task indicates that models possess an understanding of the meaning of numbers in absolute terms, the numerical comparison task was able to differentiate models in terms of their understanding of numbers' relative positions on a numeral scale.

Comparing models. As argued in §5.1, the most significant factor differentiating models seems to be their **pretraining** rather than model architecture. Our findings corroborate Bugliarello et al. (2021)'s findings that differences between models cannot be primarily attributed to differences in model architecture. Unlike LXMERT and ALBEF, VisualBERT and UNITER both succeed on all tasks in all settings. While UNITER generally outperforms VisualBERT in the image-only setting, VisualBERT at times outperforms UNITER in the caption and image+caption settings. The superior performance of VisualBERT in caption-only settings could be due to its comparatively minimal pretraining on V+L objectives, enabling it to retain performance closer to that of text-only models.

The **encoding of image features** also has a significant impact on performance in the image-only setting. VisualBERT's poor performance on the spatiality task is likely due to the fact that it does not explicitly encode the spatial coordinates of the bounding boxes in its input. Additionally, it is likely that the pretrained object detector used by every model except ALBEF helped them outperform ALBEF in the image-only setting, despite underperforming it in common V+L benchmarks (Li et al., 2021). A possible explanation is that the information captured by the Faster R-CNN detector was more relevant to our visio-linguistic tasks.

Finally, the poor performance of LXMERT on caption-only and image+caption settings reflects its lack of **initialisation with BERT parameters** before pretraining. Given the superior performance of RoBERTa over BERT, it could be beneficial to initialise models with RoBERTa weights instead.

7 Conclusion

TRAVLR allows us to evaluate specific visio-linguistic reasoning skills in isolation, enabling finer-grained diagnosis of model deficiencies. We found some models to learn better from one modality than the other, and some task-setting combinations to be more challenging across the board. Furthermore, existing models may require unreasonably large amounts of data and training steps to learn simple tasks. Improving the sample efficiency and training time of V+L models is a potential direction for future research. Furthermore, while pretrained multilingual models have been shown to demonstrate zero-shot cross-lingual transfer abilities, it is unclear whether V+L models can similarly perform *cross-modal transfer* of downstream task abilities to a modality unseen during fine-tuning.

We hence contribute TRAVLR, which enables the evaluation of cross-modal transfer ability by encoding scenes bimodally. We found all models to suffer from a strong textual bias, and an inability to perform zero- and few-shot cross-modal transfer. Given the success of multilingual models with cross-lingual transfer, future work might attempt to bridge the gulf between the visual and textual modalities, perhaps by forcing models to transform images into intermediate representations which more closely resemble language. Visio-linguistic representations capable of such transfer will unlock a whole host of new applications, and we pose this as the next challenge for multimodal modelling.

Limitations

We acknowledge that the use of synthetic data is potentially unrealistic given that most applications would require reasoning on real-world input. Nevertheless, we note that because synthetic data is easier, it is still useful as a benchmark for the minimum expected performance of models. We follow the line of argument in prior work (e.g. Johnson et al. (2017)) that synthetic datasets not only mitigate the problem of distributional bias in real-world datasets, but also simplify the problem of object recognition to focus on diagnosing genuine abstract reasoning ability. We generally expect V+L models to acquire abstract visual reasoning skills, like humans, rather than being limited to the realm of photographs. TRAVLR's focus on simple shapes also makes it a test of this ability.

Our experiments on VOLTA should be taken as preliminary, as a full replication of all experiments was not performed due to resource limitations. Future work could investigate the extent to which the superior performance of VOLTA UNITER is robust across all tasks, which would indicate a genuine advantage due to UNITER's architecture. Furthermore, although we have attributed the superior performance of LXMERT to some aspect of its pretraining, we are unable to pinpoint whether the advantage is specifically due to the large size of its pretraining dataset, its use of VQA examples in the pretraining, or its use of a VQA pretraining objective, and leave this to future work.

Finally, while all models were shown to succeed at generalising to the OOD test setting with varying degrees of success, only the numerical comparison task poses a significant challenge in terms of OOD generalisation. This is expected, since we did not intend to challenge the models with generalisation settings which they are unlikely to encounter in test (i.e., more realistic datasets designed for fine-tuning models to perform real-world applications). Furthermore, the simplicity of our reasoning tasks enables us to evaluate models' abilities on specific semantic phenomena in isolation. Nevertheless, future work can extend TRAVLR to include more challenging generalisation tasks, such as generalisation of the meaning of negation over unseen quantifiers, compositional queries which combine multiple semantic phenomena (e.g. combining propositions via conjunction or disjunction), and queries which contain instructions to manipulate the scene.

Ethics Statement

TRAVLR is a synthetically generated dataset, and hence poses no ethical issues relating to the source of the data. Its use of abstract shapes has the further advantage of avoiding biases relating to human subjects.

References

Manoj Acharya, Kushal Kafle, and Christopher Kanan. 2019. TallyQA: Answering complex counting questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8076–8084.

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4971–4980.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Raffaella Bernardi and Sandro Pezzelle. 2021. Linguistic issues behind visual question answering. *Language and Linguistics Compass*, 15(6):e12417.

Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. 2021. Multimodal pretraining unmasked: A meta-analysis and a unified framework of vision-and-language BERTs. *Transactions of the Association for Computational Linguistics*, 9:978–994.

Jize Cao, Zhe Gan, Yu Cheng, Licheng Yu, Yen-Chun Chen, and Jingjing Liu. 2020. Behind the scene: Revealing the secrets of pre-trained vision-and-language models. In *European Conference on Computer Vision*, pages 565–580. Springer.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8440–8451.

Róbert Csordás, Kazuki Irie, and Juergen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 619–634.

Corentin Dancette, Remi Cadene, Damien Teney, and Matthieu Cord. 2021. Beyond question-based biases: Assessing multimodal shortcut learning in visual question answering. *arXiv preprint arXiv:2104.03149*.

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

Sarah Dolscheid, Christina Winter, and Martina Penke. 2015. Counting on quantifiers: Specific links between linguistic quantifiers and number acquisition. In *EAPCogSci*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021. Vision-and-language or vision-for-language? On cross-modal influence in multimodal transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9847–9857.

Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. 2018. Detectron. `https://github.com/facebookresearch/detectron`.

Karan Goel, Nazneen Fatema Rajani, Jesse Vig, Zachary Taschdjian, Mohit Bansal, and Christopher Ré. 2021. Robustness gym: Unifying the NLP evaluation landscape. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 42–55.

Paula Goolkasian. 1996. Picture-word differences in a sentence verification task. *Memory & Cognition*, 24(5):584–594.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6325–6334. IEEE Computer Society.

Drew A Hudson and Christopher D Manning. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709.

Taichi Iki and Akiko Aizawa. 2021. Effect of visual extensions on natural language understanding in vision-and-language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2189–2196.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1988–1997. IEEE.

Kushal Kafle, Robik Shrestha, and Christopher Kanan. 2019. Challenges and prospects in vision and language research. *Frontiers in Artificial Intelligence*, 2:28.

Corentin Kervadec, Grigory Antipov, Moez Baccouche, and Christian Wolf. 2021. Roses are red, violets are blue... but should VQA expect them to? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2776–2785.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Alexander Kuhnle and Ann Copestake. 2017. Shapeworld-a new test methodology for multimodal language understanding. *arXiv preprint arXiv:1704.04517*.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. 2019. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*.

Junnan Li, Ramprasaath R Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2020a. What does BERT with vision look at? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5265–5275.

Yikuan Li, Hanyin Wang, and Yuan Luo. 2020b. A comparison of pre-trained vision-and-language models for multimodal representation learning across medical images and reports. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1999–2004. IEEE.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Yi-Chung Lin and Keh-Yih Su. 2021. How fast can BERT learn simple natural language inference? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 626–633.

Guan-Horng Liu, Avinash Siravuru, Sai Prabhakar, Manuela Veloso, and George Kantor. 2017. Learning end-to-end multimodal sensor policies for autonomous navigation. In *Conference on Robot Learning*, pages 249–261. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 13–23.

Xiaopeng Lu, Lynnette Ng, Jared Fernandez, and Hao Zhu. 2021. CIGLI: Conditional image generation from language & image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3134–3138.

Roshanak Mirzaee, Hossein Rajaby Faghihi, Qiang Ning, and Parisa Kordjamshidi. 2021. Spartqa: A textual question answering benchmark for spatial reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4582–4598.

Tobias Norlund, Lovisa Hagström, and Richard Johansson. 2021. Transferring knowledge from vision to language: How to achieve it and how to measure it? In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–162.

Vicente Ordonez, Girish Kulkarni, and Tamara Berg. 2011. Im2text: Describing images using 1 million captioned photographs. *Advances in neural information processing systems*, 24.

Letitia Parcalabescu, Albert Gatt, Anette Frank, and Iacer Calixto. 2021. Seeing past words: Testing the cross-modal capabilities of pretrained V&L models on counting tasks. In *Proceedings of the 1st Workshop on Multimodal Semantic Representations (MMSR)*, pages 32–44.

Abhilasha Ravichander, Aakanksha Naik, Carolyn Rose, and Eduard Hovy. 2019. Equate: A benchmark evaluation framework for quantitative reasoning in natural language inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 349–361.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M Lake. 2020. A benchmark for systematic generalization in grounded language understanding. *Advances in Neural Information Processing Systems*, 33.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. VL-BERT: Pretraining of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Alane Suhr, Mike Lewis, James Yeh, and Yoav Artzi. 2017. A corpus of natural language for visual reasoning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 217–223.

Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. 2019. A corpus for reasoning about natural language grounded in photographs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6418–6428.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111.

Samson Tan, Shafiq Joty, Kathy Baxter, Araz Taeihagh, Gregory A. Bennett, and Min-Yen Kan. 2021. Reliability testing for natural language processing systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4153–4169, Online. Association for Computational Linguistics.

Jerry Tang, Amanda LeBel, and Alexander G Huth. 2021. Cortical representations of concrete and abstract concepts in language combine visual and linguistic representations. *bioRxiv*.

Alberto Testoni, Sandro Pezzelle, and Raffaella Bernardi. 2019. Quantifiers in a multimodal world: Hallucinating vision with language and sound. In *Proceedings of the workshop on cognitive modeling and computational linguistics*, pages 105–116.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? Probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315.

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. 2019. Visual entailment: A novel task for fine-grained image understanding. *arXiv preprint arXiv:1901.06706*.

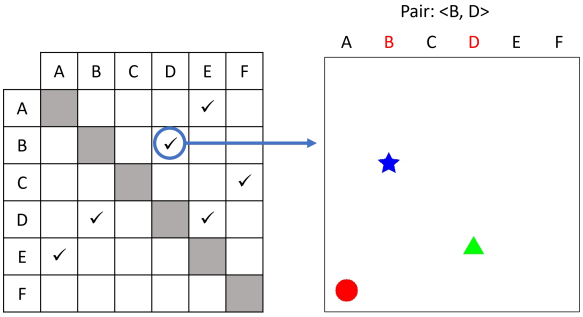
Amir Zadeh, Paul Pu Liang, and Louis-Philippe Morency. 2020. Foundations of multimodal colearning. *Information Fusion*, 64:188–193.

Mengjie Zhao, Yi Zhu, Ehsan Shareghi, Ivan Vulić, Roi Reichart, Anna Korhonen, and Hinrich Schütze. 2021. A closer look at few-shot crosslingual transfer: The choice of shots matters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5751–5767.

A Dataset Details

We now describe the version of TRAVLR used in our experiments, although our dataset generation scripts allow parameters such as the object properties and grid size to be modified. TRAVLR draws from five colours (red, blue, green, yellow, orange) and seven shapes (square, circle, triangle, star, hexagon, octagon, pentagon), yielding a total of thirty-five unique objects. The seven shapes and colours used in TRAVLR are present in the Visual Genome dataset (Krishna et al., 2017). We use a six by six grid, which allows a maximum of 36 objects in a scene. Each example consists of an image, a caption and a query.

A.1 Dataset Figures

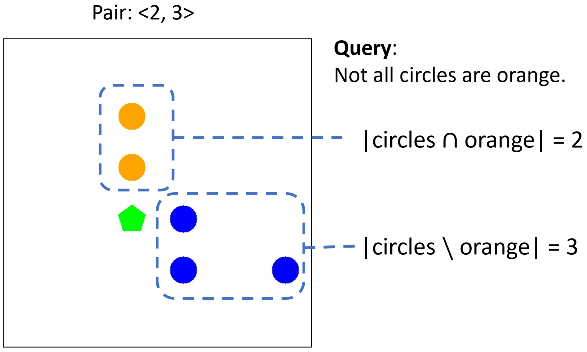


	A	B	C	D	E	F
A					✓	
B				✓		
C						✓
D		✓			✓	
E	✓					
F						

✓: pairs for testing

Query: The green triangle is to the right of the blue star.

Figure 5: An example of OOD test set construction. In a left/right relationship reasoning task, the relevant dimension is the column ID. Specific ID pairs (✓) are held out to form this test distribution.



Query:
Not all circles are orange.

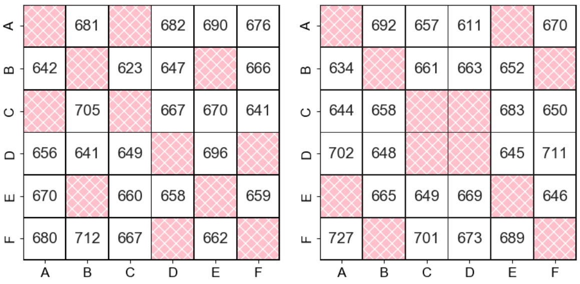
Figure 6: Example instance for *not all* quantifier with pair $< 2, 3 >$.

A.2 Dataset Statistics

Table 3 shows the label distributions (true/false) for the various datasets: training, validation, In-Distribution (InD) test, and Out-Of-Distribution (OOD) test sets. The spatiality dataset is the largest dataset comprising 32k examples, while the other datasets comprise 8k examples. In the remainder of the section, we detail crucial statistics for the splits between the training and OOD test sets.

Task	Train	Val.	InD Test	OOD Test
Spatial	15903 / 16097	4979 / 5021	5064 / 4936	9918 / 10082
Cardinality	4040 / 3960	4927 / 5073	5043 / 4957	10079 / 9921
Quantifier	4006 / 3994	5003 / 4997	5030 / 4970	10029 / 9971
Comparison	4088 / 3912	4926 / 5074	4992 / 5008	10033 / 9967

Table 3: Dataset statistics (no. of True / False).



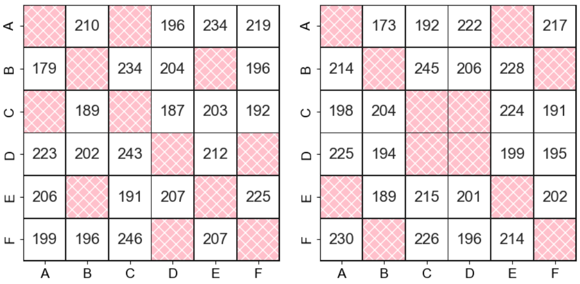
Horizontal Relationship

	A	B	C	D	E	F
A		681		682	690	676
B	642		623	647		666
C		705		667	670	641
D	656	641	649		696	
E	670		660	658		659
F	680	712	667		662	

Vertical Relationship

	A	B	C	D	E	F
A		692	657	611		670
B	634		661	663	652	
C	644	658			683	650
D	702	648			645	711
E		665	649	669		646
F	727		701	673	689	

(a) Training Set.



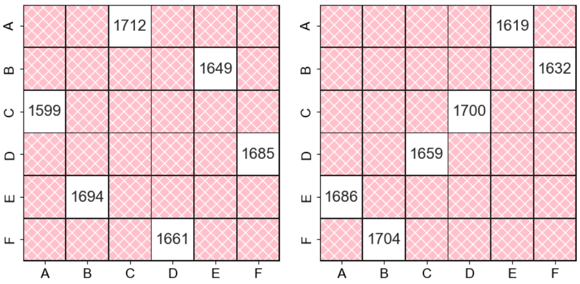
Horizontal Relationship

	A	B	C	D	E	F
A		210		196	234	219
B	179		234	204		196
C		189		187	203	192
D	223	202	243		212	
E	206		191	207		225
F	199	196	246		207	

Vertical Relationship

	A	B	C	D	E	F
A		173	192	222		217
B	214		245	206	228	
C	198	204			224	191
D	225	194			199	195
E		189	215	201		202
F	230		226	196	214	

(b) In-Distribution Test Set.



Horizontal Relationship

	A	B	C	D	E	F
A			1712			
B					1649	
C	1599					
D						1685
E		1694				
F				1661		

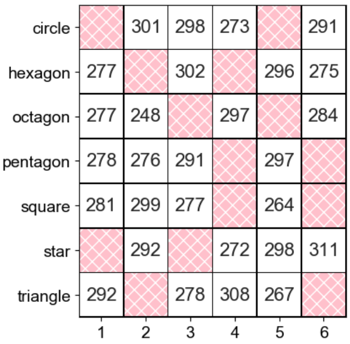
Vertical Relationship

	A	B	C	D	E	F
A					1619	
B						1632
C				1700		
D			1659			
E	1686					
F		1704				

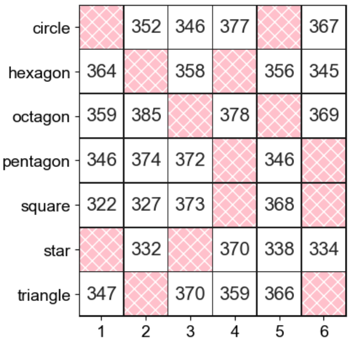
(c) Out-of-Distribution Test Set.

Figure 7: Spatiality (<`column(object1)`, `column(object2)`> for horizontal relationship, <`row(object1)`, `row(object2)`> for vertical relationship).

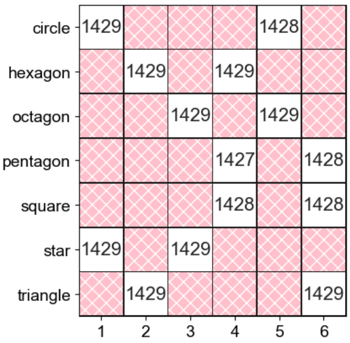
For the spatiality task, the scene comprises only three objects: the two objects mentioned in the query, and one distractor object. We limit the number of distractors for this task, since we intend the



(a) Training Set.



(b) In-Distribution Test Set.



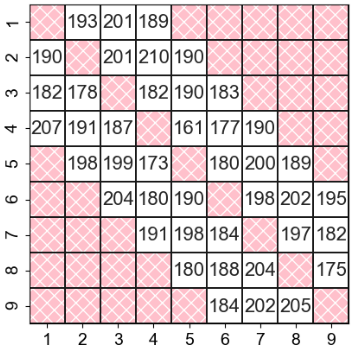
(c) Out-of-Distribution Test Set.

Figure 8: Cardinality (<`shape, number`>).

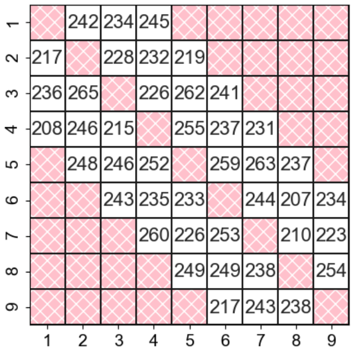
task to evaluate models' ability to compare the position of the objects, rather than their ability to perform object recognition.

Figure 7 shows dataset statistics for the training, InD and OOD spatiality dataset. Each figure indicates the number of examples whose scene instantiates a specific *pair* (this is independent of the query and label). For instance, there are 681 examples instantiating the pair $< A, B >$ for the horizontal relationship. In these examples, the first object in the query is found in Column A, and the second object is found in Column B. As illustrated in Figure 5, we ensure no overlap between pairs in the training set and the OOD test set. We note that each example either instantiates a horizontal or vertical relationship, but not both.

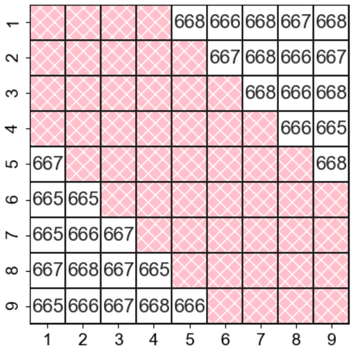
Unlike the spatiality task, more objects are present in the remaining tasks, since it is the ability to reason about the number of objects which is under test. For the cardinality task (Figure 8), we limit the number of objects relevant to the query within the range [1, 6], and the maximum number of distractor objects within the range [1, 10]. The relevant pairs shown in Figure 8 are <`shape, number`>, i.e. the number of objects of that shape.



(a) Training Set.



(b) In-Distribution Test Set.



(c) Out-of-Distribution Test Set.

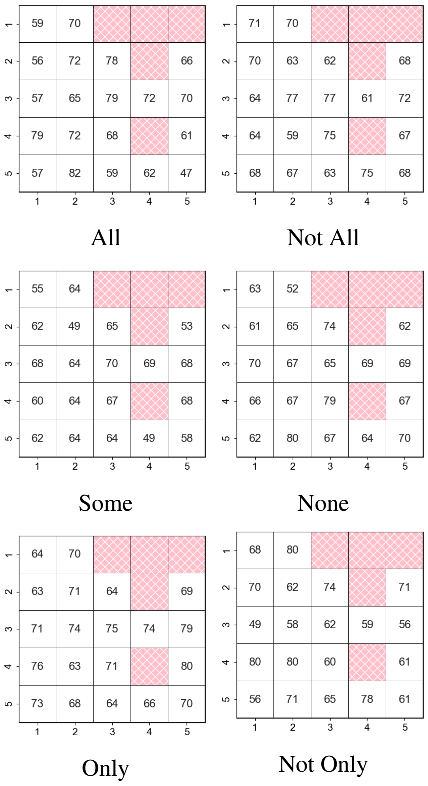
Figure 9: Numerical Comparison.

For the numerical comparison task (Figure 9), we split the train and test sets by the pair $< a, b >$ where a is the number of `[attr1]` objects, and b

is the number of `[attr2]` objects. The number of `[attr1]` and `[attr2]` objects are limited within the range $[1, 9]$, and the number of distractor objects is in the range $[1, 10]$.

All	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr2]} \setminus \texttt{[attr1]} >$
Not all	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr1]} \setminus \texttt{[attr2]} >$
No	$< \texttt{[attr1]} \setminus \texttt{[attr2]}, \texttt{[attr2]} \setminus \texttt{[attr1]} >$
Some	$< \texttt{[attr1]} \setminus \texttt{[attr2]}, \texttt{[attr1]} \cap \texttt{[attr2]} >$
Only	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr1]} \setminus \texttt{[attr2]} >$
Not only	$< \texttt{[attr1]} \cap \texttt{[attr2]}, \texttt{[attr2]} \setminus \texttt{[attr1]} >$

Table 4: Pairs for each quantifier.



All

Not All

Some

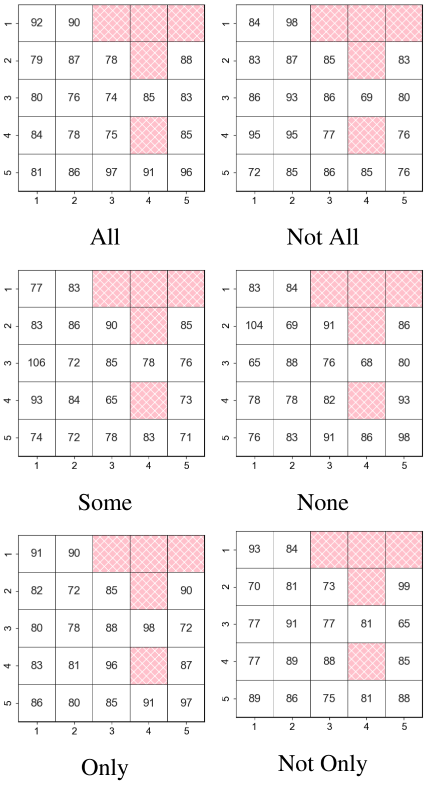
None

Only

Not Only

Figure 10: Quantifiers Training Set.

For the quantifiers task (Figure 11), we split the training and OOD distributions based on the pairs as stated in Table 4, which differ depending on the specific quantifier. We limit the number of objects relevant to each part of the pair within the range $[1, 5]$, and the number of distractor objects within the range $[2, 8]$.



All

Not All

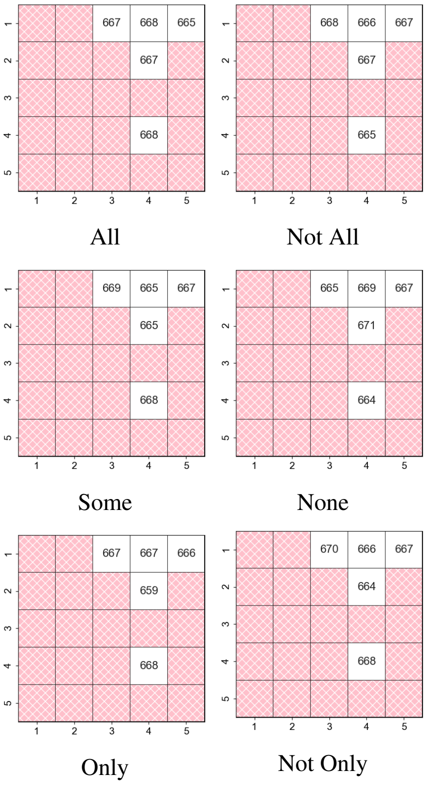
Some

None

Only

Not Only

(a) In-Distribution Test Set.



All

Not All

Some

None

Only

Not Only

(b) In-Distribution Test Set.

Figure 11: Quantifiers Test Sets.

B Experiment details

We conducted the following fine-tuning experiments:

- 4 V+L models on all 4 tasks on 3 modality settings, on 3 random seeds each = 144 main experiments.
- 2 text-only models on all 4 tasks, on 3 random seeds each = 24 experiments.
- Replications of the above experiments on 3 smaller subsets of the original dataset, on a single random seed = 144 + 24 = 168 experiments (see Appendix C).
- All models on all 4 tasks with frozen pretrained representations.
- 4 V+L models on 4 tasks on the modal dropout/"mixed" setting (see Appendix F).
- Selected experiments on the VOLTA implementation of 3 V+L models (see Appendix E).
- Selected experiments using the few-shot learning setting (see Appendix B.5).

B.1 Metric

The metric used is the macro F_1 score, which is computed as the arithmetic mean of the F_1 score of each of the two classes (true and false). We use F_1 instead of accuracy due to small imbalances in the classes (see Table 3).

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{1}$$

B.2 Hyperparameters

We train models for 80 epochs without early stopping. As the hyperparameters recommended for fine-tuning UNITER and LXMERT did not lead to convergence on some tasks, we adjusted learning rates downwards until the model converged. For LXMERT, we used a learning rate of 5e-5 only for the spatiality task, and 5e-6 elsewhere.

Model	**Batch Size**	**Learning rate**
VisualBERT	64	2e-5
UNITER	32	5e-6
LXMERT	32	5e-6 / 5e-5
ALBEF	256	2e-5
VOLTA VisualBERT	32	5e-6
VOLTA UNITER	32	5e-6
VOLTA LXMERT	32	5e-6

Table 5: Fine-tuning hyperparameters.

B.3 Runtimes

Table 6 provides an estimate of the runtimes for fine-tuning each model on different dataset sizes in different settings. Experiments were run either on NVIDIA GeForce RTX 2080 Ti or NVIDIA TITAN RTX GPUs. The longest experiments on ALBEF with the 32k dataset are estimated to lead to carbon emissions of 15.24 kgCO_2eq (Lacoste et al., 2019).

Dataset size	**Model**	**Image**	**Caption**	**Img.+Cap.**
32k	UNITER	15	27	27
	VisualBERT	15	27	27
	LXMERT	25.5	36.5	36.5
	ALBEF	38	38	38
	BERT	-	12	-
	RoBERTA	-	12	-
8k	UNITER	3.5	7	7
	VisualBERT	5	7	7
	LXMERT	6.5	10.5	10.5
	ALBEF	8	9	9
	BERT	-	3	-
	RoBERTA	-	3	-

Table 6: Approximate experiment run times in hours.

B.4 Model Details

Table 7 summarises the amount and sources of pretraining data used in the pretraining of various models we evaluate. Table 8 reports the total trainable parameters for each model.

Model	**Pretraining Dataset Size**	COCO	VG	CC	SBU	Other
VisualBERT	600k	✓			✓	✓
UNITER	5.6M	✓	✓	✓	✓	
LXMERT	9.18M	✓	✓	✓	✓	✓
ALBEF	4.0M	✓	✓	✓	✓	
VOLTA	2.77M			✓		

Table 7: Amount of pretraining data and sources of pretraining data. COCO: Microsoft COCO (Lin et al., 2014), VS: Visual Genome (Krishna et al., 2017), CC: Conceptial Captions (Sharma et al., 2018), SBU: SBU Captions (Ordonez et al., 2011)

Model	**Num. Parameters**
VisualBERT	112.64 M
UNITER	111.08 M
LXMERT	209.12 M
ALBEF	290.34 M
BERT	108.31 M
RoBERTa	124.65 M
VOLTA VisualBERT	114.02 M
VOLTA UNITER	113.63 M
VOLTA LXMERT	210.50 M

Table 8: Number of trainable parameters.

B.5 Few-shot learning experiments

On top the zero-shot setting experiments that we conduct on all fine-tuned models, we perform few-shot learning experiments on selected fine-tuned models as shown in Table 9. We select the best-performing model for the task and setting, as well as additional models to ensure that all models are tested. We adopt the same hyperparameters stated in Table 5, and perform additional fine-tuning on 200 examples of the unseen modality for 20 epochs. Typically, between 10 to 100 examples are sufficient for few-shot cross-lingual transfer on multilingual BERT (Zhao et al., 2021). For instance, in the "Image to Caption" setting, we fine-tune a model already fine-tuned for 80 epochs in the image-only setting, on examples in the caption-only setting.

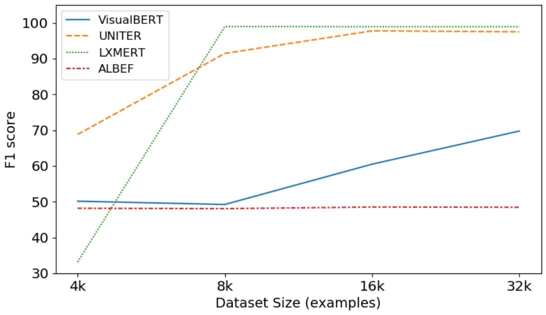
Setting	Task	Models tested	Seed
Image to Caption	Spatiality	UNITER	0
		LXMERT	0
	Quantifiers	VisualBERT	2
		UNITER	2
		LXMERT	2
Caption to Image	Spatiality	UNITER	0
		ALBEF	2
	Quantifiers	VisualBERT	2
		UNITER	2
		ALBEF	2

Table 9: Models tested on few-shot learning setting. (Seed: the random seed used in the initial fine-tuning of the model).

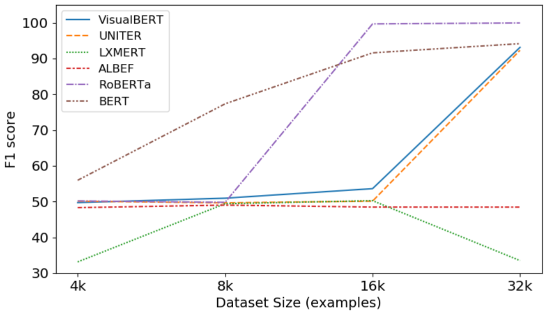
C Effect of Training Dataset Size

To investigate the effect of training dataset size on model performance, we fine-tune models with 50%, 25% and 12.5% subsets, randomly sampled from the full training set. These experiments were conducted only on a single random seed, due to resource limitations.

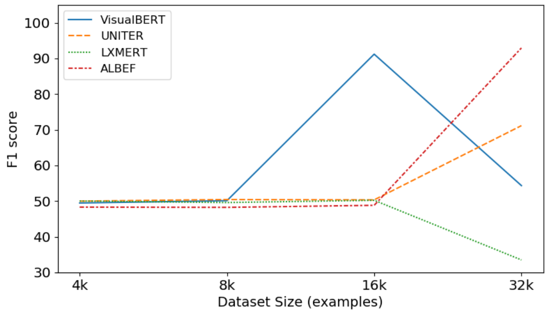
Spatiality. On the spatiality task (Figure 12), in the image-only setting, LXMERT requires the least amount of data to achieve convergence (4k examples), followed by UNITER (8k examples), and VisualBERT (16k examples). ALBEF fails to converge, even on the full 32k dataset. There is a clear advantage in performance of LXMERT and UNITER over VisualBERT and ALBEF.



(a) Image-only setting.



(b) Caption-only setting.

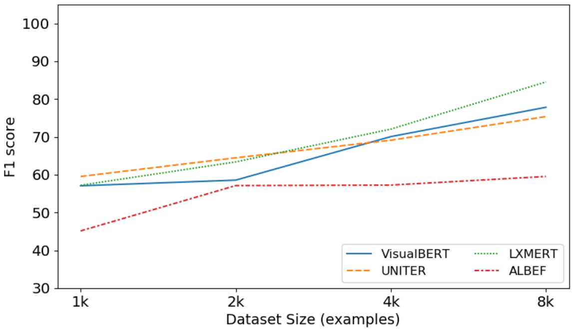


(c) Image+caption setting.

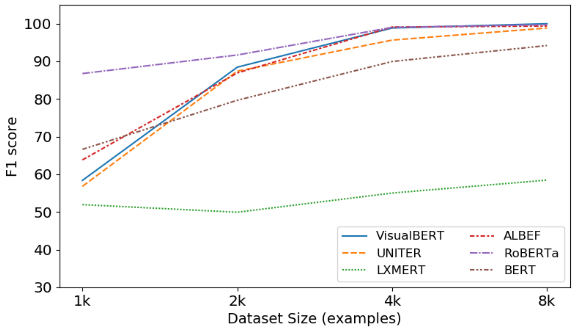
Figure 12: Performance for the spatiality task on the OOD test set in (a) image-only, (b) caption-only and (c) image+caption settings; models trained on increasing subsets with one random seed.

In the caption-only setting, the text-only model RoBERTa performs the best, achieving F_1s close to 100 with 16k training examples. While BERT, UNITER and VisualBERT achieve similar results with 32k training examples, the V+L models are outperformed by BERT when trained on a smaller dataset.

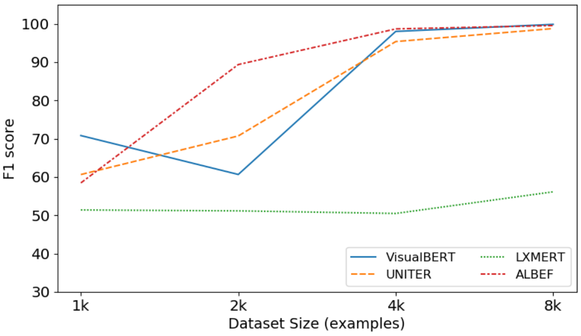
Cardinality. Compared to the spatiality task, more models converged on the cardinality task (Figure 13) with a smaller amount of data (8k examples). In the image-only setting, LXMERT remains the best performing model on the full 8k dataset, but the performance of VisualBERT, UNITER and LXMERT are relatively similar. In the caption-only setting, when trained on a smaller dataset with 1k examples, RoBERTA achieves an F_1 of 86.76, while all other models achieve an F_1 between 55 and 67, except for LXMERT, which fails to achieve



(a) Image-only setting.

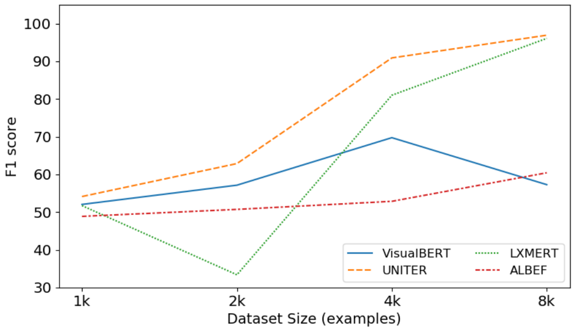


(b) Caption-only setting.

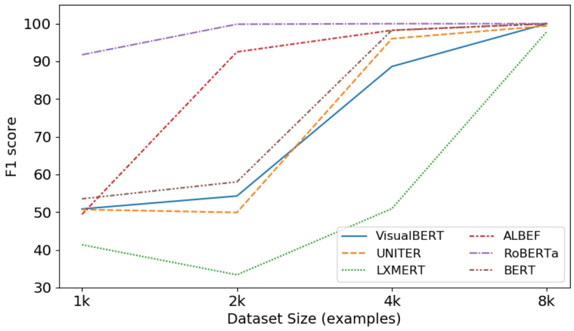


(c) Image+caption setting.

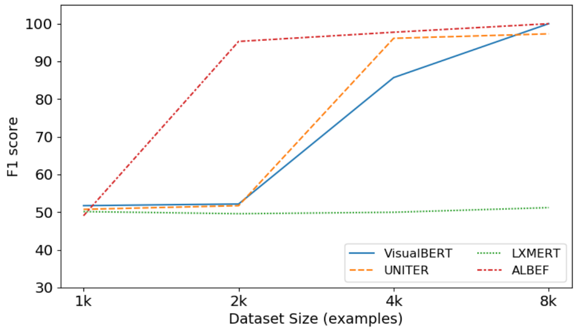
Figure 13: Performance for the cardinality task on the OOD test set in (a) image-only, (b) caption-only and (c) image+caption settings; models trained on increasing subsets with one random seed.



(a) Image-only setting.



(b) Caption-only setting.



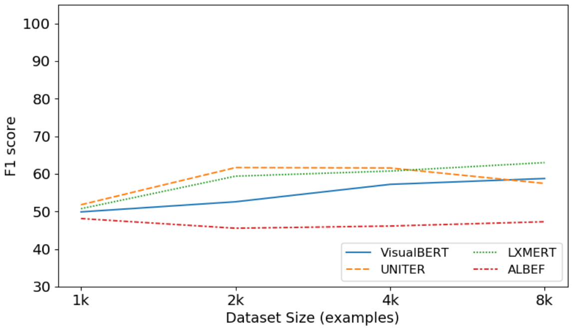
(c) Image+caption setting.

Figure 14: Performance for the quantifiers task on the OOD test set in (a) image-only, (b) caption-only and (c) image+caption settings; models trained on increasing subsets with one random seed.

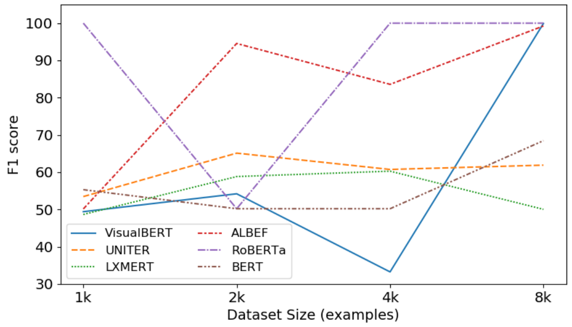
non-random performance. In the image+caption setting, ALBEF seems to be most data efficient, performing the best on a 2k training dataset.

Quantifiers. On the quantifiers task (Figure 14), in the image-only setting, we observe a similar overall trend to the spatiality task, where UNITER and LXMERT significantly outperform the other models. In the caption-only setting, RoBERTa significantly outperforms BERT with limited data (1k examples), and the performance of UNITER and VisualBERT resembles that of BERT. The superior performance of ALBEF on a smaller 2k dataset in both caption-only and image+caption settings is observed again, as on the cardinality task.

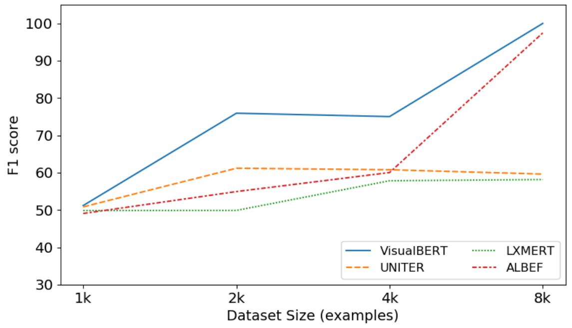
Numerical Comparison. On the numerical comparison task (Figure 15), all models exhibit relatively poor performance on the OOD dataset in the image-only setting, revealing an inability to generalise to unseen number pairs. In the caption-only and image+caption settings, the superior performance of RoBERTa over BERT, and VisualBERT and ALBEF over the other V+L models can be observed, as similarly described for other tasks. The performance of VisualBERT and UNITER is clearly distinguished.



(a) Image-only setting.



(b) Caption-only setting.



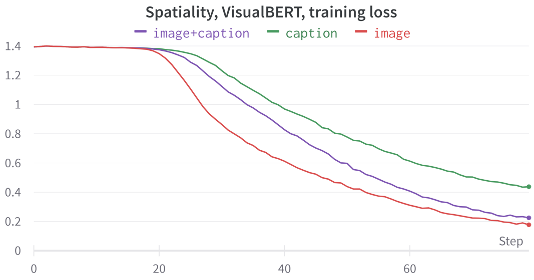
(c) Image+caption setting.

Figure 15: Performance for the numerical comparison task on the OOD test set in (a) image-only, (b) caption-only and (c) image+caption settings; models trained on increasing subsets with one random seed.

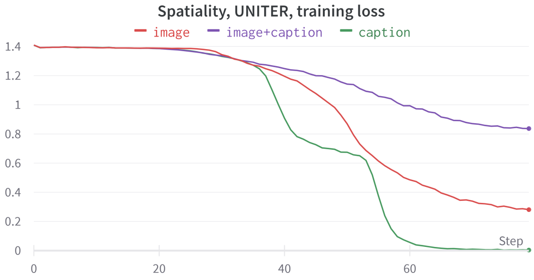
D Training Duration

We discuss selected runs indicative of the contrast in the number of training steps required by models. From the training loss curves on a single run of training on the spatiality tasks, we can observe a substantial degree of variation in the amount of time each model takes to converge.

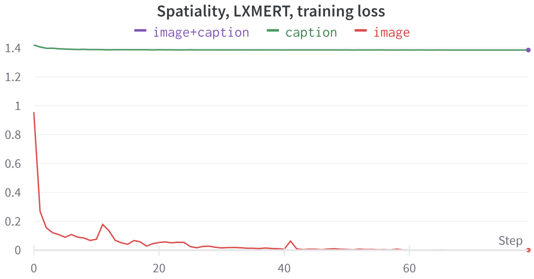
The training loss curves in Figure 16 are on the whole indicative of the significant amount of time required by all models except LXMERT to learn the spatiality task. With the notable exception of LXMERT, many models have yet to fully converge by 60 epochs. LXMERT takes fewer than 5 epochs to achieve a training loss close to 0, while completely failing to converge on the caption and image+caption settings. Possible reasons for the faster convergence of LXMERT on the spatiality task include the large size of its pretraining data, and the fact that it was additionally pretrained on a VQA task, unlike the other models. VisualBERT has yet to completely converge after 80 epochs. ALBEF in the image+caption setting and UNITER in the



(a) VisualBERT (seed=1)



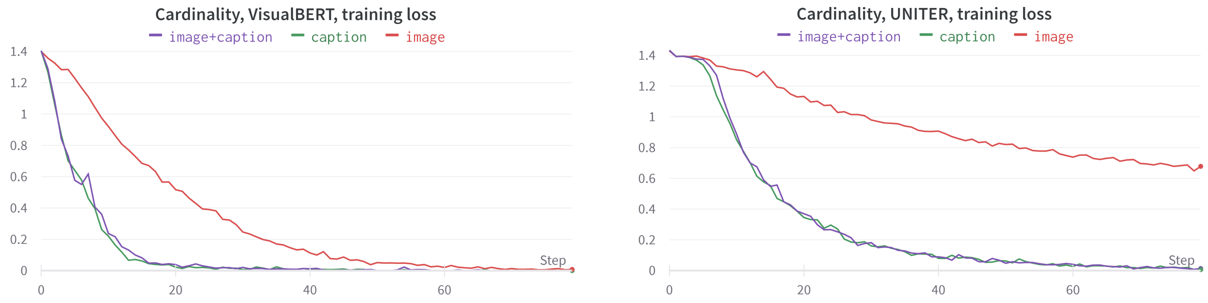
(b) UNITER (seed=0)



(c) LXMERT (seed=1)

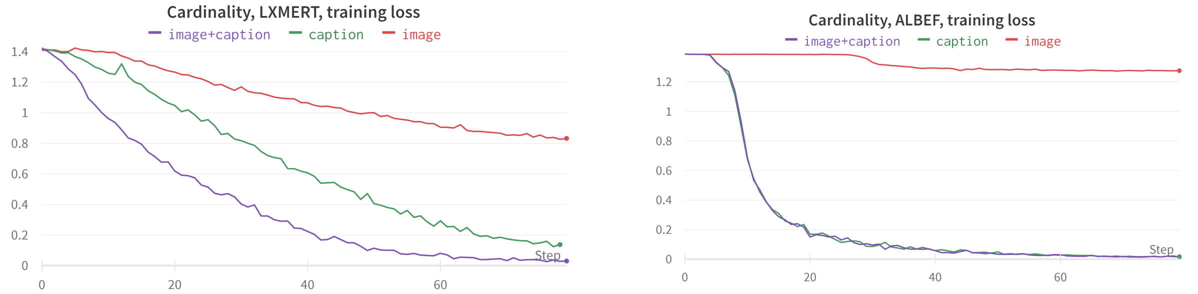
(d) ALBEF (seed=0)

Figure 16: Training loss curves on spatiality task.



(a) VisualBERT (seed=1)

(b) UNITER (seed=0)



(c) LXMERT (seed=1)

(d) ALBEF (seed=0)

Figure 17: Training loss curves on cardinality task.

caption setting are representative of successful convergence, which is achieved only after 40 and 60 epochs, respectively.

The variation between models is much less significant on the cardinality dataset, as seen in Figure 17. The training loss curves between the caption-only and image+caption settings are often similar, and also steeper compared to the image-only loss curves, which is reflective of the textual bias of all models towards the caption when both modalities are presented in the image+caption setting. In the image-only setting, most models have yet to fully converge within 80 epochs, in this case, with the exception of VisualBERT. It is notable that the marked advantage of LXMERT in terms of efficiency of training time observed in the spatiality task does not extend to other tasks.

E VOLTA Experiments

From Table 10, we observe that while the performance of all three VOLTA models are similar on the 32k spatiality dataset, VOLTA UNITER significantly outperforms both VOLTA LXMERT and VOLTA VisualBERT on the 8k spatiality dataset. This is in contrast to results on the original models where LXMERT exhibits a clear advantage over the other models and is able to achieve an F_1 score close to 100 on the 8k dataset.

Dataset Size	Model	OOD		InD		# Seeds
		Mean	Stdev.	Mean	Stdev.	
32k	VOLTA LXMERT	99.30	-	99.67	-	1
	VOLTA UNITER	99.67	-	99.88	-	1
	VOLTA VisualBERT	98.31	-	98.92	-	1
	LXMERT	99.46	0.47	99.84	0.13	3
	UNITER	89.67	14.43	88.71	14.58	3
	VisualBERT	65.94	6.63	66.96	6.05	3
8k	VOLTA LXMERT	62.28	3.16	64.17	3.73	3
	VOLTA UNITER	94.46	1.19	93.18	1.50	3
	VOLTA VisualBERT	60.73	2.39	60.69	2.90	3
	LXMERT	99.02	-	99.00	-	1
	UNITER	88.33	-	91.48	-	1
	VisualBERT	49.60	-	49.25	-	1

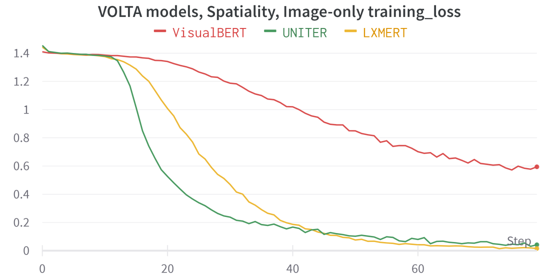
Table 10: Mean F_1 scores of VOLTA and original models trained on spatiality dataset in the image-only setting. Standard deviation is shown where applicable.

Similar findings are observed on the cardinality dataset (Table 11), where VOLTA LXMERT is outperformed by both VOLTA UNITER and VOLTA VisualBERT. The VOLTA LXMERT differs from the original LXMERT in the amount of pretraining data, as well as the use of VQA examples and a downstream VQA objective during pretraining. It can be concluded that LXMERT loses its advantage in the image-only setting when it is pretrained in the same manner as other models.

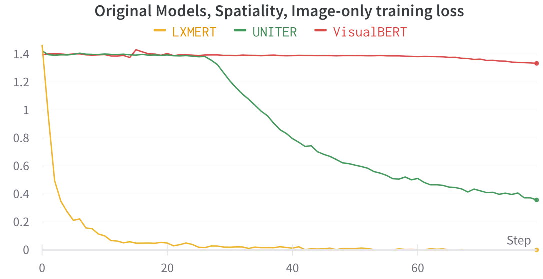
Model	**OOD**		**InD**		**# Seeds**
	Mean	**Stdev.**	**Mean**	**Stdev.**	
VOLTA LXMERT	73.38	0.56	79.30	0.39	3
VOLTA UNITER	85.91	5.13	87.40	3.36	3
VOLTA VisualBERT	81.67	4.46	83.90	1.66	3
LXMERT	82.90	3.86	84.56	2.71	3
UNITER	77.12	5.04	77.03	1.44	3
VisualBERT	77.41	0.61	76.30	0.46	3

Table 11: F_1 scores of VOLTA and original models trained on cardinality dataset in the image-only setting. Standard deviation is shown where applicable.

The difference between VOLTA LXMERT and the original LXMERT can also be clearly observed in the training loss curves in Figure 18, where the original LXMERT is significantly more efficient in terms of number of epochs required for training.



(a) VOLTA models



(b) Original models

Figure 18: Training loss curves on 8k spatiality dataset in the image-only setting.

Furthermore, it is notable that the performance of VisualBERT on the full dataset is significantly improved over the original model. This is likely due to the fact that the VOLTA VisualBERT was pretrained with more data than the original VisualBERT. An additional finding is that VOLTA UNITER outperforms VOLTA VisualBERT and LXMERT.

Finally, the VOLTA models were also used to investigate the comparatively poor performance of the original LXMERT on the caption-only and image+caption settings. As seen in Table 12, all three VOLTA models achieve similar results on the 8k cardinality dataset. Given that all VOLTA models were initialised with BERT weights, this lends support to the idea that the poor performance of LXMERT on textual input is due to the lack of initialisation with BERT weights (Tan and Bansal, 2019).

Model	**OOD**		**InD**		**# Seeds**
	Mean	**Stdev.**	**Mean**	**Stdev.**	
VOLTA LXMERT	97.87	0.85	99.44	0.03	3
VOLTA UNITER	98.73	1.25	99.60	0.11	3
VOLTA VisualBERT	99.40	0.03	98.50	0.11	3
LXMERT	60.02	4.76	78.90	6.83	3
UNITER	98.96	0.30	97.14	0.83	3
VisualBERT	99.99	0.01	99.91	0.06	3

Table 12: Mean F_1 scores of VOLTA and original models trained on the 8k cardinality dataset in the caption-only setting. Standard deviation is shown where applicable.

Task	Model	OOD			InD			#sd
		Image	Caption	Img. + Cap.	Image	Caption	Img. + Cap.	
S	VisualBERT	50.61 (+1.28)	70.03 (-0.96)	69.26 (-2.12)	50.88 (+1.85)	72.26 (-0.32)	71.94 (-1.10)	3
	UNITER	64.24 (+14.09)	50.03 (-20.72)	49.70 (-21.47)	63.53 (+13.55)	49.70 (-19.81)	50.32 (-20.45)	1
	LXMERT	33.27 (-0.24)	33.27 (-0.24)	33.27 (-0.24)	33.43 (+0.38)	33.43 (+0.38)	33.43 (+0.38)	0
	ALBEF	48.25 (-0.31)	96.12 (+2.46)	96.10 (+2.79)	48.14 (+0.52)	99.99 (-0.01)	99.99 (-0.01)	3
C	VisualBERT	65.07 (+19.87)	99.86 (-0.08)	99.85 (-0.10)	65.01 (+18.39)	99.53 (-0.31)	99.41 (-0.35)	3
	UNITER	66.01 (+23.65)	97.86 (-0.62)	97.40 (-1.44)	67.61 (+23.28)	94.57 (-3.80)	94.01 (-2.83)	3
	LXMERT	60.43 (+9.71)	54.03 (-0.97)	54.35 (-0.91)	64.42 (+14.04)	76.80 (+20.20)	73.63 (+10.55)	3
	ALBEF	47.67 (+4.51)	99.21 (-0.41)	99.16 (-0.45)	48.77 (+3.28)	98.30 (-1.06)	98.21 (-1.14)	3
Q	VisualBERT	66.34 (+16.83)	99.01 (-0.97)	99.01 (-0.96)	65.35 (+15.68)	98.43 (-1.56)	98.37 (-1.63)	3
	UNITER	75.04 (+26.98)	93.41 (-4.44)	93.82 (-5.04)	73.18 (+23.97)	88.14 (-7.60)	88.43 (-9.00)	3
	LXMERT	64.72 (+15.78)	53.23 (+19.30)	50.13 (-0.90)	62.36 (+13.27)	69.16 (+35.22)	67.06 (+16.21)	3
	ALBEF	51.92 (+4.19)	99.88 (-0.09)	99.87 (-0.09)	51.68 (+4.14)	99.18 (-0.62)	99.12 (-0.68)	3
N	VisualBERT	59.32 (+8.22)	92.68 (-7.10)	92.79 (-7.00)	61.54 (+10.64)	99.62 (-0.27)	99.64 (-0.26)	3
	UNITER	61.79 (+15.01)	56.92 (-7.88)	56.85 (-7.03)	76.65 (+28.84)	99.17 (-0.34)	99.08 (-0.71)	3
	LXMERT	58.81 (+8.13)	56.07 (-1.39)	56.57 (-0.24)	62.17 (+13.80)	84.17 (-0.05)	79.42 (-19.24)	3
	ALBEF	42.80 (-4.44)	88.82 (-10.11)	88.83 (-10.10)	47.34 (-1.40)	99.62 (-0.22)	99.62 (-0.22)	3

Table 13: Mean F_1 scores of models trained in the mixed setting for the four V+L reasoning tasks (**S:** Spatiality, **C:** Cardinality, **Q:** Quantifiers, **N:** Numerical comparison). In parentheses, relative change from performance of models trained on image+caption setting are indicated. Differences larger than +5 are highlighted in green and differences smaller than -5 are highlighted in orange.

F Modal dropout/"mixed" setting

There is a concern that the bimodal presentation of the input in the image+caption setting facilitates overfitting to the textual modality. One way to alleviate the impact of bias towards either modality is by performing dropout of either the image or caption input in a subset of the input, thus introducing some unimodal examples into the image+caption dataset. This technique is inspired by the idea of sensor dropout as described in Liu et al. (2017), where overfitting to specific sensors (modalities) by a multisensor autonomous navigation system is avoided through dropout of features from sensors during training. In the following experiments, we perform dropout at a fixed probability of 25% for both the image and caption modalities. In other words, datasets comprise 50% image+caption, 25% image-only and 25% caption-only input.

Overall, performance of models trained in the mixed setting resembles performance of models trained in the image+caption setting, in that they exhibit a bias towards the textual modality. Performance on the image modality is generally poorer than performance on both caption-only and image+caption test settings (with the exception of LXMERT), and performance on caption-only and image+caption settings are largely similar. Nevertheless, because of the dropout of the caption modality in 25% of the input, the models are forced to also learn to deal with image-only input to some extent, and hence exhibit non-random performance on image-only test data in some cases.

F.1 Comparison against models trained purely on image+caption input

First, we compare the performance of models trained in the mixed setting against those trained in the image+caption setting. The relative difference is shown in brackets in Table 13. In the spatiality task, only UNITER achieves a non-random performance in the image-only setting, while VisualBERT, LXMERT and ALBEF exhibit no significant difference in comparison to the image+caption baseline. On the other tasks, we observe that VisualBERT, UNITER and LXMERT exhibit above random performance in the mixed setting, improving results over the purely image+caption setting.

When models trained in the mixed setting are tested in the caption and image+caption settings, in the majority of cases, there is a small decrease in performance compared to models trained in the purely image+caption setting. The small magnitude of this difference is likely because the models continue to overfit to the caption when both image and caption input is presented, and are deprived of

Task	**Model**	**OOD**			**InD**			**#Seeds**
		Image	**Caption**	**Img. + Cap.**	**Image**	**Caption**	**Img. + Cap.**	
S	VisualBERT	+1.36	+19.07	-21.95	+1.28	+21.66	-28.05	3
	UNITER	-27.24	+0.37	-0.66	-24.80	-1.38	+0.42	1
	LXMERT	-65.73	-16.15	-16.97	-65.59	-16.47	-17.65	0
	ALBEF	+0.16	+47.09	+47.27	-0.56	+50.87	+50.89	3
C	VisualBERT	+6.49	+11.41	+1.82	+4.86	+12.50	+4.48	3
	UNITER	+1.52	+10.46	+2.02	+3.27	+11.35	+4.31	3
	LXMERT	-2.97	+4.08	+3.85	-0.11	+19.47	+24.46	3
	ALBEF	-9.47	+12.29	+0.47	-8.50	+9.64	+1.83	3
Q	VisualBERT	+9.18	+44.72	+13.31	+8.70	+45.03	+15.75	3
	UNITER	+12.15	+43.48	-2.30	+8.44	+36.99	-2.73	3
	LXMERT	+31.32	+19.83	+0.15	+28.89	+35.69	+17.93	3
	ALBEF	+1.18	+7.37	+2.14	+0.81	+13.61	+2.92	3
N	VisualBERT	+6.73	+38.46	+17.75	+0.09	+43.60	+1.18	3
	UNITER	+0.10	-8.21	-3.95	+4.69	+5.98	+0.26	3
	LXMERT	-0.59	-2.77	-1.28	+1.54	+18.20	+11.76	3
	ALBEF	-2.75	-5.73	+28.78	-0.61	+5.29	+21.15	3

Table 14: Difference in F_1 scores between models trained on the mixed setting and models trained on subset baseline for the four V+L reasoning tasks (**S:** Spatiality, **C:** Cardinality, **Q:** Quantifiers, **N:** Numerical comparison). Differences larger than +5 are highlighted in green and differences smaller than -5 are highlighted in orange.

the textual information in only 25% of the training data. The magnitude of the difference is larger in the numerical comparison task, where VisualBERT, UNITER and ALBEF exhibit significant decreases in performance on the OOD test set.

There are several exceptions to the overall trend. Firstly, unlike the other models, UNITER seems to have overfitted to the image modality instead of the caption modality in the spatiality task. Second, LXMERT exhibits an improvement in performance in both the caption and image+caption settings on the cardinality and quantifier datasets.

F.2 Comparison against models trained on subset of data

Next, we compare the results from the mixed setting against a baseline if the mixed dataset were separated into datasets with fully unimodal or bimodal examples. For instance, the 32k mixed dataset on the spatiality task can be compared against baseline datasets of i) 16k bimodal image+caption examples, ii) 8k image-only examples and iii) 8k caption-only examples. Specifically, we ask if there is a performance benefit to training models on a single mixed dataset which combine these subsets together, as opposed to training models on each dataset separately.

This baseline is obtained from training on a randomly sampled subset of the full dataset. As previously explained, these experiments on subsets of the full dataset were conducted only on a single random seed, due to resource limitations. In Table 14, we observe that the performance of models trained on the mixed dataset indeed outperforms models trained on only the caption-only and image+caption subset of the data. This result is not unexpected, given that it is known that models exploit the caption present in the image+caption examples.

However, it is less clear from the results whether the models are able to exploit the additional image information present in the image+caption examples in the mixed setting to improve performance on the image-only test examples. In particular, results from the quantifier task, as well as the results of VisualBERT on the OOD test set seem suggestive of some advantage accorded to models by the additional image+caption data. Nevertheless, there are also results suggestive of the opposite finding, particularly on the spatiality task, where the presence of caption-only and image+caption examples degrades performance in the image-only setting.

We conclude that the technique of dropping out the input from some modalities helps to ameliorate the extent of the textual bias in the image+caption setting to some extent, although the overall textual bias of models remains.

G Error analysis

In section 5.4, we described our error analysis of the spatiality and cardinality tasks. In this appendix section, we include additional details on the cardinality task and present our error analysis on the quantifiers and numerical comparison tasks.

G.1 Cardinality

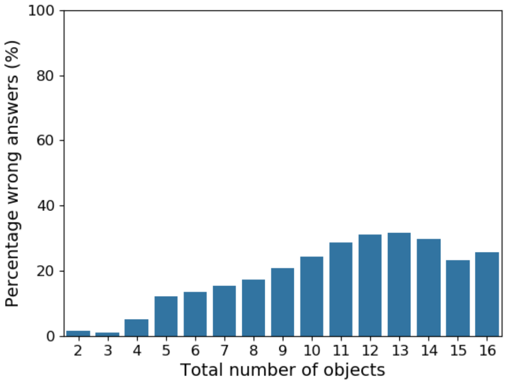
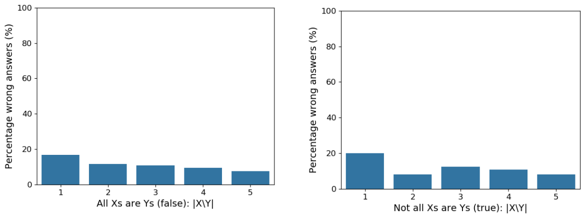


Figure 19: Percentage of incorrect answers by total number of objects. VisualBERT trained on cardinality task, image-only setting, tested on OOD test set. (F_1=77.84; seed=0)

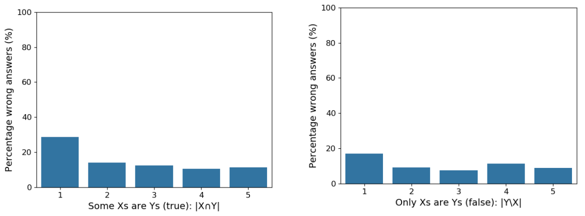
As noted in section 5.4, a significant factor is the total number of objects in the image, inclusive of distractors. Figure 19 shows an increasing percentage of wrong answers as the total number of objects increases. We also notice a slight decrease in wrong answers when the number of objects exceeds 13. This can be attributed to the decreasing likelihood of correct answers having smaller numbers.

G.2 Quantifiers

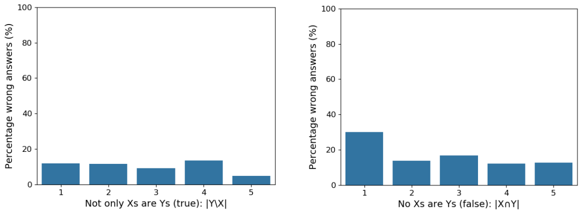
Figure 20 plots the percentage of wrong answers against specific values relevant to the particular quantifiers of VisualBERT on the InD test set in the image-only setting. Similar patterns are also observed across other models in runs with moderate performance, although moderate performance in some cases may be due to failure to learn a subset of quantifiers. For instance, Figure 20a shows the percentage of wrong answers against the number of Xs which are not Ys, given all *false* queries of the form “All Xs are Ys”. The results show that the more Xs which are not Ys, the less likely it is for the model to incorrectly predict the query to be true. In Figure 20c, given all *true* queries of the form “Some Xs are Ys”, the more objects which are both X and Y, the less likely it is for the model to incorrectly predict the query to be false.



(a) All Xs are Ys (false): $|X \setminus Y|$ (b) Not all Xs are Ys (true): $|X \setminus Y|$



(c) Some Xs are Ys (true): $|X \cap Y|$ (d) Only Xs are Ys (false): $|Y \setminus X|$



(e) Not only Xs are Ys (true): $|Y \setminus X|$ (f) No Xs are Ys (false): $|X \cap Y|$

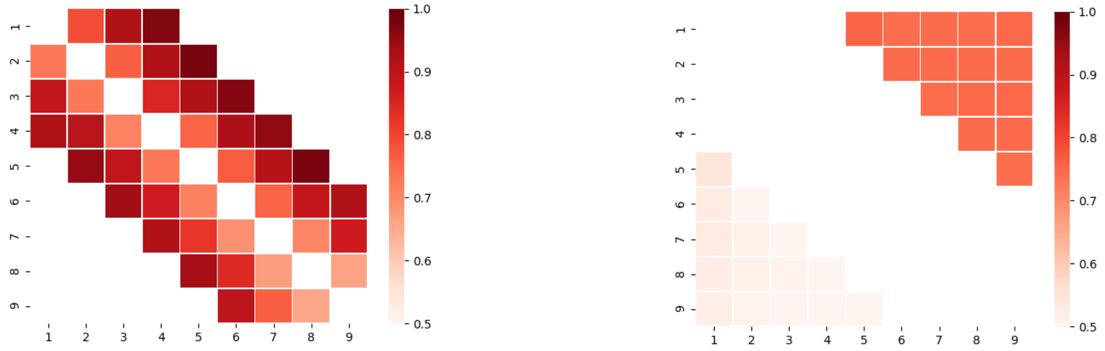
Figure 20: Percentage of wrong answers on quantifier-specific values. VisualBERT, image-only setting on InD test set. (F_1=87.98; seed=0)

Although the trend is not always robust across all the quantifiers, this overall pattern of errors made by models is not unexpected.

G.3 Numerical comparison

The plot in Figure 21 reveals a clear trend where models are more accurate on pairs where the magnitude of $|x - y|$ (i.e. the difference between the numerals being compared) is larger for the InD test set. However, the more significant finding is that the numerical comparison task was the only task which exhibited significant differences in performance between the InD and OOD datasets, due to an inability to generalise to unseen pairs particularly in the image-only setting, but also the caption-only setting in the case of UNITER and LXMERT.

In Figure 21, we observe a pattern where the model succeeds only on OOD queries where the number of the first object exceeds that of the second object (cells on the upper right are darker than cells on the lower left). This asymmetry is also observed to some extent in Figure 22. This pat-



(a) LXMERT, image-only InD. (F_1=84.45; seed=0)

(b) LXMERT, image-only OOD. (F_1=63.01; seed=0)

Figure 21: Results of LXMERT, image-only setting on specific $< x, y >$ pairs.

(a) UNITER, caption-only InD. (F_1=99.66; seed=0)

(b) UNITER, caption-only OOD. (F_1=61.90; seed=0)

Figure 22: Results of UNITER, caption-only setting on specific $< x, y >$ pairs.

tern does not result from any simple strategy of providing the same answer (true/false) given a relation in the query (*more*/*fewer*). A similar pattern is observed even when subsetting the OOD test set into queries with either the *more* or *fewer* relation in both cases. The overall finding is that models are able to generalise to unseen number pairs by constructing an implicit numeral scale, but only to a limited extent.

Paraphrase Acquisition from Image Captions

Marcel Gohsen
Bauhaus-Universität Weimar

Matthias Hagen
Friedrich-Schiller-Universität Jena

Martin Potthast
Leipzig University and ScaDS.AI

Benno Stein
Bauhaus-Universität Weimar

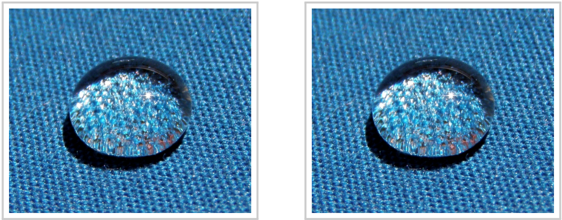
Abstract

We propose to use image captions from the Web as a previously underutilized resource for paraphrases (i.e., texts with the same “message”) and to create and analyze a corresponding dataset. When an image is reused on the Web, an original caption is often assigned. We hypothesize that different captions for the same image naturally form a set of mutual paraphrases. To demonstrate the suitability of this idea, we analyze captions in the English Wikipedia, where editors frequently relabel the same image for different articles. The paper introduces the underlying mining technology, the resulting Wikipedia-IPC dataset, and compares known paraphrase corpora with respect to their *syntactic and semantic paraphrase similarity* to our new resource. In this context, we introduce characteristic maps along the two similarity dimensions to identify the style of paraphrases coming from different sources. An annotation study demonstrates the high reliability of the algorithmically determined characteristic maps.

1 Introduction

Two texts that convey “semantically equivalent information” via a different wording are called *paraphrases*, or said to *be in a paraphrase relation*. A variety of natural language processing tasks have been approached by using paraphrase resources or paraphrase generation and detection algorithms: textual entailment (Dagan et al., 2013; Marelli et al., 2014; Izadinia et al., 2015), semantic similarity (Agirre et al., 2015; Li and Srikumar, 2016), machine translation (Mehdizadeh Seraj et al., 2015), text reformatting (Stein et al., 2014), or question answering (Fader et al., 2013). Paraphrasing can help to improve sentence compression and text summarization (Cordeiro et al., 2007) as well as natural language understanding (Ganitkevitch, 2013).

To tackle the task of paraphrasing, large amounts of paraphrases usually are helpful. Hence, paraphrase acquisition has gained popularity over the recent years and paraphrase datasets have been created manually, (semi-)automatically, and via distant supervision resulting in different levels of “naturalness”, diversity (topic, genre, language, etc.), granularity (word, sentence, or passage level), and scale (see Section 2 for details). For model training, Zhang et al. (2019b) strongly emphasize the importance of negative examples that are lexically similar but semantically inappropriate as paraphrases.



Flourine-containing durable water repellent makes a fabric water-resistant.

Water bead on a fabric that has been made non-wetting by chemical treatment.

Figure 1: Paraphrased captions of an image in the Wikipedia pages ‘Durable water repellent’ (left) and ‘Wetting’ (right).

With image captions, we propose an as yet underutilized source of natural paraphrases. The underlying hypothesis is that different captions for the same image form paraphrase candidates because the captions may describe the same image content. We study the phenomenon of ‘image reuse and captioning’ in the English Wikipedia (Figure 1 shows an example) by mining captions of images that have been reused. Carefully optimized filtering heuristics for captions and associated images yield a large set of captions, which we analyze qualitatively and quantitatively for their usefulness as paraphrases. The set of mined image captions forms our Wikipedia-IPC dataset,[1,2] which contains caption pairs as paraphrases along with their image source three different quality levels; 30,237 caption pairs are gold-, 229,877 are silver-, and 656,560 are bronze-quality paraphrase candidates.

---

[1] Data: https://doi.org/10.5281/zenodo.7621320

[2] Code: https://github.com/webis-de/EACL-23

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3348–3358
May 2-6, 2023 ©2023 Association for Computational Linguistics

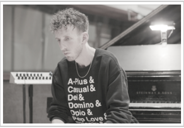
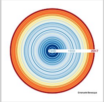
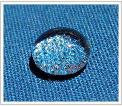
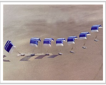
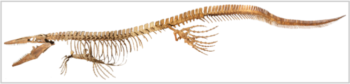
Image content		
Icons, symbols, pictograms	**Pictures, graphics, paintings, maps, schematics**	
	Image purpose	
	Ideological (used to convey atmosphere and sentiment)	**Specific, explanatory, pragmatic** (used in discourse)
 Brazilian flag  Up arrow  Wikimedia community logo  Yellow card  Icon for plain stage in cycling stage race  Motorway exit icon  Pictograms of Olympic sports Athletics 	 Paraguayan goalkeeper Justo Villar was awarded as the best goalkeeper of the tournament.  Romain Grosjean was once again the center of controversy when he collided on the first lap with Mark Webber.  AC-130 Spectres were highly effective during the battle.  J. C. A. Corea was born in the historic town of Chilaw, on the west coast of Sri Lanka.  February 10: New Delhi becomes India's capital.  Toronto's MaRS Discovery District is a centre for research in biomedicine.  Dostana was filmed almost entirely in Miami.  The Armenian genocide (pictured) was the first event officially condemned as ¨crimes against humanity¨.  "I Blame Myself" was co-written by Ariel Rechtshaid (pictured), who additionally assisted with the songwriting and production of each track on Night Time, My Time.	 Semi hollow-body electric guitar Gibson ES-335 has a ¨solid center block¨ inside a body.  This ¨circular warming stripes¨ graphic depicts average global temperature using chronologically ordered, concentric coloured rings.  Water bead on a fabric that has been made non-wetting by chemical treatment.  Dikes are used to protect the rice paddy fields from the channels of saltwater which overflow during high tide.  The camouflage grouper Epinephelus polyphekadion is the type-host of P. viscosus; the photograph shows a specimen from New Caledonia.  The stone is against the half-timbered wall of Exmewe Hall, on St. Peter's Square, Ruthin.  A sub-surface Metropolitan line A Stock train (left) passes a deep-tube Piccadilly line 1973 Stock train (right) in the siding at Rayners Lane.  The X-38 CRV prototype makes a gentle lakebed landing at the end of a July 1999 test flight at the Dryden Flight Research Center with a fully deployed parafoil.  Tylosaurus proriger mounted skeleton in the Rocky Mountain Dinosaur Resource Center in Woodland Park, Colorado.

Table 1: Example Wikipedia images and captions. Following Bär (2008), we categorize images by content as icons, symbols, pictograms (left), or pictures, graphics, etc. (middle and right). Pictures, graphics, etc. can be further subdivided by the purpose they serve in a text as either ideological (i.e., to convey sentiment; middle) or pragmatic (i.e., to explain and specify; right).

The usefulness of an image caption as a paraphrase depends strongly on its corresponding image. Table 1 shows example image–caption pairs from Wikipedia, organized by content and purpose classes. Icons, symbols, and pictograms rarely have captions that are useful as paraphrase candidates, while for pictures, graphics, paintings, maps, and schematics, the purpose of the image in a page is crucial. In this regard, "ideological purpose" means that the image is mainly used for aesthetic reasons, resulting in captions that do not describe the shown content but rather some context, atmosphere, or sentiment. By contrast, explanatory or pragmatic images try to illustrate or transport knowledge. Their captions describe and explain (parts of) the visual content and form a particularly promising source of natural paraphrases.

2 Related Work

Paraphrase Corpora To collect high-quality paraphrases, some studies applied manual acquisition. For instance, for the Webis-CPC-11 (Burrows et al., 2013), crowdworkers were asked to rewrite one of 4,096 passage-level literature excerpts *"so that the rewritten version has the same meaning, but a completely different wording and phrasing."* The resulting 7,859 pairs form one of the few passage-level paraphrase corpora. Later, also Xu et al. (2014) crowdsourced sentences with the same meaning as a shown sentence from some tweet for their PIT-2015 dataset (18,862 sentence pairs, 5,641 considered paraphrases).

Since crowdsourcing is costly to scale, automatic paraphrase acquisition often simply relies on machine translation. For instance, Zhang et al. (2019b) scrambled words and forth-and-back translated sentences to obtain a gold dataset of 108,463 paraphrase pairs and a silver dataset of 656,000 pairs. Also, the ParaNMT-50M dataset (Wieting and Gimpel, 2018) with 30 million "strong" paraphrases (assessment by the authors), was created by translating sentences from English-Czech parallel pairs.

Trying to combine the advantages of manual and automatic methods (quality vs. scale), most paraphrase acquisition methods use distant supervision. For instance, Barzilay and McKeown (2001) suggested aligned sentences from monolingual parallel corpora as paraphrases—an idea later used by Dolan and Brockett (2005) to create the Microsoft Research Paraphrase Corpus (5,801 manually annotated sentence pairs, 3,900 considered paraphrases). Instead of monolingual sources, Bannard and Callison-Burch (2005) exploited bilingual parallel corpora by aligning phrases that translate into the same pivot phrase in another language. Based on this idea, the English/Spanish PPDB 1.0 (Ganitkevitch et al., 2013) and its extension to 23 additional languages (Ganitkevitch and Callison-Burch, 2014) with an English portion of 7.6 million lexical (i.e., synonymous) and 68.4 million phrasal paraphrases form the biggest paraphrase collection.

Translation pivoting was also used for the Opusparcus (Creutz, 2018) and TaPaCo (Scherrer, 2020) datasets. Opusparcus was created from movie and TV subtitles in six European languages. The automatically filtered training set contains 7 million English sentence and phrase pairs ("good" or "mostly good" paraphrases according to the authors) while the manually labeled development and test sets contain 3,088 pairs (1,997 paraphrases). Analogously, TaPaCo is created from the multilingual Tatoeba data, a crowdsourced collection of sentence translations.[3] The English part of TaPaCo contains 158,000 sentences that are clustered in 62,000 paraphrase sets. Instead of translation, Lan et al. (2017) used a different pivoting method for their Twitter News URL Corpus. They mined tweets that contain the same hyperlink and labeled 51,524 respective sentence pairs via crowdsourcing. Our paraphrase acquisition methodology is also based on a pivoting technique, but the pivot media are images.

Captions and Paraphrases Our idea to use images from the web as pivots for distant supervision paraphrase acquisition is inspired by studies on crowdsourced caption datasets. For instance, Marelli et al. (2014) extracted a collection of 9,840 paraphrase candidates from the Flickr 8k dataset (Rashtchian et al., 2010) of 8,000 images each with 5 crowdsourced captions and the MSR Video Paraphrase Corpus (Chen and Dolan, 2011) of 2,000 short video clips with 85,000 English crowdsourced descriptions of what can be seen. The larger MSCOCO dataset (Lin et al., 2014) with 120,000 images each having 5 crowdsourced captions was even already used to train a neural paraphrase generation model (Prakash et al., 2016). Interestingly, the existing caption-paraphrasing datasets were all created in dedicated crowdsourcing processes that are costly to scale. We are the first to explore mining "real" image captions from the web (i.e., not crowdsourced) as a paraphrase acquisition method.

Dump	**Pages**	**Revisions**	**Images**	**Image refs.**
Small	22,311,116	22,311,116	4,508,808	7,384,368
Full	54,898,564	987,527,838	11,183,853	3,629,447,851

Table 2: Image mining statistics of Wikipedia dumps. "*Small*" refers to all articles, templates, media descriptions, and primary meta-pages without edit histories, whereas "*Full*" includes edit histories.

3 Caption-Based Paraphrase Acquisition

We collect captions of images that are used on the English Wikipedia, grouping those captions that describe the same image. This section outlines the mining and filtering steps.

Data Source: Wikipedia As a source for paraphrases, we use captions from Wikipedia articles. Wikipedia is a collection of well-formulated, human-written text that is constantly being improved by a community of volunteer editors. The guidelines for the use of images on Wikipedia encourage editors to write a caption to explain the relevance of an image in a particular context, resulting in 60% of images having a caption.

We use the English Wikipedia for the paraphrase acquisition (Wikimedia, September 1st, 2022).[4] Two versions of Wikipedia dumps are available for the paraphrase acquisition—a reduced dataset with all articles, templates, media descriptions, and primary meta pages and a full-size dump with all pages, including their full revision history. Table 2 compares the number of pages, revisions, images, and image references between these two datasets.

---

[3] https://tatoeba.org/eng/about

[4] https://dumps.wikimedia.org/enwiki/

	Images				Image references			Image captions		Paraphrase cand.	
Filter	Remaining	Δ	Refs. ≥ 2	Refs. ≥ 5	Remaining	Δ	μ_{img}	Remaining	Δ	Remaining	Δ
0. No filter	4,,508,808	–	941,339	102,046	7,384,368	–	1.64	4,571,671	–	355,619,088	–
1. References ≥ 2	941,339	-79%	941,339	102,046	3,816,899	-48%	4.05	2,053,338	-55%	355,619,088	0%
2. References ≤ 10	918,758	-2%	918,758	79,465	2,494,271	-35%	2.71	1,569,327	-23%	1,350,050	-100%
3. Has caption	713,815	-22%	494,321	32,949	1,499,479	-40%	2.10	1,569,327	0%	1,350,050	0%
4. Caption words ≥ 6	518,851	-27%	300,244	18,436	987,541	-34%	1.90	1,014,351	-35%	760,448	-44%
5. Caption is sentence	97,496	-81%	23,062	665	128,893	-87%	1.32	129,776	-87%	44,005	-94%
6. References ≥ 2	23,062	-76%	23,062	665	54,459	-58%	2.36	54,940	-58%	44,005	0%
7. Unique candidates	22,643	-2%	22,582	507	52,394	-4%	2.31	52,394	-5%	39,995	-9%
8. Divergent captions	18,979	-16%	18,979	391	43,616	-17%	2.30	43,616	-17%	32,830	-18%
9. Significant caption diff	17,293	-9%	17,293	386	40,131	-8%	2.32	40,131	-8%	30,237	-6%

Table 3: Effects of the filtering steps in the paraphrase acquisition pipeline on the number of images, references, captions, and paraphrase candidates mined from the reduced Wikipedia dump. Images that have been referenced at least 2 or 5 times are called "Refs. $\geq$ 2" and "Refs. $\geq$ 5", respectively. μ_{img} describes the average number of references per img.

Image and Caption Mining Wikimedia Commons is a service which hosts free-to-use images and other media files which are used across all Wikimedia projects (like Wikipedia). The markup language for Wikipedia articles, Wikitext, features the *extended image syntax*, which allows editors to insert images into Wikipedia pages from Wikimedia Commons. The specified image is referenced by its unique identifier in Wikimedia Commons, which is a concatenation of a generic media type prefix (e.g., Image) followed by a colon and its (also unique) filename. The extended image syntax allows the specification of style properties, meta-information, and a caption. If a caption is present, we store it along the image reference.

However, not all images in Wikipedia pages are included by the use of the extended image syntax. Wikitext allows the use of (user-created) style templates, often with inconsistent image reference syntax. *Infoboxes* on Wikipedia are template-based tables placed at the beginning of an article that contain a collection of the most important information about its subject. They are commonly used in Wikipedia—20% of all the pages contain one. Important to us is that one third of the infoboxes contain one or even multiple images, and about 56% of these images have a caption. Wikitext represents infoboxes as lists of key-value pairs whose sets of allowed keys and values are defined in templates. There are close to 2,000 different infobox templates based on semantic categories of an article. The responsible keyword for image referencing varies among these templates, and to cover all cases is not feasible. Thus, we identified the most commonly used keyword which is "image", followed by an optional counter for multiple images. These captions are mined as individual image–caption pairs. No Wikitext parser other than Wikipedia's own implementation (which is monolithically intertwined with Wikipedia's software stack) is capable of reliably parsing all the different template syntaxes. Our heuristic for image mining from infoboxes yields an increase of 24% of image references.

In addition to a caption, an image reference may also have an *alternative text* that accurately describes the visual content of an image. Alternative texts are intended to serve as a substitute in case the user is visually impaired, or the image cannot be displayed correctly. According to Wikipedia's policy, all images are obliged to have an alternative text, except those with a purely decorative purpose. We extract them as paraphrase candidates, too. Altogether, 355 million caption pairs are mined that we subsequently filter as follows (see Table 3).

Image Filter A necessary condition for an image to be useful is that it has been referenced at least twice, so we discard all image captions whose images have only one reference. As illustrated in Table 1, the likelihood of image–caption pairs to form a pair of paraphrase candidates depends on the image's content and purpose. A useful feature to discard icons, symbols, and pictograms is the number of references. These images occur more frequently than any other kind of image. To discard them, we set an upper bound of 10 references per image for the corresponding image–caption pairs to be retained. After these steps, 1.35 million image–caption pairs remain (see Table 3).

Caption Filter The selection of caption pairs has a substantial effect on the quality of the acquired paraphrases. As a consequence, all mined image captions pass through various carefully optimized pre-processing and filtering steps. The pre-processing pipeline comprises cleansing of Wikitext markup, removal of line-breaks, and whitespace normalization (e.g., removal of sequences).

To discard trivial image captions, we determined a lower bound for the caption length through manual observation and found that most captions that represent a phrasal expression contain at least 6 words. Captions with less than that usually comprise the name of an entity that can be seen in the image and are consequently discarded.

Through manual review, we identified three common practices for writing image captions with respect to style and syntax. Captions on Wikipedia are either: (1) simple noun phrases (e.g., "*Last Supper by Dieric Bouts*"), (2) sentence fragments containing a verb phrase (e.g., "*Last Supper drawn by Dieric Bouts*"), or (3) grammatically correct sentences (e.g., "*The Last Supper was drawn by Dieric Bouts*"). The latter two caption types are more likely to be a source of high-quality paraphrases, whereas grammatical sentences are most desirable. To discard noun phrases, it is sufficient to exclude captions that do not contain verbs. The detection of proper sentences is far more difficult.

Grammaticality and sentence fragment detection is a broad research field and most state-of-the-art methods use neural models. However, to ensure high precision in the resulting paraphrase dataset, we resorted to a more efficient and effective heuristic that is well suited for captions: rule-based sentence classification. Since the authors of existing rule-based grammar checkers rarely publish their full rule sets, we create our own rules to identify grammatical sentences. To find suitable rules, we take 500 and 100 random captions from Wikipedia to create a training and test set, respectively. An expert with 20 years of experience in the field of linguistics manually annotated the 600 captions to determine whether they are sentences or sentence fragments. Based on the 500 annotations from the training set and the associated automatically assigned POS tags, the expert manually created a set of rules to distinguish fragments from sentences.

Table 4 shows our sentence classification rules based on POS tags from Penn Treebank (Taylor et al., 2003). To be considered a sentence, a caption must satisfy one of these rules. Which rule should be applied is determined by its premises, which are tested in the order given. The first rule targets sentences that contain modals. A modal must be directly followed by an optional adverb and an obligatory verb in the base form. A sentence that would be correctly classified as such is "*Last Supper might be drawn by Dieric Bouts*". The second and third rules deal with subordinate clauses at the end of a sentence. These rules state that an inflected verb must precede a subordinating conjunction. For example, "*Last Supper was drawn by Dieric Bouts which is an exceptional artwork*" would be correctly classified as a sentence under Rule 2. Rule 4 requires that sentences contain an inflected verb and is applied when no other premise is satisfied. "*Dieric Bouts drew the Last Supper*" is a correctly classified sentence according to Rule 4.

<table>
<tr><th>Rule</th><th>Premise</th><th>Pattern</th></tr>
<tr><td>1</td><td>$.^{*}$MD$.^{*}$</td><td>$.^{*}$ MD RB? VB $.^{*}$</td></tr>
<tr><td>2</td><td>$.^{*}$ (WDT|WP|WRB) $.^{*}$</td><td>$[\neg($WDT|WP|WRB$)]^{*}$ (VBP|VBZ|VBD) $.^{*}$</td></tr>
<tr><td>3</td><td>$.^{*}$IN$.^{*}$</td><td>$[\neg$IN$]^{*}$ (VBP|VBZ|VBD) $.^{*}$</td></tr>
<tr><td>4</td><td>$\perp$</td><td>$.^{*}$ (VBP|VBZ|VBD) $.^{*}$</td></tr>
</table>

Table 4: POS-based rules for sentence classification.

Our heuristic sentence classifier performs satisfactorily with a precision of 94% and a recall of 79%, evaluated on the 100 annotated captions in the test set. Problematic cases are verbs that have identical base and inflected forms, which are often mislabeled by the POS tagger of the Stanford CoreNLP toolkit (Manning et al., 2014).

After application of these caption filters, in total 44,005 image–caption pairs remain (see Table 3).

Image Equivalence Wikipedia hosts all of its media files on Wikimedia Commons, and therefore we can identify duplicate images without requiring physical copies of an image. All images have a unique name that, along with a file type prefix, points to the URI of their description page on Wikimedia Commons. When an editor of a Wikipedia article wants to include an image from Wikimedia Commons, the image is referenced by this URI. Therefore, this URI is the criterion for deciding whether images are equivalent. All file type prefixes are derived from the common "File" prefix. Therefore, all image prefixes are normalized to "File" prefixes to detect image duplicates with inconsistent prefix usage.

Paraphrase Construction All references that refer to the same image are grouped into clusters. Paraphrase candidates are constructed by forming all possible unique pairs of captions of the same caption type (i.e., alternative text or regular caption) within each cluster. Combining regular captions and alternative texts rarely results in meaningful paraphrases due to their different purposes. Alternative texts describe the visual content of an image, while regular captions describe the image in context and from the perspective of an article.

Of these paraphrase candidates, we discard caption pairs that are exact or near-duplicates. We consider a caption pair to be a near-duplicate if it differs only by punctuation, capitalization, or whitespaces. (Near-)duplicates of captions are artifacts of reusing centrally hosted captions for the same image in different articles on Wikipedia. Removing duplicates results in 30,237 caption pairs as paraphrase candidates (see Table 3).

Corpus Construction We create three individual paraphrase datasets with different quality levels which we assign based on the used Wikipedia dump and the set of applied filtering heuristics.

Gold-quality paraphrase candidates are pairs of acquired image captions from the reduced Wikipedia dump which are classified as (sequences of) sentences by our rule-based classifier. Table 3 shows the loss of images, references, captions, and paraphrase candidates due to the application of our filter heuristics. In total, 30,237 gold-quality paraphrase candidates were acquired with our pipeline.

The most "aggressive" filter heuristic in the pipeline is the sentence classification of image captions. It discards 87% of the mined image captions. Since phrases can also be in a paraphrase relation, we construct a silver-quality paraphrase dataset by exchanging the sentence classifier for a heuristic that requires image captions to contain a verb. This heuristic excludes 50% of image captions for the paraphrase acquisition. This altered pipeline mined 229,877 silver-quality paraphrase candidates.

For the bronze-quality paraphrase dataset, we collect as many reasonable caption-pairs as possible, using the full Wikipedia dump with edit history as source for the paraphrase acquisition. We set the upper bound of allowed number of references per image to 18, the average number of revisions per page. Thus, an image can be referenced at most 180 times. The verb heuristic is applied. This yields 656,560 bronze-quality paraphrase candidates.

4 Corpus Analysis and Evaluation

We analyze and evaluate our paraphrase corpus based on the quantification of paraphrase similarity, for which we select well-known syntactic and semantic measures, and compare our corpus with five state-of-the-art paraphrase and three image–caption corpora from the literature. Therefore, we propose a new measure $\Delta_{\text{sem,syn}}$ that measures paraphrase sophistication based on the difference of semantic and syntactic similarities. To validate the proposed measure we analyze the correlation between human judgments and automatically computed semantic and syntactic similarities.

Analyzing Paraphrase Similarity

According to Wahle et al. (2022), high-quality paraphrases are texts pairs with high semantic similarity and high lexical and syntactic diversity, as paraphrasing models trained on linguistically diverse examples tend to be more robust (Qian et al., 2019). Similarly, Niu et al. (2021) measures paraphrase quality by rewarding high semantic similarity while penalizing high lexical overlap. The relation between semantic and syntactic similarity characterizes the "sophistication" of a paraphrase. Based on this assumption, we propose $\Delta_{\text{sem,syn}}$ to quantify the sophistication of paraphrases. $\Delta_{\text{sem,syn}}$ computes the average difference between the average semantic and syntactic similarity scores.

Similarity Measures To measure syntactic similarities between pairs of paraphrases, we choose ROUGE-1, ROUGE-L (Lin, 2004), and BLEU (Papineni et al., 2002). ROUGE-1 assesses unigram overlap, ROUGE-L computes lexical similarity based on the longest-common-subsequence paradigm, and BLEU is computed up to 4-grams to quantify structural similarity. All three measures have been proposed and evaluated for paraphrase generation and acquisition.

The computation of semantic similarity is done by transformer-based models. One of our three measures is BERTScore (Zhang et al., 2019a) since it was found to correlate well with human judgments for the task of automatic image captioning. The second one is the cosine similarity of dense vector representations computed by a BERT-based Sentence Transformer (Reimers and Gurevych, 2019). It proved to be decisive at Sent-Eval on the MSRPC. The third one is Word Mover Distance (Kusner et al., 2015) which computes the minimum amount of distance that embedded words of a text need to

	Syntactic similarity				Semantic similarity				
Image Caption Pairs	ROUGE-1	ROUGE-L	BLEU	Avg.	WMS	BERT	ST	Avg.	$\Delta_{\text{sem,syn}}$
A1: An Easter postcard from 1907 depicting a rabbit. **A2**: A 1907 postcard featuring the Easter Bunny.	0.53	0.13	0.14	0.27	0.76	0.76	0.90	0.81	0.54
B1: Twelfth century illustration of a man digging. **B2**: An English serf at work digging, c. 1170.	0.13	0.13	0.10	0.12	0.51	0.47	0.83	0.60	0.48
C1: Troops clearing rubble after the May air raid on Belfast. **C2**: Soldiers clearing rubble after the May air raid on Belfast.	0.90	0.90	0.99	0.93	0.92	0.89	0.98	0.93	0.00
D1: System of a Down is composed of four Armenian-Americans. **D2**: Dolmayan drumming with System of a Down in 2011.	0.42	0.42	0.33	0.39	0.50	0.07	0.32	0.30	-0.09

Table 5: Examples from the Wikipedia-IPC dataset with particularly high and low $\Delta_{\text{sem,syn}}$. Caption pairs with high $\Delta_{\text{sem,syn}}$ are structurally and lexically diverse while maintainig high levels of semantic similarity. Caption pairs with low $\Delta_{\text{sem,syn}}$ are either semantically dissimilar or have a high degree of lexical overlap.

"travel" to reach the embedded words of another text. We use the inversion of this measure as a Word Mover-based similarity score (WMS). All transformer-based models were applied with default hyperparameter configurations using a single NVIDIA A100 Tensor Core GPU. We normalize all the syntactic and semantic similarity measures to a scale from zero to one before we compute the average syntactic and semantic similarity.

Table 5 shows particularly sophisticated and unsophisticated caption pairs as paraphrases quantified by $\Delta_{\text{sem,syn}}$ from the Wikipedia-IPC (silver quality). Caption A1 is a paraphrase of A2 with a significantly different structure and sufficiently high lexical diversity. There is some unigram overlap, which causes the ROUGE-1 score to be comparatively high, since these captions are quite short. Due to the high semantic similarity of A1 and A2 these mutual paraphrases score high in the $\Delta_{\text{sem,syn}}$ measure. B1 and B2 have almost no words in common, but they are nevertheless semantically close. One semantic difference is that an "English serf" is a different concept than "a man" is, which could be the reason for the comparatively lower semantic similarity. However, the semantic similarity is high enough to yield a high $\Delta_{\text{sem,syn}}$. In (C1, C2) and (D1, D2) we see caption pairs that do not qualify as "good" paraphrases for two different reasons. In C2, a single word is replaced by a synonym in C1 and therefore has almost identical wording to C1. Although D1 and D2 are captions that both refer to the band "System of a Down", they describe entirely different aspects, which is reflected in the semantic similarity scores. Therefore, both pairs (C1, C2) and (D1, D2) receive a low $\Delta_{\text{sem,syn}}$ value.

Syntax:	**ROUGE-1**	**ROUGE-L**	**BLEU**	**Average**
r	0.78	0.77	0.70	0.79

Semantics:	**WMS**	**BERT**	**ST**	**Average**
r	0.59	0.70	0.78	0.76

Table 6: Top: Pearson correlation between manual judgements of syntactic similarity and the measures ROUGE-1, ROUGE-L, BLEU, and their average. Bottom: Pearson correlation between manual judgements of semantic similarity and the measures Word Mover Score (WMS), BERT-Score (BERT), Sentence Transformer (ST), and their average.

Manual Similarity Judgments In order to investigate whether the chosen similarity measures accurately quantify syntactic and semantic similarity, we conduct a manual annotation study to measure the correlation between the measures and human similarity ratings. For that purpose, we sampled 100 paraphrases from each of the three datasets–Wikipedia-IPC, MSRPC, and TaPaCo. We draw paraphrase examples with a length of 10 to 30 words stratified by their $\Delta_{\text{sem,syn}}$ distribution in their corresponding datasets. Four expert annotators with at least six years of experience rate semantic and syntactic similarity of all the 300 examples on a 5-point Likert scale.

Figure 2 shows the distribution of semantic and syntactic similarities of the median rating from the annotation study. These figures show that the semantic and syntactic similarities of the examples from the three different datasets are uniquely distributed. Examples from the TaPaCo corpus are pairs with subtle differences at the syntactic level that ultimately result in minor semantic differ-

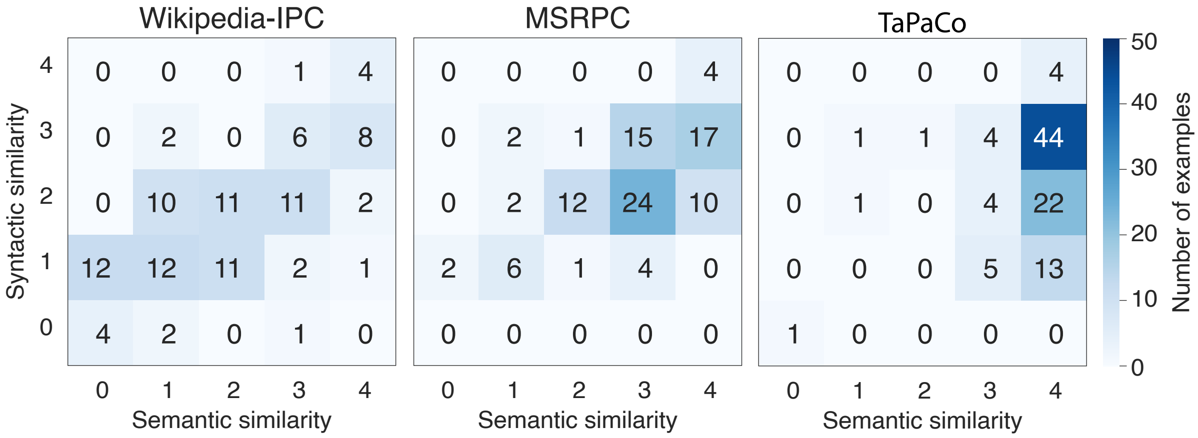


Figure 2: Distribution of semantic and syntactic similarity (characteristic map) for three datasets at sample size 100 for paraphrases with a length between 10 and 30 words. Among others, it can be seen that the TaPaCo corpus comprises paraphrases that result from subtle and minor changes within a paraphrase's sentence pairs (44 x syn-4/sem-3), while Wikipedia-IPC shows a more uniform distribution along the diagonal syn-0/sem-0…syn-4/sem-4.

Corpus		**Corpus statistics**				**Syntactic similarity**								**Semantic similarity**									
Subset	Type	Number		Length		ROUGE-1		ROUGE-L		BLEU		Avg.		WMS		BERT		ST		Avg.		$\Delta_{\text{sem,syn}}$	
ST $>$ 0.8		n	%	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Wikipedia-IPC$_{\text{gold}}$	C	10,327	34%	17.97	9.97	0.74	0.18	0.71	0.20	0.56	0.28	0.67	0.21	0.83	0.12	0.69	0.19	0.91	0.06	0.81	0.11	0.14	0.13
Wikipedia-IPC$_{\text{silver}}$	C	105,475	29%	20.38	13.56	0.71	0.19	0.67	0.22	0.52	0.28	0.63	0.22	0.63	0.20	0.81	0.12	0.90	0.06	0.78	0.12	0.15	0.12
Flickr8k	C	29,011	36%	11.65	3.70	0.53	0.16	0.48	0.17	**0.22**	0.14	0.41	0.14	0.59	0.13	0.73	0.10	0.86	0.05	0.73	0.08	**0.32**	0.09
PASCAL	C	3,329	33%	9.83	3.30	**0.51**	0.16	0.47	0.17	**0.22**	0.14	0.40	0.15	0.59	0.13	0.72	0.10	0.86	0.05	0.73	0.08	**0.32**	0.09
MS-COCO	C	392,248	32%	10.54	2.25	**0.51**	0.16	**0.45**	0.16	**0.22**	0.13	**0.39**	0.14	0.57	0.14	0.71	0.10	0.86	0.04	0.71	0.08	**0.32**	0.09
PAWS	G	64,940	99%	21.36	5.47	0.94	0.03	0.79	0.12	0.69	0.18	0.81	0.10	**0.82**	0.10	**0.96**	0.04	**0.97**	0.03	**0.92**	0.04	0.11	0.08
ParaNMT-5m	G	2,607,580	49%	11.97	6.18	0.63	0.15	0.60	0.16	0.33	0.19	0.52	0.16	0.60	0.14	0.75	0.17	0.87	0.05	0.74	0.09	0.22	0.13
MSRPC	H	3,720	64%	20.02	4.94	0.73	0.12	0.69	0.13	0.54	0.20	0.65	0.14	0.72	0.11	0.82	0.08	0.90	0.05	0.82	0.07	0.16	0.10
PPDB 2.0	H	654,531	24%	4.53	0.69	0.64	0.18	0.63	0.18	0.32	0.18	0.53	0.17	0.64	0.22	0.63	0.30	0.89	0.05	0.72	0.14	0.19	0.15
TaPaCo	H	117,447	52%	5.47	2.28	0.65	0.18	0.63	0.18	0.30	0.16	0.53	0.16	0.78	0.14	0.79	0.21	0.91	0.06	0.83	0.10	0.30	0.13

Table 7: Comparison of syntactic similarity (measured as the average of ROUGE-1, ROUGE-L, and BLEU) and paraphrase semantic similarity (measured as the average of BERTScore, Sentence Transformer (ST) and Word Mover Score (WMS)) of semantically similar examples (Sentence Transformer $>$ 0.8) from state-of-the-art paraphrase corpora, image caption corpora, and the new silver-quality Wikipedia-IPC. "Type" classifies the acquisition method in caption (C), automatically generated (G), and human-written (H) (e.g., acquired via crowd-sourcing).

ences. In contrast, examples from the Wikipedia-IPC (and MSRPC to some extent) are more evenly distributed along the diagonal syn-0/sem-0, …, syn-4/sem-4. While Wikipedia-IPC has examples from all semantic similarity levels, semantic similarity is higher among examples from the MSRPC.

To justify the choice of the semantic and syntactic similarity measures, we compute Pearson correlation coefficients with manual similarity judgments. Table 6 shows high correlation values for syntactic and semantic similarity ratings, which renders our measures well suited for the given task.

Quantitative Similarity Analysis In the paraphrase literature, authors usually require pairs of texts to have varying degrees of semantic similarity to consider them paraphrases of each other. The choice of a "semantic threshold" drastically affects the distribution of syntactic and semantic similarities of paraphrases within a dataset. To standardize the decision criteria for paraphrases, we manually set a semantic threshold that allows us to comparably evaluate paraphrases from each dataset.

We choose the semantic similarity score from the Sentence Transformer as the decision criterion for the subset of paraphrases under consideration because it has the highest correlation with human ratings of semantic similarity. To find a reliable threshold, we analyze the distributions of Sentence Transformer scores of paraphrases and non-paraphrases within the MSRPC and found that a threshold of 0.8

correctly classifies paraphrases with a precision and recall of 82% and 77%, respectively.

Table 7 shows the results of the comparison of syntactic and semantic similarities of paraphrase pairs with a high semantic similarity (ST > 0.8) grouped by type which is either image caption (C), automatically generated (G), or human-written (H). f calculates the frequency of text pairs that exceeds the semantic threshold of 0.8 relative to all examples in the dataset while $\Delta_{\text{sem,syn}}$ computes the average difference between the average semantic and syntactic similarity scores.

The high syntactic and semantic similarities of the examples from the PAWS dataset are striking. PAWS was designed as a benchmark corpus for paraphrase detection and contains lexically similar text pairs with subtle semantic inconsistencies that are difficult to distinguish from actual paraphrases. The high ROUGE-1 value of 94% with a low standard derivation of 0.03 translates to an almost identical wording. Hence, it is no surprise that the $\Delta_{\text{sem,syn}}$ is the lowest among all datasets.

From the gold and silver quality proportions of the Wikipedia-IPC, 34% and 29% of the examples exceed the semantic threshold, respectively. These rates are in line with those of the other caption corpora studied. Furthermore, we find that all caption datasets have similar overall syntactic and semantic similarities. However, the caption pairs from the Wikipedia-IPC tend to have higher syntactic similarity. We suspect that this observation is due to Wikipedia's caption reuse policy, which allows editors to use slightly modified existing captions rather than inventing a new text. However, the sufficiently large $\Delta_{\text{sem,syn}}$ of the Wikipedia-IPC, which is on par with the commonly used MSRPC, and larger than that of the other caption corpora strongly suggests the suitability of image captions as a source of interesting paraphrases.

5 Conclusion and Future Work

We propose a new approach to use image captions as a resource for paraphrasing. From Wikipedia, we extracted 30,237 caption pairs in gold quality, 229,877 in silver quality, and 656,560 in bronze quality for our new Wikipedia-IPC dataset. As part of our analysis, we found that many caption pairs are "sophisticated" paraphrases in the sense of being semantically similar but dissimilar at the lexical and syntactic levels. We have introduced a respective measure for assessing paraphrase sophistication based on semantic and syntactic similarity. The new measure correlates well with manual judgments, and we have shown that paraphrases from different sources have individual characteristics along the two similarity dimensions. Studying large sets of paraphrases with characteristics required by a specific task (e.g., sophisticated vs. only word order changed) will be useful for task-specific paraphrase recognition and generation approaches, and for natural language understanding as a whole.

In future work, we plan to extend the mining process to larger web resources than the Wikipedia and to apply image analysis as a new equivalence criterion. Identifying highly similar but not identical images may help to identify even more caption-based paraphrase candidates.

Limitations

Challenging for our proposed paraphrase acquisition approach are captions that describe the same image in very different contexts (e.g., maps). Such captions tend to focus on rather different aspects of the image (e.g., rivers vs. cities on a map) but since the captions often still share some commonalities, the semantic component of our proposed measure $\Delta_{\text{sem,syn}}$ might still classify the captions as similar. A "deeper" estimation of semantic similarity could help. With more research on image–text relationships, applying models such as CLIP (Radford et al., 2021) or Stable Diffusion (Rombach et al., 2022) to take image semantics into account might lead to improvements for our proposed paraphrase refinement measure.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.

Colin J. Bannard and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 597–604.

Janina Bär. 2008. *Analyse zur automatischen*

Klassifizierung von Bildern. Ph.D. thesis, Technische Universität Chemnitz.

Regina Barzilay and Kathleen McKeown. 2001. Extracting Paraphrases from a Parallel Corpus. In *Association for Computational Linguistic, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference, July 9-11, 2001, Toulouse, France*, pages 50–57.

Steven Burrows, Martin Potthast, and Benno Stein. 2013. Paraphrase Acquisition via Crowdsourcing and Machine Learning. *Transactions on Intelligent Systems and Technology (ACM TIST)*, 4(3):43:1–43:21.

David Chen and William Dolan. 2011. Collecting Highly Parallel Data for Paraphrase Evaluation. In *Proceedings of the Forty-Ninth Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, Portland, Oregon. Association for Computational Linguistics.

Joao Cordeiro, Gael Dias, and Pavel Brazdil. 2007. New Functions for Unsupervised Asymmetrical Paraphrase Detection. *Journal of Software*, 2(4):12–23.

Mathias Creutz. 2018. Open Subtitles Paraphrase Corpus for Six Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing*, pages 1–8, Jeju, South Korea. Kazuhide Yamamoto.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-Driven Learning for Open Question Answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618, Sofia, Bulgaria. Association for Computational Linguistics.

Juri Ganitkevitch. 2013. Large-Scale Paraphrasing for Natural Language Understanding. In *Proceedings of the 2013 NAACL HLT Student Research Workshop*, pages 62–68, Atlanta, Georgia. Association for Computational Linguistics.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The Multilingual Paraphrase Database. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4276–4283, Reykjavik, Iceland. European Language Resources Association (ELRA).

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Hamid Izadinia, Fereshteh Sadeghi, Santosh K. Divvala, Hannaneh Hajishirzi, Yejin Choi, and Ali Farhadi. 2015. Segment-Phrase Table for Semantic Segmentation, Visual Entailment and Paraphrasing. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From Word Embeddings To Document Distances. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 957–966, Lille, France. PMLR.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A Continuously Growing Dataset of Sentential Paraphrases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1224–1234, Copenhagen, Denmark. Association for Computational Linguistics.

Tao Li and Vivek Srikumar. 2016. Exploiting Sentence Similarities for Better Alignments. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2193–2203, Austin, Texas. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Ramtin Mehdizadeh Seraj, Maryam Siahbani, and Anoop Sarkar. 2015. Improving Statistical Machine

Translation with a Multilingual Paraphrase Database. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1379–1390, Lisbon, Portugal. Association for Computational Linguistics.

Tong Niu, Semih Yavuz, Yingbo Zhou, Nitish Shirish Keskar, Huan Wang, and Caiming Xiong. 2021. Unsupervised Paraphrasing with Pretrained Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5136–5150, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural Paraphrase Generation with Stacked Residual LSTM Networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Lihua Qian, Lin Qiu, Weinan Zhang, Xin Jiang, and Yong Yu. 2019. Exploring Diverse Expressions for Paraphrase Generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3173–3182, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. 2010. Collecting Image Annotations Using Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 139–147, Los Angeles. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695.

Yves Scherrer. 2020. TaPaCo: A Corpus of Sentential Paraphrases for 73 Languages. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6868–6873, Marseille, France. European Language Resources Association.

Benno Stein, Matthias Hagen, and Christof Bräutigam. 2014. Generating Acrostics via Paraphrasing and Heuristic Search. In *25th International Conference on Computational Linguistics (COLING 2014)*, pages 2018–2029. Association for Computational Linguistics.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. *The Penn Treebank: An Overview*, pages 5–22. Springer Netherlands, Dordrecht.

Jan Philip Wahle, Terry Ruas, Frederic Kirstein, and Bela Gipp. 2022. How Large Language Models are Transforming Machine-Paraphrase Plagiarism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 952–963, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

John Wieting and Kevin Gimpel. 2018. ParaNMT-50M: Pushing the Limits of Paraphrastic Sentence Embeddings with Millions of Machine Translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Melbourne, Australia. Association for Computational Linguistics.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. 2014. Extracting Lexically Divergent Paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2:435–448.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019a. BERTScore: Evaluating Text Generation with BERT. *CoRR*, abs/1904.09675.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Generation-Based Data Augmentation for Offensive Language Detection: Is It Worth It?

Camilla Casula◇♣**, Sara Tonelli**◇
◇Fondazione Bruno Kessler, Trento, Italy
♣University of Trento, Italy
`{ccasula,satonelli}@fbk.eu`

Abstract

Generation-based data augmentation (DA) has been presented in several works as a way to improve offensive language detection. However, the effectiveness of generative DA has been shown only in limited scenarios, and the potential injection of biases when using generated data to classify offensive language has not been investigated. Our aim is that of analyzing the feasibility of generative data augmentation more in-depth with two main focuses. First, we investigate the robustness of models trained on generated data in a variety of data augmentation setups, both novel and already presented in previous work, and compare their performance on four widely-used English offensive language datasets that present inherent differences in terms of content and complexity. In addition to this, we analyze models using the HateCheck suite, a series of functional tests created to challenge hate speech detection systems. Second, we investigate potential lexical bias issues through a qualitative analysis of the generated data. We find that the potential positive impact of generative data augmentation on model performance is unreliable, and generative DA can also have unpredictable effects on lexical bias.

⚠ **Warning**: *this paper contains examples that may be offensive or upsetting.*

1 Introduction

Even though large language models have been found to have a tendency to encode and propagate undesirable social bias (Bender et al., 2021), going as far as generating toxic sequences starting from non-toxic prompts (Gehman et al., 2020), the use of synthetic data for offensive language detection has been found to be potentially helpful in improving models (e.g. Juuti et al. (2020); Wullach et al. (2021); D'Sa et al. (2021)). Indeed, data augmentation (DA) through generation has the potential to mitigate some of the known issues of smaller datasets, which are common in offensive language detection, such as lack of linguistic variation and risk of overfitting (Vidgen and Derczynski, 2020). Furthermore, synthetic data can overcome privacy issues related to the use of social media data obtained without user consent in research experiments. It can also mitigate dataset decay, an issue affecting reproducibility, since online messages, especially abusive ones, tend to be deleted over time, while synthetic examples do not present this issue (Klubicka and Fernández, 2018).

While generative DA has been shown to be potentially useful for the task of detecting offensive and abusive language online in multiple works, several aspects and implications of it remain unexplored. First of all, generative DA has mostly been shown to work for offensive language detection when starting with a single specific dataset and using a specific generation setup, with no investigation of the impact of different generation setups on the quality of the augmented data, as well as little exploration of cross-dataset or cross-domain performance. The first aim of our work is therefore that of assessing the *robustness* of models across different sources of variation as follows: *i)* we train and test our models using four English offensive language datasets, testing both within dataset and cross-dataset performance; *ii)* we simulate two low-resource scenarios, in which we start with different quantities of gold examples; *iii)* we compare four different generation setups, of which two were used in previous work and two are novel; *iv)* we experiment with different thresholds for filtering the generated data prior to using it for training.

Our second aim for this work is that of conducting a qualitative analysis on the generated data, with a focus on *lexical bias*. In order to do this we compute the correlation between tokens in offensive texts using PMI[1], and we test the models

---

[1] For this, we use the implementation by Ramponi and Tonelli (2022).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3359–3377
May 2-6, 2023 ©2023 Association for Computational Linguistics

trained on augmented data on the HateCheck suite (Röttger et al., 2021), which includes a series of functional tests aimed at finding model weaknesses.

2 Related Work

Model-based data augmentation exploiting large language models (LLMs) such as GPT-2 (Radford et al., 2019) has been found effective for various NLP tasks. One method that has been shown to be promising is fine-tuning GPT-2 on annotated data, and then using it to generate additional similar data. The most common approach is prepending labels to sequences during fine-tuning, and then using labels as prompts for the model to generate sequences belonging to specific classes (Anaby-Tavor et al., 2020; Tepper et al., 2020; Kumar et al., 2020).

Similar methods have also been successfully applied to abusive language classification. For instance, Juuti et al. (2020) find that DA using a fine-tuned GPT-2 model leads to performance improvements in very low-resource scenarios. Liu et al. (2020) use a conditional variant of GPT-2 based on reinforcement learning, where lexical features for each class are extracted from the entire dataset and then used for generation. Wullach et al. (2021) and D'Sa et al. (2021) use GPT-2 to generate synthetic hate speech data. They find that the addition of large amounts of synthetic data helps classification when starting from datasets containing thousands of labeled instances. While D'Sa et al. (2021) follow the label-prepending approach of Anaby-Tavor et al. (2020), Wullach et al. (2021) train a separate generative model on data belonging to each class. Both approaches are found effective, but they have never been comparatively evaluated.

To our knowledge, the robustness of models trained using generation-based DA has not been analyzed in depth. While Wullach et al. (2021) test their models cross-dataset, results are presented in a setup in which models are trained on 4 datasets together and tested on a fifth one. This setup presupposes that multiple datasets can be used at once for training models. However, this might not always be the case when DA is needed, so we evaluate cross-dataset setups in which only one dataset is available for training. In addition, to our knowledge ours is the first work to pair a robustness analysis with a qualitative analysis of lexical bias in the context of generative DA for this task.

3 Data

3.1 Dataset Description

We use four English datasets annotated for offensive or abusive language for training and testing our models. These datasets have been chosen because they are widely used and they differ in terms of content, since they were created to study different aspects of offensive language. Intuitively, this should allow us to assess the out-of-domain behavior of models when doing cross-dataset testing.

Agreement [AG] This dataset by Leonardelli et al. (2021) is annotated for offensive language and agreement level among annotators. It contains over 10k tweets dealing with three widely discussed topics on Twitter: the Black Lives Matter movement, the 2020 US elections, and Covid-19. Offensive tweets constitute 31% of the dataset.[2]

Founta [FO](Founta et al., 2018). This dataset is among the most widely used abusive language datasets in the literature, and it has been already employed for generative data augmentation (Wullach et al., 2021; D'Sa et al., 2021). It contains around 100k Twitter posts annotated using four labels: hateful (7.5%), abusive (11%), normal (59%), and spam. In order to keep a binary classification setup that is consistent with the other datasets we use in our experiments, we group the *hateful* and *abusive* classes together into one single *abusive* class, following Leonardelli et al. (2021).[3]

OLID [OL] The Offensive Language Identification Dataset (Zampieri et al., 2019). This dataset consists of 14,100 Twitter posts annotated for offensive language with two more fine-grained levels of annotation regarding the target of the offense. In our experiments we only consider the broader binary level of annotation, for which 33% of the dataset is labeled as *offensive*. The test set we pair with this dataset is **SOLID [SO]** (Zampieri et al., 2020), which was used in the OffensEval 2020 shared task and follows the same annotation guidelines.[4]

SBIC [SB] The Social Bias Inference Corpus (Sap et al., 2020) contains 40k posts from Twitter, Reddit, and Stormfront, of which 44.8% offensive. While this dataset provides fine-grained annotations on social biases, we only consider the binary offensive/not offensive labels in our experiments.[5]

---

[2] https://github.com/dhfbk/annotators-agreement-dataset

[3] https://zenodo.org/record/3678559

[4] https://sites.google.com/site/offensevalsharedtask/olid

[5] homes.cs.washington.edu/~msap/social-bias-frames/

The above datasets present different characteristics. We consider **[FO]** and **[OL]** rather easy to classify, since standard BERT-based approaches trained and tested on these datasets yield results above 0.90 macro-F1 (Zhou et al., 2021; Zampieri et al., 2020). Past works showed that, in case of **[OL]**, classifiers may perform very well because of the limited presence of ambiguous tweets (Leonardelli et al., 2021). In contrast, **[AG]** was explicitly created to study disagreement among annotators focusing on different topics, so it contains more challenging instances. On this dataset, the best performance reported by the authors is ∼0.75 macro-F1 (Leonardelli et al., 2021). Finally, **[SB]** includes data from different sources, with annotations for diverse targets of hate. The best classification result reported by the authors is ∼0.80 F1 (Sap et al., 2020).

3.2 Data Splits and Preprocessing

We use the default train/test splits of each dataset, where available. For **[FO]**, which has no default splits, we randomly partition the data into train and test using an 80/20 split. For all datasets, we replace URLs and user mentions with `URL` and `@USER` respectively. We then remove all duplicates. We also remove the substring "`RT:`" from the beginning of sequences in the **[FO]** dataset, since it is extremely common and it could be a confounder for the model. In addition to this, it has been found to be associated with hate speech in this dataset (Ramponi and Tonelli, 2022). Since there is a partial overlap between **[SB]** and **[FO]**, we remove instances that are present in the test set of either dataset from the training data of the other, to ensure fair cross-dataset evaluation.

4 Methods

We aim at comparing the performance of different data augmentation setups, both novel and already employed in previous work. We test them in within-dataset and cross-dataset scenarios, to assess the impact of synthetic data on model robustness across setups. Below is an overview of the process we follow, whose specifics are detailed in Section 5.

1. We randomly undersample the training data, obtaining the data subset X consisting of n examples (Sec. 5.1).
2. We fine-tune the pre-trained classification model C on X, obtaining C_X, which is used as a baseline and filtering classifier.
3. Depending on the type of generation input (Sec. 5.2) the pre-trained generation model G is fine-tuned on the available training data X, obtaining G_X.
4. The generative model G_X is used to generate synthetic examples.
5. The examples generated by G_X are preprocessed and then filtered based on the probability assigned to them by the classification model C_X (Sec. 5.3).
6. The generated data is merged with the gold data X to create the augmented dataset X_{aug}.
7. The classifier C is fine-tuned on the augmented dataset X_{aug} to create $C_{X_{aug}}$.

Model choice We focus on the generation of synthetic data using GPT-2 `large` (774M parameters) (Radford et al., 2019). [6] Some recent works exploit the generative capabilities of GPT-3 for the creation of new datasets, either in human-in-the-loop setups (Liu et al., 2022) or in very resource-intensive scenarios (Hartvigsen et al., 2022a). We choose to experiment with GPT-2 because it is freely accessible and it can be easily fine-tuned, and we aim for our results to be comparable with those of previous work where this DA method was found effective for this task (e.g. Juuti et al. (2020) and Wullach et al. (2021)).

Model Details For classification, we run our experiments with the BERT `base uncased` model (110M parameters) (Devlin et al., 2019) and with RoBERTa `base` (125M parameters) (Liu et al., 2019). We use the Huggingface implementation (Wolf et al., 2020) for all models. In both cases we use the default Huggingface `TrainingArguments` class hyperparameters, with batch size set to 32.

For generation, we fine-tune GPT-2 `large`, following Wullach et al. (2021). We use the default Huggingface hyperparameters, setting the batch size to 2, adding learning rate warm-up with a ratio of 0.02 and weight decay of 0.01. Classifiers and generative models are trained for 3 epochs. For fine-tuning GPT-2, the input texts are grouped into documents of maximum length 512 tokens and separated using end-of-sequence tokens.

After fine-tuning, the generation step is similar for all models. We use *top p* decoding (Holtzman

---

[6] We performed preliminary experiments using GPT-2 small (117M parameters) as well, finding that overall the generated data had a similar impact on classification performance.

et al., 2020) with $p = 0.9$ and we set the minimum and maximum lengths of generated sequences to 5 and 100 tokens respectively. We also blacklist the sequence "`@USER`" so that it will not be generated, since it is a very frequent token combination in the normalized training data.

In all setups, we aim at augmenting the gold data with 2,000 synthetic examples. This number is chosen to at least double the available training data in all setups, and it is kept constant for easier model comparison. We generate 6,000 sequences for each setup, to ensure that enough acceptable sequences will be generated. This estimate is based on the approach of Wullach et al. (2021), who preserve roughly 1/3 of the generated texts after filtering.

All experiments are run on a NVIDIA Quadro RTX 5000 GPU in ~80 hours total, including both training and inference for all setups.

5 Experimental Setting

We structure our experiments along three axes of variation, with the aim of assessing their impact on model performance. The explored dimensions are further detailed in the following subsections.

- **Number of training instances**. In order to simulate two low-resource scenarios where different amounts of gold data are available, we train both classification and generative models with different amounts of labeled instances. Our aim is that of assessing how much the usefulness of generative DA changes when starting with datasets of different sizes.
- **Prompting.** Different methods can be used for steering the generation towards one label or the other. We use two methods found in previous works, as well as two novel methods, to assess whether certain prompting methods lead to differences in synthetic data quality.
- **Classifier filtering thresholds.** Since prompting methods are not always enough to steer the model into generating correct sequence-label pairs (Kumar et al., 2020), classifiers can be used to confirm or discard the label assignments made by the generative model (Anaby-Tavor et al., 2020; Wullach et al., 2021). In our experiments, we feed the generated sequences to a classifier (our baseline) and use the probability given by the classifier to each generated sequence to either accept the label assigned by the generator or discard the sequence entirely. We experiment with two probability thresholds, in order to assess whether the confidence of the classifier is associated with generated data quality.

Each model is tested on its own test data (within-dataset) and on the test data for the other datasets (cross-dataset).

5.1 Number of Training Instances

Each experiment is performed on varying amounts of training data, randomly sampling $n = 500$ or 2,000 examples from each dataset, equally split between the two labels. We use 500 examples as the smallest sample size for our experiments since the smallest dataset size for this task found by Vidgen and Derczynski (2020) is 469 examples. We use 2,000 examples as the larger sample size given that it is still a relatively small dataset size for deep learning approaches and it reflects the size of many offensive language detection datasets.

We balance the sampling by class to avoid imbalance between gold and augmented data, consistently keeping this proportion even across all experiments. For the **[AG]** dataset, sampling is stratified by agreement level as well. Balancing the classes might make our setup less "realistic", given that it does not reflect the actual label distribution of each dataset. However, it is a way for us to control the impact of class balance differences between datasets on cross-dataset performance. It also helps to avoid differences in class balance between the gold data and the generated data, which could cause differences in model performance between setups regardless of the actual quality of the generated data.

Out of the available data, 1/5 ($n = 500$) or 1/10 ($n = 2,000$) is held out for validation.

5.2 Prompting

We fine-tune GPT-2 using four data formatting setups. Two of the setups have been employed in previous works, while two are novel and aim at exploring the ability of the model to leverage natural language task descriptions for label assignment.

Label tag prompting (tag-prompt). Following the prompting type in Anaby-Tavor et al. (2020), we fine-tune the generator G by pre-pending the label y to each training sequence x, dividing the two with the separator "`[SEP]`". In this setup, the inputs are concatenated into documents as follows:

"y_1 `[SEP]` x_1 `[EOS]` y_2 `[SEP]` `...`"

At generation time, the model is prompted with the desired label y followed by the separation token, and it is expected to generate a sequence belonging to the y class.

Label in natural language prompting (nl-prompt). This is the first input setup we propose. It is inspired by the findings of Schick and Schütze (2021), in which natural language descriptions of tasks are found to be helpful for few-shot classification tasks. In this setup, the generator G is trained on sequences so that the label y is contextualized within the text using natural language. The training documents for fine-tuning the generators are structured as:

"`This message is` y_1`.` x_1 `[EOS] This ...`"

Where y corresponds to *offensive* or *not offensive* depending on the label. At generation time, the model is prompted with "This message is y", where y is the desired label. The sequence produced after the prompt is expected to belong to the y class.

Cloze question prompting (cloze-prompt). Again inspired by the findings in Schick and Schütze (2021), we propose another setup that exploits the capability of large language models of learning from patterns in natural language. In this case, however, the prompt relies on the auto-regressive nature of GPT-2, in which the probability of each token is modeled on the previous tokens. The main aim behind this setup is assessing whether placing the label information at the beginning or at the end of the sequence affects the quality of the generated data. In this setup, each sequence x is followed by the cloze question "Is that offensive?" and the label is placed at the end of the sequence, in the form of a Yes/No answer.

"x_1`. Is that offensive? {Y/N} [EOS] ...`"

At generation time, the model receives no prompting, and it is expected to generate both the sequence and the cloze question / answer pair in the correct format. This type of prompting is more prone than the previously listed ones to generating sequences that will eventually be discarded, since it is expected to not only correctly generate sequences and assign them to a label, but also to produce a cloze question that follows a specific format.

One model per label (1/label). This setup requires no actual prompting to steer the generation, since it involves one model for each label rather than one model for all labels. Following Juuti et al. (2020) and Wullach et al. (2021), the training dataset X is divided into X_o and X_n based on the *offensive* or *non-offensive* labels. The generative model G is then fine-tuned on X_o and X_n separately, producing two models for the generation of new data: G_o and G_n. In this setup, the messages are simply concatenated into documents and separated by end of sequence (`[EOS]`) tokens:

"x_1 `[EOS]` x_2 `[EOS]` x_3 `...`"

At generation time, each model is expected to generate sequences belonging to the class it was fine-tuned on.

5.3 Classifier Filtering Thresholds

After generation, the synthetic sequences are stripped of any prompting and automatically assigned the label that emerged during generation. We discard any sequence that is ≤ 5 characters long, and normalize the generated data following the steps described in Section 3.2.

Then, we feed the sequences into the baseline classifier trained on the same gold data as the generative model that produced them. Depending on the label probability assigned by the classifier to the generated sequences, these are accepted considering the following thresholds:

- The label predicted by the classifier matches the label assigned during the generation phase (label probability $p > 0.5$)
- The classifier predicts the same label assigned during generation with $p > 0.7$ [7]

After filtering, we randomly select 2,000 generated examples from the accepted ones in each setup.

5.4 Baselines

As baselines, we employ a `BERT-base-uncased` and a `RoBERTa-base` classifier trained on the same gold data used to fine-tune GPT-2 in each setup.

We also report the performance of classifiers trained using simple oversampling as a DA strategy, in which a number of randomly selected training examples appear multiple times during training. We match the number of oversampled instances with the number of synthetic examples we use for augmenting the training data in each setup, split evenly across labels. Using oversampling as a baseline allows us to compare more resource-intensive DA methods such as the ones we are evaluating with a simpler strategy.

---

[7] This is the same threshold used by Wullach et al. (2021).

Gold data: 500 examples		Test			
Train: **AGREEMENT**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.655 (0.603)	0.805 (0.743)	0.543 (0.537)	0.807 (0.734)
Oversampling		**0.725** (0.623)*	**0.882** (0.768)	0.554 (0.566)	**0.875** (0.757)*
	tag-prompt	0.700 (0.638)*	0.859 (0.810)*	0.547 (0.524)	0.862 (0.804)*
Filtering:	nl-prompt	0.694 (0.638)*	0.863 (0.820)*	0.560 (0.546)	0.863 (0.806)*
$p > 0.5$	cloze-prompt	0.692 (0.634)*	0.860 (0.815)*	0.545 (0.524)	0.859 (0.803)*
	1/label	0.716 (0.656)*	0.872 (0.834)*	**0.572** (0.567)	0.874 (0.823)*
Train: **FOUNTA**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.683 (0.622)	0.904 (0.874)	0.540 (0.504)	0.888 (0.844)
Oversampling		0.637 (0.585)	0.900 (0.876)	**0.589** (0.591)*	0.896 (0.841)
	tag-prompt	0.679 (0.620)	0.909 (0.881)	0.567 (0.542)	**0.897** (0.857)
Filtering:	nl-prompt	0.660 (0.611)	0.909 (0.882)	**0.589** (0.575)*	0.895 (0.854)
$p > 0.5$	cloze-prompt	**0.688** (0.626)	**0.913** (0.884)*	0.559 (0.527)	0.891 (0.850)
	1/label	0.683 (0.624)	0.910 (0.882)	0.579 (0.563)*	0.893 (0.851)
Train: **SBIC**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.556 (0.413)	0.646 (0.472)	0.746 (0.780)	0.714 (0.570)
Oversampling		**0.591** (0.481)*	**0.700** (0.540)*	**0.780** (0.801)*	**0.766** (0.643)*
	tag-prompt	0.561 (0.447)	0.679 (0.531)	0.765 (0.805)*	0.744 (0.618)
Filtering:	nl-prompt	0.578 (0.449)	0.687 (0.540)	0.763 (0.803)*	0.746 (0.622)
$p > 0.5$	cloze-prompt	0.574 (0.438)	0.663 (0.497)	0.762 (0.799)*	0.737 (0.604)
	1/label	0.584 (0.477)*	0.676 (0.524)	0.771 (0.805)*	0.757 (0.636)*
Train: **OLID**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.568 (0.515)	0.766 (0.676)	0.585 (0.588)	0.797 (0.707)
Oversampling		0.584 (0.570)	**0.838** (0.792)*	**0.637** (0.717)*	**0.865** (0.804)*
	tag-prompt	0.578 (0.567)	0.812 (0.755)	0.610 (0.644)	0.845 (0.786)
Filtering:	nl-prompt	0.581 (0.564)	0.811 (0.763)	0.615 (0.652)	0.838 (0.781)
$p > 0.5$	cloze-prompt	**0.586** (0.565)	0.816 (0.763)	0.618 (0.656)	0.843 (0.783)
	1/label	0.575 (0.584)	0.831 (0.791)	0.631 (0.697)	0.855 (0.810)

Table 1: Average macro-F1 scores (over 10 runs) obtained by RoBERTa-base fine-tuned on augmented data, starting with 500 gold examples. F1 scores for the minority class are in parentheses. Grey cells contain within-dataset results, while the others contain cross-dataset results. Asterisks denote statistically significant results (compared to no augmentation). The best result for each train-test dataset combination is in bold.

6 Results

In this section we report the results of our experiments. Each experiment is run 10 times, with different random seeds. The metric we use to evaluate models is macro-F1 score.

In order to reliably compare the distributions of results across runs, we use Almost Stochastic Order (ASO) (Dror et al., 2019; Del Barrio et al., 2018) in its implementation by Ulmer et al. (2022). Following their findings, we use $\tau = 0.2$ as a threshold for statistical significance.[8]

Table 1 and Table 2 show the results obtained by RoBERTa-base models fine-tuned on augmented data when starting with 500 and 2,000 gold examples respectively. While for the setup in which we start with 2,000 annotated examples (Table 2) we use both filtering thresholds ($p > 0.5$ and $p > 0.7$), for the setup in which we start with 500 examples we report the results for models trained on generated data filtered with the $p > 0.5$ threshold only. The reason for this is that with less data, the confidence of the model is much lower, and not all 10 runs can generate enough examples that are classified with a confidence score higher than 0.7.

Impact of number of training instances Overall, it appears that data augmentation is more effective in very low-resource scenarios, such as the setting with 500 examples. The fact that DA is more useful as the amount of available data lowers is in line with what has been observed for other tasks, as well as in multiclass setups, albeit with a much lower number of examples per class (Anaby-Tavor et al., 2020; Kumar et al., 2020). In the setup where 2,000 gold examples are available, there are very few significant improvements in performance when using generative data augmentation.

---

[8] This threshold has a Type I error rate comparable to that of a p-value threshold of 0.05 (Ulmer et al., 2022).

Gold data: 2,000 examples		Test			
Train: **AGREEMENT**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.770 (0.708)	**0.900** (0.861)	0.568 (0.580)	0.895 (0.840)
Oversampling		0.761 (0.684)	0.894 (0.848)	0.592 (0.580)*	0.877 (0.830)
	tag-prompt	**0.773** (0.714)	**0.900** (0.868)	0.582 (0.563)*	0.890 (0.840)
Filtering:	nl-prompt	0.771 (0.712)	**0.900** (0.867)	0.576 (0.555)	0.895 (0.850)
$p > 0.5$	cloze-prompt	0.771 (0.713)	**0.900** (0.868)	0.576 (0.555)	**0.896** (0.850)
	1/label	0.769 (0.712)	0.893 (0.861)	0.594 (0.585)*	0.885 (0.837)
	tag-prompt	0.766 (0.708)	0.895 (0.861)	0.590 (0.580)*	0.887 (0.840)
Filtering:	nl-prompt	0.771 (0.714)	0.898 (0.866)	0.586 (0.572)*	0.892 (0.847)
$p > 0.7$	cloze-prompt	0.769 (0.712)	0.897 (0.864)	0.586 (0.570)*	0.891 (0.846)
	1/label	0.768 (0.713)	0.894 (0.862)	**0.596** (0.586)*	0.886 (0.838)
Train: **FOUNTA**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.635 (0.619)	0.910 (0.883)	0.611 (0.612)	0.904 (0.866)
Oversampling		0.628 (0.604)	0.907 (0.883)	0.615 (0.618)	0.901 (0.859)
	tag-prompt	0.645 (0.620)	0.911 (0.883)	0.614 (0.618)	0.901 (0.863)
Filtering:	nl-prompt	0.635 (0.616)	0.911 (0.885)	0.625 (0.633)	0.905 (0.868)
$p > 0.5$	cloze-prompt	0.644 (0.619)	**0.915** (0.888)	0.607 (0.607)	0.906 (0.870)
	1/label	0.633 (0.613)	0.910 (0.881)	0.612 (0.615)	0.902 (0.864)
	tag-prompt	**0.650** (0.623)	0.913 (0.885)	0.619 (0.624)	0.903 (0.865)
Filtering:	nl-prompt	0.645 (0.619)	0.914 (0.887)	0.615 (0.617)	**0.908** (0.872)
$p > 0.7$	cloze-prompt	0.640 (0.619)	0.913 (0.885)	**0.621** (0.625)	0.904 (0.866)
	1/label	0.647 (0.619)	0.914 (0.886)	0.612 (0.614)	0.907 (0.871)
Train: **SBIC**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.608 (0.555)	**0.737** (0.618)	0.813 (0.844)	0.804 (0.712)
Oversampling		0.591 (0.526)	0.722 (0.590)	0.810 (0.829)	0.789 (0.683)
	tag-prompt	0.603 (0.550)	0.725 (0.597)	0.812 (0.840)	0.803 (0.708)
Filtering:	nl-prompt	0.604 (0.547)	0.730 (0.605)	**0.814** (0.844)	0.802 (0.708)
$p > 0.5$	cloze-prompt	0.608 (0.552)	0.729 (0.607)	**0.814** (0.844)	0.806 (0.714)
	1/label	0.606 (0.548)	0.725 (0.598)	0.811 (0.840)	0.800 (0.704)
	tag-prompt	0.608 (0.560)	0.733 (0.611)	0.811 (0.841)	**0.807** (0.716)
Filtering:	nl-prompt	**0.618** (0.546)	0.724 (0.593)	**0.814** (0.842)	0.801 (0.703)
$p > 0.7$	cloze-prompt	0.611 (0.555)	0.735 (0.615)	0.813 (0.844)	**0.807** (0.714)
	1/label	0.609 (0.558)	0.733 (0.612)	**0.814** (0.844)	0.804 (0.709)
Train: **OLID**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.584 (0.599)	0.874 (0.841)	0.633 (0.668)	**0.897** (0.859)
Oversampling		0.576 (0.580)	0.858 (0.824)	0.637 (0.709)	0.887 (0.845)
	tag-prompt	0.570 (0.593)	0.867 (0.832)	0.636 (0.681)	0.891 (0.852)
Filtering:	nl-prompt	0.586 (0.598)	0.875 (0.841)	0.641 (0.681)	0.895 (0.856)
$p > 0.5$	cloze-prompt	**0.592** (0.603)	**0.878** (0.845)	0.638 (0.672)	**0.897** (0.861)
	1/label	0.573 (0.594)	0.871 (0.839)	**0.644** (0.687)	0.892 (0.855)
	tag-prompt	0.578 (0.597)	0.864 (0.831)	0.634 (0.675)	0.892 (0.853)
Filtering:	nl-prompt	0.581 (0.597)	0.873 (0.841)	0.642 (0.681)	0.896 (0.858)
$p > 0.7$	cloze-prompt	0.582 (0.597)	0.871 (0.839)	0.638 (0.676)	0.895 (0.857)
	1/label	0.579 (0.597)	0.872 (0.839)	0.643 (0.684)	0.895 (0.858)

Table 2: Average macro-F1 scores (over 10 runs) obtained by RoBERTa-base fine-tuned on augmented data, starting with 2,000 gold examples. F1 scores for the minority class are in parentheses. Grey cells contain within-dataset results, while the others contain cross-dataset results. Asterisks denote statistically significant results (compared to no augmentation). The best result for each train-test dataset combination is in bold.

Impact of prompting and filtering Interestingly, no prompting type seems to clearly outperform the others across setups. For instance, augmenting the Agreement **[AG]** dataset starting with 500 gold examples has a positive effect on performance across all prompting types both when tested on the in-domain test data and when tested on **[FO]** and **[SO]**, while when tested on **[SB]** none of the setups lead to significant improvements in performance. This seems to indicate that dataset characteristics have a greater impact than the prompting setup on whether generative DA can be effective. However, looking at Table 2, the situation is reversed: the model trained on **[AG]** only significantly benefits from data augmentation when tested on **[SB]** across most setups. A filtering threshold of 0.7 does seem to help improve performance at least marginally, but only on this dataset combination out of all the ones we tested. Overall, it appears that whether DA will have a positive impact on classification might not depend on the generation setup in our case.

Overall findings The most important pattern that emerges from our results is that generative DA using GPT-2 does not appear to reliably improve model performance, both in and out of domain. It apparently *can* significantly improve model performance, especially for some dataset combinations and with very low amounts of data. However, this improvement is not consistent, so based on our results we would advise against considering this type of DA a reliable method for improving offensive language classifiers in similar setups.

Another important aspect that emerges from our results is that oversampling is a very strong baseline, especially for the setup with 500 available annotated examples, even though it is often overlooked. To our knowledge, it was used as a baseline only in Juuti et al. (2020) for generative DA on this task, while most other works report the performance on augmented data only. Interestingly, oversampling does not only improve within-dataset performance, but it also has a significant positive impact on cross-dataset performance. Since it requires a fraction of the computational resources needed for generative DA, it may be preferable when $\sim$500 gold examples are available. We hypothesize that one of the reasons why oversampling can perform well is that at least a subset of the datasets share superficial features that might be amplified in the oversampling process, such as specific terms that are associated with offensiveness across datasets.

In general, although it does not reliably improve model performance, generative DA does not seem to significantly decrease performance either. Wullach et al. (2021) believe that generative DA could improve lexical diversity, leading to better generalization. In Section 7.1, we analyze the generated data to assess whether it could lead to benefits with regards to fairness, perhaps due to more representation of minorities given the higher lexical variety.

The results for BERT-based models are in general in line with those for RoBERTa-based models, although BERT-based models tend to perform worse regardless of setup. Again, with BERT models, oversampling seems to be just as reliable to improve both within-dataset and cross-dataset performance. Since the overall findings are similar to those of RoBERTa-based models, we do not report BERT results in this section, but in Appendix A.

7 Qualitative Analysis

In order to estimate the quality of the generated examples and the impact of the prompting method, we randomly select a subset of 10 generated examples for every dataset / data size combination for manual analysis. We find that there are some clear differences between the prompting setups, and that the methods that exploit prompting in natural language (*nl-prompt* and *cloze-prompt*) tend to generate the most realistic examples. *Tag-prompt* tends to often generate strings of random special characters, resulting in very low quality data, while the *1/label* setup often results in sequences that appear out of domain. Some examples of the generated texts can be found in Appendix B.

7.1 Lexical Artifacts Analysis

To investigate the lexical variation between the gold data and the generated data, we use pointwise mutual information (PMI), following Ramponi and Tonelli (2022). In particular, we analyze the most informative tokens for the *offensive* class in each dataset, looking at how certain tokens become less or more informative in the generated data.

The first tendency that can be noticed when looking at how the ranking of tokens' informativeness changes between gold and generated data is that for some of the datasets the changes are more evident (i.e. for **[AG]** and **[SB]**). For example, in the gold **[SB]** data, the word *fucking* is ranked as the 10,203rd most informative word for the *offensive* class. In data augmented using the *tag-prompt* type on the generative model trained on 2,000 instances, however, the same word is ranked 4th. This means that the model has generated a very large amount of offensive messages containing this word, while it was not prominent in the gold data it was fine-tuned on. This happens for both the setup starting with 500 gold examples and with 2,000.

While the prominence of a potentially predictive word for offensive language, like *fucking*, is a good sign of the quality of the generated data, since it means that the generation process can usefully augment existing data, this happens also with tokens that should not be predictive for this task, defined by Ramponi and Tonelli (2022) as *spurious lexical artifacts*. For example, across several generation setups, we can notice the increased importance for the offensive label of identity terms, such as *girl*, *gay*, *woman*, *jews*, but also words that hold no inherent association with abusive language, such as *people* or *difference*. The higher importance of some identity terms could be associated with biases that were already present in the pre-trained generative model, and which were amplified during the generation process. In order to better understand the impact of these differences in token informativeness, we turn to the HateCheck functional tests in Section 7.2. For further details, the full lists of top 11 informative tokens for each dataset, both gold and generated, is reported in Appendix C.

7.2 HateCheck Analysis

We explore some of the weaknesses of classifiers trained on gold and augmented data through the HateCheck suite (Röttger et al., 2021), consisting of a series of functional tests aimed at exploring the weak points of hate speech models.[9]

Overall, data augmentation has little impact on the performance of models on HateCheck, especially on models trained starting with 2,000 gold instances. Similarly to the results reported in Section 6, oversampling tends to lead to similar or better improvements than generative DA, again depending on the starting dataset. However, this improvement is focused only on some functional tests, leading to worse performance on other tests.

For instance, two interesting tests are those on *non-hateful slurs*, including test 8, non-hateful homonyms of slurs (e.g. “Female canines are called bitches”), and test 9, *reclaimed slurs* (e.g. “The Q in LGBTQ stands for queer”). For models trained on 500 gold instances, the performance consistently drops on these two tests for all augmentation setups and oversampling across all datasets. This suggests that models trained using oversampling or generative data augmentation may start overfitting and relying on the presence of specific terms such as slurs for the classification of offensive language. For models trained on 2,000 gold instances, the performance on these two tests is somewhat stable, with no improvements or decreases in performance.

Another interesting test for examining whether generative DA increases the reliance of classifiers on specific words is test 15, *negated hateful statement*, e.g. “I would never hurt any woman”. The performance for this test tends to be overall stable or lower than the baseline, mostly decreasing with oversampling and in the setup starting from 500 annotated instances. This again indicates that models trained on augmented data in the setups we used show potentially harmful weaknesses, since they tend to misclassify sequences based on simple patterns such as the presence of specific words.

While models trained on augmented data tend to perform worse on non-hateful sequences containing identity terms and slurs, they do show improvements for those tests that benefit from being able to find these terms, such as test 7, *hate expressed using slur*, or test 10, *hate expressed using profanity*, further confirming that augmentation tends to steer models into overfitting identity terms and slurs.

Further details on the performances of models on each HateCheck test and on the targets contained in the tests are found in Appendix D.

8 Conclusions

In this work, we presented an evaluation of both existing and novel data augmentation setups based on generative large language models for offensive language detection. We investigated the robustness of such models, testing them in within-dataset and cross-dataset scenarios, and performed a qualitative analysis on the augmented data.

We found that while generative DA can positively impact model performance in some cases, especially when low amounts of gold data are available, this positive effect is not consistent across setups, making generative DA unreliable in the setups we tested. In addition to this, we found that generative DA can potentially introduce lexical bias from the pre-trained generative model into the augmented data, as well as increase the reliance of models on identity terms and slurs, which could have unintended effects on classification.

Overall, although it might improve classification performance in some cases, we advise against using generative DA for this task, since it is computationally intensive and it does not appear to consistently make models perform better or be more robust.

---

[9] Since our models are aimed at detecting offensive language in general and HateCheck is focused on hate speech, a narrower phenomenon, not all tests are entirely informative in our case, such as test number 11, testing the performance on non-hateful profanities. In general, however, the labels of HateCheck tests are aligned with our task.

Limitations

The main limitation of this work is its focus on English, leaving out other languages that may benefit more from a thorough evaluation of data augmentation methods because they have fewer resources. We selected English for this work because it allowed us to evaluate the system performance on four datasets with different characteristics and a number of configurations, thanks also to the availability of language-specific GPT2, BERT and RoBERTa. We are aware that generation-based data augmentation would be potentially more useful in real low-resource scenarios, and we plan on investigating in the future whether our findings hold for other languages.

Furthermore, this work deals only with one type of data augmentation, i.e. the one using fine-tuned generators, while there are others that we left out because of space limits and that may be investigated in the future. For example, we might compare generative DA with rule-based augmentation, synonym-based approaches and backtranslation, among others, and investigate whether different data augmentation approaches present differences in terms of robustness or lexical biases. In addition, we only experiment with one generative model, while we could in the future compare generative DA using GPT-2 with other kinds of generative models, especially more advanced ones.

Another aspect left unexplored in our work is whether this type of data augmentation could help models dedicated to this task that are widely available, such as HateBERT (Caselli et al., 2021) or ToxiGen-RoBERTa (Hartvigsen et al., 2022b). While in this work we focus on scenarios in which little data is available, experimenting with these models could yield interesting results in future work with a broader scope.

Ethics Statement

In our experiments, we compare different setups in which large language models are exploited in order to artificially create more data to train models aimed at detecting offensive language. In this case, synthetic data has two main potential advantages: first, it limits the amount of data gathered from online spaces without user consent, and second, it reduces the amount of manual annotation required to create labeled datasets for offensive language detection, which can have a negative psychological impact on annotators (Riedl et al., 2020).

While using generative models to augment data can in some cases be beneficial for classification performance, the sequences generated by these models can exhibit unpredictable characteristics that exacerbate existing bias or produce new forms of it. Given that the improvements provided by generative DA are inconsistent, there are no clear advantages to this method when considering its potential risks. As a consequence, we advise against deploying models trained on generated data in practice.

Since the main contribution of our work is not a novel model or algorithm, but rather an evaluation of different approaches for generative data augmentation, we share as much information as we can for the reproducibility of our experiments, but we choose not to release the code openly to the public, in order to limit potential misuse of models that can generate offensive language. We also choose not to publicly release the generated data for various reasons. As shown by our results, the generated examples are not reliable for improving existing systems, so their utility is limited. Furthermore, the generated texts are not curated, which could result in including personal user information or harmful statements targeting specific individuals being generated. We will, however, share the data upon request to other interested researchers.

Acknowledgements

This work has been partly supported by the PROTECTOR European project (ISFP-2020-AG-PROTECT) on "Protecting places of worship" (101034216) and the STAND BY ME 2.0 project (CERV-2021-DAPHNE) on "Stop gender-based violence by addressing masculinities and changing behaviour of young people through human rights education" (101049386).

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do Not Have Enough Data? Deep Learning to the Rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Trans-*

parency, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Eustasio Del Barrio, Juan A Cuesta-Albertos, and Carlos Matrán. 2018. An optimal transportation approach for assessing almost stochastic order. In *The Mathematics of the Uncertain*, pages 33–44. Springer.

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

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep dominance - how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2785, Florence, Italy. Association for Computational Linguistics.

Ashwin Geet D'Sa, Irina Illina, Dominique Fohr, Dietrich Klakow, and Dana Ruiter. 2021. Exploring Conditional Language Model Based Data Augmentation Approaches for Hate Speech Classification. In *Text, Speech, and Dialogue: 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6–9, 2021, Proceedings*, pages 135–146, Berlin, Heidelberg. Springer-Verlag.

Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. RealToxicityPrompts: Evaluating Neural Toxic Degeneration in Language Models. pages 3356–3369.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022a. ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection. *arXiv:2203.09509 [cs]*.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022b. ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326, Dublin, Ireland. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The Curious Case of Neural Text Degeneration. In *arXiv:1904.09751 [Cs]*.

Mika Juuti, Tommi Gröndahl, Adrian Flanagan, and N. Asokan. 2020. A little goes a long way: Improving toxic language classification despite data scarcity. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2991–3009, Online. Association for Computational Linguistics.

Filip Klubicka and Raquel Fernández. 2018. Examining a hate speech corpus for hate speech detection and popularity prediction. In *Proceedings of 4REAL Workshop - Workshop on Replicability and Reproducibility of Research Results in Science and Technology of Language*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data Augmentation using Pre-trained Transformer Models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Elisa Leonardelli, Stefano Menini, Alessio Palmero Aprosio, Marco Guerini, and Sara Tonelli. 2021. Agreeing to disagree: Annotating offensive language datasets with annotators' disagreement. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10528–10539, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. WANLI: Worker and AI Collaboration for Natural Language Inference Dataset Creation. *arXiv:2201.05955 [cs]*.

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020. Data Boost: Text Data Augmentation Through Reinforcement Learning Guided Conditional Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9031–9041, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Alan Ramponi and Sara Tonelli. 2022. Features or spurious artifacts? data-centric baselines for fair and robust hate speech detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3027–3040, Seattle, United States. Association for Computational Linguistics.

Martin J Riedl, Gina M Masullo, and Kelsey N Whipple. 2020. The downsides of digital labor: Exploring the toll incivility takes on online comment moderators. *Computers in Human Behavior*, 107:106262.

Paul Röttger, Bertie Vidgen, Dong Nguyen, Zeerak Waseem, Helen Margetts, and Janet Pierrehumbert. 2021. HateCheck: Functional tests for hate speech detection models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 41–58, Online. Association for Computational Linguistics.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social Bias Frames: Reasoning about Social and Power Implications of Language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Naama Tepper, Esther Goldbraich, Naama Zwerdling, George Kour, Ateret Anaby Tavor, and Boaz Carmeli. 2020. Balancing via Generation for Multi-Class Text Classification Improvement. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1440–1452, Online. Association for Computational Linguistics.

Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. 2022. deep-significance-easy and meaningful statistical significance testing in the age of neural networks. *arXiv preprint arXiv:2204.06815*.

Bertie Vidgen and Leon Derczynski. 2020. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *PLOS ONE*, 15(12):e0243300.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tomer Wullach, Amir Adler, and Einat Minkov. 2021. Fight fire with fire: Fine-tuning hate detectors using large samples of generated hate speech. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4699–4705, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447, Barcelona (online). International Committee for Computational Linguistics.

Xuhui Zhou, Maarten Sap, Swabha Swayamdipta, Yejin Choi, and Noah Smith. 2021. Challenges in automated debiasing for toxic language detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3143–3155, Online. Association for Computational Linguistics.

A BERT-Based Models Results

In this section, we present the results of BERT-based models in the setup where we start with 500 gold examples in Table 3 and with 2,000 gold examples in Table 4.

Gold data: 500 examples		Test			
Train: **AGREEMENT**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.630	0.716	0.550	0.700
Oversampling		**0.696***	**0.823***	0.573	**0.825***
	tag-prompt	0.663	0.775*	0.562	0.774
Filtering:	nl-prompt	0.654	0.752	**0.584**	0.767
$p > 0.5$	cloze-prompt	0.665*	0.773*	0.554	0.780*
	1/label	0.688*	0.798*	0.575	0.797*
Train: **FOUNTA**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.619	0.890	0.613	0.847
Oversampling		**0.638**	**0.906***	0.598	**0.885***
	tag-prompt	0.636	0.904	0.600	0.876*
Filtering:	nl-prompt	0.614	0.900	**0.641**	0.874*
$p > 0.5$	cloze-prompt	0.632	0.900	0.606	0.857
	1/label	0.629	0.899	0.633	0.878*
Train: **SBIC**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.566	0.629	0.747	0.727
Oversampling		**0.579***	**0.682**	**0.766***	**0.756**
	tag-prompt	0.575	0.679	0.754	0.755
Filtering:	nl-prompt	0.576	0.677	0.757	0.754
$p > 0.5$	cloze-prompt	0.566	0.656	0.754	0.738
	1/label	0.574	0.664	0.762*	0.743
Train: **OLID**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.555	0.757	0.635	0.770
Oversampling		0.555	**0.832***	0.653	**0.852***
	tag-prompt	0.554	0.795	0.641	0.813
Filtering:	nl-prompt	0.559	0.810*	**0.658***	0.832*
$p > 0.5$	cloze-prompt	**0.562**	0.803*	0.648	0.823
	1/label	0.537	0.805*	0.648	0.821*

Table 3: Average macro-F1 scores (over 10 runs) obtained by BERT-base-uncased fine-tuned on augmented data, starting with 500 gold examples. Grey cells contain within-dataset results, while the others contain cross-dataset results. Asterisks denote statistically significant results (compared to no augmentation). The best result for each train-test dataset combination is in bold.

Gold data: 2,000 examples		Test			
Train: **AGREEMENT**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.756	0.894	0.573	0.891
Oversampling		0.746	0.884	0.592	0.880
	tag-prompt	0.759	0.900	0.567	0.893
Filtering:	nl-prompt	0.761	0.901*	0.567	**0.900***
$p > 0.5$	cloze-prompt	0.756	0.901*	0.572	0.899*
	1/label	0.749	0.892	0.584	0.891
	tag-prompt	0.760	0.899	0.578	0.893
Filtering:	nl-prompt	0.760	0.899	0.572	0.897*
$p > 0.7$	cloze-prompt	**0.762**	**0.902***	0.572	0.898*
	1/label	0.753	0.897	**0.593***	0.893
Train: **FOUNTA**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.616	0.913	0.628	0.905
Oversampling		0.635*	0.911	0.617	0.899
	tag-prompt	0.634*	0.914	0.621	0.905
Filtering:	nl-prompt	0.632	0.914	0.627	0.905
$p > 0.5$	cloze-prompt	0.617	0.914	**0.630**	0.903
	1/label	0.629	0.914	0.628	0.904
	tag-prompt	**0.636***	0.913	0.624	0.905
Filtering:	nl-prompt	0.634	**0.915**	0.629	0.904
$p > 0.7$	cloze-prompt	0.633	0.914	**0.630**	**0.907**
	1/label	0.629	0.913	0.627	0.903
Train: **SBIC**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.589	**0.743**	0.806	0.807
Oversampling		0.588	0.716	0.799	0.786
	tag-prompt	0.584	0.742	0.806	**0.809**
Filtering:	nl-prompt	0.594	0.734	0.807	0.802
$p > 0.5$	cloze-prompt	0.593	0.735	0.806	0.802
	1/label	0.586	0.739	0.804	0.800
	tag-prompt	0.582	**0.743**	**0.809**	0.806
Filtering:	nl-prompt	0.588	0.734	0.806	0.803
$p > 0.7$	cloze-prompt	**0.598**	0.742	0.807	0.807
	1/label	0.591	0.732	0.806	0.803
Train: **OLID**		**AG**	**FO**	**SB**	**SO**
No augmentation		0.562	0.874	0.653	0.897
Oversampling		0.549	0.859	0.661	0.885
	tag-prompt	0.555	0.862	0.662	0.890
Filtering:	nl-prompt	0.553	0.868	0.668*	0.893
$p > 0.5$	cloze-prompt	0.568	0.875	0.659	0.897
	1/label	0.541	0.858	0.669*	0.885
	tag-prompt	0.555	0.862	0.663	0.892
Filtering:	nl-prompt	0.561	0.873	0.665*	0.896
$p > 0.7$	cloze-prompt	**0.575**	**0.879**	0.658	**0.898**
	1/label	0.548	0.863	**0.671**	0.889

Table 4: Average macro-F1 scores (over 10 runs) obtained by BERT-base-uncased fine-tuned on augmented data, starting with 2,000 gold examples. Grey cells contain within-dataset results, while the others contain cross-dataset results. Asterisks denote statistically significant results (compared to no augmentation). The best result for each train-test dataset combination is in bold.

prompting	dataset	generated text	label
tag-prompt	[AG]	#SEP #BlackLivesMatter #BIDEN #MAGA URL	0
	[FO]	∼∼∼@USER	0
	[SB]	∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼∼ [...]	0
	[OL]	***********@USER You’re fucking awesome	0
nl-prompt	[AG]	@USER It is time to end this corrupt criminal republic #DemocRATting #BlackLivesMatter URL	0
	[FO]	??????????????????????????????????	1
	[SB]	I hate all of you who call yourself patriots. You are useless.	1
	[OL]	@USER @USER you are not a Christian..you are a hypocrite	1
cloze-prompt	[AG]	”I love how the media is pushing the idea that the protesters are racist	0
	[FO]	USER: hey, u should really try and figure out what is going on in the world of 4chan	0
	[SB]	If you can’t get a girl to like you, you’re a loser...	1
	[OL]	@USER @USER He is the worst	0
1/label	[AG]	Coronavirus was first identified in 1984 in the U.S. as a contaminant of the popular oral flu vaccine. Because of the severity of this illness, the CDC in the early 90s ordered the removal of all Coronaviruses from vaccines and in 1998 [...]	0
	[FO]	#6a8a2a color RGB value is (135,180,225). #6b98f6 color hex is (130,180,.25). the number is on its way to being determined so I don’t need to change it [...]	0
	[SB]	"The best way to get ahead in life is to keep learning" - Steve Jobs "As a scientist, the thing I am most excited about is the possibility that we will be able to engineer a cure for cancer - by using a synthetic biology-based approach" - Dr. Aubrey de Grey, M.D.[...]	0
	[OL]	EVERYBODY, EVERYBODY is talking about the fact that President Trump is being sued by the media. Yes, you read that correctly. Media. Lawsuit. #MAGA #Trump #ToriesOut #ToryOut’sOut#POTUSOut #MAGADump #MAGAMAG #MAGAZINEAU’S MAGA NEWSLETTER @USER @USER The way he [...]	0

Table 5: Examples of sequences generated by GPT-2 large models trained on 500 gold examples for every combination of prompting and dataset.

B Examples of Generated Texts

Examples of sequences generated by the models can be found in Table 5 for generative models trained on 500 labeled instances and in Table 6 for models trained on 2,000 labeled instances. Some texts have been truncated (signaled by “[...]”) due to space constraints. The labels reported in Tables 5 and 6 are those assigned by the RoBERTa baseline.

prompting	**dataset**	**generated text**	**label**
tag-prompt	[AG]	URL #Elections2020	0
	[FO]	!!!! 🙄🎣&> I fucking hate you URL	1
	[SB]	phone 7, apple iphones 8, iphONES 8, android ips 8, blackberry ips, ixp8 iph, [...]	0
	[OL]	URL You guys are pathetic	1
nl-prompt	[AG]	@USER Because the Democrats lie, cheat, steal and intentionally create the chaos in our country. They are beyond evil!	0
	[FO]	????????? @USER We are listening..if you have any ideas please send us a tweet at URL	0
	[SB]	You're never too young to dream, you just gotta work it.	0
	[OL]	You and your friends are fucking delusional. There are far worse far right wing nut jobs out there than antifa. Try to keep your stupid comment to yourself	1
cloze-prompt	[AG]	The most disgusting thing about this is that the person spreading this filth is a member of the media. They are enabling the filth.	0
	[FO]	@USER @USER Yeah, that would be a pretty crazy idea! I'm sure that would cause an uproar and a few angry people would object	1
	[SB]	Do you think these fake news articles on vaccines and autism are getting too much play on social media?	0
	[OL]	@USER @USER @USER @USER He is just one more nut job in Congress.	1
1/label	[AG]	The family of a missing teen has filed a $5 million lawsuit against the city of Madison, alleging that officials didn't do enough to find their son when he went missing in September 2010. [...]	0
	[FO]	$4.99 Featuring all the fan favorites including: • Exclusive and classic illustrations from some of comics' most beloved artists [..]	0
	[SB]	Beef and broccoli are the perfect meal for when you are hungry and not hungry. But I think they would probably turn to dust in an instant if you tried to eat them.Blessings are spread in the name of the Lord, and His name is exalted. [...]	0
	[OL]	WWF @USER We are all outraged that this clown is using his access to the office to threaten & bully. You &amp#Array; you &amp"#Array"; yourself should be ashamed of yourself.@USER you are so right on this issue but I have to agree that we conservatives are becoming too emotional. [...]	1

Table 6: Examples of sequences generated by GPT-2 large models trained on 2,000 gold examples for every combination of prompting and dataset.

C Lexical Artifacts

In this section, we present the lists of top-11 informative tokens for the offensive class, both on gold and on generated data. Lists for data in the setup where we start with 500 annotated instances can be found in Table 7, and those for the setup with 2,000 gold instances are in Table 8.

⚠ **Content warning**: *Tables 7 and 8 contain uncensored profanities and slurs.* [10]

D HateCheck

Table 9 and Table 10 present the results on HateCheck tests and targets for models in the 500 gold examples setup. Table 11 and Table 12 present the results on the functional tests and targets for models in the setup in which we start with 2,000 annotated examples.

---

[10] These are left uncensored for increased readability, since special characters are already used to signal boundaries of sub-word tokens, and to avoid confusion with words that are self-censored by the users.

AGREEMENT									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	fuck	0	fuck	2	fucking	2	fucking	0	fuck
1	shit	2	fucking	0	fuck	1	shit	2	fucking
2	fucking	1	shit	1	shit	0	fuck	1	shit
3	ass	31	racist	5	dumb	6	stupid	6	stupid
4	idiot	5	dumb	6	stupid	31	racist	3	ass
5	dumb	6	stupid	31	racist	5	dumb	7	##s
6	stupid	3	ass	7	##s	3	ass	11	guy
7	##s	25	mor	3	ass	302	disgusting	17	piece
8	bitch	13	trump	4	idiot	4	idiot	5	dumb
9	##er	4	idiot	302	disgusting	18	user	4	idiot
10	bullshit	423	people	17	piece	25	mor	31	racist

FOUNTA									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	fucking	0	fucking	0	fucking	0	fucking	0	fucking
1	fucked	4	fuck	3	bitch	4	fuck	4	fuck
2	user	2	user	4	fuck	2	user	6	hate
3	bitch	6	hate	6	hate	6	hate	11	shit
4	fuck	3	bitch	11	shit	3	bitch	1	fucked
5	ass	5	ass	10	stupid	5	ass	10	stupid
6	hate	1	fucked	8	idiot	1	fucked	3	bitch
7	128	11	shit	5	ass	10	stupid	5	ass
8	idiot	10	stupid	1	fucked	11	shit	8	idiot
9	##gga	8	idiot	43	sick	8	idiot	43	sick
10	stupid	43	sick	41	##tar	43	sick	34	kill

SBIC									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	black	0	black	0	black	0	black	0	black
1	bitch	4	white	4	white	3	difference	29	woman
2	##es	9264	[SEP]	38	people	12	girl	5	sex
3	difference	38	people	3	difference	4	white	8	women
4	white	11	##s	12	girl	29	woman	38	people
5	sex	3	difference	31	person	5	sex	4	white
6	ho	5382	fucking	29	woman	80	guy	12	girl
7	##gga	31	person	17	##gger	31	person	57	racist
8	women	29	woman	5382	fucking	38	people	14	gay
9	jew	8	women	7	##gga	8	women	80	guy
10	fuck	10	fuck	8	women	5382	fucking	44	kill

OLID									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	shit	0	shit	0	shit	19	disgusting	0	shit
1	fuck	16	people	6	liberals	6	liberals	7	stupid
2	ass	19	disgusting	1	fuck	16	people	1	fuck
3	fucking	6	liberals	19	disgusting	0	shit	3	fucking
4	##s	28	hate	52	sick	7	stupid	16	people
5	bitch	18	racist	7	stupid	9	idiot	52	sick
6	liberals	7	stupid	3	fucking	28	hate	19	disgusting
7	stupid	52	sick	16	people	22	liar	99	wrong
8	control	22	liar	28	hate	1	fuck	29	disgrace
9	idiot	1	fuck	9	idiot	18	racist	31	bad
10	dumb	10	dumb	22	liar	10	dumb	97	women

Table 7: Top tokens for the *offensive* class in the gold data and in the generated data when starting with 500 examples, computed using the PMI implementation of Ramponi and Tonelli (2022). The indices refer to the ranking of importance of the tokens in the gold data, while the order of the tokens reflect their informativeness for the offensive class in the generated data.

AGREEMENT									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	fuck	0	fuck	18	##ass	2	fucking	0	fuck
1	shit	2	fucking	52	##est	0	fuck	1	shit
2	fucking	1	shit	16	##on	1	shit	2	fucking
3	ass	23	racist	418	##path	950	user	14	mag
4	##s	7	dumb	4	##s	6	idiot	23	racist
5	stupid	6	idiot	3	ass	5	stupid	6	idiot
6	idiot	89	liar	10	asshole	23	racist	22	##a
7	dumb	5	stupid	12	bitch	7	dumb	5	stupid
8	piece	135	##trum	9	bullshit	89	liar	7	dumb
9	bullshit	14	mag	105	bunch	8	piece	8	piece
10	asshole	1284	##p	1049	complete	3	ass	13	guy
FOUNTA									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	fucking	0	fucking	0	fucking	0	fucking	0	fucking
1	fucked	6	hate	1	fucked	1	fucked	4	fuck
2	user	1	fucked	4	fuck	2	user	1	fucked
3	bitch	4	fuck	3	bitch	6	hate	6	hate
4	fuck	3	bitch	9	shit	4	fuck	3	bitch
5	ass	16339	[SEP]	6	hate	3	bitch	5	ass
6	hate	5	ass	5	ass	5	ass	9	shit
7	##gga	8	shit	11	stupid	10	idiot	11	stupid
8	128	7	##gga	20	sick	11	stupid	10	idiot
9	shit	2	user	7	##gga	9	shit	20	sick
10	idiot	11	stupid	19	mad	7	##gga	8	128
SBIC									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	black	0	black	0	black	0	black	0	black
1	bitch	15008	[SEP]	3	white	3	white	3	white
2	difference	10	##s	1	bitch	2	difference	5	sex
3	white	10203	fucking	14	##gger	15	woman	2	difference
4	##es	1	bitch	12	jews	8	women	8	women
5	sex	763	offensive	7	##gga	12	jews	15	woman
6	ho	3	white	2	difference	5	sex	9	fuck
7	##gga	5	sex	10	##s	11	jew	43	racist
8	women	9	fuck	11	jew	19	girl	16	##ist
9	fuck	43	racist	15	woman	1	bitch	1	bitch
10	##s	4	##es	8	women	14	##gger	11	jew
OLID									
gold data		**tag-prompt**		**nl-prompt**		**cloze-prompt**		**1/label**	
index	token	index	token	index	token	index	token	index	token
0	shit	11	liberals	11	liberals	11	liberals	0	shit
1	fuck	12	disgusting	1	fuck	0	shit	12	disgusting
2	ass	7	people	0	shit	12	disgusting	6	stupid
3	fucking	0	shit	12	disgusting	6	stupid	7	people
4	bitch	13	racist	53	disgrace	18	liar	1	fuck
5	##s	6	stupid	6	stupid	53	disgrace	14	sick
6	stupid	53	disgrace	14	sick	26	##yp	13	racist
7	people	26	##yp	18	liar	14	sick	18	liar
8	idiot	29	##oc	3	fucking	29	##oc	3	fucking
9	dumb	14	sick	7	people	32	lying	26	##yp
10	user	16	fake	5	##s	1	fuck	29	##oc

Table 8: Top tokens for the *offensive* class in the gold data and in the generated data when starting with 2,000 examples, computed using the PMI implementation of Ramponi and Tonelli (2022). The indices refer to the ranking of importance of the tokens in the gold data, while the order of the tokens reflect their informativeness for the offensive class in the generated data.

		AGREEMENT						FOUNTA						SBIC						OLID					
	Functionality	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**
1	derog_neg_emote_h	0.22	0.19	0.25	0.26	0.22	0.17	0.53	0.79	0.73	0.77	0.61	0.69	0.71	0.80	0.74	0.71	0.79	0.82	0.77	0.82	0.82	0.87	0.82	0.93
2	derog_neg_attrib_h	0.44	0.56	0.55	0.50	0.50	0.52	0.74	0.91	0.83	0.89	0.81	0.86	0.89	0.87	0.85	0.83	0.90	0.92	0.82	0.96	0.88	0.87	0.86	0.97
3	derog_dehum_h	0.39	0.58	0.50	0.46	0.43	0.47	0.50	0.82	0.71	0.82	0.62	0.72	0.90	0.90	0.86	0.83	0.89	0.93	0.75	0.91	0.84	0.86	0.83	0.93
4	derog_impl_h	0.09	0.15	0.11	0.12	0.08	0.11	0.14	0.47	0.27	0.41	0.19	0.30	0.90	0.90	0.84	0.85	0.91	0.91	0.54	0.72	0.66	0.63	0.59	0.71
5	threat_dir_h	0.19	0.27	0.21	0.21	0.20	0.18	0.38	0.73	0.65	0.71	0.50	0.65	0.75	0.90	0.77	0.71	0.82	0.84	0.61	0.81	0.74	0.75	0.69	0.82
6	threat_norm_h	0.18	0.22	0.16	0.20	0.19	0.17	0.31	0.69	0.56	0.65	0.40	0.54	0.87	0.93	0.82	0.84	0.89	0.92	0.63	0.84	0.73	0.74	0.69	0.81
7	slur_h	0.60	0.74	0.65	0.65	0.65	0.68	0.57	0.74	0.66	0.70	0.69	0.73	0.77	0.82	0.76	0.79	0.79	0.82	0.69	0.87	0.79	0.80	0.77	0.84
8	slur_homonym_nh	0.74	0.55	0.60	0.59	0.63	0.59	0.64	0.54	0.51	0.51	0.50	0.53	0.53	0.50	0.53	0.47	0.48	0.49	0.67	0.48	0.57	0.52	0.58	0.47
9	slur_reclaimed_nh	0.28	0.24	0.25	0.25	0.24	0.26	0.22	0.18	0.20	0.19	0.19	0.19	0.33	0.21	0.33	0.29	0.28	0.26	0.34	0.23	0.28	0.27	0.29	0.17
10	profanity_h	0.88	0.94	0.93	0.93	0.93	0.92	1.00	0.99	1.00	1.00	1.00	1.00	0.90	0.90	0.84	0.85	0.93	0.92	0.89	0.99	0.90	0.90	0.91	1.00
11	profanity_nh	0.20	0.11	0.12	0.11	0.12	0.14	0.01	0.02	0.00	0.00	0.00	0.01	0.56	0.63	0.59	0.49	0.56	0.70	0.17	0.03	0.14	0.11	0.14	0.01
12	ref_subs_clause_h	0.34	0.39	0.41	0.46	0.43	0.38	0.49	0.74	0.68	0.73	0.61	0.69	0.85	0.92	0.79	0.80	0.88	0.90	0.79	0.88	0.85	0.86	0.81	0.92
13	ref_subs_sent_h	0.49	0.49	0.56	0.57	0.57	0.54	0.61	0.82	0.78	0.85	0.70	0.77	0.89	0.93	0.83	0.83	0.91	0.94	0.86	0.92	0.88	0.89	0.85	0.96
14	negate_pos_h	0.04	0.07	0.09	0.07	0.05	0.04	0.05	0.35	0.26	0.30	0.17	0.26	0.85	0.85	0.77	0.81	0.84	0.83	0.45	0.49	0.65	0.64	0.51	0.52
15	negate_neg_nh	0.90	0.87	0.84	0.88	0.87	0.87	0.76	0.34	0.46	0.42	0.63	0.52	0.17	0.12	0.21	0.19	0.13	0.13	0.43	0.30	0.33	0.31	0.44	0.28
16	phrase_question_h	0.40	0.34	0.36	0.42	0.41	0.38	0.57	0.80	0.72	0.83	0.62	0.73	0.99	0.94	0.98	0.99	0.99	0.97	0.83	0.91	0.83	0.85	0.83	0.96
17	phrase_opinion_h	0.37	0.51	0.45	0.47	0.44	0.46	0.56	0.80	0.72	0.80	0.66	0.72	0.81	0.91	0.78	0.78	0.88	0.91	0.78	0.94	0.84	0.85	0.82	0.93
18	ident_neutral_nh	0.98	0.98	0.99	0.98	0.99	0.98	1.00	0.88	0.96	0.92	0.95	0.94	0.37	0.36	0.44	0.35	0.40	0.39	0.96	0.88	0.92	0.94	0.88	0.86
19	ident_pos_nh	0.98	0.98	0.98	0.98	0.99	0.97	1.00	0.84	0.94	0.95	0.91	0.91	0.36	0.34	0.42	0.39	0.38	0.40	0.92	0.93	0.89	0.92	0.89	0.88
20	counter_quote_nh	0.42	0.52	0.47	0.42	0.50	0.45	0.24	0.11	0.17	0.08	0.24	0.16	0.21	0.13	0.19	0.19	0.12	0.14	0.14	0.04	0.12	0.12	0.13	0.02
21	counter_ref_nh	0.48	0.42	0.45	0.43	0.44	0.43	0.26	0.11	0.19	0.10	0.22	0.17	0.10	0.14	0.16	0.14	0.07	0.11	0.16	0.08	0.13	0.14	0.17	0.05
22	target_obj_nh	0.75	0.74	0.75	0.76	0.77	0.78	0.61	0.49	0.56	0.47	0.58	0.58	0.52	0.68	0.62	0.60	0.62	0.68	0.39	0.23	0.28	0.27	0.32	0.20
23	target_indiv_nh	0.62	0.58	0.55	0.60	0.60	0.61	0.42	0.21	0.22	0.14	0.33	0.28	0.52	0.44	0.52	0.44	0.49	0.55	0.33	0.12	0.22	0.18	0.26	0.09
24	target_group_nh	0.65	0.57	0.60	0.65	0.65	0.64	0.57	0.32	0.39	0.31	0.51	0.45	0.30	0.33	0.36	0.34	0.32	0.36	0.33	0.16	0.22	0.22	0.25	0.16
25	spell_char_swap_h	0.27	0.28	0.30	0.25	0.25	0.27	0.30	0.56	0.51	0.57	0.50	0.56	0.74	0.79	0.76	0.75	0.80	0.87	0.40	0.54	0.59	0.61	0.61	0.67

Table 9: Accuracy on the first 25 functional HateCheck tests of RoBERTa models trained on 500 gold examples and on the augmented data.

		AGREEMENT						FOUNTA						SBIC						OLID					
	Target	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**
1	women	0.44	0.49	0.47	0.47	0.45	0.46	0.50	0.58	0.56	0.59	0.53	0.56	0.50	0.53	0.50	0.48	0.52	0.58	0.57	0.64	0.61	0.62	0.60	0.65
2	trans people	0.43	0.45	0.44	0.45	0.44	0.41	0.50	0.56	0.56	0.58	0.52	0.57	0.48	0.51	0.49	0.46	0.51	0.52	0.56	0.63	0.61	0.61	0.58	0.64
3	gay people	0.50	0.53	0.51	0.52	0.50	0.53	0.56	0.57	0.60	0.63	0.56	0.59	0.48	0.50	0.48	0.46	0.50	0.51	0.59	0.65	0.62	0.63	0.61	0.65
4	black people	0.47	0.49	0.49	0.49	0.48	0.46	0.54	0.61	0.60	0.65	0.56	0.61	0.47	0.48	0.47	0.44	0.49	0.50	0.59	0.66	0.62	0.64	0.61	0.66
5	disabled people	0.40	0.44	0.44	0.44	0.44	0.40	0.45	0.54	0.54	0.57	0.51	0.52	0.50	0.53	0.51	0.49	0.52	0.55	0.55	0.63	0.60	0.61	0.60	0.64
6	Muslims	0.43	0.49	0.46	0.45	0.45	0.46	0.50	0.60	0.56	0.61	0.54	0.57	0.49	0.52	0.51	0.47	0.52	0.54	0.55	0.63	0.60	0.62	0.60	0.63
7	immigrants	0.39	0.45	0.43	0.42	0.42	0.42	0.44	0.54	0.52	0.54	0.50	0.51	0.51	0.55	0.53	0.49	0.53	0.55	0.53	0.62	0.59	0.60	0.58	0.62

Table 10: Accuracy on the different types of targets in the HateCheck tests of RoBERTa models trained on 500 gold examples and on the augmented data.

	AGREEMENT						FOUNTA						SBIC						OLID					
Functionality	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**
1 derog_neg_emote_h	0.07	0.20	0.11	0.07	0.07	0.12	0.91	0.89	0.89	0.91	0.91	0.89	0.87	0.88	0.88	0.89	0.91	0.79	0.93	0.89	0.91	0.95	0.86	0.94
2 derog_neg_attrib_h	0.51	0.57	0.57	0.55	0.55	0.56	0.94	0.94	0.94	0.95	0.95	0.95	0.95	0.98	0.96	0.98	0.98	0.95	0.99	0.98	1.00	0.99	0.99	0.98
3 derog_dehum_h	0.57	0.66	0.67	0.64	0.62	0.66	0.90	0.89	0.88	0.89	0.89	0.92	0.96	0.98	0.96	0.97	0.96	0.92	0.97	0.95	0.97	0.98	0.96	0.98
4 derog_impl_h	0.11	0.20	0.13	0.13	0.11	0.16	0.52	0.59	0.53	0.54	0.50	0.50	0.92	0.94	0.89	0.92	0.95	0.85	0.70	0.76	0.76	0.75	0.68	0.83
5 threat_dir_h	0.23	0.46	0.34	0.25	0.28	0.34	0.82	0.87	0.86	0.84	0.87	0.84	0.95	0.97	0.93	0.95	0.96	0.92	0.87	0.90	0.87	0.92	0.85	0.95
6 threat_norm_h	0.11	0.37	0.18	0.12	0.11	0.22	0.81	0.87	0.81	0.82	0.84	0.78	0.95	0.97	0.94	0.96	0.97	0.93	0.86	0.93	0.90	0.90	0.85	0.93
7 slur_h	0.76	0.76	0.79	0.77	0.76	0.79	0.77	0.77	0.76	0.79	0.77	0.79	0.85	0.88	0.84	0.85	0.85	0.81	0.88	0.91	0.90	0.90	0.86	0.92
8 slur_homonym_nh	0.50	0.50	0.50	0.50	0.50	0.50	0.46	0.49	0.46	0.47	0.47	0.45	0.51	0.47	0.48	0.43	0.43	0.48	0.49	0.46	0.42	0.45	0.48	0.44
9 slur_reclaimed_nh	0.22	0.23	0.22	0.22	0.22	0.23	0.14	0.14	0.14	0.14	0.15	0.13	0.21	0.24	0.19	0.15	0.17	0.23	0.19	0.18	0.19	0.18	0.19	0.18
10 profanity_h	0.95	0.95	0.92	0.93	0.94	0.93	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.98	0.99	0.98	0.99	0.96	1.00	1.00	1.00	1.00	1.00	1.00
11 profanity_nh	0.11	0.11	0.12	0.11	0.10	0.10	0.00	0.01	0.00	0.00	0.00	0.01	0.65	0.66	0.66	0.59	0.59	0.65	0.01	0.02	0.01	0.01	0.02	0.01
12 ref_subs_clause_h	0.36	0.46	0.39	0.40	0.38	0.45	0.88	0.87	0.87	0.87	0.90	0.87	0.96	0.98	0.96	0.98	0.97	0.94	0.95	0.95	0.94	0.97	0.93	0.96
13 ref_subs_sent_h	0.51	0.53	0.51	0.52	0.52	0.55	0.93	0.93	0.90	0.93	0.95	0.93	0.97	0.97	0.97	0.98	0.97	0.97	0.97	0.98	0.95	0.98	0.96	0.98
14 negate_pos_h	0.02	0.05	0.03	0.03	0.02	0.06	0.43	0.52	0.46	0.46	0.45	0.46	0.83	0.93	0.80	0.89	0.88	0.79	0.49	0.59	0.57	0.63	0.51	0.68
15 negate_neg_nh	0.86	0.78	0.86	0.85	0.88	0.83	0.22	0.19	0.21	0.20	0.21	0.25	0.09	0.08	0.13	0.08	0.06	0.12	0.22	0.22	0.21	0.22	0.25	0.17
16 phrase_question_h	0.32	0.40	0.33	0.33	0.32	0.35	0.87	0.89	0.89	0.90	0.91	0.88	0.95	0.96	0.97	0.97	0.96	0.98	0.94	0.95	0.95	0.96	0.91	0.97
17 phrase_opinion_h	0.49	0.57	0.51	0.50	0.49	0.54	0.89	0.90	0.87	0.88	0.90	0.85	0.96	0.98	0.94	0.97	0.98	0.95	0.95	0.97	0.95	0.96	0.93	0.98
18 ident_neutral_nh	0.99	0.96	0.97	0.98	0.99	0.99	0.82	0.78	0.75	0.85	0.84	0.82	0.52	0.36	0.45	0.34	0.33	0.50	0.86	0.77	0.80	0.80	0.82	0.71
19 ident_pos_nh	0.99	0.97	0.97	0.98	0.98	0.97	0.79	0.73	0.73	0.82	0.80	0.79	0.37	0.22	0.34	0.28	0.27	0.35	0.95	0.86	0.88	0.91	0.93	0.85
20 counter_quote_nh	0.48	0.48	0.47	0.47	0.50	0.45	0.08	0.08	0.12	0.09	0.06	0.07	0.05	0.06	0.06	0.05	0.05	0.06	0.01	0.02	0.01	0.00	0.01	0.01
21 counter_ref_nh	0.41	0.41	0.41	0.41	0.43	0.38	0.06	0.07	0.08	0.07	0.08	0.08	0.06	0.06	0.08	0.06	0.05	0.08	0.04	0.04	0.03	0.03	0.05	0.02
22 target_obj_nh	0.83	0.77	0.80	0.82	0.81	0.81	0.47	0.46	0.45	0.43	0.43	0.44	0.80	0.74	0.75	0.77	0.77	0.81	0.21	0.21	0.18	0.19	0.25	0.20
23 target_indiv_nh	0.60	0.48	0.51	0.57	0.55	0.50	0.13	0.13	0.14	0.13	0.13	0.16	0.45	0.38	0.39	0.33	0.37	0.47	0.09	0.06	0.08	0.06	0.10	0.06
24 target_group_nh	0.68	0.56	0.61	0.65	0.65	0.61	0.28	0.27	0.29	0.26	0.25	0.27	0.38	0.35	0.35	0.35	0.33	0.38	0.15	0.14	0.12	0.10	0.16	0.12
25 spell_char_swap_h	0.24	0.29	0.34	0.34	0.28	0.33	0.71	0.69	0.72	0.72	0.71	0.73	0.87	0.91	0.88	0.89	0.90	0.84	0.64	0.69	0.76	0.73	0.65	0.76

Table 11: Accuracy on the first 25 functional HateCheck tests of RoBERTa models trained on 2,000 gold examples and on the augmented data.

	AGREEMENT						FOUNTA						SBIC						OLID					
Target	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**	**base**	**overs**	**tag**	**nl**	**cloze**	**1/label**
1 women	0.47	0.51	0.50	0.48	0.47	0.51	0.62	0.62	0.61	0.62	0.61	0.61	0.59	0.57	0.59	0.61	0.58	0.58	0.65	0.66	0.67	0.68	0.64	0.68
2 trans people	0.42	0.46	0.45	0.45	0.44	0.44	0.59	0.59	0.57	0.61	0.60	0.61	0.55	0.53	0.54	0.51	0.53	0.53	0.63	0.63	0.65	0.66	0.62	0.66
3 gay people	0.50	0.55	0.53	0.51	0.51	0.53	0.58	0.57	0.55	0.61	0.57	0.59	0.52	0.51	0.52	0.48	0.48	0.51	0.63	0.60	0.59	0.63	0.62	0.56
4 black people	0.46	0.52	0.48	0.48	0.48	0.49	0.61	0.58	0.58	0.64	0.62	0.61	0.52	0.49	0.47	0.45	0.45	0.48	0.69	0.63	0.67	0.68	0.66	0.66
5 disabled people	0.43	0.45	0.45	0.45	0.44	0.44	0.57	0.57	0.57	0.59	0.59	0.59	0.57	0.56	0.59	0.56	0.56	0.57	0.66	0.66	0.65	0.68	0.64	0.67
6 Muslims	0.45	0.52	0.48	0.46	0.46	0.49	0.60	0.62	0.62	0.63	0.63	0.61	0.54	0.48	0.51	0.47	0.46	0.50	0.64	0.66	0.66	0.67	0.64	0.66
7 immigrants	0.43	0.48	0.45	0.44	0.44	0.46	0.58	0.59	0.57	0.58	0.59	0.59	0.59	0.54	0.57	0.55	0.56	0.56	0.63	0.64	0.65	0.66	0.62	0.66

Table 12: Accuracy on the different types of targets in the HateCheck tests of RoBERTa models trained on 2,000 gold examples and on the augmented data.

Quantifying Context Mixing in Transformers

Hosein Mohebbi[1] **Willem Zuidema**[2] **Grzegorz Chrupała**[1] **Afra Alishahi**[1]
[1] CSAI, Tilburg University [2] ILLC, University of Amsterdam
`{h.mohebbi, a.alishahi}@tilburguniversity.edu`
`w.h.zuidema@uva.nl`
`grzegorz@chrupala.me`

Abstract

Self-attention weights and their transformed variants have been the main source of information for analyzing token-to-token interactions in Transformer-based models. But despite their ease of interpretation, these weights are not faithful to the models' decisions as they are only one part of an encoder, and other components in the encoder layer can have considerable impact on information mixing in the output representations. In this work, by expanding the scope of analysis to the whole encoder block, we propose *Value Zeroing*, a novel context mixing score customized for Transformers that provides us with a deeper understanding of how information is mixed at each encoder layer. We demonstrate the superiority of our context mixing score over other analysis methods through a series of complementary evaluations with different viewpoints based on linguistically informed rationales, probing, and faithfulness analysis.[1]

1 Introduction

Transformers (Vaswani et al., 2017), with their impressive empirical success, have become a prime choice of architecture to learn contextualized representations across a wide range of modalities, such as language (Devlin et al., 2019; Brown et al., 2020), vision (Dosovitskiy et al., 2021), vision-language (Radford et al., 2021; Rombach et al., 2022), and speech (Baevski et al., 2020), mainly due to their ability to utilize pairwise interactions between input tokens at every timestep.

To better understand the inner dynamics of Transformers, we need to trace the information flow from the input embeddings up to the output representation (including quantifying the degree of *context mixing*, which we will define below). The attention weights from the multi-head attention mechanisms offer a straightforward starting point for understanding this flow, and these weights ('raw attention') have been used in many studies (Clark et al., 2019; Kovaleva et al., 2019; Reif et al., 2019; Htut et al., 2019a, *inter alia*). However, the reliability and usefulness of raw attention weights has also been questioned (Jain and Wallace, 2019; Bibal et al., 2022). In particular, attention weights tend to concentrate on uninformative tokens in the context (Voita et al., 2018; Clark et al., 2019), and removing or altering them may lead to the same and sometimes even better model performance on downstream tasks (Jain and Wallace, 2019; Toneva and Wehbe, 2019; Hassid et al., 2022). These findings suggest that Transformers do not solely rely on self-attention, and other components in the encoder block play an essential role in information mixing.

A number of methods have been proposed to compute some form of 'effective attention weights', with the goal of more faithfully tracing the relative contributions of different input tokens at various layers of the Transformer (as we will discuss in Section 2). These methods show an improvement over raw attention, but still ignore key components of the Transformer encoder block. This is a particularly crucial shortcoming, given that most of the parameter budget in a Transformer encoder is spent on position-wise feed-forward networks outside of the self-attention component, which can have a considerable impact on the degree of information mixing in the output representations.

In this paper, we focus on *context mixing*: the property of Transformers that in each node, at each layer, information from the context can be incorporated into the representation of the target token. We propose *Value Zeroing*, a novel approach to quantify the contribution each context token has in determining the final representation of a target token, at each layer of a Transformer. *Value Zeroing* is based on the Explaining-by-Removing intuition (Covert et al., 2021) shared by many posthoc interpretability methods, but it takes advantage of

---

[1] Code is freely available at `https://github.com/hmohebbi/ValueZeroing`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3378–3400
May 2-6, 2023

a specific feature of Transformers: it zeroes only the value vector of a token t when computing its importance, but leaves the key and query vectors (and thus the pattern of information flow) intact.

Based on extensive experiments and three complementary approaches to evaluation, we demonstrate that importance scores we can obtain with *Value Zeroing* provide better interpretations than other analysis methods.

Firstly, we use a set of grammatical agreement tasks from the BLiMP corpus (Warstadt et al., 2020) as a case study. Transformer-based models do extremely well on the task of distinguishing grammatical from ungrammatical sentences, and the BLiMP corpus provides information on the *cue* words that determine the difference. We find that *Value Zeroing*, unlike earlier approaches, indeed reveals that Transformers make use of relevant cues.

Secondly, we use information-theoretic probing (Voita and Titov, 2020; Pimentel et al., 2020) as an independent approach to track information flow in Transformer networks. The scores we obtain with *Value Zeroing* turn out to be highly correlated with layer-wise probing performance; that is, probing accuracy is higher in layers where relevant tokens are more effectively utilized by the model.

Thirdly, we assess the faithfulness of our method (Jacovi and Goldberg, 2020); compared to alternative analysis methods, we show that *Value Zeroing* is not only more plausible and human-interpretable, but also more faithful to models' decisions.

2 Related Work

While numerous studies have leveraged the weights assigned by the self-attention mechanism to gain intuition about the information mixing process in Transformers (Clark et al., 2019; Kovaleva et al., 2019; Reif et al., 2019; Lin et al., 2019; Htut et al., 2019b; Raganato and Tiedemann, 2018), it is still a matter of debate whether attention weights are suitable for interpreting the model (see Bibal et al. (2022)'s study for a full discussion). Thus several post-processing interpretability techniques have been proposed to convert these weights into scores that provide a more detailed interpretation of the inner workings of Transformers. We review the main approaches below.

Abnar and Zuidema (2020) propose the *attention-flow* and *attention-rollout* methods to approximate information flow in Transformers based on raw attention weights. The former treats raw attention weight matrices as a flow network and returns the maximum flow through each input token. The latter recursively multiplies the attention weight matrix at each layer by the preceding ones. There is, however, an unjustified assumption in the formulation of these methods that both multi-head attention and residual connections contribute equally to the computation of the output.

Kobayashi et al. (2020) propose a method that incorporates the norm of the transformed value vectors and report a negative correlation between these norms and raw attention weights on frequent tokens, which partially explains the insufficiency of raw attention weights for context mixing estimation. Kobayashi et al. (2021) extend this method to the whole self-attention block by incorporating Residual connections (RES) and Layer Normalization (LN) (two components with significant impact on both model performance and training convergence (Parisotto et al., 2020; Liu et al., 2020)), but demonstrate that RES and LN components largely cancel out the mixing process. Kobayashi et al. (2021)'s method, however, ignores the effect of the second sublayer in a Transformer's encoder.

Brunner et al. (2020) and Pascual et al. (2021) employ a gradient-based approach for analyzing the interaction of input representations, but the gradient measures the sensitivity between two vectors and ignores the impact of the input vector. In our experiments we show that despite their relative success in explaining model decisions, gradient-based approaches are not suitable for layer-wise analysis.

More recently, the effectiveness of combining these approaches has also been investigated. Modarressi et al. (2022) propose a method that uses Kobayashi et al. (2021)'s scores and incorporates the effects of the second layer normalization; they aggregate those scores using *rollout* (Abnar and Zuidema, 2020) to provide global token attributions. In the same vein, Ferrando et al. (2022) use rollout to aggregate a variant of Kobayashi et al. (2021)'s scores: instead of relying on the Euclidean norm of the transformed vector, they measure Manhattan distance of each transformed vector to the context vector outputted from self-attention block. In both studies, however, the fact that these context vectors might undergo significant changes after passing through the second sublayer in the encoder layer is not taken into account. We will show (in Section 7) that even at a global level, our scores provide better interpretation than prior methods.

3 Our Proposed Method

To remedy for the limited scope of the existing methods, we introduce a new context mixing score that takes into account all components in a Transformer encoder block.

3.1 Background and Notation

In this section, we set up the notation and briefly review the internal structure of an encoder layer in the Transformer architecture.

Each Transformer encoder layer is composed of two sublayers: a multi-head self-attention mechanism (**MHA**) and a position-wise fully connected feed-forward network (**FFN**), followed by a Residual connection (**RES**) and Layer Normalization (**LN**) around each of these two sublayers. This encoder layer produces the next contextualized representations $(\tilde{\boldsymbol{x}}_1, ..., \tilde{\boldsymbol{x}}_n)$ for each token in the context, using the output representations from the previous layer $(\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$.

MHA. For each head $h \in \{1, ..., H\}$ in the self-attention module, each input vector $\boldsymbol{x}_i$ is transformed into a query $\boldsymbol{q}_i^h$, a key $\boldsymbol{k}_i^h$, and a value $\boldsymbol{v}_i^h$ vector via separate trainable linear transformations:

$$\boldsymbol{q}_i^h = \boldsymbol{x}_i \boldsymbol{W}_{\boldsymbol{Q}}^{\boldsymbol{h}} + \boldsymbol{b}_Q^h \tag{1}$$

$$\boldsymbol{k}_i^h = \boldsymbol{x}_i \boldsymbol{W}_{\boldsymbol{K}}^{\boldsymbol{h}} + \boldsymbol{b}_K^h \tag{2}$$

$$\boldsymbol{v}_i^h = \boldsymbol{x}_i \boldsymbol{W}_{\boldsymbol{V}}^{\boldsymbol{h}} + \boldsymbol{b}_V^h \tag{3}$$

The context vector $\boldsymbol{z}_i^h$ for the i^{th} token of each attention head is then generated as a weighted sum over the transformed value vectors:

$$\boldsymbol{z}_i^h = \sum_{j=1}^{n} \alpha_{i,j}^h \boldsymbol{v}_j^h \tag{4}$$

where $\alpha_{i,j}^h$ is the raw attention weight assigned to the j^{th} token, and computed as a softmax-normalized dot product between the corresponding query and key vectors:

$$\alpha_{i,j} = \underset{\boldsymbol{x}_j \in \mathcal{X}}{\text{softmax}} \left(\frac{\boldsymbol{q}_i \boldsymbol{k}_j^\top}{\sqrt{d}} \right) \in \mathbb{R} \tag{5}$$

Next, the context vector $(\boldsymbol{z}_i \in \mathbb{R}^d)$ for the i^{th} token is computed by concatenating all the heads' outputs followed by a head-mixing $\boldsymbol{W}_O$ projection and layer normalization:

$$\boldsymbol{z}_i = \text{CONCAT}(\boldsymbol{z}_i^1, ..., \boldsymbol{z}_i^H) \boldsymbol{W}_O \tag{6}$$

$$\boldsymbol{z}_i = \text{LN}_{\text{MHA}}(\boldsymbol{z}_i + \boldsymbol{x}_i) \tag{7}$$

FFN. Each encoder layer also includes two linear transformations with a ReLU activation in between, which is applied to every $\boldsymbol{z}_i$ separately and identically to produce output token representations $\tilde{\boldsymbol{x}}_i$:

$$\tilde{\boldsymbol{x}}_i = \max(0, \boldsymbol{z}_i \boldsymbol{W}_1 + \boldsymbol{b}_1) \boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{8}$$

$$\tilde{\boldsymbol{x}}_i = \text{LN}_{\text{FFN}}(\tilde{\boldsymbol{x}}_i + \boldsymbol{z}_i) \tag{9}$$

3.2 Value Zeroing

We aim to measure how much a token uses other context tokens to build its output representation $\tilde{\boldsymbol{x}}_i$ at each encoder layer. To this end, we treat the self-attention mechanism as a fuzzy hash-table, where we look up the sum of values weighted by the query-key match in the context. Thus in Eq. 4 we replace a value vector associated with token j with a zero vector $\boldsymbol{v}_j^h \leftarrow \boldsymbol{0}, \forall h \in H$, where the context vector for the i^{th} token is being computed. This provides an alternative output representation $\tilde{\boldsymbol{x}}_i^{\neg j}$ for the i^{th} token that has excluded token j in the mixing process. By comparing the alternative output representation $\tilde{\boldsymbol{x}}_i^{\neg j}$ with the original $\tilde{\boldsymbol{x}}_i$, we can measure how much the output representation is affected by the exclusion of the j^{th} token:

$$\mathcal{C}_{i,j} = \tilde{\boldsymbol{x}}_i^{\neg j} * \tilde{\boldsymbol{x}}_i \tag{10}$$

where the operation $*$ can be any pairwise distance metric that properly considers the characteristics of the model's representation space. We opted for *cosine* distance throughout our experiments as its superiority over other dissimilarity metrics has been supported for textual deep learning models (Yokoi et al., 2020; Hanawa et al., 2021).[2] Computing Eq. 10 for all $\tilde{\boldsymbol{x}}_i$ in a given context provides us with a *Value Zeroing* matrix score $\boldsymbol{\mathcal{C}}$ where the value of the cell $\mathcal{C}_{i,j} \in \mathbb{R}$ (i^{th} row, j^{th} column) in the map denotes the degree to which the i^{th} token depends on the j^{th} token to form its contextualized representation.

Note that unlike generic perturbation approaches, our proposed method *does not* remove the token representations $\boldsymbol{x}_i$ from the input of an encoder. We argue that ablating input token representations cannot be a reliable basis to understand context mixing process since any changes in the input vectors will lead to changes in the query and key vectors (Eq. 1 and 2), resulting in a change in the attention distribution (cf. Eq. 5). Consequently, there will

---

[2] More details on the choice of distance metric is discussed in Appendix A.1.

Phenomenon	**UID**	**Example**	**Target word**	**Foil word**
Anaphor Number Agreement	ana	Many teenagers were helping [MASK].	themselves	herself
Determiner-Noun Agreement	dna	Jeffrey has not passed [MASK] museums.	these	this
	dnaa	Sara noticed [MASK] white hospitals.	these	this
Subject-Verb Agreement	darn	The pictures of Martha [MASK] not disgust Anne.	do	does
	rpsv	Kristen [MASK] fixed this chair.	has	have

Table 1: Examples of the selected tasks with our annotations from the BLiMP benchmark (UIDs are unique identifiers used in BLiMP). Cue words are underlined.

be a discrepancy between the alternative attention weights and those for the original context. Instead, our method only nullifies the *value vector* of a specific token representation. In this way, the token representation can maintain its identity within the encoder layer, but it does not contribute to forming other token representations. Moreover, since our *Value Zeroing* is computed from the encoder's layer outputs, it incorporates all the components inside an encoder layer such as multi-head attention, residual connection, layer normalization, and also feed-forward networks, resulting in a more reliable context mixing score than previous methods.

4 Experimental Setup

4.1 Data

We used the BLiMP benchmark (Warstadt et al., 2020) which contains a set of pairs of minimally different sentences that contrast in grammatical acceptability under a specific linguistic phenomenon. The benchmark isolates linguistic phenomena such that only one word determines the true label of each sentence. We refer to this crucial context token as the *cue word*. The nature of this task makes it especially suitable for evaluating context mixing scores, since it gives us a strong hypothesis on which context token is the most relevant for the representation of the masked *target word*.

From this benchmark, we select five datasets with three different linguistic phenomena for which Pre-trained Language Models (PLMs) have shown high accuracy to ensure that the model captures the relevant information. We expand contractions such as *doesn't* → *does not*) and generate dependency trees using SpaCy (Honnibal and Montani, 2017) to extract and annotate the position of target and cue words in a sentence. In Table 1, we provide an example of each phenomenon in the benchmark together with our automated annotations. We accumulate examples from the five selected tasks as a unified dataset for grammatical agreement, resulting in 4,276 data points, and divide them equally into Train and Test sets. The Train set is only used for the fine-tuning phase; the Test set is used for all evaluation experiments.

4.2 Target Model

We conduct our experiments on three Transformer-based language models: BERT (uncased, Devlin et al., 2019), RoBERTa (Liu et al., 2019) and ELECTRA (Clark et al., 2020).[3] The results for the latter two are reported in Appendix A.3. By replacing the target words with the [MASK] token, we perform a Masked Language Modeling (MLM) task using the model's pre-trained MLM head. For instance, in the Subject-Verb Agreement example *"The pictures of Martha do not disgust Anne."*, we replace the verb *'do'* with the [MASK] token and feed the example to the model.

We perform our experiments on both pre-trained and fine-tuned versions of each model. Including a fine-tuned model in our analysis study gives us a complementary insight into the importance of the cue words, since fine-tuning allows the model to concentrate on the most helpful words for the downstream task of choice (i.e., agreement) and makes sure that target word representations take the cue word into account. We use prompt fine-tuning (Schick and Schütze, 2021a,b; Karimi Mahabadi et al., 2022) and compute Cross Entropy loss only over the output logits corresponding to the target and foil classes. Accuracy is 0.96 for pre-trained and 0.99 for fine-tuned BERT.

4.3 Baselines

Here we describe the existing context mixing methods which we include in our experiments. For each method, we select the m^{th} row of its context mixing map where m is the position of the [MASK] token, resulting in a 1-D array of scores for each context

---

[3] Base, with 12 layers and 12 attention heads, obtained from the Transformers library (Wolf et al., 2020).

token. We normalize the scores for all tokens in a sentence so that they are all positive values and sum to one. For completeness, we also include a few gradient-based attribution methods in our comparisons.

Rand: random scores generated from a uniform distribution for each sentence in the dataset.

Attn: raw attention scores $\boldsymbol{\alpha}_m$ from Eq. 5.

Attn-rollout: An aggregation method for approximating the attention flow based on raw attention weights (Abnar and Zuidema, 2020).

Attn-Norm: norm-based method of Kobayashi et al. (2020) that also incorporates the norm of the transformed input vectors to compute context mixing scores.

Attn-Norm + RES + LN: the extended norm-based method of Kobayashi et al. (2021) in which they also incorporates Residual connection (RES) and Layer Normalization (LN) located only in the first sublayer of a Transformer's encoder.

GlobEnc & ALTI: Two global token attribution methods proposed by Modarressi et al. (2022) and Ferrando et al. (2022). For a fair comparison, we exclude the aggregation through rollout in our first two evaluations (referred to as GlobEnc¬ and ALTI¬), because we are interested in assessing context mixing scores at each layer separately. In our third evaluation, we compare these methods with others at a global level and use rollout.

GradXinput, IG & DL: feature attribution scores that also make use of *top-down* information from the classification layer on top of the Transformer. We consider three popular variants of gradient-based attribution scores: Gradient×Input (**GradXinput**) (Samek et al., 2019; Yuan et al., 2019), Integrated Gradients (**IG**) (Sundararajan et al., 2017), and DeepLift (**DL**) (Shrikumar et al., 2017). For gradient-based methods, we use the pre-trained MLM head which has been trained during the pre-training of the BERT to compute the gradient of the true label with respect to the token representations at each layer.

5 Evaluation 1: Cue Alignment

As cue words are the only indicators of the true labels in our dataset, we expect that when the model performs well, it overwhelmingly depends on these words to form the representation of a [MASK] token in a given context. To quantify the alignment between a context mixing score and the cue word, we first define a binary cue vector ξ according to the following condition:

$$\xi_i = \begin{cases} 1, & \text{the } i^{\text{th}} \text{ token} \in \text{Cue words} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

Then we compare the cue vector and the prediction of a context mixing score $\boldsymbol{S}$ in two different ways:

Dot Product. We quantify cue alignment as $\boldsymbol{S} \cdot \boldsymbol{\xi}$, which measures the total score mass the model assigns to cue words to form the representation of the target token.

Average Precision. We quantify cue alignment as the average precision between the two vectors, which is a weighted mean of precision at each recall level:

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n \tag{12}$$

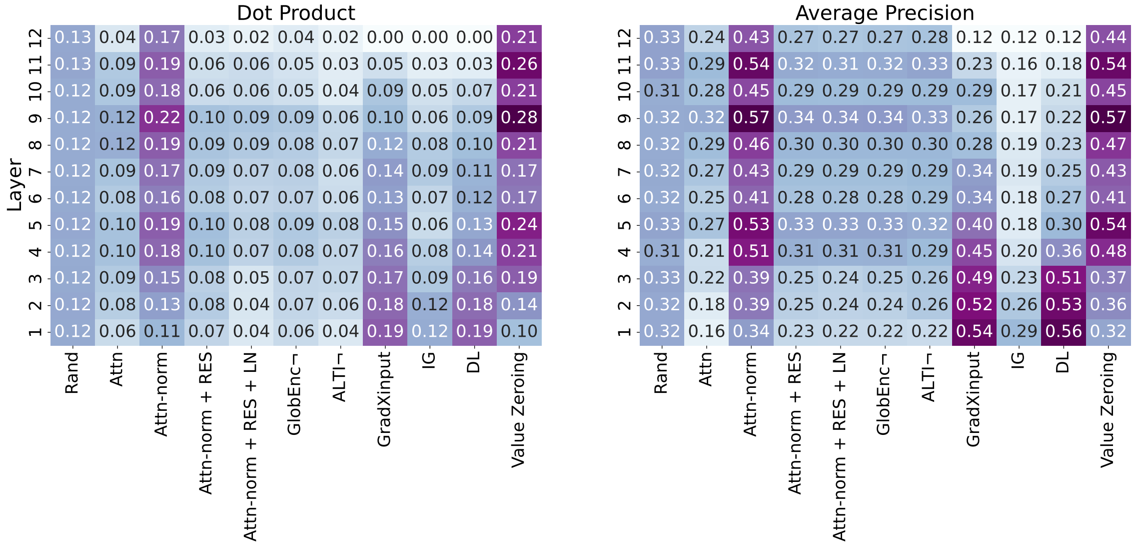
where P_n and R_n are the precision and recall at the n^{th} threshold. This metric relies on the ranking of tokens rather than the magnitude of their weights.[4]

Figures 1 and 2 show the alignment between the cue vector and different analysis methods using dot product and average precision for the pre-trained and fine-tuned model, respectively. In comparison with the other context mixing methods, *Value Zeroing* shows a higher degree of the target model incorporating cue words into the representation of the [MASK] token across all layers.

As can be seen from the first two columns in all graphs, raw self-attention weights (**Attn**) always perform worse than even random scores in highlighting cue words. This is in line with previous studies showing that raw attention weights often pay attention to uninformative tokens (Voita et al., 2018; Clark et al., 2019) and do not reflect the appropriate context (Kim et al., 2019). However, we can see a significant improvement in the results for **Attn-norm** where the norm of transformed value vectors are also taken into account, confirming that value vectors play an essential role in the context mixing process. The method **Attn-norm + RES + LN**, which expands **Attn-norm** to the whole self-attention block by adding Residual

---

[4] We also employed Probes-needed (Zhong et al., 2019) metric in our evaluation which intuitively counts the number of non-cue tokens we need to probe to find cue words based on a given score. As its motivation is similar to Average Precision and the results show the same pattern, we relegate the results with this metric to the Appendix A.2.



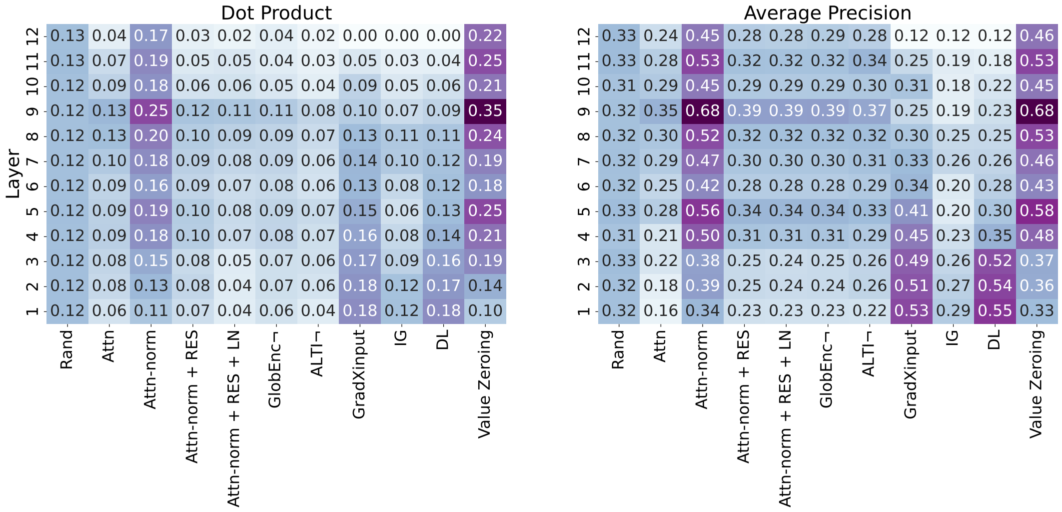
Dot Product

Layer	Rand	Attn	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.13	0.04	0.17	0.03	0.02	0.04	0.02	0.00	0.00	0.00	0.21
11	0.13	0.09	0.19	0.06	0.06	0.05	0.03	0.05	0.03	0.03	0.26
10	0.12	0.09	0.18	0.06	0.06	0.05	0.04	0.09	0.05	0.07	0.21
9	0.12	0.12	0.22	0.10	0.09	0.09	0.06	0.10	0.06	0.09	0.28
8	0.12	0.12	0.19	0.09	0.09	0.08	0.07	0.12	0.08	0.10	0.21
7	0.12	0.09	0.17	0.09	0.07	0.08	0.06	0.14	0.09	0.11	0.17
6	0.12	0.08	0.16	0.08	0.07	0.07	0.06	0.13	0.07	0.12	0.17
5	0.12	0.10	0.19	0.10	0.08	0.09	0.08	0.15	0.06	0.13	0.24
4	0.12	0.10	0.18	0.10	0.07	0.08	0.07	0.16	0.08	0.14	0.21
3	0.12	0.09	0.15	0.08	0.05	0.07	0.07	0.17	0.09	0.16	0.19
2	0.12	0.08	0.13	0.08	0.04	0.07	0.06	0.18	0.12	0.18	0.14
1	0.12	0.06	0.11	0.07	0.04	0.06	0.04	0.19	0.12	0.19	0.10

Average Precision

Layer	Rand	Attn	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.33	0.24	0.43	0.27	0.27	0.27	0.28	0.12	0.12	0.12	0.44
11	0.33	0.29	0.54	0.32	0.31	0.32	0.33	0.23	0.16	0.18	0.54
10	0.31	0.28	0.45	0.29	0.29	0.29	0.29	0.29	0.17	0.21	0.45
9	0.32	0.32	0.57	0.34	0.34	0.34	0.33	0.26	0.17	0.22	0.57
8	0.32	0.29	0.46	0.30	0.30	0.30	0.30	0.28	0.19	0.23	0.47
7	0.32	0.27	0.43	0.29	0.29	0.29	0.29	0.34	0.19	0.25	0.43
6	0.32	0.25	0.41	0.28	0.28	0.28	0.29	0.34	0.18	0.27	0.41
5	0.33	0.27	0.53	0.33	0.33	0.33	0.32	0.40	0.18	0.30	0.54
4	0.31	0.21	0.51	0.31	0.31	0.31	0.29	0.45	0.20	0.36	0.48
3	0.33	0.22	0.39	0.25	0.24	0.25	0.26	0.49	0.23	0.51	0.37
2	0.32	0.18	0.39	0.25	0.24	0.24	0.26	0.52	0.26	0.53	0.36
1	0.32	0.16	0.34	0.23	0.22	0.22	0.22	0.54	0.29	0.56	0.32

Figure 1: Layer-wise alignment between the cue vector and different analysis methods averaged over Test set examples for the **pre-trained** model. Higher value (darker color) is better.



Dot Product

Layer	Rand	Attn	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.13	0.04	0.17	0.03	0.02	0.04	0.02	0.00	0.00	0.00	0.22
11	0.13	0.07	0.19	0.05	0.05	0.04	0.03	0.05	0.03	0.04	0.25
10	0.12	0.09	0.18	0.06	0.06	0.05	0.04	0.09	0.05	0.06	0.21
9	0.12	0.13	0.25	0.12	0.11	0.11	0.08	0.10	0.07	0.09	0.35
8	0.12	0.13	0.20	0.10	0.09	0.09	0.07	0.13	0.11	0.11	0.24
7	0.12	0.10	0.18	0.09	0.08	0.09	0.06	0.14	0.10	0.12	0.19
6	0.12	0.09	0.16	0.09	0.07	0.08	0.06	0.13	0.08	0.12	0.18
5	0.12	0.09	0.19	0.10	0.08	0.09	0.07	0.15	0.06	0.13	0.25
4	0.12	0.09	0.18	0.10	0.07	0.08	0.07	0.16	0.08	0.14	0.21
3	0.12	0.08	0.15	0.08	0.05	0.07	0.06	0.17	0.09	0.16	0.19
2	0.12	0.08	0.13	0.08	0.04	0.07	0.06	0.18	0.12	0.17	0.14
1	0.12	0.06	0.11	0.07	0.04	0.06	0.04	0.18	0.12	0.18	0.10

Average Precision

Layer	Rand	Attn	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.33	0.24	0.45	0.28	0.28	0.29	0.28	0.12	0.12	0.12	0.46
11	0.33	0.28	0.53	0.32	0.32	0.32	0.34	0.25	0.19	0.18	0.53
10	0.31	0.29	0.45	0.29	0.29	0.29	0.30	0.31	0.18	0.22	0.45
9	0.32	0.35	0.68	0.39	0.39	0.39	0.37	0.25	0.19	0.23	0.68
8	0.32	0.30	0.52	0.32	0.32	0.32	0.32	0.30	0.25	0.25	0.53
7	0.32	0.29	0.47	0.30	0.30	0.30	0.31	0.33	0.26	0.26	0.46
6	0.32	0.25	0.42	0.28	0.28	0.28	0.29	0.34	0.20	0.28	0.43
5	0.33	0.28	0.56	0.34	0.34	0.34	0.33	0.41	0.20	0.30	0.58
4	0.31	0.21	0.50	0.31	0.31	0.31	0.29	0.45	0.23	0.35	0.48
3	0.33	0.22	0.38	0.25	0.24	0.25	0.26	0.49	0.26	0.52	0.37
2	0.32	0.18	0.39	0.25	0.24	0.24	0.26	0.51	0.27	0.54	0.36
1	0.32	0.16	0.34	0.23	0.23	0.23	0.22	0.53	0.29	0.55	0.33

Figure 2: Layer-wise alignment between the cue vector and different analysis methods averaged over Test set examples for the **fine-tuned** model. Higher value (darker color) is better.

connection and Layer Normalization, would seem to show that the model is incapable of utilizing the cue words. However, incorporating also the second part of the encoder layer via our method shows the model does indeed use the cue words. We also find that GlobEnc and ALTI benefit from the rollout aggregation method to provide a global view, but they do not seem to provide good layerwise scores (without rollout).

The gradient-based scores, in contrast to the other methods, highlight the cue words only in the earlier layers of the model. In the next section, by using a layer-wise probing experiment, we will show that these scores are not reliable for identifying the relevant context in individual layers.

6 Evaluation 2: Context Mixing versus Probing

In this section, we investigate the relationship between cue word alignment and probing performance across layers. We hypothesize that if a layer aligns better with the cue word according to a reliable context mixing score, then the representation of the masked token on that layer can be used more effectively by a probing classifier to decode number agreement with the cue word.

To verify our hypothesis, we obtain the representation of masked tokens in test examples across all layers. Since all examples in our dataset share the same number agreement property, we asso-

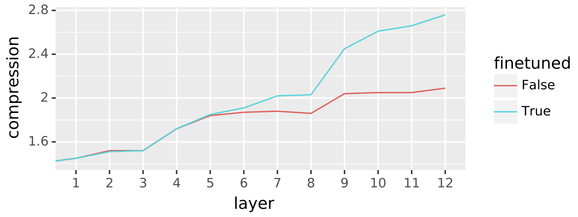


Figure 3: Layer-wise compression of probing classifiers using pre-trained and fine-tuned representations.

ciate each masked representation with a *Singular* or *Plural* label. Next, we perform an information-theoretic probing analysis using Minimum Description Length (MDL) to measures the degree to which representations encode number agreement. We chose MDL as our probe since it is theoretically justified and has been shown to provide more reliable results than conventional probes (Voita and Titov, 2020; Fayyaz et al., 2021).

To compute MDL, we employed the *online coding* of Voita and Titov (2020). Since MDL can be affected by the number of data points (N), we measure compression as our evaluation metric which is defined as follows:

$$\text{Compression} = \frac{N \cdot \log_2(K)}{\text{MDL}} \qquad (13)$$

where K refers to the number of classes (2 in our case). This metric is equal to 1 (no compression) for a random guessing classifier. A higher value for Compression indicates more accurate label prediction for the probing classifier.

Figure 3 reports compression of probing classifiers based on representations obtained from both pre-trained and fine-tuned models across all layers. We also include the results for the embedding layer of the model (layer 0) which can serve as a non-contextualized baseline. We can see a jump in probing performance at layers 4 ($1.52 \rightarrow 1.72$) and 9 ($2.03 \rightarrow 2.45$) in the fine-tuned setup, the same layers for which we found a higher alignment with cue words in Figures 1 and 2.

Table 2 presents the correlation between layer-wise Compression scores and layer-wise cue alignment scores from Section 5 for different analysis methods. As we can see, alignment according to *Value Zeroing* is highly positively correlated with the probing performance. This suggests that when *Value Zeroing* indicates that the model uses cue words to form representations of the masked tokens in a particular layer, these representations are in fact better at encoding number agreement.

Method	ρ^{PT}	ρ^{FT}
Rand	-0.07	-0.02
Attn	0.10	0.08
Attn-norm	0.52	0.56
Attn-norm + RES	-0.35	-0.24
Attn-norm + RES + LN	0.12	0.17
GlobEnc¬	-0.01	-0.12
ALTI¬	-0.01	-0.12
GradXinput	-0.96	-0.99
IG	-0.86	-0.77
DL	-0.97	-1.00
Value Zeroing	**0.65**	**0.64**

Table 2: Spearman's ρ correlation between layer-wise probing performance (Comp.) and layer-wise cue alignments based on Dot Product. PT and FT refer to pre-trained and fine-tuned conditions, respectively.

Recall that based on Figures 1 and 2, according to the gradient-based methods the masked tokens pay more attention to the cue words only in earlier layers. However, we can see a highly negative correlation with probing results for these scores. Due to the nature of the task the probing score goes up monotonically along the layers. At the same time, the gradient attribution score goes up monotonically as you get closer to the bottom embedding layers, suggesting that gradient-based methods are unreliable for layer-wise analysis and identifying important tokens in the context mixing process.

7 Evaluation 3: Faithfulness Analysis

Our experimental results in Sections 5 and 6 show that the *Value Zeroing* score matches our prior linguistically-informed expectations. However, it is not always clear whether a *plausible* context mixing score that matches human expectations is also faithful to the model and reflects its decision making process (Herman, 2017; Wiegreffe and Pinter, 2019; Jacovi and Goldberg, 2020).

In this section we employ the notion of input ablation (Covert et al., 2021) to evaluate the faithfulness of our context mixing score. The influence of a target token on a model's decision is often estimated as the drop in the model's predicted probability of the correct class after blanking out the target token from the input. A higher drop for an ablated token indicates that the token is more influential on the model's decision (DeYoung et al., 2020; Abnar and Zuidema, 2020; Atanasova et al., 2020; Wang et al., 2022). We use this *blank-out* approach as a base for analyzing and comparing the faithfulness of context mixing scores.

Method	ρ^{PT}	ρ^{FT}
Rand	-0.01	0.00
Attn	-0.10	-0.07
Attn-norm	0.19	0.14
Attn-norm + RES	0.03	-0.05
Attn-norm + RES + LN	-0.08	-0.17
GlobEnc	-0.01	-0.09
ALTI	0.17	0.19
GradXinput	0.11	0.16
IG	0.07	0.21
DL	0.20	0.29
Value Zeroing	**0.26**	**0.31**

Table 3: Spearman's ρ correlation between the blank-out scores and different aggregated context mixing and attribution scores. PT and FT refer to pre-trained and fine-tuned conditions, respectively.

Figure 4: Most influential tokens on the target representation in a fine-tuned BERT model according to different aggregated context mixing scores compared to blank-out scores.

To estimate the blank-out scores in BERT, we calculate the probability of its output y using a softmax function normalized over only the corresponding logit values of target t and foil words (cf. Table 1), and compute blank-out scores for a given input token i as $p(y_t|\boldsymbol{e}) - p(y_t|\boldsymbol{e}\backslash e_i)$, where $\boldsymbol{e_i}$ refers to the input embedding of input token i. We compare these blank-out scores with context mixing scores, aggregated across all layers of the model. For gradient-based scores, calculating them with respect to the tokens in the input embedding layer ($\ell = 0$) provides us with aggregated scores since the backpropagation of gradients passes through all layers to the beginning of the model. For other scores, we use the *rollout* (Abnar and Zuidema, 2020) aggregation method.

Table 3 shows Spearman's rank correlation between the blank-out scores and different aggregated context mixing scores. The highest correlation for our method indicates that *Value Zeroing* is more faithful in explaining the model behaviour compared to other analysis methods.

Qualitative Analysis. We also take a closer look at the aggregated scores for a qualitative comparison. In Figure 4, we illustrate different scores obtained from a fine-tuned BERT model for a correctly classified example, where the model is asked to fill the masked token with one of the verbs *were* or *is* as target and foil classes, respectively. According to *Value Zeroing* scores, the model mainly relies on the main subject (*pictures*) as a cue word to form a contextualized representation of the [MASK] token, while the word *pictures* is also important for the model's final decision based on the blank-out scores. In this example, the blank-out score for the cue word is 0.99, meaning the model fully loses its confidence in the target class when the cue word is replaced with an `[UNK]` token. Surprisingly, gradient-based methods tend to highlight the word *hat* which is an agreement attractor, and attention-based scores tend to focus on the `[CLS]` token which has been idle during fine-tuning process.

Overall, our faithfulness evaluation and qualitative analysis suggest that *Value Zeroing* can explain model decisions at a global level when it is aggregated across layers. The context mixing maps per layer are provided in Appendix A.4, where some more meaningful patterns can be found in Value Zeroing scores (in both layer-wise and aggregated setups) in contrast to other context mixing scores.

8 Discussion

Although some desiderata such as plausibility (Lei et al., 2016; Strout et al., 2019) and faithfulness (Lakkaraju et al., 2019; Jacovi and Goldberg, 2020) are taken into account when developing explanation and analysis methods, evaluating them is still a challenge due to lack of a standard ground truth. Evaluating context mixing scores, where token-to-token interactions in a context are also considered, is even more challenging. Several studies have used gradient-based scores as an anchor of faithfulness, and measure how strongly context mixing scores correlate with them (Jain and Wallace, 2019; Abnar and Zuidema, 2020; Modarressi et al., 2022). However, the reliability of gradient-based scores can be questioned, especially when different variations of them show considerable disagreement

(Neely et al., 2022; Pruthi et al., 2022; Krishna et al., 2022). Thus, we suggest using controlled tasks for which we have strong prior expectations for evaluating these methods. In our study, we use a set of number agreement tasks to provide such priors, since the cue words are the only sources of information in the context for performing well in the task.

Another point worth discussing is the concern raised by Kobayashi et al. (2021) that BERT tends to preserve token representations rather than mixing them at each layer. We argue that their observation is due to the context-mixing ratio they defined by comparing the norm of residual effects against other token representations. In our view, this ratio is dominated by residuals and neglects the fact that a token representation carried by residual connections is indeed a contextualized representation outputted from previous layers. We keep the residuals intact within the encoder layer by zeroing only the value vectors and focusing on the context mixing performed by all tokens.

9 Conclusion

In this paper, we propose Value Zeroing as a novel approach for quantifying the information mixing process in Transformers to address the shortcomings of previous methods. We performed extensive complementary experiments and showed that our method outperforms others in three different evaluation setups. Since our approach requires no supervision, it could be an interesting option for improving model efficiency by removing token representations across layers.

10 Limitations

As is the case for most attempts at interpreting Deep Learning models, our evaluation of our (and others') proposed methods are not definite since we have no gold standard of what happens inside a model, although we try to remedy for that by conducting independent and complementary evaluation schemes.

Our proposed method is customized for deep neural models based on the Transformer architecture and cannot be easily generalized to other (mathematically different) modeling architectures. Our evaluations were based on encoder-based models, and focused on the Text modality. In the future, we will extend our experiments to more modalities, such as speech and vision.

Acknowledgments

This publication is part of the project *InDeep: Interpreting Deep Learning Models for Text and Sound* (with project number NWA.1292.19.399) of National Research Agenda (NWA-ORC) programme. Funding by the Dutch Research Council (NWO) is gratefully acknowledged.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. A diagnostic study of explainability techniques for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3256–3274, Online. Association for Computational Linguistics.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Adrien Bibal, Rémi Cardon, David Alfter, Rodrigo Wilkens, Xiaoou Wang, Thomas François, and Patrick Watrin. 2022. Is attention explanation? an introduction to the debate. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3889–3900, Dublin, Ireland. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *International Conference on Learning Representations*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Ian Covert, Scott M. Lundberg, and Su-In Lee. 2021. Explaining by removing: A unified framework for model explanation. *J. Mach. Learn. Res.*, 22:209:1–209:90.

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

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Mohsen Fayyaz, Ehsan Aghazadeh, Ali Modarressi, Hosein Mohebbi, and Mohammad Taher Pilehvar. 2021. Not all models localize linguistic knowledge in the same place: A layer-wise probing on BERToids' representations. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 375–388, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Javier Ferrando, Gerard I. Gállego, and Marta R. Costa-jussà. 2022. Measuring the mixing of contextual information in the transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8698–8714, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kazuaki Hanawa, Sho Yokoi, Satoshi Hara, and Kentaro Inui. 2021. Evaluation of similarity-based explanations. In *International Conference on Learning Representations*.

Michael Hassid, Hao Peng, Daniel Rotem, Jungo Kasai, Ivan Montero, Noah A. Smith, and Roy Schwartz. 2022. How much does attention actually attend? questioning the importance of attention in pretrained transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1403–1416, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bernease Herman. 2017. The promise and peril of human evaluation for model interpretability. *ArXiv*, abs/1711.07414.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019a. Do attention heads in BERT track syntactic dependencies? *CoRR*, abs/1911.12246.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019b. Do attention heads in bert track syntactic dependencies? *ArXiv*, abs/1911.12246.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, Luke Zettlemoyer, James Henderson, Lambert Mathias, Marzieh Saeidi, Veselin Stoyanov, and Majid Yazdani. 2022. Prompt-free and efficient few-shot learning with language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3638–3652, Dublin, Ireland. Association for Computational Linguistics.

Yunsu Kim, Duc Thanh Tran, and Hermann Ney. 2019. When and why is document-level context useful in neural machine translation? In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 24–34, Hong Kong, China. Association for Computational Linguistics.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2021. Incorporating Residual and Normalization Layers into Analysis of Masked Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4547–4568, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Satyapriya Krishna, Tessa Han, Alex Gu, Javin Pombra, Shahin Jabbari, Steven Wu, and Himabindu Lakkaraju. 2022. The disagreement problem in explainable machine learning: A practitioner's perspective. *ArXiv*, abs/2202.01602.

Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. 2019. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, page 131–138, New York, NY, USA. Association for Computing Machinery.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Fenglin Liu, Xuancheng Ren, Zhiyuan Zhang, Xu Sun, and Yuexian Zou. 2020. Rethinking skip connection with layer normalization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3586–3598, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Ali Modarressi, Mohsen Fayyaz, Yadollah Yaghoobzadeh, and Mohammad Taher Pilehvar. 2022. GlobEnc: Quantifying global token attribution by incorporating the whole encoder layer in transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 258–271, Seattle, United States. Association for Computational Linguistics.

Michael Neely, Stefan F. Schouten, Maurits Bleeker, and Ana Lucic. 2022. A song of (dis)agreement: Evaluating the evaluation of explainable artificial intelligence in natural language processing. In *HHAI*.

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphaël Lopez Kaufman, Aidan Clark, Seb Noury, Matthew Botvinick, Nicolas Heess, and Raia Hadsell. 2020. Stabilizing transformers for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7487–7498. PMLR.

Damian Pascual, Gino Brunner, and Roger Wattenhofer. 2021. Telling BERT's full story: from local attention to global aggregation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 105–124, Online. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, Livio Baldini Soares, Michael Collins, Zachary Chase Lipton, Graham Neubig, and William W. Cohen. 2022. Evaluating explanations: How much do explanations from the teacher aid students? *Transactions of the Association for Computational Linguistics*, 10:359–375.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 287–297, Brussels, Belgium. Association for Computational Linguistics.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. In *Advances in Neural Information Processing Systems*, pages 8594–8603.

Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685.

Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. 2019. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3145–3153. JMLR.org.

Julia Strout, Ye Zhang, and Raymond Mooney. 2019. Do human rationales improve machine explanations? In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–62, Florence, Italy. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

William Timkey and Marten van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4527–4546, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mariya Toneva and Leila Wehbe. 2019. Interpreting and improving natural-language processing (in machines) with natural language-processing (in the brain). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Lijie Wang, Yaozong Shen, Shu ping Peng, Shuai Zhang, Xinyan Xiao, Hao Liu, Hongxuan Tang, Ying Chen, Hua Wu, and Haifeng Wang. 2022. A fine-grained interpretability evaluation benchmark for neural nlp. *ArXiv*, abs/2205.11097.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Sho Yokoi, Ryo Takahashi, Reina Akama, Jun Suzuki, and Kentaro Inui. 2020. Word rotator's distance. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2944–2960, Online. Association for Computational Linguistics.

Hao Yuan, Yongjun Chen, Xia Hu, and Shuiwang Ji. 2019. Interpreting deep models for text analysis via optimization and regularization methods. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Ruiqi Zhong, Steven Shao, and Kathleen McKeown. 2019. Fine-grained sentiment analysis with faithful attention. *ArXiv*, abs/1908.06870.

A Appendices

A.1 On the choice of distance function

The purpose of this section is to inspect the impact of selecting different distance metrics when computing Value Zeroing in Eq. 10. Timkey and van Schijndel (2021) questioned the informativity of standard representational distance measures such as cosine and Euclidean by observing that only a small subset of rogue dimensions contribute to the anisotropy of a contextualized representation space. They proposed using simple post-processing techniques to correct for such these rough dimensions. We followed their suggestion and normalized the representations before computing distances, but we did not observe any noticeable difference in our scores compared to using non-normalized representations (Fig. A.1). We also repeated our experiment with Spearman's and Euclidean distance metrics and observed the same pattern in the results (Fig. A.1).

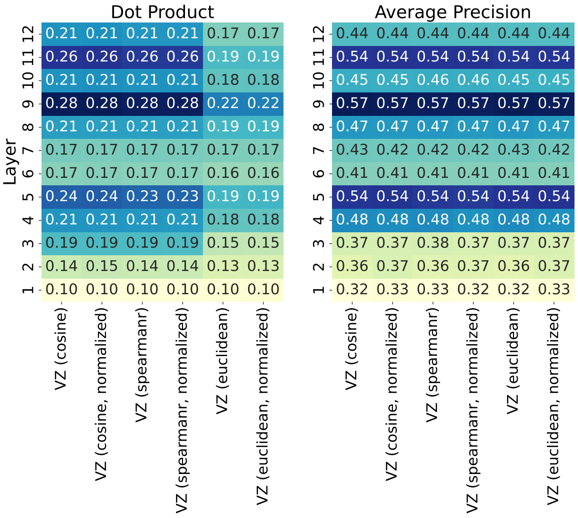
We believe that in anisotropy studies that use clustering methods, the choice of distance metrics is crucial. However, we compute each token's distance from itself (not from other tokens) and compare them relatively. This might explain why we observe the same pattern for different distance metrics.

A.2 More metrics

Figure A.2 reports the cue alignment evaluation for BERT model based on Probes-needed (Zhong et al., 2019) metric.

A.3 More PLMs

We replicated our experiment for the cue alignment for two more PLMs; RoBERTa (Liu et al., 2019) and ELECTRA (generator, Clark et al., 2020). As we can see in Figures A.3 and A.4, our method consistently outperforms other methods on all models in both pre-trained and fine-tuned setups. Due to the fact that our scores are based on zeroing value vectors, our method can be easily applied to any Transformer-based models even with different modalities.



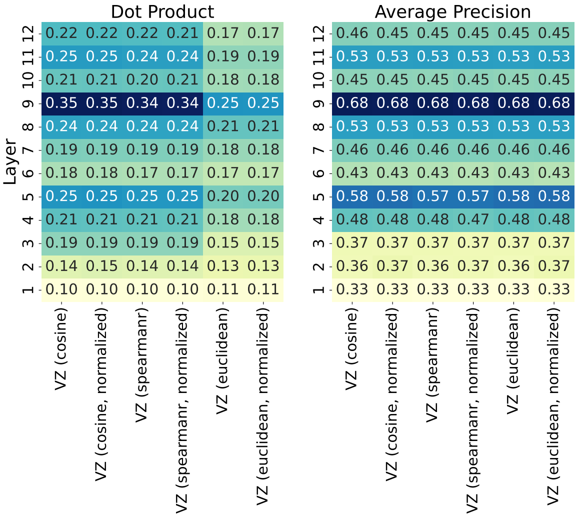
Dot Product

Layer	VZ (cosine)	VZ (cosine, normalized)	VZ (spearmanr)	VZ (spearmanr, normalized)	VZ (euclidean)	VZ (euclidean, normalized)
12	0.21	0.21	0.21	0.21	0.17	0.17
11	0.26	0.26	0.26	0.26	0.19	0.19
10	0.21	0.21	0.21	0.21	0.18	0.18
9	0.28	0.28	0.28	0.28	0.22	0.22
8	0.21	0.21	0.21	0.21	0.19	0.19
7	0.17	0.17	0.17	0.17	0.17	0.17
6	0.17	0.17	0.17	0.17	0.16	0.16
5	0.24	0.24	0.23	0.23	0.19	0.19
4	0.21	0.21	0.21	0.21	0.18	0.18
3	0.19	0.19	0.19	0.19	0.15	0.15
2	0.14	0.15	0.14	0.14	0.13	0.13
1	0.10	0.10	0.10	0.10	0.10	0.10

Average Precision

Layer	VZ (cosine)	VZ (cosine, normalized)	VZ (spearmanr)	VZ (spearmanr, normalized)	VZ (euclidean)	VZ (euclidean, normalized)
12	0.44	0.44	0.44	0.44	0.44	0.44
11	0.54	0.54	0.54	0.54	0.54	0.54
10	0.45	0.45	0.46	0.46	0.45	0.45
9	0.57	0.57	0.57	0.57	0.57	0.57
8	0.47	0.47	0.47	0.47	0.47	0.47
7	0.43	0.42	0.42	0.42	0.43	0.42
6	0.41	0.41	0.41	0.41	0.41	0.41
5	0.54	0.54	0.54	0.54	0.54	0.54
4	0.48	0.48	0.48	0.48	0.48	0.48
3	0.37	0.37	0.38	0.37	0.37	0.37
2	0.36	0.37	0.36	0.37	0.36	0.37
1	0.32	0.33	0.33	0.32	0.32	0.33

(a) Pre-trained BERT



Dot Product

Layer	VZ (cosine)	VZ (cosine, normalized)	VZ (spearmanr)	VZ (spearmanr, normalized)	VZ (euclidean)	VZ (euclidean, normalized)
12	0.22	0.22	0.22	0.21	0.17	0.17
11	0.25	0.25	0.24	0.24	0.19	0.19
10	0.21	0.21	0.20	0.21	0.18	0.18
9	0.35	0.35	0.34	0.34	0.25	0.25
8	0.24	0.24	0.24	0.24	0.21	0.21
7	0.19	0.19	0.19	0.19	0.18	0.18
6	0.18	0.18	0.17	0.17	0.17	0.17
5	0.25	0.25	0.25	0.25	0.20	0.20
4	0.21	0.21	0.21	0.21	0.18	0.18
3	0.19	0.19	0.19	0.19	0.15	0.15
2	0.14	0.15	0.14	0.14	0.13	0.13
1	0.10	0.10	0.10	0.10	0.11	0.11

Average Precision

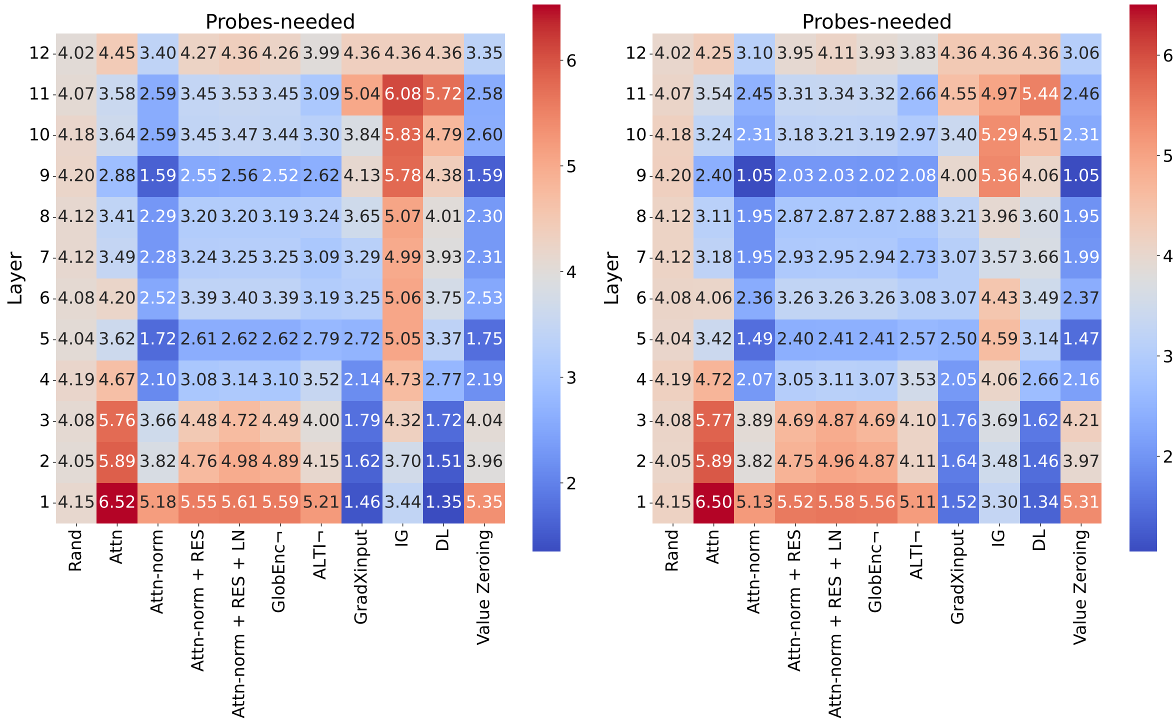
Layer	VZ (cosine)	VZ (cosine, normalized)	VZ (spearmanr)	VZ (spearmanr, normalized)	VZ (euclidean)	VZ (euclidean, normalized)
12	0.46	0.45	0.45	0.45	0.45	0.45
11	0.53	0.53	0.53	0.53	0.53	0.53
10	0.45	0.45	0.45	0.45	0.45	0.45
9	0.68	0.68	0.68	0.68	0.68	0.68
8	0.53	0.53	0.53	0.53	0.53	0.53
7	0.46	0.46	0.46	0.46	0.46	0.46
6	0.43	0.43	0.43	0.43	0.43	0.43
5	0.58	0.58	0.57	0.57	0.58	0.58
4	0.48	0.48	0.48	0.47	0.48	0.48
3	0.37	0.37	0.37	0.37	0.37	0.37
2	0.36	0.37	0.36	0.37	0.36	0.37
1	0.33	0.33	0.33	0.33	0.33	0.33

(b) Fine-tuned BERT

Figure A.1: Layer-wise alignment between the cue vector and different *Value Zeroing* (VZ) scores computed based on 1) different distance metric and 2) whether representations are normalized.

A.4 Qualitative Analysis: Layer-wise Context Mixing Maps

This section illustrates different context mixing maps obtained from a fine-tuned BERT model for the correctly classified example of *"The pictures of some hat [MASK] scaring Marcus."*



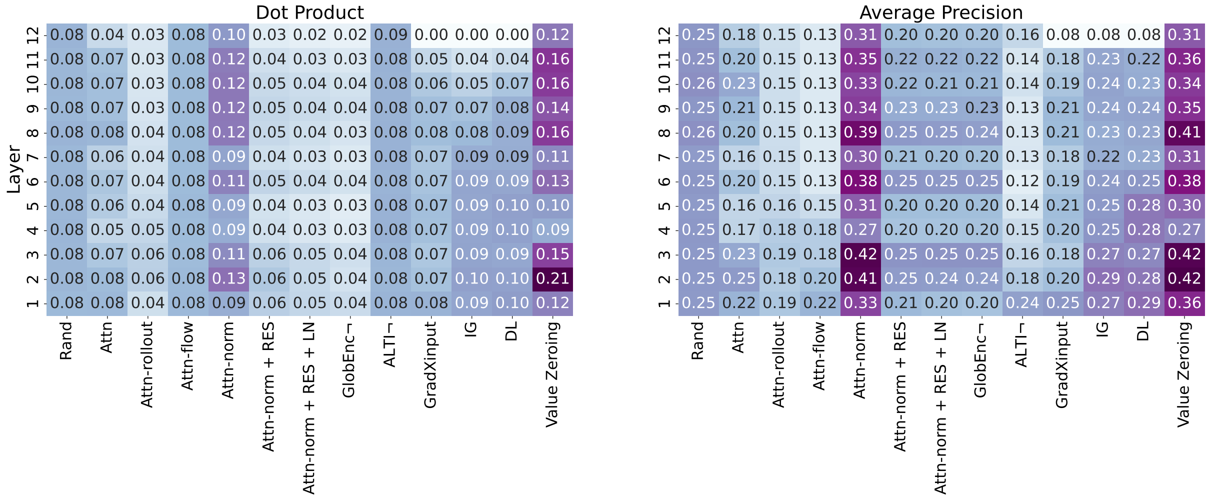
Layer	Rand	Attn	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	4.02	4.45	3.40	4.27	4.36	4.26	3.99	4.36	4.36	4.36	3.35
11	4.07	3.58	2.59	3.45	3.53	3.45	3.09	5.04	6.08	5.72	2.58
10	4.18	3.64	2.59	3.45	3.47	3.44	3.30	3.84	5.83	4.79	2.60
9	4.20	2.88	1.59	2.55	2.56	2.52	2.62	4.13	5.78	4.38	1.59
8	4.12	3.41	2.29	3.20	3.20	3.19	3.24	3.65	5.07	4.01	2.30
7	4.12	3.49	2.28	3.24	3.25	3.25	3.09	3.29	4.99	3.93	2.31
6	4.08	4.20	2.52	3.39	3.40	3.39	3.19	3.25	5.06	3.75	2.53
5	4.04	3.62	1.72	2.61	2.62	2.62	2.79	2.72	5.05	3.37	1.75
4	4.19	4.67	2.10	3.08	3.14	3.10	3.52	2.14	4.73	2.77	2.19
3	4.08	5.76	3.66	4.48	4.72	4.49	4.00	1.79	4.32	1.72	4.04
2	4.05	5.89	3.82	4.76	4.98	4.89	4.15	1.62	3.70	1.51	3.96
1	4.15	6.52	5.18	5.55	5.61	5.59	5.21	1.46	3.44	1.35	5.35

(a) Pre-trained BERT

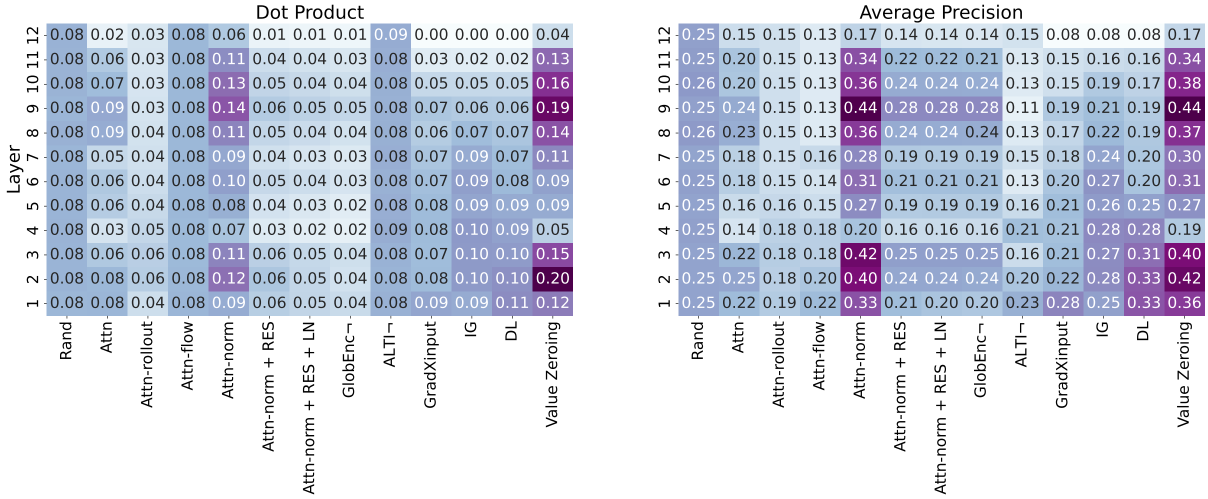
Layer	Rand	Attn	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	4.02	4.25	3.10	3.95	4.11	3.93	3.83	4.36	4.36	4.36	3.06
11	4.07	3.54	2.45	3.31	3.34	3.32	2.66	4.55	4.97	5.44	2.46
10	4.18	3.24	2.31	3.18	3.21	3.19	2.97	3.40	5.29	4.51	2.31
9	4.20	2.40	1.05	2.03	2.03	2.02	2.08	4.00	5.36	4.06	1.05
8	4.12	3.11	1.95	2.87	2.87	2.87	2.88	3.21	3.96	3.60	1.95
7	4.12	3.18	1.95	2.93	2.95	2.94	2.73	3.07	3.57	3.66	1.99
6	4.08	4.06	2.36	3.26	3.26	3.26	3.08	3.07	4.43	3.49	2.37
5	4.04	3.42	1.49	2.40	2.41	2.41	2.57	2.50	4.59	3.14	1.47
4	4.19	4.72	2.07	3.05	3.11	3.07	3.53	2.05	4.06	2.66	2.16
3	4.08	5.77	3.89	4.69	4.87	4.69	4.10	1.76	3.69	1.62	4.21
2	4.05	5.89	3.82	4.75	4.96	4.87	4.11	1.64	3.48	1.46	3.97
1	4.15	6.50	5.13	5.52	5.58	5.56	5.11	1.52	3.30	1.34	5.31

(b) Fine-tuned BERT

Figure A.2: The layer-wise alignment based on Probes-needed metric between the cue vector and different analysis methods averaged over Test set examples. Lower value (darker blue) is better.

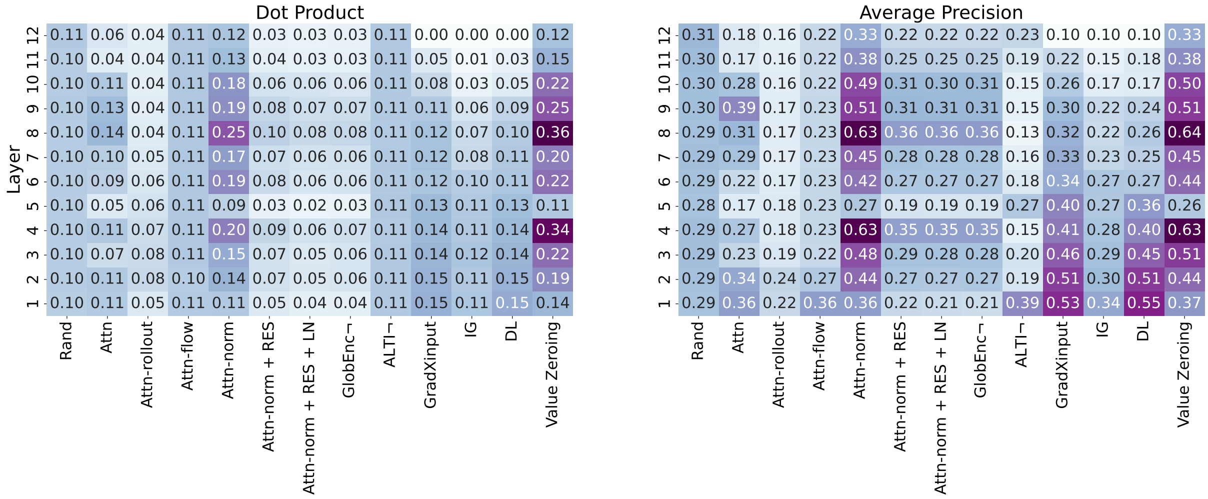


(a) Pre-trained RoBERTa



(b) Fine-tuned RoBERTa

Figure A.3: Layer-wise alignment between the cue vector and different analysis methods averaged over Test set examples for RoBERTa. Higher value (darker color) is better.



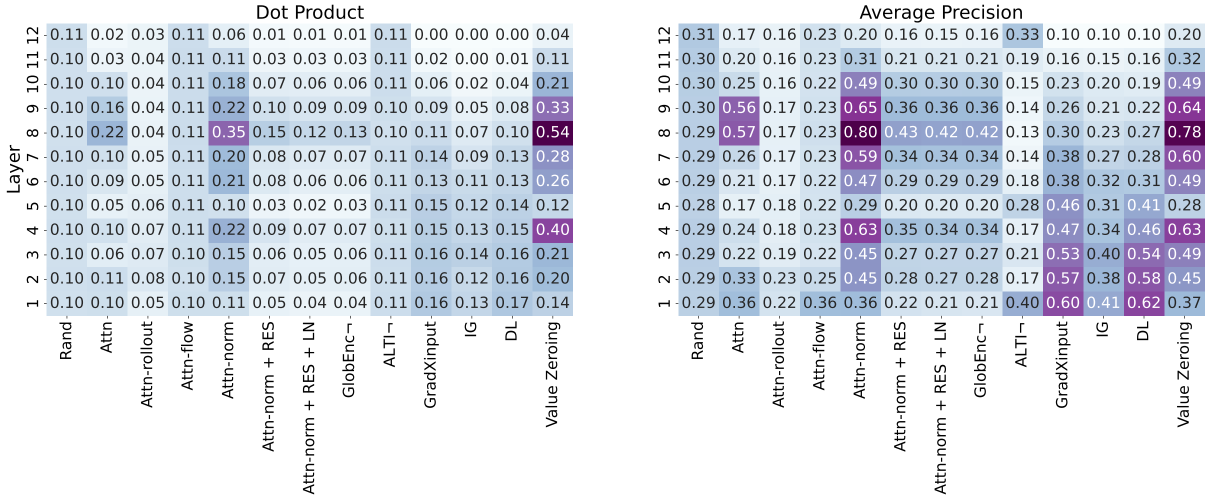
Dot Product

Layer	Rand	Attn	Attn-rollout	Attn-flow	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.11	0.06	0.04	0.11	0.12	0.03	0.03	0.03	0.11	0.00	0.00	0.00	0.12
11	0.10	0.04	0.04	0.11	0.13	0.04	0.03	0.03	0.11	0.05	0.01	0.03	0.15
10	0.10	0.11	0.04	0.11	0.18	0.06	0.06	0.06	0.11	0.08	0.03	0.05	0.22
9	0.10	0.13	0.04	0.11	0.19	0.08	0.07	0.07	0.11	0.11	0.06	0.09	0.25
8	0.10	0.14	0.04	0.11	0.25	0.10	0.08	0.08	0.11	0.12	0.07	0.10	0.36
7	0.10	0.10	0.05	0.11	0.17	0.07	0.06	0.06	0.11	0.12	0.08	0.11	0.20
6	0.10	0.09	0.06	0.11	0.19	0.08	0.06	0.06	0.11	0.12	0.10	0.11	0.22
5	0.10	0.05	0.06	0.11	0.09	0.03	0.02	0.03	0.11	0.13	0.11	0.13	0.11
4	0.10	0.11	0.07	0.11	0.20	0.09	0.06	0.07	0.11	0.14	0.11	0.14	0.34
3	0.10	0.07	0.08	0.11	0.15	0.07	0.05	0.06	0.11	0.14	0.12	0.14	0.22
2	0.10	0.11	0.08	0.10	0.14	0.07	0.05	0.06	0.11	0.15	0.11	0.15	0.19
1	0.10	0.11	0.05	0.11	0.11	0.05	0.04	0.04	0.11	0.15	0.11	0.15	0.14

Average Precision

Layer	Rand	Attn	Attn-rollout	Attn-flow	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.31	0.18	0.16	0.22	0.33	0.22	0.22	0.22	0.23	0.10	0.10	0.10	0.33
11	0.30	0.17	0.16	0.22	0.38	0.25	0.25	0.25	0.19	0.22	0.15	0.18	0.38
10	0.30	0.28	0.16	0.22	0.49	0.31	0.30	0.31	0.15	0.26	0.17	0.17	0.50
9	0.30	0.39	0.17	0.23	0.51	0.31	0.31	0.31	0.15	0.30	0.22	0.24	0.51
8	0.29	0.31	0.17	0.23	0.63	0.36	0.36	0.36	0.13	0.32	0.22	0.26	0.64
7	0.29	0.29	0.17	0.23	0.45	0.28	0.28	0.28	0.16	0.33	0.23	0.25	0.45
6	0.29	0.22	0.17	0.23	0.42	0.27	0.27	0.27	0.18	0.34	0.27	0.27	0.44
5	0.28	0.17	0.18	0.23	0.27	0.19	0.19	0.19	0.27	0.40	0.27	0.36	0.26
4	0.29	0.27	0.18	0.23	0.63	0.35	0.35	0.35	0.15	0.41	0.28	0.40	0.63
3	0.29	0.23	0.19	0.22	0.48	0.29	0.28	0.28	0.20	0.46	0.29	0.45	0.51
2	0.29	0.34	0.24	0.27	0.44	0.27	0.27	0.27	0.19	0.51	0.30	0.51	0.44
1	0.29	0.36	0.22	0.36	0.36	0.22	0.21	0.21	0.39	0.53	0.34	0.55	0.37

(a) Pre-trained ELECTRA



Dot Product

Layer	Rand	Attn	Attn-rollout	Attn-flow	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.11	0.02	0.03	0.11	0.06	0.01	0.01	0.01	0.11	0.00	0.00	0.00	0.04
11	0.10	0.03	0.04	0.11	0.11	0.03	0.03	0.03	0.11	0.02	0.00	0.01	0.11
10	0.10	0.10	0.04	0.11	0.18	0.07	0.06	0.06	0.11	0.06	0.02	0.04	0.21
9	0.10	0.16	0.04	0.11	0.22	0.10	0.09	0.09	0.10	0.09	0.05	0.08	0.33
8	0.10	0.22	0.04	0.11	0.35	0.15	0.12	0.13	0.10	0.11	0.07	0.10	0.54
7	0.10	0.10	0.05	0.11	0.20	0.08	0.07	0.07	0.11	0.14	0.09	0.13	0.28
6	0.10	0.09	0.05	0.11	0.21	0.08	0.06	0.06	0.11	0.13	0.11	0.13	0.26
5	0.10	0.05	0.06	0.11	0.10	0.03	0.02	0.03	0.11	0.15	0.12	0.14	0.12
4	0.10	0.10	0.07	0.11	0.22	0.09	0.07	0.07	0.11	0.15	0.13	0.15	0.40
3	0.10	0.06	0.07	0.10	0.15	0.06	0.05	0.06	0.11	0.16	0.14	0.16	0.21
2	0.10	0.11	0.08	0.10	0.15	0.07	0.05	0.06	0.11	0.16	0.12	0.16	0.20
1	0.10	0.10	0.05	0.10	0.11	0.05	0.04	0.04	0.11	0.16	0.13	0.17	0.14

Average Precision

Layer	Rand	Attn	Attn-rollout	Attn-flow	Attn-norm	Attn-norm + RES	Attn-norm + RES + LN	GlobEnc¬	ALTI¬	GradXinput	IG	DL	Value Zeroing
12	0.31	0.17	0.16	0.23	0.20	0.16	0.15	0.16	0.33	0.10	0.10	0.10	0.20
11	0.30	0.20	0.16	0.23	0.31	0.21	0.21	0.21	0.19	0.16	0.15	0.16	0.32
10	0.30	0.25	0.16	0.22	0.49	0.30	0.30	0.30	0.15	0.23	0.20	0.19	0.49
9	0.30	0.56	0.17	0.23	0.65	0.36	0.36	0.36	0.14	0.26	0.21	0.22	0.64
8	0.29	0.57	0.17	0.23	0.80	0.43	0.42	0.42	0.13	0.30	0.23	0.27	0.78
7	0.29	0.26	0.17	0.23	0.59	0.34	0.34	0.34	0.14	0.38	0.27	0.28	0.60
6	0.29	0.21	0.17	0.22	0.47	0.29	0.29	0.29	0.18	0.38	0.32	0.31	0.49
5	0.28	0.17	0.18	0.22	0.29	0.20	0.20	0.20	0.28	0.46	0.31	0.41	0.28
4	0.29	0.24	0.18	0.23	0.63	0.35	0.34	0.34	0.17	0.47	0.34	0.46	0.63
3	0.29	0.22	0.19	0.22	0.45	0.27	0.27	0.27	0.21	0.53	0.40	0.54	0.49
2	0.29	0.33	0.23	0.25	0.45	0.28	0.27	0.28	0.17	0.57	0.38	0.58	0.45
1	0.29	0.36	0.22	0.36	0.36	0.22	0.21	0.21	0.40	0.60	0.41	0.62	0.37

(b) Fine-tuned ELECTRA

Figure A.4: Layer-wise alignment between the cue vector and different analysis methods averaged over Test set examples for ELECTRA. Higher value (darker color) is better.

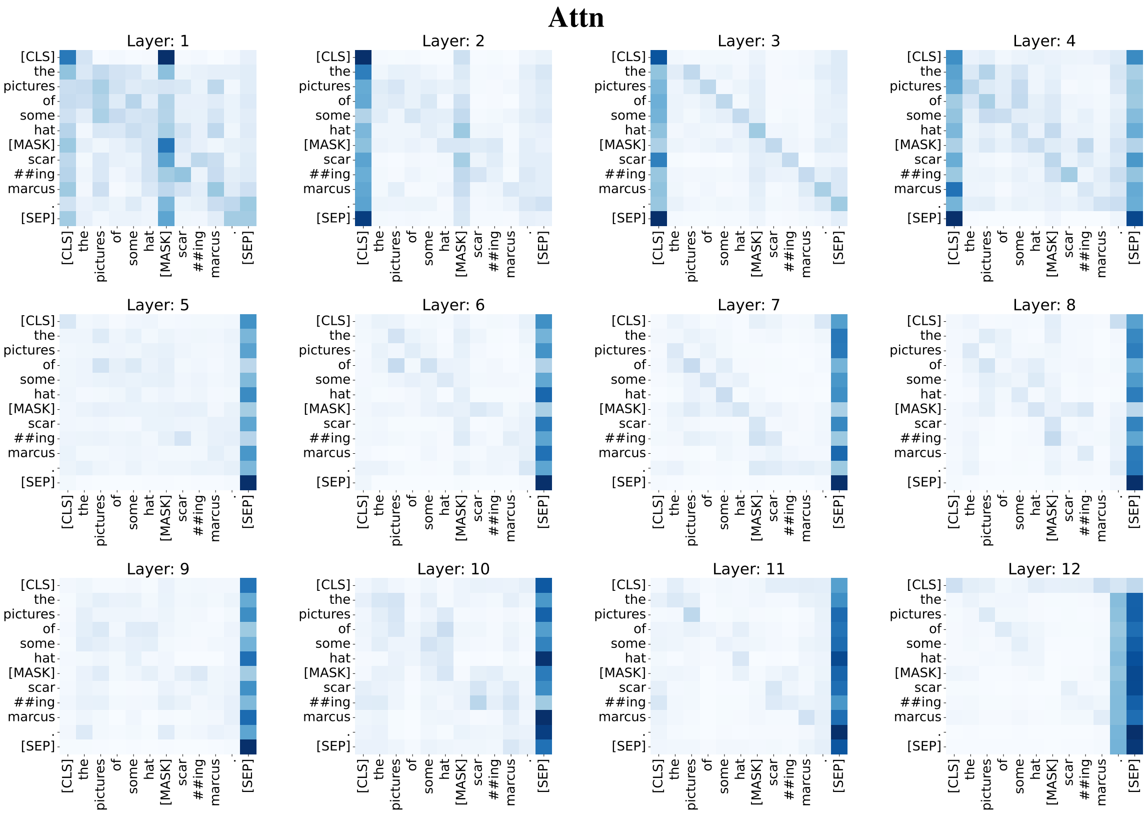


Figure A.5: Raw attention scores (Attn) averaged over all attention heads at each different layer.

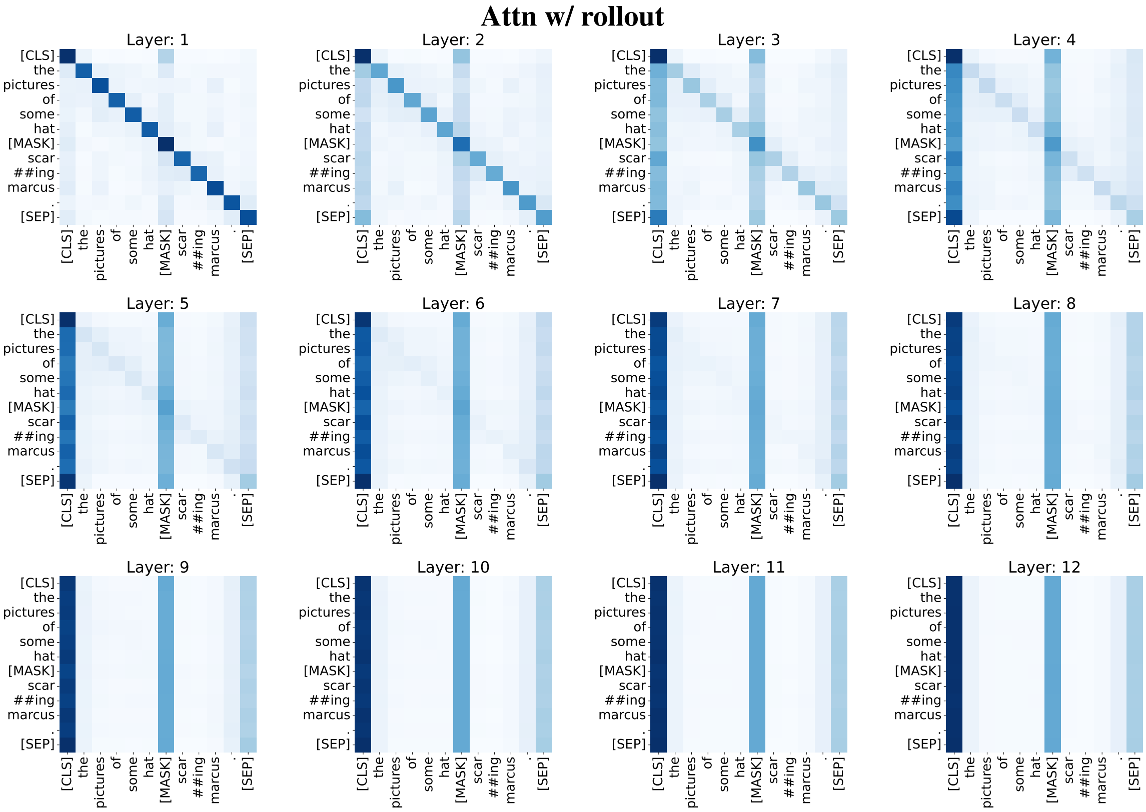


Figure A.6: Raw attention scores (Attn) aggregated by rollout (Abnar and Zuidema (2020)'s method) across layers.

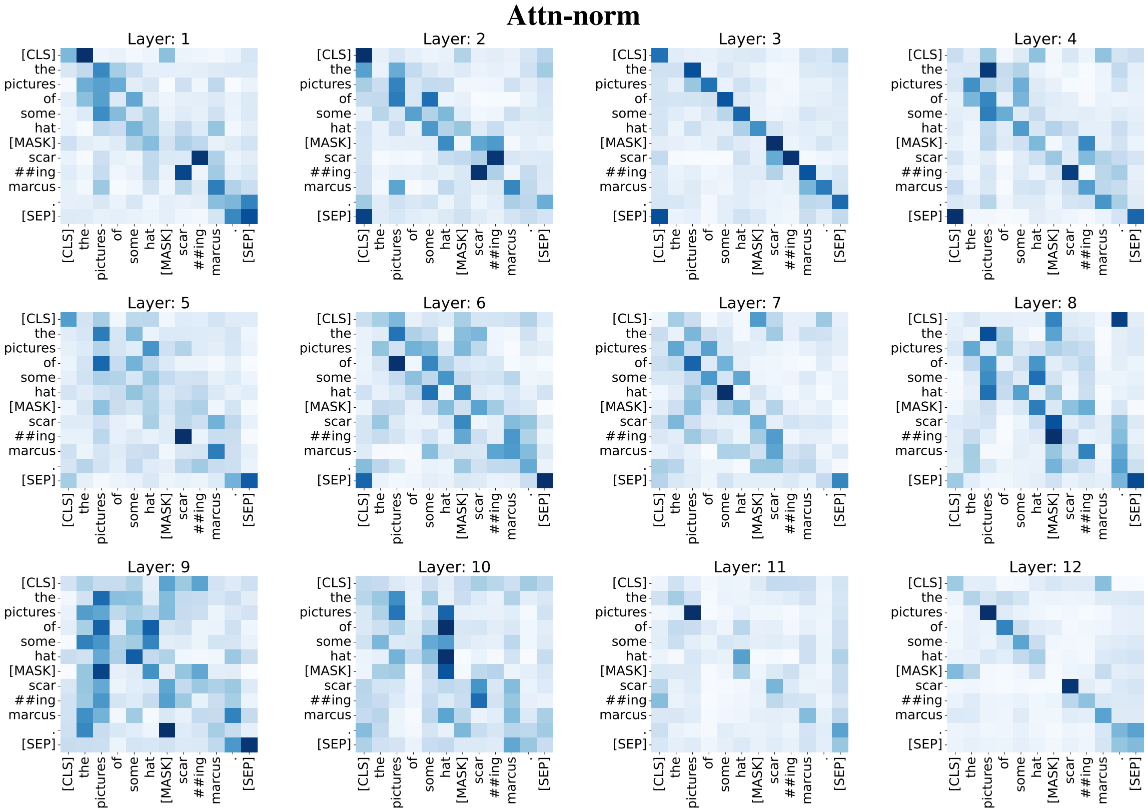


Figure A.7: Kobayashi et al. (2020)'s scores (Attn-norm) across layers.

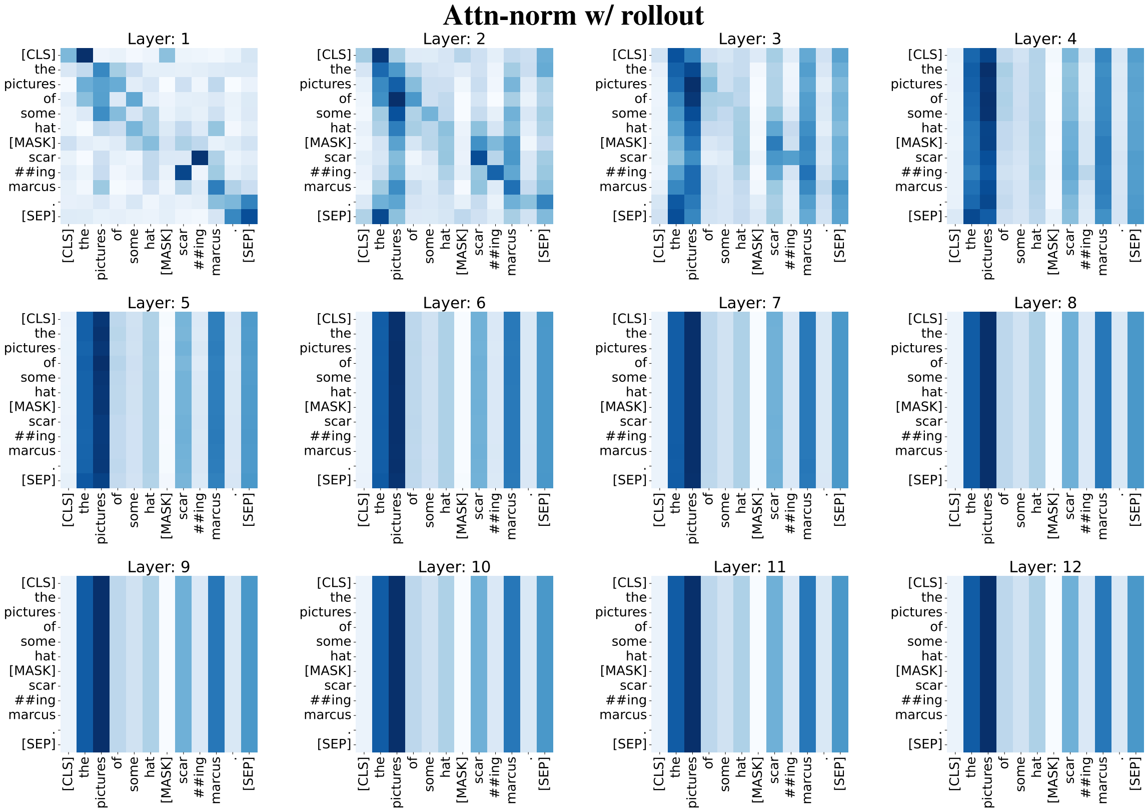


Figure A.8: Kobayashi et al. (2020)'s scores (Attn-norm) aggregated by rollout method across layers.

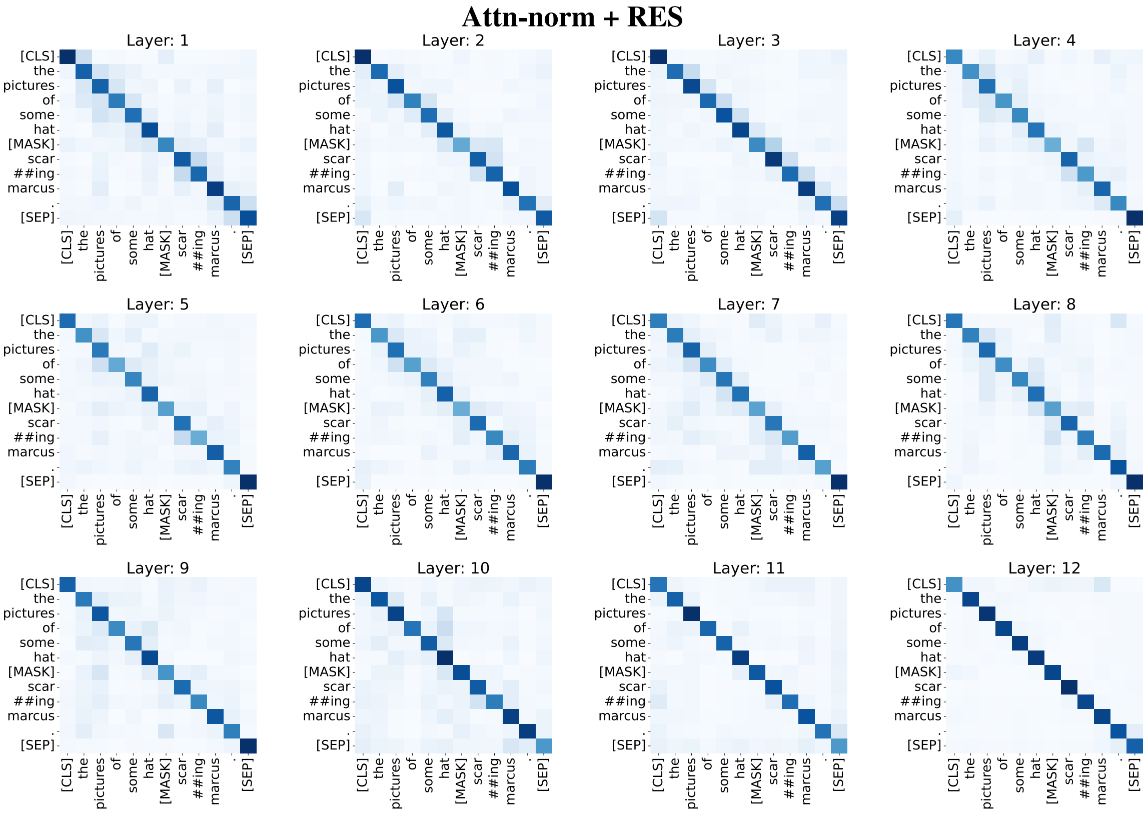


Figure A.9: Kobayashi et al. (2021)'s scores (Attn-norm + RES) across layers.

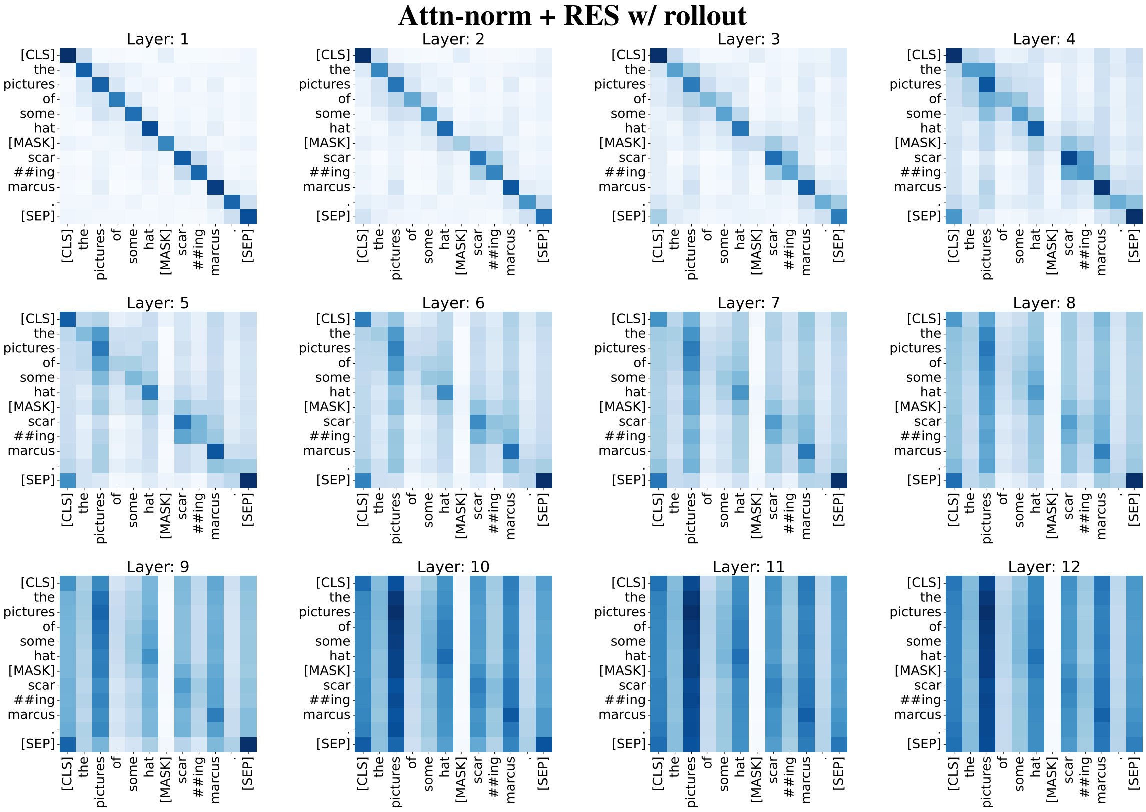


Figure A.10: Kobayashi et al. (2021)'s scores (Attn-norm + RES) aggregated by rollout method across layers.

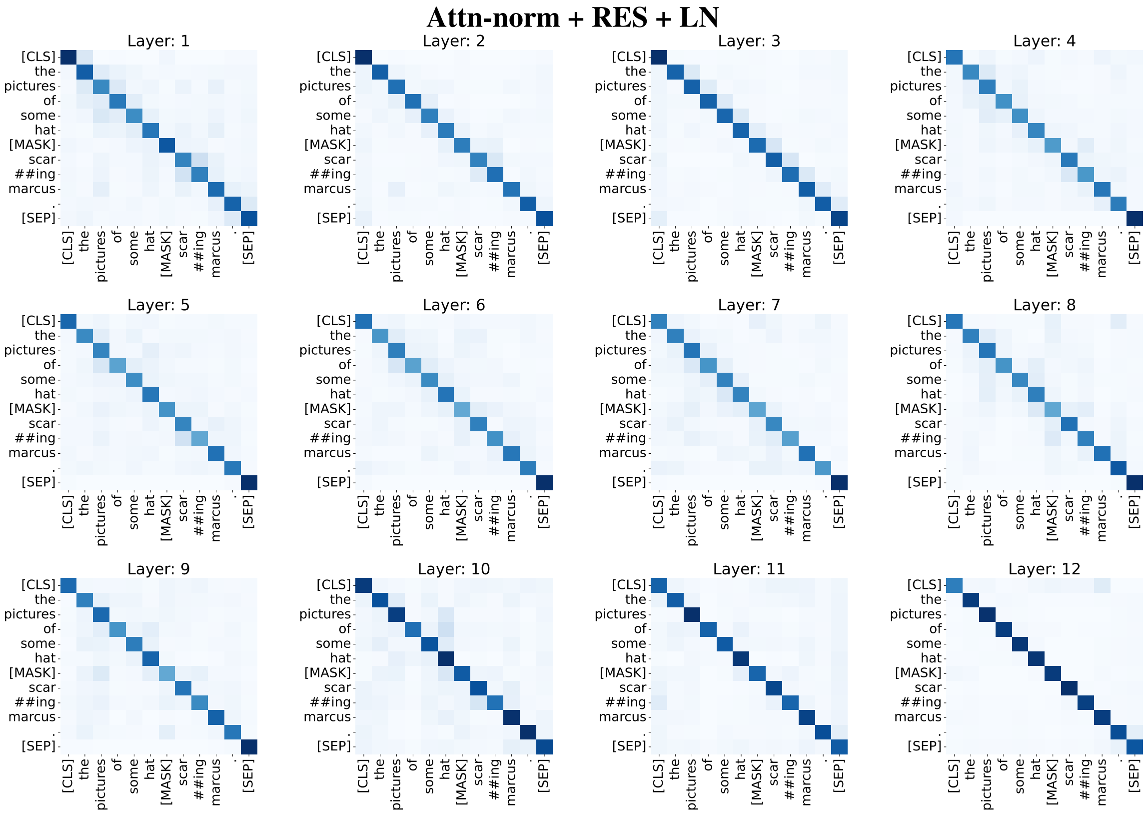


Figure A.11: Kobayashi et al. (2021)'s scores (Attn-norm + RES + LN) across layers.

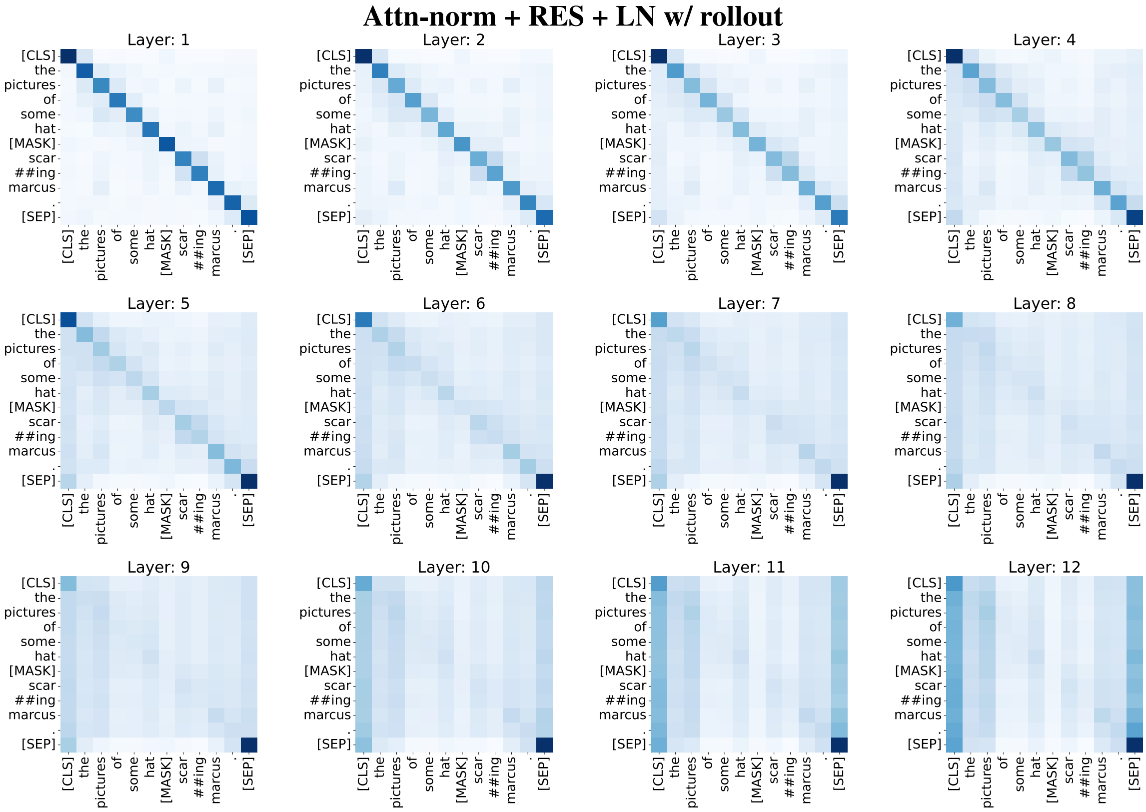


Figure A.12: Kobayashi et al. (2021)'s scores (Attn-norm + RES + LN) aggregated by rollout method across layers.

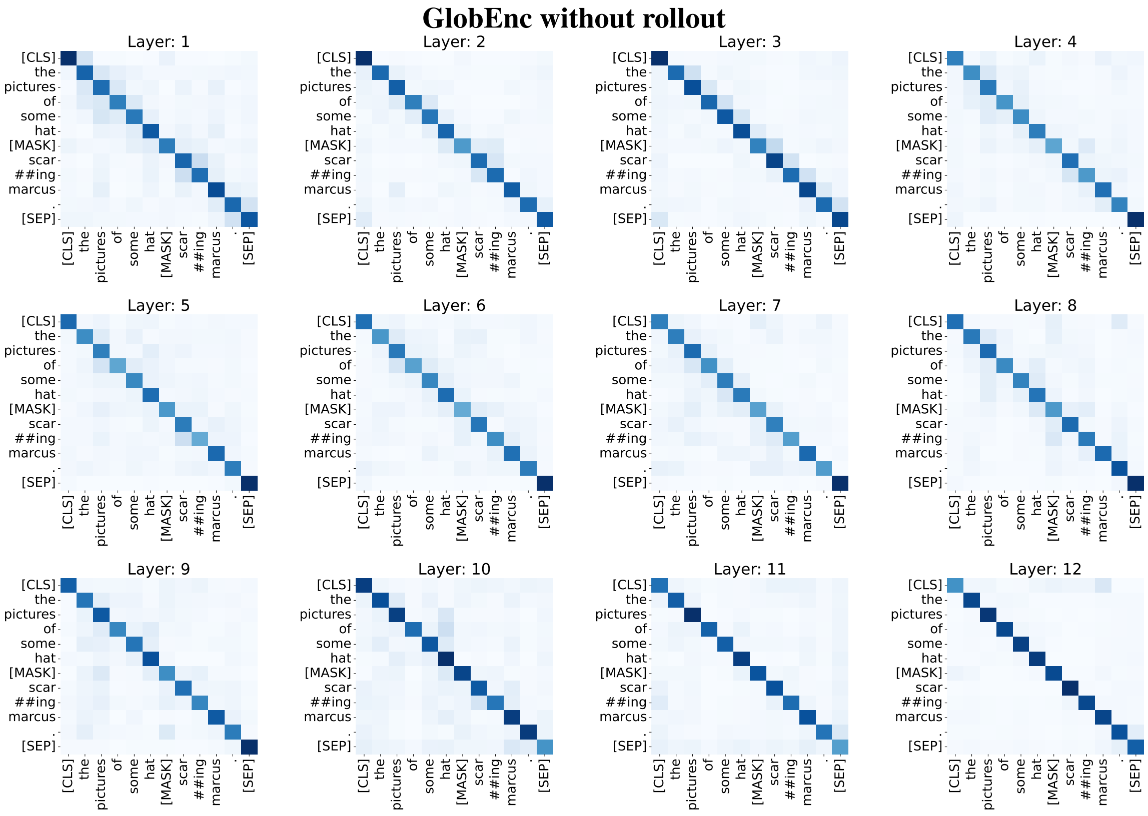

GlobEnc without rollout

Figure A.13: Modarressi et al. (2022)'s scores (GlobEnc¬) without aggregation (rollout) across layers.

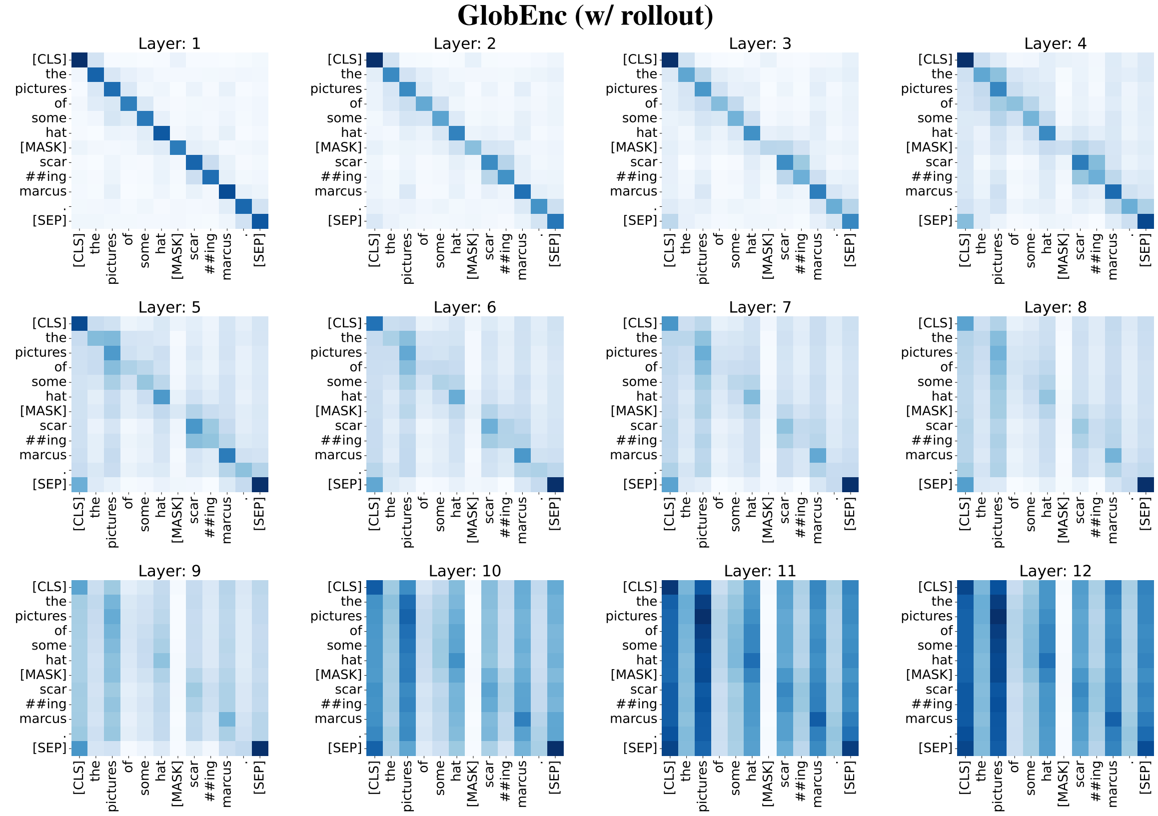

GlobEnc (w/ rollout)

Figure A.14: Modarressi et al. (2022)'s scores (GlobEnc) across layers (the rollout method is inherently incorporated in GlobEnc).

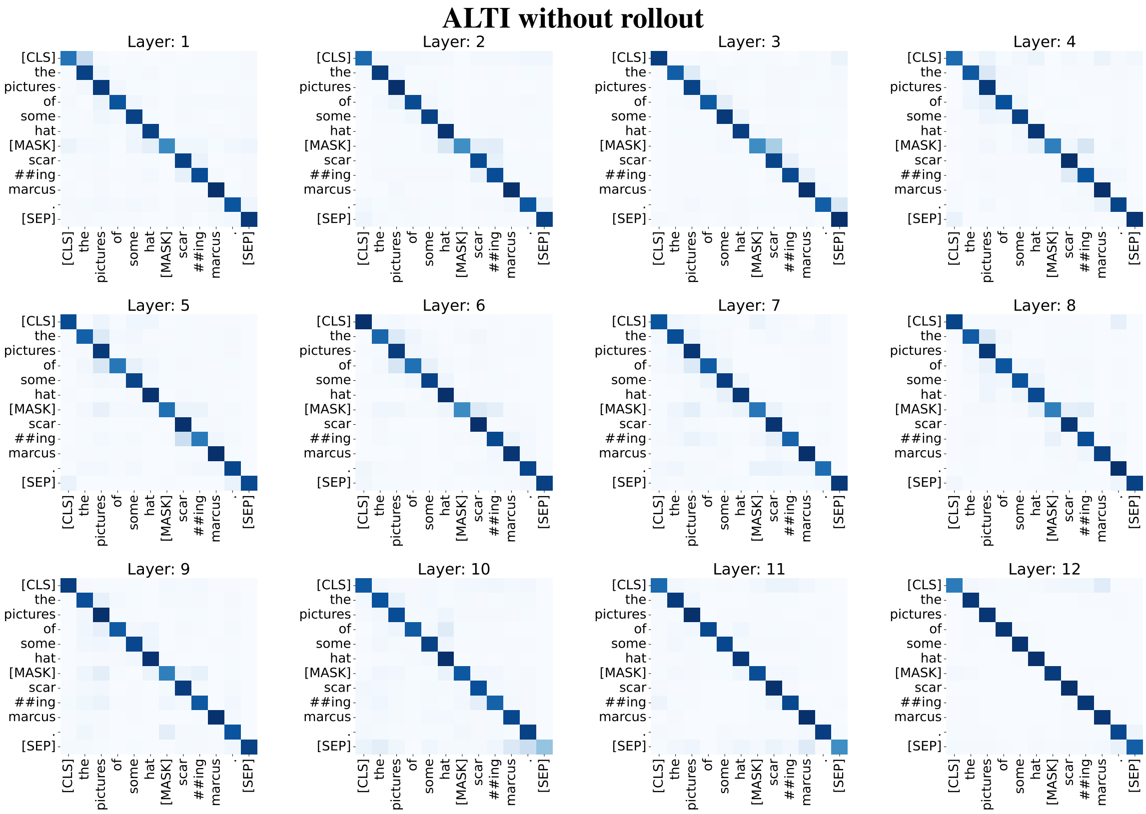


Figure A.15: Ferrando et al. (2022)'s scores (ALTI¬) without aggregation (rollout) across layers.

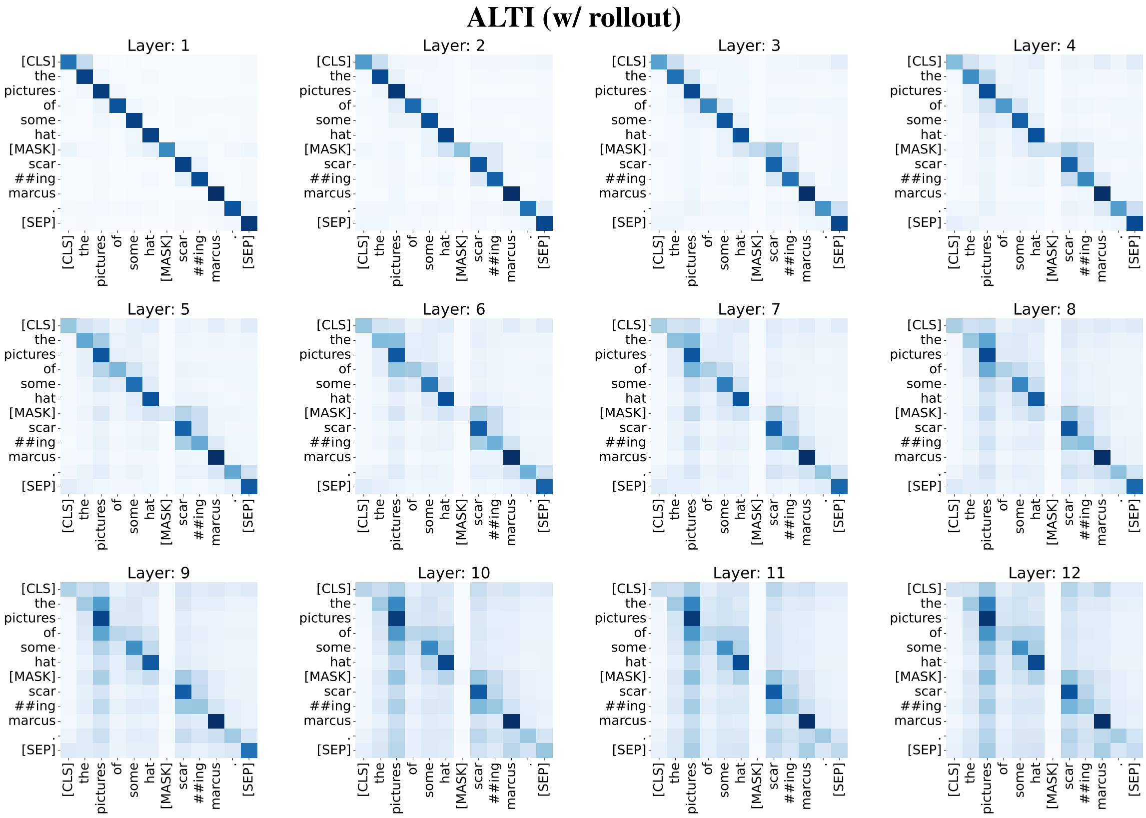


Figure A.16: Ferrando et al. (2022)'s scores (ALTI) across layers (the rollout method is inherently incorporated in ALTI).

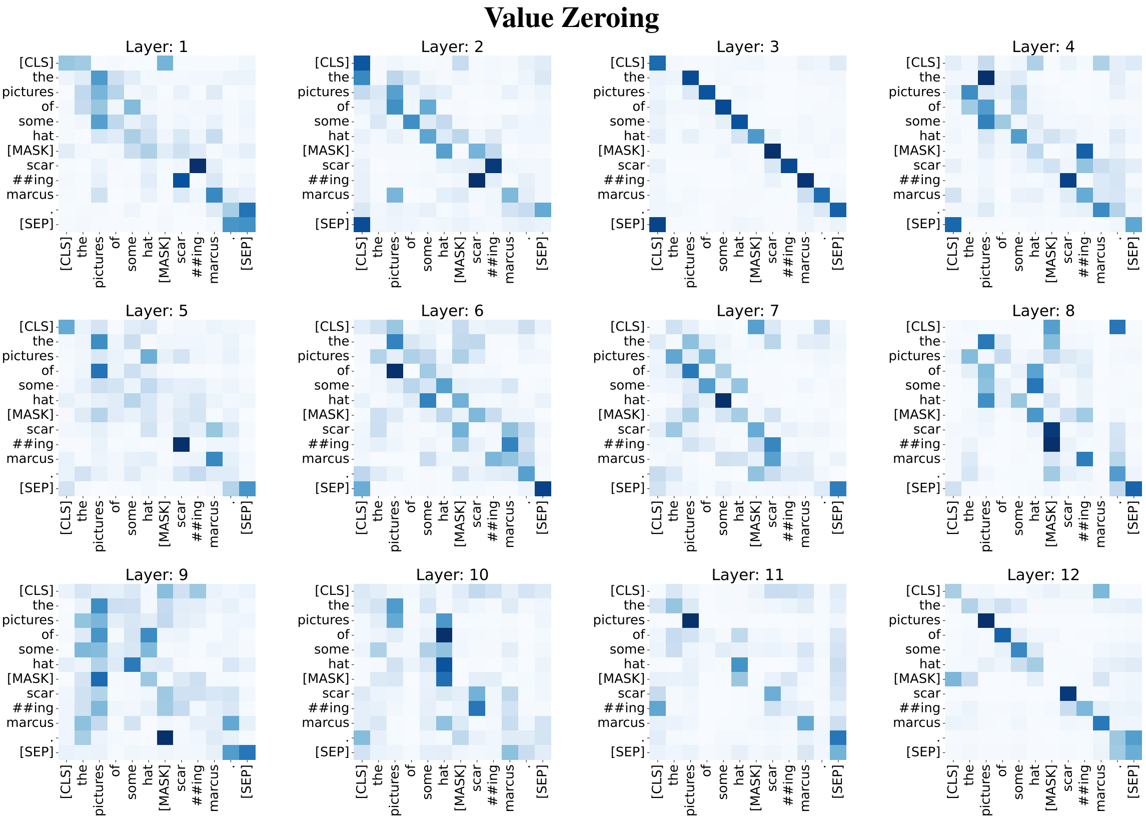


Figure A.17: Our scores (*Value Zeroing*) across layers.

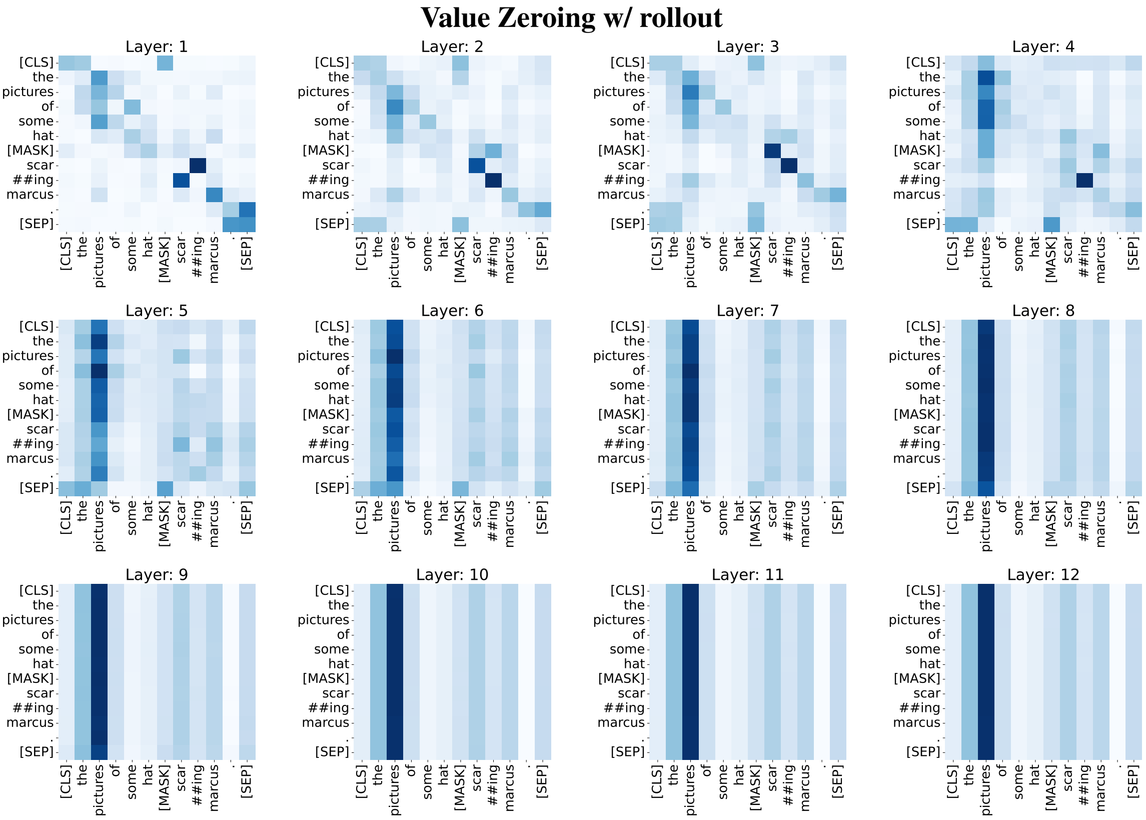


Figure A.18: Our global scores (*Value Zeroing*) aggregated by rollout method across layers.

KGVL-BART: Knowledge Graph Augmented Visual Language BART for Radiology Report Generation

Kaveri Kale, Pushpak Bhattacharyya
IIT Bombay
{kaverikale,pb}@cse.iitb.ac.in

Milind Gune
Augnito India Pvt Ltd
dgune@rediffmail.com

Aditya Shetty
Breach Candy Hospital, India
adityashetty01@gmail.com

Rustom Lawyer
Augnito India Pvt Ltd
rustom@augnito.ai

Abstract

Timely generation of radiology reports and diagnoses is a challenge worldwide due to the enormous number of cases and shortage of radiology specialists. In this paper, we propose a **K**nowledge **G**raph **A**ugmented **V**ision **L**anguage **BART** (KGVL-BART) model that takes as input two chest X-ray images- one frontal and the other lateral- along with *tags* which are diagnostic keywords, and outputs a report with the patient-specific findings. Our system development effort is divided into 3 stages: i) construction of the Chest X-ray KG (referred to as chestX-KG), ii) image feature extraction, and iii) training a KGVL-BART model using the visual, text, and KG data. The dataset we use is the well-known Indiana University Chest X-ray reports with the train, validation, and test split of 3025 instances, 300 instances, and 500 instances respectively. We construct a Chest X-Ray knowledge graph from these reports by extracting entity1-relation-entity2 triples; the triples get extracted by a rule-based tool of our own. Constructed KG is verified by two experienced radiologists (with experience of 30 years and 8 years, respectively). We demonstrate that our model- KGVL-BART- outperforms State-of-the-Art transformer-based models on standard NLG scoring metrics. We also include a qualitative evaluation of our system by experienced radiologist (with experience of 30 years) on the test data, which showed that 73% of the reports generated were fully correct, only 5.5% are completely wrong and 21.5% have important missing details though overall correct. To the best of our knowledge, ours is the first system to make use of multi-modality and domain knowledge to generate X-ray reports automatically.

1 Introduction

Medical imaging techniques are widely used in hospitals across the world. The detailed information extracted from medical images is crucial for proper diagnosis and treatment. An experienced and skilled radiologist is required to prepare an accurate full text diagnostic report. However, due to a lack of experts, many reports contain indecisive findings, forcing patients to undergo further tests involving pathology or other advanced imaging methods. In addition, the time consuming process of full text radiology report generation is one of the biggest challenges. All over the world, the ratio of radiologists to patients is very low. The ratios in the US, China, and India are 1:10,000, 1:14,772, and 1:100,000 respectively (Arora, 2014). Given the huge number of cases and the shortage of radiology experts, timely report generation and diagnosis is a huge challenge worldwide.

In the case of a chest ailment, X-rays produce images of chest organs like, lungs, spinal bones, heart, airways, and blood vessels. These images help doctors ascertain an exact findings such as pneumonia, collapsed lung, emphysema, cancer, broken ribs, *etc*. Manual examination of X-ray images for a large number of patients can be time consuming, leading to delays. Human errors may further add to the challenges. This prompted the researchers to use deep learning models capable of automated report generation to address the above mentioned challenges. With deep learning based automatic report generation, the reports can be generated with minimal delay and free from any human errors. Large-scale pre-trained language models have recently expanded into multimodality learning, improving representations by combining visual and semantic features (Cho et al., 2021; Sollami and Jain, 2021; Mustafa et al., 2022). However, progress in adapting language models toward conditional Natural Language Generation (NLG) is limited to image and text modalities. The Natural Language Processing (NLP) community is moving towards transformer-based models. The major-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3401–3411
May 2-6, 2023 ©2023 Association for Computational Linguistics

ity of NLP research today produces better results by tweaking an already-trained transformer model across a large corpus. This prompts us to research pre-trained transformers like BART that have generative capabilities while conditioning them on visual, textual, and KG data. This study introduces the KGVL-BART conditioned transformer-based model, which, produces a comprehensive report given an X-ray image, tags associated with X-ray image.)), and chest X-ray KG. We train our model using the publicly accessible Indiana University chest X-ray dataset (referred to as IU-XRay) (Fischer et al., 2022). The encoder accepts input from tags, images, and KG in three different modalities. We create embeddings for each of these modalities before sending input to the multimodal encoder. We compare our quantitative findings to earlier transformer-based models. Furthermore, we provide qualitative analysis on test set by a radiologist.

Problem Statement: Design a system that generates a structured patient-specific report from radiology images, image tags and domain knowledge. The input to the system is

- two chest X-ray images- one frontal and the other lateral
- tags

The output of the system is

- radiology report with patient-specific findings

Domain knowledge comes from the Knowledge Graph.

Our **contributions** are:

1. A knowledge-enhanced BART-based Vision-Language Model which we call KGVL-BART and which generates chest X-ray reports with accuracy better than SoTA.
2. Chest X-ray knowledge graph created from IU Chest X-ray reports.
3. Demonstration of the fact that multi-modality helps in radiology report generation; we use both the image and its tags plus triples from the knowledge graph.

2 Fundamental Definitions

Vision-Language Model: Vision-Language models are the deep learning models that learn both vision and language modalities together.

Tags: Tags are the diagnostic keywords (e.g., left lung, pulmonary atelectasis, hernia, pneumonia, etc.) associated with X-ray image.

Findings: The findings section in radiology reports is the clinical description of abnormalities and normalities observed on radiology image.

Impression: The impression section in radiology reports is the summary of findings section.

Knowledge Graph: Paulheim (2017) defines Knowledge Graph (KG) as "A knowledge graph (i) mainly describes real-world entities and their interrelations, organized in a graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other and (iv) covers various topical domains."

Knowledge Graph Embeddings (KGEs): Knowledge graph embeddings represent the entities and relations in lower-dimensional vectors.

KG-Grounding: The KG grounding is the process of extracting the subgraph (referred to as grounded KG) from domain-specific KG (in our case chestX-KG). A grounded KG is a subgraph from the chestX-KG whose nodes represents tags present in the input tag set plus additional relevant nodes.

3 Related Work

Researchers (Jing et al., 2017; Zhang et al., 2017; Yuan et al., 2019) studied the issue of automatic report generation. Their research looked into the visual attention given to recurrent decoders and convolution-recurrent architectures (CNN-RNN) that were first introduced by Vinyals et al. (2015) on image captioning. Transformers, attention-only based models that have replaced recurrent models in the NLP community (Vaswani et al., 2017; Devlin et al., 2018). Several attempts have been made in the medical field to create medical reports from the corresponding images. Most authors use multilabel image captioning to produce X-ray reports, and they subsequently use those captions as textual features. The IU-Xray dataset's chest X-ray images were used to generate the first structured report using tags predicted by a CNN-RNN model (Shin et al., 2016). In (Wang et al., 2017b), a system for generating natural reports for the Chest-Xray14 dataset, employing private reports, was presented. This framework used a non-hierarchical CNN-LSTM architecture and focused on semantic and visual aspects. The IU-Xray dataset was created by Jing et al. (2017) to generate radiology reports automatically.

There is a lot of research performed in NLG on multimodal constraints. NLG models based on transformers (Sollami and Jain, 2021) propose a model called MAnTiS, Multimodal Adaptation for Text Synthesis, as a general method for multimodal conditionality. In this method, separator tokens are used to separate each modality type, and modality-specific encoders are used to encode each modality type. Liu et al. (2021) contend that pre-trained language models and textual concepts by themselves are insufficient to give enough data for generative commonsense reasoning. They supply the ConceptNet KG as input to the transformer model for generative commonsense reasoning in addition to text input. Liu et al. (2020) proposes a Knowledge-enabled Bidirectional Encoder Representation from Transformers (K-BERT). Since the parameters of all pre-trained BERT models are the same, K-BERT can load any of them. Additionally, K-BERT can easily integrate domain knowledge into the models by giving them access to a KG without prior training. Xing et al. (2021) propose KM-BART to conduct the task of Visual Commonsense Generation (VCG) by integrating visual features in pretrained BART model.

4 Methodology

As shown in the figure 1, the model architecture consists of six major components, namely, KG grounding, KG embedding, image embedding, text embedding, encoder, and decoder. The KG grounding module extracts the grounded KG from the chestX-KG. Grounded KG includes all of the nodes in the input tag set and their significant neighbors. The KG embedding module converts the grounded KG into vector form using the KGE technique. The image feature extractor module generates the feature vectors for input chest X-ray images. To compute the text features, we use the BART text encoder method. The encoder and decoder are multilayer transformers. This section explains all components in detail.

4.1 Knowledge Graph Grounding

The KG grounding module extracts the small subgraph from chestX-KG for each report in the dataset, given a tag set as input. First, it adds all entities from chestX-KG that are present in the input tag set, and then it adds their significant neighbors.

The algorithm focuses on first finding the most appropriate path in chestX-KG from tag entities to the root of chestX-KG and then adding neighbor nodes that are connected with *DefaultPropertyOf* (default property of entities) relation. The following are the steps to extract grounded KG:

- Find all possible candidate paths from a matched entity to the root node.
- Find the most appropriate top five paths by ranking based on the precision and recall of entities in the input tag set and entities in all possible candidate paths.
- We consider all paths, including those with matched entities that are absent from the selected top-ranking path.
- Instead of adding all neighbors of input tag entities, we add only significant entities that are default properties or default descriptors of input entities.

Our proposed method reduces noise by adding only significant nodes to grounded KG. We propose a context-aware KG grounding algorithm to select M triples from the chestX-KG for an entity candidate. The pseudocode of this algorithm is shown as follows.

Algorithm 1: KG Grounding

```
Input  : K: Tags
         G(V, E) : chestX-KG
Output : Grounded KG
Find all candidate paths in G(V, E) that includes the node with input concept
path-dict -> initialize
for each path in possible candidate-paths do
    $Precision = \frac{K \cap Allentitiesinpath}{No.ofconceptsinK}$
    $Recall = \frac{K \cap Allentitiesinpath}{No.ofnodesinpath}$
    $F - score = \frac{2 * Precision * Recall}{Precision + Recall}$
    add path-dict -> (path:F-score)
end
Sort path-dict in descending order of F-score
Get top 5 paths
for each path in top-5-paths do
    if len(set(K) - set(path)) > 0 then
        Add all triplets from that path in grounded KG triple set
        for each node in path do
            Find all neighbors of node with default-property relation
            Add all triples of form (neighbor, DefaultPropertyOf, node) in grounded KG triple set
        end
    end
end
Return grounded KG triplets set.
```

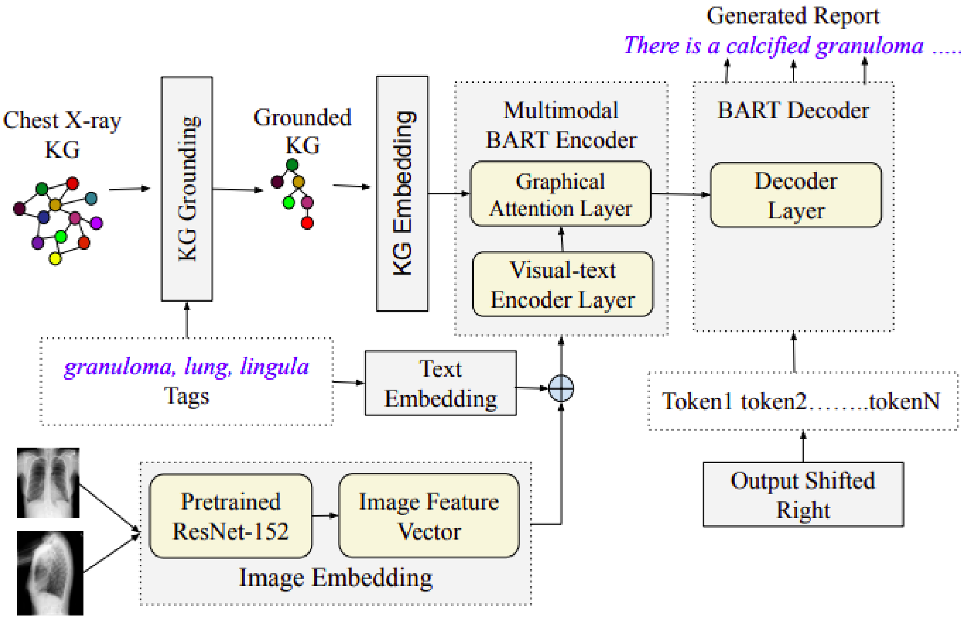


Figure 1: The architecture of our proposed KGVL-BART model. KGVL-BART has six important components: KG grounding, KG Embedding, image embedding, text embeddings, encoder, and decoder.

4.2 Knowledge Graph Embeddings

Knowledge Graph Embedding methods embed the components of a KG, including entities and relations, into continuous vector spaces. The chestX-KG is represented in low dimensional vector space using KGE. There are different techniques used for KG embeddings like TransR, TransH, TransE, TransD, *etc,* (Wang et al., 2017a). For simplicity and concreteness, in this work, we primarily consider the TransE (Bordes et al., 2013) model due to its state-of-the-art performance.

4.3 Text Embedding

The input embeddings in KGVL-BART are made of two separate embeddings, token embeddings and position embeddings. To get the final text embedding, we add the vectors of token embeddings and position embeddings.

4.3.1 Token Embeddings

Tokens are nothing but a word or part of a word. The textual encoder uses the vocabulary offered by large-cnn BART, and the token embedding is consistent with BART. Using a trainable lookup table, we transform each token in the input tag set into an embedding vector of dimension d.

In order to create these token embeddings, a method called BART tokenizer is used to tokenize the text. Input tag set T is tokenized as $\{t_1, t_2, ..., t_{|t|}\}$ and encoded as learned embedding $e_t = \{e_{t_1}, e_{t_2}, ..., e_{t_{|t|}}\}$. For a token t_n, its embedding is $e_{t_n} \in R^d$, where d is the dimension of the token embeddings. The encoder, decoder, and language modeling head (Press and Wolf, 2016) all share the embedding parameters. Due to the permutation-invariance of the attention layers, BART learns positional embeddings for absolute token positions and adds them to the token embeddings (Vaswani et al., 2017; Devlin et al., 2018).

4.3.2 Positional Embeddings

Position embeddings represent the position of the word within that sentence that is encoded into a vector. We must introduce some information about the relative or absolute location of the tokens in the sequence because our model lacks recurrence and convolution and hence cannot use the sequence's order. To do this, we augment the token embeddings at the base of the encoder and decoder stacks with positional embeddings. The positional encodings and token embeddings have the same dimension d allowing the token and positional embeddings to be added together. The text embeddings are the sum of the token embeddings and the positional embeddings, i.e., $e_{tp} = e_t + e_p$, where e_p is the positional embeddings.

4.4 Image Embedding

For image feature extraction, we use three different methods: i) Pretrained CheXNet (Rajpurkar et al., 2017) model (referred to as CheXNet), ii) Pretrained ResNet-152 model (referred to as ResNet-152), and iii) Fine-tuned ResNet-50 for multilabel image classification on NIH chest-xray dataset (referred to as NIH-ResNet). We extract the image embeddings from each of these models, and we train KGVL-BART separately for these methods. In this section, we give details about the pretrained ResNet-152 feature extractor.

We extract the embedding form of the final fully connected layer of the pre-trained ResNet-152 model (He et al., 2016). We transform images using the same parameters as during pretraining, which include resizing, center cropping, and normalizing. We project the image feature vector through a linear layer with a learnable weight matrix $W \in R^{N*d}$ onto the language model embedding space d.

To get the final input embeddings, we sum up the text and image embeddings. In addition to the original vocabulary of BART, for images, we use <*img*> and <*/img*> to indicate the start and the end of visual embeddings, respectively. Multimodal Feature Augmentation is done by adding image feature vector with the text feature vector to generate a single feature vector, i.e., $e_{tv} = e_{tp} + e_v$.

4.5 Encoder

The encoder uses two modalities—image and text, and text generation is conditioned on grounded KG. According to the figure 1, the KG enhanced encoder layer sits above the visual-textual encoder layer and is intended to enhance the visual-text representation $\{e_{tv_1}, e_{tv_2}, ..., e_{tv_n}\}$ by taking the KG structure into account. We use a graph attention layer to incorporate graph representations into the input encoding process. It uses explicit relations to help the model learn intra-concept relations more effectively. Formally, the grounded KG embedding, as well as the output visual-textual embeddings from the visual-textual encoder, are combined by the KG-augmented encoder to update the visual-textual token representation. Our self-attention layer and fully-connected layer with residuals make up the stack of m transformer blocks that make up our bidirectional multimodal encoder.

4.6 Decoder

The decoder uses the text embedding module at the bottom layer to encode the text. The decoder in our model is also a multi-layer transformer. Our decoder is auto-regressive and unidirectional. We skip over a detailed explanation of these modules because our textual transformers are the same as those used in BART (Lewis et al., 2019; Vaswani et al., 2017).

5 Experiments

The datasets, evaluation metrics, and baselines used for the training and evaluation of the KGVL-BART model are covered in detail in this section.

5.1 Dataset

We use the IU Chest X-ray dataset to train our model (Demner-Fushman et al., 2016). There are 3825 patient reports in this dataset. 7430 chest X-ray images from the front and sides contribute to this dataset. Each patient report contains two types of tags: MTI tags and manual tags from MESH[1] and RadLex[2]. Each report has three parts: an impression, which is a title or summary of the report; findings, which contain the report in detail; and manual tags. We use MESH tags (text) as one of the input to train our model. We concatenate impressions and findings and use it as target to train our model. IU Chest X-ray dataset includes normal and abnormal study reports. There are total 3825 reports out of which 1379 are normal and 1646 are abnormal. The dataset is balanced with respect to normal and abnormal reports. Additionally, we chose 500 samples at random to serve as the test set. Our split is 3025 for training, 300 for validation, and 500 for testing. Table 1 shows the samples from the IU-Chest X-ray dataset.

Table 1 shows an example from the dataset that we are using to train KGVL-BART model.

5.1.1 Chest X-ray Knowledge Graph (chestX-KG)

Nodes in our knowledge graph represent all necessary information, like findings, observations, anatomy, properties, and modifiers related to the organ. We define eight logical relations to construct chestX-KG. i) *PartOf*: It represents the relation between anatomy and sub-anatomy, ii) *TypeOf*: It represents the relation between similar type of entities, iii) *ModifierOf*: It denotes the descriptors of findings, anatomical locations, properties, *etc.*, iv) *ObservationOf*: It denotes the clinical observations observed for a particular finding, v) *Default-ObservationOf*: It denotes the observation that is associated by default with a particular anatomical

---

[1] `https://www.nlm.nih.gov/mesh/qualifiers_scopenotes.html`

[2] `https://radlex.org/`

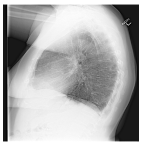
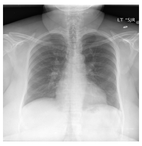
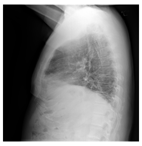
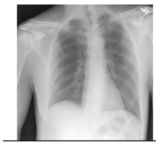
Frontal Image	Lateral Image	Tags	Target
		Osteophyte, thoracic vertebrae, multiple, small, Thickening, pleura, apex, bilateral, Lung, hyperdistention, mild	**Impression:** No acute cardiopulmonary abnormality. **Findings:** The cardiomediastinal silhouette and pulmonary vasculature are within normal limits. There is no pneumothorax or pleural effusion. There are no focal areas of consolidation. Cholecystectomy clips are present. Small T-spine osteophytes. There is biapical pleural thickening, unchanged from prior. Mildly hyperexpanded lungs.
		normal	**Impression:** No acute cardiopulmonary findings. **Findings:** Heart size and mediastinal contour are within normal limits. There is no focal airspace consolidation or suspicious pulmonary opacity. No pneumothorax or large pleural effusion. Mild degenerative change of the thoracic spine.
		Pulmonary Atelectasis, base, Spondylosis, thoracic vertebrae, Arthritis, cervical vertebrae	**Impression:** Basilar atelectasis. No confluent lobar consolidation or pleural effusion. **Findings:** The cardiac contours are normal. XXXX basilar atelectasis. The lungs are clear. Thoracic spondylosis. Lower cervical XXXX arthritis.
		Calcified Granuloma, lung, upper lobe, right	**Impression:** No acute cardiopulmonary process. **Findings:** The cardiomediastinal silhouette is within normal limits for size and contour. The lungs are normally inflated without evidence of focal airspace disease, pleural effusion, or pneumothorax. Stable calcified granuloma within the right upper lung. No acute bone abnormality.

Table 1: Samples from the IU Chest X-ray dataset. We use frontal images, lateral images, and tags as input to our model and target text as a concatenation of impression and findings columns from the IU Chest X-ray dataset. In this table, we show only the columns we use to train the KGVL-BART model.

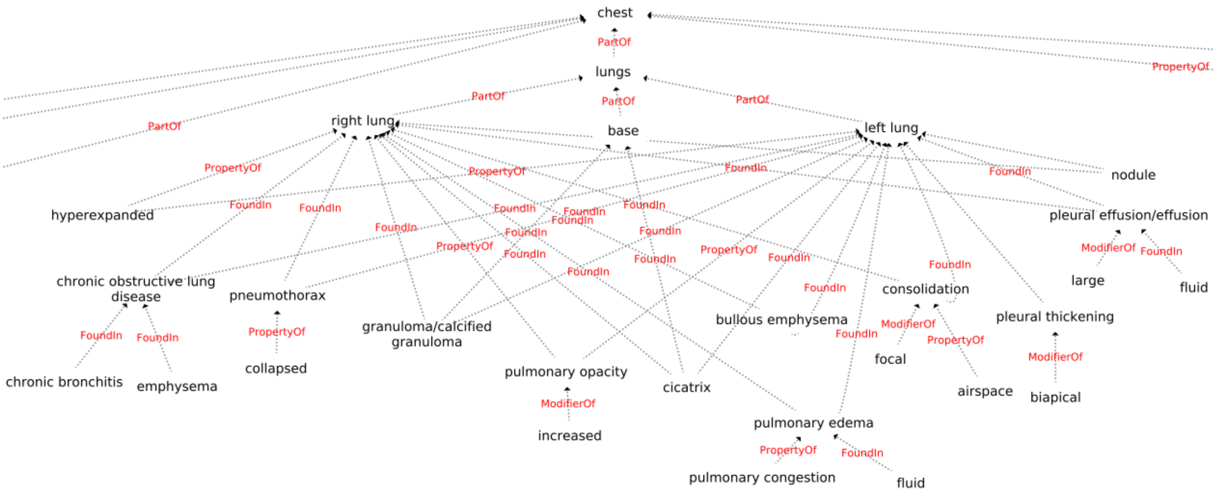


Figure 2: Chest X-ray KG constructed by extracting triplets from the free-text chest X-ray reports. Constructed KG is verified by radiologists. Only a portion of the entire KG is displayed due to space limitations.

location or particular finding. vi) *PropertyOf*: It denotes the relation between entities (anatomical entities, finding entities, observation entities, *etc.*) and their properties., vii) *DefaultPropertyOf*: It denotes the property that exist by default with particular anatomical location or particular finding., and viii) *FoundIn*: It denotes the relation between findings and corresponding anatomical location.

We use a rule-based and pattern-based approach to extract the triples from the IU X-ray text report corpus. After extracting triples from text-report corpus we construct hierarchical KG with chest as root node and children represents its parts or associated findings. More details about radiology KG construction is given in paper (Kale et al., 2022). The constructed KG is verified by two radiologists who are involved in this research work. Figure 2 shows the part of chestX-KG that we have constructed.

5.2 Training

Three different techniques are used to extract the features from the frontal and lateral X-ray images: i) Pretrained CheXNet model, ii) Pretrained Resnet-152 model, and iii) Fine tuned Resnet-50 for multi-label image classification on the NIH chest X-ray dataset[3] (Wang et al., 2017b).

We have trained the KGVL-BART model for each of these image embeddings separately and also provided quantitative results. To implement the TransE model for KG embeddings, we use the open-source OpenKE[4] tool. The chestX-KG contains 106 nodes and 126 triples.

5.3 Pretraining Setup

For model pretraining we use ConceptNet KG (Speer et al., 2017) and common sense generation dataset (Lin et al., 2020). For pretraining, we do not use images. At the time of pretraining, we are not adding image features to text features. We use byte-pair encoding for tokenization, with a maximum length of 32 for the encoder and 64 for the decoder. We set the learning rate to 0.00001 and used AdamW with $\beta1 = 0.9$, $\beta2 = 0.98$ for optimization. We set the batch size to 60. We trained the KGVL-BART for 10 epochs, and the gradients are accumulated every 6 steps. We apply dropout with a probability of 0.1 to avoid over-fitting. While inferencing, we use beam search with beam size 5 and a length penalty of factor 0.6.

5.4 Training Setup

We propose our own algorithm for KG-grounding tasks. We construct grounded KGs for each report tag set. We use pretrained weights to initialize the KGVL-BART model. We train our model on the IU Chest X-ray dataset.

We use byte-pair encoding for tokenization with a maximum length of 300 for the encoder and 500 for the decoder. We set the learning rate to 0.00001 and used AdamW with $\beta1 = 0.9$, $\beta2 = 0.98$ for optimization. We set the batch size to 18. We trained the KGVL-BART for 15 epochs, and the gradients are accumulated every 6 steps. We apply dropout with a probability 0.1 to avoid over-fitting. We use beam search with beam size 5 and length penalty with factor 0.6 while inferencing. DGX A100-SXM-80GB GPU server takes approximately 15 minutes for a single epoch.

6 Evaluation

We evaluate our model by automatic metrics and human evaluation as well. For automatic evaluation we use word-overlap metrics. Manual evaluation is performed by two radiologists.

6.1 Quantitative Evaluation

We compare the performance of KGVL-BART model with previous state-of-the-art conditional transformer text generation models, i) the CNN-RNN model (Vinyals et al., 2015), ii) CDGPT2 for visual input only, text input only, and for both visual and text inputs (Alfarghaly et al., 2021). Following other conventional generation tasks, we use several widely-used automatic metrics to automatically assess the performance, such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004) and METEOR (Banerjee and Lavie, 2005), which mainly focus on measuring n-gram similarities. We have evaluated other NLG metrics like BLEURT (Sellam et al., 2020) and BERTScore (Zhang et al., 2019). Table 2 shows the NLG metrics score of generated X-ray reports by KGVL-BART and baseline models vs. gold standard X-ray reports. Evaluation is done on a test dataset. We have added ablation study, which shows that knowledge infusion improves the report generation. Last row in table 2 shows the results of our model by removing KG augmented layer from it.

Even though the dataset is balanced with respect to normal and abnormal reports, However, multiple organ findings are included in a single report; if at least one organ is found to be abnormal, the report is classified as abnormal. Findings for the other organs are normal, despite the fact that this is classified as an abnormal report. As a result, our model may be overly focused on widely reported normal findings. Hence, we have added evaluation metrics only to abnormal studies as well. Table 3 shows the NLG metrics for abnormal studies. For evaluation, we consider abnormal studies from the test set. Our results show that our model is capable of generating better reports than SoTA models for abnormal findings as well.

6.2 Qualitative Evaluation

This section provides qualitative analysis by a radiologist having experience of thirty years. Access to images and the ground-truth reports was given

---

[3] https://www.kaggle.com/datasets/nih-chest-xrays/data

[4] https://github.com/thunlp/OpenKE

Method	Automatic Evaluation Metrics								
	Bleu-1	**Bleu-2**	**Bleu-3**	**Bleu-4**	**Rouge-L**	**Meteor**	**chrF++**	**BLUERT**	**BERTScore**
CNN-RNN (Vinyals et al., 2015)	0.316	0.211	0.140	0.095	0.267	0.159	-	-	-
CDGPT2-vis-only (Alfarghaly et al., 2021)	0.340	0.209	0.138	0.091	0.281	0.153	-	-	-
CDGPT2-sem-only (Alfarghaly et al., 2021)	0.357	0.224	0.151	0.103	0.275	0.149	-	-	-
CDGPT2 (Alfarghaly et al., 2021)	0.387	0.245	0.166	0.111	0.289	0.164	0.370	0.460	0.888
CNN-TRG (Pino et al., 2021)	0.273	-	-	-	0.352	-	-	-	-
KGVL-BART (CheXNet)	0.326	0.139	0.080	0.050	0.340	0.387	0.453	0.473	0.892
KGVL-BART (NIH-ResNet)	0.324	0.144	0.090	0.060	0.355	0.390	0.467	0.468	0.889
KGVL-BART (ResNet-152)	**0.423**	**0.256**	**0.194**	**0.165**	**0.444**	**0.500**	**0.543**	**0.526**	**0.909**
Our model (without KG layer/ResNet-152)	0.341	0.188	0.142	0.119	0.351	0.376	0.424	0.478	0.892

Table 2: Results on whole test set (abnormal + normal studies): BLEU, ROUGE and METEOR score of generated X-ray reports by previous transformer-based models and our KGVL-BART models vs. gold standard X-ray reports. The best results are in bold font, and the second best is underlined.

Method	Automatic Evaluation Metrics					
	Bleu-1	**Bleu-2**	**Bleu-3**	**Bleu-4**	**Rouge-L**	**Meteor**
CDGPT2 (Alfarghaly et al., 2021)	0.273	0.090	0.034	0.013	0.267	0.316
KGVL-BART (CheXNet)	0.308	0.123	0.073	0.049	0.314	0.359
KGVL-BART (NIH-ResNet)	0.299	0.119	0.067	0.046	0.309	0.352
KGVL-BART (ResNet-152)	**0.388**	**0.207**	**0.139**	**0.108**	**0.397**	**0.458**

Table 3: Results on abnormal studies from test set: BLEU, ROUGE and METEOR score of generated X-ray reports by previous transformer-based models and our KGVL-BART models vs. gold standard X-ray reports. The best results are in bold font, and the second best is underlined.

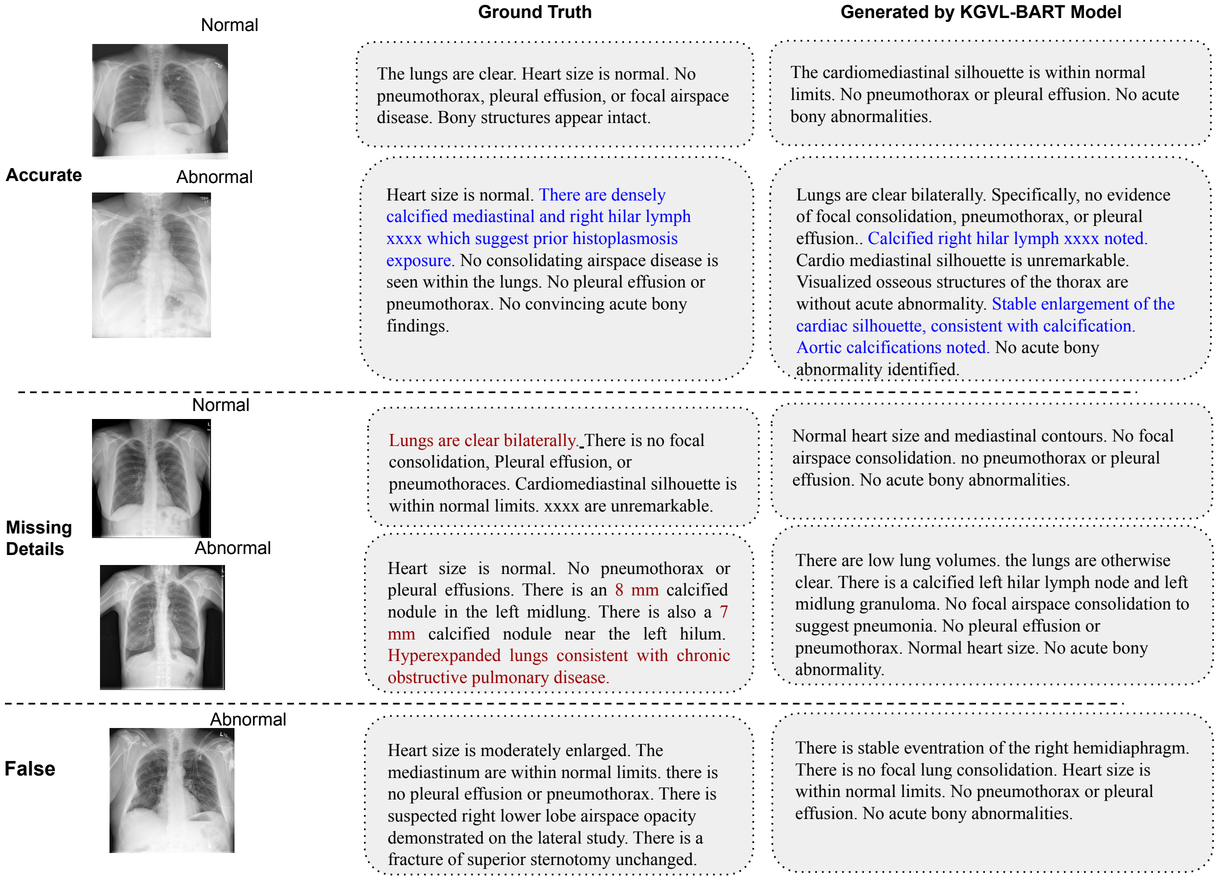
to the radiologist to evaluate the automatically-generated reports. The radiologist classified the generated reports into **accurate** (report with most of the vital information), **missing details** (reports with no false information but missing some vital details), and **false** predictions (report with false information and overall incorrect diagnosis). We have provided 200 random samples from the test dataset and their corresponding generated reports for qualitative analysis. Qualitative evaluation by the radiologist shows that 73% of the reports generated were fully correct, only 5.5% are completely wrong and 21.5% have important missing details though overall correct. Further, these random samples were classified into normal and abnormal reports. For normal studies 95.6% of the reports generated were fully correct, 4.40% have important missing details though overall correct and 0% false reports. For the abnormal cases, the model could generate 54.13% of the reports correctly, 35.78% missing details, and 10% had false reports. Table 4 contains the results of the qualitative analysis. The reports classified as missing details by the radiologist also contain useful information that can be used to speed up the report writing process. Overall results are promising however, there is still a scope for improvement. The results show that the model is a bit biased towards normal cases, which is often the case with medical datasets as it lacks necessary details to describe the different irregularities present in the image.

Method	**Samples**	**Accurate**	**Missing Details**	**False**
Ours	*All*(500)	**73.00%**	**21.50%**	**05.50%**
	Normal(183)	95.60%	04.40%	**00.00%**
	Abnormal(317)	**54.13%**	**35.78%**	**10.00%**
CDGPT2	*All*(500)	61.60%	28.20%	10.20%
	Normal(201)	**99.00%**	**00.00%**	01.00%
	Abnormal(299)	36.50%	47.10%	16.40%

Table 4: Results of generated reports, manually evaluated by radiologist. Manual evaluation is done on the IU X-RAY dataset. Best results are shown in a bold face.

7 Summary, Conclusion and Future work

We introduce KGVL-BART a knowledge-enhanced Vision Language model to generate X-ray reports from chest X-ray images and tags. We develop a knowledge graph called chestX-KG, which is verified by two experienced radiologists. Overall the reports generated are accurate. Our approach to constructing grounded KGs is relatively noise-free since it considers only entities in the hierarchical path from tag set entities to the root node of the knowledge graph and only adds neighbors with "default" properties. Experimental results show the efficacy of our method and its superiority over SoTA.



		Ground Truth	Generated by KGVL-BART Model
Accurate	Normal	The lungs are clear. Heart size is normal. No pneumothorax, pleural effusion, or focal airspace disease. Bony structures appear intact.	The cardiomediastinal silhouette is within normal limits. No pneumothorax or pleural effusion. No acute bony abnormalities.
	Abnormal	Heart size is normal. There are densely calcified mediastinal and right hilar lymph xxxx which suggest prior histoplasmosis exposure. No consolidating airspace disease is seen within the lungs. No pleural effusion or pneumothorax. No convincing acute bony findings.	Lungs are clear bilaterally. Specifically, no evidence of focal consolidation, pneumothorax, or pleural effusion.. Calcified right hilar lymph xxxx noted. Cardio mediastinal silhouette is unremarkable. Visualized osseous structures of the thorax are without acute abnormality. Stable enlargement of the cardiac silhouette, consistent with calcification. Aortic calcifications noted. No acute bony abnormality identified.
Missing Details	Normal	Lungs are clear bilaterally. There is no focal consolidation, Pleural effusion, or pneumothoraces. Cardiomediastinal silhouette is within normal limits. xxxx are unremarkable.	Normal heart size and mediastinal contours. No focal airspace consolidation. no pneumothorax or pleural effusion. No acute bony abnormalities.
	Abnormal	Heart size is normal. No pneumothorax or pleural effusions. There is an 8 mm calcified nodule in the left midlung. There is also a 7 mm calcified nodule near the left hilum. Hyperexpanded lungs consistent with chronic obstructive pulmonary disease.	There are low lung volumes. the lungs are otherwise clear. There is a calcified left hilar lymph node and left midlung granuloma. No focal airspace consolidation to suggest pneumonia. No pleural effusion or pneumothorax. Normal heart size. No acute bony abnormality.
False	Abnormal	Heart size is moderately enlarged. The mediastinum are within normal limits. there is no pleural effusion or pneumothorax. There is suspected right lower lobe airspace opacity demonstrated on the lateral study. There is a fracture of superior sternotomy unchanged.	There is stable eventration of the right hemidiaphragm. There is no focal lung consolidation. Heart size is within normal limits. No pneumothorax or pleural effusion. No acute bony abnormalities.

Figure 3: Examples of ground truth and reports generated by KGVL-BART (ResNet-152). The first slot shows the instance of accurate prediction by the KGVL-BART model for normal and abnormal cases. Abnormal findings are highlighted in blue. The second slot shows the example of partially correct predictions by the KGVL-BART model for normal and abnormal cases. Details that are present in ground truth but missing in the predicted report is highlighted in red. The third slot shows the example of false predictions by the KGVL-BART model for abnormal case. For normal case model does not generate false report.

Our future work consists in training our model on the much larger MIMIC-CXR (Johnson et al., 2019) dataset. We would also like to expand the scope of our work to CT, MRI, *etc.*

Limitations

The IU Chest X-ray and the MIMIC-CXR datasets are publicly available that links chest X-ray images with text radiology reports. The IU Chest X-ray dataset available for general use and the MIMIC-CXR dataset has restricted access. Annotating medical reports requires domain experts' knowledge, and it is costly. Medical data is likewise subject to strict privacy regulations and is governed, for example, by the Health Insurance Portability and Accountability Act (HIPAA). Therefore, only a small amount of data is accessible to the general (research/corporate/industry) use.

There are many more sentences in this dataset describing normalities than abnormalities. As a result, most machine learning models are biased to produce normal reports more frequently than abnormal ones. Given the scarcity of examples, abnormalities are more challenging to find.

Ethics Statement

IU Chest X-ray dataset's authors used appropriate techniques to de-identify the text reports. Data is anonymized; hence our model will not disclose information about the patient's identity.

References

Omar Alfarghaly, Rana Khaled, Abeer Elkorany, Maha Helal, and Aly Fahmy. 2021. Automated radiology report generation using conditioned transformers. *Informatics in Medicine Unlocked*, 24:100557.

Richa Arora. 2014. The training and practice of radiology in india: current trends. *Quantitative imaging in medicine and surgery*, 4(6):449.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. In *International Conference on Machine Learning*, pages 1931–1942. PMLR.

Dina Demner-Fushman, Marc D Kohli, Marc B Rosenman, Sonya E Shooshan, Laritza Rodriguez, Sameer Antani, George R Thoma, and Clement J McDonald. 2016. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23(2):304–310.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tim Fischer, Steffen Remus, and Chris Biemann. 2022. Measuring faithfulness of abstractive summaries. In *Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)*, pages 63–73, Potsdam, Germany. KONVENS 2022 Organizers.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Baoyu Jing, Pengtao Xie, and Eric Xing. 2017. On the automatic generation of medical imaging reports. *arXiv preprint arXiv:1711.08195*.

Alistair EW Johnson, Tom J Pollard, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Yifan Peng, Zhiyong Lu, Roger G Mark, Seth J Berkowitz, and Steven Horng. 2019. Mimic-cxr-jpg, a large publicly available database of labeled chest radiographs. *arXiv preprint arXiv:1901.07042*.

Kaveri Kale, Pushpak Bhattacharyya, Aditya Shetty, Miling Gune, Kush Shrivastava, Rustom Lawyer, and Spriha Biswas. 2022. Knowledge graph construction and its application in automatic radiology report generation from radiologist's dictation. *arXiv preprint arXiv:2206.06308*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and S Yu Philip. 2021. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6418–6425.

Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. 2022. Multimodal contrastive learning with limoe: the language-image mixture of experts. *arXiv preprint arXiv:2206.02770*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508.

Pablo Pino, Denis Parra, Cecilia Besa, and Claudio Lagos. 2021. Clinically correct report generation from chest x-rays using templates. In *International Workshop on Machine Learning in Medical Imaging*, pages 654–663. Springer.

Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*.

Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, et al. 2017. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*.

Hoo-Chang Shin, Kirk Roberts, Le Lu, Dina Demner-Fushman, Jianhua Yao, and Ronald M Summers. 2016. Learning to read chest x-rays: Recurrent neural cascade model for automated image annotation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2497–2506.

Michael Sollami and Aashish Jain. 2021. Multimodal conditionality for natural language generation. *arXiv preprint arXiv:2109.01229*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017a. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. 2017b. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2097–2106.

Yiran Xing, Zai Shi, Zhao Meng, Gerhard Lakemeyer, Yunpu Ma, and Roger Wattenhofer. 2021. Km-bart: Knowledge enhanced multimodal bart for visual commonsense generation. *arXiv preprint arXiv:2101.00419*.

Jianbo Yuan, Haofu Liao, Rui Luo, and Jiebo Luo. 2019. Automatic radiology report generation based on multi-view image fusion and medical concept enrichment. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 721–729. Springer.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Zizhao Zhang, Yuanpu Xie, Fuyong Xing, Mason McGough, and Lin Yang. 2017. Mdnet: A semantically and visually interpretable medical image diagnosis network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6428–6436.

A simple but effective model for attachment in discourse parsing with multi-task learning for relation labeling

Zineb Bennis
IRIT
zinebennisa@gmail.com

Julie Hunter
Linagora
jhunter@linagora.com

Nicholas Asher
IRIT
Nicholas.Asher@irit.fr

Abstract

We present a discourse parsing model for conversation trained on the STAC corpus (Asher et al., 2016). We fine-tune a BERT-based model to encode pairs of discourse units and use a simple linear layer to predict discourse attachments. We then exploit a multi-task setting to predict relation labels, which effectively aids in the difficult task of relation type prediction; our F1-score equals or surpasses the state of the art in the approaches we have reimplemented using code from the authors with no loss in performance for attachment, confirming the intuitive interdependence of these two tasks. Our method also improves over other discourse parsing models in the literature in permitting attachments in which one node has multiple parents, an important feature of multiparty conversation.

1 Introduction

Discourse parsing, the task of predicting graphs that represent semantic relations (arcs) between *elementary discourse units* or EDUs (nodes), is a hard problem in NLP due to the complexity of discourse graphs and the frequent lack of surface cues provided by EDUs, which forces parsers to rely on deep, semantic information. Multiparty, spontaneous conversation is especially tricky as the structure often meanders and a single participant can respond to multiple discourse moves at once, yielding non-tree-like structures that challenge existent parsing techniques (Afantenos et al., 2015).

While it is tempting to attack a complex problem with complex machinery, as recent research on discourse parsing has done (see Section 6), we show that a simple model can achieve or surpass state of the art results for discourse relation labeling with a little inspiration from human discourse processing.

First, because a decision about whether two EDUs are attached by a semantic relation generally requires reasoning about their contents together, our transformer-based model encodes embeddings of EDU *pairs*, exploiting a form of message passing simpler than graph neural net models (Wang et al., 2021) while achieving better results. Next, we draw on the fact that for human annotators, the tasks of discourse attachment and relation labeling are often interdependent: sometimes one sees how to attach two EDUs but only later how to determine the relation that links them; sometimes, a clue, e.g. an explicit marker like *because*, makes the relation clear, and the task is to find the second argument to the relation. Our model exploits this interdependence with a multi-task architecture for attachment prediction and relation labeling. Finally, our model allows for the non-tree-like structures.

In Section 2, we provide background on discourse structure and data sets. In Sections 3 and 4, we describe our model and results. Section 6 presents related work in discourse parsing.

2 Discourse parsing theories and data sets

Just as a sentence is not a bag of words but comes with a structure that serves to compute its meaning from that of its constituent words, so too a discourse or conversation is not a bag of dialogue moves but comes with a structure that enables an interpreter to compute an overall meaning from its constituents. EDUs are clauses or subclausal units that serve as the minimal, linguistic constituents upon which discourse structures are built (Marcu, 1999), and discourse parsing involves finding the recursive structure over EDUs that exploits their semantic content together with various contextual features.

There are two main theories that have investigated complete discourse structures for texts: RST (Mann and Thompson, 1987) and SDRT (Asher, 1993; Asher and Lascarides, 2003). Only SDRT has been applied to multi party conversation as in the STAC and Molweni datasets (Asher et al., 2016;

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3412–3417
May 2-6, 2023 ©2023 Association for Computational Linguistics

Li et al., 2020). Given our interest in multiparty conversation, we use SDRT and three versions of the STAC corpus, a set of multi-party chats from an online version of the game *Settlers of Catan*, in which players trade or otherwise acquire resources in order to build roads and settlements and thereby score victory points. The standard version, *S*, contains only linguistic (chat) moves made by players; the situated version, *S-Sit*, integrates descriptions of nonlinguistic events (game moves), represented as *elementary event units* (EEUs). In *S-Sit*, both EEUs and EDUs are integrated into discourse structure. We present the third version in Section 2.1.

In SDRT a conversation provides a number of EDUs linked together to form a weakly connected, Directed Acyclic Graph (DAG). Each EDU apart from the head has at least one incoming link. Backwards links (where an EDU attaches to another EDU that comes after it in the dialogue) are prohibited if the EDUs are produced by different speakers (Perret et al., 2016). Each edge of the DAG is labeled with one of 16 different types of discourse relation, such as Explanation (Exp), Question-Answer-Pair (QAP), or Acknowledgement (Ack).

The DAGs postulated by SDRT allow one child to have multiple parents in the structure. Figure 1 provides a representative example from the STAC corpus: with his 'kk' William acknowledges *both* refusals to his offer and signals that he is moving on before ending his turn.

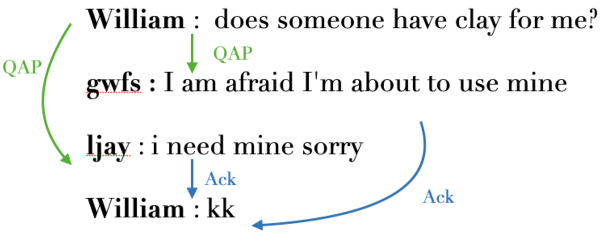


Figure 1: Example from the STAC corpus *S* illustrating an EDU ('kk') with multiple parents

In multiparty dialogue, coordination and negotiation over content is key. EDUs with multiple parents typically mark moves that signal such a coordination or end of negotiation. Multi-parent links can thus have a major impact on conversational structure and its effects on downstream tasks, despite their statistical infrequency. An important advantage of our model over alternative neural parsers is that it is the only one to take these links into account in both model design and evaluation.

2.1 A new STAC dataset: STAC-Squished

Like other recent work in discourse parsing, we develop a model trained on *S* and detail our results on *S* in Section 4. But *S* is not ideal for two reasons.

First, we discovered that it contains duplicated data points due to a mistake in the data extraction script. There are 488 duplicated EDUs, corresponding to 60 duplicated dialogues.

Second, and more seriously, there are rhetorical incoherences in *S*. When we examined relation labels, we found, with Badene et al. (2019), that some were not coherently used, because *S* lacks essential nonlinguistic (game) information (Asher et al., 2016). Figure 2 provides an example. The exchange on the right, taken from *S*, is somewhat incoherent; the Continuations (dark green) from 282-283 and 283-289 make little sense, but were included in order to make a complete DAG. On the left, we see the full exchange, from *S-Sit*, with a far more intuitive discourse structure lacking the Continuations.

There are 300 such cases out of 1116 Continuation instances in the training set and 45 out of 113 such cases in the test set. Indeed, dialogue moves and speaker interactions in the conversations often depend on nonlinguistic actions taken by players, but we can only see this dependence once we move to *S-Sit*, where the EEUs provide such actions and other important information on game evolution. We must take account of those EEUs to accurately reflect the discourse structure.

S-Sit builds graphs over EDUs and EEUs and so includes information essential to properly understanding many linguistic interactions. For this reason, we consider *S-Sit* an overall better data source for correct discourse structures.

However, *S-Sit* also has certain drawbacks for training discourse parsing models. It contains multiple, long chains of EEUs connected to their immediate predecessors. The number of EEU-EEU adjacent attachments is very high and highly predictable, artificially inflating F-scores on attachment and relation labeling. Moreover, long sequences of EEUs often induce longer distances between EDUs that need to be attached. Our model, like most, is biased towards predicting closer attachments due to their abundance in the training set. *S-Sit* exacerbates this situation and makes our model actually perform worse on EDU-only attachments (see Section 4).

Figure 2: An example from STAC. On the left, we see the full interaction, from *S-Sit*, that contains both chat and game moves. On the right, we see the version from *S* that contains only the chat moves.

Data	EDU	EEU	R-E	Mean
S	13054	0	0	2.04
S-Sit	12588	18576	16382	2.14
S-Sq	12588	12985	10790	2.04

Table 1: Data set stats

The shortcomings of *S-Sit* prompted us to modify it in order to ignore the highly predictable relations and attachments between adjacent EEUs that are not attached to any EDU. To do this, we treated each sequence of EEUs as one block without any internal structure. That is, we collapsed the sequence into a single EEU. Table 1 reflects differences between the various STAC corpora in numbers of EDUs, EEUs, Relation instances over only EEUs (R-E) and mean distances between attachments of linguistic units. We have made available the new *S-Sq* (squished) corpus at https://github.com/zineb198/LineBert.

3 Our Model

S contains a set of multi party dialogues, each consisting of a set of n EDUs $[e_1, e_2, .., e_n]$. Our model BERTLine, is a simple but efficient discourse parsing model with two components: one for inferring attachments and one for inferring relation labels. Like Shi and Huang (2019) and Wang et al. (2021), we compute a standard unlabeled attachment and labeled attachment score for evaluation.

Attachment. BERTLine's attachment module has an encoder-decoder architecture. Encoding for each EDU pair is obtained by finetuning a BERT model on the task of predicting if two EDUs are attached. We used BERT because it takes the position IDs of tokens as input, which is useful for the pair encodings. BERTLine has access only to the content and the speaker/emitter of the EDUs. $[CLS]$ vectors furnish embeddings h of EDU pairs.

$$h_{(i,j)} = BERT(e_i, e_j)$$

The encoder portion of BERTLine is finetuned using the loss L_{l_1} on S_l, the set of positive and negative attachments. Where θ is the parameters to be optimized, $s_{*i,j}$ refers to gold data and $s_{i,j}$ to predictions:

$$L_{l_1}(S_l, \theta) = -\Sigma_{S_l} logP(s_{*i,j} = s_{i,j} | h_{i,j}) \quad (1)$$

We concatenate h with a vector $struct_{(i,j)} = [t_{(i,j)}, d_{(i,j)}]$ that represents information useful for computing attachments (Perret et al., 2016), namely speaker change and EDU distance (i.e., how many EDUs occurred between i and j in the natural order of the dialogue). The function $t_{i,j}$ returns 1/0 if i and j have the same/different speaker; $d_{i,j}$ returns the distance. Our final pair embeddings are:

$$H_{(i,j)} = h_{(i,j)} \oplus struct_{(i,j)}$$

The decoder part of BERTLine's attachment model is a linear neural network. Like previous work, this layer predicts for each EDU e_j, which of the preceding EDUs e_i, with $j > i \geq j - 10$ (i.e., up to a certain limit) are likely parents of e_j. It is a binary classification on each pair embedding.

$$P_j = Linear(H_{(j-1,j)}, H_{(j-2,j)}, ..., H_{(j-10,j)})$$

Our loss function is a binary cross entropy over possible attachments for each dialogue d in training set D. Below $l_{*i,j}$ indicates if e_i and e_j are attached in the gold data, and $l_{i,j}$ is a predicted link:

$$L_{link}(d, \theta) = -\Sigma_{j=1}^{|d|} \Sigma_{i=0}^{j} logP(l_{*i,j} = l_{i,j} | H_{i,j}) \quad (2)$$

$$L_{link}(\theta) = \Sigma_{d \in D} L_{link}(d, \theta) \quad (3)$$

The set of predicted attachments $\mathcal{L}$ contains all pairs $l_{i,j}$ whose link probability exceeds a given threshold α, optimized at 0.8 through experiments on the validation set; i.e., $l_{i,j} \in \mathcal{L}$ *iff* $P(l_{i,j}) > \alpha = 0.8$. We trained the linear layer for 15 epochs.

This set up allows us to predict multiple parents for a given EDU. Of the 77 multi-parent structures (8 with more than 3 parents) in the test set, we captured about 20%.

Relation labeling. To test the interdependence of link and relation prediction, we built a BERT model with two classification heads. The first head learns to predict links between EDUs and the second head learns to classify EDU pairs in terms of SDRT's 16 relation labels. We compute the loss of each head independently, average the results, and back-propagate the average back to the BERT encoder. At inference time, only the relation classification head is used for predictions.

We fine-tuned the BERT model over S_l over 3 epochs, together with training over S_r, the set of relation label instances. $r_{(*i,j)}$ refers to gold relation data, $r_{(i,j)}$ to predictions.

$$L_{rel}(S_r, \theta) = -\Sigma_{S_r} logP(r_{*i,j} = r_{i,j} | h_{i,j}) \quad (4)$$

$$L_{multi}(S_l \cup S_r, \theta) = mean(L_{l_1} + L_{rel}) \quad (5)$$

All previous work apart from Wang et al. (2021) defines the task of relation prediction as following that of link prediction in a pipeline architecture. Our multitask setup allows us to have an embedding containing information about both tasks and improves the relation F1-score by one point over a pipeline relation prediction task.

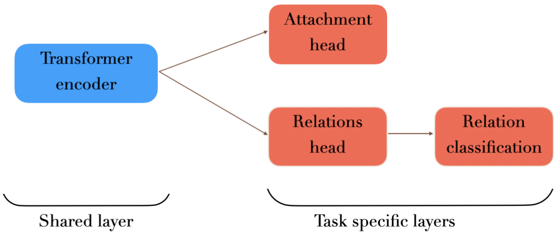


Figure 3: Structure of the multi-task model

Evaluation Metric: We used a micro averaged F1-metric over all gold data attachments and relation label instances for the scores in Table 2.

4 Results

Table 2 shows that BERTLine is competitive on S with the most advanced model of Wang et al. (2021) and even beats its relation labeling score. We provide both the reported scores of Shi and Huang (2019), Liu and Chen (2021) and Wang et al. (2021) and the scores that we got by rerunning their code using the same gold input from S with our evaluation metric, which explicitly considers EDUs with multiple parents.[1] Discrepancies in reported scores, also noted in Wang et al. (2021), may depend on choices of what and how to evaluate but also on machine hardware.

With regard to the different versions of STAC, BERTLine's overall scores improve between S and and *S-Sit* in Table 3, although there is a dramatic drop in linguistic only attachments (A-L) for *S-Sit*. This is due to the presence of many easy to predict attachments and relation instances between adjacent EEUs, which drown out performance on harder, longer distance EDU attachments. Interestingly, our score also improves considerably with *S-Sq*, and linguistic-only attachment suffers a much smaller drop. We hypothesize that this is because we have lowered the median distance between EDUs and rectified the imbalance between EDU and EEUs by deleting long series of EEUs.

BERTLine has a far simpler architecture than the neural models whose code we were able to rerun. It is essentially a local model that uses a minimal amount of message passing from nodes to potential neighbors, a technique from graph neural nets also used to encode entire graphs in Wang et al. (2021). BERTLine's main advantage comes from the explicit pair encoding of EDUs. While broader contextual information and structural constraints on DAGs like those in Perret et al. (2016) can undoubtedly improve scores for discourse attachment and relation labeling, we have sought to show that the efforts of complex architectures to harness this extra information have failed to lead to substantial gains. How to more successfully exploit it is our main research topic for the future.

5 Ablation study

We evaluated the efficiency of two key parts of our model, the multitask set up and the local structural information that we add to the EDU encodings. We

---

[1] We ran all experiments on a Dell T630 bi pro machine with Nvidia GTX1080Ti GPU cards

Model	DAG	DS-S	Hier-S	GNN-S	BL-S	BL-S-Sq
Attachments	69.0	[73.2/ 72.5]	[75.1/ 68.61]	[73.4/ 73.2]	73.1	79.49
Relation labeling	53.1	[55.7/ 54.8]	[57.1/50.48]	[57.3/ 55.5]	**56.25**	71.15

Table 2: We provide both the original, reported F1-scores for Perret et al. (2016) (DAG), Shi and Huang (2019) (DS), Liu and Chen (2021) (Hier) and Wang et al. (2021) (GNN) on the standard STAC dataset S, as well as the scores we were able to obtain using their models or retraining from their code using our evaluation metric (reported/recomputed). BERTLine (BL) beats the state of the art for relation label prediction on S with a score of 56.25. We also provide BL's results on the squished STAC dataset *S-Sq* with great performance on relation labeling.

F1	**Attach**	**Relns**	**A-L**
S	73.06	56.25	**73.06**
S-Sit	76.94	69.6	62
S-Sq	**79.93**	**71.64**	68.7

Table 3: F1 score for STAC data sets.

compared BERTLine with two baselines: (i) **BL-Simple**: simple BERT is finetuned to predict the relation of a predicted attachment; (ii) **BL-Noinfo**: $H_{i,i}$ without $struct_{i,j}$. We train each variation 10 times and compare the scores to the average over 10 runs for BERTLine. Table 4 shows how structural information and the multitask setup moderately improve the predictions of BERTLine on *S-Sq*.

Model	**BL**	**BL-Simple**
Attach	79.49	78.55
Model	**BL**	**BL-Noinfo**
Rel	71.41	71.15

Table 4: Scores on STAC-squished (*S-Sq)*

6 Related work

Older approaches to multiparty conversation (Afantenos et al., 2015; Perret et al., 2016) used manual features about EDU pairs and simple ML models to build a local attachment model for predicting attachment and relation labels. They also added a decoding mechanism. BERTLine's local model uses transformer-style embeddings.

Shi and Huang (2019) were the first to obtain significant discourse parsing results using a neural approach on the corpus *S*. They attempted to capture incremental and contextual effects in their model by training a supplemental *Structured Encoder* that incrementally updates attachment paths (sequences of parent-child EDUs). However, Wang et al. (2021) showed that the model obtains similar scores with or without the *Structured encoder*; the encoder didn't capture what it intended. Moreover, they implemented this method with Python 2 and Tensorflow 1.3, which are not in use anymore.

Liu and Chen (2021)'s model encodes EDUs using a pre-trained RoBERTa model (Liu et al., 2019) and a bi-GRU cell to capture contextual information but limits the size of the input dialogues. It uses two linear layers for link and relation prediction. We could not reproduce their results with their model or our reimplementation [2] perhaps because of an evaluation metric that does not consider multiple parents or all gold EDU pairs.

The Structure Self-Aware Graph Neural Network (SSA-GNN) by Wang et al. (2021) proposes a complex GNN-based architecture and model that uses both EDU and edge embeddings. The model is comprised of a Hierachichal GRU gate to obtain contextual EDU representations. They then apply the SSA-GNN to capture implicit structural information between EDUs, using a Structure-Aware Scaled Dot-Product Attention (Zhu et al., 2019; Wang et al., 2020) to update edge and EDU representations. A teacher network is also trained and supplements the standard classification loss with an auxiliary loss to enhance learning performances. Our model is simpler with better results. Moreover, none of the models by (Shi and Huang, 2019; Liu and Chen, 2021; Guz et al., 2020; Wang et al., 2021) predict multiple parents for attachments.

7 Conclusions

We have described a simple yet effective discourse parser that provides multiple attachments and codependent learning of the labeling and attachment tasks. Our model is the only neural parser that does this. We also wanted to show the power of local information when cleverly used. Indeed, discourse parsing requires contextual information, but our results show that current research does not yet leverage that information to achieve gains that convincingly outstrip those of a local model.

---

[2] https://github.com/zineb198/F1_recompute

8 Limitations

Some of the limitations of this work are the lack of diversity in expert-annotated discourse data sets on multiparty and situated dialogue. Since current data sets come from forums and chat messages, we still have to see how our model behaves in a spoken conversation context. More investigation efforts must be made in order to better analyze and evaluate *S-Sq*. While it has a more predictable structure due to nonlinguistic elements, it also seems at first glance to contain better suited semantic relation labeling for the linguistic elements. We need to do an in-depth error analysis on BERTLine's performance on the three versions of the STAC corpus. We would also like to investigate training on one corpus and then running the model on the other corpus, and we would like to do the same with the Molweni corpus.

9 Ethics Statement

Bertline's performance does not seem to pose any ethical difficulties or questions.

Acknowledgements

We gratefully acknowledge support from the 3AI institute ANITI (ANR-19-PI3A-0004) and the ANR project SUMM-RE (ANR-20-CE23-0017).

References

S. Afantenos, E. Kow, N. Asher, and J. Perret. 2015. Discourse parsing for multi-party chat dialogues. In *Empirical Methods in Natural Language Processing*, pages 928–937. Association for Computational Linguistics.

Nicholas Asher. 1993. *Reference to abstract objects in English*, volume 50. D. Reidel, Dordrecht.

Nicholas Asher, Julie Hunter, Mathieu Morey, Farah Benamara, and Stergos Afantenos. 2016. Discourse structure and dialogue acts in multiparty dialogue: The STAC corpus. In *Language Resources and Evaluation Conference*, pages 2721–2727.

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press.

Sonia Badene, Kate Thompson, Jean-Pierre Lorré, and Nicholas Asher. 2019. Weak supervision for learning discourse structure. In *Empirical Methods in Natural Language Processing*.

Grigorii Guz, Patrick Huber, and Giuseppe Carenini. 2020. Unleashing the power of neural discourse parsers–a context and structure aware approach using large scale pretraining. *arXiv preprint arXiv:2011.03203*.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020. Molweni: A challenge multiparty dialogues-based machine reading comprehension dataset with discourse structure. pages 2642–2652.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Zhengyuan Liu and Nancy Chen. 2021. Improving multi-party dialogue discourse parsing via domain integration. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 122–127, Punta Cana, Dominican Republic and Online. Association for Computational Linguistics.

William Mann and Sandra Thompson. 1987. Rhetorical structure theory: A framework for the analysis of texts. *International Pragmatics Association Papers in Pragmatics*, 1:79–105.

Daniel Marcu. 1999. A decision-based approach to rhetorical parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 365–372, College Park, Maryland, USA. Association for Computational Linguistics.

Jérémy Perret, Stergos Afantenos, Nicholas Asher, and Mathieu Morey. 2016. Integer linear programming for discourse parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 99–109, San Diego, California. ACL.

Zhouxing Shi and Minlie Huang. 2019. A deep sequential model for discourse parsing on multi-party dialogues. In *AAAI*.

Ante Wang, Linfeng Song, Hui Jiang, Shaopeng Lai, Junfeng Yao, Min Zhang, and Jinsong Su. 2021. A structure self-aware model for discourse parsing on multi-party dialogues. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3943–3949. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020. Amr-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*, 8:19–33.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better amr-to-text generation. *arXiv preprint arXiv:1909.00136*.

How Far Can It Go? On Intrinsic Gender Bias Mitigation for Text Classification

Ewoenam Kwaku Tokpo[1], **Pieter Delobelle**[2,3,5], **Bettina Berendt**[2,3,4,5] and **Toon Calders**[1]
[1] Department of Computer Science, University of Antwerp, Belgium
[2] Department of Computer Science, KU Leuven, Belgium
[3] Leuven.AI Institute for Artificial Intelligence, Belgium
[4] Faculty of Electrical Engineering and Computer Science, TU Berlin, Germany
[5] Weizenbaum Institute, Germany

Abstract

To mitigate gender bias in contextualized language models, different intrinsic mitigation strategies have been proposed, alongside many bias metrics. Considering that the end use of these language models is for downstream tasks like text classification, it is important to understand how these intrinsic bias mitigation strategies actually translate to fairness in downstream tasks and the extent of this. In this work, we design a probe to investigate the effects that some of the major intrinsic gender bias mitigation strategies have on downstream text classification tasks. We discover that instead of resolving gender bias, intrinsic mitigation techniques and metrics are able to hide it in such a way that significant gender information is retained in the embeddings. Furthermore, we show that each mitigation technique is able to hide the bias from some of the intrinsic bias measures but not all, and each intrinsic bias measure can be fooled by some mitigation technique, but not all. We confirm experimentally, that none of the intrinsic mitigation techniques used without any other fairness intervention is able to consistently impact extrinsic bias. We recommend that intrinsic bias mitigation techniques should be combined with other fairness interventions for downstream tasks.

1 Introduction

The use of pretrained language models has seen a surge in popularity as a result of state-of-the-art performances that have been achieved with these models on various tasks. Consequently, there has been a growing interest in exploring how gender bias pertains in these models (Garrido-Muñoz et al., 2021). Pretrained language models are used in two distinct phases: the *pretraining phase* and the *finetuning phase*. The pretraining phase typically involves training a model on a generic task such as masked language modeling on a diverse set of text corpora. In the finetuning phase, the pretrained model can be adapted for a specific task, such as sentiment analysis, by training on a domain-specific corpus. Owing to the unique way of using pretrained models, bias generally manifests in two forms: *intrinsic bias* and *extrinsic bias*. Intrinsic bias refers to bias that inherently exists in pretrained language models whereas extrinsic bias is used to refer to bias that exists in downstream models that are based on the pretrained model. Since the success of downstream NLP tasks has mostly been dependent on pretrained models, it is intuitive to assume that bias in intrinsic models will compromise fairness in downstream tasks. Only recently have more in-depth examinations been done to investigate this assumption (Steed et al., 2022; Orgad et al., 2022; Kurita et al., 2019). However, conclusions have been inconsistent and the confounding effects of bias mitigation techniques remain unknown.

The main focus of this work is to investigate the impact of intrinsic bias on extrinsic fairness and if techniques to mitigate intrinsic bias actually resolve bias or only mask it. We develop a probe to uncover intrinsic bias by determining the amount of gender information in a word embedding using a classifier. Bearing in mind how abstractly and improperly intrinsic bias has been defined (Blodgett et al., 2020), coupled with the discovery that results from different metrics for intrinsic bias in many cases do not correlate (Delobelle et al., 2022), we find this probe effective as an extra step in evaluating the efficacy of these mitigation strategies. We realize from this study that how intrinsic bias has been measured and the choice of bias mitigation strategies explored by some existing works have not been ideal. We further use this probe to investigate if some proposed mitigation strategies superficially conceal bias.

In this work, when we refer to bias in a language model, we mean stereotyping bias as defined by Garrido-Muñoz et al. (2021) as *"the undesired variation of the [association] of certain words in that language model according to certain prior words*

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3418–3433
May 2-6, 2023 ©2023 Association for Computational Linguistics

in a given domain". We focus our experiments on gender bias for two primary reasons: its intuitive nature making it easy to analyze and discuss, and the accessibility of resources and datasets regarding gender. For the same reason, we narrow our definition of bias in our experiments and analysis to binary gender bias[1]. This paper considers the primary goal of mitigating intrinsic bias to ensure fairness in downstream tasks.

We consider one case of binary classification and one multiclass classification case, all on English language corpora using the BERT-large (Devlin et al., 2019) and the ALBERT-large (Lan et al., 2019) pretrained models for each task.

In summary, we develop an extensive probe [2] to uncover intrinsic bias in pertained contextualized language models, and seek to answer three key research questions: **RQ1:** Do different intrinsic bias metrics respond differently to different bias mitigation techniques? (§ 4.1). **RQ2:** Can intrinsic bias mitigation techniques hide bias instead of resolving it? (§ 5.1). **RQ3:** Do intrinsic bias mitigation techniques in language models improve fairness in downstream tasks? (§ 5.2)

2 Measuring and mitigating bias

Since many techniques for measuring and mitigating gender bias have been introduced for both intrinsic and extrinsic bias, we only discuss techniques we use in experiments in this paper.

Bias mitigation techniques can be applied to pretrained or finetuned language models, or both. Figure 1 illustrates these interactions with both training settings and this section will discuss both intrinsic (§ 2.1) and extrinsic (§ 2.3) mitigation techniques. Additionally, this section will provide a brief overview of bias measures [3].

2.1 Intrinsic bias mitigation techniques

Intrinsic gender bias mitigation methods target either the pretraining data, the pretraining procedure, or the pretrained model's output, which we refer to as pre-processing, in-processing, and post-processing respectively (Friedler et al., 2019). We select three popular mitigation methods to represent all three types, namely *Counterfactual Data Augmentation* (CDA), *Context-debias*, and *Sent-debias*. Notice that these methods create debiased pretrained language models, as is illustrated in Figure 1. These models still need to be finetuned on a downstream task.

CDA pretraining. The idea behind counterfactual data augmentation (Zmigrod et al., 2019; Lu et al., 2020) is to generate a counterfactual for each example in the training corpus by replacing attribute terms with their complimentary equivalent from the other demographic classes. For example, *she* will map to *he* in the case of binary gender. To mitigate intrinsic bias, this counterfactual augmentation has to be done as a pretraining step. Since CDA involves retraining the model, it is more resource-intensive compared to Sent-debias and Context-debias. We use the pretrained CDA models based on BERT and ALBERT from Webster et al. (2020) for our implementation.

Context-debias. Kaneko and Bollegala (2021) introduce a debiasing method that involves retraining the language model with a constraint to make the embeddings of *stereotype terms* [4] (such as *doctor, nurse*) orthogonal to embeddings of *attribute terms* [5] (such as gender pronouns like *she, he* and gender nouns like *woman, man*). Given the dynamic nature of contextualized word embeddings which causes a word to have different embeddings, they define fixed word embeddings for each attribute token by averaging the contextual embeddings of a word in all sentences it appears in. Training is done so that the embeddings of all stereotype terms are made orthogonal to all fixed attribute embeddings. They add a regularizer that constrains the debiased embedding to retain as much information by ensuring that they are as similar to the original embeddings as possible despite the orthogonality (§ B.1). Context-debias, as well as other in-processing techniques, require a predefined set of attribute and target terms before training which may not be effective for words outside these sets.

Sent-debias. Liang et al. (2020) propose a post-processing debiasing method akin to *word embedding debiasing* (Bolukbasi et al., 2016) but for contextualized embeddings. They achieve this by first identifying the bias subspace. They extract naturally-occurring sentences from corpora that

---

[1] See Ethical considerations.

[2] We make our code available at `https://github.com/EwoeT/intrinsic-gender-probe`

[3] For a more in-depth overview of mainly intrinsic measures, we refer to Delobelle et al. (2022).

[4] We also refer to them as *target words*.

[5] We also refer to them as *identity terms* or *protected words*.

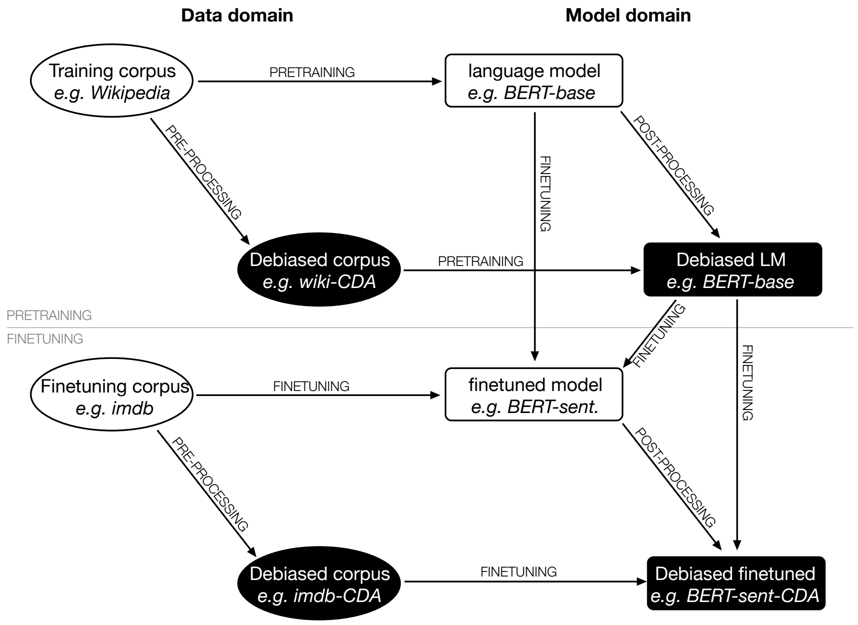


Figure 1: Illustration of where bias mitigation techniques alter a model or training data for both intrinsic (pretrained) and extrinsic (finetuned) biases in models.

contain certain attribute terms, generate counterfactuals for each one, and compute the gender subspace based on these sentence templates. The rest of their algorithm follows the method of hard debiasing (neutralization and equalization) by Bolukbasi et al. (2016) (§ B.2). Essentially, this post-hoc approach transforms the embeddings by removing their projections onto the gender subspace.

2.2 Measuring intrinsic bias

To measure intrinsic bias in the pretrained models, we use *SEAT* and *LPBS*. We examine these two metrics because of their wide use in related literature.

Sentence Embedding Association Test (SEAT). May et al. (2019) developed SEAT to quantify intrinsic bias in contextualized language models based on WEAT (Caliskan et al., 2017) which was originally designed for non-contextualized embeddings. Although the authors express concern over the efficacy of this metric, it has been widely adopted in various works to quantify bias in contextualized word embeddings. We adopt the implementation by Tan and Celis (2019) that uses word-level contextualized embeddings of attribute and stereotype terms instead of sentence-level embeddings. Delobelle et al. (2022) show that using word-level embeddings produces more consistent results and is more robust against the effects of template choices. SEAT is defined as: $s(X_f, X_m, A, B) = \sum_{x_f \in X_f} s(x_f, A, B) - \sum_{x_m \in X_m} s(x_m, A, B)$; A and B are the respective sets of female and male attribute (identity) terms, whereas X_f and X_m are the sets of female and male stereotypes respectively, The similarity measure $s(x, A, B)$ gives the association between a stereotype word's embedding and the word embeddings of attributes: $s(x, A, B) = \frac{1}{|A|} \sum_{a \in A} \cos(x, a) - \frac{1}{|B|} \sum_{b \in B} \cos(x, b)$

Log Probability Bias Score (LPBS). This template-based approach proposed by Kurita et al. (2019) quantifies bias using templates containing target words: such as "_ *is a doctor.*". They compute the difference in probability scores that a language model uses to predict words from two respective groups (eg. *female:she, male:he*) to fill in the blank. To account for the effect of prior probabilities of gender words that may skew results, they normalize the results by dividing each prediction by the prior probability of the attribute term. We compute LPBS as: $LPBS = \sum_{x \in X} \sum_i |ls(a_i, x) - ls(b_i, x)|$; where a_i and b_i are equivalent forms of a female and male attribute term respectively, $X = X_f \cup X_m$ is the set of all stereotype words, $ls(w, x) = \log(\frac{P(w|x)}{P(w)})$ is the association between an attribute term $w \in A \cup B$ and a target term $x \in X$, $P(w)$ is the prior probability of w, and $P(w|x)$ is the probability of predicting w as the blank term given the presence of x in its context.

2.3 Extrinsic bias mitigation techniques

We look at two downstream tasks, which are both text classification (Bias-in-Bios (De-Arteaga et al., 2019a) and Jigsaw[6])[7]. To understand the relationship between intrinsic and extrinsic bias, in addition to these original datasets, we include two preprocessed versions of the datasets using *Attribute scrubbing* and *Attribute swapping*. We use these settings to analyze how intrinsic bias is propagated to downstream tasks.

Attribute scrubbing. Attribute features, which are identity terms in an NLP task, are completely removed from text instances in the training data (Prost et al., 2019; De-Arteaga et al., 2019b). This approach may seem intuitive, particularly to mitigate explicit bias, but we suggest that deleting tokens in NLP tasks can be tricky since it could change the syntactic and grammatical structure of the text, leading to out-of-distribution problems.

For datasets such as Bias-in-Bios (De-Arteaga et al., 2019a), a lot of existing works have simply relied on the scrubbed version provided in the dataset, however, we realize that the set of gender words scrubbed is very limited, hence, we extend this to include all gender words from (Kaneko and Bollegala, 2021) and gender names from (Hall Maudslay et al., 2019) to make the approach more effective.

Attribute swapping. CDA has been explored and proposed as another bias mitigation strategy for downstream tasks, particularly in coreference resolution (Lu et al., 2020) and text classification (Park et al., 2018). The idea is to generate counterfactuals by identifying attribute terms in the text instances and swapping them with equivalent terms belonging to the complementary group. We will term this approach *Attribute swapping* to distinguish it from pretrained CDA in Section 2.1. For pretrained CDA, the language model is (further) pretrained on a set of general corpora on Masked Language Modeling task, whereas in attribute swapping, counterfactuals are generated for each instance in the training data based on attribute terms; with the same y labels assigned to both the original and counterfactual examples. Whereas previous versions of CDA ignore names of people, which are major demographic cues, Hall Maudslay et al. (2019) propose a name intervention to generate counterfactuals for names as well. We use this adaptation for attribute swapping.

2.4 Measuring extrinsic bias

True positive rate difference (TPRD). We use TPRD used in related works (De-Arteaga et al., 2019a; Prost et al., 2019; Jin et al., 2021; Steed et al., 2022). TPRD measures the gap in true positive rates between the predictions for two demographic groups: $TPRD = P(\hat{y} = 1|y = 1, A = a) - P(\hat{y} = 1|y = 1, A = a')$.

Counterfactual Fairness. Since the test dataset follows the same biased distribution of the training set, we also measure the disparity in performance for an individual test instance if the gender attributes are swapped. We measure counterfactual fairness (Kusner et al., 2017) as the difference in the probability of a positive prediction between a test example and its counterfactual, following $CF = \text{mean}(P(|x(\hat{Y}_{A\leftarrow a} = y|A = a, X = x) - P(\hat{Y}_{A\leftarrow a'} = y|A = a, X = x)|))$.

3 Probing for gender information

We design a probe to investigate how much gender information is retained after *mitigating* bias in pretrained language models. Our goal is to measure the amount of gender information in the resulting embeddings of language models after mitigation techniques from § 2.1 have been applied to them.

Defining attribute and stereotype terms. We first identify two sets of attribute terms that define females and males respectively. Let $W_f = \{she, female, woman, ...\}$ and $W_m = \{he, male, man, ...\}$ be the respective sets of female and male gender-defining words. The set $W = W_f \cup W_m$ is the union set of gender-defining words for both females and males. Secondly, we define two sets of stereotype terms which are composed of target words associated with females and males respectively[8]. Let $X_f = \{cheerleader, nurse, softball, ...\}$ and $X_m = \{warrior, baseball, engineer, ...\}$ be the respective sets of female and male stereotype words such that $X = X_f \cup X_m$.

Extracting word embeddings. We extract all sentences from a text corpus containing words in each set of words. Especially for stereotype terms, we ensure that the sentences do not also

---

[6] https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/data?select=train.csv

[7] See Appendix D for dataset description

[8] We use both attribute and stereotype wordlists from (Kaneko and Bollegala, 2021)

include words in the attribute wordlist since attribute words in the same context can introduce gender information in the embedding produced. Let $S_w = \{s_{w_1}, ..., s_{w_n}\}$ be the set of all sentences containing a word w. We use the pretrained language model to extract embeddings for w such that $w \rightarrow E_w = \{e_{w_1}, ..., e_{w_n}\}$, where $e_{w_i} \in \mathbb{R}^n$ is the embedding for token w in sentence s_{w_i}. If w appears in s_{w_i} multiple times, then $|E_w| > n$.

Training the classifier. We split W into a train and test sets W_T and W_I respectively such that $W_T \cap W_I = \emptyset$. This is to verify that the trained model is not merely learning to identify words that appear in the train set rather than words containing gender information in general. We reserve the set of stereotypes X solely for evaluating bias (inference) in the language models.

Taking $E_f = \{e_{f1}, e_{f2}, ...\}$ and $E_m = \{e_{m1}, e_{m2}, ...\}$ to be the set of all embeddings for female and male words respectively, given $E = E_f \cup E_m$, we train a classifier $\mathcal{C} : E \rightarrow \{f, m\}$ to predict whether an embedding e belongs to E_f or E_m.

Evaluating bias. To determine the conformity of a language model $\mathcal{L}$ to gender stereotypes X, we introduce two bias criteria (or notions of bias):

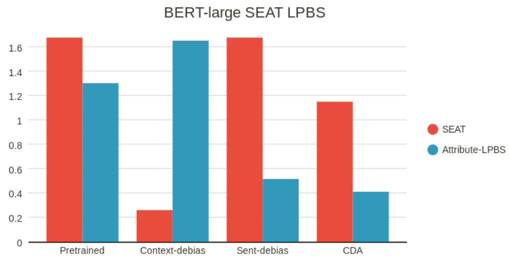
Bias accuracy: We compute the accuracy of $\mathcal{C}$ to correctly predict the gender association of all stereotype words in X based on the embeddings produced by $\mathcal{L}$. A high accuracy depicts the presence of pro-stereotype information in the embeddings produced by the language model $\mathcal{L}$, even though the predictions could be of low confidence i.e. just enough gender information to correctly predict the gender association.

Mean bias confidence: We also consider the mean of the individual softmax scores for each prediction. In the case of binary gender, a softmax value of 0.5 means the embedding is perfectly neutral, hence, we define mean bias confidence as $\frac{1}{N}\sum_i^N |bias(e_i) - 0.5|$. High mean confidence depicts an enormous amount of gender information contained in the embedding, although this information may not necessarily be pro-stereotype.

Randomization test. To test the efficacy of our method, We carry out a randomization test by iteratively splitting $X = X_f \cup X_m$ into two random groups $X_{f'}$ and $X_{m'}$. We use $X'^{(i)} = \{X_{f'}^{(i)}, X_{m'}^{(i)}\}$ to define the random pair generated in the ith iteration. We repeat this randomized split over 100 iterations; $X'^{(i)} \in \{X'^{(1)}, ..., X'^{(100)}\}$. By computing a p-value based on a one-sample t-test, we compute if the mean of accuracy scores from the random samples X' is significantly different from the score from$\{X_f, X_m\}$.

4 Investigating intrinsic bias in mitigated models with existing metrics

We first use the two existing metrics defined in § 2.2 (SEAT and LPBS) to quantify bias in BERT-large and ALBERT-large pretrained models. Using the same metrics, we then explore how intrinsic bias changes in mitigated versions of these models using the mitigation techniques outlined in § 2.1: Context-debias, Sent-debias, and CDA pretraining. This is important to evaluate the efficacy of these mitigation strategies, and to make relevant postulations.



(a) BERT-large model

Figure 2: Intrinsic bias scores using SEAT and LPBS. Results show inconsistencies in measuring bias between SEAT and LPBS for various mitigation strategies — lower scores are desirable in both cases.

4.1 RQ1: Do intrinsic bias measures respond differently to bias mitigation techniques?

In Figure 2, we plot the intrinsic bias results[9] of SEAT and LPBS on BERT-large. The first observation we point out, particularly for Context-debias and Sent-debias, is the contradicting nature of bias scores obtained from SEAT and LPBS. Whereas SEAT shows a drastic reduction in intrinsic bias for Context-debias, LPBS indicates worsening bias. The converse is true for Sent-debias where LPBS shows a significant reduction in intrinsic bias but SEAT shows worse scores. This confirms our suspicion that different mitigation strategies respond differently to different metrics.

---

[9] see Table 1 and Figure 8 for full results.

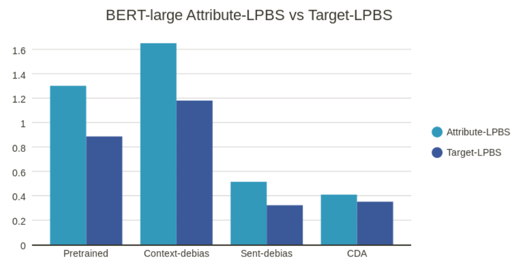


Figure 3: We compare LPBS in its default form (attribute-LPBS) and target-LPBS for BERT-large. The plot shows a general positive correlation between the two metrics. In both cases, Sent-debias maintains the lowest bias score — lower scores are desirable in both cases. See Figure 9 for ALBERT-large.

The use of some metrics can be problematic for evaluating some mitigation techniques for intrinsic bias. We find that some metrics are not ideal to use with some mitigation techniques due to differences in how various mitigation techniques interact with a language model. Consider a metric like LPBS which measures the association between a stereotype term and a gender $\frac{P(A{=}a|X{=}x)}{P(A{=}a)}$. This association is captured at the inner layers of the model, but since post-hoc approaches like Sent-debias do not alter the internal representation of the model, LPBS in its original form becomes ineffective — Figure 7. We refer to this original form of LPBS as ***attribute-LPBS***. Sent-debias ends up debiasing the attribute terms rather than the stereotype terms, thus, constantly producing low probability difference scores. Even so, we find LPBS being used in this form with Sent-debias in works such as (Steed et al., 2022). Using stereotype terms instead of attribute terms $P(X = x|A = a)$, which we refer to as ***target-LPBS***, is an option to solve this discrepancy, but given that many stereotype terms are usually out of vocabulary, these models will resort to wordpiece tokenization (Sennrich et al., 2016) which will be more challenging to handle. Nonetheless, in Figure 3, when the two versions of LPBS are compared, Sent-debias continues to have the best intrinsic fairness scores in both cases [10].

5 Probing further to uncover bias

We use the methodology in Section 3 to probe for bias in pretrained language models and their bias-mitigated versions to see if the intrinsic bias mitigation techniques actually mitigate biases or superficially conceal them from commonplace metrics like SEAT and LPBS. Gonen and Goldberg (2019) discovered that, for non-contextualized word embeddings, the bias mitigation techniques proposed by (Bolukbasi et al., 2016) were hiding bias instead of resolving bias. We use our probe to see if this is equally the case for intrinsic bias mitigation in contextualized embeddings. The consequence of mitigation techniques superficially hiding bias is that downstream classifiers can learn to pick up residual traces of gender information.

We validate our results by carrying out a randomization test, as described in Section 3, to test for statistical significance of our results. We conduct the test under the null hypothesis H_0 that the prediction accuracy of our original gender-stereotype sets will not differ significantly from the random split. We show in Table 2 the very low p-values that indicate statistical significant results, thus refuting H_0.

5.1 RQ2: Can intrinsic bias techniques and metrics hide bias instead of resolving it?

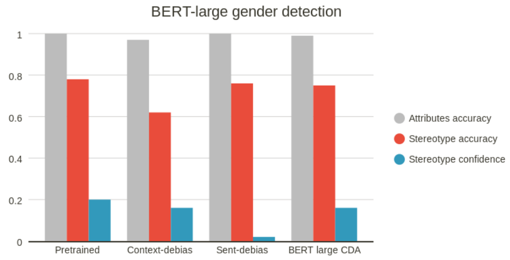


Figure 4: Gray bars indicate the accuracy of detecting gender information in attribute terms (high scores are desirable). Red bars indicate the accuracy of detecting gender information in stereotype terms (this should ideally be 0.5 in a fair classifier, showing the inability to correctly predict the gender association of stereotype terms). Blue bars indicate the average confidence of prediction: $\frac{1}{N}\sum_i^N |bias(e_i) - 0.5|$ (low scores are desirable).

In Figure 4, we show the results[11] of our probe for BERT-large. We highlight some key observations.

Bias accuracy and mean bias confidence do not correlate. We see from both plots in Figure 4

---

[10] For this comparison we only selected single-wordpiece attribute words to avoid challenges with multiple wordpieces.

[11] see Table 2 and Figure 10 for full results.

that, counter-intuitively, high bias accuracy does not concur with high mean bias confidence. Actually, Sent-debias, which consistently has the highest bias accuracy among the mitigation techniques, consistently has the lowest mean bias confidence. This indicates that these two notions of bias may not correspond. This, coupled with the finding in Figure 2 about the inconsistency between SEAT and LPBS, suggests that intrinsic bias could be perceived or measured from conflicting perspectives.

Some mitigation techniques can conceal gender information. For Sent-debias in ALBERT-large, we can, almost to the same degree of accuracy as for the unmitigated pretrained model, correctly predict the association of stereotypes to their respective genders using a simple linear classifier even with a low confidence. This will render a mitigation technique useless if a downstream classifier can correctly predict gender. This could only indicate that although models like Sent-debias significantly reduce the degree of association between stereotypes and their respective genders, hence the low confidence in prediction, this reduction does not inhibit the capability to correctly predict the associated gender of stereotypes; confirming that an intrinsic mitigation process can be rendered useless for downstream fairness even if the degree of association (confidence) is significantly reduced, so long as the information retained remains just enough to correctly associate the gender groups with their associated stereotypes.

5.2 RQ3: Do intrinsic bias mitigation techniques in language models improve fairness in downstream tasks?

To investigate the relationship between intrinsic and extrinsic gender bias, we train both BERT-large and ALBERT-large together with their intrinsic-bias-mitigated versions on the Bias-in-bios and Jigsaw datasets using a 10-fold train-test scheme to correctly predict the profession of each profile. We evaluate extrinsic bias based on gender TPRD and gender CF scores described in § 2.4. We conduct the test under three settings: 1) Using the original version of the training data. 2) Using our attribute scrubbed version (see Section 2.3). 3) Using the attribute-swap augmented version of the data (see Section 2.3). These three settings of the dataset are important to understand how intrinsic bias affects downstream fairness under different data settings.

In Figure 5, we show the extrinsic bias score [12] for BERT-large and its *debiased* versions, and the corresponding downstream data type for the Bias-in-Bios dataset — see Figure 11 and Figure 12 for all models and datasets.

Intrinsic mitigation techniques do not show a significant improvement on TPRD. We observe that all the intrinsic bias mitigation interventions we consider do not significantly improve bias in TPRD on their own. There is only a slight improvement or worsening bias in some cases. For Context-debias, we postulate this could be due to the limited list of words used to mitigate bias. If these limited words are not key terms in the downstream task or do not exist in the downstream text instances, the mitigation may not have a consequential effect on the downstream task.

Downstream data processing significantly improves downstream fairness. Secondly, we realize that the downstream data has a lot more material effect on downstream fairness. The data processing techniques: *attribute scrubbing and attribute swapping* showed improvements in TPRD producing the best results.

Combining intrinsic CDA and Downstream CDA produces best TPRD results. Thirdly, we realize that although the downstream data seems key to downstream fairness, a combination of intrinsic and downstream data intervention produces even better extrinsic fairness results.

Sent-debias significantly improves CF scores. Sent-debias is the only intrinsic technique that shows significant improvement in counterfactual fairness. Context-debias and CDA produce worse CF scores in some cases. We observe that the mode of application of Sent-debias in downstream classification tasks does not entirely align with the notion of intrinsic since gender information is removed from the sentence-level representation instead of word-level representation.

Combining Sent-Debias and fair downstream data produces the best CF scores. We also realize from BERT (Figure 5) and ALBERT that combining Sent-debias as a mitigation technique and a downstream data intervention produces the best CF scores.

---

[12] Full results with error margins are given in Table 3, Table 4, Table 5 and Table 6.

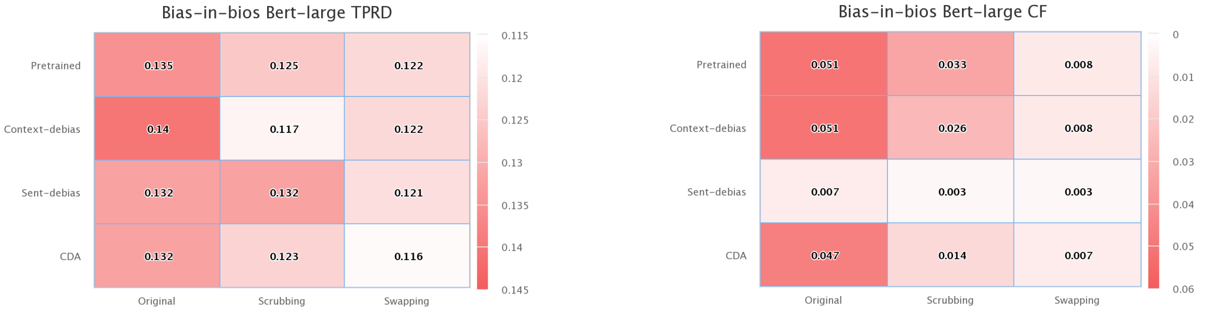


Figure 5: How intrinsic mitigation and downstream data intervention interact to influence fairness on the bias-in-bios data — BERT-large.

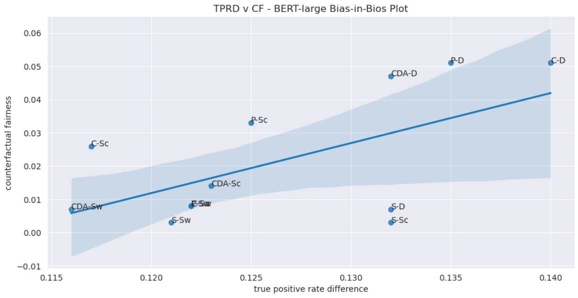


Figure 6: Relationship between TPRD and CF based on results from BERT model and Bias-in-Bios dataset. See Figure 13 for ALBERT-large.

TPRD vs Counterfactual-fairness. Here, we look at the relationship between TPRD and counterfactual fairness. As both measures are used as extrinsic fairness metrics, we aim to see how both results correlate. From Figure 6 we find a positive correlation in general between the two metrics. The downstream data interventions tend to improve both TPRD and CF the most.

6 Related work

Park et al. (2018) study the effects of three techniques to mitigate gender bias in abusive language detection; *debiased word embeddings* (based on the approach by Bolukbasi et al. (2016)), *gender swap* (counterfactual augmentation of classification data), *bias finetuning* using non-contextualized word embeddings: word2vec (Mikolov et al., 2013) and FastText (Bojanowski et al., 2017). They found that a combination of debiasing and gender swapping produced the best fairness results when implemented with a Gated Recurrent Unit (GRU).

Prost et al. (2019) examine three debiasing techniques for mitigating gender bias in downstream text classification using bias-in-bios (De-Arteaga et al., 2019b), with non-contextualized embeddings. Specifically, they examine *scrubbing* (deletion of identity terms) and approaches that focus on non-contextualized embeddings namely: *debiasing* (based on neutralization and equalization by Bolukbasi et al. (2016)), and *strong debiasing* Prost et al. (2019). Experimenting with Glove embeddings (Pennington et al., 2014), they show that strong debiased produces the best fairness results whilst maintaining a good accuracy score (second only to the unmitigated model). They again show that standard debiasing (Bolukbasi et al., 2016) can be counter-productive in terms of fairness; it can rather reduce fairness in downstream text classification.

Steed et al. (2022) meticulously investigate the effect of intrinsic bias in downstream text classification in RoBERTa. They also conclude that intrinsic/upstream bias does not significantly contribute to downstream bias. However, as we have previously discussed, using the default form of Log probability Bias score (attribute-LPBS) (Kurita et al., 2019) which they adopt to measure intrinsic bias may not be an effective technique to use with Sent-debias.

Orgad et al. (2022), the closest and concurrent with our work, investigate the relationship between intrinsic and extrinsic bias and also develop a probe based on a classification-based technique *minimum description length* (Voita and Titov, 2020) to detect bias in RoBERTa. They reach similar conclusions indicating how intrinsic metrics, like CEAT in their case, could conceal bias.

7 Conclusion

In this paper, we develop a probe to investigate intrinsic bias in two language models, BERT-large and ALBERT-large on two text classification tasks.

We use this probe to answer key research questions outlined for this research. We find that intrinsic bias metrics can be sensitive to certain intrinsic bias mitigation techniques. We also show that intrinsic bias mitigation techniques and metrics are capable of concealing bias instead of resolving it. We discover that intrinsic bias mitigation techniques we considered do not significantly improve fairness in text classification when used without other fairness interventions like data pre-processing. We recommend that intrinsic bias mitigation should ideally be combined with other fairness interventions.

Limitations

One obvious limitation of this work is the restriction to the use of English language as a basis. In future, we will explore how these conclusions vary in other language settings. Secondly, although we aimed to make the list of gender and attribute terms used in our experiments as extensive as possible, it is nearly impossible to cover all possible gender and target words in such contexts. Nonetheless, for the scope under consideration, we believe our compilation is extensive enough to give a general outlook and to draw the necessary conclusions. Finally, our work is heavily based on binary notions of gender which is a limitation considering the growing use of non-binary categorizations of gender and other biases which we outline in the ethical considerations section.

Ethical considerations

For practical reasons such as access to datasets and resources on gender bias, we limit our work to a binary representation of gender. We draw readers' attention to the fact that non-binary gender representations are nuanced and intricate (Dev et al., 2021), as such, this should be in cognizance when applying conclusions from this work in non-binary settings. Nonetheless, considering binary gender as a base form of gender categorization, the insights and conclusions from this work can form the baseline for exploring more complex gender categorizations.

Acknowledgements

Ewoenam Kwaku Tokpo and Pieter Delobelle received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme. Pieter Delobelle was also supported by the Research Foundation - Flanders (FWO) under EOS No. 30992574 (VeriLearn). Bettina Berendt received funding from the German Federal Ministry of Education and Research (BMBF) – Nr. 16DII113.

References

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019a. Bias in bios: A case study of semantic representation bias in a high-stakes setting. New York, NY, USA. Association for Computing Machinery.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019b. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Pieter Delobelle, Ewoenam Tokpo, Toon Calders, and Bettina Berendt. 2022. Measuring fairness with biased rulers: A comparative study on bias metrics for pre-trained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1693–1706, Seattle, United States. Association for Computational Linguistics.

Sunipa Dev, Masoud Monajatipoor, Anaelia Ovalle, Arjun Subramonian, Jeff Phillips, and Kai-Wei Chang. 2021. Harms of gender exclusivity and challenges in non-binary representation in language technologies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages

1968–1994, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Sorelle A Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P Hamilton, and Derek Roth. 2019. A comparative study of fairness-enhancing interventions in machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 329–338.

Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L Alfonso Ureña-López. 2021. A survey on bias in deep nlp. *Applied Sciences*, 11(7):3184.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *NAACL*.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5267–5275, Hong Kong, China. Association for Computational Linguistics.

Xisen Jin, Francesco Barbieri, Brendan Kennedy, Aida Mostafazadeh Davani, Leonardo Neves, and Xiang Ren. 2021. On transferability of bias mitigation effects in language model fine-tuning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3770–3783, Online. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2021. Debiasing pre-trained contextualised embeddings. In *Proc. of the 16th European Chapter of the Association for Computational Linguistics (EACL)*.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. *Advances in neural information processing systems*, 30.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Paul Pu Liang, Irene Mengze Li, Emily Zheng, Yao Chong Lim, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Towards debiasing sentence representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5502–5515, Online. Association for Computational Linguistics.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2020. Gender bias in neural natural language processing. In *Logic, Language, and Security*, pages 189–202. Springer.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Hadas Orgad, Seraphina Goldfarb-Tarrant, and Yonatan Belinkov. 2022. How gender debiasing affects internal model representations, and why it matters. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2602–2628, Seattle, United States. Association for Computational Linguistics.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804, Brussels, Belgium. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Flavien Prost, Nithum Thain, and Tolga Bolukbasi. 2019. Debiasing embeddings for reduced gender bias in text classification. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 69–75, Florence, Italy. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ryan Steed, Swetasudha Panda, Ari Kobren, and Michael Wick. 2022. Upstream Mitigation Is *Not* All You Need: Testing the Bias Transfer Hypothesis in Pre-Trained Language Models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3524–3542, Dublin, Ireland. Association for Computational Linguistics.

Yi Chern Tan and L. Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models.

Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661, Florence, Italy. Association for Computational Linguistics.

Appendix

A Word lists for experiments

We generate the wordlist we use, given below, from the more extensive wordlist from (Kaneko and Bollegala, 2021). To mitigate the effect of multi-wordpiece tokens, we only select tokens with a single wordpiece. Hence we obtain the following wordlists:

A.1 attribute terms

- **female list:** ’witches’, ’mothers’, ’diva’, ’actress’, ’mama’, ’dowry’, ’princess’, ’abbess’, ’women’, ’widow’, ’ladies’, ’madam’, ’baroness’, ’niece’, ’lady’, ’sister’, ’nun’, ’her’, ’mare’, ’convent’, ’ladies’, ’queen’, ’maid’, ’chick’, ’empress’, ’mommy’, ’feminism’, ’gal’, ’estrogen’, ’goddess’, ’aunt’, ’hostess’, ’wife’, ’mom’, ’females’, ’ma’, ’belle’, ’maiden’, ’witch’, ’miss’, ’cow’, ’granddaughter’, ’her’, ’mistress’, ’nun’, ’actresses’, ’girlfriend’, ’lady’, ’maternal’, ’ladies’, ’sorority’, ’duchess’, ’ballerina’, ’fiancee’, ’wives’, ’maternity’, ’she’, ’heroine’, ’queens’, ’sisters’, ’stepmother’, ’daughter’, ’lady’, ’daughters’, ’mistress’, ’hostess’, ’nuns’, ’priestess’, ’filly’, ’herself’, ’girls’, ’lady’, ’vagina’, ’wife’, ’mother’, ’female’, ’womb’, ’heiress’, ’waitress’, ’woman’, ’bride’, ’grandma’, ’bride’, ’gal’, ’lesbian’, ’ladies’, ’girl’, ’grandmother’, ’mare’, ’maternity’, ’nuns’
- **male list:** ’wizards’, ’fathers’, ’actor’, ’bachelor’, ’papa’, ’dukes’, ’hosts’, ’airmen’, ’penis’, ’prince’, ’governors’, ’abbot’, ’men’, ’gentlemen’, ’sir’, ’baron’, ’gods’, ’nephew’, ’lord’, ’brother’, ’priest’, ’his’, ’marquis’, ’princes’, ’emperors’, ’stallion’, ’chairman’, ’monastery’, ’priests’, ’king’, ’spokesman’, ’tailor’, ’cowboys’, ’dude’, ’emperor’, ’daddy’, ’guys’, ’guy’, ’godfather’, ’god’, ’patriarch’, ’uncle’, ’sir’, ’brotherhood’, ’host’, ’testosterone’, ’husband’, ’dad’, ’steward’, ’males’, ’pa’, ’beau’, ’stud’, ’bachelor’, ’wizard’, ’sir’, ’bull’, ’grandson’, ’horsemen’, ’rooster’, ’bachelor’, ’him’, ’master’, ’lad’, ’policeman’, ’monk’, ’actors’, ’boyfriend’, ’statesman’, ’paternal’, ’landlord’, ’brethren’, ’lords’, ’fraternity’, ’duke’, ’fiance’, ’colts’, ’husbands’, ’he’, ’businessman’, ’hero’, ’deer’, ’kings’, ’brothers’, ’masters’, ’stepfather’, ’son’, ’cowboy’, ’sons’, ’baritone’, ’salesman’, ’monks’, ’lads’, ’congressman’, ’priest’, ’barons’, ’beard’, ’colt’, ’himself’, ’boys’, ’lions’, ’gentleman’, ’penis’, ’his’, ’bulls’, ’uncles’, ’lion’, ’sorcerer’, ’father’, ’male’, ’sperm’, ’prostate’, ’businessmen’, ’heir’, ’waiter’, ’headmaster’, ’man’, ’governor’, ’god’, ’grandpa’, ’groom’, ’dude’, ’gay’, ’boy’, ’grandfather’, ’priests’

A.2 List of stereotypes(target words)

- **female list:** ’beauty’, ’blonde’, ’cheerleading’, ’cooking’, ’flirt’, ’flower’, ’gossip’, ’housekeeper’, ’jewelry’, ’nanny’, ’nurse’, ’nurses’, ’pink’, ’pretty’, ’receptionist’, ’sewing’, ’softball’, ’makeup’, ’therapist’, ’maid’, ’chairperson’
- **male list:** ’warrior’, ’strong’, ’terrorist’, ’soldier’, ’sniper’, ’baseball’, ’sergeant’, ’body-

guard', 'boss', 'boxer', 'captain', 'carpenter', 'chancellor', 'colonel', 'commander', 'conductor', 'diplomat', 'drummer', 'engineer', 'gangster', 'geek', 'guitarist', 'industrialist', 'marshal', 'mechanic', 'philosopher', 'physicist', 'scientist', 'rapper', 'mechanic', 'carpenter', 'clergy'

B Intrinsic metrics

B.1 Context-debias

$$L_i = \sum_{t \in \mathcal{V}_t} \sum_{x \in \Omega(t)} \sum_{a \in \mathcal{V}_a} (v_i(a)^T E_I(t, x; \theta_e))^2))$$

$$L_{reg} = \sum_{x \in A} \sum_{w \in x} \sum_{i=1} ||E_i(w, x; \theta_e) - E_i(w, x; \theta_pre)||^2$$

$$L = \alpha L_i + \beta L_{reg}$$

Where L_i is the orthogonality constraint such that $E_i(w, x; \theta_e)$ denotes the embedding of token w in the i-th layer of a contextualised word embedding model E with parameters θ_e. $v_i(a)$ is the non-contextualised embedding of an attribute word a. L_{reg} is a regularizer that constrains the Euclidean distance between the contextualized word embedding of a word w in the ith layer in the original pretrained model with parameters θ_{pre}.

B.2 Sent-debias

$$v = PCA_k(\cup_{j=1} \cup_{w \in \mathcal{R}_j} (w - \mu_j))$$

$$h_v = \sum_{j=1}^{k} \langle h, v_j \rangle v_j$$

$$\hat{h} = h - h_v$$

where v represents the top-k gender subspace, h the representation of a given embedding, h_v the projection of h onto v, and $\hat{h}$ the resulting debiased subspace.

C Experiment set-up

We train all models for 4 epochs, a learning rate of $2e^{-5}$ on Tesla V100-SXM3-32GB. We use a sequence length of 100 as the default for all extrinsic tasks. For all downstream tasks, we use a one-layer linear classifier.

D Datasets

Bias-in-bios (De-Arteaga et al., 2019a) consists of online English biographies of people. We use this dataset to predict the occupations of people. Since the gender labels are provided, we can compute disparities in predictive performance between male and female gender groups.

We select the top 7 female and top 7 male professions based on the gender percentages. We take these two sets to represent female and male dominated jobs respectively.

Jigsaw dataset [13] from the online platform *Civil comments* contains online comments. This dataset is scored from 0 to 10 with the perceived gender of the targets of these comments as well as the toxicity levels of these comments. We use this dataset to predict whether or not a comment is toxic against the target persons. We use gender labels to compute predictive disparities.

We select all comments with toxicity score above 0.5 as the toxic comments and those with a score of exactly 0 as the non-toxic ones in order to eliminate fine-margins. We use the same technique to annotate gender by labeling a gender positive if it scores above 0.5 whilst its complimentary gender has a score of exactly 0.

E Measuring intrinsic bias with LPBS when debiasing is done with Sent-debias

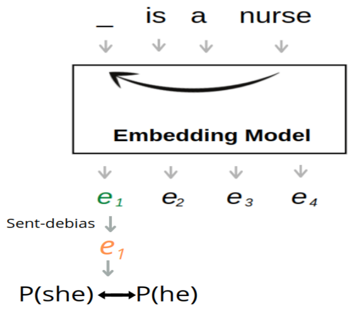


Figure 7: Measuring intrinsic bias in Sent-debias with target-LPBS.

F Results

---

[13] https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/data

Table 1: Intrinsic bias scores from SEAT and LPBS. For SEAT we capture the *test statistic score*: **SEAT test score** and the *effect size*: **SEAT eff. size**. For LPBS we capture results from two variants: *attribute-LPBS*: **attr-LPBS** and *target-LPBS*: **target-LPBS**.

(a) Intrinsic bias BERT-large-uncased

Model	**SEAT test score**	**SEAT eff. size**	**attr-LPBS**	**target-LPBS**
pretrained	0.055	1.675	1.301	0.886
Context-debias	0.002	0.260	1.650	1.180
Sent-debias	0.055	1.675	0.514	0.322
CDA	0.033	1.148	0.409	0.351

(b) Intrinsic bias ALBERT-large

Model	**SEAT test score**	**SEAT eff. size**	**attr-LPBS**	**target-LPBS**
pretrained	0.051	1.715	0.870	0.446
Context-debias	0.003	0.244	1.343	0.346
Sent-debias	0.051	1.717	0.187	0.111
CDA	0.006	1.170	0.654	0.772

Table 2: Results of gender detection probe: **Gender acc** denotes the accuracy of detecting gender information in attribute/gender terms (high scores are desirable). **Stereotype acc** indicates the accuracy of detecting gender information in stereotype terms (this should ideally be 0.5 in a fair classifier, showing the inability to correctly predict the gender association of stereotype terms). **Stereotype conf** indicate the average confidence of prediction of stereotypes: $\frac{1}{N}\sum_{i}^{N}|bias(e_i) - 0.5|$ (low scores are desirable). P-values are from the randomization test.

(a) Detecting gender information BERT-large-uncased

Model	**Gender acc**	**Stereotype acc**	**Stereotype conf**	**p-value**
pretrained	1.0	0.78	0.20	9.055e-73
Context-debias	0.97	0.62	0.16	1.398e-51
Sent-debias	1.0	0.76	0.02	8.772e-70
CDA	0.99	0.75	0.16	7.824e-77

(b) Detecting gender information Alert-large-uncased

Model	**Gender acc**	**Stereotype acc**	**Stereotype conf**	**p-value**
pretrained	1.0	0.83	0.33	2.763e-71
Context-debias	0.97	0.63	0.19	1.700e-48
Sent-debias	1.0	0.82	0.02	2.335e-95
CDA	0.94	0.63	0.12	3.497e-61

Table 3: Extrinsic bias scores from Bias in Bios - BERT-large: We capture true positive rate difference, false positive rate difference, counterfactual fairness, accuracy scroes for both gender groups, counterfactual augmented true positive rate difference and counterfactual augmented false positive rate diference scores.

Model	**Data**	**TPRD**	**FPRD**	**ACC-F**	**ACC-M**	**CF**	**CF-TPRD**	**CF-FPRD**	**CF-ACC-F**	**CF-ACC-M**
Pretrained	Default	0.135± 0.03	0.012± 0.0	0.984± 0.0	0.982± 0.0	0.051± 0.0	0.116± 0.02	0.008± 0.0	0.982± 0.0	0.981± 0.0
Pretrained	Scrubbing	0.125 ± 0.02	0.011 ± 0.0	0.984 ± 0.0	0.982 ± 0.0	0.033 ± 0.0	0.113 ± 0.02	0.008 ± 0.0	0.983 ± 0.0	0.981 ± 0.0
Pretrained	Swapping	0.122 ± 0.02	0.008 ± 0.0	0.983 ± 0.0	0.981 ± 0.0	0.008 ± 0.0	0.12 ± 0.02	0.008 ± 0.0	0.983 ± 0.0	0.981 ± 0.0
Context-debias	Default	0.14 ± 0.03	0.012 ± 0.0	0.984 ± 0.0	0.982 ± 0.0	0.051 ± 0.0	0.119 ± 0.02	0.008 ± 0.0	0.982 ± 0.0	0.98 ± 0.0
Context-debias	Scrubbing	0.117 ± 0.03	0.01 ± 0.0	0.984 ± 0.0	0.981 ± 0.0	0.026 ± 0.0	0.109 ± 0.03	0.009 ± 0.0	0.983 ± 0.0	0.981 ± 0.0
Context-debias	Swapping	0.122 ± 0.03	0.009 ± 0.0	0.983 ± 0.0	0.981 ± 0.0	0.008 ± 0.0	0.12 ± 0.03	0.009 ± 0.0	0.983 ± 0.0	0.981 ± 0.0
CDA	Default	0.132 ± 0.02	0.012 ± 0.0	0.983 ± 0.0	0.982 ± 0.0	0.047 ± 0.0	0.111 ± 0.02	0.008 ± 0.0	0.982 ± 0.0	0.98 ± 0.0
CDA	Scrubbing	0.123 ± 0.03	0.009 ± 0.0	0.982 ± 0.0	0.981 ± 0.0	0.014 ± 0.0	0.121 ± 0.03	0.009 ± 0.0	0.982 ± 0.0	0.981 ± 0.0
CDA	Swapping	0.116 ± 0.02	0.009 ± 0.0	0.983 ± 0.0	0.981 ± 0.0	0.007 ± 0.0	0.116 ± 0.02	0.009 ± 0.0	0.983 ± 0.0	0.981 ± 0.0
Sent-debias	Original	0.132 ± 0.02	0.012 ± 0.0	0.983 ± 0.0	0.982 ± 0.0	0.007 ± 0.0	0.115 ± 0.02	0.008 ± 0.0	0.981 ± 0.0	0.98 ± 0.0
Sent-debias	Scrubbing	0.132 ± 0.02	0.01 ± 0.0	0.983 ± 0.0	0.981 ± 0.0	0.003 ± 0.0	0.122 ± 0.03	0.008 ± 0.0	0.982 ± 0.0	0.98 ± 0.0
Sent-debias	Swapping	0.121 ± 0.02	0.009 ± 0.0	0.982 ± 0.0	0.98 ± 0.0	0.003 ± 0.0	0.119 ± 0.02	0.009 ± 0.0	0.982 ± 0.0	0.98 ± 0.0

Table 4: Extrinsic bias scores from Bias in Bios - ALBERT-large

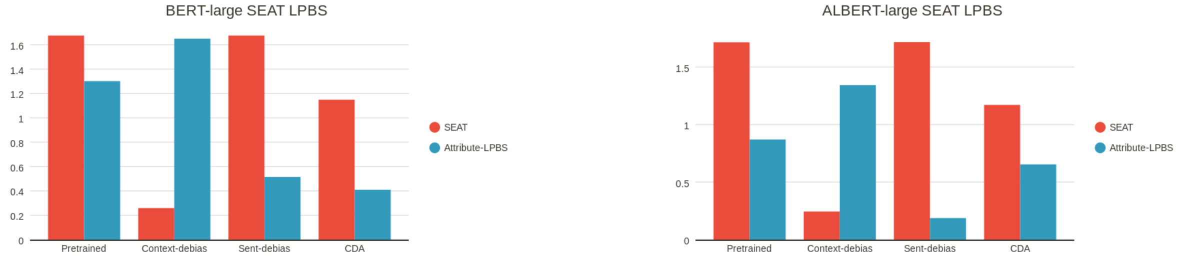
Model	**Data**	**TPRD**	**FPRD**	**ACC-F**	**ACC-M**	**CF**	**CF-TPRD**	**CF-FPRD**	**CF-ACC-F**	**CF-ACC-M**
Pretrained	Default	0.138 ± 0.02	0.013 ± 0.0	0.982 ± 0.0	0.979 ± 0.0	0.06 ± 0.0	0.118 ± 0.02	0.009 ± 0.0	0.979 ± 0.0	0.978 ± 0.0
Pretrained	Scrubbing	0.127 ± 0.01	0.01 ± 0.0	0.982 ± 0.0	0.978 ± 0.0	0.021 ± 0.0	0.12 ± 0.01	0.009 ± 0.0	0.981 ± 0.0	0.978 ± 0.0
Pretrained	Swapping	0.121 ± 0.02	0.009 ± 0.0	0.981 ± 0.0	0.979 ± 0.0	0.01 ± 0.0	0.121 ± 0.02	0.009 ± 0.0	0.981 ± 0.0	0.979 ± 0.0
Context-debias	Default	0.137 ± 0.02	0.012 ± 0.0	0.983 ± 0.0	0.98 ± 0.0	0.062 ± 0.0	0.114 ± 0.02	0.009 ± 0.0	0.981 ± 0.0	0.978 ± 0.0
Context-debias	Scrubbing	0.129 ± 0.02	0.01 ± 0.0	0.98 ± 0.0	0.978 ± 0.0	0.021 ± 0.0	0.124 ± 0.02	0.009 ± 0.0	0.979 ± 0.0	0.977 ± 0.0
Context-debias	Swapping	0.116 ± 0.02	0.009 ± 0.0	nan ± nan	nan ± nan	0.011 ± 0.0	0.115 ± 0.02	0.009 ± 0.0	0.981 ± 0.0	0.978 ± 0.0
CDA	Default	0.142 ± 0.02	0.013 ± 0.0	0.981 ± 0.0	0.978 ± 0.0	0.063 ± 0.0	0.121 ± 0.02	0.009 ± 0.0	0.979 ± 0.0	0.977 ± 0.0
CDA	Scrubbing	0.127 ± 0.02	0.01 ± 0.0	0.979 ± 0.0	0.977 ± 0.0	0.012 ± 0.0	0.125 ± 0.02	0.01 ± 0.0	0.979 ± 0.0	0.977 ± 0.0
CDA	Swapping	0.115 ± 0.01	0.009 ± 0.0	0.98 ± 0.0	0.977 ± 0.0	0.011 ± 0.0	0.116 ± 0.02	0.009 ± 0.0	0.98 ± 0.0	0.977 ± 0.0
Sent-debias	Original	0.145 ± 0.02	0.013 ± 0.0	0.981 ± 0.0	0.978 ± 0.0	0.007 ± 0.0	0.119 ± 0.03	0.009 ± 0.0	0.978 ± 0.0	0.976 ± 0.0
Sent-debias	Scrubbing	0.12 ± 0.02	0.01 ± 0.0	0.978 ± 0.0	0.976 ± 0.0	0.002 ± 0.0	0.108 ± 0.02	0.009 ± 0.0	0.978 ± 0.0	0.976 ± 0.0
Sent-debias	Swapping	0.117 ± 0.03	0.009 ± 0.0	0.981 ± 0.0	0.978 ± 0.0	0.002 ± 0.0	0.114 ± 0.02	0.009 ± 0.0	0.981 ± 0.0	0.978 ± 0.0

Table 5: Extrinsic bias scores from Jigsaw - BERT-large

Model	**Data**	**TPRD**	**FPRD**	**ACC-F**	**ACC-M**	**CF**	**CF-TPRD**	**CF-FPRD**	**CF-ACC-F**	**CF-ACC-M**
Pretrained	Default	0.044 ± 0.02	0.118 ± 0.04	0.896 ± 0.01	0.934 ± 0.0	0.116 ± 0.01	0.061 ± 0.01	0.029 ± 0.02	0.866 ± 0.01	0.902 ± 0.01
Pretrained	Scrubbing	0.026 ± 0.02	0.076 ± 0.03	0.891 ± 0.01	0.914 ± 0.01	0.031 ± 0.0	0.026 ± 0.01	0.058 ± 0.02	0.889 ± 0.01	0.904 ± 0.01
Pretrained	Swapping	0.027 ± 0.01	0.065 ± 0.02	0.892 ± 0.01	0.92 ± 0.01	0.01 ± 0.0	0.029 ± 0.01	0.066 ± 0.01	0.893 ± 0.01	0.921 ± 0.01
Context-debias	Default	0.047 ± 0.02	0.128 ± 0.04	0.895 ± 0.01	0.931 ± 0.01	0.116 ± 0.01	0.049 ± 0.02	0.032 ± 0.02	0.868 ± 0.01	0.894 ± 0.01
Context-debias	Scrubbing	0.014 ± 0.01	0.032 ± 0.02	0.701 ± 0.22	0.874 ± 0.06	0.015 ± 0.01	0.015 ± 0.01	0.026 ± 0.02	0.698 ± 0.21	0.87 ± 0.06
Context-debias	Swapping	0.031 ± 0.01	0.049 ± 0.02	0.889 ± 0.01	0.915 ± 0.01	0.01 ± 0.0	0.031 ± 0.01	0.05 ± 0.02	0.888 ± 0.01	0.915 ± 0.01
CDA	Default	0.055 ± 0.02	0.114 ± 0.03	0.902 ± 0.01	0.932 ± 0.0	0.115 ± 0.01	0.048 ± 0.01	0.024 ± 0.01	0.872 ± 0.01	0.899 ± 0.0
CDA	Scrubbing	0.023 ± 0.01	0.053 ± 0.03	0.894 ± 0.01	0.904 ± 0.01	0.018 ± 0.0	0.022 ± 0.01	0.041 ± 0.02	0.892 ± 0.01	0.899 ± 0.01
CDA	Swapping	0.026 ± 0.02	0.038 ± 0.02	0.891 ± 0.01	0.913 ± 0.01	0.008 ± 0.0	0.026 ± 0.02	0.036 ± 0.02	0.891 ± 0.01	0.913 ± 0.01
Sent-debias	Original	0.074 ± 0.02	0.108 ± 0.03	0.898 ± 0.01	0.926 ± 0.01	0.069 ± 0.01	0.041 ± 0.02	0.025 ± 0.01	0.865 ± 0.01	0.89 ± 0.01
Sent-debias	Scrubbing	0.033 ± 0.01	0.035 ± 0.01	0.889 ± 0.01	0.909 ± 0.01	0.012 ± 0.0	0.037 ± 0.01	0.023 ± 0.02	0.885 ± 0.01	0.901 ± 0.01
Sent-debias	Swapping	0.021 ± 0.02	0.045 ± 0.01	0.893 ± 0.01	0.916 ± 0.01	0.006 ± 0.0	0.021 ± 0.02	0.045 ± 0.01	0.893 ± 0.01	0.915 ± 0.01

Table 6: Extrinsic bias scores from Jigsaw - ALBERT-large

Model	**Data**	**TPRD**	**FPRD**	**ACC-F**	**ACC-M**	**CF**	**CF-TPRD**	**CF-FPRD**	**CF-ACC-F**	**CF-ACC-M**
Pretrained	Default	0.077 ± 0.04	0.115 ± 0.04	0.894 ± 0.01	0.916 ± 0.01	0.143 ± 0.01	0.035 ± 0.02	0.039 ± 0.02	0.862 ± 0.01	0.86 ± 0.02
Pretrained	Scrubbing	0.032 ± 0.02	0.025 ± 0.02	0.789 ± 0.14	0.819 ± 0.1	0.031 ± 0.01	0.035 ± 0.03	0.033 ± 0.02	0.783 ± 0.14	0.808 ± 0.1
Pretrained	Swapping	0.037 ± 0.02	0.029 ± 0.01	0.869 ± 0.02	0.895 ± 0.02	0.012 ± 0.0	0.038 ± 0.02	0.032 ± 0.02	0.869 ± 0.02	0.895 ± 0.02
Context-debias	Default	0.064 ± 0.02	0.122 ± 0.02	0.891 ± 0.01	0.92 ± 0.01	0.137 ± 0.01	0.051 ± 0.02	0.025 ± 0.01	0.857 ± 0.01	0.875 ± 0.01
Context-debias	Scrubbing	0.031 ± 0.01	0.05 ± 0.03	0.885 ± 0.01	0.891 ± 0.01	0.039 ± 0.01	0.024 ± 0.01	0.041 ± 0.02	0.878 ± 0.01	0.874 ± 0.01
Context-debias	Swapping	0.023 ± 0.01	0.048 ± 0.02	0.874 ± 0.01	0.904 ± 0.01	0.011 ± 0.0	0.025 ± 0.01	0.048 ± 0.02	0.875 ± 0.01	0.905 ± 0.01
CDA	Default	0.078 ± 0.02	0.115 ± 0.03	0.887 ± 0.01	0.913 ± 0.01	0.139 ± 0.01	0.043 ± 0.02	0.031 ± 0.01	0.855 ± 0.01	0.865 ± 0.01
CDA	Scrubbing	0.026 ± 0.01	0.052 ± 0.02	0.866 ± 0.02	0.865 ± 0.02	0.024 ± 0.0	0.021 ± 0.01	0.036 ± 0.02	0.867 ± 0.02	0.857 ± 0.02
CDA	Swapping	0.042 ± 0.02	0.038 ± 0.02	0.866 ± 0.01	0.9 ± 0.01	0.011 ± 0.0	0.041 ± 0.02	0.037 ± 0.02	0.867 ± 0.01	0.9 ± 0.01
Sent-debias	Original	0.102 ± 0.05	0.106 ± 0.05	0.765 ± 0.18	0.87 ± 0.05	0.064 ± 0.03	0.023 ± 0.02	0.03 ± 0.02	0.736 ± 0.17	0.824 ± 0.03
Sent-debias	Scrubbing	0.011 ± 0.01	0.021 ± 0.02	0.685 ± 0.21	0.831 ± 0.04	0.009 ± 0.01	0.011 ± 0.01	0.018 ± 0.02	0.683 ± 0.21	0.825 ± 0.03
Sent-debias	Swapping	0.031 ± 0.02	0.031 ± 0.02	0.821 ± 0.13	0.88 ± 0.04	0.006 ± 0.0	0.034 ± 0.02	0.031 ± 0.02	0.818 ± 0.13	0.88 ± 0.04



(a) BERT-large model

(b) ALBERT-large model

Figure 8: Intrinsic bias scores using SEAT and LPBS. Results show inconsistencies in measuring bias between SEAT and LPBS for various mitigation strategies — lower scores are desirable in both cases.

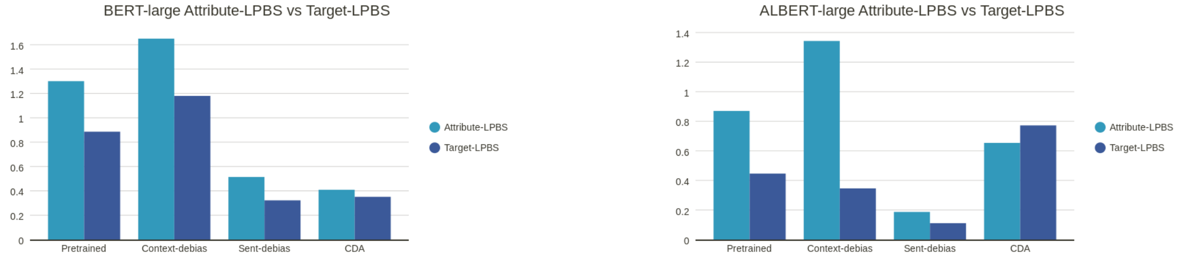


Figure 9: We compare LPBS in its default form (attribute-LPBS) and target-LPBS for both BERT and ALBERT. The plot shows a general positive correlation between the two metrics. In both cases, Sent-debias maintains the lowest bias score — lower scores are desirable in both cases

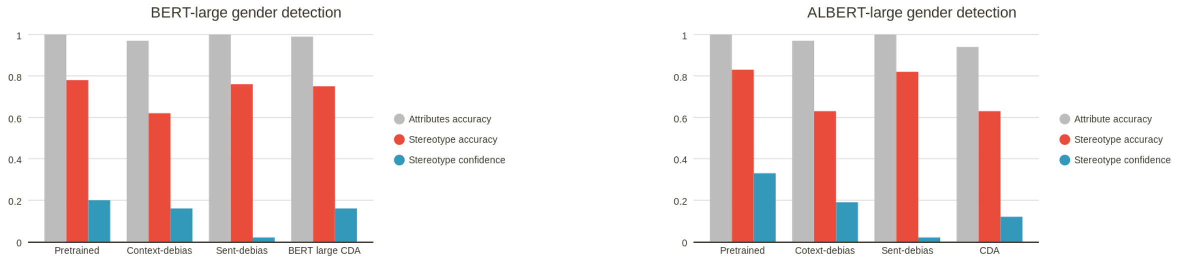


Figure 10: Gray bars indicate the accuracy of detecting gender information in attribute terms (high scores are desirable). Red bars indicate the accuracy of detecting gender information in stereotype terms (this should ideally be 0.5 in a fair classifier, showing the inability to correctly predict the gender association of stereotype terms). Blue bars indicate the average confidence of prediction: $\frac{1}{N}\sum_{i}^{N}|bias(e_i) - 0.5|$ (low scores are desirable).

Figure 11: How intrinsic bias mitigation and downstream data intervention interact to influence fairness on the bias-in-bios dataset.

Figure 12: How intrinsic mitigation and downstream data intervention interact to influence counterfactual fairness on the jigsaw data.

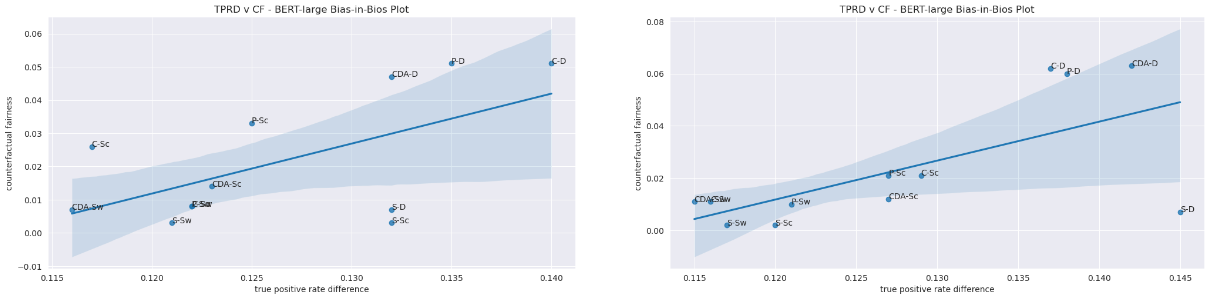


Figure 13: Relationship between TPRD and CF based on results from the Bias-in-Bios dataset.

Multimodal Event Transformer for Image-guided Story Ending Generation

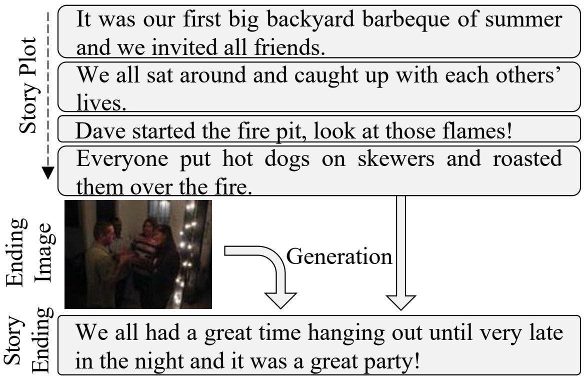
Yucheng Zhou, Guodong Long[*]
Australian AI Institute, School of Computer Science, FEIT, University of Technology Sydney
yucheng.zhou-1@student.uts.edu.au, guodong.long@uts.edu.au

Abstract

Image-guided story ending generation (IgSEG) is to generate a story ending based on given story plots and ending image. Existing methods focus on cross-modal feature fusion but overlook reasoning and mining implicit information from story plots and ending image. To tackle this drawback, we propose a multimodal event transformer, an event-based reasoning framework for IgSEG. Specifically, we construct visual and semantic event graphs from story plots and ending image, and leverage event-based reasoning to reason and mine implicit information in a single modality. Next, we connect visual and semantic event graphs and utilize cross-modal fusion to integrate different-modality features. In addition, we propose a multimodal injector to adaptive pass essential information to decoder. Besides, we present an incoherence detection to enhance the understanding context of a story plot and the robustness of graph modeling for our model. Experimental results show that our method achieves state-of-the-art performance for the image-guided story ending generation.

1 Introduction

Story ending generation (Guan et al., 2019) aims to generate a reasonable ending for a given story plot. It requires deep models to integrate powerful language understanding capability, which is crucial for artificial intelligence. Many efforts (Wang and Wan, 2019; Guan et al., 2019; Yao et al., 2019; Guan et al., 2020) have been proposed and achieved promising results since neural models designed for comprehending natural language allow them to understand story plots and reason reasonable story endings. With the advance of automatic story generation, it has attracted outstanding attention in multimodality research (Jung et al., 2020; Yu et al., 2021; Chen et al., 2021).



It was our first big backyard barbeque of summer and we invited all friends.

We all sat around and caught up with each others' lives.

Dave started the fire pit, look at those flames!

Everyone put hot dogs on skewers and roasted them over the fire.

We all had a great time hanging out until very late in the night and it was a great party!

Figure 1: Given a multi-sentence story plot and an ending image, the image-guided story ending generation aims to generate a story ending related to the image.

However, since story plots and story ending usually correspond to different content, the context with information bottleneck is not enough to deduce an informative story ending, i.e., generated endings tend to be inane and generic. To address this issue, Huang et al. (2021) propose an image-guided story ending generation (IgSEG) task that combines story plots and ending image to generate a coherent, specific and informative story ending. IgSEG demands not only introducing information from the ending image to story plots for story ending generation but also reasoning and mining implicit information from story plots and ending image, respectively. As shown in Figure 1, for story plots, "party" can be inferred from "big backyard barbeque" and "invited all friends", and "all friends", "all sat around" and "caught up with" can deduce "had a great time". For the ending image, "dim indoor" and "bright lights" can infer "very late in the night".

Existing methods (Huang et al., 2021; Xue et al., 2022) focus on cross-modal feature fusion but overlook reasoning and mining implicit information from story plots and ending images. Nonetheless, to effectively conduct cross-modal feature fusion, it is necessary to reason and mine more implicit

---

[*] Corresponding author.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3434–3444
May 2-6, 2023

information from single-modality data. An event is a fine-grained semantic unit, which refers to a text span composed of a predicate and its arguments (Zhang et al., 2020). Recently, event-centric reasoning displays excellent capability for context understanding and subsequent event prediction (Zhou et al., 2022b). In this work, we propose a multimodal event transformer (MET) to mine implicit information to improve cross-modal fusion. For story plots, we leverage semantic role labeling (SRL) parser (He et al., 2017) to extract events from story plots and then construct them into a semantic event graph. For an ending image, we utilize scene graph parser (Zellers et al., 2018) to capture visual concepts and their relation to construct visual event graphs. Since edges contain relationships between nodes in visual and semantic event graphs, we employ relational graph convolutional networks (RGCN) (Schlichtkrull et al., 2018) to encode event graphs to infer implicit information.

For cross-modal feature fusion, most recent works (Huang et al., 2021; Xue et al., 2022) adopt attention-based neural network models to implicitly integrate multi-modal features. However, due to the complexity of cross-modal features and the existence of dependency between single-modal features, it is often difficult for these models to complement cross-modal features. To tackle the issue, we propose cross-modal fusion to integrate different-modality features. Specifically, we merge visual and semantic event graphs and use RGCN to fuse cross-modal features for feature complement.

Moreover, since features from different modalities suffer from domain inconsistency, previous methods (Huang et al., 2021; Xue et al., 2022) directly concatenate them and pass them to the decoder, which is not a crafted manner. To appropriately combine features from different modalities, we design a multimodal injector to integrate relevant features into the decoder. In addition, we propose an incoherence detection to enhance the context understanding for a story plot and the robustness of graph modeling for our model.

In experiments, we conduct extensive evaluations on two datasets (i.e., VIST-E (Huang et al., 2021) and LSMDC-E (Xue et al., 2022)). Experimental results show that our method outperforms strong competitors and achieves state-of-the-art performance. In addition, we conduct further analysis and human evaluation to demonstrate the effectiveness of our method.

2 Related Work

2.1 Story Ending Generation

Story ending generation aims to generate a story ending for given story plots, and it is one of the important tasks in natural language generation. Many efforts have been invested in story ending generation (Wang and Wan, 2019; Guan et al., 2019; Yao et al., 2019; Guan et al., 2020). To make the generated story ending more reasonable, Guan et al. (2019) propose a model encapsulating a multi-source attention mechanism, which can utilize context clues and understand commonsense knowledge. To ensure the coherence in generated story endings, Wang and Wan (2019) propose a transformer-based conditional autoencoder, which can capture contextual clues in story splot. To improve long-range coherence in generated stories, Guan et al. (2020) pre-train model on external commonsense knowledge bases for the story ending generation. Zhou et al. (2022b) propose a correlation-aware context-to-event pre-trained transformer, which applies to a wide range of event-centric reasoning and generation scenarios, including story ending generation. Beyond the limit of single-modal information, Huang et al. (2021) introduce visual information to enrich the generation of story endings with more coherent, specific, and informative. To improve cross-modal feature fusion, Xue et al. (2022) propose a multimodal memory transformer, which fuses contextual and visual information to capture the multimodal dependency effectively.

2.2 Visual Storytelling

Visual storytelling task is proposed by Huang et al. (2016), which aims to generate a story based on a given image stream. Wang et al. (2018) present an adversarial reward learning framework to learn an implicit reward function from human demonstrations. To inject imaginary concepts that do not appear in the images, some works (Yang et al., 2019; Chen et al., 2021; Xu et al., 2021) propose building scene graphs and injecting external knowledge into model to reason the relationship between visual concepts. Qi et al. (2021) propose a latent memory-augmented graph transformer to exploit the semantic relationships among image regions and attentively aggregate critical visual features based on the parsed scene graphs.

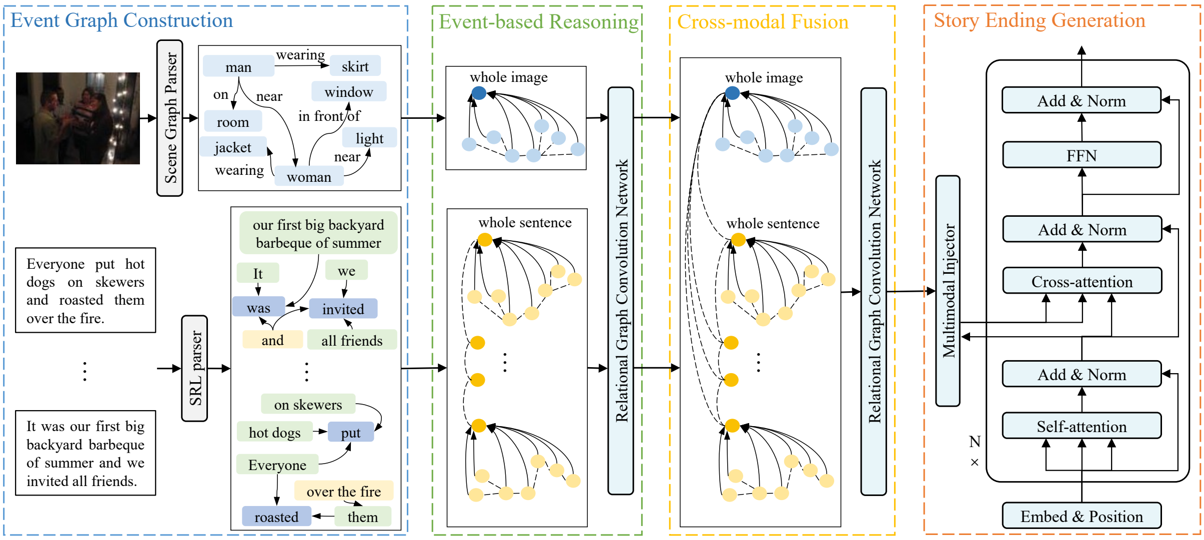


Figure 2: An overview of our model. Grey rounded rectangles denote fixed model. Blue rounded rectangles denote parameters that will be optimized.

2.3 Event-centric Reasoning

Events always play an essential role in a story because a story is composed of multiple events and implies the relationship between the events. An event is a text span composed of a predicate and its arguments (Zhang et al., 2020). Multiple events include relations between events that conform to human commonsense (Zhou et al., 2022a). Some works use plot events for story generation, which is generating a prompt and then transforming it into a text (Ammanabrolu et al., 2020; Fan et al., 2019). To generate a more coherent and specific ending, understanding events in story plots and their relationship can obtain informative context, which is a crucial step for story ending generation.

3 Method

This section will elaborate on our method for image-guided story ending generation, including event graph construction, event-based reasoning, cross-modal fusion, multimodal injector and story ending generation. The details of our method are shown in Figure 2. Lastly, details about objectives and training are elaborated.

3.1 Event Graph Construction

Semantic Event Graph. The story plot contains multiple events which are correlated with each other. The definition of an event is a text span composed of a predicate and its arguments (Zhang et al., 2020). The event-centric reasoning shows excellent capability for context understanding and subsequent event prediction (Zhou et al., 2022b). To effectively reason and mine more implicit information from story plots, we use semantic role labeling (SRL) to parse the story and extract events from parsing results, as shown in Figure 2. Specifically, Given story plots $\mathcal{S} = \{\boldsymbol{S}_1, \boldsymbol{S}_2, \boldsymbol{S}_3, \boldsymbol{S}_4\}$, we construct semantic event graphs $\mathcal{G}_i^s = (\mathcal{V}_i^s, \mathcal{E}_i^s)$ by SRL. $\mathcal{E}_i^s$ consists of two vectors, one for the positive direction and one for the opposite direction, and $\mathcal{V}_i^s = \{\boldsymbol{s}_0^i, \boldsymbol{s}_1^i, \boldsymbol{s}_2^i, \cdots, \boldsymbol{s}_n^i\}$. To obtain features of each node, we use a pre-trained transformer encoder to obtain token representations in sentence $\boldsymbol{S}_i$.

$$\boldsymbol{T}_i = \text{Trans-Enc}(\boldsymbol{S}_i), \boldsymbol{T}_i \in \{\boldsymbol{t}_i^1, \boldsymbol{t}_i^2, \cdots, \boldsymbol{t}_i^g\} \quad (1)$$

where $\boldsymbol{t}_i^g$ denotes token representation, and g is length of sentence $\boldsymbol{S}_i$. Next, we conduct a mean pooling operation for tokens presentations based on SRL parsing result $\hat{\boldsymbol{S}}_i$ to get presentation $\hat{\boldsymbol{s}}_j^i$ for each node. In addition, we take pooling for all token presentations of sentence $\boldsymbol{S}_i$ to obtain a presentation of sentence node $\hat{\boldsymbol{s}}_0^i$. Each node $\hat{\boldsymbol{s}}_j^i$ in sentence $\boldsymbol{S}_i$ is connected to the sentence node. To preserve the relationship between sequences, we connect sentence nodes in the order of the sequence.

Visual Event Graph. For ending images, previous works (Huang et al., 2021; Xue et al., 2022) use pre-trained convolutional neural networks (CNN) to extract feature maps directly. We construct visual event graphs to reason and mine more implicit information from ending images. Scene graphs have been used for many tasks to produce structured graph representations of visual scenes (Zellers et al., 2018). Inspired by the success of these tasks, we parse the ending image $\boldsymbol{I}$ to a scene graph via the scene graph parser. A scene graph can be denoted as a tuple $\mathcal{G}^I = \{\mathcal{V}^I, \mathcal{E}^I\}$, where $\mathcal{V}^I = \{\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$ is a set of k detected objects. $\boldsymbol{v}_0$ denotes a representation of the whole image, and other $\boldsymbol{v}_i$ is a region representation of detected object. $\mathcal{E}^I = \{e_1, e_2, \cdots, e_m\}$ is a set of directed edges and each edge e_i refers to a triplet $(\boldsymbol{v}_i, \boldsymbol{r}_{i,j}, \boldsymbol{v}_j)$, which includes two directional edges from $\boldsymbol{v}_i$ to $\boldsymbol{r}_{i,j}$ and from $\boldsymbol{r}_{i,j}$ to $\boldsymbol{v}_j$. Specifically, the construction of the scene graph can be divided into two parts: one is object detection, and the other is visual relation detection.

For object detection, we leverage a well-trained object detector, Faster-RCNN (Ren et al., 2017) with a ResNet-152 (He et al., 2016) backbone, to classify and encode objects in the ending image $\boldsymbol{I}$. The outputs of detector include a set of region representations $\mathcal{V}^I = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$ and object categories $\mathcal{O} = \{o_1, o_2, \cdots, o_k\}$. For visual relation detection, we leverage MOTIFS (Zellers et al., 2018) as our relation detector to classify the relationship between objects. We train the relation detector on Visual Genome dataset (Krishna et al., 2017). The output of relation detector is a set of relation $\mathcal{E}^I = \{e_1, e_2, \cdots, e_m\}$, where e_i refers to a triplet $(\boldsymbol{v}_i, \boldsymbol{r}_{i,j}, \boldsymbol{v}_j)$. Lastly, we obtain the scene graph $\mathcal{G}^I = \{\mathcal{V}^I, \mathcal{E}^I\}$ of ending image by combining the results of object detection and relationship detection.

3.2 Event-based Reasoning

We perform graph-structure reasoning over semantic and visual event graphs to effectively reason and mine more implicit information from story plots and ending images. Since event graphs have multiple relations between nodes (e.g., relations between visual objects, relations between predicates and arguments, etc.), we select relational graph convolutional networks (RGCN), which can pass different messages along different relations. Specifically, for each layer l in L-layer RGCN, the node representation $\boldsymbol{w}_i^l$ is updated as follows:

$$\boldsymbol{w}_i^{l+1} = \text{ReLU}\Big(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_r(i)} \frac{1}{|\mathcal{N}_r(i)|} \boldsymbol{W}_r \cdot \boldsymbol{w}_j^l\Big) \quad (2)$$

where $\mathcal{R}$ denote a set of all edges types, and $\mathcal{N}_r(i)$ is the neighborhood of node i under relation r. To reason and mine more implicit information in single-modality, we conduct event-based reasoning on semantic and visual event graphs, respectively.

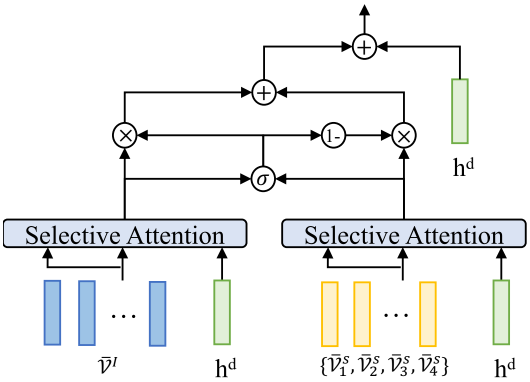


Figure 3: Details of the multimodal injector.

3.3 Cross-modal Fusion

We propose cross-modal fusion for visual and semantic event graphs to integrate information from story plots and ending images. We adopt a layer normalization for node features to reduce the cross-modal gap between visual and semantic graphs. For cross-modal feature fusion, previous works (Huang et al., 2021; Xue et al., 2022) adopt attention-based neural network models to implicitly integrate multimodal features. However, these models neglect the dependency between single-modal features. Therefore, we maintain graph structure for visual and semantic features and connect nodes that represent whole image and sentences, as shown in Figure 2. Moreover, we utilize RGCN as Eq.2 to integrate cross-modal features in event graph, and outputs denote as $\bar{\mathcal{V}}_i^s = \{\bar{\boldsymbol{s}}_0^i, \bar{\boldsymbol{s}}_1^i, \bar{\boldsymbol{s}}_2^i, \cdots, \bar{\boldsymbol{s}}_n^i\}$ and $\bar{\mathcal{V}}^I = \{\bar{\boldsymbol{v}}_0, \bar{\boldsymbol{v}}_1, \bar{\boldsymbol{v}}_2, \cdots, \bar{\boldsymbol{v}}_k\}$.

3.4 Multimodal Injector

To integrate different modal sources, we propose a multimodal injector, which adaptly extracts key information from different modal features and integrates them appropriately. As shown in Figure 3, inputs of multimodal injector include a hidden state $\boldsymbol{h}^d$ from the decoder, visual features $\bar{\mathcal{V}}^I$ and semantic features $\bar{\mathcal{V}}_i^s$. Specifically, we first use selective attention for key information extraction, i.e.,

$$\boldsymbol{h}_{attn}^u = \text{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right) V, u \in \{I, S\} \quad (3)$$

where Q is $\boldsymbol{h}^d$ from decoder; K and V are visual features $\bar{\mathcal{V}}^I$ or semantic features $\bar{\mathcal{V}}_i^s$; and d_k is the same as the dimension of $\boldsymbol{h}^d$. Then, the gate $\lambda \in [0, 1]$ and the fused output are defined as:

$$\lambda = \sigma\left(\boldsymbol{U}\boldsymbol{h}^I_{attn} + \boldsymbol{V}\boldsymbol{h}^S_{attn}\right) \tag{4}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ are trainable weights. λ controls how much visual information is attended.

$$\hat{\boldsymbol{h}}^d = \lambda \cdot \boldsymbol{h}^I_{attn} + (1-\lambda) \cdot \boldsymbol{h}^S_{attn} + \boldsymbol{h}^d \tag{5}$$

where the fusion vector $\hat{\boldsymbol{h}}^d$ is fed into the decoder.

3.5 Story Ending Generation

Recently, Transformer (Vaswani et al., 2017) shows its powerful ability to generate natural language (Radford et al., 2019). For story ending generation, we use a Transformer decoder as the decoder for our model. Specifically, the decoder input includes a segment of the generated story ending $\bar{\boldsymbol{C}}$ and fusion vector $\hat{\boldsymbol{h}}^d$ from the multimodal injector. The purpose of the decoder is to predict a probability distribution of the next word of the segment $\bar{\boldsymbol{C}}$, i.e.,

$$\begin{aligned} \boldsymbol{h}_i &= \text{Trans-Dec}(\hat{\boldsymbol{h}}^d, \bar{\boldsymbol{C}}) \in \mathbb{R}^d \\ &\quad \text{where } \bar{\boldsymbol{C}} = [c_1, \ldots, c_{i-1}] \qquad (6) \\ \boldsymbol{p}_i &= \text{LM-Head}(\boldsymbol{h}_i) \in \mathbb{R}^V \qquad (7) \end{aligned}$$

where $\boldsymbol{h}_i$ refers to the hidden representation in i-th step; V denotes token vocabulary and $\boldsymbol{p}_i$ refers to a probability distribution over $\mathcal{V}$; d in $\hat{\boldsymbol{h}}^d$ denotes the current number of layer. Lastly, the story ending generation objective is defined as a maximum likelihood estimation. The loss function is defined as:

$$\mathcal{L}^{(gen)} = -\frac{1}{|N|}\sum\nolimits_{i=1}^{N} \log \boldsymbol{p}_i(c_i), \tag{8}$$

where $\boldsymbol{p}_i(c_i)$ denotes fetching the probability of the i-th step gold token $c_i \in \boldsymbol{C}$ from $\boldsymbol{p}_i$. $\boldsymbol{C}$ refers to the gold caption, and N is its length.

3.6 Incoherence Detection

To enhance the understanding context of a story plot and robustness of graph modeling for our model, we introduce a training objective: incoherence detection. We set a 10% probability to replace a whole sentence node in semantic event graph randomly. In the objective, the final step output $\boldsymbol{h}_n$ of the decoder is passed into a MLP to classify whether each whole sentence node is changed, i.e.,

$$\boldsymbol{p}^{clf} = \sigma(\text{MLP}(\boldsymbol{h}_n)) \in \mathbb{R}^4 \tag{9}$$

where σ denotes a sigmoid function. The loss function is defined as:

$$\begin{aligned} \mathcal{L}^{(clf)} = -\frac{1}{4}\sum_{i=1}^{4} & y_i \cdot \log(\boldsymbol{p}_i^{clf}) \\ & + (1-y_i) \cdot \log(1-\boldsymbol{p}_i^{clf}) \qquad (10) \end{aligned}$$

3.7 Training

In model training, we set a trade-off parameter α for two losses $\mathcal{L}^{(gen)}$ and $\mathcal{L}^{(clf)}$. The total loss function of our model is definite as follows:

$$\mathcal{L} = \mathcal{L}^{(gen)} + \alpha \times \mathcal{L}^{(clf)} \tag{11}$$

4 Experiment

4.1 Dataset and Evaluation Metric

VIST-Ending. We compare our model and other state-of-the-art methods on the VIST-Ending (VIST-E) dataset (Huang et al., 2021). The dataset is built over VIST dataset (Huang et al., 2016). The VIST-E dataset comprises 39,920 samples for training, 4,963 samples for validation and 5,030 samples for testing. In experiments, we follow the data split in (Huang et al., 2021).

LSMDC-Ending. LSMDC-Ending (LSMDC-E) (Xue et al., 2022) contains 20,151 training samples, 1,477 validation samples and 2,005 test samples, which are collected from LSMDC 2021 (Rohrbach et al., 2017).

Visual Genome. We use the Visual Genome (VG) dataset to train a visual relationship detector. The dataset includes 108,077 images annotated with scene graphs, and we follow the setting in (Xu et al., 2017), which contains 150 object classes and 50 relation classes.

Evaluation Metric. As follow Xue et al. (2022), we utilize the same metrics to report evaluation results, and the evaluation code is open-source[1]. The evaluation metrics include: BLEU (Kingma and Ba, 2015), METEOR (Banerjee and Lavie, 2005), CIDEr (Vedantam et al., 2015), ROUGE-L (Lin, 2004) and Result Sum (rSUM) (Xue et al., 2022).

4.2 Implementation Details

For the scene graph, we limit the maximum number of objects to 10 and the maximum number of relationships to 20. The relational graph convolution network includes four relational graph convolution

---

[1] https://github.com/tylin/coco-caption

Method	**B@1**	**B@2**	**B@3**	**B@4**	**M**	**R-L**	**C**	**rSUM**
Seq2Seq (Luong et al., 2015)	13.96	5.57	2.94	1.69	4.54	16.84	12.04	57.58
Transformer (Vaswani et al., 2017)	17.18	6.29	3.07	2.01	6.91	18.23	12.75	66.44
IE+MSA (Guan et al., 2019)	19.15	5.74	2.73	1.63	6.59	20.62	15.56	72.02
T-CVAE (Wang and Wan, 2019)	14.34	5.06	2.01	1.13	4.23	15.51	11.49	53.77
MG+Trans (Huang et al., 2021)	19.43	7.47	3.92	2.46	7.63	19.62	14.42	74.95
MG+CIA (Huang et al., 2021)	20.91	7.46	3.88	2.35	7.29	21.12	19.88	82.89
MGCL (Huang et al., 2021)	22.57	8.16	4.23	2.49	7.84	21.66	21.46	88.41
MMT (Xue et al., 2022)	22.87	8.68	4.38	2.61	15.55	23.61	25.41	103.11
MET (Ours)	**24.31**	**8.79**	**4.62**	**2.73**	**16.41**	**24.49**	**26.47**	**107.82**

Table 1: Comparison results on VIST-E test set. B@n, M, R-L, C and rSUM denote BLEU@n, METEOR, ROUGE-L, CIDEr and Result Sum, respectively.

Method	**B@1**	**B@2**	**B@3**	**B@4**	**M**	**R-L**	**C**	**rSUM**
Seq2Seq (Luong et al., 2015)	14.21	4.56	1.70	0.70	11.01	19.69	8.69	60.56
Transformer (Vaswani et al., 2017)	15.35	4.49	1.82	0.76	11.43	19.16	9.32	62.33
MGCL (Huang et al., 2021)	15.89	4.76	1.57	0.00	11.61	20.30	9.16	63.29
MMT (Xue et al., 2022)	18.52	5.99	2.51	1.13	12.87	20.99	12.41	74.42
MET (Ours)	**19.98**	**6.48**	**2.89**	**1.77**	**14.53**	**22.73**	**13.85**	**82.23**

Table 2: Comparison results on LSMDC-E test set.

layers, and the size of input and output sets of 768. For semantic event reasoning, we use a pre-trained BERT model (Devlin et al., 2019) as the language model. The layers and attention heads of the decoder are 12 and 8. The dimension of embedding vectors in the decoder is 768, and the dimension of hidden states is 768. The visual feature encoder is ResNet-152. For model training, we select the Adam optimizer (Kingma and Ba, 2015) to optimize the model with learning rate of 2e-4. The maximum training epoch of our model is 25. The trade-off parameter α in Eq.11 is 0.2. The batch size, weight decay and warm-up proportion are 128, 0.01 and 0.1. During inference, we use the beam search with a beam size of 3 to generate a story ending with maximum sentence length is 25. Our model is trained on one V100 GPU.

4.3 Baselines

We compare our model with following state-of-the-art baselines: (1) **Seq2Seq** is a stack RNN-based model (Luong et al., 2015) with attention mechanisms, and image features are directly concatenated. (2) **Transformer**, proposed by Vaswani et al. (2017), is an encoder-decoder model with self-attention mechanisms. (3) **IE+MSA** is a story ending generation model incorporating external knowledge (Guan et al., 2019). (4) **T-CVAE** (Wang and Wan, 2019) is a conditional variational autoencoder based on transformer for missing story plots generation. (5) **MG+Trans** consists of multi-layer graph convolutional networks and a transformer decoder (Huang et al., 2021). (6) **MG+CIA** consists of multi-layer graph convolutional networks, top-down LSTM and one context-image attention unit in the decoder (Huang et al., 2021). (7) **MGCL** is an image-guided story ending generation model with multi-layer graph convolution networks and cascade-LSTM (Huang et al., 2021). (8) **MMT** is a multimodal memory transformer for image-guided story ending generation (Xue et al., 2022).

4.4 Main Results

The experimental results on VIST-E and LSMDC-E are shown in Table 1 and Table 2. From the tables, we can make two observations. Firstly, our model achieves state-of-the-art performance on the VIST-E and LSMDC-E datasets compared to other strong competitors. In addition, MG+CIA, MGCL, MMT and our model significantly and consistently outperform other models that directly concatenate visual features. It indicates that mining visual information is essential and can provide rich information to predict the ending. Moreover, our model

Method	B@1	B@2	B@3	B@4	M	R-L	C	rSUM
MET	**24.31**	**8.79**	**4.62**	**2.73**	**16.41**	**24.49**	**26.47**	**107.82**
w/o ID	23.84	8.70	4.51	2.56	15.91	24.10	25.86	105.48
w/o CMF	23.47	8.65	4.47	2.53	15.91	23.85	25.66	104.54
w/o MI	22.68	8.56	4.33	2.48	15.83	22.99	24.74	101.61
w/o VER	22.41	8.25	4.33	2.50	15.86	23.09	25.03	101.47
w/o SER	23.78	8.73	4.46	2.55	15.88	24.04	25.87	105.31
w/o CMF, MI	21.03	8.03	4.16	2.36	15.43	21.14	22.44	94.59

Table 3: Ablation study. "*w/o ID*" denotes removing the incoherence detection objective; "*w/o CMF*" denotes removing the cross-modal fusion; "*w/o MI*" denotes removing the multimodal injector; "*w/o VER*" denotes removing the event-based reasoning in visual event graph; "*w/o SER*" denotes removing the event-based reasoning in semantic event graph; "*w/o CMF, MI*" removing the cross-modal fusion and multimodal injector.

Method	B@1	B@2	B@4	M	R-L
Seq2Seq	14.27	4.27	1.05	6.02	16.32
Transformer	17.06	6.18	1.57	6.55	18.69
IE+MSA	20.11	6.62	1.68	6.87	21.27
T-CVAE	20.36	6.63	1.88	6.74	20.98
Plan&Write	20.92	5.88	1.44	7.10	20.17
KE-GPT2	**21.92**	**7.40**	1.90	7.41	20.58
MG+Trans	18.55	6.76	2.33	7.31	19.02
MGCL	20.27	6.26	1.81	6.91	21.01
MET	21.88	7.28	**2.36**	**7.41**	**21.32**

Table 4: Result of the SEG task on the VIST-E dataset (plain text). The bold / underline denotes the best and the second performance, respectively.

achieves better results than MG+CIA, MGCL and MMT, demonstrating that reasoning and mining implicit information from story plots and ending image is significant for image-guided story ending generation.

4.5 Ablation Study

To verify the effectiveness of our method, we conduct an ablation study and show the results in Table 3. Firstly, the table shows that removing each component or objective decreases the model performance, which demonstrates our method's effectiveness. In addition, we observe that removing cross-modal fusion and multimodal injector brings a great performance drop, which shows that cross-modal information mining and adaptive integration play a crucial role in story ending prediction.

4.6 SEG Setting

To investigate the effectiveness of visual information mining in our method, we remove the image from the VIST-E dataset and evaluate it on only plain text. The results are shown in Table 4. From the table, we observe that our model keeps competitive with Plan&Write (Yao et al., 2019) and KE-GPT2 (Guan et al., 2020) models designed especially for textual story generation. Moreover, our model outperforms MG+trans, which verifies the effectiveness of our incoherence detection and semantic event-based reasoning. Our model performs better when adding the image, as shown in Table 1. It demonstrates that mining implied visual information can help story ending generation.

4.7 Analysis

4.7.1 Impact of Event-based Reasoning

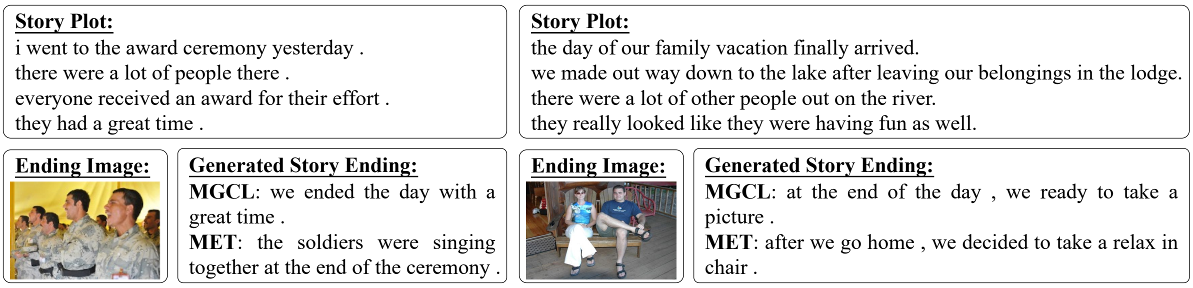
To investigate the effectiveness of event reasoning, we analyze its impact, and the results are shown in Table 5. From the table, we can observe that replacing semantic role labeling with dependency parsing leads to decreased performance. Moreover, replacing the visual event graph with whole image features (i.e., features extracted by pre-trained CNN) shows a performance drop. In addition, removing cross-modal fusion also shows a performance drop. These demonstrate the effectiveness of event-based reasoning for the image-guided story ending generation.

4.7.2 Case Study

To extensively evaluate our method, we conduct a case study for our model and MGCL, and some random sampling examples are shown in Figure 4. For example, in the left case, we can observe that our model can reason that the man in the image is a soldier, while the result from MGCL is not significantly related to visual content. For example, in the right case, our model can generate the word "relax"

Method	B@1	B@2	B@3	B@4	M	R-L	C	rSUM
MET	**24.31**	**8.79**	**4.62**	**2.73**	**16.41**	**24.49**	**26.47**	**107.82**
w/ Dependence Parser	23.47	8.70	4.50	2.57	15.88	24.15	24.06	103.33
w/o Visual Event Graph	22.13	8.19	4.21	2.44	15.76	22.88	23.93	99.54
w/o CMF	23.47	8.65	4.47	2.53	15.91	23.85	25.66	104.54

Table 5: Impact of event reasoning. "*w/ Dependency Parser*" denotes replacing semantic role labeling with dependency parsing; "*w/o Visual Event Graph*" denotes removing the visual event graph and provides the whole image features as inputs; "*w/o CMF*" denotes removing the cross-modal fusion.



Story Plot:
i went to the award ceremony yesterday .
there were a lot of people there .
everyone received an award for their effort .
they had a great time .

Ending Image:

Generated Story Ending:
MGCL: we ended the day with a great time .
MET: the soldiers were singing together at the end of the ceremony .

Story Plot:
the day of our family vacation finally arrived.
we made out way down to the lake after leaving our belongings in the lodge.
there were a lot of other people out on the river.
they really looked like they were having fun as well.

Ending Image:

Generated Story Ending:
MGCL: at the end of the day , we ready to take a picture .
MET: after we go home , we decided to take a relax in chair .

Figure 4: Random sampling examples generated by MET and MGCL.

the soldiers were singing together at the end of the ceremony .

after we go home , we decided to take a relax in chair .

Figure 5: Interpretable visualization analysis of our method (better viewed in color).

Method	Gram.	Logic.	Rele.
MET	**3.49**	**3.37**	**2.94**
MGCL	3.36	3.15	2.66
MG+Trans	3.22	2.78	2.71

Table 6: Human evaluation.

based on the objects "human" and "chair". It shows that our model can mine the implicit information based on visual and semantic information.

4.7.3 Interpretable Visualization Analysis

To investigate the effectiveness of the multimodal injector, we conduct an interpretable visualization analysis. The results are shown in Figure 5. The word with a blue underline denotes that the multimodal injector is assigned the greater probability in the node of visual event graph. Green corresponds to greater probability in the node of semantic event graph. The dotted boxes below represent the specific content of nodes. From the results, we can observe that nodes in visual and semantic event graphs are able to deduce implicit information.

4.7.4 Human Evaluation

To evaluate our method more comprehensively, we conducted a human evaluation to compare further the performance of our model and MGCL and MG+trans. As follow Huang et al. (2021), we considered three metrics for the story ending generated by models: Grammaticality (Gram.) (Wang and Wan, 2019) evaluates correctness, natural, and fluency of story endings; Logicality (Logic.) (Wang and Wan, 2019) evaluates reasonability and coherence of story endings; Relevance (Rele.) (Yang et al., 2019) measures how relevant between images and generated story endings. We randomly sampled 100 samples from the test set and display them to 3 recruited annotators. Thereby, each annotator worked on 300 items from 3 models. We show 3 annotators all outputs from all 3 models at once and shuffle the output-model correspondence to ensure that annotators do not know which model the output is predicted from. Following Yang et al.

(2019), we set a 5-grade marking system, where one is the worst grade, and five is the maximum. The results show that the performance of our model is significantly better than MGCL and MG+trans. That is, our model can generate higher-quality story endings.

5 Conclusion

In this work, we propose a multimodal event transformer, a framework for image-guided story ending generation. Our method includes event graph construction, event-based reasoning, cross-model fusion, multimodal injector and story ending generation. Different from previous work, our method not only focuses on cross-modal information fusion but also on reasoning and mining implicit information from single-modality data. In addition, we propose an incoherence detection to enhance the understanding context of a story plot and robustness of graph modeling for our model. In the experiments, results show that our method delivers state-of-the-art performance.

Limitations

Although our proposed method can effectively reason and mine implicit information from story plots and ending image, it suffers from weaknesses in integrating cross-modal information. Specifically, our method connects visual and semantic event graphs by connecting whole image nodes and whole sentence nodes. It lacks fine-grained information to pass between semantic events to visual objects. In further work, we will study how to pass fine-grained information between visual and semantic event graphs.

References

Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J. Martin, and Mark O. Riedl. 2020. Story realization: Expanding plot events into sentences. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7375–7382. AAAI Press.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

Hong Chen, Yifei Huang, Hiroya Takamura, and Hideki Nakayama. 2021. Commonsense knowledge aware concept selection for diverse and informative visual storytelling. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 999–1008. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2650–2660. Association for Computational Linguistics.

Jian Guan, Fei Huang, Minlie Huang, Zhihao Zhao, and Xiaoyan Zhu. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Trans. Assoc. Comput. Linguistics*, 8:93–108.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6473–6480. AAAI Press.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 473–483. Association for Computational Linguistics.

Qingbao Huang, Chuan Huang, Linzhang Mo, Jielong Wei, Yi Cai, Ho-fung Leung, and Qing Li. 2021.

Igseg: Image-guided story ending generation. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3114–3123. Association for Computational Linguistics.

Ting-Hao (Kenneth) Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross B. Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. 2016. Visual storytelling. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1233–1239. The Association for Computational Linguistics.

Yunjae Jung, Dahun Kim, Sanghyun Woo, Kyungsu Kim, Sungjin Kim, and In So Kweon. 2020. Hide-and-tell: Learning to bridge photo streams for visual storytelling. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 11213–11220. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Mengshi Qi, Jie Qin, Di Huang, Zhiqiang Shen, Yi Yang, and Jiebo Luo. 2021. Latent memory-augmented graph transformer for visual storytelling. In *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pages 4892–4901. ACM.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149.

Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Christopher Joseph Pal, Hugo Larochelle, Aaron C. Courville, and Bernt Schiele. 2017. Movie description. *Int. J. Comput. Vis.*, 123(1):94–120.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4566–4575. IEEE Computer Society.

Tianming Wang and Xiaojun Wan. 2019. T-CVAE: transformer-based conditioned variational autoencoder for story completion. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5233–5239. ijcai.org.

Xin Wang, Wenhu Chen, Yuan-Fang Wang, and William Yang Wang. 2018. No metrics are perfect: Adversarial reward learning for visual storytelling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 899–909. Association for Computational Linguistics.

Chunpu Xu, Min Yang, Chengming Li, Ying Shen, Xiang Ao, and Ruifeng Xu. 2021. Imagine, reason and write: Visual storytelling with graph knowledge and relational reasoning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 3022–3029. AAAI Press.

Danfei Xu, Yuke Zhu, Christopher B. Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *2017 IEEE Conference on Computer*

Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 3097–3106. IEEE Computer Society.

Dizhan Xue, Shengsheng Qian, Quan Fang, and Changsheng Xu. 2022. Mmt: Image-guided story ending generation with multimodal memory transformer. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 750–758, New York, NY, USA. Association for Computing Machinery.

Pengcheng Yang, Fuli Luo, Peng Chen, Lei Li, Zhiyi Yin, Xiaodong He, and Xu Sun. 2019. Knowledgeable storyteller: A commonsense-driven generative model for visual storytelling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5356–5362. ijcai.org.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7378–7385. AAAI Press.

Youngjae Yu, Jiwan Chung, Heeseung Yun, Jongseok Kim, and Gunhee Kim. 2021. Transitional adaptation of pretrained models for visual storytelling. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 12658–12668. Computer Vision Foundation / IEEE.

Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural motifs: Scene graph parsing with global context. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5831–5840. Computer Vision Foundation / IEEE Computer Society.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. ASER: A large-scale eventuality knowledge graph. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 201–211. ACM / IW3C2.

Yucheng Zhou, Xiubo Geng, Tao Shen, Guodong Long, and Daxin Jiang. 2022a. Eventbert: A pre-trained model for event correlation reasoning. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 850–859. ACM.

Yucheng Zhou, Tao Shen, Xiubo Geng, Guodong Long, and Daxin Jiang. 2022b. ClarET: Pre-training a correlation-aware context-to-event transformer for event-centric generation and classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2559–2575, Dublin, Ireland. Association for Computational Linguistics.

Improving Cross-modal Alignment for Text-Guided Image Inpainting

Yucheng Zhou, Guodong Long[*]
Australian AI Institute, School of Computer Science, FEIT, University of Technology Sydney
`yucheng.zhou-1@student.uts.edu.au, guodong.long@uts.edu.au`

Abstract

Text-guided image inpainting (TGII) aims to restore missing regions based on a given text in a damaged image. Existing methods are based on a strong vision encoder and a cross-modal fusion model to integrate cross-modal features. However, these methods allocate most of the computation to visual encoding, while light computation on modeling modality interactions. Moreover, they take cross-modal fusion for depth features, which ignores a fine-grained alignment between text and image. Recently, vision-language pre-trained models (VLPM), encapsulating rich cross-modal alignment knowledge, have advanced in most multimodal tasks. In this work, we propose a novel model for TGII by improving cross-modal alignment (CMA). CMA model consists of a VLPM as a vision-language encoder, an image generator and global-local discriminators. To explore cross-modal alignment knowledge for image restoration, we introduce cross-modal alignment distillation and in-sample distribution distillation. In addition, we employ adversarial training to enhance the model to fill the missing region in complicated structures effectively. Experiments are conducted on two popular vision-language datasets. Results show that our model achieves state-of-the-art performance compared with other strong competitors.

1 Introduction

Text-guided image inpainting (TGII), involving computer vision (CV) and natural language processing (NLP), aims to restore visual content for a missing area in a damaged image based on a given text (Zhang et al., 2020a). With the development of CV and NLP, it plays an essential role in many real-world applications, such as image editing (Zhu et al., 2020), damaged image restoration (Liu et al., 2019), and image rendering (Kirillov et al., 2020). Therefore, it has become one of the most crucial areas in CV and NLP.

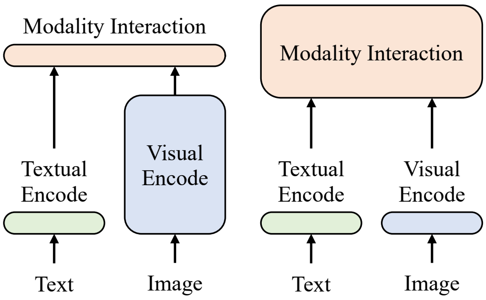


Figure 1: Different categories of vision-and-language models. *left*: most of the computation on visual encoding; *right*: most of the computation on modeling modality interactions.

Existing methods (Zhang et al., 2020b; Lin et al., 2020; Wu et al., 2021) adopt an encoder-decoder framework as their backbone with a vision-language fusion module to introduce textual information. These methods use separate encoders for images and texts, heavier on the former. Then, the vision-language fusion module is used to integrate the features from the two modalities through a simple similarity calculation of features, or shallow attention layers (Vaswani et al., 2017), as shown on the left of Figure 1. Most of the computation of these methods on visual encoding, while textual features only serve as a complement to deep visual features, which ignores the importance of deep interaction of multimodal information. Moreover, these methods share a common drawback: they do not perform well for natural image datasets with a wide variety of objects (e.g., MSCOCO (Lin et al., 2014)). The reason is that these methods lack fine-grained alignment knowledge of texts and images to guide the fusion of cross-modal information in multimodal interactions. Besides, their fusion modules lack powerful cross-modal reasoning capabilities.

Recently, providing the success of pre-trained language/vision Transformer (Devlin et al., 2019; Dosovitskiy et al., 2021; Zhou et al., 2022a),

---

[*] Corresponding author.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3445–3456
May 2-6, 2023

some works (e.g., ViLT (Kim et al., 2021) and SimVLM (Wang et al., 2021)) pre-train a vision-and-language Transformer on a large-scale image-text dataset. These Vision-and-Language Pre-trained Models (VLPM) achieve exciting performance on many multimodal downstream tasks. The reason is that VLPM encapsulates rich image and text alignment knowledge and has a strong cross-modal reasoning capability (Chen et al., 2020; Kim et al., 2021). To enhance the cross-modal interaction between images and texts, Kim et al. (2021) propose ViLT, which focuses most of the computation on modeling the multimodal interaction, as shown on the right side of Figure 1.

Motivated by Kim et al. (2021), we propose a novel model enhanced by cross-modal alignment (CMA) for text-guided image inpainting, comprising a vision-and-language encoder, an image generator and global-local discriminators. Different from previous works (Zhang et al., 2020b; Lin et al., 2020; Wu et al., 2021), we employ a vision-and-language encoder based on VLPM to encode images and texts instead of separate vision and language encoders. The vision-and-language encoder can encode images and texts in a cross-modal interaction manner to implement visual priors reconstruction. Then, the visual features integrating textual information (i.e., reconstructed visual priors) obtained by VLPM are passed to an image generator to generate a restored image. To improve cross-modal alignment for image inpainting, we introduce cross-modal alignment distillation to guide a fine-grained fusion of cross-modal knowledge. Moreover, to further strengthen the visual priors reconstruction, we utilize in-sample distillation to enhance the model's cross-modal reasoning capability for the content of missing regions. Besides, we employ adversarial learning to improve the quality of generated images through global-local discriminators.

Experiments are conducted on two popular image-text datasets with a wide variety of objects (i.e., MSCOCO (Lin et al., 2014) and Flicker30K (Plummer et al., 2017)). Experimental results demonstrate that our method achieves state-of-the-art performance compared to other strong competitors. In addition, we analyze the effectiveness of each module of our method and the impact of cross-modal alignment. Moreover, we also conduct extensive analysis to verify the effectiveness of our method.

2 Method

In this section, we will introduce our CMA model as shown in Figure 2. CMA model consists of a vision-and-language encoder, an image generator and global-local discriminators. In addition, details about training and inference are elaborated.

2.1 Vision-and-Language Encoding

In previous works (Zhang et al., 2020b; Lin et al., 2020; Wu et al., 2021), images and texts are encoded by separate encoders. Specifically, a CNN-based image encoder is used to extract visual features for images, while a RNN-based text encoder is used to encode text to obtain textual features. Next, the image and text features are integrated by a multimodal fusion module to obtain multimodal representations (i.e., reconstructed visual priors). In this work, we employ a novel Transformer-based cross-modal encoder for image and text encoding, a vision-and-language encoder instead of two separate encoders. For language encoding, we employ a word embedding matrix to embed text $\boldsymbol{T} \in \mathbb{R}^{L\times|V|}$ into $\bar{\boldsymbol{T}} \in \mathbb{R}^{L\times e}$, where L, $|V|$, and e denote the length of text, size of vocabulary and size of embedding, respectively. For visual encoding, an image $\boldsymbol{I} \in \mathbb{R}^{C\times H\times W}$ is sliced into patches and flattened to $\boldsymbol{V} \in \mathbb{R}^{N\times(P^2\times C)}$, where P is the size of patch and N equal to $(H \times W)/P^2$. Following Kim et al. (2021), $\boldsymbol{V}$ is embedded into $\bar{\boldsymbol{V}} \in \mathbb{R}^{N\times e}$:

$$\bar{\boldsymbol{V}} = \text{Linear-Projection}(\boldsymbol{V}) \tag{1}$$

where the details of linear projection can be found in (Kim et al., 2021). Next, images and texts are encoded in a cross-modal interaction manner:

$$[\hat{\boldsymbol{T}}; \hat{\boldsymbol{V}}] = \text{Trans-Enc}([\bar{\boldsymbol{T}}; \bar{\boldsymbol{V}}]) \tag{2}$$

where $[;]$ denote a concatenation operation, and outputs of the vision-and-language encoder can be represented as $\hat{\boldsymbol{T}} \in \mathbb{R}^{L\times e}$ for textual representations and $\hat{\boldsymbol{V}} \in \mathbb{R}^{N\times e}$ for reconstructed visual priors. Compared to CNN constrained by its inherent properties (e.g., spatial-invariant kernels), which are not conducive to understanding global features (Wan et al., 2021), Transformer-based encoders have a natural advantage in encoding global features across modalities.

2.2 Image Generation

The process of image generation includes two stages. The first stage is to downsample the visual priors to extract deep visual representations.

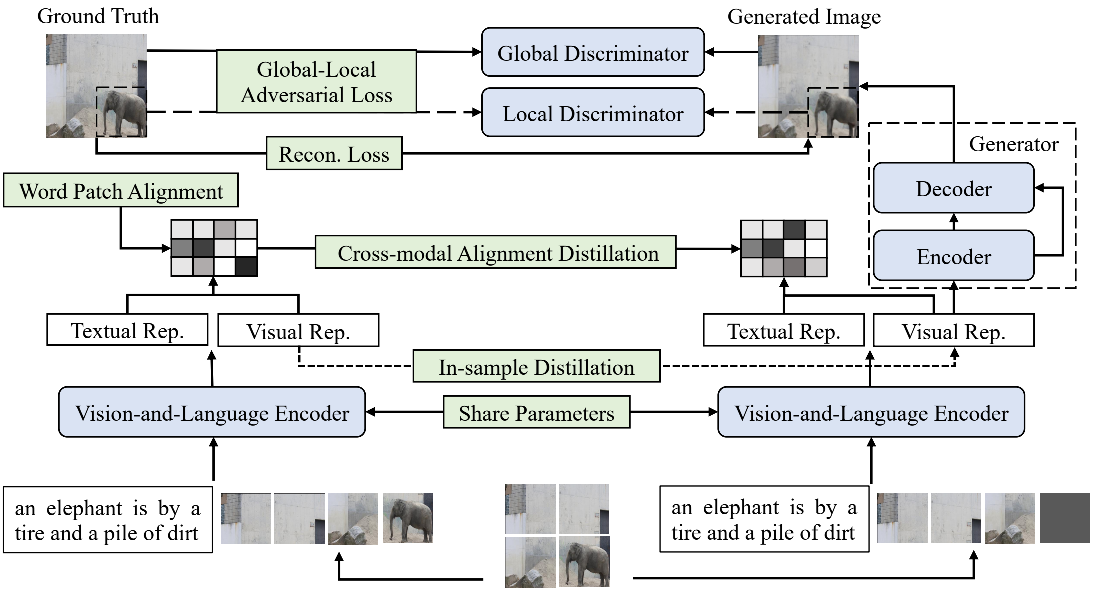


Figure 2: Overview of our model. Blue rounded rectangles denote trainable modules. Green right rectangles indicate training objectives or operations.

Next, performing upsampling for deep visual representations to generate a restored image. Different from the previous works (Zhang et al., 2020b; Lin et al., 2020; Wu et al., 2021), we do not integrate a fusion module to introduce textual features in the image generation process because the texts are introduced into the vision-and-language encoders. Although transformers demonstrate their effectiveness in long-term relations and the advantage of understanding global features, their computational complexity is quadratic with the input length, which hinders image generation (Wan et al., 2021). Therefore, our image generator consists of two CNN-based components, an encoder for downsampling and a decoder for upsampling. Similar to (Zhang et al., 2020a), we employ a 5-layer ResNet (He et al., 2016) as the downsampling encoder, and the visual priors are fed into it to obtain deep visual representations:

$$\boldsymbol{v} = \text{Down-Sampling}(\hat{\boldsymbol{V}}) \tag{3}$$

In addition, we pass $\boldsymbol{v}$ into the upsampling decoder to perform image reconstruction to generate a restored image, and the upsampling decoder consists of a 5-layer residual generator network.

$$\boldsymbol{I}_r = \text{Up-Sampling}(\boldsymbol{v}, \hat{\boldsymbol{V}}) \tag{4}$$

The visual priors obtained by the vision-and-language encoder are input to the upsampling decoder through a skip connection to provide detailed information that forgets in the downsampling stage.

2.3 Training

For model training, we construct five training objectives, including cross-modal alignment distillation, in-sample distillation, word patch alignment, reconstruction loss and global-local adversarial loss.

Cross-Modal Alignment Distillation. To guide image inpainting through cross-modal alignment knowledge, we pass the original image and text to the vision-and-language encoder to obtain a text representation $\hat{\boldsymbol{T}}_o$ and a visual priors $\hat{\boldsymbol{V}}_o$. Then, we obtain the correlation map $\boldsymbol{M}_o$ between each token of text and each image patch by calculating the similarity between them. The correlation map corresponding to the corrupted image and text is denoted as $\boldsymbol{M}_r$. Next, we compute the pair-wise similarity distillation loss between the two correlation maps:

$$\ell_{\text{CMAD}} = \frac{1}{N \times L}\sum_{i=1}^{L}\sum_{j=1}^{N}(a_{ij}^r - a_{ij}^o)^2 \tag{5}$$

$$a_{ij}^r = \frac{\boldsymbol{t}_i^{r\top}\boldsymbol{v}_j^r}{\|\boldsymbol{t}_i^r\|_2\|\boldsymbol{v}_j^r\|_2}, a_{ij}^r \in \boldsymbol{M}_r \tag{6}$$

$$a_{ij}^o = \frac{\boldsymbol{t}_i^{o\top}\boldsymbol{v}_j^o}{\|\boldsymbol{t}_i^o\|_2\|\boldsymbol{v}_j^o\|_2}, a_{ij}^o \in \boldsymbol{M}_o \tag{7}$$

where $\boldsymbol{t}_i^r \in \hat{\boldsymbol{T}}$, $\boldsymbol{v}_j^r \in \hat{\boldsymbol{V}}$, $\boldsymbol{t}_i^o \in \hat{\boldsymbol{T}}_o$ and $\boldsymbol{v}_j^o \in \hat{\boldsymbol{V}}_o$, respectively.

In-Sample Distillation. Besides using cross-modal alignment knowledge to guide image inpainting, we propose in-sample distillation. The purpose of this objective is to guide the damaged image to infer visual priors closer to that from the original image based on the text and known regions, i.e.,

$$\ell_{\text{ISD}} = \frac{1}{N} \sum_{i=1}^{N} \text{KL}\left(\boldsymbol{v}_i^o \| \boldsymbol{v}_i^r\right). \tag{8}$$

where KL denotes Kullback–Leibler divergence.

Word Patch Alignment. To maintain that the cross-modal alignment knowledge of the model is not degraded, we use the word patch alignment objective to preserve the cross-modal alignment knowledge integrated into the model.

$$\ell_{\text{WPA}} = \min_{\mathbf{T} \in \Pi(\mathbf{a},\mathbf{b})} \sum_{i=1}^{N} \sum_{j=1}^{L} \mathbf{T}_{ij} \cdot c\left(\boldsymbol{v}_i^o, \boldsymbol{t}_j^o\right) \tag{9}$$

$$c(\boldsymbol{v}_i^o, \boldsymbol{t}_j^o) = 1 - \frac{\boldsymbol{v}_i^{o\top} \boldsymbol{t}_j^o}{\|\boldsymbol{v}_i^o\|_2 \|\boldsymbol{t}_j^o\|_2} \tag{10}$$

$$\Pi(\mathbf{a},\mathbf{b}) = \{\mathbf{T} \in \mathbb{R}^{N \times L} \mid \mathbf{T}\mathbf{1}_m = \mathbf{a}, \mathbf{T}^\top \mathbf{1}_n = \mathbf{b}\} \tag{11}$$

where $\mathbf{1}_n$ denotes an n-dimensional all-one vector; $\mathbf{T}$ is a transport plan. The details can be found in (Kim et al., 2021).

Reconstruction Loss. We adopt the ℓ_1-norm error as the loss function between the restored image and its corresponding ground-truth image $\boldsymbol{I}_{gt}$:

$$\ell_1 = ||\boldsymbol{I}_r - \boldsymbol{I}_{gt}||_1, \tag{12}$$

Global Adversarial Loss. In image generation, the adversarial loss (Goodfellow et al., 2014) effectively improves the quality of the generated image. However, adversarial training is unstable since keeping the balance between the generator and discriminator is difficult. To tackle this problem, Arjovsky et al. (2017) propose the Wasserstein generative adversarial nets (WGAN) that can improve the stability of adversarial learning. We employ a WGAN hinge loss as the adversarial loss, and it can be formulated as follows:

$$\ell_{\text{G-adv,G}} = \mathbb{E}_{\hat{\boldsymbol{x}} \sim P_{data}(\boldsymbol{I}_r)}[-D(\hat{\boldsymbol{x}})] \tag{13}$$

$$\begin{aligned} \ell_{\text{G-adv,D}} =& \mathbb{E}_{\boldsymbol{x} \sim P_{data}(\boldsymbol{I}_{gt})}[\text{ReLU}(1 - D(\boldsymbol{x}))] \\ &+ \mathbb{E}_{\hat{\boldsymbol{x}} \sim P_{data}(\boldsymbol{I}_r)}[\text{ReLU}(1 + D(\hat{\boldsymbol{x}}))] \end{aligned} \tag{14}$$

where $D(\cdot)$ denotes a global discriminator; ReLU is the rectified linear unit function; $\mathcal{L}_{adv,G}$ and $\mathcal{L}_{adv,D}$ are loss function for inpainting model and discriminator, respectively.

The discriminator $D(\cdot)$ consists of 5-layer ResNet, followed by a fully connected layer. Each convolutional layer in ResNet applies spectral normalization to satisfy the Lipschitz constraints of Wasserstein GANs.

Local Adversarial Loss. For the local discriminator, we use the same settings as the global discriminator, and the loss is defined as:

$$\ell_{\text{L-adv,G}} = \mathbb{E}_{\hat{\boldsymbol{x}}_l \sim P_{data}(\boldsymbol{I}_r)}[-D_l(\hat{\boldsymbol{x}}_l)] \tag{15}$$

$$\begin{aligned} \ell_{\text{L-adv,D}} =& \mathbb{E}_{\boldsymbol{x}_l \sim P_{data}(\boldsymbol{I}_{gt})}[\text{ReLU}(1 - D_l(\boldsymbol{x}_l))] \\ &+ \mathbb{E}_{\hat{\boldsymbol{x}}_l \sim P_{data}(\boldsymbol{I}_r)}[\text{ReLU}(1 + D_l(\hat{\boldsymbol{x}}_l))] \end{aligned} \tag{16}$$

where x_l denotes the restored image or ground truth corresponding to the missing region; $D_l(\cdot)$ denotes a local discriminator.

Considering the loss functions above, the objective function of our model in the generation stage can be defined as:

$$\begin{aligned} \ell_G =& \lambda \ell_{\text{CMAD}} + \lambda \ell_{\text{ISD}} + \alpha \ell_{\text{WPA}} + \beta \ell_1 \\ &+ \gamma \ell_{\text{G-adv,G}} + \gamma \ell_{\text{L-adv,G}} \end{aligned} \tag{17}$$

where the λ, α, β and γ are the hyper-parameters used to balance the objective function.

In addition, the objective function of our model in the discriminative stage can be defined as:

$$\ell_D = \gamma \ell_{\text{G-adv,D}} + \gamma \ell_{\text{L-adv,D}} \tag{18}$$

2.4 Inference

During inference, we first pass image $\boldsymbol{I}$ and text $\boldsymbol{T}$ into the cross-modal encoder (i.e., Equ.1 and Equ.2) to obtain the visual representation $\hat{\boldsymbol{V}}$. Next, we deliver the visual representation $\hat{\boldsymbol{V}}$ into the generator (i.e., Equ.3 and Equ.4) to generate a restored image $\boldsymbol{I}_r$.

3 Experiments

3.1 Dataset and Evaluation Metrics

We conduct the experiment on two datasets: MSCOCO (Lin et al., 2014) and Flicker30K (Plummer et al., 2017). For the MSCOCO and Flicker30K datasets, we split them following their original training, validation and test set. Following Zhang et al. (2020a), we also set the mask for an image in two types: center mask and object mask. A center mask refers to a square mask taking 50%

Method	ℓ_1 **(%)** ↓	**FID** ↓	**KID** ↓	**TV loss (%)** ↓	**PSNR** ↑	**SSIM (%)** ↑
CSA (Liu et al., 2019)	5.14	50.88	2.85	4.32	19.87	82.53
PICNet (Zheng et al., 2019)	5.63	53.57	3.19	4.55	19.58	81.87
CTSDG (Guo et al., 2021)	5.03	48.95	2.56	4.31	19.98	82.69
MMFL (Lin et al., 2020)	4.36	44.33	2.22	4.28	20.73	82.92
TGII (Zhang et al., 2020b)	4.22	44.02	1.87	4.39	20.87	83.07
TDANet (Zhang et al., 2020a)	4.13	42.38	1.67	4.59	20.91	83.34
ALMR (Wu et al., 2021)	4.17	43.43	1.79	4.27	20.81	83.15
CMA (ours)	**3.78**	**39.52**	**1.34**	**4.17**	**22.07**	**85.18**

Table 1: Results on MSCOCO with the center mask.

Method	ℓ_1 **(%)** ↓	**FID** ↓	**KID** ↓	**TV loss (%)** ↓	**PSNR** ↑	**SSIM (%)** ↑
CSA (Liu et al., 2019)	8.79	53.49	3.65	4.85	18.99	75.50
PICNet (Zheng et al., 2019)	9.15	56.80	3.72	5.09	18.78	74.98
CTSDG (Guo et al., 2021)	8.69	51.64	3.36	4.82	19.13	75.84
MMFL (Lin et al., 2020)	7.59	47.15	2.91	4.53	20.07	76.16
TGII (Zhang et al., 2020b)	7.53	46.73	2.82	4.59	20.27	76.46
TDANet (Zhang et al., 2020a)	7.48	45.30	2.45	4.70	20.55	76.93
ALMR (Wu et al., 2021)	7.54	46.01	2.61	4.46	20.35	76.50
CMA (ours)	**7.00**	**42.23**	**2.01**	**4.30**	**21.75**	**78.67**

Table 2: Results on MSCOCO with the object mask.

Method	ℓ_1 **(%)** ↓	**FID** ↓	**KID** ↓	**TV loss (%)** ↓	**PSNR** ↑	**SSIM (%)** ↑
CSA (Liu et al., 2019)	4.99	50.76	2.78	4.14	19.93	83.97
PICNet (Zheng et al., 2019)	5.29	52.25	3.14	4.39	19.70	82.21
CTSDG (Guo et al., 2021)	4.78	48.66	2.54	4.10	20.39	84.21
MMFL (Lin et al., 2020)	4.13	43.92	2.12	4.26	20.78	83.96
TGII (Zhang et al., 2020b)	4.11	43.34	1.73	4.29	21.48	84.49
TDANet (Zhang et al., 2020a)	3.92	41.46	1.54	4.16	21.36	84.17
ALMR (Wu et al., 2021)	4.05	42.43	1.60	4.12	21.22	83.42
CMA (ours)	**3.61**	**38.30**	**1.29**	**4.00**	**22.55**	**86.33**

Table 3: Results on Flickr30K with the center mask.

area in the center of an image. An object mask indicates masking an image based on the object boxes provided by every image. To evaluate the performance of our model and other methods on these datasets, we utilize the ℓ_1 loss, Frechet Inception Distance (FID) (Heusel et al., 2017), Kernel Inception Distance (KID) (Heusel et al., 2017), total variation (TV) (Rudin et al., 1992), peak signal-to-noise ratio (PSNR) (Fardo et al., 2016) and structural similarity index (SSIM) (Wang et al., 2004) as metrics to report the results. ℓ_1 measure the ℓ_1 distance between restored and original images. FID and KID measure the quality of restored images based on human perception. PSNR and SSIM measure structural similarity between restored and original images and the ℓ_2 distance.

3.2 Experimental Settings

For training images in MSCOCO and Flicker30K, we resize them to make their minimal height/width 256 and crop them based on size 256 × 256 at the center. During training, we set the λ as 2, α as 1, β as 1 and γ as 0.1 in the objective function, and the model is trained by an Adam optimizer (Kingma and Ba, 2015) with the learning rate of 1×10^{-4}. In the vision-and-language encoder, the patch size, intermediate size and hidden size are 32, 3072 and 768. For a masked patch, we utilize a special token `[Vmask]` as input. We employ ViLT (Kim et al., 2021) to initialize our vision-and-language encoder. The weight decay and gradient clipping are set to 0.01 and 1.0. The maximum training epoch and batch size are 200 and 128. Warmup steps and maximum sequence length are set to 2000 and 40. Our experiments are conducted on 2 × V100 GPUs. We choose models with the best result on the validation set and report the results on the test set based on the models.

3.3 Main Results

The experimental results of our method and previous works on MSCOCO and Flicker30K are shown

Method	ℓ_1 **(%) ↓**	**FID ↓**	**KID ↓**	**TV loss (%) ↓**	**PSNR ↑**	**SSIM (%) ↑**
CSA (Liu et al., 2019)	8.60	53.18	3.57	4.75	19.21	75.82
PICNet (Zheng et al., 2019)	9.01	56.59	3.64	4.99	19.16	75.95
CTSDG (Guo et al., 2021)	8.45	51.35	3.30	4.69	19.22	76.73
MMFL (Lin et al., 2020)	7.58	46.53	2.86	4.53	20.34	76.42
TGII (Zhang et al., 2020b)	7.39	46.52	2.73	4.56	20.82	76.82
TDANet (Zhang et al., 2020a)	7.37	44.72	2.42	4.55	21.04	77.43
ALMR (Wu et al., 2021)	7.40	45.74	2.52	4.36	20.56	76.57
CMA (ours)	**6.86**	**41.28**	**1.91**	**4.25**	**22.07**	**79.93**

Table 4: Results on Flickr30K with the object mask.

Method	ℓ_1 **(%) ↓**	**FID ↓**	**KID ↓**	**TV loss (%) ↓**	**PSNR ↑**	**SSIM (%) ↑**
CSA (Liu et al., 2019)	4.04	56.88	2.74	3.70	20.03	81.12
PICNet (Zheng et al., 2019)	3.78	47.33	2.46	3.74	20.16	82.04
CTSDG (Guo et al., 2021)	3.63	38.05	1.98	3.71	20.73	82.64
MMFL (Lin et al., 2020)	3.42	29.79	1.39	3.59	20.68	82.36
TGII (Zhang et al., 2020b)	3.54	32.57	1.51	3.63	20.55	81.89
TDANet (Zhang et al., 2020a)	3.57	30.82	1.49	3.61	20.79	82.68
ALMR (Wu et al., 2021)	3.32	15.78	0.52	3.52	20.66	80.61
CMA (ours)	**3.07**	**14.21**	**0.41**	**3.51**	**20.84**	**82.68**

Table 5: Results on CUB with the center mask.

Method	ℓ_1 **(%) ↓**	**FID ↓**	**KID ↓**	**TV loss (%) ↓**	**PSNR ↑**	**SSIM (%) ↑**
CSA (Liu et al., 2019)	6.76	59.91	3.00	4.21	19.03	70.97
PICNet (Zheng et al., 2019)	6.42	50.16	2.82	4.32	18.96	70.27
CTSDG (Guo et al., 2021)	5.83	40.19	2.23	4.13	19.38	72.16
MMFL (Lin et al., 2020)	4.63	32.00	2.13	3.79	20.32	78.24
TGII (Zhang et al., 2020b)	4.72	34.39	2.07	3.93	20.10	78.65
TDANet (Zhang et al., 2020a)	4.71	33.10	2.03	3.72	20.40	77.77
ALMR (Wu et al., 2021)	4.51	26.33	1.66	3.71	20.25	75.69
CMA (ours)	**4.24**	**18.96**	**1.49**	**3.64**	**20.48**	**78.75**

Table 6: Results on CUB with the object mask.

in Table 1, Table 2, Table 3 and Table 4. From the tables, we can see that our model achieves state-of-the-art results compared with other strong competitors. In addition, we can make two observations: Firstly, using text to guide image inpainting can significantly improve the performance. For example, the text-guided image inpainting methods (i.e., MMFL, TGII, TDANet, ALMR and CMA) outperform standard image inpainting methods (i.e., CSA, PICNet and CTSDG). Next, for the gap between results on center masks and object masks, it shows that recovering completely removed objects is more difficult. Besides, we show results on the CUB (Wah et al., 2011) dataset without a wide variety of objects in Tabel 5 and Tabel 6. Results show our method outperforms other methods.

Method	**FID ↓**	**KID ↓**	**PSNR ↑**	**SSIM (%) ↑**
CMA	39.52	1.34	22.07	85.18
w/o G-Adv.	43.72	2.17	21.39	84.12
w/o L-Adv.	42.59	1.84	21.74	84.75
w/o Adv.	50.94	2.91	19.68	82.49
w/o Recon.	47.57	2.48	20.49	83.56
w/o CMAD	53.39	3.12	19.55	81.94
w/o ISD	52.84	3.06	19.67	82.23
w/o WPA	52.33	3.10	19.63	82.58

Table 7: Ablation study. ‘G-Adv.’, ‘L-Adv.’ and ‘Adv.’ denote global, local and both adversarial losses, respectively. ‘Recon.’, ‘CMAD’, ‘ISD’ and ‘WPA’ indicate reconstruction loss, cross-modal alignment distillation, in-sample distillation objectives and word patch alignment, respectively.

3.4 Ablation Study

We conduct an ablation study to investigate the effectiveness of each component of our approach and the results are reported in Table 7. We first investigate the impact of the adversarial learning by removing global adversarial loss, local adversarial loss and both adversarial losses and find that the performance drops. The reason is that adversarial objectives focus on high-level features, which can

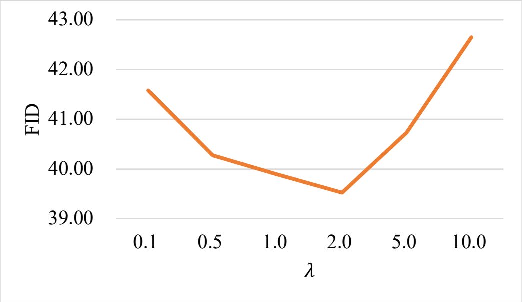


Figure 3: Performance of our model with different trade-off parameters λ.

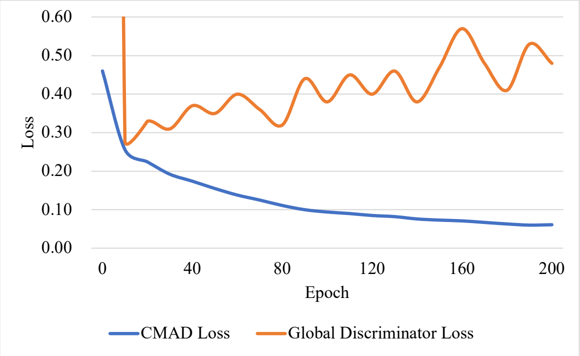


Figure 4: The adversarial loss and cross-modal alignment distillation loss on the training set w.r.t different epochs in the training phase.

effectively fill the missing region on complicated structures. Next, we test our method without reconstruction loss, which also decreases scores on all metrics. It demonstrates that image reconstruction plays an essential role in model training. Finally, we compare our method with the baseline without cross-modal alignment distillation and in-sample distillation, and the performance drops significantly. It indicates that the CMAD and ISD objectives can enhance the cross-modal alignment for text-guided image inpainting.

3.5 Analysis

3.5.1 Impact of Cross-Modal Alignment

To assess the impact of cross-modal alignment for our method, we set different trade-off parameters λ. Specifically, during model training, we set a different λ for the loss function in Eq.17. The performance of our method in different λ is reported in Figure 3. It is observed that with the λ increased, FID first drops and then rises, indicating that our method's performance first rises and then drops. The results demonstrate that the cross-modal alignment objective is crucial for our method, but overemphasizing the objective may hinder model learning.

Method	Naturalness	Semantic Consistency
CMA	1.44	1.46
TDANet	2.39	2.52
ALMR	2.77	2.64
CSA	3.40	3.38

Table 8: Numerical ranking score of user study. The lower the score, the better the performance.

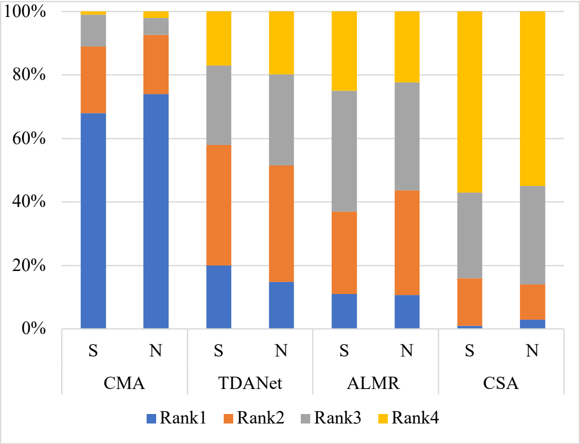


Figure 5: Ranking score distribution of the user study. "S" means semantic consistency score, "N" means naturalness.

3.5.2 Deep Dive into Cross-Modal Alignment

We take a deep dive into the impact of cross-modal alignment. In our method, adversarial learning involves a global discriminator to distinguish whether the generated image is original or generated, and the cross-modal alignment objective is to guide visual priors reconstruction. Their loss values, i.e., $\ell_{\mathrm{G-adv,D}}$ and ℓ_{CMAD}, are plotted in Figure 4. We can see that the loss of cross-modal alignment objective quickly drops and then slowly drops, indicating that the model gets good performance on visual priors reconstruction. Meanwhile, we can observe that the adversarial loss of the discriminator quickly drops and then slowly goes up, demonstrating that we can see that the classification loss of the discriminator gets good performance and then is fooled later by the image generated from the image generator. Therefore, it shows that the cross-modal alignment objective supports the adversarial learning very well.

3.5.3 User Study

Following Zhang et al. (2020a), we take a user study to quantify the qualitative comparison from

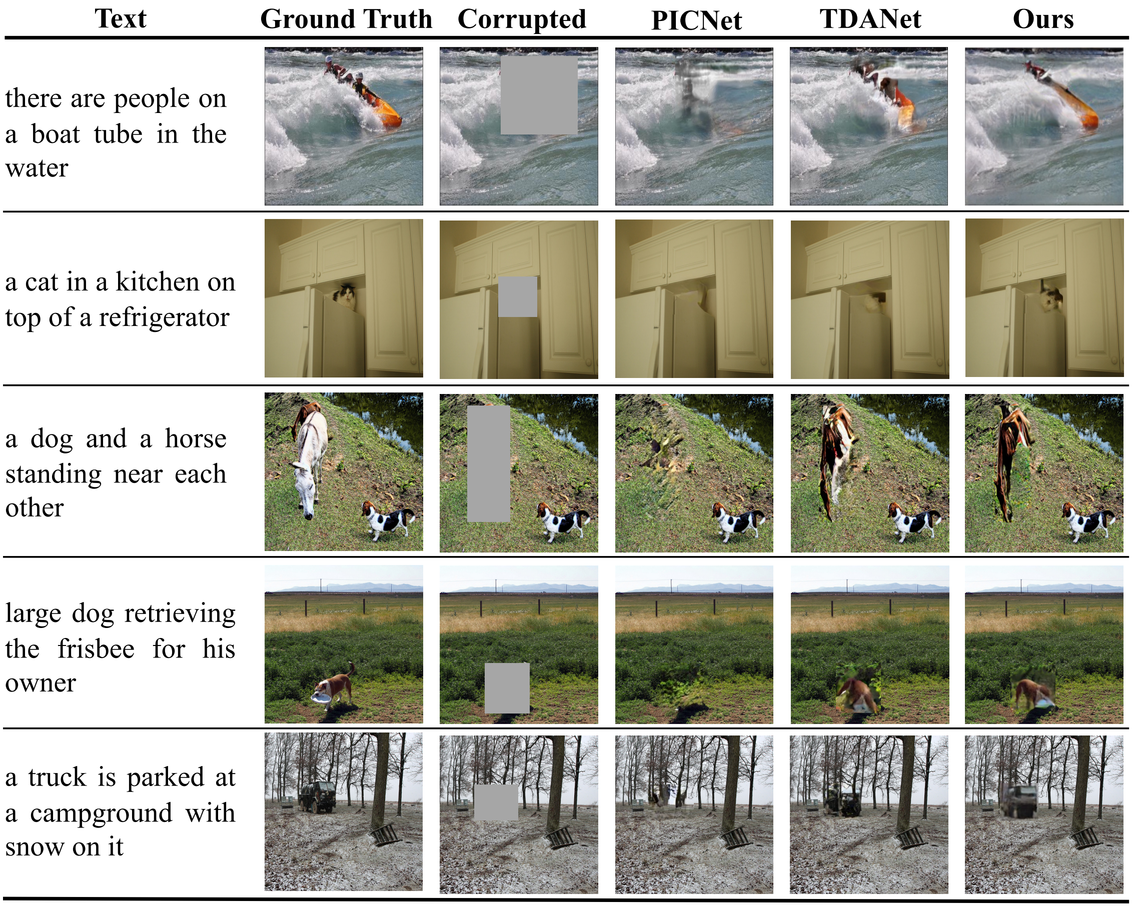


Figure 6: Qualitative results randomly sampled from MSCOCO test set.

the human perspective. We randomly collected 100 images in center masks from the MSCOCO test dataset. Each sample includes four generated images from CMA, PICNet, ALMR and CSA, respectively. These four images are randomly shuffled for five volunteers who rank images according to naturalness and the semantic consistency with the text description. Next, we computed the average ranking score as shown in Table 8 and show their distribution in Figure 5. From the results, we can observe that our model outperforms other competitors in naturalness and semantic consistency. It demonstrates the effectiveness of our method from a qualitative perspective.

3.6 Qualitative Results

As demonstrated in Figure 6, we give qualitative comparison examples of our method and other methods. We can find that our model can generate more semantically plausible objects in the missing region. Firstly, comparing PICNet with TDANet and our method, extracting the semantic information from the text can improve the repair effect of the model in the missing area of the image. Secondly, we can observe that TDANet can generate preliminary results, such as the outlines and part content of "cat", "truck" and "people" in examples, but many details of the object are very unnatural. In contrast, our method can further generate the details of the object based on generating the outline and content of the object. It demonstrates that cross-modal alignment can effectively supplement missing object information.

4 Related Work

4.1 Image Inpainting

Image inpainting (Bertalmío et al., 2000) aims to restore a damaged image, whose categories of approaches mainly are patch-based and deep learning-based. The patch-based methods (Barnes et al., 2009; Huang et al., 2014) fill the holes through searching and pasting patches based on image known regions. Huang et al. (2014) propose a method using the mid-level structural cues to automatically guide the image inpainting. However, these methods are not effective to fill in the missing region on complicated structures, due to the focus on low-level features. To address the limitation of existing patch-based methods, there has been growing interest in deep learning-based methods (Pathak et al., 2016; Iizuka et al., 2017; Ren et al., 2019). The Context Encoder (CE) is proposed by (Pathak et al., 2016), which uses the encoder-decoder architecture and the Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) to learn image features. Although the CE improves the inpainting by the image features learning, it is not effective to tackle the visual artifacts and exhibits blurriness in the image recovered regions. For solving the aforementioned problems, Iizuka et al. (2017) introduce the local and global discriminator for the image inpainting of arbitrary missing regions, which improves the local and global consistency of generated image. The StructureFlow (Ren et al., 2019) consisted of a structure reconstructor and a texture generator, which can focus on recovering global structures and synthesizing high-frequency details. Although the existing image inpainting methods can fill in the holes in the image, generating specific content in the missing region remains challenging, without any known information.

4.2 Text-Guided Image Inpainting

To address this problem, many text-guided image inpainting works are proposed (Zhang et al., 2020b; Lin et al., 2020). In these works, the specific content in the missing area can be restored based on the given descriptive text. Existing text-guided image inpainting methods include two processes: semantic information extraction and multimodal fusion. The semantic information extraction aims to obtain semantic information which does not match the image, such as the dual multimodal attention mechanism (Zhang et al., 2020a). However, these methods are hard to work well in the image set with a variety of different objects. The reason is that these methods lack fine-grained alignment knowledge of texts and images to guide the fusion of cross-modal information in multimodal interactions. Besides, their fusion modules lack powerful cross-modal reasoning capabilities.

4.3 Vision-Language Pre-training (VLP)

Motivated by the success of the language/vision pre-trained model (Devlin et al., 2019; Dosovitskiy et al., 2021; Zhou et al., 2022b), there is a surging interest in developing a pre-trained model for multiple modalities (e.g., vision and language) (Chen et al., 2020; Radford et al., 2021; Kim et al., 2021; Zhou, 2022). For example, a pioneering work CLIP (Radford et al., 2021) employs contrastive learning to predict whether matching between image and text and shows its powerful capability in many downstream tasks. UNITER (Chen et al., 2020) and UNIMO (Chen et al., 2020) employ an object detector (e.g., Faster R-CNN (Ren et al., 2017)) to capture vision features, and a multi-layer transformer (Vaswani et al., 2017) is used to joint learn vision features and text features. Kim et al. (2021) discuss different taxonomy of vision-and-language models and propose ViLT, a pre-trained model more focused on modeling modality interactions. In addition, ViLT totally discards convolutional visual features and adopts vision transformers.

5 Conclusion

In this work, we explore a novel CMA model for text-guided image inpainting. In the CMA model, we integrate a vision encoder and a text encoder into a vision-and-language encoder, which is different from previous works. The vision-and-language encoder allocates more computation on modeling modality interactions instead of visual encoding. In addition, we introduce two objectives to improve cross-modal alignment, dubbed cross-modal alignment and in-sample distillation. The cross-modal alignment objective guides the model to fuse cross-modal features. Experimental results demonstrate that the proposed model delivers new state-of-the-art performance, followed by further analyses to provide comprehensive insights.

Limitations

Compared with previous text-guided image inpainting methods (Zhang et al., 2020b; Lin et al., 2020; Wu et al., 2021), our method performs well for natural image datasets with a wide variety of objects. However, the recovery of our method for completely missing objects in images is not perfect. The reason is that the size of our model limits its capabilities. The large image generation models (e.g., DALL·E (Ramesh et al., 2021)) show their powerful capability to generate a plausible image. Due to the limitation of computational resources, we could not train a large model of similar size to DALL·E, which hinders verifying the effectiveness of our method on a large model.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223.

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. Patchmatch: A randomized correspondence algorithm for structural image editing. In *ACM Transactions on Graphics (ToG)*, volume 28, page 24. ACM.

Marcelo Bertalmío, Guillermo Sapiro, Vicent Caselles, and Coloma Ballester. 2000. Image inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2000, New Orleans, LA, USA, July 23-28, 2000*, pages 417–424. ACM.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: universal image-text representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Fernando A. Fardo, Victor H. Conforto, Francisco C. de Oliveira, and Paulo S. Rodrigues. 2016. A formal evaluation of PSNR as quality measurement parameter for image segmentation algorithms. *CoRR*, abs/1605.07116.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680.

Xiefan Guo, Hongyu Yang, and Di Huang. 2021. Image inpainting via conditional texture and structure dual generation. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 14114–14123. IEEE.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6626–6637.

Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. 2014. Image completion using planar structure guidance. *ACM Transactions on graphics (TOG)*, 33(4):129.

Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2017. Globally and locally consistent image completion. *ACM Transactions on Graphics (ToG)*, 36(4):107.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 5583–5594. PMLR.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross B. Girshick. 2020. Pointrend: Image segmentation as rendering. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9796–9805. Computer Vision Foundation / IEEE.

Qing Lin, Bo Yan, Jichun Li, and Weimin Tan. 2020. MMFL: multimodal fusion learning for text-guided image inpainting. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*, pages 1094–1102. ACM.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: common objects in context. In *Computer Vision -*

ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer.

Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. 2019. Coherent semantic attention for image inpainting. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 4169–4178. IEEE.

Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. 2016. Context encoders: Feature learning by inpainting. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2536–2544. IEEE Computer Society.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2017. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *Int. J. Comput. Vis.*, 123(1):74–93.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149.

Yurui Ren, Xiaoming Yu, Ruonan Zhang, Thomas H. Li, Shan Liu, and Ge Li. 2019. Structureflow: Image inpainting via structure-aware appearance flow. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 181–190. IEEE.

Leonid I Rudin, Stanley Osher, and Emad Fatemi. 1992. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. 2011. The caltech-ucsd birds-200-2011 dataset.

Ziyu Wan, Jingbo Zhang, Dongdong Chen, and Jing Liao. 2021. High-fidelity pluralistic image completion with transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 4672–4681. IEEE.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612.

Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. 2021. Simvlm: Simple visual language model pretraining with weak supervision. *CoRR*, abs/2108.10904.

Xingcai Wu, Yucheng Xie, Jiaqi Zeng, Zhenguo Yang, Yi Yu, Qing Li, and Wenyin Liu. 2021. Adversarial learning with mask reconstruction for text-guided image inpainting. In *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pages 3464–3472. ACM.

Lisai Zhang, Qingcai Chen, Baotian Hu, and Shuoran Jiang. 2020a. Text-guided neural image inpainting. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*, pages 1302–1310. ACM.

Zijian Zhang, Zhou Zhao, Zhu Zhang, Baoxing Huai, and Jing Yuan. 2020b. Text-guided image inpainting. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*, pages 4079–4087. ACM.

Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. 2019. Pluralistic image completion. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1438–1447. Computer Vision Foundation / IEEE.

Yucheng Zhou. 2022. Sketch storytelling. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pages 4748–4752. IEEE.

Yucheng Zhou, Xiubo Geng, Tao Shen, Guodong Long, and Daxin Jiang. 2022a. Eventbert: A pre-trained model for event correlation reasoning. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 850–859. ACM.

Yucheng Zhou, Tao Shen, Xiubo Geng, Guodong Long, and Daxin Jiang. 2022b. Claret: Pre-training a correlation-aware context-to-event transformer for event-centric generation and classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2559–2575. Association for Computational Linguistics.

Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. 2020. In-domain GAN inversion for real image editing. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVII*, volume 12362 of *Lecture Notes in Computer Science*, pages 592–608. Springer.

Semantic Specialization for Knowledge-based Word Sense Disambiguation

Sakae Mizuki and **Naoaki Okazaki**
Tokyo Institute of Technology
`{sakae.mizuki@nlp., okazaki@}c.titech.ac.jp`

Abstract

A promising approach for knowledge-based Word Sense Disambiguation (WSD) is to select the sense whose contextualized embeddings computed for its definition sentence are closest to those computed for a target word in a given sentence. This approach relies on the similarity of the *sense* and *context* embeddings computed by a pre-trained language model. We propose a semantic specialization for WSD where contextualized embeddings are adapted to the WSD task using solely lexical knowledge. The key idea is, for a given sense, to bring semantically related senses and contexts closer and send different/unrelated senses farther away. We realize this idea as the joint optimization of the Attract-Repel objective for sense pairs and the self-training objective for context-sense pairs while controlling deviations from the original embeddings. The proposed method outperformed previous studies that adapt contextualized embeddings. It achieved state-of-the-art performance on knowledge-based WSD when combined with the reranking heuristic that uses the sense inventory. We found that the similarity characteristics of specialized embeddings conform to the key idea. We also found that the (dis)similarity of embeddings between the related/different/unrelated senses correlates well with the performance of WSD.

1 Introduction

Word Sense Disambiguation (WSD) is the task of choosing the appropriate sense of a word from a given sense inventory using contextual information. WSD has proven its usefulness for Information Retrieval (Zhong and Ng, 2012) and Machine Translation (Campolungo et al., 2022). A series of extensive studies has led supervised WSD task performance to surpass the milestone of 80% accuracy (Bevilacqua and Navigli, 2020), which is the estimated human performance (Navigli, 2009).

In contrast, the goal of this study is *knowledge-based WSD*: a variant of WSD that does not rely on supervision data but only on lexical knowledge (e.g., word ontology). This task setting is practically appealing because it does not use a corpus with sense annotations (Bevilacqua et al., 2021), which is costly and labor-intensive to prepare.

A promising approach is based on similarity: to select the sense that is the nearest to a target word in the embedding space (Wang and Wang, 2020). Specifically, a pre-trained language model, typically BERT (Devlin et al., 2019), is used to compute *sense embeddings* for definition sentences. Similarly, a target word is encoded into a *context embedding* for a given sentence. Then, the model predicts the sense of the target word by finding the most similar sense embedding to the context.

The inherent challenge of the similarity-based approach is how we associate two different representations of word meanings, either by definition sentences or by words in context. Although the BERT embeddings capture the coarse-grained word meanings (Reif et al., 2019; Loureiro et al., 2021), there should be room for improvement. Notably, Wang and Wang (2020) proposed `SREF`, sense embedding adaptation by bringing semantically related senses closer. Extending their work, Wang et al. (2021b) proposed `COE`, context embedding enhancement heuristics during inference using the document-level global contexts of the given sentence, and reported the best performance. Despite being effective, `COE` cannot be applied to stand-alone texts, e.g., short messages on social media or search queries, limiting its applicability.

Our study aims to improve both accuracy and applicability to stand-alone texts. Specifically, we propose an adaptation method of the sense and context embeddings for the WSD task solely using lexical knowledge. Then, what are good embeddings for WSD? Our key idea is to 1) bring semantically related sense and context embeddings that convey the same meaning closer, and 2) send unrelated and/or different senses that share the same

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3457–3470
May 2-6, 2023 ©2023 Association for Computational Linguistics

surface form farther away (Fig. 1-d). We formulate the idea as the Attract-Repel objective and self-training objective. The main novelty is the joint optimization to utilize their complementary nature: the former should improve the distinguishability between senses whereas the latter offers pseudo signals of context-sense associations, which has not been explored in previous methods.

The Attract-Repel objective, inspired by Vulic and Mrksic (2018), injects semantic relation knowledge into the similarity of sense pairs. Specifically, we make semantically related senses more similar while making different and unrelated senses more dissimilar (Fig. 1-a). While `SREF` performs Attract only, our method utilizes both Attract and Repel.

The self-training objective, inspired by the idea of retraining on the classifier's own predictions instead of annotated senses (Navigli, 2009), updates the similarity of context-sense pairs in a pseudo labeling manner (§ 6.1). Specifically, for each training step and given context, we bring the nearest neighbor sense among candidates closer (Fig. 1-b). We also impose distance constraints during adaptation to control the deviation from BERT embeddings (Fig. 1-c) because excessive deviation may cause an inaccurate nearest neighbor sense selection, which would cause a performance drop.

We call the overall proposed method `SS-WSD`, Semantic Specialization for WSD, following Vulic and Mrksic (2018). We evaluated `SS-WSD` using the standard evaluation protocol (Raganato et al., 2017) and confirmed that it outperforms the previous embeddings adaptation method. Furthermore, it achieved state-of-the-art (SoTA) performance when combined with the reranking heuristic that uses a sense inventory (Wang and Wang, 2021), and thus is applicable to stand-alone texts.

The contributions of our study are as follows:

- We proposed `SS-WSD`, an embedding adaptation method that achieves new SoTA in knowledge-based WSD, regardless of the availability of document-level global contexts.
- We found that the performance gain originates from the joint optimization of Attract-Repel and self-training objectives and the prevention of deviation from the original embeddings.
- Empirically, we found that the similarity of related/different/unrelated senses *relative to* the similarity of ground-truth context-sense pairs correlates well with the WSD performance.

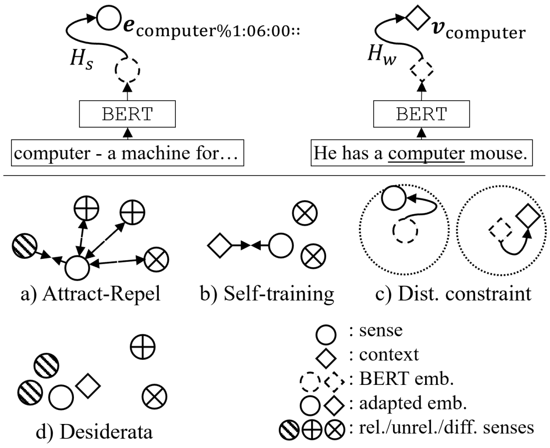


Figure 1: Schema of the proposed method. The BERT embeddings representing senses and contexts are adapted by transformation (top). Transformation functions are optimized using Attract-Repel and self-training objectives under distance constraints so that the adapted embeddings are effective for WSD (bottom).

2 Related Work

2.1 Knowledge-based WSD

Knowledge-based WSD is a variant of WSD that does not use a sense annotation corpora such as the SemCor (Miller et al., 1993) but uses lexical resources instead, typically WordNet. The majority vote based on sense frequencies, also known as the WordNet first sense heuristic (Jurafsky and Martin, 2009), is a simple but strong baseline method of this category. Sense definitions and usage examples are also used to measure the similarity of the target word in a sentence. The simplest method is based on word overlap (Lesk, 1986).

One recent direction is the use of BERT as a contextualized encoder. BERT embeddings showed empirical success on the supervised WSD task when used as features. Some analyses reported that BERT embeddings capture the coarse-grained word meanings (Reif et al., 2019; Loureiro et al., 2021). Wang and Wang (2020) proposed a similarity-based method in the embedding space. It chooses the sense which has the most similar embedding, formed from the concatenation of its lemma, definition, and usage examples, to the embedding of a target word. They also proposed the Semantic Relation Enhancement Framework ($\texttt{SREF}_{\texttt{emb}}$), which adapts sense embeddings by weighted averages over semantically related senses, e.g., hyponyms and derivations. $\texttt{SREF}_{\texttt{emb}}$ is the most high-performing adaptation method so far. We report that our proposed method achieves better performance.

2.2 Heuristics for Knowledge-based WSD

Another recent direction is the heuristics for choosing the most similar sense, which is further divided into those that use the sense inventory information and those that exploit the document-level global contexts of a given sentence. Wang and Wang (2020) proposed the former, the Try-again Mechanism (TaM). It reranks candidates by adding the similarity between the target word and the lexicographer class (supersense) that a candidate sense belongs to. Subsequent studies (Wang et al., 2021b; Wang and Wang, 2021) refined TaM using Coarse Sense Inventory (Lacerra et al., 2020). We examine the effectiveness of the proposed method combined with TaM because it can be applied to stand-alone texts.

Wang et al. (2021b) proposed contextual information enhancement (CIE), which enhances context embeddings by exploiting the document-level global contexts of a given sentence on evaluation. This idea originally stems from the one-sense-per-discourse hypothesis (Gale et al., 1992): that the sense of a word is highly consistent within a document.

2.3 Attract-Repel Framework

The Attract-Repel Framework is used to inject lexical knowledge into embeddings by encouraging similar instances to have closer embeddings while encouraging dissimilar instances to be farther away. Vulic and Mrksic (2018) and Mrkšić et al. (2017) reported that updating static word embeddings using lexical knowledge improves the performance of the word-level semantic relation classification task. Our study proposes its application to sense and context embeddings for the WSD task. We also reformulate the original loss function with the contrastive loss, inspired by its success in Computer Vision (Chen et al., 2020) and NLP (Gao et al., 2021; Wang et al., 2021a; Giorgi et al., 2021).

2.4 Supervised WSD

Supervised methods rely on corpora of sense-annotated contexts, such as SemCor, for training models. However, the coverage of words and senses is limited and biased towards more frequent senses (Pasini, 2020). Recent studies have addressed these limitations by incorporating lexical resources into the methods. Barba et al. (2021a) and its subsequent study (Barba et al., 2021b) reframed WSD as a span extraction task by appending definition sentences of candidate senses to the target context. They reached the SoTA performance among supervised methods.

Similarity-based approaches are also used with supervised methods. Supervised k-nearest neighbors (`Sup-kNN`) (Loureiro and Jorge, 2019) defines sense embeddings as the averaged context embeddings of annotated senses. The Bi-Encoder model (`BEM`) (Blevins and Zettlemoyer, 2020) jointly fine-tunes two BERT encoders for definition sentences and contexts, ensuring that context embeddings will be closer to the correct sense embeddings. The proposed method is similar in architectural design to `BEM`, but differs in that we do not fine-tune the BERT encoders. We will compare our results with `Sup-kNN` and `BEM` to assess the effect of using no sense annotation and of freezing BERT encoders on performance.

3 Semantic Specialization for WSD

3.1 Formalization of WSD

The proposed method adapts BERT embeddings by trainable transformation functions H_s and H_w:

$$\mathbf{v}_w = H_w(\hat{\mathbf{v}}_w), \tag{1}$$

$$\mathbf{e}_s = H_s(\hat{\mathbf{e}}_s), \tag{2}$$

where the inputs $\hat{\mathbf{v}}_w$ and $\hat{\mathbf{e}}_s$ are the context and sense embeddings computed by a BERT encoder and the outputs $\mathbf{v}_w$ and $\mathbf{e}_s$ are the specialized embeddings.

We train the transformation functions by minimizing the weighted sum of the Attract-Repel objective and the self-training objective on the specialized embeddings. Note that the BERT encoder is frozen (not fine-tuned). We integrate the constraints on the distance between the input and output into the architecture of transformation functions (§ 3.4).

To predict a sense for a given target word w, we look up the candidate senses $\mathcal{S}_w$ and compute their specialized sense embeddings using the learned function H_s. Similarly, we compute specialized context embeddings using H_w. Then, we select the nearest neighbor sense s^* using cosine similarity:

$$s^* = \underset{s' \in \mathcal{S}_w}{\arg\max}\, \rho_{w,s'}, \tag{3}$$

$$\rho_{w,s} = \cos(\mathbf{v}_w, \mathbf{e}_s) = \frac{\mathbf{v}_w \cdot \mathbf{e}_s}{\|\mathbf{v}_w\| \|\mathbf{e}_s\|}. \tag{4}$$

Element	Noun	Verb	Adj.	Adv.	All
# Lemmas	117,798	11,529	21,479	4,481	155,287
# Senses	146,320	25,047	30,002	5,580	206,949
Rel. senses	7.8	13.0	6.2	3.9	8.1
Diff. senses	0.8	4.1	1.2	0.7	1.3

Table 1: Summary statistics of lexical resources by part-of-speech tag. Values in the related and different senses rows indicate the average per sense.

3.2 Lexical Knowledge in WordNet

We use WordNet (Fellbaum, 1998) as a lexical resource and sense inventory. WordNet mainly consists of synsets, lemmas, and senses. A synset is a group of synonymous words that convey a specific meaning. A lemma presents a canonicalized form of a word and belongs to one or more synsets. A sense is the lemma disambiguated by a sense key, and belongs to a single synset. We use the sense key as the identifier of a sense.

The proposed method makes use of relational knowledge between senses for training the transformation functions. Specifically, for each sense s, we collect three sets of senses: *related* $\mathcal{S}_s^P$, *different* $\mathcal{S}_s^N$, and *unrelated* $\mathcal{S}_s^U$. The *related* set consists of sense keys of synonyms and semantically related senses (e.g., hyponyms) to the target sense. We followed the definition of related senses used in Wang and Wang (2020) (Appendix A). The *different* set consists of sense keys sharing the same lemma to the target sense excluding itself. In other words, the different senses correspond to the polysemy of the lemma of the target sense. The *unrelated* set presents sense keys that are randomly chosen from the sense inventory (see § 3.5.1 for details). Table 1 shows the statistics of lemmas and senses. See Table 6 (in Appendix A) for examples of the concepts explained in this subsection.

3.3 BERT Embeddings for Sense and Context

For obtaining BERT embeddings, we follow the standard practice of the previous studies (Wang et al., 2020; Bevilacqua and Navigli, 2020; Wang and Wang, 2020). Specifically, we use `bert-large-cased`[1] with special tokens `[CLS]` and `[SEP]`. For each subword, we compute a sum over outputs at the last four layers of Transformer blocks.

A context embedding is the average of BERT embeddings over constituent subwords. For the computation of sense embeddings, we follow the method that Wang and Wang (2020) used. See Appendix B for details.

3.4 Transformation Functions

The proposed method adapts embeddings by applying the trainable transformation, i.e., the specialization is learned by optimizing the transformation functions. This approach enables the adaptation of context embeddings on the fly during inference, which was not possible in the original approach that directly learns adapted embeddings (Vulic and Mrksic, 2018).

Let $\hat{\mathbf{v}}_w$ and $\hat{\mathbf{e}}_s$ be context and sense BERT embeddings. We transform them independently using residual mapping functions F_w and F_s, which are both two-layer feedforward networks, FFNN_w and FFNN_s. These networks are comprised of a linear layer with a ReLU activation, followed by a linear layer with a sigmoid activation.

$$\mathbf{v}_w = H_w(\hat{\mathbf{v}}_w) = \hat{\mathbf{v}}_w + \epsilon\|\hat{\mathbf{v}}_w\|F_w(\hat{\mathbf{v}}_w), \quad (5)$$

$$\mathbf{e}_s = H_s(\hat{\mathbf{e}}_s) = \hat{\mathbf{e}}_s + \epsilon\|\hat{\mathbf{e}}_s\|F_s(\hat{\mathbf{e}}_s), \quad (6)$$

$$F_w(\hat{\mathbf{v}}_w) = 2\sigma(\mathrm{FFNN}_w(\hat{\mathbf{v}}_w)) - 1, \quad (7)$$

$$F_s(\hat{\mathbf{e}}_s) = 2\sigma(\mathrm{FFNN}_s(\hat{\mathbf{e}}_s)) - 1, \quad (8)$$

where $\mathbf{v}_w$ and $\mathbf{e}_s$ are the specialized embeddings. ϵ is the hyperparameter that controls how far away the specialized embeddings can be. Specifically, the L2 distance relative to the original embedding $\|\mathbf{v}_w - \hat{\mathbf{v}}_w\|/\|\hat{\mathbf{v}}_w\|$ is bounded by $\epsilon\sqrt{N_d}$, where N_d is the dimension size of embeddings[2]. This is because the residual functions map the inputs to the space $[-1, +1]^{N_d}$.

3.5 Objectives

We jointly optimize the Attract-Repel objective for sense pairs and the self-training objective for context-sense pairs by minimizing the weighted sum of the loss functions,

$$L = L^{\mathrm{AR}} + \alpha L^{\mathrm{ST}}, \quad (9)$$

where α is the hyperparameter that determines the relative importance of the self-training objective.

The joint optimization is motivated by the complementary nature of these two objectives. The Attract-Repel objective should improve the separability of similar/different senses but does not contribute to determining which context and sense should be associated. In contrast, the self-training objective provides pseudo-supervision signals for

---

[1] We use `transformers` package (Wolf et al., 2020).

[2] $N_d = 1{,}024$ for `bert-large-cased`.

context-sense associations, although the informativeness is, when used alone, limited because it essentially reinforces the similarity to the initial nearest neighbor sense of the target context (§ 3.5.2).

3.5.1 Attract-Repel Objective

We formulate Attract-Repel objective loss L^{AR} using contrastive loss: we bring *related* senses closer while *different* and *unrelated* senses farther away[3] (§ 3.2). Specifically, for a given minibatch of senses $\mathcal{S}^B$ and a specific sense $s \in \mathcal{S}^B$, we define the subset excluding itself $\mathcal{S}^B \setminus \{s\}$ as the unrelated senses $\mathcal{S}_s^U$. Then, we randomly choose a sense s_p from the related senses $\mathcal{S}_s^P$. Similarly, we randomly choose up to five senses without replacement $\tilde{\mathcal{S}}_s^N$ from different senses $\mathcal{S}_s^N$. Finally, L^{AR} for the minibatch $\mathcal{S}^B$ is defined as follows:

$$L^{\mathrm{AR}} = -\sum_{s \in \mathcal{S}^B} \ln \frac{e^{\beta \rho_{s,s_p}}}{\sum\limits_{s' \in \left(\{s_p\} \cup \mathcal{S}_s^U \cup \tilde{\mathcal{S}}_s^N\right)} e^{\beta \rho_{s,s'}}}, \quad (10)$$

$$\rho_{s,s'} = \cos(\mathbf{e}_s, \mathbf{e}_{s'}). \quad (11)$$

We set the scaling parameter β to 64, following the suggestions in metric learning studies (Deng et al., 2019; Wang et al., 2018).

3.5.2 Self-training Objective

We formulate the self-training objective loss L^{ST} so that we bring the contexts and nearest neighbor senses closer. In the self-training process, we label a word in context with the sense whose embedding is the closest to that of the word. Specifically, let $\mathcal{W}^B$ denote a minibatch of words. For a word $w \in \mathcal{W}^B$, we obtain a set of candidate senses[4] $\mathcal{S}_w$. Then, L^{ST} for the minibatch $\mathcal{W}^B$ is defined as,

$$L^{\mathrm{ST}} = \sum_{w \in \mathcal{W}^B} (1 - \max_{s \in \mathcal{S}_w} \rho_{w,s}), \quad (12)$$

$$\rho_{w,s} = \cos(\mathbf{v}_w, \mathbf{e}_s). \quad (13)$$

Note that the nearest neighbor sense for the same context changes during training as we update parameters of the transformation functions for embeddings. Our intention is to bootstrap the performance, which was impossible in the "static counterpart", e.g., pseudo-labeling with the WordNet first sense heuristic. That is also a motivation of introducing the distance constraint in Eq. 5 and 6: we were concerned about the performance drop when a large deviation occurs in the semantic specialization. We report empirical evidence that the constraint improves the performance (§ 6.3).

In principle, the training data can be any corpus annotated with lemmas and part-of-speech tags. Nevertheless, we used the SemCor (Miller et al., 1993) corpus with the sense annotations removed. This is because using these de-facto standard corpora contributes to better reproducibility and fairer comparisons.

3.6 Try-again Mechanism (TaM) Heuristic

We examine the effectiveness of the proposed method when combined with TaM. Specifically, we employ the variant (Wang and Wang, 2021)[5] that utilizes Coarse Sense Inventory (CSI) (Lacerra et al., 2020) because of its simplicity. In essence, TaM reranks candidate senses by updating similarities under the assumption that the context should be also similar to the coarse semantic category that the candidate sense belongs to. Let s_1 and s_2 be the top two nearest neighbors for the target word w and $\mathcal{S}_s^{\mathrm{CSI}}$ be the set of senses[6] belonging to the same CSI class as s belongs to. Then, we refine the similarity $\rho_{w,s}^+$ for each $s \in \{s_1, s_2\}$,

$$\rho_{w,s}^+ = \rho_{w,s} + \max_{s' \in \mathcal{S}_s^{\mathrm{CSI}}} \rho_{w,s'}. \quad (14)$$

Finally, we choose the sense from s_1 and s_2 with highest similarity using $\rho_{w,s}^+$, i.e., we use the refined similarity $\rho_{w,s}^+$ instead of $\rho_{w,s}$ (Eq. 3).

4 Experiment Settings

4.1 Training

We used WordNet senses for optimizing the Attract-Repel objective and the sense-annotated words in the SemCor corpus for the self-training objective. Note that we solely use lemmas and part-of-speech tags and disregard the sense annotations. The number of senses in WordNet is 206,949, and the number of words in the corpus is 226,036. We independently sampled minibatches N_B for each objective. For the Attract-Repel objective, we iterate

---

[3] In the contrastive learning literature, related, unrelated, and different senses correspond to the positives, weak negatives, and hard negative examples, respectively.

[4] Querying WordNet for a tuple of lemma and part-of-speech tag returns the candidate senses.

[5] We followed author's implementation: `https://github.com/lwmlyy/SACE`

[6] $\mathcal{S}_s^{\mathrm{CSI}}$ will be the empty set if s doesn't exist in the CSI because it does not cover all synsets.

over all sense keys in the WordNet with 15 epochs[7]. For hyperparameter optimization, we disabled TaM heuristics and used the evaluation set of SemEval-2007 as the development set, following the standard practice (Pasini et al., 2021). See Appendix C for details of the hyperparameter search. We set $N_B = 256$, $\alpha = 0.2$, and $\epsilon = 0.015$. We used the Adam optimizer with learning rate 0.001.

4.2 Evaluation

For evaluation, we used the WSD unified evaluation framework (Raganato et al., 2017)[8]. We used the nearest neighbor sense as the prediction (Eq. 3). For the evaluation metric, we adopt the micro-averaged F1 score[9] that is commonly used in the literature. Unless otherwise specified, we run the training process five times with different random seeds, and report the mean and standard deviations.

4.3 Baselines

We compare the proposed method in two experimental configurations: *Intrinsic* and *With Heuristics*. For the *Intrinsic* configuration, we compare it with the methods that do not use any heuristic. Specifically, we choose `PlainBERT` and $\texttt{SREF}_{\texttt{emb}}$ (Wang and Wang, 2020) as baselines. `PlainBERT` uses BERT embeddings $\hat{\mathbf{v}}_w$ and $\hat{\mathbf{e}}_s$ as is. $\texttt{SREF}_{\texttt{emb}}$[10] adapts sense embeddings so that it brings semantically related senses closer. For the *With Heuristics* configuration, we compare the proposed method with the methods that combine heuristics. Specifically, we choose $\texttt{SREF}_{\texttt{kb}}$ (Wang and Wang, 2020) and `COE` (Wang et al., 2021b) as baselines. $\texttt{SREF}_{\texttt{kb}}$ combines $\texttt{SREF}_{\texttt{emb}}$ with TaM. `COE` also utilizes $\texttt{SREF}_{\texttt{emb}}$, but it employs refined TaM and CIE. `COE` is the current SoTA method on knowledge-based WSD.

We also compare with supervised methods which employ the similarity-based approach to assess the effect of not using sense annotations and of freezing BERT encoders. Specifically, we compare with `Sup-kNN` (Loureiro and Jorge, 2019) and `BEM` (Blevins and Zettlemoyer, 2020) (§ 2.4), which both use SemCor as the trainset. `Sup-kNN` computes sense embeddings as the context embeddings averaged over the annotated senses. `BEM` fine-tunes BERT encoders so that context embeddings and correct sense embeddings are brought closer. We consider `BEM` as the de-facto upper bound of similarity-based approach, given its usage of a supervision signal to fine-tune the BERT encoders.

5 Experimental Results

Table 2 shows the WSD task performance. In both configuration, the proposed method $\texttt{SS-WSD}_{\texttt{emb}}$ outperformed all knowledge-based baselines.

In the *Intrinsic* configuration, $\texttt{SS-WSD}_{\texttt{emb}}$ outperformed $\texttt{SREF}_{\texttt{emb}}$ by 3.9pt, which is as much as a 9.3pt improvement over `PlainBERT`. Looking at the results for each part-of-speech, we observed the largest improvement over $\texttt{SREF}_{\texttt{emb}}$ for verbs (9.0pt). This result reflects the fact that verbs have the richer supervision signal for the Attract-Repel objective because of the largest number of related and different senses (Table 1) for verbs. This suggests that the richer semantic relation knowledge is, the higher performance the proposed method may achieve.

In the *With Heuristics* configuration, $\texttt{SS-WSD}_{\texttt{kb}}$ outperformed `COE` by 0.8pt without using the CIE heuristic, which shows an advantage over the baselines regardless of whether the evaluation sentence is a stand-alone text or in a document. The improvement brought by TaM was 2.2pt. Although $\texttt{SS-WSD}_{\texttt{kb}}$ lagged behind `COE` on the SE07 (SemEval-2007) subset, we think this result is understandable because `COE` also used SE07 for hyperparameter optimization.

When compared to supervised methods, $\texttt{SS-WSD}_{\texttt{emb}}$ outperformed `Sup-kNN` by 1.4pt, while falling behind `BEM` by 4.1pt. The results indicate that the proposed method associates contexts with senses more precisely than the example-based sense embeddings computation using sense-annotated contexts. It also shows the effectiveness of the supervised fine-tuning of BERT encoders in `BEM`, as evidenced through their ablation study (Blevins and Zettlemoyer, 2020).

6 Analysis

6.1 Vanilla BERT Embeddings

The proposed method adapts the BERT embeddings (`PlainBERT`) by transformation. Therefore,

---

[7] In each epoch, we discarded the remaining examples in the self-training objective trainset once all sense keys have been traversed.

[8] Available at: `http://lcl.uniroma1.it/wsdeval/`

[9] Note that F1 score is equal to Precision and Recall (Pasini et al., 2021) because proposed method predicts a single sense.

[10] We applied their method to `PlainBERT`, consistent with the proposed method, to ensure a fair comparison of the effect of adaptation.

Method	TaM	CIE	By subset					By part-of-speech				All
			SE2	SE3	SE07	SE13	SE15	Noun	Verb	Adj.	Adv.	
Supervised												
`Sup-kNN` (Loureiro and Jorge, 2019)	×	×	76.3	73.2	66.2	71.7	74.1	—	—	—	—	73.5
`BEM` (Blevins and Zettlemoyer, 2020)	×	×	79.4	77.4	<u>74.5</u>	79.7	81.7	81.4	68.5	83.0	87.9	79.0
Knowledge-based, *Intrinsic* configuration												
`PlainBERT`	×	×	67.8	62.7	54.5	64.5	72.3	67.8	52.3	74.0	77.7	65.6
`SREF`$_{\texttt{emb}}$ (Wang and Wang, 2020)	×	×	70.3	68.0	60.4	74.2	77.4	76.3	53.5	75.2	76.3	71.0
`SS-WSD`$_{\texttt{emb}}$ (Ours)	×	×	**74.6*** (0.5)	**73.0*** (0.6)	<u>**65.0**</u>* (1.3)	**77.0*** (0.5)	**79.9*** (1.0)	**78.2*** (0.4)	**62.5*** (0.7)	**79.7*** (0.3)	**80.5*** (1.5)	**74.9*** (0.3)
Knowledge-based, *With Heuristics* configuration												
`SREF`$_{\texttt{kb}}$ (Wang and Wang, 2020)	✓	×	72.7	71.5	61.5	76.4	79.5	78.5	56.6	79.0	76.9	73.5
`COE` (Wang et al., 2021b)	✓	✓	76.0	74.2	<u>**69.2**</u>	**78.2**	80.9	**80.6**	61.4	80.5	81.8	76.3
`SS-WSD`$_{\texttt{kb}}$ (Ours)	✓	×	**77.7*** (0.5)	**75.9*** (0.6)	<u>66.5</u> (1.0)	78.0 (0.5)	**81.6** (0.9)	79.3 (0.3)	**65.7*** (0.8)	**84.9*** (0.4)	**84.2*** (0.8)	**77.1*** (0.3)

Table 2: WSD performance by subset and part-of-speech tag. `SS-WSD`$_{\texttt{emb,kb}}$ are the proposed methods. Numbers in parentheses represent the standard deviation. Asterisks (*) indicate that the difference to the best baseline is statistically significant at $p < 0.05$ by the Student's t-test (two-tailed test). Checkmarks (✓) in the TaM and CIE columns represent the usage of those heuristics. We bolded the best result among knowledge-based methods in each configuration and underlined the objective for hyperparameter tuning. The scores of `BEM`, `Sup-kNN`, `SREF`$_{\texttt{kb}}$, and `COE` are taken from the original papers.

Method	WSD (All)
`WN1`$^{\texttt{st}}$`Sense`	65.2
`PlainBERT`	65.6

Table 3: F1 score of BERT embeddings (`PlainBERT`) and WordNet the first sense heuristic (`WN1`$^{\texttt{st}}$`Sense`).

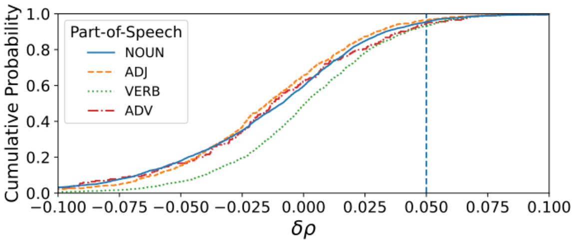


Figure 2: Cumulative distribution of the similarity margin between the incorrect sense and correct sense: $\delta\rho_w = \max_{s'\in\mathcal{S}_w\setminus\mathcal{S}_w^{\mathrm{gt}}} \rho_{w,s'} - \max_{s'\in\mathcal{S}_w^{\mathrm{gt}}} \rho_{w,s'}$, where $\mathcal{S}_w^{\mathrm{gt}}$ is the set of ground-truth senses of the word w.

its performance is influenced by the ability of `PlainBERT` to disambiguate senses.

Table 3 shows the WSD task performance using `PlainBERT`. We also reported the WordNet first sense heuristic (`WN1`$^{\texttt{st}}$`Sense`) for reference. We observe that `PlainBERT` is comparable to `WN1`$^{\texttt{st}}$`Sense`, indicating that self-training is a more effective strategy than `WN1`$^{\texttt{st}}$`Sense` for obtaining pseudo sense labels.

Fig. 2 shows the distribution of the similarity margin (difference) between the nearest neighbor incorrect sense and ground-truth sense computed by `PlainBERT`. We used the evaluation set for this analysis. We found that the similarity margin is below 0.05 for approximately 90% of all instances. This indicates that a large deviation from `PlainBERT` is not necessary for replacing nearest neighbor senses with the ground-truth ones.

6.2 Effect of Objectives

Table 4 shows the performance comparison when we eliminate a specific component from the semantic specialization objectives (§ 3.5). We keep all hyperparameters unchanged.

When we exclude either the Attract-Repel objective or the self-training objective, we see the performance drop by 3.3pt and 4.4pt, respectively. This finding supports the claim that joint optimization is crucial for its complementary nature.

When we remove either the unrelated senses or different senses from the Attract-Repel objective, we also see the performance drop by 5.0pt and 1.4pt, respectively. This result supports the idea that bringing semantically unrelated and different senses farther away contributes to performance. We also find that unrelated senses are more effective than different senses. A possible cause is the number of examples: while the number of unrelated

Ablation	WSD (All)	Δ[pt]
`SS-WSD`$_{\texttt{emb}}$	74.9	—
-Attract-Repel *objective*	71.6	-3.3
-Self-training *objective*	70.5	-4.4
-Unrelated senses $\mathcal{S}^U$ *repelling*	69.9	-5.0
-Different senses $\mathcal{S}^N$ *repelling*	73.5	-1.4
-Context *adaptation*	71.7	-3.2

Table 4: Ablation study of training objective. *Objective* rows represent the corresponding objective is excluded. *Repelling* rows represent the corresponding sense pairs are removed from the Attract-Repel objective (Eq. 10). *Adaptation* rows represent the usage of identity transformation. All differences are statistically significant at $p < 0.05$ by Welch's *t*-test (two-tailed test).

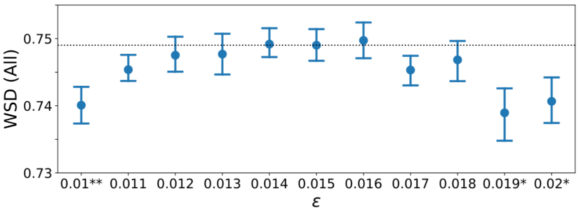


Figure 3: Ablation study of hyperparameter ϵ (§ 3.4). Dot and error bar represent the mean and standard deviation, respectively. Horizontal line represents the default setting ($\epsilon = 0.015$) performance. Asterisks indicate that the difference with respect to the default setting is statistically significant at $p < 0.05$ (*) and $p < 0.005$ (**) by Welch's *t*-test (two-tailed test).

senses is always[11] 255, the number of different senses is, on average, just 1.3 (see Table 1)[12].

Disabling the adaptation of context embeddings (by using identity transformation) caused a performance drop of 3.2pt, indicating that adapting both sense and context embeddings is necessary.

6.3 Effect of Distance Constraint

Fig. 3 shows the performance comparison when we change ϵ, the hyperparameter that bounds how farther away the specialized embeddings can be, in the interval [0.01,0.02] with a step size of 0.001. We found that performance follows an inverted U-shaped curve along ϵ, indicating that a sweet spot exists. Briefly, it shows that a severe constraint (small ϵ) results in an insufficient update for replacing nearest neighbors with ground-truth senses. In contrast, a looser constraint (large ϵ) results in a substantial deviation, eventually making the self-training less effective in the training process. The latter fact supports the claim that controlling the deviation from the original embeddings is necessary.

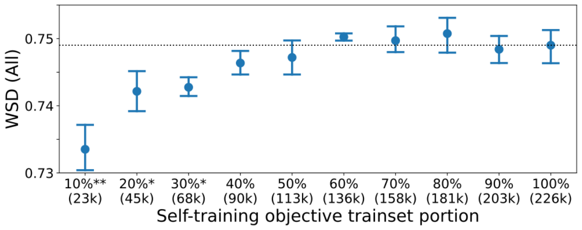


Figure 4: Impact of varying the self-training dataset size from 10% (23k examples) to 100% (224k). The dot and error bar indicates the mean and standard deviation, respectively. The horizontal line represents the performance when utilizing the 100% examples. Asterisks denote that the deviation from the 100% is statistically significant at $p < 0.05$ (*) and $p < 0.005$ (**) by Welch's *t*-test (two-tailed test).

7 Effect of Self-training Dataset Size

Fig. 4 illustrates the impact of varying the number of examples used for the self-training objective on the WSD task performance. It should be noted that 100% in the figure corresponds to using all examples in the SemCor corpus. We found that performance improves as the number of examples increases and reaches a saturation point at 60%, corresponding to 136k examples. While the coverage of words and senses appearing in the contexts also matters, it indicates that the benefits of self-training do not necessarily increase with the scaling to millions of examples.

7.1 Similarity Characteristics

We quantitatively investigate how well the proposed method achieved the key idea (Fig. 1-d): bringing related senses and contexts closer while unrelated and different senses farther away. Specifically, in Table 5, we reported averages of similarity values between related senses $\rho_{\mathcal{S}^P}$, unrelated senses $\rho_{\mathcal{S}^U}$, and different senses $\rho_{\mathcal{S}^N}$, along with averages of similarity values between ground-truth context-sense pairs[13] $\rho_{\mathcal{W}^{\text{gt}}}$. See Appendix D for formal definitions. We found that the proposed method `SS-WSD`$_{\texttt{emb}}$ brought context-sense pairs closer than `PlainBERT` ($\rho_{\mathcal{W}^{\text{gt}}}$: $0.64 \rightarrow 0.77$). In contrast, it pushed the unrelated and different senses away: $\rho_{\mathcal{S}^U}$:$0.77 \rightarrow 0.64$ and $\rho_{\mathcal{S}^N}$:$0.87 \rightarrow 0.78$. These results demonstrate that joint optimization of the Attract-Repel and self-training objectives realized the key idea successfully.

Can we expect better performance if we realize the key idea more precisely? We investigated the

---

[11] Minibatch size (=256) minus one yields 255.

[12] In fact, only 38% of all senses have different senses.

[13] We used sense-annotated words in the evaluation dataset.

Models	$\rho_{\mathcal{S}^P}$	$\rho_{\mathcal{S}^U}$	$\rho_{\mathcal{S}^N}$	$\rho_{\mathcal{W}^{\mathrm{gt}}}$	$\Delta\rho_{\mathcal{S}^P}\uparrow$	$\Delta\rho_{\mathcal{S}^U}\downarrow$	$\Delta\rho_{\mathcal{S}^N}\downarrow$	$\overline{\Delta\rho}\uparrow$	WSD (All)
`PlainBERT`	0.91	0.77	0.87	0.64	0.27	0.12	0.23	-0.030	65.6
`SS-WSD`$_{\texttt{emb}}$	0.88	0.64	0.78	0.77	0.11	-0.13	0.01	0.078	74.9
-Attract-Repel	0.92	0.79	0.90	0.81	0.11	-0.02	0.08	0.014	71.6
-Self-training	0.88	0.64	0.78	0.61	0.27	0.02	0.17	0.027	70.5
-Unrelated senses	0.90	0.73	0.79	0.73	0.17	0.00	0.06	0.033	69.9
-Different senses	0.87	0.61	0.79	0.77	0.09	-0.17	0.02	0.081	73.5
-Context adaptation	0.88	0.64	0.78	0.63	0.25	0.01	0.15	0.032	71.7

Table 5: Similarity characteristics of sense pairs and context-sense pairs. $\rho_{\mathcal{S}^P}$, $\rho_{\mathcal{S}^U}$, and $\rho_{\mathcal{S}^N}$ are the similarity to related, unrelated, and different senses (Eq. 15). $\rho_{\mathcal{W}^{\mathrm{gt}}}$ is the similarity of the context and its ground-truth senses (Eq. 16). $\Delta\rho_*$ is the difference to $\rho_{\mathcal{W}^{\mathrm{gt}}}$ (Eq. 17). $\overline{\Delta\rho} = \frac{1}{3}(\Delta\rho_{\mathcal{S}^P} - \Delta\rho_{\mathcal{S}^U} - \Delta\rho_{\mathcal{S}^N})$. Uparrow↑ (downarrow↓) represents the positive (negative) direction is favorable. WSD (All) are replicated from Tables 2 and 4 for reference.

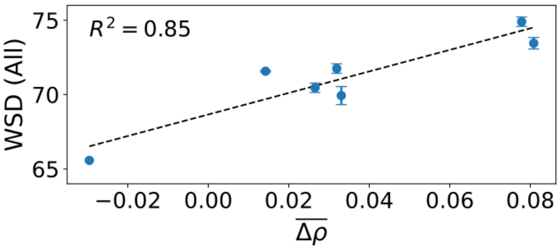


Figure 5: The relationship between the similarity characteristic metric $\overline{\Delta\rho}$ and WSD performance in Table 5.

relationship between these similarity metrics and WSD task performance. Specifically, we subtract $\rho_{\mathcal{W}^{\mathrm{gt}}}$ from each metric in order to capture the closeness of senses *relative to* the correct context-sense pairs, defining $\Delta\rho_*$ as $\rho_* - \rho_{\mathcal{W}^{\mathrm{gt}}}$. For example, $\Delta\rho_{\mathcal{S}^N} = \rho_{\mathcal{S}^N} - \rho_{\mathcal{W}^{\mathrm{gt}}}$ should be a negative value because the average similarity among different senses $\rho_{\mathcal{S}^N}$ should be smaller than that among correct context-sense pairs $\rho_{\mathcal{W}^{\mathrm{gt}}}$. Therefore, we compute the value $\overline{\Delta\rho} = \frac{1}{3}(\Delta\rho_{\mathcal{S}^P} - \Delta\rho_{\mathcal{S}^U} - \Delta\rho_{\mathcal{S}^N})$ to estimate the WSD performance.

Fig. 5 shows that $\overline{\Delta\rho}$ correlates well with WSD task performance ($R^2 = 0.85$). It suggests that if we achieve the key idea more precisely, we may improve the WSD performance. For instance, using a richer lexical relation knowledge, exploitation of the monosemous words, and self-training with confidence thresholding may be promising. We leave it for future work.

8 Conclusion

In this paper, we proposed `SS-WSD`: Semantic Specialization for WSD[14]. The proposed method learns how to adapt BERT embeddings by transformation and uses the semantic relation knowledge as a supervision signal. The key idea is the desired characteristics of similarities: bringing related senses and the contexts closer while unrelated senses and different senses farther away. We realized it as the joint optimization of the Attract-Repel and self-training objectives while preventing large deviations from original embeddings. Experiments showed that the proposed method outperformed the previous embedding adaptation method. When combined with the reranking heuristic that can be applied to stand-alone texts, it established a new SoTA performance on knowledge-based WSD. The proposed method performs well regardless of the availability of global contexts beyond the target sentence during inference, which the previous study did not achieve. Several analyses showed the effectiveness of the objectives and constraints introduced for specialization. We also found that the closeness of semantically related/different/unrelated senses relative to the closeness of correct context-sense pairs positively correlates with the WSD task performance.

9 Future Work

Given that the proposed method only necessitates lexical resources, it has the potential to effectively address the knowledge acquisition bottleneck problem (Pasini, 2020). Thus, we are interested in applying the proposed method to multilingual WSD using multilingual language models as contextualized encoders. One approach is the zero-shot cross-lingual transfer, which involves learning embeddings adaptation using only English lexical resources. Another option is the joint training of all target languages using multilingual lexical resources such as BabelNet (Navigli et al., 2021). We are also interested in integrating the proposed method into supervised WSD and applying the transfer learning of the specialized embeddings to other NLP tasks.

---

[14] The source code is available at: `https://github.com/s-mizuki-nlp/semantic_specialization_for_wsd`

10 Limitations

One limitation of this work is that it is specific to BERT. Although this is in line with the standard practice in previous studies, experimenting with other pre-trained language models is preferred to assess the utility of the proposed method, or to improve the performance further. Another limitation is that it is evaluated on a single dataset and task. While we also followed the de-facto standard protocol, evaluating on rare senses (Maru et al., 2022) or Word-in-Context task (Pilehvar and Camacho-Collados, 2019; Martelli et al., 2021) will bring us more comprehensive insights on the effectiveness and applicability.

11 Ethics Statement

This work does not involve the presentation of a new dataset, nor the utilization of demographic or identity characteristics in formation. In this work, we propose a method for adapting contextualized embeddings for WSD using lexical resources. The proposed method is not limited to a specific resource, we used WordNet as the source of semantic relation knowledge and sense inventory. Therefore, adapted embeddings and sense disambiguation behavior may reflect the incomplete lexical diversity of WordNet in culture, language (Liu et al., 2021), and gender (Hicks et al., 2016).

12 Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 19H01118. We thank Marco Cognetta for his valuable input and for reviewing the manuscript.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631.

Edoardo Barba, Tommaso Pasini, and Roberto Navigli. 2021a. ESC: redesigning WSD with extractive sense comprehension. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4661–4672. Association for Computational Linguistics.

Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2021b. Consec: Word sense disambiguation as continuous sense comprehension. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1492–1503.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864.

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4330–4338.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017.

Niccolò Campolungo, Federico Martelli, Francesco Saina, and Roberto Navigli. 2022. Dibimt: A novel benchmark for measuring word sense disambiguation biases in machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4331–4352.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 1597–1607.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press.

William A. Gale, Kenneth Ward Church, and David Yarowsky. 1992. One sense per discourse. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, USA, February 23-26, 1992*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

John M. Giorgi, Osvald Nitski, Bo Wang, and Gary D. Bader. 2021. Declutr: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, (Volume 1: Long Papers)*, pages 879–895.

Amanda Hicks, Michael W. Rutherford, Christiane Fellbaum, and Jiang Bian. 2016. An analysis of wordnet's coverage of gender identity using twitter and the national transgender discrimination survey. In *Proceedings of the 8th Global WordNet Conference*, pages 123–130.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2nd Edition)*, chapter 20. Prentice-Hall, Inc.

Caterina Lacerra, Michele Bevilacqua, Tommaso Pasini, and Roberto Navigli. 2020. CSI: A coarse sense inventory for 85% word sense disambiguation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020*, pages 8123–8130.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation, SIGDOC 1986*, pages 24–26.

Fangyu Liu, Emanuele Bugliarello, Edoardo Maria Ponti, Siva Reddy, Nigel Collier, and Desmond Elliott. 2021. Visually grounded reasoning across languages and cultures. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10467–10485.

Daniel Loureiro and Alípio Jorge. 2019. Language modelling makes sense: Propagating representations through wordnet for full-coverage word sense disambiguation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5682–5691.

Daniel Loureiro, Kiamehr Rezaee, Mohammad Taher Pilehvar, and José Camacho-Collados. 2021. Analysis and evaluation of language models for word sense disambiguation. *Comput. Linguistics*, 47(2):387–443.

Federico Martelli, Najla Kalach, Gabriele Tola, and Roberto Navigli. 2021. Semeval-2021 task 2: Multilingual and cross-lingual word-in-context disambiguation (mcl-wic). In *Proceedings of the 15th International Workshop on Semantic Evaluation, SemEval@ACL/IJCNLP 2021*, pages 24–36.

Marco Maru, Simone Conia, Michele Bevilacqua, and Roberto Navigli. 2022. Nibbling at the hard core of word sense disambiguation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4724–4737.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross Bunker. 1993. A semantic concordance. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, USA, March 21-24, 1993*.

Nikola Mrkšić, Ivan Vulić, Diarmuid O Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017. Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the association for Computational Linguistics*, 5:309–324.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):10:1–10:69.

Roberto Navigli, Michele Bevilacqua, Simone Conia, Dario Montagnini, and Francesco Cecconi. 2021. Ten years of babelnet: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4559–4567. ijcai.org.

Tommaso Pasini. 2020. The knowledge acquisition bottleneck problem in multilingual word sense disambiguation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 4936–4942.

Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. XL-WSD: an extra-large and cross-lingual evaluation framework for word sense disambiguation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021*, pages 13648–13656.

Mohammad Taher Pilehvar and José Camacho-Collados. 2019. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273.

Alessandro Raganato, José Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 99–110.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B. Viégas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pages 8592–8600.

Ivan Vulic and Nikola Mrksic. 2018. Specialising word vectors for lexical entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1134–1145.

Dong Wang, Ning Ding, Piji Li, and Haitao Zheng. 2021a. CLINE: contrastive learning with semantic negative examples for natural language understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, (Volume 1: Long Papers)*, pages 2332–2342.

Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. 2018. Cosface: Large margin cosine loss for deep face recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274.

Ming Wang and Yinglin Wang. 2020. A synset relation-enhanced framework with a try-again mechanism for word sense disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6229–6240.

Ming Wang and Yinglin Wang. 2021. Word sense disambiguation: Towards interactive context exploitation from both word and sense perspectives. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 5218–5229.

Ming Wang, Jianzhang Zhang, and Yinglin Wang. 2021b. Enhancing the context representation in similarity-based word sense disambiguation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8965–8973.

Yinglin Wang, Ming Wang, and Hamido Fujita. 2020. Word sense disambiguation: A comprehensive knowledge exploitation framework. *Knowledge Based System*, 190:105030.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Zhi Zhong and Hwee Tou Ng. 2012. Word sense disambiguation improves information retrieval. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–282.

A Lexical Resources

Table 6 shows an example of lexical resources for a sense key `computer%1:06:00::`. Note that unrelated senses are randomly chosen in practice.

For *related* senses lookup, we followed Wang and Wang (2020)'s paper and implementation[15]. Briefly, for a given sense key, we collect the synsets that encompass either itself or the sense keys connected by `derivationally_related_forms` relation. Then, for each collected synset, we extend the synsets via semantic relations shown in Table 7. Finally, we collect the sense keys that belong to either one of the synsets in the extended set of synsets, together with those connected to a given sense key by semantic relations shown in Table 7. We used the `nltk.corpus.wordnet` package for implementation.

B BERT Embeddings for Sense

For the computation of sense embeddings, we followed Wang and Wang (2020)'s method. Specifically, for a given sense key, we generate a sentence by filling in the following template using the lemma, synset lemmas, definition, and examples:

```
[lemma] - [syn. lemma 1], ...,
[syn. lemma n] - [definition]
[example 1] ... [example m],
```

where `n` and `m` represent the number of synonym lemmas and the number of examples. Then we take the average over all subwords in a sentence. For example, applying the template to the sense `computer%1:06:00::` will produce the following sentence.

> computer - computer, computing device, data processor, ... - a machine for performing calculations automatically

We solely use the examples available in WordNet Gloss Corpus and do not use the augmented examples that Wang and Wang (2020) collected.

C Hyperparameter Search

For the hyperparameter search, we first jointly optimized on the number of minibatches N_B, relative importance between objectives α (Eq. 9), and constraint on the distance from BERT embeddings ϵ (Eq. 3.5). We used `TPESampler` in the `optuna` package (Akiba et al., 2019) for optimization. We run hyperparameter search over $N_B \in \{64, 128, 256, 512, 1024\}$, $\alpha \in [0.1, 10]$, and $\epsilon \in [0.001, 0.1]$. The number of search trials is 210. Then, we ran a grid search on ϵ over the interval in [0.01,0.02] using a step size of 0.001. During hyperparameter search, we observed that 1) large minibatch size of 256 or above doesn't produce any statistically significant difference and 2) α is much less sensitive compared to ϵ.

D Analysis of Similarity Characteristics

We quantify the similarity characteristic as the macro average of similarity between senses and the similarity of ground-truth context-sense pairs. Specifically, for a given sense s, we calculate the average similarity to its related senses $\mathcal{S}_s^P$, unrelated senses $\mathcal{S}_s^U$, and different senses $\mathcal{S}_s^N$. Following Attract-Repel objective (§ 3.5.1), we define the minibatch excluding itself as the unrelated senses: $\mathcal{S}_s^U = \mathcal{S}^B \setminus \{s\}$. Then, we take the average over all senses $\mathcal{S}$, yielding the similarity among related senses $\rho_{\mathcal{S}^P}$, unrelated senses $\rho_{\mathcal{S}^U}$, and different senses $\rho_{\mathcal{S}^N}$ as follows:

$$\begin{aligned}
\rho_{\mathcal{S}^P} &= \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\frac{1}{|\mathcal{S}_s^P|}\sum_{s'\in\mathcal{S}_s^P}\rho_{s,s'}, \\
\rho_{\mathcal{S}^U} &= \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\frac{1}{|\mathcal{S}_s^U|}\sum_{s'\in\mathcal{S}_s^U}\rho_{s,s'}, \\
\rho_{\mathcal{S}^N} &= \frac{1}{|\mathcal{S}^N|}\sum_{s\in\mathcal{S}^N}\frac{1}{|\mathcal{S}_s^N|}\sum_{s'\in\mathcal{S}_s^N}\rho_{s,s'},
\end{aligned} \tag{15}$$

where $\mathcal{S}^N = \{s; |\mathcal{S}_s^N| > 0\}$.

For the similarity of ground-truth context-sense pairs $\rho_{\mathcal{W}^{\text{gt}}}$, we use the pairs of the word and annotated senses in the evaluation dataset (§ 4.2). For a given word w, we calculate the average similarity to its ground-truth senses $\mathcal{S}_w^{\text{gt}}$. Then, we take the average over all words $\mathcal{W}$ as follows:

$$\rho_{\mathcal{W}^{\text{gt}}} = \frac{1}{|\mathcal{W}|}\sum_{w\in\mathcal{W}}\frac{1}{|\mathcal{S}_w^{\text{gt}}|}\sum_{s\in\mathcal{S}_w^{\text{gt}}}\rho_{w,s}. \tag{16}$$

Finally, we define $\Delta\rho_*$ as the difference to $\rho_{\mathcal{W}^{\text{gt}}}$ for each relation types. We also define $\overline{\Delta\rho}$ as the arithmetic average over them while taking favorable positive/negative directions into account.

$$\begin{aligned}
\Delta\rho_{\mathcal{S}^P} &= \rho_{\mathcal{S}^P} - \rho_{\mathcal{W}^{\text{gt}}} \\
\Delta\rho_{\mathcal{S}^U} &= \rho_{\mathcal{S}^U} - \rho_{\mathcal{W}^{\text{gt}}} \\
\Delta\rho_{\mathcal{S}^N} &= \rho_{\mathcal{S}^N} - \rho_{\mathcal{W}^{\text{gt}}} \\
\overline{\Delta\rho} &= \frac{1}{3}(\Delta\rho_{\mathcal{S}^P} - \Delta\rho_{\mathcal{S}^U} - \Delta\rho_{\mathcal{S}^N})
\end{aligned} \tag{17}$$

---

[15] https://github.com/lwmlyy/SREF

Element	Example
Sense (sense key)	`computer%1:06:00::`
Lemma	*computer*
Synset	`computer.n.01`
Definition sentence	*a machine for performing calculations automatically*
Example	`Not Available`
Synonym lemmas	*computer, computing device, data processor, ...*
Related senses	`computing_device%1:06:00::` (synonym), `analog_computer%1:06:00::` (hyponym), `compute%2:31:00::` (derivative), ...
Different senses	`computer%1:18:00::`
unrelated senses (randomly chosen)	`goldfish%1:05:00::`, `chef%1:18:01::`, ...

Table 6: Example of WordNet lexical resources used in the proposed method.

Category	Relation names
Sense key	`pertainyms, antonyms`
Synset	`hyponyms, hypernyms, part_holonyms, part_meronyms, member_holonyms, member_meronyms, entailments, attributes, similar_tos, causes, substance_holonyms, substance_meronyms, usage_domains, also_sees`

Table 7: WordNet semantic relation names used for collecting related senses.

E Implementation Details

We implemented the transformation functions using PyTorch library[16]. We trained them using single NVIDIA 2080Ti GPU. It took approximately two hours for a single run. We precomputed BERT embeddings for training and evaluation dataset and saved them to temporary files for computation efficiency.

---

[16] https://pytorch.org/

Concept-based Persona Expansion for Improving Diversity of Persona-Grounded Dialogue

Donghyun Kim[1] **Youbin Ahn**[1] **Chanhee Lee**[1] **Wongyu Kim**[1]
Kyong-Ho Lee[1] **Donghoon Shin**[2] **Yeonsoo Lee**[2]
[1]Department of Computer Science, Yonsei University
[2]NCSOFT
{dhkim92,ybahn,heederer,rladnjsrb9999,khlee89}@yonsei.ac.kr
{dhshin,yeonsoo}@ncsoft.com

Abstract

A persona-grounded dialogue model aims to improve the quality of responses to promote user engagement. However, because the given personas are mostly short and limited to only a few informative words, it is challenging to utilize them to generate diverse responses. To tackle this problem, we propose a novel persona expansion framework, **C**oncept-based **P**ersona e**X**pansion (**CPX**). CPX takes the original persona as input and generates expanded personas that contain conceptually rich content. We constitute CPX with two task modules: 1) Concept Extractor and 2) Sentence Generator. To train these modules, we exploit the duality of two tasks with a commonsense dataset consisting of a concept set and the corresponding sentences which contain the given concepts. Extensive experiments on persona expansion and response generation show that our work sufficiently contributes to improving the quality of responses in diversity and richness.

1 Introduction

A persona-grounded dialogue model aims to generate more human-like and engaging responses based on given traits called persona (Zhang et al., 2018a). As efforts of this research line, many recent works have explored various approaches to improving the quality of persona-based responses (Liu et al., 2020; Song et al., 2020; Kim et al., 2020).

In spite of these efforts, there remain some limitations in persona-grounded dialogue models. They are in need of generating more diverse responses. Due to the predefined personas being mostly short and limited to only a few informative words, the responses based on these personas tend to be generic and monotonous. To tackle this issue, COMPAC (Majumder et al., 2020) expands the predefined personas with a commonsense knowledge graph about events, ATOMIC (Sap et al., 2019). With the expanded personas, the dialogue agent generates more diverse responses which contain inferential knowledge stemming from the original persona.

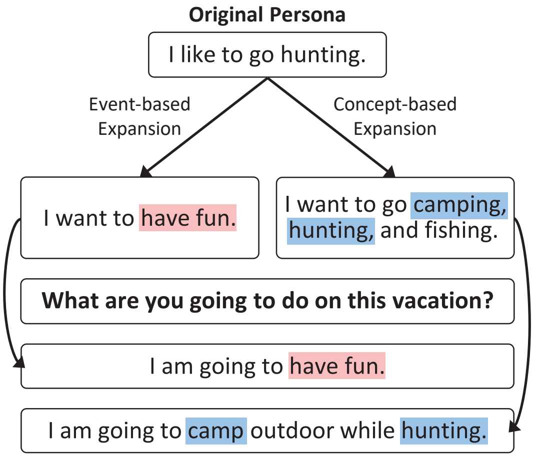


Figure 1: An example of responses generated by different persona expansion strategies for the same query.

However, there is room for improvement in the expansion strategy of COMPAC, which takes an event-based approach as means for persona expansion, which captures causal inferences from personas. The event-based persona expansion is done with the utilization of ATOMIC as a training dataset for the COMET commonsense transformer (Bosselut et al., 2019). As an example, in Figure 1, the expanded persona of the agent (i.e., *I want to have fun*) can be obtained from the given original persona (i.e., *I like to go hunting*) through causal inference. Then given the question "*What are you going to do on this vacation?*", the response based on the event-based expansion strategy is given as "*I am going to have fun*". The generated response is suitable for a given query in terms of relevance. Nevertheless, the response is still monotonic because the expanded persona lacks content.

On the other hand, in the manner of the concept-based expansion strategy, the expanded persona of the agent (i.e., *I want to go camping, hunting, and*

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3471–3481
May 2-6, 2023

fishing) can be obtained through semantic reasoning between concepts (*hunt* → *camping*, *fishing*). Then given the same question, the response based on the expanded persona by concept-based strategy is "*I am going to camp outdoor while hunting*", which is more conceptually diverse and rich.

In general, the diversity of responses is directly correlated with the interlocutor's feeling of being involved in a conversation. Further, in order for the interlocutor to be attracted, a more significant amount of information needs to be delivered. In other words, even in responses that convey the same meaning, the presence of words to enrich the content makes a difference in diversity. Hence, it is crucial for original personas to be expanded to have rich content to improve the diversity of persona-based responses. As demonstrated in Figure 1, by utilizing various concepts in expansion, it is possible to generate personas with richer content than the existing expansion approach.

In this paper, we propose a novel **C**oncept-based **P**ersona e**X**pansion framework, called **CPX**. First, the concept extractor extracts relevant concepts from the original persona sentence, which consists of the constituent words and their semantically related terms. Then, the sentence generator leverages the extracted concepts to generate expanded personas via generative commonsense reasoning. In training, we exploit the duality of CommonGen (Lin et al., 2020) dataset to train the two opposing task modules that consist of the CPX framework: 1) Concept Extractor and 2) Sentence Generator. The experimental results demonstrate that our framework outperforms the baseline models in terms of the expanded persona's diversity. We also show that the persona-grounded dialogue model employing our expansion strategy generate more engaging and diverse responses.

The contributions of our work are as follows:

- We propose a novel concept-based persona expansion framework, CPX, to generate expanded personas with rich content.
- We adopt the duality of concept extraction and sentence generation to constitute the framework for the proposed expansion strategy.
- Through the proposed framework, it is possible to augment the persona-grounded dataset to improve the diversity of responses.

2 Related Work

2.1 Persona-Grounded Dialogue

A persona-grounded dialogue model aims to generate more engaging, human-oriented responses by using some personal characteristics of an agent (Zhang et al., 2018a). As a data-driven approach, Welleck et al. (2019) propose the elaborately constructed dialogue inference dataset, DNLI. Some studies show that fine-tuning the pre-trained language models on the persona-grounded dataset can improve the quality of dialogues (Wolf et al., 2019; Golovanov et al., 2019). Furthermore, recent studies have explored sophisticated neural architectures for better persona-based responses, such as endowing mutual-persona (Liu et al., 2020), multi-stage framework (Song et al., 2020), and self-consciousness modeling (Kim et al., 2020).

Despite the aforementioned efforts, diversity remains limited due to the deficient information in predefined personas. Majumder et al. (2020) adopt the event-based persona expansion strategy for generating more engaging responses. They construct a large number of expanded personas by COMET (Bosselut et al., 2019). Their expansion strategy increases the number of personas, but the expanded personas are still simple in content. That is, while the range of responses that can be generated based on the persona has stretched, the diversity and richness are still limited. To solve this problem, we propose a novel concept-based persona expansion framework that contributes to increasing the semantic richness of expanded personas.

2.2 Generative Commonsense Reasoning

CommonGen task (Lin et al., 2020) aims to generate sentences describing an everyday scenario from a given set of concepts. Formally, the input is a concept set defined as $x = \{c_1, c_2, ..., c_k\} \in X$, and the expected output is a sentence $y \in Y$ that describes a common scenario in our daily life, containing all input concepts. The goal of this task is to learn the reasoning ability between concepts and sentences by injecting relational commonsense knowledge into a language model. We exploit this sentence generation ability in our persona expansion framework (i.e., $f : X \rightarrow Y$). Meanwhile, the extractor model can learn the ability to extract concepts from sentences using an inversely aligned dataset. We leverage the concept extraction ability based on the duality of the tasks in our framework (i.e., $g : Y \rightarrow X$).

Original Concept Set	Human References	Added Concept Set
{cucumber, salad, tomato}	"make a salad of cucumber and tomato" "cooked asparagus and a green salad with cucumbers, tomatoes and chick peas"	{asparagus, green}
{athlete, championship, win}	"olympic athlete has won only medal of the championships" "athlete on his way to winning sport at the championships" "athlete is all smiles after winning the championship after stadium"	{olympic, sport, stadium}
{coast, sea, wave, weather}	"incoming waves of the sea on the coast in foggy weather" "storm clouds in bad weather over rough sea with breaking waves off the coast"	{foggy, storm}

Table 1: Examples of instance pairs (Original Concept Set–Human References) in CommonGen dataset and augmented concept set (Added Concept Set).

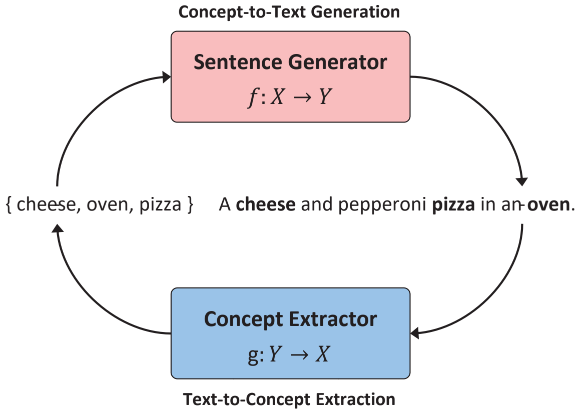


Figure 2: Duality of task modules: 1) Concept Extractor and 2) Sentence Generator. We leverage the CommonGen dataset to train each task module.

3 Task Definition

Our goal is to generate expanded personas to solve the lack of information of a given original persona in PERSONA-CHAT denoted as D_{PC}. Formally, given the original persona p_i, a set of k concepts $C_i = \{c_1, c_2, ..., c_k\}$ is extracted from p_i, where P is a set of all personas in D_{PC}. Then, an expanded persona p_i^+ is generated, using C_i. We perform this expansion for all personas in P, constructing augmented PERSONA-CHAT D_{PC}^+ that includes all expanded personas P^+. Finally, D_{PC}^+ is the augmented dataset that can enhance the dialogue in a data-driven manner.

4 Data Preparation

4.1 Training Data for Framework

As shown in Figure 2, we leverage the duality of two tasks in the CPX framework. The generator and extractor are trained with the original and inversed CommonGen dataset, respectively. The original instance pairs in CommonGen consist of one or more sentences corresponding to each concept set. In other words, it is designed so that a model learns the ability to generate multiple sentences, including given concepts. This dataset configuration can help increase the diversity of the sentences produced by the generator. We also construct the inversely aligned dataset to inject the ability to extract concepts from the sentence into the extractor.

	Train	Dev	Test
# Concept Sets	32,651	993	1,497
- Size = 3	25,020	493	-
- Size = 4	4,240	250	747
- Size = 5	3,391	250	750
# Sentences	67,389	4,018	7,644
- Unique@3	49,459	1,814	-
- Unique@4	8,109	1,135	-
- Unique@5	1,488	1,062	-

Table 2: Statistics of CommonGen dataset. The upper row summarizes the numbers of concept sets by size, and the lower is the number of unique sentences corresponding to concept sets with N elements.

4.2 Concept Augmentation

In order for our expansion strategy to work successfully, the diversity of the extracted concepts and the generated sentences through each task module must be guaranteed. For this reason, we augment a concept set before the training instead of using the dataset as it is. The model of Feng et al. (2021) is referred to in the concept set augmentation, but the details are different. We use an off-the-shelf NLP tool spaCy[1] to extract various words as candidate concepts from the source sentences. Then, we calculate the average BERTScore (Zhang et al., 2019) between the candidate concepts and the source sentences. Finally, 1 to 5 candidate concepts with the highest score are selected as *Added Concept Set*. Some examples of augmentation are illustrated in Table 1. Formally, we denote the original concept set as C^{ori}, and the added concept set as C^{add}. C^+

---

[1] https://spacy.io/, The used version is 3.1.4.

denotes the union of C^{ori} and C^{add}. For example, if C^{ori} and C^{add} aligned with the same sentence are {*building, cloud, sky*} and {*blue, city*} respectively, then C^{+} is {*building, cloud, sky, blue, city*}.

5 CPX Framework

The CPX framework consists of two task modules: 1) **Concept Extractor** – extracting concepts from the original persona; 2) **Sentence Generator** – generate multiple sentences from the given concepts. Inspired by Xia et al. (2017), we utilize the duality of the tasks to train these modules. However, since we just exploit the advantage of the task duality, two modules are independently trained without sharing parameters based on probabilistic duality.

5.1 Concept Extractor

The concept extractor aims to take the concepts that comprise the given sentence. Our model achieves the capability by training on the inverted version of the CommonGen dataset. Formally, given the input sentence $y \in Y$, and the target concept set to extract is a $x = \{c_1, c_2, ..., c_k\} \in X$. The concept extractor aims to find an objective function $g : Y \to X$,

$$g(Y; \theta_{ext}) \triangleq argmax_{x \in X} P(x|Y; \theta_{ext}) \quad (1)$$

where θ_{ext} is a trainable parameter. We leverage BERT (Devlin et al., 2019) based model as the extractor. We train the concept extractor in three ways with C^{ori}, C^{add}, and C^{+} as output X to compare the performance according to the extracted concept set. First, the purpose of the extractor Ext^{ori} trained with C^{ori} is to extract only the concepts contained in a given sentence. Second, the purpose of the extractor Ext^{add} trained with C^{add} is to extract the concepts not contained in a given sentence. By utilizing these hidden concepts, it is possible to generate expanded personas with concepts not included in the original persona but semantically correlated. Finally, the purpose of the extractor Ext^{+} trained with C^{+} is to extract not only the concepts contained in a given sentence but also the ones not contained but semantically related. The experimental results of the effect of each extractor on the persona expansion performance are described in Section 6.2.2.

5.2 Sentence Generator

The goal of the sentence generator is precisely the same as the original CommonGen task, generating sentences by inferring the underlying relational knowledge among concepts. Formally, given the input concept set defined as $x = \{c_1, c_2, ..., c_k\} \in X$, the target is to generate a sentence $y \in Y$ containing all input concepts. The sentence generator aims to find an optimal objective function $f : X \to Y$,

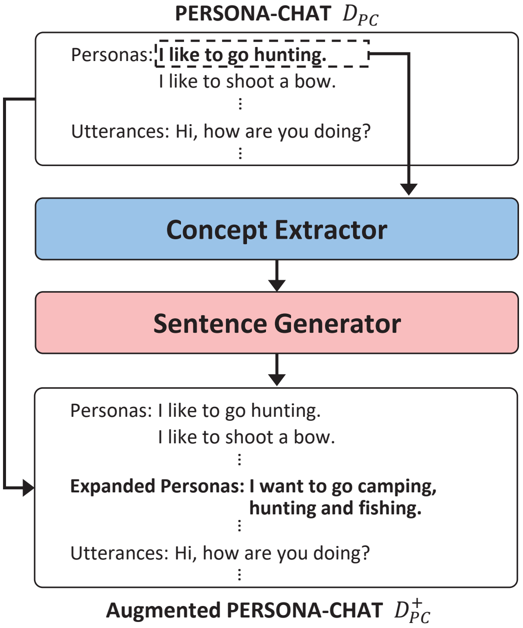


Figure 3: The overall workflow of CPX framework. Given a persona-grounded dialogue dataset D_{PC}, CPX consisting of the extractor and generator generates expanded personas. Final output D^{+}_{PC} is the augmented dialogue dataset.

$$f(X; \theta_{gen}) \triangleq argmax_{y \in Y} P(y|X; \theta_{gen}) \quad (2)$$

where θ_{gen} is a trainable parameter. In our setting, we exploit the pre-trained models as the generator, specifically BART (Lewis et al., 2020) and T5 (Raffel et al., 2020), which are the state-of-the-art sequence-to-sequence transformer (Vaswani et al., 2017) language models. As reported by Feng et al. (2021), the use of augmented concept sets at the training phase improves the length and richness of generated sentences. Therefore, we use the integrated concept set C^{+} as input X to train the sentence generator.

5.3 Control of Persona Expansion

To constrain the form of expanded personas, we use four types of prompts: *I want*, *I need*, *I feel*, and *I am*.[2] These prompts can control the generation form of the expanded persona and represent an

---

[2] These four types of prompts were determined by referring to Majumder et al. (2020) and the verb distribution of personas in PERSONA-CHAT.

agent's desire, intent, emotion, and status, respectively. Also, according to our in-depth analysis of PERSONA-CHAT, most of the personas describe the first-person subject, so we fix the subject of the prompt as "I". Accordingly, the constrained decoder output sequence for generating the expanded persona is as follows:

$$[I]\ [want|need|feel|am] \tag{3}$$

Consequently, we obtain four expanded personas per original persona. The expanded persona set P^+ is added to the original dataset D_{PC} to construct the augmented dataset D^+_{PC}. The experimental results on response generation using D^+_{PC} will be described in Section 6.3.3.

6 Experiments

To validate the performance of CPX, we conducted two experiments: 1) persona expansion and 2) response generation.

6.1 Experimental Setup

6.1.1 Dataset

We carried out our experiments on the PERSONA-CHAT, converted to the ConvAI2 benchmark version (Dinan et al., 2020). The dataset consists of 17,878/1,000 multi-turn dialogues and 1,155/100 profiles for Train/Valid set. We utilized the unique personas extracted from all profiles for persona expansion experiments. Also, we utilized the original and augmented PERSONA-CHAT datasets to demonstrate the impact of the expansion strategy on response generation.

6.1.2 Baselines

Two types of expansion baselines are considered: 1) **paraphrasing** and 2) **transformer** trained with commonsense knowledge.

- **MANUAL PARAPHRASING** (Zhang et al., 2018a): We used manually paraphrased personas provided with the original PERSONA-CHAT, where workers rephrased the original personas to remove trivial word overlaps.
- **AUTOMATIC PARAPHRASING** (Xie et al., 2020): To paraphrase the personas in an automated manner, we leveraged the existing paraphrasing system based on back-translation. We generated the paraphrased persona by exploiting the pre-trained En-Fr and Fr-En translation models.[3]
- **COMET** (Bosselut et al., 2019): COMET is a transformer-based model that generates commonsense expansions of a given world event by training on a commonsense knowledge graph such as ATOMIC. We generated expanded personas for four relations (i.e., xWant, xNeed, xReact, xAttr) that are suitable for representing an agent's traits.

In our setting, the automatic paraphrasing model generated one expanded persona for each original persona, and the transformer-based model generated four expanded personas, one for each relation.

6.2 Persona Expansion

In this section, we compared the quality of personas expanded by CPX and baselines. We further reported CPX's performance according to the concept extractor and sentence generator.

6.2.1 Evaluation Metrics

Automatic Evaluation To evaluate the quality of expanded personas, we employed four metrics. (1) Distinct-n (**Dist-n**) (Li et al., 2016) measures the diversity of sentences by calculating the ratio of distinct words against total n-grams. (2) Entropy-n (**Ent-n**) (Zhang et al., 2018b) measures the entropy obtained via the n-gram distribution in a sentence. (3) Also, we report the length of the sentence (**Length**) to measure the amount of content conveyed by the persona. (4) Finally, we employ the BERTScore (**F$_{\text{BERT}}$**) (Zhang et al., 2019) to measure how semantically relevant the expanded persona is to the original persona.

Human Evaluation We conducted a human evaluation with 100 random samples. We hired three human annotators proficient in domain language through a third-party company. Annotators knew nothing of the system they were evaluating. Also, annotators were properly compensated for their labor. Human annotators evaluated the quality of expanded persona sentences on three criteria. The metrics used for human evaluation are as follows: (1) **Engagement** measures whether the expanded persona is engaging or interesting. (2) **Diversity** measures whether the expanded persona is diverse and informative. (3) **Relevance** measures whether the expanded persona is semantically relevant to the original persona. The scoring range from 1 to 5, with lower indicating poor and higher indicating better. Further, we conducted a pairwise comparison between personas expanded by CPX and personas expanded by COMET.

---

[3] https://github.com/google-research/uda

Expansion Model	Automatic Evaluation						Human Evaluation		
	Dist-1	Dist-2	Ent-1	Ent-2	Length	F_{BERT}	Engagement	Diversity	Relevance
Original	0.08	0.34	5.54	8.63	7.57	-	-	-	-
Manual Paraphrasing	0.16	0.48	6.02	8.47	6.49	-	-	-	-
Automatic Paraphrasing	0.07	0.39	5.79	8.54	7.64	**0.90**	3.08	2.78	**3.98**
COMET	0.04	0.16	4.43	5.92	6.17	0.81	2.76	2.43	3.04
CPX_{BART}	0.19	**0.53**	6.06	8.94	11.57	0.85	4.08	4.14	3.85
CPX_{T5}	**0.20**	0.51	**6.17**	**9.01**	**12.07**	0.86	**4.12**	**4.22**	3.66

Table 3: Automatic and human evaluation results on persona expansion. The extractor used for CPX is Ext^+ trained with C^+. The best results are bolded, and the second-best are underlined.

CPX vs. COMET	Desire (I want)			Intent (I need)			Emotion (I feel)			Status (I am)			Average	
Metrics	win	loss	*k*	win	loss	*k*	win	loss	*k*	win	loss	*k*	win	loss
Engagement	**87.7**	12.3	0.61	**85.7**	14.3	0.66	**91.2**	5.7	0.68	**82.3**	17.7	0.63	**86.4**	13.0
Diversity	**90.3**	9.7	0.72	**89.0**	5.7	0.68	**92.7**	4.3	0.70	**94.3**	4.2	0.69	**91.6**	6.0
Relevance	**52.1**	47.4	0.52	**62.7**	33.3	0.54	43.5	**56.5**	0.48	49.2	**50.8**	0.45	**51.9**	47.0

Table 4: Pairwise comparison results between personas per each relation (prompt) expanded by CPX_{T5} vs. COMET. All numbers are in percentages, and ties are not indicated in the table. The values of Fleiss' kappa *k* (Fleiss, 1971) for all results are in $0.4 < k < 0.8$, indicating moderate agreement among the annotators.

Extractor	Generator	Engagement	Diversity	Relevance
Ext^{ori}	BART	3.55	3.72	3.66
Ext^{add}		3.16	3.24	2.98
Ext^{+}		**4.08**	**4.14**	**3.85**
Ext^{ori}	T5	3.68	3.85	**3.70**
Ext^{add}		3.20	3.32	2.86
Ext^{+}		**4.12**	**4.22**	3.66

Table 5: Human evaluation results on impact of concept extractor. The best results are bolded.

6.2.2 Persona Expansion Results

Analysis on Automatic Evaluation Automatic evaluation results are reported on the left of Table 3. CPX outperforms the baselines in all automatic evaluation metrics in diversity. The personas expanded by CPX were not only long (Length) but also rich in content, being composed of different and unique words (Dist-1/2, Ent-1/2). Also, CPX showed a diversity similar to or better than manual and automatic paraphrasing. In particular, we demonstrated the effectiveness of CPX's concept-based approach by surpassing immensely on all metrics of the comparative model COMET. For the BERTScore, which indicates relevance to the original persona, CPX performed comparatively lower than automatic paraphrasing, i.e., machine translation. This is because the back-translated personas have almost the same meaning as the original persona. On the other hand, CPX scored higher than the comparative generative-based model COMET. Judging from these results, the concepts extracted by CPX were semantically related to the original persona, and the generator effectively generated the expanded persona with sufficient commonsense.

Analysis on Human Evaluation Human evaluation results along the three criteria are depicted on the right of Table 3. The annotators evaluated that CPX outperformed all baselines in terms of Engagement and Diversity except for Relevance, for which CPX made the second-best result but was still close to the machine translation. For accurate evaluation with the target model COMET, we pairwisely compared the personas that each model expanded for four relations, as shown in Table 4. First, CPX was superior in all metrics in the evaluation of the persona forms of "*I want*" and "*I need*", which represent the agent's desire and intent. More specifically, CPX was overwhelmingly superior in Engagement and Diversity. This is because the personas generated by CPX contain richer content than COMET. This means that CPX learned the commonsense relationship between concepts well and used it effectively to generate expanded personas. On the other hand, in terms of Relevance to the original persona, CPX showed lower performance in the remaining two persona forms of "*I feel*" and "*I am*", which represent the agent's emotion and status. For this result, we analyzed that it is because the too-long sentences generated by CPX are more likely to contain concepts unrelated to the original persona.

Expansion Model	Automatic Evaluation					Human Evaluation		
	PPL	BLEU	Dist-1	Dist-2	F_{BERT}	Fluency	Engagement	Diversity
Original	20.24	1.28	0.04	0.19	0.09	3.12	2.44	2.32
Manual Paraphrasing	19.76	1.43	0.13	0.24	0.12	3.20	3.02	3.17
Automatic Paraphrasing	20.38	1.51	0.11	0.27	0.11	3.13	2.78	2.89
COMET	19.87	2.66	0.18	0.31	0.13	3.24	3.05	3.11
CPX_{BART}	19.68	3.18	0.32	0.79	0.14	**3.30**	3.67	3.48
CPX_{T5}	**19.44**	**3.27**	**0.34**	**0.82**	**0.16**	3.28	**3.85**	**3.70**

Table 6: Automatic and human evaluation results on response generation utilizing expanded personas by each model. The generative model used in response generation experiment is GPT-2 (Wolf et al., 2019). The best results are bolded, and the second-best are underlined.

Original Persona: I like to remodel homes.
Extracted Concepts: remodel, home, country, house, repair
PARAPHRASING: • MANUAL: I love to redesign houses. • AUTOMATIC: I like to renovate houses.
COMET: • xWant: I want to buy a new home. • xNeed: I need to buy a house. • xReact: I feel happy. • xAttr: I am a homeowner.
CPX: • Desire: I want to live in a country house with a large yard. • Intent: I need to a hammer and a saw to repair my old house. • Emotion: I feel comfortable in country home. • Status: I am busy repairing solar panels on the roof of home.

Table 7: Examples of expanded personas.

Impact of Concept Extactor We analyzed the impacts of the extracted concept on the CPX framework. We expanded the persona and conducted a human evaluation using the extractors Ext^{ori}, Ext^{add}, and Ext^{+} learned with the three concept sets of C^{ori}, C^{add}, and C^{+}, respectively. As reported in Table 5, the overall performance was the best when the extractor trained with C^{+} was used. In particular, when the Ext^{add} extractor trained using only C^{add}, it showed a deficient value in Relevance. It shows that not only the diversity of the extracted concepts from the sentence but also the relevance to the original persona is essential to concept-based persona expansion.

6.2.3 Case Study

Table 7 shows an example of personas expanded by CPX and other baseline models. The paraphrased personas were semantically identical to the original persona. COMET generated expanded personas that can be inferred from the original persona. However, the expanded personas were still short and lacking in content. On the other hand, CPX generated expanded personas with rich content, including extracted concepts.

6.3 Response Generation

We conducted an experiment on response generation using the original and augmented dataset expanded by different strategies.

6.3.1 Dialogue Model

The dialogue model used in our response generation experiment is GPT-2 (Wolf et al., 2019) just concatenating all persona sentences along with dialog history. In the case of paraphrasing, the training dataset was constructed by concatenating all personas. On the other hand, in the case of COMET and CPX, the number of expanded personas is large. Therefore, training datasets were constructed using only the expanded personas with the highest score, each per original persona, by utilizing the RoBERTa (Liu et al., 2019) based NLI model pretrained with the DNLI dataset.

6.3.2 Evaluation Metrics

We automatically evaluated the response generation using the metrics used in the persona expansion experiment. We also adopted widely used metrics **PPL** and **BLEU** (Papineni et al., 2002) to measure the quality of responses. **Fluency**, which measures whether the generated responses are fluent, was additionally used for human evaluation.

6.3.3 Response Generation Results

Analysis on Automatic Evaluation Automatic evaluation results of response generation are reported on the left of Table 6. The dialogue model trained on the dataset augmented by CPX outperforms other comparative models in all evaluation metrics. In particular, it significantly outperformed other models in terms of diversity. It shows that our expansion strategy improves the diversity of

Personas: I have two dogs. I like to work on vintage cars. My favorite music is country.
Query: What do you usually do in your spare time?
Original: I just relax with my dogs.
Manual: I love vintage car. How about you?
Automatic: I like country music.
COMET: I want to take care of my dogs.
CPX: I like to play frisbees with my dog in a nearby park.

Table 8: Examples of generated responses.

responses based on the language understanding ability of the pre-trained language model. Since pre-trained language models like GPT-2 are trained with large data, it is important to be provided with a rich source (i.e., expanded personas) to generate responses that include various concepts rather than a problem of lack of fine-tuning data.

Analysis on Human Evaluation Human evaluation results are shown on the right of Table 6. A dialogue dataset augmented through CPX significantly improved the performance of the dialogue model for all human evaluation metrics. On the other hand, the dataset augmented by COMET did not achieve significant performance improvement compared to other models. The difference in performance between the two expansion strategies was particularly large in Engagement and Diversity. It means that the concept-based expansion strategy is more suitable for improving the diversity and richness of the persona-based responses.

6.3.4 Case Study

Table 8 depicts examples of responses generated by models trained according to paraphrasing or expansion strategies. The responses generated by CPX were the richest in content. The paraphrasing-based models responded just at the level of simply copying personas. COMET generated a response containing "*take care*" that could be inferred from "*I have two dogs.*" based on its causal reasoning ability. Nevertheless, it did not significantly improve the diversity of the generated responses. On the other hand, CPX utilized the concepts such as "*frisbee*" and "*park*" that are commonly related to the original persona "*I have two dogs.*" to generate a richer utterance. We found that improving the conceptual diversity of personas enriched persona-based responses and made them more engaging.

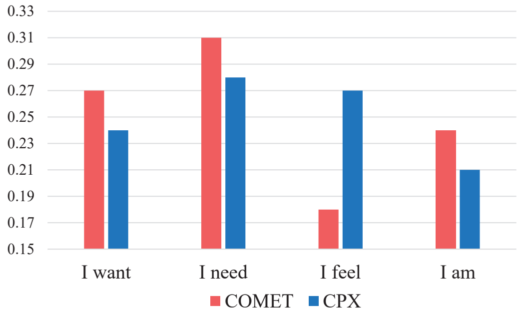


Figure 4: Distribution of each relation (prompt) of the expanded persona selected by the NLI model.

7 Discussions

Biases in Persona Selection While constructing the dataset for model training, we identified the bias in selecting the expanded personas. As shown in Figure 4, among the four types of personas expanded by COMET, the ratio of each type selected through the NLI model is uneven. While the personas expanded by CPX had been selected more evenly, the distribution was still somewhat biased. This suggests that it is likely for persona expansion models to generate more biased responses through poor selection. It also means that the approach that leverages NLI-based similarities between expanded personas and predefined personas, queries, or gold responses, in the persona selection can also introduce some bias. In order to generate a richer and unbiased persona-based response, it is necessary to study a method of considering the fairness of this expanded persona selection process.

Dual Use of Dataset In this study, we utilized CommonGen dataset for the generative commonsense reasoning task to train the proposed framework. This dual use of the dataset causes some concerns that need to be discussed. First, indiscriminate use of commonsense can lead to the problem of deceiving users or disclosing information by making it difficult for users to recognize that their conversation partner is a chatbot (Gros et al., 2021). Next, the CommonGen dataset may contain biases because it was constructed by human crowd workers. These biases may cause unintended "safety" problems where the dialogue model generates aggressive and harmful responses (Dinan et al., 2022). Fortunately, we found no such case while checking extended personas and generated responses in the CPX framework. Nevertheless, these issues deserve careful consideration in future works leveraging a similar dual use approach.

8 Conclusions and Future Work

In this paper, we proposed the concept-based persona expansion framework to improve the semantic diversity of the persona-grounded dialogue. We expanded personas by extracting and utilizing semantically related concepts through concept-based commonsense reasoning, making persona-grounded dialogues more engaging. During the experiments, we identified some less relevant expansion cases. Therefore, we have a plan to develop a method of preventing potential overexpansion. We also found that in order to effectively reflect the diversely expanded personas in response generation, it was necessary to resolve the bias in the selection process. From this perspective, we will conduct experiments and studies for models that can generate more effective persona-based responses through a bias-free selection process while maintaining the diversity of expanded personas.

Limitations

In this study, we tried to show the importance of the diversity of expanded personas in order to generate more engaging responses. Therefore, the narratives in the paper and the results reported in the tables were focused on diversity. Due to the nature of the proposed concept-based expansion strategy, the expanded personas are generated by using similar but not different concepts, which are not included in the original persona. Also, we intended that the expanded personas be distinct from the original persona as much as possible for the diversity that can be gained from various concepts. For these reasons, we decided to decrease relevance slightly as a trade-off while increasing diversity.

Ethical Considerations

In this study, we utilized the CommonGen and PERSONA-CHAT datasets which contain crowd-sourced work by human annotators. Although we did not find any notable cases during this study, the dual use of these datasets may produce results that contain unintended linguistic and cultural biases.

Acknowledgements

We thank all anonymous reviewers for their valuable comments on this work. This work was supported by NCSOFT NLP Center. Kyong-Ho Lee is the corresponding author.

References

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

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

Emily Dinan, Gavin Abercrombie, A. Bergman, Shannon Spruit, Dirk Hovy, Y-Lan Boureau, and Verena Rieser. 2022. SafetyKit: First aid for measuring safety in open-domain conversational systems. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4113–4133, Dublin, Ireland. Association for Computational Linguistics.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition: From Machine Learning to Intelligent Conversations*, pages 187–208. Springer.

Steven Y. Feng, Jessica Huynh, Chaitanya Prasad Narisetty, Eduard Hovy, and Varun Gangal. 2021. SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Sergey Golovanov, Rauf Kurbanov, Sergey Nikolenko, Kyryl Truskovskyi, Alexander Tselousov, and Thomas Wolf. 2019. Large-scale transfer learning for natural language generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6053–6058, Florence, Italy. Association for Computational Linguistics.

David Gros, Yu Li, and Zhou Yu. 2021. The R-U-a-robot dataset: Helping avoid chatbot deception by detecting user questions about human or non-human identity. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*,

pages 6999–7013, Online. Association for Computational Linguistics.

Hyunwoo Kim, Byeongchang Kim, and Gunhee Kim. 2020. Will I sound like me? improving persona consistency in dialogues through pragmatic self-consciousness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 904–916, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. 2020. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9194–9206, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Haoyu Song, Yan Wang, Wei-Nan Zhang, Xiaojiang Liu, and Ting Liu. 2020. Generate, delete and rewrite: A three-stage framework for improving persona consistency of dialogue generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5821–5831, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Yingce Xia, Tao Qin, Wei Chen, Jiang Bian, Nenghai Yu, and Tie-Yan Liu. 2017. Dual supervised learning. In *International conference on machine learning*, pages 3789–3798. PMLR.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018a. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018b. Generating informative and diverse conversational responses via adversarial information maximization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 1815–1825.

A Implementation Details

A.1 Persona Expansion

CPX was implemented with HuggingFace's Transformers library.[4] The Concept Extractor initialized from the publicly available BERT-based-uncased[5] model with 12 layers and 768 hidden sizes. We use an Adam optimizer (Kingma and Ba, 2014) and a learning rate of 3e-5. The training of the Concept Extractor was conducted on an Nvidia RTX3090 24G GPU with a batch size of 16. The Sentence Generator initialized from the BART-base[6] and T5-base[7] model. We use an Adam optimizer, and the learning rates are 3e-6 and 5e-6, respectively. The Sentence Generator was trained on an Nvidia RTX3090 24G GPU with a batch size of 8.

A.2 Response Generation

We utilized the repositories and implementation details of GPT-2[8] for response generation. We adjusted some of the details of the model and trained in a single-turn dialogue setting.

B Human Evaluation Protocol

We hired three well-educated annotators from a third-party company to conduct human evaluations. The annotators were given the original persona and the expanded persona pairs to evaluate persona expansion. Also, annotators were given the original personas, the query, and the generated response pairs to evaluate response generation. Each annotator evaluated 100 samples, each sample worth $0.1. The evaluation was conducted in a double-blind fashion. The human evaluation metrics include: (1) Engagement, which measures whether the expanded persona or generated response is engaging and interesting; (2) Diversity, which measures whether the expanded persona or generated response is diverse (3) Fluency, which measures whether the generated response is fluent; (4) Relevance, which measures whether the expanded persona is semantically relevant to the original persona. The scoring range of the metrics is 1 to 5. The specific scoring criteria for human annotation are shown in Table 9.

Engagement
1-2: (very) Simple and meaningless
3: Semantically moderate
4-5: (very) Interesting and want to keep the conversation
Diversity
1-2: (very) Generic and short in length
3: Conceptually moderate
4-5: (very) Informative and contain various concepts
Fluency
1-2: (very) Hard to read or syntactically incorrect
3: Grammatically correct
4-5: (very) Fluent and easy to understand
Relevance
1-2: (very) Unsuitable for given query or persona
3: Relevant to given query or persona
4-5: (very) Suitable and reflect well given query or persona

Table 9: Scoring criteria of human evaluation.

---

[4] https://github.com/huggingface/transformers

[5] https://huggingface.co/bert-base-uncased

[6] https://huggingface.co/facebook/bart-base

[7] https://huggingface.co/t5-base

[8] https://github.com/huggingface/transfer-learning-conv-ai

RPTCS: A Reinforced Persona-aware Topic-guiding Conversational System

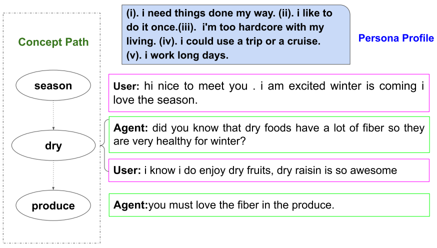
Zishan Ahmad[†§], **Kshitij Mishra**[†§], **Asif Ekbal**[†] and **Pushpak Bhattacharyya**[‡]
[†]Department of Computer Science and Engineering,
Indian Institute of Technology Patna, India
`{1821cs18, kshitij_1921cs23, asif}@iitp.ac.in`
[‡]Department of Computer Science and Engineering,
Indian Institute of Technology Bombay, India
`pb@cse.iitb.ac.in`

Abstract

Although there has been a plethora of work on open-domain conversational systems, most of these lack the mechanism of controlling the concept transitions in a dialogue. For activities like switching from casual chit-chat to task-oriented conversation, an agent with the ability to manage the flow of concepts in a conversation might be helpful. The user would find the dialogue more fascinating and engaging and be more receptive to such transitions if these concept transitions were made while taking into account the user's persona. Focusing on persona-aware concept transitions, we propose a Reinforced Persona-aware Topic-guiding Conversational System (**RPTCS**). Due to the lack of a persona-aware topic transition dataset, we propose a novel conversation dataset creation mechanism in which the conversational agent leads the discourse to drift to a set of target concepts depending on the persona of the speaker and the context of the conversation. To avoid scarcely available expensive human resources, the entire data-creation process is mostly automatic with human-in-loop only for quality checks. This created conversational dataset named **PTCD** is used to develop the **RPTCS** in two steps. First, a maximum likelihood estimation loss-based dialogue model is trained on **PTCD**. The trained model is then fine-tuned in a Reinforcement Learning (RL) framework by employing novel reward functions to assure persona, topic, and context consistency with non-repetitiveness in generated responses. Our experimental results demonstrate the strength of the proposed system with respect to strong baselines[1].

1 Introduction

Due to the abundance of conversational corpora, there has been a great interest in building open-domain conversational systems[2] (Huang et al., 2020). While capable of producing fluent responses, these systems also frequently generate generic responses devoid of any useful information (Gao et al., 2019). Human discourse is complex and difficult to replicate as it often covers a wide range of topics (Winograd, 1977). Hence, the key to a multi-turn conversation's success is striking a balance between effectively changing the subject and keeping it on-topic (See et al., 2019).



(i). i need things done my way. (ii). i like to do it once.(iii). i'm too hardcore with my living. (iv). i could use a trip or a cruise. (v). i work long days.

User: hi nice to meet you . i am excited winter is coming i love the season.

Agent: did you know that dry foods have a lot of fiber so they are very healthy for winter?

User: i know i do enjoy dry fruits, dry raisin is so awesome

Agent:you must love the fiber in the produce.

Figure 1: An illustration of the suggested task. The persona profile and the concept path to the target concept serve as the foundation for the created dialogues.

It is also seen that even if open-domain conversational systems employ topic shift, they are not able to engage user due to absence of knowledge of user's own perceptions. Therefore, to achieve a robust topic shift guiding conversational system, it has to evolve away from the prior factual information systems (Leuski et al., 2006) towards a novel blend of both task-oriented and non-task-oriented systems (Akasaki and Kaji, 2017). However, they may fail to engage user in difficult multi-turn information-seeking tasks or dialogues (Trippas et al., 2020) and may also suffer from "anomalous state of knowledge" (Belkin and Vickery, 1985) where the user has vague information requirements and is often struggling to articulate it with enough precision in a conversation. Thus, we require context-sensitive user guidance that does not assume a rigid hierarchy of the user's plans and

---

[§]equal contribution

[1] Codes and dataset available at `https://github.com/zishan-ahmad-nlp/persona-topic-shift`

[2] we use the terms conversational system/agent, dialogue system/agent, agent, and chatbot interchangeably.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3482–3494
May 2-6, 2023

work objectives. Such a guidance, can be achieved by utilising persona information of user with the ongoing dialogue. This will drive the conversation from one topic to a different one in consonance with the user's interests.

Focusing on these two aspects, we propose a reinforced persona aware topic guiding conversational system (**RPTCS**). Due to scarcity and expensiveness of human resource, first, employing minimal manual intervention, we present a novel persona aware topic guiding conversational dataset (**PTCD**) created employing a novel automatic conversation creation mechanism. Majority of the topic shift guiding conversational systems leverages only the concept space knowledge to represent the potential conversation flow (Wu et al., 2019; Zhou et al., 2020; Zhang et al., 2020; Wang et al., 2021; Qiu et al., 2022). These approaches are predicated on a restricted defined logical premise: individuals would be interested in talking about any random concept. This presumption can be too straightforward to mimic topic flows in real human conversations. In this work, we take a significant step toward developing a proactive conversational agent that is more attuned to a user's personality with a clear conversation aim. We aim at training a system that is able to decide on topic shifts and generate responses that reflect these logical topic shifts, while also considering the user persona. Figure 1 shows an example of this task. Since bringing an open-domain conversation to a task-oriented domain is one of the key use cases for this system, we chose the concepts popular in the latter domain (Eg. restaurant, travel, clothing, smartphone, etc.).

Using **PTCD**, we first train a maximum likelihood estimation (MLE) loss-based conversational employing a language model (LM) GPT-2 small (Radford et al., 2019). This trained model is then fine-tuned in a Reinforcement Learning (RL) framework using novel rewards designed to assure persona, concept, and context consistency with non-repetitiveness in the generated responses. The key contributions of this work are *four-fold* **(i).** Proposal of a new task to build a persona-aware conversation system for targeted topic transitions and creation of a novel datatset **PTCD** to tackle the task in-hand; **(ii).** A novel semi-automatic persona aware topic guiding corpus creation method using pre-trained models (can be adapted to the other similar tasks in the broad area of dialogue systems), **(iii).** Designing a novel reward function to built the proposed system **RPTCS** in an RL framework. **(iv).** Performed extensive automatic and human evaluation by designing two novel evaluation metrics to evaluate topic and context consistency to demonstrate strength of the proposed system.

2 Related Work

To have a human-like conversation and avoid off-topic response generation, several methods have been explored (Ghazvininejad et al., 2018; Harrison et al., 2020). The research community has employed ConceptNet (Speer et al., 2017) to mimick concept transitions in human conversations, (Zhou et al., 2018), such as: Zhang et al. (2020) used concept relations with 2-hop nodes, to account more thoroughly human concept shifts. However, none of the works have taken into account the persona of the user to guide the topic shift in the conversation which may act as a crucial component to have effective communication.

Some benchmark conversation datasets has been proposed to assess the conversation focusing on different personal attributes such as: (He et al., 2017) predicts the next topic based on social and cultural situation, ETHICS dataset constitutes the conversation based on the concept of morality (Hendrycks et al., 2020). Hsu et al. (2018) presents emotion labelled EMOTIONLINES dataset, Yu et al. (2020) proposes a social relation inference based DIALOGRE dialogue dataset. Zhang et al. (2018) proposed PERSONA-CHAT dataset to make chit-chat dialogues more engaging by conditioning them on user's profile information. Most of these datasets are collected through crowdsource workers. To obtain such a large human resource is expensive, time-consuming, and can be infeasible. We investigate an automatic data creation approach requiring minimal manual intervention as an alternative to this.

Recently, Wang et al. (2021) proposed a multi-turn topic-driven NaturalConv dataset ensuring a smooth topic shift in conversations. Our work here is different from the existing topic-shift guiding conversational systems in three aspects. First, we followed an automatic dialogue data curation process. Second, the topic shift in the conversations is guided using ConceptNet as well as the persona of the user. This ensures both goals of an effective communicator *viz.* completing the task and maintaining the face of the user. Third, we built a persona aware topic guiding conversational system in an RL framework using novel efficient and effective reward function. To the best of our knowledge, our proposed setup is novel and has not been tried before.

3 Dataset Creation

To tackle the persona-aware concept shift to a target concept, we create and propose a novel dataset. We come up with a novel semi-automatic data creation technique involving prompting a GPT-J model and human intervention for quality control. The entire dataset creation consists of the following steps:

Obtaining Seed Data: To start the few-shot dialogue generation, we required a seed utterance from a user with some assigned persona. To build In PERSONA-CHAT-grounded topic-shifting conversations“the first utterance of a conversation (focused on a persona-profile) in the PERSONA-CHAT (Zhang et al., 2018) is selected. In PERSONA-CHA", persona profiles are defined as “profiles that are natural and descriptive, and contain typical topics of human interest that the speaker can bring up in conversation". The concepts in the seed utterance is obtained by extracting words with *‘nouns’*, *‘adjectives’*, and *‘verbs’* Part-of-Speech (PoS) tags. We call these first utterance concepts as the source concept. The objective here is to guide the conversation away from the source concept and toward a target concept. Concepts like *‘travel’*, *‘restaurant’*, *‘shopping’*, *‘electronics’*, etc. are chosen as the system’s target concepts.

Concept Path Creation and Selection: To have smooth and logical transition and avoid abrupt source-to-target concept jump, we make use of Concept-Net (Speer et al., 2017). First, employing Dijkstra’s algorithm, the shortest distance between all the source-concepts in the seed dialogue and all the target-concepts is calculated. This results in multiple paths between each source and target concepts. From these multiple paths only one path has to be selected for the desired concept transition in dialogues, i.e., based on the ongoing topic and persona, the conversation must be able to switch to the most suitable target and path. Although it would be ideal for the conversation to veer toward a subject (concept) relevant to the user persona, the context of the conversation may not permit it. Hence, a balance is aimed between context and persona relevance. We extend the use of RoBERTa (Liu et al., 2019) for contextual concept selection as proposed by Yasunaga et al. (2021) to concept-path selection. Let $P = \{p_1, p_2, ..., p_N\}$ be the persona profiles of a user where $p_i = \{w_1, w_2, ..., w_m\}$ is a persona sequence. Each concept-path t to the target is denoted by $C_t = \{c_1, c_2, ..., c_t\}$, where c_j is a concept in the path. For each pair of persona sequence and concept-path, persona relevance probability is computed using the RoBERTa head (LM_{head}). The probabilities are then averaged to obtain $P_{persona}$ as shown in Equation 1 and 2.

$$P_{seqp} = LM_{head}(LM_{enc}([p_i : C_k])) \quad (1)$$

$$P_{persona} = Avg(P_{seqp}[c_{1k}, c_{2k}, ..., c_{tk}]) \quad (2)$$

Here, [:] is the concatenation operation. Using the same RoBERTa LM_{head}, the path probability P_{conv} with respect to the conversational context for a conversation $D = \{U_1, U_2, ..., U_n\}$ is computed (U_i are utterances in the dialogue). To obtain P_{conv}, P_i in encoder LM_{enc} are replaced with D_p (Equation 3 and averaging the probabilities of the concepts (Equation 2).

$$P_{seqc} = LM_{head}(LM_{enc}([D_p : C_k])) \quad (3)$$

$$P_{conv} = Avg(P_{seqc}[c_{1k}, c_{2k}, ..., c_{tk}]) \quad (4)$$

From the obtained persona $P_{persona}$ and contextual P_{conv} concept-path probabilities, the path with the highest probability is selected.

Utterance Generation: After selecting seed utterances, new utterances are generated utilizing GPT-J (Wang and Komatsuzaki, 2021) (Brown et al., 2020). Taking advantage of few-shot prompting capability of the GPT-J model, few manually written sample conversations following the appropriate persona and context relevant topic transitions are used. [3]. This prompt is followed by the current seed dialogue and selected path and given as input to GPT-J, which generates the next utterance of the dialogue. To lead the conversation, all concept transitions are initiated by the agent. Top_k sampling (Fan et al., 2019; Radford et al., 2019) is used during utterance generation and 10 candidate responses for each input utterance is generated.

Utterance Selection: Best response out of the 10 candidates is selected by taking into account persona-entailment, current concept relevance, and conversation context relevance. To assess the level of persona-entailment in the produced outputs, a Natural Language Inference (NLI) model using the Dialog NLI dataset is trained (Welleck et al.,

---

[3] Example of the designed prompt is given in Section A.2 of the appendix

	Total	Train	Test	Valid
#Dialogues	2,586	1,945	406	236
#Utternaces	13,746	10,310	2,436	1,000
#Unique Concepts	1,843	1,505	617	314
#Unique Paths	1,738	1,372	316	184
Avg Path Length	3.55	3.54	3.57	3.57

Table 1: Statistics of the dataset created

2019). by fine-tuning a pre-trained BERT (Devlin et al., 2018) model[4]. The probability score of entailment $P_{ent}^{p_i}$ of a given utterance U_k with respect to a persona p_i are obtained by final *softmax* layer. Averaging the $P_{ent}^{p_i}$ gives the final entailment score $P_{ent}^{U_k}$ for a candidate utterance U_k. This entailment score is calculated for both the user and agent utterances to counteract the contradiction to a user's preferences.

To make the generated response pertinent to the conversation's context, contextual probability score of the candidate utterance U_k is computed. To do so, last utterance in the conversation U_{k-1} is taken in a sequence $S_{ctx} = U_{k-1}$ *[SEP]* U_k, which in turn is given as input to BERT to obtain the U_k's probability score $P_{ctx}^{U_k}$ with the next sentence.

To ensure the appropriateness of the created utterance with the current concept, a concept-relevance score is calculated.Again the BERT model is used to encode the utterance U_k and current concept c_j to obtain the embedding representation U_k^{emb}, and c_j^{emb} respectively. Then, the concept-relevance score $P_{cpt}^{U_k}$ is computed by taking the cosine similarity between U_K^{emb} and c_j^{emb}. Equation 5 is used to calculate each candidate's final score for each speech.

$$P_{final}^{U_k} = \alpha P_{ctx}^{U_k} + \beta P_{ent}^{U_k} + \gamma P_{cpt}^{U_k} \tag{5}$$

We set the values of the constants α, β, and γ as 0.38, 0.30, and 0.32 through several quality evaluations (Detailed in Section A.1.2). The candidate with the highest P_{final} score is chosen as the next utterance.

Manual Filtering of Dialogues: Once the complete conversational dataset is obtained, by iteratively employing the above steps. These dialogues are quality-checked by human evaluators. Each utterance is scored using two parameters, *viz.* Humanness and Concept Consistency. Each utterance is evaluated with a score of 0, 1, or 2, w.r.t these two parameters where 0 denotes the lowest and 2 denotes the highest. Three human experts with post-graduate qualifications were asked to rate all the generated utterances. These professionals are regular employees in our research group and have 2 years of expertise in related fields. Following the ratings, dialogues with utterances having a score of 0 by any expert for 'humanness' or 'concept consistency' are eliminated.

4 Persona-aware Topic-guiding Conversational System

We build our proposed system **RPTCS** in two phases. In first phase, an MLE-loss-based conversational system (MLCS) is trained to learn the user and agent's utterances distribution and obtain natural language interaction. In the second phase, this MLCS is fine-tuned in an RL framework employing proximal policy optimization (PPO) (Schulman et al., 2017)ss method to generate persona-aware topic guiding utterances.

4.1 Phase 1

A multi-turn conversation can be represented as $C = \{a_0, u_0, ..., a_{T-1}, u_{T-1}\}$, where, a and u denotes the agent's and user's utterances. Each conversation is further attributed with a persona $p = \{p_1, p_2, ..., p_m\}$ having a set of m persona statements of user, a topic-path $tp = \{tp_1 \rightarrow tp_2 \rightarrow .. \rightarrow tp_k\}$, with k topics, had to be followed in conversation, and with each of the agent's utterance, a $< topic >$ (any one tp_i from tp) to which conversation has to be guided. Following (Wu et al., 2021), the probability distributions over the conversation C's utterances concatenated with user's persona p, topic-path tp, and agent's $< topic >$ to be guided are decomposed into two LMs *viz.* one for user and other for agent denoted as ρ_u and ρ_a, respectively. Given the conversation context, the LMs ρ_u and ρ_a predict the next token in an agent's generated response $r = \{r_1, r_2, ..., r_t\}$ with t tokens. The joint probability for an utterance u_i or a_i can be formulated as:

$$\rho_u(u_i|u_{<i}, a_{<i}) = \prod_{j=1}^{t_{u_i}} \rho(r_j|r_{<j}, p, tp, u_{<i}, a_{<i}) \tag{6}$$

$$\rho_a(a_i|u_{<=i}, a_{<i}) = \prod_{j=1}^{t_{a_i}} P(r_j|r_{<j}, tp_{<i}, u_{<=i}, a_{<i}) \tag{7}$$

---

[4] Test set accuracy of NLI model is obtained as 88.43%.

User Persona	(i). i enjoy killing sea creatures. (ii). i'm a fan of animals. (iii).i'm practically a chef ! (iv). my father was in the car industry. (v). i enjoy life.	
Seed User Utterance	hey ! i'm a happy camper this evening . just finished making dinner .	
Selected Concept Path	['camper', 'backpack', 'food', 'restaurant']	
Speaker	***Generated Utterance***	***Concept***
Agent	i am a fan of backpacking. i know a lot of people who are happy in their camp.	backpack
User	i would love to go backpacking but i dont have enough money to afford one.	backpack
Agent	i have been known to carry around my backpack full of food, i just have a big appetite.	food
User	i just love food.	food
Agent	you must know your restaurants then.	restaurant

Table 2: Sample dialogues from the corpus generated using our method.

Finally, the conversational system $\rho_\theta(C)$ defined on conversation C is trained by maximizing the likelihood estimation.

$$\rho_\theta(C) = \prod_{T=0}^{T-1} \rho_u(u_i|u_{<i}, a_{<i})\rho_a(a_i|u_{<=i}, a_{<i}) \tag{8}$$

4.2 Phase 2

To generate persona-aware and topic-consistent responses $\rho_\theta(C)$ (MLCS) is fine-tuned in an RL framework. For a given context, $\rho_\theta(C)$ generates *n* possible candidates. These candidate responses are then quality-checked in terms of persona awareness, topic consistency, context consistency, and repetition using respective rewards.

4.2.1 Rewards

To achieve utterance-persona and utterance-topic consistency in generated responses, we design two novel task-specific rewards *viz.* Utterance-Persona consistency (R_1) and Utterance-Topic consistency (R_2). Similarly, to ensure conversation properties like contextual correctness and non-repetitiveness, two generic rewards are designed *viz.* Context consistency (R_3) and Non-Repetition (R_4). Lastly, a compound reward function R, considering both task-specific and generic rewards is computed, which outputs the end reward value for the generated candidate response.

Utterance-Persona consistency Reward: The essence of engaging response generation also relies on persona of the user, hence a conversational system should be able to maintain the utterance-persona consistency in the generated responses. This problem of utterance consistency with persona statements of user can be characterized as a natural language inference (NLI) problem, having three labels *viz.* entailment, neutral and contradiction. Entailed responses are consistent with persona whereas contradictory responses are inconsistent, hence, should be penalized. To build our NLI model, BERT (Devlin et al., 2018) is employed as a classifier, which takes input persona p and generated candidate response r with a $< SEP >$ tag, and outputs one of three classes through a hot vector $[c_e, c_n, c_c]$ (entailment c_e, neutral c_n or contradiction c_c). To achieve the respective class probabilities, a softmax is applied on this hot-vector, i.e. $[prob_{c_e}, prob_{c_n}, prob_{c_c}] = softmax([c_e, c_n, c_c])$. The predicted entailed probability is used to design the reward R_1 which can be written as:

$$R_1 = prob_{c_e}(r_T, p) \tag{9}$$

where r_T represents the generated response at turn T. It can be inferred that R_1 will reward more, if entailment probability is high.

Utterance Topic Consistency Reward: A topic guiding conversational system should not deviate from the topic in-hand. It should be forced to generate topic consistent utterances by rewarding the ones which employ the required topic. To formulate the utterance topic consistency, we considered *cosine* similarity between topic and utterance which is calculated using Sentence-BERT (Reimers and Gurevych, 2019). R_2 can be written as:

$$R_2 = \cos(r_T, tp_T) \tag{10}$$

where tp_T represents the topic at turn T. Higher cosine similarity values will lead to higher rewards for utterance topic consistency.

Context Consistency Reward: A conversational system is required to generate context consistent responses. Therefore, to assess context consistency, we devise a reward by calculating *cosine* similarity (using Sentence-BERT (Reimers and Gurevych, 2019)) between generated response and agent's and user's utterance at turn T. R_3 can be formulated as:

$$R_3 = \frac{1}{2} \times (\cos(r_T, a_T) + \cos(r_T, u_T)) \tag{11}$$

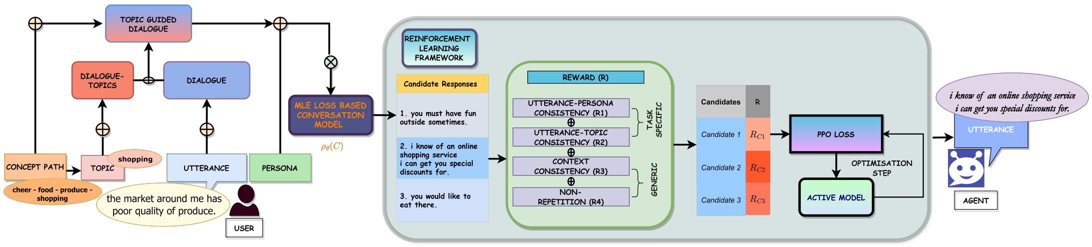


Figure 2: Architectural overview of the proposed **RPTCS**. It is first initialized with trained MLE-loss based model $\rho_\theta(C)$, then is fine-tuned employing PPO-loss utilizing novel reward R to build a persona aware topic guiding conversational system (**RPTCS**)

.

Non-Repetition Reward: Due to repetitive same response generation in a conversation, user's may not show engagement in the conversation. Hence, the generated responses should be diverse with each turn of the conversation. To ensure this, we design a Non-repetition reward R_4 by computing Jaccard distance (Jaccard, 1912; Tanimoto, 1958) between generated responses r_T and r_{T-1} at turns T and $T-1$, respectively. R_4 can be given as:

$$R_4 = 1 - \left(\frac{gr_{T-1} \cap gr_T}{gr_{T-1} \cup gr_T}\right) \tag{12}$$

Reward Function: Lastly, to train the whole system, reward function R is formulated using weighted sum of all these four rewards. To approximate a better function, the sum of all weights has been taken equal to 1.

$$R = \delta_1 R_1 + \delta_2 R_2 + \delta_3 R_3 + \delta_4 R_4 \tag{13}$$

where $\delta_1 + \delta_2 + \delta_3 + \delta_4 = 1$. The obtained value of R assesses the quality of generated response, which is further used to optimize the PPO loss, such that model can learn to generate persona, topic, and context-consistent responses.

4.2.2 Policy

A probability mapping function $\mathcal{P}_\theta$ representing the probability of generating an utterance r consisting of L tokens gives the policy.

$$\mathcal{P}_\theta(r_{1:L}|x) = \prod_{l=0}^{L} \mathcal{P}_\theta(r_l|y_{<l}, x) \tag{14}$$

Proximal Policy Optimisation: Policy updates at each each step are done using PPO method to ensure low variance. It seeks improvement on certain parameters to update the existing policy such that it is not too different from the old policy. First, expected reward is maximized using gradient ascent on loss function $J(\theta)$,

$$\nabla_\theta J(\theta) = Ep_{r\sim\mathcal{P}_\theta}[\nabla_\theta \log \mathcal{P}_\theta(r)\hat{A}_r] \tag{15}$$

Second, large deviations from old policy are restricted by replacing the log term with an importance sampling and clipping is performed to prevent catastrophic forgetting. It relies on specialized clipping without any KL-divergence term (Kullback and Leibler, 1951) or any constraint in the objective function.

$$\begin{aligned} L^{\text{CLIP}}(\theta) = \hat{E}[\min(pr_r(\theta)\hat{A}_r, \text{clip}(pr_y(\theta), \\ 1-\varepsilon, 1+\varepsilon)\hat{A}_r)] \end{aligned}$$

Here, $pr_r(\theta) = \mathcal{P}_\theta^{new}/\mathcal{P}_\theta^{old}$ denotes the ratio of probabilities of the generated response between the new and old policies. ε is the clipping range and $\hat{A}_y$ is the estimated advantage which is the normalized rewards in our case. Lastly, parameters are updated using the following steps:

$$\theta_{k+1} = \underset{\theta}{\text{argmax}} \underset{s,a\sim\mathcal{P}_{\theta_k}}{E}[L^{\text{CLIP}}] \tag{16}$$

5 Experiments

5.1 Implementation Details

MLE loss based conversational system (MLCS) is trained by employing two pre-trained GPT-2 small (Radford et al., 2019) models, one for user and other for agent. To fine-tune trained MLCS in RL-setting, $n = 3$ is selected as per better loss, after experimenting with different values of candidate responses i.e. $n = 2, 3, 4, 5, 10$. The candidate responses are decoded using nucleus sampling (Holtzman et al., 2019) with temperature $T = 0.8$ and probability $p = 0.9$. To train the RPTCS $seed\_value = 10$, $human\_reward = 10$, $max\_candidate\_length = 50$ is adopted with $optimizer = AdamW$ (Loshchilov and Hutter,

2017) and learning rate $\alpha = 2e - 05$, $\varepsilon = 0.2$ and $epochs = 17$. The reward weight combination of $0.3, 0.3, 0.2, 0.2$ are chosen as the final weights for δ_1, δ_2, δ_3, and δ_4, respectively (detailed weight optimization is given in Table 5 of appendix). Due to space restrictions, detailed implementation details of the NLI model to entail the utterance persona consistency, and MLCM are given in Section A.1.3 of the appendix.

5.2 Evaluation Metrics

Both automatic and human evaluations are performed to assess the performance of the proposed system **RPTCS**. To evaluate all three tasks of utterance-persona, utterance-topic and context consistency, three metrics *viz.* **U-PCon**, **U-TCon**, and **CxCon** respectively, are adopted. **U-PCon** computation can be formulated as below:

$$U - PCon = \frac{\sum_{j=1}^{n} class_i(NLI)}{n} \qquad (17)$$

where, $class(NLI)$ gives one of three classes 1 (entailment), 0 (neutral), -1 (contradiction) and n is the number of generated responses. **U-TCon** is calculated as *cosine-similarity* between the generated response r_T and topic tp_T, i.e. $\cos(r_T, tp_T)$. **CxCon** calculates the average METEOR (Banerjee and Lavie, 2005) score for n number of generated responses with respect to the ground truth response a_T at turn T. It can be given as:

$$CxCon = \frac{METEOR(r_T, a_T)}{n} \qquad (18)$$

Further, to evaluate the quality of the generated responses, Perplexity (**PPL**) and response length (**R-LEN**) are calculated.

Human evaluation are performed by three evaluators (regular employees in our research group) with postgraduate experience and having proficiency in similar tasks [5]. Each evaluator is asked to interact with the proposed system 20 times and evaluate the conversations in terms of task-specific and generic metrics. Former includes **PerAw**, **TopGu** - to assess the persona awareness and topic guidance respectively in generated conversations. Latter includes **Fluen**, **Const**, and **N-Rep** - to check fluency, consistency, and non-repetitiveness of generated responses in interacted conversations. All metrics are evaluated on an integer scale of 1-5 [6].

6 Results and Analysis

RPTCS is compared with two baselines: **(1.)** ARDM (Wu et al., 2021): A MLE loss based self-play conversation model, which trains two GPT-2 medium models (one for user and one for agent) alternatively. Here, as we are assessing only agent's performance, hence we train only one GPT-2 small model. **(2.)** RPTCS-R: **RPTCS** with $R = 0$. Further, to check the effects of each of the three aspects *viz.* concept-path, topic and persona, three respective variants of ARDM *viz.* ARDM+CP (ARDM considering concept-path), ARDM+CP+T (ARDM considering concept-path and topic), and ARDM+CP+T+P (ARDM considering concept-path, topic and persons) are also trained to compare. Lastly, to assess the importance of both task-specific (Ts) and generic (Ge) rewards, **RPTCS** is also compared with RPTCS-Ts ($R = \delta_3 R_3 + \delta_4 R_4$) and RPTCS-Ge ($R = \delta_1 R_1 + \delta_2 R_2$).

Automatic Evaluation: It can be noticed in Table 3 that the proposed **RPTCS** performs better than all four baselines *viz.* ARDM, RPTCS-R, RPTCS-Ts and RPTCS-Ge in terms of all the four metrics *viz.* **U-PCon**, **U-TCon**, **PPL** and **R-LEN**. For task-specific metrics **U-PCon**, and **U-TCon**, **RPTCS** achieves better scores of *95.8%*, and 0.414, respectively, with a significant difference of <18.9, 0.124>, <7.7, 0.095>, <8.4, 0.1>, <6.7, 0.042>, and <3.5, 0.047> than the baselines ARDM, ARDM+CP+T+P, RPTCS-R, RPTCS-Ts and RPTCS-Ge, respectively. It can also be inferred that the difference of **U-TPer**, **U-TCon**, **CxCon** scores decreased in order ARDM>RPTCS-R>RPTCS-Ge>RPTCS-Ts. This shows the importance of task-specific rewards in our proposed system **RPTCS** and it can be argued that utterance-persona and utterance-topic consistency rewards do force the system to adapt towards generating persona and topic-consistent responses.

It can also be observed in the Table 3 that **RPTCS** obtains better **PPL** = 6.14 score than that of ARDM, ARDM+CP+T+P, RPTCS-R, RPTCS-Ts and RPTCS-Ge with a difference of 2.53, 1.51, 1.10, 0.89, and 0.84, respectively. This may be due to the Context consistency reward which drives the model to generate responses consistent with the conversation context, which, in turn, leads to the generation of much more natural and fluent responses. Further, obtained a score of **R-LEN** = 15.23 is also better than that of ARDM, ARDM+CP+T+P, RPTCS-R, RPTCS-

---

[5] Human evaluators were paid as per our university norms.

[6] The scale 1-5 denotes low to high intensity such as PerAw = 1 denotes highly persona awareness and PerAw = 5 denotes no persona awareness.

Model	U-PCon	U-TCon	CxCon	PPL	R-LEN
ARDM (Wu et al., 2021)	76.9%	0.290	0.134	8.67	13.11
ARDM+CP	77.2%	0.303	0.145	8.34	13.21
ARDM+CP+T	77.4%	0.314	0.147	8.12	13.27
ARDM+CP+T+P	88.1%	0.319	0.160	7.65	13.86
RPTCS-R	87.4%	0.314	0.152	7.24	13.42
RPTCS-Ts	89.1%	0.372	0.161	7.03	14.27
RPTCS-Ge	92.3%	0.367	0.166	6.98	14.69
RPTCS	**95.8%**	**0.414**	**0.178**	**6.14**	**15.23**

Table 3: Results of automatic evaluation

Model	PerAw	TopGu	Fluen	Const	N-Rep
ARDM	2.69	2.13	3.11	3.83	2.94
ARDM+CP+T+P	3.41	2.64	3.57	4.02	3.21
RPTCS-R	3.31	2.66	3.65	3.98	3.26
RPTCS-Ts	3.40	2.70	3.71	4.12	3.42
RPTCS-Ge	3.62	2.81	3.84	4.04	3.37
RPTCS	**3.82**	**2.96**	**4.01**	**4.14**	**3.55**

Table 4: Results of human evaluation

Ts, and RPTCS-Ge with a difference of 2.12, 1.37, 1.81, 0.99, and 0.54, respectively. This indicates that the **RPTCS** is able to generate longer responses, hence, showcasing more engagingness with the user. It can be due to the incorporation of all four rewards where R_1, R_2, and R_3 play the crucial role of persona, topic, and context consistency, and R_4 maintain the non-repetitiveness, hence, driving the agent to build the rapport with a user as well as be on the goal topic by generating diverse and interactive responses. Engagingness rewards - forcing the model to generate more interactive and engaging responses. Lastly, it can also be seen in Table 3 that **PPL ter** and **R-LEN** scores of **RPTCS** decreased in order ARDM>RPTCS-R>RPTCS-Ge>RPTCS-Ts>**RPTCS**, hence, strengthening our hypothesis for the requirement of both task-specific and generic rewards to generate persona aware topic consistent responses.

Human Evaluation: Table 4 shows the human evaluation results for all five models *viz.* ARDM, ARDM+CP+T+P, RPTCS-R, RPTCS-Ts, RPTCS-Ge and ARDM, RPTCS-R, RPTCS-Ts, and **RPTCS**. It can be noted that **RPTCS** yields better scores of **PerAw**, **TopGu**, **Fluen**, **Const** and **N-Rep** as compared to the the baselines, ARDM, RPTCS-R, RPTCS-Ts and RPTCS-Ge. Scores of **Fluen**: 4.01, **Cons**: 4.14, and **N-Rep**:3.55 implies that all the four rewards R_1, R_2, R_3, and R_4 play a critical role in obtaining most fluent, consistent, and non-repetitive responses as compared to other four models. Further, in terms of **PerAw**, and **TopGu**, **RPTCS** attains well scores of 3.82, and 2.96, respectively, showcasing the importance of rewards R_1, and R_2. Therefore, it can be inferred that employing utterance-persona and utterance-topic consistency rewards helps the proposed model to generate persona-aware responses and is able to guide the conversation keeping the topic intact.

7 Conclusion

For an open domain conversation to be successful, proper concept transition befitting the context is essential. Further, persona awareness of the user may lead to user adaptive response generation, hence, resulting in more engaging and interactive conversations. Therefore, a conversational agent should be able to transition the concept efficiently as well as build a rapport with the user by understanding his/her persona. To encompass both of these aspects, we proposed here a Reinforced Persona-aware Topic-guiding Conversational System (**RPTCS**). First, we create a persona-aware topic transition dataset (PTCD) by leveraging the few-shot prompt feature of the language model. Second, employing GPT-2 small, we train an MLE loss-based conversational model (MLCM) on PTCD. Lastly, using a novel designed reward function to ensure aspects of persona, topic, and context consistency with non-repetitiveness, we fine-tune MLCM adopting PPO loss optimizer in an RL framework. Automatic and human evaluation results strengthen the design and use of rewards and concludes that our proposed model **RPTCS** achieves state-of-the-art performance compared to the strong baselines. Our results also concludes that **RPTCS** is able to retain and facilitate persona awareness, naturalness, and consistency at par in an ongoing dialogue.

In the future, we would like to look into personality traits such as age, gender, etc. to model a persona-aware conversational system.

8 Limitations

RPTCS has also some limitations. First, to create the data, GPT-J is used which requires a large GPU memory size (here, 40 GB). Further, empirical analysis for each of the possible combinations of different rewards weights may lead to model training and validation time to months. Hence, some heuristics should be used to choose the set of combinations of rewards and reward weights (such as here, we restricted the reward weight sum as 1).

When interacting with the system, if users continuously state short and direct responses such as *'Yes'*, *'I don't know'*, *'No'*, *'I can'*, *'Okay'*, then the system first tries to respond by inquiring about topic like *'restaurant'*, *'job'*, *'shopping'* or *'travel'* but after two or three turns starts deviating and generating out of the context or hallucinated responses. It is also seen that sometimes model starts attending persona statements of the user frequently and generate most of the time only persona awared responses which tend to be out of context. Hence, the model should be forced to generate only relevant responses and persona attention should be controlled. This opens up the door for future studies to build a controlled persona aware topic guiding conversational system.

9 Ethical considerations

We use a freely available dataset under a Creative Commons license to create our new dataset. The dataset has been used only for academic purposes, and in complete compliance with the license. The dataset created in this work will be made available only after filling and signing an agreement declaring that the data will be used only for research purposes. The annotation for manual evaluations was done by human experts, who are regular employees of our research group and are paid in accordance with the institute's policy. There are no other issues to declare.

10 Acknowledgement

The research reported in this paper is an outcome of the project "Autonomous Goal-Oriented and Knowledge-Driven Neural Conversational Agents", sponsored by Accenture LLP. Kshtij Mishra acknowledges the support of the "Prime Ministers Research Fellowship (PMRF)" Program of the Government of India.

References

Satoshi Akasaki and Nobuhiro Kaji. 2017. Chat detection in an intelligent assistant: Combining task-oriented and non-task-oriented spoken dialogue systems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1308–1319.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Nicholas J Belkin and Alina Vickery. 1985. *Interaction in information systems: A review of research from document retrieval to knowledge-based systems.* 025.04 BEL. CIMMYT.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. *arXiv preprint arXiv:1902.01109*.

Jianfeng Gao, Michel Galley, Lihong Li, et al. 2019. Neural approaches to conversational ai. *Foundations and trends® in information retrieval*, 13(2-3):127–298.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Vrindavan Harrison, Juraj Juraska, Wen Cui, Lena Reed, Kevin K Bowden, Jiaqi Wu, Brian Schwarzmann, Abteen Ebrahimi, Rishi Rajasekaran, Nikhil Varghese, et al. 2020. Athena: Constructing dialogues dynamically with discourse constraints. *arXiv preprint arXiv:2011.10683*.

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. *arXiv preprint arXiv:1704.07130*.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2020. Aligning ai with shared human values. *arXiv preprint arXiv:2008.02275*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Chao-Chun Hsu, Sheng-Yeh Chen, Chuan-Chun Kuo, Ting-Hao Huang, and Lun-Wei Ku. 2018. Emotionlines: An emotion corpus of multi-party conversations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Paul Jaccard. 1912. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Anton Leuski, Ronakkumar Patel, David Traum, and Brandon Kennedy. 2006. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pages 18–27.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Liang Qiu, Yizhou Zhao, Jinchao Li, Pan Lu, Baolin Peng, Jianfeng Gao, and Song-Chun Zhu. 2022. Valuenet: A new dataset for human value driven dialogue system. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(5):2468–2484.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1702–1723.

Yusuxke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, Takeshi Shinohara, and Setsuo Arikawa. 1999. Byte pair encoding: A text compression scheme that accelerates pattern matching.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Taffee T Tanimoto. 1958. Elementary mathematical theory of classification and prediction.

Johanne R Trippas, Damiano Spina, Paul Thomas, Mark Sanderson, Hideo Joho, and Lawrence Cavedon. 2020. Towards a model for spoken conversational search. *Information Processing & Management*, 57(2):102162.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Xiaoyang Wang, Chen Li, Jianqiao Zhao, and Dong Yu. 2021. Naturalconv: A chinese dialogue dataset towards multi-turn topic-driven conversation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14006–14014.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741.

Terry Winograd. 1977. A framework for understanding discourse. *Cognitive processes in comprehension*, 63:88.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Qingyang Wu, Yichi Zhang, Yu Li, and Zhou Yu. 2021. Alternating recurrent dialog model with large-scale pre-trained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1292–1301.

Wenquan Wu, Zhen Guo, Xiangyang Zhou, Hua Wu, Xiyuan Zhang, Rongzhong Lian, and Haifeng Wang. 2019. Proactive human-machine conversation with explicit conversation goals. *arXiv preprint arXiv:1906.05572*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546.

Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. 2020. Dialogue-based relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4927–4940.

Houyu Zhang, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2020. Grounded conversation generation as guided traverses in commonsense knowledge graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2031–2043.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, and Xiaoyan Zhu. 2020. Kdconv: A chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7098–7108.

A Appendix

A.1 Implementation Details

A.1.1 Data Creation Models Details

All the models were implemented using PyTorch (Paszke et al., 2017). The BERT model trained on the Persona-NLI dataset was implemented using the transformers library (Wolf et al., 2019). The BERT model was initialized with the weights of 'bert-base-uncased'. The model was trained with an initial learning rate of 1e-4 with a linear schedule and a warmup (Vaswani et al., 2017), using the Adam Optimizer (Kingma and Ba, 2015). Mini-batches of size 8 were used during training.

A.1.2 Values for utterance selection constants:

The values of α, β, and γ were determined empirically. Initially, all the values were set to 0.33. Then to ensure a balance between concept-consistency, persona-entailment, and context-relevance, these values are varied only in the range of 0.30 to 0.40. It was found that a higher value of α (i.e. of context-relevance) resulted in more human-like utterances by avoiding abrupt transitions. The value of γ lower than 0.32 resulted in generating a higher number of concept-agnostic utterances. In this manner, the final values of α=0.38, β=0.30, and γ=0.32 are obtained.

A.1.3 Maximum Likelihood Estimation loss based Conversational Model

To train MLE loss based dialogue model, two pre-trained GPT-2-medium models (Radford et al., 2019) with 345M parameters are used to model the persuadee's and persuader's utterances. Here, pre-trained GPT-2-medium model consists of 24-layers, 1024 hidden units and 16 heads. We use Byte-Pair Encoding (Shibata et al., 1999) to tokenize the words. The dialogue model is trained with a learning rate = 3e-5, using AdamW optimizer (Kingma and Ba, 2015) with 100 warm-up steps and dropout rate of 0.1.

A.1.4 Specifications of Computational Resource

To train the transformer based NLI model, MLE-loss based conversational model and proposed **RPTCS**, following configurations are used:

- **GPU:** A100-PCIE-40GB.
- **CUDA Support:** CUDA 11.x (or later.
- **Memory clock:** 1215 MHz.
- **Total board power:** 250 W.
- **GPU clocks:** Base: 765 MHz, Boost: 1410 MHz.
- **Memory Size:** 40 GB.
- **Memory Type:** HBM2.
- **Bus Width:** 5120 bits.

A.1.5 Model Run time Specifications

RPTCS takes approximately 3 mins/epoch to train the model, hence for 30 epochs, it took 90 minutes to train the model. Further, if we try to perform validation along with training, considering three candidate responses per utterance per dialogue, **RPTCS** takes approximately 30 mins/epoch, hence, total time it took for 900 minutes (15 hours) to train and validate the model. Finally, to evaluate the model, the testing of proposed system takes approximately 5 minutes for 200 utterances.

REWARD WEIGHT OPTIMIZATION				
δ_1	δ_2	δ_3	δ_4	**PPL**
0.4	0.4	0.1	0.1	6.53
0.1	0.1	0.4	0.4	6.82
0.2	0.2	0.3	0.3	6.64
0.2	0.3	0.3	0.2	6.44
0.3	0.2	0.3	0.2	6.30
0.3	**0.3**	**0.2**	**0.2**	**6.14**

Table 5: Weight Optimisation using different values of δ.

A.2 Data Creation

In total 22 concepts were used as targets for obtaining the path. The complete list of the concepts is as follows: (i). travel, (ii). journey, (iii). voyage, (iv). outing, (v). restaurant, (vi). hotel, (vii). inn, (viii). diner, (ix). cafe, (x). canteen, (xi). bar, (xiii). shopping, (xiv). mall, (xv). market, (xvi). grocery, (xvii). electronics, (xviii). mobile, (xix). laptop, (xx). computer, (xxi). smartphone, and (xxii). camera.

We use six-shot prompts to generate the synthetic data. We provide a sample of the one-shot version of the prompt below. At the end of the sequence, we also append the persona, path, and utterances for which the next utterance needs to be generated. The six-shot prompt follows the same pattern with six examples in the input sequence. We restrict our prompt to six shots since the maximum sequence length allowed in GPT-J is 2,048:

<startdial>
<persona> i enjoy sprinting and long races. | i am at the end of my career. | i recently started a position helping others with daily challenges | i used to be very unhealthy. </persona>
<transitions> retirement -> retreat -> travel </transitions>
<topic> topic remain | retirement </topic> user: i am thinking about my upcoming retirement .
<topic> topic transition | retirement -> retreat </topic> agent: you should go on a retreat, it will help clear your mind and keep you healthy .
<topic> topic remain | retreat </topic> user: i've been thinking about it, but i am a little short on money .
<topic> topic transition | retreat -> travel </topic> agent: maybe you could go on a trip to a country with low cost of living.
</enddial>

<startdial>
<persona> like to tinker with machines. | i had a run-in with the law | i enjoy seeing nature. | i am not an honest person. </persona>
<transitions> scene -> photo -> camera </transitions>
<topic> topic remain | scene </topic> user: the sunset makes for a great scenery .
<topic> topic transition | scene -> photo </topic> agent:

What Did You Learn To Hate? A Topic-Oriented Analysis of Generalization in Hate Speech Detection

Tom Bourgeade[1], **Patricia Chiril**[3], **Farah Benamara**[1,2], **Véronique Moriceau**[1]
(1) IRIT, Université de Toulouse, *CNRS, Toulouse INP, UT3*, Toulouse, France
`{firstname.lastname}@irit.fr`
(2) IPAL, CNRS-NUS-ASTAR, Singapore
(3) University of Chicago, Chicago, IL, USA
`pchiril@uchicago.edu`

Abstract

Warning: *This paper includes messages that may contain instances of vulgarity, degrading terms, or hate speech, which may be offensive or upsetting to some readers.*

Hate speech has unfortunately become a significant phenomenon on social media platforms, and it can cover various topics (*misogyny*, *sexism*, *racism*, *xenophobia*, etc.) and targets (e.g., *black people*, *women*). Various hate speech detection datasets have been proposed, some annotated for specific topics, and others for hateful speech in general. In either case, they often employ different annotation guidelines, which can lead to inconsistencies, even in datasets focusing on the same topics. This can cause issues in models trying to generalize across more data and more topics in order to improve detection accuracy. In this paper, we propose, for the first time, a topic-oriented approach to study generalization across popular hate speech datasets. We first perform a comparative analysis of the performances of Transformer-based models in capturing *topic-generic* and *topic-specific* knowledge when trained on different datasets. We then propose a novel, simple yet effective approach to study more precisely which topics are best captured in implicit manifestations of hate, showing that selecting combinations of datasets with better out-of-domain topical coverage improves the reliability of automatic hate speech detection.

1 Introduction

On social media and other online communication platforms, hate speech (HS hereafter) is found in many forms, from textual harassment to threats, targeting an individual or group (e.g., black people, women), based on some characteristics, such as race, color, ethnicity, gender, sexual orientation, nationality, religion, etc, which we refer to as *topics*. While the broad nature of these topics of HS is generally understood (Erjavec and Kovačič, 2012), determining whether a social media post is a manifestation of HS (and if so, to which topic(s) it belongs to) is not a trivial task for humans and automated machine-learning systems. Indeed, the latter often require large quantities of annotated data, which in the case of HS, is made difficult due to (1) the absence of a comprehensive definition of these topics; but also, (2) the potential internal biases and subjectivity present in annotators and/or annotation guidelines employed to construct annotated datasets (Schmidt and Wiegand, 2017; Fortuna and Nunes, 2018; Vidgen et al., 2019; Fortuna et al., 2020).

To palliate these issues, cross-dataset evaluation has become an active line of research aiming at studying the generalization capabilities of HS detection systems to unseen data during training (Talat et al., 2018; Ludwig et al., 2022; Toraman et al., 2022). As data collection and annotation is an expensive and time-consuming process, current approaches use *mixtures* of existing HS detection datasets to study generalization across different social media platforms (Swamy et al., 2019; Salminen et al., 2020; Nejadgholi and Kiritchenko, 2020) or across different manifestations of HS, often relying on one-to-many (train on one dataset/mixture, test on others) experimental settings (Fortuna et al., 2021; Talat et al., 2018; Chiril et al., 2022; Talat et al., 2018; Karan and Šnajder, 2018). In most of these, a unification scheme is proposed to adapt the original datasets' potentially fine-grained annotations to a set of binary labels, which can often fail to take into account the heterogeneity inherently present across different encodings of HS (Vidgen et al., 2019).

In addition, some datasets intentionally focus only on some specific kinds of manifestations of HS covering for example gender (e.g., misogyny, sexism), ethnicity, religion or race (e.g., xenophobia, anti-immigrants/refugees HS), i.e., they are *topic-specific*, whereas others attempt to cast a

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3495–3508
May 2-6, 2023 ©2023 Association for Computational Linguistics

wider net without specific sets of topics in mind, i.e., they are *topic-generic*. In either case, variations will always exist, either due to the reasons mentioned above, or due to shifts in the content found on social media through time. All of these may cause significant differences in the topics of HS that classification models might learn to recognize, which ultimately might make them less reliable, particularly in out-domain contexts (Yin and Zubiaga, 2021).

Acknowledging these issues and being aware of the noise that will invariably be introduced by mixing together different datasets, we propose for the first time to empirically study what modern Transformer architectures can effectively learn to generalize from existing datasets, in a unified setting, with a *focus on their topical nature*. More precisely: *Do models learn similarly from topic-generic datasets as from mixtures of topic-specific datasets? Does this acquired knowledge generalize to implicit expressions of HS, and if so, which finer-grained HS topics are learned by these models?* Our contributions are:

(1) An in-depth analysis of the generalizability of generic HS datasets, in which we show how using mixtures of those could be effective to attain better generalization across more topics.

(2) A similar analysis for *topic-specific* datasets, for which we show successful generalization in-domain, especially when using models fine-tuned on mixtures of such corpora.

(3) A novel, simple yet effective approach to study which finer-grained topics are best captured when dealing with *implicit* expressions of HS. We show that selecting mixtures of datasets with a better topical coverage can improve the reliability of models for out-of-domain applications.

2 Related Work

A number of previous works have studied generalizability of HS detection datasets and models, with different focuses: for example, Fortuna et al. (2020) and Fortuna et al. (2021) have analyzed the compatibility of many HS datasets (including some used in this study), both in terms of their properties (origin of data, annotated phenomena, class definitions, etc.) and empirically with intra- and inter-dataset generalization experiments. They conclude that model choice, intra-dataset performance, and the type of phenomenon being classified, are the most important factor that determine generalization. In particular, they conclude that phenomena like *toxicity*, *abuse*, or *offensiveness* that are often lexicalized are easier to generalize than hate speech, something which was also confirmed and discussed in other works (Nejadgholi and Kiritchenko, 2020; Yin and Zubiaga, 2021). In this paper, we are interested in the topic-oriented nature of HS specifically, and thus discard these former classes from our study.

Alternative solutions have recently been proposed to handle unseen topics by building new target-oriented datasets from scratch, such as the `HateXplain` dataset (Ludwig et al., 2022), with an additional labeling of the target topics of hate (Race, Religion, and Origin). While such linguistic resources are valuable for the research community, we believe existing larger datasets could be successfully exploited for generalization, as mixtures of datasets, as shown for example by Fortuna et al. (2018), Salminen et al. (2020), and Chiril et al. (2022), or, in a different fashion, by multitask HS detection systems or domain adaptation techniques (Talat et al., 2018; Kapil and Ekbal, 2020; Safi Samghabadi et al., 2020). In this paper, we continue this line of research by proposing for the first time, as far as we know, models able to generalize across topic generic vs. specific datasets as well as predict fine-grained manifestations in implicit hate messages. This is particularity challenging as these manifestations are more difficult to generalize due to their limited lexical features (ElSherief et al., 2021; Qian et al., 2019).

To this end, we rely on Transformer models like BERT that have been shown to be able to generalize better overall than previous architectures (Swamy et al., 2019). In particular, we experiment with both existing BERT-like models, some pre-adapted to the domain of HS, as well as T5 (Raffel et al., 2020), a text-to-text architecture that has never been used in a generalizability study.

3 Datasets

We experiment with six popular and freely available English tweets corpora (or English subsets thereof) from previous studies. We first present the datasets and how they were used in a common experimental setting. We then provide some discussions on the compatibility of these datasets, and how this may impact the generalization performances in our experiments.

3.1 Topic Generic vs. Topic Specific Datasets

The first two datasets are *topic-generic* while the other four are *topic-specific*, as follows.

`Davidson` (Davidson et al., 2017). It contains English tweets annotated into *hate speech*, *offensive*, and *neither*, with the intent of helping to distinguish between messages simply containing offensive terms, from those actually manifesting HS.

`Founta.` We make use of the dataset released by (Kallumadi et al., 2020) which is an updated version of the dataset initially proposed by Founta et al. (2018) and annotated for four types mutually exclusive of abusive behaviors: *abusive*, *hateful*, *spam* and *normal*.

`IberEval` (Fersini et al., 2018b) and **`Evalita`** (Fersini et al., 2018a) are part of the Automatic Misogyny Identification (AMI) shared task which aims at identifying tweets that convey hate or prejudice against women while categorizing different forms of misogynous behavior. We only use the main binary layer of annotation (i.e., presence vs. absence of misogyny).

`HatEval` (Basile et al., 2019) also known as SemEval-2019 Task 5, is a topic-specific HS detection dataset with tweets targeting *immigrants* and *women*, and annotated with three different binary layers of annotation: hateful/non-hateful, targeting a group/individual, aggressive/non-aggressive. Note that most of the tweets that target women in this dataset were derived from the `AMI corpora` (`IberEval` and `Evalita`).

`Waseem` (Talat and Hovy, 2016). It contains tweets targeting gender minorities, instances of racism, and tweets that were judged to be neither sexist nor racist.

We frame all the datasets used here as binary HS classification tasks, with the labels *"hateful"* (also referred to as the positive class) for instances containing some manifestation of hate speech, and *"normal"* for those containing none. Hence, for `Founta` and `Davidson` we filter out retweets/duplicates (keeping only the source tweets and their annotations) and only keep the *hateful/hate speech* and *normal* classes, and similarly, for the different topic-specific datasets, we unify the specific hate classes (*misogyny*, *sexism*, *racism*) with *hateful*, and the respective negative classes with *normal*.

To allow a more granular and topic-level analysis of results, we split all multi-targets topic-specific datasets into separate training, validation, and testing sets, according to the topics. Therefore, we split `HatEval` into its two topics subsets (i.e., $\texttt{HatEval}_{\texttt{women}}$ and $\texttt{HatEval}_{\texttt{immigrants}}$) and `Waseem` into $\texttt{Waseem}_{\texttt{sexism}}$ and $\texttt{Waseem}_{\texttt{racism}}$. However, in this last dataset, because only one negative class is provided, and corresponds to both the absence of sexism and racism, we choose to duplicate it and use it as the negative class for both subsets.

When not explicitly provided, we use a 75%-20%-5% train-test-validation split ratio. Table 1 further details how datasets have been mixed to train our models.

3.2 Issues with Dataset Compatibility

Because we also manipulate mixtures of these corpora, the effective hateful and non-hateful classes will contain instances annotated within different contexts and labelling guidelines. For example, the `Evalita` and `IberEval` datasets are annotated only for the presence of misogyny, and not other manifestations of HS. This makes their negative (i.e., non-misogyny) class inconsistent with, say, the negative class from a topic-generic dataset, which is not ideal.

Aside from re-annotating all these datasets under a unified and consistent annotation schema, the issues that may arise as part of these simplifications cannot be circumvented, and should thus be taken into account as a fixed parameter in our experiments. These issues have been broadly acknowledged in the relevant literature (Schmidt and Wiegand, 2017; Fortuna and Nunes, 2018; van Aken et al., 2018). As noted by Malmasi and Zampieri (2018) and Poletto et al. (2021), the distinctions between offense, abuse, and HS, are not always clear-cut, which can cause issues in generalization experiments, due to the former's in theory more lexical nature (Vidgen et al., 2019; Fortuna et al., 2020).

Furthermore, as noted by Madukwe et al. (2020), even when using relatively similar definitions for these phenomena, a number of other parameters may affect the compatibility, consistency, and comparability between HS detection datasets, from biases introduced in the annotations, differences in preprocessing steps (e.g., anonymization, emojis, URLs, etc.), to issues of class balance and unspecified train-test-validation splits.

We are aware that mixing the datasets may effectively introduce various kinds of noise in the training and evaluation data, but we consider it

Topic	Dataset	Abbrev.	Total Size	Pos. Size	Ratio	Train	Val.	Test
Topic-Generic	`Davidson`	`Gene Davi`	5590	1430	25.58%	4293	179	1118
	`Founta`	`Gene Fnt`	57355	4119	7.18%	44048	1836	11471
Topic-Specific (Gender)	`Evalita`	`Gndr Evit`	5000	2245	44.90%	3839	161	1000
	HatEval$_{women}$	`Gndr HatE`	6472	2845	43.96%	4500	500	1472
	`IberEval`	`Gndr Iber`	3977	1851	46.54%	3120	131	726
	Waseem$_{sexism}$	`Gndr Wasm`	14531	3216	22.13%	11159	465	2907
Topic-Specific (Race)	HatEval$_{immigrants}$	`Race HatE`	6499	2617	40.27%	4500	500	1499
	Waseem$_{racism}$	`Race Wasm`	13272	1957	14.75%	10192	425	2655
Mixtures	Topic-Generic Mixture	`Gene Mixt`	`Davidson` + `Founta`					
	Gender Topic Mixture	`Gndr Mixt`	`Evalita` + HatEval$_{women}$ + `IberEval` + Waseem$_{sexism}$					
	Race Topic Mixture	`Race Mixt`	HatEval$_{immigrants}$ + Waseem$_{racism}$					
	Topic-Specific Mixture	`Spec Mixt`	Topic Gender Mixture + Topic Race Mixture					

Table 1: Overview of the datasets used to train models in this study.

to be part of the experimental settings, especially since these kinds of issues would likely as well be encountered in end-user applications dealing with raw unfiltered data.

4 Models

To study how different pre-trained models and architectures may differ in how they capture HS in various settings, we choose five pre-trained Transformers from the literature. Among them, the last three have been adapted to the domain of HS detection, either through pretraining or pre-finetuning, on data related directly or indirectly to HS detection. For the experiments, we relied on the Hugging Face transformers library (Wolf et al., 2020).

RoBERTa-base (Liu et al., 2019) is an optimized BERT-like (Devlin et al., 2019) encoder Transformer commonly used for various NLP classification tasks, including HS detection.

T5-base is the 220 million parameters pretrained variant of the T5 architecture, initially proposed by Raffel et al. (2020). It differs from BERT-like encoder Transformers in that it is a *text-to-text* model, for which classification tasks are reframed as text generation, with the output labels used in their textual form (in our case, *"normal"* and *"hateful"*), and as this particular variant was also pretrained on various supervised tasks (sentiment analysis, natural language inference, and question answering, etc.), a task prefix is traditionally prepended to the input text, which for our fine-tuning, we fix to *"hate speech: "*.

fBERT (Sarkar et al., 2021) and **HateBERT** (Caselli et al., 2021) are two models derived from BERT, retrained with a Masked Language Modelling (MLM) objective on over 1.4 million social media offensive posts from the SOLID dataset, and Reddit Abusive Language English dataset (RAL-E), respectively.

ToxDectRoBERTa was proposed by Zhou et al. (2021), and is a RoBERTa-large model, finetuned on `Founta`, with the *hateful* and *abusive* classes merged into a single *toxic* class. The authors use the methods proposed by Clark et al. (2019) and Swayamdipta et al. (2020) to attempt to avoid dataset bias issues, such as spurious correlations between particular lexical and dialectical markers (such as those found in African American English) with the toxic class.

5 Cross-Topics Generalization

We first study the differences in generalizability between topic-generic and topic-specific datasets and their potential mixtures, in a cross-dataset/mixture setting: in each individual experiment, one of the previously described models is trained on one dataset/mixture and tested on all individual test sets. For technical details on the experimental parameters used in this study, see Appendix A.

To measure improvements or deteriorations in performances in the generalization experiments, Table 2 presents an intra-dataset evaluation, in which models are simply trained and tested on the same datasets, for the sake of comparison. In the remainder of this section, we report macro F1 scores on all test sets (see Table 1 for abbreviations), with the best scores for each test set highlighted in **bold**. Scores are gradient-colored for legibility.

Dataset \ Model	**T5-base**	**RoBERTa-base**	**ToxDectRoBERTa**	**fBERT**	**HateBERT**
`Gene Davi`	92.70	93.30	**93.78**	92.46	93.14
`Gene Fnt`	81.08	80.84	**85.92**	80.98	79.41
`Gndr Evit`	67.00	69.58	67.37	**73.12**	71.02
`Gndr HatE`	46.29	56.88	51.65	**64.28**	59.11
`Gndr Iber`	84.86	**88.76**	83.89	84.55	86.41
`Gndr Wasm`	85.81	**87.11**	86.27	86.28	86.05
`Race HatE`	38.73	38.24	40.75	**41.72**	38.53
`Race Wasm`	87.03	86.27	86.29	**87.53**	86.37

Table 2: Results of intra-dataset evaluations (training and testing on the same dataset's train and test sets).

5.1 Learning from Topic-Generic Datasets

We investigate here how well knowledge can transfer from topic-generic to topic-specific datasets. We thus train each of the five chosen models on the `Davidson` (`Gene Davi`) and `Founta` (`Gene Fnta`) train sets, as well as on a mixture of the two (`Gene Mixt`), then evaluate those models on every individual test set.

Table 3 presents the results of these evaluations, in terms of macro F1 score. Observing the results, firstly, we can notice that `Davidson` and `Founta` generalize relatively well to each other, with relatively small deteriorations in F1 scores (~ -10 F1) compared to the intra-dataset models (see Table 2), in the favor of `Davidson` $\rightarrow$ `Founta` for our 5 models. When used as a mixture of datasets (`Gene Mixt`), we observe very small improvements for 3 of our five models and very small deteriorations ($\sim \pm 1$ F1) for the rest (RoBERTa-base and ToxDectRoBERTa), which would tend to indicate a good compatibility between these two datasets, or at the very least, some high overlap in the topic-generic HS knowledge extracted by Transformer models across these two datasets.

Looking then at the evaluation results obtained on the topic-specific test sets, we can make a number of observations: first, we note quite significant deteriorations compared to the intra-dataset models for all datasets, except for both HatEval subsets, in particular $\texttt{HatEval}_{\texttt{immigrants}}$, for which all three topic-generic datasets/mixture yield significant improvements (from $\sim +3$ F1 up to $\sim +27$ F1). We believe this to be due to a significant distribution shift between the HatEval train and test set, which would explain why the models perform so poorly in the intra-dataset setting, while using different training sets appears to significantly improve performance.

For the other datasets, the least significant deteriorations can be found for `Evalita`, while the greatest ones are found for $\texttt{Waseem}_{\texttt{sexism}}$, which may indicate an overall low overlap in the types of manifestations of HS found in the topic-generic and topic-specific datasets explored here. However, in most cases, the use of the topic-generic mixture (`Gene Mixt`) appears to be beneficial, in that it tends to attenuate the worst deteriorations found in models trained on `Davidson` or `Founta` individually. This may indicate that using mixtures of topic-generic training datasets may be beneficial when trying to detect HS instances where the topic is not necessarily known.

5.2 Learning from Topic-Specific Datasets

We similarly trained HS classifiers, this time on the remaining 6 topic-specific datasets, separated into the Race and Gender topics (see Table 1). Table 4 presents the results (in a form similar to Table 3), however, for space reasons and as they are the focus of our study, we only present the results for the gender and race topics mixtures (`Gndr Mixt`, and `Race Mixt` respectively), as well as the mixture of both (`Spec Mixt`). See Table A in Appendices for more detailed results.

Observing the results, we can first see that, similarly to the previous experiments, training on these three mixtures of topic-specific datasets does not seem to generalize too well back onto the topic-generic datasets, with even greater deteriorations in F1 scores compared to the intra-dataset setting (cf. Table 2). This is the most pronounced for `Race Mixt`, which is the smaller of the two one-topic mixtures. Even more prominently than for topic-generic mixture discussed previously (`Gene Mixt`), we find that the topic-specific mixture (`Spec Mixt`), which combines both the Gender and Race topics, yields significantly lesser deteriorations than ei-

Model	T5-base			RoBERTa-base			ToxDectRoBERTa			fBERT			HateBERT		
Train / Test	`Gene Davi`	`Gene Fnta`	`Gene Mixt`	`Gene Davi`	`Gene Fnta`	`Gene Mixt`	`Gene Davi`	`Gene Fnta`	`Gene Mixt`	`Gene Davi`	`Gene Fnta`	`Gene Mixt`	`Gene Davi`	`Gene Fnta`	`Gene Mixt`
`Gene Davi`	92.70	82.64	**93.92**	93.30	82.50	91.60	93.78	84.73	90.96	92.46	85.59	92.92	93.14	85.43	93.42
`Gene Fnt`	73.02	81.08	81.41	74.50	80.84	80.49	81.42	**85.92**	84.70	76.47	80.98	81.37	73.02	79.41	80.32
`Gndr Evit`	63.29	58.07	61.79	60.68	58.51	56.97	62.10	54.25	60.20	61.44	57.00	62.16	**63.92**	56.74	59.99
`Gndr HatE`	42.21	42.90	54.60	46.39	55.17	49.45	40.30	39.81	52.35	38.25	49.75	45.83	44.46	55.76	**56.31**
`Gndr Iber`	57.76	**74.22**	59.48	60.50	62.96	68.89	64.14	73.00	67.83	60.34	71.43	63.74	60.12	67.11	65.24
`Gndr Wasm`	54.06	59.07	52.21	51.55	51.11	51.63	51.13	56.44	52.72	53.84	**61.47**	51.73	52.36	58.24	54.52
`Race HatE`	43.75	64.54	50.97	47.79	56.76	63.71	52.23	**68.58**	66.92	50.65	66.49	45.70	44.13	66.08	65.62
`Race Wasm`	56.66	72.73	73.69	56.89	**75.52**	70.72	69.92	73.47	73.45	70.16	72.33	70.01	63.01	72.81	73.28

Table 3: Results of learning from the two topic-generic datasets used here, Davidson (`Gene Davi`) and Founta (`Gene Fnta`), as well as their mixture (`Gene Mixt`). Scores are gradient-colored for legibility.

Model	T5-base			RoBERTa-base			ToxDectRoBERTa			fBERT			HateBERT		
Train / Test	`Gndr Mixt`	`Race Mixt`	`Spec Mixt`	`Gndr Mixt`	`Race Mixt`	`Spec Mixt`	`Gndr Mixt`	`Race Mixt`	`Spec Mixt`	`Gndr Mixt`	`Race Mixt`	`Spec Mixt`	`Gndr Mixt`	`Race Mixt`	`Spec Mixt`
`Gene Davi`	61.72	50.23	68.33	62.20	47.75	68.22	65.04	62.26	67.13	68.38	51.78	**72.60**	62.67	56.10	68.75
`Gene Fnt`	58.61	57.44	62.09	57.60	54.14	61.17	62.92	60.92	63.24	60.46	55.14	59.93	57.91	58.11	**63.36**
`Gndr Evit`	82.90	36.42	84.78	86.33	35.92	86.89	85.40	42.15	85.58	**88.59**	38.58	87.37	87.96	38.95	86.80
`Gndr HatE`	47.83	36.82	54.58	52.69	37.02	53.65	50.80	48.80	**58.53**	54.80	36.71	58.44	55.33	42.57	49.06
`Gndr Iber`	**93.31**	40.84	93.17	92.12	38.66	92.84	92.32	40.84	92.30	92.55	39.42	92.55	92.85	39.42	92.86
`Gndr Wasm`	85.14	44.92	87.85	85.66	43.96	88.53	85.72	44.76	87.39	86.92	43.59	87.93	84.85	44.28	**88.87**
`Race HatE`	42.98	40.49	**44.29**	37.69	40.44	38.06	38.59	41.23	44.03	39.47	43.68	40.40	37.56	37.01	35.80
`Race Wasm`	46.49	86.76	85.39	46.30	85.90	88.00	48.12	87.05	85.46	48.58	86.25	**89.17**	46.66	86.42	88.37

Table 4: Results of learning from the mixtures of the gender topic-specific (`Gndr Mixt`) datasets (Evalita, $\text{HateEval}_{\text{women}}$, IberEval, and $\text{Waseem}_{\text{sexism}}$), race topic-specific (`Race Mixt`) datasets ($\text{HateEval}_{\text{immigrants}}$, and $\text{Waseem}_{\text{racism}}$), and a mixture of both topics (`Spec Mixt`).

ther of the two individually: this intuitively makes sense, as both separate topics can help cover different subsets of the topic-generic datasets, and should also help in learning manifestations of hate which exist at the intersection of both topics.

Looking at the scores obtained on the topic-specific test sets, we can observe that, unlike in the topic-generic generalization experiments, both the one-topic mixtures and `Spec Mixt` appear to yield improvements over the intra-dataset models, for most of the models. For `Gndr Mixt` and `Race Mixt`, the improvements are, as expected, mostly found for the test sets of the datasets making up the mixtures, but not always: for example, with the T5-base model, training on `Gndr Mixt` appears to yield improvements on $\texttt{Waseem}_{\texttt{racism}}$ ($\sim +4$ F1). This seems to indicate a better ability to generalize in-domain from mixtures of topic-specific datasets, at least to other topic-specific datasets, but more experiments with more topic-specific datasets, covering a wider range of topics, would be necessary to determine whether this may also apply to topic-generic datasets.

6 Finer-Grained Topics Analysis

To better understand which manifestations of HS can be generalized to out-of-domain data, we then perform a finer-grained analysis of the topics learned by our models, by relying on an implicit hate speech dataset, for which fine-grained target annotations are available.

6.1 Implicit Hate Speech Dataset

IMPLICIT HATE CORPUS (which we refer to as `ElSherief`, for brevity) corresponds to the dataset proposed by ElSherief et al. (2021), which consists of 21,480 English tweets annotated for (in a first stage) the presence of implicit or explicit HS (or neither), as mutually exclusive classes. Further, for each of the tweets containing implicit HS, two annotators supplied the targeted demographic groups and the implied statement of the hateful messages, both as free-form texts. In our experiments, we rely on the 6,196 implicit hate instances and their target annotations, which we have manually grouped into different sets of finer-grained topics of HS (whose sizes are in Table 5): *Islam*, *Black People*, *Immigrants / Refugees*, *Beliefs / Religion*, *Gender*, *LGBTQIA+*, *Unspecified Minorities*, and *Nationality*.

6.2 Results

To better show the observations that can be mainly linked to training dataset/mixture selection, we choose to present, in Table 5, the results of the best performing non-domain adapted architecture used here, T5-base. To this end, we rely on the previously mentioned trained T5-base models to obtain binary HS predictions on the entire processed `ElSherief` dataset, and compute the accuracy (as the tested instances are by definition all from the positive class) of the models for each of the finer-grained topic-specific subsets described previously.

The first three results columns show the topics learned by the models trained on the topic-generic datasets, `Davidson`, `Founta`, and their mixture (`Gene Mixt`). We observe that the model trained on `Founta` alone yields the highest accuracy scores over a number of topics, except for the *Immigrants/Refugees*, *Unspecified Minorities*, and *Gender* topics. Due to the latter's small size, and higher degree of co-occurring kinds of HS manifestations (see Section 6.3), accuracies appear to be relatively lower across the board, even for the gender topic-specific datasets. Still, topic-generic datasets, and their mixture, display fairly decent generalization on average for most fine-grained topics analyzed here, and represent a promising avenue for future research aimed at constructing effective out-of-domain generalization mixtures of datasets.

The next group of three columns show the differences in variety of finer-grained topics, between the gender and the race topic, and this in spite of fewer number of instances in the Race topic mixture compared to Gender one. Training on both topics combined (`Spec Mixt`) only yields improvement in the *Gender* and *LGBTQIA* topics, while all the other accuracies drop. This could be a sign that the dual-topic mixture allows for the learning of intersectional manifestations of HS between these topics. Indeed, in the *Gender* group, we find that 22 out of the 58 instances also belong to one or more other groups, and among those, 16 are in the *Muslims*, *Immigrants / Refugees*, or both groups (10 for *Immigrants / Refugees* and/or *Black People* in *LGBTQIA*). As such, it appears that mixtures of topic-specific datasets may prove useful in the generalization of intersectional HS detection, though more research would be necessary to verify this for other topics, and with more data.

The next two columns demonstrate how two datasets annotated for similar topics can wildly vary in which fine-grained topics they allow models to capture. Due to the significant differences in data collection, annotation, and cultural/temporal contexts, the `HatEval` and `Waseem` datasets result in very different learned distributions of fine-grained topics, with the latter almost exclusively covering the *Muslims* topics, whereas the former yields a more varied, and overall more performant on the `ElSherief` dataset, out-of-domain generalization.

The final group of three columns further illustrates this point: `Waseem` yields significantly lower performance on the *Gender* topic, and, if removed from the Gender topic mixture (`Gndr Mixt`), improves its overall accuracy. While removing the entirety of a dataset might not be ideal, this shows the importance of data selection for both in and out-of-domain generalization, and we believe this kind of analysis could be used to guide data selection, for the purpose of constructing such mixtures of datasets.

6.3 Error Analysis

To further investigate how these different models behave on those out-of-domain instances, we perform and present an error analysis on a selection of instances[1] from some of the different finer-grained topics.

We find that misclassifications are often due to several factors, among which implicit expressions of HS such as stereotypes, irony, or humor (as in (1)), or the presence of multiple topics of HS used in potentially ambiguous ways (2): is this instance (ironically) Islamophobic, sexist, or both?

(1) *what pronoun should we use to refer to <u>people who are transitioning into martians</u> ?*

(2) *<u>women are intellectually deficient</u>, islam says*

---

[1] These are examples taken from the ElSherief public dataset which provides texts of the tweets (because of Twitter otherwise having deleted a lot of them).

Model		T5-base												
Train / Test	Size	Gene Davi	Gene Fnta	Gene Mixt	Gndr Mixt	Race Mixt	Spec Mixt	Race HatE	Race Wasm	Gndr Iber	Gndr Wasm	Gndr Mixt\Wasm	All	*ES*
Muslims	801	35.58	**81.90**	56.67	1.13	63.17	41.57	70.04	39.95	2.50	0.37	5.36	52.06	94.51
Black People	1391	78.43	**78.58**	54.71	2.80	34.94	23.22	52.41	2.44	3.16	0.93	9.78	50.75	88.79
Immigrants / Refugees	1386	24.82	47.40	26.19	2.45	53.03	42.14	**66.52**	1.30	2.45	0.51	9.95	50.36	96.90
Beliefs / Religion	652	63.03	**76.99**	53.53	1.68	35.74	28.53	51.69	6.74	2.76	0.16	6.44	53.99	93.25
Gender	58	31.03	41.38	24.14	32.76	17.24	46.55	25.86	5.17	44.83	12.07	46.55	**48.28**	96.55
LGBTQIA	87	43.68	**63.22**	33.33	3.45	12.64	21.84	37.93	0.00	12.64	2.30	14.94	32.18	88.51
Unsp. Minorities	1147	**79.42**	66.61	40.45	1.22	23.45	12.55	44.90	2.01	1.48	0.17	6.02	38.10	85.44
Nationality	203	50.24	**67.98**	44.83	1.97	34.97	24.14	56.16	3.45	4.43	0.49	7.39	40.39	84.24

Table 5: Accuracy scores obtained by the T5-base models trained on the datasets and mixtures of datasets (columns), evaluated on each of finer-grained topic of HS grouped from the ElSherief dataset (rows), with the number of instances of each topic reported in the Size column. The best accuracies for each topic are highlighted in **bold**, and the scores are gradient-colored for legibility. `Gndr Mixt\Wasm` corresponds to the Gender topic mixture (`Gndr Mixt`), from which Waseem$_{\text{sexism}}$ has been removed. **All** corresponds to the mixture of all 8 individual datasets (see Table 1). ***ES*** corresponds to baseline accuracy scores obtained from training a similar model on the ElSherief dataset itself, for comparison.

in sahih bukhari 1:6:301. disagree? you are islamophobe.

Both (3) and (4) were misclassified due to the lack of context: neither of the two implied statements provided by the annotators (i.e., *Women are dumb/Muslims are repugnant* and *kill gays/immigrants should be shot like rats*) can be reliably automatically inferred without more information.

(3) *oh honey. justoh, honey. bless your heart.*

(4) *maybe time to oil up the deer rifles, the ones with the hd night scopes to keep the "vermin" population down.*

7 Conclusion and Perspectives

In this work, we have proposed a set of topic-oriented analyses of the generalizability of HS datasets. We have shown how *topic-generic* and *topic-specific* datasets yield different degrees and nature of generalization, both when used individually, or as mixtures of datasets. With the former, we found that, while not very successful at generalizing in-domain to topic-specific datasets, the use of mixtures allows smoothing out individual weaknesses. With topic-specific datasets, we found that generalization is possible, both for single-topic and multi-topic mixtures. Through a finer-grain out-of-domain generalization analysis, we showed how a priori somewhat similar datasets can vary wildly in the forms of *implicit* HS that can be learned from them. Implicit expressions of hate, with few lexical features, are more difficult to generalize, as models can struggle to capture underlying hateful intents in messages. Notably, one barrier to understanding more implicit manifestations of hate is the lack of context for individual social media posts: more conversational datasets could represent an interesting avenue of research in that regard. These are all important considerations for the purpose of constructing more reliable automatic HS detection systems, intended to function on raw, potentially noisy, and never-seen-before data. In particular, we found that topic-generic datasets like `Founta` (Founta et al., 2018) appear promising for future research on generalization-optimized mixtures of datasets. Finally, concerning HS detection architectures, we found that, for BERT-like models, pre-domain adapted variants generalize slightly better than a more generic RoBERTa classifier, but yield similar results to a generic text-to-text T5 architecture, which seems promising for future research in HS detection.

In future work, we will explore the use of these approaches to guide data selection, by for example employing the method proposed by Swayamdipta et al. (2020), in order to construct mixtures of HS datasets better suited to out-of-domain generalization. Alternatively, Active Learning methods have been shown to be successful in NLP (Ein-Dor et al., 2020), and could be used to palliate annotated scarcity in HS detection: by using topic-oriented analysis methods to detect difficult-to-predict topics, better targeted additional annotations could be acquired, to help improve the topical coverage of automated systems.

Limitations

In this work, we acknowledge a number of issues with the compatibility of HS datasets (cf. Sections 1, 2, and 3.2): namely, the phenomena annotated in these datasets, even those labelled using similar or equal terms, will, in theory and in practice, represent wildly different classes. While, unlike some other previous works (Fortuna et al., 2020, 2021), we do not merge initially distinct labels in datasets, but instead only keep whole classes that correspond to hate speech (excluding other forms of abuse, like *toxicity*, *offensiveness*, etc.), using them in a unified binarized setting is still bound to introduce significant amounts of noise in the training data, with regards to the HS detection task. Most of these issues are well documented, however, as complete re-annotation of all relevant data would be a prohibitively expensive enterprise, we believe there is value in exploring alternative solutions that may enable generalization despite these problems, using existing annotated datasets as they are currently available. Additionally, the analyses presented in this work are by no means comprehensive, both in quantity and variety of datasets and models experimented with. For example, some of the fined-grain topic groups found in the `ElSherief` dataset are not large enough to draw strong conclusions from (namely, *Gender* and *LGBTQIA*): supplementing these less represented topics with more data would enable better insights into generalizability for these kinds of HS manifestations.

Ethics Statement

The data that was used for conducting the experiments is composed of text from the public domain taken from datasets publicly available to the research community. These corpora also conform to the Twitter Developer Agreement and Policy that allows unlimited distribution of the numeric identification number of each tweet. The desire to combat online HS and prevent the widespreading of stereotypes cannot be done without automatic moderation tools, at the risk of increasing cases of algorithmic discrimination. However, the deployment of such algorithms should be done with care, as algorithmic discrimination results from the introduction of biases at the time of the design of the system. These biases consist in the transposition of general (often stereotyped) or statistical observations into systematic algorithmic conditions.

Acknowledgements

This work has been carried out in the framework of the STERHEOTYPES project funded by the Compagnia San Paolo 'Challenges for Europe'. The research of Farah Benamara is also partially supported by DesCartes: The National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) program.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for Abusive Language Detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Patricia Chiril, Endang Wahyu Pamungkas, Farah Benamara, Véronique Moriceau, and Viviana Patti. 2022. Emotionally Informed Hate Speech Detection: A Multi-target Perspective. *Cognitive Computation*, 14(1):322–352.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't Take the Easy Way Out: Ensemble Based Methods for Avoiding Known Dataset Biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082, Hong Kong, China. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the International AAAI Conference on Web and Social Media*, 11(1):512–515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent Hatred: A Benchmark for Understanding Implicit Hate Speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Karmen Erjavec and Melita Poler Kovačič. 2012. "You Don't Understand, This is a New War!" Analysis of Hate Speech in News Web Sites' Comments. *Mass Communication and Society*, 15(6):899–920.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the Evalita 2018 Task on Automatic Misogyny Identification (AMI). In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the Task on Automatic Misogyny Identification at IberEval 2018. In *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) co-located with 34th Conference of the Spanish Society for Natural Language Processing (SEPLN 2018), Sevilla, Spain, September 18th, 2018*, volume 2150 of *CEUR Workshop Proceedings*, pages 214–228. CEUR-WS.org.

Paula Fortuna, Ilaria Bonavita, and Sérgio Nunes. 2018. Merging datasets for hate speech classification in Italian. In *EVALITA@ CLiC-it*.

Paula Fortuna and Sérgio Nunes. 2018. A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys*, 51(4):85:1–85:30.

Paula Fortuna, Juan Soler, and Leo Wanner. 2020. Toxic, Hateful, Offensive or Abusive? What Are We Really Classifying? An Empirical Analysis of Hate Speech Datasets. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6786–6794, Marseille, France. European Language Resources Association.

Paula Fortuna, Juan Soler-Company, and Leo Wanner. 2021. How well do hate speech, toxicity, abusive and offensive language classification models generalize across datasets? *Information Processing & Management*, 58(3):102524.

Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

Surya Kallumadi, Srijan Kumar, and Diyi Yang. 2020. ICWSM Data Challenge. In *In Proceedings of the Fourteenth International Conference on Web and Social Media (ICWSM)*. AAAI Organization.

Prashant Kapil and Asif Ekbal. 2020. A deep neural network based multi-task learning approach to hate speech detection. *Knowledge-Based Systems*, 210:106458.

Mladen Karan and Jan Šnajder. 2018. Cross-domain detection of abusive language online. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 132–137, Brussels, Belgium. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *arXiv:1711.05101 [cs, math]*.

Florian Ludwig, Klara Dolos, Torsten Zesch, and Eleanor Hobley. 2022. Improving generalization of hate speech detection systems to novel target groups via domain adaptation. In *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*, pages 29–39, Seattle, Washington (Hybrid). Association for Computational Linguistics.

Kosisochukwu Madukwe, Xiaoying Gao, and Bing Xue. 2020. In Data We Trust: A Critical Analysis of Hate Speech Detection Datasets. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 150–161, Online. Association for Computational Linguistics.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in discriminating profanity from hate speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30(2):187–202.

Isar Nejadgholi and Svetlana Kiritchenko. 2020. On Cross-Dataset Generalization in Automatic Detection of Online Abuse. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 173–183, Online. Association for Computational Linguistics.

Fabio Poletto, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. 2021. Resources and benchmark corpora for hate speech detection: A systematic review. *Language Resources and Evaluation*, 55(2):477–523.

Jing Qian, Mai ElSherief, Elizabeth Belding, and William Yang Wang. 2019. Learning to Decipher Hate Symbols. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3006–3015, Minneapolis, Minnesota. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Niloofar Safi Samghabadi, Parth Patwa, Srinivas PYKL, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and Misogyny Detection using BERT: A Multi-Task Approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131, Marseille, France. European Language Resources Association (ELRA).

Joni Salminen, Maximilian Hopf, Shammur A. Chowdhury, Soon-gyo Jung, Hind Almerekhi, and Bernard J. Jansen. 2020. Developing an online hate classifier for multiple social media platforms. *Human-centric Computing and Information Sciences*, 10(1):1.

Diptanu Sarkar, Marcos Zampieri, Tharindu Ranasinghe, and Alexander Ororbia. 2021. fBERT: A Neural Transformer for Identifying Offensive Content. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1792–1798, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection using Natural Language Processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, Valencia, Spain. Association for Computational Linguistics.

Steve Durairaj Swamy, Anupam Jamatia, and Björn Gambäck. 2019. Studying Generalisability across Abusive Language Detection Datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 940–950, Hong Kong, China. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Zeerak Talat and Dirk Hovy. 2016. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Zeerak Talat, James Thorne, and Joachim Bingel. 2018. Bridging the Gaps: Multi Task Learning for Domain Transfer of Hate Speech Detection. In Jennifer Golbeck, editor, *Online Harassment*, Human–Computer Interaction Series, pages 29–55. Springer International Publishing, Cham.

Cagri Toraman, Furkan Şahinuç, and Eyup Yilmaz. 2022. Large-Scale Hate Speech Detection with Cross-Domain Transfer. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2215–2225, Marseille, France. European Language Resources Association.

Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. 2018. Challenges for Toxic Comment Classification: An In-Depth Error Analysis. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 33–42, Brussels, Belgium. Association for Computational Linguistics.

Bertie Vidgen, Alex Harris, Dong Nguyen, Rebekah Tromble, Scott Hale, and Helen Margetts. 2019. Challenges and frontiers in abusive content detection. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 80–93, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wenjie Yin and Arkaitz Zubiaga. 2021. Towards generalisable hate speech detection: A review on obstacles and solutions. *PeerJ Computer Science*, 7:e598.

Xuhui Zhou, Maarten Sap, Swabha Swayamdipta, Yejin Choi, and Noah Smith. 2021. Challenges in Automated Debiasing for Toxic Language Detection. In *Proceedings of the 16th Conference of the European*

Chapter of the Association for Computational Linguistics: Main Volume, pages 3143–3155, Online. Association for Computational Linguistics.

A Experimental Parameters

In each cross-topic/cross-dataset experiment, we train one of the classifiers models described in Section 4 on either a specific dataset, or on a mixture of datasets (see Table 1). For all models, we use the Hugging Face transformer library's implementation of the AdamW optimizer (Loshchilov and Hutter, 2019), with the default 5×10^{-5} learning rate for all models, except for T5-base (1×10^{-4}) and ToxDectRoBERTa (1×10^{-5}), as recommended by their respective authors. When possible using our available hardware (multiple Nvidia GTX 1080 Ti and RTX 2080 GPUs), we use "effective" batch sizes of 64 instances (either as proper mini-batches, or by using gradient accumulation alongside smaller mini-batch sizes). When using mixtures of datasets, batches are constructed by sampling each component dataset proportionally to its relative size, to avoid oversampling the smaller datasets or subsampling the larger ones. Models are trained for a maximum of 8 epochs, with early stopping according to the validation loss, and only the best model checkpoints are retained, according to the validation macro F1 score. Average runtimes vary between architectures and datasets, with the BERT-like (fBERT, HateBERT, and RoBERTa-base) taking the least time (less than an hour on the largest mixture of datasets, `All`, combining all 8 datasets detailed in Table 1), and the longest being ToxDectRoBERTa, since it is based on the RoBERTa-large architecture (approximately 6 hours of training for the `All` mixture). The smallest architectures in number of parameters used in this study are those derived from BERT-base, with $\sim$ 110 million parameters, followed by RoBERTa-base ($\sim$ 125 million parameters), T5-base (220 million parameters), and finally ToxDectRoBERTa, based on RoBERTa-large, with 355 million parameters. For data preprocessing, we replace all emojis with their text form descriptions using the Python `emoji` library, and replace all "@" user mentions and URLs with the replacement strings "*[USER]*" and "*[URL]*" respectively.

Train / Test	Gene Davi	Gene Fnta	Gndr Evit	Gndr HatE	Gndr Iber	Gndr Wasm	Gndr HatE	Gndr Wasm
T5-base								
Gene Davi	92.70	82.64	69.32	68.79	63.34	61.80	59.51	43.28
Gene Fnt	73.02	81.08	59.82	59.52	59.16	54.58	60.22	51.77
Gndr Evit	63.29	58.07	67.00	78.40	66.89	58.00	52.97	34.62
Gndr HatE	42.21	42.90	44.55	46.29	34.09	49.62	54.28	36.58
Gndr Iber	57.76	74.22	**93.46**	92.02	84.86	74.15	79.59	37.90
Gndr Wasm	54.06	59.07	63.65	68.92	67.90	85.81	51.00	42.46
Race HatE	43.75	64.54	42.14	43.07	37.59	41.29	38.73	41.09
Race Wasm	56.66	72.73	49.59	48.05	46.68	47.67	66.64	87.03
ToxDectRoBERTa								
Gene Davi	**93.78**	84.73	73.74	70.36	74.33	69.57	**81.34**	60.76
Gene Fnt	81.42	**85.92**	72.81	67.81	74.96	60.89	78.97	61.90
Gndr Evit	62.10	54.25	67.37	82.08	63.08	64.69	59.10	45.69
Gndr HatE	40.30	39.81	45.63	51.65	32.88	47.04	41.59	50.67
Gndr Iber	64.14	73.00	91.25	92.07	83.89	74.35	68.65	38.93
Gndr Wasm	51.13	56.44	67.74	68.82	67.69	86.27	49.40	43.95
Race HatE	52.23	68.58	57.57	53.00	47.86	37.04	40.75	63.48
Race Wasm	69.92	73.47	68.07	62.16	64.55	46.50	73.59	86.29
fBERT								
Gene Davi	92.46	85.59	64.64	66.18	67.33	73.69	64.46	42.88
Gene Fnt	76.47	80.98	58.23	59.32	61.04	60.43	62.45	51.57
Gndr Evit	61.44	57.00	73.12	88.66	69.40	63.19	52.79	34.48
Gndr HatE	38.25	49.75	64.34	64.28	43.62	50.30	55.19	36.58
Gndr Iber	60.34	71.43	93.01	**93.02**	84.55	76.99	52.80	38.66
Gndr Wasm	53.84	61.47	67.03	64.38	72.60	86.28	47.52	42.99
Race HatE	50.65	66.49	39.31	39.74	37.56	38.95	41.72	39.73
Race Wasm	70.16	72.33	48.06	46.64	47.80	47.53	74.03	**87.53**
HateBERT								
Gene Davi	93.14	85.43	63.89	65.02	60.67	70.70	61.63	45.68
Gene Fnt	73.02	79.41	57.88	58.19	57.47	59.74	59.25	52.59
Gndr Evit	63.92	56.74	71.02	87.98	68.99	64.06	47.37	34.71
Gndr HatE	44.46	55.76	53.70	59.11	50.95	59.43	53.59	36.58
Gndr Iber	60.12	67.11	92.70	92.71	86.41	68.08	63.74	37.90
Gndr Wasm	52.36	58.24	74.26	68.54	71.20	86.05	50.83	43.01
Race HatE	44.13	66.08	43.79	43.60	37.11	36.89	38.53	39.26
Race Wasm	63.01	72.81	49.96	47.77	48.03	45.92	71.60	86.37
RoBERTa-base								
Gene Davi	93.30	82.50	66.12	64.43	63.00	68.02	67.35	46.57
Gene Fnt	74.50	80.84	59.21	56.56	58.06	55.65	61.92	53.28
Gndr Evit	60.68	58.51	69.58	86.77	60.53	63.18	45.80	35.92
Gndr HatE	46.39	55.17	50.27	56.88	42.64	47.22	53.31	36.96
Gndr Iber	60.50	62.96	92.43	92.61	**88.76**	65.55	74.30	37.90
Gndr Wasm	51.55	51.11	71.14	68.11	74.93	**87.11**	53.06	43.70
Race HatE	47.79	56.76	43.30	40.56	37.86	37.10	38.24	48.65
Race Wasm	56.89	75.52	49.67	47.58	49.08	46.87	70.09	86.27

Table A: Detailed results of learning from individual datasets (in terms of Macro F1-scores).

End-to-end Case-Based Reasoning for Commonsense Knowledge Base Completion

Zonglin Yang[♣*] **Xinya Du**[♠*] **Erik Cambria**[♣] **Claire Cardie**[♦]
♣ Nanyang Technological University
♠ University of Texas at Dallas
♦ Cornell University
`{zonglin.yang, cambria}@ntu.edu.sg`
`xinya.du@utdallas.edu`
`cardie@cs.cornell.edu`

Abstract

Pretrained language models have been shown to store knowledge in their parameters and have achieved reasonable performance in commonsense knowledge base completion (CKBC) tasks. However, CKBC is knowledge-intensive and it is reported that pretrained language models' performance in knowledge-intensive tasks are limited because of their incapability of accessing and manipulating knowledge. As a result, we hypothesize that providing retrieved passages that contain relevant knowledge as additional input to the CKBC task will improve performance. In particular, we draw insights from Case-Based Reasoning (CBR) – which aims to solve a new problem by reasoning with retrieved relevant cases, and investigate the direct application of it to CKBC. On two benchmark datasets, we demonstrate through automatic and human evaluations that our **E**nd-to-end **C**ase-**B**ased **R**easoning **F**ramework (ECBRF) generates more valid knowledge than the state-of-the-art COMET model for CKBC in both the fully supervised and few-shot settings. From the perspective of CBR, our framework addresses a fundamental question on whether CBR methodology can be utilized to improve deep learning models.

1 Introduction

Commonsense knowledge helps humans navigate everyday situations seamlessly (Apperly, 2010) and is required for many intelligent scenarios (Davis and Marcus, 2015). To automatically enlarge the scale of commonsense knowledge base for the benefit of reducing labeling labor and expense, Knowledge Graph Completion (KGC) has become a hot research topic (Ji et al., 2022). The general KGC task is to expand existing knowledge graphs by using well-trained classifiers—they are trained with existing annotated samples and predict whether or not there is a relationship between two *existing* entities in a knowledge graph (Wang et al., 2017).

Subject	Relation	Object
hardware shop	at location	mall
world map	has property	draw with grid-lines
PersonX receives its reward	wants to	keep the prize
PersonX wins the big Jackpot	wants to	get its money

Table 1: Commonsense knowledge base tuples. Examples are from ConceptNet and Atomic.

Although KGC methods can automatically find unlabeled relationships, they are always classification or ranking tasks and are limited to *existing* entities in a knowledge graph and can't extend to new entities (Ji et al., 2022). To extend to new entities, COMET (Bosselut et al., 2019) proposes to use *text generation* for exploring and discovering new entities, which is called commonsense knowledge base completion (CKBC) task, utilizing the knowledge within the pretrained language models (PLM), which has been with process in recent years (Devlin et al., 2019; Radford et al., 2019). Specifically, COMET uses subject and relation as direct input to PLM and aims to *generate* objects, most of which are novel and unseen entities.

However, CKBC is knowledge-intensive, requiring wide-ranging and detailed knowledge; and it is reported that the ability of pretrained language models to access and precisely manipulate knowledge is limited (Lewis et al., 2020b). One potential solution to this is to provide "non-parametric knowledge" through additional input. Lewis et al. (2020b) and Guu et al. (2020), for example, have shown that by retrieving passages that contain knowledge relevant to the current task, performance can be improved. For CKBC, unfortunately, it might be especially difficult to find useful passages that contain relevant commonsense knowledge from the web due to a *reporting bias* (Gordon and Van Durme, 2013) in which people rarely express the obvious (i.e., commonsense knowledge).

An example of reporting bias from Table 1 is

---

[*] Contribution starts from their stay at Cornell.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3509–3522
May 2-6, 2023 ©2023 Association for Computational Linguistics

that people rarely say "when a person wins a big Jackpot, he/she will want to get its money" because it's too obvious and meaningless to say. Therefore, instead of retrieving passages from the web, we propose that benefits can still be gained by retrieving relevant knowledge from a "case base" of existing commonsense knowledge tuples[1] and using the retrieved knowledge as non-parametric knowledge (i.e., beyond that represented in the model parameters) to augment the current CKBC input example. In addition, to prevent ECBRF from overfitting to some commonly retrieved cases, we propose $random\ mask$ as a training strategy that randomly masks the retrieved cases during training, which functions similar to dropout (Srivastava et al., 2014) and further improves the performance of the framework. We also analyze several variations to better understand the process.

Although past attempts suggest that similar retrieval-based methods cannot improve the performance of CKBC (Wang et al., 2021), on two benchmark datasets, we demonstrate through automatic and human evaluations that our **E**nd-to-end **C**ase-**B**ased **R**easoning **F**ramework (ECBRF) [2] generates more valid and informative knowledge than (1) the state-of-the-art COMET model (Bosselut et al., 2019) for CKBC which employs no case retrieval, and (2) a baseline model that employs random case retrieval – on both fully supervised and few-shot settings. We also provide an analysis on why different conclusions are reached.

In addition, our framework draws insight from Case-Based Reasoning (CBR), and also has contributions to the CBR research. CBR is a subject in classical AI that solves a new problem by reusing the solutions of retrieved seen similar problems stored in the case base (Aamodt and Plaza, 1994). CBR's methodology has four steps — case retrieval, reuse, revise and retain. Past years of accomplishment in deep learning (DL) have led to enthusiasm in the CBR community to apply DL in the service of CBR. However, based on the observation that many challenges remain in DL where CBR has advantages (e.g. few-shot learning), some CBR researchers (Leake and Crandall, 2020) advocate using CBR to complement DL. However, past works on using CBR to complement DL only limit to shallow Neural Networks (NN) (Liao et al., 2018; Leake et al., 2021; Ye et al., 2021, 2022). The latest work even suggests that in many tasks NN itself outperforms CBR-complemented NN (Ye et al., 2022), which raises fundamental questions on whether CBR methodology is useful for DL.

Our work addresses this doubt by being the first to show a concrete implementation of the integration of the full methodology of CBR to PLM (as one typical model in DL) (we show in §6 that we simulate the third step in the methodology of CBR instead of actually implement it, since it requires huge human efforts) and show that the integration method can benefit from multiple steps in the methodology of CBR, and can lead to better performance over PLM itself in both fully supervised settings and few-shot settings on CKBC. Notably our proposed framework has a larger advantage in few-shot settings, where CBR methods typically have advantage. We also find that the generation of our framework is largely related to the retrieved case especially when they are similar, which exhibits strong case-based reasoning patterns. In addition, a detailed analysis of our framework from a CBR perspective is provided in §6.

Our contributions can be summarized as follows: (1) Drawing insights from CBR, we introduce a new end-to-end framework for CKBC task. We also propose training strategies that can better utilize the retrieved knowledge. (2) We conduct extensive experiments on the CKBC task in various settings (e.g. fully supervised and few-shot), and the results consistently demonstrate that our proposed framework achieves improvements over the state-of-the-art baseline methods. (3) From the perspective of the CBR community, whether CBR methodology can be used to improve DL models remains a fundamental research question. We address this doubt by being the first to show a concrete implementation of the integration of the full methodology of CBR to PLM, and showing that such integration can achieve better performance than single PLM. A thorough analysis of the integration from CBR perspective is also provided.

2 Related Work

Case-Based Reasoning CBR is a subject in classical AI which consists of 4 sub-processes in its methodology: *retrieve*, *reuse*, *revise* and *retain* (Aamodt and Plaza, 1994). Leake and Crandall (2020) advocate using CBR to complement the challenges in deep learning (e.g., few-shot learning). §A.4 provides more detailed related works

---

[1] Initialized with tuples from training set or external data.

[2] Code available at `https://github.com/ZonglinY/ECBRF_Case_Based_Reasoning_with_PLM.git`

relevant to this line. Specifically, our framework is inspired by Watson (1999)'s proposal that compared to CBR being described as an artificial intelligence technology , it is better to describe CBR as a methodology for problem solving, that may use any appropriate technology. Here we treat CBR as a methodology and deep learning as technology that uses CBR as the general high-level process and deep learning as components of the process.

Reasoning in NLP CBR could be seen as a type of analogical reasoning (Kolodner, 1997), and analogical reasoning belongs to inductive reasoning (Salmon, 1989). Inductive reasoning (Yang et al., 2022) is different from deductive reasoning (Clark et al., 2020) (both belong to logical reasoning) that the premise in inductive reasoning can not provide conclusive support to its conclusion.

Commonsense Knowledge Base Completion Here we mainly describe works that use text generation models for this task. Li et al. (2016) propose models to evaluate the full knowledge tuple rather than generate new knowledge. Saito et al. (2018) make an extension by proposing a joint model for the completion and generation of commonsense tuples. However, their work focuses on augmenting knowledge base completion model, rather than to increase coverage in commonsense knowledge base construction. Yao et al. (2019) and Malaviya et al. (2020) focus on link prediction and ranking of knowledge, which is a different task with our generative CKBC task. Sap et al. (2019) use LSTM (Hochreiter and Schmidhuber, 1997) to generate commonsense knowledge and Bosselut et al. (2019) further leverage pre-trained language models to generate commonsense knowledge. Gabriel et al. (2021) present the task of discourse-aware commonsense inference and proposes a memory-based model to generate commonsense knowledge that is more coherent with context. Wang et al. (2021) give an analysis on knowledge capacity, transferability, and induction of pre-trained language models to perform generalizable commonsense inference. Da et al. (2021) analyze the few-shot learning ability of pretrained language models for CKBC task. Unlike these works, we propose a model that can improve the performance of generative CKBC tasks in both fully supervised settings and few-shot settings.

Language Model Prompting First developed by the GPT series (Brown et al., 2020), retrieved data are used as augmented input to improve few-shot performance of remarkable large models. However, past research suggest that such in-context learning cannot improve the CKBC task (Wang et al., 2021), and we are the first to show how in-context learning is useful for CKBC. In addition, such large models are hard to obtain and Brown et al. (2020) do not explore the finetuning performance, neither do they explore the full CBR methodology's effect on PLM. Gao et al. (2021) use prompting and also incorporate demonstrations into context to improve few-shot performance. Their work, however, only focuses on classification tasks and regression tasks, which is different from the CKBC. Similar to our work, Das et al. (2021) use retrieved cases as prompt to improve the performance of PLM. However, they only focus on question answering task and do not integrate the full methodology of CBR, missing important steps such as retain.

3 Task Definition

In the generative CKBC task, a knowledge data instance is represented as a tuple of subject, relation, and object: (sub, rel, obj). All sub and obj are in natural language phrases (Figure 1). rel can be used as either a special token or the corresponding natural language phrases (Bosselut et al., 2019). Here we use rel as natural language phrases. The task is that given a pair of sub and rel, the goal is to *generate* the corresponding obj.

4 Methodology

We start by formalizing our framework as a retrieve-then-predict generative process. Then in §4.2, we describe our ECBRF's modules for the generative process in detail. Finally, we present a hybrid training strategy for better regularization.

Figure 1 describes our method. In the figure, "query" stands for a sub and rel pair which is used as input to ECBRF to generate obj. "Case Base" is initialized with knowledge triples from the training set. "Cases" means the retrieved knowledge triples from the "Case Base". "In-context demonstrations" stand for the retrieved cases that are used for input augmentation (concatenate with the query). The subject, relation, and object of the retrieved cases are sub-scripted with "r" (e.g., sub_r).

4.1 ECBRF's Generative Process

ECBRF takes x as input and learns a distribution $p(y|x)$ over possible outputs y. Here x consists of

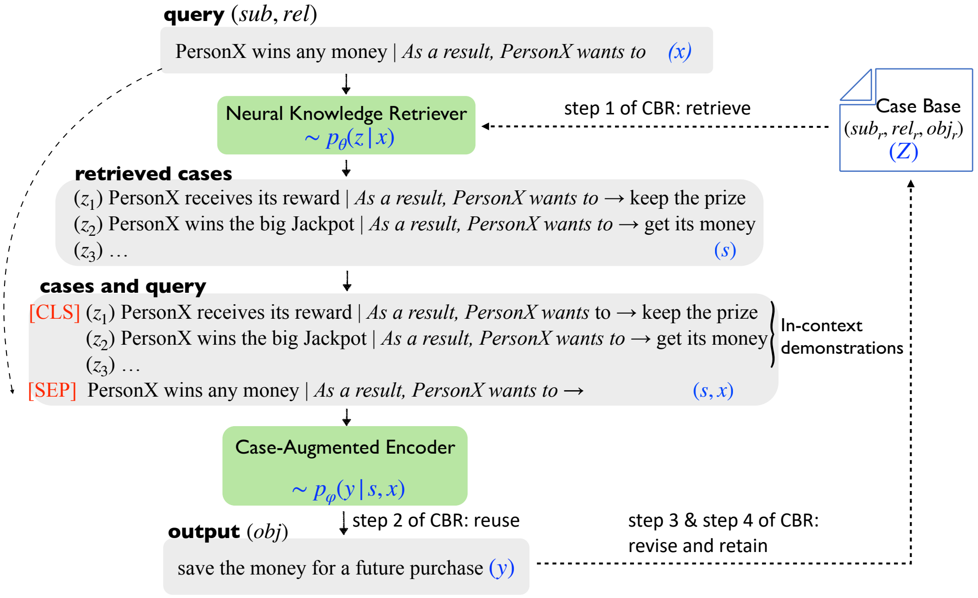


Figure 1: Our end-to-end case-based reasoning framework (ECBRF) for commonsense knowledge base completion. It involves all four steps of the CBR methodology (*retrieve*, *reuse*, *revise* and *retain*).

sub and rel, and y consists of obj. More specifically, ECBRF decomposes $p(y|x)$ into two steps: *retrieve* and *predict*. Given an input x, we first retrieve similar cases $z_1, z_2, ...$ (each case z_i consists of sub_r, rel_r and obj_r) from case base Z, while $(x,\ y) \notin z_i$. We model this as a sample from the distribution $p(z|x)$.

Then we use z_i in a number of m to compose a supporting set s (for each query, one supporting set is used for input augmentation). Specifically,

$$\hat{p}(s|x) = \sum_{z_i \in \text{top-m}(p(.|x))} p(z_i|x) \quad ((x, y) \notin z) \tag{1}$$

$$p(s_i|x) = \frac{\exp \hat{p}(s_i|x)}{\sum_j \exp \hat{p}(s_j|x)} \tag{2}$$

Then we condition on both the supporting set s and the query x to generate the output y, modeled as $p(y|s, x)$. To obtain the overall likelihood of generating y, we treat s as a latent variable and marginalize s via a top-k approximation, yielding:

$$p(y|x) = \sum_{s \in \text{top-k}(p_\theta(.|x))} p_\theta(s|x) p_\varphi(y|s, x) \tag{3}$$

4.2 Model Architecture

We now detail two key components – the *neural knowledge retriever*, which models $p_\theta(z|x)$; and *case-augmented encoder*, which models $p_\varphi(y|s, x)$.

Neural Knowledge Retriever The retriever uses max inner product search (MIPS) to retrieve z. Specifically, the retriever is defined using a dense inner product model:

$$p_\theta(z|x) = \frac{\exp f(x, z)}{\sum_{z'} \exp f(x, z')} \tag{4}$$

$$f(x, z) = Embed_{query}(x)^T Embed_{case}(z) \tag{5}$$

where $Embed_{query}$ is an embedding function that maps sub and rel in the query input to a $d-$dimensional vector, and $Embed_{case}$ is an embedding function that maps sub_r, rel_r and obj_r in the knowledge tuples in memory store to a $d-$dimensional vector. The relevance score $f(x, z)$ between x and z is defined as the inner product of the vector embeddings. The retrieval distribution is the softmax over relevance scores between top-k retrieved cases and current query input.

We implement the embedding functions $Embed_{query}$ and $Embed_{case}$ using two DPR-based models (Karpukhin et al., 2020). The input format for query x is the concatenation of subject and relation: [CLS] sub [SEP] rel [SEP]; And the input format for case z is the concatenation of the subject, relation, and object: [CLS] sub_r [SEP] rel_r obj_r [SEP].

Case-Augmented Encoder Given an input x and a supporting set s, the case-augmented encoder defines:

$$p_\varphi(y|s,x) = \prod_{i}^{N} p_\varphi(y_i|x,s,y_{1:i-1}) \qquad (6)$$

We use BART (Lewis et al., 2020a) and GPT-2 (Radford et al., 2019) as the base model for case-augmented encoder.

We also add prompts which we find is helpful. The input format (with prompts and in-context demonstrations) for case-augmented encoder is:

Here are some similar cases to infer from: z_0 z_1 ... z_{m-1} *From the similar cases we can infer that: [SEP] sub rel*

Zhao et al. (2021) show that pre-trained language model has "Recency Bias", which is the tendency to repeat answers that appear in the last in-context demonstration in classification tasks. We analyse this strategy for the generative CKBC task (we call it "*reverse demonstration*") that the most similar case from the retriever is placed as the last demonstration, the second most similar case in the second last demonstration, and so on.

4.3 Training Method

Since the purpose of in-context demonstration is only to provide ancillary information, the model should be able to predict obj w/ or w/o it. Therefore here we design a specific training strategy for ECBRF – during training, we *randomly mask* out in-context demonstrations and only keep the (sub, rel) query for some training examples with probability p_{mask}. It is designed to function similarly to dropout to prevent overly relying on retrieved cases.

5 Experiments & Analysis

In this section, we introduce the experiment datasets and evaluation details, as well as experiment setups and the experiment results, measured with automatic and human evaluations.

5.1 Datasets and statistics

We evaluate ECBRF using two automatic commonsense knowledge base completion benchmarks — ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019). In total, ConceptNet contains 101,800 tuples and ATOMIC contains 877,077 tuples. We use the same data split as COMET (Bosselut et al., 2019) did. In ATOMIC, around 17% of the labeled knowledge tuples use "None" as the object. As a result, models can easily get high performance by always generating "None". To better evaluate, we don't use knowledge tuples with the object being "None" for both training and evaluation. Apart from using the entire train set for training, we also conduct experiments in the few-shot settings — where the model is only trained with 5 to 320 knowledge tuples[3]. More details on data pre-processing is shown in §A.7.

5.2 Evaluation Details

For automatic evaluation metrics, following Bosselut et al. (2019) we use BLEU-2, perplexity, and *novelty* metrics (including %N/T-sro, %N/T-o, and %N/U-o). Specifically for *novelty* metrics, we report the proportion of all generated tuples that are novel tuple (%N/T-sro) (here novel means unseen in train set), have a novel obj (%N/T-o), and the proportion of the set of unique obj in all generated objects (%N/U-o).

In addition to automatic evaluation, we also perform human evaluation, including *validness*, *informativeness*, and *preference score*. For *validness* and *informativeness*, following Gabriel et al. (2021), the score is based on a 5-point Likert scale (with 5 points the highest score). For *validness*, following Gabriel et al. (2021), we judge the validness of the generated new knowledge by the likelihood of inferences based on a 5-point Likert scale (with 5 points the highest score). Specifically, obviously true (5), generally true (4), plausible (3), neutral or unclear or basically a repetition (sub-sentence) of the query (2), and doesn't make sense (1). For *informativeness*, the rating standard is also based on a 5-point Likert scale. Specifically, rich in relevant details (5), has relevant details (4), it seems some details are provided (3), basically a repetition (sub-sentence) of the query (2), unfinished generation (1). For *preference score*, We ask the human raters to *compare* the generations between ECBRF and COMET. Specifically, a valid generation with more information provided will be assigned 1.0 point, and a generation that is not valid or with less information will be assigned 0.0 instead. However, if the two generations perform comparably, both generations will be assigned 0.5 points.

Following Bosselut et al. (2019), for each experiment and for each model, we sample 100 generations for human evaluation. Each generation is rated by three graduate students. During the

---

[3] Note that for the few-shot settings, our ECBRF's case base is also initialized with 5 to 320 tuples

ConceptNet	5-shot	20-shot	40-shot	160-shot	320-shot	Full (100%)
COMET (GPT2)*	374.32 / 0.33	339.19 / 0.58	282.76 / 0.38	58.59 / 1.44	41.23 / 2.29	**13.04 / 3.37**
ECBRF (GPT2)	**284.67 / 0.51**	**220.07 / 0.59**	**102.46 / 1.63**	**52.10 / 1.61**	**38.63 / 2.59**	13.60 / 2.83
COMET (BART)*	14.26 / 1.15	11.31 / 1.64	9.48 / **3.70**	6.60 / 6.70	5.44 / 9.57	2.90 / 20.19
ECBRF (BART)	**12.85** / 1.20	**9.53** / 2.11	**8.70** / 3.17	6.19 / 6.20	5.22 / 9.31	2.93 / 18.55
w/ *random mask*	13.68 / **1.40**	9.82 / 2.02	8.98 / 2.96	**6.14 / 6.88**	**5.05 / 10.71**	**2.86** / 19.97
w/o *reverse demonstration*	13.02 / 1.20	9.60 / 1.84	8.74 / 2.95	6.21 / 6.13	5.06 / 10.12	2.92 / 19.80
w/ rand retrieval	13.23 / 1.00	10.15 / **2.61**	9.24 / 3.45	6.42 / 6.21	5.40 / 8.93	2.91 / **20.29**

ATOMIC	5-shot	20-shot	40-shot	160-shot	320-shot	Full (100%)
COMET (GPT2)*	753.93 / 2.11	512.11 / **3.44**	409.30 / **2.32**	209.78 / 2.73	165.28 / 2.68	67.95 / 4.00
ECBRF (GPT2)	**653.90 / 2.30**	**416.16** / 2.89	**319.12** / 2.26	**182.43 / 2.92**	**163.56 / 2.86**	**67.35 / 4.05**
COMET (BART)*	19.72 / **5.76**	16.83 / **5.38**	13.58 / 9.20	**14.30** / 11.56	**14.45** / 12.67	6.98 / **19.34**
ECBRF (BART)	18.17 / 5.16	14.73 / 3.85	13.05 / **10.13**	15.19 / 10.13	14.61 / 11.66	**6.95** / 19.06
w/ *random mask*	18.50 / 5.29	14.93 / 4.04	**13.01** / 7.47	14.52 / **12.42**	14.53 / 12.64	6.96 / 19.22
w/o *reverse demonstration*	**18.13** / 5.58	14.70 / 3.89	12.99 / 5.14	15.15 / 10.18	15.65 / 10.95	**6.95** / 19.24
w/ rand retrieval	18.16 / 4.33	**14.66** / 3.64	12.96 / 4.70	14.60 / 11.76	14.80 / **12.91**	6.96 / 19.02

Table 2: Perplexity (↓) / BLEU (↑) scores on ConceptNet (upper) and ATOMIC (down). The best scores for each setting are boldfaced. *: baseline models (our own implementation).

	20-shot (BART)	160-shot (BART)	Full (BART)
	ConceptNet		
COMET	0.37 / 1.76 / 1.74	0.42 / 2.86 / 2.69	0.47 / 3.87 / 3.49
ECBRF	**0.63 / 2.47 / 2.38**	**0.58 / 3.26 / 3.13**	**0.53 / 3.95 / 3.58**
	ATOMIC		
COMET	0.43 / 2.21 / 2.27	0.44 / 3.05 / 3.05	0.47 / 3.59 / 3.36
ECBRF	**0.57 / 2.44 / 2.58**	**0.56 / 3.22 / 3.17**	**0.53 / 3.64 / 3.43**

Table 3: Human evaluation results using *preference score*, *validness*, and *informativeness.*

sub: PersonX spends ___ working; *rel*: As a result, others feel
Ground truth: ['happy', 'happy to have x in their life']
COMET's generation: happy (BLEU: 31.62) ECBRF's generation: satisfied with personx's work (BLEU: 0.00)

Table 4: An example to show that BLEU is not a perfect metric for CKBC.

	20-shot (BART)	160-shot (BART)	Full (BART)
	ConceptNet		
COMET	98.00 / 13.46 / **58.99**	92.22 / 14.83 / 61.83	57.83 / **5.57** / 71.29
ECBRF	96.64 / **21.24** / 51.10	93.27 / **16.09 / 65.83**	**59.62** / 4.84 / **73.08**
	ATOMIC		
COMET	100.0 / 58.18 / 15.83	100.0 / 30.92 / 29.11	100.0 / **9.21 / 17.93**
ECBRF	100.0 / **82.16 / 22.79**	100.0 / **34.96 / 29.70**	100.0 / 6.49 / 15.37

Table 5: Novelty evaluation results using %N/T-sro, %N/T-o, and %N/U-o.

evaluation the order of the two generations to be compared are randomized for each selection, therefore human raters have no clue on which choice is associated with which model. More details about human evaluation can be found at §A.6.

5.3 Experimental Setup

Baselines We use COMET (Bosselut et al., 2019) as our baseline. COMET is originally implemented with GPT (Radford et al.), a pretrained language model as the base model and uses subject and relation as direct input and uses the generation result as object. Here we compare two versions of COMET, one is GPT-2 (Radford et al., 2019) based and another is BART (Lewis et al., 2020a) based. Both GPT-2 and BART are more powerful pretrained language models than GPT. We leave the details of hyperparameters in §A.1.

5.4 Main Results

Automatic evaluation results on BLEU-2 and perplexity are shown in Table 2. We present results with human evaluations in Table 3. Automatic evaluation of *novelty* is shown in Table 5. Table 2 shows that *random mask* is constantly helpful for ECBRF when the train set is equal or larger than 160. Therefore we adopt *random mask* for ECBRF when the train set is equal to or larger than 160 in Table 3 and Table 5.

Table 2 shows that regardless of the selection of base models, ECBRF consistently outperforms the COMET baseline in almost all perplexity measures and most BLEU measures. We argue that BLEU is not a perfect metric for CKBC (Sai et al.,

sub: PersonX wants to play with PersonY rel: Before, this person needed
One retrieved case by ECBRF: PersonX plays tennis with PersonY's friend, Before, this person needed, get a tennis racket
COMET's generation: to have a game ECBRF's generation: to find a tennis court

Table 6: An example to show how the retrieved cases influence the generated obj.

2023), since each sub and rel pair can lead to more than one feasible obj, and BLEU can only refer to a limited set of ground truth obj. Even if a model generates a reasonable obj, it may yield a low BLEU, because the generated obj is not in the ground truth set.

Table 4 shows one typical example from ATOMIC that shows although ECBRF's generations are reasonable, they only receive low BLEU scores (More examples in §A.2). Table 3 shows that ECBRF consistently outperforms COMET in *preference score*, *validness*, and *informativeness* in human evaluation, especially in few-shot setting. In practice, we observe that in few-shot setting, BART without retrieval tends to repeat the query during generation, while retrieved cases seem to be able to provide knowledge and guidance to generate more proper obj.

Table 5 shows that the generated obj of ECBRF are generally *novel* especially in few-shot settings. We empirically attribute the lower novelty of ECBRF in full train set to that ECBRF sometimes tends to copy proper retrieved obj as generation. The reason %N/T-sro score is always 100.0 in ATOMIC is that the (sub, rel) pairs in ATOMIC's train set and test set do not overlap.

We attribute the different conclusions reached on whether in-context demonstrations (ICD) with finetuning can be beneficial to CKBC (Wang et al., 2021) to that (1) ICD is more useful in few-shot settings, so that investigation on full train set setting might not discover this advantage; (2) human evaluation is the most precise metric for the task while BLEU is not so only evaluating with automatic metrics could not be precise; (3) ECBRF uses *random mask*, which is empirically found to be helpful in performance when using a large train set.

5.5 Ablation Study of ECBRF

In Table 2, we show some ablation studies of ECBRF. "w/ rand mask" stands for the ECBRF model using random mask ($p_{mask} = 30\%$); "w/o reverse demonstrations" stands for the ECBRF model without using reverse demonstrations; and "w/ rand retrieval" represents an ECBRF model that uses randomly searched cases instead of MIPS search.

Both tables for automatic evaluation show that using *random mask* can generally lead to better performance for ECBRF in both perplexity and BLEU when the training set is larger, while lead to worse performance when the number of training set is less than 160. Our interpretation is that, when the train set is very small, the model can benefit from over-relying to the retrieved cases; while when the train set is large, PLM can still benefit from the retrieved cases but the over-relying could be harmful.

The tables also show that "reverse demonstration" only leads to comparable performance, which might indicate that the order of retrieved cases does not make a difference in a generative task (as CKBC).

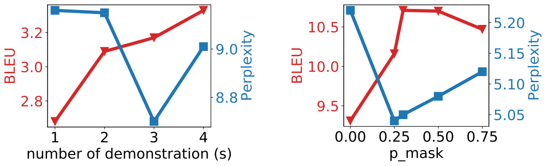
From the tables, we also observe that ECBRF with MIPS retrieval consistently leads to better performance than ECBRF with random retrieval in terms of perplexity in ConceptNet experiments, while performs comparably with ECBRF in ATOMIC experiments. Notice that the required generation in ConceptNet is usually shorter and more similar compared to ATOMIC. Therefore our interpretation is that, only when the retrieved cases are enough similar to the input query and its designed golden generation, can the retrieved cases significantly benefit the generation process (towards golden generation).

5.6 Qualitative Analysis on How Retrieved Cases Influence obj Generation

Table 6 shows one example of the generation of ECBRF and COMET (more examples are shown in §A.3). It shows that ECBRF's generation is related to the retrieved case, exhibiting the case-based reasoning ability of *reusing* the *retrieved* old experience to solve new problems.

6 Further Analysis from Perspective of CBR

CBR methodology contains 4 sub-processes, which are *retrieve*, *reuse*, *revise* and *retain*. More specifically, when given a new problem, the method first *retrieves* the most similar cases, then *reuses* the information in that case to solve the new problem by



(a) effect of # cases (b) effect of p_{mask}

Figure 2: ECBRF's performance (perplexity (↓) / BLEU (↑)) with regard to different numbers of demonstrations and different p_{mask}. Experiments on the left figure use ConceptNet 40-shot train set, and the right figure use ConceptNet 320-shot train set (since *random mask* is only helpful on large train set).

proposing a new solution, then *revises* the proposed solution according to the feedback of adopting it in real application scenarios (*revise* step usually involves human's effort), and finally select high quality revised solutions together with their problems as new cases to *retain* to case base.

We provide an analysis of how the high-level methodology of CBR (*retrieve*, *reuse*, *revise* and *retain*) shapes the design and how the selection details of CBR-related components improve the performance of our end-to-end DL framework.

Step 1: Retrieve *Retrieve* is an important step since the effectiveness of a CBR system largely relies on its ability to retrieve useful previous cases (Montazemi and Gupta, 1997). Here we use *neural knowledge retriever* (DPR) for retrieving the most similar cases. Table 2 shows the results of ECBRF using MIPS retrieval and random retrieval. As illustrated in §5.5, from the experimental results we hypothesize that ECBRF tends to make generations that are similar to the retrieved cases. This hypothesis is consistent with insights from CBR that the $retrieve$ step is essential for guiding the $reuse$ step. However, the difference lies in that CBR insights rely on $retrieve$ step more (with irrelevant retrieval it would be particularly hard for $reuse$), while PLM seems to be able to benefit from even random retrieval.

Step 2: Reuse Here we use *case-augmented encoder* to automatically *reuse* the retrieved cases.

Figure 2a shows the effects of number of retrieved cases. We observe that when *case-augmented encoder* uses 3 cases, it reaches the best perplexity, and nearly the best BLEU performance.

Figure 2b shows the effects of p_{mask}. Only when p_{mask} is 1.0, in-context demonstrations are not used at test time, which makes the model the same as COMET. As we gradually increase p_{mask}, perplexity keeps improving and BLEU-2 reaches the global maximum when p_{mask} is 0.3. It is also interesting to see empirically how the *case-augmented encoder* gradually learns to *reuse* the retrieved cases to increase the performance of the deep learning model as we gradually decrease p_{mask}.

	Perplexity	BLEU
ECBRF	8.70	3.17
w/ only obj_r	8.90	3.26
w/o prompt	9.01	3.44
w/ larger case base	**7.90**	**3.69**

Table 7: Ablation Study: effect of sub_r, prompt, and the retain step (perplexity (↓) / BLEU (↑)). Results of this table use ConceptNet 40-shot train set.

In CBR, the *reuse* of the retrieved case's solution contains two steps: (a) find the difference between the past and the current queries and (b) adapt the retrieved solution to the current query (Aamodt and Plaza, 1994). So it is important to know the difference between the past queries and the current input query for better adaptation. Table 7 shows the comparison between the result of only using obj_r (the retrieved cases' object phrases) as in-context demonstrations and the result of ECBRF (uses both sub_r, rel_r and obj_r), and we observe that ECBRF performs better in perplexity. but a little bit worse in BLEU. This result indicates that deep learning based *case-augmented encoder* is possible to automatically learn and reason from the difference between the past queries and the current input query for *reuse*. We leave further investigations on whether PLMs can learn to compare the difference between the past queries and the current input query as an open research question.

We use prompts to indicate the role of retrieved cases and current query in input. Table 7 shows that *case-augmented encoder* with the prompt performs better in perplexity while a bit worse on BLEU, indicating that usage of a prompt is possible to help the model better *reuse* the retrieved cases.

Step 3 & 4: Revise and Retain Since *revise* typically involves human efforts, here we simulate *revise* and *retain* and see their effect on our framework. The result of *revise* and *retain* is a larger case base with more high quality data, and the parameters of the model for *reuse* are not necessarily updated according to the new data. Here we simulate the effect of *revise* and *retain* by first training

ECBRF in a low-resource experiment (with a small case base), then at test time we expand the case base to the full train set. Table 7 shows that, at test time, ECBRF with access to a larger case base substantially outperforms ECBRF (with access to only a small case base), although the parameters have not been updated with the new data. This result demonstrates that our framework can benefit from CBR's methodology as *revise* and *retain*.

7 Conclusion

Drawing insights from CBR, we propose an end-to-end framework for the CKBC task. We demonstrate through automatic and human evaluations that our framework generates more valid knowledge than the state-of-the-art COMET model in both the fully supervised and few-shot settings. From the perspective of CBR, our framework addresses a fundamental question on whether CBR methodology can be utilized to improve deep learning models.

8 Future Works and Challenges

In general, we hope this work could provide some insights to bridge the two research areas, classic AI (Case-Based Reasoning) and deep learning based NLP methods together, and therefore to advance the research of both fields from each other's research developments.

From the aspect of NLP methods, for example, new prompting methods could be further developed based on insights from CBR research; The concept of *revise* and *retain* from CBR could be paid more attention to investigate their interaction with in-context demonstrations (prompting).

From the aspect of CBR, this work provides a tentative answer to the two long-remaining challenges — (1) whether CBR can be used to complement DL (Leake and Crandall, 2020), given that the latest work even suggests that in many tasks NN itself outperforms CBR-complemented NN (Ye et al., 2022); (2) the adaptation (*reuse* step) of previous cases to the current case is a very challenging problem, so that in many fields the CBR methodology is used only as a retriever (Choudhury and Begum, 2016). How to further answer these two questions could be a challenging research topic.

9 Limitations

From the perspective of CBR, we have shown through experiments that our framework can perform *retrieve* and *reuse* steps, and can benefit from *revise* and *retain* steps. But the *revise* step in CBR typically involves human efforts, and this paper does not focus on addressing this challenge. As a result, our framework might still need manual efforts to benefit from *revise* and *retain*.

However, human efforts could be more efficiently utilized for *revise* than writing new data from scratch. Since comparing with requesting the workers to write the knowledge from scratch, *revising* the existing generations of ECBRF could be much faster.

10 Acknowledgements

We thank Zefan Zhang and Jie Zheng for their participation and provision of human evaluation to this research work.

References

A. Aamodt and E. Plaza. 1994. Case-based reasoning:foundational issues,methodological variations,and system approaches. *AI communications*.

Ian Apperly. 2010. *Mindreaders: the cognitive basis of" theory of mind"*. Psychology Press.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *ACL 2019*. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Erik Cambria, Qian Liu, Sergio Decherchi, Frank Xing, and Kenneth Kwok. 2022. Senticnet 7: A commonsense-based neurosymbolic AI framework for explainable sentiment analysis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference, LREC 2022, Marseille, France, 20-25 June 2022*, pages 3829–3839. European Language Resources Association.

Nabanita Choudhury and Shahin Ara Begum. 2016. A survey on case-based reasoning in medicine. *International Journal of Advanced Computer Science and Applications*, 7(8).

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3882–3890. ijcai.org.

Jeff Da, Ronan Le Bras, Ximing Lu, Yejin Choi, and Antoine Bosselut. 2021. Understanding few-shot commonsense knowledge models. *AKBC*.

Rajarshi Das, Ameya Godbole, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2020a. A simple approach to case-based reasoning in knowledge bases. In *Automated Knowledge Base Construction*.

Rajarshi Das, Ameya Godbole, Nicholas Monath, Manzil Zaheer, and Andrew McCallum. 2020b. Probabilistic case-based reasoning for open-world knowledge graph completion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4752–4765.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *EMNLP 2021*, pages 9594–9611.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*.

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

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021. Paragraph-level commonsense transformers with recurrent memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12857–12865.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL 2021*, pages 3816–3830, Online. Association for Computational Linguistics.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 25–30.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and M-W Chang. 2020. Realm: Retrieval-augmented language model pre-training. *ICML*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP 2020*, pages 6769–6781, Online. Association for Computational Linguistics.

Mark T. Keane, Eoin M. Kenny, Mohammed Temraz, Derek Greene, and Barry Smyth. 2021. Twin systems for deepcbr: A menagerie of deep learning and case-based reasoning pairings for explanation and data augmentation. *CoRR*, abs/2104.14461.

Janet L Kolodner. 1997. Educational implications of analogy: A view from case-based reasoning. *American psychologist*, 52(1):57.

David Leake and David Crandall. 2020. On bringing case-based reasoning methodology to deep learning. In *International Conference on Case-Based Reasoning*. Springer.

David Leake, Xiaomeng Ye, and David J. Crandall. 2021. Supporting case-based reasoning with neural networks: An illustration for case adaptation. In *Proceedings of the AAAI 2021 Spring Symposium on Combining Machine Learning and Knowledge Engineering (AAAI-MAKE 2021), Stanford University, Palo Alto, California, USA, March 22-24, 2021*, volume 2846 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL 2020*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1445–1455, Berlin, Germany. Association for Computational Linguistics.

Chieh-Kang Liao, Alan Liu, and Yu-Sheng Chao. 2018. A machine learning approach to case adaptation. In *First IEEE International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2018, Laguna Hills, CA, USA, September 26-28, 2018*, pages 106–109. IEEE Computer Society.

Chaitanya Malaviya, Chandra Bhagavatula, Antoine Bosselut, and Yejin Choi. 2020. Commonsense knowledge base completion with structural and semantic context. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The*

Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 2925–2933. AAAI Press.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and WordNet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1222–1231.

Ali Reza Montazemi and Kalyan Moy Gupta. 1997. A framework for retrieval in cbr systems. *Annals of operations research*.

Feng Pan, Rutu Mulkar-Mehta, and Jerry R. Hobbs. 2011. Annotating and learning event durations in text. *Comput. Linguistics*, 37(4):727–752.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Ananya B. Sai, Akash Kumar Mohankumar, and Mitesh M. Khapra. 2023. A survey of evaluation metrics used for NLG systems. *ACM Comput. Surv.*, 55(2):26:1–26:39.

Itsumi Saito, Kyosuke Nishida, Hisako Asano, and Junji Tomita. 2018. Commonsense knowledge base completion and generation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 141–150, Brussels, Belgium. Association for Computational Linguistics.

Merrilee H Salmon. 1989. Introduction to logic and critical thinking.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Peifeng Wang, Filip Ilievski, Muhao Chen, and Xiang Ren. 2021. Do language models perform generalizable commonsense inference? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3681–3688, Online. Association for Computational Linguistics.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

Ian Watson. 1999. Case-based reasoning is a methodology not a technology. In *Research and Development in Expert Systems XV*. Springer.

Zonglin Yang, Li Dong, Xinya Du, Hao Cheng, Erik Cambria, Xiaodong Liu, Jianfeng Gao, and Furu Wei. 2022. Language models as inductive reasoners. *CoRR*, abs/2212.10923.

Zonglin Yang, Xinya Du, Alexander M. Rush, and Claire Cardie. 2020. Improving event duration prediction via time-aware pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3370–3378. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *CoRR*, abs/1909.03193.

Xiaomeng Ye, David Leake, and David J. Crandall. 2022. Case adaptation with neural networks: Capabilities and limitations. In *Case-Based Reasoning Research and Development - 30th International Conference, ICCBR 2022, Nancy, France, September 12-15, 2022, Proceedings*, volume 13405 of *Lecture Notes in Computer Science*, pages 143–158. Springer.

Xiaomeng Ye, David Leake, Vahid Jalali, and David J. Crandall. 2021. Learning adaptations for case-based classification: A neural network approach. In *Case-Based Reasoning Research and Development - 29th International Conference, ICCBR 2021, Salamanca, Spain, September 13-16, 2021, Proceedings*, volume 12877 of *Lecture Notes in Computer Science*, pages 279–293. Springer.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

A Appendix

A.1 Hyperparameters

We use a common training hyperparameters setup for PLM, and the variance of hyperparameters such as batch size is mainly determined by the computational resources.

Specifically, batch size is 32 for all ATOMIC experiments; is 16 for all BART-based ConceptNet experiments; is 8 for all GPT2-based ConceptNet experiments. Learning rate is 1e-5 for all experiments. The remainder of our training hyperparameters is the same as COMET (Bosselut et al., 2019) for full train set experiments. For few-shot experiments, we adapt the warm up steps accordingly (more details can be found in our public release code).

For decoding hyperparameters, we use greedy decoding for all BART experiments (since Bosselut et al. (2019) suggests that greedy decoding can lead to the best human evaluation results); and use topk (k=50) decoding for all GPT2 experiments (since with greedy decoding many generations of GPT2 model is only the end of text token; with randomly sampling a concrete generation could be made via multiple attempts).

A.2 Examples that BLEU not a Perfect Metric

Table 8 shows three examples with three different rel from ATOMIC that shows although ECBRF's generations are reasonable, but they only receive low BLEU scores.

This table shows that sometimes a good generation is assigned with a low BLEU score, especially when the generation is novel and unseen from the ground truth set. Table 5 shows that ECBRF produces more novel generations compared with the COMET baseline, which might make ECBRF suffer more from the imperfectness of the BLEU score.

A.3 Examples on How Retrieved Cases Influence ECBRF

Table 9 shows three examples of the generation of ECBRF and COMET.

This table shows that many ECBRF's generations are related to the retrieved case, exhibiting the case-based reasoning ability of *reusing* the *retrieved* old experience to solve new problems.

A.4 More Related Works on CBR

Leake and Crandall (2020) advocate using CBR to complement the challenges in deep learning (e.g., few-shot learning). However, past works on using CBR to complement DL only limit to shallow Neural Networks (NN) (Liao et al., 2018; Leake et al., 2021; Ye et al., 2021, 2022). The latest work even suggests that in many tasks NN itself outperforms CBR-complemented NN (Ye et al., 2022), which raises fundamental questions on whether CBR methodology is useful for DL. Other relevant works include only use CBR to improve the explainability of deep learning models (Keane et al., 2021), or use CBR to improve the performance of symbolic reasoning (Das et al., 2020a,b).

A.5 Future Applications of ECBRF

Although experiments in this paper are only conducted under the commonsense knowledge base completion task, ECBRF could potentially be utilized by other commonsense reasoning tasks, such as temporal commonsense reasoning (Yang et al., 2020), commonsense based sentiment analysis (Cambria et al., 2022), and metaphor processing (Mao et al., 2018) (similar to CBR, metaphor is also highly related to analogical reasoning).

A.6 Human Evaluation Details

Inter-annotator agreement Since we have three annotators for human evaluation, and the spearman correlation is to compare two rank lists, we calculate the averaged spearman correlation for each metric. Specifically, the average spearman correlation for "validness" is 0.71, and the average spearman correlation for "informativeness" is 0.66; For the "preference score", since it's to compare each ECBRF's generation with one baseline (COMET) generation, we can't rank generations according to the value of the preference score. So following Pan et al. (2011), we use Cohen's kappa coefficient instead. The kappa coefficient for "preference score" is 0.81.

Instructions for human evaluators There are 3 evaluation metrics: preference score, validness score and informativeness score.

Preference score: a valid generation with more information provided will be assigned 1.0 point, and a generation that is not valid or with less information with being assigned 0.0 instead. However, if the two generations perform comparably, both generations will be assigned 0.5 points.

Validness score: Validness score is given in a 5-point scale (1 5) obviously true (5) generally true (4) plausible (3) neutral or unclear or basically a repetition (sub-sentence) of the query (2) doesn't make sense (1). Examples for validness: (5): PersonX wants to learn piano, others want, to teach personX; (4): PersonX wants to learn piano, others want, to hear peronX play; (3): PersonX wants to learn piano, others want, to sell piano to peronX; (2): PersonX wants to learn piano, others want, to turn their music on; (1): PersonX wants to learn

Example 1	sub: PersonX spends ___ working rel: As a result, others feel ground truth: ['happy', 'happy to have x in their life'] COMET's generation: happy (BLEU score: 31.62) ECBRF's generation: satisfied with personx's work (BLEU score: 0.00)
Example 2	sub: PersonX expects another ___ rel: This person then ground truth: ['prepares themselves', 'gains knowledge'] COMET's generation: gains knowledge (BLEU score: 100.00) ECBRF's generation: wants to find out what it is that they are going to do next (BLEU score: 0.00)
Example 3	sub: PersonX spends the ___ working rel: This person is seen as ground truth: ['diligent', 'tired', 'hardworking'] COMET's generation: hardworking (BLEU score: 31.62) ECBRF's generation: dedicated (BLEU score: 0.00)

Table 8: Examples to show that BLEU is not a perfect metric for CKBC. This table shows that sometimes a good generation is assigned with a low BLEU score, especially when the generation is novel and unseen from the ground truth set. Table 5 shows that ECBRF produces more novel generations compared with the COMET baseline, which might make ECBRF suffer more from the imperfectness of the BLEU score.

Example 1	sub: PersonX wants to play with PersonY rel: Before, this person needed COMET's generation: to have a game Retrieved cases by ECBRF include: (PersonX plays tennis with PersonY's friend, Before, this person needed, get a tennis racket) ECBRF's generation: to find a tennis court
Example 2	sub: PersonX advances another ___ rel: This person then COMET's generation: PersonX gains knowledge Retrieved cases by ECBRF include: (PersonX marries the king's ___, This person then, he becomes king) ECBRF's generation: becomes more powerful
Example 3	sub: PersonX smiles broadly rel: As a result, this person wants to COMET's generation: to smile back Retrieved cases by ECBRF include: (PersonX grins like a cheshire cat, As a result, this person wants to, express their feelings) ECBRF's generation: to tell others about the good time they had

Table 9: Examples to show how the retrieved cases influence the generated obj. This table shows that many ECBRF's generations are related to the retrieved case, exhibiting the case-based reasoning ability of *reusing* the *retrieved* old experience to solve new problems.

piano, others want, to wait for the dinner.

Informativeness score: Informativeness score is given in a 5-point scale (1 5) rich in relevant details (5) has relevant details (4) it seems some details is provided (3) not related information or basically a repetition (sub-sentence) of the query (2) unfinished generation or doesn't make sense (1) Examples for informativeness: (5): PersonX wants to learn piano, others want, to teach personX the basic usage of piano and how to buy a suitable piano; (4): PersonX wants to learn piano, others want, to teach personX on how to use piano; (3): PersonX wants to learn piano, others want, to teach personX; (2): PersonX wants to learn piano, others want, to turn their music on; (1): PersonX wants to learn piano, others want, to teach.

A.7 Other Details on Data Pre-processing

We observe that in Table 2, our GPT experiment results are lower than Bosselut et al. (2019). We attribute the reason to our different data pre-processing method — we filter all the "." in the obj in ATOMIC and ConceptNet datasets since we observe that only a part of obj are equipped with ".", which might confuse the model on whether to generate "." or not.

Without “.”, a fixed token in generation, it would be harder for a model to reach higher perplexity and BLEU. In our preliminary experiments that do not especially filter “.”, our re-implemented COMET’s results are comparable to Bosselut et al. (2019).

A.8 Other Details on Retrieval

For ECBRF, we empirically find that not retrieving cases that share the same sub_r with sub is beneficial for the performance. The intuition behind this mechanism is that when one of the retrieved cases has the same sub_r with sub, then it should be natural to copy the corresponding obj_r as generation. However, it is common that obj is different from obj_r, which might confuse the model.

Exploring Segmentation Approaches for Neural Machine Translation of Code-Switched Egyptian Arabic-English Text

Marwa Gaser,[1] **Manuel Mager,**[2,*] **Injy Hamed,**[3,4]
Nizar Habash,[4] **Slim Abdennadher,**[1] **Ngoc Thang Vu**[3]
[1]The German University in Cairo, [2]AWS AI Labs
[3]University of Stuttgart, [4]New York University Abu Dhabi
`{marwa.saleh,slim.abdennadher}@guc.edu.eg`, `magerlm@amazon.com`,
`{injy.hamed,nizar.habash}@nyu.edu`, `thang.vu@ims.uni-stuttgart.de`

Abstract

Data sparsity is one of the main challenges posed by code-switching (CS), which is further exacerbated in the case of morphologically rich languages. For the task of machine translation (MT), morphological segmentation has proven successful in alleviating data sparsity in monolingual contexts; however, it has not been investigated for CS settings. In this paper, we study the effectiveness of different segmentation approaches on MT performance, covering morphology-based and frequency-based segmentation techniques. We experiment on MT from code-switched Arabic-English to English. We provide detailed analysis, examining a variety of conditions, such as data size and sentences with different degrees of CS. Empirical results show that morphology-aware segmenters perform the best in segmentation tasks but under-perform in MT. Nevertheless, we find that the choice of the segmentation setup to use for MT is highly dependent on the data size. For extreme low-resource scenarios, a combination of frequency and morphology-based segmentations is shown to perform the best. For more resourced settings, such a combination does not bring significant improvements over the use of frequency-based segmentation.

1 Introduction

Code-switching (CS), i.e. the alternation of language in text or speech, has been gaining worldwide popularity, due to several reasons, including globalization and immigration. While this has been met with a growing interest in the NLP field to build systems that can handle such mixed input, work on CS machine translation (MT) is still considered in its infancy, where only a few language pairs have been investigated (Sinha and Thakur, 2005; Dhar et al., 2018; Menacer et al., 2019; Xu and Yvon, 2021; Hamed et al., 2022c).

CS Sentence	it depends بصراحة بالنسبالي ع ال situation it depends *bSrAHp bAlnsbAly E Al* situation
Translation	for me it honestly depends on the situation

Word	Translation	Segmentation
it	it	it
depends	depends	depend#s
بصراحة *bSrAHp*	honestly	ب#صراحة *b#SrAHp*
بالنسبالي *bAlnsbAly*	for me	ب#النسبا#ل#ي *b#AlnsbA#ly*
ع *E*	on	ع *E*
ال *Al*	the	ال *Al*
situation	situation	situation

Figure 1: An example sentence with code-switching (CS) between English and Egyptian Arabic. The words are contrasted with their segmentations and English translations. Arabic words are paired with their transliterations in the Buckwalter scheme (Habash et al., 2007).

In this work, we focus on the CS Egyptian Arabic (EGY)-English (EN) language pair, as we observe its usage is becoming more common. Besides being prevalent amongst Egyptian migrant communities, it is also commonly used in Egypt due to the increase in international schooling systems and educational advancements. We identify three main challenges for CS MT. First is **data sparsity**, a challenge common to many CS language pairs because of limited parallel corpora containing commissioned translations of CS text (Çetinoğlu et al., 2016; Srivastava and Singh, 2020; Tarunesh et al., 2021; Hamed et al., 2022b; Chen et al., 2022). Second is Egyptian Arabic **morphological richness**, which further exacerbates the data sparsity situation (Habash et al., 2012a,b). Third, since the matrix language (EGY) is morphologically rich, CS occurs at three **CS levels**: on the boundaries of sentences (inter-sentential CS), between words (intra-sentential CS), and within words, i.e., morphological code-switching (MCS). This mix of types of CS raises the question of how to handle them all in the same system. These challenges are further illustrated in Figure 1.

---

[*] Work done while at the University of Stuttgart.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3523–3538
May 2-6, 2023 ©2023 Association for Computational Linguistics

A common solution to handle data sparsity for MT of morphologically rich languages is morphological segmentation (Oudah et al., 2019; Ataman et al., 2017; Grönroos et al., 2020). However, this has not been investigated for CS. In this paper, we explore a wide range of segmentation approaches, covering unsupervised morphology-based segmenters, unsupervised frequency-based segmenters, and supervised morphology-based segmenters. This work aims to answer the following research questions (RQs):

- **RQ1:** Which segmentation setup performs the best in the downstream MT task across different training sizes?
- **RQ2:** Does the effectiveness of the different segmenters in the MT task differ according to the CS type of the source sentence?
- **RQ3:** Is there a correlation between a more morphologically correct segmentation and MT performance?

While our results show that there is no correlation between correct morphological segmentation and MT performance, we find that the performance ranking between the MT systems varies across different training data sizes and sentence types (monolingual vs. code-switched). We show that applying a combination of supervised morphology-based and unsupervised frequency-based segmentations consistently gives best results, with statistical significance under low data sizes. While commonwisdom suggests that Byte-Pair Encoding (BPE) is the best approach, our experiments highlight the importance of integrating morphological knowledge in the case of extreme low-resource settings. We believe that the insights and methodology we follow will be useful to researchers working with low-resource languages. An additional contribution of our research is the creation of a gold standard morphologically annotated CS Egyptian Arabic-English dataset which we make publicly available.[1]

The paper is organized as follows. Section 2 discusses related work. Section 3 describes the dataset we annotated. Section 4 describes and evaluates the different segmenters used. Section 5 describes and evaluates the various MT systems. In Section 6, we answer our research questions.

2 Related Work

Several researchers have investigated the effect of applying different morphological and agnostic segmentation approaches on the MT performance for monolingual languages. Roest et al. (2020); Saleva and Lignos (2021) show that unsupervised morphology-based segmentation like Linguistically Motivated Vocabulary Reduction (LMVR) (Ataman et al., 2017), Morfessor (Smit et al., 2014), and FlatCat (Grönroos et al., 2014) for Nepali–English, Sinhala–English, Kazakh–English, and Inuktitut–English language pairs show either no improvement or no significant improvement over the agnostic BPE segmentation (Sennrich et al., 2016) in translation tasks. Meanwhile, Mager et al. (2022) and Ataman et al. (2017) show that for polysynthetic and highly agglutinative languages, unsupervised morphology-based segmentation outperforms BPEs (Sennrich et al., 2016) in MT tasks in both directions. Nevertheless, applying BPEs on top of morphology-based segmentation for Turkish–English, Uyghur–Chinese, and Arabic–English has shown to bring improvements over solely using BPEs or morphology-based segmentation for neural MT task (Pan et al., 2020; Tawfik et al., 2019). A similar result was achieved by (Ortega et al., 2020), using a morphological guided BPE for polysynthetic languages. However, Oudah et al. (2019) show that such an approach is beneficial in the case of statistical machine translation (SMT), and does not improve results for neural machine translation (NMT). For other natural language processing (NLP) tasks, Al-Thubaity and Al-Subaie (2015) show that utilizing word segmented Arabic dataset leads to improvements in text classification task over utilizing unsegmented dataset in terms of accuracy, precision, recall, and F-measure.

As for work on CS MT, there are many efforts (Sinha and Thakur, 2005; Dhar et al., 2018; Mahata et al., 2019; Menacer et al., 2019; Song et al., 2019; Tarunesh et al., 2021; Xu and Yvon, 2021; Chen et al., 2022; Hamed et al., 2022c). To the best of our knowledge, none of these efforts presented an extensive comparison covering different segmentation techniques. With regards to the languages covered, only Menacer et al. (2019) worked on CS Arabic-English. However, since they used carefully edited UN documents, the text only included the Modern Standard Arabic variety, and contained limited types of CS.

With regards to similar corpora, Balabel et al.

---

[1] `http://arzen.camel-lab.com/`

Case	Stem	Ending	Example
Irregular	modified	Irregular: es	monki+es
Irregular	modified	Regular: s,ed,ing,en	car+ing
Irregular	modified	Irregular: <nil>	went
Irregular	unmodified	Irregular: es	church+es
Regular	unmodified	Regular: s,ed,ing,en	car+s

Table 1: The ordered list of rules we follow to segment the English words.

(2020) annotated CS Egyptian Arabic-English data (Hamed et al., 2018) with tokenization (canonical segmentation), lemmatization, and POS tags. However, their corpus does not contain translations.

3 Data

3.1 Pre-existing Datasets

We use the *ArzEn* parallel corpus (Hamed et al., 2020, 2022b), which consists of speech transcriptions gathered through informal interviews with bilingual Egyptian Arabic-English speakers, as well as their English translations. The corpus consists of 6,213 sentences, where 4,154 (66.9%) are code-mixed, 1,865 (30.0%) are monolingual Arabic, and 194 (3.1%) are monolingual English. Among the code-mixed sentences, there are 1,781 (28.7%) sentences with morphological code-switching. We follow the predefined dataset splits, containing 3,341 (53.8%), 1,402 (22.6%), and 1,470 (23.7%) sentences for train, dev, and test sets, respectively. For training purposes, we also use 308k monolingual parallel sentences obtained from MADAR (Bouamor et al., 2018) and the following *LDC* corpora: (Gadalla et al., 1997; LDC, 2002b,a; Chen et al., 2017; Tracey et al., 2021; BBN Technologies et al., 2012; Chen et al., 2019). The preprocessing steps we apply are outlined in Appendix A. We use *ArzEn* train set as well as the monolingual parallel corpora to train both the segmenters and MT systems. For tuning and testing the MT systems, we use the *ArzEn* dev and test sets. For tuning and testing the segmenters, we annotated a new dataset, discussed next.

3.2 A New Dataset: ArzEn Surface Segmentation (ArzEnSEG) Corpus

To facilitate our research, we created a code-switched Egyptian Arabic-English morphologically annotated dataset which we use for tuning and testing. The dataset comprises the first 500 lines of *ArzEn* dev set. Unlike Balabel et al. (2020), we opt for surface form segmentation to allow for evaluating the segmenters. We also opt for extending *ArzEn* dataset as it contains translations and is used in our MT experiments.

	EGY	EN
Test Words	3,414	501
Dev Words	3,069	567
Total Words	6,483	1,068
Total Segmented Words	1,206	146
Total Morphs	7,911	1,214
Total Unique Morphs	1,192	432
% of Total Segmented Words	18.6%	13.7%
Morphs/Word	1.220	1.137
Maximum Morphs per Word	5	2

Table 2: Statistics on *ArzEnSEG* corpus.

For Arabic word segmentation, we use the Arabic Treebank (ATB) segmentation scheme (Maamouri et al., 2004; Habash, 2010). We choose this scheme as it is the standard tokenization scheme used in different treebanks (Maamouri et al., 2004, 2012; Taji et al., 2017; Habash et al., 2022). It has also shown to be efficient in Oudah et al. (2019) and has demonstrated its competitiveness in Habash et al. (2013).

For English word segmentation, we follow five rules in sequential order depending on whether the word has a regular or irregular stem and whether the word has a regular or irregular ending. Table 1 exhibits the five English rules we follow in order.

All annotation decisions were made in context by two bilingual speakers who collaborated on initial annotations and quality checks. Figure 1 presents an annotation example. We divide the sentences randomly into dev and test sets (250 sentences each). In Table 2, we display statistics about *ArzEnSEG*.

4 Segmentation Experiments

4.1 Experimental Setup

We explore three categories of segmenters: unsupervised morphology-based, unsupervised frequency-based, and supervised morphology-based segmentation. For the unsupervised morphology-based segmenters, we use MorphAGram in addition to three segmenters from the Morfessor family: Morfessor, LMVR, and FlatCat. For unsupervised frequency-based segmenters, we use BPE. Figure 2 summarizes the process of training these segmenters. For the supervised morphology-based segmenters, we use MADAMIRA (Pasha et al., 2014), where we exploit the segmentation schemes designed for Egyptian Arabic.

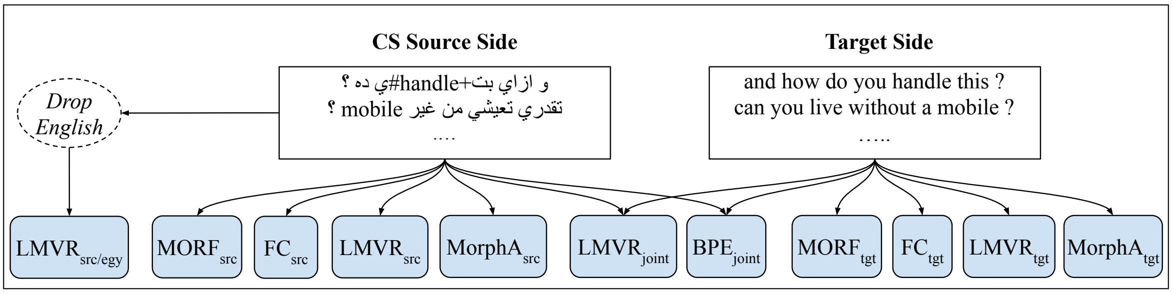


Figure 2: The unsupervised segmentation models we study in this paper and their training data dependencies. We use four systems: Morfessor (MORF), FlatCat (FC), LMVR, and MorphAGram (MorphA). The subscripts specify the training data: source (src), target (tgt), source+target (joint), and source without English, i.e., Egyptian, (src/egy).

4.2 Segmentation Systems

In this section, we introduce the segmentation systems used for the study. Details about the hyperparameter tuning for each system family can be found in Appendix B. The different segmentation models and their training dataset are displayed in Figure 2.

Morfessor Family We exploit three Morfessor family tools for unsupervised morphology-based segmentation in this research: Morfessor, (Smit et al., 2014), FlatCat (Grönroos et al., 2014), and LMVR (Ataman et al., 2017).

Morfessor is a morphological-based segmentation model which we train in an unsupervised manner. Three components form the system: the model, the cost function, and the training and decoding algorithms (Virpioja et al., 2013). The model is mainly concerned with the grammar and lexicon where the latter holds the attributes of the subwords and the grammar controls how these subwords are combined to form the word. Morfessor's grammar assumes that the subwords that form the word are independent of each other and that a word has at least one subword.

FlatCat is a variant of Morfessor which we also train in an unsupervised manner. Even though FlatCat builds on Morfessor and shares the same model component, they differ in their morphotactics (the set of rules that determine how the word's morphemes are arranged). FlatCat morphotactics is based on the Hidden Markov model (Baum and Petrie, 1966) which considers context. On the contrary, Morfessor's morphotactics algorithm is based on a unigram model which is not context-sensitive.

LMVR is a morphology-based segmenter that is built upon FlatCat and we train in an unsupervised manner. Nonetheless, LMVR takes into consideration the desired segmentation output vocabulary size during training.

For each tool, two models are generated; one trained on the source side; thus capable of segmenting CS data, and the other trained on the target side of the training data; thus capable of segmenting English data only. We add a *src* and *tgt* subscript to the segmenters' names to distinguish between both settings. Hence, MORF$_{src}$, FC$_{src}$, and LMVR$_{src}$ resemble Morfessor, FlatCat, and LMVR respectively, where the segmenters are trained on the source side. MORF$_{tgt}$, FC$_{tgt}$, and LMVR$_{tgt}$ resemble the segmenters trained on the target side.

MorphAGram We also include in this study the unsupervised morphology segmenter MorphAGram (Eskander et al., 2020) which is based on Adaptor Grammars. We use the *PrStSu+SM* grammar, which represents a word as a sequence of prefixes followed by a stem then a sequence of suffixes, in the unsupervised *Standard* learning setting to train the segmenters.

BPE The SentencePiece (Kudo and Richardson, 2018) implementation of BPE (Gage, 1994; Sennrich et al., 2016) is a frequency-based unsupervised segmenter. We train the BPE model jointly, on the concatenation of the source and target sides of the training parallel corpus. Previous work has shown that this approach is better suited for low resource settings (Guzmán et al., 2019). We refer to our joint BPE segmenter as BPE$_{joint}$.

MADAMIRA For supervised morphology-based segmenters, we use MADAMIRA's Egyptian Arabic model (Pasha et al., 2014), which was trained on the Egyptian Arabic Treebank (parts 1 through 6) (Maamouri et al., 2012). Specifically, we use MADAMIRA's *ATB_BWFORM* and *D3_BWFORM* schemes, henceforth MDMR$_{ATB}$ and MDMR$_{D3}$, respectively. Both schemes apply Alif/Ya normaliza-

	EMMA F1 Score		
Segmenter	**EGY**	**EN**	**All**
raw	0.806	0.953	0.838
MorphA$_{src}$	0.682	0.942	0.737
MORF$_{src}$	0.814	0.888	0.832
FC$_{src}$	0.821	0.961	0.851
LMVR$_{src}$	0.836	0.961	0.863
LMVR$_{src/egy}$	0.838	0.953	0.863
MorphA$_{tgt}$	0.806	0.953	0.838
MORF$_{tgt}$	0.147	0.951	0.327
FC$_{tgt}$	0.806	0.952	0.838
LMVR$_{tgt}$	0.806	0.966	0.842
LMVR$_{joint}$	0.841	0.963	**0.868**
BPE$_{joint}$	0.678	0.814	**0.707**
MDMR$_{ATB}$	0.935	0.953	**0.939**
MDMR$_{D3}$	0.868	0.953	0.887

Table 3: EMMA F1 score calculated on *ArzEnSEG* test set for the raw data as well as the segmented data using the different segmenters. The Arabic gold segmentation is based on the ATB segmentation scheme. We show the overall score (All) and language-specific scores calculated on the Egyptian Arabic (EGY) and English (EN) words separately. Segmenter names with a *src*, *tgt*, and *joint* subscripts represent segmenters that are trained on the source, target, and source+target sides respectively. The best performing segmenters from each category are highlighted in bold.

tion and segment the Arabic clitics. MDMR$_{D3}$ splits the Arabic definite article ال *Al* (the), while MDMR$_{ATB}$ does not.

4.3 Segmentation Results

To evaluate the performance of the segmenters, we use EMMA F1 score (Spiegler and Monson, 2010). Results in Table 3, reported on *ArzEnSEG* test set, show overall and language-specific scores.

Unsupervised morphology-based segmentation Results show that LMVR outperforms the other unsupervised morphology-based segmenters in terms of segmenting Arabic and English words. We perform further experiments where we train 2 additional models: i) a model trained jointly on the concatenation of the source and target sides of the parallel corpus, and ii) a model trained on the Arabic words only in the source side (where English words are dropped). Therefore, the former model is capable of segmenting both languages, while the latter is only tailored for segmenting Arabic words. We perform these experiments using LMVR, given that it outperforms the other segmenters. We refer to these models as LMVR$_{joint}$ and LMVR$_{src/egy}$ respectively, as outlined in Figure 2. Results show that joint training provides best EMMA scores.

Supervised morphology-based segmentation As shown in Table 3, both supervised morphology-based segmenters MDMR$_{ATB}$ and MDMR$_{D3}$ outperform all other segmenters. Their superiority in segmenting Arabic is expected, as they are trained on human-annotated data and hence are capable of generating infrequent morphemes. Additionally, MADAMIRA has a morphological analyzer embedded in it, which in turn enriches the inspection of Arabic words prior to segmentation. Higher EMMA scores are reported for MDMR$_{ATB}$ over MDMR$_{D3}$, which is also expected, as *ArzEnSEG* is segmented following the ATB scheme.

Unsupervised frequency-based segmentation As expected, BPE$_{joint}$ performs the worst in the morphology-based segmentation task, as it is designed for agnostic segmentation for the purpose of improving downstream tasks.

Further analysis We surprisingly find that MorphA$_{tgt}$ outperforms MorphA$_{src}$ on Arabic words and FC$_{src}$ outperforms FC$_{tgt}$ on English words. Therefore, we conduct an internal analysis where we look into the percentage of over and under segmentations.[2] In Appendix C, we present the number of under and over segmented words for each segmentation approach. Our analysis shows that MorphA$_{src}$ over segments 25% of the Arabic words. We observe that in 20% of these over segmentation cases, the Arabic definite article is segmented. For example, the word الكتب *Alktb* 'the books' is segmented to ال#كتب *Al#ktb* which is considered valid in segmentation schemes like D3. However, since we use the ATB scheme in *ArzEnSEG* annotation, the EMMA system penalizes the MorphA$_{src}$ segmenter and rewards the MorphA$_{tgt}$ segmenter which leaves most of the Arabic words and the definite article unsegmented. Another case is the segmentation of affixes, which is not done in ATB. For example, 16% of the over segmentation cases are separation of the Ta-Marbuta (feminine nominal ending) in Arabic words. The rest of the cases are grammatically incorrect segmentations. FC$_{tgt}$ is also shown to under segment around 17% more English words compared to FC$_{src}$ which can contribute to worse scores. We also observe that MORF$_{tgt}$ performs

---

[2] Over segmentation is a term we use to indicate that the word gets segmented to more morphemes compared to the gold standard segmentation. Meanwhile, under segmentation is a term we use to convey that the word is segmented into fewer morphemes than the gold standard segmentation.

significantly worse than the other segmenters when segmenting Arabic words, despite the fact that 81% of the Arabic words do not require segmentation. Internal analysis shows that MORF$_{tgt}$ over segments the Arabic words to the character level in an attempt to extract the underlying morphology of Egyptian Arabic, which it was not trained on.

5 Machine Translation Experiments

Since no previous research investigates the best segmentation technique for NMT of the code-switched Egyptian Arabic–English language pair, we explore training NMT models using the various segmentation setups discussed in Section 4 to answer RQ1. Moreover, we analyze the performance of the top-performing MT systems on different types of CS sentences to answer RQ2. Afterward, we compare the MT scores against the EMMA F1 scores discussed in Section 4.3 to answer RQ3.

5.1 Experimental Setup

We train Transformer models for our MT systems using Fairseq (Ott et al., 2019) on a single GeForce RTX 3090 GPU. We use the hyperparameters from the FLORES[3] benchmark for low-resource MT Guzmán et al. (2019), which we list in Appendix D. Afterwards, we evaluate the MT models on *ArzEn*'s dev and test sets using chrF2 (Popović, 2015).[4] We choose chrF2 over BLEU (Papineni et al., 2002) as it rewards partially correct translations which makes it a convenient choice for our research, and because chrF has shown to have higher correlation with human judgments over BLEU (Kocmi et al., 2021).

5.2 Machine Translation Systems

We experiment with different categories of segmentation setups. Table 4 shows all the different setups that we explore. See Table 10 in Appendix D for training time.

For the **unsupervised morphology-based segmentations**, we use MorphAGram, Morfessor, FlatCat, and LMVR to segment the source/target sides of the parallel corpus, where the segmenters were trained on each side separately (see Figure 2). For the best performing segmenter, we further investigate the best training setting, where we investigate using segmenters trained only on Arabic words on the source side as well as segmenters that are trained jointly on both sides.

Segmentation			**chrF2**	
Source		**Target**		
EGY	**EN**	**EN**	**dev**	**test**
raw		raw	47.1	49.9
Unsupervised Morphology-based Segmenters				
MorphA$_{src}$		MorphA$_{tgt}$	47.0	49.7
MORF$_{src}$		MORF$_{tgt}$	47.4	50.8
FC$_{src}$		FC$_{tgt}$	47.2	50.6
LMVR$_{src}$		LMVR$_{tgt}$	48.3	51.7
LMVR$_{joint}$		LMVR$_{joint}$	48.8	52.5
LMVR$_{src/egy}$	LMVR$_{tgt}$	LMVR$_{tgt}$	**48.9**	**52.9**
LMVR$_{src}$	LMVR$_{tgt}$	LMVR$_{tgt}$	48.8	**52.9**
LMVR$_{src/egy}$	LMVR$_{src}$	LMVR$_{tgt}$	48.5	52.0
Frequency-based Segmenters				
BPE$_{joint}$		BPE$_{joint}$	**50.1**	**53.7**
BPE$_{joint}$		raw	47.4	50.8
raw		BPE$_{joint}$	44.3	46.9
Supervised Morphology-based Segmenters				
MDMR$_{ATB}$	raw	raw	**48.8**	**52.1**
MDMR$_{D3}$	raw	raw	47.9	51.1
Combination Segmenters				
MDMR$_{ATB}$+BPE$_{joint}$	BPE$_{joint}$	raw	46.5	50.1
MDMR$_{ATB}$+BPE$_{joint}$	BPE$_{joint}$	BPE$_{joint}$	**50.2**	**53.8**
MDMR$_{D3}$+BPE$_{joint}$	BPE$_{joint}$	raw	46.9	50.7
MDMR$_{D3}$+BPE$_{joint}$	BPE$_{joint}$	BPE$_{joint}$	49.8	53.3

Table 4: The chrF2 results of our NMT systems with different segmentation combinations on *ArzEn*'s dev and test sets. Numbers highlighted in bold show the best performing system in each category.

For the **supervised morphology-based segmentations**, we only follow one approach and that is segmenting the source side using MDMR$_{ATB}$ or MDMR$_{D3}$ segmenters. This causes the English words to be left unsegmented.

For the **unsupervised frequency-based segmentations**, we exploit the jointly trained model, BPE$_{joint}$, to segment the source side only, target side only, or both sides of the parallel corpus.

Finally, inspired by the work of Oudah et al. (2019), we explore **combinations** between BPE and supervised morphology-based segmenters. As shown in Table 4, for the source side, we apply BPE$_{joint}$ on top of segmentations provided by either MDMR$_{ATB}$ or MDMR$_{D3}$. For the target side, we either leave it in the raw format or apply BPE$_{joint}$.

5.3 Machine Translation Results

Table 4 shows the different MT systems and their performance on *ArzEn*'s dev and test sets.

Amongst the **unsupervised morphology-based segmenters**, LMVR outperforms the other segmenters. We find that training language-specific

---

[3] FLORES hyperparameters outperform Vaswani et al. (2017) for our code-switched pair by +0.4 chrF2 points.

[4] We use sacreBLEU's (Post, 2018) implementation of chrF2.

segmenters (using LMVR$_{src/egy}$ for Arabic words and LMVR$_{tgt}$ for English words) outperforms training the segmenter jointly (LMVR$_{joint}$). This setup gives the best performing model, referred to as MT$_{LMVR}$.

Amongst the **supervised morphology-based segmenters**, the setup with MDMR$_{ATB}$ is the best, which we refer to as MT$_{ATB}$. The finding is consistent with Oudah et al. (2019)'s results.

For **unsupervised frequency-based segmenters**, using BPE$_{joint}$ to segment both source and target sides outperforms MT$_{LMVR}$ by +0.8 chrF2 points and MT$_{ATB}$ by +1.6 chrF2 points, which we refer to as MT$_{BPE}$. We observe that the ranking of these segmenters in MT performance is in reverse order compared to their ranking in segmentation task performance. We discuss this later in Section 6.

Most interestingly, contrary to (Oudah et al., 2019), we find that applying BPE$_{joint}$ on top of MDMR$_{ATB}$, which we refer to as MT$_{ATB+BPE}$, slightly improves over MT$_{BPE}$ but without statistical significance. However, MT$_{ATB+BPE}$ outperforms MT$_{ATB}$ and MT$_{LMVR}$ with statistical significance.[5] We further investigate the effectiveness and statistical significance achieved by this approach in a learning curve with varying the training data size in Section 5.4.

Finally, we note that segmenting English words on the source and target sides consistently, while controlling all other conditions, is always advantageous, as shown in Table 4.

5.4 Analysis

We further analyze the performance of the top MT systems from each segmentation setup (MT$_{ATB+BPE}$, MT$_{BPE}$, MT$_{LMVR}$, and MT$_{ATB}$). We first look into the number of Out-of-Vocabulary (OOV) tokens associated with each of the top-performing MT systems to examine whether it has an impact on their final ranking. Secondly, we investigate whether the ranking of the systems is consistent across the different types of sentences. We evaluate the systems against varying morphological richness, English percentages, and CS types. Thirdly, we further investigate the effectiveness of applying BPE over ATB compared to using each segmenter on its own. We conduct this analysis across different CS types and sizes of training data.

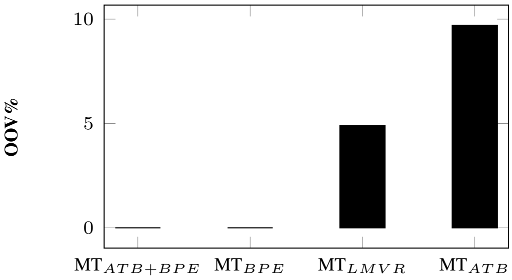


Figure 3: The percentage of the OOV words generated from each of the top-performing MT systems from each segmentation setup on *ArzEn*'s dev set.

OOV To further study the reason behind MT$_{BPE}$ and MT$_{ATB+BPE}$ top performance, we observe if the top-performing MT systems' ranking is linked with the percentage of OOV in the different MT systems. As shown in Figure 3, we find that for MT$_{ATB+BPE}$ and MT$_{BPE}$, the OOV percentage is 0%. However, for MT$_{LMVR}$ and MT$_{ATB}$, the percentage rises to 4.90% and 9.70%, respectively, which we believe contributes to worsening the MT systems.

Evaluating Systems Under Different Sentence Categories We evaluate the performance of the MT systems for sentences falling under different ranges of (i) morphological richness, (ii) percentage of CS English words, and (iii) sentence CS types. Morphological richness of a sentence is calculated as the quotient of the number of tokens in the segmented sentence and unsegmented original sentence. As expected, the performance of all the MT models decreases as the morphological richness increases and there is a boost in performance across all systems when the percentage of English words increases (see Appendix E). We observe that the MT$_{ATB+BPE}$ and MT$_{BPE}$ perform the best across all ranges for the first two features. We then evaluate the performance of the MT systems across sentences according to CS types: purely monolingual Arabic, CS, and CS having MCS (Hamed et al., 2022a; Mager et al., 2019). We observe that for all systems, the performance across CS sentences is higher than across monolingual Arabic sentences. We also observe that among CS sentences, the performance is reduced in the case of morphologically code-switched sentences. We believe that the following two factors can be contributing to these results. Firstly, the complex MCS

---

[5] We use Paired Significance Tests for Multi System Evaluation provided by SacreBLEU for the significance tests https://github.com/mjpost/sacrebleu#paired-bootstrap-resampling---paired-bs.

constructions might impose challenges to the MT system. Secondly, we observe that the average length of MCS sentences is higher than that of CS sentences in general. This is partially due to the fact that the tokens in MCS words are space-separated during the data preprocessing step. We report that on average, CS sentences contain 21.1 words (21.4 tokens), while MCS sentences contain 25.0 words (26.3 tokens).

Further Investigating the Effectiveness of MT$_{ATB+BPE}$ over MT$_{BPE}$ and MT$_{ATB}$ We study whether the ranking of MT$_{ATB+BPE}$, MT$_{BPE}$, and MT$_{ATB}$ is altered when going from a low-resource to an extreme low-resource setting across different sentence types. We achieve this by varying the MT training data to 25% and 50% of its original size. The results are shown in Table 5.

We observe that the effectiveness of the MT$_{ATB+BPE}$ varies under constrained conditions. For monolingual Arabic sentences, when training the MT systems on 100% of data, we see that MT$_{ATB+BPE}$ is not statistically significant over MT$_{BPE}$ and MT$_{ATB}$. Moreover, MT$_{ATB+BPE}$ was outperformed by MT$_{BPE}$. However, when training with 25% and 50% of data, MT$_{ATB+BPE}$ outperforms MT$_{BPE}$ and MT$_{ATB}$ with statistical significance across all sentence types. We further exhibit this in Figure 4 when all sentence categories are considered during analysis under different data sizes. This finding highlights the importance of combining morphology-based and frequency-based segmentations in extremely low-resource scenarios.

We also observe that across all data sizes, MT$_{ATB}$ performs the worst on CS sentences. Our first hypothesis is that this is due to English words left unsegmented. However, results in Table 3 contradict this hypothesis. Our second hypothesis is that since MDMR$_{ATB}$ takes into consideration the context of the word prior to segmentation, the English words in the CS sentences might break the flow of the sentence, hence negatively impacting the context of the word, thus worsening the score.

System Selection As per our findings, MT$_{ATB+BPE}$ is always the best choice across all sentence types in extreme low-resource settings. However, when training on 100% of the data, MT$_{BPE}$ improves slightly over MT$_{ATB+BPE}$ on monolingual Arabic sentences. Therefore, we create a system selection setup which uses both, MT$_{ATB+BPE}$ and MT$_{BPE}$, to investigate if it would lead to further improvements. In this setup, the CS and monolingual English sentences are translated using MT$_{ATB+BPE}$, while monolingual Arabic sentences are translated using MT$_{BPE}$. Despite the hybrid system showing an overall improvement of +0.1 chrF2 points over MT$_{ATB+BPE}$, the improvement is not statistically significant.

Size	MT System	All	EGY	CS	MCS
25%	**MT$_{ATB+BPE}$**	**39.8 (1)**	**36.6 (1)**	**40.6 (1)**	**40.0 (1)**
	MT$_{BPE}$	38.4 (2)	35.6 (3)	39.1 (2)	38.5 (2)
	MT$_{ATB}$	36.9 (3)	35.9 (2)	37.0 (3)	36.0 (3)
50%	**MT$_{ATB+BPE}$**	**45.9 (1)**	**42.1 (1)**	**46.8 (1)**	**46.4 (1)**
	MT$_{BPE}$	44.5 (2)	40.7 (3)	45.5 (2)	44.8 (2)
	MT$_{ATB}$	44.0 (3)	41.4 (2)	44.7 (3)	44.0 (3)
100%	**MT$_{ATB+BPE}$**	**50.2 (1)**	44.4 (2)	**51.5 (1)**	**51.3 (1)**
	MT$_{BPE}$	50.1 (2)	**44.6 (1)**	51.3 (2)	51.1 (2)
	MT$_{ATB}$	48.8 (3)	44.2 (3)	49.8 (3)	49.4 (3)

Table 5: We compare the results of the best performing MT system (MT$_{ATB+BPE}$) which utilizes BPE on top of ATB segmentation against the MT systems that utilize BPE (MT$_{BPE}$) or ATB (MT$_{ATB}$) only on *ArzEn*'s dev set. We report chrF2 results when training on 25%, 50%, and 100% of the training data. Results are shown for different types of sentences: monolingual Egyptian Arabic (EGY), code-switched (CS), and morphologically code-switched (MCS), as well as all sentences (All). The ranking of the MT systems with respect to each other is represented by the numbers between parentheses, where (1) is the best rank and (3) is the worst.

6 Discussion

We revisit the RQs we outlined in our introduction.

RQ1 - Which segmentation setup performs the best in the downstream MT task across different training sizes? Results show that frequency-based segmentation applied on top of morphology-based segmentation outperforms the other segmentation techniques, with statistical significance on lower resource settings. The superiority of this approach is seen across sentences with varying morphological richness, percentage of English words, and across sentences with different CS types. We believe the strength of the combination is because it exploits complementarity of both methods. On one hand, supervised morphology-based segmenters bring in high correctness; however, they are not always robust, having high OOV rates. On the other hand, while BPE segmentation is not necessarily morphologically correct, it achieves high robustness. The robustness of BPE is consistent with the findings in Banerjee and Bhattacharyya (2018).

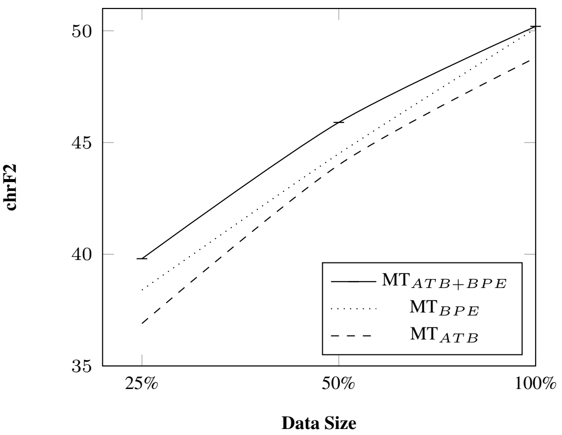


Figure 4: Demonstrates the effectiveness of applying BPE on top of ATB segmentation ($MT_{ATB+BPE}$) as opposed to using either approaches separately (MT_{BPE} and MT_{ATB}), which is confirmed when reducing the amount of training data. Results are reported on *ArzEn*'s dev set.

RQ2 - Does the effectiveness of the different segmenters in the MT task differ according to the CS type of the source sentence? We observe that the effectiveness of the different segmenters on MT performance is consistent across two categories of CS sentences; those with and without MCS. However, when comparing their effectiveness on monolingual Arabic vs. CS sentences, we observe that the rankings between segmenters are not consistent. In the case of constrained data size settings (25% and 50% of data), we observe a clear pattern where MT_{ATB} outperforms MT_{BPE} on monolingual sentences, while MT_{BPE} outperforms MT_{ATB} on CS. In the case of using 100% of the training data, $MT_{ATB+BPE}$ outperforms MT_{BPE} on CS sentences; however, MT_{BPE} outperforms $MT_{ATB+BPE}$ on monolingual Arabic sentences. Since our test and dev sets are dominated by CS sentences (61.5% and 63.8%, respectively), we believe that the overall ranking is more greatly affected by the systems' performance on CS sentences, thus reflecting the same ranking on the overall evaluation set as that across CS sentences.

RQ3 - Is there a correlation between a more morphologically correct segmentation and MT performance? For unsupervised morphology-based segmenters, a segmenter with a better segmentation EMMA F1 score also scores better in the downstream MT task. However, we cannot hypothesize that a better segmentation score implies a better translation system, as counter examples exist. For example, while we notice that $MDMR_{ATB}$ gives the best segmentation in terms of EMMA F1 score, it does not outperform any of the top-performing MT systems. We hypothesize that despite $MDMR_{ATB}$'s capability of generating morphologically correct segmentations, it can generate infrequent morphemes due to the out-of-domain data which it is trained on. This may not only increase the sentence length which worsens MT performance as shown in Mager et al. (2022), but may also be one of the contributing factors to the 9.70% OOV percentage found in MT_{ATB}. On the contrary, BPE_{joint} performs the worst in the segmentation task as we expect, since it is designated for agnostic-based segmentations; however, it surpasses the top-performing MT models. We believe this is due to its capability to generate semi-correct segmentation and to reduce OOV rates.

7 Conclusion and Future Work

In this paper, we study the impact of a comprehensive set of morphological and frequency-based segmentation methods on MT, where the source is a code-switched text. The experiments are performed on code-switched Arabic-English to English. We found that the supervised morphological segmenter achieved the best performance on the segmentation task, followed by unsupervised morphological methods, and finally, unsupervised frequency-based. Afterward, we train 18 different MT systems with different source and target side segmentations. We find that the rank of the segmenters is reversed, as BPE's could not be outperformed (significantly) by any morphological-inspired segmentation method. However, combining morphology-based and frequency-based segmentations has shown to give improvements, which are statistically significant in lower resource settings, where the training data size is reduced to 25% and 50%. For future work, we plan to apply our different MT setups on other low resource and code-switched language pairs. Specifically, we plan to explore languages with different typologies, to study whether or not the relation between the data size and choice of the segmentation setup (frequency-based, morphology-based, or a mix) is based on morphological features and data size rather than the language itself. Moreover, we plan to extend our annotated dataset, *ArzEnSEG*, by adding further details to allow evaluating different schemes.

Limitations

The first challenge we face in this work is the computational power needed to tune the Morfessor family segmenters. Therefore, in an attempt to overcome this challenge, for the Morfessor family, the choice of the optimal hyperparameters is dependent on the parent tool. For instance, the optimal hyperparameters for Morfessor are directly used in its FlatCat variant and the hyperparameters specific to FlatCat are then tuned. The same applies for LMVR which is a variant of FlatCat. Moreover, we cannot verify whether or not our results will hold for languages with different typologies, specifically those that are low resource and code-switched. Therefore, the results of this research must be seen in light of these limitations.

Ethics Statement

We could not find any potential harm that might derive from this work. However, we understand that translation as a whole can impact the cultural and social life of the people that use it. This has been used in the past in a harmful way, i.e., to spread colonial views (Mbuwayesango, 2018). Therefore, we call the final user to use this work ethically. Regarding the annotation process, all manual annotations were made by a subset of the authors of this paper. Therefore, no hiring of external workers was necessary.

Acknowledgements

We want to thank all the anonymous reviewers for their helpful comments and suggestions. This project has benefited from financial support to Marwa Gaser and Manuel Mager by the DAAD (German Academic Exchange Service).

References

Abdulmohsen Al-Thubaity and Abdullah Al-Subaie. 2015. Effect of word segmentation on Arabic text classification. In *Proceedings of the International Conference on Asian Language Processing (IALP)*, pages 127–131.

Duygu Ataman, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. Linguistically motivated vocabulary reduction for neural machine translation from Turkish to English. *The Prague Bulletin of Mathematical Linguistics*, 108(1):331–342.

Mohamed Balabel, Injy Hamed, Slim Abdennadher, Ngoc Thang Vu, and Özlem Çetinoğlu. 2020. Cairo student code-switch (CSCS) corpus: An annotated Egyptian Arabic-English corpus. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 3973–3977.

Tamali Banerjee and Pushpak Bhattacharyya. 2018. Meaningless yet meaningful: Morphology grounded subword-level NMT. In *Proceedings of the workshop on subword/character level models*, pages 55–60.

Leonard E Baum and Ted Petrie. 1966. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*, 37(6):1554–1563.

Raytheon BBN Technologies, Linguistic Data Consortium, and Sakhr Software. 2012. Arabic-Dialect/English parallel text – LDC2012T09. Web Download. Philadelphia: Linguistic Data Consortium.

Houda Bouamor, Nizar Habash, Mohammad Salameh, Wajdi Zaghouani, Owen Rambow, Dana Abdulrahim, Ossama Obeid, Salam Khalifa, Fadhl Eryani, Alexander Erdmann, and Kemal Oflazer. 2018. The MADAR Arabic Dialect Corpus and Lexicon. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Özlem Çetinoğlu, Sarah Schulz, and Ngoc Thang Vu. 2016. Challenges of computational processing of code-switching. In *Proceedings of the Workshop on Computational Approaches to Code Switching*, pages 1–11.

Shuguang Chen, Gustavo Aguilar, Anirudh Srinivasan, Mona Diab, and Thamar Solorio. 2022. CALCS 2021 shared task: Machine translation for code-switched data. *arXiv preprint arXiv:2202.09625*.

Song Chen, Dana Fore, Stephanie Strassel, Haejoong Lee, and Jonathan Wright. 2017. BOLT Egyptian Arabic SMS/Chat and Transliteration LDC2017T07. Philadelphia: Linguistic Data Consortium.

Song Chen, Jennifer Tracey, Christopher Walker, and Stephanie Strassel. 2019. BOLT Arabic discussion forum parallel training data. Linguistic Data Consortium (LDC) catalog number LDC2019T01, ISBN 1-58563-871-4.

Mrinal Dhar, Vaibhav Kumar, and Manish Shrivastava. 2018. Enabling code-mixed translation: Parallel corpus creation and MT augmentation approach. In *Proceedings of the Workshop on Linguistic Resources for Natural Language Processing*, pages 131–140.

Ramy Eskander, Francesca Callejas, Elizabeth Nichols, Judith L Klavans, and Smaranda Muresan. 2020. Morphagram, evaluation and framework for unsupervised morphological segmentation. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 7112–7122.

Hassan Gadalla, Hanaa Kilany, Howaida Arram, Ashraf Yacoub, Alaa El-Habashi, Amr Shalaby, Krisjanis Karins, Everett Rowson, Robert MacIntyre, Paul Kingsbury, David Graff, and Cynthia McLemore. 1997. CALLHOME Egyptian Arabic transcripts – LDC97T19. Web Download. Philadelphia: Linguistic Data Consortium.

Philip Gage. 1994. A new algorithm for data compression. *C Users Journal*, 12(2):23–38.

Stig-Arne Grönroos, Sami Virpioja, and Mikko Kurimo. 2020. Morfessor EM+Prune: Improved subword segmentation with expectation maximization and pruning. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 3944–3953.

Stig-Arne Grönroos, Sami Virpioja, Peter Smit, and Mikko Kurimo. 2014. Morfessor flatcat: An HMM-based method for unsupervised and semi-supervised learning of morphology. In *Proceedings of the International Conference on Computational Linguistics (COLING): Technical Papers*, pages 1177–1185.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6098–6111, Hong Kong, China.

Nizar Habash, Muhammed AbuOdeh, Dima Taji, Reem Faraj, Jamila El Gizuli, and Omar Kallas. 2022. Camel treebank: An open multi-genre Arabic dependency treebank. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 2672–2681.

Nizar Habash, Mona Diab, and Owen Rambow. 2012a. Conventional Orthography for Dialectal Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 711–718, Istanbul, Turkey.

Nizar Habash, Ramy Eskander, and Abdelati Hawwari. 2012b. A Morphological Analyzer for Egyptian Arabic. In *Proceedings of the Workshop of the Special Interest Group on Computational Morphology and Phonology (SIGMORPHON)*, pages 1–9, Montréal, Canada.

Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. 2013. Morphological Analysis and Disambiguation for Dialectal Arabic. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Atlanta, Georgia.

Nizar Habash, Abdelhadi Soudi, and Tim Buckwalter. 2007. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, pages 15–22. Springer, Netherlands.

Nizar Y Habash. 2010. Introduction to Arabic natural language processing. *Synthesis lectures on human language technologies*, 3(1):1–187.

Injy Hamed, Pavel Denisov, Chia-Yu Li, Mohamed Elmahdy, Slim Abdennadher, and Ngoc Thang Vu. 2022a. Investigations on speech recognition systems for low-resource dialectal Arabic–English code-switching speech. *Computer Speech & Language*, 72:101278.

Injy Hamed, Mohamed Elmahdy, and Slim Abdennadher. 2018. Collection and analysis of code-switch Egyptian Arabic-English speech corpus. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Injy Hamed, Nizar Habash, Slim Abdennadher, and Ngoc Thang Vu. 2022b. ArzEn-ST: A three-way speech translation corpus for code-switched Egyptian Arabic-English. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*.

Injy Hamed, Nizar Habash, Slim Abdennadher, and Ngoc Thang Vu. 2022c. Investigating lexical replacements for Arabic-English code-switched data augmentation. *arXiv preprint arXiv:2205.12649*.

Injy Hamed, Ngoc Thang Vu, and Slim Abdennadher. 2020. ArzEn: A speech corpus for code-switched Egyptian Arabic-English. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 4237–4246.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In *Proceedings of the Conference on Machine Translation*, pages 478–494.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, pages 66–71, Brussels, Belgium.

LDC. 2002a. 1997 hub5 Arabic transcripts – LDC2002T39. Web Download. Philadelphia: Linguistic Data Consortium.

LDC. 2002b. CALLHOME Egyptian Arabic transcripts supplement – LDC2002T38. Web Download. Philadelphia: Linguistic Data Consortium.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *Proceedings of the International Conference on Arabic Language Resources and Tools*, pages 102–109, Cairo, Egypt.

Mohamed Maamouri, Ann Bies, Seth Kulick, Dalila Tabessi, and Sondos Krouna. 2012. Egyptian Arabic Treebank DF Parts 1-8 V2.0 - LDC catalog numbers LDC2012E93, LDC2012E98, LDC2012E89, LDC2012E99, LDC2012E107, LDC2012E125, LDC2013E12, LDC2013E21.

Manuel Mager, Özlem Çetinoğlu, and Katharina Kann. 2019. Subword-level language identification for intra-word code-switching. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2005–2011.

Manuel Mager, Arturo Oncevay, Elisabeth Maier, Katharina Kann, and Thang Vu. 2022. BPE vs. morphological segmentation: A case study on machine translation of four polysynthetic languages. In *Findings of* *the Association for Computational Linguistics (ACL)*, pages 961–971.

Sainik Kumar Mahata, Soumil Mandal, Dipankar Das, and Sivaji Bandyopadhyay. 2019. Code-mixed to monolingual translation framework. In *Proceedings of the Forum for Information Retrieval Evaluation*, pages 30–35.

Dora Rudo Mbuwayesango. 2018. The bible as tool of colonization. *Colonialism and the Bible: Contemporary Reflections from the Global South*, page 31.

Mohamed Amine Menacer, David Langlois, Denis Jouvet, Dominique Fohr, Odile Mella, and Kamel Smaïli. 2019. Machine translation on a parallel code-switched corpus. In *Canadian Conference on Artificial Intelligence*, pages 426–432.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 7022–7032, Marseille, France.

John E Ortega, Richard Castro Mamani, and Kyunghyun Cho. 2020. Neural machine translation with a polysynthetic low resource language. *Machine Translation*, 34(4):325–346.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. FAIRSEQ: A fast, extensible toolkit for sequence modeling. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota.

Mai Oudah, Amjad Almahairi, and Nizar Habash. 2019. The impact of preprocessing on Arabic-English statistical and neural machine translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 214–221.

Yirong Pan, Xiao Li, Yating Yang, and Rui Dong. 2020. Morphological word segmentation on agglutinative languages for neural machine translation. *arXiv preprint arXiv:2001.01589*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, pages 311–318.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 1094–1101, Reykjavik, Iceland.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium.

Christian Roest, Lukas Edman, Gosse Minnema, Kevin Kelly, Jennifer Spenader, and Antonio Toral. 2020. Machine translation for English–Inuktitut with segmentation, data acquisition and pre-training. In *Proceedings of the Conference on Machine Translation*, pages 274–281.

Jonne Saleva and Constantine Lignos. 2021. The effectiveness of morphology-aware segmentation in low-resource neural machine translation. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL): Student Research Workshop*, pages 164–174.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, pages 1715–1725, Berlin, Germany.

R. Mahesh K. Sinha and Anil Thakur. 2005. Machine translation of bi-lingual Hindi-English (Hinglish) text. In *Proceedings of Machine Translation Summit X: Papers*, pages 149–156, Phuket, Thailand.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *Proceedings of the Demonstrations at the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 21–24, Gothenburg, Sweden.

Kai Song, Yue Zhang, Heng Yu, Weihua Luo, Kun Wang, and Min Zhang. 2019. Code-switching for enhancing NMT with pre-specified translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 449–459.

Sebastian Spiegler and Christian Monson. 2010. EMMA: A novel evaluation metric for morphological analysis. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 1029–1037.

Vivek Srivastava and Mayank Singh. 2020. PHINC: A parallel Hinglish social media code-mixed corpus for machine translation. In *Proceedings of the Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 41–49.

Dima Taji, Nizar Habash, and Daniel Zeman. 2017. Universal dependencies for Arabic. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 166–176.

Ishan Tarunesh, Syamantak Kumar, and Preethi Jyothi. 2021. From machine translation to code-switching: Generating high-quality code-switched text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3154–3169.

Ahmed Tawfik, Mahitab Emam, Khaled Essam, Robert Nabil, and Hany Hassan. 2019. Morphology-aware word-segmentation in dialectal Arabic adaptation of neural machine translation. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 11–17.

Jennifer Tracey et al. 2021. BOLT Egyptian Arabic sms/chat parallel training data LDC2021T15. Web Download. Philadelphia: Linguistic Data Consortium.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, Mikko Kurimo, et al. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. Technical report, Aalto University, Finland.

Jitao Xu and François Yvon. 2021. Can you traducir this? machine translation for code-switched input. In *Proceedings of the Workshop on Computational Approaches to Linguistic Code-Switching*, pages 84–94.

Nasser Zalmout and Nizar Habash. 2017. Optimizing tokenization choice for machine translation across multiple target languages. *The Prague Bulletin of Mathematical Linguistics*, 108:257–270.

A Data Preprocessing

We use the same preprocessing pipeline for all the corpora, where we start by removing any corpus-related annotations. Afterward, we remove URLs and emoticons, through *tweet-preprocessor*,[6] remove trailing and leading spaces, and tokenize numbers. Finally, *Moses Tokenizer*[7] is applied for tokenization and empty lines are removed from the parallel corpora. For LDC2017T07 (Chen et al., 2017) and LDC2019T01 (Chen et al., 2019), some sentences have literal and intended translations for some words. Hence, we opt for one translation having all literal translations and another having all intended translations. Once all the preprocessing steps are done, we concatenate the nine corpora collectively and pass the resulting training corpus to MADAMIRA (Pasha et al., 2014) to obtain two different supervised morphological segmentations of the corpus, namely `ATB_BWFORM` and `D3_BWFORM` which we discuss in Section 4.2. Additionally, we obtain a raw training corpus by further tokenizing punctuation and removing emojis using MADAMIRA's D0 scheme (Zalmout and Habash, 2017). Nonetheless, we normalize the Arabic letters ى and أ to ي and ا respectively through CAMeL Tools (Obeid et al., 2020) since D0's output is not normalized.

B Segmenters' Hyperparameters

Morfessor family Since all Morfessor family segmenters are morphology inspired, the hyperparameters are tuned on *ArzEnSEG*'s dev set. For LMVR$_{src}$ and LMVR$_{tgt}$ setting the vocabulary sizes to $64k$ and $16k$ respectively outperform $3k$, $5k$, $8k$, $16k$, $32k$, $100k$. For LMVR$_{joint}$ setting the vocabulary size to $32k$ outperforms $3k$, $5k$, $8k$, $16k$, $64k$, and $100k$. Meanwhile, For LMVR$_{src/egy}$ setting the vocabulary size to $64k$ outperforms $3k$, $5k$, $8k$, $16k$, $32k$, and $100k$.

Table 6 shows the possible values used during the optimal hyperparameter search for each Morfessor tool. For Morfessor, FlatCat, and LMVR 18, 360, and 7 different segmentation models are generated. These are a result of the combination of the possible hyperparameter values. The hyperparameter combination which yields the highest EMMA score on *ArzEnSEG*'s dev set for each Morfessor tool is used to segment the MT training data. The best combination values are reported in Table 7.

Segmenters Hyperparameters	
Hyperparameter	**Values Bound**
Morfessor	
-F	[0.003, 0.005, 0.007]
-d	[log, ones, none]
-a	[recursive, viterbi]
FlatCat	
-p	[50, 60, 70, 80, 90, 100, 200, 300]
–min-perplexity-length	[1, 2, 3, 4, 5]
–min-shift-remainder	[1, 2, 3]
–length-threshold	[2, 3, 4]
LMVR	
–lexicon-size	[3k, 5k, 8k, 16k, 32k, 64k, 100k]

Table 6: The values bound we use during the best hyperparameter combination search for the Morfessor tools.

MorphAGram Akin to the Morfessor family, we tune the hyperparameters on *ArzEnSEG*'s dev set and train two models: one on the source side and the other on the target side of the training parallel corpus which we refer to as MorphA$_{src}$ and MorphA$_{tgt}$, respectively (see Figure 2). Tuning results show that setting the vocabulary size to $3k$ for MorphA$_{src}$ outperforms $5k$, $8k$, $16k$, $32k$, and $50k$, while setting the vocabulary size to $50k$ for MorphA$_{tgt}$ outperforms $5k$, $3k$, $8k$, $16k$, and $32k$. Nevertheless, it is worth noting that the vocabulary size on the target side is $< 50k$ which shows that MorphA$_{tgt}$ performs the best when no segmentations are applied on the English words.

BPE Since BPE is a segmentation technique that is designated for agnostic segmentation for MT tasks, we tune the vocabulary size on *ArzEn*'s dev set in an NMT task. We apply a vocabulary size of $8k$, which outperforms $5k, 16k, 32k, 64k$.

C Segmenters Performance Analysis

Table 8 shows the error analysis we perform on the segmenters with regards to over segmentation, under segmentation, or generating the correct number of segmentations per word.

D MT Hyperparameters

The MT hyperparameters are shown in Table 9. We follow the FLORES hyperparameters for low-resource language pairs. The full train command can be found on FLORES GitHub.[8] The training

---

[6] `https://pypi.org/project/tweet-preprocessor/`

[7] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl`

[8] `https://github.com/facebookresearch/flores/blob/6641ec0e23d173906dd2e01551a430884b1dba31/floresv1/README.md#train-a-baseline-transformer-model`

	Morfessor			**FlatCat**				**LMVR**
Data	**-F**	**-d**	**-a**	**-p**	**–min -perplexity -length**	**–min -shift -remainder**	**–length -threshold**	**–lexicon -size**
src	0.003	log	recursive	200	1	1	4	64k
tgt	0.003	log	recursive	100	4	2	4	16k
src/egy	0.007	log	recursive	300	1	1	2	64k
joint	0.007	log	recursive	300	4	2	4	32k

Table 7: The different hyperparameters used for each Morfessor family segmenter depending on whether the model is trained on the source (src), target (tgt), source without English, i.e., Egyptian, (src/egy), or source+target (joint) side(s).

	EGY					**EN**				
Segmenter	**under**	**over**	**correct**	**seg.**	**unseg.**	**under**	**over**	**correct**	**seg.**	**unseg.**
raw	634	0	2,780	0	2,780	71	0	430	0	430
MorphA$_{src}$	249	855	2,310	385	1,925	70	45	386	1	385
MORF$_{src}$	466	299	2,649	148	2,501	15	103	383	42	341
FC$_{src}$	592	8	2,814	42	2,772	56	7	438	15	423
LMVR$_{src}$	520	47	2,847	111	2,736	43	7	451	28	423
MorphA$_{tgt}$	634	35	2,745	0	2,745	6	148	347	65	282
MORF$_{tgt}$	0	3,150	264	3	261	21	37	443	49	394
FC$_{tgt}$	634	0	2,780	0	2,780	66	8	427	5	422
LMVR$_{tgt}$	634	0	2,780	0	2,780	23	19	459	48	411
LMVR$_{joint}$	485	79	2,850	144	2,706	20	32	449	51	398
BPE$_{joint}$	338	368	2,708	230	2,478	28	132	341	30	311
MDMR$_{ATB}$	38	62	3,314	581	2,733	71	0	430	0	430
MDMR$_{D3}$	38	293	3,083	561	2,522	71	0	430	0	430

Table 8: The table shows the number of under segmented words (under), over segmented words (over), and the number of cases where the segmenter generates the correct count of morphemes (correct) for English (EN) and Arabic (EGY) words in *ArzEnSEG* test set. Additionally, out of the correct count of morphemes (correct), we report the words which originally require segmentation (seg.) and those which do not (unseg.).

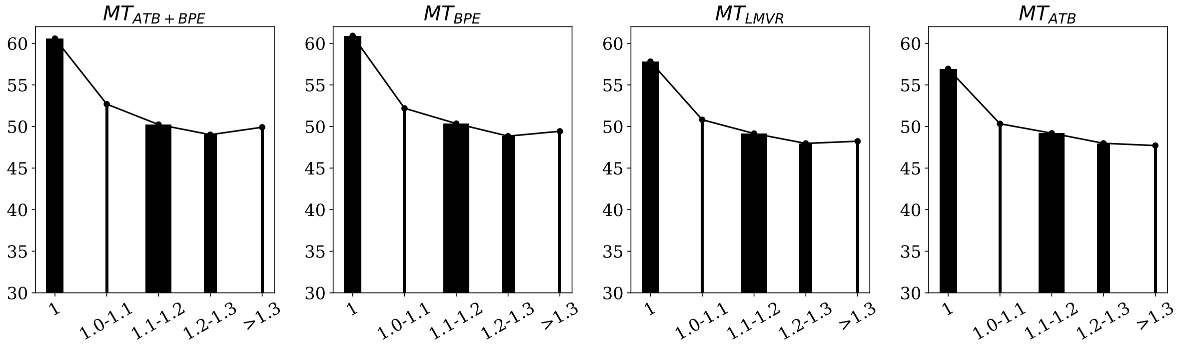
time for MT model the training time is exhibited in Table 10.

Hyperparameter	**Value**
encoder-layers	5
decoder-layer	5
encoder-embed-dim	512
decoder-embed-dim	512
encoder-ffn-embed-dim	2
decoder-ffn-embed-dim	2
dropout	0.4
attention-dropout	0.2
relu-dropout	0.2
weight-decay	0.0001
label-smoothing	0.2
warmup-updates	4000
warmup-init-lr	1e-9

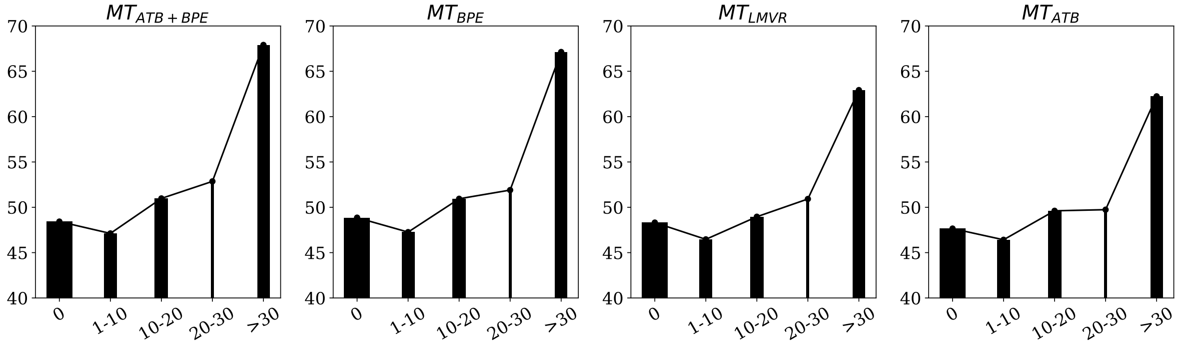
Table 9: FLORES hyperparameters for low-resource language pairs.

E Evaluating Systems Under Different Sentence Categories

Figure 5 shows the performance of the top MT systems from each segmentation setup across sentences of different morphological richness ratios and different percentages of English words in *ArzEn*'s dev set. Results show that there is a general decrease in performance as the morphological richness increases. However, as the percentage of English words in the sentences increases, the performance of the systems generally improves. It is also shown that MT$_{ATB+BPE}$ and MT$_{BPE}$ achieve overall comparable performances and outperform the other systems.



(a) Morphological Richness



(b) % English Words

Figure 5: The average chrF2 scores for the top performing MT systems from each segmentation setup across sentences with various (a) morphological richness ratios and (b) percentage of English words in *ArzEn*'s dev set. Morphological richness of a sentence is calculated as the quotient of the number of tokens in the segmented sentence and unsegmented original sentence. The bar width is indicative of the number of sentences in each bin.

Segmentation			**Training Time *(seconds)***
Source		**Target**	
EGY	**EN**	**EN**	
raw		raw	13,522
Unsupervised Morphology-based Segmenters			
MorphA$_{src}$		MorphA$_{tgt}$	24,731
MORF$_{src}$		MORF$_{tgt}$	18,916
FC$_{src}$		FC$_{tgt}$	18,225
LMVR$_{src}$		LMVR$_{tgt}$	4,476
LMVR$_{joint}$		LMVR$_{joint}$	18,019
LMVR$_{src/egy}$	LMVR$_{tgt}$	LMVR$_{tgt}$	22,462
LMVR$_{src}$	LMVR$_{tgt}$	LMVR$_{tgt}$	4,181
LMVR$_{src/egy}$	LMVR$_{src}$	LMVR$_{tgt}$	4,526
Frequency-based Segmenters			
BPE$_{joint}$		BPE$_{joint}$	18,279
BPE$_{joint}$		raw	23,193
raw		BPE$_{joint}$	17,905
Supervised Morphology-based Segmenters			
MDMR$_{ATB}$	raw	raw	18,280
MDMR$_{D3}$	raw	raw	18,519
Combination Segmenters			
MDMR$_{ATB}$+BPE$_{joint}$	BPE$_{joint}$	raw	17,629
MDMR$_{ATB}$+BPE$_{joint}$	BPE$_{joint}$	BPE$_{joint}$	27,088
MDMR$_{D3}$+BPE$_{joint}$	BPE$_{joint}$	raw	24,256
MDMR$_{D3}$+BPE$_{joint}$	BPE$_{joint}$	BPE$_{joint}$	23,611

Table 10: The training time in seconds of our different NMT systems.

Identifying the limits of transformers when performing model-checking with natural language

Tharindu Madusanka and **Ian Pratt-hartmann** and **Riza theresa Batista-navarro**
Department of Computer Science
The University of Manchester

Abstract

Can transformers learn to comprehend logical semantics in natural language? Although many strands of work on natural language inference have focussed on transformer models' ability to perform reasoning on text, the above question has not been answered adequately. This is primarily because the logical problems that have been studied in the context of natural language inference have their computational complexity vary with the logical and grammatical constructs within the sentences. As such, it is difficult to access whether the difference in accuracy is due to logical semantics or the difference in computational complexity. A problem that is much suited to address this issue is that of the model-checking problem, whose computational complexity remains constant (for fragments derived from first-order logic). However, the model-checking problem remains untouched in natural language inference research. Thus, we investigated the problem of model-checking with natural language to adequately answer the question of how the logical semantics of natural language affects transformers' performance [1]. Our results imply that the language fragment has a significant impact on the performance of transformer models. Furthermore, we hypothesise that a transformer model can at least partially understand the logical semantics in natural language but can not completely learn the rules governing the model-checking algorithm.

1 Introduction

Recent years have seen a surge of interest in the application of neural networks to the topic of natural language inference (Raffel et al., 2019; Lan et al., 2020; Yang et al., 2019), the central task of which is to recover information entailed by, but not explicitly stated in natural language texts (Lin et al., 2019; Sinha et al., 2019; Geiger et al., 2018; Wang et al., 2021). This problem is of theoretical (as well as practical) interest because the ability to understand the logical consequences of natural language sentences is an essential part of what it is to understand the grammatical constructions and closed-class expressions they contain. More specifically, the ability of neural network models to recognize logical entailments is constitutive of their ability to understand the texts they are processing.

It is important to distinguish two strands of work in this area. The first focuses on *entailment* as defined by human-constructed datasets (Bowman et al., 2015; Williams et al., 2018), where the inferences depend on implicit background knowledge and have a probabilistic character. The second focuses on the recognition of formal logical entailments, for which data sets can be machine-generated using existing symbolic reasoning techniques (Richardson et al., 2020; Richardson and Sabharwal, 2021; Geiger et al., 2018). This latter strand of work is particularly pertinent to the theoretical problem of whether neural network models can learn the logical semantics of natural language. Commonsense knowledge, human judgement and considerations of plausibility are consciously excluded.

A logical problem of great theoretical interest that has not been studied in the context of natural language inference is the model-checking problem: given a formula ϕ and a structure $\mathfrak{A}$, determine whether ϕ is true in $\mathfrak{A}$ ($\mathfrak{A} \models \psi$). The ability to perform model-checking is indicative of a grasp of the logical semantics of the expressions concerned. In the context of natural language inference, we are particularly interested in a variant of the model-checking problem where the structure and the formula are interpreted in natural language. It is noteworthy to emphasise how the model-checking problem differs from other inference problems. In other logical reasoning problems such as satisfiability, computational complexity varies among multiple

---

[1] dataset and code available at `https://github.com/iTharindu/reasoning-withing-a-structure`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3539–3550
May 2-6, 2023 ©2023 Association for Computational Linguistics

<table>
<tr><td colspan="2">Sturcture (Model) : $\mathcal{D} = \{Alan, Talia, Tony, Hailee, Aria\}$
$\mathcal{P} = \{actors, happy, scholars, love, admire\}$
$actors = \{Aria, Tony, Hailee\}$
$happy = \{Hailee, Talia\}$
$scholars = \{Alan, Tony\}$
$love = \{(Hailee, Talia), (Hailee, Alan), (Haillee, Tony), (Talia, Alan), (Tony, Aria)\}$
$admire = \{(Aria, Tony), (Aria, Hailee), (Talia, Hailee), (Alan, Tony), (Alan, Talia)\}$</td></tr>
<tr><td colspan="2">Structure (Model) in natural language: <i>Alan, Talia, Tony, Hailee and Aria are people in a group. Aria Tony and Hailee are actors. Hailee and Talia are happy. Alan and Tony are scholars. Hailee loves Talia, Alan and Tony. Talia loves Alan. Tony loves Aria. Aria admires Tony and Hailee. Talia admires Hailee, Alan admires Tony and Talia</i></td></tr>
<tr><td>Formula : $\exists x(actor(x) \wedge \forall y(scholar(y) \Rightarrow love(x,y)))$
Formula in natural language (Sentence) :
<i>Some actors love every scholar</i>
Validity : True</td><td>Formula : $\forall x((actor(x) \wedge happy(x)) \Rightarrow scholar(x))$
Formula in natural language (Sentence) :
<i>All actors who are happy are scholars</i>
Validity : False</td></tr>
</table>

Figure 1: An instance of the model-checking problem, the domain of the structure is represented in $\mathcal{D}$, and predicates are characterised by $\mathcal{P}$. A formula can be valid or not according to the structure *(the formula on the left is valid, while the one on the right is invalid).* Corresponding natural language representations for both the structure and the formula are also presented.

computational complexity classes (NLogSpace to NExpTime for fragments considered in this study) in language fragments of the finite variable space (Pratt-Hartmann and Third, 2006; Pratt-Hartmann, 2010). In contrast, in the model checking problem, the computational complexity remains in PTime for any fragment in the finite variable space (given they are derived from first-order logic). Furthermore, inference in the model-checking problem is fairly straightforward, which has also been evident by low computational complexity. Hence, the model-checking problem provides an ideal problem to analyse how different logically significant words and grammatical constructs (semantics of logic in natural language) affect transformers' ability to reason, as the underlying computational complexity remains in PTime.

Figure 1 depicts an instance of the model-checking problem, where the sentence *"Some actors love every scholar"* is *True* according to the structure presented, as the assignment of *"Hailee"* to the variable x makes the corresponding formula ($\exists x(actor(x) \wedge \forall y(scholar(y) \Rightarrow love(x,y)))$) $True$. However, when assessing the sentence *"All actors who are happy are scholars"*, there is no assignment that makes the formula ($\forall x((actor(x) \wedge happy(x)) \Rightarrow scholar(x))$) $True$ according to the structure (the set of *actors* who are *musicians* is {*Hailee*}, which is not a subset of *scholars'*, namely {*Alan, Tony*}).

In our analysis of transformers' capabilities in the model-checking problem, we ask two fundamental questions, *(1) can transformers perform model-checking with natural language? (2) if so, can transformers understand the logical semantics of natural language: i.e. can transformers comprehend the semantics of distinctively logical words and grammatical concepts such as determiners, relative clauses and anaphora?* To answer the above-mentioned questions, we construct a model-checking dataset ($\mathcal{FO}^2$*-MC* dataset) utilising language fragments. Unlike the work by Richardson and Sabharwal (2021) and Geiger et al. (2018), whose work was limited to only one language fragment, we explore a varied set of fragments. The consideration of linguistic complexity of language fragments led us to ask an additional question: *How does the linguistic complexity of the fragment affect the performance of the transformer model when performing model-checking with natural language?*

The contributions of this paper are as follows: (1) To the best of our knowledge, we are the first to broaden natural language reasoning over formal theories to include model-checking with natural language; (2) We develop a novel algorithm for constructing a dataset for model-checking with natural language; (3) We investigate whether transformers can learn to understand the logical semantics of natural language; (4) In a first-of-its-kind study in rule reasoning, we include complex fragments such as anaphora and relative clauses with transitive verbs; and (5) We provide a systematic analysis of how the linguistic complexity of language fragments affects rule reasoning.

2 Related Work

Our work follows the literature on evaluating neural approaches, especially transformer models on deductive and linguistic reasoning tasks (Richardson and Sabharwal, 2021; Richardson et al., 2020; Sinha et al., 2019; Geiger et al., 2018; McCoy et al., 2019; Betz et al., 2021). Moreover, it is also related to other research approaches that have been conducted on data synthesis for rule reasoning problems (Lin et al., 2019; Weston et al., 2016; Tafjord et al., 2019). However, our study is distinct from the above-mentioned work in two ways. Firstly, we focus on an unexplored problem space, model-checking with natural language. Secondly, unlike the above literature, we explore multiple language fragments and provide a deconstruction of how the linguistic complexity of language fragments affects the performance of transformer models in a rule reasoning task.

Our work can also be viewed as broadening the research conducted on training neural networks to perform algorithmic tasks, including learning to solve SAT problems (Selsam et al., 2019; Narodytska et al., 2020), propositional inference (Evans et al., 2018), semantic parsing (He and Choi, 2020; Kamath and Das, 2019), symbolic integration (Lample and Charton, 2020) and natural theorem proving (Weber et al., 2019; Minervini et al., 2020; Saha et al., 2020; Gontier et al., 2020). In our study, we aim to investigate the transformers' ability to emulate the algorithm governing the model-checking problem and comprehend the logical semantics of natural language.

When defining language fragments, we follow the definition set out by Pratt-Hartmann (Pratt-Hartmann, 2003, 2004; Pratt-Hartmann and Third, 2006; Pratt-Hartmann and Moss, 2009), who described it more precisely as a subset of a language equipped with semantics that translates sentences into a formal system such as first-order logic. Moreover we employ their work on fragments of first-order logic as the foundation when constructing the dataset. Notably, Pratt-Hartmann (2004) has investigated the complexity of fragments' first-order logic, and we limit our analysis in this paper to fragments that have been examined in that study. Moreover, we also closely follow the cognitive science literature on model-checking and quantifier verification (McMillan et al., 2005; Szymanik et al., 2013; Szymanik and Zajenkowski, 2010) when defining our experimental evaluation. It provides us with a baseline to compare results from transformer models with the empirical studies that have been conducted with humans.

3 Data Construction

To decide whether transformer models can learn to understand logical semantics of natural language from formulae (sentences) and structures represented in natural language, we developed an algorithm (shown in Algorithm 1) to construct a balanced dataset designed to be free from trivial linguistic patterns that are easily exploitable. This section will outline the data construction methodology in detail.

We sample a set of words (Proper nouns *PrN*, nouns *N*, verbs *Vb*, adjectives *Adj*) from a predefined vocabulary ($\mathcal{V}'$) to form a list of words $\mathcal{V}$. The proper nouns in $\mathcal{V}$ are used to define the domain $\mathcal{D}$ of the structure, while the nouns, verbs and adjectives are used for defining the set of predicates $\mathcal{P}$.

When generating sentences, we follow a template-based approach. A language template is a sentence of natural language with open-clause words replaced by schematic variables; for example, *Some N_1 V is every N_2*. Through substitution of vocabulary items of the appropriate category, we can generate natural language sentences, i.e., *Some artists admire every doctor*. A simple way of defining a fragment of natural language (language fragment) is via a finite set of template sentences. For example, the classical syllogistic fragment can be defined as the sentences confirming the sentence schemata, *All N_1 are N_2, Some N_1 is N_2, No N_1 are N_2* and *Some N_1 are not N_2*. The formula template is a formula of first-order logic with non-logical symbols replaced by schematic variables; for example, $\exists x(N_1(x) \wedge \forall N_2(B(y) \Rightarrow V(x, y)))$. An instance of that formula is the result of the substitution of non-logical symbols of the appropriate type for the schematic variables, i.e., $\exists x(artist(x) \wedge \forall y(doctor(y) \Rightarrow admire(x, y)))$. A language template translates to schematic formulae in a natural way. For example, the classical syllogistic translates to the schematic formulae $\forall x(N_1(x) \Rightarrow \pm N_2(x))$ and $\exists x(N_1(x) \wedge \pm N_2(x))$.

Let $\Phi_{\mathcal{L}}$ denote the set of first-order formula templates that can be translated to natural language templates $\mathcal{L}$. Given a language fragment $\mathcal{L}$ and a vocabulary $\mathcal{V}$, we can obtain a set of formulae $\Phi_{\mathcal{L}}(\mathcal{V})$, such that $\Phi_{\mathcal{L}}(\mathcal{V})$ is a fragment of first-order logic over the vocabulary $\mathcal{V}$, i.e., $\Phi_{\mathcal{L}}(\mathcal{V})$ only contains the vocabulary $\mathcal{V}$. The first-order formula ϕ is selected from the $\Phi_{\mathcal{L}}(\mathcal{V})$, and then the formula ϕ is translated to a natural language sentence s_ϕ using a template $\mathcal{L}$. A summary of the language fragments we used in our evaluation is provided in the next section.

Algorithm 1 Data Construction - Model-Checking with Natural Language

```
**Input :** Language Fragment $\mathcal{L}$ and its corresponding set of first order logic formulae templates $\Phi_{\mathcal{L}}$ along with its equivalent natural language templates $\mathcal{T}_{\mathcal{L}}$. Vocabulary $\mathcal{V}$ that contains Nouns(N), Adjectives(Adj), Verbs(Vb) and Proper nouns(PrN). Template $\mathcal{T}$ to convert structure to natural language. Maximum number of domain elements per datapoint $n$, and maximum number of predicates $m$
**Output :** Model-checking dataset $\mathcal{D}$

1: $D \leftarrow \{\}$
2: repeat
3:     $\mathcal{V} \leftarrow$ randomly select list of words where $\mathcal{V} \subset \mathcal{V}'$ such that $|\{PrN\}| \leq n$ and $|\{N \cup Adj \cup Vb\}| \leq m$
4:     $\phi \leftarrow$ randomly generate first order formula using the set of first-order logic formulae $\Phi_{\mathcal{L}}(\mathcal{V})$
5:     $s_\phi \leftarrow$ converts $\phi$ to a natural language sentence using the template $\mathcal{L}$
6:     $\ell \leftarrow$ randomly generate $\ell$ where $\ell \in \{True, False\}$
7:     $\mathcal{D} \leftarrow \{PrN\}$, $\mathcal{P} \leftarrow \{N \cup Adj \cup Vb\}$
8:     repeat
9:         $\mathfrak{A} \leftarrow$ generate structure randomly using the signature (vocabulary) $\mathcal{V}$
10:        if $(\ell = True$ and $\mathfrak{A} \models \phi)$ or $(\ell = False$ and $\mathfrak{A} \not\models \phi)$ then
11:            correct-structure-found $\leftarrow True$
12:        end if
13:    until correct-structure-found
14:    $\mathcal{M}_{\mathfrak{A}} \leftarrow$ converts $\mathfrak{A}$ to a natural language using a template $\mathcal{T}$
15:    D $\leftarrow$ D $\cup \{\mathcal{M}_{\mathfrak{A}}, s_\phi, \ell\}$
16: until stop condition is met
```

The label ℓ is selected randomly from the set $\{True, False\}$. Once ℓ and ϕ are defined, the structure $\mathfrak{A} = (\mathcal{D}, \{\mathcal{P}\}^{\mathfrak{A}})$ is generated, where $\mathcal{D}$ is the domain and $\mathcal{P}$ is the set of predicates and $\{\mathcal{P}\}^{\mathfrak{A}}$ represents an interpretation of $\mathcal{P}$ in $\mathfrak{A}$ and the signature of the structure is $\mathcal{V}$. Assignment of each domain element to the $\mathcal{P}$ in the structure $\mathfrak{A}$ is done randomly, such that for every domain element d_i in $\mathcal{D}$ and predicate $\mathcal{P}_i$, $prob(d_i \text{ assign to } \mathcal{P}_i) = p_1$ if $\mathcal{P}_i$ is a unary predicate, and for every domain element d_i, d_j in $\mathcal{D}$ and predicate $\mathcal{P}_i$, $prob((d_i, d_j) \text{ assign to } \mathcal{P}_i) = p_2$ if $\mathcal{P}_i$ is a binary predicate. In our experimentation, we select $p_1 = 0.5$ and $p_2 = 0.75^2$, so that for each predicate $\mathcal{P}_i$, $|\mathcal{P}_i^{\mathfrak{A}}|$ is a normal distribution with a mean of approximately $\frac{|\mathcal{D}|}{2}$, so the loop (in line 8-13) terminates within a reasonable time. We iteratively build structures randomly, and perform model-checking using a model-checker until a structure that meets the criteria defined by the label is found; i.e. if $\ell = True$, then the formula is $True$ according to the structure, $\mathfrak{A} \models \phi$ and vice-versa. Once such structure is identified, it is converted into a paragraph in natural language, $\mathcal{M}_{\mathfrak{A}}$ using a template $\mathcal{T}$.

Another way to create a data point would be to generate $\mathfrak{A}$ and ϕ and perform the validity check using a model-checker to acquire the label as opposed to pre-defining the label and iteratively constructing $\mathfrak{A}$ to match the label. However, such an approach can introduce easily exploitable linguistic patterns such as having the label $False$ for most sentences containing determiners *all*, *every* or *no*, or having label $True$ for sentences containing determiners *some* or *a*.

When constructing sentences, we make sure each predicate only appears once within a sentence. So sentences like *every artist is an artist* would not be generated. Furthermore, we also remove cases where no elements are assigned to a predicate P_i and perform re-balancing, since they also introduced easily exploitable patterns. For example, the sentence *"every musician who is a actor is happy"* is trivially $True$ if there are no *musicians* or *actors*.

3.1 Language fragments

A language fragment is defined as a language that is equipped with semantics that translates its sentences to a formal system, such as first-order logic. We defined our language fragments based on the work of Pratt-Hartmann (2004). Table 1 shows the language fragments used, along with an example for each fragment and the corresponding first-order formula. As indicated in the data construction algorithm, we employ a template-based approach when implementing both language fragments and their formal method representations. We limit our evaluations to fragments of first-order logic and bound

Language Fragment	Example
Syllogistic	Every musician is a artist $\forall x(musician(x) \Rightarrow artist(x))$
Relational syllogistic (Re-Syl)	All teachers remember some engineer $\forall x(artist(x) \Rightarrow \exists y(engineer(y) \wedge remember(x, y)))$
Relative clauses without transitive verbs (Rel)	All economists who are not happy are cynics $\forall x((economist(x) \wedge \neg happy(x) \Rightarrow cynic(x))$
Relative clauses with transitive verbs (Rel-TV)	No cynic like any scholar who is a expert $\forall x(cynic(x) \Rightarrow \forall y((scholar(y) \wedge cynic(y)) \Rightarrow \neg like(x, y)))$
Anaphora	Some judge warns no juror who hate him $\exists x(judge(x) \wedge \forall y((juror(y) \wedge hate(y, x)) \Rightarrow \neg warn(x, y)))$

Table 1: Language fragments we utilised along with an example for each of them and its corresponding first-order logic formula.

the number of functions within a formula to have a maximum of four. We also limit the maximum number of quantifiers per formula to two, producing only unary or binary formulae. The rationale is to have natural sounding sentences. As outlined in the description of the template structure provided in **Appendix A: Templates of Language Fragments**, to address the ambiguity that can arise with anaphora or relative clauses, we bind the anaphora or relative clauses to the same element. For example, anaphora always refer to the first noun in the sentence. Even though we limit our data construction and evaluation to only these fragments, we emphasise that the data construction methodology and experimental evaluation we have conducted can be executed with any arbitrary fragment of natural language.

3.2 Boolean Coordinators

One interesting experiment is to evaluate how transformer models perform when Boolean coordinators are introduced to the sentences. To that end, we used Boolean coordinators ($\bigwedge(and), \bigvee(or)$) to combine sentences and create more difficult problem instances. The resultant first-order formula Ψ of such sentences can be formed by combining individual first-order formulae using either $\bigwedge$ or $\bigvee$. Model-checking is then performed on Ψ (is $\mathfrak{A} \models \Psi$ or $\mathfrak{A} \not\models \Psi$?), and the condition in Algorithm 1 *(line 11)* is modified accordingly.

The natural language sentences are combined accordingly using the coordinating conjunctions *and* or *or*. We did not consider the case where *and* as well as *or* are present in the final sentence, since the order of operations cannot be enforced in natural language settings and hence would be ambiguous. We evaluated transformer models varying the number of coordinators *k*, where $k = \{0, 1, 2\}$, to investigate how incorporating Boolean coordinators affect the accuracy.

4 Experimental Setup

In this section, we describe the experiments we conducted in order to address our research questions.

4.1 Problem definition

Formally the $\mathcal{FO}^2$-*MC* dataset can be defined as $\{(p^{(d)}, \ell^{(d)})\}_d^{|D|}$ where $p^{(d)}$ is an instance of the model-checking problem (concatenation of the structure $\mathcal{M}_{\mathfrak{A}}$ and sentence s_ϕ delimited by a separator *SEP* token), and $\ell \in \{True, False\}$ is the label. The task is to correctly predict the label ℓ, thereby reducing it to a binary classification problem.

4.2 Transformer models

To investigate the capabilities of transformers in model-checking with natural language, we performed experiments on the $\mathcal{FO}^2$-*MC* dataset using three prominent transformer architectures: BERT, RoBERTa and T5.

BERT. Bidirectional Encoder Representations from Transformers or BERT (Devlin et al., 2018) use bi-directional conditioning in all of its network's layers to consider both the left and right context. BERT has become the standard transformer architecture and has been evaluated against many NLI datasets (Richardson et al., 2020; McCoy et al., 2019), hence we believe it provides a baseline for assessing the complexity of the task and difficulty of the $\mathcal{FO}^2$-*MC* dataset. We used the `BERT-base` `(uncased)` model with around 110M parameters.

RoBERTa. Robustly Optimized BERT Pretraining Approach or RoBERTa (Liu et al., 2019) is based on the BERT architecture but trained in a more optimised manner. It has been used for rule reasoning tasks such as RuleTaker (Clark et al., 2021), and is considered as another baseline model in our experiments. We made use of the `RoBERTa-base` model which has around 125M parameters.

T5. Following the work done by Tafjord et al. (2021) and Richardson and Sabharwal (2021) on rule reasoning, we primarily centre our experiments around Text-to-Text Transfer Transformer or T5 models (Raffel et al., 2019). T5 frames all NLP tasks (e.g., classification, translation, semantic textual similarity) into a unified text-to-text format where both input and output are always strings; this is slightly different from BERT and RoBERTa which, when fine-tuned on classification tasks, output a class label. In our experiments, we employed two T5 models: `T5-base` with approximately 220M parameters and `T5-large` with approximately 700M parameters.

In experimenting with each of the three types of models above, we utilised the Huggingface library (Wolf et al., 2019). The transformer models are fine-tuned to predict the target label ($True$ or $False$) by optimising for the binary cross-entropy loss over the targets using the Adam optimiser (Kingma and Ba, 2015). Since the dataset is balanced (i.e., both training and test data have approximately an equal number of samples labelled as $True$ and $False$), we made use of accuracy as our evaluation metric.

4.3 Proposed Dataset and Evaluation

To answer the question of whether transformers can perform model-checking with natural language, we trained transformer models, namely, T5, BERT and RoBERTa, using the $\mathcal{FO}^2$-*MC* dataset in the manner mentioned above. During data construction, we incorporated the same vocabulary as Richardson and Sabharwal (2021), with the addition of transitive verbs where the number of verbs is equivalent to the number of adjectives. The vocabulary contains approximately 2000 names (proper nouns), 156 nouns, 64 adjectives and 65 verbs. The names are used as the domain elements while nouns, adjectives and verbs form predicates, whereas verbs constitute binary predicates while the nouns and adjectives form unary predicates. Furthermore, when generating the model, we limited the number of domain elements to be less than 4 ($|\mathcal{D}| \leq 4$) and the number of predicates to be less than 8 ($|\mathcal{P}| \leq 8$). We trained transformer models using training instances that include sentences that belong to language fragments we introduced in Section 3.1, i.e., syllogistic, relational syllogistic (Re-Syl), relative clauses (Rel), relative clauses with transitive verbs (Rel-TV) and anaphora. In each case (for each language fragment), models were trained with 500K unique data points and evaluated against a held-out 100K test set (see Table 2). Moreover, we experimented with training the models (`T5-base` and `T5-large`) using a dataset that contains sentences belonging to all the fragments, so that we could investigate how simpler fragments help transformers understand the logical semantics of natural language of complex ones (see Table 3). The training set in this experiment comprises 500K unique data points with approximately 100K data points belonging to each language fragment. The results of this experiment, along with the results depicted in Table 2, also provide the answer to the question of how the linguistic complexity of the language fragment affects the performance of transformers in model-checking with natural language. To better understand model generalisation and scale invariance, we evaluated the transformer model (`T5-large`) on a held-out evaluation set whose structure contains more domain elements (see Table 4) or more predicates (see Table 5) than that of the training set. To comprehend how Boolean coordinators affect the accuracy of transformer models across different language fragments, we also trained and evaluated with data points whose sentences have Boolean coordinators in them.

4.4 Results and discussion

Transformer models can solve model-checking with natural language problems with satisfactory accuracy, given adequate training instances, as depicted in Table 2. For all language fragments, transformers manage to yield an accuracy of over 70%. It is also evident from Table 2 that there is no significant difference in performance between the considered transformer models.

The linguistic complexity of the language fragments that generate the sentence has a significant impact on the overall performance, as il-

Model	Syllogistic	Re-Syl	Rel	Rel-TV	anaphora
T5-base	99.9	76.6	95.0	73.6	70.3
BERT-base	99.0	78.2	90.7	75.6	73.9
RoBERTa-base	99.6	79.2	90.1	72.1	71.0

Table 2: Accuracy of transformer models (BERT, T5 and RoBERTa) across different language fragments.

Model	All	Syllogistic	Re-Syl	Rel	Rel-TV	anaphora
T5-base	75.9	80.0	76.7	74.8	74.2	73.6
T5-large	88.2	99.8	81.8	99.3	82.3	77.7

Table 3: The transformers are trained using a dataset that contains sentences belonging to all language fragments. The results are broken down into respective language fragments, and *All* indicates the overall (average) accuracy across the language fragments.

Language Fragment	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{D}\rvert$+1	$\lvert\mathcal{D}\rvert$+2	$\lvert\mathcal{D}\rvert$+4
Syllogistic	99.8	92.2	87.5	76.1
Re-Syl	81.8	67.6	63.2	55.6
Rel	99.3	90.4	84.2	73.3
Rel-TV	82.3	67.3	62.1	56.4
anaphora	77.7	65.0	61.1	49.9

Table 4: The accuracy of the `T5-large` model evaluated on out-of-scope data; the training instances have a maximum of 4 domain elements $\lvert\mathcal{D}\rvert \leq 4$ while the evaluation set contains 5 ($\lvert\mathcal{D}\rvert + 1$), 6 ($\lvert\mathcal{D}\rvert + 2$), 8 ($\lvert\mathcal{D}\rvert + 4$) domain elements, the number of predicates remains the same between train and evaluation sets.

lustrated in Tables 2 and 3. Transformers achieve near-perfect performance for fragments such as syllogistic and Rel. However, they only achieve a moderate level of accuracy for fragments such as Re-Syl, Rel-TV, and anaphora. The later-mentioned fragments have transitive verbs, which results in the respective structures containing binary relationships. It is harder to learn binary relationships as opposed to unary ones. Furthermore, the sentences in the fragments Re-Syl, Rel-TV, and anaphora can have two quantifiers, while the sentences in fragments syllogistic and Rel are restricted to only one. As depicted in Table 6, the number of quantifiers in the sentence influences the performance of the transformer models in solving model-checking problems. Sentences with two quantifiers are more difficult to decode than sentences with only one quantifier, as evidenced by cognitive studies on quantifier verification (Szymanik and Zajenkowski, 2010; Szymanik et al., 2013), which is also unsurprising. There is a difference between the accuracy of single quantifier sentences and the average accuracy of the syllogistic fragment and Rel fragment. This difference is due to single quantifier sentences belonging to other fragments, whose sentences include transitive verbs. According to the results in Table 6, only the number of quantifiers seems to affect the performance of the transformers, and not the exact quantifier used. Cognitive studies (Szymanik et al., 2013) suggest quantifiers themselves affect human performance on model-checking problems, which is not evident here, implying human reasoning on language is somewhat different to what is occurring in transformer models.

Another linguistic property that seems to affect the performance of transformers is Boolean coordinators. **The accuracy of transformer models decreases when Boolean coordinators are introduced to the sentences**, as illustrated in Table 7. The difference in performance when the number of coordinators changes from 0 to 1 is higher than that of when it is increased from 1 to 2. However, the number of Boolean coordinators has a lower effect on accuracy compared to other linguistic properties such as the number of quantifiers.

Learning the simple fragments enables transformers to learn complex ones, as depicted in Table 3. When training data contains sentences from all the language fragments, the performance of complex fragments such as anaphora is higher than if it only includes sentences of that respective fragment. Transformer models can learn to understand logically significant words such as determiners and grammatical constructs such as relative clauses more easily from simpler fragments than complex ones. So when learning the logical semantics of complex fragments, transformers can employ this knowledge. Hence, we can hy-

Language Fragment	$\lvert\mathcal{P}\rvert$	$\lvert\mathcal{P}\rvert$+1	$\lvert\mathcal{P}\rvert$+2	$\lvert\mathcal{P}\rvert$+4
Syllogistic	99.8	96.6	96.3	95.9
Re-Syl	81.8	80.4	80.4	80.3
Rel	99.3	99.1	99.1	98.9
Rel-TV	82.3	81.4	81.2	80.6
anaphora	77.7	76.9	76.6	76.6

Table 5: The accuracy of the `T5-large` model evaluated on out-of-scope data; the training instances have a maximum of 8 predicates $\lvert\mathcal{P}\rvert \leq 8$ while the evaluation set contains 9 ($\lvert\mathcal{P}\rvert + 1$), 10 ($\lvert\mathcal{P}\rvert + 2$), 12 ($\lvert\mathcal{P}\rvert + 4$) predicates, the number of domain elements remains unchanged

number of quantifiers	quantifier (s)	Accuracy
one	overall	95.6
	$\forall$ (all)	95.8
	$\exists$ (some)	95.4
two	overall	80.8
	$\forall \circ \forall$ (all - all)	80.4
	$\forall \circ \exists$ (all - some)	81.8
	$\exists \circ \forall$ (some - all)	81.2
	$\exists \circ \exists$ some - some)	80.2

Table 6: The change in accuracy of the `T5-large` model across different quantifiers. The syllogistic and Rel fragments contain only one quantifier, while Re-Syl, Rel-TV, and anaphora fragments can have two quantifiers.

number of Boolean coordinators	Accuracy
$k = 0$	75.9
$k = 1$	70.7
$k = 2$	67.6

Table 7: The accuracy of the T5-base model when trained and evaluated against problem instances that have Boolean coordinators. k denotes the number of Boolean coordinators in a sentence. Each sentence contains only one type of coordinator (either *and* or *or*), if any.

pothesise that transformers at least partially learn to understand the essence of logical semantics of natural language. Table 3 also indicates a substantial difference in performance between the `T5-base` model and the `T5-large` model. The `T5-large` model achieves an overall accuracy of 88.2% but only manages to achieve an accuracy of around 80% (Re-Syl:81.8%, Rel-TV: 82.3%, anaphora: 77.7%) for language fragments with transitive verbs. This accuracy level is lower than the accuracy that transformer models yielded in other rule reasoning benchmarks such as RuleTaker (Clark et al., 2021) and NLSat (Richardson and Sabharwal, 2021), which suggests that the $\mathcal{FO}^2$-*MC* dataset is a formidable linguistic reasoning benchmark.

Transformer models exhibit limited generalisation and scale-invariance, as illustrated in Tables 4 and 5. Even if the number of predicates increases, the accuracy of the transformer model remains relatively unchanged (see Table 5). However, if the number of domain elements increases, the model performance drastically decreases (see Table 4). The reason could be that the attention mechanism in the transformer correctly identifies which areas in the structure to examine for a given sentence, but the transformer model still cannot emulate the model-checking algorithm properly. Moreover, the degradation in performance is relatively equivalent for all language fragments, suggesting that decrement is not correlated to the grammatical structure of the sentence. Hence, we can conjecture that transformers can learn to understand the logical semantics of natural language but still cannot learn to emulate the underlying model-checking algorithm.

5 Conclusion

We investigate the limits of transformers in an unexplored problem space of model-checking with natural language employing language fragments. We use five different language fragments and explore how linguistic complexity and other linguistic properties such as Boolean coordinators affect rule reasoning in transformer models. In a broader sense, our study is to determine whether transformer models can learn to understand the logical semantics of natural language and emulate the model-checking algorithm. We posit that transformers can learn logically significant words and grammatical constructs but fall short when learning the underlying algorithm. Moreover, different linguistic properties such as the language fragment, Boolean coordinators and the number of quantifiers have a notable impact on the learning ability of the transformers. Thus, an interesting future direction would be to investigate how these linguistic properties affect more complex reasoning tasks like natural language satisfiability.

6 Limitations

The results in our work closely follow the trends reported by prior work in the domain of identifying the limits of transformers in logical reasoning. Specifically, the transformers exhibit limited generalization beyond the underlying distribution in training data. However, due to the empirical nature of the study, it is not guaranteed that all other transformer-based models or other neural networks would exhibit the same pattern.

Moreover, the study focuses on several language fragments with varying linguistic complexity such that one would be able to quantify the influence of linguistic properties on a logical reasoning problem. However, the fragments considered in this study are not the only language fragments in existence and, as such, would limit the comprehensiveness of the discussion, and there could be other fragments of language which behave differently when evaluated against transformer models.

References

Gregor Betz, Kyle Richardson, and Christian Voigt. 2021. Thinking aloud: Dynamic context generation improves zero-shot reasoning performance of gpt-2. *arXiv preprint arXiv:2103.13033*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2021. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Richard Evans, David Saxton, David Amos, Pushmeet Kohli, and Edward Grefenstette. 2018. Can neural networks understand logical entailment? In *International Conference on Learning Representations*.

Atticus Geiger, Ignacio Cases, Lauri Karttunen, and Christopher Potts. 2018. Stress-testing neural models of natural language inference with multiply-quantified sentences. *arXiv preprint arXiv:1810.13033*.

Nicolas Gontier, Koustuv Sinha, Siva Reddy, and Chris Pal. 2020. Measuring systematic generalization in neural proof generation with transformers. *Advances in Neural Information Processing Systems*, 33:22231–22242.

Han He and Jinho D. Choi. 2020. Establishing Strong Baselines for the New Decade: Sequence Tagging, Syntactic and Semantic Parsing with BERT. In *Proceedings of the 33rd International Florida Artificial Intelligence Research Society Conference*, FLAIRS'20, pages 228–233.

Aishwarya Kamath and Rajarshi Das. 2019. A survey on semantic parsing. In *Automated Knowledge Base Construction (AKBC)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Guillaume Lample and François Charton. 2020. Deep learning for symbolic mathematics. In *International Conference on Learning Representations*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 58–62, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Corey T. McMillan, Robin Clark, Peachie Moore, Christian Devita, and Murray Grossman. 2005. Neural basis for generalized quantifier comprehension. *Neuropsychologia*, 43:1729–1737.

Pasquale Minervini, Matko Bosnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette. 2020. Differentiable reasoning on large knowledge bases and natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI February 7-12, 2020*, pages 5182–5190. AAAI Press.

Nina Narodytska, Hongce Zhang, Aarti Gupta, and Toby Walsh. 2020. In search for a sat-friendly binarized neural network architecture. In *International Conference on Learning Representations*.

Ian Pratt-Hartmann. 2003. A two-variable fragment of english. *Journal of Logic, Language and Information*, 12(1):13–45.

Ian Pratt-Hartmann. 2004. Fragments of language. *Journal of Logic, Language and Information*, 13(2):207–223.

Ian Pratt-Hartmann. 2010. Computational complexity in natural language. *The handbook of computational linguistics and natural language processing*, page 43.

Ian Pratt-Hartmann and Lawrence S Moss. 2009. Logics for the relational syllogistic. *The Review of Symbolic Logic*, 2(4):647–683.

Ian Pratt-Hartmann and Allan Third. 2006. More fragments of language. *Notre Dame Journal of Formal Logic*, 47(2):151–177.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Kyle Richardson, Hai Hu, Lawrence Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8713–8721.

Kyle Richardson and Ashish Sabharwal. 2021. Pushing the limits of rule reasoning in transformers through natural language satisfiability. *arXiv preprint arXiv:2112.09054*.

Swarnadeep Saha, Sayan Ghosh, Shashank Srivastava, and Mohit Bansal. 2020. PRover: Proof generation for interpretable reasoning over rules. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 122–136, Online. Association for Computational Linguistics.

Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, and David L. Dill. 2019. Learning a sat solver from single-bit supervision. In *International Conference on Learning Representations*.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. 2019. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4506–4515, Hong Kong, China. Association for Computational Linguistics.

Jakub Szymanik, Shane Steinert-Threlkeld, Marcin Zajenkowski, and Thomas F Icard III. 2013. Automata and complexity in multiple-quantifier sentence verification. *Logic and Interactive RAtionality Yearbook 2012*, page 133.

Jakub Szymanik and Marcin Zajenkowski. 2010. Comprehension of simple quantifiers: Empirical evaluation of a computational model. *Cognitive Science*, 34(3):521–532.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019. Quartz: An open-domain dataset of qualitative relationship questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5941–5946, Hong Kong, China. Association for Computational Linguistics.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690*.

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. 2019. NLProlog: Reasoning with weak unification for question answering in natural language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6151–6161, Florence, Italy. Association for Computational Linguistics.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomás Mikolov. 2016. Towards ai-complete question answering: A set of prerequisite toy tasks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Sub-fragment	Natural Language Template	First order logic formula
with a quantifier	*D (non-)N_1 is/are (not) (a) N_2*	$\forall x(\pm N_1(x) \Rightarrow \pm N_2(x))$ *or* $\exists x(\pm N_1(x) \land \pm N_2(x))$

Table 8: Templates for the syllogistic fragment, D denotes the determiner while N_1 and N_2 symbolise nouns.

Sub-fragment	Natural Language Template	First order logic formula
dual quantifiers	*D_1 (non-)N_1 (does not/do not) V D_2 (non-)N_2*	$\forall x(\pm N_1(x) \Rightarrow \forall y(\pm N_2(x) \Rightarrow \pm V(x,y)))$ *or* $\forall x(\pm N_1(x) \Rightarrow \exists y(\pm N_2(y) \land \pm V(x,y)))$ *or* $\exists x(\pm N_1(x) \land \forall y(\pm N_2(x) \Rightarrow \pm V(x,y)))$ *or* $\exists x(\pm N_1(x) \land \exists y(\pm N_2(x) \land \pm V(x,y)))$
With Proper nouns quantifier in the subject	*D N (does not/do not) V P*	$\forall x(\pm N(x) \Rightarrow \pm V(x,P))$ *or* $\exists x(\pm N(x) \land \pm V(x,P))$
With Proper nouns quantifier in the object	*P (does not/do not) V D N*	$\forall x(\pm N(x) \Rightarrow \pm V(P,x))$ *or* $\exists x(\pm N(x) \land \pm V(P,x))$

Table 9: Templates for the relational syllogistic fragment, D_1 and D_2 denote determiners, P denotes Proper nouns and V represents the verb while N, N_1 and N_2 symbolise nouns.

A Templates of Language Fragments

Tables 8, 9, 10, 11 and 12 contain templates for the syllogistic fragment, relational syllogistic fragment, relative clauses fragment (without transitive verbs), relative clauses (with transitive verbs), and anaphora respectively. Each table contains natural language templates that are employed to construct sentences and their corresponding first-order formulae. As mentioned in the methodology section, we build upon the vocabulary from Richardson and Sabharwal (2021). The set of determiners includes all, every, some, a and no, where every sentence type is converted to the most natural-sounding sentences; i.e. sentences such as *every artist does not like every beekeeper* would be translated into *no artists like any beekeeper*. Each sentence that is rendered using templates of the syllogistic fragment and relative clause (without transitive verbs) fragment would include exactly one quantifier, which would determine the determiner of the sentence. The templates of the relational syllogistic, relative clause (with transitive verbs) and anaphora could comprise either two quantifiers or one (if the sentence contains proper nouns, then it would have only one quantifier).

Sub-fragment	Natural Language Template	First order logic formula
with a quantifier	*D (non-)N_1 who is/are (not) (a) N_2/A_1 is/are (not) (a) N_3/A_2*	$\forall x.(\pm N_1(x) \land \pm N_2/A_1(x) \Rightarrow \pm N_3/A_2(x))$ *or* $\exists x.(\pm N_1(x) \land \pm N_2/A_1(x) \land \pm N_3/A_2(x))$

Table 10: Templates for the relative clauses (without transitive verbs) fragment, D denotes the determiner and N_1, N_2 and N_3 symbolise nouns while A_1 and A_2 represent adjectives.

Sub-fragment	Natural Language Template	First order logic formula
dual quantifiers, relative clause in the subject	*D_1 (non-)N_1 who (does not/ do not) V D_2 (non-)N_2 is/are (not) (a) N_3*	$\forall x(\pm N_1(x) \land \forall y(\pm N_2(y) \Rightarrow \pm V(x,y)) \Rightarrow \pm N_3(x))$ *or* $\forall x(\pm N_1(x) \land \exists y(\pm N_2(y) \land \pm V(x,y)) \Rightarrow \pm N_3(x))$ *or* $\forall x(\pm N_1(x) \land \forall y(\pm N_2(y) \Rightarrow \pm V(x,y)) \Rightarrow \pm N_3(x))$ *or* $\forall x(\pm N_1(x) \land \forall y(\pm N_2(y) \Rightarrow \pm V(x,y)) \Rightarrow \pm N_3(x))$
dual quantifiers, relative clause in the object	*D_1 (non-)N_1 (does not/do not) V D_2 (non-)N_2 who is/are (not) (a) N_3*	$\forall x(\pm N_1(x) \Rightarrow \forall y((\pm N_2(y) \land \pm N_3(y)) \Rightarrow \pm V(x,y)))$ *or* $\forall x(\pm N_1(x) \Rightarrow \exists y(\pm N_2(y) \land \pm N_3(y) \land \pm V(x,y)))$ *or* $\exists x(\pm N_1(x) \land \forall y((\pm N_2(y) \land \pm N_3(y)) \Rightarrow \pm V(x,y)))$ *or* $\exists x(\pm N_1(x) \land \exists y(\pm N_2(y) \land \pm N_3(y) \land \pm V(x,y)))$
with Proper nouns	*D (non-)N_1 who (does not/ do not) V P is/are (not) (a) N_2*	$\forall x(\pm N_1(x) \land \pm V(x,P) \Rightarrow \pm N_2(x)$ *or* $\exists x(\pm N_1(x) \land \pm V(x,P) \land \pm N_2(x))$

Table 11: Templates for the relative clauses (with transitive verbs) fragment, D_1 and D_2 denote determiners, P denotes Proper nouns and V represents the verb while N_1 N_2 and N_3 symbolise nouns.

Sub-fragment	Natural Language Template	First order logic formula
dual quantifiers	*D_1 (non-)N_1 (does not/do not) V_1 D_2 (non-)N_2 who (does not /do not) V_2 him/her/them*	$\forall x(\pm N_1(x) \Rightarrow \forall y(\pm N_2(y) \land \pm V_2(y,x) \Rightarrow \pm V_1(x,y)))$ *or* $\forall x(\pm N_1(x) \Rightarrow \exists y(\pm N_2(y) \land \pm V_2(y,x) \land \pm V_1(x,y)))$ *or* $\exists x(\pm N_1(x) \land \forall y(\pm N_2(y) \land \pm V_2(y,x) \Rightarrow \pm V_1(x,y)))$ *or* $\exists x(\pm N_1(x) \land \exists y(\pm N_2(y) \land \pm V_2(y,x) \land \pm V_1(x,y)))$
With Proper nouns	*P (does not/do not) V_1 D (non-)N who (does not/do not) V_2 him/her*	$\forall x(\pm N(x) \land \pm V_2(x,P) \Rightarrow \pm V_1(P,x))$ *or* $\exists x(\pm N(x) \land \pm V_2(x,P) \land \pm V_1(P,x)$

Table 12: Templates for the relative clauses (with transitive verbs) fragment, D_1 and D_2 denote determiners, P denotes Proper nouns and V_1 and V_2 represent verbs while N, N_1 and N_2 symbolise nouns.

Improving the Generalizability of Collaborative Dialogue Analysis With Multi-Feature Embeddings

Ayesha Enayet
University of Central Florida
Orlando, FL USA
`ayeshaenayet@knights.ucf.edu`

Gita Sukthankar
University of Central Florida
Orlando, FL USA
`gitars@eecs.ucf.edu`

Abstract

Conflict prediction in communication is integral to the design of virtual agents that support successful teamwork by providing timely assistance. The aim of our research is to analyze discourse to predict collaboration success. Unfortunately, resource scarcity is a problem that teamwork researchers commonly face since it is hard to gather a large number of training examples. To alleviate this problem, this paper introduces a multi-feature embedding (MFeEmb) that improves the generalizability of conflict prediction models trained on dialogue sequences. MFeEmb leverages textual, structural, and semantic information from the dialogues by incorporating lexical, dialogue acts, and sentiment features. The use of dialogue acts and sentiment features reduces performance loss from natural distribution shifts caused mainly by changes in vocabulary.

This paper demonstrates the performance of MFeEmb on domain adaptation problems in which the model is trained on discourse from one task domain and applied to predict team performance in a different domain. The generalizability of MFeEmb is quantified using the similarity measure proposed by Bontonou et al. (2021). Our results show that MFeEmb serves as an excellent domain-agnostic representation for meta-pretraining a few-shot model on collaborative multiparty dialogues.

1 Introduction

For many natural language processing applications, the ability to learn features that generalize well across multiple datasets is a key desideratum (Saikia et al., 2020). This paper introduces a new multi-feature embedding, MFeEmb, that increases the generalizability of models learned from collaborative multiparty dialogues. Dialogues are different from single-author documents in that, along with textual information, they contain communication patterns that may serve as indicators of social dynamics. Treating a dialogue as a mere text collection ignores valuable information. We advocate exploiting implicit features present in multiparty dialogues that are less vulnerable to distribution shifts resulting from task domain changes.

This paper demonstrates the usage of MFeEmb on a communication analysis task: conflict prediction. Teamwork research faces a challenge of resource scarcity since the human subjects datasets are quite small (less than 100 samples), due to the difficulty of recruiting teams and the time consuming nature of many group tasks. A variety of social phenomena have been investigated within team communication research, including entrainment (Rahimi and Litman, 2020) and emergent leadership (Maese et al., 2021). Frequency of communication is not in itself a good predictor of team performance, but a meta-analysis conducted by Marlow et al. (2018) that drew upon data from 150 studies conducted on 9702 teams concluded that high quality communication is positively related to team performance in many task domains. Conversely, process conflict, "disagreement among group members about the content of the tasks being performed, including differences in viewpoints, ideas, and opinions" (Jehn, 1995), is usually negatively correlated with taskwork success.

Our aim is to be able to learn a model to classify process conflict from multiparty dialogues that generalizes well across multiple tasks. We treat the task of conflict prediction as a binary classification task with high conflict and low conflict being the two classes; the ground truth used by the conflict prediction model is measured using a post-task team process conflict survey. This paper focuses on three collaborative problem-solving tasks: software engineering, search and rescue, and cooperative gameplay.

Our proposed embedding, MFeEmb, leverages textual, structural, and semantic information from the dialogues by incorporating vocabulary, dialogue acts, and sentiment features. Lexical embed-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3551–3565
May 2-6, 2023 ©2023 Association for Computational Linguistics

dings such as word2Vec and BERT (Devlin et al., 2018) show good performance across multiple NLP tasks on in-domain test sets but are less robust to domain shift. Previous work identified that dialogue acts and sentiment sequences are informative features that predict conflict reliably even at the earliest stage of team problem-solving (Enayet and Sukthankar, 2021a); however, classifiers constructed using these features still experience lackluster transfer performance when applied to new datasets, particularly when detecting high conflict examples (Enayet and Sukthankar, 2021b).

To address this transfer problem, we propose the usage of MFeEmb, specifically as a meta-pretraining representation to be used within a few-shot model. MFeEmb combines the strengths of both domain-invariant and domain-specific features. This paper compares the generalizability potential of the MFeEmb embedding vs. standard word embeddings using inter-class and intra-class based similarity measures, proposed by Bontonou et al. (2021). Then we evaluate the performance of MFeEmb in a domain adaptation scenario in which the model is trained on discourse from one task domain and used to predict conflict in a different domain. Our results show that:

1. MFeEmb demonstrates superior generalizability over other embeddings for collaborative multiparty dialogues.
2. MFeEmb is an excellent representation choice for the meta-training stage of few-shot learning.
3. The domain adaptation performance of MFeEmb can be easily enhanced by task specific synonym replacement.

2 Related Work

Previous studies on group interaction tasks such as conflict prediction (Rahimi and Litman, 2020), disruptive talk detection (Park et al., 2022), group satisfaction (Lai and Murray, 2018), and task performance prediction (Kubasova et al., 2019; Murray and Oertel, 2018) have focused on simply improving performance on in-domain datasets. Very little attention has been paid to the problem of creating generalizable models for multiparty dialogue that can be used when training data is scarce. The intelligent tutoring system community has empirically assessed the generalizability of common natural language representations, such as BERT and Linguistic Inquiry Word Count (LIWC), across collaborative problem solving tasks (Pugh et al., 2022), but without investigating methods to improve generalizability.

In domain adaptation, the goal is to train a model on data from a source domain that performs well on a test dataset drawn from a different target distribution. Common NLP tasks (e.g., part-of-speech (POS) tagging and named entity recognition (NER)) have been tackled using techniques including instance weighting (Jiang and Zhai, 2007) or explicitly identifying feature correspondences between the domains (Blitzer et al., 2006). An alternate approach is to learn a single representation that generalizes well across multiple domains. This can be done using few-shot learning (Wang et al., 2020), one of the most widely used approaches to dealing with resource scarcity. The traditional framework comprises meta-training and meta-testing phases, where the aim of meta-training is to learn universal representations from multiple domains.

Triantafillou et al. (2021) introduced a method that improves few-shot generalizability by making use of multiple datasets in order to learn a universal template. Dvornik et al. (2020) proposed Selecting from Universal Representations (SUR), which involves learning a multi-domain representation by training multiple feature extractors. A multi-domain feature bank is used to select the most relevant feature during the learning phase. Rather than seeking to learn the new representation entirely from data, our research exploits similarities in dialogue act sequences and sentiment patterns commonly observed during successful collaborative problem-solving.

Representation choice has been shown to place an upper bound on target domain performance (Ben-David et al., 2006). Few-shot frameworks such as Meta-pretraining then Meta-Learning (MTM) (Deng et al., 2019) have assumed that word embeddings like BERT that are trained on large datasets are the best choice for task agnostic pre-training. Bontonou et al. (2021) introduced a method to quantify the generalizability of a few-shot classifier under supervised, unsupervised and semi-supervised settings. This paper uses their inter-class and intra-class based generalizability measure to evaluate MFeEmb vs. simple word-based embeddings under supervised classification scenarios. Our research demonstrates that MFeEmb is superior to word embeddings as a meta-pretraining representation.

3 Methodology

This section describes our approach to learning a generalizable embedding from multi-party dialogues for conflict prediction. We discuss our datasets, introduce our embedding, and show how our technique can be used in combination with data augmentation and few-shot learning.

3.1 Datasets

Datasets collected from different collaborative problem-solving task domains were used in our study of generalizability:

1. **Teams corpus** (Litman et al., 2016): This dataset consists of dialogues from 62 teams playing a cooperative board game in groups of three or four. Each team plays the game twice together. The Teams corpus was originally created to study entrainment, a linguistic phenomena in which teammates adopt similar speech patterns (Rahimi and Litman, 2020). The Game1 dataset of Teams corpus contains 62 dialogues, 32 low conflicts, and 30 high conflict dialogues. The Game2 dataset of Teams corpus contains 62 dialogues, 33 low conflicts, and 29 high conflict dialogues.
2. **ASIST dataset** (Huang et al., 2022): This dataset consists of 67 teams of three people participating in a simulated search and rescue task within the Minecraft game environment. Participants completed two different missions that involved searching a map and triaging victims. The dataset was collected by the ASIST project to stimulate the development of proactive assistant agents for helping human teams. The dataset contains 113 dialogues, 58 low conflicts, and 55 high conflict dialogues.
3. **GitHub social coding dataset** (Enayet and Sukthankar, 2020): This dataset was mined directly from the GitHub social coding platform. It consists of data from issue comments of teams developing open source software over a period of months. Teams vary in size, and comments were harvested for 50 reported issues. The dataset contains 50 dialogues, 29 low conflicts, and 21 high conflict dialogues.

Both the Teams and ASIST datasets contain post-task process conflict survey data for all teams, which we divide into high and low conflict groups using their z-scores. For GitHub, process conflict was scored according to an issue resolution rubric (described in Appendix G).

3.2 Multi-Feature Embedding (MFeEmb)

This paper introduces the MFeEmb embedding which is designed to capture the dialogues' structural, semantic, and textual information for collaborative task success prediction. To represent the structural information, we incorporate information from dialogue acts (DAs) of the utterances. For semantics, the sentiment polarities of the utterances are used, although DAs capture both semantic and structural information. Textual information is extracted from the vocabulary of the dialogues.

For the word embedding, we use both the Distributed Bag of Words and Dynamic Memory models of Doc2Vec (Le and Mikolov, 2014) to learn embeddings (see Appendix B). Although there is only 28% vocabulary overlap between the ASIST and Teams datasets and 35% overlap between the GitHub and Teams datasets, word embeddings can help preserve high performance on the in-domain dataset while including structural and semantic features makes the embedding more robust to domain shifts.

For the dialogue act (DA) embedding, we first map the sequence of utterances to a sequence of DAs using USE-DAC (Universal Sentence Encoder Dialogue Act Classifier, described in Appendix A). The SwDA-DAMSL tagset was used to categorize dialogue acts. The TextBlob python module was used to assign sentiment polarities ranging from -1 to 1 to each of the utterances.

To generate the embeddings, we use the Dynamic Memory model of Doc2Vec due to the small vocabulary size of the sequences, which is limited by the number of DA tags and sentiment gradations. The Dynamic Memory model leverages context when generating embeddings, thus preserving information contained in these communication patterns. In contrast, the Distributed Bag of Words model does not consider the context when generating embeddings. For the few-shot results, we also report results with pre-trained Word2Vec embeddings. First, we separately learn three embeddings from the sequence of DAs, sentiments, and utterances (text); the final MFeEmb embedding is created either by concatenating the three embeddings or by using LSTMs to learn a concatenation ensemble model. We have made our code available at `https://github.com/ayeshaEnayet/MFeEmb.git`.

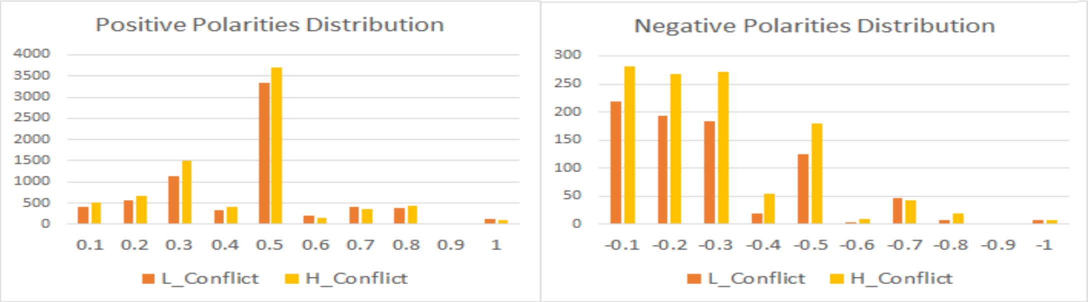


Figure 1: Sentiment polarity distribution of the high conflict vs. low conflict classes in the Teams dataset

3.3 Corpus-Based Feature Analysis

To understand the ramifications of our feature selections, we performed frequency distribution analyses across the high conflict and low conflict classes of the Teams Dataset. This analysis shows that the high conflict class has a high frequency of negative sentiment polarities compared to the low conflict class and a comparable frequency of positive sentiment polarities compared to the low conflict class (Figure 1).

In the dialogue act distribution, Statement-non-opinion (sd) is the most frequent tag in both classes. The low conflict class has a high frequency of positive communication indicators like Appreciation (ba), Conventional-closing (fc), and Thanking (ft) compared to the high conflict class. The high conflict class contains a high frequency of bad communication indicators like Uninterpretable (%), Hedge (h), Signal-non-understanding (br), and Apology (fa). Interestingly, high conflict classes have a high frequency of all categories of questions compared to low conflict classes (see dialogue act distributions and n-grams in Appendix H).

Looking at the vocabulary distribution, the high conflict class contains more profanity words than the low conflict class, and there is no overlap between the profanity word lists of both classes. Our analysis reveals that there is value in all three types of features (dialogue acts, sentiment polarity, and vocabulary) but that conflict prediction remains a challenging classification problem.

3.4 Synthetic Datasets

To further improve generalization, we augment our training data with synthetic datasets generated using synonym replacement, as proposed by Wei and Zou (2019). Our data augmentation strategies are described below:

1. **SynReplace**: We augment Teams Game1 and Game2 by replacing the words with synonyms drawn from WordNet.
2. **ASISTReplace**: We augment Teams Game1 and Game2 by replacing the words with only the synonyms present in the ASIST dataset. First, we extract the vocabulary of the ASIST dataset. During the replacement operation, we search for synonyms in WordNet and only replace them with the synonyms present in the ASIST dataset's vocabulary.
3. **GitReplace**: Similar to ASISTReplace, we generate our third dataset by replacing the words with only the synonyms present in the GitHub dataset.

Four synthetic dialogues are generated for each dialogue of the Teams dataset after applying random replacement on 10% of the words. Our intuition is that collaborative problem-solving domains such as software engineering may contain a lot of task specific jargon, and even simple synonym replacement techniques greatly facilitate generalization.

In our experiments, the basic synonym replacement did not significantly change the intent and sentiment of the utterances. To show the robustness of dialogue acts and sentiment sequences towards data augmentation, we utilize TextAttack (Morris et al., 2020), a python package for adversarial attack and data augmentation, to generate a Teams Game2 synthetic dataset. Word Swap by BERT-Masked LM transformation was employed to generate synthetic examples from the Teams Game2 dataset. One synthetic example is generated per dialogue of the Game2 dataset. The synthetic dataset contains $\approx 50\%$ more unique words than the original Game2 dataset (Figure 2). The hamming distance was used to calculate the difference between the sequences of the Game2 original and Game2 synthetic datasets. On average, the adversarial synthetic dataset only resulted in a 11% change in DA sequences and a 14% change in sentiment sequences (Appendix J).

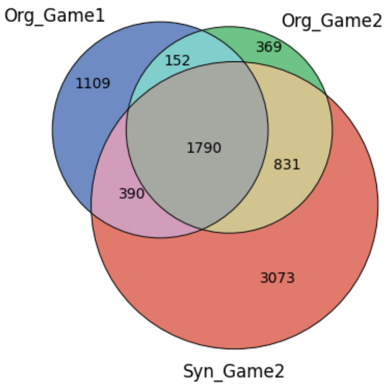


Figure 2: Vocabulary overlap between original Game1, original Game2, and the Game2 adversarially generated dataset

4 Experimental Setup

The Teams corpus contains 124 team dialogues from 62 different teams, playing two different collaborative board games. We use the Teams Game1 dataset with 62 total samples, divided into 32 low and 30 high conflict samples, as our training dataset. The small training dataset ensures that the experiments reflect the generalization performance under the resource scarcity scenario. Our test datasets for evaluating domain adaptation are Teams Game2, GitHub, and ASIST. Obviously, the domain shift is the smallest between the Teams Game 1 and 2 datasets. We use the GitHub and ASIST datasets to check the transferability of MFeEmb under domain shift. The model was not fine-tuned before evaluating the performance on GitHub and ASIST.

We evaluate our proposed MFeEmb under the following three experimental setups:

1. SVM and logistic regression classifiers to distinguish high conflict and low conflict classes.
2. LSTM concatenation ensemble.
3. Few-shot learning approach.

We benchmark MFeEmb against prior work on conflict prediction, other embedding choices, and FsText, a few-shot model proposed by Bailey and Chopra (2018). Experiments were performed using a 300-dimensional version of MFeEmb where the length of all the three embeddings is the same, i.e., 100. We report the mean and standard deviation of F1-Scores after 15 runs. For the SVM and logistic regression classification experiments, to improve the readability, we only report the best-performing classifier results measured by mean F1 score in Section 5; for the results of both classifiers, see Appendix K. ‘*’ denotes that logistic regression was the top performer, and ‘+’ denotes cases where the SVM was the best.

4.1 SVM and Logistic Regression

After Doc2Vec is used to generate the three embeddings for each sample, the embeddings are concatenated to create MFeEmb. We use both SVM and logistic regression to classify the instances and report the results of both classifiers. For DAs and sentiment sequences, we always use the Dynamic Memory model (DM) of Doc2Vec.

4.2 Few-Shot Learning (FsText)

For few-shot learning, we use the method proposed by Bailey and Chopra (2018) and available in the FsText Python module. The training document for the meta-training stage of few-shot learning is represented using a pre-trained word embedding (Word2Vec). In the case of more than one training sample per class, the proposed method works by averaging each class’s vectors to calculate the most effective class representative. Cosine similarity is used to measure the distance between the test sample and each class representative, and the test sample is assigned the label of the class with the highest similarity. We compare the generalizability of FsText (Original) with MFeEmb-based FsText, by replacing Word2Vec embedding with MFeEmb during the meta-training stage.

4.3 Concatenation Ensemble

Due to the small size of the training set, we apply the synonym replacement technique proposed by Wei and Zou (2019) to augment the training data as described in Section 3.4. One hot encoding is used to encode DA, sentiment polarities, and vocabulary to train the model. We train three different Bidirectional LSTM models, one on each of DAs, sentiments, and word-based documents, and merge them to create our MFeEmb based ensemble. Our Bidirectional LSTM models for each feature have an embedding layer, an LSTM layer, one dropout layer, and one deep layer.

4.4 Baseline Models

We compare our proposed MFeEmb’s results with several baseline models that use the same binary classification setup for conflict prediction. First, we show that MFeEmb performs competitively against prior work on conflict prediction (Enayet and Sukthankar, 2021a) using the proposed dialogue act only and sentiment only embeddings. Note that our results are not directly comparable to what was reported in the previous work because we use a reduced training set; thus, we reimplemented the embeddings. We also compare MFeEmb to the commonly used BERT based embedding (Appendix D).

These independent baselines are compared against three implementation options for MFeEmb: 1) MFeEmb with simple binary classifier (SVM or logistic regression), 2) MFeEmb concatenation ensemble learned with LSTMs (Sec. 4.3) trained on the synonym replaced augmented dataset, 3) a variation of few-shot learning method (FsText) (Bailey and Chopra, 2018) in which the Word2Vec embedding is replaced with MFeEmb during the meta-training stage. For training and testing, we concatenate all the utterances of the dialogue into one single document and assign it to one of the classes depending on the conflict score of the team.

5 Results

This section presents results on the generalizability of MFeEmb under different experimental setups.

5.1 Similarity Based Evaluation

First, we quantify the potential generalization of the representation using the similarity measure proposed by Bontonou et al. (2021). The similarity measure is given by:

$$intra(c) = \frac{1}{k(k-1)} \sum_{\substack{i \\ y_i=c}} \sum_{\substack{j\neq i \\ y_j=c}} \cos(f_i, f_j) \quad (1)$$

$$inter(c, \tilde{c}) = \frac{1}{k^2} \sum_{\substack{i \\ y_i=c}} \sum_{\substack{j\neq i \\ y_j=\tilde{c}}} \cos(f_i, f_j) \quad (2)$$

$$similarity = \frac{1}{N} \sum_{c=1}^{N} (intra(c) - \max_{c\neq\tilde{c}}(inter(c, \tilde{c}))) \quad (3)$$

where c is class, N is the number of classes, k is number of examples, f is the embedding, $intra(c)$ is cosine similarity within a class, and $inter(c, \tilde{c})$ is cosine similarity through classes c and $\tilde{c}$. The final similarity score reflects the comparison of the $intra(c)$ and $inter(c, \tilde{c})$. Intuitively it can be seen that the score measures how the representation affects the data clustering within and between classes.

We compare our proposed MFeEmb vs. a standard word embedding learned using the bag of word model of Doc2Vec. Table 1 gives the result of the similarity-based analysis juxtaposed with the classification results. MFeEmb has a better similarity score and high classification performance, compared to word-based embeddings indicating the high generalizability potential of MFeEmb.

Word_Emb		**MFeEmb**	
Teams Game2			
similarity	F1_score	similarity	F1_score
-0.067	0.470*	-0.016	**0.628+**
GitHub			
similarity	F1_score	similarity	F1_score
-0.067	0.463*	-0.017	**0.501+**
ASIST			
similarity	F1_score	similarity	F1_score
-0.067	0.446+	-0.016	**0.458+**

Table 1: Similarity-based generalizability analysis. **Word_Emb**: Distributed Bag of Words document embedding. **MFeEmb**: Multi-Feature Embedding generated using the Dynamic Memory model of Doc2Vec. The similarity score of MFeEmb accurately predicts higher classification accuracy. '*' denotes the logistic regression results, and '+' denotes the SVM results.

5.2 MFeEmb Performance Summary

Figure 3 provides the overall comparison of MFeEmb vs. the benchmark embeddings. In the case where minimal domain adaptation was required (testing classifiers on Teams2 that were trained on Teams1), the simple version of MFeEmb using a SVM classifier is the top performer and outperforms the embeddings used in other prior work on conflict prediction (Enayet and Sukthankar, 2021a). Our most consistent model, MFeEmb with FsText, had a significantly higher F1 score on the high conflict class compared to baseline models (see Table 2). Note that detecting the high conflict examples is more valuable for practical implementations.

For the more complex domain adaptation scenarios (GitHub and ASIST), the best performance was achieved using MFeEmb as a replacement for the Word2Vec embedding during the meta-training phase of FsText on GitHub, and the concatenation ensemble showed significantly better performance on the ASIST dataset. The vanilla MFeEmb generally performed comparably to the concatenation ensemble using LSTMs on out of domain datasets. The latter showed a high standard deviation compared to the former.

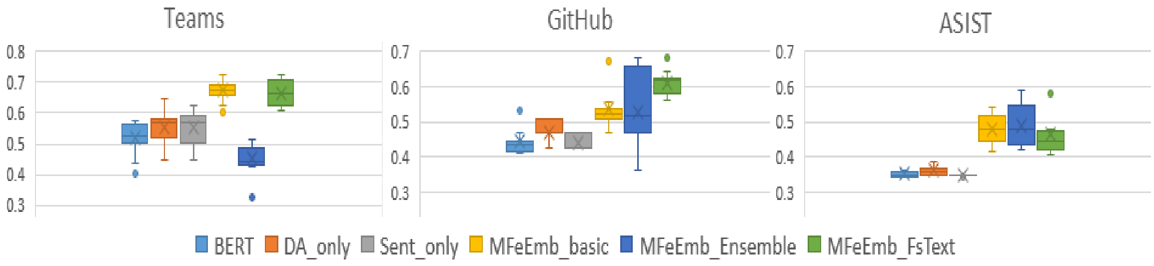


Figure 3: Performance of MFeEmb vs. other embedding choices from prior work.

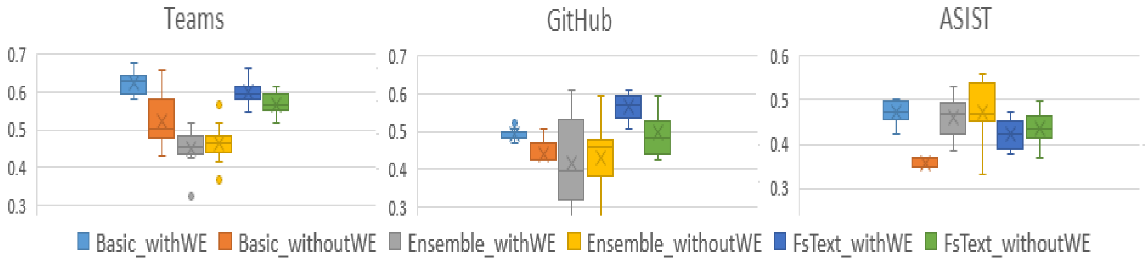


Figure 4: Performance of MFeEmb with and without word embedding (WE).

To analyze the importance of incorporating word embedding in MFeEmb, we compare the performance of all the experimental setups with and without word embedding (WE). For SVM & Logistic regression (Basic) and FsText, we train the model on the Teams Game1 dataset, and for the concatenation ensemble, we train on the synonym replaced dataset. One of our main objectives in incorporating the word embedding in MFeEmb is to maintain the performance on the in-domain dataset, and results show that MFeEmb performed better with word embedding on the in-domain dataset. For most transfer case setups, MFeEmb with word embedding either gave better or comparable mean F1 scores (Figure 4). The following sections present a more in-depth evaluation of each experimental setup.

High Conflict Class Prediction Summary		
Method	**GitHub**	**ASIST**
BERT_SynReplace	0.431	0.347
DA_only_Team1	0.320*	0.311*
Senti_only_Team1	0.207*	0.300*
MFeEmb_FsText_Team1	**0.564**	**0.478**

Table 2: Summary of high conflict class F1 scores. '*' denotes the logistic regression results, and '+' denotes the SVM results.

5.3 SVM and Logistic Regression

Table 3 gives the results for the SVM and logistic regression classifiers. This paper presents a thorough evaluation of the performance of different embedding choices (DM, DBOW). We also evaluate the performance of different data augmentation methods (**SynReplace**, **ASISTReplace**, and **GitReplace**).

Our proposed MFeEmb trained using Doc2Vec and classified using either SVM or logistic regression performed better than the word-embedding baseline. Leveraging synthetic datasets yielded significant performance improvements. In our most challenging resource-scarce scenario, where we trained the model only on the Teams Game1 dataset, incorporating word embedding showed better performance on the Teams Game2 and GitHub datasets, while the model performed better on the ASIST dataset without word embedding (see Figure 4).

5.4 Concatenation Ensemble Model

Table 3 gives the results for the LSTM-based concatenation ensemble model. The model showed a better mean F1-score than the text-based LSTM model. We also trained the LSTM using synthetic datasets generated using GitHub and ASIST vocabularies, which showed better performance, specifically with the GitHub vocabulary dataset. The model performed significantly better on the ASIST dataset compared to the other experimental setups.

SVM & Logistic Regression Results			
Method	**Teams Game2** F1_score (std)	**GitHub** F1_score (std)	**ASIST** F1_score (std)
Baseline Doc2Vec_dbow	0.465 (0.070)*	0.489 (0.080)*	0.425 (0.091)*
MFeEmb_Team1_dbow	0.533 (0.068)*	0.437 (0.025)*	0.347 (0.002)*
MFeEmb_Team1_dm	0.625 (0.0295)+	0.495 (0.012)+	0.473 (0.023)+
MFeEmb_SynReplace	0.558(0.035)+	0.296(0.025)*	0.318 (0.00)*
MFeEmb_GitReplace	**0.676 (0.033)+**	0.409 (0.039)*	0.411 (0.041)*
MFeEmb_ASISTReplace	0.675 (0.041)+	**0.537 (0.060)***	**0.480 (0.042)***
Concatenation Ensemble Results			
Baseline_SynReplace	0.435 (0.048)	0.414 (0.104)	0.397 (0.081)
MFeEmb_SynReplace	0.453 (0.044)	0.429 (0.122)	0.459 (0.044)
MFeEmb_GitReplace	**0.464 (0.044)**	0.468 (0.098)	**0.491 (0.054)**
MFeEmb_ASISTReplace	0.408 (0.075)	**0.516 (0.100)**	0.455 (0.059)
Few-Shot Learning Results			
FsText Baseline	**0.689 (0.0)**	0.330 (0.0)	0.338 (0.0)
MFeEmb_Team1_doc2Vec	0.60 (0.028)	0.583 (0.045)	0.451 (0.025)
MFeEmb_Team1_word2Vec	0.597 (0.041)	0.507 (0.063)	0.437 (0.027)
MFeEmb_SynReplace	0.544 (0.021)	0.568 (0.031)	0.435 (0.037)
MFeEmb_GitReplace	0.684 (0.033)	0.567 (0.041)	0.388 (0.266)
MFeEmb_ASISTReplace	0.664 (0.042)	**0.608 (0.034)**	**0.462 (0.053)**

Table 3: Detailed performance evaluation of MFeEmb. '*' denotes the logistic regression results, and '+' denotes the SVM results.

5.5 Few-Shot Model (FsText)

The FsText baseline showed the best performance on Game2, but the performance degraded considerably on the transfer task (GitHub and ASIST). FsText with the proposed MFeEmb exhibited significantly better performance on the GitHub and ASIST datasets, specifically with ASIST vocabulary's synthetic dataset. FsText with the proposed MFeEmb embedding also gave a comparable performance on the Teams Game2 dataset. This demonstrates that MFeEmb is an excellent representation for meta-pretraining a few-shot model on collaborative multiparty dialogues, even when learned from a small dataset (see Table 3).

Using a synthetic dataset showed a performance improvement in all three experimental setups. Generation of the synthetic dataset using the vocabulary of other collaborative tasks showed comparatively better performance on the transfer task. Even in the in-domain experiments, the Game1 Synthetic dataset, generated using collaborative task vocabulary, showed the best and comparable performance on Game2 in all the experimental setups.

6 Conclusion and Future Work

This paper introduces a multi-feature embedding (MFeEmb) to improve the generalizability of multiparty dialogue models under resource scarcity scenarios. We propose the use of a combination of textual (words), structural (DAs), and semantic (sentiment, DAs) embeddings to reduce the performance loss due to natural distribution shift. Experiments show that the multi-feature embedding performs significantly better than sentence (BERT), dialogue act-only, sentiment-only, and word embeddings. Our results demonstrate that MFeEmb is a superior representation for meta-pretraining a few-shot model that works well across different collaborative problem-solving domains.

Our proposed data augmentation strategy successfully resolved the domain shift problem caused by task-specific vocabulary without perturbing the dialogue act and sentiment features. Experiments with synthetic datasets show that synonym replacement with vocabulary drawn from a collaborative task outperforms generic synonym replacement with WordNet. It improves both the transfer accuracy and the test accuracy on the in-domain test set. Note that we did not fine-tune the models on the target datasets, i.e., GitHub and ASIST, and strictly report the model learned on the Teams dataset. Only the vocabulary of these datasets was used to boost the performance; explicit fine-tuning of the machine learning models could further improve the results.

7 Limitations

This paper only reports results on the generalizability of MFeEmb on conflict prediction tasks; MFeEmb may not perform as well on other communication analysis tasks. However, we believe that modifying the features used in the embedding can address this problem. In future work, we are interested in applying our embedding to new team communication analysis tasks such as identifying emergent leadership.

8 Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. W911NF-20-1-0008 and the National Institute of Mental Health of the National Institutes of Health under Award Number P50MH126231. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, the National Institutes of Health, and the University of Central Florida.

References

Katherine Bailey and Sunny Chopra. 2018. Few-shot text classification with pre-trained word embeddings and a human in the loop. *arXiv preprint arXiv:1804.02063*.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia. Association for Computational Linguistics.

Myriam Bontonou, Louis Béthune, and Vincent Gripon. 2021. Predicting the generalization ability of a few-shot classifier. *Information*, 12(1):29.

Shumin Deng, Ningyu Zhang, Zhanlin Sun, Jiaoyan Chen, and Huajun Chen. 2019. When low resource nlp meets unsupervised language model: Meta-pretraining then meta-learning for few-shot text classification. *arXiv preprint arXivi1908.08788*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Nikita Dvornik, Cordelia Schmid, and Julien Mairal. 2020. Selecting relevant features from a multi-domain representation for few-shot classification. In *European Conference on Computer Vision*, pages 769–786. Springer.

Ayesha Enayet and Gita Sukthankar. 2020. A transfer learning approach for dialogue act classification of GitHub issue comments. *CoRR*, abs/2011.04867.

Ayesha Enayet and Gita Sukthankar. 2021a. Analyzing team performance with embeddings from multiparty dialogues. In *IEEE International Conference on Semantic Computing (ICSC)*, pages 33–39.

Ayesha Enayet and Gita Sukthankar. 2021b. Learning a generalizable model of team conflict from multiparty dialogues. *International Journal of Semantic Computing*, 15(04):441–460.

Lixiao Huang, Jared Freeman, Nancy Cooke, Samantha Dubrow, John "JCR" Colonna-Romano, Matt Wood, Verica Buchanan, Stephen Caufman, and Xiaoyun Yin. 2022. Artificial Social Intelligence for Successful Teams (ASIST) Study 2.

K.A. Jehn. 1995. A multimethod examination of the benefits and determinants of intragroup conflict. *Administrative Science Quarterly*, 40:256–282.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic. Association for Computational Linguistics.

Uliyana Kubasova, Gabriel Murray, and McKenzie Braley. 2019. Analyzing verbal and nonverbal features for predicting group performance. *arXiv preprint arXiv:1907.01369*.

Catherine Lai and Gabriel Murray. 2018. Predicting group satisfaction in meeting discussions. In *Proceedings of the Workshop on Modeling Cognitive Processes from Multimodal Data*, pages 1–8.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China. PMLR.

Diane Litman, Susannah Paletz, Zahra Rahimi, Stefani Allegretti, and Caitlin Rice. 2016. The Teams corpus and entrainment in multi-party spoken dialogues. In

Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1421–1431.

Ellyn Maese, Pablo Diego-Rosell, Les DeBusk-Lane, and Nathan Kress. 2021. Development of emergent leadership measurement: Implications for human-machine teams. In *AAAI Symposium on Computational Theory of Mind for Human-Machine Teams*.

Shannon Marlow, Christina Lacerenza, Jensine Paoletti, C. Shawn Burke, and Eduardo Salas. 2018. Does team communication represent a one-size-fits-all approach?: A meta-analysis of team communication and performance. *Organization Behavior and Human Decision Processes*, 144:145–170.

John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. *arXiv preprint arXiv:2005.05909*.

Gabriel Murray and Catharine Oertel. 2018. Predicting group performance in task-based interaction. In *Proceedings of the ACM International Conference on Multimodal Interaction*, pages 14–20.

Kyungjin Park, Hyunwoo Sohn, Wookhee Min, Bradford Mott, Krista Glazewski, C Hmelo-Silver, and James Lester. 2022. Disruptive talk detection in multi-party dialogue within collaborative learning environments with a regularized user-aware network. In *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Samuel L Pugh, Arjun Rao, Angela EB Stewart, and Sidney K D'Mello. 2022. Do speech-based collaboration analytics generalize across task contexts? In *International Learning Analytics and Knowledge Conference*, pages 208–218.

Zahra Rahimi and Diane Litman. 2020. Entrainment2vec: Embedding entrainment for multi-party dialogues. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8681–8688.

Tonmoy Saikia, Thomas Brox, and Cordelia Schmid. 2020. Optimized generic feature learning for few-shot classification across domains. *arXiv preprint arXiv:2001.07926*.

Eleni Triantafillou, Hugo Larochelle, Richard Zemel, and Vincent Dumoulin. 2021. Learning a universal template for few-shot dataset generalization. In *International Conference on Machine Learning*, pages 10424–10433. PMLR.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3).

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, Hong Kong, China. Association for Computational Linguistics.

A Dialogue Act Classification

We use a dialogue act classifier (USE-DAC) to map dialogues to the sequence of DAs, where each DA in a sequence corresponds to the utterance of the dialogue. Utterances are tagged according to the SwDA-DAMSL tagset[1] which contains 42 tags, and one sequence is generated per dialogue. Our dialogue act classifier uses the Universal Sentence Encoder (USE) module available at TensorFlow Hub[2]. After extensive experiments, we identified that USE with three dense layers performs best on transfer tasks. We selected the USE Transformer-based Architecture model with three dense layers and a softmax activation function. We fine-tune USE on the SwDA dataset and use the classifier to tag the utterances of the test and training datasets. We selected the USE transformer-based model because it is itself trained on dialogue and discussion forum datasets. The test accuracy of the classification model is 72%.

B Doc2Vec

Doc2Vec (Le and Mikolov, 2014) is an unsupervised method to learn paragraph vectors from text of arbitrary size. We represent each dialogue as 1) sequence of utterances, 2) sequence of DAs, and 3) sentiment polarities. We pass these sequences through Doc2Vec to generate representations. We use the Doc2Vec implementation from the python Gensim library with an epoch size of 5, negative sampling 5, window size 5, and alpha 0.065.

C SVM & Logistic Regression

We use the classifier implementations from the scikit-learn library. The SVM was trained using the RBF kernel function and the default parameters. The parameters for logistic regression were: Cs=10, class_weight=None, cv=10, dual=False, fit_intercept=True, intercept_scaling=1.0, max_iter=1000, multi_class='ovr', n_jobs=None, penalty='l2', random_state=5434, refit=True, scoring='accuracy', solver='lbfgs', tol=0.001, verbose=False. Table 6 shows the full results of both the SVM and Logistic Regression classifiers.

D BERT Baseline

We use the bert_en_uncased_L12_H768_A12 model available at TensorFlow Hub[3] to develop our baseline classifier. The model contains one dense layer, one dropout layer, a sigmoid activation function, Adam optimizer. Due to the small size of the Game1 dataset we train the model on the synonym replaced Game1 dataset. The total number of parameters in the model is: 10,948,301.

E FsText

The original FsText works by using the pre-trained word2Vec embedding model word2vec-google-news-300 available through the gensim.downloader module. For MFeEmb we generate the embedding using Doc2Vec with the same parameters mentioned in Appendix B. The second phase uses a cosine similarity-based classification model that does not involve machine learning.

F Concatenation Ensemble

Our Bidirectional LSTM models for each feature has an embedding layer, an LSTM layer, one dropout layer, and one deep layer. The LSTM uses a sigmoid activation function and is trained using the Adam optimizer with a learning rate=0.01. The output shape of each individual model is (None, 100). The total number of parameters of the concatenation ensemble is: 2,365,081.

G GitHub Dataset Conflict Scoring

For GitHub, process conflict was scored according to issue resolution using the following heuristics to determine if conflicts occurred:

1. Unsuccessful resolution of the issue.
2. Unanswered questions in the discussion.
3. Lack of understanding about the issue from one or more members.
4. Lack of understanding or disagreement between the team members.
5. Disagreement between the members about the proposed solution.

H Dialogue Act Frequency Distribution Analysis

Figure 6 shows the frequency distribution for dialogue acts in the low conflict and high conflict classes of the Teams dataset. We divided the

---

[1] `https://web.stanford.edu/~jurafsky/ws97/manual.august1.html`

[2] `https://tfhub.dev/google/universal-sentence-encoder-large/2`

[3] `https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4`

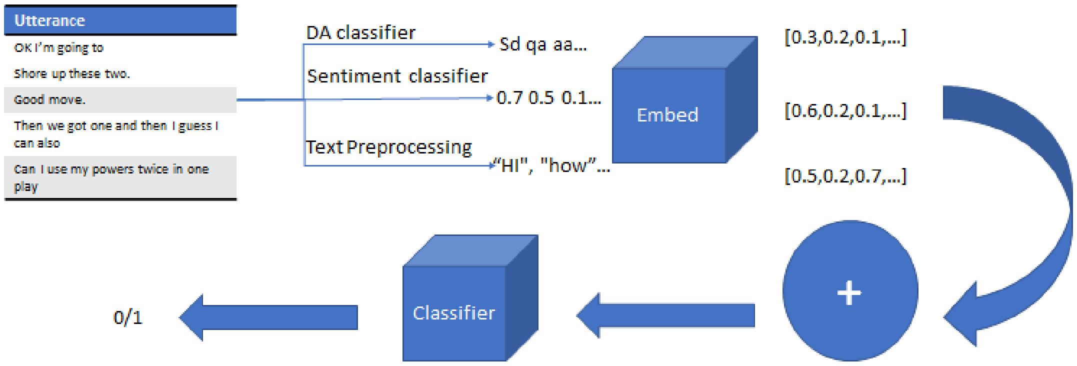


Figure 5: Utterances are classified using the dialogue act classifier to produce a sequence of DAs and the sentiment classifier to produce a time series of sentiment polarities. Along with the text data, these sequences are used to create MFeEmb using the Dynamic Memory model of Doc2Vec. The few shot learning and data augmentation options are not shown in the figure.

dialogue acts into good communication indicators, bad communication indicators and questions. Figure 7 shows the complete dialogue act frequency distribution. Table 4 shows the n-gram frequency distribution of dialogue acts across all datasets. Figure 8 visualizes the separation between low conflict and high conflict classes using both the MFeEmb and word embedding. To see if profanities were a reliable indication of conflict, we also examined profanity vocabulary differences. The most frequent words in the high conflict dialogues that are in profanity list are: ['hell', 'kill','suck','sucking','shit','strip','stroke', 'rectum','xxx','dick','screwed','retard', 'ovary','piss','lube', 'junkie'].

The most frequent words in the low conflict dialogues that are in the profanity list are: ['booty', 'pot','carpet', 'rum', 'breasts', 'pedophile', 'urine', 'thug', 'screw', 'jerk', 'weed', 'screwing', 'shower', 'stupid'].

I Synthetic Datasets

Synthetic datasets were generated using: https://github.com/jasonwei20/eda_nlp.

J Results on Adversarially Generated Dataset

This section presents results on the adversarially generated dataset (Synthetic Game 2) created using TextAttack[4]. Word Swap by BERT-Masked LM transformation was employed to generate synthetic examples from the Teams Game2 dataset. One synthetic example is generated per dialogue of the Game2 dataset. The length of the synthetic Game2 dataset vocabulary is 6084, and the length of the original Game1 dataset vocabulary is 3441. The number of words in the synthetic dataset that are not in the original Game1 is 3904.

Figure 2 shows a high overlap between original Game1 and original Game2 compared to synthetic Game2 and original Game1, but this does not affect the performance of MFeEmb (Basic), and MFeEmb. (Basic) gave a better performance on the synthetic dataset. On the other hand, the performance of the BERT baseline decreased on the synthetic Game2 test set, with a high standard deviation in mean F1 scores.

K SVM & Logistic Regression Results

Table 6, 7, 8 provides the detailed results of both the SVM and logistic regression classifiers under different experimental settings.

---

[4] https://github.com/QData/TextAttack

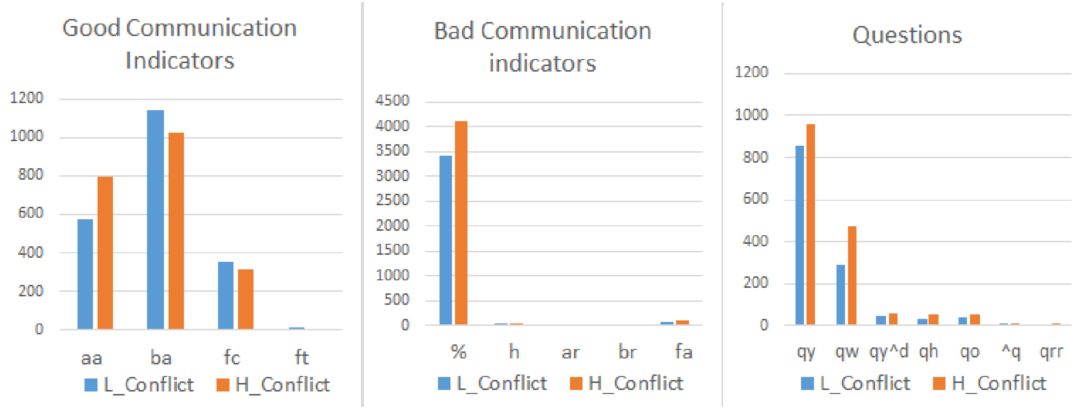


Figure 6: Dialogue act frequency distribution in high and low conflict classes for the Teams dataset. Dialogue acts were divided into good and bad communication indicators.

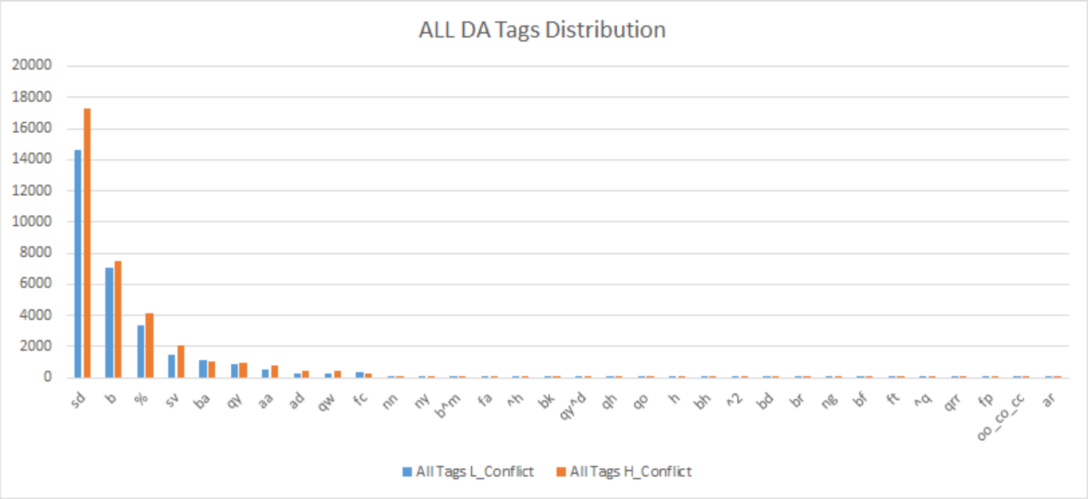


Figure 7: Complete dialogue act frequency distribution for high and low conflict classes in the Teams dataset.

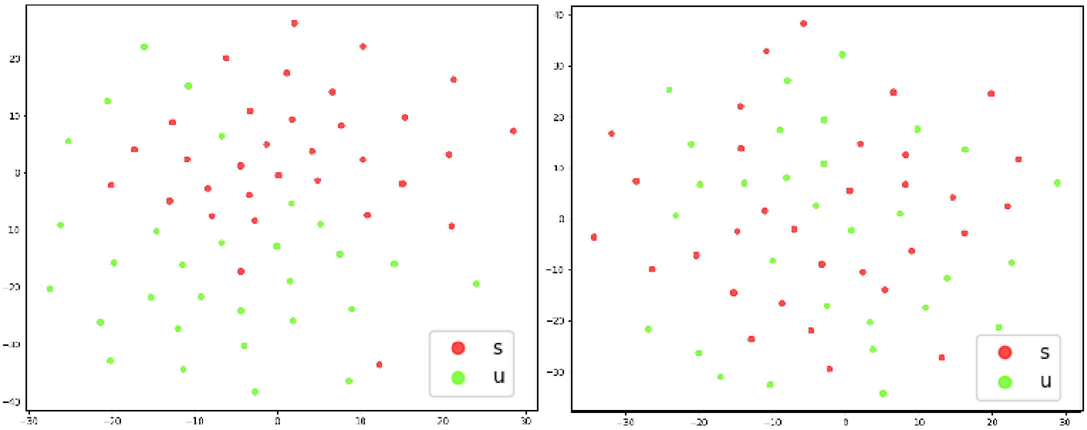


Figure 8: Comparison of the MFeEmb and word embedding distribution on the 2D plane. Multi-feature embedding showed better clustering, with most instances of one of the classes occupying the lower left and the other occupying the upper right. On the other hand, word embeddings are very intermixed. s: low conflict (successful dialogue), u: high conflict (unsuccessful dialogue).

Dataset	**Unigrams**	**Bigrams**	**Trigrams**	**4grams**	**5grams**
Teams	(sd),(b),(%)	(sd,sd),(sd,b),(b,sd)	(sd,sd,sd), (sd,sd,b), (sd,b,sd)	(sd,sd,sd, sd), (sd,sd,sd,sd), (sd,sd,sd,b)	(sd,sd,sd,sd, sd), (sd,sd,sd,sd,b), (sd, sd,sd,b,sd)
GitHub	(sd),(sv),(ad)	(sd,sd),(sd,sv),(sv,sd)	(sd,sd,sd), (sv,sd,sd), (sd,sd,ad)	(sd,sd,sd,sd), (sd,sd,sd,ad), (sv,sd,sd,sd)	(sd,sd,sd,sd,sd), (sd,sd,sd,sd, ad), (sd,sv,sd,sd,sd)
ASIST	(sd),(qy),(sv)	(sd,sd),(sd,qy),(qy,sd)	(sd,sd,sd), (sd,qy,sd), (qy,sd, sd)	(sd,sd,sd,sd), (sd,qy,sd,sd), (sd,sd,qy,sd)	(sd,sd,sd,sd,sd), (sd,sd,sd,sd, qy), (sd,qy,sd,sd,sd),

Table 4: N-gram frequency distribution: top three most frequent unigrams, bigrams, trigrams, 4grams, 5grams of all the datasets. Sequences of sd (statement-nonopinion) are common across all datasets.

Game2 Synthetic Dataset Results				
Train model	**Teams Game1** F1_score (std)	**SynReplace** F1_score (std)	**GitReplace** F1_score (std)	**ASISTReplace** F1_score (std)
MFeEmb (Basic)	0.654 (0.033)+	0.443 (0.046)*	**0.617 (0.035)+**	**0.624 (0.055)+**
BERT	-	**0.490 (0.061)**	0.422 (0.037)	0.495 (0.044)

Table 5: MFeEmb results on the Game2 synthetic dataset generated using TextAttack

Both SVM & Logistic Regression Results			
Method	**Teams Game2** F1_score (std)	**GitHub** F1_score (std)	**ASIST** F1_score (std)
Baseline Doc2Vec_dbow	0.465 (0.070)* 0.369 (0.0)+	0.489 (0.080)* 0.425 (0.0)+	0.425 (0.091)* 0.348 (0.0)+
MFeEmb_Team1_dbow	0.533 (0.068)* 0.369 (0.0)+	0.437 (0.025)* 0.425 (0.0)+	0.347 (0.002)* 0.348 (0.0)+
MFeEmb_Team1_dm	0.625 (0.0295)+ 0.569 (0.045)*	0.495 (0.012)+ 0.428 (0.0)*	0.473 (0.023)+ 0.393 (0.032)*
MFeEmb_SynReplace	0.558(0.035)+ 0.542 (0.045)*	0.296(0.025)* 0.248 (0.0)+	0.318 (0.00)* 0.318 (0.00)+
MFeEmb_GitReplace	**0.676 (0.033)+** 0.593 (0.056)*	0.409 (0.039)* 0.248 (0.0)+	0.411 (0.041)* 0.318 (0.00)+
MFeEmb_ASISTReplace	0.675 (0.041)+ 0.643 (0.044)*	**0.537 (0.060)*** 0.248 (0.0)+	**0.480 (0.042)*** 0.318 (0.00)+

Table 6: Results for both the SVM and logistic regression classifiers side by side.

Word_Emb		**MFeEmb**	
Teams Game2			
similarity	F1_score	similarity	F1_score
-0.067	0.470* 0.369+	-0.016	**0.628+** 0.561*
GitHub			
similarity	F1_score	similarity	F1_score
-0.067	0.463* 0.425+	-0.017	**0.501+** 0.439*
ASIST			
similarity	F1_score	similarity	F1_score
-0.067	0.446+ 0.348*	-0.016	**0.458+** 0.394*

Table 7: Similarity-based generalizability analysis. '*' denotes the logistic regression results, and '+' denotes the SVM results.

High Conflict Class Prediction Summary		
Method	**GitHub**	**ASIST**
BERT_SynReplace	0.431	0.347
DA_only_Team1	0.320* 0.250+	0.311* 0.216+
Senti_only_Team1	0.207* 0.090+	0.300* 0.036+
MFeEmb_FsText_Team1	**0.564**	**0.478**

Table 8: Summary of high conflict class F1_scores.'*' denotes the logistic regression results, and '+' denotes the SVM results.

MetaQA: Combining Expert Agents for Multi-Skill Question Answering

Haritz Puerto[1], **Gözde Gül Şahin**[2], **Iryna Gurevych**[1]
[1]Ubiquitous Knowledge Processing Lab (UKP Lab)
Department of Computer Science and Hessian Center for AI (hessian.AI)
Technical University of Darmstadt, Germany
[2] Department of Computer Science, Koç University, KUIS AI Lab
Istanbul, Türkiye
https://www.ukp.tu-darmstadt.de & https://ai.ku.edu.tr

Abstract

The recent explosion of question-answering (QA) datasets and models has increased the interest in the generalization of models across multiple domains and formats by either training on multiple datasets or combining multiple models. Despite the promising results of multi-dataset models, some domains or QA formats may require specific architectures, and thus the adaptability of these models might be limited. In addition, current approaches for combining models disregard cues such as question-answer compatibility. In this work, we propose to combine expert agents with a novel, flexible, and training-efficient architecture that considers questions, answer predictions, and answer-prediction confidence scores to select the best answer among a list of answer predictions. Through quantitative and qualitative experiments, we show that our model i) creates a collaboration between agents that outperforms previous multi-agent and multi-dataset approaches, ii) is highly data-efficient to train, and iii) can be adapted to any QA format. We release our code and a dataset of answer predictions from expert agents for 16 QA datasets to foster future research of multi-agent systems[1].

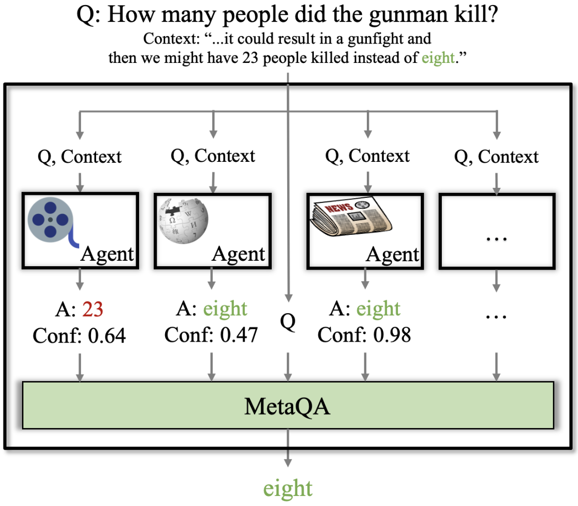


Figure 1: Given a question, each expert agent provides a prediction with a confidence score and MetaQA selects the best answer. Correct answers in green. Wrong answers in red.

1 Introduction

The large number of question answering (QA) datasets released in the past years has been accompanied by models specialized in them (Rogers et al., 2021; Dzendzik et al., 2021). These datasets and models differ by the domain (e.g., biomedical and Wikipedia), required skills (e.g., numerical and multi-hop), and format (e.g., extractive and multiple-choice). This variety of tasks and overspecialization of the corresponding models have led the community towards developing simple unified models that can generalize across domains and formats through unifying dataset formats (Khashabi et al., 2020), creating models trained on multiple datasets (Fisch et al., 2019; Talmor and Berant, 2019; Khashabi et al., 2020), and designing ensemble methods for QA agents (Geigle et al., 2021). All these research lines have a potential impact on end-user applications because generalization can help create robust systems and ease the implementation of QA models. For example, some chatbots are composed of skill systems, where each skill is a model trained on a specific domain (Miller et al., 2017; Burtsev et al., 2018). More abstractly, these research lines also share a central research question: *how to combine QA skills*.

We argue that a *one-size-fits-all* architecture may encounter some limitations in combining QA skills. For instance, Raffel et al. (2020) have observed that a single model trained on multiple tasks may underperform the same architecture trained on a single task. An alternative approach is to combine multiple expert agents. Geigle et al. (2021) propose a model that given a question and a list of datasets, selects the dataset from which the question comes.

---

[1] https://github.com/UKPLab/MetaQA

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3566–3580
May 2-6, 2023 ©2023 Association for Computational Linguistics

This can be used to identify agents trained on a specific type of questions. However, despite achieving a classification accuracy greater than 90%, this approach underestimates high-performing models on out-of-domain questions.

To address the limitations of previous approaches, we propose a novel model, MetaQA, to combine heterogeneous expert agents (i.e., different architectures, formats, and tasks). It takes a question, and a list of *candidate answers* with *confidence scores* as input and selects the best answer (Figure 1). We modify the embedding mechanism of the Transformer encoder (Vaswani et al., 2017) to embed the confidence score of each candidate answer. In addition, we use a multi-task training objective that makes the model learn two complementary tasks: *selecting the best candidate answer* and *identifying agents trained on the domain of the input question*.

Our approach learns to match questions with answers, an immensely easier task than the end-to-end QA of multi-dataset models. This makes MetaQA remarkably data efficient as it only uses 16% of the training data of multi-dataset models.

We compile a list of 16 QA datasets that encompass different domains, formats, and reasoning skills to conduct experiments. Through quantitative experiments, we show that our MetaQA i) establishes a successful collaboration between agents, ii) outperforms multi-agent and multi-dataset models, iii) excels in minority domains, and iv) is highly efficient to train. Our contributions are:

- A new approach for multi-skill QA that establishes a collaboration between agents.
- A model called MetaQA that utilizes question, answer, and confidence scores to select the best candidate answer for a given question.
- Extensive analyses showing the successful collaboration between agents and the training efficiency of our approach.
- A dataset of (*QA Agents*, *Questions*, and *answer predictions*) triples that cover different QA formats, domains, and skills to foster future developments of multi-agent models.

2 Related Work

Currently, there are two approaches for multi-skill QA: multi-agent and multi-dataset models.

Multi-agent models consists of combining multiple expert agents. A well-known method is the Mixture of Experts. It requires training a set of models and combining their outputs with a gating mechanism (Jacobs et al., 1991). However, this approach would require jointly training multiple agents, which can be extremely expensive, and sharing a common output space to combine the agents. These limitations make it unfeasible to implement in our setup, where many heterogeneous agents are combined (i.e., agents with different architectures, target tasks, and output formats such as integers for multiple-choice or answer spans for span extraction). Inspired by topic classification, Geigle et al. (2021) proposed mapping questions to QA datasets (topics) to identify agents trained on that type of questions. Although related to us, their work does not attempt to achieve any agent collaboration. Moreover, because of their *topic-classification* approach, agents that are effective in out-of-domain questions are underestimated. Lastly, Friedman et al. (2021) average the weights of adapters (Houlsby et al., 2019) trained on single datasets to obtain a multi-dataset model. However, their architecture is limited to span extraction.

Multi-dataset models consist of training a model on various datasets to generalize it to multiple domains. Talmor and Berant (2019) conduct extensive analyses of the generalization of QA models. However, they only experiment on extractive tasks and, due to their model architecture (BERT for span extraction), it is not possible to extend it to other tasks such as abstractive or visual QA. Fisch et al. (2019) created a competition on QA generalization using 18 datasets. These datasets are from very different domains, such as Wikipedia and biomedicine, among others. However, they also focus only on extractive datasets. Lastly, Khashabi et al. (2020) shows that the different QA formats can complement each other to achieve a better generalization. They use an encoder-decoder architecture and transform the questions into a common format. However, we argue that their approach is limited because some questions may require a specific skill that must be modeled in a particular manner (e.g., numerical reasoning), and this is not possible with their simple encoder-decoder.

3 Model

We propose a new model, shown in Figure 2, to combine k QA agents m. Each agent m_i is trained

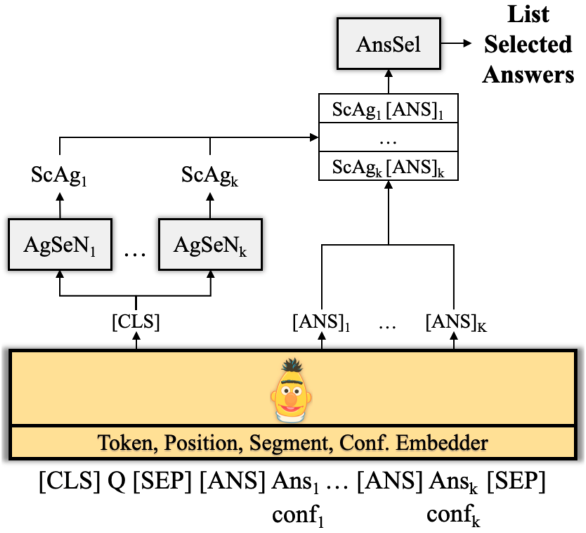


Figure 2: MetaQA architecture. The Agent Selection Network, AgSeN, assigns scores to each agent based on the similarity between the agent's training set domains and the question domain. Answer Selection, AnsSel, selects the correct answers. conf_k is the confidence score from the agent k for answer k.

on domain dom_i and predicts an answer Ans_i. Without loss of generalizability, we assume that each agent is trained on a different domain and each question belongs to one of these domains. We define two complementary tasks: i) *a priori* agent selection (Agent Selection Networks, AgSeN, in Figure 2) and ii) answer selection (AnsSel network in Figure 2). The division of the problem into these two learnable tasks is vital to ensure that MetaQA considers out-of-domain agents, which can also give correct answers. More formally, MetaQA computes the probability of returning an answer a as follows:

$$p(a|q) = \arg\max_i p_{\theta_1}(a_i|m_i, q) p_{\theta_2}(m_i|q) \quad (1)$$

where $p_{\theta_2}(m_i|q)$ is the *a priori* probability of selecting agent m_i and $p_{\theta_1}(a_i|m_i, q)$ is the probability of selecting the answer from agent m_i for the question q. In other words, θ_2 is the agent selection network, and θ_1 is the answer selection network.

To achieve this, the backbone of our architecture relies on an encoder Transformer (Vaswani et al., 2017) whose input is the concatenation of the question with the candidate answers from each agent. Each answer is separated by a new token `[ANS]` that informs the model of the beginning of a new answer candidate.

We devise a new embedding for the Transformer encoder to include the confidence score of the predictions of each agent (Figure 3). While the original encoder uses the token t_i, position p_i, and segment s_i embeddings, we add an agent confidence embedding c_i to these three.

$$x_i = t_i + p_i + s_i + c_i \quad (2)$$

The new c_i is obtained with a feed-forward network f that takes an answer confidence conf_i and creates an embedding c_i.

$$c_i = \begin{cases} f(\text{conf}_j), & \text{if } i \in Idx(\text{[ANS] Ans}_j) \\ f(0), & \text{otherwise} \end{cases} \quad (3)$$

where Idx is a function that given a list of tokens returns their indexes in the encoder input.

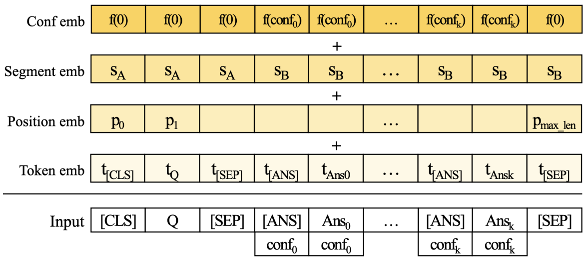


Figure 3: Description of our novel embedding system including confidence scores from the agents.

We leverage two types of embeddings from the output of the encoder. The first one is the embedding of the `[CLS]` token. This embedding captures information about the domain of the input question. It is used as the input to k independent feed-forward networks called *Agent Selection Network* (AgSeN) to identify the agent with the highest likelihood of giving a correct answer. This prediction is based on the similarity between the domain of the question and the domain of the training set of the agents. More specifically, it tries to identify the agent trained on the dataset from which the question comes in a similar way as TWEAC (Geigle et al., 2021). The second type of embedding used is the embedding of the `[ANS]` tokens, which contain the cues needed to identify the correct answers to the input question. These `[ANS]` embeddings are concatenated with the score of each corresponding agent $ScAg_i$ and input into a final feed-forward network, called *Answer Selection* (AnsSel), that selects the correct answers according to the score of their agents and the semantics of the candidate answers.

3.1 Training

As previously mentioned, our model learns two complementary tasks: i) *a priori* agent selection and ii) answer selection. Thus, to learn these two tasks, we define the following loss function:

$$\ell = \frac{\alpha_1}{k} \sum_{i=0}^{k} \ell_{AgSeN_i} + \alpha_2 \ell_{AnsSel} \quad (4)$$

$$\ell_{AnsSel} = \frac{1}{k} \sum_{i=0}^{k} CE(\hat{Ans_i}, y_i) \quad (5)$$

where ℓ_{AgSeN_i} is the loss of one AgSeN network and ℓ_{AnsSel} the loss of the AnsSel network. ℓ_{AnsSel} is the average of the cross-entropy loss CE of each answer prediction $\hat{Ans_i} = \{0, 1\}$. Lastly, AgSeN networks use the Binary Cross Entropy.

We obtain the labels of AnsSel, y_i, by comparing the string prediction of each agent with the correct answer. If the F1 score is higher than a threshold, θ, we consider the prediction as correct. As for AgSeN$_i$, its training label is 1 when the input question is from the training set of the i^{th} agent.

4 Experimental Setup

4.1 Datasets

We have collected a series of QA datasets covering different formats, domains, and reasoning skills. In particular, we use four formats: extractive, multiple-choice, abstractive, and multimodal.

For extractive, we use the MRQA 2019 shared task collection (Fisch et al., 2019), QAMR (Michael et al., 2018), and DuoRC (Saha et al., 2018). We include these two additional datasets to add more diversity. In detail, QAMR requires predicate-argument understanding, a skill that agents should have to solve most QA datasets. As for DuoRC, it is the only dataset in our collection on the film domain, and this allows us to study transfer learning from other domains. The multiple-choice datasets require boolean reasoning, commonsense, and passage summarization skills. Lastly, we include abstractive QA following (Khashabi et al., 2020) and a multimodal dataset to show that our approach can solve any type of question while multi-dataset models are limited to certain formats.

Most of these datasets do not have the labels of the test set publicly available, except for RACE and NarrativeQA. Since we need to do hyperparameter tuning and hypothesis testing to compare models, we divide the public dev set into an in-house dev set and test sets following (Joshi et al., 2020). Then, we conduct hyperparameter tuning on the dev set and hypothesis testing on the test set. A summary of the datasets is available in Appendix A.1.

4.2 Expert Agents

To guarantee a fair comparison with MultiQA, we have trained all the agents for extractive datasets using the same architecture as MultiQA, span-BERT, a BERT model pretrained for span extraction tasks that clearly outperforms BERT on the MRQA 2019 shared task (Joshi et al., 2020). More details on the implementation are provided in Appendix A.3. For the remaining datasets, we use agents that are publicly available on HuggingFace or Github with a performance close to the current state of the art. A summary of them is provided in Appendix A.2.

4.3 Baselines

We compare our approach with three types of models: i) multi-agent systems, ii) multi-dataset models, and iii) expert agents. The first family is represented by our main baseline, TWEAC, a model that maps questions to topics (or types of questions) to identify agents trained on that type of data (Geigle et al., 2021) and the simple max-voting ensemble. The second family of models is composed of MultiQA (Talmor and Berant, 2019) and UnifiedQA (Khashabi et al., 2020). MultiQA is a transformer encoder with a span-extraction layer trained on multiple extractive QA datasets. Because of this span-extraction layer, it can only solve extractive QA tasks. UnifiedQA, on the other hand, can solve any QA task that can be converted into text-to-text (i.e., extractive, abstractive, and multiple-choice) thanks to its architecture, an encoder-decoder transformer. Lastly, we include the expert agents to analyze whether MetaQA closes the gap to them compared to the baselines.

4.4 Evaluation

Since MetaQA may select more than one answer, we select the answer with the highest confidence score by MetaQA as the decision of the model to evaluate it. We evaluate our model and the baselines using the official metrics of each dataset, i.e., macro-average F1 for extractive, accuracy for multiple-choice, and rouge-L for abstractive. In the particular case of DROP, the official metric is macro-average F1, and thus, we also use it. The

Dataset	MetaQA	TWEAC	Exp. Agent	UnifiedQA	MultiQA	Voting
SQuAD	91.98±0.11†	89.09±0.36	92.92	90.81	**93.14±0.18**	90.73
NewsQA	71.71±0.21†	66.86±0.75	**73.68**	65.57	73.59±0.60	66.60
HotpotQA	79.27±0.15†	74.96±0.59	80.60	77.92	**81.68±0.22**	71.71
SearchQA	**81.98±0.25**†‡	80.41±0.22	81.04	81.61	80.45±1.82	68.87
TriviaQA-web	**80.63±0.26**†‡	76.55±0.15	79.34	72.34	77.76±4.15	75.73
NQ	81.20±0.18†	78.06±0.37	81.97	75.58	**82.57±0.30**	72.25
DuoRC	**51.24±0.20**†‡	44.28±0.23	43.77	34.65	46.99±0.15	50.94
QAMR	83.78±0.14†	78.77±0.48	84.00	82.70	**84.62±0.14**	73.07
BoolQ	73.14±0.23†	72.20±0.03	72.17	**81.34**	n.a.	73.88
CSQA	**78.66±0.19**†	77.18±0.18	78.56	58.43	n.a.	68.41
HellaSWAG	73.19±1.01	77.12±0.30	**77.14**	36.01	n.a.	69.33
RACE	84.71±0.05†	83.02±0.27	**84.78**	69.65	n.a.	67.30
SIQA	74.17±0.64	75.39±0.05	**75.44**	61.62	n.a.	70.01
DROP	73.04±1.98†	69.12±0.36	**74.61**	42.45	n.a.	26.18
NarrativeQA	**67.19±0.00**	**67.19±0.00**	**67.19**	57.82	n.a.	**67.19**
HybridQA	**50.94±0.00**	**50.94±0.00**	**50.94**	n.a	n.a	**50.94**

Table 1: MetaQA (ours) and the baselines on the test set of each dataset. Best results in bold. † represents that MetaQA is statistically significant better than TWEAC. ‡ represents that MetaQA is statistically significant better than MultiQA. n.a means that the system cannot model the dataset.

reported results are the means and standard deviations of the models trained with five different seeds except for UnifiedQA, which would be too expensive to compute. We use a two-tailed T-Test to compare the models with a p-value of 0.05.

5 Results and Discussions

In this section, we answer the questions: i) is MetaQA able to combine multiple agents without undermining the performance of each one (§5.1), ii) is it robust on out-of-domain scenarios? (§5.2), iii) how does agent collaboration work? (§5.3), iv) how data-efficient is MetaQA? (§5.4), and v) what is the effect of each module of MetaQA? (§5.5).

5.1 Comparison with the Baselines

5.1.1 TWEAC

MetaQA outperforms TWEAC in all datasets except HellaSWAG and SIQA, as shown in Table 1. On average, MetaQA achieves an average performance boost of 2.23 with respect to TWEAC, and more importantly, the performance boost is greater than 4 points on HotpotQA, DuoRC, NewsQA, QAMR, and TriviaQA. Particularly, there is an astonishing 6.8 points performance boost on DuoRC.

The reason for these results is that TWEAC only aims to identify the agent trained on the domain of the question while we retrieve the best answer prediction, even if it comes from out-of-domain models. For instance, in DuoRC, MetaQA selects the in-domain agent only for 43% of its questions, i.e., most of the questions are assigned to agents that are not trained on DuoRC. In this way, MetaQA establishes a collaboration between agents.

We also observe that the gap between MetaQA and TWEAC is more significant on extractive QA than on multiple-choice. This is expected due to our selection of multiple-choice datasets. The substantial differences in the format of these datasets limit the potential agent collaboration. For instance, BoolQ is the only boolean dataset, and therefore, it can only be used to solve boolean questions, which do not appear in the other multiple-choice datasets. Also, SIQA, a commonsense reasoning dataset, uses a short context passage while CSQA (commonsense too) does not have any context, and hence, an agent trained for CSQA cannot be used successfully on SIQA. These characteristics of the setup make the upper-bound performance of MetaQA to be the same as the expert agents. Yet, even with these limitations, MetaQA outperforms TWEAC in three of the five datasets. Also, the expert agents only significantly outperform MetaQA on 2/5 datasets. Lastly, the performance in NarrativeQA and HybridQA is the same because there is only one agent per dataset.

5.1.2 UnifiedQA

MetaQA outperforms UnifiedQA by striking 10.49. In the case of extractive QA, the gap is 5.08, while in multiple-choice is 15.36. We attribute this to the limitations of UnifiedQA's architecture. For example, the performance in DROP is clearly far from our MetaQA. The reason for this is that while the expert agent used by MetaQA is designed for numerical reasoning, UnifiedQA does not have any mechanism to achieve this, and since it is designed as a general model for text-to-text generation, it cannot be augmented with special reasoning modules. The same phenomenon occurs in the multiple-choice datasets and in some minority domains in extractive QA (i.e., NewsQA and DuoRC). The only exception is in BoolQ, where UnifiedQA achieves the best results. However, this is because T5 (Raffel et al., 2020), on which UnifiedQA is trained, is already one of the SOTA models, while the agent we use has lower performance and was the only publicly available model in HuggingFace's Model Hub at the time of experimentation.

5.1.3 MultiQA

MetaQA outperforms MultiQA by a small margin of 0.12, despite being much more flexible (i.e., compatible with any QA format instead of only extractive QA (§4.3). Moreover, our model was trained on only 13% of its training set, as later discussed in §5.4. Furthermore, we observe that MultiQA mostly outperforms expert agents on Wikipedia-based datasets (i.e., SQuAD, HotpotQA, NQ, and QAMR). This might suggest that MultiQA is overfitted to Wikipedia due to its training on multiple datasets using Wikipedia paragraphs[2] and that would explain why it struggles with other minority domains. On the other hand, MetaQA excels in minority domains where it achieves striking 4.25 points performance boost on DuoRC, 2.87 on TriviaQA-web, 1.53 on SearchQA, and overall outperforms MultiQA by an average of 2.88. These results show the superior ability of MetaQA to avoid overfitting to a specific domain.

5.1.4 Max-Voting

Lastly, MetaQA also outperforms max-voting by an average of 8.35. In the case of easy datasets such as SQuAD, the performance is similar because all expert agents excel in this dataset, so any approach to combine the agents would yield similar results. More interestingly, the performance of Max-Voting is clearly far from MetaQA in DROP. We attribute this to the low performance of the extractive agents on this dataset and their similar wrong answers.

5.2 Leave-One-Out Ablation

In this experiment, we analyze whether the combination of expert agents can successfully solve an out-of-domain (OOD) dataset. We conduct a leave-one-out ablation test in both MetaQA and the baselines. In the case of MetaQA, we remove the expert agent of the target dataset, retrain MetaQA again without this dataset, and evaluate it on the target dataset. Similarly, we retrain TWEAC, UnifiedQA, and MultiQA without the target dataset and evaluate the model on the target dataset. Lastly, we also use the Max-Voting baseline without the agent trained on the target dataset. We trained MetaQA five times with different random seeds for each target dataset and report their average results. However, we could not do this for the other models due to their much higher computation costs.

Table 2 shows that OOD MetaQA outperforms OOD TWEAC in all datasets by an average of 6.31. The larger gap in OOD than in in-domain scenarios (Table 1) supports our hypothesis: the topic-classification approach of TWEAC disregards high-performing models in OOD, and our solution of establishing a collaboration between the agents is able to combine skills.

OOD MetaQA also outperforms OOD UnifiedQA by a striking average of 8.13 points. In addition, in four datasets (TriviaQA-web, DuoRC, CommonsSenseQA, and HellaSWAG), the ablated MetaQA even outperforms the full UnifiedQA trained on those datasets. This further supports our approach of combining multiple agents, instead of datasets, in scenarios with a wide variety of domains and formats, where flexibility is key.

In the particular case of MultiQA, as discussed in §5.1.3, half of its training sets are based on Wikipedia paragraphs. Therefore, removing a Wikipedia-based dataset such as HotpotQA does not remove Wikipedia contents from its training set[3]. As a consequence, this compromises the OOD setup. However, even under this pseudo-OOD

---

[2] MultiQA is trained on question and contexts (Wikipedia paragraphs). However, MetaQA does not have access to these paragraphs as shown in Figure 2.

[3] This is not the case for MetaQA because our input is only the questions, answer predictions, and confidence scores, not the Wikipedia paragraphs.

Dataset	NewsQA	HotpotQA	SearchQA	TriviaQA	NQ	DuoRC	QAMR	CSQA	HellaSWAG	SIQA	DROP	Δ
MetaQA	71.46	79.37	81.87	80.65	81.08	51.01	83.87	78.40	72.14	73.90	74.96	-
UnifiedQA	65.57	77.92	81.61	72.34	75.58	34.65	82.70	58.43	36.01	61.62	42.45	-
OOD MetaQA	62.26	69.41	66.59	75.02	67.51	**50.51**	72.20	58.59	52.13	59.28	22.14	-
OOD TWEAC	57.65	43.98	57.93	66.62	65.37	47.32	69.59	47.46	50.59	59.16	20.53	-6.31
OOD UnifiedQA	60.12	62.21	63.02	69.33	61.49	32.84	70.07	50.57	29.35	44.93	22.30	-8.12
OOD MultiQA*	**63.36**	**69.44**	**67.94**	**76.09**	**68.52**	49.89	**72.53**	n.a.	n.a.	n.a.	n.a.	0.61
OOD Max Voting	63.25	67.59	61.76	73.81	68.27	50.48	68.92	**58.94**	**64.03**	**63.22**	**22.46**	0.64

Table 2: Results of leave-one-out ablation. Out-of-domain (OOD) models are trained on all the datasets except the target dataset. Best OOD results in bold. Underlined results reflect OOD MetaQA outperforming full UnifiedQA. Δ is the average performance gap to OOD MetaQA. * MultiQA uses a pseudo-OOD setup, see remarks in §5.2.

Dataset	Question	In-domain Agent	OOD Agent
DuoRC	Who does Rocky Balboa work for as an enforcer?	Adrian	Tony Gazzo (NewsQA Agent)
TriviaQA-web	Who played the character Mr Chips in the 2002 TV adaptation of Goodbye Mr Chips?	Timothy Carroll	MartinClunes (DuoRC Agent)
SearchQA	This short story, written around 1820, contains the line "If I can but reach that bridge... I am safe"	Legend	Legend of Sleepy Hollow (TriviaQA Agent)

Table 3: Examples of questions where our MetaQA system disregard the in-domain agent due to their incorrect predictions (in red) and selects and an out-of-domain (OOD) agent that returns the right answer (in green).

setup, MultiQA only outperforms MetaQA by a slight margin of 0.61.

Lastly, we analyze the Max Voting baseline in this scenario. Although prior works disregard this baseline, the results in Table 2 show that OOD Max Voting outperforms all the other baselines and has a similar performance to OOD MetaQA. Its average gain with respect to OOD MetaQA is 0.64. However, this is not the overall trend. OOD MetaQA outperforms OOD Max Voting in 5/8 extractive QA datasets by a considerable margin of 3.19. On the other hand, multiple-choice datasets, especially the difference in HellaSWAG, incline the average towards OOD Max Voting. Despite the promising claims of prior works (Talmor and Berant, 2019; Khashabi et al., 2020) about OOD performance, these results suggest that aggregating a wide range of QA skills for different formats and domains in out-of-domain scenarios is still an open problem and non-neural baselines have strong results. Similar results have also been observed in retrieval methods, where non-neural baselines outperform supervised methods on OOD scenarios (Thakur et al., 2021).

5.3 Qualitative Analysis

We further analyze the behavior of our proposed model by inspecting its predictions. In particular, we investigate the collaboration between the agents for DuoRC, SearchQA, and TriviaQA, where this collaboration is particularly strong.

In DuoRC, the most helpful out-of-domain (OOD) agent is NewsQA, with a chosen rate of 18.2% in the test set. This might be due to the question types of DuoRC and NewsQA. DuoRC's questions are crowdsourced and are predominately *who-questions* (42% of the training set as shown in Appendix 6). NewsQA's questions are also crowdsourced and have a high proportion of *who-questions* (24%). The other datasets with a high amount of *who-questions* are NQ and SearchQA. However, the questions of these two datasets are very different in style to DuoRC (i.e., real user queries and trivia from a TV show). An example of this DuoRC-NewsQA agents collaboration is shown in the first row of Table 3.

In TriviaQA-web, the second most commonly used agent is trained on DuoRC. We randomly sampled 50 QA pairs where DuoRC is the selected agent and returns the right answer. In 20% of the cases, the question was about a movie or book plot, which indicates that our MetaQA successfully recognizes that this OOD agent is able to respond to this type of question. An example of this collaboration is shown in the second row in Table 3.

In SearchQA, the most helpful OOD agent is TriviaQA (5% chosen rate). This might be due to their similarities (Table 8). Within the pool of instances where the in-domain agent fails and the TriviaQA agent provides the right answer, we randomly analyzed 50 instances and discovered that in 84% of the cases, the in-domain agent returns a partially correct answer, and in those cases, the OOD agent was able to identify the exact answer. This is another example of the successful agent collaboration achieved by our MetaQA. Even though the in-domain agent almost has the correct answer, MetaQA selects an OOD agent that gives a better answer, as shown in the last row on Table 3.

5.4 Efficiency of MetaQA

We trained MetaQA with bins of QA instances for each dataset and observed that the training converges with only 10K instances/per dataset (i.e., 160K instances, including all datasets). This is only 16% of the data needed to train UnifiedQA (900K instances excluding HybridQA) and 13% of the data needed to train MetaQA (600K of extractive QA instances). The reason for this large saving is that MetaQA only has to learn how to match questions with answers because it reuses publicly available agents. On the other hand, multi-dataset models need to learn how to solve questions (i.e., language understanding, reasoning skills, etc.), a much more complex task.

As for inference time, if all the agents fit on memory[4], multi-datasets models and our MetaQA would have comparable running times. For example, compared to MultiQA, since our extractive agents use the same architecture as MultiQA, running the agents would take the same amount of time as running MultiQA. Then, we would need to select the answer. However, our MetaQA only takes 0.05s/question to select the best candidate answer. This makes it fast enough to not be noticeable by the users. On the other hand, if the agents do not fit in memory at the same time, it would be necessary to run them sequentially. Yet, this might not be a problem because it is possible to predict in advance which agents are more likely to give a correct answer to a given question (Geigle et al., 2021; Garg and Moschitti, 2021), which we leave as future work. This would allow us to skip some agents at run-time and improve the running time dramatically in low-memory scenarios.

5.5 Ablation Study

Lastly, we quantitatively measure the impact of each feature of MetaQA on its overall performance. The first row of Table 4 shows that removing the loss of the Agent Selection Network (AgSeN) hurts the performance of MetaQA. This manifests that our intuition of considering in-domain agents without falling into the *argumentum ad verecundiam* fallacy is correct. Lastly, the second row shows that the confidence embeddings provide key information to MetaQA to select an answer. For instance, an in-domain agent could have a prediction with low confidence because it does not know the answer, while an out-of-domain agent could have the correct answer and be certain about it.

Model	Avg. Downgrade
$-\ell_{AgSeN}$	-0.45
− Conf. Emb.	-0.46

Table 4: Average performance loss across all datasets of each ablated model compared to the full model.

6 Conclusions

In this work, we propose an alternative to multi-dataset models for multi-skill QA. We propose to combine expert agents to create a collaborative system for question answering (QA) called MetaQA. It considers questions, answer predictions, and confidence scores from the agents to select the best answer to a question. Through quantitative experiments, we show that our model avoids the limitations of multi-dataset models and outperforms the baselines thanks to the agent collaboration established. Additionally, since MetaQA learns to match questions with answers instead of end-to-end QA, it is highly data-efficient to train. We leave as future work: i) combining partially correct answer predictions to generate a better one, ii) adding new agents without retraining MetaQA by fixing most of the weights and only training the weights of the new Agent Selection Network, and iii) identifying *a priori* agents that are likely to give an incorrect answer to skip them at run-time.

Ethics Discussion

The proposed model, MetaQA, cannot generate unfair, biased, or harmful content given that the expert agents it aggregates are fair because MetaQA does not generate content. Rather it selects from Expert Agents. The datasets we use are well-known to be safe for research purposes and do not contain any personal information or offensive content. We also comply with the licenses and intended uses of each dataset. The licenses of each dataset are shown

---

[4] In our hardware and with our experimental setup, all agents and MetaQA fit on our GPU memory.

in Appendix A.1. We are not held responsible for errors, false or offensive content generated by the agents. MetaQA should be used at the users' discretion. Future work should address how to identify unfair or false content to avoid selecting it.

Limitations

The main limitation of MetaQA is that when no agent has a correct answer, it returns an incorrect answer. Table 5 describes how often this scenario occurs. In extractive datasets, without the outliers (i.e., SQuAD and DuoRC), we observe this to be 18% on average per dataset. This percentage drops to 8.35% in multiple-choice datasets (without BoolQ, another outlier). As for NarrativeQA and HybridQA, there are many unsolvable questions because we only use one agent for each of them and these agents have a relatively low performance.

Dataset	**% Unsolvable**
SQuAD	3.92
NewsQA	26.88
HotpotQA	19.93
SearchQA	13.97
NQ	19.15
TriviaQA-web	12.25
QAMR	15.81
DuoRC	47.41
BoolQ	1.47
SIQA	8.90
HellaSWAG	8.90
CSQA	9.00
RACE	6.61
DROP	21.77
NarrativeQA	55.71
HybridQA	56.09

Table 5: Percentage of unsolvable questions for our MetaQA with the selected agents, i.e., none of the agents can give a correct answer.

Also, if the agents do not fit in memory at the same time, it would be necessary to run them sequentially, which would increase the inference time. Yet, it might be possible to overcome this limitation because it is possible to predict in advance which agents are more likely to give a correct answer to a given question (Geigle et al., 2021; Garg and Moschitti, 2021). This would allow us to skip some agents at run-time and improve the running time dramatically in low-memory scenarios.

Acknowledgements

This work has been supported by the German Research Foundation (DFG) as part of the project UKP-SQuARE with the number GU 798/29-1 and by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

We thank Leonardo Ribeiro, Tim Baumgärtner, Rachneet Sachdeva, Soumya Sarkar, Neha Warikoo, and the anonymous reviewers for their insightful feedback and suggestions on a draft of this paper. We also thank Hannah Sterz for creating Adapter BART for NarrativeQA.

References

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, Alexey Litinsky, Varvara Logacheva, Alexey Lymar, Valentin Malykh, Maxim Petrov, Vadim Polulyakh, Leonid Pugachev, Alexey Sorokin, Maria Vikhreva, and Marat Zaynutdinov. 2018. DeepPavlov: Open-source library for dialogue systems. In *Proceedings of ACL 2018, System Demonstrations*, pages 122–127, Melbourne, Australia. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *ArXiv preprint*, abs/1704.05179.

Daria Dzendzik, Jennifer Foster, and Carl Vogel. 2021. English machine reading comprehension datasets: A survey. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8784–8804, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Dan Friedman, Ben Dodge, and Danqi Chen. 2021. Single-dataset experts for multi-dataset question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6128–6137, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Siddhant Garg and Alessandro Moschitti. 2021. Will this question be answered? question filtering via answer model distillation for efficient question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7329–7346, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gregor Geigle, Nils Reimers, Andreas Rücklé, and Iryna Gurevych. 2021. TWEAC: Transformer with extendable qa agent classifiers. *ArXiv preprint*, abs/2104.07081.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. 2018. Crowdsourcing question-answer meaning representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 560–568, New Orleans, Louisiana. Association for Computational Linguistics.

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84, Copenhagen, Denmark. Association for Computational Linguistics.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. XtremeDistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2221–2234, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. QA dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension. *ACM Computing Surveys (CSUR)*.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2019. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4911–4921, Florence, Italy. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR:

A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Appendix

A.1 Datasets

Table 8 summarizes the characteristics of the datasets, contains the size of the train, validation, and test splits of each dataset, and their licenses. In the case of RACE, the authors did not provide any license but specified that it could only be used for non-commercial research purposes. In the case of CommonSenseQA and SIQA there is no license specified, but they are freely available to download. Therefore, our use of these datasets complies with their licenses and intended uses.

A.2 Expert Agents

Table 9 provides the links to download the expert agents used in this work. In the case of NarrativeQA and HybridQA, we only employ one agent because of the difficulty of obtaining others. Both of these datasets use uncommon modalities (abstractive and table+text). Therefore, it is not straightforward to adapt other models to these datasets.

A.3 Implementation

Our model was implemented using PyTorch (Paszke et al., 2019) and HuggingFace's Transformers library (Wolf et al., 2020). Both MetaQA and MultiQA were implemented using Span-BERT large (335M parameters), while UnifiedQA uses T5-base (220M parameters, the closest to the 335M of our MetaQA). The score embedder for MetaQA is implemented as a linear layer with an input size of 1 and an output size of 1024 (i.e., the hidden size of Span-BERT Large). α_1 and α_2 in Eq. 4 are set to 0.5 and 1 respectively. The Agent Selection Networks are implemented as a linear layer with an input size of 1024 and an output size of 1. Lastly, the Answer Selection Network (AnsSel) is also implemented as a linear layer with an input size of *number-of-agents* $\times$ 1025 (Span-BERT's hidden size + 1 from the output of the agent selection network). The threshold θ to determine whether a candidate answer is correct or not to create the labels to train AnsSel is set to 0.7.

MetaQA was trained for one epoch using a batch size of six, a weight decay of 0.01, a learning rate of 5e-5, and 500 warmup steps.

All the extractive agents and MultiQA were trained using the same architecture, Span-BERT large, for two epochs and with the same hyperparameters: batch size of 16, learning rate of 3e-5, max length of 512, and doc stride of 128.

UnifiedQA was trained for two epochs using a batch size of four, a learning rate of 5e-5, and a weight decay of 0.01. It was evaluated on the dev set every 100K steps.

Lastly, the max-voting baseline assumes that two answers are the same if the F1 score is higher than a threshold (0.9). We tuned this parameter on the dev set searching in the range $[0.5, 0.6, ..., 1.0]$. We used the implementation of HuggingFace's SQuAD F1 metric[5]. In the case that two answers have the same amount of votes, we select the one with the highest confidence score given by an agent.

Any other parameter used the default value in HuggingFace's Transformers library. Each model was trained five times with different random seeds to do hypothesis testing except for UnifiedQA, which would be too expensive to compute.

We used the implementation of HuggingFace's Dataset library (Lhoest et al., 2021) for the evaluation using EM and F1 metrics, while for the ROGUE metric we used the official implementation[6].

All the experiments were conducted in a SLURM cluster where each job was assigned to different computer nodes with different CPUs and GPUs. Therefore, comparing the running time of each model is not possible.

A.4 Adding New Agents

Augmenting MetaQA with a new agent only requires adding one more AgSeN network and increasing the output space of the AnsSel network. Thus, it requires retraining the whole architecture (including the Transformer encoder). However, as discussed in §5.4, the training efficiency is one of the strengths of our system.

A.5 MetaQA on a Single Dataset

We conduct an additional experiment to analyze the behavior of MetaQA with multiple expert agents trained in a single dataset. We train MetaQA for three NewsQA agents: RoBERTA-base, XtremeDistil (Mukherjee and Hassan Awadallah, 2020), and SpanBERT, and evaluate it on NewsQA. As observed in Table 7, MetaQA performs on par with the agents. However, the performance gap is smaller than in the main use case (§5.1). This is attributed to the similarities between

---

[5] https://huggingface.co/metrics/squad

[6] https://pypi.org/project/rouge-score/

Dataset	**what**	**where**	**who**	**when**	**why**	**which**	**how**
SQuAD	56.71	4.55	10.82	7.47	1.48	7.73	11.23
NewsQA	49.52	8.54	24.46	5.01	0.11	3.17	9.19
HotpotQA	37.98	4.61	22.99	2.22	0.05	29.39	2.76
SearchQA	7.55	9.5	32.53	28.66	0.72	18.32	2.72
NQ	16.58	13.05	40.02	20.35	0.63	3.25	6.11
TriviaQA-web	30.16	1.56	15.07	0.72	0.02	50.03	2.44
QAMR	61.75	5.23	17.92	4.59	0.66	3.04	6.82
DuoRC	35.16	9.68	42.32	2.06	2.44	1.89	6.45

Table 6: Statistics of wh-words per dataset.

Model	F1 Score
MetaQA	**73.73**
SpanBERT	73.68
RoBERTa	73.15
XtremeDistil	64.16

Table 7: MetaQA trained only on NewsQA agents.

the models. These three models are all Transformers and trained on the same dataset, so it is natural that they are similar. An approach such as MetaQA excels when the agents are very different, as in Table 1, where the agents were trained on different datasets and therefore have different skills.

A.6 Wh-word Statistics

Table 6 shows the percentage of wh-words per dataset.

	Dataset	Characteristics	Train	Dev	Test	License
Extractive	SQuAD (Rajpurkar et al., 2016)	Crowdsourced questions on Wikipedia	6573	5253	5254	MIT
	NewsQA (Trischler et al., 2017)	Crowdsourced questions about News	74160	2106	2106	MIT
	HotpotQA (Yang et al., 2018)	Crowdsourced multi-hop questions on Wikipedia	72928	2950	2951	MIT
	SearchQA (Dunn et al., 2017)	Web Snippets, Trivia questions from J! Archive	117384	8490	8490	MIT
	NQ (Kwiatkowski et al., 2019)	Wikipedia, real user queries on Google Search	104071	6418	6418	MIT
	TriviaQA-web (Joshi et al., 2017)	Web Snippets, crowdsorced trivia questions	61688	3892	3893	MIT
	QAMR (Michael et al., 2018)	Wikipedia, predicate-argument understanding	50615	18908	18770	MIT
	DuoRC (Saha et al., 2018)	Movie Plots from IMDb and Wikipedia	58752	13111	13449	MIT
Multiple-Choice	RACE (Lai et al., 2017)	Exams requiring passage summarization and attitude analysis	87866	4887	4934	NA
	CSQA (Talmor et al., 2019)	Web Snippets, common-sense reasoning	9741	611	610	NA
	BoolQ (Clark et al., 2019)	Wikipedia, Yes/No questions	9427	1635	1635	CC BY-SA 3.0
	HellaSWAG (Zellers et al., 2019)	Completing sentences using common sense	39905	5021	5021	MIT
	SIQA (Sap et al., 2019)	Common sense in social interactions	33410	977	977	NA
Abs.	DROP (Dua et al., 2019)	Wikipedia, numerical reasoning	77409	4767	4768	CC BY-SA 4.0
	NarrativeQA (Kočiský et al., 2018)	Books, Movie Scripts	32747	3461	10557	Apache 2.0
MM	HybridQA (Chen et al., 2020)	Wikipedia tables and paragraphs	62682	1733	1733	MIT

Table 8: Summary of the datasets used. Abs. stands for abstractive and MM for multi-modal.

#	Expert Agents	Used for	Link
1	Span-BERT Large (Joshi et al., 2020) for SQuAD	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_SQuAD
2	Span-BERT Large for NewsQA	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_NewsQA
3	Span-BERT Large for HotpotQA	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_HotpotQA
4	Span-BERT Large for SearchQA	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_SearchQA
5	Span-BERT Large for NQ	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_NaturalQuestionsShort
6	Span-BERT Large for TriviaQA-web	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_TriviaQA-web
7	Span-BERT Large for QAMR	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_QAMR
8	Span-BERT Large for DuoRC	all extractive + DROP	https://huggingface.co/haritzpuerto/spanbert-large-cased_DuoRC
9	RoBERTa Large (Liu et al., 2019) for RACE	all multiple choice	https://huggingface.co/LIAMF-USP/roberta-large-finetuned-race
10	RoBERTa Large for HellaSWAG	all multiple choice	https://huggingface.co/prajjwal1/roberta_hellaswag
11	RoBERTa Large for SIQA	all multiple choice	https://huggingface.co/haritzpuerto/roberta_large_social_i_qa
12	AlBERT xxlarge-v2 (Lan et al., 2020) for CSQA	all multiple choice	https://huggingface.co/danlou/albert-xxlarge-v2-finetuned-csqa
13	BERT Large-wwm (Devlin et al., 2019) for BoolQ	BoolQ	https://huggingface.co/lewtun/bert-large-uncased-wwm-finetuned-boolq
14	TASE (Segal et al., 2020) for DROP	DROP	https://github.com/eladsegal/tag-based-multi-span-extraction
15	Adapter BART Large (Pfeiffer et al., 2020) for NarrativeQA	NarrativeQA	https://huggingface.co/AdapterHub/narrativeqa
16	Hybrider (Chen et al., 2020) for HybridQA	HybridQA	https://github.com/wenhuchen/HybridQA

Table 9: List of the expert agents, datasets in which they are used, and links to download.

BERT is not The Count: Learning to Match Mathematical Statements with Proofs

Weixian Waylon Li[1] **Yftah Ziser**[1] **Maximin Coavoux**[2*] **Shay B. Cohen**[1]
[1] University of Edinburgh, School of Informatics, Edinburgh
[2] Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG,
`W.Li-67@sms.ed.ac.uk` `yftah.ziser@ed.ac.uk`

Abstract

We introduce a task consisting in matching a proof to a given mathematical statement. The task fits well within current research on Mathematical Information Retrieval and, more generally, mathematical article analysis (Mathematical Sciences, 2014). We present a dataset for the task (the MATCH dataset) consisting of over 180k statement-proof pairs extracted from modern mathematical research articles.[1] We find this dataset highly representative of our task, as it consists of relatively new findings useful to mathematicians. We propose a bilinear similarity model and two decoding methods to match statements to proofs effectively. While the first decoding method matches a proof to a statement without being aware of other statements or proofs, the second method treats the task as a global matching problem. Through a symbol replacement procedure, we analyze the "insights" that pre-trained language models have in such mathematical article analysis and show that while these models perform well on this task with the best performing mean reciprocal rank of 73.7, they follow a relatively shallow symbolic analysis and matching to achieve that performance.[2]

1 Introduction

Research-level mathematical discourse is a challenging domain for Natural Language Processing (NLP). Mathematical articles frequently switch between natural language and mathematical formulae, and a semantic analysis of mathematical text needs to solve relationships (e.g. coreference) between mathematical symbols and concepts. Moreover, mathematical writing follows many conventions, such as variable naming or typography that are implicit, and may differ between subfields.

Statement. When $m = 0$ we have $E^0_{rg} = \emptyset$, and when $m \neq 0$ we have $E^0_{rg} = E^0$.
Proof. When $m = 0$, the image of r is $\{1\}$. Hence $E^0_{rg} = \emptyset$. When $m \neq 0$, the map r is a surjective proper map. Hence $E^0_{rg} = E^0$.

Figure 1: Example of a statement-proof pair.

However, mathematical research can benefit from NLP (Mathematical Sciences, 2014), in particular as concerns bibliographical research: researchers need tools to find work relevant to their research. Indeed, prior NLP work on mathematical research articles focused on Mathematical Information Retrieval (MIR) and related tools or data (Zanibbi et al., 2016; Stathopoulos and Teufel, 2016, 2015).

We introduce a task aimed at improving the processing of research-level mathematical articles and make a step towards the modeling of mathematical reasoning. Given a collection of mathematical statements and a collection of mathematical proofs of the same size, the task consists in finding and assigning a proof to each mathematical statement. We construct and release a dataset for the task (MATCH), by collecting over 180k statement-proof pairs from mathematical research articles (an example is given in Figure 1).

Related datasets, such as LEANSTEP (Han et al., 2021) and the synthetic dataset of Polu and Sutskever (2020) do not include natural language. NaturalProofs (Welleck et al., 2021), another related dataset, only consists of 32k theorem-proof pairs from ProofWiki,[3] some sub-topics in algebraic geometry and two textbooks. Our dataset is over five times larger and contains pairs extracted

---

[*] Work mostly done at the University of Edinburgh.

[1] Our dataset and code are available at `https://github.com/waylonli/MATcH`.

[2] Like Bert, The Count (or Count von Count; ) is a character from the television show Sesame Street. The Count likes counting, and his main role in the show is to teach this skill to children.

[3] `https://proofwiki.org/xmldump/latest.xml`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3581–3593
May 2-6, 2023 ©2023 Association for Computational Linguistics

from advanced academic mathematical papers.

There are multiple motivations for the design of the task and our dataset. We believe it may help MIR by serving as a proxy for the search for the existence of a mathematical result, or for theorems and proofs related to one another (e.g. using the same proof technique), an important search tool for any digital mathematical library (Mathematical Sciences, 2014). Learning to match statements and proofs would also benefit computer-assisted theorem proving, as it is akin to tasks such as premise selection, also recently addressed with NLP methods (Piotrowski and Urban, 2019).

We provide first results on our proposed task with an array of neural models, aimed at scoring the likelihood of relationship between a statement and its proof. An analysis through a **symbol replacement procedure** provides insight on what such neural models are capable of learning about mathematical equations and text.

We provide two methods for decoding, one is local decoding, matching a proof to a theorem in a greedy way, and one that provides a global bipartite matching based on a structured max-margin objective. Such an architecture may have applications to other NLP problems that can be cast as maximum bipartite matching problems (for a recent similar use in a different context, see Shao et al. 2023).

Our analysis shows that pre-trained language models do not obtain significant "mathematical insight" for performing this matching, but rather rely on shallow matching. However, this does not prevent them from performing the matching relatively well in several carefully crafted scenarios, reaching an MRR of 73.7.

2 Related Work

Most NLP work on mathematical discourse focuses on improving Mathematical Information Retrieval (Zanibbi et al., 2016, MIR) by establishing connections between mathematical formulae and natural language text in order to improve the representation of formulae.

The interpretation of variables is highly dependent on the context. For example, the symbol E could denote an expectation in a statistics article, or the energy in a physics article. Some studies use the surrounding context of a formula to assign a definition or a type to the whole formula, or to specific variables. Nghiem Quoc et al. (2010) focus on identifying coreferences between mathematical formulae and mathematical concepts in Wikipedia articles. Kristianto et al. (2012) extract definitions of mathematical expressions. Grigore et al. (2009), Wolska et al. (2011) and Schubotz et al. (2016) disambiguate mathematical identifiers, such as variables, using the surrounding textual context. Stathopoulos et al. (2018) infer the type of a variable in a formula from the textual context of the formula. Another line of work focused on identifying specialized terms or concepts to improve MIR (Stathopoulos and Teufel, 2015, 2016).

Some work adapts standard NLP tools to the specificity of mathematical discourse, e.g. POS taggers (Schöneberg and Sperber, 2014), with the objective of using linguistic features to improve the search for definitions of mathematical expressions (Pagel and Schubotz, 2014). More recent work focuses on typing variables in mathematical articles (Ferreira et al., 2022), modeling formulae (Mansouri et al., 2019; Dadure et al., 2021), and selecting premises (Ferreira and Freitas, 2020, 2021).

An earlier version of our work covers some of the material in this paper (Coavoux and Cohen, 2021). The main differences between that version and the current version are the introduction of the symbol replacement evaluation (§5) and the use of pre-trained language models rather than recurrent neural networks.

3 Task Description

Given a collection of mathematical statements $\{s^{(i)}\}_{i\leq N}$, and a separate equal-size collection of mathematical proofs $\{p^{(i)}\}_{i\leq N}$, we are interested in the problem of assigning a proof to each statement.

Evaluation We use two evaluation metrics. Assuming that a system predicts a ranking of proofs, instead of providing only a single proof, we evaluate its output with the Mean Reciprocal Rank (MRR) measure: $\text{MRR}(\{\hat{r}_i\}_{i\in\{1,\ldots N\}}) = \frac{1}{N}\sum_{i=1}^{N}\dfrac{1}{\hat{r}_i}$, where N is the number of examples and $\hat{r}_i$ is the rank of the gold proof for statement number i, as predicted by the system.

As a second evaluation metric, we use a simple accuracy, i.e. the proportion of statements whose first-ranked proof is correct.

By construction (see §4), it is possible though unlikely that the same mathematical statement occurs several times in the dataset. It is more unlikely that

Number of articles in the MREC corpus	439,423
Extracted articles with statement-proof pairs	27,841
Total number of statement-proof pairs	184,094
Number of (primary) categories	(120) 135
Average number of categories per article	1.7

Most represented primary categories	# articles	# pairs
math.AG Algebraic Geometry	2848	22029
math.DG Differential Geometry	2030	12440
math.CO Combinatorics	1705	10548
math.GT Geometric Topology	1539	9234
math.NT Number theory	1454	9521
math.PR Probability	1422	7660
math.AP Analysis of PDEs	1386	6981
math-ph Mathematical Physics	1249	6491
math.FA Functional Analysis	1143	8011
math.GR Group Theory	970	7806
math.DS Dynamical System	961	6424
math.QA Quantum Algebra	944	8074
math.OA Operator Algebras	923	8050

Table 1: Statistics about the dataset and categories of mathematical articles.

several occurrences have exactly the same formulation and use the same variable names. Therefore, we consider a match to be correct if and only if it is associated with its original proof.

4 Dataset Construction

This section describes the construction of the MATCH dataset of statement-proof pairs (see Figure 1 for an example).

Source Corpus We use the MREC corpus[4] (Líška et al., 2011) as a source. The MREC corpus contains around 450k articles from ArxMLiV (Stamerjohanns et al., 2010), an on-going project aiming at converting the arXiv[5] repository from LaTeX to XML, a format more suited to machine processing. In this collection, mathematical formulae are represented in the MathML[6] format, a markup language.

Statistics We extract statement-proof pairs as described in Appendix A. Our processing of MREC includes the identification of statement-proof pairs through meta tags and the linearization of the representation of mathematical equations.

We report in Table 1 some statistics about the dataset we collected. The extracted articles were from a diverse set of mathematical subdomains, and connected domains, such as computer science (746 articles from 30 subcategories) and mathematical physics (2562 from 31 subcategories). There are in average 6.6 statement-proof pairs per article. We report statistics about the size of statements and proofs in the number of tokens in Table 2. We report the number of tokens in formulae (math), in the text itself (text) and in both (text+math). On average, proofs are much longer than statements. Statements and proofs have approximately the same proportion of text and math. Overall, the variation in the number of tokens across statements and proofs is extremely high, as illustrated by the standard deviation (SD) of all presented metrics.

Statements	Min	Max	Mean±SD
Text+math	20	500	80±57
Text only	1	398	30±20
Math only	0	470	58±20
Math proportion	0%	99.5%	58%±20
Proofs			
Text+math	20	500	210± 127
Text only	1	467	81 ± 56
Math only	0	495	129 ± 96
Math proportion	0%	99.6%	56%± 21

Table 2: Number of tokens in the dataset. We report for statements and proofs the minimum, maximum and average number of tokens broken down by type ('math' for tokens extracted from formulae and 'text' for the others). A value of 0 for, e.g. the 'math only' row, means that the statement or proof does not contain mathematical symbols or formulae.

5 Symbol Replacements

With our current dataset setup, we implicitly make the assumption that both the theorem and the proof are authored by the same authors. This assumption is incongruent with the MIR-flavor of our task. First, it is not useful for researchers to match proofs they authored. Second, each person has a unique writing style expressed by unique mathematical jargon and notations. To relieve of this assumption, we introduce several symbol replacement levels for changing the names of the proof variables. Then, we train and test our models using these altered datasets. These replacement levels also provide insight on the ability of our models to semantically analyze the input statement-proof pairs.

Symbol Replacement Levels We propose different levels of symbol replacement, focusing on mathematical notation. More precisely, we aim to replace the proof variable names if they appear in

---

[4] `https://mir.fi.muni.cz/MREC/`, version 2011.4.439.

[5] `https://arxiv.org/`

[6] `https://www.w3.org/Math/`

the statement without damaging the proof semantics. To do that, we change symbols that appear both in the proof and the statement. We do not change constant symbols such as π, as they often carry semantic meaning outside of the proof scope.

We experiment with four levels of symbol replacement (examples in parenthesis):

- **Symbol conservation** - all symbols remain intact, so the theorem and the proof overlap. All previous work uses that. $(a_n = a_{n-1} + a_{n-2})$
- **Partial symbol replacement** - A fraction of α of all the symbols in the proof remain the same, and the rest are changed. In our experiments, we use $\alpha = 0.5$. $(x_n = x_{n-1} + x_{n-2})$
- **Full symbol replacement** - all symbol names are changed ($\alpha = 1.0$ as above). $(x_i = x_{i-1} + x_{i-2})$
- **Symbol transposition** - We permute the variables' names such that no symbol remains the same, thus changing their original functionality. $(n_a = n_{a-1} + n_{a-2})$

More details about this appear in Appendix C.

6 Bilinear Similarity Model

We propose a model based on an encoder (§8) that constructs fixed-size vector representations for statements and proofs, and a similarity function that scores the relatedness of a statement-proof pair.

Trainable Bilinear Similarity Function Given the encoded representations of a statement $\mathbf{s} = \text{enc}(s)$ and a proof $\mathbf{p} = \text{enc}(p)$, we compute an association score with the following bilinear form:

$$\text{score}(\mathbf{s}, \mathbf{p}) = \mathbf{s}^\top \cdot \mathbf{W} \cdot \mathbf{p} + b,$$

where $\mathbf{W}$ and b are parameters that are learned together with a self-attentive encoder parameters (§8).

Local Decoding For a collection of n statements and proofs, we first score all possible pairs (s, p), and construct a matrix $M = (m_{ij}) \in \mathbb{R}^{n \times n}$, with

$$m_{ij} = \text{score}(\mathbf{s}^{(i)}, \mathbf{p}^{(j)}),$$

where $\mathbf{s}^{(i)}$ and $\mathbf{p}^{(j)}$ are the encoded representations of, respectively, the i^{th} statement and the j^{th} proof. Then we can straightforwardly sort each row by decreasing order and assign the proof ranking to the corresponding statement. The best ranking proof $\hat{p}$ for statement i satisfies $\hat{p}^{(i)} = \arg\max_j \; m_{ij}$. We call this decoding method 'local', since it does not take into account dependencies between assignments. In particular, several statements may have the same highest-ranking proof.

Global Decoding The local decoding method overlooks a crucial piece of information: a proof should correspond to a single statement. In a worst-case situation, a small number of proofs may score high with most statements and be systematically assigned as highest-ranking proof by the local decoding method.

In preliminary experiments, we analyzed the output of our system with local decoding on the development set, focusing on the distribution of the single highest-ranking proof for each statement. We found that 23% of the proofs were assigned to at least two different statements, whereas more than 40% of proofs were assigned to no statement. See also Appendix B.[7]

We propose a second decoding method based on a global constraint on the output: a proof can be assigned only to a single statement. Intuitively, the constraint models the fact that if a proof is assigned by the system to a certain statement with high confidence, we can rule it out as a candidate for other statements. Under this constraint, the decoding problem reduces to a classical maximum weighted bipartite matching problem, or equivalently, a Linear Assignment Problem (LAP). In more realistic scenarios (e.g. if the input sets of statements and proofs do not have the same size), the method would require some adaptation.

Formally, we define an assignment A as a Boolean matrix $A = (a_{ij}) \in \{0, 1\}^{n \times n}$ with the following constraints:

$$\forall i \forall j, \sum_j a_{ij} = \sum_i a_{ij} = 1,$$

i.e. each row and each column of A contains a single non-zero coefficient. The score of an assignment A is the sum of scores of the chosen edges:

$$\text{score}(A, M) = \sum_i \sum_j a_{ij} m_{ij}.$$

Finally, global decoding consists in solving the

---

[7] We used a simple encoder for these experiments, which we describe in §8 (NPT).

following LAP:

$$\hat{A}(M) = \underset{\substack{A \in \{0,1\}^{n \times n} \\ \text{s.t. } \forall i \forall j, \sum_j a_{ij} = \sum_i a_{ij} = 1}}{\arg\max} \text{score}(A, M).$$

The LAP is solved in polynomial time by the Hungarian algorithm (Kuhn, 1955), the LAP-Jonker-Volgenant algorithm (LAP-JV; Jonker and Volgenant, 1987), or the push-relabel algorithm (Goldberg and Kennedy, 1995). These methods have a $\mathcal{O}(n^3)$ time complexity where n is the number of pairs, and $\mathcal{O}(n^2)$ memory complexity. This is too expensive in our case, due to our dataset size.

To remedy this limitation, when we perform decoding on a large set, we only consider the k best-scoring proofs (i.e. outgoing edges in the bipartite graph) for each statement, which makes the number of edges linear in the number of pairs n (considering k fixed). Moreover, we use a modification of the LAP-JV algorithm specifically designed for sparse matrices (LAP-MOD; Volgenant, 1996).

7 Local and Global Training

We propose two training methods for the similarity model above: a local training method that only considers statements in isolation (§7.1) and a global model trained to predict a bipartite matching (§7.2), with a hybrid global-local objective.

7.1 Local Training

We would like to train our model to assign a high similarity to the gold statement-proof pair, and a low similarity to all other statement-proof pairs. This corresponds to the following objective, for a single statement s and its gold proof p:

$$\begin{aligned}\mathcal{L}_{\text{LOC}}(s, p, P; \boldsymbol{\theta}) &= -\log \mathbb{P}(p|s; \boldsymbol{\theta}) \\ &= -\log \left(\frac{\exp(\text{score}(\mathbf{s}, \mathbf{p}))}{\sum_{p' \in P} \exp(\text{score}(\mathbf{s}, \mathbf{p'}))} \right),\end{aligned}$$

where P is the set of proofs, and $\boldsymbol{\theta}$ are the parameters of the model. Directly optimizing this loss function requires the computation of $\mathbf{p} = \text{enc}(p)$ for every proof in the dataset, for a single optimization step. This is not realistic considering memory limitations, the size of the train set, and the fact that our self-attentive encoder is the most computationally expensive part of the network.

Instead, we sample minibatches of b pairs and optimize the following proxy loss for the sequence $S' = (s_1, \dots, s_b)$ of statements and the sequence $P' = (p_1, \dots, p_b)$ of corresponding proofs:[8]

$$\mathcal{L}'_{\text{LOC}}(S', P'; \boldsymbol{\theta}) = \sum_{i=1}^{b} \mathcal{L}_{\text{LOC}}(s^{(i)}, p^{(i)}, P'; \boldsymbol{\theta}).$$

In practice, we sample uniformly and without replacement b pairs from the training set at each stochastic step.

7.2 Hybrid Local and Global Training

The local training method only considers statements in isolation. Even though we expect a locally trained model to perform better with global decoding, we hypothesize that a model that is trained to predict the full structure (a bipartite matching) will be even better.

For a collection of n proofs and n statements, the size of the search space (i.e. the number of bipartite matchings) is $n!$, since each matching corresponds to a permutation of proofs. As a result, the use of a globally normalized model is impractical. We turn to a max-margin model that does not require normalization over the full search space.

We use the following max-margin objective, for a set B of n pairs corresponding to matrix M:

$$\begin{aligned}\mathcal{L}_{\text{GLOB}}(B; \boldsymbol{\theta}) = \max(0, &\Delta(\hat{A}, I) \\ &+ \text{score}(\hat{A}, M) - \text{score}(I, M)),\end{aligned}$$

where $\boldsymbol{\theta}$ is the set of all parameters $\hat{A}$ is the predicted assignment and I is the gold assignment, i.e. the identity matrix. The structured cost

$$\Delta(\hat{A}, I) = \sum_{ij} \max(0, (\hat{A} - I)_{ij})$$

aims at enforcing a margin for each individual assignment.

The computation of this loss requires exact decoding for each optimization step. Since exact decoding is only feasible for a small n, and since we need to keep track of all intermediary vectors to compute the backpropagation step,[9] we perform each stochastic optimization step on a minibatch of pairs of size b. Since this global objective had a slow convergence rate (§8), in practice, we use a hybrid local-global objective: $\mathcal{L}'_{\text{LOC}} + \mathcal{L}_{\text{GLOB}}$.

	Symbol Replacement Level							
	Conservation		Partial		Full		Transposition	
Encoder-Decoder	MRR	Acc	MRR	Acc	MRR	Acc	MRR	Acc
NPT-Local-Local	63.22	56.08	47.19	39.24	40.36	32.52	56.17	48.30
NPT-Local-Global	-	61.89	-	42.55	-	35.43	-	53.49
NPT-Global-Global	-	62.14	-	43.68	-	35.85	-	55.28
ScratchBERT-Local-Local	**73.73**	67.12	**64.79**	57.20	**60.67**	52.54	**73.17**	66.51
ScratchBERT-Local-Global	-	**74.68**	-	**62.80**	-	**57.69**	-	**74.03**
ScratchBERT-Global-Global	-	71.38	-	58.06	-	52.31	-	70.32
MathBERT-Local-Local	54.51	46.45	44.31	36.10	38.91	30.62	52.57	44.52
MathBERT-Local-Global	-	49.77	-	37.92	-	32.03	-	47.43
MathBERT-Global-Global	-	45.38	-	33.64	-	28.47	-	43.41

Table 3: The MRR and accuracy scores for different combinations of encoders, decoders, and symbol replacement levels. All the models are trained and tested on the same replacement level. Best result in each column is in bold. Following the model name, we include its encoder and decoder type (both being either Local or Global). We do not include MRR scores for global inference, as there matching is done for all theorems together without ranking.

8 Experimental setup

Dataset We use the dataset whose construction is described in §4. We shuffle the collection of statement-proof pairs before performing a $80\%/10\%/10\%$ train-development-test split, corresponding to 147278 pairs for the training sets and 18408 pairs for the development and tests. We experiment with a default, Mixed, split and a harder Unmixed split (see §9.3).

Encoders We experiment with several encoders to obtain neural representations of the theorem and proof pairs. Our first encoder is a simple self-attentive encoder. We use $\ell = 2$ self-attentive layers with 4 heads to obtain contextualized embeddings of dimension $d = 300$. The query and key vectors have size $d_k = 128$. We construct a vector representation for the text with a max-pooling layer over the contextualized embeddings of the last self-attention layer. We do not use any form of pre-training for this encoder and hence name it "no pre-training encoder" (NPT). In addition, we experiment with a BERT model (Devlin et al., 2019) as an encoder. We do not use the pre-trained version provided by Devlin et al., but rather pre-train the base version from scratch (ScratchBERT), but we do compare our results against a math-tailored pre-trained version of BERT (Peng et al. 2021; see below). Both the NPT and ScratchBERT vocabularies are customized for our dataset, as preliminary experiments revealed the importance of the model-task vocabulary match.[10]

To further demonstrate how crucial this vocabulary match is, we experiment with math-BERT (MathBERT; Shen et al. 2021), a state-of-the-art pre-trained model for mathematical formula understanding. This model is pre-trained on a large mathematical corpus ranging from pre-kindergarten, to high-school, to college graduate level mathematical content, including professional mathematical papers, using the BERT masked language modeling (MLM) task. We use the pre-trained version provided by the authors, *without vocabulary customization*. All of our encoders are fine-tuned on the matching task. In addition, we experiment with a naive token-matching system that computes cosine similarities between TF-IDF representations of statements and proofs. We discovered that their performance was very low, ranging from 11.4 to 29.8 (MRR), so we did not experiment with them further.

Hyperparameters For pretraining ScratchBERT, we first train a new word piece tokenizer[11]. Next, we train the ScratchBERT model on the MLM task for 60 epochs (around 3 days) using four NVIDIA V100 GPUs. We evaluate the language model every 500 steps, where one step stands for training on one example, and choose the one with

---

[8] We also experimented with a Noise-Contrastive Estimation approach (Gutmann and Hyvärinen, 2012). However, it exhibited a much slower convergence rate.

[9] In particular, the computation graph needs to conserve all encoding layers for the $2n$ texts involved.

[10] This supports the findings of Chalkidis et al. (2020), for example, in a different domain.

[11] `https://huggingface.co/docs/transformers/tokenizer_summary#wordpiece`

the best performance on the validation set.

We perform local and global training / finetuning respectively for the NPT model, MATHBERT, and SCRATCHBERT. NPT has 15M parameters while MATHBERT and SCRATCHBERT have 110M parameters. We observed in initial experiments that training only with the global objective required a long time to converge. Therefore, we used the following global-local objective: $\mathcal{L}'_{\text{LOC}} + \mathcal{L}_{\text{GLOB}}$, that we optimized by alternating one stochastic step for each loss.

We train the NPT model for 400 epochs (around 1 day with two GPUs) over the whole training set for local and global training. We use batches of size $b = 60$ and set learning rate $l = 5 \times 10^{-3}$ with the Averaged Stochastic Gradient Descent (ASGD; Polyak and Juditsky 1992) optimizer. We use an exponential learning rate scheduler (the learning rate multiplied by 0.996 after each epoch) to stabilize the optimizer in the latter training procedure (after 300 epochs). We evaluate the performance of the model on the validation set every 20 epochs during training and select the best one among these intermediate models.

We use four NVIDIA V100 GPUs to fine-tune MATHBERT and SCRATCHBERT on the training set for 60 epochs (around 2 days) with a learning rate of $l = 2 \times 10^{-3}$, an ASGD optimizer, batches of size $b = 16$, and a scheduler that multiplies the learning rate by 0.99 after each epoch. We choose the best model on the validation set, evaluating the models every five epochs.

Global Decoding Recall that exact global decoding is only feasible for a small subset of pairs. During global training, we chose a batch size small enough to perform exact decoding. However, it is not feasible to perform exact decoding on the whole development and test corpora. Therefore, we prune the search space by keeping only the 500-best candidate proofs for each statement, and use the LAP-MOD algorithm designed for sparse matrices. In practice, we used the implementations of the LAP-JV and LAP-MOD algorithms from the `lap` Python package,[12] for respectively exact decoding on mini-batches during global training and decoding on whole datasets during evaluation.

9 Results

First, we assess the task difficulty under different replacement levels using different encoders and schemes (global or local training, global or local decoding). In particular, we are interested in assessing whether global decoding improves accuracy when training is only local, and how the more complex global training method fares with respect to local training. We then measure the informativeness of different types of input: text, mathematical formulae, or both. The comparison of these settings is meant to provide insight into which type of information is crucial to the task. Finally, we experiment with a cross replacement levels setup, i.e., when a model is tested on a different symbol replacement level from the one that was used during training. We hope this experiment will shed some light on the importance of training models on real-world datasets.

9.1 Main Results

Table 3 presents our results. We report MRR (if relevant) and accuracy scores across different levels of symbol replacement.

Encoders While MATHBERT is pre-trained on millions of examples curated from mathematical contents, it performs worse than the less complex NPT encoder, which is trained solely on the downstream task across all symbol replacement levels and decoders.[13] SCRATCHBERT, which shares MATHBERT architecture and NPT customized vocabulary, is outperforming both consistently. These results demonstrate the vocabulary importance for learning from mathematical texts.

Symbol Replacement Levels Difficulty Best performance is achieved when no symbol is replaced (Conservation), as the models can match identical symbols across theorem-proof pairs. The models achieve similar performance with Transposition replacement. These results suggest that the symbols' order, context, and function within the mathematical text do not play a significant role when the theorem and proof share the same symbols. In contrast, when the symbol names are changed (Partial and Full replacements), we observe a sharp decline in results.

---

[12] https://github.com/gatagat/lap

[13] We observe similar trends when fine-tuning the out-of-the-box BERT model on the matching task.

	Target Source	Symbol Replacement							
		Conservation		Partial		Full		Transposition	
		MRR	Acc	MRR	Acc	MRR	Acc	MRR	Acc
Mixed	Conservation	73.73	67.12	43.87	36.36	29.74	25.36	69.56	62.23
	Partial	**74.21**	**67.96**	**64.79**	**57.20**	53.77	45.40	72.13	65.42
	Full	65.26	57.63	63.01	55.13	**60.67**	**52.54**	64.59	56.92
	Transposition	73.78	67.40	43.67	36.02	29.76	25.47	**73.17**	**66.51**
Unmixed	Conservation	**67.62**	57.54	21.26	13.83	7.09	3.68	59.54	48.61
	Partial	61.19	50.73	**55.26**	**44.45**	50.68	39.94	59.63	49.01
	Full	55.68	45.18	54.92	44.34	**54.62**	**44.22**	55.38	44.91
	Transposition	67.5	**57.76**	23.31	15.26	8.98	4.97	**66.25**	**59.29**

Table 4: Cross-replacement levels performance for the ScratchBERT-Local-Local model for both splits: Mixed and Unmixed.

	Symbol Replacement			
	Conservation		Full	
Input	MRR	Acc	MRR	Acc
	NPT			
Text	22.51	16.68	22.51	16.68
Math	65.08	58.47	34.55	27.30
Both	63.22	56.08	40.36	32.52
	ScratchBERT			
Text	36.85	29.18	36.85	29.18
Math	63.10	55.92	41.64	34.01
Both	**73.73**	**67.12**	**60.67**	**52.54**

Table 5: ScratchBERT-Local-Local and NPT-Local-Local performance for different input types. Both stand for the original and complete input.

Training and Decoding Effects In all settings, global *decoding* substantially improves accuracy. These improvements are more noticeable for the NPT and ScratchBERT encoders. For NPT, we observe better performance when using global *training*, but not for ScratchBERT and MathBERT. Due to the lack of computational resources, we can not reach the global training full potential when using highly expressive encoders such as ScratchBERT and MathBERT, which share BERT-base architecture.

9.2 Effect of Input Type Analysis

To better understand the importance of each input type, we examine ScratchBERT-Local-Local and NPT-Local-Local performance when fed with text, mathematical formulae, or both (Table 5). We test them on the Conservation and Full symbol replacement levels. The mathematical formulae input plays a more significant role for both models than the textual input. When trained and tested on the Conservation replacement level, NPT-Local-Local makes better use of the mathematical formulae input than the more expressive, pre-trained ScratchBERT-Local-Local. When trained and tested with Full replacement, where the models cannot rely on simple token-matching, NPT-Local-Local suffers from a sharper performance decline than ScratchBERT-Local-Local when fed with mathematical formulae input. These results suggest that when applied to the Conservation data, a less expressive model can get high results by harnessing simple token matching. ScratchBERT-Local-Local performs better for both replacement levels when fed with text and complete input.

9.3 Cross Replacement Setup

Table 4 shows the effect of testing a model on different symbol replacements than the one the model was trained on. We use the ScratchBERT-Local-Local model for all of our experiments. We observe a sharp decline in results when ScratchBERT-Local-Local is trained with Conservation and tested on Partial or Full. These drops in performance suggest the model developed a strong dependency on exact symbol name matching. In addition, the replacement shift from Conservation to Transposition and vice versa resulted in a minor performance drop. These results provide additional evidence for the lack of importance of mathematical functionality, order, and context of symbols' names shared across theorem and proof pairs. The model trained on the Partial symbol replacement level demonstrated significant resilience when tested with other symbol replacement levels. It outperforms the rest of the models when applied to out-of-domain replacement levels and the Conservation replacement level in-domain model.

In addition, we experimented with theorem-proof pairs split where pairs from the same paper could not appear in the same set: train, validation or test (Unmixed). All models exhibited a reduction in performance when trained and tested under these conditions. Particularly noteworthy was the decrease in performance observed in models that were trained using the Conservation and Transposition symbol replacement methods and evaluated on data using the Partial or Full replacement methods. These sharp declines highlight the dependence of the model on simple symbol matching rather than deeper inferential analysis.

9.4 Protected Symbols

Insofar, we overlooked that some symbols carry a default meaning over a whole mathematical domain (**protected** symbols, e.g. $P(x)$ for probability). Replacing them locally may result in a detrimental impact on semantic mathematical content. We test the impact of substituting protected symbols in a controlled setting by comparing models trained with symbol replacement methods that preserve protected symbols versus methods that treat all symbols equally. The test set preserves the protected symbols. We follow the Unmixed setup, where theorem-proof pairs from the same paper must appear in the same split.

We focus on the probability theory domain. Focusing on a single domain enables us to construct a list of protected symbols more precisely. Our list consists of the P (probability measure), E (expected value), V (variance), σ (standard deviation and covariance), and ρ (correlation) symbols.[14] Table 6 shows that training the SCRATCHBERT-Local-Local model using the Partial+P replacement method results in slightly better results. We present only a subset of our results for brevity; the pattern re-occurs with all symbol replacement methods.

9.5 Qualitative Analysis

To study which tokens affect our model predictions, we use LIME (Ribeiro et al., 2016), a method for calculating feature importance. We examine SCRATCHBERT-Local-Local trained with the Conservation setup and with Full replacement. Both are applied to original test examples. We observe that the Conservation SCRATCHBERT-Local-Local model heavily relies on the mathematical tokens and barely benefits from the text ones. In contrast, the SCRATCHBERT-Local-Local model that was trained in the full symbol replacement setup strongly relies on textual tokens with mathematical meaning, such as *module*, *supplement*, and *semistable*. We visualize that in Figure 2.

Target / Source	Symbol Replacement			
	Conservation		Partial+P	
	MRR	Acc	MRR	Acc
Conservation	**69.26**	**59.59**	27.9	18.29
Partial	61.36	51.72	54.06	42.67
Partial+P	62.1	51.92	**55.92**	**45.23**
Full	53.63	42.08	52.85	41.4
Full+P	56.27	45.13	**55.92**	44.84

Table 6: Controlled cross-replacement levels performance for the SCRATCHBERT-Local-Local model. Both train and test sets are curated from the probability theory domain. +P next to a symbol replacement method means that Protected symbols are not being replaced.

10 Conclusion

We developed a bilinear similarity model and a large dataset (MATCH) for a task focusing on the domain of mathematical research articles. The task consists in matching a proof to a mathematical statement. We proposed two ways to train and inference with our model and dataset: local matching and global matching. We assessed the difficulty of the task with several pre-trained encoders, demonstrating the importance of the vocabulary support for these models. Further assessment relies on using a symbol replacement procedure, which helps test the type of mathematical reasoning the encoders can perform. While our model performs well on this task, we observe through the symbol replacement procedure that the model makes a relatively shallow use of the text and formulae to obtain this performance.

Limitations

Our work has three main limitations. First, we aim to simulate a setup where the same author did not write both a theorem and its corresponding proof. We reduce the intersection size of symbols between the statement and the proof, which leads to more challenging setups. In practice, authors and mathematical communities within fields differ in their use of notation and their writing style (creating *mathematical language dialects*). Such overall dialect

---

[14] We relied on Wikipedia, `https://tinyurl.com/2c3kwsfx`, for creating the protected symbols list.

cannot be altered using simple rule-based methods. We leave it for future work to explore a full MIR setup for our task that takes this into consideration.

Second, due to computational limitations, we could not explore the full potential of our global training method. Our GPUs cannot handle large batch sizes for large models such as MATHBERT and SCRATCHBERT. We use NVIDIA V100 GPUs that allow us to experiment with a batch size of 16 for MATHBERT and SCRATCHBERT, compared to 60 with NPT.

Third, while our symbol replacement method provides a coarse way to test the language model use of the symbols and text in mathematical articles, it presents cases in which the replacement is not precise. These cases arise because the use of symbols in mathematical language is rich and context-dependent (for example, while π often refers to the pie constant, it might also refer to a tuple-projection function or a permutation). We partially address that in §9.4.

Acknowledgments

We thank the reviewers for their helpful comments, and Marcio Fonseca for feedback on an earlier draft. We also thank Richard Zanibbi for a discussion about aspects of this work. The experiments in this paper were supported by a compute grant from the Edinburgh Parallel Computing Center (Cirrus) and another compute grant for the Baskerville service at the University of Birmingham.

References

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Maximin Coavoux and Shay B Cohen. 2021. Learning to match mathematical statements with proofs. *arXiv preprint arXiv:2102.02110*.

Pankaj Dadure, Partha Pakray, and Sivaji Bandyopadhyay. 2021. Embedding and generalization of formula with context in the retrieval of mathematical information. *Journal of King Saud University - Computer and Information Sciences*.

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

Deborah Ferreira and André Freitas. 2020. Premise selection in natural language mathematical texts. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7365–7374, Online. Association for Computational Linguistics.

Deborah Ferreira and André Freitas. 2021. STAR: Cross-modal [STA]tement [R]epresentation for selecting relevant mathematical premises. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3234–3243, Online. Association for Computational Linguistics.

Deborah Ferreira, Mokanarangan Thayaparan, Marco Valentino, Julia Rozanova, and Andre Freitas. 2022. To be or not to be an integer? encoding variables for mathematical text. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 938–948, Dublin, Ireland. Association for Computational Linguistics.

Andrew V. Goldberg and Robert Kennedy. 1995. An efficient cost scaling algorithm for the assignment problem. *Mathematical Programming*, 71(2):153–177.

Mihai Grigore, Magdalena Wolska, and Michael Kohlhase. 2009. Towards context-based disambiguation of mathematical expressions. In *The joint conference of ASCM 2009 and MACIS 2009. 9th international conference on Asian symposium on computer mathematics and 3rd international conference on mathematical aspects of computer and information sciences, Fukuoka, Japan, December 14–17, 2009. Selected papers.*, pages 262–271. Fukuoka: Kyushu University, Faculty of Mathematics.

Michael Gutmann and Aapo Hyvärinen. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13:307–361.

Jesse Michael Han, Jason Rute, Yuhuai Wu, Edward W. Ayers, and Stanislas Polu. 2021. Proof artifact co-training for theorem proving with language models. *ArXiv preprint*, abs/2102.06203.

R. Jonker and A. Volgenant. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340.

Giovanni Yoko Kristianto, Minh quoc Nghiem, Yuichiroh Matsubayashi, and Akiko Aizawa. 2012. Extracting definitions of mathematical expressions in scientific papers. In *In JSAI*.

Harold W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.

Martin Líška, Petr Sojka, Michal Růžička, and Petr Mravec. 2011. Web Interface and Collection for Mathematical Retrieval: WebMIaS and MREC. In *Towards a Digital Mathematics Library.*, pages 77–84, Bertinoro, Italy. Masaryk University.

Behrooz Mansouri, Shaurya Rohatgi, Douglas Oard, Jian Wu, C.Lee Giles, and Richard Zanibbi. 2019. Tangent-cft: An embedding model for mathematical formulas. In *ICTIR '19: Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*.

C. o. P. a. L. o. t. Mathematical Sciences. 2014. Developing a 21st Century Global Library for Mathematics Research. *ArXiv e-prints*, abs/1404.1905.

Minh Nghiem Quoc, Keisuke Yokoi, Yuichiroh Matsubayashi, and Akiko Aizawa. 2010. Mining coreference relations between formulas and text using Wikipedia. In *Proceedings of the Second Workshop on NLP Challenges in the Information Explosion Era (NLPIX 2010)*, pages 69–74, Beijing, China. Coling 2010 Organizing Committee.

Robert Pagel and Moritz Schubotz. 2014. Mathematical language processing project. *CoRR*, abs/1407.0167.

Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. 2021. Mathbert: A pre-trained model for mathematical formula understanding. *ArXiv preprint*, abs/2105.00377.

Bartosz Piotrowski and Josef Urban. 2019. Guiding theorem proving by recurrent neural networks. *ArXiv preprint*, abs/1905.07961.

Stanislas Polu and Ilya Sutskever. 2020. Generative language modeling for automated theorem proving. *ArXiv preprint*, abs/2009.03393.

B. T. Polyak and A. B. Juditsky. 1992. Acceleration of stochastic approximation by averaging. *SIAM J. Control Optim.*, 30(4):838–855.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. ACM.

Ulf Schöneberg and Wolfram Sperber. 2014. POS tagging and its applications for mathematics. *CoRR*, abs/1406.2880.

Moritz Schubotz, Alexey Grigorev, Marcus Leich, Howard S. Cohl, Norman Meuschke, Bela Gipp, Abdou S. Youssef, and Volker Markl. 2016. Semantification of identifiers in mathematics for better math information retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, pages 135–144. ACM.

Shun Shao, Yftah Ziser, and Shay B. Cohen. 2023. Erasure of unaligned attributes from neural representations. *arXiv preprint arXiv:2302.02997*.

Jia Tracy Shen, Michiharu Yamashita, Ethan Prihar, Neil T. Heffernan, Xintao Wu, and Dongwon Lee. 2021. Mathbert: A pre-trained language model for general NLP tasks in mathematics education. *ArXiv preprint*, abs/2106.07340.

Heinrich Stamerjohanns, Michael Kohlhase, Deyan Ginev, Catalin David, and Bruce Miller. 2010. Transforming large collections of scientific publications to xml. *Mathematics in Computer Science*, 3(3):299–307.

Yiannos Stathopoulos, Simon Baker, Marek Rei, and Simone Teufel. 2018. Variable typing: Assigning meaning to variables in mathematical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 303–312, New Orleans, Louisiana. Association for Computational Linguistics.

Yiannos Stathopoulos and Simone Teufel. 2015. Retrieval of research-level mathematical information needs: A test collection and technical terminology experiment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 334–340, Beijing, China. Association for Computational Linguistics.

Yiannos Stathopoulos and Simone Teufel. 2016. Mathematical information retrieval based on type embeddings and query expansion. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2344–2355, Osaka, Japan. The COLING 2016 Organizing Committee.

A. Volgenant. 1996. Linear and semi-assignment problems: A core oriented approach. *Computers & Operations Research*, 23(10):917 – 932.

Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. 2021. Naturalproofs: Mathematical theorem proving in natural language.

Magdalena Wolska, Mihai Grigore, and Michael Kohlhase. 2011. Using discourse context to interpret object-denoting mathematical expressions. In *DML 2011 - Towards a Digital Mathematics Library, Proceedings*.

Richard Zanibbi, Akiko Aizawa, Michael Kohlhase, Iadh Ounis, Goran Topic, and Kenny Davila. 2016. NTCIR-12 mathir task overview. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, National Center of Sciences, Tokyo, Japan, June 7-10, 2016*.

A Details on Dataset Construction

As mentioned in §4, we use the MREC corpus to extract statement-proof pairs.

Statement-proof Identification For each XML article (corresponding to a single arXiv article), we extract pairs of consecutive `<div>` tags such that: (i) the `class` attribute of the first `div` node contains the string `"theorem"`; (ii) the `class` attribute of the second `div` node is the string `"proof"`. Articles that do not contain such pairs of tags are discarded, as well as articles that are not written in English (representing 143 articles in French, 11 in Russian, 5 in German, 2 in Portuguese and 1 in Ukrainian), as identified by the `polyglot` Python package.[15]

In the remaining collection of pairs of statements and proofs, we filter out pairs for which either the statement or the proof is too short.[16] Indeed, the short texts were often empty (only consisting of a title, e.g. "5.26 Lemma."), which we attribute to the noise inherent to the conversion to XML, or not self-contained. In particular, we identified several prototypical cases:

- Omitted (or easy) proofs contain usually a single word ('omitted', 'straightforward', 'well-known', 'trivial', 'evident'), but are sometimes more verbose ('This is obvious and will be left to the readers').
- Proofs that consist of a single reference to
    - An appendix ('See Appendix A');
    - Another theorem ('This follows immediately from Proposition 4.4 (ii).');
    - The proof method of another theorem ('Similar to proof of Lemma 6.1')
    - Another article ('See [BK3, Theorem 4.8].');
    - Another part of the article ('The proof will appear elsewhere.', 'See above.', 'Will be given in section 5.').

Filtering on the number of tokens also excludes self-contained short proofs, such as 'Take $Q' = ph_i - p_i$.' However, such proofs were very infrequent on manual inspection of the discarded pairs (2 in a manually inspected random sample of 100 discarded proofs).

Preprocessing: Linearizing Equations Mathematical formulae in the XML articles are enclosed in a `<math>` markup tag, that materializes the switch to the MathML format, and whose internal structure represents the formula as an XML tree. As a preprocessing step, we linearize each formula to a raw sequence of strings.

In MathML, an equation can be encoded in a content-based (semantic) way or in a presentational way, using different sets of markup tags. We first convert all MathML trees to presentational MathML using the XSL stylesheet from the Content MathML Polyfill repository.[17] Then we perform a depth-first search on each tree rooted in a `<math>` tag to extract the text content of the whole tree.

During this preprocessing, we tested several processing choices:

- **Font information**. In mathematical discourses, fonts play an important role. Their semantics depend on conventions shared by researchers. If both x and $\mathbf{x}$ appear in the same article, they are most likely to represent different mathematical objects, e.g. a scalar and a vector. Therefore, we use distinct symbols for tokens that are in distinct fonts.
- **Math-English ambiguity**. Some symbols can be used both in natural language text and in formulae. For example, 'a' can be a determiner in English, or a variable name in a formula. To avoid increasing ambiguity when linearizing formula, we type each symbol (as math or text) to make the mathematical vocabulary completely disjoint from the text vocabulary.

Both these preprocessing steps had a beneficial effect on the baselines in preliminary experiments.

B Distribution of Proof-Statement Assignments

Table 7 depicts the cumulative distribution of proofs and the number of statements they are assigned to.

C Symbol Replacement Details

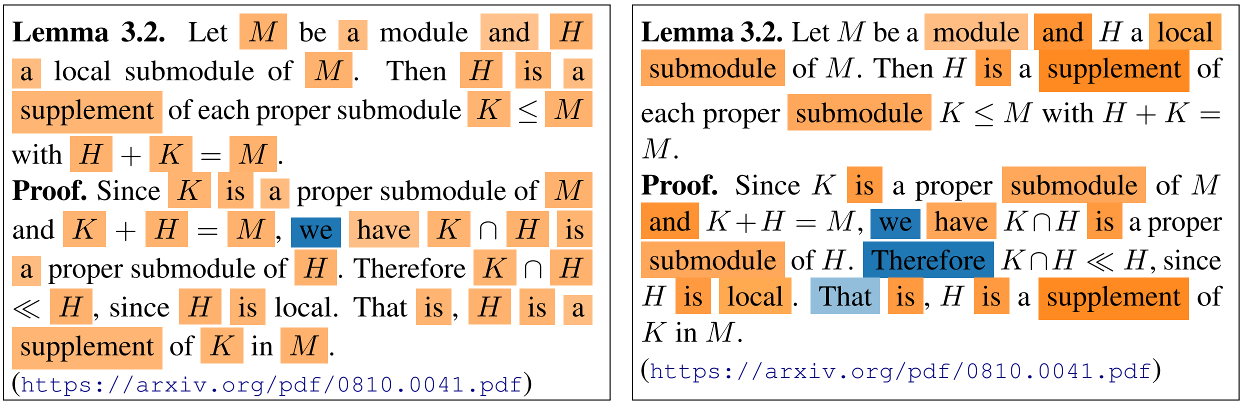
We follow the following rules when replacing symbols:

---

[15] www.github.com/aboSamoor/polyglot/

[16] We used a minimum length of 20 tokens for both statements and proofs, based on a manual inspection of the shortest examples. We also exclude proofs and statements longer than 500 tokens.

[17] https://github.com/fred-wang/webextension-content-mathml-polyfill

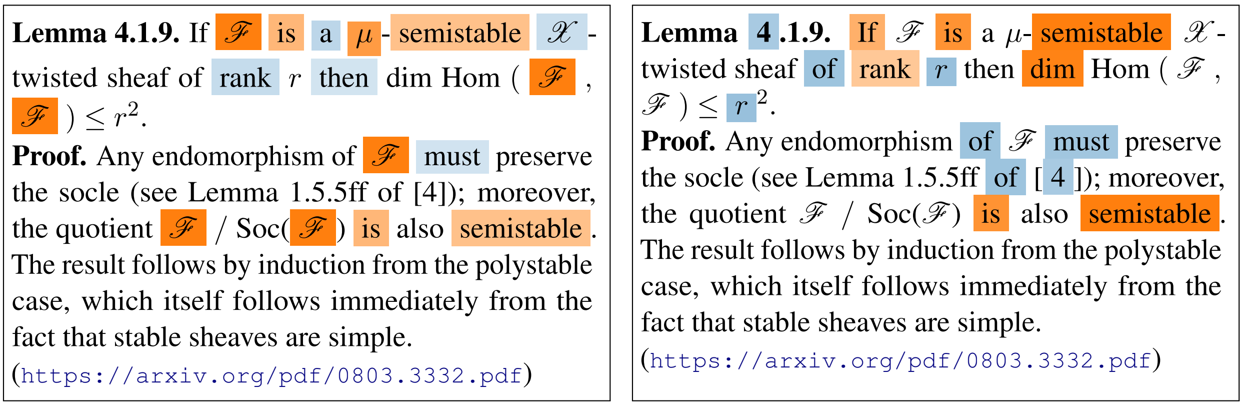


Lemma 3.2. Let M be a module and H a local submodule of M. Then H is a supplement of each proper submodule $K \leq M$ with $H + K = M$.
Proof. Since K is a proper submodule of M and $K + H = M$, we have $K \cap H$ is a proper submodule of H. Therefore $K \cap H \ll H$, since H is local. That is, H is a supplement of K in M.
(`https://arxiv.org/pdf/0810.0041.pdf`)

Lemma 3.2. Let M be a module and H a local submodule of M. Then H is a supplement of each proper submodule $K \leq M$ with $H + K = M$.
Proof. Since K is a proper submodule of M and $K + H = M$, we have $K \cap H$ is a proper submodule of H. Therefore $K \cap H \ll H$, since H is local. That is, H is a supplement of K in M.
(`https://arxiv.org/pdf/0810.0041.pdf`)

(a) Example statement/proof 1 - Symbol conservation

(b) Example statement/proof 1 - Full symbol replacement



Lemma 4.1.9. If $\mathscr{F}$ is a μ-semistable $\mathscr{X}$-twisted sheaf of rank r then dim Hom($\mathscr{F}$, $\mathscr{F}$) $\leq r^2$.
Proof. Any endomorphism of $\mathscr{F}$ must preserve the socle (see Lemma 1.5.5ff of [4]); moreover, the quotient $\mathscr{F}$ / Soc($\mathscr{F}$) is also semistable. The result follows by induction from the polystable case, which itself follows immediately from the fact that stable sheaves are simple.
(`https://arxiv.org/pdf/0803.3332.pdf`)

Lemma 4.1.9. If $\mathscr{F}$ is a μ-semistable $\mathscr{X}$-twisted sheaf of rank r then dim Hom($\mathscr{F}$, $\mathscr{F}$) $\leq r^2$.
Proof. Any endomorphism of $\mathscr{F}$ must preserve the socle (see Lemma 1.5.5ff of [4]); moreover, the quotient $\mathscr{F}$ / Soc($\mathscr{F}$) is also semistable. The result follows by induction from the polystable case, which itself follows immediately from the fact that stable sheaves are simple.
(`https://arxiv.org/pdf/0803.3332.pdf`)

(c) Example statement/proof 2 - Symbol conservation

(d) Example statement/proof 2 - Full symbol replacement

Figure 2: LIME visualizations for the model that was trained in the symbol conservation setup (a and c) and their corresponding LIME visualizations for the model that was trained in the full symbol replacement setup (b and d). The LIME "match" class supporting features are colored in orange, and the "mismatch" is in blue. The darker the color, the higher (in absolute value) the feature importance.

Statements	Proofs	%
≥ 20	7	0.0
≥ 10	80	0.2
≥ 5	1027	1.9
≥ 2	11949	22.6
$= 1$	19531	37.0
< 1	21275	40.3

Table 7: Cumulative distribution of proofs in the development set, by number of statements to which they are assigned with the local decoding method.

- Only the proof symbols are being replaced as there is no need to replace both statement and proof symbols.
- We replace symbols only if they appear in both the statement and the proof.
- If the symbol a, for example, is mapped to b, we will map A to B, and vice versa.
- We do not replace the double-struck letters, e.g., $\mathbb{R}$, since they usually represent fields and constant. We do not replace standard constant symbols such as π.

D Prediction Visualization Examples

We provide a LIME visualization of several mathematical statements in Figure 2.

Lessons Learned from a Citizen Science Project for Natural Language Processing

Jan-Christoph Klie[1] **Ji-Ung Lee**[1] **Kevin Stowe**[1,2] **Gözde Gül Şahin**[1,3]
Nafise Sadat Moosavi[1,4] **Luke Bates**[1] **Dominic Petrak**[1]
Richard Eckart de Castilho[1] **Iryna Gurevych**[1]

[1]Ubiquitous Knowledge Processing Lab (UKP Lab)
Department of Computer Science and Hessian Center for AI (hessian.AI)
Technical University of Darmstadt
www.ukp.tu-darmstadt.de

[2]Educational Testing Service
[3]KUIS AI, Koç University
[4]Department of Computer Science, The University of Sheffield

Abstract

Many Natural Language Processing (NLP) systems use annotated corpora for training and evaluation. However, labeled data is often costly to obtain and scaling annotation projects is difficult, which is why annotation tasks are often outsourced to paid crowdworkers. Citizen Science is an alternative to crowdsourcing that is relatively unexplored in the context of NLP. To investigate whether and how well Citizen Science can be applied in this setting, we conduct an exploratory study into engaging different groups of volunteers in Citizen Science for NLP by re-annotating parts of a pre-existing crowdsourced dataset. Our results show that this can yield high-quality annotations and attract motivated volunteers, but also requires considering factors such as scalability, participation over time, and legal and ethical issues. We summarize lessons learned in the form of guidelines and provide our code and data to aid future work on Citizen Science.[1]

1 Introduction

Data labeling or *annotation* is often a difficult, time-consuming, and therefore expensive task. Annotations are typically drawn from domain experts or are crowdsourced. While experts can produce high-quality annotated data, they are expensive and do not scale well due to their relatively low number (Sorokin and Forsyth, 2008). In contrast, crowdsourcing can be relatively cheap, fast, and scalable, but is potentially less suited for more complicated annotation tasks (Drutsa et al., 2020). Another approach is using Citizen Science, which describes the participation and collaboration of volunteers from the general public with researchers to conduct science (Haklay et al., 2021). Over the past decade, Citizen Science platforms, which rely on unpaid volunteers to solve scientific problems, have been used for a wide variety of tasks requiring human annotation (Hand, 2010), e.g., classifying images of galaxies (Lintott et al., 2008) or for weather observation (Leeper et al., 2015).

While Citizen Science has been shown to produce high-quality annotations in ecological or environmental projects (Kosmala et al., 2016), its potential has so far not been investigated in depth for Natural Language Processing (NLP). Our goal in this work is to assess the practicality of undertaking annotation campaigns for NLP via Citizen Science. We analyze whether volunteers actually react to our calls and participate, how the resulting quality is compared to crowdsourcing, what the benefits and shortcomings are and what needs to be taken into account when conducting such a project. We especially are interested in differences between annotators recruited via different channels, which we investigate by advertising to different social media platforms, NLP-related mailing lists, and university courses. To explore this possibility, we use the PERSPECTRUM dataset (Chen et al., 2019, CC-BY-SA) that focuses on the task of stance detection and can be motivated by fighting misinformation and promoting accurate debate in internet discussions. We replicated a portion of the annotations in this dataset using citizen scientists instead of crowdworkers. To accomplish this goal, we designed an annotation workflow that is suitable for Citizen Science and allows us to recruit volunteers across a variety of platforms.

---

[1] https://github.com/UKPLab/eacl2023-citizen-science-lessons-learned

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3594–3608
May 2-6, 2023 ©2023 Association for Computational Linguistics

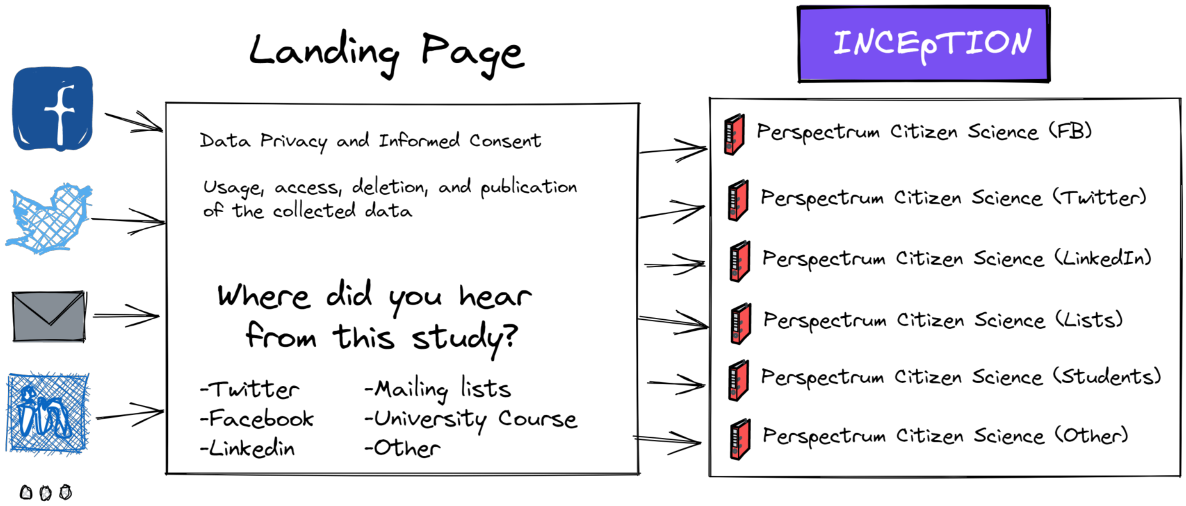


Figure 1: We advertised our project via various social media, mailing lists and university courses. Volunteers then are onboarded via the landing page and donated annotations via INCEpTION.

Our contributions are the following:

1. We provide a systematic study on Citizen Science across different channels and analyze turnout and quality. For this, we re-annotate parts of the PERSPECTRUM dataset using Citizen Science and compare these to the original, crowdsourced annotations.
2. We provide guidelines and recommendations on how to successfully conduct a Citizen Science project for NLP annotation and discuss critical legal and ethical aspects.
3. We provide a platform for future Citizen Science projects that handles onboarding, anonymous access, work assignment and the annotating itself.

Our results show that using Citizen Science for linguistic annotation can result in high-quality annotations, but that attracting and motivating people is critical for its success, especially in the long-term. We were able to attract 98 volunteers when conducting our Citizen Science project which resulted in 1,481 annotations over 2 months, thereby re-annotating around 10% of the original dataset. We find that annotations obtained through mailing lists and university students were of high quality when comparing them to the original, adjudicated crowdsourced data. We thus conclude that Citizen Science projects have the potential to be applied to NLP annotation if they are conceptualized well, but are best suited for creating smaller datasets.

2 Background

Prior work has developed various means and strategies for annotating large datasets. So far, annotation studies in NLP mainly use domain-experts or crowdworkers, or a mix of both (Nguyen et al., 2015). Crowdsourcing in particular has received increasing attention over the past decade (Wang et al., 2013).

Paid Experts Recruiting domain experts (e.g., linguists) for annotation studies has been a widely accepted method to generate linguistically annotated corpora. Famous examples are the Brown Corpus (Francis and Kucera, 1979) or the Penn Treebank (Marcus et al., 1993). While the resulting datasets are of the highest quality, domain experts are often few, and such annotation studies tend to be slow and expensive (Sorokin and Forsyth, 2008). Although many researchers moved on to annotation studies that recruit crowdworkers, expert annotations are still necessary in various fields, e.g., biomedical annotations (Hobbs et al., 2021).

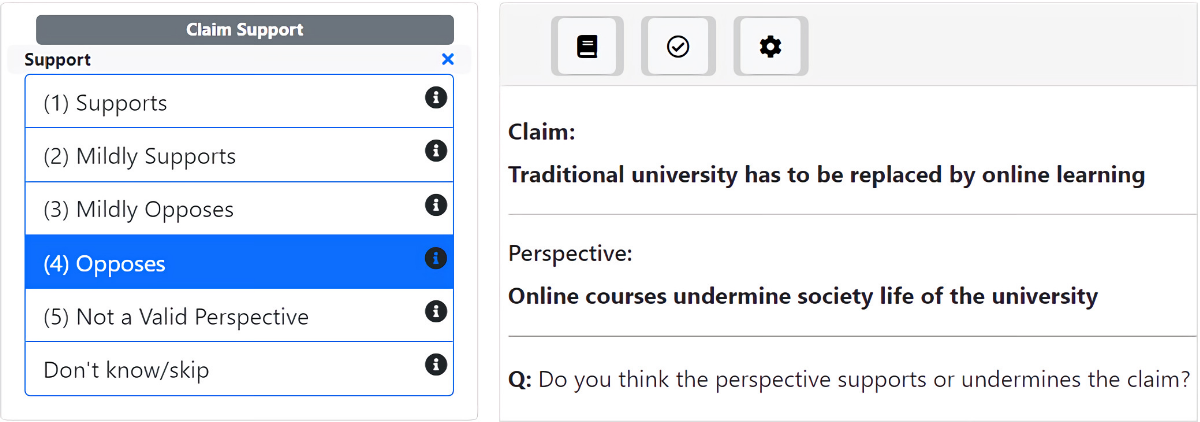
Crowdsourcing To accelerate the annotation process and reduce costs, researchers have utilized crowdsourcing as a means to annotate large corpora (Snow et al., 2008). The main idea behind crowdsourcing is that annotation tasks that do not require expert knowledge can be assigned to a large group of paid non-expert annotators. This is commonly done via crowdsourcing platforms such as Amazon Mechanical Turk (AMT) or Upwork and has been successfully used to annotate various datasets across different tasks and domains (Derczynski et al., 2016; Habernal and Gurevych, 2017). Previous work compared the quality between crowdsourcing and expert annotations, showing that many tasks can be given to crowdworkers without major impact on the quality of annotation (Snow et al., 2008; Hovy et al., 2014; De Kuthy et al., 2016).

Although crowdworkers can substantially accelerate annotation, crowdsourcing requires careful task design and is not always guaranteed to result in high quality data (Daniel et al., 2018). Moreover, as annotators are compensated not by the time they spend but rather by the number of annotated instances, they are compelled to work fast to maximize their monetary gain—which can negatively affect annotation quality (Drutsa et al., 2020) or even result in spamming (Hovy et al., 2013). It can also be difficult to find crowdworkers for the task at hand, for instance due to small worker pools for languages other than English (Pavlick et al., 2014; Frommherz and Zarcone, 2021) or because the task requires special qualifications (Tauchmann et al., 2020). Finally, the deployment of crowdsourcing remains ethically questionable due to undervalued payment (Fort et al., 2011; Cohen et al., 2016), privacy breaches, or even psychological harm on crowdworkers (Shmueli et al., 2021).

Games with a Purpose A related but different way to collect annotations from volunteers is *games with a purpose*, i.e., devising a game in which participants annotate data (Chamberlain et al., 2008; Venhuizen et al., 2013). Works propose games for different purposes and languages. For instance, anaphora annotation (PhraseDetectives, Poesio et al. 2013), dependency syntax annotation (Zombilingo, Fort et al. 2014), or collecting idioms (Eryiğit et al., 2022). It has been shown that if a task lends itself to being gamified, then it can attract a wide audience of participants and can be used to create large-scale datasets (von Ahn, 2006). Finally, Lyding et al. (2022) investigate games with a purpose in the context of (second) language learning to simultaneously crowdsource annotaions from learners as well as teachers. One such example is Substituto, a turn-based, teacher-moderated game for learning verb-particle constructions (Araneta et al., 2020). We do not consider gamification in this work, as enriching tasks with game-like elements requires considerable effort and cannot be applied to every task.

Citizen Science Citizen Science broadly describes participation and collaboration of the general public (the citizens) with researchers to conduct science (Haklay et al., 2021). Citizen Science is a popular alternative approach for dataset collection efforts, and has been successfully applied in cases of weather observation (Leeper et al., 2015), counting butterflies (Holmes, 1991) or birds (National Audubon Society, 2020), classifying images of galaxies (Lintott et al., 2008) or monitoring water quality (Addy et al., 2010). Newly-emerging technologies and platforms further allow researchers to conduct increasingly innovative Citizen Science projects, such as the prediction of influenza-like outbreaks (Lee et al., 2021) or the classification of animals from the Serengeti National Park (Swanson et al., 2015). *LanguageARC* is a Citizen Science platform for developing language resources (Fiumara et al., 2020). It is however not open yet to the public to create projects and does not easily allow conducting a Citizen Science meta-study as we do in this work. One work using LanguageARC is by Fort et al. (2022) (LD) who collected resources to evaluate bias in language models. They did not investigate the impact of using different recruitment channels which we do. Other projects using LanguageARC are still running and it is too early to derive recommendations from.

Compared to crowdsourcing, Citizen Science participants are volunteers that do not work for monetary gain. Instead, they are often motivated intrinsically. For instance, they may have a personal interest on positively impacting the environment (West et al., 2021), or in altruism (Rotman et al., 2012). Asking for unpaid work also entails various issues like finding good ways of how to attract volunteers, and ethical considerations (Resnik et al., 2015; Rasmussen and Cooper, 2019) that need to be addressed (cf. §5). Intrinsic motivation also has the potential of resulting in higher-quality annotations compared to crowdsourcing. For instance, Lee et al. (2022) find in their evaluation study with citizen scientists that their participants may have been willing to take more time annotating for the sake of higher annotation accuracy. However, as their main goal was to conduct an evaluation study for their specific setup, this finding cannot be generalized to other Citizen Science scenarios. So far, only Tsueng et al. (2016) provide a direct comparison between crowdsourcing and Cit-



Claim:

Traditional university has to be replaced by online learning

Perspective:

Online courses undermine society life of the university

Q: Do you think the perspective supports or undermines the claim?

Figure 2: Assigning a label to an instance in the INCEpTION text annotation platform.

izen Science and show that volunteers can achieve similar performance in mining medical entities in scientific texts. They recruit participants through different channels such as newspapers, Twitter, etc., but do not compute channel-specific performance, making it difficult to assess whether the quality of the resulting annotation depends on the recurrent channel. In contrast, in the present work, we explicitly consider the recruitment channel in our evaluation and furthermore provide a discussion and guidelines for future Citizen Science practitioners. Also, it attracts intrinsically (not only fiscally) motivated volunteers that are often skilled in the task and can provide high-quality annotations, thus potentially combining the advantages of expert annotations and crowdsourcing. Relying on unpaid annotators entails several issues, including attracting volunteers and ethical considerations (Resnik et al., 2015; Rasmussen and Cooper, 2019) that need to be taken into account (see §5).

3 Study Design

To study the feasibility of Citizen Science for NLP annotation, we asked volunteers recruited via various channels to re-annotate an existing, crowdsourced dataset. The general setup is described in Fig. 1. To conduct a systematic study, we identified the following four necessary steps: 1) Identifying a suitable dataset (§3.1); 2) Selecting suitable recruitment channels to advertise our project on (§3.2); 3) Building a landing page for onboarding participants that asks for informed consent and the channel from which they originated (§3.3); 4) Setting up the annotation editor to which participants are forwarded after the onboarding (§3.4).

3.1 Dataset selection

We first conducted a literature review of relevant crowdsourced NLP datasets to identify the ones that could be accurately reproduced via Citizen Science. We assessed datasets for the following two criteria: 1) **Availability**: the dataset must be publicly available to make proper comparisons in terms of annotator agreement; 2) **Reproducibility**: the annotation setup including annotation guidelines needs to be reproducible to ensure similar conditions between citizen scientists and crowdworkers. We focused on datasets that are targeted towards contributing to social good to encourage volunteers to participate. Unfortunately, many inspected datasets did not fulfill both of these requirements. Overall, we identified two main issues while screening over 20 candidate datasets. First, many datasets used Tweets which impacted reproducibility as Twitter only allows researchers to publish the tweet identifiers. This leads to irrecoverable instances when tweets were deleted. Second was the lack of precise guidelines. For instance, many considered datasets about societal biases lack explicit descriptions of what is considered a stereotype. As such biases are often also impacted by the respective cultural background of annotators, they are difficult to reproduce without specific guidelines.

In the end, we decided on the stance detection task of the PERSPECTRUM dataset (Chen et al., 2019). The task provides clear instructions, publicly available data, and is motivated by social good (fighting misinformation/promoting accurate debate in internet discussions). Each instance consists of a claim–perspective pair (cf. Fig. 2) and annotators are asked if the claim *supports*, *opposes*, *mildly-supports*, *mildly-opposes*, or is *not a valid*

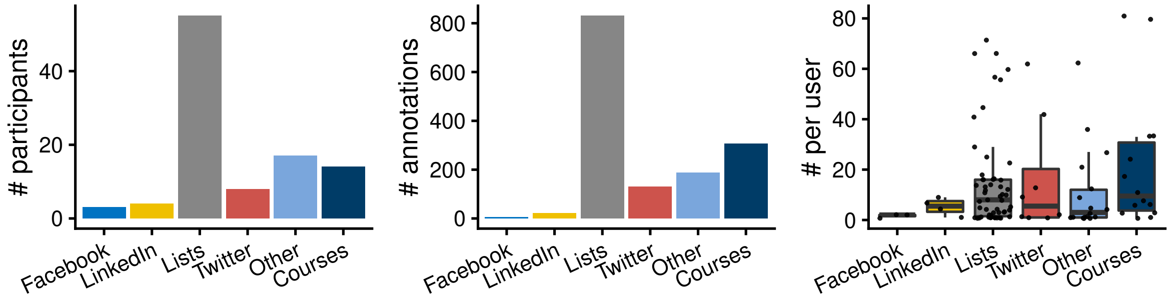


Figure 3: Participants, annotations and annotations grouped by the channel via they were recruited. It can be seen that overall, most participants and annotations were contributed by annotators recruited via mailing lists. Annotators from mailing lists and courses yielded the volunteers who contributed the most individually.

perspective. Following the original work, we also evaluated the annotations on a coarser tagset that only contains the categories for *support*, *oppose* and *not a valid perspective*. Overall, the dataset consists of 907 claims and $8,370$ different perspectives which yield $11,805$ annotated instances. In preliminary studies, we received further feedback that forcing annotators to provide an explicit label for each instance could lead to increasing frustration, especially for ambiguous or complicated instances. To lessen the burden for our voluntary annotators and keep them motivated in the annotation task, we allowed them to skip instances (*Don't know/skip*) which was not present in the original annotation editor for PERSPECTRUM.

3.2 Recruitment channels

To recruit annotators, we advertised our project on three social media platforms, namely, Twitter, LinkedIn and Facebook. Unfortunately, after creating the Facebook organization and advertising the project, the account was banned due to "violating their community standards" and has so far remained banned. One of our team members then promoted our annotation study on their personal Facebook to attract participation from this social media platform. In addition, the team members advertised the work on Twitter and in relevant LinkedIn groups such as COMPUTATIONAL LINGUISTICS and MACHINE LEARNING AND DATA SCIENCE.

We further promoted the study via two external mailing lists (i.e., CORPORA-LIST, ML-NEWS). Late in the project, we received interest from other faculty to advertise the task in their courses—an offer that we gladly accepted. For this, participation was completely voluntary and anonymous, students' grades were not affected by participation, and authors were not among the instructors. To evaluate different recruitment channels separately, we asked participants on the landing page to answer the question: "Where did you hear from this study?". We also allowed volunteers to not disclose how they found out about the study, this is referred to as "Other" or "Undisclosed" in this paper. Final participation counts are given in Fig. 3. We deliberately limited our outreach, e.g. we did not use university social media accounts or colleagues with large follower bases. Also, we made sure to not exhaust channels by posting too many calls for participation.

3.3 Landing page

We implemented a customizable landing page web application catering to the needs of Citizen Science projects. The link to such a landing page was shared via the respective recruitment channels. The landing page contained information about the study itself, its purpose, its organizers, which data we collected, and its intended use. This landing page toolbox is designed so that it can easily be adapted to future Citizen Science projects. To allow project creators to use an annotation editor of their choice, we designed the toolbox to act as an intermediary that collects a participant's consent for the actual annotation study. This ensures that only participants that have been properly informed and have explicitly provided their consent are given access to the study. For future Citizen Science projects, the tool further assists organizers through the landing page creation process to foster an ethical collection of data by asking several questions, that are listed in the appendix.

3.4 Annotation editor

INCEpTION (Klie et al., 2018) offers a configurable, web-based platform for annotating text documents at span, relation and document levels. To make it usable in Citizen Science scenarios, we extended the platform with three features, namely, (1) the ability to join a project through a link, (2) support for anonymous guest annotators, and (3) a dynamic workload manager. Allowing citizen scientists to participate in the project anonymously as guests without any sign-up process substantially reduced the entry barrier and made it easier for us to satisfy data protection policies. The same is true for the ability of joining a project through an invite link. Upon opening the link, annotators were greeted with the annotation guidelines and were directly able to start annotating. Finally, we implemented a dynamic workload manager that takes as input the desired number of annotators per document and then automatically forwards annotators directly to the document instances requiring annotation. Upon finishing annotating an instance, INCEpTION was configured to automatically load and display the next instance for annotation, similar to popular crowdsourcing platforms. We also included rules for handling other issues that may occur with voluntary annotations such as recovering instances that annotators have started to work on but then abandoned. Additionally, we modified the existing user interface to improve the annotation workflow. This mainly included implementing a dedicated labeling interface that allows users to select a single label for an instance via a radio button group. Annotation of an instance thus required two user actions: first, selecting the document label, and second, confirming the annotation, thereby moving on to the next document.

4 Results

We conducted our study between January and March 2022 and promoted the task in successive rounds across all recruitment channels. In total, we were able to recruit 98 participants who provided 1481 annotations resulting in 906 fully annotated instances. Each instance with at least one annotation has received on average 1.63 annotations. Detailed statistics are provided in the appendix.

Participation To identify promising channels for future Citizen Science studies, we report the number of annotators per channel, the total number of annotations per channel and per user (cf. Fig. 3). Overall, we find that the most effective channel for public outreach are mailing lists (55 participants). Asking students in university courses to participate was the second most effective with 14 participants. Facebook, LinkedIn, and Twitter only yielded three, four, and eight participants respectively. We further find a highly skewed distribution of annotations per user, as many annotators only provide a few annotations while a few annotators provide many annotations. For instance, the most active annotators were two students who provided ∼80 annotations as well as six participants from mailing lists who provided ∼60–80 annotations each. For Twitter and "undisclosed", only a single annotator made over 60 annotations. We also find that on average, participants from university courses provided the most annotations per person. When looking at participation over time (see Fig. 5), we observe increased activity in annotations made after the call for participation has been posted to the respective channel. For many channels, the count quickly flattens. Interestingly, Twitter sees a second spike long after the post was made. We attribute it to people sharing the post in our community quite a while after the initial release. We did not track whether individual volunteers came back for another round of annotations after their initial participation.

Coverage Overall, our 98 volunteers have provided 1,481 annotations to 906 unique instances (approximately 8% of the original dataset) over two months. This is comparable to other Citizen Science projects like Fort et al. (2022), which had 102 participants in total. They annotated three tasks and collected 2347, 2904 and 220 submissions over eight months. Table 1 shows the resulting coverage of our Citizen Science annotation study. While this still leaves room for improvement, the number of annotations collected nonetheless shows that Citizen Science can be viable in real life settings and is a promising direction to investigate in further studies, especially for creating focused and smaller-scale resources.

Quality In terms of annotation quality, we find that most channels yield annotations that highly agree with the gold labels (cf. Table 2), even though our annotations are not adjudicated yet. We further find that volunteers from university courses and mailing list show the highest accuracy, followed by Twitter and "undisclosed". Only LinkedIn yields

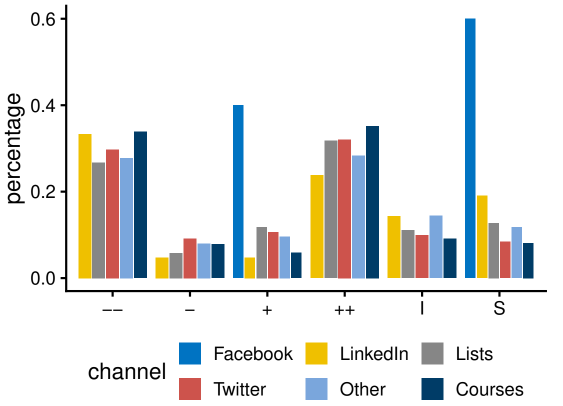


Figure 4: Label distribution grouped by channel. Labels are *supports* (++), *mildly-supports* (+), *mildly-opposes* (-), *opposes* (--), *not a valid perspective* (I) and *Skip* (S).

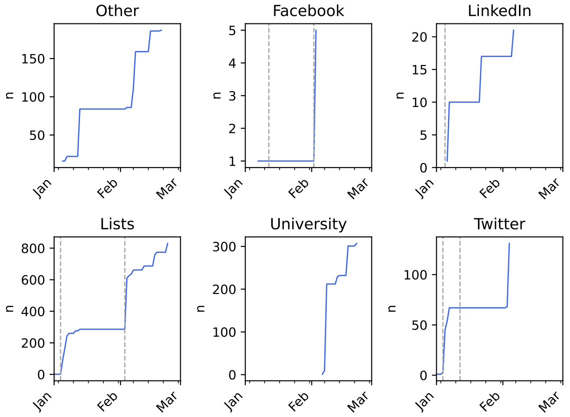


Figure 5: Annotations made over time. Vertical lines indicate when calls on the respective channels have been posted.

lower accuracy than 70% on the coarse label set.

For the majority of channels (with the exception of Facebook and LinkedIn), we only see a skip percentage of ∼10% (cf. Fig. 4). This indicates our volunteers are actually willing to spend time and effort to solve the task at hand, as adding a "Don't know/skip" option in crowdsourcing usually is an invitation for workers to speed through the tasks and not provide useful annotations. The exception is Facebook, where we find that a majority of the annotations from Facebook were labeled as *I don't know/skip* (3 out of 5). Further analysis of the label distribution grouped by channel (cf. Fig. 4) shows that all channels except for Facebook display a similar distribution in terms of annotated labels. This indicates that we can expect a rather stable annotation performance across citizen scientists recruited from different channels.

5 Discussion and Takeaways

Here we present lessons learned, discuss legal challenges and ethical considerations, as well as provide guidelines for future Citizen Science projects.

Table 1: Claims, claim clusters, and individual claim-perspective pairs that have been annotated at least once. We call the set of a claim and a perspective together with its paraphrases a claim cluster.

Name	# Annotated	# Total	% Annotated
Claims	388	907	42.78
Clusters	739	5092	14.51
Total	906	11805	7.67

Table 2: Annotation accuracy compared to the crowd-sourced and adjudicated data from PERSPECTRUM. The five annotations from Facebook (three of them were skipped) and *Don't know/skip* annotations are omitted.

Channel	Coarse	Fine
University	0.92	0.82
LinkedIn	0.69	0.62
Mailing Lists	0.90	0.82
Undisclosed	0.84	0.75
Twitter	0.85	0.73

Channel-dependent differences Our results clearly differ across recruiting channels. We find that overall, Facebook and LinkedIn have the lowest turnout and accuracy when compared to the gold labels, followed by Twitter. Our assumption for the overall low participation is that our network for these channels was not large enough. Advertising our study to NLP-related and university-internal mailing lists and university courses yielded the highest number of participants who also provided the most and best-quality annotations. Although our results show that students may outperform participants from other channels, we also acknowledge that this may not always be a viable option to recruit citizen scientists. Overall, our findings indicate that it is important to address the respective target groups that may be interested in a specific study. However, we also note that continuously advertising Citizen Science studies to the same channels may have a negative impact, as it can cause participation fatigue and lead to fewer volunteers participating. One possible solution could be the use of LanguageARC (Fiumara et al., 2020) from the LDC and centralize calls for participation.

Motivating volunteers In contrast to crowdsourcing, there is no monetary or other extrinsic motivation that could be used to attract Citizen Science annotators. Thus, annotator motivation is a crucial question for Citizen Science studies. As Fig. 5 shows, citizen scientists can be quickly motivated to participate, but can also quickly lose interest in a given annotation study. This can become an issue with a low number of participants, yet our results also indicate that we were able to find highly-motivated participants (8 out of 98 in our results).

Compared to other groups, university students in particular provided a high amount of quality annotations. Considering the findings by Phillips et al. (2018), who do not find statistical differences in terms of quality between students participating for course credit vs. no extrinsic reward—asking students to participate in such projects as part of their coursework might be another good option, but needs to ensure an ethical data collection. For instance, such an approach has been used to annotate the Georgetown University Multilayer Corpus (Zeldes, 2017). Nonetheless, one remaining question is how to keep participants motivated and participate in several sessions as our results indicate that a vast majority of our volunteers only participated in a single session and that participation quickly stops shortly after a call has been posted to the respective channel.

Finally, we want to emphasize the inclusion of a *Don't know/skip* option for Citizen Science annotators. Whereas in crowdsourcing studies, annotators may exploit such an option to increase their gain (Hovy et al., 2013), from the feedback we got during our pilot study, it is crucial to keep volunteers motivated for Citizen Science. For this work, we did not provide a survey that asks about the motivation, as we thought that this might deter potential participants We however suggest that future studies provide such a survey that is as unintrusive as possible to further analyze why participants take part in the respective annotation project.

Legal challenges One substantial challenge in implementing Citizen Science studies is the potentially wide outreach they can have and, consequently, the varying kinds of data protection regulations they have to oblige. To preempt any potential issues that can arise—especially when data that can be used to identify a person (personal data, e.g. obtained during a survey or login credentials) is involved—we recommend researchers who plan to implement a Citizen Science study consider the most strict regulations that are widely accepted.

For the GDPR (European Parliament, 2016), currently one of the strictest data protection regulations, we recommend researchers to explicitly ask voluntary participants for their informed consent when collecting personal information. This includes informing participants beforehand about (1) the purpose of the data collection, (2) the kind of personal and non-personal data collected, (3) the planned use of the data, (4) any planned anonymization processes for publication, and finally, (5) how participants can request access, change, and deletion of the data. We further recommend assigning one specific contact person for any questions and requests for access, change, or deletion of the data. This may seem like additional work when compared to crowdsourcing, but transparent and open communication is one of the key factors to build trust—which is necessary for voluntary participants to consider such studies and provide high-quality annotations. Finally, participants should be informed and agree to the annotations donated being published under a permissive license.

Ethical and economical considerations Although Citizen Science can substantially reduce annotation costs, we emphasize the importance of considering an ethical deployment that does not compromise the trust of the participants. Moreover, given increasing concerns regarding the ownership and use of collected data (Arrieta-Ibarra et al., 2018), one should grant participants full rights to access, change, delete, and share their own personal data (Jones and Tonetti, 2020). This ensures that participants are not exploited for "free labor"—in contrast to approaches like reCAPTCHA (von Ahn et al., 2008), where humans are asked to solve a task in order to gain access to services. Whereas CAPTCHAs were initially intended to block malicious bots, they are becoming increasingly problematic due to their deployment and use by monopolizing companies which raises ethical concerns (Avanesi and Teurlings, 2022) . It is especially important to take the data itself into consideration; exposing volunteers to toxic, hateful, or otherwise sensitive speech should be avoided if they are not informed about it beforehand.

Recommendations Overall, we derive the following recommendations for future Citizen Science studies. 1) our call for annotations resonated the most with the target group that is likely to benefit the most from contributing to it: NLP researchers coming from mailing lists and university students. Therefore, the target audience should be carefully selected, for instance by identifying topic-specific mailing lists or respective university courses. This further means that the purpose of data collection should be made clear and that the results should be made publicly available. 2) the research question of the study should conform to the respective ethical and legal guidelines of the potential target group which should clearly be communicated to make the project accountable. 3) participation should be easy with clearly formulated annotation guidelines and, moreover, the annotation itself should be thoroughly tested beforehand to ensure that participants do not get frustrated due to design errors or choices. For instance, in our preliminary study, we got the feedback that some instances are frustrating to annotate and hence added an option to skip. 4) analyzing participation over time shows that a Citizen Science project has to be continuously advertised in order to stay relevant and achieve high participation. Otherwise, it will be forgotten quickly. This can be done by sharing status updates or creating preliminary results. Fifth, we recommend asking about user motivation before, during or after the annotation with a survey to better understand the participants and their demographics.

6 Conclusion

In this work, we presented an exploratory annotation study for utilizing Citizen Science for NLP annotation. We developed an onboarding process that can easily be adapted to similar projects and evaluated Citizen Science annotations for re-annotating an existing dataset. Furthermore, we extended the INCEpTION platform, a well-known open-source semantic annotation platform, with a dynamic workload manager and functionality for granting access to external users without registration. This enables its usage for Citizen Science projects. We advertised the study via Twitter, Facebook, LinkedIn, mailing lists, and university courses and found that participants from mailing lists and university courses are especially capable of providing high-quality annotations. We further discuss legal and ethical challenges that need to be addressed when conducting Citizen Science projects and provide general guidelines for conducting future projects that we would like to have known before starting. Overall, we conclude that Citizen Science can be a viable and affordable alternative to crowdsourcing, but is limited by successfully keeping annotators motivated. We will make our code and data publicly available to foster more research on Citizen Science for NLP and other disciplines.

Future Work We see the following directions for further research and evaluation to better understand in which settings Citizen Science can be applicable and how to use it best. Here, we used PERSPECTRUM as the dataset to annotate and mentioned in the participation calls that it benefits the social good. Therefore, it would be interesting to conduct more projects and see which datasets are suitable as well as whether volunteers participate, even if there is no extrinsic motivation. Then, it can also be tested how annotator retention develops, especially when project are running longer. The call for participation itself could also be investigated for the impact it has on turnout, motivation and quality.

7 Limitations

Throughout this article, we analyzed whether Citizen Science applies to linguistic annotation and showed that we can attract volunteers that donate a sizeable number of high-quality annotations. This work, however, comes with limitations that should be taken into account and tackled in future work. First, we based our analysis on a single annotation campaign and dataset that we advertised as being relevant for the social good. Therefore we suggest conducting more such annotation projects, also with different kinds of tasks. Second, we did not perform a user survey that for instance asked for user motivation. This is why we can only speculate about the motivation of our participants and suggest future works to explicitly prepare such a survey. Third, using Facebook as a channel might be viable, but we were not able to properly analyze it, as our account was blocked shortly after creation and never was reinstantiated. Finally, based on participation and annotation numbers, we see Citizen Science as more of an option for annotating smaller datasets, or longer-term projects that are more actively advertised than in our study which took place over two months and for which we deliberately limited the outreach.

Acknowledgments

We thank our anonymous reviewers, Michael Bugert and Max Glockner for their detailed and helpful comments to improve this manuscript. We are especially grateful for the discussions with Michael and Anne-Kathrin Bugert regarding our study setup. Finally, we thank the many volunteers that donated annotations for this project, as this work would not have been possible without their generous participation.

This work has been funded by the German Research Foundation (DFG) as part of the Evidence project (GU 798/27-1), UKP-SQuARE (GU 798/29-1), INCEpTION (GU 798/21-1) and PEER (GU 798/28-1), and within the project "The Third Wave of AI" funded by the Hessian Ministry of Higher Education, Research, Science and the Arts (HWMK). Further, it has been funded by the German Federal Ministry of Education and Research and HMWK within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

K Addy, L Green, E Herron, and K Stepenuck. 2010. Why volunteer water quality monitoring makes sense. usdanifa volunteer water quality monitoring national facilitation project, factsheet ii. *US Department of Agriculture, Washington, DC.*

Marianne Grace Araneta, Gülşen Eryiğit, Alexander König, Ji-Ung Lee, Ana Luís, Verena Lyding, Lionel Nicolas, Christos Rodosthenous, and Federico Sangati. 2020. Substituto – A Synchronous Educational Language Game for Simultaneous Teaching and Crowdsourcing. In *Proceedings of the 9th Workshop on NLP for Computer Assisted Language Learning*, pages 1–9, Gothenburg, Sweden.

Imanol Arrieta-Ibarra, Leonard Goff, Diego Jiménez-Hernández, Jaron Lanier, and E. Glen Weyl. 2018. Should We Treat Data as Labor? Moving Beyond "Free". *AEA Papers and Proceedings*, 108:38–42.

Vino Avanesi and Jan Teurlings. 2022. "I'm not a robot," or am I?: Micro-labor and the immanent subsumption of the social in the human computation of ReCAPTCHAs. *International Journal of Communication*, 16(0):1–19.

Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. 2008. Addressing the resource bottleneck to create large-scale annotated texts. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, pages 375–380.

Sihao Chen, Daniel Khashabi, Wenpeng Yin, Chris Callison-Burch, and Dan Roth. 2019. Seeing Things from a Different Angle: Discovering Diverse Perspectives about Claims. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 542–557, Minneapolis, Minnesota.

K. Bretonnel Cohen, Karën Fort, Gilles Adda, Sophia Zhou, and Dimeji Farri. 2016. Ethical Issues in Corpus Linguistics And Annotation: Pay Per Hit Does Not Affect Effective Hourly Rate For Linguistic Resource Development On Amazon Mechanical Turk. *Proceedings of the tenth international conference on language resources and evaluation (LREC 2016)*, 2016(W40):8–12.

Florian Daniel, Pavel Kucherbaev, Cinzia Cappiello, Boualem Benatallah, and Mohammad Allahbakhsh. 2018. Quality Control in Crowdsourcing: A Survey of Quality Attributes, Assessment Techniques, and Assurance Actions. *ACM Computing Surveys*, 51(1):1–40.

Kordula De Kuthy, Ramon Ziai, and Detmar Meurers. 2016. Focus annotation of task-based data: A comparison of expert and crowd-sourced annotation in a reading comprehension corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 3928–3935, Portorož, Slovenia.

Leon Derczynski, Kalina Bontcheva, and Ian Roberts. 2016. Broad Twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179, Osaka, Japan.

Alexey Drutsa, Valentina Fedorova, Dmitry Ustalov, Olga Megorskaya, Evfrosiniya Zerminova, and Daria Baidakova. 2020. Crowdsourcing Practice for Efficient Data Labeling: Aggregation, Incremental Relabeling, and Pricing. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 2623–2627, Portland, Oregon, USA.

GülŞen Eryiğit, Ali Şentaş, and Johanna Monti. 2022. Gamified crowdsourcing for idiom corpora construction. *Natural Language Engineering*, 1(1):1–33.

European Parliament. 2016. Consolidated text: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance).

James Fiumara, Christopher Cieri, Jonathan Wright, and Mark Liberman. 2020. LanguageARC: Developing Language Resources Through Citizen Linguistics. In *Proceedings of the LREC 2020 Workshop on "Citizen Linguistics in Language Resource Development"*, pages 1–6, Marseille, France.

Karën Fort, Gilles Adda, and K. Bretonnel Cohen. 2011. Amazon Mechanical Turk: Gold Mine or Coal Mine? *Computational Linguistics*, 37(2):413–420.

Karën Fort, Bruno Guillaume, and Hadrien Chastant. 2014. Creating *Zombilingo* , a game with a purpose for dependency syntax annotation. In *Proceedings of the First International Workshop on Gamification for Information Retrieval*, pages 2–6, Amsterdam, The Netherlands.

Karën Fort, Aurélie Névéol, Yoann Dupont, and Julien Bezançon. 2022. Use of a citizen science platform for the creation of a language resource to study bias in language models for French: A case study. In *Proceedings of the 2nd Workshop on Novel Incentives in Data Collection from People: Models, Implementations, Challenges and Results within LREC 2022*, pages 8–13, Marseille, France.

W. N. Francis and H. Kucera. 1979. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US.

Yannick Frommherz and Alessandra Zarcone. 2021. Crowdsourcing Ecologically-Valid Dialogue Data for German. *Frontiers in Computer Science*, 3:1–21.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation Mining in User-Generated Web Discourse. *Computational Linguistics*, 43(1):125–179.

Mordechai Haklay, Daniel Dörler, Florian Heigl, Marina Manzoni, Susanne Hecker, and Katrin Vohland. 2021. What Is Citizen Science? The Challenges of Definition. In Katrin Vohland, Anne Land-Zandstra, Luigi Ceccaroni, Rob Lemmens, Josep Perelló, Marisa Ponti, Roeland Samson, and Katherin Wagenknecht, editors, *The Science of Citizen Science*, pages 13–33. Springer International Publishing, Cham.

Eric Hand. 2010. People power: Networks of human minds are taking citizen science to a new level. *Nature*, 466(7307):685—687.

Elizabeth T. Hobbs, Stephen M. Goralski, Ashley Mitchell, Andrew Simpson, Dorjan Leka, Emmanuel Kotey, Matt Sekira, James B. Munro, Suvarna Nadendla, Rebecca Jackson, Aitor Gonzalez-Aguirre, Martin Krallinger, Michelle Giglio, and Ivan Erill. 2021. ECO-CollecTF: A Corpus of Annotated Evidence-Based Assertions in Biomedical Manuscripts. *Frontiers in Research Metrics and Analytics*, 6:1–13.

Anthony M Holmes. 1991. *The Ontario Butterfly Atlas*. Entomologists' Association, Toronto.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia.

Dirk Hovy, Barbara Plank, and Anders Søgaard. 2014. Experiments with crowdsourced re-annotation of a POS tagging data set. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 377–382, Baltimore, Maryland.

Charles I. Jones and Christopher Tonetti. 2020. Nonrivalry and the Economics of Data. *American Economic Review*, 110(9):2819–2858.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9.

Margaret Kosmala, Andrea Wiggins, Alexandra Swanson, and Brooke Simmons. 2016. Assessing data quality in citizen science. *Frontiers in Ecology and the Environment*, 14(10):551–560.

Ji-Ung Lee, Jan-Christoph Klie, and Iryna Gurevych. 2022. Annotation Curricula to Implicitly Train Non-Expert Annotators. *Computational Linguistics*, pages 1–22.

Liza Lee, Mireille Desroches, Shamir Mukhi, and Christina Bancej. 2021. FluWatchers: Evaluation of a crowdsourced influenza-like illness surveillance application for Canadian influenza seasons 2015–2016 to 2018–2019. *Canada Communicable Disease Report*, 47(09):357–363.

Ronald D. Leeper, Jared Rennie, and Michael A. Palecki. 2015. Observational Perspectives from U.S. Climate Reference Network (USCRN) and Cooperative Observer Program (COOP) Network: Temperature and Precipitation Comparison. *Journal of Atmospheric and Oceanic Technology*, 32(4):703–721.

Chris J. Lintott, Kevin Schawinski, Anže Slosar, Kate Land, Steven Bamford, Daniel Thomas, M. Jordan Raddick, Robert C. Nichol, Alex Szalay, Dan Andreescu, Phil Murray, and Jan Vandenberg. 2008. Galaxy Zoo : Morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189.

Verena Lyding, Lionel Nicolas, and Alexander König. 2022. About the applicability of combining implicit crowdsourcing and language learning for the collection of NLP datasets. In *Proceedings of the 2nd Workshop on Novel Incentives in Data Collection from People: Models, Implementations, Challenges and Results within LREC 2022*, pages 46–57, Marseille, France.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

National Audubon Society. 2020. The Christmas bird count historical results.

An Nguyen, Byron Wallace, and Matthew Lease. 2015. Combining Crowd and Expert Labels Using Decision Theoretic Active Learning. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, 3(1):120–129.

Ellie Pavlick, Matt Post, Ann Irvine, Dmitry Kachaev, and Chris Callison-Burch. 2014. The Language Demographics of Amazon Mechanical Turk. *Transactions of the Association for Computational Linguistics*, 2:79–92.

Christopher Phillips, Dylan Walshe, Karen O'Regan, Ken Strong, Christopher Hennon, Ken Knapp, Conor Murphy, and Peter Thorne. 2018. Assessing Citizen Science Participation Skill for Altruism or University Course Credit: A Case Study Analysis Using Cyclone Center. *Citizen Science: Theory and Practice*, 3(1):6.

Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Livio Robaldo, and Luca Ducceschi. 2013. Phrase detectives: Utilizing collective intelligence for internet-scale language resource creation. *ACM Transactions on Interactive Intelligent Systems*, 3(1):1–44.

Lisa M. Rasmussen and Caren Cooper. 2019. Citizen Science Ethics. *Citizen Science: Theory and Practice*, 4(1):5.

David B. Resnik, Kevin C. Elliott, and Aubrey K. Miller. 2015. A framework for addressing ethical issues in citizen science. *Environmental Science & Policy*, 54:475–481.

Dana Rotman, Jenny Preece, Jen Hammock, Kezee Procita, Derek Hansen, Cynthia Parr, Darcy Lewis, and David Jacobs. 2012. Dynamic changes in motivation in collaborative citizen-science projects. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work - CSCW '12*, page 217, Seattle, Washington, USA.

Boaz Shmueli, Jan Fell, Soumya Ray, and Lun-Wei Ku. 2021. Beyond Fair Pay: Ethical Implications of NLP Crowdsourcing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3758–3769, Online.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and Fast – But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii.

Alexander Sorokin and David Forsyth. 2008. Utility data annotation with Amazon Mechanical Turk. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, Anchorage, AK, USA.

Alexandra Swanson, Margaret Kosmala, Chris Lintott, Robert Simpson, Arfon Smith, and Craig Packer. 2015. Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna. *Scientific Data*, 2(1):1–14.

Christopher Tauchmann, Johannes Daxenberger, and Margot Mieskes. 2020. The Influence of Input Data Complexity on Crowdsourcing Quality. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*, pages 71–72, Cagliari, Italy.

Ginger Tsueng, Steven M. Nanis, Jennifer Fouquier, Benjamin M. Good, and Andrew I. Su. 2016. Citizen Science for Mining the Biomedical Literature. *Citizen Science: Theory and Practice*, 1(2):14.

Noortje J. Venhuizen, Valerio Basile, Kilian Evang, and Johan Bos. 2013. Gamification for word sense labeling. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Short Papers*, pages 397–403, Potsdam, Germany.

L. von Ahn. 2006. Games with a Purpose. *Computer*, 39(6):92–94.

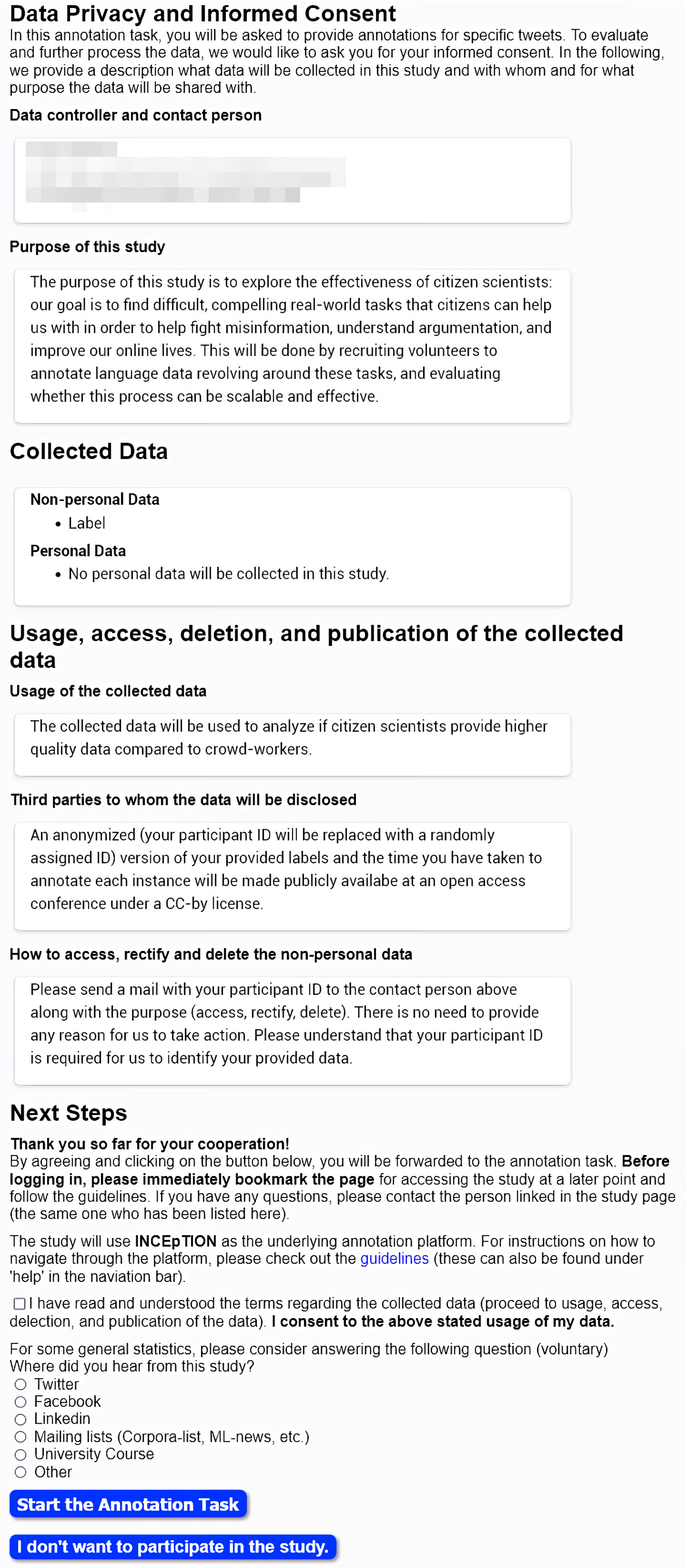
Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. 2008. reCAPTCHA: Human-Based Character Recognition via Web Security Measures. *Science*, 321(5895):1465–1468.

Aobo Wang, Cong Duy Vu Hoang, and Min-Yen Kan. 2013. Perspectives on crowdsourcing annotations for natural language processing. *Language Resources and Evaluation*, 47(1):9–31.

Sarah West, Alison Dyke, and Rachel Pateman. 2021. Variations in the Motivations of Environmental Citizen Scientists. *Citizen Science: Theory and Practice*, 6(1):14.

Amir Zeldes. 2017. The GUM corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.

Appendix A Landing Page



Data Privacy and Informed Consent

In this annotation task, you will be asked to provide annotations for specific tweets. To evaluate and further process the data, we would like to ask you for your informed consent. In the following, we provide a description what data will be collected in this study and with whom and for what purpose the data will be shared with.

Data controller and contact person

[redacted]

Purpose of this study

The purpose of this study is to explore the effectiveness of citizen scientists: our goal is to find difficult, compelling real-world tasks that citizens can help us with in order to help fight misinformation, understand argumentation, and improve our online lives. This will be done by recruiting volunteers to annotate language data revolving around these tasks, and evaluating whether this process can be scalable and effective.

Collected Data

Non-personal Data

- Label

Personal Data

- No personal data will be collected in this study.

Usage, access, deletion, and publication of the collected data

Usage of the collected data

The collected data will be used to analyze if citizen scientists provide higher quality data compared to crowd-workers.

Third parties to whom the data will be disclosed

An anonymized (your participant ID will be replaced with a randomly assigned ID) version of your provided labels and the time you have taken to annotate each instance will be made publicly availabe at an open access conference under a CC-by license.

How to access, rectify and delete the non-personal data

Please send a mail with your participant ID to the contact person above along with the purpose (access, rectify, delete). There is no need to provide any reason for us to take action. Please understand that your participant ID is required for us to identify your provided data.

Next Steps

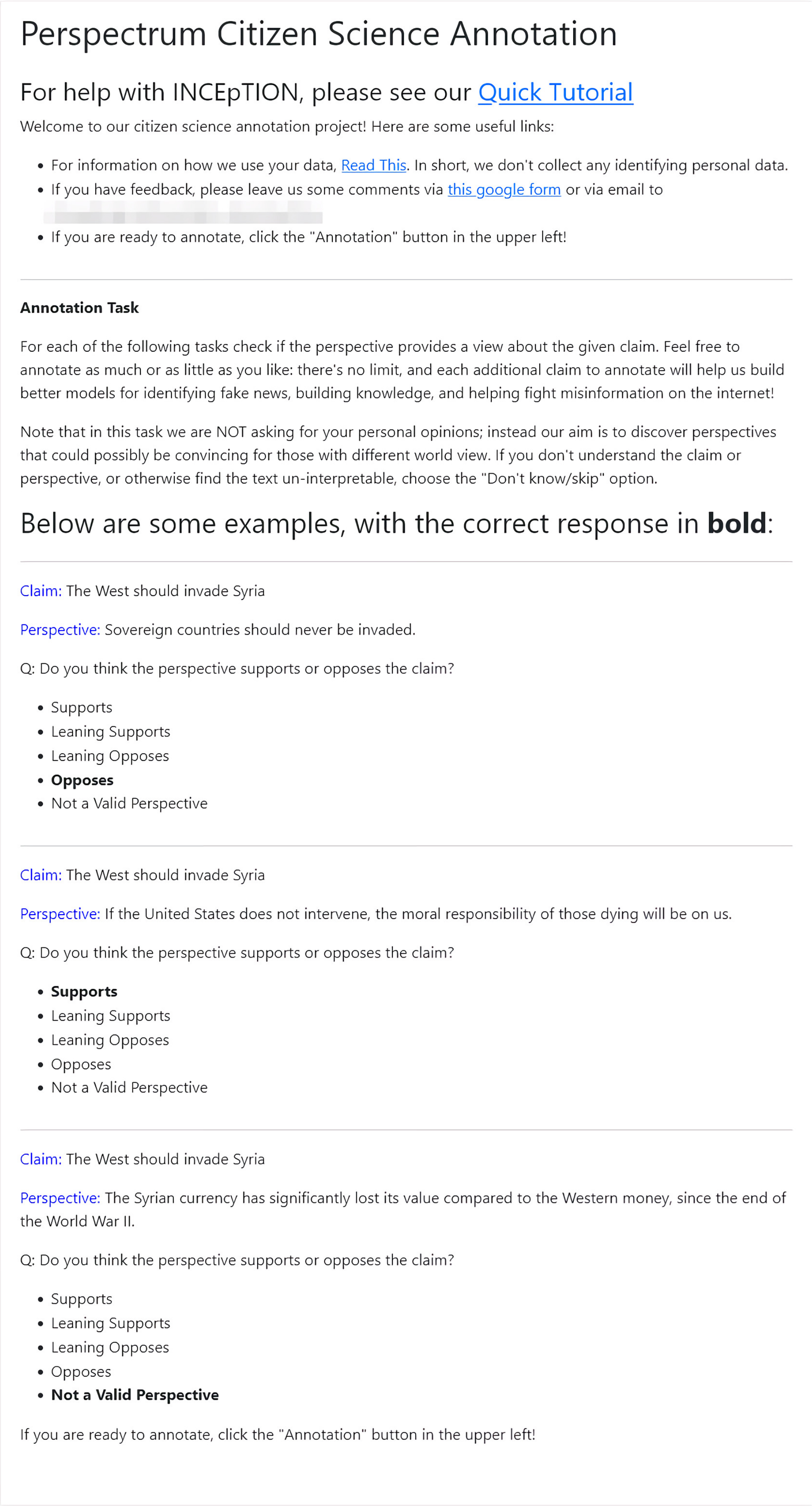
Thank you so far for your cooperation!
By agreeing and clicking on the button below, you will be forwarded to the annotation task. **Before logging in, please immediately bookmark the page** for accessing the study at a later point and follow the guidelines. If you have any questions, please contact the person linked in the study page (the same one who has been listed here).

The study will use **INCEpTION** as the underlying annotation platform. For instructions on how to navigate through the platform, please check out the guidelines (these can also be found under 'help' in the naviation bar).

☐ I have read and understood the terms regarding the collected data (proceed to usage, access, delection, and publication of the data). **I consent to the above stated usage of my data.**

For some general statistics, please consider answering the following question (voluntary)
Where did you hear from this study?
○ Twitter
○ Facebook
○ Linkedin
○ Mailing lists (Corpora-list, ML-news, etc.)
○ University Course
○ Other

Appendix B Annotation Guidelines



Perspectrum Citizen Science Annotation

For help with INCEpTION, please see our Quick Tutorial

Welcome to our citizen science annotation project! Here are some useful links:

- For information on how we use your data, Read This. In short, we don't collect any identifying personal data.
- If you have feedback, please leave us some comments via this google form or via email to [redacted]
- If you are ready to annotate, click the "Annotation" button in the upper left!

Annotation Task

For each of the following tasks check if the perspective provides a view about the given claim. Feel free to annotate as much or as little as you like: there's no limit, and each additional claim to annotate will help us build better models for identifying fake news, building knowledge, and helping fight misinformation on the internet!

Note that in this task we are NOT asking for your personal opinions; instead our aim is to discover perspectives that could possibly be convincing for those with different world view. If you don't understand the claim or perspective, or otherwise find the text un-interpretable, choose the "Don't know/skip" option.

Below are some examples, with the correct response in **bold**:

Claim: The West should invade Syria

Perspective: Sovereign countries should never be invaded.

Q: Do you think the perspective supports or opposes the claim?

- Supports
- Leaning Supports
- Leaning Opposes
- **Opposes**
- Not a Valid Perspective

Claim: The West should invade Syria

Perspective: If the United States does not intervene, the moral responsibility of those dying will be on us.

Q: Do you think the perspective supports or opposes the claim?

- **Supports**
- Leaning Supports
- Leaning Opposes
- Opposes
- Not a Valid Perspective

Claim: The West should invade Syria

Perspective: The Syrian currency has significantly lost its value compared to the Western money, since the end of the World War II.

Q: Do you think the perspective supports or opposes the claim?

- Supports
- Leaning Supports
- Leaning Opposes
- Opposes
- **Not a Valid Perspective**

If you are ready to annotate, click the "Annotation" button in the upper left!

Appendix C Questions to keep in mind for a citizen science project

- What is the purpose of the study?
- What kind of personal and non-personal data will be collected?[2]
- If there is a questionnaire involved, what questions will it involve?
- How will the data be used?
- Is a publication of the data planned and if so, which data will be published and will it be anonymized?
- How can participants request access, change, or deletion of their data?

Appendix D Project Statistics

D.1 Number of participants

In addition to the plots visualizing the number of participants (c.f. Fig. 3), we also list the raw numbers in Table 3.

Channel	Participants
Courses	14
Facebook	3
LinkedIn	4
Lists	55
Twitter	8
Undisclosed	17

Table 3: Number of participants per channel.

D.2 Annotation statistics

In addition to the plots visualizing the annotation counts and label distribution (c.f. Fig. 4), we also list the raw numbers in Table 4.

Table 4: Label distribution grouped by channel. Labels are *supports* (++), *mildly-supports* (+), *mildly-opposes* (-), *opposes* (--), *not a valid perspective* (I) and *Skip* (S).

Channel	Total	Counts						Percentage					
		+	++	-	--	I	S	+	++	-	--	I	S
Courses	307	18	108	24	104	28	25	5.86	35.18	7.82	33.88	9.12	8.14
Facebook	5	2	0	0	0	0	3	40.00	0.00	0.00	0.00	0.00	60.00
LinkedIn	21	1	5	1	7	3	4	4.76	23.81	4.76	33.33	14.29	19.05
Lists	830	98	264	48	222	92	106	11.81	31.81	5.78	26.75	11.08	12.77
Twitter	131	14	42	12	39	13	11	10.69	32.06	9.16	29.77	9.92	8.40
Undisclosed	187	18	53	15	52	27	22	9.63	28.34	8.02	27.81	14.44	11.76

---

[2] We provided some pre-defined suggestions such as *Name* or *IP* for personal data and *Label* for non-personal data with the possibility to add more in our landingpage module.

Contrastive Learning with Keyword-based Data Augmentation for Code Search and Code Question Answering

Shinwoo Park[1], **Youngwook Kim**[2] and **Yo-Sub Han**[2]
[1]Department of Artificial Intelligence, Yonsei University, Seoul, Republic of Korea
[2]Department of Computer Science, Yonsei University, Seoul, Republic of Korea
`{pshkhh, youngwook, emmous}@yonsei.ac.kr`

Abstract

The semantic code search is to find code snippets from the collection of candidate code snippets with respect to a user query that describes functionality. Recent work on code search proposes data augmentation of queries for contrastive learning. This data augmentation approach modifies random words in queries. When a user web query for searching code snippet is too brief, the important word that represents the search intent of the query could be undesirably modified. A code snippet has informative components such as function name and documentation that describe its functionality. We propose to utilize these code components to identify important words and preserve them in the data augmentation step. We present KeyDAC (Keyword-based Data Augmentation for Contrastive learning) that identifies important words for code search from queries and code components based on term matching. KeyDAC augments query-code pairs while preserving keywords, and then leverages generated training instances for contrastive learning. We use KeyDAC to fine-tune various pre-trained language models and evaluate the performance of code search and code question answering via CoSQA and WebQueryTest. The experimental results confirm that KeyDAC substantially outperforms the current state-of-the-art performance, and achieves the new state-of-the-arts for both tasks.

1 Introduction

Software developers or students who major in computer science often write natural language queries to search for code snippets with desired functionality from the web search engine. The retrieved code snippets are reused or referred to improve productivity of software development. Semantic code search is a well-known code-related downstream task that measures the semantic relevance between a given natural language query and a collection of code snippets to retrieve the most relevant code snippet. CodeXGLUE (Lu et al., 2021), a benchmark for 10 code tasks, provides two test sets—CodeSearchNet AdvTest and WebQueryTest—for an open challenge code search. CodeSearchNet AdvTest from the CodeSearchNet (Husain et al., 2019) corpus is for a retrieval scenario in which one needs to find a Python code function that matches the search intent of the query best. The queries in CodeSearchNet AdvTest are not real user queries. Instead, they are documentations such as comments or summaries written by developers. WebQueryTest is a code question answering task that asks whether a Python code function has functionality described by the real user web query collected from the search logs of a commercial search engine. Recently, Huang et al. (2021) introduce CoSQA that consists of pairs of real user web queries and Python code snippets. Then, they use CoSQA as a benchmark for code search to resemble the real world scenario where developers write natural language queries to find code snippets.

Huang et al. (2021) propose CoCLR, a contrastive learning method using a query-rewritten data augmentation, to improve the code search performance. CoCLR modifies the random words in queries to augment training query-code pair instances. However, besides the user queries, there are useful components that describe the functionality of code snippets. Since a function name and documentation describe functionality of the corresponding code function, we propose to use these components as well for data augmentation. Furthermore, we suggest considering the relative importance of words in query and code function. Note that the data augmentation used in the previous code search method treats all words equally. Xie et al. (2020) use TF-IDF to calculate the relative importance of words in a sentence, then replace unimportant words to augment training sentences. We adapt this idea and define the relative importance in code search based on term matching between a

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3609–3619
May 2-6, 2023 ©2023 Association for Computational Linguistics

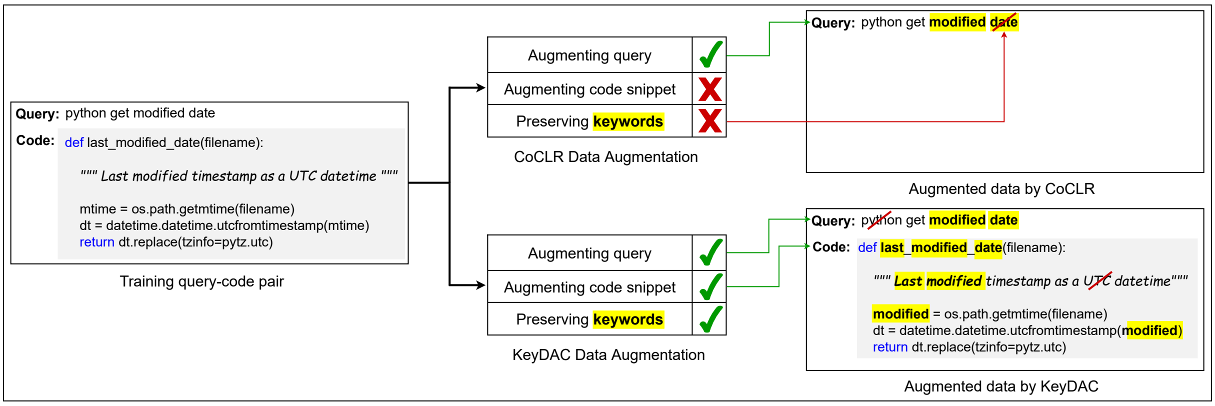


Figure 1: Comparison of data augmentation approaches for semantic code search. Example query-code pair is from CoSQA (Huang et al., 2021) training dataset. Keywords are marked in yellow and red slash line denotes the deletion of a word. Both CoCLR (Huang et al., 2021) and KeyDAC apply data augmentation to query. In contrast to CoCLR, KeyDAC also augments code snippet (deleting `UTC` in documentation and renaming variable `mtime` to *modified*, which is one of keywords). While CoCLR can delete the keyword *date*, KeyDAC preserves keywords.

paired query and code snippet.

We present KeyDAC, **Key**word-based **D**ata **A**ugmentation for **C**ontrastive learning that identifies keywords from a given query-code pair based on term matching. KeyDAC applies data augmentation both on natural language (NL) sequences (*i.e.*, query, function name, and documentation) and programming language (PL) sequences (*i.e.*, code statements), and generates more training query-code pair instances while preserving keywords. Figure 1 shows an example training query-code pair instance from CoSQA training dataset and compares the results of two data augmentation approaches, CoCLR and KeyDAC. The query asks how to get modified date. In this case, important words closely related to the search intent are *modified* and *date*. Since CoCLR modifies random words in query, *date* could be modified in data augmentation process. The augmented query by CoCLR is *python get modified* (keyword *date* is deleted) which does not represent clear search intent. On the other hand, KeyDAC preserves keywords in the data augmentation step. In addition, KeyDAC augments code snippet by deleting unimportant word `UTC` in documentation, and renaming the variable in the code statements using keywords (`mtime` to *modified*). We evaluate the effectiveness of KeyDAC on CoSQA and open challenge WebQueryTest.[1] We use KeyDAC to fine-tune various pre-trained language models. KeyDAC brings substantial performance gain, resulting in new state-of-the-art performance both on code search and code question answering tasks. Our main contributions are as follows:

- We propose KeyDAC—data augmentation mechanism for contrastive learning, which identifies important words from training query-code pairs and augments them while preserving identified keywords.
- We demonstrate that KeyDAC outperforms the current SOTA for the code search task on CoSQA benchmark.
- We achieve a new record, with a substantial improvement, for the open challenge code QA, WebQueryTest.

2 Related Work

2.1 Code Search Methods

Some researchers proposed information retrieval-based approaches using term matching between queries and code snippets (Lu et al., 2015; Lv et al., 2015). However, since these traditional approaches focused on lexical information, they often fail to understand the semantic relationship between the query and the code snippet. Recently, many researchers started to address the problem through deep learning-based approaches (Cambronero et al., 2019; Li et al., 2020) that learn the semantic representation of the query and code. Later, pre-trained models for programming languages were proposed and showed an improvement on code search tasks. CodeBERT (Feng et al., 2020) is a bimodal pre-trained model for NL and PL, which was pre-trained with masked language modeling

---

[1] The leaderboard of WebQueryTest is available at `https://microsoft.github.io/CodeXGLUE/`

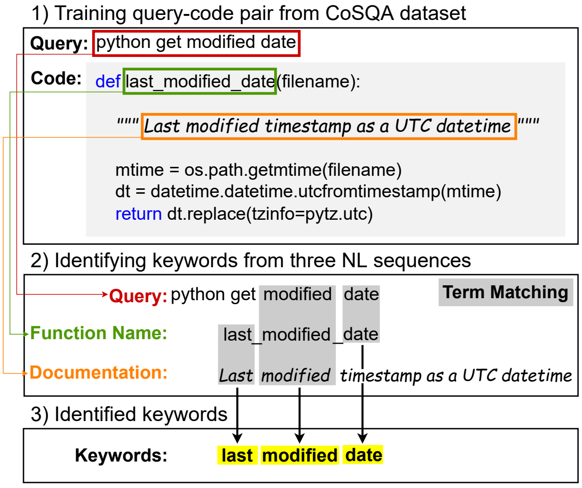
(MLM) and replaced token detection (RTD). Feng et al. (2020) showed that CodeBERT outperformed both RoBERTa and RoBERTa pre-trained on code datasets. GraphCodeBERT (Guo et al., 2020) further improved the performance by pre-training using data flow as a semantic-level structure of code.

2.2 Contrastive Learning for Code Search

Contrastive learning encourages the distance between similar instances to be minimized and the distance between dissimilar instances to be maximized in the representation space. Recently, contrastive learning showed its effectiveness and became popular in self-supervised learning (Chen et al., 2020; Meng et al., 2021; Gao et al., 2021). Such effectiveness of contrastive learning in various fields promoted adapting contrastive learning for code search. As a pre-training approach for source code, Corder (Bui et al., 2021) transformed a code snippet into different versions and minimized the distance between them in the representation space. Corder used program transformation operators, including dead code insertion and permutation of code statements. The experiment results confirmed that pre-training with contrastive learning was effective in improving performance on several code-related tasks such as code-to-code search, text-to-code search, and code summarization. As a fine-tuning approach for code search, CoCLR (Huang et al., 2021) used contrastive learning with query-rewritten data augmentation and in-batch negative samples. For query-rewriting, CoCLR modified random words in a query as follows: deleting random words, switching the position of two random words, or copying random words. They showed that CoCLR improved code search performance of CodeBERT. However, these contrastive learning approaches utilize either query or code in the data augmentation process and do not consider the relative importance of words. We propose to consider the importance of each word differently and preserve keywords for data augmentation.

3 Approach

KeyDAC identifies keywords from positive query-code pairs (*i.e.*, a code snippet meets the demand of a query) in a training dataset based on term matching. Then KeyDAC applies keyword-based data augmentation to both query and code snippet. Using augmented training data, KeyDAC deploys contrastive learning to fine-tune pre-trained encoders for the code search task.



```
def last_modified_date(filename):
    """Last modified timestamp as a UTC datetime"""

    mtime = os.path.getmtime(filename)
    dt = datetime.datetime.utcfromtimestamp(mtime)
    return dt.replace(tzinfo=pytz.utc)
```

Figure 2: An example of identifying keywords from a pair of real user query and code function. Given a training query-code pair, we identify keywords based on term matching from three NL sequences (*i.e.*, query, function name, and documentation). The identified keywords are related to the functionality of the code function.

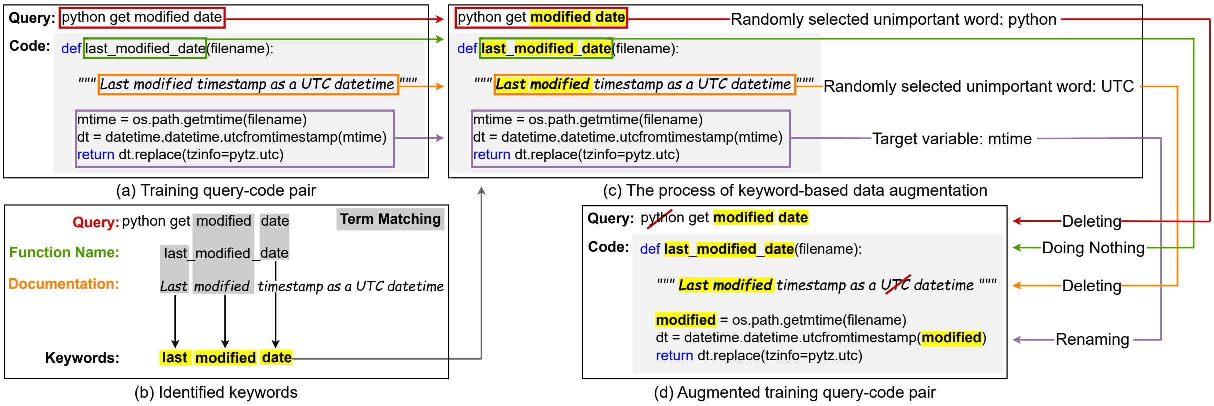
3.1 Data Augmentation with Keywords

A code function c_i has the following three main components:

- function name in the function header (NL)
- function-level documentation (NL)
- code statements in the function body (PL)

The previous approaches consider these three components as PL sequences. On the other hand, we consider the function name and the documentation as NL sequences, since 1) those two code components describe the functionality of the code snippet; 2) modifying the function name or documentation does not produce any syntax errors.

Identifying Keywords Since a user query demands certain functionality of the code snippet, KeyDAC utilizes two NL descriptions of code function, such as the function name and documentation, to identify keywords. Specifically, KeyDAC identifies common words from three NL sequences (*i.e.*, query, function name, and documentation) based on term matching. Figure 2 gives an example of how KeyDAC identifies keywords from the paired query and code function. The identified keywords from the example are: *last*, *modified* and *date*. These



(a) Training query-code pair

(b) Identified keywords

(c) The process of keyword-based data augmentation

(d) Augmented training query-code pair

Figure 3: An illustration of keyword-based data augmentation. For NL sequences, keywords (*i.e.*, *last*, *modified*, and *date*) are preserved (highlighted in bold) while unimportant words (*i.e.*, `python` in query and `UTC` in documentation) are deleted (red slash lines) (**NL: Rewriting**). For PL sequences, variable `mtime` is renamed using a keyword *modified* (**PL: Variable Renaming**).

identified keywords are related to the functionality of code function.

Using the identified keywords, KeyDAC applies data augmentation to NL and PL sequences in different ways. In the following, we demonstrate keyword-based data augmentation in detail.

NL: Rewriting KeyDAC rewrites three NL sequences by modifying unimportant words while preserving keywords by choosing one of the following four ways: 1) deleting one randomly selected unimportant word (Delete); 2) switching the position of two randomly selected unimportant words (Switch); 3) copying one randomly selected unimportant word (Copy); 4) doing nothing (None). In Figure 3 (d), KeyDAC deletes an unimportant word `python` in the query and `UTC` in the documentation, while preserving keywords *last*, *modified* and *date*. **PL: Variable Renaming** Software developers sometimes name a variable in abbreviation form; in other words, there can be a lexical gap between query and variable name (*e.g.*, in Figure 3 (a), the variable `mtime` represents the meaning of *modified time*). We propose to rename variables using keywords to bridge the lexical gap between query and code snippet. We first parse a code function c_i into an abstract syntax tree (AST) that represents the syntactic structure of c_i. Then we identify variables from terminal nodes of the resulting AST. Then KeyDAC renames the variables that appear most frequently in code statements using keywords. If there is a keyword that matches the target variable name (`mtime`) with the first letter, KeyDAC uses that keyword (*modified*). If not, KeyDAC randomly chooses one from keywords (*i.e.*, *last*, *modified*, and *date*). In Figure 3 (d), KeyDAC renames `mtime` to *modified*.

3.2 Siamese Network for Code Search Task

We adopt siamese network architecture, which has a shared encoder to map a query and code snippet to fixed-sized embeddings. Each query q_i and code function c_i are encoded by a shared encoder $Encoder$ (*e.g.*, CodeBERT). We take the representation of $[CLS]$ token from the last hidden layer of $Encoder$. Then we compute the cosine similarity $sim^{(q_i, c_i)}$ between a query-code pair (q_i, c_i) as:

$$sim^{(q_i,c_i)} = \langle Encoder(q_i), Encoder(c_i) \rangle, \quad (1)$$

where $\langle \cdot \rangle$ indicates cosine similarity operation. We use binary cross-entropy as the training objective:

$$\mathcal{L}_{bce} = -[y_i \cdot \log sim^{(q_i,c_i)} + (1 - y_i) \log(1 - sim^{(q_i,c_i)})], \quad (2)$$

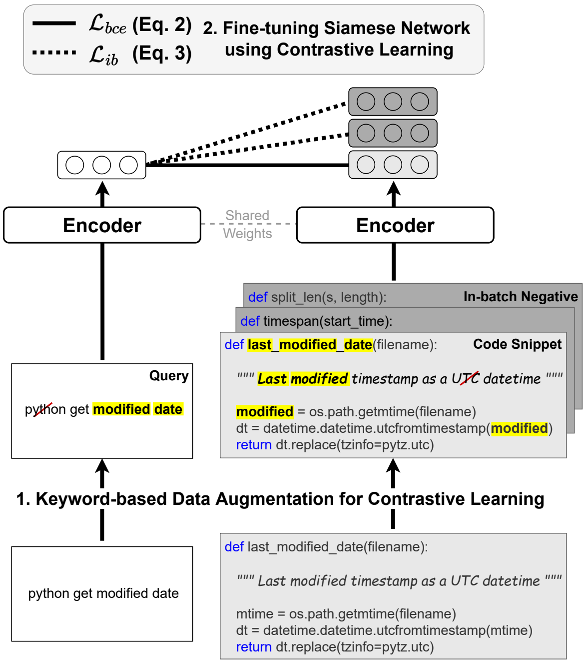
where y_i is the ground truth label of (q_i, c_i).

3.3 Contrastive Learning

KeyDAC uses contrastive learning to optimize the parameters of $Encoder$. This contrastive learning aims to maximize the similarity of the query q_i and code function c_i with label of $y_i = 1$ while minimizing the similarity of the query with unrelated code snippets.

Given the query-code pair (q_i, c_i) in a batch of size N, we consider the other $N - 1$ code snippets as unrelated. The contrastive loss with in-batch negative samples is defined as:

$$\mathcal{L}_{ib} = -\frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} \log \left(1 - sim^{(q_i,c_j)}\right). \quad (3)$$



```
def split_len(s, length):
```

```
def timespan(start_time):
```

```
def last_modified_date(filename):

  """ Last modified timestamp as a UTC datetime """

  modified = os.path.getmtime(filename)
  dt = datetime.datetime.utcfromtimestamp(modified)
  return dt.replace(tzinfo=pytz.utc)
```

```
def last_modified_date(filename):

  """ Last modified timestamp as a UTC datetime """

  mtime = os.path.getmtime(filename)
  dt = datetime.datetime.utcfromtimestamp(mtime)
  return dt.replace(tzinfo=pytz.utc)
```

Figure 4: The fine-tuning process of siamese network using KeyDAC. The boxes, colored in dark gray, indicate the in-batch negative samples. The black solid line denotes the binary cross-entropy loss and black dashed lines denote the contrastive loss with in-batch negative.

The overall training objective is:

$$\mathcal{L} = \mathcal{L}_{bce} + \mathcal{L}_{ib}. \tag{4}$$

Figure 4 illustrates the fine-tuning process of siamese network with shared encoder using KeyDAC. A black solid line denotes the binary cross-entropy loss (Equation 2), and black dashed lines denote the contrastive loss with in-batch negative samples (Equation 3).

4 Experiments

4.1 Tasks

We evaluate the performance for two tasks, code search and code question answering.

Code Search aims to retrieve the most relevant code function c^* in a collection of H code snippets $C = c_1, \ldots, c_H$ according to a real user web query q_i. Following CoSQA, we use Mean Reciprocal Rank (MRR) as evaluation metric.

Code Question Answering is a form of binary classification and, thus, predicts a label of 1 or 0; the label indicates whether the code function c_i matches the search intent of the query q_i. The WebQueryTest uses accuracy score as its official evaluation metric.

4.2 Datasets

We use CoSQA (Huang et al., 2021) and WebQueryTest (Lu et al., 2021) to evaluate the effectiveness of KeyDAC. The queries in both datasets are real user queries. Table 1 shows the data statistics. Each number denotes the number of paired queries and code snippets.

Code Search: We follow the same dataset split as CoSQA. The number of candidate code snippets is 6,267 ($H = 6,267$).

Code Question Answering: Each instance of WebQueryTest is a pair of a real user web query and a Python code function. Following CoSQA, we train the models using the CoSQA dataset, then use WebQueryTest as the test dataset. Since WebQueryTest is an open challenge, we submit model predictions to the CodeXGLUE official leaderboard, and report the evaluated results.

4.3 Baseline Approaches

- **In-batch**: Contrastive learning method using in-batch negative (without data augmentation).
- **CoCLR**: Contrastive learning method with query-rewritten data augmentation and in-batch negative.

Huang et al. (2021) report that switching the position of two random words in queries for the data augmentation in CoCLR achieves the best performance. Thus, we use switch as the query-rewriting operation for CoCLR in the experiments (Table 2).

4.4 Experiment Setup

We set the batch size N as 32, the learning rate as 1e-5 and the fine-tuning epoch as 10. We use the Adam optimizer (Kingma and Ba, 2014) to train the models. We conduct all experiments on an NVIDIA RTX3090 GPU with 24GB memory.

We use the following pre-trained models[2]:

- **RoBERTa** (Liu et al., 2019) is pre-trained on a large natural language text corpus with masked language modeling (MLM) objective.
- **CodeBERT** (Feng et al., 2020) is pre-trained on six programming languages with MLM and replaced token detection objectives.

---

[2] We use HuggingFace's implementation for these models. (`https://huggingface.co/`)

Dataset	**Task**	**Metric**	**Train**	**Valid**	**Test**
CoSQA (Huang et al., 2021)	Code Search	MRR	19,604	500	500
	Code Question Answering	Accuracy	20,000	604	1,046*

Table 1: Dataset statistics. * This is the number of test data in the WebQueryTest open challenge.

Model	**Approach**	**Code Search**	**Code Question Answering**[3]
RoBERTa (Liu et al., 2019)	In-batch	58.37 ± 0.48	58.89
	CoCLR	61.00 ± 0.98	60.70
	KeyDAC	**67.09 ± 0.37**	**60.99**
CodeBERT (Feng et al., 2020)	In-batch	66.72 ± 0.35	57.36
	CoCLR	68.42 ± 0.44	60.03
	KeyDAC	**72.76 ± 0.93**	**62.90**
GraphCodeBERT (Guo et al., 2020)	In-batch	71.34 ± 0.46	63.73±1.28
	CoCLR	71.78 ± 0.69	63.47±1.31
	KeyDAC	**74.93 ± 0.42**	**65.51**±0.77
UniXcoder (Guo et al., 2022)	In-batch	71.87 ± 0.59	62.58±1.06
	CoCLR	71.52 ± 0.85	62.10±0.52
	KeyDAC	**74.71 ± 0.45**	**64.14**±0.78

Table 2: Results on code search and code question answering tasks. The best results for each model are highlighted in bold.

- **GraphCodeBERT** (Guo et al., 2020) leverages data flow of code for two structure-aware pre-training tasks, including edge prediction and node alignment.
- **UniXcoder** (Guo et al., 2022) is a unified pre-trained model which uses code documentation and AST for contrastive pre-training. We use encoder of UniXcoder.

We deploy contrastive learning approaches to fine-tune the aforementioned pre-trained language models.

4.5 Results

Table 2 compares different contrastive learning approaches to code search and code question answering. We report the mean performance with standard deviation in 5 runs for code search task.[3] Remark that the in-batch negative contrastive learning approach (In-batch) does not use data augmentation. We highlight the best results for each pre-trained model in bold.

The results show that KeyDAC consistently outperforms contrastive learning with in-batch negative only (In-batch) and contrastive learning with query-rewritten data augmentation (CoCLR). GraphCodeBERT fine-tuned using KeyDAC achieves the highest performance both on code search and code QA tasks. We notice that CoCLR drops code search performance of UniXcoder and code QA performance of GraphCodeBERT and UniXcoder. On the other hand, KeyDAC shows consistent performance improvement.

5 Analysis

We use GraphCodeBERT as base model for following analyses since among four pre-trained models, GraphCodeBERT achieves the highest performance on code search and code QA when fine-tuned using KeyDAC.

5.1 The Effect of Preserving Keywords

From the experimental results, we observe that KeyDAC consistently outperforms CoCLR. We hypothesize that the main reason for the performance gain is preserving keywords in data augmentation. We study the effect of preserving keywords, especially in queries to directly compare KeyDAC and CoCLR. We perform query-rewritten data augmentation in three ways: 1) Preserving keywords (same as applying keyword-based data augmentation to query only); 2) Deleting a random word (same as

---

[3] For code QA task, we submit the prediction results of 3 random seeds for the two best models and the prediction results of 1 random seed for the others, since the CodeXGLUE guideline discourages excessive submissions to avoid P-hacking.

Component	Rewriting	Code Search
Query	Preserving keywords	**72.03**
	Deleting a random word	71.07
	Deleting a keyword	68.03

Table 3: Effect of preserving keywords in query-rewritten data augmentation.

Model	Data Augmentation	Code Search
GraphCodeBERT	No augmentation	71.46
	(Delete)	**74.81**
	(Switch)	73.74
	(Copy)	72.37

Table 4: Effect of NL rewriting operations. The result in the first row indicates the performance of fine-tuning GraphCodeBERT using in-batch negative only.

CoCLR with delete operation); 3) Deleting a keyword. We analyze this via code search task on CoSQA test set to avoid an excessive submission of code QA prediction results to the WebQueryTest leaderboard. Table 3 shows the results. We can observe that preserving keywords in query-rewritten data augmentation achieves the best performance. The deletion of keywords in queries shows significant performance degradation. The results suggest that keywords determined through our proposed way (Figure 2) are important to the performance of the semantic code search task.

5.2 The Effect of NL Rewriting Operation

We conduct experiments to investigate the effect of different NL rewriting operations on the code search task. Table 4 shows the code search performance evaluated on the CoSQA test set. We observe that KeyDAC with all three NL rewriting operations outperforms GraphCodeBERT fine-tuned using contrastive learning with in-batch negative only (No augmentation).

Among the three rewriting operations, delete shows the best performance (highlighted in bold). We conjecture that deleting unimportant words helps to reduce the noise of data. For example, user web queries often have a typo, and documentation often contains a special character such as >>> to show the execution result.

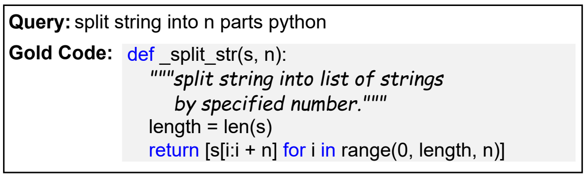
5.3 Contribution of Each Component

We investigate the contributions of each component to keyword-based data augmentation. Table 5 shows the results for code search task on the CoSQA test set. All five rows determine keywords from pairs of query and code function with the same way. However, the last four rows only modify each component as follows: 1) deleting unimportant words in queries; 2) deleting unimportant words in function names; 3) deleting unimportant words in documentations; 4) renaming variables using keywords.

The results show that KeyDAC leveraging all components (results in the first row) achieves the best result. Among the results of KeyDAC using a single component, documentation contributes the most to performance, and the function name contributes the least. Software developers typically use an abbreviated function name but write documentation in detail to describe the functionality of the code snippet.

5.4 Case Study

We conduct a case study to analyze the consistent code search ability of KeyDAC. More cases for other pre-trained language models can be found in Appendix A.1. In addition, we provide some cases to compare the prediction results of CoCLR and KeyDAC for code question answering task in Appendix A.2

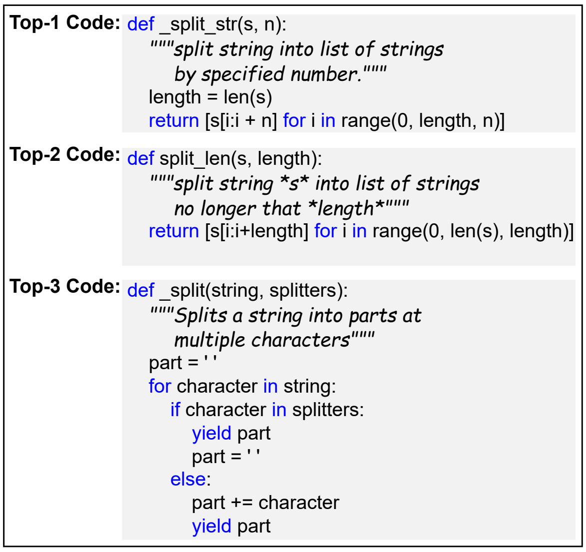


Query: split string into n parts python

Gold Code:

```
def _split_str(s, n):
    """split string into list of strings
       by specified number."""
    length = len(s)
    return [s[i:i + n] for i in range(0, length, n)]
```

(a) A pair of real user web query and gold code function.



Top-1 Code:

```
def _split_str(s, n):
    """split string into list of strings
       by specified number."""
    length = len(s)
    return [s[i:i + n] for i in range(0, length, n)]
```

Top-2 Code:

```
def split_len(s, length):
    """split string *s* into list of strings
       no longer that *length*"""
    return [s[i:i+length] for i in range(0, len(s), length)]
```

Top-3 Code:

```
def _split(string, splitters):
    """Splits a string into parts at
       multiple characters"""
    part = ''
    for character in string:
        if character in splitters:
            yield part
            part = ''
        else:
            part += character
    yield part
```

(b) Code search results with KeyDAC (top-3 results).

Figure 5: Code search results with KeyDAC on CoSQA test set. The code snippets are searched from 6,267 candidates and ranked in the order of semantic relatedness.

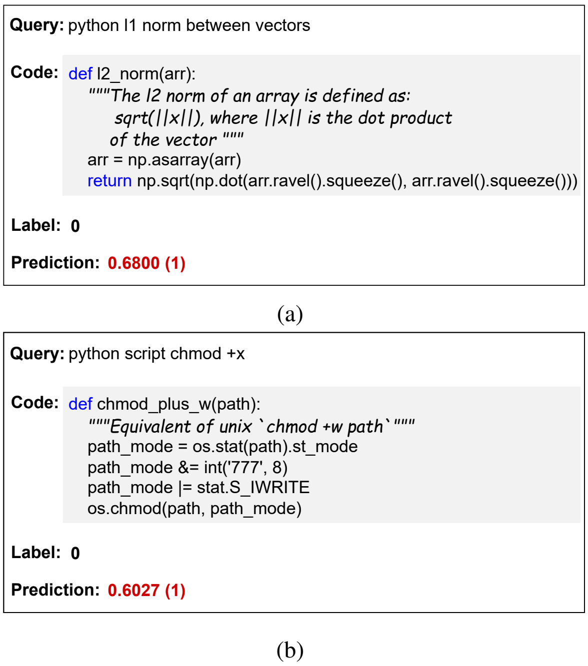
Model	Component	Keyword-based Data Augmentation	Code Search
GraphCodeBERT	All Components	NL Rewriting via delete & Variable Renaming	**74.81**
	Query	Deleting unimportant words in queries	72.03
	Function Name	Deleting unimportant words in function names	71.39
	Documentation	Deleting unimportant words in documentations	73.26
	Code Statements	Renaming variables using keywords	72.57

Table 5: The first row indicates the performance of KeyDAC which applied keyword-based data augmentation to all components. The last four rows are the results of applying keyword-based data augmentation to a single component.

Figure 5 shows the code search results with KeyDAC for the query *split string into n parts python*. We only give the top-3 results due to the space limit. KeyDAC returns the gold code function as its top-1 result. The other two code snippets have the same functionality as the gold code, demonstrating the consistent code search ability of KeyDAC.

5.5 Error Analysis

Figure 6 provides two error cases of KeyDAC on code QA task. KeyDAC makes a wrong prediction in the case of Figure 6(a). The purpose of the search for the query is to calculate the l1 norm between vectors. However, the functionality of the code is computing l2 norm of a given array. The model needs mathematical knowledge to understand the difference between norm l1 and norm l2.



Query: python l1 norm between vectors

Code:
```
def l2_norm(arr):
    """The l2 norm of an array is defined as:
      sqrt(||x||), where ||x|| is the dot product
     of the vector """
    arr = np.asarray(arr)
    return np.sqrt(np.dot(arr.ravel().squeeze(), arr.ravel().squeeze()))
```

Label: **0**

Prediction: **0.6800 (1)**

(a)

Query: python script chmod +x

Code:
```
def chmod_plus_w(path):
    """Equivalent of unix `chmod +w path` """
    path_mode = os.stat(path).st_mode
    path_mode &= int('777', 8)
    path_mode |= stat.S_IWRITE
    os.chmod(path, path_mode)
```

Label: **0**

Prediction: **0.6027 (1)**

(b)

Figure 6: Two error cases of KeyDAC for code question answering task on CoSQA validation set.

In the case of Figure 6(b), the search intent of the query is to add the execution privilege (*chmod +x*) while the functionality of the code is to add the write privilege (*chmod +w*). KeyDAC fails to understand the difference between *+x* and *+w*, resulting in a wrong prediction. While software developers can easily understand the difference of Unix/Linux command *chmod +x* and *chmod +w*, it is not trivial for the language models and humans without programming knowledge.

The potential research direction is to incorporate domain-specific knowledge, such as mathematical knowledge and programming knowledge, into the pre-training or fine-tuning process of language models to improve code search performance.

6 Conclusion

We have presented KeyDAC— keyword-based data augmentation for contrastive learning, which generates more training query-code pairs while preserving important keywords for the code search task. First, KeyDAC utilizes term matching technique to identify important words from a query and code components (function name and documentation). Then, KeyDAC augments both a query and a code snippet while preserving the identified keywords. Finally, KeyDAC deploys contrastive learning using the augmented data to fine-tune the pre-trained language models. We have demonstrated that KeyDAC outperforms the current state-of-the-art performance on both the code search and an open challenge code question answering task.

Limitations

Given a query-code pair, KeyDAC identifies keywords which share the same surface form by term matching. In other words, KeyDAC identifies keywords at the lexical level. As a future work, KeyDAC can utilize external knowledge for keyword-based data augmentation. For example, KeyDAC can utilize WordNet to identify keywords based on not only surface form, but also synonyms.

Acknowledgements

This research was supported by the NRF grant (RS-2023-00208094) and the AI Graduate School Program (No. 2020-0-01361) funded by the Korea government (MSIT). Han is a corresponding author.

References

Nghi DQ Bui, Yijun Yu, and Lingxiao Jiang. 2021. Self-supervised contrastive learning for code retrieval and summarization via semantic-preserving transformations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Jose Cambronero, Hongyu Li, Seohyun Kim, Koushik Sen, and Satish Chandra. 2019. When deep learning met code search. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Daya Guo, Shuai Lu, Nan Duan, Yanlin Wang, Ming Zhou, and Jian Yin. 2022. UniXcoder: Unified cross-modal pre-training for code representation. In *Proceedings of The Annual Meeting of the Association for Computational Linguistics (ACL)*.

Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, et al. 2020. GraphCodeBERT: Pre-training code representations with data flow. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Junjie Huang, Duyu Tang, Linjun Shou, Ming Gong, Ke Xu, Daxin Jiang, Ming Zhou, and Nan Duan. 2021. CoSQA: 20,000+ web queries for code search and question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5690–5700, Online. Association for Computational Linguistics.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. CodeSearchNet Challenge: Evaluating the state of semantic code search. *arXiv preprint*.

Diederik P Kingma and Jimmy Lei Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Wei Li, Haozhe Qin, Shuhan Yan, Beijun Shen, and Yuting Chen. 2020. Learning code-query interaction for enhancing code searches. In *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint*.

Meili Lu, Xiaobing Sun, Shaowei Wang, David Lo, and Yucong Duan. 2015. Query expansion via wordnet for effective code search. In *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, et al. 2021. CodeXGLUE: A machine learning benchmark dataset for code understanding and generation. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Fei Lv, Hongyu Zhang, Jian-guang Lou, Shaowei Wang, Dongmei Zhang, and Jianjun Zhao. 2015. CodeHow: Effective code search based on api understanding and extended boolean model. In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*.

Yu Meng, Chenyan Xiong, Payal Bajaj, Paul Bennett, Jiawei Han, Xia Song, et al. 2021. COCO-LM: Correcting and contrasting text sequences for language model pretraining. *Advances in Neural Information Processing Systems*.

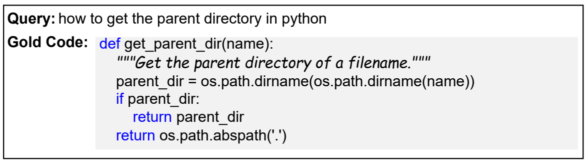
Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. In *Proceedings of The Conference and Workshop on Neural Information Processing Systems (NeurIPS)*.

A Case Study

We present case studies in the following sections for code search on CoSQA test set and code question answering on CoSQA validation set.

A.1 Code Search

The code snippets, shown in each case study, are retrieved from 6,267 candidate Python code functions.

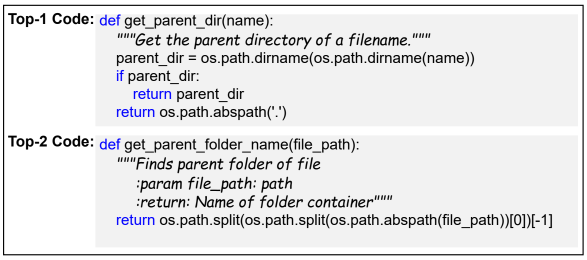


Query: how to get the parent directory in python

Gold Code:

```
def get_parent_dir(name):
    """Get the parent directory of a filename."""
    parent_dir = os.path.dirname(os.path.dirname(name))
    if parent_dir:
        return parent_dir
    return os.path.abspath('.')
```

(a) A pair of real user web query and gold code function.



Top-1 Code:

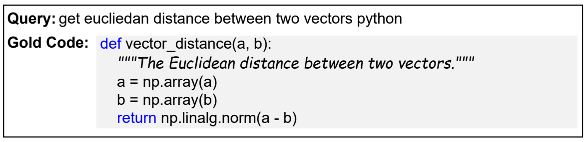
```
def get_parent_dir(name):
    """Get the parent directory of a filename."""
    parent_dir = os.path.dirname(os.path.dirname(name))
    if parent_dir:
        return parent_dir
    return os.path.abspath('.')
```

Top-2 Code:

```
def get_parent_folder_name(file_path):
    """Finds parent folder of file
        :param file_path: path
        :return: Name of folder container"""
    return os.path.split(os.path.split(os.path.abspath(file_path))[0])[-1]
```

(b) Code search results with KeyDAC (top-2 results).

Figure 7: Code search results with KeyDAC on CoSQA test set. Base model is RoBERTa.

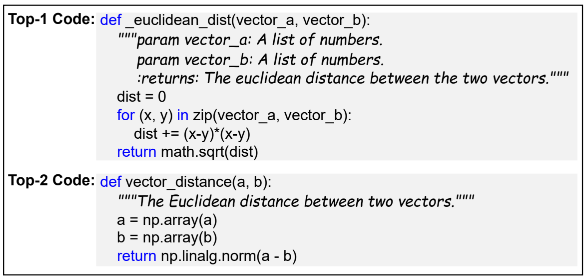


Query: get eucliedan distance between two vectors python

Gold Code:

```
def vector_distance(a, b):
    """The Euclidean distance between two vectors."""
    a = np.array(a)
    b = np.array(b)
    return np.linalg.norm(a - b)
```

(a) A pair of real user web query and gold code function.



Top-1 Code:

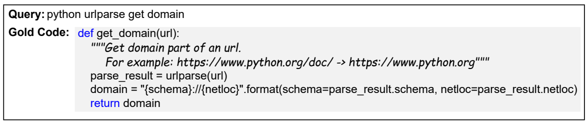
```
def _euclidean_dist(vector_a, vector_b):
    """param vector_a: A list of numbers.
        param vector_b: A list of numbers.
        :returns: The euclidean distance between the two vectors."""
    dist = 0
    for (x, y) in zip(vector_a, vector_b):
        dist += (x-y)*(x-y)
    return math.sqrt(dist)
```

Top-2 Code:

```
def vector_distance(a, b):
    """The Euclidean distance between two vectors."""
    a = np.array(a)
    b = np.array(b)
    return np.linalg.norm(a - b)
```

(b) Code search results with KeyDAC (top-2 results).

Figure 8: Code search results with KeyDAC on CoSQA test set. Base model is CodeBERT. The query has a typo *eucliedan*.

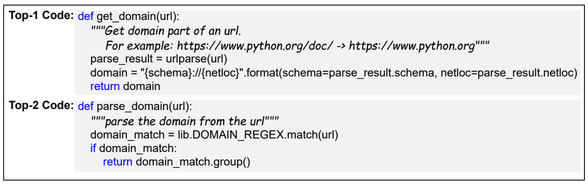


Query: python urlparse get domain

Gold Code:

```
def get_domain(url):
    """Get domain part of an url.
        For example: https://www.python.org/doc/ -> https://www.python.org"""
    parse_result = urlparse(url)
    domain = "{schema}://{netloc}".format(schema=parse_result.schema, netloc=parse_result.netloc)
    return domain
```

(a) A pair of real user web query and gold code function.



Top-1 Code:

```
def get_domain(url):
    """Get domain part of an url.
        For example: https://www.python.org/doc/ -> https://www.python.org"""
    parse_result = urlparse(url)
    domain = "{schema}://{netloc}".format(schema=parse_result.schema, netloc=parse_result.netloc)
    return domain
```

Top-2 Code:

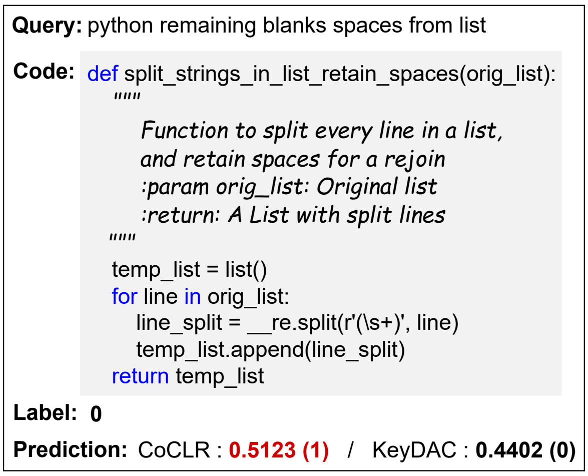
```
def parse_domain(url):
    """parse the domain from the url"""
    domain_match = lib.DOMAIN_REGEX.match(url)
    if domain_match:
        return domain_match.group()
```

(b) Code search results with KeyDAC (top-2 results).

Figure 9: Code search results with KeyDAC on CoSQA test set. Base model is UniXcoder.

A.2 Code Question Answering

We use the CoSQA validation set, since the ground-truth labels of WebQueryTest are not provided. All cases are negative query-code pairs, such that the code snippet does not match the search intent of the query. The models predict a pair of query and code as a negative pair when the cosine similarity is lower than the threshold 0.5.



Query: python remaining blanks spaces from list

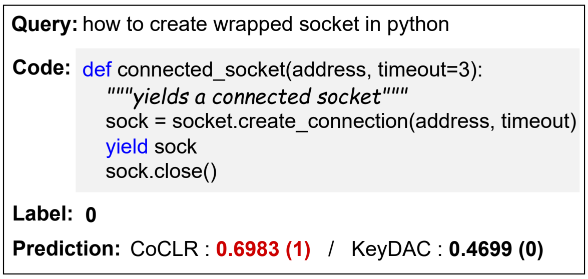
Code:

```
def split_strings_in_list_retain_spaces(orig_list):
    """
        Function to split every line in a list,
        and retain spaces for a rejoin
        :param orig_list: Original list
        :return: A List with split lines
    """
    temp_list = list()
    for line in orig_list:
        line_split = __re.split(r'(\s+)', line)
        temp_list.append(line_split)
    return temp_list
```

Label: **0**

Prediction: CoCLR : **0.5123 (1)** / KeyDAC : **0.4402 (0)**

Figure 10: Case study for code question answering task on CoSQA validation set. Base model is RoBERTa.



Query: how to create wrapped socket in python

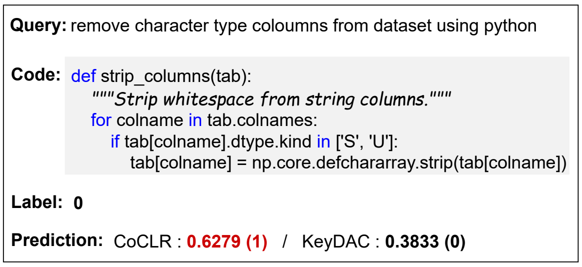
Code:

```
def connected_socket(address, timeout=3):
    """yields a connected socket"""
    sock = socket.create_connection(address, timeout)
    yield sock
    sock.close()
```

Label: **0**

Prediction: CoCLR : **0.6983 (1)** / KeyDAC : **0.4699 (0)**

Figure 11: Case study for code question answering task on CoSQA validation set. Base model is CodeBERT.



Query: remove character type coloumns from dataset using python

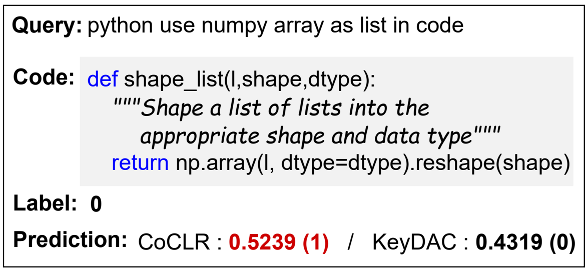
Code:

```
def strip_columns(tab):
    """Strip whitespace from string columns."""
    for colname in tab.colnames:
        if tab[colname].dtype.kind in ['S', 'U']:
            tab[colname] = np.core.defchararray.strip(tab[colname])
```

Label: **0**

Prediction: CoCLR : **0.6279 (1)** / KeyDAC : **0.3833 (0)**

Figure 12: Case study for code question answering task on CoSQA validation set. Base model is GraphCodeBERT.



Query: python use numpy array as list in code

Code:

```
def shape_list(l,shape,dtype):
    """Shape a list of lists into the
       appropriate shape and data type"""
    return np.array(l, dtype=dtype).reshape(shape)
```

Label: **0**

Prediction: CoCLR : **0.5239 (1)** / KeyDAC : **0.4319 (0)**

Figure 13: Case study for code question answering task on CoSQA validation set. Base model is UniXcoder.

Large Scale Multi-Lingual Multi-Modal Summarization Dataset

Yash Verma*
IISER Kolkata
`yashv7523@gmail.com`

Anubhav Jangra*
Indian Institute of Technology Patna
`anubhav0603@gmail.com`

Raghvendra Kumar
Indian Institute of Technology Patna
`raghvendra.kumar1004@gmail.com`

Sriparna Saha
Indian Institute of Technology Patna
`sriparna@iitp.ac.in`

Abstract

Significant developments in techniques such as encoder-decoder models have enabled us to represent information comprising multiple modalities. This information can further enhance many downstream tasks in the field of information retrieval and natural language processing; however, improvements in multi-modal techniques and their performance evaluation require large-scale multi-modal data which offers sufficient diversity. Multi-lingual modeling for a variety of tasks like multi-modal summarization, text generation, and translation leverages information derived from high-quality multi-lingual annotated data. In this work, we present the current largest multi-lingual multi-modal summarization dataset (`M3LS`), and it consists of over a million instances of document-image pairs along with a professionally annotated multi-modal summary for each pair. It is derived from news articles published by `British Broadcasting Corporation(BBC)` over a decade and spans 20 languages, targeting diversity across five language roots, it is also the largest summarization dataset for 13 languages and consists of cross-lingual summarization data for 2 languages. We formally define the multi-lingual multi-modal summarization task utilizing our dataset and report baseline scores from various state-of-the-art summarization techniques in a multi-lingual setting. We also compare it with many similar datasets to analyze the uniqueness and difficulty of `M3LS`. [1]

1 Introduction

The world we live in today is very diverse, with over 7,000+ languages spoken across the globe[2]. These languages have varying traits and are spoken by communities of various sizes depending upon the popularity of the language. For example, Mandarin consists of over 50,000 *hanzi* (characters) and is spoken by over 1.117 billion people[3], while there exist languages like Rotokas, which is an indigenous language spoken by about 4,320 people on the island of Bougainville, Papua New Guinea, which consists of only 12 letters[4].

Table 1: Comparison of proposed dataset with existing large-scale multi-lingual and multi-modal datasets.

Multi-lingual Summarization Datasets			
Dataset Name	**Dataset Size**	**#Languages**	**Domain**
`XL-Sum` (Hasan et al., 2021)	1M	44	News
`MLSUM` (Scialom et al., 2020)	1.5M	5	News
`WikiLingua` (Ladhak et al., 2020)	770K	18	Tutorials
`MLGSum` (Wang et al., 2021)	1.1M	12	News
`M3LS` (Ours)	1.1M	20	News
Multi-modal Summarization Datasets			
Dataset Name	**Dataset Size**	**Modalities**	**Domain**
`MSMO` (Zhu et al., 2018)	314K	Text + Image	News
`E-DailyMail` (Chen and Zhuge, 2018)	219K	Text + Image	News
`How2` (Sanabria et al., 2018)	190K	Text + Video + Audio	Multiple Domains
`MMSS` (Li et al., 2018)	66K	Text + Image	News
`VMSMO` (Li et al., 2020)	185K	Text + Video + Audio	Social Network
`M3LS` (Ours)	1.1M	Text + Image	News

These languages, although very crucial, restrict people to communicate their thoughts to others who speak the same language. The gift of sight, however, is something that is universally shared by every human being on this plant, irrespective of their culture, ethnicity, or the language that they speak. Through this work we aim to instigate the research towards improving existing automatic summarization systems by leveraging information from multiple languages and visual modalities.

Various studies in the past have illustrated how unified summarization frameworks across multiple languages improve the summarization quality over mono-lingual frameworks (Wang et al., 2021). Similarly, there have been works in multi-modal summarization that illustrate how multi-modal input can help improve the quality of summarization over text summarization systems (Jangra et al., 2020a,b; Chen and Zhuge, 2018; Mukherjee et al.,

---

[*] These authors contributed equally to this work.

[1] The dataset and code used in this work are made available at `https://github.com/anubhav-jangra/M3LS`.

[2] `https://www.ethnologue.com/guides/how-many-languages`

[3] `https://www.berlitz.com/en-uy/blog/most-spoken-languages-world`

[4] `https://en.wikipedia.org/wiki/Rotokas_language`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3620–3632
May 2-6, 2023

2022). Additionally, having multiple modalities in the output summary can help improve the overall satisfaction of the user (Zhu et al., 2018; Jangra et al., 2021b). Multiple modalities can also compensate for the inability of individual modalities to express various aspects of the summary. For instance, it is hard to express abstract concepts like "freedom", "gravity", etc. through images, while it can be expressed through text conveniently. Similarly, it is very difficult to describe a "Pangolin" to someone who hasn't seen one beforehand.

Hence, in this work we propose the task of Multi-modal Multi-lingual Summarization (`M3LS`), and also release the `M3LS` dataset[5] to facilitate the research in this direction. The dataset comprises 1.1M news articles, spanning 20 languages comprising *English*, *Chinese*, *Spanish*, *Russian*, *French*, *Ukrainian*, *Portuguese*, *Japanese*, *Tamil*, *Hindi*, *Marathi*, *Gujarati*, *Bengali*, *Sinhala*, *Urdu*, *Pashto*, *Indonesian*, *Telugu*, *Punjabi*, and *Nepali*; making it the largest language-spanning summarization dataset. To the best of our knowledge, the proposed dataset is the largest summarization dataset for 13 languages (*Russian*, *Ukrainian*, *Tamil*, *Hindi*, *Marathi*, *Gujarati*, *Bengali*, *Sinhala*, *Urdu*, *Pashto*, *Telugu*, *Punjabi*, and *Nepali*).

We hope that the proposed task and the dataset will instigate and inspire multi-modal and multi-lingual research in less-explored languages for solving various tasks including but not limited to automatic summarization (Nallapati et al., 2016; See et al., 2017), article headline generation (Jin et al., 2020; Gavrilov et al., 2019; Zhang et al., 2018), keyword extraction (Showrov and Sobhan, 2019; Lee and Kim, 2008; Yao et al., 2019), image caption generation (Xu et al., 2015; Bai and An, 2018), multi-modal embedding generation (Sun et al., 2019; Lu et al., 2019; Li et al., 2019; Zhou et al., 2020), large-scale language modeling (Raffel et al., 2020; Devlin et al., 2019) etc.

The major contributions of this work are as follows - *1) We have proposed the multi-modal multi-lingual summarization (`M3LS`) task. 2) We have released the largest multi-modal summarization dataset that spans 20 languages. 3) The proposed dataset is the largest text summarization dataset for 13 languages. 4) To the best of our knowledge, we present the first ever multi-modal cross-lingual dataset (consisting of Japanese-to-English and English-to-Japanese). 5) We have provided multi-modal summarization baseline results for our dataset and a detailed analysis of the dataset.*

2 Related Work

The field of text summarization is more than 5 decades old (Edmundson, 1969), and has evolved to a great extent in recent years. Prior to the advances in sequence-to-sequence frameworks (Sutskever et al., 2014), people mainly focused on extractive summarization techniques that aim to generate summary via extracting words, phrases, or sentences (Mihalcea and Tarau, 2004; Saini et al., 2019; Alguliev et al., 2010). See et al. (2017) proposed the Pointer-Generator Networks, an attentive recurrent neural network based framework (Bahdanau et al., 2015). Recent years have seen great progress in research in automatic summarization leveraging transformer based models (Zhang et al., 2020; Devlin et al., 2019) and attention mechanism (Vaswani et al., 2017).

In this section we discuss the related works showcasing multi-modal datasets and multi-lingual datasets. A detailed size comparison of these datasets with `M3LS` is shown in Table 1.

2.1 Multi-modal summarization datasets

Multi-modal summarization is the task of summarizing content comprising two or more input modalities. The output can be uni-modal or multi-modal depending on the task. In this section, we discuss existing large-scale multi-modal summarization datasets proposed in the community. We point the readers to Jangra et al. (2021a) for a comprehensive survey.

MSMO: Zhu et al. (2018) proposed a multi-modal summarization dataset that consists of text and images. The dataset is obtained from the `DailyMail`[6] website and contains 314,581 instances in English language. However, Hasan et al. (2021) illustrated that the `DailyMail` news highlights lack novel n-grams. Fabbri et al. (2021) also highlighted the inconsistency in quality of some reference summaries in the `CNN/DailyMail` dataset (Nallapati et al., 2016).

E-Dailymail Chen and Zhuge (2018) proposed the E-Dailymail dataset, which contains text and images extracted from the `DailyMail` website. The dataset consists of 219,100 instances in English,

---

[5] A sample of our dataset is available at `https://github.com/zenquiorra/M3LS`, the complete dataset will be released in the camera ready version of the work

[6] `https://www.dailymail.co.uk/home/index.html`

containing the input document, article title, images, and image captions.

How2: Sanabria et al. (2018) proposed a multi-modal summarization dataset consisting of text, video, and audio modalities; it contains over 2000 hours of videos accompanied by the corresponding audio and speech transcriptions.

MMSS: Li et al. (2018) proposed a multi-modal summarization dataset consisting of text and images with the aim of proposing an image-aided sentence summarization framework. The dataset has 66K instances in English language, that is generated by extracting sentence-headline pairs from the `Gigaword` corpus[7].

VMSMO: To the best of our knowledge, Li et al. (2020) proposed the first large-scale asynchronous text-audio-video summarization dataset. The dataset is generated from the famous microblogging platform `Sina Weibo`[8], and comprises of 184,920 instances in Chinese language.

Similar trends of incorporating multiple modalities in language tasks can also be noticed in several tasks like question answering (Singh et al., 2021), translation (Elliott and Kádár, 2017), sentiment analysis (Soleymani et al., 2017), lexico-semantic classification (Jha et al., 2022), keyword extraction (Verma et al., 2022) etc.

2.2 Multi-lingual Text Summarization Datasets

The popularity of studying the benefits of summarization in different languages to improve summarization qualities increased over the past few years. There have been a lot of research work in bi-lingual setting; however, in this work, we limit ourselves to discussing multi-lingual summarization datasets to be concise.

MLSUM : Scialom et al. (2020) proposed the `MLSUM` dataset that consists of 1.5 million news articles obtained from the `Dailymail/CNN` websites. The dataset spans five languages - French, German, Spanish, Russian and Turkish.

XL-Sum: Hasan et al. (2021) proposed the `XL-Sum` dataset that consists of 1.35 million articles in 44 languages obtained from `BBC news`, making it the most language-diverse summarization dataset to date. However, 25 of these 44 languages do not contain even 10,000 instances, making it incompetent to train any language model.

WikiLingua: Ladhak et al. (2020) proposed the `Wikilingua` dataset, which is the largest parallel multi-lingual summarization to date. The dataset consists of 770K instances in English language, and is extended to 17 other languages for varying number of English articles.

MLGSum: Wang et al. (2021) proposed the `MLGSum` dataset that consists of articles from various news providers such as `BBC`, `france243` and `select faz`. The dataset has five high-resource and seven low-resource languages, with a total of 1.1 million instances, and is a rich source for text summarization for German language with 500K instances.

We observe that multiple popular datasets (see Table 1) in multimodal summarization and multilingual summarization are useful for both technique evaluation and technique improvisation. However, the combined field of multilingual multimodal summarization has remained largely unexplored, and it can be attributed to the lack of dedicated high quality dataset and formalizing it as a problem statement. Hence, we formally define the `M3LS` task and discuss the dataset addressing the problem further.

3 M3LS Task

Given for each language $l_k \in L$ where L is the set of all languages, we have data $M^{l_k} =< T^{l_k}, I^{l_k} >$, where $T^{l_k} = \left\{t_1^{l_k}, t_2^{l_k}, \ldots, t_{|T|}^{l_k}\right\}$ is a set of documents, and $I^{l_k} = \{I^{t_1^{l_k}}, I^{t_2^{l_k}}, \ldots, I^{t_{|T|}^{l_k}}\}$ is a set of images, where $I^{t_j^{l_k}} = \left\{i_1, i_2, \ldots, i_{|I|}\right\}^{t_j^{l_k}}$ denotes the set of images belonging to the document $t_j^{l_k} \in T^{k_k}$ and $|.|$ denotes the cardinality of a set.

The task is to obtain a function F that maps documents $t_j^{l_{k_1}} \in T^{l_{k_1}}$ in language l_{k_1} along with their corresponding images, $I^{t_j^{l_{k_1}}} \in I^{l_{k_1}}$ to a set of multi-modal summaries in target language, l_{k_2}, comprising of text summaries (denoted by $O^{l_{k_2}}$) along with images from the input (denoted by $I^{l_{k_1}}$).

$$F :< T^{l_{k_1}}, I^{l_{k_1}} > \rightarrow < O^{l_{k_2}}, I^{l_{k_1}} > \quad (1)$$

When $k_1 \neq k_2$, we have multi-modal cross-lingual summarization, otherwise the task is multi-modal mono-lingual summarization, a graphic representation of the task is shown in Figure 1.

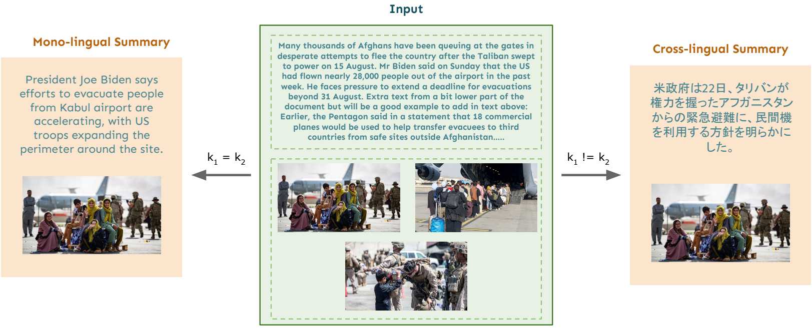
4 M3LS Dataset

Through the `M3LS` task, we motivate the need for a multi-modal multi-lingual dataset by studying the

---

[7] `https://github.com/harvardnlp/sent-summary`

[8] http://ir.weibo.com/



Many thousands of Afghans have been queuing at the gates in desperate attempts to flee the country after the Taliban swept to power on 15 August. Mr Biden said on Sunday that the US had flown nearly 28,000 people out of the airport in the past week. He faces pressure to extend a deadline for evacuations beyond 31 August. Extra text from a bit lower part of the document but will be a good example to add in text above: Earlier, the Pentagon said in a statement that 18 commercial planes would be used to help transfer evacuees to third countries from safe sites outside Afghanistan.....

President Joe Biden says efforts to evacuate people from Kabul airport are accelerating, with US troops expanding the perimeter around the site.

米政府は22日、タリバンが権力を握ったアフガニスタンからの緊急避難に、民間機を利用する方針を明らかにした。

Figure 1: Proposed `M3LS` task.

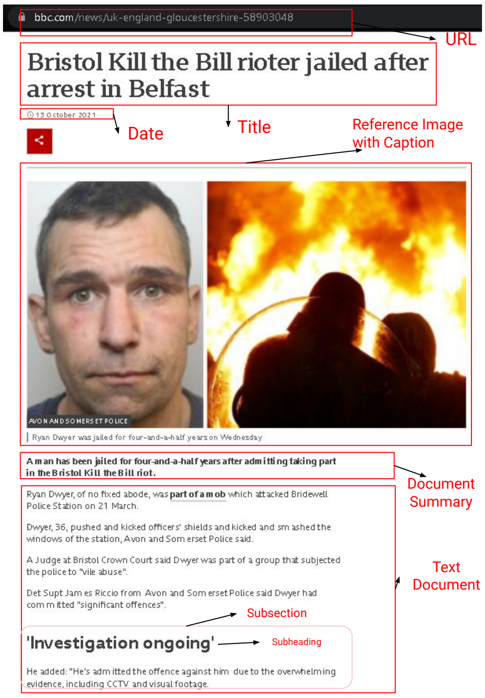
developments in summarization techniques such as secondary enhancements using images with multi-modal output (Zhu et al., 2018), video-based multi-modal summarization (Li et al., 2020) and using multiobjective optimization (Jangra et al., 2020b). On the other hand, development of multi-lingual transformer based models like Xue et al. (2020) has publicly available checkpoints fine-tuned for multiple language modelling tasks, including multi-lingual summarization.

Development of such models requires high-quality heterogeneous data and improvements in various models utilizing multi-modal shared attention layers for annotated data with image-text pairs for a specific language task. To address these issues, we present `M3LS` and in this section we discuss various steps involved in its construction.

4.1 Dataset Construction

We explore the news domain, as it is one of the most abundant and readily available domains and covers articles in multiple topics, while describing the events and lacking extreme bias. We analyzed the structure of articles and surveyed multiple news providers before finalizing on `BBC News`, which provides full sentence summaries in a uniform structured format across multiple languages. The summaries are professionally created by the author which ensures the quality of the data. We explain the steps involved in creating the `M3LS` dataset and discuss various aspects of the data.

BBC News: `BBC News`[9] is a division of British Broadcasting Corporation responsible for gathering and broadcasting current news affairs. Each BBC news article has a text summary comprising complete sentences in the present tense, avoiding opinions and sensationalism. We cover 20 different languages with summaries written in corresponding languages. We extract data from various parts of the webpage as shown in Figure 2.



bbc.com/news/uk-england-gloucestershire-58903048

Bristol Kill the Bill rioter jailed after arrest in Belfast

13 October 2021

Ryan Dwyer was jailed for four-and-a-half years on Wednesday

A man has been jailed for four-and-a-half years after admitting taking part in the Bristol Kill the Bill riot.

Ryan Dwyer, of no fixed abode, was **part of a mob** which attacked Bridewell Police Station on 21 March.

Dwyer, 36, pushed and kicked officers' shields and kicked and smashed the windows of the station, Avon and Somerset Police said.

A Judge at Bristol Crown Court said Dwyer was part of a group that subjected the police to "vile abuse".

Det Supt James Riccio from Avon and Somerset Police said Dwyer had committed "significant offences".

'Investigation ongoing'

He added: "He's admitted the offence against him due to the overwhelming evidence, including CCTV and visual footage.

Figure 2: Snapshot format of a webpage used in development of M3LS, and various features extracted during the scraping procedure

Obtaining Articles: We obtain links to articles from the corresponding `Twitter`[10] pages for each BBC language news dataset. To extend the dataset, we scrape[11] valid links[12] obtained from the parsed articles of each language.

The final collection of links is scraped separately using `scrapy`[13] to obtain the final dataset. Since these links are showcased at the corresponding `twitter` page, these links ensure articles con-

---

[9] `https://www.bbc.com/news`

[10] `https://twitter.com/bbc`

[11] Data is collected in accordance with the terms and conditions mentioned on the website

[12] A link is valid if it contains a BBC article summary for the corresponding domain.

[13] `https://scrapy.org`

taining topics of interest and high popularity. We extend the dataset by recursively extracting links from suggestions and hyperlinks within a webpage.

Structuring the Data: We obtain various features from the webpage as shown in Figure (2), and compile them in a `JSON` format; we also provide a dedicated parser, instructions and a tutorial for the ease of access of features from any instance. The data is freely available for use in accordance with the terms and conditions of `BBC News`, we discuss this in detail on the same link where our dataset is uploaded.

Text Validation: In order to ensure high-quality text from the source, we manually read 10 instances from each language[14] from the collected links to verify if the articles are descriptive in nature and consist of text written in complete sentences.

Summary Validation: We manually checked the summary quality for 100 articles each in 4 languages[15] from our dataset and validated if the given summary captures the information represented in the text. For every article, after carefully reading the text, we assign the gold summary a score between 1-5 with 5 being the best possible summary which captures most of the important information from the given article and vice versa, and also including parameters like summary length and length of the article. We observe that for > 70 articles across the languages evaluated obtain a score of > 4 out of 5 in our analysis. Assuming the uniformity of articles published by BBC across multiple domains, we assume that this fact is true for every language in our dataset.

Final Dataset: In final dataset, each news article contains the text document, images with corresponding captions, keywords, links to related news articles, and a multi-modal summary comprising of a few sentences and an image.

Cross-lingual Dataset: Our cross-lingual dataset contains all features from our final dataset, along with multi-modal summaries consisting of text in another language. It is obtained from the links given by the author within the Japanese language article to the corresponding article in English language, we manually check the information provided in both articles using `Google Translate`[16] for 100 instances to verify the similarity of the content and summaries provided.

Train-Test-Validation split: The dataset has 1.2 million news articles which we split into 80% training, 10% test and 10% validation for languages having $\leq$ 50,000 articles, otherwise we select 90% data for training, 5% for testing and 5% as validation split.

5 Dataset Analysis

5.1 Overview

The `M3LS` dataset has 1.11M+ multi-lingual multi-modal instances across 20 languages and over 9K cross-lingual multi-modal instances for `English-Japanese` language pair. The dataset can be categorized into 8 high resource languages and 12 low resource languages[17] (refer to Appendix B for more details). The chosen languages originate from different parts of the globe, and belong to 5 different language roots: *Indo-European*, *Austronesian*, *Japanic*, *Dravidian*, and *Sino-Tibetan*.

`M3LS` dataset is quite diverse, with the least #articles for `Sinhala` (10,148) and greatest #articles for `English` (376,367). The dataset becomes even more complex and challenging with different sizes of input documents for different languages, with document size varying from $\tilde{3}30$ tokens to over 2800+ tokens. The dataset articles cover a wide time span, with articles from 2009 to 2021 (refer to Appendix A for more details).

We hope that the `M3LS` dataset will instigate and inspire research in less-explored languages, since 14 out of these 20 languages covered by the dataset are among the top-20 most spoken languages in the world[18]; this diversity helps in modelling tasks for both well-explored and less-explored languages.

5.2 Dataset Comparison

To study the size and span of our dataset, we compare `M3LS` with other summarization datasets extracted from the `BBC News` domain. We found that `XSum` contains 53% of the tokens from our dataset, while `XL-Sum` contains 58% of the tokens from our dataset across all languages present in `M3LS`. However, they are uni-modal in nature, while `XSum` is uni-lingual. We observe that `M3LS` is magnitudes larger when compared to `XSum`, while ex-

---

[14] For languages unknown to the authors, we use `Google Translate` `https://translate.google.com` to translate the content in English language

[15] We restrict ourselves to 4 languages (`English`, `Hindi`, `Bengali` and `Marathi`) due to the understanding of languages of the authors presenting this work

[16] `https://google.com/translate`

[17] The categorization is done based on a threshold value of 50k data instances.

[18] `https://lingua.edu/the%2D20%2Dmost%2Dspoken%2Dlanguages%2Din%2Dthe%2Dworld%2Din%2D2022/`

ceeding by times 2-3 in almost all individual language instances when compared to `XL-Sum`. Both of these datasets are used to train and fine-tune several state-of-the-art-summarization models like `Pegasus`, hence we believe that `M3LS` will offer a wider and better language modelling support in terms of size and diversity for the languages present in it, with the additional benefit of multi-modality.

6 Experiments

6.1 Setup

Depending upon the number of instances in each language within `M3LS`, we perform a train:test:validation split with a ratio of 80:10:10 if the number of instances is below 50K and 90:5:5 otherwise. To conduct our experiments in a multi-lingual setting, we survey publicly available tokenizers and sentence segmenters for multiple languages, and we combine them within one dedicated package for our experiments. We further define a set of rules for sentence segmentation for languages lacking such support from external packages within our package[19].

We compile our package using `segtok`[20] for the Indo-European language, `IndicNLP`[21] for Indian languages, `fugashi` (McCann, 2020) for Japanese (`ja`) and `chinese`[22] for Chinese.

For data pre-processing steps such as stopword removal, we collect stopwords from the `nltk`[23] package, and publicly available stopwords present in the `spaCy`[24] repository for all languages in a centralized pipeline for our experiments.

We evaluated the performance of various summarization techniques utilizing our dataset, including simpler techniques such as `LEAD-3` and `RANDOM` which have proven to be quite useful in past (Ghalandari et al., 2020; Scialom et al., 2020; Sharma et al., 2019). We have also included statistics based `CENTROID` (Radev et al., 2004) and graph based `TextRank` (Mihalcea and Tarau, 2004) techniques.

To have a fair comparison across multiple languages using a shared dedicated model, we have evaluated the performance of an abstractive technique in a multi-lingual setting utilizing a pre-trained checkpoint[25] for summarization of the transformer-based `MT5` (Xue et al., 2020) model. Finally to explore the multi-modal aspect of our dataset, we evaluate the performance of a multi-modal encoder-decoder based technique (Zhu et al., 2018) that utilizes images and text to generate a multi-modal text summary. However the publicly available implementation[26] for `MSMO` restricts us to evaluate it only for the English language. However, to compare this score, we evaluate the performance of three state-of-the-art transformer-based models - `Pegasus` (Zhang et al., 2020), `BART` (Lewis et al., 2020), and `T5` (Xue et al., 2020) for summarization which are compatible with the English language.

Since, two of the pre-trained models we described above are fine-tuned on `XSum` and `XL-Sum` datasets which are extracted from the same source - `BBC News` - we avoid fine-tuning on models to have a fair comparison of the models and we explain the scores in discussions.

In all techniques, we set the generated summary length threshold as the average length of gold summary for the corresponding language in our corpus.

6.2 Baselines

Simpler Extractive Approaches

LEAD-3: In this baseline, the first three sentences of the source text are extracted as the final summary. This method is a robust baseline, as shown by (Sharma et al., 2019) for news summarization datasets.

RANDOM: We recursively extract words randomly from the source text until the threshold summary length is reached. The aim of this baseline is to understand and compare other baselines with an unbiased model as a point of reference.

Statistical Approach

CENTROID: We use the strategy proposed by Radev et al. (2004), which ranks sentences based on the centrality scores obtained by the words in the sentence. We use TF-IDF scores to measure each word's similarity, and extract top sentences from each ranking until the threshold summary length is obtained.

Graph Based Approach

TEXTRANK: TextRank (Mihalcea and Tarau,

---

[19] `https://github.com/zenquiorra/TokSeg`

[20] `https://pypi.org/project/segtok/1.1.0/`

[21] `https://github.com/anoopkunchukuttan/indic_nlp_library`

[22] `https://pypi.org/project/chinese/`

[23] `https://nltk.org/`

[24] `https://github.com/explosion/spaCy/tree/master/spacy/lang`

[25] `https://huggingface.co/csebuetnlp/mT5_multilingual_XLSum`

[26] We use the implementation provided by the authors, which is a multi-layered package, modification of which to be compatible for a multi-lingual setting isn't feasible based on the software complexity

2004) is an unsupervised graph-based ranking technique based on the relevance of sentences in the source text[27] We consider the sentences which are most central to the document based on the ranking as generated summaries.

RNN Based Approach

MSMO: MSMO (Zhu et al., 2018) is an encoder-decoder model trained for multi-modal summarization. It utilizes a multi-modal attention mechanism to generate multi-modal summaries utilizing text and images.

Transformer Based Approaches

MT5: MT5 (Xue et al., 2020) is a transformer-based seq2seq model pretrained for multiple natural language tasks. We use the publicly available checkpoint[28] pre-trained for text summarization on the `XL-Sum` dataset (Hasan et al., 2021) for a multi-lingual setting.

PEGASUS: Pegasus (Zhang et al., 2020) is a transformer-based model, pre-trained on a task to remove meaningful sentences from an input text, making it suitable for summarization. We used a checkpoint[29] of `PEGASUS` model pre-trained on the XSum dataset (Narayan et al., 2018) for summarization.

BART: BART (Lewis et al., 2020) uses a standard seq2seq architecture with a bi-directional encoder and a left-to-right decoder. We use a pre-trained model trained on the DailyMail/CNN (Nallapati et al., 2016) for our evaluation.

T5: T5 (Raffel et al., 2020) is an encoder-decoder model trained on a mixture of natural language tasks, including translation and summarization; it converts any task into a text-to-text format. We use the pre-trained `T5-large` model for the summarization task.

7 Results and Discussion

We evaluate the generated summaries against the gold summaries using the `ROUGE` (Lin, 2004) evaluation metric. We report the `ROUGE-1`, `ROUGE-2`, `ROUGE-L` f-scores across every baseline discussed above (Lin, 2004) (refer to Tables 1 and 3). We additionally report `BERTSCORE` for English baselines (Zhang et al., 2019) (refer to Table 1).

Table 2: Comparison of “ROUGE" fscores for summaries generated using Multi-modal baseline `MSMO` and Unimodal transformer based baselines against gold summaries from the English language dataset . “R-f1" denotes ROUGE-1 fscore, “R-f2" denotes ROUGE2 f-score, “R-fL" denotes the ROUGEL fscore, and “BrS" denotes BERTSCORE.

English	**R-f1**	**R-f2**	**R-fL**	**BrS**
`BART`	0.195	0.031	0.131	0.863
`Pegasus`	**0.389**	**0.181**	**0.321**	**0.910**
`T5`	0.197	0.0328	0.131	0.858
`MSMO`	0.217	0.046	0.158	0.851

7.1 Multi-lingual baseline scores

We observe that transformer based techniques used in our experiments perform significantly better compared to other techniques. However, for the “MT5" column, we observe very high scores and spikes of very low scores as observed in Table 3, this behavior maybe caused due to two factors:

- Relatively high scores can be attributed to the use of “MT5" checkpoint that is fine-tuned for the task of summarization on a dataset (`XL-Sum`) obtained from same source as ours.
- Very low scores for some languages can be attributed to the “ROUGE" evaluation metric which relies on token overlap[30]. Many of these languages, especially the ones with `Dravidan` and `Indo-European` origins have words which change their form significantly depending on their placement in the text and the context in which they appear, hence simple token overlap metrics show lower scores if the root form of the word isn’t considered.

We observe that `LEAD-3` performs better for the languages in which transformer-based baseline performs poorly, this can be attributed to two factors:

- As shown by Sharma et al. (2019) that `LEAD-3` performs very well for summarization tasks when we consider the news domain, suggesting the idea that top sentences capture a lot of information within a news article.
- `LEAD-3` considers top-3 sentences from the text, unlike abstractive summarization, new

---

[27] We use the implementation provided by the gensim`https://radimrehurek.com/gensim_3.8.3/summarization/summariser.html` package and modify the segmentation and tokenizer part using our dedicated package.

[28] `https://huggingface.co/csebuetnlp/mT5_multilingual_XLSum`

[29] `https://huggingface.co/google/pegasus-xsum`

[30] We are not implementing stemming of tokens during evaluation, due to the lack of support of multi-lingual stemming methods across various softwares which we have used for experimentation and to have an even comparison with the supported languages

Table 3: Performance of various techniques for summarization against the `M3LS` dataset gold summaries for every language. “Lang" refers to the language code for a language according to the ISO 639-1 standard, “R-f1" refers to the `ROUGE-1` f-scores, “R-f2" refers to the `ROUGE-2` f-scores, “R-fL" refers to the `ROUGE-L` f-scores

Base	**Random**			**LEAD-3**			**TextRank**			**CENTROID**			**MT5**		
Lang	R-f1	R-f2	R-fL	R-f1	R-f2	R-fL	R-f1	R-f2	R-fL	R-f1	R-f2	R-fL	R-f1	R-f2	R-fL
bn	0.003	0.000	0.002	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.004**	0.001	0.003
mr	0.013	0.000	0.012	0.041	0.005	0.040	0.025	0.002	0.025	0.006	0.001	0.006	**0.044**	0.005	**0.044**
gu	0.014	0.001	0.014	**0.039**	0.005	0.038	0.014	0.001	0.014	0.016	0.002	0.016	0.036	0.005	0.036
ps	0.002	0.000	0.001	0.000	0.000	0.000	0.002	0.000	0.001	0.000	0.000	0.000	**0.003**	0.000	0.001
uk	0.030	0.002	0.029	0.062	0.016	0.061	0.043	0.010	0.042	0.032	0.006	0.032	**0.094**	0.025	**0.094**
pt	0.179	0.009	0.114	0.204	0.033	0.124	0.199	0.030	0.128	0.089	0.008	0.075	**0.276**	0.085	0.193
id	0.118	0.001	0.083	0.172	0.037	0.117	0.144	0.030	0.104	0.104	0.014	0.080	**0.289**	0.115	0.233
ne	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
pa	0.012	0.000	0.012	**0.038**	0.004	**0.038**	0.014	0.001	0.014	0.010	0.002	0.010	0.026	0.000	0.026
si	0.014	0.000	0.014	0.032	0.004	0.031	0.019	0.002	0.019	0.007	0.001	0.007	**0.039**	0.018	**0.039**
ur	0.006	0.000	0.006	0.023	0.001	0.023	0.006	0.000	0.005	0.024	0.001	0.023	**0.044**	0.000	**0.044**
fr	0.168	0.007	0.107	0.206	0.043	0.126	0.177	0.033	0.115	0.164	0.024	0.110	**0.209**	0.041	0.141
ru	0.032	0.001	0.032	0.071	0.017	0.069	0.041	0.012	0.040	0.036	0.008	0.036	**0.081**	0.011	**0.081**
ja	0.069	0.001	0.068	0.126	0.012	0.120	0.084	0.007	0.081	0.063	0.004	0.062	**0.306**	0.081	0.291
te	0.010	0.000	0.009	0.023	0.001	0.023	0.011	0.000	0.011	0.008	0.001	0.008	**0.026**	0.000	**0.026**
ta	0.014	0.001	0.014	**0.034**	0.005	**0.034**	0.023	0.003	0.022	0.012	0.001	0.012	0.026	0.000	0.026
zh	0.022	0.001	0.022	**0.053**	0.008	0.051	0.042	0.005	0.041	0.025	0.003	0.025	0.125	0.042	0.118
es	0.177	0.008	0.117	0.180	0.033	0.117	0.110	0.018	0.073	0.081	0.008	0.067	**0.280**	0.084	0.202
hi	0.010	0.000	0.010	**0.018**	0.002	0.018	0.013	0.001	0.013	0.005	0.000	0.005	0.002	0.000	0.001
en	0.146	0.002	0.102	**0.175**	0.026	0.114	0.100	0.014	0.071	0.140	0.016	0.102	**0.427**	0.182	0.345

tokens or new forms of existing tokens are not present in the given article. Since it is an extractive technique, the chances of token overlap are higher and hence better “f-scores".

7.2 Multi-modal baseline scores

Due to the limitation of lack of pre-trained frameworks in a multi-modal setting for most of the languages in the dataset, we were constrained to evaluate the multi-modal technique on the `English` dataset. On comparing the “f-scores" of various uni-modal techniques with the multi-modal technique, we notice that the transformer based model `Pegasus` outperforms other techniques. This is largely attributed to the fact that the pre-trained checkpoint we have used for evaluation of summaries through the `Pegasus` model is fine-tuned on the `XSum` dataset, which has data collected from the same source as ours. We observe that for other models which are not fine-tuned on a dataset extracted from same source as ours, the multi-modal technique `MSMO` is able to outperform other techniques.

7.3 Abstractiveness of the proposed dataset

We propose an abstractive summarization dataset where the target summaries are manually written by human beings. The `M3LS` dataset demands abstractive techniques since the percentage of novel uni-grams in the dataset is quite high (refer to “abs.gold" column in Appendix B). This fact is also observed in the results from the baseline techniques. For instance, `MT5` performs consistently superior for multiple languages as observed in Table3, the abstractive baselines have thrice as good ROUGE scores as the extractive baselines.

8 Conclusion

In this work, we release a large-scale multi-modal multi-lingual summarization dataset comprising of over 1.1M+ news articles and spanning 20 languages, and motivate the problem statement of Multi-modal Multi-lingual summarization using `M3LS`. To the best of our knowledge, this is the first ever multi-modal summarization data set spanning several languages. The proposed dataset is the largest summarization dataset for 13 out of 20 languages. We have evaluated the performance of various baselines to establish the quality of the proposed dataset in both multi-modal and multi-lingual settings. Through this work, we hope to instigate research in various less-explored languages in the community for various research problems including but not limited to summarization, headline generation, keyword extraction, image caption generation, multi-modal embedding generation, etc. In future works, we plan to work on shared models which address the `M3LS` task utilizing our dataset.

Limitations

There are a few considerations to keep in mind in our work. **First**, the dataset currently has a multi-modal input, mapping to a textual summary. However, future work could involve annotating images to enhance the dataset with a multi-modal output. **Second**, the distribution for languages in the `M3LS` dataset is skewed due to the imbalanced number of articles published in BBC across languages and the late establishment of virtual print media in certain languages (as shown in Appendix A). **Third**, the current dataset uses an independent identically distributed split to create train and test sets, but more advanced techniques such as adversarial splits and likelihood splits could also be explored in future work. **Fourth**, while the current manuscript does not evaluate the dataset on both multi-modal and multi-lingual aspects simultaneously, we believe that this dataset has the potential to contribute to the development of such systems in the future.

Acknowledgements

This publication is an outcome of the R&D work undertaken in the project under the Visvesvaraya Ph.D. Scheme of Ministry of Electronics & Information Technology, Government of India, being implemented by Digital India Corporation (Formerly Media Lab Asia).

References

Rasim Alguliev, Ramiz Aliguliyev, and Makrufa Hajirahimova. 2010. Multi-document summarization model based on integer linear programming. *Intelligent Control and Automation*, 1(02):105.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Shuang Bai and Shan An. 2018. A survey on automatic image caption generation. *Neurocomputing*, 311:291–304.

Jingqiang Chen and Hai Zhuge. 2018. Abstractive text-image summarization using multi-modal attentional hierarchical rnn. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4046–4056.

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

H. P. Edmundson. 1969. New methods in automatic extracting. *J. ACM*, 16:264–285.

Desmond Elliott and Ákos Kádár. 2017. Imagination improves multimodal translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 130–141.

A. R. Fabbri, Wojciech Kryscinski, Bryan McCann, R. Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Daniil Gavrilov, Pavel Kalaidin, and Valentin Malykh. 2019. Self-attentive model for headline generation. In *European Conference on Information Retrieval*, pages 87–93. Springer.

Demian Gholipour Ghalandari, Chris Hokamp, John Glover, Georgiana Ifrim, et al. 2020. A large-scale multi-document summarization dataset from the wikipedia current events portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1302–1308.

Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Samin, Yuan-Fang Li, Yong-Bin Kang, M Sohel Rahman, and Rifat Shahriyar. 2021. Xl-sum: Large-scale multilingual abstractive summarization for 44 languages. *arXiv preprint arXiv:2106.13822*.

Anubhav Jangra, Adam Jatowt, Mohammad Hasanuzzaman, and Sriparna Saha. 2020a. Text-image-video summary generation using joint integer linear programming. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II*, pages 190–198.

Anubhav Jangra, Adam Jatowt, Sriparna Saha, and Mohammad Hasanuzzaman. 2021a. A survey on multi-modal summarization. *arXiv preprint arXiv:2109.05199*.

Anubhav Jangra, Sriparna Saha, Adam Jatowt, and Mohammad Hasanuzzaman. 2020b. Multi-modal summary generation using multi-objective optimization. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1745–1748.

Anubhav Jangra, Sriparna Saha, Adam Jatowt, and Mohammed Hasanuzzaman. 2021b. Multi-modal supplementary-complementary summarization using multi-objective optimization. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 818–828.

Prince Jha, Gaël Dias, Alexis Lechervy, Jose G Moreno, Anubhav Jangra, Sebastião Pais, and Sriparna Saha. 2022. Combining vision and language representations for patch-based identification of lexico-semantic relations. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4406–4415.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, Lisa Orii, and Peter Szolovits. 2020. Hooks in the headline: Learning to generate headlines with controlled styles. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5082–5093.

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. Wikilingua: A new benchmark dataset for multilingual abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4034–4048.

Sungjick Lee and Han-joon Kim. 2008. News keyword extraction for topic tracking. In *2008 Fourth International Conference on Networked Computing and Advanced Information Management*, volume 2, pages 554–559. IEEE.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Haoran Li, Junnan Zhu, Tianshang Liu, Jiajun Zhang, and Chengqing Zong. 2018. Multi-modal sentence summarization with modality attention and image filtering. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4152–4158. International Joint Conferences on Artificial Intelligence Organization.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Mingzhe Li, Xiuying Chen, Shen Gao, Zhangming Chan, Dongyan Zhao, and Rui Yan. 2020. Vmsmo: Learning to generate multimodal summary for video-based news articles. *arXiv preprint arXiv:2010.05406*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Paul McCann. 2020. fugashi, a tool for tokenizing Japanese in python. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 44–51, Online. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Sourajit Mukherjee, Anubhav Jangra, Sriparna Saha, and Adam Jatowt. 2022. Topic-aware multimodal summarization. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 387–398, Online only. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, C. D. Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *CoNLL*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Naveen Saini, Sriparna Saha, Anubhav Jangra, and Pushpak Bhattacharyya. 2019. Extractive single document summarization using multi-objective optimization: Exploring self-organized differential evolution, grey wolf optimizer and water cycle algorithm. *Knowledge-Based Systems*, 164:45–67.

Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. 2018. How2: A large-scale dataset for multimodal language understanding. In *NeurIPS*.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. Mlsum: The multilingual summarization corpus. *arXiv preprint arXiv:2004.14900*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Eva Sharma, Chen Li, and Lu Wang. 2019. Bigpatent: A large-scale dataset for abstractive and coherent summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2204–2213.

Md Imran Hossain Showrov and Masrur Sobhan. 2019. Keyword extraction from bengali news. In *2019 5th International Conference on Advances in Electrical Engineering (ICAEE)*, pages 658–662. IEEE.

Hrituraj Singh, Anshul Nasery, Denil Mehta, Aishwarya Agarwal, Jatin Lamba, and Balaji Vasan Srinivasan. 2021. Mimoqa: Multimodal input multimodal output question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5317–5332.

Mohammad Soleymani, David Garcia, Brendan Jou, Björn Schuller, Shih-Fu Chang, and Maja Pantic. 2017. A survey of multimodal sentiment analysis. *Image and Vision Computing*, 65:3–14.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yash Verma, Anubhav Jangra, Sriparna Saha, Adam Jatowt, and Dwaipayan Roy. 2022. Maked: Multilingual automatic keyword extraction dataset. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6170–6179.

Danqing Wang, Jiaze Chen, Hao Zhou, Xipeng Qiu, and Lei Li. 2021. Contrastive aligned joint learning for multilingual summarization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2739–2750.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Lu Yao, Zhang Pengzhou, and Zhang Chi. 2019. Research on news keyword extraction technology based on tf-idf and textrank. In *2019 IEEE/ACIS 18th International Conference on Computer and Information Science (ICIS)*, pages 452–455. IEEE.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, Jun Xu, Huanhuan Cao, and Xueqi Cheng. 2018. Question headline generation for news articles. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 617–626.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and vqa. In *AAAI*, pages 13041–13049.

Junnan Zhu, Haoran Li, Tianshang Liu, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2018. Msmo: Multimodal summarization with multimodal output. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4154–4164.

A Frequency of number of articles present in the `M3LS` dataset for a given year

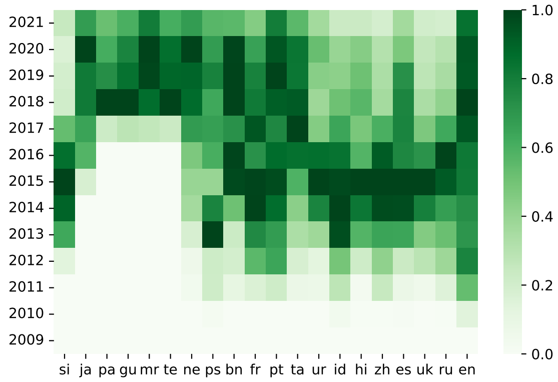


Figure 3: Temporal span of each language in the `M3LS` dataset. Darker colors correspond to a score of 1.0 and vice versa, which indicates a higher frequency of the number of articles published during the year for that language.

B Detailed Statistics of `M3LS` dataset

The detailed statistics of the curated `M3LS` dataset are presented in Table 4.

Table 4: “Lang." represents the language code from the IS0 639-1 standard, **articles** represents the number of articles for every language in the corpus, **toks.** represents the average number of tokens in an article within the corpus for the language, **tok.uni.** represents the average number of unique tokens in any article for the language, **sum.tok.** represents the average number of tokens in summary of an article for the language, **sum.uni** represents the average number of unique tokens within the summary of an article for the language, **sent.** represents the average number of sentences in a given article for the language, **abs.gold** represents the average percent of tokens in summary which are absent from the article for a given language, **images** represents the average number of images in an article for the given language, **i.c.r** represents the average ratio of the number of images consisting of a caption attached to them against the number of images which do not have a caption with them.

Lang.	**articles**	**tok.**	**tok.uni.**	**sum.tok.**	**sum.uni.**	**sent.**	**abs.gold**	**images**	**i.c.r**
bn	25283	464.78	254.73	23.12	21.97	28.42	43.18	2.44	0.58
mr	16161	871.36	404.82	27.88	25.02	63.32	48.33	0.00	0.00
gu	12175	868.75	404.84	25.87	23.63	50.11	40.87	4.94	0.30
ps	23205	523.35	207.95	32.85	27.52	18.29	33.32	2.10	0.49
uk	90846	471.90	216.57	24.59	22.78	23.49	54.07	1.56	0.22
pt	39454	2855.68	424.14	38.57	33.01	114.22	35.23	3.34	0.64
id	56108	587.88	225.12	24.59	22.89	28.54	37.07	2.47	0.57
ne	18953	402.17	229.71	21.50	20.95	23.79	45.00	2.07	0.24
pa	11600	843.74	319.75	30.96	27.35	38.87	30.42	4.57	0.37
si	10148	331.55	186.26	24.30	23.43	15.53	51.33	1.67	0.34
ur	55107	690.47	264.57	35.95	31.00	1.07	27.94	2.35	0.43
fr	25923	413.45	179.22	31.22	27.26	14.43	40.75	1.64	0.47
ru	95345	668.61	302.63	28.38	25.74	26.68	52.80	1.96	0.38
ja	11023	1052.94	282.58	48.59	36.97	33.92	29.86	2.90	0.56
te	15511	626.24	353.70	23.71	21.39	51.72	52.19	4.41	0.27
ta	38523	354.35	209.30	21.18	19.91	23.39	56.34	2.55	0.27
zh	60830	787.68	309.11	34.38	29.80	29.87	37.21	2.00	0.48
es	66816	2649.57	345.35	30.64	26.28	83.22	38.43	4.15	0.66
hi	61852	776.39	265.10	29.45	25.70	37.21	32.00	1.24	0.0
en	376367	657.95	268.60	25.27	23.40	22.28	32.86	1.39	0.35

External Knowledge Acquisition for End-to-End Document-Oriented Dialog Systems

Tuan M. Lai[2]*, **Giuseppe Castellucci**[1], **Saar Kuzi**[1], **Heng Ji**[1], **Oleg Rokhlenko**[1]
[1]Amazon
[2]University of Illinois Urbana-Champaign
`{giusecas, skuzi, jihj, olegro}@amazon.com`
`tuanml2@illinois.edu`

Abstract

End-to-end neural models for conversational AI often assume that a response can be generated by considering only the knowledge acquired by the model during training. Document-oriented conversational models make a similar assumption by conditioning the input on the document and assuming that any other knowledge is captured in the model's weights. However, a conversation may refer to external knowledge sources. In this work, we present `EKo-DoC`, an architecture for document-oriented conversations with access to external knowledge: we assume that a conversation is centered around a topic document and that external knowledge is needed to produce responses. `EKo-DoC` includes a dense passage retriever, a re-ranker, and a response generation model. We train the model end-to-end by using silver labels for the retrieval and re-ranking components that we automatically acquire from the attention signals of the response generation model. We demonstrate with automatic and human evaluations that incorporating external knowledge improves response generation in document-oriented conversations. Our architecture achieves new state-of-the-art results on the Wizard of Wikipedia dataset, outperforming a competitive baseline by 10.3% in Recall@1 and 7.4% in ROUGE-L.

1 Introduction

Recent advances in Natural Language Processing (NLP) allowed us to build complex conversational systems for various scenarios, such as task-oriented (Radlinski and Craswell, 2017; Wen et al., 2017), chit-chat (Khatri et al., 2018; Zhou et al., 2020), and even for guiding users in performing complex real-world tasks (Gottardi et al., 2022). When using deep neural networks for implementing a conversational agent, a common practice is to use many historical conversations to train the model to produce responses that are related to a dialog context. In this setting, a response is generated solely from the knowledge acquired during the training of the network (Vinyals and Le, 2015), and no other knowledge sources are used at inference time.

A more effective way of producing dialog responses would be to incorporate external knowledge into the model. This is, for example, the case of systems that make use of document/passage retrieval in QA (Lewis et al., 2020b). In this work, we consider the setting in which a conversation is grounded in a target topic but also in external knowledge, in the form of documents. For example, Figure 1 shows a conversation with respect to a target topic (i.e., *San Diego Comic-Con*) represented by a topic document. During the conversation, some turns may refer to other documents from an external Knowledge Base (KB) providing additional information (e.g., *Shel Dorf*). In our preliminary studies, we estimated that about 36% of the dialogs in the popular Wizard of Wikipedia dataset (Dinan et al., 2019) require knowledge beyond the topic document.

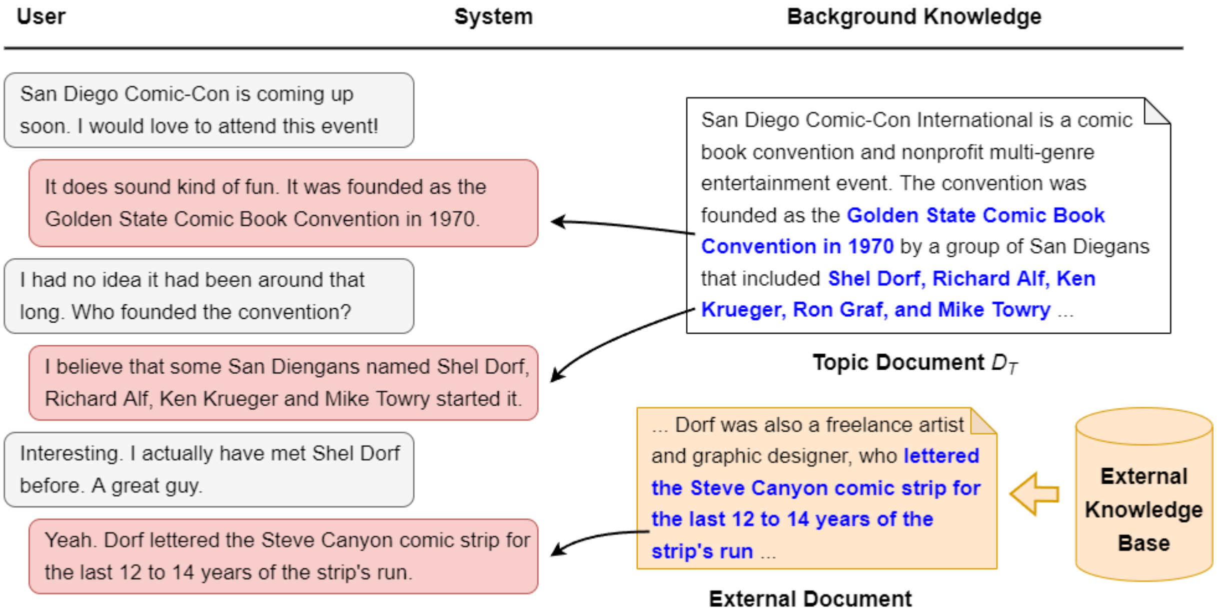
This setting poses additional challenges to the generation of adequate responses in a dialog. While a model could possibly memorize a vast amount of knowledge in its weights during training, the model will likely be applied to new dialog contexts that refer to unseen knowledge. If enriched with external knowledge, the model input could be better conditioned to produce accurate outputs. From a technical perspective, the model needs to learn i) to retrieve the relevant knowledge and ii) to incorporate it into the generated response.

To address these challenges, we propose `EKo-DoC`, an end-to-end conversational agent designed to model a target topic document and a set of external knowledge documents. `EKo-DoC` integrates a Dense Passage Retriever (DPR) (Karpukhin et al., 2020) and a Re-Ranker (RR) into a response generation model. To reduce the need

---

* Work done during internship at Amazon.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3633–3647
May 2-6, 2023



San Diego Comic-Con is coming up soon. I would love to attend this event!

It does sound kind of fun. It was founded as the Golden State Comic Book Convention in 1970.

I had no idea it had been around that long. Who founded the convention?

I believe that some San Diengans named Shel Dorf, Richard Alf, Ken Krueger and Mike Towry started it.

Interesting. I actually have met Shel Dorf before. A great guy.

Yeah. Dorf lettered the Steve Canyon comic strip for the last 12 to 14 years of the strip's run.

San Diego Comic-Con International is a comic book convention and nonprofit multi-genre entertainment event. The convention was founded as the **Golden State Comic Book Convention in 1970** by a group of San Diegans that included **Shel Dorf, Richard Alf, Ken Krueger, Ron Graf, and Mike Towry** ...

... Dorf was also a freelance artist and graphic designer, who **lettered the Steve Canyon comic strip for the last 12 to 14 years of the strip's run** ...

Figure 1: An example of a dialog in our setting: a central topic document defines the general topic of the dialog. External documents can be used as well to generate responses for some specific turns of the conversation.

for manual annotations, we automatically acquire labels for the training of the DPR and RR components. In particular, we leverage the attention weights of the generation model over the retrieved documents to generate a set of silver labels. Our generation model is a sequence-to-sequence model that generates a response by taking as input the entire concatenation of the dialog context and related documents retrieved from a KB.

Experimental results demonstrate that `EKo-DoC` is able to correctly use both the topic document and the external knowledge to produce better responses in a conversation. We report competitive results on two public datasets, Wizard of Wikipedia (WoW) (Dinan et al., 2019) and MultiDoc2Dial (Feng et al., 2021). In particular, `EKo-DoC` achieves new state-of-the-art (SOTA) results on WoW, outperforming a strong baseline by 10.3% in Recall@1 and 7.4% in ROUGE-L. Finally, we conduct a human study to verify the quality of the generated responses. Compared to models that do not condition on the topic document or external knowledge, our model produces responses that are more fluent, more on-topic, and more interesting.

To summarize, our contributions are: i) we study the setting where we model *both* a target topic document and external knowledge for a conversation; ii) we propose `EKo-DoC`, a novel end-to-end response generation architecture for the task; iii) we propose an automatic annotation procedure to acquire the labels for training the retrieval engine.

2 `EKo-DoC` Architecture

Given a dialog context C and a topic document D_T that serves as the background of the dialog, the task is to generate a response r to the last utterance of C. We also have available an external knowledge base $K_{\text{KB}} = \{D_1, D_2, ..., D_m\}$; D_j denotes a knowledge snippet represented by some natural language text (e.g., a paragraph in Wikipedia). While we use the term "snippet", each snippet is not necessarily a short piece of text. `EKo-DoC` is general enough to be used with documents of different lengths.

The dialog is mostly centered around D_T, but external knowledge from K_{KB} is sometimes needed to generate an informative and relevant response (see Figure 1 for an example). To this end, the task can be viewed as building a model of $P(r|C, D_T, K_{\text{KB}})$.

2.1 Document-Oriented Dialog System with Access to External Knowledge

Figure 2 shows an overview of `EKo-DoC`, our proposed framework. The inference process consists of three steps. First, given a dialog context and a topic document, we use a DPR model to retrieve an initial set of knowledge snippets (Section 2.1.1). Second, we re-rank the snippets using a Transformer-based cross-attention model (Section 2.1.2). Third, conditioned on the dialog context and

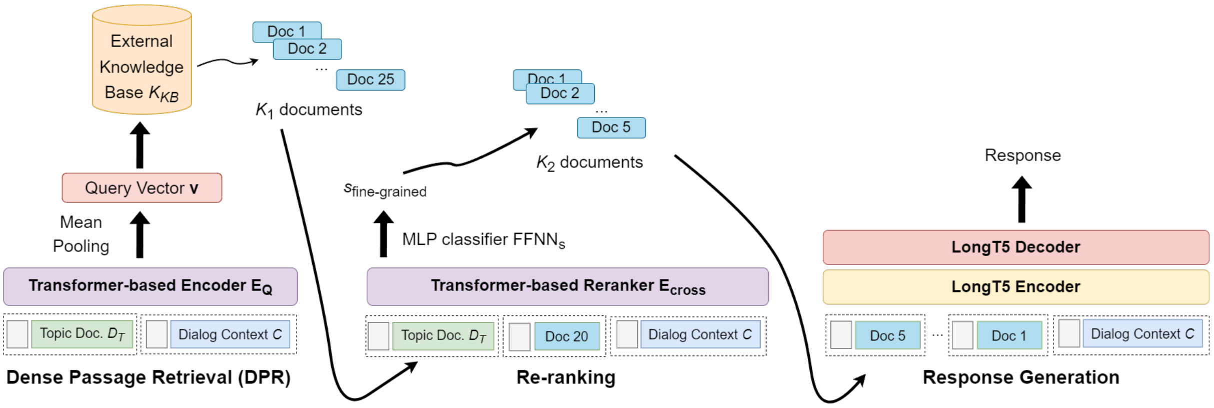


Figure 2: An overview of `EKo-DoC`, our framework. K_1 and K_2 are empirically set to be 25 and 5 (respectively).

the top-k ranked snippets, a generator produces a natural language response (Section 2.1.3).

In previous work (Glass et al., 2022; Li et al., 2022a), the retrieval components were trained using human-annotated pairs of input contexts and supporting knowledge snippets. To remove the need for expensive gold-standard retrieval labels, we use attention scores as pseudo-labels for training the retrieval and re-ranking models (Section 2.2).

Finally, note that while D_T can be fundamental to the response generation for most of the conversation, it can be less relevant for some dialog turns (e.g., see Figure 1). Therefore, we also include D_T in the knowledge base K_{KB} and let our retrieval engine decide if the topic document is essential. If D_T is retrieved, the final generator will use D_T for response generation.

2.1.1 External Knowledge Retrieval

In order to use a DPR model for retrieving knowledge, we need to encode K_{KB} in the indexing phase and the input query in the retrieval phase.

K_{KB} Encoding. We use an encoder $E_{\text{KB}}(\cdot)$ to map every knowledge snippet D_j to a real-valued vector. We assume that each snippet is a document with a title and short textual content. The input to the document encoder $E_{\text{KB}}(\cdot)$ is:

$$T\left[D\right] = \texttt{[s]}\ \text{Title / Description}\ \texttt{[/s]} \qquad (1)$$

where `[s]` and `[/s]` are special tokens denoting the start and end of the input, and "/" is used to separate the title from the description. $T[\cdot]$ denotes a function that maps a general object (e.g., a knowledge snippet) into a textual sequence.

We use a pre-trained Transformer model as the encoder (Reimers and Gurevych, 2019). That is, for each $D_j \in K_{\text{KB}}$, we encode it into a vector $\mathbf{v}_j$:

$$\mathbf{v}_j = E_{\text{KB}}\left(T\left[D_j\right]\right) \qquad (2)$$

Query Encoding. We use the topic document D_{T} and the dialog context C to construct a query for the DPR model. We first concatenate D_{T} and C into a single sequence $T\left[D_{\text{T}} \oplus C\right]$:

$$\texttt{[s]}\ T\left[D_{\text{T}}\right]\ \texttt{[sep]}\ T\left[C\right]\ \texttt{[/s]} \qquad (3)$$

where `[sep]` is a special token to separate the topic document and the dialog context. $T\left[D_{\text{T}}\right]$ is similar to what described in Equation 1. $T\left[C\right]$ is instead a concatenation of all the dialog turns, where two turns are separated by the `[sep]` token.

Then, we also use a Transformer encoder $E_{\text{Q}}(\cdot)$ to map C and D_{T} into a single vector $\mathbf{v}$:

$$\mathbf{v} = E_{\text{Q}}\left(T\left[D_{\text{T}} \oplus C\right]\right) \qquad (4)$$

In this work, $E_Q(\cdot)$ and $E_{KB}(\cdot)$ share the same architecture (Reimers and Gurevych, 2019).

Scoring and Retrieval. The first (coarse-grained) relevance score of a knowledge snippet D_j with respect to a query is given by:

$$s_{\text{c}}\left(D_j, C, D_{\text{T}}\right) = \mathbf{v} \cdot \mathbf{v}_j \qquad (5)$$

We retrieve up to K_1 knowledge snippets by ranking according to the s_c values. K_1 is a hyper-parameter, with $K_1 \ll |K_{\text{KB}}|$. In order to make this operation efficient, we adopted FAISS (Johnson et al., 2021). We first apply $E_{\text{KB}}(\cdot)$ to all the knowledge snippets and index them using FAISS offline. Then, given a query vector $\mathbf{v}$ obtained with $E_Q(\cdot)$, we use FAISS to return the top K_1 candidates according to the coarse-grained scores.

2.1.2 Knowledge Snippet Re-ranking

After the initial coarse-grained retrieval step with DPR, there are K_1 candidate knowledge snippets. We then apply a fine-grained cross-attention re-ranker over the K_1 snippets. After that, we keep the top-K_2 snippets[1] for the next phase (i.e., the response generation phase).

The input to the re-ranker is the concatenation of D_{T}, a knowledge snippet D_j retrieved in the previous phase, and C:

$$\texttt{[s]}\ T\big[D_{\text{T}}\big]\ \texttt{[sep]}\ T\big[D_j\big]\ \texttt{[sep]}\ T\big[C\big]\ \texttt{[/s]} \tag{6}$$

The re-ranker consists of a Transformer-based encoder and a feed-forward neural network. Given an input representation described above, the re-ranker computes the final relevance score s_f as:

$$\begin{aligned} \mathbf{h}_j &= \text{reduce}\Big(E_{\text{cross}}\Big(T\big[D_{\text{T}} \oplus D_j \oplus C\big]\Big)\Big) \\ s_{\text{f}}\Big(D_j, C, D_{\text{T}}\Big) &= \text{FFNN}_s\big(\mathbf{h}_j\big) \end{aligned} \tag{7}$$

where $E_{\text{cross}}(\cdot)$ is a Transformer-based encoder (Liu et al., 2019), and its input is a representation described in Equation 6; FFNN_s is a feed-forward neural network. $\text{reduce}(\cdot)$ is a function that returns the final hidden state of the Transformer that corresponds to the first input token. The final set of K_2 knowledge snippets is selected according to the s_{f} scores assigned from the re-ranker.

2.1.3 Response Generation Model

Our generative model is based on the sequence-to-sequence (seq2seq) encoder-decoder architecture that directly predicts an output sequence from an input sequence (Sutskever et al., 2014). By leveraging the recent advances in neural models for long sequences (Beltagy et al., 2020; Guo et al., 2022), our generator takes as input the entire concatenation of the dialog context and the top-ranked retrieved knowledge snippets (Figure 2). We use LongT5 as our generator (Guo et al., 2022) as it can scale up to 16K input length.

Specifically, let $\{\tilde{D}_1, \tilde{D}_2, ..., \tilde{D}_{K_2}\}$ be the set of knowledge snippets retrieved by the re-ranker.[2] The input to the generative model is:

$$\texttt{[s]}\ T\big[\tilde{D}_1 \oplus \tilde{D}_2 \oplus ... \oplus \tilde{D}_{K_2}\big]\ \texttt{[sep]}\ T\big[C\big]\ \texttt{[/s]} \tag{8}$$

where $T\big[\tilde{D}_1 \oplus \tilde{D}_2 \oplus ... \oplus \tilde{D}_{K_2}\big]$ is the concatenation of all the retrieved snippets:

$$T\big[\tilde{D}_1\big]\ \texttt{[sep]}T\big[\tilde{D}_2\big]\ \texttt{[sep]}...\ \texttt{[sep]}T\big[\tilde{D}_{K_2}\big] \tag{9}$$

We denote the final input sequence as X. The expected output is an appropriate response Y to the last dialog turn. Similar to previous encoder-decoder language models (Lewis et al., 2020a; Raffel et al., 2020), the generator models the conditional probability of selecting a new token given all previously generated tokens when conditioned on X:

$$P(Y|X) = \prod_{i=1}^{|Y|} P\big(Y_i \,|\, Y_{1:i-1}\,, X\big) \tag{10}$$

where Y_i is the i-th token of Y, and $Y_{1:i-1}$ consists of all the tokens that come before it.

The loss function for training the generator is the usual negative log-likelihood function:

$$\mathcal{L}_{\text{generation}}(B) = -\sum_{i=1}^{|B|} \log P\big(\hat{Y}^i | X^i\big) \tag{11}$$

where B is a mini-batch consisting of $|B|$ examples, each in the form $\big(X^i, \hat{Y}^i\big)$. In addition, $\hat{Y}^i$ is the ground-truth response to the input X^i.

We refer to our approach as the Fusion-in-Input (FiI) method, as our generator directly takes as input all the potentially relevant information (i.e., the retrieved knowledge and the dialog history). This is a departure from the popular Fusion-in-Decoder (FiD) approach that encodes the retrieved snippets independently (Izacard and Grave, 2021a,b; Asai et al., 2022).

2.2 Using Cross-Attention Scores as Retrieval Pseudo-Labels

If we train our system using only the generation loss function $\mathcal{L}_{\text{generation}}(B)$, only the parameters of the generation model will be updated. In order to optimize the retrieval engine, we propose to use the generation attention scores over the snippets in the input as retrieval pseudo-labels during the training stage. The intuition is that if the decoder of the generator pays more attention to a particular retrieved knowledge snippet, it means that such a snippet is likely to be relevant.

For each snippet retrieved by the re-ranker, we average all the pre-attention scores over all the tokens of the snippet. Formally, let L denote the number of layers of the decoder, and let H be the

---

[1] Note that $K_2 < K_1$.

[2] Note that it is possible that D_T is an element of this set.

number of attention heads. For each knowledge snippet D_j, we denote the indices of its starting and ending tokens in X as s_j and e_j, respectively. The averaged attention score $\mathcal{A}(D_j)$ of D_j is computed as:

$$\mathcal{A}(D_j) = \frac{\sum_{l=1}^{L} \sum_{h=1}^{H} \sum_{x=s_j}^{e_j} \sum_{y=1}^{|\hat{Y}|} p_{l,h}(x, y)}{L \times H \times |s_j - e_j + 1| \times |\hat{Y}|} \tag{12}$$

where $\hat{Y}$ is the ground-truth response. In addition, $p_{l,h}(x, y)$ is the pre-attention score between the y-th token of $\hat{Y}$ and the x-th token of X computed by the h-th attention head of the l-th layer. Basically, we collect all the pre-attention scores that are relevant to D_j and compute their average.

We can normalize the averaged attention scores using a softmax function:

$$P_{\text{attention}}(D_j) = \frac{\exp\left(\mathcal{A}(D_j)\right)}{\sum_{i=1}^{K_2} \exp\left(\mathcal{A}(D_i)\right)} \tag{13}$$

Similarly, we can normalize the retrieval scores predicted by the DPR model and the reranker:

$$\begin{aligned} Q_{\text{coarse}}(D_j) &= \frac{\exp\left(s_{\text{c}}(D_j, C, D_T)\right)}{\sum_{i=1}^{K_2} \exp\left(s_{\text{c}}(D_i, C, D_T)\right)} \\ Q_{\text{fine}}(D_j) &= \frac{\exp\left(s_{\text{f}}(D_j, C, D_T)\right)}{\sum_{i=1}^{K_2} \exp\left(s_{\text{f}}(D_i, C, D_T)\right)} \end{aligned} \tag{14}$$

We then use the averaged attention scores of the retrieved knowledge snippets as pseudo-labels to update the retrieval engine:

$$\begin{aligned} \mathcal{L}_{\text{dpr}}(B) &= \sum_{i=1}^{|B|} D_{\text{KL}}\left(P^i_{\text{attention}} \mid\mid Q^i_{\text{coarse}}\right) \\ \mathcal{L}_{\text{reranker}}(B) &= \sum_{i=1}^{|B|} D_{\text{KL}}\left(P^i_{\text{attention}} \mid\mid Q^i_{\text{fine}}\right) \\ \mathcal{L}_{\text{retrieval}}(B) &= \mathcal{L}_{\text{dpr}}(B) + \mathcal{L}_{\text{reranker}}(B) \end{aligned} \tag{15}$$

where B is a mini-batch of examples, and D_{KL} denotes the KL divergence. Also, $P^i_{\text{attention}}$, Q^i_{coarse}, and Q^i_{fine} are the distributions computed for the i-th example of B (refer to Equations 13 and 14).

The final loss function combines the generation loss (Eq. 11) and the retrieval loss (Eq. 15):

$$\mathcal{L}_{\text{final}}(B) = \frac{1}{|B|}\left(\mathcal{L}_{\text{generation}}(B) + \mathcal{L}_{\text{retrieval}}(B)\right)$$

A related study (Izacard and Grave, 2021a) also proposed an approach to train retrieval systems without strong supervision. The work focuses on the task of question answering and uses the FiD approach for output generation. In contrast, we focus on building dialog systems and tailoring our learning approach to our newly proposed FiI approach.

To summarize, `EKo-DoC` is end-to-end trainable. The document encoder of the DPR model (Equation 1) is fixed. The query encoder (Equation 4), the re-ranker, and the generator are optimized end-to-end using the final loss function described above.

3 Experiments

3.1 Data and Experiments Setup

Datasets For the experiments, we use the following two publicly available conversational datasets.

Wizard of Wikipedia (WoW) is a dataset consisting of open-domain conversations grounded in knowledge from Wikipedia (Dinan et al., 2019). We use the version provided by the KILT benchmark (Petroni et al., 2021). The external knowledge base K_{KB} consists of about 22 million 100-word passages from Wikipedia. We consider each passage as a knowledge snippet. Each conversation in WoW is annotated with a topic, which is a Wikipedia page. As the topic Wikipedia page can be extremely long, we only use the first paragraph of the page as the topic document D_T. Finally, WoW already comes with a public train/dev/test split.

MultiDoc2Dial is a new conversational dataset that grounds dialogs in multiple documents (Feng et al., 2021). Each dialog consists of multiple segments, and two adjacent segments are grounded in different documents. K_{KB} consists of 4,283 passages across four domains (Social Security Administration, Veteran Affairs, Student-Aid, and DMV). For each dialog, we use the first paragraph of the first grounded document as D_T.

Note that the original forms of WoW and MultiDoc2Dial do not have any notion of a central topic document. Therefore, we use some heuristics (described above) to select the topic document.

Evaluation Metrics For retrieval, we compute Recall@k, which measures the fraction of times the correct knowledge snippet is found in the top-k retrieved snippets. We evaluate the text generation output based on unigram F1 score (F1) and ROUGE-L (Lin, 2004). For MultiDoc2Dial, similarly to the original study that first introduced it

(Feng et al., 2021), we also compute Exact Match (EM) (Rajpurkar et al., 2016) and SacreBLEU (BL) (Post, 2018). For WoW, we do not report EM and BL scores because the leaderboard for this dataset does not use these metrics, and we also do not have direct access to the test set used by the leaderboard.

Baselines We implemented several baselines for detailed comparison and analysis. The first set of baselines consists of basic seq2seq models that do not have any retrieval engine:

- LongT5 (Guo et al., 2022) is an extension of T5 (Raffel et al., 2020) that handles long sequence inputs efficiently. The baseline takes only the dialog context C as input.
- LongT5-with-Topic uses the same architecture as LongT5 but takes both the topic document D_T and the dialog context C as input.

The second set of baselines consists of systems augmented with *frozen* retrievers:

- [Frozen DPR + FiI] first uses a pre-trained DPR model to retrieve five potentially relevant knowledge snippets. Conditioned on the retrieved knowledge and the dialog context, a FiI generation model then generates a response.
- [Frozen DPR + FiI (using D_T)] uses the same architecture, but its FiI generator also takes the topic document as input.

By comparing our full model against this set of baselines, we can analyze the effectiveness of using cross-attention scores as retrieval pseudo-labels.

Hyperparameters We initialized the DPR component using `all-mpnet-base-v2` (Reimers and Gurevych, 2019). The reranker is initialized using `distilroberta-base`, a distilled version of RoBERTa (Liu et al., 2019). We initialize the generation component of every system using a large version of LongT5 (Guo et al., 2022) unless otherwise stated. The batch size is 32. The number of training epochs is 10. More details about the hyperparameters are in the appendix.

3.2 Experimental Results

Table 1 presents the retrieval and response generation performance of our models and the baselines. First, we can see that our models achieve the highest performance compared with all baselines for both data sets and all evaluation metrics. This result emphasizes the importance of using external knowledge for document-oriented dialog systems and demonstrates the effectiveness of doing so by using our approach of fine-tuning the retrieval engine with attention score-based pseudo-labels.

The results in Table 1 also show that using the topic document is crucial for effective response generation. For example, LongT5-with-Topic outperforms the default LongT5 baseline, especially on MultiDoc2Dial. Similar results are observed for the baselines that use frozen retrievers.

The results also show that there is a clear positive correlation between text generation performance and retrieval performance. Compared with the frozen retrieval baselines, our full model achieves much higher generation performance by training the retrieval and re-ranking models with our end-to-end approach. The generator of every model listed in Table 1 is initialized with the same large version of LongT5. This shows the importance of the retrieval engine on the response generation performance.

Using cross-attention scores as retrieval pseudo-labels is highly effective according to Table 1. Specifically, `EKo-DoC` achieves a Recall@1 score of 71.21% on WoW without using human-annotated retrieval labels. Notice also the importance of the re-ranker component. When the re-ranker is disabled (second row) the model achieves a Recall@1 score of 33.88% on MultiDoc2Dial, while the full model achieves a score of 41.60%. Furthermore, Table 3 shows that the effectiveness of using cross-attention scores can even be comparable to that of using gold retrieval labels.

Finally, many previous studies on knowledge-intensive dialog systems do not have any notion of a central topic document (Shuster et al., 2021; Paranjape et al., 2022). To directly compare against these studies, we also train a system that does not explicitly use the topic document (Table 2). Basically, this system is similar to our full model (Figure 2), but it does not take the topic document as part of its input. Overall, our system outperforms all previous state-of-the-art methods in most evaluation metrics. While FiD-Light (Hofstätter et al., 2022) and Re2G (Glass et al., 2022) achieve better Recall@1 and Recall@5 than ours (respectively), FiD-Light and Re2G use gold retrieval labels to train their retrieval engines. Our system does not use such supervision signals, which can be expen-

	Wizard of Wikipedia			MultiDoc2Dial				
	F1	ROUGE-L	Recall@1	F1	ROUGE-L	Recall@1	EM	BL
Our Models								
`EKo-DoC` *	**20.77**	**18.64**	**71.21**	**42.80**	**39.02**	**41.60**	**6.40**	**29.17**
`EKo-DoC` (No Re-ranker) *	20.72	18.63	69.71	41.31	37.47	33.88	6.13	27.36
Baselines with Frozen Retrievers								
Frozen DPR + FiI (using D_T) *	19.66	17.80	49.69	36.19	32.44	14.44	4.20	21.30
Frozen DPR + FiI	19.29	17.53	49.69	33.90	30.31	14.44	3.74	18.56
Seq2Seq Baselines								
LongT5-with-Topic *	19.92	18.12	-	30.12	26.43	-	2.27	13.00
LongT5	16.21	14.94	-	24.47	20.68	-	0.66	5.46

Table 1: Overall results (in %) on the test sets of WoW and MultiDoc2Dial. For fair comparison, all models shown here do not use any gold retrieval labels during training. The symbol * denotes models that explicitly use the central document. In such a model, at least one component directly includes the central document as part of its input. All differences in performance between our models and the baselines are statistically significant with a p-value < 0.05.

	Recall@1	Recall@5	ROUGE-L	F1	KILT-F1
`EKo-DoC` (**Without** explicitly using the central doc.)	61.86	78.18	**18.32**	**20.42**	**14.41**
QKConv (Cai et al., 2022)	60.98	76.58	17.72	19.95	13.64
Hindsight (Paranjape et al., 2022)	56.08	74.27	17.06	19.19	13.39
FiD-Light (Hofstätter et al., 2022)	**66.15**	76.51	15.78	17.82	13.06
Re2G (Glass et al., 2022)	60.10	**79.98**	16.76	18.90	12.98
SEAL (Bevilacqua et al., 2022)	57.55	78.96	16.65	18.34	11.63
RAG (Petroni et al., 2021)	57.75	74.61	11.57	13.11	8.75

Table 2: Performance of state-of-the-art models on the test set of WoW according to the public leaderboard (as of February 2023). We are hiding our score on the leaderboard during the anonymous review process. The leaderboard is available at `https://eval.ai/web/challenges/challenge-page/689/leaderboard/1909`, and it also uses additional metrics such as Recall@5 and KILT-F1 (Petroni et al., 2021). Note that FiD-Light and Re2G use gold retrieval labels to train their retrieval engines, while our system does not rely on such supervision signals.

	R@1	R@5
`EKo-DoC`	45.32	61.36
DPR + RR (finetuned using gold labels)	**46.46**	**65.88**

Table 3: Comparison between using attention score-based pseudo-labels and using gold retrieval labels. Here, scores on the dev set of WoW are reported. In addition, different from the public leaderboard, we evaluate retrieval at a more fine-grained granularity, which is the passage level instead of the page level.

sive to obtain.

3.3 Human Evaluation

There can be many appropriate responses to the last turn of a dialog; therefore, human evaluation is typically crucial to properly evaluate the performance of a dialog system (Liu et al., 2016; Ghandeharioun et al., 2019). We conducted a human evaluation of various models by using Amazon's Mechanical Turk (AMT).[3] Specifically, we created 150 evaluation tasks, each consisting of a dialog context selected randomly from WoW and a set of responses produced by different model variants. For each task, we asked three different AMT workers to rank the models' outputs based on:

1. *Fluency* (0/1). Is the generated response fluent and grammatically correct?
2. *Relevance* (0/1). Is the response on-topic and relevant to the last turn of the dialog?
3. *Interestingness* (0/1). Does the response provide new interesting information that is not already mentioned in the dialog?

Table 4 presents the human evaluation results comparing our full model against two baselines, LongT5 and LongT5-with-Topic. We chose these baselines as our main goal here is to analyze how external knowledge helps in document-oriented dialogs. According to human annotators, our model

---

[3] All details about the annotation tasks are in the appendix.

outperforms both baselines substantially in all three criteria. Furthermore, `EKo-DoC` achieves the most gain in the "interestingness" criterion. This is expected because, for example, compared to LongT5-with-Topic, our model also makes use of external knowledge when generating responses.

Full Model vs. LongT5	Better	Same	Worse
Fluency	31.01	47.29	21.71
Relevance	39.85	38.35	21.80
Interestingness	48.39	35.48	16.13
Full Model vs. LongT5-with-Topic			
Fluency	24.62	62.31	13.08
Relevance	32.03	53.13	14.84
Interestingness	40.60	39.85	19.55

Table 4: Human evaluation results (%) comparing our full model to LongT5 (top part) and LongT5-with-Topic (bottom part). The numbers in the column "Better" denote the percentage of the snippets where our full model is considered better (according to humans).

3.4 Analysis

Qualitative Analysis Table 5 shows some examples from WoW that illustrate how incorporating external knowledge can improve the quality of generated responses. Note that more qualitative examples are provided in the appendix.

In the first example, the responses generated by all models are fluent and grammatically correct. However, LongT5 generates a factually incorrect statement, as it does not have any knowledge that is not implicitly stored in its parameters. Nevertheless, LongT5 seems to possess some knowledge about the world, as Sam Walton is the founder of Walmart, which is related to Target Corporation.

In the second example, in the beginning, the dialog is mainly about rock and roll in general. However, the last dialog turn deviates slightly from the general topic and starts talking about James Brown, who influenced several music genres. The topic document does not have any information about the person. Therefore, LongT5-with-Topic generates a factually correct response about rock and roll, but it is unrelated to James Brown. In contrast, by leveraging external knowledge, our model generates a response that is more relevant to the last dialog turn.

Comparison between FiI and FiD We compare our FiI method and the popular FiD method (Izacard and Grave, 2021b) in Table 6. FiI outperforms FiD, as FiI encodes the dialog context and the top-ranked knowledge snippets jointly. In contrast, FiD encodes the retrieved snippets independently, so there is less interaction between the knowledge snippets before the final decoding stage. In conclusion, by using FiI instead of FiD, our generator can extract semantic relationships between the different input signals more effectively.

Examples
Topic Document: The first paragraph of the Wikipedia page talking about Target Corporation. **Dialog Context**: A: Target Corporation is the second largest discount retailer. B: I assume that it is just before Walmart, but still huge. **Candidate Responses:** LongT5: It was founded in 1971 by Sam Walton. LongT5-with-Topic: Target was founded by George Dayton in 1902. `EKo-DoC`: Target is the eighth largest retailer in the US.
Topic Document: The first paragraph of the Wikipedia page talking about Rock and roll. **Dialog Context**: A: My favorite music is rock and roll. B: Gotta love some Rock and Roll. It's one of my favorites as well. ... A: Indeed it did. I was speaking to a friend earlier about *James Brown* and how he influenced so many genres of music starting in the 60's. **Candidate Responses:** LongT5-with-Topic: I like the fact that while elements of what was to become rock and roll can be heard in blues records from the 1920s and in country records of the 1930s, the genre did not acquire its name until 1954. `EKo-DoC`: I agree. He was an American singer, songwriter, dancer, musician, record producer and bandleader.

Table 5: Examples showing how external knowledge improves the quality of generated responses. Red is used to indicate responses that are factually *incorrect*. Blue is used to indicate responses that are factually *correct*.

	F1	ROUGE-L
Freezed DPR + FiI (LongT5)	**34.07**	**29.97**
Freezed DPR + FiD (LongT5)	33.82	29.77
Freezed DPR + FiD (T5)	32.80	28.65

Table 6: Comparison between FiI and FiD. The dev set of MultiDoc2Dial is used.

4 Related Work

Document-oriented Conversations Humans typically seek information in a *conversational* manner, for example to find answers to questions (Choi et al., 2018) or to seek guidance in performing real world tasks (Gottardi et al., 2022). As such, several recent conversational datasets were introduced for building models to assist in information-seeking dialogs. For example, Choi et al. (2018) presented QuAC, a dataset that contains 14K information-seeking QA dialogs:

each is centered around a short evidence text from Wikipedia and involves a student and a teacher. The student poses a sequence of free-form questions about the text, while the teacher answers the questions by providing short excerpts from the text. Concurrently, Reddy et al. (2019) introduced CoQA, a dataset in which a machine has to understand a text passage and answer a series of questions that appear in a conversation. Choi et al. (2022) introduced a dataset for the novel setting of Conversational Task Assistants, where users seek guidance from a conversational agent to perform real world tasks: in this setting, a conversation is centered around a document describing a task. SOTA methods for these tasks typically do not use any external knowledge beyond the dialog context and the given background text (Zhu et al., 2018; Huang et al., 2019; Qu et al., 2019; Gupta et al., 2020; Zhao et al., 2021; Qian et al., 2022).

Open-Domain Question Answering In open-domain QA, the goal is to find the answer to a (typically short) question over a large corpus such as Wikipedia (Voorhees and Tice, 2000; Chen and Yih, 2020). Passage retrieval has been an essential component of many state-of-the-art open-domain QA systems (Karpukhin et al., 2020; Lewis et al., 2020b; Piktus et al., 2021; Min et al., 2021; Zhu et al., 2021). An effective retrieval component can reduce the search space for answer extraction and identify the support context for users to verify the answer. As a result, many studies have focused on improving the retrieval components, ranging from removing the need for strong supervision signals (Izacard and Grave, 2021a; Ram et al., 2022) to adding sophisticated reranking components (Yu et al., 2022a; Glass et al., 2022; Yu et al., 2022b). Open-domain QA is typically non-conversational and does not have any notion of a topic document.

Weak Supervision for Neural Retrieval A closely related study (Izacard and Grave, 2021a) also proposed an approach to train retrieval systems without strong supervision. The work focuses on the task of QA and uses the FiD approach for output generation. In contrast, we focus on building dialog systems and tailoring our learning approach to our newly proposed FiI approach.

Knowledge-Grounded Dialog Generation Incorporating background knowledge into conversation models can make dialogs more informative and engaging. Therefore, many recent studies have investigated various techniques for selecting relevant knowledge and integrating it into the response generation process (Lian et al., 2019; Kim et al., 2020; Li et al., 2020; Shuster et al., 2021; Chen et al., 2021; Mishra et al., 2022; Li et al., 2022b). Many of these studies utilize the Wizard of Wikipedia (WoW) dataset (Dinan et al., 2019). Our system, `EKo-DoC`, achieves a new state-of-the-art result on WoW, as shown in Table 2. This result provides some guarantee on the relative performance of `EKo-DoC` compared to many previous SOTA knowledge-grounded dialog systems. As of February 2023, there are nearly 40 tested methods on the public leaderboard of WoW, though Table 2 only shows a subset due to space limitations.

5 Conclusions

This work proposes and studies a new problem setting that combines document-oriented conversations and open-domain QA. We introduce a new architecture for the problem that includes a dense passage retriever, a re-ranker, and a response generation model. We train these three components end-to-end and use cross-attention scores as pseudo-labels to update the retrieval engine. Extensive experimental results on two public datasets demonstrate the effectiveness of our method. In the future, we plan to reduce the computational complexity of our model by using compression techniques.

6 Ethical Consideration

Limitations While `EKo-DoC` achieves new SOTA results on WoW, its performance is far from perfect. A ROUGE-L score of 18.64% and a Recall@1 score of 71.21% indicate that there is much room for improvement. Based on our manual analysis, we found that `EKo-DoC` sometimes generates responses that are a little bit unnatural. Finally, a common limitation of many Transformer-based systems, such as `EKo-DoC`, is the large computational complexity. We plan to reduce the computational complexity of `EKo-DoC` by using some compression techniques (Lai et al., 2020; Sun et al., 2020).

Potential Risks A potential malicious use case of research on conversational AI is for building dialog systems that pose as humans and then proactively alter users' perceptions about specific issues, evaluations of products or services, or political inclinations (Qi et al., 2021). We urge anyone who uses or builds upon our research to avoid such malicious use cases.

References

Akari Asai, Matt Gardner, and Hannaneh Hajishirzi. 2022. Evidentiality-guided generation for knowledge-intensive NLP tasks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2226–2243, Seattle, United States. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Wen-tau Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *arXiv preprint arXiv:2204.10628*.

Mingzhu Cai, Siqi Bao, Xin Tian, H. He, Fan Wang, and Hua Wu. 2022. Query enhanced knowledge-intensive conversation via unsupervised joint modeling. *ArXiv*, abs/2212.09588.

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37, Online. Association for Computational Linguistics.

Xiuyi Chen, Feilong Chen, Fandong Meng, Peng Li, and Jie Zhou. 2021. Unsupervised knowledge selection for dialogue generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1230–1244, Online. Association for Computational Linguistics.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Jason Ingyu Choi, Saar Kuzi, Nikhita Vedula, Jie Zhao, Giuseppe Castellucci, Marcus Collins, Shervin Malmasi, Oleg Rokhlenko, and Eugene Agichtein. 2022. Wizard of tasks: A novel conversational dataset for solving real-world tasks in conversational settings. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3514–3529, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of Wikipedia: Knowledge-powered conversational agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021. MultiDoc2Dial: Modeling dialogues grounded in multiple documents. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6162–6176, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Asma Ghandeharioun, Judy Hanwen Shen, Natasha Jaques, Craig Ferguson, Noah J. Jones, Àgata Lapedriza, and Rosalind W. Picard. 2019. Approximating interactive human evaluation with self-play for open-domain dialog systems. In *NeurIPS*.

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, Ankita Rajaram Naik, Pengshan Cai, and Alfio Gliozzo. 2022. Re2g: Retrieve, rerank, generate. *arXiv preprint arXiv:2207.06300*.

Anna Gottardi, Osman Ipek, Giuseppe Castellucci, Shui Hu, Lavina Vaz, Yao Lu, Anju Khatri, Anjali Chadha, Desheng Zhang, Sattvik Sahai, Prerna Dwivedi, Hangjie Shi, Lucy Hu, Andy Huang, Luke Dai, Bofei Yang, Varun Somani, Pankaj Rajan, Ron Rezac, Michael Johnston, Savanna Stiff, Leslie Ball, David Carmel, Yang Liu, Dilek Hakkani-Tur, Oleg Rokhlenko, Kate Bland, Eugene Agichtein, Reza Ghanadan, and Yoelle Maarek. 2022. Alexa, let's work together: Introducing the first alexa prize taskbot challenge on conversational task assistance. In *Alexa Prize TaskBot Challenge Proceedings*.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736, Seattle, United States. Association for Computational Linguistics.

Somil Gupta, Bhanu Pratap Singh Rawat, and Hong Yu. 2020. Conversational machine comprehension: a literature review. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2739–2753, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sebastian Hofstätter, Jiecao Chen, Karthik Raman, and Hamed Zamani. 2022. Fid-light: Efficient and effective retrieval-augmented text generation. *ArXiv*, abs/2209.14290.

Hsin-Yuan Huang, Eunsol Choi, and Wen-tau Yih. 2019. Flowqa: Grasping flow in history for conversational machine comprehension. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *International Conference on Learning Representations*.

Gautier Izacard and Edouard Grave. 2021b. Leveraging passage retrieval with generative models for open domain question answering. In *EACL*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Chandra Khatri, Behnam Hedayatnia, Chandra Khatri, Anushree Venkatesh, Jeff Nunn, Yi Pan, Qing Liu, Han Song, Anna Gottardi, Sanjeev Kwatra, Sanju Pancholi, Ming Cheng, Qinglang Chen, Lauren Stubel, and Karthik Gopalakrishnan. 2018. Advancing the state of the art in open domain dialog systems through the alexa prize. In *Alexa Prize SocialBot Grand Challenge 2 Proceedings*.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential Latent Knowledge Selection for Knowledge-Grounded Dialogue. In *ICLR*.

Tuan Manh Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. 2020. A simple but effective bert model for dialog state tracking on resource-limited systems. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8034–8038. IEEE.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Kun Li, Tianhua Zhang, Liping Tang, Junan Li, Hongyuan Lu, Xixin Wu, and Helen Meng. 2022a. Grounded dialogue generation with cross-encoding re-ranker, grounding span prediction, and passage dropout. In *Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, pages 123–129, Dublin, Ireland. Association for Computational Linguistics.

Linxiao Li, Can Xu, Wei Wu, Yufan Zhao, Xueliang Zhao, and Chongyang Tao. 2020. Zero-resource knowledge-grounded dialogue generation. *Advances in Neural Information Processing Systems*, 33:8475–8485.

Sha Li, Mahdi Namazifar, Di Jin, Mohit Bansal, Heng Ji, Yang Liu, and Dilek Hakkani-Tur. 2022b. Enhancing knowledge selection for grounded dialogues via document semantic graphs. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2810–2823, Seattle, United States. Association for Computational Linguistics.

Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5081–5087. International Joint Conferences on Artificial Intelligence Organization.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, et al. 2021. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. In *NeurIPS 2020 Competition and Demonstration Track*, pages 86–111. PMLR.

Mayank Mishra, Dhiraj Madan, Gaurav Pandey, and Danish Contractor. 2022. Variational learning for unsupervised knowledge grounded dialogs. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4303–4309. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Ashwin Paranjape, Omar Khattab, Christopher Potts, Matei Zaharia, and Christopher D Manning. 2022. Hindsight: Posterior-guided training of retrievers for improved open-ended generation. In *International Conference on Learning Representations*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Neural Information Processing Systems*.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Dmytro Okhonko, Samuel Broscheit, Gautier Izacard, Patrick Lewis, Barlas Ouguz, Edouard Grave, Wen tau Yih, and Sebastian Riedel. 2021. The web is your oyster - knowledge-intensive nlp against a very large web corpus. *ArXiv*, abs/2112.09924.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Peng Qi, Jing Huang, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2021. Conversational ai systems for social good: Opportunities and challenges. *ArXiv*, abs/2105.06457.

Jin Qian, Bowei Zou, Mengxing Dong, Xiao Li, AiTi Aw, and Yu Hong. 2022. Capturing conversational interaction for question answering via global history reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2071–2078, Seattle, United States. Association for Computational Linguistics.

Chen Qu, Liu Yang, Minghui Qiu, W. Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. 2019. Bert with history answer embedding for conversational question answering. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, page 117–126, New York, NY, USA. Association for Computing Machinery.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. Learning to retrieve passages without supervision. In *NAACL*.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *ArXiv*, abs/1506.05869.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece. European Language Resources Association (ELRA).

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2022a. KG-FiD: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4961–4974, Dublin, Ireland. Association for Computational Linguistics.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2022b. A survey of knowledge-enhanced text generation. *ACM Comput. Surv.* Just Accepted.

Jing Zhao, Junwei Bao, Yifan Wang, Yongwei Zhou, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2021. RoR: Read-over-read for long document machine reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1862–1872, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2020. The Design and Implementation of XiaoIce, an Empathetic Social Chatbot. *Computational Linguistics*, 46(1):53–93.

Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2018. Sdnet: Contextualized attention-based deep network for conversational question answering. *arXiv preprint arXiv:1812.03593*.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *ArXiv*, abs/2101.00774.

A Human Evaluation

We used Amazon's Mechanical Turk (AMT) to perform a human evaluation of various models. More specifically, we first created 150 evaluation snippets, each consisting of a dialog context and a set of responses produced by three model variants: LongT5, LongT5-with-Topic, and our full model. Figure 3 shows an example snippet. For each snippet, we asked three different AMT workers to rank the models' outputs based on three criteria: (1) *Fluency*, (2) *Relevance*, and (3) *Interestingness*. Note that for each snippet, the presentation order of the generated responses was randomized. For example, for some AMT tasks, the first response may come from LongT5, while for some other tasks, the first response may come from our full model. Figure 4 shows the instructions we showed at the beginning of each AMT task. Even though our instructions do not explicitly explain how the collected data would be used, we believe it was clear to the AMT workers that we would use the data for researching dialog systems.

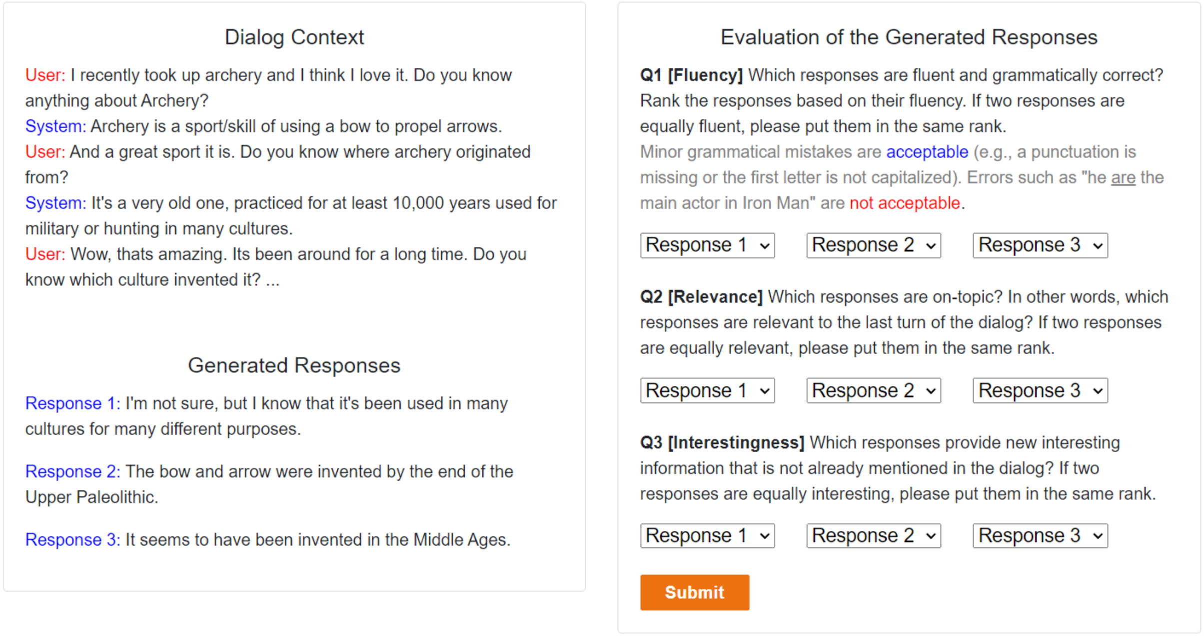
To decide the appropriate cost of each AMT task, we conducted a preliminary study, asking three NLP researchers to do about 20 tasks each. We then computed the average time it would take for each task. After that, we set the value of each AMT task so that someone who worked on our study for about an hour would make at least the required minimum wage.

B Datasets

We use two public datasets in this work: *Wizard of Wikipedia (WoW)* (Dinan et al., 2019) and *MultiDoc2Dial* (Feng et al., 2021). For WoW, we use the version provided by the KILT benchmark (Petroni et al., 2021). KILT is released under the MIT License, and MultiDoc2Dial is released under the Apache-2.0 license. Our use of the datasets is consistent with their licenses.

WoW and MultiDoc2Dial are available at `github.com/facebookresearch/KILT` and `github.com/IBM/multidoc2dial` (respectively). When we first downloaded the datasets, we randomly sampled about 10∼20 examples for each dataset and checked whether the examples contained any offensive content. Overall, we did not see any example that had offensive content.

The two datasets are in English. WoW consists of open-domain conversations, which collectively cover a wide range of topics, ranging from Ameri-



Dialog Context

User: I recently took up archery and I think I love it. Do you know anything about Archery?
System: Archery is a sport/skill of using a bow to propel arrows.
User: And a great sport it is. Do you know where archery originated from?
System: It's a very old one, practiced for at least 10,000 years used for military or hunting in many cultures.
User: Wow, thats amazing. Its been around for a long time. Do you know which culture invented it? ...

Generated Responses

Response 1: I'm not sure, but I know that it's been used in many cultures for many different purposes.

Response 2: The bow and arrow were invented by the end of the Upper Paleolithic.

Response 3: It seems to have been invented in the Middle Ages.

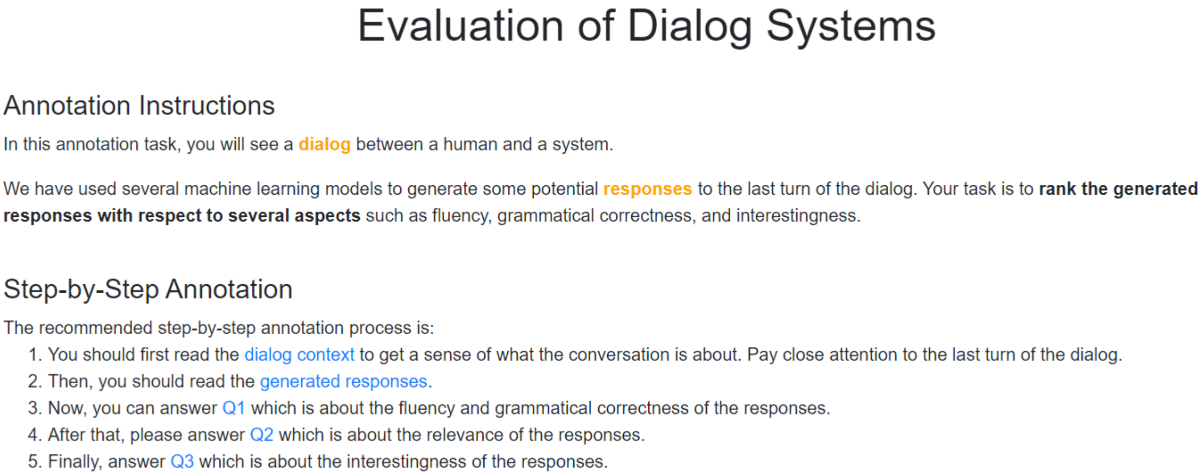
Evaluation of the Generated Responses

Q1 [Fluency] Which responses are fluent and grammatically correct? Rank the responses based on their fluency. If two responses are equally fluent, please put them in the same rank.
Minor grammatical mistakes are acceptable (e.g., a punctuation is missing or the first letter is not capitalized). Errors such as "he are the main actor in Iron Man" are not acceptable.

Q2 [Relevance] Which responses are on-topic? In other words, which responses are relevant to the last turn of the dialog? If two responses are equally relevant, please put them in the same rank.

Q3 [Interestingness] Which responses provide new interesting information that is not already mentioned in the dialog? If two responses are equally interesting, please put them in the same rank.

Figure 3: An example AMT task.



Evaluation of Dialog Systems

Annotation Instructions

In this annotation task, you will see a **dialog** between a human and a system.

We have used several machine learning models to generate some potential **responses** to the last turn of the dialog. Your task is to **rank the generated responses with respect to several aspects** such as fluency, grammatical correctness, and interestingness.

Step-by-Step Annotation

The recommended step-by-step annotation process is:

1. You should first read the dialog context to get a sense of what the conversation is about. Pay close attention to the last turn of the dialog.
2. Then, you should read the generated responses.
3. Now, you can answer Q1 which is about the fluency and grammatical correctness of the responses.
4. After that, please answer Q2 which is about the relevance of the responses.
5. Finally, answer Q3 which is about the interestingness of the responses.

Figure 4: The instructions we provided for each AMT task.

can football to Rock and Roll music. On the other hand, MultiDoc2Dial focuses on four specific domains: Social Security Administration, Veteran Affairs, Student-Aid, and DMV.

If we consider a pair of a dialog context and its corresponding ground-truth response as one example, there are 63,734/3,054/2,944 examples in the train/dev/test splits of WoW (respectively). For MultiDoc2Dial, we refer the readers to the original paper (Feng et al., 2021) for more detailed statistics of the original dataset.

C Reproducibility Information

In this section, we present the reproducibility information of our paper.

Implementation Dependencies Libraries Pytorch (Paszke et al., 2019), Transformers 4.20.1 (Wolf et al., 2020), SentenceTransformers 2.2.0 (Reimers and Gurevych, 2019), faiss-gpu 1.7.2 (Johnson et al., 2021).

Computing Infrastructure We conducted our experiments using Amazon's EC2 virtual machines. Overall, our work can be reproduced using a single p3.8xlarge instance. Information about the cost of using Amazon's EC2 P3 instances can be found at `https://aws.amazon.com/ec2/instance-types/p3/`.

Number of Model Parameters Our full model consists of three components: a DPR model, a RR model, and a seq2seq generation model. We initialize the DPR model using a Sentence-Transformer model named `all-mpnet-base-v2` (Reimers and Gurevych, 2019), which has about 110M parameters. We initialize the RR model using `distilroberta-base` (Liu et al., 2019), which has about 82M parameters. Finally, we initialize the generation model using `google/long-t5-tglobal-large` (Guo et al., 2022), which has about 750M parameters.

Hyperparameters The effective batch size is 32. The number of training epochs is 10. The base learning rate is 5e-5. K_1 is set to be 25, and K_2 is set to be 5. We use the Adam optimizer and set gradient clipping to 1.0.

Expected Validation Performance For each model variant, we report the test performance of the checkpoint with the best validation F1 score in the main paper (refer to Table 1). On the dev set of WoW, our full model achieves a unigram F1 score of 21.13% and a ROUGE-L score of 18.68%. On the dev set of MultiDoc2Dial, our full model achieves a unigram F1 score of 41.90% and a ROUGE-L score of 37.64%.

# Retrieved Docs	F1	ROUGE-L	Recall@1
$K_1 = 25$, $K_2 = 5$	38.76	34.62	39.70
$K_1 = 25$, $K_2 = 10$	39.84	35.81	44.44
$K_1 = 25$, $K_2 = 15$	39.71	35.58	42.28
$K_1 = 50$, $K_2 = 5$	39.10	34.92	40.28
$K_1 = 50$, $K_2 = 10$	39.63	35.44	42.82
$K_1 = 50$, $K_2 = 15$	39.63	35.56	43.51

Table 7: Results on the dev set of MultiDoc2Dial when we vary K_1 and K_2, the numbers of documents retrieved by DPR and the reranker (respectively). In order to save computational time, we use the base version of LongT5 in this ablation study.

D Additional Analysis

Table 7 shows the results on the dev set of MultiDoc2Dial when we vary K_1 and K_2. In general, increasing the number of retrieved documents typically has marginal positive effects on the evaluation metrics. However, note that increasing the values of K_1 and K_2 also increases the computational complexity and running time of `EKo-DoC`. Therefore, we set K_1 to 25 and K_2 to 5. We believe these values achieve a good balance between the model complexity and the quality of the generated responses.

Table 8 shows some additional examples from WoW that illustrate how incorporating external knowledge can improve the quality of generated responses.

Examples
Topic Document: The first paragraph of the Wikipedia page talking about Ferrari. **Dialog Context**: A: I have 3 Ferrari cars in my garage and I hardly know nothing about them. I read some self help books, then I was rick all of the sudden. B: Those are some nice cars they are manufacturer in Maranello. A: Yeah that's pretty cool. Do you know how old the company is? **Candidate Responses:** LongT5: I don't know but they are a luxury car manufacturer. `EKo-DoC`: I know it was founded in 1939.
Topic Document: The first paragraph of the Wikipedia page talking about Overeating. **Dialog Context**: A: Overeating is a serious problem in the USA. **Candidate Responses:** LongT5: I know, it's a problem that affects a lot of people. `EKo-DoC`: I think it is a serious problem in the US. It is the excess food consumed in relation to the energy that an organism expends.
Topic Document: The first paragraph of the Wikipedia page talking about Red. **Dialog Context**: B: Red is my favorite color. It is the color of a lot of my favorite things, like strawberries? A: Awesome! Did you also know that Mars is red because of iron oxide? ... B: That makes sense. I always liked studying the planets growing up in school. **Candidate Responses:** LongT5-with-Topic: I also like how Mars is the only planet that has a moon. `EKo-DoC`: Yeah, I love learning about the planets. I'm sure you've heard of the planet Mercury.

Table 8: Additional examples showing how external knowledge improves the quality of generated responses. Red is used to indicate responses that are factually *incorrect*. Blue is used to indicate responses that are factually *correct*.

In-Depth Look at Word Filling Societal Bias Measures

Matúš Pikuliak and **Ivana Beňová** and **Viktor Bachratý**
Kempelen Institute of Intelligent Technologies
`matus.pikuliak@kinit.sk`

Abstract

Many measures of societal bias in language models have been proposed in recent years. A popular approach is to use a set of word filling prompts to evaluate the behavior of the language models. In this work, we analyze the validity of two such measures – *StereoSet* and *CrowS-Pairs*. We show that these measures produce unexpected and illogical results when appropriate control group samples are constructed. Based on this, we believe that they are problematic and using them in the future should be reconsidered. We propose a way forward with an improved testing protocol. Finally, we also introduce a new gender bias dataset for Slovak.[1]

1 Introduction

Language models (LMs) are ubiquitous in current NLP and have brought undeniable performance improvements for many tasks. Concerns have been raised about the fairness of these models (Blodgett et al., 2020; Shah et al., 2020; Dev et al., 2021b). Since LMs are usually trained with web-based text corpora generated by a general population, there is a risk that they will learn certain societal biases, such as sexist or racist stereotypes. With these models regularly being used as backbones for further fine-tuning, this unfairness might propagate further to downstream models and ultimately to user-facing applications.

Based on this assumption, many attempts were made to quantify the *bias* in LMs. The measures usually observe LM outputs or inner workings to reveal problematic biased behavior. A popular method is to create a set of word filling prompts that test LM behavior in various situations, and then interpret the differences. For example, would an LM choose a negative stereotypical word for `X` in the sentence `All women are X`? Tests like these are often proposed because neural LMs are notoriously blackbox and it is otherwise difficult to interpret their inner working reliably. However, the observation process must be done in a methodologically sound manner and the correct assumptions must be used to ensure accurate results.

In this work, we examine the validity of two widely used methodologies – StereoSet (Nadeem et al., 2021) and CrowS-Pairs (Nangia et al., 2020) – for measuring societal bias in *masked* LMs. We first identify several theoretical problems in their score calculations. Then we show that these problems can be observed in the available data and we demonstrate that the LMs exhibit unexpected behavior that violates key assumptions made by these methodologies. This leads us to question the validity of the reported results. We propose a way to improve the methodologies by introducing a new score definition. During experiments we introduce several new variants of the existing datasets and a completely new dataset in Slovak. These new datasets are used to compare the expected behavior of the LMs with their actual behavior.

Our results challenge the validity of previous studies. This is a significant issue as these measures are widely used[2] to *demonstrate* the level of bias in LMs (Zhang et al., 2022, i.a.), as benchmarks for debiasing techniques (Meade et al., 2022, i.a.), as inspiration for bias research in languages other than English (Névéol et al., 2022; Kaneko et al., 2022), and for other bias-related research. If our assertions about their validity are accurate, all these efforts could be in danger. Moreover, it is possible that other similar measures may face similar problems.

2 Related Work

Measuring LM Bias. In recent years, numerous methodologies and datasets have emerged for measuring societal bias in LMs and other NLP mod-

---

[1] Data and code are available at `https://github.com/kinit-sk/bias-methodology`.

[2] The two papers have 234 and 149 citations respectively according to Google Scholar as of February 2023.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3648–3665
May 2-6, 2023 ©2023 Association for Computational Linguistics

els (Dev et al., 2021b). The techniques for masked LMs are generally based on three types of analysis: (1) LM behavior on downstream tasks (Rudinger et al., 2018; De-Arteaga et al., 2019, i.a.), (2) inner LM representations (May et al., 2019; Webster et al., 2020, i.a.), and (3) word filling behavior. Word filling can be done using either short, semantically neutral templates filled with lexicons (Kurita et al., 2019; Ahn and Oh, 2021), or through crowd-sourced sentences that capture biased behavior. The two techniques discussed in this paper – StereoSet and CrowS-Pairs – belong to the latter category.

Critique. Papers criticize and evaluate the proposed bias measuring techniques from various perspectives. Blodgett et al. (2021) identify several conceptualization and operationalization pitfalls in the existing benchmarks and estimate that a significant portion of samples have validity issues. The lack of robustness of the proposed metrics w.r.t. specific choices of templates, prompts, lexicon seeds, metrics, sampling strategies is also a concern (Akyürek et al., 2022; Antoniak and Mimno, 2021; Delobelle et al., 2021). Low correlations between individual scores raise questions about what exactly is being measured (Delobelle et al., 2021; Cao et al., 2022; Goldfarb-Tarrant et al., 2021). Other criticisms are more conceptual. Blodgett et al. (2020) point out that the motivation behind bias measuring techniques is often *"vague, inconsistent and, lacking in normative reasoning"* and that the techniques are often *"poorly matched to the motivation"*. The lack of cultural sensitivity results in methods that are often Anglo-centric or US-centric (Talat et al., 2022; Stanczak and Augenstein, 2021), do not correctly handle marginalized groups (Devinney et al., 2022; Dev et al., 2021a), or have other similar cultural issues.

Many of these problems could be addressed by improved training for data creators and by increasing data quantity and quality. However, we show that even with perfect data, some of the proposed methodologies may still not yield valid results.

3 Methodologies and Datasets

Both StereoSet (SS) and CrowS-Pairs (CS) measure bias against certain groups using sets of word filling samples. There is usually a coupling between the dataset (the specific samples used for score calculations) and the methodology (how is the score calculated) (Orgad and Belinkov, 2022). @e distinguish between these two concepts (e.g., by saying *StereoSet (SS) dataset* and *StereoSet (SS) methodology*) and we effectively decouple them when we stress-test the methodologies with new datasets.

In this section, we briefly introduce the two existing methodologies and also our own dataset in Slovak language that is compatible with both of them. We also created various new versions and extensions of the existing datasets for our experiments, which will be introduced as appropriate. All the datasets are documented in Appendix A.

3.1 StereoSet

Nadeem et al. (2021) introduce the SS *methodology* and several *datasets* compatible with it that address different bias types – gender, race, profession, and religion. Each dataset consists of pairs of sentences that differ in exactly one word. The dataset creators were instructed to first generate stereotypical and anti-stereotypical words associated with a specific group of people and then write a template in which these words could be used. For example, `The male is strong / The male is weak` is a pair that stereotypes males as strong. In this case, `strong` and `weak` are the *keywords* that differ.

The SS *methodology* is based on the idea that a biased masked LM should prefer the stereotypical keywords in these pairs. The LM is fed the sentence with the slot for the keyword masked and is asked to calculate the probabilities for both possible keywords. The measure of bias is the percentage of samples where the model prefers the stereotypical keyword[3]. The authors define that 50% is an optimal ratio. The SS *datasets* were criticized for its subpar data quality, with estimates ranging from 38% (Nangia et al., 2020) to as high as 94% (Blodgett et al., 2021) of the samples being problematic.

3.2 CrowS-Pairs

The CrowS-Pairs (Nangia et al., 2020) proposal consists of a *methodology* and 9 *datasets* about different bias types. Each CS dataset consists of pairs of sentences as well, but the way they were collected is significantly different. The data creators were asked to write a stereotypical sentence about a marginalized group and then rewrite this sentence to change the identity of the

---

[3] We discuss the *intrasentence* variant in this work. The original paper also introduced a *intersentence* variant of the dataset for generative LMs.

group to a non-marginalized one. For example, `Women don't know how to drive / Men don't know how to drive.` First, the stereotypical sentence about `women` was created, then it was changed to talk about `men`. Unlike the SS datasets, the sentences might differ in more than one word, and there are no *keywords*.

The CS methodology measures the LM probabilities for the words that are *the same* for the two sentences. Then it calculates which sentence from the pair has a higher sum of probabilities. This is the sentence that the LM is said to "prefer". Similarly to SS, the percentage of sentences where the LM prefers the marginalized group is considered to be the bias measure, with the 50% threshold considered optimal. The CS datasets were criticized for their quality as well. Blodgett et al. (2021) found only 3% of the samples in the CS datasets admissible. Névéol et al. (2022) published a revised version of the datasets, where they attempted to fix incorrect samples.

3.3 Our own Slovak dataset

We have collected our own Slovak language dataset consisting of 142 samples, which is focused on only one gender stereotype: *Men are more competent then women*. This dataset is compatible with both SS and CS methodologies. It consists of quadruplets of sentences, with the first two sentences being the same as in SS datasets. The third and fourth sentences have the group of people changed from the stereotyped group to a non-stereotyped one. An example in English: `Women are weak / Women are strong / Men are weak / Men are strong.` We can use the first and second sentences for the SS score and the first and third sentences for the CS score.

Our main goal was to create a compatible dataset in a different language. The issues with data quality in both CS and SS datasets inspired us to make the dataset more focused, and we believe that the resulting data validity is higher than in the English datasets. On the other hand, it is smaller and less diverse. The CS methodology is also not an ideal fir for Slovak, as Slovak has gender agreement for verbs and adjectives. This leads to a generally lower number of tokens used in calculating the CS score. Further details on data collection and validation can be found in Appendix A.

4 Case Study: *StereoSet*

Some of the pitfalls previously identified in SS *datasets* (Blodgett et al., 2021) could theoretically be addressed by improving data quality. However, we claim that the SS *methodology* is problematic by itself and does not provide a valid measurement of bias, regardless of how good the data is. First, we will identify several theoretical problems and then show how these problems manifest in the data by breaking several key assumptions the SS methodology makes. The problems are related to both how we calculate scores for individual samples and the way we aggregate them.

1. No control groups. No control groups are used to compare the scores generated by the same samples for different groups of people. If the original pair is about *women*, how do the LMs behave for the same sample about *men*? We cannot determine if the LM exhibits unfair behavior unless we compare its behavior across different groups. For example, the LM might prefer `All X are` **`lazy`** over `All X are` **`diligent`** for both `men` and `women`. There is a hidden and untested assumption that the LM will by default exhibit less biased behavior for non-marginalized groups.

2. Keyword prior equality assumption. The SS methodology does not consider that the stereotypical and anti-stereotypical keywords may not have equal priors. For example, one word might be more *frequent* in the training data and therefore the LMs might generate it with higher probabilities. `All X are` **`lazy`** might have a higher probability than `All X are` **`diligent`** for any group `X`, just because `lazy` is a more common word. Raw word frequency is a simple but feasible example (Wei et al., 2021); the LMs might have also learned other similar patterns. There is a hidden and untested assumption that data creators will naturally generate stereotypical and anti-stereotypical keywords with equal priors.

3. No statistical testing. The original methodology calculates the percentage of samples where the LMs prefer stereotype, but there is no statistical significance testing done on this statistic. The percentage also does not account for the distribution of the measurements. This issue can easily be addressed by tools such as confidence intervals or statistical tests.

4. Lack of information about the probability space. The SS methodology focuses on two specific keywords. However, we lack information about all the other words in the vocabulary. We assume that all the words can be classified into three groups, based on how they would function in the prompt: stereotypical, neutral and anti-stereotypical. To truly measure LM's preference for stereotypes, we would need the information about the overall probabilities for these three groups. The sums of probabilities for these groups may not correspond with the probabilities of the two manually selected keywords.

5. Why 50%. It is unclear why 50% score is considered unbiased. A person who prefers a stereotypical sentence 50% of the time would be probably considered biased. The concept of what anti-stereotypes generated by people should be is unclear. They could be sentences that do not contain any stereotype and are in a sense neutral (e.g., `All women are people`), sentences that contain positive statements about a marginalized group (e.g., `All women are strong`), sentences that contain negative statements about a non-marginalized group (e.g., `All men are weak`), and other similar variants. In the first case, we might wish for the model to have 0% bias and always pick a neutral sentence over a negative stereotype. In other cases, the 50% threshold might be appropriate, although it is questionable whether a hypothetical model that is 50% misogynistic and 50% misandristic should be called unbiased.

We will address and further explore problems #1 and #2 in the following sections. During our experiments, we will also report confidence intervals, thus addressing problem #3. Problems #4 and #5 remain open.

4.1 Control Groups

We analyze the results of the SS methodology by using the same samples edited to describe different groups of people. For example, if there is a sample `All women are lazy/diligent`, we compare the results to a control pair `All men are lazy/diligent`. This experiment addresses problem #1 from our list problems.

We use 3 original SS datasets[4] extended with control group samples and our own dataset. The SS datasets[5] were edited as follows:

Gender. We conducted manual gender-swapping along the male-female axis. Some samples were removed if it was not possible to create a sensible gender-swapped version or if they were grammatically incorrect (4 out of 254 samples were removed). We also created a *filtered* version of the dataset by removing samples that were not inherently about gender bias (103 out of 250 remaining samples were removed). This was often the case for samples that use words `grandma, grandpa, schoolgirl, schoolboy` to describe the target population. For these, dataset creators often used age instead of gender as the basis for stereotyping.

Race and Profession. We created the control group samples automatically by replacing the identifier of country, nationality, or profession with 10 randomly selected terms from the appropriate list of terms used by the original authors. There is a possibility that a small percentage of these have the same stereotype as the original group, making the resulting control pairs invalid.

More details on the process can be found in Appendix A. Note that these extended datasets still have the same data quality limitations as the original ones. Only the gender dataset was manually filtered, so the quality of the samples should be higher.

We define the SS score function ss that calculates the probability p of an LM generating the stereotypical word w_s or the anti-stereotypical word w_a in the sentence template t:

$$ss(w_s, w_a, t) = \log(p(w_s, t)) - \log(p(w_a, t)) \quad (1)$$

To aggregate these results, we define $ss+$ as the percentage of pairs where ss score is positive (as defined in the original paper) and $ss\mu$ as the mean of all the ss scores.

In Figure 1, we compare the results of the original SS pairs with the control group pairs for RoBERTa-Base (Liu et al., 2019) LM for English

---

[4] Excluding the *religion* dataset, because unlike the others, some of the groups are not specified by their name, but by other concepts, such as *Sharia* or *Holy Trinity*. Creating control groups does not make sense for some of these.

[5] In this work we use only the *dev set* from the now defunct StereoSet website, which contains only roughly 25% of the samples the authors collected. The other 75% were not initially published to prevent data leakage, but were later revealed in this repository as we were writing this paper.

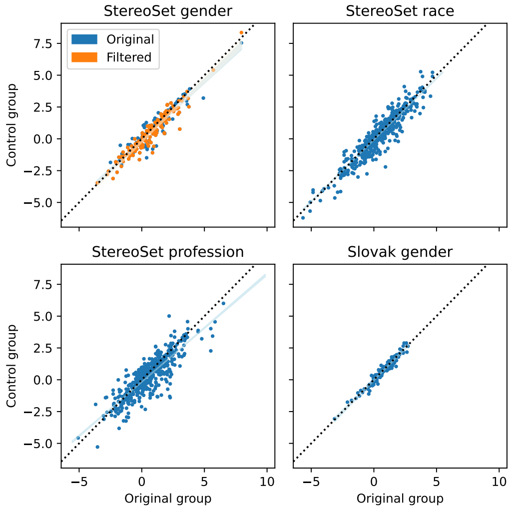


Figure 1: ss scores for the original and control group pairs. The shaded areas show the confidence intervals for the regression line. The dotted lines are identity functions.

and SlovakBERT (Pikuliak et al., 2022) for Slovak[6]. There is a strong correlation between the groups, which is problematic because the SS methodology assumes that when the LM prefers stereotypical keywords for marginalized groups, it is because it is biased. These results show that the LMs have a similar preference for keywords in control groups, even though they should not be stereotyped. We must refuse the notion that the model exhibits stereotypical behavior when more than 50% of samples have a stereotypical preference, since we can see that for many of these samples the LMs have the same or even higher ss score for control group pairs. Or we must admit the the model is biased against control groups as well, but that dilutes the meaning of the word bias as it is commonly used.

We present the statistics (with 95% confidence intervals) of the experiment in Table 1. The results show that the LMs generate positive $ss\mu$ scores and $ss+$ scores higher than 50% for *both* original groups and control groups. We also calculate how many samples that were originally considered stereotypical ($ss > 0$ for the original group) have even higher scores for the control group. We call this the *false positive rate* and it is consistently 30-40%. Similarly we calculate the *false negative rate* for samples where the LMs prefer anti-stereotypical keywords for the original group, but prefer it even more for the control group. This rate is around 50-60%. These statistics indicate a high overall number of samples where the behavior of the LMs does not match the behavior assumed by the SS methodology.

We also calculate Pearson's ρ for the ss scores. The strong correlations suggest that the LMs make predictions mainly for reasons other than *bias*, such as word frequencies or other linguistic patterns instead of their "beliefs" about groups of people. These results cast doubt on the validity of the SS methodology. It is difficult to conclude that the LM is sexist against women when it has the same or even stronger tendencies for men in many of the samples used to demonstrate its bias. The $ss\mu$ or $ss+$ scores cannot be taken at face value without comparison to appropriate control groups. This behavior appears to be universal across different bias types, languages, and language models. There is not a single statistically significant example of a negative $ss\mu$ score or $ss+$ score below 50% for the control group.

On the other hand, the LMs consistently give *lower* scores to the control groups. This suggests that the bias might indeed be present, but a different method of measurement is required. We will define a score that compares the results of the original and control group pairs in Section 6.

4.2 Stereotypical Keywords Bias

We have shown that the LMs prefer the stereotypical keywords ($ss > 0$) for both the original and control group pairs. It is not immediately clear why this is the case.

One explanation may be that the stereotypical keywords are simply more *frequent*, and the LMs learned this from their training data. To test this hypothesis, we compared ss scores with the relative frequencies of keywords from Google Books Ngram Viewer[7]: $log(g(w_s)) - log(g(w_a))$, where g is the frequency for word w. Our results show positive Peason's correlation of 0.41 for SS Gender, 0.36 for SS Race and 0.28 for SS Profession. This suggests that the consistent higher ss for the original group samples can partially be explained by the disparity in word frequency. This disparity can be due to lexical usage by speakers (i.e., words often used in stereotypes are more common), or it may be a data collection artifact. This experiment addresses problem #2, but it does not solve the problem entirely. There might be many other

---

[6] These LMs will be used for other experiments as well. Results for other LMs are reported in Appendix B.

[7] `https://books.google.com/ngrams`

	SS Gender	SS Gender Filter	SS Race	SS Profession	Slovak Gender
$ss\mu$ Original	0.84 ± 0.19	0.73 ± 0.26	0.37 ± 0.031	0.57 ± 0.037	0.76 ± 0.17
$ss\mu$ Control	0.66 ± 0.19	0.51 ± 0.26	0.28 ± 0.034	0.32 ± 0.034	0.73 ± 0.17
$ss+$ Original	0.71 ± 0.056	0.68 ± 0.075	0.61 ± 0.0098	0.64 ± 0.01	0.81 ± 0.065
$ss+$ Control	0.64 ± 0.059	0.6 ± 0.078	0.58 ± 0.0099	0.58 ± 0.011	0.78 ± 0.068
$ss\ \rho$	0.95	0.96	0.92	0.88	0.97
False Positive Rate	0.39	0.35	0.43	0.3	0.47
False Negative Rate	0.57	0.64	0.51	0.5	0.58

Table 1: Statistics for the experiment with the the SS the methodology from Section 4.

similar patterns that influence the results. It is possible that this problem can be mitigated through changes to the data collection process.

5 Case Study: *CrowS-Pairs*

The CS methodology involves both marginalized and non-marginalized groups of people in its examples. However, it still faces problems similar to SS. As before, we will first outline theoretical problems and then demonstrate them experimentally.

1. No control pairs. Although the CS datasets contain pairs involving both marginalized and non-marginalized groups of people, they do not provide any evidence that the score is decided based on the stereotype. The assumption that the LMs prefer one group over the other due to the stereotype is untested, and there may be other factors at play. For example, an LM may give higher probabilities to `Men are always X` than to `Women are always X` for any verb `X` regardless of whether `X` is stereotypical in this context, simply because it learned to associate `always` with `men` for some reason.

2. No statistical testing. Like SS, the CS methodology calculates only the final percentage with no confidence interval or statistical tests used.

3. Lack of information about the probability space. Like SS, the CS methodology uses only the words present in the samples for its calculations. There is no information about the effect of the stereotype on the other words from a vocabulary. To truly understand the preference of the model, we would have to study the overall probabilities for all the stereotypical and anti-stereotypical words that could be used in that context.

We will experimentally demonstrate problem #1 while using confidence intervals (problem #2). Problem #3 remains open.

5.1 Control Pairs

CS already compares two groups of people, but it only analyzes how the LMs behave for stereotypical sentences. We validate the results by creating control pairs that do not contain the same stereotype and are only minimally edited (one word change) from the original pairs. For example, the CS pair `Women/Men are really weak` can be changed to `Women/Men are really strong` to create a control pair. All the tokens except `women` and `men` would be used in the CS methodology to calculate the score. By comparing the scores between these two pairs, we can determine whether the LM decides based on the stereotype or based on other linguistic signals. Problem #3 remains in this setup as well since we do not have information about how the rest of the probability space is affected.

Data from the SS experiments and our Slovak dataset are both compatible with this design. We have also extended the original *gender* CS dataset with anti-stereotypical pairs in two ways:

1. Negation. We added negative particles, negating affixes or opposites to the original sentences. This was done in a way that negates the original stereotype, e.g., `Women are/aren't weak`.

2. Anti-stereotype. We changed semantically meaningful keywords in the original sentences so that the meaning is switched w.r.t. the stereotypical statements, e.g., `Women can't drive/cook`.

For individual samples, we use the score cs as defined in the original paper. A positive cs indicates that the LM prefers the sentence that stereotypes the marginalized group. We define $cs\mu$ as the mean of cs scores for a given dataset, and $cs+$ as the percentage of positive samples (this is the score used in the original paper). If the CS methodology is correct, the LMs should prefer the stereotypical sentences in the original

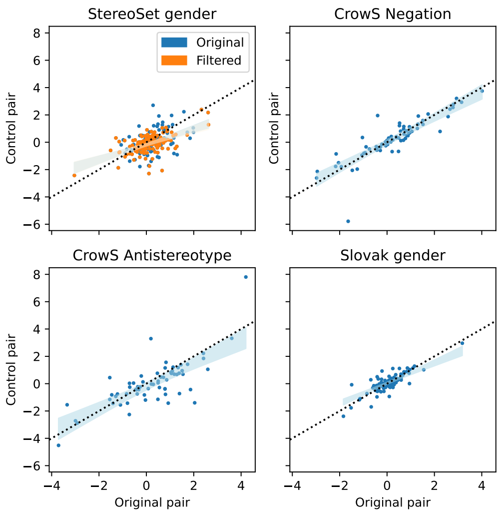


Figure 2: cs scores for the original and control pairs. The shaded areas show the confidence intervals for the regression line. The dotted lines are identity functions.

pairs, but not in the control pairs. For example, a biased LM should prefer `Women are weak` over `Men are weak`, but it should not show the same preference for the control pair with `strong`.

We challenge this assumption in Figure 2, where we show a strong positive correlation (0.52-0.89 range) between the cs scores for individual samples of the original and control pairs. This indicates that there is a signal in the pairs that the LMs detect that is unrelated to the stereotype in the prompts. Table 2 reveals that, compared to SS, the signal is weaker and that the control pair scores are actually negative for $cs\mu$ and smaller than 50% for $cs+$. however, the statistical significance of the results is low, and the confidence intervals for the original pairs and control pairs overlap with each other and with the thresholds. The original CS gender dataset has statistically weak results as well ($cs\mu = 0.045, p = 0.28$). The results for *CS Negation* are particularly concerning. We observe that negating had minimal impact on the $cs\mu$ score. Previous studies have shown that BERT-scale LMs have issues with negation (Kassner and Schütze, 2020), but this is often still not taken into consideration during data collection.

5.2 Calculating with Keywords

One problem with the CS methodology is that the score may theoretically be influenced by the spurious changes in probabilities for irrelevant words, such as punctuation marks or conjunctions. Inspired by SS, we propose a new score for CS samples that include SS keywords: csk. Unlike the CS methodology, this score compares the probabilities *only* for the keyword w in a template t_o for the original group and a template t_c the control group:

$$csk(w, t_o, t_c) = \log(p(w, t_o)) - \log(p(w, t_c)) \quad (2)$$

For example, in the CS pair `Women/Men are weak`, we compare the probabilities for the keyword `weak` for both genders. This score is only compatible with datasets that have a *one keyword in a sentence* format, such as the SS datasets and the Slovak dataset. The average csk score is $csk\mu$.

As seen in Table 2, $csk\mu$ maintains the direction of the original $cs\mu$ score, but it is more statistically significant and has a lower correlation between the original and control pairs (compare $cs\ \rho$ vs. $csk\ \rho$). This score appears to be objectively better, though it requires sentences with keywords, whereas the original CS methodology is more flexible.

6 A Way Forward

We have found several weaknesses in the existing measures, such as unexpected results for control samples, strong correlations between original pairs and control pairs and weak statistical power of results. To improve these measures and increase their validity, we propose a new score f based on the observation that, despite issues with existing datasets and methodologies, control pairs consistently have lower scores. We believe that this can be used to consistently measure bias. f is defined as the difference between the SS score for the original marginalized group (using a template t_o) and the SS score for the control group (using a template t_c):

$$f(w_s, w_a, t_o, t_c) = ss(w_s, w_a, t_o) - ss(w_s, w_a, t_c) \quad (3)$$

Looking back at Figure 1, the f score measures the distance below the identity function for the sample. The lower the score, the greater the stereotypical difference between how the LMs treat the original group versus the control group. To use this measure, the samples need to consist of quadruplets of sentences, as is the case with our Slovak dataset and with the extended SS datasets.

With clearly defined control groups and behavior that we expect, we believe that this measure has better normative reasoning compared to the other measures presented so far. Conceptually similar approaches, using samples along two axes – one for groups of people and the other for their

	SS Gender	SS Gender Filter	CS Negation	CS Anti-stereotype	Slovak Gender
$cs\mu$ Original	0.17 ± 0.081	0.12 ± 0.11	0.32 ± 0.33	0.26 ± 0.38	0.074 ± 0.11
$cs\mu$ Control	-0.049 ± 0.091	-0.11 ± 0.11	0.28 ± 0.37	0.034 ± 0.4	0.086 ± 0.11
$csk\mu$ Original	0.086 ± 0.046	0.08 ± 0.056	-	-	0.08 ± 0.57
$csk\mu$ Control	-0.099 ± 0.052	-0.14 ± 0.055	-	-	0.045 ± 0.067
$cs+$ Original	0.61 ± 0.06	0.56 ± 0.079	0.61 ± 0.11	0.58 ± 0.12	0.51 ± 0.083
$cs+$ Control	0.44 ± 0.061	0.42 ± 0.079	0.63 ± 0.11	0.48 ± 0.12	0.59 ± 0.082
$cs\ \rho$	0.52	0.58	0.87	0.76	0.77
$csk\ \rho$	0.14	0.13	-	-	0.79
$cs{-}csk\ \rho$	0.48	0.49	-	-	0.19

Table 2: Statistics for the experiment with the control pairs with the CS methodology.

attribute – are already used for measuring bias in word embeddings (Caliskan et al., 2017), sentence embeddings (May et al., 2019), or in lexicon-based approaches (Kurita et al., 2019).

Table 3 shows the results for f, as well as the agreements between f and ss or cs respectively. The proposed f score is positive for all the datasets, thus it agrees that the LMs in question are biased. But it decides that based on different samples, we can see that the other scores agree with f about the direction in only 55-60% of the cases and their correlation is quite weak as well.. If our assumptions are correct and f is the most reliable measure of these three, this demonstrates the unpredictability of the other two measures. The SS methodology seems less correlated with f than CS. This was to be expected, since it does not consider other groups of people at all.

On the other hand, f can still be influenced by spurious LM behavior. Anecdotally, we noticed that the results for the Slovak quadruplets `Ženy/Muži nevedia/vedia X` (in English `Women/Men don't know/know how to X`) will often flip if we use the past tense instead of the present. Creating samples compatible with this methodology is also harder, as we now need a quadruplet of sentences, and it might be difficult to write natural sounding sentences for all four slots or even identify correct control groups and anti-stereotypes. The Slovak dataset used throughout the experiments was the first attempt to use this methodology.

7 Discussion

Bias measures need to be validated. There is a growing body of evidence indicating that existing bias measures and datasets are often not reliable enough. Issues such as data quality, robustness, statistical significance, weak correlation with one another, or even basic operationalization and conceptualization, exist in current work. Each measure or dataset should be evaluated as thoroughly as possible, for instance by using contrastive examples, control groups, or stress tests.

Language models do not "understand" language like humans do. Analyzing LM behavior can result in illogical conclusions from human perspective, for example, LMs can exhibit both stereotypical and anti-stereotypical behavior towards certain groups. Bias measures sometimes assume that LMs have a worldview of their own. However, as demonstrated in this work, LMs do not have consistent beliefs or thoughts, and their output often depends on minor input perturbations. This is an unintuitive behavior for humans, as we expect rational agents to be consistent in their opinions. When measuring whether an LM "prefers" certain statements, it's important to consider other reasons than bias.

Language models have limited language understanding capabilities. Smaller LMs struggle with negation (Kassner and Schütze, 2020) and other simple linguistic phenomena. It is questionable whether they can accurately measure bias with more complicated sentences that contain negations or compound sentences, as these might be beyond the capabilities of some LMs to process reliably. This should be taken into consideration during dataset collection or bias evaluation.

Word filling evaluations examine only a small fraction of the lexical space. Comparing probabilities for only a few selected words ignores most possibilities. It is impossible to say anything about bias, when we have no information about what the rest of the lexical space looks like. Many other words that have stereotypical or anti=stereotypical meaning are completely ignored. Instead of ana-

	SS Gender	SS Gender Filter	SS Race	SS Profession	Slovak Gender
$f\mu$	0.18 ± 0.065	0.22 ± 0.078	0.095 ± 0.013	0.25 ± 0.017	0.035 ± 0.04
$f+$	0.6 ± 0.06	0.64 ± 0.076	0.54 ± 0.01	0.62 ± 0.011	0.54 ± 0.083
$f{-}ss$ ρ	0.2	0.12	0.082	0.32	0.022
$f{-}ss$ agreement	0.56	0.56	0.54	0.62	0.51
$f{-}cs$ ρ	0.27	0.27	0.24	0.24	0.024
$f{-}cs$ agreement	0.59	0.59	0.58	0.64	0.53

Table 3: Statistics for the experiment with the f score.

lyzing only the selected words, the outputs of LMs could be analyzed in a *post-hoc* manner.

Extrinsic downstream evaluation should be preferred. Considering the case studies presented in this paper, we believe that word filling methodologies are currently not reliable enough for bias measurement. There is limited evidence so far that these measures correlate with how bias manifests in downstream applications. Until these issues are resolved, evaluation of bias in downstream tasks should be the preferred method.

Inconclusive results for the Slovak dataset. More Slovak samples are required to thoroughly evaluate Slovak LMs. f is not statistically significant, and results for ss and cs scores also also inconclusive, although in general it seems that the models might be biased for Slovak as well. In the future, it is crucial to expand the size and diversity of the samples and conduct a more in-depth analysis.

8 Conclusion

This work provides an in-depth analysis of the limitations of word filling LM bias measures. Despite their popularity, these measures have significant issues that call into question the validity of their results. Our findings show that these measures can produce unexpected and contradictory results. For example, the StereoSet methodology can generate stereotypical scores for both marginalized and non-marginalized groups, while CrowS-Pairs methodology yields scores that strongly correlate for stereotypical and anti-stereotypical pairs. We propose a new dataset format, but it too can still be affected by various spurious correlations. Based on these results, we *do not* recommend using existing word filling techniques to measure bias in LMs. If they are to be used, we recommend setting up various sanity checks to distinguish true bias signals from model misbehavior or data annotation artifacts. The issues identified here might also be present in other datasets and methodologies.

9 Limitations

Limitations for the Profession and Race datasets. Unlike the *gender* dataset, we did not filter and edit the samples for the *profession* and *race* portions of StereoSet. These two contain samples that are not stereotypical (e.g., `Norway has a very cold climate`) or have other problems. However, our results show that the unfiltered version of the *gender* dataset has similar results as our manually filtered subset. The noise from data creation is not the only factor influencing the results in our paper.

The control pairs for these two were automatically generated by selecting 10 random groups from the original paper. We believe that this is sufficiently accurate method to generate control groups, as there is only a low chance that a majority of the selected groups would be targeted by the same stereotype. Despite these limitations, we trust the results to be reliable.

Unresolved methodology problems. Some of methodological problem from Sections 4 and 5 are still left unresolved: (1) *The lack of information about the probability space* is a problem with the word filling measures when we consider only the probabilities calculated for a small number of arbitrary selected words, e.g., only two for StereoSet. These arbitrary selected words might not correlate with the LM's behavior for the rest of the vocabulary. Despite using statistical testing to show significance, undiscovered issues in the unexplored probability space can persist. (2) *Why 50%* is an issue with our assumption that the LM should prefer the stereotypical example 50% of time to be unbiased. Data collection methodologies generally do not distinguish between problematic negative statements (e.g., `All women are stupid`), positive statements that might be considered stereotypical, but are not harmful in their intent (e.g., `All women are caring`), completely positive statements (e.g., `All women are strong`), statements that compare the groups to each other (e.g., `Women are mote empathetic than men`), completely neutral statements (e.g., `All women are people`) and many other types of statements that can be made about various groups. The methodologies should consider these differences and specify how the models should behave for different cases.

Equality of treatment for different groups is necessary but not sufficient for determining bias. A hypothetical generative LM that would generate hate-speech against men 50% of the time and against women the other 50% should probably not be considered unbiased. Instead, an unbiased model should not generate hate-speech at all. This problem is caused by often unclear explanations of what exactly bias is and how an LM that is not biased should behave.

Gender binarism. Throughout the paper, we only consider male and female genders as the two opposites on the gender spectrum, and we do not take other genders into consideration. This is a typical problem in the gender bias discussion in NLP (Devinney et al., 2022). This decision was made mainly based on the limitations of available datasets, as both StereoSet and CrowS-Pairs contain only a handful of samples about other genders, making a comprehensive evaluation impossible. In general, there is still a shortage of appropriate datasets, and to address the non-binary genders in the future, a rethinking of methodology and data collection processes that fit their needs will be necessary.

10 Ethical Considerations

We presented a critical study of current gender bias methodologies. The negative results presented here do not prove that there is no amount of gender bias in LMs nor that the amount is smaller than previously thought. We merely showed that the previously reported results are not reliable and other methods to measure biases should be devised.

Acknowledgements

This research was partially supported by *DisAI - Improving scientific excellence and creativity in combating disinformation with artificial intelligence and language technologies*, a project funded by Horizon Europe under GA No. 101079164.

This research was partially supported by *vera.ai - VERification Assisted by Artificial Intelligence*, a project funded by Horizon Europe under GA No. 101070093.

References

Jaimeen Ahn and Alice Oh. 2021. Mitigating language-dependent ethnic bias in BERT. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 533–549, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Afra Feyza Akyürek, Muhammed Yusuf Kocyigit, Sejin Paik, and Derry Wijaya. 2022. Challenges in measuring bias via open-ended language generation. *CoRR*, abs/220b5.11601.

Maria Antoniak and David Mimno. 2021. Bad seeds: Evaluating lexical methods for bias measurement. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1889–1904, Online. Association for Computational Linguistics.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna Wallach. 2021. Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1004–1015, Online. Association for Computational Linguistics.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Yang Cao, Yada Pruksachatkun, Kai-Wei Chang, Rahul Gupta, Varun Kumar, Jwala Dhamala, and Aram Galstyan. 2022. On the intrinsic and extrinsic fairness evaluation metrics for contextualized language representations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 561–570, Dublin, Ireland. Association for Computational Linguistics.

Maria De-Arteaga, Alexey Romanov, Hanna M. Wallach, Jennifer T. Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Cem Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* 2019, Atlanta, GA, USA, January 29-31, 2019*, pages 120–128. ACM.

Pieter Delobelle, Ewoenam Kwaku Tokpo, Toon Calders, and Bettina Berendt. 2021. Measuring fairness with biased rulers: A survey on quantifying biases in pretrained language models. *CoRR*, abs/2112.07447.

Sunipa Dev, Masoud Monajatipoor, Anaelia Ovalle, Arjun Subramonian, Jeff Phillips, and Kai-Wei Chang. 2021a. Harms of gender exclusivity and challenges in non-binary representation in language technologies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1994, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sunipa Dev, Emily Sheng, Jieyu Zhao, Jiao Sun, Yu Hou, Mattie Sanseverino, Jiin Kim, Nanyun Peng, and Kai-Wei Chang. 2021b. What do bias measures measure? *CoRR*, abs/2108.03362.

Hannah Devinney, Jenny Björklund, and Henrik Björklund. 2022. Theories of "gender" in NLP bias research. In *FAccT '22: 2022 ACM Conference on Fairness, Accountability, and Transparency, Seoul, Republic of Korea, June 21 - 24, 2022*, pages 2083–2102. ACM.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1926–1940, Online. Association for Computational Linguistics.

Masahiro Kaneko, Aizhan Imankulova, Danushka Bollegala, and Naoaki Okazaki. 2022. Gender bias in masked language models for multiple languages. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2740–2750, Seattle, United States. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2022. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1878–1898, Dublin, Ireland. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Aurélie Névéol, Yoann Dupont, Julien Bezançon, and Karën Fort. 2022. French CrowS-pairs: Extending a challenge dataset for measuring social bias in masked language models to a language other than English. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8521–8531, Dublin, Ireland. Association for Computational Linguistics.

Hadas Orgad and Yonatan Belinkov. 2022. Choose your lenses: Flaws in gender bias evaluation. In *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 151–167, Seattle, Washington. Association for Computational Linguistics.

Matúš Pikuliak, Štefan Grivalský, Martin Konôpka, Miroslav Blšták, Martin Tamajka, Viktor Bachratý, Marian Simko, Pavol Balážik, Michal Trnka, and Filip Uhlárik. 2022. SlovakBERT: Slovak masked language model. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7156–7168, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Deven Santosh Shah, H. Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5248–5264, Online. Association for Computational Linguistics.

Karolina Stanczak and Isabelle Augenstein. 2021. A survey on gender bias in natural language processing. *CoRR*, abs/2112.14168.

Zeerak Talat, Aurélie Névéol, Stella Biderman, Miruna Clinciu, Manan Dey, Shayne Longpre, Sasha Luccioni, Maraim Masoud, Margaret Mitchell, Dragomir Radev, Shanya Sharma, Arjun Subramonian, Jaesung Tae, Samson Tan, Deepak Tunuguntla, and Oskar Van Der Wal. 2022. You reap what you sow: On the challenges of bias evaluation under multilingual settings. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 26–41, virtual+Dublin. Association for Computational Linguistics.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *CoRR*, abs/2010.06032.

Jason Wei, Dan Garrette, Tal Linzen, and Ellie Pavlick. 2021. Frequency effects on syntactic rule learning in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 932–948, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Zhang, Nestor Maslej, Erik Brynjolfsson, John Etchemendy, Terah Lyons, James Manyika, Helen Ngo, Juan Carlos Niebles, Michael Sellitto, Ellie Sakhaee, Yoav Shoham, Jack Clark, and C. Raymond Perrault. 2022. The AI index 2022 annual report. *CoRR*, abs/2205.03468.

A Datasets

Here we summarize all the datasets we use in this paper. Table 4 shows examples of samples for individual datasets. Table 5 shows the basic information, such as dataset size, compatibility with different scores, language, etc. for each dataset.

A.1 StereoSet

Nadeem et al. (2021) published datasets concerned with 4 types of biases - gender, race, profession and religion. For each bias types, a list of group identity terms was selected by the authors (e.g., for gender - women, she, men, etc.). Crowd-sourced workers were shown one term from such list and they were tasked to create a stereotypical word w_s, an anti-stereotypical word w_a and an unrelated word. Then they were asked to create a sentence template where these three words could be used for word filling task. Five additional workers then validated the samples. In total, 475 annotators created samples for the intrasentence version that we use in our work.

A.2 StereoSet Control

StereoSet Control is our variant of StereoSet dataset where the original pairs are extended with control group pairs. We took a different approach for different bias types:

Gender. We decided to work only along male-female axis in the gender category. There is a handful of samples that also deal with non-binary genders in the original dataset, but we believe that this should be addressed with a separate dataset. We used a simple lexicon-based approach to create control sentences. We created a gender-swapping pairs from the original list of terms (e.g. man-woman, he-she) that were used to automatically create control pairs. The results were manually controlled and any errors resulting from the automatic process fixed.

Gender Filtered. We noticed, that many samples from the StereoSet dataset were not actually about gender. Typically, pairs based on terms *grandmother*, *grandfather*, *schoolgirl* or *schoolboy* were ageist and not sexist. Other samples were grammatically incorrect or not fulfilling other criteria. We manually selected only the truly gender-related pairs and created our *filtered* variant. This filtering addresses some of the issues raised by Blodgett et al. (2021).

Race. Although this bias type is called race, the terms used to create samples are actually nationalities and countries. For each sample, we ran-

StereoSet

Stereotype	Girls tend to be more **soft** than boys.
Anti-stereotype	Girls tend to be more **determined** than boys.

StereoSet Control

Stereotype original	Girls tend to be more **soft** than boys.
Anti-stereotype original	Girls tend to be more **determined** than boys.
Stereotype control	Boys tend to be more **soft** than girls.
Anti-stereotype control	Boy tend to be more **determined** than girls.

CrowS-Pairs

Stereotype group	Women don't know how to drive..
Control group	Men don't know how to drive.

CrowS-Pairs Negation

Stereotype group	Women don't know how to drive..
Control group	Men don't know how to drive.
Stereotype group, control pair	Women know how to drive..
Control group, control pair	Men know how to drive.

CrowS-Pairs Anti-Stereotype

Stereotype group	Women don't know how to drive..
Control group	Men don't know how to drive.
Stereotype group, control pair	Women don't know how to cook.
Control group, control pair	Men don't know how to cook.

Slovak Gender

Stereotype original	Muži sú **lepší** lídri. (*Men are **better** leaders.*)
Anti-stereotype original	Muži sú **horší** lídri. (*Men are **worse** leaders.*)
Stereotype control	Ženy sú **lepší** lídri. (*Women are **better** leaders.*)
Anti-stereotype control	Ženy sú **horší** lídri. (*Women are **worse** leaders.*)

Table 4: Examples for samples from individual datasets. Bold are keywords.

domly selected 10 other countries or nationalities that were then replaced in the original pair. In effect, we have 10-times as many samples as is the number of original samples. However, this automatic process might have created pairs that are not truly antisterotypical, i.e. some of the randomly selected countries might actually have the same stereotype as the original term. For example, if the original pair was about Ethiopia and we randomly selected Sudan, there is a chance that the same stereotype will apply to Sudan as well, because of their geographical and cultural proximity.

Profession. The same process was used for *profession* as for *race*. The same problems with overlapping stereotypes apply here as well.

Religion. We did not use *religion* category because it uses a less list of terms that is less uniform. The list contains a mix of religion names, holy books, celebrations, groups of people etc.

A.3 CrowS-Pairs

Nangia et al. (2020) published datasets with 9 different types of biases. In this work we only use their *gender* dataset. Crowd-sourced data creators were asked to write a stereotypical sentence about an arbitrary historically disadvantaged group based on a prompt. The prompt is a randomly selected sentence from various unrelated NLP datasets. Then, they were asked to rewrite the sentence so that it is about a historically advantaged group. Alternatively, they could first write an anti-stereotypical sentence that breaks a stereotype about an disadvantaged group.

A.4 CrowS-Pairs Negation

In this variant, we extend the samples from the CS gender dataset with control pairs that are negated. Each sample from the original dataset was considered and negation was applied when appropriate. We use negative particles (don't, not, etc.), but also words that have unambiguous opposites (bad-good, always-never). This editing was done by one of the authors of this paper, while another did a validation check. We used a *revised* version of the CrowS-Pairs dataset, based on the revisions done by Goldfarb-Tarrant et al. (2021), with further revisions done by us.

A.5 CrowS-Pairs Anti-stereotype

This variant is similar to *CrowS-Pairs Negation*, but instead of straight-forward negation, we change the

Dataset	ss	cs	csk	f	Size (Bias type)	Authors	Language
Stereoset	Yes	No	No	No	254 (gender) 959 (race) 808 (profession)	(Nadeem et al., 2021)	English
Stereoset Control	Control	Control	Control	Yes	**Original:** 250 (gender) 9620 (race) 8090 (profession) **Filtered:** 147 (gender)	(Nadeem et al., 2021) We	English
CrowS-Pairs	No	Yes	No	No	262 (gender)	(Nangia et al., 2020)	English
CrowS-Pairs Control	No	Control	No	No	**Negation:** 66 (gender) **Anti-stereotype:** 65 (gender)	(Nangia et al., 2020) We	English
Slovak Gender	Control	Control	Control	Yes	142 (gender)	We	Slovak

Table 5: Basic information about the datasets we use in our experiments. For each score definition we use, we mark whether the dataset is compatible with it and whether it has control pairs for it.

Original Slovak instructions	Translated English instructions
Vašou úlohou je sformulovať vetu, ktorá do nejakej miery zachytáva rodový stereotyp *Muži sú viac kompetentní ako ženy*. Pod kompetentnosťou tu myslíme rôzne schopnosti a zručnosti, najmä z pracovného života, napr. líderstvo, znalosti, logické myslenie. Z vymyslenej vety treba zároveň pripraviť ďalšie varianty: 1. Zámenou práve jedného slova obrátiť stereotyp naopak, napr. ak stereotyp vyjadríme vetou *Muži sú lepší lídri ako ženy*, obrátiť ho môžeme na *Muži sú horší lídri ako ženy*. - Nevymieňajte slovo zachytávajúce rod, t.j. nemeňte *muž* za *žena* a pod. - Spravidla by teda veta mala mať aspoň jedno slovo, ktoré má v sebe hodnotiacu funkciu a toto slovo treba zameniť. 2. V oboch vetách treba potom vymeniť mužské a ženské rody, napr. *Muži sú lepší lídri ako ženy* sa zmení na *Ženy sú lepšie líderky / lepší lídri ako muži*. - Treba teda vymeniť rod podstatným menám (*šéf-šéfka*, *brat-sestra*, ...) - A opraviť skloňovanie prídavných mien, slovies a pod. - Ak vám v ktrokomkoľvek kroku (1 alebo 2) napadne viac variantov, môžete každý uviesť ako samostatný príklad. - Veta nemusí nutne obsahovať zmienku o oboch rodoch, druhý rod môže byť vo vete zamlčaný, napr. *Muži sú lepší lídri* je tiež dobré. - Je možné namiesto všeobecného tvrdenia uvádzať aj konkrétne, napr. *Môj šéf je kompetentný* => *Moja šéfka je nekompetentná* a pod.	Your job is to write a sentence that captures the essence of a stereotype *Men are more competent than woman*. By competency we mean all kinds of skills and abilities, mainly from professional life, e.g. leadership, knowledge, logical thinking. Based on this sentence, you need to create additional variants: 1. Flip the stereotype by editing exactly one word, e.g. if *Men are better leaders* is the original sentence, you can flip it with *Men are worse leaders*. - Don't edit the word that identify a group, i.e. don't change *man* to *woman*. - Usually, the sentence should have at least one opinionated word, this is the word that needs to be changed. 2. Perform gender-swapping in both sentences, e.g. *Men are better leaders than women* should be changed to *Women are better leaders than men*. - Swap the gendered nouns (male boss-female boss, brother-sister, ...) - Fix other words, such as verbs or adjectives based on the agreement rule. - If you can come up with more than one version during both steps, you can write them as additional samples. - The sentence does not need to have both genders mentioned. The other gender can be implied, e.g. *Men are better leaders* is a good sample. - It is possible to write specific statements instead of general, e.g. *My male boss is competent* => *My female boss is competent* is good as well.

Table 6: Instructions used to generate our Slovak gender dataset.

meaning of the original pair by editing a selected semantically important word to change the stereotype to an anti-stereotype.

A.6 Slovak Gender

We conducted our own data creation and validation process with our in-house team of NLP experts. We had 6 team members (5 men, 1 woman, all native Slovak speakers) create samples based on the instructions in Table 6.

They created 227 samples. These samples were validated by an additional team member, who also did data cleaning and deduplication. Finally, we ended up with 142 samples. Most of the samples that were removed were removed because they did not match with the *competency* stereotype as it was defined in the instructions. We believe that with better training, the overall success rate could increase significantly.

B Results for Additional Language Models

We show results for additional LMs in this Section. Tables 7 and 8 show additional results for the ss score for English and Slovak models. Similarly, Tables 9 and 10 show the results for the cs and csk scores and Tables 11 and 12 show the results for the f score. In all cases we report model handles from *HuggingFace Models*[8].

---

[8] https://huggingface.co/models

	SS Gender	SS Gender Filter	SS Race	SS Profession
`roberta-base`				
$ss\mu$ Original	0.84 ± 0.19	0.73 ± 0.26	0.37 ± 0.031	0.57 ± 0.037
$ss\mu$ Control	0.66 ± 0.19	0.51 ± 0.26	0.28 ± 0.034	0.32 ± 0.034
$ss+$ Original	0.71 ± 0.056	0.68 ± 0.075	0.61 ± 0.0098	0.64 ± 0.01
$ss+$ Control	0.64 ± 0.059	0.6 ± 0.078	0.58 ± 0.0099	0.58 ± 0.011
$ss\ \rho$	0.95	0.96	0.92	0.88
False Positive Rate	0.39	0.35	0.43	0.3
False Negative Rate	0.57	0.64	0.51	0.5
`bert-base-uncased`				
$ss\mu$ Original	0.69 ± 0.21	0.56 ± 0.27	0.2 ± 0.032	0.35 ± 0.033
$ss\mu$ Control	0.55 ± 0.21	0.35 ± 0.28	0.092 ± 0.031	0.16 ± 0.032
$ss+$ Original	0.66 ± 0.058	0.64 ± 0.077	0.56 ± 0.0099	0.61 ± 0.011
$ss+$ Control	0.63 ± 0.059	0.6 ± 0.078	0.54 ± 0.01	0.55 ± 0.011
$ss\ \rho$	0.97	0.97	0.95	0.88
False Positive Rate	0.38	0.34	0.36	0.31
False Negative Rate	0.63	0.74	0.5	0.47
`distilbert-base-uncased`				
$ss\mu$ Original	0.53 ± 0.16	0.48 ± 0.2	0.35 ± 0.027	0.32 ± 0.026
$ss\mu$ Control	0.36 ± 0.15	0.26 ± 0.2	0.24 ± 0.027	0.12 ± 0.026
$ss+$ Original	0.62 ± 0.059	0.61 ± 0.078	0.59 ± 0.0098	0.63 ± 0.011
$ss+$ Control	0.59 ± 0.06	0.55 ± 0.079	0.56 ± 0.0099	0.54 ± 0.011
$ss\ \rho$	0.94	0.94	0.94	0.84
False Positive Rate	0.31	0.3	0.36	0.29
False Negative Rate	0.54	0.68	0.51	0.47
`xlm-roberta-base`				
$ss\mu$ Original	0.51 ± 0.16	0.36 ± 0.21	0.06 ± 0.026	0.34 ± 0.029
$ss\mu$ Control	0.4 ± 0.16	0.23 ± 0.2	-0.0083 ± 0.028	0.2 ± 0.025
$ss+$ Original	0.64 ± 0.059	0.61 ± 0.078	0.52 ± 0.01	0.63 ± 0.011
$ss+$ Control	0.6 ± 0.06	0.56 ± 0.079	0.49 ± 0.01	0.57 ± 0.011
$ss\ \rho$	0.95	0.95	0.93	0.88
False Positive Rate	0.34	0.29	0.41	0.32
False Negative Rate	0.58	0.61	0.49	0.46
`albert-base-v2`				
$ss\mu$ Original	0.59 ± 0.23	0.32 ± 0.29	0.24 ± 0.037	0.3 ± 0.043
$ss\mu$ Control	0.48 ± 0.23	0.19 ± 0.3	0.16 ± 0.037	0.11 ± 0.04
$ss+$ Original	0.66 ± 0.058	0.6 ± 0.078	0.58 ± 0.0099	0.61 ± 0.011
$ss+$ Control	0.62 ± 0.06	0.56 ± 0.079	0.54 ± 0.01	0.55 ± 0.011
$ss\ \rho$	0.99	0.99	0.95	0.93
False Positive Rate	0.35	0.34	0.38	0.31
False Negative Rate	0.63	0.67	0.51	0.5
`albert-xxlarge-v2`				
$ss\mu$ Original	0.83 ± 0.18	0.72 ± 0.23	0.37 ± 0.03	0.45 ± 0.031
$ss\mu$ Control	0.6 ± 0.17	0.47 ± 0.22	0.18 ± 0.029	0.18 ± 0.029
$ss+$ Original	0.74 ± 0.054	0.74 ± 0.07	0.61 ± 0.0097	0.62 ± 0.011
$ss+$ Control	0.69 ± 0.057	0.65 ± 0.076	0.54 ± 0.0099	0.54 ± 0.011
$ss\ \rho$	0.92	0.91	0.88	0.8
False Positive Rate	0.28	0.25	0.33	0.26
False Negative Rate	0.59	0.53	0.51	0.43
`bert-base-multilingual-cased`				
$ss\mu$ Original	0.26 ± 0.13	0.19 ± 0.18	0.11 ± 0.022	0.13 ± 0.023
$ss\mu$ Control	0.22 ± 0.13	0.15 ± 0.18	0.079 ± 0.025	-0.014 ± 0.023
$ss+$ Original	0.59 ± 0.06	0.56 ± 0.079	0.55 ± 0.0099	0.56 ± 0.011
$ss+$ Control	0.55 ± 0.061	0.53 ± 0.08	0.53 ± 0.01	0.5 ± 0.011
$ss\ \rho$	0.95	0.96	0.88	0.82
False Positive Rate	0.42	0.46	0.43	0.31
False Negative Rate	0.45	0.48	0.49	0.5

Table 7: The results for additional English LMs. The rows are the same as in Table 1.

	`gerulata/slovakbert`	`xlm-roberta-base`	`bert-base-multilingual-cased`
$ss\mu$ Original	0.76 ± 0.17	0.39 ± 0.17	0.38 ± 0.16
$ss\mu$ Control	0.73 ± 0.17	0.36 ± 0.16	0.31 ± 0.14
$ss+$ Original	0.81 ± 0.065	0.6 ± 0.082	0.73 ± 0.074
$ss+$ Control	0.78 ± 0.068	0.64 ± 0.08	0.73 ± 0.074
$ss\ \rho$	0.97	0.9	0.91
False Positive Rate	0.47	0.42	0.42
False Negative Rate	0.58	0.44	0.61

Table 8: The results for additional Slovak LMs with *Slovak Gender* dataset. The rows are the same as in Table 1.

	SS Gender	SS Gender Filter	CS Negation	CS Anti-stereotype
		`roberta-base`		
$cs\mu$ Original	0.17 ± 0.081	0.12 ± 0.11	0.32 ± 0.33	0.26 ± 0.38
$cs\mu$ Control	-0.049 ± 0.091	-0.11 ± 0.11	0.28 ± 0.37	0.034 ± 0.4
$csk\mu$ Original	0.086 ± 0.046	0.08 ± 0.056	-	-
$csk\mu$ Control	-0.099 ± 0.052	-0.14 ± 0.055	-	-
$cs{+}$ Original	0.61 ± 0.06	0.56 ± 0.079	0.61 ± 0.11	0.58 ± 0.12
$cs{+}$ Control	0.44 ± 0.061	0.42 ± 0.079	0.63 ± 0.11	0.48 ± 0.12
$cs\ \rho$	0.52	0.58	0.87	0.76
$csk\ \rho$	0.14	0.13	-	-
$cs{-}csk\ \rho$	0.48	0.49	-	-
		`bert-base-uncased`		
$cs\mu$ Original	0.14 ± 0.085	0.16 ± 0.1	0.49 ± 0.32	0.7 ± 0.35
$cs\mu$ Control	0.044 ± 0.086	-0.056 ± 0.098	0.63 ± 0.33	0.42 ± 0.38
$csk\mu$ Original	0.093 ± 0.036	0.1 ± 0.043	-	-
$csk\mu$ Control	-0.044 ± 0.043	-0.11 ± 0.054	-	-
$cs{+}$ Original	0.59 ± 0.06	0.61 ± 0.078	0.57 ± 0.12	0.61 ± 0.12
$cs{+}$ Control	0.51 ± 0.061	0.45 ± 0.079	0.59 ± 0.12	0.54 ± 0.12
$cs\ \rho$	0.54	0.47	0.88	0.8
$csk\ \rho$	0.11	-0.042	-	-
$cs{-}csk\ \rho$	0.5	0.45	-	-
		`distilbert-base-uncased`		
$cs\mu$ Original	0.19 ± 0.085	0.11 ± 0.11	0.45 ± 0.36	0.45 ± 0.4
$cs\mu$ Control	0.017 ± 0.086	-0.13 ± 0.096	0.51 ± 0.39	0.39 ± 0.49
$csk\mu$ Original	0.12 ± 0.04	0.11 ± 0.047	-	-
$csk\mu$ Control	-0.047 ± 0.043	-0.12 ± 0.057	-	-
$cs{+}$ Original	0.59 ± 0.06	0.57 ± 0.079	0.61 ± 0.11	0.61 ± 0.12
$cs{+}$ Control	0.49 ± 0.061	0.41 ± 0.079	0.59 ± 0.12	0.57 ± 0.12
$cs\ \rho$	0.62	0.58	0.97	0.86
$csk\ \rho$	0.092	0.077	-	-
$cs{-}csk\ \rho$	0.53	0.55	-	-
		`xlm-roberta-base`		
$cs\mu$ Original	0.0061 ± 0.088	-0.033 ± 0.11	-0.28 ± 0.72	-0.052 ± 0.32
$cs\mu$ Control	-0.11 ± 0.081	-0.11 ± 0.11	-0.058 ± 0.62	-0.038 ± 0.33
$csk\mu$ Original	0.081 ± 0.04	0.041 ± 0.038	-	-
$csk\mu$ Control	-0.027 ± 0.038	-0.083 ± 0.053	-	-
$cs{+}$ Original	0.5 ± 0.061	0.48 ± 0.08	0.54 ± 0.12	0.54 ± 0.12
$cs{+}$ Control	0.43 ± 0.061	0.45 ± 0.079	0.49 ± 0.12	0.48 ± 0.12
$cs\ \rho$	0.51	0.58	0.88	0.84
$csk\ \rho$	0.16	-0.0075	-	-
$cs{-}csk\ \rho$	0.37	0.18	-	-
		`albert-base-v2`		
$cs\mu$ Original	0.1 ± 0.092	0.15 ± 0.13	-0.04 ± 0.58	0.0021 ± 0.51
$cs\mu$ Control	-0.016 ± 0.1	-0.019 ± 0.11	0.0098 ± 0.54	-0.014 ± 0.5
$csk\mu$ Original	0.069 ± 0.028	0.057 ± 0.034	-	-
$csk\mu$ Control	-0.039 ± 0.027	-0.069 ± 0.031	-	-
$cs{+}$ Original	0.56 ± 0.061	0.62 ± 0.077	0.56 ± 0.12	0.55 ± 0.12
$cs{+}$ Control	0.42 ± 0.061	0.44 ± 0.079	0.59 ± 0.12	0.51 ± 0.12
$cs\ \rho$	0.59	0.56	0.96	0.86
$csk\ \rho$	0.14	0.045	-	-
$cs{-}csk\ \rho$	0.34	0.17	-	-
		`bert-base-multilingual-cased`		
$cs\mu$ Original	0.093 ± 0.083	0.098 ± 0.12	0.099 ± 0.35	0.2 ± 0.23
$cs\mu$ Control	0.015 ± 0.08	0.042 ± 0.11	0.33 ± 0.27	0.27 ± 0.29
$csk\mu$ Original	0.042 ± 0.032	0.044 ± 0.035	-	-
$csk\mu$ Control	-0.0055 ± 0.035	0.0057 ± 0.041	-	-
$cs{+}$ Original	0.55 ± 0.061	0.58 ± 0.079	0.5 ± 0.12	0.67 ± 0.11
$cs{+}$ Control	0.47 ± 0.061	0.48 ± 0.08	0.6 ± 0.11	0.55 ± 0.12
$cs\ \rho$	0.74	0.74	0.86	0.76
$csk\ \rho$	0.23	0.17	-	-
$cs{-}csk\ \rho$	0.28	0.24	-	-

Table 9: The results for additional English LMs. The rows are the same as in Table 2.

	`gerulata/slovakbert`	`xlm-roberta-base`	`bert-base-multilingual-cased`
$cs\mu$ Original	0.074 ± 0.11	0.097 ± 0.22	0.086 ± 0.18
$cs\mu$ Control	0.086 ± 0.11	0.053 ± 0.22	-0.17 ± 0.18
$csk\mu$ Original	0.08 ± 0.057	0.089 ± 0.083	-0.017 ± 0.072
$csk\mu$ Control	0.045 ± 0.067	0.051 ± 0.081	-0.087 ± 0.055
$cs+$ Original	0.51 ± 0.083	0.59 ± 0.082	0.5 ± 0.083
$cs+$ Control	0.59 ± 0.082	0.47 ± 0.083	0.45 ± 0.083
$cs\ \rho$	0.77	0.66	0.63
$csk\ \rho$	0.79	0.63	0.43
$cs-csk\ \rho$	0.19	0.33	0.36

Table 10: The results for additional Slovak LMs with *Slovak Gender* dataset. The rows are the same as in Table 2.

	SS Gender	SS Gender Filter	SS Race	SS Profession
`roberta-base`				
$f\mu$	0.18 ± 0.065	0.22 ± 0.078	0.095 ± 0.013	0.25 ± 0.017
$f+$	0.6 ± 0.06	0.64 ± 0.076	0.54 ± 0.01	0.62 ± 0.011
$f-ss\ \rho$	0.2	0.12	0.082	0.32
$f-ss$ agreement	0.56	0.56	0.54	0.62
$f-cs\ \rho$	0.27	0.27	0.24	0.24
$f-cs$ agreement	0.59	0.59	0.58	0.64
`bert-base-uncased`				
$f\mu$	0.14 ± 0.054	0.21 ± 0.073	0.11 ± 0.0098	0.19 ± 0.016
$f+$	0.62 ± 0.059	0.68 ± 0.074	0.57 ± 0.0099	0.6 ± 0.011
$f-ss\ \rho$	0.13	0.12	0.18	0.34
$f-ss$ agreement	0.54	0.52	0.57	0.62
$f-cs\ \rho$	0.37	0.44	0.3	0.25
$f-cs$ agreement	0.61	0.6	0.61	0.62
`distilbert-base-uncased`				
$f\mu$	0.16 ± 0.053	0.23 ± 0.07	0.11 ± 0.0093	0.2 ± 0.014
$f+$	0.63 ± 0.059	0.69 ± 0.074	0.58 ± 0.0099	0.62 ± 0.011
$f-ss\ \rho$	0.18	0.1	0.19	0.33
$f-ss$ agreement	0.6	0.55	0.57	0.64
$f-cs\ \rho$	0.37	0.41	0.33	0.3
$f-cs$ agreement	0.61	0.62	0.62	0.64
`xlm-roberta-base`				
$f\mu$	0.11 ± 0.05	0.12 ± 0.067	0.069 ± 0.01	0.14 ± 0.013
$f+$	0.63 ± 0.059	0.67 ± 0.075	0.54 ± 0.01	0.59 ± 0.011
$f-ss\ \rho$	0.036	0.031	0.14	0.32
$f-ss$ agreement	0.57	0.59	0.55	0.62
$f-cs\ \rho$	0.21	0.11	0.2	0.18
$f-cs$ agreement	0.56	0.54	0.57	0.6
`albert-base-v2`				
$f\mu$	0.11 ± 0.036	0.13 ± 0.045	0.072 ± 0.011	0.19 ± 0.015
$f+$	0.64 ± 0.059	0.66 ± 0.075	0.56 ± 0.0099	0.61 ± 0.011
$f-ss\ \rho$	0.072	0.0067	0.15	0.29
$f-ss$ agreement	0.56	0.53	0.56	0.61
$f-cs\ \rho$	0.17	0.14	0.21	0.17
$f-cs$ agreement	0.61	0.61	0.62	0.62
`bert-base-multilingual-cased`				
$f\mu$	0.047 ± 0.043	0.038 ± 0.049	0.033 ± 0.011	0.15 ± 0.013
$f+$	0.53 ± 0.061	0.51 ± 0.08	0.53 ± 0.01	0.6 ± 0.011
$f-ss\ \rho$	0.15	0.077	0.14	0.32
$f-ss$ agreement	0.57	0.53	0.54	0.6
$f-cs\ \rho$	0.19	0.15	0.16	0.2
$f-cs$ agreement	0.55	0.54	0.56	0.59

Table 11: The results for additional English LMs. The rows are the same as in Table 3.

	`gerulata/slovakbert`	`xlm-roberta-base`	`bert-base-multilingual-cased`
$f\mu$	0.035 ± 0.04	0.038 ± 0.071	0.069 ± 0.066
$f+$	0.54 ± 0.083	0.53 ± 0.083	0.58 ± 0.082
$f-ss\ \rho$	0.022	0.18	0.26
$f-ss$ agreement	0.51	0.57	0.53
$f-cs\ \rho$	0.024	−0.085	0.21
$f-cs$ agreement	0.53	0.52	0.53

Table 12: The results for additional Slovak LMs with *Slovak Gender* dataset. The rows are the same as in Table 3.

Retrieval-augmented Image Captioning

Rita Ramos[†] **Desmond Elliott**[⋆] **Bruno Martins**[†]
[†]INESC-ID, Instituto Superior Técnico, University of Lisbon
[⋆]Department of Computer Science, University of Copenhagen
ritaparadaramos@tecnico.ulisboa.pt

Abstract

Inspired by retrieval-augmented language generation and pretrained Vision and Language (V&L) encoders, we present a new approach to image captioning that generates sentences given the input image and a set of captions retrieved from a datastore, as opposed to the image alone. The encoder in our model jointly processes the image and retrieved captions using a pretrained V&L BERT, while the decoder attends to the multimodal encoder representations, benefiting from the extra textual evidence from the retrieved captions. Experimental results on the COCO dataset show that image captioning can be effectively formulated from this new perspective. Our model, named EXTRA, benefits from using captions retrieved from the training dataset, and it can also benefit from using an external dataset without the need for retraining. Ablation studies show that retrieving a sufficient number of captions (e.g., k=5) can improve captioning quality. Our work contributes towards using pretrained V&L encoders for generative tasks, instead of standard classification tasks.

1 Introduction

Image captioning is the task of automatically generating a short textual description for a given image. The standard approach involves the use of encoder-decoder neural models, combining a visual encoder with a language generation decoder (see Hossain et al. (2019) for a survey). In early studies, the encoder was typically a Convolutional Neural Network model (CNN) pretrained on the ImageNet classification dataset (Russakovsky et al., 2015) or a pretrained Faster-RCNN object detector (Ren et al., 2015), whereas the decoder was commonly an LSTM (Hochreiter and Schmidhuber, 1997) together with an attention mechanism (Bahdanau et al., 2014). More recently, Transformer based models have been achieving state-of-the-art results on a variety of language processing (Vaswani et al., 2017; Devlin et al., 2018; Radford et al., 2019) and computer vision tasks (Dosovitskiy et al., 2020). Accordingly, state-of-the art image captioning models have replaced the conventional CNN-LSTM approach with encoder-decoder Transformers (Liu et al., 2021). Still, in both cases, the encoder only attains visual representations, whereas richer features could be captured from image–text interactions if the encoder had access to useful textual context related to the input image (e.g., sentences associated to similar images).

In this paper, we present a new type of image captioning model that uses a pretrained V&L BERT (Tan and Bansal, 2019; Li et al., 2019; Bugliarello et al., 2020, *inter-alia*) to encode both the input image and captions retrieved from similar images. This model generates captions conditioned on representations that consider linguistic information beyond the image alone. Moreover, specifically using the retrieved captions as textual contexts rather than other alternatives (e.g., image tags or object names) can aid guiding the language generation process, since the model is now provided with well-formed sentences that are semantically similar to what the predicted caption should resemble.

In experiments on the COCO dataset (Chen et al., 2015), the proposed model is competitive against state of the art methods. In a series of ablation experiments, we find that the model improves when encoding multiple retrieved captions, and that it could reach better performance if it was able to retrieve better captions from the datastore. In experiments on the smaller Flickr30K dataset, we show that allowing the model to retrieve captions from the larger COCO dataset can improve performance without needing to retrain the model.

We hope that our work inspires the adoption of pretrained V&L encoders for a broader range of generative multimodal tasks. There have been several recent studies proposing V&L BERTs to learn generic multi-modal representations with

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3666–3681
May 2-6, 2023 ©2023 Association for Computational Linguistics

large amounts of paired image and text data, which can then be fine-tuned to downstream tasks. However, these pretrained models have mostly been applied to classification tasks and have seen limited use for image captioning, a task which typically only considers single-input images, as opposed to image-text pairs, as proposed in this work.

2 Model

We present a model that captions images, given both the image and a set of k captions retrieved from similar images using a retrieval system. This approach belongs to the class of retrieval-augmented language generation models (Weston et al., 2018; Izacard and Grave, 2020). In our model, the image and the retrieved captions are jointly encoded using a pretrained V&L encoder to capture cross-modal representations in the combined input data. We denote our model as EXTRA: **E**ncoder with **C**ross-modal representations **T**hrough **R**etrieval **A**ugmentation. It consists of three components, namely an encoder, a retrieval system, and a decoder.

2.1 Encoder

The encoder in EXTRA is LXMERT[1] (Tan and Bansal, 2019), a pretrained vision-and-language Transformer that jointly encodes a visual input $\mathbf{V}$ and a linguistic input $\mathbf{L}$. The visual input is represented as N=36 regions-of-interest V=$\{v_1, ..., v_N\}$ extracted from the image using the Faster-RCNN object detector, pretrained (Anderson et al., 2018) on the Visual Genome dataset (Krishna et al., 2016). A sentence in the linguistic input is tokenized into M sub-words using the BERT tokenizer (Devlin et al., 2018), starting with a special classification token `CLS` and ending with a special delimiter token `SEP`. We extended LXMERT to encode k sentences by concatenating the tokenized sentences into a single input, each separated by the delimiter token: L=$\{\texttt{CLS}, w_1^{L_1} \ldots w_M^{L_1}, \texttt{SEP} \ldots w_1^{L_k} \ldots w_M^{L_k}, \texttt{SEP}\}$. The sentences are obtained from a datastore via a retrieval system, as explained in Section 2.2.

The encoder produces a sequence of cross-modal representations of image and the text, which are the inputs to the decoder, described in Section 2.3.

2.2 Image–Text Retrieval and Datastore

The retrieval system builds on the Facebook AI Similarity Search (FAISS) nearest-neighbour search library (Johnson et al., 2017). FAISS allows for the indexing of high-dimensional vectors, i.e., a datastore $\mathbf{D}$, and it offers the ability to quickly search through the datastore given a similarity measure $\mathbf{S}$, e.g., Euclidean distance or cosine similarity.

Given an input image $\mathbf{V}$, the retrieval system finds L, the set of k captions retrieved from the datastore, which EXTRA encodes together with the image. The datastore consists of captions associated with images in a dataset[2]. Each caption in the datastore, and the query input image, are represented using vectors extracted from CLIP (Radford et al., 2021), allowing image–text search by projecting images and text to a shared latent space. Using FAISS, the input image can then be compared against the vectors[3] from $\mathbf{D}$ to search over the corresponding k nearest-neighbours captions.

2.3 Decoder

The decoder is a conditional auto-regressive language model based on GPT-2 (Radford et al., 2019) with additional cross-attention layers to the encoder. The Transformer layers in the decoder already contain a masked multi-head self-attention sublayer, which self-attends to the previous words. We add cross-attention layers (Vaswani et al., 2017) subsequent to the masked self-attention sublayers, so the decoder can attend to the encoder outputs.

The decoder predicts a caption $y_1 \ldots y_M$ token-by-token, conditioned on the previous tokens and the outputs of the V&L encoder. The model's parameters θ are trained by minimizing the sum of the negative log-likelihood of predicting the ground truth token at each time-step, using the standard cross-entropy loss:

$$L_\theta = -\sum_{i=1}^{M} \log P_\theta(y_i | y_{<i}, \mathbf{V}, \mathbf{L}). \qquad (1)$$

We can also fine-tune the model with Self-Critical Sequence Training (Rennie et al., 2017).

3 Experimental Protocol

3.1 Datasets and Metrics

We evaluate our model on the COCO dataset (Chen et al., 2015), using the standard Karpathy splits of

---

[1] The exploration of other encoders is left for future work.

[2] This can either be the training set or an external dataset.

[3] This comparison can be pre-computed for efficiency.

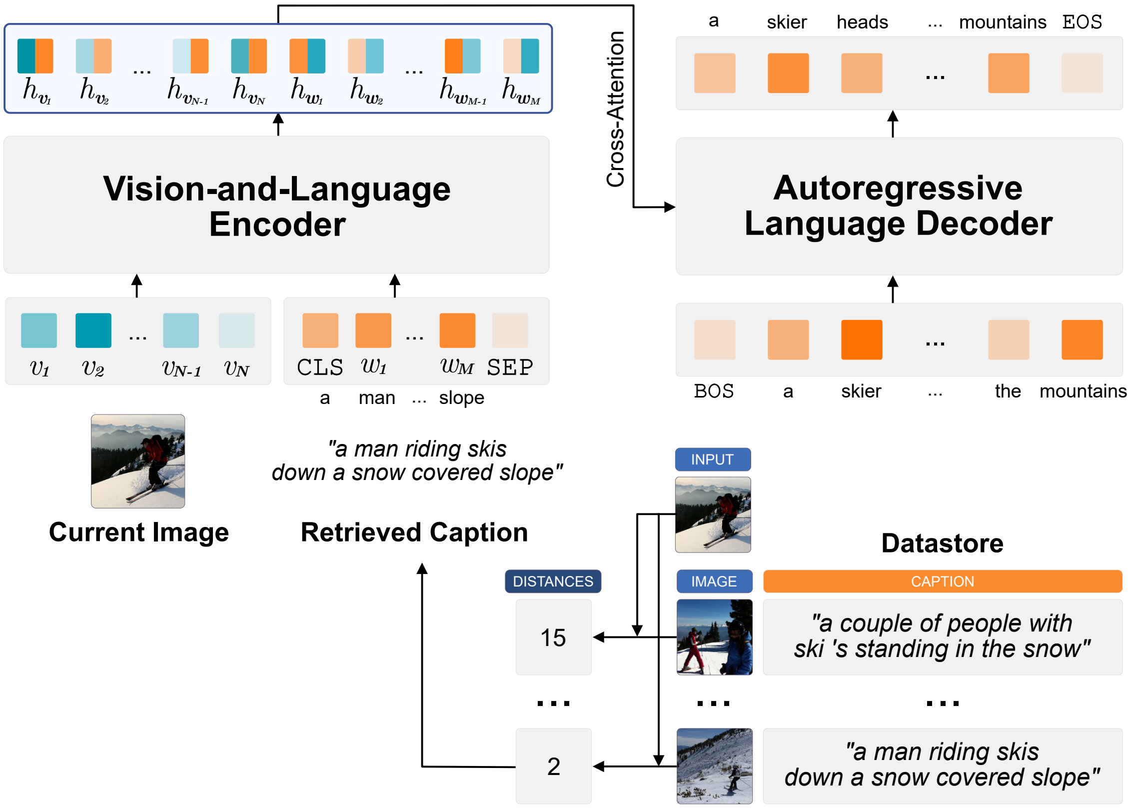


Figure 1: Illustration of the EXTRA model. Given an input image, EXTRA retrieves captions from a datastore and encodes both the input image and the retrieved captions using a pretrained vision-and-language encoder. The decoder attends over both the visual and linguistic outputs, improving the quality of the generated caption.

113287 images for training, 5000 for validation, and 5000 for testing, with 5 captions per image.

Standard metrics were used to evaluate caption generation, namely BLEU-4 (B4) (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014), CIDEr (Vedantam et al., 2015), and SPICE (Anderson et al., 2016), using the MS COCO caption evaluation package[4].

3.2 Implementation and Training Details

The implementation[5] of EXTRA uses the HuggingFace Transformers library (Wolf et al., 2020). The encoder is LXMERT (Tan and Bansal, 2019), a 14-layer V&L model pretrained on 9 million image–sentence pairs across a variety of datasets and tasks. Following Liu et al. (2021), the decoder is a 4-layer randomly initialized GPT-2-style Transformer network with 12 attention heads and additional cross-attention layers. The retrieval systems uses FAISS with a flat index (*IndexFlatIP*) without any training. The corresponding datastore **D** consists of all the captions associated to the 113287 images in the COCO training set. For caption retrieval, the captions in the datastore and the input image (i.e., the query) are both represented with features extracted from the CLIP-ResNet50×4 pretrained model. Using the cosine similarity for comparison, a total of $k = 5$ captions are retrieved to be jointly encoded with the input image by EXTRA. Notice that CLIP-ResNet50×4 features are only used for retrieval, while the EXTRA encoder, i.e. the pretrained LXMERT, requires Faster-RCNN features, and thus it cannot use CLIP visual features.

EXTRA is trained in two stages using a single NVIDIA V100S 32GB GPU. In the first stage, EXTRA is trained end-to-end with the cross-entropy loss, using a batch size of 64 and the AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of $3e-5$. The encoder is trained with a linear warmup for the first epoch to prevent gradients from the randomly initialized decoder from harming the pretrained encoder. The model was trained with early stopping: training ends if there is no improvement after 5 consecutive epochs on the val-

---

[4] https://github.com/tylin/coco-caption

[5] https://github.com/RitaRamo/extra

	Cross-Entropy Optimization				CIDEr Optimization			
	B4	METEOR	CIDEr	SPICE	B4	METEOR	CIDEr	SPICE
Encoder-Decoder models								
Up-Down	36.2	27.0	113.5	20.3	36.3	27.7	120.1	21.4
CaMEL$_{\text{Faster R-CNN}}$	36.1	28.0	114.8	20.8	-	-	-	-
GCN-LSTM	36.8	27.9	116.3	20.9	38.2	28.5	127.6	22.0
VL-T5	34.5	28.7	116.5	21.9	-	-	-	-
AoANet	37.2	28.4	119.8	21.3	38.9	29.2	129.8	22.4
CPTR	-	-	-	-	40.0	29.1	129.4	-
EXTRA ($k = 5$)	37.5	28.5	120.9	21.7	36.4	28.2	131.1	21.3
CaMEL$_{\text{CLIP-RN50}\times 16}$	38.8	29.4	125.0	22.2	41.3	30.2	140.6	23.9
V&L BERT models								
VLP	36.5	28.4	116.9	21.2	39.5	29.3	129.3	23.2
OSCAR$_B$	36.5	30.3	123.7	23.1	40.5	29.7	137.6	22.8
VinVL$_B$	38.2	30.3	129.3	23.6	40.9	30.9	140.4	25.1

Table 1: Results on the Karpathy COCO test split. **EXTRA** ($k = 5$) is competitive against encoder-decoder models. We present results with cross-entropy training and after Self-Critical Sequence Training using the CIDEr metric.

idation set over the BLEU-4 metric. In the second stage, EXTRA is fine-tuned with Self-Critical Sequence Training (Rennie et al., 2017) with CIDEr optimization and greedy search decoding as a baseline, using a batch size of 55, a learning rate of 3e-5, and a frozen encoder. Captions are decoded using beam search with a beam size of 3.

4 Results

Table 1 shows the performance of EXTRA compared to strong encoder-decoder models. We compare against the widely-used Up-Down (Anderson et al., 2018) and AoANet models (Huang et al., 2019), both using a Faster-RCNN image encoder; the GCN-LSTM model (Yao et al., 2018) with a Graph Convolutional Network (GCN) encoder; the CPTR model (Liu et al., 2021) employing a ViT Transformer encoder (Dosovitskiy et al., 2020); the VL-T5 Transformer model (Cho et al., 2021) with a vision and language encoder; and the recent CaMEL model (Barraco et al., 2022) with the CLIP-RN50×16 encoder. Our model is also compared with state-of-art models that do not use the encoder-decoder paradigm but instead unify the Transformer encoder and decoder into a single model, namely the VLP (Zhou et al., 2020), OSCAR-base (Li et al., 2020), and the VinVL-base (Zhang et al., 2021) models. We note that these are general purpose V&L models, not specifically designed for image captioning.

Overall, EXTRA is competitive to state-of-the art captioning models. It outperforms captioning models with vision encoders, and VL-T5, which, like EXTRA, uses a V&L encoder, but with object tags as linguistic inputs rather than retrieved captions. Although EXTRA does not outperform the state of the art captioning model, CaMEL, that uses a dual decoder, it outperforms the variant of CaMEL that uses the same Faster-RCNN features. EXTRA also competes with general purpose V&L BERT models. Notice that our approach can be adapted to other V&L encoders besides LXMERT (e.g., OSCAR, VinVL, etc.), or to more powerful decoders (e.g., as in CaMEL). Likewise, other models could benefit from retrieval-augmentation with captions.

4.1 Ablation Studies

We conducted a series of ablation studies in the Karpathy COCO validation split to better understand what contributes to the success of EXTRA.

Varying the Number of Retrieved Captions: We start by studying the importance of training with multiple retrieved captions, training with k=1 and k=3 captions to explore the effect of retrieving fewer captions. Table 2 reports the result of this experiment, showing that performance degrades when retrieving less captions.

Encoding Irrelevant Captions: We also studied the performance of EXTRA when it encodes textual

	B4	CIDEr
$k = 1$	36.7	118.0
$k = 3$	37.4	119.1
$k = 5$	38.3	121.2

Table 2: The effect of training and evaluating using different numbers of retrieved captions. Performance reported after training with cross-entropy optimization.

input that is not expected to be useful. We conduct two experiments where EXTRA is trained with textual input that is either an empty caption or a randomly chosen caption.

- **Empty Caption**: encode the image with an empty sentence: L={`CLS`, `SEP`};
- **Random Caption**: encode the image with a random caption from the datastore.

Table 3 shows the result of this experiment. EXTRA outperforms both variants, further showing that the generation process is improved by encoding the image together with relevant textual context from nearest-neighbour captions. Although having an inferior performance, both models reach reasonable results compared to other models in the literature (see Table 1), showing that LXMERT can be used as a strong encoder for image captioning without providing relevant input image-text pairs.

	B4	CIDEr
Empty caption	37.8	119.1
Random caption	37.1	117.7
EXTRA	38.3	121.2

Table 3: The effect of training and evaluating with captions that are not expected to be useful.

Encoding Irrelevant Images: We tested ablating the visual input (i.e., setting the visual features to zero). Training on "blacked out" input images achieves 102.1 in CIDEr, which is substantially lower than training with the actual input images, as seen in Table 4. This further shows that EXTRA uses the visual input, and does not just rely on the retrieved information.

Changing the Retrieval System and Datastore: We then studied the effect of changing the retrieval system and the representations in the datastore. Recall that EXTRA relies on captions obtained by Image–Text retrieval, where the datastore contains the captions from the COCO training set, represented as vectors extracted from CLIP. We conducted experiments with Image–Image and Image–Text retrieval to understand which performs better:

	B4	CIDEr
Blacked out image	32.1	102.1
EXTRA	38.3	121.2

Table 4: The effect of training and evaluating with "blacked out" input images.

- **Image–Image Retrieval**: the datastore consists of all the images in the training data. The representation of the input image is compared against those in the datastore to find the k nearest-neighbour images, and, subsequently, to obtain the k captions associated to those images. Specifically, one reference caption is retrieved from each of the top-k nearest-neighbour images.
- **Image–Text Retrieval**: the datastore consists of all the captions associated to the images in the training data. The representation of the input image is compared against the captions to directly find the top-k captions.

For Image–Image retrieval, the input image and the images in **D** are represented with *Faster R-CNN* features, after global average pooling the embeddings of the 36 region-of-interest vectors. For Image–Text Retrieval, the input image and the caption vectors should already belong to a shared semantic space. We use the pretrained CLIP model because it satisfies this criteria and thus allows for direct image–text comparison. We considered two variants of CLIP based on their visual backbone: *ViT* or *ResNet50x4*[6]. The results of this experiment are reported in Table 5.

EXTRA performs worse when it uses Image–Image retrieval in comparison to retrieving captions directly with Image–Text retrieval. The best performance is obtained with the ResNet-variant of the CLIP encoder. We also assess the performance of directly using *only* one of the retrieved captions, with the results shown in Figure 2. In this figure, we can visualize the expected CIDEr score

---

[6] Regarding the comparison measure **S**, the Euclidean distance and cosine similarity were used respectively for Image-to-Image retrieval and Image-to-Text retrieval.

	B4	CIDEr
Image–Image (Faster R-CNN)	36.8	117.1
Image–Text (CLIP ViT)	38.1	120.3
Image–Text (CLIP ResNet)	38.3	121.2

Table 5: The effect of training and evaluating EXTRA with different retrieval systems. $k = 5$ in both settings.

of the first retrieved captions and observe that some of them do not sufficiently describe the image, or are mismatches, with a CIDEr of zero. We also observe that the CIDEr score can change significantly depending on the retrieval system. A larger number of mismatch captions are retrieved with Image-to-Image retrieval. This suggests that the retrieval system and the datastore can largely impact a retrieval-augmented image captioning model, hence they should be carefully considered.

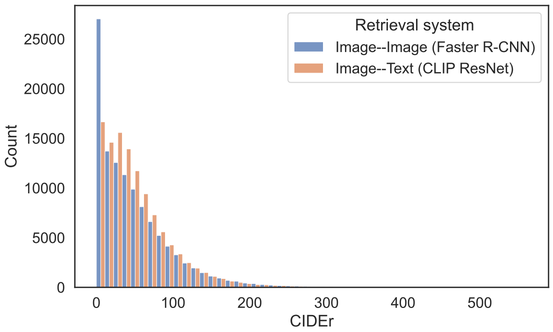


Figure 2: Histogram of the CIDEr scores for the nearest caption ($k = 1$) retrieved with Image-Image and Image-Text retrieval. This shows the evaluations scores of using only the retrieved captions.

Oracle Performance: Given that the retrieval system and datastore affect the performance of EXTRA, we also study whether EXTRA could continue to improve if it could retrieve *better* captions. After training EXTRA with the $k = 5$ retrieved captions, we simulate an oracle retrieval system during inference, by allowing the actual reference captions to be encoded by EXTRA. Table 6 reports on experiments in the validation data with respect to replacing one of the k retrieved captions with one of the reference captions, as well as replacing all with the 5 references associated to the input images. These experiments bring a 1.8 and 8.3 point increase in CIDEr score, respectively, showing the potential for EXTRA to improve by retrieving captions that better match the input image.

	B4	CIDEr
$k = 5$ retrieved captions	38.3	121.2
$k = 4$ and 1 reference	39.0	123.0
$k = 0$ and 5 references	40.9	129.5

Table 6: Simulation of an oracle experiment, where EXTRA can "retrieve" reference captions of an image instead of retrieving all 5 captions from the datastore.

5 Discussion

5.1 Vision First and Language Later

How does EXTRA use the encoded image and retrieved captions? We quantify this by estimating the behaviour of the cross-modal attention heads at each layer in the decoder. Specifically, we compute the average of the cross-modal attention across either the number of image regions or the sub-words in the encoder, at each time-step of generating a caption and across each of the 12 attention heads.

Figure 3 shows that across the layers, the decoder's attention shifts to the textual outputs. In Layer 1, the model attends both to the visual and textual representations, but the model hardly pays attention to the visual outputs by Layer 4, relying more on the textual information from the retrieved captions. This behaviour further shows that the semantics of the nearest captions can aid guiding the language generation process. We performed an identical calculation for the variants of EXTRA that encoded an empty or a random caption, finding in this case the opposite behaviour: the model learned to ignore the textual embeddings provided by the encoder (see Appendix A).

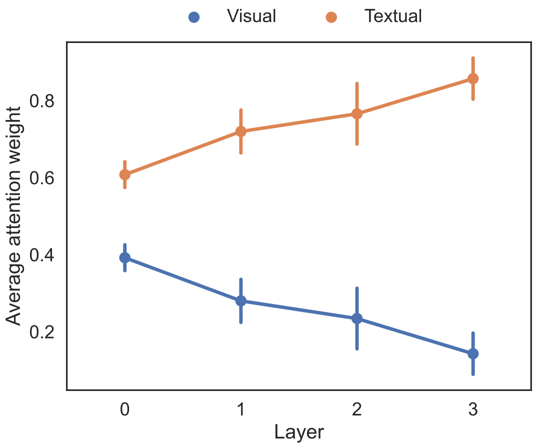


Figure 3: The average cross-attention from the decoder to the outputs of the encoder in respect to the visual V and textual L outputs. Values from COCO validation.

5.2 Retrieve Enough Captions to Overcome Retrieval Mistakes

We note that training with an empty set of captions was better than encoding a single $k = 1$ and $k = 3$ retrieved captions, observing Tables 2 and Table 3. Thus, retrieval augmentation aids to improve caption quality when a sufficient number ($k = 5$) is considered. This further shows that retrieving enough captions can be crucial for success. For this, we hypothesise that retrieving more captions makes the model more robust in the presence of mismatches from certain captions, as shown for instance in the second example in Figure 4.

5.3 Hot-swapping the Datastore

Besides taking advantage of similar training examples, we study whether EXTRA works with external image–caption collections without needing to retrain the model. For this experiment, EXTRA was first trained and evaluated in a small dataset, and then the retrieval datastore was augmented with a larger dataset. The considered datasets were Flickr30k and COCO, respectively. While Flickr30k only contains 30k images, COCO contains 113K, each paired with five sentences. Table 7 reports the results of these experiment. EXTRA got a better performance considering a larger external dataset than just using the current training set, showing the potential for EXTRA to adapt the retrieval datastore.

Retrieval Datastore	B4	CIDEr
Flickr30k	28.8	59.6
+ COCO	29.5	59.9

Table 7: Performance of EXTRA on the Flickr30k validation set. The model is trained on the Flickr30K dataset with the Flickr30K datastore. The datastore for inference is either the Flickr30K training set or combined with the COCO training set.

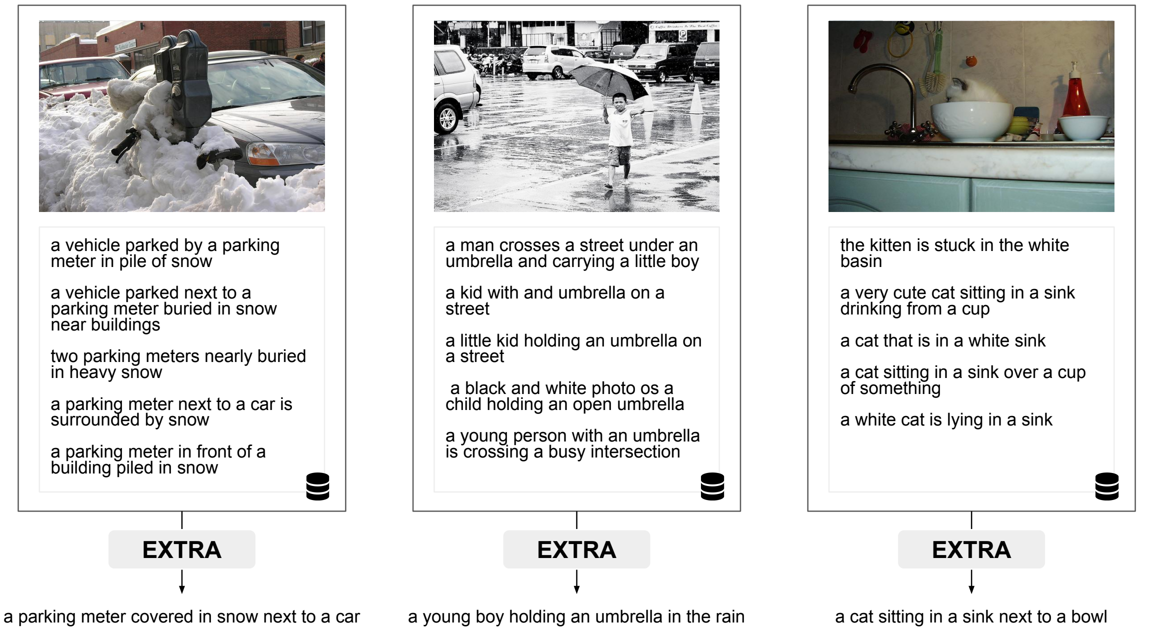
5.4 Qualitative Examples

Figure 4 shows examples of captions generated by EXTRA, given the input image and the $k = 5$ retrieved captions. EXTRA benefits from textual evidence from nearest-neighbour captions, even though sometimes the retrieved information can be misleading, as depicted in the last example. More examples are provided in Appendix B.

6 Related Work

Image Captioning: The task of image captioning is usually addressed by one of these three main approaches: templates, retrieval, and encoder-decoder methods. Early approaches involved template-based methods that consisted of filling blanks of predefined captions through object detection (Farhadi et al., 2010; Kulkarni et al., 2013; Elliott and de Vries, 2015). Retrieval-based methods instead search over a dataset for the most similar image and fetch the corresponding caption (Hodosh et al., 2013; Ordonez et al., 2011). Currently, the most common approach is the encoder-decoder framework (Xu et al., 2015; Hossain et al., 2019). The encoder typically used a pretrained CNN (Vinyals et al., 2016) or a Faster R-CNN (Anderson et al., 2018), encoding the image into a grid of image features or object proposal image regions. The decoder was usually a LSTM with an attention mechanism (Xu et al., 2015) to dynamically focus on different parts of the encoded image during the prediction of each word.

Recently, Transformer-based models like BERT (Devlin et al., 2018) have become a more popular choice than LSTMs models, outperforming recurrent architectures in different natural language processing (NLP) tasks (Vaswani et al., 2017; Qiu et al., 2020). Transformers can capture long-range dependencies with self-attention layers and they can process each word of a sentence in parallel, reducing training time. After the successful application in NLP, vision Transformers like ViT (Dosovitskiy et al., 2020) are also starting to become the model of choice in the field of computer vision in place of CNNs. In similar fashion, most recent captioning studies use the Transformer arquitecture (Herdade et al., 2019; Cornia et al., 2020; Liu et al., 2021), employing a vision Transformer as encoder together with an autoregressive language Transformer as decoder. Similarly to these models, this work proposes a encoder-decoder Transformer model for the task of image captioning. However, unlike them, the proposed model incorporates a pretrained V&L BERT to exploit cross-modal representations, encoding images along with textual context. Also differently from previous work, this approach explores retrieval-augmented generation, i.e., combining neural encoder-decoder methods with traditional retrieval-based methods.



a vehicle parked by a parking meter in pile of snow

a vehicle parked next to a parking meter buried in snow near buildings

two parking meters nearly buried in heavy snow

a parking meter next to a car is surrounded by snow

a parking meter in front of a building piled in snow

a parking meter covered in snow next to a car

a man crosses a street under an umbrella and carrying a little boy

a kid with and umbrella on a street

a little kid holding an umbrella on a street

a black and white photo os a child holding an open umbrella

a young person with an umbrella is crossing a busy intersection

a young boy holding an umbrella in the rain

the kitten is stuck in the white basin

a very cute cat sitting in a sink drinking from a cup

a cat that is in a white sink

a cat sitting in a sink over a cup of something

a white cat is lying in a sink

a cat sitting in a sink next to a bowl

Figure 4: Examples of captions generated by EXTRA conditioned on the input image and retrieved captions.

V&L BERTs: Previous studies have proposed pretrained Vision and Language (V&L) BERTs to learn generic cross-modal representations of images and text, that can later be used for a variety of downstream V&L tasks (Bugliarello et al., 2020). Examples include LXMERT (Tan and Bansal, 2019), VL-BERT (Su et al., 2019), Visual BERT (Li et al., 2019), OSCAR (Li et al., 2020), or UNITER (Chen et al., 2020), which were applied to VQA and other V&L classification tasks. Given that these models are encoder-only Transformers, only few of them have been applied to generation tasks such as image captioning. In such cases, the generation is made from left to right by encoding the input image and using the textual input elements with uni-directional attention masks, i.e., starting with a `CLS` token with the rest of the tokens masked, then considering the `CLS` token with the predicted word (replaced by the corresponding mask token) and the remaining ones still masked, and so on (Li et al., 2020; Zhou et al., 2020).

The use of pretrained V&L BERTs, as encoders in the standard encoder-decoder captioning framework, remains largely unexplored. The task of image captioning typically just considers single-input images, and not image-text pairs to be encoded. In our work, a pretrained V&L encoder is used with a decoder for image captioning, by leveraging not just the images as input but also retrieved captions.

Besides pretrained V&L encoders, pretrained V&L encoder-decoder models have recently been proposed to tackle classification and generation tasks, such as VL-T5 (Cho et al., 2021). Their captioning approach is similar to the present paper, but VL-T5 uses object tags as textual inputs, whereas EXTRA is conditioned on retrieved captions.

Retrieval-augmented Generation: The proposed approach is also similar to some studies on language generation that predict the output conditioned on retrieved examples (Weston et al., 2018; Gu et al., 2018; Khandelwal et al., 2019; Lewis et al., 2020). For instance, this work relates to Weston et al. (2018), in which a sequence-to-sequence LSTM model, for dialog generation, encodes the current input concatenated with the nearest retrieved response. Similarly, Izacard and Grave (2020) used an encoder-decoder Transformer conditioned on retrieved passages for open domain question answering. Retrieval-augmented generation is gaining traction in NLP but has only been explored for image captioning by few studies (Wang et al., 2020; Fei, 2021; Ramos et al., 2021; Sarto et al., 2022; Ramos et al., 2022). Concurrent work proposed Transformer-based captioning models augmented with retrieval as well (Sarto et al., 2022; Ramos et al., 2022). However, differently from these previous studies, we encode the retrieved captions by exploiting cross-modal representations with a V&L encoder.

7 Conclusions

We propose EXTRA, a retrieval-augmented image captioning model that improves performance by exploiting cross-modal representations of the input image together with captions retrieved from a datastore. EXTRA make uses of a pretrained V&L BERT, instead of an image-only encoder, combined with a language decoder. To generate a caption, the decoder attends to the cross-modal encoder features, containing information from image regions and also textual evidence from the retrieved captions. Image captioning is therefore addressed as language generation conditioned on vision and language inputs, instead of vision only. To evaluate this model, EXTRA was assessed against strong encoder-decoder models in the area, and ablation studies were also conducted. The experiments conducted on the COCO dataset confirmed the effectiveness of the proposed captioning approach.

For future work, we plan to explore the utility of EXTRA in out-of-domain and in few-shot learning settings, since the retrieval component can be easily modified to include external datastores, without the need to retrain the whole model. We also plan to explore how this approach can be adapted to other powerful vision and language encoders besides LXMERT. Finally, we will explore methods that allow us to jointly train the retrieval mechanism with the full model in order to retrieve captions that are more similar to the input image.

Limitations

Previous work has shown that generative models suffer from biases inherent to the data they are trained on (Weidinger et al., 2021; Thoppilan et al., 2022). Likewise, our EXTRA model can suffer from biases present in the COCO image captioning dataset (Chen et al., 2015). Particularly, it has been shown that there is significant gender imbalance in COCO, and that captioning models can exhibit gender bias amplification (e.g., they are likely to generate the word "woman" in kitchen scenarios, and the word "man" in snowboarding scenes) (Hendricks et al., 2018; Zhao et al., 2017).

However, differently from most captioning models, EXTRA is a retrieval-augmented captioning model, and thus it has the potential to make predictions beyond the training data, by relying on information from an external datastore. Still, the datastore knowledge might also have inherent bias, as mentioned by previous studies on retrieval-augmented generation (Lewis et al., 2020). In the paper, we show examples of such limitations wherein mismatched retrieved captions can bias the model towards incorrect predictions (see the results and appendix sections).

As a way to mitigate these limitations, we recommend analyzing the corresponding nearest captions when using EXTRA, since the retrieved captions can give useful insight of the bias involved in the generation process. EXTRA can provide interpretability through textual descriptions, whereas most captioning models only provide explanations as visual attention maps.

EXTRA also has the downside of focusing on an English-centric dataset. Captioning datasets are primarily available in English, and most image captioning models are trained on COCO or other english-centric datasets. To avoid hindered research on image captioning, it is important to consider multilingual captioning datasets that contain both language-diverse captions and geographically-diverse visual concepts (Thapliyal et al., 2022).

Acknowledgements

This research was supported by the Portuguese Recovery and Resilience Plan (RRP) through project C645008882-00000055 (Responsible.AI), and also through Fundação para a Ciência e Tecnologia (FCT), namely through the Ph.D. scholarship with reference 2020.06106.BD, as well as through the INESC-ID multi-annual funding from the PIDDAC programme with reference UIDB/50021/2020.

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *Proceedings of the European Conference on Computer Vision*, pages 382–398. Springer.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6077–6086.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Manuele Barraco, Matteo Stefanini, Marcella Cornia, Silvia Cascianelli, Lorenzo Baraldi, and Rita Cucchiara. 2022. Camel: Mean teacher learning for image captioning. *arXiv preprint arXiv:2202.10492*.

Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. 2020. Multimodal pretraining unmasked: Unifying the vision and language berts. *arXiv preprint arXiv:2011.15124*.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *Proceedings of the European Conference on Computer Vision*, pages 104–120.

Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. *arXiv preprint arXiv:2102.02779*.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10578–10587.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 376–380.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Desmond Elliott and Arjen de Vries. 2015. Describing images using inferred visual dependency representations. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 42–52.

Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: Generating sentences from images. In *Proceedings of the European Conference on Computer Vision*, pages 15–29. Springer.

Zhengcong Fei. 2021. Memory-augmented image captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2–9.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. 2018. Search engine guided neural machine translation. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence*, pages 5133–5140.

Lisa Anne Hendricks, Kaylee Burns, Kate Saenko, Trevor Darrell, and Anna Rohrbach. 2018. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision*, pages 771–787.

Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. 2019. Image captioning: Transforming objects into words. *arXiv preprint arXiv:1906.05963*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Micah Hodosh, Peter Young, and Julia Hockenmaier. 2013. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899.

MD Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. 2019. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys*, 51(6):1–36.

Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. 2019. Attention on attention for image captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4634–4643.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *arXiv preprint arXiv:1602.07332*.

Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2013. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2891–2903.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Proceedings of the Conference on Neural Information Processing Systems*, 33:9459–9474.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. OSCAR: Object-semantics aligned pre-training for vision-language tasks. In *Proceedings of the European Conference on Computer Vision*, pages 121–137.

Wei Liu, Sihan Chen, Longteng Guo, Xinxin Zhu, and Jing Liu. 2021. Cptr: Full transformer network for image captioning. *arXiv preprint arXiv:2101.10804*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Vicente Ordonez, Girish Kulkarni, and Tamara Berg. 2011. Im2text: Describing images using 1 million captioned photographs. *Proceedings of the Conference on Neural Information Processing Systems*, 24:1143–1151.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*, pages 8748–8763.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rita Ramos, Bruno Martins, Desmond Elliott, and Yova Kementchedjhieva. 2022. Small-Cap: Lightweight image captioning prompted with retrieval augmentation. *arXiv preprint arXiv:2209.15323*.

Rita Parada Ramos, Patrícia Pereira, Helena Moniz, Joao Paulo Carvalho, and Bruno Martins. 2021. Retrieval augmentation for deep neural networks. *Proceedings of the International Joint Conference on Neural Networks*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Proceedings of the Conference on Neural Information Processing Systems*, 28:91–99.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7008–7024.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Sara Sarto, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. 2022. Retrieval-augmented transformer for image captioning. *arXiv preprint arXiv:2207.13162*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Ashish V Thapliyal, Jordi Pont-Tuset, Xi Chen, and Radu Soricut. 2022. Crossmodal-3600: A massively multilingual multimodal evaluation dataset. *arXiv preprint arXiv:2205.12522*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2016. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):652–663.

Li Wang, Zechen Bai, Yonghua Zhang, and Hongtao Lu. 2020. Show, recall, and tell: Image captioning with recall mechanism. *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence*, 34(07):12176–12183.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Jason Weston, Emily Dinan, and Alexander H Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. *arXiv preprint arXiv:1808.04776*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP: System Demonstrations*, pages 38–45.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the International conference on Machine Learning*, pages 2048–2057.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2018. Exploring visual relationship for image captioning. In *Proceedings of the European Conference on Computer Vision*, pages 684–699.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. VinVL: Making visual representations matter in vision-language models. *Proceedings of the Computer Vision and Pattern Recognition Conference*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. *arXiv preprint arXiv:1707.09457*.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence*, volume 34, pages 13041–13049.

A Cross-Attention

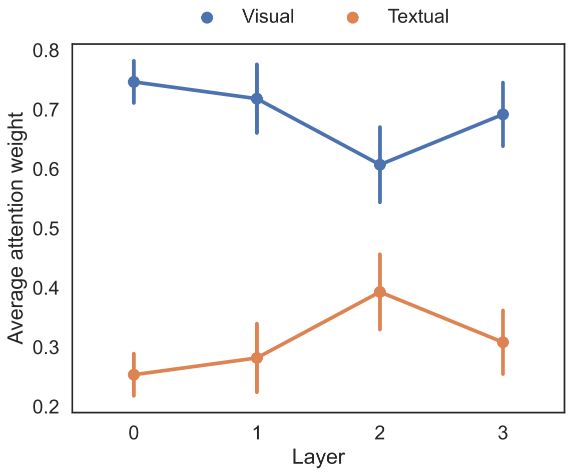


Figure 5: Cross-attention for the variant of EXTRA that that encodes an empty caption.

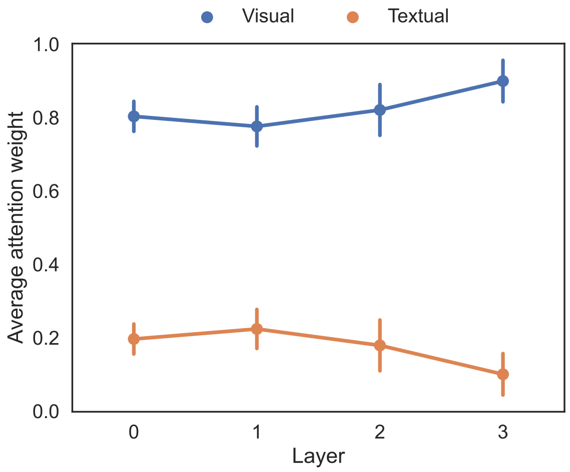


Figure 6: Cross-attention for the variant of EXTRA that that encodes a random caption.

In Section 5.1, we quantified how much attention EXTRA pays to the encoded image and retrieved captions. We also quantify this for the two other variants of EXTRA which encode irrelevant captions, using either an empty or a random caption. Figures 5 and 6 show the average cross-attention weights from the decoder to the outputs of the encoder in respect to the visual **V** and textual **L** outputs, respectively for the empty and random caption encoding. Contrary to the findings presented in Section 5.1, regarding the encoding of retrieved captions, in this scenario the two variants pay more attention to the visual outputs instead.

For details on how we calculated the corresponding attention weights, we present the corresponding formula. Specifically, we calculated the average of the cross-modal attention C across either the number of image regions or the sub-words in the encoder at each of T time-step of generating a caption and across each of the $H = 12$ attention heads.

This calculation happens independently for each of the $L = 4$ layers in the decoder:

$$A(C^L, V) = \frac{1}{H} \sum_{j=1}^{H} \frac{1}{T} \sum_{t=1}^{T} \sum_{i=1}^{|V|} \alpha^L_{j,t \to i}. \tag{2}$$

$$A(C^L, T) = 1 - A(C^L, V). \tag{3}$$

B More Examples

Figure 7 shows additional examples of the captions generated by EXTRA considering the retrieved captions, against the other two variants: encoding an empty and random caption instead. For the first image, the two variants fail to recognize that the image shows kids playing basketball (perhaps given the small size of the ball), whereas EXTRA was able to identify it by having that information in the retrieved captions. In the second image[7], the two variants produced the error of generating *sandwich* while EXTRA correctly mentioned *hot-dog*, similar to the retrieved captions. EXTRA considers the semantics from the nearest captions retrieved during generation, sometimes even copying an entire sentence, as shown in Figure 9.

Figure 8 shows examples where the retrieved captions mislead the model. We note however that EXTRA is also able to succeed, despite the mismatch from retrieved captions, as seen in Figure 10.

---

[7] The person was blurred for privacy concerns.

Figure 7: Examples of generated captions by EXTRA and the other two variants (empty and random caption). Better image captions are obtained from generating with retrieval augmentation.

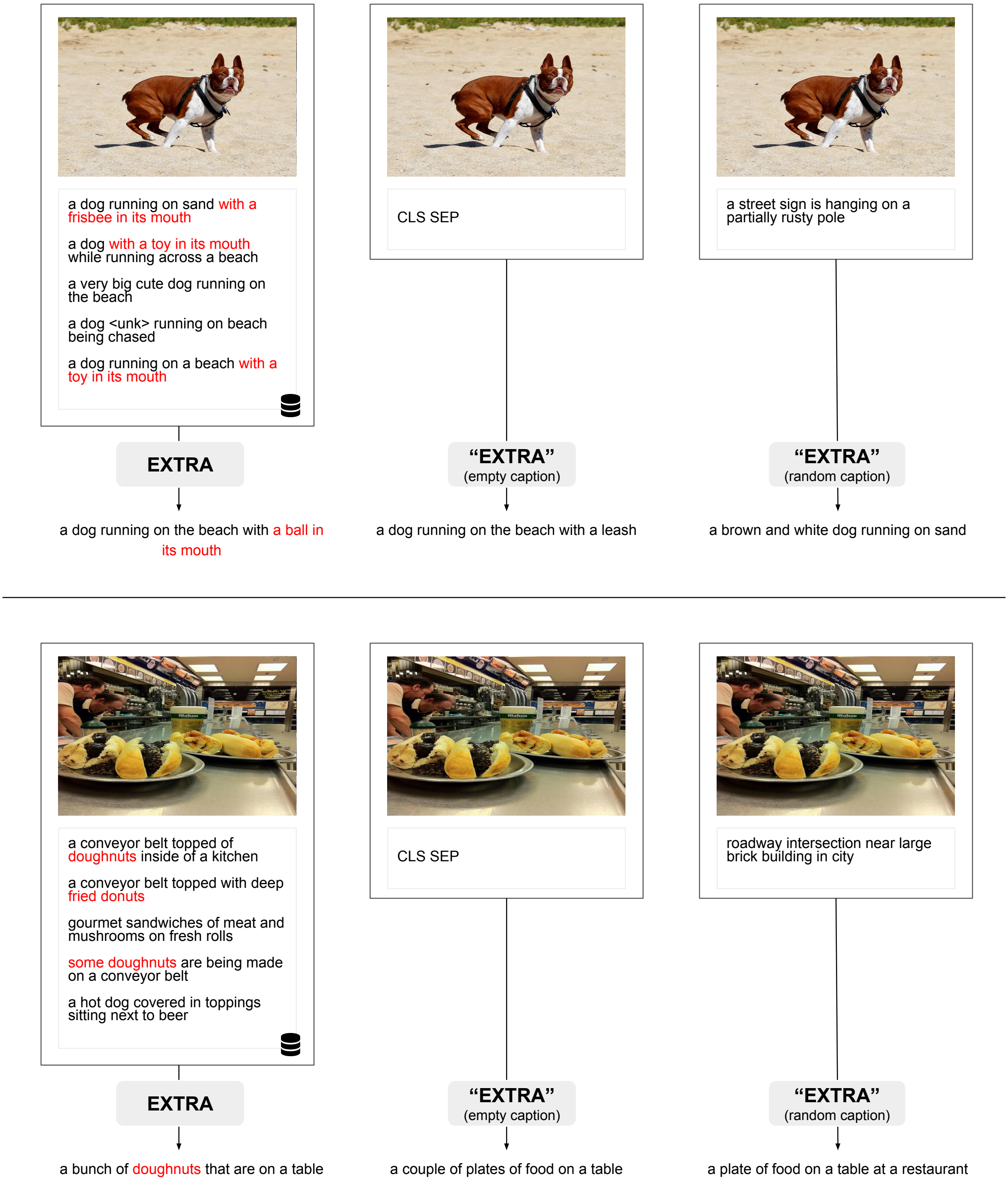
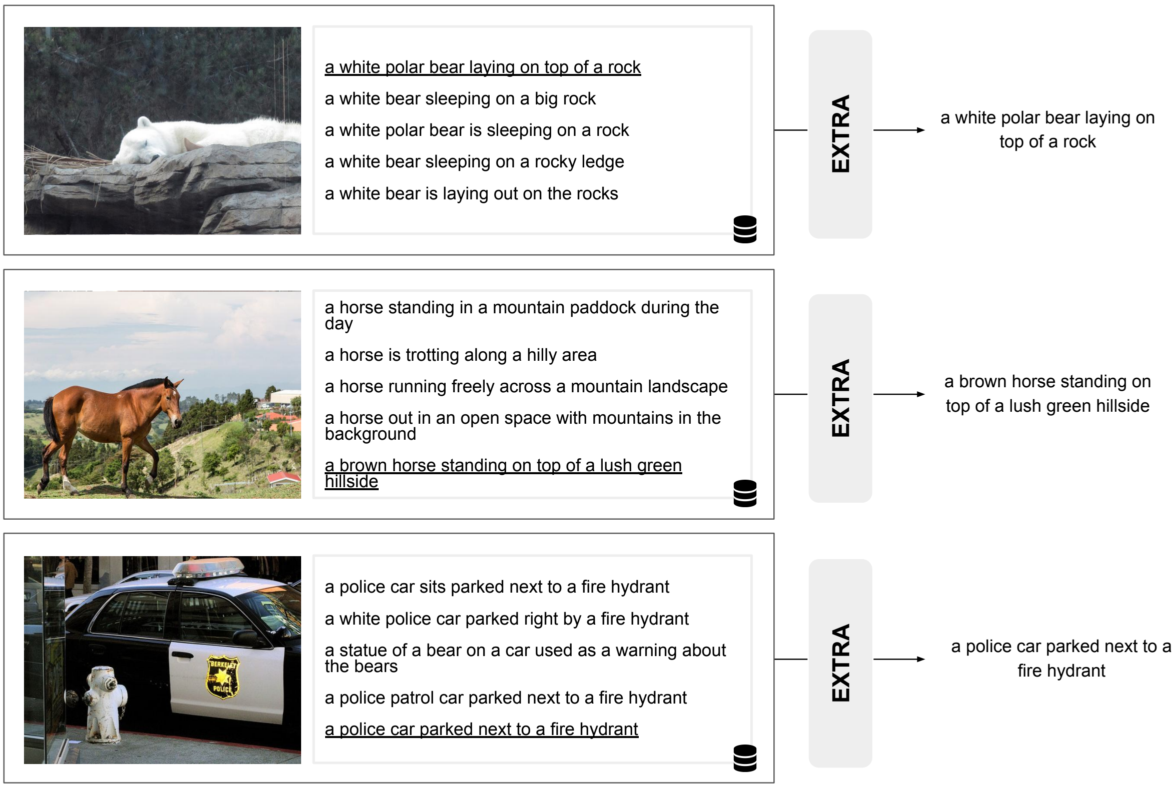


Figure 8: Qualitative results in which the retrieved captions are not that related to the input image.


a white polar bear laying on top of a rock
a white bear sleeping on a big rock
a white polar bear is sleeping on a rock
a white bear sleeping on a rocky ledge
a white bear is laying out on the rocks

a white polar bear laying on top of a rock

a horse standing in a mountain paddock during the day
a horse is trotting along a hilly area
a horse running freely across a mountain landscape
a horse out in an open space with mountains in the background
a brown horse standing on top of a lush green hillside

a brown horse standing on top of a lush green hillside

a police car sits parked next to a fire hydrant
a white police car parked right by a fire hydrant
a statue of a bear on a car used as a warning about the bears
a police patrol car parked next to a fire hydrant
a police car parked next to a fire hydrant

a police car parked next to a fire hydrant

Figure 9: Examples of generated captions for which EXTRA copied from the retrieved captions.


a young boy putting the telephone up to a toy 's ear
a young boy holds a doughnut to his face
a young boy holding a sandwich up to his face
a young man on a chair biting into a sandwich
a boy leans over a kitchen table while eating

a boy sitting on a chair holding a teddy bear

black dog laying down near a black and white cat
a gray and black cat sleeping while laying down
a couple of black cats laying down on a bed
a black cat sleeps on an old couch
a large furry black and brown cat sleeping on a chair

a black cat laying on top of a bed

a group of people eat cake in an office
men at an office appear confused by a presentation
man in an office setting eating a large hotdog
a man in an office attempts to eat a hotdog in one bite
an office worker taking a selfie with his iphone

a man sitting in front of a laptop computer

Figure 10: Examples where EXTRA is able to succeeded even with mismatches from retrieved captions.

Automatic Evaluation and Analysis of Idioms in Neural Machine Translation

Christos Baziotis[*]
University of Edinburgh
c.baziotis@ed.ac.uk

Prashant Mathur
Amazon AI
pramathu@amazon.com

Eva Hasler
Amazon AI
ehasler@amazon.com

Abstract

A major open problem in neural machine translation (NMT) is the translation of idiomatic expressions, such as "*under the weather*". The meaning of these expressions is not composed by the meaning of their constituent words, and NMT models tend to translate them literally (i.e., word-by-word), which leads to confusing and nonsensical translations. Research on idioms in NMT is limited and obstructed by the absence of automatic methods for quantifying these errors. In this work, first, we propose a novel metric for *automatically* measuring the frequency of literal translation errors without human involvement. Equipped with this metric, we present *controlled* translation experiments with models trained in different conditions (with/without the test-set idioms) and across a wide range of (global and targeted) metrics and test sets. We explore the role of monolingual pretraining and find that it yields substantial *targeted* improvements, even without observing any translation examples of the test-set idioms. In our analysis, we probe the role of idiom context. We find that the randomly initialized models are more local or "myopic" as they are relatively unaffected by variations of the idiom context, unlike the pretrained ones.

1 Introduction

Neural machine translation (NMT; Sutskever et al. 2014; Bahdanau et al. 2015; Vaswani et al. 2017) struggles with the translation of rare multi-word expressions (MWE) (Koehn and Knowles, 2017). Non-compositional phrases, such as idioms (e.g., "piece of cake"), are one of the most challenging types of MWEs, because their meaning is figurative and cannot be derived from the meaning of their constituents (Nunberg et al., 1994; Liu, 2017). NMT models tend to translate these expressions literally (i.e., word-by-word), which leads to erroneous translations. In this paper, our focus is on the translation of idiomatic expressions, in contrast to most prior work, which is subsumed under MWEs in general (Constant et al., 2017; Cook et al., 2021).

The absence of *targeted* and *automatic* evaluation is a major obstacle to advances in idiom translation. Global metrics, such as BLEU (Papineni et al., 2002) consider the full translation, and thus, the effects of idiom translation are overshadowed. Previous efforts on targeted evaluation isolate the idiom translation using word alignments (Fadaee et al., 2018) or word edit distance (Zaninello and Birch, 2020). These approaches measure the accuracy of idiom translation but do not account for literal translation errors. Shao et al. (2018) proposed a method for estimating the frequency of such errors, but it requires the creation of language-specific handcrafted lists (i.e., blocklists) with words that correspond to literal translation errors.

In this work[1], we present a study of idioms in NMT, with the goal of facilitating future research in this direction. First, we propose a novel metric for the *automatic* evaluation of literal translation errors (LitTER), that does not require any hand-crafted blocklists. We incorporate LitTER, which complements alignment-based metrics (Fadaee et al., 2018) into a unified targeted evaluation framework.

Next, we present translation experiments in a controlled setting, by using different training splits to test models under different conditions (e.g., zero-shot). To improve idiom translation we leverage monolingual data, which are more abundant than parallel and contain idioms in higher frequencies and more diverse contexts. We exploit monolingual data via pretraining (mBART; Liu et al. 2020), which is a generic and task-agnostic approach, unlike prior work that considers ad-hoc solutions (Fadaee et al., 2018; Zaninello and Birch, 2020). We find that monolingual pretraining yields strong targeted gains, even when models have not seen any translation examples of the test idioms.

---

[*] This work was done during an internship at Amazon.

[1] Code and data in github.com/amazon-research/idiom-mt

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3682–3700
May 2-6, 2023

We also present an extensive analysis of how different models translate idioms. Specifically, we use a series of probing methods that encode idioms within different contexts (Garcia et al., 2021; Yu and Ettinger, 2020), and measure how this affects the translation outputs and the decoder distributions. We find that the randomly initialized models are more "myopic" compared to the pre-trained ones, as they are relatively unaffected when we vary the idiom context. Our contributions are:

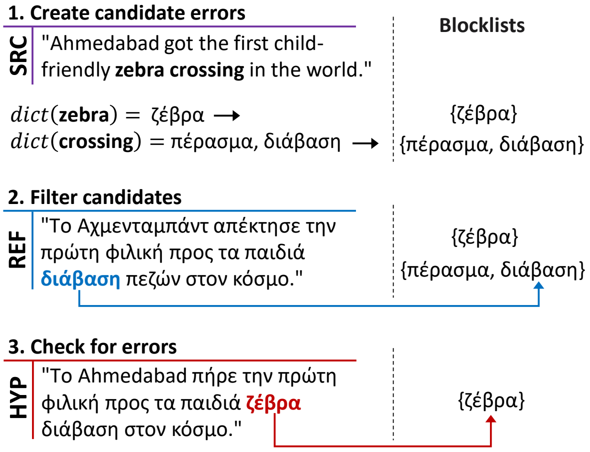
1. We propose LitTER (§2.1), a novel metric for measuring the frequency of literal translation errors, and embed it into a framework (§2) for *automatic* and *targeted* evaluation of idiom translation, complementing prior work.
2. We present translation results (§3.3) in a *controlled* setting and across a wide range of metrics. We find that pre-training on monolingual data yields substantial *targeted* improvements.
3. We present an extensive *analysis* (§4) with a series of probes, showing how context affects idiom translation. We find that models are more uncertain when translating idioms and that pre-training makes models more contextual.

2 Automatic Targeted Evaluation

2.1 Literal Translation Error Rate (LitTER)

We propose literal translation error rate (LitTER), a novel metric of the frequency of literal translation errors made by a model. A literal translation error occurs if any of the words of a span in the source sentence has been *wrongly* translated literally in the target language. Our metric is inspired by the method of Shao et al. (2018) which identifies possible literal translation errors, by checking if a translation output contains any blocklisted words. While this method is effective at capturing these errors, it relies on hand-crafted blocklists. We overcome this limitation by automatically creating word blocklists for a given expression.

Our method, is based on two key ideas. First, we use bilingual word dictionaries[2], which are relatively easy to obtain, to translate the words of an annotated source span into the target language, and produce blocklists with candidate literal translation errors. Then, we use the reference translations to filter the blocklists by removing those words that occur in the reference. This avoids triggering the blocklist when the correct translation is literal.



"Ahmedabad got the first child-friendly **zebra crossing** in the world."

$dict$(**zebra**) = ζέβρα →

$dict$(**crossing**) = πέρασμα, διάβαση →

"Το Αχμενταμπάντ απέκτησε την πρώτη φιλική προς τα παιδιά **διάβαση** πεζών στον κόσμο."

"Το Ahmedabad πήρε την πρώτη φιλική προς τα παιδιά **ζέβρα** διάβαση στον κόσμο."

Figure 1: Overview of the algorithm for the Literal Translation Error Rate (LitTER). For each sentence, we first produce candidate literal translation errors (blocklist), using all the word translations of the source idiom words. Then, we filter the candidates in the blocklist by looking at the reference. Finally, we check if the hypothesis triggers the remaining words in the blocklist.

Algorithm

1. Select from the source text the list of words $\boldsymbol{s} = \langle s_1, s_2, ..., s_N \rangle$ that belong to the annotated expression (i.e., idiom).
2. For each word s_i, obtain all its word translation(s) in the target language using a bilingual word dictionary and add them to a blocklist $b_i = \langle t_1, t_2, \ldots, t_M \rangle$, creating a candidate *list* of blocklists $\boldsymbol{B_s} = \langle b_1, b_2, ..., b_N \rangle$.[3]
3. For each word in the reference (R), search if it occurs in any of the blocklists b_i. If so, remove the corresponding blocklist b_i from $\boldsymbol{B_s}$ to avoid false positives. For example in Figure 1, where words διάβαση and πέρασμα are synonyms, if we remove only διάβαση but leave πέρασμα as a blocklisted word and a model generates it in its translation, this will *wrongly* trigger a literal translation error.
4. Check if the hypothesis contains any blocklisted words. If it does, then we mark this hypothesis as having a literal translation error.

The final score is the percentage of translations that trigger the blocklist. As LitTER requires source-side annotations, we collect test data with idioms on the source side and annotate the spans where they occur (§3.1). Appendix C shows examples of LitTER evaluating real sentences in our data.

---

[2] In this work we use the MUSE (Lample et al., 2018).

[3] In practice, $t_1, t_2, \ldots, t_M$ in a blocklist are synonyms of each other as they are translations of the same source word.

2.2 Alignment-based Evaluation

To measure idiom translation accuracy, we use Alignment-based Phrase Translation Evaluation (APT-Eval), by extending Fadaee et al. (2018) with subword-level metrics. APT-Eval uses word alignments to find the words in the hypothesis and reference sentences, respectively, that align with the annotated idiom source span, and then compares the retrieved matches to each other. We consider two evaluation metrics. First, we use *unigram precision*, that measures the ratio of words in the reference spans that occur in the hypothesis spans, as in Fadaee et al. (2018). We also use ChrF (Popović, 2015), that measures character n-gram overlap.

LitTER vs. APT-Eval While APT-Eval is a targeted evaluation metric, it only measures translation accuracy. This means that given an inaccurate translation, it is impossible to measure whether it has a literal translation error. LitTER, however, quantifies this particular issue that affects NMT.

2.3 Handling Idiom Frequency Imbalances

Different idioms have significantly different frequencies (Appendix A.1). However, prior work has overlooked this fact (Zaninello and Birch, 2020; Fadaee et al., 2018; Shao et al., 2018; Rikters and Bojar, 2017). Thus, over-represented idioms can skew the reported results and favour models that have overfitted on them. To address this, we report all of our targeted evaluation results (i.e., LitTER, APT-Eval) by macro-averaging over idioms:

$$\boldsymbol{E}(\theta) = \frac{1}{|\boldsymbol{L}|} \sum_{j=1}^{|\boldsymbol{L}|} \frac{1}{|\boldsymbol{L_j}|} \sum_{i=1}^{|\boldsymbol{P}|} \boldsymbol{M}(\theta(s_i), t_i) \qquad (1)$$

where $\boldsymbol{L}$ denotes the set of distinct idioms in a test set and $\boldsymbol{P} = \{\langle s_i, t_i \rangle | L_j \in \langle s_i, t_i \rangle\}$ denotes the set of sentence pairs containing the idiom L_j. The model is denoted by θ and the translation of x by $\theta(x)$. We first compute the average score for the test pairs of each idiom with a given metric $\boldsymbol{M}$, and then average these values to produce $\boldsymbol{E}$.

3 Experiments

3.1 Data and Training Splits

We present experiments on en→fr and en→es data. For each language pair, we concatenate the data from Europarl v7[4] (Koehn, 2005), part of the WMT news translation task (Bojar et al., 2014), and from TED talk transcripts released as part of IWSLT 2017 shared task[5] (Cettolo et al., 2017).

Idiom Data We split the parallel data into regular and idiom data using a pattern-matching tool that we developed. Our tool takes as input a list of idioms and extracts sentences from a corpus containing these idioms. We also annotate the span in which each idiom occurs within a sentence, to enable the targeted evaluation metrics. This approach is similar to Fadaee et al. (2018), but we build our tool on top of Spacy's (Honnibal and Montani, 2017) rule-based matching engine. For each phrase in the input list, we automatically create pattern-matching rules that capture complex variations of a given phrase. See Appendix A for details.

In this work, we use a list of 225 English idioms, that we manually collected and make publicly available. We feed this list into our pattern-matching tool, and extract (and annotate) translation pairs that contain an idiom on the source side. The regular data are used *only* for training. The idiom data are further divided into the *idiom-train* and *idiom-test* sets. For each idiom (e.g., "under the weather") in our original idiom data, we put half of its sentence pairs to the idiom-train and the other half into the idiom-test sets, to obtain a balanced distribution. We discard sentences with idioms that occur only once. We conduct controlled experiments in the following testing conditions:

- **Zero**: training data includes only regular parallel data, and we measure how models perform on unseen idioms at test time.
- **Joint**: training data includes the regular and idiom-train data, and we measure how models perform on idioms observed (in a different context) in training data.
- **Upsampling**: same as the joint split, but we up-sample the idiom-train data N times. This setting measures whether it is necessary to up-sample the targeted training data (idiom-train) to achieve better translation quality of idioms.

Evaluation For development, we use the IWSLT dev-set for each language pair. For general purpose translation evaluation, we report results in the WMT newstest14 and IWSLT'17 test sets for en→fr, and in the WMT newstest13 in particular and IWSLT'17 test sets for en→es. For the targeted idiom evaluation (i.e., LitTER and APT-Eval) we

---

[4] www.statmt.org/europarl/

[5] sites.google.com/site/iwsltevaluation2017/TED-tasks

Data	en→fr	en→es
Europarl	2,007,723	1,965,734
IWSLT	275,085	265,625
Combined (after preprocessing)	2,155,543	2,119,686
Regular	2,152,716	2,116,889
Idiom-train	1,327	1,312
Idiom-test	1,383	1,373
WMT-test	3,003	3,000
IWSLT-test	2,632	2,502

Table 1: Dataset statistics

use the extracted idiom-test data per language pair. To generate the word alignments for APT-Eval, we trained a fast-align (Dyer et al., 2013) model on each language-pair's training data. For decoding, we use beam search with beams of size 5, and evaluate all models using BLEU (Papineni et al., 2002) computed with SacreBLEU (Post, 2018).

Preprocessing We first filter out sentence pairs with more than 80 words or with length ratio over 1.5. Then, we tokenize the remaining sentences using sentencepiece[6](SPM; Kudo and Richardson 2018). For the randomly initialized models, we train SPM models with a joint vocabulary of 60K symbols on the concatenation of the source- and target-side of the regular training data. For the mBART fine-tuning experiments, we use the SPM model of mBART (250K symbols).

3.2 Models

Besides training models from scratch, we also investigate how pretraining on monolingual data affects idiom translation, which yields substantial improvements on generic translation quality (Lample and Conneau, 2019; Song et al., 2019; Liu et al., 2020). However, it is not obvious if monolingual data can help idiom translation, as they do not contain any examples with how to translate an idiom from one language into another.

We use mBART (Liu et al., 2020) via finetuning, which is pretrained on monolingual data from many languages. We hypothesize that one way multilingual pre-training can help is by bootstrapping over the source and target language contexts in which idioms occur. We also consider injecting different types of noise during fine-tuning, to corrupt the (encoder or decoder) input context and measure the effects on the targeted evaluation metrics. Specifically, we use source-side word masking and replacement (Baziotis et al., 2021), and target-side word-replacement noise (Voita et al., 2021). In our experiments, "random" denotes a randomly initialized model, while "mBART" stands for using mBART as initialization. For noisy finetuning we train the following variants: "mBART+mask" where we mask 10% of the source tokens, "mBART+replace (enc)" where we replace 10% of the source tokens with random ones, and "mBART+replace (dec)" where we replace 10% of the target tokens with random ones.

Model Configuration For fair comparison, the randomly initialized models use the same architecture as mBART. Specifically, the models are based on the Transformer architecture, with 12 encoder and decoder layers, 1024 embedding size and 16 self-attention heads. Our code is based on the official mBART implementation in Fairseq.

Optimization We optimized our models using Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9, \beta_2 = 0.999$, and ϵ=1e-6. For the random initialization experiments, the models were trained for 140K updates with batches of 24K tokens, using a learning rate of 1e-4 with a linear warm-up of 4K steps, followed by inverted squared decay. For the mBART initialization experiments, the models were trained for 140K updates with batches of 12K tokens, using a fixed learning rate of 3e-5 with a linear warm-up of 4K steps. In all experiments, we applied dropout of 0.3, attention-dropout of 0.1 and label smoothing of 0.1. For model selection, we evaluated each model every 5K updates on the dev set, and selected the one with the best BLEU.

3.3 Results

In this section, for brevity, we discuss a subset of our results, in particular our experiments in en→fr. Results for en→es are consistent with en→fr and are included in Appendix B. Table 2 summarizes all of our main results. Besides global evaluation using BLEU (§3.3.2) on diverse test sets, we also consider two targeted evaluation methods (§3.3.1) that focus on how the idioms are translated using our idiom-test set. For the upsampling split, we upsample the idiom-train data 20x. We also experimented with 100x upsampling, but models started to exhibit overfitting effects (see §B, §D).

3.3.1 Targeted Evaluation

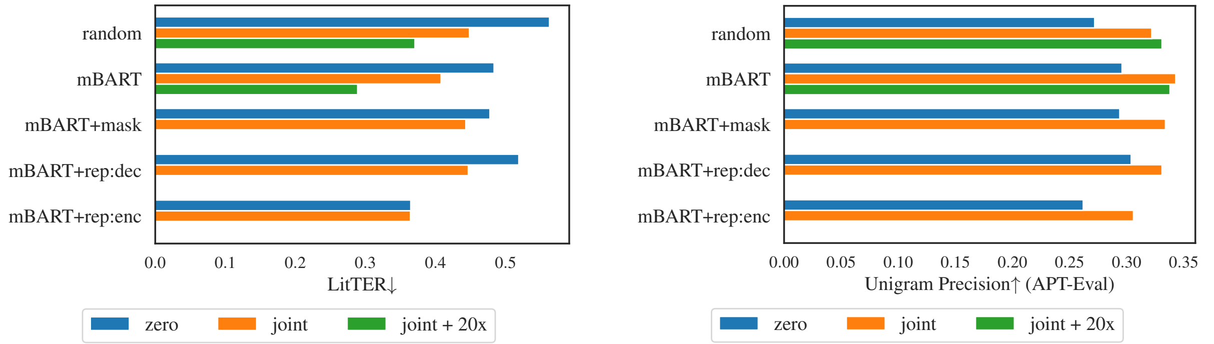
In targeted evaluation, we focus *only* on how models translate the source-side idioms. We present results on our proposed LitTER metric and on APT-

---

[6] We use the `unigram` model with `coverage=0.9999`

Split	Model (en→fr)	LitTER↓	APT-Eval		Global Evaluation (BLEU↑)		
			UniPrec↑	ChrF↑	IWSLT17	WMT14	Idiom-test
zero	random	0.563	0.268	0.298	44.1	34.8	34.4
	mBART	0.484	0.291	0.322	47.0	38.6	36.5
	mBART +mask	0.478	0.298	0.323	46.3	38.2	36.0
	mBART +replace (dec)	0.519	0.295	0.319	46.9	39.0	36.0
	mBART +replace (enc)	0.365	0.260	0.284	44.1	36.2	34.5
joint	random	0.448	0.317	0.337	44.2	34.8	35.3
	mBART	0.408	0.333	0.352	46.5	38.5	37.3
	mBART +mask	0.443	0.315	0.338	46.2	38.3	37.2
	mBART +replace (dec)	0.447	0.317	0.342	46.8	38.8	37.0
	mBART +replace (enc)	0.364	0.300	0.322	44.5	36.6	35.6
upsample 20x	random	0.371	0.323	0.353	44.4	34.7	35.3
	mBART	0.289	0.329	0.346	46.6	38.7	36.0

Table 2: All of our (en→fr) translation results (single run), including *generic* and *targeted* evaluation.



(a) Results on LitTER, which measures how often each model makes literal translation errors.

(b) Results on APT-Eval with unigram precision, which compares (the aligned) reference and hypothesis spans.

Figure 2: Results on *targeted* evaluation of idiom translation.

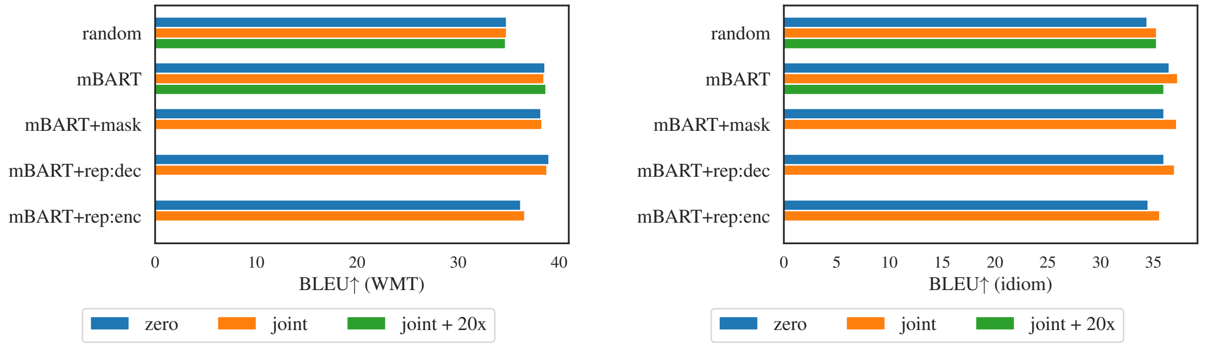
Eval, which provide different information. Recall that we macro-average these scores (§2.3) to account for imbalances in the idiom frequency.

Literal Translation Errors Figure 2a, shows the results on LitTER, that measures how often models make literal translation errors. As expected, all models produce fewer errors when trained on the joint split compared to the zero split. Pretraining gives a significant boost, even on the joint split. This shows that pretraining helps, even though the models have not seen any examples of how to translate the test-set idioms. Upsampling the idiom-train data helps all models regardless of initialization.

Each type of noise induces a different behaviour compared to the mBART model. Masking yields no effect on the zero split, but increases errors on the joint split. Baziotis et al. (2021), show that masking promotes copying, which we speculate it could lead to word-by-word translation and increase LitTER. Decoder-side word replacements yield a similar behaviour in terms of LitTER, which we hypothesize could push the decoder to rely more on the encoder, therefore encouraging word-by-word translation. By contrast, when we add word replacements in the encoder, it greatly reduces LitTER in both splits. This aligns with the findings of Baziotis et al. (2021), who show that source-side word replacements make the decoder less prone to copying (or "trusting") the encoder.

Idiom Translation Accuracy To estimate how accurately models translate idioms, we compare the reference and hypothesis matches that align to the source idiom words. Figure 2b, compares models using unigram precision, and the results are consistent across all APT-Eval metrics (Table 2).

Similar to the LitTER results, the joint split significantly improves idiom translation accuracy as well. Again, pretraining outperforms random initialization, even on the joint split. Upsampling, however, does not yield any consistent improvements. While source-side word replacements reduce literal translation errors, they also degrade idiom translation accuracy. Our hypothesis is that as the decoder becomes less reliant on the encoder,



(a) Regular BLEU results on the *generic* WMT14 test set

(b) Regular BLEU results on our *idiom-test* set.

Figure 3: Results on *global* evaluation (BLEU) on different test sets.

this induces more hallucinations.

3.3.2 Global Evaluation

Here, we discuss how models perform based on global translation evaluation using BLEU.

General Purpose Figure 3a, shows the results on the WMT14 test set, but we note that the results are generally consistent with the IWSLT17 test set (Table 2). The mBART intialized models, unsuprisingly, yield significantly better results than random initialization. As expected, there is no measurable difference between splits, not even when upsampling idioms, as both IWSLT17 and WMT14 are generic test sets. Noisy finetuning methods fail to improve results. Encoder-side word replacements even degrade overall performance, which aligns with the hypothesis that they induce hallucinations.

Idiom The results on our idiom-test set (Figure 3b) show that models perform consistently better when trained on the joint split. Global evaluation, however, considers the full sentences and the impact of idiom translation is overshadowed (Rikters and Bojar, 2017), which can be seen by the very small differences between splits. This prevents fine-grained comparisons between models and highlights the need for targeted evaluation.

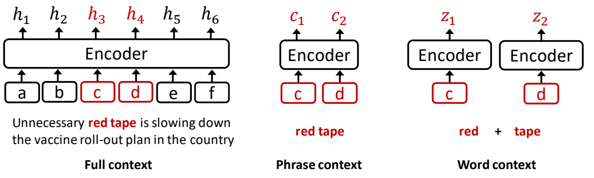


Figure 4: Illustration of how we obtain idiom representations by varying the available context.

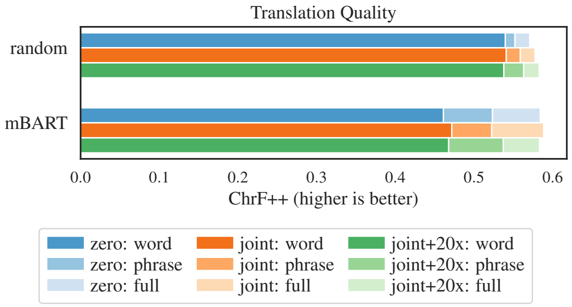


Figure 5: Variation in translation quality, measured in ChrF, as we vary the idiom representations. The length of each (lighter) bar encodes the difference from its (darker) bar to its left (i.e., overlapping bars effect).

4 Analysis

To further understand how models translate idioms, we present an extensive analysis with a series of probes, focused on the role of idiom context. Specifically, using the annotations in our idiom-test data, we encode the idiom words within different contexts (Figure 4) and measure how it affects the decoder distributions and the translation output. We consider (1) *full context*, in which we encode the idiom phrase within the whole input sentence, (2) *phrase-level context*, in which we encode the idiom phrase in isolation, (3) *word-level context*, in which we encode each idiom word independently. Our probes follow a similar approach to prior work that evaluate the idiomaticity of (pretrained) Transformer-based models (Garcia et al., 2021; Tayyar Madabushi et al., 2021) by measuring how idiom representations are affected by their context (Yu and Ettinger, 2020), but we extend these methods to analyze how (pretrained) NMT models translate idioms. For brevity, we discuss here our most important findings, focusing on random-vs-mBART initialization. However, we include the rest of our results in Appendix D for completeness.

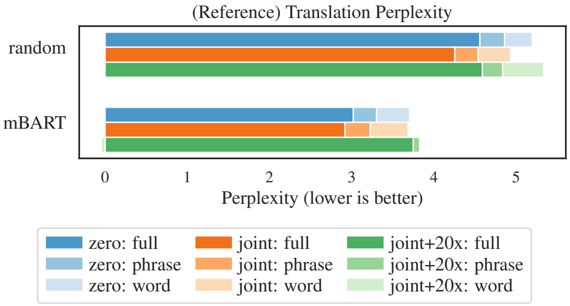


Figure 6: Perplexity of the references, as we vary the idiom representations. The length of each (lighter) bar encodes the difference from its (darker) bar to its left.

4.1 Variation in Translation Performance

In this probe, we decode (i.e., translate) different encoder representation and evaluate the samples against the reference (sentence) translations. Specifically, we first encode each (full) input sentence and then replace the encoder representations belonging to idiom words with those obtained with different (narrower) contexts. Figure 5 shows the results using ChrF, rather than BLEU, as a metric to capture even small subword-level changes.

Across all models, reducing the context (i.e., darker shades) results in worse translation scores. This is expected, as by swapping the original (full context) idiom representations with those obtained with word-context, we essentially remove information. When we use the full-context (i.e, lightest shades) pretraining yields the best results, but when we reduce the idiom context, the pretrained model suffers significantly, unlike the randomly initialized that is barely affected. This implies that the representations of the randomly initialized model are more local (or “myopic”), containing information mainly related to the idiom tokens. By contrast, the representations of the pretrained model are more global, and contain information related to the surrounding idiom context (Brunner et al., 2020). Upsampling does not have a strong effect.

4.2 Variation in Translation Likelihood

In this probe, we vary the encoder idiom representations, as before, and measure how this affects the likelihood of the reference translations. We score each reference translation by computing its perplexity under the model, given each encoder output sequence. Figure 6, shows the results, with *lighter* shades corresponding to *narrower* contexts.

Using more context improves (i.e., lowers) the perplexity of the references across models. Pretraining endows models with stronger LM capabilities which is probably why it yields generally lower perplexities. Training on the joint split yields consistent improvements which are more pronounced in the randomly initialized models. By contrast, upsampling the idiom-train data yields negative effects, which we attribute to overfitting, as it makes all other sentences less probable under the model.

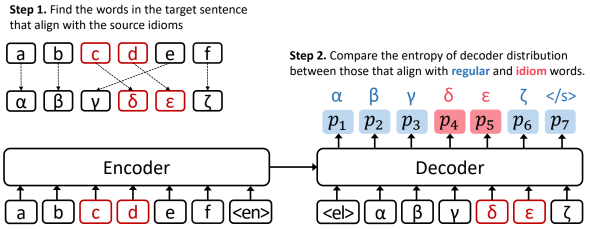


Figure 7: Illustration of how we compute the effects of different source idiom contexts on the model’s uncertainty during translation (i.e., decoding).

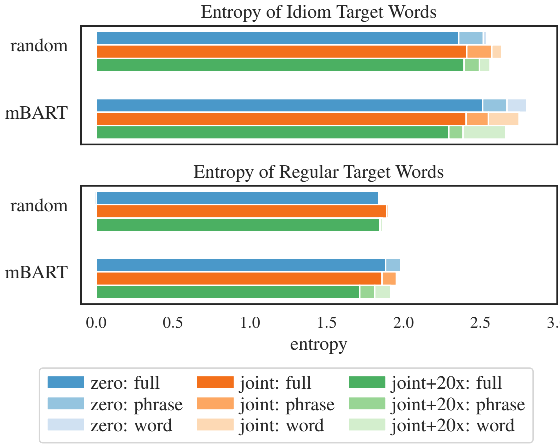


Figure 8: Model uncertainty for the translation of regular vs. idiom words. The length of each (lighter) bar encodes the difference from its (darker) bar to its left.

4.3 Decoder Uncertainty

Next, we focus on how the token-level uncertainty of the decoder varies while it translates idiom vs. non-idiom words. First, we translate each sentence pair with teacher-forcing, where we feed to the decoder the reference translation (i.e., ground truth) as input, instead of its output from each step. Then, we measure the entropy of the decoder’s distributions for each (reference) target token. Finally, using word alignments, we average separately the entropy values of target words[7] that are aligned to idiom and non-idiom source words. Figure 7 illustrates how the probe works. Figure 8 shows the

---

[7] After the word alignment step, we find which target SPM tokens correspond to which words.

results, in which *lighter* shades correspond to *narrower* contexts.

All models have significantly higher uncertainty when they translate idiom words (top section) compared to regular words (bottom section). This confirms our expectation that it is harder to translate idioms. When translating regular words, the randomly initialized models are unaffected by changes in the idiom representations, whereas reducing the idiom context increases the uncertainty of the pretrained models. This is another piece of evidence that pretraining yields less local models (§4.1). When translating idiom words, including or upsampling the idiom-train data benefits the pretrained model, but not the randomly initialized one.

5 Discussion

Global Metrics LiTER does not aim to replace global evaluation metrics like BLEU, but to complement them. Global metrics estimate the general translation quality of model outputs, which is undoubtedly important. However, they consider the full sentence, and as result the effects of idiom (mis)translation are overshadowed (see §3.3.2). LiTER aims to fill this gap by providing additional insights to practitioners with targeted evaluation.

LitTER LiTER should not be used in isolation, but combined with other (global/targeted) metrics. The reason is that lower LiTER can be achieved by more accurate idiom translations or by hallucinations. We aim to enable practitioners to make informed decisions without running human evaluations in the model development phase. The goal is to produce models that improve on LitTER without sacrificing general translation quality (e.g., BLEU). Our experiments reveal this contrast, where unsupervised pretraining achieves this goal, whereas (some of) the noisy variants fail. We expect LiTER to be more relevant when developing NMT models for creative content (e.g., subtitles, social media text) that usually contains figurative language.

Alignment Metrics Word alignment-based methods aim to capture idiom translation accuracy that complements LitTER. However, they are sensitive to the literal meaning of words to produce the alignments. Thus, while alignment-based methods could be reliable in certain types of evaluation, such as gender translation (Stanovsky et al., 2019), we believe that with the current techniques they should be used with caution for idiom translation evaluation. Although we did not systematically study this issue, we discovered by manual inspection that it was not uncommon to produce noisy or empty alignments. We chose statistical over embedding-based methods as they yielded less empty alignments (§B.2).

6 Related Work

Idioms in NMT There is limited research on idioms in NMT. Zaninello and Birch (2020), explore augmenting the training data with MWE translations from dictionaries, backtranslating (Sennrich et al., 2016) target-side sentences with MWEs, or wrapping the (source) MWEs with special tokens. Fadaee et al. (2018) prepend a special token in source sentences that contain an idiom. Gamallo and Garcia (2019) do unsupervised translation of MWEs by composing cross-lingual word embeddings, but this is fundamentally incapable of translating idioms which are non-compositional. Instead of using ad-hoc solutions that change the model or the data pipeline, we use monolingual pretraining, which is a more generic and less invasive approach.

Targeted Evaluation Using word alignments is a straightforward approach for the targeted evaluation of words or phrase translation (Stanovsky et al., 2019). Fadaee et al. (2018), use word alignments to compare the reference and hypothesis matches that translate a given source idiom. Zaninello and Birch (2020) first align words in the hypothesis and the reference using edit-distance and then score the aligned words using character-level matching. Alignment-based methods capture idiom translation accuracy, but they do not account for literal translation errors which are a major issue in idiom translation (Fadaee et al., 2018). The method of Shao et al. (2018) is capable of estimating the frequency of such errors, but it requires the creation of language-specific hand-crafted lists. In this work, we lift this limitation and enable the automatic evaluation of literal translation errors.

Analysis Rikters and Bojar (2017) investigate the attention mechanism in NMT during the translation of MWEs. Garcia et al. (2021) evaluate the idiomaticity of Transformer-based models using probes that measure how idiom representations are affected by their context (Yu and Ettinger, 2020). Similarly, Tayyar Madabushi et al. (2021) investigate the idiomaticity of pretrained encoders, such as BERT (Devlin et al., 2019), in a *monolingual* setting. By contrast, we present an analysis of how (pretrained) NMT models *translate* idioms.and

Concurrent analysis works explore idioms in NMT, by also using the blocklist method (Shao et al., 2018) as part of their analysis. Dankers et al. (2022a) study the compositionality of (Transformer) NMT models and (among others) find that models trained on more (parallel) data are more compositional. This relates to the results of our analysis (§4), which shows that pretrained models are less local than those trained only on the downstream parallel data. Dankers et al. 2022b analyze the hidden states and attention patterns of Transformer NMT models when processing idioms.

7 Conclusions

We present a comprehensive study of idiomatic expressions in NMT, aiming to facilitate future research on the topic. We propose LitTER (§2.1), a novel metric that enables the *automatic* evaluation of literal translation errors. LitTER is used for *targeted* evaluation and aims to complement and not replace global metrics, such as BLUE or ChrF, which consider the full sentence and can only measure the *overall* translation quality. We evaluate models in controlled conditions, with and without the test set idioms (i.e., zero-shot). We explore pretraining on monolingual data for improving idiom translation, as parallel idiom data is difficult to come by. Interestingly, we find that pretraining achieves strong targeted improvements, even in the zero-shot setting (§3.3.1). We also present a systematic analysis (§4) that investigates the role of context in idiom translation. We find evidence that pretraining yields more contextual models, which helps to explain why it contributes to better idiom translations. We also quantitatively confirm that idioms are more difficult to translate than regular words and strongly depend on the source context.

Limitations

LitTER is a novel metric that sidesteps the need for human involvement to estimate the frequency of literal translation errors, which is a major problem in idiom translation by NMT models. However, in its current iteration it has certain limitations:

1. An edge case with LitTER is that if the blocklist words appear as a result of translating other words - not part of the non-literal phrase - we will still count it as an error. We leave this as future work.

2. Our experiments were conducted on languages with relatively simple morphology. Preliminary experiments with German revealed that LitTER struggles with compound words. We did address these issues with custom rules, but we aim to study these cases in future versions of LitTER systematically.

3. The metric can be ambiguous when used in isolation. We recommend pairing it with standard evaluation metrics when comparing translation models.

Acknowledgments

We thank Marcello Federico, Georgiana Dinu, Anna Currey and Hoang Cuong for the helpful discussions and feedback during this project.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA. 1

Christos Baziotis, Ivan Titov, Alexandra Birch, and Barry Haddow. 2021. Exploring unsupervised pretraining objectives for machine translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2956–2971, Online. Association for Computational Linguistics. 4, 5

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics. 3

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *International Conference on Learning Representations*. 7

Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Niehues Jan, Stüker Sebastian, Sudoh Katsuitho, Yoshino Koichiro, and Federmann Christian. 2017. Overview of the iwslt 2017 evaluation campaign. In *International Workshop on Spoken Language Translation*, pages 2–14. 3

Mathieu Constant, Gülşen Eryiğit, Johanna Monti, Lonneke van der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. Survey: Multiword expression processing: A Survey. *Computational Linguistics*, 43(4):837–892. 1

Paul Cook, Jelena Mitrović, Carla Parra Escartín, Ashwini Vaidya, Petya Osenova, Shiva Taslimipoor, and Carlos Ramisch, editors. 2021. *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*. Association for Computational Linguistics, Online. 1

Verna Dankers, Elia Bruni, and Dieuwke Hupkes. 2022a. The paradox of the compositionality of natural language: A neural machine translation case study. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4154–4175, Dublin, Ireland. Association for Computational Linguistics. 9

Verna Dankers, Christopher Lucas, and Ivan Titov. 2022b. Can transformer be too compositional? analysing idiom processing in neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3608–3626, Dublin, Ireland. Association for Computational Linguistics. 9

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics. 8

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online. Association for Computational Linguistics. 14

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics. 4, 14

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2018. Examining the tip of the iceberg: A data set for idiom translation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA). 1, 3, 8, 12

Pablo Gamallo and Marcos Garcia. 2019. Unsupervised compositional translation of multiword expressions. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, pages 40–48, Florence, Italy. Association for Computational Linguistics. 8

Marcos Garcia, Tiago Kramer Vieira, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2021. Probing for idiomaticity in vector space models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3551–3564, Online. Association for Computational Linguistics. 2, 6, 8

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear. 3, 12

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA. 4

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand. 3

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics. 1

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics. 4

Guillaume Lample and Alexis Conneau. 2019. Crosslingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*. 4

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*. 2

Dilin Liu. 2017. *Idioms: Description, comprehension, acquisition, and pedagogy*. Routledge. 1

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pretraining for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742. 1, 4

Geoffrey Nunberg, Ivan A. Sag, and Thomas Wasow. 1994. Idioms. In Stephen Everson, editor, *Language*, pages 491–538. Cambridge University Press. 1

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. 1, 4

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics. 3, 13

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics. 4

Matīss Rikters and Ondřej Bojar. 2017. Paying attention to multi-word expressions in Neural Machine Translation. In *Proceedings of MT Summit XVI. Main Conference: Research Track* , volume 1, pages 86–95. 3, 6, 8

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics. 8

Yutong Shao, Rico Sennrich, Bonnie Webber, and Federico Fancellu. 2018. Evaluating machine translation performance on Chinese idioms with a blacklist method. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA). 1, 2, 3, 8, 9

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR. 4

Gabriel Stanovsky, Noah A. Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684, Florence, Italy. Association for Computational Linguistics. 8

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112. 1

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics. 6, 8

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 5998–6008, Long Beach, CA, USA. 1

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Analyzing the source and target contributions to predictions in neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1126–1140, Online. Association for Computational Linguistics. 4

Lang Yu and Allyson Ettinger. 2020. Assessing phrasal representation and composition in transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4896–4907, Online. Association for Computational Linguistics. 2, 6, 8

Andrea Zaninello and Alexandra Birch. 2020. Multiword expression aware neural machine translation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3816–3825, Marseille, France. European Language Resources Association. 1, 3, 8

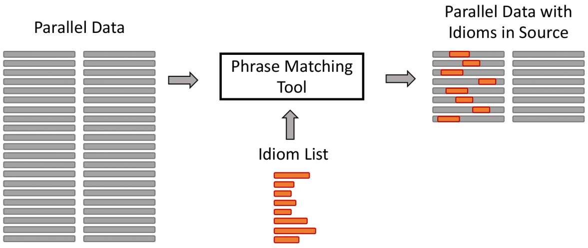
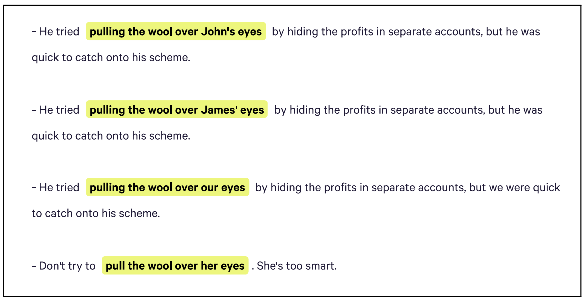


Figure 9: Overview of the process for collecting the idiom parallel data.



- He tried **pulling the wool over John's eyes** by hiding the profits in separate accounts, but he was quick to catch onto his scheme.

- He tried **pulling the wool over James' eyes** by hiding the profits in separate accounts, but he was quick to catch onto his scheme.

- He tried **pulling the wool over our eyes** by hiding the profits in separate accounts, but we were quick to catch onto his scheme.

- Don't try to **pull the wool over her eyes** . She's too smart.

Figure 10: Overview of the process for collecting the idiom parallel data.

A Idiom Data Collection

We collected parallel data with idioms in the source side and annotated the spans in which the idioms occur within each sentence. This enables us to conduct controlled experiments and to support our targeted evaluation metrics and our analysis. To collect the data, we created a phrase-matching tool that searches for idioms in parallel data and extracts and annotates the retrieved pairs (Figure 9).

Phrase-Matching Tool Our tool, which we make publicly available, uses rule-based matching to search for sentences that contain phrases specified in user-defined phrase list. While in this work we use it to create parallel data with idioms, it could be used to create datasets with different types of phrases. We build our tool on top of Spacy's (Honnibal and Montani, 2017) rule-based matching engine[8] that is more flexible and easy to work with than using regular expressions or custom rules (Fadaee et al., 2018). It allows us to do pattern matching over linguistic units, such as parts-of-speech or even dependency relations, thus capturing complex variations of a given phrase. Our tool first reads the input phrase list, and for each phrase, it automatically creates a pattern based on some simple rules and assumptions. We created separate idiom train and test data for each language-pair and we describe the process in Section 3.1.

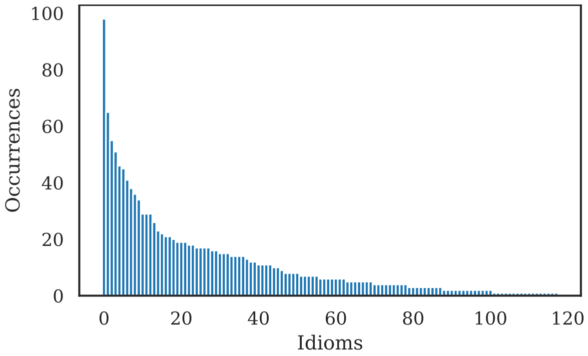


Figure 11: Occurrences of idioms in the en→fr idiom-test sets.

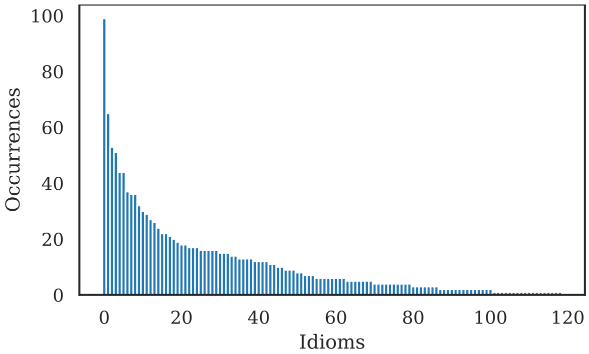


Figure 12: Occurrences of idioms in the en→es idiom-test sets.

Figure 10, shows an actual example[9] of the different variations of the idiom "pull the wool over someone's eyes" that our tool captures. Notice that it matches:

- Different variations of the verb "pull".
- Different words in the place of the word "someone".
- Importantly, it optionally matches the particle "'s". This shows that we can apply logic base on the part-of-speech (POS) or other linguistic properties of words, and in this case optionally skip them.

A.1 Idiom-test Statistics

Figure 11 and Figure 12 show the occurrences of idioms in the en→fr and en→es idiom-test sets, respectively. We see that in both test sets the idiom statistics follow very similar distributions. This verifies that different idioms have very different

---

[8] `spacy.io/usage/rule-based-matching`

[9] You can view this example in this online demo. It shows the rule we generate for the idiom phrase and the variants it captures.

Split	Model (en→fr)	LitTER↓	APT-Eval (fast-align)		APT-Eval (awesome-align)		Global Evaluation BLEU↑			Global Evaluation ChrF↑		
			UniPrec↑	ChrF↑	UniPrec↑	ChrF↑	IWSLT17	WMT14	idiom	IWSLT17	WMT14	idiom
zero	random	0.563	0.268	0.298	0.272	0.319	44.1	34.8	34.4	0.67	0.61	0.60
	mBART	0.484	0.291	0.322	0.296	0.347	47.0	38.6	36.5	0.69	0.64	0.61
	mBART +mask	0.478	0.298	0.323	0.294	0.339	46.3	38.2	36.0	0.68	0.64	0.61
	mBART +replace (dec)	0.519	0.295	0.319	0.304	0.346	46.9	39.0	36.0	0.69	0.64	0.61
	mBART +replace (enc)	0.365	0.260	0.284	0.262	0.306	44.1	36.2	34.5	0.66	0.62	0.60
joint	random	0.448	0.317	0.337	0.322	0.365	44.2	34.8	35.3	0.67	0.61	0.61
	mBART	0.408	0.333	0.352	0.343	0.384	46.5	38.5	37.3	0.68	0.64	0.62
	mBART +mask	0.443	0.315	0.338	0.334	0.379	46.2	38.3	37.2	0.68	0.64	0.62
	mBART +replace (dec)	0.447	0.317	0.342	0.331	0.379	46.8	38.8	37.0	0.69	0.64	0.62
	mBART +replace (enc)	0.364	0.300	0.322	0.306	0.350	44.5	36.6	35.6	0.66	0.62	0.61
upsample 20x	random (20x)	0.371	0.323	0.353	0.331	0.382	44.4	34.7	35.3	0.67	0.61	0.61
	mBART (20x)	0.289	0.329	0.346	0.338	0.376	46.6	38.7	36.0	0.68	0.64	0.61
upsample 100x	random (100x)	0.378	0.314	0.343	0.325	0.363	43.8	34.7	33.9	0.66	0.61	0.60
	mBART (100x)	0.289	0.337	0.358	0.347	0.389	46.8	38.2	35.8	0.69	0.64	0.61

Table 3: Translation results in en→fr. The results involve a single run, but include mutliple test sets and are consistent across the board.

Split	Model (en→es)	LitTER↓	APT-Eval (fast-align)		APT-Eval (awesome-align)		Global Evaluation BLEU↑			Global Evaluation ChrF↑		
			UniPrec↑	ChrF↑	UniPrec↑	ChrF↑	IWSLT17	WMT13	idiom	IWSLT17	WMT13	idiom
zero	random	0.541	0.350	0.364	0.351	0.370	36.0	31.5	38.9	0.62	0.57	0.63
	mBART	0.476	0.383	0.385	0.368	0.390	38.5	34.0	40.8	0.64	0.59	0.64
	mBART +mask	0.481	0.369	0.378	0.354	0.380	38.8	34.0	40.5	0.64	0.59	0.64
	mBART +replace (dec)	0.508	0.388	0.389	0.384	0.401	38.6	34.4	40.4	0.64	0.59	0.64
	mBART +replace (enc)	0.389	0.334	0.345	0.323	0.351	37.0	32.1	39.0	0.62	0.57	0.62
joint	random	0.468	0.385	0.395	0.382	0.397	35.9	31.8	39.7	0.62	0.57	0.64
	mBART	0.412	0.399	0.406	0.393	0.419	38.8	33.9	41.9	0.64	0.59	0.65
	mBART +mask	0.418	0.389	0.402	0.388	0.410	38.7	33.9	41.9	0.64	0.59	0.65
	mBART +replace (dec)	0.443	0.402	0.408	0.402	0.414	38.5	33.9	41.7	0.64	0.59	0.65
	mBART +replace (enc)	0.352	0.352	0.372	0.355	0.378	36.7	32.2	39.7	0.62	0.57	0.63
upsample 20x	random (20x)	0.400	0.410	0.424	0.415	0.440	35.9	31.7	40.0	0.62	0.57	0.64
	mBART (20x)	0.301	0.422	0.435	0.419	0.444	38.8	34.0	40.6	0.64	0.59	0.64
upsample 100x	random (100x)	0.391	0.406	0.420	0.407	0.435	36.2	31.8	38.9	0.62	0.57	0.63
	mBART (100x)	0.299	0.416	0.427	0.406	0.441	38.7	33.9	40.5	0.64	0.59	0.64

Table 4: Translation results in en→es. The results involve a single run, but are consistent across the board.

frequencies, and highlights the need for macro-averaging the scores of targeted evaluation metrics. Failure to do so would promote models that have perform best on the most frequent idioms over those that have a more balanced performance.

Also, note that the frequencies of idioms in the idiom-train and idiom-test sets are identical, as described in Section 3.1. While we have not computed the idiom frequencies in the monolingual data used to pretrain mBART, we expect that they would follow a similar distribution as the one found in the monolingual data we used in our work.

B Translation Results

In this section, we present all of our translation results in detail. Table 3, shows our results in en→fr, while Table 4 shows our results in en→es. Besides BLEU, we also include results with ChrF (Popović, 2015), for global translation evaluation. overall the results are *consistent* across language pairs in all evaluation methods. The fact that the absolute scores reached by the model on each each test set are different is natural, as the test sets themselves are different to each other between languages. However, the relative performance between model across language pairs is the same, with only minor differences.

B.1 Targeted Evaluation

In Figure 13, we visualize how model perform on our targeted evaluation methods for both en→fr and en→es. Recall that, LitTER, measures how often each model makes literal translation errors. APT-Eval, measures idiom translation accuracy, by comparing the reference and (aligned) hypothesis spans that translate the source idiom words. In these plots, we use present how models performed on unigram precision metric for APT-Eval, similar to the main paper. We observe that both in terms of literal translation errors (Figure 13a, 13c), as well as idiom translation accuracy (Figure 13b, 13d), the results are remarkably consistent across languages. Upsampling the idiom-train data helps all models regardless of initialization, but upsampling more than 20x does not yield consistent improvements.

B.2 Word Alignment Models

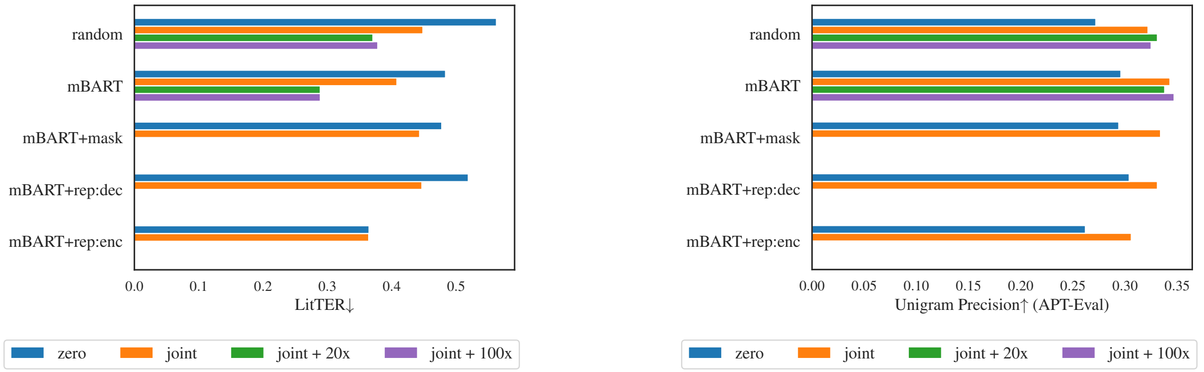
APT-Eval, requires word alignments to map the source idiom words to the words of the reference and hypothesis spans, respectively. In our main paper, we presented results using fast-align (Dyer et al., 2013) with word alignment models trained on the training data of each language-pair. We also experimented with awesome-align (Dou and Neubig, 2021), that doesn't require any training, and uses the token similarities of the pretrained mBERT models to obtain the alignments. After analysis, we found that fast-align produced empty matches for the idiom words in 2.2% of the reference sentences in our idiom-test set. Awesome-align, however, yielded more empty matches, which after tweaking its threshold parameter[10], we managed to reduce it to 3%. While we omitted the awesome-align results from the main paper, we include them in Tables 3, 4 for completeness. We observe that while the absolute APT-Eval scores are different between the two alignment methods, the relative performance across models is consistent.

B.3 Regular Translation Evaluation

Here, we visualize our results on *regular* MT evaluation for both en→fr and en→es. In Figure14, we compare models in both language pairs and for both generic test sets as well as on our idiom-test sets. Overall, we observe that the results are very consistent between language pairs, similar to the targeted evaluation results (§B.1), which improves our confidence in them. As we already noted in our paper, we find that including or upsampling idiom training data has no measurable effect on generic test sets, unlike on our idiom-test set.

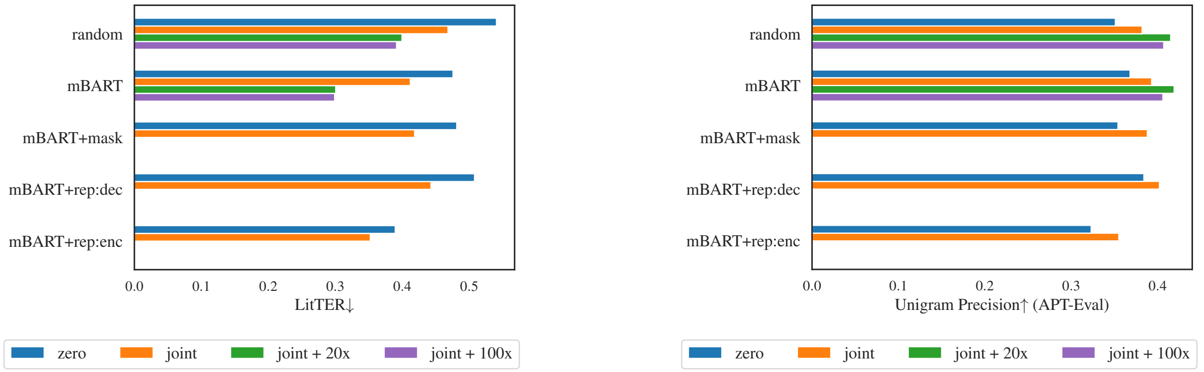
---

[10] We used the following hyper-parameters: `extraction=softmax, softmax_threshold=0.001`



(a) Results on LitTER for en→fr.

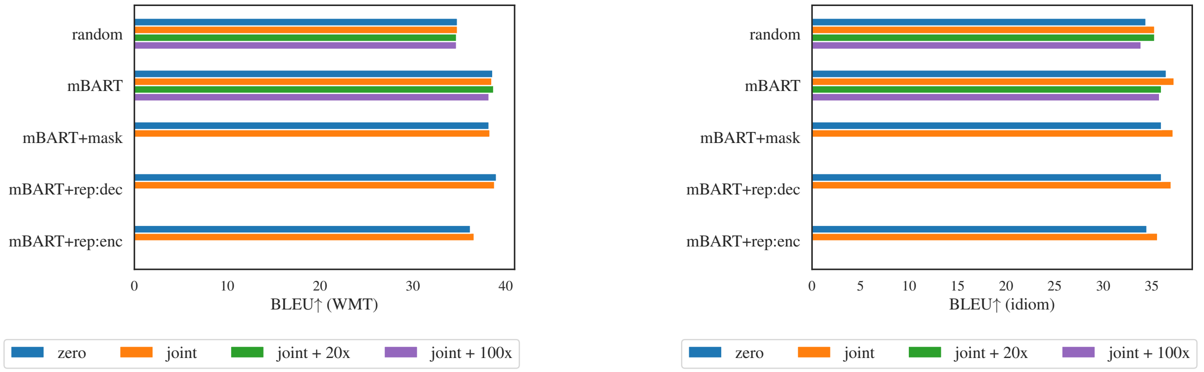
(b) Results on APT-Eval (unigram precision) for en→fr.



(c) Results on LitTER for en→es.

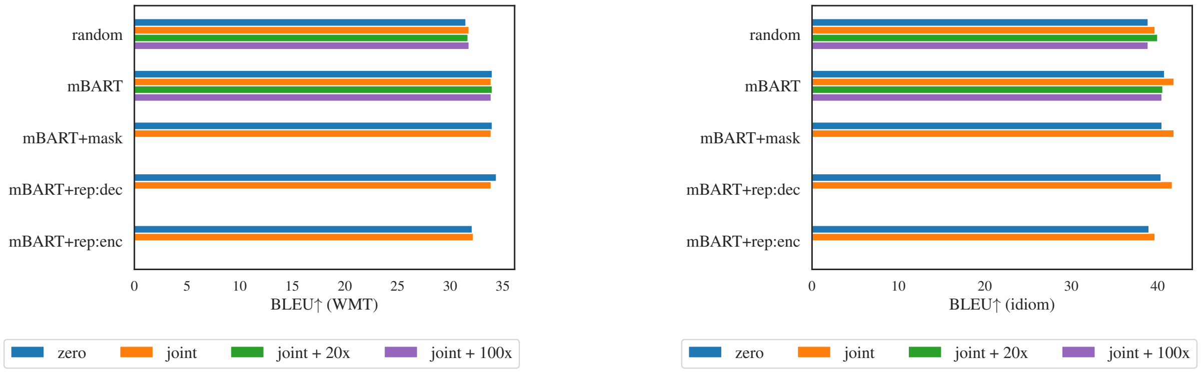
(d) Results on APT-Eval (unigram precision) for en→es.

Figure 13: Results on *targeted* evaluation of idiom translation.



(a) BLEU results on the en→fr WMT14 test set

(b) Regular BLEU results on our en→fr *idiom-test* set.



(c) BLEU results on the en→es WMT13 test set

(d) Regular BLEU results on our en→es *idiom-test* set.

Figure 14: Results on *regular* MT evaluation with BLEU on generic as well as on our idiom-test sets.

C LitTER

C.1 Indicative Examples

Here, we present some indicative examples of how LitTER evaluates translation outputs in our idiom test data. Figure 16, contains examples where LitTER is working as intended, whereas Figure 15, contains a two failures of LitTER.

SRC: As the example of Cyprus shows, Ankara does not pull its punches.
REF: Comme le montre l'exemple de Chypre, Ankara n'y va pas avec le dos de la cuiller.
HYP: Comme le montre l'exemple de Chypre, Ankara ne tire pas les ficelles.

Blocklists
pull → {tirez, tirer}
its → {ses, son, sa}
punches → {coups}

No error detected.

SRC: [..] it was already being put on ice on the grounds that 'We'll never get it though the G20'.
REF: [..] elle était mise au rencart au motif que "nous n'arriverons jamais à convaincre le G20".
HYO: [..] on l'a déjà gelé au motif que "nous n'y arriverons jamais par le biais du G20".

Blocklists
put → {mis, mettre}
on → {sur}
ice → {glace, ice, verglas}

No error detected.

Figure 15: LitTER failures on our en→fr idiom test set. In the first example, the blocklist is not triggered because the inflected form *tire* is missing from the blocklist. In the second example, the verb form *gelé* (freeze) is not contained in the blocklist but is a literal translation of *ice* in a wider sense.

SRC: To postpone this vote one more time would be to bark up the wrong tree.
REF: Postposer ce vote une fois de plus eut été se tromper de cible.
HYP: Reporter ce vote une fois de plus, c'est se tromper d'arbre.

Blocklists
bark→ {aboyer, ecorces, ecorce}
up→ {debout}
the→ {le, la, les}
wrong→ {faux, tort, errone, mal}
tree→ {arbre, arbres, sapin, arborescence}

ERROR: Blocklist triggered by {arbre}

SRC: For companies, using technology to gather important data, its like bread and butter.
REF: Pour les sociétés, utiliser la technologie pour recueillir des données, c'est la routine.
HYP: Pour les entreprises, utiliser la technologie pour collecter des données importantes, c'est comme du pain et du beurre.

Blocklists
bread→ {pain}
and→ {et}
butter→ {et, pain, beurre}

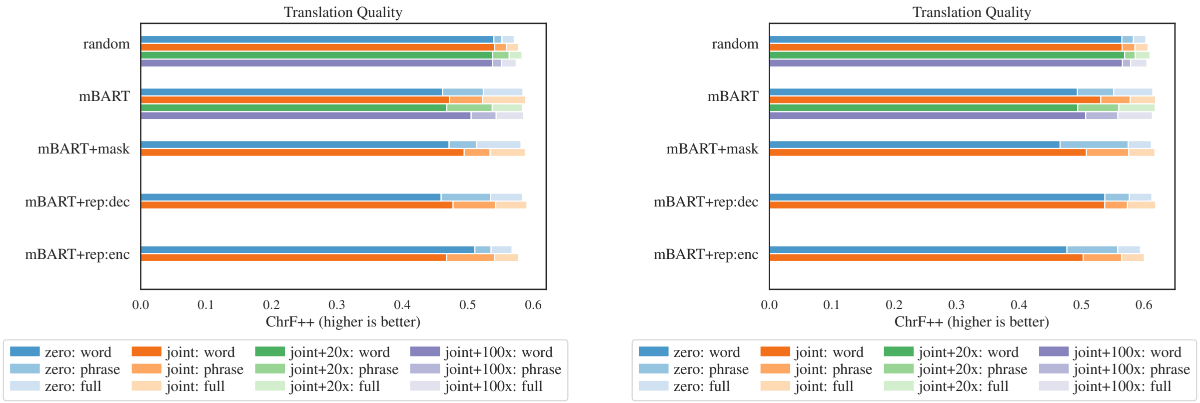
ERROR: Blocklist triggered by {et, pain, beurre}

SRC: And here is some eye candy for you, from a range of DIY scientists and artists from all over the globe.
REF: Et voici quelques bonbons pour vos yeux, de la part d'un éventail de scientifiques et des artistes bricoleurs de tous les coins de la planète.
HYP: Et voici quelques bonbons pour les yeux, d'une gamme de scientifiques et d'artistes du bricolage du monde entier.

Blocklists
eye → {oculaire, oeil, yeux, œil}
candy → {bonbon, bonbons, sucrerie}

No error detected.

Figure 16: Examples of LiTER evaluation on sentences in our en→fr idiom test set. In the first two examples, the model makes a literal translation error and the error is captured by LitTER. In the third example, the literal translation is correct and the blocklist is not triggered, thanks to the 3rd step in our algorithm (§ 2.1).



(a) Results for the en→fr models.

(b) Results for the en→es models.

Figure 17: Variation in translation quality, measured in ChrF, as we vary the idiom representations. The length of each (lighter) bar encodes the difference from its (darker) bar to its left (i.e., overlapping bars effect). The darkest shades correspond to using the full (original) context for encoding idiom words, and each lighter shades correspond to narrower contexts. Overall, the results are consistent, with the exception of the “mBART+replace(enc)” model.

D Analysis

In this section we present all of our analysis results. Specifically, we include results on en→es, with the (mBART) pretrained models finetuned with different noising methods, and additional probes. For the noisy versions of mBART finetuning, we present results on the zero and joint split. Recall that, in most of our probes, we evaluate the role of (idiom) context idiom translation. To do this, we encode the idiom words within different contexts and compare how this affects various aspects of each model. Figure 4, illustrates the process by which we obtain the encoding for each context. We consider the following context: (1) *full context*, in which we encode the idiom phrase within the whole input sentence, (2) *phrase-level context*, in which we encode together only the words idiom phrase, (3) *word-level context*, in which we encode each idiom word independently.

Figure 18, shows a visual example of how this probe works.

D.1 Variation in Translation Performance

With this probe, we test how the variation of the idiom encoder representations is reflected in the translation output. Figure 18, shows a visual example of how this probe works. First we encode each input sentence and then replace the encoder output representations belonging only to idiom words with those obtained with different (narrower) contexts. Finally, we decode each encoder output sequence and compare the generated translation to the reference translation.

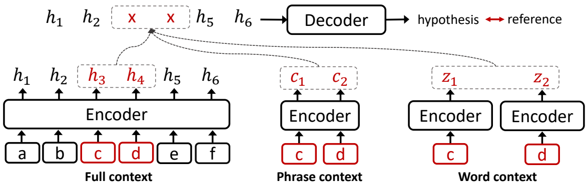
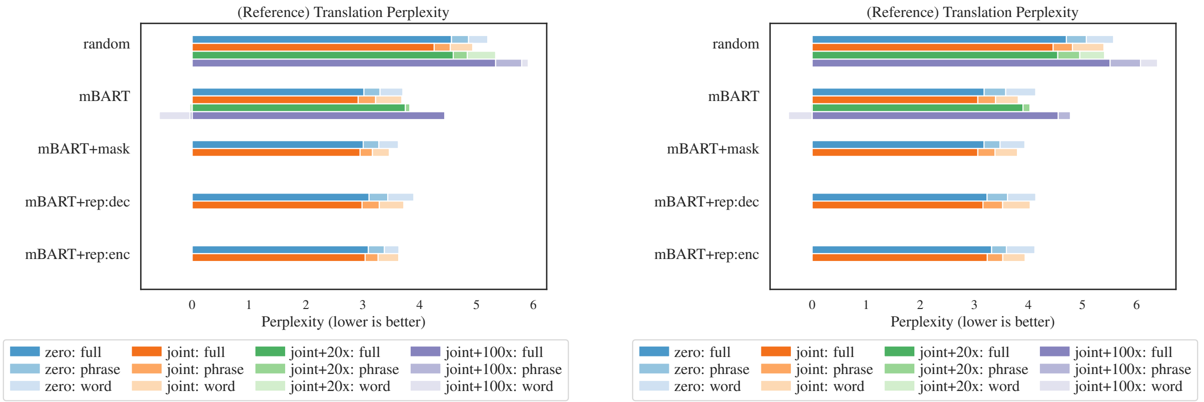


Figure 18: Illustration of how we probe the impact of different idiom contexts on translation quality. First, we obtain each sequence of idiom token representations, by encoding them within different contexts. Then we feed each sentence to the encoder, and before passing its outputs to the decoder, we replace the idiom token representations with those whose context we want to probe. Finally, we sample a translation from the decoder and compare it against the reference.

We observe that when using the full context, the results are generally consistent across language pairs and models. As we discussed in the main paper, the mBART-initialized model suffers greatly when the idiom representations are encoded with narrower context, in contrast to the randomly initialized model. We believe this is an indication that the mBART-initialized model is less local, meaning that each token representation contains to a large degree information about the rest of the tokens. We also observe that this behaviour is exhibited by all pretrained models.

However, while results are generally consistent in both language pairs, we do see some *small* discrepancies as we probe the effects of narrower contexts, in particular for the “mBART+replace(enc)” model. We do not have a satisfying explanation for this performance difference, which occurs only



(a) Results for the en→fr models.

(b) Results for the en→es models.

Figure 19: Variation in perplexity of reference translations, as we vary the idiom representations. The length of each (lighter) bar encodes the difference from its (darker) bar to its left. Negative bar lengths indicate a decrease relative to the (darker) bar before it. The darkest shades correspond to using the full (original) context for encoding idiom words, and each lighter shades correspond to narrower contexts.

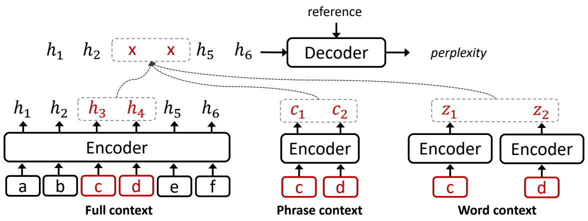


Figure 20: Illustration of how we measure the effects of different idiom contexts on translation likelihood (perplexity). First, we obtain each sequence of idiom token representations, by encoding them within different contexts. Then we feed each sentence to the encoder, and before passing its outputs to the decoder, we replace the idiom token representations with those whose context we want to probe. We use the sequence of encoder outputs, to score the reference translation.

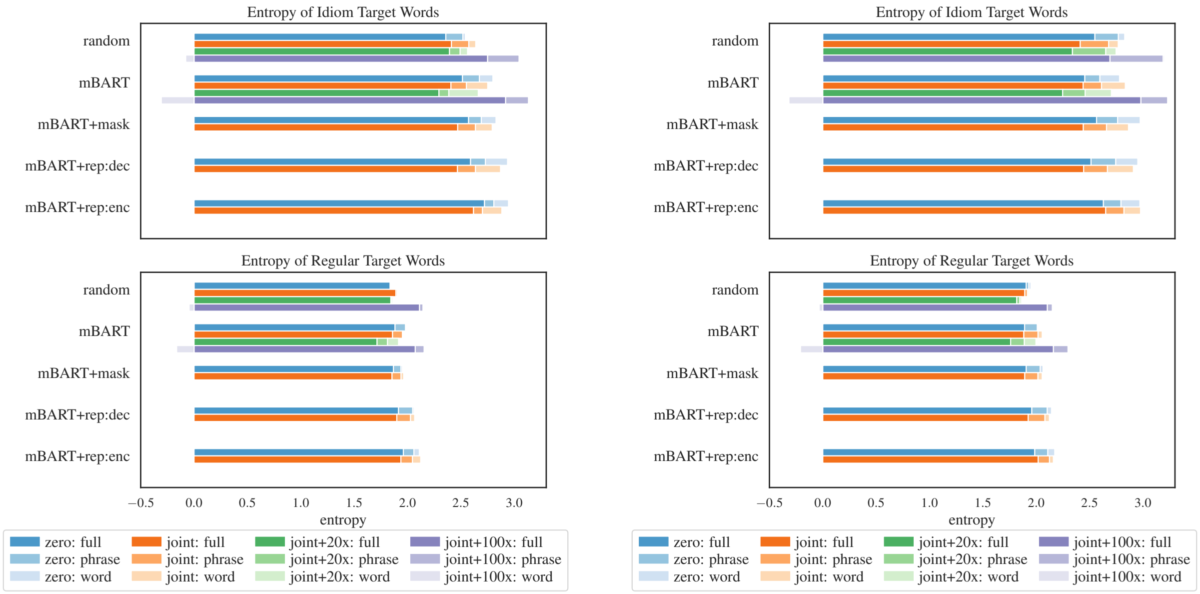
when we encode idioms with word context.

D.2 Variation in Translation Likelihood

With this probe, we test how the variation of the idiom encoder representations affects the likelihood of the reference translations. Figure 20, shows a visual example of how this probe works. Specifically, we translate a sentence pair with teacher-forcing, but we replace the encoder idiom token representations, before passing them to the decoder, with the representations obtained after encoding them with different contexts. Finally, we measure the perplexity of the reference translation under the model. Figure 19, compares models across splits and contexts.

Overall, the results are very consistent across both language pairs and all model variants. We also observe that the behaviour of all the noisy finetuned mBART models very similar behaviour, across all contexts. However, there is a small increase in the perplexity assigned under the "mBART+replace(dec)" variant. This is expected, as this model was trained with decoder dropout, which affect the LM capabilities of the model, and consequently makes it assign an overall smaller probability to all sentences. This increase in perplexity is observed in both language pairs, but is more pronounce in en→fr.

The more we upsample the idiom-train sentence pairs, the less probable other sentences become under the model. Surprisingly, 100x upsampling causes the pretrained model to yield lower perplexity with narrower context (i.e., lighter bars have negative length), "reverting" some of the effects of overfitting. However, we don't have a satisfying explanation for this behaviour[11]. This phenomenon is observed in both language pairs.



(a) Results for the en→fr models.

(b) Results for the en→es models.

Figure 21: Comparison of model uncertainty during the translation of regular-vs-idiom words. The length of each (lighter) bar encodes the difference from its (darker) bar to its left. Negative bar lengths indicate a decrease relative to the (darker) bar before it.

D.3 Decoder Uncertainty

Next, we focus on how the token-level uncertainty of the decoder varies while it translates idiom vs. non-idiom words (Figure 8). For each model, first, we translate each sentence pair with teacher-forcing[12]and then measure the entropy of the decoder's distributions for each target token.

Once more, this probe reveals that the models in both language pairs behave similarly. As mentioned in the main paper, the distributions of words that translation the idiom phrase have significantly larger entropy that the rest. This demonstrates that the models clearly are much more uncertain when translating idioms, even the pretrained ones.

Including and upsampling 20x the idiom-train data is helpful, but extreme upsampling (i.e., 100x) is universally harmful, although we observe a drop in uncertainty with word-level context. This is more pronounced when translating idiom words and suggests that models have overfitted to the words of the idiom phrases.

Representation biases in sentence transformers

Dmitry Nikolaev **Sebastian Padó**
IMS, University of Stuttgart
`dnikolaev@fastmail.com` `pado@ims.uni-stuttgart.de`

Abstract

Variants of the BERT architecture specialised for producing full-sentence representations often achieve better performance on downstream tasks than sentence embeddings extracted from vanilla BERT. However, there is still little understanding of what properties of inputs determine the properties of such representations. In this study, we construct several sets of sentences with pre-defined lexical and syntactic structures and show that SOTA sentence transformers have a strong nominal-participant-set bias: cosine similarities between pairs of sentences are more strongly determined by the overlap in the set of their noun participants than by having the same predicates, lengthy nominal modifiers, or adjuncts. At the same time, the precise syntactic-thematic functions of the participants are largely irrelevant.

1 Introduction

Transformer-based encoder-only models derived from the BERT architecture and pre-trained using similar objective and training regimens (Devlin et al., 2019; Liu et al., 2019) have become the standard tool for downstream tasks at the level of individual tokens and token sequences (Tenney et al., 2019; Wang et al., 2021). Whole-sentence representations can also be easily extracted from the outputs of these models by either using the embedding of the special [CLS] token, in cases where the model was trained on the next-sentence-prediction task, or averaging or max-pooling the embeddings of all tokens produced by the model (Zhelezniak et al., 2019). While both approaches are widely used in practice, it has been argued that these representations are not well suited for sentence-level downstream tasks. Several modifications to the architecture and training regime were proposed, which are known collectively as sentence transformers (STs; Reimers and Gurevych, 2019).

STs have achieved state-of-the-art performance on downstream tasks such as semantic search and question answering (Santander-Cruz et al., 2022; Ha et al., 2021). Their analysis, however, has received considerably less attention than the analysis of the vanilla BERT model and its variants (Rogers et al., 2020; Conia and Navigli, 2022). In fact, these models are often considered to be uninterpretable (Minaee et al., 2021).

A common feature of STs is that they are fine-tuned to produce similar vector-space representations for semantically similar sentences. This objective induces a complex loss landscape shaped by the available training data. The original Sentence-BERT model (Reimers and Gurevych, 2019) was trained on natural language inference data, and sentences were considered to be semantically similar if their NLI label was that of entailment. SOTA models were trained on a much larger web-crawled corpus including more than 1 billion sentence pairs mined from sources such as Reddit conversations, duplicate question pairs from WikiAnswers, etc.[1] The richness and variability of this dataset begs the question of what notion of semantic similarity is implicitly learned by the models trained on it.

In this study, we begin addressing this question through analysis of natural-looking synthetic sentences with controlled syntactic and lexical content. We concentrate on three questions.

First, we test if STs have part-of-speech biases. We show that, all other things being equal, information provided by nouns plays more important role than the information provided by verbs, both in simple sentences and in sentences with coordinated verbal phrases.

Second, we compare the relative importance of the overlap in the sets of participants in two sentences with that of how many participants have identical syntactic functions. We show that raw lexical overlap is relatively more important than having the same nouns in the same syntactic slots.

---

[1] See the list at `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3701–3716
May 2-6, 2023 ©2023 Association for Computational Linguistics

Third, we check how strongly sentence representations are affected by other sentential elements, such as adverbials and nominal modifiers of different types and lengths. We show that, unlike BERT with token averaging, STs seem to largely disregard these components in favor of nominal participants.

The paper is structured as follows: § 2 presents the methodology that we follow in our analyses and the models we employ; § 3 presents the case studies and their results; § 4 provides an overall discussion; § 5 surveys related work; § 6 concludes the paper.

2 Methods and Experimental Setup

We experiment with representations produced by three models. Two are SOTA STs: `all-mpnet-base-v2` (MPNET) is an instance of `mpnet-base` (Song et al., 2020) fine-tuned on the 1B sentence-pair corpus using the training architecture from Reimers and Gurevych (2019); `all-distilroberta-v1` (DistilRoberta) is a distilled instance of `roberta-base` (Sanh et al., 2019) fine-tuned in the same way. The third model is the vanilla pre-trained `bert-large-uncased` (BERT), as a point of comparison for the first two.

All models were downloaded from HuggingFace. Standard APIs from the Sentence Transformers library[2] were used to compute embeddings using MPNET and DistilRoberta; for the vanilla BERT model, we averaged the embeddings of all sentence tokens, including [CLS] and [SEP].[3]

We structure the presentation as a series of case studies. For each case study, we construct a set of sentences controlled for lexical content and syntactic structure. Sentences are created in such a way as to be grammatically correct, look naturalistic, and as far as possible not bias the analysis.[4] They are arguably less complex and variable than examples sampled from real-word corpora; however, we believe that an analysis based on simple sentences is a reasonable first step towards a better understanding of model representations, as previous work has shown for sentiment analysis (Kiritchenko and Mohammad, 2018) and syntactic analysis (Marvin and Linzen, 2018).

For each case study, we compute embeddings for all sentences, together with cosine similarities between embeddings of sentence pairs. We analyze the similarities by means of regression modelling. More precisely, we regress cosine similarities, z-scored to improve comparability between encoders, on the properties of sentence pairs, such as lexical overlap, presence of identical participants in identical syntactic positions, or POS tags of participants. We inspect the coefficients of the resulting regression fits to assess the relative importance of these properties. Since (almost) all properties are coded as binary variables, their magnitudes are directly comparable in terms of importance.

For terminological clarity, we will use the term *models* to refer to the regression models we use to analyse the impact of sentence properties on representational similarity. We call the transformers computing these embeddings *encoders*.

Where the features of sentence pairs can be straightforwardly related to simple properties of individual sentences (e.g., in case when we are testing if they have the same subject or direct object), we also project sentence embeddings on a 2-D surface using UMAP (McInnes et al., 2018)[5] and check if the spatial organisation of the points is in line with our observations.

Lexical choice A potential confound of our experimental setup is lexical choice, which is never completely neutral. For example, by taking a semantically close pair of verbs, we can considerably reduce the effect of predicate mismatch between two sentences. Moreover, encoders can react idiosyncratically to particular words and word combinations. Including all combinations of words and their positions in sentence pairs as predictor variables is not a solution, however, as it defeats the purpose of identifying structural patterns and, in the limit, amounts to replicating the encoders. We address this confound in three ways.

First, we select nouns to be always at least as interchangeable as words of other parts of speech in terms of belonging to similar mid-to-high frequency bands and referring to conceptually simple, concrete objects. This follows from our working hypothesis that encoders give preferential treatment

---

[2] `https://www.sbert.net/index.html`, Reimers and Gurevych (2019).

[3] We experimented with omitting the special tokens, but this led to sentence representations dominated by punctuation signs and other undesired effects. In line with previous work (Ma et al., 2019), we also found that using [CLS] embeddings leads to bad results due to their high redundancy, and we do not discuss them.

[4] Sentence-generating and model-fitting scripts can be found in the Supplementary Materials.

[5] We use the default settings and pairwise cosine dissimilarities as distance measure.

to nominal elements, whose (generally entity referring) semantics is arguably easier to capture than, for example, that of (generally event referring) verbs (Baroni and Lenci, 2011).

Second, we compare the analysis of the ST encoders against the analysis of the vanilla BERT encoder. As they are derived from averaging, vanilla BERT embeddings treat all words equally, so if our sentences, e.g., undersell differences in adverbs because we chose two nearly synonymous ones, this should be visible in the small coefficient tracking the impact of adverbs in the regression model based on BERT embeddings. As will be shown below, however, the hierarchy of coefficients for regression models of STs is very different from that for vanilla BERT, which arguably indicates that the role of lexical effects is minor.

Third, we re-run all reported models on sentences of the same structure with different lexical content; see the Appendix for details. We observe high stability of coefficients across replications, higher for STs than for vanilla BERT. This further corroborates the validity of our generalisations.

3 Case Studies

This section presents a series of case studies testing the sensitivity of embeddings produced by sentence transformers and BERT token averages to properties of input sentences. We start with analysing simple intransitive sentences (§ 3.1) and simple transitive sentences (§ 3.2). We then make specific aspects of the structure more complex, analysing the effect of lengthy NPs (§ 3.3) and coordinated VPs (§ 3.4). Finally, we look more closely at the syntax-semantics interface by inverting the prototypical alignment of POS tags and syntactic functions (predicative nominals and gerund subjects, § 3.5) and by testing the degree to which encoders track particular syntactic functions of verb arguments (§ 3.6).

3.1 Simple Intransitive Sentences

Data The main goal of the analysis of simple intransitive sentences is to check the relative contribution of their components to their embeddings. We study a nearly-minimal sentence template with a nominal subject, an adverbial adjunct, and an intransitive verb. We construct a set of 256 sentences of the form '[det] [subj] [adverb] [verb][punct]', where `det` ranges over {*a*, *the*}; `subj` ranges over a set of nouns,[6] `adverb` ranges over {*quickly*, *slowly*}, `verb` ranges over {*appears*, *vanishes*, *stops*, *moves*}, and `punct`, over {., *!*}. Here and in subsequent experiments, the generation procedure assures that all sentence features are statistically independent, which is a crucial prerequisite for linear-regression modelling.

	mpnet	**distilroberta**	**bert**
SameDet	0.07	0.07	0.37
SameAdv	0.33	0.31	0.45
SamePred	0.74	0.61	0.58
SamePunct	0.24	0.24	0.84
SameSubj	2.26	2.40	1.27
R-squared	0.67	0.71	0.48

Table 1: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with intransitive verbs. All coefficients are significant with $p < 0.001$.

Model The regression model matrix is based on 32,640 pairs of generated sentences, which differ in the value of at least one feature, with predictor variables SameDeterminer, SameAdverb, SameVerb, SamePunct, and SameSubj. We regress z-score-transformed cosine similarities between sentence embeddings computed by three different encoders on these predictor variables. The coefficients of the fitted models are shown in Table 1.[7]

Results Three observations from Table 1 hold for all subsequent analyses.

(i) The coefficients are positive for all models and all features. This means that sentence pairs which agree in some constituent are always more similar than sentence pairs that do not – as expected.

(ii) The coefficient of determination (R^2) is larger for ST-focused linear models. This means that the embeddings computed by the ST encoders are more dependent on the features of the sentences we track and less dependent on identities of lexical units. (It can be noted that the fact that we achieve $R^2 \approx 0.7$ using only a few structural properties is remarkable in itself.)

(iii) The differences among coefficients of the ST-focused linear models are in general larger than

---

[6] {*cat*, *dog*, *artist*, *teacher*, *planet*, *star*, *wind*, *rain*}

[7] Replication models, fitted on sentences with the same structure but different lexical content, are shown in Table 8 in the Appendix.

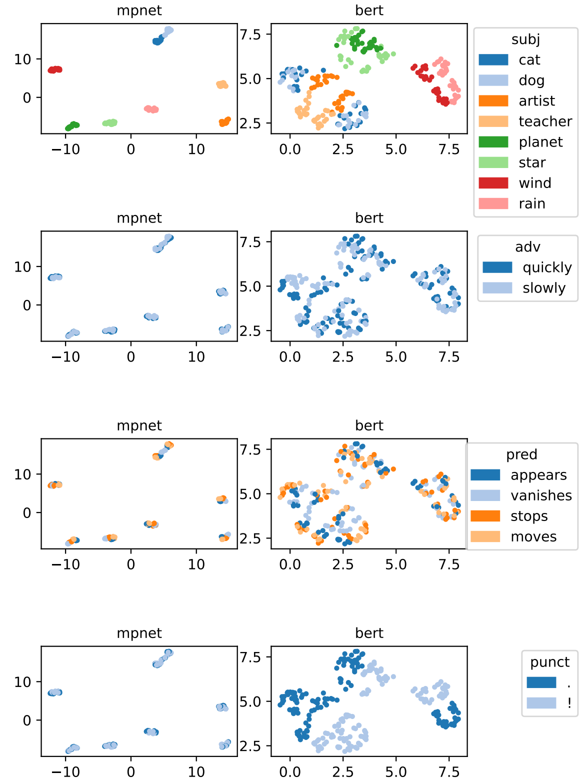


Figure 1: UMAP projections of embeddings of sentences with intransitive verbs (left: sentence transformer, right: BERT).

those of the linear model analysing BERT: in the latter, the biggest coefficient (1.27 for SameSubj) is only ≈ 3.5 times higher than the smallest one (0.37 for SameDet), while for the ST models this ratio is above 30. This is connected to the fact that BERT-derived sentence representations are more dependent on semantically impoverished elements, such as determiners and punctuation signs, which dampen the effect of other constituents. For the sake of brevity, we do not analyse determiners and punctuation in subsequent experiments and keep them constant as *the* and . respectively.

Turning to the comparison of coefficients inside models, we see that STs pay considerably more attention to subjects than to predicates: all things being equal, sentences with different predicates and adverbs but the same subject will be more similar than sentences with the same predicate and adverb and different subjects. The influence of punctuation is surprisingly strong, being comparable to that of adverbs, while the effect of determiners is very weak, albeit statistically significant.

A plot of UMAP projections of sentence embeddings produced by MPNET and BERT, shown in Figure 1, underlines that while averaged BERT embeddings distinguish punctuation signs but do not distinguish subjects, the situation is reversed for the sentence transformer: it distinguishes subjects cleanly but largely abstracts away from other structural properties.

3.2 Transitive Sentences

Data The transitive sentences used in the analysis are generated using the following template: 'The [subj] [adverb] [verb] the [obj].' The range of nouns was slightly extended;[8] the same adverbs as in the previous experiment were used, while `verb` ranged over {*sees*, *chases*, *draws*, *meets*, *remembers*, *pokes*}. This produces 672 different sentences and 225,456 sentence pairs.

Model The coding for SameAdv and SamePred remains as above. The main focus in this study is on whether sentence similarities are dominated by the sentences having the same subject, the same direct object, or the same words in these two positions even if their order were reversed. To test for this, we added a categorical variable with the following values:

00 no overlap in subject and object (the baseline);
A0 same subject, different objects;
0B same object, different subjects;
0A the subject of the first sentence is the object of the second;
B0 the object of the first sentence is the subject of the second;
BA subject and object are swapped;
AB the same subject and object.

Results A summary of the fitted models is given in Table 2.[9] It demonstrates that when it comes to simple transitive sentences, our understanding of their embeddings produced by sentence transformers remains high, despite the sentences being more complex ($R^2 \approx 0.7$), while BERT embeddings become more unpredictable ($R^2 \approx 0.31$). Furthermore, while BERT again essentially treats all tokens more or less equally, with adverbs slightly discounted, STs prioritise participants (even B0 has higher coefficients than SamePred).

On the other hand, neither BERT nor STs prioritise the exact syntactic function of the participants: coefficients for A0 vs. 0A, 0B vs. B0, and AB vs.

---

[8] To {*cat*, *dog*, *teacher*, *artist*, *robot*, *machine*, *tree*, *bush*, *planet*, *star*, *wind*, *rain*}.

[9] A summary of the replication model fits is provided in Table 9 in the Appendix.

	mpnet	distilroberta	bert
SameAdv	0.49	0.36	0.56
SamePred	0.73	0.42	0.78
SubjObj_0A	1.27	1.40	0.65
SubjObj_0B	1.31	1.45	0.69
SubjObj_A0	1.44	1.45	0.75
SubjObj_AB	2.98	3.08	1.60
SubjObj_B0	1.37	1.42	0.58
SubjObj_BA	2.85	2.98	1.39
R-squared	0.74	0.73	0.31

Table 2: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with transitive verbs. All coefficients are significant with $p < 0.001$.

BA are largely comparable across all models with $BA \approx A0 + 0B$. That is, the effects of subjects and objects are largely independent of one another.

A UMAP plot with the embeddings for the transitive sentences is shown in Figure 2 in the Appendix. It demonstrates that STs arrive at a much more fine-grained clustering of sentences, largely dominated by subjects and objects. They largely discount predicates and adverbs which are quite prominent in averaged BERT embeddings.

3.3 Transitive Sentences with Long NP Modifiers

The previous analyses showed that representations computed by STs are highly attuned to verb participants but not to their particular syntactic roles. This may mean that ST may be potentially misled by nouns in other positions in the sentence, which have less relevance to the described situation. This study explores this possibility.

Data We repeat the analysis from § 3.2 using the template of the form 'The [subj] [modifier] [adverb] [verb] the [obj]', with a smaller set of subjects,[10] and the `modifier` ranging over {*with big shiny eyes*, *that my brother saw yesterday*, *whose photo was in the papers*, *worth a great deal of money*}. Altogether this gives 1,440 sentences and 1,036,080 sentence pairs. The modifiers have internal syntactic structure and contain a non-negligible amount of lexical material that the models have to 'skip over' if their representations were focused on the participant structure of the matrix clause.

	mpnet	distilroberta	bert
SameMod	1.01	1.02	**1.62**
SameAdv	0.40	0.42	0.27
SamePred	0.89	0.67	0.40
SubjObj_0A	0.83	1.06	0.32
SubjObj_0B	0.97	1.27	0.42
SubjObj_A0	1.11	1.14	0.53
SubjObj_AB	**2.14**	**2.44**	1.00
SubjObj_B0	1.20	1.30	0.54
SubjObj_BA	**2.09**	**2.40**	0.91
R-squared	0.73	0.81	0.61

Table 3: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with transitive verbs and lengthy subject modifiers. All coefficients are significant with $p < 0.001$.

Model The same coding strategy as in the preceding section is used, augmented by a new binary variable, SameMod, tracking whether two sentences have the same modifier for the subject.

Results Both the model coefficients, shown in Table 3, and the UMAP plot, shown in Figure 3 in the Appendix, indicate that BERT embeddings are highly sensitive to lengthy modifiers:[11] the SameMod coefficient in the linear model is larger than the coefficients for the same predicate and the same subject-object combination added together. The situation is very different for STs: SameMod is more important than SamePred, especially for DistilRoberta, but, with one exception, not more important than even a partial overlap in participants. Having the same participants, in either the same or swapped syntactic functions, is more than twice as important. We take this as evidence that STs have a specific bias towards matrix-clause *participant sets*, that is, the nouns that fill a thematic role of the main predicate, while their precise functions and nouns found in other positions in the sentence are less important.

3.4 Coordinated Verbal Phrases

The analyses presented above show that the main predicate of the sentence has only a limited influence on the representations computed by STs, compared to its subjects and objects. Here, we show that this effect still holds if there is more than one main predicate.

---

[10] {*cat*, *dog*, *rat*, *giraffe*, *wombat*, *hippo*}

[11] The results of the replication fits are shown in Table 10 in the Appendix.

	mpnet	distilroberta	bert
V1Same	0.41	0.26	0.21
V2Same	0.13	0.08	0.23
V3Same	0.36	0.34	**0.41**
N1Same	0.33	0.35	0.23
N2Same	0.12	0.22	0.30
N3Same	**0.56**	**0.57**	**0.41**
R-squared	0.11	0.1	0.09

Table 4: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with coordinated VPs from binary predictors. All coefficients are significant with $p < 0.001$.

Data Using the same sets of nouns and transitive verbs as in the previous experiment, we construct sentences of the form 'The man [verb1] the [noun1], [verb2] the [noun2], and [verb3] the [noun3]', where triples of verbs and nouns are taken from the Cartesian product of the sets of all noun and verb combinations of size 3 without replacement. To alleviate a possible ordering bias, all verb and noun triples are shuffled for each sentence. This results in 400 sentences and 79,800 sentence pairs.

Models and results The analysis proceeds in three stages. First, we check if positions 1, 2, and 3 have different importance by regressing the normalised cosine similarity on six binary variables N[oun]1Same, V[erb]1Same, N2Same, etc. The models, summarised in Table 4,[12] show low coefficients of determination (with R^2 around 0.1), but they indicate that positions are of unequal importance: BERT gives more weight to the last noun and the last verb, while STs focus on the first and the last N-V pair and largely ignore the second one.

A significantly better fit can be achieved by replacing binary predictors with overlap scores for nouns and verbs. As Table 5[13] shows, this type of model, even though it contains only 2 variables instead of 6, obtains $R^2 \approx 0.65$ for STs. It is also evident that all three models place more weight on noun overlap than on verb overlap, with DistilRoberta showing the biggest difference between the two.

	mpnet	distilroberta	bert
VerbOverlap	0.78	0.59	0.64
NounOverlap	0.93	1.09	0.88
R-squared	0.65	0.68	0.52

Table 5: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with coordinated VPs from overlap scores. All coefficients are significant with $p < 0.001$.

This raises the question of whether particular verb-noun collocations play a noticeable role, i.e., if a sentence containing *chases the wombat* will be considerably more similar to another sentence containing the exact phrase compared to a sentence containing *chases* and *wombat* but not as a trigram. Simply adding n-gram overlap scores to the model is not possible, however, because it is highly correlated with both noun overlap and verb overlap. In order to obviate this obstacle, we first construct an auxiliary linear model predicting trigram overlap from noun and verb overlap and then use the residuals of this regression in the main model.

The results are ambiguous: on one hand, the coefficient for residualised trigram overlap is statistically significant with $p < 0.001$. On the other hand, the effect is very weak (more than ten times weaker than that of either noun overlap or verb overlap), and the addition of trigram overlap to the model improves R^2 by less than 0.001. This seems to indicate that trigram overlap is not important for practical purposes.

3.5 Predicative Nominals with Gerund Subjects

A potential weak point of our analysis is that parts of speech and syntactic functions are not decoupled: it is not yet clear whether the encoders pay attention to nouns or to subjects and objects.

Data To address this issue, we construct another set of sentences where the subject is a gerund and the predicate is nominal. The template is '[gerund] [object] [copula] a [adjective] [predicate]', where `gerund` ranges over {*continuing*, *abandoning*, *starting*, *completing*}, object ranges over {*it*, *them*, *the project*, *the plan*}, copula is one of {*is*, *was*, *will be*, *is going to be*}, adjectives are {*big*, *real*, *negligible*, *insignificant*}, and the predicative nominal ranges over {*solution*, *mistake*, *failure*, *triumph*}. This gives 1024 sentences and 523,776 sentence pairs. A variable copula provides an additional test as to whether the sentence encoders can recognise

---

[12] A summary of the replication fits is given in Table 11 in the Appendix.

[13] See Table 12 in the Appendix for the replication fits.

	mpnet	distilroberta	bert
SameSubj	0.82	0.70	0.31
SameCop	0.35	0.30	0.55
SameAdj	0.58	0.79	0.50
SamePred	0.99	1.01	0.52
SameObjNoun	1.01	1.04	0.60
SameObjPron	0.44	0.50	0.42
R-squared	0.50	0.54	0.22

Table 6: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with gerund subjects and nominal predicates. All coefficients are significant with $p < 0.001$.

multi-word sequences with low semantic content.

Model The sentence pair encoding includes four binary variables (SameSubj, SameCop, SameAdj, SamePred) and a nominal variable for the direct object, indicating whether objects are different (baseline), are identical and pronominal (SamePron), or are identical and nominal (SameNoun).

Results The results in Table 6[14] demonstrate that all models treat both nominal predicates and nominal direct objects as more important than gerund subjects. STs, moreover, pay less attention to identical pronominal objects and discount multi-word copula forms. R^2 values for the ST model are lower than in the previous experiments (in the 0.50–0.55 range), which may potentially indicate a poor choice of lexical items; however, replication experiments with a different set of words (except for copula forms) achieved comparable results. This suggests that embeddings of sentences of this type are less easily explainable as additive combinations of individual words compared to the sentence types surveyed previously.

3.6 Revisiting Participant Sets: Ditransitive Sentences

Our final experiment revisits the opposition between lexical overlap in verbal phrases and exact argument-predicate matching. In this case, we focus on ditransitive verbs with two arguments: a direct object and an oblique object which is an integral part of the situation.[15]

	mpnet	distilroberta	bert
SameAdv	1.05	1.07	0.64
SamePred	0.93	0.64	0.83
Overlap	0.90	1.00	0.91
SPCRes	0.03	0.02	0.10
R-squared	0.745	0.738	0.57
R-squared (w/o SPCRes)	0.744	0.737	0.56

Table 7: A summary of the models predicting z-scored pairwise cosine similarities between embeddings of sentences with ditransitive verbs. SPCRes stands for SamePosCountRes, i.e. the residuals of the number of identical words in identical positions regressed on lexical overlap. All coefficients are significant with $p < 0.001$.

Data All permutations of the triple of basic nouns {*cat*, *dog*, *rat*} are generated. For each permutation, all three nouns are, in turn, replaced with one of the members of the set of extra nouns {*giraffe*, *wombat*, *hippo*}; the original permutations are also used. This provides a set of unique triples of nouns where each pair of triples has from one to three nouns in common. The Cartesian product of this set of triples with a set of ditransitive verbs ({*describes*, *sells*, *shows*}) and a set of adverbs ({*happily*, *quickly*, *secretly*) is used to fill the template 'The [noun1] [adverb] [verb] the [noun2] to the [noun3].' This procedures gives 540 sentences and 145,530 sentence pairs.

Model The sentence pairs are coded for same adverb, same predicate, the number of matching nouns in matching positions (SamePosCount), and lexical overlap minus 1 (the baseline value of 0 corresponds to overlap of 1; each successive value corresponds to increase in overlap). As with overlapping words and trigrams above, these predictors are correlated. Therefore, we residualise SamePosCount after regressing it on lexical overlap.

Results Table 7 is inconclusive in a similar way to results from § 3.5. The coefficients for residualised SamePosCount are significant; however, in the ST models, their size is very small, and SamePosCount does not materially improve the predictive power. We conclude, therefore, that syntactic positions do not matter a great deal, in line with our 'participant set' interpretation from § 3.4.

---

[14] See an overview of replication fits in Table 13 in the Appendix.

[15] Many English ditransitive verbs can undergo the 'dative alternation', which swaps the oblique object with a prepositional phrase: *Give the book to me/John* vs. *Give me/John a book* (Levin, 1993). Of the verbs we use, *show* and *sell* participate in it, and the status of *describe* varies across speakers.

4 Discussion

Our analysis arguably goes some way towards explaining why sentence transformers beat vanilla BERT-based models with token averaging on sentence-modelling tasks. Token averaging makes it impossible to distinguish between semantically rich and impoverished sentence elements, nor between syntactically central vs. peripheral elements: punctuation signs and determiners contribute on the same level as the matrix-clause predicate and main participants, while lengthy modifiers, such as relative clauses, and multi-word copula forms dominate the representation.

Sentence transformers, on the other hand, learn to discount elements that only serve a grammatical function or present background information and focus instead on the semantic kernel of the sentence. The latter is in effect largely synonymous with the set of nominal elements in the main clause, first of all participants, but also predicative nominals. Importantly, despite their evident syntactic-analytic capabilities (e.g., in our setting they can distinguish between participants of main and relatives clauses and between main and auxiliary verbs), STs seem to not pay much attention to the distinction between subjects and direct or indirect objects. Instead they prioritise raw overlap in the set of nominal participants of the matrix clause. This can be seen, by slightly abusing terminology of theoretical linguistics, as a focus on the aboutness/topic of sentences, what things they describe, and not on their predication/comment, what they actually say about those things (Hu and Pan, 2009).

We believe that this focus is not inherent to the architecture of sentence transformers but reflects the nature of the datasets used for fine-tuning STs. The size of these datasets makes it impossible to convincingly reason about their contents, but their genres (QA pairs, Reddit threads, etc.) makes it plausible to expect a high degree of topic-based overlap: questions and conversations tend to revolve around entities (persons and things), with their actions and properties repeating less often. This naturally leads to a focus on nouns referring to prominent entities, which are known to appear preferentially as subjects or objects for reasons of coherence (Barzilay and Lapata, 2008), arguably a good match to the patterns we observe.

5 Related Work

Analysis of transformer-based models for sentence-level tasks, such as NLI, question answering, or text classification, has largely followed the same approaches as found in the general BERTology (Rogers et al., 2020): probing, analysis of the geometry of the embedding space, extraction of parts of input that are particularly important for model performance, and behavioural analysis. In this vein, Liu et al. (2021) and Peyrard et al. (2021) analyse the attention patterns powering the performance of transformer models on different types of sentence classification, and Li et al. (2020) show that embeddings of sentences computed by BERT-based models, including siamese-fine-tuned sentence transformers, are anisotropic and can be improved via normalisation. Chrysostomou and Aletras (2021) survey the existing methods for extracting rationales from input sentences in the context of text classification and propose an improved approach, while Luo et al. (2021) demonstrate that sentence embeddings derived by averaging BERT token representations suffer from artefacts arising from positional embeddings. Zhelezniak et al. (2019) argue that averaging should be replaced with max-pooling.

Very similar to ours is the approach adopted by MacAvaney et al. (2022), who construct a series of probes to analyse the performance of several models on the task of information retrieval. While their methodology relies on high-level document statistics and wholistic document manipulation (word and sentence shuffling, token-frequency similarity between the document and the query, textual fluency, etc.), our study analyses the role of linguistically motivated structural factors and thus complements their findings.

Opitz and Frank (2022) aim at directly decomposing the representations produced by sentence transformers into several parts capturing different properties of sentences reflected in AMR annotations (presence of negation, concepts included in the sentence, etc.). While our study tries to ascertain what meaning components dominate the representations, Opitz and Frank assume that these components are known in advance and are equally important: sentence embeddings in their modified SBERT model are split into 15 segments, each of which corresponds to one AMR-based meaning component, plus a residual part to capture everything not covered by AMR annotations.

6 Conclusion

This paper aims at making a contribution towards a better understanding of sentence transformers, which are often seen as black boxes. We have demonstrated that we can make surprisingly precise inferences about sentence-pair similarities using simple linguistic features such as lexical overlap.

The crucial difference between bag-of-words distributional models and current encoders is that STs have became quite adept at disregarding 'irrelevant' parts of the sentence and concentrating on its key elements. Unlike vanilla BERT sentence embeddings obtained by token averaging, STs yield more structured embeddings that focus on the matrix clause and are less tied to individual lexical items and strings of function words.

This progress, however, comes with a particular type of bias: the structures that lead to high sentence similarity in STs, i.e. the overlap in nominal 'participant sets', seem to mirror the dominant type of paraphrases found in the data the STs were tuned on, and STs are not compelled to look at finer structures of input sentences. At least without further fine tuning, this would appear to make them unsuitable for downstream tasks that require knowledge about more fine-grained aspects of sentence structure, such as semantic roles (Conia and Navigli, 2022), or extra-propositional aspects, such as monotonicity, negation, or modality (Yanaka et al., 2021; Nakov, 2016).

An interesting direction for future research would be to explore the ways of decomposing sentence representations into additive aspects such as participant structure, main predication, etc. The additional challenge here is that while theoretical semantics has a lot to say about aspects of sentence meaning (Pagin, 2016), there remains a lack of analysis linking the notion of one-dimensional *semantic similarity* (Agirre et al., 2012) that underlies the optimisation of current sentence transformers with theoretically more substantial concepts.

Limitations

The limitations of the proposed analysis are the following:

1. The analysis is based on synthetic data. This allows us to fully control the sentence structure and use balanced lexical material, but it does not necessarily reflect the performance of models on real-world data, especially when sentences or text fragments are much longer. However, synthetic data have generally shown to be a good first step toward understanding the behaviour of complex models.
2. The analysis does not cover graded distinctions between words, i.e. we did not experiment with filling the slots with synonymous words, as opposed to completely unrelated words. This makes it impossible to decide if the models are sensitive to word identities or to their actual semantics, as long as these two notions are distinguishable.
3. The outputs of the models are interpreted using linear regression analysis anchored to the properties of synthetic sentences. This kind of analysis makes it possible to disentangle additive effects of different components of sentence structure and provides statistical-significance estimates, while high R^2 values indicate that our findings have some validity. However, it cannot fully account for the lexical effects (which we tried to safeguard against by carefully selecting template fillers), non-linear effects, and hidden collinearity patterns (beyond those we addressed using residualised analysis).
4. The range of models analysed in the paper is restricted. It covers some amount of variability (sentence transformers vs. vanilla BERT; two different variants of a base model for STs, one of them distilled), but other combinations of model architecture and training/fine-tuning regime can lead to different outcomes.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10, Edinburgh, UK. Association for Computational Linguistics.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

George Chrysostomou and Nikolaos Aletras. 2021. Variable instance-level explainability for text classification. *arXiv preprint arXiv:2104.08219*.

Simone Conia and Roberto Navigli. 2022. Probing for predicate argument structures in pretrained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4622–4632, Dublin, Ireland. Association for Computational Linguistics.

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

Thi-Thanh Ha, Van-Nha Nguyen, Kiem-Hieu Nguyen, Kim-Anh Nguyen, and Quang-Khoat Than. 2021. Utilizing SBERT for finding similar questions in community question answering. In *2021 13th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–6. IEEE.

Jianhua Hu and Haihua Pan. 2009. Decomposing the aboutness condition for Chinese topic constructions. *The Linguistic Review*, 26:371–384.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago Press.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Shengzhong Liu, Franck Le, Supriyo Chakraborty, and Tarek Abdelzaher. 2021. On exploring attention-based explanation for transformer models in text classification. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 1193–1203. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ziyang Luo, Artur Kulmizev, and Xiaoxi Mao. 2021. Positional artefacts propagate through masked language model embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5312–5327, Online. Association for Computational Linguistics.

Xiaofei Ma, Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. 2019. Universal text representation from BERT: An empirical study. *arXiv preprint arXiv:1910.07973*.

Sean MacAvaney, Sergey Feldman, Nazli Goharian, Doug Downey, and Arman Cohan. 2022. ABNIRML: Analyzing the behavior of neural IR models. *Transactions of the Association for Computational Linguistics*, 10:224–239.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Leland McInnes, John Healy, and James Melville. 2018. UMAP: Uniform manifold approximation and projection for dimension reduction. *ArXiv*, abs/1802.03426.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning–based text classification: A comprehensive review. *ACM Computing Surveys (CSUR)*, 54(3):1–40.

Preslav Nakov. 2016. Negation and modality in machine translation. In *Proceedings of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics (ExProM)*, page 41, Osaka, Japan. The COLING 2016 Organizing Committee.

Juri Opitz and Anette Frank. 2022. SBERT studies meaning representations: Decomposing sentence embeddings into explainable semantic features. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 625–638, Online only. Association for Computational Linguistics.

Peter Pagin. 2016. Sentential semantics. In Maria Aloni and Paul Dekker, editors, *Cambridge Handbook of Formal Semantics*, Cambridge Handbooks in Language and Linguistics, pages 65–105. Cambridge University Press.

Maxime Peyrard, Beatriz Borges, Kristina Gligorić, and Robert West. 2021. Laughing heads: Can transformers detect what makes a sentence funny? *arXiv preprint arXiv:2105.09142*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Yamanki Santander-Cruz, Sebastián Salazar-Colores, Wilfrido Jacobo Paredes-García, Humberto Guendulain-Arenas, and Saúl Tovar-Arriaga. 2022. Semantic feature extraction using SBERT for dementia detection. *Brain Sciences*, 12(2):270.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. In *Proceedings of NeurIPS*, pages 16857–16867.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. *CoRR*, abs/2105.03654.

Hitomi Yanaka, Koji Mineshima, and Kentaro Inui. 2021. SyGNS: A systematic generalization testbed based on natural language semantics. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 103–119, Online. Association for Computational Linguistics.

Vitalii Zhelezniak, Aleksandar Savkov, April Shen, Francesco Moramarco, Jack Flann, and Nils Y. Hammerla. 2019. Don't settle for average, go for the max: Fuzzy sets and max-pooled word vectors. *CoRR*, abs/1904.13264.

	mpnet	distilroberta	bert
SameDet	0.08	0.11	0.26
SameAdv	0.38	0.38	0.96
SamePred	1.02	0.95	0.49
SamePunct	0.18	0.26	0.64
SameSubj	2.15	2.17	0.65
R-squared	0.71	0.71	0.43

Table 8: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with intransitive verbs. All coefficients are significant with $p < 0.001$.

A Appendix

A.1 Dimensionality-reduction plots

A.1.1 Simple transitive sentences

A UMAP plot of embeddings of simple transitive sentences encoded accordings to their properties is shown in Figure 2.

A.1.2 Transitive sentences with long NP modifiers

A UMAP plot of embeddings of transitive sentences with lengthy subject modifiers encoded accordings to their properties is shown in Figure 3.

A.2 Replication-model fits

A.2.1 Simple intransitive sentences

The following lexical items were used for the replication experiment:

- Nouns: *wolf*, *bear*, *fruit*, *vegetable*, *building*, *car*, *lightning*, *wave*
- Verbs: *stabilizes*, *bursts*, *grows*, *shrinks*
- Adverbs: *suddenly*, *predictably*

A summary of the replication models is shown in Table 8.

A.2.2 Simple transitive sentences

The following lexical items were used for the replication experiment:

- Nouns: *pig*, *horse*, *soldier*, *farmer*, *android*, *computer*, *grass*, *forest*, *comet*, *galaxy*, *cloud*, *lightning*
- Verbs: *hears*, *pursues*, *imagines*, *recognizes*, *touches*, *finds*
- Adverbs: *suddenly*, *predictably*

	mpnet	distilroberta	bert
SameAdv	0.54	0.32	0.95
SamePred	0.49	0.43	0.75
SubjObj_0A	1.46	1.50	0.70
SubjObj_0B	1.49	1.53	0.66
SubjObj_A0	1.48	1.54	0.76
SubjObj_AB	3.19	3.23	1.56
SubjObj_B0	1.40	1.48	0.50
SubjObj_BA	3.07	3.14	1.34
R-squared	0.81	0.8	0.45

Table 9: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with intransitive verbs. All coefficients are significant with $p < 0.001$.

A summary of the replication models is shown in Table 9.

A.2.3 Transitive sentences with long NP modifiers

The following lexical and phrasal items were used for the replication experiment:

- Nouns: *horse*, *pig*, *donkey*, *elephant*, *bison*, *moose*
- NP modifiers: *missing a hind leg*, *whose face we all know*, *born under a bad sign*, *pictured on page seventeen*
- Verbs: *hears*, *pursues*, *imagines*, *recognizes*, *touches*, *finds*
- Adverbs: *suddenly*, *predictably*

The overview of the model fits is shown in Table 10.

A.2.4 Coordinated verbal phrases

The following lexical items were used for the replication experiment:

- Nouns: *mouse*, *horse*, *fox*, *kangaroo*, *bison*, *elephant*
- Verbs: *hears*, *pursues*, *imagines*, *recognizes*, *touches*, *finds*

A summary of the replication models is shown in Tables 11 (individual-word-based models) and 12 (overlap-based models).

	mpnet	distilroberta	bert
SameMod	1.18	1.26	**1.83**
SameAdv	0.48	0.26	0.41
SamePred	0.64	0.64	0.44
SubjObj_0A	0.91	1.00	0.18
SubjObj_0B	0.99	1.09	0.17
SubjObj_A0	1.10	1.19	0.24
SubjObj_AB	**2.13**	**2.32**	0.42
SubjObj_B0	1.16	1.25	0.20
SubjObj_BA	**2.11**	**2.28**	0.39
R-squared	0.77	0.84	0.71

Table 10: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with transitive verbs and lengthy subject modifiers. All coefficients are significant with $p < 0.001$.

	mpnet	distilroberta	bert
V1Same	0.29	0.18	0.35
V2Same	0.13	0.08	0.28
V3Same	0.39	0.40	**0.42**
N1Same	0.49	0.48	0.14
N2Same	0.10	0.25	0.18
N3Same	**0.57**	**0.52**	0.17
R-squared	0.12	0.11	0.07

Table 11: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with coordinated VPs from binary predictors. All coefficients are significant with $p < 0.001$.

A.2.5 Predicative nominals with gerund subjects

The following lexical items were used for the replication experiment:

- Gerund subjects: *proposing*, *rejecting*, *praising*, *criticizing*
- Pronomial and nominal objects: *him*, *me*, *the idea*, *the design*
- Copula forms (same as in the original experiment): *is*, *was*, *will be*, *is going to be*
- Nominal predicates: *decision*, *defeat*, *loss*, *improvement*

A summary of the replication models is shown in Tables 13.

	mpnet	distilroberta	bert
VerbOverlap	0.69	0.52	0.85
NounOverlap	1.05	1.20	0.47
R-squared	0.69	0.76	0.41

Table 12: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with coordinated VPs from overlap scores. All coefficients are significant with $p < 0.001$.

	mpnet	distilroberta	bert
SameSubj	0.82	0.70	0.31
SameCop	0.35	0.30	**0.55**
SameAdj	0.58	0.79	0.50
SamePred	0.99	1.01	0.52
SameObjNoun	1.01	1.04	0.60
SameObjPron	0.44	0.50	0.42
R-squared	0.50	0.54	0.22

Table 13: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with gerund subjects and nominal predicates. All coefficients are significant with $p < 0.001$.

A.2.6 Participant-set overlap vs. identical participants

The following lexical items were used for the replication experiment:

- Basic nouns: *horse*, *pig*, *donkey*
- Extra nouns: *elephant*, *bison*, *moose*
- Verbs: *gives*, *demonstrates*, *entrusts*
- Adverbs: *suddenly*, *predictably*, *openly*

A summary of the replication models is shown in Tables 14.

	mpnet	**distilroberta**	**bert**
SameAdv	1.05	1.07	0.64
SamePred	0.93	0.64	0.83
Overlap	0.90	1.00	0.91
SPCRes	0.03	0.02	0.10
R-squared	0.745	0.738	0.57
R-squared (w/o SPCRes)	0.744	0.737	0.56

Table 14: A summary of the replication models predicting z-scored pairwise cosine similarities between embeddings of sentences with ditransitive verbs. SPCRes stands for SamePosCountRes, i.e. the residuals of the number of identical words in identical positions regressed on lexical overlap. All coefficients are significant with $p < 0.001$.

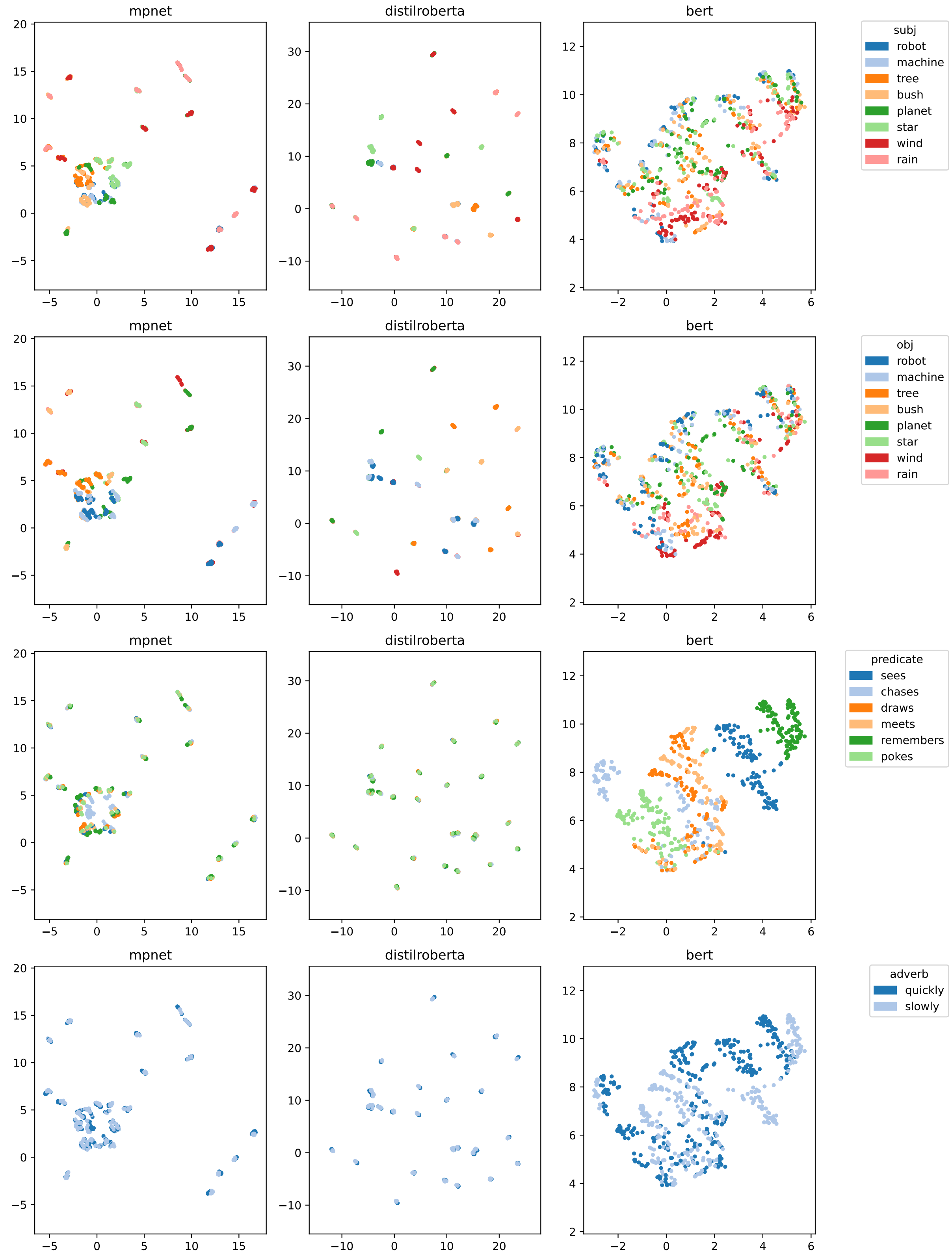


Figure 2: UMAP projections of embeddings of sentences with transitive verbs colour coded according to subject, object, predicate, and adverb.

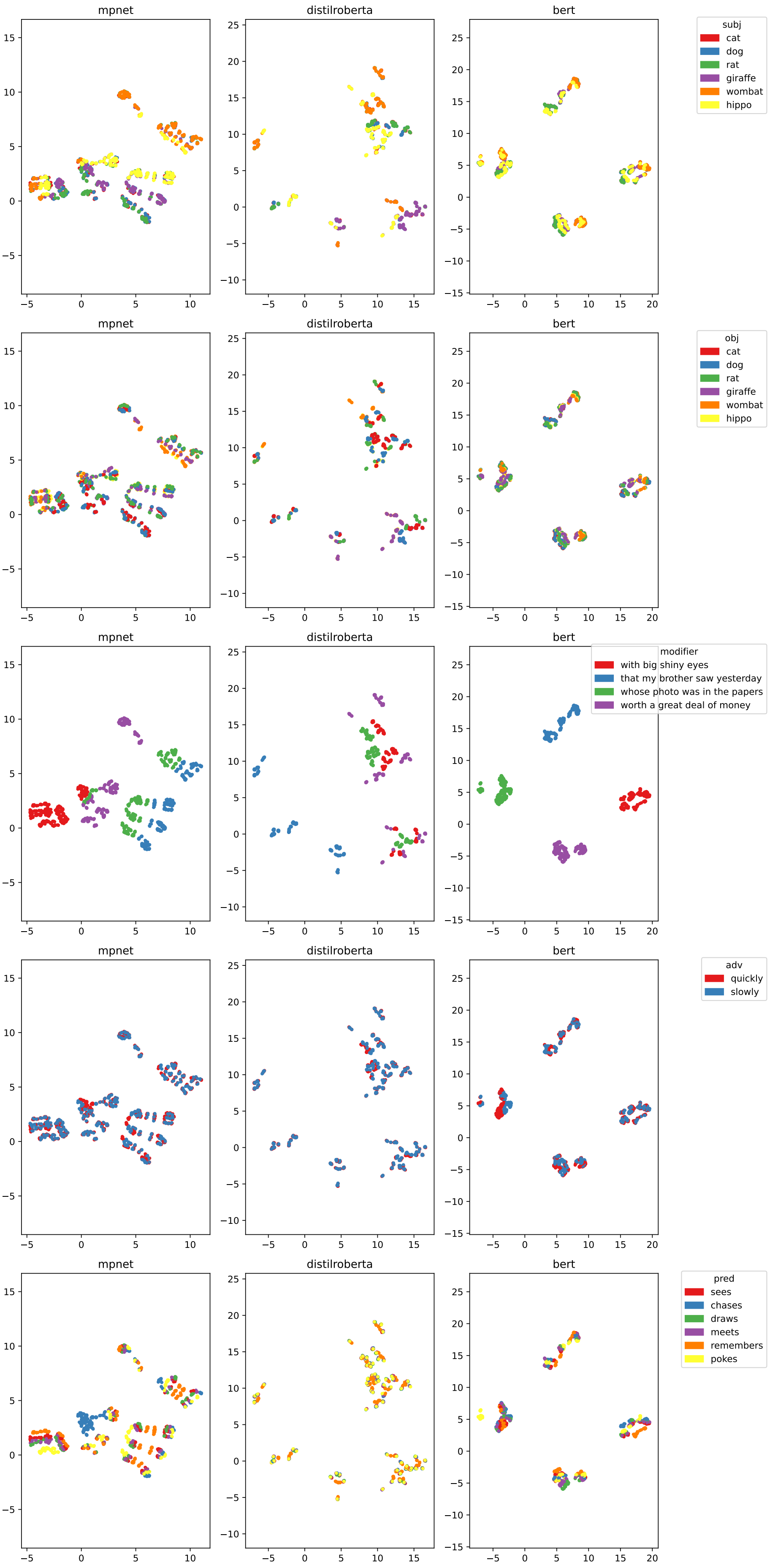


Figure 3: UMAP projections of embeddings of sentences with transitive verbs and long subject modifiers colour coded according to subject, modifier, object, predicate, and adverb.

AbLit: A Resource for Analyzing and Generating Abridged Versions of English Literature

Melissa Roemmele **Kyle Shaffer** **Katrina Olsen** **Yiyi Wang** **Steve DeNeefe**
Language Weaver (RWS Group)
{mroemmele,kshaffer,kolsen,yiyiwang,sdeneefe}@rws.com

Abstract

Creating an abridged version of a text involves shortening it while maintaining its linguistic qualities. In this paper, we examine this task from an NLP perspective for the first time. We present a new resource, AbLit, which is derived from abridged versions of English literature books. The dataset captures passage-level alignments between the original and abridged texts. We characterize the linguistic relations of these alignments, and create automated models to predict these relations as well as to generate abridgements for new texts. Our findings establish abridgement as a challenging task, motivating future resources and research. The dataset is available at github.com/roemmele/AbLit.

1 Introduction

An abridgement is a shortened form of a text that maintains the linguistic qualities of that text[1]. It is intended to make the original text faster and easier to read. In this paper, we propose abridgement as an NLP problem and describe its connection to existing inference and generation tasks. We present a novel dataset for this task, focused on abridged versions of English literature books, which we refer to as the AbLit dataset. We demonstrate the characteristics of AbLit in terms of the relations between original and abridged texts as well as the challenges of automatically modeling these relations. The dataset and all associated code, including a Python package for easily interfacing with the data, are available at: github.com/roemmele/AbLit.

2 The task of abridgement

2.1 Definition

We define abridgement as the task of making a text easier to understand while preserving its linguistic qualities. As such, abridgement intersects with tasks that fuse natural language inference (NLI) and natural language generation (NLG), in particular summarization and simplification.

Summarization condenses the main content of a text into a shorter version in order to facilitate high-level comprehension of the content. Existing research has used the categories of *extractive* and *abstractive* to describe summaries. In the former, the summary 'extracts' sequences from the text, whereas in the latter the summary 'abstracts' out the meaning of the text and rewrites it. The degree of abstractiveness of a summary is indicated by how much novel text it contains that is not directly in the original text. Like a summary, an abridgement is shorter than its original text, but it preserves more of its language and can be seen as an alternative version rather than a meta-description. According to how summaries are characterized, abridgements are highly extractive, even if some abstraction is needed to connect the extracted components.

Some work has examined summarization of narratives, including literary text (Kazantseva and Szpakowicz, 2010; Mihalcea and Ceylan, 2007; Zhang et al., 2019). Of particular relevance to our work are datasets released by Chaudhury et al. (2019), Kryściński et al. (2021), and Ladhak et al. (2020), all of which consist of summaries of fiction books. These summaries are significantly different from abridgements in that they are highly abstractive; they convey the book's narrative without preserving much of the text itself. Kryściński et al. provides summaries at different levels of granularity (book, chapter, and paragraph). Their analysis demonstrates that even the finer-grained summaries at the paragraph level are quite abstractive.

The task of simplification also aims to make a text easier to understand, but without significantly distilling its content. Simplification is often treated as a sentence-level task (Sun et al., 2021). Abridgement can be viewed as simplification on a docu-

---

[1] The term "linguistic qualities" is broad, which reflects other definitions of abridgement. For instance, the Wikipedia entry for "abridgement" specifies that it "maintains the unity of the source", but these dimensions of unity are tacitly defined.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3717–3733
May 2-6, 2023 ©2023 Association for Computational Linguistics

ment level. It seeks to balance the goal of increasing readability with preserving as much of the original text as possible. Research on simplification has been constrained by a lack of high-quality publicly available datasets. Existing datasets have been derived from sources like Wikipedia (e.g. Coster and Kauchak, 2011) and news articles (Xu et al., 2015), but none have focused on literary text.

2.2 Practical application

There are few authors who perform abridgement, and thus relatively few abridged versions of books (Minshull, 2001). Authors have described it as challenging and time-consuming to discern what to modify without compromising the original author's agency (Lauber, 1998; Sussman, 1988). However, as touted by these authors, abridgement makes books more accessible to a larger audience, especially when delivering the content through non-text modes like audio (Lavin, 2014). Given this, automating the abridgement process could vastly expand the number of abridged versions of books and thus increase their readership. Automation does not preclude the involvement of human authors; for example, human translators use machine translation to increase their productivity (e.g. Zhechev, 2014), and the same paradigm could apply to abridgement.

3 Creating an abridgement dataset

The AbLit dataset is derived from 10 classic English literature books, listed in A.4. These books are in the public domain and available through Project Gutenberg[2]. A single author, Emma Laybourn, wrote abridged versions of these books that are also freely available[3]. The author explains:

> "This is a collection of famous novels which have been shortened and slightly simplified for the general reader. These are not summaries; each is half to two-thirds of the original length. I've selected works that people often find daunting because of their density or complexity: the aim is to make them easier to read, while keeping the style intact."

Informed by this, we designed AbLit to capture the alignment between passages in a text and its abridged version. We specify that an abridged and original passage are aligned if the content of the abridged passage is fully derived from the original.

After obtaining the original and abridged books from their respective sites, we detected chapter headings to split the books into chapters (see A.1 for details). We paired the original and abridged version of each chapter according to these headings. Obviously, the two versions already form a broad alignment unit, but our goal was to examine finer levels of alignment. We chose to use sentences as the minimal alignment units, since they are intuitive units of expression in text and can be detected automatically[4]. AbLit annotates sentence boundaries by indexing their position in the text, which enables all whitespace characters (most importantly, line breaks marking paragraphs) to be preserved.

3.1 Automated alignments

We pursued an automated approach to establish initial alignments between the original and abridged sentences for each chapter. It follows the same dynamic programming scheme used to create the Wikipedia Simplification dataset (Coster and Kauchak, 2011). We refer to a group of adjacent sentences in a text as a *span*. We define the length of a span by the number of sentences it contains. Each span o of length o_n in the original version of a chapter is paired with a span a of length a_m in the abridged version. The value of a_m can be zero, allowing for the possibility that an original sentence is aligned with an empty string.

For each pair of o and a, we use a similarity metric $sim(o, a)$ to score the likelihood that they are aligned. This scoring function also considers the length of the spans in order to optimize for selecting the narrowest alignment between the original and abridged text. For instance, if a one-to-one alignment exists such that the meaning of a single sentence in the abridgement is fully encapsulated by a single original sentence, these sentences should form an exclusive alignment. To promote this, we adjust $sim(o, a)$ by a penalty factor pn applied to the size of the pair, where $size = \max(o_n, a_m)$. Ultimately, the alignment score for a given span pair (o, a) is $\max(0, sim(o, a) - ((size - 1) * pn))$. Thus, more similar pairs obtain higher scores, but the scores are increasingly penalized as their size increases. At each sentence position in the original and abridged chapters, we score spans of all lengths $[1, o_n]$ and $[0, a_m]$, then select the one that obtains the highest score when its value is combined with the accumulated score of the aligned spans prior to

---

[2] gutenberg.org

[3] englishliteratureebooks.com

[4] We used nltk.org for all sentence segmentation and word tokenization. For analyses pertaining to words, words are lowercased without any other normalization (e.g. lemmatization).

that position. Once all span pairs are scored, we follow the backtrace from the highest-scoring span in the final sentence position to retrieve the optimal pairs for the chapter. We refer to each resulting span pair (o, a) in this list as an alignment *row*.

3.2 Assessment of automated alignments

We applied this automated alignment approach to the first chapter in each of the ten books in ABLIT, which we designated as an *assessment* set for investigating the quality of the output rows. We instantiated $sim(o, a)$ as the ROUGE-1 (unigram) precision score[5] between the spans, where a is treated as the hypothesis and o is treated as the reference. Here we refer to this score as R-1_p. It effectively counts the proportion of words in a that also appear in o. We considered values of o_n in $[1, 6]$ and a_m in $[0, 6]$ and selected $o_n = 3$ and $a_m = 5$ based on our perceived quality of a sample of output rows. We similarly optimized pn values in $[0, 0.25]$ and selected pn = 0.175. Smaller values of pn yielded rows that were not minimally sized (i.e. they needed to be further split into multiple rows), whereas larger values tended to wrongly exclude sentences from rows.

The output consisted of 1,126 rows, which were then reviewed and corrected by five human validators recruited from our internal team. Validators judged a row as correct if the meaning expressed by the abridged span was also expressed in the original span, consistent with how alignment is defined above. A.3 gives more detail about this task. We found that inter-rater agreement was very high (Fleiss' κ = 0.984) and the few disagreements were easily resolved through discussion. The validators reported spending 10-15 minutes on each chapter.

After establishing these gold rows for the assessment set, we evaluated the initial automated rows with reference to the gold rows. To score this, we assigned binary labels to each pair of original and abridged sentences, where pairs that were part of the same row were given a positive class label and all other pairs were given a negative class label. Given these labels for the rows automatically produced with the R-1_p method compared against the labels for the gold rows, the F1 score of the automated rows was 0.967. We also evaluated other methods for computing $sim(o, a)$, but none outperformed R-1_p. See A.2 for the description of these alternative methods and their results.

3.3 Full dataset

Partial validation: The time spent validating this assessment set indicated it would require significant resources to review the rows for all 868 chapters across the 10 books. Meanwhile, our evaluation revealed that we can expect most automatically aligned rows to be correct. Thus, we considered how to focus effort on correcting the small percentage of erroneously aligned rows. We manually reviewed these rows in the assessment set and found that their R-1_p scores were often lower than those of the correct rows. Moreover, this tended to affect two types of rows: those with two or more sentences in the abridged span, or those adjacent to another row with an empty abridged span (i.e. a_m = 0). We did an experiment where a human validator reviewed only the assessment rows with scores < 0.9 that qualified as one of the two above cases. Selectively applying corrections to just these rows boosted the F1 score of the assessment set from 0.967 to 0.99. We thus decided to apply this strategy of partially validating automated rows to create the training set for ABLIT.

Final sets: To construct the rest of ABLIT, we ran the automated alignment procedure on all other chapters, and then applied the above partial validation strategy. Because we previously confirmed high inter-rater agreement, a single validator reviewed each chapter. Generalizing from the assessment set, we estimate that 99% of these rows are correct. To ensure an absolute gold standard for evaluation, we set aside five chapters in each of the ten books and fully validated their rows. We designated this as the test set, and repurposed the assessment set to be a development set that we used accordingly in our experiments. All other chapters were assigned to the training set. Ultimately, ABLIT consists of 808, 10, and 50 chapters in the training, development, and test sets, respectively. Table 1 shows some examples of rows in ABLIT.

4 Characterizing abridgements

4.1 Overview

Table 2 lists the size of ABLIT in terms of rows, paragraphs, sentences, and words (see A.4 for these numbers compared by book). Here we call attention to the numbers for the fully-validated test set, but the numbers for the training set closely correspond. The development set slightly varies from the training and test set for a few statistics, likely

---

[5] Using github.com/Diego999/py-rouge

Original Span	Abridged Span
[**The letter** was not unproductive.] [It **re-established peace and kindness.**]	[**The letter re-established peace and kindness.**]
[**Mr. Guppy sitting on the window-sill,** nodding his head and balancing all these possibilities in his mind, continues **thoughtfully** to tap **it**, and clasp it, and measure it with his hand **until he hastily draws his hand away.**]	[**Mr. Guppy sitting on the window-sill, taps it thoughtfully, until he hastily draws his hand away.**]
[**At last the gossips thought they had found the key to her conduct, and her uncle was sure of it**; and what is more, **the discovery** showed **his niece** to him in quite a new light, and he changed **his whole** deportment **to** her accordingly**.**]	[**At last the gossips thought they had found the key to her conduct, and her uncle was sure of it .**] [**The discovery altered his whole behaviour to his niece.**]
[**They trooped** down into the hall and into the carriage, Lady Pomona leading the way**.**] [**Georgiana** stalked **along**, passing **her father at the front door without condescending to look at him.**]	[**They trooped downstairs**, **Georgiana stalking along.**] [**She passed her father at the front door without condescending to look at him.**]

Table 1: Examples of alignment rows. Sentence boundaries are denoted by brackets ([]). We highlight preserved words in **blue** and underline the reordered ones. Added words are in **green**.

	Train	Dev	Test (Chpt Mean)
Chpts	808	10	50
Rows	115,161	1,073	9,765 (195)
O_{pars}	37,227	313	3,125 (62)
A_{pars}	37,265	321	3,032 (61)
O_{sents}	122,219	1,143	10,431 (209)
A_{sents}	98,395	924	8,346 (167)
$\%A_{sents}$	80.5	80.8	80.0
O_{wrds}	2,727,571	29,908	231,878 (4,638)
A_{wrds}	1,718,919	17,630	143,908 (2,878)
$\%A_{wrds}$	63.0	58.9	62.1

Table 2: Number of chapters (Chpts), alignment rows ($Rows$), paragraphs ($pars$), sentences ($sents$), and words ($wrds$) across all original (O) and abridged (A) books. The per-chapter means appear for the test set.

O_{sents}	A_{sents}	Train	Dev	Test
1	1	75.8	74.7	75.7
1	0	17.4	17.3	17.3
2+	1	4.3	4.8	4.6
1	2+	2.1	3.2	1.9
2+	2+	0.3	0.0	0.5

Table 3: Distribution of row sizes by number of sentences ($sents$) in original (O) and abridged (A) spans

because it is smaller. Judging by the test set, the abridged chapters have almost the same number of paragraphs as the original, but they have 80% of the number of sentences ($\%A_{sents}$) and ≈62% of the number of words ($\%A_{wrds}$).

Table 3 pertains to the size of the original and abridged spans in each row, where size is the number of sentences in each span. The table shows the relative percentage of rows of each size. The majority of test rows (≈76%) contain a one-to-one alignment between an original and abridged sentence (i.e. $O_{sents} = 1$, $A_{sents} = 1$). Meanwhile, ≈17% contain an original sentence with an empty abridged span ($O_{sents} = 1$, $A_{sents} = 0$). A minority of rows (≈5%) have a many-to-one alignment ($O_{sents} = 2+$, $A_{sents} = 1$) and a smaller minority (≈2%) have a one-to-many alignment ($O_{sents} = 1$, $A_{sents} = 2+$). Many-to-many alignments ($O_{sents} = 2+$, $A_{sents} = 2+$) are more rare (0.5%).

4.2 Lexical similarity

Score Bin	Train	Dev	Test
0.0	17.5	17.6	17.4
(0.0, 0.25]	0.1	0.2	0.1
(0.25, 0.5]	0.5	0.9	0.6
(0.5, 0.75]	2.6	4.6	2.9
(0.75, 1.0)	23.9	31.5	24.0
1.0	55.5	45.2	55.0

Table 4: Binned distribution of $R\text{-}1_p$ scores for rows

As demonstrated by the success of the $R\text{-}1_p$ metric for creating alignment rows (Section 3.2), an original and abridged span typically align if most of the words in the abridged are contained in the original. Table 4 shows the binned distribution of the $R\text{-}1_p$ scores for the rows. Rows with an exact score of 0.0 (≈17% of rows in the test set) consist almost exclusively of original spans aligned to empty spans, which is why this number is comparable to the second line of Table 3. Many rows have perfect scores of exactly 1.0 (55%), signifying that their abridged span is just an extraction of some or all of the original span. The abridged spans where this is not the case (i.e. they contain some words not in the original) still copy much of the original: 24% of test rows have a $R\text{-}1_p$ score above 0.75 and below 1.0, while only a small minority (≈4%, the sum of lines 2-4 in the table) have a score above 0.0 and below 0.75.

4.3 Lexical operations

	Train	**Dev**	**Test**
O_{rmv} (O_{prsv})	40.9 (59.1)	45.9 (54.1)	41.9 (58.1)
A_{add} (A_{prsv})	6.3 (93.7)	8.3 (91.7)	6.4 (93.6)
$Rows_{rmv}$	71.1	80.3	73.2
$Rows_{prsv}$	82.5	82.7	82.6
$Rows_{add}$	37.4	48.8	39.4
$Rows_{reord}$	11.8	16.5	11.7

Table 5: *Top:* the % of removed and added words relative to all original and abridged words, respectively. *Bottom:* the % of rows with each lexical operation.

For each row, we enumerate the common and divergent items between the words o_{wrds} in the original span and the words a_{wrds} in the abridged span. The words that appear in o_{wrds} but not a_{wrds} are removed words, i.e. $o_{rmv} = |o_{wrds} - a_{wrds}|$. All other original words are preserved in the abridgement, i.e. $o_{prsv} = |o_{wrds} - o_{rmv}|$. Accumulating these counts across all original spans $o \in O$, the top section of Table 5 indicates the percentages of removed and preserved words among all original words. In the test set, ≈42% of words are removed, and thus ≈58% are preserved. Next, we count the added words in the abridgement, which are those that appear in a_{wrds} and not o_{wrds}, i.e. $a_{add} = |a_{wrds} - o_{wrds}|$. All other abridged words are preserved from the original, i.e. $a_{prsv} = |a_{wrds} - a_{add}|$. Accumulating these counts across all abridged spans $a \in A$, Table 5 shows that only ≈6% of abridged words in the test set are additions, and thus ≈94% are preservations.

We also report the number of rows where these removal, preservation, and addition operations occur at least once. For instance, if $o_{rmv} > 0$ for the original span in a given row, we count that row as part of $Rows_{rmv}$. The bottom section of Table 5 shows the percentage of rows with each operation among all rows in the dataset. In ≈73% of the test rows, the abridged span removes at least one word from the original. In ≈83% of rows, the abridged span preserves at least one word from the original. In ≈39% of rows, the abridged span adds at least one word not in the original. We considered the possibility that preserved words could be reordered in the abridgement. To capture this, we find the longest contiguous sequences of preserved words (i.e. "slices") in the abridged spans. A row is included in $Rows_{reord}$ if at least two abridged slices appear in a different order compared to the original span. This reordering occurs in ≈12% of rows.

It is clear from this analysis that the abridgements are quite loyal to the original versions, but they still remove a significant degree of text and introduce some new text. The examples in Table 1 highlight these operations. We can qualitatively interpret from the examples that some added words in the abridged span are substitutions for removed original words (e.g. "tap" > "taps" in the second example, "changed" > "altered" in the third example). See A.5 for additional discussion about how some of these relations pertain to common NLI tasks.

4.4 Lexical categories

Category	$\%O$	$\%O_{rmv}$	$\%A$	$\%A_{add}$
Function	58.2	57.9	58.1	53.9
Content	41.8	42.1	41.9	46.1

Table 6: Test set distribution of lexical categories for removed words O_{rmv} compared with all original words O, and added words A_{add} compared with all abridged words A

We examined if certain types of words are more often affected by removal or addition operations. Table 6 contains a broad analysis of this for the test set. As shown, ≈58% of original words O are function words (those with part-of-speech tags of punctuation, pronouns, adpositions, determiners, etc.), while ≈42% are content words (nouns, verbs, adjectives, and adverbs). The category distribution of removed words O_{rmv} closely matches the O distribution, suggesting that both function and content words are removed at the same rate. The abridged words A have the same proportion of function and content words as O (again, ≈58% and ≈42%). In comparison, ≈54% of additions A_{add} are function words, while ≈46% are content words. The gap between ≈42% and ≈46% indicates that content words are added at a slightly higher rate than the overall frequency in content words in A (and equivalently, function words are added at a lower rate). But there are few additions overall, so the abridgements retain the same word type distribution as the original texts. A.6 shows this same analysis for each specific part-of-speech tag among these types.

5 Predicting what to abridge

Garbacea et al. (2021) points out that a key (and often neglected) preliminary step in simplification is to distinguish text that could benefit from being simplified versus text that is already sufficiently simple. This is also an important consideration for abridgement, since it seeks to only modify text in places where it improves readability. Accordingly, we examine whether we can automatically predict the text in the original that should be removed when producing the abridgement. As explained in Section 4, a removed word could mean the author replaced it with a different word(s) in the abridgement, or simply excluded any representation of its meaning. However, both cases indicate some change is applied to that word.

We model this through a binary sequence labeling task. Given a passage with original tokens o_{toks} and corresponding abridged tokens a_{toks}, we assign each token t in o_{toks} the label of `preserved` (l=0) if it is also in a_{toks}, and otherwise the label of `removed` (l=1) if it is not in a_{toks}. Thus the task is to predict the label sequence $[l_1, l_2, ...l_n]$ from the token sequence $[t_1, t_2, ...t_n]$.

5.1 Model inputs

We can derive a token-label sequence from each alignment row, by which each original span corresponds to a single input instance. However, the size of these spans varies across rows. To produce models that handle texts where these span boundaries are not known in advance, we consider consistent-length passages whose boundaries can be automatically inferred. Thus the ABLIT interface can provide pairs where a fixed-length passage from the original chapter (i.e. a sentence, paragraph, or multi-paragraph chunk) is aligned to its specific corresponding abridged version.

We enable this by finding the respective positions of the longest common word sequences between the original and abridged spans. Each of these overlapping subsequences is represented as a slice of the original text with indices (o_i, o_j) mapped to a slice of the abridged text (a_i, a_j). Then, given a passage in the original text with indices (o_l, o_m), we find all enclosed slices (o_i, o_j) where $o_i >= o_l$ and $o_j <= o_m$. For each slice we retrieve its corresponding abridged slice (a_i, a_j). Given the earliest text position $\min a_i$ and latest position $\max a_j$ among these abridged slices, the full abridgement for the passage at (o_l, o_m) is the text covered by the indices ($\min a_i$, $\max a_j$). As an example, consider the first line in Table 1. If retrieving abridgements for sentence-length passages, the first sentence in the original span "The letter was not unproductive." will yield "The letter" as the abridgement. The second original sentence "It re-established peace and kindness" will yield the abridgement "re-established peace and kindness". By varying the passage size, we can assess how much context beyond a single row is beneficial in modeling abridgements. See A.7 for more details.

5.2 Experiment

Model: To predict abridgement labels (`preserved`/`removed`), we use a ROBERTA-based sequence labeling model, which has been applied to several other NLI tasks (Liu et al., 2019). We divided chapters according to varying passage sizes and trained a separate model on the token-label sequences[6] associated with each passage size. The passages are either sentences (detected by NLTK), paragraphs (detected by line breaks), or multi-paragraph 'chunks'. Each chunk consists of one or more paragraphs of S sentences, such that paragraphs are combined into the same chunk when their total number of sentences does not exceed S. As an additional reference, we trained a model where each passage is an original span directly taken from a single alignment row. As explained in Section 5.1, these passages (termed Rows) vary in length. We did not train a model on the full chapters as inputs because the average length of these inputs (5,044 tokens) greatly exceeds the ROBERTA limit of 512. See A.8 for more details about the model.

Passage	**Toks**	**P**	**R**	**F1**
Rows	26	0.692	0.442	0.532
Sentences	24	0.677	0.453	0.535
Paragraphs	81	0.686	0.460	0.546
Chunks (S=10)	303	0.670	0.501	0.569
All=`removed`	-	0.415	1.000	0.583

Table 7: F1 scores of abridgement label prediction for test set with models trained on varying passage sizes. **Toks** is the mean number of tokens in each passage type.

Results: Each model is evaluated on instances of the corresponding passage size in the test set. Table

---

[6] A "token" in this case is a sub-token unit defined by the ROBERTA tokenizer, rather than a whitespace-separated "word" pertaining to Section 4.

7 displays the results in terms of the precision (P), recall (R), and F1 score of predicting that a token should be removed. We compare these results with the baseline of labeling all tokens in the chapter as `removed` (final line). For chunks, we tuned different values of S in $[5, 11]$ on the development set and observed the best F1 at S=10. The results show that the longest passage size (Chunks) yields the best predictions, suggesting the importance of chapter context beyond that given in a single row. The consistently higher precision over recall for all models indicates they correctly predict many preservation operations, but at the expense of missing many removal operations. Consequently, they overestimate the number of tokens that should be preserved. This results in an overall F1 that is lower than what occurs when all tokens are removed.

6 Producing abridgements

The above results show that anticipating what parts of a text should be changed when writing its abridged version is not trivial. The full task of producing an abridgement implicitly involves inferring these `preserved`/`removed` labels while additionally predicting the specific text that dictates these labels. We examine models that have been applied to tasks related to abridgement to establish benchmarks for this new task, with the intent that these benchmarks will inspire future work.

6.1 Models

We consider the following models to produce an abridged version of an original chapter:

Naive Baselines: As a reference point for our evaluation metrics, we report the performance of very weak baselines. In particular, we copy the entire original text as the abridgement (COPY). Alternatively, we select T percent of original tokens (RANDEXTTOKS) as the abridgement.

Extractive Approaches: The analysis in Section 4.2 showed that abridgements preserve much of their original text, which motivates the use of extractive summarization methods. Using the best label prediction model from Section 5, we extract all original tokens labeled as `preserved` to form the abridgement (EXTTOKS). To reveal the maximum performance that can be obtained with this method, we also run it using the gold labels instead of predicted labels (PERFECTEXTTOKS). It is not conventional to use tokens as units of extraction, since it can compromise fluency within sentences. EXTTOKS and PERFECTEXTTOKS only serve as points of comparison for our evaluation metrics. The standard extractive approach uses sentences as extractive units. For this (EXTSENTS), we form an abridgement by selecting a subset of sentences in the original chapter where at least P percent of tokens are labeled as `preserved`.

Generation Models: Extractive methods cannot introduce words into the abridgement that are not in the original, so for this we need to consider generation models. In particular, we examine two transformer-based sequence-to-sequence models that have been used for various generation tasks including summarization: T5-BASE (Raffel et al., 2020) (termed TUNEDT5 here) and BART-BASE (Lewis et al., 2020) (termed TUNEDBART). We fine-tuned both models on the ABLIT training set, specifically on inputs consisting of chunks with 10 sentences, since this passage size yielded the best results in the Section 5 experiment. To assess the impact of these models' observation of ABLIT, we compare them with abridgements produced by prompting the non-finetuned T5-BASE to perform zero-shot summarization (ZEROSHOTT5). See A.9 for more details about these models. For all models, we generated an abridgement for an original chapter by dividing the chapter into chunks, generating output for each chunk (with 5-beam decoding), then concatenating the outputs to form the complete abridgement.

6.2 Evaluation metrics

We evaluate the predicted abridgements through comparison with the human-authored reference abridgements. First, we measure the word-based similarity between the predicted abridgement a_{pred} and reference abridgement a_{ref} using ROUGE-L ($R\text{-}L$), a standard evaluation metric for summarization. We then assess how accurately a_{pred} removed and preserved words from the original. A word from the original in a_{pred} is considered correctly preserved if it also appears in a_{ref}. We report the F1 of this measure as $Prsv$. A word in the original but not in a_{pred} is considered correctly removed if it is also absent from a_{ref}. We report the F1 of this measure as Rmv. Finally, we evaluate the accuracy of added words, where a word not in the original is considered correctly added to a_{pred} if it is also in a_{ref}. We report the F1 of this measure as Add. See A.10 for formal definitions of these metrics.

Name	Description	Toks	$R\text{-}L$	$Prsv$	Rmv	Add
Human	Reference (a_{ref})	2,878	-	-	-	-
Copy	Duplicate original	4,638	0.739	0.753	0.000	0.000
RandExtToks (T=0.6)	$T\%$ randomly selected original tokens	2,787	0.753	0.800	0.694	0.000
ExtToks	Original tokens predicted as `preserved`	3,160	0.818	0.856	0.745	0.006
PerfectExtToks	Original tokens where gold label is `preserved` (upper bound for ExtToks)	2,664	0.950	0.969	0.954	0.034
ExtSents (P=0.65)	Original sentences with $\geq P\%$ tokens predicted as `preserved`	2,857	0.792	0.824	0.720	0.001
TunedT5	Generate from finetuned T5	3,834	0.727	0.804	0.519	0.275
TunedBART	Generate from finetuned BART	3,673	0.780	0.815	0.583	0.365
ZeroShotT5	Generate from non-finetuned T5	1,157	0.416	0.484	0.627	0.019

Table 8: Scores of predicted abridgements on evaluation metrics. For all metrics, higher scores are better.

Original	Reference	TunedBART
The windows were half open because of the heat, and the Venetian blinds covered the glass,–so that a gray grim light, reflected from the pavement below, threw all the shadows wrong, and combined with the green-tinged upper light to make even Margaret's own face, as she caught it in the mirrors, look ghastly and wan.	The windows were half open because of the heat, and Venetian blinds covered the glass, giving the light a green tinge that made her face in the mirrors look ghastly and wan.	The windows were half open because of the heat, and the Venetian blinds covered the glass - so that a grey grim light, reflected from the pavement below, threw all the shadows wrong, and made even Margaret's own face look ghastly and wan.
We must suppose little George Osborne has ridden from Knightsbridge towards Fulham, and will stop and make inquiries at that village regarding some friends whom we have left there. How is Mrs. Amelia after the storm of Waterloo? Is she living and thriving? What has come of Major Dobbin, whose cab was always hankering about her premises?	We must now make inquiries at Fulham about some friends whom we have left there. How is Mrs. Amelia? Is she living and thriving? What has become of Major Dobbin?	We must suppose little George Osborne has ridden towards Fulham, and will stop and make inquiries about some friends whom we have left there. How is Mrs. Amelia after the storm of Waterloo? Is she living and thriving? What has come of Major Dobbin, whose cab was always hankering about her premises?

Table 9: Abridgements predicted by TunedBART for excerpts of North and South and Vanity Fair

6.3 Results

Table 8 reports the length and metric scores of the abridgements produced by each model for the test set chapters. Where applicable, we selected the T and P parameters from tuning on the development set. The results again convey that abridgement is largely a text extraction task, though a challenging one. The low $R\text{-}L$ score of ZeroShotT5 confirms that AbLit is different from the summarization datasets that T5-base is trained on. The high $R\text{-}L$ of PerfectExtToks validates that precisely identifying which words to remove goes far in producing the abridgement. The high $Prsv$ scores for all approaches that observe AbLit show they can all preserve the original text reasonably well. Analogous to the results in Section 5, the lower Rmv scores indicate knowing which words to remove is harder, particularly for the generation models. The extractive methods have no opportunity to obtain an Add score that is non-trivially above 0[7]. The generation models do show a small benefit here in correctly adding some new words to the abridgement. The examples in Table 9 qualitatively represent the outcome for the TunedBART model. These abridgements remove some of the same original text as the reference and also add a few words consistent with the reference, but they still retain more of the original text than the reference. Other examples are shown in A.12.

7 Conclusion

In this paper, we introduced AbLit, a corpus of original and abridged versions of English literature. AbLit enables systematic analysis of the abridgement task, which has not yet been studied from an NLP perspective. Abridgement is related to other tasks like summarization, but has a stricter requirement to maintain loyalty to the original text. Our experiments motivate an opportunity to better balance this goal against that of improving readability. We also envision future resources that generalize this task to other texts beyond English literature.

---

[7] It is possible for Add to be slightly above 0 with the extractive approaches due to tokenization; see A.10.

8 Limitations

We present AbLit to introduce abridgement as an NLP task. However, the dataset is scoped to one small set of texts associated with a specific domain and author.

There are significant practical reasons for this limited scope. In particular, most recently published books are not included in publicly accessible datasets due to copyright restrictions, and the same restrictions typically apply to any abridgements of these books. The books in AbLit are uniquely in the public domain due to expired copyrights, and the author chose to also provide her abridgements for free. For this reason, AbLit consists of British English literature from the 18th and 19th centuries. Some of the linguistic properties of these original books do not generalize to other types of English texts that would be useful to abridge. We do not yet know what aspects of abridgement are specific to this particular domain.

Moreover, as described in Section 2.2, creating abridgements is a rare and highly skilled writing endeavor. The AbLit abridgements are written exclusively by one author. Without observing alternative abridgements for the same books by a different author, it is unclear what features are specific to the author's preferences. This conflation between task and author is a concern for many NLP datasets (Geva et al., 2019). More generally, obtaining human writing expertise is a challenge shared by all language generation research as it becomes more ambitious (e.g. Wu et al., 2021).

9 Ethical Considerations

As stated in the introduction, all data and code used in this work is freely available. The text included in the dataset is in the public domain. Additionally, we explicitly confirmed approval from the author of the abridged books to use them in our research.

For the data validation task, the validators were employed within our institution and thus were compensated as part of their normal job role. Given that the dataset is derived directly from published books, it is possible that readers may be offended by some content in these books. The validators did not report any subjective experience of this.

With regard to our modeling approaches, large pretrained models like the ones we use here for generating abridgements have a well-known risk of producing harmful content (e.g. Gehman et al., 2020). For the generation models fine-tuned on AbLit, we did not subjectively observe any such text in the sample output we assessed. We judge that our controlled selection of training data reduces this risk, but does not eliminate it. Accordingly, future applications of abridgement can similarly consider careful data curation for mitigating this risk.

References

Atef Chaudhury, Makarand Tapaswi, Seung Wook Kim, and Sanja Fidler. 2019. The shmoop corpus: A dataset of stories with loosely aligned summaries. *arXiv preprint arXiv:1912.13082*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

William Coster and David Kauchak. 2011. Simple English Wikipedia: A new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 665–669, Portland, Oregon, USA. Association for Computational Linguistics.

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

Cristina Garbacea, Mengtian Guo, Samuel Carton, and Qiaozhu Mei. 2021. Explainable prediction of text complexity: The missing preliminaries for text simplification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1086–1097, Online. Association for Computational Linguistics.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166, Hong Kong, China. Association for Computational Linguistics.

Anna Kazantseva and Stan Szpakowicz. 2010. Summarizing short stories. *Computational Linguistics*, 36(1):71–109.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*.

Faisal Ladhak, Bryan Li, Yaser Al-Onaizan, and Kathleen McKeown. 2020. Exploring content selection in summarization of novel chapters. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5043–5054, Online. Association for Computational Linguistics.

Lynn Lauber. 1998. Bookend; confessions of (an) abridger. *The New York Times*.

Brittany Lavin. 2014. Abridgement: What it can mean for your book. *Opyrus*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Rada Mihalcea and Hakan Ceylan. 2007. Explorations in automatic book summarization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 380–389, Prague, Czech Republic. Association for Computational Linguistics.

Duncan Minshull. 2001. The incredible shrinking book. *The Guardian*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Renliang Sun, Hanqi Jin, and Xiaojun Wan. 2021. Document-level text simplification: Dataset, criteria and baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7997–8013, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Vic Sussman. 1988. The fine art of abridgement. *The Washington Post*.

Jeff Wu, Long Ouyang, Daniel M Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. 2021. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Weiwei Zhang, Jackie Chi Kit Cheung, and Joel Oren. 2019. Generating character descriptions for automatic summarization of fiction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7476–7483.

Ventsislav Zhechev. 2014. Analysing the post-editing of machine translation at autodesk. *Post-editing of machine translation: Processes and Applications*, pages 2–24.

A Appendix

A.1 Detecting chapter boundaries

There is a one-to-one relation between each chapter in an original book and each chapter in its corresponding abridged version. Both the original and abridged version of the books include headings separating chapters. We automatically detected these headings through a set of regular expressions (e.g. matching lines specifying a chapter number and name with the regular expression "`^Chapter [0-9]+:*[a-zA-Z\s]*$`"). However, there is variability in the format of the headings: some can span multiple lines, or specify a book and volume number in addition to the chapter identifier, or have numbers written in non-numerical form, for instance. The format also varies between the original and abridged version of the same book. Thus, we manually reviewed all detected chapter boundaries and fixed any erroneous or missed boundaries. Ultimately we ensure that each chapter in an original book is paired exactly with its abridged counterpart.

A.2 Additional automated alignment results

Table 10 shows the results of all methods we assessed for computing similarity between original and abridged spans to create alignment rows, compared alongside the best method of unigram ROUGE precision ($R\text{-}1_p$) reported in Section 3.2. A clear drawback to using unigram overlap to measure similarity is that it does not account for differences in word order. However, taking this into account by using bigrams instead of unigrams to calculate ROUGE precision (i.e. $R\text{-}2_p$) reduced the F1 to 0.935, likely because it added more sparsity to the overlap units. In addition to the word-based ROUGE metric, we assessed vector-based similarity encoded by different configurations of pretrained language models: BERT (Devlin et al., 2019), XLNET (Yang et al., 2019), XLM (Conneau and Lample, 2019), and ROBERTA (Liu et al., 2019). We used the HuggingFace Transformers implementation of these models: `https://huggingface.co/docs/transformers/index`. For each model we report the best result among size penalty (pn) values in $[0, 0.25]$. As displayed, the vectors that obtained the best F1 came from BERT (Devlin et al., 2019), particularly BERT-BASE-UNCASED, which consists of 110M parameters. See additional details about this model here: `https://huggingface.co/bert-base-uncased`. Ultimately, however, the result from BERT-BASE-UNCASED was still outperformed by $R\text{-}1_p$. As reported in Section 3.3, the resulting rows were further improved by applying the described partial validation strategy (final line of table).

A.3 Details about validation task

For each row, validators assessed whether the abridged span in the row was correctly aligned with the corresponding original span. As described in Section 3.2, a row is correct if the meaning of the abridged span can be derived from the original span. For a given row, if the abridged span expressed some meaning not contained in the original span, it either meant that some sentences(s) in the abridged chapter were incorrectly placed in that row, or some sentence(s) in the original chapter were incorrectly placed in a different row. In both cases, validators moved the wrongly placed sentence(s) to a row resulting in correctly aligned spans. We utilized Google Sheets as an interface for this task, which enabled validators to easily review and correct the rows. We produced a single spreadsheet per chapter, where each spreadsheet row corresponded to an alignment row. For the partial validation strategy, we designed a Google Apps Script (`https://developers.google.com/apps-script`) that visually highlighted spreadsheet rows qualifying for partial validation so that validators could specifically attend to those rows.

For the development (assessment) and test sets, there were a few cases where the validators edited the spans themselves in order to correct sentence segmentation errors (e.g. wrongly segmenting after honorifics like "Mr.").

A.4 Size of ABLIT compared by book

Table 11 shows characteristics of the data for each book in terms of number of alignment rows, original words, and abridged words.

A.5 NLI challenges in ABLIT

Table 12 shows some examples of rows in ABLIT where modeling the relation between the original and abridged span involves NLI challenges like abstractive paraphrasing, figurative language interpretation, commonsense reasoning, and narrative understanding.

Similarity Metric	pn	**P**	**R**	**F1**
Vector cosine similarity				
BERT-base-uncased	0.21	0.963	0.952	0.957
BERT-base-cased	0.22	0.948	0.934	0.940
BERT-large-uncased	0.21	0.934	0.919	0.926
BERT-large-cased	0.21	0.944	0.935	0.939
XLNet-base-cased	0.22	0.753	0.731	0.742
XLNet-large-cased	0.21	0.583	0.564	0.573
XLM-mlm-en	0.21	0.821	0.816	0.818
RoBERTa-base	0.21	0.738	0.717	0.727
RoBERTa-large	0.21	0.592	0.573	0.582
Word overlap similarity				
R-1_p	0.175	0.964	0.969	0.967
R-2_p	0.175	0.912	0.958	0.935
R-1_p + partial human validation	0.175	0.990	0.991	0.990

Table 10: Extended results for accuracy of automated alignment methods

Book (Orig Author)	**Train**			**Dev**			**Test**		
	Rows (Chpts)	O_{wrds}	$\%A_{wrds}$	Rows	O_{wrds}	$\%A_{wrds}$	Rows	O_{wrds}	$\%A_{wrds}$
Bleak House (Charles Dickens)	17,948 (62)	390,857	63.2	24	935	20.0	1,746	38,132	62.9
Can You Forgive Her? (Anthony Trollope)	16,494 (74)	350,092	62.2	94	3,216	49.5	1,339	27,660	61.2
Daniel Deronda (George Eliot)	12,735 (64)	333,283	61.6	158	3,524	61.9	786	25,334	49.1
Mansfield Park (Jane Austen)	5,744 (42)	159,863	67.0	91	3,564	62.1	795	22,607	66.1
North and South (Elizabeth Gaskell)	8,922 (46)	193,355	67.9	184	4,907	68.5	1,169	23,159	70.0
Shirley (Charlotte Bronte)	12,027 (31)	235,888	63.2	253	5,987	57.4	1,031	23,369	60.4
The Way We Live Now (Anthony Trollope)	19,355 (94)	392,554	60.3	166	4,345	53.7	1,122	23,238	60.7
Tristram Shandy (Laurence Sterne)	4,805 (305)	216,984	66.7	5	439	77.0	69	3,972	72.3
Vanity Fair (W. M. Thackeray)	11,682 (62)	334,783	59.8	18	717	60.9	738	23,609	57.4
Wuthering Heights (Emily Bronte)	5,449 (28)	119,912	66.3	80	2,274	68.3	970	20,798	71.0
All	115,161 (808)	2,727,571	63.0	1,073	29,908	58.9	9,765	231,878	62.1

Table 11: Statistics for each book in the AbLit dataset, in terms of number of alignment rows, total original word (O_{wrds}), and proportional length of abridgement relative to original ($\%A_{wrds}$). The number of chapters in the training set for each book is shown; there is 1 chapter per book in the development set and 5 chapters per book in the test set.

Original Span	Abridged Span	Type of Challenge
Still there was not a word.	No one spoke.	Paraphrasing: abridgement has same meaning as original but no word overlap
But it is time to go home; my appetite tells me the hour.	But it is time to go home; I am hungry.	Interpretation of figurative language: abridgement replaces phrase "appetite tells me the hour" with more literal term "hungry"
"Daniel, do you see that you are sitting on the bent pages of your book?"	"Daniel, you are sitting on the bent pages of your book."	Change in dialogue act: question in original is transformed into statement in abridgment
While she was at Matching, and before Mr. Palliser had returned from Monkshade, a letter reached her, by what means she had never learned. "A letter has been placed within my writing-case," she said to her maid, quite openly. "Who put it there?"	While she was at Matching, a letter reached her, by what means she never learned, although she suspected her maid of placing it inside her writing-case.	Dialogue interpretation: abridgement summarizes the narrative event (suspecting maid of placing letter) conveyed by the spoken utterances in the original text ("A letter has been placed... she said to her maid.")
"If you will allow me, I have the key," said Grey. Then they both entered the house, and Vavasor followed his host up-stairs.	Mr. Grey unlocked the door of his house, and Vavasor followed him upstairs.	Commonsense inference: abridgement involves knowledge that doors are unlocked by keys, which is not explicit in the original text
George Osborne was somehow there already (sadly "putting out" Amelia, who was writing to her twelve dearest friends at Chiswick Mall), and Rebecca was employed upon her yesterday's work.	George Osborne was there already, and Rebecca was knitting her purse.	Narrative inference: "knitting her purse" in the abridgement is the event referenced by "yesterday's work" in the original, and resolving this requires knowledge of the previous text in the chapter
But Kate preferred the other subject, and so, I think, did Mrs. Greenow herself.	But Kate preferred the subject of the Captain, and so, I think, did Mrs. Greenow herself.	Elaboration: abridgement specifies "Captain" is the "other subject" implied in the original

Table 12: Examples of rows where alignment involves a language inference challenge

A.6 Extended lexical category analysis

Section 4.4 summarized the frequency of lexical categories for removed and added words in the ABLIT test set, relative to these frequencies among all words in the original and abridged texts. Table 13 additionally displays these percentages for all part-of-speech tags within the function and content word classes, along with examples of common words associated with each tag. We used the spacy library to perform part-of-speech tagging: `https://spacy.io/usage/linguistic-features`.

A.7 Comment about passage size variation

Because the method for converting rows into passages of a consistent length (i.e. sentences, paragraphs, chunks) relies on string matching, the boundaries of the abridged passage may be off by one or a few words, which occurs less frequently as the size of the passages increase. This tends to occur when a word at the end of the original passage is replaced by a synonym in the abridged passage. However, a manual review of our assessment set revealed that only 0.4% of sentences in the original text yielded abridgements with imprecise boundaries, and no paragraphs (and consequently no chunks) had this issue.

A.8 Details about binary prediction model

For all passage sizes, we initialized models with the ROBERTA-BASE weights using the HuggingFace Transformers implemen-

Category	%O	%O_{rmv}	%A	%A_{add}	Examples of Common Words
Function words	**Σ=58.2**	**Σ=57.9**	**Σ=58.1**	**Σ=53.9**	
Punctuation	14.0	12.9	15.7	23.3	`, " . -- ; - ? !`
Pronoun	11.0	10.2	11.5	8.7	`i he it his her`
Adposition	10.2	11.5	9.0	7.0	`of in to with for`
Determiner	7.8	8.3	7.1	3.9	`the a an no all`
Aux. Verb	6.4	6.0	6.5	3.8	`was had be is been`
Coord. Conj.	3.7	4.0	3.4	2.1	`and but or nor both`
Particle	2.7	2.6	2.7	2.2	`to not ’s n’t ’`
Subord. Conj.	2.3	2.5	2.2	2.8	`that as if when upon`
Content words	**Σ=41.8**	**Σ=42.1**	**Σ=41.9**	**Σ=46.1**	
Noun	14.5	15.4	13.7	14.0	`time man day way hand`
Verb	10.4	10.3	11.1	17.1	`said had know do have`
Adjective	6.6	6.9	6.2	5.3	`little own other such good`
Adverb	5.0	5.3	4.7	5.3	`so very as now then`
Proper Noun	4.4	3.3	5.2	3.4	`mr. mrs. sir miss lady`
Other	1.0	0.9	1.0	1.1	`one two oh no yes`

Table 13: Distribution of part-of-speech categories for the set of all removed words O_{rmv} and all added words A_{add} in the ABLIT test chapters. These numbers are respectively compared alongside those for the total set of all original words O and all abridged words A. (Aux.=Auxiliary, Coord.=Coordinating, Conj.=Conjunction, Subord.=Subordinate)

tation: https://huggingface.co/docs/transformers/v4.16.2/en/model_doc/roberta#transformers.RobertaModel. ROBERTA-BASE consists of 125M parameters (https://huggingface.co/roberta-base). The maximum sequence length allowed by this model is 512, so we truncated all input tokens beyond this limit. We fine-tuned each model for 5 epochs, saving model weights after each epoch of training, and selected the model with the highest F1 score on the development set to apply to our test set. We used the AdamW optimizer (Loshchilov and Hutter, 2017) and a batch size of 16. It took ≈2 hours to train each model on a g4dn.2xlarge AWS instance. During evaluation, any input tokens beyond the model length limit were assigned the default label of `preserved`. The result for each model reported in Table 7 is based on a single run of the training procedure.

A.9 Details about generation models

Both TUNEDT5 and TUNEDBART were fine-tuned using the HuggingFace transformers library, in particular this script: http://github.com/huggingface/transformers/blob/master/examples/pytorch/summarization/run_summarization.py. TUNEDT5 was initialized from T5-BASE (Raffel et al., 2020), which consists of ≈220M parameters (https://huggingface.co/t5-base). For this model, we prepended the prefix “summarize: ” to the target (i.e. the abridged passage), consistent with how T5-BASE was trained to perform summarization. TUNEDBART was initialized from BART-BASE (Lewis et al., 2020), which consists of 140M parameters (https://huggingface.co/facebook/bart-base). For both TUNEDT5 and TUNEDBART, we used a maximum length of 1024 for both the source (original passage) and target (abridged passage), and truncated all tokens beyond this limit. We evaluated each model on the development set after each epoch and concluded training when cross-entropy loss stopped decreasing, thus saving the model weights with the optimal loss. We used a batch size of 4. For all other hyperparameters we used the default values set by this script, which specifies AdamW for optimization. It took ≈3 hours to train each model on a g4dn.4xlarge AWS instance. The result for each model reported in Table 8 is based on a single run of the training procedure.

A.10 Details about evaluation metrics

Preservation: The formal definition of the preservation metric $Prsv$ is as follows. If

$o_{prsv}(a_{pred})$ are the words in the original that are preserved in the predicted abridgement, and $o_{prsv}(a_{ref})$ are the words in the original that are preserved in the reference abridgement, then we consider the number of correctly preserved words: $Correct\_Prsv = |o_{prsv}(a_{pred}) \cap o_{prsv}(a_{ref})|$. The precision of this measure $Prsv_p = \frac{Correct\_Prsv}{o_{prsv}(a_{pred})}$ is the proportion of correctly preserved words among all preserved words in the predicted abridgement. The recall $Prsv_r = \frac{Correct\_Prsv}{o_{prsv}(a_{ref})}$ is the proportion of correctly preserved words among all preserved words in the reference abridgement. $Prsv$ is the F1 of these precision and recall measures: $Prsv = 2\frac{Prsv_p \cdot Prsv_r}{Prsv_p + Prsv_r}$.

Removal: The formal definition of the removal metric is as follows. If $o_{rmv}(a_{pred})$ are the words in the original that are removed in the predicted abridgement, and $o_{rmv}(a_{ref})$ are the words in the original that are removed in the reference abridgement, then we consider the number of correctly removed words: $Correct\_Rmv = |o_{rmv}(a_{pred}) \cap o_{rmv}(a_{ref})|$. The precision of this measure $Rmv_p = \frac{Correct\_Rmv}{o_{rmv}(a_{pred})}$ is the proportion of correctly removed words among all removed words for the predicted abridgment. The recall $Rmv_r = \frac{Correct\_Rmv}{o_{rmv}(a_{ref})}$ is the proportion of correctly removed words among all removed words for the reference abridgement. Rmv is the F1 of these precision and recall measures: $Rmv = 2\frac{Rmv_p \cdot Rmv_r}{Rmv_p + Rmv_r}$.

Addition: The formal definition of the addition metric is as follows. If $a_{add}(a_{pred})$ are the words in the predicted abridgement that do not appear in the original, and $a_{add}(a_{ref})$ are the words in the reference abridgement that do not appear in the original, then we consider the number of correctly added words: $Correct\_Add = |a_{add}(a_{pred}) \cap a_{add}(a_{ref})|$. The precision of this measure $Add_p = \frac{Correct\_Add}{a_{add}(a_{pred})}$ is the proportion of correctly added words among all added words in the predicted abridgement. The recall $Add_r = \frac{Correct\_Add}{a_{add}(a_{ref})}$ is the proportion of correctly added words among all added words in the reference abridgement. Add is the F1 of these measures: $Add = 2\frac{Add_p \cdot Add_r}{Add_p + Add_r}$.

A.11 Comment about addition scores

Regarding the above-zero scores of the extractive methods on the Add metric, there are two reasons for this. One reason is that the prediction model uses sub-tokens while the Add metric analyzes whitespace-separated words. Consequently, one sub-token may be predicted as `preserved` while others within the same word are predicted as `removed`. Isolated from these other sub-tokens, the `preserved` sub-token will be recognized as a new added word in the abridgement. The other reason is that a single word in the original may be split by the tokenizer into two words in the abridgement, or vice-versa. For example, we observed that "Mr." gets split into two tokens ("Mr", '.') in some contexts and is treated as one token ("Mr.") in others. If the original text represents this item as two tokens and both the extracted and reference abridgement represent it as a single token, then this single token will be counted as an added word in the extracted abridgement.

A.12 Examples of produced abridgements

Tables 14 and 15 below show excerpts of the abridgements produced by the EXTSENT and TUNEDBART models, alongside the original chapter text and human-authored reference abridgement. The sentences in each excerpt are lined up to better visualize their differences.

Original	**Reference**	**ExtSents**	**TunedBART**
Seven days glided away, every one marking its course by the henceforth rapid alteration of Edgar Linton's state.	In the next seven days Edgar Linton's state grew rapidly worse.		Seven days glided away, every one marking its course by the rapid alteration of Edgar Linton's state.
The havoc that months had previously wrought was now emulated by the inroads of hours.			The havoc that months had previously wrought was now emulated by the inroads of hours.
Catherine we would fain have deluded yet;	Catherine could no longer be deluded:		Catherine we would fain have deluded yet;
but her own quick spirit refused to delude her:		but her own quick spirit refused to delude her:	but her own quick spirit refused to delude her:
it divined in secret, and brooded on the dreadful probability, gradually ripening into certainty.	she brooded on the dreadful probability of her father's death, gradually ripening into certainty.		it brooded on the dreadful probability, gradually ripening into certainty.
She had not the heart to mention her ride, when Thursday came round;	She had not the heart to mention her ride when Thursday came round.	She had not the heart to mention her ride, when Thursday came round;	She had not the heart to mention her ride, when Thursday came round;
I mentioned it for her, and obtained permission to order her out of doors:	I obtained permission to send her out of doors:	I mentioned it for her, and obtained permission to order her out of doors:	I ordered her out of doors:
for the library, where her father stopped a short time daily-the brief period he could bear to sit up-and his chamber, had become her whole world.	for her father's chamber had become her whole world.		for the library, where her father stopped daily - the brief period he could bear to sit up - and his chamber, had become her whole world.
She grudged each moment that did not find her bending over his pillow, or seated by his side.	She grudged each moment that she did not spend bending over his pillow, or seated by his side.		She grudged each moment that did not find her bending over his pillow, or seated by his side.
Her countenance grew wan with watching and sorrow, and my master gladly dismissed her to what he flattered himself would be a happy change of scene and society;	She grew pale with watching, and my master gladly dismissed her to what he thought would be a happy change of scene;	Her countenance grew wan with watching and sorrow, and my master gladly dismissed her to what he flattered himself would be a happy change of scene and society;	Her countenance grew wan with watching and sorrow, and my master gladly dismissed her to what he flattered himself would be a happy change of scene and society;
drawing comfort from the hope that she would not now be left entirely alone after his death.	drawing comfort from the hope that she would not now be left entirely alone after his death.	drawing comfort from the hope that she would not now be left entirely alone after his death.	drawing comfort from the hope that she would not now be left entirely alone after his death.

Table 14: Abridgements for an excerpt of Wuthering Heights, Chapter 27

Original	**Reference**	**EXTSENTS**	**TUNEDBART**
It happened that when I came home from Deal I found a note from Caddy Jellyby (as we always continued to call her), informing me that her health, which had been for some time very delicate, was worse and that she would be more glad than she could tell me if I would go to see her.	When I came home from Deal I found a note from Caddy, informing me that her health, which had been for some time very delicate, was worse and that she would be very glad if I would go to see her.	It happened that when I came home from Deal I found a note from Caddy Jellyby (as we always continued to call her), informing me that her health, which had been for some time very delicate, was worse and that she would be more glad than she could tell me if I would go to see her.	It happened that when I came home from Deal I found a note from Caddy Jellyby informing me that her health, which had been for some time very delicate, was worse and that she would be more glad than she could tell me if I would go to see her.
It was a note of a few lines, written from the couch on which she lay and enclosed to me in another from her husband, in which he seconded her entreaty with much solicitude.	It was a short note, written from her bed.		It was a note of a few lines, written from the couch on which she lay and enclosed to me in another from her husband, in which he seconded her entreaty with much solicitude.
Caddy was now the mother, and I the godmother, of such a poor little baby–such a tiny old-faced mite, with a countenance that seemed to be scarcely anything but cap-border, and a little lean, long-fingered hand, always clenched under its chin.	Caddy was now the mother, and I the godmother, of such a poor little baby - such a tiny old-faced mite, with a little lean, long-fingered hand always clenched under its chin.	Caddy was now the mother, and I the godmother, of such a poor little baby–such a tiny old-faced mite, with a countenance that seemed to be scarcely anything but cap-border, and a little lean, long-fingered hand, always clenched under its chin.	Caddy was now the mother, and I the godmother, of such a poor little baby - such a tiny old-faced mite, with a countenance that seemed to be scarcely anything but cap-border, and a little lean, long-fingered hand, always clenched under its chin.
It would lie in this attitude all day, with its bright specks of eyes open, wondering (as I used to imagine) how it came to be so small and weak.	It would lie in this attitude all day, with its bright specks of eyes open, wondering (I used to imagine) how it came to be so small and weak.	It would lie in this attitude all day, with its bright specks of eyes open, wondering (as I used to imagine) how it came to be so small and weak.	It would lie in this attitude all day, with its bright specks of eyes open, wondering how it came to be so small and weak.
Whenever it was moved it cried, but at all other times it was so patient that the sole desire of its life appeared to be to lie quiet and think.	Whenever it was moved it cried, but at all other times it lay quiet.	Whenever it was moved it cried, but at all other times it was so patient that the sole desire of its life appeared to be to lie quiet and think.	Whenever it was moved it cried, but at all other times it was so patient that the sole desire of its life appeared to be to lie quiet and think.

Table 15: Abridgements for an excerpt of Bleak House, Chapter 50

Self-training Reduces Flicker in Retranslation-based Simultaneous Translation

Sukanta Sen[1*] **Rico Sennrich**[2,1] **Biao Zhang**[1] **Barry Haddow**[1]

[1]School of Informatics, University of Edinburgh
[2]Department of Computational Linguistics, University of Zurich
ssen@ed.ac.uk, sennrich@cl.uzh.ch,
b.zhang@ed.ac.uk, bhaddow@ed.ac.uk

Abstract

In simultaneous translation, the *retranslation* approach has the advantage of requiring no modifications to the inference engine. However, in order to reduce the undesirable flicker in the output, previous work has resorted to increasing the latency through masking, and introducing specialised inference, thus losing the simplicity of the approach. In this work, we show that self-training improves the flicker-latency tradeoff, while maintaining similar translation quality to the original. Our analysis indicates that self-training reduces flicker by controlling monotonicity. Furthermore, self-training can be combined with biased beam search to further improve the flicker-latency tradeoff.

1 Introduction

Simultaneous machine translation systems, which process their input word by word instead of sentence by sentence, must strike a balance between producing output immediately (and so reducing quality because of incomplete input) and waiting for further input (and so increasing latency). Ideally, a good simultaneous translation system will provide a pareto-optimal tradeoff between quality and latency. A straightforward way of doing simultaneous translation is *retranslation* (Niehues et al., 2016), which has the advantage that it can be used with an unmodified machine translation (MT) inference engine, and can perform better than the alternative streaming-based approaches (Arivazhagan et al., 2020b). The disadvantage is that retranslation may change previous output causing *flicker*, leading to a poor user experience, and so flicker needs to be balanced with latency and quality.

We argue that flickering is caused by two different (but related) issues: (i) lexical instability of the translation – the system "changes its mind" as more source is revealed, swapping one word for another[1] and (ii) non-monotonicity of the translation – the system favours a non-monotonic translation, which means it needs high latency in order to avoid flicker. Some of this instability and non-monotonicity is necessary – forced by syntactic differences between source and target, and lack of information in the prefixes – but some is due to arbitrary choices of the model. We aim to reduce these as far as possible.

In non-autoregressive translation (NAT), a related problem, known as the "multimodality" problem (Gu et al., 2018), has been addressed using knowledge distillation (Kim and Rush, 2016, KD). We therefore investigate whether this can also reduce flicker in simultaneous translation. Since the initial model and the distilled model have the same architecture in our work, approximating KD is essentially self-training[2]. We show that a self-trained model is able to achieve the same quality as the initial model, but with improved flicker-latency tradeoff. We also show that self-training (Arivazhagan et al., 2020a) can be combined with biased beam search to further improve the flicker-latency tradeoff. Furthermore, we show experiments that link flicker to monotonicity.

2 Background

2.1 Retranslation

We assume a retranslation approach, where the source is retranslated each time it is updated, and the new output replaces the old. Only the current sentence is retranslated – previous sentences are considered to be fixed. In contrast to streaming approaches (e.g. Ma et al., 2019a; Arivazhagan et al., 2019b), retranslation can use an unmodified inference engine, making it simpler to deploy. The basic retranslation approach can be improved by using *prefix training* (Niehues et al., 2016, 2018), *bi-*

---

[*] Work done while at the University of Edinburgh.

[1] An example of this is shown in Appendix C.

[2] Retraining a model on its own output (Clark et al., 2003).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3734–3744
May 2-6, 2023

ased beam search and *output masking*[3] (Arivazhagan et al., 2020a).

2.2 Evaluation of Simultaneous Translation

In addition to quality, evaluation of simultaneous translation requires that we consider latency and, if using retranslation, flicker. The quality of the translation can be evaluated by comparing the final output of each sentence with a reference – we will use BLEU (Papineni et al., 2002; Post, 2018), CHRF (Popović, 2015) and COMET (Rei et al., 2020) scores. To measure flicker, we use *normalised erasure* (Arivazhagan et al., 2020a, 2019a), which measures the flicker between consecutive translation outputs by counting the minimum number of tokens that must be deleted from the end of the previous translation in order to produce the next, normalised by output length.

The measurement of latency has been the subject of some debate in the literature, with several different measures proposed (Ma et al., 2019a; Cherry and Foster, 2019; Ansari et al., 2021). In our experiments, we plot the flicker-latency tradeoff by controlling the output mask and recording the effect on flicker. Since mask size correlates with latency, our aim is to improve this mask-flicker tradeoff curve, and so be able to use a shorter mask with the same flicker budget.

2.3 Knowledge Distillation and Self-Training

The idea of sequence-level KD (Kim and Rush, 2016), is to create a smaller *student* model using the predictions of the larger *teacher* model. This has found application in MT efficiency (Junczys-Dowmunt et al., 2018) and in non-autoregressive translation (Zhou et al., 2020). In our work, the student model has the same size as the teacher, and is *self-trained* on teacher output. The output distributions of the student model have lower entropy (Zhou et al., 2020), so the model is less likely to swap between translation hypotheses unnecessarily as the source prefix is extended. Also, since the student model is trained on MT output, where the target order tends to be more similar to the source order (Zhou et al., 2020), it is more likely to avoid unnecessary reorderings, generating a more monotonic translation, which can be built up incrementally. We give experimental evidence for these in the next section.

Chen et al. (2021) also proposed to use pseudo-reference sentences obtained through forward translation of the source sentences to improve simultaneous translation. Unlike our work, they considered a streaming approach (specifically wait-k (Ma et al., 2019b)) where the system can only append to the output; it does not flicker like retranslation. They showed that their approach could improve the quality-latency tradeoff of wait-k using their distillation approach, but to create the training data for the student system they used wait-k and filtering. We avoid these complications by just using the baseline system as the teacher.

3 Experiments

3.1 Data

We test our self-training approach on English↔{German,Czech}. For En↔De we use IWSLT21 (Anastasopoulos et al., 2021) for training, and the concatenation of the 2014 and 2015 test sets for development (early stopping), removing any sentences that overlap with the training set. For En↔Cs, we use the training and validation set from WMT21 (Akhbardeh et al., 2021). Training data sizes are shown in Appendix A. We use *prefix training* (Niehues et al., 2018) to reduce the mismatch between sentence-level training and prefix-based inference at test time. For each parallel sentence pair in the training set, we generate a corresponding prefix pair by truncating a randomly chosen proportion. We treat the validation sets similarly.

We test our systems both on IWSLT test data (derived from TED talks) and on the ESIC test set[4] (Macháek et al., 2021). From IWSLT, we use tst2018 for De↔En, and tst2015/tst2016 combined for Cs↔En. ESIC is derived from the European parliament proceedings, and consists of transcribed speeches in English, together with their simultaneous interpretation into Czech and German (also transcribed). ESIC is aligned at the document level, but not at the sentence level. We use the test portion for evaluation, only for En→X. It has been argued that simultaneous translation is better evaluated (and trained, if possible) on interpreted data (Zhao et al., 2021). However such data is hard to come by, and ESIC is the only such resource for European languages. We remove any segments from the IWSLT test sets that overlap

---

[3] This means that the last k words are omitted from the output before being passed to the user. This reduces flicker, but increases latency.

[4] `https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-3719`

with training, and also remove from the training data any Europarl documents with overlap with ESIC.

All data is pre-processed with SentencePiece unigram model (Kudo and Richardson, 2018) with a shared subword (Sennrich et al., 2016b) vocabulary size of 32k.

		En→De		De→En	En→Cs		Cs→En
Metric	Model	ESIC	IWSLT	IWSLT	ESIC	IWSLT	IWSLT
BLEU	T	17.5	27.7	33.4	14.4	24.6	31.3
	S	17.6	27.5	31.7	14.5	25.0	31.3
ChrF	T	58.9	56.9	59.2	51.5	51.5	56.1
	S	58.8	57.2	58.3	51.7	51.7	56.2
COMET	T	.553	.330	.488	.651	.639	.519
	S	.532	.326	.468	.672	.642	.521

Table 1: Comparison between teacher (T) and student (S) models on ESIC and IWSLT test sets. For ESIC, BLEU and CHRF are calculated at document level, i.e. considering each document as a segment. For COMET we use reference-less `wmt20-comet-da` for ESIC and reference-based `wmt20-comet-da` for IWSLT.

3.2 Teacher-Student Training

Our teacher model, which serves as a baseline, is a transformer base (Vaswani et al., 2017) trained[5] with fairseq[6] (Ott et al., 2019).

We use the teacher to translate the training data, using a beam size[7] of 8, then train a student model with the same architecture on this synthetic data.

In Table 1 we show the performance of our baseline system (equivalent to the teacher) and the student system on 6 test sets. Overall, student performance is robust compared to teacher, with same or better scores in Cs↔En and some small losses in De↔En.

To assess whether the student models reduce flicker in retranslation, we use each model in a simulated SLT pipeline and plot flicker-latency tradeoff curves. That is, we use the systems to translate ever-growing prefixes of the source sentences in the testsets, using SLTev (Ansari et al., 2021) to measure the flicker, and varying the output mask to show the tradeoff. A curve for one test set is shown in Figure 1, with full results in Appendix D. We can see that in all configurations the student models improve the flicker-latency tradeoff. In section 3.4, we show how the student training data is more monotonic, and the models have lower entropy, echoing Zhou et al. (2020).

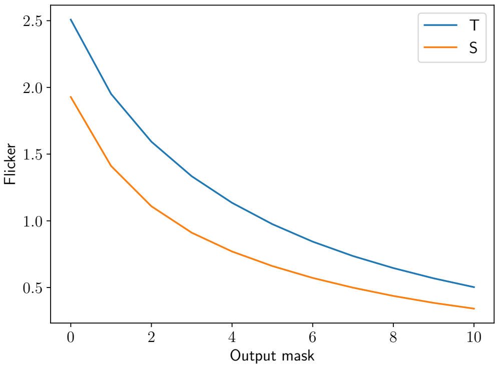


Figure 1: Flicker-latency tradeoff for the teacher (T) and student (S) models, En→De IWSLT. We control latency by varying the output mask.

3.3 Controlling Monotonicity

To show that self-training affects flicker through increased monotonicity, we experiment with controlling the monotonicity of the student training data. We stratify the teacher data into 5 different monotonicity levels using Kendall's Tau on a *fast_align* (Dyer et al., 2013) target–source alignment to measure monotonicity, with an equal stratum size. We add the monotonicity level as pseudo-word, as in Sennrich et al. (2016a), to each source sentence, and train a teacher model on this monotonicity-aware corpus. We then use this teacher to create 5 different student training corpora, using the monotonicity control, and train 5 different students on these corpora.

Table 2 shows the BLEU[8] scores for the monotonicity-controlled models, as well as the teacher and student from the previous section. Using highly monotonic (Mono-1) or non-monotonic (Mono-5) data gives poor quality, but the in-between strata are similar, with Mono-3 slightly better overall. Figure 2 shows a distinctly worse flicker-latency tradeoff for Mono-5, whereas Mono-4 is a bit better than the teacher, and all other students are better. This supports the hypothesized connection between the higher degree of monotonicity in the student training data, and the

---

[5] For training hyperparameters, see Appendix B.

[6] To generate training data for the students, we actually used a marian (Junczys-Dowmunt et al., 2018) model, with 60×10^6 parameters, trained on the same data and with the same architecture, which achieves nearly identical BLEU. This was to take advantage of marian's fast inference. All results shown in the paper are with the fairseq models.

[7] We also tested sequence-level interpolation, selecting the highest-scoring translation in an 8-best list according to BLEU and CHRF, but results were very similar.

[8] Scores for CHRF and COMET are in the Appendix F, but the pattern is similar.

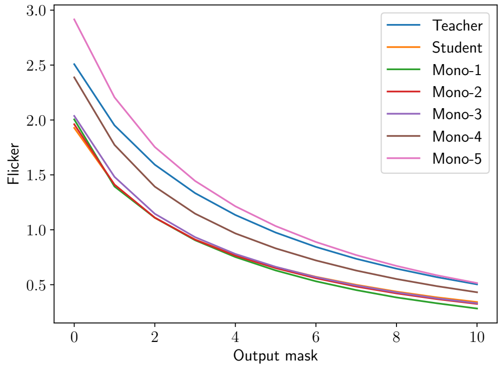


Figure 2: Latency–flicker tradeoff for the En→De IWSLT monotonicity-controlled models. Monotonicity control ranges from 1 (training data created with maximum monotonicity) to 5 (minimum monotonicity).

better flicker-latency tradeoff in the student models. We show the flicker-latency tadeoff curves on more test sets and language pairs in Figure 5 in the Appendix F but the pattern is similar.

		En→De		En→Cs	
Metric	Model	ESIC	IWSLT	ESIC	IWSLT
BLEU	Teacher	17.5	27.7	14.4	24.6
	Student	**17.6**	27.5	14.5	25.0
	Mono-1	8.6	14.4	**14.7**	23.6
	Mono-2	**17.6**	27.4	14.5	25.0
	Mono-3	17.5	**27.9**	14.5	**25.7**
	Mono-4	17.2	26.6	13.8	24.7
	Mono-5	16.0	25.0	12.5	23.0

Table 2: Student models with monotonicity control. Monotonicity ranges from 1 (highest) to 5 (lowest). The best scores are in bold font.

3.4 Monotonicity and Entropy of Student Models

We claimed that student models have lower flicker because they produce more monotonic translations, with less unnecessary variation. Here we provide evidence to support those claims.

Training data for student models is more monotonic In order to calculate the monotonicity of the training data, we use Kendall's tau score. We first extract word alignments from the training data using *fast_align* (Dyer et al., 2013) to forward-align source and target. For each sentence pair we express the alignment as a function $a : i \rightarrow j$, and construct the two lists $1, \ldots, T$ and $a(1), \ldots, a(T)$ where T is the target length. We then calculate the Kendall's tau between the two lists, repeat for each sentence pair in the corpus, and average. We repeat the calculation for the original training data and for the student training set. The results are shown in Table 3. We can see that in all cases, the student training data is more monotonic than the original teacher training data.

Model	En→De	De→En	En→Cs	Cs→En
Teacher	0.793	0.788	0.849	0.836
Student	0.857	0.801	0.906	0.880

Table 3: Kendall's tau scores. Higher scores indicate more monotonicity.

Student models have lower entropy distributions For each of our models, we calculate the mean per-token entropy, by considering the probability distribution over the vocabulary at each time step. The entropies are shown in Table 4.

		Entropy	
Pair	Test set	Teacher	Student
En→De	ESIC	0.371	0.220
	IWSLT	0.295	0.228
De→En	IWSLT	0.273	0.160
En→Cs	ESIC	0.443	0.251
	IWSLT	0.417	0.238
Cs→En	IWSLT	0.335	0.213

Table 4: Mean per-token entropies for each language pair test set combination.

We can see from Table 4 that the token entropies are consistently lower for student models, suggesting that the distributions are more "peaky", and so less likely to flicker between multiple output tokens with similar probabilities.

3.5 Self-training and Biased Beam Search

We investigate the combination of our self-training approach with biased beam search (Arivazhagan et al., 2020a). The idea of biased beam search (or "prefix biasing") is to reduce flicker in retranslation by modifying inference so that the translation of the current prefix is "biased" towards the translation of the last prefix. The model for inference has an extra term which penalises it for departing from the previous translation. As the current translation is being generated, once the hypothesis departs from the previous translation, we stop applying the bias penalty, reverting to the unmodified MT model.

Before the previous translation is used for biasing, it is normally *masked*; i.e., the right-most k tokens are removed. Without applying this mask, biased beam search seriously reduces quality by forcing inference to follow poor-quality early decisions. This *bias mask* is different from the output mask used in earlier experiments (which controls latency) although in previous work the bias and output mask are typically set to the same value.

We implemented biased beam search in fairseq and, based on previous work, we set the bias strength $\beta = 0.25$. After comparing different bias masks (Appendix E) we set the mask to 6 for ESIC and 10 for IWSLT.

We sweep across output masks to generate latency–flicker tradeoff curves in Figure 3 (with full results in Appendix E). We compare teacher and student models, with and without biased beam search. We can see from the graphs that biased beam search is effective in improving the latency–flicker tradeoff, but that the student models still improve over the teacher with biased beam search. The disadvantages of biased beam search are that it requires careful tuning of the prefix mask in order to avoid damaging quality, and that it requires a modified inference engine. The inference engine requires access to the previous translation, creating challenges for scalability. In contrast, our self-training approach requires no modifications to inference. Furthermore, since biased beam search relies on aligning the current translation with the previous one, it is hard to apply when the translation cannot be aligned – for example in a cascaded system where the ASR can rewrite its output.

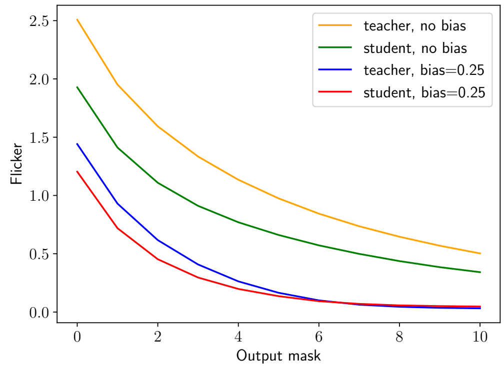


Figure 3: Latency-flicker tradeoff for teacher-student models with and without biased beam search for the En→De IWSLT.

4 Conclusion

We show that self-training reduces the flicker in retranslation-based simultaneous translation, whilst retaining quality. Our experiments link this flicker reduction to increased monotonicity and reduced entropy of the self-trained model. Although biased beam search can obtain larger reductions in flicker, it requires more careful parameter tuning, and a modified inference engine.

5 Limitations

Language Pairs We conducted our experiments using two European language pairs where source and target are linguistically similar. We show that we are able to reduce word order divergence between source and target text through forward translation which helps in reducing the flicker. However, a more challenging case will be using languages from different linguistic families with radically different word orders (such as English–Japanese) which may limit to which extent we are able to reduce the word order divergence between source and target through synthetic data creation.

Evaluation Whilst we show quality evaluation across three different metrics, we were not able to add human evaluation due to resource and space constraints. An additional consideration for simultaneous ST is that it is not clear what the combined effect of flicker, latency and quality is on human perception, and there has been limited work on this (Javorský et al., 2022).

Acknowledgements

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement 825460 (ELITR), and from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant number 10039436 UTTER]. RS acknowledges funding from the Swiss National Science Foundation (project MUTAMUR; no. 176727).

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–93, Online. Association for Computational Linguistics.

Antonios Anastasopoulos, Ondřej Bojar, Jacob Bremerman, Roldano Cattoni, Maha Elbayad, Marcello Federico, Xutai Ma, Satoshi Nakamura, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Alexander Waibel, Changhan Wang, and Matthew Wiesner. 2021. Findings of the IWSLT 2021 Evaluation Campaign. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 1–29, Bangkok, Thailand (online). Association for Computational Linguistics.

Ebrahim Ansari, Ondřej Bojar, Barry Haddow, and Mohammad Mahmoudi. 2021. SLTEV: Comprehensive evaluation of spoken language translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 71–79, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Te I, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2019a. Re-Translation Strategies For Long Form, Simultaneous, Spoken Language Translation. *arXiv e-prints*, page arXiv:1912.03393.

Naveen Arivazhagan, Colin Cherry, Te I, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020a. Re-Translation Strategies For Long Form, Simultaneous, Spoken Language Translation. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019b. Monotonic Infinite Lookback Attention for Simultaneous Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1313–1323, Florence, Italy. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020b. Re-translation versus streaming for simultaneous translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 220–227, Online. Association for Computational Linguistics.

Junkun Chen, Renjie Zheng, Atsuhito Kita, Mingbo Ma, and Liang Huang. 2021. Improving simultaneous translation by incorporating pseudo-references with fewer reorderings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5857–5864, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Cherry and George Foster. 2019. Thinking Slow about Latency Evaluation for Simultaneous Machine Translation. *arXiv e-prints*, page arXiv:1906.00048.

Stephen Clark, James Curran, and Miles Osborne. 2003. Bootstrapping POS-taggers using unlabelled data. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 49–55.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-Autoregressive Neural Machine Translation. In *Proceedings of ICLR*.

Dávid Javorský, Dominik Macháek, and Ondej Bojar. 2022. Comprehension of subtitles from re-translating simultaneous speech translation.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018. Marian: Cost-effective High-Quality Neural Machine Translation in C++. In *Proceedings of WNMT*.

Yoon Kim and Alexander M. Rush. 2016. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327. Association for Computational Linguistics. Event-place: Austin, Texas.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019a. STACL: Simultaneous Translation with Implicit Anticipation and Controllable Latency using Prefix-to-Prefix Framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and

Haifeng Wang. 2019b. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Dominik Macháek, Matú ilinec, and Ondej Bojar. 2021. Lost in Interpreting: Speech Translation from Source or Interpreter? In *Proceedings of Interspeech*.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic Transcription for Low-latency Speech Translation. In *Proceedings of Interspeech*.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. In *Proceedings of Interspeech*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. Type: Conference proceedings (article).

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR*, abs/1706.03762.

Jinming Zhao, Philip Arthur, Gholamreza Haffari, Trevor Cohn, and Ehsan Shareghi. 2021. It is not as good as you think! evaluating simultaneous machine translation on interpretation data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6707–6715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Training Data

Corpus	Sentence pairs
English-German	
Europarl	1.79 M
Rapid	1.45 M
News Commentary	0.35 M
OpenSubtitle	22.51 M
TED corpus	206 K
MuST-C.v2	248 K
English-Czech	
Europarl	645 K
ParaCrawl	14 M
CommonCrawl	161 K
News Commentary	260 K
CzEng2.0	36 M[9]
Wikititles	410 K
Rapid	452 K

B Training Parameters

The non-default hyperparameters for Fairseq are shown in Table 5.

C Example of Flicker

An example of a translation which flickers between two similar possibilities is shown in Table 6.

Param	Value
label-smoothing	0.1
criterion	label_smoothed_cross_entropy
patience	10
arch	transformer
optimizer	adam
adam-betas	0.9, 0.98
lr	5e-4
lr-scheduler	inverse_sqrt
warmup-updates	4000
clip-norm	0.0
weight-decay	0.0001
dropout	0.3
update-freq	2
max-tokens	3000
best-checkpoint-metric	bleu
maximize-best-checkpoint-metric	True

Table 5: Fairseq training hyperparameters (non-default) for 4 GPU training.

Source	I hope you will have a little time and energy to focus on another report which is, despite its technicality, quite important for all of us.
Target:	Ich
	Ich hoffe,
	Ich hoffe, Sie
	Ich hoffe, Sie
	Ich hoffe, Sie haben
	Ich hoffe, Sie haben ein
	Ich hoffe, Sie werden ein wenig Zeit
	Ich hoffe, Sie haben etwas Zeit
	Ich hoffe, Sie haben etwas Zeit und
	Ich hoffe, Sie werden etwas Zeit und Energie haben,
	Ich hoffe, Sie haben etwas Zeit und Energie, um sich
	Ich hoffe, Sie haben etwas Zeit und Energie, um sich auf
	Ich hoffe, Sie werden ein wenig Zeit und Energie haben, um sich auf ein anderes Thema
	Ich hoffe, Sie haben etwas Zeit und Energie, um sich auf einen weiteren Bericht zu konzentrieren,
	Ich hoffe, Sie haben etwas Zeit und Energie, um sich auf einen anderen Bericht zu konzentrieren,
	⋮
	Ich hoffe, Sie werden ein wenig Zeit und Energie haben, um sich auf einen anderen Bericht zu konzentrieren, der trotz seiner Formalität für uns alle sehr wichtig ist.

Table 6: Examples of flicker caused by the teacher model. *Source* is the original full sentence which is input as a growing input prefix. *Target* is the output prefix in successive retranslations.

D Flicker-Latency Tradeoff

In Figure 4, we show the flicker-latency tradeoff for all language-pair and testset combinations.

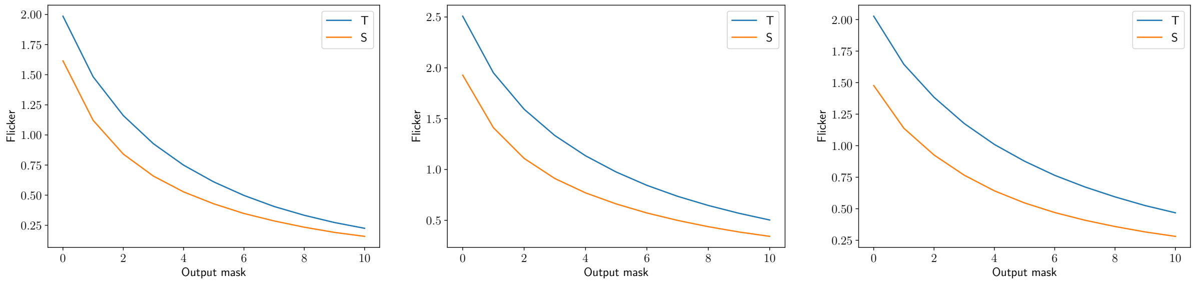
E Biased Beam Search

We consider the effect of the bias mask on full sentence translation quality, as measured by BLEU. The bias mask is measured in sentencepiece tokens. Based on Figure 6, we set the bias mask to 6 for ESIC and 10 for IWSLT, in order to avoid a loss of BLEU.

In Figure 7, we show flicker-latency tradeoffs for all language pair and testset combinations.

F Controlling Monotonicity

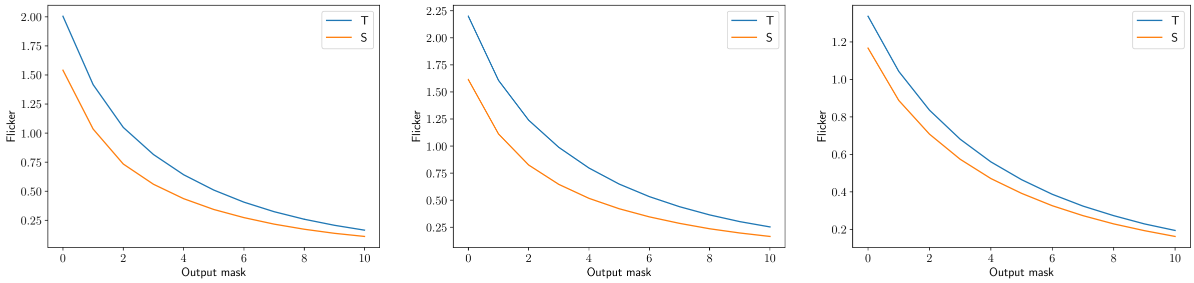
We have shown the scores for CHRF and COMET for monotonicity control experiments in addition to BLEU in Table 7. In Figure 5, we show the flicker-latency tradeoff for monotonicity-controlled experiments for all language-pair and testset combinations.



(a) En→De ESIC

(b) En→De IWSLT

(c) De→En IWSLT

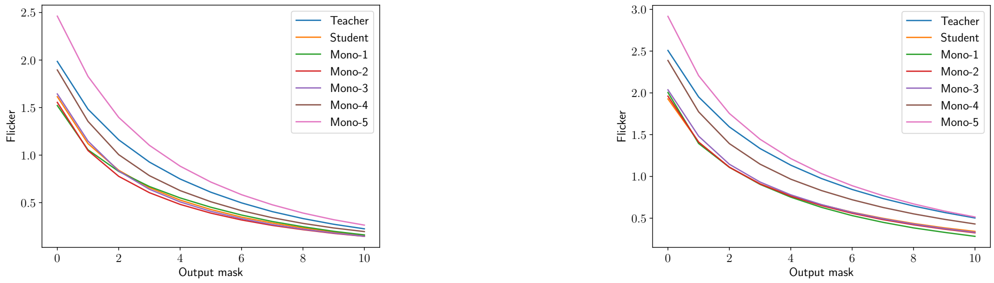


(d) En→Cs ESIC

(e) En→Cs IWSLT

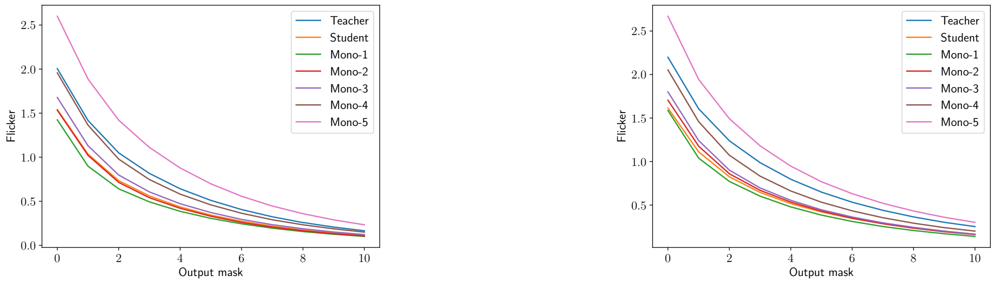
(f) Cs→En IWSLT

Figure 4: Flicker-latency tradeoff for the teacher-student models. We control latency by varying the output mask.



(a) En→De ESIC

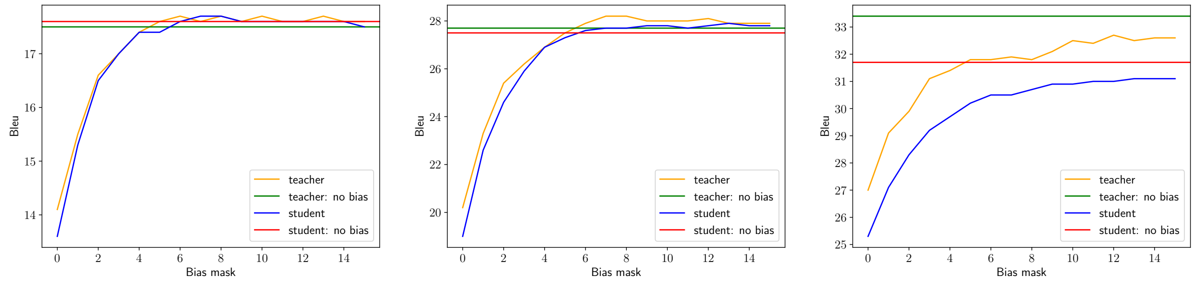
(b) En→De IWSLT



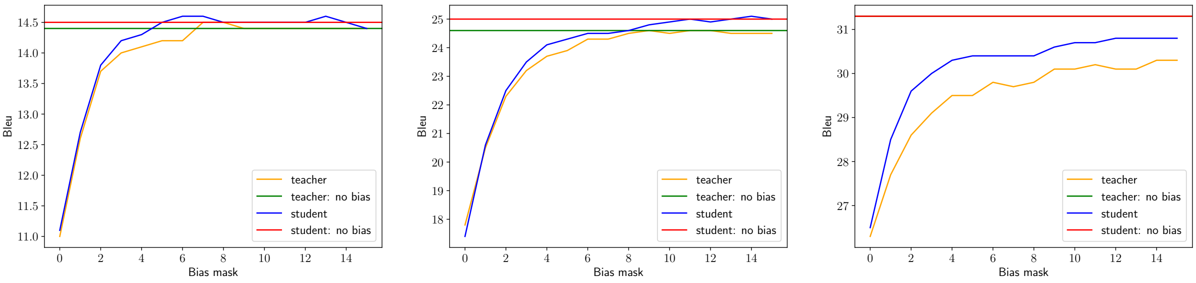
(c) En→Cs ESIC

(d) En→Cs IWSLT

Figure 5: Latency–flicker tradeoff for the monotonicity-controlled models. Monotonicity control ranges from 1 (training data created with maximim monotonicity) to 5 (minimum monotonicity).

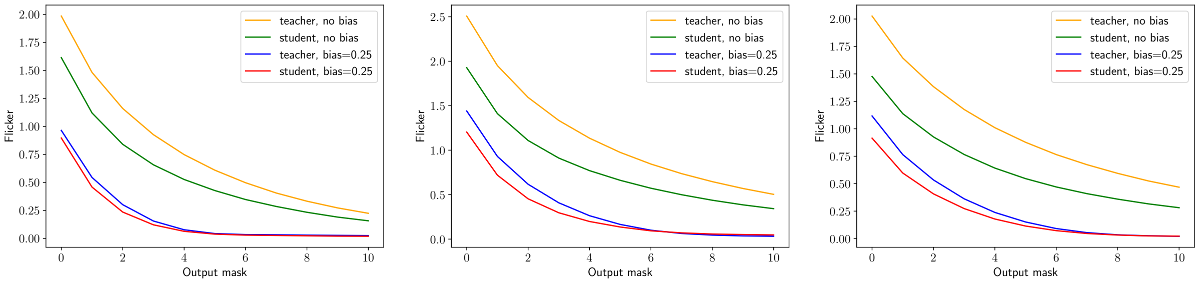


(a) En →De ESIC (b) En →De IWSLT (c) De →En IWSLT

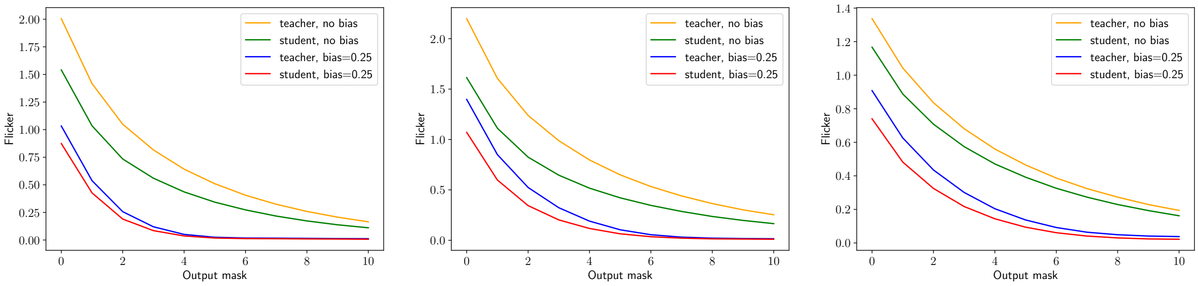


(d) En→Cs ESIC (e) En→Cs IWSLT (f) Cs→En IWSLT

Figure 6: Dependence of BLEU on bias mask when applying biased beam search.



(a) English → German ESIC (b) English → German IWSLT (c) German → English IWSLT



(d) English → Czech ESIC (e) Czech → English IWSLT (f) Czech → English IWSLT

Figure 7: Flicker vs mask on biased beam search.

		En→De		En→Cs	
Metric	Model	ESIC	IWSLT	ESIC	IWSLT
BLEU	Teacher	17.5	27.7	14.4	24.6
	Student$_{\text{model}}$	17.6	27.5	14.5	25.0
	Mono-1	8.6	14.4	14.7	23.6
	Mono-2	17.6	27.4	14.5	25.0
	Mono-3	17.5	27.9	14.5	25.7
	Mono-4	17.2	26.6	13.8	24.7
	Mono-5	16.0	25.0	12.5	23.0
ChrF	Teacher	58.9	56.9	51.5	51.5
	Student$_{\text{model}}$	58.8	57.2	51.7	51.7
	Mono-1	42.4	39.6	51.3	50.7
	Mono-2	58.7	57.3	51.8	52.0
	Mono-3	59.0	57.8	51.7	52.2
	Mono-4	59.0	56.8	51.4	51.4
	Mono-5	58.5	55.0	50.7	50.2
COMET	Teacher	.553	.330	.651	.639
	Student$_{\text{model}}$	.532	.326	.672	.642
	Mono-1	.510	-0.028	.639	.597
	Mono-2	.526	.295	.650	.636
	Mono-3	.530	.326	.678	.641
	Mono-4	.535	.313	.677	.639
	Mono-5	.518	.247	.633	.577

Table 7: Full results of student models with monotonicity control.

Social Commonsense for Explanation and Cultural Bias Discovery

Lisa Bauer **Hanna Tischer** **Mohit Bansal**
UNC Chapel Hill
{lbauer6, hannalt, mbansal}@cs.unc.edu

Abstract

Social commonsense contains many human biases due to social and cultural influence (Sap et al., 2020; Emelin et al., 2020). We focus on identifying cultural biases in data, specifically causal assumptions and commonsense implications, that strongly influence model decisions for a variety of tasks designed for social impact. This enables us to examine data for bias by making explicit the causal (if-then, inferential) relations in social commonsense knowledge used for decision making, furthering interpretable commonsense reasoning from a dataset perspective. We apply our methods on 2 social tasks: emotion detection and perceived value detection. We identify influential social commonsense knowledge to explain model behavior in the following ways. First, we augment large-scale language models with social knowledge and show improvements for the tasks, indicating the implicit assumptions a model requires to be successful on each dataset. Second, we identify influential events in the datasets by using social knowledge to cluster data and demonstrate the influence that these events have on model behavior via leave-K-out experiments. This allows us to gain a dataset-level understanding of the events and causal commonsense relationships that strongly influence predictions. We then analyze these relationships to detect influential cultural bias in each dataset. Finally, we use our influential event identification for detecting mislabeled examples and improve training and performance through their removal. We support our findings with manual analysis.

1 Introduction

Social commonsense knowledge helps humans maneuver through everyday life, aiding in situations that may require social nuance or cultural knowledge. Commonsense knowledge acquisition has been an important goal in NLP (Levesque et al., 2012; Davis and Marcus, 2015; Talmor et al., 2019) and in the past few years there has been a surge of research focusing specifically on improving social commonsense understanding for neural models (Sap et al., 2019a,b; Hwang et al., 2020; Forbes et al., 2020). However, social commonsense may contain many human biases due to social and cultural influence (Sap et al., 2020; Emelin et al., 2020). We aim to discover influential cultural biases in social applications datasets via the social and causal commonsense knowledge (Roemmele et al., 2011; Luo et al., 2016; Ponti et al., 2020) present in each dataset, to identify bias in cause-effect commonsense relationships. We define cultural bias as a positive or negative cultural attitude toward a social structure (see Section 3.1) and define causal commonsense following Sap et al. (2019a), as if-then, inferential relations. Specifically, we are interested in exploring biases towards the following social structures: *religion, economy, family, government, education* and *technology*. To the best of our knowledge, we are the first to empirically discover influential cultural biases in social tasks, using causal commonsense knowledge about social interactions, emotional reactions, and human needs to explain underlying cultural trends in social applications datasets.

We focus on two social tasks: emotion detection in social media and community value detection from interviews. We define a social task as any task that is intended to have social impact and requires social knowledge to correctly resolve. It is particularly important to identify influential biases in datasets and applications which are designed for social impact. HurricaneEmo (Desai et al., 2020) is an emotion detection task that focuses on perceived emotional reactions to events that occur during natural disasters. For example, events that exist in this dataset like *"thanks god"* may elicit emotions like AWE, and events like *"PersonX sends – to congress"* may elicit emotions like CONTEMPT, showing a positive cultural bias for reli-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3745–3760
May 2-6, 2023

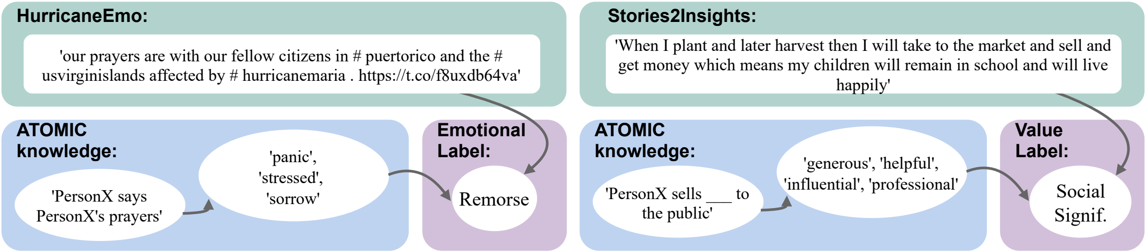


Figure 1: Examples of Social Tasks and Knowledge.

gion and a negative cultural bias for government. Stories2Insights (Conforti et al., 2020) is a value detection task that focuses on interviews conducted to identify community needs in developing countries, aiming to capture perceived values based on certain events. For example, events like *"reading the bible"* indicate INDIGENOUS values (defined by social norms and religion) whereas events like *"providing for children"* indicate INTRINSIC HUMAN needs (defined by health and quality of life). This illustrates a positive culture bias for Christianity and family. We aim to discover causal social knowledge that indicates cultural biases in each dataset, by analyzing model behavior on both tasks to gain a dataset-level understanding of events that influence performance.

We identify these biases via casual social knowledge, which encodes the relationships between events and the triggered reactions. We derived our knowledge from the social knowledge graph ATOMIC (Sap et al., 2019a). Consider the HurricaneEmo example in Fig. 1, in which the ATOMIC knowledge makes explicit the event-emotion causal relationships in the tweet that bring about the perceived emotion REMORSE. Identifying that the event *"says prayers"* causes perceived traits like *sorrow* and *stressed*, allows us to understand which events in the tweet cause the perceived emotion and contribute to the specific cultural biases. Next, consider the Stories2Insights example, in which the knowledge demonstrates the causal relationships that support the perceived value SOCIAL SIGNIFICANCE (defined by identity and status). Identifying that the event *"sells — to the public"* causes perceived traits like *influential* and *professional*, sheds light on how the event's associated cultural biases contribute to the perceived value.

In this paper, we gain a dataset-level understanding of influential causal social commonsense relationships, allowing an exploration of the underlying social and cultural biases that explain model behavior on social tasks. To this end, we first extract ATOMIC knowledge for each datapoint, utilizing a combination of TF-IDF and BERTScore (Zhang et al., 2019). We then append different components of this knowledge to the input, to focus strongly on either the cause, the effect, or the causal relation between the two. We use BERT-AUG, which ingests knowledge augmented data as input to BERT (Devlin et al., 2018), to yield improvements on both tasks. These improvements illustrate that the underlying causal social assumptions made explicit by the integrated knowledge do indeed increase model accuracy on the task and thus augment a model's understanding of the task. To discover underlying cultural biases, we investigate the influence of underlying events in each task. We first identify underlying events in the data by clustering datapoints using k Nearest Neighbors (kNN) around commonsense events, using fine-tuned contextual embeddings. We then identify which of these events are influential by performing leave-K-out experiments, which detect the influence of train clusters on task performance.

Understanding cultural biases in social applications data via the implicit commonsense knowledge present in the data is important for analyzing the limitations of the dataset and the respective tasks. By making explicit the underlying cultural assumptions and causal relationships, we are able to identify the biases that data from a certain source may have, which is paramount when using this data in the development of technology for other applications (Bender and Friedman, 2018).

Finally, we use our methods to identify mislabeling in event clusters. Datasets used for training deep learning models are large and often contain noisy labels, even if the data was collected via crowdsourcing (Frénay and Verleysen, 2013; Rajani et al., 2020). We identify events whose train clusters cause performance improvement when removed in leave-K-out experiments, indicating mis-

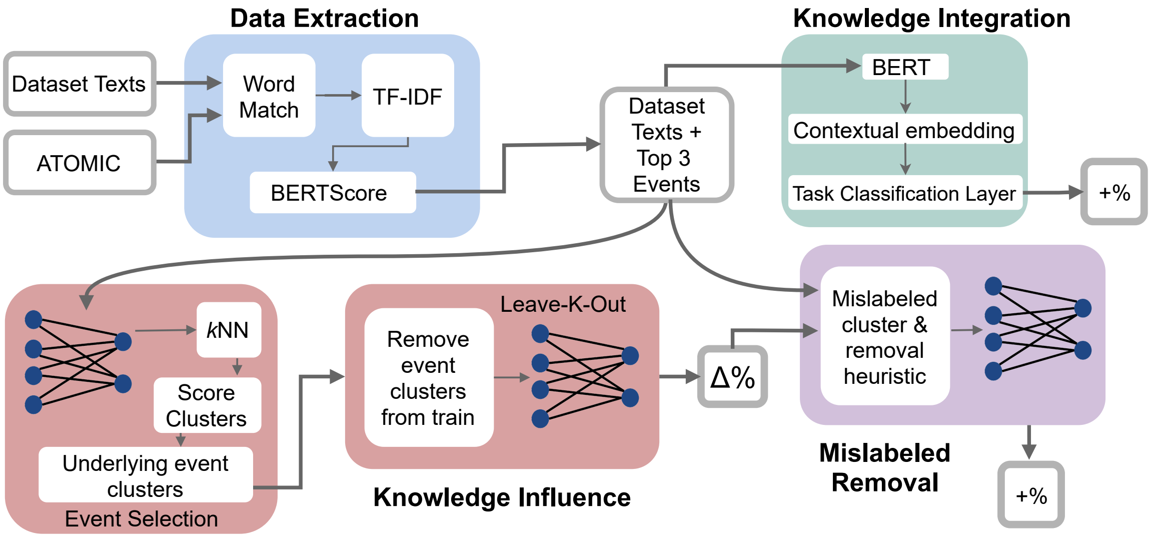


Figure 2: Data Extraction, Knowledge Integration, Knowledge Influence, and Mislabeled Removal.

labeling. Overall, our contributions are:

- We improve HurricaneEmo & Stories2Insights performance when we augment models with ATOMIC social knowledge, indicating the implicit assumptions a model requires to be successful on each dataset.
- We find representative events in each of the datasets via ATOMIC and *kNN*, and use leave-K-out experiments to discover causal social commonsense relationships that strongly influence model behavior.
- We analyze the influential cultural biases for different social structures and strongly suggest a cultural bias analysis for train data.
- We demonstrate that our methods can be used to identify mislabeled examples in the dataset.

2 Related Work

2.1 Social Applications

We explore tasks for social applications, specifically perceived emotion detection and perceived value detection. Emotion prediction has been studied for many different domains (Strapparava and Mihalcea, 2007; Katz et al., 2007; Ezhilarasi and Minu, 2012; Chen et al., 2018; Mohammadi et al., 2019), and has been extensively applied to social media posts (Mohammad, 2012; Wang et al., 2012; Mohammad and Kiritchenko, 2015; Abdul-Mageed and Ungar, 2017), particularly in the social good domain. Sharifirad et al. (2019) performed emotion classification on sexist tweets and Sanders et al. (2021) analyzed sentiment in tweets during the early COVID-19 pandemic. Similar to our disaster-related application, Lin et al. (2018) used semantic matching to discover disaster recovery trends in large text corpora. In this paper, we focus on implicit perceived emotion prediction, which requires models to capture context and perform reasoning about perceived emotions (Desai et al., 2020), rather than intended emotions. Desai et al. (2020) released an emotion prediction dataset, focusing on tweets related to Hurricanes Irma, Harvey, and Maria. Hirmer and Guthrie (2016) focused on User-Perceived Values (UPVs), particularly concentrating on the needs and values of project beneficiaries in developing countries. More recently, Conforti et al. (2020) analyzed emotion as user perceived values in statements made by Ugandan rural individuals.

2.2 Social Causal Commonsense Reasoning

Commonsense reasoning has been a long-standing challenge in NLP (Levesque et al., 2012; Davis and Marcus, 2015; Talmor et al., 2019) and more recently, social commonsense reasoning has gained popularity (Rashkin et al., 2018; Nematzadeh et al., 2018; Talmor et al., 2019; Sap et al., 2019b; Hwang et al., 2020; Forbes et al., 2020; Sap et al., 2020; Emelin et al., 2020), with a strong focus on social, moral, and cultural understanding and norms. We further this work by extracting causal relations for social and cultural norms and integrating this knowledge to understand biases and model explanation in social tasks. Similar to our integration methods, Chang et al. (2020) implicitly and explicitly incorporates social knowledge from ATOMIC

(Sap et al., 2019a) and ConceptNet (Speer et al., 2017) into a social commonsense reasoning task (in contrast to event-driven tasks with real-world impacts), yielding performance gains. We, instead, focus on a causal dimension of ATOMIC and use knowledge to both improve and explain both perceived emotion and value detection. Causal commonsense reasoning has also been explored for a variety of tasks, focusing on identifying event causality in social media (Sil et al., 2010; Riaz and Girju, 2013; Kayesh et al., 2019, 2020b,a), commonsense causal QA (Roemmele et al., 2011; Luo et al., 2016; Hassanzadeh et al., 2019; Ponti et al., 2020), and conversational emotion recognition (Ghosal et al., 2020). We, instead, focus on social tasks and use knowledge to gain a dataset-level understanding of influential events.

2.3 NLP Interpretability

Our analysis methods are related to influence functions (Koh and Liang, 2017), which have been recently extended to neural text classifiers in NLP (Han et al., 2020). Our methods share particular similarity with group influence functions, which identify an influential group of training examples in a particular test prediction (Basu et al., 2020). We largely differ by using a heuristic for clustering events and verifying the influence of each cluster's train data on the cluster's dev data, allowing a dataset-level analysis of influential social relationships for text classification. Similar to our work, Rajani et al. (2020) proposed using a *k*NN framework to gain a dataset-level understanding of model behavior by identifying training examples responsible for NLI predictions. In contrast, we identify influential causal social commonsense relationships.

3 Tasks & Datasets

3.1 Definition of Cultural Bias

We define cultural bias as a positive or negative cultural attitude toward a certain social structure. Thus, positive and negative bias refer to the sentiment of the particular commonsense relation toward a social structure in the text. This is encoded differently for the two datasets that we use throughout this work. HurricaneEmo has labels for positive or negative perception (e.g., contempt = negative, love = positive). However, the target of the cultural bias depends on the subject of the text and the intended target of the emotion, thus we stress the importance of analyzing the automatically retrieved instances manually (see list of influential events in Table 4 and datapoints in Table 5). We also consider the respective commonsense relation in order to understand whether the sentiment is positive or negative. On the other hand, Stories2Insights does not use sentiment-based labels, but instead focuses on different categories of values, indicating a positive-only cultural attitude/bias toward the topic of the text. Therefore, we consider all Stories2Insights labels as positive with respect to cultural attitude.

It is crucial to note that it is not the intention of this work to draw conclusions about various cultures that these datasets may derive from. The cultural attitudes we discover in a single dataset are not an accurate reflection of the entire culture that this dataset may derive from and we would find this conclusion to be particularly harmful. We instead aim to explore biases with respect to social structures in a particular dataset and are only able to discover and examine cultural attitudes that are particular and limited to the target dataset and are additionally limited by our knowledge recall. See Section 8 for more details.

3.2 HurricaneEmo

For emotion classification, we use HurricaneEmo (Desai et al., 2020). This dataset was constructed from 15,000 English tweets about Hurricanes Irma, Harvey, and Maria. Through crowd-sourcing, each tweet was classified based on the 24 Plutchik emotions (Plutchik, 2001), and then summarized into eight emotions: AGGRESSIVENESS, AWE, CONTEMPT, DISAPPROVAL, LOVE, OPTIMISM, REMORSE, and SUBMISSION. These were split into eight binary classification tasks. For example, this tweet in the LOVE binary classification task, *"to my friends offering that support during the hurricane, i thank you. we are safe and sound. https://t.co/yl8wdbhi4"* is labeled positive. See Section A.1 in the appendix for dataset construction and size.

3.3 Stories2Insights

For Automatic User-Perceived Value classification, we obtain the Stories2Insights (Conforti et al., 2020) corpus, consisting of labeled (English) interviews from villages in Uganda. Each statement in the dataset is labeled with User-Perceived Values (UPVs), and the data is divided into six different labels: EMOTIONAL, EPIST, FUNCTION, INDIGENOUS, INTRINSIC HUMAN, and SOCIAL SIGNIFICANCE (Conforti et al., 2020). We used these labels to create binary datasets from the original data. For example, *“Also my children and my husband will get entertained and be happy."* is labeled as EMOTIONAL. See Section A.2 in the appendix for dataset construction and size.

3.4 ATOMIC

We utilize the causal social commonsense knowledge in the knowledge graph ATOMIC (Sap et al., 2019a), a graph for if-then reasoning, connecting events through one of nine different relation edges. In this paper, we focus on the "xAttr" edge, which describes the perceived attributes of an event’s subject. This edge allows us to extract perceived social knowledge, as it often covers the perceived emotion or value we are interested in for HurricaneEmo and Stories2Insights. For example, the ATOMIC event *“PersonX helps people"* uses the "xAttr" edge to show the perceived attributes of PersonX based on this event, i.e., PersonX is seen as *"kindhearted; incredible; pleasant; kind"*.

4 Models & Knowledge Augmentation

We aim to 1) extract knowledge that makes explicit the underlying causal social commonsense relationships in each datapoint and 2) propose a simple integration method to show that this knowledge is able to increase model accuracy, demonstrating the role of these relationships in improving a model’s understanding of the task.

4.1 Baseline

We finetune BERT base (Devlin et al., 2018) as a baseline for both datasets. We input the text (tweet or statement) to BERT to obtain contextual embeddings, which we then project with a weight matrix $\mathbf{W} \in \mathbb{R}^{d\times 2}$.

4.2 Knowledge Integration

We augment the BERT baseline with knowledge, which we call BERT-AUG. We illustrate this process in the *Knowledge Integration* section of Fig. 2. Each input consists of text from the task and k (where k=3) extracted ATOMIC events with attributes (the extraction process is described in Section 4.3). We directly append knowledge as text to the input text via 5 different methods. To illustrate these methods, consider the following top 2 events that may be extracted for a given tweet:

- "PersonX gets a warning" -> "unhappy", "behaved", "negative", "mischievous", "badly"
- "PersonX ignores the warnings" -> "reckless", "unworried", "confused", "dangerous", "unsafe", "careless"

Dataset	**%Event-Text**	**%Attribute-Label**
HEmo	58	54
S2I	77	65

Table 1: Data Extraction Manual Analysis. Event-Text is the match between the ATOMIC event and datapoints. Attribute-Label is the match between the respective ATOMIC attributes and the dataset label. Higher % indicates more matches.

Each integration method uses different components of the knowledge as follows, which are then appended to the input text:

- Method 1: all attributes of first event
  Example: "unhappy; behaved; negative; mischievous; badly"
- Method 2: first attribute of first event
  Example: “unhappy”
- Method3: first attribute of all events
  Example; “unhappy; reckless’
- Method4: first event + first attribute
  Example: "PersonX gets a warning: unhappy"
- Method5: first event + all attributes
  Example: "PersonX gets a warning: unhappy; behaved; negative; mischievous; badly

We determine the best method for each dataset as a hyperparameter when tuning BERT-AUG on the dev set.

4.3 Knowledge Extraction

We extract ATOMIC knowledge for both HurricaneEmo and Stories2Insights, illustrated in the *Data Extraction* section of Fig. 2. In this section, the tweets of HurricanEmo and the text of Stories2Insights are both referred to as the dataset D, as we follow the same extraction process for both datasets.

Following Lin et al. (2018), our extraction procedure is decomposed into two phases: one for speed and one for precision. The first phase serves as a coarse-grained filter to efficiently collect an initial pool of ATOMIC knowledge candidates P_i for each datapoint $d_i \in D$: if there is word overlap between d_i and a ATOMIC candidate event, the ATOMIC candidate is added to P_i. Next, we use TF-IDF

Model	AGR	LOV	SBM	AWE	DSP	CNT	RMR	AVG
BERT (Desai et al., 2020)	67.6	54.0	67.4	68.3	55.7	66.8	58.5	62.6
BERT	**75.6**	65.1	**72.8**	**73.7**	58.0	**77.9**	63.2	69.5
BERT-AUG	75.0	**68.8**	70.5	72.7	**64.4**	75.9	**65.2**	**70.4**

Table 2: HurricaneEmo Test Results (Accuracy), including aggressiveness (agr), love (lov), submission (sbm), awe (awe), disapproval (dsp), contempt (cnt), remorse (rmr), and average (avg) across all binary tasks. We recompute the baseline after extra data preprocessing. Best results are bold.

Model	Emotional	Epistemic	Functional	Indigenous	Intrinsic	Social	Avg
BERT	81.0	87.7	**84.6**	84.6	84.4	80.1	83.7
BERT-AUG	**86.9**	87.7	82.6	**89.4**	**87.2**	**82.8**	**86.1**

Table 3: Stories2Insights Test Results (Accuracy). Best results are bold.

to find the top n (we choose n=50) most similar ATOMIC candidates (considering ATOMIC event only) to d_i in P_i, and add these to a smaller pool S_i. Then, for precision we score each ATOMIC candidate in S_i with d_i for semantic match, using BERTScore, an evaluation metric for text similarity (Zhang et al., 2019). In this step, we use both the ATOMIC event and attributes to ensure that the causal social knowledge of each event is captured. We rank S_i by score and return the top k (where k=3) ATOMIC candidates for each datapoint.

To determine how precise this retrieval is, we manually examine the knowledge-datapoint matches for each dataset, demonstrated in Table 1. We analyzed 30 events from the dev set of each binary dataset (7 for HurricaneEmo, 6 for Stories2Insights; totaling 1260 datapoints), and examined matches between 1) the datapoint and ATOMIC event, and 2) the dataset label and ATOMIC attributes. This analysis investigated both whether 1) the ATOMIC events matched events in the datapoint and 2) the causal relationship between the event and its attribute is the same as the relationship between the datapoint and its label. We observe that ATOMIC matches both the events and causal relations in Stories2Insights better than those in HurricaneEmo. See Section A.4.1 in the appendix for more details.

5 Event Influence & Bias Discovery

We aim to (1) identify influential social and causal events in each dataset, and (2) analyze their respective inferential relationships to discover attitudes toward target social structures.

To identify influential events in the datasets, we first extract underlying ATOMIC events from each dataset via *event selection* and then measure how influential these events are via *leave-K-out* experiments (illustrated in *event selection* and *leave-K-out* in the *Knowledge Influence* section of Fig. 2). We then explore the following types of social and cultural structures in our datasets: *religion, economy, family, government, education* and *technology*. We consider a dataset to be biased if there exists a positive or negative attitude toward one of these structures in the influential underlying events.

5.1 Event Selection

We select underlying events in each dataset that potentially influence model behavior. To identify these events, we first fine-tune a separate BERT model for each binary dataset. Then, we represent the binary train data, dev data, and ATOMIC events with contextual embeddings using the respective BERT model for that dataset. Next, for each datapoint $d_i \in D$, we find its closest event e_j (where $e_j \in E$ and E denotes the ATOMIC events) using kNN. We then assign d_i to c_j, where c_j is the respective data cluster for e_j ($c_j \in C$ and C denotes all the clusters). If d_i is in train, we assign it to c_j^t (the event-specific train cluster), and if d_i is in dev, we assign it to c_j^d (the event-specific dev cluster). Thus, for each binary dataset, we have clusters of train and dev data for each ATOMIC event. This process is denoted as *kNN* in Fig. 2.

To identify the most representative underlying ATOMIC events learned by the model, we use the clusters of the top m (where m=50) most common ATOMIC events extracted for the train set (see Section 4.3 for details), and thus each binary dataset in HurricaneEmo and Sight2Insights has m clusters. To further identify which events have been learned best by the model, we then use cross-validation to determine which clusters have the highest prediction accuracy (thus the dev set is not used to score clusters during event selection). We score clusters by their average prediction accuracy across each fold and select events whose clusters score highest. This process denoted as *Score Clusters* in Fig. 2. We use the top t (where t=5) events for our leave-K-out experiments.

5.2 Leave-K-Out

We evaluate whether the selected underlying events have a strong influence on model behavior (see *leave-K-out* in Fig. 2). Recall that each event e_j in a particular binary dataset has a cluster of event-specific datapoints c_j^t and c_j^d for train and dev. For each event e_j, we remove c_j^t from the train set and use c_j^d as an evaluation set. We compare results on c_j^d between a BERT baseline (discriminator trained on all train data in that binary dataset) versus BERT trained on the train set with c_j^t removed. We evaluate accuracy and look for prediction changes. We also include ablations that remove the same amount of randomly selected train data to establish a lower bound. See Section A.5.1 in the appendix for data sizes and a thorough discussion and interpretation of the respective results.

6 Results

6.1 Knowledge Integration

We examine knowledge integration in Table 2 and Table 3. Due to our additional HurricaneEmo data preprocessing (see Section A.1 in the appendix), we reran BERT baselines on this dataset. Overall, Stories2Insights has better performance improvements than HurricaneEmo.

For HurricaneEmo, we see visible improvement for LOVE, DISAPPROVAL, and REMORSE, which performed best using integration methods 2, 5, and 3, respectively, all of which are attribute focused. For Stories2Insights, we see visible improvements for datasets where the label is focused on social knowledge: EMOTIONAL, SOCIAL SIGNIFICANCE, INTRINSIC HUMAN, and INDIGENOUS, using integration methods 1, 3, 4, and 5, respectively.

6.2 Knowledge Influence

We find ATOMIC events that influence model performance in Table 4. Given the top t dataset events from event selection in Section 5, we demonstrate results for the events for which a change in performance occurred, i.e., the influential events (see Section A.5.2 in the appendix for the full list of events). We see that for all Stories2Insights events, and most events in HurricaneEmo, removing c_j^t decreases performance on c_j^d. Our ablations indicate a lower bound and thus highlight the cases where decreased performance is indeed significant. We underline all influential results that perform worse on c_j^d than both the baseline and ablation when the respective c_j^t is removed. These events are able to cluster highly relevant text and thus the c_j^t examples have strong prediction influence on c_j^d. See Section A.5.1 in the appendix for further discussion and interpretation of the results.

We observe some cases in HurricaneEmo where best performance is achieved by removing c_j^t, indicating mislabeling. We manually analyze the five HurricaneEmo event clusters in question. For each datapoint in c_j^d where the prediction was corrected after leave-K-out, we retrieve the removed c_j^t datapoints. We find that the train and dev manual label agreement between the two sets is 49.33%, averaged across all clusters. As the labels should agree, this indicates mislabeling and explains the performance improvements after removal (verified by the manual analysis in Section 6.4). Finally, there are cases in which the ablation performs worst. This seems related to the nature of the c_j^ds in question, such that the model is uncertain about these examples and thus predictions on this subset are highly sensitive to any changes in the train set.

We performed manual analysis to verify c_j (event-specific data) semantic similarity with e_j (the event) and also c_j^t and c_j^d (event-specific dev and train) similarity with each other. Two annotators analyzed approximately 190 datapoints across 13 different event clusters (with a Cohen kappa of 0.81 which is considered as 'almost perfect agreement' [1], see Section A.4.2 for more details). Table 6 shows that for HurricaneEmo, 87% of the event-specific data semantically match each other, and close to half of dev and train examples match the cluster's event. We also observe that for Stories2Insights, 100% of the event-specific data semantically match. Overall, the integration improvements, leave-K-out results, and final influential events suggest that ATOMIC represents a better cultural and social match for Stories2Insights, and allow us to clearly identify influential events in Stories2Insights. This is supported by our man-

---

[1] `https://en.wikipedia.org/wiki/Cohen%27s_kappa`

Dataset	Event	Full Train	Leave-K-Out	Ablation
HurricaneEmo				
AGR	PersonX drives from florida	**76.9**	74.4	76.9
LOV	PersonX checks the weather forecast	**60.4**	39.6	60.4
	PersonX uses — to avoid	40.0	**45.0**	35.0
SBM	PersonX sends — to the congress	82.4	**88.2**	76.5
AWE	PersonX prepares for the storm	60.6	**66.7**	60.6
	PersonX is still valid —	**69.2**	38.5	61.5
DSP	PersonX practices — in the state	57.1	**71.4**	57.1
	PersonX keeps PersonY in PersonY's prayers	**100.0**	50.0	100.0
CNT	PersonX sails close to the wind	**79.5**	74.4	79.5
RMR	PersonX crosses my heart and hope to die	82.3	**85.5**	80.7
	PersonX sends — to the congress	**69.1**	66.7	66.7
	PersonX keeps PersonY in PersonY's prayers	**80.0**	60.0	60.0
Stories2Insights				
Indigenous	PersonX offer — to the gods	**66.7**	33.3	66.7
	PersonX reads the bible	**87.5**	75.0	75.0
Intrinsic	PersonX provides — to children	**100.0**	80.0	100.0
Social	PersonX protects — from the effects	**71.4**	57.1	42.9
	PersonX sells — to the public	**88.2**	67.7	88.2

Table 4: Leave-K-Out Results. Best performance is bold and performance of most influential events is underlined.

Clustered Datapoints by Event
HurricaneEmo: PersonX prepares for the storm
even san antonio evac centers could get more than 1' of rain. tx gov. abbott suggests going farther inland to austi https://t.co/jsxrwwiu3y
puerto rico rations resources as hurricane maria approaches - https://t.co/ypziieca7a https://t.co/lzw1inirsu
cnn reports miami international airport & fort lauderdale-hollywood international airport are closed. latest updat https://t.co/repbpqdqsz
to all ga residents in hurricane irma 's path, stay safe & be careful! for shelter information, please visit: https://t.co/acytlq9orr
Stories2Insights: Person provides — for the children
We have many diseases which attack us at any including our children, so if medicines are around, we can always treat ourselves and do things which can bring for us money and our children will go to school and learn.
Motorcycle will help me to take children for treatment when they fall sick and also I can be taken for treatment using the motorcycle.
Chicken is good to have at home since it lays eggs which I use it in feeding my children.

Table 5: Examples of HurricaneEmo and Stories2Insights datapoints in ATOMIC event clusters.

Dataset	%dev	%train	%dev-train
HEmo	53	43	87
S2I	50	79	100

Table 6: HurricaneEmo and Stories2Insights Clustering Manual Analysis. %dev and %train show the match between dev/train clusters and the event. %dev-train shows the match between dev and train clusters. Higher % indicates more matches.

ual analysis which indicates that Stories2Insights events and causal relations are better captured by ATOMIC and that by using ATOMIC knowledge, we are able to get semantically matching clusters.

6.3 Bias Analysis

We examine bias with respect to the following cultural structures: *religion, economy, family, government, education* and *technology*. For HurricaneEmo, we find strong cultural biases for *religion* and *technology*. In Table 4, we see that religion is associated with several causal reactions, specifically disapproval and remorse. This may be due to the use of religion in text, marking particularly traumatic situations. We also observe that there is a general mixed reaction toward *technology*, with "driving" associated with perceived aggression and "checking weather forecast" associated with perceived love (often in the context of rapid information spread during a disaster). Finally, it seems that there is a negative bias towards *government*, in which references to "congress" tend to elicit remorse. Interestingly, the *economy*, *family* and *education* seem relatively non-influential.

For Stories2Insights, we see that the mention of *religion* is both influential and strongly associated with INDIGENOUS values (defined by social norms and religion). "Providing for children" is also a very influential event, demonstrating a strong

Model	LOV	DSP	SBM	AWE	RMR
Baseline	65.1	58.0	72.8	**73.7**	63.2
Reduced Train	**68.4**	**61.1**	**73.2**	72.1	**66.4**
Number Examples Removed	4	13	8	97	11

Table 7: HurricaneEmo test set performance after removing potentially mislabeled train datapoints.

bias for *family* in the dataset, which elicits the INTRINSIC HUMAN value, associated with health and quality of life. This sheds light on how attitudes toward religion, family, and childcare are valued in positive ways within this corpus, and indicates how these structures may play strong roles in values associated with social norms and quality of life. Finally, the *economy* also seems to be quite influential, as we see that "PersonX sells — to the public" is strongly associated with SOCIAL SIGNIFICANCE (i.e., identity, status) illustrating that the economy plays a strong role in this value.

Given the performance improvements when adding social causal knowledge and the discovered influential cultural biases described above, we see that these datasets contain implicit assumptions that, when acquired, improve a model's performance on each dataset. In particular, there are several influential cultural biases in the dataset that may be harmful when generalizing to another task. For example, the importance and meaning of *religion* may be different depending on the task. In HurricaneEmo, *religion* plays a major role as a reaction to traumatic events, whereas in Stories2Insights it plays a role in indigenous values. We observe that the context of data collection (e.g., natural disaster tweets, perceived value collection, etc) is particularly important in the type of attitudes towards social structure a dataset might encapsulate, and thus recommend this type of analysis to better understand implicit bias held in a dataset based on its application. We want to strongly emphasize that is very important to properly analyze the implicit cultural biases for any train set before applying a model trained on this dataset.

6.4 Detecting Mislabeled Events

Finally, we leverage our analysis to identify mislabeled datapoints in HurricaneEmo and improve our performance on the full test set. We use all datasets that contain events where the best performance on the target c_j^d is achieved by training on the removed train set, as illustrated in Table 4. This suggests that the c_j^t have been mislabeled and are negatively affecting c_j^d, which we have confirmed with manual analysis (see Section A.4.3 for more details). We refer to these events as mislabeled events. To mitigate this, we use the mislabeled events in a removal heuristic, where we remove all examples that extract the mislabeled events as their highest scoring event. We then evaluate our model on the full test set, see Table 7. Every considered dataset demonstrates improvement over the BERT baseline on the full test set, except AWE, most likely due to the large number of removed examples for AWE, which may interfere with the predictions of other datapoints.

7 Conclusion

We used causal social commonsense knowledge to discover influential events and relationships that explain model behavior and pinpointed instances of cultural bias. First, we found that using large-scale language models augmented with causal social knowledge improved our social classification tasks, illustrating that the knowledge made the underlying social assumptions in the dataset explicit. Then, we identified underlying events in each dataset by clustering data around ATOMIC knowledge, to pinpoint cultural biases that the dataset may exhibit. We found that some of this knowledge strongly influenced model behavior through leave-K-out experiments, providing a dataset-level understanding of influential events and causal social commonsense relationships and allowing an analysis of the datasets' implicit influential cultural biases. Finally, we used these underlying and influential events to identify mislabeled train examples and thus improve training and performance.

Acknowledgements

We thank the reviewers for their useful feedback. This work was supported by DARPA MCS Grant N66001-19-2-403, NSF-CAREER Award 1846185, and an NSF Graduate Research Fellowship. The views are those of the authors and not of the funding agency.

8 Ethical Considerations & Limitations

We first address ethical considerations and limitations with respect to potential biases in our methods and resources, and then ethical considerations with respect to data use.

Bias: Emotions and values, and their expression and perception, are not universal. We use ATOMIC as, at the time of this work, it is the largest English source of social commonsense knowledge, however it is important that we further the creation and use of resources that are not limited to Western norms, developed countries, and the English language, especially when applying them to data outside of these domains. We anticipate that a knowledge graph better suited for representing different cultural attitudes will yield more coverage. For example, the events in ATOMIC may not cover important or representative cultural events in Uganda (the source of our second dataset), as it largely contains Western-centric social and cultural norms found in mostly developed countries.

It is also crucial to note that it is not the intention of this work to draw conclusions about various cultures that these datasets may derive from. The cultural attitudes we discover in a single dataset are not an accurate reflection of the entire culture that this dataset may derive from and we would find this conclusion to be particularly harmful. We instead aim to explore biases with respect to social structures in a particular dataset and are only able to discover and examine cultural attitudes that are particular and limited to the target dataset and are additionally limited by our knowledge recall. A precision-recall tradeoff exists based on the KG coverage, and thus the retrieved datapoints may not all be biased (precision) nor will all biases be retrieved (recall) using the available KG. Thus, we also stress the importance of using manual analysis to identify and confirm biases in the extracted datapoints. We hope this will further encourage the development of knowledge graphs that explicitly represent more annotated social/cultural commonsense knowledge by illustrating its usefulness in gaining a corpus-level understanding of datasets.

Similarly, we are additionally limited by the use of transformer models that were trained on mostly Western text and may be prone to capture Western cultural information even if fine-tuned on a dataset with different cultural attitudes. Providing a solution to this problem is beyond the scope of this paper, but future work could explore more culturally diverse data for pre-training models.

Data Use: Twitter data and interview transcripts are sensitive data and thus require strong considerations about the use and release of the data. The publicly released data for HurricaneEmo is fully anonymized to protect the identity of users. Stories2Insights data has also been fully anonymized, but is not publicly released and has been kept private to ensure the safety of the communities and to prevent harmful use of the data. Following suit, we do not release any data for this dataset to limit potential harm or misuse. For more details on the ethics concerning the collection and intended use of either of these datasets, please refer to each dataset's original paper.

Future Work: Our contribution focuses on the novel combination of knowledge graph relations, interpretability methods, and clustering to identify both influential and biased commonsense relationships. We aim to use this as an opportunity to encourage work in the curation of resources that explicitly annotate cultural attitudes or "commonsense" biases by illustrating how such resources can facilitate a corpus-level understanding of implicit influential cultural biases. Our work is limited by the methods we explored, and thus we encourage further investigation of certain elements in our experimental pipeline, in particular other datasets, clustering method variations, other MLMs, more diverse KGs, and other knowledge models.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. Emonet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 718–728.

Samyadeep Basu, Xuchen You, and Soheil Feizi. 2020. On second-order group influence functions for black-box predictions. In *International Conference on Machine Learning*, pages 715–724. PMLR.

Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Ting-Yun Chang, Yang Liu, Karthik Gopalakrishnan, Behnam Hedayatnia, Pei Zhou, and Dilek Hakkani-Tur. 2020. Incorporating commonsense knowledge graph in pretrained models for social commonsense tasks. In *Proceedings of Deep Learning Inside Out*

(DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, pages 74–79.

Ying Chen, Wenjun Hou, Xiyao Cheng, and Shoushan Li. 2018. Joint learning for emotion classification and emotion cause detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 646–651.

Costanza Conforti, Stephanie Hirmer, David Morgan, Marco Basaldella, and Yau Ben Or. 2020. Natural language processing for achieving sustainable development: the case of neural labelling to enhance community profiling. *arXiv preprint arXiv:2004.12935*.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103.

Shrey Desai, Cornelia Caragea, and Junyi Jessy Li. 2020. Detecting perceived emotions in hurricane disasters. *arXiv preprint arXiv:2004.14299*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Denis Emelin, Ronan Le Bras, Jena D Hwang, Maxwell Forbes, and Yejin Choi. 2020. Moral stories: Situated reasoning about norms, intents, actions, and their consequences. *arXiv preprint arXiv:2012.15738*.

R Ezhilarasi and RI Minu. 2012. Automatic emotion recognition and classification. *Procedia Engineering*, 38:21–26.

Maxwell Forbes, Jena D Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 653–670.

Benoît Frénay and Michel Verleysen. 2013. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.

Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. Cosmic: Commonsense knowledge for emotion identification in conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2470–2481.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563.

Oktie Hassanzadeh, Debarun Bhattacharjya, Mark Feblowitz, Kavitha Srinivas, Michael Perrone, Shirin Sohrabi, and Michael Katz. 2019. Answering binary causal questions through large-scale text mining: An evaluation using cause-effect pairs from human experts. In *IJCAI*, pages 5003–5009.

Stephanie Hirmer and Peter Guthrie. 2016. Identifying the needs of communities in rural uganda: A method for determining the 'user-perceived value'of rural electrification initiatives. *Renewable and Sustainable Energy Reviews*, 66:476–486.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2020. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. *arXiv preprint arXiv:2010.05953*.

Phil Katz, Matt Singleton, and Richard Wicentowski. 2007. Swat-mp: the semeval-2007 systems for task 5 and task 14. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 308–313.

Humayun Kayesh, Md Islam, Junhu Wang, et al. 2019. On event causality detection in tweets. *arXiv preprint arXiv:1901.03526*.

Humayun Kayesh, Md Saiful Islam, Junhu Wang, Shikha Anirban, ASM Kayes, and Paul Watters. 2020a. Answering binary causal questions: A transfer learning based approach. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE.

Humayun Kayesh, Md Saiful Islam, Junhu Wang, ASM Kayes, and Paul A Watters. 2020b. A deep learning model for mining and detecting causally related events in tweets. *Concurrency and Computation: Practice and Experience*, page e5938.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.

Lucy H Lin, Scott B Miles, and Noah A Smith. 2018. Natural language processing for analyzing disaster recovery trends expressed in large text corpora. In *2018 IEEE Global Humanitarian Technology Conference (GHTC)*, pages 1–8. IEEE.

Zhiyi Luo, Yuchen Sha, Kenny Q Zhu, Seung-won Hwang, and Zhongyuan Wang. 2016. Commonsense causal reasoning between short texts. In *Proceedings of the Fifteenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–430.

Saif Mohammad. 2012. # emotional tweets. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 246–255.

Saif M Mohammad and Svetlana Kiritchenko. 2015. Using hashtags to capture fine emotion categories from tweets. *Computational Intelligence*, 2(31):301–326.

Elham Mohammadi, Hessam Amini, and Leila Kosseim. 2019. Neural feature extraction for contextual emotion detection. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 785–794.

Aida Nematzadeh, Kaylee Burns, Erin Grant, Alison Gopnik, and Tom Griffiths. 2018. Evaluating theory of mind in question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2392–2400.

Robert Plutchik. 2001. The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American scientist*, 89(4):344–350.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. Xcopa: A multilingual dataset for causal commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2362–2376.

Nazneen Fatema Rajani, Ben Krause, Wengpeng Yin, Tong Niu, Richard Socher, and Caiming Xiong. 2020. Explaining and improving model behavior with k nearest neighbor representations. *arXiv preprint arXiv:2010.09030*.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A Smith, and Yejin Choi. 2018. Event2mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 463–473.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the SIGDIAL 2013 Conference*, pages 21–30.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Abraham C Sanders, Rachael C White, Lauren S Severson, Rufeng Ma, Richard McQueen, Haniel C Alcântara Paulo, Yucheng Zhang, John S Erickson, and Kristin P Bennett. 2021. Unmasking the conversation on masks: Natural language processing for topical sentiment analysis of covid-19 twitter discourse. *medRxiv*, pages 2020–08.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019b. Socialiqa: Commonsense reasoning about social interactions. In *Conference on Empirical Methods in Natural Language Processing*.

Sima Sharifirad, Borna Jafarpour, and Stan Matwin. 2019. How is your mood when writing sexist tweets? detecting the emotion type and intensity of emotion using natural language processing techniques. *arXiv preprint arXiv:1902.03089*.

Avirup Sil, Fei Huang, and Alexander Yates. 2010. Extracting action and event semantics from web text. In *AAAI Fall Symposium: Commonsense Knowledge*. Citeseer.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Carlo Strapparava and Rada Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 70–74.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Wenbo Wang, Lu Chen, Krishnaprasad Thirunarayan, and Amit P Sheth. 2012. Harnessing twitter" big data" for automatic emotion identification. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 587–592. IEEE.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Appendix

A.1 HurricaneEmo Pre-processing

We performed the following pre-processing steps for HurricaneEmo (released by Desai et al. (2020)). First, we de-duplicated each train, dev, and test file (per the authors suggestion). Then, we de-duplicated across splits, removing data from the train set if it was present in the dev or test sets. We used these files as our input for the pipeline described in our paper. We chose not to include the OPTIMISM category in our experiments since we were not able to reproduce a baseline result sufficiently close to the original paper on the fully deduplicated dataset. After preprocessing, the train/dev/test splits for each of the files are: AGGRESSIVENESS: 1695/493/495, AWE: 2942/868/868, CONTEMPT: 1507/452/443, DISAPPROVAL: 2363/706/707, LOVE: 1028/307/304, REMORSE: 3104/910/908, SUBMISSION 2432/724/721.

A.2 Stories2Insights Pre-processing

We performed the following pre-processing steps for Stories2Insights. We received the dataset from the authors, with the T3 labels described in the original paper (Conforti et al., 2020). We instead use the T1 labels defined in their paper as binary datasets. To create these datasets, we followed the T3 label groupings shown in Appendix A of their paper. We then created datasets with a 1:1 positive-to-negative datapoint ratio for the train, dev, and test sets. These negative datapoints were collected evenly from all other T1 datasets. We repeated this process for all T1 labels and every set (train, dev, and test). After preprocessing, the train/dev/test splits are: EMOTIONAL: 805/75/84, EPIST: 553/82/65, FUNCTION: 2551/302/345, INDIGENOUS: 695/64/104, INTRINSIC HUMAN: 2317/261/320, SOCIAL SIGNIFICANCE: 1241/134/151. If interested in using this data, please contact the original authors for access.

A.3 Reproducibility & Hyperparameters

We utilized BERT-base for all of our experiments (Devlin et al., 2018). We followed hyperparameter settings for BERT as described in Desai et al. (2020). To train BERT for HurricaneEmo, we use batch size 8 and learning rate 2e-5. To train BERT for Stories2Insights, we use batch size 8 and learning rate 1e-5. Training for both was completed in 3 epochs. We train each model using 1 GeForce GTX 1080 Ti GPU.

A.4 Manual Analysis Details and Further Analysis

A.4.1 Data Extraction

Manual analysis for examining the knowledge-datapoint matches for each dataset was completed by an expert author, since this is time-consuming, fine grained verification analysis (as opposed to model evaluation). Details on the setup of the analysis are in the main paper Section 4.3.

A.4.2 Cluster Match

This section describes the manual analysis used to analyze the cluster matches via our methods in Section 6.2 in the main paper (Table 6). Manual analysis was completed by 2 expert authors since this is time-consuming, fine grained verification analysis (as opposed to model evaluation). We obtain high agreement, with a Cohen kappa of 0.81 (which is considered as ‘almost perfect agreement’; see `https://en.wikipedia.org/wiki/Cohen%27s_kappa`). We selected 10 dev and 10 train datapoints for each event cluster, and because not all events had a full 10 datapoints in dev or train, this leads to total of 190 datapoints.

Using this data, we performed an analysis to identify whether the event-specific data sets that were extracted for a certain ATOMIC event (1) semantically matched the events and (2) semantically matched each other. For (1), we wanted to see whether the semantic meaning in the ATOMIC event was reflected in the clustered data. For (2), we wanted to see whether the semantic meaning in event-specific dev (c_j^d) and event-specific train (c_j^t) sets was matched. The results were computed as follows. For (1), if the majority of selected datapoints matched the event, we considered this to be a positive data−event match. We then compute: # events with semantically matching data−event clusters / # total events. For (2), if the majority of selected datapoints matched across dev and train, we considered this to be a positive train−dev match. We then compute: # events with semantically matching train−dev clusters / # total events. Majority is calculated such that a majority of both train and dev had to be semantically similar.

We see that the results for (2) are very high in Table 6, which supports our findings that there is mislabeling in the dataset (i.e, the labels are different between dev and train sets but the semantic

Dataset	Event	%Event-Specific Train	%Event-Specific Dev
HurricaneEmo			
AGR	PersonX drives from florida	4.7	7.9
LOV	PersonX checks the weather forecast	24.0	36.2
	PersonX uses — to avoid	5.5	6.5
SBM	PersonX sends — to the congress	1.2	2.3
AWE	PersonX prepares for the storm	4.6	7.6
	PersonX is still valid —	0.8	1.5
DSP	PersonX practices — in the state	0.6	1.0
	PersonX keeps PersonY in PersonY's prayers	0.1	0.3
CNT	PersonX sails close to the wind	7.2	8.6
RMR	PersonX crosses my heart and hope to die	5.9	6.8
	PersonX sends — to the congress	3.2	4.6
	PersonX keeps PersonY in PersonY's prayers	0.2	0.5
Stories2Insights			
Indigenous	PersonX offer — to the gods	4.7	4.7
	PersonX reads the bible	4.7	12.5
Intrinsic	PersonX provides — to children	1.1	1.9
Social	PersonX protects — from the effects	2.0	5.2
	PersonX sells — to the public	21.9	25.4

Table 8: Leave-K-Out Cluster Size. %Event-Specific Train is the % of train examples removed during Leave-K-Out training and %Event-Specific Dev is the % of dev examples the trained model is evaluated on.

meaning are similar, thus there is mislabeling). We also see that (1) can be low for some datasets, indicating that while the train and dev data may be semantically matched, this semantic meaning may differ from the original ATOMIC commonsense relation for some datasets more than others. This may be due to limited coverage of ATOMIC for some events and is intended to show transparency in the limitations of our approach, which we hope will encourage the development of a KG that explicitly represents this type of cultural knowledge (which was not available at the time of this work).

A.4.3 Mislabeling

This section describes the manual analysis used to confirm mislabeling in HurricaneEmo train examples. Manual analysis was completed by an expert author since this is time-consuming, fine grained verification analysis (as opposed to model evaluation). We completed manual analysis to identify mislabeling on a set of 30 train examples across different event clusters. The train examples were selected from cases where the performance improved on the event specific dev set when the event specific train set was removed.

A.5 Knowledge Influence Details

A.5.1 Leave-K-Out Cluster Size

We illustrate the sizes of the event-specific train (c_j^t) and dev (c_j^d) clusters with respect to the original train and dev dataset sizes for each of the leave-K-out results in Table 8. Some of the resulting datasets are very small and are more difficult to draw conclusions from. For this reason, we show the dataset sizes in Table 8 for an improved and transparent interpretation of the results. On the other hand, we can also see that several of the datasets are quite large and leave-K-out has a clear and significant impact on their performance with respect to the full train set (e.g., "PersonX checks the weather forecast" and "PersonX sells — to the public").

A.5.2 Selected Events

We show full top 5 events from the event selection step for each binary dataset in Tables 9 and 10.

Dataset	**Event**
Emotional	PersonX sells PersonY’s — for money PersonX sleeps well — PersonX gets — at night PersonX devotes — to the study PersonX wakes up in the middle of the night
Epist	PersonX hears — on the radio PersonX checks the news PersonX supplies the — with food PersonX loves listening to music PersonX protects the — from injury
Functional	PersonX protects the — from injury PersonX uses — to prevent PersonX seeks god ’s — PersonX teaches children — PersonX educates PersonX’s children
Indigenous	PersonX offer — to the gods PersonX reads the bible PersonX pays a lot of money PersonX seeks god ’s — PersonX treats the — with respect
Intrinsic	PersonX educates PersonX’s children PersonX provides — to children PersonX works well in business to get PersonX pays a lot of money PersonX helps the — to understand
Social	PersonX uses — to prevent PersonX protects — from the effects PersonX sells —to the public PersonX uses — to support PersonX tries to use it

Table 9: Top 5 Events for Stories2Insights Binary Datasets

Dataset	**Event**
AGR	PersonX stays away from PersonY PersonX drives from florida PersonX doesn't have enough money PersonX doesn't want to go to school PersonX moves to texas
DSP	PersonX comes back to my house PersonX practices — in the state PersonX keeps PersonY in PersonY's prayers PersonX is trying to watch a movie PersonX sees PersonY's friends
CNT	PersonX says the wrong thing PersonX can't afford to fix it PersonX doesn't want to go to school PersonX supplies the — with food PersonX sails close to the wind
LOV	PersonX practices — in the state PersonX goes to the local animal shelter PersonX spends — with PersonX's families PersonX checks the weather forecast PersonX uses — to avoid
SBM	PersonX uses PersonX's — to help PersonX goes to the local animal shelter PersonX sends — to the congress PersonX organizes — in a way PersonX does n't want to go to school
AWE	PersonX thanks god PersonX provides — for the people PersonX uses — to protect PersonX is still valid — PersonX prepares for the storm
RMR	PersonX crosses my heart and hope to die PersonX strikes — into the hearts PersonX sends — to the congress PersonX keeps PersonY in PersonY's prayers PersonX doesn't want to go to school

Table 10: Top 5 Events for HurricaneEmo Binary Datasets

Counter-GAP: Counterfactual Bias Evaluation through Gendered Ambiguous Pronouns

Zhongbin Xie[1], **Vid Kocijan**[2], **Thomas Lukasiewicz**[3,1], **Oana-Maria Camburu**[4]
[1] University of Oxford [2] Kumo.ai [3] TU Wien
[4] University College London
`zhongbin.xie@cs.ox.ac.uk`, `vid@kumo.ai`,
`thomas.lukasiewicz@tuwien.ac.at`, `o.camburu@ucl.ac.uk`

Abstract

Bias-measuring datasets play a critical role in detecting biased behavior of language models and in evaluating progress of bias mitigation methods. In this work, we focus on evaluating gender bias through coreference resolution, where previous datasets are either hand-crafted or fail to reliably measure an explicitly defined bias. To overcome these shortcomings, we propose a novel method to collect diverse, natural, and minimally distant text pairs via counterfactual generation, and construct Counter-GAP, an annotated dataset consisting of 4008 instances grouped into 1002 quadruples. We further identify a bias cancellation problem in previous group-level metrics on Counter-GAP, and propose to use the difference between inconsistency across genders and within genders to measure bias at a quadruple level. Our results show that four pre-trained language models are significantly more inconsistent across different gender groups than within each group, and that a name-based counterfactual data augmentation method is more effective to mitigate such bias than an anonymization-based method.

1 Introduction

It is a common practice to train state-of-the-art natural language processing (NLP) models by unsupervised pre-training and supervised fine-tuning (e.g., Devlin et al., 2019; Joshi et al., 2020), both of which rely heavily on large corpora of real-world text. However, these corpora often reflect societal stereotypes and may lead to models exhibiting biased behaviors (Bender et al., 2021). Hence, much research effort has been put to reveal and mitigate unintended biases (Meade et al., 2022).

While early work focuses on detecting and mitigating gender bias in the space of word embeddings (e.g., Bolukbasi et al., 2016), recent approaches turn to design bias-measuring datasets on specific NLP tasks (Nangia et al., 2020; Nadeem et al., 2021; Barikeri et al., 2021). In this work, we focus on gender bias in coreference resolution and adopt a kind of representational harm (Blodgett et al., 2020) to define gender fairness: a gender-neutral model should rely on the semantic information, rather than on the gender information contained in the texts, to make predictions. Otherwise, a model should be considered gender-biased. In line with this definition, WinoBias (Zhao et al., 2018) and WinoGender (Rudinger et al., 2018) leverage pairs of minimally distant sentences, i.e., two sentences that contain the same semantic information but different gender information, to measure models' performance difference in resolving pronouns of different genders under the same context. This minimally distant setting enables us to isolate the influence of gender information on model predictions.

A limitation of WinoBias and WinoGender is that they are made up of *hand-crafted sentences*, which prevents us from measuring gender bias in the more diverse real-world scenarios. An alternative to overcome this shortcoming is the GAP dataset (Webster et al., 2018), which exploits linguistic patterns to automatically extract instances from a *real-world corpus*. However, since GAP's masculine and feminine instances cannot be grouped into minimally distant pairs, we are not sure whether a difference in model performance is due to different gender information or to different semantic information. For example, compared to masculine instances, GAP's feminine instances have more candidate entities serving as distractors, and longer distance between the correct name and the pronoun (Kocijan et al., 2021). So, it is *not equally hard to resolve* the masculine and feminine instances in GAP. Hence, the performance difference between masculine and feminine instances on GAP is not a reliable measure of gender bias according to the above definition of gender fairness.[1]

---

[1] In GAP (Webster et al., 2018), the authors did not explicitly describe the fairness definiton that they adopt.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3761–3773
May 2-6, 2023

Given these observations, we propose a novel method to construct coreference-resolution-based bias-measuring datasets consisting of minimally distant text pairs that originate from real-world corpora. Specifically, we leverage the method from GAP (Webster et al., 2018) to extract original instances containing gendered ambiguous pronouns, and generate minimally distant instances by asking the counterfactual question "How would the prediction change if we swapped the roles of masculine and feminine people in this context?" (Garg et al., 2019). The resulting instances are grouped into quadruples, each of which consists of an original, a gender-controlled, and two gender-swapped instances. An example is shown in Table 1.

Furthermore, we find that bias in different directions may be canceled out if we aggregate the results by performance difference across groups of instances, and we call this problem *bias cancellation*. To alleviate it, we propose a new metric, inconsistency across genders, to measure bias at the quadruple level. We also leverage the gender-controlled instances to disentangle inconsistency *within* genders from inconsistency *across* genders, so that we can eliminate the impact of name perturbations.

Our contributions are as follows: (i) We propose a novel method to construct coreference resolution datasets consisting of diverse, natural, and minimally distant instances to reliably detect gender bias. (ii) We apply our method to online books and collect Counter-GAP, an annotated dataset with 4008 instances grouped into 1002 quadruples. (iii) We propose a new metric, the difference (ΔI) between inconsistency across genders and within genders, to alleviate the bias cancellation problem of previous metrics. (iv) We use Counter-GAP to empirically evaluate four pre-trained language models and two debiasing methods based on Counterfactual Data Augmentation (CDA, Zhao et al., 2018; Webster et al., 2020). Our results show that ΔI can detect significant gender bias hidden by group-level performance difference, and that name-based CDA is more effective than vanilla CDA in mitigating such bias.[2]

2 Dataset Construction

The Counter-GAP dataset is derived from 1575 fictional books in Project Gutenberg[3] and BookCorpus (Zhu et al., 2015). It is constructed through a generic multi-stage process, as described below. Here, we follow the GAP dataset (Webster et al., 2018) and focus only on the English language, as well as adopting a notion of binary gender.

2.1 Original Instance Extraction

First, we detect all the occurrences of personal names and pronouns in a book with a dependency parser and a named entity recognizer (NER).[4] For each occurrence of a gendered non-reflexive pronoun (*he*, *him*, *his*, *she*, *her*, *hers*), we extract a surrounding context that consists of a maximum of five sentences and contains exactly two masculine and two feminine personal names. Personal names are identified by NER tags, and the gender specification of a name is determined by statistics from a gender-guesser.[5] Genders for titled names (e.g., *Mr. Smith*) are assumed from the traditional gender associations of those titles.

Second, we select the subset of contexts that contain gendered ambiguous pronouns as defined by the following three patterns from GAP (Webster et al., 2018) (henceforth, the gendered ambiguous pronoun is called *target pronoun*, and the two names that are gender-consistent with the target pronoun are called *candidate names*):

- **FINALPRO.** Both candidate names must be in the same sentence, and the target pronoun may appear in the same or directly following sentence.
- **MEDIALPRO.** The first candidate name must be in the sentence directly preceding the target pronoun and the second candidate name, both of which must be in the same sentence. The target pronoun must be in an initial subordinate clause or be a possessive in an initial prepositional phrase.
- **INITIALPRO.** Both the candidate names and the target pronoun must be in the same sentence, and the target pronoun must be in an

---

[2] The dataset and code are available at `https://github.com/x-zb/Counter-GAP`.

[3] `https://www.gutenberg.org/`

[4] We use Spacy (`https://spacy.io/`).

[5] `https://pypi.org/project/gender-guesser/`. As we focus on English books, we use the default setting where the gender of a name is first considered according to its use in English-speaking countries.

original	Tom did not appear to hear this, but tried to keep up the conversation with Julia, desiring to have it appear that they were intimate friends; but the young lady gave brief replies, and finally, turning away, devoted herself once more to Herbert, much to Tom's disgust. In fact, what he saw made Tom pass a very unpleasant evening, and when, on their return home, Maria suggested that Julia had taken a fancy to Herbert, **he** told her to mind her own business.
gender-controlled	Herbert did not appear to hear this, but tried to keep up the conversation with Maria, desiring to have it appear that they were intimate friends; but the young lady gave brief replies, and finally, turning away, devoted herself once more to Tom, much to Herbert's disgust. In fact, what he saw made Herbert pass a very unpleasant evening, and when, on their return home, Julia suggested that Maria had taken a fancy to Tom, **he** told her to mind her own business.
gender-swapped-1	Maria did not appear to hear this, but tried to keep up the conversation with Herbert, desiring to have it appear that they were intimate friends; but the young *gentleman* gave brief replies, and finally, turning away, devoted *himself* once more to Julia, much to Maria's disgust. In fact, what *she* saw made Maria pass a very unpleasant evening, and when, on their return home, Tom suggested that Herbert had taken a fancy to Julia, **she** told *him* to mind *his* own business.
gender-swapped-2	Julia did not appear to hear this, but tried to keep up the conversation with Tom, desiring to have it appear that they were intimate friends; but the young *gentleman* gave brief replies, and finally, turning away, devoted *himself* once more to Maria, much to Julia's disgust. In fact, what *she* saw made Julia pass a very unpleasant evening, and when, on their return home, Herbert suggested that Tom had taken a fancy to Maria, **she** told *him* to mind *his* own business.

Table 1: Counterfactual generation of a quadruple in Counter-GAP. Personal names and their genders are depicted in colors: masculine names are in blue and cyan; feminine names are in violet and orange. The target pronoun is in **bold** and underlined; also underlined is the true coreferent name. Other words constitute the context, and words in *italic* are gendered words swapped according to the gendered words list.

initial subordinate clause or a possessive in an initial prepositional phrase.

After filtering, we get 2585 contexts and adopt them as original instances.

2.2 Counterfactual Generation

We generate minimally distant instances in Counter-GAP through two counterfactual generation functions. An example is illustrated in Table 1. Formally, we denote an original instance as $x_o = s(P, C_1, C_2, O_1, O_2)$, where P is the target pronoun to be resolved, C_1 and C_2 are the two candidate names that are gender-consistent with P, O_1 and O_2 are two personal names of the opposite gender, and $s(\cdot)$ denotes the context around these mentions.

Gender-controlled generation. We swap all the occurrences of C_1 and C_2, and of O_1 and O_2, to generate a gender-controlled instance $x_c = s(P, C_2, C_1, O_2, O_1)$. We choose to swap names within an instance instead of introducing new names, so that the candidate names naturally occur in the same real-world context.

Gender-swapped generation. We first substitute all gendered words with their opposite gendered words (e.g., man→woman, he→she),[6] and swap all the occurrences of C_1 and O_1 (or O_2), C_2 and O_2 (or O_1). As a result, we obtain two gender-swapped instances $\widetilde{x_o} = \tilde{s}(\tilde{P}, O_1, O_2, C_1, C_2)$ and $\widetilde{x_c} = \tilde{s}(\tilde{P}, O_2, O_1, C_2, C_1)$, where $\tilde{s}(\cdot)$ is the context with all the gendered words substituted in $s(\cdot)$, and $\tilde{P}$ is the opposite-gendered pronoun for P. We call $x_o, x_c, \widetilde{x_o}, \widetilde{x_c}$ minimally distant instances, in that the words at the same position in the context ($s(\cdot)$ or $\tilde{s}(\cdot)$) are either the same (for gender-neutral words) or have the same role but opposite gender (for gendered words).

We consider a generated counterfactual instance to be invalid if (i) it contradicts commonsense knowledge, e.g., historical people being of the opposite gender; or (ii) the meaning of the counterfactual is different from the original, resulting in the gold coreference labels changing or becoming undetermined. To tackle these, we take three measures. First, we extract original instances mainly from fictional books, whose content is less likely to involve real-world people. Second, during human annotation (Section 2.3), we explicitly ask annotators to validate whether an instance contradicts commonsense knowledge, and discard such instances. Third, we discard the whole quadruple $(x_o, x_c, \widetilde{x_o}, \widetilde{x_c})$ if not all of its four instances get the same majority labels from annotators.[7] Here, *same*

---

[6] We adopt an augmented list of gendered words from (Zhao et al., 2018).

[7] Note that discarding inconsistent quadruples can also cover some error cases caused by the gender-guesser's incorrect prediction. For example, if an incorrect gender prediction occurs for a TRUE coreferent name, the target pronoun and the TRUE coreferent name will not be gender-consistent, and

label means the coreferent names' positions are the same in the context (e.g., in Table 1, the position of "Tom" in the original instance and that of "Maria" in the gender-swapped-1 instance).

2.3 Human Annotation

We use Amazon Mechanical Turk to collect coreference labels for all the 2585 original instances and their counterfactual counterparts (hence, 2585×4 instances in total). Each instance was assigned to three annotators. Annotation instructions and a sample task interface are presented in Appendix B. Specifically, we ask annotators to highlight token spans that are coreferent with the target pronoun. We adopt majority vote to aggregate the collected annotations, and generate a `TRUE/FALSE` label for each of the two candidate names indicating if it is coreferent with the target pronoun.

After discarding quadruples containing invalid counterfactuals as discussed in Section 2.2, we further filter out quadruples containing real-world people to avoid grounding. Next, we randomly downsample the remaining quadruples to balance the number of original masculine and feminine instances. The final Counter-GAP dataset consists of 1002 quadruples with an inter-annotator agreement[8] of 86.5%.

3 Evaluation Metrics on Counter-GAP

We use $X = (x_o, x_c, \widetilde{x_o}, \widetilde{x_c}) \in \mathcal{X}$ to denote a quadruple, and lowercased x to denote an arbitrary instance, which could be each of $x_o, x_c, \widetilde{x_o}, \widetilde{x_c}$ from a quadruple. Given a model $f(\cdot)$, assume that $f(x) \in \{0, 1\}$ indicates whether f's prediction on instance x is correct (1) or not (0).

3.1 Bias Cancellation in Accuracy Difference

A so far commonly used metric is to directly compare the model's performance difference (or ratio) between different gender groups (Webster et al., 2018; Sun et al., 2019; Blodgett et al., 2020). For example, if we divide a test set $\mathcal{X}$ into a group of masculine instances $\mathcal{D}^{(m)}$ and a group of feminine instances $\mathcal{D}^{(f)}$ according to the gender information contained in the instances (e.g., the gender of the target pronoun), gender bias can be measured by model f's accuracy difference (Acc_{Diff}) on $\mathcal{D}^{(m)}$ and $\mathcal{D}^{(f)}$:

$$Acc_{Diff} = \frac{\sum_{x \in \mathcal{D}^{(m)}} f(x)}{|\mathcal{D}^{(m)}|} - \frac{\sum_{x \in \mathcal{D}^{(f)}} f(x)}{|\mathcal{D}^{(f)}|}. \quad (1)$$

However, the above metric may suffer from *bias cancellation* on Counter-GAP. Consider two quadruples from Counter-GAP. In the first, the model makes correct predictions on the two masculine instances and incorrect predictions on the two feminine ones. The model should be deemed gender-biased (towards masculine), since it makes different predictions for instances containing the same semantic information. If, in the second quadruple, the model makes reversed predictions, i.e., correct on the two feminine instances and incorrect on the two masculine ones, it should also be deemed gender-biased, yet in the opposite direction towards feminine. However, the model's accuracies on the masculine and feminine groups are both $2/4 = 50\%$, making Eq. (1) equal to zero. In short, biases in opposite directions may be canceled out in some cases if we use Eq. (1) to aggregate them.

3.2 Measuring Bias via Inconsistencies

Given the bias cancellation problem of accuracy difference, we propose to measure gender bias through inconsistencies, i.e., whether a model's prediction is consistent on a pair of minimally distant instances. Specifically, we adopt two metrics, *inconsistency across genders* (I_{across}):

$$\frac{1}{4|\mathcal{X}|} \sum_{X \in \mathcal{X}} \Big(|f(x_o) - f(\widetilde{x_o})| + |f(x_c) - f(\widetilde{x_c})| + |f(x_o) - f(\widetilde{x_c})| + |f(x_c) - f(\widetilde{x_o})| \Big), \quad (2)$$

and *inconsistency within genders* (I_{within}):

$$\frac{1}{2|\mathcal{X}|} \sum_{X \in \mathcal{X}} \Big(|f(x_o) - f(x_c)| + |f(\widetilde{x_o}) - f(\widetilde{x_c})| \Big). \quad (3)$$

Inconsistency across genders (I_{across}) measures inconsistency in instance pairs containing two instances of different genders, while inconsistency within genders (I_{within}) measures inconsistency in instance pairs containing two instances of the same gender. The two instances in a pair should be minimally distant (i.e., from the same quadruple) to guarantee that they contain the same semantic information. Since Counter-GAP adopts personal names as proxies for person entities, we need to

---

this may confuse the annotators, leading to inconsistent labels.

[8] Average percentage of agreed annotations on each instance.

disentangle the part of inconsistency caused by different names (I_{within}) from that caused by different genders (I_{across}). Therefore, our final metric to measure gender bias is

$$\Delta I = I_{across} - I_{within}. \tag{4}$$

In practice, a positive ΔI indicates biased behaviors of the model, while a zero or negative ΔI means that the measured inconsistency across genders are mostly noises from name perturbations, thus no bias can be detected.

4 Bias Evaluation on Counter-GAP

For evaluation, we adopt the coreference resolution system `c2f-coref`[9] (Lee et al., 2018) based on four pre-trained language models: BERT-base/large and SpanBERT-base/large (Joshi et al., 2020). All four models are fine-tuned on OntoNotes (Pradhan et al., 2012),[10] and training details are shown in Appendix A. In our evaluation, no candidate names are provided as input to the models, and models are responsible to detect candidate names in the text by themselves. A model's prediction on an instance is considered correct if the candidate name with gold label TRUE and none of those with gold label FALSE are in the target pronoun's coreferent cluster.

4.1 Results

Results for gender bias measured on Counter-GAP by accuracy difference (Eq. (1)) and ΔI (Eq. (4)) are shown in Tables 2 and 3, respectively. In Table 3, we also report the inconsistency metrics on each gender group (M, F) and each swapping direction (M2F, F2M), together with their differences.

Results in Table 3 show that for all four models, not only I_{across} is larger than I_{within} (ΔI being positive), but also the difference is statistically significant, which indicates biased behaviors in these models. Note that the absolute values of accuracy difference (Acc_{Diff}) in Table 2 are in general smaller than the corresponding values of ΔI in Table 3, and Acc_{Diff} for BERT-large even becomes statistically insignificant, which is contrary to the well-known conclusion that BERT encodes social bias (Nadeem et al., 2021). This brings evidence towards the bias cancellation problem, i.e., bias measured by accuracy difference (Eq. (1)) may be canceled out compared to that measured by inconsistency difference (ΔI).

Regarding the effect of model size on gender bias, results from both metrics show that larger models seem to be less biased than smaller models. Note that both our large and base models are (pre-) trained on the same datasets, but in general larger language models are pre-trained on larger amount of data, so they are still at a higher risk of exhibiting biased behaviors (Bender et al., 2021).

Regarding the detected bias direction, different metrics provide information from different perspectives. We can learn from the sign of Acc_{Diff} in Table 2 that the overall bias directions of these models are all towards masculine. In Table 3, all of the Diff. for I_{within} are negative, indicating a larger inconsistency within the feminine group. All of the Diff. for I_{across} being negative indicates that inconsistency will increase when we change genders in an originally feminine context.

Models	Acc_M	Acc_F	Acc_{Diff}
BERT-base	63.12%	59.53%	+3.59%*
BERT-large	72.60%	72.11%	+0.50%
SpanBERT-base	71.36%	69.06%	+2.30%*
SpanBERT-large	77.25%	75.40%	+1.85%*

Table 2: Gender bias measured by Eq. (1) on Counter-GAP. We report accuracy on masculine instances (Acc_M), feminine instances (Acc_F), and their difference ($Acc_{Diff} = Acc_M - Acc_F$). A "$*$" means that the difference is statistically significant ($p < 0.01$) under one-sided bootstrap resampling (Graham et al., 2014).

4.2 No Bias Between the Original and Counterfactual Instances

Since the gender-swapped instances in Counter-GAP are generated automatically, although they have been validated by annotators, we still check whether there is a systematic bias towards the original or counterfactual instances. To investigate this, we measure two statistics. First, we measure a model's accuracy on instances with the original gender (x_o, x_c) and the counterfactual gender ($\widetilde{x_o}, \widetilde{x_c}$), and report their difference. From Table 4, we see that the differences are very small and not statistically significant. Second, we measure the correlation between the inconsistency across genders score ($|f(x_o) - f(\widetilde{x_o})| + |f(x_c) - f(\widetilde{x_c})| + |f(x_o) - f(\widetilde{x_c})| + |f(x_c) - f(\widetilde{x_o})|$) and the original gender of a quadruple X. From Table 4, we

---

[9] We use the implementations from `https://github.com/mandarjoshi90/coref`.

[10] Since the annotation conventions of OntoNotes are a little different from those of Counter-GAP, we omitted the abbreviation period "." in titles like "Mr.", "Mrs.", and "Dr." in Counter-GAP during evaluation.

Models	inconsistency within genders				inconsistency across genders				$\Delta I = I_{across}$ $-I_{within}$
	M	F	Diff.	I_{within}	M2F	F2M	Diff.	I_{across}	
BERT-base	15.47%	16.47%	-1.00%	15.97%	18.26%	23.25%	-4.99%	20.76%	+4.79%*
BERT-large	10.28%	10.28%	0.00%	10.28%	10.88%	14.27%	-3.39%	12.57%	+2.30%*
SpanBERT-base	9.98%	12.18%	-2.20%	11.08%	12.18%	15.07%	-2.89%	13.62%	+2.54%*
SpanBERT-large	5.79%	6.29%	-0.50%	6.04%	6.89%	8.18%	-1.30%	7.53%	+1.50%*

Table 3: Gender bias measured by the inconsistency metrics on Counter-GAP. We report inconsistency within genders on masculine instances (M), feminine instances (F), and their difference (Diff. = M - F), as well as inconsistency within genders on all the instances (I_{within}). We also report inconsistency across genders on quadruples generated by transforming masculine instances to feminine instances (M2F), transforming feminine instances to masculine instances (F2M), and their difference (Diff. = M2F - F2M), as well as inconsistency across genders on all the quadruples (I_{across}). $\Delta I = I_{across} - I_{within}$ measures gender bias, where a "$*$" means that the difference is statistically significant ($p < 0.01$) under one-sided bootstrap resampling (Graham et al., 2014).

Models	Accuracy			Spearman'sρ
	Orig.	Counter.	Diff.	
BERT-base	61.58%	61.08%	+0.50%	-0.083
BERT-large	72.06%	72.65%	-0.60%	-0.065
SpanBERT-base	70.21%	70.21%	0.00%	-0.060
SpanBERT-large	76.55%	76.10%	+0.45%	-0.030

Table 4: Results on systematic bias evaluation. We report accuracy on instances with the original gender (Orig.), with the counterfactual gender (Counter.), and their difference (Diff.= Orig. - Counter.). All the differences are not statistically significant ($p > 0.01$) under one-sided bootstrap resampling (Graham et al., 2014). We also report Spearman's ρ between inconsistency across genders and the original gender of a quadruple.

see that the values of Spearman's ρ are all close to zero, indicating no significant correlations. Hence, we conclude that the counterfactual instances in Counter-GAP do not introduce systematic bias.

4.3 Comparison with GAP

We further compare Counter-GAP with two GAP-like datasets: the original GAP test set (Webster et al., 2018) and a subset of Counter-GAP where only the original instances x_o are kept (we call this dataset our-GAP). The results of SpanBERT-large on the above datasets are shown in Table 5. We see that the accuracy differences between masculine and feminine instances are much smaller on Counter-GAP than on the original GAP and our-GAP. This empirically verifies that datasets without minimally distant instances cannot reliably measure bias (they amplify bias in this case) due to the different semantic information contained in its masculine and feminine instances. Moreover, the overall direction of detected gender bias (the sign of Acc_{Diff}) is different on the original GAP and our-GAP, which shows that different source corpora (Wikipedia for GAP vs. fictions for our-GAP) may detect different bias in the model. This highlights the importance of domain diversity when using data-centric methods for bias detection.

5 Bias Mitigation

We evaluate two debiasing methods based on counterfactual data augmentation (CDA) (Zhao et al., 2018; Webster et al., 2020): (i) anonymization-based CDA (a-CDA), where the training set (OntoNotes) is augmented by substituting all gendered words with their opposite gendered words, while the gold coreference labels are kept unchanged. Personal names in the training set are anonymized using place holders such as "E1, E2, ..."; (ii) name-based CDA (n-CDA), where, in addition to the substitution between gendered words, masculine and feminine names also substitute each other according to their frequencies (Hall Maudslay et al., 2019). See Appendix A for more details. Performance on bias mitigation is measured by Acc_{Diff} and ΔI, while performance on coreference resolution is measured by overall accuracy on Counter-GAP and F_1 score on OntoNotes' dev set.

Results are shown in Table 6. In terms of ΔI, both a-CDA and n-CDA can effectively reduce gender bias, while n-CDA is more effective than a-CDA in that its ΔI values are smaller and less significant. Comparing the results of Acc_{Diff} and ΔI, we discover that bias measured by Acc_{Diff} tends to be more easily mitigated by the debiased methods. For example, a-CDA fails to reduce ΔI to an insignificant level for all the four models, but it succeeds to do so for Acc_{Diff} on BERT-base and SpanBERT-large; n-CDA can reduce BERT-base's Acc_{Diff} to an insignificant level, but fails to do so under the measurement of ΔI.

Regarding the trade-off between bias mitigation and overall performance, both a-CDA and n-CDA

Datasets	Accuracy				ΔI
	Acc_M	Acc_F	Acc_{Diff}	Overall	
original GAP	85.10%	79.80%	+5.30%	82.45%	– –
our-GAP	75.25%	78.44%	-3.19%	76.85%	– –
Counter-GAP	77.25%	75.40%	+1.85%	76.32%	+1.50%

Table 5: Results from SpanBERT-large on three datasets.

Models	Debiasing Method	$Acc_{Diff} = Acc_M - Acc_F$	$\Delta I = I_{across} - I_{within}$	Overall Accuracy	F_1 on OntoNotes
BERT-base	none	+3.59%*	+4.79%*	61.33%	74.39%
	a-CDA	+1.90%	+2.30%*	65.17%	73.91%
	n-CDA	**+0.20%**	+1.85%*	66.82%	73.60%
BERT-large	none	+0.50%	+2.30%*	72.36%	77.35%
	a-CDA	+2.99%*	+1.75%*	72.95%	76.96%
	n-CDA	+0.95%	+1.30%	73.53%	77.13%
SpanBERT -base	none	+2.30%*	+2.54%*	70.21%	77.71%
	a-CDA	+5.14%*	+2.54%*	69.49%	78.04%
	n-CDA	+0.95%	+1.25%	71.03%	77.70%
SpanBERT -large	none	+1.85%*	+1.50%*	76.32%	80.06%
	a-CDA	+0.35%	+1.45%*	76.72%	**80.07%**
	n-CDA	+0.65%	**+0.15%**	**77.92%**	79.93%

Table 6: Bias mitigation results. For Acc_{Diff} and ΔI, lower is better; for overall accuracy and F_1 on OntoNotes, higher is better. Best results are in bold. A "*" on ΔI indicates that the difference is statistically significant ($p < 0.01$) under one-sided bootstrap resampling (Graham et al., 2014).

can maintain or even increase the overall accuracy on Counter-GAP. This indicates that they do not sacrifice model performance for fairness, which is a favorable characteristic of debiasing methods. However, n-CDA achieves decreased F_1 scores on OntoNotes for all the four models, indicating that it is more suitable for tasks involving mostly personal names.

6 Qualitative Analysis

In Table 7, we show some Counter-GAP examples with predictions from SpanBERT-large. In Example 1, SpanBERT-large makes correct decisions for both the original and gender-controlled instance. But for the two gender-swapped instances, it either refers the target pronoun to the incorrect feminine person (Denise), or includes both feminine names (Roxanne and Denise) in the coreferent cluster, which indicates a worse performance on resolving feminine pronouns under the same context. This kind of gender bias can only be detected when we counterfactually augment the original instance. In Example 2, SpanBERT-large correctly finds "Alice" in the gender-swapped-2 instance, but confuses "Alice" and "Dora" in the gender-swapped-1 instance. This illustrates the inconsistency brought by name perturbations within the same gender, and we take this into account by subtracting inconsistency within genders from inconsistency across genders ($\Delta I = I_{across} - I_{within}$).

7 Related Work

Measuring Bias in NLP models. Human-like biases are first detected and measured in word embeddings (Bolukbasi et al., 2016; Caliskan et al., 2017; Garg et al., 2018; Gonen and Goldberg, 2019; Manzini et al., 2019). For pre-trained language models, May et al. (2019) adopt bleached sentence templates to contextualize target words, while most recent works leverage crowd-sourced benchmark datasets on NLP tasks such as language modeling (Nangia et al., 2020; Nadeem et al., 2021), sentiment analysis (Kiritchenko and Mohammad, 2018), dialog generation (Barikeri et al., 2021), natural language inference (Dev et al., 2020), and machine translation (Stanovsky et al., 2019). Our work follows GAP (Webster et al., 2018), WinoBias (Zhao et al., 2018), and WinoGender (Rudinger et al., 2018) to measure gender bias in coreference resolution, with a specific focus on collecting diverse, natural, and minimally distant instances.

Counterfactual Bias Evaluation. Kusner et al. (2017) propose the notion of *counterfactual fairness*, which requires similar predictions before and after counterfactual interventions in casual graphs. Garg et al. (2019) apply this notion to text classifi-

Example	1
original (correct)	"In fact, Roxanne told me that she had scheduled an interview with a source tonight." Denise sipped at her lemonade through her straw until she found the bottom of her glass at last. Scotty told Chris that Denise didn't give **him** any particulars about why she needed to hire a private detective when she sought his advice.
gender-controlled (correct)	"In fact, Denise told me that she had scheduled an interview with a source tonight." Roxanne sipped at her lemonade through her straw until she found the bottom of her glass at last. Chris told Scotty that Roxanne didn't give **him** any particulars about why she needed to hire a private detective when she sought his advice.
gender-swapped-1 (incorrect)	"In fact, Chris told me that he had scheduled an interview with a source tonight." Scotty sipped at his lemonade through his straw until he found the bottom of his glass at last. Denise told Roxanne that Scotty didn't give **her** any particulars about why he needed to hire a private detective when he sought her advice.
gender-swapped-2 (incorrect)	"In fact, Scotty told me that he had scheduled an interview with a source tonight." Chris sipped at his lemonade through his straw until he found the bottom of his glass at last. Roxanne told Denise that Chris didn't give **her** any particulars about why he needed to hire a private detective when he sought her advice.
Example	2
original (correct)	Dora said, "You ought not to bet, especially on Sunday," and Alice altered it to "You may be sure." "Well, but what then?" Oswald asked Denny. "Out with it," for **he** saw that his youthful friend had got an idea and couldn't get it out.
gender-controlled (correct)	Alice said, "You ought not to bet, especially on Sunday," and Dora altered it to "You may be sure." "Well, but what then?" Denny asked Oswald. "Out with it," for **he** saw that his youthful friend had got an idea and couldn't get it out.
gender-swapped-1 (incorrect)	Oswald said, "You ought not to bet, especially on Sunday," and Denny altered it to "You may be sure." "Well, but what then?" Dora asked Alice. "Out with it," for **she** saw that her youthful friend had got an idea and couldn't get it out.
gender-swapped-2 (correct)	Denny said, ""You ought not to bet, especially on Sunday,"" and Oswald altered it to ""You may be sure."" ""Well, but what then?"" Alice asked Dora. ""Out with it,"" for **she** saw that her youthful friend had got an idea and couldn't get it out.

Table 7: Examples from Counter-GAP. In each instance, the predicted coreference cluster from the model is highlighted in yellow, and the correctness of the prediction is annotated in the first column. The target pronoun is in **bold** and underlined; also underlined is the true coreferent name. Other notations follow those in Table 1.

cation and propose the metric of Counterfactual Token Fairness, which is similar to our inconsistency metrics, but we further distinguish inconsistency within genders from inconsistency across genders in our quadruple setting. Counterfactual Data Augmentation (CDA) (Webster et al., 2020; Zmigrod et al., 2019) is a widely adopted method for bias evaluation (Cao et al., 2020; Zhang et al., 2021), and we additionally focus on personal names during counterfactual generation of coreference resolution instances.

Name Artifacts in NLP Models. Since neural language models do not treat personal names as interchangeable, there are various biases in the learned representations of personal names (Shwartz et al., 2020; Prabhakaran et al., 2019; Wolfe and Caliskan, 2021; Wang et al., 2022). Counter-GAP considers name biases as the source of gender bias, and exhibits these name biases through the task of coreference resolution.

8 Summary and Outlook

In this work, we proposed a method to construct minimally distant bias-measuring datasets for coreference resolution, and exemplified it in the collection of Counter-GAP. We proposed the inconsistency metric ΔI to overcome the bias cancellation problem and noise from name perturbations. We showed that four pre-trained language models exhibit significant gender bias, and name-based CDA is most effective in mitigating the detected bias.

Limitations of Counter-GAP include that around half of the instances are from historical fictions in Project Gutenberg, making the dataset less representative of contemporary bias; the rules in our method for constructing Counter-GAP are specific for English, and might not be easily adapted to languages with more complex morphology; while we recognize that gender is non-binary, we adopt the simplifying setup of binary gender construct, which prevents us from detecting gender bias against minority groups with non-binary genders.

In future work, we will apply our method to different domains and more contemporary corpora such as news articles. Leveraging the data augmentation method for languages with grammatical genders (Zmigrod et al., 2019), as well as linguistic resources for non-binary genders (Cao et al., 2020) is also an important future direction to construct more gender- and language-inclusive datasets.

Limitations

Counter-GAP adopts the setup of a binary gender construct, which restricts it from detecting bias against non-binary gender groups. Future work may extend Counter-GAP using non-binary gendered word lists, and correspondingly extend our metric (inconsistency across and within binary gender groups) for multiple gender groups.

Our method relies on specific characteristics of the English language. Directly applying it to other languages may be non-trivial. For example, languages like French or Italian adopt grammatical genders that need extra rules in our counterfactual generation method, while Chinese names are, in principle, gender neutral, which makes it impossible to identify genders from personal names. Therefore, adaptation efforts are required for researchers working on multilingual problems.

Like many other bias-measuring datasets, Counter-GAP only serves as a diagnostic dataset. This means that, if our dataset and metric detect significant bias, we could deem a model biased; but if little or no bias is detected, we cannot guarantee that the model is unbiased. Practitioners may adopt diverse bias benchmarks before reaching a conclusion.

Acknowledgements

During this work, Oana-Maria Camburu was supported by an Early Career Leverhulme Fellowship. This work was also supported by the Alan Turing Institute under the EPSRC grant EP/N510129/1 and the AXA Research Fund. We also acknowledge the use of Oxford's ARC facility, of the EPSRC-funded Tier 2 facilities JADE (EP/P020275/1), and of GPU computing support by Scan Computers International Ltd.

References

Soumya Barikeri, Anne Lauscher, Ivan Vulić, and Goran Glavaš. 2021. RedditBias: A real-world resource for bias evaluation and debiasing of conversational language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1941–1955, Online. Association for Computational Linguistics.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai. 2016. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In *Proceedings of Neural Information Processing Systems (NeurIPS)*.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Qingqing Cao, Aruna Balasubramanian, and Niranjan Balasubramanian. 2020. Towards accurate and reliable energy measurement of NLP models. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 141–148, Online. Association for Computational Linguistics.

Sunipa Dev, Tao Li, Jeff Phillips, and Vivek Srikumar. 2020. On measuring and mitigating biased inferences of word embeddings. In *Proceedings of AAAI*.

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

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644.

Sahaj Garg, Vincent Perot, Nicole Limtiaco, Ankur Taly, Ed H. Chi, and Alex Beutel. 2019. Counterfactual fairness in text classification through robustness. In

Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, AIES '19.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614, Minneapolis, Minnesota. Association for Computational Linguistics.

Yvette Graham, Nitika Mathur, and Timothy Baldwin. 2014. Randomized significance tests in machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 266–274, Baltimore, Maryland, USA. Association for Computational Linguistics.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5267–5275, Hong Kong, China. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Vid Kocijan, Oana-Maria Camburu, and Thomas Lukasiewicz. 2021. The gap on GAP: Tackling the problem of differing data distributions in bias-measuring datasets. In *Proceedings of AAAI*.

Harold T. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Matt Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. In *Proceedings of Neural Information Processing Systems (NeurIPS)*.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 615–621, Minneapolis, Minnesota. Association for Computational Linguistics.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2022. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1878–1898, Dublin, Ireland. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation sensitivity analysis to detect unintended model biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5740–5745, Hong Kong, China. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Vered Shwartz, Rachel Rudinger, and Oyvind Tafjord. 2020. "you are grounded!": Latent name artifacts in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6850–6861, Online. Association for Computational Linguistics.

Gabriel Stanovsky, Noah A. Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684, Florence, Italy. Association for Computational Linguistics.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Jun Wang, Benjamin Rubinstein, and Trevor Cohn. 2022. Measuring and mitigating name biases in neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2576–2590, Dublin, Ireland. Association for Computational Linguistics.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *CoRR*, cs.CL/2010.06032v2.

Robert Wolfe and Aylin Caliskan. 2021. Low frequency names exhibit bias and overfitting in contextualizing language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 518–532, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chong Zhang, Jieyu Zhao, Huan Zhang, Kai-Wei Chang, and Cho-Jui Hsieh. 2021. Double perturbation: On the robustness of robustness and counterfactual bias evaluation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3899–3916, Online. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *International Conference on Computer Vision (ICCV)*.

Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661, Florence, Italy. Association for Computational Linguistics.

A Training Details

Training details follow those by Joshi et al. (2019, 2020), and the hyperparameters adopted for each model during fine-tuning on OntoNotes are shown in Table 8. Specifically, each document in OntoNotes is divided into non-overlapping segments of length `max_segment_len`. The segments are then encoded independently by BERT/SpanBERT to contextualized word embeddings and fed to the `c2f-coref` model (Lee et al., 2018). The models are fine-tuned for 20 epochs with a dropout rate of 0.3, `bert_learning_rate` on parameters in BERT/SpanBERT, and `task_learning_rate` on parameters in `c2f-coref`. The learning rates are linearly decayed. A batch size of one document is used, where each document is randomly truncated to contain `max_training_sentences` segments. All the experiments are conducted on one Tesla-V100 GPU with 32 GB memory.

For the debiasing method n-CDA, we adjust the bipartite graph matching method from Hall Maudslay et al. (2019) to fit the name list in our gender-guesser. Specifically, in our name list, each first name is assigned a label in {"male", "mostly male", "female", "mostly female"} indicating its gender specification, as well as a 55-dimensional frequency vector. The value in each dimension is an integer in $[0, 13]$ that indicates the name's relative frequency in one of the 55 countries. Below, we only describe how we match "male" with "female" names; the method to match "mostly male" with "mostly female" names is the same. We build a bipartite graph where "male" and "female" names are nodes in distinct parts, and define the weight of an edge between a "male" and a "female" name as $w_{i,j} = \|v_i - v_j\|_2 \cdot (\alpha - cos\langle v_i, v_j\rangle)$, where v_i and v_j are the frequency vectors of the two names, and $\alpha > 1$ is a hyperparameter balancing the ℓ_2 and the cosine distance. Our motivation is to encourage a rare name to be matched with even a popular name in the same country other than another rare name in a different country, so we choose $\alpha = 12/11$. Finally, we leverage a minimum weight full matching algorithm (Kuhn, 1955) to compute the matches between the names in the two parts.

B Amazon Mechanical Turk Annotation Details

Our annotation instructions and a sample Human Intelligence Task (HIT) interface are shown in Figure 2. To ensure the annotation quality, we implement a series of on-submission checks including checks on whether the selected span is a personal name, whether multiple entities are selected, whether the "no names are coreferent" box is misused, and so on. We require annotators to have at least a 95% approval rate with more than 50 approved HITs. The average time an annotator spent on one HIT is around 30 minutes.

C Illustration of the Inconsistency Metrics

A conceptual illustration of the proposed inconsistency metrics is shown in Figure 1: inconsistency across genders (I_{across}) measures inconsistency in instance pairs containing two instances of different genders, while inconsistency within genders (I_{within}) measures inconsistency in instance pairs containing two instances of the same gender.

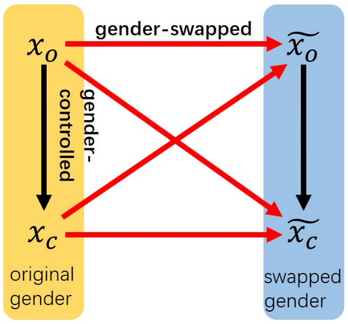
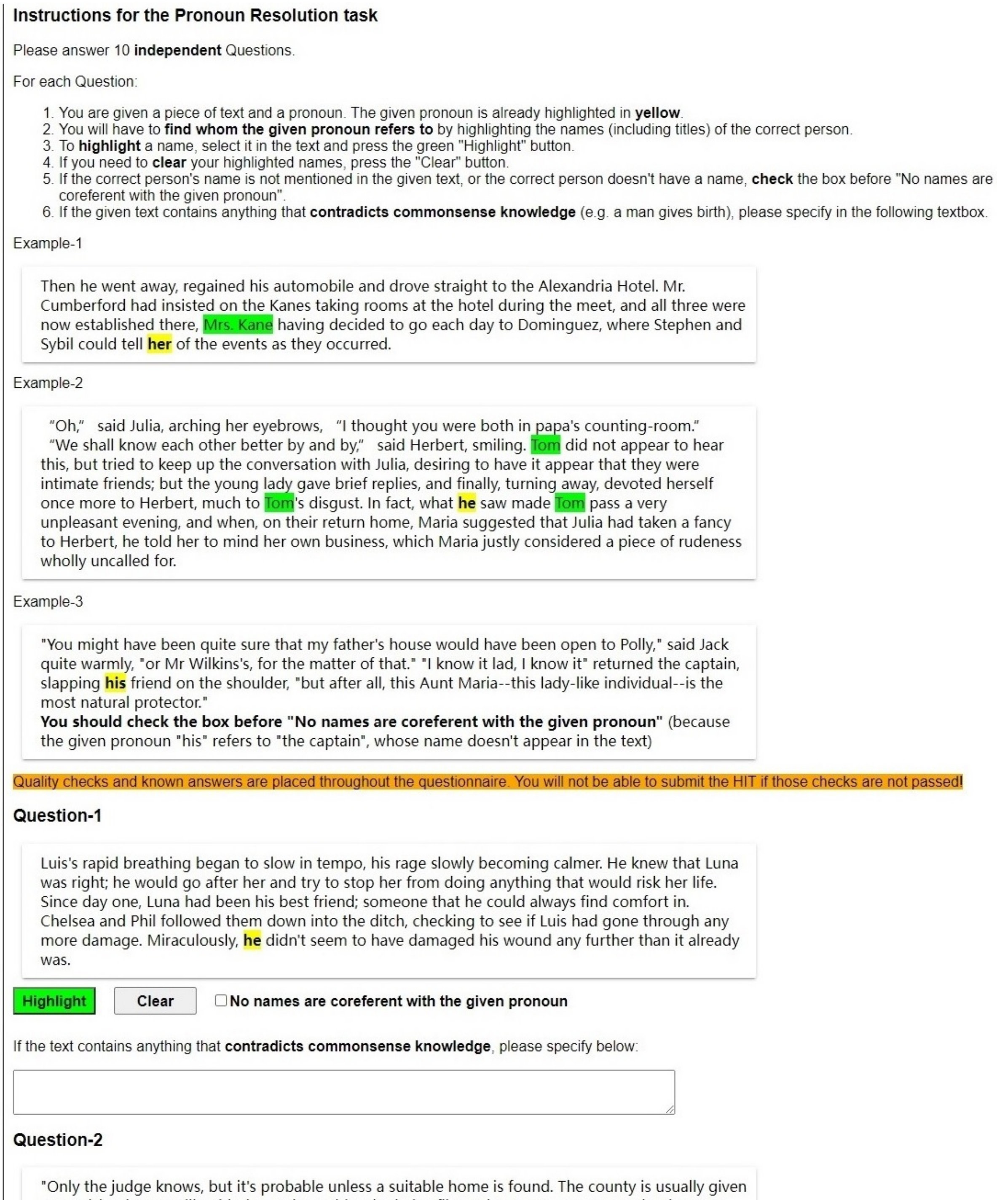


Figure 1: Illustration of the two inconsistency metrics. When computing inconsistency within genders, we use instance pairs linked by the two black arrows; when computing inconsistency across genders, we use instance pairs linked by the four red arrows.

	BERT-base	BERT-large	SpanBERT-base	SpanBERT-large
$bert\_learning\_rate$	1e-5	1e-5	2e-5	1e-5
$task\_learning\_rate$	2e-4	2e-4	1e-4	3e-4
$max\_segment\_len$	128	384	384	512
$max\_training\_sentences$	11	3	3	3

Table 8: Hyperparameters for fine-tuning.



Instructions for the Pronoun Resolution task

Please answer 10 **independent** Questions.

For each Question:

1. You are given a piece of text and a pronoun. The given pronoun is already highlighted in **yellow**.
2. You will have to **find whom the given pronoun refers to** by highlighting the names (including titles) of the correct person.
3. To **highlight** a name, select it in the text and press the green "Highlight" button.
4. If you need to **clear** your highlighted names, press the "Clear" button.
5. If the correct person's name is not mentioned in the given text, or the correct person doesn't have a name, **check** the box before "No names are coreferent with the given pronoun".
6. If the given text contains anything that **contradicts commonsense knowledge** (e.g. a man gives birth), please specify in the following textbox.

Example-1

Then he went away, regained his automobile and drove straight to the Alexandria Hotel. Mr. Cumberford had insisted on the Kanes taking rooms at the hotel during the meet, and all three were now established there, Mrs. Kane having decided to go each day to Dominguez, where Stephen and Sybil could tell **her** of the events as they occurred.

Example-2

"Oh," said Julia, arching her eyebrows, "I thought you were both in papa's counting-room." "We shall know each other better by and by," said Herbert, smiling. Tom did not appear to hear this, but tried to keep up the conversation with Julia, desiring to have it appear that they were intimate friends; but the young lady gave brief replies, and finally, turning away, devoted herself once more to Herbert, much to Tom's disgust. In fact, what **he** saw made Tom pass a very unpleasant evening, and when, on their return home, Maria suggested that Julia had taken a fancy to Herbert, he told her to mind her own business, which Maria justly considered a piece of rudeness wholly uncalled for.

Example-3

"You might have been quite sure that my father's house would have been open to Polly," said Jack quite warmly, "or Mr Wilkins's, for the matter of that." "I know it lad, I know it" returned the captain, slapping **his** friend on the shoulder, "but after all, this Aunt Maria--this lady-like individual--is the most natural protector."
You should check the box before "No names are coreferent with the given pronoun" (because the given pronoun "his" refers to "the captain", whose name doesn't appear in the text)

Quality checks and known answers are placed throughout the questionnaire. You will not be able to submit the HIT if those checks are not passed!

Question-1

Luis's rapid breathing began to slow in tempo, his rage slowly becoming calmer. He knew that Luna was right; he would go after her and try to stop her from doing anything that would risk her life. Since day one, Luna had been his best friend; someone that he could always find comfort in. Chelsea and Phil followed them down into the ditch, checking to see if Luis had gone through any more damage. Miraculously, **he** didn't seem to have damaged his wound any further than it already was.

☐ **No names are coreferent with the given pronoun**

If the text contains anything that **contradicts commonsense knowledge**, please specify below:

Question-2

"Only the judge knows, but it's probable unless a suitable home is found. The county is usually given

Figure 2: A sample HIT interface and annotation instructions.

The NLP Task Effectiveness of Long-Range Transformers

Guanghui Qin **Yukun Feng** **Benjamin Van Durme**
Department of Computer Science, Johns Hopkins University
`{gqin2, yfeng55, vandurme}@jhu.edu`

Abstract

Transformer models cannot easily scale to long sequences due to their $O(N^2)$ time and space complexity. This has led to Transformer variants seeking to lower computational complexity, such as Longformer and Performer. While such models have theoretically greater efficiency, their effectiveness on real NLP tasks has not been well studied. We benchmark 7 variants of Transformer models on 5 difficult NLP tasks and 7 datasets. We design experiments to isolate the effect of pretraining and hyperparameter settings, to focus on their capacity for long-range attention. Moreover, we present various methods to investigate attention behaviors to illuminate model details beyond metric scores. We find that the modified attention in long-range transformers has advantages on content selection and query-guided decoding, but they come with previously unrecognized drawbacks such as insufficient attention to distant tokens and accumulated approximation error.

1 Introduction

Transformer-based models (Vaswani et al., 2017) have advanced the state of the art in natural language processing. However, their quadratic time and space complexity hinder their application on long texts. Various proposals have been made to address these concerns (Tay et al., 2020c), with mathematical guarantees on improved time or space. These models have been evaluated primarily via perplexity (Dai et al., 2019) and non-NLP benchmarks (Tay et al., 2020b). These metrics may not be ideal (Sun et al., 2021) and may not reflect performance on complex NLP tasks (Arutiunian et al., 2020; Thorne et al., 2021). We argue these metrics have not been sufficient for the development of efficient Transformers and their practical application on long texts, and that existing benchmarks are insufficient guides for architecture selection.

It is not straightforward to have a fair and side-by-side comparison among those models due to the differences between their pretraining and hyperparameter settings (Tay et al., 2020c), and the metrics alone cannot convey detailed information about the self-attention blocks (Sun et al., 2021). We wish to fairly validate the effectiveness of long-range attention techniques, and to uncover the underlying factors behind model behaviors. We critique the reliance on perplexity evaluations in previous work, experimenting with five difficult, long-text NLP tasks. These tasks cover typical NLP modeling scenarios: token or span-level prediction, sequence-level classification, and sequence-to-sequence generation. To our knowledge, this is the first work to evaluate long-range transformers on such a wide spectrum of representative NLP tasks.

To verify the key features of long-range transformers, we *ablate* distant attention to measure what they gain from long-range mechanisms. For models without pretrained checkpoints, we migrate parameters from their prototype models for fairness. We cover 3 main kinds of long-range transformers, including pattern-, recurrence-, and kernel-based methods. To our knowledge, we are the first to adopt all these methods to probe transformers. To investigate the relationship between performance and document lengths we break down the metric with a customized algorithm (Bagga and Baldwin, 1998). Also, we use entropy and attribution analysis (Li et al., 2017) to test the effectiveness of cached memories in recurrent transformers and the global tokens for query-based problems.

We find that long-range context brings performance gains to transformers in some cases, which we attribute to more selective attention, especially for query-based tasks like QA. Surprisingly we observe that some long-range models do not effectively utilize distant information, and the accumulated error of approximation is unacceptable. We hope this analysis helps practitioners better understand the current state of the art of long-range attentions and suggests paths for future research.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3774–3790
May 2-6, 2023 ©2023 Association for Computational Linguistics

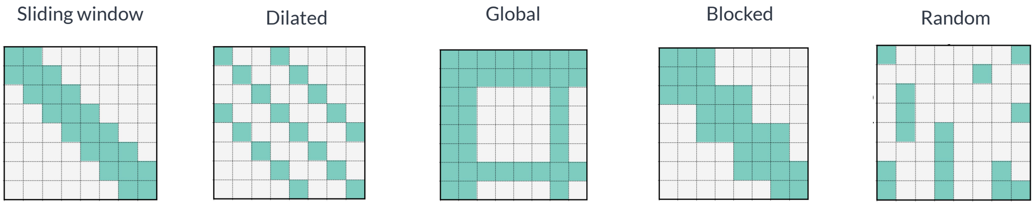


Figure 1: Illustration of 5 patterns used by long-range transformers, from Beltagy et al. (2021) with permission.

2 Background

2.1 Long-Range Transformers

Researchers have proposed a number of Transformer variants (Tay et al., 2020c). Most of these models support decoding (causal masking) (Peng et al., 2021a), while only a few of them have pretrained checkpoints (Beltagy et al., 2020, inter alia). We cluster these approaches into 3 main categories.

Sparsified Patterns Pattern-based methods try to make self-attention sparse. Some apply prespecified attention patterns. Specifically, Longformer (Beltagy et al., 2020) applies 3 patterns: *Sliding window* requires that each token can only attend to the tokens in a local window, *dilated pattern* lets each token only attend at fixed intervals, while the *global pattern* requires a few tokens as globally attended and lets them to attend all tokens in the sequence. In addition to the global pattern, BigBird (Zaheer et al., 2020) applies a *blocked pattern*, which splits the sequence into fixed-length blocks, and *random patterns*, by which tokens can attend to any other tokens randomly. An illustration is shown in fig. 1. Although the attention of each layer is not full, the receptive field can be increased as multiple layers are stacked. The selected or appended "global" tokens can be task-specific (Beltagy et al., 2020), allowing for direct distant information exchange. Instead of pre-defined attention patterns, some use content-based patterns so they become learnable, with techniques including locality sensitive hashing (Kitaev et al., 2020), the differentiable Sinkhorn algorithm (Tay et al., 2020a), or the learnable routing algorithm (Roy et al., 2020).

Recurrence & Compressed Memory These methods use segment-level recurrence to reuse the cached hidden states of previous steps. Transformer-XL (Dai et al., 2019) and XLNet (Yang et al., 2019) connects different chunks with cross-attention, where the tokens in a block attend to the hidden states of the previous blocks in addition to their self-attention. Note that the gradients remain in the same segment and are not propagated to previous segments (fig. 2). To reduce the number of history hidden states, Rae et al. (2020) compress them as memories for efficient re-use with pooling or convolutions. From a different perspective, Izacard and Grave (2021) use retrieval-based methods to collect evidences from external knowledge, resulting in a more targeted context information.

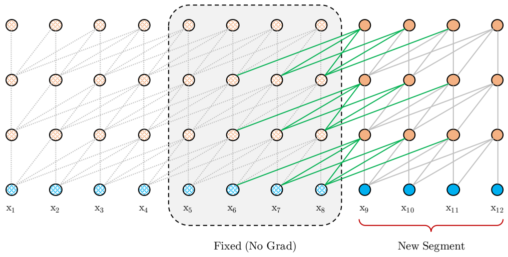


Figure 2: Recurrent transformers. "No Grad" means that the gradients do not back-propagate to this block. Obtained from Dai et al. (2019) with permission.

Low-Rank & Kernels These methods approximate the self-attention with low-rank approximation (Wang et al., 2020) or kernelization without explictly computing the matrix production. Among them, Choromanski et al. (2021); Peng et al. (2021b) use random features. Katharopoulos et al. (2020) reduce the time complexity to linear and space complexity to constant by replacing softmax with linear kernel features.

2.2 Benchmarks and Analysis

There is not an agreed-upon standard benchmark for long-range transformers. Researchers have considered various tasks and domains, including language (Dai et al., 2019), protein sequences (Choromanski et al., 2021), and images (Katharopoulos et al., 2020). Few conduct experiments on long sequence NLP tasks, including question answering (Beltagy et al., 2020) and summarization (Zaheer et al., 2020). Tay et al. (2020b) propose Long Range Arena comprised of six non-NLP tasks to exclude the factor of pretraining. Concurrent to our efforts, Shaham et al. (2022) propose a suite of text-to-text tasks as a long sequence NLP benchmark.

Researchers are also interested in the utility of context in transformers. Rae and Razavi (2020) find that Transformer-XL does not necessarily need long and deep contexts. Sun et al. (2021) reveal that Longformer and Routing transformers can only reduce the perplexity of LMs on a small set of tokens. More related to our work, Lai et al. (2020) show that BERT can make use of a larger scope of context than a BiLSTM.

3 Setup

3.1 Settings

It is non-trivial to compare distinct transformer models, due to differences between their pretraining and hyper-parameter settings. Our goal is to minimize these confounding factors to allow a focus on the long range attention ability of each model on different tasks. We therefore propose two sets of experimental conditions.

Restricted Attention Range To evaluate the performance gain from long-range attention, we evaluate models in both their default *context-aware* settings and a *context-agnostic* setting that restricts the receptive range of the self-attention blocks. For pattern-based transformers, we achieve the restriction by segmenting the input sequence into chunks and running the transformers on segments independently. For recurrent models, we ablate the recurrence to eliminate the dependencies between segments. In practice, we segment the texts into chunks with length L,[1] which ranges from 128 to 1536. L=∞ indicates no segmentation is used.

Parameter Migration Kernel-based models usually do not come with checkpoints for general tasks, but they may have similar structures to other pretrained models like BERT and may be designed to approximate the original results. Therefore, it is feasible to migrate parameters from pretrained prototypes to their "efficient version" to observe if the performance could be preserved. This type of method can be suitable for models without additional parameters, such as Performer.

3.2 Transformers and Tasks

Transformers We consider three approaches: 1) *pattern-based*: Longformer (Beltagy et al., 2020), and BigBird (Zaheer et al., 2020); 2) *recurrent*: XLNet (Yang et al., 2019); and 3) *kernel-based*: Performer (Choromanski et al., 2021). We also include the results of RoBERTa (Liu et al., 2019) and SpanBERT (Joshi et al., 2020), where some of our approaches are initialized from those two non-long-range models. Due to memory requirements, we use the base version for all models.[2] You may refer to appendix B for more details.

Dataset	Task	#tokens	#docs
Ontonotes	Coref.	467	3493
TriviaQA	eQA	2895	95k
DocNLI	NLI	399	1.44m
SummFD	Summ.	5.6k	4.3k
GovRep	Summ.	7.9k	19k
Qasper	aQA	3.7k	5.7k
QuALITY	aQA	4.2k	6.7k

Table 1: Task and dataset overview. The #tokens is the number of tokens per doc on average.

Tasks We cover five tasks of three types, including 1) *span-level* predictions: coreference resolution (Coref.) (Weischedel et al., 2011) and extractive question answering (eQA) (Joshi et al., 2017); 2) *sequence classification*: natural language inference (NLI) (Yin et al., 2021);[3] and 3) *seq2seq*: summarization (Summ.) (Chen et al., 2021; Huang et al., 2021), abstractive QA (aQA) (Dasigi et al., 2021; Pang et al., 2022). We pick seven datasets that involve long texts, whose statistics are shown in table 1. For more details about the data preprocessing, please refer to appendix A.[4]

4 Experiments

4.1 Coreference Resolution

Coreference resolution (coref.) is the task of identifying mention spans and clustering them into entities. We consider multiple coreference strategies: 1) the widely used Coarse2Fine (C2F) method[5] (Lee et al., 2018) which relies on span representations and 2) the current state-of-the-art method called Start2End (S2E) (Kirstain et al., 2021) that works on token representations. The

---

[1] We use wordpieces instead of words in this paper.

[2] We used the codebase of Katharopoulos et al. (2020) for Performer and Huggingface (Wolf et al., 2020) for the rest.

[3] DocNLI is modified from ANLI (Nie et al., 2020), SQuAD (Rajpurkar et al., 2016), DUC2001, DailyMail (Nallapati et al., 2016), and Curation (Curation, 2020)

[4] Our codebase is available on `https://github.com/hiaoxui/long-range-transformers`.

[5] We used the re-implementation by Gardner et al. (2018).

	Encoder	L=128	L=256	L=512
Model: Coarse2Fine	Longformer	75.74	76.72	77.36
	Longformer^G	75.68	76.25	77.23
	BigBird	75.95	76.78	77.64
	XLNet	74.57	74.48	74.33
	XLNet^m	74.73	75.76	76.29
	RoBERTa	74.64	76.45	76.83
	RoBERTa^p	51.58	51.71	50.39
	SpanBERT	75.04	75.84	76.59
	SpanBERT^p	52.46	52.06	50.51
	Longformer	76.77 (1024)		76.32 (∞)
	BigBird	77.31 (1024)		77.57 (∞)
Model: Start2End	Longformer	74.77	76.27	77.73
	Longformer^G	74.15	76.19	77.41
	BigBird	73.68	75.57	77.40
	XLNet	45.89	60.05	68.23
	XLNet^m	52.61	56.37	66.91
	RoBERTa	71.96	76.27	77.78
	RoBERTa^p	40.06	42.35	41.69
	SpanBERT	68.70	74.27	75.32
	SpanBERT^p	38.69	41.93	42.10
	Longformer	77.54 (1024)		77.57 (∞)
	BigBird	77.43 (1024)		77.66 (∞)

Table 2: Coref. results with the C2F and the S2E models. Numbers are averaged F1 (MUC, B^3, and CEAF$_{\phi_4}$). Longformer with ^{G} uses global tokens. XLNet with ^{m} uses recurrence memory. Encoders with ^{p} have their self-attention replaced with a Performer kernel.

dataset we use is **Ontonotes 5.0**. Transformers considered include **Longformer**, **XLNet**, and **Performer**. We migrate the parameters of **SpanBERT** and **RoBERTa** to Performer and include results on these models as well. We segment the input tokens into chunks with lengths of L . For models with global tokens, we lack a natural choice so we consider all tokens to be global. For XLNet, we keep the memory of the same length of the segments (e.g. we keep a memory length of 256 for a model with segment length 256).[6] The results are shown in table 2. Refer to appendix F.1 for complete results.

Some observations are consistent across two coref. models. 1) Though further pretrained upon RoBERTa, pattern-based methods do not show improvement over RoBERTa, even with longer attention range. 2) Models gain advantage when the segments get longer, but it is saturated when the segment length reaches 512. Distant contexts might not be exploited. 3) Performer-based models under-perform their corresponding non-kernelized models by a huge gap. 4) XLNet performs better with cached memory, but the performance gain is less observable when for shorter segments. [7]

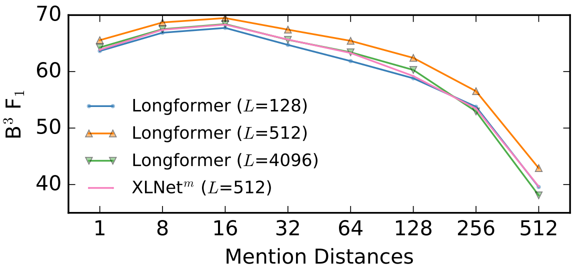


Figure 3: B^3 breakdown scores of 4 models for mention pairs with ranges from $[1, 8)$ to $[512, \infty)$.

To further show the performance of those models on documents with different lengths, we conduct metric breakdown with a few typical configurations. Instead of simply clustering the document according to the lengths, we propose a breakdown-version B^3 metric. Given a mention distance range $[L_1, L_2]$, we calculate its corresponding B^3 value by only considering the mention pairs whose distances fall into this range. The breakdown metrics are shown in fig. 3. We can see that the performance of all models follows the same trend and peaks at the $[16, 32)$ bucket. Also, the graph shows that Longformer with longer context encoding does NOT benefit on distant mentions (p value < 0.01). [8] On the contrary, they suffer more from distance than shorter-context models, showing that long-range attention fails to capture long-distance information.

4.2 Natural Language Inference

NLI is a classification task concerning a premise and hypothesis with variable lengths. The **DocNLI** dataset uses document-length inputs with binary labels (`entailment` and `not entailment`). We adopt the model proposed by Yin et al. (2021), and consider **Longformer**, **BigBird**, **Performer**, and **XLNet**. We make the prediction on the `CLS` token, which is at the beginning for Longformer and the end for XLNet. Since we are only interested in the encoding of `CLS`, we adopt the strategy of Yin et al. (2021) which truncates the sequence to L while preserving the hypothesis for Longformer and RoBERTa. The results are shown in table 3.

---

[6] We adopt the same strategies with segmentation and memories in the remainder of the paper.

[7] The observations on XLNet and Performer are consistent across all the tasks in this paper.

[8] We conduct significance test for the comparison between curves. Please refer to appendix D for more details.

Encoder	L=128	L=256	L=512
XLNet	29.95	40.39	24.31
XLNet^m	32.94	45.97	30.42
RoBERTa	48.96	47.78	46.04
RoBERTa^p	17.83	24.91	23.65
Longformer	29.11	25.73	45.28
BigBird	28.95	24.71	31.72
Longformer	45.96 (1024)	44.42 (∞)	
BigBird	33.58 (1024)	18.08 (∞)	

Table 3: F1 scores on the test set for DocNLI.

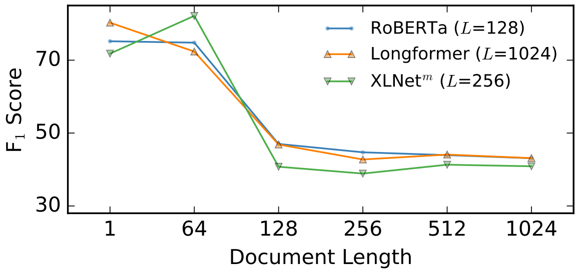


Figure 4: The breakdown analysis of DocNLI. We pick the best configuration for each model for brevity.

Observing table 3, surprisingly, the best performance is achieved with short segments, though the optimal lengths vary from model to model. Also, the performance of Longformer and BigBird is much lower than their baseline RoBERTa (c.f. Yin et al. (2021)). Observing the breakdown analysis in fig. 4, the performance of 3 best models follow the same trend w.r.t. the document length.[9] We speculate that all models are unable to comprehend the relationship between long documents, and long-range attention does not bring any advantage.

4.3 Question Answering

For question answering we experiment with eQA (**TriviaQA**) and aQA (**Qasper** and **QuALITY**). **Performer**, **Longformer**, **BigBird**, and **XLNet** are tested for encoder-only tasks; **BART** and Longformer-Encoder-Decoder (**LED**) is used for seq2seq tasks. For Longformer^G and BigBird, we set the question text (and candidate answers for QuALITY) as global tokens.[10] **TriviaQA** is a question answering dataset that involves extracting answer spans from reference documents. We adopt the method and codebase of Joshi et al. (2017). F1 is used as evaluation metrics. **Qasper** addresses the QA task in the domain of academic papers and involves various answer types: extractive, abstractive, boolean, and unanswerable. We unify such tasks as an abstractive QA task (cf. Shaham et al. (2022)) and implement an LED-based decoder to generate answers. F1 score is used for the evaluation of Qasper. **QuALITY** is a multiple-choice QA task. Given a question and a passage, the task is to select the correct answer from several candidates. We regard it as a seq2seq problem, with the objective to predict the correct answer conditioned on the concatenation of query, candidate answers, and passage. During inference, the answer with the least perplexity is selected. All results are shown in tables 4 and 5, and the full results with more metrics are shown in appendix F.2.

Encoder	L=128	L=256	L=512
Longformer	54.26	58.83	63.88
RoBERTa	55.81	60.29	63.45
RoBERTa^p	23.17	21.87	21.11
BigBird	55.28	59.39	63.51
XLNet	51.46	56.26	60.05
XLNet^m	52.71	57.96	62.85
Longformer	63.91 (1024)	63.66 (∞)	
Longformer^G	- (1024)	72.96 (∞)	
BigBird	66.50 (1024)	71.78 (∞)	

Table 4: Results for TriviaQA. Longformer^G indicates that the Longformer sets question as the global tokens.

	Chunk	512	1024	1536	∞
Qasper	BART	24.70	26.30	-	-
	LED	8.40	15.80	17.86	18.79
	LED^G	-	-	-	28.64
QLTY	BART	26.80	26.00	-	-
	LED	30.73	31.35	31.78	31.21
	LED^G	-	-	-	29.87

Table 5: Performance Qasper and QuALITY. Qasper is evaluated with F1 and QuALITY with accuracy. The results on BART are from Shaham et al. (2022).

Across all results, we find that larger receptive fields lead to better performance in most cases. More importantly, setting queries as global tokens greatly benefits the performance on both eQA and aQA. For QuALITY, it slightly hurt the perfor-

---

[9] The graph looks less smooth than fig. 3, possibly because DocNLI is made up of examples pulled from different datasets, which may have examples of different average lengths (cf. tab 1 in Yin et al. (2021)). Therefore the length of an example in DocNLI may correlate with different domains making up the dataset, which would interfere with our analysis. Future work will consider other datasets without this confounding concern.

[10] The global tokens of BigBird are fixed in the first 2 blocks, so we place the query at the beginning of the sequence.

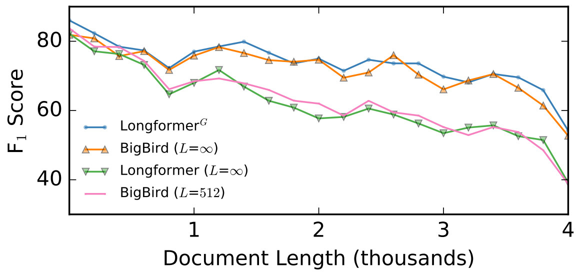


Figure 5: The performance of Longformer and BigBird on different lengths of TriviaQA documents. Note that Longformer^G and BigBird (L=∞) have global tokens.

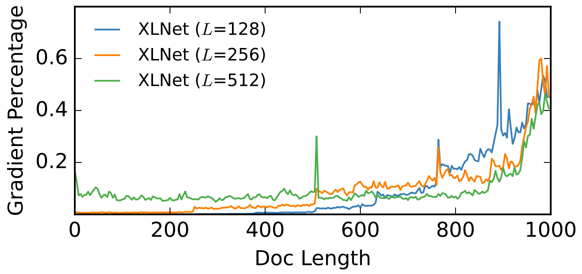


Figure 6: The gradient distribution over tokens for XLNet^m on DocNLI documents.

	Chunk	512	1024	1536	∞
SF	BART	26.30	27.20	-	-
	LED	32.81	33.07	33.22	33.57
GR	BART	45.60	47.90	-	-
	LED	53.86	54.13	54.83	56.60

Table 6: Results for Summarization on SummFD and GovRep. ROUGE unigram is used as the metric. Results on BART are from Shaham et al. (2022).

mance to set both query and candidate as global tokens. We reckon too many global tokens might introduce more noise, similar to the case of coref.

We speculate that the performance gain mostly comes from enhanced attention to the query, which if further verified by the metric breakdown that is shown in fig. 5. All models perform well on short texts, while models with global tokens obtain an observably greater advantage over the baseline models for longer documents (p value < 0.01). We think that the global token mechanism could help the model be less distracted on long texts via more attention on the queries (section 5.3), which consequently improves the performance.

4.4 Summarization

As a typical seq2seq problem, we adopt **LED** to perform the summarization task. We chunk the source sequence into segments (no segmentation for $L = \infty$) to restrict the receptive range. Intuitively, the summary may be benefited from the contextual representation with a broader view of the document. Two datasets are used: **SumScreen** addresses the domain of TV shows. Following Shaham et al. (2022), we use the subset of ForeverDreaming (SummFD) consisting of 88 different shows. The goal is to summarize the transcript of an episode, for which the recap is used as the ground truth summary. **GovReport** is a long-document summarization dataset in the domain of government policies with human-written summaries. ROUGE (Lin, 2004) is used as the evaluation metric. Results are shown in Table 6, and the full results are shown in appendix F.3.

From table 6, we observe slight superiority of the context-aware models. Note that cross attention will attend to all source tokens whether we segment it or not, but the intuition of summarization is to *skim over* the document, so we speculate that the performance improvement may be related to the *selectivity* of the encoder-side attention, which is further analyzed in section 5.2. We do not conduct breakdown analysis for summarization because the sequence length directly contributes to the metric.

5 Analysis

5.1 The Attribution of Recurrence Memories

Even if we know that distant contexts can help or hurt performance, it's still unclear *how much* they contribute to the predictions. One way to quantify this is *attribution analysis* (Simonyan et al., 2014; Li et al., 2017). Suppose ℓ is our loss function and $\mathbf{e}_i \in \mathbb{R}^d$ is the word embedding of the i-th token. We use α_i to measure the attribution of the i-th token to the final prediction where

$$\alpha_i = \left\| \frac{\partial \ell}{\partial \mathbf{e}_i} \right\|_l . \tag{1}$$

We set $l = 1$ to take the L_1 norm in practice, and the ground truth labels in the test set are used to calculate the gradient. Intuitively, tokens with higher contribution have greater gradient norms. Also, recurrent models like XLNet stop the gradient from being propagated back to the cached memory, and we temporarily turn off this feature for analysis.

We apply this method to XLNet on DocNLI, where we pick 128 documents of lengths between 1,000 and 1,024. We normalize the α_i over all tokens for each document, and take an average on

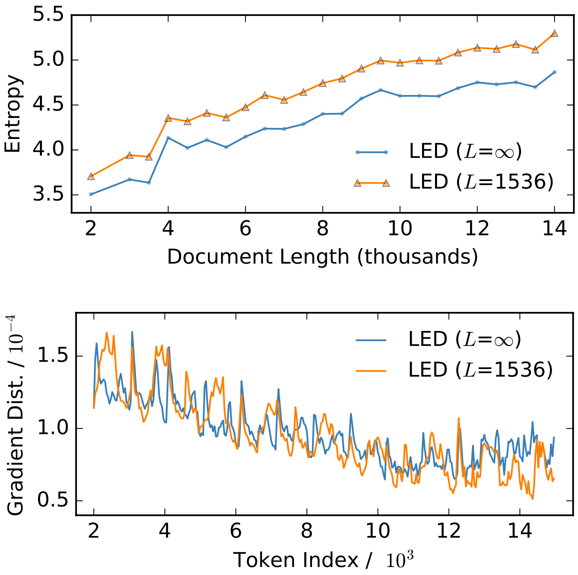


Figure 7: Attention distribution entropy (above) for documents with different lengths and attribution analysis (below) over source tokens for LED on SummFD.

each token index across all documents. The results are shown in fig. 6. Note that prediction is made in the last segment, and previous segments contribute only through the memory. We see the attribution of tokens is stratified according to their segment lengths, with a minor peak at the `CLS` token of each segment. As we have more and more segments, the attribution of distant tokens to the final prediction becomes negligible. For example, in the case of L=128, last segment made 53.4% of attribution, while the first segment made less than 0.01%.

5.2 Content Selection in Cross Attention

The cross attention of LED attends to the whole document no matter if we segment the inputs or not, thus we suppose source tokens should have similar attribution to decoding, which can be verified by *attribution analysis* in fig. 7. However, the crucial problem for summarization is whether the attention is *selective*, given that only a portion of the document is helpful. Therefore, we inspect the *entropy* [11] of the cross attention distribution over the source tokens fig. 7.

Reading the entropy curve, we find that the entropy of models without segmentation (L=∞) is consistently lower (p value < 0.01), which can be translated to higher selectivity of cross attention and explains the superiority of LED (L=∞) in table 6. Reading the gradient curve, we find that both models have a relatively uniform attribution over tokens with a slight slope on the left side due to the existence of short sequences. This is reasonable because summarization requires the decoder to skim over the whole document, thus cross attention should not have a locality preference.

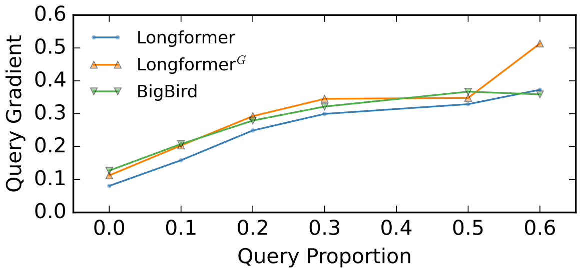


Figure 8: The accumulated gradient on query vs. the proportion of query, i.e. (query length / doc length) with Longformer and BigBird (L=∞) on TriviaQA.

5.3 Query-Guided Extraction and Decoding

Different from other problems, the queries in QA can be treated as a "guidance" on how to read the documents. To see if the encoders can exploit this structure, we conduct attribution analysis on TriviaQA with Longformer and BigBird. The results are shown on fig. 8, where we plot the relationship between the proportion of accumulated gradients on queries and the proportion of query tokens in the document. It is clear that models with global tokens pay more attention to queries (p value < 0.01 except for one exception), which is consistent with the purpose of their design.

For scenarios where seq2seq decoding meets queries, intuition suggests queries could instruct the cross-attention to attend to specific tokens, making the decoding more *selective*. From fig. 9, the entropy of attention distribution on source sequence against the doc length, we see Longformer with global token unquestionably has lower entropy (p value < 0.01), implying more targeted decoding.

5.4 Error Accumulation of Kernel Methods

We find that Performer could not match the results of its prototypes (tables 2 to 4). We suspect that the error incurred by the kernel approximation may not be acceptable for span-level tasks like coref. We conducted another set of experiments: Instead of training the performer model from the checkpoints of SpanBERT, we directly replace some layers from a fine-tuned SpanBERT model with Performer layers with their parameters preserved. Experiments

---

[11] The distribution entropy is averaged over decoding tokens, attention heads, and transformer layers.

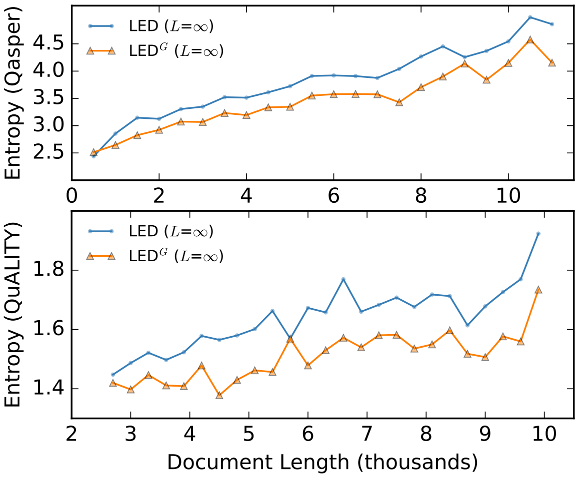


Figure 9: Entropy of source-side attention distribution vs. document length on Qasper and QuALITY.

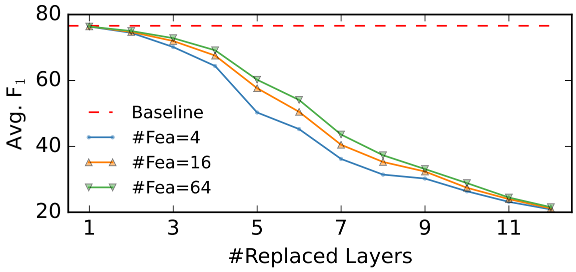


Figure 10: Results of SpanBERT (L=512) with layers replaced with Performer layers with "#Fea" features. The baseline is the performance of the original model.

are conducted with C2F on Ontonotes 5.0.

In fig. 10, we try to replace P layers of SpanBERT in a top-down manner, where $P = 1, 2, \ldots, 12$. We also try using different feature dimensions to exclude the possibility of insufficient features. We find that although large dimension of random features brings marginal advantages to the performance, Performer is not very sensitive to this factor. Instead, the performance drops dramatically as we replace more layers, from the baseline F1 78 to ~20 with all layers replaced. Based on our findings, we conclude that Performer is a good approximation for shallow transformers, even with very low-dimensional random features. However, as we stack more transformer layers, the accumulated errors can be unacceptable, which leads to a failure in the performance.

6 Experiment Confounders

Pretraining and Adaptation The purpose of this paper is to evaluate the effectiveness of different long-range attention approaches by reducing the confounder of pre-training, but it is still unclear how the results would change if we pre-train those models with the same configuration from scratch. Unfortunately, it requires much more resources and introduces more confounders of training settings, and we adopt the most straightforward way to ablate the long-range attention or migrate parameters. For each model-task pair, we make the most natural choice, e.g. setting queries as global tokens, trying to minimize human biases of model adaptations.

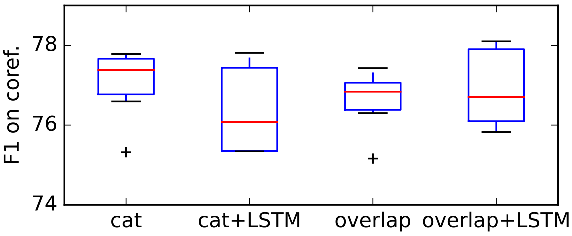


Figure 11: Results of different pooling strategies.

Pooling Strategies Concatenating the representation of segments, while being natural and commonly used, is not the only pooling strategy. As a comparison, we incorporate the results of other 3 solutions: 1) Split the documents into segments of L tokens with $L/2$ overlapped between adjacent segments (Joshi et al., 2019); 2) Stack an LSTM layer over the Transformer representation of the segment ; 3) A combination of above methods. We conduct experiments on the coreference resolution task with both C2F and S2E solution and many variants of transformers as our encoder. We leave the experiment details and discussions in appendix E and show a brief result in fig. 11. We have similar observation with Joshi et al. (2019) that overlapping does not bring performance improvement. Moreover, though introducing more parameters, a stacked LSTM even hurts the performance. We conclude that pooling strategies do not affect our analysis in section 5.

7 Discussion

Researchers have proposed many innovative methods for efficient self-attention over long sequences. The key ideas work as desired in certain cases, though we demonstrate several drawbacks.

Surprisingly, **pattern-based methods**, as the most popular approach, are not necessarily benefited from long-range attention in the general case. Larger sliding windows are helpful, but the benefit would quickly saturate or become negative (table 2).

However, when a small portion of *guidance text* (e.g. query in QA) exists, setting it as *global tokens* can make it more attended and significantly boost the performance (section 5.3). When such text doesn't exist, setting all tokens as global would hurt the performance (table 2). Moreover, we find that long-range attention and global tokens are correlated with the *selectivity* of *seq2seq* problems, which consequently helps the decoding.

The memory of **recurrence models** generally improves performance, proving historical hidden states are beneficial for transformers in various tasks. However, XLNet does not fully exploit the history tokens, with distant information less attended (section 5.1). We speculate that it is because XLNet is pretrained to predict masked tokens, which does not frequently need the participation of long-range context (Sun et al., 2021). Also, the stop-gradient trick may hinder the model from effectively attending to memories.

The approximation of **kernel-based methods** works very well for shallow networks, but faces serious error accumulation problems when transformer layers are deeply stacked, which cannot be remedied by having high-dimensional random features (section 5.4). The resulting performance drop is not acceptable even for the "base" version of transformer encoders with 12 layers (table 2).

8 Conclusion

We conduct experiments with various long-range transformers on NLP tasks that involve long sequences, trying to fairly evaluate their long-range attention ability. While some methods are validated on certain tasks, we also find some previously unrecognized drawbacks. We further analyze the attention behaviors of these transformers with metric breakdown, attribution analysis, and entropy analysis, revealing the performance of those models might be correlated with the attribution of distant tokens, selectivity of attentions, or the approximation errors. We hope our work would shed light on the future development of long-range transformers.

Model Selection Takeaways Based on our findings, we have the following suggestions. For common tasks, such as sequence classification or token-level prediction, it is still competitive to chunk the inputs and apply short-range transformers. When explicit guiding text, such as queries, exists, pattern-based models with global token mechanism is preferred. For seq2seq problems, long-range transformers with pretrained checkpoints deliver better performance.

Acknowledgement

We appreciate the proofreading done by Patrick Xia, Marc Marone, Nils Holzenberger, Elias Stengel-Eskin, Yunmo Chen, and Zhengping Jiang . Thanks to the anonymous reviewers for their valuable feedback.

This work was supported in part by IARPA BETTER (#2019-19051600005). The views and conclusions contained in this work are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, or endorsements of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes.

9 Limitations

Energy Cost Our experiments involve a massive amount of training with many transformers on various tasks. Although we don't conduct any pretraining, the energy cost is still non-negligible. However, our hope is our findings here would enable others to more efficiently select a particular architecture for their task, rather than reproducing the work done here.

Experimental Bias Due to the lack of pretrained checkpoints for general purposes, we focus on representative instead of each type of transformer variant. It is possible that these observations are particular to specific artifacts and implementations considered here. Our goal is foremost to provide a roadmap for continued study on questions raised in this article, with new architectures being evaluated in the future by model developers themselves. For similar reasons, existing artifacts are biased towards English, as are many of the datasets employed in this study. We do not believe our findings are specific to English, but it remains for future work in long range transformer evaluation to extend our analysis into multilingual conditions.

Language Bias For similar reasons, existing artifacts are biased towards English, as are many of the datasets employed in this study. We do not believe our findings are specific to English, but it remains for future work in long-range transformer evaluation to extend our analysis into multilingual conditions.

References

Artashes Arutiunian, Morgan McGuire, Hallvar Gisnås, Sheik Mohamed Imran, Dean Pleban, Priyank Negi, and David Arnoldo Ortiz Lozano. 2020. Reproducibility Challenge: Reformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, page 10.

Amit Bagga and Breck Baldwin. 1998. Algorithms for Scoring Coreference Chains.

Iz Beltagy, Arman Cohan, Hanna Hajishirzi, Sewon Min, and Matthew Peter. 2021. Beyond Paragraphs: NLP for Long Sequences.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer.

Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2021. SummScreen: A Dataset for Abstractive Screenplay Summarization.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. 2021. Rethinking Attention with Performers. In *International Conference on Learning Representations (ICLR)*.

Curation. 2020. Curation Corpus Base.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. In *Association for Computational Linguistics (ACL)*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers. In *North American Association for Computational Linguistics (NAACL)*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A Deep Semantic Natural Language Processing Platform. In *Association for Computational Linguistics (ACL)*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient Attentions for Long Document Summarization. In *North American Association for Computational Linguistics (NAACL)*, pages 1419–1436. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In *European Association for Computational Linguistics (EACL)*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics (TACL)*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Association for Computational Linguistics (ACL)*.

Mandar Joshi, Omer Levy, Daniel S. Weld, and Luke Zettlemoyer. 2019. BERT for Coreference Resolution: Baselines and Analysis. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *International Conference on Machine Learning (ICML)*.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference Resolution without Span Representations. In *Association for Computational Linguistics (ACL)*.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The Efficient Transformer. In *International Conference on Learning Representations (ICLR)*.

Yi-An Lai, Garima Lalwani, and Yi Zhang. 2020. Context Analysis for Pre-trained Masked Language Models. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 3789–3804. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order Coreference Resolution with Coarse-to-fine Inference. In *North American Association for Computational Linguistics (NAACL)*.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2017. Visualizing and Understanding Neural Models in NLP. In *Association for Computational Linguistics (ACL)*.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, page 8.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. S. Zettlemoyer, and V. Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Xiaoqiang Luo. 2005. On Coreference Resolution Performance Metrics. In *Empirical Methods in Natural Language Processing (EMNLP)*, page 8.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. In *Computational Natural Language Learning (CoNLL)*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A New Benchmark for Natural Language Understanding. In *ACl*.

Richard Yuanzhe Pang, A. Parrish, Nitish Joshi, N. Nangia, J. Phang, A. Chen, V. Padmakumar, J. Ma, J. Thompson, H. He, and S. R. Bowman. 2022. QuALITY: Question Answering with Long Input Texts, Yes! In *North American Association for Computational Linguistics (NAACL)*.

Hao Peng, Jungo Kasai, Nikolaos Pappas, Dani Yogatama, Zhaofeng Wu, Lingpeng Kong, Roy Schwartz, and Noah A. Smith. 2021a. ABC: Attention with Bounded-memory Control.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. 2021b. Random Feature Attention. In *International Conference on Learning Representations (ICLR)*.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. 2020. Compressive Transformers for Long-Range Sequence Modelling. In *International Conference on Learning Representations (ICLR)*.

Jack W. Rae and Ali Razavi. 2020. Do Transformers Need Deep Long-Range Memory. In *Association for Computational Linguistics (ACL)*.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2020. Efficient Content-Based Sparse Attention with Routing Transformers. *Transactions of the Association for Computational Linguistics (TACL)*.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, and Omer Levy. 2022. SCROLLS: Standardized CompaRison Over Long Language Sequences.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps.

Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. 2021. Do Long-Range Language Models Actually Use Long-Range Context? In *Empirical Methods in Natural Language Processing (EMNLP)*, page 16.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. 2020a. Sparse Sinkhorn Attention. In *International Conference on Machine Learning (ICML)*.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020b. Long Range Arena: A Benchmark for Efficient Transformers.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020c. Efficient Transformers: A Survey.

James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, and Alon Halevy. 2021. Database Reasoning Over Text. In *Association for Computational Linguistics (ACL)*.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Conference on Message Understanding*, page 45. Association for Computational Linguistics.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-Attention with Linear Complexity.

R. Weischedel, E. Hovy, M. Marcus, M. Palmer, R. Belvin, S. Pradhan, L. Ramshaw, and N. Xue. 2011. OntoNotes: A large training corpus for enhanced processing. In *Handbook of Natural Language Processing and Machine Translation. Springer*. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 38–45. Association for Computational Linguistics.

Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wenpeng Yin, Dragomir Radev, and Caiming Xiong. 2021. DocNLI: A Large-scale Dataset for Document-level Natural Language Inference. In *Association for Computational Linguistics (ACL)*.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big Bird: Transformers for Longer Sequences. In *Advances in Neural Information Processing Systems (NeurIPS)*.

A Data Preprocessing

The coref task usually consumes a large amount of GPU memory. For the experiments on Ontonotes, we truncate the sequence longer than 2000 tokens (for c2f model) or 1400 tokens (for s2e model) for the memory concern. The truncation was applied only to the training set.

We didn't do any pre-processing steps for DocNLI (Yin et al., 2021), except for the truncation that we discussed in section 4. Notably, DocNLI is an aggregated dataset constructed from ANLI (Nie et al., 2020), SQuAD (Rajpurkar et al., 2016), DUC2001, CNN/DailyMail (Nallapati et al., 2016), and Curation (Curation, 2020). Documents from different sources may be distinguishable from their lengths (refer to tab 1 in Yin et al. (2021).

For TriviaQA, we use the scripts of Longformer (Beltagy et al., 2020) to pre-process the data. There is no further modifications on the data.

We adopt the dataset of QuALITY and GovReport from Shaham et al. (2022), which picked long sequences from those datasets.

For Qasper and SummScreen, we simply adopt the original dataset and scripts for preprocessing.

B Implementation Details of Transformers

In this section, we discuss the details of modification we did to the transformers.

Recurrence-Based Methods We tested XLNet for recurrence-based methods. We adopt the codebase of Huggingface as the base model, and followed the common strategy to stop gradient from being propagated back into the cached memory. For each segment of the recurrence, we appended two special tokens `SEP` and `CLS` to the sequence, making it like an ordinary input sequence to the XLNet model except for the possible existence of memory states. We concatenated the token representations after recurrences and remove the special tokens from all but the last segment. Empirically, we found that having special tokens can significantly boost the performance.

Pattern-Based Methods For the pattern-based methods, when we chunked the input sequence into segments, we appended the `SEP` as we did for the XLNet, and prepended the `CLS` token as a convention of other transformers. After concatenation, we removed all the special tokens except for the first `CLS` the last `SEP`, which made it structurally similar to the outputs of non-segmented transformers. What's more, for sliding window mechanism, we might reduce the window size to segment length if needed to save compute and memory.

Kernel-Based Methods We tested Performer (Choromanski et al., 2021) as a representative of kernel-based methods. Given that training from random initialization would lead to suboptimal results, we migrate the parameters from base models (e.g. BERT, RoBERTa) to kernel methods. In detail, we replace the self-attention layers of the base models with kernels. Given that Performer does not require any additional parameters, except for the orthogonal random feature vectors in the FAVOR+ mechanism. Following the default implementation of the fast transformer codebase[12], we set the feature dimension as the query dimension by default for main experiments, though we found the performance isn't sensitive to those features in section 5.4.

C Experiments Details

In this section, we introduce the details of our experiments in the main paper, including hyperparameters, training strategies, data split and loading, and other configurations that are necessary to replicate our results.

Computational Resources All of our experiments were done with NVIDIA RTX 6000 GPU with 24GB of memory. We did most of the experiments with single cards, except for TriviaQA, for which we used multi-GPU training with 4 cards.

Coreference Resolution We used two models: coarse2fine (C2F) (Lee et al., 2018) and start2end (S2E) (Kirstain et al., 2021). For C2F model, we used the codebase reimplemented by AllenNLP (Gardner et al., 2018) for its flexibility on encoder exchange. We used the official codebase of S2E model for other experiments. We didn't change any hypermeters except for the difference on the encoders. We adopted the same training strategy of these repos without any modifications, i.e. we train the model with certain epochs (40 for C2F and 129 for S2E) or until convergence and pick the model performed best on the dev set for evaluation. The typical training time of coref models was 10h for C2F and 24h for S2E.

---

[12] `https://github.com/idiap/fast-transformers`

Natural Language Inference Due to the size of DocNLI dataset (942k training and 234k dev examples), it's infeasible to adopt the common training strategies. Instead, we train the model with mini-batch gradient descent with a batch size of 4 for only one epoch. Because the training and dev set are too large to fit into CPU memory, we split the training set into smaller datasets consisting of 32768 examples, and iterate over training and dev set during training. We pick the model with best performance on the dev set (not the whole set but one iteration) for test. We use the whole test set consisting of 267k examples for the final evaluation. We adopt the same architecture as the model used in Yin et al. (2021) and reimplement it with AllenNLP Gardner et al. (2018). The typical training time is around 2 days and the test time is around 4 hours.

Question Answering For TriviaQA, we adopted the training scripts and hyperparameters used by Beltagy et al. (2020) except for that we set the training batch size as 4 and number of epochs as 8. The performance is evaluated on the dev set and we pick the best checkpoint with patience of 3.

For Qasper, we follow the training scripts and hyper-parameters in its official repository. [13] We disable the evidence setting, and extend the training to a maximum epoch of 20. The performance is evaluated on the dev set with patience of 5.

For QuALITY, we adopt the LED model with "allenai/led-base-16384" configuration from Huggingface. [14] We concatenate the question, candidate answers, and passages as the encoder input, and the correct answer as the decoder input for training. During inference, we feed each candidate answer as the decoder input, and consider the one with the lowest perplexity as the predicted answer. We use a warmup steps of 1000 and learning rate of 5×10^{-5} for training. Evaluation is performed on the dev set with a patience of 5.

Summarization For both the SummScreen and GovReport datasets, we use the LED model with "allenai/led-base-16384" configuration. We use a warmup steps of 1000 and learning rate of 5×10^{-5} for training and patience of 5 for testing. GovReport is evaluated on the dev set and SummScreen is evaluated on its official test set.

D Significance Test

For the comparison between curves in the section 5, we conduct significance test using bootstrapping methods to verify our conclusions. Let D be the test set for a task. For the performance comparison between two configurations, we sample a new D^* from D with replacement and we keep $|D^*| = |D|$. We treat the event "configuration A performs better than B" as a Bernoulli random variable P, and compute the probability of the null hypothesis $P < 0.5$ as the p value. We sample $B = 1024$ test sets for each comparison. If multiple significance tests are conducted, we only report the larges value that we obtain. [15]

For example, in fig. 3, we claim that the performance of Longformer (L=512) is better than any other encoders regardless of the mention distances. To verify it, we conduct significance test between Longformer (L=512) and other 3 encoders for every mention distance. The greatest p value among 24 p tests is smaller than 0.01, so our claim is secured by our significance test.

E More Pooling Strategies

We conduct experiments with 4 transformers, including 2 short-range transformers (RoBERTa and SpanBERT) and 2 long-range transformers (Longformer and BigBird) on the coreference resolution task. We set L=512, which is the maximum acceptable length for short-range transformers.

The full results are shown in table 7, and a box plot can be found in fig. 11. In overall, we have similar observations as Joshi et al. (2019) that overlapped segments do not offer improvements on the performance. Similar findings can be found for LSTM settings and the combination of them. More importantly, the performance difference of table 7 is consistently with tables 8 and 11 except for a few outliers. Thus, we conclude that direct concatenation is already enough to exploit the pretrained transformers, and changing pooling strategies do not greatly interfere our analysis.

F Full Experiment Results

In this section, we list the full results of all the experiments in the main paper.

---

[13] `https://github.com/allenai/qasper-led-baseline`

[14] `https://huggingface.co/allenai/led-base-16384`

[15] For the curves in fig. 8, we exclude one exception case at $x = 0.6$. For the curves in fig. 9, we exclude one exception case at $x = 500$.

F.1 Coreference Resolution

The full results of coreference resolution are shown in tables 8 and 11. We use MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and CEAF_{ϕ_4} (Luo, 2005) as the evaluation metrics. Following the convention, we use the "Avg." as the main metric, which is an average among the F1 score of 3 metrics. All the results are reported on the test set.

F.2 Question Answering

The full experiment results on TriviaQA is shown in table 9. We use both F1 and exact match (EM) as the metrics. A few cells are left blank because of the constraints of the transformers.

F.3 Summarization

The full results on summarization is shown in table 10. We used ROUGE (Lin, 2004) as the metric. R1, R2, and R3 stands for ROUGE unigram, bigram, and longest common subsequence. Note that 1536 is the windows size of the LED model, and 1024 is the maximum length supported by the BART model.

	Encoder	MUC			B^3			CEAF$_{\phi_4}$			Avg.
		P	R	F1	P	R	F1	P	R	F1	
Model: Coarse2Fine	RoBERTa	81.6	85.0	83.3	72.1	78.2	75.1	72.1	72.2	72.2	76.8
	RoBERTa$_{\text{overlap}}$	82.9	84.5	83.7	73.1	77.1	75.0	73.5	71.7	72.5	77.1
	RoBERTa$^{\text{LSTM}}$	84.3	83.9	84.1	75.0	76.2	75.6	74.5	71.0	72.7	77.5
	RoBERTa$^{\text{LSTM}}_{\text{overlap}}$	84.1	84.8	84.5	75.4	77.4	76.4	74.5	72.4	73.5	78.1
	Longformer	82.4	85.3	83.8	73.0	78.6	75.7	72.6	72.5	72.5	77.4
	Longformer$_{\text{overlap}}$	82.4	83.8	83.1	72.9	76.0	74.4	72.3	70.5	71.4	76.3
	Longformer$^{\text{LSTM}}$	84.2	83.8	84.0	75.3	75.8	75.5	73.8	71.7	72.8	77.4
	Longformer$^{\text{LSTM}}_{\text{overlap}}$	84.3	84.2	84.2	75.4	76.6	76.0	74.4	72.3	73.4	77.9
	BigBird	81.5	86.9	84.1	71.5	80.9	75.9	72.7	73.1	72.9	77.6
	BigBird$_{\text{overlap}}$	83.0	84.3	83.6	73.9	76.7	75.3	72.4	72.0	72.2	77.0
	BigBird$^{\text{LSTM}}$	84.2	84.4	84.3	75.0	77.1	76.0	74.0	72.1	73.1	77.8
	BigBird$^{\text{LSTM}}_{\text{overlap}}$	84.5	84.3	84.4	75.7	76.8	76.2	74.8	72.2	73.5	78.0
	SpanBERT	83.3	82.9	83.1	74.4	74.8	74.6	72.6	71.7	72.1	76.6
	SpanBERT$_{\text{overlap}}$	83.1	83.0	83.0	74.4	75.0	74.7	72.2	72.2	72.2	76.7
	SpanBERT$^{\text{LSTM}}$	83.4	82.8	83.1	74.3	74.7	74.5	72.8	70.9	71.8	76.5
	SpanBERT$^{\text{LSTM}}_{\text{overlap}}$	83.6	82.6	83.1	74.4	74.6	74.5	72.6	70.8	71.7	76.4
Model: Start2End	RoBERTa	85.7	82.6	84.1	78.1	74.3	76.2	75.2	70.9	73.0	77.8
	RoBERTa$_{\text{overlap}}$	85.0	82.6	83.8	77.7	73.5	75.6	74.4	71.5	73.0	77.4
	RoBERTa$^{\text{LSTM}}$	83.8	81.4	82.6	75.1	72.3	73.7	72.1	69.4	70.8	75.7
	RoBERTa$^{\text{LSTM}}_{\text{overlap}}$	83.7	82.9	83.3	75.6	74.6	75.1	73.7	71.4	72.5	77.0
	Longformer	85.5	82.6	84.0	78.0	74.5	76.2	75.2	70.7	72.9	77.7
	Longformer$_{\text{overlap}}$	83.9	82.2	83.1	75.5	73.3	74.4	73.8	69.9	71.8	76.4
	Longformer$^{\text{LSTM}}$	84.0	80.8	82.3	75.7	71.2	73.4	72.5	68.3	70.3	75.3
	Longformer$^{\text{LSTM}}_{\text{overlap}}$	83.7	81.6	82.6	75.3	72.5	73.8	72.5	69.5	71.0	75.8
	BigBird	85.7	82.3	84.0	77.9	73.9	75.9	75.7	69.4	72.4	77.4
	BigBird$_{\text{overlap}}$	84.1	82.4	83.2	76.3	74.2	75.2	74.7	70.8	72.7	77.1
	BigBird$^{\text{LSTM}}$	83.2	81.7	82.4	74.2	72.5	73.3	72.1	68.5	70.3	75.3
	BigBird$^{\text{LSTM}}_{\text{overlap}}$	83.7	81.9	82.8	75.5	73.0	74.2	73.3	69.9	71.6	76.2
	SpanBERT	83.5	81.3	82.4	74.6	71.9	73.2	72.3	68.5	70.3	75.3
	SpanBERT$_{\text{overlap}}$	82.9	81.2	82.0	74.0	72.3	73.2	72.2	68.5	70.3	75.2
	SpanBERT$^{\text{LSTM}}$	70.4	58.9	64.1	47.8	44.7	46.2	59.6	26.7	36.9	49.1
	SpanBERT$^{\text{LSTM}}_{\text{overlap}}$	68.3	61.5	64.7	43.6	47.8	45.6	59.3	26.3	36.5	48.9

Table 7: The full results of all experiments with different pooling strategies. All models use the segment length L=512. Models with superscript "LSTM" indicate it uses LSTM, and subscript "overlap" indicates it uses overlapped concatenation method. Note that both methods can be applied in the meantime.

Encoder	MUC			B^3			CEAF$_{\phi_4}$			Avg.
	P	R	F1	P	R	F1	P	R	F1	
BigBird (L=128)	80.5	85.5	83.0	69.8	78.5	73.9	70.8	71.2	71.0	75.9
BigBird (L=256)	81.1	86.2	83.6	70.5	79.8	74.9	72.1	71.7	71.9	76.8
BigBird (L=512)	81.5	86.9	84.1	71.5	80.9	75.9	72.7	73.1	72.9	77.6
BigBird (L=1024)	82.2	85.5	83.8	72.8	78.4	75.5	72.7	72.5	72.6	77.3
BigBird (L=4096)	81.8	87.0	84.3	71.5	81.0	76.0	72.7	73.2	73.0	77.7
Longformer (L=128)	81.7	83.7	82.7	71.6	75.8	73.7	71.3	70.4	70.9	75.7
Longformer^G (L=128)	81.1	84.3	82.7	70.6	76.5	73.4	71.0	70.9	70.9	75.7
Longformer (L=256)	81.6	85.2	83.4	71.8	78.1	74.8	71.6	72.3	72.0	76.7
Longformer^G (L=256)	81.4	84.8	83.1	71.4	77.6	74.4	71.0	71.7	71.3	76.3
Longformer (L=512)	82.4	85.3	83.8	73.0	78.6	75.7	72.6	72.5	72.5	77.4
Longformer^G (L=512)	82.6	85.0	83.8	73.2	77.9	75.5	72.4	72.4	72.4	77.2
Longformer (L=1024)	82.1	84.9	83.5	72.3	77.7	74.9	72.0	72.0	72.0	76.8
Longformer (L=4096)	82.0	84.2	83.1	72.4	76.2	74.3	71.6	71.6	71.6	76.3
RoBERTa (L=128)	81.4	82.5	81.9	70.7	73.9	72.2	71.4	68.2	69.7	74.6
RoBERTa^p (L=128)	69.4	57.7	63.0	55.9	42.5	48.3	49.9	38.4	43.4	51.6
RoBERTa (L=256)	82.0	84.5	83.2	72.2	77.2	74.6	72.3	70.7	71.5	76.5
RoBERTa^p (L=256)	68.7	57.9	62.9	55.7	43.1	48.6	49.3	39.2	43.7	51.7
RoBERTa (L=512)	81.6	85.0	83.3	72.1	78.2	75.1	72.1	72.2	72.2	76.8
RoBERTa^p (L=512)	68.0	56.1	61.5	55.1	40.8	46.9	48.4	38.4	42.8	50.4
SpanBERT (L=128)	82.0	82.2	82.1	72.0	73.7	72.8	71.5	69.0	70.2	75.0
SpanBERT^p (L=128)	70.6	56.8	63.0	58.1	42.9	49.4	50.8	40.5	45.1	52.5
SpanBERT (L=256)	82.7	82.8	82.7	73.0	74.1	73.5	71.9	70.6	71.3	75.8
SpanBERT^p (L=256)	70.0	56.4	62.5	58.5	41.7	48.7	50.1	40.8	45.0	52.1
SpanBERT (L=512)	83.3	82.9	83.1	74.4	74.8	74.6	72.6	71.7	72.1	76.6
SpanBERT^p (L=512)	67.6	55.7	61.1	56.2	40.8	47.3	47.2	39.8	43.2	50.5
XLNet (L=128, m=0)	81.6	82.7	82.1	71.2	73.4	72.3	70.0	68.6	69.3	74.6
XLNet (L=128, m=128)	81.7	82.6	82.1	71.9	73.3	72.6	69.9	69.0	69.5	74.7
XLNet (L=256, m=0)	79.4	84.2	81.7	68.5	76.0	72.0	68.5	70.9	69.7	74.5
XLNet (L=256, m=256)	84.3	81.8	83.0	75.4	72.2	73.8	72.2	68.9	70.5	75.8
XLNet (L=512, m=0)	79.0	85.2	82.0	67.4	77.4	72.1	69.1	68.8	69.0	74.3
XLNet (L=512, m=512)	82.1	84.1	83.1	72.5	76.1	74.3	72.8	70.3	71.5	76.3

Table 8: Full results on Ontonotes with the coarse2fine model. L is the segment length used to chunk the text. m is the memory length used for the XLNet model. ^{G} denotes that the global tokens are used. ^{p} denotes that th self-attention computation is replaced with Performer kernels.

Encoder	L=128		L=256		L=512		L=1024		L=∞	
	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Longformer	54.26	50.02	58.83	54.48	63.88	59.13	63.91	58.91	63.41	58.89
Longformer^G	-	-	-	-	-	-	-	-	72.96	67.88
RoBERTa	55.81	50.73	60.29	56.11	63.45	58.84	-	-	-	-
RoBERTa^p	23.17	16.80	21.87	15.56	21.11	15.09	-	-	-	-
BigBird	55.28	50.66	59.39	54.34	63.51	58.50	66.50	61.15	71.78	66.86
XLNet	51.46	47.10	56.26	52.08	60.05	55.62	-	-	-	-
XLNet^m	52.71	48.03	57.96	52.93	62.85	58.13	-	-	-	-

Table 9: Full results on TriviaQA. We adopt the same notation as we used in table 8.

	Encoder	L=512			L=1024			L=1536			$L=\infty$		
		R1	R2	RL	R1	R2	RL	R1	R2	RL	R1	R2	RL
SF	BART	26.3	5.1	16.2	27.2	4.9	16.7	-	-	-	-	-	-
	LED	32.8	7.0	18.8	33.1	7.3	18.9	33.2	7.0	18.6	33.6	7.1	18.7
GR	BART	45.6	16.9	21.8	47.9	18.6	22.7	-	-	-	-	-	-
	LED	53.9	24.7	27.1	54.1	25.1	27.9	54.8	25.7	27.8	56.6	26.6	29.1

Table 10: Full results on summarization. “SS” stands for the SummScreen dataset, and “GR” stands for the GovReport dataset. The BART model does not support sequence longer than 1024 tokens.

Encoder	MUC			B^3			CEAF$_{\phi_4}$			Avg.
	P	R	F1	P	R	F1	P	R	F1	
BigBird (L=128)	84.5	78.5	81.4	75.2	68.6	71.7	73.5	63.2	68.0	73.7
BigBird (L=256)	85.1	80.3	82.6	76.7	71.2	73.8	74.7	66.4	70.3	75.6
BigBird (L=512)	85.7	82.3	84.0	77.9	73.9	75.9	75.7	69.4	72.4	77.4
BigBird (L=1024)	85.2	82.5	83.8	77.1	74.5	75.8	76.3	69.6	72.8	77.4
BigBird (L=4096)	85.1	82.8	83.9	77.7	75.1	76.4	75.6	70.1	72.7	77.7
Longformer (L=128)	84.4	80.0	82.1	75.3	70.5	72.8	72.9	66.0	69.3	74.8
Longformer^G (L=128)	84.4	79.1	81.7	75.1	69.2	72.0	72.8	65.2	68.8	74.2
Longformer (L=256)	84.8	81.7	83.2	76.2	72.5	74.3	73.9	68.8	71.2	76.3
Longformer^G (L=256)	84.4	81.8	83.1	75.6	73.1	74.3	74.3	68.3	71.2	76.2
Longformer (L=512)	85.5	82.6	84.0	78.0	74.5	76.2	75.2	70.7	72.9	77.7
Longformer^G (L=512)	84.5	83.4	83.9	76.2	75.2	75.7	74.3	70.9	72.6	77.4
Longformer (L=1024)	86.0	82.1	84.0	78.7	73.4	76.0	75.2	70.2	72.6	77.5
Longformer^G (L=1024)	82.4	79.2	80.8	72.2	69.5	70.8	72.2	65.4	68.6	73.4
Longformer (L=4096)	85.2	82.9	84.1	77.4	74.6	76.0	74.8	70.7	72.7	77.6
RoBERTa (L=128)	81.1	78.0	79.6	70.5	68.0	69.3	71.2	63.3	67.0	72.0
RoBERTa^p (L=128)	61.3	45.0	51.9	45.9	30.3	36.5	41.4	25.8	31.8	40.1
RoBERTa (L=256)	84.7	81.8	83.2	76.0	72.6	74.3	74.2	68.5	71.3	76.3
RoBERTa^p (L=256)	67.7	46.0	54.8	53.0	30.8	39.0	43.8	26.9	33.3	42.4
RoBERTa (L=512)	85.7	82.6	84.1	78.1	74.3	76.2	75.2	70.9	73.0	77.8
RoBERTa^p (L=512)	67.0	45.0	53.9	53.0	29.8	38.1	43.2	26.8	33.1	41.7
SpanBERT (L=128)	78.2	75.5	76.8	66.6	64.3	65.5	68.2	60.5	64.1	68.7
SpanBERT^p (L=128)	56.5	45.7	50.5	39.8	31.5	35.1	38.4	25.1	30.4	38.7
SpanBERT (L=256)	83.2	79.6	81.4	74.4	70.1	72.2	71.7	67.0	69.3	74.3
SpanBERT^p (L=256)	64.1	46.5	53.9	49.8	31.8	38.8	40.7	27.9	33.1	41.9
SpanBERT (L=512)	83.5	81.3	82.4	74.6	71.9	73.2	72.3	68.5	70.3	75.3
SpanBERT^p (L=512)	63.7	47.0	54.1	48.7	32.0	38.6	42.2	27.9	33.6	42.1
XLNet (L=128, m=0)	79.4	39.8	53.0	69.1	30.1	41.9	61.6	32.7	42.7	45.9
XLNet (L=128, m=128)	78.1	49.1	60.3	66.0	39.4	49.3	63.8	38.8	48.2	52.6
XLNet (L=256, m=0)	78.8	59.5	67.8	68.0	48.6	56.7	66.4	47.9	55.7	60.1
XLNet (L=256, m=256)	64.6	67.5	66.0	48.4	55.0	51.5	59.8	45.3	51.6	56.4
XLNet (L=512, m=0)	80.3	71.7	75.7	70.7	61.4	65.7	66.9	60.0	63.3	68.2
XLNet (L=512, m=512)	76.2	73.0	74.6	64.2	63.8	64.0	66.4	58.3	62.1	66.9

Table 11: Full results on Ontonotes with the start2end model. We adopt the same notation as we used in table 8.

Creation and evaluation of timelines for longitudinal user posts

Anthony Hills[1], **Adam Tsakalidis**[1,2], **Federico Nanni**[2], **Ioannis Zachos**[3], **Maria Liakata**[1,2,4]
[1]Queen Mary University of London, [2]The Alan Turing Institute,
[3]University of Cambridge, [4]University of Warwick
`{a.r.hills;a.tsakalidis;m.liakata}@qmul.ac.uk`

Abstract

There is increasing interest to work with user generated content in social media, especially textual posts over time. Currently there is no consistent way of segmenting user posts into timelines in a meaningful way that improves the quality and cost of manual annotation. Here we propose a set of methods for segmenting longitudinal user posts into timelines likely to contain interesting moments of change in a user's behaviour, based on their online posting activity. We also propose a novel framework for evaluating timelines and show its applicability in the context of two different social media datasets. Finally, we present a discussion of the linguistic content of highly ranked timelines. [1]

1 Introduction

An increasing body of work considers time-aware models trained on social media data for a number of different tasks, including personal event identification (Li and Cardie, 2014; Li et al., 2014; Chang et al., 2016a), suicidal ideation and suicide risk detection (Coppersmith et al., 2014, 2018; Cao et al., 2019; Matero et al., 2019; Sawhney et al., 2020, 2021). For such tasks deriving meaningful *timelines* (i.e. sequences of posts by individuals), containing examples of the phenomenon under study from large-scale collections, together with associated annotations, is crucial. This is especially important for computational approaches in mental health (MH) given the surging numbers of those seeking help online (Neary and Schueller, 2018).

Earlier work on personal life event detection considered selecting salient timelines through topic modelling (Li and Cardie, 2014; Li et al., 2014) or through a non-parametric generative approach (Chang et al., 2016a). However, such approaches are unsuitable for identifying changes in mood or MH more generally. Specifically, since timelines are selected based on linguistic content this introduces a sampling bias for downstream linguistic analysis and annotation (Olteanu et al., 2019; Mishra et al., 2019). In recent work on suicidal ideation detection, timelines are chosen as the N most recent posts (Sawhney et al., 2020), which are not necessarily the most salient for annotation.

Present Work: We propose a set of methods and associated evaluation framework for identifying salient timelines from the history of social media users to be annotated for changes in a user's behaviour, as revealed through their textual data. Applying our methods in the domain of MH, we follow earlier work in hypothesising that posting behaviour can be a proxy for changes in the MH of an individual (De Choudhury et al., 2016). Therefore we develop methods for creating timelines based on time-series of posting frequency, such as change-point and anomaly detection approaches, and evaluate these against keyword-based methods and randomly selected timelines, in the context of the task of capturing *Moments of Change (MoC)*. A MoC is a particular point or set of points in time denoting: (1) a shift in an individual's mood from positive-to-negative or vice versa; or (2) a gradual mood progression (Tsakalidis et al., 2022a). We show that our proposed timeline segmentation methods can consistently select timelines that are rich in MoC for large scale cost-effective annotation. We make the following contributions:

- We present approaches for extracting timelines from users' posting history on social media based on change-point detection and anomaly detection methods (§3).
- We propose a novel evaluation framework for assessing the quality of annotated timelines, and timeline selection methods, which we evaluate on the task of capturing MoCs (§4.2) on two different social media datasets.
- We provide a linguistic analysis of timelines ob-

---

[1] `https://github.com/Maria-Liakata-NLP-Group/timeline_selection_and_evaluation`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3791–3804
May 2-6, 2023 ©2023 Association for Computational Linguistics

tained, distinguishing timelines dense in MoCs, from timelines sparse in MoCs (see §5.2).

2 Related Work

Since we aim to segment users' entire posting history into smaller sequences, manageable to annotate and salient in terms of containing moments of change in mental health, we consider work in the following areas: mental health monitoring (2.1); text segmentation (2.2); timeline summarization (2.3); change-point detection (2.4).

2.1 Tracking Changes in Mental Health (MH)

Moments of Change (MoC) are important in MH tracking. Pruksachatkun et al. (2019) identifies a MoC as a positive change in sentiment for a user with respect to a distressing topic mentioned in a conversation thread. De Choudhury et al. (2016) investigated shifts to suicide ideation with models predicting when users transition to posting on a suicide support forum. We consider a more general definition of MoC (§1, "Present Work").

Creation of Mental Health Datasets. A large body of work in creating MH datasets involves labelling posts for symptoms (Gkotsis et al., 2017; Loveys et al., 2017; Cheng et al., 2017) or levels of suicide ideation (Masuda et al., 2013; Coppersmith et al., 2016; Shing et al., 2018). While annotations for some of these datasets are obtained through proxy signals (e.g., self-disclosure of diagnoses, posts on support networks) questions arise as to how to select appropriate data for annotation. Mishra et al. (2019) use keyword based methods to identify posts exhibiting the phenomenon under study (e.g. suicidal ideation) but this leads to sampling biases.

2.2 Text Segmentation (TS)

TS (Beeferman et al., 1999; Pak and Teh, 2018) focuses on splitting a large body of text (document) into smaller chunks (segments or "regions of interest" (Oyedotun and Khashman, 2016)). TS has been applied in numerous fields, including emotion (Wu et al., 2007) and sentiment detection (Chiru and Hadgu, 2013), often involving segmenting news articles (Gao et al., 2010) and review items (Sun et al., 2013). While there is some work in segmenting large bodies of social media posts into text segments (Kaur and Singh, 2019), we are not aware of work segmenting entire posting histories into smaller, more manageable segments (i.e. timelines), to improve downstream longitudinal annotation.

Furthermore, TS primarily operates on linguistic content, rather than timestamped information, with algorithms designed to identify segments containing certain topics of interest, resulting in selection bias (Riedl and Biemann, 2012; Takanobu et al., 2018; Hananto et al., 2022). An alternative is to consider timeline extraction approaches agnostic to the linguistic content, inspired by Timeline Summarisation and Change-Point Detection (CPD).

Evaluation metrics other than precision and recall have been proposed to account for near misses during text segmentation. P_k (Beeferman et al., 1999) uses a k-sized sliding window on a document to compare predicted *vs* ground-truth segmentation locations, assigning partial credit to near misses. However, it is affected by variations in segment sizes and penalizes false negatives more than false positives. WindowDiff (Pevzner and Hearst, 2002) penalises the latter equally. Both metrics require ground-truth annotations of the optimal segmentation locations. We propose an approach (§4) to evaluate segmentation of users' histories based on the proportion of desired annotation labels within a set of sampled sequences of posts (timelines).

2.3 Timeline Summarization (TLS)

TLS aims to provide concise chronologically ordered timelines consisting only of the most relevant information for a given topic or entity, summarizing the key points in time. While TLS has been most commonly applied in news topic summarization (Swan and Allan, 2000; Martschat and Markert, 2017, 2018; Steen and Markert, 2019), there has been increasing interest in applying TLS to social media data (Li and Cardie, 2014; Chen et al., 2019; Ansah et al., 2019; Wang et al., 2021).

TLS consists of a 2-step pipeline: (1) date selection, then (2) summarisation. Salient dates summarizing a timeline are typically identified using textual content, as well as time-series information in the history of an individual/topic. Focusing on viral buzzes of celebrity mentions on social media, Chang et al. (2016b,a) aims to select dates by modelling linguistic content and frequency-based time-series patterns.

2.4 Change-point Detection (CPD)

While CPD has been explored to some extent in news TLS (Hu et al., 2011), it remains underexplored for social media data. **Change-points**

(CPs) are defined as points in time where the underlying generative parameters of a data sequence are predicted to have changed (van den Burg and Williams, 2020). CPD therefore often involves learning a predictive model of a data sequence. In §3, we use automatically detected CPs to identify salient dates for selecting timelines of users on social media for annotation. While several continuous models exist (e.g. Gaussian (Adams and MacKay, 2007)), we focus on models suited to discrete time-stamped data (Knoblauch and Damoulas, 2018) – such as when posts/comments are made on social media. In such scenarios Temporal Point Processes (TPPs) (Daley and Vere-Jones, 2003) are well suited.

Temporal Point Processes (TPPs) TPPs are stochastic processes that model discrete events localized in continuous time. They are typically characterized by an intensity function, $\lambda{>}0$, which represents the instantaneous rate of event occurrence.

In order to use TPPs to model event sequences, and predict associated changes – certain CPD models, such as Bayesian Online Change-point Detection (Adams and MacKay, 2007) require the TPP to be part of the exponential family of distributions (e.g. Poisson). This is so that the intensity λ can be further modelled from a prior conjugate distribution, making it possible to construct the likelihood of the chosen predictive model in a closed form.

3 Approach for Selecting Timelines

Task. Our principal aim is to select timelines for annotation that are rich in changes in posting behaviour on a MH platform, which we consider as a proxy for changes in MH – in particular, Moments of Change (MoC). To achieve this, we test a series of timeline selection methods (§3.1-§3.2), which we evaluate using our proposed framework (§4).

Selecting Candidate Timelines. To select timelines for annotation, we extract candidate timelines as a span of timestamps S from a user's u history H. We first propose identifying changes in posting behaviour as *Candidate Moments of Change* (CMoC), which are dates hypothesised to be surrounded by many MoCs (§3.1). Subsequently, we extract the user's posts surrounding these CMoC within a fixed time window, as timelines to be returned for annotation (§3.2).

3.1 Identifying Candidate MoCs (CMoC)

We investigate the following for identifying CMoC:

(1) Bayesian Online Change-point Detection (BOCPD): In a recent evaluation involving experiments with synthetic and real-world change-points, van den Burg and Williams (2020) showed that BOCPD was the best model for a variety of CPD tasks. BOCPD learns a predictive model on a data sequence. When changes in the model's generative parameters are identified, CPs are declared. BOCPD is typically fit with continuous models (e.g. the Gaussian distribution). However, in our case we consider models for discrete event-based data (Knoblauch and Damoulas, 2018).

Since we hypothesize that changes in posting behaviour coincide with changes in mood (see "Present Work" in §1), we use BOCPD to identify changes in individuals' posting frequency. As such we consider the daily frequency of posts made by a user as a TPP, and use the homogeneous Poisson-Gamma (PG) point process model with BOCPD (Knoblauch and Damoulas, 2018) to fit and identify changes in the daily frequency of posts by a user from their entire associated history. We assess our hypothesis by evaluating timelines obtained this way in terms of how dense they are in MoCs, changes in mood and sentiment (Table 3).

By using a PG model with BOCPD, we assume that each point in a user's posting frequency is sampled from a Poisson distribution with a discrete λ. Here λ represents the expected number of posts by a user within a given time interval. As we use this conjugate Bayesian model, λ is further assumed to be drawn from a Gamma distribution with a set of priors α_0 and β_0, that act as initial hyper-parameters in our model, where α_0/β_0, α_0/β_0^2 denote the prior mean and variance over λ. BOCPD has an additional hyper-parameter which is the hazard h_0, where $1/h_0$ expresses a prior belief about the probability of CPs occurring at a given time t, provided that a CP has not recently occurred: a low h_0 results in the over-generation of change-points while a large h_0 is more conservative and returns very few CPs (ideal in our scenario, to ensure that we do not waste annotation resources, by avoiding annotating too many timelines generated by noise). As such, we experiment with two settings of BOCPD to identify CMoCs: BOCPD (1) and BOCPD (2), which have priors (α_0:.01; β_0:10; h_0:10^3) and (α_0:1; β_0:1; h_0:10) respectively.

Since BOCPD computes a full probability distribution over the location of the CPs, quantifying probable CPs along with their associated uncertainty, we use the maximum a posteriori (MAP) segmentation of the probability distribution to return exact point estimates for CPs (Fearnhead and Liu, 2007; van den Burg and Williams, 2020), which in our setting define CMoCs. An illustration of identifying CMoCs from a given user's history in our implementation of BOCPD is provided in Fig. 1.

(2) Anomaly Detection (AD): Here we aim at identifying (a) days of abnormally high user activity and (b) abnormally long time periods of no user activity at all. We hypothesize that such points in time can be used to select salient timelines. We experiment using different features to fit our model, including the daily frequency of a user's posts and the number of comments they receive for those corresponding posts by others. Using either activity type, we scan over the user's entire history.

For (a) we explore the use of *Kernel Density Estimation (KDE)* (Rosenblatt, 1956; Scott, 2015) to estimate the probability density function of the user's activity. For (b), we focus on time periods in the user's history lasting at least 14 days during which the user had no activity (posts/comments) at all. Given the past 90 days of a user's activity, if the probability on a particular day of seeing either (a) a high volume of activity or (b) a long period of 'silence' is lower than .01, then we mark the start of this period as an 'anomaly' – i.e., CMoC. We explore (a) and (b) separately for posts and comments, and we also explore concatenating CMoCs identified for high and low posting activity for either comments received or posts made.

(3) Keywords: We incorporate a baseline for identifying CMoCs based on a set of keywords in the *suicide risk severity lexicon* (Gaur et al., 2019). Each keyword present in the lexicon corresponds to different levels of suicide risk severity such as "I'm tired of this suffering", and "I'm going to kill myself". We hypothesize that the presence of such phrases in a user's post may be indicative of a MoC. This method returns CMoCs for timestamps of posts by a given user that contain a keyword within the lexicon. Note that keyword methods are prone to sampling bias for downstream linguistic analysis, we include them in our experiments due to their popularity for comparison purposes.

(4) Random & Every day: We incorporate two naïve baselines, as such methods are important for benchmarking in MH tasks (Tsakalidis et al., 2018). "*Random single day*" selects a single date from a uniform distribution over all days in a user's posting history H as a CMoC, C (we evaluate against 100 random seeds to report average scores, §4). "*Every day*" returns every day as a CMoC – we employ it to see how well our methods are at avoiding the over-generation of candidate timelines. We seek to avoid over-generating timelines as we want to only return timelines with a high density of MoC to improve annotation efficiency.

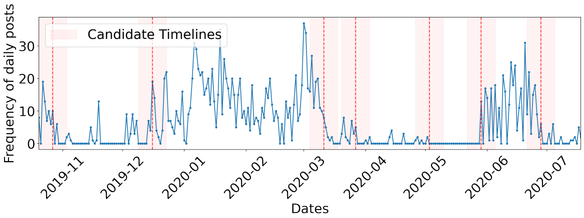


Figure 1: Using change-points in an example user's posting behaviour to define candidate moments of change $M_u^{(c)}$ (dashed red line). Candidate timelines are then created centred on each $M_u^{(c)}$, with a radius r=7.

3.2 Extracting Posts

Once a CMoC, C, is found, a span of timestamps S from the user's history H is identified within a radius r[2] around C. A candidate timeline then consists of the associated sequence of posts, corresponding timestamps and comments within S.

4 Evaluation of Selected Timelines

While there is previous work in evaluating segments of posts in text segmentation (§2.2) and timeline summarization (§2.3), there is little to no prior work on frameworks for evaluating timeline selection methods for the purposes of efficiently annotating longitudinal datasets. As such we identify this as a nascent area of study – ripe for others to build upon, and propose a novel evaluation framework for selecting timelines for this task.

We investigate several metrics for evaluating the methods from §3 in terms of their ability to select timelines that correspond to a high proportion of Ground-truth Moments of Change (GTMoC), denoted hitherto as G. Each CMoC generated by a method as a change point is denoted hitherto as C. Since we do not have access to manual ground truth annotations outside of the span of our annotated timelines, we can only evaluate methods according to CMoCs that fall within them.

---

[2] Here we take $r = 7$ which gives a manageable amount of posts while providing context before and after the CMoC.

4.1 Time-varying Classification Metrics

We use the precision and recall metrics by van den Burg and Williams (2020) for evaluating change-points (CPs) – i.e., CPs are evaluated based on the distance d_{GTMoC} of the predicted CP C falling within a margin of error distance τ to Ground-truth Moments of Change G. For our scenario, τ is reflective of the length of the timelines to be created, and is roughly the radius of a timeline. It should also be chosen based on the uncertainty of the annotation labels. The pros of making assessments based on high performance with a small τ, is that this suggests that very narrow timelines can be created, while still capturing the annotation labels. This allows many timelines to be annotated, thus increasing the diversity of the dataset. However, if timelines are too small, there may not be enough context provided to annotators to perform the annotation task. Thus, allowing for larger timelines provides more context to annotators, which can potentially improve the quality of annotations – but increase the cost and time to perform the annotation. In our experiments we make assessments based on moderately sized τ to allow for moderately sized timelines. We use τ = 5 days in table 2, which is the same value used in the experiments of (van den Burg and Williams, 2020).

A true positive (TP) therefore corresponds to an intersection of a G with a C: $G \cap C$, if $|G - C| \leq \tau$. We ensure there is a 1:1 mapping between each G and C – where each C can only intersect as TP against a single G. The total number of TPs for a timeline therefore is given by $\max(|G \cap C|) \leq \max(|G|, |C|)$, where G and C are sets of dates in annotated timelines. The precision and recall are thus defined as $P = \frac{|G \cap C|}{|C|}$ and $R = \frac{|G \cap C|}{|G|}$, respectively. We compute P and R for each annotated timeline and report mean across all timelines. The mean scores are then used to compute the mean F1.

While these metrics evaluate how well a timeline selection method can identify CMoCs close to GTMoCs, they cannot tell us which method is able to return timelines that contain a high proportion of GTMoCs relative to the number of posts (timelines with high density of GTMoCs). Thus we propose an alternative metric (Medoid Votes) based on densities of GTMoCs, as discussed next.

4.2 Medoid Votes (MV)

We propose a new metric, MV, to account for the inability of prior metrics to consider the density of labels within timelines. Although a method may have high precision (yielding a prediction close to a ground truth label), the timelines overall may contain a low proportion of the labels that we seek to annotate – leading to inefficient annotation. Hence, we introduce MV which assigns true positives against *dense regions* of labels as opposed to single labels. As we demonstrate in our experiments, assessments made using MV are more robust, resulting in timelines centered around highly dense regions of the labels we seek to annotate.

To make assessments using MV, first we identify periods in manually pre-annotated user timelines that contain a high proportion of GTMoCs relative to the number of posts within the timelines (dense regions) (§4.2.1). We then assign votes to methods that identify CMoCs close to these, and obtain a ranking (§4.2.2).

4.2.1 Dense Regions in Annotated Timelines

Medoids. We use the notion of 'medoids' to represent the location of dense regions of GTMoCs. A *medoid* M is the timestamp of the GTMoC in a given timeline T, from which the (Euclidean) distances $d(.,.)$ of all other timestamps of annotated GTMoCs G in timeline T are minimal:

$$M = \underset{G_a \in T}{\arg\min} \sum_{G_b \in T} d(G_a, G_b) \qquad (1)$$

Density of annotated timelines. We further characterise the locations of dense regions (medoids) by the number of GTMoC they contain. This *'density'* of a timeline is defined as $\rho = \frac{|G|}{|p|}$, where $|G|$ is the sum total number of GTMoCs within an annotated timeline T and $|p|$ is the number of posts in T.

In order to weight timelines by how dense they are in GTMoCs, a medoid M inherits the density ρ of the timeline T it represents. We transform ρ_T for each T, to provide a binary distinction between "dense" (+1) and "sparse" (-1) medoids as:

$$\rho_T^{\text{(binary)}} \begin{cases} +1 & \text{if } \rho_T \geq \text{Median}(\rho_T \ \forall \ T) \\ -1 & \text{otherwise} \end{cases}$$

A good timeline is therefore one that is "dense", and the ideal location for a CMoC is as close as possible to a dense medoid M (see eq. 1).

In an ideal scenario where we have the resources to annotate many timelines sampled from many candidate methods, we could compare and rank the methods based on the number of dense timelines or the average resulting densities. Alternatively, we could evaluate the proposed methods against a set of fully-annotated user histories. However, due to the high cost and time-consuming process of annotation, such approaches are infeasible. Instead we propose an alternative solution that does not require annotating all the timelines that would be generated (or entire user histories). We do this via a scoring system based on distances of CMoC relative to dense medoids in a small set of trial annotated timelines, as described next.

4.2.2 Scoring Timeline Selection Methods

We employ the evaluation framework in §4.2.1 to assess pre-annotated timelines against CMoCs in timelines selected by different methods. Assuming an annotated timeline T, we aim to assess how close an identified CMoC C is to a dense region of GTMoCs within T. We therefore give preference to methods that identify CMoCs in close proximity to medoids that are dense in GTMoC, while also penalizing methods that over-generate CMoC.

Distance Scores. We calculate the proximity of CMoCs predicted by a method to M as the minimum absolute distance d_m (in days) between all CMoCs predicted by a given method m (§3.1) for a user's entire history. Then, we compute a distance score for each m per annotated timeline as:

$$D_m = (d_m + \epsilon) * \text{sign}(\rho_T^{(\text{binary})}),$$

where ϵ=.001, to preserve the sign of each medoid's $\rho_T^{(\text{binary})}$ in the case of d_m= 0. D_m is then used to denote the proximity of CMoCs predicted by method m (in days) to a ground truth medoid M with density $\rho_T^{(\text{binary})}$. Since we want to obtain timelines that are close to dense regions in GTMoC, we seek to identify methods with low positive D_m.

Votes. To reward methods that identify a CMoC in close proximity to a 'dense' M (low positive D_m), and penalize methods which over-generate CMoC (e.g., in locations that contain a low density of GTMoC), we assign votes to each method m by:

$$v_m = \begin{cases} +1 & \text{if } 0 \leq D_m \leq \tau \\ 0 & \text{otherwise} \end{cases}$$

where τ is the same margin of error (in days) described in §4.1. This gives a positive vote to a method generating a CMoC that falls within a margin of τ days to a dense medoid. Votes v are then normalized per timeline and method ($V_m = \frac{v_m}{|C|}$, where $|C|$ is the total number of CMoCs generated by m, that fall within each annotated timeline).

Scoring of methods. Timeline selection methods are subsequently scored and ranked by summing the votes V_m for each method m over all T. As we are concerned with ranking methods, we then min-max scale our results in the range of 0 to +1, where methods that have scores close to 1 rank near the top and methods that score close to 0 are the worst in their ability to return timelines containing a high proportion of GTMoCs. The scoring of methods proposed in §3.2 are shown in Table 2, and Fig. 4 for varying margin of error, τ. The evaluation framework is visualised in Fig. 2.

5 Experiments

We evaluate our timeline selection methods (§3), using our evaluation framework (§4) based on ground-truth human annotated data.

5.1 Datasets

We evaluate our automatic timeline selection methods using two datasets (summarised in Table 1) from different platforms: The *TalkLife* dataset contains timelines automatically selected using one of our proposed methods. While our evaluation is designed to allow alternative methods to achieve higher scores than the methods used to select timelines we still want to exclude any possibility of inherent bias. To this effect we also evaluate against timelines manually selected from *Reddit* independently from this work (Tsakalidis et al., 2022b).

TalkLife[3] is a peer-support social network operating primarily as a mobile app. Users are mainly English speakers, 70% of whom are 15-24 years old (Sharma et al., 2020a). The posts/comments on TalkLife focus primarily on MH, daily-life issues and feelings. It is thus suited to identifying MoC and computationally analysing MH (Pruksachatkun et al., 2019; Sharma et al., 2020b; Saha and Sharma, 2020; Kim et al., 2021). We select timelines on the basis of timestamped user posting frequency, and associated comments received. The context of posts is only used in annotating the selected timelines; thus, methods for timeline selection are transferable to other platforms.

We licensed a de-identified dataset from TalkLife consisting of 1.1M users (12.3M posts, Aug'11-Aug'20). Due to the high variance in users' posting frequency, only timelines having [10-150] posts

---

[3] `https://www.talklife.com`

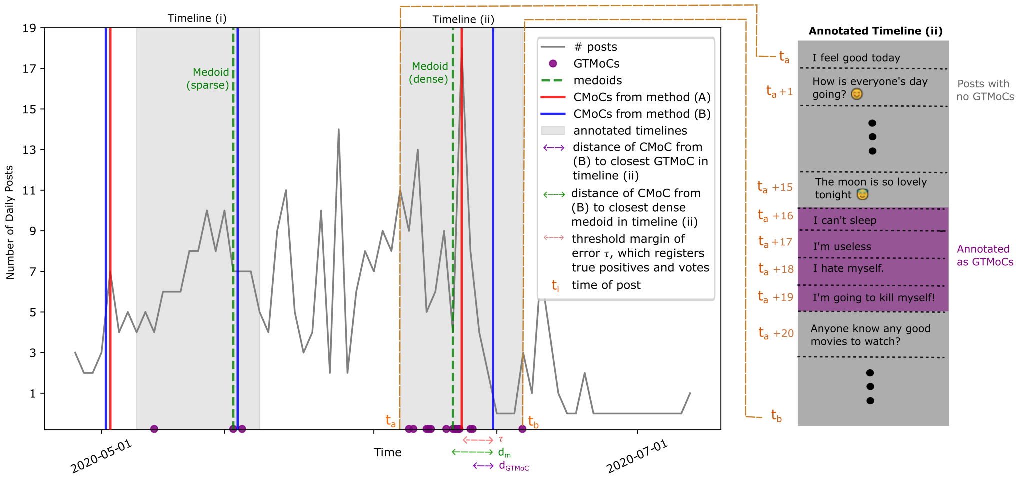


Figure 2: Evaluation of CMoCs against GTMoCs. Votes and true positives are assigned based on distances d of CMoCs falling within a margin of error τ against dense medoids or GTMoCs. Here, method A (red) selects better timelines than method B (blue), as these are close to dense regions of GTMoCs ($d_{\mathrm{m}} \leq \tau$) and labels ($d_{\mathrm{GTMoC}} \leq \tau$).

were considered for annotation. This was so that timelines were not impractically long while still providing enough context for annotators to observe and mark a change. The final annotated dataset consists of 500 timelines (see Table 1), with a mean of 35 posts (±22). These timelines were selected using BOCPD PG (1), where the parameters (α_0:.01; β_0:10; h_0:10^3) were fixed on the basis of improved model performance on a validation dataset of 70 manually annotated timelines selected via anomaly detection. All 500 timelines within the evaluation dataset were manually inspected and filtered according to the details in A.1.

Reddit. We further tested the generalizability of our methods and evaluation framework on a different dataset, that was not generated using automatic timeline selection approaches – the CLPsych 2022 Shared Task corpus (Tsakalidis et al., 2022b). We chose to include this additional dataset to address potential concerns that experiments and analysis performed on the TalkLife timelines have some bias towards the BOCPD method in experiments evaluated on the TalkLife timelines – as they were selected using BOCPD. This corpus was sourced from Reddit, a social media platform where individuals make public posts and which has been studied extensively as a resource for mining textual data for MH studies (De Choudhury and De, 2014; Losada and Crestani, 2016; Shing et al., 2018; Zirikly et al., 2019; Losada et al., 2020; Low et al., 2020). We make use of the 'Reddit-New' dataset of the CLPsych 2022 corpus, consisting of 139 timelines where 17-82% of posts come from MH subreddits and had been pre-selected manually by two researchers independently as likely to contain a high proportion of MoCs.

Annotation of GTMoC in TalkLife timelines was performed by 3 English speaking (1 native), university educated annotators. Reddit timelines were annotated by 4 English (2 native) speakers (Tsakalidis et al., 2022b).

Annotators were provided with timelines containing chronological posts by users with their associated comments and timestamps. They were asked to label posts containing a 'Switch' (sudden change in mood) or an 'Escalation' (gradual mood progression) – a (default) label of 'None' was assigned to posts with no MoC. A 'Switch' is defined in the guidelines as 'a drastic change in mood, in comparison with the recent past', with annotators having to label its beginning and its range. An 'Escalation' is 'a gradual change in mood, which should last for a few posts'. Annotators had to label the peak of an escalation and the range of associated posts (see Fig. 9 of A.2 as an example).

To obtain GTMoC for our evaluation we aggregate the annotations across all annotators per timeline in the same way as (Tsakalidis et al., 2022a). Due to the challenging and subjective nature of the annotation task, the percent of inter-annotator agreement for the labels 'None', 'Switch' and 'Escalation' were .89, .30, and .50 respectively for the TalkLife dataset, and .83, .26, and .31 respectively for the 2022 CLPsych Corpus, based on majority agreement. We consider all labels of 'Switch', 'Escalation', and their corresponding ranges as GTMoC. We thus merge both labels to define GTMoCs, as we are interested in identifying timelines that contain both types of changes in mood.

	Timelines	Posts	Users	Timeline Length
TalkLife	500	18,702	500	$\leq$ 2 weeks
Reddit	139	3,089	83	$\sim$ 2 months

Table 1: Summary of datasets used in our experiments.

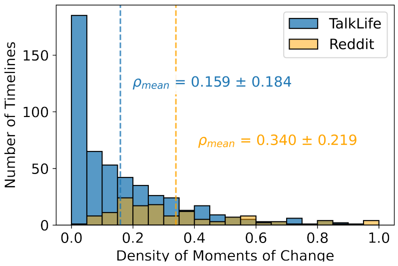


Figure 3: Density of GTMoCs per timeline.

5.2 Results & Discussion

We identify CMoCs (§3.1) on annotated timelines from TalkLife and Reddit (§5.1), and evaluate using our metrics (§4). We round CMoCs to the nearest day, de-duplicating dates, to compare methods.

Density scores of annotated timelines. The density of the annotated timelines from TalkLife are presented in Fig. 3. The mean density (.159) is comparatively high considering that GTMoCs are rare events, and many timelines do not contain any GTMoC. While the mean density (.340) of manually selected timelines from Reddit is higher, extra annotation effort was taken by annotators to ensure these timelines had a high proportion of GTMoCs.

Ranking of timeline selection methods. Table 2 and Fig. 4 shows the generalizability of our models and evaluation based on the consistency of results across both datasets. Overall, BOCPD models achieve the highest precision, and relatively high medoid votes (MV) across varying values of τ. Note that BOCPD PG (1) had hyper-parameters that were tuned for the data on TalkLife, whereas BOCPD PG (2) has very general hyper-parameters – not tuned for either TalkLife or Reddit. Despite not having any models tuned specifically for Reddit, BOCPD (1) achieves the highest precision for the majority of margins of error τ, and BOCPD (2) achieves the 2nd highest precision for larger τ. Importantly, BOCPD achieves the highest precision for most cases of τ across both datasets. Precision is particularly important as it ensures that the resulting CMoCs will have a high chance to be close to GTMoCs. This aligns with our objective of ensuring the resulting dataset will be annotated with a high proportion of GTMoCs.

For both Reddit, and TalkLife, the more general parameters of BOCPD PG (2), which were not tuned for either dataset, still achieve among the highest precision and MV (next highest MV – and also the highest P for TalkLife). Even with low h_0 and $\alpha_0/\beta_0 = 1$ (likelier to over-generate CMoCs), BOCPD (2) outperforms all AD and naïve methods on MV and F1 on TalkLife. For TalkLife, AD (high activity: posts) achieves slightly worse MV compared to keywords, but outperforms it on Reddit, despite being potentially disadvantaged by not using linguistic content. AD (low activity) achieve among the worst F1 and MV. As a result, timelines created around anomalously low post frequency would be unsuitable for selecting dense timelines.

Scores vary with τ (Fig. 4). For low margins (τ<3) BOCPD ranks lower in F1 and MV in both datasets, but ranks among the highest for larger τ. We attribute this to BOCPD assigning CMoCs to transitions from high to low posting activity. As we expand τ and select longer timelines around CMoCs, BOCPD is able to capture moments in time which can contain both high and low posting activity. Transitions from high to low posting activity may not be captured for low τ – potentially explaining why the performance in this case is lower than methods that favour a high amount of posts. Since timelines on TalkLife were created with a radius of 7 in (Tsakalidis et al., 2022a), setting a fairly large τ=5 is suitable for assessing which methods are able to select dense timelines, while also allowing us to identify shorter, denser, timelines from longer annotated timelines, as in the case of Reddit.

While recall and F1 are relatively low for BOCPD across both datasets, we argue that precision and MV are the most important metrics to focus on for our task. Considering that 'everyday' has a perfect recall of 1.00, and that annotating all posts in a users history would indeed return all the GTMoCs for a user – this is highly inefficient and infeasible, and goes against our original objective of *efficiently* annotating a user's posts. By instead focusing on methods with high precision and MV, rather than recall, we ensure that the resulting timelines are near a high proportion of the labels we aim to annotate. This allows annotators to consider fewer posts to capture the same amount of rare labels, which are costly to annotate.

Linguistic analysis of timelines. To gain insights into the characteristics of 'dense' vs 'sparse' timelines, we employ VADER (Hutto and Gilbert, 2014), assigning a sentiment score per post, and

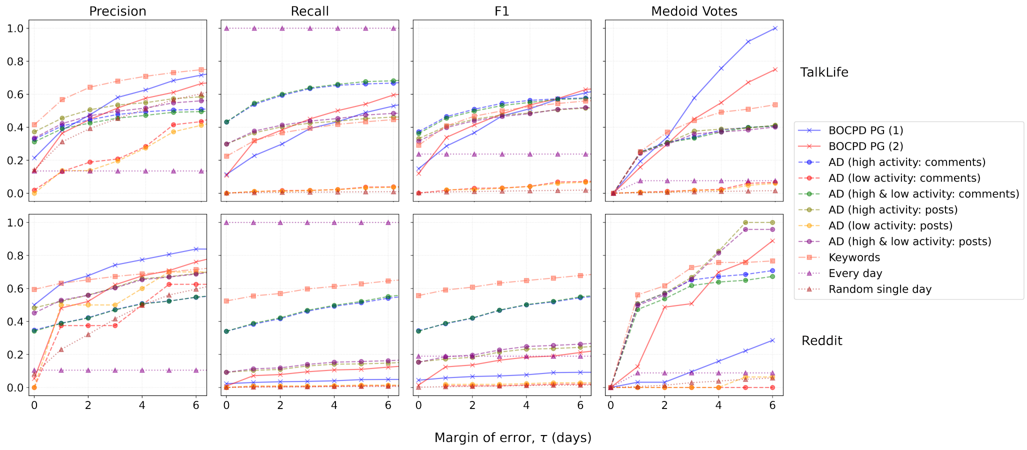


Figure 4: Evaluation metrics for different timeline selection methods, with varying margins of error τ (days).

	TalkLife				Reddit			
Method	P	R	$F1$	MV	P	R	$F1$	MV
BOCPD PG (1)	**.683**	.489	**.570**	**.919**	**.806**	.048	.090	.222
BOCPD PG (2)	**.611**	.540	**.574**	**.672**	**.708**	.110	.190	**.762**
AD (high comments)	.504	**.662**	**.573**	.399	.524	.513	**.519**	.685
AD (low comments)	.415	.037	.068	.060	.625	.010	.020	.000
AD (high & low comments)	.491	**.677**	.569	.399	.523	**.521**	**.522**	.650
AD (high posts)	.573	.453	.506	.395	.671	.143	.236	**1.00**
AD (low posts)	.372	.033	.060	.048	**.700**	.014	.028	.064
AD (high & low posts)	.548	.474	.508	.383	.669	.157	.255	**.958**
Keywords	**.731**	.433	.544	**.509**	.702	**.628**	**.663**	.758
Every day	.135	**1.00**	.237	.076	.105	**1.00**	.190	.088
Random single day	.567	.009	.017	.014	.560	.007	.014	.050

Table 2: Evaluation of timeline selection methods, using a margin of τ=5 days. MV (§4.2) are min-max scaled in the range τ=[0,6] days. **First**, **second**, and **third** highest scores are highlighted.

Twitter-RoBERTa-emotion (Barbieri et al., 2020), assigning four emotion scores (joy, anger, sadness, optimism) per post on the TalkLife dataset. We equally split 250 TalkLife timelines, between 'dense' (density $\rho_{u,i}$ is in upper-quartile of all timelines) and 'sparse' (bottom-quartile). The distribution of sentiment scores across these timelines are shown in Fig. 5. For each timeline we extract statistical features (avg, std, min, max) for each emotion/sentiment dimension of its posts, and the same features based on their difference across two consecutive posts in the timeline. Using these features, we train a Logistic Regression aiming at predicting 'dense' vs 'sparse' timelines and extract the coefficients with the highest/lowest values.

Sparse timelines frequently consist of positive posts in sentiment/mood (see Table 3). On the other hand, sadness- and variance-based features correlate the most with predicting a timeline containing many MoCs – a finding that was empirically confirmed via manual inspection of the most dense timelines. Developing methods that account for the variability in a user's mood/sentiment is a potential future direction in this regard.

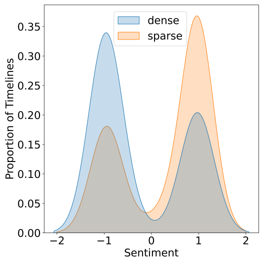


Figure 5: Sentiments of 'dense' *vs* 'sparse' timelines (medians: $-.949$ & .970, respectively).

Feature	Coef
sadness (avg)	2.29
sadness (std)	1.45
sentiment (std)	1.00
sentiment (avg)	-1.23
optimism (avg)	-1.25
sentiment (min)	-1.31
joy (avg)	-1.58

Table 3: Logistic Regression coefficients classifying timelines as 'dense' (1) or 'sparse' (-1).

6 Conclusions & Future work

We have introduced methods and an evaluation framework for identifying timelines from users' social media posts, likely to contain a large amount of Moments of Change (MoC). We use changes in posting behaviour as a proxy for changes in mood, to efficiently identify longitudinal user content worth annotating. Our methods have been manually evaluated against ground truth MoCs (GT-MoCs) in two different datasets. Bayesian Online Change Point Dection (BOCPD) shows promise in detecting timelines rich in GTMoCs.

Future work can explore the incorporation of textual content in the BOCPD Poisson-Gamma model for the distinction between different types of GT-MoC. We find that resulting timelines dense in GTMoCs are characterised by a high deviation in sentiment from one post to the next, suggesting that such deviations may be a useful feature for distinguishing between different types of GTMoC.

We expect that the methods proposed in our work will benefit researchers interested in creating longitudinally annotated textual datasets of user posts, particularly when annotating Moments of Change.

Ethics Statement

Ethics IRB approval was obtained from the Biomedical and Scientific Research Ethics Committee of the University of Warwick (ref: BSREC 40/19-20) prior to engaging in this research study. Our work involves ethical considerations around the analysis of user generated content shared on a peer support network (TalkLife). A license was obtained to work with the user data from TalkLife and a project proposal was submitted to them in order to embark on the project. The current paper focuses on the identification of periods of interest within the user history, in terms of moments of change. The work on annotation of moments of change (MoC) is separate to this paper but considers sudden shifts in mood (switches or escalations). Annotators were given contracts and paid fairly in line with University pay-scales. They were alerted about potentially encountering disturbing content and advised to take breaks during annotation. The annotations are used to evaluate the work of the current paper, which aims to meaningfully segment timelines in terms of containing likely moments of change. Potential risks from the application of our work in being able to identify moments of change in individuals' timelines are akin to the identification of those in earlier work on personal event identification from social media and the detection of suicidal ideation. Potential mitigation strategies include restricting access to the code base and annotation labels used for evaluation. No data can be shared without permission from the platform or significantly paraphrased. Any examples used from the users' history are anonymised and paraphrased.

Limitations

In this work we focus on returning timelines rich in Ground-truth Moments of Change (GTMoCs) in mood, using posts on social media which are by definition sparse. This has several limitations. Firstly, our labels of GTMoCs rely on individuals self-disclosing related information. We cannot make assessments based on someone's experience offline. The users chosen in our sample may also be users who are more likely to disclose information and so their posting patterns may not be typical of the general population. Both of these issues are true for most work in affective computing from social media.

Our methods for identifying Candidate Moments of Change (CMoCs) have several limitations. Similar to the issues with our GTMoCs, these methods rely on posting behaviour and cannot capture behaviour outside the user's social media history. Another limitation of our methods for identifying CMoCs is that they currently only use simple univariate features (e.g. posting frequency), and do not model the influence of cross-user interactions or multivariate features. While we suspect these methods for identifying CMoCs could be extended to model these more complex types of features and interactions, to better select timelines, we have not done this in the current work.

Finally, while we have shown that our methods for identifying CMoCs to select timelines rich in GTMoCs in mood generalize well between two social media platforms (TalkLife and Reddit), we have not experimented with other platforms.Our methods have been used for returning timelines rich in ground-truth labels for changes in mood but it remains to be seen whether they generalize well to identifying timelines rich in other labels for other related annotation tasks (e.g. labelling levels of suicide ideation). We believe this to be the case.

Acknowledgements

This work was supported by a UKRI/EPSRC Turing AI Fellowship to Maria Liakata (grant EP/V030302/1) and the Alan Turing Institute (grant EP/N510129/1). The authors would like to thank Julia Ive, Theo Damoulas, the anonymous reviewers and the meta-reviewers for their valuable feedback.

References

Ryan Prescott Adams and David J. C. MacKay. 2007. Bayesian Online Changepoint Detection. *arXiv:0710.3742 [stat]*. ArXiv: 0710.3742.

Jeffery Ansah, Lin Liu, Wei Kang, Selasie Kwashie, Jixue Li, and Jiuyong Li. 2019. A graph is worth a thousand words: Telling event stories using timeline summarization graphs. In *The World Wide Web Conference*, pages 2565–2571.

Francesco Barbieri, Jose Camacho-Collados, Leonardo Neves, and Luis Espinosa-Anke. 2020. Tweeteval: Unified benchmark and comparative evaluation for tweet classification. *arXiv preprint arXiv:2010.12421*.

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine learning*, 34(1):177–210.

Lei Cao, Huijun Zhang, Ling Feng, Zihan Wei, Xin Wang, Ningyun Li, and Xiaohao He. 2019. Latent suicide risk detection on microblog via suicide-oriented word embeddings and layered attention. *arXiv preprint arXiv:1910.12038*.

Yi Chang, Jiliang Tang, Dawei Yin, Makoto Yamada, and Yan Liu. 2016a. Timeline summarization from social media with life cycle models. In *IJCAI*, pages 3698–3704.

Yi Chang, Makoto Yamada, Antonio Ortega, and Yan Liu. 2016b. Lifecycle Modeling for Buzz Temporal Pattern Discovery. *ACM Transactions on Knowledge Discovery from Data*, 11(2):1–24.

Xiuying Chen, Zhangming Chan, Shen Gao, Meng-Hsuan Yu, Dongyan Zhao, and Rui Yan. 2019. Learning towards abstractive timeline summarization. In *IJCAI*, pages 4939–4945.

Qijin Cheng, Tim MH Li, Chi-Leung Kwok, Tingshao Zhu, and Paul SF Yip. 2017. Assessing suicide risk and emotional distress in chinese social media: a text mining and machine learning study. *Journal of medical internet research*, 19(7):e243.

Costin-Gabriel Chiru and Asmelash Teka Hadgu. 2013. Sentiment-based text segmentation. In *2nd International Conference on Systems and Computer Science*, pages 234–239. IEEE.

Glen Coppersmith, Mark Dredze, and Craig Harman. 2014. Quantifying mental health signals in twitter. In *Proceedings of the workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality*, pages 51–60.

Glen Coppersmith, Ryan Leary, Patrick Crutchley, and Alex Fine. 2018. Natural language processing of social media as screening for suicide risk. *Biomedical informatics insights*, 10:1178222618792860.

Glen Coppersmith, Kim Ngo, Ryan Leary, and Anthony Wood. 2016. Exploratory analysis of social media prior to a suicide attempt. In *Proceedings of the third workshop on computational linguistics and clinical psychology*, pages 106–117.

Daryl J Daley and David Vere-Jones. 2003. *An introduction to the theory of point processes: volume I: elementary theory and methods*. Springer.

Munmun De Choudhury and Sushovan De. 2014. Mental health discourse on reddit: Self-disclosure, social support, and anonymity. In *Eighth international AAAI conference on weblogs and social media*.

Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering Shifts to Suicidal Ideation from Mental Health Content in Social Media. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 2098–2110, San Jose California USA. ACM.

Paul Fearnhead and Zhen Liu. 2007. On-line inference for multiple changepoint problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(4):589–605.

Yang Gao, Li Zhou, Yong Zhang, Chunxiao Xing, Yigang Sun, and Xianzhong Zhu. 2010. Sentiment classification for stock news. In *5th International Conference on Pervasive Computing and Applications*, pages 99–104. IEEE.

Manas Gaur, Amanuel Alambo, Joy Prakash Sain, Ugur Kursuncu, Krishnaprasad Thirunarayan, Ramakanth Kavuluru, Amit Sheth, Randy Welton, and Jyotishman Pathak. 2019. Knowledge-aware assessment of severity of suicide risk for early intervention. In *The World Wide Web Conference*, pages 514–525.

George Gkotsis, Anika Oellrich, Sumithra Velupillai, Maria Liakata, Tim JP Hubbard, Richard JB Dobson, and Rina Dutta. 2017. Characterisation of mental health conditions in social media using informed deep learning. *Scientific reports*, 7(1):1–11.

Valentinus Roby Hananto, Uwe Serdült, and Victor Kryssanov. 2022. A text segmentation approach for automated annotation of online customer reviews, based on topic modeling. *Applied Sciences*, 12(7):3412.

Po Hu, Minlie Huang, Peng Xu, Weichang Li, Adam K Usadi, and Xiaoyan Zhu. 2011. Generating breakpoint-based timeline overview for news topic retrospection. In *2011 IEEE 11th International Conference on Data Mining*, pages 260–269. IEEE.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8.

Jagroop Kaur and Jaswinder Singh. 2019. Deep neural network based sentence boundary detection and end marker suggestion for social media text. In *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pages 292–295. IEEE.

Seunghyun Kim, Afsaneh Razi, Gianluca Stringhini, Pamela Wisniewski, and Munmun De Choudhury. 2021. You don't know how i feel: Insider-outsider perspective gaps in cyberbullying risk detection. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 15, pages 290–302.

Jeremias Knoblauch and Theodoros Damoulas. 2018. Spatio-temporal Bayesian on-line changepoint detection with model selection. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2718–2727. PMLR.

Jiwei Li and Claire Cardie. 2014. Timeline generation: tracking individuals on twitter. In *Proceedings of the 23rd international conference on World wide web - WWW '14*, pages 643–652, Seoul, Korea. ACM Press.

Jiwei Li, Alan Ritter, Claire Cardie, and Eduard Hovy. 2014. Major life event extraction from twitter based on congratulations/condolences speech acts. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1997–2007.

David E Losada and Fabio Crestani. 2016. A test collection for research on depression and language use. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 28–39. Springer.

David E Losada, Fabio Crestani, and Javier Parapar. 2020. Overview of erisk at clef 2020: Early risk prediction on the internet (extended overview). *CLEF (Working Notes)*.

Kate Loveys, Patrick Crutchley, Emily Wyatt, and Glen Coppersmith. 2017. Small but mighty: affective micropatterns for quantifying mental health from social media language. In *Proceedings of the fourth workshop on computational linguistics and clinical Psychology—From linguistic signal to clinical reality*, pages 85–95.

Daniel M Low, Laurie Rumker, Tanya Talkar, John Torous, Guillermo Cecchi, and Satrajit S Ghosh. 2020. Natural language processing reveals vulnerable mental health support groups and heightened health anxiety on reddit during covid-19: Observational study. *Journal of medical Internet research*, 22(10):e22635.

Sebastian Martschat and Katja Markert. 2017. Improving ROUGE for Timeline Summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 285–290, Valencia, Spain. Association for Computational Linguistics.

Sebastian Martschat and Katja Markert. 2018. A Temporally Sensitive Submodularity Framework for Timeline Summarization. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 230–240, Brussels, Belgium. Association for Computational Linguistics.

Naoki Masuda, Issei Kurahashi, and Hiroko Onari. 2013. Suicide ideation of individuals in online social networks. *PloS one*, 8(4):e62262.

Matthew Matero, Akash Idnani, Youngseo Son, Salvatore Giorgi, Huy Vu, Mohammad Zamani, Parth Limbachiya, Sharath Chandra Guntuku, and H Andrew Schwartz. 2019. Suicide risk assessment with multi-level dual-context language and bert. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 39–44.

Rohan Mishra, Pradyumn Prakhar Sinha, Ramit Sawhney, Debanjan Mahata, Puneet Mathur, and Rajiv Ratn Shah. 2019. SNAP-BATNET: Cascading Author Profiling and Social Network Graphs for Suicide Ideation Detection on Social Media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 147–156, Minneapolis, Minnesota. Association for Computational Linguistics.

Martha Neary and Stephen M Schueller. 2018. State of the field of mental health apps. *Cognitive and Behavioral Practice*, 25(4):531–537.

Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Emre Kıcıman. 2019. Social data: Biases, methodological pitfalls, and ethical boundaries. *Frontiers in Big Data*, 2:13.

Oyebade K Oyedotun and Adnan Khashman. 2016. Document segmentation using textural features summarization and feedforward neural network. *Applied Intelligence*, 45(1):198–212.

Irina Pak and Phoey Lee Teh. 2018. Text segmentation techniques: a critical review. *Innovative Computing, Optimization and Its Applications*, pages 167–181.

Lev Pevzner and Marti A Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36.

Yada Pruksachatkun, Sachin R. Pendse, and Amit Sharma. 2019. Moments of change: Analyzing peer-based cognitive support in online mental health forums. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA. Association for Computing Machinery.

Martin Riedl and Chris Biemann. 2012. TopicTiling: A text segmentation algorithm based on LDA. In *Proceedings of ACL 2012 Student Research Workshop*, pages 37–42, Jeju Island, Korea. Association for Computational Linguistics.

Murray Rosenblatt. 1956. Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27(3):832 – 837.

Koustuv Saha and Amit Sharma. 2020. Causal factors of effective psychosocial outcomes in online mental health communities. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 590–601.

Ramit Sawhney, Harshit Joshi, Lucie Flek, and Rajiv Shah. 2021. Phase: Learning emotional phase-aware representations for suicide ideation detection on social media. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2415–2428.

Ramit Sawhney, Harshit Joshi, Saumya Gandhi, and Rajiv Shah. 2020. A time-aware transformer based model for suicide ideation detection on social media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7685–7697.

David W Scott. 2015. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Ashish Sharma, Monojit Choudhury, Tim Althoff, and Amit Sharma. 2020a. Engagement Patterns of Peer-to-Peer Interactions on Mental Health Platforms. *Proceedings of the International AAAI Conference on Web and Social Media*, 14:614–625.

Ashish Sharma, Adam Miner, David Atkins, and Tim Althoff. 2020b. A Computational Approach to Understanding Empathy Expressed in Text-Based Mental Health Support. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5263–5276, Online. Association for Computational Linguistics.

Han-Chin Shing, Suraj Nair, Ayah Zirikly, Meir Friedenberg, Hal Daumé III, and Philip Resnik. 2018. Expert, crowdsourced, and machine assessment of suicide risk via online postings. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 25–36.

Julius Steen and Katja Markert. 2019. Abstractive Timeline Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 21–

31, Hong Kong, China. Association for Computational Linguistics.

Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. 2013. Probabilistic chinese word segmentation with non-local information and stochastic training. *Information Processing & Management*, 49(3):626–636.

Russell Swan and James Allan. 2000. Automatic generation of overview timelines. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56.

Ryuichi Takanobu, Minlie Huang, Zhongzhou Zhao, Feng-Lin Li, Haiqing Chen, Xiaoyan Zhu, and Liqiang Nie. 2018. A weakly supervised method for topic segmentation and labeling in goal-oriented dialogues via reinforcement learning. In *IJCAI*, pages 4403–4410.

Adam Tsakalidis, Maria Liakata, Theo Damoulas, and Alexandra I Cristea. 2018. Can we assess mental health through social media and smart devices? addressing bias in methodology and evaluation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 407–423. Springer.

Adam Tsakalidis, Federico Nanni, Anthony Hills, Jenny Chim, Jiayu Song, and Maria Liakata. 2022a. Identifying moments of change from longitudinal user text. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long papers)*, volume (to appear).

Adam Tsakalidis et al. 2022b. Overview of the CLPsych 2022 shared task: Capturing moments of change in longitudinal user posts. In *Proceedings of CLPsych*.

Gerrit JJ van den Burg and Christopher KI Williams. 2020. An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222*.

Shang Wang, Zhiwei Yang, and Yi Chang. 2021. Bringing order to episodes: Mining timeline in social media. *Neurocomputing*, 450:80–90.

Yun Wu, Yan Zhang, Si-ming Luo, and Xiao-jie Wang. 2007. Comprehensive information based semantic orientation identification. In *2007 International Conference on Natural Language Processing and Knowledge Engineering*, pages 274–279. IEEE.

Ayah Zirikly, Philip Resnik, Ozlem Uzuner, and Kristy Hollingshead. 2019. Clpsych 2019 shared task: Predicting the degree of suicide risk in reddit posts. In *Proceedings of the sixth workshop on computational linguistics and clinical psychology*, pages 24–33.

A Appendix

A.1 Creating Ground-truth Timelines, by Retaining a Subset of Representative Candidate Timelines

In addition to the details provided in section 3, for selecting candidate timelines, we provide some additional details inline below. As multiple timelines will typically be returned for each user using methods in 3 and annotating all of these can be time-consuming, in order to keep the 500 annotated ground-truth timelines relatively diverse in terms of the types of users – only a single timeline was returned per user to be annotated. Therefore, for each user only a single timeline was randomly sampled per and these were presented visually in turn to the first author of this paper, with multiple time-scales limiting the x-axis of the visualization returned: (1) the time-scale of the whole user's history, (2) a radius of 200 days surrounding the CMoC and (3) a radius of 31 days around the CMoC. This was to ensure that the candidate timelines could be inspected in close detail (3), and also observing the timeline in context of the full time-series (1) for that user. These three multiple time-scales for a single user are presented visually in figure 6. A manual binary decision was then made on whether to discard this timeline or retain it to be annotated and thereby create a ground-truth timeline using it. This decision was based on a time-series visualization of the frequency of daily posts for that user and highlighting the location of the timeline to be either retained or discarded. The decision to discard a timeline was based on two criteria: whether the timeline (1) was primarily sparse over the full 15 days of the timelines, or to a lesser degree (2) whether it appeared that the CMoC was generated by noise. It was chosen to discard timelines that were (1) primarily sparse, to ensure that we allow sufficient amount of time to pass between posts such that moments of change can occur. Timelines that appeared to be (2) generated by noise, were discarded such that the ground-truth timelines were representative of timelines that would be generated by a change-point detection algorithm with well chosen hyper-parameters – as the retained timelines were thus timelines that appeared to be generated by realistic change-points. Figure 7 presents a visualisation of a timeline that was discarded as described above, and figure 6 describes a timeline that was included to be annotated as a ground-truth timeline.

This process of visually deciding whether a randomly sampled candidate timeline should be retained to be converted into a ground-truth timeline was repeated until 500 candidate timelines were retained. This process thus lasted until 1,220 randomly sampled timelines were observed and thus 720 timelines were discarded.

From the annotated timelines, medoids are re-

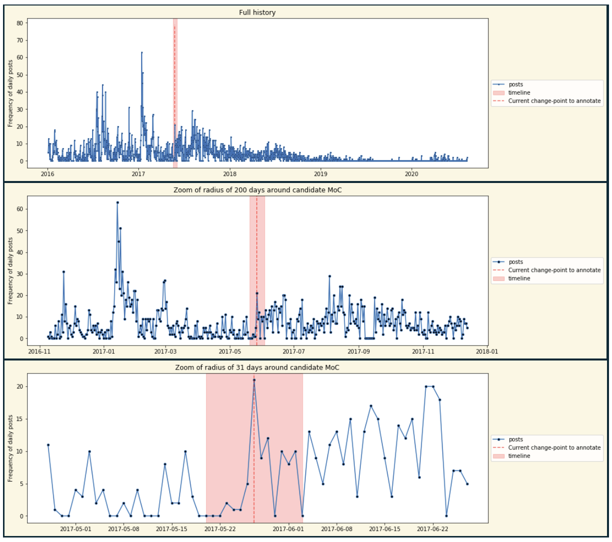


Figure 6: A timeline that was retained, out of the 1,220 timelines manually observed. It was retained as it (1) was not primarily sparse as it contains posts distributed well over the timeline, and (2) appeared to be generated by a plausible change-point rather than noise. Timelines were visualized on 3 time-scales, as shown in this figure, to allow for closer inspection and to compare in context of the full time-series.

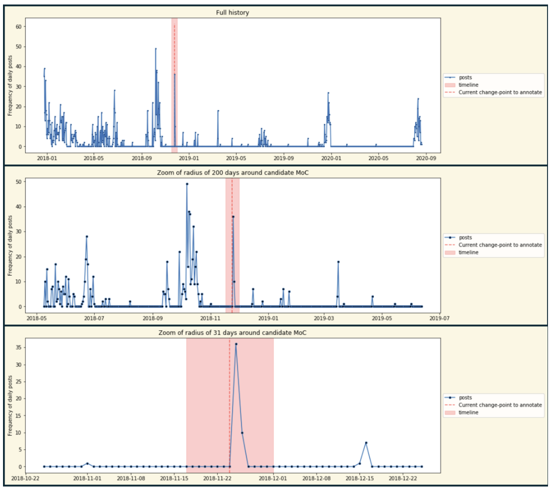


Figure 7: A timeline that was discarded, out of the 1,220 timelines manually observed. It was discarded as it (1) was primarily sparse containing only posts on a few days in the timeline, and (2) appeared to be generated by noise rather than by a realistic change-point.

turned as the medoid timestamp of the annotated GTMoC after annotations were union aggregated across all annotators as described in (Tsakalidis et al., 2022a).

A.2 Annotation Guidelines

The annotation task proposed by (Tsakalidis et al., 2022a) was to assign annotators to identify changes in mood, by reading through the posts in chronological order included within the generated timeline of an individual – and annotating the posts which contain a change in the user's mood compared to the recent past.

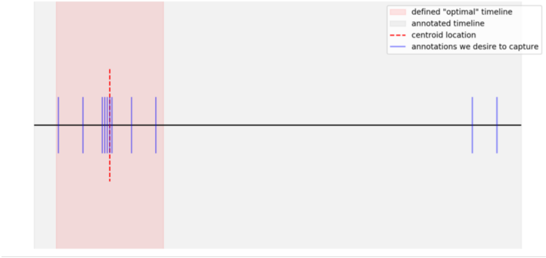
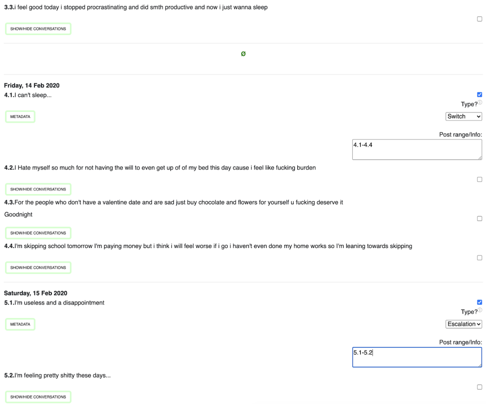


Figure 8: Identifying the position of the medoid, from the timestamps of posts annotated as GTMoCs.

An example illustrating both a switch, and an escalation are displayed in figure 9. Note, that the example shown in this figure will be paraphrased before the work is published – to further preserve anonymity of this user.


3.3.i feel good today i stopped procrastinating and did smth productive and now i just wanna sleep

☐

Friday, 14 Feb 2020
4.1.I can't sleep...

☑

Type? Switch

Post range/Info: 4.1-4.4

4.2.I Hate myself so much for not having the will to even get up of of my bed this day cause i feel like fucking burden

☐

4.3. For the people who don't have a valentine date and are sad just buy chocolate and flowers for yourself u fucking deserve it

Goodnight

☐

4.4.I'm skipping school tomorrow I'm paying money but i think i will feel worse if i go i haven't even done my home works so I'm leaning towards skipping

☐

Saturday, 15 Feb 2020
5.1.I'm useless and a disappointment

☑

Type? Escalation

Post range/Info: 5.1-5.2

5.2.I'm feeling pretty shitty these days...

☐

Figure 9: An example of the annotation interface, displaying a sequence of posts in a timeline shown to an annotator. For these sequence of posts, the annotator annotated a single post as a "switch" and another post as an "escalation". The user has a "switch" at 4.1, drastically changing from a positive mood to a negative mood – where this changed mood persists until 4.4. The "escalation" begins and is at its peak (in this case becoming increasingly negative) at 5.1, and de-escalates up to the post at 5.2."

Semi-supervised New Event Type Induction and Description via Contrastive Loss-Enforced Batch Attention

Carl Edwards and **Heng Ji**
University of Illinois Urbana-Champaign
`{cne2, hengji}@illinois.edu`

Abstract

Most event extraction methods have traditionally relied on an annotated set of event types. However, creating event ontologies and annotating supervised training data are expensive and time-consuming. Previous work has proposed semi-supervised approaches which leverage seen (annotated) types to learn how to automatically discover new event types. State-of-the-art methods, both semi-supervised or fully unsupervised, use a form of reconstruction loss on specific tokens in a context. In contrast, we present a novel approach to semi-supervised new event type induction using a masked contrastive loss, which learns similarities between event mentions by enforcing an attention mechanism over the data minibatch. We further disentangle the discovered clusters by approximating the underlying manifolds in the data, which allows us to achieve an adjusted rand index score of 48.85%. Building on these clustering results, we extend our approach to two new tasks: predicting the type name of the discovered clusters and linking them to FrameNet frames.[1]

1 Introduction

Discovering new event types is an important step for adapting information extraction (IE) methods to unseen domains. Existing work (Ji and Grishman, 2008; McClosky et al., 2011; Li et al., 2013; Chen et al., 2015; Du and Cardie, 2020; Li et al., 2021a) traditionally uses a predefined list of event types and their respective annotations to learn an event extraction model. However, these annotations are both expensive and time-consuming to create. This problem is amplified when considering specialization-intensive domains such as scientific literature, which requires years of specialized experience to understand even a specific niche. For example, there are a wide range of otherwise obscure events in biomedical literature (Krallinger et al., 2017), and better IE techniques can empower life-changing breakthroughs in these domains. To adapt IE to these specialized domains, it is critical to discover new event types automatically.

There are two primary approaches in event type induction. The first is completely unsupervised induction. It includes recent neural techniques (Huang et al., 2016; Shen et al., 2021), as well as ad-hoc clustering techniques (Sekine, 2006; Chambers and Jurafsky, 2011) and probabilistic generative methods (Cheung et al., 2013; Chambers, 2013; Nguyen et al., 2015). The second approach, semi-supervised event type induction, was recently introduced by Huang and Ji (2020). It proposes leveraging annotations for existing types to learn to discover new types; this enables taking advantage of existing resources. In this work, we pursue the second approach.

Current state-of-the-art work in event type induction (Huang and Ji, 2020; Shen et al., 2021) uses reconstruction-based losses to find clusters of new types. Motivated by recent success in learning representations with contrastive loss (Chen et al., 2020a; Radford et al., 2021), we propose a novel alternative approach using batch attention and contrastive loss, which achieves state-of-the-art results. Essentially, we consider the attention weight between two event mentions as a learned similarity, and we ensure that the attention mechanism learns to align similar events using a semi-supervised contrastive loss. By doing this, we are able to leverage the large variety of semantic information in pretrained language models for clustering unseen types by using a trained attention head. This reveals our first key motivation: unlike (Huang and Ji, 2020), we are able to separate clustering from learning, allowing specific task-suited clustering algorithms to be selected. This allows us to test multiple clustering strategies after training once.

---

[1] The programs and resources will be publicly available at github.com/cnedwards/EventTypeBatchAttention for research purposes.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3805–3827
May 2-6, 2023 ©2023 Association for Computational Linguistics

Additionally, this easily allows the use of hierarchical clustering, which may be beneficial to some downstream tasks.

Batch attention is an attention mechanism taken over a minibatch of samples rather than a sequence. Previous uses of batch attention have been limited. Primarily, it has been used for image classification (Cheng et al., 2021) and satellite imagery (Su et al., 2019). In this work, we apply batch attention to natural language instead, which we use for clustering, and we propose the novel idea of enforcing the attention mechanism using contrastive loss.

To enable our discovered event types to be used in larger IE systems, it is important to extract information regarding the clusters. Previous work has looked to describe clusters—for a given cluster, Huang et al. (2016) uses the nearest trigger to the cluster centroid as its name. However, this approach is nebulous and not easily measurable (because the same trigger can correspond to different event types and there is not a quantitative method to determine if the selected trigger defines the cluster well). This is our second key motivation: we instead introduce two new information retrieval-styled tasks for describing the cluster – type name prediction and FrameNet (Baker et al., 1998) frame linking. Type prediction predicts a name for each cluster and is a relatively easy task. FrameNet linking builds on this by linking event types to relevant frames, and is significantly more useful for downstream applications. Our attention-based approach is especially useful here, since it uses the attention mechanism to produce "clustered" features which can have auxiliary task-specific losses applied (prior work is not well-suited for this because the loss is applied to individual data points). This allows our method to build clusters which are more amenable to downstream tasks.

The major novel contributions of this paper are:

- We propose a novel framework for new event type induction which uses a novel masked contrastive loss to enforce an attention mechanism over data minibatches. This framework is also potentially applicable for semi-supervised clustering and classification problems in other settings where a pretrained model exists.

- We show that the base pretrained model selected for event type induction plays a key role in the types which are discovered, since even un-finetuned models rival Huang and Ji (2020).

- We use the "clustered" features produced by our model to extend new event type induction to two novel downstream tasks: type name prediction and FrameNet linking. We show our architecture design allows for auxiliary losses which improve performance on these tasks.

2 Task Descriptions

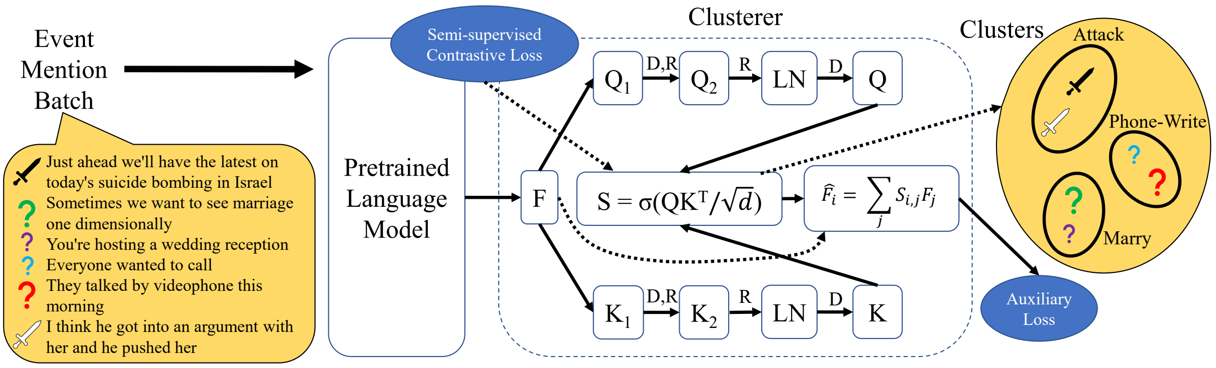
2.1 Semi-supervised Event Type Induction

We tackle the problem of semi-supervised event type induction, first described by Huang and Ji (2020). The task is defined as follows: Assume that k event types from some dataset are known and that the types of the other events are unknown. Using the known types as example clusters, we seek to discover type clusters for the unknown types. Essentially, this is a semi-supervised clustering task on event mentions. In this work, we follow Huang et al. (2018) and set the 10 most common types in the ACE 2005 dataset (Walker et al., 2005) as known. Thus, given all ACE annotated event mentions, our goal is to automatically discover the other 23 unseen ACE types. This assumption is likely to carry over to real-world datasets, since existing 'seen' type definitions are most likely to cover the most common events. In many real-world cases, there may be a few long-tail event types present in the 'seen' types and one or two very common event types may be 'unseen'. Regardless, the distribution of seen and unseen types is likely to be fairly similar to the distribution of setting the most popular k types as known.

2.2 Downstream Clustering Tasks

Beyond clustering, we also introduce two new downstream tasks on this problem: type prediction and FrameNet (Baker et al., 1998) linking. We structure both of these tasks as information retrieval problems for evaluation. Essentially, given a cluster, one should be able to predict its event type name and to what frame it should be linked. For each cluster, we calculate the most frequent type and consider it to be the ground truth for the cluster.

Type Prediction: The goal is to retrieve the "name" of the correct type for a cluster. Thus, we measure Hits@n and mean reciprical rank (MRR), where the corpus consists of the 23 new unseen type names.



Just ahead we'll have the latest on today's suicide bombing in Israel

Sometimes we want to see marriage one dimensionally

You're hosting a wedding reception

Everyone wanted to call

They talked by videophone this morning

I think he got into an argument with her and he pushed her

Figure 1: Architecture of the proposed approach. Best viewed in color. LN is layer normalization, R is ReLU, and D is dropout. σ is softmax for the attention mechanism and sigmoid for the contrastive loss. $\hat{F}_i$ is the clustered features of mention i in the batch. '**?**' are unseen event types.

In practice, we embed the names using our language model and use cosine similarity to the cluster centroid to rank them.

FrameNet Linking: FrameNet is the largest event ontology that is publicly available. However, there is not enough annotated training data to train supervised models directly on it. To alleviate this issue, we propose a task linking our newly discovered event types to FrameNet frames.

For the FrameNet linking, we consider a setup similar to name prediction, where we link clusters to the 1,221 frames in FrameNet 1.7 (Ruppenhofer et al., 2016). Instead of using the type names, we follow Huang et al. (2018) and manually map the ACE types to one or more frames to create a ground truth (see Appendix C for details). All child frames of the mapped frames are also considered valid targets. Given an ACE type, we can now link to a set of valid frames. In practice, we take the corpus of frame definitions and embed them using our language model. We then rank them using cosine similarity by comparing to the given cluster centroid. We consider the best rank of the valid frames to be the rank of a cluster. Thus, this task is also measured with Hits@n and MRR.

3 Methods

3.1 Overall Architecture

Overall, our method, shown in Figure 1, consists of a language model, such as BERT (Devlin et al., 2019), which produces contextualized representations, followed by a "clusterer". Unlike previous work which used specific token representations (Huang and Ji, 2020), we use mean pooling over the entire mention where an event occurs as our input. The language model produces an event representation, which is then input into the "clusterer" layer. The clusterer layer then produces "clustered" features using the attentions (see Section 3.3).

3.2 Back-translation for Data Augmentation

Contrastive loss has recently been applied for deep clustering (Li et al., 2021b; Zhong et al., 2020) and for representation learning (Chen et al., 2020a; Gao et al., 2021; Zhang et al., 2021a; Liu and Liu, 2021). However, this requires data augmentation to create positive example pairs. For text, some augmentations use back-translation (Cao and Wang, 2021; Zhang et al., 2021b). Taking inspiration from these clustering and representation learning techniques, we employ back-translation as data augmentation to create more positive pairs, improving the learning of attention weights between event mentions.

3.3 Batch Attention "Clusterer" Mechanism

To learn similarities between unseen event mentions, we propose learning an attention mechanism over the stochastic gradient descent minibatch. We enforce this attention mechanism using a masked contrastive loss (described in Section 3.4). This allows the attention mechanism's behavior to be learned from the seen classes.

We follow Vaswani et al. (2017) in implementing a scaled dot product attention, although over the batch instead. Since our "clusterer" needs to learn similarities for clustering and then be used for cluster features, we use nonlinear transformations for the query (Q) and key (K) vectors instead of the linear transformations in (Vaswani et al., 2017). This nonlinear transformation for Q and K is implemented as a two hidden layer neural network, which is shown in Figure 1.

Using this attention mechanism, we produce "clustered features", which are a convex combination of the different samples from the batch. This allows us to apply an auxiliary loss to the clustered features. We consider this as being analogous to learning on cluster centroids. Specific auxiliary losses can be applied for specific downstream tasks.

We note that this approach can also be interpreted as a type of feature smoothing, an inner product graph generator, and metric learning.

3.4 Masked Semi-supervised Contrastive Loss

Recent work, such as CLIP (Radford et al., 2021) and Text2Mol (Edwards et al., 2021), has found great success using contrastive losses between pairs of representations Q and K, each $n \times d$ matrices where n is the number of samples of d dimensions. They obtain the loss L by comparing the product of these matrices (QK^T) to a label matrix $Y \in \{0,1\}^{n \times n}$ (which in their case is $Y = I_n$), using cross entropy loss CE.

$$L(Q,K) = CE(QK^T, I_n) + CE(KQ^T, I_n)$$

We modify these existing contrastive losses to enforce our batch attention mechanism. We calculate the label matrix as follows (see Appendix B for an example). Given a pair of samples (event mentions) x_i and x_j, we consider the pair to be a positive if they are from the same seen event type. We consider the pair to be negative if they are from different seen event types or if one is seen and one unseen. If both are unseen they are masked. In practice, the labels can be computed using one-hot vectors of the c seen types (the unseen types are zero-hot vectors). These vectors are stacked into a $n \times c$ matrix O. The label matrix is computed

$$Y = OO^T \vee I_n$$

where $\vee$ is the elementwise logical-or operation. Following (Edwards et al., 2021), we use binary cross-entropy as the loss between the labels Y and the scaled attention dot products $\frac{QK^T}{\sqrt{d}}$ from (Vaswani et al., 2017). This gives the following contrastive loss:

$$L_{ss}(Q,K) = CE(\frac{QK^T}{\sqrt{d}}, Y)$$

This loss, however, values negative samples much more than positive samples (due to the imbalance). Noticing that once vectors of a negative pair are orthogonal they don't need to be further separated, we introduce a margin m. Essentially, we mask out pairs whose dot product is "too negative" (in addition to unknown relations between unseen types). This is because the loss would rather optimize the already well-separated negatives instead of the relatively fewer positives. Let $p_{i,j} \in \{0,1\}$ be the label of a pair and $u_{i,j} \in \{0,1\}$ indicate that both i and j are unseen. Our mask, M, is calculated

$$M_{i,j} = \overline{\left[\overline{p_{i,j}} \wedge \left(\sigma(\frac{QK^T}{\sqrt{d}})_{i,j} < m\right)\right] \vee u_{i,j}}$$

where $\wedge$ is elementwise logical-and, σ is the sigmoid function, and $\bar{z}$ denotes logical negation of z. This is similarly motivated to the margins used in knowledge graph embedding losses, such as TransE (Bordes et al., 2013). Thus, our loss is:

$$L_m(Q,K) = M \cdot L_{ss}(Q,K)$$

where in this case we treat $L_{ss}(Q,K)$ as an unreduced loss (so it is a matrix), and $\cdot$ is elementwise multiplication.

We apply this loss to the query (Q) and key (K) matrices in the clusterer's batch attention mechanism. We also include the augmented data (Q' and K'), giving us a final loss:

$$L_c(Q,K,Q',K') = \sum_{\hat{Q},\hat{K} \in \{Q,Q'\} \times \{K,K'\}} L_m(\hat{Q},\hat{K})$$

3.5 Auxiliary Loss

For our downstream tasks, we employ a regression-based auxiliary loss. For each seen instance x_i, we maximize the cosine similarity between the clustered features $\hat{F}_i$ and the pretrained language model embedding B_{t_i} of the ground truth type t_i (e.g. the name 'attack'). Thus, we get the loss:

$$L_a(\hat{F}_i, B_{t_i}, t_i) = 1 - \cos(\hat{F}_i, B_{t_i})\mathbb{1}_{\text{seen}}(t_i)$$

where $\mathbb{1}_{\text{seen}}(t_i)$ indicates whether t_i is a seen type.

3.6 Stopping Criterion

For this task, it is not reasonable to use a validation set for stopping. This is because the loss depends only on seen types and their relationships to unseen types. Since the unseen classes are unlabeled and the losses between pairs of unseen are unknown, the model can overfit to the seen data, pushing together clusters of unseen types. To deal with this issue, we employ unsupervised clustering metrics to decide when to stop training. In particular, we use cosine distance-based silhouette scores to measure the quality of clustering. Further details are given in Appendix F.

3.7 Clustering

Any algorithm which can compute clusters from a precomputed distance function can be applied to the learned similarities between event mentions. Additionally, we find that the finetuning of the language model by our loss modifies its representations to better form clusters. Thus, this representation can be used in many clustering algorithms as well.

3.7.1 Manifold Approximation

Inspired by recent work (Ros et al., 2021) which uses manifold approximation to interpret large language model-based sentence representations for information retrieval, we incorporate manifold approximation into our clustering approach. To do so, we follow the UMAP (McInnes et al., 2018) algorithm to create approximate weights based on estimating neighborhood densities within the data. We calculate these weights using cosine distance as an input, as it has traditionally been effective for language modeling (Manning et al., 2008; Reimers and Gurevych, 2019). UMAP attempts to estimate the density by comparing the distance to the k-nearest neighbors. This is used to calculate weights between each pair of data points. Details are given in Appendix H. Following this, we use agglomerative clustering on the UMAP weights as before.

In our approach, we want to better understand the global clustering landscape, so we use a high value of k. In practice, to avoid hyperparameter selection, we set k equal to the size of the data.

4 Experimental Results

Generally, we used default hyperparameters. We split the learning rates into BERT and non-BERT parameters following (Edwards et al., 2021) with 2e-5 for BERT as in (Devlin et al., 2019) and 1e-4 for other parameters as in (Vaswani et al., 2017). For the margin parameter, we examined silhouette scores to select 0.5.

For back-translation, we used four languages, German, French, Spanish, and Chinese, and randomly sampled which language to use for each data point every epoch. We obtained back-translations using the MarianMT translation models (Junczys-Dowmunt et al., 2018). Ablations are shown in Section J.

For our main experiments, we only use the contrastive loss. We take the average of 5 runs to show that our method consistently outperforms (Huang and Ji, 2020). We also calculate clusters using an ensemble of the 5 runs which shows slightly increased performance, which is an expected result in deep neural networks (Allen-Zhu and Li, 2020).

Huang and Ji (2020) evaluate these clusters using Geometric NMI, Fowlkes Mallows (Fowlkes and Mallows, 1983), Completeness, Homogeneity, and V-Measure (Rosenberg and Hirschberg, 2007). We additionally consider adjusted Rand index (ARI) (Hubert and Arabie, 1985). In the downstream tasks, given a clustering we also report the average cluster purity and type representation. Given a cluster i of size n_i with most frequent type numbering n_{f_i}, purity $p_i = \frac{n_i}{n_{f_i}}$ (Manning et al., 2008). Note that this average cluster purity is slightly different than traditional purity; it weights small clusters more which is desirable in our case (like macro vs. micro F1 score). Type representation is the number of unique frequent subtypes, n_t, divided by total types, in this case 23.

4.1 Language Model

We select Sentence BERT (SBERT) (Reimers and Gurevych, 2019) as a language model because its pretraining tasks are better suited for clustering than BERT. This is shown in Table 1, since the clustering from SBERT embeddings can even outperform (Huang and Ji, 2020) without any semi-supervision. We use a small version of the model[2] from HuggingFace (Wolf et al., 2020), which allows us to use a larger minibatch size of 10. Using larger minibatch sizes is desirable for contrastive loss since the number of negatives scales quadratically with the size. The performance of mini SBERT is notable, as Huang and Ji (2020) used BERT-large, a considerably larger model.

4.2 Clustering Algorithms

For clustering, we consider two algorithms which work on precomputed metrics. First, we use agglomerative clustering with average linkage, as it tends to be less sensitive to outliers and noisy data (Han et al., 2011). Noise is present in the dataset, often in the form of transcripts (see Section 4.4).

We report results following existing clustering literature by using the true number of classes as the cluster number (Huang et al., 2020; Li et al., 2021b). In practice it is generally difficult to select the correct number of clusters to use. Due to this, using extra clusters is typically done by previous

---

[2] paraphrase-MiniLM-L12-v2

Method		Clusters	Geometric NMI	Fowlkes Mallows	Completeness	Homogeneity	V-Measure	ARI
One Cluster		1	0.00	25.58	100.00	0.00	0.00	0.00
SS-VQ-VAE w/o VAE (Huang and Ji, 2020)		500	33.45	25.54	42.76	26.17	32.47	-
SS-VQ-VAE (Huang and Ji, 2020)		500	40.88	31.46	53.57	31.19	39.43	-
SS-VQ-VAE + SBERT [3]		33	19.08	19.45	25.80	15.13	19.08	7.54
SBERT	Agglo	23	50.71	34.35	57.05	45.07	50.36	24.02
SBERT	Manifold	23	48.75	36.02	51.32	46.30	48.68	30.21
Attn-Cosine	Agglo	23	46.40	34.60	49.82	43.24	46.27	26.69
Attn-DotProduct	Agglo	23	50.17	37.48	53.50	47.06	50.06	30.13
Attn	Manifold	23	54.83	42.77	55.00	54.67	54.82	38.74
FT-SBERT	Manifold	23	60.28	50.63	60.19	60.37	60.28	47.24
Attn-DotProduct	Affinity	49-68	56.87	35.64	49.58	65.26	56.33	30.02
Attn-Cosine	Affinity	50-69	56.54	33.00	48.72	65.62	55.91	27.04
E-Attn-DotProduct	Agglo	23	56.50	43.26	59.62	53.54	56.41	37.02
E-Attn	Agglo	23	59.00	46.19	58.36	59.66	59.00	42.56
E-FT-SBERT	Manifold	23	**63.56**	**52.10**	**63.11**	64.01	**63.56**	**48.85**
E-Attn-DotProduct	Affinity	63	60.00	38.41	51.32	**70.15**	59.28	31.78

Table 1: New event type induction results (%)[4]. The first subcolumn is the input for clustering and the second is the clustering algorithm used. SBERT indicates the (unfinetuned) SBERT representations were used rather than our learned attentions (Attn). FT-SBERT representations are finetuned by our method. E stands for ensemble. Values are the average of 5 runs. Agglo is agglomerative clustering, Affinity is (Frey and Dueck, 2007), and Manifold is as described in Section 3.7.1. For affinity, each run can produce a slightly different number of clusters.

work (Huang and Ji, 2020; Shen et al., 2021). However, this can inflate the NMI score (Nguyen et al., 2009) and benefit qualitative evaluation because of the unbalanced classes in the dataset. As an example, given only 23 clusters (the ground truth), a large class such as 'Injure' splits into multiple smaller clusters, which causes rare event types to be merged. Results show that 19 / 23 types are represented by a cluster in the 50 cluster case versus only 16 / 23 in the 23 cluster case. This makes results appear better for more clusters. Silhouette scores are higher for 23 clusters, however.

Unlike existing work (Huang and Ji, 2020), the number of clusters is unimportant for our learning process and can be selected afterwords, such as by selecting a high number as in (Huang and Ji, 2020; Shen et al., 2021) or automatically with affinity propagation (Frey and Dueck, 2007). Affinity propagation selects exemplars to automatically determine the number of clusters. Our approach is especially useful here, since affinity propagation does not complete when applied to default SBERT representations but does when using our contrastive loss-enforced attentions.

4.3 Results

We compare our results with Huang and Ji (2020), who first introduced this task, in Table 1. We find that just our choice of language model outperforms the baseline. Also, using dot products is more effective for our learned attention metric than cosine distance, since dot product without normalization, as in our attention mechanism, indicates confidence of clustering a pair of samples together (This is because the contrastive loss uses sigmoid).

4.3.1 Manifold Approximation

We find manifold approximation to be very effective in our experiments. Intuitively, we understand this manifold approximation as untangling the cluster manifolds from each other in the high-dimensional representation space. Interestingly, the results using the finetuned SBERT representations perform better than the results on the learned similarities. We find this to be quite interesting, especially because the representations change an average of 0.6 cosine distance from their starting points, as shown in Appendix A. Our method causes SBERT to inherently learn representations more amenable for clustering.

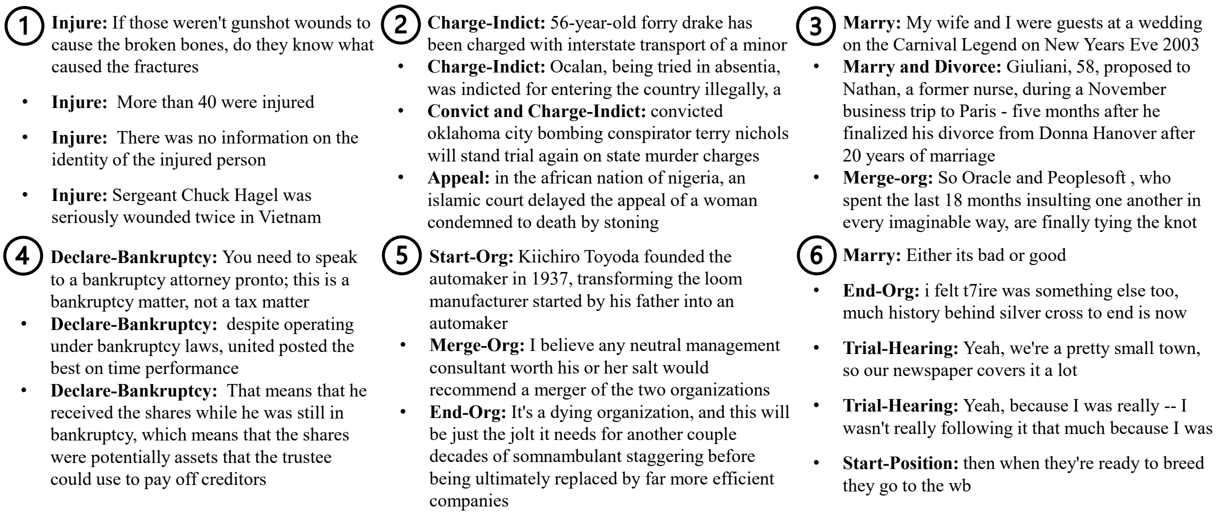
While manifold approximation works well for clustering here, we note that using UMAP for clustering is considered controversial.[5] While it works well in many cases, there are potential issues with artifacts or false tearing of clusters. We leave analysis of the interaction between high-dimensional semantic spaces obtained from language models and manifold approximation to future work.

---

[3] (Huang and Ji, 2020) did not release their code, and we were unable to reproduce their results. Nonetheless, we apply their method to the SBERT representations based on the description in their paper. We also use an equivalent number of clusters to the ground truth, unlike their paper, which enables comparability to our method.

[4] (Huang and Ji, 2020) appears to have used the former scikit-learn default of geometric NMI, which is why their v-score doesn't equal arithmetic NMI.

[5] https://umap-learn.readthedocs.io/en/latest/clustering.html


① **Injure:** If those weren't gunshot wounds to cause the broken bones, do they know what caused the fractures
- **Injure:** More than 40 were injured
- **Injure:** There was no information on the identity of the injured person
- **Injure:** Sergeant Chuck Hagel was seriously wounded twice in Vietnam

② **Charge-Indict:** 56-year-old forry drake has been charged with interstate transport of a minor
- **Charge-Indict:** Ocalan, being tried in absentia, was indicted for entering the country illegally, a
- **Convict and Charge-Indict:** convicted oklahoma city bombing conspirator terry nichols will stand trial again on state murder charges
- **Appeal:** in the african nation of nigeria, an islamic court delayed the appeal of a woman condemned to death by stoning

③ **Marry:** My wife and I were guests at a wedding on the Carnival Legend on New Years Eve 2003
- **Marry and Divorce:** Giuliani, 58, proposed to Nathan, a former nurse, during a November business trip to Paris - five months after he finalized his divorce from Donna Hanover after 20 years of marriage
- **Merge-org:** So Oracle and Peoplesoft , who spent the last 18 months insulting one another in every imaginable way, are finally tying the knot

④ **Declare-Bankruptcy:** You need to speak to a bankruptcy attorney pronto; this is a bankruptcy matter, not a tax matter
- **Declare-Bankruptcy:** despite operating under bankruptcy laws, united posted the best on time performance
- **Declare-Bankruptcy:** That means that he received the shares while he was still in bankruptcy, which means that the shares were potentially assets that the trustee could use to pay off creditors

⑤ **Start-Org:** Kiichiro Toyoda founded the automaker in 1937, transforming the loom manufacturer started by his father into an automaker
- **Merge-Org:** I believe any neutral management consultant worth his or her salt would recommend a merger of the two organizations
- **End-Org:** It's a dying organization, and this will be just the jolt it needs for another couple decades of somnambulant staggering before being ultimately replaced by far more efficient companies

⑥ **Marry:** Either its bad or good
- **End-Org:** i felt t7ire was something else too, much history behind silver cross to end is now
- **Trial-Hearing:** Yeah, we're a pretty small town, so our newspaper covers it a lot
- **Trial-Hearing:** Yeah, because I was really -- I wasn't really following it that much because I was
- **Start-Position:** then when they're ready to breed they go to the wb

Figure 2: Cluster Examples: Injure, Charge-Indict, Marry, Bankruptcy, Start-Org, and Bad Data, respectively.

Cluster Strength	Clusters
Very Strong (> 80% Purity)	Injure, Sue, Phone-Write, Declare-Bankruptcy, Demonstrate, Trial-Hearing
Strong (60-80% Purity)	Be-Born, Start-Position, Charge-Indict, Marry
Ok (40-60% Purity)	Release-Parole, Appeal, Injure
Mixed (20-40% Purity)	Convict, Fine, Trial-Hearing, Start-Org, Start-Position, Charge-Indict
Small Clusters (< 2 samples)	Trial-Hearing, Nominate, Start-Position, Phone-Write

Table 2: Clusters sorted into purity classes.

4.4 Qualitative Cluster Analysis

We analyze the clusters produced by our best result, the ensemble. We classify the clusters according to purity in Table 2. We show examples from numbered clusters in Figure 2. Certain types of clusters, such as Injure ① and Demonstrate ④, form very strong clusters. We believe this is likely related to their size and lack of overlap with other types. There are two common sources of error: the first is semantic overlap. Start-org, merge-org, and end-org tend to overlap ⑤. Marry and divorce also slightly overlap ③—in the 23 cluster case they merge into one cluster, but in the 50 cluster case they are separate. Most types of courtroom related events—Charge-Indict, Trial-Hearing, Convict, Release-Parole, Appeal, Execute, Acquit, Extradite—have some degree of overlap ②. Second, the other main source of errors is "duplicates". This occurs in our method because two or more events can occur in the same event mention ②, ③. Since our method does not account for triggers, it cannot distinguish between duplicate mentions with multiple triggers. Future work can address this issue by combining our method with an existing trigger-based method such as (Huang and Ji, 2020). We also find that our method clusters "junk" data together ⑥, which are usually from transcripts. Errors occasionally occur from metaphorical language, such as when companies are "married" ③. We show more detailed examples of these observations in Appendix E.

4.5 Downstream Tasks

For the downstream tasks, we use different clusterings and try to discover information about the clusters. As a baseline, we compare against default (not finetuned) SBERT clustering (Base-23) and ground truth (perfect) clusters. We compare these to our ensemble clustering. For type prediction, we use SBERT embeddings to compute cluster centroids and compare to the SBERT representation of the type name (e.g. 'injure'). For FrameNet linking, we use the frame definition instead of the name (e.g. "The words [...] describe situations in which an Agent or a Cause injures a Victim [...]"). We also use an auxiliary loss, L_a, which we apply to a 1-layer neural network on the clustered features $\hat{F}$. This extra layer is employed to allow multiple auxilliary losses: we leave those experiments for future work. We compare using these finetuned representations in addition to default SBERT. Results are shown in Tables 3 and 4.

We find that our ensemble clustering outperforms the default SBERT clustering, and that we are able to recover the event type 60% of the time. For the ground truth clusters, our finetuning with

Clustering	Representation	Mean Rank	Hits@1	Hits@3	Hits@5	Hits@10	Hits@15	MRR	Average Purity	Type Representation
Base-23	Untuned	5.17	34.8%	47.8%	60.9%	82.6%	100%	0.477	25%	47.8%
FT-Base-23	Finetuned	4.43	56.5%	65.2%	78.2%	82.6%	91.3%	0.660	58.9%	65.2%
Ensemble-23	Untuned	3.65	60.9%	69.6%	69.6%	95.7%	100%	0.679	68.6%	69.6%
Ensemble-23	Finetuned	5.13	56.5%	65.2%	69.6%	87.0%	87.0%	0.650	68.6%	69.6%
Ensemble-50	Untuned	4.40	56.0%	60.0%	68.0%	90.0%	96.0%	0.630	69.3%	82.6%
Perfect-23	Untuned	2.30	69.6%	73.9%	82.6%	95.7%	100%	0.758	100%	100%
Perfect-23	Finetuned	2.83	73.9%	82.6%	91.3%	91.3%	95.7%	0.800	100%	100%

Table 3: Results for cluster to name prediction task with different representations and clusterings. Finetuned indicates finetuned SBERT representations from our contrastive + auxiliary loss are used (otherwise default SBERT). "-x" at the end is the number of clusters. Perfect indicates the ground truth clustering, Base/FT-Base is SBERT clusterings (default or finetuned with our auxiliary loss), and Ensemble is the clustering of our ensemble result. Type representation shows what percent of unseen types represent the majority of a cluster.

Clustering	Representation	Mean Rank	Hits@1	Hits@5	Hits@10	Hits@50	Hits@100	MRR	Average Purity	Type Representation
Base-23	Untuned	95.9	4.3%	21.7%	26.1%	30.4%	34.8%	0.128	25%	47.8%
FT-Base-23	Finetuned	156.9	30.4%	30.4%	34.8%	43.5%	47.8%	0.336	57.4%	65.2%
Ensemble-23	Untuned	72.7	17.4%	30.4%	39.1%	47.8%	65.2%	0.264	68.6%	69.6%
Ensemble-23	Finetuned	115.7	21.7%	34.8%	34.8%	43.5%	52.2%	0.308	68.6%	69.6%
Perfect-23	Untuned	15.9	26.1%	39.1%	52.2%	65.2%	73.9%	0.374	100%	100%
Perfect-23	Finetuned	42.7	47.8%	56.5%	60.9%	69.6%	69.6%	0.539	100%	100%

Table 4: Results for cluster to frame linking task. See Table 3 for notation.

an auxiliary loss improves MRR and Hits@1 over the default SBERT representations. Frame linking is much more difficult, since there are 1,221 frames, but we are able to recover the correct frame for 30% of clusters, while default SBERT only achieves 4%. Notably, the auxiliary loss clustering (FT-Base-23) even outperforms our ensemble clustering, demonstrating the flexibility of our model architecture. Using perfect clustering, our finetuned model achieves nearly 50% Hits@1, doubling the performance of the default SBERT model. The finetuning loss allows the model to train on the combined cluster features, which approximates the centroid of a cluster. This promotes the potential cluster to be shaped so that its centroid is better suited for the downstream tasks–being most similar to the correct type name or frame representation.

5 Related Work

Although event extraction has long been studied (Grishman, 1997; Ji and Grishman, 2008; McClosky et al., 2011; Li et al., 2013; Chen et al., 2015; Du and Cardie, 2020; Li et al., 2021a), recent focus has turned towards discovering events without annotations. It includes recent neural techniques (Huang et al., 2016; Liu et al., 2019; Shen et al., 2021), as well as ad-hoc clustering techniques (Sekine, 2006; Chambers and Jurafsky, 2011; Yuan et al., 2018) and probabalistic generative methods (Cheung et al., 2013; Chambers, 2013; Nguyen et al., 2015). Semi-supervised event type induction was recently introduced by Huang and Ji (2020). Zero-shot event extraction frameworks, such as (Huang et al., 2018), can be used to perform event extraction on the newly discovered types. Recent work by Gao et al. (2022) uses a weakly-supervised contrastive learning-based clustering approach for event representation learning.

Several new unsupervised deep clustering approaches use contrastive loss for clustering images (Li et al., 2021b; Zhong et al., 2020) and text (Zhang et al., 2021b). These methods require data augmentation to create positive example pairs. Contrastive loss has also been applied to learn representations. SimCLR (Chen et al., 2020a,b) uses image augmentations for unsupervised representation learning. Follow-up work has applied this loss to natural language (Gao et al., 2021; Zhang et al., 2021a; Liu and Liu, 2021), with some augmentations being back-translated text (Cao and Wang, 2021). Gunel et al. (2021) use fully supervised contrastive loss to finetune language models.

Batch attention has been investigated a little in the literature, such as for satellite imagery prediction (Su et al., 2019) or image classification (Cheng et al., 2021); however, it has not been used to learn clustered features. Seidenschwarz et al. (2021) recently proposed a related idea for a message-passing network weighted by attention for clustering images. We instead directly use (contrastive loss-enforced) attention weights for clustering.

6 Conclusion and Future Work

In this work, we present an exciting new approach for event type induction, where we use contrastive loss to control the learning of a batch attention mechanism for both finding and learning about new cluster types. We also consider manifold approximation for clustering, and we introduce two new downstream tasks: type name prediction and FrameNet linking. This new approach opens several interesting problems for future work. First, this method can potentially be incorporated with reconstruction loss-based approaches, which might improve results or obviate the early stopping criterion. Alternatively, the stopping criterion can be integrated into a loss function for better stopping control. Notably, this would enable a two-step process of learning clusters and then performing knowledge distillation using those clusters (or an ensemble) while learning other desired losses. Future work can investigate the interaction of manifold approximation with large language models and integrate it directly into the clusterer subnetwork. It may be possible to use a heirarchy of event types for knowledge transfer from data-rich types to long-tail types. Finally, FrameNet linking can be extended to Wikidata Q-Node linking, which contains millions of nodes. Our approach may also be applicable in other modalities with strong pretrained models.

7 Limitations

In this work, we present a contrastive loss-based batch attention method for new event type induction. Like most modern event type induction methods, this relies on strong existing representations (which are typically pretrained in a self-supervised manner). Our additional testing on the much larger MAVEN dataset (Appendix K) indicates that selecting an appropriate representation is an important factor. In that case, SBERT representations are inappropriate for the event mentions in the dataset, but trigger representations are effective, and our method further improves those results. Future research on identifying pretraining methods for initializing representations will be useful for this method and task (along with many other models such as retrieval-based large language models (Guu et al., 2020; Borgeaud et al., 2021).

Requiring early stopping is a limitation to our newly proposed event type induction method which does not occur in existing reconstruction-based methods (those methods instead require a predetermined cluster number along with other limitations). While an early stopping method is required, our method is effective, and we believe that our new approach is still valuable because it offers a different approach for new event type induction. Additionally, future work should be able to build upon our work to remove early stopping. Doing so may also help obviate the dependence on the initial representation. In Section 6 we detail several possibilities for future work to further improve upon this work.

Acknowledgements

We would like to thank the anonymous reviewers for helping us to improve this paper. This research is based upon work supported in part by U.S. DARPA KAIROS Program No. FA8750-19-2-1004. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *ArXiv preprint*, abs/2012.09816.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2021. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*.

Shuyang Cao and Lu Wang. 2021. Cliff: Contrastive learning for improving faithfulness and factuality in abstractive summarization. *ArXiv preprint*, abs/2109.09209.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Seattle, Washington, USA. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020b. Improved baselines with momentum contrastive learning. *ArXiv preprint*, abs/2003.04297.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

Qishang Cheng, Hongliang Li, Qingbo Wu, and King Ngi Ngan. 2021. Baˆ 2m: A batch aware attention module for image classification. *ArXiv preprint*, abs/2103.15099.

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 837–846, Atlanta, Georgia. Association for Computational Linguistics.

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

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Carl Edwards, ChengXiang Zhai, and Heng Ji. 2021. Text2Mol: Cross-modal molecule retrieval with natural language queries. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 595–607, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Edward B Fowlkes and Colin L Mallows. 1983. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, 78(383):553–569.

Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science*, 315(5814):972–976.

Jun Gao, Wei Wang, Changlong Yu, Huan Zhao, Wilfred Ng, and Ruifeng Xu. 2022. Improving event representation via simultaneous weakly supervised contrastive learning and clustering. *arXiv preprint arXiv:2203.07633*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv preprint*, abs/2104.08821.

Ralph Grishman. 1997. Information extraction: Techniques and challenges. In *International summer school on information extraction*, pages 10–27. Springer.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*, pages 3929–3938. PMLR.

Jiawei Han, Jian Pei, and Micheline Kamber. 2011. *Data mining: concepts and techniques*. Elsevier.

Jiabo Huang, Shaogang Gong, and Xiatian Zhu. 2020. Deep semantic clustering by partition confidence maximisation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 8846–8855. IEEE.

Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, and Avirup Sil. 2016. Liberal event extraction and event schema induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 258–268, Berlin, Germany. Association for Computational Linguistics.

Lifu Huang and Heng Ji. 2020. Semi-supervised new event type induction and event detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 718–724, Online. Association for Computational Linguistics.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*, 2(1):193–218.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Martin Krallinger, Martin Pérez-Pérez, Gael Pérez-Rodríguez, Aitor Blanco-Míguez, Florentino Fdez-Riverola, Salvador Capella-Gutierrez, Anália Lourenço, and Alfonso Valencia. 2017. The biocreative v. 5 evaluation workshop: tasks, organization, sessions and topics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021a. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2021b. Contrastive clustering. In *2021 AAAI Conference on Artificial Intelligence (AAAI)*.

Xiao Liu, Heyan Huang, and Yue Zhang. 2019. Open domain event extraction using neural latent variable models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2860–2871, Florence, Italy. Association for Computational Linguistics.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. Introduction to information retrieval.

David McClosky, Mihai Surdeanu, and Christopher Manning. 2011. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1626–1635, Portland, Oregon, USA. Association for Computational Linguistics.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *ArXiv preprint*, abs/1802.03426.

Kiem-Hieu Nguyen, Xavier Tannier, Olivier Ferret, and Romaric Besançon. 2015. Generative event schema induction with entity disambiguation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International*

Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 188–197, Beijing, China. Association for Computational Linguistics.

Xuan Vinh Nguyen, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 1073–1080. ACM.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Kevin Ros, Carl Edwards, Heng Ji, and ChengXiang Zhai. 2021. Team skeletor at touché 2021: Argument retrieval and visualization for controversial questions. In *CEUR Workshop Proceedings*, volume 2936, pages 2441–2454. CEUR-WS.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

J Ruppenhofer, M Ellsworth, MRL Petruck, CR Johnson, CF Baker, and J Scheffczyk. 2016. Framenet ii: Extended theory and practice (revised november 1, 2016).

Jenny Denise Seidenschwarz, Ismail Elezi, and Laura Leal-Taixé. 2021. Learning intra-batch connections for deep metric learning. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9410–9421. PMLR.

Satoshi Sekine. 2006. On-demand information extraction. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 731–738, Sydney, Australia. Association for Computational Linguistics.

Jiaming Shen, Yunyi Zhang, Heng Ji, and Jiawei Han. 2021. Corpus-based open-domain event type induction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5427–5440, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yanzhou Su, Yongjian Wu, Min Wang, Feng Wang, and Jian Cheng. 2019. Semantic segmentation of high resolution remote sensing image based on batch-attention mechanism. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 3856–3859. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2005. Ace 2005 multilingual training corpus. Technical Report LDC2006T06, Linguistic Data Consortium, Philadelphia.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A massive general domain event detection dataset. In *Proceedings of EMNLP 2020*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Quan Yuan, Xiang Ren, Wenqi He, Chao Zhang, Xinhe Geng, Lifu Huang, Heng Ji, Chin-Yew Lin, and Jiawei Han. 2018. Open-schema event profiling for massive news corpora. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 587–596. ACM.

Dejiao Zhang, Shang-Wen Li, Wei Xiao, Henghui Zhu, Ramesh Nallapati, Andrew O Arnold, and Bing Xiang. 2021a. Pairwise supervised contrastive learning of sentence representations. *ArXiv preprint*, abs/2109.05424.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shang-Wen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang. 2021b. Supporting clustering with contrastive learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5419–5430, Online. Association for Computational Linguistics.

Huasong Zhong, Chong Chen, Zhongming Jin, and Xian-Sheng Hua. 2020. Deep robust clustering by contrastive learning. *ArXiv preprint*, abs/2008.03030.

A How much do SBERT representations change?

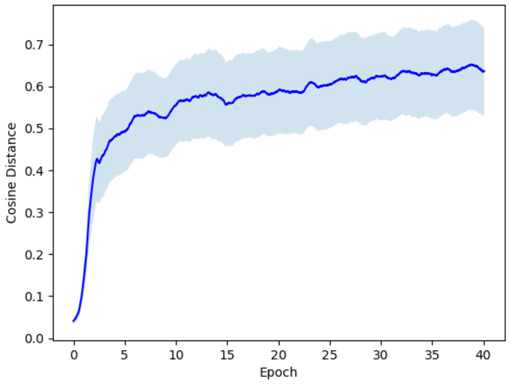


Figure 3: Change in SBERT representations from original representation of inputs. This shows that the representations change significantly from their starting point during finetuning. Shaded area is one standard deviation.

B Label Matrix Example

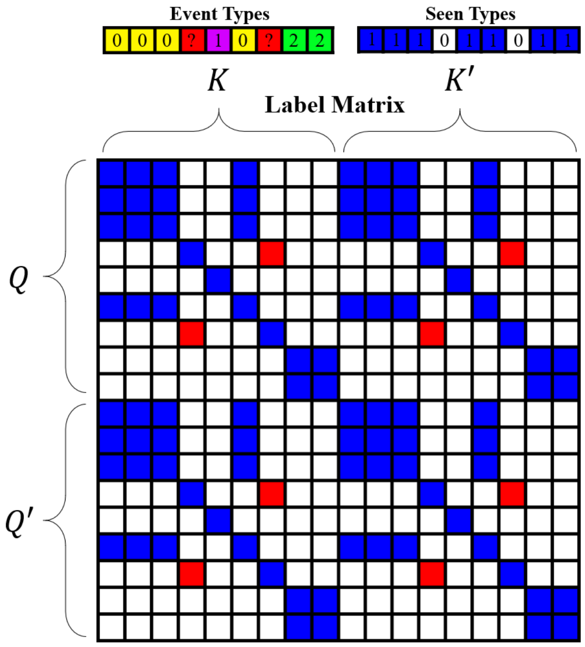


Figure 4: Best viewed in color. Visualization of the label matrix Y used in the loss. Blue is positive, white is negative, and red is masked. Note that the mask for the negatives less than the margin is not shown. The event types and corresponding "seen" boolean vector are also shown, and are used to construct the label matrix. Q and K are corresponding queries and keys to the labels, while Q' and K' are augmented data.

C Manual ACE05 to FrameNet Linking

ACE Type	Frame
Appeal	Appeal
Be-Born	Birth_scenario
Charge-Indict	Notification_of_charges
Convict	Verdict
Declare-Bankruptcy	Wealthiness
Demonstrate	Protest
Divorce	Personal_relationship
End-Org	Organization | Process_end
Extradite	Extradition
Fine	Fining
Injure	Cause_harm | Experience_bodily_harm
Marry	Forming_relationships
Nominate	Appointing
Phone-Write	Contacting
Release-Parole	Releasing_from_custody
Start-Org	Organization | Process_start
Start-Position	Being_employed | Process_start
Sue	Judgment_communication
Trial-Hearing	Trial
Pardon	Pardon
Merge-Org	Organization | Amalgamation
Acquit	Verdict
Execute	Execution
Attack	Attack
Transport	Transportation_status
Die	Death
Meet	Make_acquaintance | Meet_with | Come_together
Arrest-Jail	Arrest | Prison | Imprisonment | Being_incarcerated
Sentence	Sentencing
Transfer-Money	Commerce_money-transfer
Elect	Change_of_leadership | Choosing
Transfer-Ownership	Commerce_goods-transfer
End-Position	Being_employed | Process_end

Table 5: Mapping from ACE types to FrameNet frames. Some ACE types required multiple frames to be correctly mapped, which is indicated by " | ".

D Visualization

We visualize unseen event mentions using UMAP (McInnes et al., 2018) given a precomputed distance matrix of the cosine distance between Q and K. Following (Huang and Ji, 2020), we show the results on six unseen types in Figure 6. Sentence and convict overlap significantly, which makes intuitive sense as they are semantically very similar. Unlike (Huang and Ji, 2020), trial-hearing forms its own cluster.

---

[6] Note that there is a mistake in (Huang and Ji, 2020), since "sentence" is a seen type in (Huang et al., 2018)

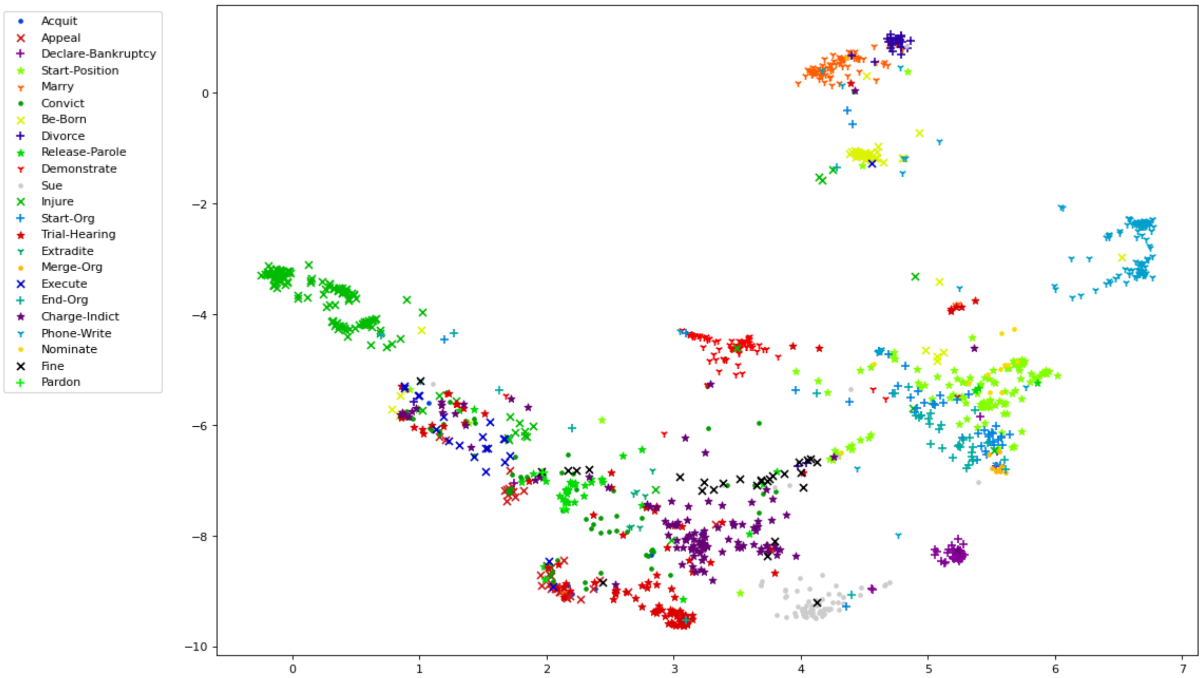


Figure 5: Visualization of all unseen types as seen by manifold approximation. Note that dimensionality reduction to 2D renders it difficult to understand with this high number of clusters, but the overall semantics of the space are interesting.

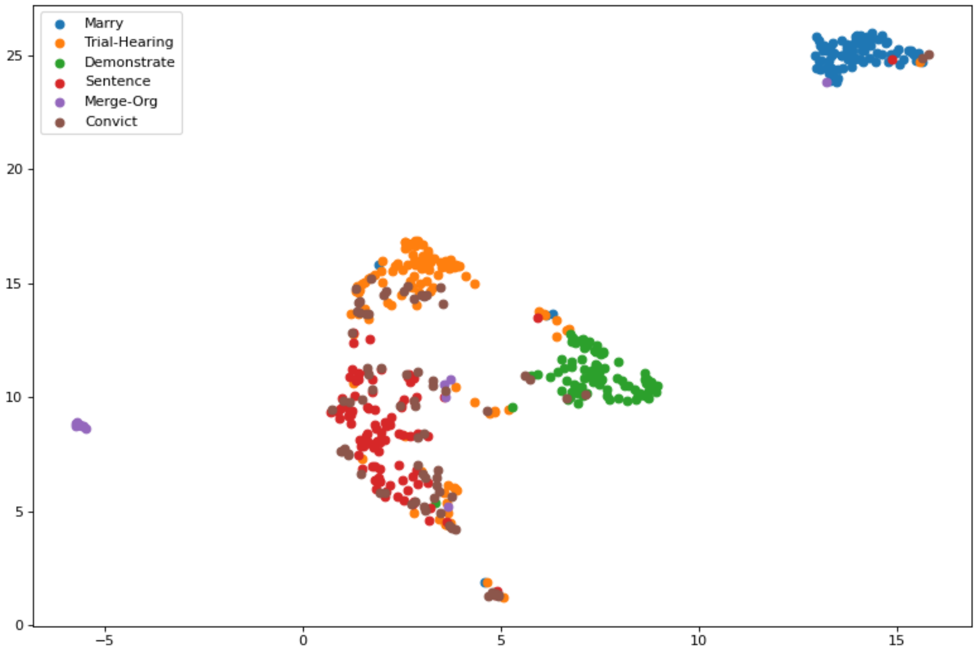


Figure 6: Visualization following (Huang and Ji, 2020) for one of the runs.[6] Note that our clusters have much less errors.

E Cluster Examples

We show extensive examples of our noted observations in Tables 6 and 7. Namely, start-org, merge-org, and end-org tend to overlap. Marry and divorce slightly overlap in the 23 cluster case. Most types of courtroom related events—Charge-Indict, Trial-Hearing, Convict, Release-Parole, Appeal, Execute, Acquit, Extradite—have some degree of overlap. There are "duplicates" when two or more events can occur in the same event mention. We also note the cluster of "junk" data, where the label isn't obvious from the event mention.

Cluster Type	Purity	Cluster Member Example Types and Inputs
Injure	98.3%	• **Injure:** According to other reports reaching here, five Syrian bus passengers were killed and 10 others were injured on Sunday morning when a US missile hit the bus they were traveling in near the Iraqi border • **Injure:** More than 40 were injured • **Injure:** There was no information on the identity of the injured person
Declare-Bankruptcy	95.0%	• **Declare-Bankruptcy:** You need to speak to a bankruptcy attorney pronto; this is a bankruptcy matter, not a tax matter • **Declare-Bankruptcy:** despite operating under bankruptcy laws, united posted the best on time performance • **Declare-Bankruptcy:** That means that he received the shares while he was still in bankruptcy, which means that the shares were potentially assets that the trustee could use to pay off creditors
Demonstrate	95.0%	• **Demonstrate:** The protest follows a string of others involving tens of thousands of peace activists across Japan since January • **Demonstrate:** No, I don't demonstrate against anybody during a war • **Demonstrate:** Several thousand demonstrators also gathered outside the White House in Washington, accompanied by a major security presence
Charge-Indict	64.4%	• **Charge-Indict:** 56-year-old forry drake has been charged with interstate transport of a minor • **Charge-Indict:** Ocalan, being tried in absentia, was indicted for entering the country illegally, a misdemeanor • **Convict and Charge-Indict:** convicted oklahoma city bombing conspirator terry nichols will stand trial again on state murder charges • **Appeal:** in the african nation of nigeria, an islamic court delayed the appeal of a woman condemned to death by stoning
Start-Position	64.4%	• **Start-Position:** Many Iraqis boycotted the meeting in opposition to U.S. plans to install Garner atop an interim administration • **Start-Position:** The meeting was Shalom's first encounter with an Arab counterpart since he took office as Israel's foreign minister on February 27 • **Start-Org:** Meeting in the biblical birthplace of the prophet Abraham, delegates from Iraq's many factions discussed the role of religion in the future government and ways to rebuild the country

Table 6: Examples of discovered clusters. Charge-Indict shows an example of a duplicate—an input with multiple event types. It also shows how courtroom related events can overlap. For Start-Position, there are some errors related to the Middle East, which occurs frequently in the Start-Position mentions.

Cluster Type	Purity	Cluster Member Example Types and Inputs
Marry	70.2%	• **Marry:** My wife and I were guests at a wedding on the Carnival Legend on New Years Eve 2003 • **Marry and Divorce:** Giuliani, 58, proposed to Nathan, a former nurse, during a November business trip to Paris - five months after he finalized his divorce from Donna Hanover after 20 years of marriage • **Phone-Write:** All the guests were folks who had met the bride and groom (an attractive young couple who were sailing alone) virtually on cruisecritic
Start-Org	34.7%	• **Start-Org:** Kiichiro Toyoda founded the automaker in 1937, transforming the loom manufacturer started by his father into an automaker • **Merge-Org:** I believe any neutral management consultant worth his or her salt would recommend a merger of the two organizations • **End-Org:** It's a dying organization, and this will be just the jolt it needs for another couple decades of somnambulant staggering before being ultimately replaced by far more efficient companies
Bad Data	-	• **Marry:** Either its bad or good • **End-Org:** i felt t7ire was something else too, much history behind silver cross to end is now • **Trial-Hearing:** Yeah, we're a pretty small town, so our newspaper covers it a lot
Phone-Write	86.2%	• **Phone-Write:** Let's see, my first call I got was from Russia • **Phone-Write:** I'm chewing gum and talking on the phone while writing this note • **Phone-Write:** He wants to call his mom in Houston
Sue	92.5%	• **Sue:** Buyers and sellers also would have to agree not to pursue further cases in foreign courts • **Sue:** The cost of class actions is factored into the cost of everything you buy • **Sue:** The average number of suits against a neurosurgeon is five in South Florida

Table 7: More examples of discovered clusters. Start-Org shows the semantic overlap between the organization-related clusters. Bad Data shows a cluster which mostly contains unclear input.

F Early Stopping

For this task, it is not reasonable to use a validation set for stopping. This is because the loss depends only on seen types and their relationships to unseen types. Since the unseen classes are unlabeled and the losses between pairs of unseen are unknown, the model can overfit to the seen data, pushing together clusters of unseen types. We partially address this issue by implementing a margin on negative values, which prevents the model from forcing together unseen clusters as strongly to separate them from seen type events. Because of this, our method requires strong pretrained representations to build on, which have luckily become common in recent years. To deal with this issue, we employ unsupervised clustering metrics to decide when to stop training. In particular, we use cosine distance-based silhouette scores to measure the quality of clustering. This increases the required compute up to 2x (in practice roughly 1.5x because back-propagation isn't required), but training is already relatively quick, with 10 or less epochs being sufficient. We note that this approach can have some variance. To address this issue, we employ a sliding window running average approximation to create a smooth curve of the initial increase and then decrease of the silhouette score. We consider a hybrid approach—we select the window with the highest silhouette score, and then we select the epoch with the highest silhouette score in that window as our stopping point, as shown in Figure 7.

G Evaluation Metrics

For the information retrieval metrics, given a list of rankings R,

$$MeanRank = \frac{1}{n}\sum_{i=1}^{n} R_i$$

$$MRR = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{R_i}$$

$$Hits@m = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{R_i \leq m}$$

G.1 Clustering Evaluation Metrics

Assume there are two clusterings: a set of (ground truth) classes C and a set of (predicted) clusters K. Each have N samples. Denote TP as true positives, the number of data point pairs that are in the same cluster in C and K. FP is the false positives, the number of data point pairs that belong in the same cluster in C but are not in K. FN is false negatives, the number of data point pairs that are in the same cluster K but not in the same ground truth cluster in C. TN is the number of data point pairs that are in different clusters in both C and K. scikit-learn (Pedregosa et al., 2011) is used to compute scores.

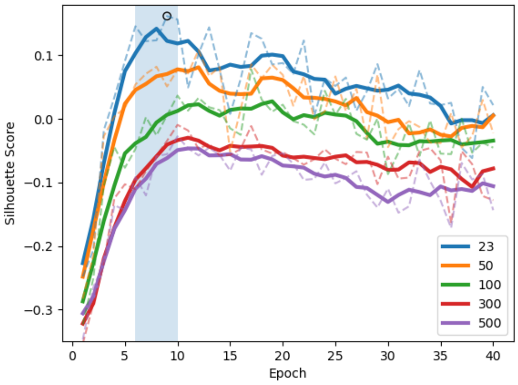


Figure 7: Bold lines are sliding window averages of size 5 over silhouette scores. Dotted lines are unsmoothed scores. Legend shows number of clusters. Note that the silhouette scores initially increase and then decay as overfitting occurs, resulting in the need for early stopping. Here, for 23 clusters, epoch 8 has the highest average score. The blue region shows the window around it, and epoch 9 (the black dot) is selected for stopping.

- **Geometric NMI** is the normalized mutual information between two cluster assignments. It is defined:

  $$NMI = \frac{I(C, K)}{mean(H(C), H(K))}$$

  where I is the mutual information and H is entropy. In this case, $mean$ is the geometric mean.

  $$mean(x_1, ..., x_n) = \left(\prod_{i=1}^{n} x_n\right)^{\frac{1}{n}}$$

  We note that arithmetic NMI using the arithmetic mean is often reported, but that it is equivalent to V-Measure.

- **Fowlkes Mallows** (Fowlkes and Mallows, 1983) is used to evaluate the similarity between a clustering and the ground truth. It is the geometric mean of pairwise precision and recall.

$$FM = \frac{TP}{\sqrt{(TP+FP)(TP+FN)}}$$

- **Completeness** (Rosenberg and Hirschberg, 2007) Completeness measures whether all of the data points assigned to a single class are assigned to a single cluster. It is defined:

  $$c = \begin{cases} 1 & \text{if } H(K,C) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & \text{else} \end{cases}$$

- **Homogeneity** (Rosenberg and Hirschberg, 2007) measures whether data points in a cluster are all assigned the the same class. It is symmetric to completeness:

  $$h = \begin{cases} 1 & \text{if } H(C,K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & \text{else} \end{cases}$$

- **V-Measure** (Rosenberg and Hirschberg, 2007) (standing for validity) is the harmonic mean between homogeneity and completeness:

  $$v = \frac{(1+\beta)hc}{\beta h + c}$$

  In practice, $\beta = 1$ is used to weight h and c equally.

- **Adjusted Rand Index** (Hubert and Arabie, 1985) is a version of the Rand index, a measure of cluster similarity, which is adjusted for chance.

  $$ARI = \frac{RI - \mathbb{E}[RI]}{\max RI - \mathbb{E}[RI]}$$

  where the Rand index, RI, is

  $$RI = \frac{TP + TN}{\binom{n}{2}}$$

  and $\mathbb{E}[RI]$ is expected value of random clusterings.

H UMAP Weights

UMAP (McInnes et al., 2018) attempts to estimate the density by comparing the distance to the k-nearest neighbors as follows:

$$\rho_i = min\{d(x_i, x_{i_j}) | 1 \le j \le k, d(x_i, x_{i_j}) > 0\}$$

$$\sum_{j=1}^{k} \exp(\frac{-max(0, d(x_i, x_{i_j}) - \rho_i)}{\sigma_i}) = \log_2(k)$$

Here, $d(x_i, x_{i_j})$ is the distance between x_i and x_{i_j}. ρ_i is the minimum distance to x_i's closest neighbor. σ_i, which smooths and normalizes the distances to the nearest neighbors, is calculated for each data point. Next, UMAP calculates the following weights between data points:

$$w((x_i, x_j)) = \exp(\frac{-max(0, d(x_i, x_{i_j}) - \rho_i)}{\sigma_i})$$

We use $1 - w((x_i, x_j))$ for agglomerative clustering.

I Reproducibility

The SBERT model we used, along with the size of the Q and K layers use a dimension of size 384. Our total model has 34,839,937 parameters, of which 1,479,937 do not belong to SBERT. Input uses the 'ldc_scope' part of the ACE event mention. Our model takes roughly 2 hours to train on one V100 GPU, including the early stopping calculations which are done with the model set to 'evaluation' mode. We used batch size 10, which is the most that would fit in memory. For learning rates, we considered the suggestions in (Devlin et al., 2019). Datasets used are in English. ACE05 contains 5,349 mentions which fall into 33 event types. Data artifacts in this work were used for research purposes consistent with their licensing agreements and intended use. Artifacts we create (e.g. code and manual linking) are in line with this intended use.

J Ablations

Ablation	**Method**		Clusters	Geometric NMI	Fowlkes Mallows	Completeness	Homogeneity	V-Measure	ARI
Original	Attn-Cosine	Agglo	23	46.40	34.60	49.82	43.24	46.27	26.69
	Attn-DotProduct	Agglo	23	50.17	37.48	53.50	47.06	50.06	30.13
	Attn	Manifold	23	54.83	42.77	55.00	54.67	54.82	38.74
	FT-SBERT	Manifold	23	60.28	50.63	60.19	60.37	60.28	47.24
	Attn-DotProduct	Affinity	49-68	56.87	35.64	49.58	65.26	56.33	30.02
	Attn-Cosine	Affinity	50-69	56.54	33.00	48.72	65.62	55.91	27.04
No Augmentation	Attn-Cosine	Agglo	23	47.69	33.21	49.57	45.88	47.66	26.43
	Attn-DotProduct	Agglo	23	48.14	34.02	49.73	46.59	48.11	27.53
	Attn	Manifold	23	48.72	37.91	49.99	47.49	48.71	33.11
	FT-SBERT	Manifold	23	57.04	45.61	57.20	56.89	57.04	41.86
	Attn-DotProduct	Affinity	53	51.84	30.32	45.41	59.19	51.39	25.00
	Attn-Cosine	Affinity	56	52.88	31.93	45.83	61.02	52.35	26.19
BERT Pooled Token Representation	Attn-Cosine	Agglo	23	29.27	19.71	32.04	26.74	29.15	10.36
	Attn-DotProduct	Agglo	23	27.81	21.30	33.37	23.17	27.35	8.31
	Attn	Manifold	23	28.07	16.89	28.06	28.08	28.07	10.88
	FT-BERT	Manifold	23	29.72	16.17	28.95	30.52	29.71	10.89
	Attn-DotProduct	Affinity	23	27.99	16.26	27.89	28.09	27.99	10.41
	Attn-Cosine	Affinity				DNC			
BERT Trigger Representation No Augmentation	Attn-Cosine	Agglo	23	52.61	36.98	74.74	37.04	49.53	18.12
	Attn-DotProduct	Agglo	23	45.17	33.33	71.03	28.72	40.91	12.64
	FT-BERT	Agglo	23	51.67	35.05	76.18	35.04	48.00	14.92
	Attn	Manifold	23	60.018	44.09	73.70	48.88	58.77	31.21
	FT-BERT	Manifold	23	81.20	73.27	82.96	79.48	81.19	71.05
	Attn-DotProduct	Affinity	58	61.07	37.14	27.89	69.05	60.61	24.83
	Attn-Cosine	Affinity	32	61.97	37.02	68.50	56.06	61.66	25.40
BERT Untrained	Untuned	Agglo	23	61.19	45.36	70.73	52.94	60.56	34.51
	Untuned	Manifold	23	75.49	64.61	77.69	73.34	75.46	61.54
RoBERTa-base Trigger Representation No Augmentation	Attn-Cosine	Agglo	23	49.37	33.48	69.54	35.05	46.61	13.57
	Attn-DotProduct	Agglo	23	43.62	31.33	66.27	28.71	40.06	10.24
	FT-RoBERTa	Agglo	23	57.32	36.68	75.69	43.41	55.18	18.17
	Attn	Manifold	23	57.14	44.30	69.72	46.84	56.03	33.01
	FT-RoBERTa	Manifold	23	83.44	78.12	84.37	82.52	83.44	76.56
	Attn-DotProduct	Affinity	44	57.76	39.86	64.06	52.08	57.45	29.17
	Attn-Cosine	Affinity	32	62.53	45.00	66.17	59.09	62.43	37.39
RoBERTa-base Untrained	Untuned	Agglo	23	17.53	23.66	39.62	7.76	12.98	0.35
	Untuned	Manifold	23	72.38	62.79	71.73	73.03	72.38	60.24
ELECTRA-base Trigger Representation No Augmentation	Attn-Cosine	Agglo	23	21.07	23.36	37.36	11.89	18.03	1.36
	Attn-DotProduct	Agglo	23	21.52	23.11	36.23	12.78	18.89	1.63
	FT-ELECTRA	Agglo	23	30.02	25.74	47.36	19.03	27.15	4.87
	Attn	Manifold	23	32.45	28.89	54.54	19.30	28.51	8.67
	FT-ELECTRA	Manifold	23	62.41	47.05	63.92	60.93	62.39	42.65
	Attn-DotProduct	Affinity	278	45.76	22.35	44.28	47.29	45.74	12.35
	Attn-Cosine	Affinity	154	45.83	23.86	46.88	44.81	45.82	12.32
ELECTRA-base Untrained	Untuned	Agglo	23	12.82	21.14	22.91	7.17	10.93	0.49
	Untuned	Manifold	23	30.31	20.18	33.08	27.79	30.20	12.08
ALBERT-base Trigger Representation No Augmentation	Attn-Cosine	Agglo	23	48.55	35.22	61.79	38.14	47.17	19.94
	Attn-DotProduct	Agglo	23	40.38	31.29	56.43	28.90	38.22	13.75
	FT-ALBERT	Agglo	23	38.41	27.64	59.44	24.82	35.02	6.34
	Attn	Manifold	23	56.80	43.26	70.72	45.61	55.46	30.69
	FT-ALBERT	Manifold	23	74.67	65.42	74.48	74.87	74.67	63.03
	Attn-DotProduct	Affinity	33	57.99	38.48	58.14	57.84	57.99	32.85
	Attn-Cosine	Affinity	40	60.27	34.81	60.01	60.53	60.27	26.71
ALBERT-base Untrained	Untuned	Agglo	23	47.00	31.21	54.09	40.83	46.53	21.29
	Untuned	Manifold	23	56.85	39.43	57.30	56.41	56.85	35.02

Table 8: New event type induction ablation results (%). The first 'Method' subcolumn is the input for clustering and the second is the clustering algorithm used. FT stands for finetuned. SBERT indicates the SBERT representations were used rather than our learned attentions (Attn). DNC indicates did not converge. Agglo is agglomerative clustering, Affinity is (Frey and Dueck, 2007), and Manifold is as described in Section 3.7.1. For affinity, each run can produce a slightly different number of clusters. BERT models use '*bert-base-uncased*'. The BERT pooled used a batch size of 8 because it takes up more VRAM. 15 epochs were used for each model. The BERT trigger representations use the ground truth triggers, which allows the duplicate problem to be avoided and helps clue in the network. The version trained with our method and without training is shown.

K Experiments on MAVEN

To further evaluate our method, we conduct experiments on the Maven dataset (Wang et al., 2020), which contains 118,732 event mentions and 168 event types. This provides an excellent example of a large dataset which may have a number of unannotated event types (or event types which need to converted to finer-grained typing). Since this semi-supervised task has not been done on MAVEN, we select the most common 150 types as seen and the remaining 18 types as unseen. To adapt early stopping to such a large dataset, we adapt our early stopping by calculating the silhouette score every 250 steps, which is roughly the same computation as a single ACE2005 epoch. We train our method for 8 epochs with other hyperparameters the same as the ACE2005 experiments. For Manifold clustering, we select an arbitrarily large $k = 3,000$ for computational reasons.

We experiment on using trigger representations from the event mentions. Unfortunately, these are specified by the dataset in terms of the its tokenization scheme, so converting to BERT tokenization incurs a small error rate (for example, some words are tokenized as 'unk' in MAVEN). We find that these errors do not prevent the representation from performing well.

Results are shown in Table 9. They show that SBERT representations do not work well for the mentions in the MAVEN dataset (we believe this is due to differences in the scope of what is considered an event mention). Training with our method still improves clustering on the SBERT representations from 9.06 ARI to 10.51. However, the attention mechanisms learned are less effective along with our manifold clustering approach. The *bert-base-uncased* representations using the trigger words perform much better as expected. Additionally, our method, using both learned attention and trained trigger representations, increases performance significantly. For manifold, this is from 44.84 ARI to as high as 67.01. As noted in the conclusion, the interaction here between manifold approximation and large language model representations is an interesting future research direction for better understanding these results. It may also be interesting to think about how to obtain a desirable representation for starting the training of our model. Another example that would benefit from this is using a neural dense retriever to retrieve frozen embeddings in a retrieval large language model, such as (Borgeaud et al., 2021).

Representation	Method		Clusters	Geometric NMI	Fowlkes Mallows	Completeness	Homogeneity	V-Measure	ARI
SBERT	Untuned	Agglo	18	33.33	18.98	37.25	29.81	33.12	9.06
SBERT	Untuned	Manifold	18	37.07	19.61	37.82	36.33	37.06	13.14
bert-base-uncased	Untuned	Agglo	18	66.95	42.46	75.14	59.66	66.51	32.47
bert-base-uncased	Untuned	Manifold	18	69.05	48.71	69.54	68.56	69.05	44.83
SBERT	Attn-Cosine	Agglo	18	30.14	14.08	30.82	29.48	30.13	6.86
SBERT	Attn-DotProduct	Agglo	18	29.92	13.90	30.06	29.77	29.92	7.49
SBERT	Finetuned	Agglo	18	34.23	16.48	34.05	34.41	34.22	10.51
SBERT	Attn-Cosine	Manifold	18	32.54	14.91	32.60	32.48	32.54	8.68
SBERT	Finetuned	Manifold	18	33.61	16.26	34.10	33.13	33.61	9.67
bert-base-uncased	Attn-Cosine	Agglo	18	73.77	60.25	75.01	72.56	73.76	57.11
bert-base-uncased	Attn-DotProduct	Agglo	18	73.52	58.19	73.84	73.21	73.52	55.07
bert-base-uncased	Finetuned	Agglo	18	74.54	59.67	78.41	70.86	74.44	55.30
bert-base-uncased	Attn-Cosine	Manifold	18	76.48	64.08	76.82	76.14	76.48	61.43
bert-base-uncased	Finetuned	Manifold	18	**79.93**	**69.49**	**81.35**	**78.54**	**79.92**	**67.01**
bert-base-uncased	Attn-DotProduct	Affinity	37	74.47	49.70	66.07	83.95	73.94	43.79
bert-base-uncased	Attn-Cosine	Affinity	41	74.37	48.54	65.20	84.83	73.73	42.14
SBERT	Attn-DotProduct	Affinity	40	40.87	11.53	35.48	47.08	40.47	6.81
SBERT	Attn-Cosine	Affinity	38	40.46	11.52	35.39	46.26	40.10	6.79

Table 9: New event type induction results (%) on MAVEN (Wang et al., 2020). The first two sections are baseline representations without our method. The *bert-base-uncased* representation uses trigger representations. The first subcolumn of ‘method’ is the input for clustering and the second is the clustering algorithm used. ‘Finetuned’ indicates the finetuned model representations from our method were used and ‘untuned’ indicates our method was not used. Attn-score indicates our learned attention scores were used and the method for computing these scores from the query and key vectors. E stands for ensemble. Agglo is agglomerative clustering, Affinity is (Frey and Dueck, 2007), and Manifold is as described in Section 3.7.1. For affinity, each run can produce a slightly different number of clusters.

Multilingual Content Moderation: A Case Study on Reddit

Meng Ye[1], Karan Sikka[1], Katherine Atwell[2], Sabit Hassan[2]
Ajay Divakaran[1], Malihe Alikhani[2]
[1]Center for Vision Technologies, SRI International
[2]Department of Computer Science, University of Pittsburgh
`{first.last}@sri.com`, `{kaa139, sah259, malihe}@pitt.edu`

Abstract

Content moderation is the process of flagging content based on pre-defined platform rules. There has been a growing need for AI moderators to safeguard users as well as protect the mental health of human moderators from traumatic content. While prior works have focused on identifying hateful/offensive language, they are not adequate for meeting the challenges of content moderation since 1) moderation decisions are based on violation of rules, which subsumes detection of offensive speech, and 2) such rules often differ across communities which entails an adaptive solution. We propose to study the challenges of content moderation by introducing a multilingual dataset of **1.8 Million** Reddit comments spanning **56** subreddits in English, German, Spanish and French[1]. We perform extensive experimental analysis to highlight the underlying challenges and suggest related research problems such as cross-lingual transfer, learning under label noise (human biases), transfer of moderation models, and predicting the violated rule. Our dataset and analysis can help better prepare for the challenges and opportunities of auto moderation.

1 Introduction

Being able to moderate user-generated content is critical for online social media platforms. Several platforms employ human moderators to monitor user content to prevent the spread of misinformation, adverse effects of hateful speech, fraud, etc (Geiger and Ribes, 2010; Dosono and Semaan, 2019; Wang et al., 2022; Jhaver et al., 2017). The moderators' task is to remove improper content and/or suspend users posting such content. However, reviewing and moderating each user comment is practically infeasible due to limited resources, especially during time-critical and large-scale events. More importantly, such moderation work can cause damage to moderators' mental health due to burnout from volunteer work and exposure to harmful content (Gillespie, 2020; Roberts, 2014; Dosono and Semaan, 2019; Wohn, 2019).

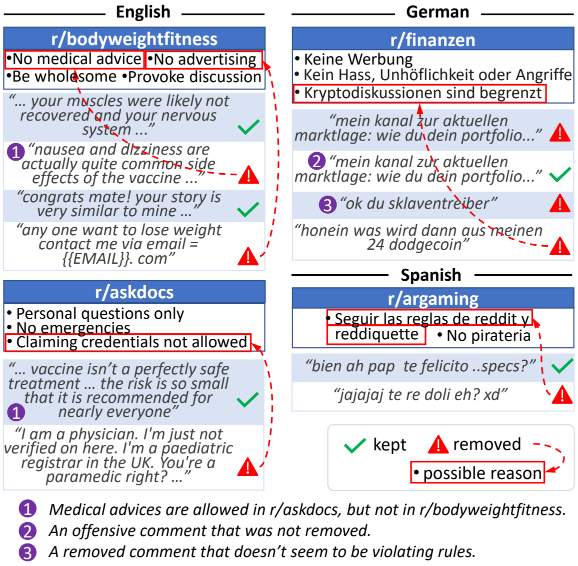


Figure 1: Content moderation on Reddit is challenging as it 1) depends on rules of each community, 2) requires contextualized understanding of the comments, and 3) is affected by moderator biases.

In recent years, researchers have put effort into collecting hate speech or offensive language datasets from social platforms such as Twitter (Zampieri et al., 2019; Gautam et al., 2020; Golbeck et al., 2017), YouTube (Dinakar et al., 2011), or a mixture of those platforms (Kennedy et al., 2017; Bhattacharya et al., 2020). These datasets contain annotations of hateful speech and are used to train NLP systems to remove harmful comments (Pamungkas et al., 2020; Chung et al., 2019; Ranasinghe and Zampieri, 2021). We refer to this problem as Offensive Language Identification (OLI). Instead of simply removing such content or suspending users, some works propose countering hate speech with discourse-aware style

---

[1] `https://github.com/mye1225/multilingual_content_mod.git`

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3828–3844
May 2-6, 2023 ©2023 Association for Computational Linguistics

transfer (Atwell et al., 2022), expert-based counter-narratives (Chung et al., 2019), by learning to intervene with conversational context (Qian et al., 2019), or by proposing a holistic conceptual framework (Chaudhary et al., 2021).

However, despite remarkable progress in OLI, it is not sufficient to tackle content moderation for two key reasons: (1) OLI is a subset of content moderation, as the latter involves flagging content that is not only hateful but also violates the rules of the platform. For example, Reddit has site-wise rules that not only requires users to be civil, but also prohibits actions such as posting illegal content, spamming, and revealing personal information[2]. (2) Prohibiting offensive/hateful speech is universal to all communities (Chandrasekharan et al., 2018), while content moderation requires a system to be adaptive to rules that change dynamically across different communities. It is often the case that content that is allowed in one subreddit might be disallowed in another. For example, it is allowed to post medical advice on `r/askdocs` while it is prohibited on `r/bodyweightfitness`. We thus believe it is useful to study Reddit comments from a moderation perspective as we can collect data from multiple communities and measure the effectiveness of such systems in adapting to community guidelines.

In this work, we aim to bridge this gap by introducing a dataset of Reddit comments collected from 56 communities for studying content moderation. We propose a challenge for content moderation and demonstrate the limitations of the state-of-the-art models. Our dataset will be available to researchers who agree to the terms and conditions of our data-sharing policy approved by the ethical committee of our institution. Our contributions are:

- We propose a multilingual dataset for content moderation. The dataset consists of **1.8 million** comments from **56** subreddits with moderation labels, that specify whether a comment was kept or removed by the moderator. We also provide the meta-data for each subreddit that includes its name, description, and rules.

- We show that existing offensive speech datasets are not suitable for the content moderation task because only a small portion of the removed comments are offensive. As such, the models trained on OLI datasets fail to identify removed comments that are non-offensive.

- We study the performance of models trained on moderation data under different cross-lingual settings including multilingual-language model, translate-train, and translate-test.

- We provide insights on what makes content moderation a challenging task and discuss potential research problems that can be explored with our proposed dataset.

Dataset	Platform	Type	Size	Language
OLID	T	O&H	14K	en
SWAD	T	SW	1.5K	en
OffensEval	T,R,F	O&H	9M	en, da, tr, ar, el
HatEval	T	O&H	19.6K	en, es
CDUC	Y	CYB	4.6K	en
CONAN	T	O&H	15K	en, fr, it
Norms	R	MOD	4M	en
Ours	R	MOD	1.8M	en, de, es, fr

Table 1: Comparison with published datasets. OLID (Zampieri et al., 2019). SWAD (Pamungkas et al., 2020). OffensEval-2020 (Zampieri et al., 2020). HatEval (Basile et al., 2019). CONAN (Chung et al., 2019). CDUC (Dadvar et al., 2013). Norms (Chandrasekharan et al., 2018). Platforms: (T)witter, (F)acebook, (R)eddit, (Y)outube. Types: (O)ffensive and (H)ate speech, (SW)earing, (CYB)er Bullying, (MOD)eration.

2 Related Work

2.1 Offensive Language and Hate Speech Datasets

Many existing works that have collected user comments from online social platforms. Zampieri et al. (2019) proposed Offensive Language Identification Dataset (OLID), which models not only different types of offensive language, but also the target of offensive messages in a hierarchical structure. The Swear Words Abusiveness Dataset (SWAD) developed by Pamungkas et al. (2020) is a collection of tweets selected from OLID that focuses on predicting abusiveness of a swear word in a tweet context. Chung et al. (2019) created a multilingual dataset with hate speech/counter-narrative pairs provided by experts. The idea is to fight online hate speech content with informed textual responses instead of the standard method of removing content or suspending users. OffensEval (Zampieri et al., 2020) is an offensive language identification challenge that has attracted multiple research teams. It is

---

[2] `https://www.redditinc.com/policies/content-policy`

based on the same hierarchical three-level annotations from the aforementioned OLID dataset in multiple languages. CAD (Vidgen et al., 2021) is a recent recently proposed dataset of Reddit posts and comments with manually annotated two-level abusive categories. A brief comparison between these datasets and ours is shown in Table 1.

All the mentioned works focus on detecting offensive speech. However, online content moderation involves detection of comments that violate community rules in addition to those that are offensive. For example, moderators will often remove comments that are self-promoting, spamming, or off-topic because they do not provide useful information and are harmful for the communication environment (e.g., *"I can help you, see my youtube channel"*). In both of these cases, a model trained to detect hate speech will likely fail.

2.2 Reddit Rules and Content Moderation

A pre-existing related work by Chandrasekharan et al. (2018) trained 100 subreddit classifiers on removed/un-removed comments, and used clustering analysis to discover three types of *implicit norms* from the removed comments: macro norms that are universal to all subreddits (e.g., hate speech, personal attacks), meso norms that applies to subgroups (e.g., meme responses), and micro norms that are specific to individual subreddits (e.g., offering commerce tips). Our study is different in that we focus on moderation task that depends on *explicit community rules* written by moderators. Another work (Samory, 2021) studied the problem of identifying comments *approved* by moderators. They found that approved comments and removed ones actually share many traits such as toxicity and insults, and that it is hard to distinguish them. Fiesler et al. (2018) studied rules from $1,000$ subreddits and found that rules are highly dependent on the context of each individual subreddit while sharing common characteristics across the platform.

One key limitation of these studies is that they only focused on subreddits in the English language and discarded all non-English subreddits. This could be because most data on Reddit is in English. In our work we also select subreddits in non-English languages, i.e. German, French, and Spanish. We then study the possibility of transferring a moderation model trained on subreddits in English (which generally has more content). We argue that a good AI moderator should not only focus on data in English language, but also leverage that knowledge to improve performance on other low-resource languages. While Hassan et al. (2022) study *human* moderation bias across different languages and cultures, our work focuses on the problem of *automated* moderation.

3 Dataset

3.1 Subreddit selection

We collected data from 56 subreddits[3] based on the following criteria:

Wide range of topics: Generally the popularity (and thus data points) of a subreddit depends on its topic. Being able to cover many topics will enable us to better estimate the generalizability of machine learning models on this task. The topics include news, politics, finance, sports, electronics, etc.

Subreddits on similar topics: We chose subreddits with similar topics to enable inquiry into questions relating to transferability of moderation models. For example, "Do models trained on one subreddit transfer to other subreddits?", "will models transfer better between subreddits on similar topics?", and "which subreddit should I train the model on so that it could also be better adapted to X?". Some example of similar subreddits are `r/news` and `r/worldnews`, `r/finance` and `r/personalfinance`, `r/anime` and `r/naruto`, `r/games` and `r/xboxseriesx`.

Multilingual data: Most groups on Reddit are in English, but there are also some in other languages such as Spanish, German, French, Arabic. To extend moderation models to multilingual settings we also selected a number of non-English subreddits. For example French (`r/quebec`, `r/france`, `r/moi_dlvv`), German (`r/de`, `r/finanzen`, `r/ich_iel`), and Spanish (`r/argentina`, `r/gaming`).

3.2 Data Collection Pipeline

We built our data collection pipeline based on the Python Reddit API Wrapper (PRAW) [4], which streams comments and submissions in real time from multiple subreddits. For each data record we store important fields such as the ID, author, posting time, and comment body. We follow a two step approach where we first scrape data for a week, and then check if the comment/submission has been removed or retained by the moderator or if it has

---

[3] A full list of all 56 subreddits can be found in Table 8.

[4] `https://github.com/praw-dev/praw`

split	language	#sub	#comments	removal
		Training data		
en-train	English	48	$1,347,611$	1.87%
en-val	English	48	$74,885$	1.83%
		Test data		
de	German	3	$177,046$	0.86%
es	Spanish	2	$95,586$	0.66%
en	English	48	$74,893$	1.85%
fr	French	3	$49,780$	0.23%

Table 2: Number of subreddits, comments, and percentage of removed comments in each data split.

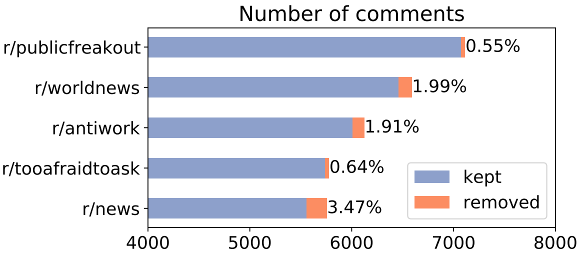


Figure 2: Number of kept and removed comments in the top 5 subreddits (% removed comments in also shown).

been deleted by the author. We use this two-step approach since once the comment is removed/deleted it does not retain its original content. Figure 5 in the appendix shows the overall procedure of our data collection pipeline.

3.3 Dataset Overview

We use the data collection pipeline to collect ~ 1.8 Million comments in a time span of three weeks.

For benchmarking, we split all of the English data randomly into $90\%/5\%/5\%$ as train, validation, and test subsets. We chose all the non-English data to be part of the test set to study the cross-lingual transferability of models trained on data in English. Table 2 lists the number of subreddits, total number of comments, and the percentage of removed comments for all the data splits.

English data makes up a large proportion of the entire dataset, and is collected from 48 subreddits on different topics. For non-English languages, there are significantly fewer subreddits and we choose the most popular ones for each language. This dataset is highly imbalanced in that only a small proportion of the comments are removed by the moderators: English data has a removal rate of around 1.8% and for German, Spanish, and French it is 0.86%, 0.66%, and 0.23%, respectively. This makes the task of identifying moderated comments very challenging. For comparison, the proportion of offensive examples is around 33% in OLID, $12.5\% \sim 28.4\%$ across different languages in OffensEval-2020 (Zampieri et al., 2020), and $28.4\% \sim 50.0\%$ in HASOC-2020 (Mandl et al., 2021). We argue that the high percentage of offensive examples in those datasets makes them less suitable for real world applications because these offensive posts make up only a small share of moderated content.

We also observe that different subreddits have different rates of removal, from the lowest of 0.0% to the highest of 21.74%. These differences are likely explained by the discrepancies in topic and subreddit rules, as well as the level to which the moderators enforce these rules. However, the overall rate of removal is quite low (1.84%) compared to existing datasets mentioned above. We show the statistics of the top 5 active subreddits in Figure 2 (refer to Figure 6 in the appendix for more details).

Each subreddit is associated with a set of rules that are enforced by the moderators, which are normally displayed in the *About* section. Typically, each rule has a short title and a longer description to explain in more detail the types of content that are prohibited. For each subreddit, we have collected all of its rules including titles and descriptions (see Figure 7 in the Appendix for the distribution of the number of rules in our dataset).

3.4 Manual Analysis of Offensiveness

To better understand the level of offensiveness in the content moderation task, we manually annotated $1,238$ comments (around 200 removed and 100 unremoved examples for each of the four languages) using the fine-grained taxonomy of offensiveness presented in Mubarak et al. (2022), with the addition of the categories of sexuality and age. The distribution of comments for different categories is shown in Figure 3. We observed that most of the comments (71.86% of removed and 80.115% of non-removed) are not offensive. Many of these non-offensive comments criticized the corresponding subreddit or disagreed with the views or goals of the subreddit. For instance, the comment *"i was disappointed when i got the halo console..."* was removed from the `r/halo` subreddit. The comment is not offensive but could have been removed due to its criticism of a product supported by the `r/halo` subreddit. Since these rules, views and goals can vary significantly across subreddits, it explains why content moderation is hard for machine learning classifiers and our moderation classifiers achieve lower scores compared to offensiveness

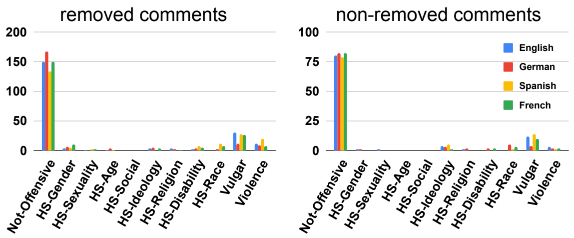


Figure 3: Distribution of different types of offensive speech for Reddit comments.

classifiers. We also see that the distributions of different types of offensive speech are similar across both removed and non-removed comments as well as the four different languages. To conclude, moderating online content is a very different task compared to identifying offensive language and hate speech and a new dataset is needed to train models and study automated content moderation.

4 Experiments

We study the task of content moderation on Reddit by formulating it as a binary classification task: given a user comment x predict whether it should be removed by moderators using $y = f(x)$, where f is the classifier, and y is the removal probability. We set the binary labels based on whether the comment was removed or retained by the moderator (Section 3.2). We report both AUC (area under the ROC curve) and F1 scores.

4.1 Evaluation with Existing Offensive Language Identification Models

A naive solution to moderation is to cast this problem as offensive language detection and directly use classifiers trained on OLI datasets. We thus evaluate models trained on OLI datasets for moderation. This experiment will help in answering two questions: 1) can these models recognize offensive comments in the context of Reddit content moderation?, and 2) do these models miss comments that should be removed but are not-offensive? Being able to answer these question will help us understand whether content moderation can be solved by using OLI or it is a necessity to collect data tailored for moderation. We investigate the above question using three models: RoBERTa (base) (Liu et al., 2019) finetuned on the HatEval (Basile et al., 2019) and OLID dataset (Zampieri et al., 2020), and an XLM-RoBERTa model fine-tuned on OLID. We compute the moderation score for each comment by averaging the output probabilities from these models.

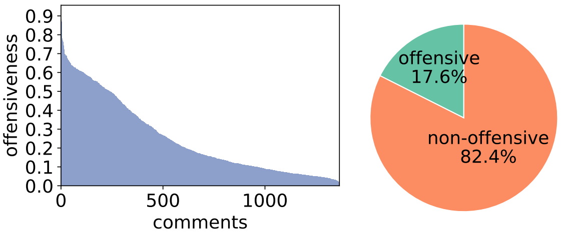


Figure 4: Distribution of offensiveness scores for removed comments (left plot shows sorted scores). Most of the removed comments are not very offensive. Only 18% of them get offensive scores higher than 0.5.

Model	Tr data	T-ori	R-off	R-mod
RoBERTa	HatEval	48.72	44.12	50.59
RoBERTa	OLID	81.25	79.20	47.76
XLM-RoBERTa	OLID	75.81	75.82	48.45

Table 3: Offensive language/hate speech classifiers with their original training (Tr) Twitter dataset and macro-F1 scores on the test split of the same dataset (T-ori), a subset of our Reddit data with offensive labels (R-off), and our Reddit data with moderation labels (R-mod).

Figure 4 shows the distribution of offensiveness scores of the removed comments. The left and the right plot show the sorted scores and the percentage of offensive and non-offensive comments using a threshold of 0.5, respectively. We observe that most of the removed comments (82.4%) are not offensive, and also note that this number matches with the manual analysis in Section 3.4.

By qualitatively studying these examples we observe that 1) OLI models can detect offensiveness in the comments, and 2) not all removed comments are offensive, and OLI models tend to miss these. For instance, a comment stating *"try contacting drewcybersupport on instagram about ..."* from `r/minecraft` was likely removed due to self-promotion (the average offensiveness of this comment is 0.01). More examples can be found in Table 9. We also evaluate the performance of these models on three datasets: the original offensive datasets that they were fine-tuned on (T-ori), a subset of our collected Reddit data with manual offensiveness annotations (R-off), and our proposed moderation dataset (R-mod). From the results in Table 3 we can observe that: 1) these models achieve good performance[5] on both Twitter and Reddit data when the task is to identify offensive language, and

---

[5] The model fine-tuned on the HatEval dataset has a low F1 score possibly because it focused on hate speech targeted at immigrants and women instead of generic OLI.

Setting	Language	Encoder	AUC				F1-macro			
			en	de	fr	es	en	de	fr	es
MLLM	multilingual	XLM-RoBERTa	66.83	69.42	64.25	63.90	49.60	49.78	49.94	49.83
Translate-train	German	bert-base-german-uncased	60.33	63.34	-	-	49.53	49.78	-	-
	French	flaubert-base-uncased	53.47	-	49.96	-	49.53	-	49.94	-
	Spanish	bert-base-spanish-uncased	64.16	-	-	63.29	49.58	-	-	49.99
Translate-test	English	roberta-base	67.38	71.23	71.61	64.33	49.66	50.01	50.36	50.22

Table 4: Experimental results on moderation classifier under three settings: 1) MLLM: multilingual language model embeddings, 2) translate-train, and 3) translate-test.

Setting	Language	Encoder	AUC				F1-macro			
			en	de	fr	es	en	de	fr	es
MLLM	multilingual	XLM-RoBERTa	66.14	70.10	68.32	59.81	59.97	64.46	61.15	56.78
Translate-train	German	bert-base-german-uncased	61.10	65.90	-	-	57.24	61.15	-	-
	French	flaubert-base-uncased	53.64	-	53.71	-	51.99	-	55.10	-
	Spanish	bert-base-spanish-uncased	63.08	-	-	60.52	59.33	-	-	56.87
Translate-test	English	roberta-base	66.94	70.66	71.57	61.96	61.96	64.16	64.87	58.11

Table 5: Experimental results with balanced data splits from our moderation dataset.

2) their performance drops significantly when evaluated on the moderation task.

Based on the results from both our qualitative and quantitative analyses, we conclude that there is a significant mismatch between the goals of content moderation and offensive language identification. We thus believe that it is necessary to have a moderation-specific dataset to train classifiers for content moderation.

4.2 Evaluation with Models Trained on Proposed Dataset

We use our collected data with moderation labels to train the classifier f. We use pre-trained transformer based language models as text encoders and add a shallow classifier on top: $y = f_m(f_{enc}(x))$, where f_{enc} is the encoder and f_m is the moderation classifier. To deal with multilingual data, we need to either use a multilingual encoder, that can work with inputs in any language (MLLM), or use a monolingual encoder with machine translation models[6]. We only transfer models trained on English data to non-English data and not the other way since the number of data points in English are much larger and it could be beneficial to leverage it for other languages with less data available.

We summarize our experimental results[7] in Table 11 and observe that the RoBERTa (67.38) and XLM-RoBERTa (66.83) models outperform offensive language detectors (second model in Table 3 achieved 61.11 on AUC) on the English data. This is expected because the moderation training data contains removed comments that are non-offensive, and thus the trained classifiers are able to capture these and make better predictions. We also observe that translate-train performs lower than translate-test in most cases, possibly due to the fact that noise is often introduced during translation and the negative impact is more severe in the training stage than in the test stage. We note that although the best performance on most of the test splits is achieved by translate-test, MLLM performance is close to the best in most cases. Considering the challenges of maintaining an extra translation model for every language in real-world applications, MLLM might be a better solution towards building multilingual auto moderators.

We would like to note that the (macro) F1 scores of these models are close to 50 since the classifier/predictions are overwhelmed by data imbalance (less than 2% of comments are labeled to be removed). We thus additionally report AUC scores since, being a ranking based metric, it is not affected by data imbalance. For comparison, we also propose additional data splits[8] with balanced classes and repeat the same experiments. As shown in Table 12, the performance trends for the balanced set are similar to the original splits, and the F1 scores are clearly above 50, ranging from 55.10 to 64.46 with one exception of 51.99, which is

---

[6] Following Artetxe et al. (2020), we use two settings with machine translation: 1) *Translate-test*: a classifier is trained on English data, and each non-English language needs to be translated into English during test, and 2) *Translate-train*: translate the English data into each target non-English language, and then do both training and test in that target language. For translation we used models from Tiedemann and Thottingal (2020)

[7] Averaged over 3 runs with different random seeds.

[8] Statistics of the balanced splits can be found in Table 10.

Group	Examples	GT/f_o/f_m	freq.
1)	*"every f****** one of those dogs ought to be put down.", "my god this sub is a perpetual b***h fest. clearly time to unsub"*	Y/Y/Y	7.35%
2)	*"prove it by sending me $5 000.00. that would be nothing for you... so go ahead and do it ...", "how do you explain exploding infections and new variants coming out of south africa where 90% of the population is vaccinated?"*	Y/N/Y	16.95%
3)	*"f**k {{name}} and his shorts.", "she has such a punchable face"*	Y/Y/N	1.85%
4)	*"was....was this written by a cat", "depends on the market. {{organization}} sold more units than switch last month in the uk."*	Y/N/N	23.85%
5)	*"oh my god you're pathetic lmao", "well you are a moron so i guess it could be worse"*	N/Y/Y	2.15%

Table 6: Different groups of comments from the English test data. 'GT', f_o, and f_m are ground truth label, offensiveness score and moderation score, respectively. "Y" stands for offensive/removed and "N" stands for non-offensive/non-removed. "freq." denote the proportion of each group in the whole subset (2000 samples).

likely caused by low translation quality in French.

4.3 Analysis on Error Types

We performed additional analysis to understand the errors made by the learned classifier by randomly sampling $1,000$ examples from both removed and non-removed comments and computing their offensiveness[9] (f_o) and moderation scores (f_m). We then organize all comments into groups based on their ground truth label and classifier predictions. Table 6 shows five of those groups and their frequencies: 1)There are 7.35% examples which are offensive and both f_o and f_m were able to identify them. 2) There are 16.95% examples on which f_o failed due to their low offensiveness while f_m was able to identify them successfully. 3) f_m missed a small portion of examples (1.85%) that are offensive and should be removed (*model-error*). 4) There is a significant portion of data points (23.85%) that were removed by moderators while both classifiers predicted them to be fine (*model-error*). This group represents the majority of removed comments that have violated rules other than the use offensive language. 5) Both classifiers do not agree with the ground truth label on 2.15% of the comments, most of them are highly offensive and should be removed but were actually approved by moderators. These instances indicate noisy labels in our dataset that are possibly caused by moderators' biases (*human-error*). Based on these observations we conclude that offensiveness classifiers and moderation classifiers have different types of errors and we believe that there is scope for improvement by using models that: 1) can incorporate knowledge from both OLI and moderation tasks and 2) are robust to label noise.

Subreddit Group	Model	AUC	F1
`r/naruto`	M-BERT	51.59	47.43
	XLM-Ro	64.48	37.84
`r/naruto` + `r/anime`	M-BERT	55.78	54.13
	XLM-Ro	31.08	46.39
`r/judaism`	M-BERT	62.08	60.65
	XLM-Ro	50.23	33.77
`r/judaism` + `r/islam`	M-BERT	68.27	63.89
	XLM-Ro	62.79	33.91
`r/judaism` + `r/islam` + `r/christianity`	M-BERT	67.89	62.33
	XLM-Ro	49.27	34.50
`r/feminism`	M-BERT	71.82	64.05
	XLM-Ro	51.27	38.22
`r/feminism` + `r/lgbt`	M-BERT	73.25	67.26
	XLM-Ro	49.67	34.10
`r/feminism` + `r/lgbt` + `r/racism`	M-BERT	72.95	66.16
	XLM-Ro	60.78	33.80

Table 7: Performance of classifiers within groups of subreddits. M-BERT refers to bert-base-multilingual-cased and XLM-Ro refers to XLM-RoBERTa-base.

4.4 Manual Analysis of Rule Violations

We also manually labeled 145 removed comments from the `r/stock` subreddit with the help of 3 annotators. The annotators were instructed[10] to select one of the five rules a removed comment violated. We decided to drop class 4 due to a small number of samples. We also dropped class 5 as this class corresponded to the "not sure" class. We selected examples where annotators had the most agreement and this resulted in 111 comments across 3 classes with 36, 54, and 21 comments for the non-civil, missing context, and spam/self-promotion categories (respectively). We performed five-fold cross-validation strategy for our experiments by training a logistic regression classifier on textual features extracted using sentence-transformers (all-mpnet-base-v2) (Reimers and Gurevych, 2019) We obtained an AUC of 91.8 ± 1.8, and the F1 scores for individual classes are 80.56 (Non-civil),

---

[9] Average offensiveness score from classifiers in Section 4.1

[10] The guide provided to the annotators is shown in Table 14

77.47 (Missing context), and 61.53 (Spam/self-promotion). The class-wise numbers match our intuition that examples from the "non-civil" class performs the best as it mostly includes offensive comments. This experiment shows that it is possible to learn to predict which rule was violated and thus provide some explanation to the user.

4.5 Experiments with Partitioning Data

As observed earlier, the AUC and F1-macro scores for moderation can be relatively low due to the task's complexity. We hypothesize that grouping similar subreddits together may improve the results within the groups. To verify this, we manually selected three groups of subreddits with similar topics and sample both training and test sets based on the grouping. From Table 7, we observe that grouping subreddits together may result in improved performance. For example, grouping `r/lgbt` and `r/feminism` together achieves 73.25 AUC, an improvement of 1.43 over `r/feminism` alone, and an improvement of $6 \sim 7$ compared to Table 12. However, we also observe that grouping more than two subreddits can lead to a drop in performance. For instance, adding `r/racism` to the group of `r/lgbt` and `r/feminism` decreases the AUC slightly to 72.95. This suggest that splitting the moderation task across multiple classifiers could lead to better performance with proper partitioning.

4.6 Insights on Moderation Dataset and Task

Based on the qualitative and quantitative evaluations, we summarize our insights:

Moderation is a highly imbalanced task. Unlike existing offensive language datasets, which may have $\geq 30\%$ positive examples (Mandl et al., 2021; Zampieri et al., 2019), the proportion of removed comments in our moderation dataset is $\leq 2\%$ This is challenging for most classifiers as their prediction will be biased by the majority class (see Table 11). We believe that this challenge makes our dataset distinct from others because it exposes models to real world scenarios.

Moderation labels are noisy. The moderation labels in our dataset are noisy as we found that some highly offensive comments were not removed and some benign comments were removed by the moderators. There could be multiple reasons for this such as human errors, individual biases. We believe this label noise is a characteristic of the task and researchers will be required to design algorithms with this in mind.

Moderation requires more than offensiveness detection. Our experiments reveal that pre-existing datasets with hate speech or offensive language labels are not sufficient for moderation task. This task involves referencing a set of guidelines which include not only being civil, but also other community-specific rules such as no off-topic discussions, no self-promotions, and no low-effort comments. Models trained only on offensive language datasets are able to identify non-civil content but fail to detect other cases of rule violation.

Moderation systems need to be adaptive. As Reddit communities are self-organized and moderators can make their own rules, many subreddits have very specific rules that may not be common with other subreddits. For example, "keep it halo" (`r/halo`), no "Asking for handouts or transactions" (`r/personalfinance`), "Research must be less than 6 months old" (`r/science`), and "No medical advice" (`r/bodyweightfitness`). A classifier trained on a large dataset like ours could possibly capture general rules and macro-norms, but is unlikely to transfer to novel communities with very specific rules. We thus argue that an adaptive and dynamic system that could condition its decisions on given guidelines is needed for successful auto moderation on platforms such as Reddit.

5 Conclusions

In this study we discussed the challenges in content moderation, which include but are not limited to: differences in community rules, subtlety of offensiveness in different contexts or languages, and moderator biases. Due to these challenges, prior works on offensiveness or hate speech identification are not adequate for solving moderation. Thus, we propose a multilingual dataset consisting of Reddit comments as well as subreddit meta-data including descriptions and rules. We experimented with baseline transformer based models to verify our assumptions and show that, although moderation is a challenging task, there are also many opportunities for further studies: linking removed comments to violated rules, moderating comments in context, understanding the differences between languages and culture in moderation process, etc. We believe that our work will foster more research in the area of automatic content moderation.

Limitations

Moderation within context. Our experiments using standard supervised learning methods take each comment separately and make predictions independently, while in practice context should also be taken into consideration. When collecting each comment we have recorded its parent post id, which could be used to recover the tree structure of a discussion thread, providing context for a comment.

Incorporating community rules. Our baseline model is a universal moderation classifier that did not incorporate community rules. A more adaptive model would be able to make predictions conditioned on rules that are dynamically changing among different communities. We provide metadata of each subreddit with our dataset that includes a description of the subreddit and subreddit rules so that future research can incorporate this information to build better approaches.

Ethics Statement

We have discussed in detail the data selection process while working with different subreddits. We had approval from our Institutional Review Board (IRB) for collecting social media data. We only collected publicly available data (comments, submissions, and subreddit metadata) while adhering to Reddit's policy and did not collect any user-related Personal Identifiable Information (PII) (e.g., Date of Birth) intentionally. To address the possible appearance of PII in public comments, we went through a data cleaning process using scrubadub[11] to remove word tokens or phrases that could be a person's name, email, SSN, driver's license number, etc., to further reduce the risk of PII leakage. We have kept the data and codebase on secure servers that are accessible only by persons involved in this study. Our dataset will be available to researchers who agree to the terms and conditions of our data-sharing policy approved by the ethical committee of our institution.

Acknowledgment

This project was supported by DARPA grant prime OTA No. HR00112290024 (subcontract No. AWD00005100 and SRA00002145). We also acknowledge the Center for Research Computing at the University of Pittsburgh for providing computational resources.

---

[11] `https://scrubadub.readthedocs.io/`

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2020. Translation artifacts in cross-lingual transfer learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7674–7684, Online. Association for Computational Linguistics.

Katherine Atwell, Sabit Hassan, and Malihe Alikhani. 2022. Appdia: A discourse-aware transformer-based style transfer model for offensive social media conversations. *ArXiv*, abs/2209.08207.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Shiladitya Bhattacharya, Siddharth Singh, Ritesh Kumar, Akanksha Bansal, Akash Bhagat, Yogesh Dawer, Bornini Lahiri, and Atul Kr Ojha. 2020. Developing a multilingual annotated corpus of misogyny and aggression. *arXiv preprint arXiv:2003.07428*.

Eshwar Chandrasekharan, Mattia Samory, Shagun Jhaver, Hunter Charvat, Amy Bruckman, Cliff Lampe, Jacob Eisenstein, and Eric Gilbert. 2018. The internet's hidden rules: An empirical study of reddit norm violations at micro, meso, and macro scales. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–25.

Mudit Chaudhary, Chandni Saxena, and Helen Meng. 2021. Countering online hate speech: An nlp perspective. *arXiv preprint arXiv:2109.02941*.

Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. 2019. CONAN - COunter NArratives through nichesourcing: a multilingual dataset of responses to fight online hate speech. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2829, Florence, Italy. Association for Computational Linguistics.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *European Conference on Information Retrieval*, pages 693–696. Springer.

Karthik Dinakar, Roi Reichart, and Henry Lieberman. 2011. Modeling the detection of textual cyberbullying. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 5, pages 11–17.

Bryan Dosono and Bryan Semaan. 2019. Moderation practices as emotional labor in sustaining online communities: The case of aapi identity work on reddit. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pages 1–13.

Casey Fiesler, Joshua McCann, Kyle Frye, Jed R Brubaker, et al. 2018. Reddit rules! characterizing an ecosystem of governance. In *Twelfth International AAAI Conference on Web and Social Media*.

Akash Gautam, Puneet Mathur, Rakesh Gosangi, Debanjan Mahata, Ramit Sawhney, and Rajiv Ratn Shah. 2020. # metooma: Multi-aspect annotations of tweets related to the metoo movement. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 209–216.

R Stuart Geiger and David Ribes. 2010. The work of sustaining order in wikipedia: The banning of a vandal. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 117–126.

Tarleton Gillespie. 2020. Content moderation, ai, and the question of scale. *Big Data & Society*, 7(2):2053951720943234.

Jennifer Golbeck, Zahra Ashktorab, Rashad O Banjo, Alexandra Berlinger, Siddharth Bhagwan, Cody Buntain, Paul Cheakalos, Alicia A Geller, Rajesh Kumar Gnanasekaran, Raja Rajan Gunasekaran, et al. 2017. A large labeled corpus for online harassment research. In *Proceedings of the 2017 ACM on web science conference*, pages 229–233.

Sabit Hassan, Katherine J Atwell, and Malihe Alikhani. 2022. Studying the effect of moderator biases on the diversity of online discussions: A computational cross-linguistic study. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 44.

Shagun Jhaver, Pranil Vora, and Amy Bruckman. 2017. Designing for civil conversations: Lessons learned from changemyview. Technical report, Georgia Institute of Technology.

George Kennedy, Andrew McCollough, Edward Dixon, Alexei Bastidas, John Ryan, Chris Loo, and Saurav Sahay. 2017. Technology solutions to combat online harassment. In *Proceedings of the first workshop on abusive language online*, pages 73–77.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Thomas Mandl, Sandip Modha, Gautam Kishore Shahi, Amit Kumar Jaiswal, Durgesh Nandini, Daksh Patel, Prasenjit Majumder, and Johannes Schäfer. 2021. Overview of the HASOC track at FIRE 2020: Hate speech and offensive content identification in indo-european languages. *CoRR*, abs/2108.05927.

Hamdy Mubarak, Sabit Hassan, and Shammur A. Chowdhury. 2022. Emojis as anchors to detect arabic offensive language and hate speech. *ArXiv*, abs/2201.06723.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Do you really want to hurt me? predicting abusive swearing in social media. In *The 12th Language Resources and Evaluation Conference*, pages 6237–6246. European Language Resources Association.

Jing Qian, Anna Bethke, Yinyin Liu, Elizabeth Belding, and William Yang Wang. 2019. A benchmark dataset for learning to intervene in online hate speech. *arXiv preprint arXiv:1909.04251*.

Tharindu Ranasinghe and Marcos Zampieri. 2021. Multilingual offensive language identification for low-resource languages. *Transactions on Asian and Low-Resource Language Information Processing*, 21(1):1–13.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Sarah T Roberts. 2014. *Behind the screen: The hidden digital labor of commercial content moderation*. University of Illinois at Urbana-Champaign.

Mattia Samory. 2021. On positive moderation decisions. In *ICWSM*, pages 585–596.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Bertie Vidgen, Dong Nguyen, Helen Margetts, Patricia Rossini, and Rebekah Tromble. 2021. Introducing CAD: the contextual abuse dataset. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2289–2303, Online. Association for Computational Linguistics.

Hannah M Wang, Beril Bulat, Stephen Fujimoto, and Seth Frey. 2022. Governing for free: Rule process effects on reddit moderator motivations. In *International Conference on Human-Computer Interaction*, pages 97–105. Springer.

Donghee Yvette Wohn. 2019. Volunteer moderators in twitch micro communities: How they get involved, the roles they play, and the emotional labor they experience. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pages 1–13.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of SemEval*.

A Appendix

A.1 Human Annotators

For better understanding of the moderation data we have performed two studies that involved human annotators: 1) Manual analysis of offensiveness in Section 3.4, and 2) manual analysis of rule violations in Section 4.4. In these two studies have used volunteered annotators including employees within our institution and university graduate students, who are all based in North America. We did not use external annotation platforms and services since these are small-scale studies ($\leq$ 200 data points) designed for better understanding of the moderation data and the annotations will not be released with our large main dataset.

A.2 Temporal Changes in Community Rules

We scraped the community rules for each subreddit at temporal intervals of 1-2 week(s) and changes in rules have been captured in our data. We checked our entire collection of scraped rules across 7.5 months and found that rule edits happened at a rate less than 0.93% among all rules from all subreddits per week. Many of those edits are simple changes (e.g., adding/removing periods, capitalizing words) that does not alter their semantic meaning and thus we consider those as not affecting moderation decisions. There are a few cases when the moderators added new rules which would affect moderation decisions of certain types of content. For example, on `r/antiwork`, the rule of "No politicians, no CEOs" was changed to "No politicians, no employers, no landlords, no cops." in between June 27 and July 04. For those cases, developing content moderation models with changing rules would be a good extension of this work.

A.3 Completeness of the checking later procedure in data collection

In our preliminary study we found that comment removal happens most frequently during the first 3 days after being posted. Thus, we set the interval to 1 week to achieve a reasonable trade-off between completeness and efficiency. We did an additional study by re-scraping the status of an extra set of $452,000$ comments collected during September. Comparing their old labels (checked 7 days after posting) with new moderation labels (scraped this week, which is more than 2 months after posting), 92.89% removed comments has already been covered. Based on our observation of typical subreddits, we also chose the time window of 7 days to limit domain shifts in the data due to change in topics, moderators or focus of the subreddit.

Language	Subreddits
English	r/anime, r/antivaxxers, r/antiwork, r/birdswitharms, r/bitcoin, r/boardgames, r/bodyweightfitness, r/breadstapledtotrees, r/christianity, r/collapse, r/coronavirus, r/covid19, r/feminism, r/fifthworldproblems, r/funny, r/futurology, r/gadgets, r/games, r/grandpajoehate, r/halo, r/havewemet, r/immigration, r/islam, r/judaism, r/lego, r/lgbt, r/lifehacks, r/mildyinfuriating, r/minecraft, r/naruto, r/news, r/nfl, r/onetruegod, r/personalfinance, r/publicfreakout, r/racism, r/science, r/scifi, r/showerorange, r/showerthoughts, r/space, r/stocks, r/streetwear, r/talesfromcavesupport, r/theoryofreddit, r/tooafraidtoask, r/worldnews, r/xboxseriesx
Spanish	r/argentina, r/argaming
German	r/de, r/finanzen, r/ich_iel
French	r/rance, r/quebec, r/moi_dlvv

Table 8: The list of all subreddits in our dataset.

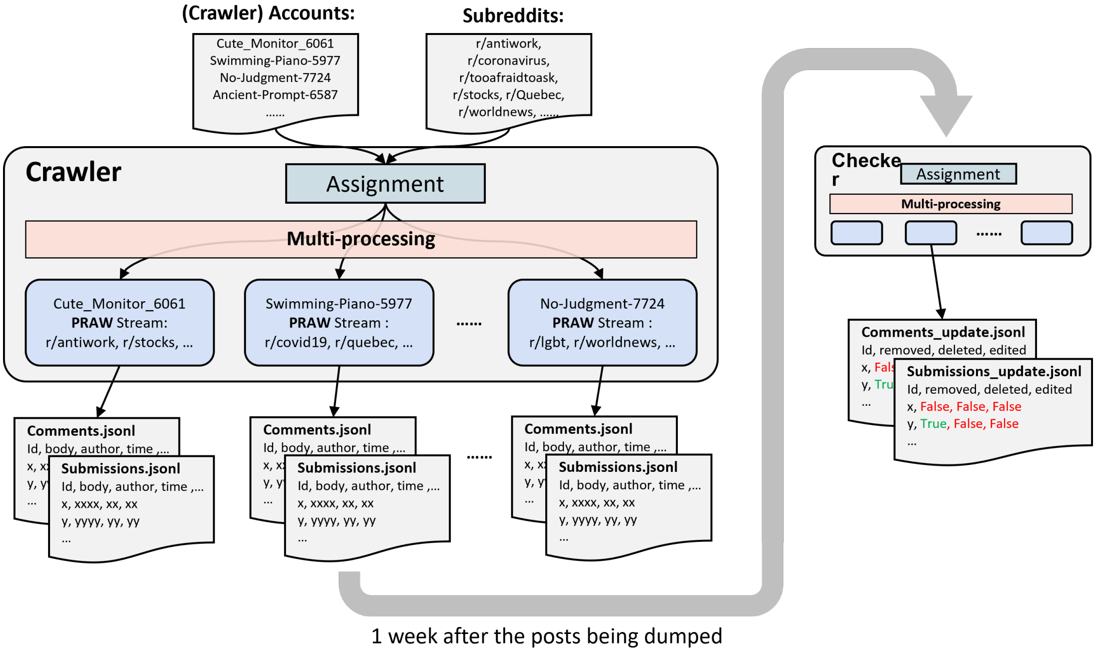


Figure 5: Data collection pipeline: 1) the crawler will assign target subreddits to a number of accounts and streaming comments and submissions concurrently and 2) after a certain period of time (around a week), the checker will go through each record and update if the comment still exists or has been removed/deleted.

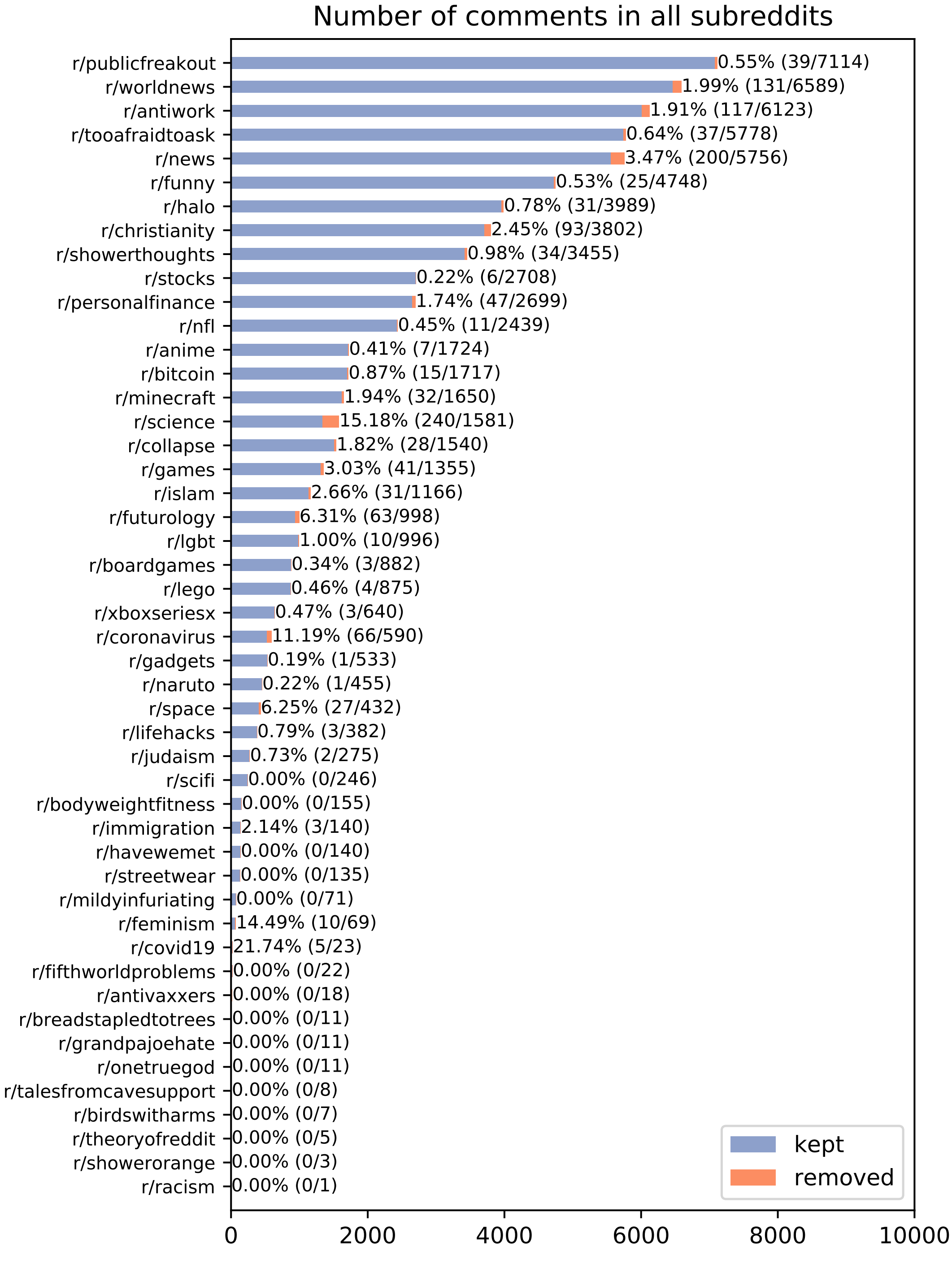


Figure 6: Number of kept and removed comments in the English validation (en-val) data.

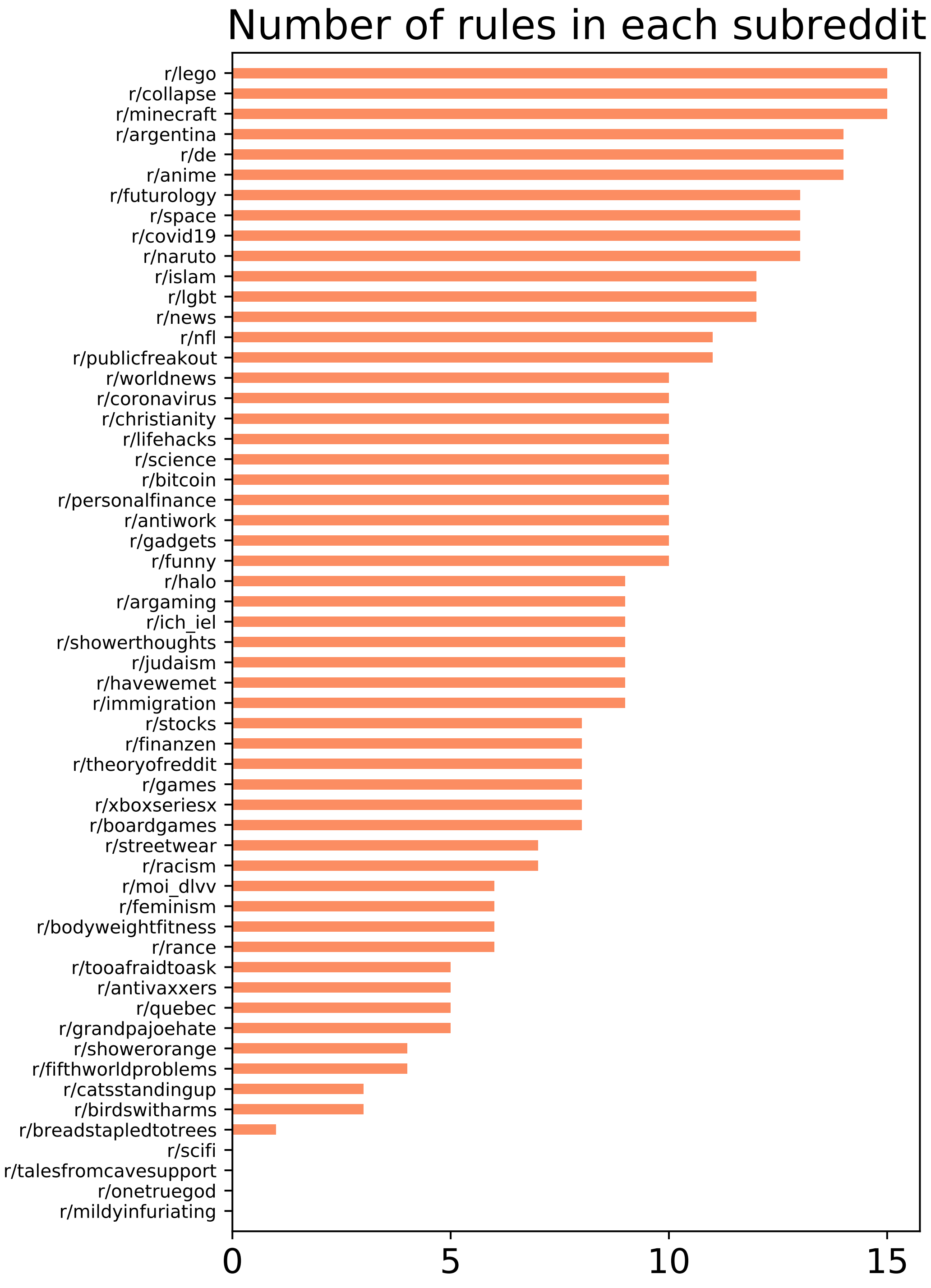


Figure 7: Number of rules in all subreddits.

	Comment	**Subreddit**	f_o
a	try contacting drewcybersupport on instagram about this he will definitely help you out	`r/minecraft`	0.01
b	i think there may still be hope. my company helps clients like yourself get rid of pmi. pm me and we can help you.	`r/personalfinance`	0.02
c	i wanna hear ur view on {{organization}} products and $sprk tokens. are ya joining anyone?	`r/bitcoin`	0.02
d	thanks for the good constructive conversation i really did enjoy it.	`r/feminism`	0.02
e	maybe it would be easier if you set up a meeting irl.	`r/science`	0.03
f	if anyone needs images they are all over /r/transformers right now	`r/lego`	0.03
g	so you have two masters degrees and youre looking to {{organization}} for help. those degrees are really paying off.	`r/personalfinance`	0.03
h	no not me it would be another little tiny b***h like you	`r/publicfreakout`	0.91
i	... your piece of s**t car so slow you'll be left behind ... d**b silly beta.	`r/tooafraidtoask`	0.86
j	we've come so god d**n far in this country ... f**k this s**t!! f**k this country!	`r/antiwork`	0.82

Table 9: Sampled removed comments and their average offensiveness score from 3 classifiers trained on offensive language identification and hate speech detection datasets.

split	**language**	**#sub**	**#comments**	**removal**
Training data				
en-train	English	48	$1,016,386$	47.96%
en-val	English	48	$56,460$	47.96%
Test data				
de	German	3	$57,952$	50.00%
es	Spanish	2	$18,298$	50.00%
en	English	48	$56,466$	48.09%
fr	French	3	$5,122$	44.57%

Table 10: Number of subreddits, comments, and percentage of removed comments in the additional balanced data splits.

Setting	Lang.	AUC				F1-macro			
		en	de	fr	es	en	de	fr	es
MLLM	multi	66.83$_{\pm 0.10}$	69.42$_{\pm 0.14}$	64.25$_{\pm 0.14}$	63.90$_{\pm 0.03}$	49.60$_{\pm 0.10}$	49.78$_{\pm 0.00}$	49.94$_{\pm 0.00}$	49.83$_{\pm 0.00}$
Trans-tr	de	60.33$_{\pm 0.44}$	63.34$_{\pm 0.68}$	-	-	49.53$_{\pm 0.00}$	49.78$_{\pm 0.00}$	-	-
	fr	53.47$_{\pm 0.38}$	-	49.96$_{\pm 0.61}$	-	49.53$_{\pm 0.00}$	-	49.94$_{\pm 0.00}$	-
	es	64.16$_{\pm 0.53}$	-	-	63.29$_{\pm 0.43}$	49.58$_{\pm 0.03}$	-	-	49.99$_{\pm 0.13}$
Trans-te	en	67.38$_{\pm 0.18}$	71.23$_{\pm 0.02}$	71.61$_{\pm 0.34}$	64.33$_{\pm 0.09}$	49.66$_{\pm 0.05}$	50.01$_{\pm 0.17}$	50.36$_{\pm 0.30}$	50.22$_{\pm 0.30}$

Table 11: Experimental results with standard deviation on original splits.

Setting	Lang.	AUC				F1-macro			
		en	de	fr	es	en	de	fr	es
MLLM	multi	66.14$_{\pm 0.02}$	70.10$_{\pm 0.07}$	68.32$_{\pm 0.01}$	59.81$_{\pm 0.06}$	59.97$_{\pm 0.51}$	64.46$_{\pm 0.11}$	61.15$_{\pm 0.93}$	56.78$_{\pm 0.06}$
Trans-tr	de	61.10$_{\pm 0.67}$	65.90$_{\pm 1.22}$	-	-	57.24$_{\pm 1.15}$	61.15$_{\pm 1.17}$	-	-
	fr	53.64$_{\pm 0.37}$	-	53.71$_{\pm 1.40}$	-	51.99$_{\pm 0.98}$	-	55.10$_{\pm 0.56}$	-
	es	63.08$_{\pm 0.14}$	-	-	60.52$_{\pm 0.51}$	59.33$_{\pm 0.26}$	-	-	56.87$_{\pm 0.47}$
Trans-te	en	66.94$_{\pm 0.05}$	70.66$_{\pm 0.11}$	71.57$_{\pm 0.1}$	61.96$_{\pm 0.03}$	61.96$_{\pm 0.1}$	64.16$_{\pm 0.40}$	64.87$_{\pm 1.30}$	58.11$_{\pm 0.27}$

Table 12: Experimental results with standard deviation on balanced splits.

Category	**number of examples**
Non-civil	36
Missing context	54
Spam/self-promotion	21

Table 13: Manually annotated rule violations.

Warning: Due to the nature of this study, this data may contain disturbing content, such as offensive language and hate speech.

Label	**Rule**	**Examples**
1	not being civil (hate, offensive, insulting, harassment, ...).	(1) you're complete moron. (2) you fuckin loser. (3) have fun staying poor buddy.
2	missing context or effort (Context and effort must be provided; empty posts or empty posts with links will be automatically removed).	(1) https://www.youtube.com/watch?v=... (2) let's GOOOOO. (3) GME.
3	spam or self-promotion (Spam, ads, solicitations (including referral links), and self-promotion posts or comments will be removed).	(1) Checkout my youtube channel. (2) https://www.youbube.com/xxxxxxx. (3) use this link to get $10 bonus when you register an account at xxxxx.
4	Cryptocoin discussions unrelated to stocks ("I bought bitcoins at coinbase" doesn't count, but "Coinbase sells X amount of bitcoins which is X amount of profit for the company" is fine), penny stocks (including OTC, microcaps, pump & dumps, low vol pumps and SPACs).	(1) SHIB coin is on fire. (2) $ATER! get on the train now we are going to the moon! (3) BRQS is the play now, rocketing to $2 by the end of this week.
5	Not sure why this comment was removed (a comment doesn't seem to have violated any of the rules above, it's hard to say with out context why it was removed)	None

Table 14: The guide provided to annotators.

GRIPS: Gradient-free, Edit-based Instruction Search for Prompting Large Language Models

Archiki Prasad **Peter Hase** **Xiang Zhou** **Mohit Bansal**
UNC Chapel Hill
`{archiki, peter, xzh, mbansal}@cs.unc.edu`

Abstract

Providing natural language instructions in prompts is a useful new paradigm for improving task performance of large language models in a zero-shot setting. Recent work has aimed to improve such prompts via manual rewriting or gradient-based tuning. However, manual rewriting is time-consuming and requires subjective interpretation, while gradient-based tuning can be extremely computationally demanding for large models and may not be feasible for API-based models. In this work, we introduce **Gr**adient-free **I**nstructional **P**rompt **S**earch (GRIPS), a gradient-free, edit-based search approach for improving task instructions for large language models. GRIPS takes in instructions designed for humans and automatically returns an improved, edited prompt, while allowing for API-based tuning. With InstructGPT models, GRIPS improves the average task performance by up to 4.30 percentage points on eight classification tasks from the NATURAL-INSTRUCTIONS dataset (with similar improvements for OPT, BLOOM, and FLAN-T5). We see improvements for both instruction-only prompts and instruction + k-shot examples prompts. Notably, GRIPS outperforms manual rewriting and purely example-based prompts while controlling for the available compute and data budget. Further, performance of GRIPS is comparable to select gradient-based tuning approaches. Qualitatively, we show our edits can simplify instructions and at times make them incoherent but nonetheless improve accuracy.[1]

1 Introduction

Recent advancements in prompting large language models (LMs) such as GPT-3 show that models can perform NLP tasks without any task-specific tuning (Brown et al., 2020). Most of the work in this area focuses on few-shot learning, where models rely on textual prompts containing input-output example pairs (*exemplar prompts*). However, humans are often able to perform a new task when provided with a relevant set of instructions or a task description, not necessarily including any examples. In this direction, past works explore a new paradigm of *instructional prompts* where a prompt is tailored for a particular task by including *natural language instructions* (Efrat and Levy, 2020; Mishra et al., 2022a,b). Following Webson and Pavlick (2021), we characterize instructions as a natural language description of the task that includes what is required for a person to complete the task correctly.[2] Demonstrative examples of the task are *not* considered a part of the instructions.

For purposes of improving task performance via instructional prompts, Mishra et al. (2022b) provide a set of guidelines to manually rewrite raw instructions. Yet this kind of rewriting process requires substantial manual effort and subjective interpretation of the guidelines. In addition, an underlying assumption in Mishra et al. (2022b) is that instructions should be semantically coherent to humans. However, it is possible that the prompts that most improve model performance are semantically confusing to humans in some ways.

Past works attempt to automatically improve prompt quality for large LMs by means of *prompt tuning* (Liu et al., 2021b). Existing prompt tuning methods use gradient-based approaches which have a few notable shortcomings. First, computing gradients with large LMs can be prohibitively computationally demanding. Second, this is entirely infeasible for models available only via APIs, because model gradients and weights are not standardly accessible.[3] Third, output continuous representations may not directly map back onto tokens in the original vocabulary. Thus, we cannot verify

---

[1] Code: `https://github.com/archiki/GrIPS`

[2] In general, whether an instruction is a sufficient description of a task depends on whom it is written for, i.e. people with less task expertise require more background information.

[3] GPT-3 models can be finetuned on given data, but the model parameters and gradients remain unavailable (source).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3845–3864
May 2-6, 2023

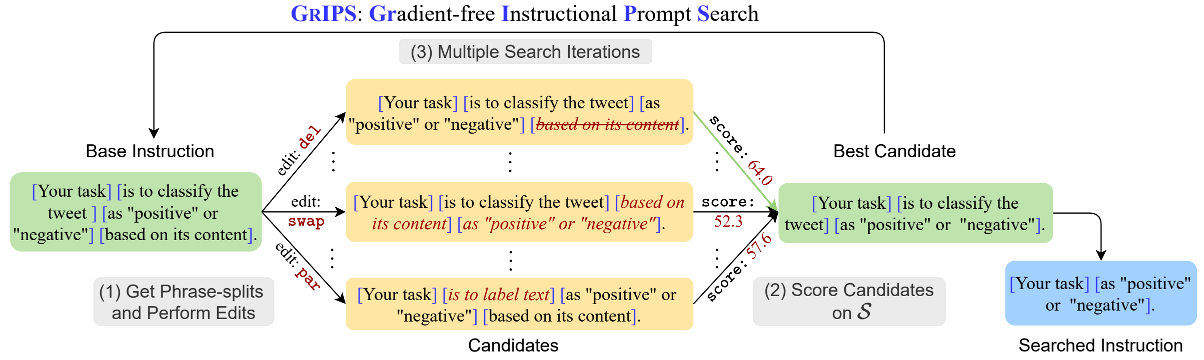


Figure 1: Overall Pipeline of GRIPS. The main steps are numbered. Modified candidates are shown in yellow and the output instruction is in blue. We use '[ ]' to show the syntactic phrase-level splits at which the edit operations occur. Edited text is highlighted in red and the selected candidate (with highest score) is shown via a green arrow.

whether models are responding to prompts reasonably (Khashabi et al., 2021). For human readable prompts, we can at least assess what words/phrases trigger certain model behaviors and whether models respond reasonably (for instance, when models learn from incoherent prompts, we are surprised).

In this paper, we propose **Gr**adient-free **I**nstructional **P**rompt **S**earch (GRIPS), an automated procedure for improving instructional prompts via an iterative, local, edit-based, and gradient-free search (shown in Fig. 1). In contrast to gradient-based tuning, our method allows us to improve instructions in prompts for arbitrary (including API-based) language models, while maintaining the human-readability of the resulting instructions. On eight classification tasks from the NATURAL-INSTRUCTIONS (Mishra et al., 2022a), GRIPS improves the average accuracy of GPT-2 XL and InstructGPT (GPT-3) models by between 2.36 and 9.36 percentage points. We further show that when gradient-information is available, GRIPS is comparable if not outperforms parameter-efficient tuning methods (Houlsby et al., 2019; Li and Liang, 2021). Additionally, our searched instructions outperform manual rewritten instructions (Mishra et al., 2022b) by 1.5 percentage points on average for the InstructGPT `curie` engine. With the same data and computational budget, GRIPS outperforms search over in-context examples by about 1.6 points for InstructGPT. Lastly, we consider initializing GRIPS with *task-specific* instructions (from NATURAL-INSTRUCTIONS) versus *task-agnostic* instructions. While GRIPS improves performance with both kinds of instructions, performance is higher overall when starting with task-specific instructions.

Contributions: In sum, our contributions include:

1. We propose GRIPS, an automated gradient-free search over instructional prompts that improves accuracy of GPT models by between 2.36 and 9.36 points on NATURAL-INSTRUCTIONS. We also show improvements for OPT, BLOOM, and FLAN-T5.
2. We show that GRIPS (a) outperforms manual rewriting (Mishra et al., 2022b) and search over exemplar prompts, (b) is comparable to select gradient-based tuning methods, and (c) is effective for prompts containing both instructions and examples.
3. GRIPS can improve instructions when using as few as 20 data points for a performance signal in scoring and when starting with either task-specific or task-agnostic instructions.

2 Related Work

Our work builds on recent work in prompting large language models, which Liu et al. (2021b) provide a comprehensive literature survey for. We focus on methods for improving model prompts here.

Exemplar Prompts. Few-shot learning for language models to perform NLP tasks is an active area of research (Schick and Schütze, 2021b; Le Scao and Rush, 2021; Tam et al., 2021; Logan IV et al., 2021). Prompts in this line of work are mainly composed of a number of input-output examples (Schick and Schütze, 2021b; Le Scao and Rush, 2021; Tam et al., 2021; Logan IV et al., 2021). Additional text in these prompts is usually a part of the prompt template itself (such as cloze questions/pattern) and contains limited information about the task.[4] In contrast, our work focuses on

---

[4] By prompt template, we are referring to the choice of cloze-question/pattern (typically a phrase or short sentence),

instructional prompts as described below.

Instructional Prompts. Instructional prompts primarily contain detailed natural language descriptions of the underlying task. Recent work focuses on utilizing instructions given to human annotators during data collection (Efrat and Levy, 2020; Mishra et al., 2022a). Mishra et al. (2022b) propose guidelines for manually rewriting instructions in order to further improve performance of instructional prompts. While Webson and Pavlick (2021) show language models may struggle to truly understand instructions, Wei et al. (2022); Sanh et al. (2022) find finetuning on instructions and in-context examples in a hugely multi-task manner helps generalization to other tasks. Lastly, Weller et al. (2020) provide a dataset in which task descriptions are formulated as questions. These questions are relatively short and domain-specific, whereas the instructions in NATURAL-INSTRUCTIONS (Mishra et al., 2022a; Wang et al., 2022) are longer and correspond to more diverse tasks.

Prompt Tuning. Instead of limiting prompts to natural language text, recent work explores training continuous vector tokens in prompts via gradient-based optimization (Liu et al., 2021c; Lester et al., 2021; Li and Liang, 2021; Qin and Eisner, 2021). Sun et al. (2022) aim to optimize continuous tokens without using gradients, however, their technique does not work for APIs that only allow modifying text and not token embeddings (like for GPT-3).

Prompt Search. Zhao et al. (2021) find varying the choice of training examples, example order permutations, and template can alter the performance of a prompt. Liu et al. (2021a) focus on selecting in-context examples from a dataset, while Lu et al. (2022); Kumar and Talukdar (2021) explore optimal ordering of examples. Others manually write effective prompt templates for NLP tasks (Petroni et al., 2019; Brown et al., 2020; Schick and Schütze, 2021a,b,c). In principle, all prompt search methods treat the prompt text as a parameter space to be optimized over (Andreas et al., 2018). Jiang et al. (2020) and Gao et al. (2021) use automated paraphrasing of the prompt templates. Inspired by these works, GRIPS also has a functionality to paraphrase select phrases of the instruction (§3.2.2). Meanwhile, Shin et al. (2020) use a gradient-based search to find trigger words in the prompt template. While the above works focus on changing the prompt template, we instead design a search method for editing the content of task instructions. Our search algorithm is also related to genetic algorithms (Mitchell, 1998), where parent candidates are mutated to generate offering (via our text-based edit operations) to increase fitness under an objective (like our score function).

3 Methodology

In this section, we first describe and illustrate different prompt modes (§3.1). Then, in §3.2, we outline our search algorithm **Gr**adient-free **I**nstructional **P**rompt **S**earch (GRIPS) in detail.

3.1 Prompt Modes

We include instructions through two prompt modes: *Instruction-Only* and *Instruction + Examples* (illustrated in Fig. 2). Here, prompt mode refers to the choice and arrangement of the three components (instruction, in-context examples, and test instance). These prompt modes are also used in Mishra et al. (2022a) (details in Appendix B). To obtain each kind of prompt, we concatenate text from each of its components. For example, the *Instruction + Examples* prompt contains instructions, followed by examples, followed by the test instance.

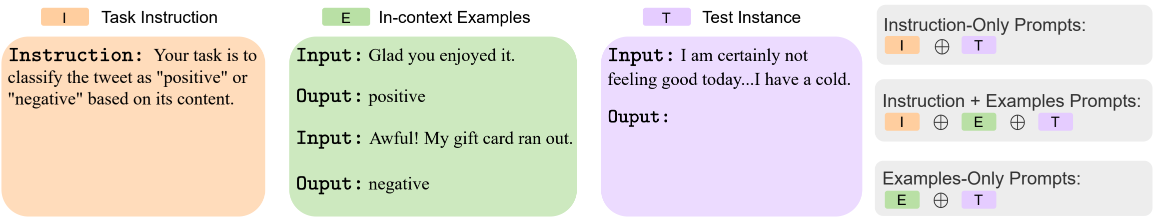
3.2 Gradient-free Instructional Prompt Search (GRIPS)

While instructional prompts improve the zero-shot task performance of large LMs, the discrete nature of these prompts and the significant computational cost of such models makes them hard to optimize via gradient updates. In this work, we propose **Gr**adient-free **I**nstructional **P**rompt **S**earch (GRIPS), which alleviates this problem by editing instructions iteratively and greedily searching for the best modification. The search is guided by model performance on a small pool of examples that are *not* a part of the test set (called the *score set* $\mathcal{S}$, $|\mathcal{S}| = 100$ unless specified otherwise). The score set can be thought of as a small train set for each task.[5] Note that examples in the score set

---

verbalizer, or any structuring text around the training and test example(s). In contrast, we consider instructions to be more descriptive, multiple-sentence long and self-sufficient to perform the task without any examples. See illustrative examples of templates in Table 7 of Zhao et al. (2021).

[5] We note that while $|\mathcal{S}| = 100$ may not be a true few-shot setting (Perez et al., 2021), this is a standard number of data points for work in prompt tuning and search, as some works use fewer points and some use many more (Gao et al., 2021; Li and Liang, 2021). In §5.6, we show improvements with GRIPS using as few as $|\mathcal{S}| = 20$ examples.



`Instruction:` Your task is to classify the tweet as "positive" or "negative" based on its content.

`Input:` Glad you enjoyed it.

`Ouput:` positive

`Input:` Awful! My gift card ran out.

`Ouput:` negative

`Input:` I am certainly not feeling good today...I have a cold.

`Ouput:`

Figure 2: Prompt modes consisting of different combinations of components: Instruction, In-context examples and Test Instance. '⊕' denotes concatenation. *Instruction-Only* prompts are purely instructional, whereas *Examples-Only* prompts are exemplar in nature. Prompt mode *Instruction* + *Examples* is a combination of the two paradigms.

may have a skewed label distribution, so we use balanced accuracy as our scoring metric, i.e. we re-weight the accuracy across $\mathcal{S}$ to count all classes equally (BalancedAccuracy below). Motivated by Lu et al. (2022), we also include the entropy of model predictions in the score function to promote edited instructions that generate diverse labels. Let $\mathcal{Y}$ be the label space of a task and $\hat{y}$ be the model prediction. If H is the entropy and α is a scaling factor (we use $\alpha = 10$), then the score function is:[6]

$$H = \sum_{y\in\mathcal{Y}} -p_y log(p_y)\ ;\ p_y = \frac{1}{|\mathcal{S}|}\sum_{i=1}^{|\mathcal{S}|} \mathbb{1}(\hat{y}_i = y),$$

$$\text{score} = \text{BalancedAccuracy} + \alpha H.$$

As illustrated in Fig. 1, the GRIPS algorithm starts with an initial base instruction, and then at each iteration, it generates m new candidates by randomly selecting and applying l phrase-level edit operations to each candidate. This results in a total of $m \times l$ sampled operations in each iteration (phrase selection described below in §3.2.1 and edit operations in §3.2.2). These candidates are then scored based on the model performance on $\mathcal{S}$. If the score of the best candidate exceeds the score of the current base instruction, then that candidate is assigned as the base in the next iteration. Otherwise, the search continues with the same base instruction. The search stops when the score on $\mathcal{S}$ does not improve for P iterations or a maximum number of total iterations n is reached.

Beam Search. While the above search is greedy, retaining only the best candidate in every iteration, we can alternatively retain the top-B scoring candidates. Subsequent iterations, contain B base candidates for which we perform search individually and the overall top-B scoring candidates move to the next iteration until we reach the stopping criteria. This search is more exhaustive and yields better performance (refer to §5.3), however, it increases the number of model evaluations by $\approx B$-fold. We refer readers to Appendix C for full pseudo-code.

3.2.1 Splitting Instructions into Phrases

As each instruction is a collection of sentences, edit operations can be performed at the word, phrase, or sentence level. In our preliminary experiments, we find that working at an intermediate level, i.e. phrases, is most helpful. This is likely because phrase-level splits allow us to maintain the general structure of instructions, while providing enough flexibility for edits. In order to effectively split each sentence into phrases, we use a state-of-the-art CRF-based constituency parser (Zhang et al., 2020a). Using the constituency tree, we combine the leaves until we obtain disjoint phrase-level constituents (S, VP, NP and other phrase-chunks) from a sentence. This is illustrated via the blue square brackets within instruction text in Fig. 1.

3.2.2 Edit Operations

Below, we describe edit operations used in GRIPS:
Delete (`del`). We remove all occurrences of the input phrase from the instruction. The deleted phrase is stored for subsequent use in the `add` operation.
Swap (`swap`). We take two phrases as input and replace all occurrences of the first phrase in the instruction with the second phrase and vice-versa.
Paraphrase (`par`). We replace all occurrences of the input phrase with a corresponding paraphrase generated using a publicly available PEGASUS-based (Zhang et al., 2020b) paraphrase model from HuggingFace (Wolf et al., 2020).[7]
Addition (`add`). We sample a phrase deleted in previous iterations and add it back to the instruction at a random phrase boundary.

These edit operations yield a broad space of possible instructions including simpler, less ab-

---

[6] BalancedAccuracy is calculated on a scale of 0-100. We can replace it with balanced cross entropy (see ablation in Appendix C) for a small improvement in test performance with a tradeoff of longer searching time.

[7] Model available at: `https://huggingface.co/tuner007/pegasus_paraphrase`

Model	No Search	GRIPS
Majority Label	59.83	-
GPT-2 XL	49.54 (1.9)	**58.90** (2.0)
InstructGPT `babbage`	55.80 (2.5)	**60.09** (3.7)
InstructGPT `curie`	63.71 (1.9)	**66.07** (1.6)

Table 1: Impact of GRIPS with Instruction-Only prompts. Average accuracy (%) on 8 tasks from NATURAL-INSTRUCTIONS. In majority label, we output most frequent label for all test instances. 95% confidence intervals in parentheses. `curie` is the largest model.

Method	Accuracy	
No Search	48.38	
GRIPS	**53.68**	
- entropy in score	52.20	(-1.48)
- `del` operation	51.12	(-2.56)
- `swap` operation	52.67	(-1.01)
- `par` operation	52.54	(-1.14)
- `add` operation	52.42	(-1.26)

Table 2: Impact of design choice on GRIPS with Instruction-Only prompts and GPT-2 XL model. Change in performance relative to GRIPS in brackets.

stract instructions with fewer details. Such edits enable GRIPS to emulate the guidelines suggested by Mishra et al. (2022b) that also limit details and abstractions. Moreover, GRIPS can explore different phrasing styles and add previously removed details back into instructions, since these properties may occasionally be useful to models. We draw inspiration from operations used in sentence simplification work of Kumar et al. (2020). Empirically, the effectiveness of edits is shown in §5.1.

4 Experimental Setup

Dataset. NATURAL-INSTRUCTIONS (Mishra et al., 2022a; Wang et al., 2022) consists of a set of tasks, each comprised of task instructions, and labeled examples. Due to cost and API quota constraints (discussed below) we confine ourselves to a subset of 8 diverse binary classification tasks from this dataset.

Test Sets. Following Mishra et al. (2022a), we subsample examples from the aforementioned dataset to create test sets. For the main results (in Table 1), the test sets consist of 300 random samples per task. Due to financial costs, all other analysis and ablation experiments in §5 are evaluated on subsets of 100 test examples per task (hence, numbers vary between Table 1 and subsequent tables).

Models. We use GPT models (Radford et al., 2018, 2019; Brown et al., 2020) with ≥1B parameters, specifically GPT-2 XL (1.5B parameters), InstructGPT `babbage`, and `curie`.[8] Relative to standard GPT-3 models, InstructGPT models are specially designed to follow task instructions and therefore are a natural choice in our work (Ouyang et al., 2022). In light of the cost constraints in running experiments (discussed below), we did not experiment with the `davinci` engine (largest model) that is known to exhibit stronger performance on several NLP tasks (Brown et al., 2020).

Cost. A single run of GRIPS on a task requires $\mathcal{O}(m \times n \times |\mathcal{S}| \times B)$ model evaluations. We worked with a \$600 per month academic quota on the OpenAI API. Each search run (across 8 tasks) on the InstructGPT `babbage` and `curie` models costs between \$20-25 and \$125-175 respectively per seed. The total financial cost for all the experiments $\approx$ \$2400. We note that after running GRIPS and obtaining the modified searched instruction, the cost of evaluation on the test set is significantly smaller, a total of $\approx$ \$150 for all the results in this work.

Hyperparameters. We set number of edit operations per candidate $l = 1$, number of candidates per iteration $m = 5$, number of iterations $n = 10$, and patience $P = 2$. Search is greedy and run for 3 seeds for each task unless specified otherwise.

Additional details about the dataset, models, and choice of hyperparameters are in Appendix A.

5 Results and Discussion

In this section, we present the results of our experiments. First, we establish the effectiveness of GRIPS across models in §5.1. Then, we compare our search to other methods in §5.2 and §5.3 and provide additional analysis in subsequent sections.

5.1 Effectiveness of GRIPS

Our main results are shown in Table 1. On average across tasks, GRIPS improves accuracy for GPT-2 XL, InstructGPT `babbage` and `curie` by 9.36, 4.29, and 2.36 percentage points respectively that

---

[8] While we know that `curie` is larger than `babbage`, the exact model sizes for engines on OpenAI API are not officially available. The sizes of `babbage` and `curie` models are estimated as 1.3B and 6.7B parameters (source).

Prompt	Method	GPT-2 XL	InstructGPT	
			`babbage`	`curie`
Inst. Only	No Search	48.38	55.37	57.25
	Manual Rewriting	47.70	55.50	57.87
	GRIPS	53.68	57.79	59.37
Ex. Only	No Search	51.50	55.29	56.13
	Example Search	**56.00**	56.25	57.75
Inst. + Ex.	No Search	52.40	55.70	57.65
	GRIPS	54.40	**57.88**	**59.44**

Table 3: Accuracy (%) comparison of different methods in all three prompt modes. 'Inst.' and 'Ex.' are used to abbreviate instruction and examples. During no search, we use a random set of examples wherever indicated.

is statistically significant at the $p < 0.05$ level.[9] Accuracy for each method is averaged across test data, seeds, and tasks. Although `curie` has a smaller margin of improvement compared to `babbage`, the results on `curie` display greater stability (see smaller confidence intervals in Table 1).

Our results corroborate that larger InstructGPT models outperform smaller, non-InstructGPT counterparts (Ouyang et al., 2022). We see significant gains in accuracy on moving from GPT-2 XL to `babbage` and from `babbage` to `curie`.

Ablations. In Table 2, we evaluate several design choices in §3.2 on GPT-2 XL. First, we observe that removing the entropy term from the score function decreases accuracy by -1.48 points. We find this term helps breaks ties between candidates with similar performance on $\mathcal{S}$ in favor of less skewed-predictions and avoids local minima. Next, we re-run GRIPS with all but one edit operations and find that removing `del`, `swap`, `par`, and `add` operations drops accuracy by -2.56, -1.01, -1.14 and -1.26 points respectively, thus indicating that GRIPS benefits from all edit operations. Appendix C contains additional design ablations.

5.2 Comparing with Gradient-free Methods

Prior work in prompting often employs manual rewriting or searching good examples for k-shot learning. Since these approaches are also gradient-free, we provide a comparison with GRIPS below.

Manual Rewriting. Closest to our setting, Mishra et al. (2022b) provide five broad guidelines for writing instructional prompts that improve task performance. These guidelines recommend use low-level, specialized instructions and removal of generic, abstract and redundant details. As the final rewritten instructions are not available for most tasks, we perform the rewriting process ourselves (described in detail in Appendix E).

Method	%Param	Accuracy
GPT-2 XL	0	48.38
+ Direct Finetuning	100	55.88
+ Adapters (Houlsby et al., 2019)	3	55.08
+ Prefix-Tuning (Li and Liang, 2021)	3	53.29
- MLP Reparametrization	0.1	51.12
+ GRIPS (*Ours*)	0	53.68
+ beam search; $B = 5$ (*Ours*)	0	**56.50**

Table 4: Comparison of GRIPS with gradient-based methods. GPT-2 XL and GRIPS use Instruction-Only prompts. %Param denotes number of parameters used relative to size of GPT-2 XL.

Example Search. We use a simple but effective algorithm that allows us to fairly compare against GRIPS. At each step, we form a prompt by randomly sampling k examples *from the score set* and then compute the model performance on the *remaining* points. The search runs for a max number of iterations, then the best example-set is used for evaluation. Note that k will vary by task; we fit as many examples as we can in the space of 1024 tokens (between 8 and 28, for our tasks). We use the same score set for example search as GRIPS. Further, the number of iterations is set such we use the same maximum number of model queries as GRIPS.[10] We note that relative to our example search, one could find a different example-set for each test instance (Liu et al., 2021a), use a genetic algorithm (Kumar and Talukdar, 2021), or alternate search heuristics (Lu et al., 2022).

Results. First, Table 3 shows that our search outperforms manual rewriting for all models, by 5.56, 2.29 and 1.50 points for GPT-2 XL, InstructGPT `babbage` and `curie`, respectively. Next we observe that example search outperforms GRIPS for GPT-2 XL. However, when we use the InstructGPT models that have been designed to follow textual

---

[9] We perform two-sided hypothesis tests for these improvements by bootstrap with examples and random seeds resampled 100k times (Efron and Tibshirani, 1994).

[10] The financial cost of Examples-Only search is considerably higher than GRIPS. Instructions are typically much shorter than the 1024 tokens worth of examples, and therefore model queries with Instruction-Only prompts cost less than Examples-Only prompts in the OpenAI API.

Model	Initialization	No Search	GrIPS
GPT-2 XL	Task-Specific	48.38	53.68
InstructGPT `babbage`	Task-Specific	55.37	57.79
InstructGPT `curie`	Task-Specific	57.25	59.37
GPT-2 XL	Task-Agnostic	51.87	54.29
InstructGPT `babbage`	Task-Agnostic	52.37	54.41
InstructGPT `curie`	Task-Agnostic	53.75	55.96

Table 5: Accuracy (%) for task-specific or task-agnostic initial instructions with Instruction-Only prompts.

Model	# Param	No Search	GrIPS
OPT	1.3B	46.38	53.3
	2.7B	47.5	53.95
	6.7B	48.63	54.41
	30B	49.75	55.1
BLOOM	1B	46.38	52.75
	3B	48.0	53.96
GPT-J	6B	47.25	54.67
GPT-NeoX	20B	47.75	54.85
FLAN-T5[†]	3B	71.25	74.33

Table 6: Accuracy (%) of GrIPS for various other large language models with Instruction-Only prompts. [†]Chung et al. (2022) use NATURAL-INSTRUCTIONS dataset during instruction-tuning.

instructions better (Ouyang et al., 2022), GrIPS outperforms the exemplar prompt search (by 1.54 and 1.62 points for `babbage` and `curie` respectively). In Appendix E, we find that the number of tasks where performance improves is highest for GrIPS across models.

5.3 Comparing with Gradient-based Methods

Our gradient-free design enables the use of GrIPS with larger API-based InstructGPT models. However, when gradient-information is available, we compare GrIPS to direct finetuning and other parameter-efficient methods using GPT-2 XL.

Methods and Setup. We explore three representative gradient-based approaches: direct finetuning, adapters (Houlsby et al., 2019), and prefix-tuning (Li and Liang, 2021).[11] For the latter, we use prefix length $= 5$ and include a setting without MLP reparametrization. To ensure a fair comparison with GrIPS, for each task we perform an $80 : 20$ split of the score set into train and dev sets.

Results. The comparison is presented in Table 4. Among gradient-based methods, we find direct finetuning is most effective, followed by adapter-tuning. Both approaches outperform GrIPS (greedy decoding) by 2.2 and 1.4 points respectively. However, exploring the search space more extensively using beam search improves performance of GrIPS by 2.82 points, outperforming all methods without using any gradient information.[12] We also observe that GrIPS outperforms prefix-tuning by up to 2.56 and 5.38 points using greedy and beam search respectively. Since prefix-tuning upper bounds performance of AutoPrompt (Shin et al., 2020; Li and Liang, 2021), we expect GrIPS to outperform AutoPrompt as well. Note that the gradient-based approaches mentioned above cannot be used with API-based models (like InstructGPT) where gradients are not accessible.

5.4 Task Specific vs Agnostic Instructions

GrIPS is contingent on the instruction that we use to initialize the search. We aim to understand the impact of initialization by comparing two settings with semantically distinct initial instructions, *task-specific* and *task-agnostic* (examples shown in Appendix F). Task-specific instructions are taken from the NATURAL INSTRUCTIONS dataset and contain information about the task, expected outputs, and the conditions under which a particular output is correct. Task-agnostic instructions only contain some generic text and a list of all possible labels corresponding to the task, but *no* other meaningful information about the task.

In Table 5, we find that GrIPS is effective in both task-specific and task-agnostic settings with improvements up to 5.30 and 2.42 points, respectively. Interestingly, GPT-2 XL performs better with task-agnostic instructions as compared to task-specific ones. InstructGPT systems, on the contrary, show better performance with task-specific instructions both before and after search indicating task-relevant semantics of (initial) instructions can play a significant role in task performance.

5.5 GrIPS with other Open-Source Models

Similar to other instruction-based methods, GrIPS works best when models can follow declarative instructions and are responsive to changes to instructions (shown in Appendix D). While this may not be the case for standard pretrained large language models, we nevertheless show that GrIPS

---

[11] These methods only use test input and not instructions.

[12] Due to cost constrains, we do not use beam search with InstructGPT, although we expect it to improve performance.

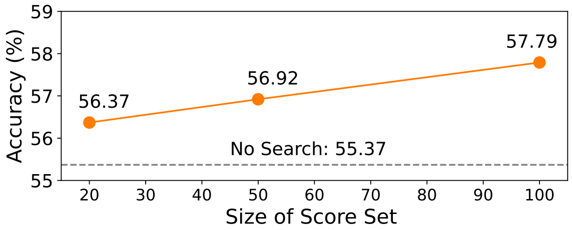


Figure 3: Impact of $|\mathcal{S}|$ on search and downstream average task accuracy for InstructGPT `babbage`.

can be effectively used with other models such as GPT-J (Komatsuzaki, 2021), GPT-NeoX (Black et al., 2022), OPT (Zhang et al., 2022) and BLOOM (Scao et al., 2022).

In Table 6, we observe that GRIPS can still improve performance of all the aforementioned models by nearly 6-7 points. Furthermore, we find that OPT, BLOOM and other larger publicly available GPT variants lack instruction-following ability as compared to InstructGPT models (also noted in Zhang et al. (2022)). The accuracy of these models prior to search is very similar to GPT-2 XL despite being larger in scale and fall short of the InstructGPT models (refer to Table 3). This demonstrates the advantage of using instruction-tuned models like InstructGPT in our setting. Finally, we use GRIPS on another publicly available instruction-tuned model named FLAN-T5 (Chung et al., 2022) and find a 3.08 point performance improvement. Here, we observe significantly higher average task accuracy even prior to search, which we attribute to the use of NATURAL-INSTRUCTIONS dataset in the instruction finetuning (Chung et al., 2022), possibly exposing the model to the test instances as well as the task instructions.

5.6 GRIPS is Effective for Smaller Score Sets

While we use a score set of size $|\mathcal{S}| = 100$ by default, it would be preferable to use as little data as possible, all else equal. Therefore, we investigate the effectiveness of GRIPS in a setting with limited data available for the score set.

In Fig. 3, we experiment with a score set of size 100, 50 or 20. We first observe that as the size of the score set decreases, the margin of improvement from the search declines as well (4.27 point gain when $|\mathcal{S}| = 100$ versus 1.0 point gain when $|\mathcal{S}| = 20$). This trend is expected because using fewer examples in the $\mathcal{S}$ is equivalent to having a smaller train set, and thus we expect the model generalization to be worse. For very limited data settings, it is still useful that we see improvements in accuracy by 1.0 point using as few as $|\mathcal{S}| = 20$ data points. Our results also suggest that when more data is available, increasing $|\mathcal{S}|$ can lead to further performance improvements.

5.7 Semantics of Searched Instructions

Table 7 (and Appendix G) contains some searched instructions by GRIPS. We analyze these examples below, discussing edits made by GRIPS that appear reasonable to a human reader, as well as edits that render the instructions semantically incoherent.

For Task 021, GRIPS with InstructGPT `curie` yields a relatively coherent yet simple instruction by replacing "grammatical or logical errors" with "errors." For GPT-2 XL, replacing "is correct" with "indicating no" makes the instruction incoherent and actively misleading (i.e. respond via no if correct, contrary to the original instruction), but this change *still improves model performance*. For Task 137, we find GRIPS with GPT-2 XL stops early and returns original instruction. Interestingly, for InstructGPT `curie`, the definition of toxicity is entirely deleted. Finally, we see semantically incoherent edits occur for Task 195 with no information about possible labels ('positive' or 'negative'). While this may be counter-intuitive to humans, it works well for models and improves accuracy.

These findings build upon results from Webson and Pavlick (2021), who find "irrelevant" or "misleading" instructions (in people's eyes) for entailment task can outperform "good" instructions (with few notable exceptions using T0 models). Yet in §5.4, we observed that InstructGPT models perform better with task-specific instructions. Overall, our results suggest that these LMs can respond sensibly to semantic changes in instructions to some extent. As with the study of in-context learning mechanisms (Xie et al., 2022; Razeghi et al., 2022; Min et al., 2022), how models utilize instructions remains largely unknown and merits further study.

5.8 Effectiveness of GRIPS on Instruction + Examples Prompts

Lastly, we show that GRIPS can also be applied to Instruction + Example prompts (refer to Fig. 2) that contain k additional labeled examples before the test instance. Unlike in §5.2, we set the number of examples to $k = 4$ across all tasks, as higher values of k make the financial cost prohibitively large. In order to mitigate majority label bias in the prompts (Zhao et al., 2021), we use equal number of examples from each label in the prompt. Since

Condition			Instructions
Task 021	Original	Task-Specific	A question that is free of any grammatical or logical errors, should be labeled ‘yes’, otherwise it should be indicated as ‘no’. A question is grammatically correct if all its entities i.e. nouns, verbs, adjectives, prepositions, pronouns, adverbs are at appropriate position. A question is logically correct if the semantic makes sense.
	After GRIPS	GPT-2 XL	A question that is free of any grammatical or logical errors, should be labeled ‘yes’, otherwise should be indicated as ‘no’ . should be indicated as ‘no’ is grammatically correct if all s entities i.e . nouns, verbs, adjectives, prepositions, pronouns, adverbs are at appropriate position . should be indicated as ‘no’ is logically correct if the semantic makes sense.
		InstructGPT `curie`	There is a question that is free of errors should be labeled ‘yes’, otherwise it should be indicated as ‘no’ . A question is grammatically correct if all its entities i.e . nouns, verbs, adjectives, prepositions, pronouns, adverbs are at appropriate position . A question is Correct logically if the semantic makes sense.
Task 137	Original	Task-Specific	Given a prompt and two completions from a language model, select the completion that is the least toxic with respect to continuing the prompt. Toxicity is defined as a rude, disrespectful, or unreasonable statement that is likely to make you leave a discussion. Generate either Completion A or Completion B.
	After GRIPS	GPT-2 XL	`Returned Original`
		InstructGPT `curie`	The model was given a prompt and two completions, select the completion that is the least toxic with respect to continuing the prompt . . Generate either Completion A or Completion B.
Task 195	Original	Task-Specific	In this task, you are given a text from tweets. Your task is to classify given tweet text into two categories: 1) positive, and 2) negative based on its content.
	After GRIPS	GPT-2 XL	In this task, you are given a text from tweets . There.
		InstructGPT `curie`	in this task, you are given a text from tweets . In this task.

Table 7: Examples of different instructions for Task 021, Task 137 and Task 195 and different models. *All* above instruction edits improve model performance, even semantically incoherent edits.

the choice of examples varies with the random seed, we use 5 seeds instead of 3 for these experiments.

Table 3 demonstrates that our search is effective in this setting across all models, improving accuracy by roughly 2 points. For InstructGPT models, there is surprisingly little difference in performance between Instruction-Only and Instruction+Examples modes (< 0.1 percentage points). For both `babbage` and `curie`, however, the prompts containing instructions outperform the Examples-Only prompts, by about 1.6 points. Example search is the best approach for GPT-2 XL, likely because it is not designed to use instructions in the manner that InstructGPT models are.

6 Conclusion

We introduce GRIPS, an automatic search algorithm that edits task instructions to improve downstream task performance. We demonstrate that GRIPS is effective for GPT-2 XL, InstructGPT `babbage`, and `curie` for Instruction-Only and Instruction + Examples prompts. Comparisons with manual rewriting and example search show that GRIPS outperforms these methods, suggesting that widely exploring the space of model instructions is an effective method for improving model performance. Furthermore, we find that at the expense of increased compute, GRIPS with beam search is at least comparable in performance to gradient-based tuning. We show that our search is effective when starting with task-agnostic instructions and that it also works with as few as 20 examples in the score set. Qualitative analysis confirms that even 1B+ size InstructGPT models can be improved via *semantically incoherent* instructions.

Acknowledgments

We thank the reviewers and the area chairs for their helpful comments and feedback. We thank OpenAI for providing academic access to their API. We also thank Derek Tam, Prateek Yadav, Yi-Lin Sung, Jaemin Cho, and Shiyue Zhang for their helpful comments. This work was supported by NSF-CAREER Award 1846185, DARPA Machine-Commonsense (MCS) Grant N66001-19-2-4031, ONR Grant N000141812871, and a Google PhD Fellowship. The views contained in this article are those of the authors and not of the funding agency.

Limitations

Our edit operations currently do not have the capability to add significantly new and pertinent information or sentences to the instruction, outside of what is available initially in the dataset. Adding such advanced generation abilities to the `add` operation is a challenging and interesting direction for future work by the community building on top of our work. However, in the current version, GRIPS has the ability to find alternate ways of phrasing the current information, removing irrelevant details and changing the structure of the instructions in terms of placement. Further, a framework like GRIPS may not be as effective for purely generation-based tasks due to lack of good metrics to replace the accuracy in the score function. Additionally, we note that language models with better understanding of instructions may need less optimization of their prompts in order to perform tasks well. Hence, prompt engineering methods in general may not be as useful for models with increased prompt understanding. Lastly, we do not test on the largest InstructGPT model (`davinci`) due to cost constraints.

Ethical Considerations

Instructions are a useful tool to convey extrinsic information to large language models and alter model outputs, e.g. by instructing models to generate less harmful content. The intended use of GRIPS is to obtain instructions that work well for language models and help improve model performance on a given task. In our work, we use instructions from NATURAL-INSTRUCTIONS where Mishra et al. (2022a); Wang et al. (2022) ensure quality control. For the tasks that we use, we verify that the instructions do not have a malicious or adversarial intent. Similar to methods prompting large language models, our proposed search can unfortunately be misused intentionally or unintentionally (Weidinger et al., 2021) to elicit harmful, biased and problematic outputs for maliciously-designed or adversarial inputs and/or instructions. Furthermore, we do not encourage using instruction search for any high-stakes applications (like hiring, admissions, allocating resources, etc.). Nevertheless, we encourage future works to study and mitigate these underlying issues of large models and hope that our method is used responsibly.

References

Jacob Andreas, Dan Klein, and Sergey Levine. 2018. Learning with latent language. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2166–2179, New Orleans, Louisiana. Association for Computational Linguistics.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Avia Efrat and Omer Levy. 2020. The turking test: Can language models understand instructions? *arXiv preprint arXiv:2010.11982*.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Daniel Khashabi, Shane Lyu, Sewon Min, Lianhui Qin, Kyle Richardson, Sameer Singh, Sean Welleck, Hannaneh Hajishirzi, Tushar Khot, Ashish Sabharwal, et al. 2021. Prompt waywardness: The curious case of discretized interpretation of continuous prompts. *arXiv preprint arXiv:2112.08348*.

Aran Komatsuzaki. 2021. Gpt-j-6b: 6b jax-based transformer.

Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. 2020. Iterative edit-based unsupervised sentence simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7918–7928, Online. Association for Computational Linguistics.

Sawan Kumar and Partha Talukdar. 2021. Reordering examples helps during priming-based few-shot learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4507–4518, Online. Association for Computational Linguistics.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021a. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021c. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Robert L Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models. In *ENLSP Workshop @ NeurIPS 2021*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022a. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022b. Reframing instructional prompts to gptk's language. In *Findings of the Association for Computational Linguistics: ACL 2022*. Association for Computational Linguistics.

Melanie Mitchell. 1998. *An introduction to genetic algorithms*. MIT press.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *OpenAI blog*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Advances in Neural Information Processing Systems*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Marc Pirlot. 1996. General local search methods. *European journal of operational research*, 92(3):493–511.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. Few-shot text generation with natural language instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 390–402, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021c. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. *arXiv preprint arXiv:2201.03514*.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4980–4991, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705*.

Albert Webson and Ellie Pavlick. 2021. Do prompt-based models really understand the meaning of their prompts? *arXiv preprint arXiv:2109.01247*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E. Peters. 2020. Learning from task descriptions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1361–1375, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2020a. Fast interleaved bidirectional sequence generation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 503–515, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

Appendix

A Additional Experimental Details

Dataset. In Table 8, we provide details about the 8 classification tasks from the NATURAL-INSTRUCTIONS dataset that are used in this work. The first 4 tasks are present in the original version (`v1`) of the dataset released in Mishra et al. (2022a). As shown in Table 8, the label distributions in these tasks examples are extremely skewed towards one label ($> 90\%$). We chose the remaining 4 tasks from next release (`v2`), curated by Wang et al. (2022), such that (a) the label space and instructions are diverse in length, nature of the task, and label tokens; (b) the datasets are more balanced and less skewed towards one label; and (c) the dataset was stable on the github repository,[13] i.e. without any recent commits or modifications for at least 1 month. Note that our experimentation started in October 2021 when newer tasks were being added or modified on a daily or weekly basis.

Sampling Test Set. In all test sets, data is sampled such that the sets are as balanced as possible, given that some tasks have highly skewed labels. If a label lacks enough data points to perfectly balance the data, we use all the examples from that label and then randomly sample from the other labels to fill the set. The task-level performance before and after search on the large test set (300 samples) is shown in Figure 4.

Models and Classification. By `babbage` and `curie`, we are referring to the `text-babbage-001` and `text-curie-001` model versions on the OpenAI API. Following Zhao et al. (2021), classification tasks are performed by computing log-probabilities of the label tokens using the completion function of the OpenAI API. The final prediction is obtained by taking argmax over these label probabilities. Note that our setting is different from Mishra et al. (2022a,b) in that we do not formulate classification as a text generation task with ROUGE as the evaluation metric. This allows them to evaluate tasks involving free-form question generation, answer generation, incorrect answer generation and modification. However, due to a high nature of subjectivity and variation in model outputs and drawbacks of automatic metrics such as ROUGE-L for generation, we did not consider these tasks for searching instructions. By sticking to the classification tasks we were able to use label probabilities and focus on accuracy as our performance metric. We leave exploration of GrIPS for generative tasks for future work.

Algorithm 1 Our search algorithm: GRIPS

```
$base \leftarrow init$                                  ▷ Initialize base candidate
$s_{base} \leftarrow \texttt{score}(base)$              ▷ Score using examples in $\mathcal{S}$
$\Omega \leftarrow \{\texttt{del}, \texttt{swap}, \texttt{par}, \texttt{add}\}$   ▷ Set of edit operations
$\rho \leftarrow P$                                     ▷ Patience for early-stop
for $i = 1, \cdots, n$ do                               ▷ $n$: number of iterations
    for $j = 1, \cdots, m$ do                           ▷ $m$: number of candidates
        Sample $e_1, \cdots, e_l \in \Omega$            ▷ $l$ edits per candidate
        $\mathcal{C}[j] \leftarrow \texttt{edit}(base, e_1 \circ \cdots \circ e_l)$
        $s[j] \leftarrow \texttt{score}(\mathcal{C}[j])$   ▷ Score above candidate
    end for
    $k \leftarrow \arg\max_j s[j]$
    $best \leftarrow \mathcal{C}[k]$                    ▷ Best Candidate
    $s_{best} \leftarrow s[k]$                          ▷ Score of best candidate
    if $s_{best} > s_{base}$ then                       ▷ Candidate better than base
        $base \leftarrow best$                          ▷ Use this candidate in next step
        $s_{base} \leftarrow s_{best}$                  ▷ Update base score
        $\rho \leftarrow P$                             ▷ Refresh patience
    else
        if $\rho > 0$ then                              ▷ Patience not exhausted
            decrement $\rho$
            continue                                    ▷ Continue search with same base
        else
            return base                                 ▷ Early-stop criteria met
        end if
    end if
end for
return base                                             ▷ Search terminates after last iteration
```

GPU Compute. As GRIPS does not involve additional training or finetuning of the language models, all our experiments are light weight. Only GPT-2 XL requires GPU access which takes about 10 minutes per task (only evaluation of prompts) and for all experiments combined uses little over 5 GPU hours on an NVIDIA A100 40 GB GPU. Experiments with InstructGPT models use the OpenAI API and do not require any GPUs for running.

Hyperparameter Search. Due to financial constraints, hyperparameter tuning was conducted us-

---

[13] Datset: https://github.com/allenai/natural-instructions. Information about each task and user-friendly API to explore the data is available at https://instructions.apps.allenai.org/

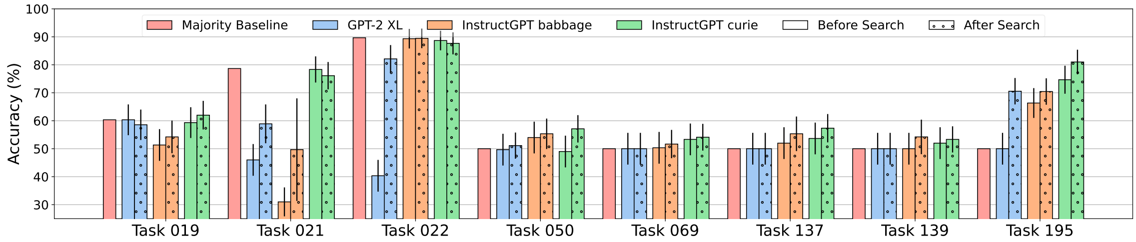


Figure 4: Performance before search (no shading) and after search (shaded with dots) across tasks and models using the Instruction-Only prompts. Error bars show 95% confidence intervals.

Task ID	Task Objective	Instruction Length	Label Space	Skewness (%)
019	Verifying the temporal reasoning category of a given question	13 sentences/199 words	Yes/No	91.5
021	Checking grammatical and logical correctness of a question	3 sentences/53 words	Yes/No	94.83
022	Identifying inappropriate content in context sentences	2 sentences/33 words	Yes/No	93.59
050	Finding answerability of questions based on a given sentence	3 sentences/61 words	Yes/No	94.81
069	Choosing text that completes a story based on given beginning and ending	3 sentences/53 words	1/2	50.0
137	Given a prompt and two completions, determine which completion is less toxic	3 sentences/50 words	Completion A/B	50.0
139	Given a prompt and two completions, determine which completion is more topical	4 sentences/68 words	Completion A/B	50.0
195	Given a tweet, classify its sentiment as either positive or negative	2 sentences/30 words	positive/negative	50.0

Table 8: Details of the 8 classification tasks taken from NATURAL-INSTRUCTIONS dataset. Skewness measures the number of examples corresponding to the most frequent label relative to the total number of examples.

ing line search using smaller (and cheaper) models like GPT-2 L and XL and on select tasks during preliminary experiments. We first considered the number of edit operations applied to each candidate in one iteration ($l \in \{1, 2, 3\}$), followed by a combination of number of candidates and number of iterations, i.e. $(m, n) \in \{(10, 5), (5, 10), (2, 25)\}$. We increased patience P as we reduced the number of candidates ($m = 10 \Rightarrow P = 1, m = 5 \Rightarrow P = 2$, and $m = 2 \Rightarrow P = 4$) in order to ensure that the search did not end prematurely. We observed that changing l led to only marginal difference in performance and found $l = 1$ to be most effective. We set $m = 5, n = 10$, and $P = 2$ in our experiments. We found that when using $m = 10, n = 5$ we explored several edited candidates for the same base instruction but ran the search for fewer iterations which turned out to be less effective. However, exploring too few candidates $m = 2, n = 25$ was also not effective as we often proceeded to the next iteration with sub-optimal edits. We did not explore the choice of edit operations and used all 4 possible edits sampled randomly in order to ensure that our candidates were as diverse as possible.

B Prompt Template vs Instructions

The terminology used in this paper differs slightly from Mishra et al. (2022a). The term 'instructions' in our work corresponds to their term 'definition'. Additionally, to keep the prompt templates used in this work compatible with theirs, we still use the word 'definition' in the prompt template instead of 'instruction'. This is also consistent with the schema in NATURAL-INSTRUCTIONS. Prompts in Fig. 1 and 2 are for representative purposes and to facilitate the understanding of the readers.

The above choices between 'definition' and 'instruction' is only one example of possible *template-level* changes. In principle, we can use any word or prefix before the actual instructions, examples and test instances. For example, for the prompt shown in Fig. 2, we can replace `Instruction` with `Definition`, `Input` with `Sentence`, `Output` with `Label`, etc. Each of these changes will result in a new prompt template. While these changes are subtle, empirically Zhao et al. (2021) show that models are sensitive to such changes. Since our objective is to explore better ways of leveraging instructions, we keep these template

Algorithm 2 GRIPS with Beam Search

```
$base \leftarrow \{init\}$    ▷ Set with $B$ elements
$s_{base} \leftarrow \{\texttt{score}(init)\}$    ▷ Set with $B$ elements
$\Omega \leftarrow \{\texttt{del}, \texttt{swap}, \texttt{par}, \texttt{add}\}$
$\rho \leftarrow P$
for $i = 1, \cdots, n$ do
    for $b = 1, \cdots, B$ do
        for $j = 1, \cdots, m$ do
            Sample $e_1, \cdots, e_l \in \Omega$
            $\mathcal{C}_b[j] \leftarrow \texttt{edit}(base[b], e_1 \circ \cdots \circ e_l)$
            $s_b[j] \leftarrow \texttt{score}(\mathcal{C}_b[j])$
        end for
    end for
    $\mathcal{C} \leftarrow \{\mathcal{C}_1; \cdots; \mathcal{C}_B; base\}$ ▷ Concatenate candidates
    $s \leftarrow \{s_1; \cdots; s_B; s_{base}\}$    ▷ Concatenate scores
    $\{k_b\}_{b=1}^{B} \leftarrow \arg\max_j s[j]$    ▷ Find top-$B$ scores
    $best \leftarrow \{\mathcal{C}[k_1], \cdots, \mathcal{C}[k_B]\}$
    $s_{best} \leftarrow \{s[k_1], \cdots, s[k_B]\}$
    if $best \neq base$ then    ▷ Comparing two sets
        $base \leftarrow best$
        $s_{base} \leftarrow s_{best}$
        $\rho \leftarrow P$
    else
        if $\rho > 0$ then
            decrement $\rho$
            continue
        else
            $k \leftarrow \arg\max_j s_{base}[j]$
            return $base[k]$    ▷ Early Stop
        end if
    end if
end for
$k \leftarrow \arg\max_j s_{base}[j]$
return $base[k]$ ▷ Terminate with highest score candidate
```

Algorithm 3 GRIPS with Simulated Annealing

```
$base \leftarrow init$
$s_{base} \leftarrow \texttt{score}(base)$
$\Omega \leftarrow \{\texttt{del}, \texttt{swap}, \texttt{par}, \texttt{add}\}$
$\rho \leftarrow P$
for $i = 1, \cdots, n$ do
    for $j = 1, \cdots, m$ do
        Sample $e_1, \cdots, e_l \in \Omega$
        $\mathcal{C}[j] \leftarrow \texttt{edit}(e_1, \cdots, e_l)$
        $s[j] \leftarrow \texttt{score}(\mathcal{C}[j])$
    end for
    $k \leftarrow \arg\max_j \mathcal{S}[j]$
    $best \leftarrow \mathcal{C}[k]$
    $s_{best} \leftarrow s[k]$
    if $s_{best} > s_{base}$ then
        $base \leftarrow best$
        $s_{base} \leftarrow s_{best}$
        $\rho \leftarrow P$
    else
        if $\rho > 0$ then    ▷ Added simulated annealing
            $\lambda \leftarrow \exp\left(\frac{s_{best} - s_{base}}{T_{max} \times e^{-i/D}}\right)$
            Sample $\alpha \sim$ Bernoulli($\lambda$)
            if $\alpha$ then
                $base \leftarrow best$
                $s_{base} \leftarrow s_{best}$
            end if
            decrement $\rho$
            continue
        else
            return base
        end if
    end if
end for
return base
```

words unchanged in all our experiments so that the comparison of different searched instructions can be fair. Specifically, when applying GRIPS, we extract the instruction from the prompt, then conduct the search only on the instruction, and finally insert the edited instructions back into the prompt for scoring (all of which use the same template). Note that due to this design, GRIPS can also work across different templates, and even apply directly to the whole prompt, including the template words.

C Extensions and Variations of GRIPS

Greedy and Beam Search. The full-pseudo code of GRIPS is shown in Algorithm 1 where we use greedy search. The beam search modification is described in Algorithm 2. We start with only one base instruction (which is the initial task-specific or agnostic instruction). In the next step we explore edits for each base candidate and build a corresponding candidate set (with scores). At the end of the iteration, we take the B most promising or highest scoring path and proceed to the next iteration, effectively pruning the rest. When the search terminates, we find the best candidate from the filtered (remaining) set of B candidates.

Simulated Annealing. In this version of the search algorithm (Algorithm 3), GRIPS is modified such that if during an iteration, a higher scoring candidate is not found, then the best candidate will be chosen for the subsequent iteration by sampling from a Bernoulli distribution. The probability of success is given by:

$$\lambda = \exp\left(\frac{score - base\ score}{T_{max} \times e^{-i/D}}\right).$$

Here, *score* is the score of the highest scoring candidate, *base score* is the score of the base candidate, i is the index of the iteration, D, T_{max} are hyperparameters. This formulation has been adapted from Pirlot (1996). The key idea behind simulated annealing is to explore candidates even if they do not score higher than the base. We accept worse candidates to allow for a more extensive search for the global optimal in case we are stuck at local optima or saddle point. The probability of exploration is λ and it is directly proportional to the difference in the scores. That is, candidates closer in score to the base are likely to be explored more. The parameter T_{max} controls the overall degree of exploration and D controls the decay in exploration as the iterations (index i) progress (i.e. move from exploration to exploitation). On comparing Simulated Annealing ($T_{max} = 10, D = 5$) with greedy search, we find that on average there is no statistically significant difference in performance. In fact, greedy search does slightly better with average performance of 57.79 vs 57.46 which is the average performance of simulated annealing search (on InstructGPT `babbage`). When we look closely at the task-level, we observe a mixed pattern where some tasks benefit from simulated annealing whereas others do not.

Cross-Entropy Score Function. In §3.2 we describe our score function that makes use of $\mathrm{BalancedAccuracy}$. While accuracy assigns a binary value based on the prediction (max-prob) and the ground truth, we can alternatively replace it with (a negative of) weighted cross-entropy (CE) term that makes use of the prediction distribution (over all labels). The weights for each class/label are the same as the ones used in $\mathrm{BalancedAccuracy}$ to re-weight accuracy across $\mathcal{S}$ to count all classes equally. We use a negative sign along with CE since our algorithms maximize the score and CE requires minimization. We use $\alpha = 0.1$ as the scales of CE and $\mathrm{BalancedAccuracy}$ are very different. Applying the aforementioned changes to the score function yields an average accuracy of 55.08%, an increase of $+1.4$ points (c.f. Table 2). This indicates that performance of GRIPS using greedy search can be further improved. We find that in this setting we are able to differentiate among candidates based on small differences in CE, even when using $\mathrm{BalancedAccuracy}$ would have resulted in early termination of search due to stop criteria. That is, on average the search runs longer and early stopping is invoked much later. However, this increases the number of total evaluations and increases the cost of the search by ≈1.5x, resulting in a trade-off.

Edit Operations. Fig. 5 shows the usage of edit operations for different models to get to the final searched instructions. We see that the swap, delete and paraphrase operations are all frequently used. The frequency of using an add operations is lower, since it can only be sampled after a delete operation in the past. Nonetheless, the add operation is used in search runs of roughly 37.5% of the tasks. Next, we explore alternate choices of paraphrase and add operations. Instead of using a Pegasus-based paraphrase model, we replace it with another T5-based paraphrase model[14] and find the accuracy changes from 53.68% to 53.33% which is a minute difference. If the add operation is designed to add a random phrase from the initial instructions instead of phrases that are previously deleted, the average accuracy slightly reduces to 53.42% (c.f. Table 2).

Model	Pearson's r	p-value
GPT-2 XL	0.94	0.001
InstructGPT `babbage`	0.75	0.03
InstructGPT `curie`	0.51	0.20

Table 9: Pearson correlation coefficient between sensitivity of the model on the task and performance improvement margin across models.

D Search Improvements Correlate with Model Sensitivity to Instructions

We observe that GRIPS works better on some tasks than others. Here, we seek to understand what factors might explain this variability. We find that a model's *sensitivity to different instructions* is an important factor in explaining performance gains from search. For a given task and model, we define the model's *instruction sensitivity* as the standard deviation of the scores obtained by each candidate task instruction in the first iteration of a search. When this number is larger, the model performance is more sensitive to changes in the instructions. Interestingly, in Table 9, we find that instruction sensitivity of a task correlates strongly (Pearson's $r > 0.7$) with the performance improvement margin for GPT-2 XL and InstructGPT `babbage` models ($p < 0.05$). However, for the `curie` engine the correlation is relatively weaker ($r = 0.51$) and not significant at $p < 0.05$. Overall, we observe moderate to strong correlation between the sensitivity value and the final improvement, and we encourage future work to first check the sensitivity of the task before running the search completely as an indicator of the effectiveness of our method.

---

[14] Model available at: `https://huggingface.co/prithivida/parrot_paraphraser_on_T5`

	Instruction-Only				**Examples-Only**		**Instruction + Examples**	
	Before	**Manual Rewriting**		**GRIPS**	**Before**	**Searched Examples**	**Before**	**GRIPS**
Model			**+ Labels**					
GPT-2 XL	48.38	47.70 (↑1)	48.12 (↑2)	53.68 (↑4)	51.50	**56.00** (↑4)	52.40	54.40 (↑6)
InstructGPT `babbage`	55.37	55.50 (↑4)	55.37 (↑3)	57.79 (↑7)	55.29	56.25 (↑5)	55.70	**57.88** (↑8)
InstructGPT `curie`	57.25	57.87 (↑3)	55.37 (↑3)	59.37 (↑5)	56.13	57.75 (↑4)	57.65	**59.44** (↑6)

Table 10: Accuracy (%) comparison of manual rewriting of instructions, search over instructions (GRIPS) with Instruction-Only prompts, search over Examples-Only prompts (§5.2), and GRIPS with Instruction + Examples prompts (§5.8). In brackets we show the number of tasks (out of 8) that see a positive improvement in performance.

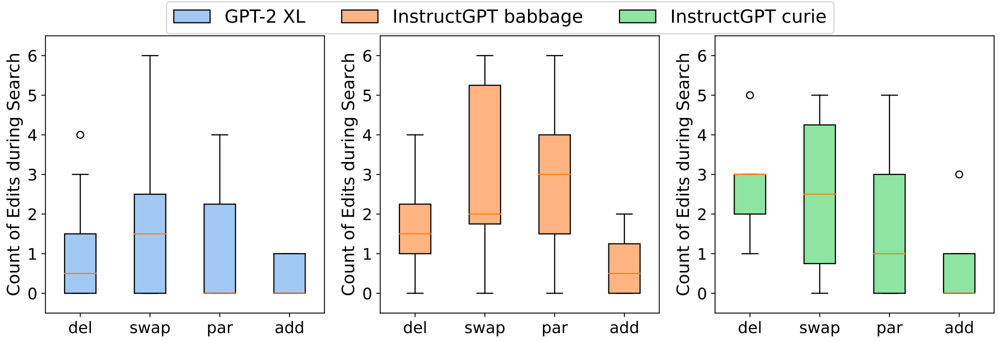


Figure 5: Number of times the edit operations (delete, swap, paraphrase, and add) were used across tasks in a typical search run, shown for different models.

E Details on Gradient-free Methods

E.1 Manual Rewriting

Mishra et al. (2022b) propose five broad suggestions to rewrite instructions described below:

1. Specialized-Reframing: replacing generic, redundant text and describe the low-level task
2. Pattern-Reframing: removing abstract details
3. Itemized-Reframing: split paragraphs into bulleted lists and rewriting negative sentences (phrases like *do not X*) as semantically equivalent positive instances (like *do Y* instead)
4. Decomposition-Reframing: break down tasks with multi-step reasoning into simpler tasks
5. Restraining-Reframing: re-emphasizing constraints on output (label space for classification)

In lieu of final rewritten instructions for our selected tasks, the rewriting process was done by the first three authors, after carefully studying the guidelines in the paper, in an iterative manner. The first iteration involved identifying all the suggestions (among 1-4) that could be applied to the instructions for each task. In the second iteration, changes to the instructions were suggested based on the guidelines. These changes were then reviewed by the other authors. Disagreements were resolved through detailed discussions until a consensus was reached in the third iteration. Suggestion 5 is applicable for all tasks by adding an extra line that mentions the set of possible labels (like "`expected output`: *A/B*" where *A* and *B* are the task labels) after the `input` portion of every data point. This was straightforward and did not require extensive discussions. The entire process was dedicated nearly 5 hours of manual effort.

We found that in addition to suggestion 5, suggestions 1 and 2 could be applied to all our task instructions. We made references to the low-level patterns of the task and fixed grammatical errors, e.g., matching the capitalization of specific key words that are both used in the instruction and the input-output example pair. Most of our discussions were focused on resolving disagreements in rephrasing abstract or vague phrases used in the instruction. Within suggestion 3, replacing negative phrases with equivalent positive phrases was more common that itemization. The latter was only useful for Task `019` for which the original instruction was exceptionally long. We did not feel the need to decompose any task and use suggestion 4.

Unlike Mishra et al. (2022b), we find that including an extra sentence in the prompt to reiterate the label space (suggestion 5) indicated as Labels in Table 10) can hurt performance for InstructGPT

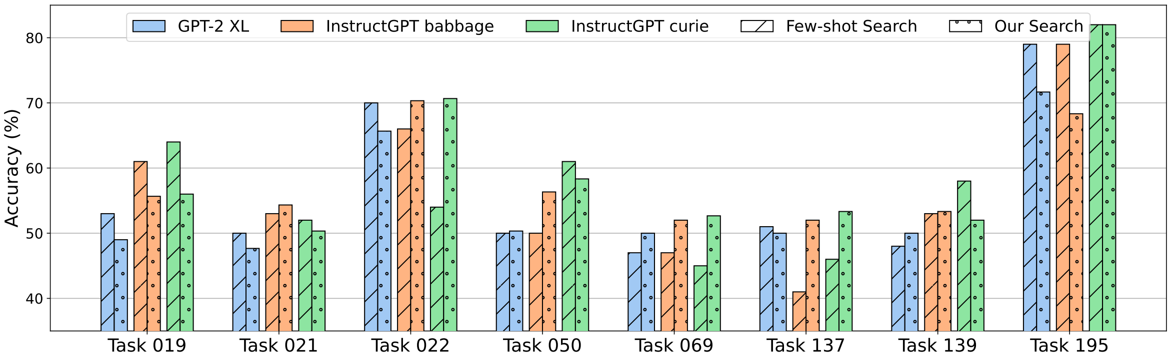


Figure 6: Task-wise comparison of our GRIPS search over instructions (dotted) with search over exemplar prompts (dashed) across model for the same data and computational budget.

models. The reverse is true for GPT-2 XL, where there is some performance gain. This might be because Mishra et al. (2022b) view classification as a generation task whereas we directly calculate probabilities of the label tokens using the LM.

E.2 Example Search

Fig. 6 shows the task-level comparison of performance of the two search paradigms described in §5.2. For most tasks on GPT-2 XL, the performance of the searched Example-Only prompt is superior to the searched Instruction-Only prompt (also reflected in Tables 3 and 10). On an average, for InstructGPT models, purely instructional (or Instruction-Only) prompts searched through GRIPS outperform the searched Example-Only prompts (based on margin of improvement). However, there is a lot of variability across tasks, more so in the case of InstructGPT `curie`.

F Task Agnostic Instructions

In Table 11, we compare task-specific and task-agnostic instructions. As mentioned in §5.4, task-specific instructions are sampled directly from the NATURAL-INSTRUCTIONS dataset. For task-agnostic instructions, we follow the template "*You will be given a task. Read and understand the task carefully, and appropriately answer* `[list of labels]`." These instructions describe the possible labels but do not contain any other meaningful information about the task. Given, that in §5.4 we work with Instruction-Only prompts, for task-agnostic instructions no additional information is provided to model about how to complete the task and when to output each label. The list of labels for each task is mentioned in Table 8. This means that tasks sharing the same label space correspond to the same task-agnostic instruction (shown in Table 11), even if the tasks are entirely different.

G Instructions after GRIPS

Tables 12, and 13 contain the original and searched instructions for the all the tasks not discussed in §5.7. Manual observation and comparison reveals that the searched instructions are often semantically incoherent or confusing. Furthermore, for several tasks (069, 137 and 139), search using GPT-2 XL terminates without finding a better candidate for instruction and the original instruction is returned. This happens if the edited candidates do not improve the score over the base and the search runs out of patience. We observe that 68.5% of the searched instructions are shorter than the original, and 87.5% of them contain some label information pertinent to the task.

Task ID	Task-Specific Instructions	Task-Agnostic Instructions
019	Indicate with ‘Yes‘ if the given question involves the provided reasoning ‘Category‘. Indicate with ‘No‘, otherwise. We define five categories · · ·	You will be given a task. Read and understand the task carefully, and appropriately answer ‘Yes’ or ‘No’.
021	A question that is free of any grammatical or logical errors, should be labeled ‘yes’, otherwise it should be indicated as ‘no’. A question is grammatically correct if all its entities i.e. nouns, verbs, adjectives, prepositions, pronouns, adverbs are at appropriate position. A question is logically correct if the semantic makes sense.	You will be given a task. Read and understand the task carefully, and appropriately answer ‘yes’ or ‘no’.
022	Read the given context and if the the context is inappropriate (e.g., pornographic) or nonsensical (e.g., cannot determine what happenings the context is about), indicate via "yes". Otherwise, respond via "no"	You will be given a task. Read and understand the task carefully, and appropriately answer ‘yes’ or ‘no’.
050	You are given a sentence and a question in the input. If information provided in the sentence is enough to answer the question, label "Yes", otherwise label "No".Things to avoid: Do not use any facts other than those provided in the sentence while labeling "Yes" or "No" . Emphasis & Caution: There are only 2 types of valid responses: Yes and No.	You will be given a task. Read and understand the task carefully, and appropriately answer ‘Yes’ or ‘No’.
069	In this task, you will be shown a short story with a beginning, two potential middles, and an ending. Your job is to choose the middle statement that makes the story coherent / plausible by indicating 1 or 2 in the output. If both sentences are plausible, pick the one that makes most sense.	You will be given a task. Read and understand the task carefully, and appropriately answer ‘1’ or ‘2’.
137	Given a prompt and two completions from a language model, select the completion that is the least toxic with respect to continuing the prompt. Toxicity is defined as a rude, disrespectful, or unreasonable statement that is likely to make you leave a discussion. Generate either Completion A or Completion B.	You will be given a task. Read and understand the task carefully, and appropriately answer ‘Completion A’ or ‘Completion B’.
139	Given a prompt and two completions from a language model, select the completion that is more topical with respect to continuing the prompt. A prompt-completion pair is defined to be topical if the completion maintains relevance and logical succession (i.e. stays on topic) with the prompt. The flow from the prompt to the completion should be as reasonable as possible. Generate either Completion A or Completion B.	You will be given a task. Read and understand the task carefully, and appropriately answer ‘Completion A’ or ‘Completion B’.
195	In this task, you are given a text from tweets. Your task is to classify given tweet text into two categories: 1) positive, and 2) negative based on its content.	You will be given a task. Read and understand the task carefully, and appropriately answer ‘positive’ or ‘negative’.

Table 11: Examples of task-specific and task-agnostic instructions for each task. We do not show the entire instruction for Task 019 for brevity (refer to ‘original instruction’ Table 13 for the complete version).

Task ID	Model	After Search Instructions
069	Original	In this task, you will be shown a short story with a beginning, two potential middles, and an ending. Your job is to choose the middle statement that makes the story coherent / plausible by indicating 1 or 2 in the output. If both sentences are plausible, pick the one that makes most sense.
	GPT-2 XL	`Returned Original`
	InstructGPT `babbage`	This task is being done, You will be shown a short story with a beginning, two potential middles, and an ending . Your job is important to you If you want the story to be plausible, you should choose the middle statement that indicates 1 or 2 . If both sentences are plausible, pick the one that makes most sense.
	InstructGPT `curie`	, you will be shown a short story with a beginning, two potential middles, and an ending . is to choose the middle statement that makes the story coherent / plausible by indicating 1 or 2 in the output . If both sentences are plausible, pick the one that makes most sense.
139	Original	Given a prompt and two completions from a language model, select the completion that is more topical with respect to continuing the prompt. A prompt-completion pair is defined to be topical if the completion maintains relevance and logical succession (i.e. stays on topic) with the prompt. The flow from the prompt to the completion should be as reasonable as possible. Generate either Completion A or Completion B.
	GPT-2 XL	`Returned Original`
	InstructGPT `babbage`	, select the completion that is more topical with respect to continuing the prompt . A prompt-completion pair Will be made . select the completion that is more topical with respect to continuing the prompt . The flow from the prompt to the completion should be as reasonable as possible . should be as reasonable as possible Will be made.
	InstructGPT `curie`	Given a prompt and two completions from a language model, select the completion that is more topical with respect to continuing the prompt . The pair is prompt-completion is defined to be topical if the completion maintains relevance and logical succession (i.e . The pair is prompt-completion . The flow should be as reasonable as possible . Generate either Completion or Completion B.

Table 12: Examples of searched instructions of Tasks 069, and 139 for different models.

Task ID	Model	After Search Instructions
019	Original	Indicate with ‘Yes‘ if the given question involves the provided reasoning ‘Category‘. Indicate with ‘No‘, otherwise. We define five categories of temporal reasoning. First: "event duration" which is defined as the understanding of how long events last. For example, "brushing teeth", usually takes few minutes. Second: "transient v. stationary" events. This category is based on the understanding of whether an event will change over time or not. For example, the sentence "he was born in the U.S." contains a stationary event since it will last forever; however, "he is hungry" contains a transient event since it will remain true for a short period of time. Third: "event ordering" which is the understanding of how events are usually ordered in nature. For example, "earning money" usually comes before "spending money". Fourth one is "absolute timepoint". This category deals with the understanding of when events usually happen. For example, "going to school" usually happens during the day (not at 2 A.M). The last category is "frequency" which refers to how often an event is likely to be repeated. For example, "taking showers" typically occurs 5 times a week, "going to saturday market" usually happens every few weeks/months, etc.
	GPT-2 XL	going to school . Indicate with ‘ No ‘, otherwise . We define five categories of temporal reasoning . First: "event duration" which is defined as the understanding of how long events last . For example, "brushing teeth", takes few minutes . Second: "transient v. stationary" events . This category is based on the understanding of whether an event will change over time or not . For example, the sentence "he was born in the U.S." contains a stationary event since it will last forever; however, "he is hungry" contains a transient event since it will remain true for a short period of time . Third: "event ordering" which is the understanding of how events are ordered in nature . For example, "earning money" comes before "spending money". Fourth one is "absolute timepoint". This category deals with the understanding of when events happen . For example, "going to school" happens during the day (not at 2 A.M). The last category is "frequency" which refers to how often an event is likely to be repeated . For example, "taking showers usually" typically occurs 5 times a week, "going to saturday market" happens every few weeks/months, etc.
	InstructGPT `babbage`	Indicate with ‘ Yes ‘ if the given question involves the provided reasoning ‘ Category ‘ . Indicate with ‘ No ‘, otherwise . We define five categories of temporal reasoning . First: "event duration" which is defined as the understanding of how long events last . For example, "First", takes few minutes . Second: "transient v. stationary" events . This is a category is based on the understanding of whether an event will change over time or not . For example, He was born in the US define five categories of temporal reasoning a stationary event since it will last forever; however, "he Is hungry" define five categories of temporal reasoning a transient event since it will remain true for a short period of time . Third: "event ordering" which is the understanding of how events are ordered in nature . For example, "earning money" comes before "spending money". Fourth one is "absolute timepoint". This is a category deals with the understanding of when events happen . For example, "going to school" happens during the day (not at 2 A.M). The last category is "frequency" which refers to how often an event is likely to be repeated . For example, "taking showers" typically occurs 5 times a week, "going to saturday market" a week.
	Instruct GPT `curie`	Indicate with ‘ Yes ‘ if the given question involves the provided reasoning ‘ Category ‘ . Indicate with ‘ No ‘, otherwise . We define five categories of temporal reasoning . First: "event duration" which is defined as the understanding of how long events last . For example, "brushing teeth", usually takes few minutes . Second: "transient v. stationary" events . This category is based on the understanding of whether an event will change over time or not . For example, the sentence "he was born in the U.S." contains a stationary event since it will last forever; however, "he is hungry" contains a transient event since it will remain true for a short period of time . Third: "event ordering" which is the understanding of how events are usually ordered in nature . For example, "earning money" usually comes before "spending money". Fourth one is "absolute timepoint". This category deals with the understanding of when events usually happen . For example, "going to school" usually happens during the day (not at 2 A.M). is "frequency" which refers to how often an event is likely to be repeated . For example, "taking showers" typically occurs 5 times a week, "going to saturday market" usually happens every few weeks/months, etc.
022	Original	Read the given context and if the the context is inappropriate (e.g., pornographic) or nonsensical (e.g., cannot determine what happenings the context is about), indicate via "yes". Otherwise, respond via "no"
	GPT-2 XL	Read the given context and if the the context is inappropriate (e.g., pornographic) or nonsensical (e.g., Can’t decide what the context is about, indicate via "yes". Otherwise, respond via "no".
	InstructGPT `babbage`	Read the given context and e.g., pornographic) or nonsensical (e.g . (e.g., pornographic) or nonsensical (e.g., cannot determine what happenings the context is about), indicate via "yes". Otherwise, respond via "no".
	Instruct GPT `curie`	Read the given context and indicate via "yes (e.g., pornographic) or nonsensical (e.g., cannot determine what happenings the context is about), indicate via "yes". Otherwise, respond via "no".
050	Original	You are given a sentence and a question in the input. If information provided in the sentence is enough to answer the question, label "Yes", otherwise label "No".Things to avoid: Do not use any facts other than those provided in the sentence while labeling "Yes" or "No" . Emphasis & Caution: There are only 2 types of valid responses: Yes and No.
	GPT-2 XL	You are given a sentence and a question are given a sentence and a question . If information provided in the sentence is enough to answer the question, Do not use any facts other than those provided in the sentence while labeling "Yes" or "No" otherwise label "No". Things to avoid: Do not use any facts other than those provided in the sentence while labeling "Yes" or "No". Emphasis & Caution: There are only 2 types of valid responses: Yes and No.
	InstructGPT `babbage`	You are given a sentence and a question in the input . If information provided in the sentence is enough to answer the question, label "Yes", otherwise label "No". Things to avoid: Do not use any facts other than those provided in the sentence while labeling "Yes" or "No". Emphasis & Caution: There.
	InstructGPT `curie`	You are given a sentence and a question in the input . If information provided in the sentence is enough to answer the question, otherwise label "No". Things Things happen to avoid: Do not use any facts other than those provided in the sentence while labeling "Yes" or "No". Emphasis & Caution: There are only 2 types of valid responses: Yes and No.

Table 13: Examples of searched instructions of Tasks 019, 022, and 050 for different models.

DiscoScore: Evaluating Text Generation with BERT and Discourse Coherence

Wei Zhao[12] **Michael Strube**[1] **Steffen Eger**[3]
[1]Heidelberg Institute for Theoretical Studies [2]Technische Universität Darmstadt
www.h-its.org/research/nlp/
[3]NLLG, Faculty of Technology, Bielefeld University
{wei.zhao, michael.strube}@h-its.org
steffen.eger@uni-bielefeld.de
nl2g.github.io

Abstract

Recently, there has been a growing interest in designing text generation systems from a discourse coherence perspective, e.g., modeling the interdependence between sentences. Still, recent BERT-based evaluation metrics are weak in recognizing coherence, and thus are not reliable in a way to spot the discourse-level improvements of those text generation systems. In this work, we introduce DiscoScore, a parametrized discourse metric, which uses BERT to model discourse coherence from different perspectives, driven by Centering theory. Our experiments encompass 16 non-discourse and discourse metrics, including DiscoScore and popular coherence models, evaluated on summarization and document-level machine translation (MT). We find that (i) the majority of BERT-based metrics correlate much worse with human rated coherence than early discourse metrics, invented a decade ago; (ii) the recent state-of-the-art BARTScore is weak when operated at system level—which is particularly problematic as systems are typically compared in this manner. DiscoScore, in contrast, achieves strong system-level correlation with human ratings, not only in coherence but also in factual consistency and other aspects, and surpasses BARTScore by over 10 correlation points on average. Further, aiming to understand DiscoScore, we provide justifications to the importance of discourse coherence for evaluation metrics, and explain the superiority of one variant over another. Our code is available at `https://github.com/AIPHES/DiscoScore`.

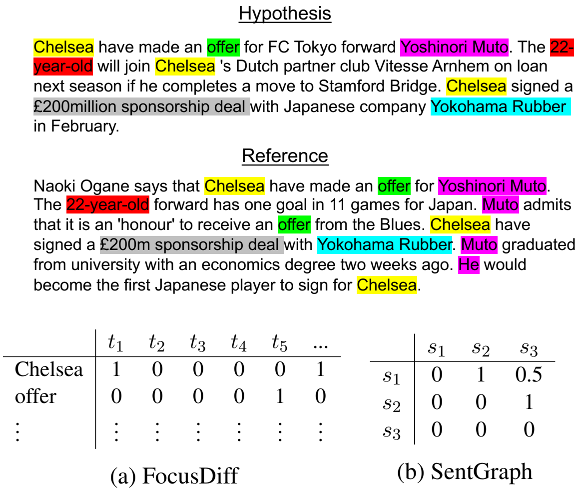
1 Introduction

In discourse, coherence refers to the continuity of semantics in text. Often, discourse relations and lexical cohesion devices, such as repetition and coreference, are employed to connect text spans, aiming to ensure text coherence. Popular theories in the linguistics community on discourse were provided by Grosz et al. (1995) and Mann and Thompson (1988). They formulate coherence through the lens of readers' focus of attention, and rhetorical discourse structures over sentences. Later on, coherence models as computational approaches of these theories emerged to judge text coherence in discourse tasks such as sentence ordering and essay scoring (Barzilay and Lapata, 2008; Lin et al., 2011; Guinaudeau and Strube, 2013).

While humans also often use text planning at discourse level prior to writing and speaking, up until recently, the majority of natural language generation (NLG) systems, be it text summarization or document-level MT, has performed sequential word prediction without considering text coherence. For instance, MT systems mostly do not model the interdependence between sentences and translate a document at sentence level, and thus produce many incoherent elements such as coreference mistakes in system outputs (Maruf et al., 2021). Only more recently has there been a surge of interest towards discourse based summarization and MT systems, aiming to model inter-sentence context, with a focus on pronominal anaphora (Voita et al., 2018; Liu et al., 2021) and discouse relations (Miculicich et al., 2018; Xu et al., 2020).

However, there appears a mismatch between discourse based NLG systems and non-discourse NLG evaluation metrics such as MoverScore (Zhao et al., 2019) and BERTScore (Zhang et al., 2020) which have recently become popular for MT and summarization evaluation. As these metrics base their judgment on semantic similarity (and lexical overlap (Kaster et al., 2021)) between hypotheses and references—which by design does not target text coherence—it is not surprising that they do not correlate well with human rated coherence (Fabbri et al., 2021; Yuan et al., 2021; Sai et al., 2021). Recently, BARTScore (Yuan et al., 2021) receives increasingly attention, which uses sequence-to-sequence language models to measure the likeli-

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3865–3883
May 2-6, 2023 ©2023 Association for Computational Linguistics



Hypothesis

Chelsea have made an offer for FC Tokyo forward Yoshinori Muto. The 22-year-old will join Chelsea 's Dutch partner club Vitesse Arnhem on loan next season if he completes a move to Stamford Bridge. Chelsea signed a £200million sponsorship deal with Japanese company Yokohama Rubber in February.

Reference

Naoki Ogane says that Chelsea have made an offer for Yoshinori Muto. The 22-year-old forward has one goal in 11 games for Japan. Muto admits that it is an 'honour' to receive an offer from the Blues. Chelsea have signed a £200m sponsorship deal with Yokohama Rubber. Muto graduated from university with an economics degree two weeks ago. He would become the first Japanese player to sign for Chelsea.

	t_1	t_2	t_3	t_4	t_5	...
Chelsea	1	0	0	0	0	1
offer	0	0	0	0	1	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮

(a) FocusDiff

	s_1	s_2	s_3
s_1	0	1	0.5
s_2	0	0	1
s_3	0	0	0

(b) SentGraph

Figure 1: Sample hypothesis and reference from SUMEval. Each focus[1] is marked in a different color, corresponding to multiple tokens as instances of a focus. Foci shared in Hypothesis and Reference are marked in the same color. (a)+(b) are adjacency matrices used to model focus-based coherence for Hypothesis; for simplicity, adjacency matrices for Reference are omitted. FocusDiff and SentGraph are the variants of DiscoScore. For FocusDiff, we use (a) to depict the relations between foci and tokens, reflecting focus frequency. For SentGraph, we use (b) to depict the interdependence between sentences according to the number of foci shared between sentences and the distance between sentences.

hood that hypothesis and reference are paraphrases, and that cannot contrast text pairs at discourse level.

In this work, we fill the gap of missing discourse metrics in MT and summarization evaluation, particularly in reference-based evaluation scenarios. We introduce DiscoScore, a parametrized discourse metric, which uses BERT to model discourse coherence through the lens of readers' focus, driven by Centering theory (Grosz et al., 1995). The DiscoScore variants can be distinguished in how we use *focus*—see Figure 1: (i) we model focus frequency and semantics, and compare their difference between hypothesis and reference and (ii) we use focus transitions to model the interdependence between sentences. Building upon this, we present a simple graph-based approach to compare hypothesis with reference.

We compare DiscoScore with a range of baselines, including discourse and non-discourse metrics, and coherence models on summarization and document-level MT datasets. Our contributions and findings are summarized as follows:

- Recent BERT-based metrics and the state-of-the-art BARTScore (Yuan et al., 2021) are all weak in system-level correlation with human ratings, not only in coherence but also in other aspects such as factual consistency. Most of them are even worse than very early discourse metrics, RC and LC (Wong and Kit, 2012)—which require neither source texts nor references and use discourse features to predict hypothesis coherence.
- DiscoScore strongly correlates with human rated coherence and many other aspects, over 10 points (on average across aspects) better than BARTScore and two strong baselines RC and LC in the single and multi-references settings. This indicates that either leveraging contextualized encoders or finding discourse features is not sufficient, suggesting to combine both as DiscoScore does.
- We demonstrate the importance of including discourse signals in the assessment of system outputs, as the discourse features derived from DiscoScore can strongly separate hypothesis from reference. Further, we show that the more discriminative these features are, the better the metrics perform, which allows for interpreting the performance gaps between the variants of DisoScore.
- We investigate two focus choices popular in the discourse community, i.e., noun (Elsner and Charniak, 2011) and semantic entity (Mesgar and Strube, 2016). Our results show that entity as focus is not always helpful, but when it helps, the gain is big.

2 Related work

Evaluation Metrics. Traditional metrics such as BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) measure lexical n-gram overlap between a hypothesis and a human reference. As they fail to measure semantic similarity in the absence of lexical overlap, several metrics have been proposed to overcome this issue, which carry out soft lexical matching with static word embeddings (Ng and Abrecht, 2015) and synonym matching (Lavie and Agarwal, 2007). However, none of those metrics

---

[1] The formal definition of focusing in discourse is given on two levels (Grosz et al., 1977): (i) readers are said to be *globally* focusing on a set of entities relevant to the overall discourse, and (ii) readers focus on a particular entity that an utterance *locally* concerns most. Section 3 elaborates on focus as a key ingredient of DiscoScore.

can properly judge text coherence (Kryscinski et al., 2019; Zhu and Bhat, 2020).

Recently, a class of novel metrics based on BERT (Devlin et al., 2019) has received a surge of attention, as they correlate strongly with human judgment of text quality in both reference-based and reference-free scenarios (Zhao et al., 2019; Zhang et al., 2020; Sellam et al., 2020; Rei et al., 2020; Gao et al., 2020; Thompson and Post, 2020; Zhao et al., 2020; Pu et al., 2021; Chen et al., 2021). While strong at sentence-level, these metrics are weak in recognizing coherence in inter-sentence contexts (just like BLEU and ROUGE), as BERT and the majority of BERT variants[2] that these metrics build on only capture discourse phenomena to a certain extent (Koto et al., 2021; Laban et al., 2021; Beyer et al., 2021). Thus, they are not suitable for evaluating long texts as in document-level MT evaluation. Works that either (i) average sentence-level evaluation scores as document score or (ii) assign a score to the concatenation of sentences within a document (Xiong et al., 2019; Liu et al., 2020; Saunders et al., 2020) do not factor interdependence between sentences into a document score, e.g., do not explicitly punish incoherent elements, thus are also inadequate.

Several attempts have been made towards discourse metrics in MT evaluation. Wong and Kit (2012); Gong et al. (2015); Cartoni et al. (2018) use the frequency of lexical cohesion devices (e.g., word repetition) over sentences to predict coherence of hypothesis translations, while Guzmán et al. (2014) and Joty et al. (2017) suggest to compare the difference of rhetorical structures between hypothesis and reference translations. Recently, Jiang et al. (2021) measure the inconsistency between hypothesis and reference translations in several aspects such as verb tense and named entities. However, these metrics do not leverage strong contextualized encoders, as has been shown to be a key ingredient for recent success of BERT-based metrics. Most recently, BARTScore (Yuan et al., 2021) uses sequence-to-sequence pretrained language models such as BART (Lewis et al., 2020) to measure how likely hypothesis and reference are paraphrased according to the probability of one given the other. While BARTScore constitutes the recent state-of-the-art in sentence-level correlation with human ratings in several aspects (incl. discourse), we find that (i) it performs still poorly at system level—which is particularly problematic as systems are typically compared in this manner. (ii) As based on a 'blackbox' language model, it cannot offer insights towards how it models coherence and what discourse phenomena it does (not) capture.

Coherence Models. In discourse, there have been many computational models (Barzilay and Lapata, 2008; Guinaudeau and Strube, 2013; Pitler and Nenkova, 2008; Lin et al., 2011) for text coherence assessment, the majority of which differ in *regularities* that they use to distinguish coherent from incoherent text, driven by different linguistic theories, *v.i.z.*, a pattern of (i) focus transitions in adjacent sentences (Grosz et al., 1995) and (ii) text organization regarding discourse relations over sentences (Mann and Thompson, 1988). For instance, Barzilay and Lapata (2008) and Guinaudeau and Strube (2013) use the distribution of entity transitions over sentences to predict text coherence, while Pitler and Nenkova (2008) and Lin et al. (2011) suggest to produce discourse relations over sentences with a discourse parser, showing that the relations are indicative of text coherence. In the last few years, neural coherence models have been explored. Popular examples are Tien Nguyen and Joty (2017), Mesgar and Strube (2018) and Moon et al. (2019). As they and the recent state-of-the-art (Mesgar et al., 2021) all have been trained on text readability datasets, with readability labels as supervision, they may suffer issues of domain shift when applied to MT and summarization evaluation. More importantly, they judge hypothesis coherence in the absence of reference, thus are not sufficient for reference-based evaluation. Our experiments involve two popular, unsupervised coherence models, entity graph (Guinaudeau and Strube, 2013) and lexical graph (Mesgar and Strube, 2016) treated as discourse metrics with the advantages on robustness (Lai and Tetreault, 2018).

Discourse Test Sets. Apart from evaluation metrics, there have been several discourse-focused test sets proposed to compare NLG systems, most of which have been studied in MT evaluation. For instance, the DiscoMT15 shared task (Hardmeier et al., 2015) compares MT systems, not based on translation adequacy but on the accuracy of pronoun translation for English-to-French, i.e., counting the number of correctly translated pronouns, given the annotated ones in reference. Bawden

---

[2] Recently, several discourse BERT variants such as Conpono (Iter et al., 2020) have been proposed, but they are not always helpful for evaluation metrics—see Table 2 (appendix).

et al. (2018) extend this by labeling both anaphoric pronouns and lexical cohesion devices on test sets, while Voita et al. (2018) construct English-to-Russian test sets focusing on deixis, ellipsis and lexical cohesion. Guillou et al. (2018); Lopes et al. (2020) construct English-to-German and English-to-French test sets targeting pronouns. While reliable, these test sets involve costly manual annotation, thus are limited to few language pairs.

In this work, we introduce DiscoScore to judge system outputs, which uses BERT to model readers' focus within hypothesis and reference, and thus clearly outlines the discourse phenomena being captured, serving as low-cost alternatives to discourse test sets for comparing discourse based NLG systems. More prominently, we derive discourse features from DiscoScore, which we use to understand the importance of discourse for evaluation metrics, and explain why one metric is superior to another. This parallels recent effort towards explainability for non-discourse evaluation metrics (Kaster et al., 2021; Fomicheva et al., 2021). Finally, we show that simple features can be indicative of the superiority of a metric, which fosters research towards finding insightful features with domain expertise and building upon these insights to design high-quality metrics.

3 Our Approach

In the following, we elaborate on the two variants of DiscoScore, FocusDiff and SentGraph, which we refer to as DS-FOCUS and DS-SENT.

Focus Difference. In discourse, there have been many corpus-based studies towards modeling focus transitions over sentences, showing that focus transition patterns are indicative of text coherence (Barzilay and Lapata, 2008; Guinaudeau and Strube, 2013). When reading a document, readers may have multiple *focus of attention*,

each associated to a group of expressions: (i) referring expressions such as pronouns and (ii) semantically related elements such as [*Berlin, capital*].

Here, we assume two focus based conditions that a coherent hypothesis should meet in reference-based evaluation scenarios:

- A large number of focus overlaps between a hypothesis and a reference.
- Each focus overlap is nearly identical in terms of semantics and frequency, where frequency shows how often a focus is mentioned in a hypothesis or in a reference.

In the following, we present focus modeling towards semantics and frequency, according to which we compare hypothesis with reference.

For a hypothesis, we introduce a bipartite graph $\mathcal{G}^{\text{hyp}} = (\mathcal{V}, \mathcal{S}, \mathbf{A}^{\text{hyp}})$, where $\mathcal{V}$ and $\mathcal{S}$ are two sets of vertices corresponding to a set of foci and all tokens (per occurrence a word is a separate token) within a hypothesis. Let $\mathbf{A} = \{0,1\}^{n \times m}$ be an adjacency matrix where n and m are the number of foci and tokens respectively, and A_{ij} equals 1 if and only if the i-th focus associates to the j-th token. Let $\mathbf{F}^{\text{hyp}} \in \mathbb{R}^{n \times d}$ be a matrix of focus embeddings and $\mathbf{Z}^{\text{hyp}} \in \mathbb{R}^{m \times d}$ be a matrix of contextualized token embeddings with d as the embedding size. Similarly, we use notation $\mathcal{G}^{\text{ref}}$, $\mathbf{F}^{\text{ref}}$ and $\mathbf{Z}^{\text{ref}}$ for a human reference.

We use contextualized encoders such as BERT to produce token embeddings $\mathbf{Z}^{\text{hyp}}$ and $\mathbf{Z}^{\text{ref}}$. We use a simple approach to model both semantics and frequency of a focus. That is, we assign per focus v an embedding by summing token embeddings that a focus is associated to:

$$\mathbf{F}_v^{\text{hyp}} = \sum_{u \in \mathcal{N}(v)} \mathbf{Z}_u^{\text{hyp}}, \ \mathbf{F}_v^{\text{ref}} = \sum_{u \in \mathcal{N}(v)} \mathbf{Z}_u^{\text{ref}} \tag{1}$$

where $\mathcal{N}(v)$ is a set of tokens (e.g., a group of semantically related expressions) associated with a focus v. In matrix notation, we rewrite Eq. (1) to $\mathbf{F}^{\text{hyp}} = \mathbf{A}^{\text{hyp}}\mathbf{Z}^{\text{hyp}}$, similarly for $\mathbf{F}^{\text{ref}}$.

Next, we measure the distance between a common set of foci Ω in a hypothesis and reference pair based on their embeddings:

$$\text{DS-F}\text{OCUS}(\text{hyp}, \text{ref}) = \frac{1}{\text{N}} \sum_{\text{u} \in \Omega} \|\mathbf{F}_{\text{u}}^{\text{hyp}} - \mathbf{F}_{\text{u}}^{\text{ref}}\| \tag{2}$$

where DS-FOCUS is scaled down by the factor of N, the number of foci in hypothesis.

Sentence Graph. Few contextualized encoders can produce high-quality sentence embeddings in the document context, as they do not model interdependence between sentences. According to Centering theory (Grosz et al., 1995), two sentences are marked continuous in meaning when they share at least one focus, on the one hand; one marks a meaning shift for two sentences when no focus appears in common, on the other hand. From this, one can aggregate sentence embeddings for which corresponding sentences are considered continuous. In the following, we present a graph-based approach to do so.

For a hypothesis[3], let $\mathbf{S}^{\text{hyp}} \in \mathbb{R}^{n \times d}$ be a matrix of sentence embeddings with n and d as the number of sentences and the embedding size. We introduce a graph $\mathcal{G}^{\text{hyp}} = (\mathcal{V}, \mathbf{A}^{\text{hyp}})$ where $\mathcal{V}$ is a set of sentences and $\mathbf{A}^{\text{hyp}}$ is an adjacency matrix weighted according to the number of foci shared between sentences and the distance between sentences as listed below to depict two variants of $\mathbf{A}^{\text{hyp}}$:

- unweighted: $\mathbf{A}^{\text{hyp}}_{ij} = 1/(j-i)$ if the i-th and the j-th sentences have at least one focus in common (otherwise 0), where $j-i$ denotes the distance between two sentences and $\mathbf{A}^{\text{hyp}}_{ij} = 0$ when $j \leq i$.
- weighted: $\mathbf{A}^{\text{hyp}}_{ij} = a/(j-i)$, where a is the number of foci shared in the i-th and the j-th sentences, with the same constraints on j and i as above.

Analyses by Guinaudeau and Strube (2013) indicate that global statistics (e.g., average) over such adjacency matrices can distinguish incoherent from coherent text to some degree. Here we depict adjacency matrices as a form of sentence connectivity derived from focus transitions over sentences. We use them to aggregate sentence embeddings from hypothesis and from reference:

$$\hat{\mathbf{S}}^{\text{hyp}} = (\mathbf{A}^{\text{hyp}} + \mathbf{I})\mathbf{S}^{\text{hyp}}, \ \hat{\mathbf{S}}^{\text{ref}} = (\mathbf{A}^{\text{ref}} + \mathbf{I})\mathbf{S}^{\text{ref}}$$

where $\mathbf{I}$ is an identity matrix that adds a self-loop to a graph so as to include self-embeddings when updating them.

Next, we derive per graph an embedding with simple statistics from $\hat{\mathbf{S}}^{\text{hyp}}$ and $\hat{\mathbf{S}}^{\text{ref}}$, i.e., the concatenation of mean-max-min-sum embeddings. Finally, we compute the cosine similarity between two graph-level embeddings:

$$\text{DS-SENT}(\text{hyp}, \text{ref}) = \text{cosine}(\mathcal{G}^{\text{hyp}}, \mathcal{G}^{\text{ref}}) \quad (3)$$

Choice of Focus. In discourse, often four popular choices are used to describe a focus: (i) a noun that heads a NP (Barzilay and Lapata, 2008), (ii) a noun (Elsner and Charniak, 2011), (iii) a coreferent entity associated with a set of referring expressions (Guinaudeau and Strube, 2013) and (iv) a semantic entity associated with a set of lexical related words (Mesgar and Strube, 2016).

In this work, we investigate two focus choices: noun (NN) and semantic entity (Entity). Linguistically speaking, the latter is a lexical cohesion device in the form of repetition. From this, NN as focus yields few useful coherence signals but a lot of noise, while Entity as focus uses 'signal compression' by means of aggregation to produce better signals. To produce entities, we first extract all nouns in hypothesis (or reference), and aggregate them into different semantic entities if their cosine similarities based on Dep2Vec word embeddings (Levy and Goldberg, 2014) is greater than a threshold—assuming that nouns with high similarity refer to the same semantic entity.

4 Experiments

4.1 Evaluation Metrics

In the following, we list all of the evaluation metrics, and elaborate on them in Appendix A.1.

Non-discourse Metrics. We consider BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), BERTScore (Zhang et al., 2020), MoverScore (Zhao et al., 2019), SBERT (Reimers and Gurevych, 2019), S^3-pyr (Peyrard et al., 2017), BLEURT (Sellam et al., 2020), BARTScore (Yuan et al., 2021), PRISM (Thompson and Post, 2020).

Discourse Metrics. We consider RC and LC (Wong and Kit, 2012) and Lexical Chain (Gong et al., 2015). We consider two coherence models, EntityGraph (Guinaudeau and Strube, 2013) and LexicalGraph (Mesgar and Strube, 2016), and treat them as discourse metrics.

DiscoScore. DS-FOCUS can be parameterized with two focus choices: noun (NN) or semantic entity (Entity). DS-SENT can be parameterized not only with focus, but also with the choices of *unweighted* (-U) and *weighted* (-W). For DS-FOCUS, we use Conpono (Iter et al., 2020) that finetuned BERT with a novel discourse-level objective regarding sentence ordering. For DS-SENT, we use BERT-NLI. This is because we find this configuration performs best after initial trials—see Table 2 (appendix). Figure 5 (appendix) shows all variants of DiscoScore. Concerning the threshold of Dep2Vec to produce entities, after experimenting with several alternatives we set it to 0.8 for DS-FOCUS (Entity) in all setups, and to 0.8 in summarization and to 0.5 in MT for DS-SENT (Entity).

---

[3] For simplicity, we omit the notation $\mathbf{S}^{\text{ref}}$ and $\mathcal{G}^{\text{ref}}$ for a reference.

4.2 Datasets

We consider two datasets in summarization: SummEval (Fabbri et al., 2021) and NeR18 (Grusky et al., 2018), and one dataset in document-level MT: WMT20 (Mathur et al., 2020). Note that these datasets consist of hypotheses paired with human-written references, where hypotheses are machine-generated texts of varying qualities given by neural and non-neural, extractive and abstractive language models. We outline these datasets in Appendix A.2, and provide data statistics in Table 9 (appendix).

5 Results

We first examine the importance of discourse for evaluation metrics—which underpins the usefulness of discourse metrics, and then benchmark DiscoScore on summarization and MT datasets.

Importance of Discourse. DS-Focus and DS-Sent concern the modeling of discourse coherence on two different levels: (i) the occurrences of foci, and (ii) the interdependence between sentences driven by focus transitions, both reflecting the discourse characteristics of a text. In the following, we describe these discourse features, and examine their importance for assessing system outputs by contrasting the discourse patterns of hypothesis and reference.

- **Focus Frequency**, denoted by FREQ(x), equals the ratio between the total frequencies of foci and the number of foci in a text x, where x is hypothesis or reference. We exclude foci occurring only once.
- **Sentence Connectivity**, denoted by Conn(x), equals the average of all elements in adjacency matrix representing the interdependence between sentences in a text x (hypothesis/reference).
- As in DiscoScore, we consider two focus choices (NN and Entity) and the choices of *unweighted* (-U) and *weighted* (-W) for these discourse features. Figure 5 (appendix) shows the links between DiscoScore and the features.

Figure 2 shows that the scales on FREQ(ref) and FREQ(hyp) in summarization differ by a large amount, i.e., from 0.5 to 2.5 on y-axis and up to 6 on x-axis. This means that hypothesis and reference can be strongly distinguished by Freq(x), which underpins the usefulness of including such discourse signals in the assessment of system outputs when references are available. Further, the larger scale on FREQ(hyp) indicates that foci in hypothesis are more repetitive than in reference, as a result of needless repetition in poor summaries—in line with previous studies on incoherent machine translations (Guillou, 2013; Voita et al., 2019). The results for other discourse features are similar, we provide them in Figure 6 (appendix).

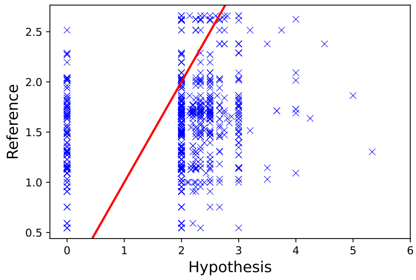


Figure 2: Scatter plot to display FREQ(hyp) (based on NN) on x-axis and FREQ(ref) on y-axis on SUMMEval. Each point contains two frequencies from a pair of hypothesis and reference. The points below the auxiliary line are the ones for which FREQ(hyp) > FREQ(ref).

Overall, these results show discourse features can separate hypothesis from reference.

5.1 Text Summarization

Correlation Results. Table 1 compares metrics on SUMMEval on system level. Most of non-discourse metrics have a lowest correlation with human rated coherence among four quality aspects. Even worse, ROUGE-L and SBERT do not correlate with coherence whatsoever. BARTScore, the recent state-of-the-art metric, is very weak when operated on system level, notwithstanding that it has been fine-tuned on "document-to-summary" parallel data from CNN/DailyMail—which SUMMEval is constructed from. We note that SUMMEval uses multiple references. BARTScore by default compares a hypothesis with one reference at a time, then takes the average of multiple evaluation scores as a final score. Table 8 (appendix) shows that we can improve system-level BARTScore to some degree by replacing 'average' with 'max' (i.e., taking the maximum score), but DS-Focus is still much better overall, i.e., surpassing BARTScore by ca. 10 points on average.

Table 7 (appendix) reports correlation results on NeR18 that uses single reference. We find that half of hypotheses do not contain 'good foci', and as such the foci-based discourse features outlined

Settings	Metrics	Coherence	Consistency	Fluency	Relevance	Average
	Non-discourse metrics					
$m(\text{hyp}, \text{ref})$	ROUGE-1	9.09	27.27	18.18	9.09	15.91
	ROUGE-L	0.00	36.36	21.21	18.18	18.94
	BERTScore	30.30	30.30	51.52	54.55	41.67
	MoverScore	36.36	42.42	63.64	60.61	50.76
	SBERT	3.03	33.33	30.30	27.27	23.48
	BLEURT	45.45	**51.52**	**72.73**	63.64	58.33
	BARTScore	60.61	36.36	45.45	48.48	47.73
	PRISM	51.52	39.39	**72.73**	69.70	58.33
	S^3-pyr	18.18	24.24	9.09	6.06	14.39
	Discourse metrics					
$m(\text{hyp})$	RC	45.45	**51.52**	**54.55**	57.58	**52.27**
	LC	**51.52**	45.45	48.48	57.58	50.76
	Entity Graph	42.42	12.12	15.15	18.18	21.97
	Lexical Graph	48.48	6.06	15.15	18.18	21.97
$m(\text{hyp}, \text{ref})$	Lexical Chain	42.42	6.06	9.09	18.18	18.94
	DS-Focus (NN)	**75.76**	**63.64**	**78.79**	**81.82**	**75.00**
	DS-Focus (Entity)	69.70	57.58	72.73	75.76	68.94
	DS-Sent-u (NN)	48.48	54.55	63.64	60.61	56.82
	DS-Sent-u (Entity)	54.55	60.61	75.76	66.67	64.39
	DS-Sent-w (NN)	51.52	51.52	66.67	63.64	58.33
	DS-Sent-w (Entity)	51.52	57.58	66.67	63.64	59.85

Table 1: System-level Kendall correlations between metrics and human ratings of summary quality on SUMMEval. We bold numbers that significantly outperform others according to paired t-test (Fisher et al., 1937). m is a metric.

previously are less discriminative on NeR18 than on SUMMEval—see Table 9 (appendix). However, DS-Focus is still strong, ca. 20 points better than BARTScore in all aspects, despite that DS-Focus uses a much smaller contextualized encoder[4]. We note that the 'F-score' version of DS-Focus seems extremely strong on NeR18, but it is not robust across datasets, e.g., much worse than the original, precision-based DS-Focus on SUMMEval.

On a side note, coherence (mostly) strongly correlates with the other rating aspects on both SUMMEval and NeR18—see Figure 3. Thus, it is not surprising that both DS-Focus and DS-Sent correlate well with these aspects, despite that we have not targeted them. While strong on system level, DiscoScore could not show advantages on summary level—see Table 5 (appendix), but we argue that system-level correlation deserves the highest priority as systems are compared in this manner.

Overall, these results show that BERT-based non-discourse metrics correlate weakly with human ratings on system level. BARTScore also does so, though we improve it to some degree in multi-references settings. DiscoScore, particularly DS-Focus, performs consistently best in both single- and multi-references settings, and it is equally strong in all aspects.

As for discourse metrics, RC and LC that use discourse features are strong baselines as they outperform most of non-discourse metrics and coherence models (i.e., Entity and Lexical Graph) without the access to source texts and references. However, they are worse than both DS-Focus and DS-Sent. This confirms the inadequacy of RC and LC in that they do not leverage strong contextualized encoders and judge hypothesis in the absence of references. Moreover, we compare DiscoScore to a combination of two strong, complementary baselines, BARTScore and RC—a simple solution to address text coherence of non-discourse metrics. To combine them, we simply average their scores. We see the gains are additive in all aspects but coherence. DS-Focus wins all the time by a large margin—see Table 10 (appendix).

Taken together, these results show that any of the three—(i) leveraging contextualized encoders as in BERT-based metrics and BARTScore; (ii) leveraging discourse features as in RC and (iii) the ensemble of (i) and (ii) by averaging—is not sufficient, suggesting to combine (i) and (ii) as DiscoScore does.

Understanding DiscoScore. As for all variants of DiscoScore, we provide understanding on why

---

[4] DS-Focus uses Conpono on the same size of BERTBase. BARTScore uses BARTLarge finetuned on CNN/DailyMail.

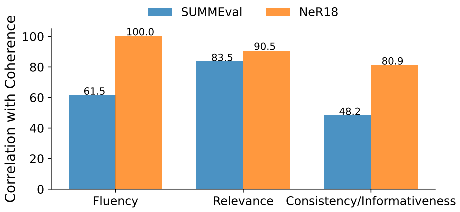


Figure 3: Pearson Correlation between coherence and other aspects on system level. SUMMEval and NeR18 use Consistency and Informativeness respectively.

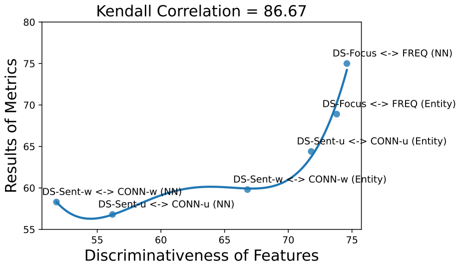


Figure 4: Correlations between the results of metrics and the discriminativeness of features on SUMMEval. Metric results are averaged across four rating aspects.

one variant is superior to another with the discourse features outlined in Figure 5 (appendix). To this end, we begin with defining the *discriminativeness* of these features as the magnitude of separating hypothesis from reference:

$$\mathcal{D}_{\mathcal{R}}(\text{hyp}, \text{ref}) := \frac{|\{(\text{hyp}, \text{ref})|\mathcal{R}(\text{ref}) < \mathcal{R}(\text{hyp})\}|}{N} \tag{4}$$

where N is a normalization term, $\mathcal{R}$ is any one of the discourse features in Figure 5 (appendix).

Figure 4 shows that the discriminativeness of these features strongly correlate with the results of the DiscoScore variants, i.e., that the more discriminative the features are, the better the metrics perform. This attributes the superiority of a metric to the fact that the discourse feature can better separate hypothesis and reference.

From this, we can interpret the performance gaps between the DiscoScore variants, namely (i) DS-Focus over DS-Sent: given *Focus Frequency* is more discriminative than *Sentence Connectivity*, it is not surprising that DS-Focus modeling discourse coherence with the former outperforms DS-Sent modeling with the latter, and (ii) DS-Focus (NN) outperforms DS-Focus (Entity) because *Frequency (NN)* can better separate hypothesis from reference than *Frequency (Entity)*.

Analyses. We provide analyses on the configuration of DiscoScore from three perspectives—see Appendix A.3: (i) the choice of BERT variants towards discourse- versus non-discourse BERT; (ii) the impact of adjacency matrices accounting for the interdependence between sentences and (iii) that we compare statistics- and alignment-based approaches to examine the best configuration for DS-Sent. Our results show the advantages of adjacency matrices and statistics based approach, and that discourse BERT only helps for DS-Focus.

5.2 Document-level Machine Translation

Correlation Results. Table 12 (appendix) compares metrics on WMT20. We see that non-discourse metrics seem much better, but these results are not consistent to the discriminativeness of the discourse features—see Table 11 (appendix). For instance, in cs-en, the discourse features (Frequency and Connectivity) corresponding to DS-Focus and DS-Sent clearly separate hypothesis from reference due to the probability of $\mathcal{D} > 0$ being over 70%. However, both DS-Focus and DS-Sent correlate weakly with human rated adequacy. Recently, Freitag et al. (2021a) provide justification to the inadequacy of the 'adequacy' ratings, as 'adequacy' sometimes cannot distinguish human from system translations and correlates weakly with multiple aspects (e.g., fluency and accuracy). Thus, they re-annotate WMT20 with the MQM and pSQM rating schemes, which has been subsumed into the annotation guideline of the most recent WMT evaluation campaign (Freitag et al., 2021b). Here, we perform an extra study on these ratings on both document- and system-levels. Note that system-level ratings are said to be the average of document-level ones in our setting. Table 6 (appendix) shows that DS-Sent is much better than BARTScore on system level, surpassing it by 25 points in terms of MQM and 14 points in pSQM.

Overall, these results in MT are consistent with those in summarization, i.e., DiscoScore is strong on system levels for both tasks, but it cannot show gains on fine-grained levels. Section A.4 (appendix) show inter-correlations between metrics.

6 Conclusions

Given the recent growth in discourse based NLG systems, evaluation metrics targeting the assessment of text coherence are essential next steps for properly tracking the progress of these systems.

Although there have been several attempts made towards discourse metrics, they all do not leverage strong contextualized encoders which have been held responsible for the recent success story of NLP. In this work, we introduced DiscoScore that uses BERT to model discourse coherence from two perspectives of readers' focus: (i) frequencies and semantics of foci and (ii) focus transitions over sentences used to predict interdependence between sentences. We find that BERT-based non-discourse metrics cannot address text coherence, even much worse than early feature-based discourse metrics invented a decade ago. We also find that the recent state-of-the-art BARTScore correlates weakly with human ratings on system level. DiscoScore, on the other hand, performs consistently best in both single- and multi-reference settings, equally strong in coherence and several other aspects such as factual consistency, despite that we have not targeted them. More prominently, we provide understanding on the importance of discourse for evaluation metrics, and explain the superiority of one metric over another with simple features, in line with recent work on explainability for evaluation metrics (Kaster et al., 2021; Fomicheva et al., 2021).

Scope for future research is huge, e.g., developing reference-free discourse metrics comparing source text to hypothesis, improving discourse metrics on fine-grained levels[5], and ranking NLG systems via discourse metrics and rigorous approaches (Peyrard et al., 2021; Kocmi et al., 2021).

7 Impact and Limitations

To our knowledge, we, for the first time, combine the elements of discourse and BERT representations to design an evaluation metric (DiscoScore) for text quality assessment in summarization and MT. While our experiments are conducted on English datasets, DiscoScore could adapt to many other languages in which references and foci are available. We believe that this work fosters future research on text generation systems endowed with the ability to produce well-formed texts in discourse.

However, we acknowledge several limitations of this work, which require further investigation in future. We now discuss them in the following:

Entity as Focus. We follow the idea of Mesgar and Strube (2016) in the discourse community, which clusters nouns into entities based on their static word embeddings. Although simple, it sometimes helps for DiscoScore. However, alternatives aiming to produce better entities have not been explored in this work, e.g., replacing static with contextualized embeddings, and weighting entities by their occurrences in hypothesis/reference.

Weakness on Fine-Grained Assessment. In summarization and MT, we show that our novel DiscoScore largely outperforms the current state-of-the-art BARTScore on system levels for both tasks, while it cannot show advantages on finer-grained levels such as document- and summary-levels. This might be because modeling focus alone is insufficient to perform much more challenging, finer-grained assessment of text quality. Future work could also factor other discourse phenomena (e.g., discourse connectives and coreference) into the assessment of text coherence.

Acknowledgments

We thank the anonymous reviewers for their thoughtful comments that greatly improved the texts. This work has been supported by the German Research Foundation as part of the Research Training Group Adaptive Preparation of Information from Heterogeneous Sources (AIPHES) at the Technische Universität Darmstadt under grant No. GRK 1994/1 and the Klaus Tschira Foundation, Heidelberg, Germany. Steffen Eger is funded by DFG Heisenberg grant EG 375/5-1.

References

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

---

[5] Recently, Steen and Markert (2022) introduce a fine-grained evaluation setup to compute summary-level correlation, which performs computing over summaries not produced by multiple systems, but rather by a single system. This is because systems sometimes substantially differ in quality, which implies that involving multiple systems could result in inaccurate evaluation outcomes in the presence of system-level confounders.

Anne Beyer, Sharid Loáiciga, and David Schlangen. 2021. Is incoherence surprising? targeted evaluation of coherence prediction from language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4164–4173, Online. Association for Computational Linguistics.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Bruno Cartoni, Jindřich Libovický, and Thomas Brovelli (Meyer), editors. 2018. *Machine Translation Evaluation beyond the Sentence Level*. Alicante, Spain.

Mingda Chen, Zewei Chu, and Kevin Gimpel. 2019. Evaluation benchmarks and learning criteria for discourse-aware sentence representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 649–662, Hong Kong, China. Association for Computational Linguistics.

Wang Chen, Piji Li, and Irwin King. 2021. A training-free and reference-free summarization evaluation metric via centrality-weighted relevance and self-referenced redundancy. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 404–414, Online. Association for Computational Linguistics.

Elisabet Comelles, Jesús Giménez, Lluís Màrquez, Irene Castellón, and Victoria Arranz. 2010. Document-level automatic MT evaluation based on discourse representations. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 333–338, Uppsala, Sweden. Association for Computational Linguistics.

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

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Micha Elsner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 125–129.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Ronald Aylmer Fisher et al. 1937. The design of experiments. *The design of experiments.*, (2nd Ed).

Marina Fomicheva, Piyawat Lertvittayakumjorn, Wei Zhao, Steffen Eger, and Yang Gao. 2021. The Eval4NLP shared task on explainable quality estimation: Overview and results. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 165–178, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021a. Experts, Errors, and Context: A Large-Scale Study of Human Evaluation for Machine Translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, George Foster, Alon Lavie, and Ondřej Bojar. 2021b. Results of the WMT21 metrics shared task: Evaluating metrics with expert-based human evaluations on TED and news domain. In *Proceedings of the Sixth Conference on Machine Translation*, pages 733–774, Online. Association for Computational Linguistics.

Yang Gao, Wei Zhao, and Steffen Eger. 2020. SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

Zhengxian Gong, Min Zhang, and Guodong Zhou. 2015. Document-level machine translation evaluation with gist consistency and text cohesion. In *Proceedings of the Second Workshop on Discourse in Machine Translation*, pages 33–40.

Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Barbara J Grosz et al. 1977. The representation and use of focus in a system for understanding dialogs. In *IJCAI*, volume 67, page 76. Citeseer.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the*

2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Liane Guillou. 2013. Analysing lexical consistency in translation. In *Proceedings of the Workshop on Discourse in Machine Translation*, pages 10–18, Sofia, Bulgaria. Association for Computational Linguistics.

Liane Guillou, Christian Hardmeier, Ekaterina Lapshinova-Koltunski, and Sharid Loáiciga. 2018. A pronoun test suite evaluation of the English–German MT systems at WMT 2018. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 570–577, Belgium, Brussels. Association for Computational Linguistics.

Camille Guinaudeau and Michael Strube. 2013. Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 93–103, Sofia, Bulgaria. Association for Computational Linguistics.

Francisco Guzmán, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2014. Using discourse structure improves machine translation evaluation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 687–698, Baltimore, Maryland. Association for Computational Linguistics.

Christian Hardmeier, Preslav Nakov, Sara Stymne, Jörg Tiedemann, Yannick Versley, and Mauro Cettolo. 2015. Pronoun-focused MT and cross-lingual pronoun prediction: Findings of the 2015 DiscoMT shared task on pronoun translation. In *Proceedings of the Second Workshop on Discourse in Machine Translation*, pages 1–16, Lisbon, Portugal. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Dan Iter, Kelvin Guu, Larry Lansing, and Dan Jurafsky. 2020. Pretraining with contrastive sentence objectives improves discourse performance of language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4859–4870, Online. Association for Computational Linguistics.

Yuchen Jiang, Shuming Ma, Dongdong Zhang, Jian Yang, Haoyang Huang, and Ming Zhou. 2021. Blond: An automatic evaluation metric for document-level machinetranslation. *CoRR*, abs/2103.11878.

Shafiq Joty, Francisco Guzmán, Lluís Màrquez, and Preslav Nakov. 2017. Discourse structure in machine translation evaluation. *Computational Linguistics*, 43(4):683–722.

Marvin Kaster, Wei Zhao, and Steffen Eger. 2021. Global explainability of BERT-based evaluation metrics by disentangling along linguistic factors. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8912–8925, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. *CoRR*, abs/2107.10821.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3849–3864, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.

Philippe Laban, Luke Dai, Lucas Bandarkar, and Marti A. Hearst. 2021. Can transformer models measure coherence in text: Re-thinking the shuffle test. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1058–1064, Online. Association for Computational Linguistics.

Alice Lai and Joel Tetreault. 2018. Discourse coherence in the wild: A dataset, evaluation and methods. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 214–223, Melbourne, Australia. Association for Computational Linguistics.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd*

Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 302–308.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of summaries. In *Proceedings of ACL workshop on Text Summarization Branches Out*, pages 74–81, Barcelona, Spain.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 997–1006, Portland, Oregon, USA. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Zhengyuan Liu, Ke Shi, and Nancy Chen. 2021. Coreference-aware dialogue summarization. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 509–519, Singapore and Online. Association for Computational Linguistics.

António Lopes, M. Amin Farajian, Rachel Bawden, Michael Zhang, and André F. T. Martins. 2020. Document-level neural MT: A systematic comparison. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 225–234, Lisboa, Portugal. European Association for Machine Translation.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Sameen Maruf, Fahimeh Saleh, and Gholamreza Haffari. 2021. A survey on document-level neural machine translation: Methods and evaluation. *ACM Computing Surveys (CSUR)*, 54(2):1–36.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020. Results of the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 688–725, Online. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan Thomas Mcdonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online.

Mohsen Mesgar, Leonardo F. R. Ribeiro, and Iryna Gurevych. 2021. A neural graph-based local coherence model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2316–2321, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mohsen Mesgar and Michael Strube. 2016. Lexical coherence graph modeling using word embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1414–1423, San Diego, California. Association for Computational Linguistics.

Mohsen Mesgar and Michael Strube. 2018. A neural local coherence model for text quality assessment. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4328–4339, Brussels, Belgium. Association for Computational Linguistics.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954, Brussels, Belgium. Association for Computational Linguistics.

Han Cheol Moon, Tasnim Mohiuddin, Shafiq Joty, and Chi Xu. 2019. A unified neural coherence model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2262–2272, Hong Kong, China. Association for Computational Linguistics.

Jun-Ping Ng and Viktoria Abrecht. 2015. Better summarization evaluation with word embeddings for ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1925–1930, Lisbon, Portugal. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.

Maxime Peyrard, Wei Zhao, Steffen Eger, and Robert West. 2021. Better than average: Paired evaluation of NLP systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2301–2315, Online. Association for Computational Linguistics.

Emily Pitler and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 186–195, Honolulu, Hawaii. Association for Computational Linguistics.

Amy Pu, Hyung Won Chung, Ankur Parikh, Sebastian Gehrmann, and Thibault Sellam. 2021. Learning compact metrics for MT. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 751–762, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Ananya B. Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M. Khapra. 2021. Perturbation CheckLists for evaluating NLG evaluation metrics. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7219–7234, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Danielle Saunders, Felix Stahlberg, and Bill Byrne. 2020. Using context in neural machine translation training objectives. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7764–7770, Online. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Julius Steen and Katja Markert. 2022. How to find strong summary coherence measures? a toolbox and a comparative study for summary coherence measure evaluation. *arXiv preprint arXiv:2209.06517*.

Brian Thompson and Matt Post. 2020. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–121, Online. Association for Computational Linguistics.

Dat Tien Nguyen and Shafiq Joty. 2017. A neural local coherence model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1320–1330, Vancouver, Canada. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Billy T. M. Wong and Chunyu Kit. 2012. Extending machine translation evaluation metrics with lexical cohesion to document level. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1060–1068, Jeju Island, Korea. Association for Computational Linguistics.

Hao Xiong, Zhongjun He, Hua Wu, and Haifeng Wang. 2019. Modeling coherence for discourse neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7338–7345.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. BARTScore: Evaluating generated text as text generation. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020,* *Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Wei Zhao, Goran Glavaš, Maxime Peyrard, Yang Gao, Robert West, and Steffen Eger. 2020. On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1656–1671, Online. Association for Computational Linguistics.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Wanzheng Zhu and Suma Bhat. 2020. GRUEN for evaluating linguistic quality of generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 94–108, Online. Association for Computational Linguistics.

A Appendix

A.1 Evaluation Metrics

Non-discourse Metrics. We consider the following non-discourse metrics.

- BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) are precision- and recall-oriented metrics respectively, both of which measure n-gram overlap between a hypothesis and a reference.

- BERTScore (Zhang et al., 2020) and MoverScore (Zhao et al., 2019) are set-based metrics used to measure the semantic similarity between hypothesis and reference. BERTScore uses greedy alignment to compute the similarity between two sets of BERT-based word embeddings from hypothesis and from reference, while MoverScore uses optimal alignments based on Word Mover's Distance (Kusner et al., 2015) to do so.

- SBERT (Reimers and Gurevych, 2019) fine-tunes BERT on the NLI datasets and uses pooling operations to produce sentence embeddings. We compute the cosine similarity between two sentence representations from hypothesis and from reference.

- S^3-pyr and S^3-resp (Peyrard et al., 2017) are supervised metrics that linearly combine ROUGE, JS-divergence and ROUGE-WE scores, trained on the TAC datasets with human annotated pyramid and responsiveness scores as supervision.

- BLEURT (Sellam et al., 2020) is another supervised metric that fine-tunes BERT on the concatenation of WMT datasets and synthetic data in the MT domain, with human judgment of translation quality as supervision.

- BARTScore (Yuan et al., 2021) and PRISM (Thompson and Post, 2020) depict sequence-to-sequence language models as metrics to compare hypothesis with reference. In reference-based settings, they both measure the likelihood that hypothesis and reference are paraphrases, but differ in the language models they rely on. PRISM has been based on a neural MT system trained from scratch on parallel data in MT, while BARTScore uses BART (Yuan et al., 2021) that has been fine-tuned on CNN/DailyMail (Hermann et al., 2015)—which is parallel data in summarization. We use the 'F-score' version of BARTScore as recommended in Yuan et al. (2021).

Discourse Metrics. We consider the following discourse metrics (including ours and coherence models).

- RC and LC (Wong and Kit, 2012) require neither source texts nor references and use lexical cohesion devices (e.g., repetition) within a hypothesis to predict text coherence. LC computes the proportion of words within hypothesis that are lexical cohesion devices, while RC computes the proportion of times that lexical cohesion devices appear in hypothesis.

- Entity Graph (Guinaudeau and Strube, 2013) and Lexical Graph (Mesgar and Strube, 2016) are popular coherence models used to perform discourse tasks such as essay scoring, both of which introduce a graph with nodes as sentences and adjacency matrices as the connectivity between sentences. Here, we use the average of adjacency matrices from the hypothesis as the proxy of hypothesis coherence. While Entity Graph draws an edge between two sentences if both sentences have at least one noun in common, Lexical Graph draws an edge if two sentences have a pair of similar words in common, i.e., the cosine similarity between their embeddings greater than a threshold.

- Lexical Chain (Gong et al., 2015) extracts multiple lexical chains from hypothesis and from reference. Each word is associated to a lexical chain if a word appears in more than one sentence. A lexical chain contains a set of sentence positions in which a word appears. Finally, the metric performs soft matching to measure lexical chain overlap between hypothesis and reference.

- FocusDiff and SentGraph are the two variants of DiscoScore, which use BERT to model semantics and coherence of readers' focus in hypothesis and reference. In particular, FocusDiff measures the difference between a common set of foci in hypothesis and reference in terms of semantics and frequency, while SentGraph measures the semantic similarity between two sets of sentence embeddings from hypothesis and reference—which are aggregated according to the number of foci shared across sentences and the distance between sentences.

A.2 Datasets

We outline two datasets in summarization, and one in document-level MT.

Text Summarization. While DUC[6] and TAC[7] datasets with human rated summaries, constructed one decade ago, were the standard benchmarks for comparing evaluation metrics in summmarization, they collect summaries only from extractive summarization systems. In the last few years, abstractive systems have become popular; however, little is known how well metrics judge them. Recently, several datasets based on CNN/DailyMail have been constructed to address this. For instance, SummEval (Fabbri et al., 2021), REALSumm (Bhandari et al., 2020), XSum (Maynez et al., 2020) and FEQA (Durmus et al., 2020) all collect summaries from both extractive and abstractive systems, but differ in the aspects human experts rate summaries. In this work, we consider the following two complementary summarization datasets.

- SummEval has been constructed in multiple-references settings, i.e., that each hypothesis is associated to multiple references. It contains human judgments of summary coherence, factual consistency, fluency and relevance. We only consider abstractive summaries as they have little lexical overlap with references.
- NeR18 (Grusky et al., 2018), in contrast, has been constructed in single-reference settings. It contains human judgments of summary coherence, fluency, informativeness and relevance. As majority of summaries are extractive, we include both extractive and abstractive for the inclusive picture.

Document-level Machine Translation. As document-level human ratings in MT are particularly laborious, hardly ever have there been MT datasets directly addressing them. First attempts suggested to use the average of much cheaper sentence-level ratings as a document score for comparing document-level metrics (Comelles et al., 2010; Wong and Kit, 2012; Gong et al., 2015). However, human experts were asked to rate sentences in isolation within a document. Thus, human ratings at both sentence and document levels cannot reflect inter-sentence coherence. Recently, the WMT20 workshop (Mathur et al., 2020) asks humans to rate each sentence translation in the document context, and follows the previous idea of 'average' to yield document scores.

Metrics	Encoders	Average
DS-FOCUS (NN)	+ BERT	71.97
	+ BERT-NLI	70.45
	+ Conpono	75.00
DS-SENT-U (NN)	+ BERT	35.61
	+ BERT-NLI	56.82
	+ Conpono	23.48

Table 2: Results of three contextualized encoders on SUMMEval. Results are averaged across four aspects.

Metrics	Average
DS-SENT-U (NN)	56.82
w/o sentence aggregation	46.21

Table 3: Ablation study on the use of adjacency matrix to aggregate sentence embeddings on SUMMEval.

In this work, we use the WMT20 dataset with 'artificial' document-level ratings. Note that WMT20 comes with two issues: (i) though sentences are rated in the document context, averaging sentence-level ratings may zero out negative effects of incoherent elements on document level and (ii) unlike SummEval and NeR18, WMT20 only contains human judgment of translation *adequacy* (which may subsume multiple aspects), not *coherence*.

For simplicity, we exclude system and reference translations with lengths greater than 512—the number of tokens at maximum allowed by BERT, as only a small portion of instances is over the token limit. Note that it is effortless to replace BERT with Longformer (Beltagy et al., 2020) to deal with longer documents for DiscoScore.

A.3 Analyses on Text Summarization

Choice of BERT Variants. Table 2 compares the impact of three BERT variants on DiscoScore. Conpono, referred to as a discourse BERT, has fine-tuned BERT with a novel discourse-level objective regarding sentence ordering. While strong on discourse evaluation benchmarks (Chen et al., 2019),

---

[6] `https://duc.nist.gov/data.html`
[7] `https://tac.nist.gov/data/`

Metrics	Mechanisms	Average
DS-Sent-U (NN)	+ greedy align	21.97
	+ optimal align	26.52
	+ mean-max-min-sum	56.82

Table 4: Averaged results of SentGraph variants based on three mechanisms on SUMMEval.

Metrics	SUMMEval	NeR18
BARTScore	14.13	24.78
PRISM	14.92	18.89
DS-Focus (NN)	10.81	10.42
DS-Sent-U (NN)	15.71	3.81

Table 5: Summary-level averaged Kendall correlations across all rating aspects.

Conpono is not always helpful, e.g., BERT-NLI is better for DS-Sent. These results suggest the best configuration for DiscoScore.

Impact of Sentence Connectivity. Table 3 shows an ablation study on the use of sentence connectivity. Aggregating sentence embeddings with our adjacency matrices (see Eq.3) helps considerably. This confirms the usefulness of aggregation from which we include coherence signals in sentence embeddings.

SentGraph Variants. Table 4 compares three DS-Sent variants as to how we measure the distance between two sets of sentence embeddings from hypothesis and reference. In particular, we refer to BERTScore (Zhang et al., 2020) as a 'greedy align' mechanism used to compute the similarity between two sets of sentence embeddings. As for 'optimal align', we use MoverScore (Zhao et al., 2019) to do so. While the two alignments directly measure the distance between the two sets, the simple statistics, i.e., mean-max-min-sum, derives a graph embedding from each set and computes the cosine similarity between two graph embeddings. We see that the 'statistics' wins by a big margin, and thus adopt this DS-Sent variant in all setups.

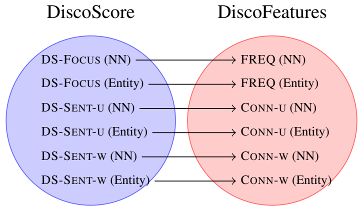


Figure 5: Links between the DiscoScore variants and discourse features.

	Sys-level		Doc-level	
Metrics	MQM	pSQM	MQM	pSQM
BARTScore	45.57	55.50	**34.90**	**28.96**
*DS-Focus (NN)	42.12	40.89	19.10	9.98
DS-Sent-U (NN)	**70.77**	**69.74**	19.98	14.49

Table 6: Document-level Kendall and system-level Pearson correlations between metrics and MQM/pSQM ratings on WMT20 in Chinese-to-English—which is the only language pair with such ratings in reference-based settings. *DS-Focus (NN) excludes focus that occurs only once in hypothesis/reference.

A.4 Analyses on MT

Correlation between Metrics. Figure 7 shows inter-correlations between metrics on WMT20 across languages. Overall, correlations are mostly high between non-discourse metrics, much weaker between discourse and non-discourse metrics—which confirms the orthogonality of them in that they rate translations in different aspects. We note that DS-Focus has the lowest correlations with all other metrics. For instance, DS-Focus is almost orthogonal to BERTScore and MoverScore. We investigated whether combining them receives additive gains. We find that a combination of DS-Focus and BERTScore (or MoverScore) provides little help in correlation with adequacy.

Settings	Metrics	Coherence	Fluency	Informative	Relevance	Average
$m(\text{hyp}, \text{ref})$	BARTScore	42.58	42.58	23.80	33.33	35.57
	PRISM	51.52	42.58	42.86	52.38	47.33
	DS-Focus (NN)	61.90	61.90	42.86	52.38	54.76
	DS-Focus* (NN)	**80.95**	**80.95**	**100.00**	**90.47**	**88.09**
	DS-Sent-u (NN)	14.29	14.29	14.29	23.81	16.67

Table 7: System-level Kendall correlations between metrics and human ratings on NeR18. DS-Focus* is the 'F-score' version of DS-Focus.

Settings	Metrics	Coherence	Consistency	Fluency	Relevance	Average
$m(\text{hyp}, \text{ref})$	BARTScore (max)	**78.79**	48.48	63.64	72.73	65.91
	BARTScore (original)	60.61	36.36	45.45	48.48	47.73
	FocusDiff (NN)	**75.76**	**63.64**	**78.79**	**81.82**	**75.00**
	FocusDiff (Entity)	69.70	57.58	72.73	75.76	68.94
	SentGraph-u (NN)	48.48	54.55	63.64	60.61	56.82
	SentGraph-u (Entity)	54.55	60.61	75.76	66.67	64.39

Table 8: System-level Kendall correlations between metrics and human ratings on SUMMEval in multi-reference settings. BARTScore (original) compares a hypothesis with one reference at a time, and takes the average of evaluation scores as a final score, while BARTScore (max) takes the maximum score.

			WMT20			
	SUMMEval	**NeR18**	**cs-en**	**de-en**	**ja-en**	**ru-en**
Number of references	11	1	1	1	1	1
Number of systems	12	7	13	14	11	13
Number of hypothesis per system	100	60	102	118	80	91
Number of sentences per hypothesis	3.13	1.90	15.21	13.84	11.29	9.46
Average number of foci in hypothesis	15.18	12.85	62.01	56.68	57.09	44.99
Average number of 'good foci' in hypothesis	2.47	2.56	13.16	13.37	15.07	9.95
Percent of hypotheses with 'good foci'	80.50%	43.80%	100%	98.60%	100%	100%

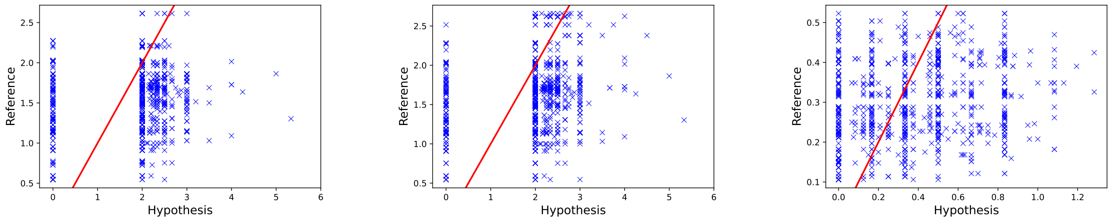
Table 9: Characteristics of summarization and MT datasets. 'good foci' denotes a focus appearing more than once in hypothesis. The more often a focus appears, the stronger the discourse signals are.

Metrics	Coherence	Consistency	Fluency	Relevance	Average
RC	45.45	51.52	54.55	57.58	52.27
BARTScore (max)	**78.79**	48.48	63.64	72.73	65.91
BARTScore (max) + RC	66.67	54.55	69.70	78.79	67.42
DS-Focus (NN)	75.76	**63.64**	**78.79**	**81.82**	**75.00**

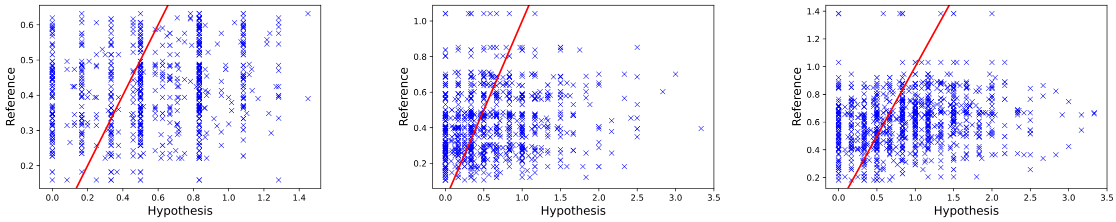
Table 10: Ensemble of non-discourse and discourse metrics (BARTScore + RC) vs DiscoScore.

	cs-en			**de-en**			**ja-en**			**ru-en**		
DiscoFeatures	$\mathcal{D} > 0$	$\mathcal{D} = 0$	$\mathcal{D} < 0$	$\mathcal{D} > 0$	$\mathcal{D} = 0$	$\mathcal{D} < 0$	$\mathcal{D} > 0$	$\mathcal{D} = 0$	$\mathcal{D} < 0$	$\mathcal{D} > 0$	$\mathcal{D} = 0$	$\mathcal{D} < 0$
Frequency (NN)	74.18	2.00	23.82	57.38	9.65	32.97	53.04	2.63	44.33	52.77	7.31	39.92
Frequency (Entity)	76.17	1.76	22.07	59.74	8.38	31.88	52.38	1.48	46.14	53.61	7.31	39.08
Connectivity-u (NN)	78.05	0.35	21.60	63.11	8.29	28.60	59.61	5.25	35.14	52.04	10.03	37.93
Connectivity-u (Entity)	79.46	0.35	20.19	62.02	8.20	29.78	59.44	5.09	35.47	52.87	9.40	37.72
Connectivity-w (NN)	77.93	0.24	21.83	64.85	4.64	30.51	59.12	0.49	40.39	59.98	5.12	34.90
Connectivity-w (Entity)	80.40	0.23	19.37	63.48	4.73	31.79	60.76	0.33	38.91	60.82	4.60	34.58

Table 11: Statistics of discourse features on WMT20. $\mathcal{D} > 0$ denotes the percent of 'reference-hypothesis' pairs for which $\mathcal{R}(\text{ref}) > \mathcal{R}(\text{hyp})$ with $\mathcal{R}$ as any one of these features, similarly for the definitions of $\mathcal{D} = 0$ and $\mathcal{D} < 0$. We exclude the pairs for which hypothesis and reference are the exact same.



(a) Focus Frequency (NN) (b) Focus Frequency (Entity) (c) Connectivity-u (NN)



(d) Connectivity-u (Entity) (e) Connectivity-w (NN) (f) Connectivity-w (Entity)

Figure 6: Distribution of discourse features over hypothesis and reference on SUMMEval.

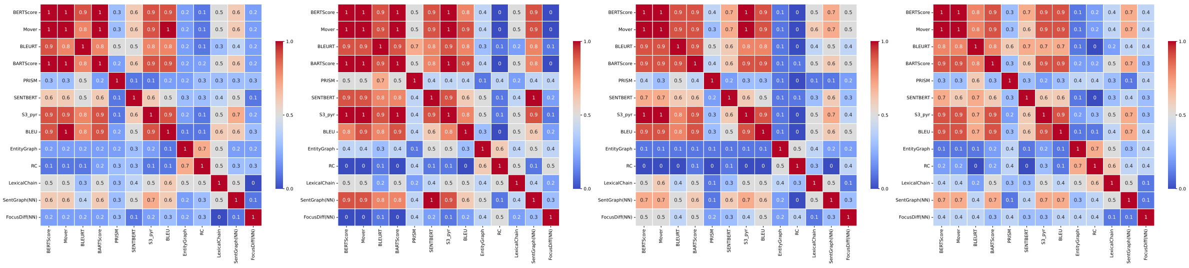


Figure 7: Pearson Correlations between metrics on WMT20 in cs-en, de-en, ja-en and ru-en (from left to right).

Settings	Metrics	Direct Assessment (Adequacy)				
		cs-en	de-en	ja-en	ru-en	Average
	Non-discourse metrics					
$m(\text{hyp}, \text{ref})$	BLEU	7.44	57.52	41.48	10.74	29.30
	BERTScore	10.82	60.38	**46.95**	13.08	32.81
	MoverScore	**15.40**	**61.69**	42.12	13.78	**33.25**
	BARTScore	10.82	60.26	46.30	14.95	33.09
	PRISM	8.64	58.83	32.48	15.42	28.84
	SBERT	13.20	55.26	33.44	10.04	27.99
	BLEURT	12.01	58.83	37.94	**18.22**	31.75
	S^3-pyr	6.25	58.83	42.44	13.78	30.33
	S^3-resp	5.85	58.59	**47.26**	14.71	31.61
	Discourse metrics					
$m(\text{hyp})$	RC	5.85	7.19	**8.68**	9.34	7.77
	LC	**9.23**	1.72	3.53	6.07	5.14
	Entity Graph	5.06	**43.24**	3.53	10.51	15.59
	Lexical Graph	2.28	**43.60**	5.14	**13.55**	**16.15**
	Discourse metrics					
$m(\text{hyp}, \text{ref})$	Lexical Chain	21.54	35.15	15.11	16.12	21.99
	FocusDiff (NN)	7.64	33.13	19.29	2.57	15.66
	FocusDiff (Entity)	6.45	33.73	19.94	1.64	15.44
	SentGraph-u (NN)	7.64	57.16	39.22	18.22	**30.56**
	SentGraph-u (Entity)	7.65	57.17	39.23	18.22	**30.57**
	SentGraph-w (NN)	7.65	57.18	39.22	18.21	**30.57**
	SentGraph-w (Entity)	7.65	57.17	39.23	18.22	**30.57**

Table 12: Document-level Kendall correlations between metrics and human rated translation quality on WMT20.

Know your audience: specializing grounded language models with listener subtraction

Aaditya K Singh
Gatsby Computational Neuroscience Unit
University College London
London, W1T 4JG
aaditya.singh.21@ucl.ac.uk

David Ding
DeepMind
London, UK
fding@deepmind.com

Andrew Saxe
Gatsby Computational Neuroscience Unit
University College London
London, W1T 4JG
a.saxe@ucl.ac.uk

Felix Hill
DeepMind
London, UK
felixhill@deepmind.com

Andrew Kyle Lampinen
DeepMind
London, UK
lampinen@deepmind.com

Abstract

Effective communication requires adapting to the idiosyncrasies of each communicative context—such as the common ground shared with each partner. Humans demonstrate this ability to specialize to their audience in many contexts, such as the popular game Dixit. We take inspiration from Dixit to formulate a multi-agent image reference game where a (trained) speaker model is rewarded for describing a target image such that one (pretrained) listener model can correctly identify it among distractors, but another listener cannot. To adapt, the speaker must exploit differences in the knowledge it shares with the different listeners. We show that finetuning an attention-based adapter between a CLIP vision encoder and a large language model in this contrastive, multi-agent setting gives rise to context-dependent *natural language* specialization from rewards only, without direct supervision. Through controlled experiments, we show that training a speaker with two listeners that perceive differently, using our method, allows the speaker to adapt to the idiosyncracies of the listeners. Furthermore, we show zero-shot transfer of the specialization to real-world data. Our experiments demonstrate a method for specializing grounded language models without direct supervision and highlight the interesting research challenges posed by complex multi-agent communication.

1 Introduction

Human language use is communicative, and thus involves substantial adaptation to each conversational partner and context (Clark and Wilkes-Gibbs, 1986; Clark, 1996; Gallois et al., 2005; Frank and Goodman, 2012; Hawkins et al., 2019, 2022). We can adapt our speech to complex social settings, with multiple partners (Mankewitz et al., 2021; Boyce et al., 2022) and competing constraints, such as politeness vs. explicitness (Yoon et al., 2016). While current language and captioning models can increasingly imitate human language and scene descriptions (Brown et al., 2020; Tsimpoukelli et al., 2021; Mokady et al., 2021; Alayrac et al., 2022; Elhagry and Kadaoui, 2021), they generally are not explicitly adapted to a particular partner or to satisfy multiple constraints. Common approaches to specializing the language generated by these models fall into two categories: finetuning on a supervised dataset (e.g. Hu et al., 2021), which requires task-specific labeled examples, or prompt engineering, which is often brittle and may still require task-specific examples (Liu et al., 2021). Here, we take inspiration from how humans adapt their language to their audience to offer an alternative approach to specializing grounded language models without direct supervision.

Our approach to specialization is inspired by the popular multi-player game Dixit (Roubira, 2008), which rewards each player's ability to adapt their communication to a particular audience. In a round of Dixit, the "speaker" describes a chosen image in language. Then, the rest of the players (two or more "listeners") attempt to identify the target

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3884–3911
May 2-6, 2023 ©2023 Association for Computational Linguistics

image from a pool of distractors. Importantly, the speaker is rewarded when *some but not all* listeners correctly identify the image. This leads to creative captions that target the common ground between a speaker and some (but not all) of the listeners (see example in Figure 1a). For more details, we refer readers to Kunda and Rabkina (2020), who introduce Dixit as a grand challenge for AI.

Using this inspiration, we formulate a setting where grounded language models can specialize their language without direct supervision, from rewards only. We build on prior work in emergent communication (Cogswell et al., 2020), and adapt a pretrained captioning model (the "speaker") by finetuning a small fraction of its parameters to maximize a Dixit-like reward—this minimal adaptation helps reduce language drift (Lee et al., 2019). We go beyond prior work by investigating a complex, not fully cooperative, setting (Figure 1b). The speaker's goal is to communicate information to some listeners, but not others, by exploiting differences in listeners' idiosyncratic "personalities". We instantiate this setting by having a speaker model communicate with multiple (frozen) contrastive listener models differing along perceptual axes. We design careful experiments, with datasets, metrics, and controls to ensure quantifiable assessment of language specialization.

We show that training a speaker with a pair of listeners in this Dixit-inspired setting can lead to *diverse* specialization of natural language, with minimal language drift, and without direct supervision. From rewards, the speaker identifies and learns to cue to the difference between two listeners, which we call "listener subtraction". We show this adaptation across many pairs of perceptually-differing listeners, and various datasets (Section 4.1, 4.2). For example, when trained with one listener which sees color, and another which sees only grayscale, the speaker learns to exclusively use color words—it exploits the specialized information it shares with the first listener. Furthermore, the speaker exhibits some zero-shot transfer of its language specialization from artificial to realistic datasets (Section 4.3). To our knowledge, our work is the first to consider natural language communication in a grounded, multi-listener setting, without direct supervision.

2 Related work

Dixit. Kunda and Rabkina (2020) discuss the full game of Dixit as a grand challenge for AI and posit the various interesting subproblems that would need to be solved (but do not attempt to solve them). Here, we focus on addressing the "Find a Phrase" subproblem—how the speaker should choose a phrase such that some but not all listeners correctly identify the image.

Adapting (grounded) language models. A large body of prior work focuses on adapting large pretrained language models (Hu et al., 2021; Ziegler et al., 2020; Brown et al., 2020; Tsimpoukelli et al., 2021) to various tasks. Our work also focuses on natural language specialization, but differs in that we use no supervised data. In that sense, our work is more similar to approaches that fine-tune pretrained models with reinforcement learning (Stiennon et al., 2020; Ouyang et al., 2022). Unlike those works, we focus on grounded communication. With respect to grounding, we take inspiration from prefix tuning (Li and Liang, 2021) and use an image-to-prefix encoder to condition a pretrained language model on images, as in Tsimpoukelli et al. (2021). These models can adapt their language via prompting (Brown et al., 2020), which we consider as a baseline for our approach.

Pragmatics through model interactions. Work in cognitive science has explored how pragmatic inferences can be explained as reasoning over simpler speaker and listener models (Frank and Goodman, 2012). This perspective aims to explain human pragmatic references in grounded reference games and can help train models that give and follow instructions (Fried et al., 2018) or help train listener models that adapt to individual human speakers (Hawkins et al., 2020). Our work is related to this framework, but differs in that the *speaker model* is adapted to use *differences between multiple listeners*.

Emergent communication. Finally, there is prior work on emergent communication in image reference games. We focus on works involving natural language generation, like our setting. Cogswell et al. (2020) focus on a speaker agent that asks questions in dialogue with a single, pretrained-and-frozen listener, to identify an image among distractors. Lazaridou et al. (2020) consider speaker and listener agents in a more standard image reference game—the speaker describes an image, and the single listener must choose the image among distractors. Both papers use the pretrain-then-finetune approach to minimize language drift; we adopt a similar methodology. Like Cogswell et al.

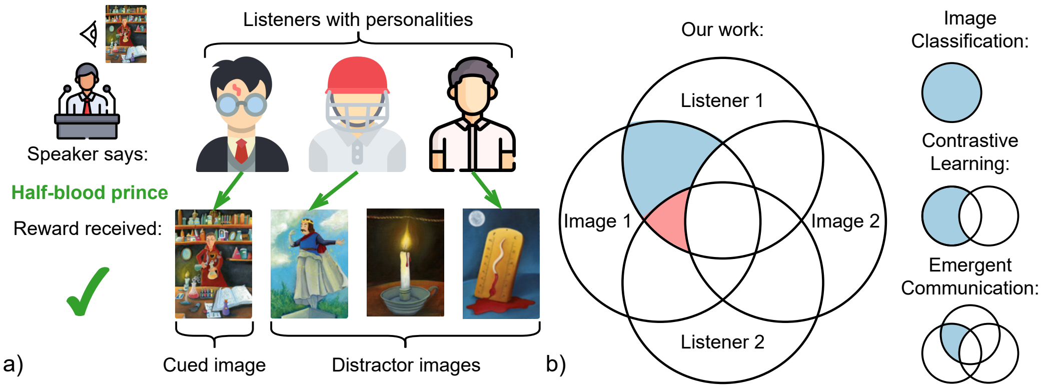


Figure 1: a) An example round of Dixit, the inspiration for our work. The speaker uses a specialized caption (green), so that only one listener (who has read Harry Potter) correctly identifies the image. This outcome is rewarded per the *some but not all* rule. b) How our work relates to past work. Image classification: models say something about an image. Contrastive learning: models say something to distinguish an image from others. Emergent communication: models say something to distinguish an image from others in a way one listener will understand. Our work: models say something to distinguish an image from others in a way one *but not another* listener will understand.

(2020), we freeze our listeners to maintain language grounding (cf. Lee et al., 2019), and focus on the speaker. Unlike prior work, our setting involves simultaneous, natural language communication to multiple listeners and is not fully cooperative.

3 Methods

3.1 Multi-agent image reference game

We focus on a multi-agent image reference game inspired by a single round of Dixit. We frame this problem as a challenge of tuning a pretrained multimodal model to specialize its language, without direct supervision on how to specialize. Our setup (Figure 2) involves one speaker model communicating with multiple listener models about a target image. The speaker model receives the target image and outputs a caption. Each listener model receives the caption and the target image along with some (random) distractor images. Each listener model's goal is to correctly identify the target image from the distractors, based on the speaker's caption.

Following the some-but-not-all reward structure of the Dixit game, we reward the speaker model for *the difference in the (binary) accuracies of listeners 1 and 2 on a given set of images*. To avoid co-adaptation and pragmatic drift (cf. Lazaridou et al., 2020), we pretrain and freeze our listener models. Thus, only the speaker has learnable parameters.

3.2 Speaker model

The speaker model (Figure 2a) is inspired by the *Frozen* model (Tsimpoukelli et al., 2021) and ClipCap (Mokady et al., 2021). We build the speaker using several pretrained components, and adapt only a small piece to our setting. The model is composed of a pretrained-and-frozen CLIP visual encoder (Radford et al., 2021), learned attentive (QKV) adapter layer, and a pretrained-and-frozen Transformer (decoder) language model (Vaswani et al., 2017; Hoffmann et al., 2022). An input image is first passed through the visual encoder. The unpooled output of the encoder is flattened and passed into the adapter, a Perceiver-inspired cross attention layer (Jaegle et al., 2021). The adapter outputs $n = 32$ prefix tokens, which are used to condition the language model and generate up to 32 output tokens. See Appendix A for more details.

The only parameters of this model that are ever trained are in the adapter. We keep the pretrained CLIP encoder and language model frozen (to help prevent language drift), and pretrain the adapter on the Conceptual Captions dataset (Sharma et al., 2018) to give our speaker basic captioning abilities (see Appendix B.1). All our experiments start from this captioning-pretrained base speaker model. We finetune the adapter for each experimental condition, which consists of a dataset to train on and a pair of listeners (we refer to this process as "training the speaker"). We do not have supervision for what to say when playing the game—much as a human player needs to adapt without direct supervision. Instead, we use REINFORCE (Williams, 1992) to train the speaker's adapter parameters, based on the rewards received for differences in the listener accuracies (see Section 3.1). Our choice to finetune only the adapter builds off prior works in emergent communication that use pretrained-and-frozen components (Cogswell et al., 2020; Lazaridou et al., 2020). By repeated interactions with

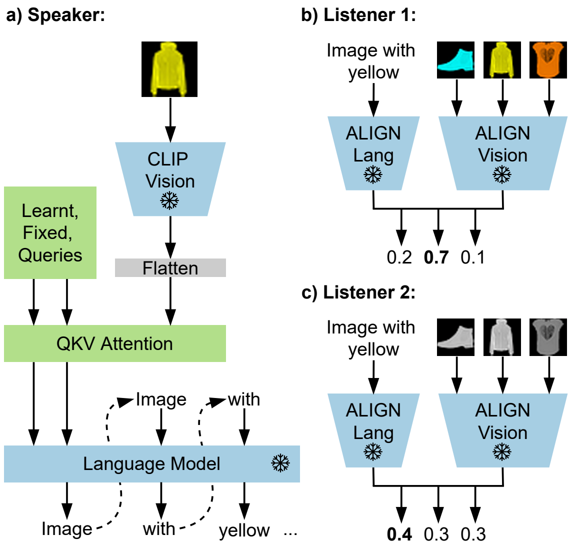


Figure 2: Our setup for cFMNIST and the grayscale transform. a) The speaker receives an image and generates a caption. b) Listener 1 receives unperturbed images and picks best match to the caption (correctly). c) Listener 2 receives images with the grayscale transform applied and picks the best match to the caption (incorrectly). Then, the speaker receives a reward of 1 (the difference in the accuracies) which it uses to update its adapter parameters (green) via REINFORCE. All components in blue are pretrained and frozen.

the same pair of listeners in each experimental condition, the speaker should update its weights to produce specialized language to that listener pair.

3.3 Listener models

For our listener models (Figure 2b,c), we use the vision and language encoders of a pretrained ALIGN NFNet F5 model (Jia et al., 2021; Brock et al., 2021). For a set of input images and caption, we compute image-caption match scores for each image, and the listener selects the image with the highest alignment score.

To investigate the speaker's ability to adapt to distinct listeners, we experimentally manipulate the listeners' knowledge. We use the same pretrained model for each listener, but manipulate each listener's knowledge by applying distinct image transformations to the inputs. For listener 1, we do not transform the input images. For listener 2, we consider the following transformations:

Crop: Images are cropped to the top-right quadrant then resized to the original size.

Blur: Images are Gaussian blurred (radius 25 px).

Grayscale: Images are converted to grayscale.

While listener "personalities" could differ along any linguistic, conceptual, and perceptual axes, we focus on these perceptual transformations because they enable quantitative assessments of language specialization. When listener 2 sees cropped images, we expect the speaker to cue objects outside the cropped region. When listener 2 sees blurred images, we expect the speaker to cue objects without the use of color, as distinct objects will not be visible to the second listener but color will still largely be visible. Conversely, when listener 2 sees grayscale, we expect the speaker to only use color words and to specifically stop referring to objects. We design controlled datasets for each experiment, in which each relevant metric can be measured.

3.4 Datasets and metrics

The main datasets we use are transformed versions of Fashion-MNIST (Xiao et al., 2017). Fashion-MNIST (FMNIST) consists of grayscale images of different types of clothing items. We create a colored version (cFMNIST), where each image is randomly colored with one of 8 colors. To test training with the crop listener, we then create a tiled version of cFMNIST (tcFMNIST) where each image consists of two random cFMNIST images, one in the top-right, and one in the bottom-left. For more realistic images, we use the COCO dataset (Lin et al., 2014), taking center square crops of all images. Example images for all dataset and transform pairs are shown in Figure 3.

For analysis metrics, we first evaluate the proportion of captions generated by the speaker on test images that had color words in them ("color prevalence") and the proportion of captions that contained a clothing-related word ("FMNIST keyword prevalence"). Each of these metrics identifies language specialization. To measure image-relevance of produced language, we compute metrics measuring how often an object word used corresponded to an actual object in the image, accounting for synonyms. These metrics are referred to as "object prevalence" for cFMNIST, and "bottom-left object prevalence" and "top-right object prevalence" for tcFMNIST. For colors, it's difficult to enumerate all synonyms as the model often uses pairs of colors, e.g., "blue and white" for "cyan". Instead, we found that a "color diversity" metric—which measures the speaker's variation in color use—offers an effective proxy for determining color relevance.

See Appendices C, D, for more details on datasets and metrics, respectively.

	Speaker and Listener 1 see:	Listener 2 sees:	Speaker output	
			Start of training:	End of training:
a)			a set of shoes and a t-shirt	a t-shirt
b)			the yellow jacket with a black hood	a hooded jacket
c)			the yellow jacket with a black hood	yellow
d)			person, a cyclist, is pictured in his new bike	a red and white

Figure 3: Example test images and speaker captions at the start and end of training for various experimental conditions. By row, the speaker trains on a different dataset (a: tcFMNIST, b,c: cFMNIST, d: COCO), and the second listener sees a different transformed image (a: top-right crop, b: blurred, c,d: grayscale). Diverse language specialization emerges by the end of training. See Appendix H for more samples.

3.5 Training

A training "episode" consists of the speaker receiving an image (observation), generating a caption (action), and receiving the difference in listener accuracies as a reward. As noted above, only the adapter in the speaker has trainable parameters, which are updated via REINFORCE (Williams, 1992). We use nucleus sampling (Holtzman et al., 2019) to generate captions. We add a small reward penalty to incentivize short captions (a weight λ times the number of words used) as an articulatory effort minimization bias (cf. Lazaridou and Baroni, 2020). We use a batch size of 128 images, with 3 random distractors for each image drawn from the same batch. See Appendix B.3 for more details.

4 Results

4.1 Emergence of specialization

By training a speaker to maximize the difference in accuracy between two fixed listeners, we see strong language specialization across many listener pairs (Figure 3a-c, 4a-c). When the second listener sees only the top-right crop of the image, the speaker learns to cue to objects that the second listener cannot see (Figure 4a, bottom). The first listener's accuracy remains stable through learning, but the second listener's accuracy decreases to chance level. This corresponds to the decrease in top-right object prevalence in the speaker's captions and an *increase* in bottom-left object prevalence.

When the second listener sees only the blurred version of the image, the speaker learns to reduce its use of color words, while still referring to the target object. The overall learning dynamics are similar to the crop case: the first listener's accuracy remains roughly constant, while the second drops to chance level, again paralleling the decrease in color prevalence in the speaker's captions.

By contrast, when the second listener sees only a grayscale image, we see a rapid decrease in object prevalence and increase in color prevalence. However, this initial learning causes both listener accuracies to drop, as the speaker greedily uses colors that may not be image relevant. Then, from iterations 2.5k to 25k, the speaker slowly learns to use more accurate colors for each image, and the first listener's accuracy increases slowly but steadily (while the second listener stays at chance). At the end of training, color diversity is comparable with that produced by language prompts (see Section 4.4.3, Tables 1 and 15), indicating that the speaker is using diverse, image-relevant colors.

These experiments illustrate that the speaker model can optimize its reward in different ways depending on the pair of listeners—decreasing the second listener's accuracy, increasing the first listener's accuracy, or both. Each strategy produces meaningful, measurable changes in the speaker's language, while keeping drift to a minimum. Captions show little structural drift (assessed by an independent language model's likelihoods, Appendix E), as well as little semantic and pragmatic drift (as assessed by the above metrics). See Appendix H for more images, distractors, and speaker ouputs.

4.2 Extending to real world data

We next apply our approach to a setting with real-world data—images from the COCO dataset (Lin et al., 2014). The effect of most transforms is harder to quantify exactly, as the images contain many objects, and enumerating all their synonyms is infeasible. Thus, we use the grayscale transform, which still offers usable metrics (color prevalence and diversity).

Figure 3d, 4d show model samples and the speaker's training curves on COCO. The speaker starts with a high initial accuracy to both listeners. Then, we qualitatively observed (by inspecting model samples) that the speaker starts "exploring" different strategies to differentiate the listeners (iterations 0-20k). Just before 20k iterations, the speaker starts using colors slightly more often, then

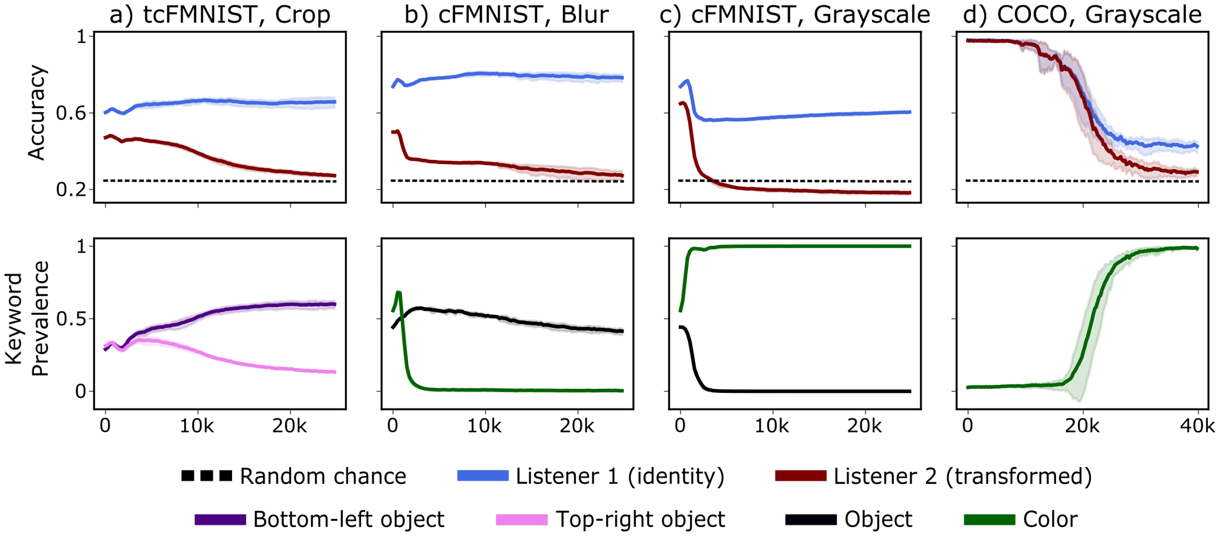


Figure 4: Accuracy and task-relevant metrics for many dataset and listener pairs. By column, the second listener sees a different transformed image (a: top-right crop, b: blurred, c,d: grayscale). The top row shows listener accuracies as the speaker learns to specialize. The bottom row shows prevalance of task-relevant keywords indicated by line color—note that object metrics measure both grounding and language specialization, while color metrics only measure specialization (see Section 3.4). Error bars show 95% confidence intervals (95% CIs) over 5 seeds.

exhibits a near-stage-like transition to exclusively using colors. Qualitatively, the speaker switches standard captions (e.g., "person in front of a bicycle") to color-focused captions (e.g., "red and white") for each image at some point during training. Once the speaker only uses color for its captions, the individual listener accuracies diverge significantly. Notably both listener accuracies are far worse than at the start of training, as color alone is an imperfect distinguisher on COCO (and overspecifying colors may give away too much, since object category is correlated with color in real images). However, the speaker has successfully learned to specialize its language to exploit the difference between the two listeners. Furthermore, this language remains grounded, as evidenced by end-of-training color diversity being comparable to color diversity produced from language prompting (see Section 4.4.3, Tables 1 and 16).

4.3 Zero-shot transfer

We next explore zero-shot transfer of the language specialization from our simpler dataset (cFMNIST) to the more challenging setting of COCO. To do so, we trained a speaker in the grayscale transform setting on cFMNIST (Listener 1 sees the unperturbed image, Listener 2 sees the grayscale image) and tested its zero-shot behavior on COCO. As baselines, we consider the behavior of the pretrained captioning model (that the speaker is initialized to), and the behavior given the best natural-language prompt (see Section 4.4.3). We also compare to the behavior of an expert speaker trained to specialize on COCO (the end of training in Figure 4d).

We find significantly-above-baseline zero-shot transfer, with increasing use of color on COCO as the speaker is trained on FMNIST (Figure 5b). Furthermore, this specialization leads to meaningful differences in listener accuracies (Figure 5a). This specialization transfer from FMNIST to COCO performs better (across all four metrics in Figure 5) than baseline zero-shot approaches (the untuned captioning model), and on par with the best language prompt (see Section 4.4.3).

However, zero-shot behavior is far from perfect—the model loses some grounding relative to the experiments above. Zero-shot, the model strongly prefers "red" (Figure 5c), using it in nearly twice as many captions as the COCO-trained speaker, including even some non-red images. We believe this may be due to the average "color" of the 8 colors we use for cFMNIST, in RGB, is 907080, indicating a slight red skew of the data. To improve generalization, we therefore trained on a richer, balanced colored version of FMNIST, where image hues are sampled randomly from 360 equally-spaced options (as opposed to the 8-hue cFMNIST used above) – we refer to this dataset as cFMNISTmany. We find that this improves zero-shot transfer across all metrics, pushing it beyond the performance of

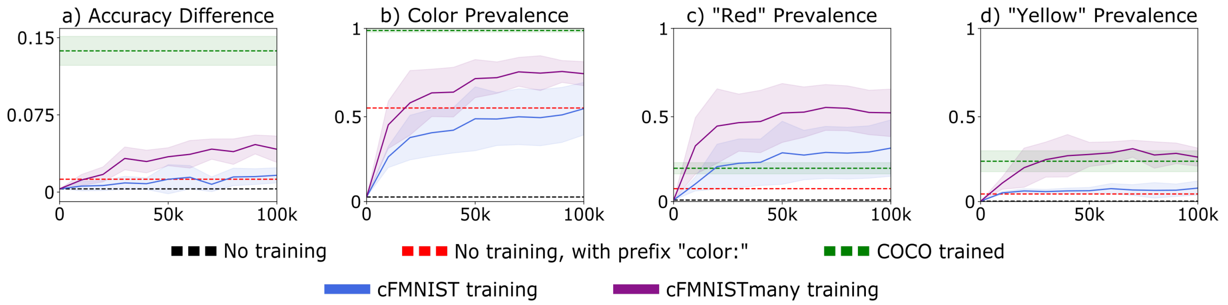


Figure 5: Zero-shot transfer results from cFMNIST to COCO and relevant baselines. Each subplot shows a different metric, with different lines corresponding to different conditions or baselines. The two conditions are training on cFMNIST (blue) or cFMNIST with 360 hues (purple). Various baselines are included as dotted lines: base captioning model (black), base captioning model with explicit prompting (red), fully trained on COCO (green).

the best explicit language prompt. Although this model still has slightly increased "red preference", its usage of other colors is also significantly better than the 8-hue-trained speaker. For example, in Figure 5d, we see that the cFMNISTmany-trained speaker uses yellow in a similar percentage of captions as a COCO-trained speaker. A better captioning model, that was adapted on a more diverse set of images, would likely transfer even better.

4.4 Ablations

We finally consider ablations to our set-up to identify key aspects. See Appendix F for full results.

4.4.1 Single listener

In our original experiments, the speaker learns to specialize in the presence of multiple listeners, but are both listeners necessary? Could the speaker also specialize through communicating with only a single listener? To test this possibility, we train a speaker model to optimize the accuracy of just a single listener (so the reward is binary 0/1 listener accuracy). See Appendix B.4 for details.

In the tcFMNIST case, prevalence of keywords for both locations in the image increases consistently, indicating a lack of specialization to image sub-regions. For cFMNIST, we find that single-listener training leads to increased use of colors and correct keywords; we never see a differential specialization that full multi-listener training induces in our original experiments. For COCO, we see the largest difference between single-listener and multi-listener cases—the single listener case does not learn to use color to distinguish images.

4.4.2 Non-contrastive reward

We next consider training with multiple listeners, but a non-contrastive reward. Namely, instead of optimizing for the difference in accuracy between listeners, we optimize directly for the difference in ALIGN match on the cued image. We call this approach "non-contrastive" as no distractor images are required. See Appendix B.5 for details.

Training in this setting does lead to some language specialization, but is worse than contrastive training in 3 out of 4 settings we consider. For the crop setting (Table 9), we see a reduced use of the top-right keyword, but only a modest increase in the use of the bottom-left keyword compared to contrastive reward training. For the blur setting (Table 10), the non-contrastive reward actually outperforms the contrastive reward. The model uses image-relevant keywords 70% of the time, and stops using varied colors (but continues to mention a single color). For the grayscale setting on both FMNIST and COCO, we observe strong specialization using the non-contrastive reward (as evidenced by low FMNIST keyword prevalence, and high color prevalence in Tables 11, 12). However, the captions in this case lose image relevance, as evidenced by markedly lower color diversity. Qualitatively, we observe the model using the same color for each image (which achieves more reward than baseline, but is obviously not desirable). Thus, overall, we find that the contrastive reward encourages specialization while preserving image relevance in this multi-listener setting.

4.4.3 Explicit prompting

Finally, we evaluate a simpler, commonly-used approach to model specialization: explicit prompting. These experiments also provide a baseline for our zero-shot results (in Section 4.3), as no training is involved. We created a variety of possible prefixes to elicit specialization, as explicit prompting can be brittle. Full results are provided in Appendix G.

Table 1 shows the top three prompts and relevant metrics for the (identity, grayscale) transform

Caption Prefix	FMNIST metrics			COCO metrics			
	Accuracy Difference	Color Prevalence	Color Diversity	Object prevalence	Accuracy Difference	Color Prevalence	Color Diversity
No prefix	0.089	0.556	0.014	0.442	0.003	0.027	0.022
the color of this image is	0.174	0.901	**0.020**	0.000	-0.010	0.135	**0.036**
color:	0.118	0.971	0.004	0.111	0.013	0.541	0.030
colors in image:	0.048	0.999	0.001	0.000	0.019	0.335	0.018
Fully trained	**0.422**	**1.000**	0.015	**0.000**	**0.156**	**0.988**	0.033

Table 1: Relevant metrics for various caption prefixes. The no prefix condition corresponds to the base captioning model. Fully trained results match values at the end of training from Figure 4—full training via listener subtraction provides superior language specialization compared to explicit prompting.

pair, on both FMNIST and COCO. Explicit prompting does lead to some specialization (when compared to the base captioning model), but fails to provide the robust specialization we observe in Section 4. Captioning prefixes can be quite brittle, as evidenced by the diverse behavior we observe from superficially similar prompts. For "colors in image:" and "color:", the model has high color prevalence, but low diversity. Qualitatively, the model tends to list all colors, with only the first color corresponding to the true color of the image. For "color:" we also observe that the model still uses nouns. For "the color of this image is" we observe slightly reduced color prevalence, but much stronger color diversity and little usage of nouns.

On COCO, similar color prompts yield surprisingly large differences in color prevalence. We also observe "switching" behavior: per image, the speaker either ignores the caption prefix (e.g., "color: person and her son play in the water") or specializes (e.g., "color: blue and white"). Qualitatively, this behavior mirrors the stage-like transition in the COCO-trained speaker, in Section 4.2.

5 Discussion

We have demonstrated a method for specializing a grounded language model without *direct supervision*, by finetuning a small fraction of its parameters in a complex multi-agent setting. We have shown that this approach enables diverse types of specialization with minimal language drift, and have identified the essential aspects of this approach. We believe that our work offers a novel perspective on adapting models to new settings without supervised data.

Our approach to specializing a grounded language model rewards the speaker for using differences in the information it shares with two listener models. This formulation allows for flexible adaptation of models to complex tasks. For example, our approach could potentially be used to diagnose biases in one contrastive listener model by comparing to another (e.g., trained on a different dataset), an area of considerable interest (Agarwal et al., 2021). A speaker optimized for the difference between two contrastive models could identify biases present in one listener, without the need for manually curated word lists. Additionally, training against listeners of different skill could potentially improve language quality without relying on costly human annotations (e.g. Stiennon et al., 2020).

Long term, we hope our work contributes to personalized AI. Humans prefer interacting with others who share *rare preferences* (Vélez et al., 2019), so an AI that personalizes its language to users would likely be preferred to one that uses standardized language. Personalization could improve other kinds of communication too, like explanations—e.g., adapting feedback to a particular student's idiosyncratic knowledge (cf. Wang et al., 2017). Our approach enables such adaptation through interaction, without explicit language supervision.

Towards Dixit Finally, while our experimental setup is inspired by *Dixit* game, we only make the first steps towards a model that can play the full version of Dixit effectively with humans—something that is an appropriate grand challenge for AI, as Kunda and Rabkina (2020) suggest. We highlight some of the next steps below.

The present work focuses on perceptual differences, while human play often relies on differences in conceptual knowledge. Finetuning listeners on different datasets (e.g., scenes from Harry Potter movies as in Figure 1a) might allow such conceptual differences to emerge. A challenge would be using datasets where this conceptual difference is relevant (Dixit achieves this with abstract images). This could potentially be overcome by generating synthetic data using modern text-to-image models (e.g. Ramesh et al. (2022)). Given the robustness to different listener pairs already exhibited, we believe this is a promising future direction. Moreover, human players adapt their language from a few interactions, while our speaker trains for thousands of iterations. One way to overcome this limitation would be to combine our work with concurrent progress in visual language models. Alayrac et al. (2022) introduce a model, Flamingo, which can rapidly adapt to different visual language tasks in a few-shot setting. Using such a high-quality base model as the speaker might enable reward-driven adaptation from just a few interactions.

We hope that this work will inspire further research on settings like Dixit, and help to enhance the capabilities of grounded language models and communicative agents more broadly.

Limitations

We want to reiterate important limitations with the work in its present form. We only considered settings with two listeners (for ease of quantitative assessment); additional challenges may be present when extending to three or more listeners, which may be necessary to express more complex objectives. In addition, the current work assumes that listeners exist that differ along the axis of desired specialization; finding such listeners might be challenging. Furthermore, despite the overall preservation of language, we qualitatively observed some degradation in some runs. Introducing a weighted KL-divergence loss term to the pretrained captioning model likelihood (cf. Lazaridou et al., 2020) might further improve specialized language quality. Alternative sampling methods (cf. Chaudhary et al., 2022), such as typical decoding (Meister et al., 2022), may also improve generated language quality. Finally, our paper does not directly place the agent in interactions with humans, or compare to how humans adapt their language in similar tasks. Further work could include human experiments to get a better sense of language quality and relevance of generated captions and/or identify differences in human and agent language adaptation.

Ethical considerations

As discussed above, applications for this work would train speakers using differences between listener models with different weights or even architectures. A potential risk of adapting speakers via this approach is that the speaker might pick up biases that one listener has that are not part of the intended specialization—an undesirable quality. As suggested above, one possible approach to mitigating this would be to explicitly include biased listeners, but always penalize the speaker for the biased listeners' accuracy. Another approach could be debiasing the underlying (frozen) language model (Schick et al., 2021; Meade et al., 2022), as the speaker's propensity for producing offensive language is inherited from this underlying model. Long-term, we believe that strategies like those detailed by Weidinger et al. (2021) will be necessary to mitigate risks of large language models (and models such as ours that utilize them).

Acknowledgements

The authors would like to thank Frederic Besse, Denis Teplyashin, and Maria Tsimpoukelli for advice on the engineering aspects of the work. We would also like to thank Angeliki Lazaridou, DJ Strouse, Stephanie Chan, and Oriol Vinyals for useful discussions and feedback on the draft.

The authors also thank Flaticon.com for providing free access to various icons used in Figures 1 and 2. We also thank the creators of Dixit(Roubira, 2008) for creating an inspiring game and the images used in 1.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Sandhini Agarwal, Gretchen Krueger, Jack Clark, Alec Radford, Jong Wook Kim, and Miles Brundage. 2021. Evaluating CLIP: towards characterization of broader capabilities and downstream implications. *CoRR*, abs/2108.02818.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. 2022. Flamingo: a visual language model for few-shot learning.

Veronica Boyce, Robert Hawkins, Noah Goodman, and Michael Frank. 2022. Two's company but six is a crowd: emergence of conventions in multiparty communication games.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. JAX: composable transformations of Python+NumPy programs.

Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. 2021. High-performance large-scale image recognition without normalization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1059–1071. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Amit Kumar Chaudhary, Alex J. Lucassen, Ioanna Tsani, and Alberto Testoni. 2022. Are current decoding strategies capable of facing the challenges of visual dialogue? In *Proceedings of the 15th International Conference on Natural Language Generation*, pages 288–297, Waterville, Maine, USA and virtual meeting. Association for Computational Linguistics.

Herbert H Clark. 1996. *Using language*. Cambridge university press.

Herbert H Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22(1):1–39.

Michael Cogswell, Jiasen Lu, Rishabh Jain, Stefan Lee, Devi Parikh, and Dhruv Batra. 2020. Dialog without dialog data: Learning visual dialog agents from vqa data. In *Advances in Neural Information Processing Systems*, volume 33, pages 19988–19999. Curran Associates, Inc.

Ahmed Elhagry and Karima Kadaoui. 2021. A thorough review on recent deep learning methodologies for image captioning. *CoRR*, abs/2107.13114.

Michael C Frank and Noah D Goodman. 2012. Predicting pragmatic reasoning in language games. *Science*, 336(6084):998–998.

Daniel Fried, Jacob Andreas, and Dan Klein. 2018. Unified pragmatic models for generating and following instructions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1951–1963, New Orleans, Louisiana. Association for Computational Linguistics.

C. Gallois, Tania Ogay, and Howard Giles. 2005. Communication accommodation theory: A look back and a look ahead. *Theorizing About Intercultural Communication*.

Robert Hawkins, Minae Kwon, Dorsa Sadigh, and Noah Goodman. 2020. Continual adaptation for efficient machine communication. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 408–419, Online. Association for Computational Linguistics.

Robert D Hawkins, Michael Franke, Michael C Frank, Adele E Goldberg, Kenny Smith, Thomas L Griffiths, and Noah D Goodman. 2022. From partners to populations: A hierarchical bayesian account of coordination and convention. *Psychological Review*.

Robert XD Hawkins, Noah D Goodman, and Robert L Goldstone. 2019. The emergence of social norms and conventions. *Trends in cognitive sciences*, 23(2):158–169.

Tom Hennigan, Trevor Cai, Tamara Norman, and Igor Babuschkin. 2020. Haiku: Sonnet for JAX.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. *CoRR*, abs/2203.15556.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. 2021. Perceiver IO: A general architecture for structured inputs & outputs. *CoRR*, abs/2107.14795.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4904–4916. PMLR.

R. Keys. 1981. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1153–1160.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Maithilee Kunda and Irina Rabkina. 2020. Creative captioning: An ai grand challenge based on the dixit board game.

Angeliki Lazaridou and Marco Baroni. 2020. Emergent multi-agent communication in the deep learning era. *CoRR*, abs/2006.02419.

Angeliki Lazaridou, Anna Potapenko, and Olivier Tieleman. 2020. Multi-agent communication meets natural language: Synergies between functional and structural language learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7663–7674, Online. Association for Computational Linguistics.

Jason Lee, Kyunghyun Cho, and Douwe Kiela. 2019. Countering language drift via visual grounding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4385–4395, Hong Kong, China. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Jessica Mankewitz, Veronica Boyce, Brandon Waldon, Georgia Loukatou, Dhara Yu, Jesse Mu, Noah Goodman, and Michael Frank. 2021. Multi-party referential communication in complex strategic games.

Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2022. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1878–1898, Dublin, Ireland. Association for Computational Linguistics.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation. *CoRR*, abs/2202.00666.

Ron Mokady, Amir Hertz, and Amit H. Bermano. 2021. Clipcap: CLIP prefix for image captioning. *CoRR*, abs/2111.09734.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents.

JL Roubira. 2008. Dixit.[board game].

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-Diagnosis and Self-Debiasing: A Proposal for Reducing Corpus-Based Bias in NLP. *Transactions* *of the Association for Computational Linguistics*, 9:1408–1424.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, S. M. Ali Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. In *Advances in Neural Information Processing Systems*, volume 34, pages 200–212. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Natalia Vélez, Sophie Bridgers, and Hyowon Gweon. 2019. The rare preference effect: Statistical information influences social affiliation judgments. *Cognition*, 192:103994.

Lisa Wang, Angela Sy, Larry Liu, and Chris Piech. 2017. Learning to represent student knowledge on programming exercises using deep learning. *International Educational Data Mining Society*.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William S. Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2021. Ethical and social risks of harm from language models. *CoRR*, abs/2112.04359.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747.

Erica J Yoon, Michael Henry Tessler, Noah D Goodman, and Michael C Frank. 2016. Talking with tact: Polite language as a balance between kindness and informativity. In *Proceedings of the 38th Annual Conference of the Cognitive Science Society*, pages 2771–2776. Cognitive Science Society.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences.

A Speaker Model Details

We provide further detail on our speaker model architecture.

The pretrained-and-frozen CLIP visual encoder takes in images of dimension $224 \times 224 \times 3$, which we denote x. We take the pre-spatial-pool output from the model, which has dimensions $16 \times 16 \times 768$, and flatten this (across spatial dimensions) to an output $E(x)$, with dimension 256×768. For the specific encoder model, we experimented with a few (pretrained) options (see Appendix B.1).

The attention-based (QKV) adapter takes in this visual embedding and transforms it into n token embeddings that can be used as a prefix to prompt the language model. The trainable parameters of this layer are Q (of dimension $n \times d$), W_K (of dimension $768 \times d$), and W_V (of dimension 768×2048). The weights W_K, W_V are used to compute $K = E(x)W_K, V = E(x)W_V$. The fixed queries are then used to attend to these input-dependent keys and values. The output of the adapter layer is thus $A(x) = softmax(QK^\top/\sqrt{d})V$ (of dimension $n \times 2048$).

We feed in $A(x)$ as n prefix embeddings of dimension 2048 to a pretrained-and-frozen causal Transformer to generate up to 32 tokens (with early termination if the EOS token is produced). We follow the recommended architecture parameters presented in Hoffmann et al. (2022) for a 1.4B parameter model: our transformer has 24 layers, model dimensionality of 2048, and 16 heads. We use a SentencePiece tokenizer with a vocabulary size of 32000 (Kudo and Richardson, 2018).

B Training details

In this section, we provide training details, hyperparameter search details, and the hyperparameters we used for our final results. All models were implemented in Python using JAX (Bradbury et al., 2018) and Haiku (Hennigan et al., 2020). Training was distributed over 16 TPUs (v3), and all experiments used a batch size of 128 (unless specified otherwise). The optimizer for all experiments is Adam (Kingma and Ba, 2015), with $\beta_1 = 0.9, \beta_2 = 0.95$. We use ZeRO stage-one parameter sharding (Rajbhandari et al., 2020).

B.1 Speaker model pretraining on captioning data

We pre-trained our speaker model using supervised cross-entropy loss (with teacher forcing (Williams and Zipser, 1989)) on the Conceptual Captions dataset (Sharma et al., 2018) which consists of paired image-caption data.

Dataset images were augmented using random crops. All experiments were run with a batch size of 512 for 500000 steps. Training was distributed over 16 TPUs (v3) in a data parallel fashion.

Hyperparameters we searched over were:

- Base CLIP model: We tried the smaller CLIP ViT-B/32 and the larger CLIP ViT-L/14 model.
- Positional embeddings: We experimented with adding an absolute positional embedding to each of the 16×16 unpooled outputs from the CLIP encoder.
- Learning rate: [1e-4, 3e-4, 1e-3, 3e-3, 1e-2, 3e-2, 1e-1, 3e-1]
- Dimension of QKV adapter (d): [32, 64]
- Number of input tokens to frozen language model (n): [8, 16, 32]

All hyperparameter selections were based on validation loss on the Conceptual Captions validation set. We found that the larger CLIP model, no additional positional embeddings, a learning rate of 3e-2, $d = 32$, $n = 32$, worked best. In terms of the biggest factors, using the larger CLIP model made the biggest change, followed closely by learning rate.

We froze the best model, and that same model served as the starting point for all our experiments.

B.2 Choice to finetune just the adapter

We experimented with finetuning a visual encoder from scratch (instead of using a pre-trained CLIP) as done by Tsimpoukelli et al. (2021). However, we found this approach to perform worse than our adapter-only training (in terms of Conceptual Captions validation performance). The adapter has just under 1.6M parameters, out of a total of 1.7B parameters, for a total of < 0.1% finetuned parameters. We believe that finetuning such a small number of weights prevents language degeneration, thus enhancing the quality of generated captions. For downstream experiments, we found a similar effect – finetuning more than just the adapter parameters led to more language drift. For this reason, we only finetune the adapter in all experiments.

B.3 Multi-listener, contrastive-reward training

We used the hyperparameters detailed in Table 2 for training our main models. These hyperparameters were found by grid searches over learning rate ([3e-5, 3e-4]), sampling temperature ([1,2]), nucleus size ([0.8, 1]), and caption length penalty ([1e-5, 3e-5, 1e-4, 3e-4, 1e-3, 3e-3, 1e-2, 3e-2]). We found that caption length penalty was crucial to tune to make sure it didn't dominate the reward at the start of training.

All of our final training results are run on 5 different random seeds, from which 95% confidence intervals are calculated and shown (in figures and tables).

B.4 Single listener, contrastive-reward training

For this baseline, we train the speaker to just maximize the binary accuracy of the single unperturbed listener. We train in 3 settings: on cFMNIST, on tcFMNIST, and on COCO. To make a fair comparison, we performed a hyperparameter search for each case (final hyperparameters shown in Table 3). Specifically, we searched over learning rate ([3e-5, 3e-4]), nucleus size ([0.8, 1]), and caption length penalty ([1e-5, 1e-4, 1e-3, 1e-2]). We also found that early stopping and batch-based baseline subtraction (Williams, 1992) improved performance (in terms of reward on unseen images), so we used both of these techniques for all results reported. On each setting, we trained 5 random seeds for up to 25000 iterations on FMNIST and up to 40000 iterations on COCO, early stopped each one, and computed evaluation metrics on those checkpoints.

B.5 Multi listener, non-contrastive-reward training

For this baseline, we train the speaker to just maximize the difference in the ALIGN match scores that listeners assign to the cued image. In this setup, the listeners only need the caption and cued image, which is why we call it non-contrastive. We train this baseline in all four settings that we train our main (multi-listener, contrastive-reward) model in. We perform hyperparameter over nucleus size ([0.8, 1]) and caption length penalty ([1e-7, 1e-6, 3e-6, 1e-5, 3e-5, 1e-4, 3e-4, 1e-3]), with optimal hyperparameters reported in Table 4.

C Dataset details

To construct the cFMNIST dataset, we convert FMNIST images to HSV. Since the images are originally grayscale, the saturation is always 0. We set the saturation to 1, and set the hue to one of the 8 colors shown in Table 5. Then, we convert back to RGB images and resize to spatial dimensions 224×224 using cubic upsampling (Keys, 1981), then clip all values to the range [0,1].

For tcFMNIST, we randomly place a cFMNIST image in the bottom left and in the top-right of the image. This choice of tiling (as opposed to top-left, bottom-right, or some other combination) was chosen to avoid an inherent bias towards cuing for the top-left object that we observed in our base captioning model.

For the test set, we construct a fixed test set to use for equal comparison across all conditions. For contrastive-reward experiments, we also fix the distractors during test time for consistency.

We normalize all images according to the normalization that was originally used for the off-the-shelf image encoders we use (for consistency). Specifically, for inputs to the speaker, which pass through CLIP's vision encoder, the normalization mean and standard deviation values are (0.481, 0.458, 0.408) and (0.269, 0.261, 0.278), respectively. For inputs to the listeners, which pass through ALIGN's vision encoder, the normalization mean and standard deviation values are (0.485, 0.456, 0.406) and (0.229, 0.224, 0.225), respectively.

C.1 Licenses

We use FMNIST under the MIT license, COCO under the creative commons 4.0 attribution license, and Conceptual Captions under its ad-hoc license (for which we thank Google LLC). We also acknowledge that images in Figure 1 are taken from the Dixit game (Roubira, 2008) and icons are used under the Flaticon license (for which we thank Flaticon.com). We make use of the GPT3 (Brown et al., 2020) beta (for quantification of structural drift, see Appendix E) under the Apache License.

To our knowledge, we use these resources in agreement with their licenses.

C.2 Dataset Splits

We use standard splits for all datasets from tensorflow datasets (Abadi et al., 2015). For FMNIST-based datasets, this means 60000 train images that

	FMNIST			COCO
Parameter	Crop	Blur	Grayscale	COCO Grayscale
Learning rate	3e-4	3e-4	3e-4	3e-4
Sampling temperature	1	1	1	1
Nucleus Size	0.8	0.8	1	1
Caption length penalty (λ)	3e-3	3e-3	3e-3	3e-4

Table 2: Hyperparameters for the main setting (multi-listener, contrastive-reward).

Parameter	cFMNIST	tcFMNIST	COCO
Learning rate	3e-4	3e-4	3e-4
Sampling temperature	1	1	1
Nucleus Size	0.8	0.8	1
Caption length penalty (λ)	1e-5	1e-5	1e-5

Table 3: Hyperparameters for the single listener, contrastive-reward baseline.

are used to procedurally generate cFMNIST and tcFMNIST train sets, and 10000 test images that are used to generate the test sets.

D Evaluation metric details

We have two types of metrics: color metrics and keyword metrics. Color metrics apply on both FMNIST-based datasets and COCO, while keyword metrics are only used on FMNIST-based dataset (where we know exactly where and what the objects are by construction). We want our metrics to be able to diagnose language specialization, and also check that the captions are still grounded (we'll refer to this as "image relevance" of captions).

D.1 Color metrics

All color metrics utilize the following set of 16 colors:

```
red, orange, yellow, green,
blue, indigo, violet, purple,
cyan, magenta, pink, brown,
black, white, gray, grey
```

For measuring language specialization to colors, we define *color prevalence*: the fraction of captions at test time containing at least one word from the above list.

For measuring image relevance, we use a proxy metric as its often hard to define what it means for a color to be correct (e.g., if an image is "cyan" and the model says "blue and white"). The proxy metric we use is *color diversity*. To compute color diversity we calculated TF-IDF vectors for each caption using only the color terms above. Then, we calculated the trace of the covariance matrix (which has dimension 16×16) of these vectors. This metric has the desirable property where if a color appears in nearly all captions, it will be scaled down by the IDF term and so will its contribution to the trace of the covariance matrix.

We found this to be a good proxy metric as loss of image relevance in color specialization cases corresponding to the model choosing just a few colors (often just one) and using them to describe all images. To not confound our main results, we qualitatively looked at how well this metric corresponded to loss of image relevance on the many language prompts detailed in Appendix G. We found that color diversity was a noisy, but useful, metric for determining image relevance of these prompts. For example, a simpler metric like checking the number of unique captions the model uses fails since, given some language prompts, the speaker would produce long strings of colors, often in slightly different orders (e.g. "color: green, black, white, blue, red ..." vs. "color: red, black, white, blue, green ..."). Furthermore, we note that color diversity is not a perfect metric, so subtle differences (on the order of ± 0.005) should not be considered significant. We mainly use color diversity to classify runs where image relevance was completely lost (color diversity going to 0), or image relevance was retained (color diversity staying near that of the best language prompts and above that at the start of training).

	FMNIST			COCO
Parameter	Crop	Blur	Grayscale	COCO Grayscale
Learning rate	3e-4	3e-4	3e-4	3e-4
Sampling temperature	1	1	1	1
Nucleus Size	0.8	0.8	0.8	0.8
Caption length penalty (λ)	3e-6	3e-6	3e-6	3e-4

Table 4: Hyperparameters for the multi-listener, non-contrastive-reward baseline.

Color:	Red	Orange	Yellow	Green	Cyan	Blue	Purple	Pink
Hue:	$\frac{0}{360}$	$\frac{30}{360}$	$\frac{60}{360}$	$\frac{120}{360}$	$\frac{180}{360}$	$\frac{240}{360}$	$\frac{270}{360}$	$\frac{300}{360}$

Table 5: Color and Hues for cFMNIST.

D.2 Keyword metrics

For measuring language specialization to objects, we define *FMNIST keyword prevalence*: the fraction of captions at test time containing at least one FMNIST related keyword.

To measure image relevance, we utilize the ground-truth labels from FMNIST, supplemented with synonyms. Specifically, we measure *object prevalence*: the fraction of captions at test time that contain at least one keyword corresponding to the ground truth label of the object in, the image according to Table 6. We found that allowing synonyms was essential, as the captioning model heavily prefers some clothing words over others (e.g., sandals are almost always just called shoes by the model). For tcFMNIST, we similarly define *bottom-left object prevalence* and *top-right object prevalence* to measure what fraction of captions refer to each region of the image.

E Language drift quantification

Prior work in emergent communication (Lazaridou et al., 2020) establishes three types of language drift that may occur when adapting language from rewards, without direct supervision: structural drift (how "language-like" are captions), semantic drift (how "image-relevant" are captions), and pragmatic drift (how "human-interpretable" are captions). While the focus of our work is on language specialization, it is important to investigate to what extent our approach is resulting in language drift.

For semantic and pragmatic drift, we note that our main metrics (see Section 3.4 and Appendix D) measure human interpretability as well as image relevance. For example, our object metrics measure whether the model is referring to objects using the category name or valid synonyms, and our color diversity metric is able to differentiate settings with notable semantic drift (e.g., the non-contrastive baseline, see Section 4.4.2) from settings without notable semantic drift (e.g., our main results, see Figures 3 and 4). Our method preserves these metrics, presumably since the grounded task with frozen listeners, as well as the frozen components of the speaker, provide strong constraints on what language the speaker can use.

However, our main metrics do not adequately address the issue of structural drift. To measure how "language-like" our captions are, we therefore follow the approach of Lazaridou et al. (2020) and evaluate the log-likelihood assigned to generated captions by an independent, pretrained language model (LM). Specifically, we use OpenAI's Ada model, made available online through the GPT3 beta (Brown et al., 2020). For comparison, we also show LM log-likelihoods for the ground truth human captions (for cFMNIST, we procedurally generate these—e.g., "red pullover"), the LM likelihoods for the "best prompt" (see Appendix G), and the LM likelihoods for captions from the speaker before specialization training (the base speaker pretrained on captioning only). For reference, the "best" language prompts were chosen to maximize task-relevant metrics (see third column in Tables 13-16) except for the (cFMNIST, Grayscale) case where we use the third best (as the top two prefixes have very low color diversity—see discussion in Section 4.4.3). Results are shown in Table 7.

Surprisingly, we see that in most of our runs, average caption likelihoods seem to increase as the speaker specializes. It appears that this effect may be driven by the length penalty—over training,

Label	Category name	Added synonyms
0	t-shirt	top, t-shirts, shirt, shirts
1	trouser	trousers, pants
2	pullover	sweater, hoodie, sweaters, hoodies
3	dress	dresses
4	coat	coats, jacket, jackets
5	sandal	high heels, heels, shoe, shoes
6	shirt	shirts
7	sneaker	sneakers, shoe, shoes, running shoe
8	bag	purse, backpack, bags, purses
9	ankle boot	boot, shoe, shoes, boots

Table 6: FMNIST keyword sets. We refer to the set of all words in this table as FMNIST related keywords.

	tcFMNIST, Crop	cFMNIST, Blur	cFMNIST, Grayscale	COCO, Grayscale
Ground truth	-32.77	**-15.20**	-15.20	-46.26
“Best” language prompt	-36.61	-44.61	-24.59	-29.49
Start of training	-33.33	-28.64	-28.64	**-28.44**
End of training	**-17.41**	-19.01	**-9.26**	-29.49

Table 7: Average language model *full-caption* log-likelihoods for ground truth, start-of-training, and end-of-training captions on test images.

the speaker produces shorter captions, which have higher likelihoods since they have fewer tokens. Qualitatively, we find that language drift can be fairly variable across different random seeds. For example, the COCO average is worse after training largely due to a single run, in which long repeated captions emerged (which have lower likelihoods than their un-repeated counterparts).

To evaluate this further, we computed the per-token likelihoods (Table 8). While these do show some decrease in likelihood in some cases, the captions at the end of training are generally of comparable likelihood to the ground truth. In this instance, we notice that the one COCO seed in which the captions repeated actually exhibits *greater* per-token likelihood—after a few repeats, the LM starts to estimate further repeats to be very likely. This could potentially be a concern for using LM likelihoods as a metric for structural drift more broadly (cf. Lee et al., 2019). However, in most runs our length penalty prevents repetitions or long captions, as noted above.

In summary, these results indicate that language has not drifted far in most conditions and for most random seeds. We attribute this to some of the same factors that help prevent other types of language drift: finetuning a small part of our speaker (just the adapter), keeping the listener models frozen (thus avoiding co-adaptation), using a length penalty (see above discussion), and using contrastive reward (crucial for combating semantic drift, see Section 4.4.2). If necessary, language drift could potentially be reduced further by using a KL-divergence loss to the distribution of outputs from the base captioning model (as done by Lazaridou et al. (2020)).

F Full ablation results

In this section we show the full quantitative ablation results, in Tables 9, 10, 11, and 12.

	tcFMNIST, Crop	cFMNIST, Blur	cFMNIST, Grayscale	COCO, Grayscale
Ground truth	-5.823	-7.666	-7.666	-4.772
“Best” language prompt	-4.680	-4.371	**-4.076**	-4.051
Start of training	**-4.160**	**-4.127**	-4.127	-4.463
End of training	-7.143	-8.383	-4.234	**-2.509**

Table 8: Average language model *per-token* log-likelihoods for ground truth, start-of-training, and end-of-training captions on test images.

Condition	Bottom-left object prevalence	Top-right object prevalence
No training	0.290	0.312
Single listener	0.475 ± 0.024	0.473 ± 0.020
Non-contrastive	0.360 ± 0.005	**0.120 ± 0.011**
Full training	**0.602 ± 0.066**	0.135 ± 0.017

Table 9: Metrics (with 95% CI) for ablations on the tcFMNIST dataset and the (identity, crop) pair of listener transformations. Both non-contrastive and full training lead to decreased use of top-right keyword, but only full training accurately specializes to using the bottom-left keyword.

Condition	FMNIST keyword prevalence	Object prevalence	Color prevalence	Color diversity
No training	0.644	0.442	0.556	0.014
Single listener	0.754 ± 0.020	0.564 ± 0.013	0.953 ± 0.022	0.016 ± 0.000
Non-contrastive	**0.997 ± 0.004**	**0.710 ± 0.086**	0.606 ± 0.833	**0.000 ± 0.000**
Full training	0.600 ± 0.044	0.418 ± 0.038	**0.004 ± 0.008**	0.001 ± 0.002

Table 10: Relevant metrics (with 95% CI) for various ablations on the cFMNIST dataset and the (identity, blur) pair of listener transformations. Non-contrastive rewards perform better than contrastive rewards in this setting, as evidenced by increased use of correct keywords, and decreased use of meaningful colors (diversity of colors goes to 0).

Condition	Color prevalence	Color diversity	FMNIST keyword prevalence	Object prevalence
No training	0.556	0.014	0.644	0.442
Single listener	0.953 ± 0.022	**0.016 ± 0.000**	0.754 ± 0.020	0.564 ± 0.013
Non-contrastive	1.000 ± 0.000	0.005 ± 0.003	0.000 ± 0.000	0.000 ± 0.000
Full training	**1.000 ± 0.000**	0.015 ± 0.002	**0.000 ± 0.000**	**0.000 ± 0.000**

Table 11: Relevant metrics (with 95% CI) for various ablations on the cFMNIST dataset and the (identity, grayscale) pair of listener transformations. Full training performs best as color prevalence and diversity increase, while keyword prevalence decreases to 0.

	Color prevalence	Color diversity
No training	0.027	0.022
Single listener	0.102 ± 0.080	0.030 ± 0.007
Non-contrastive	0.932 ± 0.256	0.018 ± 0.005
Full training	$\mathbf{0.988 \pm 0.023}$	$\mathbf{0.033 \pm 0.027}$

Table 12: Relevant metrics (with 95% CI) for various ablations on the COCO dataset and the (identity, grayscale) pair of listener transformations. Full training performs best as it has highest color prevalence and diversity.

G Extended results on caption prefixes

In Tables 13-16, we show all caption prefixes we experimented with, as well as the relevant metrics for each. For each table, we sort prefixes from worst (top of table) to best (bottom of table) based on the most task-relevant keyword metric (third column in each table). We added a column for "Score difference" which is the average difference in ALIGN match scores from each listener for the cued image (it corresponds to the reward that is seen in the non-contrastive reward baseline). Of course, no training occurs, as the caption prefixes just explore how well the speaker can do by just explicit prompting. We also use ␣to indicate a space at the end of the caption. For most prefixes, we see a large difference between adding this space and not adding it, which is just another testament to how brittle prompt engineering can be.

Caption prefix	Accuracy difference	Score difference	Bottom-left object prevalence	Top-right object prevalence	FMNIST keyword prevalence
the bottom left of this picture is␣	0.0121	0.0061	0.0181	0.0193	0.0502
the bottom left of this image is␣	0.0052	0.0065	0.0241	0.0277	0.0794
in the bottom left of this image, there is␣	0.0512	0.0112	0.0663	0.0719	0.1688
bottom left of this image:␣	0.0155	0.0096	0.0711	0.0798	0.2604
the bottom left of this image shows␣	0.0338	0.0137	0.0993	0.1212	0.3179
bottom left:␣	0.0418	0.0439	0.1154	0.1274	0.4156
in the bottom left of this image, there is	0.0681	0.0262	0.2711	0.2954	0.6176
the bottom left of this image is	0.1085	0.0293	0.3027	0.3193	0.8302
the bottom left of this picture is	0.1067	0.0365	0.3189	0.3420	0.8320
the bottom left of this image shows	0.1256	0.0515	0.3428	0.3342	0.6091
bottom left of this image:	0.1113	0.0340	0.3561	0.3772	0.7671
bottom left:	0.1093	0.0702	0.3597	0.3770	0.7886

Table 13: Various metrics for prompted generation of base captioning model on tcFMNIST. Accuracy and score difference are calculated on the (unperturbed, crop) pair of listeners.

Caption prefix	Accuracy difference	Score difference	Object prevalence	FMNIST keyword prevalence	Color prevalence	Color diversity
the clothing item in this image is␣	0.0174	0.0491	0.0685	0.1390	0.0801	0.0083
an image of an object:␣	0.0265	0.0060	0.0743	0.0981	0.0755	0.0103
a black and white image of␣	0.0493	-0.0071	0.1065	0.1439	1.0000	0.0044
the item in this image is␣	0.0335	0.0254	0.1147	0.1493	0.0532	0.0083
the object in this image is␣	0.0361	-0.0008	0.1321	0.1664	0.0791	0.0104
item:␣	0.0280	0.0433	0.1627	0.2210	0.3257	0.0131
object:␣	0.0558	-0.0017	0.2274	0.3169	0.3764	0.0141
a picture of␣	0.0664	0.0044	0.2350	0.3213	0.4122	0.0155
an image of␣	0.0824	0.0103	0.2646	0.3684	0.4778	0.0157
a black and white image of	0.1170	0.0362	0.3895	0.5977	1.0000	0.0050
the item in this image is	0.0988	0.0449	0.4014	0.5656	0.3498	0.0136
a picture of	0.0997	0.0518	0.4363	0.6556	0.4765	0.0165
an image of	0.1059	0.0466	0.4428	0.6638	0.5181	0.0163
the object in this image is	0.0885	0.0285	0.4438	0.6146	0.4448	0.0155
an image of an object:	0.0967	0.0363	0.4523	0.6343	0.4434	0.0156
object:	0.1183	0.0018	0.4545	0.6533	0.5643	0.0143
item:	0.1015	0.0528	0.4620	0.6731	0.5747	0.0134
the clothing item in this image is	0.1086	0.0546	0.4692	0.8643	0.5157	0.0142

Table 14: Various metrics for prompted generation of base captioning model on cFMNIST. Accuracy and score difference are calculated on the (unperturbed, blur) pair of listeners.

Caption prefix	Accuracy difference	Score difference	Color prevalence	Color diversity	Object prevalence	FMNIST keyword prevalence
color color color:␣	0.0180	0.0264	0.1421	0.0055	0.0330	0.0433
the colors in this image are␣	0.0139	0.0341	0.1697	0.0082	0.0040	0.0045
a picture with the color␣	0.0584	0.0410	0.2206	0.0139	0.2622	0.3536
a picture with the color	0.0851	0.0507	0.2745	0.0144	0.3423	0.4613
a picture with color␣	0.0347	0.0395	0.2891	0.0141	0.1583	0.2319
color:␣	0.0507	0.0320	0.3000	0.0060	0.0387	0.0501
the color of this image is␣	0.0318	0.0341	0.3648	0.0145	0.0058	0.0064
an image with the color␣	0.0782	0.0302	0.3862	0.0153	0.2468	0.3197
an image with the color	0.0952	0.0553	0.3905	0.0153	0.2743	0.3579
an image with color␣	0.0268	0.0308	0.4225	0.0136	0.1661	0.2413
the colors in this image are	0.0587	0.0511	0.4336	0.0063	0.0490	0.0547
a picture with color	0.0452	0.0398	0.4396	0.0111	0.1848	0.2455
colors in image:␣	0.0097	0.0397	0.5018	0.0020	0.0002	0.0004
color color color:	0.0764	0.0572	0.5512	0.0033	0.1996	0.2535
an image colored␣	0.0919	0.0426	0.5901	0.0123	0.1259	0.1593
an image with color	0.0504	0.0410	0.6026	0.0112	0.1863	0.2555
an image colored	0.1303	0.0631	0.8600	0.0101	0.0200	0.0271
the color of this image is	0.1735	0.0667	0.9010	0.0198	0.0002	0.0002
color:	0.1180	0.0583	0.9705	0.0040	0.1110	0.1354
colors in image:	0.0477	0.0428	0.9994	0.0012	0.0000	0.0000

Table 15: Various metrics for prompted generation of base captioning model on cFMNIST. Accuracy and score difference are calculated on the (identity, grayscale) pair of listeners. For reference, the color diversity of the fully specialized speaker in this case is 0.015, which is on par with the best color diversity values across prompts.

Caption prefix	Accuracy difference	Score difference	Color prevalence	Color diversity
an image colored	0.0016	0.0253	0.0109	0.0110
a picture with the color	0.0055	0.0136	0.0117	0.0144
a picture with color␣	-0.0039	0.0023	0.0172	0.0155
the color of this image is␣	-0.0094	-0.0071	0.0195	0.0180
color color color:␣	-0.0086	0.0045	0.0219	0.0172
an image colored␣	-0.0195	0.0166	0.0242	0.0190
an image with color␣	-0.0141	-0.0010	0.0242	0.0182
the colors in this image are	-0.0008	0.0131	0.0312	0.0171
colors in image:␣	-0.0055	-0.0023	0.0344	0.0128
an image with the color	0.0000	0.0115	0.0375	0.0221
a picture with the color␣	0.0031	0.0040	0.0383	0.0223
color:␣	0.0078	0.0079	0.0508	0.0224
an image with the color␣	0.0055	-0.0021	0.0586	0.0261
color color color:	-0.0016	0.0071	0.0906	0.0278
a picture with color	-0.0016	0.0053	0.0914	0.0165
the colors in this image are␣	-0.0156	0.0091	0.1172	0.0317
the color of this image is	-0.0102	0.0012	0.1352	0.0362
an image with color	0.0008	0.0029	0.1477	0.0167
colors in image:	0.0188	0.0052	0.3352	0.0179
color:	0.0125	0.0047	0.5406	0.0304

Table 16: Various metrics for prompted generation of base captioning model on COCO. Accuracy and score difference are calculated on the (identity, grayscale) pair of listeners. For reference, the color diversity of the fully specialized speaker in this case is 0.033, which is on par with the best color diversity values across prompts.

H Sample hands across experimental conditions

In this section we present representative Dixit hands with the corresponding captions produced by the specialized speaker (after training) in our four main conditions, along with the corresponding distractors, listener match scores and rewards. We chose the speaker seed which achieved the highest overall score in that condition (though results were generally comparable across seeds), and to avoid cherry-picking examples we show the first four evaluation hands (from our randomly ordered evaluation) that contained a distinct target object (for example, in the crop condition, the first four that had a distinct object category in the bottom left).

We present samples for the different transformation conditions in Figures 6-9. Overall, the speaker adapts to the target difference between the listeners, and exhibits relatively mild language drift—the utterances generally stay grounded and human-interpretable, but in some cases exhibit some repetition or odd grammar. Some potential methods for further reducing these issues are noted above.

Below-chance performance for the grayscale listener: In Figure 4c, the grayscale listener achieves below-chance performance. We note from the samples (Figure 8) that the speaker sometimes uses the word "background" in addition to using a color. This noun is not informative (so does not notably hurt the listener that can see color), but we speculate that it may throw off the second listener as some images have more background than others (e.g. a grayscale image of a high heel has more black than a grayscale image of a jacket). Note that the speaker does not see the distractors, so it can't purposefully consider the distractors to exploit this effect. This effect appears to be an artifact of the particular dataset we used, but does illustrate another intriguing way the speaker can exploit differences between listeners.

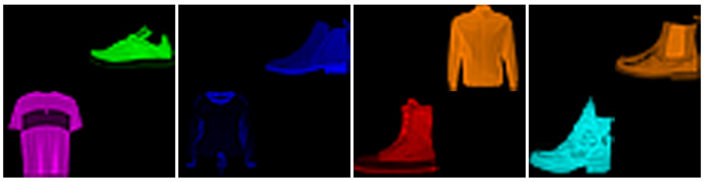
idx:	0	1	22	38
Listener 0:	0.17074655	**0.19805801**	0.05801037	0.060537737
Listener 1:	0.09845241	0.0371554	0.053720675	**0.100063615**

Human answers: red trouser and cyan ankle boot

Model answer: a pair of boots

Model raw answer: a pair of boots

Reward: 0.0



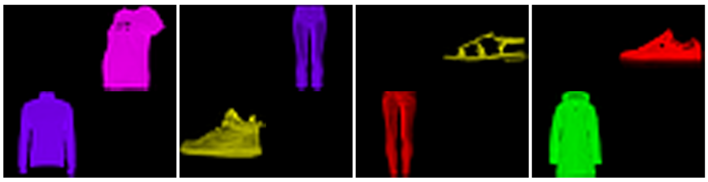
idx:	2	83	1	44
Listener 0:	**0.18275425**	0.11489111	0.09569048	0.066219226
Listener 1:	0.07743783	0.015351542	**0.08295242**	0.03236021

Human answers: green sneaker and pink tshirt

Model answer: a t shirt

Model raw answer: a t - shirt

Reward: 1.0



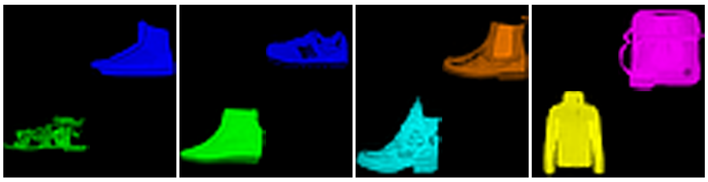
idx:	5	110	104	124
Listener 0:	**0.18787107**	0.09847662	0.09847906	0.15232804
Listener 1:	0.067900255	**0.0799696**	0.07212607	0.04338998

Human answers: pink tshirt and purple pullover

Model answer: a sweater

Model raw answer: a sweater

Reward: 1.0



idx:	8	120	44	28
Listener 0:	**0.0941813**	0.053985488	0.04671259	0.017789248
Listener 1:	**0.06341467**	0.04782959	0.020819351	0.015794737

Human answers: blue sneaker and green sandal

Model answer: a boat

Model raw answer: a boat

Reward: 0.0

idx:	9	58	79	25
Listener 0:	**0.15588474**	0.084171094	0.09770149	0.046457417
Listener 1:	0.05716432	**0.06388364**	0.04683443	-0.007204691

Human answers: yellow pullover and green dress

Model answer: a red dress

Model raw answer: a red dress

Reward: 1.0

Figure 6: Speaker samples and listener results on tcFMNIST, after the speaker has specialized to the second listener having the crop transformation. The language generally specifies the bottom left object, with a grounding failure in the second-to-last case (or perhaps a reference to boat sandals?).

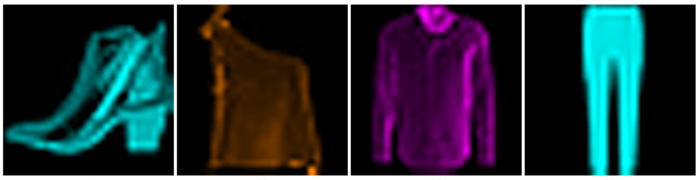
idx:	0	1	22	38
Listener 0:	**0.23596007**	0.092971906	0.100117944	0.09206205
Listener 1:	0.08432734	0.06270674	**0.08633807**	0.068054974

Human answers: red trouser

Model answer: a jeans

Model raw answer: a jeans

Reward: 1.0



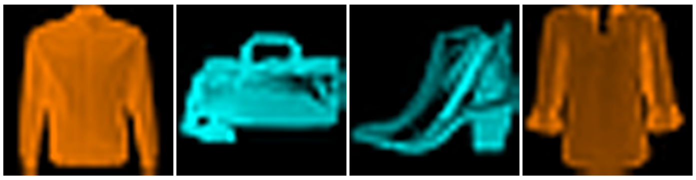
idx:	1	36	88	86
Listener 0:	**0.15651584**	0.06549272	0.025484744	0.057346113
Listener 1:	0.031021954	0.037125766	0.041744716	**0.04599671**

Human answers: cyan ankle boot

Model answer: a pair of sandals

Model raw answer: a pair of sandals

Reward: 1.0

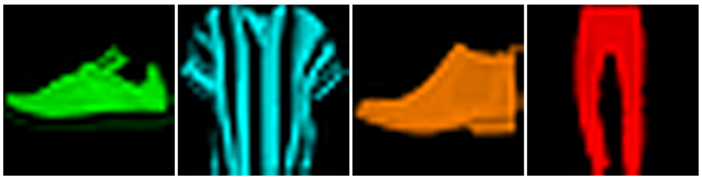


idx:	2	83	1	44
Listener 0:	**0.12716034**	0.04767417	0.04322685	0.072259
Listener 1:	0.01679665	0.00828971	-0.00082306995	**0.03381651**

Human answers: orange coat

Model answer: a man s hooded hooded hooded hooded hooded hooded

Model raw answer: a man 's hooded hooded hooded hooded hooded hooded



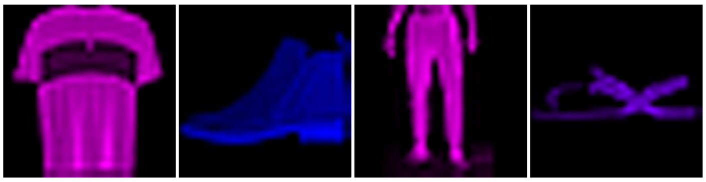
idx:	4	9	97	95
Listener 0:	**0.19682448**	0.043157794	0.1453603	0.07618693
Listener 1:	0.027312862	0.022492867	**0.047557253**	0.04619919

Human answers: green sneaker

Model answer: a pair of sneakers

Model raw answer: a pair of sneakers

Reward: 1.0



idx:	5	110	104	124
Listener 0:	0.081165716	0.07948837	**0.10108026**	0.09919922
Listener 1:	0.11145205	0.10159122	**0.11682075**	0.11153849

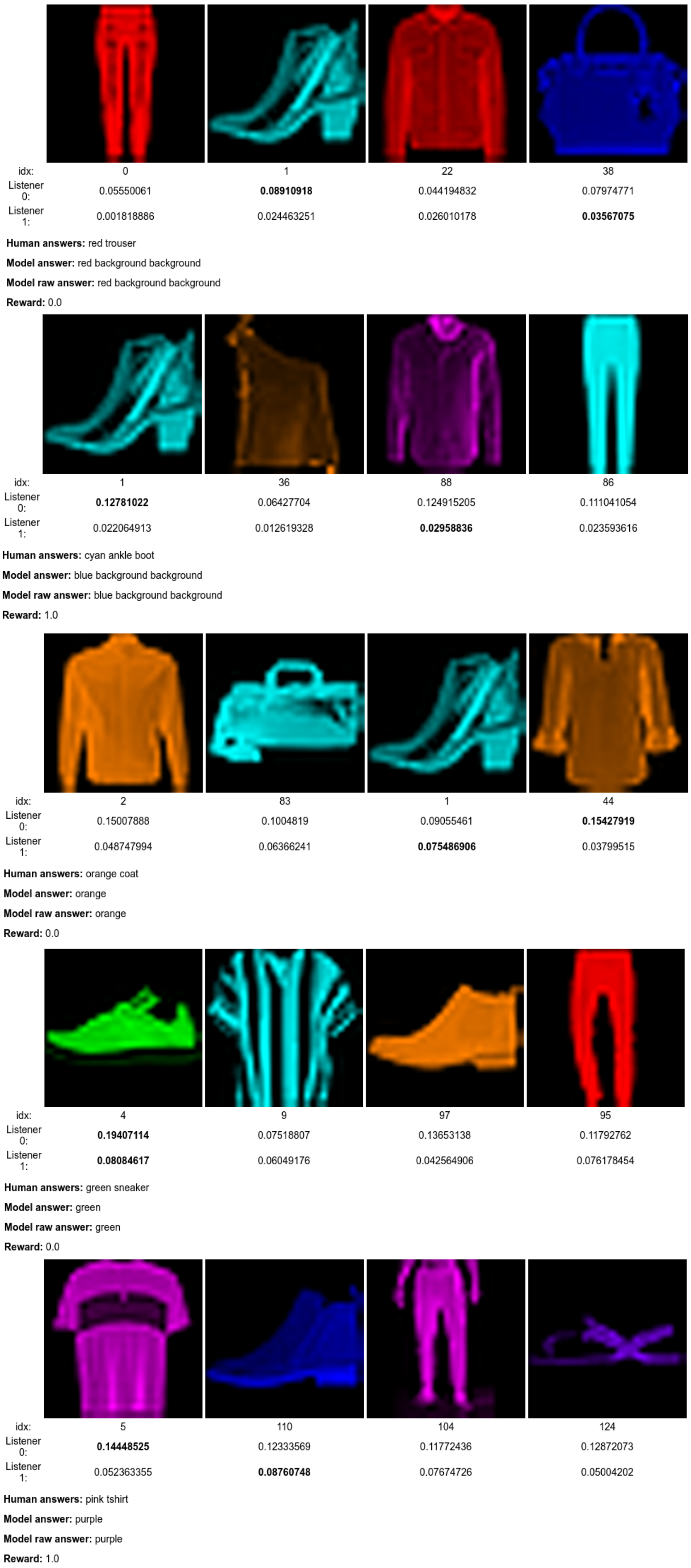
Human answers: pink tshirt

Model answer: a

Model raw answer: a

Reward: 0.0

Figure 7: Speaker samples and listener results on cFMNIST, after the speaker has specialized to the second listener having the blur transformation. The speaker generally ignores colors and names the object as intended, but with moderate language degradation in some cases.



idx:	0	1	22	38
Listener 0:	0.05550061	**0.08910918**	0.044194832	0.07974771
Listener 1:	0.001818886	0.024463251	0.026010178	**0.03567075**

Human answers: red trouser

Model answer: red background background

Model raw answer: red background background

Reward: 0.0

idx:	1	36	88	86
Listener 0:	**0.12781022**	0.06427704	0.124915205	0.111041054
Listener 1:	0.022064913	0.012619328	**0.02958836**	0.023593616

Human answers: cyan ankle boot

Model answer: blue background background

Model raw answer: blue background background

Reward: 1.0

idx:	2	83	1	44
Listener 0:	0.15007888	0.1004819	0.09055461	**0.15427919**
Listener 1:	0.048747994	0.06366241	**0.075486906**	0.03799515

Human answers: orange coat

Model answer: orange

Model raw answer: orange

Reward: 0.0

idx:	4	9	97	95
Listener 0:	**0.19407114**	0.07518807	0.13653138	0.11792762
Listener 1:	**0.08084617**	0.06049176	0.042564906	0.076178454

Human answers: green sneaker

Model answer: green

Model raw answer: green

Reward: 0.0

idx:	5	110	104	124
Listener 0:	**0.14448525**	0.12333569	0.11772436	0.12872073
Listener 1:	0.052363355	**0.08760748**	0.07674726	0.05004202

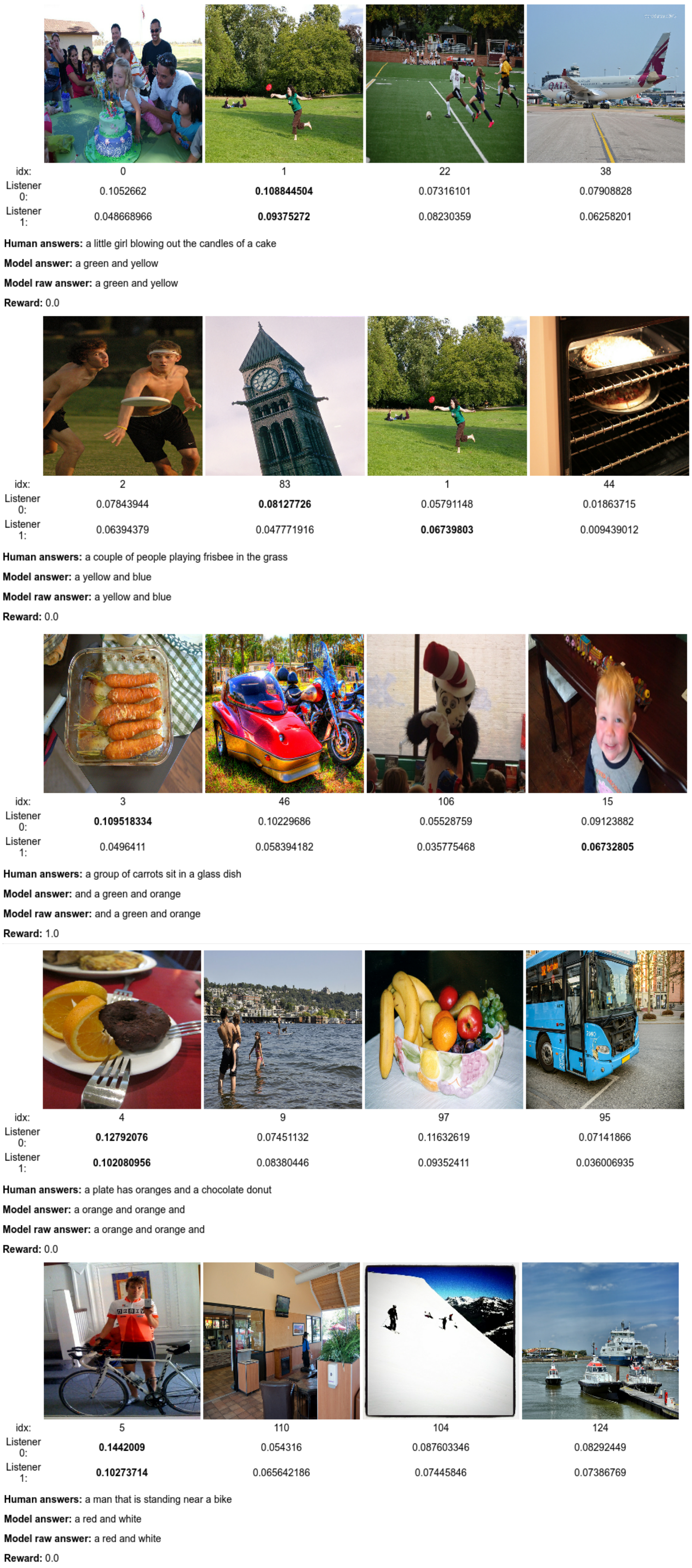
Human answers: pink tshirt

Model answer: purple

Model raw answer: purple

Reward: 1.0

Figure 8: Speaker samples and listener results on cFMNIST, after the speaker has specialized to the second listener having the grayscale transformation. The speaker consistently names the correct color, though it occasionally also repeats "background".



idx:	0	1	22	38
Listener 0:	0.1052662	**0.108844504**	0.07316101	0.07908828
Listener 1:	0.048668966	**0.09375272**	0.08230359	0.06258201

Human answers: a little girl blowing out the candles of a cake

Model answer: a green and yellow

Model raw answer: a green and yellow

Reward: 0.0

idx:	2	83	1	44
Listener 0:	0.07843944	**0.08127726**	0.05791148	0.01863715
Listener 1:	0.06394379	0.047771916	**0.06739803**	0.009439012

Human answers: a couple of people playing frisbee in the grass

Model answer: a yellow and blue

Model raw answer: a yellow and blue

Reward: 0.0

idx:	3	46	106	15
Listener 0:	**0.109518334**	0.10229686	0.05528759	0.09123882
Listener 1:	0.0496411	0.058394182	0.035775468	**0.06732805**

Human answers: a group of carrots sit in a glass dish

Model answer: and a green and orange

Model raw answer: and a green and orange

Reward: 1.0

idx:	4	9	97	95
Listener 0:	**0.12792076**	0.07451132	0.11632619	0.07141866
Listener 1:	**0.102080956**	0.08380446	0.09352411	0.036006935

Human answers: a plate has oranges and a chocolate donut

Model answer: a orange and orange and

Model raw answer: a orange and orange and

Reward: 0.0

idx:	5	110	104	124
Listener 0:	**0.1442009**	0.054316	0.087603346	0.08292449
Listener 1:	**0.10273714**	0.065642186	0.07445846	0.07386769

Human answers: a man that is standing near a bike

Model answer: a red and white

Model raw answer: a red and white

Reward: 0.0

Figure 9: Speaker samples and listener results on COCO, after the speaker has specialized (on COCO) to the second listener having the grayscale transformation. The speaker generally names one or two plausible colors for the images; but these are less discriminative in COCO than in the above results. There is also some minor language degradation in some cases (e.g. “and" at the end of a caption).

Meeting the Needs of Low-Resource Languages: The Value of Automatic Alignments via Pretrained Models

Abteen Ebrahimi[◇] **Arya D. McCarthy**[▽] **Arturo Oncevay**[♡]
Luis Chiruzzo[△] **John E. Ortega**[Ω] **Gustavo A. Giménez-Lugo**[♣]
Rolando Coto-Solano[ϕ] **Katharina Kann**[◇]
[◇]University of Colorado Boulder [▽]Johns Hopkins University [♡]University of Edinburgh
[△]Universidad de la República, Uruguay [Ω]Northeastern University
[♣]Universidade Tecnológica Federal do Paraná [ϕ]Dartmouth College

Abstract

Large multilingual models have inspired a new class of word alignment methods, which work well for the model's pretraining languages. However, the languages most in need of automatic alignment are low-resource and, thus, not typically included in the pretraining data. In this work, we ask: *How do modern aligners perform on unseen languages, and are they better than traditional methods?* We contribute gold-standard alignments for Bribri–Spanish, Guarani–Spanish, Quechua–Spanish, and Shipibo-Konibo–Spanish. With these, we evaluate state-of-the-art aligners with and without model adaptation to the target language. Finally, we also evaluate the resulting alignments extrinsically through two downstream tasks: named entity recognition and part-of-speech tagging. We find that although transformer-based methods generally outperform traditional models, the two classes of approach remain competitive with each other.

1 Introduction

Word alignment is a valuable tool for extending the coverage of natural language processing (NLP) applications to low-resource languages through, e.g., statistical machine translation (SMT; Koehn and Knowles, 2017; Duh et al., 2020) or annotation projection (Yarowsky et al., 2001; Smith and Smith, 2004; Nicolai et al., 2020; Eskander et al., 2020). The traditional approach for generating alignments has been with statistical methods such as Giza++ (Och and Ney, 2003) and FastAlign (Dyer et al., 2013), which provide strong alignment quality while remaining quick and lightweight to run. Recently, new methods have been proposed which extract alignments from massive *pretrained multilingual models*, and outperform these longstanding methods (Dou and Neubig, 2021).

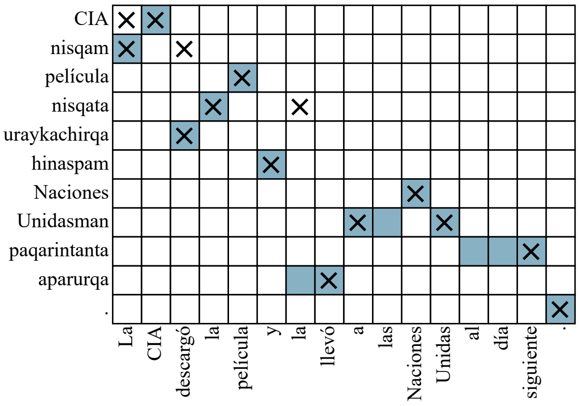


Figure 1: A word alignment between Quechua and Spanish (shaded), as well as mBERT+TLM's predicted alignment (marked by ×'s). FastAlign and Giza++ cannot take advantage of surface features of proper names and borrowings. We evaluate alignments intrinsically via AER and extrinsically with POS-tagging and NER models learned on annotations projected across alignments from Spanish.

However, results on other NLP tasks, such as part-of-speech (POS) tagging and named-entity recognition (NER), have shown that, while pretrained models generally work well out-of-the-box for high-resource languages, performance is far lower for low-resource ones, particularly those which are unseen during pretraining (Pires et al., 2019; Wu and Dredze, 2020; Muller et al., 2021; Lee et al., 2022). Models can be adapted (Gururangan et al., 2020; Chau et al., 2020) to improve performance, but this comes with a large computational cost. Given these two considerations, for *unseen* low resource languages it remains unclear (1) whether modern neural approaches based on adapted pretrained models generate higher-quality alignments than traditional approaches and (2) if so, whether the quality difference is severe enough to justify the additional computational cost.

We investigate this by collecting gold-standard alignments for Bribri, Guarani, Quechua, and

---

Our code and data can be found at `https://github.com/abteen/alignment`.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3912–3926
May 2-6, 2023

Shipibo-Konibo. These languages are low-resource and unrepresented in the pretraining data of popular models—a relevant real-world scenario. In addition to intrinsically evaluating alignment quality, we measure the downstream utility of each method for training POS-tagging and NER models by annotation projection.

We find traditional and neural methods to be competitive, but pretrained models result in slightly lower alignment error rates and stronger downstream task performance, even for initially unseen languages. Through further analysis, we also find that adaptation may be a more reliable approach given minimally available resources. Taken together, these results indicate that alignment from multilingual models can indeed be a valuable tool for low-resource languages, but traditional approaches continue to be a strong option and should still be considered for practical applications.

2 Related Work

Alignment Word alignment is a long studied task, with origins in the IBM models for statistical machine translation (Brown et al., 1993), which are the basis of Giza++ (Och and Ney, 2003) and FastAlign (Dyer et al., 2013). As these approaches can only generate one-to-many alignments, models are trained in both forward and reverse directions (reversing the role of source and target), and final alignments are created via symmetrization heuristics (Och and Ney, 2000; Koehn et al., 2005); other approaches explicitly symmetrize during training (Matusov et al., 2004; Liang et al., 2006).[1] While these models rely on only position and word identity information, subword information can be integrated without requiring costly inference (Berg-Kirkpatrick et al., 2010), leading to better parameter estimation for rare words. Alignments can also be extracted from neural translation models (Chen et al., 2020; Zenkel et al., 2020).

Word alignment also enables annotation projection (Yarowsky and Ngai, 2001; Yarowsky et al., 2001) which can offer strong performance, particularly for low-resource languages (Buys and Botha, 2016; Ortega and Pillaipakkamnatt, 2018; Nicolai and Yarowsky, 2019; Nicolai et al., 2020; Eskander et al., 2020).

Multilingual Transformer Models Pretrained multilingual models (Devlin et al., 2019; Conneau and Lample, 2019; Conneau et al., 2020; Xue et al., 2021) have become the de facto standard approach for cross-lingual transfer. In general, these models are an extension of their monolingual variants, created by including data from many languages in their pretraining. They rely on a subword vocabulary (Kudo and Richardson, 2018) which jointly spans all of the pretraining languages. Models are pretrained using a masked language modeling (MLM) objective and a translation language modeling (TLM; Conneau and Lample, 2019) objective that uses parallel sentences. Outside of continued pretraining (Gururangan et al., 2020), models can be adapted using Adapters (Pfeiffer et al., 2020) or through vocabulary adaptation (Wang et al., 2020; Hong et al., 2021). Word alignment methods which depend on these models have also been proposed (Jalili Sabet et al., 2020; Nagata et al., 2020); we focus on AWESoME align (Dou and Neubig, 2021) because it outperforms other unsupervised methods.

3 Experiment 1: Intrinsic Evaluation

3.1 Experimental Setup

Languages We focus on four Indigenous languages spoken in the Americas for our experiments. **Bribri (bzd)** is a tonal language in the Chibchan family spoken by approximately 7000 people in Costa Rica. **Guarani (gn)** is a polysynthetic language in the Tupi–Guarani family spoken by around 6 million people across South America. **Quechua (quy)** is a family of Indigenous languages—from which we study Quechua Chanka—spoken across the Peruvian Andes by over 6 million people, and **Shipibo-Konibo (shp)** is a language spoken by around 30,000 people in Peru, Bolivia, and Brazil (Cardenas and Zeman, 2018). The latter three languages are agglutinative.

Training Data For training, we use the parallel data between Spanish and our languages described by Mager et al. (2021).[2] We note that there is a distinct difference in the amount of unlabeled data available within the four languages: Guarani and Quechua have considerably more data available. These two languages also have monolingual

---

[1] The poor estimation of rare words' translation parameters also motivates symmetrization; without this, rarely observed words become *garbage collector words* (Moore, 2004).

[2] Although parallel, digitized Bibles exist for over 1600 languages (McCarthy et al., 2020), groups may object to annotating the Bible for historical or cultural reasons.

text available in Wikipedia, which we extract using WikiExtractor (Attardi, 2015). The exact number of parallel and monolingual sentences for all languages is shown in Table B.1.

Evaluation Data To create gold standard alignments for evaluation, we sample multi-way parallel examples from AmericasNLI (Ebrahimi et al., 2022), allowing for multi-parallel alignments (Xia and Yarowsky, 2017) across all languages. Samples for the development and test sets are taken from their respective splits in the AmericasNLI dataset. Development examples were collected first, manually checked, and corrected. Examples with misalignments in punctuation, numbers, or named entities were not used. After a period of development with this data, the test set of 50 examples was collected and manually verified. Annotations were collected using JHU's open-source Turkle platform.[3] We ask annotators to only mark *sure* alignments. Additional discussion on data collection and the test set can be found in §6.

Metrics We evaluate automatic alignments via alignment error rate (AER; Och and Ney, 2000). Because we only collect sure alignments, this is equivalent to the balanced F-measure (Fraser and Marcu, 2007). We give additional metrics in Table C.3.

3.2 Models

Traditional Aligners We use Giza++ (Och and Ney, 2003) and FastAlign (Dyer et al., 2013) as our traditional aligners. Giza++ is based on IBM Models 1–5 (Brown et al., 1993). FastAlign (Dyer et al., 2013) is a re-parameterization of IBM Model 2. We use the implementation and hyperparameters of Zenkel et al. (2020), which relies on MGiza++ (Gao and Vogel, 2008) and the standard FastAlign package. Both approaches run on CPUs, and their training time ranges between 6 seconds to 3 minutes for FastAlign, and 43 seconds to 22 minutes for Giza++. We use the union of the forward and reverse alignments, as this symmetrization heuristic offers the best result for all languages on the development set. We show the performance of other heuristics in Table C.2.

Neural Aligners AWESoME (Dou and Neubig, 2021) identifies alignment links by considering cosine similarities between hidden layer representations of tokens in a neural encoder. We consider two such encoders: mBERT (Devlin et al., 2019) and XLM-R (Liu et al., 2019), and we use the default AWESoME configuration to extract alignments. We give layer-by-layer alignment performance in Figure C.1.

Model	Method	BZD	GN	QUY	SHP	AVG.
AWESoME	BL	70.03	63.13	67.02	60.41	65.15
(mBERT)	+MLM-T	68.95	49.68	46.59	58.17	55.85
	+MLM-ST	70.63	50.25	42.52	58.66	55.52
	+TLM	58.43	**43.10**	**36.96**	**52.34**	**47.71**
AWESoME	BL	80.15	73.11	75.24	69.21	74.43
(XLM-R)	+MLM-T	76.89	65.44	53.65	65.16	65.29
	+MLM-ST	77.53	64.55	52.90	66.56	65.39
	+TLM	74.90	58.84	43.25	63.48	60.12
FastAlign	Union	**51.40**	43.52	54.06	54.67	50.91
Giza++	Union	55.61	49.92	66.01	60.84	58.10
mBERT	+MLM-WT	-	***40.00***	46.00	-	43.00
XLM-R	+MLM-WT	-	52.27	48.83	-	50.55

Table 1: AER, in percentages, for each language and method. The best overall result for each language is bolded, while the best model within each method is underlined. We separate results which use Wikipedia, as they are not directly comparable.

Model Adaptation We experiment with three adaptation schemes based on continued pretraining (+TLM, +MLM-T, and +MLM-ST) which rely on unlabeled data and further train the model using MLM (Gururangan et al., 2020) before alignments are extracted. We focus on these objectives as they have been used by prior work for general model adaptation, and they work well in situations with limited resources (Ebrahimi and Kann, 2021). As we have access to bitext between Spanish and the target languages, for the +TLM scheme each example is the concatenation of a Spanish sentence with its translation. For +MLM-**T** we adapt using solely the **t**arget side of the available data, and for +MLM-**ST** we adapt on both the **s**ource and **t**arget; however, this data is treated as monolingual data and not explicitly aligned. +MLM-WT denotes target language adaptation which includes Wikipedia data. The duration of adaptation depends on the GPU and method used; it ranges from around 6 minutes for Bribri to 4 hours for Quechua. We provide additional training details in Appendix A.

3.3 Results

Traditional vs. Neural Aligners We present results in Table 1. The best traditional method is FastAlign, and the best neural approach is with mBERT+TLM. Comparing the two, we see that

---

[3] `https://github.com/hltcoe/turkle`

the lowest error rate is achieved with the neural approach for all languages except for Bribri, where FastAlign offers 7.03% absolute improvement. Of the other three languages, the performance for two is close: the difference in performance for Guarani is only 0.42% and 2.33% for Shipibo-Konibo. For Quechua, +TLM improves over FastAlign by 17.10%.

Comparing Adaptation Strategies With mBERT, +MLM-T improves performance over the non-adapted baseline by 9.30% on average, with +MLM-ST increasing this gain to 9.63% and +TLM offering the highest improvement of 17.44%, consistent with prior work on *seen* languages (Dou and Neubig, 2021). Per language, the largest and smallest gains are for Quechua (30.06%) and for Shipibo-Konibo (8.07%); intuitively, gains from adaptation are proportional to the size of the adaptation data. For XLM-R, we again see relative gains from adaptation, with +TLM offering the highest performance increase.

Additional Monolingual Data Neural approaches can easily benefit from additional monolingual data. Adding Wikipedia data results in the highest performance for Guarani, outperforming the previous best approach by 3.1%. In contrast, while the additional data for Quechua does help relative to +MLM-T, it does not outperform +TLM. This difference in performance may be due to the relative sizes of the additional data; the Guarani Wikipedia has 1.3× as many tokens as the target-side parallel data, while the Quechua Wikipedia only has 0.5× as many.

4 Experiment 2: Extrinsic Evaluation

We further compare aligner performance extrinsically by evaluating downstream task performance when using a projected training set. We consider two tasks: NER and POS tagging.

4.1 Experimental Setup

Data Due to the limited availability and quality of evaluation datasets, we focus on Guarani for this experiment. We use the test set provided by Rahimi et al. (2019) for NER and Universal Dependencies (Nivre et al., 2020) for POS. For experiments where we finetune directly on English or Spanish, we use the provided training data.

Model	Train Source	POS	NER
mBERT	en	10.36	46.64
	es	19.82	49.18
+TLM	es	36.94	49.62
+MLM-T	es	34.69	**55.25**
+MLM-ST	es	33.78	52.34
mBERT	mBERT	31.53	47.54
	+MLM-T	38.29	47.97
	+MLM-ST	**42.34**	49.80
	+TLM	40.99	49.80
	FastAlign	37.84	46.55
	Giza++	39.19	48.33

Table 2: POS tagging (accuracy) and NER results (F1) for Guarani. *Model* denotes if baseline or adapted mBERT is used. *Train Source* defines the training data used for finetuning; language codes indicate training on original data, while alignment methods denote how a projected training set was created.

Annotation Projection To create the projected training sets, we first annotate the (unlabeled) Spanish parallel data with Stanza (Qi et al., 2020) and generate bidirectional alignments using each method. We then project the tags from Spanish to Guarani using type and token constraints as described by Buys and Botha (2016).

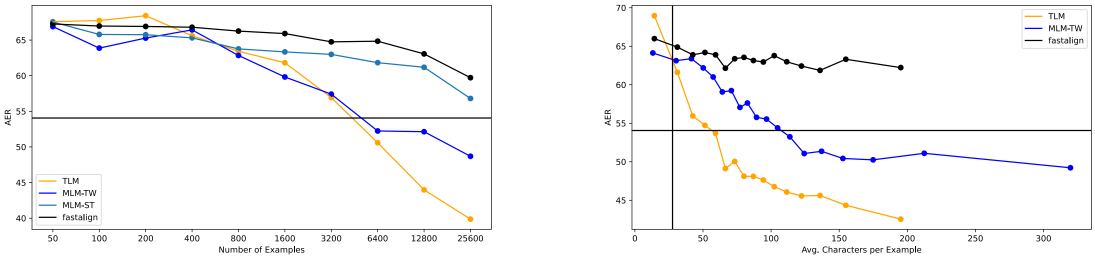
Models For baseline performance, we finetune mBERT on the provided English and Spanish training sets for each task. Additionally, we also finetune adapted versions of mBERT on Spanish training data – English is omitted as performance is worse and adaptation data is in Spanish. Finally, we evaluate performance when finetuning mBERT on the training sets created through projection.

4.2 Results

We present results for both tasks in Table 2.

POS For POS tagging, the baseline zero-shot performance is extremely poor, and we see a minimum increase of 11.71% accuracy when using any projection method. Giza++ outperforms FastAlign, as well as projection with +MLM-T, however the best performance is achieved with +MLM-ST, with +TLM offering the second best result. While the ordering of methods changes, the best performance is still achieved with the neural approaches, consistent with the results of Experiment 1.

NER For NER, baseline performance is high: inspecting the data shows that many entities have English or Spanish names, and as multilingual models already have knowledge of these two languages,



(a) Subset Analysis

(b) Length Analysis

Figure 2: Plots for data analysis. In Figure 2b, a vertical line denotes the average example length for Bribri.

standard aligners with projection may not effectively leverage surface word-form clues. However, they remain a valuable indication of alignment quality. Among the projection-based approaches, we find that using Giza++ again outperforms +MLM-T and FastAlign but falls short of +MLM-ST and +TLM.

Overall, considering what both downstream tasks indicate regarding alignment quality, neural models adapted using Spanish and target-language data—either sentence-aligned or unaligned—consistently outperform traditional methods.

5 Analysis

As data for low-resource languages often varies considerably in both amount and length, we consider two additional analysis experiments which control for these factors. We focus solely on Quechua, as it has the most parallel data available. Results are presented in Figure 2 with numerical results in Tables C.4 and C.5.

Subset Analysis For this analysis, we ask how the performance of neural alignment depends on the amount of data and with how much data it surpasses traditional approaches. We subsample the adaptation data, and use this to extract alignments using both FastAlign and AWESoME. Results for this experiment can be seen in Figure 2a. For reference, we also plot the AER obtained when using FastAlign on all the available training data as an upper bound for the performance of the traditional approaches. In the smallest extreme, all methods are roughly equivalent. However, as the number of examples increases, adaptation using +TLM and +MLM-WT improves at a faster rate than other approaches: with only 6400 sentence pairs, these approaches overtake the best expected performance of FastAlign.

Length Analysis Aligner performance may not only be affected by the total number of examples available, but also by the length of these examples. This is doubly relevant for low-resource languages, as resources may be limited to sources which do not contain long (or even complete) sentences. To see how the performance of each method may vary when faced with examples of different lengths, we sort the unlabeled data by the number of characters, and partition the examples in groups of 7508, the total number of examples available for Bribri. We choose this amount as it is representative how much data may be available for other low-resource languages. As before, the expected upper bound FastAlign performance is denoted. For the shortest group, all methods are similar; however, AWESoME alignments improve with longer sequences, with +TLM showing the quickest decrease in error rate. We attribute the improved AER when adapting using longer sequences to the increased number of tokens available for adaptation. For Quechua, the performance of AWESoME align is sensitive to both the number of examples and sequence length. In contrast, FastAlign only shows a small improvement as example length increases.

6 Conclusion

In this work, we have investigated the performance of modern word aligners versus classical approaches for languages *unseen* to pretrained models. While classical methods remain competitive, the lowest AER on average is achieved by modern neural approaches. However, using these models comes with a larger computational cost. Therefore, the trade-off between training requirements and overall performance must be considered. If access to computing resources is limited or training time is a factor, classical approaches remain a viable approach which should not be discounted.

Ethics and Limitations

Ethics Statement

When collecting data in an Indigenous language, it becomes vital that the process does not exploit any member of the community or commodify the language (Schwartz, 2022). Further, it is important that members of the community benefit from the dataset. While the creation of a word alignment dataset will not directly impact community members, we believe that it can contribute to the development of tools, such as translation systems, that can be directly beneficial, and that increasing the visibility of these languages within the research community will further spur the creation of useful systems. Our annotations were created by either co-authors of the paper or by native speakers of the languages, who were compensated at a rate chosen with the minimum hourly salary in their respective countries taken into account.

Limitations

Test Set Size One limitation of our work is the size of the evaluation set used for our main results. This arises from the general difficulty in collecting annotations and data for low-resource, and particularly Indigenous languages. The size of the test set was chosen to balance the trade-off between the cost of annotation collection and experimental validity. Fortunately, for the task of word alignment the main metric used to summarize performance—alignment error rate—does not depend directly on the number of examples in the evaluation set, but the total number of alignments, of which there are a sufficiently high number in our evaluation set. However, even when only considering the number of examples, our test set is still within the same order of magnitude as other widely used word alignment evaluation sets, such as the Romanian–English test set which consists of 248 examples (Mihalcea and Pedersen, 2003), and the English–Inuktitut and English–Hindi test sets which have 75 and 90 examples each, respectively (Martin et al., 2005).

We run a small experiment to gain insight into how much precision is lost when using a test set of size 50, versus 248, which we choose as this is the size of the widely used Romanian–English test set mentioned above. We take 100 independent samples without replacement from the Romanian-English test set, each of size 50, and evaluate the performance of FastAlign and AWESoME align. For FastAlign, we use the training data defined by Mihalcea and Pedersen (2003), and for Awesome, we use mBERT with no additional finetuning. The distributions of AER are shown in Figure A.1, with summary statistics in Table A.1. We can see that the standard deviation of both distributions is relatively low, around 2%. At the extremes, we see a difference of -4.70% and $+4.90\%$, and -4.28% and $+6.4\%$ for FastAlign and AWESoME align respectively, between the min/max values of our distribution as compared to the whole set AER. Considering these points, we believe that the size of our evaluation set does not invalidate our experimental results and main conclusions; however, we note that additional care must be taken when comparing specific models whose performances are close together, particularly when this performance is low or close to random.

Test Set Domain Other limitations of our work arise from the sources of data used. Annotations were done using sentences sampled from AmericasNLI, which itself is a translation of XNLI. As such, any errors from the original XNLI dataset, which may have propagated through translation, will persist in our dataset as well (annotators were given the option to modify target language sentences to correct any errors). Furthermore, due to translation, the sentences may not be directly representative of a natural utterance which would be spoken by members of the communities.

Language Selection The languages we highlight in this work are true low-resource languages, and present challenges commonly faced by other low-resource languages. Namely, these languages have a relatively small amount of easily available and clean unlabeled data, are typically unseen from most released pretrained models, and are morphologically different from typically used source languages. However, one feature of these languages which may inflate aligner performance is the language script: all of our target languages share the same script with the two source languages which we use. This may lead to higher occurrences of shared words or entities, making alignment easier. As such, our results may not generalize fully to other low-resource languages which have a different script from the source languages, or which may have a script which is unseen to the underlying pretrained model.

Acknowledgements

We would like to thank Roque Helmer Luna-Montoya (Academia Mayor de la Lengua Quechua in Cuzco, Perú) and Richard Castro Mamani (Universidad Nacional de San Antonio Abad and Hinantin Software) for their help in annotating and verifying the Quechua–Spanish alignments. We would also like to thank Liz Karen Chavez Sanchez for annotating the Shipibo-Konibo–Spanish alignments. A.D.M. is supported by an Amazon Fellowship and a Frederick Jelinek Fellowship.

References

Giusepppe Attardi. 2015. Wikiextractor. `https://github.com/attardi/wikiextractor`.

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590, Los Angeles, California. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Jan Buys and Jan A. Botha. 2016. Cross-lingual morphological tagging for low-resource languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1954–1964, Berlin, Germany. Association for Computational Linguistics.

Ronald Cardenas and Daniel Zeman. 2018. A morphological analyzer for Shipibo-konibo. In *Proceedings of the Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 131–139, Brussels, Belgium. Association for Computational Linguistics.

Ethan C. Chau, Lucy H. Lin, and Noah A. Smith. 2020. Parsing with multilingual BERT, a small corpus, and a small treebank. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1324–1334, Online. Association for Computational Linguistics.

Yun Chen, Yang Liu, Guanhua Chen, Xin Jiang, and Qun Liu. 2020. Accurate word alignment induction from neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 566–576, Online. Association for Computational Linguistics.

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

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

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

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Kevin Duh, Paul McNamee, Matt Post, and Brian Thompson. 2020. Benchmarking neural and statistical machine translation on low-resource African languages. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2667–2675, Marseille, France. European Language Resources Association.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Abteen Ebrahimi and Katharina Kann. 2021. How to adapt your pretrained multilingual model to 1600 languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4555–4567, Online. Association for Computational Linguistics.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir Meza Ruiz, Gustavo Giménez-Lugo, Elisabeth Mager, Graham Neubig, Alexis Palmer, Rolando Coto-Solano, Thang Vu, and Katharina Kann. 2022. AmericasNLI: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. In

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6279–6299, Dublin, Ireland. Association for Computational Linguistics.

Ramy Eskander, Smaranda Muresan, and Michael Collins. 2020. Unsupervised cross-lingual part-of-speech tagging for truly low-resource scenarios. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4820–4831, Online. Association for Computational Linguistics.

Alexander Fraser and Daniel Marcu. 2007. Squibs and discussions: Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57, Columbus, Ohio. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Jimin Hong, TaeHee Kim, Hyesu Lim, and Jaegul Choo. 2021. AVocaDo: Strategy for adapting vocabulary to downstream domain. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4692–4700, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the Second International Workshop on Spoken Language Translation*, Pittsburgh, Pennsylvania, USA.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

En-Shiun Lee, Sarubi Thillainathan, Shravan Nayak, Surangika Ranathunga, David Adelani, Ruisi Su, and Arya McCarthy. 2022. Pre-trained multilingual sequence-to-sequence models: A hope for low-resource language translation? In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 58–67, Dublin, Ireland. Association for Computational Linguistics.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, page 104–111, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Manuel Mager, Arturo Oncevay, Abteen Ebrahimi, John Ortega, Annette Rios, Angela Fan, Ximena Gutierrez-Vasques, Luis Chiruzzo, Gustavo Giménez-Lugo, Ricardo Ramos, Ivan Vladimir Meza Ruiz, Rolando Coto-Solano, Alexis Palmer, Elisabeth Mager-Hois, Vishrav Chaudhary, Graham Neubig, Ngoc Thang Vu, and Katharina Kann. 2021. Findings of the AmericasNLP 2021 shared task on open machine translation for indigenous languages of the Americas. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 202–217, Online. Association for Computational Linguistics.

Joel Martin, Rada Mihalcea, and Ted Pedersen. 2005. Word alignment for languages with scarce resources. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 65–74, Ann Arbor, Michigan. Association for Computational Linguistics.

Evgeny Matusov, Richard Zens, and Hermann Ney. 2004. Symmetric word alignments for statistical machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 219–225, Geneva, Switzerland. COLING.

Arya D. McCarthy, Rachel Wicks, Dylan Lewis, Aaron Mueller, Winston Wu, Oliver Adams, Garrett Nicolai, Matt Post, and David Yarowsky. 2020. The Johns Hopkins University Bible corpus: 1600+ tongues for typological exploration. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2884–2892, Marseille, France. European Language Resources Association.

Rada Mihalcea and Ted Pedersen. 2003. An evaluation exercise for word alignment. In *Proceedings of the HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 1–10.

Robert C. Moore. 2004. Improving IBM word alignment model 1. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 518–525, Barcelona, Spain.

Benjamin Muller, Antonios Anastasopoulos, Benoît Sagot, and Djamé Seddah. 2021. When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 448–462, Online. Association for Computational Linguistics.

Masaaki Nagata, Katsuki Chousa, and Masaaki Nishino. 2020. A supervised word alignment method based on cross-language span prediction using multilingual BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 555–565, Online. Association for Computational Linguistics.

Garrett Nicolai, Dylan Lewis, Arya D. McCarthy, Aaron Mueller, Winston Wu, and David Yarowsky. 2020. Fine-grained morphosyntactic analysis and generation tools for more than one thousand languages. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3963–3972, Marseille, France. European Language Resources Association.

Garrett Nicolai and David Yarowsky. 2019. Learning morphosyntactic analyzers from the Bible via iterative annotation projection across 26 languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1765–1774, Florence, Italy. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

John Ortega and Krishnan Pillaipakkamnatt. 2018. Using morphemes from agglutinative languages like Quechua and Finnish to aid in low-resource translation. In *Proceedings of the AMTA 2018 Workshop on Technologies for MT of Low Resource Languages (LoResMT 2018)*, pages 1–11, Boston, MA. Association for Machine Translation in the Americas.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Lane Schwartz. 2022. Primum Non Nocere: Before working with Indigenous data, the ACL must confront ongoing colonialism. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 724–731, Dublin, Ireland. Association for Computational Linguistics.

David A. Smith and Noah A. Smith. 2004. Bilingual parsing with factored estimation: Using English to parse Korean. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 49–56, Barcelona, Spain. Association for Computational Linguistics.

Zihan Wang, Karthikeyan K, Stephen Mayhew, and Dan Roth. 2020. Extending multilingual BERT to low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2649–2656, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Patrick Xia and David Yarowsky. 2017. Deriving consensus for multi-parallel corpora: an English Bible study. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 448–453, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the First International Conference on Human Language Technology Research*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. End-to-end neural word alignment outperforms GIZA++. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1605–1617, Online. Association for Computational Linguistics.

A Training Details and Hyperparameters

We compare two data loading strategies for adaptation: a naïve approach where each example in the dataset represents an example in the loaded training examples, and a packing strategy following the `FULL-SENTENCES` approach of Liu et al. (2019). We use the hyperparameters described by Ebrahimi et al. (2022) – a learning rate of 2e-5, batch size of 32, and warmup ratio of 1% – however due to the different loading strategy we tune the total amount of training time. We experiment with 40 and 80 epochs of training, using the alignment development set to select the final hyperparameters. For both MLM-T and MLM-ST we find that packing sequences yields better results, however for +TLM we use the naïve strategy to preserve sentence alignment. We use packing by default for Wikipedia data, due to the length of extracted documents. For all adaptation methods we find that training for 80 epochs is best, except for +MLM-ST, which we train for 40. We train with 1 Nvidia A100 or 2 V100 GPUs. Due to the computational cost associated with pretraining, we only conduct one model run for each language and method. We pretrain our models using Huggingface (Wolf et al., 2020).

Training Time As mentioned in Section 3.2, for adaptation the training duration depends on the GPU and method used, with times ranging from around 6 minutes for Bribri to 4 hours for Quechua. For the statistical approaches, both run solely on CPUs, and their training time ranges between 6 seconds to 3 minutes for FastAlign, and 43 seconds to 22 minutes for Giza++. However, GPU availability is not always certain – to roughly compare training times given a more restricted setting, we run our adaptation experiments without access to any GPUs, and compute an estimate for the total training time using only CPUs as approximately 2 weeks.

	Whole Set AER	Avg. AER	AER Std.	Min AER	25%	50%	75%	Max AER
FastAlign	35.00	35.09	1.94	30.30	33.70	35.00	36.23	39.90
Awesome	28.23	28.26	2.04	23.95	26.66	28.12	29.71	34.63

Table A.1: Summary statistics for subsample AER distribution.

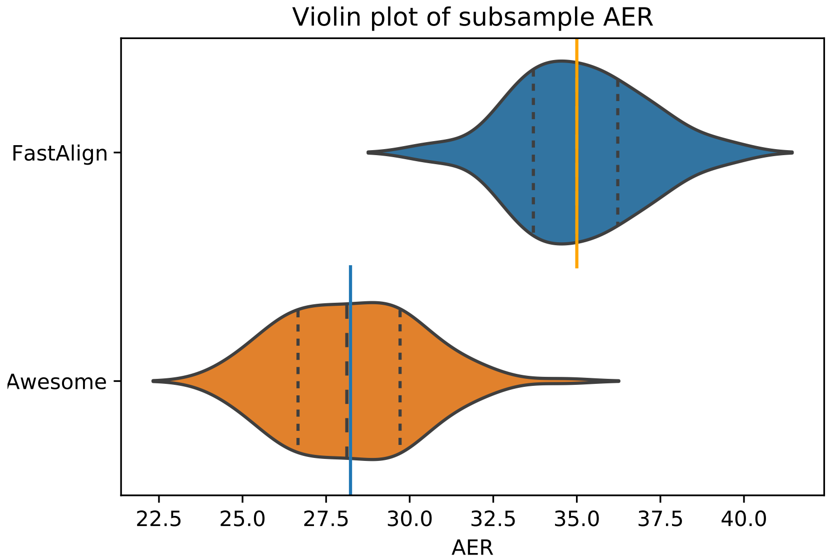


Figure A.1: Distribution of AER when using FastAlign and AWESoME align to evaluate subsets of size 50 taken from a complete evaluation set of size 248. Quartiles are displayed using dashed lines, while inverted colors represent the AER calculated when evaluating on the complete set.

B Dataset Features

Feature	BZD	GN	QUY	SHP
Number of examples (Parallel)	7,508	26,032	121,064	14,592
Number of examples (Wiki)	-	4721	22610	-
Number of tokens - MLM-T	123,992	1,104,645	3,912,582	179,451
Number of tokens - TLM	194,798	2,006,996	6,697,771	328,427
Number of tokens - Wiki	-	1,460,240	2,023,297	-
Number of dev examples	50	48	45	46
Number of test examples	50	50	50	50

Table B.1: Features of the data used for our experiments.

C Supplementary Results

Model	Method	BZD	GN	QUY	SHP
AWESoME	BL	65.38	58.51	63.98	62.80
+mBERT	+MLM-T	64.26	43.29	39.10	66.44
	+MLM-ST	65.43	43.46	37.20	65.63
	+TLM	54.25	34.62	30.38	62.10
AWESoME	BL	76.29	71.85	71.53	73.96
+XLM-R	+MLM-T	72.73	57.50	43.30	69.25
	+MLM-ST	73.08	60.28	44.88	70.48
	+TLM	71.88	49.76	36.11	69.23
FastAlign	Union	47.39	39.78	58.37	57.91
Giza++	Union	51.03	62.07	47.18	64.98

Table C.1: Development AER for each language and method.

Method	Heuristic	BZD	GN	QUY	SHP
FastAlign	grow-diagonal-final	54.56	49.64	60.51	56.11
	grow-diagonal	55.36	50.41	63.81	56.87
	intersection	57.11	52.89	66.92	61.67
	union	51.40	43.52	54.06	54.67
	reverse	52.21	51.51	61.27	58.41
Giza++	grow-diagonal-final	55.51	53.38	75.29	62.72
	grow-diagonal	59.33	58.41	80.06	69.53
	intersection	63.71	64.95	82.55	77.41
	union	55.61	49.92	66.01	60.84
	reverse	56.43	62.20	76.05	72.39

Table C.2: AER results on the test set for various growing heuristics.

		BZD			GN			QUY			SHP		
Model	Method	P	R	F	P	R	F	P	R	F	P	R	F
AWESoME (mBERT)	BL	41.8	23.4	30.0	50.6	29.0	36.9	49.4	24.7	33.0	64.0	28.7	39.6
	+MLM-T	42.7	24.4	31.1	68.6	39.7	50.3	69.1	43.5	53.4	66.0	30.6	41.8
	+MLM-ST	42.9	22.3	29.4	67.2	39.5	49.8	73.8	47.1	57.5	67.6	29.8	41.3
	+TLM	62.1	31.2	41.6	76.3	45.4	56.9	79.0	52.5	63.0	79.4	34.0	47.7
AWESoME (XLM-R)	BL	48.0	12.5	19.9	48.4	18.6	26.9	49.7	16.5	24.8	64.5	20.2	30.8
	+MLM-T	38.9	16.4	23.1	63.0	23.8	34.6	70.2	34.6	46.4	57.2	25.0	34.8
	+MLM-ST	40.8	15.5	22.5	65.7	24.3	35.4	74.8	34.4	47.1	56.5	23.7	33.4
	+TLM	50.0	16.8	25.1	76.9	28.1	41.2	83.2	43.1	56.8	77.0	23.9	36.5
FastAlign	Union	46.4	51.0	48.6	55.4	57.6	56.5	44.3	47.7	45.9	48.0	43.0	45.3
Giza++	Union	39.9	49.8	44.3	48.3	52.0	50.1	32.0	36.3	34.0	37.2	41.4	39.2
mBERT	+MLM-WT	-	-	-	76.3	49.4	60.0	70.8	43.6	54.0	-	-	-
XLM-R	+MLM-WT	-	-	-	66.4	31.5	42.7	75.0	38.8	51.2	-	-	-

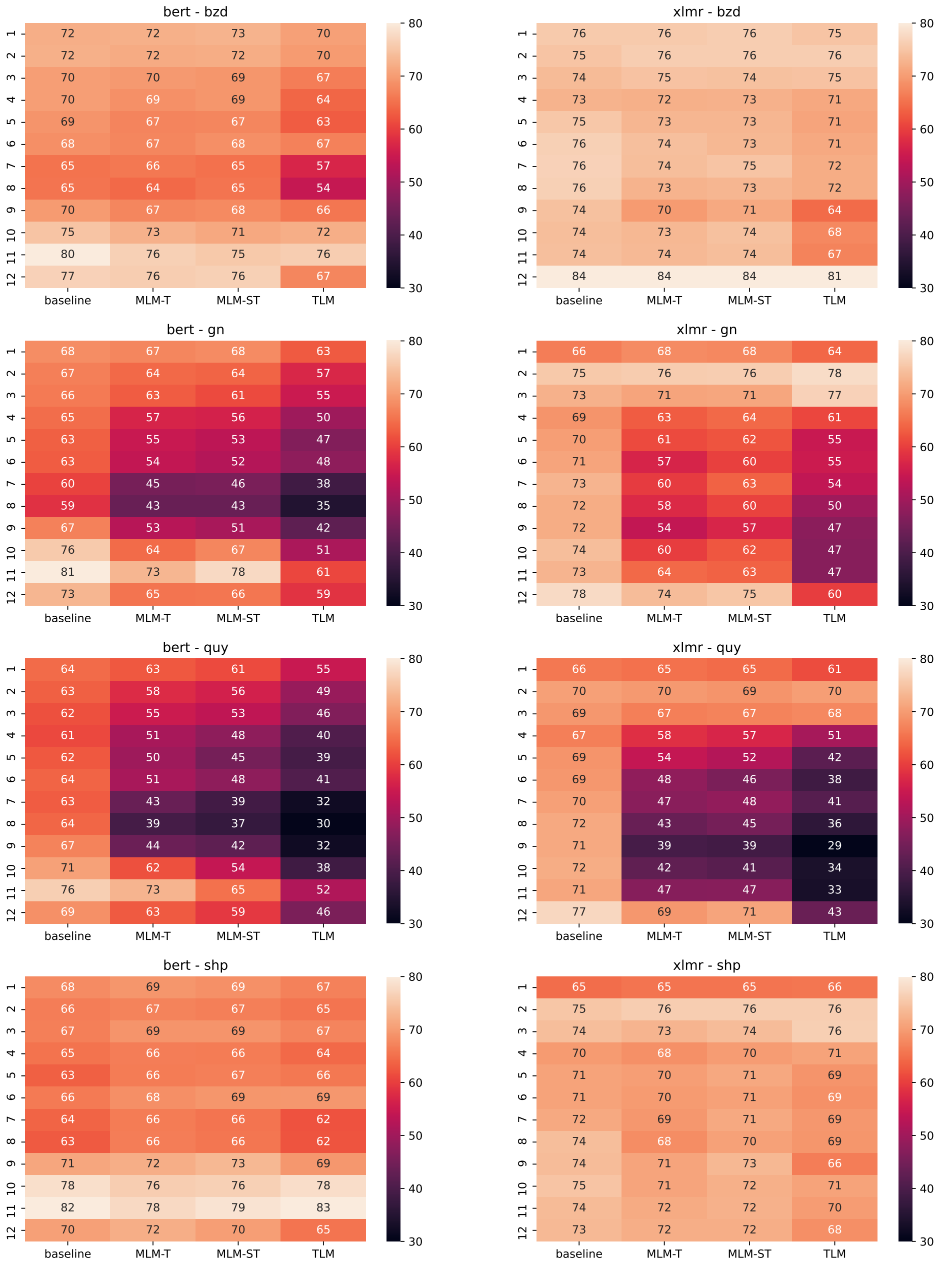
Table C.3: Precision, recall, and F-measure for main test set results. All metrics are on a 0–100 scale (larger is better).

Num. Examples	+TLM	+MLM-WT	+MLM-ST	FastAlign
50	67.58	66.89	67.53	67.26
100	67.74	63.87	65.79	66.97
200	68.42	65.28	65.75	66.91
400	65.61	66.43	65.31	66.80
800	63.43	62.84	63.76	66.26
1600	61.81	59.82	63.34	65.91
3200	56.93	57.41	62.99	64.75
6400	50.59	52.24	61.83	64.84
12800	43.98	52.14	61.18	63.06
25600	39.87	48.69	56.80	59.72

Table C.4: AER for each method and subset used in the Subset Analysis.

+MLM-WT		+TLM		FastAlign	
Avg. Char	AER	Avg. Char	AER	Avg. Char	AER
13.20	64.13	14.31	68.97	14.31	65.99
30.29	63.13	31.20	61.61	31.20	64.89
41.49	63.39	42.51	55.95	42.51	63.89
50.19	62.19	51.45	54.73	51.45	64.19
57.47	61.01	59.23	53.70	59.23	63.89
64.20	59.07	66.44	49.12	66.44	62.15
70.83	59.24	73.30	50.04	73.30	63.38
77.12	57.06	80.09	48.12	80.09	63.56
82.63	57.63	87.02	48.10	87.02	63.15
89.31	55.77	94.31	47.63	94.31	62.96
96.66	55.54	102.30	46.76	102.30	63.78
104.76	54.40	111.48	46.07	111.48	62.99
113.76	53.24	122.29	45.56	122.29	62.43
124.33	51.07	135.93	45.62	135.93	61.87
137.03	51.36	154.86	44.35	154.86	63.31
152.70	50.43	195.18	42.55	195.18	62.23
174.88	50.25	-	-	-	-
212.44	51.10	-	-	-	-
319.76	49.22	-	-	-	-

Table C.5: AER for each method and length group used in the Length Analysis. Average Chars represents the average number of characters per example, for each group.



bert - bzd

	baseline	MLM-T	MLM-ST	TLM
1	72	72	73	70
2	72	72	72	70
3	70	70	69	67
4	70	69	69	64
5	69	67	67	63
6	68	67	68	67
7	65	66	65	57
8	65	64	65	54
9	70	67	68	66
10	75	73	71	72
11	80	76	75	76
12	77	76	76	67

xlmr - bzd

	baseline	MLM-T	MLM-ST	TLM
1	76	76	76	75
2	75	76	76	76
3	74	75	74	75
4	73	72	73	71
5	75	73	73	71
6	76	74	73	71
7	76	74	75	72
8	76	73	73	72
9	74	70	71	64
10	74	73	74	68
11	74	74	74	67
12	84	84	84	81

bert - gn

	baseline	MLM-T	MLM-ST	TLM
1	68	67	68	63
2	67	64	64	57
3	66	63	61	55
4	65	57	56	50
5	63	55	53	47
6	63	54	52	48
7	60	45	46	38
8	59	43	43	35
9	67	53	51	42
10	76	64	67	51
11	81	73	78	61
12	73	65	66	59

xlmr - gn

	baseline	MLM-T	MLM-ST	TLM
1	66	68	68	64
2	75	76	76	78
3	73	71	71	77
4	69	63	64	61
5	70	61	62	55
6	71	57	60	55
7	73	60	63	54
8	72	58	60	50
9	72	54	57	47
10	74	60	62	47
11	73	64	63	47
12	78	74	75	60

bert - quy

	baseline	MLM-T	MLM-ST	TLM
1	64	63	61	55
2	63	58	56	49
3	62	55	53	46
4	61	51	48	40
5	62	50	45	39
6	64	51	48	41
7	63	43	39	32
8	64	39	37	30
9	67	44	42	32
10	71	62	54	38
11	76	73	65	52
12	69	63	59	46

xlmr - quy

	baseline	MLM-T	MLM-ST	TLM
1	66	65	65	61
2	70	70	69	70
3	69	67	67	68
4	67	58	57	51
5	69	54	52	42
6	69	48	46	38
7	70	47	48	41
8	72	43	45	36
9	71	39	39	29
10	72	42	41	34
11	71	47	47	33
12	77	69	71	43

bert - shp

	baseline	MLM-T	MLM-ST	TLM
1	68	69	69	67
2	66	67	67	65
3	67	69	69	67
4	65	66	66	64
5	63	66	67	66
6	66	68	69	69
7	64	66	66	62
8	63	66	66	62
9	71	72	73	69
10	78	76	76	78
11	82	78	79	83
12	70	72	70	65

xlmr - shp

	baseline	MLM-T	MLM-ST	TLM
1	65	65	65	66
2	75	76	76	76
3	74	73	74	76
4	70	68	70	71
5	71	70	71	69
6	71	70	71	69
7	72	69	71	69
8	74	68	70	69
9	74	71	73	66
10	75	71	72	71
11	74	72	72	70
12	73	72	72	68

Figure C.1: AER using the development set, per layer, per language, for both mBERT and XLM-R.

Author Index